# CMS80F752x User Manual

**Enhanced flash memory 8-bit 1T 8051 microcontrollers**

**Rev. 0.5.3**

Please note the following CMS IP policy

* China Micro Semicon Co., Ltd. (hereinafter referred to as the Company) has applied for patents and holds absolute legal rights and interests. The patent rights associated with the Company's MCUs or other products have not been authorized for use, and any company, organization, or individual who infringes the Company's patent rights through improper means will be subject to all possible legal actions taken by the Company to curb the infringement and to recover any damages suffered by the Company as a result of the infringement or any illegal benefits obtained by the infringer.

* The name and logo of Cmsemicon are registered trademarks of the Company.

* The Company reserves the right to further explain the reliability, functionality and design improvements of the products in the data sheet. However, the Company is not responsible for the use of the Specification Contents. The applications mentioned herein are for illustrative purposes only and the Company does not warrant and does not represent that these applications can be applied without further modification, nor does it recommend that its products be used in places that may cause harm to persons due to malfunction or other reasons. The Company's products are not authorized for use as critical components in lifesaving, life-sustaining devices or systems. The Company reserves the right to modify the products without prior notice. For the latest information, please visit the official website at www.mcu.com.cn.

# Table of Contents

# 1. Central Processing Unit (CPU)

This series is an 8-bit microcontroller based on the 8051 architecture. The CPU is the core component of the microcontroller, consisting of the arithmetic logic unit (ALU), control unit, and dedicated register bank. The ALU module primarily performs arithmetic and logic operations, bit-variable processing, and data transfer operations. The control unit mainly decodes instructions and generates various control signals. The dedicated register bank is mainly used to represent the memory address of the current instruction to be executed, store operands, and indicate the status after the instruction execution. The dedicated register bank includes the accumulator (ACC), universal register (B), stack pointer (SP), data pointer (DPTR, which is a single DPTR), program status word (PSW), and program counter (PC), among others.

## 1.1 Reset Vector (0000H)

The microcontroller has a word-length system reset vector at address 0000H. After a reset, the program will start executing from address 0000H, and all system registers will be restored to their default values. The following program demonstrates how to define the reset vector in FLASH memory.

Example: Defining the reset vector.

```
        ORG     0000H       ;System reset vector
        LJMP    START
        ORG     0010H       ;User program start
START:
        …                   ;User program
        …
        END                 ;Program end
```

## 1.2 Accumulator (ACC)

The ALU is an 8-bit wide arithmetic logic unit, and all mathematical and logical operations in the MCU are performed through it. It can perform addition, subtraction, shifting, and logical operations on data. The ALU also controls the status bits (in the PSW status register), which are used to indicate the state of the operation result.

The ACC register is an 8-bit register, and the ALU's operation results can be stored here.

## 1.3 B Register (B)

The B register is used when executing multiplication and division instructions. If multiplication or division instructions are not used, it can also be used as a general-purpose register.

## 1.4 Stack Pointer Register (SP)

The SP register points to the address of the stack. After a reset, its default value is 0x07, meaning the stack area starts at RAM address 08H. The value of SP can be modified. For example, if the stack area is set to start at 0xC0, the SP value must be set to 0xBF after a system reset.

Operations that affect the SP include: the instructions PUSH, LCALL, ACALL, POP, RET, RETI, and interrupt entry.

The PUSH instruction occupies one byte on the stack, while LCALL, ACALL, and interrupts occupy two bytes. The POP instruction releases one byte, and the RET/RETI instructions release two bytes.

When the PUSH instruction is used, the current value of the operated register is automatically saved to RAM.

## 1.5 Data Pointer Register (DPTR0)

The data pointer is mainly used in the MOVX and MOVC instructions. Its role is to locate the addresses of XRAM and ROM. The chip has one data pointer register, DPTR0.

Each pointer consists of two 8-bit registers: DPTR0 = {DPH0, DPL0}.

For example, the assembly code to operate on XRAM is as follows:

```
MOV     DPTR,#0001H
MOV     A,#5AH
MOVX    @DPTR,A          ;Write the data in register A to the XRAM address 0001
```

## 1.6 Data Pointer Select Register (DPS)

Data pointer select register DPS

| 0x86 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| DPS | ID1 | ID0 | -- | AU | -- | -- | -- | -- |
| R/W | R | R/W | R | R/W | R | R | R | R |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | | |
|---|---|---|
| Bit7~Bit6 | ID<1:0>: | Decrement/increment function selection |
| | 00= | DPTR0 increments by 1 |
| | 01= | DPTR0 decrements by 1 |
| | Other | Settings are disabled. |
| Bit5 | -- | Reserved, set to 0. |
| Bit4 | AU: | Increment/decrement enable bit |
| | 1= | Enable the MOVX @DPTR or MOVC @DPTR instruction to perform an increment or decrement operation on DPTR (as determined by ID1-ID0). |
| | 0= | DPTR-related instructions do not affect DPTR itself. |
| Bit3~Bit0 | -- | Reserved, set to 0. |

## 1.7 Program Status Word (PSW)

Program status word PSW

| 0xD0 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PSW | CY | AC | F0 | RS1 | RS0 | OV | -- | P |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | | |
|---|---|---|
| Bit7 | CY: | Carry flag bit |
| | 1= | Carry occurred |
| | 0= | No carry |
| Bit6 | AC: | Auxiliary carry flag (half-carry flag) bit |
| | 1= | Carry occurred |
| | 0= | No carry |
| Bit5 | F0: | General-purpose flag bit |
| Bit4~Bit3 | RS<1:0> | The selection bit of the working register BANK. |
| | 00= | Select Bank0 |
| | 01= | Select Bank1 |
| | 10= | Select Bank2 |
| | 10= | Select Bank3 |
| Bit2 | OV: | Overflow flag bit |
| | 1= | Arithmetic or logical operation overflow occurred |
| | 0= | No arithmetic or logical operation overflow |
| Bit1 | -- | Reserved, set to 0. |
| Bit0 | P: | Parity flag |
| | 1= | A carry occurred in the most significant bit of the result |
| | 0= | No carry in the most significant bit of the result. |

## 1.8 Program Counter (PC)

The Program Counter (PC) controls the execution sequence of instructions in program memory (FLASH). It can address the entire range of FLASH, and after fetching an instruction, the PC will automatically increment by one to point to the next instruction. However, if operations like jumps, conditional jumps, subroutine calls, initialization resets, interrupts, interrupt returns, and subroutine returns are performed, the PC will load an address related to the instruction instead of the next instruction's address.

When encountering a conditional jump instruction and the jump condition is met, the next instruction, which would have been fetched during the execution of the current instruction, is discarded, and a dummy instruction cycle is inserted. Only then will the correct instruction be fetched. If the jump condition is not met, the next instruction will be executed sequentially.

## 1.9 Timing Access Register (TA)

Timing access register TA

| 0x96 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| TA | TA7 | TA6 | TA5 | TA4 | TA3 | TA2 | TA1 | TA0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 TA<7:0>: Timing access control bit.

Some protected registers must perform the following operations on TA before they can be written.

```
MOV   TA, #0AAH
MOV   TA, #055H
```

No other instructions can be inserted in the middle, and this sequence needs to be re-executed when it is modified again.

Protected register: WDCON, IREMAP,WDKEY

# 2. Memory and Register Mapping

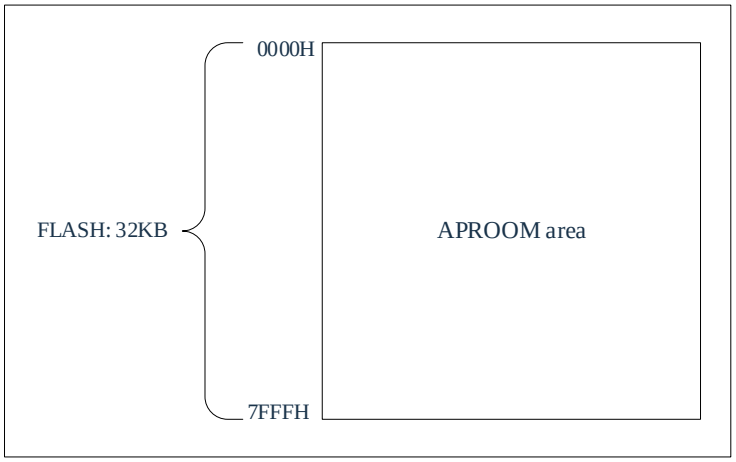
The microcontroller series features several types of memory, including:

- Up to 32KB of FLASH program memory (APROM area)
- Up to 1KB of non-volatile data memory (Data FLASH)
- Up to 256B of general-purpose internal data memory (RAM)
- Up to 2KB of general-purpose external data memory (XRAM)
- Special Function Registers (SFR)
- External Special Function Registers (XSFR)

## 2.1 Program Memory (APROM)

The program memory (APROM) is used to store the source program and lookup table data, with the program counter (PC) serving as the address pointer. Since the PC is a 16-bit program counter, it can address up to a 32KB address space.

The FLASH space allocation structure diagram is shown below:

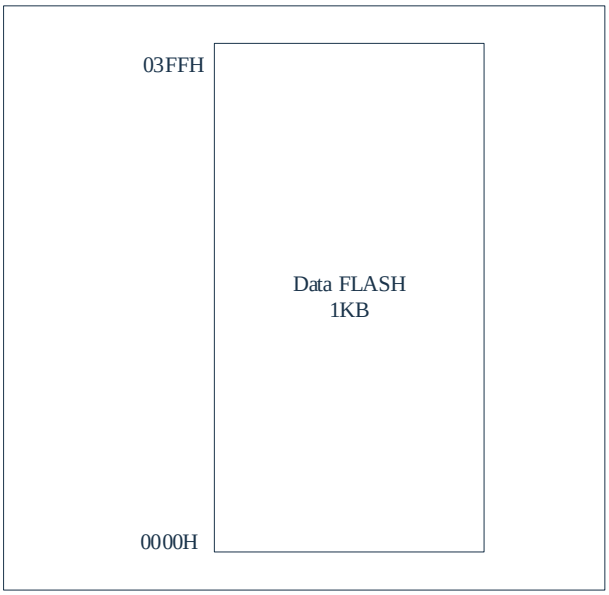



After the chip is reset, the CPU begins execution from address 0000H. Each interrupt is assigned a fixed address in the program memory, and when an interrupt occurs, the CPU jumps to that address to start executing the corresponding service routine.

For example, External Interrupt 1 is assigned the address 0013H. If External Interrupt 1 is used, its service routine must start from the 0013H address. If the interrupt is not used, the service address is treated as a regular program storage address.

## 2.2 Non-Volatile Data Memory (Data FLASH)

The non-volatile data memory (Data FLASH) is used to store important data such as constant values, calibration data, and security-related information. The data stored in this area retains its integrity even when the chip is powered off or experiences sudden, unexpected power loss. The space allocation structure diagram for Data FLASH is shown below:



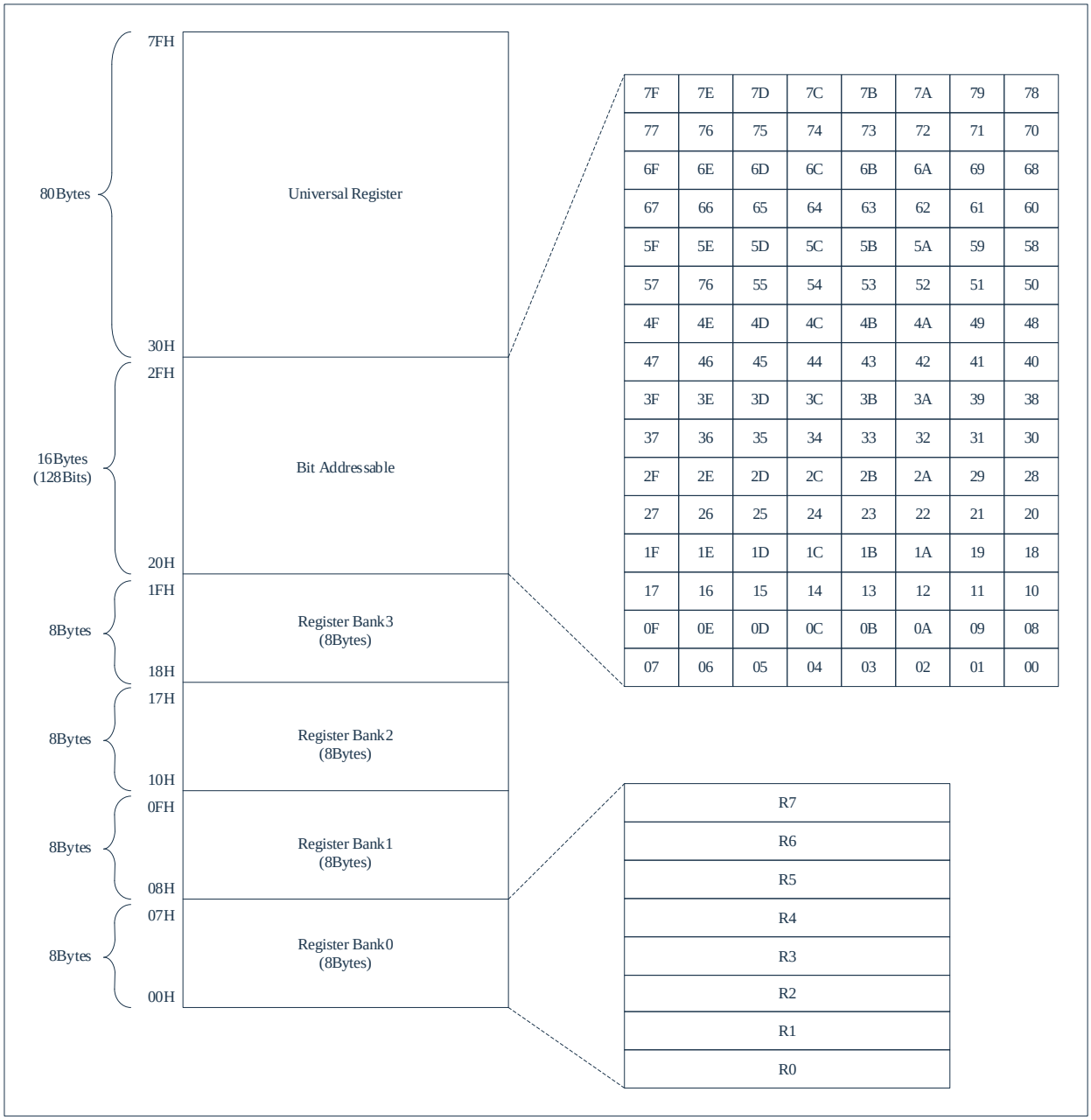

The read, write, and erase operations for the Data FLASH memory are performed through the FLASH control interface.

## 2.3 Universal Data Memory (RAM)

The internal data memory is divided into three sections: lower 128 bytes, higher 128 bytes, and Special Function Registers (SFR). The RAM space allocation structure diagram is shown below:

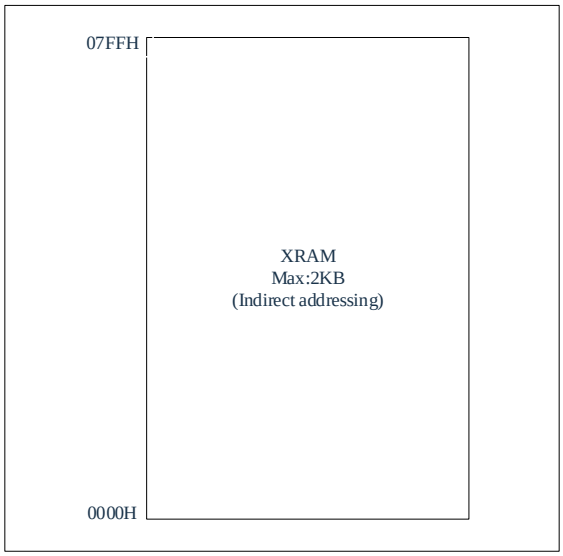

As shown in the diagram, the higher 128 bytes and the SFR occupy the same address range (80H~FFH), but they are independent from each other. Direct addressing of storage above 7FH accesses the SFR, while indirect addressing above 7FH accesses the higher 128 bytes storage, entering different memory spaces.

The lower 128 bytes section is divided as shown in the diagram. The lowest 32 bytes (00H~1FH) form four register banks, each with eight memory units, numbered as R0~R7. These are used for storing operands and intermediate results. After a reset, Bank0 is selected by default. To select other register banks, the program state must be changed accordingly. The 16 bytes following the register banks (20H~2FH) form a bit-addressable memory space. The RAM units in this area can be accessed by byte or used for bit-level operations on individual bits. The remaining 80 memory units (30H~7FH) can be used by the user to set up a stack area or store intermediate data.


7FH
80Bytes
Universal Register
30H
2FH
16Bytes
(128Bits)
Bit Addressable
20H
1FH
8Bytes
Register Bank3
(8Bytes)
18H
17H
8Bytes
Register Bank2
(8Bytes)
10H
0FH
8Bytes
Register Bank1
(8Bytes)
08H
07H
8Bytes
Register Bank0
(8Bytes)
00H
7F 7E 7D 7C 7B 7A 79 78
77 76 75 74 73 72 71 70
6F 6E 6D 6C 6B 6A 69 68
67 66 65 64 63 62 61 60
5F 5E 5D 5C 5B 5A 59 58
57 76 55 54 53 52 51 50
4F 4E 4D 4C 4B 4A 49 48
47 46 45 44 43 42 41 40
3F 3E 3D 3C 3B 3A 39 38
37 36 35 34 33 32 31 30
2F 2E 2D 2C 2B 2A 29 28
27 26 25 24 23 22 21 20
1F 1E 1D 1C 1B 1A 19 18
17 16 15 14 13 12 11 10
0F 0E 0D 0C 0B 0A 09 08
07 06 05 04 03 02 01 00
R7
R6
R5
R4
R3
R2
R1
R0

## 2.4 Universal External Data Register (XRAM)

The chip has a maximum of 2K XRAM area, which is independent of FLASH/RAM. The XRAM space allocation structure diagram is shown below:




XRAM/XSFR space is accessed through the DPTR0 data pointer. For example, the MOVX instruction is used for indirect addressing operations, with assembly code as follows:

```
MOV     R0,#01H
MOV     A,#5AH
MOVX    @R0,A      ;Write the data in A to the XRAM address 01H, the higher 8-bit address is determined by the DPH0.
```

In Keil51, when the Target --> Memory Model is set to Large, the C compiler will use XRAM for variable addressing. Typically, DPTR is used to perform operations on XRAM/XSFR.

## 2.5 Special Function Registers (SFR)

Special Function Registers (SFRs) refer to a collection of registers with specific functions. Essentially, they are special-purpose on-chip RAM units, discretely distributed within the address range 80H to FFH. Users can access them via direct addressing instructions for byte-level access. Registers with lower four address bits of 0000 or 1000 are bit-addressable, such as P0, TCON, P1.

The register table is as follows:

| | 0/8 | 1/9 | 2/A | 3/B | 4/C | 5/D | 6/E | 7/F |
|---|---|---|---|---|---|---|---|---|
| 0xF8 | - | MREGION | MMODE | MDATA | MADRL | MADRH | MSTATUS | MLOCK |
| 0xF0 | B | I2CCON | I2CCLR | I2CSTAT | I2CDAT | I2CCLK | I2CADR | I2CXAR |
| 0xE8 | - | - | SCON1 | SBUF1 | SPCR | SPSR | SPDR | SSCR |
| 0xE0 | ACC | - | - | - | - | - | - | - |
| 0xD8 | P5 | - | - | - | ADRESL | ADRESH | ADCON1 | ADCON0 |
| 0xD0 | PSW | - | - | TL5 | TH5 | T5CON | -- | -- |
| 0xC8 | T2CON | T2IF | RLDL | RLDH | TL2 | TH2 | CCEN | T2IE |
| 0xC0 | - | - | CCL1 | CCH1 | - | - | - | - |
| 0xB8 | IP | EIP1 | EIP2 | EIP3 | - | - | - | - |
| 0xB0 | P3 | - | EIF2 | - | EI0IF | - | - | - |
| 0xA8 | IE | - | EIE2 | - | EI0IE | - | - | - |
| 0xA0 | P2 | IREMAP | - | - | - | - | - | - |
| 0x98 | SCON0 | SBUF0 | P0TRIS | P1TRIS | P2TRIS | - | - | P5TRIS |
| 0x90 | P1 | - | PCGEN0 | PCGEN1 | - | RSTCF | TA | WDCON |
| 0x88 | TCON | TMOD | TL0 | TL1 | TH0 | TH1 | CKCON | WDKEY |
| 0x80 | P0 | SP | DPL0 | DPH0 | - | - | DPS | PCON |

## 2.6 External Special Function Registers (XSFR)

The XSFR (External Special Function Registers) are special registers whose addressing space is shared with the XRAM (External RAM). These registers primarily include port control registers and other function control registers. The addressing range is shown in the diagram below:

Here is a list of the External Special Function Registers:

| Address | Register | Description |
|---|---|---|
| F060H | P0DIG | P0 port analog control register |
| F061H | P1DIG | P1 port analog control register |
| F062H | P2DIG | P2 port analog control register |
| -- | -- | -- |
| F065H | P5DIG | P5 port analog control register |
| -- | -- | -- |
| F068H | P0UP | P0 port pull-up resistor control register |
| F069H | P1UP | P1 port pull-up resistor control register |
| F06AH | P2UP | P2 port pull-up resistor control register |
| -- | -- | -- |
| F06DH | P5UP | P5 port pull-up resistor control register |
| -- | -- | -- |
| F070H | P0OD | P0 port open-drain output control register |
| F071H | P1OD | P1 port open-drain output control register |
| F072H | P2OD | P2 port open-drain output control register |
| -- | -- | -- |
| F075H | P5OD | P5 port open-drain output control register |
| -- | -- | -- |
| F078H | P0RD | P0 port pull-down resistor control register |
| F079H | P1RD | P1 port pull-down resistor control register |
| F07AH | P2RD | P2 port pull-down resistor control register |
| -- | -- | -- |
| F07DH | P5RD | P5 port pull-down resistor control register |
| -- | -- | -- |
| F0C2H | PS_T0 | Timer0 external clock I/O allocation register |
| F0C3H | PS_T0G | Timer0 gate input port allocation register |
| F0C4H | PS_T1 | Timer1 external clock I/O allocation register |
| F0C5H | PS_T1G | Timer1 gate input port allocation register |
| F0C6H | PS_T2 | Timer2 external event or gate input port allocation register |
| F0C7H | PS_T2EX | Timer2 falling edge auto-reload input port allocation register |
| F0C8H | PS_EINT0 | External interrupt extension input EINT0 channel GPIO select register |
| F0C9H | PS_EINT1 | External interrupt extension input EINT1 channel GPIO select register |
| F0CAH | PS_EINT2 | External interrupt extension input EINT2 channel GPIO select register |
| F0CBH | PS_EINT3 | External interrupt extension input EINT3 channel GPIO select register |
| -- | -- | -- |
| F0D0H | PS_TXD1 | UART1 data output port allocation register |
| F0D1H | PS_RXD1 | UART1 data input port allocation register |
| F0D2H | PS_TXD0 | UART0 data output port allocation register |
| F0D3H | PS_RXD0 | UART0 data input port allocation register |
| F0D4H | PS_SCL | IIC clock input port allocation register |

| Address | Register | Description |
|---|---|---|
| F0D5H | PS_SDA | IIC data input port allocation register |
| F0D6H | PS_PG0 | EPWM output channel 0 port select register |
| F0D7H | PS_PG1 | EPWM output channel 1 port select register |
| F0D8H | PS_PG2 | EPWM output channel 2 port select register |
| F0D9H | PS_PG3 | EPWM output channel 3 port select register |
| F0DAH | PS_PG4 | EPWM output channel 4 port select register |
| F0DBH | PS_PG5 | EPWM output channel 5 port select register |
| F0DCH | PS_CC0 | Timer2 output channel 0 port select register |
| F0DDH | PS_CC1 | Timer2 output channel 1 port select register |
| -- | -- | -- |
| F0E0H | EI0CFGL | External extension interrupt group interrupt control register |
| -- | -- | -- |
| F0E4H | INTFC | External interrupt filter control register |
| -- | -- | -- |
| F0F0H | PS_SCLKOI | SPI clock input port select register |
| F0F1H | PS_MOSI | SPI master output/slave input port select register |
| F0F2H | PS_MISO | SPI master input/slave output port select register |
| F0F3H | PS_NSS0OI | SPI chip select port 0 select register |
| F0F4H | PS_NSS1OI | SPI chip select port 1 select register |
| F0F5H | PS_NSS2OI | SPI chip select port 2 select register |
| F0F6H | PS_NSS3OI | SPI chip select port 3 select register |
| F0F7H | PS_CLO | Clock output port select register |
| -- | -- | -- |
| F120H | PWMCON | PWM control register |
| F121H | PWMOE | PWM output enable register |
| F122H | PWMPINV | PWM output polarity select register |
| -- | -- | -- |
| F126H | PWMCNTE | PWM counting start control register |
| F127H | PWMCNTM | PWM counting mode select register |
| F128H | PWMCNTCLR | PWM counting clear control register |
| F129H | PWMLOADEN | PWM load enable control register |
| F12AH | PWM0DIV | PWM0 clock division control register |
| F12BH | PWM1DIV | PWM1 clock division control register |
| F12CH | PWM2DIV | PWM2 clock division control register |
| -- | -- | -- |
| F130H | PWMP0L | PWM01 period data register lower 8 bits |
| F131H | PWMP0H | PWM01 period data register higher 8 bits |
| | | |
| | | |
| F134H | PWMP1L | PWM23 period data register lower 8 bits |
| F135H | PWMP1H | PWM23 period data register higher 8 bits |
| | | |
| | | |
| F138H | PWMP2L | PWM45 period data register lower 8 bits |
| F139H | PWMP2H | PWM45 period data register higher 8 bits |
| | | |
| | | |
| -- | -- | -- |

| Address | Register | Description |
|---|---|---|
| F140H | PWMD0L | PWM0 compare data register lower 8 bits |
| F141H | PWMD0H | PWM0 compare data register higher 8 bits |
| F142H | PWMD1L | PWM1 compare data register lower 8 bits |
| F143H | PWMD1H | PWM1 compare data register higher 8 bits |
| F144H | PWMD2L | PWM2 compare data register lower 8 bits |
| F145H | PWMD2H | PWM2 compare data register higher 8 bits |
| F146H | PWMD3L | PWM3 compare data register lower 8 bits |
| F147H | PWMD3H | PWM3 compare data register higher 8 bits |
| F148H | PWMD4L | PWM4 compare data register lower 8 bits |
| F149H | PWMD4H | PWM4 compare data register higher 8 bits |
| F14AH | PWMD5L | PWM5 compare data register lower 8 bits |
| F14BH | PWMD5H | PWM5 compare data register higher 8 bits |
| -- | -- | -- |
| F160H | PWMDTE | PWM dead zone enable control register |
| F161H | PWM01DT | PWM0/1 dead zone delay data register |
| F162H | PWM23DT | PWM2/3 dead zone delay data register |
| F163H | PWM45DT | PWM4/5 dead zone delay data register |
| -- | -- | -- |
| F169H | PWMZIE | PWM zero interrupt mask register |
| -- | -- | -- |
| F16BH | PWMDIE | PWM down compare interrupt mask register |
| -- | -- | -- |
| F16DH | PWMZIF | PWM zero interrupt flag register |
| -- | -- | -- |
| F16FH | PWMDIF | PWM down compare interrupt flag register |
| -- | -- | -- |
| -- | -- | -- |
| -- | -- | -- |
| -- | -- | -- |
| F5C1H | BRTDL | BRT timer data load value lower 8 bits |
| F5C2H | BRTDH | BRT timer data load value higher 8 bits |
| -- | -- | -- |
| F690H | LVDCON | Power monitor register |
| F691H | LVDEICFG | LVD interrupt control register |
| F692H | LVDS | LVD voltage select register |
| F693H | ADCLDO | AD reference voltage control register |
| -- | -- | -- |
| F697H | XT_SCM | LSE/HSE clock stop detection control register |
| -- | -- | -- |
| -- | -- | -- |
| F708H | CRCIN | CRC module data input register |
| F709H | CRCDL | CRC result lower 8-bit data register |
| F70AH | CRCDH | CRC result higher 8-bit data register |
| -- | -- | -- |
| F740H | LEDC0DATA0/LED0DATA | SEG data register corresponding to COM0 |
| F741H | LEDC0DATA1/LED1DATA | |
| F742H | LEDC0DATA2 | |
| F743H | LEDC0DATA3 | |

| Address | Register | Description |
|---|---|---|
| F744H | LEDC1DATA0/LED2DATA | SEG data register corresponding to COM1 |
| F745H | LEDC1DATA1/LED3DATA | |
| F746H | LEDC1DATA2 | |
| F747H | LEDC1DATA3 | |
| F748H | LEDC2DATA0/LED4DATA | SEG data register corresponding to COM2 |
| F749H | LEDC2DATA1/LED5DATA | |
| F74AH | LEDC2DATA2 | |
| F74BH | LEDC2DATA3 | |
| F74CH | LEDC3DATA0/LED6DATA | SEG data register corresponding to COM3 |
| F74DH | LEDC3DATA1/LED7DATA | |
| F74EH | LEDC3DATA2 | |
| F74FH | LEDC3DATA3 | |
| F750H | LEDC4DATA0/LED0SEL | SEG data register corresponding to COM4 |
| F751H | LEDC4DATA1/LED1SEL | |
| F752H | LEDC4DATA2 | |
| F753H | LEDC4DATA3 | |
| F754H | LEDC5DATA0/LED2SEL | SEG data register corresponding to COM5 |
| F755H | LEDC5DATA1/LED3SEL | |
| F756H | LEDC5DATA2 | |
| F757H | LEDC5DATA3 | |
| F758H | LEDC6DATA0/LED4SEL | SEG data register corresponding to COM6 |
| F759H | LEDC6DATA1/LED5SEL | |
| F75AH | LEDC6DATA2 | |
| F75BH | LEDC6DATA3 | |
| F75CH | LEDC7DATA0/LED6SEL | SEG data register corresponding to COM7 |
| F75DH | LEDC7DATA1/LED7SEL | |
| F75EH | LEDC7DATA2 | |
| F75FH | LEDC7DATA3 | |
| F760H | LEDCOMEN/LEDIOEN | COM port enable control register |
| F761H | LEDSEGEN0/SCAN1W | SEG port enable control register |
| F762H | LEDSEGEN1/SCAN2W | SEG port enable control register |
| F763H | LEDSEGEN2 | SEG port enable control register |
| F764H | LEDSEGEN3 | SEG port enable control register |
| F765H | LEDCON/LEDCON1 | LED control register |
| F766H | LEDCKS | LED clock select register |
| F768H | LEDCOMTIME | COM port active time select register |
| F769H | LEDMODE | LED drive mode select register |
| F76AH | LEDSTATUS | LED LED dot matrix scanning interrupt register |
| F76BH | LEDDRV | LED matrix drive port current control register |
| F76CH | LEDRESEQ | LED dot matrix pin mapping register |
| F76DH | LEDSEGTIME | Dimming time select register |
| 0xF76E | LEDCOMDRV | LED COM port drive current select register |
| 0xF76F | LEDSEGDR0 | LED pin drive enable control register0 |
| 0xF770 | LEDSEGDR1 | LED pin drive enable control register1 |
| 0xF771 | LEDSEGDR2 | LED pin drive enable control register2 |
| 0xF772 | LEDSEGDR3 | LED pin drive enable control register3 |

# 3. Reset

Reset Time refers to the time interval from when the chip is reset to when it begins executing instructions. The default design value for this time is approximately 22ms. This time includes the oscillator startup time and configuration time. After a hardware reset, the chip will always have this reset time.

The chip supports several hardware reset methods:

- Power-on Reset
- External Reset
- Low Voltage Reset
- Watch Dog Overflow Reset

The chip also supports a software reset method:

- Write SWRST Reset (A software reset immediately clears the Program Counter (PC), and the program starts running from the reset vector.)

In any of the above reset scenarios, all system registers will be restored to their default state, the program will stop running, and the program counter (PC) will be cleared. After the reset, the program will start executing from the reset vector at address 0000H.

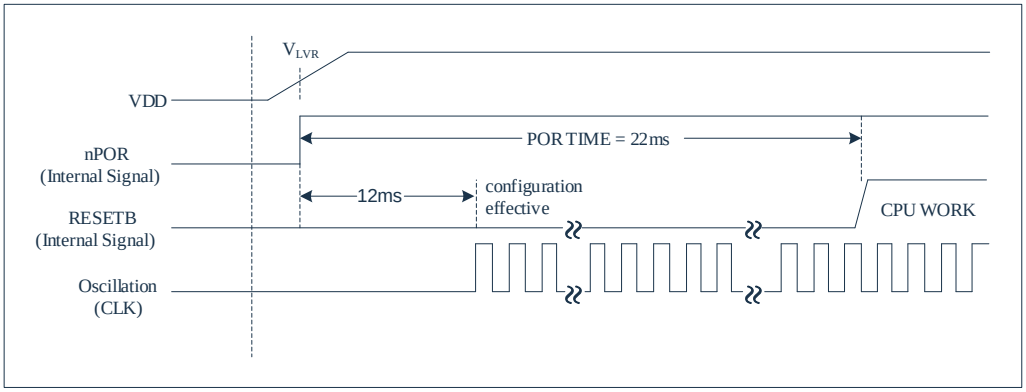
Each reset method requires a certain response time, and the system provides a complete reset process to ensure the reset action is carried out smoothly.

## 3.1 Power-On Reset

Power-on reset is closely related to the Low Voltage Reset (LVR) operation. The process of powering on the system follows a gradually increasing curve and requires some time to reach a stable voltage level. The normal timing sequence for the power-on reset is as follows:

- Power-on: The system detects the rise in power supply voltage and waits for it to stabilize.
- System Initialization: All system registers are set to their initial values.
- Oscillator Starts: The oscillator begins to provide the system clock.
- Program Execution: Once the power-on process is complete, the program starts running.

The Stabilization Time is typically set to 22ms by default. The timing diagram for the power-on reset is as shown below:




The system's power-on reset status can be determined by checking the PORF (RSTCF.6) flag. The following reset types can set the PORF flag to 1: Power-on Reset (POR), Low Voltage Reset (LVR), and External Reset.

## 3.2 External Reset

External reset refers to a reset signal coming from the external port (NRST), which is input through a Schmitt trigger to reset the chip. If the NRST pin remains low for approximately 4*$T_{LSI}$ or longer, under proper operating voltage and stable oscillation conditions, a reset request will be triggered. After the internal state is initialized and the reset status becomes 1, a stabilization time of 22ms is required before the internal RESETB signal changes to 0, and the program begins execution from the vector address 0000H.

The stabilization time (Stabilization Time) involves the chip going through a reconfiguration process, similar to the power-on reset configuration process. The external reset pin NRST and its pull-up resistor are enabled through the CONFIG.

The system's external reset status can be determined by checking the EXTIF (RSTCF.5) flag.

## 3.3 Low Voltage Detection Reset/Interrupt

The chip integrates an LVD (Low Voltage Detection) reset/interrupt function. When the system voltage (VDD) drops below a set threshold, it can trigger either a reset or an interrupt. Whether the system generates a reset or an interrupt is determined by the LVDCFG bits [1:0] in the CONFIG register.

(a) Reset Mode (LVDCFG[1:0] = 0x03, default is Reset Mode)

In reset mode, the voltage comparison result will generate an internal reset signal.

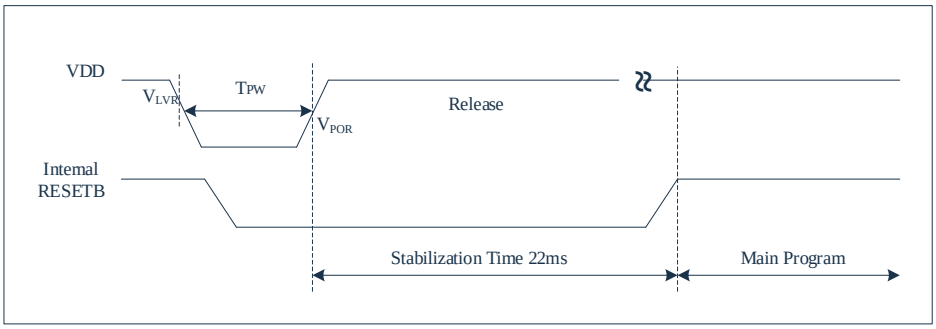
The power supply voltage detection threshold is selectable through bits [4:2] of the LVDCFG register. The available detection voltages are: 2.5V, 2.7V, 3.0V, 3.3V, 3.7V, 4.0V, and 4.3V (7 different voltage options).

(b) Interrupt Mode (LVDCFG[1:0] = 0x01)

In interrupt mode, the voltage comparison result will generate an internal interrupt signal.

The power supply voltage detection threshold is selected via bits [2:0] of the LVDS register. The available detection voltages are: 2.5V, 2.7V, 3.0V, 3.3V, 3.7V, 4.0V, and 4.3V (7 different voltage options).

A reset timing diagram is shown below:




Note 1: $V_{POR}$ is the power-on reset release voltage. It is higher than $V_{LVR}$, with a hysteresis voltage of approximately 50mV.

Note 2: $T_{PW}$ is the time required to ensure a proper power-on reset when the system voltage (VDD) drops below $V_{LVR}$ and then rises above $V_{POR}$. This time ensures the system operates correctly during power-up reset (Note: This does not mean that the chip will not generate a valid reset if the time is shorter than this threshold.). $T_{PW} > 32$ μs.

Note 3: Stabilization Time refers to the period during which the chip undergoes a reconfiguration process. This process is identical to the power-up reset configuration process.

The chip includes a continuously active low-power POR detection circuit. Regardless of the system's state, the POR detection circuit is always monitoring. If VDD drops below $V_{POR}$, the chip immediately generates a reset. ($V_{POR} < V_{LVR}$).

## 3.4 Watch Dog Reset

The watch dog reset is a protection mechanism for the system. Under normal conditions, the program resets the watch dog timer. If an error occurs and the system enters an unknown state, the watch dog timer overflows, triggering a system reset. After the watch dog reset, the system restarts and enters normal operation.

The WDT counter is not addressable. By default, the WDT overflow reset function is enabled after power-up, but this function can be disabled via a register. Additionally, it can be forced to be enabled after power-up reset in the CONFIG register. When setting the WDT register, it is recommended to clear the WDT counter first to ensure accurate control of the WDT overflow time.

The sequence of the watch dog reset is as follows:

1) Watch Dog Timer Status: The system checks whether the watch dog timer has overflowed. If it has, the system is reset.
2) Initialization: All system registers are set to their default state.
3) Program: After the reset is complete, the program starts executing from address 0000H.

The WDT clock source is provided by the LSI (Low-Speed Internal oscillator), and the basic timing period of the WDT counter is TLSI. After the WDT overflows, it resets the CPU and all registers. In debug mode, the WDT reset does not reconfigure the power-on reset. After 1 Tsys (system clock period), the program immediately starts executing from address 0000H. In non-debug mode, the power-on reset configuration will be re-applied. The watch dog overflow time can be set by the program. The WDCON register's WDS2-WTS0 bits allow the selection of the overflow time. The watch dog overflow times are listed in the table below:

| WTS[2:0] | Watch dog interval | Number of clocks | OVT@$F_{LSI}$=32KHz |
|---|---|---|---|
| 000 | $2^8$ | 256 | 8ms |
| 001 | $2^9$ | 512 | 16ms |
| 010 | $2^{10}$ | 1024 | 32ms |
| 011 | $2^{11}$ | 2048 | 64ms |
| 100 (default) | $2^{12}$ | 4096 | 128ms |
| 101 | $2^{14}$ | 16384 | 512ms |
| 110 | $2^{15}$ | 32768 | 1s |
| 111 | $2^{17}$ | 131072 | 4s |

## 3.5 Software Reset

The chip supports a software reset internally, which can relocate the program flow to the reset address 0000H and restart the program execution. Users can initiate a custom software reset by setting the software reset control bit RSTCF[7] (SWRST = 1). The software reset will not reconfigure the power-on reset settings.

## 3.6 Reset Control Register (RSTCF)

| 0x95 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| RSTCF | SWRST | PORF | EXTIF | — | -- | WDTRF | -- | LVRRF |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 SWRST: Software reset control bit
1: Perform a system software reset (after the reset, the bit needs to be cleared by writing 0, and the timing needs to be done by TA).
0: --

Bit6 PORF: Power-on reset flag
1: The system is reset by power-on reset (write 0 to clear, no TA timing required).
0: --

Bit5 EXTIF: External reset flag
1= The system is reset by an external reset (write 0 to clear, no TA timing required).
0= --

Bit4 - Reserved

Bit3 - Reserved, set to 0.

Bit2 WDTRF: WDT reset flag
1= The system is reset by the Watch Dog Timer (write 0 to clear).
0= The system is not reset by the WDT.

Bit1 - Reserved, set to 0.

Bit0 LVRRF LVD reset flag
1= The system is reset by the LVD (write 0 to clear, no TA timing required).
0= --

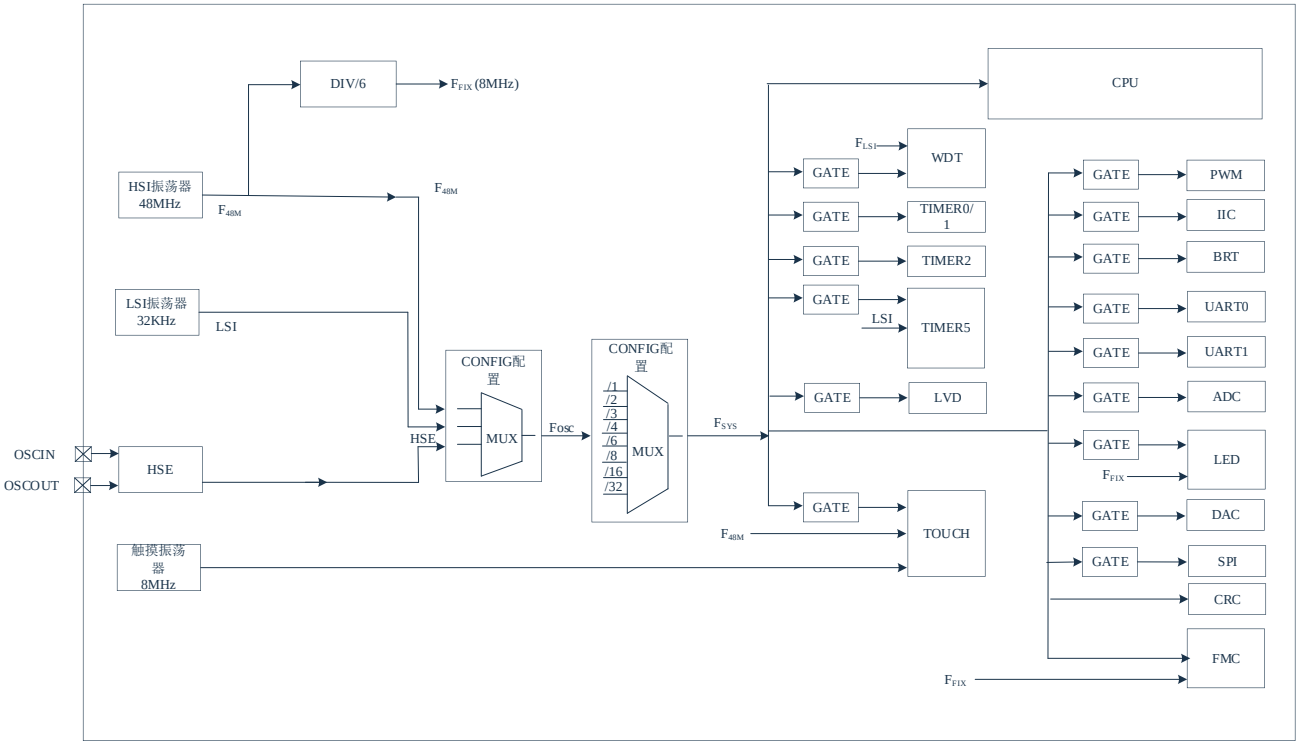
# 4. Clock Structure

The system clock source has three types, and the clock source and clock division can be selected through the system configuration register or user register settings. The system clock sources are as follows:

- Internal High-Speed Oscillator (HSI) (48 MHz)
- External High-Speed Oscillator (HSE) (8 MHz/16 MHz)
- Internal Low-Speed Oscillator (LSI) (32 kHz)

## 4.1 System Clock Structure

The block diagram of the system clock structure is shown below:



## 4.2 Relevant Registers

### 4.2.1 Peripheral Clock Source Selection Register (CKCON)

| 0x8E | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| CKCON | -- | -- | -- | U1M | U0M | T2M | T1M | T0M |
| R/W | R/W | R/W | R/W | W | W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit5 -- Reserved, set to 0.

Bit4 U1M: UART1 timer clock source selection

0= BRT overflow clock

1= Timer1 overflow clock

Bit3 U0M: UART0 timer clock source selection

0= BRT overflow clock

1= Timer1 overflow clock

T2M: Timer2 clock source selection

Bit2 0= Selected by T2PS

1= Fsys

Bit1 T1M Timer1 clock source selection

0= Fsys/12

1= Fsys/4

Bit0 T0M Timer0 clock source selection

0= Fsys/12

1= Fsys/4

### 4.2.2 System Clock Monitoring Register (XTSCM)

| F697H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| XTSCM | SCMEN | SCMIE | -- | -- | -- | -- | SCMIF | SCMSTA |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R |
| Reset value | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 SCMEN: Clock stop detection module enable
- 1= Enable
- 0= Disable

Bit6 SCMIE: Clock stop detection interrupt enable
- 1= Enable
- 0= Disable

Bit5~Bit2 -- Reserved, set to 0.

Bit1 SCMIF: Clock stop interrupt flag
- 1= Clock stop
- 0= Cleared by software; after clearing, the system will automatically switch to HSE main frequency (can only be cleared by software)

Bit0 SCMSTA: Clock stop status (read-only)
- 1= Clock stop
- 0= Clock stop recovered

Description:

1) Both SCMIF and SCMSTA can reflect the status of HSE as the system clock. The main difference is that once HSE stops, SCMSTA remains high until HSE recovers, while SCMIF can trigger an interrupt (if interrupt is enabled) and can be cleared via software. After clearing SCMIF, the main frequency will switch back to HSE. If HSE is still in the stopped state, the interrupt will be triggered again.
2) After a clock stop, the system clock will switch from HSE to HSI. When HSE recovers, SCMSTA will automatically clear, and the system clock will switch back from HSI to HSE.
3) During a system reset (when performing CONFIG), if a clock stop has occurred (or if the clock stop has not recovered during the system reset), SCMIF will be set to 1. In normal operation, if SCMIF is 1 during a software reset, SCMIF will remain 1 after the reset.
4) When SCMSTA is 1, software is prohibited from disabling SCMEN.

### 4.2.3 Module Clock Enable Register (PCGEN0)

| 92H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PCGEN0 | UART1EN | UART0EN | SPIEN | IICEN | WDTEN | TIMER5EN | TIMER2EN | TIMER01EN |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Note: Before accessing module-related registers, the module clock must be enabled.

Bit7 UART1EN: UART1 clock enable control
1= Enable UART1 clock
0= Disable UART1 clock

Bit6 UART0EN: UART0 clock enable control
1= Enable UART0 clock
0= Disable UART0 clock

Bit5 SPIEN: SPI clock enable control
1= Enable SPI clock
0= Disable SPI clock

Bit4 IICEN: IIC clock enable control
1= Enable IIC clock
0= Disable IIC clock

Bit3 WDTEN: WDT clock enable control
1= Enable WDT clock
0= Disable WDT clock

Bit2 TIMER5EN: TIMER5 clock enable control
1= Enable TIMER5 clock
0= Disable TIMER5 clock

Bit1 TIMER2EN: TIMER2 clock enable control
1= Enable TIMER2 clock
0= Disable TIMER2 clock

Bit0 TIMER01EN: TIMER01 clock enable control
1= Enable TIMER01 clock
0= Disable TIMER01 clock

### 4.2.4 Module Clock Enable Register (PCGEN1)

| 93H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PCGEN1 | - | TOUCHEN | LEDEN | ADCEN | -- | PWMEN | LVDEN | BRTEN |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Note: Before accessing module-related registers, the module clock must be enabled.

Bit7 -: Reserved

Bit6 TOUCHEN: TOUCH clock enable control

1= Enable TOUCH clock

0= Disable TOUCH clock

Bit5 LEDEN: LED clock enable control

1= Enable LED clock

0= Disable LED clock

Bit4 ADCEN: ADC clock enable control

1= Enable ADC clock

0= Disable ADC clock

Bit3 --

Bit2 PWMEN: PWM clock enable control

1= Enable PWM clock

0= Disable PWM clock

Bit1 LVDEN: LVD clock enable control

1= Enable LVD clock

0= Disable LVD clock

Bit0 BRTEN: BRT clock enable control

1= Enable BRT clock

0= Disable BRT clock

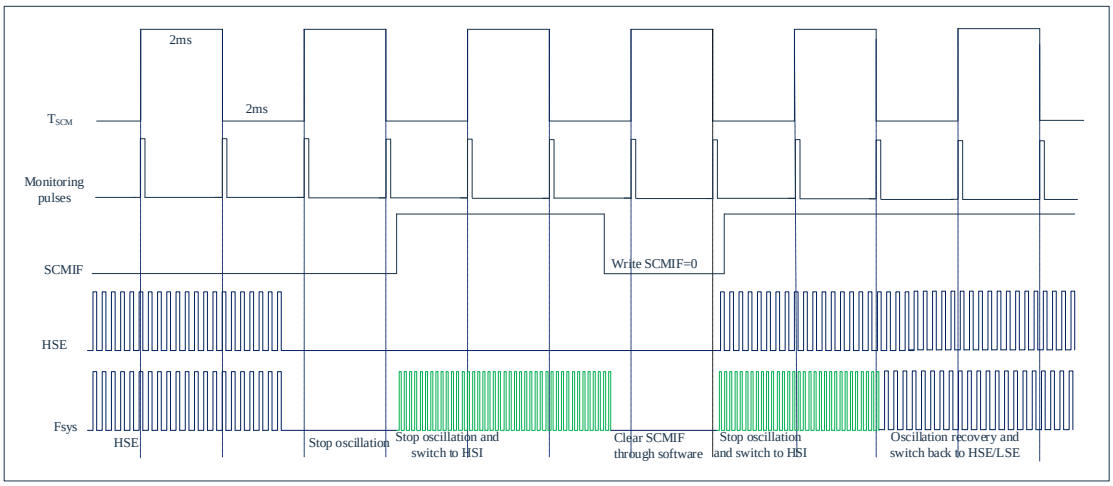
## 4.3 System Clock Monitoring (SCM)

System Clock Monitoring (SCM) is a protective circuit designed to prevent system malfunction caused by the failure of the crystal oscillator. When using the High-Speed External (HSE) clock as the system clock, if the HSE clock stops, the system will forcibly switch to the High-Speed Internal (HSI) clock source. Once the HSI stabilizes, the system will operate at an 8 MHz main frequency. If the HSE clock recovers and stabilizes, the system clock will automatically switch back from HSI to HSE.

