## Functional overview

- High-performance low-power 8-bit LGT8XM core
- Advanced RISC architecture
  - 131 instructions, more than 80% of the implementation of a single cycle
  - 32x8 universal working register
  - 32MHz work up to 32MIPS implementation efficiency
  - Internal single cycle multiplier (8x8)

Nonvolatile program and data storage space
  - 32Kbytes chip can be programmed online FLASH program memory
  - 2Kbytes internal data SRAM
  - Programmable E2PROM analog interface for byte access
  - A new program encryption algorithm to ensure user code security
- Peripheral controller
  - Two 8-bit timers with independent prescaler support compare output mode
  - Two 16-bit timers with independent prescaler support input capture and compare output
  - The internal 32KHz calibrates the RC oscillator to implement the real-time counter function
  - Supports up to 9 PWM outputs, 4 programmable deadband controls
  - 12-channel 12-bit high-speed analog-to-digital converter (ADC)
    - optional internal, external reference voltage
    - Programmable gain (X1 / 8/16/32) Differential amplification input channel
    - 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% calibrable reference voltage source
  - An 8-bit programmable DAC that can be used to generate a reference voltage source
  - Programmable Watchdog Timer (WDT)
  - Programmable Synchronous / Asynchronous Serial Interface (USART / SPI)
  - Synchronous peripheral interface (SPI), programmable master / slave operating mode
  - Two-wire serial interface (TWI), compatible with I2C master-slave mode
  - 16-bit digital arithmetic acceleration unit (DSC) that supports direct 16-bit data access access
- Special processor function
  - SWD Lens 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% can be calibrated 32MHz RC oscillator, support multiplier output
  - Built-in 1% calibrable 32KHz RC oscillator
  - External support 32.768KHz and 400K ~ 32MHz crystal input
  - 6x high current push-pull drive IO, support high-speed PWM applications

## 8-bit LGT8XM

RISC Microcontroller with
In-System Programmable
FLASH Memory

# LGT8F88P
# LGT8F168P
# LGT8F328P

Data book
Version 1.0.1

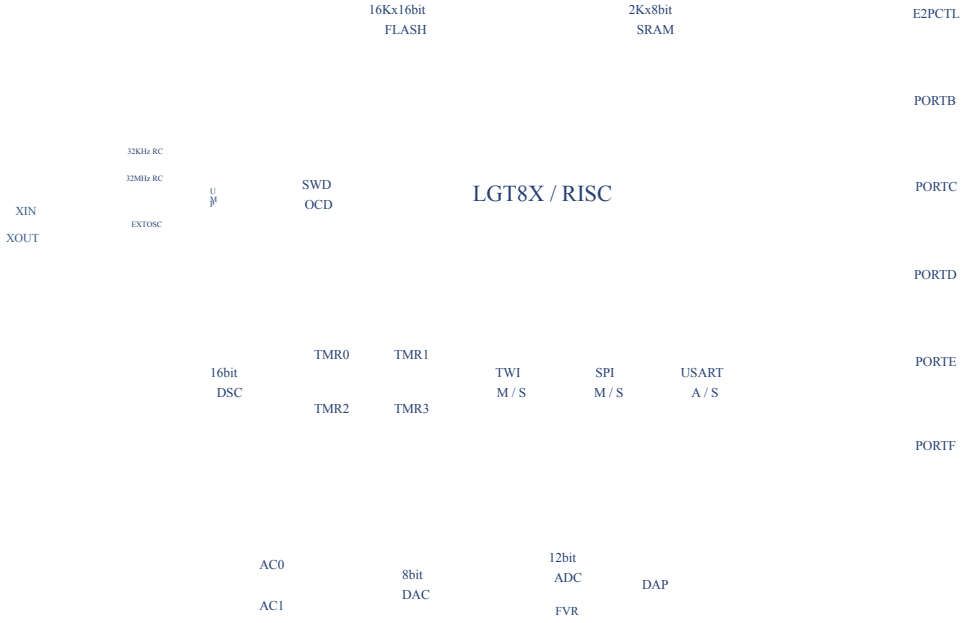
Application areas
Home appliances
Motor drive
Automatic control

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

**Page 2**



## system framework



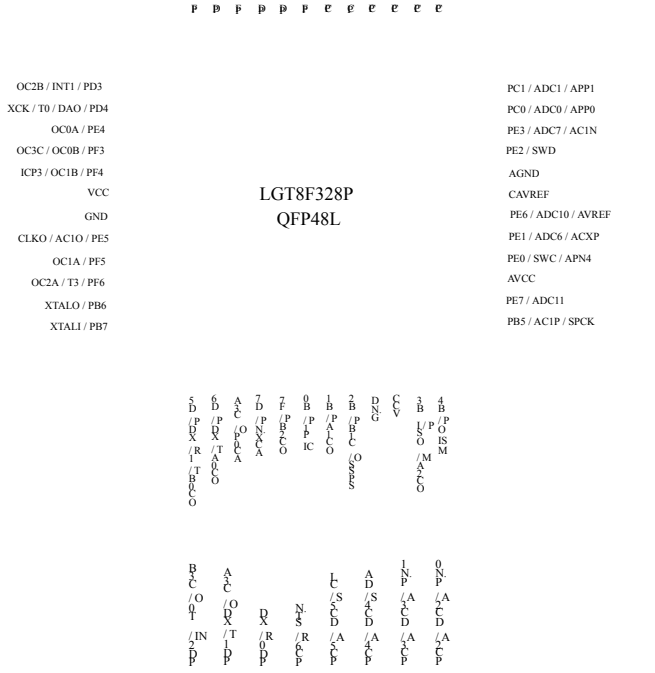

| Module name | Module function |
|---|---|
| SWD | Debugging module, while achieving online debugging and ISP functions |
| LGT8X | 8bit high performance RISC kernel |
| E2PCTL | Data FLASH access interface controller |
| PMU | Power management module, responsible for managing the conversion between the working state of the system |
| PORTB / C / D / E / F | Universal programmable input and output ports |
| DSC | 16-bit digital arithmetic acceleration unit |
| ADC | 8-channel 12-bit analog-to-digital converter |
| DAP | Programmable Gain Differential Amplifier |
| IVREF | 1.024V / 2.048V / 4.096V internal reference |
| AC0 / 1 | Analog comparator |
| TMR0 / 1/2/3 | 8/16-bit timer / event 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 |



**Page 3**



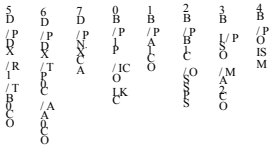
## Package definition













**Page 4**



## Pin description

In the LGT8FX8P family of packages, the QFP48L package leads all pins. Other packages are bundled with multiple internal I / O on a QFP48 basis Pin generated on the pin. Special attention should be paid when configuring pin orientation. The following table lists the bindings for the various package pins:

| QFP48 | QFP32 | Function Description |
|---|---|---|
| 01 | 01 | PD3 / INT1 / OC2B *<br>PD3: Programmable port D3<br>INT1: External interrupt input 1<br>OC2B: Timer 2 compare match output B |
| 02 | 02 | PD4 / DAO / T0 / XCK<br>PD4: Programmable port D4<br>DAO: Internal DAC output<br>T0: Timer0 external clock input |

| | | |
|---|---|---|
| | | XCK: USART Synchronous transfer clock |
| | | PE4 / 0C0A * |
| 03 | 03 | PE4: Programmable port E4 |
| | | OC0A: Timer 0 compare match output A |
| | | PF3 / OC3C / OC0B * |
| 04 | - | PF3: Programmable port F3 |
| | | OC3C: Timer 3 Compare Match Output C |
| | | OC0B: Timer 0 compare match output B |
| | | PF4 / OC1B * / ICP3 |
| 05 | 03 | PF4: Programmable port F4 |
| | | OC1B: Timer 1 Compare Match Output B |
| | | ICP3: Timer 3 capture input |
| 06 | 04 | VCC |
| 07 | 05 | GND |
| | | PE5 / AC1O / CLKO * |
| 08 | | PE5: Programmable port E5 |
| | | C1O: Analog comparator AC1 output |
| | 06 | CLKO: System clock output |
| | | PF5 / OC1A * |
| 09 | | PF5: Programmable port F5 |
| | | OC1A: Timer 1 compare match output A |
| | | PF6 / T3 / OC2A * |
| 10 | - | PF6: Programmable port F6 |
| | | T3: Timer 3 external clock input |
| | | OC2A: Timer 2 compare match output A |
| | | PB6 / XTALO |
| 11 | 07 | PB6: Programmable port B6 |
| | | XTALO: Crystal IO output port |





| | | |
|---|---|---|
| | | PB7 / XTALI |
| 12 | 08 | PB7: Programmable port B7 |
| | | XTALI: Crystal IO input port |
| | | PD5 / RXD * / T1 / OC0B |
| | | PD5: Programmable port D5 |
| 13 | 09 | RXD: USART data reception (optional) |
| | | T1: Timer 1 external clock input |
| | | OC0B: Timer 0 compare match output B |
| | | PD6 / TXD * / OC0A |
| 14 | | PD6: Programmable port D6 |
| | | TXD: USART data sent (optional) |
| | 10 | OC0A: Timer 0 compare match output A |
| | | AC0P / 0C3A |
| 15 | | AC0P: Analog Comparator 0 Positive input |
| | | OC3A: Timer 3 Compare Match Output A |
| | | PD7 / ACXN |
| 16 | 11 | PD7: Programmable port D7 |
| | | ACXN: Analog Comparator 0/1 Common Negative Input |
| | | PF7 / OC2B |
| 17 | - | PF7: Programmable port F7 |
| | | OC2B: Timer 2 compare match output B |
| | | PB0 / ICP1 |
| 18 | 12 | PB0: Programmable port B0 |

| | | |
|---|---|---|
| | | ICP1: Timer 1 capture input |
| | | PB1 / OC1A |
| 19 | 13 | PB1: Programmable port B1 |
| | | OC1A: Timer 1 compare match output A |
| | | PB2 / OC1B / SPSS |
| 20 | 14 | PB2: Programmable port B2 |
| | | OC1B: Timer 1 Compare Match Output B |
| | | SPSS: SPI Slave Mode Chip Select |
| twenty one | - | GND |
| twenty two | - | VCC |
| | | PB3 / MOSI / OC2A |
| twenty three | 15 | PB3: Programmable port B3 |
| | | MOSI: SPI master output / slave input |
| | | OC2A: Timer 2 compare match output A |
| | | PB4 / MISO |
| twenty four | 16 | PB4: Programmable port B4 |
| | | MISO: SPI master input / slave output |
| | | PB5 / SPCK / AC1P |
| 25 | 17 | PB5: Programmable port B5 |
| | | SPCK: SPI clock signal |
| | | AC1P: Analog Comparator 1 positive input |





| | | |
|---|---|---|
| | | PE7 / ADC11 |
| 26 | - | PE7: Programmable port E7 |
| | | ADC11: ADC analog input channel 11 |
| 27 | - | AVCC: Internal analog circuit power supply |
| | | PE0 / SWC / APN4 |
| 28 | 18 | PE0: Programmable port E0 |
| | | SWC: SWD debug interface clock |
| | | APN4: Differential Amplifier Reverse Input Channel 4 |
| | | PE1 / ADC6 / ACXP |
| 29 | 19 | PE1: Programmable port E1 |
| | | ADC6: ADC Analog Input Channel 6 |
| | | ACXP: Analog Comparator 0/1 Common Positive Input |
| | | PE6 / ADC10 / AVREF |
| 30 | 20 | PE6: Programmable port E6 |
| | | ADC10: ADC Analog Input Channel 10 |
| | | AVREF: ADC external reference input |
| 31 | - | CVREF: ADC reference voltage output |
| | | Only for external 0.1uF filter capacitor |
| 32 | - | AGND: Internal analog circuit ground |
| | | PE2 / SWD |
| 33 | twenty one | PE2: Programmable port E2 |
| | | SWD: SWD debug interface data cable |
| | | PE3 / ADC7 / AC1N |
| 34 | twenty two | PE3: Programmable port E3 |
| | | ADC7: ADC analog input channel 7 |
| | | AC1N: Analog Comparator Negative Input |
| | | PC0 / ADC0 / APP0 |
| 35 | twenty three | PC0: Programmable port C0 |
| | | ADC0: ADC analog input channel 0 |
| | | APP0: Differential amplifier forward input channel 0 |
| | | PC1 / ADC1 / APP1 |
| | | PC1: Programmable port C1 |

| | | |
|---|---|---|
| 36 | twenty four | ADC1: ADC Analog Input Channel 1<br>APP1: Differential amplifier forward input channel 1 |
| 37 | 25 | PC2 / ADC2 / APN0<br>PC2: Programmable port C2<br>ADC2: ADC analog input channel 2<br>APN0: Differential amplifier reverse input channel 0 |
| 38 | 26 | PC3 / ADC3 / APN1<br>PC3: Programmable port C3<br>ADC3: ADC Analog Input Channel 3<br>APN1: Differential Amplifier Reverse Input Channel 1 |





| | | |
|---|---|---|
| 39 | 27 | PC4 / ADC4 / SDA<br>PC4: Programmable port C4<br>ADC4: ADC Analog Input Channel 4<br>SDA: I2C controller data cable |
| 40 | 28 | PC5 / ADC5 / SCL<br>PC5: Programmable port C5<br>ADC5: ADC Analog Input Channel 5<br>SCL: I2C controller clock line |
| 41 | 29 | PC6 / RESETN<br>PC6: Programmable port C6<br>RESETN: External reset input |
| 42 | - | PC7 / ADC8 / APN2<br>PC7: Programmable port C7<br>ADC8: ADC analog input channel 8<br>APN2: Differential Amplifier Reverse Input Channel 2 |
| 43 | - | PF0 / ADC9 / APN3<br>PF0: Programmable port F0<br>ADC9: ADC analog input channel 9<br>APN3: Differential Amplifier Reverse Input Channel 3 |
| 44 | 30 | PD0 / RXD<br>PD0: Programmable port D0<br>RXD: USART data reception input |
| 45 | 31 | PD1 / TXD<br>PD1: Programmable port D1<br>TXD: USART data transmission output |
| 46 | | PF1 / OC3A<br>PF1: Programmable port F1<br>OC3A: Timer 3 Compare Match Output A |
| 47 | 32 | PD2 / INT0 / AC0O<br>PD2: Programmable port D2<br>INT0: External interrupt input 0<br>AC0O: Analog compare 0 output |
| 48 | | PF2 / OC3B<br>PF2: Programmable port F2<br>OC3B: Timer 3 Compare Match Output B |

## LGT8XM kernel

- Low power design

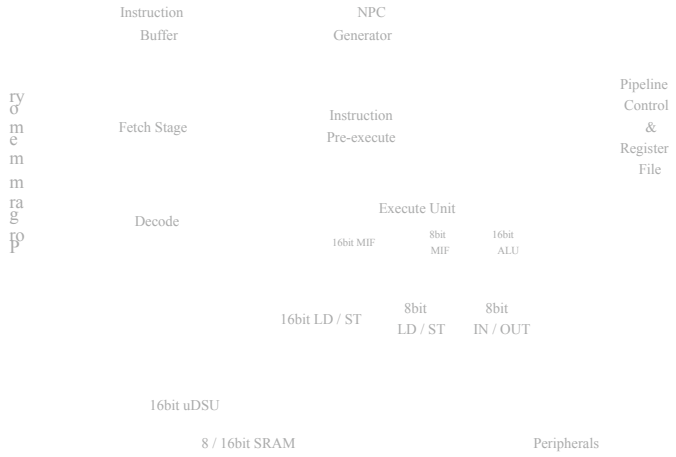
High efficiency RISC architecture

- 16-bit LD / ST extension (dedicated for uDSU)
- 130 instructions, of which more than 80% for a single cycle
- Embedded online debugging (OCD) support

### Overview

This section describes the LGT8XM kernel architecture and functionality. The kernel is the brain of the MCU, responsible for ensuring that the program is correct Execution, so the kernel must be able to perform computations, control peripherals, and handle various interrupts accurately.

LGT8XM kernel structure




To achieve greater efficiency and parallelism, the LGT8XM core uses a Harvard architecture - a separate data and program bus.
The instruction is executed through an optimized two-stage pipeline, and the two-stage pipeline can reduce the number of invalid instructions in the pipeline
The FLASH program memory access, so you can reduce the power consumption of the kernel. While the LGT8XM kernel in the fetch
So that the order of the increase in the instruction cache (which can cache two instructions), through the instruction cycle in the pre-execution module,
Further reducing the FLASH program memory access frequency; after a lot of testing, LGT8XM can be compared to other similar architecture
Of the kernel to reduce the access to about 50% of the FLASH, greatly reducing the operating power of the system.

The LGT8XM core has 32 8-bit high-speed access to the common file register (Register file), help to achieve a single week
Period of arithmetic and logic operations (ALU). Under normal circumstances, ALU operation of the two operands are from the general working register, ALU
The result of the operation is also written to the register file in one cycle.

**Page 9**



32 of the 6 working registers are used to combine the two 16-bit registers, which can be used for indirect addressing
Address pointer, used to access external storage space and FLASH program space. LGT8XM supports single cycle 16-bit arithmetic
Count, greatly improve the efficiency of indirect addressing. The three special 16-bit registers in the LGT8XM core are named X, Y,
The Z register will be described later in detail.

ALU supports the arithmetic and logic operations between registers and between constants and registers. A single register operation can also be performed
Executed in ALU. After the ALU operation is complete, the effect of the operation result on the kernel state is updated to the status register (SREG).

Program flow control through the conditions and unconditional jump / call to achieve, can be addressed to the program area. most
The LGT8XM instruction is 16 bits. Each program address space corresponds to a 16-bit or 32-bit LGT8XM instruction.

After the kernel responds to an interrupt or subroutine call, the return address (PC) is stored on the stack. The stack is assigned to the system one
In the data SRAM, so the size of the stack is limited only by the size and usage of the SRAM in the system. All support interrupted or
Subroutine calls must be initialized by initializing the stack pointer register (SP), which can be accessed through IO space. data
SRAM can be accessed through 5 different addressing modes. LGT8XM's internal storage space is linearly mapped to one
Uniform address space. Please refer to the description of the storage section.

The LGT8XM core contains a flexible interrupt controller, which can be accessed via a status register
Board interrupt enable bit control. All interrupts have a separate interrupt vector. The interrupt priority is associated with the interrupt vector address
Corresponding relationship, the smaller the interrupt address, the higher the priority of the interrupt.

The I / O space contains 64 register spaces that can be addressed directly by the IN / OUT instruction. These registers are realistic
Core control, and status registers, SPI and other I / O peripherals control functions. This part of the space can be through IN / OUT
Direct access can also be accessed through the address they are mapped to the data memory space (0x20 - 0x5F). In addition,
The LGT8FX8P also includes extended I / O space, which is mapped to data storage space 0x60 - 0xFF, which can only be used
ST / STS / STD and LD / LDS / LDD instructions.

To enhance the computing power of the LGT8XM core, the 16-bit LD / ST extension is added to the instruction line. This is 16 bits
LD / ST expansion with 16 digital arithmetic acceleration unit (uDSU) work, to achieve efficient 16-bit data operations. At the same time the kernel also
Increases 16-bit access to RAM space. So 16-bit LD / ST extensions can be sent in uDSU, RAM, and work
The 16-bit data is passed between the registers. For details, refer to the "Digital Operation Accelerator" section.

### Arithmetic logic operation unit (ALU)

The LGT8XM contains a 16-bit arithmetic logic unit that can be completed in one cycle.
The arithmetic operation. The efficient ALU is connected to 32 general purpose working registers. Be able to complete two registers in one cycle
Or the arithmetic and logic operations between the register and the immediate data. ALU operations are divided into three kinds: arithmetic, logic and bit operations.
At the same time ALU part also contains a single cycle of the hardware multiplier, in a cycle to achieve two 8-bit register
Direct sign or unsigned operation. Please refer to the instruction set section for details.

### Status register (SREG)

The status register mainly stores the result information generated by the last ALU operation. This information is used
Control the program execution flow. The status register is updated after the ALU operation is complete, which eliminates the need for a separate
Of the comparison instructions, can bring more compact and efficient code to achieve. The value of the status register is in response to the interrupt and retry from the interrupt
When the time will not automatically save and restore, which requires software to achieve.



**Page 10**



#### SREG register definition

SREG system status register

Address: 0x3F (0x5F) Default: 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 | A carry flag indicates that an arithmetic or a logical operation has caused a carry. For details, refer to the instruction description |
| [1] | Z | Zero flag, indicating that the result of arithmetic or logical operation is zero, refer to the instruction description section Minute |
| [2] | N. | A negative sign indicates that a mathematical or logical operation produces a negative number, please refer to the instruction Part |
| [3] | V | Overflow flag, indicating that the result of the two's complement operation overflow, please refer to the instruction description Part |
| [4] | S | Sign bit, equivalent to N and V of the XOR operation results, please refer to the specific instructions section |
| [5] | H | The semi-carry flag, which is useful in the BCD operation, indicates that the byte operation produces a half advance Bit |
| [6] | T | Temporary bit, bit copy (BLD) and bit memory (BST) instructions, the T bit will be used as a A temporary storage bit that is used to temporarily store the value of a bit in the general purpose register. Please refer to the instruction description section |
| [7] | I | The global interrupt enable bit must be set to 1 for this bit to enable the kernel to respond to the event Pieces. The different interrupt sources are controlled by independent control bits. The global interrupt enable bit is Control the interrupt signal into the kernel of the last barrier. I bit in the kernel response interrupt The hardware is automatically cleared by the hardware and is automatically set after the interrupt return instruction (RETI) is executed. The I bit can also be changed using the SEI and CLI instructions, refer to the instruction description section |

### Generic working register

The general purpose register is optimized for the LGT8XM instruction set architecture. In order to achieve the efficiency and flexibility required for kernel execution, LGT8XM internal common working registers support several access modes:

An 8-bit read while an 8-bit write operation

- Two 8-bit read at the same time an 8-bit write operation
- Two 8-bit read at the same time a 16-bit write operation
- A 16-bit read while a 16-bit write operation



LGT8XM universal working register

| 7 | 0 | Addr |
|---|---|---|
| | R0 | 0x00 |
| | R1 | 0x01 |
| | R2 | 0x02 |
| | ... | |
| | R13 | 0x0D |
| | R14 | 0x0E |
| through | R15 | 0x0F |
| use | R16 | 0x10 |
| work | R17 | 0x11 |
| For | | |
| send | | |

| Save Device | ... | | |
|---|---|---|---|
| | R26 | 0x1A | X register low byte |
| | R27 | 0x1B | X register high byte |
| | R28 | 0x1C | Y register low byte |
| | R29 | 0x1D | Y register high byte |
| | R30 | 0x1E | Z register low byte |
| | R31 | 0x1F | Z register high byte |

Most instructions have direct access to all common working registers, and most of them are single-cycle instructions.
As shown in the figure above, each register corresponds to the address of a data memory space, which is mapped to
Data storage space. As soon as they do not really exist in the SRAM, but this unified mapping of the storage organization to visit
They brought great flexibility. The X / Y / Z register can be indexed into any general register as a pointer.

### X / Y / Z register

The registers R26 ... R31 can be combined in two to form three 16-bit registers. These three 16-bit registers are mainly used for indirect
Addressing address pointer, X / Y / Z register structure is as follows:

| | | | | | | |
|---|---|---|---|---|---|---|
| | 15 | XH | | | XL | 0 |
| X register | 7 | | 0 | 7 | | 0 |
| | R27 (0x1B) | | | R26 (0x1A) | | |
| | 15 | YH | | | YL | 0 |
| Y register | 7 | | 0 | 7 | | 0 |
| | R29 (0x1D) | | | R28 (0x1C) | | |
| | 15 | ZH | | | ZL | 0 |
| Z register | 7 | | 0 | 7 | | 0 |
| | R31 (0x1F) | | | R30 (0x1E) | | |

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



### Stack pointer

The stack is used to store temporary data, local variables, and return addresses for interrupts and subroutine calls. Need special attention
Yes, the stack is not designed to grow from a high address to a low address. The stack pointer register (SP) always points to the top of the stack. Stack
The pointer points to the physical space where the data SRAM is located, where the subroutine or interrupt call must hold the stack space. PUSH means
So that the stack pointer will be decremented.

The location of the stack in the SRAM must be set correctly by the software before the subroutine is executed or the interrupt is enabled. General situation
In this case, the stack pointer is initialized to the highest address of the SRAM. The stack pointer must be set to the high bit SRAM at the beginning
site. SRAM Refer to the system data storage section for the address of the system data storage map.

Stack pointer related to the instruction

| instruction | Stack pointer | description |
|---|---|---|
| PUSH<br>CALL | Increase by 1 | Data is pushed onto the stack |
| ICALL<br>RCALL | Increase by 2 | The return address of the interrupt or subroutine call is pushed onto the stack |
| POP | Reduced by 1 | The data is fetched from the stack |
| RET<br>RETI | Reduced by 2 | The return address of the interrupt or subroutine call is removed from the stack |

The stack pointer consists of two 8-bit registers allocated in the I / O space. The actual length of the stack pointer matches the system implementation
turn off. In some chip implementations of the LGT8XM architecture, the data space is so small that only SPLs can satisfy addressing

In this case, the SPH register will not appear.

SPH / SPL Stack Pointer Register Definition

SPH / SPL Stack Pointer Register

| | |
|---|---|
| SPH: 0x3E (0x5E)<br>SPL: 0x3D (0x5D) | Default: RAMEND |
| SP | SP [15: 0] |
| R / W | R / W |

Bit definition

| | | |
|---|---|---|
| [7: 0] | SPL | The stack pointer is low for 8 bits |
| [15: 8] | SPH | The stack pointer is 8 bits high |

## Instruction execution timing

This section describes the general timing concepts for instruction execution. The LGT8XM kernel is driven by the kernel clock (CLKcpu), this time
The clock comes directly from the system with the clock source selection circuit.

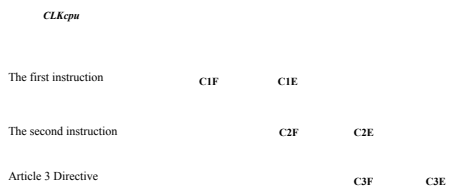
The following figure shows the execution timing of the instruction pipeline based on the concept of the Harvard architecture and the fast access register file. This is to make





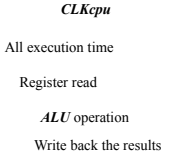
The kernel can get the physical guarantee of 1MIPS / MHz execution efficiency.

As can be seen from the above figure, the first instruction will be read during the implementation of the second instruction. When the second instruction goes into execution




Line period, while reading the third instruction at the same time. So that during the entire execution, there is no need to spend extra for reading instructions
Cycle, from the pipeline point of view, to achieve every Monday to implement the efficiency of a directive.

The following figure shows the access timing of the general working register. In one cycle, the ALU operation uses two registers as
Operand, and the ALU execution result is written to the destination register during this period.




## Reset and interrupt handling

LGT8XM supports multiple interrupt sources. These interrupts and reset vectors in the program space correspond to a separate program
Volume entrance. In general, all interrupts have separate control bits. When the control bit is set, and enabled
After the kernel's global interrupt enable bit, the kernel can respond to this interrupt.

The lowest program space is retained by default as the reset and interrupt vector area. LGT8FX8P supports a complete interrupt list
Please refer to the description of the interrupt section. This list also determines the priority of the different interrupts. The lower the vector address is the interrupt,
The corresponding interrupt priority is higher. The reset (RESET) has the highest priority and then the 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.
To be implemented by the IVSEL bit in the MCU control register (MCUCR) and the IVBASE vector base address register.

When the kernel response is interrupted, the global interrupt enable flag, I, is automatically cleared by hardware. The user can make the I bit by
Can achieve interrupt nesting. So that any subsequent interruption will interrupt the current interrupt service routine. I bit in the execution interrupt
After the return instruction (RETI) is set automatically, it can normally respond to subsequent interrupts.

There is a basic type of interrupt. The first type is triggered by an event, and the interrupt flag is set after an interrupt event occurs. for
This interrupt, the kernel response to the interrupt request, the current PC value is directly replaced by the actual interrupt vector address,
Line corresponding to the interrupt service subroutine, while the hardware automatically clear the interrupt flag. The interrupt flag can also be passed to the interrupt
The position of the flag bit is cleared by 1. If the interrupt enable bit is cleared when an interrupt occurs, the interrupt flag bit will still be set
To record an interrupt event. Wait until the interrupt is enabled, this record of the interrupt event will be immediately respond. Again, if in the interruption
When executed, the global interrupt enable bit (SERG.I) is cleared and the corresponding interrupt flag bit is set to record the interrupt event, etc.



**Page 14**



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

The second interrupt type is when the interrupt condition is always present, the interrupt is always responding. This interrupt does not require an interrupt flag
Bit. If the interrupt condition disappears before the interrupt is enabled, the interrupt will not be acknowledged.

When the LGT8XM kernel exits from the interrupt service routine, the execution flow returns to the main program. In the main program
After executing one or more instructions, you can respond to other waiting interrupt requests.

It should be noted that the system status register (SREG) does not automatically save after entering the interrupt service,
The interrupt service is automatically restored after returning. It must be handled by the software.

Interrupts are disabled immediately when interrupts are disabled using the CLI instruction. After the CLI instruction occurs so the interrupt is both
Will not get a response. Even if the interrupt is executed concurrently with the CLI instruction, it will not be responded. The following example says
How to use the CLI to avoid interrupting the write sequence of the EEPROM:

### Interrupt response time

The LGT8XM core is optimized for interruptions, making any interrupts available in the four system clock cycles
response. After four system clock cycles, the interrupt service routine enters the execution cycle. In the four clocks, before the interruption
PC value is pushed onto the stack, the system execution process flow jumps to the interrupt vector corresponding to the interrupt service routine. If an interrupt occurs
During a multi-cycle instruction execution, the kernel will ensure that the correct execution of the current instruction ends. If the interrupt occurs at the system
In the sleep state (SLEEP), the interrupt response requires an additional 4 clock cycles. This increased clock cycle is used from the selection
The synchronization period of the wake-up operation in sleep mode. For details of the sleep mode, refer to the relevant section of Power Management.

It takes 2 clock cycles to return from the interrupt service routine. In the two clock cycles, the PC is restored 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 cells within the LGT8FX8P family. The LGT8XM architecture supports two main internal Storage space, respectively, data storage space and program storage space. LGT8FX8P internal also contains the data FLASH, through The internal controller can implement the data storage function of the EEPROM interface. In addition, LGT8FX8P system also contains a special Of the storage unit, used to store system configuration information and the chip's global device number (GUID).

LGT8FX8P series chip contains the LGT8F88P / 168P / 328P four different models; four types of peripherals and Package is fully compatible, the difference is FLASH program storage space and internal data SRAM, the following table is more clear Describes the LGT8FX8P series chip different storage space configuration:

| DEVICE | FLASH | SRAM | E2PROM | Interrupt vector |
|---|---|---|---|---|
| LGT8F88P | 8KB | 1KB | 2KB | 1 instruction word |
| LGT8F168P | 16KB | 1KB | 4KB | 2 instruction words |
| LGT8F328P | 32KB | 2KB | Can be configured as 0K / 1K / 2K / 4K / 8K (Shared with FLASH) | 2 instruction words |

LGT8F328P is not used internally to simulate the E2 space of the E2PROM interface; used to simulate the E2PROM Storage space and program FLASH share, the user can according to application requirements, select the appropriate configuration.

Due to the unique implementation of the analog E2PROM interface, the system requires twice the program FLASH space to simulate the E2PROM Storage space, such as for LGT8F328P, if the user configured 1KB of E2PROM space, there will be 2KB bytes The program space is retained, leaving the 30KB of FLASH space for storing the program.

LGT8F328P program FLASH and E2PROM shared configuration table:

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

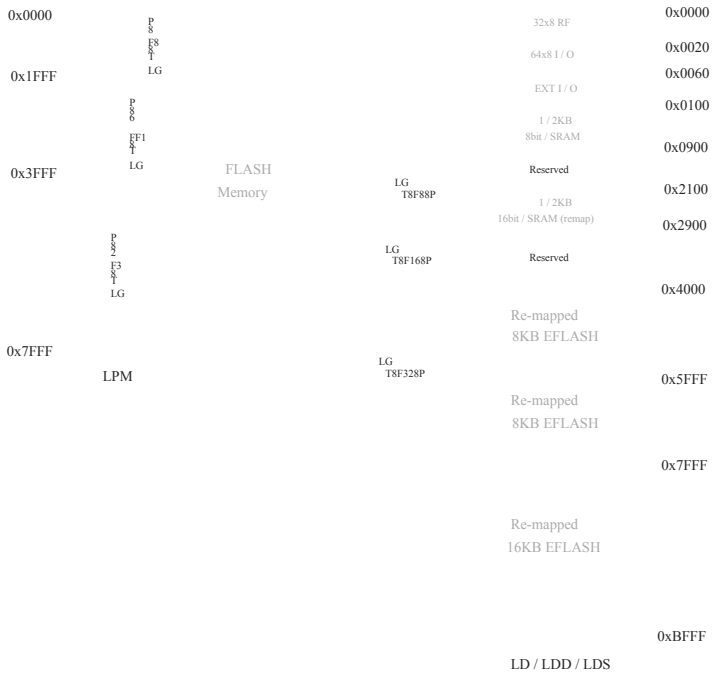
## System programmable FLASH program storage unit

LGT8FX8P series microcontrollers, respectively, including 8K / 16K / 32K bytes of on-chip online programmable FLASH program Storage unit.

The program FLASH guarantees at least 100,000 erase cycles. LGT8FX8P internal integrated FLASH interface control (ISP) and the program since the upgrade function. Please refer to this chapter for details FLASH interface controller part of the description.

Program space can also be accessed directly through the LPM instruction (read), this feature can be applied to the application of constant search

table. At the same time FLASH program space is also mapped to the system data storage space, the user can also use LD / LDD / LDS real FLASH space is now on the visit. The program space is mapped to the address range starting from the data memory space 0x4000.
As shown below:




## SRAM data storage unit

The LGT8FX8P family of microcontrollers is a relatively complex microcontroller that supports a variety of different types of peripherals. Some peripheral controllers are allocated in 64 I / O register spaces. Can be accessed directly through the IN / OUT instruction. others Peripheral control register is allocated in the 0x60 ~ 0xFF area, because this part of the space is mapped to the data storage space, Can only be accessed via ST / STS / STD and LD / LDS / LDD instructions.

LGT8FX8P system data storage space from the 0 address, respectively, mapping the general working register file, I / O empty Inter-expansion I / O space and internal data SRAM space. The beginning of the 32-byte address corresponds to the LGT8XM kernel 32 Generic working register. The next 64 addresses are standard I / O spaces that can be accessed directly through the IN / OUT instruction. Then the 160 addresses are extended I / O space, followed by up to 2K bytes of data SRAM. Open from 0x4000 The beginning of this part of the space to 0xBFFF, mapped FLASH program storage unit.

The 1K / 2K bytes of SRAM are mapped to two spaces. From 0x0100 to 0x0900 end of this The space is read and written by the kernel in 8-byte width. Starting from 0x2100 to 0x2900 End this area is 16-bit wide Access space. System RAM is mapped to 0x2100 at the beginning of the high address is mainly used to cooperate with the uDSU module, real Efficient 16-bit data storage. In the programming, the ordinary 8-bit address variable address plus 0x2000 offset, You can switch to 16-bit access mode.



The system supports five different addressing modes that can cover the entire data space: direct access, offset with indirect access Indirect access, indirect access to submissions before access, indirect access to incremental addresses after access. Generic working register R26 to R31 for indirect access to the address pointer. Indirect access can address the entire data storage space. With offset address Indirect access can be addressed to 63 address spaces near the Y / Z register base address.

The address register X / Y / Z will occur at the time of access when using a register indirect access mode that supports auto-increment / decrement of address Front / rear is automatically decremented / incremented by hardware. Please refer to the instruction set description section.

The 16-bit register X / Y / Z and the associated auto addressing mode (increment, decrement), also in 16-bit extended mode
Has a very important role. 16-bit extended mode can use the LD / ST increment / decrement mode to achieve automatic variables with variables
Incremental, decreasing addressing. This mode will be very effective when performing operations on arrays. For details, please refer to the "figures."
Operation Accelerator (uDSU) "related section.

### General I / O register

LGT8FX8P I / O space has three general-purpose I / O registers GPIOR2 / 1/0, these three registers can use IN / OUT means
Make access to user-defined data.

### Peripheral register space

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

LGT8FX8P so the peripherals are assigned to the I / O space. All I / O space addresses can be LD / LDS / LDDD
As well as ST / STS / STD instruction access. The accessed data is passed through 32 general purpose working registers. In 0x00 ~ 0x1F
The I / O registers can be accessed by bit addressing instructions SBI and CBI. In these registers, the value of a bit can be
To use the SBIS and SBIC instructions to detect the execution of the program. Please refer to the instruction set description section.

When using the IN / OUT instruction to access the I / O register, the address between 0x00 and 0x3F must be addressed. When using LD
Or the ST instruction accesses the I / O space, the mapping address must be mapped through the I / O space in the system data memory
Access (plus offset 0x20). Some other peripherals allocated in the extended I / O space register (0x60 ~ 0xFF), only
Enough to use ST / STS / STD and LD / LDS / LDD instructions.

In order to be compatible with future devices, the reserved bit must be written to 0 when writing. Can not write on reserved I / O space
operating.

Some registers include a status flag that needs to be written to 1 to clear. It should be noted that the CBI and SBI instructions
Only support a specific bit, so CBI / SBI can only work in the register containing these status flags. In addition,
CBI / SBI instructions can only work in the 0x00 to 0x1F address range of the register.

### FLASH controller (E2PCTL)

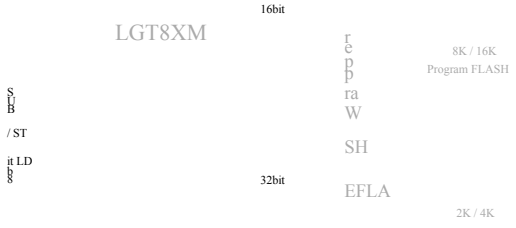
LGT8FX8P internal integration of a flexible and reliable EFLASH read and write controller, you can use the system has been
Data FLASH memory space, to achieve byte read and write access to the storage space, to achieve similar E2PROM storage applications; E2PROM
Interface simulation using erase equalization algorithm, the data FLASH can be used to improve the cycle of about 1 times, to ensure that
100,000 times more erase cycles.

E2PCTL controller also achieved on the FLASH program space online erase operation, you can achieve through the software online



Dynamic upgrade firmware. Through the FLASH controller to access the program FLASH program space, only supports page erase (1024 bytes)
As well as 32-bit read and write access.

LGT8F88D / 168D E2PCTL controller structure diagram




8bit

E2PCTL analog E2PROM function access data FLASH space, you can support 8-bit, 32-bit read and write width. visit
When asked for program FLASH space, support page erase and 32-bit data read and write. Due to the minimum deposit of LGT8FX8P internal FLASH
The storage unit is 32 bits, so it is recommended to use 32-bit access, especially for write operations. 32-bit access to read and write operations do not
Only high efficiency, but also help protect the FLASH memory unit erase life.

LGT8F328P E2PCTL controller structure diagram


LGT8F328P internal no extra data FLASH. Therefore, LGT8XM kernel and E2PCTL share internal 32K words
Section FLASH storage space. Users can, according to need, 32K bytes of FLASH space is divided into program space and data empty
between. By configuring the E2PCTL controller, you can set the spatial size of the analog E2PROM. E2PCTL uses page exchange mode
Implementation of analog E2PROM logic, the algorithm to page (1K bytes) as a unit. So simulate 1K bytes of E2PROM space, need to be
To take 2K bytes of FLASH space, and so on, to achieve 4K bytes of E2PROM, need to take 8K bytes
FLASH space. For details, please refer to the description of E2PCTL algorithm implementation.



## E2PCTL data register

E2PCTL controller has a 4-byte data cache (E2PD0 ~ 3), the 4 bytes of the cache composition of the final visit
Flash space 32-bit data interface.

When the E2PCTL controller works in byte read and write mode, EEDR as the interface to read and write byte data, E2PCTL more
Plus EEARL [1: 0] address information to load the data into the correct data cache, and according to the current FLASH target address data
Fill the other three bytes of data, the final combination of the complete 32-bit data update to FLASH.

When the E2PCTL is operating in 32-bit read / write mode, the EEDR register can still be used as a common
Data interface, through EEARL [1: 0] as address addressing internal data cache, to read and write a complete 32-bit data.
In addition, you can directly use the data cache to map directly to the IO space register (E0 ~ 3).

E2PCTL work in 8-bit read and write mode when the data access diagram:


E2PCTL

E2PCTL work in 32-bit word read and write mode when the data access diagram:


The byte mode is used for backward compatibility with the LGT8FX8D byte read and write mode. LGT8FX8P built-in FLASH for 32-bit access Mouth width, the use of 32-bit read and write mode will give read and write efficiency and FLASH erase life to bring great benefits, so built Use 32-bit read and write mode.

### E2PCTL Simulation E2PROM Interface Algorithm

We know that the FLASH memory must be erased before writing, and the erase operation is in page units. LGT8FX8P Built-in FLASH memory The size of a page is 1K bytes. So in order to update the page in one byte of data, too Need to first erase the entire page of the data, and then update the target address data, and at the same time restore the number of bytes in the page According to the whole operation is not only time-consuming, but also bring unexpected risk of loss of data due to power.

E2PCTL internal use of page exchange algorithm to achieve analog E2PROM. The page exchange algorithm mode ensures that the page erase is performed Operation, not because of power failure and other unexpected circumstances lead to the loss of the original data. Also exchange the algorithm using 2 page spaces



**Page 20**

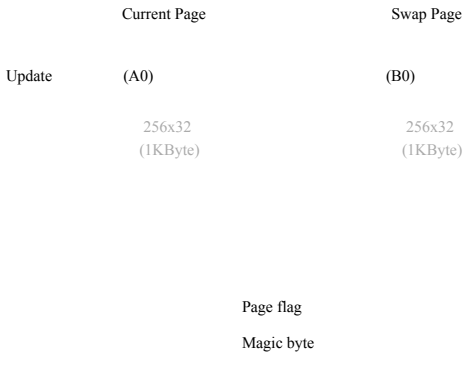


Alternating use of each other, but also increased the life of the analog E2PROM space.

In terms of efficiency, the E2PCTL controller implements a continuous data update model that reduces the ability to update data bands To repeat the erase process.

In terms of implementation, E2PCTL is managed separately for each page and occupies the last 2 bytes of a page as a page Status information. So users use more than 1K E2PROM analog space, the need to pay attention to the address across the 1K space Special treatment. Because the last 2 bytes per 1K space is reserved for E2PCTL, and the user can not use this 2-byte Space for normal reading and writing.

The following figure shows the logical diagram of the E2PCTL based page exchange algorithm:




As shown in the figure, E2PCTL uses two pages to simulate a page-sized E2PROM space. These two pages One is marked as the current page, and the other is the exchange page. E2PCTL uses the last 2 bytes of the page to store the page information. when We need to update a page in a byte, such as the above figure A0 bytes. First, we will not erase the current page But instead erase the swap page. And then the current page is divided into three parts of the operation. The first is before the A0 data we have This part of the space to become CP0, then A0 after the data, this part of the space for the CP1. E2PCTL will be based on the user with Set the corresponding data of CP0 to the corresponding address of the exchange page, and then need to update the data written to the corresponding page Address (B0), and finally copy the CP1 data to the swap page.

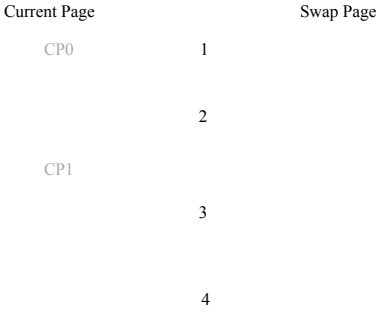
After completing the above operation, the data has been exchanged, but the page status has not been updated. So if it happened before that Power-down or other anomalies, this update operation because it is not completed, the previous data will not be destroyed, to ensure that the data

The integrity of the. If all goes well, E2PCTL will exchange the updated page status before CP1 exchanges data.
The exchange page page information, to achieve the face page of the replacement. After that, the swap page becomes the current page.

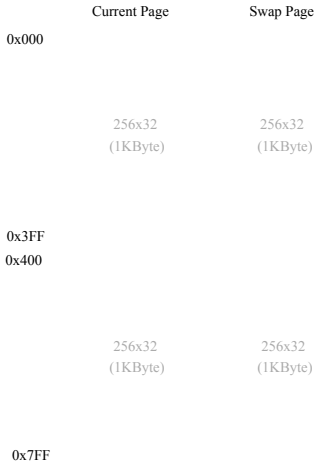
E2PCTL page exchange process as shown below (1-> 2-> 3-> 4):








When the system configuration E2PROM simulation space is greater than 1K, E2PCTL or page as the smallest unit to achieve E2PROM
Space simulation algorithm. For example, if the user configured 2K E2PROM area, in fact E2PCTL will take 4 pages
Surface (4K) space. One of the two pages is a group, used to simulate a page size of the E2PROM space.




It should be noted that the user configured 2K bytes of E2PROM space is not continuous, because the last two of each page
Bytes will be used to save page status information.

### E2PCTL continuous programming mode

As the E2PCTL update will lead to page exchange, the page exchange process will erase the exchange page, page rub
In addition to not only time-consuming, but also increase the loss of FLASH life. So E2PCTL adds a continuous write mode. In continuous writing
Mode, the user can continuously update the E2PROM area, only in the final address of the continuation of the page exchange operation,
Continuous mode is more efficient for applications that need to continuously update a single piece of data.

The CSM bit of the continuous programming mode E2PCTL control register ECCR is enabled. Continuous mode enabled, the subsequent write operation The data will be written directly to the address corresponding to the swap page. In SWM mode, the write operation does not execute the CP0 / 1 area According to copy operation. Before writing the last byte, the software disables continuous mode via SWM, and then writes, then E2PCTL The full CP0 / 1 copy operation will be performed and the page status information will be updated.





### E2PCTL read and write FLASH program space

Through the E2PCTL controller, you can achieve read and write access to the program FLASH space. Unlike analog E2PROM Yes, access to the program FLASH space via E2PCTL completely requires software control. Proceed as follows:

1. Erase the target page, update the data before the first need to erase the target page, the page address through the EEAR register to Out. For erase command control on FLASH pages, refer to the definition of the EECR register.
2. Write the program FLASH space must be 32 units for the smallest unit. Set the data via E2PD0 ~ 3;
3. The destination address is given by the EEAR register and the address EEAR [1: 0] will be ignored;

Through E2PCTL read and write procedures FLASH space, you can achieve online program update (IAP) function, in some need on-site It is useful to update your application data and applications that need to provide product custom updates.

### E2PCTL interface operation flow

E2PCTL controller is the main work through the four registers to achieve, respectively, E2PCTL control status register EECR, ECCR; data register EEDR (E2PD0 ~ E2PD3) and address register EEAR (EEARL / EEARH).

ECCR register is used to set the working status of E2PCTL, most of the state need to set up before E2PCTL work is completed, This process is generally implemented in the system initialization process. The SWM bit in the ECCR register is used to enable the continuous write mode, This control bit needs to be set during the continuous write operation.

The EECR register is used to control the selection of the operation type, for selecting the operation instruction, such as setting the read and erase commands.

EEDR register for 8-bit byte mode interface, E2PD0 ~ 3 for 32-bit mode read and write operations;

The EEAR register is used to set the read and write destination addresses and also to set the page address of the page erase operation. The page address is The page units are aligned, and the size of the page is 1K bytes. Note that the address specified by EEAR is the byte address.

#### Through the E2PCTL interface to access FLASH program space:

Through the E2PCTL interface can be achieved on the FLASH program space to read and write and erase. FLASH only read and write space Hold 32-bit access width. Erase operation to page units, the size of each page 1K bytes (256x32).

Before writing the FLASH program space, first erase the page where the destination address is located. E2PCTL write FLASH program space The continuous mode is not supported and the user needs to complete the write in sequence. The following is to erase the FLASH program space process:

1. Program FLASH page erase operation
   - Set EEAR [14: 0] for the target page address to be erased, the program FLASH page size is 1K bytes, So EEAR [14:10] will be set to 0 as the page address, EEAR [9: 0]
   - Set EEPM [3: 0] = 1X01, where EEPM [2] can be set to 0 or 1
   - Set EEMPE = 1 while EEPE = 0
   - In the four cycles, set EEPE = 1 to start the program FLASH erase process

2. Program FLASH programming operation
   - Write E2PD0 ~ 3 to prepare 32-bit programming data
   - Set EEAR to the destination address, where the address is 4 bytes
   - Set EEPM [3: 0] = 1X10, where EEPM [2] can be set to 0 or 1
   - Set EEMPE = 1 while EEPE = 0
   - In the four cycles, set EEPE = 1 to start the FLASH programming flow

**Page 23**



Access E2PROM via E2PCTL interface Analog space:

The E2PCTL controller accesses the data FLASH space by simulating the E2PROM interface. Analog E2PROM support 8 Bit, 16-bit, and 32-bit data width. 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 and write mode for the proposed read and write mode formula. E2PROM analog interface supports continuous read and write mode, in the need to update a number of consecutive addresses of data applications, excellent Obvious, recommended.

For LGT8F88P / 168P, the data FLASH is a separate storage space. It is not necessary to configure and enable via ECCR registers Can FLASH data space. LGT8F328P does not have independent data FLASH space, data FLASH and program FLASH total Enjoy 32K bytes FLASH space. Need to pass the ECCR register to enable data FLASH partition function, and through ECCR register The size of the ECS [1: 0] bit configuration data FLASH. After the configuration takes effect, the other uses are the same as LGT8F88P / 168P.

FLASH controller in the realization of E2PROM interface, the internal has been achieved when necessary to automatically erase the data FLASH Logic, so the EPROM erase command is optional, and this command is only used if the user needs to perform a separate erase. The EECR register controls the erase / write timing of the FLASH, including the program FLASH and E2PROM. The specific type of operation needs to pass The EEPME and EEPM [3: 0] settings of the EECR register are set. Read the E2PROM relatively simple, set the target in the set Address and mode, write the EERE bit is the target address corresponding to the 32-bit data into the FLASH controller, the user can pass The EEDR register reads the bytes of interest. FLASH controller does not realize the program FLASH space read operation, The user can easily use the LPM or through the program FLASH in the data unified mapping space at the address LD / LDD / LDS instruction read.

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 while EEPE = 0
   - Set the EEPE = 1 in four cycles

   When the setting is complete, the FLASH controller will start the programming operation, during programming CPU will remain in the current instruction Address, until the operation is completed will continue to run. In the programming process, if you need to erase the data FLASH, The FLASH controller will automatically start the erase process.

2. 32-bit mode, programming E2PROM
   - Prepare 32-bit data via E2PD0 ~ 3
   - Set the destination address to the EEARH / L register. Note that here is the byte-aligned address of the FLASH controller Use 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 while EEPE = 0
   - Set the EEPE = 1 in four cycles

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 E2PROM read operation
   - Wait 2 cycles (perform two NOP operations)
   - The data corresponding to the destination address is updated to the EEDR register



**Page 24**



4. 32-bit mode, read E2PROM
   - Set EEARH / L as the destination address, and the address is 4 bytes
   - Set EEPM [3: 1] = 010 to enable 32-bit interface mode

- Set EERE = 1 to start E2PROM read operation
- Wait for 2 system clock cycles (two NOP instructions are executed)

E2PCTL access to analog E2PROM space to support continuous programming mode, continuous access mode for the need for an update one
The application of the data block is very efficient, but also help improve the life of FLASH. The continuous programming mode supports only 32-bit wide
The degree of data programming operation.

The continuous access mode is enabled by the SWM bit in the ECCR register. After SWM is enabled, write the module via E2PCTL
The operation of the proposed E2PROM space is in the continuous programming mode. In continuous programming mode, the E2PCTL controller will be based on the target
The data in the field is automatically processed. However, if a page change occurs during a continuous programming mode, the controller is programmed continuously
During the process, the data in the CP0 / 1 area is not automatically exchanged and the page information is not updated.

Before continuous programming to the last operation, clear the SWM bit by turning off the continuous programming mode, and then in the non-SWM mode
Type the last programming operation, E2PCTL will automatically copy the data in the CP0 / 1 area to the exchange.
Page, and update the exchange page information, making it a valid page, thus completing the entire continuous programming operation.

5. Continuous programming mode Operation flow:

1. Configure the size of the data FLASH via ECCR and enable the SWM bit
2. Simulate the E2PROM area 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, the continuous programming mode is disabled by SWM, and then the
   The operation process completes the last programming

### E2PCTL efficient FLASH data management

E2PCTL controller in addition to the realization of continuous programming mode, can also ECCR register CP0 / 1 bit on the page exchange
Process data exchange replication for independent control. ECCR register CP0 / 1 is used to control the page exchange process for the current
Page exchange of CP0 / 1 area data. Clearing the CP0 / 1 bit will not exchange the current page pair during page exchange
Should be the data of the area. This section provides an efficient management approach that will take advantage of this feature.

In the FLASH data update process, the most time-consuming operation occurred in the exchange page erase process. So we can address it
A data management method that minimizes the number of page erasures can improve programming efficiency and reduce life loss.

Here we provide a reference algorithm for data-based data management applications:

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

The above three conditions, we can take full advantage of E2PCTL continuous programming mode and automatic page exchange mechanism, to achieve
An efficient FLASH data management method.

Since each updated data is a data block of the same size, and each data structure is stored to point to the next
Block data address information, so we can update the data each time by address order FLASH, do not need to do CP0 / 1
Data replication. At the same time as a result of each time to update the data to an erased area, it will not happen page erase.

When the data is written in the last block, its structure information points to the next data area back to the start address of the page. Then happen again
Data write operation, E2PCTL will start a page erase process and update the current active page.



### FLASH operation of the protection measures

If the VCC voltage is low, FLASH erase operation may be due to the voltage is too low and an error.

FLASH / data erase operation at low pressure may be caused by two reasons. First, the normal FLASH erase operation needs
To a minimum operating voltage, below this voltage, the operation will fail and cause the data to be erroneous. The second reason is yes
Kernel running at a certain frequency, also need a minimum voltage requirements, when below this voltage, will lead to instruction
Line error, which makes FLASH operation error.

You can avoid similar problems by the following simple method:

When the supply voltage is low, let the system into the reset state. This can be done by configuring the internal low voltage detection circuit (VDT)
achieve. If VDT detects that the current operating voltage is below the set threshold, VDT will output a reset signal. in case
VDT threshold can not meet the needs of the application, you can consider adding a reset circuit in the external.

## Register description

FLASH address register - EEARH / EEARL

EEARH / EEARL

EEARH: 0x22 (0x42) Default: 0x0000

EEARL: 0x21 (0x41)

| bits | EEAR [15: 0] |
|---|---|
| R / W | R / W |

Bit definition

| | | |
|---|---|---|
| [7: 0] | EEARL | EFLASH / E2PROM access address is low 8 bits. |
| [14: 8] | EEARH | EFLASH / E2PROM access address is high 7 digits |
| [15] | - | Keep not used |

When using the E2PCTL controller to access the program FLASH area, EEAR [14: 2] is used to access the 4-byte alignment Program space. EEAR [1: 0] is used only when accessing the data register EEDR. Please refer to the following for EEDR data Register description. E2PCTL controller supports 8/16/32 bit mode, no matter what mode, where the EEAR are Byte-aligned addressing.

FLASH Data Register - EEDR / E2PD0

EEDR / E2PD0 - FLASH / E2PROM Data register 0

EEDR / E2PD0: 0x20 (0x40) Default: 0x00

| bits | EEDR [7: 0] |
|---|---|
| R / W | R / W |

Bit definition

| | | |
|---|---|---|
| [7: 0] | EEDR | E2PCTL data register |
| | E2PD0 | 16/32 bit mode, used to access the lowest byte |

FLASH Data Register - E2PD1

E2PD1 - E2PCTL Data Register

E2PD1: 0x5A Default: 0x00

| bits | E2PD1 [7: 0] |
|---|---|



**Page 26**



| R / W | R / W |
|---|---|

Bit definition

| | | |
|---|---|---|
| [7: 0] | E2PD1 | 16-bit mode is used to store the upper 8 bits of 16-bit data<br>32-bit mode 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: 0x00

| Bits | E2PD2 [7: 0] |
|---|---|
| R / W | R / W |

Bit definition

| | |
|---|---|
| [7: 0] | E2PD2 32-bit mode is used to store the lower 8 bits of the upper 16 bits of data |

FLASH data register - E2PD3

E2PD3 - FLASH DATA REGISTER 3

E2PD3: 0x5C Default: 0x00

| Bits | E2PD3 [7: 0] |
|---|---|
| R / W | R / W |

Bit definition

| | |
|---|---|
| [7: 0] | E2PD3 32-bit mode is used to store the upper 8 bits of the upper 16 bits of data |

### FLASH mode control register - ECCR

ECCR - FLASH / E2PROM Configuration register

ECCR: 0x36 (0x56) Default: 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<br>Before modifying ECCR, you must first write WEN 1 and then in 6 system cycles<br>, Update the contents of the ECCR register |
| [6] | EEN | E2PROM enabled, only valid for LGT8F328P<br>1: Enable E2PROM simulation, will be part of the space from 32KFLASH<br>0: disable E2PROM simulation, 32KFLASH all for program space |
| [5] | ERN | Writing a 1 will reset the E2PCTL controller |
| [4] | SWM | Continuous write mode for analog E2PROM controller operation |
| [3] | CP1 | Page exchange CP1 area enable control |
| [2] | CP0 | Page exchange CP0 area enable control |
| [1: 0] | ECS [1: 0] | E2PROM space configuration<br>00: 1KB E2PROM, 30KB program FLASH<br>01: 2KB E2PROM, 28KB program FLASH |





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

### FLASH access control register - EECR

EECR - FLASH / E2PROM control register

EECR: 0x1F (0x3F) Default: 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 bit<br>[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 operation) |
| | | 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 disabled. when<br>After the EEPE is cleared by hardware, the E2PROM ready interrupt is valid. In EPROM<br>This interrupt will not be generated during operation | | | | |
| [2] | EEMPE | FLASH / E2PROM programming operation enable control bits<br>EEMPE is used to control whether EEPE is valid, while setting EEMPE to 1, EEPE at the same time<br>After 0, in the next four cycles, setting EEPE to 1 will start the programming operation<br>For. Otherwise the programming operation is invalid. After four cycles, EEMPE is automatically cleared | | | | |
| [1] | EEPE | FLASH / E2PROM programming operation enable bit | | | | |
| [0] | EERE | E2PROM read enable bit, the data will be valid after two system cycles | | | | |

General Purpose I / O Registers - GPIOR2

GPIOR2 - General Purpose I / O Register 2

| GPIOR2: 0x2B (0x4B) | Default: 0x00 |
|---|---|
| Bits | GPIOR2 [7: 0] |
| R / W | R / W |
| Initial value | 0x00 |

Bit definition

| | | |
|---|---|---|
| [7: 0] | GPIOR2 | General purpose I / O register 2 for storing user-defined data |





General Purpose I / O Register - GPIOR1

GPIOR1 - General Purpose I / O Register 1

| GPIOR1: 0x2A (0x4A) | Default: 0x00 |
|---|---|
| Bits | GPIOR1 [7: 0] |
| R / W | R / W |
| Initial value | 0x00 |

Bit definition

| | | |
|---|---|---|
| [7: 0] | GPIOR1 | General purpose I / O register 1 for storing user-defined data |

General Purpose I / O Register - GPIOR0

GPIOR0 - General Purpose I / O Register 0

| GPIOR0: 0x1E (0x3E) | Default: 0x00 |
|---|---|
| Bits | GPIOR0 [7: 0] |
| R / W | R / W |
| Initial value | 0x00 |

Bit definition

| | | |
|---|---|---|
| [7: 0] | GPIOR0 | The general purpose I / O register 0 is used to store user-defined data |

## Computing Accelerator (uDSC)

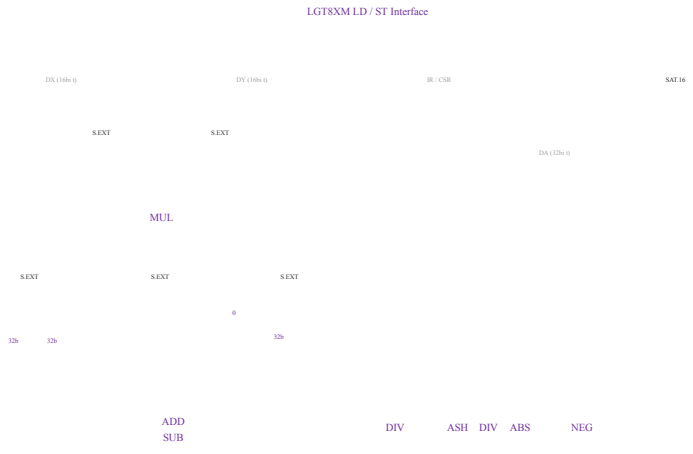
- 16-bit memory mode (LD / ST)
- 32-bit accumulator (DX)
- Single cycle 16-bit multiplier (MUL)
- 32-bit arithmetic and logic operation unit (ALU)
- 16-bit saturation operation (SD)
- 8-cycle 32/16 divider
- Single cycle multiplication / subtraction (MAC / MSC)

### Overview

Digital Computing Accelerator (uDSC) as an operator of the LGT8XM kernel co-processing module, with LGT8XM kernel 16-bit LD / ST mode, to achieve a 16-bit digital signal processing unit. Can meet most of the control class digital signal deal with.

uDSC function internal and function:

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




uDSC structure diagram

**Page 30**

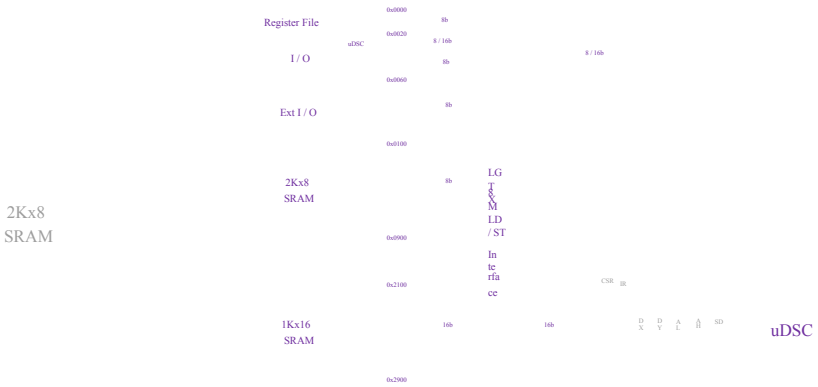


### 16-bit LD / ST operating mode

To improve the efficiency of uDSC to handle large amounts of data operations, the LGT8XM core implements a dedicated 16-bit LD / ST memory Channel, you can use the LDD / STD instruction to efficiently perform a 16-bit number between uDSC and SRAM and common register files According to exchange.

In order not to destroy the normal LD / ST instruction system, LGT8XM kernel to SRAM space remapping to 0x2100 ~ 0x28FF. Use the LD / ST instruction from 0x2100 ~ 0x28FF space to access SRAM, the kernel automatically open 16-bit LD / ST function, playing Open the direct access channel between SRAM and uDSC.

The following figure shows the data space address distribution for the LGT8XM kernel:




As shown above, the LGT8XM core can be used by using the LD / ST instruction in the uDSC DX / DY / DA register with SRAM directly between the 16-bit data access to access. At the same time uDSC internal registers are also mapped to the I / O space, The access uDSC register is divided into 8/16 modes.

In addition to the DX / DY / DA registers used for operation, the uDSC contains two additional 8-bit registers: the uDSC control status register CSR, and the operation instruction register IR. CSR / IR can only be in bytes by I / O space Access; 16-bit mode when accessing DX / DY / AL / AH. You can use IN / OUT and LD / ST / LDD / STD / LDS / STD Instruction access.

The control status and data registers associated with the uDSC are mapped to IO space and are directly addressed using the IN / OU instruction. To complete 8/16 bit of data access in one instruction cycle.

CSR is used to control the operating mode of the uDSC and to record the status flags of the current uDSC execution operations. IR control uDSC To achieve the specific operation. Most of the operations supported by uDSC are completed in one cycle, and seven operations are required for division operations Cycle, you can also use the flag in the CSR register to determine whether the current division operation is complete.

The standard LD / ST instruction uses the LGT8XM internal general purpose register as the LD / ST data, using the X / Y / Z As the destination address. When the destination address falls within the 16-bit SRAM mapping space, the LD / ST instruction operand has the meaning , Where X / Y / Z is still the target address, the meaning of the general working register addressing according to the uDSC mapping mode There are two ways to deal with it. The mapping mode for uDSC only applies to access to 0x2100 ~ 0x28FF address access. Mapping mode pass The sixth bit (MM) setting of the CSR register is set.



