DATA SHEET

SUNPLUS

SPMC65P2404A/2408A

Micro-Controllers (OTP) With ADC

Nov. 11, 2005

Version 1.2

Table of Contents

PAGE

1. GENERAL DESCRIPTION

SPMC65P2404A and SPMC65P2408A are the members of 65X series and OTP (One Time Program) solutions. SPMC65P2408A is a superset of SPMC65P2404A. These two devices share similar cores and peripherals with the following exceptions. For example, SPMC65P2404A does not have UART interfaces, Port C Bit[5:4], Port D Bit[4:3], and compare function on timer 2. In addition, ROM and RAM sizes on the SPMC65P2404A are less than on SPMC65P2408A. For detailed memory information, please refer to the following section. Major application fields are small home appliance, industry controller, and battery charger. Main features of these two devices are depicted in the next section.

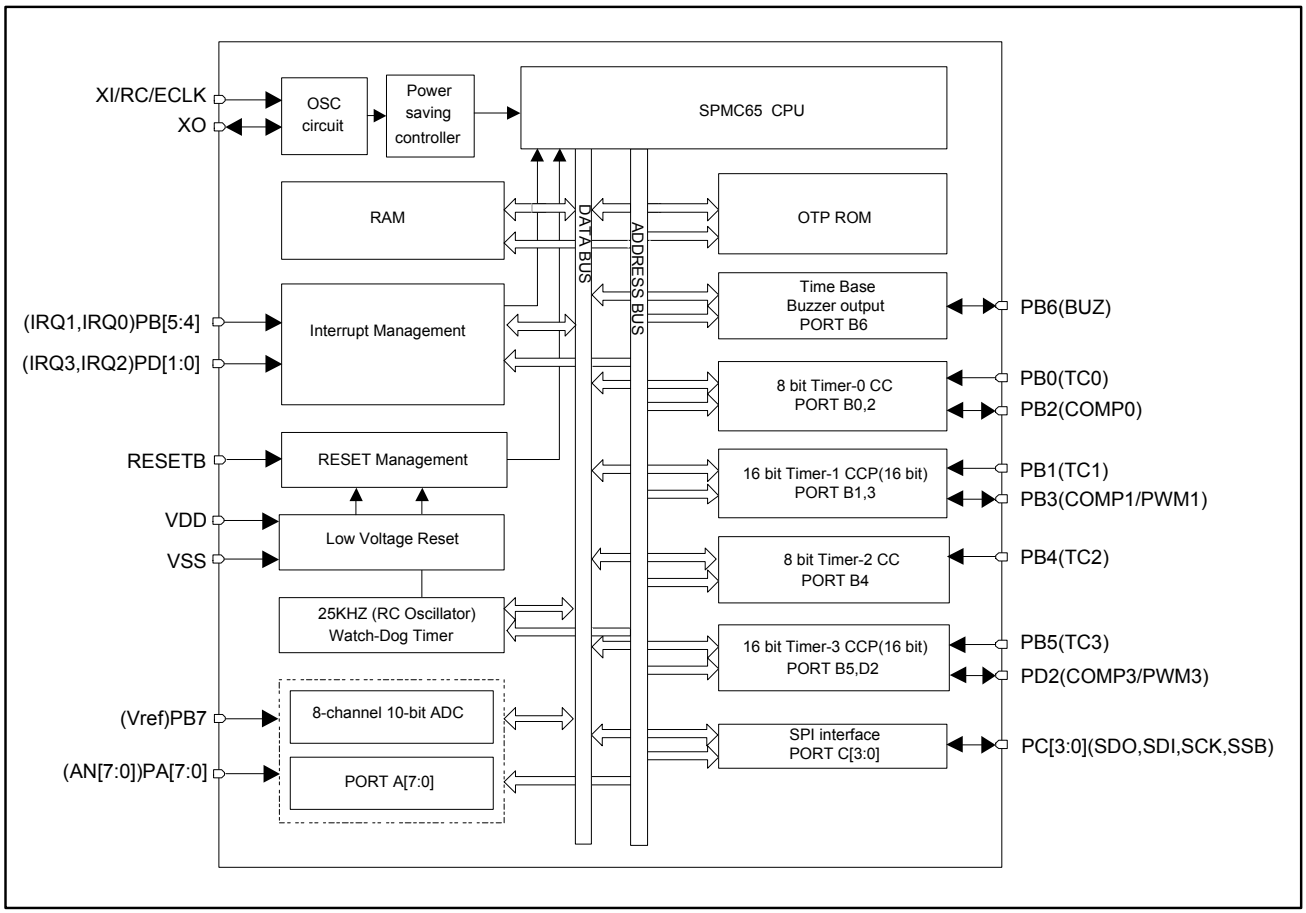
2. FEATURES

- **SPMC65 CPU**
  - 182 instructions
  - 11 addressing modes
  - Up to 8MHz clock operation
  - Supports bit operation instruction (Set, Clear, Inverse, Test)
- **Memories**
  - 4K bytes program memory (OTP) with security protection (SPMC65P2404A)
  - 8K bytes program memory (OTP) with security protection (SPMC65P2408A)
  - 192 bytes RAM including stack area (for 2404A)
  - 256 bytes RAM including stack area (for 2408A)
- **I/O Ports**
  - 23 multifunction bi-directional I/Os (for 2404A)
  - 27 multifunction bi-directional I/Os (for 2408A)
  - All I/Os are Schmitt Trigger inputs
  - Each incorporate with pull-up resistor, pull-down resistor or floating input, depending on programmer's settings on the corresponding registers
  - I/O ports with LED driving capability
  - 2 I/O ports with 20mA current sink
