



# **CMS8S588x series**

# **Reference Manual**

**Enhanced flash memory 8-bit 1T 8051 microcontroller**  
**Rev. 1.0.3**

Please be reminded about following CMS's policies on intellectual property

\* Cmsemicon Limited(denoted as 'our company' for later use) has already applied for relative patents and entitled legal rights. Any patents related to CMS's MCU or other products is not authorized to use. Any individual, organization or company which infringes our company's intellectual property rights will be forbidden and stopped by our company through any legal actions, and our company will claim the lost and required for compensation of any damage to the company.

\* The name of Cmsemicon Limited and logo are both trademarks of our company.

\* Our company preserve the rights to further elaborate on the improvements about products' function, reliability and design in this manual. However, our company is not responsible for any usage about this manual. The applications and their purposes in this manual are just for clarification, our company does not guarantee that these applications are feasible without further improvements and changes, and our company does not recommend any usage of the products in areas where people's safety is endangered during accident. Our company's products are not authorized to be used for life-saving or life support devices and systems.our company has the right to change or improve the product without any notification, for latest news, please visit our website: [www.mcu.com.cn](http://www.mcu.com.cn).

## Index

<b>CMS8S588x series .....</b>	<b>1</b>
<b>Index .....</b>	<b>2</b>
<b>1. Central processing unit (CPU) .....</b>	<b>10</b>
1.1 Reset vector (0000H) .....	10
1.2 BOOT Partition .....	10
1.3 Accumulator (ACC) .....	12
1.4 B register (B) .....	12
1.5 Stack pointer register (SP) .....	13
1.6 Data pointer register (DPTR0/DPTR1) .....	13
1.7 Data pointer selection register (DPS) .....	13
1.8 Program status register (PSW) .....	14
1.9 Program counter (PC) .....	14
1.10 Timing access register (TA) .....	15
<b>2. Memory and register map .....</b>	<b>16</b>
2.1 Program memory FLASH .....	16
2.2 Non-volatile data storage Data FLASH .....	17
2.3 General data memory RAM .....	17
2.4 General external data register XRAM .....	19
2.5 Special function register SFR .....	20
2.6 External special function register XSFR .....	21
<b>3. Reset .....</b>	<b>26</b>
3.1 Power-on reset .....	26
3.2 External reset .....	28
3.3 LVRLow voltage reset .....	28
3.4 Watchdog reset .....	29
3.5 Software reset .....	29
3.6 CONFIG state protection reset .....	30
3.7 Power-on configuration monitoring reset .....	30
<b>4. Clock Structure .....</b>	<b>31</b>
4.1 System Clock Structure .....	31
4.2 Related Registers .....	32
4.2.1 Oscillator Control Register CLKDIV .....	32
4.2.2 System clock switch register SCKSEL .....	32
4.2.3 System clock status register SCKSTAU .....	33
4.2.4 System Clock Monitor Register SCM .....	34
4.2.5 Function clock control register .....	35
4.3 System Clock Switch .....	36
4.4 System Clock Monitoring .....	37
<b>5. Power management .....</b>	<b>38</b>
5.1 power management register PCON .....	38
5.2 Power monitoring register LVDCON .....	39
5.3 IDLE idle mode .....	39
5.4 STOP sleep mode .....	40
5.4.1 Sleep wakeup After .....	40
5.4.2 Wake-up wait state .....	40
5.4.3 Sleep wake-up time .....	40
5.4.4 Reset operation .....	41

5.4.5	Sleep power consumption in debug mode .....	41
5.4.6	Sleep mode application: example .....	41
<b>6.</b>	<b>Interrupt .....</b>	<b>42</b>
6.1	Interrupt Overview .....	42
6.2	External interrupt .....	43
6.2.1	INT0/INT1 interrupt .....	43
6.2.2	GPIO interrupt .....	43
6.3	Interrupt and sleep wake-up .....	43
6.4	Interrupt Register .....	44
6.4.1	Interrupt Mask Register .....	44
6.4.2	Interrupt Priority Control register .....	48
6.4.3	Interrupt Flag Register .....	51
6.4.4	Clear operation of interrupt flag .....	56
6.4.5	Special interrupt flag in debug mode .....	57
<b>7.</b>	<b>I/O port .....</b>	<b>58</b>
7.1	GPIO function .....	58
7.1.1	PORTx data register .....	58
7.1.2	PORTx direction register PxTRIS .....	58
7.1.3	PORTx open drain control register PxOD .....	59
7.1.4	PORTx pull-up resistor control register PxUP .....	59
7.1.5	PORTx pull-down resistance control register PxRD .....	59
7.1.6	PORTx Slope Control Register PxSR .....	60
7.1.7	PORTx data input selection register PxDS .....	60
7.2	Multiplexed function .....	61
7.2.1	Port multiplexed function table .....	61
7.2.2	Port Multiplexing Configuration Register Function .....	63
7.2.3	Port input function allocation register .....	63
7.2.4	Communication input function allocation register .....	65
7.2.5	Port external interrupt control register .....	66
7.2.6	Reuse function application note .....	67
<b>8.</b>	<b>Watch Dog Timer .....</b>	<b>68</b>
8.1	Overview .....	68
8.2	Related registers .....	68
8.2.1	Watchdog control register WDCON .....	68
8.2.2	Watchdog overflow control register CKCON .....	69
8.3	The WDT interrupt .....	70
8.3.1	Interrupt mask register EIE2 .....	70
8.3.2	Interrupt priority control register EIP2 .....	71
<b>9.</b>	<b>Timer 0/1 .....</b>	<b>72</b>
9.1	Overview .....	72
9.2	Related register .....	73
9.2.1	Timer0/1 Mode register TMOD .....	73
9.2.2	Timer0/1 control register TCON .....	74
9.2.3	Timer0 data register low TL0 .....	75
9.2.4	Timer0 data register high bit TH0 .....	75
9.2.5	Timer1 data register low TL1 .....	75
9.2.6	Timer1 data register high TH1 .....	75
9.2.7	Function clock control register CKCON .....	76
9.3	Timer0/1 interrupt .....	77
9.3.1	Interrupt mask register IE .....	77

9.3.2	Interrupt priority control register IP .....	78
9.3.3	Timer0/1, INT0/1 interrupt flag register TCON.....	78
9.4	Timer0 working mode .....	80
9.4.1	T0-Mode 0 (13-bit timer/counting mode).....	80
9.4.2	T0-Mode 1 (16-bit timer/counting mode).....	80
9.4.3	T0 - Mode 2 (8-bit auto reload timer / counter mode).....	81
9.4.4	T0-Mode 3 (two separate 8-bit timers/counters).....	81
9.5	Timer1 working mode .....	82
9.5.1	T1-Mode 0 (13-bit timer/counting mode).....	82
9.5.2	T1-Mode 1 (16-bit timer/counting mode).....	82
9.5.3	T1-Mode 2 (8-bit auto-reload timer/counting mode).....	83
9.5.4	T1-Mode 3 (stop counting).....	83
<b>10.</b>	<b>Timer 2 .....</b>	<b>84</b>
10.1	SUMMARY .....	84
10.2	Associated register .....	85
10.2.1	control register of Timer2 .....	85
10.2.2	Timer2 data register low TL2.....	85
10.2.3	Timer2 data register high bit TH2 .....	86
10.2.4	Timer2 compare/capture/auto-reload register low bit RLDL .....	86
10.2.5	Timer2 compare/capture/auto-reload register high RLDH.....	86
10.2.6	Timer2 compare/capture channel 1 register low bit CCL1.....	86
10.2.7	Timer2 compare/capture channel 1 register high CCH1 .....	86
10.2.8	Timer2 Compare / Capture channel 2 register lower.....	87
10.2.9	Timer2 compare/capture channel 2 register high CCH2 .....	87
10.2.10	Timer2 Compare / Capture channel 3 register lower.....	87
10.2.11	Timer2 Compare / Capture channel 3 registers high.....	87
10.2.12	Timer2 compare capture control register CCEN .....	88
10.3	Timer2 interrupt .....	89
10.3.1	Interrupt related registers .....	89
10.3.2	Timer interruption .....	92
10.3.3	External trigger interrupt.....	92
10.3.4	Compare interrupt .....	92
10.3.5	Capture interrupt .....	92
10.4	Timer2 function description .....	93
10.4.1	Timing mode .....	93
10.4.2	Reload Mode.....	93
10.4.3	Gated timing mode.....	94
10.4.4	Event counting mode .....	94
10.4.5	Comparison mode.....	94
10.4.6	Capture mode .....	96
<b>11.</b>	<b>Timer 3/4 .....</b>	<b>98</b>
11.1	Overview .....	98
11.2	Related register .....	98
11.2.1	Timer3/4 Control register T34MOD .....	98
11.2.2	Timer3 data register low TL3.....	99
11.2.3	Timer3 data register high TH3.....	99
11.2.4	Timer4 data register low TL4.....	99
11.2.5	Timer4 data register high bit TH4 .....	99
11.3	Timer3/4 interrupt .....	100
11.3.1	Interrupt mask register EIE2.....	100
11.3.2	Interrupt priority control register EIP2.....	100
11.3.3	Peripheral interrupt flag register EIF2 .....	101

11.4	Timer3 working mode .....	102
11.4.1	T3-Mode 0 (13-bit timer mode).....	102
11.4.2	T3-Mode 1 (16-bit timing mode).....	102
11.4.3	T3-Mode 2 (8-bit auto-reload timing mode).....	103
11.4.4	T3-Mode 3 (two separate 8-bit timers) .....	103
11.5	Timer4 working mode .....	104
11.5.1	T4-Mode 0 (13-bit timing mode).....	104
11.5.2	T4-Mode 1 (16-bit timing mode).....	104
11.5.3	T4- Mode 2 (8-bit auto-reload timing mode).....	105
11.5.4	T4-Mode 3 (stop counting) .....	105
<b>12.</b>	<b>LSE Timer (LSE_Timer) .....</b>	<b>106</b>
12.1	Overview .....	106
12.2	Related register .....	106
12.2.1	LSE timer data register low 8 bits LSECRL.....	106
12.2.2	LSE timer data register high 8 bits LSECRH.....	106
12.2.3	LSE Timer Control Register LSECON .....	107
12.3	Interrupt and sleep wakeup .....	108
12.4	Function description .....	108
<b>13.</b>	<b>Wake Up Time(WUT).....</b>	<b>109</b>
13.1	Overview .....	109
13.2	Related register .....	109
13.2.1	WUTCRH register .....	109
13.2.2	WUTCRL register.....	109
13.3	Function description .....	110
<b>14.</b>	<b>Baud rate timer (BRT).....</b>	<b>111</b>
14.1	Overview .....	111
14.2	Related registers .....	111
14.2.1	BRT module control register BRTCON.....	111
14.2.2	BRT timer data load low 8-bit register BRTDL.....	111
14.2.3	BRT timer data load upper 8 bits register BRTDH.....	111
14.3	Function description .....	112
<b>15.</b>	<b>Cyclic redundancy check (CRC).....</b>	<b>113</b>
15.1	Overview .....	113
15.2	Related register .....	113
15.2.1	CRC data input register CRCIN .....	113
15.2.2	CRC operation result low 8-bit data register CRCDL .....	113
15.2.3	CRC operation result high 8-bit data register CRCDH .....	113
15.3	Function description .....	114
<b>16.</b>	<b>Buzzer driver (BUZZER) .....</b>	<b>115</b>
16.1	Overview .....	115
16.2	Related register .....	115
16.2.1	BUZZER control register BUZCON .....	115
16.2.2	BUZZER frequency control register .....	115
16.3	Function description .....	116
<b>17.</b>	<b>PWM module .....</b>	<b>117</b>
17.1	Overview .....	117
17.2	Feature.....	117
17.3	Port configuration .....	118
17.4	Functional description .....	119

17.4.1	Functional block diagram .....	119
17.4.2	Edge alignment .....	120
17.4.3	Complementary mode .....	121
17.4.4	Synchronous mode .....	122
17.5	PWM related register.....	123
17.5.1	PWM control register PWMCON .....	123
17.5.2	PWM output enable control register PWMOE .....	123
17.5.3	PWM0/1 clock prescaler control register PWM01PSC .....	124
17.5.4	PWM2/3 clock prescaler control register PWM23PSC .....	124
17.5.5	PWM4/5 clock prescaler control register PWM45PSC .....	124
17.5.6	PWM clock division control register PWMnDIV(n=0-5) .....	125
17.5.7	PWM data load enable control register PWMLOADEN .....	125
17.5.8	PWM output polarity control register PWMPINV .....	125
17.5.9	PWM Counter Mode Control Register PWMCNTM.....	126
17.5.10	PWM counter enable control register PWMCNTE.....	126
17.5.11	PWM counter mode control register PWMCNTCLR.....	126
17.5.12	PWM period data register low 8 bits PWMPnL (n=0-5) .....	126
17.5.13	PWM period data register high 8 bits PWMPnH (n=0-5).....	127
17.5.14	PWM compare data register low 8 bits PWMDnL (n=0-5).....	127
17.5.15	PWM compare data register high 8 bits PWMDnH (n=0-5).....	127
17.5.16	PWM Dead Band Enable Control Register PWMDTE.....	127
17.5.17	PWM0/1 Dead Time Delay Data Register PWM01DT.....	128
17.5.18	PWM2/3 Dead Time Delay Data Register PWM23DT.....	128
17.5.19	PWM4/5 Dead Time Delay Data Register PWM45DT.....	128
17.6	PWM Interrupt .....	129
17.6.1	Interrupt mask register EIE2.....	129
17.6.2	Interrupt priority control register EIP2.....	130
17.6.3	PWM Zero Interrupt Mask Register PWMZIE.....	130
17.6.4	PWM Down Compare Interrupt Mask Register PWMDIE.....	131
17.6.5	PWM Zero Interrupt Flag Register PWMZIF .....	131
17.6.6	PWM Down Compare Interrupt Flag Register PWMDIF .....	131
<b>18.</b>	<b>SPI module .....</b>	<b>132</b>
18.1	Overview .....	132
18.2	SPI port configuration.....	133
18.3	SPI hardware description .....	134
18.4	SPI related registers.....	135
18.4.1	SPI control register SPCR.....	135
18.4.2	SPI data register SPDR .....	135
18.4.3	SPI slave device selection control register SSCR.....	136
18.4.4	SPI status register SPSR .....	136
18.5	SPI master mode.....	137
18.5.1	Write conflict error .....	138
18.6	SPI slave mode .....	139
18.6.1	Addressed error .....	139
18.6.2	Write conflict error .....	140
18.7	SPI clock control logic .....	141
18.7.1	SPI clock phase and polarity control .....	141
18.7.2	SPI transmission format .....	141
18.7.3	CPHA=0 Transmission Format.....	141
18.7.4	CPHA=1 transmission format.....	142
18.8	SPI data transmission .....	143
18.8.1	SPI transmission start .....	143

18.8.2	SPI transmission ends .....	143
18.9	SPI timing diagram .....	144
18.9.1	Master mode transmission .....	144
18.9.2	Slave mode transmission .....	144
18.10	SPI Interrupt .....	145
18.10.1	Interrupt mask register EIE2 .....	145
18.10.2	Interrupt priority control register EIP2 .....	146
18.10.3	Peripheral interrupt flag register EIF2 .....	147
<b>19.</b>	<b>I<sup>2</sup>C module .....</b>	<b>148</b>
19.1	overview .....	148
19.2	I <sup>2</sup> C port configuration .....	149
19.3	I <sup>2</sup> C master control mode .....	149
19.3.1	I <sup>2</sup> C master mode timing period register .....	149
19.3.2	I <sup>2</sup> C master mode control and status register .....	150
19.3.3	I <sup>2</sup> C slave address register .....	153
19.3.4	I <sup>2</sup> C master mode sending and receiving data register .....	153
19.4	I <sup>2</sup> C slave mode .....	154
19.4.1	I <sup>2</sup> C self address register I2CSADR .....	154
19.4.2	I <sup>2</sup> C slave mode control and status register I2CSCR/I2CSSR .....	154
19.4.3	I <sup>2</sup> C slave mode to send and receive buffer register .....	155
19.5	I <sup>2</sup> C interrupt .....	156
19.5.1	Interrupt mask register EIE2 .....	156
19.5.2	Interrupt priority control register EIP2 .....	157
19.5.3	Peripheral interrupt flag register EIF2 .....	158
19.6	I <sup>2</sup> C Slave Mode Transmission Mode .....	159
19.6.1	Single reception .....	159
19.6.2	Single Transmission .....	160
19.6.3	Continuous reception .....	161
19.6.4	Continuous transmission .....	162
<b>20.</b>	<b>UARTn module .....</b>	<b>163</b>
20.1	overview .....	163
20.2	UARTn port configuration .....	163
20.3	UARTn baud rate .....	164
20.3.1	Baud rate clock source .....	164
20.3.2	Baud rate calculation .....	165
20.3.3	baud rate error .....	166
20.4	UARTn register .....	168
20.4.1	UART0/1 baud rate selection register FUNCCR .....	168
20.4.2	UARTn buffer register SBUFn .....	168
20.4.3	UART control register SCOn .....	169
20.4.4	PCON register .....	170
20.5	UARTn interrupt .....	171
20.5.1	Interrupt mask register IE .....	171
20.5.2	Interrupt priority control register IP .....	172
20.5.3	Interrupt priority register EIP3 .....	173
20.6	UARTn mode .....	174
20.6.1	Mode 0-synchronous mode .....	174
20.6.2	Mode 1-8 bit asynchronous mode (variable baud rate) .....	174
20.6.3	Mode 2-9 bit asynchronous mode (fixed baud rate) .....	175
20.6.4	Mode 3-9 bit asynchronous mode (variable baud rate) .....	175
<b>21.</b>	<b>Analog-to-digital converter (ADC) .....</b>	<b>176</b>

21.1	overview .....	176
21.2	ADC configuration .....	177
21.2.1	Port configuration .....	177
21.2.2	Channel selection.....	177
21.2.3	ADC reference voltage.....	177
21.2.4	Clock conversion.....	178
21.2.5	Result format The result of.....	178
21.3	ADC hardware trigger start.....	179
21.3.1	External port edge trigger ADC .....	179
21.3.2	PWM trigger ADC.....	179
21.3.3	Hardware trigger start delay.....	179
21.4	ADC result comparison .....	180
21.5	ADC working principle .....	180
21.5.1	Start conversion .....	180
21.5.2	Completion of the conversion.....	180
21.5.3	Termination of conversion .....	180
21.5.4	A/D conversion steps .....	180
21.5.5	When the conversion process .....	181
21.6	related registers related.....	182
21.6.1	AD control register ADCON0.....	182
21.6.2	AD control register ADCON1.....	183
21.6.3	AD control register ADCON2.....	184
21.6.4	AD Channel selection register ADCCHS.....	185
21.6.5	AD comparator control register ADCMPC .....	185
21.6.1	AD hardware trigger delay data register ADDLYL .....	186
21.6.2	AD data register high ADRESH, ADFM=0 (left align).....	186
21.6.3	AD data register lower ADRESL, ADFM = 0(left align).....	186
21.6.4	AD data register high ADRESH, ADFM=1 (right align).....	186
21.6.5	AD data register low bit ADRESL, ADFM = 1 (right-align).....	186
21.6.6	AD comparator data register ADCMPH.....	187
21.6.7	AD comparator data register ADCMPL .....	187
21.6.8	AD reference voltage control register.....	187
21.7	ADC Interrupt .....	188
21.7.1	Interrupt mask register EIE2.....	188
21.7.2	Interrupt priority control register EIP2.....	188
21.7.3	Peripheral interrupt flag register EIF2 .....	189
<b>22.</b>	<b>Flash memory .....</b>	<b>190</b>
22.1	Overview .....	190
22.2	Related register .....	191
22.2.1	FLASH protection lock register MLOCK.....	191
22.2.2	FLASH memory data register MDATA.....	191
22.2.3	FLASH memory address register MADRL .....	191
22.2.4	FLASH memory address register MADRH .....	191
22.2.5	Lower 8 bits of the program CRC operation result data register PCRCDL.....	192
22.2.6	Program CRC operation result data register high 8 bits PCRCDH.....	192
22.2.7	FLASH memory control register .....	192
22.3	Function description .....	193
<b>23.</b>	<b>Unique ID (UID) .....</b>	<b>194</b>
23.1	Overview .....	194
23.2	UID register description.....	194
<b>24.</b>	<b>User configuration.....</b>	<b>197</b>



---

<b>25. Online programming and debugging .....</b>	<b>199</b>
25.1 Online programming mode .....	199
25.3 Online debugging mode .....	200
<b>26. Instructions .....</b>	<b>201</b>
26.1 Symbol description .....	201
26.2 instruction list.....	202
<b>27. Version revision description.....</b>	<b>205</b>

# 1. Central processing unit (CPU)

This series is a microcontroller with 8-bit 8051 frame structure. The CPU is the core component of the microcontroller, which is composed of arithmetic units, controllers, and special register groups. The arithmetic unit module mainly implements data arithmetic and logic operations, bit variable processing and data transfer operations; the controller module mainly decodes instructions, and then sends out various control signals; the special register group is mainly used to indicate the memory address of the current instruction to be executed, Store the operand and indicate the state after the instruction is executed. The special register group mainly includes accumulator ACC, general register B, stack pointer SP, data pointer DPTR, Program status register PSW, Program counter PC, etc.

## 1.1 Reset vector (0000H)

The microcontroller has a word-length system reset vector (0000H). After a reset occurs, the program will restart from 0000H, and the system registers will all be restored to default values. The following program demonstrates how to define the reset vector in FLASH.

Example: define reset vector

	ORG	0000H	; System reset vector
	LJMP	START	
	ORG	0010H	; User program start
START:			
	...		; User program
	...		
	END		; End of program

## 1.2 BOOT Partition

The size of the program area is 32K\*8 bit, and the program is divided into BOOT area and APROM area. The size of the BOOT area is allocated by the user configuration register.

The chip is powered on, if the program starts from the BOOT area, it needs to meet: the address space allocation method is 1/2/3 (set BOOT\_1K/BOOT\_2K through CONFIG/BOOT\_4K), otherwise the program will start from APROM area.

Take the BOOT area 1K space as an example: CONFIG configures BOOT\_1K, after the chip is powered on and configured, the program starts to run from address 7C00. If the program needs to switch between the BOOT area and APROM area, write 0xAA/0x55 to the BOOT area control register BOOTCON (see register description for details), and then execute Software reset or generate watchdog reset.

When Power-on reset, External reset, and voltage reset, BOOTCON reset value is 0x00, Software reset and Watchdog reset cannot clear this register.

## BOOT control register (BOOTCON)

F691H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
BOOTCON	D7	D6	D5	D4	D3	D2	D1	D0
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      D<7:0>: BOOT area control bit (this register can only be written when the chip is configured as BOOT\_1K/BOOT\_2K/BOOT\_4K);  
                   0x55= If you switch from APROM area to BOOT area, you need to write 0x55 to it, then execute Software reset or generate Watchdog reset;  
                   0xAA= If you switch from BOOT area to APROM area, you need to write 0xAA to it, then execute Software reset or generate Watchdog reset;  
                   Other value= Invalid.

For example: After the chip is powered on and started from the BOOT area, use the Software reset method to switch to the APROM area, and the configuration is as follows:

- 1) BOOTCON register needs to be written with AAH

```
MOV  DPTR,#BOOTCON
MOV  A,#0AAH
MOVX @DPTR,A
```

- 2) Execution software reset

```
MOV  TA,#0AAH
MOV  TA,#055H
MOV  WDCON,#080H
```

For example: use software reset method, then switch from APROM area to BOOT area, the configuration is as follows:

- 1) BOOTCON register needs to be written with 55H

```
MOV  DPTR, #BOOTCON
MOV  A,#055H
MOVX @DPTR,A
```

- 2) Execution software reset

```
MOV  TA,#0AAH
MOV  TA,#055H
MOV  WDCON,#080H
```

Note: When the BOOT function is valid, the APROM program needs to ensure that the PC will not overflow (overflow means that the PC exceeds the address range of the APROM). If the PC overflows, the system may operate abnormally.

### 1.3 Accumulator (ACC)

ALU is an 8Bit wide arithmetic logic unit, and all the mathematics and logic operations of the MCU are completed through it. It can add, subtract, shift and logic operations on data; ALU also controls the status bit (in the PSW status register) to indicate the status of the operation result.

The ACC register is an 8Bit register, the result of the ALU operation can be stored here.

### 1.4 B register (B)

The B register is used when using multiplication and division instructions. If you don't use multiplication and division instructions, it can also be used as a general-purpose register.

## 1.5 Stack pointer register (SP)

The SP register points to the address of the stack. The default value after reset is 0x07, which means that the area of the stack starts from 08H of the RAM address. The value of the SP can be modified. If the stack area is set to start from 0xC0, the value of SP needs to be set to 0xBF after the system is reset.

The operations that affect the SP are: instructions PUSH, LCALL, ACALL, POP, RET, RETI and entering interrupt.

The PUSH instruction occupies one byte on the stack; LCALL, ACALL and interrupt occupies two bytes on the stack, the POP instruction releases one byte, and the RET/RETI instruction releases two bytes.

Use the PUSH instruction to automatically save the current value of the operated register to RAM.

## 1.6 Data pointer register (DPTR0/DPTR1)

The data pointer is mainly used in MOVX, MOVC instruction, its function is to locate the address in XRAM and ROM. There are two data pointer registers DPTR0 and DPTR1 inside the chip, which are selected by the DPS register.

Each group of pointers includes two 8-bit registers: DPTR0={DPH0,DPL0}; DPTR1={DPH1,DPL1};

For example, the assembly code for operating XRAM is as follows:

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

## 1.7 Data pointer selection register (DPS)

Data pointer selection register DPS

0x86	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
DPS	ID1	ID0	TSL	AU	--	--	--	SEL
Read/write	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 ID<1:0>: Self-decreasing/self-increasing function selection.  
 00= DPTR0 plus 1 or DPTR1 plus 1;  
 01= DPTR0 minus 1 or DPTR1 plus 1;  
 10= DPTR0 plus 1 or DPTR1 minus 1;  
 11= DPTR0 minus 1 or DPTR1 minus 1.
- Bit5 TSL: Flip selection enable;  
 1= After executing the DPTR instruction, the SEL bit will automatically flip;  
 0= DPTR related instructions do not affect the SEL bit.
- Bit4 AU: Self-add/subtract enable bit;  
 1= After the MOVX @DPTR or MOVC @DPTR instruction is allowed to run, the self-decrement/self-increment operation (determined by ID1-ID0) is executed.  
 0= DPTR related instructions do not affect DPTR itself
- Bit3~Bit1 -- Reserved, all must be 0.
- Bit0 SEL: Data pointer selection bit;  
 1= Choose DPTR1;  
 0= Choose DPTR0.

## 1.8 Program status register (PSW)

Program status register PSW

0xD0	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PSW	CY	AC	F0	RS1	RS0	OV	--	P
Read/write	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; 1= Carry; 0= No carry.
Bit6	AC: Auxiliary carry flag (half carry flag); 1= Carry; 0= No carry.
Bit5	F0: General flag.
Bit4~Bit3	RS<1:0>: Working register BANK selection bit; 00= Choose Bank0; 01= Choose Bank1; 10= Choose Bank2; 11= Choose Bank3.
Bit2	OV: Overflow flag 1= Overflow in arithmetic or logical operation; 0= There is no overflow in arithmetic or logical operations.
Bit1	-- Reserved, must be 0.
Bit0	P: Check Digit; 1= The highest bit of the result has a carry. 0= The highest bit of the result is not carried.

## 1.9 Program counter (PC)

The Program counter (PC) controls the execution order of instructions in the FLASH of the program memory. It can address the entire range of the FLASH. After obtaining the instruction code, the Program counter (PC) will automatically increase by one and point to the address of the next instruction code. But if you perform operations such as jump, conditional jump, subroutine call, initialization reset, interrupt, interrupt return, subroutine return, etc., the PC will load the address related to the instruction instead of the address of the next instruction.

When a conditional jump instruction is encountered and the jump condition is met, the next instruction read during the execution of the current instruction will be discarded, and a dummy instruction operation cycle will be inserted, and then the correct instruction can be obtained. Otherwise, the next instruction will be executed in sequence.

## 1.10 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 be written before the following operations are performed on TA.

```
MOV TA, #0AAH
```

```
MOV TA, #055H
```

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

Protected register: WDCON, CLKDIV, SCKSEL

## 2. Memory and register map

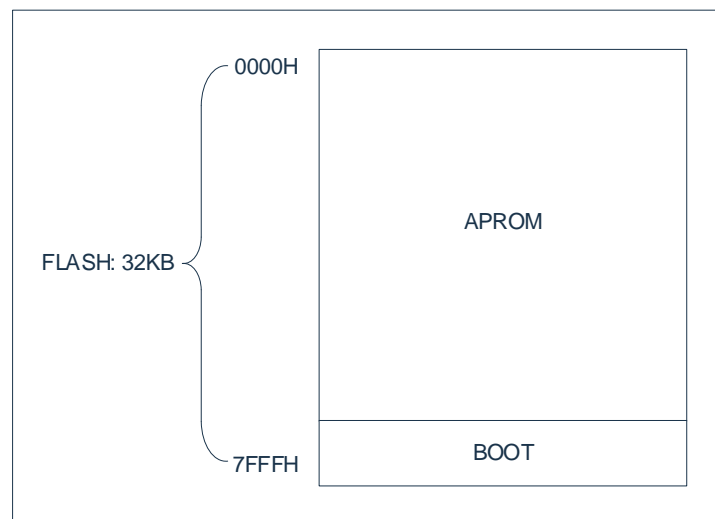
This series of microcontrollers has the following types of memories:

- ◆ Up to 32KB of FLASH program memory (shared by APROM area and BOOT area).
- ◆ Non-volatile data storage (Data FLASH) up to 1KB.
- ◆ General purpose internal data memory (RAM) up to 256B.
- ◆ General-purpose external data memory (XRAM) up to 2KB
- ◆ Special function register SFR
- ◆ External Special function register XSFR.

### 2.1 Program memory FLASH

The program memory FLASH is used to store source programs and table data, and Program counter PC is used as an address pointer. The PC is a 16-bit Program counter, so the address space that can be addressed is 64KB, but this chip only has 32K bytes program memory space

The block diagram of the FLASH space allocation structure is shown in the figure below:



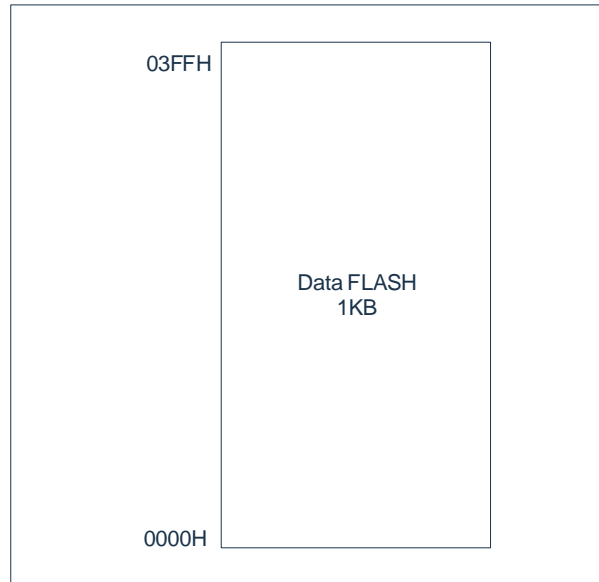
After the chip is reset, the CPU starts to execute from 0000H. Each interrupt is assigned a fixed address in the program memory, and the interrupt causes the CPU to jump to this address and start executing the service program.

For example, external interrupt 1 is assigned the address 0013H. If External Interrupt 1 is used, its service program must start at 0013H. If the interrupt is not used, its service address is used as the storage address of the ordinary program.



## 2.2 Non-volatile data storage Data FLASH

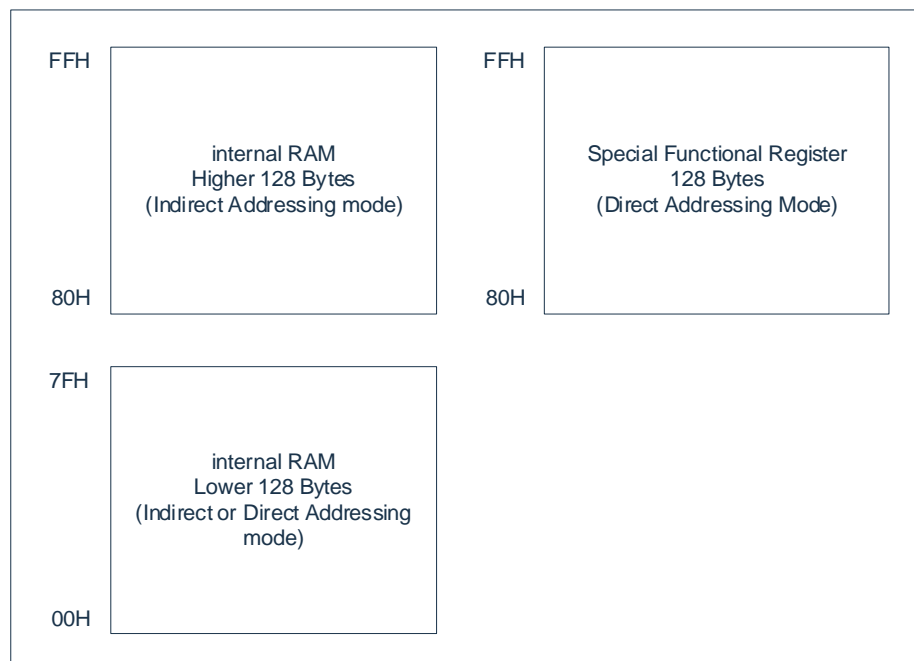
Non-volatile data storage data FLASH can be used to store important data such as constant data, calibration data, protection and safety-related information. The data stored in this area has the characteristic that the data will not be lost when the chip is powered off or suddenly or unexpectedly. The block diagram of the data FLASH space allocation structure is shown in the figure below:



Data FLASH memory read, write, and erase operations are realized through the FLASH control interface.

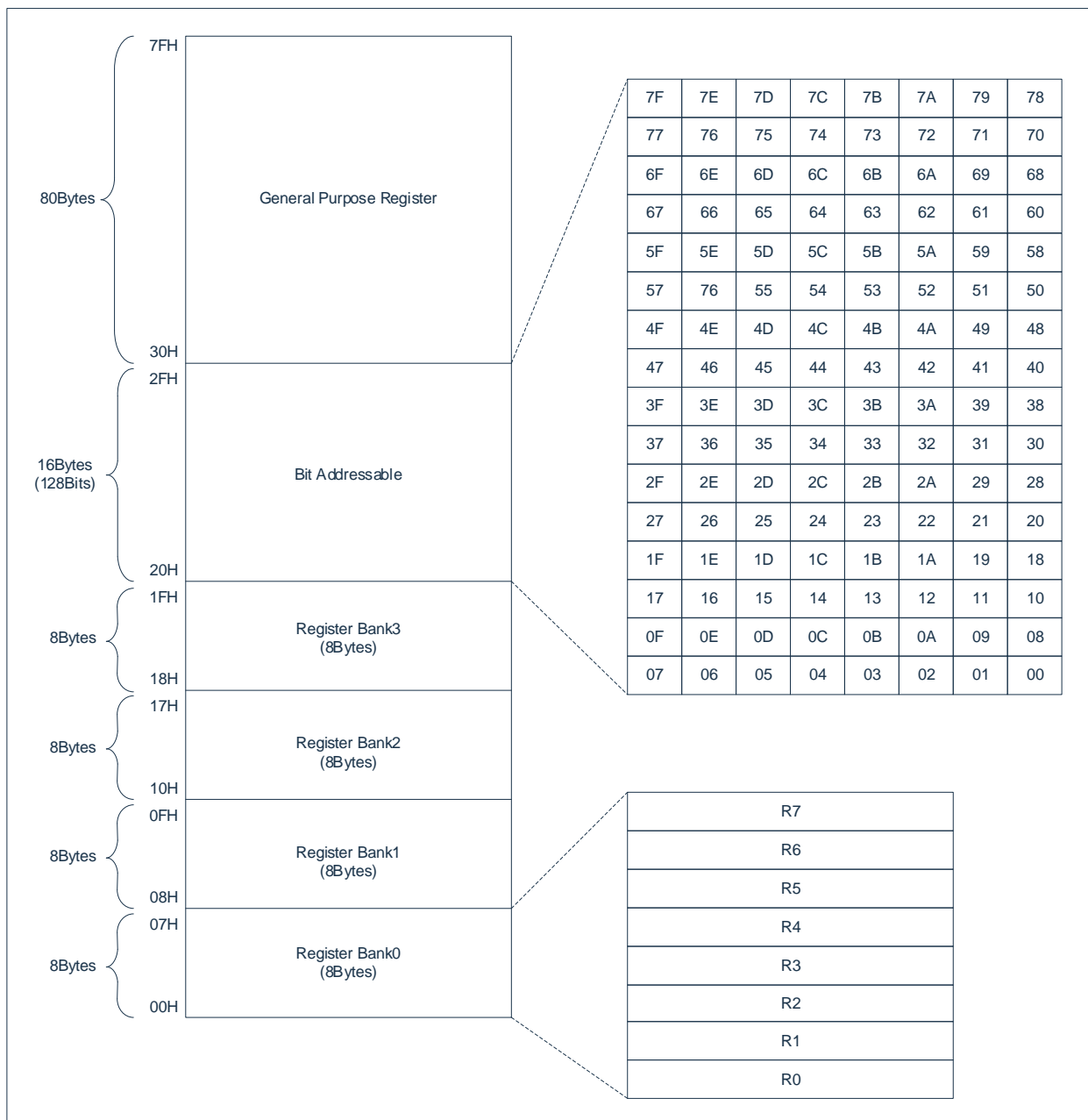
## 2.3 General data memory RAM

The internal data memory is divided into 3 parts: low 128Bytes, high 128Bytes, and Special function register SFR. The structure diagram of RAM space allocation is shown in the figure below:



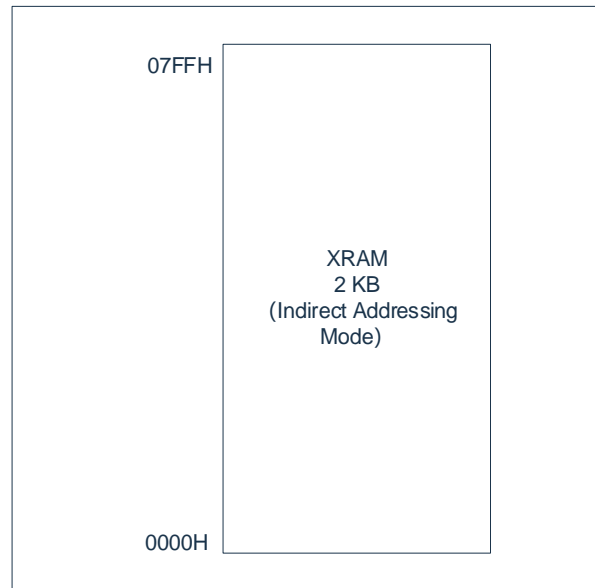
The high 128 bytes and SFR shown in the figure above occupy the same area (80H~FFH), but they are independent. Directly address the storage space (SFR) higher than 7FH and indirectly address the storage space higher than 7FH (high 128 bytes) into different storage spaces.

The lower 128 bytes space register allocation shown in the figure above is shown in the figure below. The lowest 32 bytes (00H~1FH) constitute 4 register groups, each group of 8 storage units, with R0~R7 as the unit number, used to store operands and intermediate results. After reset, group 0 is selected by default. If another register group is selected, it needs to be determined by changing the program state. The 16Bytes (20H~2FH) behind the register group constitute a bit-addressable storage space. The RAM unit in this area can be operated either by byte or directly on each bit in the unit. The remaining 80 storage units (30H~7FH), the user can set the stack area and store intermediate data.



## 2.4 General external data register XRAM

There is a maximum 4KB XRAM area inside the chip, which is not related to FLASH/RAM. The structure diagram of XRAM space allocation is shown in the figure below:



XRAM/XSFR space access is operated by the DPTR data pointer. DPTR includes two sets of pointers: DPTR0, DPTR1, which are selected by the DPS register. For example, through MOVX indirect addressing operation, the assembly code is as follows:

MOV	R0,#01H	
MOV	A,#5AH	
MOVX	@R0,A	; Write the data in A into XRAM address 01H, the upper 8-bit address is determined by DPH0/1

After setting Target-->Memory Model to Large in Keil51, the C compiler will use XRAM as the variable address. Generally use DPTR for XRAM/XSFR operations.

## 2.5 Special function register SFR

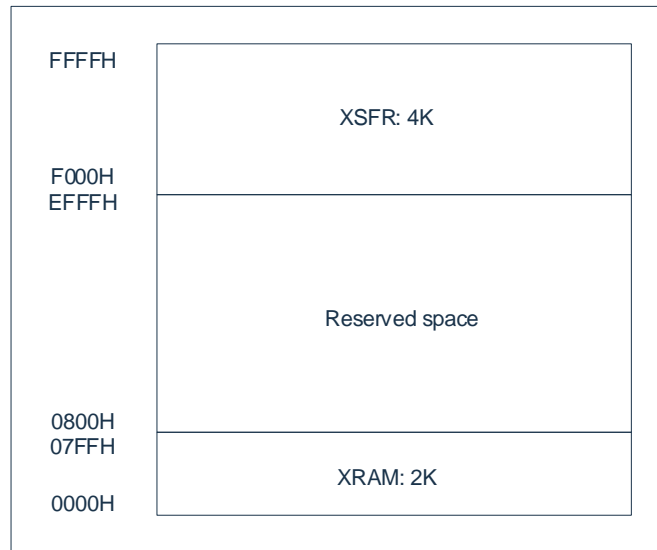
Special function register refers to a collection of special-purpose registers, which are essentially on-chip RAM units with special functions, which are discretely distributed in the address range 80H~FFH. Users can perform byte access to them through direct addressing instructions. The lower four bits of the address are 0000 or 1000 for bit addressing, such as P0, TCON, P1.

The Register Table is as following:

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
0xF8	--	PCRCDL	PCRCDH	MLOCK	MADRL	MADRH	MDATA	MCTRL
0xF0	B	I2CSADR	I2CSCR	I2CSBUF	I2CMSA	I2CMCR	I2CMBUF	I2CMTP
0xE8	--	ADCON2	SCON1	SBUF1	SPCR	SPSR	SPDR	SSCR
0xE0	ACC	--	TL4	TH4	--	--	--	--
0xD8	P5	ADCCHS	TL3	TH3	ADRESL	ADRESH	ADCON1	ADCON0
0xD0	PSW	ADCMPC	T34MOD	ADDLYL	ADCMPPL	ADCMPH	SCKSEL	SCKSTAU
0xC8	T2CON	T2IF	RLDL	RLDH	TL2	TH2	CCEN	T2IE
0xC0	--	--	CCL1	CCH1	CCL2	CCH2	CCL3	CCH3
0xB8	IP	EIP1	EIP2	EIP3	WUTCRL	WUTCRLH	BUZDIV	BUZCON
0xB0	--	--	EIF2	--	P0EXTIF	P1EXTIF	P2EXTIF	--
0xA8	IE	--	EIE2	--	P0EXTIE	P1EXTIE	P2EXTIE	--
0xA0	P2	P1TRIS	P2TRIS	--	--	P5TRIS	--	P5EXTIF
0x98	SCON0	SBUF0	P0TRIS	--	P5EXTIE	--	--	--
0x90	P1	FUNCCR	--	--	--	--	TA	WDCON
0x88	TCON	TMOD	TL0	TL1	TH0	TH1	CKCON	CLKDIV
0x80	P0	SP	DPL0	DPH0	DPL1	DPH1	DPS	PCON

## 2.6 External special function register XSFR

XSFR is a special register shared by the addressing space and XRAM, mainly including: port control register, other function control register. Its addressing range is shown in the figure below:



The external Special function register list is as follows:

Address	Register	Register description
F000H	P00CFG	P00 port configuration register
F001H	P01CFG	P01 port configuration register
F002H	P02CFG	P02 port configuration register
F003H	P03CFG	P03 port configuration register
F004H	P04CFG	P04 port configuration register
F005H	P05CFG	P05 port configuration register
F006H	P06CFG	P06 port configuration register
F007H	P07CFG	P07 port configuration register
F009H	P0OD	P0 port open drain control register
F00AH	P0UP	P0 port pull-up resistor control register
F00BH	P0RD	P0 port pull-down resistor control register
F00CH	P0DR	P0 port drive current selection register
F00DH	P0SR	P0 port slope control register
F00EH	P0DS	P0 port data input selection register
--	--	--
F010H	P10CFG	P10 port configuration register
F011H	P11CFG	P11 port configuration register
F012H	P12CFG	P12 port configuration register
F013H	P13CFG	P13 port configuration register
F014H	P14CFG	P14 port configuration register
F015H	P15CFG	P15 port configuration register
F016H	P16CFG	P16 port configuration register
F017H	P17CFG	P17port configuration register
F019H	P1OD	P1 port open drain control register
F01AH	P1UP	P1 port pull-up resistor control register
F01BH	P1RD	P1 port pull-down resistor control register
F01CH	--	--

Address	Register	Register description
F01DH	P1SR	P1 port slope control register
F01EH	P1DS	P1 port data input selection register
--	--	--
F020H	P20CFG	P20 port configuration register
F021H	P21CFG	P21 port configuration register
F022H	P22CFG	P22 port configuration register
F023H	P23CFG	P23 port configuration register
F024H	P24CFG	P24 port configuration register
F025H	P25CFG	P25 port configuration register
F026H	P26CFG	P26port configuration register
F027H	P27CFG	P27 port configuration register
F029H	P2OD	P2 port open drain control register
F02AH	P2UP	P2 port pull-up resistor control register
F02BH	P2RD	P2 port pull-down resistor control register
F02CH	--	--
F02DH	P2SR	P2 port slope control register
F02EH	P2DS	P2 port data input selection register
--	--	--
F050H	P50CFG	P50 port configuration register
F051H	P51CFG	P51 port configuration register
F052H	P52CFG	P52 port configuration register
F053H	P53CFG	P53 port configuration register
F054H	P54CFG	P54 port configuration register
F055H	P55CFG	P55 port configuration register
F056H	--	--
F057H	P57CFG	P57 port configuration register
F059H	P5OD	P5 port open drain control register
F05AH	P5UP	P5 port pull-up resistor control register
F05BH	P5RD	P5 port pull-down resistor control register
F05CH	--	--
F05DH	P5SR	P5 port slope control register
F05EH	P5DS	P5 port data input selection register
--	--	--
F080H	P00EICFG	P00 port interrupt control register
F081H	P01EICFG	P01 port interrupt control register
F082H	P02EICFG	P02 port interrupt control register
F083H	P03EICFG	P03 port interrupt control register
F084H	P04EICFG	P04 port interrupt control register
F085H	P05EICFG	P05 port interrupt control register
F086H	P06EICFG	--
F087H	P07EICFG	--
F088H	P10EICFG	P10 port interrupt control register
F089H	P11EICFG	P11 port interrupt control register
F08AH	P12EICFG	P12 port interrupt control register
F08BH	P13EICFG	--
F08CH	P14EICFG	--
F08DH	P15EICFG	--

Address	Register	Register description
F08EH	P16EICFG	P16 port interrupt control register
F08FH	P17EICFG	--
F090H	P20EICFG	P20 port interrupt control register
F091H	P21EICFG	P21 port interrupt control register
F092H	P22EICFG	P22 port interrupt control register
F093H	P23EICFG	--
F094H	P24EICFG	--
F095H	P25EICFG	P25 port interrupt control register
F096H	P26EICFG	--
F097H	P27EICFG	--
--	--	--
F0B8H	P50EICFG	P50 port interrupt control register
F0B9H	P51EICFG	--
F0BAH	P52EICFG	--
F0BBH	P53EICFG	P53 port interrupt control register
F0BCH	P54EICFG	P54 port interrupt control register
F0BDH	P55EICFG	--
F0BEH	--	--
F0BFH	P57EICFG	P57 port interrupt control register
--	--	--
F0C0H	PS_INT0	External Interrupt 0 input port allocation register
F0C1H	PS_INT1	External Interrupt 1 input port allocation register
F0C2H	PS_T0	Timer0 external clock input port allocation register
F0C3H	PS_T0G	Timer0 gate input port allocation register
F0C4H	PS_T1	Timer1 external clock input port 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 automatic reloading on falling edge input port allocation register
F0C8H	PS_CAP0	Timer2 input capture channel 0 port allocation register
F0C9H	PS_CAP1	Timer2 input capture channel 1 port allocation register
F0CAH	PS_CAP2	Timer2 input capture channel 2 port allocation register
F0CBH	PS_CAP3	Timer2 input capture channel 3 port allocation register
F0CCH	PS_ADET	ADC external trigger input port allocation register
--	--	--
F120H	PWMCON	PWM control register
F121H	PWMOE	PWM output enable register
F122H	PWMPINV	PWM output polarity select register
F123H	PWM01PSC	PWM0/1 clock pre-division control register
F124H	PWM23PSC	PWM2/3 clock pre-division control register
F125H	PWM45PSC	PWM4/5 clock pre-division control register
F126H	PWMCNTE	PWM count start control register
F127H	PWMCNTM	PWM count mode select register
F128H	PWMCNTCLR	PWM counter cleared control register
F129H	PWMLOADEN	PWM load enable control register
F12AH	PWM0DIV	PWM0 clock division control register
F12BH	PWM1DIV	PWM1 clock division control register

Address	Register	Register description
F12CH	PWM2DIV	PWM2 clock division control register
F12DH	PWM3DIV	PWM3 clock division control register
F12EH	PWM4DIV	PWM4 clock division control register
F12FH	PWM5DIV	PWM5 clock division control register
F130H	PWMP0L	PWM0 Low 8 bits of period data register
F131H	PWMP0H	PWM0 High 8 bits of period data register
F132H	PWMP1L	PWM1 Low 8 bits of period data register
F133H	PWMP1H	PWM1 High 8 bits of period data register
F134H	PWMP2L	PWM2 Low 8 bits of period data register
F135H	PWMP2H	PWM2 High 8 bits of period data register
F136H	PWMP3L	PWM3 Low 8 bits of period data register
F137H	PWMP3H	PWM3 High 8 bits of period data register
F138H	PWMP4L	PWM4 Low 8 bits of period data register
F139H	PWMP4H	PWM4 High 8 bits of period data register
F13AH	PWMP5L	PWM5 Low 8 bits of period data register
F13BH	PWMP5H	PWM5 High 8 bits of period data register
--	--	--
F140H	PWMD0L	PWM0 Low 8 bits of the compare data register
F141H	PWMD0H	PWM0 High 8 bits of the compare data register
F142H	PWMD1L	PWM1 Low 8 bits of the compare data register
F143H	PWMD1H	PWM1 High 8 bits of the compare data register
F144H	PWMD2L	PWM2 Low 8 bits of the compare data register
F145H	PWMD2H	PWM2 High 8 bits of the compare data register
F146H	PWMD3L	PWM3 Low 8 bits of the compare data register
F147H	PWMD3H	PWM3 High 8 bits of the compare data register
F148H	PWMD4L	PWM4 Low 8 bits of the compare data register
F149H	PWMD4H	PWM4 High 8 bits of the compare data register
F14AH	PWMD5L	PWM5 Low 8 bits of the compare data register
F14BH	PWMD5H	PWM5 High 8 bits of the compare data register
--	--	--
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
F164H	PWMMASKE	PWM mask control register
F165H	PWMMASKD	PWM mask data register
--	--	--
F169H	PWMZIE	PWM zero point Interrupt mask register
--	--	--
F16BH	PWMDIE	PWM compare down Interrupt mask register
--	--	--
F16DH	PWMZIF	PWM zero point interrupt flag register
--	--	--
F16FH	PWMDIF	PWM Compare down interrupt flag register
--	--	--
F5C0H	BRTCON	BRT module control register
F5C1H	BRTDL	BRT timer lower 8 bits of the data load value



Address	Register	Register description
F5C2H	BRTDH	BRT timer higher 8 bits of the data load value
--	--	--
F5E0H	UID0	UID<7:0>
F5E1H	UID1	UID<15:8>
F5E2H	UID2	UID<23:16>
F5E3H	UID3	UID<31:24>
F5E4H	UID4	UID<39:32>
F5E5H	UID5	UID<47:40>
F5E6H	UID6	UID<55:48>
F5E7H	UID7	UID<63:56>
F5E8H	UID8	UID<71:64>
F5E9H	UID9	UID<79:72>
F5EAH	UID10	UID<87:80>
F5EBH	UID11	UID<95:88>
--	--	--
F690H	LVDCON	Power monitoring register
F691H	BOOTCON	BOOT control register
F692H	ADCLDO	ADC reference voltage control register
--	--	--
F694H	LSECRH	LSE timer data register lower 8 bits
F695H	LSECRH	LSE timer data register higher 8 bits
F696H	LSECON	LSE timer control register
F697H	XT_SCM	LSE/HSE clock stop oscillation detection control register
F698H	PS_SCLK	SPI clock input port allocation register
F699H	PS_MOSI	SPI slave input port allocation register
F69AH	PS_MISO	SPI master input port allocation register
F69BH	PS_NSS	SPI chip select input port allocation register
F69CH	PS_SCL	IIC clock input port allocation register
F69DH	PS_SDA	IIC data input port allocation register
F69EH	PS_RXD1	UART1 data input port allocation register
F69FH	PS_RXD0	UART0 data input port allocation register
--	--	--
F708H	CRCIN	CRC module data input register
F709H	CRCDL	CRC the lower 8 bits of the operation result data register
F70AH	CRCDH	CRC the high 8-bit data register of the operation result
--	--	--

## 3. Reset

The reset time (Reset Time) refers to the time from the chip reset to the chip starting to execute instructions, and its default design value is about 16ms. This time includes the oscillator start-up time and configuration time. Whether the chip is power-on reset or reset caused by other reasons, there will be this reset time. In addition, when the oscillator is selected as an external low-speed crystal oscillation (32.768KHz), the reset time (including the start-up time) is about 1.5s by default (external capacitor 10pF~22pF).