**Page 31**



In the 16-bit LD / ST mode, the instruction "LDD Rn, Z + q" indicates that the 16-bit data of the [Z] address is loaded into the uDSC Data register, and then add the value of Z to an offset "q". Here the meaning of Rn is related to the mapping mode CSR [MM]

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<br>[DX / DY / AL / AY] = [Z]; Z = Z + q<br>Rn keep unchanged |

STD Rn, Z / Y + q

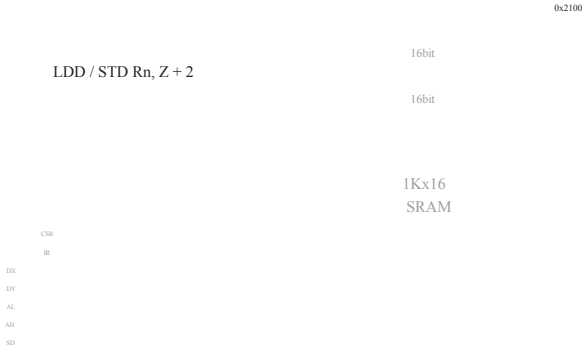
| | | | |
|---|---|---|---|
| 0 | 0x2100 ~ 0x28FF | STD Z + q, R0 | [Z] = DX; Z = Z + q; R0 kept unchanged |
| | | STD Z + q, R1 | [Z] = DY; Z = Z + q; R1 kept unchanged |
| | | STD Z + q, R2 | [Z] = AL; Z = Z + q; R2 kept unchanged |
| | | STD Z + q, R3 | [Z] = AH; Z = Z + q; R3 kept unchanged |
| | | 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<br>[Z] = [DX / DY / AL / AH / SD] addressed by {Rn}<br>Rn keep unchanged |

LG / ST, LDS / STS can access 0x2100 ~ 0x28FF area, but LDD / STD

The Y / Z + q addressing mode is more efficient. LDD / STD mode addressing based on a base address, we can set the Y / Z to
The base address of the data in the RAM can be executed in one cycle by using the Y / Z + q addressing mode of the LDD / STD instruction.
And access the data, and automatically move the address pointer to the next destination address.

The Y / Z + q offset addressing mode of the LDD / STD instruction of the LGT8XM core standard uses [Y / Z + q] as the instruction execution
8-bit data address, after the completion of the implementation of Y / Z value does not increase. When using LDD / STD addressing 0x2100 ~ 0x28FF interval
, The instruction behavior of the LDD / STD is changed: When the instruction is executed, [Y / Z] is used as the 16-bit data addressing address,
After execution, the value of Y / Z increases the offset specified by "q". This feature can improve the efficiency of our continuous addressing through
The "q = 2" can achieve continuous 16-bit data addressing.





uDSC 0x28FF

### The mapping between the variable address and the 16-bit mode address

The LGT8XM is an 8-bit processor with data access in bytes. PDN1512A built-in 2K bytes of data space.
This part of the space is mapped to 0x0100 ~ 0x08FF address. C / C ++ compilation automatically assigns variables to 0x0100 ~ 0x08FF
between. If we need to use uDSC in a 16-bit array defined in C / C ++, we need to
The address of the variable is mapped to the 16-bit LD / ST access address area (0x2100 ~ 0x28FF). The method is simple,
Variable address increase 0x2000 offset can be.

## uDSC operation instruction definition

The software specifies the operations that need to be implemented by the IR register of the uDSC. All operation of uDSC is done on DX / DY / DA
Between. Users can use 16-bit LD / ST channels in DX / DY / DA and SRAM directly and quickly exchange data.

| classification | IR [7: 0] | | | | | | | | Functional 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 |

Description:

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

## Register definition

| name | IO address | Functional description |
|---|---|---|
| DCSR | 0x20 (0x00) | Operation instruction register |
| DSIR | 0x21 (0x01) | uDSC control status register |

| | | |
|---|---|---|
| DSSD | 0x22 (0x02) | The 16-bit saturation operation of the 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 [31:16], 16-bit read and write access |

### DSCR - Control status register

DSCR - uDSC control status register

Address: 0x20 (0x00) Default: 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 = disabled |
| 6 | MM | uDSC register mapping mode; detailed definition please refer to 16-bit mode of operation.<br>0 = fast access mode, 1 = IO mapping mode |
| 5 | D1 | Division operation completion flag, 1 = operation is completed |
| 4 | D0 | Divide operation except 0 flag |
| 3 | - | Unimplemented |
| 2 | N. | The result is a negative flag |
| 1 | Z | The result of the operation is zero flag |
| 0 | C | 32 Adder carry / borrow flag |





### DSIR - operation instruction register

DSIR - uDSC operation instruction register

Address: 0x21 (0x01) Default: 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 of the "Operation instruction definition" section |

### DSDX - Operand Register DSDX

DSDX - uDSC Operand Register DX

Address: 0x30 (0x10) Default: 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: 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 - Low 16 bits of the 32-bit accumulator DA

DSAL is - uDSC operand register DSA lower 16 bits

Address: 0x58 (0x38) Default: 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 | DSAT | 32-bit accumulator DSA low 16 bits |





DSAH - The upper 16 bits of the 32-bit accumulator DA

DSAH - uDSC operand register The upper 16 bits of the DSA

Address: 0x59 (0x39) Default: 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: 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 saturated operation result of 32-bit accumulator DSA |

## uDSC application example

Example 1. Basic configuration and operation

The following is a simple subroutine (AVRGCC), which implements a 16-bit multiplication and returns 32-bit results.

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

The following is the compilation implementation code for the C function:

```
#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
```





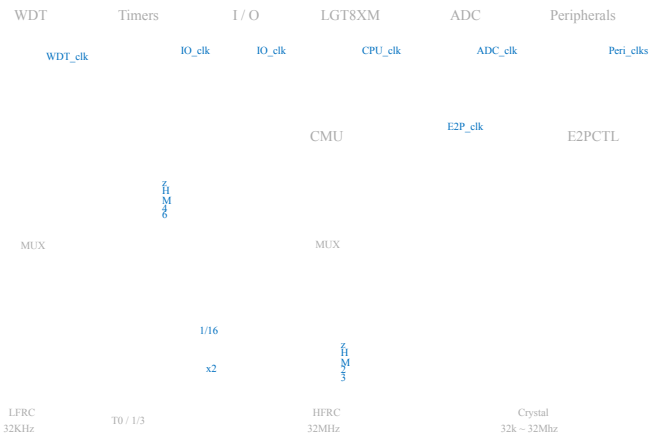


# System clock and configuration

## System clock distribution

The LGT8FX8P supports multiple clock inputs. The system can work in three main clock sources, namely internal 32KHz Calibrated RC Oscillator, Internal 32MHz Calibrated RC Oscillator, and External 400KHz ~ 20MHz Crystal Input.

The following figure shows the LGT8FX8P clock system distribution, CMU is the center of the entire clock management, responsible for the system clock frequency, The same module to generate a separate clock and control the clock and so on. In general applications, do not all the clock with the same When working in order to reduce system power consumption, the system power management switches the unused module clock according to the different sleep modes. For details on the operation, refer to the section on power management.




### CPU_clk

Used to drive the LGT8XM core and SRAM operation. Such as driving general purpose working registers, status registers, and so on.

After the CPU clock is stopped, the kernel will not continue to execute the instructions and perform the calculations. The system executes the SLEEP instruction to go to sleep Mode, the kernel clock will be turned off.

### Peri_clk

Used to drive most peripheral modules, such as timers / counters, SPI, USART, and so on. The IO clock is also used to drive the external Interrupt module. When the peripheral clock is halted, some peripherals that can use the wake-up system work independently Clock or asynchronous mode. For example, TWI address recognition function can wake up most of the sleep mode, the address at this time The recognition part works in asynchronous mode.

### E2P_clk

The E2P_clk clock is used to generate the FLASH interface access timing. E2P_clk generates access to the E2PCTL access to the FLASH interface The timing of the. E2P_clk Fixed 32-frequency (1MHz) from internal 32MHz HFRC oscillator. If the user needs to make Use E2PCTL module to read and write internal program FLASH or data FLASH space, need to enable the internal 32MHz vibration Device.

**Page 37**



Asy_clk

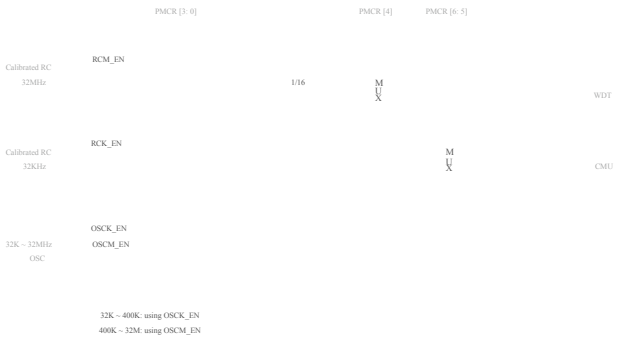
Asynchronous timer clock. The timer / event counter can be driven directly using an external clock or crystal (32.768K). This independence Clock mode, the timer can remain running while the system is in sleep mode.

WDT_clk

Internal watchdog timer clock source, can be configured to select the internal 32KHz LFRC oscillator, or from the internal 32MHz HFRC 16-way (2MHz). After the system is powered on, the watchdog default clock source is 32KHz LFRC oscillator.

## Clock source selection

LGT8FX8P supports four kinds of clock source input, the user can PMCR register to achieve the clock source enable control As well as the completion of the main clock switch. The following is the control structure of the PMCR:




LGT8FX8P internal OSC oscillator can work in high frequency and low frequency two modes, the user needs according to the external crystal The actual size of the internal OSC oscillator is controlled in the correct mode. The same internal RC oscillator is also divided into high frequency and Low frequency two kinds. 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 closed |
| PMCR [3] | 32K ~ 400K OSC mode enabled, 1 enabled, 0 off |

LGT8FX8P system power, the default use of 32MHz RC as a system clock source, the kernel work in the clock source of 8 points Frequency (2MHz). 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 main clock source configuration, it is necessary to ensure that the clock source after switching is stable before switching the clock. State. It is therefore necessary to enable the required clock source by PMCR [3: 0] before switching to the master clock source and wait for the clock to stabilize After the set to switch.

When the user switches the master clock to the external crystal, although the user enables the external crystal, but does not rule out the configuration error or Crystal failure can not cause crystal vibration. If you switch to an external crystal at this time, the system will stop working immediately after switching. because This, from the system reliability considerations, it is recommended by opening the watchdog timer, from the software design point of view to avoid such questions

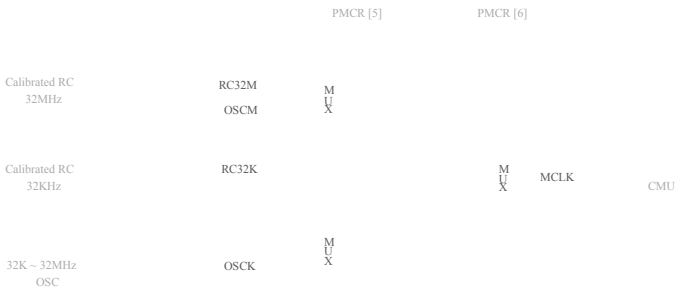


**Page 38**



question.

After the clock source is enabled and waiting for stabilization, the master clock can be switched by PMCR [6: 5]. Where PMCR [5] is used for selection Section RC oscillator and external crystal, PMCR [6] is used to select the high-speed clock source and low-speed clock source.




Main clock source selection:

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

### Clock source control timing

In order to protect the PMCR register from accidental modification, the modification of the PMCR register requires strict installation of the specified timing. The highest bit of the PMCR register (PMCR [7]) is used to implement timing control. The user must first change the other bits of the PMCR Set PMCR [7] first to change the value of other registers of PMCR during 6 cycles after setting operation. 6 cycles , The direct modification to PMCR will fail.

Below to switch to the external high-speed crystal, for example, list the recommended steps:

(1) Enable the clock source

- Set PMCR [7] = 1
- Set PMCR [2] = 1 in six cycles to enable external high-speed mode external crystal
- wait for external crystal stability (waiting time due to different crystal and different, generally us level can wait)

(2) toggle the main clock source

- Set PMCR [7] = 1
- Set PMCR [6: 5] = 01 in six cycles, the system will automatically switch the working clock to the external crystal
- Perform several NOP operations to improve stability (optional operation)

[Note]: In the above operation to switch the main clock, to ensure that the current system clock to work properly, in the switch to the external crystal After the vibration, you can turn off the internal RC oscillator before.



## System clock prescaler control

The LGT8FX8P has a system clock prescaler internally that can be controlled by the clock prescaler register (CLKPR). This function can be used to reduce system power consumption when the system does not require very high processing power. Prescaler settings for system support The clock source is valid. Clock prescaler can affect the kernel execution clock and so the synchronization peripherals.

When switching between different clock prescaler settings, the system clock prescale ensures that no hair is generated during the switching process Thorns, but will ensure that there will be no high frequency of the middle state. The frequency switching is performed immediately, and when the register change takes effect, Up to 2 to 3 current system clock cycles, the system clock switches to the new divide clock.

In order to avoid erroneous operation on the clock divider register, the modification to CLKPR must also follow a special timing flow

Cheng:

- Set the clock prescaler change enable bit (CLKPCE) to 1, CLKPR other bit is 0
- Write the required value to CLKPS in four cycles, while CLKPCE is written to 0

Before changing the clock prescaler register, it is necessary to disable the interrupt function to ensure that the write timing can be performed intact.
Refer to the Register Description section of this section for a detailed definition of the main clock prescaler register CLKPR.

## Internal RC oscillator calibration

The LGT8FX8P includes two calibratable RC oscillators that are calibrated to achieve accuracy of less than ± 1%. its
The 32MHz RC is used by default for the system operating clock.

LGT8FX8P pre-production, the internal 32MHz HFRC and 32KHz LFRC are calibrated, and the calibration value into the system
Configure the information area. During system power saving, these calibration values will be read into the internal registers,
Recalibrate RC frequency.

The calibration register is located in the IO address space and the user program can read and write. For the frequency of special needs of the application, you can
Adjust the frequency output of the internal oscillator by modifying the calibration register. Modifying the calibration register does not change the factory configuration letter
The system register will return to the factory settings if the operation is reloaded or the user-initiated configuration bit is reloaded.

## Register definition

### 32MHz HFRC Oscillator Calibration Register - RCMCAL

RCMCAL - 32MHz HFRC calibration register

| RCMCAL: 0x66 | Default: factory configuration |
|---|---|
| Bits | RCCAL [7: 0] |
| R / W | R / W |

Bit definition

| | | |
|---|---|---|
| [7: 0] | RCCAL | After the system is powered up, the value of the register will be replaced by the RC calibration value in the system configuration information change. |



### 32KHz RC Oscillator Calibration Register - RCKCAL

RCKCAL - 32MHz RC Calibration Register

| RCKCAL: 0x67 | Default: Factory setting |
|---|---|
| Bits | RCKCAL [7: 0] |
| R / W | R / W |

Bit definition

| | | |
|---|---|---|
| [7: 0] | RCKCAL | Writes Calibration Values to RCKCAL Register Complete Calibration of 32KHz RC Oscillator |

### Clock Source Management Register - PMCR

PMCR - clock source management register

| PMCR: 0xF2 | Default: 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 enable control, 1 enabled, 0 disabled<br>WDT clock source selection, |
| [4] | WCLKS | 0 - Selects an internal 32MHz HFRC oscillator with 16 divisions<br>1 - Internal 32KHz LFRC oscillator |
| [5] | CLKSS | Main clock source selection control, select the clock source type, refer to the clock source selection unit<br>Minute |
| [6] | CLKFS | Main clock source frequency control, select the clock frequency type, please refer to the clock source selection<br>section |
| [7] | PMCE | The PMCR register changes the enable control bits.<br>Before changing other locations of PMCR, you must first set this bit, then at four<br>Set the value of the other bits in the cycle. |

### Master clock prescaler register - CLKPR

CLKPR - Master clock prescaler register

CLKPR: 0x61 Default: 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

| | | Clock prescaler selection bit | | | | |
|---|---|---|---|---|---|---|
| | | PS3 | PS2 | PS1 | PS0 | Frequency division parameters |
| [3: 0] | CLKPS | 0 | 0 | 0 | 0 | 1 |
| | | 0 | 0 | 0 | 1 | 2 |
| | | 0 | 0 | 1 | 0 | 4 |



**Page 41**



| | | | | | | |
|---|---|---|---|---|---|---|
| | | 0 | 0 | 1 | 1 | 8 (default configuration) |
| | | 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 | | | | Keep it |
| [4] | - | Keep not used | | | | |
| [5] | CKOEN0 | Set whether the system clock is output on the PB0 pin | | | | |
| [6] | CKOEN1 | Set whether the system clock is output on the PE5 pin | | | | |
| [7] | WCE | Clock prescaler changes clock control<br>Before changing the other bits of the CLKPR register, you must first set it separately<br>CKWEN is 1, and then in the next four system cycles, the other bits<br>Set up. After four cycles, CKWEN is automatically cleared. | | | | |

# Power management

## Overview

Sleep mode reduces system power by turning off the system clock and the clock module. LGT8FX8P offers very much
Flexible and diverse sleep mode and module controller, the user can be based on the application, to achieve the best low-power configuration.

LGT8FX8P does not automatically turn off analog function modules when entering Sleep mode, such as ADC, DAC, Comparator
(AC), low-voltage reset module (LVD), etc., the software needs to be in accordance with the application requirements, before entering the unwanted analog power
Can, and after the system wake up to restore the correct state.

The LGT8FX8P supports a variety of sleep modes, including the ADC-specific noise cancellation mode, which eliminates ADC conversion
The digital part of the process is interfering with the ADC power supply. In addition, the other are power control mode, divided into five kinds:

| Sleep mode | Function Description |
|---|---|
| Idle mode (IDLE) | Only close the kernel clock, other peripheral modules work properly, all valid interrupt source can be To wake up the kernel |
| Power saving mode (Save) | As with the DPS0 mode, the Save mode is compatible with the LGT8FX8D |
| Power-down mode (DPS0) | As with the Save mode, support wake-up sources include:<br>● All pin levels change<br>● Watchdog time wake-up<br>● Asynchronous mode TMR2 wake-up |
| Power down mode (DPS1) | Turn off all internal and external oscillators to support wake-up sources including:<br>● All pin external level changes<br>● External interrupt 0/1<br>● Watchdog timer operating at 32K LFRC |
| Power-down mode (DPS2) | Off the core power, the lowest power mode, support wake-up source include:<br>● External reset<br>● PORTD pin change<br>● LPRC timer wake-up (128ms / 256ms / 512ms / 1s)<br>It is important to note that the process of waking from DPS2 is the same as a power-on reset |

LGT8FX8P support deep sleep DPS2, in this mode, the system internal LDO in power-down state, the kernel register,
All peripheral controllers and SRAM are all powered down, and the data will not be maintained. FLASH memory unit also
Will be in the power-down state, so DPS2 mode to achieve the minimum system power consumption. Power-down mode can be via port D (PORTD)
Pin level change wake up, you can also choose 5 level wake-up. The timer for wake-up DPS2 is not supported for calibration,

Accuracy of about 15%, only suitable for low-precision time to wake up the application.

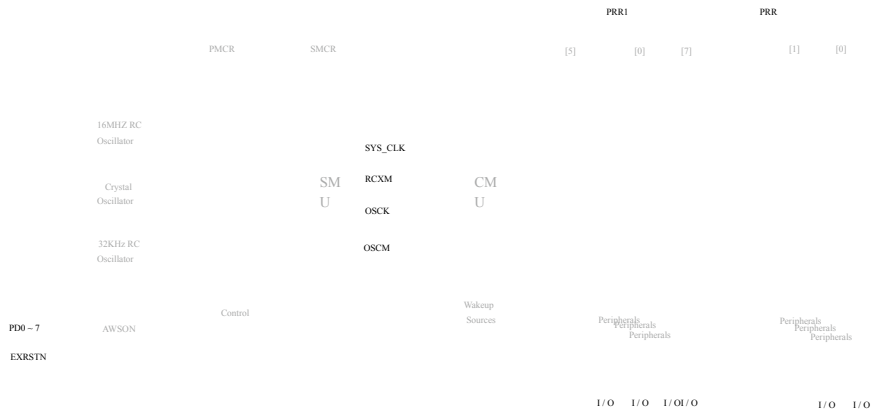
The system wake up from the DPS2 mode, will first open the LDO, the process and the same power process. The chip will perform complete
The power-on reset process starts, loads the configuration information, and then runs the program from the address pointed to by the reset vector.

In addition to DPS2 other than the mode, will not turn off the internal power supply, in the sleep process, all the register information and RAM
Data will not be lost. After waking up, the kernel continues from the last instruction before hibernation.





System power management diagram:




As shown in the figure above, the LGT8FX8P controls the entire configuration through the Sleep Mode Controller (SMU) and the Clock Management Unit (CMU)
Power consumption of the system. From the level of power savings, we can divide the power consumption into four levels:

The first level is controlled by the PRR register module operating clock, by turning off the clock without using the module, saving the system
The dynamic power consumption of the system. Under normal circumstances, this level can save power consumption is not obvious.

The second level is by switching the master clock source to the low frequency clock and turning off the unused clock source module as well as other simulations
Module, this mode can basically get a very considerable system running power and sleep power consumption.

The third level is achieved by allowing the system to enter power-down mode (DPS1), DPS1 mode LGT8FX8P can get polar
Standby power, wake-up from power-down mode, the software can read through the MCUSR register before the reset state.

The fourth level is power-down mode (DPS2), this mode will turn off the core power supply, can achieve the lowest system power consumption. because
Off the kernel power, this mode all the data will be lost. Wake up immediately after the implementation of a power-on reset process,
The system restarts from the reset vector.

## AWSON power management

Compared with the LGT8FX8D, power-down mode DPS2 is a new power mode. DPS2 mode is used for hibernation power consumption
Have higher requirements for applications. After entering the DPS2 mode, the system only maintains a static module (AWSON) in the working state,
Other circuits are in a completely power-down state.

AWSON module dedicated to the DPS2 mode is responsible for sleep and wake-up control, AWSON module mainly by the IO wake-up control
Logic and a low-power LPRC. The software can be implemented by the IOCWK register and the DPS2R register
AWSON control.

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 LPRC. Refer to the register definition section at the end of this section for specific information.

Before using DPS2 mode, the software sets IOCWK to enable the desired wakeup IO, or via the DPS2R register enable
LPRC and configure the timer wake-up period, and then enable the DPS2 mode through the DPS2EN bit in the DPS2R register. Set up finished
After the software, the software needs to set the DPS2 sleep mode through the SMCR register, and then execute the SLEEP instruction to enter the sleep.

**Page 44**



### Sleep mode and wake source

LGT8FX8P supports five sleep modes, the user can choose the appropriate sleep mode according to application requirements. SMCR deposit Device contains the sleep mode control settings, the implementation of SLEEP instructions, the kernel into sleep mode. To get more ideal Sleep power, it is recommended that the kernel into the sleep mode before the closure of all unused clock and analog modules. But need to note It is intended that some wake-up sources need to work clocks, and if you need to use such wake-up sources, keep the relevant clock source Working state.

Sleep mode and wake mode:

| Sleep mode | Valid clock | | | | | Wake source | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Inside nuclear Time bell | outer Set Time bell | ADC Time bell | different step Time bell | lead foot Electricity level change The | outer unit Interrupt Off / I | TWI To the site match With | TMR 2 ground in Off | ADC turn change Knot bundle | Look door dog overflow Out | outer Set Interrupt Off | PD Electricity level change The |
| 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 saving 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 you need to enter the above five kinds of sleep mode, SMCR in the SE bit must be set to enable sleep mode control. then Execute a SLEEP instruction. SM0 / 1/2 in SMCR is used to select a different sleep mode. Please refer to the specific information Test the following description.

When the MCU is in Sleep mode, if the wake source is active, the MCU will wake up after 4 cycles and continue execution instruction. If the interrupt remains active, the interrupt will also respond immediately to the interrupt service routine. If in SLEEP mode A system reset occurs and the MCU will be woken up and executed from the reset vector.

When the MCU is in Power / Off mode, the system can wake up via external interrupt INT0 / 1, wake up after the MCU will sleep before the location to continue.

### Idle mode (IDLE)

When SM2 ... 0 is set to 000, after the SLEEP instruction is executed, the MCU enters IDLE mode and IDLE mode will be turned off Out of the kernel working clock, in addition to other peripherals can work properly.

IDLE mode can be awakened by external interrupts and internal interrupts. If you do not need to use the comparator and ADC To wake up the source, it is recommended to turn it off.

IDLE mode because only the kernel to run off the clock, it can not be significantly reduced power consumption. IDLE mode , The kernel will also stop executing and fetch instructions, thus reducing the power consumption of the internal program FLASH.

But IDLE mode has a more flexible way to wake up, the user can reduce the system by reducing the main clock and do not need Of the module to get more ideal running power consumption.



**Page 45**



### ADC noise suppression mode

When SM2 ... 0 is set to 001, the MCU enters the ADC noise suppression mode after executing the SLEEP instruction. In this mode, The kernel and most peripherals will stop working, ADC, external interrupt, TWI address match, WDT, and work on asynchronous Clock mode 2 / timer 2 can work normally.

ADC noise always used to provide a good working environment for ADC conversion. Reduce the number of analog modules Conversion of high frequency interference. After entering this mode, the ADC will automatically start the sample conversion, the converted data is saved to the ADC data Register, the ADC conversion end interrupt will wake the MCU from ADC noise mode.

### Power saving mode (Save)

When SM2 ... 0 is set to 010, the MCU enters the Save mode after executing the SLEEP instruction. In this mode, the system Will close the working clock of all modules. This mode is due to the shutdown of all modules of the working clock, so only through the asynchronous Mode wake-up, external interrupt, TWI address match, and WDT operating in independent clock source mode can generate this mode Type of wake-up signal.

This mode can turn off all modules except the master clock source. In order to achieve a more ideal operating power, it is recommended to enter Before entering this mode, switch the system master clock to internal 32K RC or external 32KHz low frequency crystal, and then turn off the To the unused clock source and the analog module.

### Power down mode DPS0

When SM [2: 0] is set to 110, the MCU will enter DPS0 mode after executing the SLEEP instruction. After entering DPS0, In addition to the internal 32KHz RC, other clock sources are turned off. This mode can be woken up by an external interrupt INT0 / 1; Enabled WDT interrupt function, can also be achieved through the WDT wake-up time.

### Power down mode DPS1

When SM [2: 0] is set to 011, the MCU will enter DPS1 mode after executing the SLEEP instruction. After entering DPS1, All system clocks are turned off. This mode can be used to change the level of IO, watchdog wake up.

### Power down mode DPS2

Set SM [2: 0] to 111 and enable AWSON module through DPS2EN of DPSR2 register, execute SLEEP means So that after entering the DPS2 mode. After entering DPS2 mode, the system shuts down the core power supply. So the register and the RAM data Will be lost. The wake-up procedure from DSP2 is the same as the power-on reset process.

In the DPS2 mode, since the core voltage is turned off and the register information is lost, the control status of the port is all To return to the input state, all IO output driver and pull-up control will also be turned off.

### FLASH power control and fast wakeup

When the system is in SLEEP mode, the kernel will not continue to execute the instruction, then you can choose to turn off the FLASH power Source to achieve lower standby power consumption. This function can be implemented by the FPDEN bit control of the MCUCR register;

In power down mode, the system can use an external interrupt or WDT wake-up, in order to filter out the external signal may interfere, The internal wake-up circuit contains a configurable filter circuit that allows the user to select the appropriate filter width as needed. Filter The configuration of the 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: 0x00 | | | |
|---|---|---|---|---|
| Bits | SM2 | SM1 | SM0 | SE |

| R / W | - | R / W | R / W | R / W | R / W |
|---|---|---|---|---|---|

Bit definition

| | | |
|---|---|---|
| [0] | SE | Sleep mode enable control bit, set to 1 after executing the SLEEP instruction, the kernel Will go into sleep mode. The SE bit protects the system from accidentally entering sleep mode. Call Wake up, it is recommended to immediately clear the SE bit. |
| [3: 1] | SM | Sleep mode selection<br>SM2 SM1 SM0 Mode description<br>0 0 0 IDLE mode<br>0 0 1 ADC noise suppression mode<br>0 1 0 Save mode<br>0 1 1 DPS1 mode<br>1 1 0 DPS0 mode<br>1 1 1 DPS2 mode<br>Others Keep not used |
| [7: 4] | - | Keep not used |

### Power Saving Control Register - PRR

PRR - Power saving control register

PRR: 0x64 Default: 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 of the USART0 module |
| [2] | PRSPI | Set to 1 to turn off the SPI module clock |
| [3] | PRTIM1 | Set to 1 to turn off the Timer of Timer / Event Counter 1 |
| - | - | Keep not used |
| [5] | PRTIM0 | Set to 1 to turn off the Timer of Timer / Event Counter 0 |
| [6] | PRTIM2 | Set to 1 to turn off the Timer of Timer / Event Counter 2 |
| [7] | PRTWI | Set to 1 to turn off the TWI module clock |





### Power Saving Control Register - PRR1

PRR1 - Power saving control register 1

PRR1: 0x65 Default: 0x00

| PRR1 | | PRWDT | - | PRTIM3 | PREFL | PRPCI | - |
|---|---|---|---|---|---|---|---|
| R / W | | R / W | - | R / W | R / W | R / W | - |

Bit definition

| | | |
|---|---|---|
| [0] | - | Keep not used |
| [1] | PRPCI | Set to 1, turn off external pin change and external interrupt module clock |
| [2] | PREFL | Set to 1, turn off the FLASH controller interface clock |
| [3] | PRTIM3 | Set to 1 to turn off the TMR3 controller clock |
| [4] | - | Keep not used |
| [5] | PRWDT | Set to 1 to turn off the WDT counter clock |
| [7: 6] | - | Keep not used |

### MCU Control Register - MCUCR

MCUCR - MCU control register

MCUCR: 0x35 (0x55) Default: 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 | MCUCR update enable bit, before updating MCUCR, you need to set this bit first,<br>The update to the MCUCR register is then completed in 6 cycles |
| [1] | IVSEL | Interrupt vector selection bit, this bit is set after the interrupt vector address will be based on IVBASE<br>The value of the register is mapped to the new address |
| [2] | IFAIL | System Configuration Bit Load Failure Flag,<br>0 = The configuration information is verified<br>1 = configuration information failed to load |
| [3] | IRLD | Writing 1 will reload the system configuration information |
| [4] | PUD | Global pull-up disabled bit<br>0 = Enable global pull-up control<br>1 = Turn off all IO pull-up resistors |
| [5] | EXRFD | External reset filter disable bit<br>0 = Enable external reset (190us) digital filter<br>1 = Disable external reset of the digital filter circuit |
| [6] | FPDEN | Flash Power / down enable control<br>0: After the system SLEEP, the FLASH remains powered on<br>1: System SLEEP after the FLASH power |
| [7] | FWKEN | Fast wake-up mode enable control, only valid for Power / Off mode<br>0: 260us filter delay<br>1: 32us filter delay |





### PD Group Level Change Wake Control Register - IOCWK

IOCWK - PD group level change wake-up control register

IOCWK: 0xAE Default: 0x00

| Bits | IOCD7 | IOCD6 | IOCD5 | IOCD1 | IOCD1 | IOCD1 | IOCD1 | |
|---|---|---|---|---|---|---|---|---|
| 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 corresponding bit to enable pin change of wakeup function of PD group IO

### DPS2 mode control register - DPS2R

DPS2R - DPS2 mode control register

DPS2R: 0xAF Default: 0x00

| Bits | - | - | - | - | DPS2E | LPRCE | TOS1 | TOS0 |
|---|---|---|---|---|---|---|---|---|
| R / W | - | - | - | - | R / W | R / W | R / W | R / W |

Bit definition

| | | |
|---|---|---|
| [1: 0] | TOS | LPRC timer wakeup settings:<br>00 = 128ms<br>01 = 256ms<br>10 = 512ms<br>11 = 1s |
| [2] | LPRCE | LPRC enable control<br>0 = disable LPRC timer<br>1 = Enable LPRC timer |
| [3] | DPS2E | DPS2 mode enable control bit<br>0 = disable DPS2 mode<br>1 = Enable DPS2 mode |

[7: 4] - Keep it





# System control and reset

## Overview

After the system is reset, all I / O registers are set to their initial values, starting with the reset vector
carried out. On the interrupt vector address of the LGT8FX8P, an RJMP - relative jump instruction must be used to jump to the reset process sequence. If the program is not used to interrupt, no interrupt source is enabled, the interrupt vector will not be used, the interrupt vector area It can be used to store the user's program code.

When the reset is active, all I / O ports immediately enter their initial state. Most I / O initialization states are input
And turn off the internal pull-up resistor. I / O with analog input function is also initialized to digital I / O function.

When the reset becomes inactive, the LGT8FX8P internal timer counter starts to be used for broadening the reset. Widen the reset The width of the signal is used to ensure that the power supply in the system and the clock and other modules enter a stable state.

## Reset source

LGT8FX8P supports a total of six reset sources:

- Power-on reset: When the operating voltage of the system is low, the reset value of the internal POR module is valid.
- External reset: the external reset pin on the chip on a certain width of the low level pulse, external reset is valid.
- Watchdog reset: When the watchdog module is enabled, the system will reset if the watchdog timer times out.
- Low voltage reset: LGT8FX8P has a low voltage detection module (LVD), when the system power supply is lower than the LVD When the reset threshold is set, the MCU will also be reset.
- Software reset: LGT8FX8P internal has a dedicated software to trigger the reset register, the user can through this The register resets the MCU at any time.
- OCD Reset: OCD Reset is issued by the debugger module for direct resetting the MCU core.

Reset system structure diagram:

Software Reset

OCD Reset

Clock Sources



**Page 50**



### Power on reset

The power-on reset signal is generated by an internal voltage detection circuit. When the system power supply (VCC) is lower than the detection threshold, power up The reset signal is valid. Refer to the Electrical Parameters section for the detection threshold for power-on reset.

Power-on reset circuit to ensure that the chip in the power process in a reset state, the chip after power from a known Steady state to start running. Power-on reset signal will be the chip internal expansion of the counter to ensure that after the power of the internal An analog module, such as an RC oscillator, can enter a stable operating state.

VTH

VCC

200us 200us

VPOR

### External reset

A low level is applied to the external reset pin (RSTN), and the external reset is active immediately. The width of the low level is greater than one Minimum reset pulse width requirement. External reset is asynchronous reset, even if the chip does not have clock operation, external reset is still possible Enough to reset the chip. The external reset pin of the LGT8FX8P can also be used as a general purpose I / O. Power on the chip After default, as an external reset function. The user can configure the external reset function of the pin by registering the register But can be used as a normal I / O. For details, refer to the description section of the IOCR register.

> 200us

Vrst

RESETN

10us

Extend

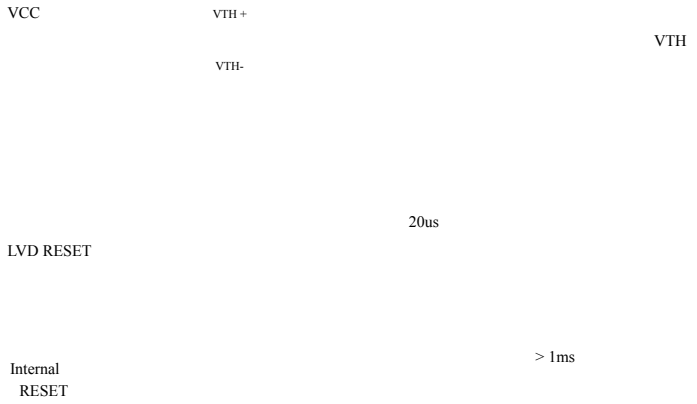
Internal RESET

### Low voltage detection (LVD) reset

The LGT8FX8P contains a programmable low voltage detection (LVD) circuit internally. LVD is also the detection of VCC voltage changes, However, unlike the power-on reset, the LVD can select the threshold for the detection voltage. The user can operate directly through the VDTCR The registers are selected between the different voltage thresholds. LVD voltage detection circuit with ± 10mV ~ ± 50mV hysteresis characteristics, Used to filter out the VCC voltage jitter. When LVD is enabled, if VCC's voltage drops to the set reset threshold, LVD is complex Bit will be effective immediately. When VCC is increased above the reset threshold, the internal reset expansion circuit starts and the reset continues to widen At least 1 millisecond.

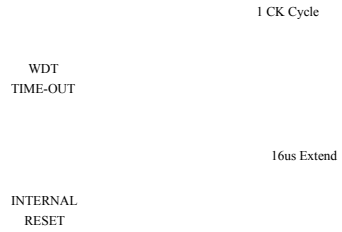
**Page 51**






### Watchdog reset

When the watchdog timer overflows, the watchdog system reset function is enabled and will immediately generate a cycle of the system Reset signal. The watchdog reset signal is also broadened by the internal delay counter. Watchdog controller for detailed operation, Please refer to the detailed section below.




### Software reset, OCD reset

Software reset is the user through the operation of the VDTCR register sixth trigger, software reset timing and watchdog reset Completely similar. The interior will reset the signal by 16us.

OCD reset generated by the internal debugger unit, OCD reset is generally controlled by the debugger, the user software can not Trigger OCD reset.



**Page 52**

### Watchdog timer

- Clock optional internal 32KHz RC or internal 32MHz RC 16-way (2MHz)
- Supports interrupt mode, reset mode and reset interrupt mode
- The timer can be up to 8 seconds

The LGT8FX8P contains an enhanced watchdog timer (WDT) module internally. The WDT timer can operate clock
Is the internal 32KHz RC oscillator, it can also be internal 32MHz RC oscillator 16 frequency. WDT counter overflow
After the output, you can output an interrupt or a system reset signal. In normal use, the need for software to implement one
WDR - The watchdog timer reset instruction restarts the counter before overflowing. If the system does not have to execute WDR
Instruction, the WDT will generate an interrupt or a system reset.

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


In interrupt mode, an interrupt request signal is generated after a WDT overflow. You can use this interrupt as a break
Sleep mode wake-up signal can also be used as a general system timer. For example, you can use this break limit
The execution time of an operation, and the termination of a current task in the overflow. In system reset mode, the WDT is counting
A system reset signal is generated immediately after the overflow. The most typical use is to prevent the system from crashing or running away.
The third mode, is to reset the interrupt mode, combined with the interrupt and reset two functions. The system will first respond to the WDT
Interrupt function, exit WDT interrupt reset program, immediately switch to reset mode. This feature can support the complex
Bit before the preservation of some of the more critical parameter information.

In order to prevent the WDT from being accidentally disabled, the operation to turn off the WDT must be performed at a strictly defined timing. the following
The code describes how to turn off the watchdog timer. The following example assumes that the interrupt has been disabled, so the entire operation flow
The process will not be interrupted.



Example of the watchdog enable and shutdown operation:

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 language 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]

If the WDT is accidentally enabled, such as the program running, the chip will be reset, but the WDT is still enabled. If used

The WDT is not processed in the user code, which will cause a cyclic reset. To avoid this, it is recommended that the user software initialize the program

Clear the watchdog reset flag (WDRF) and the WDE control bit.



**Page 54**



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 language 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】

It is recommended to reset the watchdog timer before changing the WDP configuration bits. Because it is possible to change the WDP bit to a relatively small time-out period Will cause the watchdog to reset on time.





## Register definition

### Low Voltage Detection (LVD) Control Register - VDTCR

VDTCR - LVD control register

VDTCR: 0x62 Default: 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 voltage detection threshold configuration bit<br>000 = 1.8V<br>001 = 2.2V<br>010 = 2.5V<br>011 = 2.9V<br>100 = 3.2V<br>101 = 3.6V<br>110 = 4.0V<br>111 = 4.4V |
| [5] | - | Keep not used |
| [6] | SWR | Soft reset enable bit, this bit is cleared to generate a software reset |
| [7] | WCE | VDTCR value change enable bit<br>Before changing the value of the VDTCR register, the user must first write this bit to 1,<br>Change the value of the other bits of VDTCR during the next six clock cycles. Four weeks<br>WCE is automatically cleared and the update operation of the VDTCR register is invalid. |

### IO Function Multiplexed Register - PMX2

PMX2 - IO function multiplexing register

PMX2: 0xF0 Default: 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 | PC6 pin defaults to the reset function, setting this bit to 1 will disable the external reset function<br>Can, 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 simulation<br>Input function, this pin can be used as a GPIO |
| 2 | XIEN | External clock input enable control |
| 4: 3 | - | Keep not used |
| 5 | STSC0 | Low speed crystal start control |
| 6 | STSC1 | High speed crystal start control |
| 7 | WCE | IOCR value changes enable bit<br>Before changing the value of the IOCR register, the user must first write this bit to 1 |





After the next 6 clock cycles, change the value of the IOCR other bits. Four cycles
After WCE is automatically cleared, the IOCR register update operation is invalid.

### MCU Status Register - MCUSR

MCUSR - IO special function control register

MCUSR: 0x34 (0x54) Default: 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 clear |
| [1] | EXTRF | External reset flag, power-on reset is automatically cleared, or write 0 cleared |
| [2] | BORF | Low voltage detection reset, power-on reset automatically cleared, or write 0 clear |
| [3] | WDRF | Watchdog reset flag, power-on reset automatically cleared, or write 0 clear |
| [4] | OCDRF | OCD debugger reset flag, power-on reset is automatically cleared, or write 0 cleared |
| [5] | PDRF | Wake up from the Power / off mode. Refer to the Power Management section for details. |
| [6] | - | Keep not used |
| [7] | SWDD | SWD interface disable bit. Writing a 1 closes the SWD interface.<br>After the SWD interface is shut down, debugging and ISP operations can not be performed. If the user process<br>The SWD interface is closed in the sequence, and the RESET side can be pulled down during power-up<br>Prohibits the operation of the internal program, and then debug and ISP operation. SWD access<br>After the port is closed, the two I / O interfaces occupied by SWD can be used as general purpose I / O.<br>To avoid misuse of SWDD, the user needs to update the SWDD bit for the first time<br>After the four cycles to write SWDD to take effect. |

[Tips]:

In order to use the reset flag information more accurately and effectively, it is advisable to read the reset flag as soon as possible before the program is initialized
Clear it.

### Watchdog Control Status Register - WDTCSR

WDTCSR - WDT control and status register

Address: 0x60 Default: 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.<br>The WDIF bit is set when the WDT is operating in interrupt mode and overflow occurs. When in the WDT<br>When the enable bit WDIE is "1" and the global interrupt is set, the WDT interrupt is generated. carried out<br>WDIF bit is cleared when WDT is interrupted and "1" for WDIF bit can also be cleared. |
| [6] | WDIE | WDT interrupt enable control bit.<br>WDT interrupt is enabled when the WDIE bit is set to "1" and the global interrupt is set. |





When the WDIE bit is set to "0", the WDT interrupt is disabled.
The WDIE bit together with the WDE bit determines the watchdog's operating mode, as shown in the following table.

| WDE | WDIE | mode | After overflow action |
|---|---|---|---|
| 0 | 0 | stop | no |
| 0 | 1 | Interrupt mode | Interrupted |
| 1 | 0 | Reset mode | Reset |
| 1 | 1 | Interrupt reset mode | Reset after reset |

| | | |
|---|---|---|
| | | WDT prescaler factor selection control 3rd bit. |
| [5] | WDP3 | WDP [3] and WDP [2: 0] make up the WDT prescaler factor selection bit WDP [3: 0] |
| | | Set the WDT overflow period. |
| | | WDT off enable control bit. |
| [4] | WDTOE | When the WDE bit is cleared, the WDTOE bit is set, otherwise the WDT will not be turned off.<br>When the WDTOE bit is set, the hardware clears the WDTOE bit after 4 clock cycles. |
| [3] | WDE | WDT enable control bit.<br>WDT is enabled when the WDE bit is set to "1". When the WDE bit is set to "0"<br>WDT is disabled.<br>WDE can only be cleared if the WDTOE bit is set. To close the already enabled<br>WDT, must follow the following timing:<br>1. Set both the WDTOE and WDE bits, even if WDE has been set and is turned off<br>A "1" must also be written to the WDE bit before the operation begins;<br>2. Write "0" to the WDE bit for the next four clock cycles. This will close<br>WDT.<br>The WDT reset system flag is set when the WDE bit is "1" and the WDT overflow resets the system<br>WDRF (located in the MCUSR register). Set when the WDRF bit is in the set state<br>Bit WDE bit. Therefore, to clear the WDE bit, the WDRF bit must be cleared first. |
| [2: 0] | WDP | WDT prescaler factor selection control.<br>Used to set the WDT overflow period. It is recommended to change the WDP when the WDT is not counting<br>Value, changing the value of WDP during counting will result in an unpredictable WDT overflow. |

Watchdog Prescaler Selection List:

| WDP3 | WDP2 | WDP1 | WDP0 | Watchdog timer<br>Number of overflow cycles | 32KHz<br>clock | 2MHz<br>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 |

**Page 58**



| | | | | |
|---|---|---|---|---|
| 1 | 0 | 1 | 0 | Keep not used |
| 1 | 0 | 1 | 1 | |
| 1 | 1 | 0 | 0 | |
| 1 | 1 | 0 | 1 | |
| 1 | 1 | 1 | 0 | |
| 1 | 1 | 1 | 1 | |



**Page 59**

## Interrupt and interrupt vector

- 28 interrupt sources
- Programmable vector start address

LGT8F88P / 168P / 328P interrupt resources are basically the same, the main difference is: LGT8F88P interrupt vector Is an instruction word (16 bits), and the LGT8F168P / 328P interrupt vector is two instruction words.

### LGT8F88P interrupt vector list

LGT8F88P interrupt vector list:

| Numbering | Vector address | Interrupt source signal | Interrupt source description |
|---|---|---|---|
| 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 |
| 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 compare match A interrupt |
| 9 | 0x0008 | TC2 COMPB | Timer 2 compare match B interrupt |
| 10 | 0x0009 | TC2 OVF | Timer 2 overflow interrupt |
| 11 | 0x000A | TC1 CAPT | Timer 1 input capture interrupt |
| 12 | 0x000B | TC1 COMPA | Timer 1 compare match A interrupt |
| 13 | 0x000C | TC1 COMPB | Timer 1 compare match B interrupt |
| 14 | 0x000D | TC1 OVF | Timer 1 overflow interrupt |
| 15 | 0x000E | TC0 COMPA | Timer 0 compare match A interrupt |
| 16 | 0x000F | TC0 COMPB | Timer 0 compare match B interrupt |
| 17 | 0x0010 | TC0 OVF | Timer 0 overflow interrupt |
| 18 | 0x0011 | SPI STC | SPI serial transmission ends interrupt |
| 19 | 0x0012 | USART RXC | The USART receives an end interrupt |
| 20 | 0x0013 | USART UDRE | The USART data register is empty |
| twenty one | 0x0014 | USART TXC | The USART sends an end interrupt |
| twenty two | 0x0015 | ADC | ADC conversion end interrupt |
| twenty three | 0x0016 | EE_RDY | EEPROM ready to interrupt |
| twenty four | 0x0017 | ANA_COMP | Analog Comparator 0 interrupt |
| 25 | 0x0018 | TWI | Two-wire serial interface is interrupted |
| 26 | 0x0019 | ANA_COMP1 | Analog Comparator 1 is interrupted |
| 27 | 0x001A | - | Keep it |
| 28 | 0x001B | PCI3 | Pin level interrupt 3 |
| 29 | 0x001C | PCI4 | Pin level interrupt 4 |
| 30 | 0x001D | TC3_INT | Timer 3 is interrupted |



### LGT8F168P / 328P interrupt vector list

LGT8F168P / 328P Interrupt Vector List:

| Numbering | Vector address | Interrupt source signal | Interrupt source description |
|---|---|---|---|
| 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 |
| 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 compare match A interrupt |
| 9 | 0x0010 | TC2 COMPB | Timer 2 compare match B interrupt |
| 10 | 0x0012 | TC2 OVF | Timer 2 overflow interrupt |
| 11 | 0x0014 | TC1 CAPT | Timer 1 input capture interrupt |
| 12 | 0x0016 | TC1 COMPA | Timer 1 compare match A interrupt |
| 13 | 0x0018 | TC1 COMPB | Timer 1 compare match B interrupt |
| 14 | 0x001A | TC1 OVF | Timer 1 overflow interrupt |
| 15 | 0x001C | TC0 COMPA | Timer 0 compare match A interrupt |
| 16 | 0x001E | TC0 COMPB | Timer 0 compare match B interrupt |
| 17 | 0x0020 | TC0 OVF | Timer 0 overflow interrupt |
| 18 | 0x0022 | SPI STC | SPI serial transmission ends interrupt |
| 19 | 0x0024 | USART RXC | The USART receives an end interrupt |
| 20 | 0x0026 | USART UDRE | The USART data register is empty |
| twenty one | 0x0028 | USART TXC | The USART sends an end interrupt |
| twenty two | 0x002A | ADC | ADC conversion end interrupt |
| twenty three | 0x002C | EE_RDY | EEPROM ready to interrupt |
| twenty four | 0x002E | ANA_COMP | Analog comparator interrupt |
| 25 | 0x0030 | TWI | Two-wire serial interface is interrupted |
| 26 | 0x0032 | ANA_COMP1 | Analog Comparator 1 is interrupted |
| 27 | 0x0034 | - | Keep it |
| 28 | 0x0036 | PCI3 | Pin level interrupt 3 |
| 29 | 0x0038 | PCI4 | Pin level interrupt 4 |
| 30 | 0x003A | TC3_INT | Timer 3 is interrupted |

The reset vector for LGT8FX8P is executed from address 0x0000. In addition to the reset vector, other vector addresses are available The IVSEL and IVBASE registers in the MCUCR register are redirected to the 512-byte aligned start address.



## Interrupt vector processing

The following code only LGT8F88P as an example, used to explain the reset and interrupt vector programming, for reference only:

Assembly code example - LGT8F88P

| address | Code | Description |
|---|---|---|
| 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 | The watchdog timer is interrupted |
| 0x007 | RJMP TIM2_COMPA | Timer 2 compare match group A interrupt |
| 0x008 | RJMP TIM2_COMPB | Timer 2 compare match group B interrupt |
| 0x009 | RJMP TIM2_OVF | Timer 2 overflow interrupt |
| 0x00A | RJMP TIM1_CAPT | |

| | | |
|---|---|---|
| | | Timer 1 traps the interrupt |
| 0x00B | RJMP TIM1_COMPA | Timer 1 compare match A group interrupt |
| 0x00C | RJMP TIM1_COMPB | Timer 1 compare match group B interrupt |
| 0x00D | RJMP TIM1_OVFR | Timer 1 overflow interrupt |
| 0x00E | RJMP TIM0_COMPA | Timer 0 compare match group A interrupt |
| 0x00F | RJMP TIM0_COMPB | Timer 0 compare match group B interrupt |
| 0x010 | RJMP TIM0_OVF | Timer 0 overflow interrupt |
| 0x011 | RJMP SPI_STC | SPI transfer complete interrupt |
| 0x012 | RJMP USART_RXC | USART reception complete interrupt |
| 0x013 | RJMP USART_UDRE | The USART data register is empty |
| 0x014 | RJMP USART_TXC | The USART sends an interrupt |
| 0x015 | RJMP ADC | ADC conversion complete interrupt |
| 0x016 | RJMP EE_RDY | The EEPROM controller is ready to interrupt |
| 0x017 | RJMP ANA_COMP | Comparator interrupt |
| 0x018 | RJMP TWI | TWI controller interrupt |
| 0x019 | NOP | Keep address |
| 0x01A | NOP | Keep address |
| 0x01B | RJMP PCI3 | Pin level change interrupt 3 |
| ; | | |
| 0x01C (RESET :) | LDI r16, high (RAMEND) | The main program starts |
| 0x01D | OUT SPH, r16 | Set the stack pointer to the top address of the RAM |
| 0x01E | LDI r16, low (RAMEND) | |
| 0x01F | OUT SPL, r16 | |
| 0x020 | SEI | Enable global interrupts |
| 0x021 | ... | |





## Register definition

### MCU Control Register - MCUCR

MCUCR - MCU control register

MCUCR: 0x35 (0x55) Default: 0x00

| MCUCR | FWKEN | FPDEN | EXRFD | | PUD | IRLD | IFAIL | IVSEL | WCE |
|---|---|---|---|---|---|---|---|---|---|
| R / W | R / W | R / W | R / W | R / W | R / W | W / O | R / O | R / W | R / W |

Bit definition

| | | |
|---|---|---|
| [0] | WCE | MCUCR update enable bit, before updating MCUCR, you need to set this bit first,<br>The update to the MCUCR register is then completed in 6 cycles |
| [1] | IVSEL | Interrupt vector selection bit, this bit is set after the interrupt vector address will be based on IVBASE<br>The value of the register is mapped to the new address |
| [2] | IFAIL | System Configuration Bit Load Failure Flag,<br>0 = The configuration information is verified<br>1 = configuration information failed to load |
| [3] | IRLD | Writing 1 will reload the system configuration information |
| [4] | PUD | Global pull-up disabled bit<br>0 = Enable global pull-up control<br>1 = Turn off all IO pull-up resistors |
| [5] | EXRFD | External reset filter disable bit<br>0 = Enable external reset (190us) digital filter<br>1 = Disable external reset of the digital filter circuit |
| [6] | FPDEN | Flash Power / down enable control<br>0: After the system SLEEP, the FLASH remains powered on<br>1: System SLEEP after the FLASH power |
| | | Fast wake-up mode enable control, only valid for Power / Off mode |

| | | |
|---|---|---|
| [7] | FWKEN | 0: 260us filter delay<br>1: 32us filter delay |

Interrupt Vector Base Address Register - IVBASE

IVBASE - interrupt vector base address register

| IVBASE: 0x75 | Default: 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 based on IVBASE<br>The address is remapped on a 512-byte page.<br>The mapped interrupt vector base addresses are: (IVBASE << 8) + Corresponding in Table 1<br>Vector address |





## External interrupt

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

### Overview

The external interrupt is triggered by the INT0 and INT1 pins. As long as the external interrupt is enabled, even if the two pins are configured for output
Can also trigger an interrupt. This can be used to generate software interrupts. The external interrupt can be triggered by a rising edge, a falling edge, or a low level,
It is configured by external interrupt control register EICRA. When an external interrupt is enabled and configured as a level trigger (only INT0 and
INT1 pin), the interrupt will always be generated as long as the pin level is low. INT0 and INT1 pins are rising or falling
The interrupt requires an IO clock to operate normally, while the INT0 and INT1 pin low-level interrupt are detected asynchronously.
In addition to the idle mode, the IO clocks in other sleep modes are stopped. Therefore, these two external interrupts can be used as
Wake-up source in other sleep modes except for Idle mode.

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

### Register definition

Register list

| register | address | Defaults | 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: 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 | - | Keep it. |
| 3 | ISC11 | INT1 pin interrupt trigger mode control bit high. |
| 2 | ISC10 | INT1 pin interrupt trigger mode control bit low.<br>When the global interrupt is set and the corresponding interrupt mask control bit of the GICR register is set,<br>The interrupt 1 is triggered by the INT1 pin. The trigger mode of the interrupt is described in the table. On the edge<br>The MCU first samples the level on the INT1 pin before detecting. If selected edge trigger<br>Mode or level change trigger mode, then the duration is greater than one system clock cycle<br>The pulse will trigger an interrupt, too short pulse can not guarantee the trigger interrupt. If you choose low power |





| | | |
|---|---|---|
| | | Flat trigger mode, then the low level must be maintained until the current instruction execution is completed before the trigger Off. |
| 1 | ISC01 | INT0 pin interrupt trigger mode control bit high. |
| 0 | ISC00 | INT0 pin interrupt trigger mode control bit low.<br>When the global interrupt is set and the corresponding interrupt mask control bit of the GICR register is set,<br>The interrupt 0 is activated by the INT0 pin. The trigger mode of the interrupt is described in the table. On the edge<br>The MCU first samples the level on the INT0 pin before detecting. If selected edge trigger<br>Mode or level change trigger mode, then the duration is greater than one system clock cycle<br>The pulse will trigger an interrupt, too short pulse can not guarantee the trigger interrupt. If you choose low power<br>Flat trigger mode, then the low level must be maintained until the current instruction execution is completed before the trigger Off. |

External interrupt 1 trigger mode see table below.

External interrupt 1 trigger mode control

| ISC1 [1: 0] | description |
|---|---|
| 0 | External pin INT1 low trigger |
| 1 | Trigger on the rising or falling edge of external pin INT1 |
| 2 | The falling edge of external pin INT1 is triggered |
| 3 | The rising edge of the external pin INT1 is triggered |

External interrupt 0 trigger mode see table below.

External interrupt 0 Trigger mode control

| ISC0 [1: 0] | description |
|---|---|
| 0 | External pin INT0 low level trigger |
| 1 | The rising edge of the external pin INT0 or the falling edge triggers |
| 2 | The falling edge of the external pin INT0 is triggered |
| 3 | The rising edge of the external pin INT0 is triggered |

### External Interrupt Mask Register - EIMSK

EIMSK - External interrupt mask register

Address: 0x3D Default: 0x00

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | - | - | - | - | - | - | INT1 | INT0 |
| R / W | - | - | - | - | - | - | R / W | R / W |

| Bit | Name | description |
|---|---|---|
| 7: 2 | - | Keep it |
| 1 | INT1 | External pin 1 interrupt enable control bit.<br>When the INT1 bit is set to "1", the global interrupt is set and the external pin 1 interrupt is enabled<br>Can wake up function enabled. Even if the INT1 pin is configured as an output, as long as the pin is electrically connected<br>The corresponding change has occurred and the interrupt will occur.<br>When the INT1 bit is set to "0", the external pin 1 interrupt is disabled and the wake-up function is also enabled<br>Forbidden. |

| | | |
|---|---|---|
| 0 | INT0 | External pin 0 interrupt enable control bit.<br>When the INT0 bit is set to "1", the global interrupt is set and the external pin 0 interrupt is enabled<br>Can wake up function enabled. Even if the INT0 pin is configured as an output, as long as the pin is electrically connected<br>The corresponding change has occurred and the interrupt will occur.<br>When the INT0 bit is set to "0", the external pin 0 interrupt is disabled and the wake-up function is also enabled<br>Forbidden. |

External Interrupt Flag Register - EIFR

EIFR - External interrupt flag register

Address: 0x3C Default: 0x00

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | - | - | - | - | - | - | INTF1 | INTF0 |
| R / W | - | - | - | - | - | - | R / W | R / W |

| Bit | Name | Description |
|---|---|---|
| 7: 2 | - | Keep it. |
| 1 | INTF1 | external pin 1 interrupt flag bit.<br>INTF1 is set when edge trigger external pin 1 is interrupted. When the low level is triggered outside<br>When the pin 1 is interrupted, the INTF1 bit is not set. If the external pin 1 is interrupted at this time<br>The INT1EN bit is set to "1" and the global interrupt flag is set, which will be generated in external pin 1<br>Off. INTF1 will be cleared automatically or written to the INTF1 bit when this interrupt service routine is executed<br>"1" can also be cleared to this bit. |
| 0 | INTF0 | External pin 0 interrupt flag.<br>INTF0 is set when edge-triggered external pin 0 is interrupted. When the low level is triggered outside<br>When the pin 0 is interrupted, the INTF0 bit is not set. If the external pin 0 interrupt at this time<br>If the INT0EN bit is set to "1" and the global interrupt flag is set, an external pin 0 is generated<br>Off. INTF0 will be cleared automatically or written to the INTF0 bit when this interrupt service routine is executed<br>"1" can also be cleared to this bit. |

# Universal Programmable Port (GPIO)

## Overview

All MCUs based on the LGT8XM core family have I / O port read-to-write functions. This means that one
The status of the ports can be changed individually using the SBI and CBI instructions without affecting any other I / O. The same, change
Change the direction of a port or control its pull-up resistor can also be the case.

Most of the I / O of the LGT8FX8P has symmetrical drive characteristics that can drive and absorb large currents. I / O has two
Level drive capability, the user can control the drive capability of each group of I / O. I / O drive capability can directly drive some LEDs.

LGT8FX8P most of the I / O can drive up to 30mA of current, can be directly used to drive segment code LED.

All I / O VCC and GND directly have independent ESD protection diodes, designed to withstand at least up to
5000V ESD pulse.

I / O equivalent circuit diagram:

Rpu

Pxn

Logic

Cpin

All of the following registers in this chapter are described in a uniform way. The lowercase "x" indicates the letter number of the port, the lower case "n"
Indicates the bit number in the port. However, when using the port register in the program, you must use the exact register name. such as
PORTB3, which represents the third bit of PORTB, where it is unified with PORTxn said. A detailed definition of I / O related registers,
Please refer to the Register Description section.

Each port is assigned three I / O register spaces, which are: Port Data Output Register (PORTx), Port Direction
Register (DDRx), port data input register (PINx). The port data input register is a read-only register. Data output
The register and port direction registers are readable or rewritten. The PUD bit in the MCUCR register is used to control all I / O
Pull-up resistor, when the PUD bit is 1, will disable the I / O pull-up resistor.

Most I / O in addition to the general input / output function, will be reused for other peripheral functions. Specific reusable work
Please refer to the chapter on port function reuse.

It should be noted that enabling multiple port reuse does not affect these ports as digital I / O. and
Some of the alternate functions may also need to control the input / output direction of the port via the I / O register. The specific settings will be in each
A description of the documentation for the multiplex module.



## Universal input / output port

As a general purpose I / O, the port is a bidirectional drive I / O port with an internal programmable pull-up.

The following figure shows the equivalent circuit diagram of the general purpose I / O port:

PUD


| | | | |
|---|---|---|---|
| PUD: PULLUP DISABLE | | WDx: | WRITE DDRx |
| SLEEP: SLEEP CONTROL | | RDx: | READ DDRx |
| IO_CLK: | I / O CLOCK | 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. Where DDxn is available for use via DDRx Register access, PORTxn can be accessed via the PORTx register, and PINxn can be accessed via 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 assigned Set to an output port. If DDxn is set to 0, Pxn is configured as an input port.






If the PORTxn bit is written 1 and the port is configured as an input port, the pull-up resistor for this port is asserted. If you want to disable the pull-up resistor on the port, PORTxn must be either 0 or configured as an output port.

The reset status of the port is the input state and the pull-up resistor is invalid.

PORTxn is set to 1, and the 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.

### Input / output switch

When the I / O state is between the tri-state ([DDxn, PORTxn]) = 0b00) and the output high ([DDxn, PORTxn] = 0b11)
In case of time, there will be a port pull-up or output to a low intermediate state. In general, the pull-up resistor can be accepted,
Because in a high resistance environment, the drive between the high and the difference between the pull is not important. If this is not the case, you can pass
The PUD bit in the MCUCR register is turned off so the pull-up function of the port.

Likewise, the same problem arises when switching between the pull-up enable input and the output low. The user must make
([DDxn, PORTxn] = 0b00) or the output high ([DDxn, PORTxn] = 0b11) as the intermediate state.

Port driver configuration table:

| DDxn | PORTxn | PUD | Port state | pull up | Function Description |
|---|---|---|---|---|---|
| 0 | 0 | X | enter | Forbidden | Tri-state |
| 0 | 1 | 0 | enter | Enable | Enter + internal pull-up mode |
| 0 | 1 | 1 | enter | Forbidden | Tri-state |
| 1 | 0 | X | Output | Forbidden | Output low (fan-in) |
| 1 | 1 | X | Output | Forbidden | Output high (fanout) |

### Read port value

Regardless of how the port direction bit DDxn is set, the current status of the port can be read via the PINxn register bit.
To avoid directly reading the metastable state generated by the port, the PINxn register bit is the result of the port passing through a synchronizer. 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 presence of the synchronizer, with a delay of up to one half of the system cycle.

We assume that the system cycle from the system clock on the first falling edge of the latch when the clock is low when the data,
When the clock is high, the data passes through the latch, as shown in the shaded area above. When the clock is low, the port data is locked

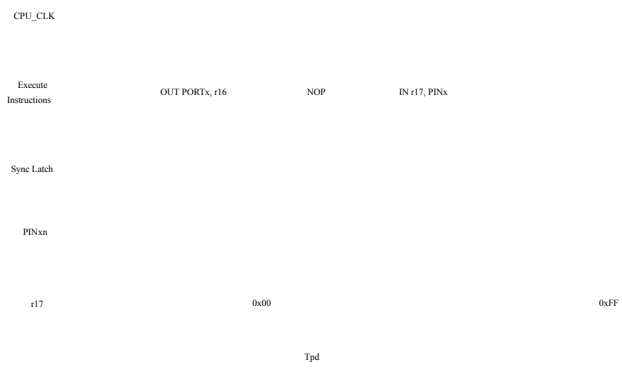






And the rising edge of the next clock is registered into the PINxn register. In the figure above, Tpd, max and Tpd, min
The maximum and minimum delays for port data are divided into 1.5 cycles and 0.5 cycles.