- **Interrupt Management**
  - Interrupt option: NMI or IRQ for external interrupts.
  - 4 external interrupts (One of them can be programmed as NMI)
  - 13 internal interrupts (for 2408A)
  - 12 Internal interrupts (for 2404A)
- **Reset Management**
  - Enhanced reset system
  - Power On Reset (POR)
  - Low Voltage Reset (LVR)
  - Watchdog Reset (WDR)
  - External Reset (ERST)
  - Illegal Address Reset (IAR)
- **Clock Management**
  - Three clock sources: RC-oscillation, crystal input and external clock input
  - Clock output capability for RC-oscillation
- **Power Management**
  - 2 power saving modes: STOP, HALT mode
- **2 Analog Peripheral**
  - 8-channel of 10-bit resolution A/D converter (100KHz)
  - LVR: Low Voltage Reset (2.5V/4V)
- **Two channels of 8-bit Timers (Timer0, Timer2)**
  - Timers, event counter mode
  - Capture with 8-bit width measurement
  - 8-bit compare mode
- **Two channels of 16-bit Timers (Timer1, Timer3)**
  - Timers, Event counter mode
  - Capture (8-bit with width/cycle measurement or 16-bit with width measurement)
  - 16-bit compare output
  - 12-bit PWM output
- **Time Base Interval Timer**
  - Frequency: 1KHz to 62.5kHz @Fsys=8MHz (for 2404A)
  - Frequency: 1Hz to 62.5kHz @Fsys=8MHz (for 2408A)
  - 7 stages pre-scale option (for 2404A)
  - 15 stages pre-scale option (for 2408A)
- **Buzzer output**
  - Frequency: 1kHz to 2MHz @Fsys=8MHz
  - 12 stages pre-scale option
- **Configurable watchdog timer**
  - Frequency: 1.5Hz to 195Hz @Fsys=8MHz
- **Serial Bus interface**
  - SPI bus: up to 2MHz @Fsys=8MHz
- **Universal asynchronous receiver / transmitter interface (2408A only)**
  - Baud rate: up to 38400 bps