The chip can be reset in the following ways:

- ◆ Power-on reset;
- ◆ External reset;
- ◆ Low voltage reset;
- ◆ Watchdog overflow reset;
- ◆ Software reset;
- ◆ CONFIG state protection reset;
- ◆ Power-on configuration monitoring reset.

When any of the above resets occurs, all system registers will return to the default state, the program will stop running, and Program counterPC will be cleared at the same time. After the reset, the program will start to run from the reset vector 0000H.

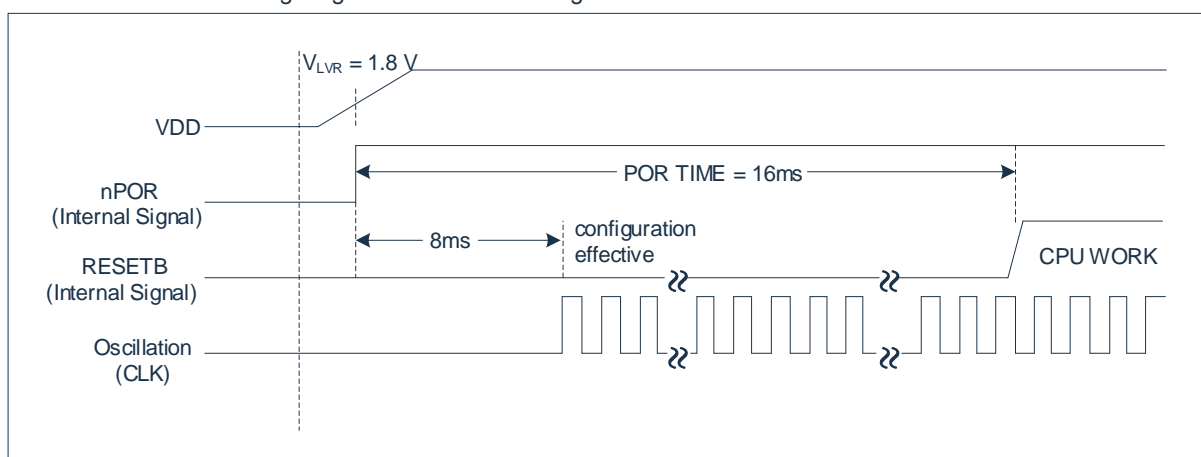
Any reset situation requires a certain response time, and the system provides a complete reset process to ensure the smooth progress of the reset action.

### 3.1 Power-on reset

Power-on reset is closely related to LVR operation. The power-on process of the system is in the form of a gradually rising curve, and it takes a certain time to reach the normal level value. The normal sequence of Power-on reset is given below:

- Power on: The system detects the power supply voltage rise and waits for it to stabilize;
- System initialization: all system registers are set to initial values;
- Oscillator starts to work: the oscillator starts to provide the system clock;
- Executing the program: the end of the power-on, the program starts to run.

The Stabilization Time defaults to 16ms. If the configuration selects 32.768KHz crystal oscillator, the stabilization time is about 1.5s. Power-on reset timing diagram is shown in the figure below:



Whether the system is power-on reset can be judged by PORF (WDCON.6) flag bit. The reset types that can set the PORF flag to 1 are: Power-on reset, LVR reset, power-on monitoring reset, CONFIG protection reset, external reset.

Reset source Flag bit	Power-on reset	LVR low voltage reset	Power-on monitoring reset	CONFIG protection reset	Software reset	External reset	Watchdog reset
SWRST	0	0	0	0	1	0	Not affected
PORF	1	1	1	1	Not affected	1	Not affected
EXTIF	0	0	Not affected	Not affected	Not affected	1	Not affected
FXTIF	0	0	Not affected	1	Not affected	Not affected	Not affected
WDTRF	0	0	0	0	Not affected	0	1
WWDTRF	0	0	Not affected	Not affected	Not affected	Not affected	Not affected

0x97	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
WDCON	SWRST	PORF	--	--	WDTIF	WDTRF	WDTRE	WDTCLR
R/W	R/W	R/W	R	R	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 system Software reset (write 0 to clear after reset).  
           0: --
- Bit6      PORF: Power-on reset flag;  
           1: The system is Power-on reset (write 0 to clear, no TA write sequence is required).  
           0: --
- Bit5~Bit4      -- Reserved, all must be 0.
- Bit3      WDTIF: WDT overflow interrupt flag bit;  
           1= WDT overflow (write 0 to clear);  
           0= The WDT did not overflow.
- Bit2      WDTRF: WDT reset flag;  
           1= The system is reset by WDT (write 0 to clear);  
           0= The system is not reset by WDT.
- Bit1      WDTRF: WDT reset enable bit;  
           1= Enable WDT to reset the CPU;  
           0= Disable WDT to reset CPU.
- Bit0      WDTCLR: WDT counter clear bit;  
           1= Clear the WDT counter (automatically cleared by hardware);  
           0= Disable WDT counter (writing 0 is invalid).

## 3.2 External reset

External reset refers to the reset signal from the external port (NRST), which resets the chip after being input by the Schmitt trigger. If the NRST pin remains low for about 16 $\mu$ s or more under the operating voltage range and stable oscillation (the internal LSI clock samples 3 rising edges), a reset will be requested. After the internal state is initialized and the reset state becomes "1", it takes 16ms to stabilize before the internal RESETB signal becomes "1", and the program starts to execute from the vector address 0000H.

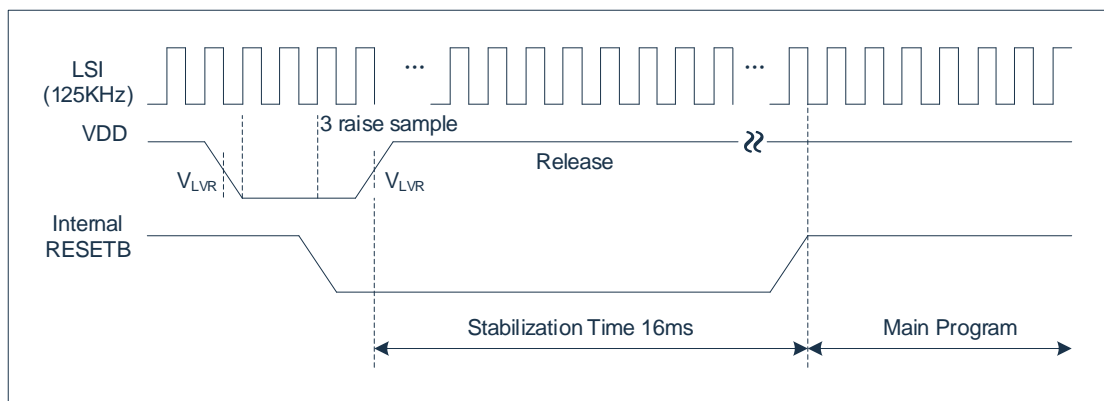
The chip reconfiguration process within the Stabilization Time is the same as the Power-on reset configuration process. External reset pin NRST and its pull-up resistor enable, configured through CONFIG.

## 3.3 LVR Low voltage reset

The low voltage reset (LVR) function is integrated inside the chip. When the system voltage VDD drops below the LVR voltage, the LVR is triggered and the system resets. The voltage point that triggers the reset can be set in CONFIG.

The LVR module detects that  $VDD < V_{LVR}$ , and it will request a reset. In sleep mode (STOP) mode, the LVR low voltage reset function is disabled.

The LVR low voltage reset timing diagram is shown in the figure below:



The chip reconfiguration process within the Stabilization Time is the same as the Power-on reset configuration process.

### 3.4 Watchdog reset

Watchdog reset is a protection setting of the system. In the normal state, the Watchdog timer is cleared by the program. If an error occurs, the system is in an unknown state, the Watchdog timer overflows, and the system is reset at this time. After Watchdog reset, the system restarts into normal state.

The WDT counter cannot be addressed. It starts counting when the program is running after the Power-on reset ends. It is recommended to clear the WDT counter first when setting the WDT register to accurately control the WDT overflow time.

The timing of Watchdog reset is as follows:

- 1) Watchdog timer status: the system detects whether the Watchdog timer overflows, if it overflows, the system resets;
- 2) Initialization: All system registers are set to the default state;
- 3) Program: reset is complete, the program starts to run from 0000H;

The clock source of the WDT is provided by the system clock, and the basic time period of the WDT counter is  $T_{sys}$ . Reset the CPU and all registers after the WDT overflows, and the program will start executing from 0000H immediately after 1  $T_{sys}$ . WDT reset will not re-configure Power-on reset. The overflow time of the watchdog can be set by the program, and the overflow time can be selected in the two bits WDS2-WTS0 of the CKCON register. The watchdog overflow time is shown in the table below:

WTS[2:0]	Watchdog Interval	Number of clocks	OVT@Fsys=16MHz	OVT@Fsys=48MHz
000	$2^{17}$	131072	8.192ms	2.731ms
001	$2^{18}$	262144	16.384ms	5.461ms
010	$2^{19}$	524288	32.768ms	10.923ms
011	$2^{20}$	1048576	65.536ms	21.845ms
100	$2^{21}$	2097152	131.072ms	43.691ms
101	$2^{22}$	4194304	262.144ms	87.381ms
110	$2^{24}$	16777216	1.048s	349.525ms
111	$2^{26}$	67108864	4.194s	1.398s

WDT can also be set to not reset the system and can generate interrupts.

### 3.5 Software reset

Program software reset can be implemented inside the chip. Software reset can relocate the program flow to the reset address 0000H, and then run the program again. The user can write Software reset control bit WDCON[7] (SWRST=1) to realize custom Software reset. Software reset will not re-configure Power-on reset.

## 3.6 CONFIG state protection reset

CONFIG state protection reset is a strengthening protection mechanism of the system. During Power-on reset, there is a set of 16-bit CONFIG registers inside, and the fixed code (A569H) set in FLASH is loaded. This register will not be operated during normal operation. If the value of the register changes and is not equal to the original fixed code in the case of a specific non-program operation, after several clock samples, the register continues to remain in the state of not being a fixed code, the system will reset.

The reset mechanism prevents the configuration bit from changing under certain conditions, causing the system to enter an unpredictable state.

In normal operation, the clock of the sampling register value is the internal RC fixed clock Fixed\_Clock (8MHz, clock source from HSI) and low-power clock (LSI 125KHz). Once the register value is not a fixed code, the LSI oscillator and the LSI oscillator are forced to be enabled. HSI oscillator, and the system clock is switched to the LSI clock, if after 12 Fixed\_Clock sampling or 3 LSI clock sampling, the register is still not in a fixed code state, the system will reset.

Under certain conditions, in order to prevent the oscillator from stopping, two kinds of clocks are used for sampling.

## 3.7 Power-on configuration monitoring reset

During the power-on configuration process, there is a configuration monitoring circuit inside the chip. If the power-on configuration time is too long, or the power-on configuration enters a certain state that cannot be reconfigured, the internal monitoring circuit starts timing from the configuration, and if it exceeds the set time, the monitoring circuit resets the configuration module and allows the configuration module to perform the configuration process again. To prevent the system from entering an unpredictable state when powering up.

The working clock of the monitoring circuit is LSI (125KHz), and the default monitoring time is 65ms. If 32.768KHz crystal oscillator is selected, the monitoring time is 2.1s.

## 4. Clock Structure

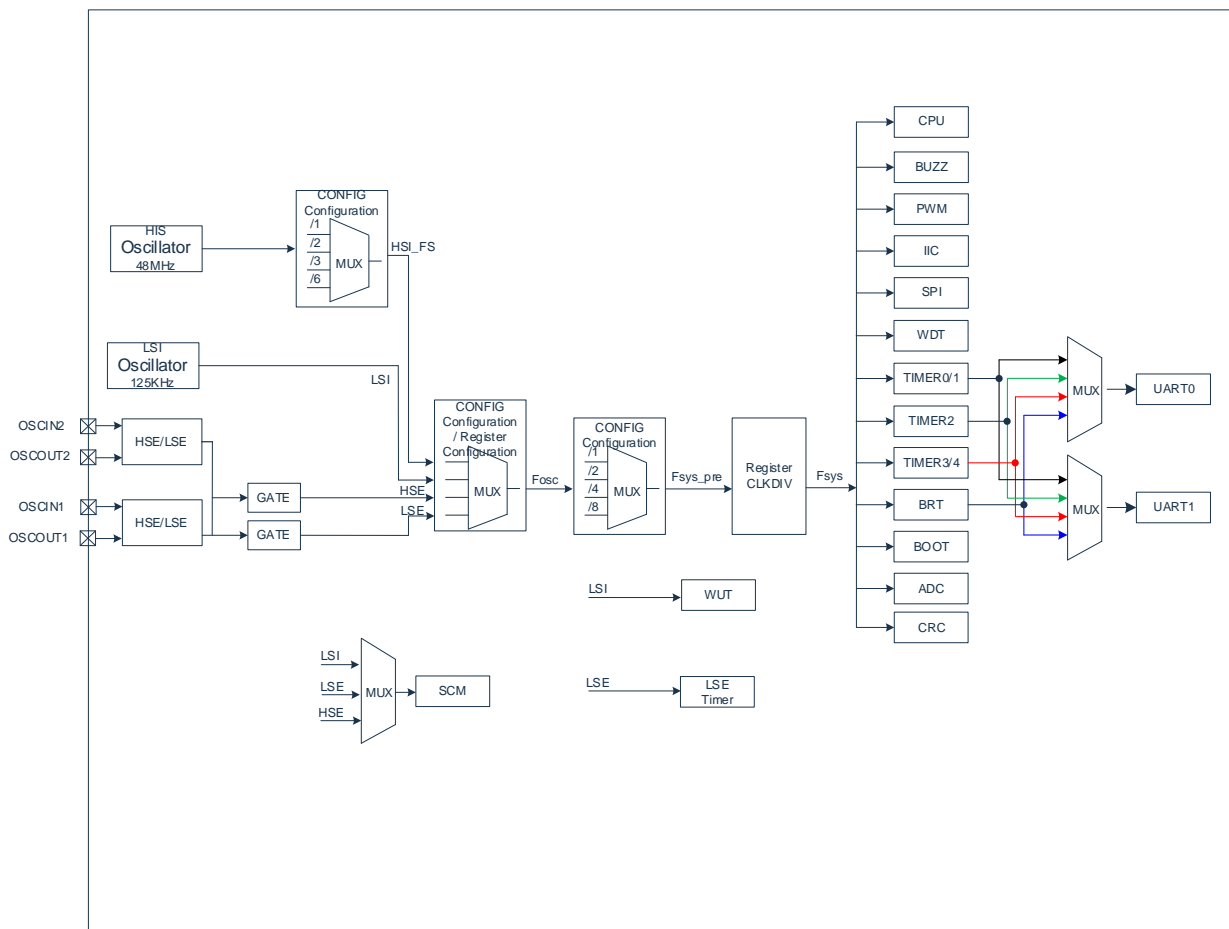
The clock source of the system clock has 4 types, and the clock source and clock division can be selected through the system configuration register or user register settings. The system clock source is as follows;

- ◆ Internal high-speed oscillation HSI (48MHz).
- ◆ External high-speed oscillator HSE (8MHz/16MHz).
- ◆ External low-speed oscillator LSE (32.768KHz).
- ◆ Internal low-speed oscillation LSI (125KHz).

The default clock source of the chip is HIS, upon completion of the chip reset, system clock will start to operate using HSI. In case change is required, system clock source can be configured via system configuration register ( this operation needs to work along with the programming tool and control PC software operation ), or via user register configuration ( following operational steps to configure relevant register, refer to below paragraphs for details ). While choosing external high speed or low speed oscillator as source of system clock, the chip supports ocilation stop detection function, refer to below paragraph for more details.

### 4.1 System Clock Structure

The clock structure block diagram of each Peripherals module system is shown in the figure below:



## 4.2 Related Registers

### 4.2.1 Oscillator Control Register CLKDIV

0x8F	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
CLKDIV	CLKDIV7	CLKDIV6	CLKDIV5	CLKDIV4	CLKDIV3	CLKDIV2	CLKDIV1	CLKDIV0
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                      CLKDIV<7:0>: System clock Fsys divider bit; ;  
 00H= Fsys=Fsys\_pre;  
 Others= Fsys=Fsys\_pre/ (2\*CLKDIV) (2,4...510 Frequency division) .

Instruction sequence required to modify CLKDIV (no other instructions can be inserted in between):

MOV	TA,#0AAH
MOV	TA,#055H
MOV	CLKDIV,#02H

### 4.2.2 System clock switch register SCKSEL

0xD6	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
SCKSEL	--	--	--	SEL	WRITE	CKSEL2	CKSEL1	CKSEL0
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                              -- Forbidden to be accessed.  
 Bit6~Bit5                      -- reserved, both must be 0s.  
 Bit4                              SEL: System clock configuration selection;  
                                     1= Use clock source as configured by CKSEL<2:0> configuration as reference;  
                                     0= Use clock source as configured by CONFIG as reference (default choice upon power up) .  
 Bit3                              WRITE: Write enable, a pulse is generated to perform clock source switch;  
                                     1= Clock source switch ( shall wait till target clock source stable bit been set to 1 before 1 to be written;  
                                     0= Do not switch clock source  
 Bit2~Bit0                      CKSEL<2:0>: System clock source selection bit;  
                                     111= LSI;  
                                     110= LSE;  
                                     101= HSE;  
                                     100= HSI;  
                                     Others= invalid, forbidden to be accessed.

After clock source switching, system will complete switching operation within few system clock cycles, it is recommended the program executes 6 NOP instructions before other following instructions to be executed.

Instruction sequence required to modify SCKSEL (no other instructions can be inserted in between):

MOV	TA,#0AAH
MOV	TA,#055H
MOV	SCKSEL, #05H



### 4.2.3 System clock status register SCKSTAU

0xD7	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
SCKSTAU	LSI_F	LSE_F	HSE_F	HSI_F	--	--	--	--
R/W	R	R	R	R	RW	R/W	R/W	R/W
Reset value	0	0	0	0	0	0	0	0

- Bit7      LSI\_F: Low speed internal stable status bit;  
           1= stable;  
           0= Not stable.
- Bit6      LSE\_F: Low speed external oscillator stable status bit;  
           1= stable;  
           0= Not stable.
- Bit5      HSE\_F: High speed external oscillator stable status bit;  
           1= stable;  
           0= Not stable.
- Bit4      HSI\_F: High speed internal clock stable status bit;  
           1= stable.  
           0= Not stable.
- Bit3~0      -- Forbidden to be accessed.

#### 4.2.4 System Clock Monitor Register SCM

F697H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
XT_SCM	SCMEN	SCMIE	--	--	--	--	SCMIF	SCMSTA
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           SCMEN: Oscillator stop detection module enable bit;

1= Enable;

0= Disable.

Bit6           SCMIE: Oscillator stop detection interrupt enable bit (This interrupt shares the same interrupt entry with LSE Timing interrupt) ;

1= Enable;

0= Disable.

Bit5~Bit2       -- reserved, must be 0.

Bit1           SCMIF: Oscillator stop interrupt flag bit;

1= Oscillator stopped;

0= Software clean to 0, Once cleared to , MCU will automatically switch to 0 HSE/LSE as main frequency source (only can be cleared by software to 0) .

Bit0           SCMSTA: Oscillator stop status bit, read only;

1= Oscillator stopped;

0= Oscillator stopped status recovered.

#### NOTE:

- Both SCMIF and SCMSTA can reflect status of HSE/LSE as system clock. The main difference between them is that when HSE/LSE oscillator stops, SCMSTA will always remain at high voltage level, until HSE/LSE recovers; SCMIF can as well indicate HSE/LSE oscillator stopping, but it can generate interrupt (while interrupt enabled), also it can be cleared via register interface once cleared, main frequency source will be switched to HSE/LSE (if the oscillator is still stopped, interrupt will be triggered again).
- While oscillator stops, main frequency source will switch from HSE/LSE to HIS, when HSE/LSE recovers, SCMSTA will automatically be cleared, main frequency source will be switch back from HIS to HSE/LSE.

## 4.2.5 Function clock control register

Watchdog overflow time / timer clock source select register CKCON

0x8E	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
CKCON	WTS2	WTS1	WTS0	T1M	T0M	--	--	--
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	1	1

Bit7~Bit5      WTS<2:0>:    WDT overflow time selection bit;  
                   000=  $2^{17} \cdot T_{sys}$ ;  
                   001=  $2^{18} \cdot T_{sys}$ ;  
                   010=  $2^{19} \cdot T_{sys}$ ;  
                   011=  $2^{20} \cdot T_{sys}$ ;  
                   100=  $2^{21} \cdot T_{sys}$ ;  
                   101=  $2^{22} \cdot T_{sys}$ ;  
                   110=  $2^{24} \cdot T_{sys}$ ;  
                   111=  $2^{26} \cdot T_{sys}$ .

Bit4            T1M:    Timer1 clock source selection bit;  
                   0=     $F_{sys}/12$ ;  
                   1=     $F_{sys}/4$ .

Bit3            T0M:    Timer0 clock source selection bit;  
                   0=     $F_{sys}/12$ ;  
                   1=     $F_{sys}/4$ .

Bit2~Bit0      --    Reserved, all must be 1.

UART0/1Baud rate select register FUNCRR

0x91	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
FUNCRR	--	UART1_CKS2	UART1_CKS1	UART1_CKS0	--	UART0_CKS2	UART0_CKS1	UART0_CKS0
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, must be 0.

Bit6~Bit4      UART1\_CKS<2:0>:    UART1 timer clock source selection;  
                   000=    Timer1 Overflow clock;  
                   001=    Timer4 Overflow clock;  
                   010=    Timer2 Overflow clock;  
                   011=    BRT Overflow clock;  
                   Others=    Forbidden to be accessed

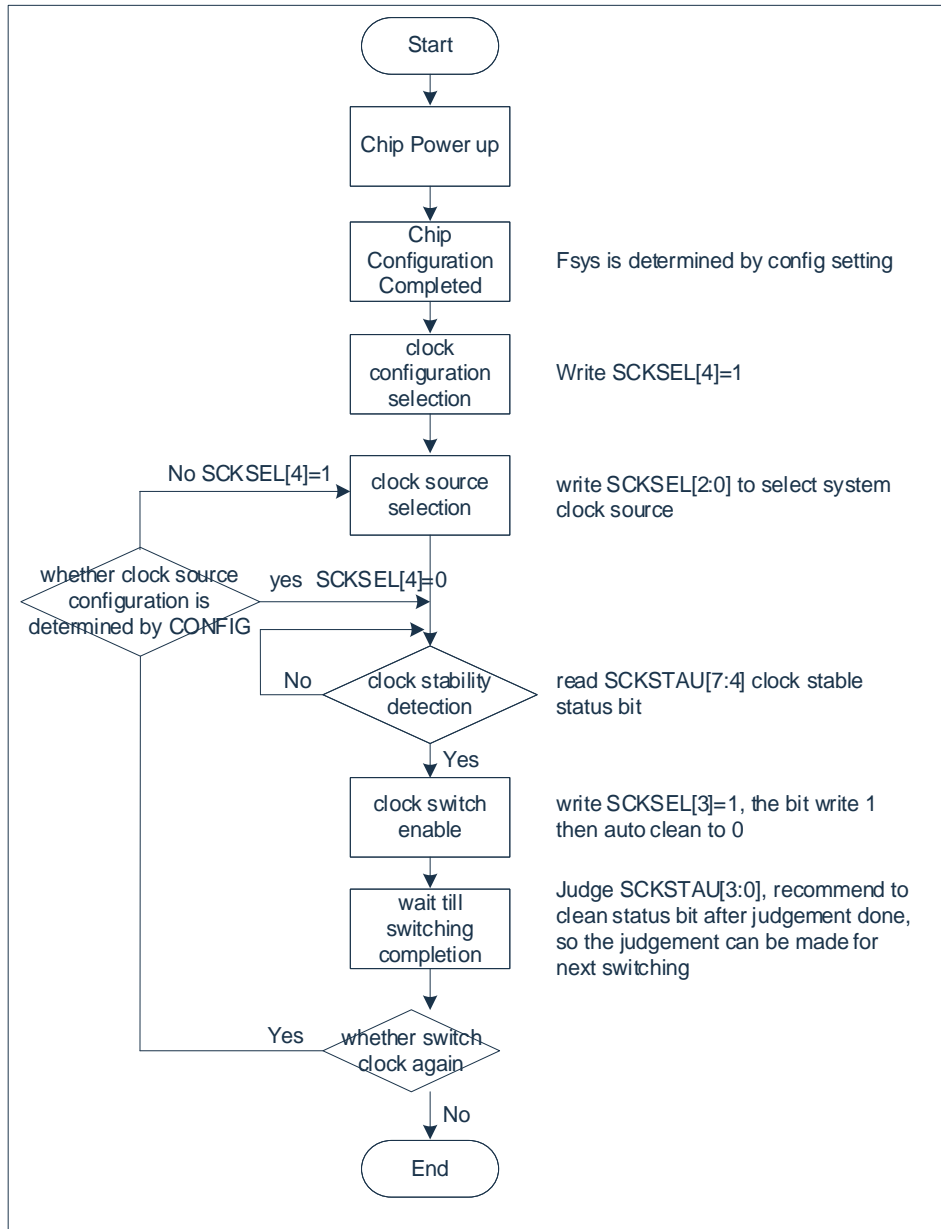
Bit3            --    Reserved, must be 0.

Bit2~Bit0      UART0\_CKS<2:0>:    UART0 timer clock source selection;  
                   000=    Timer1 Overflow clock;  
                   001=    Timer4 Overflow clock;  
                   010=    Timer2 Overflow clock;  
                   011=    BRT Overflow clock;  
                   Others=    Forbidden to be accessed

## 4.3 System Clock Switch

Chip has 2 sets of oscillator ports, at any particular point of time, only 1 set of oscillator port is valid, therefore HSE/LSE switching to each other functionality is forbidden. When current chip select external HSE clock, all LSE relevant functionalities are forbidden. When current chip select external LSE clock, all HSE relevant functionalities are forbidden.

System clock switching steps are shown in below diagram:



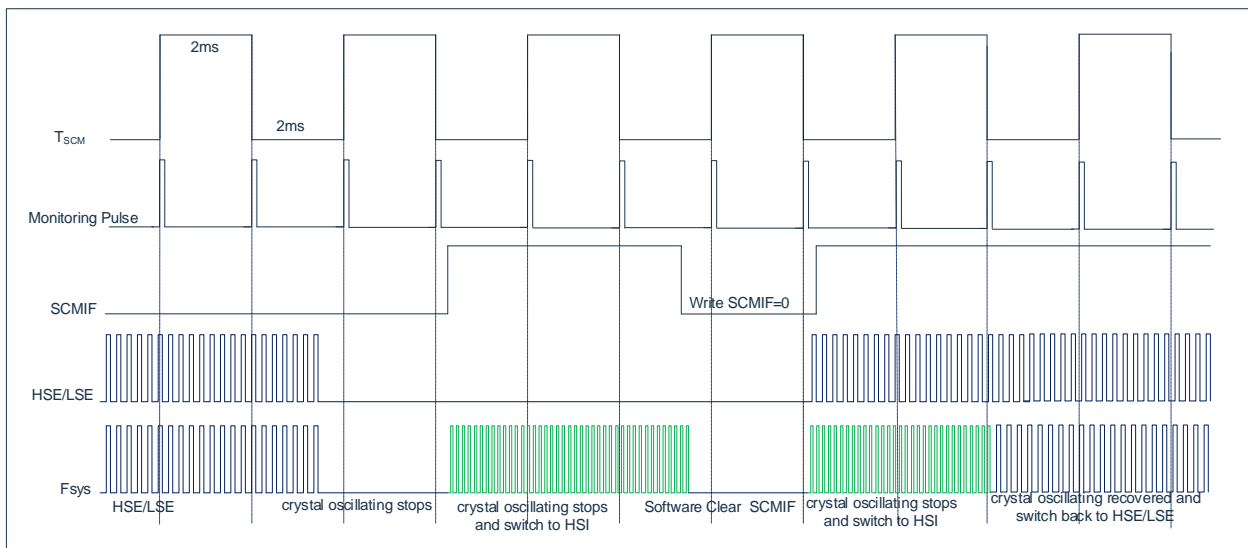
NOTE: When system clock source switches, it must ensure the respective clock source exists and is correctly configured.

## 4.4 System Clock Monitoring

System Clock Monitoring (SCM) is a detection and protection circuit designed to prevent the scenario that the system stop working due to oscillator stop. While HSE/LSE is used as system clock source, once the HSE/LSE clock is detected as stopped, system will force HSI clock source to start, once the HSI stabilized, system will use 8Mhz frequency to operate, when HSE/LSE clock recovered and stabilized, system clock will automatically switch back from HIS to HSE/LSE.

SCM module monitors system clock every 4ms, the period of monitoring  $T_{SCM}$  has duty-cycle of 1:1. When  $T_{SCM}$  at high voltage level SCM will stop monitoring the oscillator stop condition of HSE/LSE, when  $T_{SCM}$  at low frequency period, the module will process the result from the monitoring, if oscillator stop condition of HSE/LSE is detected, the system clock will switch to HSI, at the same time, the interrupt flag bit of oscillator stop SCMIF will be set to 1. If clear SCMIF, even though HSE/LSE is in oscillator stop condition, the system clock will automatically switch back to HSE/LSE.

System clock monitoring structural diagram is shown as following:



## 5. Power management

Low power consumption modes are divided into two categories:

- ◆ IDLE: idle mode
- ◆ STOP: sleep mode When

Users use C language for program development, it is strongly recommended to use the IDLE and STOP macro instructions to control the system mode, and do not directly set the IDLE and STOP bits. The macro is as follows:

enter idle mode: IDLE();

enter sleep mode: STOP();

### 5.1 power management register PCON

0x87	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PCON	SMOD0	SMOD1	--	--	--	SWE	STOP	IDLE
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	SMOD0:	UART0 baud rate multiplier bit; 0 = UART0 baud rate is normal; 1= Double the baud rate of UART0.
Bit6	SMOD1:	UART1 baud rate multiplication bit; 0= UART1 baud rate is normal; 1= UART1 baud rate is doubled.
Bit5 ~ Bit3	-	Reserved, must be 0's.
Bit2	SWE:	STOP status function wake-up enable bit; (Regardless of the value of SWE, the system can be restarted by power-off reset or enabled external reset) 0= Function wake-up is disabled; 1= Function wake-up is enabled (can be waked up by external interrupt and timed) wake).
Bit1	STOP:	Sleep state control bit; 0= not enter the dormant state; 1= enter the dormant state (automatically cleared when exiting STOP mode).
Bit0	IDLE:	Idle state control bit; 0= not enter idle state; 1= enter idle state (automatically cleared when exiting IDLE mode).

## 5.2 Power monitoring register LVDCON

This MCU has its own power detection function. If the LVD module is enabled (LVDEN=1) and the voltage monitoring point LVDSSEL is set at the same time, when the power supply voltage drops below the LVD setting value, an interrupt will be generated to remind the user.

If the LVD module is enabled before hibernation, the hardware will not close the module circuit after entering hibernation, and the software needs to be closed (LVDEN=0).

0xF690	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
LVDCON	LVDSSEL3	LVDSSEL2	LVDSSEL1	LVDSSEL0	LVDEN	-	LVDINTE	LVDINTF
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~Bit4 LVDSSEL<3:0>: LVD voltage monitoring point;

0000=	2.00V;	1000=	Forbidden;
0001=	2.20V;	1001=	Forbidden;
0010=	Forbidden;	1010=	3.70V ;
0011=	2.40V;	1011=	Forbidden;
0100=	Forbidden;	1100=	4.00V;
0101=	2.70V;	1101=	4.30V;
0110=	Forbidden;	1110=	Forbidden;
0111=	3.00V;	1111=	Forbidden.

Bit3 LVDEN: LVD enable;  
 0= disable;  
 1= enable.

Bit2 - Reserved, must be zero.

Bit1 LVDINTE: LVD interrupt enable bit;  
 0= LVD interrupt disable;  
 1= LVD interrupt enable.

Bit0 LVDINTF: LVD interrupt flag bit;  
 0= power supply voltage is higher than the monitoring voltage;  
 1= power supply voltage is lower than the monitoring voltage (cleared by software).

## 5.3 IDLE idle mode

In this mode, only the CPU clock source is turned off. Therefore, in this state, peripheral functions (such as timers, PWM, and I<sup>2</sup>C) and clock generators (HSI/crystal oscillator drivers) still work normally.

After the system enters the idle mode, it can be awakened by any interrupt. After waking up, it enters the interrupt processing program. After the interrupt returns, it continues to execute the instruction after the sleep operation.

If you enter the idle mode in the interrupt service routine, the system can only be awakened by the interrupt with higher priority.

## 5.4 STOP sleep mode

In this mode, all circuits except LVD module and LSE module are closed (LVD/LSE module must be closed by software), the system is in low power consumption mode, and the digital circuits are not working.

### 5.4.1 Sleep wakeup After

entering the sleep mode, the sleep wakeup function can be turned on (SWE=1 must be set) to wake up the sleep mode. There are several ways to wake up the sleep mode:

- 1) INT0/1 interrupt  
uses the INT0/1 interrupt to wake up the sleep mode. The total interrupt enable and INT0/1 interrupt enable must be turned on before entering the sleep mode to wake up the system. INT0, INT1 interrupt related registers include IE, IP, TCON, IO multiplexing mapping register, INT0/1 interrupt wake-up can only be a falling edge interrupt wake-up sleep.
- 2) External (GPIO) interrupts  
are awakened by external GPIO interrupts. The total interrupt enable and port interrupt enable must be turned on before entering sleep to wake up the system. External GPIO interrupt wake-up can choose rising edge, falling edge, and dual-edge interrupt to wake up sleep. The interrupt wake-up edge is set by the external interrupt control register PxnEICFG.
- 3) WUT timing wake-up periodically awakened  
is by WUT. The timing wake-up function must be turned on before entering sleep, and the time from sleep to wake-up must be set at the same time. The clock source of the timing wake-up circuit is provided by the LSI (low-power oscillator). When the timing wake-up function is turned on, the LSI is automatically turned on in the sleep state.
- 4) LSE timing wake-up periodically awakened  
is by LSE. The LSE module must be enabled, counting enable, and timing wake-up functions must be turned on before entering sleep. At the same time, the time from sleep to wake-up must be set.

### 5.4.2 Wake-up wait state

whether INT0 / 1 interrupt, external interrupt GPIO, or WUT timed wake-up, LSE timed wake-sleep mode, an interrupt is generated or after the regular time, we need to wait for some time to wake up the system, the next instruction execution of the program. After the interrupt is generated or the time is up, the system oscillator is started, but the oscillation frequency is not stable yet, the CPU is not working, and the PC still stops in the dormant state. The system needs to wait for a period of time before providing the clock to the CPU. The waiting time for waking up the CPU is set in the programming CONFIG, and the waiting time can be set to 50us~1s. After the wake-up waiting time has elapsed, the MCU considers that the system clock is stable, and then provides the clock to the CPU, and the program continues to execute.

If the internal wake-up timer and external interrupt wake-up function are both enabled, after the system enters the sleep mode, any wake-up method can wake up the CPU. If the internal timer wakes up the oscillator first, and then there is an external interrupt input, after the wake-up wait time has elapsed, the program executes the interrupt handler first and then continues to execute the instruction after the sleep operation.

### 5.4.3 Sleep wake-up time

Use external interrupts to wake up the system. The total wake-up time is:

power manager stability time (200us) + wake-up waiting time.

The total wake-up time of the system using timing wake-up is:

power manager stable time (200us) + wake-up timer timing + Wake-up waiting time

(the time given above is  $F_{sys} > 1\text{MHz}$ )



### 5.4.4 Reset operation

In sleep mode, the system can also be restarted by power-off reset or external reset. This restart method has nothing to do with the value of SWE, even if SWE= 0 The system can also be restarted by the above reset operation.

**Power-down reset:** No other conditions are required. After VDD drops to 0V, power on again to the working voltage and enter the power-on reset state.

**External reset:** The external reset function needs to be turned on, and the related ports are configured as dedicated reset ports. The reset port maintains a low level of >1us during sleep, and the system generates a reset. Release the reset port, and the system restarts.

### 5.4.5 Sleep power consumption in debug mode

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

In the debug mode, after the system enters the sleep state, the related power management circuit and oscillator do not turn off, but continue to turn on. The wake-up operation can also be performed in the debug mode, and the wake-up method is the same as that in the normal mode.

Therefore, in this state, the sleep current obtained by the test is not the true sleep power consumption. It is recommended to close the debug mode after the development of the sleep wake-up function in the debug mode, and then restart the system. The measured current at this time is the actual sleep power consumption..

### 5.4.6 Sleep mode application: example

Before the system enters the sleep mode, if the user needs to obtain a smaller sleep current, please confirm the status of all I/Os. If there are floating I/O ports in the user program, set all floating ports to Output ports, ensure that each input port has a fixed state to avoid when the I/O is in the input state, the port line level is in an uncertain state and increases the sleep current; turn off the ADC module, LSE module, LVD module and others Peripherals to reduce sleep current.

Example: When using timing wake-up, enter sleep processing program (assembler)

```

SLEEP_MODE:
    MOV                WUTCRL,#31h
    MOV                WUTCRH,#80h
    MOV                P0TRIS,#0FFh
    MOV                P0,#0FFh
    MOV                P1TRIS,#0FFh
    MOV                P1,#0FFh
    MOV                P2TRIS ,#0FFh
    MOV                P2,#0FFh
    Operation instructions to turn off
    other functions
    MOV                PCON,#06H;   Perform functional wake-up sleep operation,
    NOP
    NOP
    NOP
    NOP
    NOP
    NOP
    ;The instruction to execute the sleep operation must be followed
    ;by 6 NOP instructions.
    Other Instructions after wake- up
    
```

## 6. Interrupt

### 6.1 Interrupt Overview

chips having 20 interrupt sources and vector:

interrupt source	Interrupt described	interrupt vector	priority order within same 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	timer 2 interrupt	5-0x002B	6
UART1	TI1 or RI1	6-0x0033	7
P0EXTIF<7:0>	P0 port external interrupt	7-0x003B	8
P1EXTIF<7:0>	P1 port external interrupt	8-0x0043	9
P2EXTIF<7:0>	P2 port external interrupt	9-0x004B	10
--	--	10-0x0053	11
--	--	11-0x005B	12
P5EXTIF<7:0>	P5 port external interrupt	12-0x0063	13
--	--	13-0x006B	14
--	--	14-0x0073	15
Timer3	timer 3 interrupt	15-0x007B	16
Timer4	Timer 4 Interrupt	16-0x0083	17
--	--	17-0x008B	18
PWM	PWM interrupt	18-0x0093	19
ADC	ADC interrupt	19-0x009B	20
WDT	WDT Interrupt	20-0x00A3	21
I <sup>2</sup> C	I <sup>2</sup> C Interrupt	21-0x00AB	22
SPI	SPI Interrupt	22-0x00B3	23
--	--	23-0x00BB	24
--	--	24-0x00C3	25
LSE_Timer/SCM	LSE timer interrupt/SCM	25-0x00CB	26
LVD	LVD power-down interrupt	26-0x00D3	27
--	--	27-0x00DB	28

The LSE Timer LSE\_Timer interrupt and oscillator stop detection SCM interrupt share the same interrupt vector entry, but they each have their independent interrupt enable bit.

The chip stipulates two interrupt priority levels, which can realize two-level interrupt nesting. When an interrupt has been responded, if a high-level interrupt sends a request, the latter can interrupt the former to achieve interrupt nesting.

## 6.2 External interrupt

### 6.2.1 INT0/INT1 interrupt

The chip supports 8051 native INT0 and INT1 external interrupts. INT0/INT1 can select falling edge or low level to trigger the interrupt. The relevant control register is TCON. INT0 and INT1 occupy two interrupt vectors.

### 6.2.2 GPIO interrupt

Each GPIO pin of the chip supports external interrupts, and can support falling edge/rising edge/double edge interrupts. The edge trigger type is configured through the PxNEICFG register. For example, configure port P12 as a falling edge interrupt:

```
P12CFG=0x00;      //Set P12 as GPIO
P1TRIS&=0xFB;    //Set P12 as input port
P12EICFG=0x02;   //Set P12 as falling edge trigger interrupt
```

GPIO interrupt totally occupies 4 interrupt vectors:

```
Port 0 occupies an interrupt vector 0x003B;
Port P1 occupies an interrupt vector 0x0043;
Port P2 occupies an interrupt vector 0x004B;
Port P5 occupies an interrupt vector 0x0063;
```

When the interrupt is generated, enter the interrupt service routine to determine which port triggered the interrupt first, and then perform the respective processing.

## 6.3 Interrupt and sleep wake-up

After the system enters the sleep mode (STOP can wake up mode), each external interrupt can be set to wake up the system.

The INT0/INT1 interrupt wake-up system needs to turn on the corresponding interrupt enable and the overall interrupt enable, and the wake-up mode is falling edge wake-up (INT0/INT1 wake-up mode and interrupt trigger mode selection bit IT0/IT1 are irrelevant).

GPIO interrupt wakes up the system, it is recommended to set the corresponding port interrupt trigger edge mode before entering sleep mode (the wake-up mode of GPIO is the same as the interrupt trigger edge mode, you can choose the rising edge/falling edge/double edge wakeup), and turn on the corresponding interrupt Enable and general interrupt enable.

After the system is awakened by an external interrupt, it first enters the interrupt service routine to process the interrupt wake-up task. After exiting the interrupt service routine, the system continues to execute the instructions after the sleep operation.

## 6.4 Interrupt Register

### 6.4.1 Interrupt Mask Register

#### 6.4.1.1 Interrupt Mask Register IE

Interrupt Mask Register IE is a readable and writable register and can be operated in bits. When an interrupt condition occurs, regardless of the status of the corresponding interrupt enable bit or global enable bit EA, the interrupt flag bit will be set to 1. User software should ensure that the corresponding interrupt flag bit is cleared before enabling an 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= allow all unmasked interrupts; 0= disable all interrupts.
Bit6	ES1: UART1 interrupt enable bit; 1= enable UART1 interrupt; 0= disable UART1 interrupt.
Bit5	ET2: TIMER2 total interrupt enable bit; 1= Enable all TIMER2 interrupts; 0= Disable all TIMER2 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; TIMER0 0= Disable interrupt.
Bit0	EX0: External interrupt 0 interrupt enable bit; 1= Enable external interrupt 0 interrupt; 0= Disable external interrupt 0 interrupt.

**6.4.1.2 Interrupt mask register EIE2**

0xAA	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIE2	SPIIE	I2CIE	WDTIE	ADCIE	PWMIE	--	ET4	ET3
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<sup>2</sup>C interrupt enable bit;  
             1= Enable I<sup>2</sup>C interrupt;  
             0= Disable I<sup>2</sup>C interrupt.
- Bit5      WDTIE: WDT interrupt enable bit;  
             1= Enable WDT overflow interrupt;  
             0= Disable WDT overflow interrupt.
- Bit4      ADCIE: ADC interrupt enable bit;  
             1= enable ADC interrupt;  
             0= disable ADC interrupt.
- Bit3      PWMIE: PWM total interrupt enable bit;  
             1= allow all PWM interrupts;  
             0= disable all PWM interrupts.
- Bit2      - Reserved, must be zero.
- Bit1      ET4: Timer4 interrupt enable bit;  
             1= Enable Timer4 interrupt;  
             0= Disable Timer4 interrupt.
- Bit0      ET3: Timer3 interrupt enable bit;  
             1= Enable Timer3 interrupt;  
             0= Disable Timer3 interrupt.

**6.4.1.3 Timer2 Interrupt mask register T2IE**

0xCF	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
T2IE	T2OVIE	T2EXIE	--	--	T2C3IE	T2C2IE	T2C1IE	T2C0IE
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	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 ~ Bit4	-	Reserved, must be 0's.
Bit3	T2C3IE:	Timer2 compare channel 3 interrupt enable bit; 1= Enable interrupt; 0= Disable interrupt.
Bit2	T2C2IE:	Timer2 compare channel 2 interrupt enable bit; 1= Enable interrupt; 0= Disable interrupt.
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 interrupt of Timer2 is enabled, the total interrupt enable bit of Timer2 ET2=1 (IE.5=1)

**6.4.1.4 P0 port Interrupt control register P0EXTIE**

0xAC	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
P0EXTIE	P07IE	P06IE	P05IE	P04IE	P03IE	P02IE	P01IE	P00IE
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	P0iIE:	P0i port interrupt enable bit (i=0-7); 1= Enable Interrupt; 0= Disable Interrupt.
-----------	--------	---

**6.4.1.5 P1 port Interrupt control register P1EXTIE**

0xAD	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
P1EXTIE	P17IE	P16IE	P15IE	P14IE	P13IE	P12IE	P11IE	P10IE
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      P1iIE: P1i port Interrupt enable bit (i=0-7) ;  
 1= Enable interrupt;  
 0= Disable interrupt.

**6.4.1.6 P2 port Interrupt control register P2EXTIE**

0xAE	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
P2EXTIE	P27IE	P26IE	P25IE	P24IE	P23IE	P22IE	P21IE	P20IE
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      P2iIE: P2i port Interrupt enable bit (i=0-7) ;  
 1= Enable interrupt;  
 0= Disable interrupt.