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




The following code shows how to set port 0 to pin 0/1 to be high, 2/3 is low, define pins 4 to 7 for input and
The pull-up resistors of pins 6, 7 are enabled. And then the value of the pin back to read the general working register, according to the previous description,
A NOP instruction is inserted directly on the pin's output and input.

Assembly code

```
; Define Pull-ups and set up 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
/ * Define pull-ups and set up 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;
```



**Page 70**



### Input enable and sleep control

From the I / O equivalent circuit diagram we can see that the digital input can be clamped to the ground under the control of the SLEEP signal Level. The SLEEP signal is controlled by the MCU's sleep controller and various sleep modes. This ensures that you are going to sleep , The system will not be due to the port input floating caused by leakage.

The SLEEP control of the port is replaced by an external interrupt function. If the external interrupt request is invalid, SLEEP control remains It can work. SLEEP control function will be replaced by some other second function, please refer to the following on the port Two functions introduced.

### Quickly flips the port state

The port status is set to the output IO, which can be changed by the PORTn register. If you need to flip the current Port output state, usually need to first read the current port state PINx, and then take back the write back to the PORTn register Turn in. The LGT8FX8P provides another more efficient way to flip the port state by writing directly to the PINx register 1 to flip the specified port status. For example, we write PINB [3] to 1, you can achieve the port state of PB3 Flip over. This is very useful for applications that need to generate an output clock.

### Digital / analog multiplex port

LGT8FX8P part of the port for the number of modular mixed function port. In addition to the internal DAC output PD4, other mixing ends The mouth is used as an analog input. 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 as needed to avoid impact on simulated revenue. The DIDR0 ~ 2 register is used to turn off the mixed function terminal The digital input channel of the port to avoid the analog input on the digital circuit caused by excessive power loss. DIDRx does not close the port Digital output function.

### High current push-pull drive port

LGT8FX8P support for more than 6 high-current push-pull drive port, support up to 80mA push-pull drive. Taking into account the core Chip VCC maximum over-current capacity limit, is not recommended to open at the same time 6-way high current drive. Especially for only a set of power terminals Mouth QFP32 package, it is recommended not to open and drive more than 4 high current load.

Ordinary port driver is 12mA, the software needs to open the port through the HDR register high current drive function. 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 | |
| | | PD1 | | The Q1 in the QFP32 package is equivalent to the PD1 of the QFP48 |

| | | | | |
|---|---|---|---|---|
| PF1 | PF1 | PF1 | HDR [2] | Parallel to PF1 |
| PF2 | PF2 | PD2 | HDR [3] | QFP32 package PD2 internal equivalent to QFP48 PD2 |
| | | PF2 | | Parallel to PF2 |
| PF4 | PF4 | PE4 | HDR [4] | QFP32 package PE4 internal equivalent to QFP48 PF4 |
| | | PF4 | | Parallel to PE4 |
| PF5 | PF5 | PE5 | HDR [5] | QFP32 package PE5 internal equivalent to QFP48 PF5 |
| | | PF5 | | Parallel to PE5 |





### Idle port processing

If some ports are not being used, it is recommended to drive them to a fixed level. In any case, floating
The pin will bring more power consumption, and will lead to the system under strong interference becomes unstable.

The easiest way to give a fixed level to a port is to turn on the pull-up resistor on the port. It should be noted that the pull-up resistor
It is forbidden during power-on reset. Pull-up resistor will also bring the excess leakage. It is therefore recommended to use an external one
Pull or pull-down resistor connection. It is not advisable to connect the port directly to the power supply or ground because if these pins are configured as
Output, it may lead to a very large current through the port, the chip caused a devastating impact.

## Port reuse function

Most of the ports have a multiplexing function, the following equivalent circuit describes the port multiplexing function of the port control. These complex
The function does not necessarily exist with the port pin.


| | |
|---|---|
| PUOExn: Pxn PULL-UP OVERRIDE ENABLE | PUD: PULLUP DISABLE |
| PUOVxn: Pxn PULL-UP OVERRIDE VALUE | WDx: WRITE DDRx |
| DDOExn: Pxn DATA DIRECTION OVERRRIDE ENABLE | RDx: READ DDRx |
| DDOVxn: Pxn DATA DIRECTION OVERRIDE VALUE | RRx: READ PORTx REGISTER |
| PVOExn: Pxn PORT VALUE OVERRIDE ENABLE | WRx: WRITE PORTx |
| PVOVxn: Pxn PORT VALUE OVERRIDE VALUE | RPx: READ PORTx PIN |
| DIEOExn: Pxn INPUT-ENABLE OVERRIDE ENABLE | WPx: WRITE PINx |
| DIEOVxn: Pxn INPUT-ENABLE OVERRIDE VALUE | IO_CLK: I / O CLOCK |
| SLEEP: SLEEP CONTROL | DIxn: INPUT PIN n ON PORTx |
| PTOExn: Pxn PORT TOGGLE OVERRIDE ENABLE | AIOxn: ANALOG I / O PIN n ON PORTx |

Alternate function control signal General description:

| signal | Full name | Functional description |
|---|---|---|
| PUOE | pull-up multiplex enable | This bit is 1, pull-up enable is controlled by PVOV; if this bit is 0, pull-up enable by DDxn, PORTxn and PUD common control system |
| PUOV | pull-up multiplexing value | If PUOE is 1, this bit will enable the pull-up of the pin Resistance, otherwise it will disable the pin pull-up resistor |
| DDOE | port direction multiplexing enable | Bit is 1, pin output enabled by DDOE control, otherwise by DDxn control |
| DDOV | port direction multiplexing value | If DDOE is 1 and bit 1 is 1, the output of the pin is enabled Function, otherwise the output of the pin is turned off |
| PVOE | port data multiplexing enable | If the bit is 1, and the pin output is enabled, the pin is output The value will be controlled by PVOV, otherwise it will be controlled by PORTxn |
| PVOV | port data reuse value | Refer to the PVOE function description |
| PTOE | port rollover is enabled | The bit is 1, and the PORTxn bit will flip |
| DIEOE | digital input enable multiplexing enable | If the bit is 1, the port digital input enable will be controlled by DIEOV System; otherwise there will be MCU running state control |
| DIEOV | digital input enables multiplexed values | If DIEOE is 1, the digital input function of the port will be bit by bit Control, independent of the MCU operating status |
| DI | Digital input | This is the digital input signal input to the alternate function module. From the I / O and other circuit can be seen in the next, this value in the dense After the special trigger, but before the I / O input synchronizer. This one Signal is connected to the peripheral module, the peripheral module will be required To be synchronized |
| AIO | Analog input | Analog input / output signal, this signal directly with I / O PAD Connected, can be used as a simulation of the two-way signal. This signal is straight Connected to the internal ADC, comparator and other analog modules port phase connection |

The following section will briefly describe the alternate function of each pin and the associated control signal.

Port B multiplexing function

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

| | |
|---|---|
| | MOSI (SPI bus master output / slave input) |
| PB3 | OC2A (Timer / Event Counter 2 Compare Match Output A) |
| | PCINT3 (pin change interrupt 3) |
| | SSN (SPI bus slave device select input) |
| PB2 | OC1B (Timer / Event Counter 1 Compare Match Output B) |
| | PCINT2 (pin change interrupt 2) |
| PB1 | OC1A (Timer / Event Counter 1 Compare Match Output A) |
| | PCINT1 (pin change interrupt 1) |
| | ICP1 (Timer / Event Counter 1 capture input) |
| PB0 | CLKO (system clock output) |
| | PCINT0 (pin change interrupt 0) |

### XTALI / TOSC2 / PCINT7 - Port B pin 7

XTALI: External Crystal Pin XI. When used as a crystal clock signal, this pin will not be used as an I / O.

TOSC2: Timer external crystal pin 2. When the internal RC is configured as the main working clock of the chip, and the difference is enabled

Step timer function (ASSR register configuration), this pin will be used as a timer external crystal pin. When ASSR is registered

AS2 is set to 1 and EXCLK is set to 0, the Timer / Event Counter 2 is enabled with an external crystal

Clock function, PB7 will be disconnected from the internal I / O port and become the reverse output pin of the internal oscillator amplifier. This mold

The external crystal is connected to the pin.

PCINT7: Pin Change Interrupt 7. PB7 is an external interrupt source.

If PB7 is used for crystal pins, the values of DDB7, PORTB7 and PINB7 will not make any sense.

### XTALO / TOSC1 / PCINT6 - Port B pin 6

XTALO: External crystal pin XO.

TOSC1: Timer external crystal pin 1. When the internal RC is configured as the main working clock of the chip, and the difference is enabled

Step timer function (ASSR register configuration), this pin will be used as a timer external crystal pin. When ASSR is registered

AS2 is set to 1 and EXCLK is set to 0, the Timer / Event Counter 2 is enabled with an external crystal

Clock function, PB6 will be connected to the internal I / O port port as the internal oscillator amplifier input pin. In this mode,

The external crystal is connected to the pin.

PCINT6: Pin Change Interrupt 6. PB6 is an external interrupt source.

If PB6 is used for the crystal 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 device,

The direction of this pin is controlled by DDB5. When this pin is forced by the SPI input, it can still pass PORTB5

Bit control pull-up resistor.

PCINT5: Pin level change interrupt. PB5 is an external interrupt source.

### MISO / PCINT4- Port B pin 4

MISO: SPI control master device data input, slave device data output. When the SPI is configured as a master device, this pin will be

Will be forced to input, not subject to DDB4 control. When the SPI is used as a slave device, this pin is the data side



To control by DDB4. When this pin is forced by the SPI controller, its pull-up resistor can still pass

PROTB4 control.

PCINT4: Pin level 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 to input, and not subject to DDB3 control. When the SPI controller is configured as a master device, this pin is

The method is controlled by DDB3. When this pin is forced by the SPI control input, it can still be controlled by PORTB3

Of the pull-up resistor.
OC2A: Timer / Counter 2 Group A compare match output. PB3 can be used as a timer / counter 2 compare match
Department of output. The pin must be set to output via DDB3. At the same time, OC2A is also the PWM mode of Timer 2
Type output pin.
PCINT3: Pin level change interrupt. PB3 is an external interrupt source.

### SSN / OC1B / PCINT2 - Port B pin 2

SSN: SPI slave device chip select input. When the SPI controller is configured as a slave, this pin will be forced for input,
And is not controlled by DDB2. As a slave device, the SPI controller is driven low on the SSN is valid. When the SPI controls
The device is configured as the master device and the direction of this pin is controlled by DDB2. When this pin is forced by the SPI controller to lose
After entering, you can still control the pull-up resistor via PORTB2.
OC1B: B / O of the Timer / Event Counter 1 compare match output. PB2 can be used as a timer / counter 1 compare match
Department of output. The pin must be set to output via DDB2. At the same time, OC1B is also the PWM mode of Timer 1
Type output pin.
PCINT2: Pin level change interrupt. PB2 is an external interrupt source.

### OC1A / PCINT1 - Port B pin 1

OC1A: Group A compare match output for Timer / Event Counter 1. PB1 can be used as a timer / counter 1 compare match
Department of output. The pin must be set to output via DDB1. At the same time, OC1A is also the PWM mode of Timer 1
Type output pin.
PCINT1: Pin level change interrupt. PB1 is an external interrupt source.

### ICP1 / CLKO / PCINT0 - Port B pin 0

ICP1: Capture input pin for Timer / Event Counter 1
CLKO: System operating clock output, when CLKPR register CLKOE bit is 1, this pin will be forced to
Output, not controlled by DDB0. The output frequency is the operating clock frequency of the current system.
PCINT0: Pin level change interrupt. PB0 is an external interrupt source.





### Port C multiplexing function

| Pin | Alternate function description |
|---|---|
| PC7 | ADC8 (ADC input channel 8)<br>APN2 (DAP reverse input 2)<br>PCINT15 (pin level change input 15) |
| PC6 | RESETN (external reset input)<br>PCINT14 (pin change input 14) |
| PC5 | ADC5 (ADC input channel 5)<br>SCL (TWI clock line)<br>PCINT13 (pin change input 13) |
| PC4 | ADC4 (ADC input channel 4)<br>SDA (TWI data cable)<br>PCINT12 (pin change input 12) |
| PC3 | ADC3 (ADC input channel 3)<br>PCINT11 (pin level change input 11) |
| PC2 | ADC2 (ADC input channel 2)<br>PCINT10 (pin level change input 10) |

| | |
|---|---|
| PC1 | ADC1 (ADC input channel 1)<br>PCINT9 (pin level change input 9) |
| PC0 | ADC0 (ADC input channel 0)<br>PCINT8 (pin level change input 8) |

### ADC8 / APN2 / PCINT15 - Port C pin 6

ADC8: ADC external input channel 8

APN2: Reverse input port of differential amplifier 2

PCINT15: Pin change interrupt. After turning off the external reset input function of this pin, PC7 can be done outside

Interrupt source.

### RESETN / PCINT14 - Port C pin 6

RESETN: External reset input pin. After a power-on reset, this pin defaults to an external reset function. Can pass through IOCR

The register turns off the external reset function. When the external reset function is turned off, this pin can be used as a general purpose I / O. But need to

It should be noted that in the power and other reset process, this pin defaults to reset input, so if the user needs

Use this pin of the general I / O function, the external circuit can not affect the chip power and reset process, it is recommended that this

The pins are configured as I / O for the output function and an external pull-up resistor is added externally.

PCINT14: Pin level change interrupt. After turning off the external reset input function of this pin, PC6 can be done outside

Interrupt source.

### SCL / ADC5 / PCINT13 - Port C pin 5

SCL: TWI interface clock signal. After the TWEN bit in the TWCR register is set, the TWI interface is enabled and PC5 will be enabled

TWI control, become TWI interface clock signal.

ADC5: ADC input channel 5. The DIDR register is used to turn off digital functions of the M / A I / O to avoid digital parts





On the impact of analog circuits. Please refer to the ADC-related section.

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, the TWI interface is enabled and PC4 will be enabled

TWI control, as TWI interface data signal.

ADC4: ADC input channel 4. The DIDR register is used to turn off digital functions of the M / A I / O to avoid digital parts

On the impact of analog circuits. Please refer to the ADC-related section.

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 digital functions of the M / A I / O to avoid digital parts

On the impact of analog circuits. Please refer to the ADC-related section.

APN1: Differential Amplifier Reverse 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 digital functions of the M / A I / O to avoid digital parts

On the impact of analog circuits. Please refer to the ADC-related section.

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 digital functions of the M / A I / O to avoid digital parts On the impact of analog circuits. Please refer to the ADC-related section.

APP1: Differential amplifier forward 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 digital functions of the M / A I / O to avoid digital parts

On the impact of analog circuits. Please refer to the ADC-related section.

APP0: Differential amplifier positive input 0

PCINT8: Pin change interrupt 8





### Port D multiplexing function

| Pin | Alternate function description |
|---|---|
| PD7 | ACXN (analog comparator 0/1 common negative input) |
| | PCINT23 (pin change interrupt 23) |
| | AC0P (QFP32: analog comparator 0 positive input) |
| PD6 | OC0A (Timer / Event Counter 0 Compare Match Output A) |
| | OC3A (QFP32: Timer / Event Counter 3 Compare Match Output A) |
| | PCINT22 (pin change interrupt 22) |
| | T1 (Timer / Counter 1 external count clock input) |
| PD5 | OC0B (Timer / Event Counter 0 Compare Match Output B) |
| | PCINT21 (pin change interrupt 21) |
| | XCK (USART external clock input / output) |
| PD4 | DAO (internal 8bit DAC analog output) |
| | T0 (Timer / Event Counter 0 External Count Clock Input) |
| | PCINT20 (pin change interrupt 20) |
| | INT1 (external interrupt input 1) |
| PD3 | OC2B (Timer / Event Counter 2 Compare Match Output B) |
| | PCINT19 (pin change interrupt 19) |
| | INT0 (external interrupt input 0) |
| PD2 | AC0O (comparator 0 output) |
| | OC3B (QFP32: Timer / Event Counter 3 Compare Match Output B) |
| | PCINT18 (pin change interrupt 18) |
| | TXD (USART data output) |
| PD1 | OC3A (QFP32: Timer / Event Counter 3 Compare Match Output A) |
| | PCINT17 (pin change interrupt 17) |
| PD0 | RXD (USART data entry) |
| | PCINT16 (pin change interrupt 16) |

### ACXN / OC2B / PCINT23 - Port D pin 7

ACXN: Analog Comparator 0/1 Common Negative Input

OC2B: B / O of the Timer / Event Counter 2 compare match output. PD7 can be used as a timer / counter 2 compare match

Department of output. The pin must be set to output via DDD7. At the same time, OC2B is also the PWM mode of Timer 2

Output pin;

PCINT23: Pin Change Interrupt 23

AC0P / OC0A / PCINT22- Port D pin 6

AC0P: Analog Comparator 0 Positive input.

OC0A: Timer / Counter 0 Group A compare match output. PD6 can be used as a timer / counter 0 compare match

Department of output. The pin must be set to output via DDD6. At the same time, OC0A is also the PWM mode of Timer 0

Type output pin

PCINT22: Pin change interrupt 22



**Page 78**



T1 / OC0B / PCINT21 - Port D pin 5

T1: External count clock input for Timer / Event Counter 1

OC0B: Timer / Counter 0 Group B compare match output. PD5 can be used as a timer / counter 0 compare match

Department of output. In this case, the pin must be set to output via DDD5. At the same time, OC0B is also the PWM mode of Timer 0

Type output pin

PCINT21: Pin change interrupt 21

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

XCK: Synchronous mode USART external clock signal

T0: External count clock input for Timer / Event 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: B / O of the Timer / Event Counter 2 compare match output. PD3 can be used as a timer / counter 2 compare match

Department of output. In this case, the pin must be set to output via DDD3. At the same time, OC2B is also the PWM mode of Timer 2

Type output pin

PCINT19: Pin Change Interrupt 19

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

INT0: External interrupt input 0

OC3B: Timing counter 3 compare match output B. Only in the QFP32 package, PD2 and QFP48 / PF2 combined into one

IO, so the OC2B function on PF2 will also be output from PD2

AC0O: Analog Comparator 0 Compare results are output directly. Controlled by the AC0FR register

PCINT18: Pin change interrupt 18

TXD / OC3A / PCINT17 - Port D pin 1

TXD: Transfer data (USART data output). After the USART transmitter is enabled, PD1 will be forced to output

DDD1 control

OC3A: Timer Counter 3 Compare Match Output A. Only in the QFP32 package, PD1 and QFP48 / PF1 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: Transfer data (USART data entry). After the USART receiver is enabled, PD0 will be forced as input

DDD0 control. When the pin is forced into the USART input, the pull-up resistor can still be controlled by the PORTD0 bit

PCINT16: Pin change interrupt 16

Port E multiplexing function

| Pin | Alternate function description |
|---|---|
| PE7 | ADC11 (ADC input channel 11) |
| | PCINT31 (pin change interrupt 31) |
| | AVREF (QFP32: ADC external reference voltage) |
| PE6 | ADC10 (ADC input channel 10) |
| | PCINT30 (pin change interrupt 30) |
| | CLKO (system clock output) |
| PE5 | AC1O (analog comparator 1 output) |
| | PCINT29 (pin change interrupt 29) |
| PE4 | OC0A (Timer / Event Counter 0 Compare Configuration Output A) |
| | PCINT28 (pin change interrupt 28) |
| | ADC7 (ADC input channel 7) |
| PE3 | AC1N (analog comparator 1 negative input) |
| | PCINT27 (pin change interrupt 27) |
| PE2 | SWD (SWD debugger data cable) |
| | PCINT26 (pin change interrupt 26) |
| | ADC6 (ADC input channel 6) |
| PE1 | ACXP (analog ratio machine 0/1 common positive input) |
| | PCINT25 (pin change interrupt 25) |
| | SWC (SWD debugger clock input) |
| PE0 | 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, used as an analog function, the need to set the corresponding digital I / O input,

And turn off the pull-up resistor to prevent digital circuits from interfering with analog circuits

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 a backup pin for PB0 / CLKO

AC1O: Analog Comparator 1 output

PCINT29: Pin Change Interrupt 29

### OC0A / PCINT28 - Port E pin 4

OC0A: Timer / Counter 0 Group A compare match output. PE4 can be used as a timer / counter 0 compare match Department of output. The pin must be set to output via DDE4. At the same time, OC0A is also the PWM mode of Timer 0 Type output pin.
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 digital functions of the M / A I / O to avoid digital parts On the impact of analog circuits. Please refer to the ADC-related section
AC1N: Analog Comparator 1 Negative Input
PCINT27: Pin change interrupt 27

### SWD / PCINT26 - Port E pin 2

SWD: SWD debugger data cable. PE2 defaults to SWD function. The user can pass the MCUSR register SWDD Position 1 closes the SWD debugger function. SWD is turned off, the debugging function will not be used.
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 digital functions of the M / A I / O to avoid digital parts On the impact of analog circuits. Please refer to the ADC-related section
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 pass the MCUSR register SWDD Position 1 closes the SWD debugger function. SWD is turned off, the debugging function will not be used
APN4: Differential amplifier reverse input 4
PCINT24: Pin change interrupt 24



### Port F multiplexing function

| Pin | Alternate function description |
|---|---|
| PF7 | OC2B (Timer / Event Counter 2 Compare Match Output B) |
| | PCINT39 (pin change interrupt 39) |
| | T3 (Timer / Event Counter 3 external clock input) |
| PF6 | OC2A (Timer / Event Counter 2 Compare Match Output A) |

| | |
|---|---|
| | PCINT38 (pin change interrupt 38) |
| PF5 | OC1A (Timer / Event Counter 1 Compare Match Output A) |
| | PCINT37 (pin change interrupt 37) |
| | OC1B (Timer / Event Counter 1 Compare Configuration Output B) |
| PF4 | 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 / Event Counter 3 Compare Match Output B) |
| | PCINT34 (pin change interrupt 34) |
| PF1 | OC3A (Timer / Event Counter 3 Compare Match Output A) |
| | PCINT33 (pin change interrupt 33) |
| | ADC9 (ADC external input channel 9) |
| PF0 | APN3 (differential amplifier reverse input 3) |
| | PCINT32 (pin change interrupt 32) |

### OC2B / PCINT39 - Port F pin 7

OC2B: Timer / Event Counter 2 Compare Match Output B. The output selection is controlled by the PMX1 register
PCINT39: Pin Change Interrupt 39

### OC2A / T3 / PCINT38 - Port F pin 6

OC2A: Timer / Event Counter 2 Compare Match Output A. The 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. The output selection is controlled by the PMX0 register
PCINT37: Pin Change Interrupt

### ICP3 / OC1B / PCINT36 - Port F pin 4

OC1B: B / O of the Timer / Event Counter 1 compare match output. The 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 / Counter 0 Group B compare match output. The output selection is controlled by the PMX0 register
OC3C: Group C / Counter 3 Compare Match Output
PCINT35: Pin Change Interrupt 35

### OC3B / PCINT34- Port F pin 2

OC3B: B / O compare match output for Timer / Event Counter 3
PCINT34: Pin Change Interrupt 34

### OC3A / PCINT33 - Port F pin 1

OC3A: B / O compare match output of Timer / Event Counter 3. The 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 reverse 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: 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 Register |

### Port B Direction Register - DDRB

DDRB - port B direction register

DDRB: 0x04 (0x24) Default: 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: 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 the PINB directly to obtain the current state of the port; Writing a PINBn bit 1 will flip the output status of PORTBn |

### Port C Output Data Register - PORTC

PORTC - Port C output data register

PORTC: 0x08 (0x28) Default: 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 | C group port output register |

### Port C direction register - DDRC

DDRC - port C direction register

DDRC: 0x07 (0x27) Default: 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 | C group port direction control bit; 1 = output, 0 = input |

### Port C Input Data Register - PINC

PINC - Port C input data register

PINB: 0x06 (0x26) Default: 0x00

| PINC | PINC7 | PINC6 | PINC5 | PINC4 | PINC2 | 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 | C group port status register; read PINC to get the current port status<br>Writing a PINC will flip the current port output |

### Port D Output Data Register - PORTD

PORTD - Port D Output Data Register

PORTD: 0x0B (0x2B) Default: 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 | D group port output register |

### Port D Direction Register - DDRD

DDRD - port D direction register

DDRD: 0x0A (0x2A) Default: 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 | D group port output direction control register |

### Port D Input Data Register - PIND

PIND - Port D input data register

PIND: 0x09 (0x29) Default: 0x00

| PIND | PIND7 | PIND6 | PIND5 | PIND1 | PIND1 | PIND1 | PIND1 | PIND |
|---|---|---|---|---|---|---|---|---|
| R / W | R / W | R / W | R / W | R / W | R / W | R / W | R / W | R / W |

Bit definition

| | | |
|---|---|---|
| [7: 0] | PIND | D group port status register<br>Read PIND to get the current port level status<br>Write PINDn to 1, flip PORTDn corresponding to the state of the bit |

### Port E Output Data Register - PORTE

PORTE - Port E output data register

PORTE: 0x0E (0x2E) Default: 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 | E group port output register |

### Port E Direction Register - DDRE

DDRE - Port E direction register

DDRE: 0x0D (0x2D) Default: 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 | E group port direction control register |

### Port E Input Data Register - PINE

PINE - Port E input data register

PINE: 0x0C (0x2C) Default: 0x00

| PINE | PINE7 | PINE6 | PINE5 | PINE4 | PINE3 | PINE1 | PINE1 | PINE0 | PINE |
|---|---|---|---|---|---|---|---|---|---|



**Page 85**



| R / W | R / W | R / W | R / W | R / W | R / W | R / W | R / W | R / W |
|---|---|---|---|---|---|---|---|---|

Bit definition

| | | |
|---|---|---|
| | | Group E Port Status Register |
| [7: 0] | PINE | Read PINE to get the current port level |
| | | Write PINEn to 1, flip the state of the PORTEn bit |

### Port F Output Register - PORTF

PINF - Port F Enter the data register

PORTF: 0x14 (0x34) Default: 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

| | | |
|---|---|---|
| | | F group port status register |
| [7: 0] | PORTF | In the input mode port, the corresponding bit is written to 1 to turn on the internal pull-up |
| | | Output mode port, corresponding to bit 1 will drive the output high |

### Port F Direction Control Register - DDRF

DDRF - Port F direction control register

DDRF: 0x13 (0x33) Default: 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 | F group port direction control register |

### Port F Status Register - PINF

PINF - Port F status register

PINF: 0x12 (0x32) Default: 0x00

| Bits | PINF7 | PINF6 | PINF1 | PINF1 | PINF1 | PINF1 | PINF | |
|---|---|---|---|---|---|---|---|---|
| R / W | R / W | R / W | R / W | R / W | R / W | R / W | R / W | R / W |

Bit definition

| | | |
|---|---|---|
| | | F group port status register |
| [7: 0] | PINF | Read PINF to get the current level of port F |
| | | PINFn write 1, flip PORTFn corresponding to the status of the bit |

### Port Drive Control Register - HDR

HDR0 - port driver control register

HDR: 0xE0 Default: 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] | - | Keep not used |
| 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 Multiplexed Control Register 0 - PMX0

PMX0 - Port Multiplexed Control Register

PMX0: 0xEE Default: 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 PMX0 / 1 register, you need to first Write the WCE bit to 1, and complete the PMX0 / 1 in the next six system cycles Update. |
| 6 | C1BF4 | OC1B Auxiliary Output Control<br>1 = OC1B is output to PF4<br>0 = OC1B is output to PB2 |
| 5 | C1AF5 | OC1A Auxiliary Output Control<br>1 = OC1A output to PF5<br>0 = OC1A output to PB1 |
| 4 | C0BF3 | OC0B Auxiliary Output Control<br>1 = OC0B output to PF3<br>0 = OC0B output to PD5 |
| 3 | C0AC0 | OC0A Auxiliary Output Control<br>The 0C0A output is controlled by the C0AC0 bit and the C0AS bit of the TCCR0B register:<br>{C0AC0, C0AS} =<br>00 = OC0A output to PD6<br>01 = 0C0A Output to PE4<br>10 = 0C0A output to PC0<br>11 = OC0A is output to both PE4 and PC0 |
| 2 | SSB1 | SPSS Auxiliary Output Control<br>1 = SPSS output to PB1<br>0 = SPSS output to PB2 |
| 1 | TXD6 | Serial TXD Auxiliary Output Control<br>1 = TXD output to PD6, 0 = TXD output to PD1 |
| 0 | RXD5 | Serial RXD Auxiliary Input Control<br>1 = RXD input from PD5, 0 = RXD input from PD0 |

### Port Multiplex Control Register 1 - PMX1

PMX1 - Port Multiplexed Control Register

PMX1: 0xED Default: 0x00

| Bit | - | - | - | - | - | C3AC | C2BF7 | C2AF6 |
|---|---|---|---|---|---|---|---|---|
| R / W | - | - | - | - | - | R / W | R / W | R / W |

Bit definition

| | | |
|---|---|---|
| [7: 3] | - | Keep not used |
| 2 | C3AC | OC3A Auxiliary Output Control<br>1 = OC3A output to QFP48 / AC0P<br>0 = OC3A output to PF1 |
| 1 | C2BF7 | OC2B Auxiliary Output Control<br>1 = OC2B output to PF7<br>0 = OC2B output to PD3 |
| 0 | C2AF6 | OC2A Auxiliary Output Control<br>1 = OC2A output to PF6<br>0 = OC2A output to PB3 |

Instructions for use

PMX0 / 1 shared register update protection control bit PMX0 [7], update PMX1, please refer to PMX0 register Control of PMX0 [7].

### Port Multiplex Control Register 2 - PMX2

PMX2 - Port Multiplexed Control Register

PMX2: 0xF0 Default: 0x00

| Bit | WCE | STSC1 | STSC0 | - | - | XIEN | E6EN | C6EN |
|---|---|---|---|---|---|---|---|---|
| R / W | R / W | R / W | R / W | - | - | R / W | R / W | R / W |

Bit definition

| | | |
|---|---|---|
| [7] | WCE | PMX2 update enable control; before updating PMX2 register, you need to write first<br>The WCE bit is 1, and the PMX2 update is completed in the next six system cycles. |
| [6] | STSC1 | High Speed Crystal IO Start Circuit Control<br>STSC1 is automatically enabled when high-speed crystal is enabled via PMCR. When switching the system<br>After the clock to the external high-speed crystal, STSC1 automatically clear. The software can also be stabilized in the crystal<br>After the timer, manually clear the STSC1, has turned off the crystal oscillator circuit, saving power. |
| [5] | STSC0 | Low-speed crystal oscillator IO start circuit control<br>STSC0 is enabled automatically after enabling low-speed crystal through PMCR. When switching the system<br>After the clock to the external low-speed crystal, STSC0 automatically clear. The software can also be stabilized in the crystal<br>After the timer, manually clear the STSC0, has turned off the crystal oscillator circuit, saving power. |
| [4: 3] | - | Keep not used |
| [2] | XIEN | To enable external clock input, you need to enable external crystal at the same time |
| [1] | E6EN | Enable the common IO function of PE6. The default PE6 is AVREF function |
| [0] | C6EN | Enable the general IO function of PC6; default PC6 is external reset input |



## Pin level change interrupt

- 40 pin change interrupt sources
- 5 interrupt entries

### Summary

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

Any enabled flip-flop of the PBn pin will trigger pin level interrupt PCI0, enabled PCn pin flip will trigger
PCI1, enabled PDn pin flip will trigger PCI2, enabling PEn pin flip will trigger PCI3. Each pin changes
The interrupt enable is controlled by the PCMSK0 ~ 4 register, respectively. All of the pin-level interruptions are detected asynchronously
Used as a wake source in some sleep modes.

## Register definition

Pin Change Interrupt register list

| register | address | Defaults | description |
|---|---|---|---|
| PCICR | 0x68 | 0x00 | The pin changes the interrupt control register |
| PCIFR | 0x3B | 0x00 | The pin changes the 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: 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 | - | Keep it. |
| 4 | | PCIE4 pin change interrupt enable control bit 4.<br>When the PCIE4 bit is set to "1" and the global interrupt is enabled, pin change interrupt 4 is enabled.<br>Any change in the level of the enabled PFn pin will cause a PCI4 interrupt. PFn pin<br>The interrupt enable can be controlled by the PCMSK4 register, respectively.<br>When the PCIE3 bit is set to "0", the pin change interrupt 3 is disabled. |
| 3 | | PCIE3 pin change interrupt enable control bit 3.<br>When the PCIE3 bit is set to "1" and the global interrupt is enabled, pin change interrupt 3 is enabled. |





| | | |
|---|---|---|
| | | Any change in the enabled PEn pin will cause a PCI3 interrupt. PEn pin<br>The interrupt enable can be controlled by the PCMSK3 register, respectively.<br>When the PCIE3 bit is set to "0", the pin change interrupt 3 is disabled. |
| 2 | | PCIE2 pin change interrupt enable control bit 2.<br>When the PCIE2 bit is set to "1" and the global interrupt is enabled, pin change interrupt 2 is enabled.<br>Any change in the enabled PDn pin will cause a PCI2 interrupt. PDn pin<br>The interrupt enable can be controlled by the PCMSK2 register, respectively.<br>When the PCIE2 bit is set to "0", the pin change interrupt 2 is disabled. |
| 1 | | PCIE1 pin change interrupt enable control bit 1.<br>When the PCIE1 bit is set to "1" and the global interrupt is enabled, pin change interrupt 1 is enabled.<br>Any change in the enabled PCn pin will cause a PCI1 interrupt. PCn pin<br>The interrupt enable can be controlled by the PCMSK1 register, respectively.<br>When the PCIE1 bit is set to "0", pin change interrupt 1 is disabled. |
| 0 | | PCIE0 pin change interrupt enable control bit 0.<br>When the PCIE0 bit is set to "1" and the global interrupt is enabled, pin change interrupt 0 is enabled.<br>Any change in the enabled bit of the PBn pin will cause a PCI0 interrupt. PBn pin |

The interrupt enable can be controlled by the PCMSK0 register, respectively.

When the PCIE0 bit is set to "0", the pin change interrupt 0 is disabled.

### PCIFR - Pin Change Interrupt Flag Register

PCIFR - Pin Change Interrupt Flag Register

Address: 0x3B Default: 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 - Keep it.

4 PCIF4 pin change interrupt flag bit 4.

The level change of any enabled PFn pin sets the PCIF4. When PCIE4 and full

When the interruption is set, the MCU will jump to the PCI4 interrupt entry address. PFn pin

The enable of the interrupt can be controlled by the PCMSK4 register, respectively.

The execution of the interrupt service routine or writing "1" to the PCIF4 bit will clear the PCIF4 bit.

3 PCIF3 pin change interrupt flag bit 3.

The PCIV3 is set for any of the enabled PEn pin changes. When PCIE3 and all

When the interruption is set, the MCU will jump to the PCI3 interrupt entry address. PEn pin

The enable of the interrupt can be controlled by the PCMSK3 register, respectively.

The execution of the interrupt service routine or writing "1" to the PCIF3 bit will clear the PCIF3 bit.

2 PCIF2 pin change interrupt flag bit 2.

The level change of any enabled PDn pin will set PCIF2. When PCIE2 and all

When the interruption is set, the MCU will jump to the PCI2 interrupt entry address. PDn pin

The enable of the interrupt can be controlled by the PCMSK2 register, respectively.

The execution of the interrupt service routine or writing "1" to the PCIF2 bit will clear the PCIF2 bit.

1 PCIF1 pin change interrupt flag bit 1.



The level change of any enabled PCn pin sets PCIF1. When PCIE1 and all

When the interruption is set, the MCU will jump to the PCI1 interrupt entry address. PCn pin

The enable of the interrupt can be controlled by the PCMSK1 register, respectively.

The execution of the interrupt service routine or writing "1" to the PCIF1 bit clears the PCIF1 bit.

0 PCIF0 pin change interrupt flag bit 0.

The level change of any enabled PBn pin sets PCIF0. When PCIE0 and all

When the interruption is set, the MCU will jump to the PCI0 interrupt entry address. PBn pin

The enable of the interrupt can be controlled by the PCMSK0 register, respectively.

The execution of the interrupt service routine or writing "1" to the PCIF0 bit clears the PCIF0 bit.

### PCMSK0 - Pin Change Interrupt Mask Register 0

PCMSK0 - Pin Change Mask Register 0

Address: 0x6B Default: 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 the PCINT7 bit is set to "1", the PB7 pin change interrupt is enabled. PB7 pin

The level change will set PCIF0. If PCIE0 bit and global interrupt are set,

PCI0 interrupt. When the PCINT7 bit is set to "0", the PB7 pin change interrupt is disabled.

6 PCINT6 pin change enable mask 6.

When the PCINT6 bit is set to "1", the PB6 pin change interrupt is enabled. PB6 pin

The level change will set PCIF0. If PCIE0 bit and global interrupt are set,

PCI0 interrupt. When setting the PCINT6 bit to "0", the PB6 pin change interrupt is disabled.

5 PCINT5 pin change enable mask bit 5.

When the PCINT5 bit is set to "1", the PB5 pin change interrupt is enabled. PB5 pin
The level change will set PCIF0. If PCIE0 bit and global interrupt are set,
PCI0 interrupt. When setting the PCINT5 bit to "0", the PB5 pin change interrupt is disabled.

4 PCINT4 pin change enable mask bit 4.

When the PCINT4 bit is set to "1", the PB4 pin change interrupt is enabled. PB4 pin
The level change will set PCIF0. If PCIE0 bit and global interrupt are set,
PCI0 interrupt. When the PCINT4 bit is set to "0", the PB4 pin change interrupt is disabled.

3 PCINT3 pin change enable mask bit 3.

When the PCINT3 bit is set to "1", the PB3 pin change interrupt is enabled. PB3 pin
The level change will set PCIF0. If PCIE0 bit and global interrupt are set,
PCI0 interrupt. When the PCINT3 bit is set to "0", the PB3 pin change interrupt is disabled.

2 PCINT2 pin change enable mask bit 2.

When the PCINT2 bit is set to "1", the PB2 pin change interrupt is enabled. PB2 pin
The level change will set PCIF0. If PCIE0 bit and global interrupt are set,
PCI0 interrupt. When setting the PCINT2 bit to "0", the PB2 pin change interrupt is disabled.

1 PCINT1 pin change enable mask bit 1.

When the PCINT1 bit is set to "1", the PB1 pin change interrupt is enabled. PB1 pin





The level change will set PCIF0. If PCIE0 bit and global interrupt are set,
PCI0 interrupt. When setting the PCINT1 bit to "0", the PB1 pin change interrupt is disabled.

0 PCINT0 pin change enable mask bit 0.

When setting the PCINT0 bit to "1", the PB0 pin change interrupt is enabled. PB0 pin
The level change will set PCIF0. If PCIE0 bit and global interrupt are set,
PCI0 interrupt. When setting the PCINT0 bit to "0", the PB0 pin change interrupt is disabled.

### PCMSK1 - Pin Change Interrupt Mask Register 1

PCMSK1 - Pin Change Mask Register 1

Address: 0x6C Default: 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 15.

When the PCINT15 bit is set to "1", the PC7 pin change interrupt is enabled. PC7 pin
The level change will set the PCIF1 if the PCIE1 bit and the global interrupt are set,
PCI1 interrupt. When the PCINT15 bit is set to "0", the PC7 pin change interrupt is disabled only.

6 PCINT14 pin change enable mask 14.

When the PCINT14 bit is set to "1", the PC6 pin change interrupt is enabled. PC6 pin
The level change will set the PCIF1 if the PCIE1 bit and the global interrupt are set,
PCI1 interrupt. When the PCINT14 bit is set to "0", the PC6 pin change interrupt is disabled only.

5 PCINT13 pin change enable mask bit 13.

When the PCINT13 bit is set to "1", the PC5 pin change interrupt is enabled. PC5 pin
The level change will set the PCIF1 if the PCIE1 bit and the global interrupt are set,
PCI1 interrupt. When the PCINT13 bit is set to "0", the PC5 pin change interrupt is disabled only.

4 PCINT12 pin change enable mask 12.

When the PCINT12 bit is set to "1", the PC4 pin change interrupt is enabled. PC4 pin
The level change will set the PCIF1 if the PCIE1 bit and the global interrupt are set,
PCI1 interrupt. When the PCINT12 bit is set to "0", the PC4 pin change interrupt is disabled

only.

3 PCINT11 pin change enable mask bit 11.
When the PCINT11 bit is set to "1", the PC3 pin change interrupt is enabled. PC3 pin
The level change will set the PCIF1 if the PCIE1 bit and the global interrupt are set,
PCI1 interrupt. When the PCINT11 bit is set to "0", the PC3 pin change interrupt is disabled
only.

2 PCINT10 pin change enable mask bit 2.
When the PCINT10 bit is set to "1", the PC2 pin change interrupt is enabled. PC2 pin
The level change will set the PCIF1 if the PCIE1 bit and the global interrupt are set,
PCI1 interrupt. When the PCINT10 bit is set to "0", the PC2 pin change interrupt is disabled





only.

1 PCINT9 Pin change enable mask bit 1.
When the PCINT9 bit is set to "1", the PC1 pin change interrupt is enabled. PC1 pin
The level change will set the PCIF1 if the PCIE1 bit and the global interrupt are set,
PCI1 interrupt. When the PCINT9 bit is set to "0", the PC1 pin change interrupt is disabled.

0 PCINT8 Pin change enable mask bit 0.
When the PCINT8 bit is set to "1", the PC0 pin change interrupt is enabled. PC0 pin
The level change will set the PCIF1 if the PCIE1 bit and the global interrupt are set,
PCI1 interrupt. When setting the PCINT8 bit to "0", the PC0 pin change interrupt is disabled.

### PCMSK2 - Pin Change Interrupt Mask Register 2

PCMSK2 - Pin Change Mask Register 2

Address: 0x6D Default: 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 the PCINT23 bit is set to "1", the PD7 pin change interrupt is enabled. PD7 pin
The level change will set PCIF2. If PCIE2 bit and global interrupt are set,
PCI2 interrupt.
When the PCINT23 bit is set to "0", the PD7 pin change interrupt is disabled.

6 PCINT22 pin change enable mask bit 6.
When the PCINT22 bit is set to "1", the PD6 pin change interrupt is enabled. PD6 pin
The level change will set PCIF2. If PCIE2 bit and global interrupt are set,
PCI2 interrupt.
When the PCINT22 bit is set to "0", the PD6 pin change interrupt is disabled.

5 PCINT21 pin change enable mask bit 21.
When the PCINT21 bit is set to "1", the PD5 pin change interrupt is enabled. PD5 pin
The level change will set PCIF2. If PCIE2 bit and global interrupt are set,
PCI2 interrupt.
When the PCINT21 bit is set to "0", the PD5 pin change interrupt is disabled.

4 PCINT20 pin change enable mask bit 20.
When the PCINT20 bit is set to "1", the PD4 pin change interrupt is enabled. PD4 pin
The level change will set PCIF2. If PCIE2 bit and global interrupt are set,
PCI2 interrupt.
When the PCINT20 bit is set to "0", the PD4 pin change interrupt is disabled.

3 PCINT19 pin change enable mask bit 19.
When the PCINT19 bit is set to "1", the PD3 pin change interrupt is enabled. PD3 pin
The level change will set PCIF2. If PCIE2 bit and global interrupt are set,
PCI2 interrupt.

When the PCINT19 bit is set to "0", the PD3 pin change interrupt is disabled.

2 PCINT18 pin change enable mask 18.



**Page 93**



When the PCINT18 bit is set to "1", the PD2 pin change interrupt is enabled. PD2 pin The level change will set PCIF2. If PCIE2 bit and global interrupt are set, PCI2 interrupt.

When the PCINT18 bit is set to "0", the PD2 pin change interrupt is disabled.

1 PCINT17 pin change enable mask bit 17.

When the PCINT17 bit is set to "1", the PD1 pin change interrupt is enabled. PD1 pin The level change will set PCIF2. If PCIE2 bit and global interrupt are set, PCI2 interrupt.

When the PCINT17 bit is set to "0", the PD1 pin change interrupt is disabled.

0 PCINT16 pin change enable mask bit 16.

When the PCINT16 bit is set to "1", the PD0 pin change interrupt is enabled. PD0 pin The level change will set PCIF2. If PCIE2 bit and global interrupt are set, PCI2 interrupt.

When the PCINT16 bit is set to "0", the PD0 pin change interrupt is disabled.

### PCMSK3 - Pin Change Interrupt Mask Register 3

PCMSK3 - Pin Change Mask Register 3

Address: 0x73 Default: 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 the PCINT31 bit is set to "1", the PE7 pin change interrupt is enabled. PE7 pin The level change will set the PCIF3 if the PCIE3 bit and the global interrupt are set, PCI3 interrupt.

When the PCINT31 bit is set to "0", the PE7 pin change interrupt is disabled.

6 PCINT30 pin change enable mask bit 30.

When the PCINT30 bit is set to "1", the PE6 pin change interrupt is enabled. PE6 pin The level change will set the PCIF3 if the PCIE3 bit and the global interrupt are set, PCI3 interrupt.

When the PCINT30 bit is set to "0", the PE6 pin change interrupt is disabled.

5 PCINT29 pin change enable mask bit 39.

When the PCINT29 bit is set to "1", the PE5 pin change interrupt is enabled. PE5 pin The level change will set the PCIF3 if the PCIE3 bit and the global interrupt are set, PCI3 interrupt.

When the PCINT29 bit is set to "0", the PE5 pin change interrupt is disabled.

4 PCINT28 pin change enable mask bit 28.

When the PCINT28 bit is set to "1", the PE4 pin change interrupt is enabled. PE4 pin The level change will set the PCIF3 if the PCIE3 bit and the global interrupt are set, PCI3 interrupt.

When the PCINT28 bit is set to "0", the PE4 pin change interrupt is disabled.

3 PCINT27 pin change enable mask bit 27.



**Page 94**

When the PCINT27 bit is set to "1", the PE3 pin change interrupt is enabled. PE3 pin
The level change will set the PCIF3 if the PCIE3 bit and the global interrupt are set,
PCI3 interrupt.

When the PCINT27 bit is set to "0", the PE3 pin change interrupt is disabled.

2 PCINT26 pin change enable mask bit 26.

When the PCINT26 bit is set to "1", the PE2 pin change interrupt is enabled. PE2 pin
The level change will set the PCIF3 if the PCIE3 bit and the global interrupt are set,
PCI3 interrupt.

When the PCINT26 bit is set to "0", the PE2 pin change interrupt is disabled.

1 PCINT25 pin change enable mask bit 25.

When the PCINT25 bit is set to "1", the PE1 pin change interrupt is enabled. PE1 pin
The level change will set the PCIF3 if the PCIE3 bit and the global interrupt are set,
PCI3 interrupt.

When setting the PCINT25 bit to "0", the PE1 pin change interrupt is disabled.

0 PCINT24 pin change enable mask bit 24.

When the PCINT24 bit is set to "1", the PE0 pin change interrupt is enabled. PE0 pin
The level change will set the PCIF3 if the PCIE3 bit and the global interrupt are set,
PCI3 interrupt.

When setting PCINT24 bit to "0", the PE0 pin change interrupt is disabled.

### PCMSK4 - Pin Change Interrupt Mask Register 4

PCMSK4 - Pin Change Mask Register 4

Address: 0x74 Default: 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 the PCINT39 bit is set to "1", the PF7 pin change interrupt is enabled. PF7 pin
The level change will set the PCIF4 if the PCIE4 bit and the global interrupt are set,
PCI4 interrupt.

When the PCINT39 bit is set to "0", the PF7 pin change interrupt is disabled.

6 PCINT38 pin change enable mask 38.

When the PCINT38 bit is set to "1", the PF6 pin change interrupt is enabled. PF6 pin
The level change will set the PCIF4 if the PCIE4 bit and the global interrupt are set,
PCI4 interrupt.

When the PCINT38 bit is set to "0", the PF6 pin change interrupt is disabled.

5 PCINT37 pin change enable mask bit 37.

When the PCINT37 bit is set to "1", the PF5 pin change interrupt is enabled. PF5 pin
The level change will set the PCIF4 if the PCIE4 bit and the global interrupt are set,
PCI4 interrupt.

When the PCINT37 bit is set to "0", the PF5 pin change interrupt is disabled.

4 PCINT36 pin change enable mask bit 36.



When the PCINT36 bit is set to "1", the PF4 pin change interrupt is enabled. PF4 pin
The level change will set the PCIF4 if the PCIE4 bit and the global interrupt are set,
PCI4 interrupt.

When the PCINT36 bit is set to "0", the PF4 pin change interrupt is disabled.

3 PCINT35 pin change enable mask bit 35.

When the PCINT35 bit is set to "1", the PF3 pin change interrupt is enabled. PF3 pin

The level change will set the PCIF4 if the PCIE4 bit and the global interrupt are set, PCI4 interrupt.

When the PCINT35 bit is set to "0", the PF3 pin change interrupt is disabled.

2 PCINT34 pin change enable mask bit 34.

When the PCINT34 bit is set to "1", the PF2 pin change interrupt is enabled. PF2 pin

The level change will set the PCIF4 if the PCIE4 bit and the global interrupt are set, PCI4 interrupt.

When the PCINT34 bit is set to "0", the PF2 pin change interrupt is disabled.

1 PCINT33 pin change enable mask 33.

When the PCINT33 bit is set to "1", the PF1 pin change interrupt is enabled. PF1 pin

The level change will set the PCIF4 if the PCIE4 bit and the global interrupt are set, PCI4 interrupt.

When the PCINT33 bit is set to "0", the PF1 pin change interrupt is disabled.

0 PCINT32 pin change enable mask bit 32.

When the PCINT31 bit is set to "1", the PF0 pin change interrupt is enabled. PF0 pin

The level change will set the PCIF4 if the PCIE4 bit and the global interrupt are set, PCI4 interrupt.

When setting the PCINT32 bit to "0", the PF0 pin change interrupt is disabled.



## Timer / Event Counter 0 (TMR0)

- 8-bit counter
- Two separate comparison units
- When the compare match occurs, the counter is automatically cleared and loaded automatically
- 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 selectable trigger sources automatically turn off the PWM output

High-speed, 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 produce waveforms precisely. TC0 package
With one count clock generation unit, one 8-bit counter, waveform generation mode control unit and two output compare single
yuan. At the same time, TC0 can be shared with TC1 10-bit prescaler, you can also use the 10-bit prescaler. Pre-divided
Frequency on the system clock clkio or high speed clock rcm2x (internal 32M RC oscillator output clock rc32m 2 times the frequency)
Divide to produce the count clock Clkt0. The waveform generation mode control unit controls the operating mode and comparison of the counter
Out of the waveform. Depending on the operating mode, the counter is cleared for each count clock Clkt0, plus one or
Minus one operation. Clkt0 can be generated by an internal clock source or an external clock source. When the counter count value TCNT0 reaches the maximum
The value (equal to the maximum value 0xFF or the output compare register OCR0A, defined as TOP, defines the maximum value of MAX to show the area
Other), the counter will be cleared or decremented. When the counter count value TCNT0 reaches the minimum value (equal to
0x00, defined as BOTTOM), the counter will be added to an operation. When the counter count value TCNT0 arrives
OCR0A / OCR0B, also known as a compare match, will clear or set the output compare signal OC0A / OC0B, to
Generates PWM waveforms. When the dead time is enabled, the dead time set (the count corresponding to the DTR0 register
Clock) will be inserted into the generated PWM waveform. The software can be turned off by clearing the COM0A / COM0B bit to zero
OC0A / OC0B waveform output, or set the corresponding trigger source, when the trigger event occurs when the hardware automatically cleared
COM0A / COM0B bits to turn off the OC0A / OC0B waveform output.

The count clock can be generated by an internal or external clock source. The clock source selection and frequency selection are determined by the TCCR0B register
CS0 bits to control, see the TC0 and TC1 prescaler sections for details.

The length of the counter is 8 bits and supports bidirectional counting. The waveform generation mode is the operating mode of the counter located by TCCR0A and
The WGM0 bit of the TCCR0B register is controlled. According to the different operating modes, the counter for each count clock Clkt0 is real
Is cleared, plus one or minus one operation. When the count overflows, the count overflow flag TOV0 is located in the TIFR0 register
Will be set. TC0 count overflow interrupt can be generated when interrupt is enabled.

The output compare unit compares the count value TCNT0 with the values of the output compare registers OCR0A and OCR0B. When TCNT0
Equal to OCR0A or OCR0B is called a compare match, the output compare flag OCF0A in the TIFR0 register or
The OCF0B bit is set. TC0 output compare match interrupt can be generated when interrupt is enabled.
Note that in the PWM operating mode, the OCR0A and OCR0B registers are double buffered registers. In normal mode and

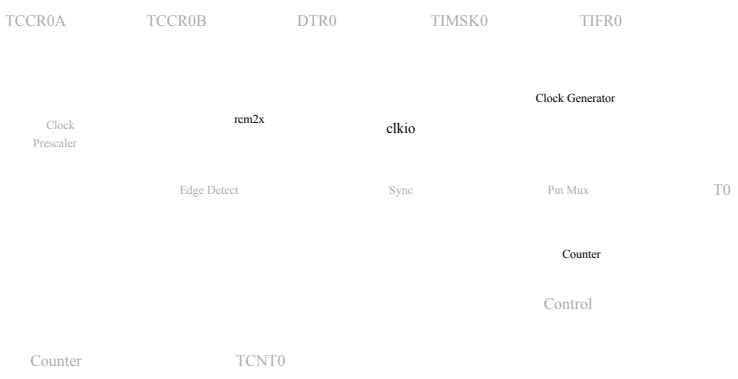


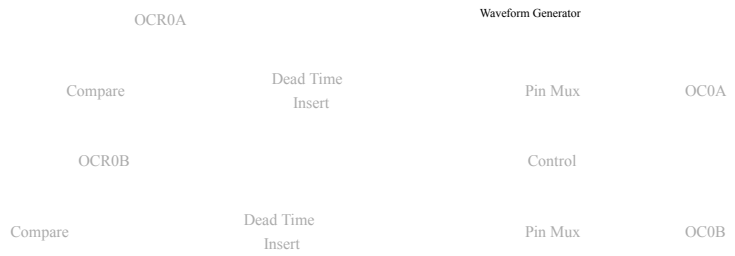


In CTC mode, the double buffering function is disabled. When the count reaches the maximum or minimum value, the value in the buffer register is updated synchronously
To compare registers OCR0A and OCR0B. See the description of the working mode section.

The waveform generator generates and controls the output mode control according to the waveform generation mode and uses the compare match and the count overflow to generate
Output compare waveform signals OC0A and OC0B. The specific way of generating is described in the working mode and register section description. Take the output ratio
When the waveform signals OC0A and OC0B are output to the corresponding pins, the data direction register of the pin must also be set to
Out.

The picture below shows the internal structure of TC0. TC0 contains one count clock generation unit, one 8-bit counter, two inputs
A comparison unit and two waveform generation control units.






TC0 structure diagram

## Operating mode

Timing counter 0 has four different modes of operation, including normal mode (Normal), compare match clear (CTC)
Mode, fast pulse width modulation (FPWM) mode and phase correction pulse width modulation (PCPWM) mode,
Generates the mode control bit WGM0 [2: 0] to select. The four modes are described below in detail. Because there are two independent output ratios
The units are represented by "A" and "B", respectively, and the lower output "x" is used to represent the two output compare cell channels.





## Normal mode

The normal mode is the simplest mode of operation for the timer counter. The waveform generation mode control bit, WGM0 [2: 0] = 0,
The maximum value of the TOP is MAX (0xFF). In this mode, the count mode is incremented by one for each count clock
After the counter reaches the TOP overflow, it returns to BOTTOM to restart. In the same value as the count value TCNT0 becomes zero
Set the timer counter overflow flag TOV0 in the clock. The TOV0 flag in this mode is like the 9th count bit, just
Will only be set to not be cleared. The overflow interrupt service routine automatically clears the TOV0 flag, which the software can use to improve it
The resolution of the counter. There is no special case in normal mode to consider, you can always write a new count value.
The waveform of the output compare signal OC0x is obtained by setting the data direction register of the OC0x pin to output. When COM0x = 1
, The OC0x signal is flipped when a compare match occurs. The frequency of the waveform in this case can be calculated using the following formula:

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

Where N represents the prescaler factor (1, 8, 64, 256, or 1024).
The output compare unit can be used to generate interrupts, but interrupts are not recommended in normal mode, which takes too much CPU
time.

## CTC mode

When the WGM0 [2: 0] = 2 is set, the timer counter 0 enters the CTC mode and the maximum value of the count is OCR0A. in
In this mode, the count mode is incremented by one for each count clock. When the value of the counter TCNT0 equals TOP
The counter is cleared. OCR0A defines the maximum value of the count, that is, the resolution of the counter. This mode allows the user to be tolerant
Easy control matches the frequency of the output and also simplifies the operation of the external event count.
When the counter reaches the maximum value of the count, the output compare match flag OCF0 is set and the corresponding interrupt enable is set
Will be interrupted. The OCR0A register can be updated in the interrupt service routine to count the maximum value. In this mode
OCR0A does not use double buffering, the counter is updated with no prescaler or very low prescaler operation
Be careful when you are near the minimum. If the value written to OCR0A is less than the current TCNT0 value, the counter will lose one
Times match. Before the next match match occurs, the counter has to count to TOP before starting from BOTTOM
Start counting to OCR0A value. As with the normal mode, the count value returns to the BOTTOM count clock to set the TOV0 flag.
The waveform of the output compare signal OC0x is obtained by setting the data direction register of the OC0x pin to output. When COM0x = 1
, The OC0x signal is flipped when a compare match occurs. The frequency of the waveform in this case can be calculated using the following formula:

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

Where N represents the prescaler factor (1, 8, 64, 256, or 1024).
As can be seen from the formula, when the set OCR0A is 0x0 and no prescaler, you can get the maximum frequency of $f_{sys} / 2$ output
Waveform.

### Fast PWM mode

When setting WGM0 [2: 0] = 3 or 7, the timer counter 0 enters the fast PWM mode and can be used to generate high frequency PWM waveform, the maximum value TOP is MAX (0xFF) or OCR0x, respectively. Fast PWM mode and other PWM modes The difference is that it is a one-way operation. The counter is incremented from the minimum value 0x00 to TOP and then back to BOTTOM to count. When the count value TCNT0 reaches OCR0x or BOTTOM, the output compare signal OC0x is set or cleared, depending on the ratio Compared with the output mode COM0x, see the register description for details. Due to the use of one-way operation, fast PWM mode operation The frequency is twice the phase corrected PWM mode using bidirectional operation. High-frequency characteristics make the fast PWM mode suitable for work Rate adjustment, rectification, and DAC applications. High-frequency signals can reduce the size of external components (inductive capacitors, etc.) and drop Low system cost.

When the count value reaches the maximum value, the timer counter overflow flag TOV0 will be set and the compare buffer value 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 compare signal OC0x is obtained by setting the data direction register of the OC0x pin to output. The frequency of the waveform The rate can be calculated using the following formula:

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

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

When a compare match occurs between TCNT0 and OCR0x, the waveform generator sets (clears) the OC0x signal. When TCNT0 is cleared At zero, the waveform generator clears (sets) the OC0x signal to generate a PWM wave. Whereby the extreme value of OCR0x will be Generates special PWM waveforms. When OCR0x is set to 0x00, the output PWM is in every (1 + TOP) count clock There is a narrow spike. When OCR0x is set to the maximum value, the output waveform is a continuous high or low level.

### Phase correction PWM mode

When setting WGM0 [2: 0] = 1 or 5, the timer counter 0 enters the phase corrected PWM mode, the maximum value of the count TOP is MAX (0xFF) or OCR0A, respectively. The counter is bidirectional, incremented by BOTTOM to TOP, and then again Decrements to BOTTOM, and repeats this operation. When the count reaches TOP and BOTTOM, the counting direction is changed and the count value is TOP Or BOTTOM only stay on a count clock. The count value TCNT0 matches OCR0x during the increment or decrement process , The output compare signal OC0x will be cleared or set, depending on the setting of the compare output mode COM0x. With one-way operation The maximum frequency available for bi-directional operation is smaller, but its excellent symmetry is more suitable for motor control.

In phase correction PWM mode, the TOV0 flag is set when the count reaches BOTTOM. When the count reaches TOP, compare The value of the buffer is updated to the comparison value. If the interrupt is enabled, the compare buffer OCR0x can be updated in the interrupt service routine Deposit.

The waveform of the output compare signal OC0x is obtained by setting the data direction register of the OC0x pin to output. The frequency of the waveform The rate can be calculated using the following formula:

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

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

During waveform counting, when the TCNT0 matches OCR0x, the waveform generator clears (sets) the OC0x signal. in In the process of decrementing the count, when the TCNT0 matches OCR0x, the waveform generator sets (clears) the OC0x signal. thus The extreme value of OCR0x produces a special PWM wave. When OCR0x is set to the maximum or minimum value, the OC0x signal will be output Keep it low or high.

In order to ensure that the output PWM wave symmetry on both sides of the minimum value, in the absence of a comparison match, there are two cases Flip OC0x signal. The first case is when the value of OCR0x changes from the maximum value 0xFF to other data. When OCR0x Is the maximum value, the count value reaches the maximum, the output of OC0x is the same as that of the previous descending order. Hold OC0x unchanged. The value of the new OCR0x will be updated (non-0xFF), the value of OC0x will remain When the ascending order counts, a compare match occurs. At this point OC0x signal is not symmetrical with the minimum value, so need to TCNT0 to reach the maximum value when the flip OC0x signal, which does not occur when the comparison match the OC0x signal flip the first situation condition. The second case is that when TCNT0 starts counting from a value higher than OCR0x, it will lose a compare match, Thus causing the occurrence of asymmetric situations. Also need to flip OC0x signal to achieve the minimum on both sides of the symmetry.

### Dead time control

When the DTEN0 bit is set to "1", the function of inserting the dead time is enabled. The output waveform of OC0A and OC0B will be set

The channel is compared with the output generated by the waveform based on the insertion of the set dead time, the length of the time for the DTR0 register count
The number of clocks corresponding to the time value. As shown in the following figure, the dead time insertion of OC0A and OC0B is based on the comparison of channel B
Out of the waveform as a benchmark. When COM0A and COM0B are the same as "2" or "3", the waveform polarity of OC0A and the waveform polarity of OC0B



**Page 100**



Similarly, when COM0A and COM0B are "2" or "3", the waveform of OC0A is opposite to the waveform of OC0B.


Figure 1 TC0 dead time control in FPWM mode



**Page 101**


Figure 2 TC0 dead time control in PCPWM mode

When the DTEN0 bit is set to "0", the function of inserting the dead time is disabled and the output waveforms of OC0A and OC0B are Output the resulting waveform.

## High speed clock mode

In high-speed clock mode, a higher frequency clock is used as the clock source for counting to produce higher speed and higher resolution Rate of the PWM waveform. This high-frequency clock is multiplied by the output clock rc32m of the internal 32M RC oscillator produced. Therefore, before entering the high-frequency mode, you must first enable the internal 32M RC oscillator multiplier function, that is, set TCKCSR Register the F2XEN bit and wait for a certain period of time until the multiplier clock signal output is stable. Then, TCKCSR can be set The TC2XS0 bit enables the timer counter to enter the high-speed clock mode.

In this mode, the system clock is asynchronous to the high-speed clock, and some registers (see the TC0 register list) are In the high-speed clock domain, therefore, configure and read such registers are also asynchronous, the operation should pay attention.

There is no special requirement for non-continuous read / write operation of the register in the high-speed clock domain, and when a continuous read / write operation is performed, To wait for a system clock, follow these steps:

1) write register A;

2) wait for a system clock (NOP or operating system clock under the register);

3) Read or write to register A or B.

4) Wait for a system clock (NOP or operating system clock under register).

When reading a register in a high-speed clock domain, the registers other than TCNT0 can be read directly. When the counter is also When counting, the value of TCNT0 changes with the high-speed clock, and the counter can be paused (set CS0 to zero) and read TCNT0 The value.