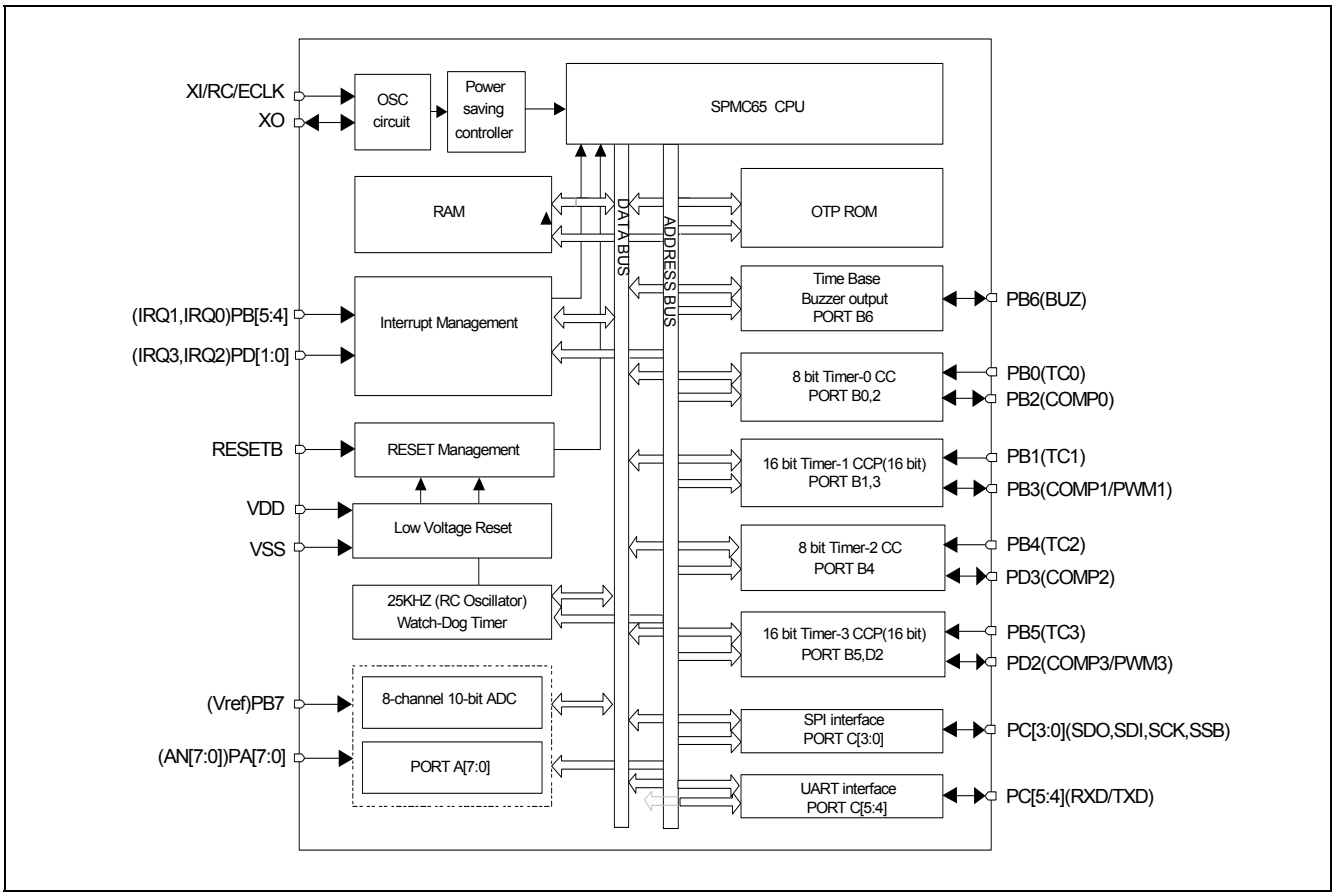
Device	OTP ROM(Byte)	RAM (Byte)	Max I/O	10 bit A/D(ch)	CCP module			SPI	UART	Package Type
					Compare	Capture	PWM			
SPMC65P2404A	4K	192	23	8	3	4	2	Y	N	PDIP28, SOP28
	4K	192	15	4	3	4	2	N	N	PDIP20, SOP20
SPMC65P2408A	8K	256	27	8	4	4	2	Y	Y	PDIP32, SOP32
	8K	256	23	8	3	4	2	Y	N	PDIP28, SOP28