The SCM module monitors the system clock (HSE) every 4 ms, with a 1:1 duty cycle for the monitoring period ($T_{SCM}$). During the high level of $T_{SCM}$, SCM monitors the HSE for any failure. During the low level of TSCM, the monitoring result is processed. If an HSE failure is detected, the system clock will switch to HSI, and the failure interrupt flag (SCMIF) will be set to 1. If SCMIF is cleared, even if HSE has failed, the system clock will automatically switch back to HSE.

When the system is powered on and HSE is selected as the system clock, the system clock monitoring is enabled by default. After the system reset is complete, the interrupt status flag should be checked to see if a failure is detected, and the clock monitoring interrupt flag should be cleared. Afterward, the register can be written to disable or re-enable the clock monitoring function.

The block diagram of the system clock monitoring structure is shown below:



# 5. Power Management

The low-power modes are divided into two categories:

- IDLE: Idle mode
- STOP: Sleep mode

When developing programs using C language, it is strongly recommended to use the IDLE and STOP macros to control the system modes, rather than directly setting the IDLE and STOP bits. The macros are as follows:

Enter idle mode: IDLE();

Enter sleep mode: STOP();

## 5.1 Power Management Registers

### 5.1.1 Power Control Register (PCON)

| 0x87 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PCON | -- | -- | -- | -- | -- | -- | STOP | IDLE |
| R/W | R | R | R | R | R | R | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | | |
|---|---|---|
| Bit7~Bit2 | -- | Reserved, set to 0. |
| Bit1 | STOP: | Sleep mode control bit |
| | 0= | Not in sleep mode |
| | 1= | Enter sleep mode (automatically cleared when exiting STOP mode) |
| Bit0 | IDLE: | Idle mode control bit |
| | 0= | Not in idle mode |
| | 1= | Enter idle mode (automatically cleared when exiting IDLE mode) |

Assembly code for entering sleep mode:

```
MOV     PCON,#02H
NOP     ;//At least 6 NOP instructions must follow
NOP
NOP
NOP
NOP
NOP
```

### 5.1.2 Low Voltage Detection Interrupt Control Register (LVDCON)

This MCU has built-in power detection functionality. The user can configure and select either the LVD (Low Voltage Detection) or LVR (Low Voltage Reset) function and set the voltage monitoring threshold. When the power supply voltage drops below the set LVD threshold, an interrupt is triggered to notify the user.

If the LVD module is enabled before entering sleep mode, the hardware will not shut down the LVD module circuit after the MCU enters sleep mode.

| 0xF690 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LVDCON | -- | -- | -- | -- | LVDOUTB_FILT | LVDOUTB | LVDINTE | LVDINTF |
| R/W | -- | -- | -- | -- | R | R | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | -- | -- | 0 | 0 |

Bit7~Bit4 -- Reserved, set to 0.
Bit3 LVDOUTB_FILT Supply voltage detection state filtered output
Bit2 LVDOUTB Power supply voltage monitoring status output
Bit1 LVDINTE: LVD interrupt enable bit
0= LVD interrupt disabled
1= LVD interrupt enabled
Bit0 LVDINTF: LVD interrupt flag
0= Power supply voltage is above the monitoring threshold
1= Power supply voltage is below the monitoring threshold (software clear)

### 5.1.3 Low Voltage Detection Interrupt Configuration Register (LVDEICFG)

| 0xF691 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LVDEICFG | LVDSNUS1 | LVDSNUS0 | LVDSCKS1 | LVDSCKS0 | -- | -- | INTEDGE1 | INTEDGE0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~6 LVDSNUS<1:0>: LVD filter time selection
00= 1 filter clock
01= 2 filter clocks
10= 3 filter clocks
11= 4 filter clocks
Bit5~4 LVDSCKS<1:0>: Filter clock selection
00= Filter clock is the system clock
01= Filter clock is the system clock divided by 4
10= Filter clock is the system clock divided by 16
11= Filter clock is the system clock divided by 64
Bit3~2 Reserved
Bit1~0 INTEDGE<1:0>: LVD interrupt edge selection
00= Edge detection disabled
01= Interrupt on rising edge
10= Interrupt on falling edge
11= Interrupt on both edges

### 5.1.4 Low Voltage Detection Interrupt Voltage Selection Register (LVDS)

| F692H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LVDS | LVDSEN3 | LVDSEN2 | LVDSEN1 | LVDSEN0 | -- | LVDS2 | LVDS1 | LVDS0 |
| R/W | W | W | W | W | -- | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | -- | 1 | 1 | 1 |

Bit[7:4] LVDSEN<3:0> LVDS[2:0] access enable

0101= LVDS[2:0] can be written when LVDSEN[3:0] is also written as 0101

Other values are invalid.

Other= Invalid

Bit3 —— Reserved

Bit[2:0] LVDS<2:0>: LVD low voltage detection interrupt threshold selection

| | | | |
|---|---|---|---|
| 000= | 4.3V | 100= | 3.3V |
| 001= | 4.0V | 101= | 3.0V |
| 010= | 3.7V | 110= | 2.7V |
| 011= | 3.0V | 111= | 2.5V |

## 5.2 Idle Mode (IDLE)

In this mode, only the CPU clock source is turned off. Therefore, in this state, peripheral functions (such as timers, PWM, and $I^2C$) and clock generators continue to operate normally.

Once the system enters the idle mode, it can be awakened by any interrupt. After waking up, it enters the interrupt service routine, and after returning from the interrupt, it continues to execute the next instructions in the sleep operation.

If the idle mode is entered during the interrupt service routine, the system can only be awakened by a higher-priority interrupt.

## 5.3 Sleep Mode (STOP)

In this mode, all circuits except for the TIMER5 (when the LSI is selected as the module count clock source) are turned off, and the system enters a low-power mode where digital circuits do not operate.

### 5.3.1 Sleep Wake-Up

After entering the sleep mode, the sleep wake-up function can be enabled to wake up the system from the sleep mode. There are several ways to wake up the system from sleep:

1) INT0/1 Interrupt

To use the INT0/1 interrupt to wake up the sleep mode, the global interrupt enable and the INT0/1 interrupt enable must be set before entering sleep to allow the system to be awakened. The relevant registers for the INT0 and INT1 interrupts include IE, IP, TCON, and IO alternate mapping registers. INT0/1 interrupt wake-up can only be triggered by a falling edge interrupt.

2) EINT0-EINT3 Interrupt

To use external interrupts for wake-up, the global interrupt enable and the port interrupt enable must be set before entering sleep to allow the system to be awakened. The external interrupt wake-up can be triggered by a rising edge, falling edge, or both edges. The edge trigger for the interrupt wake-up is set by the external interrupt control register EI0CFGL.

3) Timer5 Wake-up

Wake-up can also occur via Timer5. Before entering sleep, the timed wake-up function must be enabled, and the wake-up time from sleep must be configured. The clock source for the timed wake-up circuit is provided by the LSI (low-power oscillator), which automatically starts when the timed wake-up function is enabled in the sleep mode.

### 5.3.2 Wake-Up Wait State

Whether it is an INT0/1 interrupt, EI00-EI03 interrupt, or T5 timed wake-up, after the interrupt occurs or the timing duration is reached, the system needs to wait for a period before the system can be awakened and the next instruction can be executed. After the interrupt or timer wake-up, the system oscillator starts, but the oscillation frequency is not stable, and the CPU does not work, so the Program Counter (PC) remains in the sleep state. The system must wait for a period to stabilize the clock before it is provided to the CPU. The wake-up wait time is set in the CONFIG during programming, and the waiting time can be set between 50μs to 1s. After the wait time, the MCU considers the system clock to be stable and then provides the clock to the CPU, and the program continues execution.

If both the internal wake-up timer and external interrupt wake-up functions are enabled, after the system enters sleep mode, either wake-up method can wake up the CPU. If the internal timer wakes up the oscillator first and then an external interrupt is triggered, the program will first execute the interrupt service routine, then continue to execute the sleep operation instructions after the wake-up wait time.

### 5.3.3 Sleep Wake-Up Time

The total wake-up time for using an external interrupt to wake up the system is:

Power Manager stabilization time (250μs) + Wake-up wait time.

The total wake-up time for using a timer to wake up the system is:

Power Manager stabilization time (250μs) + Wake-up timer timing + Wake-up wait time.

(The above times are given under the condition that Fsys > 1 MHz)

### 5.3.4 Sleep Mode Reset Operations

In sleep mode, the system can also be restarted through a power-down reset or an external reset.

Power-down reset: No additional conditions are required. When VDD drops to 0V and then is restored to the operating voltage, the system enters a power-on reset state.

External reset: This requires enabling the external reset function, configuring the relevant pin as a dedicated reset pin. During sleep mode, the reset pin should remain low for more than 4*$T_{LSI}$. When the reset pin is released, the system will restart.

### 5.3.5 Sleep Power Consumption in Debug Mode

The sleep state in debug mode does not reflect the actual sleep state of the chip.

In debug mode, when the system enters sleep, the relevant power management circuits and oscillators do not shut down but remain active. Debug mode allows for wake-up operations, which function the same as in normal mode.

Therefore, the measured sleep current in this state is not the actual sleep power consumption. It is recommended that after completing the development of the sleep-wake-up function in debug mode, debug mode should be disabled, and the system restarted. The current measured after this will reflect the actual sleep power consumption.

### 5.3.6 Example Application of Sleep Mode

Before entering sleep mode, if users require minimal sleep current, they should first check the status of all I/O pins. If there are floating I/O pins in the user design, all floating pins should be configured as output to ensure that each input pin has a fixed state. This will prevent the I/O pins from being in an indeterminate state when set as input, which would increase the sleep current. Additionally, shutting down ADC modules and other peripherals helps reduce sleep current.

Example: Entering Sleep Mode with Timer5 Wake-up (Assembly Code)

```
SLEEP_MODE:  MOV                 P0DIG, #0FFh
             MOV                 P0TRIS,#0FFh
             MOV                 P0,#0FFh
             MOV                 P1DIG, #0FFh
             MOV                 P1TRIS,#0FFh
             MOV                 P1,#0FFh
             MOV                 P2DIG, #0FFh
             MOV                 P2TRIS,#0FFh
             MOV                 P2,#0FFh
             MOV                 P5DIG, #0FFh
             MOV                 P5TRIS,#0FFh
             MOV                 P5,#0FFh
             MOV                 T5CON,#0E0h
             MOV                 EIE2,#004h
             MOV                 IE,#080h
             Disable other functions
             MOV                 PCON,#02H     ;Execute the sleep operation for wake-up functionality
             NOP
             NOP
             NOP
             NOP
             NOP
             NOP                 ; 6 NOP instructions required after sleep command
             Other operations after wake-up
```

# 6. Interrupts

## 6.1 Interrupt Overview

The chip has multiple interrupt sources and interrupt vectors, with the default interrupt vector offset address being 0x0000 (the interrupt offset address can be set via the IREMAP register):

| Interrupt source | Description | Interrupt vector | Priority level |
|---|---|---|---|
| INT0 | External interrupt 0 | 0-0x0003 | 1 |
| Timer0 | Timer 0 interrupt | 1-0x000B | 2 |
| INT1 | External interrupt 1 | 2-0x0013 | 3 |
| Timer1 | Timer 1 interrupt | 3-0x001B | 4 |
| UART0 | TI0 or RI0 | 4-0x0023 | 5 |
| Timer2 | Timer2 interrupt | 5-0x002B | 6 |
| UART1 | TI1 or RI1 | 6-0x0033 | 7 |
| EXT0IF<3:0> | Extended external interrupt group 0 | 7-0x003B | 8 |
| -- | -- | 8-0x0043 | 9 |
| -- | -- | 9-0x004B | 10 |
| -- | -- | 10-0x0053 | 11 |
| -- | -- | 11-0x005B | 12 |
| XTDET | Stop oscillation detection interrupt | 12-0x0063 | 13 |
| -- | -- | 13-0x006B | 14 |
| -- | -- | 14-0x0073 | 15 |
| -- | -- | 15-0x007B | 16 |
| -- | -- | 16-0x0083 | 17 |
| T5 | Timer 5 interrupt | 17-0x008B | 18 |
| PWM | PWM interrupt | 18-0x0093 | 19 |
| ADC | ADC interrupt | 19-0x009B | 20 |
| -- | -- | 20-0x00A3 | ~~21~~ |
| $I^2C$ | $I^2C$ interrupt | 21-0x00AB | 22 |
| SPI | SPI interrupt | 22-0x00B3 | 23 |
| -- | -- | 23-0x00BB | ~~24~~ |
| -- | -- | 24-0x00C3 | ~~25~~ |
| LED | LED interrupt | 25-0x00CB | 26 |
| LVD | LVD power-down interrupt | 26-0x00D3 | 27 |
| TOUCH | Touch interrupt | 27-0x00DB | 28 |
| -- | -- | 28-0x00E3 | 29 |

The chip defines two interrupt priority levels, allowing for two-level interrupt nesting. When one interrupt is being processed, if a higher-priority interrupt request occurs, the latter can interrupt the former, enabling interrupt nesting.

## 6.2 Interrupt Remapping

The system's default interrupt vector offset base address is 0x0000. However, in certain application scenarios, the base address may need to be set to a different value. This can be achieved by modifying the Interrupt Offset Register (IREMAP).

When the IREMAP register is set to 0x01, the offset base address becomes 0x0200 (i.e., 0x01 << 9 = 0x0200). After the remapping, all other interrupt vector addresses need to be adjusted by adding the offset base address. For example, after remapping, the interrupt vector address for INT0 should be: 0x0200 + 0x0003 = 0x0203.

From the above description, it can be inferred that the smallest offset unit for the interrupt vector is 0.5KB.

Example of Interrupt Remapping Usage:

In the last 2KB region of the program memory, an ISP (In-System Programming) function is implemented. Upon power-up, the program jumps to the starting address of the last 2KB. Then, the IREMAP register is set to 0x3C, meaning the interrupt offset base address is 0x7800. After completing the ISP function in this 2KB region or skipping it, all interrupt enables are disabled, the program jumps to the main program area, and the IREMAP register is set to 0x00, so the interrupt offset base address becomes 0x0000.

## 6.3 External Interrupts

### 6.3.1 INT0/INT1 Interrupt

The chip supports INT0 and INT1 external interrupts. These interrupts can be triggered on a falling edge or low level. The related control register is TCON. INT0 and INT1 occupy two interrupt vectors.

### 6.3.2 External Expansion Interrupt

The external expansion interrupt (EINT) includes four input channels: EINT0 to EINT3. Each channel has all GPIO pins (PS_EINT0, PS_EINT1, PS_EINT2, PS_EINT3), and each channel supports falling edge, rising edge, or both edges for triggering interrupts. The edge triggering type is configured via the EI0CFGL register.

For example, to configure the EINT0 pin to trigger on a falling edge interrupt:

```
……;                    //Set the related port to input mode
PS_EINT0   =0x00;       //EINT0 channel selects the EINT0 pin input
EI0CFGL=0x02;           //Set EINT0 to trigger interrupts on the falling edge
```

EINT0-EINT3 occupy a single interrupt vector at 0x003B.

After an interrupt occurs, the interrupt service routine (ISR) can first check which pin triggered the interrupt and then perform the corresponding action.

### 6.3.3 External Interrupt Filtering

External interrupts can be configured with filtering parameters, including the sampling clocks and the number of samples. The filtering parameters for all external interrupt pins are unified and set in the INTFC register.

## 6.4 Interrupt and Sleep Wake-Up

When the system enters sleep mode (STOP wake-up mode), each external interrupt can be configured to wake up the system.

INT0/INT1 Interrupt Wake-up: To wake up the system with INT0/INT1 interrupts, you need to enable the corresponding interrupt and the global interrupt. The wake-up trigger is the falling edge (note that the wake-up method for INT0/INT1 is independent of the interrupt trigger type set by the IT0/IT1 bits).

External Extension Interrupt (EINT) Wake-up: It is recommended to configure the edge trigger type (falling edge, rising edge, or both) for the corresponding port before entering sleep mode. Additionally, enable the relevant interrupt and global interrupt.

After the system is woken up by an external interrupt, it first enters the interrupt service routine (ISR) to handle the wake-up interrupt task. Once the ISR completes and exits, the system will continue to execute the instructions that follow the sleep operation.

## 6.5 Interrupt Registers

### 6.5.1 Interrupt Offset Base Address Register (IREMAP)

| 0xA1 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| IREMAP | - | - | MAP05 | MAP04 | MAP03 | MAP02 | MAP01 | MAP00 |
| R/W | R | R | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Note: IREMAP is a protected register.

| | | |
|---|---|---|
| Bit7-6 | - | Reserved |
| Bit5-0 | MAP<5:0> | Interrupt offset address |
| | 0X00= | Interrupt offset address is0X0000; |
| | 0X01= | Interrupt offset address is (0X01<<9), 0X0200; |
| | 0X02= | Interrupt offset address is (0X02<<9), 0X0400; |
| | …… | …… |
| | 0X20= | Interrupt offset address is (0X20<<9), 0X4000; |
| | 0X20= | Interrupt offset address is (0X20<<9), 0X4000; |
| | …… | …… |

To modify the IREMAP register, the following sequence of instructions must be followed strictly, without inserting any other instructions in between:

```
CLR     EA
MOV     TA,#0AAH
MOV     TA,#055H
ORL     IREMAP,#01H
SETB    EA
```

## 6.5.2 Interrupt Mask Registers

### 6.5.2.1 Interrupt Mask Register (IE)

The Interrupt Mask Register (IE) is a read-write register that supports bit-level operations. When an interrupt condition occurs, the corresponding interrupt flag will be set to 1, regardless of the state of the corresponding interrupt enable bit or the global enable bit (EA). User software should ensure that the interrupt flag is cleared before enabling the corresponding interrupt.

| 0xA8 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| IE | EA | ES1 | ET2 | ES0 | ET1 | EX1 | ET0 | EX0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 EA: Global interrupt enable bit
1= Enable all unmasked interrupts
0= Disable all interrupts
Bit6 ES1: UART1 interrupt enable bit
1= Enable UART1 interrupt
0= Disable UART1 interrupt
Bit5 ET2: TIMER2 global interrupt enable bit
1= Enable TIMER2 all interrupts
0= Disable TIMER2 all interrupts
Bit4 ES0: UART0 interrupt enable bit
1= Enable UART0 interrupt
0= Disable UART0 interrupt
Bit3 ET1: TIMER1 interrupt enable bit
1= Enable TIMER1 interrupt
0= Disable TIMER1 interrupt
Bit2 EX1: External interrupt 1 interrupt enable bit
1= Enable external interrupt 1 interrupt
0= Disable external interrupt 1 interrupt
Bit1 ET0: TIMER0 interrupt enable bit
1= Enable TIMER0 interrupt
0= Disable TIMER 0 interrupt
Bit0 EX0: External interrupt 0 interrupt enable bit
1= Enable external interrupt 0 interrupt
0= Disable external interrupt 0 interrupt

#### 6.5.2.2 Interrupt Mask Register (EIE2)

| 0xAA | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| EIE2 | SPIIE | I2CIE | -- | ADCIE | PWMIE | ET5 | -- | -- |
| R/W | R/W | R/W | R | R/W | R/W | R/W | R | R |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 SPIIE: SPI interrupt enable bit
1= Enable SPI interrupt
0= Disable SPI interrupt
Bit6 I2CIE: $I^2C$ interrupt enable bit
1= Enable $I^2C$ interrupt
0= Disable $I^2C$ interrupt
Bit5 --: Reserved, set to 0.
Bit4 ADCIE: ADC interrupt enable bit
1= Enable ADC interrupt
0= Disable ADC interrupt
Bit3 PWMIE: PWM global interrupt enable bit
1= Enable PWM all interrupts
0= Disable PWM all interrupts
Bit2 ET5: Timer5 interrupt enable bit
1= Enable Timer5 interrupt
0= Disable Timer5 interrupt
Bit1~0 -- Reserved, set to 0.

#### 6.5.2.3 Timer2 Interrupt Mask Register (T2IE)

| 0xCF | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| T2IE | T2OVIE | T2EXIE | -- | -- | -- | -- | T2C1IE | T2C0IE |
| R/W | R/W | R/W | R | R | R | R | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 T2OVIE: Timer2 overflow interrupt enable bit
1= Enable interrupt
0= Disable interrupt
Bit6 T2EXIE: Timer2 external loading interrupt enable bit
1= Enable interrupt
0= Disable interrupt
Bit5~Bit2 -- Reserved, set to 0.
Bit1 T2C1IE: Timer2 compare/capture channel 1 interrupt enable bit
1= Enable interrupt
0= Disable interrupt
Bit0 T2C0IE: Timer2 compare/capture channel 0 interrupt enable bit
1= Enable interrupt
0= Disable interrupt

If the Timer2 interrupt is enabled, you also need to enable the global interrupt enable bit for Timer2, ET2 = 1 (IE.5 = 1).

#### 6.5.2.4 External Interrupt Extension Group 0 Control Register (EI0IE)

| 0xAC | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| EI0IE | - | - | - | - | EI0IE3 | EI0IE2 | EI0IE1 | EI0IE0 |
| R/W | R | R | R | R | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit4 -- Reserved, set to 0.
Bit3~Bit0 EIiIE: EI0i port interrupt enable bit (i=0-3)
1= Enable interrupt
0= Disable interrupt

## 6.5.3 Interrupt Priority Control Registers

### 6.5.3.1 Interrupt Priority Control Register (IP)

The Interrupt Priority Control Register (IP) is a read-write register that supports bit-level operations.

| 0xB8 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| IP | -- | PS1 | PT2 | PS0 | PT1 | PX1 | PT0 | PX0 |
| R/W | W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 -- Reserved, set to 0.
Bit6 PS1: UART1 interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit5 PT2: TIMER2 interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit4 PS0: UART0 interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit3 PT1: TIMER1 interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit2 PX1: External interrupt 1 interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit1 PT0: TIMER0 interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit0 PX0: External interrupt 0 interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt

### 6.5.3.2 Interrupt Priority Control Register (EIP1)

| 0xB9 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| EIP1 | -- | -- | -- | -- | -- | -- | -- | PEI0 |
| R/W | R | R | R | R | R | R | R | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7-Bit1 -- Reserved, set to 0.
Bit0 PEI0: External extended interrupt group 0 priority control bit
1= High-level interrupt
0= Low-level interrupt

#### 6.5.3.3 Interrupt Priority Control Register (EIP2)

| 0xBA | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| EIP2 | PSPI | PI2C | -- | PADC | PPWM | PT5 | -- | -- |
| R/W | R/W | R/W | R | R/W | R/W | R/W | R | R |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 PSPI: SPI interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt

Bit6 PI2C: $I^2C$ interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt

Bit5 -- Reserved, set to 0.

Bit4 PADC: ADC interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt

Bit3 PPWM: PWM interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt

Bit2 PT5: TIMER5 interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt

Bit1~0 -- Reserved, set to 0.

#### 6.5.3.4 Interrupt Priority Control Register (EIP3)

| 0xBB | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| EIP3 | PXTDET | -- | -- | PTOUCH | PLVD | PLED | -- | -- |
| R/W | R/W | R | R | R/W | R/W | R/W | R | R |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | | |
|---|---|---|
| Bit7 | PXTDET | Crystal oscillator stop interrupt priority control bit |
| | 1= | High-level interrupt |
| | 0= | Low-level interrupt |
| Bit6-Bit5 | -- | Reserved, set to 0. |
| Bit4 | PTOUCH | TOUCH interrupt priority control bit |
| | 1= | High-level interrupt |
| | 0= | Low-level interrupt |
| Bit3 | PLVD: | LVD interrupt priority control bit |
| | 1= | High-level interrupt |
| | 0= | Low-level interrupt |
| Bit2 | PLED: | LED interrupt priority control bit |
| | 1= | High-level interrupt |
| | 0= | Low-level interrupt |
| Bit1-Bit0 | -- | Reserved, set to 0. |

## 6.5.4 Interrupt Flag Register

### 6.5.4.1 Timer0/1, INT0/1 Interrupt Flag Register (TCON)

| 0x88 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| TCON | TF1 | TR1 | TF0 | TR0 | IE1 | IT1 | IE0 | IT0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 TF1: Timer1 overflow interrupt flag
1= Timer1 overflow; automatically cleared by hardware when entering the interrupt service routine, or can be cleared by software.
0= No overflow in Timer1.

Bit6 TR1: Timer1 operation control bit
1= Start Timer1
0= Stop Timer1

Bit5 TF0: Timer0 overflow interrupt flag
1= Timer0 overflow; automatically cleared by hardware when entering the interrupt service routine, or can be cleared by software.
0= No overflow in Timer0.

Bit4 TR0: Timer0 operation control bit
1= Start Timer0
0= Stop Timer0

Bit3 IE1: External interrupt 1 flag
1= External interrupt 1 occurred; automatically cleared by hardware when entering the interrupt service routine, or can be cleared by software.
0= No interrupt from external interrupt 1.

Bit2 IT1: External interrupt 1 trigger control bit
1= Triggered by falling edge.
0= Triggered by low level.

Bit1 IE0: External interrupt 0 flag
1= External interrupt 0 occurred; automatically cleared by hardware when entering the interrupt service routine, or can be cleared by software.
0= No interrupt from external interrupt 0.

Bit0 IT0: External interrupt 0 trigger control bit
1= Triggered by falling edge.
0= Triggered by low level.

#### 6.5.4.2 Timer2 Interrupt Flag Register (T2IF)

| 0xC9 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| T2IF | TF2 | T2EXIF | -- | -- | -- | -- | T2C1IF | T2C0IF |
| R/W | R/W | R/W | R | R | R | R | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 TF2: Timer2 overflow interrupt flag
1= Timer2 has overflowed. It requires software to clear it.
0= Timer2 no overflow
Bit6 T2EXIF: Timer2 external load flag
1= The T2EX port of Timer2 generates a falling edge, which needs to be cleared by software;
0= --
Bit5~Bit2 -- Reserved, set to 0.
Bit1 T2C1IF: Timer2 compare/capture channel 1 flag bit
1= Timer2 Compare channel 1 {CCH1:CCL1} = {TH2:TL2} or Capture channel 1 has generated a capture operation and needs to be cleared by software.
0= --
Bit0 T2C0IF: Timer2 compare/capture channel 0 flag bit
1= Timer2 compare channel 0 {RLDH:RLDL}={TH2:TL2} or Capture channel 0 has generated a capture operation and needs to be cleared by software.
0= --

#### 6.5.4.3 Peripheral Interrupt Flag Register (EIF2)

| 0xB2 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| EIF2 | SPIIF | I2CIF | -- | ADCIF | PWMIF | TF5 | -- | -- |
| R/W | R | R | R | R/W | R | R/W | R | R |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 SPIIF: SPI global interrupt flag (read-only)
1= SPI generates an interrupt, (this bit is automatically cleared when the specific interrupt flag bit is cleared)
0= No interrupt occurred in the SPI module
Bit6 I2CIF: $I^2C$ gobal interrupt flag (read-only)
1= $I^2C$ generates an interrupt, (this bit is automatically cleared when the specific interrupt flag bit is cleared)
0= No interrupt occurred in the $I^2C$
Bit5 -- Reserved, set to 0.
Bit4 ADCIF: ADC interrupt flag
1= ADC conversion is complete and requires software clearing
0= ADC conversion is not completed.
Bit3 PWMIF: PWM global interrupt flag (read-only)
1= PWM generates an interrupt, (this bit is automatically cleared when the specific interrupt flag bit is cleared)
0= No interrupt occurred in the PWM
Bit2 TF5: Timer5 overflow interrupt flag bit
1= Timer5 overflow interrupt flag bit
0= Timer5 timer no overflow
Bit1~0 -- Reserved, set to 0.

#### 6.5.4.4 SPI Interrupt Flag Register (SPSR)

| 0xED | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| SPSR | SPISIF | WCOL | -- | -- | -- | -- | -- | SSCEN |
| R/W | R | R | -- | -- | -- | -- | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 SPISIF: SPI transfer complete interrupt flag bit, read-only;
1= SPI transfer complete (read SPSR first, then read/write SPDR and clear);
0= SPI has not been transferred.
Bit6 WCOL: SPI write conflict interrupt flag bit, read-only;
1= SPI transfer incomplete generates write SPDR operation conflict (read SPSR first, then read/write SPDR and clear);
0= No writing conflicts.
Bit5~Bit1 -- Reserved, set to 0.
Bit0 SSCEN: SPI master mode NSS output control bit
1= NSS outputs high when SPI is idle;
0= NSS outputs the contents of register SSCR.

#### 6.5.4.5 UART Control Rgister (SCONn)

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| SCONn | UnSM0 | UnSM1 | UnSM2 | UnREN | UnTB8 | UnRB8 | TIn | RIn |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register SCON0 address 0x98, Register SCON1 address 0xEA.

Bit7~Bit2 UnSM0, UnSM1, UnSM2, UnREN, UnTB8, UnRB8n, see UARTn function description for details

Bit1 TIn: Transmit interrupt flag bit (requires software clearing);
1= The transmit buffer is empty and the next frame data can be transmitted.
0= --

Bit0 RIn: Receive interrupt flag bit (requires software clearing);
1= The receive buffer is full and the next frame data can be received after reading.
0= --

The TIn and RIn share the same interrupt vector. To determine whether it is a receive interrupt or a transmit interrupt, it is necessary to query the relevant status registers.

#### 6.5.4.1 External Extension Interrupt Group 0 Flag Register (EI0IF)

| 0xB4 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| EI0IF | - | - | - | - | EINT3IF | EINT2IF | EINT1IF | EINT0IF |
| R/W | R | R | R | R | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7-Bit4 -- Reserved, set to 0.

Bit3-Bit0 EINTiIF: External extended interrupt group 0 flag bit (i=0-3)
1= EINTi generates an interrupt that needs to be cleared by software.
0= EINTi does not generate an interrupt

### 6.5.5 Interrupt Flag Clearing Operations

The clearing operations for interrupt flags can be categorized into the following types:

- Hardware Automatic Clear (Requires Entering the Interrupt Service Routine)
- Software Clear

1) Interrupt Flags Automatically Cleared by Hardware

Interrupt flags that support hardware automatic clearing include those generated by INT0, INT1, T0, and T1. The condition for automatic flag clearing is: The global interrupt enable bit EA must be set to 1. The corresponding interrupt enable bit must be enabled. After the interrupt occurs, the system enters the corresponding interrupt service routine, and the flag is automatically cleared. If the interrupt enable is turned off, these flags can also be cleared using software.

2) Interrupt Flags Cleared by Software

Some interrupt flags can only be cleared via software. These flags are not automatically cleared when entering the interrupt service routine and need to be cleared by writing 0 to them. Otherwise, upon exiting the interrupt service routine, the flag will be re-set, causing the interrupt service routine to trigger again.

Software Clear Operations Considerations: When multiple interrupt flags exist in the same register and are not related to each other in timing (e.g., the PWMDIF interrupt flag register, which contains flags for the down-count compare interrupts of channels PG0 to PG5), it is not recommended to use the read-modify-write operation for clearing them. If a down-count compare interrupt is generated for PG0, the value of PWMDIF will be 0x01. After entering the interrupt service routine, a read-modify-write operation like:

PWMDIF &= 0xFE;

The operation is implemented as follows: first, the value of PWMDIF is read back into the CPU, then the operation is performed, and finally, the result is written back to PWMDIF. If the interrupt flag for PG1 (PWMDIF[1]) is set to 1 after reading, but PWMDIF[1] was 0 at the time of reading, then after the operation, when the value is written back, PWMDIF[1] will be set to 0. As a result, the previously triggered PG1 up-count interrupt flag (PWMDIF[1]) will be cleared.

To avoid this issue, it is recommended to directly write 0 to the interrupt flag, ensuring that unrelated flags are written as 1. For example: PWMDIF = 0xFE. This operation has no actual effect when writing 1 to unrelated interrupt flags.

### 6.5.6 External Interrupt Filter Control Register

#### 6.5.6.1 Interrupt Mask Register (INTFC)

| F0E4H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| INTFC | - | - | - | - | INTSNUS1 | INTSNUS0 | INTSCKS1 | INTSCKS0 |
| R/W | R | R | R | R | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit4 - Reserved, set to 0.

Bit3~Bit2 INTSNUS<1:0>: External interrupt filter sample count selection (valid for INT0/INT1/ EINT0-EINT3 at the same time)

00= 1

01= 2

10= 3

11= 4

Bit1~Bit0 INTSCKS<1:0>: External interrupt filter sample clock selection (valid for INT0/INT1/EINT0-EINT3 at the same time)

00= $F_{SYS}$/1

01= $F_{SYS}$/4

10= $F_{SYS}$/16

11= $F_{SYS}$/64

### 6.5.7 Special Interrupt Flags in Debug Mode

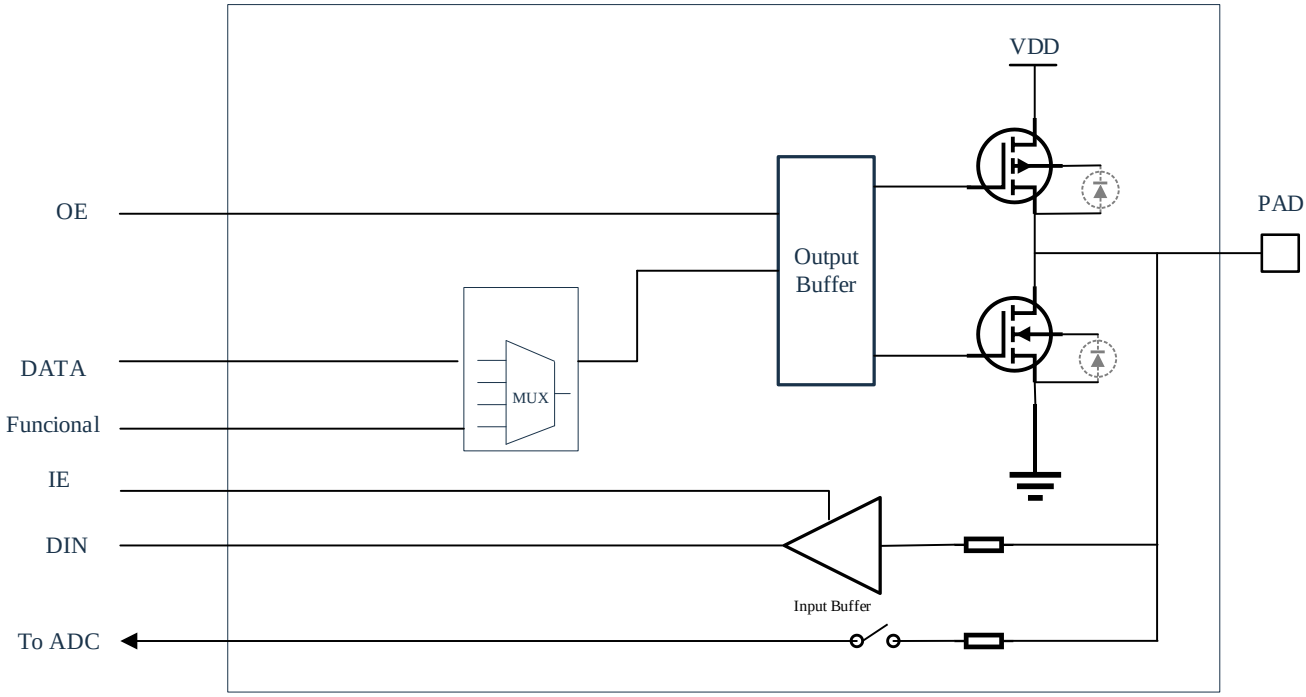
In the system, some flags are not cleared by writing 0 to the flag itself, but instead require reading or writing other registers to clear the flag.

In debug mode, after a breakpoint, single-step execution, or stop operation, the debugger reads all the register values from the system into the simulation software. The read/write operations of the debugger are identical to those in normal mode.

Therefore, during debugging, there may be cases where an interrupt flag, which should be set to 1, appears as 0 in the observation window after pausing. This behavior is something to be aware of in debug mode.

# 7. I/O Ports

## 7.1 Port Structure




## 7.2 GPIO Functions

The chip has 4 groups of I/O ports: PORT0, PORT1, PORT2, and PORT5.

PORTx is a bidirectional port. Its state is configured through the registers Px, PxDIG, PxTRIS, PxUP, and PxOD.

When PORTx is configured as an output port, writing to the Px register will write to the port latch. All write operations are read-modify-write operations. Therefore, writing to a port means first reading the pin level of the port, then modifying the read value, and finally writing the modified value back to the port data latch.

When PORTx pins are used as analog inputs, the I/O pin read value will be 0.

### 7.2.1 PORTx Digital Function Control Register (PxDIG)

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PxDIG | PxDIG7 | PxDIG6 | PxDIG5 | PxDIG4 | PxDIG3 | PxDIG2 | PxDIG1 | PxDIG0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | X | X | X | X | X | X | X | X |

Register P0DIG Address: F060H; Register P1DIG Address: F061H; Register P2DIG Address: F062H; Register P5DIG Address: F065H.

P0DIG reset value: 0x2C, P1DIG/P2DIG reset value: 0x00; P5DIG reset value: 0x18.

Bit7~Bit0 PxDIG<7:0>: Digital function control bit

1= Pin is configured for digital function

0= Pin is configured for analog function (disable GPIO port input/output function)

Note: Pins P02, P03, P53, and P54 are by default configured as digital inputs.

### 7.2.2 PORTx Direction Register (PxTRIS)

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PxTRIS | PxTRIS7 | PxTRIS6 | PxTRIS5 | PxTRIS4 | PxTRIS3 | PxTRIS2 | PxTRIS1 | PxTRIS0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register P0TRIS Address: 0x9A; Register P1TRIS Address: 0x9B; Register P2TRIS Address: 0x9C; Register P5TRIS Address: 0x9F.

Bit7~Bit0 PxTRIS<7:0>: Tri-state control bit

1= Pin is configured as output

0= Pin is configured as input (tri-state)

### 7.2.3 PORTx Data Register (Px)

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| Px | Px7 | Px6 | Px5 | Px4 | Px3 | Px2 | Px1 | Px0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | X | X | X | X | X | X | X | X |

Register P0 Address: 0x80; Register P1 Address: 0x90; Register P2 Address: 0xA0; Register P3 Address: 0xB0.

Bit7~Bit0 Px<7:0>: Px I/O pin bit

1= Port pin level > $V_{IH}$ (positive threshold voltage).

0= Port pin level < $V_{IL}$ (negative threshold voltage).

### 7.2.4 PORTx Pull-Up Resistor Register (PxUP)

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PxUP | PxUP7 | PxUP6 | PxUP5 | PxUP4 | PxUP3 | PxUP2 | PxUP1 | PxUP0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register P0UP Address: F068H; Register P1UP Address: F069H; Register P2UP Address: F06AH; Register P5UP Address: F06DH.

Bit7~Bit0 PxUP<7:0>: Pull-up resistor control bit (independent control, unrelated to other I/O configurations or alternated functions)

1= Pull-up resistor enabled

0= Pull-up resistor disabled

### 7.2.5 PORTx Pull-Down Resistor Register (PxRD)

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PxRD | PxRD7 | PxRD6 | PxRD5 | PxRD4 | PxRD3 | PxRD2 | PxRD1 | PxRD0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register P0RD Address: 0F078H; Register P1RD Address: 0F079H. Register P2RD Address: 0F07AH; Register P5RD Address: 0F07DH.

Bit7~Bit0 PxRD<7:0>: Pull-down resistor control bit (independent control, unrelated to other I/O configurations or alternated functions)

1= Pull-down resistor enabled

0= Pull-down resistor disabled

### 7.2.6 PORTx Open Drain Control Register (PxOD)

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PxOD | PxOD7 | PxOD6 | PxOD5 | PxOD4 | PxOD3 | PxOD2 | PxOD1 | PxOD0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register P0OD Address: F070H; Register P1OD Address: F071H; Register P2OD Address: F072H; Register P5OD Address: F075H.

Bit7~Bit0 PxOD<7:0>: Open-drain control bit (independent control, unrelated to other I/O configurations or alternated functions)

1= Pin is configured as open-drain: The pin is set to open-drain output.

0= Pin is configured as normal: The pin is set to push-pull output.

# 7.3 Alternate Functions

## 7.3.1 Port Alternate Function Table

Each I/O port can be flexibly configured for specific analog or digital functions, with multiple functions shared across the pins.

Set the PxDIG register to 0 to configure the corresponding pin as an analog function pin (for selecting analog channels ANn/TKn, refer to the corresponding sections). Set the PxDIG register to 1 to configure the corresponding pin as a digital function pin.

When a digital function pin is used for external input, it can be configured through the alterante function for options such as RXD1/RXD0/SCL/SDA, or set as a general-purpose I/O pin for input (set PxTRIS to 0). If the configured function has multiple pins available for allocation, the corresponding PS_xx setting must be applied to the relevant pins.

When the digital function pin is used for alternated functions, it is configured through the corresponding function registers (PS_xx) and the PxTRIS register. Depending on the application, the PxUP register can be used to configure pull-up resistors, and the PxOD register can be used to configure open-drain functionality.

When the digital function pin is used as a general-purpose I/O, the PxTRIS register controls the input/output direction, the PxUP register configures the pull-up resistors, and the PxOD register controls open-drain functionality.

Note: When a pin is used for digital functions, the priority of the alternated function is higher than that of the general-purpose I/O function. That is, when the pin is used for general-purpose I/O, the corresponding alternated function must be disabled.

The function configuration is shown in the table below:

| | AD | TK | LED矩阵 | LED点阵 | HSE/LSE | INT0 | INT1 | EINT0 | EINT1 | EINT2 | EINT3 | T0 | T1 | T2 | IIC | UART0 | UART1 | SPI | T2_O | PWM | CLKO | SWE | DEBUG |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| P00 | AN0 | TK0 | COM0 | LED0 | | | | EINT0 | EINT1 | EINT2 | EINT3 | | | | SCL | | TXD1/RXD1 | SCLK | CC0 | PG0 | | | |
| P01 | AN1 | TK1 | COM1 | LED1 | | | | EINT0 | EINT1 | EINT2 | EINT3 | | | | SDA | | TXD1/RXD1 | MOSI | CC1 | PG1 | | | |
| P02 | AN2 | TK2 | COM2 | LED2 | | | | EINT0 | EINT1 | EINT2 | EINT3 | T0G | T1G | T2EX | | | TXD1/RXD1 | MISO | | PG2 | | | DSDA2 |
| P03 | AN3 | TK3 | COM3 | LED3 | | | | EINT0 | EINT1 | EINT2 | EINT3 | T0 | T1 | T2 | | | TXD1/RXD1 | NSS(NSSO0) | | PG3 | | | DSCK2 |
| P04 | AN4 | TK4 | COM4/SEG0 | LED4 | | | | EINT0 | EINT1 | EINT2 | EINT3 | | | | | TXD0/RXD0 | TXD1/RXD1 | NSS(NSSO1) | | PG4 | | | |
| P05 | AN5 | TK5 | COM5/SEG1 | LED5 | | | | EINT0 | EINT1 | EINT2 | EINT3 | | | | | TXD0/RXD0 | TXD1/RXD1 | NSS(NSSO2) | | PG5 | | | |
| P06 | AN6 | TK6 | COM6/SEG2 | LED6 | | | | EINT0 | EINT1 | EINT2 | EINT3 | | | | | | TXD1/RXD1 | NSS(NSSO3) | | | | | |
| P07 | AN7 | TK7 | COM7/SEG3 | LED7 | | | | EINT0 | EINT1 | EINT2 | EINT3 | | | | | | TXD1/RXD1 | | | | CLO | | |
| P10 | AN8 | TK8 | SEG4 | LED8 | | | | EINT0 | EINT1 | EINT2 | EINT3 | | | | SCL | | TXD1/RXD1 | SCLK | | PG0 | CLO | | |
| P11 | AN9 | TK9 | SEG5 | | | | | EINT0 | EINT1 | EINT2 | EINT3 | | | | SDA | | TXD1/RXD1 | MOSI | | PG1 | | | |
| P12 | AN10 | TK10 | SEG6 | | | | | EINT0 | EINT1 | EINT2 | EINT3 | T0G | T1G | T2EX | | | TXD1/RXD1 | MISO | | PG2 | | | |
| P13 | AN11 | TK11 | SEG7 | | | | | EINT0 | EINT1 | EINT2 | EINT3 | T0 | T1 | T2 | | | TXD1/RXD1 | NSS(NSSO0) | | PG3 | | | |
| P14 | AN12 | TK12 | SEG8 | | | | | EINT0 | EINT1 | EINT2 | EINT3 | | | CAP0 | | TXD0/RXD0 | TXD1/RXD1 | NSS(NSSO1) | CAP0 | PG4 | | | |
| P15 | AN13 | TK13 | SEG9 | | | | | EINT0 | EINT1 | EINT2 | EINT3 | | | CAP1 | | TXD0/RXD0 | TXD1/RXD1 | NSS(NSSO2) | CAP1 | PG5 | | | |
| P16 | AN14 | TK14 | SEG10 | | | | | EINT0 | EINT1 | EINT2 | EINT3 | | | | | | TXD1/RXD1 | NSS(NSSO3) | | | | | |
| P17 | AN15 | TK15 | SEG11 | | | | | EINT0 | EINT1 | EINT2 | EINT3 | | | | | | TXD1/RXD1 | - | | | CLO | | |
| P20 | AN16 | TK16 | SEG12 | | | | | EINT0 | EINT1 | EINT2 | EINT3 | | | | SCL | | TXD1/RXD1 | SCLK | CC0 | PG0 | | | |
| P21 | AN17 | TK17 | SEG13 | | | | | EINT0 | EINT1 | EINT2 | EINT3 | | | | SDA | | TXD1/RXD1 | MOSI | CC1 | PG1 | | | |
| P22 | AN18 | TK18 | SEG14 | | | | | EINT0 | EINT1 | EINT2 | EINT3 | T0G | T1G | T2EX | | | TXD1/RXD1 | MISO | | PG2 | | | |
| P23 | AN19 | TK19 | SEG15 | | | | | EINT0 | EINT1 | EINT2 | EINT3 | T0 | T1 | T2 | | | TXD1/RXD1 | NSS(NSSO0) | | PG3 | | | |
| P24 | AN20 | TK20 | SEG16 | | | | | EINT0 | EINT1 | EINT2 | EINT3 | | | | | TXD0/RXD0 | TXD1/RXD1 | NSS(NSSO1) | | PG4 | | | |
| P25 | AN21 | TK21 | SEG17 | | | | | EINT0 | EINT1 | EINT2 | EINT3 | | | | | TXD0/RXD0 | TXD1/RXD1 | NSS(NSSO2) | | PG5 | | | |
| P26 | AN22 | TK22 | SEG18 | | | INT0 | | EINT0 | EINT1 | EINT2 | EINT3 | | | | | | TXD1/RXD1 | NSS(NSSO3) | | | | | |
| P27 | AN23 | TK23 | SEG19 | | | | INT1 | EINT0 | EINT1 | EINT2 | EINT3 | | | | | | TXD1/RXD1 | - | | | | | |
| P50 | AN39 | TK39 | SEG20 | | | | | EINT0 | EINT1 | EINT2 | EINT3 | | | | SCL | | TXD1/RXD1 | SCLK | | PG0 | | | |
| P51 | AN40 | TK40 | SEG21 | | OSCIN | | | EINT0 | EINT1 | EINT2 | EINT3 | | | | SDA | | TXD1/RXD1 | MOSI | | PG1 | | | |
| P52 | AN41 | TK41 | SEG22 | | OSCOUT | | | EINT0 | EINT1 | EINT2 | EINT3 | | | | | | TXD1/RXD1 | MISO | | PG2 | | | |
| P53 | AN42 | TK42 | SEG23 | | | | | EINT0 | EINT1 | EINT2 | EINT3 | | | | | | TXD1/RXD1 | NSS(NSSO0) | | PG3 | | SWE | DSDA1 |
| P54 | AN43 | TK43 | SEG24 | | | | | EINT0 | EINT1 | EINT2 | EINT3 | | | | | TXD0/RXD0 | TXD1/RXD1 | NSS(NSSO1) | | PG4 | | | DSCK1 |
| P55 | AN44 | TK44 | SEG25 | | | | | EINT0 | EINT1 | EINT2 | EINT3 | | | | | TXD0/RXD0 | TXD1/RXD1 | NSS(NSSO2) | | PG5 | | | |

### 7.3.2 Port Alternate Function Configuration

All GPIOs can be alternated to serve specific functions, such as GPIO interrupt input, INT0/1 interrupt input, EPWM output, SPI, IIC, UART, and others. The detailed functions of pin alternate can be found in the pinout or alternate function list. To configure any GPIO for alternated functions, it must follow a specific configuration method. For detailed information, refer to the alternate function register list.

| | | Register | Priority | Function configuration method | | |
|---|---|---|---|---|---|---|
| 0 | | PS_EINT0 | | PxDIG=1 | PxTRIS=0 | PS_EINT0=0xmn |
| 1 | | PS_EINT1 | | PxDIG=1 | PxTRIS=0 | PS_EINT1=0xmn |
| 2 | | PS_EINT2 | | PxDIG=1 | PxTRIS=0 | PS_EINT2=0xmn |
| 3 | | PS_EINT3 | | PxDIG=1 | PxTRIS=0 | PS_EINT3=0xmn |
| 4 | | PS_T0 | | PxDIG=1 | PxTRIS=0 | PS_T0=0xn |
| 5 | | PS_T0G | | PxDIG=1 | PxTRIS=0 | PS_T0G=0xn |
| 6 | | PS_T1 | | PxDIG=1 | PxTRIS=0 | PS_T1=0xn |
| 7 | | PS_T1G | | PxDIG=1 | PxTRIS=0 | PS_T1G=0xn |
| 8 | | PS_T2 | | PxDIG=1 | PxTRIS=0 | PS_T2=0xn |
| 9 | | PS_T2EX | | PxDIG=1 | PxTRIS=0 | PS_T2EX=0xn |
| 10 | | PS_RXD0 | | PxDIG=1 | PxTRIS=0 | PS_RXD0=0xn |
| 11 | | PS_RXD1 | | PxDIG=1 | PxTRIS=0 | PS_RXD1=0xmn |
| 12 | Output | PS_SCL | High | PxDIG=1 | PxTRIS=1 | PS_SCL=0xn |
| 13 | | PS_SDA | | PxDIG=1 | PxTRIS=1 | PS_SDA=0xn |
| 14 | | PS_TXD0 | | PxDIG=1 | PxTRIS=1 | PS_TXD0=0xn |
| 15 | | PS_TXD1 | | PxDIG=1 | PxTRIS=1 | PS_TXD1=0xmn |
| 16 | | PS_SCLKOI | | PxDIG=1 | Master PxTRIS=1<br>Slave PxTRIS=0 | PS_SCLKOI=0xn |
| 17 | | PS_MOSI | | PxDIG=1 | Master PxTRIS=1<br>Slave PxTRIS=0 | PS_MOSI=0xn |
| 18 | | PS_MISO | | PxDIG=1 | Master PxTRIS=0<br>Slave PxTRIS=1 | PS_MISO=0xn |
| 19 | | PS_NSS0OI | | PxDIG=1 | Master PxTRIS=1<br>Slave PxTRIS=0 | PS_NSS0OI=0xn |
| 20 | | PS_NSS1OI | | PxDIG=1 | Master PxTRIS=1<br>Slave PxTRIS=0 | PS_NSS1OI=0xn |
| 21 | | PS_NSS2OI | | PxDIG=1 | Master PxTRIS=1<br>Slave PxTRIS=0 | PS_NSS2OI=0xn |
| 22 | | PS_NSS3OI | | PxDIG=1 | Master PxTRIS=1<br>Slave PxTRIS=0 | PS_NSS3OI=0xn |
| 23 | | PS_CC0 | | PxDIG=1 | PxTRIS=1 | PS_CC0=0xn |
| 24 | | PS_CC1 | | PxDIG=1 | PxTRIS=1 | PS_CC1=0xn |
| 25 | | PS_PG0 | | PxDIG=1 | PxTRIS=1 | PS_PG0=0xn |
| 26 | | PS_PG1 | | PxDIG=1 | PxTRIS=1 | PS_PG1=0xn |
| 27 | | PS_PG2 | | PxDIG=1 | PxTRIS=1 | PS_PG2=0xn |
| 28 | | PS_PG3 | | PxDIG=1 | PxTRIS=1 | PS_PG3=0xn |
| 29 | | PS_PG4 | | PxDIG=1 | PxTRIS=1 | PS_PG4=0xn |
| 30 | | PS_PG5 | | PxDIG=1 | PxTRIS=1 | PS_PG5=0xn |
| 31 | | PS_CLO | Low | PxDIG=1 | PxTRIS=1 | PS_CLO=0xn |