## Register definition

TC0 register list

| register | address | Defaults | 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 |

【note】

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

### TC0 control register A-TCCR0A

TCCR0A -TC0 Control Register A

Address: 0x44 Default: 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.<br>COM0A1 and COM0A0 together constitute the compare output mode control COM0A [1: 0], with<br>To control the OC0A output waveform. If bit 1 or bit 2 of COM0A is set,<br>The output compare waveform occupies the OC0A pin, but the pin's data direction register<br>Must be set to output this waveform. In different operating modes, COM0A is the output ratio<br>The control of the waveform is also different, see the specific comparison of the output mode control table description. |
| 6 | COM0A0 | TC0 compare match A output mode control low.<br>COM0A0 and COM0A1 together make up compare output mode control COM0A [1: 0], with<br>To control the OC0A output waveform. If bit 1 or bit 2 of COM0A is set,<br>The output compare waveform occupies the OC0A pin, but the pin's data direction register<br>Must be set to output this waveform. In different operating modes, COM0A is the output ratio<br>The control of the waveform is also different, see the specific comparison of the output mode control table description. |
| 5 | COM0B1 | TC0 compare match B output mode control high.<br>COM0B1 and COM0B0 together constitute the comparison output mode control COM0B [1: 0], with<br>To control the OC0B output waveform. If bit 1 or 2 of COM0B is set, |







| | | |
|---|---|---|
| 4 | COM0B0 | The output compare waveform occupies the OC0B pin, but the pin's data direction register<br>Must be set to output this waveform. In different operating modes, COM0B is the output ratio<br>The control of the waveform is also different, see the specific comparison of the output mode control table description.<br>TC0 compare match B output mode control low.<br>COM0B0 and COM0B1 together constitute the comparison output mode control COM0B [1: 0], with<br>To control the OC0B output waveform. If bit 1 or 2 of COM0B is set,<br>The output compare waveform occupies the OC0B pin, but the pin's data direction register<br>Must be set to output this waveform. In different operating modes, COM0B is the output ratio<br>The control of the waveform is also different, see the specific comparison of the output mode control table description. |
| 3 | DOC0B | TC0 OFF Output Compare Enable Control High.<br>When the set value of the DOC0B bit is set to "1", the trigger source turns off the output compare signal OC0B<br>can. When a trigger event occurs, the hardware automatically turns off the OC0B's waveform output.<br>When the set value of the DOC0B bit is set to "0", the trigger source turns off the output compare signal OC0B<br>only. When a trigger event occurs, the OC0B's waveform output is not turned off.<br>TC0 OFF Output Compare Enable Control Low. |

| | | |
|---|---|---|
| 2 | DOC0A | When the set value of the DOC0A bit is set to "1", the trigger source turns off the output compare signal OC0A can. When a trigger event occurs, the hardware automatically turns off the OC0A waveform output. When the set value of the DOC0A bit is set to "0", the trigger source turns off the output compare signal OC0A only. When a trigger event occurs, the OC0A waveform output is not turned off. |
| 1 | WGM01 | TC0 waveform generation mode control center. WGM01 and WGM00, WGM02 together constitute the waveform generation mode control WGM0 [2: 0], control counter count mode and waveform generation mode, see waveform Generate a pattern table description. |
| 0 | WGM00 | TC0 waveform generation mode control low. WGM00 and WGM01, WGM02 together constitute the waveform generation mode control WGM0 [2: 0], control counter count mode and waveform generation mode, see waveform Generate a pattern table description. |

### TC0 control register B- TCCR0B

TCCR0B -TC0 Control Register B

Address: 0x45 Default: 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 Force Output Compare A control bit. When operating in non-PWM mode, it can be written to the forced output compare bit FOC0A "1" way to produce a comparison match. Forcing the match does not set the OCF0A target The timer will not be overloaded or cleared, but the output pin OC0A will be followed COM0A settings corresponding to the update, just like the real match has occurred. The return value of reading FOC0A is always zero. |
| 6 | FOC0B | TC0 Forces the output compare control bit. |





| | | |
|---|---|---|
| 5 | OC0AS | When operating in non-PWM mode, it can be written to the forced output compare bit FOC0B "1" way to produce a comparison match. Forcing the match does not set the OCF0B flag The timer will not be overloaded or cleared, but the output pin OC0B will be followed COM0B settings corresponding to the update, just like the real match has occurred. The return value of reading FOC0B is always zero. OC0A output port selection control bit. When the OC0AS bit is set to "0", OC0A is set The waveform is output from the pin PD6; when the OC0AS bit is set to "1", the waveform of OC0A is set Output from pin PE4 (valid under QFP32 package). |
| 4 | DTEN0 | TC0 dead time enable control bit. When the DTEN0 bit is set to "1", the dead time is enabled. OC0A and OC0B are both In the B channel comparison output generated by the waveform based on the dead time inserted into the inserted The dead time interval is determined by the count time corresponding to the DTR0 register. OC0A lose The polarity of the waveform is determined by the correspondence between COM0 and COM0B, see OC0A Insert the dead time after the waveform polarity table. When setting the DTEN0 bit to "0", the dead time insertion is disabled, OC0A and OC0B are disabled The waveform is the waveform produced by the comparison output. |
| 3 | WGM02 | TC0 waveform generation mode control high. WGM02 and WGM00, WGM01 together constitute the waveform generation mode control WGM0 [2: 0], control counter count mode and waveform generation mode, see the specific wave Form generation pattern table description. |
| 2 | CS02 | TC0 clock selection control high. Used to select the clock source for timer counter 0. |
| 1 | CS01 | TC0 clock selection control bit. |

| | | |
|---|---|---|
| 0 | CS00 | Used to select the clock source for timer counter 0.<br>TC0 clock selection control low.<br>Used to select the clock source for timer counter 0. |

| CS0 [2: 0] | description |
|---|---|
| 0 | No clock source, stop counting |
| 1 | clk sys |
| 2 | clk sys / 8, from prescaler |
| 3 | clk sys / 64, from prescaler |
| 4 | clk sys / 256, from prescaler |
| 5 | clk sys / 1024, from prescaler |
| 6 | External clock T0 pin, falling edge trigger |
| 7 | External clock T0 pin, rising edge triggered |

The following table shows the control of the output compare waveform for the non-PWM mode (ie, normal mode and CTC mode) system.

| COM0x [1: 0] | description |
|---|---|
| 0 | OC0x disconnect, general-purpose IO port operation |
| 1 | Match the OC0x signal when comparing the match |
| 2 | The OC0x signal is cleared when compare match |
| 3 | The OC0x signal is set when compare match |





The following table shows the control of the output compare waveform for the compare output mode in fast PWM mode.

| COM0x [1: 0] | description |
|---|---|
| 0 | OC0x disconnect, general-purpose IO port operation |
| 1 | Keep it |
| 2 | The OC0x signal is cleared when the compare match is set and the OC0x signal is set when the maximum match is made |
| 3 | The OC0x signal is set when the compare match is cleared and the OC0x signal is cleared when the maximum match is reached |

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

| COM0x [1: 0] | description |
|---|---|
| 0 | OC0x disconnect, general-purpose IO port operation |
| 1 | Keep it |
| 2 | The OC0x signal is cleared when the compare match is cleared in the ascending count, and the match is compared in descending order<br>Set the OC0x signal |
| 3 | The OC0x signal is set when the compare match is set in the ascending count, and the match is compared in descending order<br>Clear the OC0x signal |

The following table shows the waveform generation mode control.

| WGM0 [2: 0] | Operating mode | TOP value | Update OCR0X moments | Set TOV0 at all times |
|---|---|---|---|---|
| 0 | Normal | 0xFF | immediately | MAX |
| 1 | PCPWM | 0xFF | TOP | BOTTOM |
| 2 | CTC | OCR0A | immediately | MAX |
| 3 | FPWM | 0xFF | TOP | MAX |
| 4 | Keep it | - | - | - |
| 5 | PCPWM | OCR0A | TOP | BOTTOM |
| 6 | Keep it | - | - | - |
| 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 Enabled Mode

| DTEN0 | COM0A [1: 0] | COM0B [1: 0] | description |
|---|---|---|---|
| 0 | - | - | The OC0A signal polarity is controlled by the OC0A compare output mode |
| 1 | 0 | - | OC0A disconnect, general-purpose IO port operation |

| | | | |
|---|---|---|---|
| 1 | 1 | - | Keep it |
| 1 | 2 | 2 | The OC0A signal is the same polarity as the OC0B signal |
| | | 3 | The OC0A signal is opposite to the OC0B signal polarity |
| 1 | 3 | 2 | The OC0A signal is opposite to the OC0B signal polarity |
| | | 3 | The OC0A signal is the same polarity as the OC0B signal |

【note】:

The polarity of the OC0B signal output waveform is controlled by the OC0B compare output mode, which is the same as the unimplemented dead time mode.





TC0 control register C - TCCR0C

TCCR0C - TC0 control register C

Address: 0x49 Default: 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 enabled bit 7.<br>When the DSX07 bit is set to "1", TC1 overflows as the output signal<br>The trigger source for OC0A / OC0B is enabled. When the DOC0A / DOC0B bit is "1"<br>The rising edge of the interrupt flag register bit of the selected trigger source is automatically turned off OC0A / OC0B waveform output.<br>When the DSX07 bit is set to "0", TC1 overflows as the output signal<br>The trigger source for OC0A / OC0B is disabled. |
| 6 | DSX06 | TC0 Trigger Source Select Control Enable bit 6.<br>When the DSX06 bit is set to "1", TC2 overflows as the output signal<br>The trigger source for OC0A / OC0B is enabled. When the DOC0A / DOC0B bit is "1"<br>The rising edge of the interrupt flag register bit of the selected trigger source is automatically turned off OC0A / OC0B waveform output.<br>When the DSX06 bit is set to "0", TC2 overflows as the output signal<br>The trigger source for OC0A / OC0B is disabled. |
| 5 | DSX05 | TC0 Trigger Source Select Control Enable Bit 5.<br>When the DSX05 bit is set to "1", the pin level changes by 0 as the output compare<br>The trigger source for signal waveform OC0A / OC0B is enabled. When the DOC0A / DOC0B bit is "1"<br>, The rising edge of the interrupt flag register bit of the selected trigger source is automatically turned off OC0A / OC0B waveform output.<br>When the DSX05 bit is set to "0", the pin level changes by 0 as the output compare<br>The trigger source for signal waveform OC0A / OC0B is disabled. |
| 4 | DSX04 | TC0 trigger source selection control enabled bit 4.<br>When the DSX04 bit is set to "1", the external interrupt 0 is used to turn off the output compare signal<br>The trigger source for waveform OC0A / OC0B is enabled. When the DOC0A / DOC0B bit is "1"<br>The rising edge of the interrupt flag register bit of the selected trigger source is automatically turned off OC0A / OC0B waveform output.<br>When the DSX04 bit is set to "0", the external interrupt 0 is used to turn off the output compare signal<br>The trigger source for waveform OC0A / OC0B is disabled. |
| 3: 2 | - | Keep not used |
| | | TC0 trigger source selection control enabled bit 1.<br>When the DSX01 bit is set to "1", the analog comparator 1 is set as the output compare signal |

| | |
|---|---|
| 1 DSX01 | The waveform of the waveform OC0A / OC0B is enabled. When the DOC0A / DOC0B bit is "1" , The rising edge of the interrupt flag register bit of the selected trigger source is automatically turned off OC0A / OC0B waveform output. |





| | |
|---|---|
| 0 DSX00 | When the DSX01 bit is set to "0", the analog comparator 1 is used as the output compare signal The waveform of the waveform OC0A / OC0B is disabled. TC0 Trigger Source Select Control Enable bit 0. When the DSX00 bit is set to "1", analog comparator 0 is used as the output signal for the close output The waveform of the waveform OC0A / OC0B is enabled. When the DOC0A / DOC0B bit is "1" , The rising edge of the interrupt flag register bit of the selected trigger source is automatically turned off OC0A / OC0B waveform output. When setting the DSX00 bit to "0", analog comparator 0 is used as the output signal for the close output The waveform of the waveform OC0A / OC0B is disabled. |

The following table controls the selection of the trigger source for the waveform output.

Turn off the trigger source selection control of the OC0A / OC0B waveform output

| DOC0x | DSX0n = 1 | Trigger source | description |
|---|---|---|---|
| 0 | - | - | The DOC0x bit is "0" and the trigger source turns off the waveform output Can be forbidden |
| 1 | 0 | Analog comparator 0 | The rising edge of ACIF0 will turn off the OC0x waveform output |
| 1 | 1 | Analog comparator | The rising edge of ACIF1 will turn off the OC0x waveform output |
| 1 | 4 | External interrupt 0 | The rising edge of INTF0 will turn off the OC0x waveform output |
| 1 | 5 | Pin level change 0 | The rising edge of PCIF0 will turn off the OC0x waveform output |
| 1 | 6 | TC2 overflow | The rising edge of TOV2 will turn off the OC0x waveform output |
| 1 | 7 | TC1 overflows | The rising edge of TOV1 will turn off the OC0x waveform output |

note:

1) When DSX0n = 1 indicates that 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: 0x00

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCNT07 | TCNT0 | TCNT0 | TCNT03 | 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. Through the TCNT0 register can be directly on the counter 8 for the count value to read and write access ask. The CPU writes to the TCNT0 register will be blocked at the next timer clock cycle Only the compare match occurs, even if the timer has stopped. This allows initialization The value of the TCNT0 register matches the value of OCR0 without causing an interrupt. If the value written to TCNT0 equals or bypasses the OCR0 value, the compare match is lost, Resulting in incorrect waveform results. The timer stops counting when the clock source is not selected, but the CPU can still access TCNT0. CPU The write counter has a higher priority than the clear or add / subtract operation. |

### TC0 output compare register A-OCR0A

OCR0A - TC0 Output compare register A

Address: 0x47 Default: 0x00

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OCR0A7 | OCR0A6 | OCR0A5 | OCR0A4 | OCR0A3 | OCR0A2 | OCR0A1 | OCR0A0 OCR0A |
| 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.<br>OCR0A contains an 8-bit data that is continuously compared with the counter value TCNT0<br>More The compare match can be used to generate an output compare interrupt, or to be used on the OC0A pin<br>Produce a waveform.<br>When using PWM mode, the OCR0A register uses a double buffered register. And ordinary work<br>Mode and match clear mode, double buffering is disabled. Double buffering can be updated<br>The OCR0A register is synchronized with the count maximum or minimum time to prevent generation<br>Asymmetric PWM pulses eliminate the interference pulses.<br>When using the double buffering function, the CPU accesses the OCR0A buffer register, which inhibits double buffering<br>When the CPU can access the OCR0A itself. |

### TC0 Output compare register B- OCR0B

OCR0B - TC0 Output compare register B

Address: 0x48 Default: 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.<br>The OCR0B contains an 8-bit data that is uninterrupted compared to the counter value TCNT0<br>More The compare match can be used to generate an output compare interrupt, or to use the OC0B pin<br>To produce a waveform.<br>When using PWM mode, the OCR0B register uses a double buffered register. While the general work<br>In mode and match clear mode, the double buffering function is disabled. Double buffering can be done<br>The update OCR0B register is synchronized with the count maximum or minimum time, thus preventing<br>Only produce an asymmetric PWM pulse, eliminating the interference pulse.<br>When using the double buffering function, the CPU accesses the OCR0B buffer register, which prohibits double buffering<br>Function when the CPU access is OCR0B itself. |



### TC0 interrupt mask register - TIMSK0

TIMSK0 - TC0 interrupt mask register

Address: 0x6E Default: 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 | | Keep it. |
| 2 | OCIE0B | TC0 Output Compare B Match Interrupt Enable bit.<br>When the OCIE0B bit is "1" and the global interrupt is set, the TC0 output compares the B match interrupt Enable. When a compare match occurs, that is, when the OCF0B bit in TIFR0 is set, it is interrupted produce.<br>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.<br>When the OCIE0A bit is "1" and the global interrupt is set, the TC0 output compares the A match interrupt Enable. When a compare match occurs, that is, when the OCF0A bit in TIFR0 is set, it is interrupted produce.<br>When the OCIE0A bit is "0", the TC0 output compare A match interrupt is disabled. |
| 0 | TOIE0 | TC0 overflow interrupt enable bit.<br>When the TOIE0 bit is "1" and the global interrupt is set, the TC0 overflow interrupt is enabled. When TC0 When an overflow occurs, that is, when the TOV0 bit in TIFR is set, an interrupt is generated.<br>When the TOIE0 bit is "0", the TC0 overflow interrupt is disabled. |

### TC0 Interrupt Flag Register - TIFR0

TIFR0 - TC0 interrupt flag register

Address: 0x35 Default: 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 compare waveform signal OC0A.<br>Output compare waveform signal OC0A, software readable but not writable. The software can be disabled The OC0A bit can be read before the OC0A signal is output to its corresponding IO pin<br>Value to get the polarity of the waveform signal to be output, and can be configured by COM0A<br>Bit and set the FOC0A bit to change its polarity, to avoid the OC0A signal output<br>To its corresponding IO pin after the generation of unnecessary interference pulse. |
| 6 | OC0B | Output compare waveform signal OC0B.<br>Output compare waveform signal OC0B, software readable but not writable. The software can be disabled The OC0B bit can be read before the OC0B signal is output to its corresponding IO pin |





| | | |
|---|---|---|
| | | Value to get the polarity of the output waveform to be compared, and can be configured by COM0B<br>Bit and set the FOC0B bit to change its polarity, to avoid the OC0B signal output<br>To its corresponding IO pin after the generation of unnecessary interference pulse. |
| 5: 3 | | Keep it |
| 2 | OCF0B | TC0 Output Compare B match flag.<br>When TCNT0 is equal to OCR0B, the comparison unit gives the match signal and sets the ratio Compared with the logo OCF0B. If this time, compare the output B interrupt enable OCIE0B to "1" and all<br>When the interrupt flag is set, an output compare B interrupt is generated. Execute this interrupt service<br>OCF0B will be cleared automatically during program or "1" for OCF0B bits. |
| 1 | OCF0A | TC0 output compare A match flag.<br>When TCNT0 is equal to OCR0A, the comparison unit gives the match signal and sets the bit ratio More than OCF0A. If this time output compare A interrupt enable OCIE0A is "1" and all<br>When the interrupt flag is set, an output compare A interrupt is generated. Execute this interrupt service<br>OCF0A will be cleared automatically during program or "1" for OCF0A bits. |

| | | |
|---|---|---|
| 0 | TOV0 | TC0 overflow flag.<br>When the counter overflows, set the overflow flag TOV0. If this time overflow interrupt<br>If TOIE0 is set to "1" and the global interrupt flag is set, an overflow interrupt is generated. carried out<br>TOV0 is automatically cleared when the interrupt routine is serviced, or "1" for the TOV0 bit<br>Clear this bit. |

### DTR0 - TC0 dead time control register

DTR0 - TC0 dead time control register

Address: 0x4F Default: 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.<br>When the DTEN0 bit of the TCCR0B register is "1", OC0A and OC0B are complementary<br>Output, insert dead time control is enabled when the dead band is inserted on the OC0B channel<br>Is determined by DTR0H, and the length of time is corresponding to DTR0H count clock<br>time. |
| [3: 0] | DTR0L | TC0 dead time register is low.<br>When the DTEN0 bit of the TCCR0B register is "1", OC0A and OC0B are complementary<br>Output, insert dead time control is enabled when the dead band is inserted on the OC0A channel<br>Is determined by DTR0L, and the length of time is the time corresponding to DTR0H count clock<br>between. |



### TCKCSR - TC clock control and status register

TCKSCR - TC clock control and status register

Address: 0xEC Default: 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 | - | Keep it |
| 6 | F2XEN | RC 32M multiplier output enable control bit.<br>When the F2XEN bit is set to "1", the multiplier output of the 32M RC oscillator is enabled,<br>Output 64M high-speed clock.<br>When the F2XEN bit is set to "1", the multiplier output of the 32M RC oscillator is disabled,<br>Can not output 64M high speed clock. |
| 5 | TC2XF1 | TC high speed clock mode flag bit 1.<br>See Timer 1 Register Description. |
| 4 | TC2XF0 | TC high speed clock mode flag 0.<br>When reading the TC2XF0 bit as "1", it indicates that the timer counter 0 is operating on the high-speed clock<br>Mode, when "0", indicates that the timer counter 0 operates in system clock mode. |
| 3: 2 | - | Keep it. |
| | | TC high speed clock mode selection control bit 1. |

| | | |
|---|---|---|
| 1 | TC2XS1 | See Timer 1 Register Description.<br>TC high speed clock mode selection control bit 0.<br>When the TC2XS0 bit is set to "1", the timer counter 0 is selected to operate on the high-speed clock mode. |
| 0 | TC2XS0 | When the TC2XS0 bit is set to "0", the timer counter 0 is selected to operate on the system clock mode. |



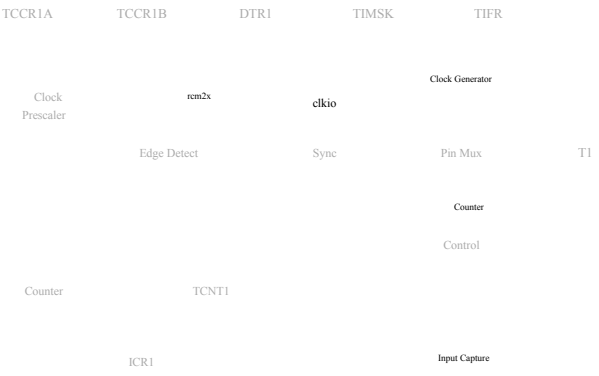
## Timer / Event Counter 1 (TMR1)

True 16-bit design allows 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 when the match matches and is automatically loaded
- Phase-corrected PWM without interference pulse
- Variable PWM cycle
- Frequency generator
- External event counter
- 4 independent interrupt sources
- PWM that supports dead time control
- 6 selectable trigger sources automatically turn off the PWM output

High-speed, high-resolution (500KHZ @ 7BIT) PWM in high-speed clock mode

### Overview




TC1 structure diagram





TC1 is a universal 16-bit timer counter module that supports PWM output and can accurately generate waveforms. TC1 package
Includes 1 16-bit counter, waveform generation mode control unit, 2 independent output compare units and 1 input capture list
yuan. At the same time, TC1 can be shared with TC0 10-bit prescaler, you can also use the 10-bit prescaler. Prescaler
On the system clock clkio or high speed clock rcm2x (internal 32M RC oscillator output clock rc32m 2 times the frequency)
The line is divided to produce the count clock Clkt1. The waveform generation mode control unit controls the operation mode and comparison output of the counter
The generation of the waveform. Depending on the operating mode, the counter is cleared for each count clock Clkt1, plus one or minus one
operating. Clkt1 can be generated by an internal clock source or an external clock source. When the counter count value TCNT1 reaches the maximum value (etc.)
At maximum 0xFFFF or fixed value or output compare register OCR1A or input capture register ICR1, defined as TOP,
When the maximum value is MAX for the difference), the counter is cleared or decremented. When the counter counts the value TCNT1
When the minimum value (equal to 0x0000, defined as BOTTOM) is reached, the counter is incremented. When the counter counts
When the value TCNT1 reaches OCR1A or OCR1B, it is also called when the compare match is cleared or the output compare signal is set
OC1A or OC1B, to generate PWM waveforms. When the dead time is enabled, the dead time set (DTR1 register) is set
The corresponding count clock number) will be inserted into the generated PWM waveform. When the input capture function is turned on, the counter
The ICR1 register will record the count value in the trigger signal trigger cycle when the trigger is started or stopped. The software can be cleared by clearing
Except when the COM1A / COM1B bit is zero to turn off the waveform output of OC1A / OC1B, or set the corresponding trigger source.
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 clock source selection and frequency selection are set by the TCCR1B register
The CS1 bits of the device are controlled, as described in the TC0 and TC1 prescaler sections.

The length of the counter is 16 bits and supports bidirectional counting. The waveform generation mode is the operating mode of the counter located by TCCR1A
And the WGM1 bit of the TCCR1B register. According to the different operating modes, the counter for each count clock Clkt1
To achieve zero, plus one or minus one operation. When the count overflows, the count overflow flag TOV1 is located in the TIFR1 register
Bit will be set. TC1 count overflow interrupt can be generated when interrupt is enabled.

The output compare unit compares the count value TCNT1 with the values of the output compare registers OCR1A and OCR1B. When TCNT1
Equal to OCR1A or OCR1B is called a compare match, the output comparison flag OCF1A in the TIFR1 register or
The OCF1B bit is set. TC1 output compare match interrupt can be generated when interrupt is enabled.
Note that in the PWM operating mode, the OCR1A and OCR1B registers are double buffered registers. In normal mode and
In CTC mode, the double buffering function is disabled. When the count reaches the maximum or minimum value, the value in the buffer register is updated synchronously
To compare registers OCR1A and OCR1B. See the description of the working mode section.

The waveform generator generates and controls the output mode control according to the waveform generation mode and uses the compare match and the count overflow to generate
Output the comparison waveform signals OC1A and OC1B. The specific way of generating is described in the working mode and register section description. Take the output ratio
When the waveform signals OC1A and OC1B are output to the corresponding pins, the data direction register of the pin must also be set
Out.

## Operating mode

Timing counter 1 has six different modes of operation, including normal mode (Normal), compare match clear (CTC)
Mode, fast pulse width modulation (FPWM) mode, phase corrected pulse width modulation (PCPWM) mode, phase frequency
Rate correction pulse width modulation (PFCPWM) mode, and input capture (ICP) mode. The waveform is generated by the mode control bit
WGM1 [3: 0] to choose. The six modes are described in detail below. Since there are two separate output comparison units, respectively





"A" and "B", with the lowercase "x" to represent the two output compare cell channels.

### Normal mode

The normal mode is the simplest mode of operation for the timer counter. The waveform generation mode control bit, WGM1 [3: 0] = 0,
The maximum value of the TOP is MAX (0xFFFF). In this mode, the count mode increments by one for each count clock,
When the counter reaches the TOP overflow, it returns to BOTTOM to restart. The same value as the count value TCNT1 becomes zero
Set the timer counter overflow flag TOV1 in the count clock. The TOV1 flag in this mode is like the 17th bit, only
Will only be set to not be cleared. The overflow interrupt service routine automatically clears the TOV1 flag, which the software can use to improve it
Timing counter resolution. There is no special case in normal mode to consider, you can always write a new count value.

The waveform of the output compare signal OC1x can be obtained by setting the data direction register of the OC1x pin to be output. When COM1x = 1
, The OC1x signal is flipped when a compare match occurs. The frequency of the waveform in this case can be calculated using the following formula:

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

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

The output compare unit can be used to generate interrupts, but interrupts are not recommended in normal mode, which takes 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 most count
The large value TOP is OCR1A, and when WGM1 [3] = 1, the maximum value TOP is ICR1. The following to WGM1 [3: 0] = 4 as an example
To describe the CTC mode In this mode, the count mode is incremented by one for each count clock, and when the counter value TCNT1
The counter is cleared when the value is equal to TOP. This mode allows the user to easily control the frequency of the compare match output and also simplify it
The operation of the external event count.

When the counter reaches TOP, the output compare match flag OCF1 is set and the corresponding interrupt enable is set
Off. The OCR1A register can be updated in the interrupt service routine. In this mode OCR1A does not use double buffering in
The counter should be careful when updating the maximum value to near minimum with no prescaler or very low prescaler operation. Such as
When the value of the write OCR1A is less than the current TCNT1 value, the counter will lose a compare match. In the next comparison
Before the match occurs, the counter has to count to MAX, and then count from BOTTOM to OCR1A. And ordinary
Mode, the count value returns to the count clock of 0x0 to set the TOV1 flag.

The waveform of the output compare signal OC1x can be obtained by setting the data direction register of the OC1x pin to be output. The frequency of the waveform
The rate can be calculated using the following formula:

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

Where N represents the prescaler factor (1, 8, 64, 256, or 1024).
As can be seen from the formula, when the set OCR1A is 0x0 and no prescaler, you can get the maximum frequency $f_{sys}$ / 2 output
Waveform.

When WGM1 [3: 0] = 12 is similar to WGM1 [3: 0] = 4, it is only necessary to replace ICCR1 with OCR1A.

**Page 115**



### Fast PWM mode

When the WGM1 [3: 0] = 5, 6, 7, 14 or 15 is set, the timer counter 1 enters the fast PWM mode with the largest count
The values TOP are 0xFF, 0x1FF, 0x3FF, ICR1, or OCR1A, respectively, and can be used to generate high frequency PWM waveforms. fast
The PWM mode differs from other PWM modes in that it is a one-way operation. The counter is returned from the BOTTOM to the TOP
To BOTTOM to count again. When the count value TCNT1 reaches TOP or BOTTOM, the output compare signal OC1x is set
Bit or clear, depending on the setting of the compare output mode COM1, see the register description for details. Due to the use of one-way operation, fast
The operating frequency of the fast PWM mode is twice the phase-corrected PWM mode using bi-directional operation. High frequency characteristics make it fast
The PWM mode is ideal for power regulation, rectification, and DAC applications. High-frequency signals can reduce external components (inductive capacitance)
Etc.), thereby reducing system cost.

When the count reaches TOP, the timer counter overflow flag TOV1 will be set and the compare buffer value will be updated to
Compare values. If the interrupt is enabled, the OCR1A register can be updated in the interrupt service routine.

The waveform of the output compare signal OC1x can be obtained by setting the data direction register of the OC1x pin to be output. The frequency of the waveform
The rate can be calculated using the following formula:

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

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

When TCNT1 and OCR1x compare match, the waveform generator sets (clears) the OC1x signal. When TCNT1 is cleared
At zero, the waveform generator clears (sets) the OC1x signal to generate a PWM wave. Whereby the extreme value of OCR1x will be
Generates special PWM waveforms. When OCR1x is set to 0x00, the output PWM is in each (1 + TOP) count clock
There is a narrow spike. When OCR1x is set to TOP, the output waveform is a continuous high or low level. Such as
If OCR1A is used as TOP and COM1A = 1 is set, the output compare signal OC1A will generate a duty cycle of 50%
wave.

### Phase correction PWM mode

When setting WGM0 [3: 0] = 1, 2, 3, 10 or 11, the timer counter 1 enters the phase correction PWM mode.
The maximum value of the number TOP is 0xFF, 0x1FF, 0x3FF, ICR1 or OCR1A. The counter is operated in two directions by BOTTOM
Increment to TOP, and then decrement it to BOTTOM, and repeat this operation. The count is changed when the count reaches TOP and BOTTOM
Number of directions, the count value in TOP or BOTTOM only stay a count clock. In the process of increment or decrement, the count value
When TCNT1 matches OCR1x, the output compare signal OC1x will be cleared or set, depending on the compare output mode COM1
setting. Compared with one-way operation, the maximum frequency available for bi-directional operation is small, but its excellent symmetry is more suitable for electricity
Machine control.

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

The output compare signal OC1x waveform can be obtained by setting the data direction register of the OC1x pin to output. The frequency of the waveform can be
Use the following formula to calculate:

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

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



**Page 116**



During the up-counting process, the waveform generator clears (sets) the OC1x signal when TCNT1 matches OCR1x. in
In the process of decrementing the count, when the TCNT1 matches OCR1x, the waveform generator sets (clears) the OC1x signal. thus
The extreme value of OCR1x will produce a special PWM wave. When OCR1x is set to TOP or BOTTOM, OC1x signal output will be
Keep it low or high. If OCR1A is used as TOP and COM1A = 1 is set, output compare signal OC1A will

Generates a PWM wave with a duty cycle of 50%.

In order to ensure that the output PWM wave symmetry on both sides of the BOTTOM, in the absence of a comparison match, there are two cases
Will flip the OC1x signal. The first case is when the value of OCR1x changes from TOP to other data. When OCR1x is
TOP, when the count value reaches TOP, the output of OC1x is the same as that of the previous descending order, that is, the OC1x
constant. The value of the new OCR1x (not TOP) is updated and the OC1x value remains until the ascending order
Count when the match occurs and flip. The OC1x signal is not centered at the minimum, so it needs to be in TCNT1
When the maximum value is reversed, the OC1x signal is flipped, which is the first case when the OC1x signal is flipped without a compare match. second
The case is that when TCNT1 starts counting from a value higher than OCR1x, it will lose a compare match, causing
The generation of asymmetric situations. Also need to flip OC1x signal to achieve the minimum on both sides of the symmetry.

### Phase frequency correction PWM mode

When setting WGM0 [3: 0] = 8 or 9, the timer counter 1 enters the phase frequency correction PWM mode, counting the maximum
The value TOP is ICR1 or OCR1A, respectively. The counter is bidirectional, incremented by BOTTOM to TOP, and then decremented
To BOTTOM, and repeat this operation. When the count reaches TOP and BOTTOM, the counting direction is changed and the count value is TOP or
BOTTOM only stay on a count clock. In the process of increment or decrement, when the count value TCNT1 matches OCR1x,
The output compare signal OC1x will be cleared or set, depending on the setting of the compare output mode COM1. And one-way operation phase
The maximum frequency available for bi-directional operation is smaller, but its excellent symmetry is more suitable for motor control.

Phase frequency correction In PWM mode, the TOV1 flag is set when the count reaches BOTTOM, and the compare buffer
Value is updated to the comparison value, the time for updating the comparison value is the phase frequency correction PWM mode and the phase correction PWM mode is the most
Big difference. If the interrupt is enabled, the compare buffer OCR1x register can be updated in the interrupt service routine. When the CPU changes
Change the TOP value that ORC1A or ICR1 value, you must ensure that the new TOP value is not less than the TOP value already used, otherwise
The match will not happen again.

The output compare signal OC1x waveform can be obtained by setting the data direction register of the OC1x pin to output. The frequency of the waveform can be
Use the following formula to calculate:

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

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

During the up-counting process, the waveform generator clears (sets) the OC1x signal when TCNT1 matches OCR1x. in
In the process of decrementing the count, when the TCNT1 matches OCR1x, the waveform generator sets (clears) the OC1x signal. thus
The extreme value of OCR1x will produce a special PWM wave. When OCR1x is set to TOP or BOTTOM, OC1x signal output will be
Keep it low or high. If OCR1A is used as TOP and COM1A = 1 is set, output compare signal OC1A will
Generates a PWM wave with a duty cycle of 50%.

Since the OCR1x register is updated at the BOTTOM time, the TOP value is on both sides in ascending and descending order.
Kind of symmetry waveforms where both the frequency and the phase are correct.







When using the fixed TOP value, it is best to use the ICR1 register as the TOP value, that is, set WGM1 [3: 0] = 8, then OCR1A
The register is only used to generate the PWM output. If you want to generate frequency changes in the PWM wave, you must change the TOP value,
OCR1A double buffering feature will be more suitable for this application.

### Enter the capture mode

The input capture is used to capture an external event and give it a time stamp to indicate when the event occurred,
The counting mode is performed, but the waveform generation mode using the ICR1 value as the count TOP value is to be removed.

Trigger signals for external events are entered by pin ICP1 and can also be implemented by analog comparator units. When pin ICP1
The logic level changes, or the analog ACO level of the analog comparator changes, and this level change is entered
Capture unit is captured, the input capture is triggered, then the 16-bit count value TCNT1 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
Interrupted.

By setting the analog compare control with the status register ACSR analog compare 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 changing the trigger After the source, ICF1 must be cleared once to avoid the wrong result.

The input capture signal is passed to an edge detector via an optional noise suppressor, and the control bit is selected according to the input capture ICES1 configuration, to see whether the detected edge to meet the trigger conditions. The noise suppressor is a simple digital filter, right The input signal is sampled four times, and the output is fed to the edge detector only when the four samples are equal. Noise suppression The controller is enabled or disabled by the ICNC1 bit of the TCCR1B register.

When the input capture function is used, the ICR1 register value should be read as early as possible when ICF1 is set, since the next time The value of ICR1 will be updated after the 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 input capture time stamp can be used to calculate the frequency, duty cycle and other characteristics of the signal, as well as to trigger the event creation date Mind When measuring the duty cycle of an external signal, it is required to change the trigger edge after each capture. Therefore, after reading the ICR1 value, Quickly change the edge of the triggered signal.

### Dead time control

When the DTEN1 bit is set to "1", the function of inserting the dead time is enabled and the output waveform of OC1A and OC1B will be set to B Channel comparison output generated by the waveform based on the insertion of the set dead time, the length of the time for the DTR1 register count The number of clocks corresponding to the time value. As shown in the following figure, the dead time insertion of OC1A and OC1B is the comparison of channel B Output waveform as a reference. When COM1A and COM1B are the same as "2" or "3", OC1A's waveform polarity and OC1B waveform pole The COM1A and COM1B are "2" or "3", OC1A waveform and OC1B waveform polarity is opposite.






Figure 3 TC1 dead time control in FPWM mode

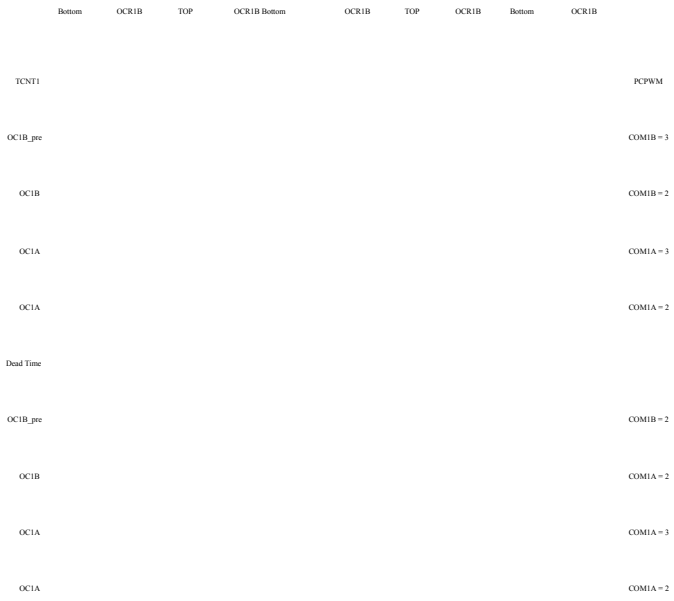



Figure 4 TC1 dead time control in PCPWM mode





When the DTEN1 bit is set to "0", the function of inserting the dead time is disabled and the output waveforms of OC1A and OC1B are
Output the resulting waveform.

### High speed counting mode

In high-speed clock mode, a higher frequency clock is used as the counting clock source to generate higher speed and higher
Resolution of the PWM waveform. This high-frequency clock is performed by multiplying the output clock rc32m of the internal 32M RC oscillator by 2 times
To produce. Therefore, before entering the high-frequency mode, you must first enable the internal 32M RC oscillator multiplier function, that is set
TCKCSR register of the F2XEN bit and wait for a certain period of time until the multiplier clock signal output is stable. Then, TCKCSR can be set
The TC2XS1 bit enables the timer counter to enter the high-speed clock mode.

In this mode, the system clock is asynchronous to the high-speed clock, and some registers (see the TC1 register list) are operating
In the high-speed clock domain, therefore, when configuring and reading such registers is also asynchronous, the operation should pay attention.

There is no special requirement for non-sequential read and write operations on registers in high-speed clock domains, and for continuous read and write operations
To wait for a system clock, follow these steps:
5) write register A;
6) wait for a system clock (NOP or operating system clock under the register);
7) Read or write to register A or B.
8) Wait for a system clock (NOP or register under operating system clock).

When a register is read in a high-speed clock domain, a register with a width of 8 bits can be read directly and 16 bits are read
Register value (OCR1A, OCR1B, ICR1, TCNT1), the value of the lower register is read first, waiting for a system clock
, Then read the value of the high register, and read the value of TCNT1, when the counter is still counting, TCNT1
Value will change with the high-speed clock, you can pause the counter (set CS1 to zero) and then read the value of TCNT1.

When reading OCR1A, OCR1B and ICR1, follow these steps:
1) Read OCR1AL / OCR1BL / ICR1L;
2) wait for a system clock (NOP);
3) Read OCR1AH / OCR1BH / ICR1H.

When reading TCNT1, follow these steps:

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 | Defaults | 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 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 | The timer counter interrupts the mask register |
| TIFR1 | 0x36 | 0x00 | Timer counter interrupt flag register |
| TCKCSR1 | 0xEC | 0x00 | TC1 clock control status register |

【note】

The register with "*" operates on the system clock and the high-speed clock domain. When the register with "*" only works on the system Under the clock domain.

### TCCR1A -TC1 control register A

TCCR1A -TC1 control register A

Address: 0x80 Default: 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.<br>COM1A1 and COM1A0 form COM1A [1: 0] to control the output compare waveform<br>OC1A. If the 1 or 2 bits of COM1A are set, the output compare waveform is occupied<br>According to OC1A pin, but the pin of the data direction register must be set high<br>Output this waveform. In different operating modes, COM1A controls the output compare waveform<br>System is also different, see the specific comparison of the output mode control table description. |

| | | |
|---|---|---|
| 6 | COM1A0 | Compare match output A mode control low.<br>COM1A1 and COM1A0 form COM1A [1: 0] to control the output compare waveform<br>OC1A. If the 1 or 2 bits of COM1A are set, the output compare waveform is occupied<br>According to OC1A pin, but the pin of the data direction register must be set high |





| | | |
|---|---|---|
| | | Output this waveform. In different operating modes, COM1A controls the output compare waveform<br>System is also different, see the specific comparison of the output mode control table description. |
| 5 | COM1B1 | Compare match output B mode control high.<br>COM1B1 and COM1B0 form COM1B [1: 0] to control the output compare waveform<br>OC1B. If bit 1 or bit 2 of COM1B is set, the output compare waveform is occupied<br>According to the OC1B pin, but the pin of the data direction register must be set high<br>Output this waveform. In different operating modes, COM1B controls the output compare waveform<br>System is also different, see the specific comparison of the output mode control table description. |
| 4 | COM1B0 | Compare match output B mode control low.<br>COM1B1 and COM1B0 form COM1B [1: 0] to control the output compare waveform<br>OC1B. If bit 1 or bit 2 of COM1B is set, the output compare waveform is occupied<br>According to the OC1B pin, but the pin of the data direction register must be set high<br>Output this waveform. In different operating modes, COM1B controls the output compare waveform<br>System is also different, see the specific comparison of the output mode control table description. |
| 3: 2 | - | Keep the same |
| 1 | WGM11 waveform generation mode control sub-low. | WGM11 and WGM13, WGM12, WGM10 together constitute the waveform generation mode control<br>System WGM1 [3: 0], control counter count mode and waveform generation mode, specific<br>See waveform generation mode table description. |
| 0 | WGM10 waveform generation mode control the lowest bit. | WGM10 and WGM13, WGM12, WGM11 together constitute the waveform generation mode control<br>System WGM1 [3: 0], control counter count mode and waveform generation mode, specific<br>See waveform generation mode table description. |

The following table shows the non-PWM mode (ie, normal mode and CTC mode), compare the output mode to the output compare waveform control.

| COM1x [1: 0] | description |
|---|---|
| 0 | OC1x disconnect, general IO port operation |
| 1 | Compare the OC1x signal when matching |
| 2 | The OC1x signal is cleared when compare match |
| 3 | Set OC1x signal when compare match |

The following table shows the control of the output compare waveform for the compare output mode in fast PWM mode.

| COM1x [1: 0] | description |
|---|---|
| 0 | OC1x disconnect, general IO port operation |
| 1 | When WGM1 is 15: The OC1A signal is flipped when compare match, OC1B is off<br>When WGM1 is other value: OC1x is disconnected, general purpose IO port operation |
| 2 | The OC1x signal is cleared when the compare match is set and the OC1x signal is set when the maximum match is made |
| 3 | The OC1x signal is set when the compare match is cleared and the OC1x signal is cleared when the maximum match is made |

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

| COM1x [1: 0] | description |
|---|---|
| 0 | OC1x disconnect, general IO port operation |
| 1 | When WGM1 is 9 or 11: OC1A signal is flipped when compare match, OC1B is off<br>When WGM1 is other value: OC1x is disconnected, general purpose IO port operation |

| | |
|---|---|
| 2 | Compare the match in the ascending count to clear the OC1x signal, compare the match in descending order Bit OC1x signal |
| 3 | The higher order count matches the OC1x signal, and the compare match is cleared in descending order Zero OC1x signal |

TCCR1B -TC1 control register B

TCCR1B -TC1 control register B

Address: 0x81 Default: 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 | The input capture noise suppressor enables the control bit.<br>When the ICNC1 bit is set to "1", the input capture noise suppressor is enabled,<br>The input of the foot ICP1 is filtered and the input signal is valid when four consecutive samples are equal,<br>This function causes the input capture to be delayed by four clock cycles.<br>When the ICNC1 bit is set to "0", the input capture noise suppressor is disabled,<br>The input of foot ICP1 is directly effective. |
| 6 | ICES1 | The input capture trigger edge selects the control bit.<br>When setting the ICES1 bit to "1", the rising edge of the selected level triggers the input capture; when set<br>When the ICES1 bit is set to "0", the falling edge of the selected level triggers the input capture.<br>When an event is captured, the counter value is copied to the ICR1 register |
| 5 | - | Set the input capture flag ICF1. If the interrupt is enabled, an input capture interrupt is generated.<br>Keep it. |
| 4 | WGM13 | Waveform Generation Mode Control High.<br>WGM13 and WGM12, WGM11, WGM10 together constitute the waveform generation mode control<br>WGM1 [3: 0], control counter count mode and waveform generation mode, see the specific wave<br>Form generation pattern table description. |
| 3 | WGM12 | Waveform generation mode control times high.<br>WGM12 and WGM13, WGM11, WGM10 together constitute the waveform generation mode control<br>WGM1 [3: 0], control counter count mode and waveform generation mode, see the specific wave<br>Form generation pattern table description. |
| 2 | CS12 | Clock select control high. Used to select the clock source for timer counter 1. |
| 1 | CS11 | Clock selection control bit. Used to select the clock source for timer counter 1. |
| 0 | CS10 | Clock select control low.<br>Used to select the clock source for timer counter 1. |

| CS1 [2: 0] | description |
|---|---|
| 0 | No clock source, stop counting |
| 1 | clk $_{sys}$ |
| 2 | clk $_{sys}$ / 8, from prescaler |
| 3 | clk $_{sys}$ / 64, from prescaler |
| 4 | clk $_{sys}$ / 256, from prescaler |
| 5 | clk $_{sys}$ / 1024, from prescaler |





| | |
|---|---|
| 6 | External clock T1 pin, falling edge trigger |
| 7 | External clock T1 pin, rising edge triggered |

The following table shows the waveform generation mode control.

| WGM1 [3: 0] working mode | | TOP value | Update the time of TOCR0 when the OCR0 is updated | |
|---|---|---|---|---|
| 0 | Normal | 0xFFFF | immediately | 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 | immediately | MAX |
| 5 | 8-bit FPWM | 0x00FF | BOTTOM | TOP |
| 6 | 9 bits FPWM | 0x01FF | BOTTOM | TOP |
| 7 | 10 bits 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 | immediately | MAX |
| 13 | Keep it | - | - | - |
| 14 | FPWM | ICR1 | TOP | TOP |
| 15 | FPWM | OCR1A | TOP | TOP |

### TCCR1C -TC1 control register C

TCCR1C -TC1 control register C

Address: 0x82 Default: 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 operating in non-PWM mode, it is possible to write "1" to the forced output compare bit FOC1A
To produce a comparison match. Forcing the match does not set the OCF1A flag, too
The timer is not reloaded or cleared, but the output pin OC1A will be followed by COM1A
Set the corresponding update, just like the real match has occurred.
When working in PWM mode, write the TCCR1A register to clear it.
The return value of the read FOC1A is always zero.

6 FOC1B Force Output Compare B.
When operating in non-PWM mode, it is possible to write "1" to the forced output compare bit FOC1B
To produce a comparison match. Forcing the match match does not set the OCF1B flag, too
The timer is not reloaded or cleared, but the output pin OC1B will be followed by COM1B
Set the corresponding update, just like the real match has occurred. Work in PWM mode
, Write the TCCR1A register to clear it. Read the return value of FOC1B



Straight zero.

5 DOC1B TC1 OFF Output Compare Enable Control High.
When the DOC1B bit is set to "1", the trigger source turns off the output compare signal OC1B
can. When a trigger event occurs, the hardware automatically turns off the OC1B's waveform output.
When the DOC1B bit is set to "0", the trigger source turns off the output compare signal OC1B
only. When a trigger event occurs, the OC1B's waveform output is not turned off.

4 DOC1A TC1 OFF Output Compare Enable Control Low.
When the DOC1A bit is set to "1", the trigger source turns off the output compare signal OC1A
can. When a trigger event occurs, the hardware automatically turns off the OC1A waveform output.
When the DOC1A bit is set to "0", the trigger source turns off the output compare signal OC1A
only. When a trigger event occurs, the OC1A's waveform output is not turned off.

3 DTEN1 TC1 dead time enable control bit.
When the DTEN1 bit is set to "1", the dead time is enabled. OC1A and OC1B are both

| | | |
|---|---|---|
| | | In the B-channel comparison output generated by the waveform based on the insertion of the dead time, the inserted dead |
| | | The zone time interval is determined by the count time corresponding to the DTR1 register. OC1A output wave |
| | | The polarity of the shape is determined by the correspondence between COM1A and COM1B. See OC1A insertion |
| | | Dead time after the waveform polarity table shown. |
| | | When setting DTEN1 bit to "0", disable dead time insertion, OC1A and OC1B |
| | | The waveform is the waveform produced by the comparison output. |
| 2: 0 | - | Keep it |

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 Enabled Mode

| DTEN1 | COM1A [1: 0] | COM1B [1: 0] | description |
|---|---|---|---|
| 0 | - | - | The OC1A signal polarity is controlled by the OC1A compare output mode |
| 1 | 0 | - | OC1A disconnect, general IO port operation |
| 1 | 1 | - | Keep it |
| 1 | 2 | 2 | The OC1A signal is the same polarity as the OC1B signal |
| | | 3 | The OC1A signal is opposite to the polarity of the OC1B signal |
| 1 | 3 | 2 | The OC1A signal is opposite to the polarity of the OC1B signal |
| | | 3 | The OC1A signal is the same polarity as the OC1B signal |

【note】:
The polarity of the OC1B signal output waveform is controlled by the OC1B compare output mode, which is the same as the unimplemented dead time mode.

### TCCR1D -TC1 control register D

TCCR1D- TC control register D

Address: 0x83 Default: 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 | description |
|---|---|---|
| 7 | DSX17 | TC1 trigger source selection control enabled bit 7.<br>When setting the DSX17 bit to "1", TC0 overflows as the output compare signal is turned off |





| | | |
|---|---|---|
| | | The trigger source for waveform OC1A / OC1B is enabled. When the DOC1A / DOC1B bit is "1" |
| | | , The rising edge of the interrupt flag register bit of the selected trigger source is automatically turned off |
| | | OC1A / OC1B waveform output. |
| | | When setting the DSX17 bit to "0", TC0 overflows as the output compare signal is turned off |
| | | The trigger source for waveform OC1A / OC1B is disabled. |
| | | TC1 Trigger Source Select Control Enable bit 6. |
| | | When the DSX16 bit is set to "1", TC2 overflows as the output compare signal is turned off |
| | | The trigger source for waveform OC1A / OC1B is enabled. When the DOC1A / DOC1B bit is "1" |
| 6 | DSX16 | , The rising edge of the interrupt flag register bit of the selected trigger source is automatically turned off |
| | | OC1A / OC1B waveform output. |
| | | When the DSX16 bit is set to "0", TC2 overflows as the output compare signal is turned off |
| | | The trigger source for waveform OC1A / OC1B is disabled. |
| | | TC1 Trigger Source Select Control Enable Bit 5. |
| | | When the DSX15 bit is set to "1", the pin level changes by 1 as the output ratio is turned off |
| | | The trigger source for the signal waveform OC1A / OC1B is enabled. When DOC1A / DOC1B bit |
| 5 | DSX15 | When "1", the rising edge of the interrupt flag register bit of the selected trigger source is automatically enabled |
| | | Turn off the OC1A / OC1B waveform output. |
| | | When the DSX15 bit is set to "0", the pin level changes by 1 as the output ratio is turned off |
| | | The trigger source for the signal waveform OC1A / OC1B is disabled. |
| | | TC1 trigger source selection control enabled bit 4. |
| | | When the DSX14 bit is set to "1", the external interrupt 1 is set as the output compare signal |
| | | The waveform of the waveform OC1A / OC1B is enabled. When the DOC1A / DOC1B bit is |

| | | |
|---|---|---|
| 4 | DSX14 | "1", the rising edge of the interrupt flag register bit of the selected trigger source is automatically turned off<br>Closed OC1A / OC1B waveform output.<br>When the DSX14 bit is set to "0", the external interrupt 1 is set as the output compare signal<br>The waveform of the waveform OC1A / OC1B is disabled. |
| 3: 2 | - | Keep it |
| 1 | DSX11 | TC1 Trigger Source Select Control Enable bit 1.<br>When setting the DSX11 bit to "1", analog comparator 1 is used as the output compare for off<br>The trigger source for signal waveform OC1A / OC1B is enabled. When the DOC1A / DOC1B bit is<br>"1", the rising edge of the interrupt flag register bit of the selected trigger source is automatically turned off<br>Closed OC1A / OC1B waveform output.<br>When the DSX11 bit is set to "0", analog comparator 1 is used as the output compare for off<br>The trigger source for signal waveform OC1A / OC1B is disabled. |
| 0 | DSX10 | TC1 Trigger Source Select Control Enable Bit 0.<br>When setting the DSX10 bit to "1", analog comparator 0 is used to turn off the output compare<br>The trigger source for signal waveform OC1A / OC1B is enabled. When the DOC1A / DOC1B bit is<br>"1", the rising edge of the interrupt flag register bit of the selected trigger source is automatically turned off<br>Closed OC1A / OC1B waveform output.<br>When setting the DSX10 bit to "0", analog comparator 0 is used to turn off the output compare<br>The trigger source for signal waveform OC1A / OC1B is disabled. |

The following table controls the selection of the trigger source for the waveform output.

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



**Page 126**



| DOC1x | DSX1n = 1 | Trigger source | description |
|---|---|---|---|
| 0 | - | - | The DOC1x bit is "0" and the trigger source turns off the waveform output<br>Can be forbidden |
| 1 | 0 | Analog comparator 0 | The rising edge of ACIF0 will turn off the OC1x waveform output |
| 1 | 1 | Analog comparator | The rising edge of ACIF1 will turn off the OC1x waveform output |
| 1 | 4 | External interrupt 1 | The rising edge of INTF1 will turn off the OC1x waveform output |
| 1 | 5 | Pin level change | The rising edge of PCIF1 will turn off the OC1x waveform output |
| 1 | 6 | TC2 overflow | The rising edge of TOV2 will turn off the OC1x waveform output |
| 1 | 7 | TC0 overflows | The rising edge of TOV0 will turn off the OC1x waveform output |

【note】:

When DSX1n = 1 indicates that 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 Count Value Register Low byte

Address: 0x84 Default: 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 | TC1 The low byte of the count value.<br>TCNT1H and TCNT1L are combined together to form TCNT1 through the TCNT1 register<br>You can read and write directly to the 16-bit count of the counter. Read and write 16 send<br>The register requires two operations. When writing 16-bit TCNT1, write TCNT1H first. read<br>16-bit TCNT1, TCNT1L should be read first.<br>The CPU writes to the TCNT1 register are blocked by the next timer clock cycle<br>The compare match occurs even if the timer has stopped. This allows initialization<br>The value of the TCNT1 register matches the value of OCR1x without causing an interrupt.<br>If the value written to TCNT1 equals or bypasses the OCR1x value, the compare match will be made |

Lost, resulting in incorrect waveform 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 the clear or add / subtract operation.

### TCNT1H -TC1 count register high byte

TCNT1H -TC1 count register high byte

Address: 0x85 Default: 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 | TC1 The high byte of the count value.<br>TCNT1H and TCNT1L are combined together to form TCNT1 through the TCNT1 register |





You can read and write directly to the 16-bit count of the counter. Read and write 16 send
The register requires two operations. When writing 16-bit TCNT1, write TCNT1H first. read
16-bit TCNT1, TCNT1L should be read first.
The CPU writes to the TCNT1 register are blocked by the next timer clock cycle
The compare match occurs even if the timer has stopped. This allows initialization
The value of the TCNT1 register matches 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 made
Lost, resulting in incorrect waveform 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 the clear or add / subtract operation.

### ICR1L -TC1 input capture register low byte

ICR1L -TC1 input capture register low byte

Address: 0x86 Default: 0x00

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ICR1L6 | ICR1L5 | ICR1L4 | ICR1L4 | ICR1L2 | ICR1L1 | ICR1L1 | ICR1L1 |
| R / W | R / W | R / W | R / W | R / W | R / W | R / W | R / W | R / W |

| Bit | Name | Description |
|---|---|---|
| 7: 0 | ICR1L | TC1 Enter the low byte of the capture value.<br>ICR1H and ICR1L combine to form 16-position ICR1. Read and write 16-bit register<br>The device needs two operations. When writing 16-bit ICR1, write ICR1H first. Read 16 bits<br>ICR1 should read ICR1L first. When the input capture is triggered, the count value TCNT1<br>It will be updated to the ICR1 register. The ICR1 register can also be used to define the count<br>The TOP value. |

### ICR1H -TC1 input capture register high byte

ICR1H -TC1 input capture register high byte

Address: 0x87 Default: 0x00

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ICR1H7 | ICR1H6 | ICR1H5 | ICR1H4 | ICR1H3 | ICR1H2 | ICR1H1 | ICR1H1 |
| 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 Enter the high byte of the capture value.<br>ICR1H and ICR1L combine to form 16-position ICR1. Read and write 16-bit register<br>The device needs two operations. When writing 16-bit ICR1, write ICR1H first. Read 16 bits<br>ICR1 should read ICR1L first. When the input capture is triggered, the count value TCNT1<br>It will be updated to the ICR1 register. The ICR1 register can also be used to define the count<br>The TOP value. |

### OCR1AL -TC1 Output compare register A low byte

OCR1AL -TC1 Output compare register A low byte

Address: 0x88 Default: 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 | Output the lower byte of compare register A.<br>OCR1AL and OCR1AH combine to form 16-position OCR1A. Read and write 16<br>The bit register requires two operations. When writing 16-bit OCR1A, write OCR1AH first.<br>When reading 16-bit OCR1A, read OCR1AL first.<br>OCR1A is compared with the counter value TCNT1 without interruption. Compare match can be<br>To generate an output compare interrupt, or to generate a wave on the OC1A pin<br>shape.<br>When using PWM mode, the OCR1A register uses a double buffered register. and<br>In normal operation mode and match clear mode, double buffering function is disabled. double<br>Buffer You can update the OCR1A register with the count maximum or minimum time<br>Step up, thus preventing the generation of asymmetric PWM pulses, eliminating the interference pulse<br>Red<br>When using the double buffering function, the CPU accesses the OCR1A buffer register, which is disabled<br>When the double buffering function, 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: 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 | Outputs the high byte of compare register A.<br>OCR1AL and OCR1AH combine to form 16-position OCR1A. Read and write 16<br>The bit register requires two operations. When writing 16-bit OCR1A, write OCR1AH first.<br>When reading 16-bit OCR1A, read OCR1AL first.<br>OCR1A is compared with the counter value TCNT1 without interruption. Compare match can be<br>To generate an output compare interrupt, or to generate a wave on the OC1A pin<br>shape.<br>When using PWM mode, the OCR1A register uses a double buffered register. and<br>In normal operation mode and match clear mode, double buffering function is disabled. double<br>Buffer You can update the OCR1A register with the count maximum or minimum time<br>Step up, thus preventing the generation of asymmetric PWM pulses, eliminating the interference pulse<br>Red<br>When using the double buffering function, the CPU accesses the OCR1A buffer register, which is disabled<br>When the double buffering function, 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: 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 | Output the lower byte of compare register B.<br>OCR1BL and OCR1BH combine to form 16-bit OCR1B. Read and write 16<br>The bit register requires two operations. When writing 16-bit OCR1B, write OCR1BH first.<br>When reading 16-bit OCR1B, you should read OCR1BL first.<br>OCR1B is compared with the counter value TCNT1 without interruption. Compare matches can<br>Used to generate an output compare interrupt, or to generate a waveform on the OC1B pin.<br>When using PWM mode, the OCR1B register uses a double buffered register. And general<br>In the mode of operation and match clear mode, the double buffering function is disabled. Double slow<br>The update can update the OCR1B register with the count maximum or minimum time<br>To prevent the generation of asymmetric PWM pulses, eliminating the interference pulse.<br>When using the double buffering function, the CPU accesses the OCR1B buffer register, which is disabled<br>Buffer function when the CPU access is OCR1B itself. |

### OCR1BH -TC1 Output compare register B high byte

OCR1BH -TC1 Output compare register B high byte

Address: 0x8B Default: 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 | Outputs the high byte of compare register B.<br>OCR1BL and OCR1BH combine to form 16-bit OCR1B. Read and write 16<br>The bit register requires two operations. When writing 16-bit OCR1B, write OCR1BH first.<br>When reading 16-bit OCR1B, you should read OCR1BL first.<br>OCR1B is compared with the counter value TCNT1 without interruption. Compare match can be<br>To generate an output compare interrupt, or to generate a wave on the OC1B pin<br>shape.<br>When using PWM mode, the OCR1B register uses a double buffered register. and<br>In normal operation mode and match clear mode, double buffering function is disabled. double<br>Buffer You can update the OCR1B register with the count maximum or minimum time<br>Step up, thus preventing the generation of asymmetric PWM pulses, eliminating the interference pulse<br>Red<br>When using the double buffering function, the CPU accesses the OCR1B buffer register, which is disabled<br>When the double buffering function, the CPU accesses the OCR1B itself. |



### TIMSK1 - TC1 interrupt mask register

TIMSK1 - TC1 interrupt mask register

Address: 0x6F Default: 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 | - | Keep it. |
| 5 | TICIE1 | TC1 input capture interrupt enable bit.<br>When the ICIE1 bit is "1" and the global interrupt is set, the TC1 input capture interrupt is enabled<br>can. When the input capture trigger, that is, TIFR1 ICF1 flag is set to interrupt the hair<br>Health<br>When the ICIE1 bit is "0", the TC1 input capture interrupt is disabled. |
| 4: 3 | - | Keep it. |
| 2 | OCIE1B | TC1 Output Compare B Match Interrupt Enable bit.<br>When the OCIE1B bit is "1" and the global interrupt is set, the TC1 output compares the B match<br>Disable enable. When a compare match occurs, that is, when the OCF1B bit in TIFR is set,<br>Interrupt generated.<br>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.<br>When the OCIE1A bit is "1" and the global interrupt is set, the TC1 output compares the A match<br>Disable enable. When a compare match occurs, that is, when the OCF1A bit in TIFR is set,<br>Interrupt generated.<br>When the OCIE1A bit is "0", the TC1 output compare A match interrupt is disabled. |
| 0 | TOIE1 | TC1 overflow interrupt enable bit.<br>When the TOIE1 bit is "1" and the global interrupt is set, the TC1 overflow interrupt is enabled. When TC1<br>When an overflow occurs, that is, when the TOV1 bit in TIFR is set, an interrupt is generated.<br>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: 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 | - | Keep it. |
| 5 | ICF1 | Enter the capture flag.<br>The ICF1 flag is set when an input capture event occurs. When ICR1 is used as a count<br>TOP value, and the count value reaches the TOP value, the ICF1 flag is set. If ICIE1 is<br>"1" and the global interrupt flag is set, an input capture interrupt is generated. Perform this break service |





| | | |
|---|---|---|
| | | ICF1 will be cleared automatically when the program is programmed, or it can be cleared by writing "1" to the ICF1 bit. |
| 4: 3 | - | Keep it. |
| 2 | OCF1B | Output Compare B match flag.<br>When TCNT1 is equal to OCR1B, the comparison unit gives the match signal and sets the compare<br>Logo OCF1B. If the output compare interrupt is enabled, OCIE1B is "1" and the global interrupt is enabled<br>Flag is set, an output compare interrupt is generated. OCF1B is executed when this interrupt service routine is executed<br>This bit can also be cleared by clearing it automatically or by writing "1" to the OCF1B bit. |
| 1 | OCF1A | Output Compare A match flag.<br>When TCNT1 is equal to OCR1A, the comparison unit gives the match signal and sets the comparison<br>Logo OCF1A. If the output compare interrupt is enabled, OCIE1A is "1" and the global interrupt is enabled<br>Flag is set, an output compare interrupt is generated. OCF1A is executed when this interrupt service routine is executed<br>This bit can also be cleared by clearing it automatically or by writing "1" to the OCF1A bit. |

| | | |
|---|---|---|
| 0 | TOV1 | Overflow flag.<br>When the counter overflows, set the overflow flag TOV1. If the overflow interrupt is enabled at this time<br>When TOIE1 is "1" and the global interrupt flag is set, an overflow interrupt is generated. Perform this<br>TOV1 will be cleared automatically when the service routine is serviced, or "1" for TOV1 bits can also be cleared.<br>Bit. |

### DTR1L -TC1 dead time register low byte

DTR1 -TC1 dead time register

Address: 0x8C Default: 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.<br>When the DTEN1 bit is high, OC1A and OC1B are complementary outputs, OC1A output<br>The dead time of the insertion is determined by the DTR1L count clock. |

### DTR1H -TC1 dead time register high byte

DTR1H -TC1 dead time register high byte

Address: 0x8D Default: 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.<br>When the DTEN1 bit is high, OC1A and OC1B are complementary outputs, OC1B output<br>The dead time of the insertion is determined by the DTR1H count clock. |





### TCKCSR -TC clock control status register

TCKCSR -TC clock control status register

Address: 0xEC Default: 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 | - | Keep it |
| 6 | F2XEN | RC 32M multiplier output enable control bit<br>When the F2XEN bit is set to "1", the multiplier output of the 32M RC oscillator is enabled,<br>Output 64M high-speed clock<br>When the F2XEN bit is set to "1", the multiplier output of the 32M RC oscillator is disabled,<br>Can not output 64M high speed clock |
| 5 | TC2XF1 | TC high speed clock mode flag bit 1<br>When the TC2XF1 bit is read as "1", it indicates that the timer counter 1 is operating on the high-speed clock<br>Mode, when "0", indicates that the timer counter 1 is operating in system clock mode |
| 4 | TC2XF0 | TC High Speed Clock Mode Flag Bit 0, Reference Timer Counter 0 Register Description |
| 3: 2 | - | Keep it |
| 1 | TC2XS1 | TC high speed clock mode selection control bit 1<br>When setting the TC2XS1 bit to "1", select the timer counter 1 to operate on the high-speed clock<br>mode |

When the TC2XS1 bit is set to "0", the timer counter 1 is selected to operate on the system clock mode

0 TC2XS0 TC High Speed Clock Mode Select Control Bit 0, Reference Timer Counter 0 Register Description





## TMR0 / 1/3 prescaler

- 3 10-bit prescaler
- TC0, TC1 and TC3 multiplexed prescaler CPS310 in multiplexed mode
- independent mode TC0 exclusive prescaler CPS310, TC1 exclusive prescaler CPS1, TC3 exclusive pre-divider Frequency controller CPS3
- Support software reset

### Overview

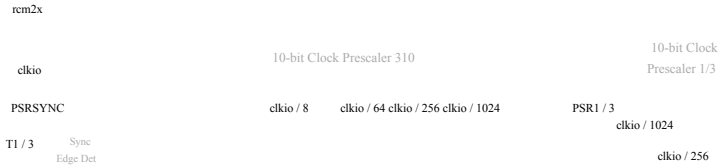
In multiplexed mode (PSS1 = 0 and PSS3 = 0), TC0, TC1 and TC3 share a 10-bit prescaler CPS310, But they have different frequency settings.