3. BLOCK DIAGRAM

3.1. Block Diagram of SPMC65P2404A



3.2. Block Diagram of SPMC65P2408A



4. SIGNAL DESCRIPTIONS

4.1. Pin Description

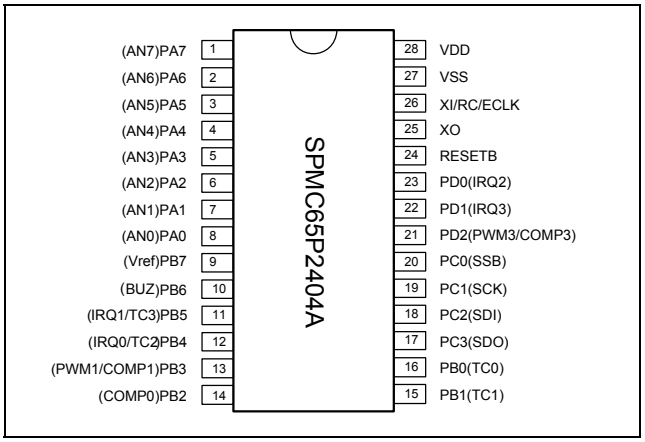
Type：I = Input, O = Output, S = Supply

Pin Name	PIN Number				Type	Main Function	Alternate Function
	32PIN (2408)	28PIN (2408)	28PIN (2404)	20PIN (2404)			
VDD	32	28	28	20	S	Main power supply	
VSS	31	27	27	19	S	Ground	
XI/RC/ECLK	30	26	26	18	I	*Crystal input or RC-oscillation input or external clock input.* An external resistive pull-up is used to connect with internal OSC circuitry for generating the internal clock and the related time base in RC-Oscillation mode. It will be connected with external crystal for a crystal oscillation circuitry in crystal mode.	
XO	29	25	25	17	O	*Crystal output.* It will be connected with external crystal for a crystal oscillation circuitry in crystal mode.	
PA7/AN7	1	1	1	--	I/O	Port A7	ADC analog input
PA6/AN6	2	2	2	--	I/O	Port A6	ADC analog input
PA5/AN5	3	3	3	--	I/O	Port A5	ADC analog input
PA4/AN4	4	4	4	--	I/O	Port A4	ADC analog input
PA3/AN3	5	5	5	1	I/O	Port A3	ADC analog input
PA2/AN2	6	6	6	2	I/O	Port A2	ADC analog input
PA1/AN1	7	7	7	3	I/O	Port A1	ADC analog input
PA0/AN0	8	8	8	4	I/O	Port A0	ADC analog input
PB7/Vref	9	9	9	5	I/O	Port B7	
PB6/BUZ	10	10	10	6	I/O	Port B6	Buzzer output
PB5/IRQ1/TC3	11	11	11	7	I/O	Port B5	External interrupt 1 input / Capture3 event input to Timer3 / external event input using Timer3
PB4/IRQ0/TC2	12	12	12	8	I/O	Port B4	External interrupt 0 input / Capture2 event input to Timer2 / external event input using Timer2
PB3/COMP1 /PWM1	13	13	13	9	I/O	Port B3	Timer1 compare output / PWM output
PB2/COMP0	14	14	14	10	I/O	Port B2	Timer0 compare output
PB1/TC1	15	15	15	11	I/O	Port B1	Capture1 event input to Timer1 / external event input using Timer1 as the counter.

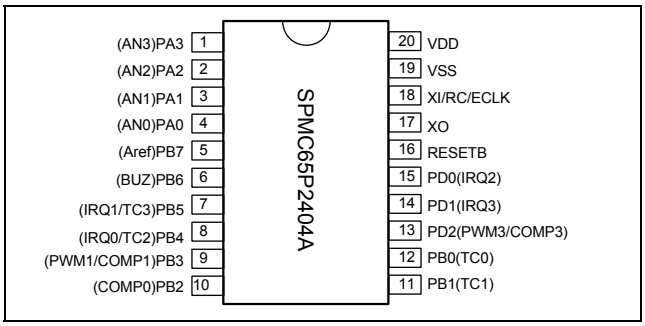
Pin Name	PIN Number				Type	Main Function	Alternate Function
	32PIN (2408)	28PIN (2408)	28PIN (2404)	20PIN (2404)			
PB0/ TC0	16	16	16	12	I/O	Port B0	Capture0 event input to Timer0 / external event input using Timer0 as the counter.
PC5/RXD	17	--	--	--	I/O	Port C5	UART receive signal (2408A only)
PC4/TXD	18	--	--	--	I/O	Port C4	UART transmit signal (2408A only)
PC3/SDO	19	17	17	--	I/O	Port C3	SPI data output
PC2/SDI	20	18	18	--	I/O	Port C2	SPI data input
PC1/SCK	21	19	19	--	I/O	Port C1	SPI clock output / clock input
PC0/SSB	22	20	20	--	I/O	Port C0	SPI chip select
PD4	23	--	--	--	I/O	Port D4	(2408A only)
PD3/COMP2	24	--	--	--	I/O	Port D3	Timer2 compare output (2408A only)
PD2/COMP3 /PWM3	25	21	21	13	I/O	Port D2	Timer3 compare output / Timer3 PWM output
PD1/IRQ3	26	22	22	14	I/O	Port D1	External interrupt 3 input
PD0/IRQ2	27	23	23	15	I/O	Port D0	External interrupt 2 input
RESETB	28	24	24	16	I	Reset pin	

4.2. PIN Assignment (Top View)

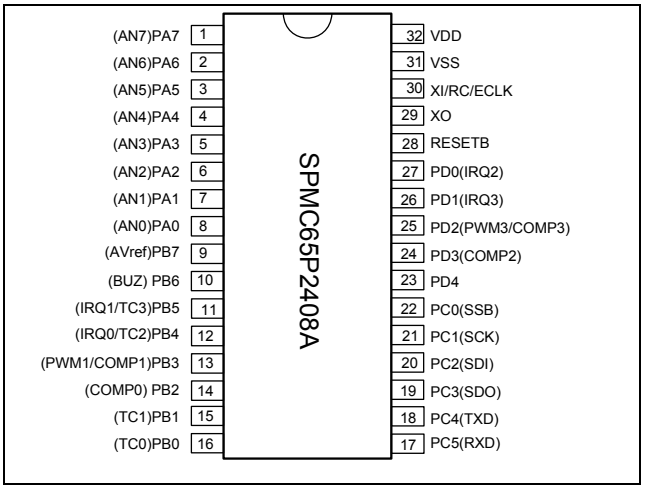
4.2.1. 28 PIN package (SPMC65P2404A)