#### 7.3.2.1 PS_EINT0 External Interrupt Extension Input EINT0 Channel Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_EINT0 | PS_EINT07 | PS_EINT06 | PS_EINT05 | PS_EINT04 | PS_EINT03 | PS_EINT02 | PS_EINT01 | PS_EINT00 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Bit7~Bit0 PS_EINT0mn<7:0>: External interrupt extension input EINT0 channel GPIO selection (m=0,1,2,5; n=0~7)

0x00= Allocated to P00
0x01= Allocated to P01
… …
0x07= Allocated to P07
0x10= Allocated to P10
0x11= Allocated to P11
… …
0x17= Allocated to P17
… …
0x55= Allocated to P55
Other: Reserved

#### 7.3.2.2 PS_EINT1 External Interrupt Extension Input EINT1 Channel Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_EINT1 | PS_EINT17 | PS_EINT16 | PS_EINT15 | PS_EINT14 | PS_EINT13 | PS_EINT12 | PS_EINT11 | PS_EINT10 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Bit7~Bit0 PS_EINT1mn<7:0>: External interrupt extension input EINT1 channel GPIO selection (m=0,1,2,5; n=0~7)

0x00= Allocated to P00
0x01= Allocated to P01
… …
0x07= Allocated to P07
0x10= Allocated to P10
0x11= Allocated to P11
… …
0x17= Allocated to P17
… …
0x55= Allocated to P55
Other: Reserved

#### 7.3.2.3 PS_EINT2 External Interrupt Extension Input EINT2 Channel Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_EINT2 | PS_EINT27 | PS_EINT26 | PS_EINT25 | PS_EINT24 | PS_EINT23 | PS_EINT22 | PS_EINT21 | PS_EINT20 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Bit7~Bit0 PS_EINT2mn<7:0>: External interrupt extension input EINT2 channel GPIO selection (m=0,1,2,5; n=0~7)

0x00= Allocated to P00
0x01= Allocated to P01
… …
0x07= Allocated to P07
0x10= Allocated to P10
0x11= Allocated to P11
… …
0x17= Allocated to P17
… …
0x55= Allocated to P55
Other: Reserved

#### 7.3.2.4 PS_EINT3 External Interrupt Extension Input EINT3 Channel Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_EINT3 | PS_EINT37 | PS_EINT36 | PS_EINT35 | PS_EINT34 | PS_EINT33 | PS_EINT32 | PS_EINT31 | PS_EINT30 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Bit7~Bit0 PS_EINT3mn<7:0>: External interrupt extension input EINT3 channel GPIO selection (m=0,1,2,5; n=0~7)

0x00= Allocated to P00
0x01= Allocated to P01
… …
0x07= Allocated to P07
0x10= Allocated to P10
0x11= Allocated to P11
… …
0x17= Allocated to P17
… …
0x55= Allocated to P55
Other: Reserved

#### 7.3.2.5 PS_T0 TIMER0 External Clock Input Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_T0 | | | | | | PS_T02 | PS_T01 | PS_T00 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.
Bit2~Bit0 PS_T0<2:0>: TIMER0 external clock input port selection
0x00= Select P03
0x01= Select P13
0x02= Select P23
Other: Reserved

#### 7.3.2.6 PS_T0G TIMER0 Gate Input Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_T0G | | | | | | PS_T0G2 | PS_T0G1 | PS_T0G0 |
| R/W | ~~R~~ | ~~R~~ | ~~R~~ | ~~R~~ | ~~R~~ | R/W | R/W | R/W |
| Reset value | ~~0~~ | ~~0~~ | ~~0~~ | ~~0~~ | ~~0~~ | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.
Bit2~Bit0 PS_T0G<2:0>: TIMER0 gate input port selection
0x00= Select P02
0x01= Select P12
0x02= Select P22
Other: Reserved

#### 7.3.2.7 PS_T1 TIMER1 External Clock Input Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_T1 | | | | | | PS_T12 | PS_T11 | PS_T10 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.
Bit2~Bit0 PS_T1<2:0>: TIMER1 external clock input port selection
0x00= Select P03
0x01= Select P13
0x02= Select P23
Other: Reserved

#### 7.3.2.8 PS_T1G TIMER1 Gate Input Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_T1G | | | | | | PS_T1G2 | PS_T1G1 | PS_T1G0 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PS_T1G<2:0>: TIMER0 gate input port selection

0x00= Select P02

0x01= Select P12

0x02= Select P22

Other: Reserved

#### 7.3.2.9 PS_T2 TIMER2 External Event/Gate Input Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_T2 | | | | | | PS_T22 | PS_T21 | PS_T20 |
| R/W | ~~R~~ | ~~R~~ | ~~R~~ | ~~R~~ | ~~R~~ | R/W | R/W | R/W |
| Reset value | ~~0~~ | ~~0~~ | ~~0~~ | ~~0~~ | 0 | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PS_T2<2:0>: TIMER2 external event or gate input port selection

0x00= Select P03

0x01= Select P13

0x02= Select P23

Other: Reserved

#### 7.3.2.10 PS_T2EX TIMER2 Falling Edge Auto Reload Input Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_T2EX | | | | | | PS_T2EX2 | PS_T2EX1 | PS_T2EX0 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PS_T2EX<2:0>: TIMER2 falling edge auto reload input port selection

0x00= Select P02

0x01= Select P12

0x02= Select P22

Other: Reserved

#### 7.3.2.11 PS_RXD0 UART0 RXD Input Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_RXD0 | -- | -- | -- | -- | PS_RXD03 | PS_RXD02 | PS_RXD01 | PS_RXD00 |
| R/W | R | R | R | R | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |

| | | |
|---|---|---|
| Bit7~Bit3 | -- | Reserved, set to 0. |
| Bit2~Bit0 | PS_RXD0<3:0>: | UART0 RXD input port selection |
| | 0x00= | Select P04 |
| | 0x01= | Select P05 |
| | 0x02= | Select P14 |
| | 0x03= | Select P15 |
| | 0x04= | Select P24 |
| | 0x05= | Select P25 |
| | 0x06= | Select P54 |
| | 0x07= | Select P55 |
| | Other: | Reserved |

#### 7.3.2.12 PS_RXD1 UART1 RXD Input Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_RXD1 | -- | PS_RXD16 | PS_RXD15 | PS_RXD14 | PS_RXD13 | PS_RXD12 | PS_RXD11 | PS_RXD10 |
| R/W | R | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| | | |
|---|---|---|
| Bit7 | --: | Reserved, set to 0. |
| Bit7~Bit0 | PS_RXD1mn<6:0>: | UART1 RXD input port selection (m=0,1,2,5; n=0~7) |
| | 0x00= | Allocated to P00 |
| | 0x01= | Allocated to P01 |
| | … | … |
| | 0x07= | Allocated to P07 |
| | 0x10= | Allocated to P10 |
| | 0x11= | Allocated to P11 |
| | … | … |
| | 0x17= | Allocated to P17 |
| | … | … |
| | 0x55= | Allocated to P55 |
| | Other: | Reserved |

#### 7.3.2.13 PS_SCL IIC Communication SCL Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_SCL | -- | -- | -- | -- | -- | PS_SCL2 | PS_SCL1 | PS_SCL0 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 - Reserved, set to 0.

Bit2~Bit0 PS_SCLmn<2:0>: SCL port selection for IIC communication

0x00= Allocated to P00

0x01= Allocated to P10

0x02= Allocated to P20

0x03= Allocated to P50

Other: Reserved

#### 7.3.2.14 PS_SDA IIC Communication SDA Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_SDA | -- | -- | -- | -- | -- | PS_SDA2 | PS_SDA1 | PS_SDA0 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 - Reserved, set to 0.

Bit2~Bit0 PS_SCLmn<2:0>: IIC communication SCL port selection

0x00= Allocated to P01

0x01= Allocated to P11

0x02= Allocated to P21

0x03= Allocated to P51

Other: Reserved

#### 7.3.2.15 PS_TXD0 UART0 TXD Output Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_TXD0 | | | | | | PS_TXD02 | PS_TXD01 | PS_TXD00 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PS_TXD0<3:0>: UART0 TXD output port selection

0x00= Select P04

0x01= Select P05

0x02= Select P14

0x03= Select P15

0x04= Select P24

0x05= Select P25

0x06= Select P54

0x07= Select P55

Other: Reserved

#### 7.3.2.16 PS_TXD1 UART1 TXD Output Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_TXD1 | -- | PS_TXD16 | PS_TXD15 | PS_TXD14 | PS_TXD13 | PS_TXD12 | PS_TXD11 | PS_TXD10 |
| R/W | R | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Bit7 -- Reserved, set to 0.

Bit6~Bit0 PS_TXD1mn<6:0>: UART1 TXD input port selection (m=0,1,2,5; n=0~7)

0x00= Allocated to P00

0x01= Allocated to P01

… …

0x07= Allocated to P07

0x10= Allocated to P10

0x11= Allocated to P11

… …

0x17= Allocated to P17

… …

0x55= Allocated to P55

Other: Reserved

#### 7.3.2.17 PS_SCLKOI SPI SCLK Input/Output Port Selection Register

| F0F0 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_SCLKOI | | | | | | PS_SCLKOI 2 | PS_SCLKOI 1 | PS_SCLKOI 0 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PS_SCLKOI<2:0>: SPI SCLK input/output port selection

0x00= Select P00

0x01= Select P10

0x02= Select P20

0x03= Select P50

Other: Reserved

#### 7.3.2.18 PS_MOSI SPI Master Mode Data Output/Slave Mode Data Input Port Selection Register

| F0F1 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_MOSI | | | | | | PS_MOSI2 | PS_MOSI1 | PS_MOSI0 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PS_MOSI<2:0>: SPI master mode data output/slave mode data input port selection

0x00= Select P01

0x01= Select P11

0x02= Select P21

0x03= Select P51

Other: Reserved

#### 7.3.2.19 PS_MISO SPI Master Mode Data Input/Slave Mode Data Output Port Selection Register

| F0F2 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_MISO | | | | | | PS_MISO2 | PS_MISO1 | PS_MISO0 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PS_MISO<2:0>: SPI master mode data input/slave mode data output port selection

0x00= Select P02

0x01= Select P12

0x02= Select P22

0x03= Select P52

Other: Reserved

#### 7.3.2.20 PS_NSS0OI SPI Slave 0 Chip Select Port Selection Register

| F0F3 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_NSS0OI | | | | | | PS_NSS0OI2 | PS_NSS0OI1 | PS_NSS0OI0 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PS_ NSS0OI<2:0>: SPI slave 0 chip select port selection

0x00= Select P03

0x01= Select P13

0x02= Select P23

0x03= Select P53

Other: Reserved

#### 7.3.2.21 PS_NSS1OI SPI Slave 1 Chip Select Port Selection Register

| F0F4 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_NSS1OI | | | | | | PS_NSS1OI2 | PS_NSS1OI1 | PS_NSS1OI0 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PS_ NSS1OI<2:0>: SPI slave 1 chip select port selection

0x00= Select P04

0x01= Select P14

0x02= Select P24

0x03= Select P54

Other: Reserved

#### 7.3.2.22 PS_NSS2OI SPI Slave 2 Chip Select Port Selection Register

| F0F5 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_NSS2OI | | | | | | PS_NSS2OI2 | PS_NSS2OI1 | PS_NSS2OI0 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PS_ NSS2OI<2:0>: SPI slave 2 chip select port selection

0x00= Select P05

0x01= Select P15

0x02= Select P25

0x03= Select P55

Other: Reserved

#### 7.3.2.23 PS_NSS3OI SPI Slave 3 Chip Select Port Selection Register

| F0F6 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_NSS3OI | | | | | | PS_NSS3OI2 | PS_NSS3OI1 | PS_NSS3OI0 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PS_ NSS3OI<2:0>: SPI slave 3 chip select port selection

0x00= Select P06

0x01= Select P16

0x02= Select P26

Other: Reserved

#### 7.3.2.24 PS_CC0 TIMER2 Channel 0 Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_CC0 | | | | | | PS_CC02 | PS_CC01 | PS_CC00 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PS_CC0<2:0>: TIMER2 channel 0 port selection

0x00= Select P00

0x01= Select P20

Other: Reserved

#### 7.3.2.25 PS_CC1 TIMER2 Channel 1 Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_CC1 | | | | | | PS_CC12 | PS_CC11 | PS_CC10 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PS_CC1<2:0>: TIMER2 channel 1 port selection

0x00= Select P01

0x01= Select P21

Other: Reserved

#### 7.3.2.26 PS_PG0 PWM Channel 0 Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_PG0 | | | | | | PS_PG02 | PS_PG01 | PS_PG00 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PS_PG0<2:0>: PWM channel 0 port selection

0x00= Select P00

0x01= Select P10

0x02= Select P20

0x03= Select P50

Other: Reserved

#### 7.3.2.27 PS_PG1 PWM Channel 1 Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_PG1 | | | | | | PS_PG12 | PS_PG11 | PS_PG10 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PS_PG1<2:0>: PWM channel 1 port selection

0x00= Select P01

0x01= Select P11

0x02= Select P21

0x03= Select P51

Other: Reserved

#### 7.3.2.28 PS_PG2 PWM Channel 2 Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_PG2 | | | | | | PS_PG22 | PS_PG21 | PS_PG20 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PS_PG2<2:0>: PWM channel 2 port selection

0x00= Select P02

0x01= Select P12

0x02= Select P22

0x03= Select P52

Other: Reserved

#### 7.3.2.29 PS_PG3 PWM Channel 3 Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_PG3 | | | | | | PS_PG32 | PS_PG31 | PS_PG30 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PS_PG3<2:0>: PWM channel 3 port selection

0x00= Select P03

0x01= Select P13

0x02= Select P23

0x03= Select P53

Other: Reserved

#### 7.3.2.30 PS_PG4 PWM Channel 4 Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_PG4 | | | | | | PS_PG42 | PS_PG41 | PS_PG40 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PS_PG4<2:0>: PWM channel 4 port selection

0x00= Select P04

0x01= Select P14

0x02= Select P24

0x03= Select P54

Other: Reserved

#### 7.3.2.31 PS_PG5 PWM Channel 5 Port Selection Register

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_PG5 | | | | | | PS_PG52 | PS_PG51 | PS_PG50 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PS_PG5<2:0>: PWM channel 5 port selection

0x00= Select P05

0x01= Select P15

0x02= Select P25

0x03= Select P55

Other: Reserved

#### 7.3.2.32 PS_CLO CLK Output Channel Port Selection Register

| F0F7 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PS_CLO | | | | | | PS_CLO | PS_CLO | PS_CLO |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PS_CLO<2:0>: CLK output channel port selection

0x00= Select P07

0x01= Select P17

0x02= Select P10

Other: Reserved

### 7.3.3 Port Analog Function Configuration

The configuration of analog functions is as shown in the table below:

| Analog function | Port mode | Channel enable control | Remark |
|---|---|---|---|
| ADC channel | High impedance | Controlled by ADCON1 | Set port mode to high impedance PxDIG=0, and enable the channel |
| TK channel | High impedance | | Set port mode to high impedance PxDIG=0, and enable the channel |

### 7.3.4 Port External Interrupt Configuration

The external interrupts for the port include EINT0, EINT1, EINT2, and EINT3. All external interrupts support arbitrary entry mapping. External interrupt configuration is done in the EI0IE, EI0CFGL, and EI0IF registers.

For example, to configure an interrupt for the falling edge trigger on EINT0, the following steps should be performed:

```
P0DIG=0x01;          //Set EINT0 as a digital function
P0TRIS=0x00;         //Set EINT0 as an input
PS_EINT0 =0x00       //Select P00
EI0CFGL= 0x02;       //Configure EINT0 for falling edge trigger interrupt
EI0IE = 0x01;        //Enable the interrupt function for EINT0
EA = 1;              //Enable global interrupts
```

When using an external interrupt, the port must be configured as a digital function, and its direction must be set as input, or the port should be alternated as an input (such as RXD0, etc.).

#### 7.3.4.1 External Extension Interrupt Group 0 Control Register Low Bit (EI0CFGL)

| F0E0H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| EI0CFGL | EI0CFGL7 | EI0CFGL6 | EI0CFGL5 | EI0CFGL4 | EI0CFGL3 | EI0CFGL2 | EI0CFGL1 | EI0CFGL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit6 EIxCFGL<7:6>: External extension interrupt EIx3 control bit
- 00= External interrupt input disabled
- 01= Rising edge triggered interrupt
- 10= Falling edge triggered interrupt
- 11= Both rising and falling edges triggered interrupt

Bit5~Bit4 EIxCFGL<5:4>: External extension interrupt EIx2 control bi
- 00= External interrupt input disabled
- 01= Rising edge triggered interrupt
- 10= Falling edge triggered interrupt
- 11= Both rising and falling edges triggered interrupt

Bit3~Bit2 EIxCFGL<3:2>: External extension interrupt EIx1 control bi
- 00= External interrupt input disabled
- 01= Rising edge triggered interrupt
- 10= Falling edge triggered interrupt
- 11= Both rising and falling edges triggered interrupt

Bit1~Bit0 EIxCFGL<1:0>: External extension interrupt EIx0 control bi
- 00= External interrupt input disabled
- 01= Rising edge triggered interrupt
- 10= Falling edge triggered interrupt
- 11= Both rising and falling edges triggered interrupt

### 7.3.5 Alternated Function Application Notes

1) Port functions have priority constraints. The priority order is as follows: CONFIG Function > Touch Configuration Function > Analog Function > Alternated Function > General IO.

For example, if port P50 is configured as both NRST and PG0, NRST will be valid, and PG0 will be invalid.

2) When multiple alternated functions are set for the same port, the functions are selected according to the following priority (from high to low):

SCL>SDA>TXD0>TXD1>PGx>CCx>CLO

3) When configuring a port as an ADC channel, it is necessary to set the port mode to high-impedance state via software.

4) Input functions and alternated functions are independent:

For example, if port P00 is configured as RXD1 and the EINT0 interrupt trigger for P00 is set to rising edge with interrupt enabled, when the input at P00 transitions from low to high, it will trigger a GPIO interrupt for P00.

5) When configuring a port for PGx or CCx functions, it is recommended to disable pull-up and open-drain features. When configuring a port for SCL or SDA functions, it is recommended to enable pull-up and open-drain features.

# 8. Watch Dog Timer (WDT)

## 8.1 Overview

The Watch dog Timer (WDT) is an on-chip timer with a configurable overflow period, clocked by the LSI (Low-Speed Internal oscillator).

The watch dog overflow reset serves as a system protection mechanism. When the system enters an unknown state, the watch dog can trigger a reset to prevent the system from entering an indefinite deadlock. For details on watch dog overflow reset, refer to Chapter Reset.

1) WDT function disabled (WDKEY==AAH)

When the watch dog function is disabled, the counter value is 0 and the timer is inactive.

2) WDT overflow reset enabled (WDKEY≠AAH)

When the WDT counter reaches the configured overflow value, a WDT overflow reset is generated, and the system reinitializes after reset. To avoid a WDT reset, the watch dog counter can be cleared by writing WDKEY = 0x99 before the overflow occurs. After clearing, the counter restarts counting from 0 until the next overflow.

3) WDT overflow reset forcibly enabled (via CONFIG)

In this mode, the reset function is forcibly enabled (regardless of the WDKEY value). However, the WDT counter can still be cleared by writing WDKEY = 0x99.

The key control register (WDKEY) has a reset value of 0x00. After a power-on reset, the WDT function is enabled by default. To disable it, configure the WDCON and WDKEY registers accordingly. Accessing WDT-related registers requires enabling the WDT clock gate (PCGEN0[3]).

Note: To modify the WDT overflow period, it is recommended to adjust the WTS<2:0> bits immediately after clearing the WDT counter or disable the WDT first. This avoids unintended resets that may occur during the modification of WTS<2:0>.

### 8.1.1 Watch Dog Control Register (WDCON)

| 0x97 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| WDCON | -- | WTS2 | WTS1 | WTS0 | -- | -- | -- | -- |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 -- Reserved, set to 0.

Bit6~Bit4 WTS<2:0>: WDT overflow time select bit

000= $2^8 * T_{LSI}$

001= $2^9 * T_{LSI}$

010= $2^{10} * T_{LSI}$

011= $2^{11} * T_{LSI}$

100= $2^{12} * T_{LSI}$

101= $2^{14} * T_{LSI}$

110= $2^{15} * T_{LSI}$

111= $2^{17} * T_{LSI}$

Bit3~Bit0 -- Reserved, set to 0.

### 8.1.2 Watch Dog Overflow Control Register (WDKEY)

| 0x8F | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| WDKEY | WDKEY7 | WDKEY6 | WDKEY5 | WDKEY4 | WDKEY3 | WDKEY2 | WDKEY1 | WDKEY0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | | |
|---|---|---|
| Bit7~Bit0 | WDKEY<7:0> | WDT key control bit |
| | Write AAH= | Disable WDT |
| | Write 99H= | Clear WDT counter |
| | Other value= | Start WDT counter |

Note:

1. If the WDT is configured as ENABLE in the CONFIG, the WDT counter is forced to start, and the WDT overflow reset function is forcibly enabled.
2. If the WDT is configured as SOFTWARE CONTROL in the CONFIG, the WDKEY can be used to control the counter and reset function.

Instruction sequence to modify WDCON (no other instructions enabled in between):

```
CLR     EA
NOP
MOV     TA,#0AAH
MOV     TA,#055H
ORL     WDCON,#50H    //Set WDT overflow timeWTS=101, it is 256ms.
SETB    EA
```

Instruction sequence to modify WDKEY (no other instructions enabled in between):

```
CLR     EA
NOP
MOV     TA,#0AAH
MOV     TA,#055H
MOV     WDKEY,#AAH    //Clear WDT counter
SETB    EA
```

# 9. Timer0/1

Timers 0 and 1 are similar in type and structure, both being 16-bit timers. Timer 1 has three operational modes, while Timer 0 has four. These timers provide basic timing and event counting functionality.

In timer mode, the timer registers increment every 12 or 4 system cycles when the timer clock is enabled.

In counter mode, Timer 0's timer register increments whenever a falling edge is detected on the corresponding input pin (T0). Similarly, Timer 1's timer register increments on the falling edge detected on the input pin (T1).

## 9.1 Overview

Timers 0 and 1 are fully compatible with the standard 8051 timers.

Each timer is composed of two 8-bit registers: {TH0 (0x8C):TL0 (0x8A)} and{TH1 (0x8D):TL1 (0x8B)}. Timers 0 and 1 operate in four similar modes. The modes for Timer 0 and Timer 1 are described as follows:

| Mode | M1 | M0 | Function description |
|---|---|---|---|
| 0 | 0 | 0 | THx[7:0] and TLx[4:0] form a 13-bit timer/counter |
| 1 | 0 | 1 | THx[7:0] and TLx[7:0] form a 16-bit timer/counter |
| 2 | 1 | 0 | TLx[7:0] forms an 8-bit auto-reload timer/counter, reloaded from THx |
| 3 | 1 | 1 | TL0, TH0 are two 8-bit timers/counters, Timer 1 stops counting |

The THx and TLx registers are special function registers that store the actual timer values. These registers can be combined into a 13-bit or 16-bit register depending on the mode selected. Each time the timer receives an internal clock pulse or an external timer pin experiences a state change, the value in the register increments by 1. The timer starts counting from the initial value loaded into the preset registers. When the timer overflows, an internal interrupt signal is generated. If the timer is set in auto-reload mode, the timer value will reload from the preset value and continue counting. Otherwise, the timer will reset to zero. To maximize the timer/counter range, the preset registers must be cleared before operation.

## 9.2 Relevant Registers

### 9.2.1 Timer0/1 Mode Register (TMOD)

| 0x89 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| TMOD | GATE1 | CT1 | T1M1 | T1M0 | GATE0 | CT0 | T0M1 | T0M0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 GATE1: Timer 1 gate control bit
1= Enable
0= Disable
Bit6 CT1: Timer 1 timer/counter selection bit
1= Counter
0= Timer
Bit5~Bit4 T1M<1:0>: Timer 1 mode selection bit
00= Mode 0 (13-bit timer/counter)
01= Mode 1 (16-bit timer/counter)
10= Mode 2 (8-bit auto-reload timer/counter)
11= Mode 3 (stop counting)
Bit3 GATE0: Timer 0 gate control bit
1= Enable
0= Disable
Bit2 CT0: Timer 0 timer/counter selection bit
1= Counter
0= Timer
Bit1~ Bit0 T0M<1:0>: Timer 0 mode selection bit
00= Mode 0 (13-bit timer/counter)
01= Mode 1 (16-bit timer/counter)
10= Mode 2 (8-bit auto-reload timer/counter)
11= Mode 3 (two independent 8-bit timers/counters)

### 9.2.2 Timer0/1 Control Register (TCON)

| 0x88 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| TCON | TF1 | TR1 | TF0 | TR0 | IE1 | IT1 | IE0 | IT0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 TF1: Timer1 counter overflow interrupt flag bit
1= Timer1 counter overflow occurs, and it enters the interrupt service routine (ISR), where the hardware automatically clears the interrupt flag.
0= Timer1 counter does not overflow.

Bit6 TR1: Timer1 control bit
1= Timer1 is started
0= Timer1 is stopped

Bit5 TF0: Timer0 counter overflow interrupt flag bit
1= Timer0 counter overflow occurs, and it enters the interrupt service routine (ISR), where the hardware automatically clears the interrupt flag.
0= Timer0 counter does not overflow

Bit4 TR0: Timer0 control bit
1= Timer0 is started
0= Timer0 is stopped

Bit3 IE1: External interrupt 1 flag bit
1= External interrupt 1 occurs, and it enters the interrupt service routine (ISR), where the hardware automatically clears the interrupt flag.
0= External interrupt 1 does not generate an interrupt

Bit2 IT1: External interrupt 1 trigger mode control bit
1= Triggered on falling edge
0= Triggered on low level

Bit1 IE0: External interrupt 0 flag bit
1= External interrupt 0 occurs, and it enters the interrupt service routine (ISR), where the hardware automatically clears the interrupt flag.
0= External interrupt 0 does not generate an interrupt

Bit0 IT0: External interrupt 0 trigger mode control bit
1= Triggered on falling edge
0= Triggered on low level

### 9.2.3 Timer0 Data Register Low Bit (TL0)

| 0x8A | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| TL0 | TL07 | TL06 | TL05 | TL04 | TL03 | TL02 | TL01 | TL00 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~ Bit0 TL0<7:0>: Timer 0 low bit data register (also used as counter low bit)

### 9.2.4 Timer0 Data Register High Bit (TH0)

| 0x8C | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| TH0 | TH07 | TH06 | TH05 | TH04 | TH03 | TH02 | TH01 | TH00 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 TH0<7:0>: Timer 0 high bit data register (also used as counter high bit)

### 9.2.5 Timer1 Data Register Low Bit (TL1)

| 0x8B | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| TL1 | TL17 | TL16 | TL15 | TL14 | TL13 | TL12 | TL11 | TL10 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 TL1<7:0>: Timer 1 low bit data register (also used as counter low bit)

### 9.2.6 Timer1 Data Register High Bit (TH1)

| 0x8D | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| TH1 | TH17 | TH16 | TH15 | TH14 | TH13 | TH12 | TH11 | TH10 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 TH1<7:0>: Timer 1 high bit data register (also used as counter high bit)

## 9.3 Timer0/1 Interrupts

Timer0/1 interrupts can be enabled or disabled through the IE (Interrupt Enable) register. The priority of the interrupts can be set to high or low through the IP (Interrupt Priority) register. The interrupt-related bits are as follows:

### 9.3.1 Interrupt Mask Register (IE)

| 0xA8 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| IE | EA | ES1 | ET2 | ES0 | ET1 | EX1 | ET0 | EX0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 EA: Global interrupt enable bit
1= Enable all unmasked interrupts.
0= Disable all interrupts

Bit6 ES1: UART1 interrupt enable bit
1= Enable UART1 interrupt
0= Disable UART1 interrupt

Bit5 ET2: TIMER2 global interrupt enable bit
1= Enable TIMER2 all interrupts
0= Disable TIMER2 all interrupts

Bit4 ES0: UART0 interrupt enable bit
1= Enable UART interrupt
0= Disable UART0 interrupt

Bit3 ET1: TIMER1 interrupt enable bit
1= Enable TIMER1 interrupt
0= Disable TIMER1 interrupt

Bit2 EX1: External interrupt 1 interrupt enable bit
1= Enable external interrupt 1 interrupt
0= Disable external interrupt 1 interrupt

Bit1 ET0: TIMER0 interrupt enable bit
1= Enable TIMER0 interrupt
0= Disable TIMER0 interrupt

Bit0 EX0: External interrupt 0 interrupt enable bit
1= Enable external interrupt 0 interrupt
0= Disable external interrupt 0 interrupt

### 9.3.2 Interrupt Priority Control Register (IP)

| 0xB8 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| IP | -- | PS1 | PT2 | PS0 | PT1 | PX1 | PT0 | PX0 |
| R/W | R | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 -- Reserved, set to 0.

Bit6 PS1: UART1 interrupt priority control bit

1= High-level interrupt

0= Low-level interrupt

Bit5 PT2: TIMER2 interrupt priority control bit

1= High-level interrupt

0= Low-level interrupt

Bit4 PS0: UART0 interrupt priority control bit

1= High-level interrupt

0= Low-level interrupt

Bit3 PT1: TIMER1 interrupt priority control bit

1= High-level interrupt

0= Low-level interrupt

Bit2 PX1: External interrupt 1 interrupt priority control bit

1= High-level interrupt

0= Low-level interrupt

Bit1 PT0: TIMER0 interrupt priority control bit

1= High-level interrupt

0= Low-level interrupt

Bit0 PX0: External interrupt 0 interrupt priority control bit

1= High-level interrupt

0= Low-level interrupt

### 9.3.3 Timer0/1, INT0/1 Interrupt Flag Register (TCON)

| 0x88 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| TCON | TF1 | TR1 | TF0 | TR0 | IE1 | IT1 | IE0 | IT0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 TF1: Timer1 counter overflow interrupt flag bit
- 1= When the Timer1 counter overflows, it enters the interrupt service routine (ISR), and the hardware automatically clears the interrupt flag. It can also be cleared manually via software.
- 0= Timer1 counter does not overflow.

Bit6 TR1: Timer1 control bit
- 1= Timer1 is started
- 0= Timer1 is stopped

Bit5 TF0: Timer0 counter overflow interrupt flag bit
- 1= When the Timer0 counter overflows, it enters the interrupt service routine (ISR), and the hardware automatically clears the interrupt flag. It can also be cleared manually via software.
- 0= Timer0 counter does not overflow

Bit4 TR0: Timer0 control bit
- 1= Timer0 is started
- 0= Timer0 is stopped

Bit3 IE1: External interrupt 1 flag bit
- 1= When External Interrupt 1 is triggered, it enters the interrupt service routine (ISR), and the hardware automatically clears the interrupt flag. It can also be cleared manually via software.
- 0= External interrupt 1 does not generate an interrupt

Bit2 IT1: External interrupt 1 trigger mode control bit
- 1= Triggered on falling edge
- 0= Triggered on low level

Bit1 IE0: External interrupt 0 flag bit
- 1= When External Interrupt 0 is triggered, it enters the interrupt service routine (ISR), and the hardware automatically clears the interrupt flag. It can also be cleared manually via software.
- 0= External interrupt 0 does not generate an interrupt

Bit0 IT0: External interrupt 0 trigger mode control bit
- 1= Triggered on falling edge
- 0= Triggered on low level

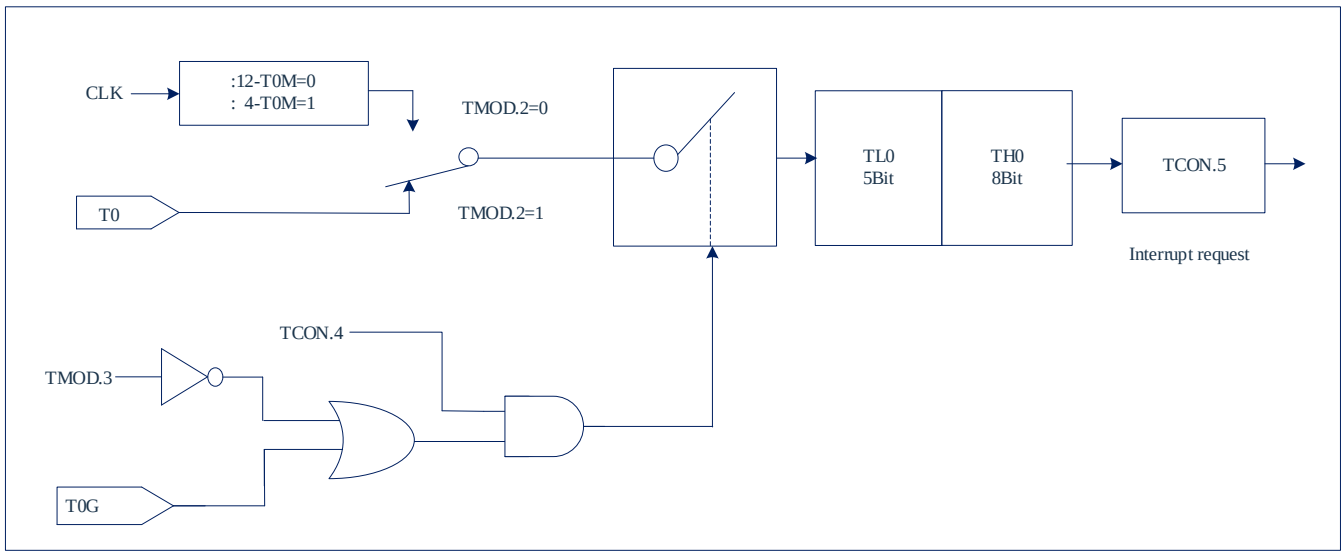
The interrupt flags can be cleared by software, which results in the same outcome as clearing them via hardware. In other words, interrupts can be generated by software (though it is not recommended to generate interrupts by writing to the flag bits) or pending interrupts can be canceled.

The TF0 and TF1 flags can be cleared by writing a 0 to them, even when interrupts are not enabled.

# 9.4 Timer0 Operating Modes

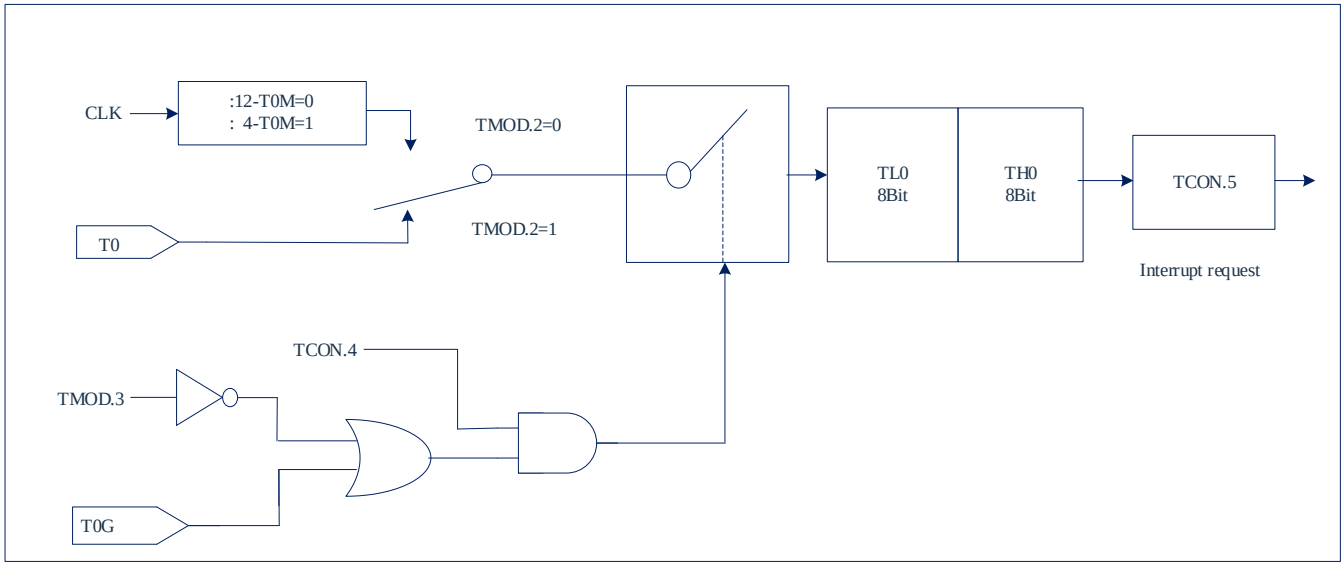
## 9.4.1 T0 - Mode 0 (13-Bit Timer/Counter Mode)

In this mode, Timer0 is a 13-bit register. When all the bits of the counter flip from 1 to 0, the Timer0 interrupt flag (TF0) is set to 1. When TCON.4 = 1 and TMOD.3 = 0, or TCON.4 = 1, TMOD.3 = 1, and T0G = 1, the counting input is enabled to Timer0. (Setting TMOD.3 = 1 allows Timer0 to be controlled by the external T0G pin for pulse width measurement.) The 13-bit register is composed of the low 5 bits of TH0 and TL0. The higher 3 bits of TL0 should be ignored. The block diagram of Timer0 in Mode 0 is shown below:



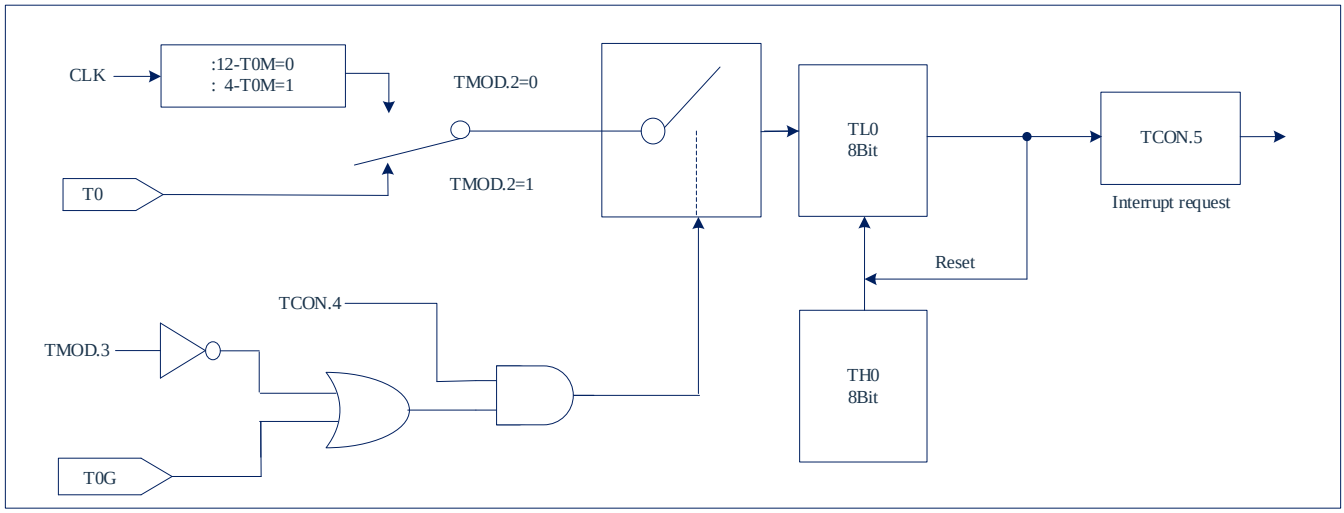

## 9.4.2 T0 - Mode 1 (16-Bit Timer/Counter Mode)

Mode 1 is similar to Mode 0, except that in Mode 1, the entire 16-bit Timer0 data register is used. The block diagram of Timer0 in Mode 1 is shown below:



### 9.4.3 T0 - Mode 2 (8-Bit Auto-Reload Timer/Counter Mode)

In Mode 2, the timer register is an 8-bit counter (TL0) with an auto-reload feature. As shown in the diagram below, the overflow from TL0 not only sets the TF0 flag to 1 but also reloads the content of TH0 into TL0 via software. During the reload, the value of TH0 remains unchanged. The block diagram of Timer0 in Mode 2 is shown below:



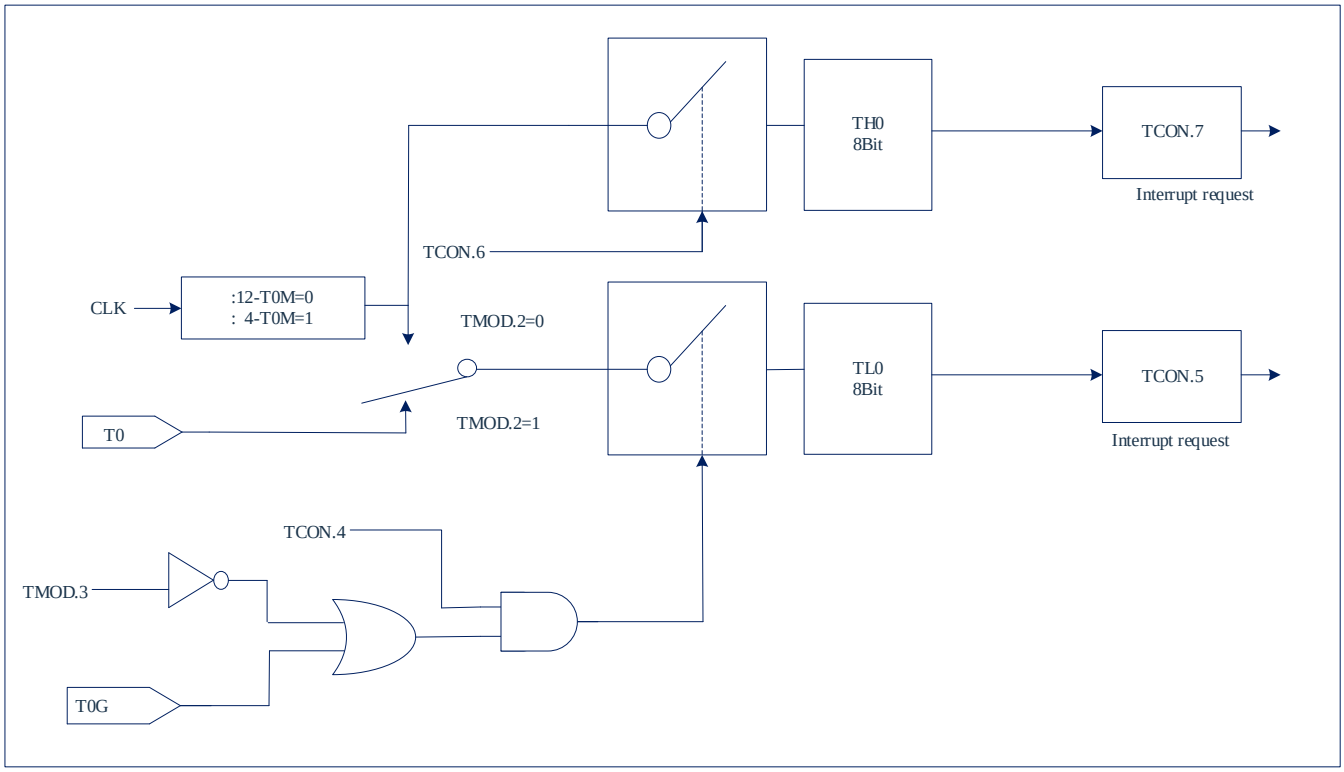

### 9.4.4 T0 - Mode 3 (Two Independent 8-Bit Timers/Counters)

In Mode 3, Timer0 configures TL0 and TH0 as two independent counters. The logic of Timer0 in Mode 3 is shown in the diagram below.

TL0 can operate as a timer or counter and uses Timer0's control bits: such as CT0, TR0, GATE0, and TF0.

TH0 can only operate as a timer and uses Timer1's TR1 and TF1 flags to control Timer1's interrupt.

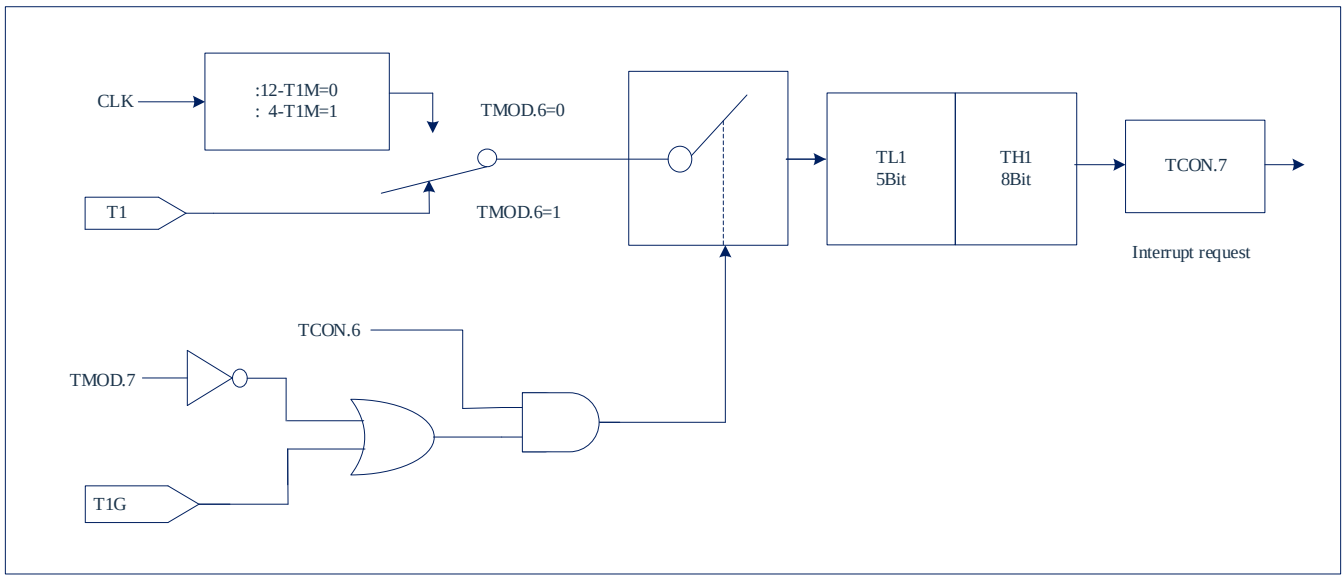
Mode 3 can be used when two 8-bit timers/counters are needed. When Timer0 is in Mode 3, Timer1 can either be disabled by switching to its own Mode 3 or can continue to be used as a baud rate generator via the serial interface, or in any application where Timer1's interrupt is not required. The block diagram of Timer0 in Mode 3 is shown below:



# 9.5 Timer1 Operating Modes

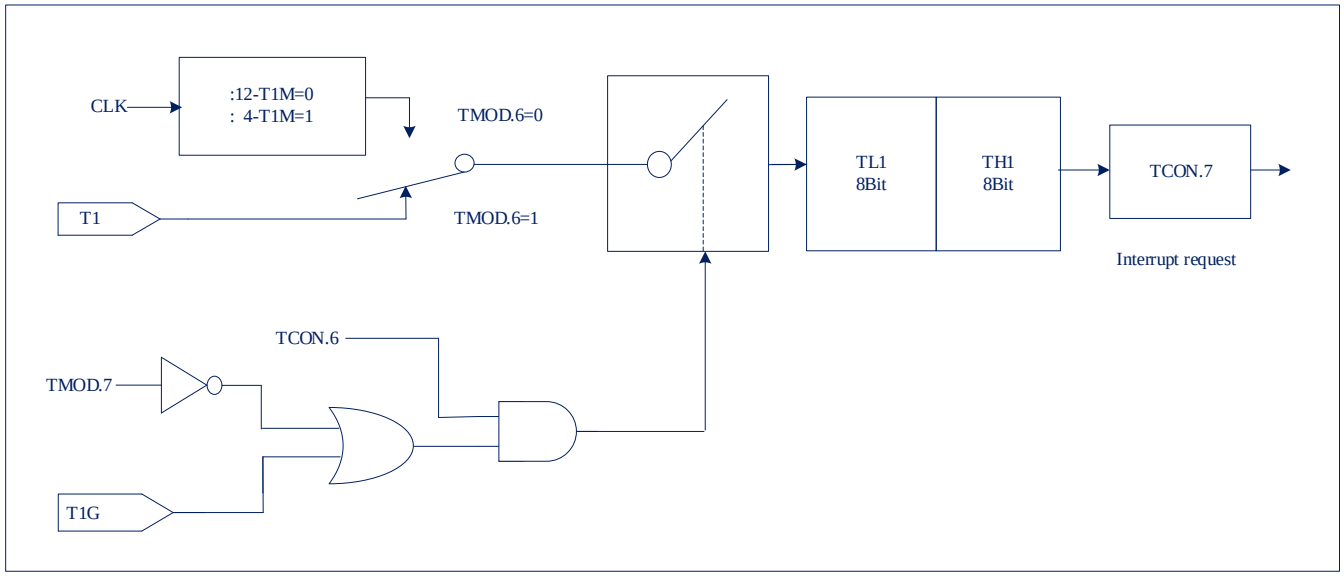
## 9.5.1 T1 - Mode 0 (13-Bit Timer/Counter Mode)

In this mode, Timer1 is a 13-bit register. When all the bits of the counter flip from 1 to 0, the Timer1 interrupt flag (TF1) is set to 1. When TCON.6 = 1 and TMOD.7 = 0, or when TCON.6 = 1, TMOD.7 = 1, and T1G = 1, the counting input is enabled to Timer1. (Setting TMOD.7 = 1 allows Timer1 to be controlled by the external pin T1G, enabling pulse width measurement). The 13-bit register consists of the 8-bit TH1 and the lower 5 bits of TL1. The higher 3 bits of TL1 should be ignored. The block diagram of Timer1 in Mode 0 is shown below:



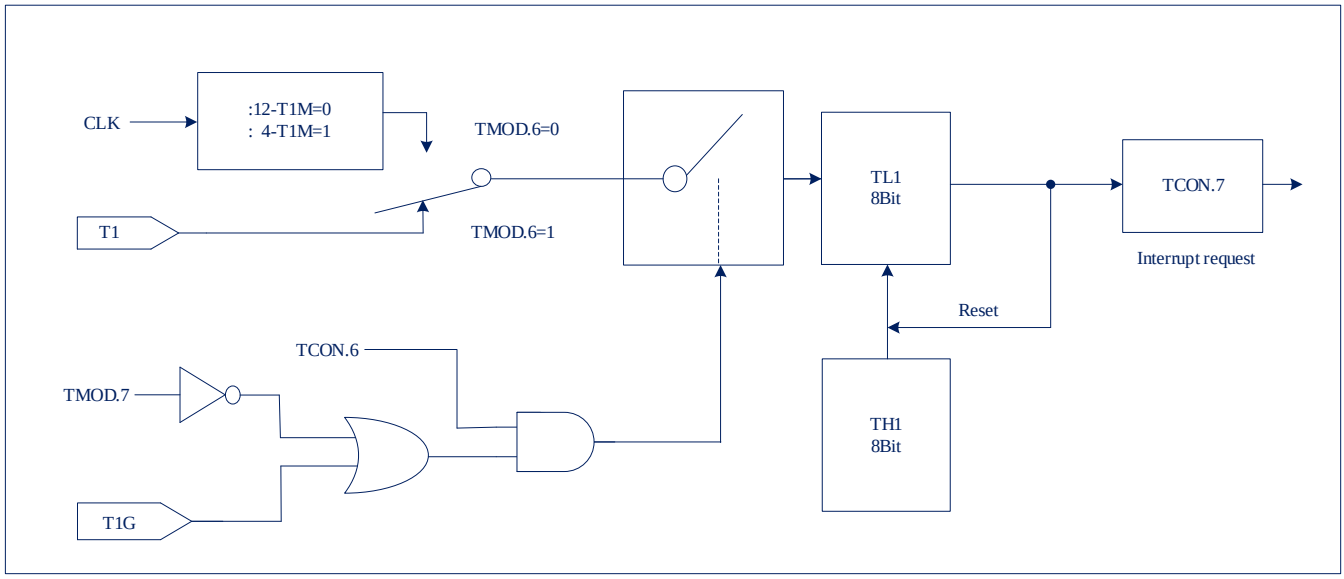

## 9.5.2 T1 - Mode 1 (16-Bit Timer/Counter Mode)

Mode 1 is similar to Mode 0, except that in Mode 1, the entire 16-bit Timer1 register is used for counting. The block diagram of Timer1 in Mode 1 is shown below:



### 9.5.3 T1 - Mode 2 (8-Bit Auto-Reload Timer/Counter Mode)

In Mode 2, Timer1 operates as an 8-bit counter (TL1) with an auto-reload feature, as shown in the diagram below. The overflow from TL1 not only sets the TF1 flag but also reloads the content of TH1 into TL1 through software. During the reload process, the value of TH1 remains unchanged. The block diagram of Timer1 in Mode 2 is shown below.