**6.4.1.7 P5 port Interrupt control register P5EXTIE**

0x9C	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
P5EXTIE	P57IE	--	P55IE	P54IE	P53IE	P52IE	P51IE	P50IE
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      -- reserved, shall all be 0.  
 Bit5~Bit0      P5iIE: P5i port interrupt enable bit (i=0-5) ;  
 1= Enable Interrupt;  
 0= Disable Interrupt.

## 6.4.2 Interrupt Priority Control register

### 6.4.2.1 Interrupt Priority Control register IP

Interrupt Priority Control register IP is a readable and writable register, which can be operated by bit.

0xB8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
IP	--	PS1	PT2	PS0	PT1	PX1	PT0	PX0
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, must be 0.
- Bit6            PS1: UART1 interrupt priority control bit;  
                   1=    Set to high-level interrupt;  
                   0=    Set as low-level interrupt.
- Bit5            PT2: TIMER2 interrupt priority control bit;  
                   1=    Set to high-level interrupt;  
                   0=    Set as low-level interrupt.
- Bit4            PS0: UART0 interrupt priority control bit;  
                   1=    Set to high-level interrupt;  
                   0=    Set as low-level interrupt.
- Bit3            PT1: TIMER1 interrupt priority control bit;  
                   1=    Set to high-level interrupt;  
                   0=    Set as low-level interrupt.
- Bit2            PX1: External interrupt 1 interrupt priority control bit;  
                   1=    Set to high-level interrupt;  
                   0=    Set as low-level interrupt.
- Bit1            PT0: TIMER0 interrupt priority control bit;  
                   1=    Set to high-level interrupt;  
                   0=    Set as low-level interrupt.
- Bit0            PX0: External interrupt 0 interrupt priority control bit;  
                   1=    Set to high-level interrupt;  
                   0=    Set as low-level interrupt.

### 6.4.2.2 Interrupt priority control register EIP1

0xB9	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIP1	--	--	PP5	--	--	PP2	PP1	PP0
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, must be 0.
- Bit6            --    Reserved, must be 0.
- Bit5            PP5: P5 interrupt priority control bit;  
                   1=    Set as high-level interrupt;  
                   0=    Set as low-level interrupt.
- Bit4~Bit3       --    Reserved, must be 0
- Bit2            PP2: P2 interrupt priority control bit;  
                   1=    Set as high-level interrupt;  
                   0=    Set as low-level interrupt.
- Bit1            PP1: P1 interrupt priority control bit;



1= Set as high-level interrupt;  
 0= Set as low-level interrupt.  
 Bit0 PP0: P0 interrupt priority control bit;  
 1= Set as high-level interrupt;  
 0= Set as low-level interrupt.

#### 6.4.2.3 Interrupt priority control register EIP2

0xBA	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIP2	PSPI	PI2C	PWDT	PADC	PPWM	--	PT4	PT3
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 PSPI: SPI interrupt priority control bit;  
 1= Set as high-level interrupt;  
 0= Set as low-level interrupt.  
 Bit6 PI2C: I2C interrupt priority control bit;  
 1= Set as high-level interrupt;  
 0= Set as low-level interrupt.  
 Bit5 PWDT: WDT interrupt priority control bit;  
 1= Set as high-level interrupt;  
 0= Set as low-level interrupt.  
 Bit4 PADC: ADC interrupt priority control bit;  
 1= Set as high-level interrupt;  
 0= Set as low-level interrupt.  
 Bit3 PPWM: PWM interrupt priority control bit  
 1= Set as high-level interrupt;  
 0= Set as low-level interrupt.  
 Bit2 -- Reserved, must be 0.  
 Bit1 PT4: TIMER4 interrupt priority control bit;  
 1= Set as high-level interrupt;  
 0= Set as low-level interrupt.  
 Bit0 PT3: TIMER3 interrupt priority control bit;  
 1= Set as high-level interrupt;  
 0= Set as low-level interrupt.

**6.4.2.4 Interrupt priority control register EIP3**

0xBB	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIP3	--	--	--	--	PLVD	PLSE	--	--
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~Bit5            -- Reserved, must be 0.  
 Bit4                -- Reserved, must be 0.  
 Bit3                PLVD: LVD interrupt priority control bit;  
                       1= Set as high-level interrupt;  
                       0= Set as low-level interrupt.  
 Bit2                PLSE: LSE interrupt priority control bit;  
                       1= Set as high-level interrupt;  
                       0= Set as low-level interrupt.  
 Bit1~Bit0           -- Reserved, must be 0.

### 6.4.3 Interrupt Flag Register

#### 6.4.3.1 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 counter overflow interrupt flag bit;  
           1= Timer1 counter overflows, the hardware is automatically cleared when entering the interrupt service routine, or software Cleared;  
           0= Timer1 counter does not overflow.
- Bit6      TR1: Timer1 running control bit;  
           1= Timer1 is started;  
           0= Timer1 is closed.
- Bit5      TF0: Timer0 counter overflow interrupt flag bit;  
           1= Timer0 counter overflows, it is automatically cleared by hardware when entering the interrupt service routine, or can be cleared by software;  
           0= Timer0 counter has no overflow.
- Bit4      TR0: Timer0 running control bit;  
           1= Timer0 starts.  
           0= Timer0 is turned off.
- Bit3      IE1: External Interrupt 1 flag bit;  
           1= External Interrupt 1 generates an interrupt, which is automatically cleared by hardware when entering the interrupt service routine, or cleared by software;  
           0= External Interrupt 1 does not generate an interrupt.
- Bit2      IT1: External interrupt 1 trigger mode control bit;  
           1= falling edge trigger;  
           0= low level trigger.
- Bit1      IE0: External interrupt 0 flag;  
           1= External interrupt 0 generates an interrupt, which is automatically cleared by hardware when entering the interrupt service routine, or can be cleared by software;  
           0= External interrupt 0 does not generate an interrupt.
- Bit0      IT0: External interrupt 0 trigger mode control bit;  
           1= falling edge trigger;  
           0= low level trigger.

### 6.4.3.2 Timer2 Interrupt flag register T2IF

0xC9	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
T2IF	TF2	T2EXIF	--	--	T2C3IF	T2C2IF	T2C1IF	T2C0IF
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      TF2: Timer2 counter overflow interrupt flag bit;  
           1= Timer2 counter overflows and needs to be cleared by software;  
           0= Timer2 counter does not overflow.
- Bit6      T2EXIF: Timer2 external load flag bit;  
           1= Timer2 T2EX port generates a falling edge, which needs to be cleared by software;  
           0= -
- Bit5~Bit4      -- Reserved, all must be 0.
- Bit3      T2C3IF: Timer2 compare/capture channel 3 flag bit;  
           1= Timer2 compare channel 3 {CCH3:CCL3}={TH2:TL2} or capture channel 3 has a capture operation, which needs to be cleared by software.  
           0= -
- Bit2      T2C2IF: Timer2 compare/capture channel 2 flag bit;  
           1= Timer2 compare channel 2 {CCH2:CCL2}={TH2:TL2} or capture channel 2 has a capture operation, which needs to be cleared by software.  
           0= -
- Bit1      T2C1IF: Timer2 compare/capture channel 1 flag bit;  
           1= Timer2 compare channel 1 {CCH1:CCL1}={TH2:TL2} or capture channel 1 has a capture operation, which 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 a capture operation, which needs to be cleared by software.  
           0= -

### 6.4.3.3 Peripheral interrupt flag register EIF2

0xB2	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIF2	SPIIF	I2CIF	--	ADCIF	PWMIF	--	TF4	TF3
R/W	R	R	--	R/W	R	--	R/W	R/W
Reset Value	0	0	0	0	0	0	0	0

- Bit7      SPIIF: SPI general interrupt indicator bit, read-only;  
           1= SPI generates an interrupt (this bit is automatically cleared after clearing the specific interrupt flag bit);  
           0= SPI does not generate an interrupt.
- Bit6      I2CIF: I<sup>2</sup>C general interrupt indicator bit, read-only;  
           1= I<sup>2</sup>C generates an interrupt (after clearing the specific interrupt flag bit, this bit is automatically cleared);  
           0= I<sup>2</sup>C does not generate an interrupt.
- Bit5      -- Reserved, must be zero.
- Bit4      ADCIF: ADC interrupt flag bit;  
           1= ADC conversion is completed and needs to be cleared by software;  
           0= ADC conversion is not completed.
- Bit3      PWMIF: PWM general interrupt indicator bit, read-only;

- 1= PWM generates an interrupt, (this bit is automatically cleared after clearing the specific interrupt flag bit);
- 0= PWM does not generate an interrupt.
- Bit2 -- Reserved, must be zero.
- Bit1 TF4: Timer4 timer overflow interrupt flag bit;
  - 1= Timer4 timer overflows, it is automatically cleared by hardware when entering the interrupt service routine, or can be cleared by software;
  - 0= Timer4 timer has no overflow.
- Bit0 TF3: Timer3 timer overflow interrupt flag bit;
  - 1= Timer3 timer overflows, it is automatically cleared by hardware when entering the interrupt service routine, or it can be cleared by software;
  - 0= Timer3 timer has no overflow.

#### 6.4.3.4 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 transmission complete interrupt flag bit, read only;
  - 1= SPI transmission is complete (read SPSR first, then clear after reading/writing SPDR);
  - 0= The SPI transmission is not complete.
- Bit6 WCOL: SPI write conflict interrupt flag bit, read only;
  - 1= Write SPDR operation conflict occurs when SPI transmission is not completed (read SPSR first, then clear after reading/writing SPDR);
  - 0= No write conflicts.
- Bit5~Bit1 -- Reserved, all must be 0.
- Bit0 SSCEN: SPI master mode NSS output control bit.
  - 1= When SPI is in idle state, NSS outputs high level;
  - 0= NSS outputs the contents of register SSCR.

#### 6.4.3.5 I<sup>2</sup>C master mode interrupt flag register I2CMCR/I2CMSR

0xF5	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
I2CMCR	RSTS	--	--	--	ACK	STOP	START	RUN
I2CMSR	I2CMIF	BUS_BUSY	IDLE	ARB_LOST	DATA_ACK	ADDR_ACK	ERROR	BUSY
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 RSTS: I<sup>2</sup>C active module reset control bit;
  - 1= reset the main control module (the entire The Imain control module<sup>2</sup>C register of the, including I2CMSR);
  - 0= The I<sup>2</sup>C interrupt flag bit is cleared to 0 in the I<sup>2</sup>C main control mode.
- I2CMIF: I<sup>2</sup>C interrupt flag bit in I<sup>2</sup>C master control mode;
  - 1= In master control mode, transmission/reception is completed, or a transmission error occurs. (Cleared by software, cleared by writing 0);
  - 0= no interrupt is generated.
- Bit6~Bit0 I<sup>2</sup>C control and flag bits in I<sup>2</sup>C master control mode, please refer to I2CM description for details.

### 6.4.3.6 I<sup>2</sup>C slave mode status register I2CSSR

0xF2	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
I2CSSR	--	--	--	--	--	SENDFIN	TREQ	RREQ
R/W	--	--	--	--	--	R	R	R
Reset Value	0	0	0	0	0	0	0	0

Bit7~Bit3 -- reserved, must be 0's.

Bit2 SENDFIN: I<sup>2</sup>C sending operation completed flag in C slave mode, read-only;  
 1= The main control device no longer needs data, TREQ is no longer set to 1, and the data transmission has been completed. (It is automatically cleared after reading I2CSCR).  
 0= -

Bit1 TREQ: I<sup>2</sup>C slave mode ready to send flag bit, read-only;  
 1= As the sending device has been addressed or the master device is ready to receive data. (It is automatically cleared after writing I2CSBUF).  
 0= -

Bit0 RREQ: I<sup>2</sup>C slave mode reception completion flag bit, read only;  
 1= reception completed. (It is automatically cleared after reading I2CSBUF);  
 0= Not received.

I<sup>2</sup>C The relevant status bits of C slave mode are also interrupt flag bits.

Note: I<sup>2</sup>C master mode interrupts and slave mode interrupts share the same interrupt vector (00ABH).

### 6.4.3.7 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

BANK0: register SCON0 address 0x98; register SCON1 address is 0xEA

Bit7~Bit2 U1SM0, U1SM1, U1SM2, U1REN, U1TB8, U1RB8: UART1 related control bits, see UARTn function description for details

Bit1 TIn: Transmit interrupt flag bit (requires software to clear);  
 1= Indicates that the sending buffer is empty, you can send the frame data  
 0= -

Bit0 RIn: Receive interrupt flag bit (requires software to clear);  
 1= Indicates that the receiving buffer is full, and the next frame of data can be received after reading  
 0= -

TIn and RIn occupy the same interrupt vector, and need to be inquired to determine whether it is a receiving interrupt or a sending interrupt.

#### 6.4.3.8 P0 port interrupt flag register P0EXTIF

0xB4	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
P0EXTIF	P07IF	P06IF	P05IF	P04IF	P03IF	P02IF	P01IF	P00IF
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 P0iIF: P0i port interrupt flag bit (i=0-7);  
 1= P0i port generates an interrupt, which needs to be cleared by software;  
 0= No interrupt is generated at port P0i.

#### 6.4.3.9 P1 port Interrupt flag register P1EXTIF

0xB5	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
P1EXTIF	P17IF	P16IF	P15IF	P14IF	P13IF	P12IF	P11IF	P10IF
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 P1iIF: P1i port interrupt flag bit (i=0-7);  
 1= P1i port generates an interrupt, which needs to be cleared by software;  
 0= No interrupt is generated on port P1i.

#### 6.4.3.10 P2 port Interrupt flag register P2EXTIF

0xB6	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
P2EXTIF	P27IF	P26IF	P25IF	P24IF	P23IF	P22IF	P21IF	P20IF
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 P2iIF: P2i port interrupt flag bit (i=0-7);  
 1= P2i port generates an interrupt and needs to be cleared by software;  
 0= P2i port does not generate an interrupt.

#### 6.4.3.11 P5 port Interrupt flag register P5EXTIF

0xA7	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
P5EXTIF	P57IF	--	P55IF	P54IF	P53IF	P52IF	P51IF	P50IF
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 P5iIF P5i port interrupt flag bit (i=7);  
 1= P5i port generates an interrupt and needs to be cleared by software;  
 0= P5i port does not generate an interrupt.

Bit6 -- reserved, shall be 0

Bit5~Bit0 P5iIF: P5i port interrupt flag bit (i=0-5);  
 1= P5i port generates an interrupt and needs to be cleared by software;  
 0= P5i port does not generate an interrupt.

#### 6.4.4 Clear operation of interrupt flag

Interrupt flag bit is divided into the following types:

- ◆ hardware automatic clear (need to enter the interrupt service program)
- ◆ software clear
- ◆ read/write operation clear

1) Flag bit automatically cleared by hardware

The bits that support automatic hardware clearing are the interrupt flag bits generated by INT0, INT1, T0, T1, T3, and T4. The condition for the hardware to automatically clear the flag is: turn on the total interrupt enable bit EA=1, and turn on the corresponding interrupt enable bit. After the interrupt is generated, the system enters the corresponding interrupt service routine, and the flag bit is automatically cleared. If the interrupt enable is turned off, these flags can also be cleared by software.

2) Flag bit cleared by software

There are flag bits in the system that can only be cleared by software. These flags will not be automatically cleared after entering the interrupt service routine, and need to be cleared by software by writing 0. Otherwise, after exiting the interrupt service program, it will enter the interrupt service program again.

3) Flag bits cleared by read and write operations

There is a flag bit in the system, instead of writing 0 to clear the flag bit, you need to read/write other registers to clear the flag bit. For example, the transfer complete flag bit SPISIF in the SPI interrupt flag register, after setting it to 1, you need to read SPSR first, and then clear it after reading/writing SPDR.

The software clearing operation requires attention: when multiple interrupt flags are in the same register and the moments when these flags are generated are not related to each other, it is not recommended to use read-modify-write operations. For example, the PWMUIF interrupt flag bit register contains the upward comparison interrupts of the PG0-PG5 channels. These interrupt flag bits are not related to each other. When PG0 generates an upward comparison interrupt, the value of PWMUIF is 0x01. After entering the interrupt service routine, perform a read-modify-write operation to clear the bit

```
PWMUIF &= 0xFE;
```

This operation is specifically implemented by first reading the value of PWMUIF back to the CPU, and then perform calculations again, and finally send back to PWMUIF. If the interrupt flag bit PWMUIF[1] of PG1 is set to 1 after the CPU is read, and PWMUIF[1] is 0 when it is read, it will be sent back to PWMUIF[1] after the calculation is also 0, and PG1 will be cleared at this time. The up interrupt flag that has been generated is PWMUIF[1].

To clear the above type of interrupt flag bit, it is recommended to write 0 directly, and write 1 to other irrelevant flag bits: PWMUIF = 0xFE. This operation has no actual effect on writing 1 to the irrelevant interrupt flag.



### 6.4.5 Special interrupt flag in debug mode

There is a flag bit in the system, instead of writing 0 to clear the flag bit, you need to read/write other registers to clear the flag bit.

In the debug state, after the breakpoint is executed, the single-step operation or the stop operation, the emulator will read all the register values from the system to the simulation software. The read/write operation of the emulator is completely the same as that in the normal mode. Same.

Therefore, during the debugging process, after a pause, the interrupt flag bit set to 1 should appear, but it is displayed as 0 in the observation window.

Example: The transmission completion flag bit SPISIF in the SPI interrupt flag register in debug mode

```
...                               //Set port and interrupt enable
SPDR = 0x56;                       //Send SPDR data
delay();
...

void SPI_int (void) interrupt SPI_VECTOR // SPI interrupt service routine
{
    O1 _nop_();                     //Set breakpoint 1
    _nop_();
    O2 k = SPSR;                     //Set breakpoint 2
    _nop_();
    ...
}
```

When the breakpoint is running, after stopping at breakpoint 1, the SPI completes the transmission operation, and the transmission completion interrupt has been generated, so SPSR.7=1, at this time the emulator has completed the operation of reading all registers (including reading SPSR).

Execute breakpoint operation again and stop at breakpoint 2. At this time, the emulator once again completes the operation of reading all registers (including SPDR), so SPSR.7=0 at this time. The above situation will also occur when single stepping twice, so you need to pay attention to it in debug mode.

## 7. I/O port

### 7.1 GPIO function

Chip has four groups of I/O ports: PORT0, PORT1, PORT2, PORT5.

PORTx is a bidirectional port. Its corresponding data direction register is PxTRIS. Set a bit of PxTRIS to 1 (=1) to configure the corresponding pin as an output. Clear a bit of PxTRIS (= 0) to configure the corresponding PORTx pin as an input.

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

When PORTx is used as an output port, reading the Px register is related to the setting of the PxDS register. A bit of PxDS is 1 (=1), the corresponding bit of Px read is the status of the pin, a bit of PxDS is cleared (=0), the corresponding bit of Px read is the state of the port data latch; When the input port is used, the status of the pin is read by reading the Px register, which has nothing to do with the setting of the PxDS register.

When using the PORTx pin as an analog input, the user must ensure that the bits in the PxTRIS register remain set to 0. I/O pins configured as analog inputs always read 0.

The registers related to PORTx include Px, PxTRIS, PxOD, PxUP, PxRD, PxDS, etc.

#### 7.1.1 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 P5 address: 0xD8.

Bit7~Bit0 Px<7:0>: Px I/O pin bits;  
 1= Port pin level > V<sub>IH</sub> (positive threshold voltage);  
 0= Port pin level < V<sub>IL</sub> (negative threshold voltage).

#### 7.1.2 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: 0xA1; Register P2TRIS address: 0xA2;

Register P5TRIS address: 0xA5.

Bit7~Bit0 PxTRIS<7:0>: Three-state control bits;  
 1= The pin is configured as an output;  
 0= The pin is configured as an input (three-state).

#### Note:

- 1) After the port is set as an output port, the data read from the port is the value of the output register.
- 2) After the port is set as an input port, the <read-modify-write> type of instruction for the port is actually an operation on the output register.

### 7.1.3 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: F009H; Register P1OD address: F019H; Register P2OD address: F029H;

Register P5OD address: F059H.

Bit7~Bit0 PxOD<7:0>: Open-drain control bits;  
 1= The pin is configured as an open-drain state (the output is an open-drain output);  
 0= The pin is configured as a normal state (the output is a push-pull output).

### 7.1.4 PORTx pull-up resistor control 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: F00AH; Register P1UP address: F01AH; Register P2UP address: F02AH;

Register P5UP address: F05AH.

Bit7~Bit0 PxUP<7:0>: Pull-up resistor control bit;  
 1= Pin pull-up resistor is on;  
 0= Pin pull-up resistor is off.

### 7.1.5 PORTx pull-down resistance control 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: F00BH; Register P1RD address: F01BH; Register P2RD address: F02BH;

Register P5RD address: F05BH.

Bit7~Bit0 PxRD<7:0>: Pull-down resistor control bit;  
 1= Pin pull-down resistor is on;  
 0= Pin pull-down resistor is off.

Note: The control of the pull-down resistor has nothing to do with the configuration of GPIO and the multiplexing function, and is controlled separately by the PxRD register.

### 7.1.6 PORTx Slope Control Register PxSR

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PxSR	PxSR7	PxSR6	PxSR5	PxSR4	PxSR3	PxSR2	PxSR1	PxSR0
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 P0SR address: F00DH; Register P1SR address: F01DH; Register P2SR address: F02DH;

Register P5SR address: F05DH.

Bit7~Bit0 PxSR<7:0>: Px slope control register (valid when the port is configured as output state);  
 1= Px pin is slow slope;  
 0= Px pin is fast slope.

### 7.1.7 PORTx data input selection register PxDS

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PxDS	PxDS7	PxDS6	PxDS5	PxDS4	PxDS3	PxDS2	PxDS1	PxDS0
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 P0DS address: F00EH; Register P1DS address: F01EH; Register P2DS address: F02EH;

Register P5DS address: F05EH.

Bit7~Bit0 PxDS<7:0>: Data input selection bit, when configured as GPIO, it affects the reading of the value of Px register;  
 1= output/input mode reads the pin status;  
 (Smit when the port is set to output state The circuit also remains open);  
 0= output mode: read as the data latch status;  
 input mode: read as the pin status.

Note: If you need to read the pin status when the port is a multiplex function input structure, you need to set the port direction control to input mode.

## 7.2 Multiplexed function

### 7.2.1 Port multiplexed function table

Pins share multiple functions, and each I/O port can be flexibly configured with digital functions or designated analog functions. The digital function of the external input is selected by the port input function allocation register (PS\_XX); the multiplexing function is selected by the port multiplexing function configuration register (PxnCFG), and the communication input function is also selected by the communication input function allocation register (PS\_XX) designation.

Digital functional configuration shown in the following table:

	External input	Digital Function configuration							
		0	1	2	3	4	5	6	7
P00	-	GPIO	ANA	SCL	SCLK	CC0	PG0	TXD1	RXD1
P01	-	GPIO	ANA	SDA	MOSI	CC1	PG1	TXD1	RXD1
P02	T0G/T1G/T2EX	GPIO	ANA	-	MISO	CC2	PG2	TXD1	RXD1
P03	T0/T1/T2	GPIO	ANA	-	NSS(NSSO0)	CC3	PG3	TXD1	RXD1
P04	-	GPIO	ANA	TXD0	NSS(NSSO1)	-	PG4	TXD1	RXD1
P05	-	GPIO	ANA	RXD0	NSS(NSSO2)	-	PG5	TXD1	RXD1
P06	ADET/INT0	GPIO	ANA	-	NSS(NSSO3)	BUZZ	-	TXD1	RXD1
P07	INT0/INT1	GPIO	ANA	-	-	CLO	-	TXD1	RXD1
P10	-	GPIO	ANA	SCL	SCLK	-	PG0	TXD1	RXD1
P11	-	GPIO	ANA	SDA	MOSI	-	PG1	TXD1	RXD1
P12	T0G/T1G/T2EX	GPIO	ANA	-	MISO	-	PG2	TXD1	RXD1
P13	T0/T1/T2	GPIO	ANA	-	NSS(NSSO0)	-	PG3	TXD1	RXD1
P14	CAP0	GPIO	ANA	TXD0	NSS(NSSO1)	-	PG4	TXD1	RXD1
P15	CAP1	GPIO	ANA	RXD0	NSS(NSSO2)	-	PG5	TXD1	RXD1
P16	CAP2	GPIO	ANA	-	NSS(NSSO3)	BUZZ	-	TXD1	RXD1
P17	CAP3	GPIO	ANA	-	-	CLO	-	TXD1	RXD1
P20	-	GPIO	ANA	SCL	SCLK	CC0	PG0	TXD1	RXD1
P21	-	GPIO	ANA	SDA	MOSI	CC1	PG1	TXD1	RXD1
P22	T0G/T1G/T2EX	GPIO	ANA	-	MISO	CC2	PG2	TXD1	RXD1
P23	T0/T1/T2	GPIO	ANA	-	NSS(NSSO0)	CC3	PG3	TXD1	RXD1
P24	-	GPIO	ANA	TXD0	NSS(NSSO1)	-	PG4	TXD1	RXD1
P25	-	GPIO	ANA	RXD0	NSS(NSSO2)	-	PG5	TXD1	RXD1
P26	ADET/INT0	GPIO	ANA	-	NSS(NSSO3)	-	-	TXD1	RXD1
P27	INT0/INT1	GPIO	ANA	-	-	-	-	TXD1	RXD1
P50	NSRT	GPIO	ANA	SCL	SCLK	-	PG0	TXD1	RXD1
P51	-	GPIO	ANA	SDA	MOSI	-	PG1	TXD1	RXD1
P52	-	GPIO	ANA	-	MISO	-	PG2	TXD1	RXD1
P53	-	GPIO	ANA	-	NSS(NSSO0)	-	PG3	TXD1	RXD1
P54	-	GPIO	ANA	TXD0	NSS(NSSO1)	-	PG4	TXD1	RXD1
P55	-	GPIO	ANA	RXD0	NSS(NSSO2)	-	PG5	TXD1	RXD1
P57	-	GPIO	ANA			-	PG5	TXD1	RXD1

Analog modue CONFIG Port configuration shown in the following table:

	ANA(1)						CONFIG
	ADC						
P00	AN0						-
P01	AN1						-
P02	AN2						DSDA2
P03	AN3						DSCCK2
P04	AN4						-
P05	AN5						-
P06	AN6						
P07	AN7						
P10	AN8						-
P11	AN9						OSCIN2
P12	AN10						OSCOU2
P13	AN11						
P14	AN12						
P15	AN13						
P16	AN14						-
P17	AN15						
P20	AN16						-
P21	AN17						-
P22	AN18						-
P23	AN19						
P24	AN20						
P25	AN21						-
P26	AN22						
P27	AN23						
P50	AN39						NRST
P51	AN40						OSCIN1
P52	AN41						OSCOU1
P53	AN42						DSDA1
P54	AN43						DSCCK1
P55	AN44						
P57	AN46						-

## 7.2.2 Port Multiplexing Configuration Register Function

Function Configuration Register PORTx

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PxnCFG	--	--	--	--	--	PxnCFG2	PxnCFG1	PxnCFG0
R/W	--	--	--	--	--	R/W	R/W	R/W
Reset value	0	0	0	0	0	0	0	1

Bit7~Bit3 -- reserved, must be 0.

Bit2~Bit0 PxnCFG<2:0>: Function configuration bits, the default is analog function. See the port function configuration description for details;

000= GPIO function;

001= analog function (ANA);

Others= Multiplex function;

PxnCFG x=0/1/2/5, n=0-7

Each port has a function configuration register PxnCFG, through which each port can be set to the corresponding digital function. For example, to set P16 as BEEP buzzer function, configure P16CFG=0x04;

The port direction register PxTRIS does not need to be configured when the port is used as a multiplexing function.

- SCL and SDA pull-up resistance registers can be configured to force open drain output.

- RXD0 and RXD1 synchronization modes force pull-up.

Other multiplexing functions are forced to close the pull-up resistance and open drain output by hardware, that is, set the pull-up resistance PxUP or open drain output PxOD by software to be invalid.

When the port is multiplexed into SCL and SDA functions, the hardware forces the port to be open-drain output, and the pull-up resistance PxUP can be set through software.

## 7.2.3 Port input function allocation register

There are digital functions with only input status inside the chip, such as INT0/INT1... etc. This type of digital input function has nothing to do with the port multiplexing status. As long as the assigned port supports digital input (such as RXD0 as a digital input and GPIO as an input function), the port supports this function.

The input function port allocation register is as follows:

Register	address	function	description
PS_INT0	F0C0H	INT0	external interrupt 0 input port allocation register
PS_INT1	F0C1H	INT1	external interrupt 1 input port allocation register
PS_T0	F0C2H	T0	Timer0 external clock input port allocation register
PS_T0G	F0C3H	T0G	Timer0 gated input port Assign register
PS_T1	F0C4H	T1	Timer1 external clock input port assignment register
PS_T1G	F0C5H	T1G	Timer1 gate control input port assignment register
PS_T2	F0C6H	T2	Timer2 external event or gate control input port assignment register
PS_T2EX	F0C7H	T2EX	Timer2 falling edge automatic reload input port assignment register
PS_HCAP0	F0C8	CAP0	Timer2 input capture channel 0 port allocation register
PS_CAP1	F0C9H	CAP1	Timer2 input capture channel 1 port allocation register
PS_CAP2	F0CAH	CAP2	Timer2 input capture channel 2 port allocation register
PS_CAP3	F0CBH	CAP3	Timer2 input capture channel 3 port allocation register

PS_ADET	F0CCH	ADET	ADC external trigger input Port allocation register
---------	-------	------	---

PS\_XX input function port allocation register PS\_XX (as described in the above table)

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PS_XX	--	PS_XX6	PS_XX5	PS_XX4	PS_XX3	PS_XX2	PS_XX1	PS_XX0
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 -- reserved, must be 0.  
 Bit6 ~ Bit0 PS\_XX <6: 0 function assignment control bit input  
 (subject to the actual port chip, the value of the unused reserved, prohibited to be used);  
 0x00 = P00 assigned to the port;  
 0x01 = assigned to the port P01;  
 .....  
 0x20 = assigned to port P20;  
 0x21 = assigned to port P21;  
 .....  
 0x50 = assigned to port P50;  
 .....  
 0xFF = not assigned to port;

- 1) If multiple ports are configured for the same digital function at the same time, the priority will decrease in the order of P00, P01,..., P54, P57. If P02 and P12 are configured as T1 functions at the same time, the P02 configuration is valid and the P12 configuration is invalid.
- 2) The input function assignment structure supports multiple input functions assigned to the same port. For example, T0 and T1 can be allocated to port P03 at the same time, the configuration is as follows:  

```
P03CFG = 0x00; //P03 port is configured as GPIO function
P0TRIS = 0x00; //P03 is used as GPIO input function
PS_T0 = 0x03; //P03 port is configured as T0 function.
PS_T1 = 0x03; //P03 port is configured as T1 function.
```
- 3) The input function configuration structure can also be used simultaneously with the port external interrupt function. If you can assign T1G and GPIO interrupt functions to port P02 at the same time, the configuration is as follows:  

```
P00CFG = 0x00; //P02 port is configured as GPIO function
P0TRIS = 0x00; //P02 is used as GPIO input function
PS_T1G = 0x00; //P02 port is configured as T1G function
P02EICFG = 0x01; //P02 port is configured as rising edge trigger interrupt
P0EXTIE = 0x04; //Enable P02 port external interrupt
```



## 7.2.4 Communication input function allocation register

Port as the communication port (UART0/UART1/SPI/IIC), there are multiple input ports to choose from, and different port inputs can be selected by setting the following registers.

The communication input function port allocation register is as follows:

Register	address	function	description
PS_SCLK	F698H	SCLK	SPI clock port allocation register
PS_MOSI	F699H	MOSI	SPI slave input port allocation register
PS_MISO	F69AH	MISO	SPI master input port allocation register
PS_NSS	F69BH	NSS	SPI slave chip select input port allocation register
PS_SCL	F69CH	SCL	IIC clock input port allocation register
PS_SDA	F69DH	SDA	IIC data input port allocation register
PS_RXD1	F69EH	RXD1	UART1 data input port allocation register
PS_RXD0	F69FH	RXD0	UART0 data input port allocation register

Communication input function port distribution register PS\_XX (as described in the above table)

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PS_XX	--	PS_XX6	PS_XX5	PS_XX4	PS_XX3	PS_XX2	PS_XX1	PS_XX0
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 -- reserved, must be 0.  
 Bit6~Bit0 PS\_XX<6:0>: Input function allocation control bit (according to the actual port of the chip, see the multiplexing function allocation table);  
 0x00= allocated to port P00;  
 0x01= allocated to port P01;  
 .....  
 0x20 = Assigned to port P20;  
 0x21= assigned to port P21;  
 .....  
 0x50= assigned to port P50;  
 .....  
 0xFF= not assigned to port.

If P11 is configured as RXD1, the configuration is as follows:

```
PS_RXD1 =0x11; //Select P11 as the RXD1 pin
P11CFG=0x07; // P11 is multiplexed as UART1's RXD1
```

### 7.2.5 Port external interrupt control register.

When using external interrupts, the port needs to be configured as a GPIO function and the direction is set as an input port. Or the multiplexing function is an input port (such as RXD0, RXD1), and each port can be configured as a GPIO interrupt function.

PORTx external interrupt control register PxN<sub>EICFG</sub>

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PxN <sub>EICFG</sub>	--	--	--	--	--	--	Px1 <sub>EICFG</sub> 1	Px0 <sub>EICFG</sub> 0
R/W	--	--	--	--	--	--	R/W	R/W
Reset value	0	0	0	0	0	0	0	0

Bit7~Bit2 -- reserved, must be 0.  
 Bit1~Bit0 PxN<sub>EICFG</sub><1:0>: PxN external interrupt control bit;  
 00 = external interrupt disabled;  
 01 = rising edge trigger interrupt;  
 10 = falling edge trigger interrupt;  
 11 = rising edge or falling edge trigger interrupt.

There are 8 external interrupt control registers of Px, including Px0<sub>EICFG</sub>~Px7<sub>EICFG</sub>, which control the external interrupts of Px0~Px7 respectively. x=0,1,2,5.

If configure P00 to trigger the interrupt on the falling edge, the configuration is as follows:

```
P00CFG = 0x00; //Configure P00 as a GPIO function
P0TRIS = 0x00; //Configure P00 as an input
P00EICFG = 0x02; //Configure P00 as a falling edge trigger interrupt
EA = 1; //Global interrupt enable
P0EXTIE = 0x01; //Allow the P00 external interrupt function
```

## 7.2.6 Reuse function application note

- 1) The multiplex function configuration register is set by default to be analog function (0x01), to use digital function, the configuration register needs to be set to 0x00.
- 2) The input of the multiplexing function is relatively independent of the external interrupt (GPIO interrupt) of the port and the structure of the port input function.  
For example, configure the P20 port as RXD1, and configure the P20 GPIO interrupt trigger mode as rising edge trigger and interrupt enable. When the P20 input changes from low to high, it will trigger the P20 GPIO interrupt.
- 3) The input structure of the digital signal is not affected by the system configuration status.  
For example, if the P50 port is powered on and configured as an external reset port, the input module of this port will be opened. If P50 is configured as RXD1 in the program and the interrupt enable is turned on, the interrupt service routine will be executed before the reset signal sampling time is valid, and then the reset operation will be generated.
- 4) It should be noted that in the debug mode, if the multiplexing function is configured on the DSDA port, the input function is also valid. It is recommended that the related multiplexing function is not configured to the DSDA port in the debugging mode.
- 5) When the port is used as an analog function, when the function configuration register is set to 0x01, the hardware closes the digital circuit to reduce power consumption, and the GPIO function-related register setting is invalid.
- 6) The port input/communication input function has priority restrictions. If there are two or more ports configured with the same input function at the same time, configure the selection according to the priority order of P00, P01, ..., P54 from high to low.
- 7) The output function of the communication port has no priority limit. If there are multiple ports configured with the same output function, the function will be output at these ports at the same time.
- 8) RXD0/1 of UART0/1 needs to be selected by the port allocation register when it is used as an input function, and has nothing to do with the port allocation register when it is used as a synchronous output function. That is, when RXD1 is used as a synchronous output function, multiple pins can be selected as RXD0/1 output at the same time.
- 9) When the SCLK of SPI is used as the clock input of the slave, it needs to be selected by the port distribution register. When it is used as the clock output of the master, it has nothing to do with the port distribution register. It is recommended to configure the port distribution register when SCLK is used as an output or input.
- 10) When the SCL of IIC is used as the clock input of the slave, it needs to be selected by the port allocation register. When it is used as the clock output of the master, it is related to the port allocation register. It is recommended that the SCL be configured with the port allocation register whether it is used as the output or input.

## 8. Watch Dog Timer

### 8.1 Overview

Watch Dog Timer is an on-chip timer with optional overflow time and clock source provided by the system clock Fsys.

When the watchdog timer counts to the set overflow value, a watchdog overflow interrupt flag bit (WDTIF=1) is generated. If the global interrupt is enabled (EA=1) and the watchdog timer interrupt is enabled (EIE2[5]=1), the CPU will execute the interrupt service routine and clear the watchdog counter by writing the register WDCON[0]=1. After the watchdog counter is cleared, the counter starts counting from 0 again until the next timer overflows.

When the watchdog timer overflows, if the watchdog overflow reset is enabled (WDCON[1]=1) and the watchdog counter is not cleared, a watchdog overflow reset will be generated. The watchdog overflow reset is a protection setting of the system. When the system runs to an unknown state, the watchdog can be used to reset the system, so as to prevent the system from entering an infinite loop. The watchdog time-out reset is detailed in the reset chapter.

### 8.2 Related registers

#### 8.2.1 Watchdog control register WDCON

0x97	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
WDCON	SWRST	PORF	-	-	WDTIF	WDTRF	WDTRE	WDTCLR
R/W	R/W	R/W	R	R	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: execute system software reset (write 0 to clear after reset). 0: -
Bit6	PORF:	Power-on reset flag; 1: The system is power-on reset (write 0 to clear, no TA write sequence is required). 0: -
Bit5~Bit4	-	Reserved, all must be 0.
Bit3	WDTIF:	WDT overflow interrupt flag bit; 1 = WDT overflow (write 0 to clear); 0 = WDT does not overflow.
Bit2	WDTRF:	WDT reset flag; 1=The system is reset by WDT (write 0 to clear); 0=The system is not reset by WDT.
Bit1	WDTRE:	WDT reset enable bit; 1= Enable WDT to reset CPU; 0= Disable WDT to reset CPU.
Bit0	WDTCLR:	WDT counter clear bit; 1= clear WDT counter (hardware automatic clearing); 0= disable WDT counter (writing 0 is invalid).

Note:

1. If the WDT configuration in CONFIG is: ENABLE, the WDT is always enabled, regardless of the state of the WDTRE control bit. And the overflow reset function of WDT is forcibly turned on.
2. If the WDT configuration in CONFIG is: SOFTWARE CONTROL, you can use the WDTRE control bit to enable or disable WDT.

Modify the instruction sequence required by WDCON (no other instructions can be inserted in the middle):

MOV	TA,#0AAH
MOV	TA,#055H
ORL	WDCON,#01H

### 8.2.2 Watchdog overflow control register CKCON

0x8E	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
CKCON	WTS2	WTS1	WTS0	T1M	T0M	-	-	-
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	1	1

- Bit7~Bit5      WTS< 2:0>:    WDT overflow time selection bit;
- 000=  $2^{17} \cdot T_{sys}$ ;
  - 001=  $2^{18} \cdot T_{sys}$ ;
  - 010=  $2^{19} \cdot T_{sys}$ ;
  - 011=  $2^{20} \cdot T_{sys}$ ;
  - 100=  $2^{21} \cdot T_{sys}$ ;
  - 101=  $2^{22} \cdot T_{sys}$ ;
  - 110=  $2^{24} \cdot T_{sys}$ ;
  - 111=  $2^{26} \cdot T_{sys}$ .
- Bit4            T1M:    Timer1 clock source selection bit;
- 0=  $F_{sys}/12$ ;
  - 1=  $F_{sys}/4$ .
- Bit3            T0M:    Timer0 clock source selection bit;
- 0=  $F_{sys}/12$ ;
  - 1=  $F_{sys}/4$ .
- Bit2 ~ Bit0      -    reservations are a must.

## 8.3 The WDT interrupt

Watchdog timer can be enabled or disabled through the EIE2 register, and the high/low priority is set through the EIP2 register. The interrupt related bits are as follows.

### 8.3.1 Interrupt mask register EIE2

0xAA	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIE2	SPIIE	I2CIE	WDTIE	ADCIE	PWMIE	-	ET4	ET3
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<sup>2</sup>C interrupt enable bit;  
             1= Enable I<sup>2</sup>C interrupt;  
             0= Disable I<sup>2</sup>C interrupt.
- Bit5      WDTIE: WDT interrupt enable bit;  
             1= Enable WDT overflow interrupt;  
             0= Disable WDT overflow interrupt.
- Bit4      ADCIE: ADC interrupt enable bit;  
             1= enable ADC interrupt;  
             0= disable ADC interrupt.
- Bit3      PWMIE: PWM total interrupt enable bit;  
             1= allow all PWM interrupts;  
             0= disable all PWM interrupts.
- Bit2      - Reserved, must be zero.
- Bit1      ET4: Timer4 interrupt enable bit;  
             1= Enable Timer4 interrupt;  
             0= Disable Timer4 interrupt.
- Bit0      ET3: Timer3 interrupt enable bit;  
             1= Enable Timer3 interrupt;  
             0= Disable Timer3 interrupt.

### 8.3.2 Interrupt priority control register EIP2

0xBA	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIP2	PSPi	PI2C	PWDT	PADC	PPWM	-	PT4	PT3
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      PSPi: SPI interrupt priority control bit;  
           1= set as high-level interrupt;  
           0= set as low-level interrupt.
- Bit6      PI2C: I<sup>2</sup>C interrupt priority control bit;  
           1= set as high-level interrupt;  
           0= set as low-level interrupt.
- Bit5      PWDT: WDT interrupt priority control bit;  
           1= set as high-level interrupt;  
           0= set as low-level interrupt.
- Bit4      PADC: ADC interrupt priority control bit;  
           1= set as high-level interrupt;  
           0= set as low-level interrupt.
- Bit3      PPWM: PWM interrupt priority control bit  
           1= set as high-level interrupt;  
           0= set as low-level interrupt.
- Bit2      - Reserved, must be zero.
- Bit1      PT4: TIMER4 interrupt priority control bit;  
           1= set as high-level interrupt;  
           0= set as low-level interrupt.
- Bit0      PT3: TIMER3 interrupt priority control bit;  
           1= set as high-level interrupt;  
           0= set as low-level interrupt.

## 9. Timer 0/1

Timer 0 is similar in type and structure to Timer 1, and is two 16-bit timers. Timer 1 has three working modes, and Timer 0 has four working modes. They provide basic timing and event counting operations.

In the "timer mode", the timer register is incremented every 12 or 4 system cycles when the timer clock is enabled.

In the "counter mode", the timing register will increase whenever it detects a falling edge on the corresponding input pin (T0 or T1).

### 9.1 Overview

Timer 0 and Timer 1 are fully compatible with standard 8051 timers.

Each timer consists of two 8-bit registers: {TH0 (0x8C): TL0 (0x8A)} and {TH1 (0x8D): TL1 (0x8B)}. Timers 0 and 1 work in four identical modes. Timer0 and Timer1 modes are described below.

Mode	M1	M0	Function description
0	0	0	THx[7:0], TLx[4:0] form a 13-bit timer/counter
1	0	1	THx[7:0], TLx[7:0] form a 16-bit timer/counter
2	1	0	TLx[7:0] form 8-bit automatic reload timer/counter, reload from THx
3	1	1	TL0, TH0 are two 8-bit timer/counters, Timer1 stop counting

Register THx and TLx are special function registers, it has the function of storing the actual timer value. THx and TLx can be cascaded into 13-bit or 16-bit registers through mode options. Each time an internal clock pulse is received or a state transition occurs on the external timer pin, the value of the register is increased by one. The timer will start counting from the value loaded in the preset register until the timer overflows, at which time an internal interrupt signal will be generated. If the automatic reload mode of the timer is selected, the timer value will be reset to the initial value of the preload register and continue counting, otherwise the timer value will be reset to zero. Note that in order to get the maximum calculation range of the timer/counter, the preset register must be cleared first.



## 9.2 Related register

### 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 timing/counting selection bit; 1= counting; 0= timing.
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 automatic reload timing /Counter; 11= Mode 3, stop counting.
Bit3	GATE0: Timer 0 gate control bit; 1= enable; 0= disable.
Bit2	CT0: Timer 0 timing/counting selection bit; 1= counting; 0= timing.
Bit1~ Bit0	T0M<1:0>: Timer 0 mode selection bits; 00= Mode 0, 13-bit timer/counter; 01= Mode 1, 16-bit timer/counter; 10= Mode 2, 8-bit automatic reload timing /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 overflows and enters the interrupt service routine. The hardware is automatically cleared;  
           0 = Timer1 counter does not overflow.
- Bit6      TR1: Timer1 running control bit;  
           1 = Timer1 is started;  
           0 = Timer1 is closed.
- Bit5      TF0: Timer0 counter overflow interrupt flag bit;  
           1 = Timer0 counter overflows, and the hardware is automatically cleared when entering the interrupt service routine;  
           0 = Timer0 counter does not overflow.
- Bit4      TR0: Timer0 running control bit;  
           1 = Timer0 is started;  
           0 = Timer0 is closed.
- Bit3      IE1: External interrupt 1 flag bit;  
           1 = External interrupt 1 generates an interrupt, and the hardware is automatically cleared when entering the interrupt service routine;  
           0 = External interrupt 1 does not generate an interrupt.
- Bit2      IT1: External interrupt 1 trigger mode control bit;  
           1 = falling edge trigger;  
           0 = low level trigger.
- Bit1      IE0: External interrupt 0 flag bit;  
           1 = External interrupt 0 generates an interrupt, and the hardware is automatically cleared when entering the interrupt service routine;  
           0 = External interrupt 0 does not generate an interrupt.
- Bit0      IT0: External interrupt 0 trigger mode control bit;  
           1 = falling edge trigger;  
           0 = low level trigger.

### 9.2.3 Timer0 data register low 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 data register (also used as counter high).

### 9.2.5 Timer1 data register low 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 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 data register (also used as counter high bit).

### 9.2.7 Function clock control register CKCON

0x8E	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
CKCON	WTS2	WTS1	WTS0	T1M	T0M	-	-	-
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	1	1

Bit7~Bit5      WTS<2:0>: WDT overflow time selection bit;  
                   000=  $2^{17} \cdot T_{sys}$ ;  
                   001=  $2^{18} \cdot T_{sys}$ ;  
                   010=  $2^{19} \cdot T_{sys}$ ;  
                   011=  $2^{20} \cdot T_{sys}$ ;  
                   100=  $2^{21} \cdot T_{sys}$ ;  
                   101=  $2^{22} \cdot T_{sys}$ ;  
                   110=  $2^{24} \cdot T_{sys}$ ;  
                   111=  $2^{26} \cdot T_{sys}$ .  
  
 Bit4            T1M: Timer1 clock source selection bit;  
                   0=  $F_{sys}/12$ ;  
                   1=  $F_{sys}/4$ .  
  
 Bit3            T0M: Timer0 clock source selection bit;  
                   0=  $F_{sys}/12$ ;  
                   1=  $F_{sys}/4$ .  
  
 Bit2 ~ Bit0      - Reserved, must be 1.

## 9.3 Timer0/1 interrupt

Timer0/1 can be enabled or disabled through the IE register, and the high/low priority can also be set through the IP 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= allow all interrupts that are not masked; 0= disable all interrupts.
Bit6	ES1: UART1 interrupt enable bit; 1= enable UART1 interrupt; 0= disable UART1 interrupt.
Bit5	ET2: TIMER2 total interrupt enable bit; 1= Enable all TIMER2 interrupts; 0= Disable all TIMER2 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.

### 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/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, must be zero.
Bit6	PS1:	UART1 interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit5	PT2:	TIMER2 interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit4	PS0:	UART0 interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit3	PT1:	TIMER1 interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit2	PX1:	External interrupt 1 interrupt priority control bit; 1= Set as high-level interrupt; 0= Set as low-level interrupt.
Bit1	PT0:	TIMER0 interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit0	PX0:	External interrupt 0 interrupt priority control bit; 1= set as high-level interrupt; 0= set as 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 = Timer1 counter overflows, automatically cleared by hardware when entering the interrupt service routine, or cleared by software; 0 = Timer1 counter does not overflow.
Bit6	TR1:	Timer1 running control bit; 1 = Timer1 is started; 0 = Timer1 is closed.
Bit5	TF0:	Timer0 counter overflow interrupt flag bit; 1 = Timer0 counter overflows, it is automatically cleared by hardware when entering the interrupt service routine, or can be cleared by software; 0 = Timer0 counter has no overflow.
Bit4	TR0:	Timer0 running control bit; 1 = Timer0 is started; 0 = Timer0 is closed.
Bit3	IE1:	External Interrupt 1 flag; 1= External Interrupt 1 generates an interrupt, which is automatically cleared by hardware when entering the interrupt service routine, or cleared by software;

	0=	External Interrupt 1 does not generate an interrupt.
Bit2	IT1:	External interrupt 1 trigger mode control bit;
	1=	falling edge trigger;
	0=	low level trigger.
Bit1	IE0:	External interrupt 0 flag bit;
	1=	External interrupt 0 generates an interrupt, which is automatically cleared by hardware when entering the interrupt service routine, or it can be cleared by software;
	0=	External interrupt 0 does not generate an interrupt.
Bit0	IT0:	External interrupt 0 trigger mode control bit;
	1=	falling edge trigger;
	0=	low level trigger.