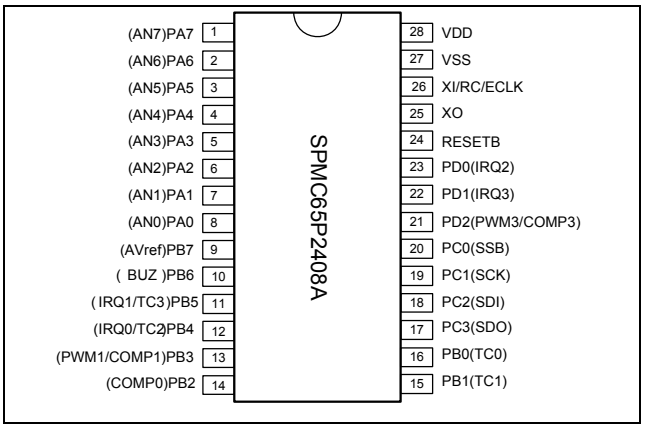
4.2.2. 20 PIN package (SPMC65P2404A)



4.2.3. 32 PIN package (SPMC65P2408A)



4.2.4. 28 PIN package (SPMC65P2408A)



5. FUNCTIONAL DESCRIPTIONS

5.1. Central Processing Unit

5.1.1. CPU Introduction

The SPMC65P2404A/2408A is a SPMC65 high performance processor equipped with 6 internal registers: accumulator, program counter, X register, Y register, stack pointer, and processor status register. This CPU is a fully static CMOS design. The oscillation frequency could be varied up to 8.0MHz depending on the application.

5.1.2. CPU register

The SPMC65 CPU has six registers that are the Program Counter (PC), an Accumulator (A), two index registers (X, Y), the Stack Pointer (SP), and the Status register (P). The program counter consists of 16-bit register.

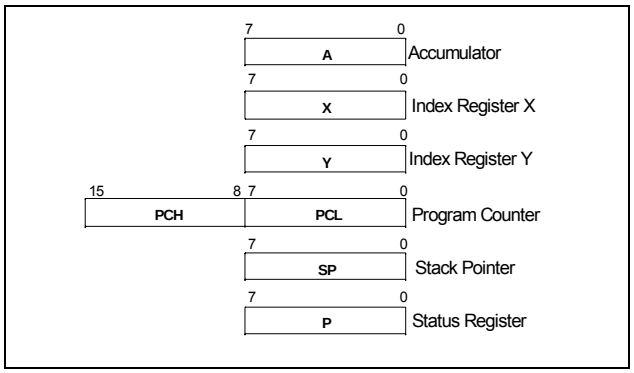


Figure 5.1-1 System registers

X, Y register

In address mode, X and Y registers can be used as index registers or buffer registers. These register contents are added to the specified address, which becomes the actual address. Some operations such as increment, decrement, comparison and data transfer function can be used in X and Y registers.

Accumulator

The Accumulation is the 8-bit general-purpose register, which can be operated with transfer, temporary saving, condition judgment, etc.

Stack pointer

SPMC65P2404A/2408A has an 8-bit-wide register indicating the location in the stack to be accessed (push or pop) when a subroutine call or interrupt occurs.

When subroutine call is executed or an interrupt occurrence is accepted, the value of stack point is updated automatically.

However, if the value of stack point is used in excess of stack area permitted by the data memory allocating configuration, the Illegal address reset would be occurred.

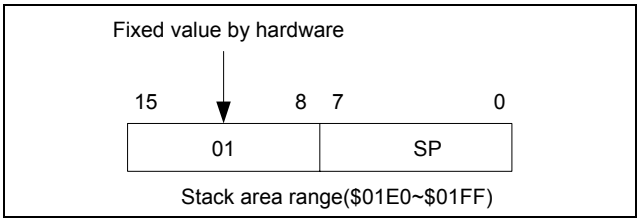


Figure 5.1-2 Stack point register

[Example] 5.1.1 Initialized stack point value

```
ldx   #C_STACK_BOTTOM    ; Initial stack pointer at $FF
txs                      ;Transfer to stack point
```

Program counter (PC)

The program counter is a 16-bit wide register. It consists of two 8-bit registers which registers are PCH and PCL. This register indicates the address of next instruction to be executed. In Reset state, the content of program counter is stored with $FFFC.

Status register (P)

The 8-bit status register contains the interrupt mask and 6 flags representative of the result of the instruction just executed. This register can also be handled by the PHP and PLP instructions. These bits can be individually controlled by specific instructions. The detailed description is shown in following description.

Note: Not all instructions affect status register. A detailed instruction description will be discussed in SPMC65 CPU instruction manual.

- **Negative flag bit**

  This flag indicated the bit7 status of the result of a data or arithmetic operation. Programmer can use this bit to do some operations, e.g. branch condition or bit operation.

- **Overflow flag bit**

  This flag indicates whether the overflow has occurred in arithmetic operation. When the result of an addition or subtraction is over +127 or less than –128, this overflow bit is set to '1'.

- **Decimal mode flag**

  This flag indicates what mode is operated by arithmetic operation. SPMC65P2404A/2408A has two operation modes,

binary mode and decimal mode for arithmetic operation. Programmer can use the instruction to alternate them.