### 9.5.4 T1 - Mode 3 (Stop Counting)

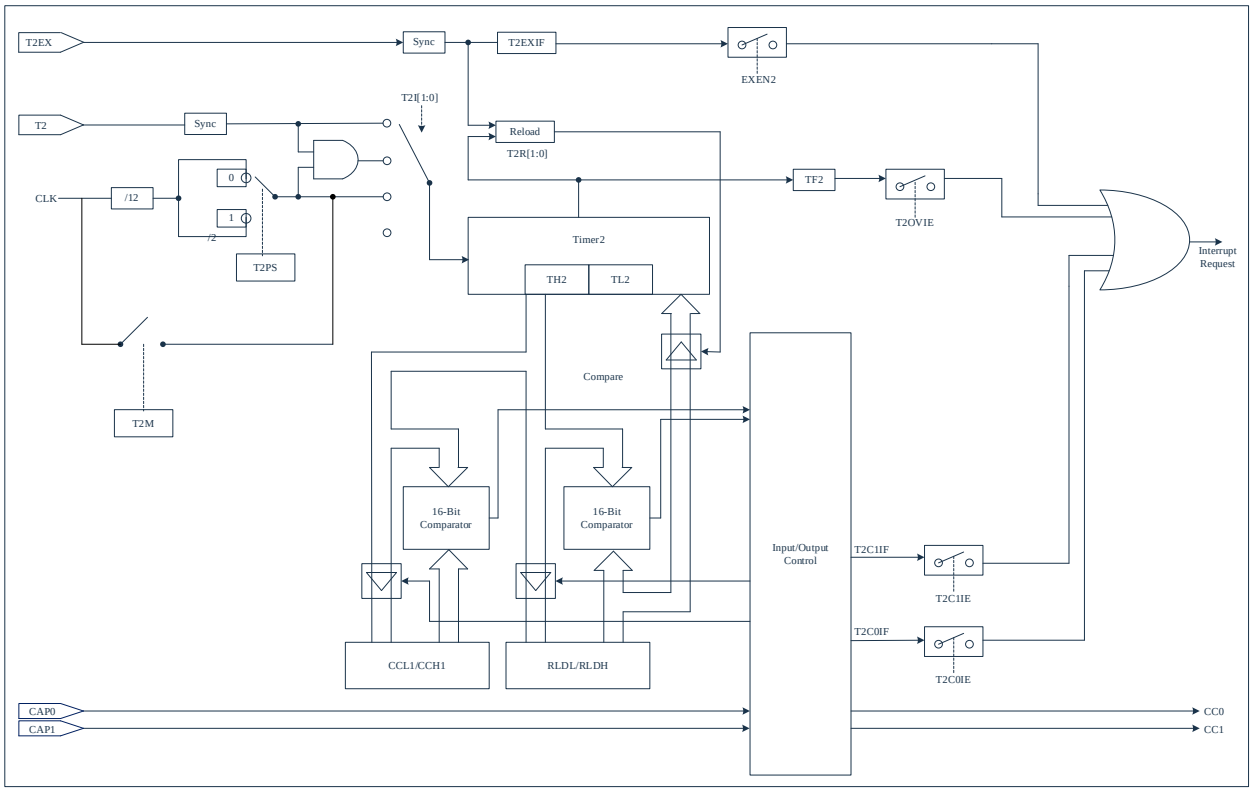
In Mode 3, Timer1 stops counting, which has the same effect as setting TR1 = 0.

# 10. Timer2

Timer2, with additional compare/capture/reload functions, is one of the core peripheral units. It can be used for various digital signal generation and event capture tasks, such as pulse generation, pulse width modulation, pulse width measurement, etc.

## 10.1 Overview

The block diagram of Timer2, which includes additional compare/capture/reload register functions, is shown below:



## 10.2 Relevant Registers

### 10.2.1 Timer2 Control Register (T2CON)

| 0xC8 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| T2CON | T2PS | I3FR | CAPES | T2R1 | T2R0 | T2CM | T2I1 | T2I0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 T2PS: Timer2 clock prescaler selection bit (effective when T2M=0, see CKCON description)
1= Fsys/24
0= Fsys/12

Bit6 I3FR: Capture channel 0 input edge selection and compare interrupt moment selection bit
Capture channel 0 mode:
1= Capture the rising edge to the RLDL/RLDH register
0= Capture the falling edge to the RLDL/RLDH register
Compare channel 0 mode:
1= Interrupt generated when TL2/TH2 and RLDL/RLDH transition from unequal to equal
0= Interrupt generated when TL2/TH2 and RLDL/RLDH transition from equal to unequal

Bit5 CAPES: Capture channel 1 input edge selection
0= Capture the rising edge to the CCL1/CCH1 register
1= Capture the falling edge to the CCL1/CCH1 register

Bit4~Bit3 T2R<1:0>: Timer2 load mode selection bit
0x= Reload disabled
10= Load Mode 1: Automatically reloads on Timer2 overflow
11= Load Mode 2: Reloads on the falling edge of the T2EX pin

Bit2 T2CM: Compare mode selection
1= Compare Mode 1
0= Compare Mode 0

Bit1~Bit0 T2I<1:0>: Timer2 clock input selection bit
00= Timer2 stopped
01= System clock prescaling (controlled by T2PS prescaler selection)
10= External pin T2 as event input (Event counter mode)
11= External pin T2 as gate input (Gate timer mode)

### 10.2.2 Timer2 Data Register Low Bit (TL2)

| 0xCC | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| TL2 | TL27 | TL26 | TL25 | TL24 | TL23 | TL22 | TL21 | TL20 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 TL2<7:0>: Timer 2 low bit data register (also used as counter low bit)

### 10.2.3 Timer2 Data Register High Bit (TTH2)

| 0xCD | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| TH2 | TH27 | TH26 | TH25 | TH24 | TH23 | TH22 | TH21 | TH20 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 TH2<7:0>: Timer 2 high bit data register (also used as counter high bit)

### 10.2.4 Timer2 Compare/Capture/Auto Reload Register Low Bit (RLDL)

| 0xCA | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| RLDL | RLDL7 | RLDL6 | RLDL5 | RLDL4 | RLDL3 | RLDL2 | RLDL1 | RLDL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 RLDL<7:0>: Timer 2 compare/capture/auto-reload register low bit.

### 10.2.5 Timer2 Compare/Capture/Auto Reload Register High Bit (RLDH)

| 0xCB | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| RLDH | RLDH7 | RLDH6 | RLDH5 | RLDH4 | RLDH3 | RLDH2 | RLDH1 | RLDH0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 RLDH<7:0>: Timer 2 compare/capture/auto-reload register high bit.

### 10.2.6 Timer2 Compare/Capture Channel 1 Register Low Bit (CCL1)

| 0xC2 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| CCL1 | CCL17 | CCL16 | CCL15 | CCL14 | CCL13 | CCL12 | CCL11 | CCL10 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 CCL1<7:0>: Timer 2 compare/capture channel 1 register low bit.

### 10.2.7 Timer2 Compare/Capture Channel 1 Register High Bit (CCH1)

| 0xC3 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| CCH1 | CCH17 | CCH16 | CCH15 | CCH14 | CCH13 | CCH12 | CCH11 | CCH10 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 CCH1<7:0>: Timer 2 compare/capture channel 1 register high bit.

### 10.2.8 Timer2 Compare/Capture Control Register (CCEN)

| 0xCE | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| CCEN | -- | -- | -- | -- | CMH1 | CML1 | CMH0 | CML0 |
| R/W | R | R | R | R | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | | |
|---|---|---|
| Bit7~Bit4 | Reserved | Set to 0. |
| Bit3~Bit2 | CMH1-CML1: | Capture/compare mode control bit |
| | 00= | Capture/compare disable |
| | 01= | Capture operation triggered on channel 1 rising or falling edge (CAPES selection) |
| | 10= | Compare mode enable |
| | 11= | Capture operation triggered on writing to CCL1 or dual edge trigger on channel 1 |
| Bit1~Bit0 | CMH0-CML0: | Capture/compare mode control bit |
| | 00= | Capture/compare disable |
| | 01= | Capture operation triggered on channel 0 rising or falling edge (I3FR selection) |
| | 10= | Compare mode enable |
| | 11= | Capture operation triggered on writing to RLDL or dual edge trigger on channel 0 |

## 10.3 Timer2 Interrupts

Timer2 can enable or disable the global interrupt through the register IE, and the priority can be set using the IP register for high/low priority. Timer2 has four types of interrupts:

- Timer Overflow Interrupt
- External Pin T2EX Falling Edge Interrupt
- Compare Interrupt
- Capture Interrupt

To enable the Timer2 interrupts, you need to configure the global interrupt enable bit (EA = 1), the Timer2 global interrupt enable bit (ET2 = 1), and the corresponding interrupt enable bit for the Timer2 interrupt type (T2IE). All four types of Timer2 interrupts share the same interrupt vector. Once entering the interrupt service routine, it is necessary to check the related flag bits to determine which interrupt type occurred.

### 10.3.1 Relevant Registers

#### 10.3.1.1 Interrupt Mask Register (IE)

| 0xA8 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| IE | EA | ES1 | ET2 | ES0 | ET1 | EX1 | ET0 | EX0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 EA: Global interrupt enable bit
1= Enable all unmasked interrupts
0= Disable all interrupts
Bit6 ES1: UART1 interrupt enable bit
1= Enable UART1interrupt
0= Disable UART1interrupt
Bit5 ET2: TIMER2 global interrupt enable bit
1= Enable TIMER2 all interrupts
0= Disable TIMER2 all interrupts
Bit4 ES0: UART0 interrupt enable bit
1= Enable UART0 interrupt
0= Disable UART0 interrupt
Bit3 ET1: TIMER1 interrupt enable bit
1= Enable TIMER1 interrupt
0= Disable TIMER1 interrupt
Bit2 EX1: External interrupt 1 interrupt enable bit
1= Enable external interrupt 1 interrupt
0= Disable external interrupt 1 interrupt
Bit1 ET0: TIMER0 interrupt enable bit
1= Enable TIMER0 interrupt
0= Disable TIMER0 interrupt
Bit0 EX0: External interrupt 0 interrupt enable bit
1= Enable external interrupt 0 interrupt
0= Disable external interrupt 0 interrupt

#### 10.3.1.2 Timer2 Interrupt Mask Register (T2IE)

| 0xCF | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| T2IE | T2OVIE | T2EXIE | -- | -- | -- | -- | T2C1IE | T2C0IE |
| R/W | R/W | R/W | -- | -- | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 T2OVIE: Timer2 overflow interrupt enable bit
1= Enable interrupt
0= Disable interrupt
Bit6 T2EXIE: Timer2 external load interrupt enable bit
1= Enable interrupt
0= Disable interrupt
Bit5~Bit2 -- Reserved, set to 0.
Bit1 T2C1IE: Timer2 compare channel 1 interrupt enable bit
1= Enable interrupt
0= Disable interrupt
Bit0 T2C0IE: Timer2 compare channel 0 interrupt enable bit
1= Enable interrupt
0= Disable interrupt

If the Timer2 interrupt is enabled, you also need to enable the Timer2 global interrupt enable bit ET2 = 1 (IE.5 = 1).

#### 10.3.1.3 Interrupt Priority Control Register (IP)

| 0xB8 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| IP | -- | PS1 | PT2 | PS0 | PT1 | PX1 | PT0 | PX0 |
| R/W | R | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 -- Reserved, set to 0.
Bit6 PS1: UART1 interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit5 PT2: TIMER2 interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit4 PS0: UART0 interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit3 PT1: TIMER1 interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit2 PX1: External interrupt 1 interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit1 PT0: TIMER0 interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit0 PX0: External interrupt 0 interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt

#### 10.3.1.4 Timer2 Interrupt Flag Register (T2IF)

| 0xC9 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| T2IF | TF2 | T2EXIF | -- | -- | -- | -- | T2C1IF | T2C0IF |
| R/W | R/W | R/W | R | R | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 TF2: Timer2 counter overflow interrupt flag
1= When Timer2 counter overflows, it needs to be cleared by software.
0= No Timer2 counter overflow

Bit6 T2EXIF: Timer2 external load flag
1= If a falling edge occurs on the T2EX pin of Timer2, it needs to be cleared by software.
0= --

Bit5~Bit2 -- Reserved, set to 0.

Bit1 T2C1IF: Timer2 compare/capture channel 1 flag
1= Timer2 compare channel 1 {CCH1:CCL1}={TH2:TL2}or a capture operation occurs on Channel 1, it needs to be cleared by software.
0= --

Bit0 T2C0IF: Timer2 compare/capture channel 0 flag
1= If Timer2 Compare Channel 0 (RLDH:RLDL) equals (TH2:TL2) or a capture operation occurs on Channel 0, it needs to be cleared by software.
0= --

### 10.3.2 Timer Interrupt

The timer interrupt enable bit is set by the register T2IE[7], and the interrupt flag can be checked through the register T2IF[7]. When Timer2 overflows, the Timer2 overflow interrupt flag (TF2) will be set to 1.

### 10.3.3 External Trigger Interrupt

The external pin T2EX falling edge trigger interrupt enable bit is set by the register T2IE[6], and the interrupt flag can be checked through the register T2IF[6]. When a falling edge occurs on the T2EX pin, the external load interrupt flag (T2EXIF) will be set to 1.

### 10.3.4 Compare Interrupt

Both two compare channels support compare interrupts. The compare interrupt enable bits are set by the register T2IE[1:0], and the interrupt flags can be checked through the register T2IF[1:0].

Compare Channel 0 allows the selection of the moment when the compare interrupt is generated. When an interrupt occurs, the interrupt flag for Compare Channel 0 (T2C0IF) will be set to 1.

If I3FR = 0, the interrupt occurs when TL2/TH2 changes from not equal to equal to RLDL/RLDH.

If I3FR = 1, the interrupt occurs when TL2/TH2 changes from equal to not equal to RLDL/RLDH.

Compare Channel 1 does not allow the selection of the interrupt trigger moment and is fixed to the moment when TL2/TH2 changes from not equal to equal to CCxL/CCxH. If an interrupt occurs, the corresponding compare channel interrupt flag (T2CxIF) will be set to 1.

### 10.3.5 Capture Interrupt

Both two capture channels support external capture interrupts. The capture interrupt enable bits are set by the register T2IE[1:0], and the interrupt flags can be checked through the register T2IF[1:0]. When a capture operation occurs, the corresponding capture channel interrupt flag (T2CxIF) will be set to 1.

Note: Write operations on capture mode do not generate an interrupt.

## 10.4 Timer2 Function Description

Timer2 is a 16-bit up-counter timer with a clock source from the system clock. Timer2 can be configured in the following functional modes:

- Timer mode
- Reload mode
- Gate timer mode
- Event counting mode
- Compare mode
- Capture mode

By setting Timer2 in different modes, it can be used for generating various digital signals and event capturing, such as pulse generation, pulse-width modulation, pulse-width measurement, etc.

### 10.4.1 Timer Mode

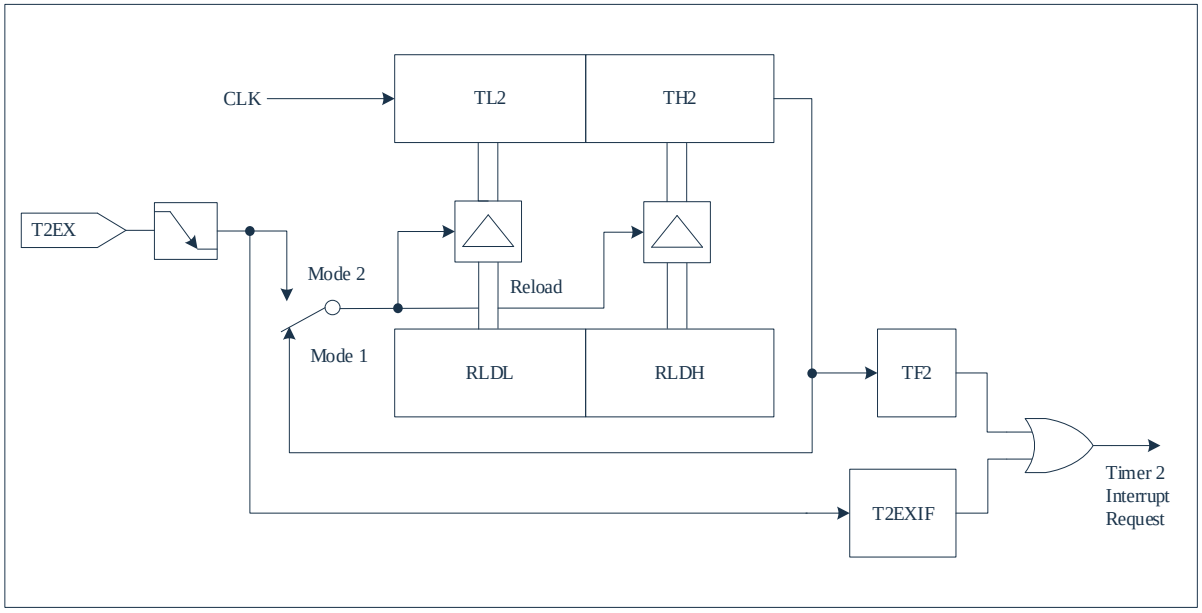
When used as a timer, the clock source is from the system clock. The prescaler provides a choice of the system clock or 1/12 or 1/24 of the system frequency. The value of the prescaler is selected by the T2PS bit in the T2CON register and the T2M bit in the CKCON register. As a result, the 16-bit timer register (composed of TH2 and TL2) increments every clock cycle, every 12 clock cycles, or every 24 clock cycles.

### 10.4.2 Reload Mode

The reload mode of Timer2 is selected by the T2R0 and T2R1 bits in the T2CON register. The reload structure block diagram is shown below.

In Load Mode 1: When the Timer2 counter overflows (all change from 1 to 0), not only is the overflow interrupt flag TF2 set to 1, but Timer2 automatically reloads the 16-bit value from the RLDL/RLDH registers, overriding the counter value of 0x0000. The required RLDL/RLDH values can be preset by software.

In Load Mode 2: The 16-bit reload operation from the RLDL/RLDH registers is triggered by the falling edge of the corresponding T2EX input pin. When the falling edge of T2EX is detected, the external load interrupt flag T2EXIF is set to 1, and Timer2 automatically loads the 16-bit value from the RLDL/RLDH registers as the initial count value.



### 10.4.3 Gate Timer Mode

When Timer2 is used as a gate timer, the external input pin T2 serves as the gate input for Timer2. If the T2 pin is at a high level, the internal clock input is passed through to the timer. If the T2 pin is at a low level, counting is halted. This feature is commonly used to measure pulse width.

### 10.4.4 Event Counting Mode

In event counting mode, Timer2 counts the value by 1 on each falling edge of the external input pin T2. The input signal is sampled on each system clock cycle. When the sampled input shows a high level in one cycle and switches to a low level in the next cycle, the count increases. When another high-to-low transition is detected on the T2 pin in subsequent cycles, the new count value is updated into the timer data register.

### 10.4.5 Compare Mode

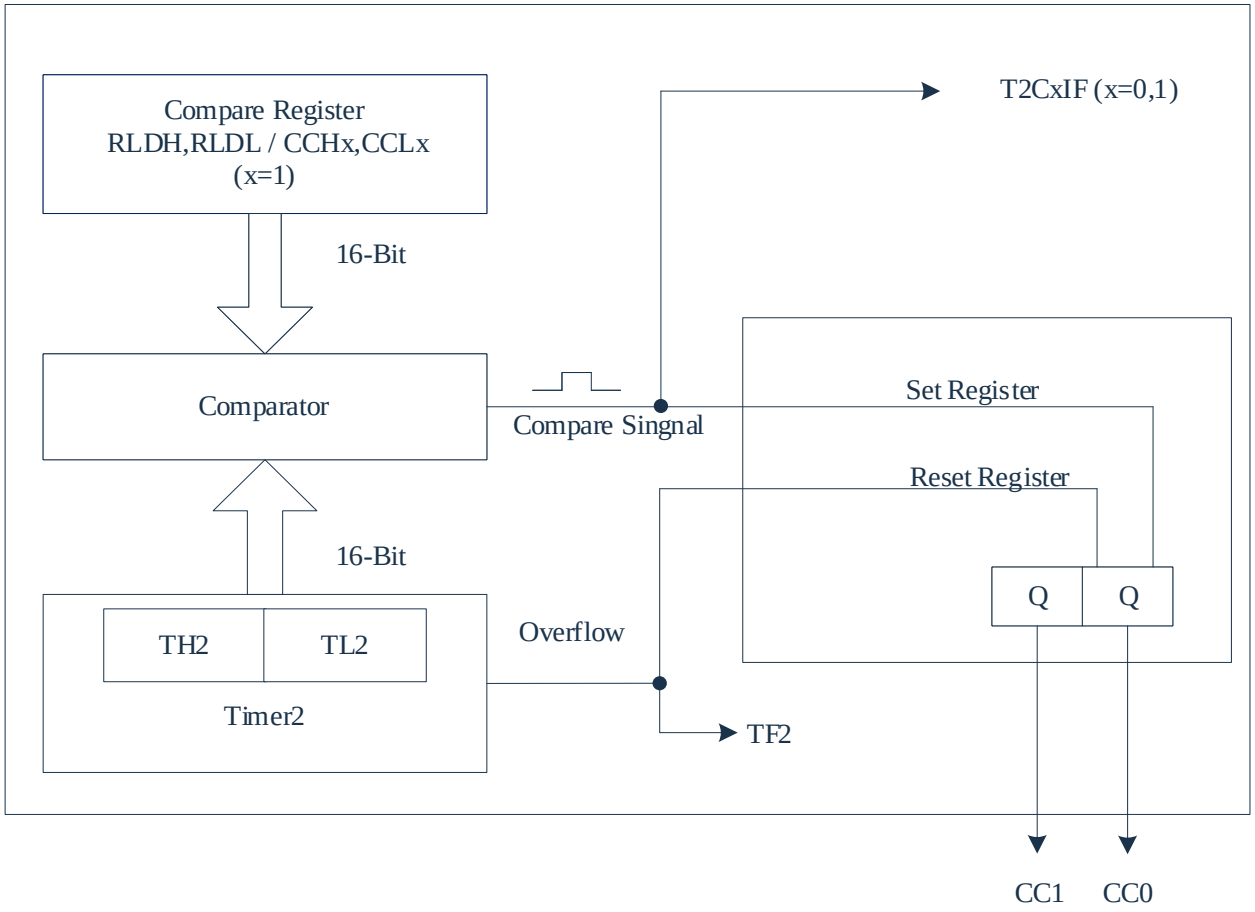
The compare function includes two modes: Compare Mode 0 and Compare Mode 1, which are selected by the T2CM bit in the special function register T2CON. These two compare modes can generate periodic signals and allow for controlling the duty cycle, which is commonly used in pulse width modulation (PWM) applications and other scenarios where continuous square waves are required.

The output channels for the compare function are CC0 and CC1, corresponding to the 16-bit compare registers {RLDH, RLDL} and {CCH1, CCL1}, and the comparison result with the timer's data registers {TH2, TL2}.

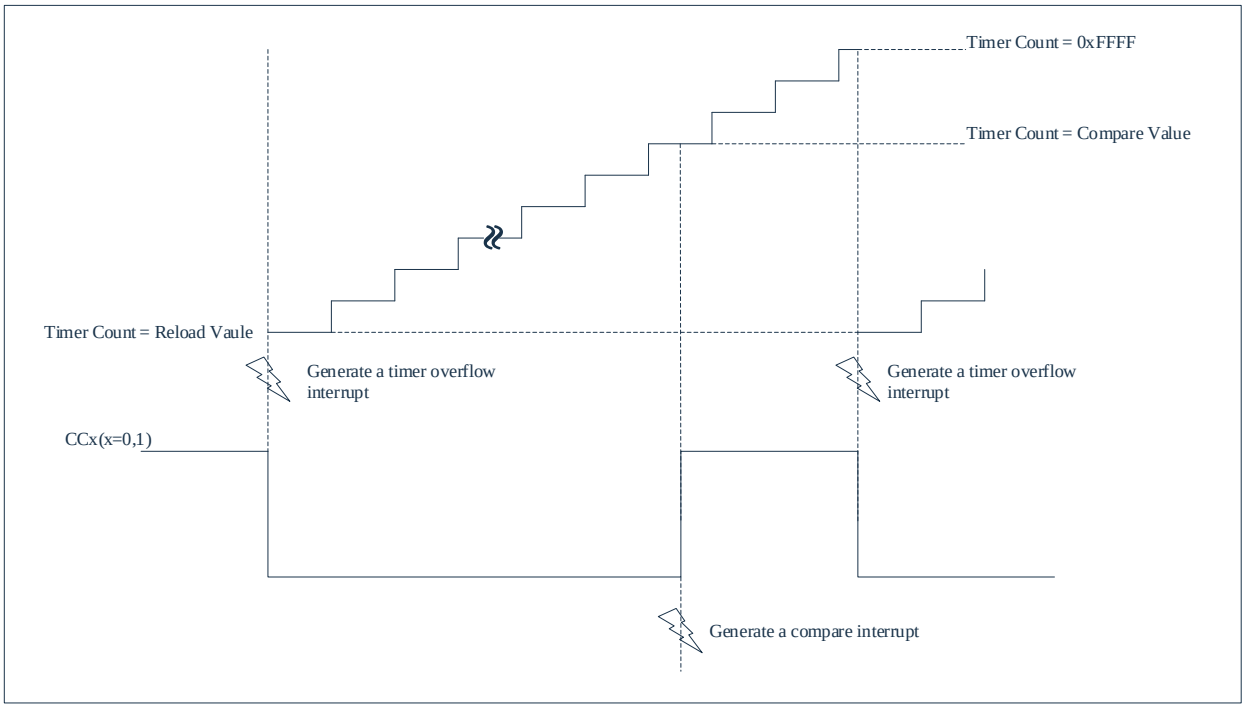
The 16-bit stored values in the compare registers are compared with the timer's count values, and if the values match, an output signal transition is generated on the corresponding port pin, along with an interrupt flag being set.

#### 10.4.5.1 Compare Mode 0

In Mode 0, when the timer count value matches the compare register, the compare output signal changes from low to high. When the timer count overflows, the compare output signal returns to low. The compare output channel is directly controlled by two events: the timer overflow and the compare operation. The block diagram of Compare Mode 0 is shown below:



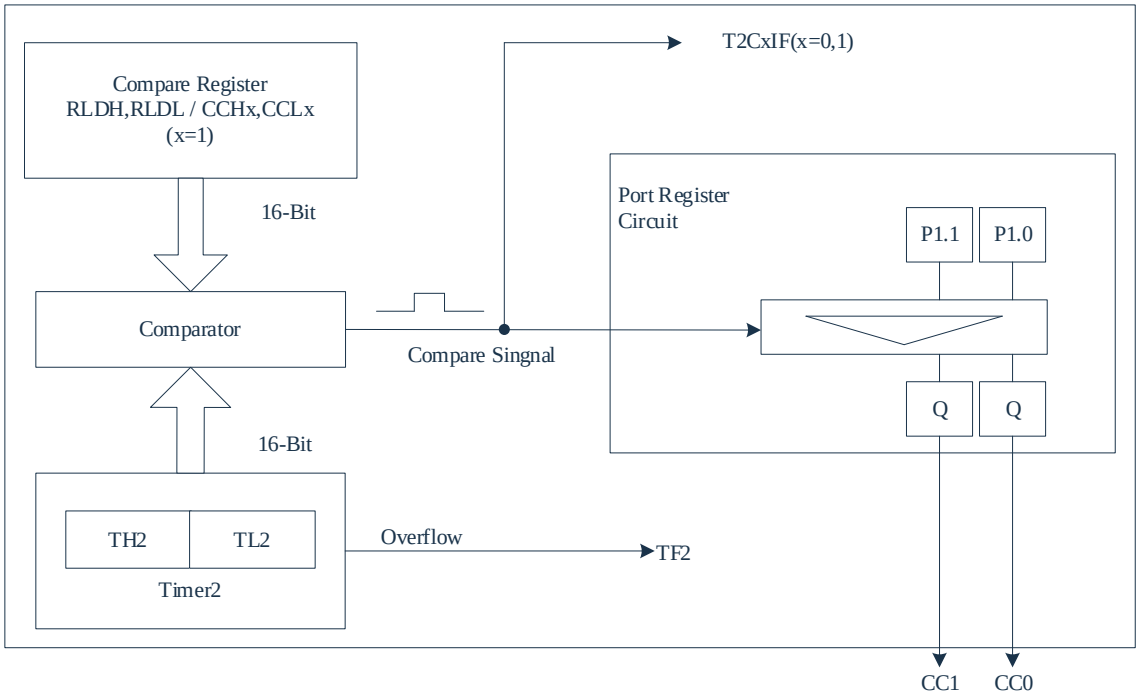

The output block diagram of Compare Mode 0 is shown below:



#### 10.4.5.2 Compare Mode 1

In Compare Mode 1, it is typically used when the output signal transition is independent of the constant signal period, and the software adaptively determines when the output signal will change.

When Mode 1 is enabled, the software writes to the corresponding output register at the CCx port, and the new value will not appear on the output pin until the next compare match occurs. When the timer 2 counter matches the stored compare value, the user can choose whether the output signal will change to a new value or retain its old value. The block diagram of Compare Mode 1 is shown below:




## 10.4.6 Capture Mode

Each pair of 16-bit registers used for the capture function, such as {RLDH, RLDL} or {CCH1, CCL1}, can latch the current 16-bit value of {TH2, TL2}. This function provides two different capture modes.

In Mode 0, an external event can latch the contents of Timer 2 into the capture register.

In Mode 1, the capture operation occurs when the low byte of the 16-bit capture register (RLDL/CCL1) is written. This mode allows software to read the contents of {TH2, TL2} during runtime.

Capture channels 0 and 1 select the capture input pins CAP0 (P14) and CAP1 (P15) as the input source signals. When P14 and P15 are used as Timer 2 capture channels, they must be configured as digital inputs, and the corresponding capture method must be configured in the CCEN register.

#### 10.4.6.1 Capture Mode 0

In Capture Mode 0, a capture event is generated on Capture Channels 0~1 (CAP0~CAP1) whenever a rising transition, falling transition, or both rising and falling transitions occur. When a capture event occurs, the current timer count value is latched into the corresponding capture register.

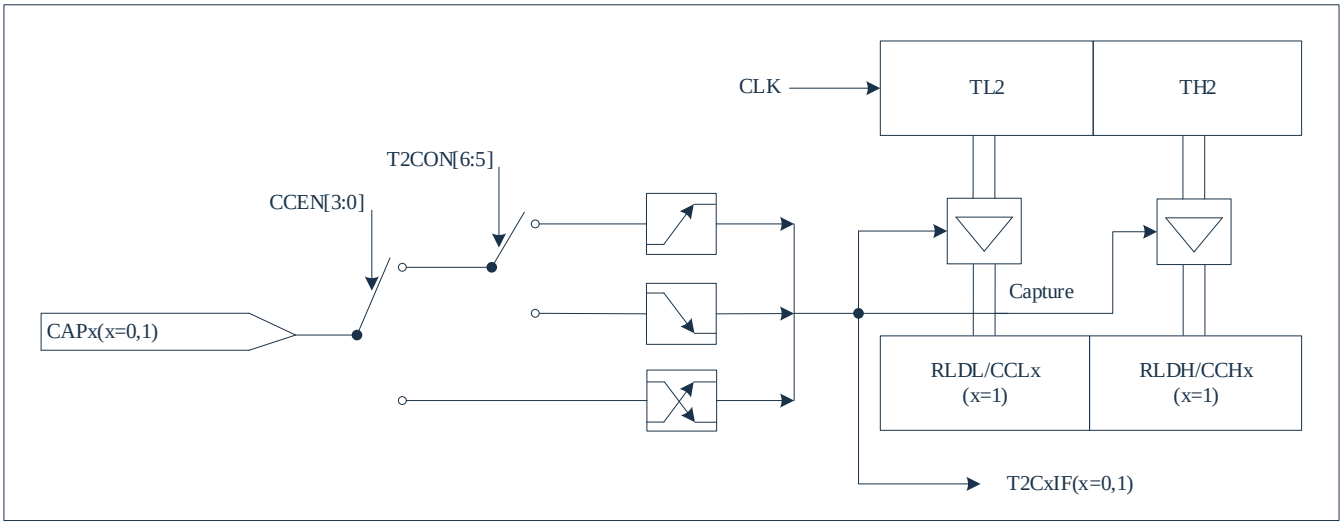
For Capture Channel 0, whether the capture operation is triggered by a rising transition or falling transition depends on the I3FR bit in the T2CON register. If I3FR = 0, the capture is triggered by a falling transition; if I3FR = 1, the capture is triggered by a rising transition.

For Capture Channel 1, whether the capture operation is triggered by a rising transition or falling transition depends on the CAPES bit in the T2CON register. If CAPES = 0, the capture is triggered by a rising transition; if CAPES = 1, the capture is triggered by a falling transition.

Capture Channels 0~1 support dual-transition capture operation. If the corresponding control bit in the CCEN register is set to 11, the channel supports dual-transition capture. It is important to note that in this mode, Capture Mode 1 is also supported, where a write operation can trigger a capture event.

In Capture Mode 0, external capture events on Capture Channels 0~1 can trigger interrupts.

The block diagram of Capture Mode 0 is shown below:

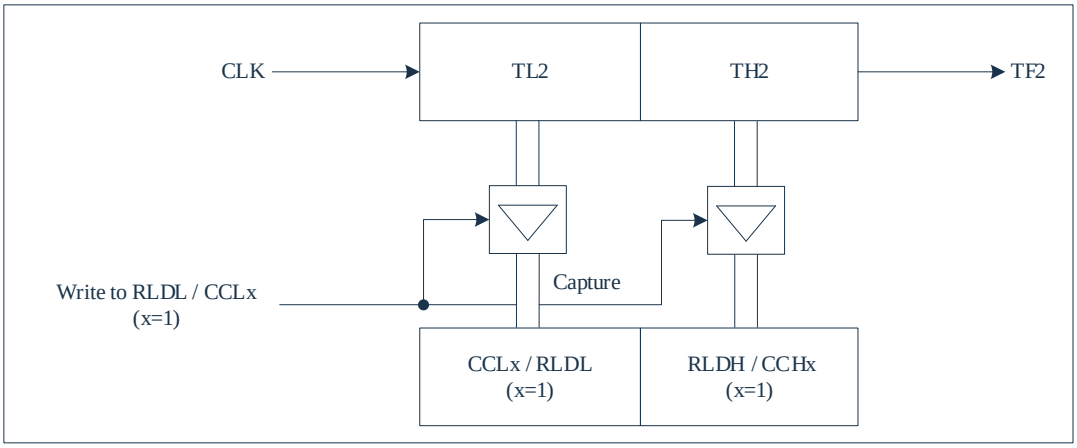


#### 10.4.6.2 Capture Mode 1

In Capture Mode 1, the capture event is triggered by the execution of a low-byte write instruction to the capture register. The write register signal (e.g., write RLDL) initiates the capture operation, and the value written is irrelevant to this function. After the write instruction is executed, the content of Timer 2 will be latched into the corresponding capture register.

In Capture Mode 1, capture events on Capture Channels 0~1 do not generate interrupt request flags.

The block diagram of Capture Mode 1 is shown below:



# 11. Timer5

Timer 5 is a 16-bit timer, and its clock source can be selected from the system clock or the internal low-speed clock (LSI).

When LSI is selected as the counting clock source, it can be used for waking up the system from sleep mode at a scheduled time. Before entering sleep mode, the wake-up time should be configured, and the wake-up function must be enabled. After the chip enters sleep mode, when the counter value equals the set value, the chip enters the wake-up waiting state.

## 11.1 Relevant Registers

### 11.1.1 T5CON Register

| D5H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| T5CON | T5E | T5WUE | T5CKS1 | T5CKS0 | -- | T5PSC2 | T5PSC1 | T5PSC0 |
| R/W | R/W | R/W | R/W | R/W | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | | |
|---|---|---|
| Bit7 | T5E | Timer enable bit |
| | 1= | Start |
| | 0= | Stop |
| Bit6 | T5WUE | Timer wake-up function enable bit |
| | 1= | Enable timer wake-up function |
| | 0= | Disable timer wake-up function |
| Bit5~Bit4 | T5CKS<1:0> | Clock source selection bit |
| | | (for clock switching, the timer enable bit must be disabled first, then the clock source can be selected, and finally, the timer enable bit can be re-enabled) |
| | 00= | System clock |
| | 01= | System clock |
| | 10= | LSI |
| | 11= | Disable configuration |
| Bit3 | -- | Reserved |
| Bit2~Bit0 | T5PSC<2:0>: | Counter clock prescaler |
| | 000= | 1 |
| | 001= | 2 |
| | 010= | 4 |
| | 011= | 8 |
| | 100= | 16 |
| | 101= | 32 |
| | 110= | 64 |
| | 111= | 128 |

### 11.1.2 TL5 Data Register Low Bit

| D3H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| TL5 | TL57 | TL56 | TL55 | TL54 | TL53 | TL52 | TL51 | TL50 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | | |
|---|---|---|
| Bit7~Bit0 | TL5<7:0>: | Timer 5 data register low 8 bits |

### 11.1.3 TH5 Data Register High Bit

| D4H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| TH5 | TH57 | TH56 | TH55 | TH54 | TH53 | TH52 | TH51 | TH50 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 TH5<7:0>: Timer 5 data register high 8 bits

## 11.2 Sleep Mode and Timer Wake-Up

When the counter value of Timer 5 equals the set timer value {TL5/T5H}, the timer interrupt flag (TF5) is set to 1. If global interrupt enable (EA = 1) and timer interrupt enable (ET5 = 1) are both enabled, the CPU will execute the interrupt service routine.

To use Timer 5 for waking up from sleep mode, T5E and T5WUE should be enabled before entering sleep mode, and the wake-up time must be set using {T5H[7:0], T5L[7:0]}.

If global interrupt enable and Timer 5 interrupt enable are turned on before entering sleep mode, upon waking up, the interrupt service routine will be executed first. After the interrupt returns, the next instruction after the sleep instruction will be executed. If the relevant interrupts are not enabled before entering sleep mode, after waking up, the next instruction following the sleep instruction will be executed.

### 11.2.1 Interrupt Mask Register (EIE2)

| 0xAA | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| EIE2 | SPIIE | I2CIE | -- | ADCIE | PWMIE | ET5 | -- | -- |
| R/W | R/W | R/W | R | R/W | R/W | R/W | R | R |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 SPIIE: SPI interrupt enable bit
1= Enable SPI interrupt
0= Disable SPI interrupt
Bit6 I2CIE: $I^2C$ interrupt enable bit
1= Enable $I^2C$ interrupt
0= Disable $I^2C$ interrupt
Bit5 --: Reserved, set to 0.
Bit4 ADCIE: ADC interrupt enable bit
1= Enable ADC interrupt
0= Disable ADC interrupt
Bit3 PWMIE: PWM global interrupt enable bit
1= Enable PWM all interrupts
0= Disable PWM all interrupts
Bit2 ET5: Timer5 interrupt enable bit
1= Enable Timer5 interrupt
0= Disable Timer5 interrupt
Bit1~Bit0 --: Reserved, set to 0.

### 11.2.2 Peripheral Interrupt Flag Register (EIF2)

| 0xB2 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| EIF2 | SPIIF | I2CIF | -- | ADCIF | PWMIF | TF5 | -- | -- |
| R/W | R | R | R | R/W | R | R/W | RW | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 SPIIF: SPI global interrupt flag (read-only)
1= SPI generates an interrupt, (this bit is automatically cleared when the specific interrupt flag bit is cleared)
0= No interrupt generated by SPI
Bit6 I2CIF: $I^2C$ global interrupt flag (read-only)
1= $I^2C$ generates an interrupt, (this bit is automatically cleared when the specific interrupt flag bit is cleared)
0= No interrupt generated by $I^2C$
Bit5 -- Reserved, set to 0.
Bit4 ADCIF: ADC interrupt flag bit
1= ADC conversion is completed, need to be cleared by software.
0= ADC conversion is not completed
Bit3 PWMIF: PWM global interrupt flag (read-only)
1= PWM generates an interrupt, (this bit is automatically cleared when the specific interrupt flag bit is cleared)
0= No interrupt generated by PWM
Bit2 TF5: Timer5 overflow interrupt flag bit
1= Timer5 overflow interrupt flag bit
0= Timer5 timer no overflow
Bit1~Bit0 -- Reserved, set to 0.

### 11.2.3 Interrupt Priority Control Register (EIP2)

| 0xBA | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| EIP2 | PSPI | PI2C | -- | PADC | PPWM | PT5 | -- | -- |
| R/W | R/W | R/W | R | R/W | R/W | R/W | R | R |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 PSPI: SPI interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit6 PI2C: $I^2C$ interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit5 -- Reserved, set to 0.
Bit4 PADC: ADC interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit3 PPWM: PWM interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit2 PT5: TIMER5 interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit1~Bit0 -- Reserved, set to 0.

## 11.3 Function Description

Timer 5 has a 16-bit up-counter. After starting the counter, when the counter value equals {T5H, T5L}, it triggers the interrupt flag, and the counter is reset to zero to restart counting.

Wake-up Timer Principle: After the system enters sleep mode, the CPU and all peripheral circuits stop working, while the low-power oscillator (LSI) continues to function, providing the clock source for the T5 counter. Once the counter triggers the interrupt flag, it will wake up the system, and the MCU will enter the wake-up wait state.

Note 1: Timer 5 can choose either the system clock or LSI as the counting clock source. If you need to switch the clock source during use, you must first disable the timer (T5E = 0), then set the desired clock source (modify T5CKS), and finally re-enable the timer (T5E = 1).

Note 2: When Timer 5 selects LSI as the counting clock source, LSI will continue to operate in sleep mode, regardless of whether the timer wake-up function is enabled or not. That is, when the chip enters sleep mode, the counter will continue counting.

Note 3: In sleep mode, if the Timer 5 wake-up function is not required to wake up the system, ensure that Timer 5 is disabled to reduce power consumption. If the Timer 5 wake-up function is needed, the wake-up function (T5WUE = 1) must be configured before the chip enters sleep mode. Once the counter reaches the value of {T5L/T5H}, it will wake up the system, and the MCU will enter the wake-up wait state, with the counter being cleared and restarting its count.

Note 4: If the values of the data registers {T5L/T5H} are modified during the Timer 5 counting process, the counter will continue counting until it reaches the original values in the data registers. Afterward, the counter will load the modified values from the data registers and continue counting.

# 12. Baud Rate Timer (BRT)

## 12.1 Overview

The chip contains a 10-bit Baud Rate Timer (BRT) that primarily provides the clock for the UART module.

## 12.2 Relevant Registers

### 12.2.1 BRT Timer Data Load Low 8-Bit Register (BRTDL)

| F5C1H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| BRTDL | BRTDL7 | BRTDL6 | BRTDL5 | BRTDL4 | BRTDL3 | BRTDL2 | BRTDL1 | BRTDL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 BRTDL<7:0>: Low 8 bits of the BRT timer load value.

### 12.2.2 BRT Timer Data Load High 2-Bit Register (BRTDH)

| F5C2H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| BRTDH | BRTEN | -- | -- | -- | -- | -- | BRTDH1 | BRTDH0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 BRTEN: BRT timer enable bit
1= Enable
0= Disable
Bit6~Bit2 -- Reserved
Bit1~Bit0 BRTDH<1:0>: High 2 bits of the BRT timer load value

## 12.3 Function Description

The BRT contains a 10-bit incrementing counter. The clock source for this counter comes from the prescaler circuit, with the clock selection determined by the timer prescaler control bit BRTCKDIV. The counter starts with an initial value of 0.

When the BRT enable bit (BRTEN = 1) is set, the counter starts working. When the 10-bit counter reaches the value set in {BRTDH, BRTDL}, the BRT counter overflows. After the overflow, the counter automatically resets to 0 and starts counting again.

The BRT overflow signal is specifically provided to the UART module as the clock source for baud rate generation. The overflow does not trigger an interrupt, and no corresponding interrupt structure is present. In debug mode, the BRT continues to run and its clock does not stop. If the UART module has already started transmitting or receiving data, the UART will complete the transmission or reception process even if the chip enters a suspended state.

BRT Timer Overflow Rate:

$$BRTov=\frac{Fsys}{\{BRTDH,BRTDL+1\}}$$

# 13. PWM Module

## 13.1 Overview

The PWM module supports 6 PWM generators, allowing the configuration of 6 PWM outputs (PG0-PG5). It can also be configured as 3 pairs of synchronized PWM outputs or 3 pairs of complementary PWM outputs with programmable dead-time generators. The pairs are PG0-PG1, PG2-PG3, and PG4-PG5.

Each pair of PWM shares a 16-bit period register, and each PWM channel has its own 16-bit duty cycle register (compare data register) to configure the PWM signal's period and adjust the duty cycle. Each pair of PWM also has independent clock division control registers with 8 selectable clock prescalers.

Each pair of PWM can be configured to operate in edge-aligned counting mode. It can also be set to one-shot mode (producing one PWM signal period) or auto-reload mode (continuously outputting the PWM waveform). The output polarity for each PWM channel can be controlled using the output polarity controller.

The PWM module also supports interrupt functionality, with 9 interrupt flags available for the 6 PWM generators, including 3 zero-crossing interrupts and 6 down-counting comparison interrupts. These interrupts share a single interrupt vector entry.

## 13.2 Features

The PWM module includes the following features:

- 6-channel 16-bit PWM control mode
  - 3 pairs of complementary PWM outputs: (PG0-PG1), (PG2-PG3), (PG4-PG5), with programmable dead-time insertion.
  - 3 pairs of synchronized PWM outputs: (PG0-PG1), (PG2-PG3), (PG4-PG5), where each pair of PWM output pins is synchronized.
- Group control support, where PG0, PG2, and PG4 outputs are synchronized, and PG1, PG3, and PG5 outputs are synchronized.
- Edge-aligned mode supported.
- One-shot mode or auto-reload mode supported.
- Independent polarity control for each PWM channel.

## 13.3 Port Configuration

Before using the PWM module, the corresponding pins need to be configured as PWM channels. PWM channels are labeled as PG0~PG5 in the alternate function allocation table, which corresponds to PWM channels 0~5.

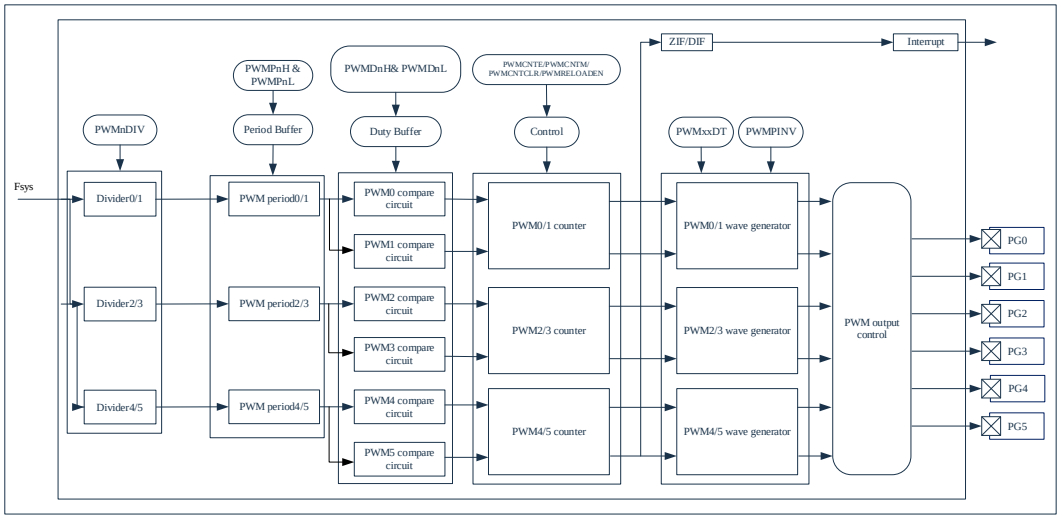
The port configuration for the PWM channels is controlled by the corresponding port configuration registers. For example:

```
PS_PG0 = 0x00;      // Configure P00 as PG0
PS_PG1 = 0x00;      // Configure P01 as PG1
PS_PG2 = 0x00;      // Configure P02 as PG2
PS_PG3 = 0x00;      // Configure P03 as PG3
PS_PG4 = 0x00;      // Configure P04 as PG4
PS_PG5 = 0x00;      // Configure P05 as PG5
```

# 13.4 Function Description

## 13.4.1 Functional Block Diagram

The PWM module consists of several components, including the clock control module, PWM counter module, output comparator unit, waveform generator, and output controller. The structural block diagram is shown below:



### 13.4.2 Edge-Alignment Mode

In edge-aligned mode, the 16-bit PWM counter (CNTn) starts counting down at the beginning of each cycle. It is compared with the value in the PWMDnH/PWMDnL registers (CMPn). When CNTn equals CMPn, PGn outputs a high level, and the PWMnDIF interrupt flag is set to 1. The counter continues counting down to zero, at which point PGn will output a low level, and the PWMnZIF interrupt flag is set to 1. When the CNTn counter reaches zero, if PWMnCNTM is set to 1, both CMPn and PERIODn will be reloaded.