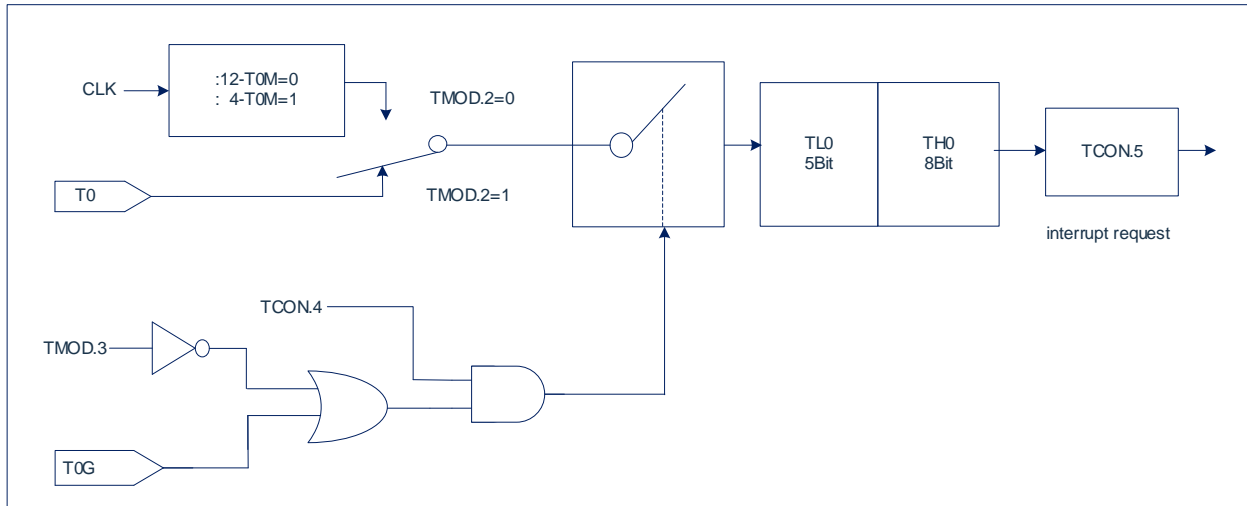
The flag bit that generates an interrupt can be cleared by software, and the result is the same as cleared by hardware. In other words, you can generate interrupts by software (it is not recommended to generate interrupts by writing flag bits) or cancel pending interrupts.

The TF0 and TF1 flags can be cleared by writing 0 when the interrupt is not enabled.

## 9.4 Timer0 working mode

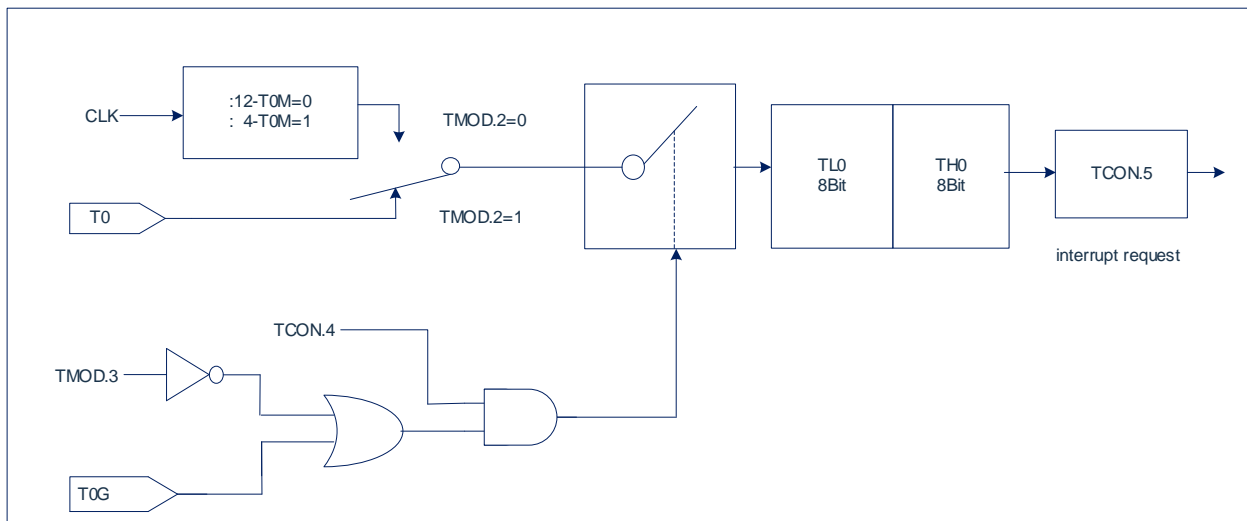
### 9.4.1 T0-Mode 0 (13-bit timer/counting mode)

In this mode, Timer0 is a 13-bit register. When all the bits of the counter are turned from 1 to 0, the timer 0 interrupt flag TF0 is set to 1. When TCON.4=1 and TMOD.3=0 or TCON.4=1, TMOD.3=1, T0G=1, the counting input is enabled to timer 0. (Set TMOD.3=1 to allow timer 0 to be controlled by external pin T0G for pulse width measurement). The 13-bit register consists of the lower 5 bits of TH0 and TL0. The upper 3 bits of TL0 should be ignored. The structure diagram of Timer0 mode 0 is shown in the following figure:



### 9.4.2 T0-Mode 1 (16-bit timer/counting mode)

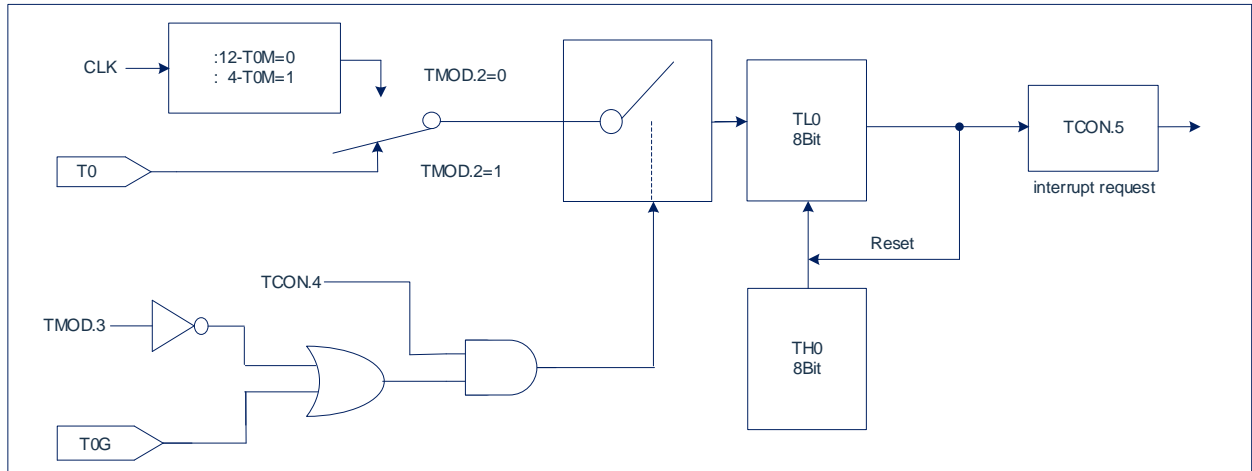
Mode 1 is the same as mode 0, except that all 16 bits of the timer 0 data register run in mode 1. Timer0 mode 1 block diagram as shown below:





### 9.4.3 T0 - Mode 2 (8-bit auto reload timer / counter mode)

The timer register in mode 2 is an 8-bit counter (TL0) with auto-reload mode, as shown in the figure below. The overflow from TL0 not only sets TF0 to 1, but also reloads the contents of TH0 to TL0 by software. The value of TH0 remains unchanged during the reinstallation process. The structure diagram of Timer0 Mode 2 is shown in the figure below:



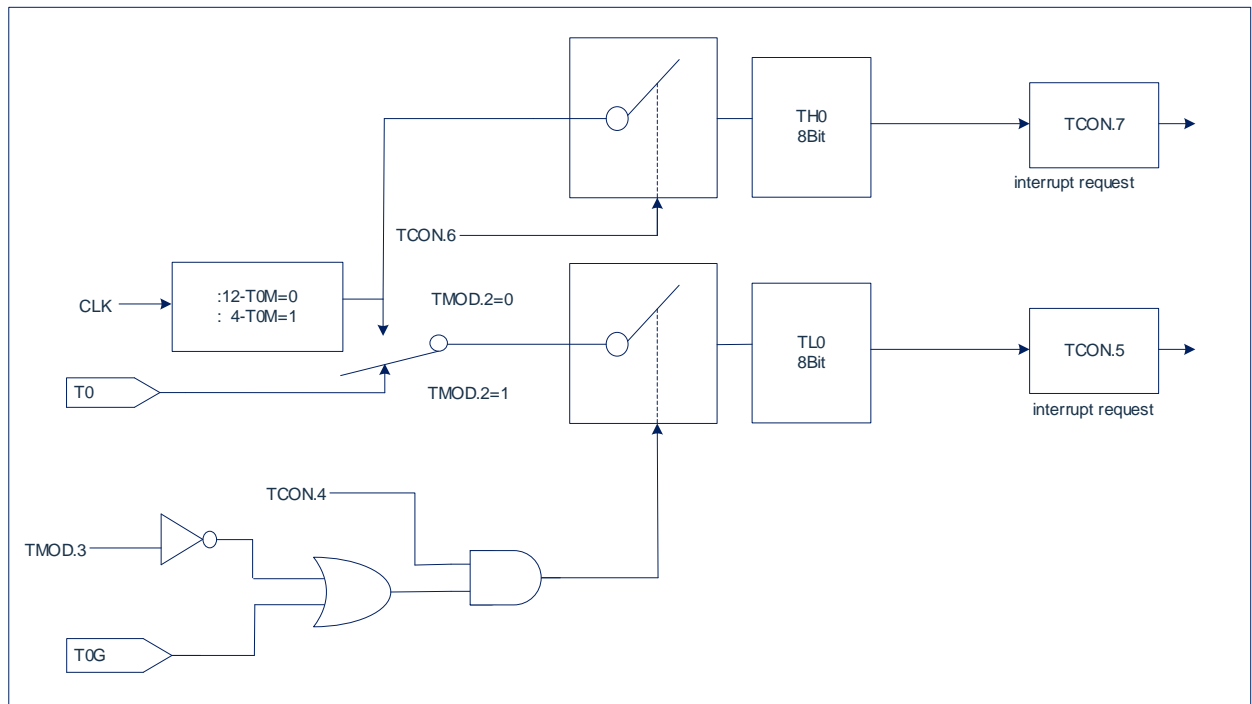
### 9.4.4 T0-Mode 3 (two separate 8-bit timers/counters)

Timer 0 in Mode 3 sets TL0 and TH0 as two independent counters. The logic of Timer 0 Mode 3 is shown in the figure below.

TL0 can work as a timer or counter, and use timer 0 control bits: such as CT0, TR0, GATE0, and TF0.

TH0 can only work as a timer, and uses the TR1 and TF1 flags of timer 1 and controls the interrupt of timer 1.

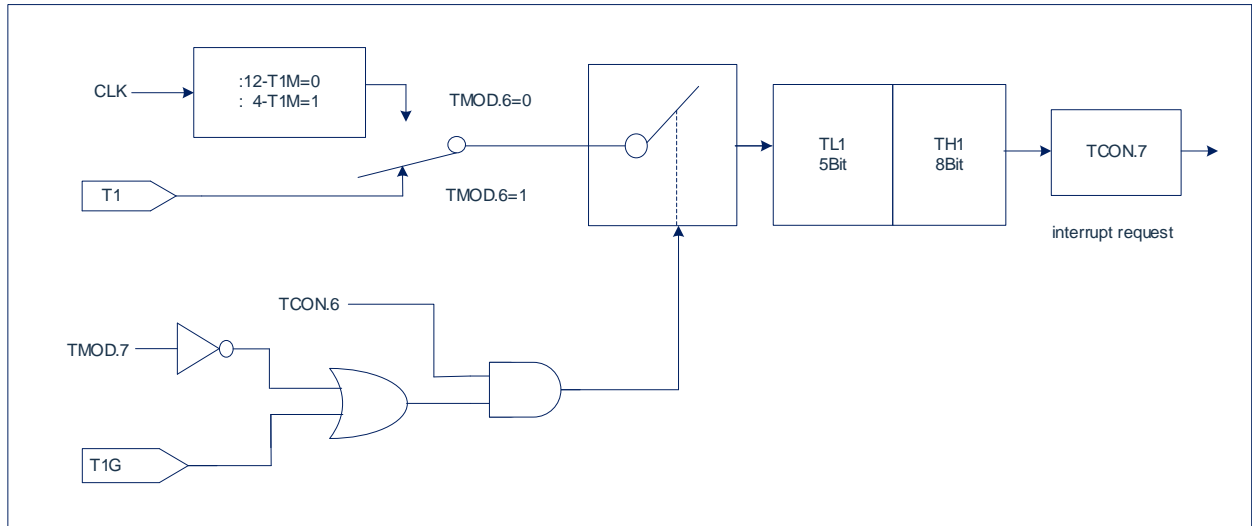
Mode 3 can be used when two 8-bit timers/counters are required. When timer 0 is in mode 3, timer 1 can be turned off by switching to its own mode 3, or it can still be used as a baud rate generator by the serial channel, or in any case that does not require timer 1 interrupts. In application. The structure diagram of Timer0 Mode 3 is shown in the figure below:



## 9.5 Timer1 working mode

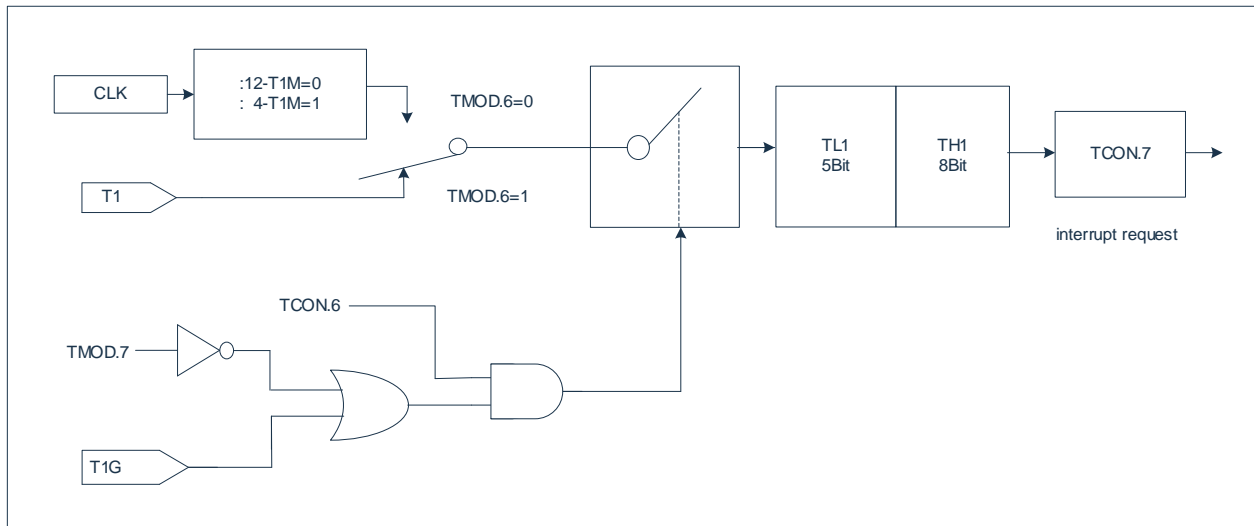
### 9.5.1 T1-Mode 0 (13-bit timer/counting mode)

In this mode, Timer 1 is a 13-bit register. When all the bits of the counter are turned from 1 to 0, the timer 1 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 timer 1. (Setting TMOD.7=1 allows Timer 1 to be controlled by the external pin T1G for pulse width measurement). The 13-bit register consists of 8 bits of TH1 and the lower 5 bits of TL1. The upper three bits of TL1 should be ignored. The structure diagram of Timer1 mode 0 is shown in the figure below:



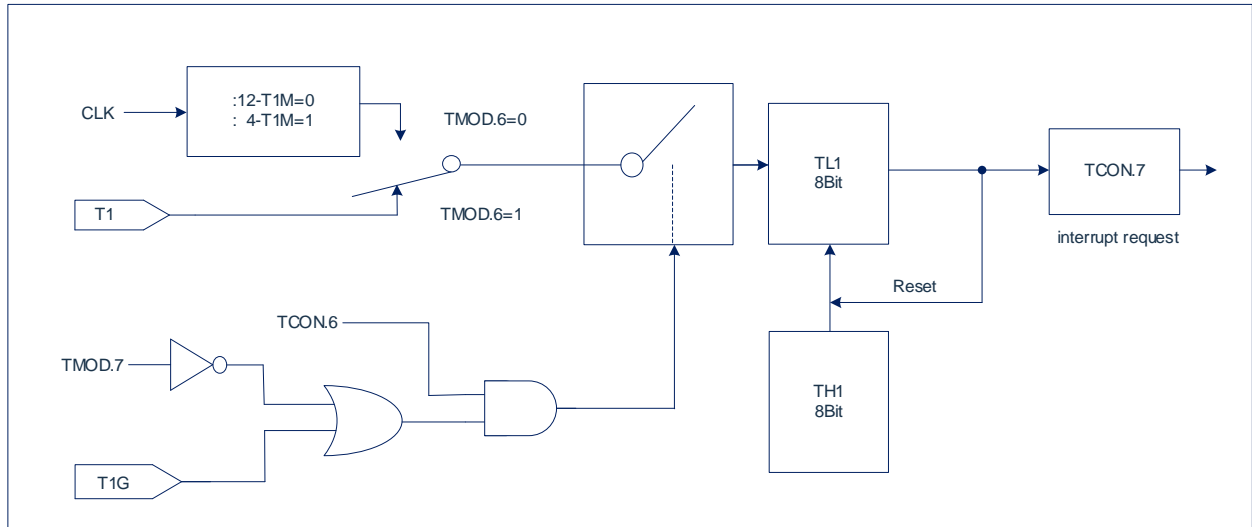
### 9.5.2 T1-Mode 1 (16-bit timer/counting mode)

Mode 1 is the same as mode 0, except that all 16 bits of the timer 1 register are running in mode 1. The block diagram of Timer1 mode 1 is shown in the figure below:



### 9.5.3 T1-Mode 2 (8-bit auto-reload timer/counting mode)

The timer 1 register in mode 2 is an 8-bit counter (TL1) with auto-reload mode, as shown in the figure below. The overflow from TL1 not only sets TF1 to 1, but also reloads the contents of TH1 to TL1 by software. The value of TH1 remains unchanged during the reinstallation process. The structure diagram of Timer1 Mode 2 is shown in the figure below:



### 9.5.4 T1-Mode 3 (stop counting)

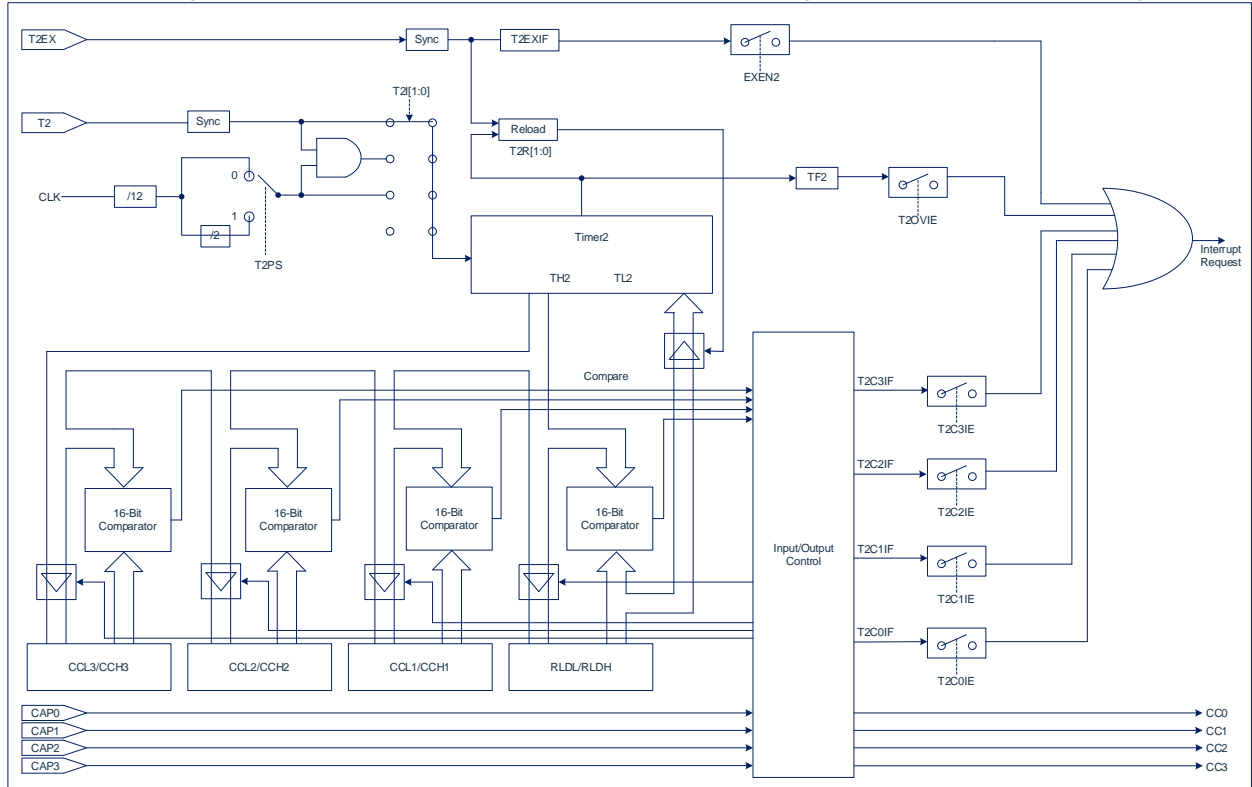
Timer 1 in mode 3 stops counting, and its effect is the same as setting TR1=0.

## 10. Timer 2

Timer 2 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, such as pulse generation, pulse width modulation, pulse width measurement, etc..

### 10.1 SUMMARY

The structure diagram of Timer 2 with additional compare/capture/reload register function is shown in the figure below



## 10.2 Associated register

### 10.2.1 control register of Timer2

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; 1= $F_{sys}/24$ ; 0= $F_{sys}/12$ .
Bit6	I3FR: Capture channel 0 input single edge selection and compare interrupt time selection bit; capture channel 0 mode: 1= capture rising edge to RLDL/RLDH register; 0= capture falling edge to RLDL/RLDH register. Compare channel 0 mode: 1 = interrupt generated when TL2/TH2 and RLDL/RLDH are not equal to equal to each other; interrupt generated when 0 = TL2/TH2 and RLDL/RLDH are equal to not equal to each other;
Bit5	CAPES: capture channel 1-3 input Single edge selection (valid for capture channels 1-3 together). 0=The rising edge is captured to the CCL1/CCH1-CCL3/CCH3 register; 1=The falling edge is captured to the CCL1/CCH1-CCL3/CCH3 register.
Bit4~Bit3	T2R<1:0>: Timer2 loading mode selection bit; 0x= reloading prohibited; 10= loading mode 1: automatic reloading when Timer2 overflows; 11= loading mode 2: reloading on the falling edge of T2EX pin.
Bit2	T2CM: Comparison mode selection; 1= comparison mode 1; 0= comparison mode 0.
Bit1~Bit0	T2I<1:0>: Timer2 clock input selection bit; 00= Timer2 stop; 01= System clock frequency division (selected by T2PS control frequency division); 10= External pin T2 as event input (event counting mode ); 11= External pin T2 is used as gate control input (Gated timing mode).

### 10.2.2 Timer2 data register low 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 TH2

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 data register (also used as counter low).

### 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 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 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 channel 2 register lower

0xC4	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
CCL2	CCL27	CCL26	CCL25	CCL24	CCL23	CCL22	CCL21	CCL20
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      CCL2<7:0>: Timer 2 compare/capture channel 2 register low bit.

### 10.2.9 Timer2 compare/capture channel 2 register high CCH2

0xC5	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
CCH2	CCH27	CCH26	CCH25	CCH24	CCH23	CCH22	CCH21	CCH20
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      CCH2<7:0>: Timer 2 compare/capture channel 2 register high bit.

### 10.2.10 Timer2 Compare / Capture channel 3 register lower

0xC6	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
CCL3	CCL37	CCL36	CCL35	CCL34	CCL33	CCL32	CCL31	CCL30
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      CCL3<7:0>: Timer 2 compare/capture channel 3 register low bit.

### 10.2.11 Timer2 Compare / Capture channel 3 registers high

0xC7	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
CCH3	CCH37	CCH36	CCH35	CCH34	CCH33	CCH32	CCH31	CCH30
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      CCH3<7:0>: Timer 2 compare/capture channel 3 register high bit.

### 10.2.12 Timer2 compare capture control register CCEN

0xCE	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
CCEN	CMH3	CML3	CMH2	CML2	CMH1	CML1	CMH0	CML0
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	CMH3-CML3:	capture/compare mode control bit; 00= capture/compare prohibited; 01= capture operation is triggered on the rising or falling edge of channel 3 (CAPES selection) ; 10= Compare mode enable; 11=The capture operation is triggered when writing CCL3 or the double-edge trigger of channel 3.
Bit5~Bit4	CMH2-CML2:	capture/compare mode control bit; 00= capture/comparison disabled; 01= capture operation is triggered on the rising or falling edge of channel 2 (CAPES selection); 10= comparison mode enable; 11= capture The operation is triggered when writing CCL2 or the double-edge trigger of channel 2.
Bit3~Bit2	CMH1-CML1:	capture/compare mode control bit; 00= capture/compare disabled; 01= capture operation is triggered on the rising or falling edge of channel 1 (CAPES selection); 10= comparison mode enable; 11= capture The operation is triggered when writing CCL1 or the double-edge trigger of channel 1.
Bit1~Bit0	CMH0-CML0:	capture/compare mode control bit; 00= capture/comparison disabled; 01= capture operation is triggered on the rising or falling edge of channel 0 (I3FR selection); 10= comparison mode enable; 11= capture The operation is triggered when RLDL is written or the double edge of channel 0 is triggered.



## 10.3 Timer2 interrupt

Timer 2 can be enabled or disabled through the register IE, the total interrupt can also be set high/low priority through the IP register. Timer2 has 4 types of interrupts:

- ◆ timer overflow interrupt.
- ◆ Interrupt on the falling edge of external pin T2EX.
- ◆ Comparison is interrupted.
- ◆ Capture interrupt.

To set the Timer2 interrupt, you need to configure the global interrupt enable bit (EA=1), the Timer2 total interrupt enable bit (ET2=1), and the corresponding interrupt type enable bit (T2IE) of Timer2. The 4 types of interrupts of Timer2 share an interrupt vector. After entering the interrupt service routine, it is necessary to determine the relevant flag bit to determine which type has generated the interrupt.

### 10.3.1 Interrupt related 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= allow all unmasked interrupts;  
0= disable all interrupts.
- Bit6 ES1: UART1 interrupt enable bit;  
1= enable UART1 interrupt;  
0= disable UART1 interrupt.
- Bit5 ET2: TIMER2 total interrupt enable bit;  
1= Enable all TIMER2 interrupts;  
0= Disable all TIMER2 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

0xCF	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
T2IE	T2OVIE	T2EXIE	--	--	T2C3IE	T2C2IE	T2C1IE	T2C0IE
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	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 ~ Bit4	- Reserved, must be 0's.
Bit3	T2C3IE: Timer2 compare channel 3 interrupt enable bit; 1= Enable interrupt; 0= Disable interrupt.
Bit2	T2C2IE: Timer2 compare channel 2 interrupt enable bit; 1= Enable interrupt; 0= Disable interrupt.
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 turn on the interrupt of Timer2, you also need to turn on the total interrupt enable bit ET2=1 (IE.5=1) of Timer2.

### 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/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, must be zero.
Bit6	PS1: UART1 interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit5	PT2: TIMER2 interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit4	PS0: UART0 interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit3	PT1: TIMER1 interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit2	PX1: External interrupt 1 interrupt priority control bit; 1= Set as high-level interrupt; 0= Set as low-level interrupt.
Bit1	PT0: TIMER0 interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit0	PX0: External interrupt 0 interrupt priority control bit; 1= Set as high-level interrupt; 0= Set as low-level interrupt.

**10.3.1.4 Timer2 interrupt flag register T2IF**

0xC9	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
T2IF	TF2	T2EXIF	-	-	T2C3IF	T2C2IF	T2C1IF	T2C0IF
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 TF2: Timer2 counter overflow interrupt flag bit;  
 1 = Timer2 counter overflows and needs to be cleared by software;  
 0 = Timer2 counter does not overflow.
- Bit6 T2EXIF: Timer2 external load flag bit;  
 1= Timer2 T2EX port generates a falling edge, which needs to be cleared by software;  
 0= -
- Bit5~Bit4 - Reserved, all must be 0.
- Bit3 T2C3IF: Timer2 compare/capture channel 3 flag bit;  
 1= Timer2 compare channel 3 {CCH3:CCL3}={TH2:TL2} or capture channel 3 has a capture operation, which needs to be cleared by software.  
 0= -
- Bit2 T2C2IF: Timer2 compare/capture channel 2 flag bit;  
 1= Timer2 compare channel 2 {CCH2:CCL2}={TH2:TL2} or capture channel 2 has a capture operation, which needs to be cleared by software.  
 0= -
- Bit1 T2C1IF: Timer2 compare/capture channel 1 flag bit;  
 1= Timer2 compare channel 1 {CCH1:CCL1}={TH2:TL2} or capture channel 1 has a capture operation, which 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 a capture operation, which needs to be cleared by software.  
 0 = -

### 10.3.2 Timer interruption

Timing is set by the interrupt enable register bit T2IE [7], the interrupt flag register review via T2IF [7]. When the Timer2 timer overflows, the timer overflow interrupt flag bit TF2 will be set.

### 10.3.3 External trigger interrupt

External pin T2EX falling edge trigger interrupt enable bit is set by register T2IE[6], and the interrupt flag bit is checked by register T2IF[6]. When the T2EX pin falls along, the external load interrupt flag bit T2EXIF will be set.

### 10.3.4 Compare interrupt

All 4 compare channels support compare interrupts. The compare interrupt enable bit is set by the register T2IE[3:0], and the interrupt flag bit is checked by the register T2IF[3:0].

The comparison channel 0 can select the time when the comparison interrupt is generated, and when an interrupt is generated, the interrupt flag T2C0IF of the comparison channel 0 will be set to 1.

When I3FR = 0, TL2/TH2 and RLDL/RLDH will generate an interrupt from unequal to equal time;

When I3FR = 1, TL2/TH2 and RLDL/RLDH will be interrupted when they are equal to unequal;

The comparison channel 1~3 cannot select the interrupt generation time, and it is fixed to generate interrupts at the time when TL2/TH2 and CCxL/CCxH are not equal to equal. If an interrupt is generated, the corresponding compare channel interrupt flag T2CxIF will be set to 1..

### 10.3.5 Capture interrupt

4 capture channels all support external capture interrupt. The capture interrupt enable bit is set by the register T2IE[3:0], and the interrupt flag bit is checked by the register T2IF[3:0]. When a capture operation occurs, the interrupt flag T2CxIF of the corresponding capture channel is set to 1.

Note that the write capture mode will not generate an interrupt.

## 10.4 Timer2 function description

Timer 2 is a 16-bit up-counting timer with clock source from the system clock. Timer2 can be configured with the following functional modes:

- ◆ Timing mode.
- ◆ Reinstall mode.
- ◆ Gated timing mode.
- ◆ Event counting mode.
- ◆ Comparison mode.
- ◆ Capture mode.

Set different modes of Timer 2, which can be used for the generation of various digital signals and event capture, such as pulse generation, pulse width modulation, pulse width measurement, etc.

### 10.4.1 Timing mode

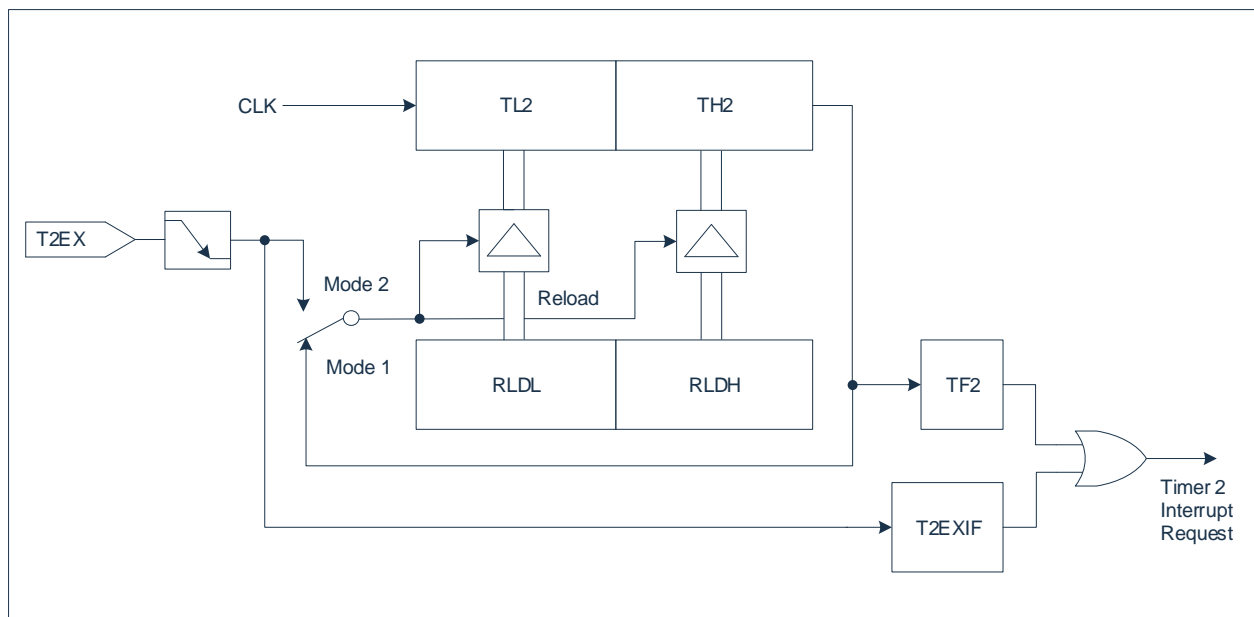
When the timing mode is used as a timer function, the clock source comes from the system clock. The prescaler provides 1/12 or 1/24 system frequency selection. The value of the prescaler is selected by the T2PS bit of the register T2CON. Therefore, the 16-bit timer register (consisting of TH2 and TL2) increments every 12 clock cycles or every 24 clock cycles.

### 10.4.2 Reload Mode

The reload mode of Timer 2 is selected by the T2R0 and T2R1 bits of the register T2CON. The block diagram of the reload structure is shown in the figure below.

In load mode 1: When the Timer2 counter rolls over from all 1s to 0 (counter overflows), not only the overflow interrupt flag bit TF2 is set to 1, but the Timer2 register automatically loads the 16-bit value from the RLDL/RLDH register, thereby overwriting the count value 0x0000, the required RLDL/RLDH value can be preset by software.

In load mode 2: The 16-bit reload operation from the RLDL/RLDH register 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 bit T2EXIF is set to 1, and Timer2 automatically loads the 16-bit value of the RLDL/RLDH register as the initial value of the count.



### 10.4.3 Gated timing mode

Timer2 is used as a gated timer function, the external input pin T2 is used as the gated input of timer 2. If the T2 pin is high, the internal clock input is gated to the timer. T2 pin is low to stop counting. This function is often used to measure pulse width.

### 10.4.4 Event counting mode

Timer2 is used as the event counting function, the timer counter adds 1 to the falling edge of the external input pin T2. The external input signal is sampled in each system clock cycle. When the sampled input shows a high level in one cycle and a low level in the next cycle, the count is increased. In the next cycle, when the T2 pin changes from high to low, the new count value is updated to the timer data register.

### 10.4.5 Comparison mode

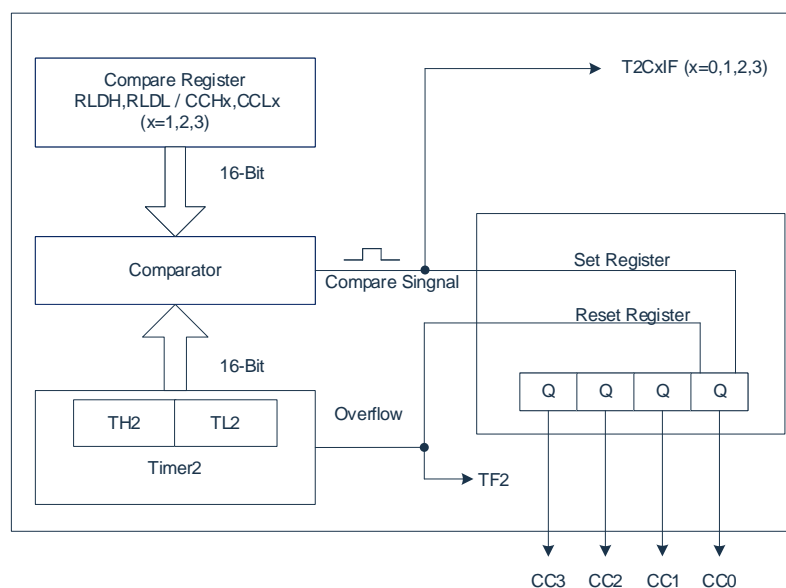
The comparison function includes two modes: comparison mode 0 and comparison mode 1, which are selected by the T2CM bit in the special function register T2CON. These two comparison modes can generate periodic signals and can change the duty cycle control mode. They are often used in pulse width modulation (PWM) and control situations that require continuous square wave generation, covering a wider range of applications.

The output channels of the comparison function are CC0, CC1, CC2, and CC3. Corresponding to the 16-bit comparison register {RLDH, RLDL}, {CCH1, CCL1}, {CCH2, CCL2}, {CCH3, CCL3} and the data register {TH2, TL2} the comparison result output signal.

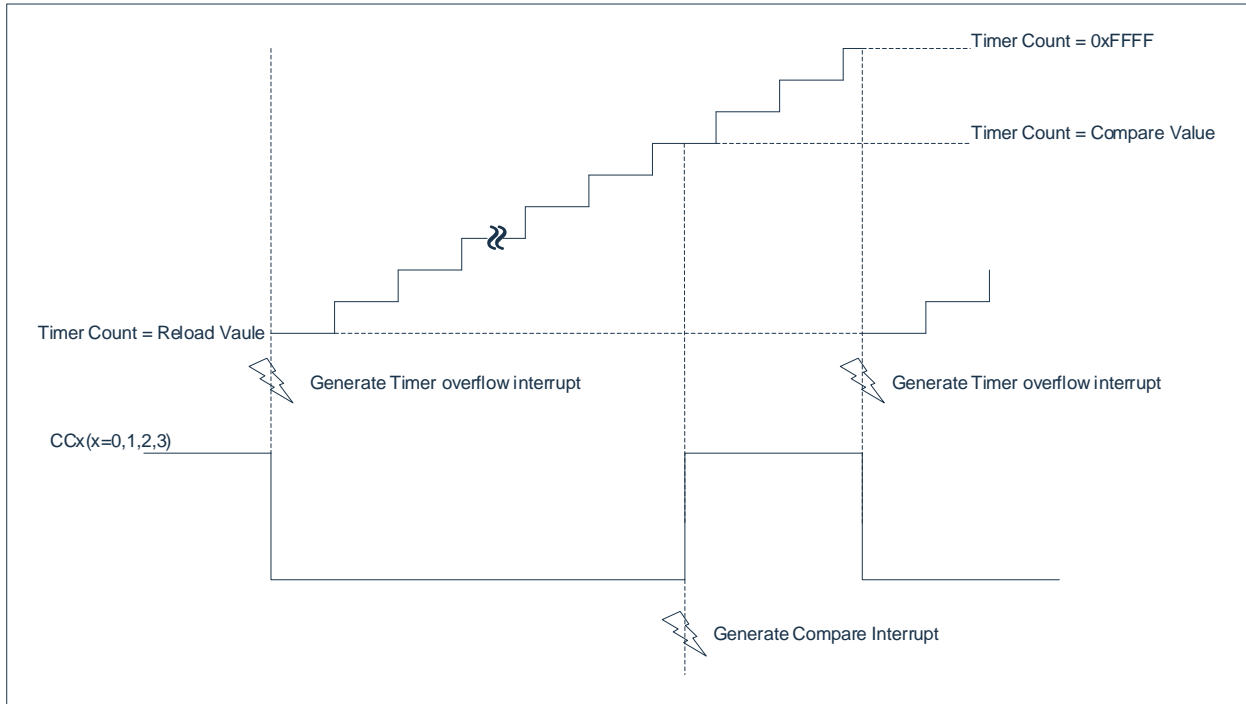
The 16-bit stored value stored in the compare register is compared with the count value of the timer. If the count value in the data register matches the stored value, a transition of the output signal will be generated on the corresponding port pin. Interrupt flag bit.

#### 10.4.5.1 Comparison mode 0

In mode 0, when the count value of the timer is equal to the comparison register, the comparison output signal changes from low level to high level. When the timer count value overflows, the comparison output signal becomes low level. The comparison output channel is directly controlled by two events: timer overflow and comparison operation. The structure block diagram of comparison mode 0 is shown in the following figure:



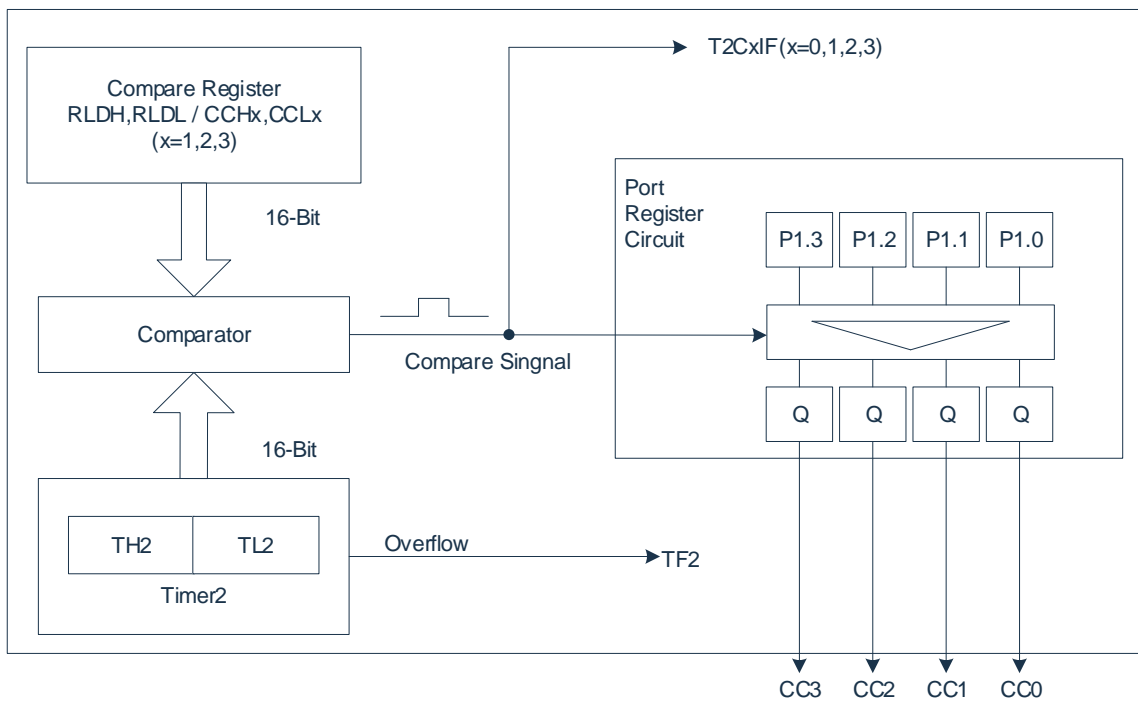
the output structure block diagram of comparison mode 0 is shown in the following figure:



#### 10.4.5.2 Comparison mode 1

In comparison mode 1, usually when the output signal has nothing to do with the constant signal period, and the software determines the output signal adaptively Transition occasions.

If mode 1 is enabled, the software will write to the corresponding output register of the CCx port, and the new value will not appear on the output pin until the next comparison match occurs. When the timer 2 counter matches the stored comparison value, the user can choose one of two ways to change the output signal or keep its old value. The block diagram of comparison mode 1 is shown in the figure below:



## 10.4.6 Capture mode

4 16-bit registers {RLDH, RLDL}, {CCH1, CCL1}, {CCH2, CCL2}, {CCH3, CCL3} as capture functions, each group of registers can be used to latch {TH2, TL2} The current 16-bit value. 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 writing the low byte (RLDL/CCL1/CCL2/CCL3) of the 16-bit capture register. This mode allows the software to read the content of {TH2, TL2} at runtime.

Capture channels 0~3 respectively select capture input pins CAP0~CAP3 as input source signals.

### 10.4.6.1 Capture mode 0

In capture mode 0, the positive transition, negative transition or positive and negative transition on the capture channel 0~3 (CAP0~CAP3) will generate a capture event. When a capture event occurs, the count value of the timer is latched in the corresponding capture register.

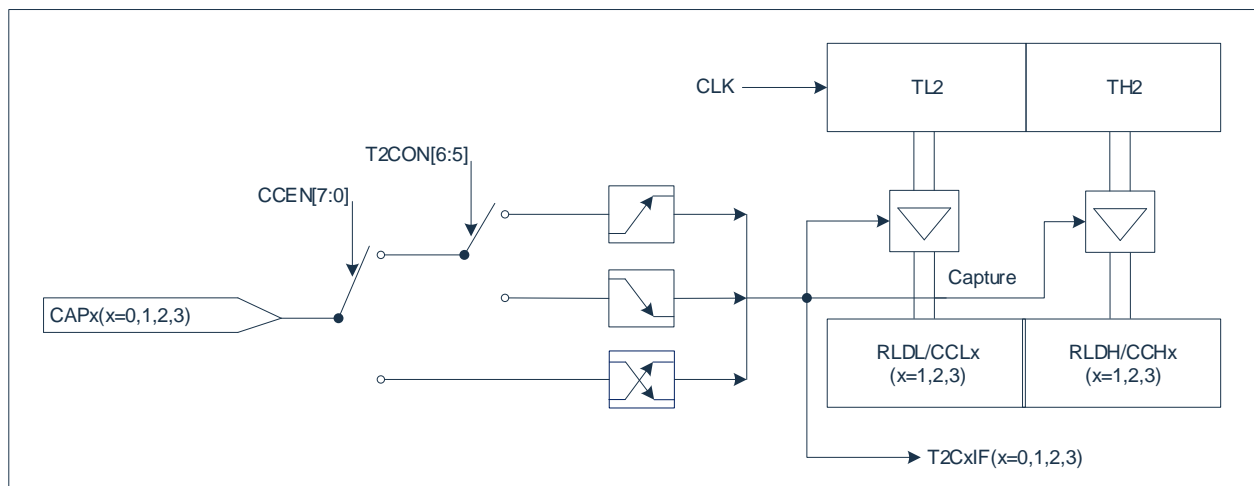
Whether the positive transition triggers the capture operation or the negative transition triggers the capture operation on the capture channel 0 depends on the I3FR bit of T2CON. I3FR=0, negative transition triggers capture; I3FR=1, positive transition triggers capture.

Whether the positive transition triggers the capture operation or the negative transition triggers the capture operation on the capture channels 1~3 depends on the CAPES bit of T2CON. CAPES=0, positive transition triggers capture; CAPES=1, negative transition triggers capture. The selected jump mode of capture channel 1~3 is that the same

capture channel 0~3 supports double jump capture operation at the same time. If the corresponding work mode control bit of the CCEN register is selected as 11, the channel supports the capture operation of double jump. It should be noted that this working mode also supports capture mode 1, that is, write operations can generate capture actions.

In capture mode 0, the external capture events of capture channels 0~3 can generate interrupts.

The structure diagram of capture mode 0 is shown in the following figure:



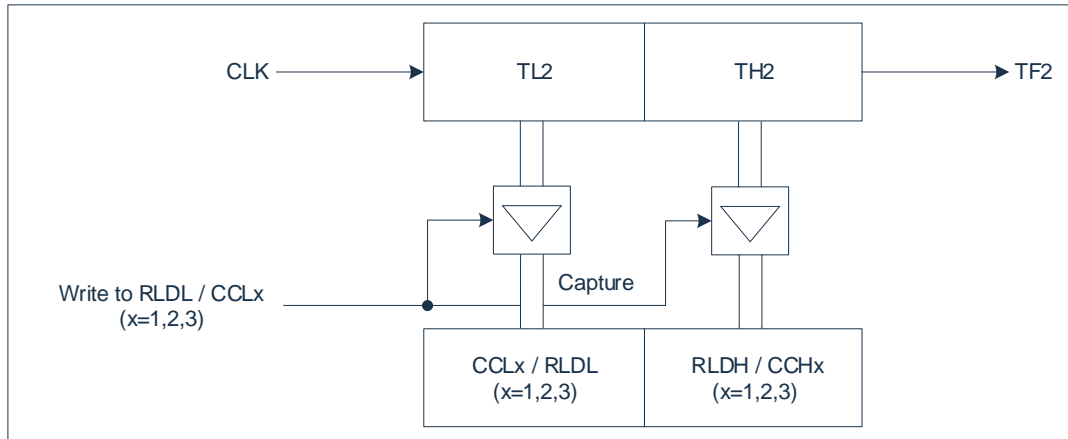


### 10.4.6.2 Capture mode 1

In capture mode 1, the capture operation event is the execution of the instruction to write the lower byte of the capture register. The write register signal (for example, write RLDL) starts the capture operation, and the written value has nothing to do with this function. After the write instruction is executed, the contents of Timer 2 will be latched into the corresponding capture register.