❑ **Interrupt disable flag**

This bit can enable or disable all interrupt except NMI interrupt source. If this bit is set to '1', CPU will ignore interrupt signal. On the contrary, if this bit is set to '0', CPU will accept interrupt signal.

❑ **Zero flag**

This flag indicated the result of a data or arithmetic operation. If the result is equal to zero, the zero flag is set to '1'. Contrary, this bit is set to '0' by other values.

❑ **Carry flag**

This bit is set to '1 if the result of addition operation generates a carry, or if the result of subtraction doesn't generate a borrowing. In addition, some shift instructions or rotate instructions also change this bit.

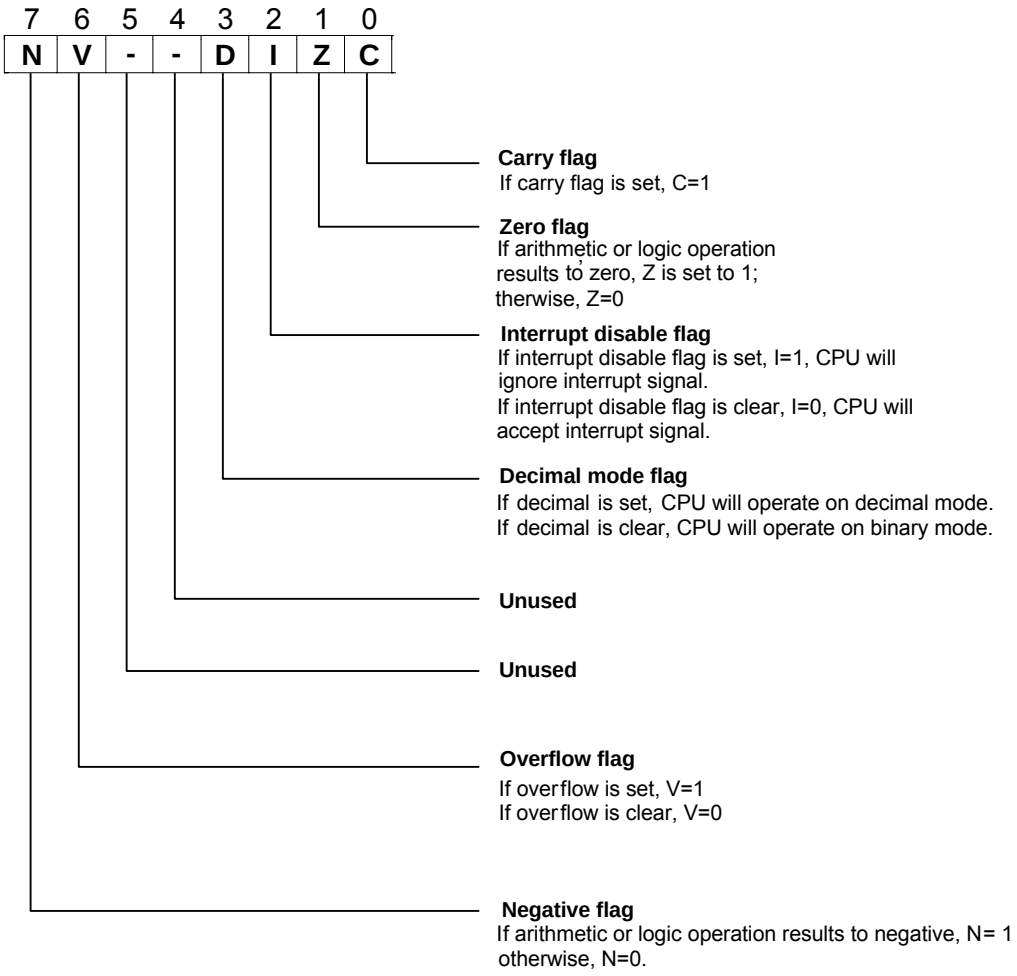


Figure 5.1-3 Status register

5.2. Memory Organization

5.2.1. Introduction

The SPMC65P2404A/2408A has separated address spaces for program memory and data memory. Program memory can only be read only. Writing program memory when chip is operating will trigger a reset signal. It contains up to 4K/8k bytes of program memory. Data memory that contains 192/256 bytes of RAM including stack area can be read and written.

5.2.2. Memory Space

Memory address allocations on the SPMC65P2404A and SPMC65P2408A are divided into several parts. Figure 5.2-2 shows SPMC65P2404A memory map and figure 5.2-3 shows SPMC65P2408A memory map. Different memory address allocations between SPMC65P2404A and SPMC65P2408A are the size of RAM and OTP ROM. In this document, we only take SPMC65P2404A as an example to explain it in details since that the usage of each allocation on the SPMC65P2404A is the same as SPMC65P2408A.