The parameters related to edge alignment are as follows:

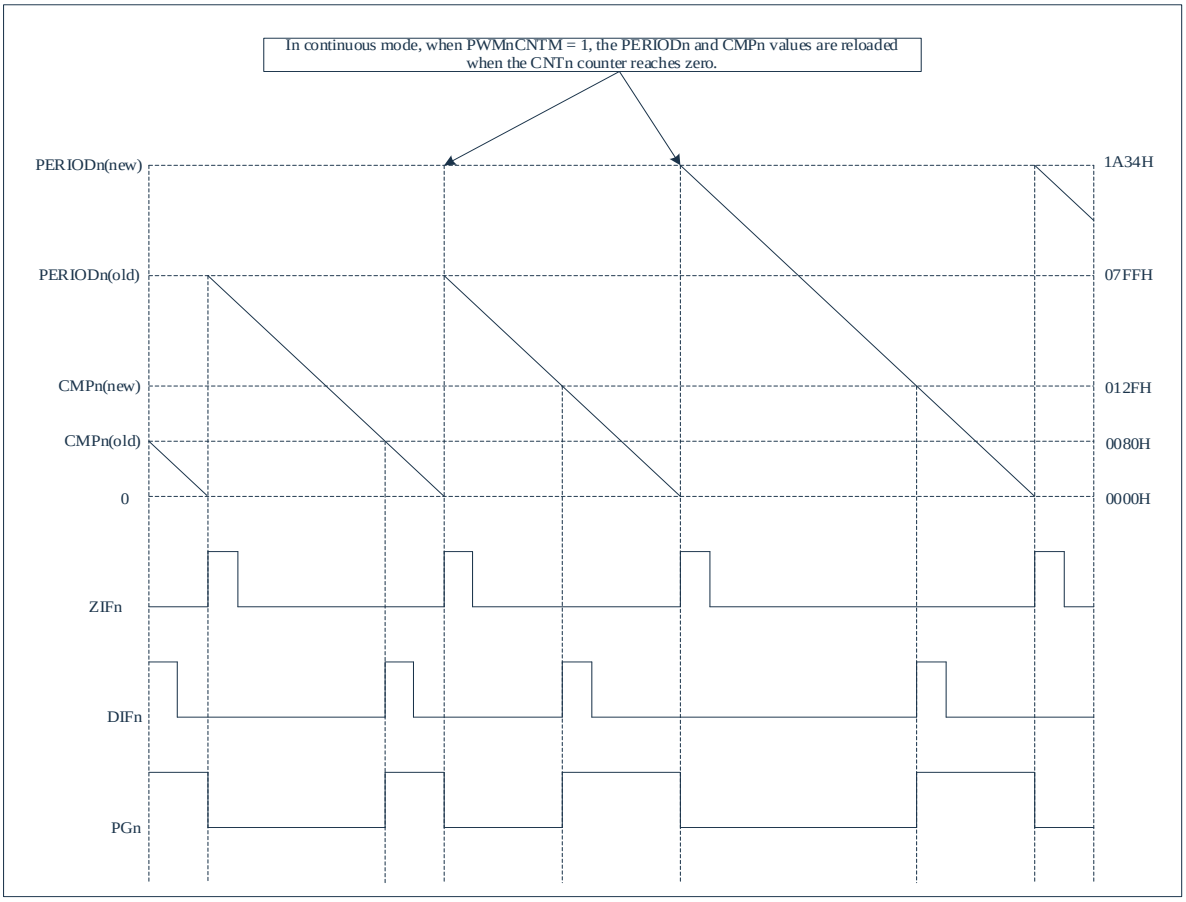
High Level Time= $(CMPn+1)\times Tpwm$(CMPn≥1)

Period= $(PERIODn+1)\times Tpwm$

Duty Cycle =(CMPn+1) / (PERIODn+1)  (CMPn≥1)

When CMPn = 0, the duty cycle will be 0%.

The edge-alignment timing diagram is shown below:



### 13.4.3 Complementary Mode

The 6 PWM channels can be configured as 3 pairs of complementary PWM outputs. In complementary mode, the period and duty cycle of PG1, PG3, and PG5 are determined by the corresponding registers of PG0, PG2, and PG4. In other words, except for the output enable control bits (PWMnOE) and polarity control, the output waveforms of PG1, PG3, and PG5 are no longer controlled by their own registers.

In complementary mode, each pair of complementary PWM channels supports the insertion of dead-time delays. The inserted dead-time is as follows:

PWM0/1 Dead Time: $(PWM01DT+1)*T_{PWM0}$;

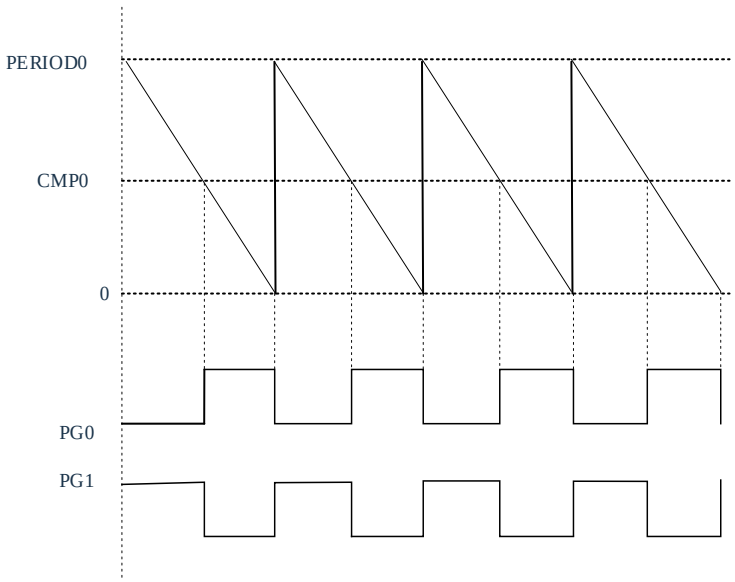
PWM2/3 Dead Time: $(PWM23DT+1)*T_{PWM2}$;

PWM4/5 Dead Time: $(PWM45DT+1)*T_{PWM4}$;

Where $T_{PWM0}$, $T_{PWM2}$, and $T_{PWM4}$ are the clock source periods for PG0, PG2, and PG4, respectively.

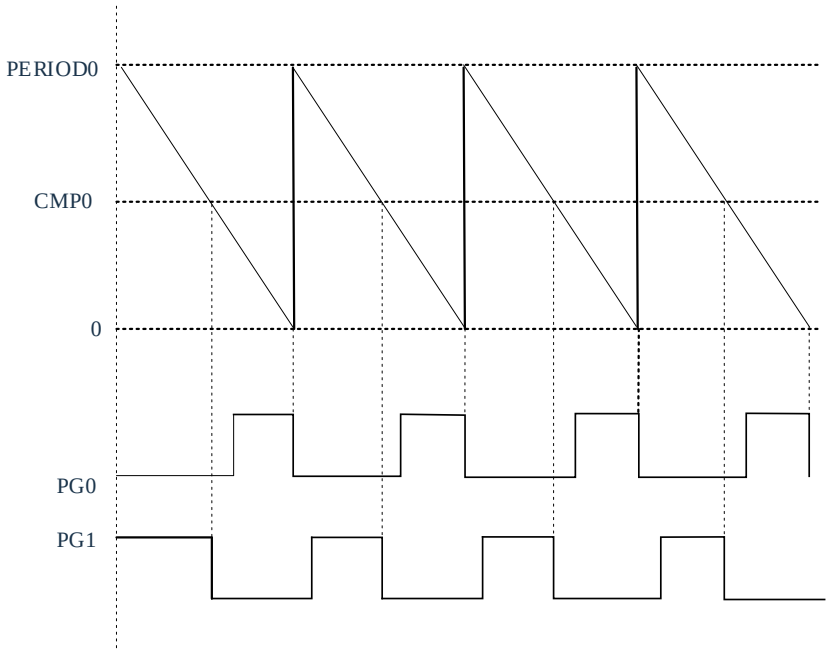
Taking PG0/PG1 as an example, the waveform in complementary mode without dead-time is shown in the diagram below:

Complementary PWM without dead-time



Taking PG0/PG1 as an example, the waveform with dead-time in complementary mode is shown in the diagram below:

Complementary PWM with dead-time



### 13.4.4 Synchronous Mode

The 6 PWM channels can be configured as 3 pairs of synchronous PWM outputs. In synchronous mode, the period and duty cycle of PG1, PG3, and PG5 are determined by the corresponding registers of PG0, PG2, and PG4. In other words, except for the output enable control bits (PWMnOE) and polarity control, the output waveforms of PG1, PG3, and PG5 are no longer controlled by their own registers. The output waveform of PG1 is the same as PG0, the output waveform of PG3 is the same as PG2, and the output waveform of PG5 is the same as PG4.

## 13.5 PWM Relevant Registers

### 13.5.1 PWM Control Register (PWMCON)

| F120H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PWMCON | -- | PWMRUN | PWMMODE1 | PWMMODE0 | GROUPEN | -- | -- | -- |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | | |
|---|---|---|
| Bit7 | -- | Reserved, set to 0. |
| Bit6 | PWMRUN: | PWM clock prescaler and mode control |
| | 1= | Disable (PWMmnDIV is cleared to 0) |
| | 0= | Enable |
| Bit5~Bit4 | PWMMODE<1:0>: | PWM mode control bit |
| | 00= | Independent mode |
| | 01= | Complementary mode |
| | 10= | Synchronous mode |
| | 11= | Reserved |
| Bit3 | GROUPEN: | PWM group function enable bit |
| | 1= | PG0 controls PG2, PG4; PG1 controls PG3, PG5; |
| | 0= | All PWM channel signals are independent of each other. |
| Bit2~Bit0 | -- | Reserved, set to 0. |

### 13.5.2 PWM Output Enable Control Register (PWMOE)

| F121H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PWMOE | -- | -- | PWM5OE | PWM4OE | PWM3OE | PWM2OE | PWM1OE | PWM0OE |
| R/W | R | R | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | | |
|---|---|---|
| Bit7~Bit6 | -- | Reserved, set to 0. |
| Bit5 | PWM5OE: | PWM channel 5 output enable bit |
| | 1= | Enable |
| | 0= | Disable |
| Bit4 | PWM4OE: | PWM channel 4 output enable bit |
| | 1= | Enable |
| | 0= | Disable |
| Bit3 | PWM3OE: | PWM channel 3 output enable bit |
| | 1= | Enable |
| | 0= | Disable |
| Bit2 | PWM2OE: | PWM channel 2 output enable bit |
| | 1= | Enable |
| | 0= | Disable |
| Bit1 | PWM1OE: | PWM channel 1 output enable bit |
| | 1= | Enable |
| | 0= | Disable |
| Bit0 | PWM0OE: | PWM channel 0 output enable bit |
| | 1= | Enable |
| | 0= | Disable |

### 13.5.3 PWM Clock Division Control Register (PWMnDIV(n=0-2))

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PWMnDIV | -- | -- | -- | -- | -- | PWMnDIV2 | PWMnDIV1 | PWMnDIV0 |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register PWMnDIV(n=0-2) Address: F12AH, F12BH, F12CH.

(n=0, corresponding to PWM channels 0,1; n=1, corresponding to PWM channels 2,3; n=2, corresponding to PWM channels 4,5)

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PWMnDIV<2:0>: PWM channel n clock division control bit

000= Fsys
001= Fsys/2
010= Fsys/4
011= Fsys/8
100= Fsys/16
101= Fsys/32
110= Fsys/64
111= Fsys/128

### 13.5.4 PWM Data Load Enable Control Register (PWMLOADEN)

| F129H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PWMLOADEN | -- | -- | -- | -- | -- | PWM2LE | PWM1LE | PWM0LE |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PWMnLE: PWM channel n data load enable bit (n=0-2) (Once the data is loaded, the hardware automatically clears the bit.)

(n=0, corresponds to PWM channels 0,1; n=1, corresponds to PWM channels 2,3; n=2, corresponds to PWM channels 4,5)

1= Enable load cycle, duty cycle data (PERIODn, CMPn)

0= Writing 0 has no effect.

### 13.5.5 PWM Output Polarity Control Register (PWMPINV)

| F122H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PWMPINV | -- | -- | PWM5PINV | PWM4PINV | PWM3PINV | PWM2PINV | PWM1PINV | PWM0PINV |
| R/W | R | R | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit6 -- Reserved, set to 0.

Bit5~Bit0 PWMnPINV: PWM channel n output polarity control bit (n=0-5)

1= Reverse output

0= Non-reverse output

### 13.5.6 PWM Counter Mode Control Register (PWMCNTM)

| F127H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PWMCNTM | -- | -- | -- | -- | -- | PWM2CNTM | PWM1CNTM | PWM0CNTM |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PWMnCNTM: PWMchannel n counter mode control bit (n=0-2)
(n=0, corresponds to PWM channels 0,1; n=1, corresponds to PWM channels 2,3; n=2, corresponds to PWM channels 4,5)

1= Auto-load mode

0= One-shot mode

### 13.5.7 PWM Counter Enable Control Register (PWMCNTE)

| F126H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PWMCNTE | -- | -- | -- | -- | -- | PWM2CNTE | PWM1CNTE | PWM0CNTE |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PWMnCNTE: PWM channel n counter enable control bit (n=0-2)
(n=0, corresponds to PWM channels 0,1; n=1, corresponds to PWM channels 2,3; n=2, corresponds to PWM channels 4,5)

1= PWMn counter started (PWMn outputs started);

0= PWMn counter stopped (Writing 0 stops the counter and clears the counter value).
(In one-shot mode, the hardware automatically clears this bit after completing the cycle.)

### 13.5.8 PWM Counter Mode Control Register (PWMCNTCLR)

| F128H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PWMCNTCLR | -- | -- | -- | -- | -- | PWM2CNTCLR | PWM1CNTCLR | PWM0CNTCLR |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PWMnCNTCLR: PWM channel n counter clear control bit (n=0-2) (cleared automatically by hardware)
(n=0, corresponds to PWM channels 0,1; n=1, corresponds to PWM channels 2,3; n=2, corresponds to PWM channels 4,5)

1= Clear the PWMn counter

0= Writing 0 has no effect

### 13.5.9 PWM Period Data Register Lower 8 Bits PWMPnL (n=0-2)

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PWMPnL | PWMPnL7 | PWMPnL6 | PWMPnL5 | PWMPnL4 | PWMPnL3 | PWMPnL2 | PWMPnL1 | PWMPnL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register PWMPnL (n=0-2) Address: F130H, F134H, F138H.

(n=0, corresponds to PWM channels 0,1; n=1, corresponds to PWM channels 2,3; n=2, corresponds to PWM channels 4,5)

Bit7~Bit0 PWMPnL<7:0>: PWM channel n period data register lower 8 bits.

### 13.5.10 PWM Period Data Register Higher 8 Bits PWMPnH (n=0-2)

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PWMPnH | PWMPnH7 | PWMPnH6 | PWMPnH5 | PWMPnH4 | PWMPnH3 | PWMPnH2 | PWMPnH1 | PWMPnH0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register PWMPnH (n=0-2) Address: F131H, F135H, F139H.

(n=0, corresponds to PWM channels 0,1; n=1, corresponds to PWM channels 2,3; n=2, corresponds to PWM channels 4,5)

Bit7~Bit0 PWMPnH<7:0>: PWM channel n period data register higher 8 bits.

### 13.5.11 PWM Compare Data Register Lower 8 Bits PWMDnL (n=0-5)

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PWMDnL | PWMDnL7 | PWMDnL6 | PWMDnL5 | PWMDnL4 | PWMDnL3 | PWMDnL2 | PWMDnL1 | PWMDnL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register PWMDnL (n=0-5) Address: F140H, F142H, F144H, F146H, F148H, F14AH.

Bit7~Bit0 PWMDnL<7:0>: PWM channel n compare data (duty cycle data) register lower 8 bits.

### 13.5.12 PWM Compare Data Register Higher 8 Bits PWMDnH (n=0-5)

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PWMDnH | PWMDnH7 | PWMDnH6 | PWMDnH5 | PWMDnH4 | PWMDnH3 | PWMDnH2 | PWMDnH1 | PWMDnH0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register PWMDnH (n=0-5) Address: F141H, F143H, F145H, F147H, F149H, F14BH.

Bit7~Bit0 PWMDnH<7:0>: PWM channel n compare data (duty cycle data) register higher 8 bits.

### 13.5.13 PWM Dead Time Enable Control Register (PWMDTE)

| F160H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PWMDTE | -- | -- | -- | -- | -- | PWM45DTE | PWM23DTE | PWM01DTE |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2 PWM45DTE: PWM4/5 channel dead time delay enable bit

1= Enable

0= Disable

Bit1 PWM23DTE: PWM2/3 channel dead time delay enable bit

1= Enable

0= Disable

Bit0 PWM01DTE: PWM0/1 channel dead time delay enable bit

1= Enabler

0= Disable

### 13.5.14 PWM0/1 Dead Time Configuration Register (PWM01DT)

| F161H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PWM01DT | PWM01 DT7 | PWM01 DT6 | PWM01 DT5 | PWM01 DT4 | PWM01 DT3 | PWM01 DT2 | PWM01 DT1 | PWM01 DT0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 PWM01DT<7:0>: PWM channel 0/1 dead time delay data register

### 13.5.15 PWM2/3 Dead Time Configuration Register (PWM23DT)

| F162H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PWM23DT | PWM23 DT7 | PWM23 DT6 | PWM23 DT5 | PWM23 DT4 | PWM23 DT3 | PWM23 DT2 | PWM23 DT1 | PWM23 DT0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 PWM23DT<7:0>: PWM channel 2/3 dead time delay data register

### 13.5.16 PWM4/5 Dead Time Configuration Register (PWM45DT)

| F163H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PWM45DT | PWM45 DT7 | PWM45 DT6 | PWM45 DT5 | PWM45 DT4 | PWM45 DT3 | PWM45 DT2 | PWM45 DT1 | PWM45 DT0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 PWM45DT<7:0>: PWM channel 4/5 dead time delay data register

## 13.6 PWM Interrupt

PWM has a total of 9 interrupt flags, which include 3 zero-crossing interrupt flags and 6 down-count comparison interrupt flags. The generation of interrupt flags is independent of whether the corresponding interrupt enable bit is set. To enable any type of PWM interrupt, the global interrupt enable bit (EA = 1) and the PWM global interrupt enable bit (PWMIE) must be set. Only then can the PWM interrupt function be successfully configured. All PWM interrupts share a single interrupt vector entry. Therefore, after entering the interrupt service routine, the user can determine which type of interrupt occurred by checking the interrupt flags.

The enable and priority of PWM interrupts can be configured via the following related register bits.

### 13.6.1 Interrupt Mask Register (EIE2)

| 0xAA | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| EIE2 | SPIIE | I2CIE | -- | ADCIE | PWMIE | ET5 | -- | -- |
| R/W | R/W | R/W | -- | R/W | R/W | R/W | R | R |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 SPIIE: SPI interrupt enable bit
1= Enable SPI interrupt
0= Disable SPI interrupt
Bit6 I2CIE: $I^2C$ interrupt enable bit
1= Enable $I^2C$ interrupt
0= Disable $I^2C$ interrupt
Bit5 --: Reserved, set to 0.
Bit4 ADCIE: ADC interrupt enable bit
1= Enable ADC interrupt
0= Disable ADC interrupt
Bit3 PWMIE: PWM global interrupt enable bit
1= Enable PWM all interrupts
0= Disable PWM all interrupts
Bit2 ET5: Timer5 interrupt enable bit
1= Enable Timer5 interrupt
0= Disable Timer5 interrupt
Bit1~Bit0 --: Reserved, set to 0.

### 13.6.2 Interrupt Priority Control Register (EIP2)

| 0xBA | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| EIP2 | PSPI | PI2C | -- | PADC | PPWM | PT5 | -- | -- |
| R/W | R/W | R/W | -- | R/W | R/W | R/W | R | R |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 PSPI: SPI interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit6 PI2C: $I^2C$ interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit5 -- Reserved, set to 0.
Bit4 PADC: ADC interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit3 PPWM: PWM interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit2 PT5: TIMER5 interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit1~Bit0 -- Reserved, set to 0.

### 13.6.3 PWM Zero-Crossing Interrupt Mask Register (PWMZIE)

| F169H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PWMZIE | -- | -- | -- | -- | -- | PWM2ZIE | PWM1ZIE | PWM0ZIE |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit3 -- Reserved, set to 0.
Bit2~Bit0 PWMnZIE: PWM channel n zero-crossing interrupt mask register ( (n=0-2)
(n=0, corresponds to PWM channels 0,1; n=1, corresponds to PWM channels 2,3; n=2, corresponds to PWM channels 4,5)
1= Enable interrupt
0= Disable interrupt

### 13.6.4 PWM Down Compare Interrupt Mask Register (PWMDIE)

| F16BH | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PWMDIE | -- | -- | PWM5DIE | PWM4DIE | PWM3DIE | PWM2DIE | PWM1DIE | PWM0DIE |
| R/W | R | R | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit6 -- Reserved, set to 0.

Bit5~Bit0 PWMnDIE: PWM channel n down compare interrupt mask register (n=0-5)

1= Enable interrupt

0= Disable interrupt

### 13.6.5 PWM Zero-Crossing Interrupt Flag Register (PWMZIF)

| F16DH | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PWMZIF | -- | -- | -- | -- | -- | PWM2ZIF | PWM1ZIF | PWM0ZIF |
| R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit3 -- Reserved, set to 0.

Bit2~Bit0 PWMnZIF: PWM channel n zero-crossing interrupt flag bit (n=0-2)
(n=0, corresponds to PWM channels 0,1; n=1, corresponds to PWM channels 2,3; n=2, corresponds to PWM channels 4,5)

1= An interrupt is generated (cleared by software)

0= No interrupt is generated

### 13.6.6 PWM Down Compare Interrupt Flag Register (PWMDIF)

| F16FH | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| PWMDIF | -- | -- | PWM5DIF | PWM4DIF | PWM3DIF | PWM2DIF | PWM1DIF | PWM0DIF |
| R/W | R | R | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit6 -- Reserved, set to 0.

Bit5~Bit0 PWMnDIF: PWM channel n down compare interrupt flag bit (n=0-5)

1= An interrupt is generated (cleared by software)

0= No interrupt is generated

# 14. Hardware LED Matrix Driver

## 14.1 Overview

This chip integrates a hardware LED matrix driver circuit, enabling users to easily implement LED display driving.

## 14.2 Features

The hardware LED matrix driver has the following features:

- Duty cycle options: 1/4, 1/5, 1/6, and 1/8.
- Clock source: fixed clock source is $F_{FIX}$ (8MHz).
- Supports up to 8 COM pins and 26 SEG pins.
- COM pins support both common cathode and common anode drive modes.
- The active time of COM pins can be set via an 8-bit register.
- COM pin current: 55 mA and 220 mA.
- 16 current levels are available for SEG pins, with a maximum current of 50 mA.
- The clock source period for LED counting is selectable.
- Dead time can be configured.
- Dimming mode is supported, with selectable dimming time.

## 14.3 Relevant Registers

### 14.3.1 LED Drive Mode Selection Register (LEDMODE)

| F769H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDMODE | LEDMODE7 | LEDMODE6 | LEDMODE5 | LEDMODE4 | LEDMODE3 | LEDMODE2 | LEDMODE1 | LEDMODE0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7-0 LEDMODE LED driver mode selection register

0x55= LED matrix driver mode active: When the LED matrix driver mode is active, the related registers take effect.

0xaa= LED dot matrix driver mode active: When the LED dot matrix driver mode is active, the relevant registers become effective.

Other= Inactive

### 14.3.2 LED Control Register (LEDCON)

| F765H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDCON | LED_EN | DUTY1 | DUTY0 | CC_CA | -- | LED_FIXCLK_EN | -- | -- |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | W | W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 LED_EN: LED enable control bit

1= LED is enabled (when LED_FIXCLK_EN=1)

0= LED is disabled

Bit6~Bit5 DUTY<1:0>: LED duty cycle selection bit

11= 1/4 DUTY

10= 1/5 DUTY

01= 1/6 DUTY

00= 1/8 DUTY

Bit4 CC_CA: LED driver mode selection bBit

1= Anode drive mode

0= Cathode drive mde

Bit3 - Reserved

Bit2 LED_FIXCLK_EN: LED operating clock enable control

1= Enable

0= Disable

Bit1~Bit0 - Reserved

COM Selection Table

| DUTY | ICOM0 | ICOM1 | ICOM2 | ICOM3 | ICOM4 | ICOM5 | ICOM6 | ICOM7 | Active SEG port |
|---|---|---|---|---|---|---|---|---|---|
| 11 | LED_C0 | LED_C1 | LED_C2 | LED_C3 | - | - | - | - | LED_S0-LED_S25 |
| 10 | LED_C0 | LED_C1 | LED_C2 | LED_C3 | LED_C4 | - | - | - | LED_S1-LED_S25 |
| 01 | LED_C0 | LED_C1 | LED_C2 | LED_C3 | LED_C4 | LED_C5 | - | - | LED_S2-LED_S25 |
| 00 | LED_C0 | LED_C1 | LED_C2 | LED_C3 | LED_C4 | LED_C5 | LED_C6 | LED_C7 | LED_S4-LED_S25 |

Note: In the table above, ICOM0-ICOM7 refer to the internal COM driving output signals of the LED.

LED_C0-LED_C7 and LED_S0-LED_S25 represent the final pin names where the internal driving signals of the LED are mapped.

### 14.3.3 LED Clock Selection Register (LEDCKS)

| F766H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDCKS | - | - | DTSEL1 | DTSEL0 | - | - | CLK1 | CLK0 |
| R/W | W | W | R/W | R/W | W | W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit6 -- Reserved

Bit5~Bit4 DTSEL<1:0>: Matrix LED dead time selection

00: 8*Tfix

01: 16*Tfix

10: 32*Tfix

11: 64*Tfix

Bit3~Bit2 --: Reserved

Bit1~Bit0 CLK<1:0>: LED counter clock (TLED_CLK)

00: 32*Tfix

10: 64*Tfix

10: 128*Tfix

11: 256*Tfix

### 14.3.4 COM Port Valid Time Selection Register (LEDCOMTIME)

| F768H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDCOMTIME | COMT7 | COMT6 | COMT5 | COMT4 | COMT3 | COMT2 | COMT1 | COMT0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 COMT<7:0>: COM port valid time setting

Note: Setting to 0x00 is prohibited.

COM time = (COMT<7:0> + 1) * $T_{LED\_CLK}$.

### 14.3.5 Dimming Time Selection Register (LEDSEGTIME)

| F76DH | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDSEGTIME | SEGT7 | SEGT6 | SEGT5 | SEGT4 | SEGT3 | SEGT2 | SEGT1 | SEGT0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 SEGT<7:0>: SEG port valid time setting.

Note: Setting to 0x00 is prohibited.

Dimming active time = (SEGT<7:0> + 1) * TLED_CLK .

Note: The effective display duty cycle for scanning a COM is: (SEGT<7:0> + 1) / (COMT<7:0> + 1) ; SEGT[7:0]≤COMT[7:0].

### 14.3.6 COM Port Enable Control Register (LEDCOMEN)

| F760H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDCOMEN | COMEN7 | COMEN6 | COMEN5 | COMEN4 | COMEN3 | COMEN2 | COMEN1 | COMEN0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 COMEN<7:0>: LED_C7-LED_C0 port enable control bit
1= Enable
0= Disable

### 14.3.7 SEG Port Enable Control Register (LEDSEGEN0)

| F761H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDSEGEN0 | SEGEN7 | SEGEN6 | SEGEN5 | SEGEN4 | SEGEN3 | SEGEN2 | SEGEN1 | SEGEN0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 SEGEN<7:0>: LED_S7-LED_S0 port enable control bit
1= Enable
0= Disable

### 14.3.8 SEG Port Enable Control Register (LEDSEGEN1)

| F762H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDSEGEN1 | SEGEN15 | SEGEN14 | SEGEN13 | SEGEN12 | SEGEN11 | SEGEN10 | SEGEN9 | SEGEN8 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 SEGEN<15:8>: LED_S15-LED_S8 port enable control bit
1= Enable
0= Disable

### 14.3.9 SEG Port Enable Control Register (LEDSEGEN2)

| F763H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDSEGEN2 | SEGEN23 | SEGEN22 | SEGEN21 | SEGEN20 | SEGEN19 | SEGEN18 | SEGEN17 | SEGEN16 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 SEGEN<23:16>: LED_S23-LED_S16 port enable control bit
1= Enable
0= Disable

### 14.3.10 SEG Port Enable Control Register (LEDSEGEN3)

| F764H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDSEGEN3 | - | - | - | - | - | - | SEGEN25 | SEGEN24 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit2 - Reserved, set to 0.

Bit1~Bit0 SEGEN<25:24>: LED_S25-LED_S24 port enable control bit

1= Enable

0= Disable

### 14.3.11 COM0 SEG Data Register (LEDC0DATAn (n=0-3))

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDC0DATAn | SEG[8n+7] | SEG[8n+6] | SEG[8n+5] | SEG[8n+4] | SEG[8n+3] | SEG[8n+2] | SEG[8n+1] | SEG[8n] |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

LEDC0DATA0 Address: F740H; LEDC0DATA1 Address: F741H; LEDC0DATA2 Address: F742H; LEDC0DATA3 Address: F743H.

(When n=3, the bit7-bit2 values are invalidly written).

Bit7~Bit0 SEG<8n+7:8n>: When COM0 port is valid, SEG[8n+7]-SEG[8n] port data is output.

1= High level

0= Low level

### 14.3.12 COM1 SEG Data Register (LEDC1DATAn (n=0-3))

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDC1DATAn | SEG[8n+7] | SEG[8n+6] | SEG[8n+5] | SEG[8n+4] | SEG[8n+3] | SEG[8n+2] | SEG[8n+1] | SEG[8n] |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

LEDC1DATA0 Address: F744H; LEDC1DATA1 Address: F745H; LEDC1DATA2 Address: F746H; LEDC1DATA3 Address: F747H.

(When n=3, the bit7-bit2 values are invalidly written).

Bit7~Bit0 SEG<8n+7:8n>: When COM1 port is valid, SEG[8n+7]-SEG[8n] port data is output.

1= High level

0= Low level

### 14.3.13 COM2 SEG Data Register (LEDC2DATAn (n=0-3))

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDC2DATAn | SEG[8n+7] | SEG[8n+6] | SEG[8n+5] | SEG[8n+4] | SEG[8n+3] | SEG[8n+2] | SEG[8n+1] | SEG[8n] |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

LEDC2DATA0 Address: F748H; LEDC2DATA1 Address: F749H; LEDC2DATA2 Address: F74AH; LEDC2DATA3 Address: F74BH.

(When n=3, the bit7-bit2 values are invalidly written).

Bit7~Bit0 SEG<8n+7:8n>: When COM2 port is valid, SEG[8n+7]-SEG[8n] port data is output.

1= High level

0= Low level

### 14.3.14 COM3 SEG Data Register ((LEDC3DATAn (n=0-3))

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDC3DATAn | SEG[8n+7] | SEG[8n+6] | SEG[8n+5] | SEG[8n+4] | SEG[8n+3] | SEG[8n+2] | SEG[8n+1] | SEG[8n] |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

LEDC3DATA0 Address: F74CH; LEDC3DATA1 Address: F74DH; LEDC3DATA2 Address: F74EH; LEDC3DATA3 Address: F74FH;

(When n=3, the bit7-bit2 values are invalidly written).

Bit7~Bit0 SEG<8n+7:8n>: When COM3 port is valid, SEG[8n+7]-SEG[8n] port data is output.

1= High level

0= Low level

### 14.3.15 COM4 SEG Data Register (LEDC4DATAn (n=0-3))

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDC4DATAn | SEG[8n+7] | SEG[8n+6] | SEG[8n+5] | SEG[8n+4] | SEG[8n+3] | SEG[8n+2] | SEG[8n+1] | SEG[8n] |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

LEDC4DATA0 Address: F750H; LEDC4DATA1 Address: F751H; LEDC4DATA2 Address: F752H; LEDC4DATA3 Address: F753H.

(When n=3, the bit7-bit2 values are invalidly written).

Bit7~Bit0 SEG<8n+7:8n>: When COM4 port is valid, SEG[8n+7]-SEG[8n] port data is output.

1= High level

0= Low level

### 14.3.16 COM5 SEG Data Register (LEDC5DATAn (n=0-3))

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDC5DATAn | SEG[8n+7] | SEG[8n+6] | SEG[8n+5] | SEG[8n+4] | SEG[8n+3] | SEG[8n+2] | SEG[8n+1] | SEG[8n] |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

LEDC5DATA0 Address: F754H; LEDC5DATA1 Address: F755H; LEDC5DATA2 Address: F756H; LEDC5DATA3 Address: F757H.

(When n=3, the bit7-bit2 values are invalidly written).

Bit7~Bit0 SEG<8n+7:8n>: When COM5 port is valid, SEG[8n+7]-SEG[8n] port data is output.

1= High level

0= Low level

### 14.3.17 COM6 SEG Data Register (LEDC6DATAn (n=0-3))

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDC6DATAn | SEG[8n+7] | SEG[8n+6] | SEG[8n+5] | SEG[8n+4] | SEG[8n+3] | SEG[8n+2] | SEG[8n+1] | SEG[8n] |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

LEDC6DATA0 Address: F758H; LEDC6DATA1 Address: F759H; LEDC6DATA2 Address: F75AH; LEDC6DATA3 Address: F75BH;

(When n=3, the bit7-bit2 values are invalidly written).

Bit7~Bit0 SEG<8n+7:8n>: When COM6 port is valid, SEG[8n+7]-SEG[8n] port data is output.

1= High level

0= Low level

### 14.3.18 COM7 SEG Data Register (LEDC7DATAn (n=0-3))

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDC7DATAn | SEG[8n+7] | SEG[8n+6] | SEG[8n+5] | SEG[8n+4] | SEG[8n+3] | SEG[8n+2] | SEG[8n+1] | SEG[8n] |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

LEDC7DATA0 Address: F75CH; LEDC7DATA1 Address: F75DH; LEDC7DATA0 Address: F75EH; LEDC7DATA3 Address: F75FH.

(When n=3, the bit7-bit2 values are invalidly written).

Bit7~Bit0 SEG<8n+7:8n>: When COM7 port is valid, SEG[8n+7]-SEG[8n] port data is output.

1= High level

0= Low level

### 14.3.19 LED Matrix Drive Port Current Control Register (LEDDRV)

| F76BH | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDDRV | -- | -- | -- | -- | SDRC3 | SDRC2 | SDRC1 | SDRC0 |
| R/W | W | W | W | W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit4 -- Reserved, set to 0.

Bit3~Bit0 SDRC<3:0>: All SEG pin source current drive selection control bi

The control bits take effect only when the corresponding pin drive enable is active; otherwise, the current defaults to the maximum value.

| | | | |
|---|---|---|---|
| 0000= | 0mA | 1000= | 26.9mA |
| 0001= | 4.9mA | 1001= | 31.8mA |
| 0010= | 7.4mA | 1010= | 34.3mA |
| 0011= | 12.2mA | 1011= | 39.2mA |
| 0100= | 14.7mA | 1100= | 41.6mA |
| 0101= | 19.5mA | 1101= | 46.1m |
| 0110= | 22.1mA | 1110= | 49mA |
| 0111= | 26.9mA | 1111= | 50mA |

### 14.3.20 LED COM Port Drive Current Selection Register (LEDCOMDRV)

| F76EH | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDCOMDRV | CDRC7 | CDRC6 | CDRC5 | CDRC4 | CDRC3 | CDRC2 | CDRC1 | CDRC0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit[7:0] CDRC<7:0>: IO port drive current selection for COM7-COM0

(This is determined solely by the value of the corresponding register bit, and is independent of the LED module enable and COM port enable. The current defaults to a lower value.)

1= 220mA

0= 55mA

### 14.3.21 Pin Drive Enable Register (LEDSEGDR0)

| F76FH | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDSEGDR0 | LEDSEGDR0_7 | LEDSEGDR0_6 | LEDSEGDR0_5 | LEDSEGDR0_4 | LEDSEGDR0_3 | LEDSEGDR0_2 | LEDSEGDR0_1 | LEDSEGDR0_0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 LEDSEGDR0_7:SEG7 pin drive enable bit

1= SEG7 (P13) pin drive enable; SEG7 (P13) pin source current drive is configured by LEDDRV[3:0].

0= SEG7 (P13) pin drive disable, SEG7 (P13) pin source current is driven to the default value.

Bit6 LEDSEGDR0 _6:SEG6 pin drive enable bit

1= SEG6 (P12) pin drive enable; SEG6 (P12) pin source current drive is configured by LEDDRV[3:0].

0= SEG6 (P12) pin drive disable, SEG6 (P12) pin source current is driven to the default value.

Bit5 LEDSEGDR0 _5:SEG5 pin drive enable bit

1= SEG5 (P11) pin drive enable; SEG5 (P11) pin source current drive is configured by LEDDRV[3:0].

0= SEG5 (P11) pin drive disable, SEG5 (P11) pin source current is driven to the default value.

Bit4 LEDSEGDR0 _4:SEG4 pin drive enable bit

1= SEG4 (P10) pin drive enable; SEG4 (P10) pin source current drive is configured by LEDDRV[3:0].

0= SEG4 (P10) pin drive disable, SEG4 (P10) pin source current is driven to the default value.

Bit3 LEDSEGDR0 _3:SEG3 pin drive enable bit

1= SEG3 (P07) pin drive enable; SEG3 (P07) pin source current drive is configured by LEDDRV[3:0].

0= SEG3 (P07) pin drive disable, SEG3 (P07) pin source current is driven to the default value.

Bit2 LEDSEGDR0_2:SEG2 pin drive enable bit

1= SEG2 (P06) pin drive enable; SEG2 (P06) pin source current drive is configured by LEDDRV[3:0].

0= SEG2 (P06) pin drive disable, SEG2 (P06) pin source current is driven to the default value.

Bit1 LEDSEGDR0 _1:SEG1 pin drive enable bit

1= SEG1 (P05) pin drive enable; SEG1 (P05) pin source current drive is configured by LEDDRV[3:0].

0= SEG1 (P05) pin drive disable, SEG1 (P05) pin source current is driven to the default value.

Bit0 LEDSEGDR0 _0:SEG0 pin drive enable bit

1= SEG0 (P04) pin drive enable; SEG0 (P04) pin source current drive is configured by LEDDRV[3:0].

0= SEG0 (P04) pin drive disable, SEG0 (P04) pin source current is driven to the default value.

### 14.3.22 Pin Drive Enable Register (LEDSEGDR1)

| F770H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDSEGDR1 | LEDSEGDR1 _7 | LEDSEGDR1 _6 | LEDSEGDR1 _5 | LEDSEGDR1 _4 | LEDSEGDR1_3 | LEDSEGDR1_2 | LEDSEGDR1 _1 | LEDSEGDR1 _0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 LEDSEGDR1 _7:SEG15 pin drive enable bit

1= SEG15 (P23) pin drive enable; SEG15 (P23) pin source current drive is configured by LEDDRV[3:0].

0= SEG15 (P23) pin drive disable, SEG15 (P23) pin source current is driven to the default value.

Bit6 LEDSEGDR1 _6:SEG14 pin drive enable bit

1= SEG14 (P22) pin drive enable; SEG14 (P22) pin source current drive is configured by LEDDRV[3:0].

0= SEG14 (P22) pin drive disable, SEG14 (P22) pin source current is driven to the default value.

Bit5 LEDSEGDR1 _5:SEG13 pin drive enable bit

1= SEG13 (P21) pin drive enable; SEG13 (P21) pin source current drive is configured by LEDDRV[3:0].

0= SEG13 (P21) pin drive disable, SEG13 (P21) pin source current is driven to the default value.

Bit4 LEDSEGDR1 _4:SEG12 pin drive enable bit

1= SEG12 (P20) pin drive enable; SEG12 (P20) pin source current drive is configured by LEDDRV[3:0].

0= SEG12 (P20) pin drive disable, SEG12 (P20) pin source current is driven to the default value.

Bit3 LEDSEGDR1 _3:SEG11 pin drive enable bit

1= SEG11 (P17) pin drive enable; SEG11 (P17) pin source current drive is configured by LEDDRV[3:0].

0= SEG11 (P17) pin drive disable, SEG11 (P17) pin source current is driven to the default value.

Bit2 LEDSEGDR1_2:SEG10 pin drive enable bit

1= SEG10 (P16) pin drive enable; SEG10 (P16) pin source current drive is configured by LEDDRV[3:0].

0= SEG10 (P16) pin drive disable, SEG10 (P16) pin source current is driven to the default value.

Bit1 LEDSEGDR1 _1:SEG9 pin drive enable bit

1= SEG9 (P15) pin drive enable; SEG9 (P15) pin source current drive is configured by LEDDRV[3:0].

0= SEG9 (P15) pin drive disable, SEG9 (P15) pin source current is driven to the default value.

Bit0 LEDSEGDR1 _0:SEG8 pin drive enable bit

1= SEG8 (P14) pin drive enable; SEG8 (P14) pin source current drive is configured by LEDDRV[3:0].

0= SEG8 (P14) pin drive disable, SEG8 (P14) pin source current is driven to the default value.

### 14.3.23 Pin Drive Enable Register (LEDSEGDR2)

| F771H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDSEGDR2 | LEDSEGDR2 _7 | LEDSEGDR2 _6 | LEDSEGDR2 _5 | LEDSEGDR2 _4 | LEDSEGDR2_3 | LEDSEGDR2_2 | LEDSEGDR2 _1 | LEDSEGDR2 _0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 LEDSEGDR2 _7:SEG23 pin drive enable bit

1= SEG23 (P53) pin drive enable; SEG23 (P53) pin source current drive is configured by LEDDRV[3:0].

0= SEG23 (P53) pin drive disable, SEG23 (P53) pin source current is driven to the default value.

Bit6 LEDSEGDR2 _6:SEG22 pin drive enable bit

1= SEG22 (P52) pin drive enable; SEG22 (P52) pin source current drive is configured by LEDDRV[3:0].

0= SEG22 (P52) pin drive disable, SEG22 (P52) pin source current is driven to the default value.

Bit5 LEDSEGDR2 _5:SEG21 pin drive enable bit

1= SEG21 (P51) pin drive enable; SEG21 (P51) pin source current drive is configured by LEDDRV[3:0].

0= SEG21 (P51) pin drive disable, SEG21 (P51) pin source current is driven to the default value.

Bit4 LEDSEGDR2 _4:SEG20 pin drive enable bit

1= SEG20 (P50) pin drive enable; SEG20 (P50) pin source current drive is configured by LEDDRV[3:0].

0= SEG20 (P50) pin drive disable, SEG20 (P50) pin source current is driven to the default value.

Bit3 LEDSEGDR2 _3:SEG19 pin drive enable bit

1= SEG19 (P27) pin drive enable; SEG19 (P27) pin source current drive is configured by LEDDRV[3:0].

0= SEG19 (P27) pin drive disable, SEG19 (P27) pin source current is driven to the default value.

Bit2 LEDSEGDR2_2:SEG18 pin drive enable bit

1= SEG18 (P26) pin drive enable; SEG18 (P26) pin source current drive is configured by LEDDRV[3:0].

0= SEG18 (P26) pin drive disable, SEG18 (P26) pin source current is driven to the default value.

Bit1 LEDSEGDR2 _1:SEG17 pin drive enable bit

1= SEG17 (P25) pin drive enable; SEG17 (P25) pin source current drive is configured by LEDDRV[3:0].

0= SEG17 (P25) pin drive disable, SEG17 (P25) pin source current is driven to the default value.

Bit0 LEDSEGDR2 _0:SEG16 pin drive enable bit

1= SEG16 (P24) pin drive enable; SEG16 (P24) pin source current drive is configured by LEDDRV[3:0].

0= SEG16 (P24) pin drive disable, SEG16 (P24) pin source current is driven to the default value.

### 14.3.24 Pin Drive Enable Register (LEDSEGDR3)

| F772H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDSEGDR3 | -- | LEDSEGDR3 _6 | LEDSEGDR3 _5 | LEDSEGDR3 _4 | LEDSEGDR3_3 | LEDSEGDR3_2 | LEDSEGDR3 _1 | LEDSEGDR3 _0 |
| R/W | R | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 -- Reserved, set to 0.

Bit6 LEDSEGDR3 _6: P03 pin drive enable bit

1= P03 pin drive enable; P03 pin source current drive is configured by LEDDRV[3:0].

0= P03 pin drive disable, P03 pin source current is driven to the default value.

Bit5 LEDSEGDR3 _5: P02 pin drive enable bit

1= P02 pin drive enable; P02 pin source current drive is configured by LEDDRV[3:0].

0= P02 pin drive disable, P02 pin source current is driven to the default value.

Bit4 LEDSEGDR3 _4: P01 pin drive enable bit

1= P01 pin drive enable; P01 pin source current drive is configured by LEDDRV[3:0].

0= P01 pin drive disable, P01 pin source current is driven to the default value.

Bit3 LEDSEGDR3 _3: P00 pin drive enable bit

1= P00 pin drive enable; P00 pin source current drive is configured by LEDDRV[3:0].

0= P00 pin drive disable, P00 pin source current is driven to the default value.

Bit2 Reserved, set to 0.

Bit1 LEDSEGDR3 _1: SEG25 pin drive enable bit

1= SEG25 (P55) pin drive enable; SEG25 (P55) pin source current drive is configured by LEDDRV[3:0].

0= SEG25 (P55) pin drive disable, SEG25 (P55) pin source current is driven to the default value.

Bit0 LEDSEGDR3 _0: SEG24 pin drive enable bit

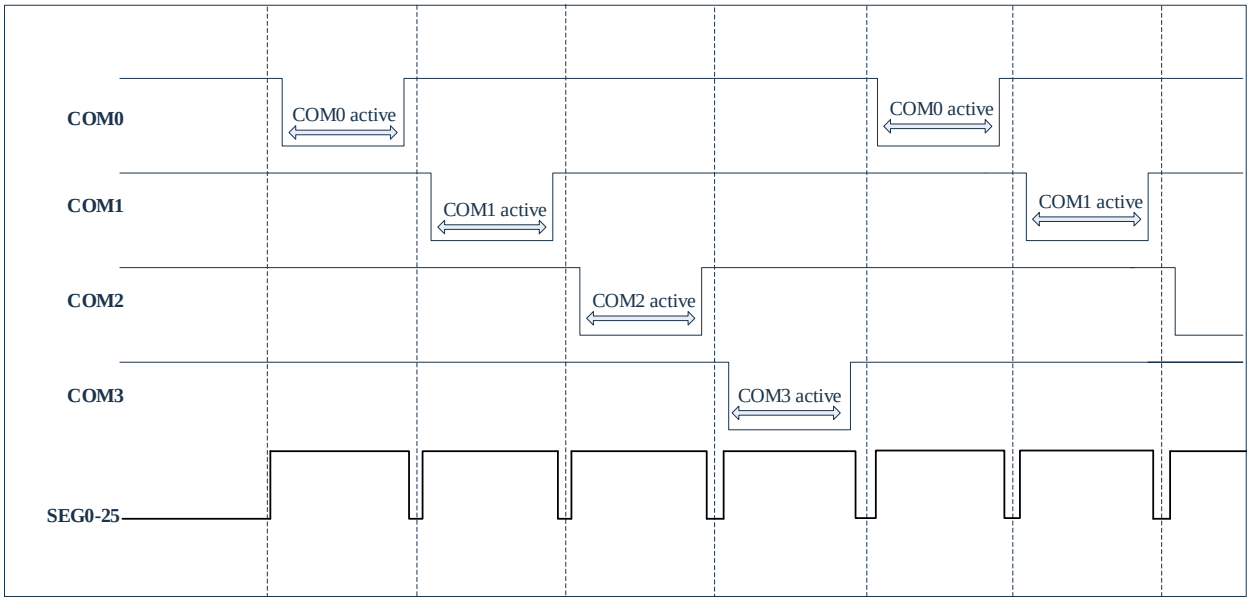
1= SEG24 (P54) pin drive enable; SEG24 (P54) pin source current drive is configured by LEDDRV[3:0].

0= SEG24 (P54) pin drive disable, SEG24 (P54) pin source current is driven to the default value.

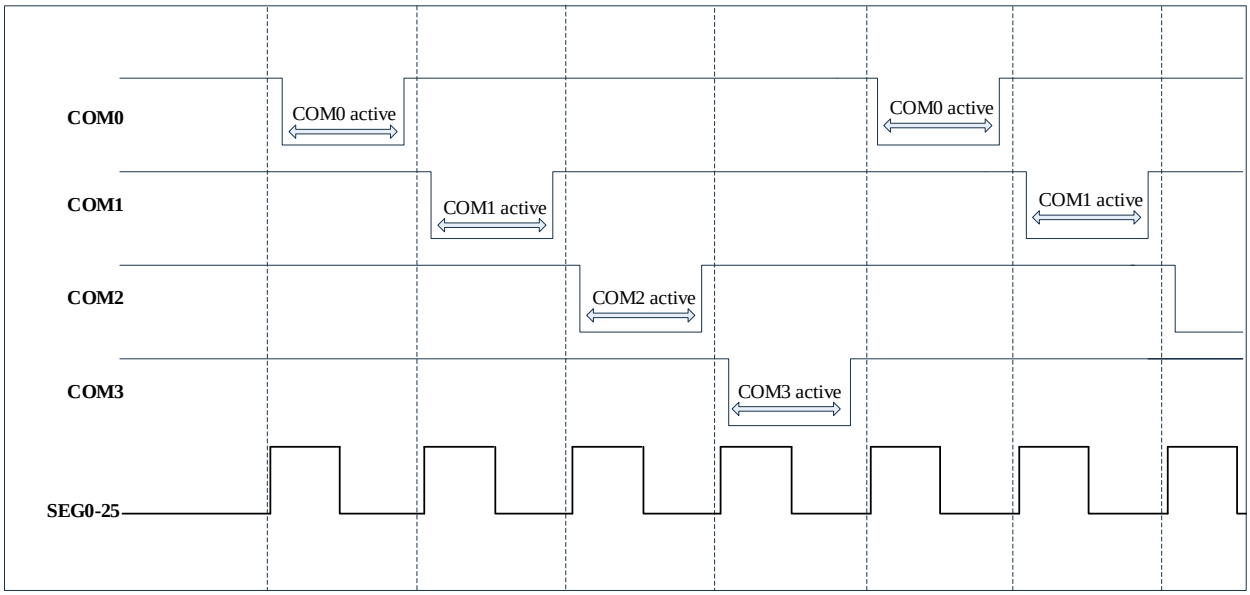
## 14.4 LED Driver Output Waveforms

According to the configuration registers related to the LED driver, the corresponding LED driver output waveform can be set.