In capture mode 1, the capture event of capture channel 0~3 will not generate an interrupt request flag.

The structure diagram of capture mode 1 is shown in the figure below:



## 11. Timer 3/4

Timer 3/4 is similar to timer 0/1, and is two 16-bit timers. Timer 3 has four working modes, and Timer 4 has three working modes. Compared with Timer0/1, Timer3/4 only provides timing operation.

When the timer is started, the value of the register is incremented every 12 or 4 system cycles.

### 11.1 Overview

Timer 3 and Timer 4 are composed of two 8-bit registers {TH3, TL3} and {TH4, TL4}. Timers 3 and 4 work in the same four modes. Timer3 and Timer4 modes are described as follows:

Mode	M1	M0	Function description
0	0	0	THx[7:0], TLx[4:0] form a 13-bit timer
1	0	1	THx[7:0], TLx[7:0] form a 16-bit timer
2	1	0	TLx[7:0] form a 8-bit auto-reload timer, reload from THx
3	1	1	TL3, TH3 is two 8-bit timers, Timer4 stops counting

### 11.2 Related register

#### 11.2.1 Timer3/4 Control register T34MOD

0xD2	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
T34MOD	TR4	T4M	T4M1	T4M0	TR3	T3M	T3M1	T3M0
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	TR4: Timer4 running control bit; 1 = Timer4 is started; 0 = Timer4 is closed.
Bit6	T4M: Timer 4 clock selection bit; 1= Fsys/4; 0= Fsys/12.
Bit5~Bit4	T4M<1:0>: Timer 4 mode selection bits; 00= Mode 0, 13-bit timer; 01= Mode 1, 16-bit timer; 10= Mode 2, 8-bit auto-reload timer; 11= Mode 3, stop counting.
Bit3	TR3: Timer3 running control bit; 1 = Timer3 is started; 0 = Timer3 is closed.
Bit2	T3M: Timer 3 clock selection bit; 1= Fsys/4; 0= Fsys/12.
Bit1~Bit0	T3M<1:0>: Timer 3 mode selection bits; 00= Mode 0, 13-bit timer; 01= Mode 1, 16-bit timer; 10= Mode 2, 8-bit auto-reload timer; 11= Mode 3, two independent 8-bit timers.

### 11.2.2 Timer3 data register low TL3

0xDA	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TL3	TL37	TL36	TL35	TL34	TL33	TL32	TL31	TL30
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            TL3<7:0>: Timer 3 low-bit data register (also used as timer low-bit).

### 11.2.3 Timer3 data register high TH3

0xDB	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TH3	TH37	TH36	TH35	TH34	TH33	TH32	TH31	TH30
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            TH3<7:0>: Timer 3 high data register (also used as timer high bit).

### 11.2.4 Timer4 data register low TL4

0xE2	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TL4	TL47	TL46	TL45	TL44	TL43	TL42	TL41	TL40
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            TL4<7:0>: Timer 4 low-bit data register (also used as timer low-bit).

### 11.2.5 Timer4 data register high bit TH4

0xE3	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TH4	TH47	TH46	TH45	TH44	TH43	TH42	TH41	TH40
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            TH4<7:0>: Timer 4 high data register (also used as timer high bit).

## 11.3 Timer3/4 interrupt

Timer 3/4 can be enabled or disabled through the EIE2 register, and the high/low priority can also be set through the EIP2 register. The interrupt related bits are as follows:

### 11.3.1 Interrupt mask register EIE2

0xAA	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIE2	SPIIE	I2CIE	WDTIE	ADCIE	PWMIE	-	ET4	ET3
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 <sup>2</sup> C interrupt enable bit; 1= Enable I <sup>2</sup> C interrupt; 0= Disable I <sup>2</sup> C interrupt.
Bit5	WDTIE:	WDT interrupt enable bit; 1= Enable WDT overflow interrupt; 0= Disable WDT overflow interrupt.
Bit4	ADCIE:	ADC interrupt enable bit; 1= enable ADC interrupt; 0= disable ADC interrupt.
Bit3	PWMIE:	PWM total interrupt enable bit; 1= allow all PWM interrupts; 0= disable all PWM interrupts.
Bit2	-	Reserved, must be zero.
Bit1	ET4:	Timer4 interrupt enable bit; 1= Enable Timer4 interrupt; 0= Disable Timer4 interrupt.
Bit0	ET3:	Timer3 interrupt enable bit; 1= Enable Timer3 interrupt; 0= Disable Timer3 interrupt.

### 11.3.2 Interrupt priority control register EIP2

0xBA	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIP2	PSPI	PI2C	PWDT	PADC	PPWM	-	PT4	PT3
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	PSPI:	SPI interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit6	PI2C:	I <sup>2</sup> C interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit5	PWDT:	WDT interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit4	PADC:	ADC interrupt priority control bit; 1= set as high-level interrupt;

- 0= set as low-level interrupt.
- Bit3 PPWM: PWM interrupt priority control bit  
1= set as high-level interrupt;  
0= set as low-level interrupt.
- Bit2 - Reserved, must be zero.
- Bit1 PT4: TIMER4 interrupt priority control bit;  
1= set as high-level interrupt;  
0= set as low-level interrupt.
- Bit0 PT3: TIMER3 interrupt priority control bit;  
1= set as high-level interrupt;  
0= set as low-level interrupt.

### 11.3.3 Peripheral interrupt flag register EIF2

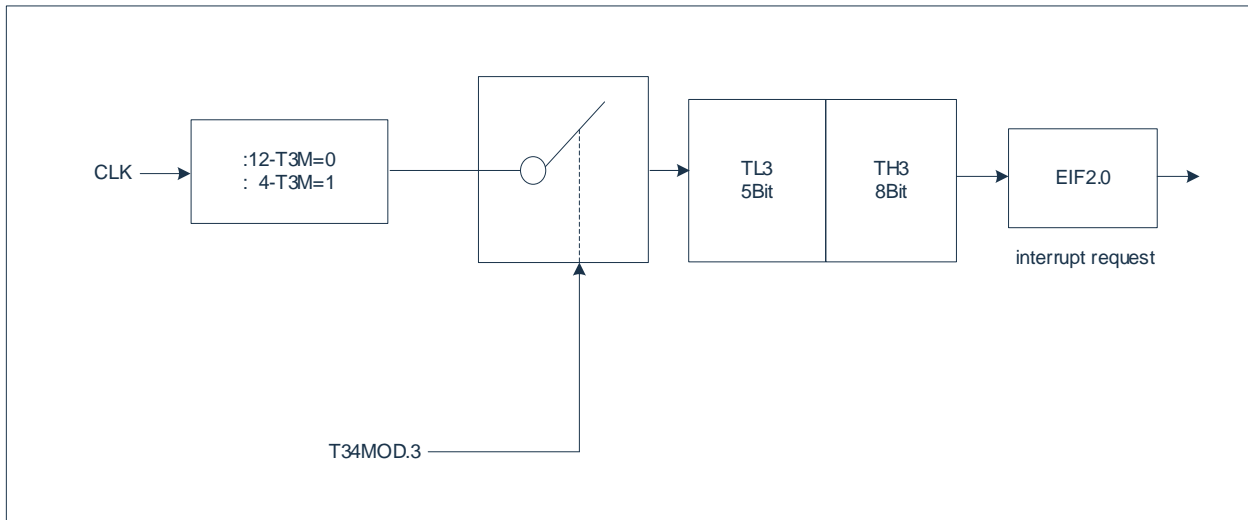
0xB2	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIF2	SPIIF	I2CIF	-	ADCIF	PWMIF	--	TF4	TF3
R/W	R	R	-	R/W	R	--	R/W	R/W
Reset value	0	0	0	0	0	0	0	0

- Bit7 SPIIF: SPI general interrupt indicator bit, read-only;  
1 = SPI generates an interrupt (this bit is automatically cleared after clearing the specific interrupt flag bit);  
0 = SPI does not generate an interrupt.
- Bit6 I2CIF: I<sup>2</sup>C total interrupt indicator bit, read-only;  
1 = I<sup>2</sup>C generates an interrupt (after clearing the specific interrupt flag bit, this bit is automatically cleared);  
0 = I<sup>2</sup>C does not generate an interrupt.
- Bit5 - Reserved, must be zero.
- Bit4 ADCIF: ADC interrupt flag bit;  
1 = ADC conversion is completed and needs to be cleared by software;  
0 = ADC conversion is not completed.
- Bit3 PWMIF: PWM general interrupt indicator bit, read-only;  
1 = PWM generates an interrupt (this bit is automatically cleared after clearing the specific interrupt flag bit);  
0 = PWM does not generate an interrupt.
- Bit2 - Reserved, must be zero.
- Bit1 TF4: Timer4 timer overflow interrupt flag bit;  
1 = Timer4 timer overflows, it is automatically cleared by hardware when entering the interrupt service routine, and can also be cleared by software;  
0 = Timer4 timer has no overflow.
- Bit0 TF3: Timer3 timer overflow interrupt flag bit;  
1 = Timer3 timer overflows, it is automatically cleared by hardware when entering the interrupt service routine, and can also be cleared by software;  
0 = Timer3 timer has no overflow.

## 11.4 Timer3 working mode

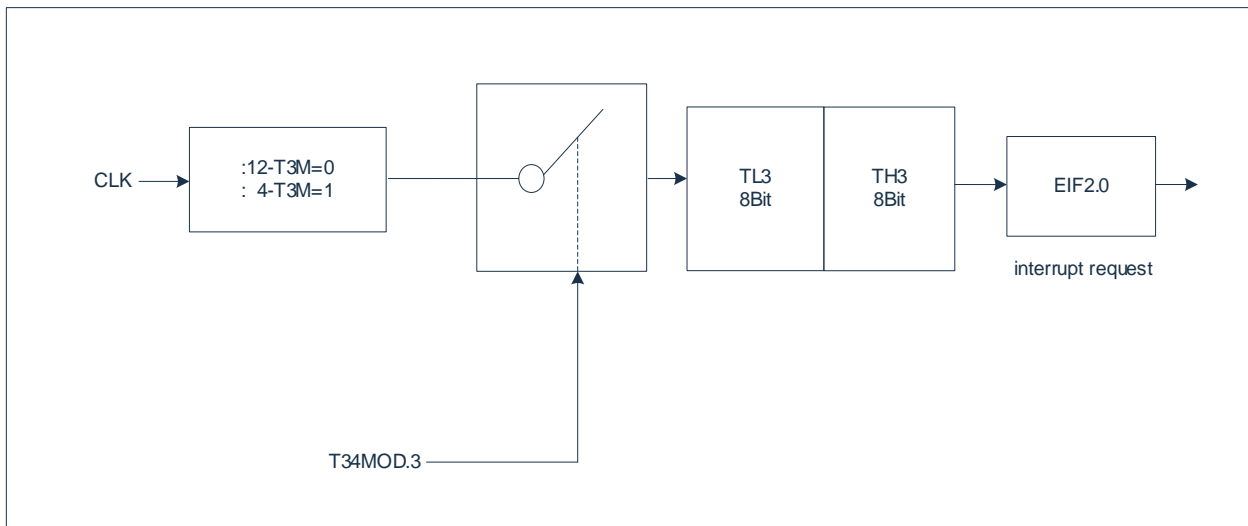
### 11.4.1 T3-Mode 0 (13-bit timer mode)

In this mode, Timer3 is a 13-bit register. When all the bits of the timer are turned from 1 to 0, the timer 3 interrupt flag TF3 is set to 1. The 13-bit register is composed of TH3 and TL3 low 5 bits. The upper 3 bits of TL3 should be ignored. The structure diagram of Timer3 Mode 0 is shown in the figure below:



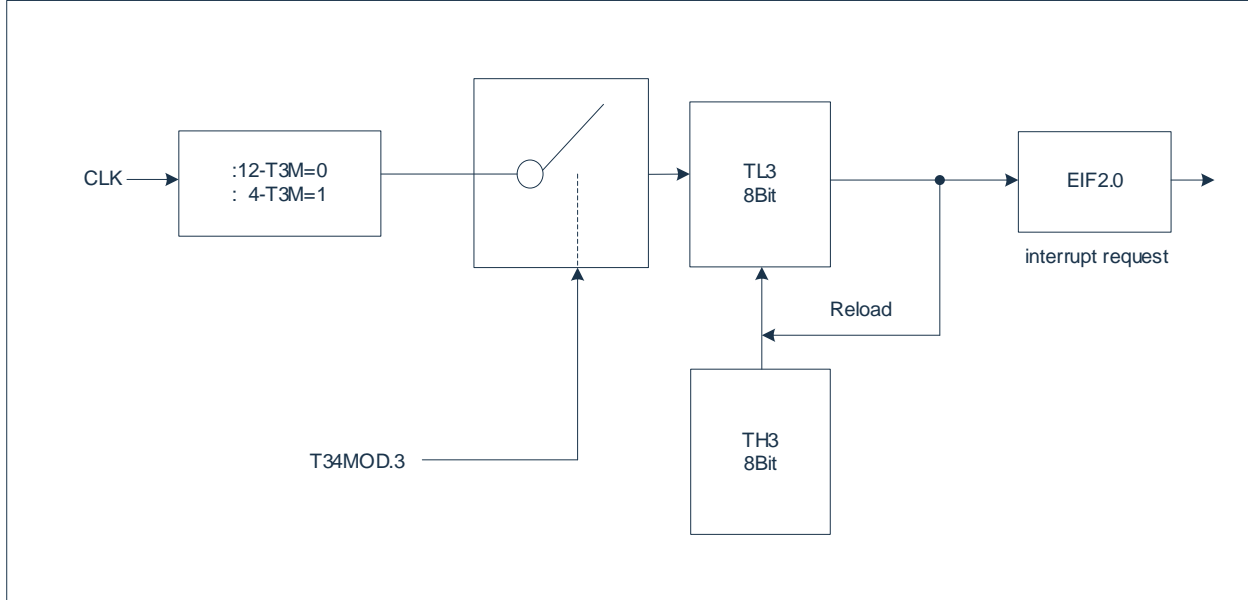
### 11.4.2 T3-Mode 1 (16-bit timing mode)

Mode 1 is the same as Mode 0, except that all 16 bits of the Timer 3 register run in Mode 1. The structure diagram of Timer3 mode 1 is shown in the figure below:



### 11.4.3 T3-Mode 2 (8-bit auto-reload timing mode)

The timer 3 register in mode 2 is an 8-bit timer (TL3) with auto-reload mode, as shown in the figure below. The overflow from TL3 not only sets TF3 to 1, but also reinstalls the contents of TH3 to TL3 by software. The value of TH3 remains unchanged during reinstallation. The structure diagram of Timer3 Mode 2 is shown in the figure below:

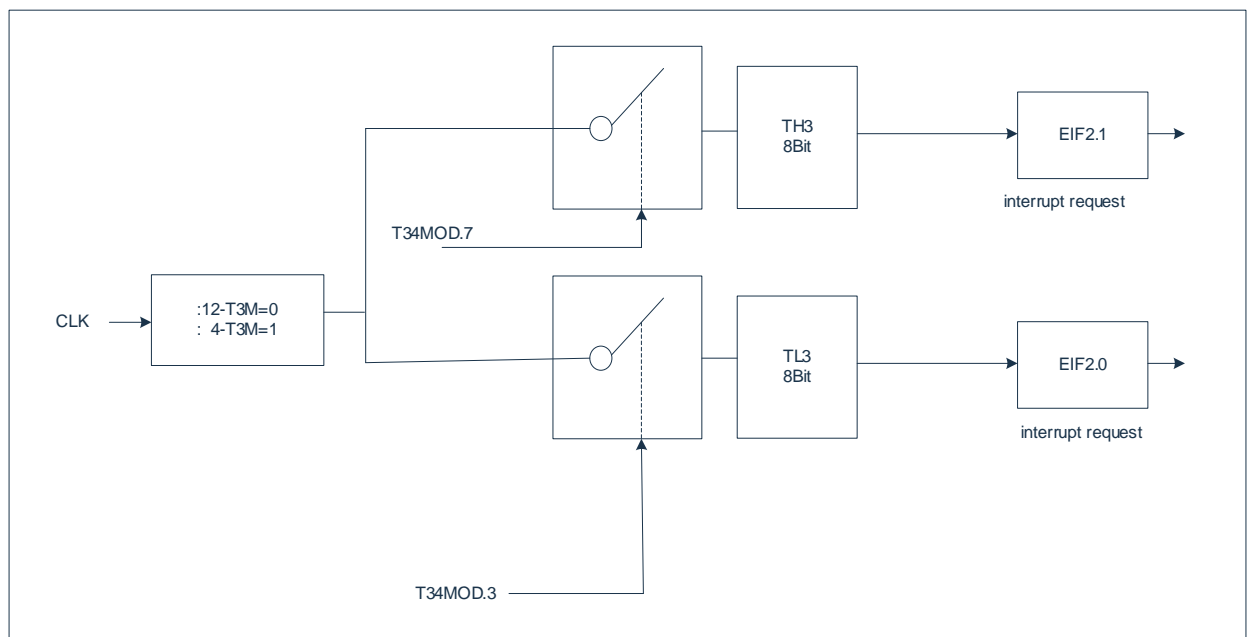


### 11.4.4 T3-Mode 3 (two separate 8-bit timers)

Timer 3 in Mode 3 sets TL3 and TH3 as two independent timers. The logic of Timer 3 Mode 3 is shown in the figure below. TL3 works as an 8-bit timer and uses timer 3 control bits: such as TR3, and TF3.

TH3 works as an 8-bit timer, and uses the TR4 and TF4 flags of timer 4 and controls the timer 4 interrupt.

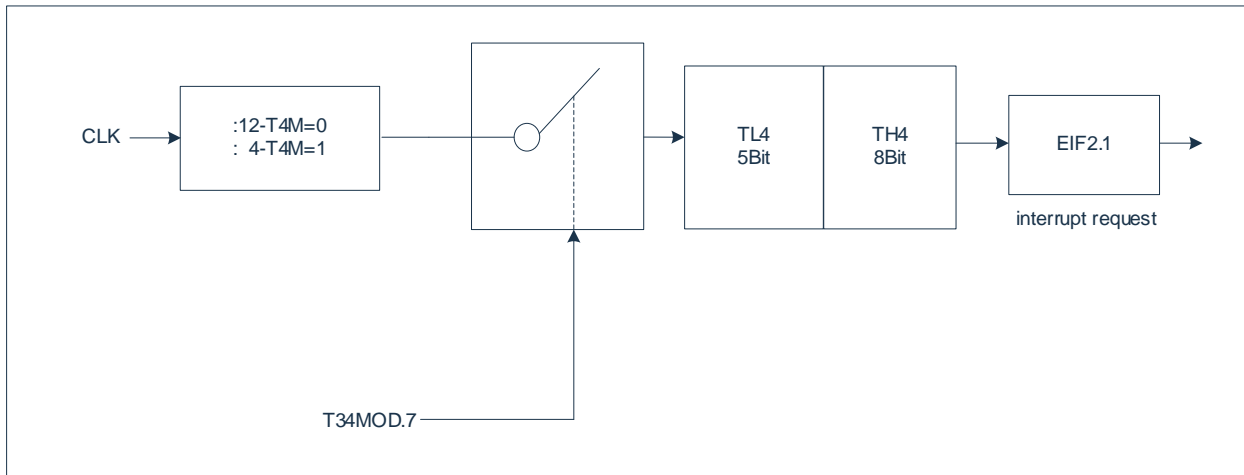
Mode 3 can be used when two 8-bit timers are required. When timer 3 is in mode 3, timer 4 can be turned off by switching to its own mode 3, or it can still be used as a baud rate generator by the serial channel, or in any case that does not require timer 4 interrupts. In application. The structure diagram of Timer3 Mode 3 is shown in the figure below:



## 11.5 Timer4 working mode

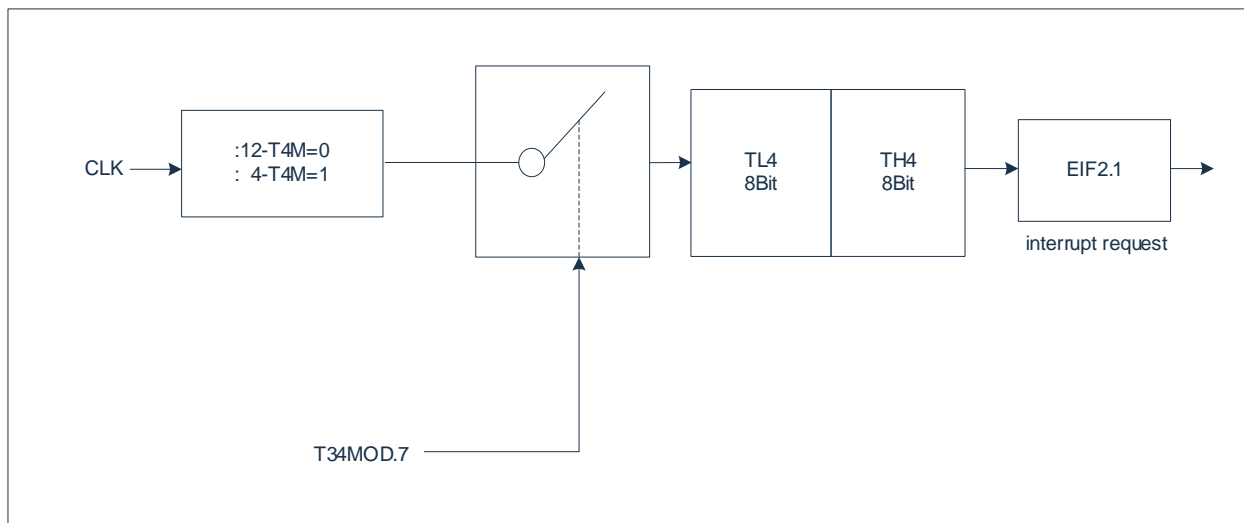
### 11.5.1 T4-Mode 0 (13-bit timing mode)

In this mode, Timer 4 is a 13-bit register. When all the bits of the timer are flipped from 1 to 0, the timer 4 interrupt flag TF4 is set to 1. The 13-bit register consists of 8 bits of TH4 and the lower 5 bits of TL4. The upper three bits of TL4 should be ignored. The structure diagram of Timer4 Mode 0 is shown in the figure below:



### 11.5.2 T4-Mode 1 (16-bit timing mode)

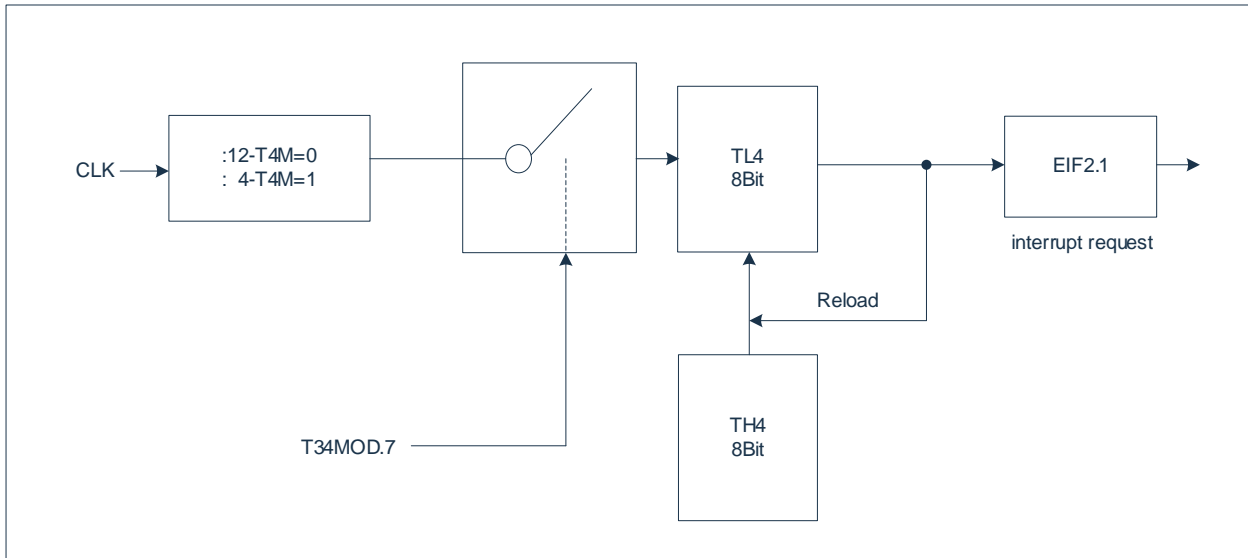
Mode 1 is the same as Mode 0, except that all 16 bits of the Timer 4 register run in Mode 1. The structure diagram of Timer4 Mode 1 is shown in the figure below:





### 11.5.3 T4- Mode 2 (8-bit auto-reload timing mode)

The Timer 4 register in mode 2 is an 8-bit timer (TL4) equipped with auto-reload mode, as shown in the figure below. The overflow from TL4 not only sets TF4 to 1, but also reloads the contents of TH4 to TL4 by software. The value of TH4 remains unchanged during reinstallation. The structure diagram of Timer4 mode 2 is shown in the figure below:



### 11.5.4 T4-Mode 3 (stop counting)

Timer 4 in mode 3 stops counting, and its effect is the same as setting TR4=0.

## 12. LSE Timer (LSE\_Timer)

### 12.1 Overview

LSE timer is a 16-bit up-counting timer with a clock source from the external low-speed clock LSE. When using the LSE timer function, you should first set the LSE module to be enabled, wait for the LSE clock to stabilize (about 1.5s), and then set the LSE count enable. The counter adds 1 to the count value at the rising edge of the LSE clock. When the count value is equal to the timing value, the interrupt flag bit LSECON[0] is set to 1, and the counter starts counting from 0 again. The timing value is set by the register {LSECRH[7:0], LSECRL[7:0]}.

If the LSE timing function is configured before sleep, the LSE oscillator and LSE timer can continue to work when the chip sleeps without being affected. If the LSE timer wake-up function is set before sleep, when the count value is equal to the timer value, the system will wake up.

### 12.2 Related register

#### 12.2.1 LSE timer data register low 8 bits LSECRL

F694H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
LSECRL	LSED7	LSED6	LSED5	LSED4	LSED3	LSED2	LSED1	LSED0
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      LSED<7:0>: LSE timing/wake-up time data lower 8 bits.

#### 12.2.2 LSE timer data register high 8 bits LSECRH

F695H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
LSECRH	LSED15	LSED14	LSED13	LSED12	LSED11	LSED10	LSED9	LSED8
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      LSED<15:8>: The upper 8 bits of the LSE timing/wake-up time data.

### 12.2.3 LSE Timer Control Register LSECON

F696H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
LSECON	LSEEN	LSEWUEN	LSECNTEN	LSESTA	LSEIE	-	-	LSEIF
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	LSEEN: LSE module enable control; 1= enable; 0= disable.
Bit6	LSEWUEN: LSE timer wake-up enable control; 1= enable; 0= disable.
Bit5	LSECNTEN: LSE is used for timer counting enable control; 1= enable; 0= disable.
Bit4	LSESTA: LSE stable status bit, read-only; 1 = LSE is stable; 0 = LSE is not stable.
Bit3	LSEIE: LSE is used as timer interrupt enable control; 1= enable; 0= disable.
Bit2 ~ Bit1	- reservations required both 0.
Bit0	LSEIF: LSE as timer interrupt flag bit (cleared by software); 1= generate interrupt. 0 = No interrupt is generated or the interrupt is cleared.

## 12.3 Interrupt and sleep wakeup

LSE timer can enable or disable interrupt through LSECON register, set high/low priority through EIP3 register, and the related bits of interrupt are as follows.

0xBB	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIP3	--	--	--	--	PLVD	PLSE	--	--
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~Bit4	-- reserved, must be both 0.
Bit3	PLVD: LVD interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit2	PLSE: LSE interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit1~Bit0	-- Reserved, must be 0;

When the count value of the LSE timer is equal to the timing value, the timer interrupt flag bit LSEIF is set to 1. If the global interrupt is enabled (EA=1) and the LSE timer interrupt is enabled (LSEIE=1), the CPU will execute the interrupt service routine.

To use LSE timer interrupt to wake up the sleep mode, you need to turn on LSEEN, LSECNT, LSEWIEN before sleep, and set the time from sleep state to wakeup{LSECRH[7:0], LSECRL[7:0]}. If the global interrupt enable and LSE interrupt enable are turned on before the sleep, after the sleep wakes up, the interrupt service routine will be executed first, and the next instruction of the sleep instruction will be executed after the interrupt returns.

## 12.4 Function description

To use the LSE timer function, you need to set LSEEN=1 to enable the LSE timer function module, and then wait for the LSE clock stable status bit LSETA=1, and then configure the LSE timing value {LSECRH[7:0], LSECRL[7:0]}, finally set LSECNT=1, enable LSE counting, and turn on the LSE counting function. The LSE timer starts counting from 0. When the count value is equal to the timing value, the interrupt flag is set to 1, and the timing value is updated to the value in the timer data register (that is, the LSE timing value is the last time before the count value and the timing value are equal Write the value of {LSECRH[7:0], LSECRL[7:0]}). The minimum timer value of the timer is 1. If the timer value is set to 0, the timer defaults to 1 as the timer value. The formula for calculating the timing time of the LSE timer is as follows:

$$\text{LSE timing time} = \frac{1}{32.768} \times (\{ \text{LSECRH}[7:0], \text{LSECRL}[7:0] \} + 1) \text{ ms}$$

If any bit of LSEEN, LSECNTEN, LSETA is 0, the count value of LSE will be cleared..

## 13. Wake Up Time(WUT)

### 13.1 Overview

Wake Up Timer is a 12-bit, up-counting timer used for wake-up from sleep and a clock source from the internal low-speed clock LSI. It can be used to wake up the system regularly in sleep mode. Configure the timing wake-up time before the system enters sleep, and enable the timing wake-up function. When the chip enters the sleep mode, the WUT starts counting, and when the count value is equal to the set value, the chip enters the sleep wake-up waiting state, timer overflow status bit set to 1, software is used to clean it to 0.

### 13.2 Related register

#### 13.2.1 WUTCRH register

0xBD	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
WUTCRH	WUTEN	TIMER_OV	WUTPS1	WUTPS0	WUTD11	WUTD10	WUTD9	WUTD8
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	WUTEN:	Timed wake-up function enable bit; 1= Timed wake-up function is turned on; 0= Timed wake-up function is turned off.
Bit6	TIMER_OV:	Timer overflow status bit; 1= Timer overflow, need to use software to set to 0. 0= Timer not overflow.
Bit5~Bit4	WUTPS<1:0>:	Timer wake-up counter clock division bits; 00= F/1; 01= F/8; 10= F/32; 11= F/256 .
Bit3~Bit0	WUTD<11:8>:	The upper 4 bits of the timing wake-up time data.

#### 13.2.2 WUTCRL register

0xBC	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
WUTCRL	WUTD7	WUTD6	WUTD5	WUTD4	WUTD3	WUTD2	WUTD1	WUTD0
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	WUTD<7:0>:	Low 8 bits of timing wake-up time data.
-----------	------------	---

## 13.3 Function description

The Principle of the internal wake-up timer is: after the system enters the sleep mode, the CPU and all peripheral circuits stop working, and the internal low-power oscillator LSI starts to work, and its oscillation clock is 125KHz (T<sub>LSI</sub> ≈ 8us). Provide clock for WUT counter.

There are two internal wake-up timing registers: WUTCRH and WUTRCL.

Bit 7 of the WUTCRH register is the internal timing wake-up enable bit:

- WUTEN=1: Turn on the timing wake-up function;
- WUTEN=0: Turn off the timing wake-up.

{WUTCRH[3:0] and WUTCRL[7:0]} form a 12-bit timing wake-up data register. After entering sleep mode, the WUT counter starts timing. When the value of the WUT counter equals the value of the timing wake-up data register, the system oscillator is started, Enter the wake-up waiting state.

Timing wake-up time:  $T=(WUTD[11:0]+1) \times WUTPS \times T_{LSI}$

## 14. Baud rate timer (BRT)

### 14.1 Overview

The chip has one 16-bit baud rate timers BRT, mainly providing the clock for the UART module.

### 14.2 Related registers

#### 14.2.1 BRT module control register BRTCON

F5C0H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
BRTCON	BRTEN	-	-	-	-	BRTCKDI V2	BRTCKDI V1	BRTCKDI V0
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=	prohibit.
Bit6~Bit3	--	Reserved, all must be 0;
Bit2~Bit0	BRTCKDIV<2:0>	BRT timer prescaler selection bit;
	000=	Fsys/1;
	001=	Fsys/2;
	010=	Fsys/4;
	011=	Fsys/8;
	100=	Fsys/16;
	101=	Fsys/32;
	110=	Fsys/64;
	111=	Fsys/128.

#### 14.2.2 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>: BRT timer load value lower 8 bits;

#### 14.2.3 BRT timer data load upper 8 bits register BRTDH

F5C2H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
BRTDH	BRTDH7	BRTDH6	BRTDH5	BRTDH4	BRTDH3	BRTDH2	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 ~Bit0 BRTDH<7:0>: BRT timer load value high 8 bits;

## 14.3 Function description

BRT has a 16-bit up counter, its clock comes from the prescaler circuit, the prescaler clock is The timer prescaler selection bit BRTCKDIV is determined, and the initial value of the counter is loaded by {BRTDH, BRTDL}.

When the timer enable bit BRTEN=1, the counter starts to work. When the value of the 16-bit counter is equal to FFFFH, the BRT counter overflows. After overflow, the initial value of the count is automatically loaded into the counter, and then the count is restarted.

The overflow signal of the BRT counter is specially provided to the UART module as the clock source of the baud rate. When it overflows, no interrupt will be generated, and there is no corresponding interrupt structure. In the debug mode of BRT, its clock will not stop. If the UART module has started sending or receiving data, even if the chip enters the pause state, the UART will complete the entire sending or receiving process.

BRT timer overflow rate:

$$BRTov = \frac{F_{sys}}{(65536 - \{BRTDH, BRTDL\}) \times 2^{BRTCKDIV}}$$



## 15. Cyclic redundancy check (CRC)

### 15.1 Overview

In order to ensure safety during operation, the IEC61508 standard requires data to be confirmed even when the CPU is running. This general-purpose CRC module can perform CRC calculations as a peripheral function during CPU operation. The universal CRC module specifies the data to be confirmed by the program to perform CRC check, not limited to the code flash area but can be used for multi-purpose checking.

CRC generator polynomial uses " $X^{16}+X^{12}+X^5+1$ " of CRC16-CCITT.

### 15.2 Related register

#### 15.2.1 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>    Enter 8-bit data that requires CRC operation.

#### 15.2.2 CRC operation 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>    CRC operation result low 8-bit data

#### 15.2.3 CRC operation 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 high 8-bit data of CRC operation result

## 15.3 Function description

After writing the CRCIN register, after a system clock, save the CRC operation result to the CRCDL/CRCDH register. If necessary, read the previous calculation data before writing overwrite, otherwise it will be overwritten by the new calculation result.

For example: send data "12345678H", write values to the CRCIN register in the order of "12H", "34H", "56H", and "78H". After writing, read from the CRCDL/CRCDH register as CRCDL=0xF0, CRCDH=0x67, That is, the bit sequence of the data "12345678H", the result of the CRC operation is 0x67F0. The register operation is as follows:

```
CRCIN=0x12; //Send the first number
```

```
CRCIN=0x34; //Send the second number
```

```
CRCIN=0x56; //Send the third number
```

```
CRCIN=0x78; //Send the fourth number
```

```
resl=CRCDL; //Read the low 8 bits of the CRC operation result to the variable resl
```

```
resh=CRCDH; //Read the high 8 bits of the CRC operation result to the variable resh
```

## 16. Buzzer driver (BUZZER)

### 16.1 Overview

Buzzer driver module is composed of an 8-bit counter, a clock driver, and a control register. The buzzer driver output is a 50% duty square wave, the frequency is set by the registers BUZCON and BUZDIV, and its frequency output can cover a wider range.

### 16.2 Related register

#### 16.2.1 BUZZER control register BUZCON

0xBF	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
BUZCON	BUZEN	-	-	-	-	-	BUZCKS1	BUZCKS0
R/W	R/W	R	R	R	R	R	R/W	R/W
Reset value	0	0	0	0	0	0	0	0

Bit7	BUZEN:	BUZZER enable bit;
	1=	enable;
	0=	disable.
Bit6 ~ Bit2	-	Reserved, must be 0's.
Bit1~Bit0	BUZCKS<1:0>:	BUZZER division ratio selection bits;
	00=	Fsys/8;
	01=	Fsys/16;
	10=	Fsys/32;
	11=	Fsys/64.

#### 16.2.2 BUZZER frequency control register

0xBE	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
BUZDIV	BUZDIV7	BUZDIV6	BUZDIV5	BUZDIV4	BUZDIV3	BUZDIV2	BUZDIV1	BUZDIV0
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	BUZDIV<7:0>:	BUZZER frequency selection bit;
	0x00=	no square wave output;
	others =	$F_{buz} = F_{sys} / (2 * CLKDIV * BUZCKS)$ .

Note: It is not recommended to modify BUZDIV during BUZEN=1

## 16.3 Function description

When using the buzzer, you need to configure the corresponding port as a buzzer driver output port. For example, to configure P16 as a buzzer driver output port, the configuration is as follows:

```
P16CFG = 0x04; // P16 is configured as a buzzer driver output port.
```

By configuring the relevant registers of the buzzer driver module, you can set the buzzer driver output port to output differently. Frequency of. For example:

- 1) Set  $F_{sys} = 8\text{MHz}$ ,  $\text{BUZCKS}\langle 1:0 \rangle = 01$ ,  $\text{BUZDIV} = 125$  The output frequency of the buzzer driver is:  $F_{buz} = 8\text{MHz} / (2 * 125) / 16 = 2\text{KHz}$
- 2) Set  $F_{sys} = 16\text{MHz}$ ,  $\text{BUZCKS}\langle 1:0 \rangle = 11$ ,  $\text{BUZDIV} = 125$  buzzer driver output frequency:  $F_{buz} = 16\text{MHz} / (2 * 125) / 64 = 1\text{KHz}$
- 3) set  $F_{sys} = 24\text{MHz}$ ,  $\text{BUZCKS}\langle 1:0 \rangle = 11$ ,  $\text{BUZDIV} = 94$  buzzer driver output The frequency is:  $F_{buz} = 24\text{MHz} / (2 * 94) / 64 = 2\text{KHz}$

Choose different system clock frequency and buzzer driver clock division ratio to get different output frequency. The buzzer driver output frequency is shown in the following table:

BUZCKS<1:0>	Fbuz@Fsys=8MHz	Fbuz@Fsys=16MHz	Fbuz@Fsys=24MHz	Fbuz@Fsys=48MHz
00	2KHz~500KHz	4KHz~1MHz	6KHz~1.5MHz	12KHz~ 3MHz
01	1KHz~250KHz	2KHz~500KHz	3KHz~750KHz	6KHz~1.5MHz
10	0.5KHz~125KHz	1KHz~	250KHz 1.5KHz~375KHz	3KHz~750KHz
11	0.25KHz~62.5KHz	0.5KHz~125KHz	0.75KHz~187.5KHz	1.5KHz~375KHz

## 17. PWM module

### 17.1 Overview

The PWM module supports 6-channel PWM generators, which can be configured as independent 6-channel PWM outputs (PG0-PG5), or as 3 groups of synchronous PWM outputs, or 3 pairs of complementary with dead-zone programming generators. PWM output, where PG0-PG1, PG2-PG3, PG4-PG5 are each pair.

Each PWM has an independent 16-bit period register and a 16-bit duty cycle register (comparison data register) to configure the period of the PWM output and adjust the duty cycle. And each PWM has an independent clock frequency division control register, and each pair of PWM shares an 8-bit prescaler control register.

Each PWM can be configured in edge-aligned counting. Each PWM can be set to output in single mode (generating a PWM signal cycle) or auto-loading (continuous output of PWM waveform), and its output polarity can also be set by the output polarity controller.

PWM also supports interrupt function. The 6-channel PWM generator provides a total of 12 interrupt flags, including zero point interrupt, and downward comparison interrupt. These interrupts share an interrupt vector entry.

### 17.2 Feature

PWM module has the following features:

- ◆ 6 independent 16-bit PWM control modes.
  - 6 independent outputs: PG0, PG1, PG2, PG3, PG4, PG5;
  - 3 sets of complementary PWM pair outputs: (PG0-PG1), (PG2-PG3), (PG4-PG5), programmable dead time can be inserted;
  - Three groups of synchronous PWM pair output: (PG0-PG1), (PG2-PG3), (PG4-PG5), each group of PWM pair pins are synchronized.
- ◆ Support group control, PG0, PG2, PG4 output synchronization, PG1, PG3, PG5 output synchronization.
- ◆ Support edge alignment.
- ◆ Support single mode or auto-load mode.
- ◆ Each PWM has independent polarity control.
- ◆ The PWM edge can trigger the start of AD conversion.

## 17.3 Port configuration

Before using the PWM module, you need to configure the corresponding port as a PWM channel. The PWM channel is marked with PG0~PG5 in the multiplexing function allocation table, corresponding to PWM channels 0~5.

The allocation of PWM channels is controlled by the corresponding port configuration registers, for example:

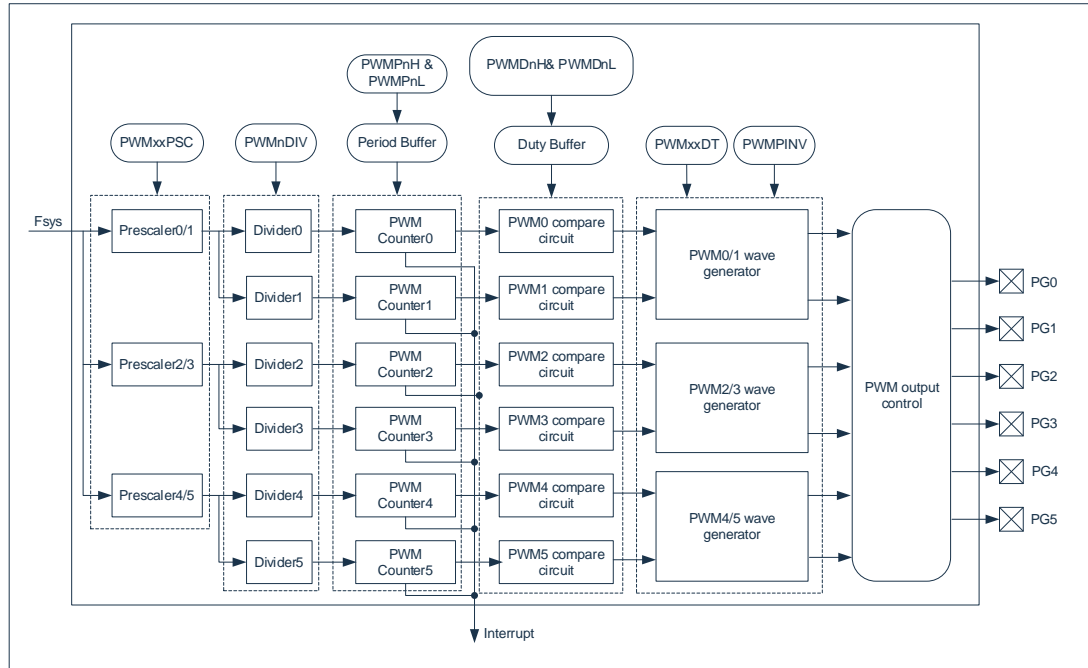
```
P00CFG=0x05;    //Configure P00 to PG0 channel
P01CFG=0x05;    //Configure P01 to PG1 Channel
P02CFG=0x05;    //Configure P02 to PG2 channel
P03CFG=0x05;    //Configure P03 to PG3 channel

P04CFG=0x05;    //Configure P04 to PG4 chanel
P05CFG=0x05;    //Configure P05 to PG5 channel
```

## 17.4 Functional description

### 17.4.1 Functional block diagram

The enhanced PWM is composed of a clock control module, a PWM counter module, an output comparison unit, a waveform generator, a brake protection module (fault detection) and an output controller. Its structure diagram is shown in the figure below:



### 17.4.2 Edge alignment

In edge-aligned mode, the 16-bit PWM counter CNTn starts counting down in each cycle and compares it with the value CMPn latched in the PWMDnH/PWMDnL register. When CNTn=CMPn, PGn outputs a high level and PWMnDIF is set to 1. CNTn continues to count down to 0, at this time PGn will output low level, PWMnZIF is set to 1. When CNTn counts to zero, if PWMnCNTM=1, CMPn and PERIODn will be reloaded.

The related parameters of edge alignment are as follows:

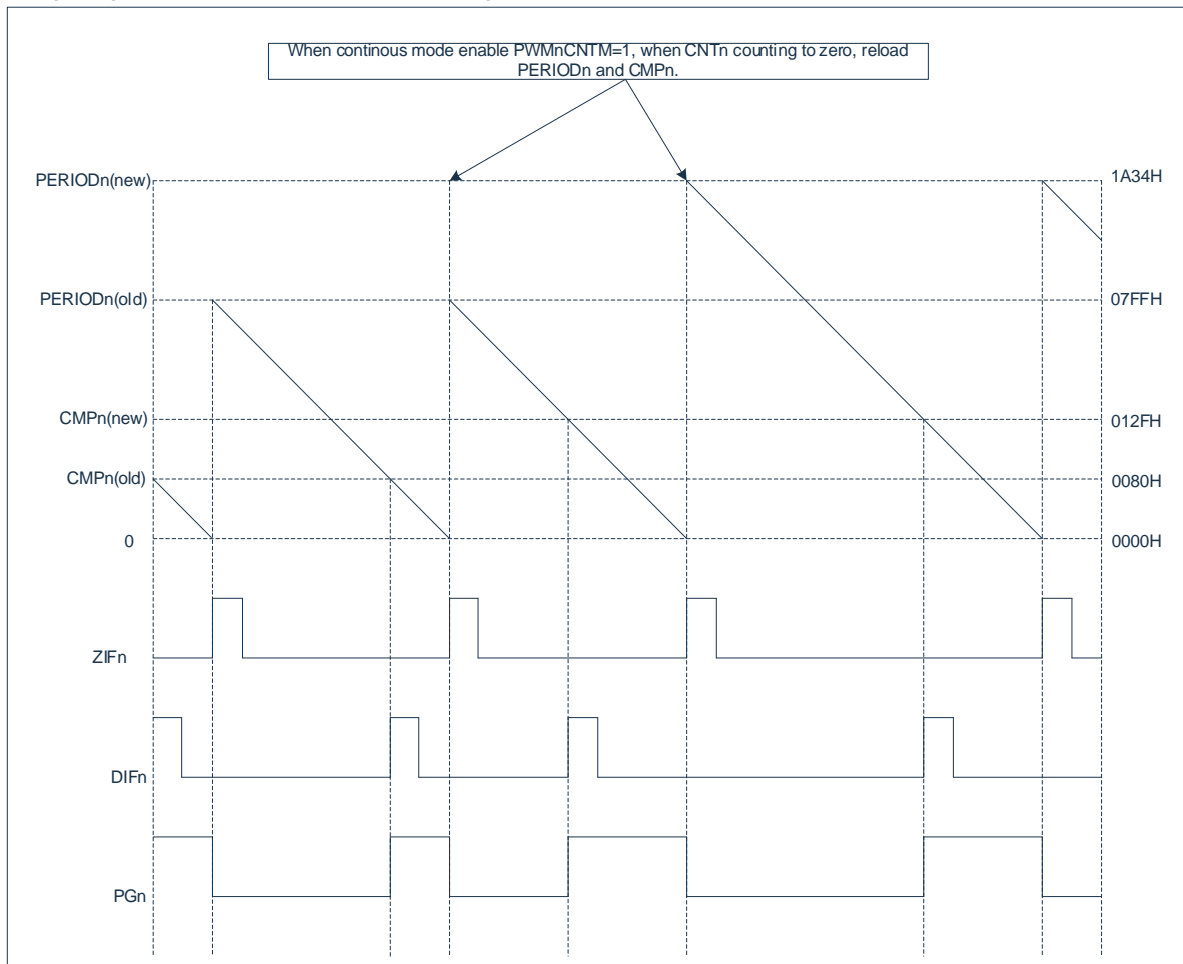
$$\text{Voltage High level timing} = (CMPn+1) \times T_{pwm}(CMPn \geq 1)$$

$$\text{Period} = (PERIODn+1) \times T_{pwm}$$

$$\text{duty cycle} = (CMPn+1)/(PERIODn+1) \quad (CMPn \geq 1)$$

CMPn=0, The duty cycle is 0%.

The edge alignment sequence is shown in the figure below:





### 17.4.3 Complementary mode

6-channel PWM can be set as 3 groups of complementary PWM pairs. In complementary mode, the period, duty cycle and clock division control of PG1, PG3, and PG5 are respectively determined by PG0, PG2, and PG4 related registers, that is, in addition to the corresponding output enable control bit (PWMnOE), PG1, PG3, PG5 The output waveform is no longer controlled by its own register.