The first 96 addresses are allocated for hardware control register, including function control registers and I/O control registers, which allow programmer to use the first page instruction in setting this register and help for program size reduction. The data memory areas (RAM) are located in $0060 ~ $00FF and $01E0~$01FF. The 32 bytes of data memory between $01E0~$01FF are also defined as stack for exception. The stack pointer is started from $01FF to $01E0 with downward. Once the stack is over the $01E0, a CPU reset will be generated.

The 4 bytes register located in $7FE0 ~ $7FE3 are used to define the device configuration registers. Programmer can use these four bytes to configure the condition of this chip. The detailed setting of the device configuration registers is show in chapter 5.13.

SPMC65P2404A also contains the user information register located in $7FF0 ~ $7FFF which can be programmed by programmer for series number or version control setting.

SPMC65P2404A supports 4K bytes of OTP ROM. The address for OTP ROM is located in $F000 ~ $FFFF (see figure 6.3). SPMC65P2404A also supports a security bit for programmer. When the security bit is set to ‘0’, the data in OTP ROM cannot be read by programmer. In contrast, if security bit is set to ‘1’, the 4K byres of OTP ROM will be readable. Application option and user information will be still readable even if the security bit is set.

The address of NMI, RESET and IRQ exception vectors are located from $FFFA to $FFFF. The exception vectors should be specified in the program to have proper operation.

To prevent the illegal accesses on undefined memory, there is a qualification block to limit the accesses. The illegal accesses will generate the CPU reset (IAR) to restart the program.

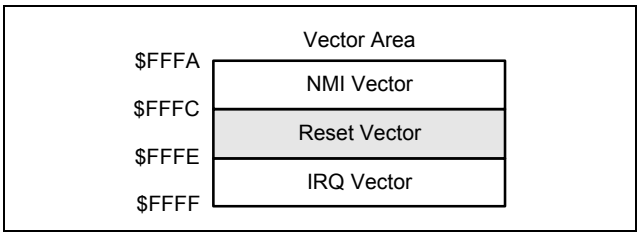


Figure 5.2-1 Interrupt vector area

[Example] 5.2.1 Interrupt vector table in software

```
VECTOR:     .SECTION
            DW      V_NMI
            DW      V_Reset
            DW      V_IRQ
```