For a 1/4 DUTY configuration in common cathode driving mode, with the segment data all set to 1, and the segment output having a duty cycle of 1, the waveform is as shown in the diagram below:




For a 1/4 DUTY configuration in common cathode driving mode, with the segment data all set to 1, and the segment output having a duty cycle of 1/2, the waveform is as shown in the diagram below:



# 15. Hardware LED Dot Matrix Driver

## 15.1 Overview

The LED dot matrix driver is used to drive multiple LED lights by configuring the LED0~LED8 pins, making it convenient for users to control LED dot matrix displays.

## 15.2 Features

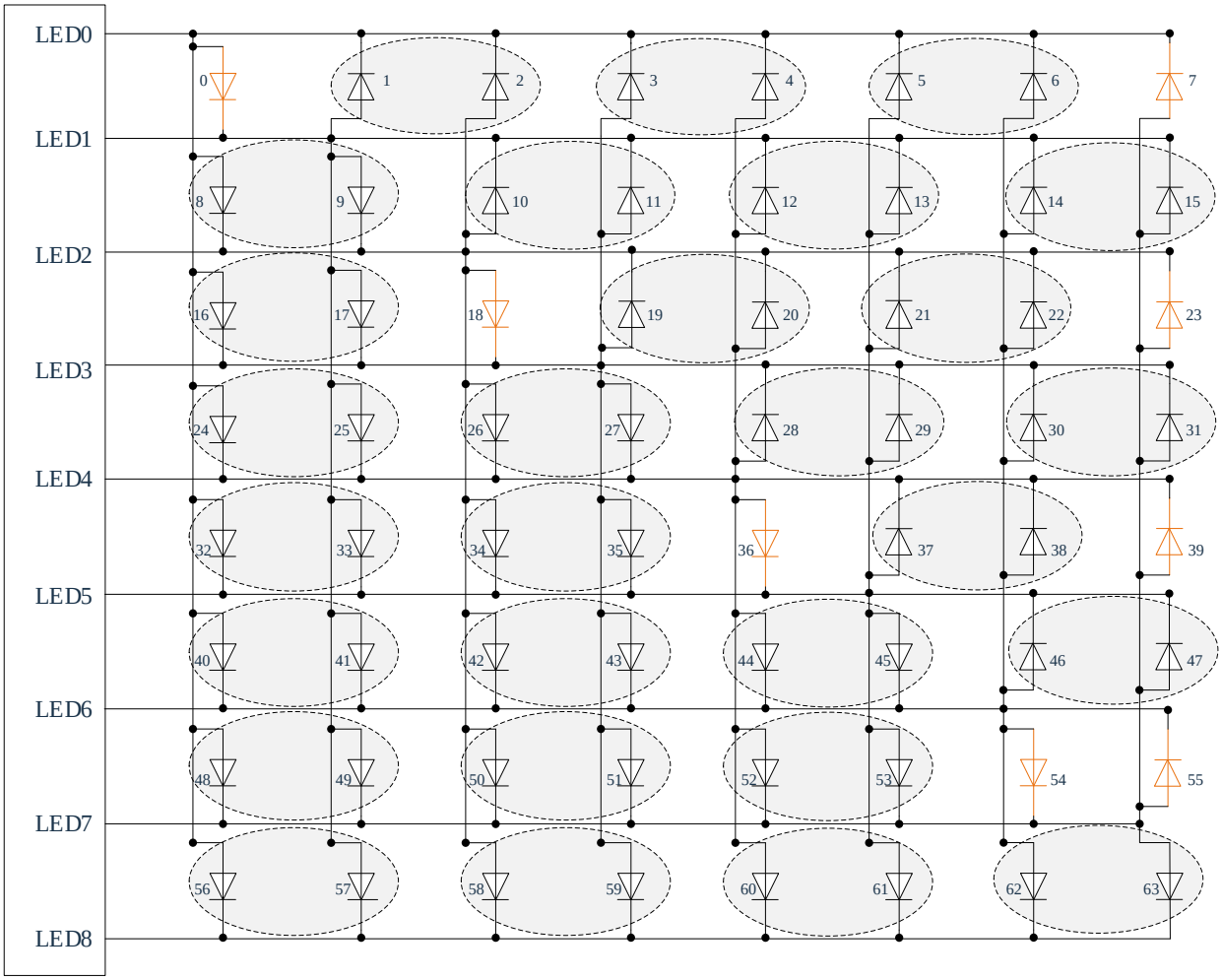
The LED dot matrix driving mode has the following features:

- Supports up to 64 LED lights, with configurable dot matrix sizes including 4x4, 5x5, 6x6, 6x7, 7x7, 7x8, and 8x8.
- Each LED supports two selectable conduction times, with each conduction time being set by an 8-bit timer.
- Each LED's display data can be selected independently.
- The LED dot matrix driver supports cyclic scan mode and interrupt scan mode.
- The clock source is fixed as $F_{FIX}$ (8 MHz).
- The LED count clock source period is selectable.
- The LED dot matrix driver mode allows for LED0-LED8 pin enable selection.
- The LED dot matrix pin current can be selected from 16 levels, with a maximum current of 50mA.
- The starting position for LED0-LED7 can be selected, with the other pins cycling in order.
- The 64-LED dot matrix address is unique, as described below, and is used to input the on/off information for the LEDs.

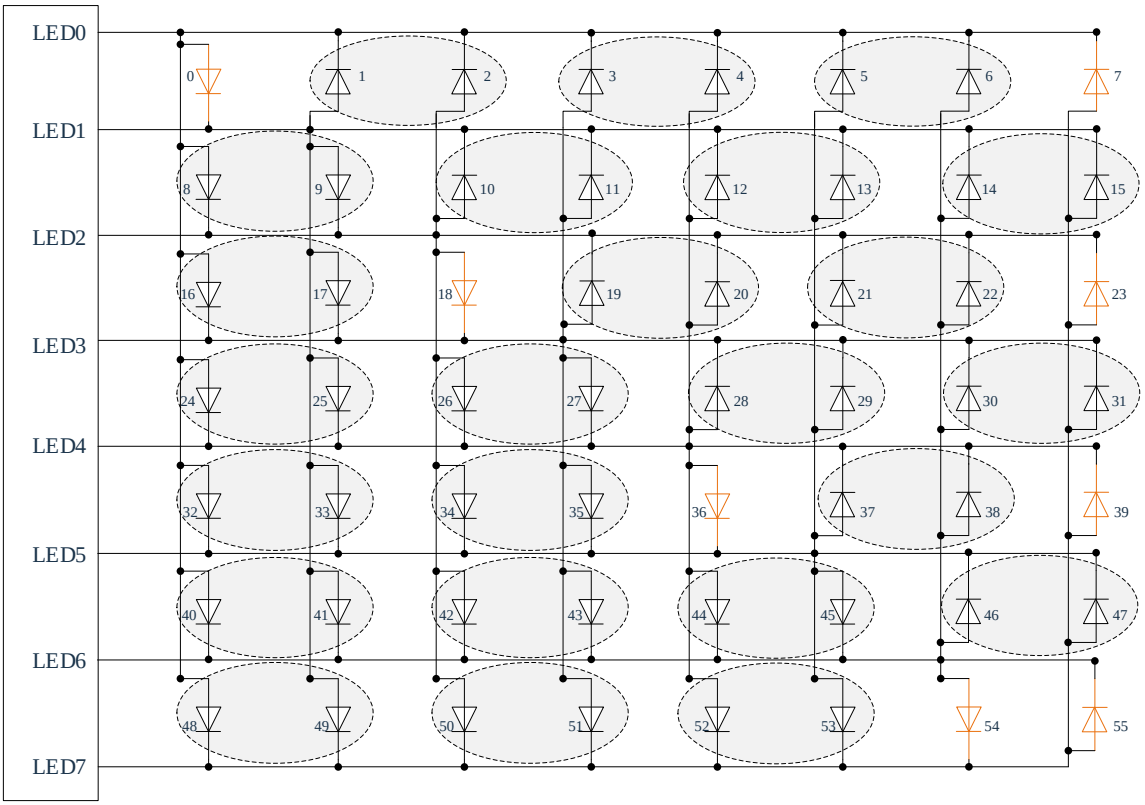
## 15.3 Function Description

The LED matrix operates by scanning in an 8*8 dot matrix dual-light mode, meaning two lights are lit up at a time (common cathode). It corresponds to the LED0~LED8 pins and can drive a maximum of 8x8=64 LEDs. The configuration of the LEDs corresponds to the address of the lights' on/off status (1 means the light is on, 0 means the light is off). The hardware will interpret the on/off address and the current scanning address, and automatically control the output of the corresponding I/O pins. Different matrix sizes, including 4x4, 5x5, 6x6, 6x7, 7x7, 7x8, and 8x8, can be configured, with the corresponding LED address remaining unchanged.

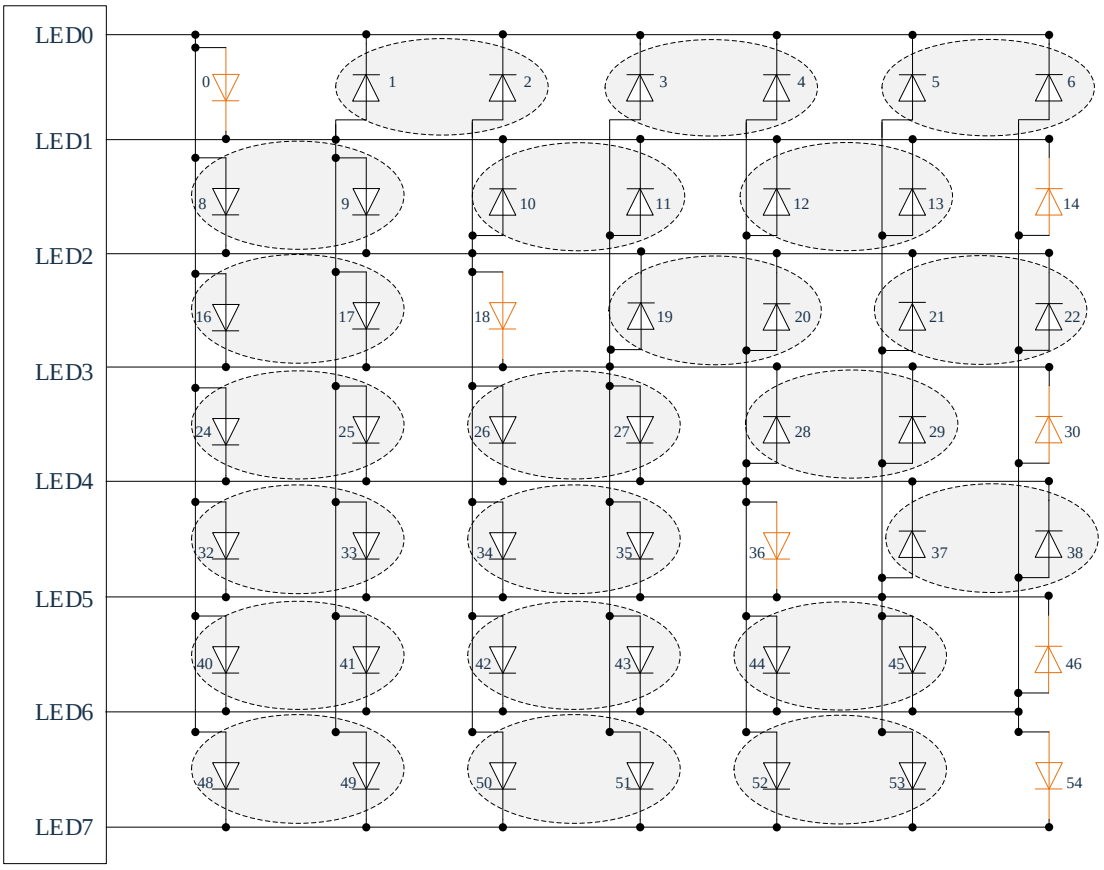
The 8*8 matrix is shown below (the yellow lights in the diagram represent the individually scanned lights).



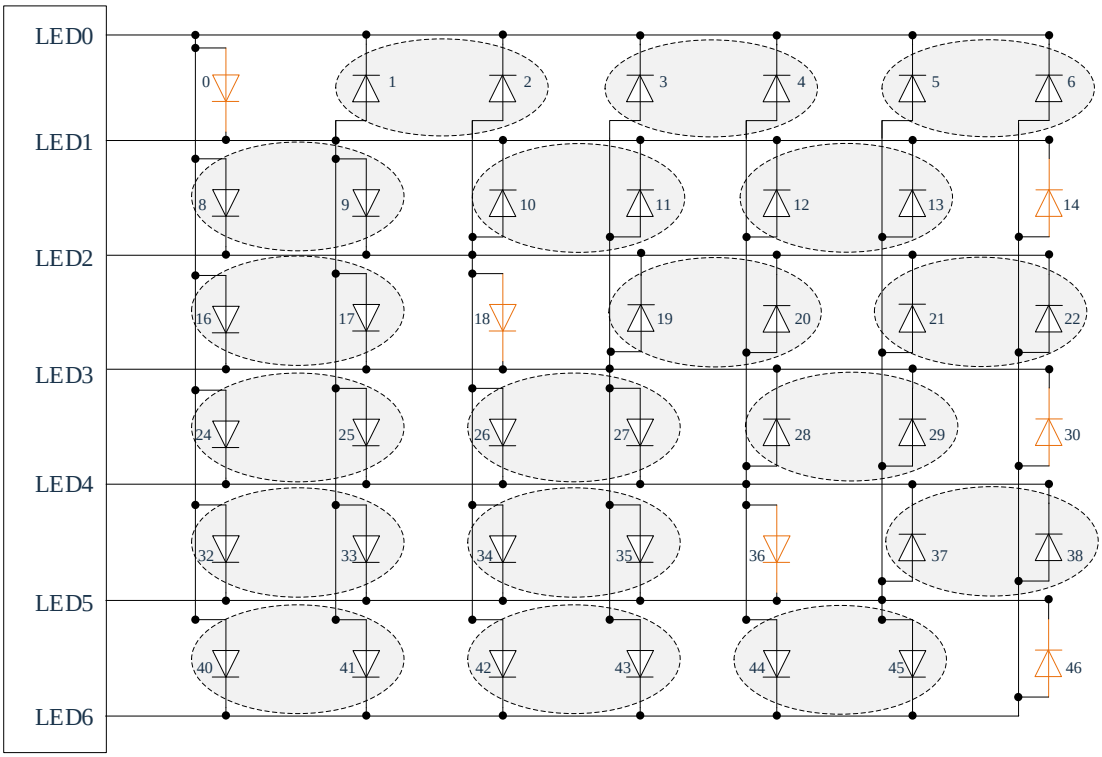
The 7*8 dot matrix is shown in the diagram below:



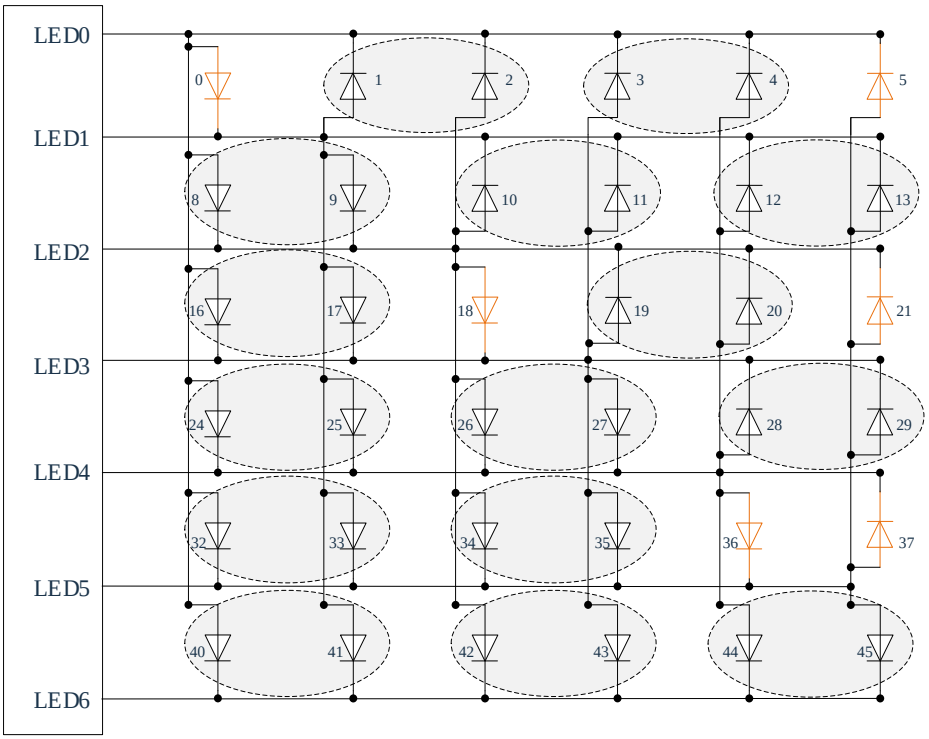

The 7*7 dot matrix is shown in the diagram below:



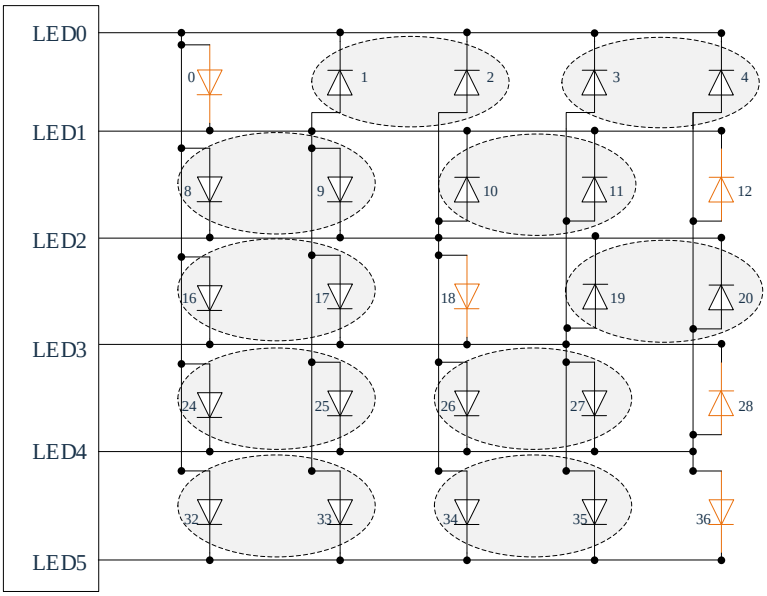
The 6*7 dot matrix is shown in the diagram below:



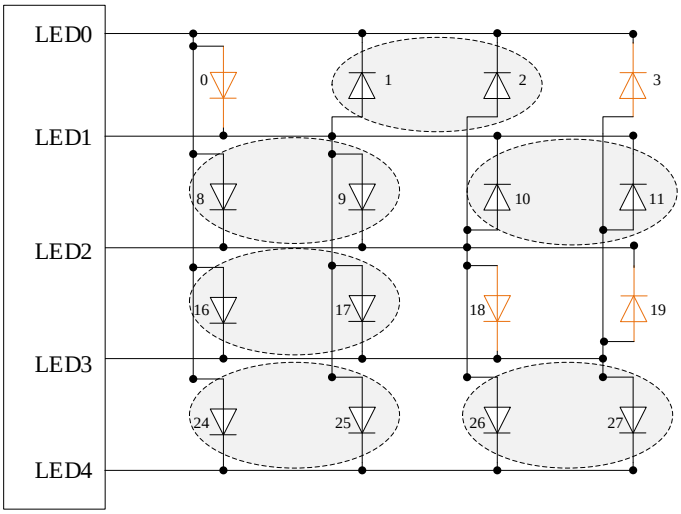

The 6*6 dot matrix is shown in the diagram below:



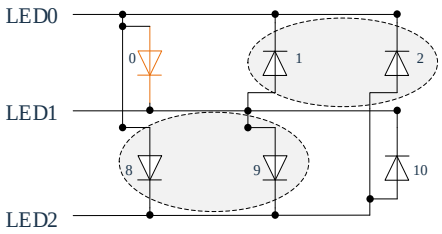
The 5*5 dot matrix is shown in the diagram below:



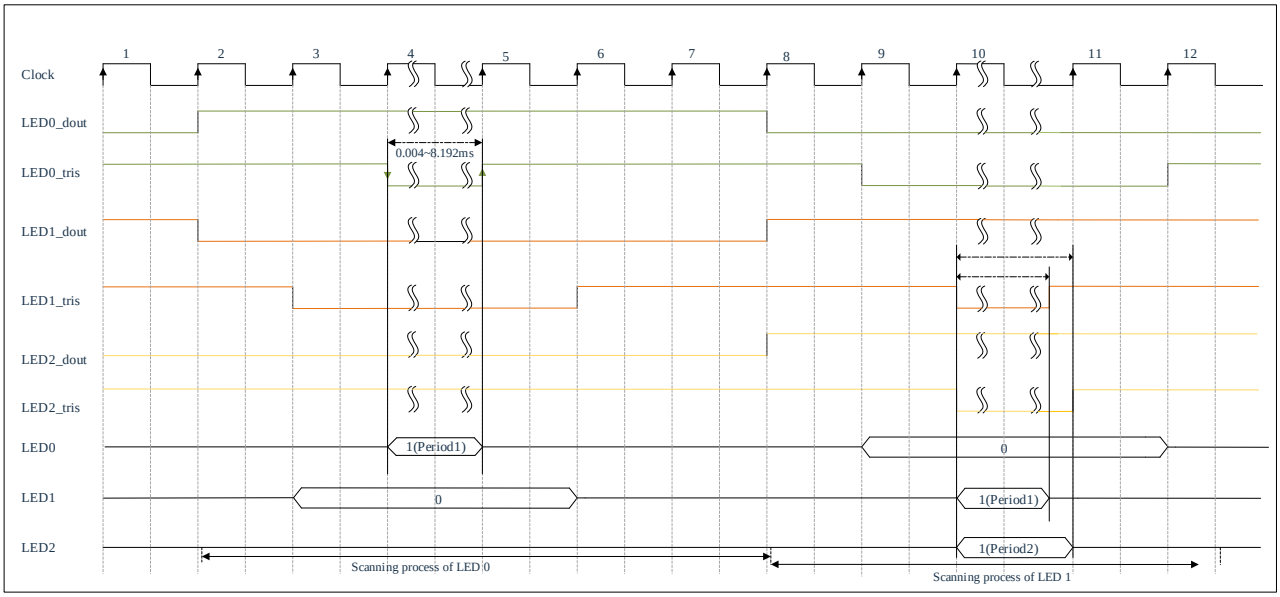

The 4*4 dot matrix is shown in the diagram below:



For example, to light up LEDs 0, 1, and 2, LEDs 0 and 1 are selected in the first segment of the cycle, while LED 2 is selected in the second segment. Additionally, SCAN1W[7:0] < SCAN2W[7:0]. The detailed digital output interface control timing is shown in the diagram below:




Schematic diagram of two lights




Digital output timing control diagram

### 15.3.1 Dot Matrix LED Period Selection

Each LED in the dot matrix has two periods that can be selected: Period1 and Period2:

- Period1 = ({SCAN1W[7:0]}+1) * $T_{LEDCLK}$, SCAN1W is the configuration register for the first period.
- Period2 = ({SCAN2W[7:0]}+1) * $T_{LEDCLK}$, SCAN2W is the configuration register for the second period.
- Period selection register is LEDnSEL (where n = 0 to 7). The 64 LEDs corresponding to the 8*8 dot matrix (LED0 to LED63) are listed in the table below, where LEDs with a value of 1 will select Period2, and those with a value of 0 will select Period1.

Note: Period1/Period2 represents the display duration of the corresponding LEDs (e.g., the time for LEDs 0/1/2 to output 1, as shown in the diagram above).

Dot matrix drive mode corresponding LED display period configuration:

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LED0SEL0 | Led7 | Led6 | Led5 | Led4 | Led3 | Led2 | Led1 | Led0 |
| LED1SEL1 | Led15 | Led14 | Led13 | Led12 | Led11 | Led10 | Led9 | Led8 |
| LED2SEL2 | Led23 | Led22 | Led21 | Led20 | Led19 | Led18 | Led17 | Led16 |
| LED3SEL3 | Led31 | Led30 | Led29 | Led28 | Led27 | Led26 | Led25 | Led24 |
| LED4SEL4 | Led39 | Led38 | Led37 | Led36 | Led35 | Led34 | Led33 | Led32 |
| LED5SEL5 | Led47 | Led46 | Led45 | Led44 | Led43 | Led42 | Led41 | Led40 |
| LED6SEL6 | Led55 | Led54 | Led53 | Led52 | Led51 | Led50 | Led49 | Led48 |
| LED7SEL7 | Led63 | Led62 | Led61 | Led60 | Led59 | Led58 | Led57 | Led56 |

## 15.4 Relevant Registers

### 15.4.1 LED Drive Mode Selection Register (LEDMODE)

| F769H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDMODE | LEDMODE7 | LEDMODE6 | LEDMODE5 | LEDMODE4 | LEDMODE3 | LEDMODE2 | LEDMODE1 | LEDMODE0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7-0 LEDMODE LED drive mode selection register

0x55= When the LED dot matrix driver mode is enabled, the corresponding registers will take effect.

0xaa= When the LED dot matrix driver mode is enabled, the corresponding registers will take effect.

Other= Invalid

### 15.4.2 LED Dot Matrix Drive Control Register (LEDCON1)

| F765H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDCON1 | SCAN_START | DUTY2 | DUTY1 | DUTY0 | SCAN_MODE | LED_FIXCLK_EN | -- | -- |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | W | W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 SCAN_START: LED dot matrix driver scanning start bit

1= LED is started (LED_FIXCLK_EN=1 active)

0= LED is stopped

Bit6~Bit4 DUTY<2:0>: LED dot matrix selection

000= Invalid

001= 4*4

010= 5*5

011= 6*6

100= 6*7

101= 7*7

110= 7*8

111= 8*8

Bit3 SCAN_MODE: LED scan mode configuration

1= LED dot matrix drive cycle scan mode

0= LED dot matrix drive interrupt scan mode. (When the interrupt scan mode is selected and a scan is completed, the hardware clears the SCAN_START bit, the interrupt flag is set to 1, and the software rewrites SCAN_START to 1 and then resumes the scan.)

Bit2 LED_FIXCLK_EN: LED operation clock enable control

1= Enable

0= Disable

Bit1~Bit0 - Reserved

### 15.4.3 LED Clock Selection Register (LEDCKS)

| F766H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDCKS | - | - | DTSEL1 | DTSEL0 | - | - | CLK1 | CLK0 |
| R/W | W | W | R/W | R/W | W | W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit6 -- Reserved

Bit5~Bit4 DTSEL<1:0>: Dot matrix LED dead time selection

00: $12*T_{fix}$

01: $24*T_{fix}$

10: $48*T_{fix}$

11: $96*T_{fix}$

Bit3~Bit2 -- Reserved

Bit1~Bit0 CLK<1:0>: LED count clock source period ($T_{LED\_CLK}$) (step time)

00: $32*T_{fix}$

01: $64*T_{fix}$

10: $128*T_{fix}$

11: $256*T_{fix}$

### 15.4.4 LED Port Enable Control Register (LEDIOEN)

| F760H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDIOEN | IOEN7 | IOEN6 | IOEN5 | IOEN4 | IOEN3 | IOEN2 | IOEN1 | IOEN0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 IOEN<7:0>: LED7-LED0 port enable control bit (LED8 port is set in LEDRESEQ register)

1= Enable (match the port selected for LED dot matrix pattern to take effect).

0= Disable

### 15.4.5 LED Dot Matrix Drive First Segment Period Configuration Register (SCAN1W)

| F761H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| SCAN1W | SCAN1W7 | SCAN1W6 | SCAN1W5 | SCAN1W4 | SCAN1W3 | SCAN1W2 | SCAN1W1 | SCAN1W0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

(LEDMODE=0xaa, this register is valid.)

Bit7~Bit0 SCAN1W<7:0>: LED dot matrix drive mode, first segment LED lighting period configuration register.

Period1 = ({SCAN1W[7:0]}+1) * $T_{LEDCLK}$

### 15.4.6 LED Dot Matrix Drive Second Segment Period Configuration Register (SCAN2W)

| F762H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| SCAN2W | SCAN2W7 | SCAN2W6 | SCAN2W5 | SCAN2W4 | SCAN2W3 | SCAN2W2 | SCAN2W1 | SCAN2W0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

(LEDMODE=0xaa, this register is valid.)

Bit7~Bit0 SCAN2W<7:0>: LED dot matrix drive mode, second segment LED lighting period configuration register.

Period2 = ({SCAN2W [7:0]}+1) * $T_{LEDCLK}$

### 15.4.7 LED Dot Matrix Drive Display Data Register (LEDnDATA (n=0-7))

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDnDATA | LEDnDATA7 | LEDnDATA6 | LEDnDATA5 | LEDnDATA4 | LEDnDATA3 | LEDnDATA2 | LEDnDATA1 | LEDnDATA0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

LED0DATA Address: F740H; LED1DATA Address: F741H; LED2DATA Address: F744H; LED3DATA Address: F745H;

LED4DATA Address: F748H; LED5DATA Address: F749H; LED6DATA Address: F74CH; LED7DATA Address: F74DH;

Bit7~Bit0 LEDnDATA<7:0>: LED dot matrix drive mode, dot matrix display data configuration register.

1= LED dot matrix drive mode, LED lighting at corresponding coordinates on the dot matrix.

0= LED dot matrix drive mode, LED not lighting at corresponding coordinates on the dot matrix.

Corresponding LED display data configuration in dot matrix drive mode:

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LED0DATA | Led7 | Led6 | Led5 | Led4 | Led3 | Led2 | Led1 | Led0 |
| LED1DATA | Led15 | Led14 | Led13 | Led12 | Led11 | Led10 | Led9 | Led8 |
| LED2DATA | Led23 | Led22 | Led21 | Led20 | Led19 | Led18 | Led17 | Led16 |
| LED3DATA | Led31 | Led30 | Led29 | Led28 | Led27 | Led26 | Led25 | Led24 |
| LED4DATA | Led39 | Led38 | Led37 | Led36 | Led35 | Led34 | Led33 | Led32 |
| LED5DATA | Led47 | Led46 | Led45 | Led44 | Led43 | Led42 | Led41 | Led40 |
| LED6DATA | Led55 | Led54 | Led53 | Led52 | Led51 | Led50 | Led49 | Led48 |
| LED7DATA | Led63 | Led62 | Led61 | Led60 | Led59 | Led58 | Led57 | Led56 |

Note: LED0 to LED63 correspond to the 64 LED coordinates of the 8x8 matrix.

### 15.4.8 LED Dot Matrix Drive Period Selection Register (LEDnSEL (n=0-7))

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDnSEL | LEDnSEL7 | LEDnSEL6 | LEDnSEL5 | LEDnSEL4 | LEDnSEL3 | LEDnSEL2 | LEDnSEL1 | LEDnSEL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

LED0SEL Address: F750H; LED1SEL Address: F751H; LED2SEL Address: F754H; LED3SEL Address: F755H;

LED4SEL Address: F758H; LED5SEL Address: F759H; LED6SEL Address: F75CH; LED7SEL Address: F75DH;

Bit7~Bit0 LEDnSEL<7:0>: LED dot matrix drive mode, dot matrix display period configuration register.

1= LED dot matrix drive mode, select second period for LED at corresponding coordinates on the dot matrix.

0= LED dot matrix drive mode, select first period for LED at corresponding coordinates on the dot matrix.

### 15.4.9 LED Dot Matrix Drive Pin Current Control Register (LEDDRV)

| F76BH | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDDRV | -- | -- | -- | -- | SDRC3 | SDRC2 | SDRC1 | SDRC0 |
| R/W | W | W | W | W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit4 -- Reserved, set to 0.

Bit3~Bit0 SDRC<3:0>: Dot matrix LED pin source current drive selection control bit

The related pin drive enable must be active for it to take effect; otherwise, the current will default to the maximum value.

| | | | |
|---|---|---|---|
| 0000= | 0mA | 1000= | 26.9mA |
| 0001= | 4.9mA | 1001= | 31.8mA |
| 0010= | 7.4mA | 1010= | 34.3mA |
| 0011= | 12.2mA | 1011= | 39.2mA |
| 0100= | 14.7mA | 1100= | 41.6mA |
| 0101= | 19.5mA | 1101= | 46.1mA |
| 0110= | 22.1mA | 1110= | 49mA |
| 0111= | 26.9mA | 1111= | 50mA |

### 15.4.10 Pin Drive Enable Register (LEDSEGDR0)

| F76FH | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDSEGDR0 | - | | | LEDSEGDR0_4 | LEDSEGDR0_3 | LEDSEGDR0_2 | LEDSEGDR0_1 | LEDSEGDR0_0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7:5 Reserved

Bit4 LEDSEGDR0 _4:P10 pin drive enable bit

1= P10 pin drive enable; P10 pin source current drive is configured by LEDDRV[3:0].

0= P10 pin drive disable, P10 pin source current is driven to the default value.

Bit3 LEDSEGDR0 _3:P07 pin drive enable bit

1= P07 pin drive enable; P07 pin source current drive is configured by LEDDRV[3:0].

0= P07 pin drive disable, P07 pin source current is driven to the default value.

Bit2 LEDSEGDR0_2: P06 pin drive enable bit

1= P06 pin drive enable; P06 pin source current drive is configured by LEDDRV[3:0].

0= P06 pin drive disable, P06 pin source current is driven to the default value.

Bit1 LEDSEGDR0 _1:P05 pin drive enable bit

1= P05 pin drive enable; P05 pin source current drive is configured by LEDDRV[3:0].

0= P05 pin drive disable, P05 pin source current is driven to the default value.

Bit0 LEDSEGDR0 _0:P04 pin drive enable bit

1= P04 pin drive enable; P04 pin source current drive is configured by LEDDRV[3:0].

0= P04 pin drive disable, P04 pin source current is driven to the default value.

### 15.4.11 Pin Drive Enable Register (LEDSEGDR3)

| F772H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDSEGDR3 | -- | LEDSEGDR3 _6 | LEDSEGDR3 _5 | LEDSEGDR3 _4 | LEDSEGDR3_3 | - | - | - |
| R/W | R | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 -- Reserved, set to 0.

Bit6 LEDSEGDR3 _6:P03 pin drive enable bit

1= P03 pin drive enable; P03 pin source current drive is configured by LEDDRV[3:0].

0= P03 pin drive disable, P03 pin source current is driven to the default value.

Bit5 LEDSEGDR3 _5:P02 pin drive enable bit

1= P02 pin drive enable; P02 pin source current drive is configured by LEDDRV[3:0].

0= P02 pin drive disable, P02 pin source current is driven to the default value.

Bit4 LEDSEGDR3 _4:P01 pin drive enable bit

1= P01 pin drive enable; P01 pin source current drive is configured by LEDDRV[3:0].

0= P01 pin drive disable, P01 pin source current is driven to the default value.

Bit3 LEDSEGDR3 _3:P00 pin drive enable bit

1= P00 pin drive enable; P00 pin source current drive is configured by LEDDRV[3:0].

0= P00 pin drive disable, P00 pin source current is driven to the default value.

Bit2:0 -- Reserved, set to 0.

### 15.4.12 LED Dot Matrix Pin Mapping Register (LEDRESEQ)

| F76CH | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDRESEQ | - | - | - | - | LED8_EN | LREMAP2 | LREMAP1 | LREMAP0 |
| R/W | W | W | W | W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | | |
|---|---|---|
| Bit7~Bit4 | - | Reserved, set to 0. |
| Bit3 | LED8_EN | LED8 port enable bit |
| | 1= | Enable LED8 port function (active when the LED dot matrix is 8X8) |
| | 0= | LED8 port is GPIO or other functions |
| Bit2~Bit0 | LEDSEQ<2:0>: | LED dot matrix start port selection |
| | 000= | LED0 start, LED0-LED1-……-LED6- LED7 |
| | 001= | LED1 start, LED1-LED2-……-LED7- LED0 |
| | … | … |
| | 111= | LED7 start, LED7-LED0-……-LED5- LED6 |

# 15.5 LED Dot Matrix Drive Interrupt

## 15.5.1 LED Dot Matrix Drive Status Register (LEDSTATUS)

| F76AH | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| LEDSTATUS | -- | -- | -- | -- | -- | -- | LEDIE | LEDIF |
| R/W | W | W | W | W | W | W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit2 -- Reserved, set to 0.

Bit1 LEDIE: LED dot matrix drive mode interrupt enable bit

1= LED dot matrix drive mode interrupt enable

0= LED dot matrix drive mode interrupt disable

Bit0 LEDIF: LED dot matrix drive mode interrupt flag bit

1= LED dot matrix drive scanning complete

0= Cleared to 0 by software

## 15.5.2 Interrupt Priority Control Register (EIP3

| 0xBB | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| EIP3 | PXTDET | -- | -- | PTOUCH | PLVD | PLED | -- | -- |
| R/W | R/W | W | W | R/W | R/W | R/W | W | W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 PXTDET Crystal oscillation stop interrupt priority control bit

1= High-level interrupt

0= Low-level interrupt

Bit6-Bit5 -- Reserved, set to 0.

Bit4 PTOUCH Touch interrupt priority control bit

1= High-level interrupt

0= Low-level interrupt

Bit3 PLVD: LVD interrupt priority control bit

1= High-level interrupt

0= Low-level interrupt

Bit2 PLED: LED interrupt priority control bit

1= High-level interrupt

0= Low-level interrupt

Bit1-Bit0 -- Reserved, set to 0.

# 16. SPI Module

## 16.1 Overview

This SPI is a fully configurable SPI master/slave device that allows users to configure the polarity and phase of the serial clock signal (SCLK). The serial clock line (SCLK) synchronizes the shifting and sampling of data on two independent serial data lines, allowing simultaneous transmission and reception of SPI data. SPI enables communication between the MCU and serial peripheral devices, and it can also be used for inter-processor communication in multi-master systems. It is a technology-independent design that can be implemented in various process technologies.

The SPI system is flexible enough to connect directly to many standard peripheral devices from various manufacturers. To accommodate most available synchronous serial peripherals, the clock control logic allows the selection of clock polarity and phase. The system can be configured as either a master or slave device. When SPI is configured as the master device, the software selects one of eight different bit rates for the serial clock, with speeds up to the system clock divided by 4 (Fsys/4).

The SPI slave chip select is used to address SPI slave devices for serial data exchange. When SPI operates as the master device, the SPI auto-drive is selected by the Slave Select Control Register (SSCR). The SPI controller includes logic for error detection to support inter-processor communication, such as a write conflict detector that indicates when data is written to the serial shift register during transfer.

## 16.2 Features

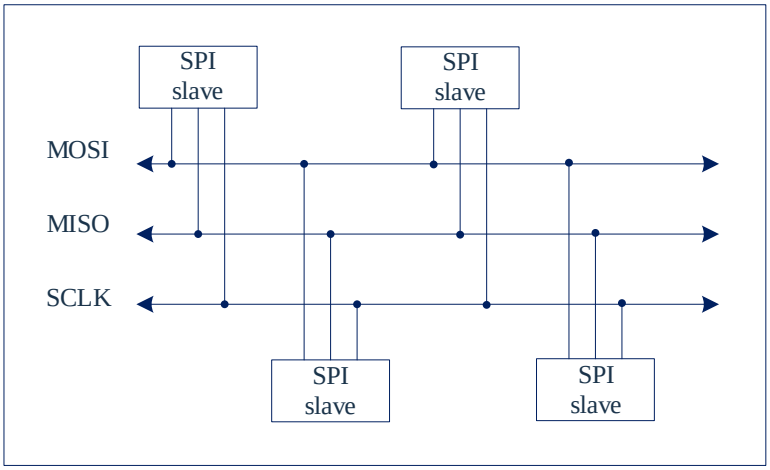
- Full-duplex synchronous serial data transfer.
- Supports master/slave mode.
- Supports multi-master systems.
- System error detection.
- Generates interrupts.
- Supports speeds up to 1/4 of the system clock ($F_{SYS} \leq 24$ MHz).
- Bit rate options: 1/4, 1/8, 1/16, 1/32, 1/64, 1/128, 1/256, 1/512 of the system clock.
- Supports four transfer formats.
- Simple interface for easy connection to microcontrollers.

## 16.3 SPI Port Configuration

To use the SPI function, the related ports must be configured as SPI channels, and the corresponding input ports should be selected through the communication input port registers. For example, to configure P00, P01, P02, and P03 as the SPI communication ports, the configuration code is as follows:

```
PS_NSS0OI = 0x00;        // Select P03 to configure as NSS channel
PS_SCLKOI = 0x00;        // Select P00 to configure as SCLK channel
PS_MOSI = 0x00;          // Select P01 to configure as MOSI channel
PS_MISO = 0x00;          // Select P02 to configure as MISO channel
```

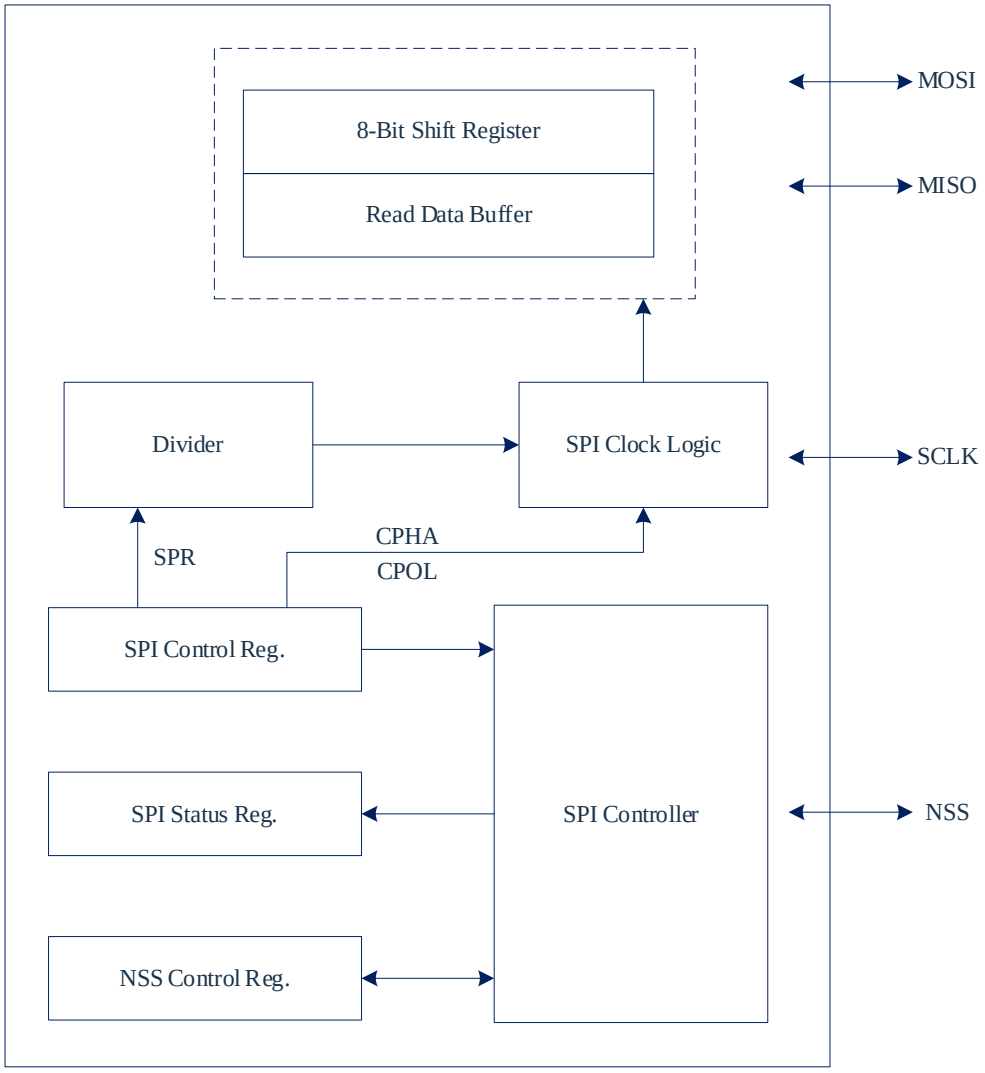
These pins are configured as SCLK, MOSI, MISO, and NSS, with pull-up resistors and open-drain output switches forcibly disabled. The multi-master SPI communication structure is shown in the diagram below:



## 16.4 SPI Hardware

During an SPI transfer, as an 8-bit character is shifted out of one data pin, another 8-bit character is shifted into a different data pin. The 8-bit shift register in the master device and another 8-bit shift register in the slave device are connected as a circular 16-bit shift register. When a transfer occurs, this distributed shift register is shifted by 8 bits, effectively swapping the characters between the master and slave.

The central element in the SPI system is the module that contains the shift register and the data buffer for reading. In the system, the transmission direction is single-buffered, while the reception direction is double-buffered. This means that new data can only be written into the shift register once the previous data transfer is complete. However, the received data is transferred into the parallel read data buffer, allowing the shift register to freely accept the next serial character. As long as the first character is read out from the read data buffer before the next serial character is ready for transfer, no overlap occurs. The SPI control block diagram is shown below:




The pins associated with SPI are: NSS, SCLK, MOSI, and MISO.

In master mode, the NSS output pin is used to select the slave device, while in slave mode, the NSS input pin is used to enable the transfer.

In master mode, the SCLK pin is used as the reference for the SPI clock signal. When the master device starts a transfer, eight clock cycles are automatically generated on the SCLK pin.

When SPI is configured as a slave device, the SI pin is the input data line for the slave, and the SO pin is the output data line for the slave.

When SPI is configured as a master device, the MI pin is the input data line for the master, and the MO pin is the output data line for the master.

## 16.5 SPI Relevant Registers

### 16.5.1 SPI Control Register (SPCR)

| 0xEC | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| SPCR | -- | SPEN | SPR2 | MSTR | CPOL | CPHA | SPR1 | SPR0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

| | | |
|---|---|---|
| Bit7 | -- | Reserved, set to 0. |
| Bit6 | SPEN: | SPI module enable bit |
| | 1= | Enable |
| | 0= | Disable |
| Bit5 | SPR2: | SPI clock frequency selection bit [2]. |
| Bit4 | MSTR: | SPI mode select bit |
| | 1= | Master mode |
| | 0= | Slave mode |
| Bit3 | CPOL: | SPI clock polarity select bit |
| | 1= | High when SCLK is idle. |
| | 0= | Low when SCLK is idle. |
| Bit2 | CPHA: | SPI clock phase select bit |
| Bit1~Bit0 | SPR<1:0>: | SPI clock frequency select bit [1:0] |
| | | (See the table below for frequency control details) |

SPR2-SPR0 control SPI clock division

| SPR2 | SPR1 | SPR0 | System clock division |
|---|---|---|---|
| 0 | 0 | 0 | 4 |
| 0 | 0 | 1 | 8 |
| 0 | 1 | 0 | 16 |
| 0 | 1 | 1 | 32 |
| 1 | 0 | 0 | 64 |
| 1 | 0 | 1 | 128 |
| 1 | 1 | 0 | 256 |
| 1 | 1 | 1 | 512 |

### 16.5.2 SPI Data Register (SPDR)

| 0xEE | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| SPDR | SPIDATA7 | SPIDATA6 | SPIDATA5 | SPIDATA4 | SPIDATA3 | SPIDATA2 | SPIDATA1 | SPIDATA0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | | |
|---|---|---|
| Bit7~Bit0 | SPIDATA: | SPI transmit/receive data |
| | Write: | Write the data to be transmitted (transmit order from high to low). |
| | Read: | Received data |

### 16.5.3 SPI Slave Selection Control Register (SSCR)

The Slave Select Control Register (SSCR) can be read or written at any time. It is used to configure which slave select output should be driven when confirming an SPI master transfer. When the SPI master transfer is initiated, the contents of the SSCR register are automatically assigned to the NSS pin.

| 0xEF | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| SSCR | -- | -- | -- | -- | -- | -- | -- | NSSO0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Bit7~Bit1 -- Reserved, set to 1.

Bit0 NSSO0: SPI slave select control bit (master chip select output NSS is NSSO0).

0= NSSO0 outputs 0 when SPI master transfer is initiated.

1= NSSO0 outputs 1 when SPI master transfer is initiated.

### 16.5.4 SPI Status Register (SPSR)

| 0xED | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| SPSR | SPISIF | WCOL | -- | -- | -- | -- | -- | SSCEN |
| R/W | R | R | -- | R | -- | -- | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 SPISIF: SPI transfer complete interrupt flag, read-only

1= SPI transfer is complete (read SPSR first, then read/write SPDR, and clear)

0= SPI is not complete

Bit6 WCOL: SPI write conflict interrupt flag, read-only

1= A write conflict occurs when SPDR is written during an ongoing SPI transfer (read SPSR first, then read/write SPDR, and clear)

0= No write conflict

Bit5~Bit1 -- Reserved, set to 0.

Bit0 SSCEN: SPI master mode NSS output control bit

1= When SPI is idle, the NSS output is high

0= NSS outputs the content of the SSCR register.

The SPI Status Register (SPSR) contains flags that indicate whether a transfer is complete or if a system error has occurred. When the corresponding event happens and is cleared sequentially by software, all flags are automatically set. By reading the SPSR and then accessing SPDR, SPISIF, and WCOL, these flags will be automatically cleared.

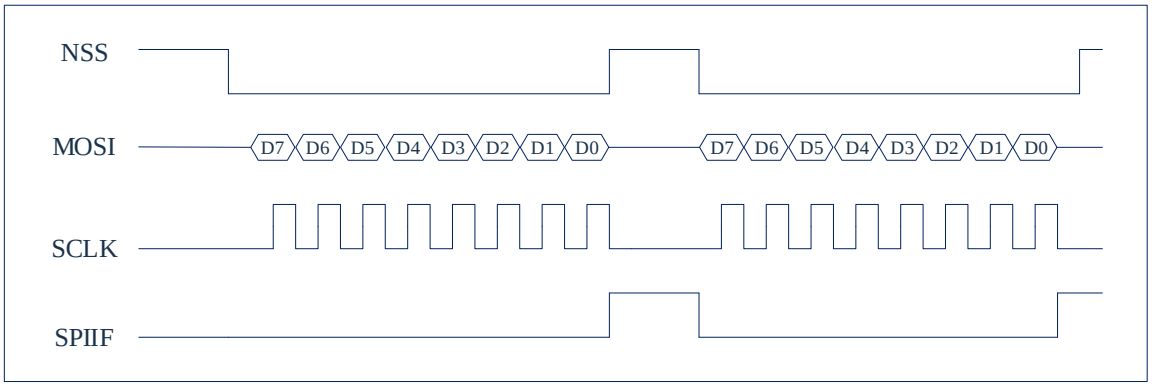
The SSCEN bit is the enable bit for automatic slave select output. When SSCEN is set to 1, during transfer, the NSS line outputs the content of the SSCR register. When the transfer is idle, the NSS line is high. When the SSCEN bit is cleared to 0, the NSS line always displays the content of the SSCR register.