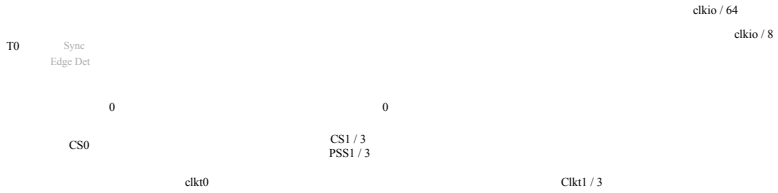
In single mode (PSS1 = 1 and PSS3 = 0), TC1 independently uses a 10-bit prescaler CPS1, TC0 and TC3 Share a 10-bit prescaler CPS310, but they have different frequency settings.

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

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

The following description is used for TC0, TC1 and TC3, where n represents 0, 1 or 3.






TC0 / TC1 / TC3 Prescaler structure

### 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 the high-speed clock rcm2x (2 times the internal 32M RC oscillator output clock). Prescaler Device can output four different clock frequencies, namely clkio / 8, clkio / 64, clkio / 256 and clkio / 1024.



### The divider is reset

#### Multiplexing mode

When setting PSS1 bit to "0" and PSS3 bit is "0", TC0, TC1 and TC3 share a prescaler CPS310.

The prescaler is run independently and its operation is independent of TC's clock selection logic, and it consists of TC0, TC1 and TC3. enjoy. The status of the prescaler has an effect on the application of the frequency divider due to the influence of clock selection control. When set When the timer is enabled and the output of the prescaler is selected as the count clock source (6> CSn [2: 0]> 1), the effect is generated. From the timer enable to the first count may take 1 to N + 1 system clock, where N is the prescaler factor (8, 64,256 or 1024).

It is possible to synchronize the timer and program operation by resetting the prescaler. But it must be noted that another timer is If you are using this prescaler, resetting the prescaler will affect all timers connected to it.

#### Single mode

When the PSS1 bit is set to "1", TC1 independently uses the prescaler CPS1, the prescaler reset is controlled by the PSR1 bit system. The respective reset works independently and does not affect other prescalers.

When the PSS3 bit is set to "1", the TC3 independently uses the prescaler CPS3. The prescaler reset is controlled by the PSR3 bit system. The respective reset works independently and does not affect other prescalers.

When the PSS1 bit is set to "1" and the PSS3 bit is "1", TC0 independently uses the prescaler CPS310, the prescaler Bit by the PSRSYNC bit to control, TC1 independent use of prescaler CPS1, TC3 independent use of prescaler CPS3, the Since the reset function alone does not affect other prescalers.

### External clock source

The external clock source provided by the T0 / T1 / T3 pin can be used as the count clock source. The signal on the T0 / T1 / T3 pin is the same Step logic and edge detector as the clock source for the counter. Each rising edge (CSn [2: 0] = 7) or falling edge (CSn [2: 0] = 6) will produce a count pulse. The external clock source is not fed into the prescaler.

Due to the presence of pin synchronization and edge detection circuitry, the change in T0 / T1 / T3 level requires a delay of 2.5 to 3.5 The system clock can cause the counter to be updated.

Disable or enable clock input must be held at T0 / T1 / T3 at least one system clock cycle is required before There is a possibility that an error count clock pulse is generated.

To ensure proper sampling, the external clock pulse width must be greater than one system clock cycle, with a duty cycle of 50%
The external clock frequency must be less than half the system clock frequency. Due to the error of the oscillator itself brings the system clock frequency
And the difference in duty cycle, it is recommended that the maximum frequency of the external clock should not be greater than $f_{sys}$ / 2.5.





## Register definition

### GTCCR - General Timing Counter Control Register

GTCCR - General Timing Counter Control Register

Address: 0x43 Default: 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 bits.<br>When setting the TSM bit to "1", it is the timer counter synchronization mode. In synchronous mode, write<br>The value of the PSRASY bit and the PSRSYNC bit is held so that the corresponding prescaler is always held<br>Reset. This ensures that the corresponding timer counter is aborted and configured to the same value.<br>When the TSM bit is set to "0", the values of the PSRASY bit and the PSRSYNC bit are cleared by hardware<br>Zero, and the timer counter starts working at the same time. |
| 6: 2 | - | Keep it. |
| 1 | PSRASY | See Timer TC2 Register Description. |
| 0 | PSRSYNC | prescaler CPS310 reset control bit.<br>When the PSRSYNC bit is set to "1", the prescaler CPS310 will be reset. When TSM<br>Bit is not set, the hardware clears the PSRSYNC bit after reset.<br>When the PSRSYNC bit is set to "0", the setting is invalid.<br>In multiplexed mode, the TC0 / TC1 / TC3 shared prescaler, the reset will affect these three settings<br>Timers.<br>In standalone mode, reset will only affect TC0.<br>The value of reading this bit will always be "0". |

### PSSR - Prescaler Select Register

PSSR - Prescaler Select Register

Address: 0xE2 Default: 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 bit.<br>When setting the PSS1 bit to "1", TC1 uses the prescaler CPS1 alone.<br>When the PSS1 bit is set to "0", the prescaler is multiplexed. TC0 and TC1 are shared<br>Prescaler CPS310. Prescaler CPS1 is invalid and will be reset. If PSS3<br>Bits are simultaneously "0", TC3 and TC0, TC1 share the prescaler CPS310. Prescaler<br>CPS1 and CPS3 are invalid and will be reset. |
| 6 | PSS3 | Prescaler selection control bit.<br>When the PSS3 bit is set to "1", the TC3 uses the prescaler CPS3 alone.<br>When the PSS3 bit is set to "0", the prescaler is multiplexed. TC0 and TC3 are shared |

**Page 136**



| | | |
|---|---|---|
| | | Prescaler CPS310. Prescaler CPS3 is invalid and will be reset. If PSS1 Bit is "0" at the same time, TC1 and TC0, TC3 share the prescaler CPS310. Prescaler CPS1 and CPS3 are invalid and will be reset. |
| 5: 2 | - | Keep it. |
| 1 | PSR3 | Prescaler CPS3 reset control bit. The PSR3 bit is valid only in TC3 mode. When the PSR3 bit is set to "1", the prescaler CPS3 will be reset. Reset after hard The part will clear the PSR3 bit. When the PSR3 bit is set to "0", the setting is invalid. The value of reading this bit will always be "0". |
| 0 | PSR1 | Prescaler CPS1 reset control bit. The PSR1 bit is valid only in TC1 mode. When the PSR1 bit is set to "1", the prescaler CPS1 will be reset. Reset after hard The part will clear the PSR1 bit. When setting the PSR1 bit to "0", the setting is invalid. The value of reading this bit will always be "0". |



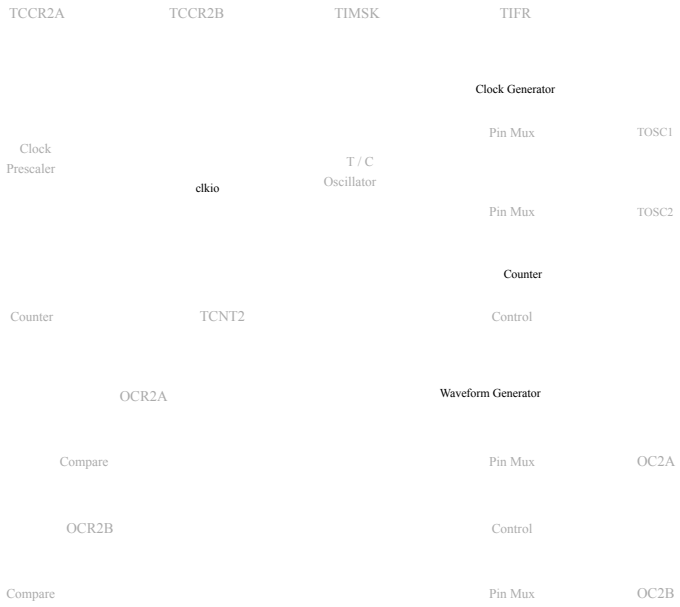
**Page 137**



## Timer / Event Counter 2 (TMR2)

- 8-bit counter
- Two separate comparison units
- When the compare match occurs, the counter is automatically cleared and loaded automatically
- Phase-corrected PWM output without interference pulse
- Frequency generator
- External event counter
- 10-bit clock prescaler
- overflow and compare match interrupt
- Allows the use of external 32.768KHz RTC crystal count

## Overview




TC2 structure diagram

TC2 is a general-purpose 8-bit timer counter module that supports PWM output and can produce waveforms precisely. TC2 package
It contains 1 8-bit counter, waveform generation mode control unit and 2 output comparison unit. Waveform generation mode control unit
Controls the operation mode of the counter and the generation of the comparison output waveform. Depending on the mode of operation, the counter is for each one
Count the clock Clkt2 to clear, add one or minus one operation. Clkt2 can be generated by an internal clock source or an external clock source.
When using an external 32.768KHz crystal count, TC2 can be used as an RTC counter. When the counter counts the value TCNT2





Reaches the maximum value (equal to the maximum value 0xFF or the output compare register OCR2A, defined as TOP, defining the maximum value for MAX
To distinguish between), the counter will be cleared or decremented. When the counter count value TCNT2 reaches the minimum value (etc.)
At 0x00, defined as BOTTOM), the counter will be added to an operation. When the counter count value TCNT2 arrives
OCR2A / OCR2B, also known as when a compare match occurs, will clear or set the output compare signal OC2A / OCR2B,
Generates PWM waveforms.

## Operating mode

Timing counter 2 has four different modes of operation, including normal mode (Normal), compare match clear (CTC)
Mode, fast pulse width modulation (FPWM) mode and phase correction pulse width modulation (PCPWM) mode,
Generates the mode control bit WGM2 [2: 0] to select. The four modes are described below in detail. Because there are two independent output ratios
The units are represented by "A" and "B", respectively, and the lower output "x" is used to represent the two output compare cell channels.

### Normal mode

The normal mode is the simplest mode of operation for the timer counter. The waveform generation mode control bit, WGM2 [2: 0] = 0,
The maximum value of the TOP is MAX (0xFF). In this mode, the count mode is incremented by one for each count clock
After the counter reaches the TOP overflow, it returns to BOTTOM to restart. In the same value as the count value TCNT2 becomes zero
Set the timer counter overflow flag TOV2 in the clock. The TOV2 flag in this mode is like the 9th count bit, just
Will only be set to not be cleared. The overflow interrupt service routine automatically clears the TOV2 flag, which the software can use to improve it
The resolution of the counter. There is no special case in normal mode to consider, you can always write a new count value.
The waveform of the output compare signal OC2x can be obtained by setting the data direction register of the OC2x pin to output. When COM2x = 1
, The OC2x signal is flipped when a compare match occurs. The frequency of the waveform in this case can be calculated using the following formula:

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

Where N represents the prescaler factor (1, 8, 64, 256, or 1024).
The output compare unit can be used to generate interrupts, but interrupts are not recommended in normal mode, which takes too much CPU
time.

### CTC mode

When the WGM2 [2: 0] = 2 is set, the timer counter 2 enters the CTC mode, and the maximum value of the count is OCR2A. in
In this mode, the count mode is incremented by one for each count clock. When the value of the counter TCNT2 equals TOP
The counter is cleared. OCR2A defines the maximum value of the count, that is, the resolution of the counter. This mode allows the user to be tolerant
Easy control matches the frequency of the output and also simplifies the operation of the external event count.
When the counter reaches the maximum value of the count, the output compare match flag OCF2 is set and the corresponding interrupt enable is set
Will be interrupted. The OCR2A register can be updated in the interrupt service routine to count the maximum value. In this mode
OCR2A does not use double buffering, the counter is updated with no prescaler or very low prescaler operation
Be careful when you are near the minimum. If the value written to OCR2A is less than the current TCNT2 value, the counter will lose one
Times match. Before the next match match occurs, the counter has to count to TOP before starting from BOTTOM
Start counting to OCR2A value. As with normal mode, the count value returns to the BOTTOM count clock to set the TOV2 flag.
The waveform of the output compare signal OC2x can be obtained by setting the data direction register of the OC2x pin to output. When COM2x = 1
, The OC2x signal is flipped when a compare match occurs. The frequency of the waveform in this case can be calculated using the following formula:

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

Where N represents the prescaler factor (1, 8, 64, 256, or 1024). As can be seen from the formula, when setting OCR2x
For 0x0 and no prescaler, the output waveform with maximum frequency $f_{sys} / 2$ can be obtained .





### Fast PWM mode

When setting WGM2 [2: 0] = 3 or 7, the timer counter 2 enters the fast PWM mode and can be used to generate high frequency
PWM waveform, the maximum value TOP is MAX (0xFF) or OCR2A, respectively. Fast PWM mode and other PWM modes
The difference is that it is a one-way operation. The counter is incremented from the minimum value 0x00 to TOP and then back to BOTTOM to count.
When the count value TCNT2 reaches OCR2x or BOTTOM, the output compare signal OC2x is set or cleared, depending on the ratio
The setting of the output mode COM2x is described in the register description. Due to the use of one-way operation, fast PWM mode operation
The frequency is twice the phase corrected PWM mode using bidirectional operation. High-frequency characteristics make the fast PWM mode suitable for work
Rate adjustment, rectification, and DAC applications. High-frequency signals can reduce the size of external components (inductive capacitors, etc.) and drop
Low system cost.
When the count value reaches the maximum value, the timer counter overflow flag TOV2 will be set and the value of the compare 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 compare signal OC2x can be obtained by setting the data direction register of the OC2x pin to output. The frequency of the waveform
The rate can be calculated using the following formula:

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

Where N represents the prescaler factor (1, 8, 64, 256, or 1024).
When a compare match occurs between TCNT2 and OCR2x, the waveform generator sets (clears) the OC2x signal. When TCNT2 is cleared
At zero, the waveform generator clears (sets) the OC2x signal to generate a PWM wave. Whereby the extreme value of OCR2x will be
Generates special PWM waveforms. When OCR2x is set to 0x00, the output PWM is in every (1 + TOP) count clock
There is a narrow spike. When OCR2x is set to the maximum value, the output waveform is a continuous high or low level.

### Phase correction PWM mode

When setting WGM2 [2: 0] = 1 or 5, the timer counter 2 enters the phase correction PWM mode, the maximum value of the count TOP is MAX (0xFF) or OCR2A, respectively. The counter is bidirectional, incremented by BOTTOM to TOP, and then again Decrements to BOTTOM, and repeats this operation. When the count reaches TOP and BOTTOM, the counting direction is changed and the count value is TOP Or BOTTOM only stay on a count clock. In the process of increment or decrement, the count value TCNT2 matches OCR2x , The output compare signal OC2x will be cleared or set, depending on the setting of the compare output mode COM2x. With one-way operation The maximum frequency available for bi-directional operation is smaller, but its excellent symmetry is more suitable for motor control.

In phase correction PWM mode, the TOV2 flag is set when the count reaches BOTTOM. When the count reaches TOP, compare The value of the buffer is updated to the comparison value. If the interrupt is enabled, the compare buffer OCR2x can be updated in the interrupt service routine Deposit.

The waveform of the output compare signal OC2x can be obtained by setting the data direction register of the OC2x pin to output. The frequency of the waveform The rate can be calculated using the following formula:

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

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

During the count up, when the TCNT2 matches OCR2x, the waveform generator clears (sets) the OC2x signal. in In the process of decrementing the count, when the TCNT2 matches OCR2x, the waveform generator sets (clears) the OC2x signal. thus The extreme value of OCR2x will produce a special PWM wave. When OCR2x is set to the maximum or minimum value, the OC2x signal will be output Keep it low or high.

In order to ensure that the output PWM wave symmetry on both sides of the minimum value, in the absence of a comparison match, there are two cases Flip OC2x signal. The first case is when the value of OCR2x changes from the maximum value 0xFF to other data. When OCR2x Is the maximum value, the count value reaches the maximum, OC2x output and the previous descending count when the results of the same match, that is, Hold OC2x unchanged. The value of the new OCR2x (non-0xFF) is updated and the value of OC2x is always maintained.





When the ascending order counts, a compare match occurs. At this point the OC2x signal is not centered at the minimum, so it needs to be When TCNT2 reaches the maximum value, it flips the OC2x signal, which means that the first match of the OC2x signal is reversed when no compare match occurs condition. The second case is that when TCNT2 starts counting from a value higher than OCR2x, it will lose a compare match, Thus causing the occurrence of asymmetric situations. Also need to flip the OC2x signal to achieve the minimum on both sides of the symmetry.

### TC2 asynchronous operation mode

When the AS2 bit in the ASSR register is "1", TC2 operates in asynchronous mode and the clock source of the counter comes from Part of the timer counter of the oscillator. Asynchronous mode TC2 operation to consider the following points.

- Conversions between synchronous and asynchronous modes are likely to result in TCNT2, OCR2A, OCR2B, TCCR2A and TCCR2B counts According to the damage. The safe operation steps are as follows:
  1. Clear the OCIE2A, TOIE2, and OCIE2B register bits to turn off TC2 interrupts;
  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, wait for TCN2UB, OCR2AUB, TCR2AUB, OCR2BUB, and TCR2BUB bits Clear
  5. Clear TC2 interrupt flag bit;
  6. Enable interrupts that need to be used.
- The oscillator is best to use 32.768KHz watch crystal. The system clock frequency must be four times higher than the crystal frequency.
- CPU write TCNT2, OCR2A, TCCR2A, OCR2B and TCCR2B, the hardware will first put the data into the temporary register, two The rising edge of the TOSC1 clock is latched into the corresponding register. The data is latched from the scratchpad into the destination register A new data write operation can not be performed before. Each register has its own independent scratchpad, so write TCNT2 and Does 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 for MCU sleep mode, it can not be entered before each register update ends Sleep mode, otherwise the MCU may enter the sleep mode before the TC2 setting takes effect, so the TC2 can not wake up the system System.
- If TC2 is used as the wake-up condition for MCU sleep mode, care must be taken to re-enter the sleep mode. in The interrupt logic requires a TOSC1 clock cycle to be reset if the time from wake-up to re-entry dormancy is less than one TOSC1 clock cycle, the interrupt will no longer occur, the device can not wake up. It is recommended to use the following methods:
  1. Write the appropriate data for each register;

2. Wait for the corresponding update busy flag of ASSR to clear;

3. Go to sleep mode.

- If the asynchronous operating mode is selected, the TC2 oscillator will always operate unless it enters the power down mode. Users must be aware that, The stabilization time of this oscillator may be as long as 1 second, so it is recommended that the user wait at least after enabling the TC2 oscillator 1 second and then use TC2 asynchronous mode of operation.
- Asynchronous operation mode Sleep mode wake-up process: After the interrupt condition is met, the next timer clock starts to call Wake up the process. That is, the counter accumulates at least one clock before the processor can read the value of the counter. After the wake-up, the MCU executes the interrupt service routine and then executes the program after the SLEEP statement.
- Reading the value of TCNT2 within a short time after waking up from Sleep mode may return incorrect data. Because TCNT2 is Driven by an asynchronous TOSC1 clock, while reading TCNT2 must be synchronized via an internal system clock register To complete, the synchronization occurs on the rising edge of each TOSC1. After wake-up from Sleep mode, the system clock is reactivated and read The TCNT2 value is entered before the sleep mode until the next rising edge of the TOSC1 is updated. The phase of the TOSC1 is completely unpredictable when awakened from sleep mode, depending on the wake-up time. Therefore, read TCNT2 The recommended sequence of values is:

1. Write an arbitrary value to OCR2A or TCCR2A;





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

3. Read TCNT2.

- In asynchronous mode, the synchronization of the interrupt flag requires 3 system clock cycles plus 1 timer period. In the MCU can be The counter accumulates at least one clock before reading the counter value that causes the interrupt flag to be set. Output the comparison letter The number of changes is synchronized with the timer clock, not the system clock.

## TC2 prescaler

The input clock of the TC2 prescaler is called clkt2s and is selected by the AS2 bit in the ASSR register. Clock clkio or external TOSC1 clock source, the default is connected with the system clock clkio. If AS2 is set, TC2 will be set by TOSC1 asynchronous drive. When the TOSC1 pin and TOSC2 pin are connected to a 32.768KHz watch crystal, TC2 can be used as RTC counter. It is not recommended to apply an external clock signal directly to the TOSC1 pin.

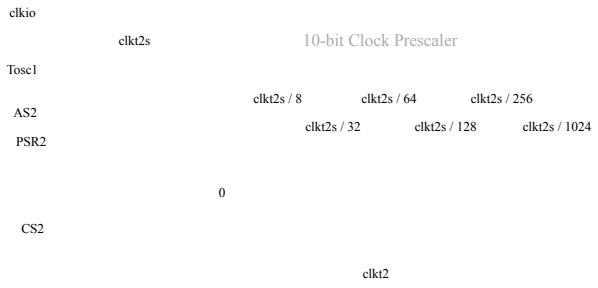



Figure 5 TC2 prescaler structure

The picture 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. You can also select clkt2s and 0 (stop counting). Set SFIOR The PSR2 bit of the register will reset the prescaler, allowing the user to start working from a predictable prescaler.

## Register definition

TC2 register list

| register | address | Defaults | description |
|---|---|---|---|
| TCCR2A | 0xB0 | 0x00 | TC2 control register A |
| TCCR2B | 0xB1 | 0x00 | TC2 control register B |
| TCNT2 | 0xB2 | 0x00 | TC2 count 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 | The timer counter interrupts the mask register |
| TIFR2 | 0x37 | 0x00 | Timer counter interrupt flag register |





TCCR2A-TC2 Control Register A

TCCR2 A -TC2 Control Register A

Address: 0xB0 Default: 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.<br>COM2A1 and COM2A0 together constitute the output compare mode control COM2A [1: 0], control<br>OC2A output waveform. If COM2A 1 bit or 2 bits are set, the output ratio<br>The waveform occupies the OC2A pin, but the pin's data direction register must be set<br>High to output this waveform. In different operating modes, COM2A compares the output waveform<br>The control is also different, see the specific comparison of the output mode control table description. |
| 6 | COM2A0 | TC2 compare match output A mode control low.<br>COM2A0 and COM2A1 together constitute the output compare mode control COM2A [1: 0], control<br>OC2A output waveform. If COM2A 1 bit or 2 bits are set, the output ratio<br>The waveform occupies the OC2A pin, but the pin's data direction register must be set<br>High to output this waveform. In different operating modes, COM2A compares the output waveform<br>The control is also different, see the specific comparison of the output mode control table description. |
| 5 | COM2B1 | TC2 compare match output B mode control high.<br>COM2B1 and COM2B0 together form the output compare mode control COM2B [1: 0], control<br>OC2B output waveform. If COMB bit 1 or 2 is set, the output ratio<br>The waveform occupies the OC2B pin, but the data direction register for this pin must be set<br>High to output this waveform. In different operating modes, COM2B compares the waveform to the output<br>The control is also different, see the specific comparison of the output mode control table description. |
| 4 | COM2B0 | TC2 compare match output B mode control low.<br>COM2B0 and COM2B1 together form the output compare mode control COM2B [1: 0], control<br>OC2B output waveform. If COMB bit 1 or 2 is set, the output ratio<br>The waveform occupies the OC2B pin, but the data direction register for this pin must be set<br>High to output this waveform. In different operating modes, COM2B compares the waveform to the output<br>The control is also different, see the specific comparison of the output mode control table description. |
| 3: 2 | - | Keep it. |
| 1 | WGM21 | TC2 waveform generation mode control high.<br>WGM20 and WGM21, WGM22 together constitute the waveform generation mode control<br>WGM2 [2: 0], control counter count mode and waveform generation mode, see waveform<br>Generate a pattern table description. |
| 0 | WGM20 | TC2 waveform generation mode control low.<br>WGM21 and WGM20, WGM22 together constitute the waveform generation mode control<br>WGM2 [2: 0], control counter count mode and waveform generation mode, see waveform<br>Generate a pattern table description. |

**Page 143**



TCCR2B -TC2 Control Register B

TCCR2B -TC2 Control Register B

Address: 0xB1 Default: 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 Force Output Compare A control bit.<br>When operating in non-PWM mode, the "1" can be written to the forced output compare bit FOC2A<br>Way to produce a comparison match. Forcing the match does not set the OCF2A flag, nor will it be<br>Overload or clear the timer, but the output pin OC2A will be set according to COM2A<br>Should be updated, just like the real match has occurred.<br>The return value of reading FOC2A is always zero. |
| 6 | FOC2B | TC2 Force Output Compare B control bits.<br>When operating in non-PWM mode, the "1" can be written to the forced output compare bit FOC2B<br>Way to produce a comparison match. Forcing the match does not set the OCF2B flag, nor will it be<br>Overload or clear the timer, but the output pin OC2B will be set according to COM2B settings<br>Should be updated, just like the real match has occurred.<br>The return value of the read FOC2B is always zero. |
| 5: 4 | - | Keep it. |
| 3 | WGM22 | TC2 waveform generation mode control high.<br>WGM22 and WGM20, WGM21 together form the waveform generation mode control WGM2 [2: 0],<br>Control the counter count mode and waveform generation mode, see the waveform generation mode table<br>description. |
| 2 | CS22 | TC2 clock selection control high. |
| 1 | CS21 | Used to select the clock source for Timer 2.<br>TC2 Clock selection control in position. |
| 0 | CS20 | Used to select the clock source for Timer 2.<br>TC2 clock selection control low. |

Used to select the clock source for Timer 2.

| CS2 [2: 0] | description |
|---|---|
| 0 | No clock source, stop counting |
| 1 | clk $_{t2s}$ |
| 2 | clk $_{t2s}$ / 8, from prescaler |
| 3 | clk $_{t2s}$ / 32, from prescaler |
| 4 | clk $_{t2s}$ / 64, from prescaler |
| 5 | clk $_{t2s}$ / 128, from prescaler |
| 6 | clk $_{t2s}$ / 256, from prescaler |
| 7 | clk $_{t2s}$ / 1024, from prescaler |

The following table shows the non-PWM mode (ie, normal mode and CTC mode), compare the output mode to the output compare waveform control.



**Page 144**



Table 1 OC2x Compare Output Mode Control in Non-PWM Mode

| COM2x [1: 0] | description |
|---|---|

| | |
|---|---|
| 0 | OC2x disconnect, general IO port operation |
| 1 | Match the OC2x signal when comparing the match |
| 2 | The OC2x signal is cleared when compare match |
| 3 | The OC2x signal is set when the compare match |

The following table shows the control of the output compare waveform for the compare output mode in fast PWM mode.

Table 2 OC2x Compare Output Mode Control in Fast PWM Mode

| COM2x [1: 0] | description |
|---|---|
| 0 | OC2x disconnect, general IO port operation |
| 1 | Keep it |
| 2 | The OC2x signal is cleared when the compare match is set and the OC2x signal is set when the maximum match is made |
| 3 | The OC2x signal is set when the compare match is cleared and the OC2x signal is cleared when the maximum match is made |

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

Table 3 Phase Correction PWM Mode OC2x Compare Output Mode Control

| COM2x [1: 0] | description |
|---|---|
| 0 | OC2x disconnect, general IO port operation |
| 1 | Keep it |
| 2 | The OC2x signal is cleared when the compare match is cleared in the ascending count, and the match is compared in descending order<br>Set the OC2x signal |
| 3 | The OC2x signal is set when the compare match is set in the ascending count, and the match is compared in descending order<br>The OC2x signal is cleared |

The following table shows the waveform generation mode control.

Table 4 Waveform Generation Mode Control

| WGM2 [2: 0] | Operating mode | TOP value | Update OCR2x at all times | Set TOV2 at all times |
|---|---|---|---|---|
| 0 | Normal | 0xFF | immediately | MAX |
| 1 | PCPWM | 0xFF | TOP | BOTTOM |
| 2 | CTC | OCR2A | immediately | MAX |
| 3 | FPWM | 0xFF | TOP | MAX |
| 4 | Keep it | - | - | - |
| 5 | PCPWM | OCR2A | TOP | BOTTOM |
| 6 | Keep it | - | - | - |
| 7 | FPWM | OCR2A | TOP | TOP |





TCNT2 -TC2 count value register

TCNT2 -TC2 count value register

Address: 0xB2 Default: 0x00

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCNT27 | TCNT25 | TCNT25 | TCNT23 | TCNT23 | TCNT21 | TCNT21 | TCNT20 |
| R / W | R / W | R / W | R / W | R / W | R / W | R / W | R / W | R / W |

| Bit | Name | description |
|---|---|---|
| | | TC2 count register.<br>Through the TCNT2 register can be directly on the counter 8 for the count value to read and write access<br>ask.<br>The CPU writes to the TCNT2 register will block the next timer clock cycle |

| | |
|---|---|
| 7: 0 TCNT2 | The match occurs even if the timer has stopped. This allows the initialization of TCNT2 to be registered<br>The value of the device coincides with the value of OCR2 without causing an interrupt.<br>If the value written to TCNT2 equals or bypasses the OCR2 value, the compare match is lost,<br>Resulting in incorrect waveform results.<br>The timer stops counting when no clock source is selected, but the CPU can still access TCNT2. CPU<br>The write counter has a higher priority than the clear or add / subtract operation. |

### OCR2A - TC2 Output Compare Register A

OCR2A - TC2 Output Compare Register A

Address: 0xB3 Default: 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.<br>The OCR2A contains an 8-bit data that is continuously compared with the counter value TCNT2.<br>The compare match can be used to generate an output compare interrupt, or to generate on the OC2A pin<br>Waveform.<br>When using PWM mode, the OCR2A register uses a double buffered register. And ordinary work<br>Mode and match clear mode, double buffering is disabled. Double buffering can be updated<br>The OCR2A register is synchronized with the count maximum or minimum time to prevent generation of<br>Symmetric PWM pulses eliminate the interference pulses.<br>When using the double buffering function, the CPU accesses the OCR2A buffer register, which inhibits double buffering<br>When the CPU can access the OCR2A itself. |

### OCR2B - TC2 Output Compare Register B

OCR2B - TC2 Output Compare Register B

Address: 0xB4 Default: 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 |



**Page 146**



| Bit | Name | description |
|---|---|---|
| 7: 0 | OCR2B | TC2 Output Compare B register.<br>The OCR2B contains an 8-bit data that is uninterrupted compared to the counter value TCNT2<br>More The compare match can be used to generate an output compare interrupt, or to be used on the OC2B pin<br>Produce a waveform.<br>When using PWM mode, the OCR2B register uses a double buffered register. While the general work<br>In mode and match clear mode, the double buffering function is disabled. Double buffering can be more<br>The new OCR2B register is synchronized with the count maximum or minimum time to prevent production<br>The asymmetric PWM pulse eliminates the interference pulse. When using the double buffering function, the CPU<br>Access is the OCR2B buffer register, disable the dual buffering function when the CPU access is<br>OCR2B itself. |

### TIMSK2 - TC2 interrupt mask register

TIMSK2 - TC2 interrupt mask register

Address: 0x70 Default: 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 | | Keep it. |
| 2 | OCIE2B | TC2 Output Compare B Match Interrupt Enable bit.<br>When the OCIE2B bit is set to "1" and the global interrupt is set, the TC2 output compares the B match interrupt enable.<br>When a compare match occurs, that is, when the OCF2B bit in TIFR2 is set, an interrupt is generated.<br>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.<br>When the OCIE2A bit is "1" and the global interrupt is set, the TC2 output compares the A match interrupt enable.<br>When a compare match occurs, that is, when the OCF2A bit in TIFR2 is set, an interrupt is generated.<br>When the OCIE2A bit is "0", the TC2 output compare A match interrupt is disabled. |
| 0 | TOIE2 | TC2 overflow interrupt enable bit.<br>When the TOIE2 bit is "1" and the global interrupt is set, the TC2 overflow interrupt is enabled. When TC2 overflows<br>When the TOV2 bit in TIFR2 is set, the interrupt is generated. When the TOIE2 bit is "0"<br>TC2 overflow interrupt is disabled. |

TIFR2 - TC2 interrupt flag register

TIFR2 - TC2 interrupt flag register

Address: 0x37 Default: 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 | - | Keep it. |
| 2 | OCF2B | TC2 Output Compare B match flag. |



**Page 147**



| | | |
|---|---|---|
| 1 | OCF2A | When TCNT2 is equal to OCR2B, the compare unit gives the match signal and sets the compare flag<br>OCF2B. If the output compare B interrupt is enabled, OCIE2B is set to "1" and the global interrupt flag is set<br>Bit, an output compare B interrupt is generated. OCF2B will automatically be executed when this interrupt service routine is executed<br>Clear this bit, or write "1" to the OCF2B bit.<br>TC2 Output Compare A match flag.<br>When TCNT2 is equal to OCR2A, the comparison unit gives the match signal and sets the compare flag<br>OCF2A. If the output compare A interrupt is enabled, OCIE2A is set to "1" and the global interrupt flag is set<br>Bit, an output compare A interrupt is generated. OCF2A will automatically be executed when this interrupt service routine is executed<br>Clear this bit, or write "1" to the OCF2A bit. |
| 0 | TOV2 | TC2 overflow flag.<br>When the counter overflows, set the overflow flag TOV2. If the overflow interrupt is enabled at this time, TOIE2<br>Is set to "1" and the global interrupt flag is set, an overflow interrupt is generated. Execute this interrupt service routine<br>TOV2 will be cleared automatically, or "1" for TOV2 bits can also be cleared. |

ASSR - Asynchronous interface status register

ASSR - TC2 Asynchronous Interface Status Register

Address: 0xB6 Default: 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 select control bit.<br>When the INTCK bit is set to 1, the internal RC32K is selected as the asynchronous clock source.<br>When the INTCK bit is set to 0, the external crystal clock is selected as the asynchronous clock source. |
| 6 | - | Keep it. |
| | | Timer 2 asynchronous mode selection control bit.<br>When setting the AS2 bit to 1, Timer 2 operates as an asynchronous mode with clock source from INTCK<br>Bit to choose. |

| | | |
|---|---|---|
| 5 | AS2 | When setting the AS2 bit to 0, Timer 2 operates as a synchronous mode with a clock source of Clk $_{io}$ .<br>When the value of AS2 changes, TCNT2, OCR2A, OCR2B, TCCR2A and TCCR2B send<br>The value of the register may be incorrect and needs to be reconfigured. |
| 4 | TCN2UB | TCNT2 register update flag.<br>When Timer 2 is operating in asynchronous mode and TCNT2 is written, the TCN2UB bit<br>Will be set. When the value of TCNT2 is updated, the hardware clears the TCN2UB bit. only<br>TCNT2 can be updated when TCN2UB bit is 0. |
| 3 | OCR2AUB | OCR2A register update flag.<br>When timer 2 is operating in asynchronous mode and OCR2A is written, OCR2AUB<br>Bit will be set. When the value of OCR2A is updated, the hardware clears the OCR2AUB bit. only<br>If OCR2AUB bit is 0, OCR2A can be updated. |
| 2 | OCR2BUB | OCR2B register update flag.<br>When the timer 2 is operated in asynchronous mode and the OCR2B is written, OCR2BUB<br>Bit will be set. When the value of OCR2B is updated, the hardware clears the OCR2BUB bit. only<br>If OCR2BUB bit is 0, OCR2B can be updated. |



**Page 148**



| | | |
|---|---|---|
| 1 | TCR2AUB | TCCR2A register update flag.<br>When Timer 2 is operating in asynchronous mode and TCCR2A is written, TCR2AUB<br>Bit will be set. When the value of TCCR2A is updated, the hardware clears the TCR2AUB bit.<br>TCCR2A can only be updated if TCR2AUB bit is 0. |
| 0 | TCR2BUB | TCCR2B register update flag.<br>When Timer 2 is operating in asynchronous mode and TCCR2B is written, TCR2BUB<br>Bit will be set. When the value of TCCR2B is updated, the hardware clears the TCR2BUB bit.<br>The TCCR2B can only be updated if the TCR2BUB bit is 0. |

## Timer / Event Counter 3 (TMR3)

True 16-bit design allows 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 when the match matches and is automatically loaded
- Phase-corrected PWM without interference pulse
- Variable PWM cycle
- Frequency generator
- External event counter
- 5 independent interrupt sources
- with dead time control
- 6 selectable trigger sources automatically turn off the PWM output

### Overview

TC3 is a universal 16-bit timer counter module that supports PWM output and can accurately generate waveforms. TC3
Includes a 16-bit counter, a waveform generation mode control unit, two independent output compare units and one input capture
Catch the unit. The waveform generation mode control unit controls the operation mode of the counter and the generation of the comparison output waveform. according to
Different operating modes, the counter for each count clock Clkt3 is cleared, plus one or minus one operation. Clkt3 can be
To be generated by an internal clock source or an external clock source. When the counter count value TCNT3 reaches the maximum value (equal to the maximum value)
0xFFFF or fixed value or output compare register OCR3A or input capture register ICR3, defined as TOP, defined pole
Large value is MAX to distinguish), the counter will be cleared or decremented. When the counter counts the value TCNT3 to
When the minimum value (equal to 0x0000, defined as BOTTOM), the counter will be incremented. When the counter is counted
When the value TCNT3 reaches OCR3A or OCR3B or OCR3C, it is also called when the compare match is cleared or cleared
Bit output compare signal OC3A or OC3B or OC3C, to generate PWM waveforms. When the input capture function is on,
The ICR3 register will record the count value of the capture signal during the trigger period.

**Page 150**




Figure 6 TC3 structure diagram

## Operating mode

Timing counter 1 has six different modes of operation, including normal mode (Normal), compare match clear (CTC) Mode, fast pulse width modulation (FPWM) mode, phase corrected pulse width modulation (PCPWM) mode, phase frequency Rate correction pulse width modulation (PFCPWM) mode, and input capture (ICP) mode. The waveform is generated by the mode control bit WGM3 [3: 0] to choose. The six modes are described in detail below. Since there are three separate output comparison units, respectively "A", "B" and "C", with the lowercase "x" to represent the two output compare cell channels.



**Page 151**



## Normal mode

The normal mode is the simplest mode of operation for the timer counter, where the waveform generation mode control bit WGM3 [3: 0] = 0, count
The maximum value TOP is MAX (0xFFFF). In this mode, the count mode is incremented by one for each count clock
After the counter reaches the TOP overflow, it returns to BOTTOM to restart. In the same value as the count value TCNT3 becomes zero
Set the timer counter overflow flag TOV3 in the clock. The TOV3 flag in this mode is like the 17th bit, just
Will only be set to not be cleared. The overflow interrupt service routine automatically clears the TOV3 flag, which the software can use to improve it
The resolution of the counter. There is no special case in normal mode to consider, you can always write a new count value.

The waveform of the output compare signal OC3x can be obtained by setting the data direction register of the OC3x pin to output. when
When COM3x = 1, the OC3x signal is flipped when a compare match occurs. In this case, the frequency of the waveform can be expressed by the following equation
To calculate:

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

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

The output compare unit can be used to generate interrupts, but interrupts are not recommended in normal mode, which takes 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
The value TOP is OCR3A, and when WGM3 [3] = 1, the maximum value TOP is ICR3. The following to WGM3 [3: 0] = 4 as an example
Description CTC mode In this mode, the count mode is incremented by one for each count clock, when the value of the counter TCNT3
The counter is cleared when the value is equal to TOP. This mode allows the user to easily control the frequency of the compare match output and also simplify it
The operation of the external event count.

When the counter reaches TOP = OCR3A, the output compare match flag OCF3A is set when the counter reaches TOP = ICR3
, The output compare match flag ICF3 is set and an interrupt will be generated when the corresponding interrupt enable is set. In the interruption of service
The OCR3A register can be updated in sequence. In this mode the OCR3A does not use double buffering in the counter with no prescaler
Or very low prescaler operation to keep the maximum value close to the minimum when careful. If written to OCR3A
When the value is less than the current TCNT3 value, the counter will lose a compare match. Before the next match occurs,
The counter has to count to MAX, and then count from BOTTOM to OCR3A. As with normal mode, count the value
The TOV3 flag is set in the count clock returned to 0x0.

The waveform of the output compare signal OC3x can be obtained by setting the data direction register of the OC3x pin to output. The waveform of the
The frequency can be calculated using the following formula:

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

Where N represents the prescaler factor (1, 8, 64, 256, or 1024).
As can be seen from the formula, when the set OCR3A is 0x0 and no prescaler, you can get the maximum frequency $f_{sys}$ / 2 output
Waveform.

When WGM3 [3: 0] = 12 and WGM3 [3: 0] = 4 similar, but with the OCR3A related to ICR3 can be.





### Fast PWM mode

When the WGM3 [3: 0] = 5, 6, 7, 14 or 15 is set, the timer counter 1 enters the fast PWM mode and counts the maximum
TOP is 0xFF, 0x1FF, 0x3FF, ICR3 or OCR3A, can be used to generate high-frequency PWM waveform. Fast PWM
The mode differs from other PWM modes in that it is a one-way operation. The counter is returned from the BOTTOM to the TOP
BOTTOM recounting. When the count value TCNT3 reaches TOP or BOTTOM, the output compare signal OC3x is set
Or clear, depending on the compare output mode COM3 settings, see the register description for details. Due to the use of one-way operation, fast
The operating frequency of the PWM mode is twice the phase-corrected PWM mode using bidirectional operation. High frequency characteristics make fast PWM
The mode is suitable for power regulation, rectification and DAC applications. High-frequency signals can be reduced by external components (inductive capacitors, etc.)
Size, thereby reducing system cost.

When the count reaches TOP, the timer counter overflow flag TOV3 will be set and the compare buffer value 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 compare signal OC3x can be obtained by setting the data direction register of the OC3x pin to output. The waveform of the
The frequency can be calculated using the following formula:

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

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

When a compare match occurs between TCNT3 and OCR3x, the waveform generator sets (clears) the OC3x signal. When TCNT3
When cleared, the waveform generator clears (sets) the OC3x signal to generate the PWM wave. Thus the extreme value of OCR3x
Will produce a special PWM waveform. When OCR3x is set to 0x00, the output PWM is counted for every (1 + TOP) count
The bell has a narrow spike. When OCR3x is set to TOP, the output waveform is a sustained high or low level.
If OCR3A is used as TOP and COM3A = 1 is set, the output compare signal OC3A generates a PWM with a duty cycle of 50%
wave.

### Phase correction PWM mode

When setting WGM3 [3: 0] = 1, 2, 3, 10 or 11, the timer counter 1 enters the phase correction PWM mode, counting
The maximum value TOP is 0xFF, 0x1FF, 0x3FF, ICR3 or OCR3A, respectively. The counter is operated in two directions by BOTTOM
Increment to TOP, and then decrement it to BOTTOM, and repeat this operation. The count is changed when the count reaches TOP and BOTTOM
Number of directions, the count value in TOP or BOTTOM only stay a count clock. In the process of increment or decrement, the count value
When TCNT3 matches OCR3x, the output compare signal OC3x will be cleared or set, depending on the compare output mode COM3
setting. Compared with one-way operation, the maximum frequency available for bi-directional operation is small, but its excellent symmetry is more suitable for electricity
Machine control.

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

The output compare signal OC3x waveform can be obtained by setting the data direction register of the OC3x pin to output. The frequency of the waveform
The rate can be calculated using the following formula:

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

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







During the count-up process, the waveform generator clears (sets) the OC3x signal when TCNT3 matches OCR3x.
The waveform generator sets (clears) the OC3x signal when TCNT3 matches OCR3x during decrement counting. by
The extreme value of this OCR3x will produce a special PWM wave. When the OCR3x is set to TOP or BOTTOM, the OC3x signal is lost
The meeting will remain low or high. If OCR3A is used as TOP and COM3A = 1 is set, the compare signal is output
OC3A generates a PWM wave with a duty cycle of 50%.

In order to ensure that the output PWM wave symmetry on both sides of the BOTTOM, in the absence of a comparison match, there are two cases
Will also flip the OC3x signal. The first case is when the value of OCR3x changes from TOP to other data. When OCR3x
Is TOP, when the count reaches TOP, the output of OC3x is the same as that of the previous descending order.
OC3x unchanged. The value of the new OCR3x (not TOP) is updated and the OC3x value remains until
When the count is matched, the match is reversed. The OC3x signal is not centered at the minimum, so it needs to be in TCNT3
When the maximum value is reached, the OC3x signal is flipped, which is the first case when the OC3x signal is flipped without a compare match. First
In both cases, when TCNT3 starts counting from a value higher than OCR3x, it will lose a compare match and thus
From the asymmetry of the situation. Also need to flip the OC3x signal to achieve the minimum on both sides of the symmetry.

### Phase frequency correction PWM mode

When setting WGM3 [3: 0] = 8 or 9, the timer counter 1 enters the phase frequency correction PWM mode, the maximum value of the count
TOP is ICR3 or OCR3A, respectively. The counter is bidirectional, incremented by BOTTOM to TOP, and then decremented to
BOTTOM, and repeat this operation. When the count reaches TOP and BOTTOM, the counting direction is changed and the count value is TOP or

BOTTOM only stay on a count clock. In the process of increment or decrement, when the count value TCNT3 matches OCR3x,
The output compare signal OC3x will be cleared or set, depending on the setting of the compare output mode COM3. And one-way operation phase
The maximum frequency available for bi-directional operation is smaller, but its excellent symmetry is more suitable for motor control.

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

The output compare signal OC3x waveform can be obtained by setting the data direction register of the OC3x pin to output. The frequency of the waveform
Can be calculated using the following formula:

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

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

During the count-up process, the waveform generator clears (sets) the OC3x signal when TCNT3 matches OCR3x.
The waveform generator sets (clears) the OC3x signal when TCNT3 matches OCR3x during decrement counting. by
The extreme value of this OCR3x will produce a special PWM wave. When the OCR3x is set to TOP or BOTTOM, the OC3x signal is lost
The meeting will remain low or high. If OCR3A is used as TOP and COM3A = 1 is set, the compare signal is output
OC3A generates a PWM wave with a duty cycle of 50%.

Since the OCR3x register is updated at the BOTTOM time, the TOP value is on both sides in ascending and descending order
Is the same, also produces a symmetrical waveform with both frequency and phase.

When using the fixed TOP value, it is best to use the ICR3 register as the TOP value, that is, set WGM3 [3: 0] = 8,





The OCR3A register is only used to generate the PWM output. If you want to generate a frequency change of the PWM wave, you must change the TOP
Value, OCR3A double buffering feature will be more suitable for this application.

### Enter the capture mode

The input capture is used to capture an external event and give it a time stamp to indicate when the event occurred.
, Except that the ICR3 value is used as the waveform generation mode for counting the TOP value.

Trigger signals for external events are entered by pin ICP3 and can also be implemented by analog comparator units. When the pin
The logic level on the ICP3 changes, or the output ACO level of the analog comparator changes, and this level changes
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 input capture flag ICF3 is set. If ICIE1 bit is "1", input capture flag will generate input
Into the capture interrupt.

The input capture is set by setting the analog compare control with the analog compare input control bit ACIC of the status register ACSR
Capture the source ICP3 or ACO. It should be noted that changing the trigger source may cause an input capture, so change the touch
The ICF3 must be cleared once it is generated to avoid erroneous results.

The input capture signal is passed to an edge detector via an optional noise suppressor, and the control bit is selected according to the input capture
ICES1 configuration, to see whether the detected edge to meet the trigger conditions. The noise suppressor is a simple digital filter, right
The input signal is sampled four times, and the output is fed to the edge detector only when the four samples are equal. Noise suppression
The controller is enabled or disabled by the ICNC1 bit of the TCCR3B register.

When using the input capture function, the ICR3 register value should be read as early as possible when ICF3 is set, since the next
The value of ICR3 will be updated after the capture event. 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 input capture time stamp can be used to calculate the frequency, duty cycle and other characteristics of the signal, as well as for triggering events
Log. When measuring the duty cycle of an external signal, it is required to change the trigger edge after each capture, so after reading the ICR3 value

Change the edge of the signal as soon as possible.

### Dead time control

When the DTEN3 bit is set to "1", the function of inserting the dead time is enabled. The output waveform of OC3A and OC3B will be
The output of the waveform is compared with the output of the waveform generated by inserting the set dead time, the length of the time for the DTR3 register count
The number of clocks corresponding to the time value. As shown in the following figure, OC3A and OC3B dead time insertion are channel B comparison to lose
Out of the waveform as a benchmark. When COM3A and COM3B are the same as "2" or "3", the waveform polarity of OC3A and the waveform polarity of OC3B
Similarly, when COM3A and COM3B are "2" or "3", OC3A waveform and OC3B waveform polarity is opposite.






Figure 7 TC3 dead time control in FPWM mode

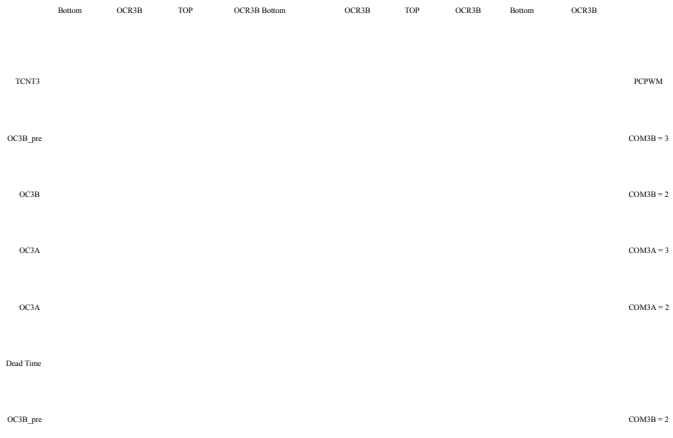



Figure 8 TC3 dead time control in PCPWM mode





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

## Register definition

TC3 register list

| register | address | Defaults | 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 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 | The timer counter interrupts the mask register |
| TIFR3 | 0x38 | 0x00 | Timer counter interrupt flag register |

### TCCR3A-TC3 Control Register A

TCCR3A -TC3 Control Register A

Address: 0x90 Default: 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.<br>COM3A1 and COM3A0 form COM3A [1: 0] to control the output compare waveform OC3A. in case<br>COM3A 1 bit or 2 bits are set, the output compare waveform occupies the OC3A pin, but<br>The pin's data direction register must be set high to output this waveform. In different working modes<br>, COM3A on the output comparison waveform control is also different, see the specific comparison output mode control table<br>Description of the grid. |
| 6 | COM3A0 | Compare match output A mode control low.<br>COM3A1 and COM3A0 form COM3A [1: 0] to control the output compare waveform OC3A. in case<br>COM3A 1 bit or 2 bits are set, the output compare waveform occupies the OC3A pin, but<br>The pin's data direction register must be set high to output this waveform. In different working modes |

**Page 157**



| | | |
|---|---|---|
| | | , COM3A on the output comparison waveform control is also different, see the specific comparison output mode control table Description of the grid. |
| 5 | COM3B1 | Compare match output B mode control high.<br>COM3B1 and COM3B0 form COM3B [1: 0] to control the output compare waveform OC3B. in case COM3B 1 bit or 2 bits are set, the output compare waveform occupies the OC3B pin, but The pin's data direction register must be set high to output this waveform. In different working modes , COM3B on the output comparison waveform control is also different, see the specific comparison output mode control table Description of the grid. |
| 4 | COM3B0 | Compare match output B mode control low.<br>COM3B1 and COM3B0 form COM3B [1: 0] to control the output compare waveform OC3B. in case COM3B 1 bit or 2 bits are set, the output compare waveform occupies the OC3B pin, but The pin's data direction register must be set high to output this waveform. In different working modes , COM3B on the output comparison waveform control is also different, see the specific comparison output mode control table Description of the grid. |
| 3 | COM3C1 | Compare match output C mode control high.<br>COM3C1 and COM3C0 form COM3C [1: 0] to control the output comparison waveform OC3C. in case COM3C 1 bit or 2 bits are set, the output compare waveform occupies the OC3C pin, however The pin's data direction register must be set high to output this waveform. In different working modes , COM3C on the output comparison waveform control is also different, see the specific comparison output mode control table Description of the grid. |
| 2 | COM3C0 | Compare match output C mode control low.<br>COM3C1 and COM3C0 form COM3C [1: 0] to control the output comparison waveform OC3C. in case COM3C 1 bit or 2 bits are set, the output compare waveform occupies the OC3C pin, however The pin's data direction register must be set high to output this waveform. In different working modes , COM3C on the output comparison waveform control is also different, see the specific comparison output mode control table Description of the grid. |
| 1 | WGM31 | waveform generation mode control sub-low.<br>WGM31 and WGM33, WGM32, WGM30 together constitute the waveform generation mode control WGM3 [3: 0], control counter count mode and waveform generation mode, see the waveform generation model Formula Description. |
| 0 | WGM30 | waveform generation mode control the lowest bit.<br>WGM30 and WGM33, WGM32, WGM31 together constitute the waveform generation mode control WGM3 [3: 0], control counter count mode and waveform generation mode, see the waveform generation model Formula Description. |

The following table shows the non-PWM mode (ie, normal mode and CTC mode), compare the output mode to the output compare waveform control.

Non-PWM mode compare output mode control

| COM3x [1: 0] | description |
|---|---|
| 0 | OC3x disconnect, general IO port operation |
| 1 | Match the OC3x signal when comparing the match |
| 2 | The OC3x signal is cleared when compare match |
| 3 | The OC3x signal is set when the compare match |

The following table shows the control of the output compare waveform for the compare output mode in fast PWM mode.



**Page 158**



Compare output mode control in fast PWM mode

| COM3x [1: 0] | Description |
|---|---|
| 0 | OC3x disconnect, general IO port operation |
| 1 | When WGM3 is 15: The OC3A signal is flipped when the compare match is reversed, and OC3B is disconnected<br>When WGM3 is other values: OC3x is disconnected and the common IO port operates |
| 2 | The OC3x signal is cleared when the compare match is set and the OC3x signal is set when the maximum match is made |
| 3 | The OC3x signal is set when the compare match is cleared and the OC3x signal is cleared when the maximum match is made |

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

Phase correction and phase frequency correction PWM mode compare output mode control

| COM3x [1: 0] | Description |
|---|---|
| 0 | OC3x disconnect, general IO port operation |
| 1 | WGM3 is 9 or 11: The OC3A signal is flipped when the compare match is reversed and the OC3B is disconnected<br>When WGM3 is other values: OC3x is disconnected and the common IO port operates |
| 2 | The compare match is cleared in the ascending order. The OC3x signal is cleared and the compare match is set in descending order<br>OC3x signal |
| 3 | The higher-order count match matches the OC3x signal, and the compare match is cleared in descending order<br>OC3x signal |

### TCCR3B-TC3 Control Register B

TCCR3B -TC3 Control Register B

Address: 0x91 Default: 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 | The input capture noise suppressor enables the control bit.<br>When the ICNC3 bit is set to "1", the input capture noise suppressor is enabled, and the external pin ICP3<br>Of the input is filtered, the input signal is valid when four consecutive samples are equal,<br>The incoming capture is delayed by four clock cycles.<br>When the ICNC3 bit is set to "0", the input capture noise suppressor is disabled, and the external pin ICP3<br>The input is directly valid. |
| 6 | ICES3 | The input capture trigger edge selects the control bit.<br>When setting the ICES3 bit to "1", the rising edge of the selected level triggers the input capture; when setting ICES3<br>When bit is "0", the falling edge of the selection level triggers the input capture.<br>When an event is captured, the counter value is copied to the ICR3 register,<br>Entry catch flag ICF3. If the interrupt is enabled, an input capture interrupt is generated. |
| 5 | - | Keep it. |
| 4 | WGM33 | Waveform Generation Mode Control High.<br>WGM33 and WGM32, WGM31, WGM30 together constitute the waveform generation mode control<br>WGM3 [3: 0], control counter count mode and waveform generation mode, see the waveform generation model<br>Formula Description. |
| 3 | WGM32 | waveform generation mode control times high. |





| | | |
|---|---|---|
| | | WGM32 and WGM33, WGM31, WGM30 together constitute the waveform generation mode control<br>WGM3 [3: 0], control counter count mode and waveform generation mode, see the waveform generation model<br>Formula Description. |
| 2 | CS32 | Clock select control high.<br>Used to select the clock source for timer counter 3. |
| 1 | CS31 | Clock selection control bit.<br>Used to select the clock source for timer counter 3. |
| 0 | CS30 | Clock select control low.<br>Used to select the clock source for timer counter 3. |

| CS3 [2: 0] | description |
|---|---|

| | |
|---|---|
| 0 | No clock source, stop counting |
| 1 | clk $_{sys}$ |
| 2 | clk $_{sys}$ / 8, from prescaler |
| 3 | clk $_{sys}$ / 64, from prescaler |
| 4 | clk $_{sys}$ / 256, from prescaler |
| 5 | clk $_{sys}$ / 1024, from prescaler |
| 6 | External clock T3 pin, falling edge trigger |
| 7 | External clock T3 pin, rising edge triggered |

The following table shows the waveform generation mode control.

Table 5 Waveform Generation Mode Control

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







### TCCR3C-TC3 control register C

TCCR3C -TC3 control register C

Address: 0x92 Default: 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.<br>When operating in non-PWM mode, the "1" mode can be written to the forced output compare bit FOC3A<br>To produce a comparison match. Forcing the match does not set the OCF3A flag, nor will it be overloaded or cleared<br>Zero timer, but the output pin OC3A will be updated according to COM3A settings accordingly<br>With the exact match of the same.<br>When working in PWM mode, write the TCCR3A register to clear it.<br>The return value of reading FOC3A is always zero. |
| 6 | FOC3B | Force the output comparison B.<br>When operating in non-PWM mode, the "1" mode can be written to the forced output compare bit FOC3B<br>To produce a comparison match. Forcing the match does not set the OCF3B flag, nor will it be overloaded or cleared |

Zero timer, but the output pin OC3B will be updated according to COM3B settings accordingly
With the exact match of the same.
When working in PWM mode, write the TCCR3A register to clear it.
The return value of the read FOC3B is always zero.

5 The DOC3B disables the output compare B enable control bit.
When the DOC3B bit is high, the hardware disable output compare B is enabled, and when the condition of the disable output is satisfied , The COM3B bit is cleared and the output pin, OC3B, is turned off and the pin becomes a common IO operation.
When the DOC3B bit is low, the hardware disable output compare B function is invalid.

4 DOC3A disables the output compare A enable control bit.
When the DOC3A bit is high, the hardware disable output compare A is enabled, and when the condition of the disabled output is satisfied , The COM3A bit is cleared and the output pin, OC3A, is turned off, and the pin becomes a general-purpose IO operation.
When the DOC3A bit is low, the hardware disable 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
Press DTR3L and DTR3H to set the dead time.
When the DTEN3 bit is low, the dead time is disabled. OC3A and OC3B are single output.

2 -

1 The DOC3C prohibits the output of the compare C enable control bits.
When the DOC3C bit is high, the hardware disable output compare C is enabled, and when the condition of the disabled output is satisfied , The COM3C bit is cleared and the output pin OC3C is disconnected, which becomes a general purpose IO operation.
When the DOC3C bit is low, the hardware disable output compare the C function is invalid.

0 FOC3C Force the output to compare C.
When operating in non-PWM mode, the "1" mode can be written to the forced output compare bit FOC3C
To produce a comparison match. Forcing the match does not set the OCF3C flag, nor will it be overloaded or cleared
Zero timer, but the output pin OC3C will be updated according to the COM3C settings





With the exact match of the same.
When working in PWM mode, write the TCCR3A register to clear it.
The return value of the read FOC3C is always zero.

### TCCR3D-TC3 control register D

TCCR3D -TC3 control register D

Address: 0x93 Default: 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 enabled bit 7.
When the DSX37 bit is set to "1", TC0 overflows as the output signal for the compare output waveform OC3x
The trigger source is enabled. When the DOC3x bit is "1", the interrupt flag register of the selected trigger source
The rising edge of the bit will automatically turn off the OC3x waveform output.
When the DSX37 bit is set to "0", TC0 overflows as the output signal for the compare output waveform OC3x
The trigger source is disabled.

6 DSX36 TC3 Trigger Source Select Control Enable bit 6.
When the DSX36 bit is set to "1", TC2 overflows as the output signal to be turned off for comparison. OC3x
The trigger source is enabled. When the DOC3x bit is "1", the interrupt flag register of the selected trigger source
The rising edge of the bit will automatically turn off the OC3x waveform output.
When the DSX36 bit is set to "0", TC2 overflows as the output signal of the output signal is closed. OC3x
The trigger source is disabled.

5 DSX35 TC3 Trigger Source Select Control Enable Bit 5.
When the DSX35 bit is set to "1", the pin level changes by 1 as the output signal
The trigger source for OC3x is enabled. When the DOC3x bit is "1", the interrupt flag of the selected trigger source
The rising edge of the register bit will automatically turn off the OC3x waveform output.

When the DSX35 bit is set to "0", the pin level changes by 1 as the output signal

The trigger source for OC3x is disabled.

4 DSX34 TC3 Trigger Source Select Control Enable bit 4.

When the DSX34 bit is set to "1", the external interrupt 1 is used to turn off the output compare signal waveform

The trigger source for OC3x is enabled. When the DOC3x bit is "1", the interrupt flag of the selected trigger source is sent

The rising edge of the register bit will automatically turn off the OC3x waveform output.

When the DSX34 bit is set to "0", the external interrupt 1 is used to turn off the output compare signal waveform

The trigger source for OC3x is disabled.

3: 2 - Keep it.

1 DSX31 TC3 Trigger Source Select Control Enable bit 1.

When the DSX31 bit is set to "1", analog comparator 1 is used to turn off the output compare signal waveform

The trigger source for OC3x is enabled. When the DOC3x bit is "1", the interrupt flag of the selected trigger source is sent

The rising edge of the register bit will automatically turn off the OC3x waveform output.

When the DSX31 bit is set to "0", the analog comparator 1 is used to turn off the output compare signal waveform

The trigger source for OC3x is disabled.

0 DSX30 TC3 Trigger Source Select Control Enable bit 0.





When setting the DSX30 bit to "1", analog comparator 0 is used to turn off the output compare signal waveform

The trigger source for OC3x is enabled. When the DOC3x bit is "1", the interrupt flag of the selected trigger source is sent

The rising edge of the register bit will automatically turn off the OC3x waveform output.

When the DSX30 bit is set to "0", the analog comparator 0 is used as the output signal to turn off the comparison

The trigger source for OC3x is disabled.

The following table controls the selection of the trigger source for the waveform output.