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

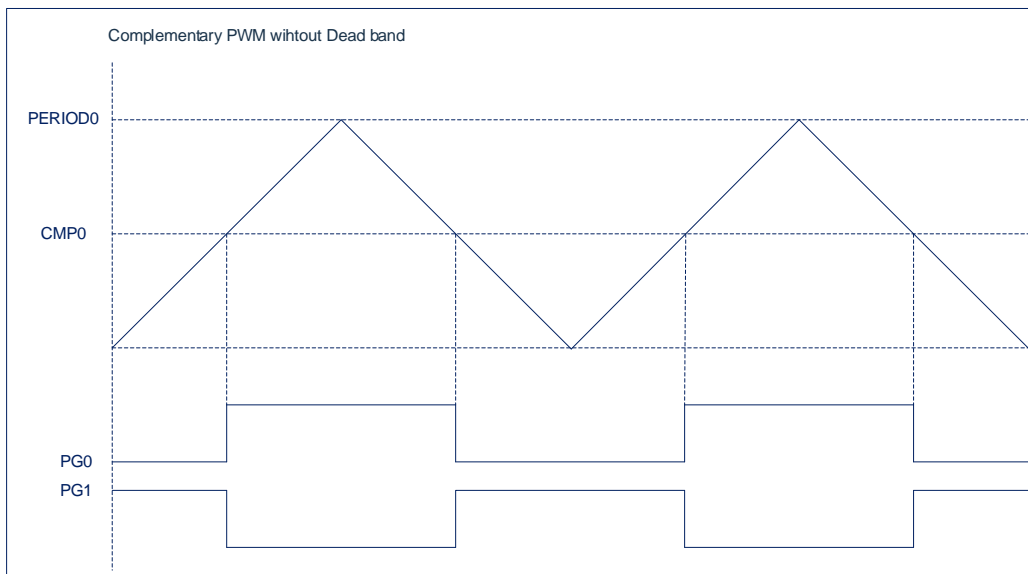
$$\text{PWM0/1 dead-time: } (\text{PWM01DT}+1) \cdot T_{\text{PWM0}} ;$$

$$\text{PWM2/3 dead-time: } (\text{PWM23DT} +1) \cdot T_{\text{PWM2}} ;$$

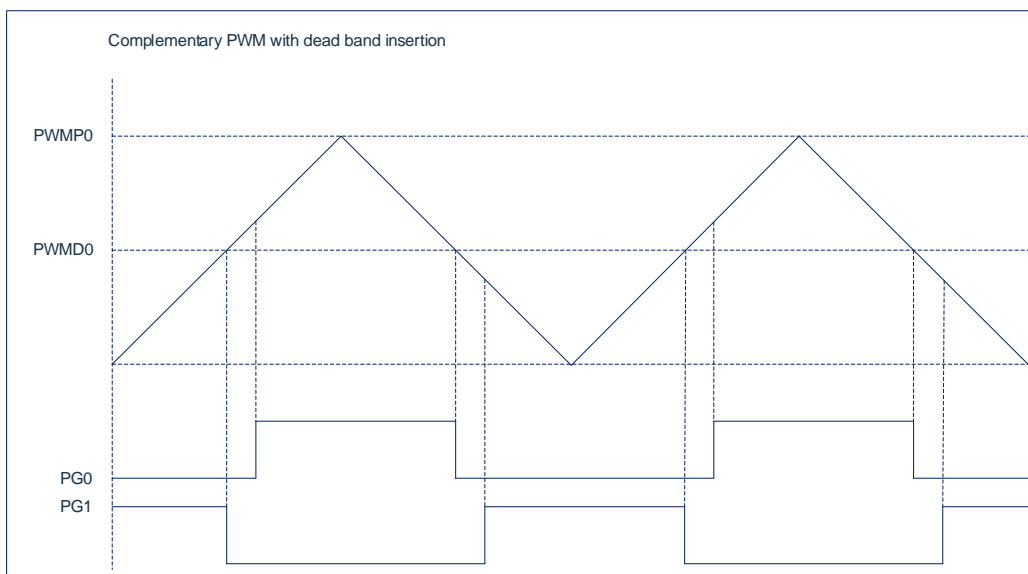
$$\text{PWM4/5 dead time: } (\text{PWM45DT}+1) \cdot T_{\text{PWM4}} ;$$

$T_{\text{PWM0}}/T_{\text{PWM2}}/T_{\text{PWM4}}$  are the clock source periods of PG0/PG2/PG4 respectively.

Take PG0/PG1 as an example, the waveform diagram without dead zone in complementary mode is shown in the figure below:



Take PG0/PG1 as an example, the waveform diagram with dead zone in complementary mode is shown in the figure below:



#### 17.4.4 Synchronous mode

6-channel PWM can be set It is 3 sets of synchronous PWM pairs. In synchronous mode, the period, duty cycle and clock division control of PG1, PG3, and PG5 are respectively determined by PG0, PG2, and PG4 related registers, that is, in addition to the corresponding output enable control bit (PWMnOE), PG1, PG3, PG5 The output waveform is no longer controlled by its own register. 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.

## 17.5 PWM related register

### 17.5.1 PWM control register PWMCON

F120H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWMCON	--	PWMRUN	PWMMODE1	PWMMODE0	GROUPEN	ASYMEN	CNTTYPE	--
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, must be zero.
Bit6	PWMRUN:	PWM clock prescaler, clock divider enable bit; 1= disable (PWMmnPSC, PWMmnDIV are cleared to 0); 0= enable.
Bit5~Bit4	PWMMODE<1:0>:	PWM mode control bits; 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		- Reserved, must be zero.
Bit1		- Reserved, must be zero.
Bit0		- Reserved, must be zero.

### 17.5.2 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/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	--	Reserved, must be both 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.

### 17.5.3 PWM0/1 clock prescaler control register PWM01PSC

F123H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWM01PSC	PWM01PSC7	PWM01PSC6	PWM01PSC5	PWM01PSC4	PWM01PSC3	PWM01PSC2	PWM01PSC1	PWM01PSC0
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      PWM01PSC<7:0>: PWM channel 0/1 prescaler control bit;  
                   00= prescaler clock stop, PWM0/1 counter stop ;  
                   Other = (PWM01PSC+1) frequency division of the system clock.

### 17.5.4 PWM2/3 clock prescaler control register PWM23PSC

F124H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWM23PSC	PWM23PSC7	PWM23PSC6	PWM23PSC5	PWM23PSC4	PWM23PSC3	PWM23PSC2	PWM23PSC1	PWM23PSC0
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      PWM23PSC<7:0>: PWM channel 2/3 prescaler control bit;  
                   00= prescaler clock stop, PWM2/3 counter stop ;  
                   Other = (PWM23PSC+1) frequency division of the system clock.

### 17.5.5 PWM4/5 clock prescaler control register PWM45PSC

F125H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWM45PSC	PWM45PSC7	PWM45PSC6	PWM45PSC5	PWM45PSC4	PWM45PSC3	PWM45PSC2	PWM45PSC1	PWM45PSC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W/W	R/W	R/W
reset value	0	0	0	0	0	0	0	0

Bit7~Bit0      PWM45PSC<7:0>: PWM channel 4/5 prescaler control bit;  
                   00= prescaler clock stop, PWM4/5 counter stop ;  
                   Others = (PWM45PSC+1) frequency division of the system clock.

### 17.5.6 PWM clock division control register PWMnDIV(n=0-5)

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWMnDIV	-	-	-	-	-	PWMnDIV2	PWMnDIV1	PWMnDIV0
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 PWMnDIV (n=0-5) address: F12AH, F12BH, F12CH, F12DH, F12EH, F12FH.

- Bit7 ~ Bit3 - Reserved, must be 0's.
- Bit2~Bit0 PWMnDIV<2:0>: PWM channel n clock division control bit;
- 000= Fpwmn-PSC/2;
  - 001= Fpwmn-PSC/4;
  - 010= Fpwmn-PSC/8;
  - 011= Fpwmn-PSC/16 ;
  - 100= Fpwmn-PSC; scaled
  - Others= Fsys (system clock);  
(PSC is the pre-clock).

### 17.5.7 PWM data load enable control register PWMLOADEN

F129H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWMLOADEN	--	--	PWM5LE	PWM4LE	PWM3LE	PWM2LE	PWM1LE	PWM0LE
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 -- Reserved, must be both 0.
- Bit5~Bit0 PWMnLE: Data loading enable bit of PWM channel n (n=0-5) (hardware cleared after loading is completed);
- 1= Enable loading period, duty cycle data (PERIODn, CMPn, CMPDn).
  - 0 = Writing 0 is invalid.

### 17.5.8 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/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 -- Reserved, must be both 0.
- Bit5~Bit0 PWMnPINV: PWM channel n output polarity control bit (n=0-5);
- 1= reverse output;
  - 0= normal output.

### 17.5.9 PWM Counter Mode Control Register PWMCNTM

F127H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWMCNTM	--	--	PWM5CNTM	PWM4CNTM	PWM3CNTM	PWM2CNTM	PWM1CNTM	PWM0CNTM
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 -- reserved, must be 0.  
 Bit5~Bit0 PWMnCNTM: PWM channel n counter mode control bits (n=0-5);  
 1= automatic loading mode;  
 0= One-shot mode.

### 17.5.10 PWM counter enable control register PWMCNTE

F126H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWMCNTE	--	--	PWM5CNTE	PWM4CNTE	PWM3CNTE	PWM2CNTE	PWM1CNTE	PWM0CNTE
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 -- reserved, must be 0.  
 Bit5~Bit0 PWMnCNTE: PWM channel n counter enable control bit (n=0-5);  
 1= PWMn counter is turned on (PWMn starts to output);  
 0= PWMn counter is stopped (software writes 0 to stop the counter and clear the counter value).  
 (This bit is cleared to 0 by hardware when the brake is triggered; this bit is cleared to 0 by hardware when the one-shot mode is completed)

### 17.5.11 PWM counter mode control register PWMCNTCLR

F128H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWMCNTCLR	--	--	PWM5CNTCLR	PWM4CNTCLR	PWM3CNTCLR	PWM2CNTCLR	PWM1CNTCLR	PWM0CNTCLR
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 -- reserved, must be both 0.  
 Bit5~Bit0 PWMnCNTCLR: PWM channel n counter clearing control bit (n=0-5) (hardware automatic clearing);  
 1= PWMn counter clearing;  
 0= writing 0 is invalid.

### 17.5.12 PWM period data register low 8 bits PWMPnL (n=0-5)

	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=0-5) address: F130H, F132H, F134H, F136H, F138H, F13AH.

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

### 17.5.13 PWM period data register high 8 bits PWMPnH (n=0-5)

	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=0-5) address: F131H, F133H, F135H, F137H, F139H, F13BH.

Bit7~Bit0 PWMPnH<7:0>: The upper 8 bits of the PWM channel n period data register.

### 17.5.14 PWM compare data register low 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>: The lower 8 bits of the PWM channel n compare data (duty cycle data) register.

### 17.5.15 PWM compare data register high 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>: The upper 8 bits of the PWM channel n compare data (duty cycle data) register.

### 17.5.16 PWM Dead Band Enable Control Register PWMDTE

F160H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWMDTE	--	--	--	--	--	PWM45DTE	PWM23DTE	PWM01DTE
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~Bit3 -- reserved, must be both 0.

Bit2 PWM45DTE: PWM4/5 channel dead-band delay enable bit;  
1= enable;  
0= disable.

Bit1 PWM23DTE: PWM2/3 channel dead-band delay enable bit;  
1= enable;  
0= disable.

Bit0 PWM01DTE: PWM0/1 channel dead-band delay enable bit;  
1= enable;  
0= disable.

### 17.5.17 PWM0/1 Dead Time Delay Data Register PWM01DT

F161H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWM01DT	PWM01DT7	PWM01DT6	PWM01DT5	PWM01DT4	PWM01DT3	PWM01DT2	PWM01DT1	PWM01DT0
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-band delay data register.

### 17.5.18 PWM2/3 Dead Time Delay Data Register PWM23DT

F162H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWM23DT	PWM23DT7	PWM23DT6	PWM23DT5	PWM23DT4	PWM23DT3	PWM23DT2	PWM23DT1	PWM23DT0
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-band delay data register.

### 17.5.19 PWM4/5 Dead Time Delay Data Register PWM45DT

F163H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWM45DT	PWM45DT7	PWM45DT6	PWM45DT5	PWM45DT4	PWM45DT3	PWM45DT2	PWM45DT1	PWM45DT0
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-band delay data register.



## 17.6 PWM Interrupt

PWM has a total of 12 interrupt flags, including 6 zero interrupt flags, 6 downward comparison interrupt flags, and interrupt flag bits. The generation has nothing to do with whether the corresponding interrupt enable bit is turned on. To enable any type of PWM interrupt, you need to enable the global interrupt enable bit (EA=1) and the PWM total interrupt enable bit PWMIE to successfully configure the PWM interrupt function. All PWM interrupts share an interrupt vector entry, so after entering the interrupt service routine, the user can judge which type of interrupt is generated by the interrupt flag bit.

The interrupt enable and priority of the PWM can be set by the following relevant register bits.

### 17.6.1 Interrupt mask register EIE2

0xAA	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIE2	SPIIE	I2CIE	WDTIE	ADCIE	PWMIE	-	ET4	ET3
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 <sup>2</sup> C interrupt enable bit; 1= Enable I <sup>2</sup> C interrupt; 0= Disable I <sup>2</sup> C interrupt.
Bit5	WDTIE: WDT interrupt enable bit; 1= Enable WDT overflow interrupt; 0= Disable WDT overflow interrupt.
Bit4	ADCIE: ADC interrupt enable bit; 1= Enable ADC interrupt; 0= Disable ADC interrupt.
Bit3	PWMIE: PWM total interrupt enable bit; 1= Allow all PWM interrupts; 0= Disable all PWM interrupts.
Bit2	- Reserved, must be zero.
Bit1	ET4: Timer4 interrupt enable bit; 1= Enable Timer4 interrupt; 0= Disable Timer4 interrupt.
Bit0	ET3: Timer3 interrupt enable bit; 1= Enable Timer3 interrupt; 0= Disable Timer3 interrupt.

### 17.6.2 Interrupt priority control register EIP2

0xBA	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIP2	PSPI	PI2C	PWDT	PADC	PPWM	-	PT4	PT3
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	PSPI:	SPI interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit6	PI2C:	I <sup>2</sup> C interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit5	PWDT:	WDT interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit4	PADC:	ADC interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit3	PPWM:	PWM interrupt priority control bit 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit2	-	Reserved, must be zero.
Bit1	PT4:	TIMER4 interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit0	PT3:	TIMER3 interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.

### 17.6.3 PWM Zero Interrupt Mask Register PWMZIE

F169H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWMZIE	--	--	PWM5ZIE	PWM4ZIE	PWM3ZIE	PWM2ZIE	PWM1ZIE	PWM0ZIE
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	--	Reserved, all must be 0.
Bit5~Bit0	PWMnZIE:	PWM channel n zero interrupt mask bit (n=0-5); 1= enable interrupt; 0= disable interrupt.

### 17.6.4 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/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 -- Reserved, all must be 0.

Bit5~Bit0 PWMnDIE: PWM channel n downward comparison interrupt mask bit (n=0-5);  
 1= enable interrupt;  
 0= disable interrupt.

### 17.6.5 PWM Zero Interrupt Flag Register PWMZIF

F16DH	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWMZIF	--	--	PWM5ZIF	PWM4ZIF	PWM3ZIF	PWM2ZIF	PWM1ZIF	PWM0ZIF
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 -- Reserved, all must be 0.

Bit5~Bit0 PWMnZIF: PWM channel n zero interrupt flag bit (n=0-5);  
 1= generate interrupt (cleared by software);  
 0= not generate interrupt.

### 17.6.6 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/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 -- Reserved, all must be 0.

Bit5~Bit0 PWMnDIF: PWM channel n downward comparison interrupt flag bit (n=0-5);  
 1= generate interrupt (cleared by software);  
 0= not generate interrupt.

## 18. SPI module

### 18.1 Overview

This SPI is a fully configurable SPI master/slave device, allowing users to configure the polarity and phase of the serial clock signal SCLK. The serial clock line (SCLK) is synchronized with the shift and sampling of information on two independent serial data lines, and SPI data is sent and received at the same time. SPI allows MCU to communicate with serial peripherals. It can also communicate between processors in a multi-host system. It is a technology-independent design that can be implemented in various process technologies.

The SPI system is flexible enough to directly connect with many standard product peripherals from multiple manufacturers. In order to accommodate most of the available synchronous serial peripherals, the clock control logic allows the clock polarity and phase to be selected. The system can be configured as a master device or a slave device. When the SPI is configured as a master device, the software selects one of eight different bit rates for the serial clock, up to the system clock divided by 4 ( $F_{sys}/4$ ).

The SPI slave chip is selected to address the SPI slave device to exchange serial data. When SPI is used as a master device, SPI is automatically driven by the slave selection control register SSCR. The SPI controller includes logic error detection to support inter-processor communication. For example, the write conflict detector can indicate when to write data to the serial shift register during transmission..

SPI has the following characteristics:

- ◆ Full-duplex synchronous serial data transmission.
- ◆ Support master/slave mode.
- ◆ Support multi-host system.
- ◆ System error detection.
- ◆ An interrupt is generated.
- ◆ Support speed up to 1/4 of the system clock ( $F_{sys} \leq 24\text{MHz}$ ).
- ◆ The bit rate generates 1/4, 1/8, 1/16, 1/32, 1/64, 1/128, 1/256, 1/512 of the system clock.
- ◆ Supports four transmission formats.
- ◆ The simple interface allows easy connection to the microcontroller.

## 18.2 SPI port configuration

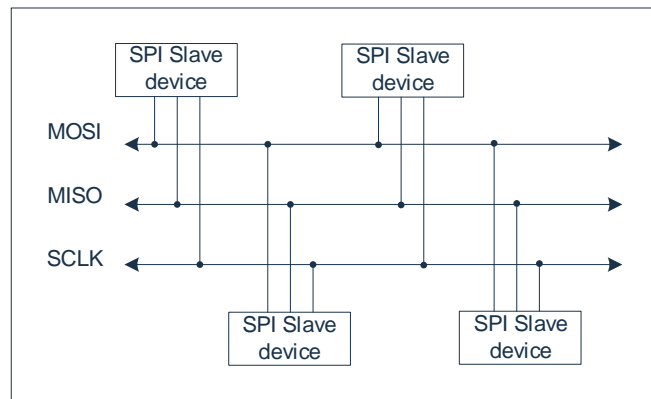
To use the SPI function, you need to configure the relevant port as an SPI channel, and select the corresponding port input through the communication input port register. For example, configure P00, P01, P02, and P03 as SPI communication ports.

The configuration code is as follows:

```

PS_SCLK = 0x00; //Select P00 as the SCLK channel of SPI
PS_MOSI = 0x01; //Select P01 as the MOSI channel of SPI
PS_MISO = 0x02; //Select P02 as the MISO channel of SPI
PS_NSS = 0x03; //Select P03 is the NSS channel of SPI
P00CFG = 0x03; //P00 is multiplexed as SCLK function
P01CFG = 0x03; //P01 is multiplexed as MOSI function
P02CFG = 0x03; //P02 is multiplexed as MISO function
P03CFG = 0x03; //P03 is multiplexed It isfor NSS function
    
```

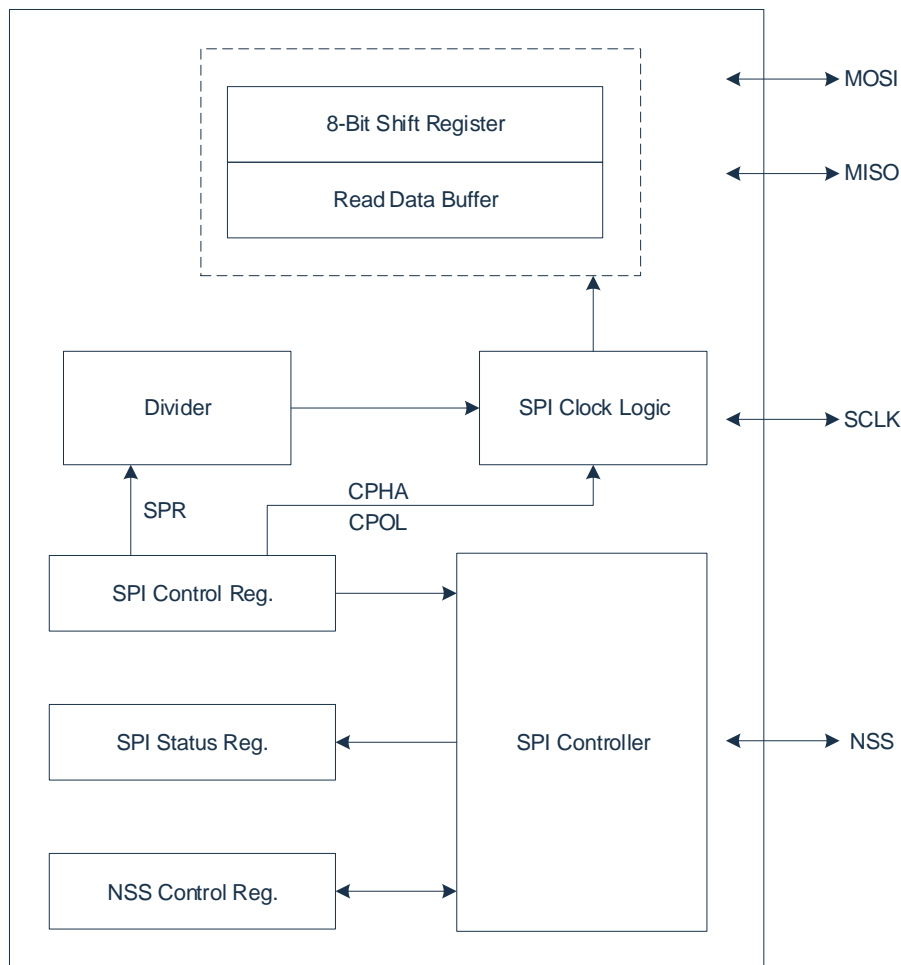
Configured as SCLK, MOSI, MISO and NSS ports, the pull-up resistor and open-drain output switch are forcibly closed. The schematic diagram of the multi-machine SPI communication structure is shown in the figure below:



## 18.3 SPI hardware description

When SPI transmission occurs, when one data pin is shifted out of an 8-bit character, another data pin is shifted into other 8-bit characters. The 8-bit shift register in the master device and the other 8-bit shift register in the slave device are connected as a circular 16-bit shift register. When a transfer occurs, the distributed shift register is shifted by 8 bits, which is effective. The characters of the master and slave are exchanged.

The central element in the SPI system is a module containing a shift register and a buffer for reading data. The system has a single buffer in the sending direction and a double buffer in the receiving direction. This means that new data cannot be written to the shifter until the previous data is transmitted; however, the received data is transferred to the parallel read data buffer, so the shifter can freely receive the second serial character. As long as the first character is read from the read data buffer before the next serial character is ready for transmission, there will be no overwriting. The SPI control block diagram is shown in the figure below:



The pins associated with SPI are: NSS, SCLK, MOSI, MISO.

The NSS output pin in the master mode is used to select the slave device, and the NSS input pin in the slave mode is used to enable transmission.

In master mode, the SCLK pin is used as the SPI clock signal reference. When the host device starts the 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 of the slave device, and SO is the output data line of the slave device.

When SPI is configured as a host device, MI pin is the input data line of the host device, and MO is the output data line of the host device.

## 18.4 SPI related registers

### 18.4.1 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, must be 0.
Bit6	SPEN:	SPI module enable bit; 1= enable; 0= disable.
Bit5	SPR2:	Bit [2] of SPI clock frequency selection bit.
Bit4	MSTR:	SPI mode selection bit; 1= active mode; 0= slave mode.
Bit3	CPOL:	SPI clock polarity selection bit; 1= High when SCLK is idle; 0= Low when SCLK is idle.
Bit2	CPHA:	SPI clock phase selection bit.
Bit1~Bit0	SPR<1:0>:	SPI clock frequency selection bits[1:0] (For frequency control, see the table below)

SPR2-SPR0 controls the SPI clock divider

SPR2	SPR1	SPR0	system clock divider
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

### 18.4.2 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<7:0>:	SPI data sent or received . Write operation: write the data to be sent (the sending order is from high to low). Read operation:that datahas been received.
-----------	---------------	--

### 18.4.3 SPI slave device selection control register SSCR

The slave device selection control register SSCR can be read or written at any time. It is used to configure which slave device selection output should be driven when confirming the SPI master transmission. When the SPI master transfer starts, the contents of the SSCR register will be automatically assigned to the NSS pin.

0xEF	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
SSCR	--	--	--	--	NSSO3	NSSO2	NSSO1	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~Bit4                                    -- Reserved, all must be 1.  
 Bit3~Bit0                                NSSOx: SPI slave device selection control bit (Master chip select output NSS is NSSOx, x=0-3).  
     0 = NSSOx outputs 0 when the SPI master transfer starts.  
     1 = NSSOx outputs 1 when the SPI master transfer starts.

### 18.4.4 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 transmission completed interrupt flag bit, read-only;  
     1 = SPI transmission completed (first read SPSR, then read/write SPDR and clear it);  
     0 = SPI transmission is not completed.  
 Bit6                                WCOL: SPI write conflict interrupt flag bit, read only;  
     1= write SPDR operation conflict occurs when SPI transmission is not completed (first read SPSR, then read/write SPDR and clear it to zero);  
     0= no write conflict.  
 Bit5 ~ Bit1                        - reservations required both 0.  
 Bit0                                SSCEN: SPI master mode NSS output control bit.  
     1= When SPI is in idle state, NSS outputs high level;  
     0= NSS outputs the content of register SSCR.

The SPI status register (SPSR) contains a flag indicating the completion of the transfer or the occurrence of a system error. When the corresponding event occurs and is cleared by software in sequence, all the flags will be automatically set. By reading SPSR and then accessing SPDR, SPISIF and WCOL will be cleared automatically.

The SSCEN bit is the enable bit for automatic slave selection output. When SSCEN is set to 1, the NSS line outputs the contents of the SSCR register when the transmission is in progress, and the NSS is high when the transmission is idle. When the SSCEN bit is cleared, the NSS line always displays the contents of the SSCR register.

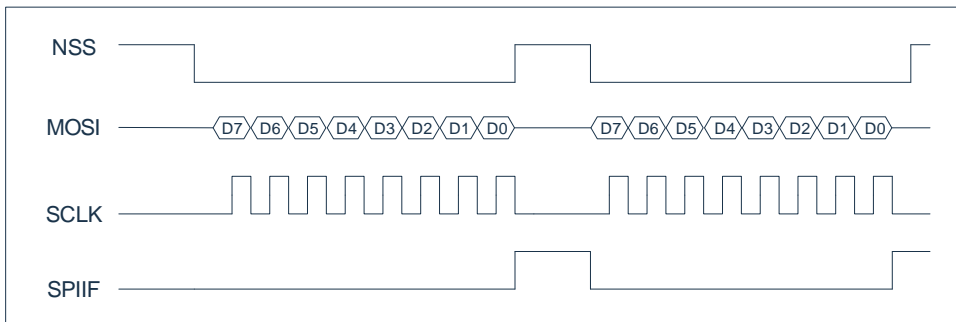


## 18.5 SPI master mode

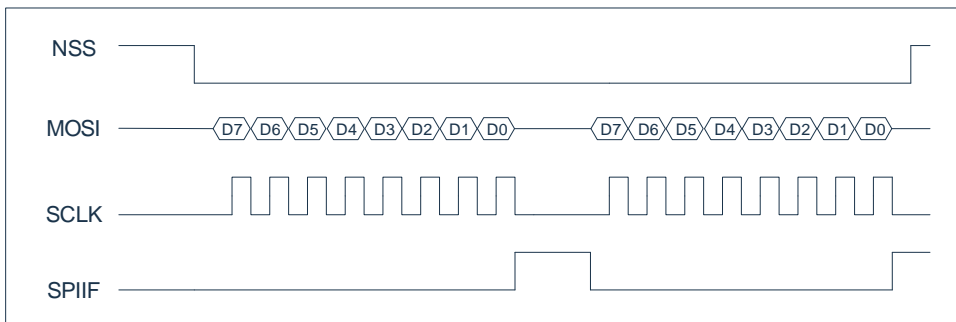
When SPI is configured as master mode, transfer is started by writing to the SPDR register. When a new byte is written to the SPDR register, the SPI starts to transfer. The serial clock SCLK is generated by SPI. In the master mode, SPI is enabled and SCLK is output.

SPI in master mode can select SPI slave device through NSS line. NSS line-The slave select output line is loaded with the contents of the SSCR register. The SSCEN bit of the SPSR register selects between automatic NSS line control and software control. Set SSCEN in the host mode. When SSCEN is set to 1, the NSS line outputs the contents of the SSCR register when the transfer is in progress, and NSS is high when the transfer is idle. When the SSCEN bit is cleared, 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, configure the SPI clock polarity CPOL=0, clock phase CPHA=0, and use the slave selection line as shown in the following figure:



When SSCEN=0, configure the SPI clock polarity CPOL=0, clock phase CPHA=0, the slave selection line is used as shown in the figure below:



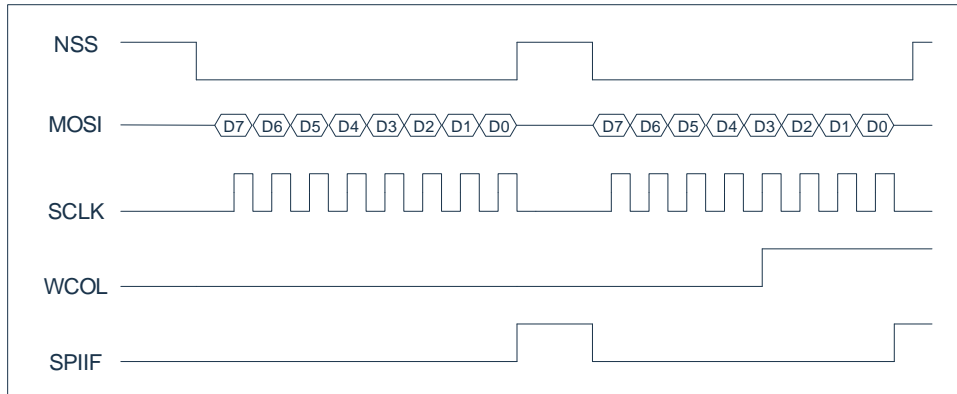
### 18.5.1 Write conflict error

If the SPI data register is written during transmission, a write conflict will occur. Transmission continues uninterrupted, and the write data that caused the error will not be written to the shifter. Write conflicts are indicated by the WCOL flag in the SPSR register.

When a WCOL error occurs, the WCOL flag is automatically set 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 the SPI master control mode, the write conflict error when the SPI clock polarity CPOL=0 and the clock phase CPHA=0 are configured as shown in the figure below:



The specific conditions for the write conflict are: during data transmission, when NSS is low, the first From the time when a data is sent to the eighth falling edge of SCLK, if SPDR is written during this period, a write conflict will occur and WCOL will be set to 1.

Note: When starting to send data, after writing SPDR, NSS does not change to low level immediately, it needs to wait at most one SPI clock before starting to be low. After NSS is low, it needs to wait for a system clock to start sending the first data, and then it enters the real data transmission state. During the period from writing SPDR to entering the real data transmission state, writing to SPDR again does not produce a write conflict. But this operation will update the data to be sent. If there are multiple write operations to SPDR, the data sent will be the last value written to SPDR.

Since SPI has only one transmit buffer, it is recommended to determine whether the last data has been sent before writing SPDR, and write the SPDR register after confirming that the transmission is completed to prevent write conflicts.

## 18.6 SPI slave mode

When configured as an SPI slave device, SPI transmission is initiated by the external SPI master module by using the SPI slave selection input and generates the SCLK serial clock.

Before the transmission starts, it is necessary to determine which SPI slave will be used to exchange data. NSS is used (clear = 0), the clock signal connected to the SCLK line will cause the SPI slave device to transfer the contents of the receive shift register of the MOSI line, and drive the MISO line with the contents of the transmitter shift register. When all 8 bits are shifted in/out, the SPI generates 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 transmission error-write conflict error.

### 18.6.1 Addressed error

In slave mode, only write conflict errors can be detected by SPI.

When the SPDR register write operation is performed while the SPI transfer is in progress, a write conflict error will occur.

In slave mode, when CPHA is cleared, as long as the NSS slave select line is driven low, even if all bits have been transmitted, a write conflict error may occur. This is because the start of the transfer is not explicitly specified, and the NSS being driven low after the full byte transfer may indicate the start of the next byte transfer.

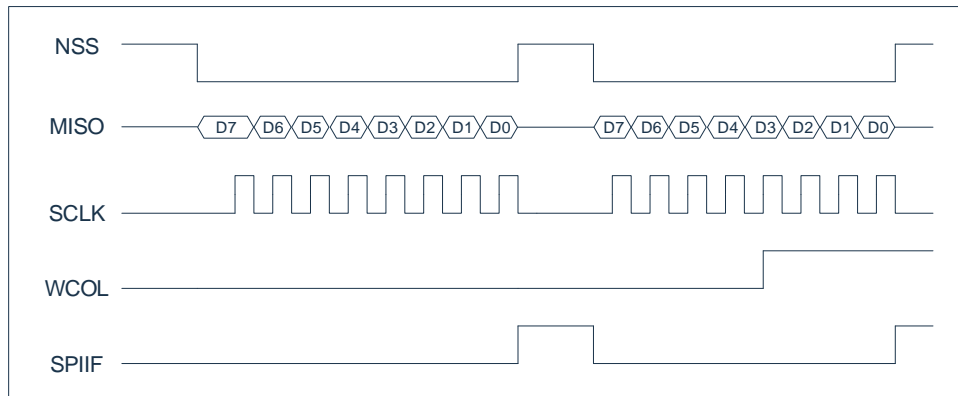
### 18.6.2 Write conflict error

If you write to the SPI data register during transmission, a write conflict will occur. Transmission continues uninterrupted, and the write data that caused the error will not be written to the shifter. Write conflicts are indicated by the WCOL flag in the SPSR register.

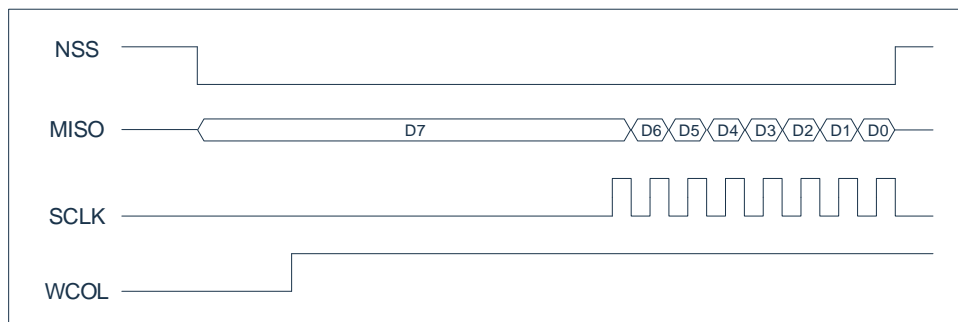
When a WCOL error occurs, the WCOL flag is automatically set by hardware. To clear the WCOL bit, the user should perform the following sequence:

- read the contents of the SPSR register;
- access the SPDR register (read or write).

The write conflict error during transmission in SPI slave mode is shown in the following figure:



In case CPHA is cleared, WCOL generation can also be caused by writing to the SPDR register when any NSS line is cleared. At this time, the SPI master does not generate the serial clock SCLK. can be completed. This is because the start of the transfer is not explicitly specified, and the NSS being driven low after the full byte transfer may indicate the start of the next byte transfer. When the NSS transmission line is low and the clock phase CPHA = 0, writing SPDR causes a write conflict error as shown in the following figure:



In addition, after writing SPDR in the slave mode, the NSS controlled by the host does not immediately become low. When NSS is low, you need to wait for the second edge of SCLK to start before entering the real data transmission state.

During the period from writing SPDR to the beginning of sending the first data, writing to SPDR again will not cause a write conflict. But this operation will update the data to be sent. If there are multiple write operations to SPDR, the data sent will be the last value written to SPDR.

During the period when the first data is sent to the second edge of SCLK, writing SPDR again will not cause a write conflict, nor will it update the data being sent. That is, the operation of writing SPDR this time is ignored.

Since SPI has only one transmit buffer, it is recommended to determine whether the last data has been sent before writing SPDR, and write the SPDR register after confirming that the transmission is completed to prevent write conflicts.

## 18.7 SPI clock control logic

### 18.7.1 SPI clock phase and polarity control

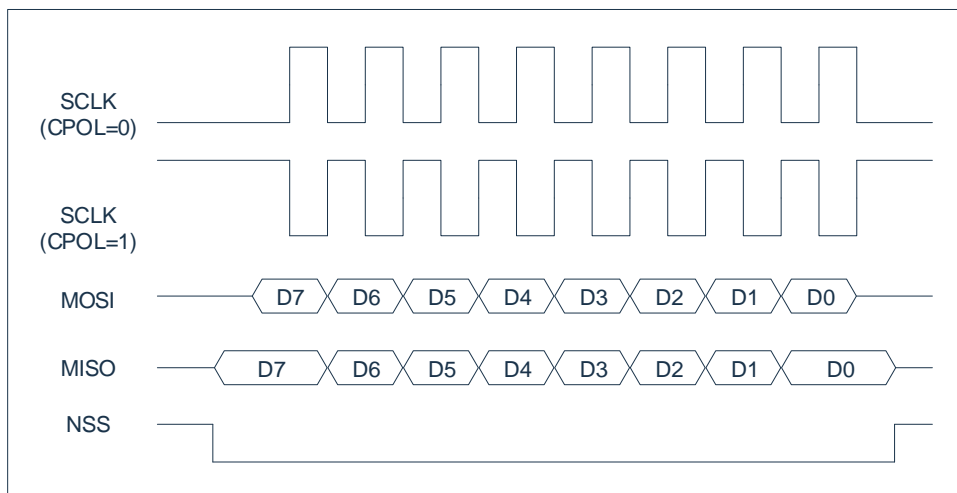
The software can select any of the four combinations using two control bits (phase and polarity of the serial clock SCLK) in the SPI control register (SPCR). The clock polarity is specified by the CPOL control bit. When the transmission is idle, the CPOL control bit selecting high or low level has no significant effect on the transmission format. The clock phase (CPHA) control bit selects one of two fundamentally different transmission formats. The clock phase and polarity of the host SPI device and the communication slave device should be the same. In some cases, the phase and polarity are changed during transmission to allow the host device to communicate with peripheral slaves with different requirements. The flexibility of the SPI system allows direct connection with almost all existing synchronous serial peripherals.

### 18.7.2 SPI transmission format

During SPI transmission, data is simultaneously sent (serial shift out) and received (serial shift in). The serial clock line is synchronized with the shift and sampling of the two serial data lines. The slave select line allows individual selection of slave SPI devices; unselected slave devices will not interfere with SPI bus activity. On the SPI master device, the slave select line can be selectively used to indicate multi-master bus contention.

### 18.7.3 CPHA=0 Transmission Format

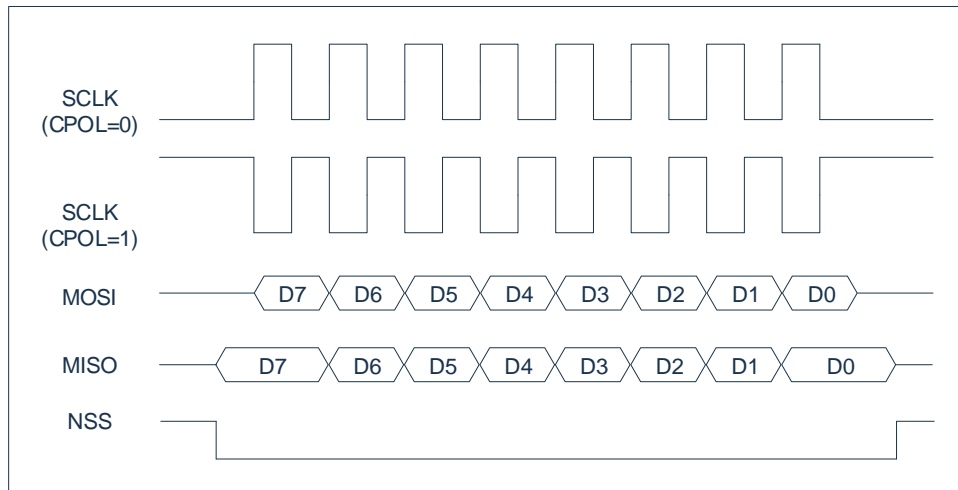
The following figure shows the timing diagram of SPI transmission with. SCLK shows two waveforms: one for CPOL equal to 0 and the other for CPOL equal to 1. The figure can be described as a master device or slave device timing diagram through SCLK. The master input/slave output (MISO) and master output/slave input (MOSI) pins are directly connected between the host and the slave. The MISO signal is the slave output, and the MOSI signal is the master output. The NSS line is the slave selection input of the slave; the NSS pin of the master is not shown, but it is assumed to be invalid. The NSS pin of the host must be high. This sequence diagram functionally describes how to transmit; it should not be used as a substitute for data sheet parameter information.



When CPHA=0, the NSS line must be de-set and reset between each successive serial byte. In addition, if the slave writes data to the SPI data register (SPDR) when NSS is at low level, a write conflict error will occur. When CPHA = 1, the NSS line may remain low between consecutive transmissions (it can always remain low). In a system with a single fixed master and a single slave driving the MISO data line, this format is sometimes preferred.

### 18.7.4 CPHA=1 transmission format

The following figure is the timing diagram of SPI transmission with CPHA=1. SCLK displays 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 the slave, this diagram can be interpreted as a 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 selection input of the slave; the NSS pin of the master is not shown, but it is assumed to be invalid. The NSS pin of the host must be high or must be reconfigured as a general-purpose output that does not affect the SPI.



## 18.8 SPI data transmission

### 18.8.1 SPI transmission start

All SPI transmissions are started and controlled by the master SPI device. As a slave device, SPI will consider the transmission start at the first SCLK edge or the falling edge of NSS according to the selected CPHA format. When CPHA = 0, the falling edge of NSS indicates the beginning of the transfer. When CPHA = 1, the first edge on SCLK indicates the beginning of the transfer. No matter which CPHA mode, the transmission can be aborted by making the NSS line high, but it will reset the SPI slave logic and counter. The selected SCLK rate has no effect on the operation of the slave, because the master's clock is controlling the transfer.

When the SPI is configured as a host, the transfer is initiated by software written to SPDR.

### 18.8.2 SPI transmission ends

When the SPIF flag is set to 1, the SPI transmission is technically completed, but depending on the configuration of the SPI system, there may be other tasks. Since the SPI bit rate does not affect the time of the end period, only the fastest rate is considered in the discussion of the end period. When SPI is configured as a master, SPIF is set at the end of the eighth SCLK cycle. When CPHA is equal to 1, SCLK is inactive during the last half of the eighth SCLK cycle.

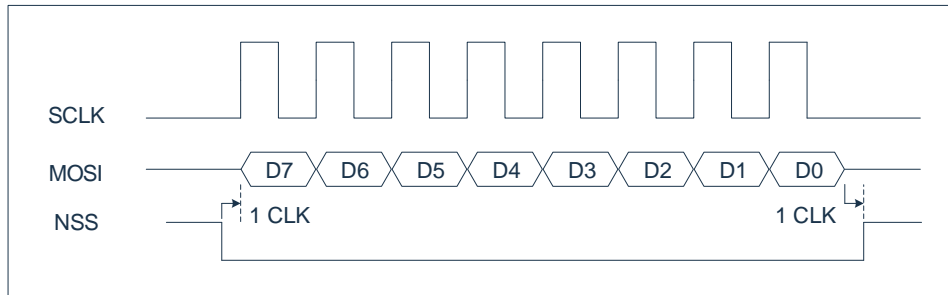
Because the SCLK line can be asynchronous with the MCU clock of the slave, and the slave cannot access as much information as the master can access the SCLK cycle, so when the SPI runs as a slave, the end cycle is different. For example, when CPHA = 1, the last SCLK edge occurs in the middle of the eighth SCLK cycle, and the slave cannot know when the last SCLK cycle ends. For these reasons, the slave considers that the transmission is complete after the last bit of the serial data is sampled, which corresponds to the middle of the eighth SCLK cycle.

The SPIF flag is set at the end of the transmission, but when the NSS line is still low, the slave is not allowed to write new data into the SPDR.

## 18.9 SPI timing diagram

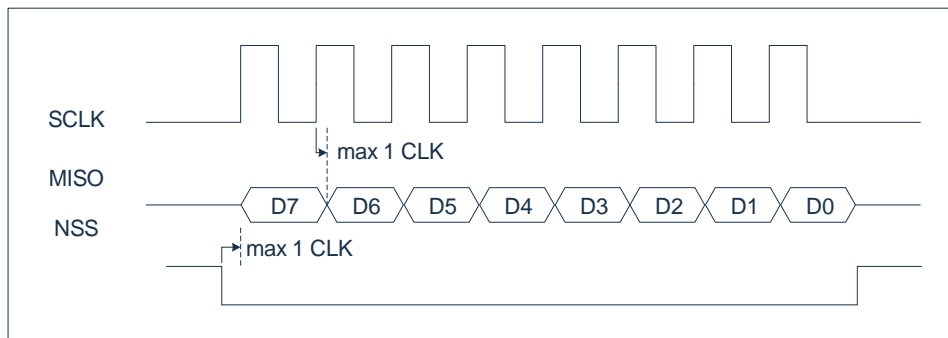
### 18.9.1 Master mode transmission

When the SPI clock polarity CPOL=0, clock phase CPHA=1, in the SPI master mode, after NSS is low, a system clock CLK, MOSI starts to output, and MOSI data is clocked in SCLK. The rising edge of the output. The timing diagram of master mode is shown in the following figure:



### 18.9.2 Slave mode transmission

When the SPI clock polarity CPOL=0 and clock phase CPHA=1, the data on MISO starts to output after the falling edge of the NSS line. The maximum difference between the MISO data output and the falling edge of NSS is 1 system clock CLK. The timing diagram of the slave mode is shown in the following figure:





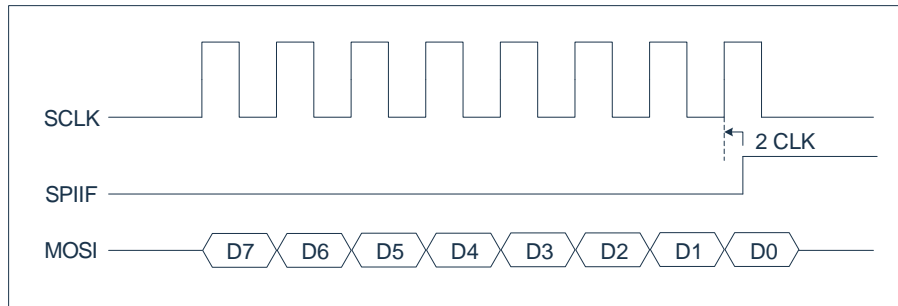
## 18.10 SPI Interrupt

The interrupt number of SPI is 22, and its interrupt vector is 0x00B3. To enable the SPI interrupt, the enable bit SPIIE must be set to 1, and the overall interrupt enable bit EA must be set to 1.

If the SPI related interrupt enable is turned on and the SPI general interrupt indicator bit SPIIF=1, the CPU will enter the interrupt service routine. The SPIIF operation attribute is read-only and has nothing to do with the state of SPIIE.

After any one of the transmission completion flag SPISIF and write conflict WCOL in the SPI status register SPSR is set to 1, the SPI total interrupt indicator bit SPIIF will be set to 1. Only when these 3 flag bits are all 0, SPIIF is automatically cleared to 0.

When the SPI clock polarity CPOL=0 and the clock phase CPHA=1, in the SPI master mode, SPIIF will generate 2 system clocks CLK after the 8th SCLK rising edge of each frame of data. The timing diagram is shown in the figure below :



### 18.10.1 Interrupt mask register EIE2

0xAA	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIE2	SPIIE	I2CIE	WDTIE	ADCIE	PWMIE	-	ET4	ET3
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 <sup>2</sup> C interrupt enable bit; 1= Enable I <sup>2</sup> C interrupt; 0= Disable I <sup>2</sup> C interrupt.
Bit5	WDTIE: WDT interrupt enable bit; 1= Enable WDT overflow interrupt; 0= Disable WDT overflow interrupt.
Bit4	ADCIE: ADC interrupt enable bit; 1= enable ADC interrupt; 0= disable ADC interrupt.
Bit3	PWMIE: PWM total interrupt enable bit; 1= allow all PWM interrupts; 0= disable all PWM interrupts.
Bit2	- Reserved, must be zero.
Bit1	ET4: Timer4 interrupt enable bit; 1= Enable Timer4 interrupt; 0= Disable Timer4 interrupt.
Bit0	ET3: Timer3 interrupt enable bit; 1= Enable Timer3 interrupt; 0= Disable Timer3 interrupt.

### 18.10.2 Interrupt priority control register EIP2

0xB8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIP2	PSPi	PI2C	PWDT	PADC	PPWM	-	PT4	PT3
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      PSPi: SPI interrupt priority control bit;  
           1= set as high-level interrupt;  
           0= set as low-level interrupt.
- Bit6      PI2C: I<sup>2</sup>C interrupt priority control bit;  
           1= set as high-level interrupt;  
           0= set as low-level interrupt.
- Bit5      PWDT: WDT interrupt priority control bit;  
           1= set as high-level interrupt;  
           0= set as low-level interrupt.
- Bit4      PADC: ADC interrupt priority control bit;  
           1= set as high-level interrupt;  
           0= set as low-level interrupt.
- Bit3      PPWM: PWM interrupt priority control bit;  
           1= set as high-level interrupt;  
           0= set as low-level interrupt.
- Bit2      - Reserved, must be zero.
- Bit1      PT4: TIMER4 interrupt priority control bit;  
           1= set as high-level interrupt;  
           0= set as low-level interrupt.
- Bit0      PT3: TIMER3 interrupt priority control bit;  
           1= set as high-level interrupt;  
           0= set as low-level interrupt.