## 16.6 SPI Master Mode

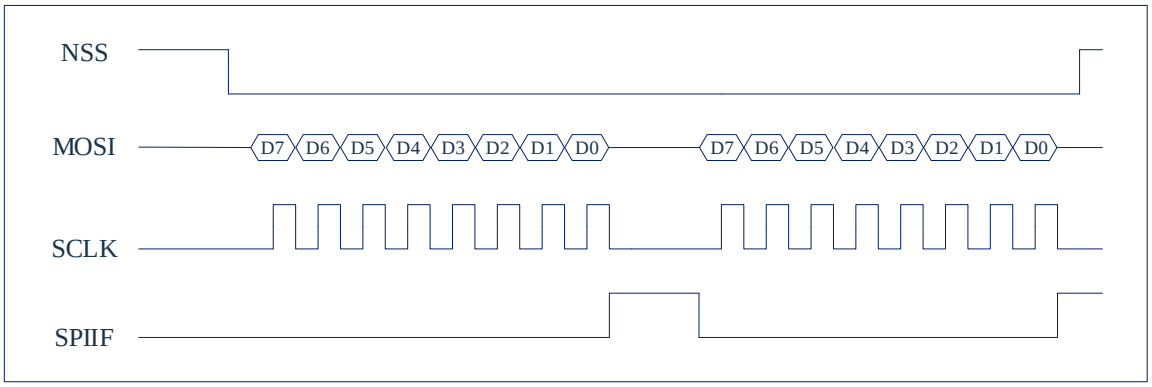
When the SPI is configured as the master mode, the transfer is initiated by writing to the SPDR register. When a new byte is written to the SPDR register, SPI starts the transfer. The serial clock SCLK is generated by SPI, and in master mode, SPI is enabled with SCLK output.

In master mode, SPI can select the SPI slave device via the NSS line. The NSS line (slave select) outputs the contents of the SSCR register. The SSCEN bit in the SPSR register controls the automatic NSS line control or software control. When SSCEN is set in master mode, and SSCEN is set to 1, during transfer, the NSS line outputs the contents of the SSCR register, and when the transfer is idle, NSS is high. When the SSCEN bit is cleared to 0, the NSS line is controlled by software and always displays the contents of the SSCR register, regardless of whether the transfer is in progress or the SPI is in an idle state.

When SSCEN=1, the SPI clock polarity CPOL=0, clock phase CPHA=0, and the slave select line is used as shown in the diagram below:




When SSCEN = 0, the SPI clock polarity CPOL = 0, clock phase CPHA = 0, and the slave select line is used as shown in the diagram below:



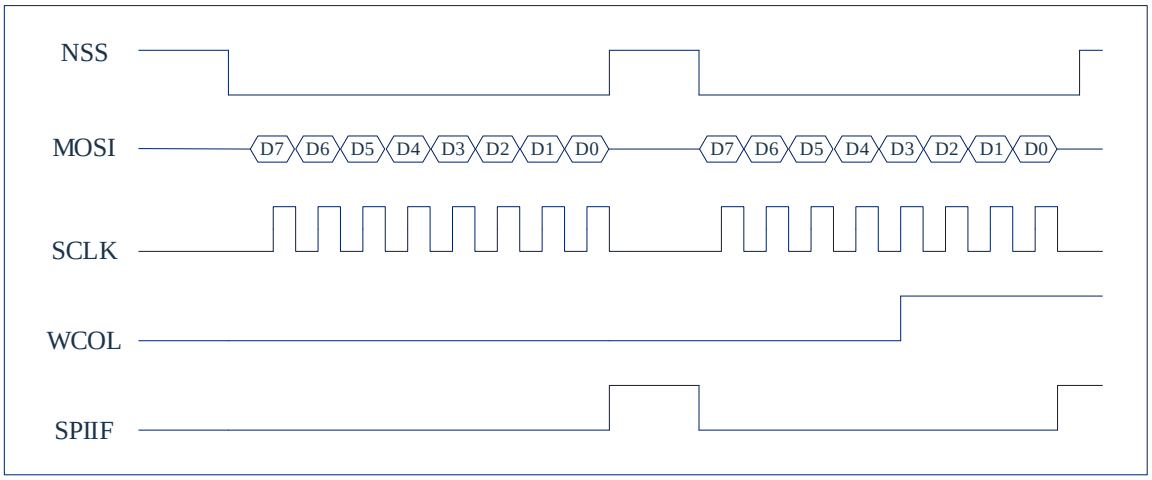
### 16.6.1 Write Conflict Error

A write conflict occurs if the SPI data register is written to during transfer. The transfer continues uninterrupted, and the erroneous data is not written to the shift register. The write conflict is indicated by the WCOL flag in the SPSR register.

When a WCOL error occurs, the WCOL flag is automatically set to 1 by hardware. To clear the WCOL bit, the user should perform the following steps:

- Read the contents of the SPSR register;
- Access the SPDR register (read or write).

In SPI master mode, when the SPI clock polarity is CPOL = 0 and the clock phase is CPHA = 0, the write conflict error is shown in the diagram below:




The specific conditions for a write conflict are as follows: During data transfer, when NSS is low, if SPDR is written to during the period from the start of the first data transmission until the 8th falling edge of SCLK, a write conflict will occur, and the WCOL flag will be set to 1.

Caution: When data transmission begins, NSS does not immediately go low after writing to SPDR; it takes up to one SPI clock cycle for NSS to go low. After NSS goes low, you must wait for one system clock cycle before the first data can be sent. Only at this point does the system enter the actual data transfer state. Writing to SPDR during the time between writing to SPDR and entering the actual data transfer state will not cause a write conflict. However, this operation will update the data prepared for transmission. If SPDR is written to multiple times, the data sent will be the last value written to SPDR.

Since the SPI has only one transmit buffer, it is recommended to check whether the previous data has been fully transmitted before writing to SPDR. Ensure that transmission is complete before writing to the SPDR register to prevent a write conflict.

## 16.7 SPI Slave Mode

When configured as an SPI slave device, SPI transfer is initiated by an external SPI master module using the SPI slave select input and generating the SCLK serial clock.

Before transfer begins, it is necessary to determine which SPI slave will be used to exchange data. The NSS is used (clear = 0), and the clock signal connected to the SCLK line will shift the contents of the SPI slave device's receive shift register to the MOSI line, and the contents of the transmit shift register will drive the MISO line. When all 8 bits have been shifted in/out, SPI will generate an interrupt request by setting the IRQ output. The contents of the shift register drive the MISO line.

In SPI slave mode, there can only be one type of transfer error—write conflict error.

### 16.7.1 Addressing Error

In slave mode, only the write conflict error can be detected by the SPI.

When a write operation to the SPDR register occurs during an ongoing SPI transfer, a write conflict error will happen.

In slave mode, when CPHA is cleared, a write conflict error may occur as long as the NSS slave select line is driven low, even if all bits have been transferred. This is because the start of the transfer is not explicitly indicated, and driving NSS low after a full-byte transfer may indicate the start of the next byte's transfer.

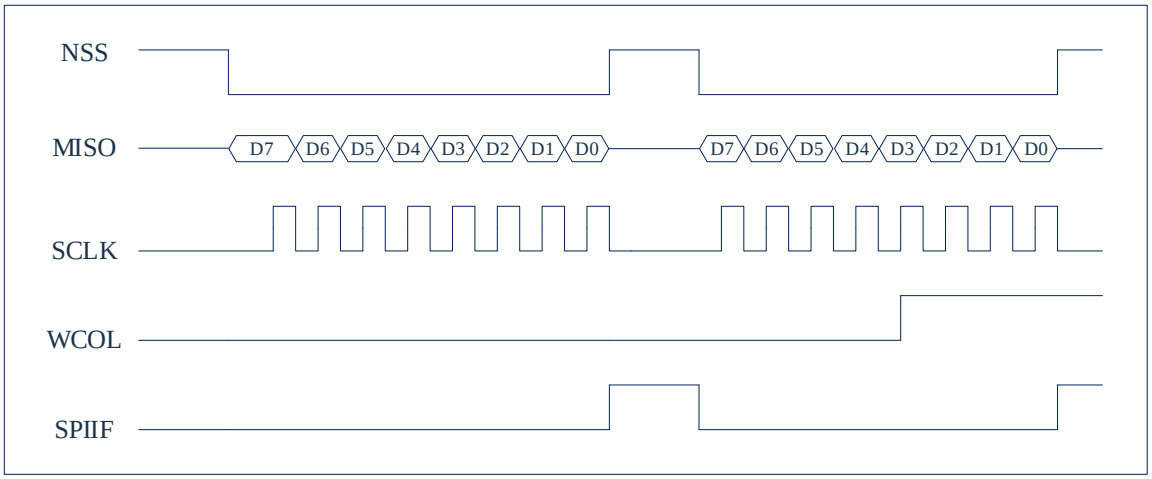
### 16.7.2 Write Conflict Error

A write conflict occurs if the SPI data register is written to during an ongoing transfer. The transfer continues without interruption, and the erroneous written data is not transferred to the shift register. The write conflict is indicated by the WCOL flag in the SPSR register.

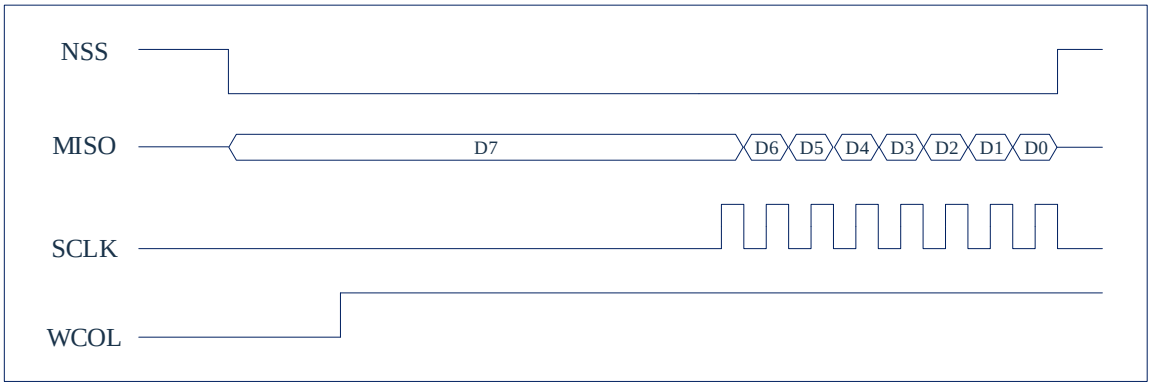
When a WCOL error occurs, the WCOL flag is automatically set to 1 by the hardware. To clear the WCOL bit, the user should follow this sequence:

- Read the contents of the SPSR register.
- Access the SPDR register (read or write).

The following diagram illustrates the write conflict error during the transfer in SPI slave mode:




In case CPHA is cleared, the WCOL generation can also be triggered by writing to the SPDR register when the NSS line is cleared. In this case, the SPI master does not generate the serial clock (SCLK), and the transfer can still proceed. This occurs because the start of the transfer is not explicitly indicated, and driving NSS low after a full byte transfer might indicate the start of the next byte's transfer. When the NSS line is low and the clock phase is CPHA = 0, writing to SPDR results in a write conflict error, as illustrated in the following diagram:




Additionally, in slave mode, after writing to the SPDR, the NSS controlled by the master does not immediately go low. When NSS is low, it is necessary to wait for the second edge of the SCLK before entering the actual data transfer state.

During the period between writing to the SPDR and starting to send the first data, writing to the SPDR again does not cause a write conflict. However, this operation will update the data that is ready to be sent. If multiple writes to the SPDR occur, the data sent will be the value of the last write to the SPDR.

During the period between the beginning of sending the first data and the second edge of the SCLK, writing to the SPDR again does not cause a write conflict, and it will not update the data being transmitted. In other words, the write to the SPDR during this period will be ignored.

Since the SPI only has one transmission buffer, it is recommended to check whether the previous data has been completely sent before writing to the SPDR, to ensure transmission completion before writing to the SPDR register, in order to avoid write conflicts.

# 16.8 SPI Clock Control Logic

## 16.8.1 SPI Clock Phase and Polarity Control

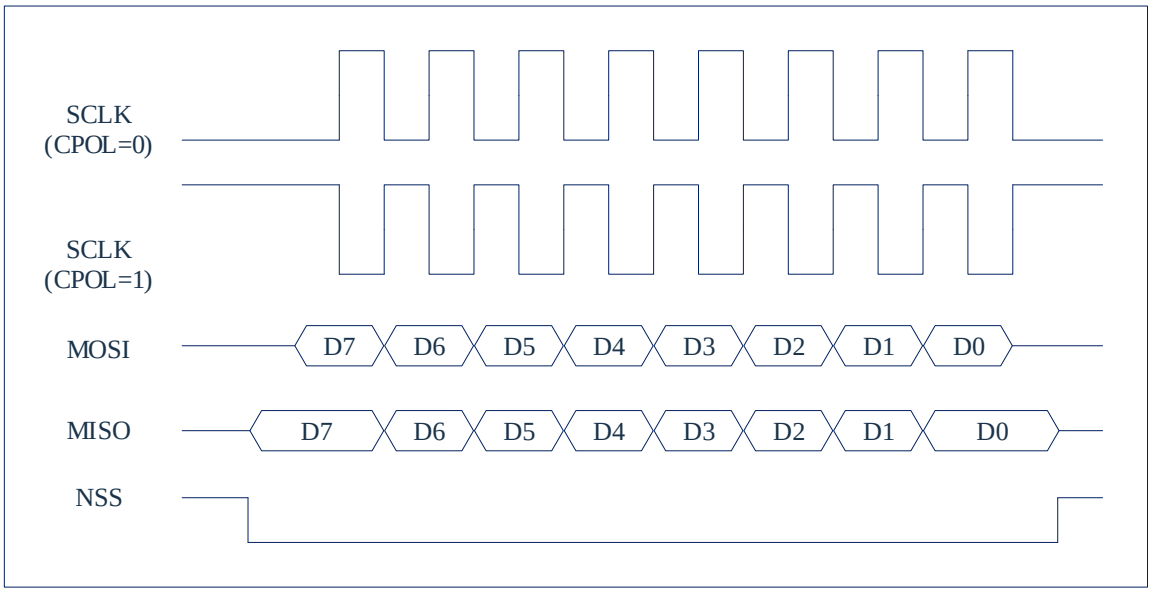
The software can choose from four combinations of clock phase and polarity in the SPI control register (SPCR) using two control bits. The clock polarity is specified by the CPOL control bit. The CPOL bit selects either a high or low level during idle periods of the transfer, which does not significantly affect the transfer format. The clock phase (CPHA) control bit selects one of two fundamentally different transfer formats. Both the master and slave SPI devices should have the same clock phase and polarity. In some cases, the phase and polarity can be changed during the transfer to allow communication between a master device and a slave device with different requirements. The flexibility of the SPI system allows it to directly connect with almost all existing synchronous serial peripherals.

## 16.8.2 SPI Transfer Format

During SPI transfer, data is simultaneously transmitted (serial shift-out) and received (serial shift-in). The serial clock line synchronizes with the two serial data lines for shifting and sampling. The slave select line allows independent selection of the slave SPI device; unselected slave devices do not interfere with the SPI bus activity. On the SPI master device, the slave select line can be used selectively to indicate multi-master bus contention.

## 16.8.3 CPHA=0 Transfer Format

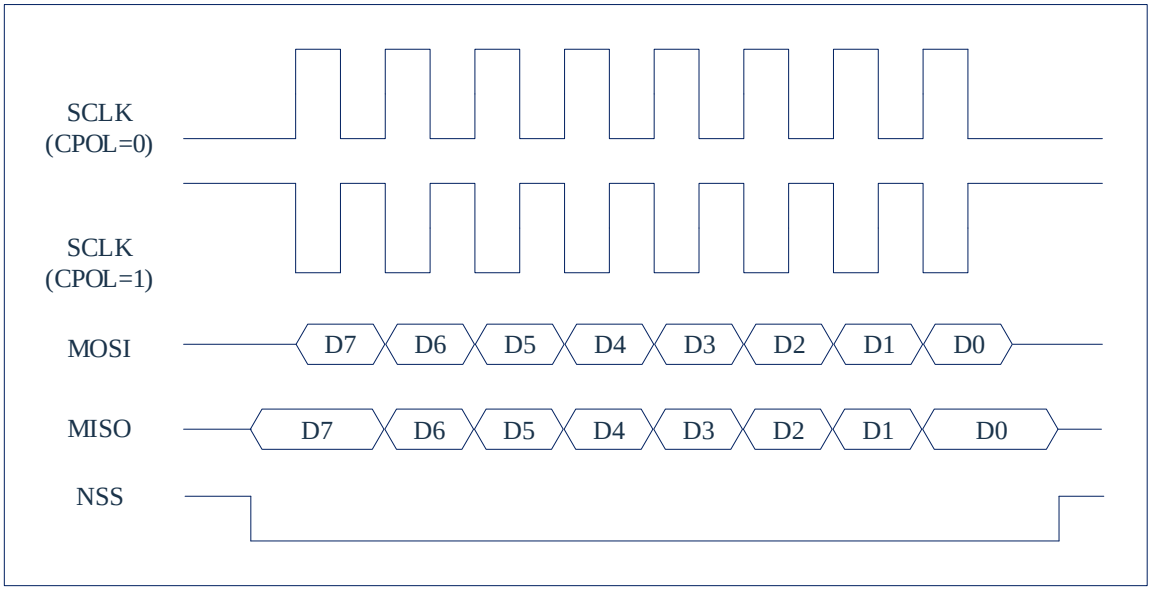
The diagram below shows the timing of an SPI transfer when CPHA is set to 0. Two waveforms are shown for SCLK: one for CPOL = 0 and the other for CPOL = 1. The diagram can describe either the master or slave device timing. The Master In/Slave Out (MISO) and Master Out/Slave In (MOSI) pins are directly connected between the master and slave. MISO is the slave output, and MOSI is the master output. The NSS line is the slave's slave select input. The master's NSS pin is not shown but is assumed to be invalid. The master's NSS pin must be high. This timing diagram functionally describes how the transfer is performed and should not be used as a substitute for data sheet parameter information.




When CPHA = 0, the NSS line must be released (set to 1) and re-set (set to 1) between each consecutive serial byte. Additionally, if the NSS line is low and the slave writes data into the SPI data register (SPDR), a write conflict error will occur. When CPHA = 1, the NSS line may remain low between consecutive transfers (it can always stay low). This format is sometimes preferred in systems with a single fixed master and a single slave driving the MISO data line.

### 16.8.4 CPHA=1 Transfer Format

The diagram below shows the timing of an SPI transfer when CPHA = 1. SCLK shows two waveforms: one for CPOL = 0 and the other for CPOL = 1. Since the SCLK, MISO, and MOSI pins are directly connected between the master and slave, the diagram can be interpreted as either the master or slave timing diagram. The MISO signal is the slave output, and the MOSI signal is the master output. The NSS line is the slave's slave select input; the master's NSS pin is not shown but is assumed to be inactive. The master's NSS pin must be high, or it must be reconfigured to not affect the general output of the SPI.



## 16.9 SPI Data Transfer

### 16.9.1 SPI Transfer Start

All SPI transfers are initiated and controlled by the master SPI device. As a slave device, SPI transfer begins either with the first SCLK edge or the falling edge of NSS, depending on the selected CPHA format. When CPHA = 0, the falling edge of NSS marks the start of the transfer. When CPHA = 1, the first edge of SCLK marks the start of the transfer. In both CPHA modes, the transfer can be terminated by setting the NSS line high, but this will reset the SPI slave logic and counters. The selected SCLK rate does not affect slave operation, as the master's clock controls the transfer.

When SPI is configured as a master, the transfer is initiated by writing to the SPDR register.

### 16.9.2 SPI Transfer End

Technically, the SPI transfer is complete when the SPIF flag is set to 1, but there may be additional tasks depending on the SPI system configuration. Since the SPI bit rate does not affect the end period, only the fastest rate is considered during the end period. When SPI is configured as a master, SPIF is set at the end of the eighth SCLK cycle. When CPHA = 1, the SCLK is inactive in the second half of the eighth SCLK cycle.

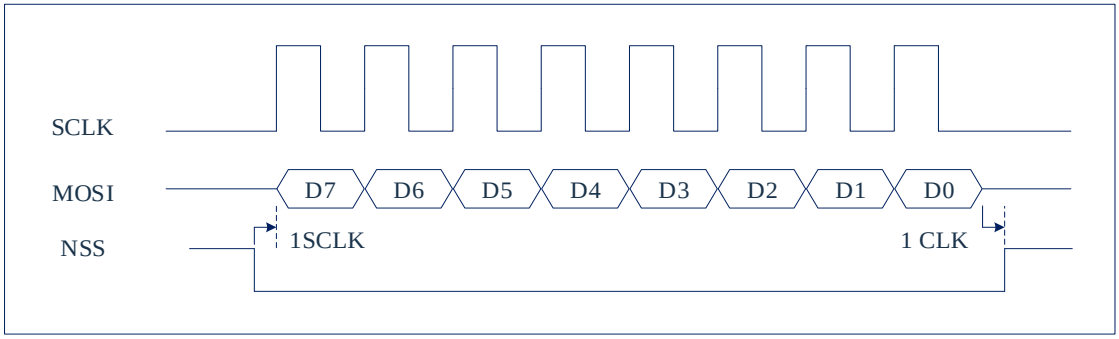
Since the SCLK line can be asynchronous with the slave's MCU clock and the slave cannot access as much information from the master as it can from the SCLK cycle, the end period is different when SPI operates as a slave. For example, when CPHA = 1, where the last SCLK edge occurs in the middle of the eighth SCLK cycle, the slave cannot determine when the previous SCLK cycle ends. For these reasons, the slave considers the transfer complete after the last bit of serial data is sampled, which corresponds to the middle of the eighth SCLK cycle.

The SPIF flag is set at the end of the transfer, but when the NSS line is still low, the slave is not allowed to write new data to the SPDR register.

# 16.10 SPI Timing Diagrams

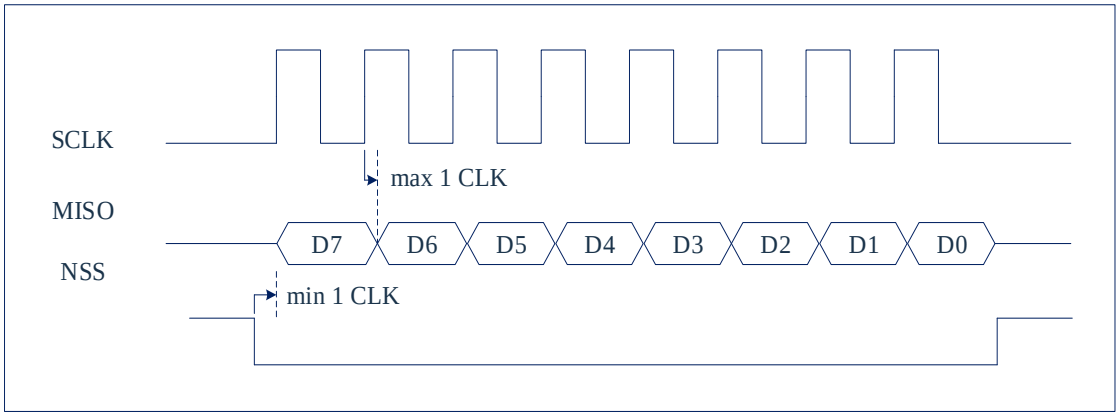
## 16.10.1 Master Mode Transfer

When the SPI clock polarity CPOL = 0 and clock phase CPHA = 1, in master mode, after the NSS line is pulled low, the MOSI starts outputting after one system clock cycle (CLK). The data on MOSI is output on the rising edge of the SCLK clock. The master mode timing diagram is shown in the figure below:




## 16.10.2 Slave Mode Transfer

When the SPI clock polarity CPOL = 0 and clock phase CPHA = 1, in slave mode, the data on MISO begins outputting after the falling edge of the NSS line. The output of MSIO data can be delayed by a maximum of 1 system clock cycle (CLK) after the falling edge of NSS. The slave mode timing diagram is shown in the figure below:



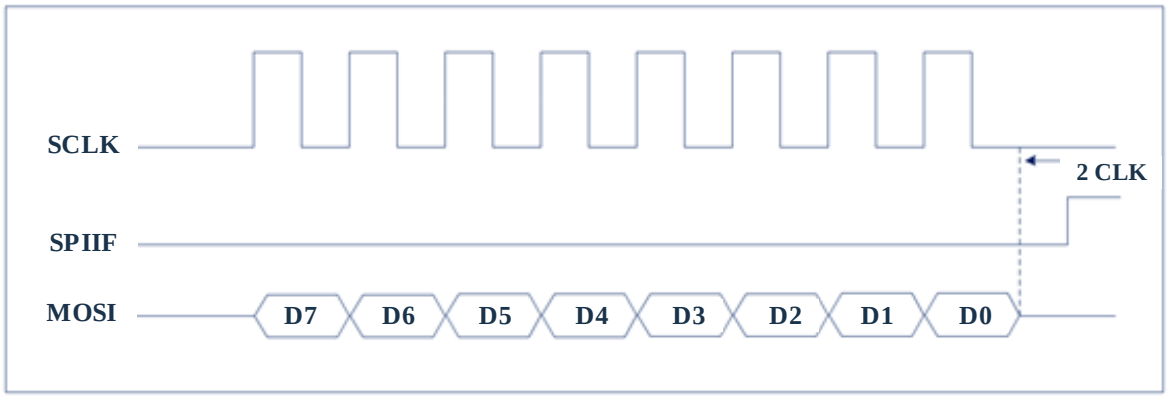
## 16.11 SPI Interrupt

The interrupt number for SPI is 22, and its interrupt vector is 0x00B3. To enable the SPI interrupt, the SPI interrupt enable bit (SPIIE) must be set to 1, and the global interrupt enable bit (EA) must be set to 1.

If all SPI-related interrupts are enabled, the SPI global interrupt flag (SPIIF) will be set to 1 when the SPI interrupt is triggered, and the CPU will enter the interrupt service routine. The SPIIF flag is read-only and is independent of the SPIIE state.

In the SPI status register (SPSR), when either the transfer completion flag (SPISIF) or the write conflict flag (WCOL) is set to 1, the SPI global interrupt flag (SPIIF) will also be set to 1. SPIIF will automatically clear to 0 only when all three of these flags are 0.

When the SPI clock polarity CPOL = 0 and clock phase CPHA = 1, in master mode, the SPIIF flag is set 2 SCLK cycles after the rising edge of the 8th SCLK clock of each data frame. The timing diagram is shown below:




### 16.11.1 Interrupt Mask Register (EIE2)

| 0xAA | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| EIE2 | SPIIE | I2CIE | -- | ADCIE | PWMIE | -- | -- | -- |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 SPIIE: SPI interrupt enable bit
1= Enable SPI interrupt
0= Disable SPI interrupt
Bit6 I2CIE: $I^2C$ interrupt enable bit
1= Enable $I^2C$ interrupt
0= Disable $I^2C$ interrupt
Bit5 -- Reserved, set to 0.
Bit4 ADCIE ADC interrupt enable bit
1= Enable ADC interrupt
0= Disable ADC interrupt
Bit3 PWMIE: PWM global interrupt enable bit
1= Enable PWM all interrupts
0= Disable PWM all interrupts
Bit2~Bit0 -- Reserved, set to 0.

### 16.11.2 Interrupt Priority Control Register (EIP2)

| 0xBA | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| EIP2 | PSPI | PI2C | -- | PADC | PPWM | PT5 | -- | -- |
| R/W | R/W | R/W | -- | R/W | R/W | R/W | R | R |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 PSPI: SPI interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit6 PI2C: $I^2C$ interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit5 -- Reserved, set to 0.
Bit4 PADC: ADC interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit3 PPWM: PWM interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit2 PT5: TIMER5 interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit1~Bit0 -- Reserved, set to 0.

### 16.11.3 Peripheral Interrupt Flag Register (EIF2)

| 0xB2 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| EIF2 | SPIIF | I2CIF | -- | ADCIF | PWMIF | TF5 | -- | -- |
| R/W | R | R | R | R/W | R | R/W | R | R |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 SPIIF: SPI global interrupt flag (read-only)
1= SPI generates an interrupt, (this bit is automatically cleared when the specific interrupt flag bit is cleared)
0= No SPI interrupt generated
Bit6 I2CIF: $I^2C$ global interrupt flag (read-only)
1= $I^2C$ generates an interrupt, (this bit is automatically cleared when the specific interrupt flag bit is cleared)
0= No $I^2C$ interrupt generated
Bit5 -- Reserved, set to 0.
Bit4 ADCIF: ADC interrupt flag bi
1= ADC conversion is completed, need to be cleared by software.
0= ADC conversion not completed
Bit3 PWMIF: PWM global interrupt flag (read-only)
1= PWM generates an interrupt, (this bit is automatically cleared when the specific interrupt flag bit is cleared)
0= No PWM interrupt generated
Bit2 TF5: Timer5 overflow interrupt flag bit
1= Timer5 overflow interrupt flag bit
0= Timer5 timer no overflow
Bit1~Bit0 -- Reserved, set to 0.

# 17. I²C Serial Interface Controller (I²C)

## 17.1 Overview

I²C is a two-wire, bidirectional serial bus that provides a simple and efficient means of data exchange between devices. I²C is a true multi-master bus, with conflict detection and arbitration mechanisms. These mechanisms are used to prevent data corruption when two or more masters attempt to control the bus simultaneously.

## 17.2 Features

- Supports master/slave modes.
- Bidirectional data transfer between master and slave.
- Multi-master bus.
- Arbitration for simultaneous data transfer between multiple masters, preventing serial data corruption on the bus.
- Uses a serial synchronized clock, allowing devices to communicate at different rates.
- Programmable clock for multiple rate control.
- Supports 7-bit/10-bit slave address modes.

## 17.3 Register Map

RO: Read-only; WO: Write-only; R/W: Read/Write.

| Register | Address | R/W | Description | Reset value |
|---|---|---|---|---|
| I2CCON | 0XF1 | R/W | I²C Control Register | 0x00 |
| I2CCLR | 0XF2 | WO | I²C Clear Register | 0x00 |
| I2CSTAT | 0XF3 | RO | I²C Status Register | 0xF8 |
| I2CDAT | 0XF4 | R/W | I²C Data Register | 0x00 |
| I2CCLK | 0XF5 | R/W | I²C Clock Control Register | 0x00 |
| I2CADR | 0XF6 | R/W | I²C Slave Address Register | 0x00 |
| I2CXAR | 0XF7 | R/W | I²C Extended Address Reset Register | 0x00 |

## 17.4 Register Description

### 17.4.1 I²C Control Register (I2CCON)

| 0XF1 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| I2CCON | I2CIE | I2CEN | STA | STO | SI | AA | XADRF | ADRF |
| R/W | R/W | R/W | R/W | R/W | R | R/W | R | R |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 I2CIE: Interrupt enable bit
0= Disable
1= Enable

Bit6 I2CEN: I²C interface enable bit
0= Disable I2C interface
1= Enable I²C Interface (default is slave mode)

Bit5 STA: Start flag bit
1= When I²C enters master mode, it sends a start signal.
- If I²C is already in master mode, a restart signal is sent.
- If I²C is in slave mode, writing 1 will end the current transfer and wait for the bus to become idle to enter master mode.

0= No effect
When the start or restart signal is sent, this bit is automatically cleared.

Bit4 STO: Stop flag bit
In master mode, writing 1 sends a stop bit.
In slave mode, writing 1 will be treated as receiving a stop bit.
Stop bit
- When both STA and STO are set, the I²C module first sends a stop bit, then a start bit.
- When the stop bit is transmitted, this bit is automatically cleared.

Bit3 SI: I²C interrupt flag bit (read-only)
This bit is set when there is a bus state change in I²C. It can be cleared by writing 1 to the SIC bit.

Bit2 AA: Acknowledge flag bit
0= No ACK signal received.
1= The ACK signal is sent in the following situations:
- When the slave address matches.
- When broadcast calling is enabled and the broadcast address is received.

This bit can be cleared by writing 1 to the AAC bit when data is received in master or slave mode.

Bit1 XADRF: I²C slave 10-bit address flag bit (read-only)
0= No I²C address match.
1= 10-bit I²C address match.
This bit is cleared when new data is sent or received.

Bit0 ADRF: I²C slave 7-bit address flag bit (read-only)
0= No I²C address match.
1= 7-bit I²C address match.
This bit is cleared when new data is sent or received.

### 17.4.2 I²C Clear Register (I2CCLR)

| 0XF2 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| I2CCLR | I2CIE | I2CEN | STAC | - | SIC | AAC | - | - |
| WO | WO | WO | WO | WO | WO | WO | WO | WO |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 I2CIE: I²C interrupt disable bit
0= No effect
1= Clear the I2CIE bit

Bit6 I2CEN: I²C intergface disable bit
0= No effect.
1= Clear the I2CEN bit

Bit5 STAC: Start flag clear bit
0= No effect.
1= Clear the STA bit

Bit4 - Reserved

Bit3 SIC: I2C interrupt flag bit
0= No effect.
1= Clear the SI bit

Bit2 AAC= I²C acknowledge flag clear bit
0= No effect.
1= Clear the AA bit

Bit1~0 - Reserved

Note: I²C operations require clearing the corresponding flag bits to transition to the next state.

### 17.4.3 I²C Status Register (I2CSTAT)

| 0XF3 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| I2CSTAT | Status | Status | Status | Status | Status | Status | Status | Status |
| RO | RO | RO | RO | RO | RO | RO | RO | RO |
| Reset value | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |

Bit7~Bit0 Status: I²C status code

00H: Bus error (valid only in master mode)
08H: Start bit transmission complete
10H: Restart bit transmission complete
18H: Address + write bit transmission complete, ACK received
20H: Address + write bit transmission complete, ACK not received
28H: Data transmission complete in master mode, ACK received
30H: Data transmission complete in master mode, ACKk not received
38H: Arbitration failure during address or data transfer
40H: Address + read bit transmission complete, ack received
48H: Address + read bit transmission complete, ack not received
50H: Data received in master mode, ACK sent
58H: Data received in master mode, no ACK sent
60H: Slave mode: address + write bit received, ack sent
68H: Master arbitration failure, slave address + write bit received, ack sent
70H: Broadcast call address received, ack sent
78H: Master arbitration failure, broadcast call address received, ACK sent
80H: Slave mode: data received after address match, ACK sent
88H: Slave mode: data received after address match, no ack sent
90H: Slave mode: data received after receiving broadcast call address, ACK sent
98H: Slave mode: data received after receiving broadcast call address, no ACK sent
A0H: Slave mode: stop or restart signal received
A8H: Slave mode: address + read bit received, ACK sent
B0H: Master arbitration failure, slave address + read bit received, ACK sent
B8H: Slave mode: data sent, ACK received
C0H: Slave mode: data sent, no ACKk received
C8H: Slave mode: last data sent, ACK received
D0H: Slave mode: last data sent, no ACK received
D8H: Unused
E0H: Second address sent in master mode, ACK received
E8H: Second address sent in master mode, no ACK received
F0H: Unused
F8H: Undefined state
Other: Reserved

### 17.4.4 I²C Data Register (I2CDAT)

| 0XF4 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| I2CDAT | Data7 | Data6 | Data5 | Data4 | Data3 | Data2 | Data1 | Data0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 Data<7:0> The data received or to be transmitted.

### 17.4.5 I²C Clock Control Register (I2CCLK)

| 0XF5 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| I2CCLK | - | M2 | M1 | M0 | N3 | N2 | N1 | N0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 - Reserved, set to 0.

Bit6~Bit4 M<2:0>: Sample clock = $F_{SYS}/2^M$

Bit3~Bit0 N<3:0>: SCL clock = $F_{SYS}$ / ($2^M$× (N+1)×10)

### 17.4.6 I²C Slave Address Register (I2CADR)

| 0XF6 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| I2CADR | Address6 | Address5 | Address4 | Address3 | Address2 | Address1 | Address0 | GC |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit1 Address<6:0>: Slave address ADRS[6:0].

Bit0 GC: Enable broadcast call address recognition control

1= Enable broadcast call address recognition

0= Disable broadcast call address recognition

### 17.4.7 I²C Extended Address Reset Register (I2CXAR)

| 0XF7 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| I2CXAR | RST | GCF | - | - | - | XADR2 | XADR1 | XADR0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 RST: Software reset

1= Generate software reset

0= Invalid

Bit6 GCF: I²C broadcast call flag, read-only

1= Broadcast call address matched

0= No broadcast call received

Bit5~Bit3 - Reserved, set to 0.

Bit2~Bit0 XADR<2:0>: High 3 bits of extended sSlave 10-bit address: ADRS[9:7]

# 18. UARTn Module

## 18.1 Overview

The Universal Asynchronous Receiver/Transmitter (UART0 / UART1) provides a flexible method for full-duplex data exchange between external devices.

UARTn internally has two physically independent buffers for reception and transmission, SBUFn. The distinction between operating the reception buffer or the transmission buffer is made through read and write instructions to SBUFn. When writing to SBUFn, data is loaded into the transmission buffer; when reading from SBUFn, the contents of the reception buffer are read.

UARTn supports two asynchronous modes: Mode 1 and Mode 3. Mode 3 features multi-device communication, which can be enabled by setting the SMn2 bit in the SCONn register to 1. The host processor first sends an address byte to identify the target slave. The address byte is different from the data byte because the 9th bit of the address byte is 1, while the data byte is 0. When SMn2=1, slaves are not interrupted by data bytes. The address byte interrupts all slaves. The addressed slave will clear its SMn2 bit and prepare to receive the incoming data byte. Slaves that are not addressed will set SMn2 to 1 and ignore the incoming data.

## 18.2 UARTn Port Configuration

Before using the UARTn module, the corresponding port needs to be configured as the TXDn and RXDn channels of UARTn. For example, the port configuration for UART0 is as follows:

```
PS_TXD0 = 0x00;      //Configure P04 as TXD0
PS_RXD0 = 0x01;      //Configure P05 as RXD0
```

The RXDn port of UARTn can be selected through PS_RXDn (only one RXDn pin can be selected), and the TXDn port can be selected through PS_TXDn (only one TXDn pin can be selected).

It is recommended to first set the working mode and then configure the corresponding ports as serial ports.

## 18.3 UARTn Baud Rate

UARTn supports only Mode 1 and Mode 3, and the baud rate is set by Timer1 or BRT (Baud Rate Timer).

### 18.3.1 Baud Rate Clock Source

In Mode 1 and Mode 3, the baud rate clock source for UARTn is selected as follows:

1) For UART0 baud rate clock source:

   When CKCON[3] = 1, Timer1 is selected as the baud rate generator for UART0.

   When CKCON[3] = 0, BRT is selected as the baud rate generator for UART0.

2) For UART1 baud rate clock source:

   When CKCON[4] = 1, Timer1 is selected as the baud rate generator for UART1.

   When CKCON[4] = 0, BRT is selected as the baud rate generator for UART1.

### 18.3.2 Baud Rate Calculation

In Mode 1 and Mode 3 of UARTn, the baud rate calculation differs depending on the selected clock source. The calculation formulas are as follows:

1) Timer1 operates in 8-bit auto-reload mode. The baud rate formula is:

$$BaudRate=\frac{Fsys}{16\times(4\times3^{1-T1M})\times(256-TH1)}$$

T1M is the Timer1 clock selection bit. To find the value of TH1 for a given baud rate:

$$TH1=256-\frac{Fsys}{16\times(4\times3^{1-T1M})\times BaudRate}$$

2) When BRT is used as the Baud Rate Generator:

$$BaudRate=\frac{Fsys}{16\times(\{BRTDH,BRTDL\}+1)}$$

{BRTDH,BRTDL}:

$$\{BRTDH,BRTDL\}=\frac{Fsys}{16\times BaudRate}-1$$

### 18.3.3 Baud Rate Error

In Mode 1 and Mode 3 of UARTn, when selecting different baud rate clock sources, the baud rate error varies.

Table 1) provides information on the baud rate and the corresponding error for Timer1's 8-bit auto-reload mode in the variable baud rate mode.

1) T1M=1

| Baud rate | Fsys=8MHz | | | Fsys=16MHz | | | Fsys=24MHz | | | Fsys=48MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| bps | TH1 | Actual Rate | %(Error) | TH1 | Actual Rate | %(Error) | TH1 | Actual Rate | %(Error) | TH1 | Actual Rate | %(Error) |
| 4800 | 230 | 4808 | 0.16 | 204 | 4808 | 0.16 | 178 | 4808 | 0.16 | 100 | 4808 | 0.16 |
| 9600 | 243 | 9615 | 0.16 | 230 | 9615 | 0.16 | 217 | 9615 | 0.16 | 178 | 9615 | 0.16 |
| 19200 | -- | -- | -- | 243 | 19231 | 0.16 | 236 | 18750 | 2.34 | 217 | 19231 | 0.16 |
| 38400 | - | - | - | | | | 246 | 37500 | 2.34 | 236 | 37500 | 2.34 |
| 115200 | -- | -- | -- | | | | | | | | | |
| 250000 | -- | -- | -- | | | | | | | | | |

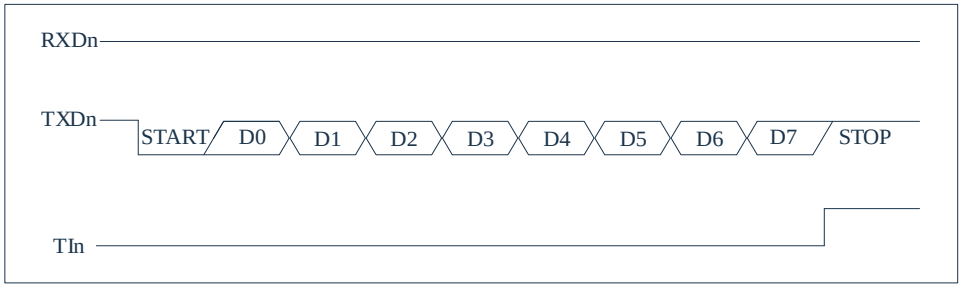
2) BRT baud rate clock

| Baud rate | Fsys=8MHz | | | Fsys=16MHz | | | Fsys=24MHz | | | Fsys=48MHz | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| bps | {BRTDH, BRTDL} | ActualRate | %(Error) | {BRTDH, BRTDL} | Actual Rate | %(Error) | {BRTDH, BRTDL} | Actual Rate | %(Error) | {BRTDH, BRTDL} | ActualRate | % (Error) |
| 4800 | 103 | 4808 | 0.16 | 207 | 4808 | 0.16 | 312 | 4792 | 0.16 | 624 | 4800 | 0 |
| 9600 | 51 | 9615 | 0.16 | 103 | 9615 | 0.16 | 155 | 9615 | 0.16 | 312 | 9585 | 0.16 |
| 19200 | 25 | 19231 | 0.16 | 51 | 19231 | 0.16 | 77 | 19231 | 0.16 | 155 | 19231 | 0.16 |
| 38400 | 12 | 38462 | 0.16 | 25 | 38462 | 0.16 | 38 | 38462 | 0.16 | 77 | 38462 | 0.16 |
| 115200 | -- | -- | -- | 8 | 111111 | 3.55 | 12 | 115385 | 0.16 | 25 | 115385 | 0.16 |
| 250000 | -- | -- | -- | -- | -- | -- | -- | -- | -- | 11 | 250000 | 0 |
| 500000 | -- | -- | -- | -- | -- | -- | -- | -- | -- | 5 | 500000 | 0 |
| 1000000 | -- | -- | -- | -- | -- | -- | -- | -- | -- | 2 | 1000000 | 0 |

## 18.4 UARTn Mode

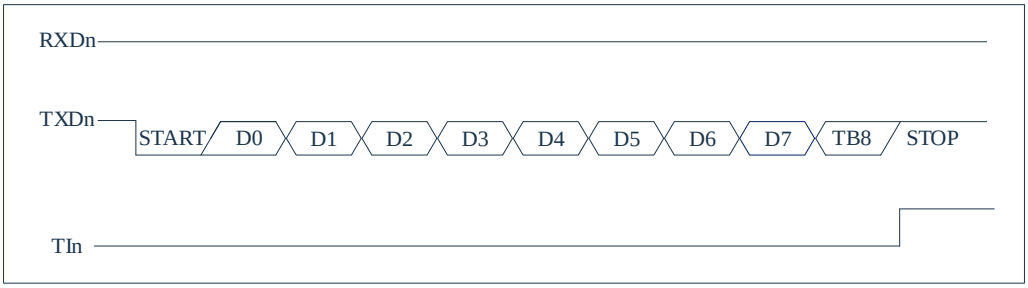
### 18.4.1 Mode 1 - 8-Bit Asynchronous Mode (Variable Baud Rate)

In Mode 1, the RXDn pin is used as input, and the TXDn pin is used for serial output. The transmission consists of 10 bits: a start bit (always 0), 8 data bits (LSB first), and a stop bit (always 1). During reception, the start bit is synchronized, and by reading SBUFn, the 8 data bits can be obtained. The stop bit is flagged in SCONn as TIN. The baud rate is variable and depends on the TIMER1/BRT mode. The timing diagram for Mode 1 is shown below:



### 18.4.2 Mode 3 - 9-Bit Asynchronous Mode (Variable Baud Rate)

The only difference between Mode 1 and Mode 3 is the addition of the 9th data bit, TB8, in Mode 3. When UnREN = 1, data reception is enabled. The baud rate is variable and depends on the TIMER1/BRT mode. The timing diagram for Mode 4 is shown below:



## 18.5 UARTn Registers

The UARTn has the same functionality as the standard 8051 UART. The related registers are: SBUFn, SCONn, PCON, IE, IP. The UARTn data buffer (SBUFn) consists of two separate registers: the transmit register and the receive register. Writing data to SBUFn sets this data in the UARTn output register and begins transmission. Reading from SBUFn retrieves data from the UARTn receive register. The SCON0 register supports bit-addressable operations, whereas the SCON1 register does not, which should be taken into account when using assembly language.

### 18.5.1 UARTn Buffer Register (SBUFn)

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| SBUFn | BUFFERn7 | BUFFERn6 | BUFFERn5 | BUFFERn4 | BUFFERn3 | BUFFERn2 | BUFFERn1 | BUFFERn0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register SBUF0 Address 0x99; Register SBUF1 Address 0xEB.

Bit7~Bit0 BUFFERn<7:0>: Buffer data register

Write: UARTn starts sending data.

Read: Read the received data.

### 18.5.2 UART Control Register (SCONn)

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| SCONn | UnSM0 | UnSM1 | UnSM2 | UnREN | UnTB8 | UnRB8 | TIn | RIn |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Register SCON0 Address 0x98; Register SCON1 Address 0xEA.

Bit7~Bit6 UnSM0-UnSM1: Multi-machine communication control bit
- 00= Reserved;
- 01= 8-bit asynchronous mode, baud rate variable
- 10= Reserved, selection forbidden
- 11= 9-bit asynchronous mode, baud rate variable

Bit5 UnSM2: Multi-machine communication control bit
- 1= Enable
- 0= Disable

Bit4 UnREN: Receive enable bit
- 1= Enable
- 0= Disable

Bit3 UnTB8: 9th data bit for transmission (mainly used for sending in 9-bit asynchronous mode)
- 1= 9th data bit is 1
- 0= 9th data bit is 0

Bit2 UnRB8: 9th data bit for reception (mainly used for receiving in 9-bit asynchronous mode)
- 1= Received 9th data bit is 1
- 0= Received 9th data bit is 0

Bit1 Tln: Transmit interrupt flag bit (requires software clearing)
- 1= The transmit buffer is empty and the next frame of data can be transmitted
- 0= --

Bit0 RIn: Receive interrupt flag bit (requires software clearing)
- 1= The receive buffer is full, and after reading, the next frame of data can be received.
- 0= --

The UARTn modes are listed in the following table:

| SMn0 | SMn1 | Mode | Description | Baud rate |
|---|---|---|---|---|
| 0 | 0 | 0 | - | - |
| 0 | 1 | 1 | 8-Bit UART | Controlled by BRT/Timer1 |
| 1 | 0 | 2 | - | 0 |
| 1 | 1 | 3 | 9-Bit UART | Controlled by BRT/Timer1 |

## 18.6 UARTn Interrupt

The interrupt number for UART0 is 4, with the interrupt vector at 0x0023.

The interrupt number for UART1 is 6, with the interrupt vector at 0x0033.

To enable UARTn interrupts, the corresponding enable bit ESn must be set to 1, and the global interrupt enable bit EA must also be set to 1. If the interrupt enables for UARTn are both active, and TIn=1 or RIn=1, the CPU will enter the corresponding interrupt service routine. The TIn/RIn status is independent of the ESn state and requires software clearing. For detailed description, refer to the register SCONn.

### 18.6.1 Interrupt Mask Register (IE)

| 0xA8 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| IE | EA | ES1 | ET2 | ES0 | ET1 | EX1 | ET0 | EX0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 EA: Global interrupt enable bit
1= Enable all unmasked interrupts
0= Disable all interrupts
Bit6 ES1: UART1 interrupt enable bit
1= Enable UART1 interrupt
0= Disable UART1 interrupt
Bit5 ET2: TIMER2 global interrupt enable bit
1= Enable TIMER2 all interrupts
0= Disable TIMER2 all interrupts
Bit4 ES0: UART0 interrupt enable bit
1= Enable UART0 interrupt
0= Disable UART0 interrupt
Bit3 ET1: TIMER1 interrupt enable bit
1= Enable TIMER1 interrupt
0= Disable TIMER1 interrupt
Bit2 EX1: External interrupt 1 interrupt enable bit
1= Enable external interrupt 1 interrupt
0= Disable external interrupt 1 interrupt
Bit1 ET0: TIMER0 interrupt enable bit
1= Enable TIMER0 interrupt
0= Disable TIMER0 interrupt
Bit0 EX0: External interrupt 0 interrupt enable bit
1= Enable external interrupt 0 interrupt
0= Disable external interrupt 0 interrupt

### 18.6.2 Interrupt Priority Control Register (IP)

| 0xB8 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| IP | -- | PS1 | PT2 | PS0 | PT1 | PX1 | PT0 | PX0 |
| R/W | W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 -- Reserved, set to 0.