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

| DOC3x | DSX3n = 1 trigger source | | description |
|---|---|---|---|
| 0 | - | - | The DOC3x bit is "0" and the trigger source off waveform output function is disabled |
| 1 | 0 | Analog comparator 0 | The rising edge of ACIF0 will turn off the OC3x waveform output |
| 1 | 1 | Analog comparator | The rising edge of ACIF1 will turn off the OC3x waveform output |
| 1 | 4 | External interrupt 1 | The rising edge of INTF1 will turn off the OC3x waveform output |
| 1 | 5 | Pin Level Change 1 | The rising edge of PCIF1 turns off the OC3x waveform output |
| 1 | 6 | TC2 overflow | The rising edge of TOV2 will turn off the OC3x waveform output |
| 1 | 7 | TC0 overflows | The rising edge of TOV0 will turn off the OC3x waveform output |

note:

2) When DSX3n = 1 indicates that 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: 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 | TC3 The low byte of the count value.<br>TCNT3H and TCNT3L together to form TCNT3, through the TCNT3 register can be directly on the counter 16 bits<br>Count value for read and write access. Reading and writing 16-bit registers requires two operations. When writing 16-bit TCNT3, write TCNT3H first.<br>When reading 16-bit TCNT3, read TCNT3L first.<br>The CPU writes to the TCNT3 register will prevent the occurrence of a compare match at the next timer clock cycle, even if the timer<br>The device has stopped. This allows the value of the initialization TCNT3 register to be consistent with the value of OCR3x without causing an interrupt.<br>If the value written to TCNT3 equals or bypasses the OCR3x value, the compare match is lost, resulting in an incorrect waveform<br>Health results.<br>The timer stops counting when no clock source is selected, but the CPU can still access TCNT3. CPU write counter is cleared or added<br>The priority of the operation is high. |

### TCNT3H-TC3 counter register high byte

TCNT3H -TC3 count register high byte

Address: 0x95 Default: 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 | TC3 The high byte of the count value.<br>TCNT3H and TCNT3L together to form TCNT3, through the TCNT3 register can be directly on the counter 16-bit meter<br>Read and write access values. Reading and writing 16-bit registers requires two operations. When writing 16-bit TCNT3, write TCNT3H first.<br>When reading 16-bit TCNT3, read TCNT3L first.<br>The CPU writes to the TCNT3 register will prevent the occurrence of a compare match at the next timer clock cycle, even if the timer<br>The device has stopped. This allows the value of the initialization TCNT3 register to be consistent with the value of OCR3x without causing an interrupt.<br>If the value written to TCNT3 equals or bypasses the OCR3x value, the compare match is lost, resulting in an incorrect waveform<br>Health results.<br>The timer stops counting when no clock source is selected, but the CPU can still access TCNT3. CPU write counter is cleared or added<br>The priority of the operation is high. |

### ICR3L-TC3 capture register low byte

ICR3L -TC3 input capture register low byte

Address: 0x96 Default: 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 Enter the low byte of the capture value.<br>ICR3H and ICR3L combine to form 16-bit ICR3. Reading and writing 16-bit registers requires two operations. Write 16 bits<br>ICR3 should be written to ICR3H first. When reading 16-bit ICR3, read ICR3L first.<br>When the input capture is triggered, the count value TCNT3 is updated to the ICR3 register. The ICR3 register can also be used<br>Defines the TOP value of the count. |

### ICR3H-TC3 Capture Register High Byte

ICR3H -TC3 input capture register high byte

Address: 0x97 Default: 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 Enter the high byte of the capture value.<br>ICR3H and ICR3L combine to form 16-bit ICR3. Reading and writing 16-bit registers requires two operations. Write 16 bits<br>ICR3 should be written to ICR3H first. When reading 16-bit ICR3, read ICR3L first.<br>When the input capture is triggered, the count value TCNT3 is updated to the ICR3 register. The ICR3 register can also be used |

Defines 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: 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 | Output the lower byte of compare register A.<br>OCR3AL and OCR3AH combine to form 16-bit OCR3A. Reading and writing 16-bit registers requires two operations. write<br>16-bit OCR3A, should first write OCR3AH. When reading 16-bit OCR3A, read OCR3AL first.<br>OCR3A is compared with the counter value TCNT3 without interruption. The compare match can be used to generate an output compare interrupt, or<br>Used to generate waveforms on the OC3A pin.<br>When using PWM mode, the OCR3A register uses a double buffered register. While the normal working mode and match clear mode<br>, Double buffering is disabled. Double buffering can be updated with the OCR3A register with the count maximum or minimum moment<br>Synchronize, thus preventing the generation of asymmetric PWM pulses, eliminating the interference pulse.<br>When using the double buffering function, the CPU accesses the OCR3A buffer register, which disables the double buffering function when the CPU accesses<br>OCR3A itself. |

### OCR3AH-TC3 Output compare register A high byte

OCR3AH -TC3 Output compare register A high byte

Address: 0x99 Default: 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 | Outputs the high byte of compare register A.<br>OCR3AL and OCR3AH combine to form 16-bit OCR3A. Reading and writing 16-bit registers requires two operations. write<br>16-bit OCR3A, should first write OCR3AH. When reading 16-bit OCR3A, read OCR3AL first.<br>OCR3A is compared with the counter value TCNT3 without interruption. The compare match can be used to generate an output compare interrupt, or<br>Used to generate waveforms on the OC3A pin.<br>When using PWM mode, the OCR3A register uses a double buffered register. While the normal working mode and match clear mode<br>, Double buffering is disabled. Double buffering can be updated with the OCR3A register with the count maximum or minimum moment<br>Synchronize, thus preventing the generation of asymmetric PWM pulses, eliminating the interference pulse.<br>When using the double buffering function, the CPU accesses the OCR3A buffer register, which disables the double buffering function when the CPU accesses<br>OCR3A itself. |

### OCR3BL-TC3 Output Compare Register B Low Byte

OCR3BL -TC3 Output Compare Register B Low Byte

Address: 0x9A Default: 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 | Output the lower byte of compare register B.<br>OCR3BL and OCR3BH combine to form 16-bit OCR3B. Reading and writing 16-bit registers requires two operations. write<br>16-bit OCR3B, should first write OCR3BH. When reading 16-bit OCR3B, read OCR3BL first.<br>OCR3B is compared with the counter value TCNT3 without interruption. The compare match can be used to generate an output compare interrupt,<br>Or used to generate waveforms on the OC3B pin.<br>When using PWM mode, the OCR3B register uses a double buffered register. While the normal working mode and matching clear mode<br>The double buffering function is disabled. Double buffering can be updated when the OCR3B register is counted with the maximum or minimum value<br>Engraved together to prevent the generation of asymmetric PWM pulses, eliminating the interference pulse.<br>When using the double buffering function, the CPU accesses the OCR3B buffer register, which disables the double buffering function when the CPU accesses<br>OCR3B itself. |

### OCR3BH-TC3 Output Compare Register B High Byte

OCR3BH -TC3 Output Compare Register B High Byte

Address: 0x9B Default: 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 | Outputs the high byte of compare register B.<br>OCR3BL and OCR3BH combine to form 16-bit OCR3B. Reading and writing 16-bit registers requires two operations. write<br>16-bit OCR3B, should first write OCR3BH. When reading 16-bit OCR3B, read OCR3BL first.<br>OCR3B is compared with the counter value TCNT3 without interruption. The compare match can be used to generate an output compare interrupt, or<br>Used to generate waveforms on OC3B pins.<br>When using PWM mode, the OCR3B register uses a double buffered register. While the normal working mode and matching clear mode<br>The double buffering function is disabled. Double buffering can be updated when the OCR3B register is counted with the maximum or minimum value<br>Engraved together to prevent the generation of asymmetric PWM pulses, eliminating the interference pulse.<br>When using the double buffering function, the CPU accesses the OCR3B buffer register, which disables the double buffering function when the CPU accesses<br>OCR3B itself. |

### OCR3CL-TC3 Output Compare Register C Low Byte

OCR3CL -TC3 Output Compare Register C Low Byte

Address: 0x9E Default: 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 | Output the lower byte of compare register C.<br>OCR3CL and OCR3CH together to form a 16-bit OCR3C. Reading and writing 16-bit registers requires two operations. Write 16<br>Bit OCR3 should be written to OCR3CH. When reading 16-bit OCR3C, you should read OCR3CL first.<br>OCR3C is continuously compared with the counter value TCNT3. The compare match can be used to generate an output compare interrupt, or either<br>Used to generate waveforms on OC3C pins.<br>When using PWM mode, the OCR3C register uses a double buffered register. While the normal working mode and match clear mode<br>, Double buffering is disabled. Double buffering can update the OCR3C register with the count maximum or minimum time |

Step up to prevent the generation of asymmetric PWM pulses, eliminating the interference pulse.
When using the double buffering function, the CPU accesses the OCR3C buffer register, which disables the dual buffering function when the CPU accesses the OCR3C itself.

### OCR3CH-TC3 Output compare register C high byte

OCR3CH -TC3 Output compare register C high byte

Address: 0x9F Default: 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 | Output the high byte of compare register C.<br>OCR3CL and OCR3CH together to form a 16-bit OCR3C. Reading and writing 16-bit registers requires two operations. write 16-bit OCR3C, you should first write OCR3CH. When reading 16-bit OCR3C, you should read OCR3CL first.<br>OCR3C is continuously compared with the counter value TCNT3. The compare match can be used to generate an output compare interrupt, or Used to generate waveforms on OC3C pins.<br>When using PWM mode, the OCR3C register uses a double buffered register. While the normal working mode and match clear mode , Double buffering is disabled. Double buffering can update the OCR3C register with the count maximum or minimum time Synchronize, thus preventing the generation of asymmetric PWM pulses, eliminating the interference pulse.<br>When using the double buffering function, the CPU accesses the OCR3C buffer register, which disables the double buffering function when the CPU accesses OCR3C itself. |

### DTR3L-TC3 dead time register low byte

DTR3L -TC3 dead time register low byte

Address: 0x9C Default: 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.<br>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 Count clock decision. |





### DTR3H-TC3 dead time register high byte

DTR3H -TC3 dead time register high byte

Address: 0x9D Default: 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.<br>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 Count clock decision. |

### TIMSK3-TC3 interrupt mask register

TIMSK3 - TC3 interrupt mask register

Address: 0x71 Default: 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 | - | Keep it. |
| 5 | ICIE3 | TC3 input capture interrupt enable control bit.<br>When the ICIE3 bit is "1" and the global interrupt is set, the TC3 input capture interrupt is enabled. When the input capture trigger,<br>That is, the ICF3 flag of TIFR3 is set and the interrupt occurs.<br>When the ICIE3 bit is "0", the TC3 input capture interrupt is disabled. |
| 4 | - | Keep it. |
| 3 | OCIE3C | TC3 Output Compare C Match Interrupt Enable bit.<br>When the OCIE3C bit is "1" and the global interrupt is set, the TC3 output compares the C match interrupt enable. When a match match occurs<br>, The interrupt is generated when the OCF3C bit in TIFR3 is set.<br>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.<br>When the OCIE3B bit is "1" and the global interrupt is set, the TC3 output compares the B match interrupt enable. When a match match occurs<br>, The interrupt is generated when the OCF3B bit in TIFR3 is set.<br>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.<br>When the OCIE3A bit is "1" and the global interrupt is set, the TC3 output compares the A match interrupt enable. When a match match occurs<br>, That is, when the OCF3A bit in TIFR3 is set, the interrupt is generated. When the OCIE3A bit is "0", the TC3 output compares A<br>Match interrupt is disabled. |
| 0 | TOIE3 | TC3 overflow interrupt enable bit.<br>When the TOIE3 bit is "1" and the global interrupt is set, the TC3 overflow interrupt is enabled. When TC3 overflows, that is, TIFR3<br>The TOV3 bit is set when the interrupt is generated. 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: 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 | - | Keep it. |
| 5 | ICF3 | Enter the capture flag.<br>The ICF3 flag is set when an input capture event occurs. When ICR3 is used as the TOP value of the count,<br>When the value reaches the TOP value, the ICF3 flag is set. If ICIE1 is "1" and the global interrupt flag is set,<br>Will generate an input capture interrupt. ICF3 is automatically cleared when writing to this interrupt service routine or written to ICF3 bit<br>"1" can also be cleared to this bit. |
| 4 | - | Keep it. |
| 3 | OCF3C | Output compare the C match flag.<br>When TCNT3 is equal to OCR3C, the comparison unit gives the match signal and sets the compare flag OCF3C.<br>If the output compare interrupt enable OCIE3C is "1" and the global interrupt flag is set, the output ratio<br>More interrupted. OCF3C will be cleared automatically when the interrupt service routine is executed, or "1" for OCF3C bits can be cleared<br>Zero this bit. |
| 2 | OCF3B | Output Compare B match flag.<br>When TCNT3 is equal to OCR3B, the compare unit gives the match signal and sets the compare flag OCF3B.<br>If the output compare interrupt enable OCIE3B is "1" and the global interrupt flag is set, the output ratio<br>More interrupted. OCF3B will be cleared automatically when this interrupt service routine is executed, or "1" for OCF3B bits can be cleared<br>Zero this bit.<br>Output Compare A match flag. |

| | | |
|---|---|---|
| 1 | OCF3A | When TCNT3 is equal to OCR3A, the compare unit gives the match signal and sets the compare flag OCF3A.<br>If the output compare interrupt enable OCIE3A is "1" and the global interrupt flag is set, the output ratio<br>More interrupted. OCF3A will be cleared automatically when the interrupt service routine is executed, or "1" for OCF3A bit can be cleared<br>Zero this bit. |
| 0 | TOV3 | Overflow flag.<br>When the counter overflows, set the overflow flag TOV3. If the overflow interrupt enable TOIE3 is "1"<br>And the global interrupt flag is set, an overflow interrupt is generated. TOV3 will be executed when this interrupt service routine is executed<br>It is also possible to clear this bit by clearing it or writing "1" to the TOV3 bit. |



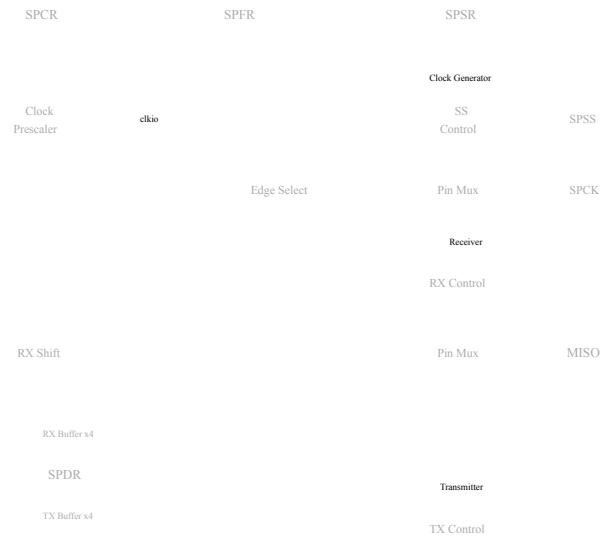


## Synchronous Serial Peripheral Interface (SPI)

- Full-duplex, three-wire synchronous data transmission
- Host or slave operation
- The least significant or most advanced bit is transmitted first
- 7 programmable bit rates
- Send end interrupt flag
- Write conflict flag protection mechanism
- wake-up from idle mode
- The host operation has speed mode
- Supports host two-wire input mode
- There are 4 cache registers for input / output

### Summary

SPI mainly consists of three parts: clock prescaler, clock detector, slave select detector, transmitter and receiver Device.



TX Shift Pin Mux MOSI

SPI structure diagram

The control and status registers are shared by these three parts. The clock prescaler works only in the master operating mode by bit
Rate control bit to select the division factor, resulting in the corresponding frequency division clock, the output to the SPCK pin. Clock detector only
Work in slave operation mode, detect the clock edge input from the SPCK pin, according to the SPI data transfer mode
Send and receive shift register for shift operation. The slave selection detector detects the slave selection signal SPSS





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

Clock generation logic is divided into host clock prescaler and slave clock detector, respectively, in the host operation and from the machine
Mode. The clock prescaler selects the division factor from the bit rate control bit and the speed control bit, resulting in the corresponding frequency division
Clock (a total of 7 kinds of optional frequency division coefficient, see the register description for details), output to the SPCK pin for communication
Clock, while providing a shift clock for the internal transmit and receive shift registers. The clock detector performs edge on the input clock SPCK
Along the detection, according to the SPI data transfer mode on the transmitter and receiver shift operation. To ensure positive for the clock signal
Sampling, SPCK clock high and low width must be greater than 2 system clock cycles.

### Send and receive

The SPI module supports both simultaneous transmission and reception in single-wire mode, and only supports dual-line reception in dual-line mode.

### Single line send and receive

SPI host will need to communicate from the slave select signal SPSS pulled low, you can start a transmission process. Host and slave
Will need to transfer the data ready, the host clock signal SPCK generate clock pulses to exchange data, the host data
Removed from the MOSI, moved from the MISO, the slave data from the MISO removed, moved from the MOSI, after the exchange of data after the host
Pull the SPSS signal to complete the communication.

When configured as a master, the SPI module does not control the SPSS pin and must be handled by the user software. Software pulls down SPSS
Pin, select the slave to communicate, start the transmission. The software will need to transfer the data to the SPDR register when it is started
Clock generator, the hardware generates the clock of the communication, and moves the 8-bit data out of the slave, while moving the slave data into the slave. Shift
After a byte of data, the clock generator is stopped and the transmit completion flag SPIF is set. The software can write the data again
SPDR register to continue transmitting the next byte, or you can pull the SPSS signal to end the current transfer. The last number
Will be stored in the receive buffer.

When configured as a slave, the SPI module will remain asleep as long as the SPSS signal is always high and keep the MISO
Feet for the three states. The software can update the contents of the SPDR register. Even if there is an input clock pulse on the SPCK pin at this time,
SPDR data will not be removed until the SPSS signal is pulled low. When a byte of data is transferred, the hardware is done
Set the transmission completion flag SPIF. At this point the software to read the data before moving to the SPDR register to continue to write data,
The last incoming data will be stored 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 hair
When the buffer is in the non-full state (ie, the transmit buffer full flag WRFULL bit is low), the SPDR register
Write operation. And when receiving data, when the receive buffer is non-empty state (ie, receive buffer empty flag RDEMPT
Bit is low), you can read the received characters by accessing the SPDR register.

### Host two-wire reception

SPI module two-line mode is only effective in the host operating mode, and the single-line mode is different MOSI and MISO are
For the host to receive data, each SPCK clock pulse simultaneously receives 2 bits of data (the data on the MISO line is

Before the data on the MOSI line is on), after receiving the two bytes of data, the hardware sets the transmission completion flag SPIF, the data Saved to the receive buffer and shift register. At this point the software must read the SPDR register twice to get the two received Byte of data. It should be noted that, although the two-line mode, the host does not send data to the slave, the software still need to SPDR Register write data to start the clock generator to generate the communication clock, write once the SPDR register to receive two bytes The data.

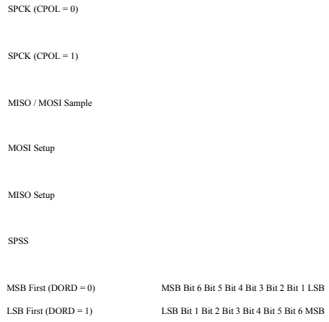
### Data pattern

Single-line mode, SPI relative to the serial data, there are four SPCK phase and polarity combination, by the CPHA and CPOL To control, as shown in the following table.

CPHA and CPOL Select the data transfer mode

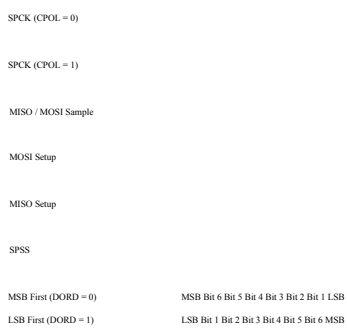
| CPOL | CPHA | Starting along | End the edge | SPI mode |
|---|---|---|---|---|
| 0 | 0 | Sampling (rising edge) | Set (falling edge) | 0 |
| 0 | 1 | Set (rising edge) | Sampling (falling edge) | 1 |
| 1 | 0 | Sampling (falling edge) | Set (rising edge) | 2 |
| 1 | 1 | Set (falling edge) | Sampling (rising edge) | 3 |

When CPHA = 0, the data samples and set the clock edge as shown below:




CPHA is "0" when SPI data transfer mode

When CPHA = 1, the data samples and set the clock edge as shown below:



CPHA is "1" when SPI data transfer mode

Two-wire mode, MISO and MISO are used as host input, data sampling time is still determined by the data transmission mode, mining
The way we look like this:

| MISO / MOSI Sample | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| MSB First (DORD = 0) | | | | | | | | |
| MISO | MSB | Bit 5 | Bit 3 | Bit 1 | MSB | Bit 5 | Bit 3 | Bit 1 |
| MOSI | Bit 6 | Bit 4 | Bit 2 | LSB | Bit 6 | Bit 4 | Bit 2 | LSB |
| LSB First (DORD = 1) | | | | | | | | |
| MISO | LSB | Bit 2 | Bit 4 | Bit 6 | LSB | Bit 2 | Bit 4 | Bit 6 |
| MOSI | Bit 1 | Bit 3 | Bit 5 | MSB | Bit 1 | Bit 3 | Bit 5 | MSB |

Bit mode when the DUAL is "1" when the SPI data sampling mode

### SPSS pin function

When configured as a slave, the slave select signal SPSS pin is always used as input. When the SPSS pin remains low, the SPI is connected
The port is activated, the MISO pin becomes the output pin (the software performs the corresponding port configuration), and the other pins are input. When SPSS
When the pin is held high, the SPI module is reset and no longer receives data. The SPSS pin is very synchronous for packet / byte synchronization
It is useful to synchronize the slave's bit counter with the host's clock generator. When SPSS is pulled high, the SPI slave is immediately reset
Receive and send logic, and discard the incomplete data in the shift register.

When configured as a host, the user software can determine the direction of the SPSS pin.
If SPSS is configured as an output, it can be used to drive the slave's SPSS pin. If SPSS is configured as an input, it must be maintained
High to ensure the normal work of the host. When configured as a master and the SPSS pin is an input, the external circuit pulls down the SPSS pin
, The SPI module will consider another host to choose itself as a slave and start transmitting data. In order to prevent bus conflicts,
The SPI module will perform the following actions:
1. Clear the MSTR bit in the SPCR register, convert to slave, so MOSI and SPCK become input;
2. Set the SPIF bit in the SPSR register. If the interrupt is enabled, an SPI interrupt is generated.
Therefore, the use of interrupt mode to deal with SPI host data transmission, and there is the possibility of being pulled down SPSS, the interrupt service
The program should check whether the MSTR bit is "1". If cleared, the software must set it to re-enable SPI master mode.

### SPI initialization

The SPI is first initialized before communication is performed. The initialization process usually includes the choice of host slave operation, data transfer
The setting of the input mode, the selection of the bit rate, and the direction control of the respective pins. Where the host and slave operate under the pin side
The controls are different, as shown in the following table:

Pin direction control

| Pin | The direction of the host mode | Slave mode direction |
|---|---|---|
| MOSI | User software definition | enter |
| MISO | enter | User software definition |
| SPCK | User software definition | enter |
| SPSS | User software definition | enter |



### SPI host initialization

The SPI host mode initialization process is as follows:
1. Set the MSTR bit, set the bit rate selection control bit, data transfer mode, data transfer order, interrupt enable or not,
As well as double-line enabled or not;
2. Set the MOSI and SPCK pins to output;
3. Set the SPE bit.
In host mode, the SPSS pin can be set to output when the SPI module is not desired to be used as a slave by another host.

### SPI slave initialization

The SPI slave mode initialization process is as follows:

1. Clear the MSTR bit, set the data transfer mode, data transfer order, interrupt enable or not;
2. Set the MISO pin to output;
3. Set the SPE bit.

### Register definition

SPI register list

| register | address | Defaults | 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: 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.
The SPI interrupt is enabled when the SPIE bit is set to "1". When the SPIF is located in the SPSR register
Bit is set and the global interrupt is enabled, an SPI interrupt is generated.
When the SPIE bit is set to "0", the SPI interrupt is disabled.

6 SPE
SPI enable bit.
The SPI module is enabled when the SPE bit is set to "1". It must be set before any SPI operation
Bit SPE.
When the SPE bit is set to "0", the SPI module is disabled.

5 DORD
Data order control bit.
When the DORD bit is set to "1", the LSB of the data is transmitted first.





When the DORD bit is set to "0", the MSB of the data is transmitted first.

4 MSTR
Host Slave Selects the control bit.
When the MSTR bit is set to "1", the host mode is selected.
When the MSTR bit is set to "0", the slave mode is selected.
In master mode, when the SPSS pin is configured as an input and is pulled low, the MSTR bit is cleared and bits
The SPIF in the SPSR register is set and the user must reset the MSTR into host mode.

3 CPOL
Clock polarity control bit.
When the CPOL bit is set to "1", SPCK is high in the idle state.
When the CPOL bit is set to "0", SPCK is low in the idle state.

| CPOL | Starting along | End the edge |
|---|---|---|
| 0 | Rising edge | Falling edge |
| 1 | Falling edge | Rising edge |

2 CPHA clock phase control bit.
When the CPHA bit is set to "1", the start edge sets the data and ends the edge of the sampled data.
When the CPHA bit is set to "0", the start edge is sampled and the edge edge sets the data.

| CPHA | Starting along | End the edge |
|---|---|---|
| 0 | sampling | Set up |

1 Set up sampling

1 SPR1 clock rate select bit 1.

SPR1 and SPR0 are used to select the SPI transfer clock rate. For specific control methods see SPCK and System clock relationship table.

0 SPR0 clock rate select bit 0.

SPR1 and SPR0 are used to select the SPI transfer clock rate. For specific control methods see SPCK and System clock relationship table.

### SPSR - SPI status register

SPSR - SPI status register

Address: 0x4D Default: 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.
After the serial transmission is set, the SPIF flag is set. In host mode, the SPSS pin is configured as input and
When pulled low, SPIF will also be set. If the SPIE bit of the SPCR register and the global interrupt
The enable bit is set and the SPI interrupt is generated. The SPIF bit is cleared automatically after entering the interrupt service routine
Zero, or clear the SPIF bit by first reading the SPSR register and then accessing the SPDR register.

6 WCOL
Write conflicting flag.
Writing to the SPDR register during data transfer will set the WCOL bit. WCOL bit can pass
Read the SPSR register first and then access the SPDR register to clear it.





5 - Keep it.

4 - Keep it.

3 - Keep it.

2 DUAL
Two-wire mode control bit.
When the DUAL bit is set to "1", the SPI 2-wire transmission mode is enabled.
When the DUAL bit is set to "0", the SPI 2-wire transfer mode is disabled.
The two-wire transmission mode is valid only in SPI master mode, and MISO and MOSI are used as host numbers
According to the input, the data transmission mode is described in the host double line receiving and data mode section description.

1 - Keep it.

0 SPI2X
SPI speed control bit.
When the SPI2X bit is set to "1", the SPI transmission speed is doubled.
When the SPI2X bit is set to "0", the SPI transmission speed is not doubled.
Refer to SPCK and the system clock for the specific control mode.

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

The relationship between SPCK and the 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: 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 | SPI sends and receives data.<br>SPI sends data and receives data to share SPI data register SPDR. Write data to SPDR<br>That is, the write data shift register, read data from the SPDR read the receive data buffer<br>Device. |





SPFR - SPI buffer register

SPFR - **SPI** buffer register

Address: 0x39 Default: 0x00

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | RDFUL1 | 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.<br>When the data in the receive buffer reaches four bytes, the RDFULL bit is high indicating that the receive buffer<br>When the device is full, the interrupt flag is set at the same time.<br>When the data in the receive buffer is less than four bytes, the RDFULL bit is low, indicating that the receive buffer<br>The device is not full. |
| 6 | RDEMPT | Receive buffer empty flag.<br>When no data is received, the RDEMPT bit is high, indicating that the receive buffer is empty.<br>When the data is received, it will be stored in the receive buffer, RDEMPT bit is low, indicating that the receiver<br>The MCU can be read by the SPDR register to read the receive buffer<br>data.<br>When the RDEMPT bit is set (write 1), the receive buffer address is reset to zero.<br>When the RDEMPT bit and the WREMPT bit are set at the same time, the receive buffer address and<br>The receive shift register pointer will be reset to zero. |
| 5 | RDPTR1 | receives the buffer address high. |
| 4 | RDPTR0 | Receive buffer address low.<br>When the SPDR register is read, the MCU will read from the receive buffer<br>Of the data, while receiving the buffer address will be accumulated.<br>When the RDEMPT bit is set (write 1), the receive buffer address is reset to zero. |
| 3 | WRFULL | Send buffer full flag.<br>When the data in the transmit buffer reaches four bytes, the WRFULL bit is high, indicating that the transmit<br>When the punch is full, the interrupt flag SPIF is set.<br>When the data in the transmit buffer is less than four bytes, the WRFULL bit is low, indicating that the transmit<br>Punch is not full. |
| 2 | WREMPT | Send buffer empty flag.<br>When the SPDR register is not written, or the data written to the transmit buffer has been sent<br>When finished, the WREMPT bit is high, indicating that the transmit buffer is empty.<br>When the SPDR register is written, the transmit buffer address is accumulated and the write buffer is sent<br>When the data of the punch is not all sent, the receiving buffer has at least one byte of data, |

| | | |
|---|---|---|
| | | The WREMPT bit is low, indicating that the transmit buffer is not empty. |
| | | When the WREMPT bit is set (write 1), the transmit buffer address is reset to zero. |
| 1 | WRPTR1 | Transmit Buffer Address High. |
| | | Send buffer address low. |
| 0 | WRPTR0 | When writing to the SPDR register, the data in the SPDR will be written to the transmit buffer, |
| | | While sending a buffer address will be accumulated. |
| | | When the WREMPT bit is set (write 1), the transmit buffer address is reset to zero. |



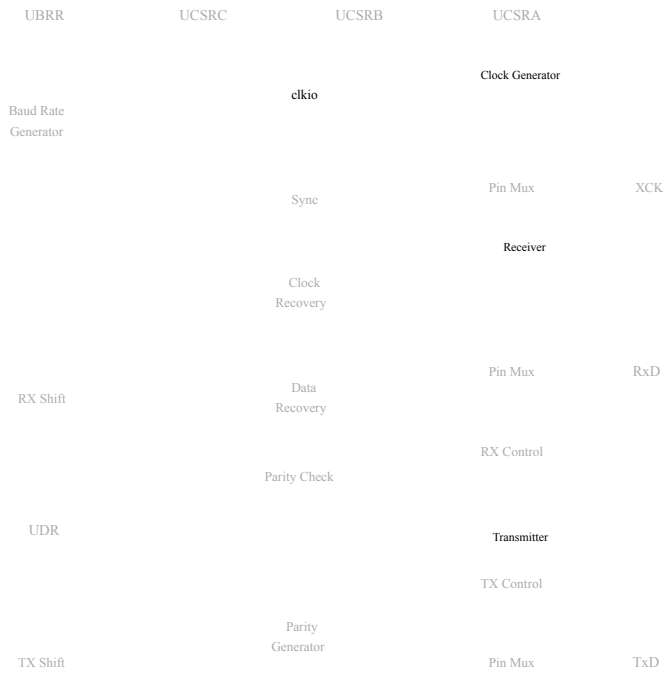
## USART0 - Universal Synchronous / Asynchronous Serial Transceiver

- Full-duplex operation (independent serial receive and transmit registers)
- Asynchronous or synchronous operation
- Host or slave operation

High-precision baud rate generator

- Supports 5, 6, 7, 8, or 9 data bits and 1, or 2 stop bits
- Hardware-enabled parity generation and verification mechanisms
- Data overspeed detection
- Frame error detection
- Noise filtering, including erroneous start bit detection and digital low-pass filter
- Three independent interrupts: Transmit end interrupt, transmit data register empty interrupt, and receive end interrupt
- Multi-processor communication mode
- Asynchronous communication mode

### Summary




USART structure diagram

The USART mainly consists of three parts: a clock generator, a transmitter and a receiver. The control and status registers are made up of these three sections Share. The clock generator consists of the synchronization logic of the external input clock from the baud rate generator and the synchronous slave operating mode. The XCK pin is used only for synchronous transfer mode. The transmitter includes a write data buffer, a serial shift register, and a parity occurs And the control logic required to handle different frame formats. The write data buffer allows continuous transmission of data without being in the data frame Between the introduction of delay. The receiver has a clock and data recovery unit for reception of asynchronous data. In addition to the recovery unit, receive The device also includes parity, control logic, serial shift register and a two-stage receive buffer UDR. Receiver support with The transmitter has the same frame format, and can detect frame errors, data overspeed and parity errors.

### Clock generation

The clock generation logic generates the base clock for the transmitter and receiver. The USART supports four modes of clock: normal asynchronous mode Type, speed asynchronous mode, host synchronization mode, and slave synchronization mode. The UCREL bit of USCRC is used to select synchronization Or asynchronous mode. The U2X bit of USCRA controls the speed enable in asynchronous mode. The XCK pin is valid only in synchronous mode The data direction register (multiplexed with IO) determines whether the clock source is generated internally (host mode) or externally (slave mode formula).

### Baud rate generator

The baud rate register UBRR and the down-count counter are connected together as a programmable prescaler or baud rate for the USART Generator. The descending counter operates under the system clock ($f_{sys}$) and is automatically incremented when its count is zero or the UBRRL register is written The value of the UBRR register is set. When the count to zero to produce a clock, the clock as the baud rate generator output clock, The frequency is $f_{sys}$ / (UBRR + 1).

The following table shows the formulas for calculating baud rates (bits per second) and UBRR values in various operating 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 |
| Synchronous host mode | BAUD = $f_{sys}$ / (2 * (UBRR + 1)) | UBRR = $f_{sys}$ / (2 * BAUD) - 1 |

Description:

1. The baud rate is defined as the bit transfer rate (bps) per second;
2. BUAD is the baud rate, $f_{sys}$ is the system clock, and UBRR is the combination value of the baud rate register UBRRH and UBRRL.

### Speed mode of operation

By setting the U2RA bit in the UCSRA register, it is possible to double the transfer rate, which is valid only in asynchronous mode Set this bit to "0" in step mode.
Setting this bit will halve the division of the baud rate divider, effectively doubling the transfer rate of asynchronous communication. In this case , The receiver uses only half the number of samples to sample and clock the data, requiring a more accurate baud rate Set and system clock. The transmitter does not change.

### External clock

The synchronous slave operating mode is driven by an external clock. The external clock passes through the sync register and the edge detector before being sent by the transmitter

And the receiver is used, this process will introduce the delay of the two system clocks, so the maximum external clock frequency of the external XCK is given by Formula Restrictions:

$$f_{XCK} < f_{sys} / 4$$

Note that fsys has a system clock stability decision, in order to prevent data loss due to frequency drift, it is recommended to keep enough Margin.

### Synchronous clock operation

In synchronous mode, the XCK pin is used for clock input (slave mode) or clock output (master mode). Clock edge with The basic rules governing the relationship between data sampling and data variation are the clock edge used for data input (RxD) sampling and data The clock edge used by the output changes is reversed.

UCPOL = 1

UCPOL = 0

XCK timing in synchronous mode

As shown in the figure above, when the UCPOL value is "1", the data output is changed on the falling edge of XCK, and the number of XCK rising edges According to the sampling; when the UCPOL value is "0", the rising edge of the XCK changes the data output, the XCK on the falling edge of the data mining kind.

### Frame format

A serial data frame consists of a data word plus a sync bit (start bit and stop bit) and a parity bit 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 at the start bit, followed by the least significant bit of the data word, followed by the other data bits, with the highest bit of the data word Beam, the most successful transmission of 9-bit data. If parity is enabled, the parity bit will be followed by the data word, and finally the stop bit. when After a complete data frame is transmitted, the next new data frame can be transmitted immediately or the transmission line is idle (high Flat state). The following figure shows the possible data frame structure, and the bits in square brackets are optional.



USART frame structure

Description:
1) There is no data transmission on the IDLE communication line (RxD or TxD) and must be high when the line is idle
2) St start bit, always low
3) 0-8 data bits
4) P parity, odd parity or even parity
5) Sp stop bit, always high
The structure of the data frame is set by UCSZB and UCSZ [2: 0], UPM [1: 0] and USBS in the UCSRB register. Receive and send

Use the same settings. Any changes made may destroy the ongoing data transfer. Among them, UCSZ [2: 0] is determined
The number of data bits of the data frame, UPM [1: 0] is used to enable and determine the type of parity, the USBS setting frame has one or two digits
Bit. The receiver will ignore the second stop bit, so the frame error is detected only when the first end bit is "0".

### Check bit calculation

The parity bit is calculated by XORing the bits of the data. If odd parity is selected, the XOR result needs to be reversed.
The relationship between the parity bit and the data bits is as follows:

$$P_{\text{the even}} = D_{n-.1} \oplus \ldots \oplus d_{.3} \oplus d_2 \oplus d_{.1} \oplus d_0 \oplus 0$$

$$P_{ODD} = D_{n-.1} \oplus \ldots \oplus d_{.3} \oplus d_2 \oplus d_{.1} \oplus d_0 \oplus 1$$

Description:
1) $P_{even}$ Even check results
2) $P_{odd}$ odd parity results
3) $d_n$ nth data bits

### The USART is initialized

The USART should be initialized before communication. The initialization process usually includes the baud rate setting, the frame structure
Set, and enable receivers or transmitters as needed. For interrupt-driven USART operations, clear the initialization
The global interrupt flag disables all interrupts of the USART.
When performing a re-initialization such as changing the baud rate or frame structure, it is necessary to ensure that no data is transmitted. TXC flag can be used
To detect whether the transmitter has completed all transmissions, the RXC flag can be used to detect whether there is data in the receive buffer
Read out. If the TXC flag is used for this purpose, it must be cleared before each data is sent (before writing the UDR register)
Zero TXC flag.

### Transmitter

Setting the TXEN bit of the UCSRB register will enable data transmission for the USART. Enable the general IO function of the TxD pin
Replaced by the USART function, becomes the serial output of the transmitter. Send the data before setting the baud rate, working mode and
Frame format. If the synchronous transmit mode is used, the clock signal applied to the XCK pin is the clock for data transmission.

### Send 5 to 8 frames for data

Will need to send the data loaded into the send buffer to start the data sent. The CPU loads the number by writing to the UDR register
according to. When the transmit 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 in the idle state (no ongoing data transfer), or the last stop of the previous frame data
When the bit is sent, it will load the new data. Once the shift register has loaded the new data, it will pass in the established settings





Lose a complete frame.

### A frame that sends 9-bit data

If the 9-bit data frame is sent, the ninth bit of the data should be written to the TXB8 bit in register UCSRB and then low
The bit data is written to the transmit data register UDR. The 9th bit data is used in the multi-machine communication to represent the address frame in synchronous communication
Can be used for protocol processing.

### Send parity bit

The parity generation circuit generates the corresponding parity bit for the serial data frame. When the parity bit is enabled (UPM1 = 1), send control
The logic circuit inserts the parity bit between the last bit of the data word and the first stop bit.

### Send flag and interrupt processing

The USART transmitter has two flags: the USART data register empty flag UDRE and the transmission end flag TXC, two
Ambitions can be interrupted.
The data register empty flag UDRE is used to indicate whether the transmit buffer can write a new data. The bit is in the send buffer

When the space is set to "1", the full time is set to "0". When the UDRE bit is "1", the CPU can write a new number to the data register UDR
According to, and vice versa can not.
When the data register empty interrupt enable bit UDRIE in the UCSRB register is "1", as long as the UDRE is set (and global
Interrupt enable), the USART data register will generate an empty interrupt request. Writing a write to register UDR will clear the UDRE.
When data is transmitted in an interrupt mode, a new data must be written to the UDR in the data register empty interrupt service routine
To clear the UDRE, or to disable the data register from being interrupted. Otherwise once the interrupt service routine ends, a new one
Break will be generated again.
When the entire data frame is shifted out of the transmit shift register and there is no new data in the transmit register, the transmit end flag
TXC will be set. When the transmit end interrupt enable bit, TXCIE (and global interrupt enable) on UCSRB, is set to "1"
The TXC flag is set and the USART transmit end interrupt will be executed. Once the interrupt service routine is entered, the TXC flag is the bit
Is cleared automatically, the CPU can also write "1" to this bit to clear.

### Disable the transmitter

When TXEN is cleared, only after all the data have been sent to send the transmitter can be really prohibited, that is, send shift send
There is no data to be transferred in both the register and the transmit buffer register. After the transmitter is disabled, the TxD pin resumes its general purpose IO work
can.

### receiver

Set the receive enable bit (RXEN) of the UCSRB register to start the USART receiver. Enabled after the RxD pin is generic
The IO function is replaced by the USART function as the serial input of the receiver. Be sure to set up the data before receiving it
Baud rate, operating mode and frame format. If the synchronous receive mode is used, the clock on the XCK pin is used as the transmit clock.

### A frame that receives 5 to 8 bits of data

Once the receiver has detected a valid start bit, it begins to accept the data. Each bit of data after the start bit will be set
Set the baud rate or XCK clock to receive until the first stop bit of a frame of data is received and the second stop bit is





Receiver ignored. Each bit of data received is sent to the receive shift register. After receiving the first stop bit, the receiver
Set the RXC bit in the receive data completion flag of the UCSRA register and transfer the complete data frame in the shift register
In the receive buffer, the CPU can obtain the received data by reading the UDR register.

### A frame that receives 9-bit data

If a 9-bit data frame is set, the register UCSRB must first be read before reading the lower 8-bit data from the UDR
RXB8 bits to get the 9th bit of data. This rule also applies to the status flags FE, DOR, and PE. Read the UDR
The memory cell changes the state of the receive buffer and changes the TXB8, FE, DOR, and PE that are also stored in the buffer
Bit.

### Receive end flag and interrupt handling

The USART receiver has a flag: the receive end flag RXC, which indicates whether the receive buffer has an unread
data. This bit is "1" and "0" when the receive buffer has data that is not read. If the receiver is disabled,
The receive buffer is refreshed and RXC is cleared.
Set the receive enable interrupt enable bit RXCIE after UCSRB, as long as the RXC flag is set (and the global interrupt is enabled)
The USART reception end interrupt is generated. When data is received using the interrupt mode, the data reception ends the interrupt service routine
The sequence must read the data from the UDR to clear the RXC flag, otherwise a new interrupt will be completed as soon as the interrupt handler ends
Will produce.

### Receive error flag

The USART receiver has three error flags: frame error FE, data overflow DOR, and parity error PE. They are all located
UCSRA register. The error flag is stored in the receive buffer along with the data frame. All the wrong signs can not be produced
Interrupted.
The frame error flag FE indicates the state of the first stop bit of the next readable frame stored in the receive buffer. Stop bit is positive

(The value is "1"), the FE flag is "0", otherwise the FE flag is "1". This flag can be used to detect synchronization loss, transmission
Broken, can also be used for protocol processing.
The data overflow flag DOR indicates that data loss is caused by the receive buffer. When the receive buffer is full, receive the shift
The data already exists in the register, and if a new start bit is detected at this time, a data overflow is generated. The DOR flag is set to that
Indicating that one or more data frames have been lost between the most recent read UDR and the next read UDR. When the data frame is successful
The DOR flag is cleared after shifting from the shift register to the receive buffer.
The parity error flag PE indicates that the next frame of data in the receive buffer has a parity error at reception. If no parity is enabled
Check, PE is cleared.

### Parity checker

Setting the parity mode bit UPM1 will start the parity checker. The calibration mode (even parity or odd parity) is determined by UPM0
set. After the parity is enabled, the validator will calculate the parity of the input data and compare the result to the parity bit of the data frame.
The check result will be stored in the receive buffer along with the data and stop bits. The CPU checks the received frame by reading the PE bit
Whether there are parity errors. If the next data read from the receive buffer has a parity error, and the parity
Yes, the UPE is set and is always valid until the receive buffer UDR is read.





### Disable the receiver

It is forbidden for the receiver to function immediately compared to the transmitter. The data being received will be lost. Disable receiver (RXEN clear)
, The receiver will no longer occupy the RxD pin and the receive buffer will be refreshed.

### Asynchronous data reception

The USART has a clock recovery unit and a data recovery unit to handle asynchronous data reception. The clock recovery logic is used to synchronize from
The RxD pin inputs the asynchronous serial data and the internal baud rate clock. Data recovery logic is used to collect data and pass low
The filter filters the input of each bit of data, thereby improving the receiver's anti-jamming performance. Asynchronous reception by the scope of work
Depending on the accuracy of the internal baud rate clock, the rate of the frame input, and the number of data bits contained in one frame.

### Asynchronous working range

The operating range of the receiver depends on the degree of mismatch between the received data rate and the internal baud rate. If the sender to
Too fast or too slow bit rate to transmit data, or the baud rate generated within the receiver does not have the same frequency, then the receiver
It can not be synchronized with the start bit. To ensure that the receiver does not miss the sampling of the next frame start bit, the data input rate and the internal
The baud rate of the receiver can not be too large, and the ratio between them is used to describe the baud rate error range. The following two tables are divided
Do not give the maximum permissible baud rate error range in normal mode and double speed mode.

The maximum receiver baud rate error range in normal mode

| Data bit + parity bit length and | Maximum error range (%) | Recommended error range (%) |
|---|---|---|
| 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 bit + parity bit length and | Maximum error range (%) | Recommended error range (%) |
|---|---|---|
| 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 |

As can be seen from the table, the baud rate in normal mode allows for a greater range of variation. The recommended baud rate error range is above
Assume that the receiver and the transmitter have the same contribution to the maximum total error. Generates the receiver baud rate error
There may be two reasons. First, the stability of the receiver system clock is related to the operating voltage and temperature. Using crystal to produce
System clocks generally do not have this problem, but when using the internal oscillator, the system clock may be biased. The second reason
The baud rate generator does not necessarily get the desired baud rate by dividing the system clock. At this time can be adjusted
The value of UBRR makes the error as low as acceptable.





### Baud rate setting and introduction error

For standard crystal and resonator frequencies, the baud rate for actual communication in asynchronous mode can be calculated by the baud rate calculation formula
The error between it and the commonly used baud rate can be calculated using the following formula:

$$Error [\%] = (Baud_{real} / Baud - 1) * 100\%$$

Among them, Baud is the commonly used communication baud rate, $Baud_{real}$ is calculated by calculating the baud rate, into the baud rate meter
Calculate the relationship between the baud rate error and the system clock $f_{sys}$ and the baud rate register UBRR value as follows:
Normal mode:

$$Error [\%] = (f_{sys} / (16 * (UBRR + 1)) / Baud - 1) * 100\%$$

Speed mode:

$$Error [\%] = (f_{sys} / (8 * (UBRR + 1)) / Baud - 1) * 100\%$$

The baud rate error UBRR is obtained when the clock error on both sides of the communication is not considered, ie the system clock $f_{sys}$ is the standard clock
The relationship between the values. The following table shows the baud rate error for different UBRR values under the 16MHz system clock.

16MHz system clock to set the UBRR value generated by the error

$f_{sys}$ = 16.000MHz

| Baud rate (bps) | Normal mode (U2X = 0) | | 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

The Multiprocessor Communication Mode (MPCM) bits of the UCSRA are set to filter the data frames received by the USART receiver.
Frames with no address information will be ignored and will not be stored in the receive buffer. In a multiprocessor system, each processing
The devices communicate via the same serial bus, which effectively reduces the number of data frames that require CPU processing.
MPCM bit settings do not affect the work of the transmitter, but in the multi-processor communication system, its use will be
with.

If the receiver receives a data frame length of 5 to 8 bits, the first stop bit is used to indicate that the current frame contains
Is data or address information. If the length of the data frame received by the receiver is 9 bits, then the number is determined by the 9th bit
According to the address information. If the frame type flag is "1", then this is the address frame, otherwise it is a data frame.

In multiprocessor communication mode, multiple slave processors are allowed to receive data from a host processor. First by decoding the address Frame to determine which address is addressed from the processor. The addressed slave processor will normally receive subsequent data while others The slave will ignore these data frames until the next address frame is received.

For a processor as a host, it can use the 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 operate in a 9-bit data frame format.
The following is the multi-processor communication mode for data exchange steps:

1. All slave processors are operating in multiprocessor communication mode (set MPCM);
2. The host processor sends an address frame, and all slave processors receive this frame. From the RXC bit of the processor UCSRA register Normal set;
3. Each slave processor reads the contents of the UDR register and decodes the address frame to determine if it is selected. If checked, The MPCM bit in the UCSRA register is cleared and the MPCM is set to "1" and the next address is waited The arrival of the frame;
4. The addressed slave processor receives all the data frames until a new address frame is received. Not addressed from the office The processor ignores these data frames;
5. After receiving the last data frame from the processor, the MPCM bit is set and the next address frame is set arrival. And then repeat from the second step.

The frame format using 5 to 8 bits of data is possible, but it is impractical because the receiver must use n and n + 1 frames Format to switch between. Since the receiver and transmitter use the same character length setting, this setting allows full duplex operation It becomes very difficult. If you use frame format of 5 to 8 bits of data, the transmitter should set two stop bits, where the first The stop bit is used to determine the frame type.

## Register definition

### UCSRA - USART Control and Status Register A

UCSRA - USART Control and Status Register A

Address: 0xC0 Default: 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.<br>When the value of RXC is "1", it indicates that there is unread data in the receive buffer. When the value of RXC When "0", it indicates that there is no unread data in the receive buffer. Receiver disabled when receiving The buffer is refreshed, causing RXC to be cleared. When the receive end interrupt enable bit, RXCIE, is "1" , RXC can be used to generate a receive end interrupt. |
| 6 | TXC | Send the end flag.<br>TXC is set when the data in the transmit shift register is sent and the transmit buffer is empty. Executive The TXC is automatically cleared when the transmit end is interrupted, or it can be cleared by writing "1" to TXC. TXC can be used to generate a transmit end interrupt when the transmit end interrupt enable bit, TXCIE, is "1". |

| | | |
|---|---|---|
| | | Data register empty flag. |
| | | When UDRE is "1", it indicates that the USART send data buffer is empty and can write data. |
| 5 | UDRE | When the UDRE is "0", it indicates that the USART send data buffer is full and can not write data. |
| | | When the data register empty interrupt enable bit UDRIE is "1", UDRE can be used to generate data |
| | | The memory is empty. |
| | | Frame error flag. |
| | | When FE is "1", it indicates that the data received by the receive data buffer has a frame error, that is, the first |
| 4 | FE | The stop bit is "0". When FE is "0", it indicates that the data received by the receive data buffer is not available |
| | | Frame error, ie the first stop bit is "1". FE is set to valid until UDR is read |
| | | take. When writing to UCSRA, FE will write "0". |
| | | Data overflow flag. |
| 3 | DOR | When the receive buffer is full (contains two data), there is data in the receive shift register. |
| | | At this point a new start bit is detected, a data overflow occurs, DOR is set, and is always active |
| | | UDR is read. When writing to UCSRA, DOR will write "0". |
| 2 | PE | Parity error flag. |
| | | When the parity is enabled (UPM1 is "1"), and the received data frame in the receive buffer |
| | | There is a parity error, PE is set, has been valid until the UDR is read. On UCSRA |
| | | Write, PE this bit to write "0". |
| 1 | U2X | speed transmit enable bit. |
| | | When U2X is "1", the transmission rate of the asynchronous communication mode is doubled. When U2X is "0", different |
| | | The transmission rate of the step communication mode is the normal rate. |
| | | This bit is valid only in asynchronous operation mode and is cleared when the synchronous operation mode is used. |
| 0 | MPCM | multiprocessor communication mode enable bit. |
| | | Setting the MPCM bit will start the multiprocessor communication mode. MPCM is set after the USART receiver |
| | | Received input frames that do not contain address information will be ignored. The transmitter is not protected by MPCM |
| | | Set the impact. |

### UCSRB - USART Control and Status Register B

UCSRB - USART Control and Status Register B

Address: 0xC1 Default: 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 |
|---|---|---|
| | | Receive end interrupt enable bit. |
| 7 | RXCIE | Set RXC interrupt and disable RXC interrupt after clearing. When RXCIE is "1", the global interrupt |
| | | Enable, the USART receive end interrupt can be generated when the RXC of the UCSRA register is "1". |
| | | The transmit end interrupt enable bit. |
| 6 | TXCIE | Enable TXC interrupt after setting, disable TXC interrupt after clearing. When TXCIE is "1", the global interrupt |
| | | Enable, and the TXC of the UCSRA register is "1" to generate a USART transmit end interrupt. |
| | | Data register empty interrupt enable bit. |
| 5 | UDRIE | Set to enable UDRE interrupt after clearing, disable UDRE interrupt after clearing. When UDRIE is "1", the global |
| | | The enable is enabled and the UDART of the UCSRA register is "1" to generate the USART data register empty |



| | | |
|---|---|---|
| | | Interrupted. |
| | | Receive enable bit. |
| 4 | RXEN | Set the USART receiver after setting. The general purpose IO function of the RxD pin is received by the USART |
| | | generation. Disabling the receiver will refresh the receive buffer and invalidate the FE, DOR, and PE flags. |
| | | Send enable bit. |
| 3 | TXEN | Set the USART transmitter after setting. The general IO function of the TxD pin is sent by the USART |
| | | generation. After TXEN is cleared, it will only be possible to wait until all data has been sent |
| | | USART sent. |

| | |
|---|---|
| | Character length control bit 2. |
| 2 UCSZ2 | UCSZ2 is combined with UCSZ1: 0 of the UCSRC register to set the data contained in the data frame<br>The number of bits. |
| 1 RXB8 | Receive data bit 8.<br>When the data frame length is 9 bits, RXB8 is the most significant bit of the received data. Read the included UDR<br>Of the low 8-bit data before the first to read RXB8. |
| 0 TXB8 | Send data bit 8.<br>When the data frame length is 9 bits, TXB8 is the most significant bit of the transmitted data. Write included in the UDR<br>Of the low 8-bit data before the first write TXB8. |

UCSRC - USART Control and Status Register C

UCSRC - USART Control and Status Register C

Address: 0xC2 Default: 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 selection bit.<br>UMSEL selects synchronous or asynchronous operation mode.<br>UMSEL mode<br>0 USART Asynchronous operation mode<br>1 USART Synchronous mode of operation<br>2 SPI slave operating mode<br>3 SPI master operating mode |
| 5: 4 | UPM1: 0 | Parity mode selection bit.<br>High UPM1 Select to enable or disable parity, low order UPM0 Select odd parity or even check.<br>UPM1: 0 mode<br>0 Disable parity<br>1 Keep it<br>2 Enable even parity<br>3 Enable odd parity |
| 3 | USBS | Stop bit select bit. Select the number of bits for the stop bit.<br>USBS Stop bit number<br>0 1 |





| | | |
|---|---|---|
| | | 1 2 |
| 2: 1 | UCSZ1: 0 | Data frame character length selection bit.<br>UCSZ1: 0 is combined with UCSZ2 of the UCSRB register to set the data contained in the data frame<br>The number of bits.<br>UCSZ2: 0 Data frame length<br>0 5 digits<br>1 6 digits<br>2 7 digits<br>3 8 bits<br>4 Keep it<br>5 Keep it<br>6 Keep it<br>7 9 digits |
| | | Clock polarity select bits.<br>In the USART synchronous operation mode, UCPOL sets the change and input of the output data |

0 UCPOL The relationship between the sampling of the data and the synchronous clock XCK. Use asynchronous work mode with UCPOL has nothing to do with this bit

| UCPOL | Send data changes | Receive data sampling |
|---|---|---|
| 0 | XCK rising edge | The falling edge of XCK |
| 1 | The falling edge of XCK | XCK rising edge |

### UBRRL - USART baud rate register low byte

UBRRL - USART baud rate register low byte

Address: 0xC4 Default: 0x00

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | UBRR7 | UBRR6 | UBRR5 | UBRR3 | 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] | The low byte portion of the USART baud rate register.<br>The USART baud rate register contains UBRRL and UBRRH, which are used together<br>To set the baud rate for communication. |

### UBRRH - USART baud rate register high byte

UBRRH - USART baud rate register high byte

Address: 0xC5 Default: 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 | - | Keep it. |





| | |
|---|---|
| 3: 0 | UBRR [11: 8] |

The high byte portion of the USART baud rate register.
The USART baud rate register contains UBRRL and UBRRH, which are combined
Used to set the baud rate for communication.
UBRR = {UBRR [11: 8], UBRRL}

| Operating mode | Baud rate calculation formula |
|---|---|
| Asynchronous normal mode | BAUD = $f_{sys}$ / (16 * (UBRR + 1)) |
| Asynchronous speed mode | BAUD = $f_{sys}$ / (8 * (UBRR + 1)) |
| Synchronous host mode | BAUD = $f_{sys}$ / (2 * (UBRR + 1)) |

### UDR - USART data register

UDR - USART data register

Address: 0xC6 Default: 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 | USART sends and receives data.<br>The USART transmit data buffer and the receive data buffer share the USART data register UDR.<br>Write data to UDR write data buffer, read data from UDR read receipt<br>Data buffer.<br>In the 5 to 8-bit data frame mode, the unused 9th bit is ignored by the transmitter, and the receiver<br>Set them to 0.<br>The transmit buffer can only be written when the UDRE flag of the UCSRA register is "1"<br>, Otherwise the operation of the transmitter will be wrong. When the transmit shift register is empty, the transmitter will put |

The data in the transmit buffer is loaded into the transmit shift register, and the data is serially derived from the TxD Pin output.

The receive buffer contains a two-stage FIFO that will change once the receive buffer is read

Change its state.





## USART0 - SPI operating mode

- Full-duplex operation, three-wire synchronous data transmission
- Host or slave operation
- Supports all four operating modes (modes 0, 1, 2 and 3)
- Low or high first transmission (configurable data transfer order)
- Queue operation (double buffer)

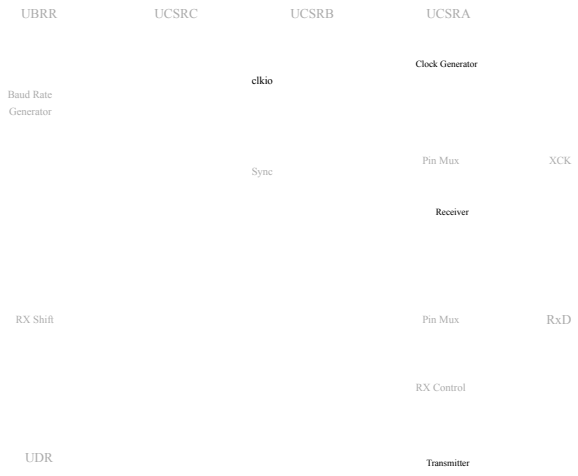
High resolution baud rate generator

### Summary

When the UCREL1 bit of the USCRC is set to "1", the SPI operating mode is enabled and is represented by USPI. This SPI module is three Line SPI operating mode, compared with the four-wire SPI mode, the lack of slave select line, the other three lines are consistent. USPI occupies USART resources, including sending and receiving shift registers and buffers, and baud rate generators. Parity generation

And check logic, data and clock recovery logic are invalid. The address of the control and status registers is the same, but is registered

The function of the bit will change with the need for the SPI operating mode.




USART in SPI structure

### Clock generation

When the SPI is in host mode, it is necessary to provide a clock for communication, multiplexing the baud rate generator of the USART to generate this
A clock. The clock is output from the XCK pin, so the XCK pin's data direction register (DDR_XCK) must be set to





"1".

The clock frequency is determined by the following formula:

BAUD = $f_{sys}$ / (2 * (UBRR + 1))

When the SPI is operating in Slave mode, the communication clock is provided by the external host, input from the XCK pin, so the XCK pin
The data direction register (DDR_XCK) must be set to "0".

### SPI data mode and timing

SPI has four clock phase and polarity combination, there are control bits UCPHA and UCPOL to determine the specific control such as
The following table and the following figure:

SPI operating mode

| SPI mode | UCPOL | UCPHA | Starting along | End the edge |
|---|---|---|---|---|
| 0 | 0 | 0 | Rising edge sampling | Falling edge setting |
| 1 | 0 | 1 | Rising edge setting | Down edge sampling |
| 2 | 1 | 0 | Down edge sampling | Rising edge setting |
| 3 | 1 | 1 | Falling edge setting | Rising edge sampling |

SPI operating mode icon

### Frame format

A serial frame of the SPI can start with the least significant bit or the most significant bit, ending with the most significant bit or the lowest bit, for a total of 8 bits of data. one
After the end of the frame, you can immediately transfer the new frame, the end of the transmission can be pulled up the data line is idle.

### data transmission

SPI sets the TXEN bit of the UCSRB register to "1" to enable the transmitter. The TxD pin is occupied by the transmitter to send the serial output
data. The receiver can not be enabled at this time.
SPI sets the RXEN bit in the UCSRB register to "1" to enable the receiver. The RxD pin is occupied by the receiver to receive the serial input
data. The transmitter must be enabled at this time.
Both SPI transmit and receive uses XCK as the transfer clock.

The SPI is first initialized before communication is performed. The initialization process usually includes the setting of the baud rate, the frame data bit

The setting of the transmission sequence, and the reception of the receiver or transmitter as needed. For interrupt-driven SPI operations, the initialization is done







The global interrupt flag is cleared and all interrupts of the SPI are disabled.
When performing a re-initialization such as changing the baud rate or frame structure, it is necessary to ensure that no data is transmitted. TXC flag can be
Used to detect whether the transmitter has completed all transmissions, the RXC flag can be used to detect whether there are still numbers in the receive buffer
It is not read. If the TXC flag is used for this purpose, before each data is sent (before writing the UDR register)
The TXC flag must be cleared.

After initializing the SPI, write data to the UDR register to start the data transfer. Since the transmitter controls the transmission clock,
Send and receive data are doing so. When sending a shift register ready to send a new frame of data, the transmitter is on
The data written to the UDR register is moved from the transmit buffer to the transmit shift register and sent out. to ensure that
The input buffer and the transmit data are synchronized, and the UDR register must be read once every byte of data has been transmitted. When made
When data is spilled, the most recently received data will be lost, not the earliest received data.

### Send flag and interrupt

The SPI transmitter has two flags: the SPI data register empty flag UDRE and the transmission end flag TXC, both flag bits
Can generate an interrupt.

The data register empty flag UDRE is used to indicate whether the transmit buffer can write a new data. The bit is sending slow
When the punch is empty, it is set to "1" and "0" when it is full. When the UDRE bit is "1", the CPU can write new to the data register UDR
Of the data, otherwise it can not.

When the data register empty interrupt enable bit UDRIE in the UCSRB register is "1", as long as the UDRE is set (and global
Interrupt enable), it will generate an SPI data register empty interrupt request. Writing to the register UDR will be cleared
UDRE. When transmitting data in an interrupt mode, a new number must be written in the data register empty interrupt service routine
To the UDR to clear the UDRE, or to disable the data register from being interrupted. Otherwise, once the interrupt service routine ends,
A new interrupt will be generated again.

When the entire data frame is shifted out of the transmit shift register and there is no new data in the transmit register, the transmit end flag
TXC will be set. When the transmit end interrupt enable bit, TXCIE (and global interrupt enable) on UCSRB, is set to "1"
As the TXC flag is set, the SPI transmit end interrupt will be executed. Once the interrupt service routine is entered, the TXC flag bit
It is automatically cleared, the CPU can also write "1" to the bit to clear.

### Disable the transmitter

When TXEN is cleared, only after all the data has been sent to send the transmitter can be really prohibited, that is, send the shift
There is no data to be transferred in the register and transmit buffer registers. After the transmitter is disabled, the TxD pin resumes its generic
IO function.

### Receive end flag and interrupt

The SPI receiver has a flag: the receive end flag RXC, which indicates whether the receive buffer has an unread number
according to. This bit is "1" and "0" when the receive buffer has data that is not read. If the receiver is disabled,
The receive buffer is refreshed and RXC is cleared. Set the UCSRB to the receive end interrupt enable bit after RXCIE only
To set the RXC flag (and the global interrupt is enabled), an SPI receive end interrupt is generated. Use the interrupt mode
When data is received, the data reception end interrupt service routine must read data from the UDR to clear the RXC flag, otherwise

When the interrupt handler ends, a new interrupt is generated.

### Disable the receiver

It is forbidden for the receiver to function immediately compared to the transmitter. The data being received will be lost. Disable the receiver (RXEN clear Zero), the receiver will no longer occupy the RxD pin and the receive buffer will be refreshed.