### 18.10.3 Peripheral interrupt flag register EIF2

0xB2	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIF2	SPIIF	I2CIF	-	ADCIF	PWMIF	--	TF4	TF3
R/W	R	R	-	R/W	R	--	R/W	R/W
Reset value	0	0	0	0	0	0	0	0

- Bit7      SPIIF: SPI general interrupt indicator bit, read-only;  
           1 = SPI generates an interrupt (this bit is automatically cleared after clearing the specific interrupt flag bit);  
           0 = SPI does not generate an interrupt.
- Bit6      I2CIF: I<sup>2</sup>C general interrupt indicator bit, read-only;  
           1 = I<sup>2</sup>C generates an interrupt (after clearing the specific interrupt flag bit, this bit is automatically cleared);  
           0 = I<sup>2</sup>C does not generate an interrupt.
- Bit5      - Reserved, must be zero.
- Bit4      ADCIF: ADC interrupt flag bit;  
           1 = ADC conversion is completed and needs to be cleared by software;  
           0 = ADC conversion is not completed.
- Bit3      PWMIF: PWM general interrupt indicator bit, read-only;  
           1 = PWM generates an interrupt, (this bit is automatically cleared after clearing the specific interrupt flag bit);  
           0 = PWM does not generate an interrupt.
- Bit2      - Reserved, must be zero.
- Bit1      TF4: Timer4 timer overflow interrupt flag bit;  
           1 = Timer4 timer overflows, it is automatically cleared by hardware when entering the interrupt service routine, or can be cleared by software;  
           0 = Timer4 timer has no overflow.
- Bit0      TF3: Timer3 timer overflow interrupt flag bit;  
           1 = Timer3 timer overflows, it is automatically cleared by hardware when entering the interrupt service routine, or it can be cleared by software;  
           0 = Timer3 timer has no overflow.

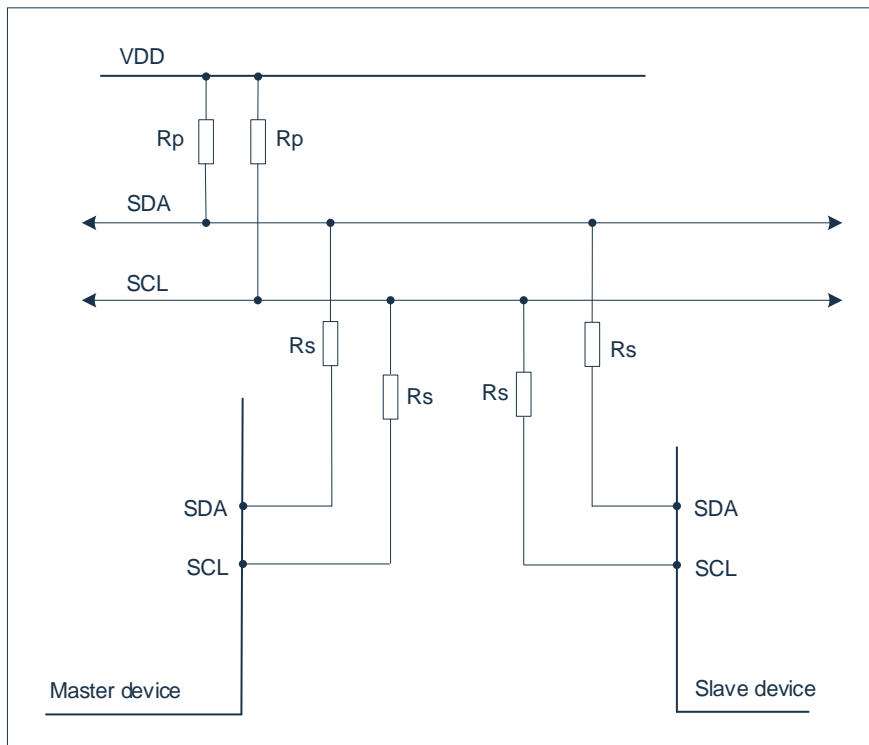
## 19. I<sup>2</sup>C module

### 19.1 overview

This module provides microcontroller and the I<sup>2</sup>C interface between the C bus. The connection diagram is shown in the figure below, and supports arbitration and clock synchronization to allow operation in a multi-master system. I<sup>2</sup>C supports normal and fast modes.

The I<sup>2</sup>C characteristics of the C module are as follows: It

- ◆ supports 4 working modes: master sending, master receiving, slave sending, and slave receiving.
- ◆ Support 2 transmission speed modes:
  - standard (up to 100Kb/s);
  - fast (up to 400Kb/s);
- ◆ perform arbitration and clock synchronization.
- ◆ Support multi-host system.
- ◆ The host mode supports I<sup>2</sup>7-bit addressing mode and 10-bit addressing mode on the C bus (software support).
- ◆ The slave mode supports I<sup>2</sup>7-bit addressing mode on the C bus.
- ◆ An interrupt is generated.
- ◆ Allows operation in a wide range of clock frequencies (built-in 8-bit timer).



## 19.2 I<sup>2</sup>C port configuration

If these the I<sup>2</sup>C function, the corresponding port should first be configured as SCL and SDA channels. For example, configure the P00 and P01 ports as I<sup>2</sup>C functions:

```
PS_SCL = 0x00; //Select port P00 as the SCL pin
PS_SDA = 0x01; //select port P01 as the SDA pin
P00CFG = 0x02; //P00 multiplexes the SCL function
P01CFG = 0x02; // P01 multiplexes the SDA function
```

After configuring I<sup>2</sup>C channel, this group of ports is in open drain state by default. You can configure whether to enable SCL, internal pull-up resistance of SDA port, or add pull-up resistance outside the chip through PxUP.

In the master control mode, IIC outputs SCL to the slave. After sending the address or data, the slave needs to pull the SCL down and send back the corresponding response signal to the host. The host needs to read back the SCL port line status to detect whether the slave releases the SCL to determine whether the next frame data transmission is required. If the pull-up resistance or board-level parasitic capacitance of SCL is larger, the reading back time will be longer, which will affect the communication speed of IIC. Please refer to IIC application manual for details.

## 19.3 I<sup>2</sup>C master control mode

There are six registers for connecting with the master: control, status, slave address, send data, receive data and timer period register.

Register		address
write	read	
slave address register I2CMSA	slave address register I2CMSA	0xF4
master mode control register I2CMCR	master mode status register I2CMSR	0xF5
master control sending data register I2CMBUF	master receiving data register I2CMBUF	0xF6
timing cycle register I2CMTP	timing cycle register I2CMTP	0xF7

Master mode control register I2CMCR and the master mode status register I2CMSR share a register address, but they are physically two different registers.

The main control sending data register and the main control receiving data register share a register address. The write operation accesses the sending register I2CMBUF, and the read operation accesses the receiving register I2CMBUF.

During write operation, it is used as a control register to write, and read operation is used as a status register to read.

### 19.3.1 I<sup>2</sup>C master mode timing period register

In order to generate a wide range of SCL frequency in the, the module has a built-in 8-bit timer. Used for standard and fast transmission.

When  $TIMER\_PRD \neq 0$ , the ideal clock cycle of SCL:  $2 * (1 + TIMER\_PRD) * 10 * T_{sys}$

$TIMER\_PRD = 0$ , the ideal clock cycle of SCL:  $3 * 10 * T_{sys}$

Refer to IIC Application Manual for the specific calculation formula of SCL.

0xF7	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
I2CMTP	--	MTP6	MTP5	MTP4	MTP3	MTP2	MTP1	MTP0
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	1

Bit7 -- Reserved, must be 0.  
 Bit6~Bit0 MTP<6:0>: Bit 6-0 of the period timer register of standard and fast mode:  $TIMER\_PRD[6:0]$ .

### 19.3.2 I<sup>2</sup>C master mode control and status register

Control register includes 4 bits: RUN, START, STOP, and ACK bits. The START bit will generate a START or REPEATED START condition. The STOP bit determines whether the data transmission stops at the end of the cycle or continues. In order to generate a single transmission cycle, the slave address register writes the required address, the R/S bit is set to 0, and the control register writes ACK=x, STOP=1, START=1, RUN=1 (I2CMCR=xxx0\_x111x) to perform operations and stop. When the operation is completed (or an error occurs), an interrupt is generated. Data can be read from the receive data register.

When I<sup>2</sup>C works in master mode, the ACK bit must be set to 1. This will cause the I<sup>2</sup>C bus controller to automatically send an answer after each byte. When the I<sup>2</sup>C bus controller no longer needs the slave to send data, this bit must be cleared to 0.

Master control mode control register

0xF5	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
I2CMCR	RSTS	-	-	-	ACK	STOP	START	RUN
R/W	W	R	R	W	W	W	W	W
Reset value	0	0	1	0	0	0	0	0

Bit7	RSTS: I <sup>2</sup> C active module reset control bit; 1= Reset the main control module (the entire I <sup>2</sup> C register of, including I2CMSR); 0= I <sup>2</sup> C clear the interrupt flag bit in main control mode.
Bit6 ~ Bit5	- Reserved.
Bit4	- Reserved, must be zero.
Bit3	ACK: acknowledge enable bit; 1= enable; 0= disable.
Bit2	STOP: Stop enable bit; 1= enable; 0= disable.
Bit1	START: Start enable bit; 1= enable; 0= disable.
Bit0	RUN: Run enable bit; 1= enable; 0= disable.

Various operations in the master control mode can be realized through the following control bit combination list:

START: Send start signal.

SEND: Send data or address.

RECEIVE: Receive data.

STOP: Send end signal.

## Combination of control bits (IDLE state)

R/S	ACK	STOP	START	RUN	OPERATION
0	-	0	1	1	START followed by SEND (the host stays in the sending mode)
0	-	1	1	1	START followed by SEND and STOP
1	0	0	1	1	Non-response is adopted for reception after START (the host remains in receiver mode)
1	0	1	1	1	START followed by RECEIVE and STOP
1	1	0	1	1	START followed by RECEIVE (main unit remains in receiver mode)
1	1	1	1	1	No combination
0	0	0	0	1	No combination

## Combination of control bits (master transmission state)

R/S	ACK	STOP	START	RUN	OPERATION
-	-	0	0	1	SEND operation
-	-	1	0	0	Stop
-	-	1	0	1	SEND followed by STOP
0	-	0	1	1	Repeat START followed by SEND
0	-	1	1	1	Repeat START, followed by SEND and STOP
1	0	0	1	1	Repeat the START condition followed by the response RECAIVE operation (The host stays in receiver mode)
1	0	1	1	1	Repeat START, followed by SEND and STOP conditions
1	1	0	1	1	Repeat START condition followed by RECEIVE (The host stays in receiver mode)
1	1	1	1	1	No combination

## Combination of control bits (master control receiving state)

R/S	ACK	STOP	START	RUN	OPERATION
-	0	0	0	1	RECEIVE operation with response (The host stays in receiver mode)
-	-	1	0	0	STOP
-	0	1	0	1	RECEIVE followed by STOP
-	1	0	0	1	RECEIVE operation (main unit remains in receiver mode)
-	1	1	0	1	No combination
1	0	0	1	1	Repeat the START, followed by the response RECEIVE operation (The host stays in receiver mode)
1	0	1	1	1	Repeat START, followed by RECEIVE and STOP
1	1	0	1	1	Repeat START followed by RECEIVE (The host stays in receiver mode)
0	-	0	1	1	Repeat START followed by SEND (The host stays in transmitter mode)
0	-	1	1	1	Repeat START, followed by SEND and STOP

## master mode status register I2CMSR

0xF5	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
I2CMSR	I2CMIF	BUS_BUSY	IDLE	ARB_LOST	DATA_ACK	ADD_ACK	ERROR	BUSY
R/W	R	R	R	R	R	R	R	R
Reset value	0	0	1	0	0	0	0	0

Bit7	I2CMIF:	I <sup>2</sup> C interrupt flag bit in master control mode; 1=In master control mode, transmission/reception is completed, or a transmission error occurs. (Cleared by software, cleared by writing 0); 0= no interrupt is generated.
Bit6	BUS_BUSY:	I master control mode/slave control mode <sup>2</sup> C bus busy flag bit in; 1 = I <sup>2</sup> C bus is busy and cannot be transmitted (the start position on the bus is 1, the stop condition is cleared). 0= -
Bit5	IDLE:	I <sup>2</sup> C master control mode idle flag bit; 1= idle state; 0= working state.
Bit4	ARB_LOST:	I <sup>2</sup> C master mode arbitration flag; 1= lost bus control. 0= -
Bit3	DATA_ACK:	I <sup>2</sup> C master mode sending data acknowledgement flag bit; 1=the last time there is no response to the data sent. 0= -
Bit2	ADD_ACK:	I <sup>2</sup> C master mode addressing acknowledgement flag bit; 1= There is no acknowledgement in the last addressing. 0= -
Bit1	ERROR:	I <sup>2</sup> C master mode error flag; 1= No response from the addressed slave/No response to data sent/I <sup>2</sup> C bus arbitration conflict. 0= -
Bit0	BUSY:	I <sup>2</sup> C main control module busy flag bit; 1= I <sup>2</sup> C module is transmitting data. 0= -



### 19.3.3 I<sup>2</sup>C slave address register

The slave address register is composed of 8 bits: 7-bit address bits (A6-A0) and receive/transmit bits R/S. The R/S bit determines whether the next operation is to receive (1) or send (0).

Master mode slave address register I2CMSA

0xF4	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
I2CMSA	SA6	SA5	SA4	SA3	SA2	SA1	SA0	R/S
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      SA<6:0>: I<sup>2</sup>C Slave address inC master mode.  
 Bit0            R/S: The I<sup>2</sup>C receiving/sending status selection bit after sending the slave address inC master control mode;  
                   1= receive data after correct addressing;  
                   0= send data after correct addressing.

### 19.3.4 I<sup>2</sup>C master mode sending and receiving data register

The sending data register consists of eight data bits, which will be sent on the bus during the next sending or burst sending operation. The first sending bit is MD7 (MSB).

Master mode data buffer register I2CMBUF

0xF6	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
I2CMBUF	MD7	MD6	MD5	MD4	MD3	MD2	MD1	MD0
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      MD<7:0>: I<sup>2</sup>C Send/receive data in master control mode.

## 19.4 I<sup>2</sup>C slave mode

There are five registers for connecting to the target device: own address, control, status, send data and receive data registers.

Register		address
write	read	
self address register I2CSADR	self address register I2CSADR	0xF1
control register I2CSCR	status register I2CSSR	0xF2
send data I2CSBUF	receive data I2CSBUF	0xF3

### 19.4.1 I<sup>2</sup>C self address register I2CSADR

The self address register consists of seven address bits that identify the I2C core on the I2C bus. This register can read and write addresses.

Self address register I2CSADR

0xF1	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
I2CSADR	--	SA6	SA5	SA4	SA3	SA2	SA1	SA0
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, must be 0.  
 Bit6~Bit0              SA<6:0>:    I<sup>2</sup>C self address of slave mode.

### 19.4.2 I<sup>2</sup>C slave mode control and status register I2CSCR/I2CSSR

Slave mode control register and slave mode status register occupy a register address, use different operations to distinguish access to these two registers:

write operation: write I2CSCR (only Write)

Read operation: Read the I2CSSR (read-only)

control register consists of two bits: RSTS and DA bits. The RSTS bit controls the entire I<sup>2</sup>C slave module. When the I<sup>2</sup>C bus encounters some problems, the software can enable this bit to reinitialize I2CS. The DA bit enables and disables I2CS device operation. Reading this address places the status register on the data bus.

Slave mode control register I2CSCR

0xF2	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
I2CSCR	RSTS	-	-	-	-	-	-	DA
R/W	W	R	R	R	R	R	R	W
Reset value	0	0	0	0	0	0	0	0

Bit7                      RSTS:    I<sup>2</sup>C slave module reset control bit;  
                                  1=    reset slave module;  
                                  0=    no effect.  
 Bit6 ~ Bit1              -    reservations required both 0.  
 Bit0                      DA:    I<sup>2</sup>C slave mode enable bit;  
                                  1=    enable;  
                                  0=    disable.

The status register consists of three bits: SENDFIN bit, RREQ bit, and TREQ bit. The sent SENDFIN bit indicates that the host I<sup>2</sup>C controller has completed the data reception during the I2CS single or continuous sending operation. The RREQ bit of the receive request indicates that the I2CS device has from the I<sup>2</sup>C received a data byte from the host, and the I2CS device should read a data byte from the receive data register I2CSBUF. The transmit request TREQ bit indicates that the I2CS device is addressed

as a slave transmitter, and the I2CS device should write a data byte into the transmit data register I2CSBUF. If the I<sup>2</sup>C interrupt enable is turned on, setting any of the three flag bits will generate an interrupt.

In the slave mode, the bus busy flag bit is judged by Bit6 (BUS\_BUSY) of the master mode status register I2CMSR. I2CMSR is 0x20 when the bus is idle, and the I2CMSR register is 0x60 when the start condition is generated until the stop condition is generated. When the stop condition is generated, I2CMSR is 0x20.

Slave mode status register I2CSSR

0xF2	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
I2CSSR	--	--	--	--	--	SENDFIN	TREQ	RREQ
R/W	--	--	--	--	--	R	R	R
Reset value	0	0	0	0	0	0	0	0

Bit7~Bit3 -- Reserved, all must be 0.

Bit2 SENDFIN: The I<sup>2</sup>C send operation completed flag bit in slave mode, read-only.

1= The main control device no longer needs data, TREQ is no longer set to 1, and this data transfer has been completed. (It is automatically cleared after reading I2CSCR).

0= -

Bit1 TREQ: I<sup>2</sup>C slave mode ready to send flag bit, read only.

1= As the sending device has been addressed or the master device is ready to receive data. (It is automatically cleared after writing I2CSBUF).

0= -

Bit0 RREQ: I<sup>2</sup>C slave mode receiving completion flag bit, read only.

1= Receive completed. (It is automatically cleared after reading I2CSBUF).

0= Receive not completed.

### 19.4.3 I<sup>2</sup>C slave mode to send and receive buffer register

0xF3	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
I2CSBUF	I2CSBUF7	I2CSBUF6	I2CSBUF5	I2CSBUF4	I2CSBUF3	I2CSBUF2	I2CSBUF1	I2CSBUF0
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 I2CSBUF<7:0>: I<sup>2</sup>C sent or received data;

write operation: write the data to be sent (the sending order is from the higher bit to the low bit);

read operation: Data has been received.

## 19.5 I<sup>2</sup>C interrupt

The interrupt number of I2C is 21, and its interrupt vector is 0x00AB. To enable I2C interrupts, the enable bit I2CIE must be set to 1, and the overall interrupt enable bit EA must be set to 1.

If the I2C related interrupt enable is turned on, and the I2C total interrupt indicator bit I2CIF=1, the CPU will enter the interrupt service routine. The I2CIF operation attribute is read-only and has nothing to do with the status of I2CIE.

I2C master mode interrupt flag bit I2CMIF, send operation completed flag bit SENDFIN in slave mode, slave mode ready to send flag bit TREQ, slave mode receive completion flag bit RREQ when any one is 1, the I2C total interrupt indicator bit I2CIF will be set to 1. Only when these 4 flag bits are all 0, I2CIF is automatically cleared to 0.

### 19.5.1 Interrupt mask register EIE2

0xAA	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIE2	SPIIE	I2CIE	WDTIE	ADCIE	PWMIE	-	ET4	ET3
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 <sup>2</sup> C interrupt enable bit; 1= Enable I <sup>2</sup> C interrupt; 0= Disable I <sup>2</sup> C interrupt.
Bit5	WDTIE: WDT interrupt enable bit; 1= Enable WDT overflow interrupt; 0= Disable WDT overflow interrupt.
Bit4	ADCIE: ADC interrupt enable bit; 1= Enable ADC interrupt; 0= Disable ADC interrupt.
Bit3	PWMIE: PWM total interrupt enable bit; 1= Allow all PWM interrupts; 0= Disable all PWM interrupts.
Bit2	- Reserved, must be zero.
Bit1	ET4: Timer4 interrupt enable bit; 1= Enable Timer4 interrupt; 0= Disable Timer4 interrupt.
Bit0	ET3: Timer3 interrupt enable bit; 1= Enable Timer3 interrupt; 0= Disable Timer3 interrupt.

### 19.5.2 Interrupt priority control register EIP2

0xBA	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIP2	PSPI	PI2C	PWDT	PADC	PPWM	-	PT4	PT3
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	PSPI: SPI interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit6	PI2C: I <sup>2</sup> C interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit5	PWDT: WDT interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit4	PADC: ADC interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit3	PPWM: PWM interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit2	- Reserved, must be zero.
Bit1	PT4: TIMER4 interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit0	PT3: TIMER3 interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.

### 19.5.3 Peripheral interrupt flag register EIF2

0xB2	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIF2	SPIIF	I2CIF	-	ADCIF	PWMIF	--	TF4	TF3
R/W	R	R	-	R/W	R	--	R/W	R/W
Reset value	0	0	0	0	0	0	0	0

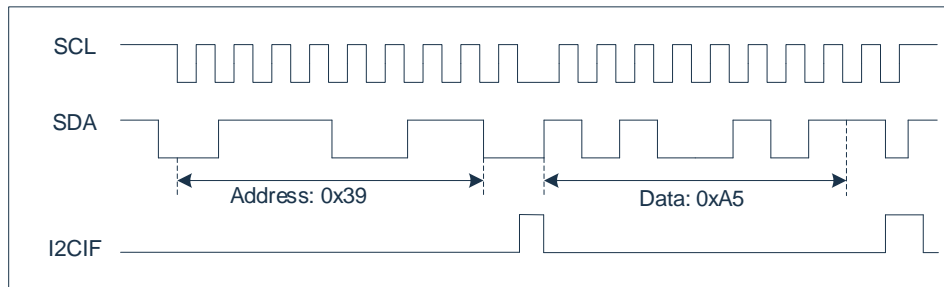
- Bit7      SPIIF: SPI general interrupt indicator bit, read-only;  
           1 = SPI generates an interrupt (this bit is automatically cleared after clearing the specific interrupt flag bit);  
           0 = SPI does not generate an interrupt.
- Bit6      I2CIF: I<sup>2</sup>C total interrupt indicator bit, read-only;  
           1 = I<sup>2</sup>C generates an interrupt (after clearing the specific interrupt flag bit, this bit is automatically cleared);  
           0 = I<sup>2</sup>C does not generate an interrupt.
- Bit5      - Reserved, must be zero.
- Bit4      ADCIF: ADC interrupt flag bit;  
           1 = ADC conversion is completed and needs to be cleared by software;  
           0 = ADC conversion is not completed.
- Bit3      PWMIF: PWM general interrupt indicator bit, read-only;  
           1 = PWM generates an interrupt (this bit is automatically cleared after clearing the specific interrupt flag bit);  
           0 = PWM does not generate an interrupt.
- Bit2      - Reserved, must be zero.
- Bit1      TF4: Timer4 timer overflow interrupt flag bit;  
           1 = Timer4 timer overflows, it is automatically cleared by hardware when entering the interrupt service routine, and can also be cleared by software;  
           0 = Timer4 timer has no overflow.
- Bit0      TF3: Timer3 timer overflow interrupt flag bit;  
           1 = Timer3 timer overflows, it is automatically cleared by hardware when entering the interrupt service routine, and can also be cleared by software;  
           0 = Timer3 timer has no overflow.



## 19.6.2 Single Transmission

The following figure shows the signal sequence sent by I2C during a single data period. Single transmission sequence:

- Starting conditions;
- I2C is addressed by the I2C host as a transmitter;
- The address is confirmed by I2C;
- Data is transmitted by I2C;
- The data is not confirmed by the I2C host;
- Stop condition.





### 19.6.3 Continuous reception

The following figure shows the signal sequence received by I2C during continuous data reception. Continuous receiving sequence:

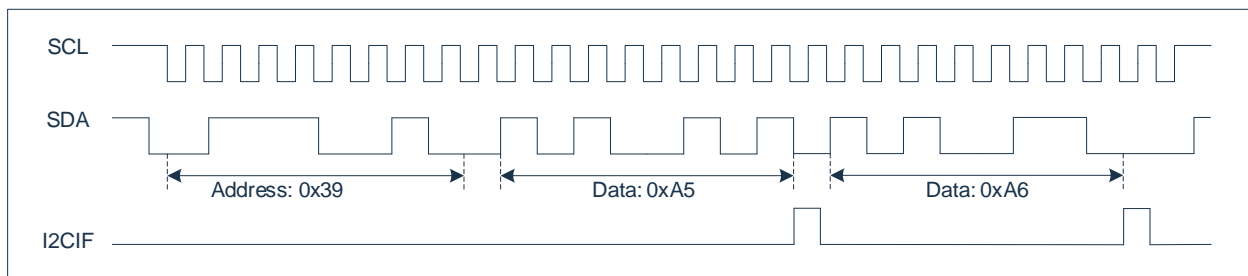
- Starting conditions.
- I2C is addressed by the I2C master as a receiver.
- The address is confirmed by I2C.

1) The data is received by I2C.

2) The data is confirmed by I2C.

- Stop condition.

Sequences 1) and 2) are repeated until the stop condition occurs.



### 19.6.4 Continuous transmission

The following figure shows the signal sequence sent by I2C during continuous data transmission. Continuous sending sequence:

- Sending conditions.
- I2C is addressed by the I2C master as a transmitter.
- The address is confirmed by I2C.

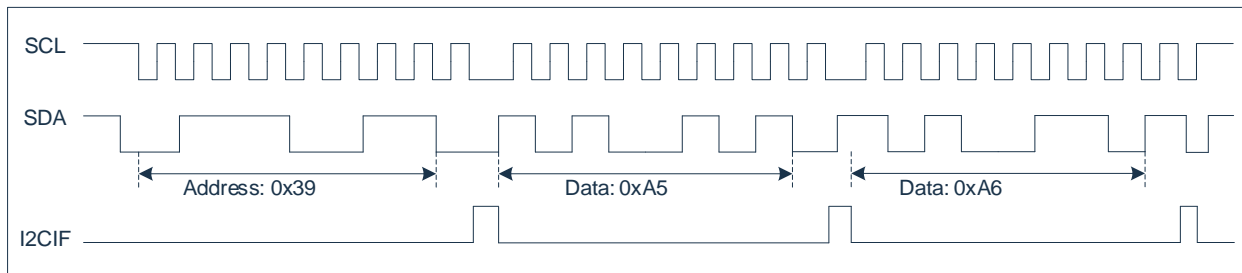
1) The data is sent by I2C.

2) The data I2C host confirms the data.

3) The final data is not confirmed by the I2C host.

- Stop condition.

Repeat sequence 1) and 2) until the last data sent is not confirmed by the I2C host 3)..



## 20. UARTn module

### 20.1 overview

The Universal Synchronous Asynchronous Receiver Transmitter (UART0 / UART1 ) provides a flexible method for full-duplex data exchange with external devices.

There are two physically independent receiving and sending buffers inside UARTn, SBUF<sub>n</sub>, which can be used to distinguish between receiving and sending buffers by reading and writing commands to SBUF<sub>n</sub>. When writing SBUF<sub>n</sub>, the data is loaded into the sending buffer; when reading SBUF<sub>n</sub>, the content in the receiving buffer is read.

UARTn has four operating modes: a synchronous mode and three asynchronous modes. Modes 2 and 3 have multi-machine communication function, which can be enabled by setting the SM<sub>n2</sub> bit in the SCON<sub>n</sub> register to 1. The host processor first sends the address byte that identifies the target slave. The address byte is different from the data byte, because the 9th bit in the address byte is 1 and the data byte is 0. When SM<sub>n2</sub>=1, the slave will not be interrupted by the data byte. The address byte will interrupt all slaves. The addressed slave will clear its SM<sub>n2</sub> bit, and is ready to receive the upcoming data byte. The unaddressed slave sets SM<sub>n2</sub> to 1 and ignores the incoming data.

### 20.2 UARTn port configuration

Before using the UARTn module, you need to configure the corresponding ports to the TXD<sub>n</sub> and RXD<sub>n</sub> channels of UARTn. For example, the port configuration of UART0 is as follows:

```
PS_RXD0 = 0x05; //Select P05 as RXD0 Pin
P05CFG = 0x02; //P05 is multiplexed as RXD0 function
P04CFG = 0x02; //P04 is multiplexed as TXD0 function
```

UARTn port RXD<sub>n</sub> can be selected by PS\_RXD<sub>n</sub> (RXD<sub>n</sub> pin Only one of them can be selected), TXD<sub>n</sub> port can be selected through the port configuration register (can be selected at the same time, or select one. If selected at the same time, all will output the corresponding waveform).

When using, it is recommended to set the working mode first, and then configure the corresponding port as a serial port.

## 20.3 UARTn baud rate

UARTn is in mode 0, the baud rate is fixed to the system clock divided by 12 ( $F_{sys}/12$ ); in mode 2, the baud rate is fixed to the system clock divided by 32 or 64 Frequency division ( $F_{sys}/32$ ,  $F_{sys}/64$ ); In mode 1 and mode 3, the baud rate is generated by the timer Timer1 or Timer4 or Timer2 or the BRT module. Which timer the chip selects as the baud rate clock source is determined by the register FUNCCR.

### 20.3.1 Baud rate clock source

UARTn is in mode 1 and mode 3, the baud rate clock source selection is as follows:

- 1) UART0 baud rate clock source selection:
  - FUNCCR[2:0]=000, select Timer1 as the baud rate of UART0 generator;
  - FUNCCR[2:0]=001, select Timer4 as the baud rate generator of UART0;
  - FUNCCR[2:0]=010, select Timer2 as the baud rate generator of UART0;
  - FUNCCR[2:0]=011, select BRT as the baud rate generator of UART0;
- 2) UART1 baud rate clock source selection:
  - FUNCCR[6:4]=000, select Timer1 as the baud rate generator of UART1;
  - FUNCCR[6:4]=001, select Timer4 as the baud rate generator of UART1;
  - FUNCCR[6:4]=010, Timer2 is selected as the baud rate generator of UART1;
  - FUNCCR[6:4]=011, BRT is selected as the baud rate generator of UART1;

### 20.3.2 Baud rate calculation

When UARTn is in mode 1 and mode 3, the baud rate calculation formula for different clock sources is as follows:

1) The formula of the baud rate when Timer1 or Timer4 works in 8-bit auto-reload mode:

$$\text{BaudRate} = \frac{F_{\text{sys}} \times 2^{\text{SMODn}}}{32 \times (4 \times 3^{1-\text{T4M}}) \times (256 - \text{THx})} \quad (x=1,4)$$

SMODn is the baud rate selection bit, which is set by the register PCON. T1M is the timer 1 clock selection bit, which is set by the register CKCON[4], and T4M is the timer 4 clock selection bit, which is set by the register T34MOD[6]. That is, the TH1/TH4 value of Timer1 or Timer4 at the corresponding baud rate should be set to:

$$\text{THx} = 256 - \frac{F_{\text{sys}} \times 2^{\text{SMODn}}}{32 \times (4 \times 3^{1-\text{T4M}}) \times \text{BaudRate}} \quad (x=1,4)$$

2) The formula of the baud rate when Timer2 works in overflow auto-reload mode:

$$\text{BaudRate} = \frac{F_{\text{sys}} \times 2^{\text{SMODn}}}{32 \times (12 \times 2^{\text{T2PS}}) \times (65536 - \{\text{RLDH}, \text{RLDL}\})}$$

T2PS is the timer 2 clock prescaler selection bit, which is set by register T2CON[7]. That is, the value of "{RLDH,RLDL}" should be set to "{RLDH,RLDL}" under the corresponding baud rate of Timer2:

$$\{\text{RLDH}, \text{RLDL}\} = 65536 - \frac{F_{\text{sys}} \times 2^{\text{SMODn}}}{32 \times (12 \times 2^{\text{T2PS}}) \times \text{BaudRate}}$$

3) When BRT is used as a baud rate generator, the baud rate formula:

$$\text{BaudRate} = \frac{F_{\text{sys}} \times 2^{\text{SMODn}}}{32 \times (65536 - \{\text{BRTDH}, \text{BRTDL}\}) \times 2^{\text{BRTCKDIV}}}$$

BRTCKDIV is the BRT timer prescaler selection bit, which is set by the register BRTCON. That is, the value of "{BRTDH,BRTDL}" should be set as BRT at the corresponding baud rate:

$$\{\text{BRTDH}, \text{BRTDL}\} = 65536 - \frac{F_{\text{sys}} \times 2^{\text{SMODn}}}{32 \times 2^{\text{BRTCKDIV}} \times \text{BaudRate}}$$

### 20.3.3 baud rate error

When UARTn is in mode 1 and mode 3, different baud rate clock sources are selected, and the error under different baud rates is as follows:

Table 1) and 2) are part of the baud rate related information in the 8-bit auto-reload mode of Timer 1/Timer 4 in the variable baud rate mode. Table 3) and 4) are part of the baud rate related information when the overflow rate of the BRT timer is used as the UART clock source in the variable baud rate mode.

1) SMODn=0, T1M=1/T4M=1

Baud rate	Fsys=8MHz			Fsys=16MHz			Fsys=24MHz			Fsys=48MHz		
	{TH1, TH4}	Actual Rate	% Error	{TH1, TH4}	Actual Rate	% Error	{TH1, TH4}	Actual Rate	% Error	{TH1, TH4}	Actual Rate	% Error
4800	243	4808	-0.16	230	4808	-0.16	217	4808	-0.16	178	4808	-0.16
9600	--	--	--	247	9615	-0.16	236	9375	2.34	217	9615	-0.16
19200	--	--	--	--	--	--	246	18750	2.34	236	18750	2.34
38400	--	--	--	--	--	--	251	37500	2.34	246	37500	2.34
115200	--	--	--	--	--	--	--	--	--	--	--	--
250000	--	--	--	--	--	--	--	--	--	--	--	--
500000	--	--	--	--	--	--	--	--	--	--	--	--

2) SMODn=1, T1M=1/T4M=1

Baud rate	Fsys=8MHz			Fsys=16MHz			Fsys=24MHz			Fsys=48MHz		
	{TH1, TH4}	Actual Rate	% Error	{TH1, TH4}	Actual Rate	% Error	{TH1, TH4}	Actual Rate	% Error	{TH1, TH4}	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	19230	-0.16	236	18750	2.34	217	19231	-0.16
38400	--	--	--	--	--	--	246	37500	2.34	236	37500	2.34
115200	--	--	--	--	--	--	--	--	--	--	--	--
250000	--	--	--	--	--	--	--	--	--	--	--	--
500000	--	--	--	--	--	--	--	--	--	--	--	--

## 3) SMODn=0, BRTCKDIV=0

Baud rate	Fsys=8MHz			Fsys=16MHz			Fsys=24MHz			Fsys=48MHz		
	{BRTH, BRTL}	Actual Rate	% Error	{BRTH, BRTL}	Actual Rate	% Error	{BRTH, BRTL}	Actual Rate	% Error	{BRTH, BRTL}	Actual Rate	% Error
4800	65484	4808	-0.16	65432	4808	-0.16	65380	4808	-0.16	65224	4808	-0.16
9600	65510	9615	-0.16	65484	9615	-0.16	65458	9615	-0.16	65380	9615	-0.16
19200	65523	19231	-0.16	65510	19231	-0.16	65497	19231	-0.16	65458	19231	-0.16
38400	--	--	--	65523	38462	-0.16	65516	37500	2.34	65497	38462	-0.16
115200	--	--	--	--	--	--	--	--	--	65523	115385	-0.16
250000	--	--	--	--	--	--	--	--	--	65530	250000	0
500000	--	--	--	--	--	--	--	--	--	65533	500000	0

## 4) SMODn=1, BRTCKDIV=0

Baud rate	Fsys=8MHz			Fsys=16MHz			Fsys=24MHz			Fsys=48MHz		
	{BRTH, BRTL}	Actual Rate	% Error	{BRTH, BRTL}	Actual Rate	% Error	{BRTH, BRTL}	Actual Rate	% Error	{BRTH, BRTL}	Actual Rate	% Error
4800	65432	4808	-0.16	65328	4808	-0.16	65224	4792	0.16	64911	4800	0
9600	65484	9615	-0.16	65432	9615	-0.16	65380	9615	-0.16	65224	9615	-0.16
19200	65510	19231	-0.16	65484	19231	-0.16	65458	19231	-0.16	65380	19231	-0.16
38400	65523	38462	-0.16	65510	38462	-0.16	65497	38462	-0.16	65458	38462	-0.16
115200	--	--	--	--	--	--	65523	115385	-0.16	65510	115385	-0.16
250000	--	--	--	--	--	--	--	--	--	65524	250000	0
500000	--	--	--	--	--	--	--	--	--	65530	500000	0
1000000	--	--	--	--	--	--	--	--	--	65533	1000000	0

## 20.4 UARTn register

UARTn has the same function as the standard 8051 UART. The related registers are: FUNCCR, SBUFn, SCOn, PCOn, IE, IP, EIP3. The UARTn data buffer (SBUFn) consists of two independent registers: send and receive registers. The data written in SBUFn will set this data in the UARTn output register and start transmission; the data read in SBUFn will read data from the UARTn receiving register. SCOn0 register supports bit addressing operation, SCOn1 register does not support bit addressing operation, please pay attention when using assembly language. The baud rate is doubled by setting the registers PCOn.

### 20.4.1 UART0/1 baud rate selection register FUNCCR

0x91	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
FUNCCR	-	UART1_CKS2	UART1_CKS1	UART1_CKS0	-	UART0_CKS2	UART0_CKS1	UART0_CKS0
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, must be 0.
- Bit6-Bit4 UART1\_CKS<2:0>: UART1 timer clock source selection
- 000 = Timer1 overflow clock;
  - 001 = Timer4 overflow clock;
  - 010 = Timer2 overflow clock;
  - 011 = BRT overflow clock; overflow clock
  - other = access is prohibited.
- Bit3 - Reserved, must be 0's.
- Bit2-Bit0 UART0\_CKS<2:0>: Timer clock source selection for UART0
- 000 = Timer1 overflow clock;
  - 001 = Timer4 overflow clock;
  - 010 = Timer2 overflow clock;
  - 011 = BRT overflow clock;
  - Other = prohibit access

### 20.4.2 UARTn buffer register SBUFn

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
SBUFn	BUFFERn7	BUFFERn6	BUFFERn5	BUFFERn4	BUFFERn3	BUFFERn2	BUFFERn1	BUFFERn0
Read and write	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

BANK0: register SBUF0 address 0x99; register SBUF1 address 0xEB.

- Bit7~Bit0 BUFFERn<7:0>: buffer data register.
- Write: UARTn starts to send data.
  - Read: read the received data.



### 20.4.3 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

BANK0: register SCOn0 address 0x98; register SCOn1 address 0xEA.

Bit7~Bit6	UnSM0-UnSM1:	Multi-machine communication control bits; 00= master synchronous mode; 01= 8-bit asynchronous mode, with variable baud rate; 10= 9-bit asynchronous mode, with baud rate $F_{sys}/32$ or $F_{sys}/64$ ; 11= 9-bit asynchronous mode, the baud rate is variable.
Bit5	UnSM2:	Multi-machine communication control bit; 1= enable; 0= disable.
Bit4	UnREN:	Receive enable bit; 1= enable; 0= disable.
Bit3	UnTB8:	The 9th bit of sending data, mainly used for sending in 9-bit asynchronous mode; 1= The 9th bit is 1; 0= The 9th bit is 0.
Bit2	UnRB8:	The 9th bit of the received data, mainly used for the transmission of 9-bit asynchronous mode; 1= The 9th bit of data is 1; 0= receivedThe 9th bit of data received is 0.
Bit1	TIn:	Transmit interrupt flag bit (requires software to clear); 1= that the transmit buffer is empty, and the next frame data can be sent. 0= -
Bit0	RIn:	Receive interrupt flag bit (requires software to clear); 1= that the receive buffer is full, and the next frame of data can be received after reading. 0= -

UARTn mode is as follows:

SMn0	SMn1	mode	description	Baud rate
0	0	0	Shift register	$F_{sys}/12$
0	1	1	8-Bit UART is	controlled by Timer4/Timer1/Timer2/BRT
1	0	2	9-Bit UART	SMODn =0: $F_{sys}/64$ ; SMODn=1: $F_{sys}/32$
1	1	3	9-Bit UART is	controlled by Timer4/Timer1/Timer2/BRT.

### 20.4.4 PCON register

0x87	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PCON	SMOD0	SMOD1	-	-	-	SWE	STOP	IDLE
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 in BANK0

Bit7 in	SMOD0: UART0 baud rate multiplier bit; 1= UART0 baud rate is doubled; 0= UART0 baud rate is normal.
Bit6	SMOD1: UART1 baud rate multiplier bit; 1= UART1 baud rate doubled; 0= UART1 baud rate is normal.
Bit5 ~ Bit3	- Reserved, must be 0's.
Bit2	SWE: STOP status function wake-up enable bit; (Regardless of the value of SWE, the system can be restarted by power-off reset or enabled external reset) 0= Function wake-up is disabled; 1= Function wake-up is enabled (can be waked up by external interrupt and timed) wake).
Bit1	STOP: Sleep status control bit; 1= Enter sleep status (automatically clear when exiting STOP mode); 0= Not enter sleep status.
Bit0	IDLE: Idle state control bit; 1= Enter idle state (automatically clear when exiting IDLE mode); 0= Not enter idle state

## 20.5 UARTn interrupt

The interrupt number of UART0 is 4, and its interrupt vector is 0x0023.

The interrupt number of UART1 is 6, and its interrupt vector is 0x0033.

To enable the UARTn interrupt, the enable bit ESn must be set to 1, and the total interrupt enable bit EA must be set to 1. If UARTn related interrupt enable is turned on, when TIn=1 or RIn=1, the CPU will enter the corresponding interrupt service routine. TIn/RIn has nothing to do with the state of ESn and needs to be cleared by software. For detailed description, refer to register SCOnn.

### 20.5.1 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= Allow all unmasked interrupts; 0= Disable all interrupts.
Bit6	ES1: UART1 interrupt enable bit; 1= Enable UART1 interrupt; 0= Disable UART1 interrupt.
Bit5	ET2: TIMER2 total interrupt enable bit; 1= Enable all TIMER2 interrupts; 0= Disable all TIMER2 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.

### 20.5.2 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/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, must be 0.
Bit6	PS1:	UART1 interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit5	PT2:	TIMER2 interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit4	PS0:	UART0 interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit3	PT1:	TIMER1 interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit2	PX1:	External interrupt 1 interrupt priority control bit; 1= Set as high-level interrupt; 0= Set as low-level interrupt.
Bit1	PT0:	TIMER0 interrupt priority control bit; 1= set as high-level interrupt; 0= set as low-level interrupt.
Bit0	PX0:	External interrupt 0 interrupt priority control bit; 1= Set as high-level interrupt; 0= Set as low-level interrupt.

### 20.5.3 Interrupt priority register EIP3

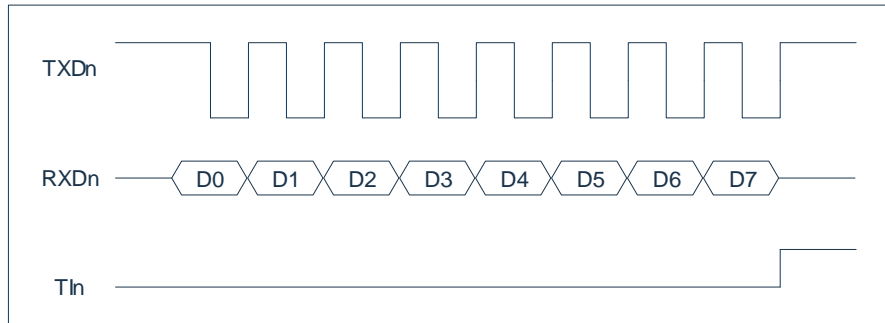
0xBB	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIP3	--	--	--	--	PLVD	PLSE	--	--
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit7~Bit4            -- Reserved, all must be 0.  
 Bit3                PLVD: LVD interrupt priority control bit;  
                       1= set as high-level interrupt;  
                       0= set as low-level interrupt.  
 Bit2                PLSE: LSE interrupt priority control bit;  
                       1= set as high-level interrupt;  
                       0= set as low-level interrupt.  
 Bit1~Bit0           -- Reserved, all must be 0.

## 20.6 UARTn mode

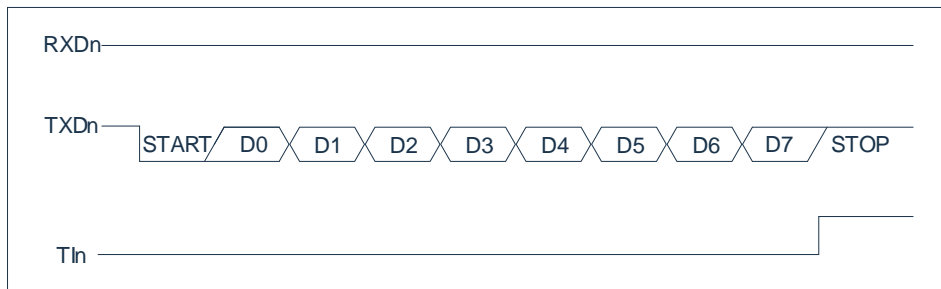
### 20.6.1 Mode 0-synchronous mode

Pin RXDn is used as input or output, and TXDn is used as clock output. The TXDn output is a shift clock. The baud rate is fixed at 1/12 of the system clock frequency. 8 bits are transmitted in LSB first. Initialize the reception by setting the flag in SCONn, set as: Rln = 0 and RENn = 1. The timing diagram of Mode 0 is shown in the figure below::



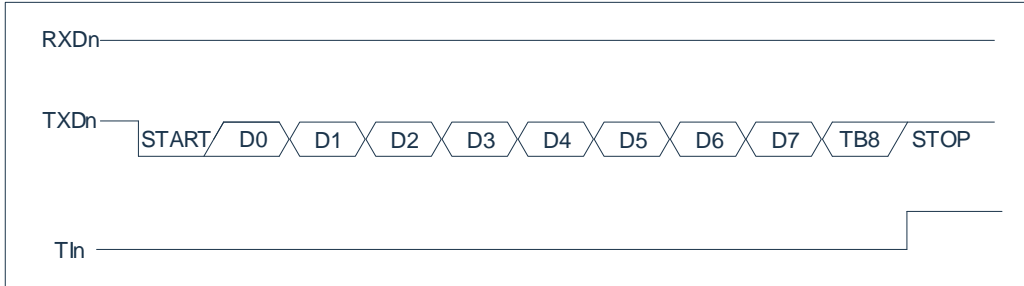
### 20.6.2 Mode 1-8 bit asynchronous mode (variable baud rate)

Pin RXDn is used as input, TXDn is used as serial output. Send 10 bits: start bit (always 0), 8-bit data (LSB first) and stop bit (always 1). When receiving, the start bit is transmitted synchronously, 8 data bits can be obtained by reading SBUFn, and the stop bit is set in SCONn with the flag RBN8. The baud rate is variable and depends on the TIMER1/TIMER2/TIMER4/BRT mode. The timing diagram of Mode 1 is shown in the figure below:



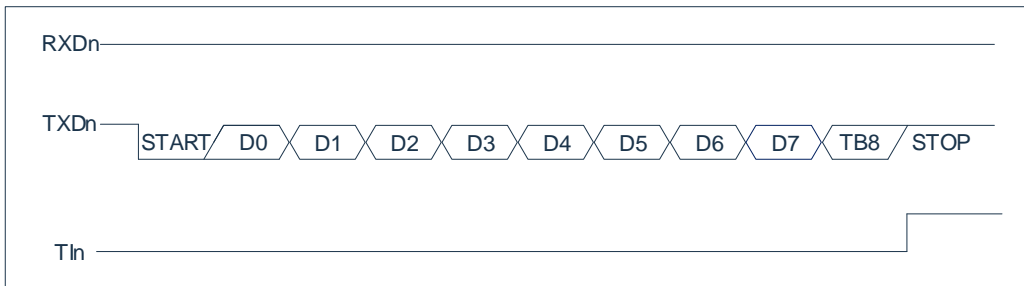
### 20.6.3 Mode 2-9 bit asynchronous mode (fixed baud rate)

This mode is similar to Mode 1, but there are two differences. The baud rate is fixed at 1/32 or 1/64 of the CLK clock frequency, 11-bit transmission and reception: start bit (0), 8-bit data (LSB first), programmable ninth bit and stop bit (1). The ninth bit can be used to control the parity check of the UARTn interface: when sending, the bit TBn8 in SCONn is output as the ninth bit, and when receiving, the ninth bit affects the RBn8 in SCONn. The timing diagram of Mode 3 is shown in the figure below:



### 20.6.4 Mode 3-9 bit asynchronous mode (variable baud rate)

The only difference between Mode 2 and Mode 3 is that the baud rate in Mode 3 is variable. When REN0=1, data reception is enabled. The baud rate is variable and depends on the TIMER1/TIMER2/TIMER4/BRT mode. The timing diagram of Mode 4 is shown in the following figure:



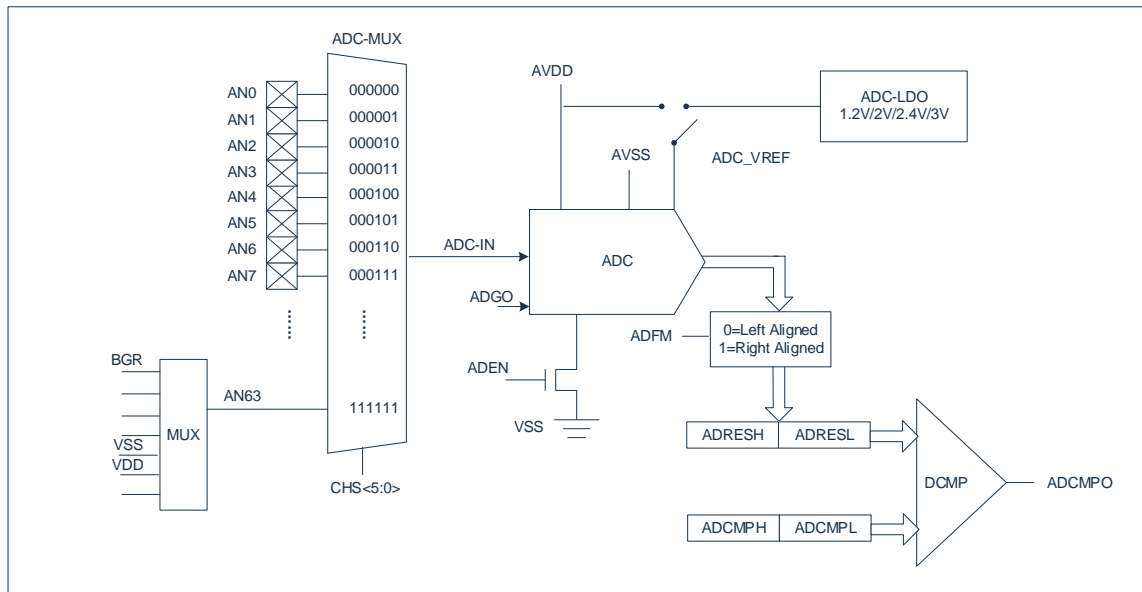
## 21. Analog-to-digital converter (ADC)

### 21.1 overview

The analog-to-digital converter (ADC) can convert the analog input signal into a 12-bit binary number representing the signal. The block diagram of the ADC structure is shown in the figure below.