Bit6 PS1: UART1 interrupt priority control bit

1= High-level interrupt

0= Low-level interrupt

Bit5 PT2: TIMER2 interrupt priority control bit

1= High-level interrupt

0= Low-level interrupt

Bit4 PS0: UART0 interrupt priority control bit

1= High-level interrupt

0= Low-level interrupt

Bit3 PT1: TIMER1 interrupt priority control bit

1= High-level interrupt

0= Low-level interrupt

Bit2 PX1: External interrupt 1 interrupt priority control bit

1= High-level interrupt

0= Low-level interrupt

Bit1 PT0: TIMER0 interrupt priority control bit

1= High-level interrupt

0= Low-level interrupt

Bit0 PX0: External interrupt 0 interrupt priority control bit

1= High-level interrupt

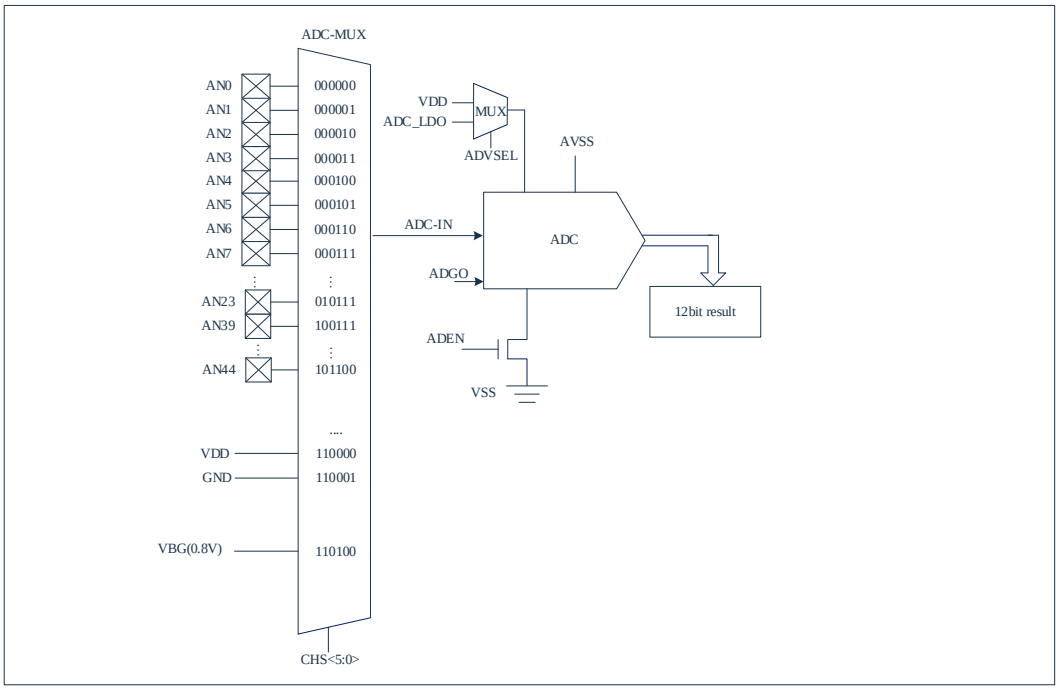
0= Low-level interrupt

# 19. Analog-to-Digital Converter (ADC)

## 19.1 Overview

The Analog-to-Digital Converter (ADC) can convert an analog input signal into a 12-bit binary number that represents the signal. The block diagram of the ADC structure is shown below.

The port analog input signal and internal analog signals are connected to the input of the ADC after passing through a multiplexer. The ADC uses a successive approximation method to generate a 12-bit binary result, which is then stored in the ADC result registers (ADRESL and ADRESH). Once the conversion is completed, the ADC can trigger an interrupt. The reference voltage for the ADC is selected by the control signal LDOEN, which can choose either VDD or ADC_LDO.



## 19.2 ADC Configuration

When configuring and using the ADC, the following factors must be considered:

- Port configuration
- Channel selection
- ADC conversion clock source
- Interrupt control

### 19.2.1 Port Configuration

The ADC can convert both analog and digital signals. When converting an analog signal, the corresponding port must be configured as an analog port.

Note: Applying an analog voltage to a pin defined as a digital input may cause overcurrent in the input buffer.

### 19.2.2 Channel Selection

The channel connected to the ADC is determined by the ADCHS bit in the ADCON1 register.

If the channel is changed, a certain delay is required before the next conversion begins. The ADC delay time is shown in the table below:

| Delay time | Operating voltage |
|---|---|
| 500ns | 2.5~4.5V |
| 200ns | 4.5~5.5V |

### 19.2.3 ADC Reference Voltage

By default, the reference voltage for the ADC is provided by the chip's VDD, but it can also be supplied by the internal ADC-LDO. The output voltage of the ADC-LDO is 2.4V.

### 19.2.4 Conversion Clock

The clock source for the conversion can be selected by setting the ADCKS bit in the ADCON1 register via software.

The time taken to complete one bit of conversion is defined as $T_{ADCK}$. A complete 12-bit conversion requires 18.5 $T_{ADCK}$ cycles (the time during which ADGO remains high to complete a conversion). The corresponding $T_{ADCK}$ specification must be met in order to obtain accurate conversion results. The table below provides examples of how to properly select the ADC clock:

| Fsys | $F_{ADCK}$ ($T_A$=25°C) | |
|---|---|---|
| | $V_{REF}$=$V_{REFE}$=AVDD (AVDD=VDD) | $V_{REF}$=$V_{REFI}$=2.4V |
| 8MHz | Fsys/4 | Fsys/16 |
| 16MHz | Fsys/8 | Fsys/32 |
| 24MHz | Fsys/16 | Fsys/64 |
| 48MHz | Fsys/32 | Fsys/128 |

Note: Any change in the system clock frequency will change the frequency of the ADC clock, which may negatively affect the ADC conversion results.

## 19.3 ADC Working Principle

### 19.3.1 Starting Conversion

To enable the ADC module, you must first set the ADEN bit in the ADCON1 register to 1, and then set the ADGO bit in the ADCON0 register to 1 to start the analog-to-digital conversion (ADGO cannot be set to 1 when ADEN is 0).

### 19.3.2 Completion of Conversion

When the conversion is complete, the ADC module will:

- Clear the ADGO bit.
- Set the ADCIF flag bit to 1.
- Update the ADRESH:ADRESL register with the new conversion result.

### 19.3.3 Termination of Conversion

If the conversion needs to be terminated before completion, the incomplete conversion result will not be updated to the ADRESH:ADRESL register. Therefore, the ADRESH:ADRESL register will retain the result from the previous conversion.

Note: A device reset will force all registers to be reset. As a result, a reset will disable the ADC module and terminate any pending conversions.

### 19.3.4 A/D Conversion Steps

The configuration steps for performing an analog-to-digital conversion using the ADC are as follows:

1) Port configuration:
   - Disable the pin output driver (refer to the PxTRIS register).
   - Configure the pin as an analog input pin.
   - Disable the pin output driver (refer to the PxTRIS register).
   - Configure the pin as an analog input pin.
2) Configure the ADC interrupt (pptional):
   - Clear the ADC interrupt flag
   - Enable the ADC interrupt
   - Enable the peripheral interrupt
   - Enable global interrupts
3) Configure the ADC module:
   - Select the ADC conversion clock
   - Select the ADC input channel
   - Enable the ADC module
4) Wait for the required sampling time
5) Set the ADGO bit to 1 to start the conversion
6) Wait for the ADC conversion to complete by one of the following methods:
   - Poll the ADGO bit
   - Wait for the ADC interrupt (if interrupts are enabled)
7) Read the ADC result
8) Clear the ADC interrupt flag (if interrupts are enabled, this step is required)

Note: If the user attempts to resume sequential code execution after waking the device from sleep mode, global interrupts must be disabled.

### 19.3.5 Entering Sleep During Conversion

When the system enters sleep mode, it is recommended to wait for the ongoing ADC conversion to complete before entering sleep.

If the system enters sleep mode during an ongoing ADC conversion, the conversion will be terminated. After waking up, a new conversion must be initiated.

## 19.4 Relevant Registers

There are four main registers related to the ADC, which are:

- AD Control Registers: ADCON0, ADCON1;
- AD Result Data Registers: ADRESH/L;

### 19.4.1 AD Control Register (ADCON0)

| 0xDF | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| ADCON0 | -- | ADCKS2 | ADCKS1 | ADCKS0 | -- | -- | ADGO | -- |
| R/W | W | R/W | R/W | R/W | W | W | R/W | W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 -- Reserved, set to 0.

Bit6~ Bit4 ADCKS<2:0>: ADC conversion clock select bit; (When switching the ADC clock, ADC enable must be turned off first)

000= Fsys/2
001= Fsys/4
010= Fsys/8
011= Fsys/16
100= Fsys/32
101= Fsys/64
110= Fsys/128
111= Fsys/256

Bit3~Bit2 -- Reserved, set to 0.

Bit1 ADGO: ADC conversion start bit (When this bit is set to 1, ADEN must be 1; otherwise, the operation is invalid);

1= Write: Start the ADC conversion.
Read: ADC is in the process of conversion.
0= Write: Invalid.
Read: ADC is idle or conversion is complete.
During the ADC conversion (ADGO = 1), any software trigger signal will be ignored.

Bit0 -- Reserved, set to 0.

### 19.4.2 AD Control Register (ADCON1)

| 0xDE | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| ADCON1 | ADEN | ADVSEL | ADCHS5 | ADCHS4 | ADCHS3 | ADCHS2 | ADCHS1 | ADCHS0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Caution: When the ADC selects VDD as the reference voltage (ADCON1[6] = 0), the ADCLDO enable (ADCLDO[7] = 0) must be turned off in advance, and it must remain off while ADC is using VDD as the reference voltage.

When the ADC selects ADCLDO as the reference voltage (ADCON1[6] = 1), the ADC reference voltage must first be set to ADCLDO (ADCON1[6] = 1), and then ADCLDO must be enabled (ADCLDO[7] = 1).

Bit7 ADEN: ADC enable bit
1= Enable ADC
0= Disable ADC, no operating current consumption.

Bit6 ADVSEL ADC reference voltage selection
0: Select VDDREF (LDOAD module must be disabled)
1: Select ADCLDO

Bit5~Bit0 ADCHS<5:0>: Analog channel selection bit

| | | | |
|---|---|---|---|
| 000000= | AN0(P00) | 010101= | AN21(P25) |
| 000001= | AN1(P01) | 010110= | AN22(P26) |
| 000010= | AN2(P02) | 010111= | AN23(P27) |
| 000011= | AN3(P03) | 011000-100110 | Reserved, selection is disabled |
| 000100= | AN4(P04) | 100111= | AN39(P50) |
| 000101= | AN5(P05) | 101000= | AN40(P51) |
| 000110= | AN6(P06) | 101001= | AN41(P52) |
| 000111= | AN7(P07) | 101010= | AN42(P53) |
| 001000= | AN8(P10) | 101011= | AN43(P54) |
| 001001= | AN9(P11) | 101100= | AN44(P55) |
| 001010= | AN10(P12) | 101101= | Reserved, selection is disabled |
| 001011= | AN11(P13) | 101110= | Reserved, selection is disabled |
| 001100= | AN12(P14) | 101111= | Reserved, selection is disabled |
| 001101= | AN13(P15) | 110000= | AN48(VDD) |
| 001110= | AN14(P16) | 110001= | AN49(GND) |
| 001111= | AN15(P17) | 110010= | Reserved, selection is disabled |
| 010000= | AN16(P20) | 110011= | Reserved, selection is disabled |
| 010001= | AN17(P21) | 110100= | AN52(vbg0.8) |
| 010010= | AN18(P22) | 110101= | Reserved, selection is disabled |
| 010011= | AN19(P23) | 110110= | Reserved, selection is disabled |
| 010100= | AN20(P24) | 110111= | Reserved, selection is disabled |
| | | Other | Reserved, selection is disabled |

### 19.4.3 AD Data Register High Bits (ADRESH)

| 0xDD | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| ADRESH | -- | -- | -- | -- | ADRES11 | ADRES10 | ADRES9 | ADRES8 |
| R/W | -- | -- | -- | -- | R | R | R | R |
| Reset value | -- | -- | -- | -- | X | X | X | X |

Bit7~Bit4 Uused.

Bit3~Bit0 ADRES<11:8>: ADC result register bit
The 11th to 8th bits of the 12-bit conversion result.

### 19.4.4 AD Data Register Low Bits (ADRESL)

| 0xDC | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| ADRESL | ADRES7 | ADRES6 | ADRES5 | ADRES4 | ADRES3 | ADRES2 | ADRES1 | ADRES0 |
| R/W | R | R | R | R | R | R | R | R |
| Reset value | X | X | X | X | X | X | X | X |

Bit7~Bit0 ADRES<7:0>: ADC result register bit
The 7th to 0th bits of the 12-bit conversion result.

### 19.4.5 AD Reference Voltage Control Register (ADCLDO)

| F693H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| ADCLDO | LDOEN | -- | -- | -- | -- | -- | -- | -- |
| R/W | R/W | -- | -- | -- | -- | -- | -- | -- |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 LDOEN ADC_LDO enable
1= LDO enabled: Reference voltage is 2.4V.
0= LDO disabled: Reference voltage is the chip's supply voltage.
Bit6~Bit0 -- Reserved, set to 0.

Caution: When the ADC selects VDD as the reference voltage (ADCON1[6] = 0), the ADCLDO enable (ADCLDO[7] = 0) must be turned off in advance, and it must remain off while the ADC uses VDD as the reference voltage.

When the ADC selects ADCLDO as the reference voltage (ADCON1[6] = 1), the ADC reference voltage must first be set to ADCLDO (ADCON1[6] = 1), and then ADCLDO must be enabled (ADCLDO[7] = 1).

## 19.5 ADC Interrupt

The ADC module allows an interrupt to be generated after the analog-to-digital conversion is completed. The ADC interrupt enable bit is the ADCIE bit in the EIE2 register, and the ADC interrupt flag bit is the ADCIF bit in the EIF2 register. The ADCIF bit must be cleared by software. After each conversion, the ADCIF bit will be set to 1, regardless of whether the ADC interrupt is enabled or not. The interrupt enable and priority for the ADC can be configured through the relevant register bits as follows.

### 19.5.1 Interrupt Mask Register (EIE2)

| 0xAA | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| EIE2 | SPIIE | I2CIE | -- | ADCIE | PWMIE | ET5 | -- | -- |
| R/W | R/W | R/W | -- | R/W | R/W | R/W | R | R |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 SPIIE: SPI interrupt enable bit
1= Enable SPI interrupt
0= Disable SPI interrupt
Bit6 I2CIE: $I^2C$ interrupt enable bit
1= Enable $I^2C$ interrupt
0= Disable $I^2C$ interrupt
Bit5 -- Reserved, set to 0.
Bit4 ADCIE: ADC interrupt enable bit
1= Enable ADC interrupt
0= Disable ADC interrupt
Bit3 PWMIE: PWM global interrupt enable bit
1= Enable PWM all interrupts
0= Disable PWM all interrupts
Bit2 ET5: Timer5 interrupt enable bit
1= Enable Timer5 interrupt
0= Disable Timer5 interrupt
Bit1~Bit0 -- Reserved, set to 0.

### 19.5.2 Interrupt Priority Control Register (EIP2)

| 0xBA | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| EIP2 | PSPI | PI2C | -- | PADC | PPWM | PT5 | -- | -- |
| R/W | R/W | R/W | -- | R/W | R/W | R/W | R | R |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 PSPI: SPI interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit6 PI2C: $I^2C$ interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit5 -- Reserved, set to 0.
Bit4 PADC: ADC interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit3 PPWM: PWM interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit2 PT5: TIMER5 interrupt priority control bit
1= High-level interrupt
0= Low-level interrupt
Bit1~Bit0 -- Reserved, set to 0.

### 19.5.3 Peripheral Interrupt Flag Register (EIF2)

| 0xB2 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| EIF2 | SPIIF | I2CIF | -- | ADCIF | PWMIF | TF5 | -- | -- |
| R/W | R | R | R | R/W | R | R/W | R | R |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 SPIIF: SPI global interrupt flag (read-only)
1= SPI generates an interrupt, (this bit is automatically cleared when the specific interrupt flag bit is cleared)
0= No SPI interrupt generated
Bit6 I2CIF: $I^2C$ global interrupt flag (read-only)
1= $I^2C$ generates an interrupt, (this bit is automatically cleared when the specific interrupt flag bit is cleared)
0= No $I^2C$ interrupt generated
Bit5 -- Reserved, set to 0.
Bit4 ADCIF: ADC interrupt flag bit
1= ADC conversion is completed, need to be cleared by software.
0= ADC conversion is not completed.
Bit3 PWMIF: PWM global interrupt flag (read-only)
1= PWM generates an interrupt, (this bit is automatically cleared when the specific interrupt flag bit is cleared)
0= No PWM interrupt generated
Bit2 TF5: Timer5 overflow interrupt flag bit
1= Timer5 overflow interrupt flag bit
0= Timer5 timer no overflow
Bit1~Bit0 -- Reserved, set to 0.

# 20. Cyclic Redundancy Check Unit (CRC)

## 20.1 Overview

To ensure the safety during operation, the IEC61508 standard requires that data be verified even while the CPU is running. This general-purpose CRC module can perform CRC calculations as a peripheral function while the CPU is operating. The general-purpose CRC module performs CRC checks on data specified by the program, and it is not limited to the code flash area but can be used for various types of checks.

The CRC generation polynomial uses CRC16-CCITT's $X^{16}+X^{12}+X^5+1$.

## 20.2 Relevant Registers

### 20.2.1 CRC Data Input Register (CRCIN)

| F708H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| CRCIN | CRCIN7 | CRCIN6 | CRCIN5 | CRCIN4 | CRCIN3 | CRCIN2 | CRCIN1 | CRCIN0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 CRCIN<7:0> The 8-bit data input required for CRC computation.

### 20.2.2 CRC Calculation Result Low 8-Bit Data Register (CRCDL)

| F709H | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| CRCDL | CRCD7 | CRCD6 | CRCD5 | CRCD4 | CRCD3 | CRCD2 | CRCD1 | CRCD0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 CRCD<7:0> The lower 8-bit data of the CRC calculation result

### 20.2.3 CRC Calculation Result High 8-Bit Data Register (CRCDH)

| F70AH | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| CRCDH | CRCD15 | CRCD14 | CRCD13 | CRCD12 | CRCD11 | CRCD10 | CRCD9 | CRCD8 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 CRCD<15:8> The higher 8-bit data of the CRC calculation result.

## 20.3 Function Description

After writing to the CRCIN register, the CRC calculation result will be saved to the CRCDL/CRCDH registers after one system clock cycle. If necessary, the previous calculation data must be read before writing new values to avoid overwriting by the new calculation result.

For example: sending the data “12345678H”, write values to the CRCIN register in the order of “12H”, “34H”, “56H”, and “78H”. After writing, read from the CRCDL/CRCDH registers, which gives CRCDL=0xF0 and CRCDH=0x67. This means that the CRC calculation result for the data “12345678H” is 0x67F0 based on the bit sequence. The register operations are as follows:

```
CRCIN = 0x12;          // Send the first byte
CRCIN = 0x34;          // Send the second byte
CRCIN = 0x56;          // Send the third byte
CRCIN = 0x78;          // Send the fourth byte
resl = CRCDL;          // Read the lower 8 bits of the CRC result into the variable resl
resh = CRCDH;          // Read the higher 8 bits of the CRC result into the variab resh
```

# 21. Touch Module

The touch module is an integrated circuit designed to implement a human touch interface, replacing mechanical tactile switches. It provides a waterproof, dustproof, sealed, isolated, and durable aesthetic for the operating interface.

Technical specifications:

- Up to 30 touch keys can be selected.
- No external touch capacitors required.

## 21.1 Touch Module Usage Precautions

- The ground line of the touch key detection part should be connected to a separate ground, with another point connecting to the common ground of the whole system.
- Avoid placing the mainboard with high voltage, high current, and high-frequency operations directly above or below the touch circuit board. If unavoidable, ensure it is positioned as far away as possible from high-voltage, high-current areas, or add shielding on the mainboard.
- Keep the wiring between the sensing plate and touch chip as short and thin as possible.
- Do not allow the wiring between the sensing plate and touch chip to cross over strong interference or high-frequency signal lines.

# 22. Memory Management Controller (MMC/FMC)

## 22.1 Overview

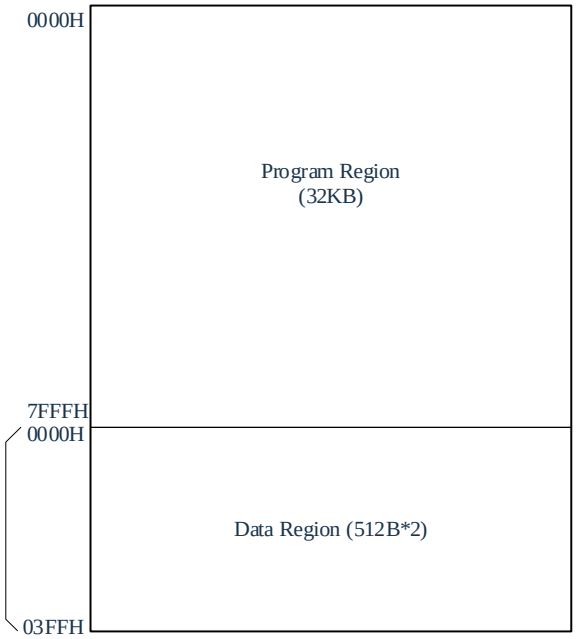
The memory includes program memory (APROM) and non-volatile data memory (Data). The program memory space has a maximum size of 32KB, while the data space is 1KB.

APROM consists of 64 sectors, with each sector containing 512 bytes.

Data consists of 2 sectors, with each sector containing 512 bytes.

Information area has a total of 32 bytes.

UID consists of 16 bytes.




The FLASH memory can be accessed and manipulated to achieve IAP functionality through specific Special Function Registers (SFR). The SFRs used to access the FLASH space are as follows:

- MLOCK
- MSTATUS
- MDATA
- MADRL
- MADRH
- MREGION
- MMODE

The MLOCK register is used to enable memory operations. The MSTATUS register indicates the status of FLASH operations and controls the start of the operation. The MDATA register holds the 8-bit data to be written. The MADRL and MADRH registers store the address of the MDATA unit being accessed. The MREGION register is used for memory region selection. The MMODE register is for selecting the memory operation mode.

Through the memory module interface, operations such as read, write, erase, and CRC verification can be performed. The memory allows byte-level read/write operations, with the write time controlled by an on-chip timer. Before writing new data, it is necessary to ensure that the data in the address has been erased. The write and erase voltages are generated by an on-chip charge pump, and this pump operates within the device’s voltage range for byte operations.

Flash memory only supports sector erasure, not byte erasure. Before modifying data at a particular address, it is recommended to first save other data, erase the current sector, and then perform the write operation.

## 22.2 Relevant Registers

### 22.2.1 Memory Protection Lock Register (MLOCK)

| 0xFF | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| MLOCK | MLOCK7 | MOCK6 | MLOCK5 | MLOCK4 | MLOCK3 | MLOCK2 | MLOCK1 | MLOCK0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 MLOCK<7:0>: Memory operation enable bit
AAH= Enable memory-relate W/E/R/CRC operations.
Other= Disable memory-relate W/E/R/CRC operations.

### 22.2.2 Memory Status Register (MSTATUS)

| 0xFE | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| MSTATUS | MLOCKF | ERROR | - | RDINC | START | - | - | - |
| R/W | R | R/W | R | R/W | R/W | R | R | R |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 MLOCKF: Memory operation enable status flag bit (When MLOCK is enabled, this bit is 1. Otherwise, it is 0.)
1= Enable, FLASH can be operated by registers.
0= Disable, FLASH operations are not allowed.
Bit6 ERROR Error flag (Write 0 to clear)
1= When programming begins, if the data at the programming address is not “FFH” (not erased), the write operation is immediately terminated
(Only single-byte write operations are supported.)
Bit5 -- Reserved
Bit4 RDINC: Single-byte read mode address auto incremen enable bit (valid only after a single-byte read operation is completed)
1= ADDRH/ADDRL will automatically increment by 1
0= ADDRH/ADDRL will not change automatically
Bit3 START: Start operation control bit
1= Initiate memory W/E/R/CRC operation (can be automatically cleared by hardware after operation).
0= Write: terminates or does not start the program memory W/E/CRC operation.
Bit2~Bit0 -- Reserved, set to 0.

### 22.2.3 Memory Data Register (MDATA)

| 0xFB | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| MDATA | MDATA7 | MDATA6 | MDATA5 | MDATA4 | MDATA3 | MDATA2 | MDATA1 | MDATA0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 MDATA<7:0>: Single-byte write: Data written to program memory.

### 22.2.4 Memory Address Register (MADRL)

| 0xFC | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| MADRL | MADRL7 | MADRL6 | MADRL5 | MADRL4 | MADRL3 | MADRL2 | MADRL1 | MADRL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7~Bit0 MADRL<7:0>: Single-byte write: Specifies the lower 8 bits of the memory address where the write operation will occur.
Multi-byte write: Bit 7: The 7th bit of the start address for multi-byte write operations. Bits 6-0: The lower 7 bits of the end address for multi-byte write operations.
CRC Check: The lower address of the end (valid only in CRC check mode), with bit 7 being valid and bits 6-0 being ignored.

### 22.2.5 Memory Address Register (MADRH)

| 0xFD | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| MADRH | -- | MADRH6 | MADRH5 | MADRH4 | MADRH3 | MADRH2 | MADRH1 | MADRH0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit7 -- Reserved
Bit6~Bit0 MADRH<6:0>: Single-byte write: Specifies the higher 8 bits of the memory address where the write operation will occur.
Multi-byte write: The higher bits of the multi-byte write address.
The higher 8 bits of the end address for memory CRC operation (valid only in CRC check mode).

APROM address range: 0x0000-0x7FFF

DATA address range: 0x000-0x3FFH

Note: Operations outside the valid address range will be ignored and have no effect.

### 22.2.6 Memory Region Control Register (MREGION)

| 0xF9 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| MREGION | REGION7 | REGION6 | REGION5 | REGION4 | REGION3 | REGION2 | REGION1 | REGION0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |

Bit7~Bit0 REGION<7:0> Flash region select bit

55H= SelectAPROM

AAH= SelectDATA

69H= Select information (Read-only; write and erase operations are prohibite)

66H= Select UID (Read-only; write and erase operations are prohibited)

Other= Invalid, all ongoing operations will be exited.

### 22.2.7 Memory Mode Control Register (MMODE)

| 0xFA | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| MMODE | MODE7 | MODE6 | MODE5 | MODE4 | MODE3 | MODE2 | MODE1 | MODE0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset value | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |

Bit7~Bit0 MODE<7:0> Flash mode select bi

55H= Erase operation (sector erase)

AAH= Single-byte write operation (single byte)

A9H= Multi-byte write operation (1~128 bytes), writes the contents of the last 128 addresses of XRAM to multiple addresses

6AH= Multi-byte read operation (1~128 bytes), writes multiple addresses to the last 128 addresses of XRAM

69H= Single-byte read operation (can be configured to automatically increment {MADRH, MADRL} by 1 after reading is completed)

96H= CRC check

Other= Reserved, not allowed to use.

## 22.3 Function Description

During the read/write/erase operations on the FLASH memory, the CPU is in a suspended state. Once the operation is completed, the CPU resumes executing instructions.

Note: For FLASH read/write/erase/CRC operations, it is required that MLOCK, MREGION, and MMODE be properly configured in order to initiate the operation. If any of these registers (MLOCK, MREGION, or MMODE) are invalid during the operation, the operation will be prohibited.

Multi-byte operations on FLASH require buffering support. In the multi-byte operation mode, the last 128 addresses of XRAM (0x0780~0x7FF) are used as the cache. To avoid read/write conflicts, it is recommended not to define these addresses as variable areas. In this mode, to speed up FMC operations, both write and read operations are done through the cache. If multi-byte operations are not used, these XRAM cache addresses (0x0780~0x7FF) can be used as regular variable areas.

### 22.3.1 Read Operation (Single-Byte Read)

The steps for a single-byte memory read operation are as follows:

1) Enable access to the memory register:
   MLOCK=0xAA;
2) Set the memory region to access:
   Configure the region through the MREGION register.
3) Set the address of the memory to access:
   Set the address through the MADRL/MADRH registers.
4) Set the single-byte read command:
   MMODE=0x69.
5) Start the write operation:
   Set MSTATUS[3] to 1.
6) Wait for 6 NOP instructions and then check if the write operation is complete:
   After the write operation completes, MSTATUS[3] will be cleared to 0 by hardware.
7) Disable memory access operation:
   MLOCK=0x00.

After the operation, the data from the corresponding address of FLASH will be loaded into MDATA, and the content of {MADRH, MADRL} can be configured to increment by 1 after the operation is completed.

### 22.3.2 Multi-Byte Read Operation (128 Bytes, Read to Cache)

The steps for a multi-byte read operation are as follows:

1) Enable access to the memory register:

MLOCK=0xAA;

2) Set the region of the memory to be accessed:

Set the region through the MREGION register.

3) Set the starting and ending addresses of the memory to be accessed:

Set the address through MADRL/MADRH registers. For example: 0x200 (starting address) + 0x7F (offset, ending address) = 0x27F.

The lower 7 bits of the starting address are fixed at 0, and MADRL[6:0] indicates the ending read address (i.e., the byte count minus 1).

4) Set the multi-byte read command:

MMODE=0x6A.

5) Start the write operation:

Set MSTATUS[3] to 1.

6) Wait for 6 NOP instructions, then check if the write operation is completed:

After the write operation completes, MSTATUS[3] will be cleared to 0 by hardware.

7) Disable memory access operation:

MLOCK=0x00.

After this operation, data from Flash starting at address 0x200 will be written to XRAM starting at 0x780 (i.e., 0x200’s data will go to 0x780, 0x201’s data will go to 0x781, and so on, until 0x27F’s data goes to 0x7FF).

### 22.3.3 3 Program Region Read Operation (MOVC Instruction)

The MOVC instruction is only used for reading from the program area.

### 22.3.4 Single-Byte Write Operation

The steps for a single-byte memory write operation are as follows:

1) Enable access to the memory register:

   MLOCK=0xAA;

2) Set the region of the memory to be accessed:

   Set the programming area through the MREGION register.

3) Set the memory address to be written to:

   Set the address through MADRL/MADRH registers. For example: 0x200.

4) Set the data to be written:

   Write the data into MDATA. For example: 0x12.

5) Set the single-byte write command:

   MMODE=0xAA.

6) Start the write operation:

   Set MSTATUS[3] to 1.

7) Wait for 6 NOP instructions, then check if the write operation is completed:

   After the write operation completes, MSTATUS[3] will be cleared to 0 by hardware.

8) Disable memory access operation:

   MLOCK=0x00.

After this operation, data 0x12 will be written to address 0x200.

### 22.3.5 Multi-Byte Write Operation (1 to 128 Bytes, Written to Cache First)

The steps for a multi-byte write operation are as follows:

1) First, place the 1 to 128 bytes of data to be written into the last 128 addresses of XRAM:
   For example, place 0x80 at 0x0780, 0x81 at 0x0781, and so on, up to 0xFF at 0x07FF. These 128 bytes act like a cache.
2) Enable access to the memory register:
   MLOCK=0xAA;
3) Set the region of the memory to be accessed:
   Set the programming area using the MREGION register.
4) Set the starting and ending memory addresses to be accessed:
   Set the addresses using MADRL/MADRH registers. For example, 0x200 (starting address) + 0x7F (ending offset) = 0x27F.
   The lower 7 bits of the starting address are fixed as 0, and MADRL[6:0] indicates the ending programming address (i.e., the byte count minus 1).
5) Set the multi-byte write command:
   MMODE=0xA9.
6) Start the write operation:
   Set MSTATUS[3] to 1.
   (If there are bytes in the cache that are empty (i.e., 0xFF), all bytes except for the first one will skip the actual programming operation and proceed to the next byte.)
7) Wait for 6 NOP instructions, then check if the write operation has finished:
   After the write operation is completed, MSTATUS[3] will be cleared to 0 by hardware.
8) Disable memory access operation:
   MLOCK=0x00.

After this operation, data from 0x0780 to 0x07FF in XRAM will be written to the Flash memory, starting at address 0x200. Specifically, 0x200 will hold 0x80, 0x201 will hold 0x81, and so on, until 0x27F holds 0x7F.

Cautions: a. If all bytes are non-0xFF data, the programming time for each byte is about 25μs. For each additional byte, the programming time increases by about 8μs. The total time for programming 128 bytes is approximately 1.05ms.

b. If all bytes are 0xFF, the programming time for the first byte is 25μs, while for subsequent bytes, the programming time is approximately 1μs per byte. The total time for programming 128 bytes in this case is approximately 140μs.

### 22.3.6 Erasure Operation (Sector Erase)

The steps for an erasure operation are as follows:

1) Enable memory access register:

   MLOCK=0xAA;

2) Set the address to be erased:

   Use the MADRL/MADRH registers to set the address of the sector to be erased.

3) Set the memory region corresponding to the address:

   Use the MREGION register to set the region to be erased.

4) Set the erasure command:

   MMODE=0x55.

5) Start the erasure operation:

   Set MSTATUS[3] = 1.

6) Wait for 6 NOP instructions and check if the erasure is complete:

   After the erasure operation is completed, MSTATUS[3] will be cleared to 0 by hardware.

7) Disable memory access operation:

   MLOCK=0x00.

### 22.3.7 Memory CRC Check

The CRC check for memory requires the use of the general-purpose CRC module, while the FMC controller generates the start, control, and end signals. The computation is done by the general-purpose CRC module.

The CRC check command is set by the MMODE register. The check region is set via the MREGION register, with the starting address fixed at 0. The end address can be freely configured via the MADRL/MADRH registers (the lower 7 bits of MADRL are ignored). The minimum unit block is configured as 128 bytes. The result of the CRC operation is stored in the CRCDL/CRCDH registers. During the CRC check process, the CPU stops working, and it will resume once the CRC calculation is complete. The CRC check is performed on a byte-by-byte basis, from address 0 to the end address. The check speed is 2 clock cycles ($2*T_{SYS}$) per byte.

The steps for performing the CRC check are as follows:

1) Clear previous CRC check result:
   CRCIN=0x00; CRCDL/CRCDH=0x00.
2) Enable program memory access:
   MLOCK=0xAA;
3) Set the end address for program CRC check:
   Set the end address using MADRL/MADRH.
4) Select the CRC check region:
   Set the check region using the MREGION register.
5) Select the CRC check command:
   MMODE=0x96.
6) Start the CRC check:
   Set MSTATUS[3] = 1.
7) Wait for 6 NOP instructions and check if the CRC check is complete:
   After completion, MSTATUS[3] will be cleared to 0 by hardware.
8) Read the program CRC check result:
   CRCDL: Lower 8 bits of the CRC result.
   CRCDH: Upper 8 bits of the CRC result.
9) Disable memory access:
   MLOCK=0x00.

### 22.3.8 Information

The information stored in this section cannot be modified by the program but can be updated via a programmer. It is used to store information such as anti-tampering codes, WiFi serial numbers, rolling codes, etc. This information is read-only.

The address for this information is from 0x480 to 0x49F, corresponding to 32 bytes.

### 22.3.9 Unique ID (UID)

The Unique ID (UID) is set at the factory and cannot be modified by the programmer or the user.

Each chip has a unique 128-bit ID, known as the UID (Unique identification). It is set during production, and users are not allowed to change it.

The UID is stored at the address range 0x680 to 0x68F, which corresponds to 16 bytes.

# 23. User Configuration

The system configuration register (CONFIG) is a set of FLASH options that define the initial conditions of the MCU. These options cannot be accessed or manipulated by the program. Below are the available configuration options:

1. WDT (Watch Dog Timer Mode Selection)
   - ◆ Enable — Forces WDT to be enabled.
   - ◆ Software control (default) — WDT is controlled by the WDKEY register
2. PROTECT
   - ◆ Disable (default) — FLASH code is not encrypted
   - ◆ Enable — FLASH code is encrypted, and reading the code is 00H.
3. FLASH_DATA_PROTECT
   - ◆ Disable (default) — FLASH data region is not encrypted.
   - ◆ Enable — FLASH data region is encrypted, and reading the values from the programming emulator is 00H.
4. LVD (Low Voltage Detection Enable)
   - ◆ Enable (default)
   - ◆ Disable
5. LVD Reset Detection Voltage Configuration
   - ◆ 2.5V (default)
   - ◆ 2.7V
   - ◆ 3.0V
   - ◆ 3.3V
   - ◆ 3.7V
   - ◆ 4.0V
   - ◆ 4.3V
6. LVD Mode Selection
   - ◆ Interrupt mode — Detection voltage settings: see register LVDS[2:0]
   - ◆ Reset mode (default) — Detection voltage settings: see user-configured LVD reset detection voltage settings
7. DEBUGEN (Debug Mode Enable)
   - ◆ Disable (default) — Debug mode is disabled , DSCK and DSDA pins are used as general IO ports.
   - ◆ Enable — Debug mode is enbled , DSCK and DSDA pins are configured as debug ports, and other functions on these pins are disabled.
8. DBGSEL (Debug Port Selection, requires debug mode to be enabled)
   - ◆ TWI (Two-wire debug)
   - ◆ SWI (Single-wire debug)
9. OSC (Oscillator Mode)
   - ◆ HSI (default) — 48MHz
   - ◆ HSE — 16MHz/8MHz (CONFIG selects HSE, P51, P52 disables configuration of other functions)
   - ◆ LSI(32KHz) — 32KHz
10. SYS_PRESCALE (System Clock Prescaler Selection)
    - ◆ Fosc/1 (default)
    - ◆ Fosc/2
    - ◆ Fosc/3
    - ◆ Fosc/4
    - ◆ Fosc/6
    - ◆ Fosc/8
    - ◆ Fosc/16
    - ◆ Fosc/32

11. EXT_RESET (External Reset Configuration)
    - Disable (default) — External reset is disabled
    - Enable(OPEN PULLUP) — External reset is enabled, and internal pull-up resistors for the reset pin are enabled.
12. WAKE UP_WAIT TIME (Sleep Mode Wake-up Oscillator Stabilization Time, default is 1.0s)
    - 50us
    - 100us
    - 500us
    - 1ms
    - 5ms
    - 10ms
    - 500ms
    - 1.0s (default)
13. WTSTEN (Program Memory Wait Time Configuration)
    - Wait time is determined by the configuration of the WTST register.
    - If Fsys = 48MHz, wait time is 2T; if Fsys = other, wait time is 1T (default)
14. WTST Program Memory Wait Time Selection
    - 1*Tsys (When Fsys=48M, 1T is not allowed)
    - 2*Tsys
15. WRITE_PROTECT Program Partition Protection (Configurable Protection Ranges, All Default Ranges Are Not Protected)
    - 0000H-07FFH (Protect/Not Protect)
    - 0800H-0FFFH (Protect/Not Protect)
    - 1000H-17FFH (Protect/Not Protect)
    - 1800H-1FFFH (Protect/Not Protect)
    - 2000H-27FFH (Protect/Not Protect)
    - 2800H-2FFFH (Protect/Not Protect)
    - 3000H-37FFH (Protect/Not Protect)
    - 3800H-3FFFH (Protect/Not Protect)
    - 4000H-47FFH (Protect/Not Protect)
    - 4800H-4FFFH (Protect/Not Protect)
    - 5000H-57FFH (Protect/Not Protect)
    - 5800H-5FFFH (Protect/Not Protect)
    - 6000H-67FFH (Protect/Not Protect)
    - 6800H-6FFFH (Protect/Not Protect)
    - 7000H-77FFH (Protect/Not Protect)
    - 7800H-7FFFH (Protect/Not Protect)
16. WRITE_PROTECTDATA Partition Protection (Configurable Protection Ranges, All Default Ranges Are Not Protected)
    - 0000H-01FFH (Protect/Not Protect)
    - 0200H-03FFH (Protect/Not Protect)

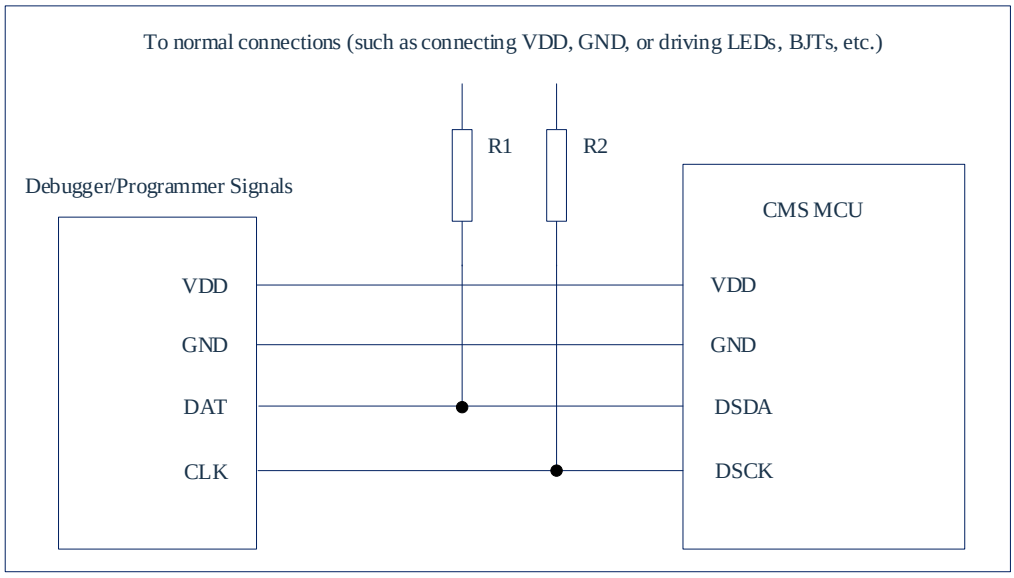
# 24. Online Programming and Debugging

## 24.1 Online Programming Mode

The chip can be serially programmed in the final application circuit. Programming can be easily completed using the following 4 lines:

- Power Line
- Ground Line
- Data Line
- Clock Line

Online serial programming allows users to manufacture circuit boards with unprogrammed devices and program the chip just before product delivery. This enables the latest version of firmware or custom firmware to be written to the chip. A typical online serial programming connection method is shown in the figure below:




In the diagram above, R1 and R2 are electrical isolation components, commonly replaced by resistors with the following values: R1 ≥ 4.7K, R2 ≥ 4.7K.

Note that during programming and debugging, DSDA should not be connected to a pull-down resistor. If the actual circuit requires a pull-down resistor, it is recommended to use a jumper structure to disconnect the pull-down resistor during programming/debugging and reconnect it after completion.

## 24.2 Online Debugging Mode

The chip supports a 2-wire (DSCK/DSDA) or a single-wire (SWE) online debugging function. If using the online debugging function, the DEBUG setting in the system configuration register must be set to ENABLE. When using the debugging mode, the following points should be noted:

- In debugging mode, DSCK/DSDA serves as a dedicated debugging interface and must not be repurposed for touch functionality. It cannot perform its GPIO or alternate functions.
- In debugging mode, if the system enters sleep or idle mode, the system power and oscillator will not stop working. In this state, the simulated sleep/wake-up function can be tested. If power consumption is a concern, it is recommended to disable the debugging function before testing the actual sleep current of the chip.
- In debugging mode and when paused, other peripheral functions continue to operate, but the WDT, Timer0/1/2/5 counters will stop. However, if Timer1 is used as the baud rate generator for UART0/1, Timer1 will continue running during the pause state. Other peripherals that continue to run during the pause state may generate interrupts, so attention is needed during debugging.
- In debugging mode, it is recommended not to use WDT reset and software reset functions, as the chip may lose connection with the debugger during reset.

# 25. Instructions

Assembly instructions include 5 categories: Arithmetic operations, logic operations, data transfer operations, Boolean operations and program branch instructions, all of these instructions are compatible with standard 8051.

## 25.1 Symbol Description

| Symbol | Description |
|---|---|
| Rn | Working register R0-R7 |
| Direct | Unit address of internal data memory RAM (00H-FFH) or address in special function register SFR |
| @Ri | Indirect addressing register (@R0 or @R1) |
| #data | 8-bit binary constants |
| #data16 | 16-bit binary constant in the instruction |
| Bit | Bit address in internal data memory RAM or special function register SFR |
| Addr16 | 16-bit address, address range 0-64KB |
| Addr11 | 11-bit address, address range 0-2KB |
| Rel | Relative address |
| A | Accumulator |

## 25.2 Instruction Set

| Mnemonic symbol | | Description |
|---|---|---|
| **Arithmetic operation** | | |
| ADD | A,Rn | Add register to accumulator. |
| ADD | A,direct | Add directly addressed data to accumulator. |
| ADD | A,@Ri | Add indirectly addressed data to accumulator. |
| ADD | A,#data | Add immediate data to accumulator. |
| ADDC | A,Rn | Add register to accumulator with carry. |
| ADDC | A,direct | Add directly addressed data to accumulator with carry. |
| ADDC | A,@Ri | Add indirectly addressed data to accumulator with carry. |
| ADDC | A,#data | Add immediate data to accumulator with carry. |
| SUBB | A,Rn | Subtract register from accumulator with borrow. |
| SUBB | A,direct | Subtract directly addressed data from accumulator with borrow. |
| SUBB | A,@Ri | Subtract indirectly addressed data from accumulator with borrow. |
| SUBB | A,#data | Subtract immediate data from accumulator with borrow. |
| INC | A | Increment accumulator. |
| INC | Rn | Increment register. |
| INC | direct | Increment directly addressed location. |
| INC | @Ri | Increment indirectly addressed location. |
| INC | DPTR | Increment data pointer. |
| DEC | A | Decrement accumulator. |
| DEC | Rn | Decrement register. |
| DEC | direct | Decrement directly addressed location. |
| DEC | @Ri | Decrement indirectly addressed location. |
| MUL | A,B | Multiply A and B. |
| DIV | A,B | Divide A by B. |
| DA | A | Decimally adjust accumulator. |
| **Logic operation** | | |
| ANL | A,Rn | AND register to accumulator. |
| ANL | A,direct | AND directly addressed data to accumulator. |
| ANL | A,@Ri | AND indirectly addressed data to accumulator. |
| ANL | A,#data | AND immediate data to accumulator. |
| ANL | direct,A | AND accumulator to directly addressed location. |
| ANL | direct,#data | AND immediate data to directly addressed location. |
| ORL | A,Rn | OR register to accumulator. |
| ORL | A, direct | OR directly addressed data to accumulator. |
| ORL | A,@Ri | OR indirectly addressed data to accumulator. |
| ORL | A, #data | OR immediate data to accumulator. |
| ORL | direct,A | OR accumulator to directly addressed location. |
| ORL | direct,#data | OR immediate data to directly addressed location. |
| XRL | A,Rn | Exclusive OR (XOR) register to accumulator. |
| XRL | A,direct | XOR directly addressed data to accumulator. |
| XRL | A,@Ri | XOR indirectly addressed data to accumulator. |
| XRL | A,#data | XOR immediate data to accumulator. |
| XRL | direct,A | XOR accumulator to directly addressed location. |
| XRL | direct,#data | XOR immediate data directly addressed location. |
| CLR | A | Clear accumulator. |
| CPL | A | Complement accumulator. |
| RL | A | Rotate accumulator left. |

| Mnemonic symbol | | Description |
|---|---|---|
| RLC | A | Rotate accumulator left through carry. |
| RR | A | Rotate accumulator right. |
| RRC | A | Rotate accumulator right through carry. |
| SWAP | A | Swap nibbles with in the accumulator. |
| **Data transfer** | | |
| MOV | A,Rn | Move register to accumulator. |
| MOV | A,direct | Move directly addressed data to accumulator. |
| MOV | A,@Ri | Move indirectly addressed data to accumulator. |
| MOV | A,#data | Move immediate data to accumulator. |
| MOV | Rn,A | Move accumulator to register. |
| MOV | Rn,direct | Move directly addressed data to register. |
| MOV | Rn,#data | Move immediate data to register. |
| MOV | direct,A | Move accumulator to direct. |
| MOV | direct,Rn | Move register to direct. |
| MOV | direct1,direct2 | Move directly addressed data to directly addressed location. |
| MOV | direct,@Ri | Move indirectly addressed data to directly addressed location. |
| MOV | direct,#data | Move immediate data to directly addressed location. |
| MOV | @Ri,A | Move accumulator to indirectly addressed location. |
| MOV | @Ri,direct | Move directly addressed data to indirectly addressed location. |
| MOV | @Ri,#data | Move immediate data to indirectly addressed location. |
| MOV | DPTR,#data16 | Load data pointer with a 16-bit immediate. |
| MOVC | A,@A+DPTR | Load accumulator with a code byte relative to DPTR. |
| MOVC | A,@A+PC | Load accumulator with a code byte relative to PC. |
| MOVX | A,@Ri | Move external RAM (8-bit address) to accumulator. |
| MOVX | A,@DPTR | Move external RAM (16-bit address) to accumulator. |
| MOVX | @Ri,A | Move accumulator to external RAM (8-bitaddress). |
| MOVX | @DPTR,A | Move accumulator to external RAM (16-bit address). |
| PUSH | direct | Push directly addressed data on to stack. |
| POP | direct | Pop directly addressed data location from stack. |
| XCH | A,Rn | Exchange register with accumulator. |
| XCH | A, direct | Exchange directly addressed location with accumulator. |
| XCH | A,@Ri | Exchange indirect RAM with accumulator. |
| XCHD | A,@Ri | Exchange low-order nibbles of indirect and accumulator. |
| Boolean operation | | |
| CLR | C | Clear carry flag. |
| CLR | bit | Clear directly addressed bit. |
| SETB | C | Set carry flag. |
| SETB | bit | Set directly addressed bit. |
| CPL | C | Complement carry flag. |
| CPL | bit | Complement directly addressed bit. |
| ANL | C,bit | AND directly addressed bit to carry flag. |
| ANL | C,/bit | AND complement of directly addressed bit to carry. |
| ORL | C,bit | OR directly addressed bit to carry flag. |
| ORL | C,/bit | OR complement of directly addressed bit to carry. |
| MOV | C,bit | Move directly addressed bit to carry flag. |
| MOV | bit,C | Move carry flag to directly addressed bit. |
| **Program branching** | | |
| ACALL | addr11 | Absolute subroutine call |
| LCALL | addr16 | Long subroutine call |

| Mnemonic symbol | | Description |
|---|---|---|
| RET | | Return from subroutine |
| RETI | | Return from interrupt |
| AJMP | addr11 | Absolute jump |
| LJMP | addr16 | Long jump |
| SJMP | rel | Short jump (relative address) |
| JMP | @A+DPTR | Jump indirect relative to the DPTR |
| JZ | rel | Jump if accumulator is zero |
| JNZ | rel | Jump if accumulator is not zero |
| JC | rel | Jump if carry flag is set |
| JNC | rel | Jump if carry flag is not set |
| JB | bit,rel | Jump if directly addressed bit is set |
| JNB | bit,rel | Jump if directly addressed bit is not set |
| JBC | bit,rel | Jump if directly addressed bit is set and clear bit |
| CJNE | A,direct,rel | Compare directly addressed data to accumulator and jump if not equal |
| CJNE | A,#data,rel | Compare immediate data to accumulator and jump if not equal |
| CJNE | Rn,#data,rel | Compare immediate data to register and jump if not equal |
| CJNE | @Ri,#data,rel | Compare immediate to indirect and jump if not equal |
| DJNZ | Rn,rel | Decrement register and jump if not zero |
| DJNZ | direct,rel | Decrement directly addressed location and jump if not zero |
| NOP | | No operation for one cycle |
| **Read-Modify-Write** | | |
| ANL | | Logical AND. (ANL direct, A and ANL direct, #data) |
| ORL | | Logical OR. (ORL direct, A and ORL direct, #data) |
| XRL | | Logical exclusive OR. (XRL direct, A and XRL direct, #data) |
| JBC | | Jump if bit = 1 and clear it. (JBC bit, rel) |
| CPL | | Complement bit. (CPL bit) |
| INC | | Increment. (INC direct) |
| DEC | | Decrement. (DEC direct) |
| DJNZ | | Decrement and jump if not zero. (DJNZ direct, rel) |
| MOV | bit,C | Move carry to bit. (MOV bit, C) |
| CLR | bit | Clear bit. (CLR bit) |
| SETB | bit | Set bit. (SETB bit) |

# 26. Revision History

| Version # | Date | Description of changes |
|---|---|---|
| V0.5.0 | Mar. 2025 | Initial release |
| V0.5.1 | Mar. 2025 | Removed redundant registers from SFR table, adjusted TIMER2 structure block diagram. |
| V0.5.2 | Apr. 2025 | Removed redundant registers from the XSFR table. |
| V0.5.3 | Jul. 2025 | 1) Remove redundant registers.<br>2) Modify the description of Chapter 8.1.2<br>3) Modify the description of Chapter 1.7<br>4) Modify the description of Chapter 6.5.2.4 |