### Register definition

USART register list

| register | address | Defaults | description |
|---|---|---|---|
| UCSRA | 0xC0 | 0x20 | USPI Control and Status Register |
| 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

UCSRA - USPI Control and Status Register

Address: 0xC0 Default: 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.<br>When the value of RXC is "1", it indicates that there is unread data in the receive buffer. When the value of RXC<br>When "0", it indicates that there is no unread data in the receive buffer. Receiver disabled when receiving<br>The buffer is refreshed, causing RXC to be cleared. When the receive end interrupt enable bit, RXCIE, is "1"<br>, RXC can be used to generate a receive end interrupt. |
| 6 | TXC | Send the end flag.<br>TXC is set when the data in the transmit shift register is sent and the transmit buffer is empty. carried out<br>TXC is automatically cleared when TX is asserted, or cleared by writing "1" to TXC.<br>TXC can be used to generate a transmit end interrupt when the transmit end interrupt enable bit, TXCIE, is "1". |
| 5 | UDRE | Data register empty flag.<br>When UDRE is "1", it indicates that the USPI send data buffer is empty and can write data. when<br>When UDRE is "0", it indicates that the USPI sends the data buffer to full and can not write data. When the number<br>The UDRE can be used to generate a data register empty when the register empty interrupt enable bit UDRIE is "1"<br>Interrupted. |
| 4: 0 | - | Retained under USPI. |



#### UCSRB - USPI Control and Status Register B

UCSRB - USPI Control and Status Register B

Address: 0xC1 Default: 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.<br>Set RXC interrupt and disable RXC interrupt after clearing. When RXCIE is "1", globally<br>When the RXC of the UCSRA register is "1", the USPI receive end interrupt can be generated. |
| 6 | TXCIE | The transmit end interrupt enable bit.<br>Enable TXC interrupt after setting, disable TXC interrupt after clearing. When TXCIE is "1", the global<br>When the TXC of the UCSRA register is "1", the USPI transmission end interrupt can be generated. |
| 5 | UDRIE | Data register empty interrupt enable bit.<br>Set to enable UDRE interrupt after clearing, disable UDRE interrupt after clearing. When UDRIE is "1", the global<br>Interrupt enable, UFLRA register UDRE is "1" can generate USPI data register empty<br>Interrupted. |
| 4 | RXEN | Receive enable bit.<br>Set the USPI receiver after setting. The general IO function of the RxD pin is replaced by the USPI reception.<br>Disabling the receiver will refresh the receive buffer. |
| 3 | TXEN | Send enable bit.<br>Set the USPI transmitter after setting. The general IO function of the TxD pin is replaced by the USPI transmission.<br>After TXEN is cleared, the USART can only be disabled until all data has been sent<br>send. |
| 2: 0 | - | Retained under USPI. |

### UCSRC - USART Control and Status Register C

UCSRC - USART Control and Status Register C

Address: 0xC2 Default: 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 selection bit.<br>UMSEL selects synchronous or asynchronous operation mode.<br>UMSEL: mode<br>0: USART Asynchronous operation mode<br>1: USART Synchronous mode of operation<br>2: SPI slave operating mode<br>3: SPI master operating mode |
| 5: 3 | - | Retained under USPI. |





| Bit | Name | Description |
|---|---|---|
| 2 | DORD | Data transfer order selection bit.<br>DORD: Data order<br>0: High first transmission<br>1: Low first transmission |
| 1 | UCPHA | Clock phase selection.<br>UCPHA Selects data sampling at the beginning or end edge.<br>UCPHA: Sampling time<br>0: Starting along<br>1: End the edge |
| 0 | UCPOL | Clock polarity selection.<br>UCPOL selects data changes and samples occur on rising or falling edges.<br>UCPOL: Send data changes: Sampling of received data |

| | | |
|---|---|---|
| 0 | XCK rising edge | The falling edge of XCK |
| 1 | The falling edge of XCK | XCK rising edge |

### UBRRL - USPI baud rate register low byte

UBRRL - USPI baud rate register low byte

Address: 0xC4 Default: 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] | USPI The low byte portion of the baud rate register. The USPI baud rate register contains UBRRL And UBRRH two parts, combined with the baud rate used to set the communication. |

### UBRRH - USPI baud rate register high byte

The UBRRH - high byte register in USP baud

Address: 0xC5 Default: 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 | - | Retained under USPI. |
| 3: 0 | UBRR [11: 8] | The high byte portion of the USPI baud rate register. USPI baud rate register contains UBRRL and UBRRH two parts, combined with Set the baud rate of the communication. UBRR = {UBRR [11: 8], UBRRL} |





| Operating mode | Baud rate calculation formula |
|---|---|
| Slave mode | The baud rate is determined by the external host |
| Host mode | BAUD = $f_{sys}$ / (2 * (UBRR + 1)) |

### UDR - USPI data register

UDR - USPI data register

Address: 0xC6 Default: 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 | USPI sends and receives data. The USPI sends the data buffer and the receive data buffer to share the USPI data register UDR. Write data to the UDR write data buffer, read data from the UDR read access Receive data buffer. In the 5 to 8-bit data frame mode, the unused 9th bit is ignored by the transmitter, and the receiver Set them to 0. The transmit buffer can only be written if the UDRE flag of the UCSRA register is "1" Operation, otherwise the operation of the transmitter will be wrong. When sending the shift register is empty, the transmitter The data in the transmit buffer is loaded into the transmit shift register, and the data is serially Output from the TxD pin. The receive buffer contains a two-stage FIFO that will change once the receive buffer is read Change its state. |

# TWI - Dual Line Serial Bus (I2C)

- Simple and powerful and flexible communication interface, only 2 lines
- Supports host and slave operation
- The device can operate in transmitter mode or receiver mode
- There are 128 slaves in the 7-bit address space
- Multi-host arbitration is supported
- Up to 400Kbps data transfer rate
- Fully programmable slave address and public address
- You can wake up when the address matches in sleep mode

## TWI bus introduction

Two-wire serial interface TWI is well suited for typical processor applications. The TWI protocol allows the system designer to use only two bi-directional Of the transmission line can be 128 different devices connected together. The two lines are the clock SCL and the data SDA. External hard Only need to pick up two pull-up resistors on each line. All devices connected to the bus have their own addresses. TWI protocol Solve the problem of bus arbitration.

## TWI terms

The terms defined below will appear frequently in this section.

| the term | description |
|---|---|
| Host | Start and stop the transmission of the device. The host is also responsible for generating the SCL clock. |
| Slave machine | The device that is addressed by the host |
| Transmitter | Place the data on the bus |
| receiver | A device that receives data from the bus |

## Electrical connections

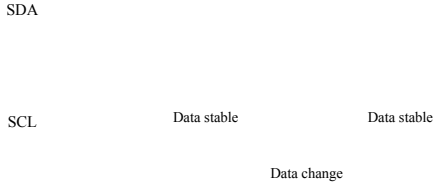
As shown in the following figure, the two lines of the TWI interface are connected to the positive supply via the pull-up resistor. The total of all TWI-compatible devices Line drivers are open-drain or open-collector, so that the interface to achieve the operation of the line and function. When TWI devices When the output is "0", the TWI bus will be low. When all the TWI devices are tri-stated, the bus is allowed to pull up Resistance will pull high voltage. To ensure that all bus operations, all devices connected to the TWI bus must be powered on.


TWI bus interconnection diagram



### Data transmission and frame structure

Each bit of data transfer on the TWI bus is synchronized with the clock. When the clock line is high, the level on the data line must be Remain stable unless it is to generate a start or stop state.




TWI data validity graph

### Start and stop status

TWI's transmission is initiated and stopped by the host. The host on the bus to send a START state to send data transmission, issued STOP status to stop data transfer. Between the START and STOP states, the bus is considered busy and does not allow the other The host tries to take control of the bus. There is a special case where only one occurs between the START and STOP states A new START state, which is called the REPEATED START state, applies to the current host without giving up bus control To start a new transmission. After the REPEATED START until the next STOP, the bus is still considered busy of. This is consistent with START, so in this document, if there is no special instructions, are used to express START And REPEATED START. As shown in the following figure, the START and STOP conditions change the power of the SDA line when the SCL line is high Flat state.


START, REPEATED START and STOP status diagrams

### Address packet format

All address packets transmitted on the TWI bus are 9-bit data length, with 7-bit address, 1-bit READ / WRITE control bit And a 1-bit response bit. When the READ / WRITE bit is "1", a read operation is performed; when the READ / WRITE bit is "0" Perform a write operation. After the slave is addressed, it must acknowledge in the 9th SCL (ACK) cycle by pulling down the SDA line. If If the slave is busy or there are other reasons that can not respond to the host, the SDA line should remain high for the ACK cycle. Then the host can be made

The STOP status or REPEATED START status is restarted.





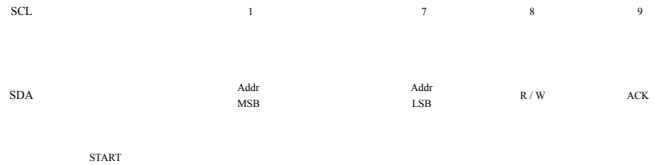


The address pack includes a slave address and a read or write control bit, denoted by SLA + R or SLA + W, respectively.

The MSB bit of the address byte first occurs. Except that the reserved address "00000000" is reserved for broadcast calls as well as all shapes
The address of the "1111xxxx" format needs to be reserved for future use. Other slave addresses can be freely assigned by the designer.

When a broadcast call occurs, all slaves should respond at the ACK cycle by pulling down the SDA line. When the host needs
The broadcast function can be used when sending the same information to multiple slaves. When the broadcast call address plus the WRITE bit is sent to
After the bus, all the slaves that need to respond to the broadcast call will pull down the SDA line during the ACK cycle. All of these responded
The slave of the broadcast call will receive the following packet. It is important to note that sending the broadcast call address plus the READ bit is
It does not make sense because if several slaves simultaneously send different data will bring bus conflicts.

The address package format is shown below:

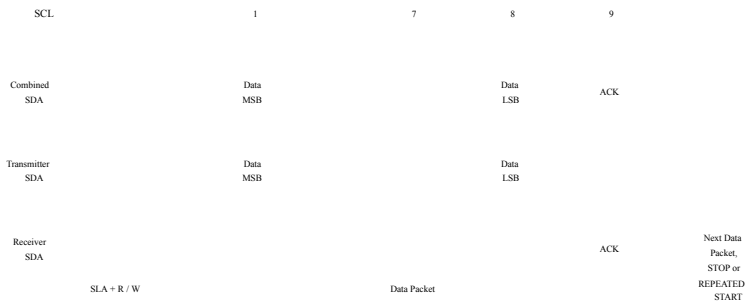



TWI address package format

## Packet format

All data packets transmitted on the TWI bus are 9-bit data length consisting of 1 data byte and 1 bit acknowledge bit. in
During data transmission, the host is responsible for generating the transfer clock SCL and START and STOP status, the transmitter sends the word to be transmitted
Section data, the receiver generates a receive response. The acknowledgment signal ACK is the receiver in the 9th SCL (ACK) cycle by pulling down
SDA line to produce. If the receiver holds the SDA line high during the ACK cycle, an unacknowledged signal NACK is issued.
When the receiver has received the last byte, or for some reason can no longer receive any data, it should be in the collection
To the last byte by sending a NACK to inform the sender. The MSB bits of the data byte are transmitted first.

The packet format is shown in the following figure:

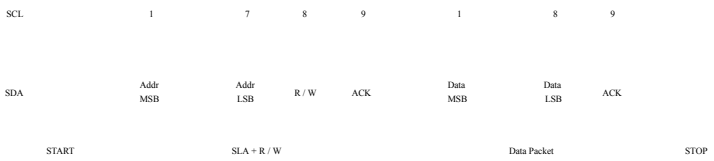



TWI packet format

Combined address and packet transmission, a transmission is basically a START, 1 SLA + R / W, one or more

Data packets and a STOP. Only START and STOP empty messages are illegal. You can use the line with the SCL line
Function to achieve the host and slave shake hands. The slave can extend the SCL level by pulling down the SCL line. When the host
This feature is useful when setting the clock speed much faster than the slave, or if the slave requires additional time to process the data.
The slave's extended SCL low period does not affect the high period of SCL, which is still determined by the host. This can be
The slave can reduce the data transfer speed of TWI by changing the duty cycle of SCL.

The following figure shows a typical data transfer. Note that multiple bytes can be transferred between SLA + R / W and STOP
Depending on the application software implementation agreement.




Typical TWI transmission

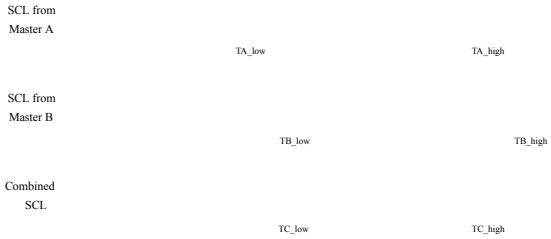
## Multi - host system and its arbitration and synchronization

The TWI protocol allows multiple hosts on the bus and uses special measures to ensure that even if two or more hosts are simultaneously
The transmission can also be handled like an ordinary transmission. Multi-host system will be two problems:

1. Implementation of the algorithm allows only one host in a host to complete the transmission. When other hosts find that they have lost their options
   To stop their transmission. The process of selection is called arbitration. When the competition in the host found its arbitration failed,
   Should immediately switch to the slave mode to detect whether it is subject to bus control by the host. In fact, more host with the same
   When the transmission should not be detected by the slave, that is not allowed to destroy the data being transmitted on the bus.
2. Different hosts may use different SCL frequencies. In order to ensure the consistency of transmission, you must design a synchronous host string
   Line clock program. This will simplify the arbitration process.

The lines and functions of the bus are used to solve the above problems. The serial clock of all the hosts will line up with one
A combined clock whose high time is equal to the shortest of all host clocks and its low level is equal to all host clocks
The longest one. All hosts are listening to SCL, and when the combined SCL clock goes high or low, they can effectively separate
Start counting their SCL high and low level overflow cycles.

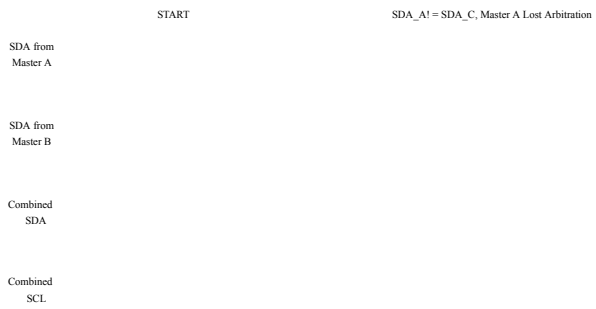
Multi-host SCL clock synchronization mechanism as shown below:




Multi - host SCL clock synchronization timing diagram



After the output data, all hosts continue to listen on the SDA line to implement the arbitration. If the value read from SDA with the main
The machine does not match the value of the output, the host that is lost arbitration. It is important to note that the host outputs a high level of SDA while the other one
The host will lose arbitration when the host outputs low-level SDA. The host that lost the arbitration should immediately switch to the slave mode and detect it
Whether it is addressed. The host that lost the arbitration must set the SDA line high, but before the end of the current data or address pack
To generate a clock signal. Arbitration will continue until the system has only one host, which may take multiple bits. If multiple
The host will address the same slave, and the arbitration will continue to the packet.




Arbitration between two hosts

Note that arbitration is not permitted in the following cases:

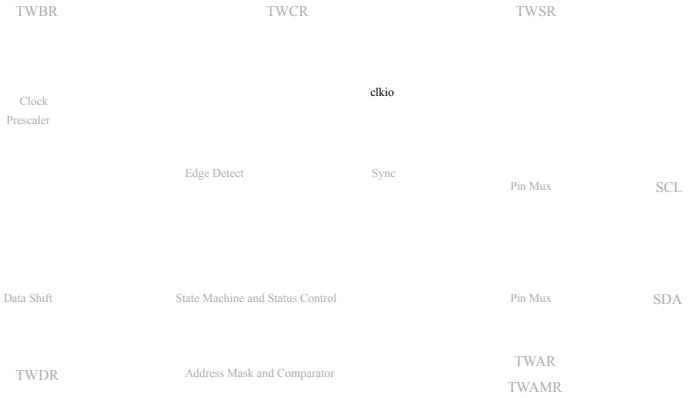
- A REPEATED START state with a data bit between;
- A STOP state and a data bit between;
- Between a REPEATED START state and a STOP status;

Application software must consider the above situation, to ensure that these illegal arbitration will not appear. 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 transmissions must contain phases With the same number of packets, otherwise the arbitration results can not be defined.

## Summary of TWI modules

The structure of the TWI module is shown in the following figure.





TWI Block structure diagram

TWI module mainly includes bit rate generator, bus interface unit, address comparator and control unit. See details below
Detailed description.

### Bit rate generator unit

The bit rate generator unit mainly controls the SCL clock period in master mode. The SCL clock period consists of the TWI bit rate register
TWBR and the prescaler control bits in the TWI status register TWSR. Slave operation is not subject to bit rate or prescaler
Set the effect, but to ensure that the working clock of the slave is at least 16 times the SCL frequency. Note that the slave may extend SCL
Of the low-level period, thereby reducing the TWI bus average clock frequency. The SCL clock frequency is generated by the following calculation formula:

$f_{sel} = f_{sys} / (16 + 2 * TWBR * 4^{TWPS})$

Where 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 the 8-bit TWDR,
The bus interface unit also includes an ACK / NACK register that is sent or received. This ACK / NACK register can not be directly addressed
Accessed by 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 statuses. When the MCU is in some
Sleep mode, START / STOP controller can still detect START and STOP status, when the host on the TWI bus search
The MCU will wake up from Sleep mode.

If TWI initiates a data transfer in host mode, the arbitration detection circuit will continuously monitor the bus to determine if it still owns
Bus control. When the TWI module loses bus control, the control unit will perform the correct action and generate the appropriate
Status code to inform 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 broadcast call identification enable bit (TWGCE) in the TWAR register is set and the address received from the bus is
Broadcast address comparison. Once the address matches successfully, the control unit will perform the correct action. The TWI module can respond or not respond
The addressing of the host depends on the setting of the TWCR register. Even in sleep mode, the address matching unit can be compared
Address, if the host on the bus address, the MCU wake from the sleep mode.

### control unit

The control unit is responsible for monitoring the bus and generating a corresponding response according to the settings of the TWCR. Occurs when the TWI bus needs to be applied
When the software participates in the event, the TWI interrupt flag bit TWINT will be set. In the next one clock cycle, TWI-like
The status register TWSR will be updated to indicate the status code for the event. When TWINT is set, TWSR contains the exact
status information. At other times, TWSR is a special status code, indicating that there is no exact status information. Once TWINT
The flag is set and the SCL line remains low, pausing the TWI transmission on the bus, allowing the application to process the event.

The TWINT flag is set in the following cases:

- TWI sends START / REPEATED START status
- TWI transmits SLA + R / W
- TWI sends an address byte after
- TWI bus after arbitration failed
- TWI is addressed by host (slave address match or broadcast mode)
- When it is addressed as a slave, it receives a STOP or REPEATED START
- When a bus error caused by an illegal START or STOP status occurs

### TWI use

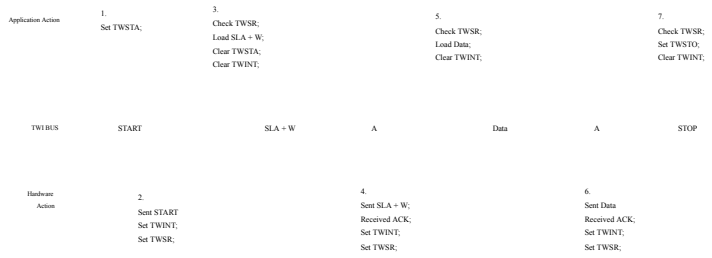
The TWI interface is byte-oriented and interrupt-based. All bus events, such as receiving a byte or sending a START
Signal, etc., will produce a TWI interrupt. Since TWI is based on the interrupt, so in the TWI byte transmission process,
Application software can be free to carry out other operations. TWECT register in TWCR interrupt enable bit TWIE and global interrupt enable
Can be used to control whether a TWI interrupt is generated when the TWINT flag is set. If the TWIE bit is cleared, the application is soft
The TWINT flag must be used to detect the action on the TWI bus.

When the TWINT flag is set, it indicates that the TWI interface completes the current operation and waits for the response of the application software. At this
In the case where the TWI status register TWSR contains a status code that reflects the current bus status. Application software can pass
Set the TWCR and TWDR registers to determine how the TWI interface will work at the next TWI bus cycle.

The following figure shows an example of how the application connects to the TWI interface. In this example, the host expects to send a byte of data to
Slave machine. The description here is very simple, the next chapter will be more detailed display.








TWI typical transmission process diagram

The TWI transmission process shown in the figure is:

1. The first step in TWI transmission is to send START. The TWI hardware is sent by writing a specific value to the TWCR register
   START signal. The values written will be described in detail later. It is very important to set TWINT in the written value,
   Writing a "1" to the TWINT bit clears this bit. TWI register TWINT set during the TWI does not start any operation.
   Once the software clears the TWINT bit, the TWI module immediately initiates the transmission of the START signal.
2. When the START status is transmitted, the TWINT flag of the TWCR is set, the TWSR is updated to the new status code,
   Indicates that the START signal was sent successfully.
3. The application looks at the value of TWSR and determines that the START status has been successfully sent. If TWSR is displayed as a different value,
   The application can perform special operations, such as calling error handlers. When the status code is determined to be consistent with the expected,
   The program loads the value of SLA + W into the TWDR register. The TWDR register can be used both in address and data.
   The software then writes a specific value to the TWCR register indicating the value of the SLA + W in the TWI hardware to send TWDR. write
   The incoming value will be described in detail later. Set TWINT in the written value to clear the TWINT flag. TWCR
   The TWINT of the register is set during TWI without initiating any operation. Once the software clears the TWINT bit, the TWI module
   Immediately start the delivery of the address packet.
4. When the address packet is sent, the TWINT flag of TWCR will be set and TWSR will be updated to the new status code.
   The address packet is sent successfully. The status code also reflects whether the slave responds to the address packet.
5. The application looks at the value of TWSR, determines that the address packet has been successfully sent, and the received ACK is the expected value. If TWSR
   Displayed as other values, the application can perform some special operations, such as calling an error handler. When determining the state
   When the code is consistent with the expected, the program loads the value of Data into the TWDR register. Then the software goes to the TWCR register
   Writes a specific value that indicates the value of Data in TWIT hardware to send TWDR. The values written will be described in detail later.
   Set TWINT in the written value to clear the TWINT flag. TWINT register of TWCR register is set
   TWI does not start any operation. Once the software clears the TWINT bit, the TWI module immediately initiates the transfer of the packet.
6. When the packet is sent, the TWINT flag of TWCR is set, TWSR is updated to the new status code,

The packet is successfully sent. The status code also reflects whether the slave responds to the packet.

7. The application looks at the value of TWSR to determine that the packet has been successfully sent and the received ACK is the expected value. If TWSR

Displayed as other values, the application can perform some special operations, such as calling an error handler. When determining the state

When the code is consistent with the expected, the software writes a specific value to the TWCR register indicating that the TWI hardware sends a STOP signal. write

The incoming value will be described in detail later. Set TWINT in the written value to clear the TWINT flag. TWCR

The TWINT of the register is set during TWI without initiating any operation. Once the software clears the TWINT bit, the TWI module

The STOP signal is transmitted immediately. It should be noted that TWINT will not be done after the STOP signal has been transmitted

Position.

Although the example is simpler, it contains all the rules in the TWI data transfer process. Summarized as follows:

- The TWINT flag is set when TWI completes an operation and waits for feedback from the application. The SCL clock line is always held





Pull down until TWINT is cleared;

- When the TWINT flag is set, the user must update all TWI register values for the next TWI bus cycle

The value. For example, the TWDR register must load the value to be sent by the next bus cycle.

- After updating all the registers while completing other necessary operations, the application writes the TWCR register. in

When TWCR is written, the TWINT bit must be set to clear the TWINT flag. After TWINT is cleared, TWI is on

The operation set by TWCR is executed.

### Transmission mode

TWI can work in the following four main modes: Host Transmitter (MT), Host Receiver (MR), Slave Transmitter (ST) and slave receiver (SR). Multiple modes can be used under the same application. For example, TWI can use the MT mode TWI EEPROM writes data and reads data from EEPROM in MR mode. If there are other hosts on the system, some It is also possible to send data to TWI, which will use SR mode. This is the application software to decide which mode to use.

These patterns will be described in detail below. In each mode of data transmission, will be combined with pictures to describe the possible shape Code. These pictures contain the following abbreviations:

S: Start state

Rs: REPEATED START state

R: Read operation flag (SDA is high)

W: write operation flag (SDA is low)

A: acknowledge bit (SDA is low)

NA: No acknowledge bit (SDA is high)

Data: 8-bit data bytes

P: STOP status

SLA: Slave Address

The circle in the picture is used to indicate that the TWINT flag is set. The number in the circle indicates the status code in the TWSR register. The prescaler control bit is masked as "0". In these places, the application must perform the appropriate action to continue or complete the TWI transmission. TWI transfers are pending until the TWINT flag is cleared.

When the TWINT flag is set, the status code in TWSR is used to determine the appropriate software operation. Each form is given in each form The code required for the software operation and subsequent serial transmission details. Note that the prescaler control bits in TWSR are masked Is "0".

### Host send mode

In the master transmit mode, TWI will send a certain number of data bytes to the slave receiver. In order to enter the host mode, Send a START signal. The next address packet format determines whether TWI is in the host transmitter mode or the host receiver mode. If you send SLA + W, enter the host send mode. If SLA + R is sent, it enters the host receive mode. This The status codes mentioned in one section assume that the prescaler control bit is "0".

Write the START signal by writing the following values to the TWCR register:

| TWINT | TWEA | TWSTA | TWSTO | TWWC | | TWEN | - | TWIE |
|---|---|---|---|---|---|---|---|---|
| 1 | x | 1 | 0 | x | | 1 | 0 | x |

TWEN bit must be set to "1" to enable TWI interface, TWSTA set to "1" to send START signal, TWINT set to "1" to clear



**Page 206**



TWINT flag. The TWI module detects the bus status and sends the START signal as soon as the bus is idle. When START is sent
, The hardware sets the TWINT flag, and updates the TWSR status code to 0x08.

In order to enter the host send mode, you must send SLA + W. This can be done by doing the following. Write to the TWDR register first
Into SLA + W, and then write "1" to the TWINT bit to clear the TWINT flag to continue transmission, that is, write to the TWCR register
Column value to send SLA + W:

| TWINT | TWEA | TWSTA | TWSTO | TWWC | | TWEN | - | TWIE |
|---|---|---|---|---|---|---|---|---|
| 1 | x | 0 | 0 | x | | 1 | 0 | x |

When SLA + W is transmitted and the acknowledge signal is received, TWINT is set and the status code of TWSR is updated. may
The status code is 0x18, 0x20 or 0x38. A suitable response to each status code is described in detail in the status code table.

When SLA + W is sent successfully, it can start sending packets. This can be done by writing data to the TWDR register.
TWDR can only be written when the TWINT flag is high. Otherwise, the access is ignored while writing the conflicting flag bit TWWC
Will be set. After updating the TWDR, write "1" to the TWINT bit to clear the TWINT flag to continue transmission. Ie to TWCR
The register writes the following values to send the data:

| TWINT | TWEA | TWSTA | TWSTO | TWWC | | TWEN | - | TWIE |
|---|---|---|---|---|---|---|---|---|
| 1 | x | 0 | 0 | x | | 1 | 0 | x |

When the packet is sent and the acknowledge signal is received, TWINT is set and the status code of the TWSR is updated. possible
The status code is 0x28 or 0x30. A suitable response to each status code is 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 host generates a STOP signal or a REPEATED START signal before the entire transmission ends.

The STOP signal is issued by writing the following values to the TWCR register:

| TWINT | TWEA | TWSTA | TWSTO | TWWC | | TWEN | - | TWIE |
|---|---|---|---|---|---|---|---|---|
| 1 | x | 0 | 1 | x | | 1 | 0 | x |

Write the REPEATED START signal by writing the following values 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 visit
Ask the new slave without sending a STOP signal. REPEATED START makes the host free of control of the bus without losing control
The switch between the different slaves, the host transmitter, and the host receiver mode.

The status code in the host transmit mode and the corresponding operation are shown in the following table:

Host status code table for sending mode

| status code | Bus and hard State of the item | Application software response Read / write TWDR | STA | STO | TWINT | TWEA | Operation of TWCR | The next step in the hardware |
|---|---|---|---|---|---|---|---|---|
| 0x08 | START has been issued | load SLA + W | 0 | 0 | 1 | x | | Will send SLA + W; ACK or NACK will be received |
| 0x10 | REPEATED START has been issued | load SLA + W | 0 | 0 | 1 | x | | Will send SLA + W; ACK or NACK will be received |



**Page 207**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | give away | load SLA + R | 0 | 0 | 1 | x | Will send SLA + R;<br>Will receive ACK or NACK;<br>Will switch to MR mode |
| 0x18 | SLA + W already send;<br>Received ACK | load data | 0 | 0 | 1 | x | Will send data;<br>ACK or NACK will be received |
| | | No operation For | 1 | 0 | 1 | x | Will send REPEATED START |
| | | No operation For | 0 | 1 | 1 | x | Will send STOP;<br>The TWSTO flag is reset |
| | | No operation For | 1 | 1 | 1 | x | Will send STOP;<br>The TWSTO flag will be reset;<br>Will send START |
| 0x20 | SLA + W has been send;<br>received NACK | load data | 0 | 0 | 1 | x | Will send data;<br>ACK or NACK will be received |
| | | No operation For | 1 | 0 | 1 | x | Will send REPEATED START |
| | | No operation For | 0 | 1 | 1 | x | Will send STOP;<br>The TWSTO flag is reset |
| | | No operation For | 1 | 1 | 1 | x | Will send STOP;<br>The TWSTO flag will be reset;<br>Will send START |
| 0x28 | Data bytes Has been sent<br>Received an ACK | load data | 0 | 0 | 1 | x | Will send data;<br>ACK or NACK will be received |
| | | No operation For | 1 | 0 | 1 | x | Will send REPEATED START |
| | | No operation For | 0 | 1 | 1 | x | Will send STOP;<br>The TWSTO flag is reset |
| | | No operation For | 1 | 1 | 1 | x | Will send STOP;<br>The TWSTO flag will be reset;<br>Will send START |
| 0x30 | Data bytes Has been sent<br>Received NACK | load data | 0 | 0 | 1 | x | Will send data;<br>ACK or NACK will be received |
| | | No operation For | 1 | 0 | 1 | x | Will send REPEATED START |
| | | No operation For | 0 | 1 | 1 | x | Will send STOP;<br>The TWSTO flag is reset |
| | | No operation For | 1 | 1 | 1 | x | Will send STOP;<br>The TWSTO flag will be reset;<br>Will send START |
| 0x38 | SLA + W or Data arbitration failure | No operation For | 0 | 0 | 1 | x | Will release the bus;<br>Will enter the unaddressed slave mode |
| | | No operation For | 1 | 0 | 1 | x | Will be sent when idle START |



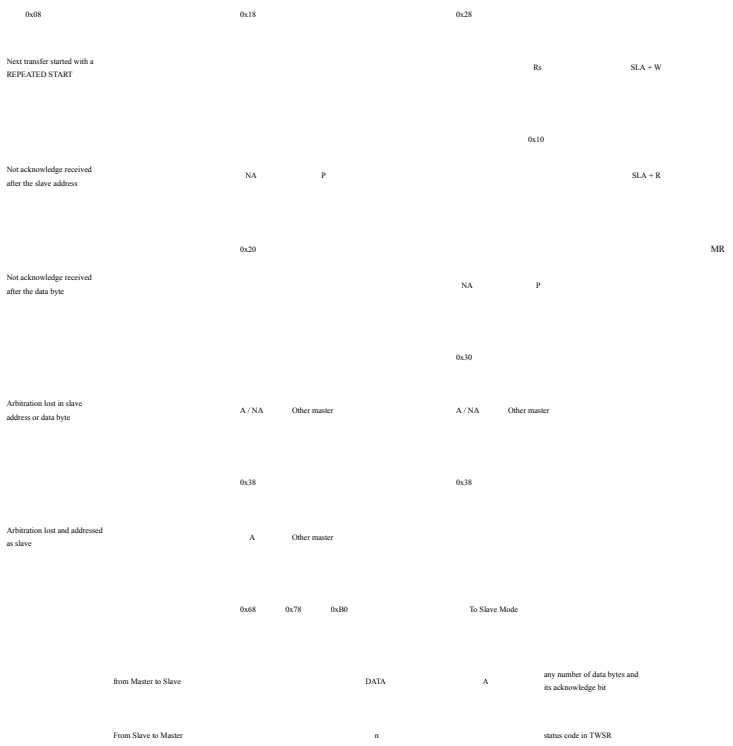
**Page 208**



The format and status of the host send mode are as follows:

MT

S SLA + W A DATA A P




The format and status of the host send mode

### Host receive mode

In host receive mode, TWI receives a certain number of data bytes from the slave transmitter. In order to enter the host mode,
Send a START signal. The next address packet format determines whether TWI is in the host transmitter mode or the host receiver
mode. If you send SLA + W, enter the host send mode. If SLA + R is sent, it enters the host receive mode. This
The status codes mentioned in one section assume that the prescaler control bit is "0".

Write the START signal by writing the following values to the TWCR register:

| TWINT | TWEA | TWSTA | TWSTO | TWWC | TWEN | - | TWIE |
|---|---|---|---|---|---|---|---|
| 1 | x | 1 | 0 | x | 1 | 0 | x |





TWEN bit must be set to "1" to enable TWI interface, TWSTA set to "1" to send START signal, TWINT set to "1" to clear
TWINT flag. The TWI module detects the bus status and sends the START signal as soon as the bus is idle. When START is sent
, The hardware sets the TWINT flag, and updates the TWSR status code to 0x08.

In order to enter the host receive mode, you must send SLA + R. This can be done by doing the following. Write to the TWDR register first
Into SLA + R, and then write "1" to the TWINT bit to clear the TWINT flag to continue transmission, that is, write to the TWCR register
Column value to send SLA + R:

| TWINT | TWEA | TWSTA | TWSTO | TWWC | TWEN | - | TWIE |
|---|---|---|---|---|---|---|---|
| 1 | x | 0 | 0 | x | 1 | 0 | x |

When the SLA + R is transmitted and the acknowledge signal is received, TWINT is set and the status code of the TWSR is updated. possible
The status code is 0x38, 0x40, or 0x48. A suitable response to each status code is described in detail in the status code table.

When SLA + R is sent successfully, it can start receiving packets. Write the "TWINT" bit by writing "1" to the TWINT bit
Continue to receive. The following values are written to the TWCR register to initiate reception:

| TWINT | TWEA | TWSTA | TWSTO | TWWC | TWEN | - | TWIE |
|---|---|---|---|---|---|---|---|

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | x | 0 | 0 | x | 1 | 0 | x |

When the packet is received and the acknowledge signal is transmitted, TWINT is set and the status code of the TWSR is updated. possible
The status code is 0x50 or 0x58. A suitable response to each status code is described in detail in the status code table.

When the data reception is successful, you can continue to receive data packets. This process is repeated until the last byte is received.
After the host receives the last byte, it must send a NACK acknowledge signal to the slave transmitter. The host generates a STOP signal
Or the REPEATED START signal is completed by the entire reception.

The STOP signal is issued by writing the following values to the TWCR register:

| TWINT | TWEA | TWSTA | TWSTO | TWWC | TWEN | - | TWIE |
|---|---|---|---|---|---|---|---|
| 1 | x | 0 | 1 | x | 1 | 0 | x |

Write the REPEATED START signal by writing the following values 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 host again, or visit
Ask the new host without sending a STOP signal. REPEATED START makes the host free of control of the bus without losing control
The switch between the different slaves, the host transmitter, and the host receiver mode.

The status code in the host receive mode and the corresponding operation are shown in the following table:

Host receive mode status code table

| status code | Bus and hard State of the item | Application software response Read / write TWDR | Operation of TWCR STA | STO | TWINT | TWEA | The next step in the hardware |
|---|---|---|---|---|---|---|---|
| 0x08 | START has been send | load SLA + R | 0 | 0 | 1 | x | Will send SLA + R; ACK or NACK will be received |
| 0x10 | REPEATED START has been | load SLA + R | 0 | 0 | 1 | x | Will send SLA + R; ACK or NACK will be received |





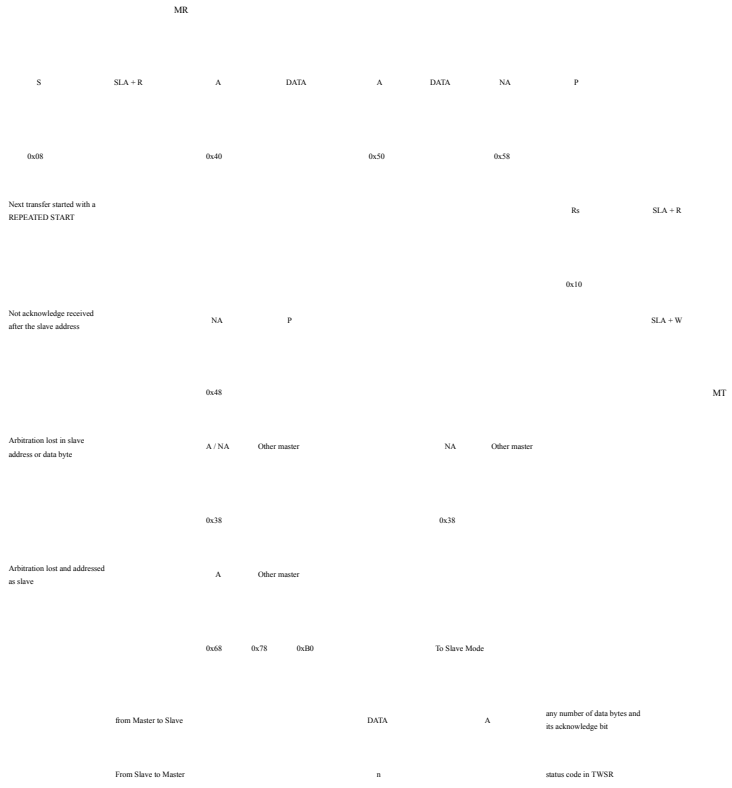
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | send | load SLA + W | 0 | 0 | 1 | x | Will send SLA + W; Will receive ACK or NACK; Will switch to MT mode |
| 0x38 | SLA + R or Data arbitration failure | No operation For | 0 | 0 | 1 | x | Will release the bus; Will enter the unaddressed slave mode |
| | | No operation For | 1 | 0 | 1 | x | Will be sent when idle START |
| 0x40 | SLA + R has been send; received ACK | No operation For | 0 | 0 | 1 | 0 | Will receive data; Will send NACK |
| | | No operation For | 0 | 0 | 1 | 1 | Will receive data; ACK will be sent |
| 0x48 | SLA + R has been send; received NACK | No operation For | 1 | 0 | 1 | x | Will send REPEATED START |
| | | No operation For | 0 | 1 | 1 | x | Will send STOP; The TWSTO flag is reset |
| | | No operation For | 1 | 1 | 1 | x | Will send STOP; The TWSTO flag will be reset; Will send START |
| 0x50 | Data bytes Received; ACK has been issued | Read data | 0 | 0 | 1 | 0 | Will receive data; Will send NACK |
| | | Read data | 0 | 0 | 1 | 1 | Will receive data; ACK will be sent |
| | give away | | | | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0x58 | Data bytes Received; | Read data | 1 | 0 | 1 | x | Will send REPEATED START |
| | NACK has been send | Read data | 0 | 1 | 1 | x | Will send STOP; The TWSTO flag is reset |
| | | Read data | 1 | 1 | 1 | x | Will send STOP; The TWSTO flag will be reset; Will send START |

The format and status of the host receive mode are as follows:








The format and status of the host receive mode

### Slave receive mode

In the slave receive mode, a certain number of data bytes can be received from the host transmitter. The status code mentioned in this chapter
It is assumed that the prescaler control bit is "0".

To start the 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 addresses that the TWI interface responds to when the host is addressed. If LSB is set, TWI responds to broadcast call
Called address (0x00), otherwise ignore the general call address.

TWCR should 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, TWEA must be set to enable the host to address (slave address or broadcast call) to
Return to the confirmation message ACK. TWSTA and TWSTO must be cleared.





After the TWAR and TWCR are initialized, the TWI interface starts waiting until its slave address (or broadcast address) is found
site. The TWI enters the slave receive mode when the data direction bit of the slave address is "0" (indicating a write operation). When the number
When the direction bit is "1" (indicating read operation), TWI enters the slave transmission mode. Receive their own slave address and write operation
After the flag is set, the TWINT flag is set and the valid status code is also updated to TWSR. Each status code under the appropriate
The response is described in detail in the status code table. It should be noted that, when the host mode under the TWI arbitration can also be failed
Enter the slave receive mode (see status codes 0x68 and 0x78).

If the TWEA bit is reset during transmission, TWI will return NACK (high) to SDA after receiving a byte
on-line. This can be used to indicate that the slave can not receive more data. When the TWEA bit is "0", TWI will not respond to itself
The slave address. However, TWI will still monitor the bus, once TWEA is set, you can restore the address recognition and response. and also
That is, you can use TWEA to temporarily isolate the TWI interface from the bus.

The clock of the TWI interface can be turned off in other sleep modes except the idle mode. If the slave can receive the mode,
The interface will continue to respond to the slave address or broadcast address using the bus clock. Address match will wake up the MCU. During awakening,
The TWI interface will remain SCL low until the TWINT flag is cleared. When the TWI interface clock returns to normal
Receive more data.

The status code of the slave receive mode is shown in the following table:

Slave receive mode status code table

| status code | Bus and hardware status | Application software response: Read / write TWDR | Operation of TWCR: STA | STO | TWINT | TWEA | The next step in the hardware |
|---|---|---|---|---|---|---|---|
| 0x60 | SLA + W has been received; ACK has been sent | No operation | x | 0 | 1 | 0 | Will receive data; Will send NACK |
| | | No operation | x | 0 | 1 | 1 | Will receive data; ACK will be sent |
| 0x68 | When sending SLA + R / W Failure of arbitration; SLA + W has been received; ACK has been sent | No operation | x | 0 | 1 | 0 | Will receive data; Will send NACK |
| | | No operation | x | 0 | 1 | 1 | Will receive data; ACK will be sent |
| 0x70 | The broadcast address has been received; ACK has been sent | No operation | x | 0 | 1 | 0 | Will receive data; Will send NACK |
| | | No operation | x | 0 | 1 | 1 | Will receive data; ACK will be sent |
| 0x78 | When sending SLA + R / W Failure of arbitration; SLA + W has been received; ACK has been sent | No operation | x | 0 | 1 | 0 | Will receive data; Will send NACK |
| | | No operation | x | 0 | 1 | 1 | Will receive data; ACK will be sent |
| 0x80 | Own data has been received; ACK has been 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 |
| 0x88 | Own data has been received; read | | 0 | 0 | 1 | 0 | Will switch to unreserved |



**Page 213**



| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | NACK has been sent | data | | | | | Slave mode;<br>Will not respond from the ground<br>Address and broadcast |
| | | Read data | 0 | 0 | 1 | 1 | Will switch to unreserved<br>Slave mode;<br>Will respond from the ground site;<br>When TWGCE = 1,<br>Should be broadcast |
| | | Read data | 1 | 0 | 1 | 0 | Will switch to unreserved<br>Slave mode;<br>Will not respond from the ground<br>Address and broadcast<br>When the bus is idle,<br>Send START |
| | | Read data | 1 | 0 | 1 | 1 | Will switch to unreserved<br>Slave mode;<br>Will respond from the ground site;<br>When TWGCE = 1,<br>Should be broadcast<br>When the bus is idle,<br>Send START |
| 0x90 | Broadcast data received; ACK has been sent | Read data | x | 0 | 1 | 0 | Will receive data;<br>Will send NACK |
| | | Read data | x | 0 | 1 | 1 | Will receive data;<br>ACK will be sent |
| 0x98 | Broadcast data received; NACK has been sent | Read data | 0 | 0 | 1 | 0 | Will switch to unreserved<br>Slave mode;<br>Will not respond from the ground<br>Address and broadcast |
| | | Read data | 0 | 0 | 1 | 1 | Will switch to unreserved<br>Slave mode;<br>Will respond from the ground site;<br>When TWGCE = 1,<br>Should be broadcast |
| | | Read data | 1 | 0 | 1 | 0 | Will switch to unreserved<br>Slave mode;<br>Will not respond from the ground<br>Address and broadcast<br>When the bus is idle,<br>Send START |
| | | Read | 1 | 0 | 1 | 1 | Will switch to unreserved |

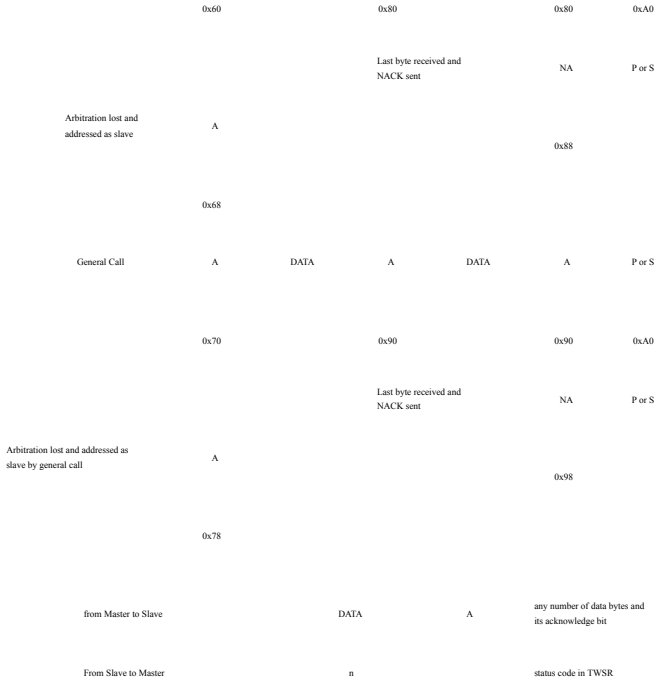
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | data | | | | | Slave mode;<br>Will respond from the ground site;<br>When TWGCE = 1,<br>Should be broadcast<br>When the bus is idle,<br>Send START |
| 0xA0 | When the machine is working,<br>To STOP or REPEATED START | No operation For | 0 | 0 | 1 | 0 | Will switch to unreserved Slave mode;<br>Will not respond from the ground Address and broadcast |
| | | No operation For | 0 | 0 | 1 | 1 | Will switch to unreserved Slave mode;<br>Will respond from the ground site;<br>When TWGCE = 1,<br>Should be broadcast |
| | | No operation For | 1 | 0 | 1 | 0 | Will switch to unreserved Slave mode;<br>Will not respond from the ground Address and broadcast<br>When the bus is idle,<br>Send START |
| | | No operation For | 1 | 0 | 1 | 1 | Will switch to unreserved Slave mode;<br>Will respond from the ground site;<br>When TWGCE = 1,<br>Should be broadcast<br>When the bus is idle,<br>Send START |

The format and status of the slave receive mode are as follows:



SR

S SLA + W A DATA A DATA A P or S




Slave receive mode format and status diagram

### Slave send mode

In the slave transmit mode, a certain number of data bytes can be sent to the host receiver. The status code mentioned in this chapter
It is assumed that the prescaler control bit is "0".

To start the 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 addresses that the TWI interface responds to when the host is addressed. If LSB is set, TWI responds to broadcast call
Called address (0x00), otherwise ignore the general call address.
TWCR should 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, TWEA must be set to enable the host to address (slave address or broadcast call) to
Return to the confirmation message ACK. TWSTA and TWSTO must be cleared.

After the TWAR and TWCR are initialized, the TWI interface starts waiting until its slave address (or broadcast address) is found
site. The TWI enters the slave receive mode when the data direction bit of the slave address is "0" (indicating a write operation). When the number
When the direction bit is "1" (indicating read operation), TWI enters the slave transmission mode. Receive their own slave address and read gymnastics
After the flag is set, the TWINT flag is set and the valid status code is also updated to TWSR. Each status code under the appropriate
The response is described in detail in the status code table. It should be noted that, when the host mode under the TWI arbitration can also be failed
Enter the slave transmit mode (see status code 0xB0).

If the TWEA bit is reset during transmission, TWI will switch to unaddressed slave mode after sending the last byte.
The status code in the TWSR register will be updated after the host receiver gives NACK or ACK for the last byte of transmission.
For 0xC0 or 0xC8. If the host receiver continues to transmit the operation, the slave transmitter will not respond, the host will receive the full
"1" data (ie 0xFF). When the slave sends the last byte of data (TWEA is cleared) and the NACK is expected

Response, and the host wants to receive more data to send ACK as a response, TWSR will be updated to 0xC8.

When the TWEA bit is "0", TWI does not respond to its own slave address. But TWI will still listen to the bus once TWEA
Is set, you can restore the address recognition and response. That is, you can use TWEA to temporarily switch the TWI interface from the bus
Isolated out.

The clock of the TWI interface can be turned off in other sleep modes except the idle mode. If the slave can receive the mode,
The interface will continue to respond to the slave address or broadcast address using the bus clock. Address match will wake up the MCU. During awakening,
The TWI interface will remain SCL low until the TWINT flag is cleared. When the TWI interface clock returns to normal
Receive more data.

The status code of the slave transmit mode is shown in the following table:

Slave status code table for sending mode

| status code | Bus and hard State of the item | Application software response Read / write TWDR | Operation of TWCR STA | STO | TWINT | TWEA | The next step in the hardware |
|---|---|---|---|---|---|---|---|
| 0xA8 | SLA + R has been receive; ACK has been issued give away | Load data x | 0 | 0 | 1 | 0 | Will send the last number according to; It is desirable to receive NACK |
| | | Load data x | 0 | 0 | 1 | 1 | Will send data; ACK will be received |
| 0xB0 | hair give away SLA + R / W When the arbitration was lost defeat; SLA + R has been receive; ACK has been issued give away | Load data x | 0 | 0 | 1 | 0 | Will send the last number according to; It is desirable to receive NACK |
| | | Load data x | 0 | 0 | 1 | 1 | Will send data; ACK will be received |





| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0xB8 | Data has been issued Send; Received | Load data x | 0 | 0 | 1 | 0 | Will send the last number according to; It is desirable to receive NACK |
| | | Load data x | 0 | 0 | 1 | 1 | Will send data; ACK will be received |
| 0xC0 | Data has been issued give away; NACK has been receive | No operation | 0 | 0 | 1 | 0 | Will switch to unreserved slave mode; Will not respond to the slave address and broadcast |
| | | No operation | 0 | 0 | 1 | 1 | Will switch to unreserved slave mode; Will respond to the slave address; TWGCE = 1 will respond widely broadcast |
| | | No operation | 1 | 0 | 1 | 0 | Will switch to unreserved slave mode; Will not respond to the slave address and broadcast; Bus will be sent when idle START |
| | | No operation | 1 | 0 | 1 | 1 | Will switch to unreserved slave mode; Will respond to the slave address; |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | TWGCE = 1 will respond widely broadcast;<br>Bus will be sent when idle START |
| 0xC8 | the last one Data has been issued give away; ACK has been received | No operation | 0 | 0 | 1 | 0 | Will switch to unreserved slave mode;<br>Will not respond to the slave address and broadcast |
| | Receive | No operation | 0 | 0 | 1 | 1 | Will switch to unreserved slave mode;<br>Will respond to the slave address;<br>TWGCE = 1 will respond widely broadcast |
| | | No operation | 1 | 0 | 1 | 0 | Will switch to unreserved slave mode;<br>Will not respond to the slave address and broadcast;<br>Bus will be sent when idle START |
| | | No operation | 1 | 0 | 1 | 1 | Will switch to unreserved slave mode;<br>Will respond to the slave address;<br>TWGCE = 1 will respond widely broadcast;<br>Bus will be sent when idle START |





The format and status of the slave transmit mode are shown in the following figure:


The format and status of the slave mode

## Other status

There are two status codes that do not have a corresponding TWI state definition, as shown in the following table:

Other status code table

| Bus and hardware | Application software response |
|---|---|

| status code | state | Read / write TWDR | STA | Operation of TWCR STO | TWINT | TWEA | The next step in the hardware |
|---|---|---|---|---|---|---|---|
| 0xF8 | Stateless information TWINT = 0 | No operation | | Do not operate TWCR | | | Wait or perform the current operation For |
| 0x00 | Illegal START Or STOP The bus is wrong | No operation | 0 | 1 | 1 | x | Only affect the internal hardware; No STOP is sent Bus; bus release And clear the TWSTO bit |

The status code 0xF8 indicates that there is no relevant information at this time because the TWINT flag is "0". This state may occur at TWI
The port is not involved in serial transmission or the current transmission has not yet been completed.





State 0x00 indicates that a bus error occurred during serial transmission. Bus error when illegal START or STOP occurs
It will happen. For example, there is START or STOP between address and data, address, and ACK. The bus error will be set
TWINT. In order to recover from the error, TWSTO must be set and cleared by writing "1" to clear TWINT. This will cause TWI to connect
The port enters the unaddressed slave mode without generating a STOP, releases SCL and SDA, and clears the TWSTO bit.
Combination mode

In some cases, several TWI modes must be combined to achieve the desired job. For example, from the serial EEPROM
Read the data, the typical transmission includes the following steps:
1. The transmission must be initiated;
2. You must tell the EEPROM where the data should be read;
3. must complete the read operation;
4. The transmission must end.
Note that data can be transferred from the host to the slave, and vice versa. The host tells the slave to read the location of the data, using the master
Machine sending mode. Next, read data from the slave, using the host receive mode. The direction of transmission will change. Host
Must maintain the various stages of the bus control, all the steps are uninterrupted operation. If in a multi-host system,
Between 2 and 3 another host to change the location of the data read, then break the principle of the host to read the data location
Would be wrong. Change the direction of data transmission by sending the address byte and receiving data between the sending REPEATED
START to achieve. After sending REPEATED START, the host still has bus control.

The following figure describes the transmission process:


Combine multiple TWI modes to access the serial EEPROM diagram

### Multi - host system and arbitration

If more than one host is connected to the same TWI bus, one or more of them may start the data transfer at the same time.
The TWI protocol ensures that in this case, through an arbitration process, one of the hosts is allowed to transmit and will not be lost
data. Here the two host to try to send data from the machine as an example to describe the bus arbitration process.

There are several different situations that will result in a bus arbitration process:

- Two or more hosts communicate with a slave at the same time. In this case, both the host and the slave do not know
  There is competition on the bus;
- Two or more hosts simultaneously access different data or operating directions for the same slave. In this case will be
  Arbitration occurs at the READ / WRITE bit or data bit. When there are other hosts to the SDA line to send "0", to the SDA line
  The host that sent "1" will fail the arbitration. The failed host will switch to the un-addressed slave mode, or so on
  When the bus is idle, a new START signal is sent, depending on the operation of the application software.

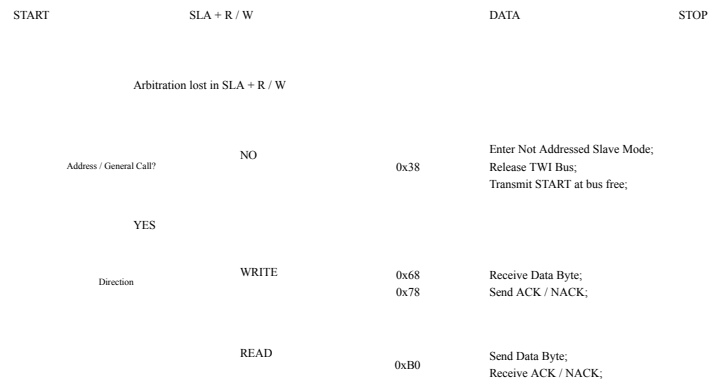
- Two or more hosts access different slaves. In this case, bus arbitration occurs in the SLA phase. When there is
  When the host sends a "0" to the SDA line, the host that sends "1" to the SDA line will fail the arbitration. On the SLA bus
  The failed host will switch to slave mode and check if it is being addressed by the host for bus control. Such as
  The value is addressed, it will enter the SR or ST mode, depending on the READ / WRITE bit following the SLA. If not found





Address, it will switch to unreserved slave mode, or wait for the bus to send a new START signal,
This depends on the operation of the application software.

The following figure describes the process of bus arbitration:




Bus arbitration process

## Register definition

TWI register list

| register | address | Defaults | 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 bit rate register

TWBR - TWI bit rate register

Address: 0xB8 Default: 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 bit.<br>TWBR is the bit rate generator division factor. The bit rate generator is a divider,<br>Used to generate the SCL clock in master mode. The bit rate is calculated as follows:<br>$f_{scl} = f_{sys} / (16 + 2 * TWBR * 4^{TWPS})$. |

### TWSR - TWI status register

TWSR - TWI status register

Address: 0xB9 Default: 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 |

| Bit | Name | description |
|---|---|---|
| 7: 3 | TWS [7: 3] | TWI status flag.<br>5-bit TWS response TWI logic and bus status. Different status values are different<br>The meaning of the TWI working mode is described in detail. The values read from TWSR include 5<br>Bit status value and 2-bit prescaler control bit, the pre-division should be masked when the status is detected<br>The frequency is "0". This is the state detection independent of the prescaler setting. |
| 2 | - | Keep it. |
| 1 | TWPS1 | TWI prescaler control high.<br>TWPS1 and TWPS0 together form TWPS [1: 0] to control the bit rate prescaler<br>Factor, and TWBR together to control the bit rate. |
| 0 | TWPS0 | TWI prescaler control low.<br>TWPS0 and TWPS1 together form TWPS [1: 0] to control the bit rate prescaler<br>Factor, and TWBR together to control the bit rate. |

| TWPS [1: 0] | Prescaler factor |
|---|---|
| 0 | 1 |
| 1 | 4 |
| 2 | 16 |
| 3 | 64 |

### TWAR - TWI address register

TWAR - TWI address register

Address: 0xBA Default: 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 |

| Bit | Name | description |
|---|---|---|
| 7: 1 | TWA [6: 0] | TWI slave address bits.<br>TWA is the TWI slave address. When TWI is operating in slave mode, TWI will be based<br>This address responds. Host mode does not require this address. But in multi-host systems,<br>Also need to set the slave address for other host access. |
| 0 | TWGCE | TWI broadcast identification enable control bit.<br>When the TWGCE bit is set to "1", the TWI bus broadcast identification is enabled.<br>When setting the TWGCE bit to "0", the TWI bus broadcast recognition is disabled.<br>When TWGCE is set and the received address frame is 0x00, the TWI module responds to this<br>Bus broadcast. |





### TWDR - TWI data register

TWDR - TWI data register

Address: 0xBB Default: 0xFF

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | TWD7 | TWD6 | TWD5 | TWD4 | TWD2 | TWD2 | 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 register.<br>TWD is going to transmit the next byte on the bus, or just received from the bus<br>To the last byte. |

### TWCR - TWI control register

TWCR - TWI control register

Address: 0xBC Default: 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.<br>When TWI completes the current job and wants the application to intervene, the hardware sets the TWINT bit.<br>If the global interrupt is set and the TWIE bit is set, a TWI interrupt will be generated and the MCU will execute<br>TWI interrupt service routine. When the TWINT flag is set, the low level of the SCL signal will be<br>Was extended.<br>The TWINT flag can only be cleared by writing a "1" to this bit. Even if the execution of the service<br>Program, the hardware will not automatically clear the bit. Also note that clearing this bit will be immediately<br>Turn on TWI operation. Therefore, before clearing the TWINT bit, first complete the TWAR,<br>TWAMR, TWSR and TWDR registers. |
| 6 | TWEA | TWI enables response control bits.<br>The TWEA bit controls the generation of the acknowledge pulse. When the TWEA bit is set to "1", and the following is satisfied<br>One of the conditions will be generated on the TWI bus response pulse:<br>1) Receives the slave address of the device;<br>2) received a call when TWGCE is set;<br>3) Receive one byte of data in the host receive or slave receive mode.<br>When the TWEA bit is set to "0", the device is temporarily disconnected from the TWI bus. Set the device<br>Re-address address recognition. |
| 5 | TWSTA | TWI start status control bit.<br>The TWSTA bit needs to be set when the CPU wishes to become a host on the TWI bus. hard<br>Will detect whether the bus is available, and when the bus is idle, it starts on the bus<br>state. When the bus is not idle, TWI will wait until the detection of the stop state occurs<br>Generate the starting state to declare that you want to be the host. After sending the start status software<br>The TWSTA bit must be cleared. |





| | | |
|---|---|---|
| 4 | TWSTO | TWI stop status control bit.<br>In the host mode, when the TWSTO bit is "1", TWI will generate a stop state on the bus,<br>The TWSTO bit is then cleared automatically. In the slave mode, setting the TWSTO bit enables TWI<br>Recover from the wrong state. At this time will not produce a stop state, will only let TWI return to<br>A defined unreserved slave mode, while releasing the SCL and SDA signal lines to<br>High impedance state. |
| 3 | TWWC | TWI write conflicting flag.<br>When the TWINT flag is low, writing the TWDR register bit will be set in the TWDR register.<br>When the TWINT flag is high, writing the TWDR register will clear the TWWC flag. |
| 2 | TWEN | TWI enable control bit.<br>The TWEN bit enables TWI operation and activates the TWI interface. When the TWEN bit is set to "1"<br>The TWI control IO pin is connected to the SCL and SDA pins. When the TWEN bit is set to "0"<br>The TWI interface module is turned off and all transmissions are terminated, including the ongoing operation. |

| | | |
|---|---|---|
| 1 | - | Keep it. |
| 0 | TWIE | TWI interrupt enable control bit.<br>When the TWIE bit is set to "1" and the global interrupt is set, the TWINT flag is set as<br>High, the TWI interrupt request is activated. |

### TWAMR - TWI address mask register

TWAMR - TWI address mask register

Address: 0xBD Default: 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.<br>TWAM is a 7-bit TWI slave address mask control. TWAM is used for each screen<br>The corresponding address bits in the TWAR. When the mask bit is set, the address matches the logic<br>The series will ignore the comparison of the received address bits with the corresponding bits of TWA. The following figure shows<br>Details of the address match logic. |
| 0 | - | Keep it. |

### TWI address matching logic

The following figure shows the TWI address matching logic block diagram:




## Analog Comparator 0 (AC0)

- 10mV more accurate
- Factory misalignment calibration
- Supports 3 off-chip analog inputs
- Supports multiplexing input (ADMUX) of ADC
- Supports internal differential amplifier input (DFFO)
- Supports internal 8-bit DAC input (DAO)
- Programmable output digital filter control

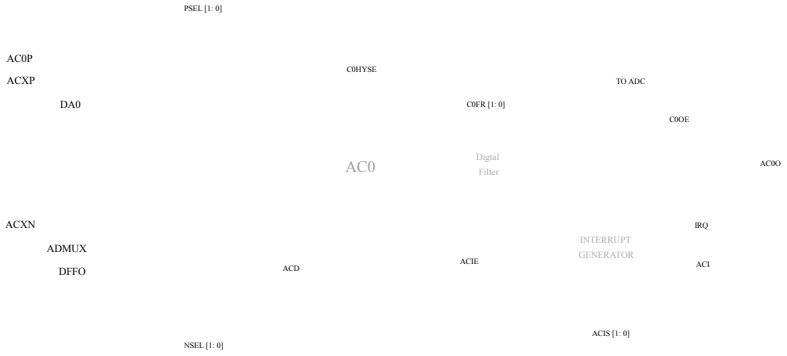
### Summary

The analog comparator compares the positive and negative levels of the input comparator. When the positive terminal voltage is higher than the negative terminal voltage, The comparator output ACO is set. When the level of ACO changes, the edge of the signal can be used to trigger an interrupt. Output The signal ACO can also be used to trigger the input capture of the timer counter 1 and to control the PWM output generated by the timer.

The LGT8FX8P integrates the analog comparator AC0, including a multiplexed analog input selector, the comparator positive and negative input sources To select from external ports or from a variety of internally generated reference sources. The analog comparator itself supports offset calibration, and can To ensure the consistency of the work of the comparator. The comparator supports an optional hardware hysteresis function to improve the stability of the comparator output Qualitative. While the comparator output integrated a hardware can be programmed digital filter, according to the application needs, select the appropriate Of the filter settings to get a more stable comparison output.

Comparator output status can be read directly through the register, can also generate interrupt requests, to achieve more efficient real-time events Capture function. The output of the comparator can also be output directly to the external IO port.

The structure of the op amp / analog comparator 0 is shown in the following figure.




Analog comparator 0 function diagram

## Analog comparator input

Both inputs of the analog comparator support a variety of optional input sources. Positive input three-way optional:

1. External independent analog input AC0P
2. Analog Comparator 0/1 Common Analog Input ACXP
3. Internal 8-bit DAC output DAO





The selection of the input source is controlled by the C0BG bit in the control status register C0SR and the C0PS0 bit of the C0XR register For details, refer to the section description of the registers in this section.

AC0P is AC0 dedicated positive mode input channel. Note that the pinings of the different packages AC0P are slightly different. QFP48 The encapsulated AC0P is a separate port. QFP32 encapsulates this AC0P port in parallel with PD6 to a port.

ACXP is a positive input for comparator 0/1. LGT8FX8P internal two analog comparators, ACXP connected to the same time The two comparators of the positive multiplexing selector, easy to achieve the two comparators work together.