The port analog input signal and internal analog signal are connected to the input of the analog-to-digital converter after passing through the multiplexer. The analog-to-digital converter uses the successive approximation method to produce a 12-bit binary result, and save the result in the ADC result register (ADRESL and ADRESH), the ADC can generate an interrupt after the conversion is completed. The ADC conversion result is compared with the value of the ADC comparison data register (ADCMPH and ADCMPL), and the comparison result is stored in the ADCMPO flag.

ADC reference voltage is always generated internally, and can be provided by AVDD or by internal ADC-LDO.





## 21.2 ADC configuration

When configuring and using ADC, the following factors must be considered:

- Port configuration.
- Channel selection.
- ADC conversion clock source.
- Interrupt control.
- The storage format of the result.

### 21.2.1 Port configuration

ADC can convert both analog signals and digital signals. When converting analog signals, the corresponding port needs to be configured as an analog port.

Note: Applying analog voltage to pins defined as digital inputs may cause overcurrent in the input buffer.

### 21.2.2 Channel selection

The register ADCCHS determines which channel is connected to the analog-to-digital converter.

If the channel is changed, a certain delay is required before the next conversion starts. The ADC delay time is shown in the table below:

Delay time	working voltage
500ns	2.5~4.5V
200ns	4.5~5.5V

### 21.2.3 ADC reference voltage

ADC reference voltage is provided by the chip's VDD by default, or it can be provided by the internal ADC-LDO. ADC-LDO can choose 4 kinds of voltage output: 1.2V/2.0V/2.4V/3.0V.

### 21.2.4 Clock conversion

The ADCKS bit of the ADCON1 register can be set by software to select the conversion clock source.

The time to complete a bit conversion is defined as  $T_{ADCK}$ . A complete 12-bit conversion requires 18.5  $T_{ADCK}$  cycles (the time that ADGO continues to be high to complete a conversion). Must comply with the corresponding  $T_{ADCK}$  specification in order to obtain the correct conversion results. The following table is an example of selecting the ADC clock correctly.

F <sub>sys</sub>	F <sub>ADCK</sub> (T <sub>A</sub> =25°C)		
	V <sub>REF</sub> =V <sub>REF1</sub> =AVDD (AVDD=VDD)	V <sub>REF</sub> =V <sub>REF2</sub> =1.2V	V <sub>REF</sub> =V <sub>REF3</sub> =2.0V V <sub>REF</sub> =V <sub>REF4</sub> =2.4V V <sub>REF</sub> =V <sub>REF5</sub> =3.0V
8MHz	F <sub>sys</sub> /4	F <sub>sys</sub> /256	F <sub>sys</sub> /16
16MHz	F <sub>sys</sub> /8	forbidden	F <sub>sys</sub> /32
24MHz	F <sub>sys</sub> /16	forbidden	F <sub>sys</sub> /64
48MHz	F <sub>sys</sub> /32	forbidden	F <sub>sys</sub> /128

Note: Any change in the system clock frequency will change the ADC clock frequency, which will negatively affect the ADC conversion result.

### 21.2.5 Result format The result of

The result of 12-bit A/D conversion can be in two formats: left-justified or right-justified. The output format is controlled by the ADFM bit in the ADCON0 register.

When ADFM=0, the AD conversion result is aligned to the left;

when ADFM=1, the AD conversion result is aligned to the right.

## 21.3 ADC hardware trigger start

In addition to software trigger ADC conversion, the ADC module also provides a hardware trigger start method. One is the edge trigger mode of the external port, and the other is the edge or period trigger mode of the PWM.

Using hardware to trigger ADC needs to set ADCEX to 1, even if the ADC function can be triggered externally. The hardware trigger signal will set the ADGO bit to 1 after a certain delay, and it will be automatically cleared after the conversion is completed. After the hardware trigger function is turned on, the software trigger function will not be turned off. In the ADC idle state, writing 1 to the ADGO bit can also start AD conversion.

### 21.3.1 External port edge trigger ADC

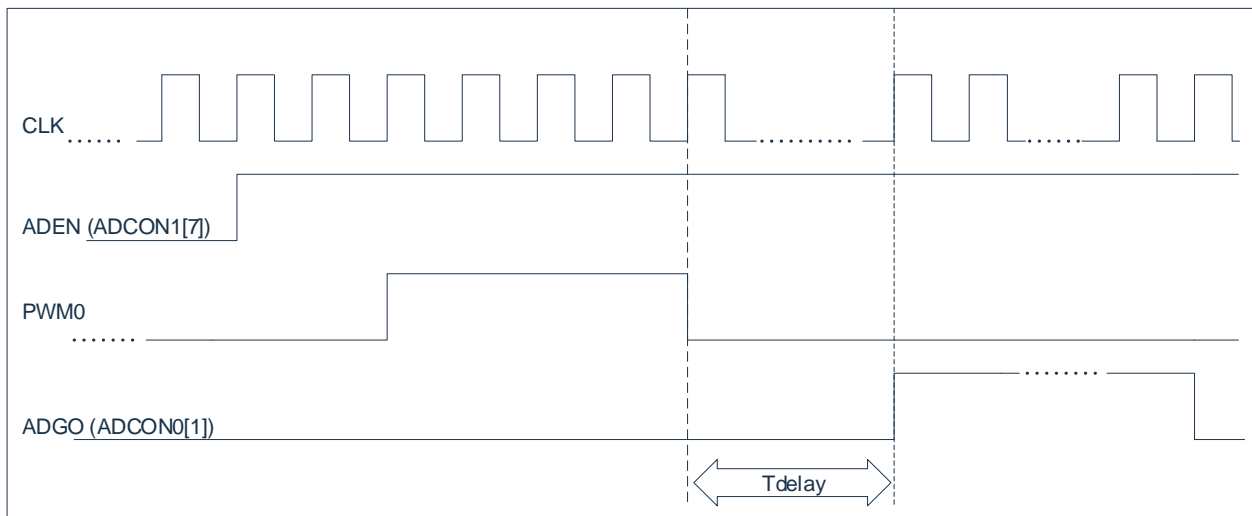
ADET pin edge automatically trigger ADC conversion. At this time, ADTGS[1:0] needs to be 11 (select external port edge trigger), and ADEGS[1:0] can select which edge trigger.

### 21.3.2 PWM trigger ADC

PWM can choose to trigger ADC conversion by edge or zero point. ADTGS[1:0] select PWM channel (PG0, PG2, PG4), ADEGS[1:0] can select edge type or period type trigger mode.

### 21.3.3 Hardware trigger start delay

After the hardware trigger signal is generated, AD conversion is not started immediately, and ADGO is set to 1 after a certain delay. The delay is determined by ADDLY[9:0]. The delay time of the hardware trigger signal:  $(ADDLY+3)*T_{sys}$ , the structure diagram of the delay trigger is shown in the figure below:



## 21.4 ADC result comparison

ADC module provides a set of digital comparators for comparing the result of ADC with pre-loaded { ADCMPH, ADCMPL} value size. The result of each ADC conversion will be compared with the preset value ADCMP. The result of the comparison is stored in the ADCMPO flag bit. After the conversion is completed, the flag bit will be automatically updated. The ADCMPPS bit can change the polarity of the output result.

## 21.5 ADC working principle

### 21.5.1 Start conversion

To enable the ADC module, you must first set the ADEN bit of the ADCON1 register to 1, and then set the ADGO bit of the ADCON0 register to 1 to start analog-to-digital conversion (ADGO cannot be set to 1 when ADEN is 0).

### 21.5.2 Completion of the 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.

### 21.5.3 Termination of conversion

If the conversion must be terminated before the conversion is completed, the uncompleted analog-to-digital conversion result will not be updated to the ADRESH:ADRESL register. Therefore, the ADRESH:ADRESL register will hold the value obtained from the last conversion.

Note: Device reset will force all registers to enter the reset state. Therefore, the reset will shut down the ADC module and terminate any pending conversions.

### 21.5.4 A/D conversion steps

The configuration steps for using ADC for analog-to-digital conversion are as follows:

- 1) Port configuration:
  - disable pin output driver (see PxTRIS register);
  - configure the pin as an analog input pin.
- 2) Configure ADC interrupt (optional):
  - clear ADC interrupt flag bit;
  - enable ADC interrupt;
  - allow peripheral interrupt;
  - allow global interrupt.
- 3) Configure the ADC module:
  - select the ADC conversion clock;
  - select the ADC input channel;
  - select the format of the result;
  - start the ADC module.
- 4) Wait for the required acquisition time.
- 5) Set ADGO to 1 to start conversion.
- 6) Wait for the ADC conversion to complete by one of the following methods:

- query the ADGO bit;
  - wait for the ADC interrupt (allow interrupts).
- 7) Read the ADC result.
  - 8) Clear the ADC interrupt flag bit (if interrupts are enabled, this operation is required).

Note: If the user attempts to resume sequential code execution after waking the device from sleep mode, the global interrupt must be disabled.

### 21.5.5 When the conversion process

When the system enters the dormant state, it is recommended to wait for the ADC ongoing conversion to complete before entering the dormant state.

If the ADC goes to sleep during the ongoing conversion process, the conversion is terminated. Need to re-convert after waking up.

## 21.6 related registers related

There are mainly 11 registers related to AD conversion, namely:

- AD control register ADCON0, ADCON1, ADCON2, ADCCHS, ADCLDO;
- comparator control register ADCMPC;
- delay data register ADDLYL;
- AD result data register ADRESH/L;
- comparator data register ADCMPH/L.

### 21.6.1 AD control register ADCON0

0xDF	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADCON0	--	ADFM	ANACH3	ANACH2	ANACH1	ANACH0	ADGO	--
Read/write	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, shall be 0.
Bit6	ADFM:	ADC conversion result format selection bit; 1= Align to the right; 0= Align to the left.
Bit5~Bit2	ANACH<3:0>:	ADC channel 63 input source selection bits; 0000= BGR(1.2V); 0001= Reserved, forbidden to be used. 0010= Reserved, forbidden to be used. 0011= Reserved, forbidden to be used 0100= Reserved, forbidden to be used. 0101= VSS (ADC Ground reference) ; 0110= Reserved, forbidden to be used. 0111= VDD (ADC default reference voltage) . Others= Reserved, forbidden to be used.
Bit1	ADGO:	ADC conversion start bit (when the bit is set to 1, ADEN must be 1, otherwise the operation is invalid); 1= write: start ADC conversion, (the bit will be 1 when ADC is triggered by hardware); read: ADC is Perform the conversion. 0= Write: invalid. Read: ADC is idle/conversion completed; during ADC conversion (ADGO=1), any software and hardware trigger signals will be ignored.
Bit0	-	Reserved, must be zero.

### 21.6.2 AD control register ADCON1

0xDE	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADCON1	ADEN	ADCKS2	ADCKS1	ADCKS0	--	--	--	--
Read/write	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                    ADEN: ADC enable bit;  
                           1= Enable ADC;  
                           0= Disable ADC, no working current is consumed.

Bit6~Bit4            ADCKS<2:0>: ADC conversion clock selection bits;  
                           000= F<sub>sys</sub>/2;                    100= F<sub>sys</sub>/32;  
                           001= F<sub>sys</sub>/4;                    101= F<sub>sys</sub>/64;  
                           010= F<sub>sys</sub>/8;                    110= F<sub>sys</sub>/ 128;  
                           011= F<sub>sys</sub>/16;                   111= F<sub>sys</sub>/256.

Bit3~Bit0            - Reserved, shall be 0.

### 21.6.3 AD control register ADCON2

0xE9	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADCON2	ADCEX	--	ADTGS1	ADTGS0	ADEGS1	ADEGS0	--	--
Read/write	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	ADCEX:	ADC hardware trigger enable bit; 1 = Enable; 0 = Disable.
Bit6	-	Reserved, must be zero.
Bit5~Bit4	ADTGS<1:0>:	ADC hardware trigger source selection bits; 00 = PG0 (PWM0); 01 = PG2 (PWM2); 10 = PG4 (PWM4); 11 = port pin (ADET).
Bit3~Bit2	ADEGS<1:0>:	ADC hardware trigger edge selection bits; 00 = Falling edge; 01 = Rising edge; 10 = Reserved, forbidden to be used. 11 = PWM cyclezero point of PWM cycle.
Bit1 ~ Bit0	-	Reserved, all must be 0.



### 21.6.4 AD Channel selection register ADCCHS

0xD9	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADCCHS	--	--	CHS5	CHS4	CHS3	CHS2	CHS1	CHS0
Read/write	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, shall be 0.

Bit5~Bit0	CHS<5:0>:	Analog channel selection bits;	
	000000=	AIN0;	010000= AIN16;
	000001=	AIN1;	010001= AIN17;
	000010=	AIN2;	010010= AIN18;
	000011=	AIN3;	010011= AIN19;
	000100=	AIN4;	010100= AIN20;
	000101=	AIN5;	010101= AIN21;
	000110=	AIN6;	010110= AIN22;
	000111=	AIN7;	010111= AIN23;
	001000=	AIN8;	011000= Forbidden ;
	001001=	AIN9;	011001= Forbidden;
	001010=	AIN10;	011010= Forbidden;
	001011=	AIN11;	011011= Forbidden;
	001100=	AIN12;	011100= Forbidden;
	001101=	AIN13;	011101= Forbidden;
	001110=	AIN14;	011110= Forbidden;
	001111=	AIN15;	011111= Forbidden;
			100000= Forbidden;
			100001= Forbidden;
			100010= Forbidden;
			100011= Forbidden;
			100100= Forbidden;
			100101= Forbidden;
			100110= Forbidden;
			100111= AIN39;
			101000= AIN40;
			101001= AIN41;
			101010= AIN42;
			101011= AIN43;
			101100= AIN44;
			101110= AIN46;
			Others= Forbidden;
			111111= Refer to ADCON0.ANACH description.

### 21.6.5 AD comparator control register ADCMPC

0xD1	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADCMPC	--	ADCMPPS	--	ADCMPO	--	--	ADDLY9	ADDLY8
Read/write	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, must be 0.

Bit6 ADCMPPS: ADC comparator output polarity selection bit;  
 1= if ADRES<ADCMP, then ADCMPO=1;  
 0= if ADRES>=ADCMP, then ADCMPO=1.

Bit5 - Reserved, must be zero.

Bit4 ADCMPO: ADC comparator output bit.  
 This bit outputs the result of the ADC comparator output, and it will be updated every time the ADC conversion ends.

Bit3 ~ Bit2 - Reserved, must be 0's.

Bit1~Bit0 ADDLY<9:8>: ADC hardware trigger delay data[9:8] bits.

### 21.6.1 AD hardware trigger delay data register ADDLYL

0xD3	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADDLYL	ADDLY7	ADDLY6	ADDLY5	ADDLY4	ADDLY3	ADDLY2	ADDLY1	ADDLY0
Read/write	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      ADDLY<7:0>:    ADC hardware trigger delay data low 8 bits.

### 21.6.2 AD data register high ADRESH, ADFM=0 (left align)

0xDD	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADRESH	ADRES11	ADRES10	ADRES9	ADRES8	ADRES7	ADRES6	ADRES5	ADRES4
Read/write	R	R	R	R	R	R	R	R
Reset value	X	X	X	X	X	X	X	X

Bit7~Bit0      ADRES<11:4>:    ADC result register bits.  
The 11-4 digits of the 12-bit conversion result.

### 21.6.3 AD data register lower ADRESL, ADFM = 0(left align)

0xDC	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADRESL	ADRES3	ADRES2	ADRES1	ADRES0	--	--	--	--
Read/write	R	R	R	R	--	--	--	--
Reset value	X	X	X	X	--	--	--	--

Bit7~Bit4      ADRES<3:0>:    ADC result register bits.  
Bits 3-0 of the 12-bit conversion result.

Bit3~Bit0      not used.  
are

### 21.6.4 AD data register high ADRESH, ADFM=1 (right align)

0xDD	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADRESH	-	-	-	-	ADRES11	ADRES10	ADRES9	ADRES8
Read/write	-	-	-	-	R	R	R	R
reset Value	-	-	-	-	X	X	X	X

Bit7~Bit4      not used.  
are

Bit3~Bit0      ADRES<11:8>:    ADC result register bits.  
The 11th to 8th bits of the 12-bit conversion result.

### 21.6.5 AD data register low bit ADRESL, ADFM = 1 (right-align)

0xDC	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADRESL	ADRES7	ADRES6	ADRES5	ADRES4	ADRES3	ADRES2	ADRES1	ADRES0
Read/write	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 bits.  
Bits 7-0 of the 12-bit conversion result.



## 21.7 ADC Interrupt

The ADC module allows an interrupt to be generated after the completion of the analog-to-digital conversion. 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, and the ADCIF bit will be set to 1 after each conversion, regardless of whether ADC interrupts are allowed. The ADC interrupt enable and priority can be set by the following relevant register bits.

### 21.7.1 Interrupt mask register EIE2

0xAA	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIE2	SPIIE	I2CIE	WDTIE	ADCIE	PWMIE	-	ET4	ET3
Read/write	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<sup>2</sup>C interrupt enable bit;  
             1= Enable I<sup>2</sup>C interrupt;  
             0= Disable I<sup>2</sup>C interrupt.
- Bit5      WDTIE: WDT interrupt enable bit;  
             1= Enable WDT overflow interrupt;  
             0= Disable WDT overflow interrupt.
- Bit4      ADCIE: ADC interrupt enable bit;  
             1= enable ADC interrupt;  
             0= disable ADC interrupt.
- Bit3      PWMIE: PWM total interrupt enable bit;  
             1= allow all PWM interrupts;  
             0= disable all PWM interrupts.
- Bit2      - Reserved, must be zero.
- Bit1      ET4: Timer4 interrupt enable bit;  
             1= Enable Timer4 interrupt;  
             0= Disable Timer4 interrupt.
- Bit0      ET3: Timer3 interrupt enable bit;  
             1= Enable Timer3 interrupt;  
             0= Disable Timer3 interrupt.

### 21.7.2 Interrupt priority control register EIP2

0xBA	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIP2	PSPI	PI2C	PWDT	PADC	PPWM	-	PT4	PT3
Read/write	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      PSPI: SPI interrupt priority control bit;  
             1= set as high-level interrupt;  
             0= set as low-level interrupt.
- Bit6      PI2C: I<sup>2</sup>C interrupt priority control bit;  
             1= set as high-level interrupt;  
             0= set as low-level interrupt.
- Bit5      PWDT: WDT interrupt priority control bit;  
             1= set as high-level interrupt;  
             0= set as low-level interrupt.

- Bit4      PADC: ADC interrupt priority control bit;  
           1= set as high-level interrupt;  
           0= set as low-level interrupt.
- Bit3      PPWM: PWM interrupt priority control bit  
           1= set as high-level interrupt;  
           0= set as low-level interrupt.
- Bit2      - Reserved, must be zero.
- Bit1      PT4: TIMER4 interrupt priority control bit;  
           1= set as high-level interrupt;  
           0= set as low-level interrupt.
- Bit0      PT3: TIMER3 interrupt priority control bit;  
           1= set as high-level interrupt;  
           0= set as low-level interrupt.

### 21.7.3 Peripheral interrupt flag register EIF2

0xB2	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EIF2	SPIIF	I2CIF	-	ADCIF	PWMIF	--	TF4	TF3
Read/write	R	R	-	R/W	R	--	R/W	R/W
Reset value	0	0	0	0	0	0	0	0

- Bit7      SPIIF: SPI general interrupt indicator bit, read-only;  
           1 = SPI generates an interrupt (this bit is automatically cleared after clearing the specific interrupt flag bit);  
           0 = SPI does not generate an interrupt.
- Bit6      I2CIF: I<sup>2</sup>C total interrupt indicator bit, read-only;  
           1 = I<sup>2</sup>C generates an interrupt (after clearing the specific interrupt flag bit, this bit is automatically cleared);  
           0 = I<sup>2</sup>C does not generate an interrupt.
- Bit5      - Reserved, must be zero.
- Bit4      ADCIF: ADC interrupt flag bit;  
           1 = ADC conversion is completed and needs to be cleared by software;  
           0 = ADC conversion is not completed.
- Bit3      PWMIF: PWM general interrupt indicator bit, read-only;  
           1 = PWM generates an interrupt (this bit is automatically cleared after clearing the specific interrupt flag bit);  
           0 = PWM does not generate an interrupt.
- Bit2      - Reserved, must be zero.
- Bit1      TF4: Timer4 timer overflow interrupt flag bit;  
           1 = Timer4 timer overflows, it is automatically cleared by hardware when entering the interrupt service routine, and can also be cleared by software;  
           0 = Timer4 timer has no overflow.
- Bit0      TF3: Timer3 timer overflow interrupt flag bit;  
           1 = Timer3 timer overflows, it is automatically cleared by hardware when entering the interrupt service routine, and can also be cleared by software;  
           0 = Timer3 timer has no overflow.

## 22. Flash memory

### 22.1 Overview

FLASH memory includes program memory (APROM/BOOT) and non-volatile data memory (Data FLASH). The maximum program memory space is 32KB, divided into 64 sectors, each sector contains 512B. The maximum data memory space is 1KB, divided into 2 sectors, each sector contains 512B.

The FLASH memory can be accessed through the relevant special function register (SFR) to realize the IAP function, and the program space can also be CRC checked through the special function register (SFR). The SFR registers used to access the FLASH space are as follows:

- MLOCK
- MDATA
- MADRL
- MADRH
- PCRCDL
- PCRCDH
- MCTRL

The MLOCK register is used to enable memory operations, the MDATA register forms a byte for storing 8-bit data to be read/written, and the MADRL/MADRH register Store the address of the accessed MDATA unit or the address of the CRC check. The PCRCDL/PCRCDH register is used to keep the running result of the program CRC, and the MCTRL register is used for memory operation control.

Through the memory module interface, the memory can be read/written/erased. The memory allows byte reads and writes, and the write time is controlled by the on-chip timer. Before writing new data, make sure that the data in the address has been erased. The write and erase voltages are generated by an on-chip charge pump. The rated operating voltage of this charge pump is within the voltage range of the device for byte operations.

Flash memory erasing operation only supports sector erasing, not byte erasing. Before modifying the data of a certain address, it is recommended to save other data first, then erase the current sector, and finally write the data.

The chip supports the CRC check of the program space code. The check code is generated using the polynomial CRC16-CCITT "  $X^{16}+X^{12}+X^5+1$ ".

## 22.2 Related register

### 22.2.1 FLASH protection lock register MLOCK

0xFB	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
MLOCK	MLOCK7	MLOCK6	MLOCK5	MLOCK4	MLOCK3	MLOCK2	MLOCK1	MLOCK0
Read/write	W	W	W	W	W	W	W	W
Reset value	0	0	0	0	0	0	0	0

Bit7~Bit0 MLOCK< 7:0>: Memory operation enable bit (this register only supports write operations, read as 00H);  
 AAH= allow memory-related R/W/E operations;  
 00H/FFH= do not allow operations;  
 others= prohibit writing.

Instruction sequence required to modify MLOCK (do not insert any instructions inbetween) :

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

### 22.2.2 FLASH memory data register MDATA

0xFE	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
MDATA	MDATA7	MDATA6	MDATA5	MDATA4	MDATA3	MDATA2	MDATA1	MDATA0
Read/write	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

Bit7~Bit0 MDATA<7:0>: Data to be read or written to the program memory.

### 22.2.3 FLASH memory address register MADRL

0xFC	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
MADRL	MADRL7	MADRL6	MADRL5	MADRL4	MADRL3	MADRL2	MADRL1	MADRL0
Read/write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
reset Value	0	0	1	1	1	1	1	1

Bit7~Bit0 MADRL<7:0>: Specify the lower 8 bits of the address for memory read/write operations.

### 22.2.4 FLASH memory address register MADRH

0xFD	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
MADRH	MADRH7	MADRH6	MADRH5	MADRH4	MADRH3	MADRH2	MADRH1	MADRH0
Read/write	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 MADRH<7:0>: Specify the upper 8 bits of the address for memory read/write operations.

### 22.2.5 Lower 8 bits of the program CRC operation result data register PCRCDL

0xF9	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PCRCDL	PCRCD<7:0>							
Read/write	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      PCRCD<7:0>      Program CRC operation result low 8 bits data

### 22.2.6 Program CRC operation result data register high 8 bits PCRCDH

0xFA	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PCRCDH	PCRCD<15:8>							
Read/write	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      PCRCD<15: 8>      program CRC calculation result of the higher 8 bits of data

### 22.2.7 FLASH memory control register MCTRL

0xFF	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
MCTRL	--	--	MERR	MREG	MMODE1	MMODE0	CRCADR	MSTART
Read/write	R	R	R/W	R/W	R/W	R/W	R/W	R/W
Reset value	1	0	0	1	0	0	0	0

- Bit7~Bit6      --      Reserved
- Bit5      MERR: Operation error flag (write 0 to clear);  
 1= Before the programming operation starts, check that the data in the programming address is not "FFH" (not erased), and the writing operation is terminated immediately.  
 0= -
- Bit4      MREG: Flash area selection bit;  
 1= Select data area (lower 10 bits address is valid);  
 0= Select program area (lower 15 bits address is valid).
- Bit3~Bit2      MMODE<1:0>: Operation mode selection bits:  
 11= Erase operation mode (the range of erase operation is: the entire sector where the current address is located);  
 10= Write operation mode;  
 01= CRC mode;  
 00 = Read operation mode.
- Bit1      CRCADR: Program CRC check address selection bit;  
 1= End address selection bit;  
 0= Start address selection bit.
- Bit0      MSTART: Operation start control bit;  
 1= Start the program memory R/W/E/CRC check operation (after the operation is completed, it can be automatically cleared by hardware);  
 0= Write: Terminate or not start the program memory R/W/E /CRC check operation;  
 read: the operation is completed or the operation is not started.



## 22.3 Function description

During the FLASH memory read/write/erase operation, the CPU is in a suspended state, and when the operation is completed, the CPU continues to run instructions.

Operation memory instruction It must be followed by 6 NOP instructions, for example:

```
MOV MCTRL,#09H           ; write operation start
NOP NOP
NOP
NOP
NOP
NOP
MOV MCTRL,#01H           ; read operation start
NOP
NOP
NOP
NOP
NOP
NOP
```

The program CRC check command is set by the register MCTRL[3:2], the start address and the end address can be freely configured through the register MADRL/MADRH, and the result is stored in the register PCRCDL/PCRCDH.

This CRC operation can only access the program storage space, but not the data storage space. In the program space CRC check process, the CPU stops working and waits for the CPU to continue running after the CRC calculation is completed. The CRC check is performed in byte mode, from the initial address to the end address. After the current CRC check is performed, the MMODE[1:0]=00 of MCTRL must be set. The CRC check operation steps are as follows:

- 1) Enable access to the program memory register:  
TA = 0xAA;  
TA = 0x55;  
MLOCK=0xAA //the default value is 00.
- 2) Check result before clearing program CRC: PCRCDL=0x00, PCRCDH=0x00.
- 3) Set the start and end addresses of the program CRC check: MCTRL[1]=0, set the start address through MADRL/MADRH; MCTRL[1]=1, set the end address through MADRL/MADRH.
- 4) Start program CRC check command: MCTRL=0x05.
- 5) Wait for the end of the program CRC check: MCTRL[0] is cleared to 0 by hardware after the CRC check ends.
- 6) Read the program CRC check result: PCRCDL stores the low 8 bits of the program CRC operation result; PCRCDH stores the high 8 bits of the program CRC operation result.

## 23. Unique ID (UID)

### 23.1 Overview

Each chip has a different 96-bit unique identification number, that is, unique identification. It has been set at the factory and cannot be modified by the user.

### 23.2 UID register description

#### UID0

F5E0H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
UID0	UID7	UID6	UID5	UID4	UID3	UID2	UID1	UID0
R/W	R	R	R	R	R	R	R	R
Reset value	X	X	X	X	X	X	X	X

Bit7~Bit0      UID<7:0>

#### UID1

F5E1H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
UID1	UID15	UID14	UID13	UID12	UID11	UID10	UID9	UID8
R/W	R	R	R	R	R	R	R	R
Reset value	X	X	X	X	X	X	X	X

Bit7~Bit0      UID<15:8>

#### UID2

F5E2H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
UID2	UID23	UID22	UID21	UID20	UID19	UID18	UID17	UID16
R/W	R	R	R	R	R	R	R	R
Reset value	X	X	X	X	X	X	X	X

Bit7~Bit0      UID<23:16>

#### UID3

F5E3H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
UID3	UID31	UID30	UID29	UID28	UID27	UID26	UID25	UID24
R/W	R	R	R	R	R	R	R	R
Reset value	X	X	X	X	X	X	X	X

Bit7~Bit0      UID<31:24>

## UID4

F5E4H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
UID4	UID39	UID38	UID37	UID36	UID35	UID34	UID33	UID32
R/W	R	R	R	R	R	R	R	R
Reset value	X	X	X	X	X	X	X	X

Bit7~Bit0      UID&lt;39:32&gt;

## UID5

F5E5H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
UID5	UID47	UID46	UID45	UID44	UID43	UID42	UID41	UID40
R/W	R	R	R	R	R	R	R	R
Reset value	X	X	X	X	X	X	X	X

Bit7~Bit0      UID&lt;47:40&gt;

## UID6

F5E6H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
UID6	UID55	UID54	UID53	UID52	UID51	UID50	UID49	UID48
R/W	R	R	R	R	R	R	R	R
Reset value	X	X	X	X	X	X	X	X

Bit7~Bit0      UID&lt;55:48&gt;

## UID7

F5E7H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
UID7	UID63	UID62	UID61	UID60	UID59	UID58	UID57	UID56
R/W	R	R	R	R	R	R	R	R
Reset value	X	X	X	X	X	X	X	X

Bit7~Bit0      UID&lt;63:56&gt;

## UID8

F5E8H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
UID8	UID71	UID70	UID69	UID68	UID67	UID66	UID65	UID64
R/W	R	R	R	R	R	R	R	R
Reset value	X	X	X	X	X	X	X	X

Bit7~Bit0      UID&lt;71:64&gt;

## UID9

F5E9H	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
UID9	UID79	UID78	UID77	UID76	UID75	UID74	UID73	UID72
R/W	R	R	R	R	R	R	R	R
Reset value	X	X	X	X	X	X	X	X

Bit7~Bit0      UID&lt;79:72&gt;

## UID10 (0xF5EA)

F5EAH	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
UID10	UID87	UID86	UID85	UID84	UID83	UID82	UID81	UID80
R/W	R	R	R	R	R	R	R	R
Reset value	X	X	X	X	X	X	X	X

Bit7~Bit0      UID&lt;87:80&gt;

## UID11

F5EBH	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
UID11	UID95	UID94	UID93	UID92	UID91	UID90	UID89	UID88
R/W	R	R	R	R	R	R	R	R
Reset value	X	X	X	X	X	X	X	X

Bit7~Bit0      UID&lt;95:88&gt;

## 24. User configuration

The system configuration register (CONFIG) is the FLASH option of the MCU's initial conditions, and the program cannot access and operate it. It contains the following:

1. WDT (watchdog working mode selection)
  - ENABLE forces to open WDT
  - SOFTWARE CONTROL (default) WDT working mode is controlled by the WDTRE bit of the WDCON register
2. PROTECT
  - ENABLE FLASH code encryption, the read code is 00H. And it is forbidden to enter debugging mode.
  - DISABLE (default) FLASH code is not encrypted
3. FLASH\_DATA\_PROTECT
  - DISABLE FLASH data area is not encrypted
  - ENABLE (default) FLASH data area is encrypted, after encryption, the value read by the programming emulator is 00H
4. LVR (low voltage reset)
  - 1.8V (default )
  - 2.0V
  - 2.5V
  - 3.5V
5. DEBUG (debug mode)
  - DISABLE (default) debug mode is disabled, DSK and DSDA pins are used as normal IO ports
  - ENABLE debug mode is enabled, DSK and DSDA pins are configured as debug ports, and the pins correspond Other functions are disabled.
6. DEBUGIO\_SEL(DEBUG port selection)
  - DEBUG1 (default) DCSK1 DSDA1 as debug port
  - DEBUG2 DCSK2 DSDA2 as debug port
7. OSC (oscillation mode)
  - HSI (default) 48MHz
  - HSE
  - LSE (32.768KHz)
  - LSI (125KHz) 125KHz
8. HSELSE\_SEL (oscillation port selection)
  - OSCIN1/OSCOU1 AS HSE HSE Port as OSCIN1/OSCOU1
  - OSCIN1/OSCOU1 AS LSE LSE Port as OSCIN1/OSCOU1
  - OSCIN2/OSCOU2 AS HSE HSE Port as OSCIN2/OSCOU2
  - None None
9. SYS\_PRESCALE (system clock prescaler selection)
  - $F_{osc}/1$  (default)
  - $F_{osc}/2$
  - $F_{osc}/4$
  - $F_{osc}/8$
10. HSI\_FS (internal RC oscillator frequency division selection)
  - $F_{Hsi}/1$  48MHz
  - $F_{Hsi}/2$  24MHz
  - $F_{Hsi}/3$  16MHz

- F<sub>Hsi</sub>/6 (default) 8MHz
11. EXT\_RESET (external reset configuration)
- DISABLE (default) external reset disable
  - ENABLE external reset enable
  - ENABLE (OPEN PULLUP) external reset enable and open the reset port internal pull-up Resistor
12. WAKE UP\_WAIT TIME (The default time for wake-up from sleep and wait for oscillator to stabilize is 1.0s)
- 50us
  - 100us
  - 500us
  - 1ms
  - 5ms
  - 10ms
  - 500ms
  - 1.0s ( Default)
13. CPU\_WAITCLOCK (memory wait clock selection)
- 1\*System Clock (1T) (default)
  - 2\*System Clock (2T)
  - 3\*System Clock (3T)
  - 4\*System Clock (4T)
  - 5\*System Clock (5T)
  - 6\*System Clock (6T)
  - 7\*System Clock (7T)
  - 8\*System Clock (8T)
14. WRITE\_PROTECT Program partition protection (protectable intervals, all default intervals are not Protection)
- 0000H-07FFH (protected/unprotected)
  - 0800H-0FFFH (protected/unprotected)
  - 1000H-17FFH (protected/unprotected)
  - 1800H-1FFFH (protected/unprotected)
  - 2000H-27FFH (protected/unprotected)
  - 2800H-2FFFH (protected/unprotected)
  - 3000H-37FFH (protected/unprotected)
  - 3800H-3FFFH (protected/unprotected)
  - 4000H-47FFH (protected/unprotected)
  - 4800H-4FFFH (protected/unprotected)
  - 5000H-57FFH (protected/unprotected)
  - 5800H-5FFFH (protected/unprotected)
  - 6000H-67FFH (protected/unprotected)
  - 6800H-6FFFH (protected/unprotected)
  - 7000H-77FFH (protected/unprotected)
  - 7800H-7FFFH (protected/unprotected)
15. BOOT space selection
- BOOT\_DIS (Default) BOOT area prohibits
  - BOOT\_1K BOOT area space is 1K
  - BOOT\_2K BOOT area space is 2K
  - BOOT\_4K BOOT area space is 4K

Note:

- 1) The machine cycle is related to the memory wait clock selection (CPU\_WAITCLOCK):  
machine cycle= T<sub>sys</sub>/CPU\_WAITCLOCK.
- 2) When the oscillation mode is selected as HSI, the internal RC oscillator frequency is selected as F<sub>Hsi</sub>/1, and the system clock prescaler is selected as F<sub>Osc</sub>/1, and all three conditions are met, if the memory waiting clock is selected as 1\*System Clock ( 1T), the actual memory waiting clock is selected as 2T, and the machine cycle= T<sub>sys</sub>/2.

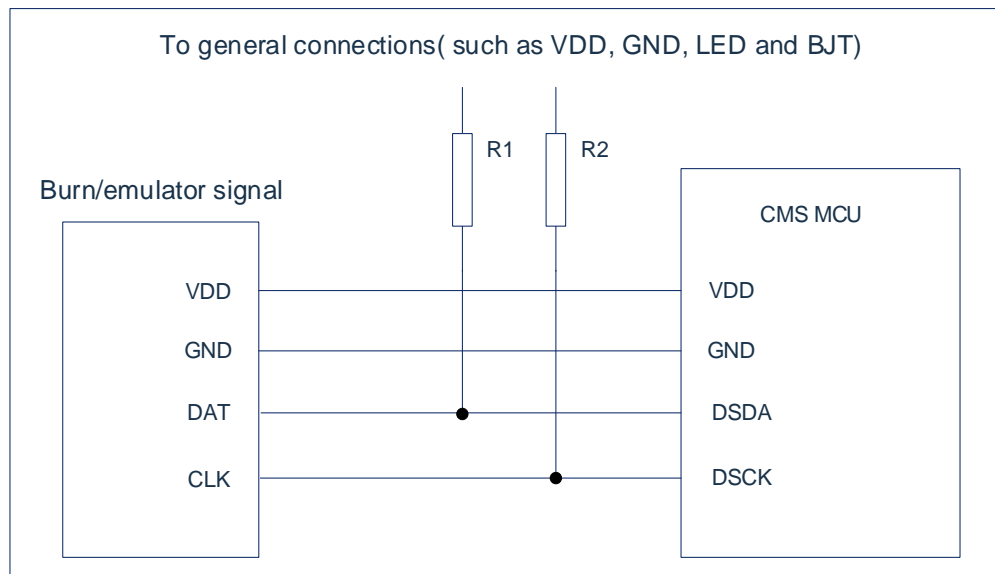
## 25. Online programming and debugging

### 25.1 Online programming mode

The chip can be serially programmed in the final application circuit. Programming can be done simply through the following 4 wires:

- power line,
- ground line,
- data line,
- clock line,

In-line serial programming allows users to use unprogrammed devices to manufacture circuit boards and program the chip only before the product is delivered, so that the latest version of firmware or customized firmware can be programmed into the chip. A typical online serial programming connection method is shown in the figure below:



In the above figure, R1 and R2 are electrical isolation devices, which are often replaced by resistors. The resistance values are as follows:  $R1 \geq 4.7K$ ,  $R2 \geq 4.7K$ .

Note that during programming and debugging, DSDA is forbidden to connect 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 then connect the pull-down resistor after completion.

## 25.3 Online debugging mode

The chip supports 2-wire (DSCK, DSDA) online debugging function. If you use the online debugging function, you need to set DEBUG in the system configuration register to ENABLE. When using debug mode, you need to pay attention to the following points:

- ◆ In debugging state, the DSCK and DSDA ports are dedicated debugging ports, and their GPIO and multiplexing functions cannot be realized.
- ◆ When entering the sleep mode/idle mode in the debug state, the system power supply and oscillator will not stop working, and the sleep wake-up function can be simulated in this state. If you need to pay attention to power consumption, it is recommended to turn off the debugging function before testing the actual sleep current of the chip.
- ◆ Pause in the debug state, other functional peripherals continue to run, WDT, Timer0/1/2/3/4 counters will stop. But if Timer1/4 is used as the baud rate generator of UART0/1, Timer1/4 will continue to run in the pause state. Peripherals that continue to run in the paused state may generate interrupts, so be careful when debugging.



## 26. Instructions

There are 5 types of assembly instructions: arithmetic operations, logic operations, data transfer operations, Boolean operations, and program branch instructions, all of which are compatible with the standard 8051.

### 26.1 Symbol description

Symbol	description
Rn	working register R0-R7
Direct	internal data memory RAM unit address (00H-FFH) or special function register SFR address
@Ri	indirect addressing register (@R0 or @R1)
#data	8-bit binary constant
#data16	16-bit binary constant in the instruction
Bit	address in internal data memory RAM or special function register SFR
Addr16	16-bit address, address range 0-64KB address space
Addr11	11-bit address, address range 0-2KB address space
Rel	relative address
A	accumulator

## 26.2 instruction list

mnemonic		description
<b>operation type</b>		
ADD	A,Rn	Accumulator plus register
ADD	A,direct	Accumulator plus direct addressing unit
ADD	A,@Ri	Accumulator plus indirect addressing RAM
ADD	A,#data	Accumulator plus immediate
ADDC	A,Rn	Accumulator plus register and carry flag
ADDC	A,direct	Accumulator plus direct addressing unit and carry flag
ADDC	A,@Ri	Accumulator plus indirect addressing RAM and carry flag
ADDC	A,#data	Accumulator plus immediate value and carry flag
SUBB	A,Rn	Accumulator minus register and carry flag
SUBB	A,direct	Accumulator minus direct addressing unit and carry flag
SUBB	A,@Ri	Accumulator minus indirect addressing RAM and carry flag
SUBB	A,#data	Accumulator minus immediate value and carry flag
INC	A	Accumulator plus 1
INC	Rn	Register plus 1
INC	direct	Direct addressing unit plus 1
INC	@Ri	Indirect addressing RAM plus 1
INC	DPTR	Data pointer plus 1
DEC	A	Accumulator minus 1
DEC	Rn	Register minus 1
DEC	direct	Direct addressing unit minus 1
DEC	@Ri	Indirect addressing RAM minus 1
MUL	A,B	Accumulator multiply register B
DIV	A,B	Accumulator divided by register B
DA	A	Decimal adjustment
<b>logic operation class</b>		
ANL	A,Rn	Accumulators and registers
ANL	A,direct	Accumulator and Direct Addressing Unit
ANL	A,@Ri	Accumulator and indirect addressing RAM
ANL	A,#data	Accumulator and immediate data
ANL	direct,A	Direct addressing unit and accumulator
ANL	direct,#data	Direct addressing unit and immediate data
ORL	A,Rn	Accumulator or register
ORL	A, direct	Accumulator or direct addressing unit
ORL	A,@Ri	Accumulator or indirect addressing RAM
ORL	A, #data	Accumulator or immediate
ORL	direct,A	Direct addressing unit or accumulator
ORL	direct,#data	Direct addressing unit or immediate data
XRL	A,Rn	Accumulator XOR Register
XRL	A,direct	Accumulator XOR Direct Addressing Unit
XRL	A,@Ri	Accumulator XOR indirect addressing RAM
XRL	A,#data	Accumulator XOR immediate
XRL	direct,A	Direct addressing unit exclusive OR accumulator
XRL	direct,#data	XOR immediate data of direct addressing unit
CLR	A	Accumulator is cleared to 0

CPL	A	Inverted accumulator
RL	A	Accumulator rotate left
RLC	A	Accumulator with carry flag to rotate left
RR	A	Accumulator rotate right
RRC	A	Accumulator with carry flag right cyclic shift
SWAP	A	Swap the high 4 bits and low 4 bits of the accumulator
<b>data transmission type</b>		
MOV	A,Rn	Register transfer to accumulator
MOV	A,direct	Direct addressing unit transfer to accumulator
MOV	A,@Ri	Indirect addressing RAM to accumulator
MOV	A,#data	Immediately send accumulator
MOV	Rn,A	Accumulator send register
MOV	Rn,direct	Direct addressing unit to register
MOV	Rn,#data	Immediate data transfer register
MOV	direct,A	The accumulator sends the direct addressing unit
MOV	direct,Rn	Direct addressing unit
MOV	direct1,direct2	Direct address unit transfer to direct address unit
MOV	direct,@Ri	Indirect addressing RAM to direct addressing unit
MOV	direct,#data	Immediate data direct addressing unit
MOV	@Ri,A	The accumulator sends indirect addressing RAM
MOV	@Ri,direct	Direct addressing unit to indirect addressing RAM
MOV	@Ri,#data	Immediate data transmission and indirect addressing RAM
MOV	DPTR,#data16	16-bit immediate data to send data pointer
MOVC	A,@A+DPTR	The look-up table data is sent to the accumulator (DPTR is the base address)
MOVC	A,@A+PC	Look-up table data sent to accumulator (PC is the base address)
MOVX	A,@Ri	External RAM unit to accumulator (8-bit address)
MOVX	A,@DPTR	External RAM unit to accumulator (16-bit address)
MOVX	@Ri,A	The accumulator sends the external RAM unit (8-bit address)
MOVX	@DPTR,A	The accumulator sends the external RAM unit (16-bit address)
PUSH	direct	The direct addressing unit is pushed onto the top of the stack
POP	direct	Direct addressing unit popped from the top of the stack
XCH	A,Rn	Accumulator and register swap
XCH	A,direct	Accumulator and direct addressing unit RAM swap
XCH	A,@Ri	Accumulator and indirect addressing unit RAM swap
XCHD	A,@Ri	The accumulator and the indirect addressing unit RAM swap the lower 4 bits
<b>Boolean operation type</b>		
CLR	C	C clear
CLR	bit	Direct addressing bit is cleared
SETB	C	C set
SETB	bit	Direct addressing bit
CPL	C	C negate
CPL	bit	Direct addressing bit inversion
ANL	C,bit	C logic and direct addressing bit
ANL	C,/bit	The inverse of C logic and direct addressing bits
ORL	C,bit	C logic or direct addressing bit
ORL	C,/bit	The inverse of C logic or direct addressing bit
MOV	C,bit	Direct addressing bit to C

MOV	bit,C	C send direct addressing bit
<b>Program jump class</b>		
ACALL	addr11	Absolute call within 2K address range
LCALL	addr16	Long call in 64K address range
RET		Subroutine return
RETI		Interrupt return
AJMP	addr11	Absolute transfer within 2K address range
LJMP	addr16	Long transfer within 64K address range
SJMP	rel	Relatively short transfer
JMP	@A+DPTR	Relatively long transfer
JZ	rel	Accumulator is 0 transfer
JNZ	rel	The accumulator is not 0 transfer
JC	rel	C is 1 transfer
JNC	rel	C is 0 transfer
JB	bit,rel	Direct addressing bit is 1 transfer
JNB	bit,rel	Direct addressing bit is 0 transfer
JBC	bit,rel	The direct addressing bit is transferred by 1, and the bit is cleared
CJNE	A,direct,rel	Accumulator and direct addressing unit unequal transfer
CJNE	A,#data,rel	Accumulator and immediate data unequal transfer
CJNE	Rn,#data,rel	Unequal transfer between register and immediate
CJNE	@Ri,#data,rel	Indirect addressing unit RAM and immediate data unequal transfer
DJNZ	Rn,rel	Register minus 1 is not transferred to 0
DJNZ	direct,rel	Direct addressing unit minus 1 is not transferred to 0
NOP		Null instruction
<b>read-modify-write instruction (Read-Modify-Write)</b>		
ANL		Logic (ANL direct, A and ANL direct, #data)
ORL		Logical OR (ORL direct, A and ORL direct, #data)
XRL		Logical XOR (XRL direct, A and XRL direct, #data)
JBC		The direct address bit is transferred by 1, and the bit is cleared (JBC bit, rel)
CPL		Invert (CPL bit)
INC		Plus 1 (INC direct)
DEC		Subtract 1. (DEC direct)
DJNZ		Subtract 1 and not transfer to 0 (DJNZ direct, rel)
MOV	bit,C	C send direct addressing bit
CLR	bit	Direct addressing bit is cleared
SETB	bit	Direct addressing bit

## 27. Version revision description

version number	time	revision content
V1.00	May 2020	initial version
V1.01	Jun 2021	Revise P57 Pin function
V1.0.3	Feb 2023	<ol style="list-style-type: none"><li>1) Complete IO port information</li><li>2) Delete ADC clock frequency example, adjust I<sup>2</sup>C description, delete FLASH operation time related description, adjust port reuse function description, add BUZZER precautions, change some register bit description, optimize some text expression</li><li>3) 3.1 Power-on reset: add the relation table between reset flag and reset signal</li><li>4) 21.2.4 Clock conversion: example of adding ADC clock</li></ol>