DAO from the internal 8-bit DAC output. The DAC reference source can be selected from the system power supply, internal reference or come Input from external reference. Refer to the DAC section for DAC configuration.

| C0BG | C0PS0 | AC0 positive input source |
|---|---|---|
| 0 | 0 | AC0P |
| 0 | 1 | ACXP |
| 1 | 0 | DAO |
| 1 | 1 | Turn off the comparator positive input channel |

Negative input can also choose 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 comparator negative input is selected as ADMUX, it needs to pass the ADC module ADMUX register CHMUX bit selection mode To enter the input channel, this mode, the comparator input can achieve a more flexible expansion.

ACXN for the comparator 0/1 common negative input, easy to achieve the comparator 0/1 co-operation;

DFFO from the internal differential amplifier output. Differential amplifier optional x1 / x8 / x16 / x32 gain control, can be achieved small Signal detection and measurement.

| CME01 | CME00 | AC0 negative input source |
|---|---|---|
| 0 | 0 | ACXN |
| 0 | 1 | ADMUX |
| 1 | 0 | DFFO |
| 1 | 1 | Turn off the comparator negative input channel |

### Comparator output filter

The comparator output supports a controlled hysteresis internally. The user can enable the C0HYSE bit in the C0XR register
Hysteresis circuit. Hysteresis circuit can eliminate the unstable state of the comparator state change process, to achieve the output filter function.

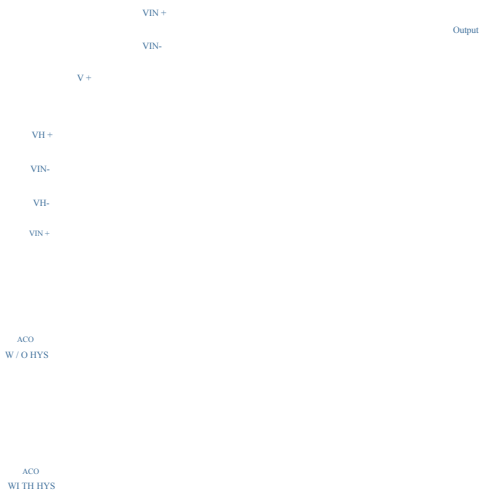
It is recommended that the user use the comparator to turn on the hysteresis circuit and obtain a stable comparator output.

As shown in the following figure, the hysteresis circuit is located between the analog output of the comparator and the digital output. When the comparator is the positive input volta
When $V_{IN+}$ is greater than ($V_{IN-} + V_{H+}$), the comparator COUT output is high; when $V_{IN+}$ voltage is less than ($V_{IN-} - V_{H-}$), the comparator output is low.
The hysteresis circuit avoids jitter due to the circuit itself when the positive terminal voltage of the comparator approaches the negative terminal voltage.

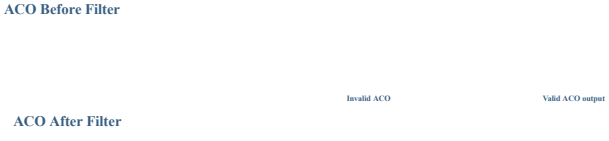
Comparator hysteresis voltage and comparator output diagram:








Although the hysteresis circuit is very effective for suppressing voltage ripple near the comparator threshold, the actual application environment,
The signal will be disturbed by different intensities. Strong interference may cause the input level to instantaneously raise the valve beyond the hysteresis circuit
Value range, can not be effectively suppressed. The LGT8FX8P integrates a programmable digital filter at the comparator output.
To filter out the effects of instantaneous interference on the output of the comparator. The digital filter can select the appropriate filter according to the application requirements
Time width, the filter circuit updates the output of the comparator only when the output of the comparator continues to meet the filter time limit.
Thus achieving a more stable output.




Comparator output filter timing

AC0 digital filter through the C0XR register C0FEN and C0FS bit control, the specific settings please refer to this chapter
Register definition section.

### Comparator output and PWM control

LGT8FX8P supports multi-channel PWM output, PWM signal can be used in conjunction with the comparator module. The input of the comparator Out, can be used to directly turn off the PWM signal, in order to achieve a more flexible PWM protection program.

For control related to PWM output, refer to the relevant section of the timer section.





## Register definition

### C0SR - AC0 control and status register

C0SR - AC0 control and status register

Address: 0x50 Default: 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.<br>When the C0D bit is set to "1", the analog comparator is turned off.<br>When the C0D bit is set to "0", the analog comparator is turned on. |
| 6 | C0BG | Analog Comparator 0 Positive Input Source Select bit. C0BG and C0PS0 bits of the C0XR register<br>Set the AC0 positive input source together, {C0BG, C0PS0} =<br>00 = AC0P as positive input<br>01 = ACXP as positive input<br>10 = The output of the internal DAC is input as a positive terminal<br>11 = Turn off the positive input source for AC0 |
| 5 | C0O | Analog comparator output status bit.<br>The analog comparator output is connected directly to the C0O bit after synchronization. Software can read C0O<br>Bit value to get the output value of the analog comparator. |
| 4 | C0I | Analog comparator interrupt flag bit.<br>When the analog comparator output event triggers the interrupt mode defined by the C0IS bit, the C0I<br>Bit is set. Interrupt is generated when interrupt enable bit C0IE is "1" and global interrupt is set.<br>When the analog comparator interrupt service routine is executed, C0I will be cleared automatically or "1" for the C0I bit<br>This bit can also be cleared. |
| 3 | C0IE | Analog comparator interrupt enable bit.<br>When the C0IE bit is set to 1 and the global interrupt is enabled, the interrupt of AC0 is enabled.<br>When setting the C0IE bit to 0, the interrupt of AC0 is disabled. |
| 2 | C0IC | Analog Comparator Input Capture Enable bit<br>C0IC = 1, the input capture source for timer counter 1 is derived from the analog comparator output.<br>C0IC = 0, the input capture source for timer counter 1 is from external pin ICP1. |
| 1 | C0IS1 | Analog Comparator Interrupt Mode Control High. |
| 0 | C0IS0 | Analog Comparator Interrupt Mode Control Low. C0IS0 and C0IS1 together to form C0IS [1: 0],<br>Used to control the analog comparator interrupt trigger mode. |

| C0IS [1: 0] | Interrupt mode |
|---|---|
| 00 | The rising or falling edge of ACO triggers |
| 01 | Keep it. |
| 10 | The falling edge of ACO triggers |
| 11 | The rising edge of ACO triggers |

### ADCSRB - ADC CONTROL AND STATUS REGISTER B

ADCSRB - ADC CONTROL AND STATUS REGISTER B

Address: 0x7B Default: 0x00

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CME01 | CME10 | 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} |
| 6 | CME00 | 00: External port ACXN is input as AC0 negative terminal<br>01: ADC multiplexed output as AC0 negative input<br>10: Differential amplifier output as AC0 negative input<br>11: Turn off the negative input source of AC0 |
| 5 | CME11 | AC1 Negative input selection, CME1 = {CME11, CME10} |
| 4 | CME10 | 00: External port ACXN is input as AC1 negative terminal<br>01: External port AC1N is input as AC1 negative terminal<br>10: ADC internal 1/5 division voltage as AC1 negative input<br>11: Differential op amp output as AC1 negative input |
| 3 | ACHS | AC trigger source channel selection<br>0 - AC0 output as the ADC auto-conversion trigger source<br>1 - AC1 output as the ADC automatic conversion trigger source |
| 2: 0 | ADTS | See ADC Register Description. |

### C0XR - AC0 Auxiliary control register

C0XR - AC0 Auxiliary control register

Address: 0x51 Default: 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 | - | Keep it |
| 6 | C0OE | AC0 comparator output to external port enable control<br>C0OE = 1, AC0 comparator output to the external port PD2<br>C0OE = 0, disables the comparator output to the external port |
| 5 | C0HYSE | AC0 Output Hysteresis Function Enable Control.<br>1 = Enable output hysteresis<br>0 = disable output hysteresis |
| 4 | C0PS0 | AC0 Positive terminal input source Select low.<br>C0PS0 and C0BG common control AC0 positive input source, please refer to the C0SR register set Righteousness |
| 3 | COWKE | AC0 Enables the enable control for sleep wake-up.<br>1 = Enables the wake-up function of the comparator output |



| | | |
|---|---|---|
| | | 0 = Turns off the wake-up function of the comparator output |
| 2 | C0FEN | Comparator digital filter enable control.<br>1 = Enable digital filter<br>0 = disable the 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 more accurate
- Factory misalignment calibration
- Supports 4 off-chip analog inputs
- Supports internal 1/5 divider input (VDO)
- Supports internal differential amplifier input (DFFO)
- Supports internal 8-bit DAC input (DAO)
- Programmable output filter control

### Summary

The analog comparator compares the positive and negative levels of the input comparator. When the positive terminal voltage is higher than the negative terminal voltage, The comparator output ACO is set. When the level of ACO changes, the edge of the signal can be used to trigger an interrupt. Output

The signal ACO can also be used to trigger the input capture of the timer counter 1 and to control the PWM output generated by the timer.

LGT8FX8P integrated analog comparator AC1, including a multi-channel analog input selector, comparator positive and negative input source

To select from external ports or from a variety of internally generated reference sources. The analog comparator itself supports offset calibration, and can

To ensure the consistency of the work of the comparator. The comparator supports an optional hardware hysteresis function to improve the stability of the comparator output

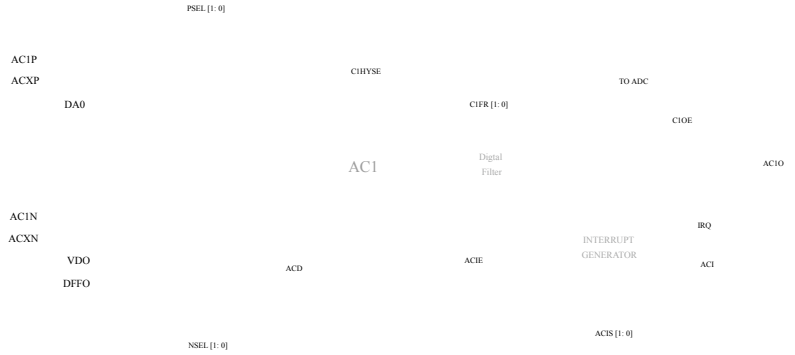
Qualitative. While the comparator output integrated a hardware can be programmed digital filter, according to the application needs, select the appropriate

Of the filter settings to get a more stable comparison output.

Comparator output status can be read directly through the register, can also generate interrupt requests, to achieve more efficient real-time events

Capture function. The output of the comparator can also be output directly to the external IO port.

The structure of the analog comparator 1 is shown in the following figure.




Analog Comparator 1 module structure diagram

## Analog comparator input

Both inputs of the analog comparator support a variety of optional input sources. Positive input three-way optional:

1. External independent analog input AC1P
2. Analog Comparator 0/1 Common Analog Input ACXP
3. Internal 8-bit DAC output DAO



The selection of the input source is controlled by the C1BG bits in the control status register C1SR and the C1PS0 bits of the C1XR register

For details, refer to the section description of the registers in this section.

AC1P is AC1 dedicated positive mode input channel.

ACXP is a positive input for comparator 0/1. LGT8FX8P internal two analog comparators, ACXP connected to the same time

The two comparators of the positive multiplexing selector, easy to achieve the two comparators work together.

DAO from the internal 8-bit DAC output. The DAC reference source can be selected from the system power supply, internal reference or come

Input from external reference. Refer to the DAC section for DAC configuration.

| C1BG | C1PS0 | AC1 positive input |
|---|---|---|
| 0 | 0 | AC1P |
| 0 | 1 | ACXP |
| 1 | 0 | DAO |
| 1 | 1 | Turn off the comparator positive input channel |

Negative input can also choose 4 different analog inputs:

1. External analog input AC1N is input as AC1 negative terminal
2. Comparator 0/1 Common negative input ACXN
3. ADC internal 1/5 divider output as negative input for AC1
4. Internal differential amplifier output DFFO as negative input for AC1

The comparator negative input channel selection is controlled by the CME11 / 10 bits in the ADCSRB register from the ADC module.

When the comparator negative input is selected as the ADC internal multiplexer output, it is necessary to pass the ADCSRC register of the ADC module

The VDS bit selects the input reference source for the multiplexed voltage.

ACXN for the comparator 0/1 common negative input, easy to achieve the comparator 0/1 co-operation;

DFFO from the internal differential amplifier output. Differential amplifier optional x1 / x8 / x16 / x32 gain control, can be achieved small Signal detection and measurement.

| CME11 | CME10 | AC1 negative input |
|---|---|---|
| 0 | 0 | ACXN |
| 0 | 1 | AC1N |
| 1 | 0 | VDO |
| 1 | 1 | DFFO |

### Comparator output filter

The comparator output supports a controlled hysteresis internally. The user can enable the C1HYSE bit in the C1XR register Hysteresis circuit. Hysteresis circuit can eliminate the unstable state of the comparator state change process, to achieve the output filter function.

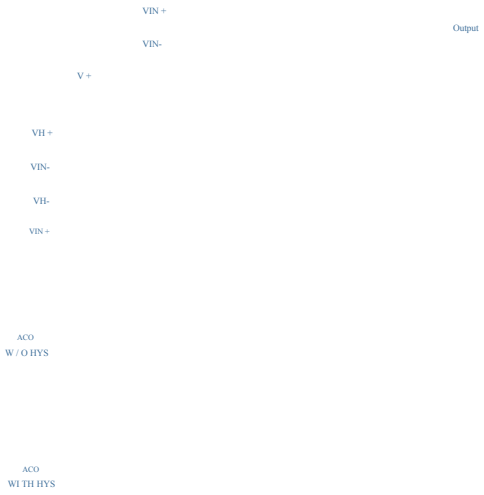
It is recommended that the user use the comparator to turn on the hysteresis circuit and obtain a stable comparator output.

As shown in the following figure, the hysteresis circuit is located between the analog output of the comparator and the digital output. When the comparator is the positive input volta When $V_{IN+}$ is greater than ($V_{IN-} + V_{H+}$), the comparator COUT output is high; when $V_{IN+}$ voltage is less than ($V_{IN-} - V_{H-}$), the comparator output is low. The hysteresis circuit avoids jitter due to the circuit itself when the positive terminal voltage of the comparator approaches the negative terminal voltage.

Comparator hysteresis voltage and comparator output diagram:








Although the hysteresis circuit is very effective for suppressing voltage ripple near the comparator threshold, the actual application environment, The signal will be disturbed by different intensities. Strong interference may cause the input level to instantaneously raise the valve beyond the hysteresis circuit Value range, can not be effectively suppressed. The LGT8FX8P integrates a programmable digital filter at the comparator output. To filter out the effects of instantaneous interference on the output of the comparator. The digital filter can select the appropriate filter according to the application requirements Time width, the filter circuit updates the output of the comparator only when the output of the comparator continues to meet the filter time limit. Thus achieving a more stable output.

**ACO Before Filter**

Comparator output filter timing

AC1 digital filter through the C1XR register C0FEN and C1FS bit control, the specific settings please refer to this chapter Register definition section.

## Comparator output and PWM control

LGT8FX8P supports multi-channel PWM output, PWM signal can be used in conjunction with the comparator module. The input of the comparator Out, can be used to directly turn off the PWM signal, in order to achieve a more flexible PWM protection program.

For control related to PWM output, refer to the relevant section of the timer section.





## Register definition

### C1SR - AC1 control and status register

C1SR - AC1 control and status register

Address: 0x2F Default: 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.<br>When the C1D bit is set to "1", the analog comparator is turned off.<br>When the C1D bit is set to "0", the analog comparator is turned on. |
| 6 | C1BG | Analog Comparator 1 Positive Input Source Select bit. C1BG and C1X0 bits of the C1XR register<br>Together set AC1 positive input source, {C1BG, C1PS0} =<br>00 = AC1P as positive input<br>01 = ACXP as positive input<br>10 = The output of the internal DAC is input as a positive terminal<br>11 = Turn off the positive input source of AC1 |
| 5 | C1O | Analog comparator output status bit.<br>The analog comparator output is connected directly to the C1O bit after synchronization. Software can read C1O<br>Bit value to get the output value of the analog comparator. |
| 4 | C1I | Analog comparator interrupt flag bit.<br>When the analog comparator output event triggers the interrupt mode defined by the C1IS bit, C1I<br>Bit is set. When the interrupt enable bit C1IE is "1" and the global interrupt is set, the interrupt is generated.<br>When the analog comparator interrupt service routine is executed, C1I is automatically cleared, or the C1I bit is written to "1"<br>This bit can also be cleared. |
| 3 | C1IE | Analog comparator interrupt enable bit.<br>When the C1IE bit is set to 1 and the global interrupt is enabled, the interrupt of AC1 is enabled.<br>When setting the C1IE bit to 0, the interrupt of AC1 is disabled. |
| 2 | C1IC | Analog Comparator Input Capture Enable bit<br>C1IC = 1, the input capture source for timer counter 1 is derived from the output of the analog comparator.<br>C1IC = 0, the input capture source for timer counter 1 is from external pin ICP1. |
| 1 | C1IS1 | Analog Comparator Interrupt Mode Control High. |
| 0 | C1IS0 | Analog Comparator Interrupt Mode Control Low. C1IS0 and C1IS1 together constitute C1PS [1: 0],<br>Used to control the analog comparator interrupt trigger mode.<br>C1IS [1: 0] Interrupt mode<br>00 The rising or falling edge of AC1 triggers<br>01 Keep it. |

| | |
|---|---|
| 10 | The falling edge of AC1 triggers |
| 11 | The rising edge of AC1 triggers |





## ADCSRB - ADC CONTROL AND STATUS REGISTER B

ADCSRB - ADC CONTROL AND STATUS REGISTER B

Address: 0x7B Default: 0x00

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | CME01 | CME10 | 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} |
| 6 | CME00 | 00: External port ACXN is input as AC0 negative terminal<br>01: ADC multiplexed output as AC0 negative input<br>10: Differential amplifier output as AC0 negative input<br>11: Turn off the negative input source of AC0 |
| 5 | CME11 | AC1 Negative input selection, CME1 = {CME11, CME10} |
| 4 | CME10 | 00: External port ACXN is input as AC1 negative terminal<br>01: External port AC1N is input as AC1 negative terminal<br>10: ADC internal 1/5 division voltage as AC1 negative input<br>11: Differential op amp output as AC1 negative input |
| 3 | ACHS | AC trigger source channel selection<br>0 - AC0 output as the ADC auto-conversion trigger source<br>1 - AC1 output as the ADC automatic conversion trigger source |
| 2: 0 | ADTS | See ADC Register Description. |

## C1XR - AC1 auxiliary control register

C1XR - AC1 auxiliary control register

Address: 0x3A Default: 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 | - | Keep it |
| 6 | C1OE | AC1 comparator output to external port enable control<br>C1OE = 1, AC1 comparator output to the external port PE5<br>C1OE = 0, disable the comparator output to the external port |
| 5 | C1HYSE | AC1 Output Hysteresis Function Enable Control.<br>1 = Enable output hysteresis<br>0 = disable output hysteresis |
| 4 | C1PS0 | AC1 Positive terminal input source Select low.<br>C1PS0 and C1BG jointly control AC1 positive input source, please refer to C1SR register set<br>Righteousness |
| 3 | C1WKE | AC1 Enables the enable control for sleep wake-up.<br>1 = Enables the wake-up function of the comparator output |

**Page 235**



| | | |
|---|---|---|
| | | 0 = Turns off the wake-up function of the comparator output |
| 2 | C1FEN | Comparator digital filter enable control.<br>1 = Enable digital filter<br>0 = disable the digital filter |
| 1: 0 | C1FS [1: 0] | Comparator digital filter width setting<br>00 = off<br>01 = 32us<br>10 = 64us<br>11 = 96us |



**Page 236**



## Digital-to-analog converter (DAC)

- 8-bit digital-to-analog conversion output
- The DAC output can be used as an analog comparator reference input
- Supports DAC output to external port (DAO)
- Optional VCC / AVREF / IVREF partial voltage power supply

## Summary

The LGT8FX8P integrates an internal 8-bit programmable digital-to-analog converter (DAC). The DAC's reference power input can be selected as From the system power supply, internal reference voltage or from the chip external port AVREF input. DAC output can be selected As an input source for the internal comparator AC0 / 1, it can also be output directly to the external pin of the chip as an external reference. When the DAC is output to an external pin, it can not be used directly to drive the load. It needs to be driven by a voltage follower or other similar drive Moving circuit. DAC internal structure as shown below:


## Register definition

### DACON - DAC control register

DACON-DAC control register

| Address: 0xA0 | | | | | 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 | - | Keep it |
| 3 | DACEN | DAC enable control bit<br>1 = Enables the DAC module<br>0 = disable the DAC module |
| 2 | DAOE | DAC output to external port enable control<br>1 = Enables the DAC to output to the external terminal PD4<br>0 = Disables DAC output to external port |
| 1 | DAVS1 | DAC reference voltage source selection bit 1 |
| 0 | DAVS0 | DAC reference voltage source selection bit 0. [DVS1, DVS0] = |







00: voltage source selection system operating voltage VCC
01: Voltage source selected for external input AVREF
10: The voltage source is selected as the internal reference voltage
11: The DAC reference source is turned off and the DAC module is turned off

### DALR - DAC data register

VRCON1 - DAC1 control register

| Address: 0xA1 | | | | | 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 DAC mode output voltage size<br>DAC output voltage and DALR relationship:<br>$V_{DAO} = V_{REF} * (DALR + 1) / 256$<br>among them:<br>$V_{DAO}$ outputs analog voltage for DAC<br>$V_{REF}$ 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, DNL is ± 1LSB, INL is ± 1.5LSB
- Sampling rate up to 500KSPS at maximum resolution
- 12 multiplexed single-ended input channels
- Multi-input programmable gain differential amplifier channel
- The input voltage range is 0-VCC
- Internal 1.024V / 2.048V / 4.096V reference voltage
- Supports AVCC and external reference voltage input
- Internal multi-input 1 / 5,4 / 5 voltage divider circuit
- Supports offset calibration in both positive and negative directions
- Automatic start conversion mode based on interrupt source
- Automatic channel monitoring for up / down overflow
- Conversion results support optional alignment mode
- Conversion end interrupt request

### Overview


ADC structure diagram

The analog-to-digital converter is a 12-bit successive approximation ADC. The ADC is connected to a 17-channel analog multiplexer , Can be from the chip external port 12 analog input and 5-channel internal voltage source sampling conversion. ADC internal Integrate a differential op amp with programmable gain of x1 / x8 / x16 / x32, the amplifier input can come from an external port or ADC multiplexer output. The result of the differential op amp can be used as an analog input to the ADC.





The ADC's internal analog input source includes multiple input voltage dividers from within the ADC; internal reference voltage source; internal mode The reference ground and the analog output from the touch key module. Internal multi-input input voltage divider output 4/5, 1/5 two Voltage; the input of the voltage divider can be selected from the level of the external port or from the system power supply.

ADC supports offset calibration. The offset calibration process is controlled by software. Offset calibration includes positive and negative calibration in both directions the amount. After the offset calibration is enabled, the ADC controller will automatically calibrate the ADC sampling results using positive and negative calibration values.

Please refer to the relevant section of this section for misalignment calibration.

## ADC operation

The ADC converts the input analog voltage into a 12-bit digital quantity by successive approximation. The minimum value represents GND, The maximum value represents the reference voltage minus 1LSB. The reference voltage source can be the ADC's supply voltage AVCC, the external reference voltage AVREF or internal reference voltage of 1.024V / 2.048V is selected by writing the REFS bit of the ADMUX register.

The analog input channels can be selected by writing the CHMUX bit in the ADMUX register. Any ADC input pin, external base The quasi-voltage pin, and the internal reference voltage source, can be used as a single-ended input to the ADC. By setting the DIFS of the ADTMR register The input channel of the ADC can be switched to an internal differential amplifier. The differential amplifier associated input source and gain can pass DAPCR register settings.

The ADEN bit of the ADCSRA register can be used to start the ADC. When the ADEN is cleared, the ADC does not consume power. Turn on the ADC before entering sleep mode.

The ADC conversion result is 12 bits, which are stored in the ADC data registers ADCH and ADCL. By default, the conversion result is right Alignment, but can be left aligned by setting the ADLAR bit in the ADMUX register.

If you set the conversion result to the left justify, and the maximum requires only 8 bits of conversion accuracy, then just read the ADCH is enough. Otherwise read the ADCL, and then read the ADCH, to ensure that the contents of the data register is the result of the same conversion. once After reading ADCL, the data registers ADCL and ADCH are latched, and the ADCH conversion result is read to the data Registers ADCL and ADCH.

The end of the ADC conversion can trigger an interrupt. Even if the end of the conversion occurs between reading ADCL and ADCH, the interrupt will still fire.

### Start a conversion

Setting the ADSC bit to the ADC bit to write "1" can initiate a single conversion. This bit remains high during conversion until conversion
After the end of the hardware cleared. If the channel is changed during the conversion process, the ADC will complete this time before changing the channel
Conversion.

ADC conversion has a different trigger source. Setting the ADCSRA register's ADC auto-enable enable bit ADATE can be enabled automatically
trigger. Setting the ADCSRB register's ADC trigger selection bit ADTS can select the trigger source. When the selected trigger signal is produced
When the rising edge occurs, the ADC prescaler is reset and starts converting. This provides a way to start the conversion at a fixed time interval
law. Even if the trigger signal persists after the conversion, a new conversion will not be initiated. If triggered during the conversion process
The signal also produces a rising edge, which will also be ignored. Even if a specific interrupt is disabled or a global interrupt is enabled
Bit is set to "0" and its interrupt flag will be set. This triggers a transition without generating an interrupt. But for that
A new conversion is triggered when the next interrupt event occurs, and the interrupt flag must be cleared.





Using the ADC interrupt flag as the trigger source, you can start the next ADC conversion after the current conversion is complete. The
After the ADC will work in continuous conversion mode, continuous sampling and ADC data register to update. First turn
Is changed by writing "1" to the ADSCRA register ADSC bit. In this mode, subsequent ADC conversions do not depend on
The ADC interrupt flag ADIF is set.

If automatic triggering is enabled, the ADSC setting the ADCSRA register will initiate a single conversion. The ADSC logo can also be used for inspection
Whether the conversion is in progress. Regardless of how the conversion was initiated, ADSC was always "1" during the conversion process.

### Prescaler and ADC conversion timing

Under the default conditions, the successive approximation circuit requires an input clock from 300 kHz to 3 MHz for maximum accuracy. Such as
If the required conversion accuracy is less than 12 bits, then the input clock frequency can be higher than 3MHz, in order to achieve a higher sampling rate.

The ADC module includes a prescaler that can generate an acceptable ADC input clock from the system clock. Prescaler
The ADPS bits of the ADCSRA register are set. Setting ADEN of the ADCSRA register will enable the ADC, prescaler
Start counting. As long as the ADEN bit is "1", the prescaler continues counting until ADEN is cleared.

ADSCRA register ADSC is set, the single-ended conversion starts at the rising edge of the next ADC clock cycle. positive
A constant conversion requires 15 ADC clock cycles. ADC enabled (ADEN set in ADCSRA register) requires 50 ADCs
The input clock cycle initializes the analog circuit before it can be used for the first time.

During ADC conversion, the sample is held at 1.5 ADC input clocks after the conversion start, and the first ADC transition
The result of the output is the 14.5 ADC input clock after the start. After the conversion, the ADC result is sent to the ADC
Data register, and the ADIF flag is set. ADSC is cleared at the same time. The software can then set the ADSC flag again
Or automatically trigger, to start a new conversion.

### Sampling channel with reference voltage

MUX and REFS in the ADMUX register are individually buffered through temporary registers. The CPU can be used with the temporary registers
Machine access. Before the switch is started, the CPU can configure the channel and reference source at any time. In order to ensure that the ADC is charged
The sampling time, once the conversion is started, does not allow the channel and the reference source to select the configuration. After the conversion is complete (ADCSRA
Register ADIF is set), the channel and reference source selection will be updated. The start time of the conversion is set for ADSC
After the next ADC enters the rising edge of the clock. Therefore, it is recommended that the user enter the ADSC after an ADC input clock
Do not operate ADMUX to select a new channel and reference source.

When using auto-triggering, the time at which the event occurred is uncertain. In order to control the impact of the new settings on the conversion, the update
ADMUX register should be particularly careful. If both ADATE and ADEN are set, the interrupt time can occur at any time,
Which automatically trigger, start the ADC conversion. If the contents of the ADMUX register are changed during this period, then the user is not available
The next step is based on the old configuration or the new configuration. It is recommended that the user make ADMUX at the following safety times

Update:
1) ADATE or ADEN bit is "0";
2) during the conversion process, but at least one ADC input clock cycle after the trigger event occurs;
3) After the conversion is complete, but before the interrupt flag of the trigger source is cleared.
If you update ADMUX in any of the cases mentioned above, the new configuration will take effect before the next conversion.
When selecting the ADC input channel, note that the channel is selected before the conversion is initiated and an ADC clock after ADSC is set





After the cycle can choose a new analog input channel, but the easiest way is to wait until the end of the conversion and then change the channel.

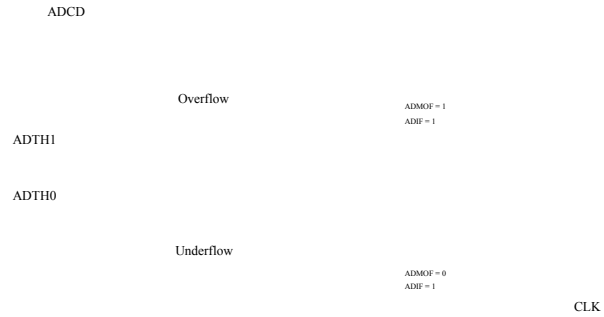
The reference voltage source V ref of the ADC reflects the conversion range of the ADC. If the single-ended channel level exceeds V ref , the conversion result will be
Close to maximum 0xFFF. V ref can be AVCC, external AREF pin voltage, internal reference voltage source.

Use internal reference (1.024V / 2.048V / 4.096V) Note:

After the chip is powered on, the internal benchmark is calibrated to 1.024V by default. If the user uses an internal benchmark of 1.024V,
Direct use, no other operation. But if you need to use 2.048V or 4.096V internal reference voltage, you need to be more self
New internal reference calibration value. The calibration value of 2.048V / 4.096V is loaded into register VCAL2 / 3 (0xCE / 0xCC) after power-up,
When the program is initialized, the value of VCAL2 / 3 is read in and written to 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. Software by setting ADCSRC
The AMEN bit of the register enables the automatic channel monitoring function. The ADC automatically converts the voltage of the selected channel. When the conversion result is
Out of the overflow range, the ADC interrupt flag (ADIF) will be set and the automatic monitoring will be stopped at the same time. The software can pass
Interrupt or query the way to respond to overflow events. The AMOF bit of the ADMSC register is used to indicate the type of overflow event. ADIF
The flag bit is automatically cleared by hardware after a reset interrupt. In query mode, it can be cleared by software. Only when ADIF
Clear and enable the automatic monitoring mode by setting the AMEN bit in the ADCSRC register.




To overcome the instability of single ADC conversion results, automatic detection supports a configurable digital filtering function. Digital filter
The wave is tested by the continuous conversion result, and only a consistent result is obtained within the defined number of consecutive conversions.
Triggering an overflow event. The number of consecutive conversions can be set via the AMFC [3: 0] bits of the ADMSC register.

The automatic channel monitoring function is controlled by the AMEN bit of the ADCSRC register. The register ADT0 is used to set the next overflow
The ADT1 is used to set the overflow threshold. ADT0 / 1 is a 16-bit register. After the software sets the AMEN bit, it will
Immediately stop the ADC's current conversion action, and reset the ADC control state, then enter the automatic conversion mode.

Before starting the automatic channel detection mode, you need to set the channel to be detected and other related configurations. Software can be done at any time
The auto detect mode is disabled by clearing the AMEN register.

### Multi-input voltage divider circuit (VDS)

The ADC contains a multi-input divider module. The divider input voltage source is available from the external ADC input channel
(ADC0 / 1/4/5), external reference AVREF or analog power supply. The divider module outputs both 4/5 and 1/5
The voltage is applied to the ADC's internal 12,13 input channels, respectively. Which 4/5 this way for ADC offset calibration; 1/5 in addition to use
In the internal offset calibration, and more for power supply voltage detection and other similar applications. The function of the voltage divider circuit is mainly sent by ADCSRD

Register control implementation.

### ADC offset calibration

Due to the deviation of the manufacturing process and the inherent characteristics of the circuit structure, will cause the ADC internal comparator circuit to produce different Degree of misalignment error. So the offset voltage compensation, for the production of high-precision ADC conversion structure is critical.
LGT8FX8P chip internal ADC support offset voltage test related interface, can be completed in the software with the offset test
Quantity and calibration.

The principle of offset calibration:

Offset calibration is done by changing the input polarity of the internal comparator, testing the ADC conversion junction in both positive and negative directions fruit. As the positive and negative direction of the two offset voltage is also expressed as two kinds of polarity, through the two conversion results subtraction, you can get An intermediate offset error value. Normal application, the conversion results according to the offset voltage can be adjusted accordingly.

Offset calibration process:

1. Configure the VDS module, select the VDS input source as an analog power supply (AVCC)
2. ADC reference voltage selection for analog power supply (AVCC)
3. ADCSRC [SPN] = 0, ADC reads 4 / 5VDO channel, conversion value is recorded as PVAL
4. ADCSRC [SPN] = 1, ADC read 4 / 5VDO channel, conversion value record bit NVAL
5. Store the value (NVAL - PVAL) >> 1 into the OFR0 register
6. ADCSRC [SPN] = 1, ADC reads 1 / 5VDO channel, conversion result is recorded as NVAL
7. ADCSRC [SPN] = 0, ADC read 1 / 5VDO channel, conversion result record bit PVAL
8. Store the value (NVAL - PVAL) >> 1 into the OFR1 register
9. Set ADCSRC [OFEN] = 1 to enable offset compensation

Special attention: As the offset error has positive and negative direction, the above data and operations are signed operation.

The offset calibration process needs to change the ADC-related configuration, so it is recommended that the offset calibration be done before the normal configuration to make. In order to improve the calibration accuracy, it is recommended that the ADC read channel conversion multiple sampling.

Offset Calibration After the OFR0 / 1 configuration is completed, auto offset compensation is enabled via the OFEN bit. After the normal conversion, ADC control will automatically use OFR0 / 1 to compensate for ADC conversion results.

### ADC dynamic calibration

The offset calibration method described above is based on an imbalance in a test environment and test input. When the system environment changes , The ADC offset will also change. So if real-time calibration compensation can be achieved, to overcome the device with the work loop Environment changes caused by the performance differences, improve the accuracy of ADC measurement, is very important.

Here is a proposed algorithm used, based on the principle of offset calibration algorithm, can achieve dynamic compensation for the work environment Bring the wrong error, get consistent and accurate test results.

This method eliminates the need to calculate the offset voltage and does not require offset compensation (OFEN). The algorithm only needs to be controlled by SPN ADC conversion polarity, in the different SPN sampling of the two measurements, the results of the two errors due to the introduction of the error Positive and negative direction, so we can simply add the average method to offset the error caused by the offset.

We assume that when the ADC conversion, the imbalance introduced the test error for the VOFS, so control the SPN for two consecutive ADC conversion, the resulting ADC conversion results can be expressed as:

When SPN = 1, $V_{ADC1} = V_{REL} + V_{OFS1}$

When SPN = 0, $V_{ADC0} = V_{REL} - V_{OFS0}$

We add the two measurements to eliminate the effect of $V_{OFS}$ on the actual sample input $V_{REL}$ . Due to the circuit , The $V_{OFS1}$ and $V_{OFS0}$ may not be identical, but the effect of compensating for the offset error can still be achieved on the whole.







Dynamic offset compensation algorithm flow:

1. Initialize the ADC conversion parameters according to the application
2. Set SPN = 1 to start ADC sampling and record ADC sampling result as VADC1

3. Set SPN = 0 to start ADC sampling and record ADC sampling result as VADC2
4. (VADC1 + VADC2) >> 1 is the conversion result of this ADC

In practice, this algorithm can be combined with the sampling average algorithm, you can get more ideal results.

## Register definition

ADC register list

| register | address | Defaults | 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 multiple select control register |
| ADCSRC | 0x7D | 0x01 | ADC control and status register C |
| DIDR0 | 0x7E | 0x00 | Digital input disable control register 0 |
| DIDR1 | 0x7F | 0x00 | Digital input disable control register 0 |
| DAPCR | 0xDC | 0x00 | Differential amplifier control register |
| OFR0 | 0xA3 | 0x00 | Offset compensation register 0 |
| OFR1 | 0xA4 | 0x00 | Offset compensation register 1 |
| ADT0L | 0xA5 | 0x00 | Automatic monitoring of the underflow threshold is 8 bits low |
| ADT0H | 0xA6 | 0x00 | Automatic monitoring of the underflow threshold is 8 bits high |
| ADT1L | 0xAA | 0x00 | Automatic monitoring of the overflow threshold is 8 bits low |
| ADT1H | 0xAB | 0x00 | Automatic monitoring overflow threshold is 8 bits high |
| ADMSC | 0xAC | 0x01 | Automatic monitoring of status 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: 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.<br>When the ADLAR bit is "0", the ADC output data is stored in the register by the low pair<br>The ADCL is ADC [7: 0], as shown by Name0; when the ADLAR bit is "1" |







ADC output data stored in the register by high-bit alignment, that is, the high 4-bit ADCL
ADC [3: 0], 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: 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 low byte register.<br>When the ADLAR bit is "0", the ADC output data is stored in the register by the low pair<br>The lower 4 bits of ADCH are ADC [11: 8], and the upper 4 bits are meaningless, such as Name0<br>; When the ADLAR bit is "1", the ADC output data is stored in the register<br>High alignment, ie 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: 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.<br>When the ADEN bit is set to "1", the ADC is enabled.<br>When the ADEN bit is set to "0", the ADC is disabled. |
| 6 | ADSC | The ADC starts to convert.<br>In the single conversion mode, ADSC is set to initiate a conversion. In the continuous conversion mode<br>The ADSC setting will start the first conversion. |
| 5 | ADATE | ADC auto trigger enable bit.<br>When the ADATE bit is set to "1", the auto trigger function is enabled. Selected trigger letter<br>The rising edge of the number is turned on. The trigger source is selected by the ADCSRB register<br>ADTS to control.<br>When the ADATE bit is set to "0", the auto trigger function is disabled. |
| 4 | ADIF | ADC interrupt flag.<br>Set ADIF when the ADC completes a conversion and updates the data register. If in the ADC<br>The enable bit is set to "1" and the global interrupt is set, and the ADC interrupt is generated. Execute the ADC<br>The interrupt clears the ADIF bit, or it can be cleared by writing "1" to this bit. |
| 3 | ADIE | ADC interrupt enable control bit. |





| | | |
|---|---|---|
| | | The ADC interrupt is enabled when the ADIE bit is set to "1" and the global interrupt is set.<br>When the ADIE bit is set to "0", the ADC interrupt is disabled. |
| 2: 0 | ADPS [2: 0] | ADC prescaler selection control bit.<br>ADPS Selects the system clock to generate the prescaler factor for 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: 0x00

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ACME01 | ACME00 | ACME1 | ACME10 | ACTS | ADTS2 | ADTS1 | ADTS0 |

| | | | 1 | | | | | |
|---|---|---|---|---|---|---|---|---|
| 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 ACME01 Comparator 0 Negative input selection

6 ACME00 00: Negative Select External Input ACIN0

01: Negative Select ADC multiplexed output

1X: Negative Selects the output of op amp 0

5 ACME11 Comparator 1 Negative input selection

4 ACME10 00: Negative Select External Input ACIN2

01: Negative Select ADC multiplexed output

1X: Negative Selects the output of op amp 1

3 ACTS AC trigger source channel selection

0 - AC0 output as the ADC auto-conversion trigger source

1 - AC1 output as the ADC automatic conversion trigger source

2: 0 ADTS [2: 0] ADC auto trigger source selection control bit.

When the ADATE bit is set to "1", the auto trigger function is enabled and the trigger source is selected by ADTS

To control. When setting the ADATE bit to "0", the ADTS setting is invalid. Selected trigger letter

The interrupt flag on the rising edge of the switch is turned on. When the trigger source is cleared from an interrupt flag

Switching to the trigger source set by the interrupt flag causes the trigger signal to generate a rising edge if this

When ADEN is set, the ADC will also turn on 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 overflows |
| 5 | Timer Counter 1 Compare Match B |
| 6 | Timer counter 1 overflows |
| 7 | Timer Counter 1 Enter the capture event |

### ADMUX - ADC Multiplex Select Control Register

ADMUX - ADC Multiplex Select Control Register

Address: 0x7C Default: 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] REFS2 with the ADCSRD register is used to select the reference voltage source for the ADC

The REFS control bit is used to select the reference voltage, if REFS is changed during the conversion process

Of the settings, only to wait until the current conversion after the end of the change will work.

| 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 alignment enable control bit.<br>When the ADLAR bit is set to "1", the conversion result is left in the ADC data register Qi.<br>When the ADLAR bit is set to "0", the conversion result is right in the ADC data register Qi. |

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 | External port input |
| 0_0100 | PC4 | |
| 0_0101 | PC5 | |
| 0_0110 | PE1 | |





| | |
|---|---|
| 0_0111 | PE3 |
| 0_1001 | PC7 |
| 0_1010 | PF0 |
| 0_1011 | PE6 |
| 0_1100 | PE7 |
| 0_1110 | 4 / 5VDO |
| 0_1000 | 1 / 5VDO |
| 0_1101 | IVREF |
| 0_1111 | AGND |
| 1_XXXX | DACO |

Internal voltage divider circuit

Internal reference

Analog ground

Internal DAC output

ADCSRC - ADC Control Status Register C

ADCSRC - ADC Control Status Register C

Address: 0x7D Default: 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 = disable offset compensation; 0 = off offset compensation |
| 6 | - | Unimplemented |
| 5 | SPN | ADC conversion input polarity control, only for offset calibration process. Normal must be clear zero |
| 4 | AMEN | Channel automatic monitoring enabled;<br>1: Enable channel auto-monitoring function<br>0: Disable channel auto-monitoring function |
| 3 | - | Unimplemented |
| 2 | SPD | 0 = ADC low speed conversion mode<br>1 = ADC high speed conversion mode, only for low impedance analog input |
| 1 | DIFS | 0 = ADC conversion from ADC multiplexer<br>1 = ADC conversion from internal differential amplifier |
| 0 | ADTM | Test mode, the internal reference voltage is output from the AVREF port |

DIDR0 - Digital input disable control register 0

DIDR0 - Digital input disable control register 0

Address: 0x7E Default: 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 function |
| 6 | PE1D | 1 = Turn off PE1 digital input function |





| | | |
|---|---|---|
| 5 | PC5D | 1 = Turn off PC5 digital input function |
| 4 | PC4D | 1 = Turn off PC4 digital input function |
| 3 | PC3D | 1 = Turn off PC3 digital input function |
| 2 | PC2D | 1 = Turn off PC2 digital input function |
| 1 | PC1D | 1 = Turn off PC1 digital input function |
| 0 | PC0D | 1 = Turn off the PC0 digital input function |

### DIDR1 - Digital input disable control register 1

DIDR1 - Digital input disable control register 1

Address: 0x7F Default: 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 the PD6 digital input function |
| 1 | PD7D | 1 = Turn off the PD7 digital input function |
| 2 | PC7D | 1 = Turn off PC7 digital input function |
| 3 | PF0D | 1 = Turn off the PF0 digital input function |
| 4 | C0PD | 1 = off AC0P digital input function (LQFP48) |
| 5 | PE0D | 1 = Turn off the PE0 digital input function |
| 6 | PE6D | 1 = Turn off PE6 digital input function |
| 7 | PE7D | 1 = Turn off PE7 digital input function |

### ADCSRD - ADC Control Register D

ADCSRD - ADC Control Register D

Address: 0xAD Default: 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 | The REFS combination with the ADMUX register is used to select the reference voltage for the ADC conversion<br>Refer to the REFS definition in the ADMUX register |
| 5: 4 | IVSEL | When the reference voltage of the ADC is selected as VCC or AVREF, the IVSEL is used to control the internal reference output<br>Voltage:<br>00 = 1.024V<br>01 = 2.048V<br>1x = 4.096V |
| 3 | - | Keep it |
| 2: 0 | VDS | [2: 0] Divider circuit input source selection<br>000/111 = Turn off the voltage divider circuit module<br>001 = ADC0 |

010 = ADC1
011 = ADC4
100 = ADC5
101 = External reference input (AVREF)
110 = System power supply

### DAPCR - Differential Op Amp Control Register

DAPCR - Differential Op Amp Control Register

Address: 0xDC Default: 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 = enable differential amplifier; 0 = turn off differential amplifier

6: 5 GA [1: 0] Differential Amplifier Gain Control
00 = x1
01 = x8
10 = x16
11 = x32

4: 2 DNS [2: 0] Differential Amplifier Reverse Input Input Source Select bit
000 = ADC2 / APN0
001 = ADC3 / APN1
010 = ADC8 / APN2
011 = ADC9 / APN3
100 = PE0 / APN4
101 = ADC multiplexed
110 = AGND
111 = OFF differential amplifier reverse input

1: 0 DPS [1: 0] Differential Amplifier Positive Input Input Source Select bit
00 = ADC multiplexed
01 = ADC0 / APP0
10 = ADC1 / APP1
11 = AGND

### OFR0 - Offset compensation register 0

OFR0 - Offset compensation register 0

Address: 0xA3 Default: 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: 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 two's complement format |

ADMSC - ADC channel monitoring status control register

ADMSC - ADC channel monitoring status control register

Address: 0xAC Default: 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 of overflow event type flag bits; 1 = overflow, 0 = underflow |
| 6: 4 | - | Unimplemented |
| 3: 0 | AMFC | Automatic monitoring of digital filter control bits:<br>0000 = Disable configuration<br>0001 = one conversion, no filter<br>0010 = twice consistent<br>0011 = three consecutive consecutive<br>...<br>1110 = 14 consecutive consecutive<br>1111 = 15 consecutive consecutive |

ADT0L - Automatically monitors the underflow threshold low of 8 bits

ADT0L - Automatically monitors the underflow threshold low of 8 bits

Address: 0xA5 Default: 0x00

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ADT0L [7: 0] | | | | | | | |
| R / W | W / R | | | | | | | |

| Bit | Name | description |
|---|---|---|
| 7: 0 | ADT0L | Automatic monitoring underflow threshold register low 8 bits |



ADT0H - Automatic monitoring of the underflow threshold 8 bits high

ADT0H - Automatic monitoring of the underflow threshold 8 bits high

Address: 0xA6 Default: 0x00

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ADT0H [7: 0] | | | | | | | |
| R / W | W / R | | | | | | | |

| Bit | Name | description |
|---|---|---|
| 7: 0 | ADT0H | Automatic monitoring underflow threshold register high 8 bits |

ADT1L - Automatic monitoring of overflow threshold 8 bits low

ADT0L - Automatically monitors the overflow threshold at least 8 bits

Address: 0xAA Default: 0x00

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ADT1L [7: 0] | | | | | | | |
| R / W | W / R | | | | | | | |

| Bit | Name | description |
|---|---|---|
| 7: 0 | ADT1L | Automatic monitoring of the upper overflow threshold register is low for 8 bits |

### ADT1H - Automatically monitors overflow thresholds up to 8 bits

ADT1H - Automatically monitors overflow thresholds up to 8 bits

Address: 0xAB Default: 0x00

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Name | ADT1H [7: 0] | | | | | | | |
| R / W | W / R | | | | | | | |

| Bit | Name | description |
|---|---|---|
| 7: 0 | ADT1H | Automatic monitoring of upper overflow threshold register high 8 bits |

### VCAL - internal reference calibration register

VCAL - internal reference calibration register

Address: 0xC8 Default: 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 default value of 1.024V is loaded after power-up.<br>A calibration value of the other reference voltage is written to this register to enable calibration of the relevant reference.<br>For example, after the reference configuration is 2.048V, VCAL2 is written to the change register to complete the 2.048V<br>Internal reference calibration. |



**Page 252**



### VCAL1 - 1.024V reference calibration register

VCAL1 - 1.024V internal reference calibration register

Address: 0xCD Default: 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: 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 factor |

VCAL3- 4.096V reference calibration register

VCAL1 - 4.096V internal reference calibration register

Address: 0xCC Default: 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 factor |



## Register lookup table

| 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 |



**Page 255**



| 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 | | | | | | | |

**Page 256**



| 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 | | | | | | | |



**Page 257**



| 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

| instruction | Operands | description | operating | Mark bit | cycle |
|---|---|---|---|---|---|
| Arithmetic logic instructions | | | | | |
| ADD | $R_d$ , $R_r$ | Register is added | $R_d \leftarrow R_d + R_r$ | Z, C, N, V, H | 1 |
| ADC | $R_d$ , $R_r$ | Supports with carry registers | $R_d \leftarrow R_d + R_r + C$ | Z, C, N, V, H | 1 |
| ADIW | $R_{dh}$ , K | Immediate additions to words | $R_{dh} : R_{dl} \leftarrow R_{dh} : R_{dl} + K$ | Z, C, N, V, S | 1 |
| SUB | $R_d$ , $R_r$ | Register addition and subtraction | $R_d \leftarrow R_d - R_r$ | Z, C, N, V, H | 1 |
| SUBI | $R_d$ , K | Register decrements | $R_d \leftarrow R_d - K$ | Z, C, N, V, H | 1 |

| | | | | | |
|---|---|---|---|---|---|
| SBC | $R_d$, $R_r$ | The register with the borrow is added and subtracted | $R_d \leftarrow R_d - R_r - C$ | Z, C, N, V, H | 1 |
| SBCI | $R_d$, K | The register with borrow is decremented | $R_d \leftarrow R_d - K - C$ | Z, C, N, V, H | 1 |
| SBIW | $R_{dh}$, K | Immediate subtraction with words | $R_{dh} : R_{dl} \leftarrow R_{dh} : R_{dl} - K$ | Z, C, N, V, S | 1 |
| AND | $R_d$, $R_r$ | Logical and | $R_d \leftarrow R_d \& R_r$ | Z, N, V | 1 |
| ANDI | $R_d$, K | Register logic and constants | $R_d \leftarrow R_d \& K$ | Z, N, V | 1 |
| OR | $R_d$, $R_r$ | Logical or | $R_d \leftarrow R_d \mid R_r$ | Z, N, V | 1 |
| ORI | $R_d$, K | Register logic or constant | $R_d \leftarrow R_d \mid K$ | Z, N, V | 1 |
| EOR | $R_d$, $R_r$ | Register exclusive OR | $R_d \leftarrow R_d \oplus R_r$ | Z, N, V | 1 |
| Com | $R_d$ | Reverse code | $R_d \leftarrow \$ FF - R_d$ | Z, C, N, V | 1 |
| NEG | $R_d$ | 2 forcibly | $R_d \leftarrow \$ 00 - R_d$ | Z, C, N, V, H | 1 |
| SBR | $R_d$, K | Set the bits in the register | $R_d \leftarrow R_d \vee K$ | Z, N, V | 1 |
| CBR | $R_d$, K | The bits in the clear register | $R_d \leftarrow R_d \vee (\$ FF - K)$ | Z, N, V | 1 |
| INC | $R_d$ | Increment | $R_d \leftarrow R_d + 1$ | Z, N, V | 1 |
| DEC | $R_d$ | Diminished | $R_d \leftarrow R_d - 1$ | Z, N, V | 1 |
| TST | $R_d$ | The test is 0 or negative | $R_d \leftarrow R_d \& R_d$ | Z, N, V | 1 |
| CLR | $R_d$ | Clear register | $R_d \leftarrow R_d \oplus R_d$ | Z, N, V | 1 |
| SER | $R_d$ | The register is all set to 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 | $R_d$, $R_r$ | Symbolic multiplication | $R_1 : R_0 \leftarrow R_d \times R_r$ | Z, C | 1 |
| MULSU | $R_d$, $R_r$ | Signed number of unsigned numbers | $R_1 : R_0 \leftarrow R_d \times R_r$ | Z, C | 1 |
| FMUL | $R_d$, $R_r$ | Unsigned multiplication | $R_1 : R_0 \leftarrow (R_d \times R_r) << 1$ | Z, C | 1 |
| FMULS | $R_d$, $R_r$ | Signed multiplication, shift | $R_1 : R_0 \leftarrow (R_d \times R_r) << 1$ | Z, C | 1 |
| FMULSU | $R_d$, $R_r$ | Signed number of unsigned numbers, shifted | $R_1 : R_0 \leftarrow (R_d \times R_r) << 1$ | Z, C | 1 |
| Jump instruction | | | | | |
| RJMP | K | Relative jump | PC ← PC + K + 1 | None | 1 |
| IJMP | | Indirect jump (to Z point to address) | PC ← Z | None | 2 |
| JMP | K | Direct jump | PC ← K | None | 2 |
| RCALL | K | Relative address subroutine call | PC ← PC + K + 1 | None | 1 |
| ICALL | | Indirect subroutine call (Z point to address) | PC ← Z | None | 2 |
| CALL | K | Direct subroutine call | PC ← K | None | 2 |
| RET | | Subroutine returns | PC ← Stack | None | 2 |
| RETI | | Interrupt the return | PC ← Stack | I | 2 |





| instruction | Operands | description | operating | Mark bit | cycle |
|---|---|---|---|---|---|
| Jump instruction (continued) | | | | | |
| CPSE | $R_d$, $R_r$ | Equal jump | If ($R_d = R_r$) PC ← PC + 2 or 3 | None | 1/2 |
| CP | $R_d$, $R_r$ | Comparison | $R_d - R_r$ | Z, N, V, C, H | 1 |
| The | $R_d$, $R_r$ | With carry comparison | $R_d - R_r - C$ | Z, N, V, C, H | 1 |
| CPI | $R_d$, K | Compare with immediate data | $R_d - K$ | Z, N, V, C, H | 1 |
| SBRC | $R_r$, b | The bit is 0 to skip the next instruction | If ($R_r$ (b) = 0) PC ← PC + 2 or 3 | None | 1/2 |
| SBRS | $R_r$, b | Bit 1 skip the next instruction | If ($R_r$ (b) = 1) PC ← PC + 2 or 3 | None | 1/2 |
| SBIC | P, b | The I / O bit is 0 to skip the next instruction | If (P (b) = 0) PC ← PC + 2 or 3 | None | 1/2 |
| SBIS | P, b | The I / O bit is 1 to skip the next instruction | If (P (b) = 1) PC ← PC + 2 or 3 | None | 1/2 |
| BRBS | s, k | The status flag is 1 to jump | If (SREG (S) = 1) PC ← PC + K + 1 | None | 1/2 |
| BRBC | s, k | The status flag is 0 to jump | If (SREG (S) = 0) PC ← PC + K + 1 | None | 1/2 |
| BREQ | k | Equal jump | if (Z = 1) then PC ← PC + k + 1 | None | 1/2 |
| BRNE | k | Ranging from jumping | if (Z = 0) then PC ← PC + k + 1 | None | 1/2 |
| BRCS | k | Carry is jump | if (C = 1) then PC ← PC + k + 1 | None | 1/2 |
| BRCC | k | No carry jump | if (C = 0) then PC ← PC + k + 1 | None | 1/2 |
| BRSH | k | Not less than the jump | if (C = 0) then PC ← PC + k + 1 | None | 1/2 |
| BRLO | k | Less than jump | if (C = 1) then PC ← PC + k + 1 | None | 1/2 |

| | | | | | |
|---|---|---|---|---|---|
| BRMI | k | Jump for negative | if (N = 1) then PC ← PC + k + 1 | None | 1/2 |
| BRPL | k | For the regular jump | if (N = 0) then PC ← PC + k + 1 | None | 1/2 |
| BRGE | k | Signed not to jump | if (N ⊕ V = 0) then PC ← PC + k + 1 | None | 1/2 |
| BRLT | k | Signed less than 0 jumps | if (N ⊕ V = 1) then PC ← PC + k + 1 | None | 1/2 |
| BRHS | k | Half carry for 1 jump | if (H = 1) then PC ← PC + k + 1 | None | 1/2 |
| BRHC | k | Half carry is 0 to jump | if (H = 0) then PC ← PC + k + 1 | None | 1/2 |
| BRTS | k | T is set to jump | if (T = 1) then PC ← PC + k + 1 | None | 1/2 |
| BRTC | k | T is cleared | if (T = 0) then PC ← PC + k + 1 | None | 1/2 |
| BRVS | k | Overflow jumps | f (V = 1) then PC ← PC + k + 1 | None | 1/2 |
| BRVC | k | Do not overflow is jump | f (V = 0) then PC ← PC + k + 1 | None | 1/2 |
| BRIE | k | The global interrupt is enabled | f (I = 1) then PC ← PC + k + 1 | None | 1/2 |
| BRID | k | The global interrupt is disabled | 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 a word of data | Rd + 1: Rd ← Rr + 1: Rr | None | 1 |
| LDI | Rd, K | Load Immediate | Rd ← K | None | 1 |
| LD | Rd, X | Indirect loading | Rd ← (X) | None | 1/2 |
| LD | Rd, X + | Indirect load, address increment | Rd ← (X), X ← X + 1 | None | 1/2 |
| LD | Rd, -X | The address is decremented and loaded indirectly | X ← X - 1, Rd ← (X) | None | 1/2 |
| LD | Rd, Y | Indirect loading | Rd ← (Y) | None | 1/2 |
| LD | Rd, Y + | Indirect load, address increment | Rd ← (Y), Y ← Y + 1 | None | 1/2 |
| LD | Rd, -Y | The address is decremented and loaded indirectly | 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 load, address increment | Rd ← (Z), Z ← Z + 1 | None | 1/2 |
| LD | Rd, -Z | The address is decremented and loaded indirectly | 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 storage, address increment | (X) ← Rr, X ← X + 1 | None | 1 |
| ST | -X, Rr | The address is decremented and stored indirectly | X ← X - 1, (X) ← Rr | None | 1 |
| ST | Y, Rr | Indirect storage | (Y) ← Rr | None | 1 |
| ST | Y +, Rr | Indirect storage, address increment | (Y) ← Rr, Y ← Y + 1 | None | 1 |
| ST | -Y, Rr | The address is decremented and stored indirectly | Y ← Y - 1, (Y) ← Rr | None | 1 |
| STD | Y + q, Rr | Indirect storage with offset | (Y + q) ← Rr | None | 1 |
| ST | Z, Rr | Indirect storage | (Z) ← Rr | None | 1 |
| ST | Z +, Rr | Indirect storage, address increment | (Z) ← Rr, Z ← Z + 1 | None | 1 |
| ST | -Z, Rr | The address is decremented and stored indirectly | Z ← Z - 1, (Z) ← Rr | None | 1 |
| STD | Z + q, Rr | Indirect storage with offset | (Z + q) ← Rr | None | 1 |
| STS | k, Rr | Directly stored in the SRAM | (k) ← Rr | None | 2 |
| LPM | | Load the program space data | R0 ← (Z) | None | 2 |
| LPM | Rd, Z | Load the program space data | Rd ← (Z) | None | 2 |
| LPM | Rd, Z + | Load the program data, the address is incremented | Rd ← (Z), Z ← Z + 1 | None | 2 |
| LD | Rd, Z + | Indirect load, address increment | Rd ← (Z), Z ← Z + 1 | None | 1 |
| LD | Rd, -Z | The address is decremented and loaded indirectly | 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 the stack | STACK ← Rr | None | 1 |
| POP | Rd | Out of stack | Rd ← STACK | None | 1/2 |

| | | | | | |
|---|---|---|---|---|---|
| SBI | P, b | Set the IO register | I / O (P, b) ← 1 | None | 1 |
| CBI | P, b | Clear the IO register | I / O (P, b) ← 0 | None | 1 |
| LSL | Rd | Logic left shift | Rd (n + 1) ← Rd (n), Rd (0) ← 0 | Z, C, N, V | 1 |
| LSR | Rd | Logically shifted right | Rd (n) ← Rd (n + 1), Rd (7) ← 0 | Z | 1 |
| ROL | Rd | The loop containing the carry is shifted left | (0) ← C, Rd (n + 1) ← Rd (n), C ← Rd (7) Z | | 1 |
| ROR | Rd | The loop containing the carry is shifted right | (7) ← C, Rd (n) ← Rd (n + 1), C ← Rd (0) Z | | 1 |
| ASR | Rd | Arithmetic right shift | Rd (n) ← Rd (n + 1), n = 0: 6 | Z | 1 |
| SWAP | Rd | Bit exchange | (3: 0) ← Rd (7: 4), Rd (7: 4) ← Rd (3: 0) None | | 1 |
| BSET | s | Set the status bit | SREG (s) ← 1 | SREG (s) | 1 |
| BCLR | s | Clear status bit | SREG (s) ← 0 | SREG (s) | 1 |
| BST | Rr, b | Stored in the T bit | T ← Rr (b) | T | 1 |
| BLD | Rd, b | Read the T bit into the register | Rd (b) ← T | None | 1 |
| SEC | | Set the carry flag | C ← 1 | C | 1 |
| CLC | | Clear carry mark | C ← 0 | C | 1 |

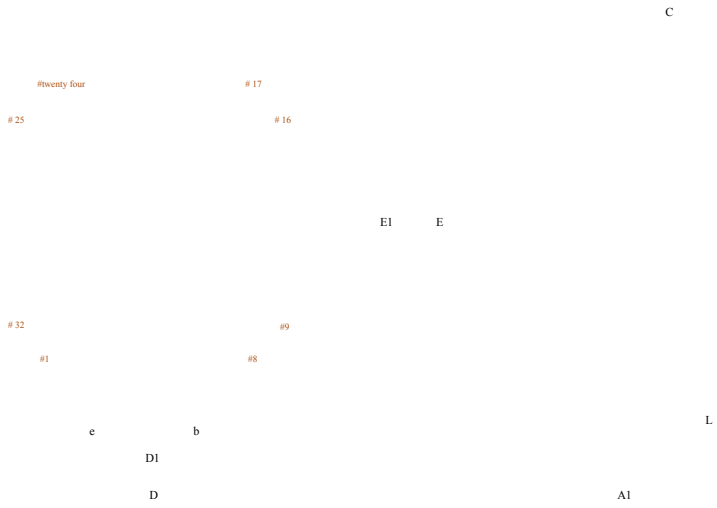


**Page 261**



| | | | | |
|---|---|---|---|---|
| SEN | Set the negative flag | N ← 1 | N. | 1 |
| CLN | Clear negative flag | N ← 0 | N. | 1 |
| SEZ | Set the zero flag | Z ← 1 | Z | 1 |
| CLZ | Clear the zero mark | Z ← 0 | Z | 1 |
| SEI | Enable global interrupts | I ← 1 | I | 1 |
| CLI | Disable global interrupts | I ← 0 | I | 1 |
| SES | Set the symbol test flag | S ← 1 | S | 1 |
| CLS | Clear the symbol test flag | S ← 0 | S | 1 |
| SEV | Set the twos complement overflow flag | V ← 1 | V | 1 |
| CLV | Clear the two'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 | Empty command | | None | 1 |
| SLEEP | Go to sleep mode | | None | 1 |
| WDR | Watchdog reset | | None | 1 |
| BREAK | Soft breakpoint | Only for debugging purposes | None | N / A |
| NOP | Empty command | | None | 1 |
| SLEEP | Go to sleep mode | | None | 1 |

## Encapsulation parameters

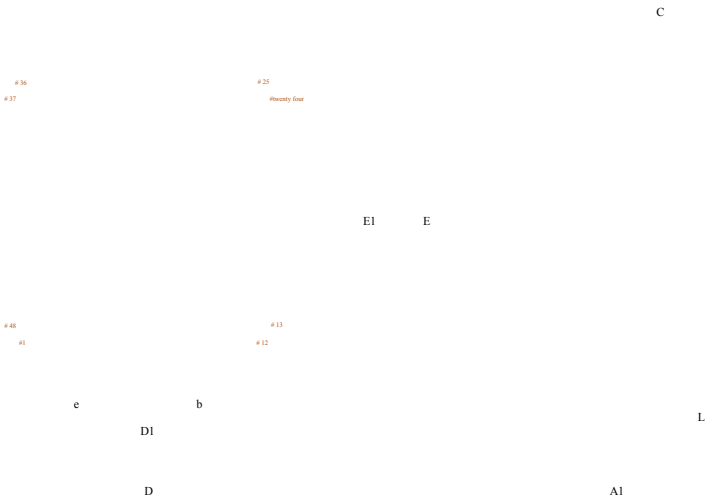



### LQFP32 Universal Dimension Definitions

| Character code | Minimum value | Typical value | The maximum value | 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 |

**Page 263**






LQFP48 Universal size definition

| Character code | Minimum value | Typical value | The maximum value | 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 |



**Page 264**

## Version history

| | |
|---|---|
| V1.0.1<br>2017/2/13 | Delete the I2C1 section, this feature is not available<br>Perfect part of the definition of the register |
| V1.0.0<br>2016/12/29 | initial version |