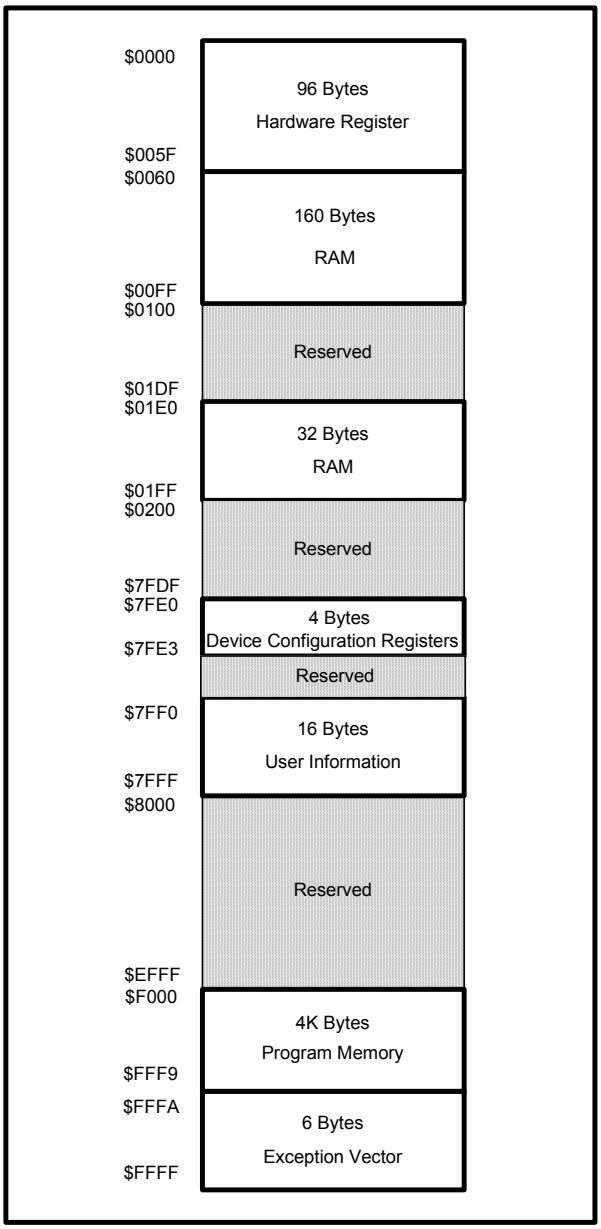


Figure 5.2-2 Memory map for SPMC65P2404A

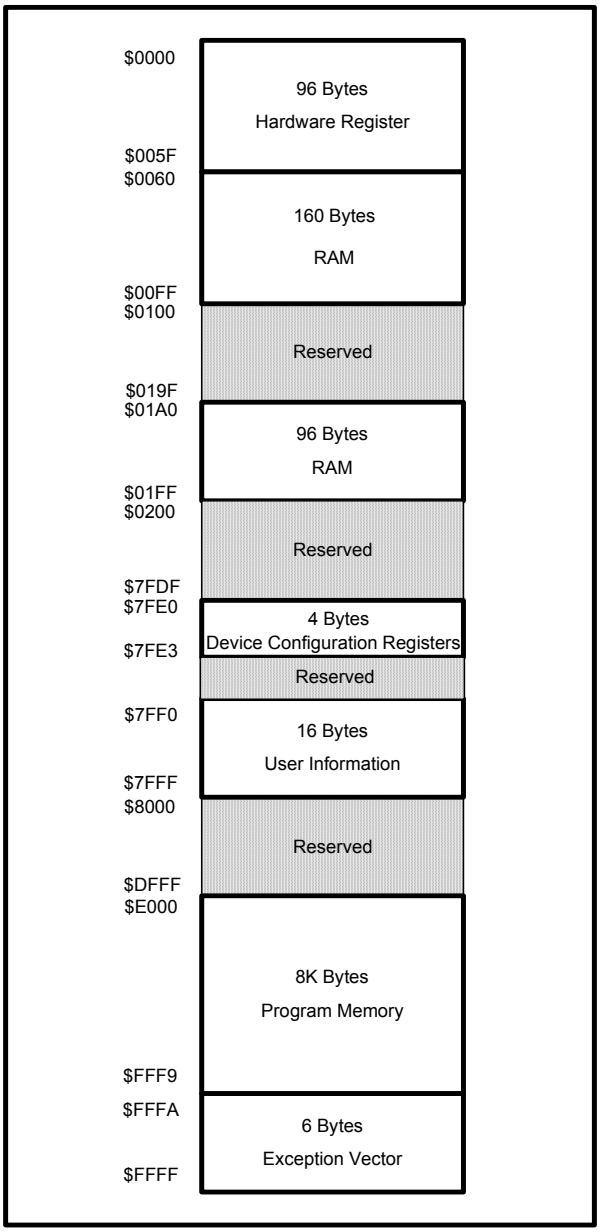


Figure 5.2-3 Memory map for SPMC65P2408A

5.2.3. Hardware Control Registers

SPMC65P2404A/2408A device have up to twenty control registers. All of the control registers are used by MCU and peripheral function block for controlling the desired operation of the device.

Some of the control registers contain control and statue bits for peripheral module such as Timer unit, A/D Converter, SPI, etc. Note that the reserved address can't be implemented on the chip. Some of bits in control register are read only, so write to those bits don't have any effect on corresponding bits. The following table shows the summary of the control registers. The detailed information of each control registers are explained in each peripheral section.

5.2.3.1. Control Register List

1). $0000~000F: I/O port and Interrupt

Address	Function	ERST value	R/W	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
$0000	Port A IO	00h	R	Port A I/O pins							
	(P_IOA_Data)		W	Port A Data Buffer ($0059)							
$0001	Port B IO	00h	R	Port B I/O pins							
	(P_IOB_ Data)		W	Port B Data Buffer ($005A)							
$0002	Port C IO	00h	R	0		Port C I/O pins					
	(P_IOC_ Data)		W	-		Port C Data Buffer ($005B)					
$0003	Port D IO	00h	R	0			Port D I/O pins				
	(P_IOD_Data)		W	-			Port D Data Buffer ($005C)				
$0004	Port A Direction	00h	R	Port A Data Direction register. 0=IN, 1=OUT.							
	(P_IOA_Dir)		W	Port A Data Direction register. 0=IN, 1=OUT.							
$0005	Port B Direction	00h	R	Port B Data Direction register. 0=IN, 1=OUT.							
	(P_IOB_ Dir)		W	Port B Data Direction register. 0=IN, 1=OUT.							
$0006	Port C Direction	00h	R	0		Port C Data Direction register. 0=IN, 1=OUT.					
	(P_IOC_ Dir)		W	-		Port C Data Direction register. 0=IN, 1=OUT.					
$0007	Port D Direction	00h	R	0			Port D Data Direction register. 0=IN, 1=OUT.				
	(P_IOD_ Dir)		W	-			Port D Data Direction register. 0=IN, 1=OUT.				
$0008	Port A Attribute	00h	R	Port A Attribute register							
	(P_IOA_Attrib)		W	Port A Attribute register							
$0009	Port B Attribute	00h	R	Port B Attribute register							
	(P_IOB_Attrib)		W	Port B Attribute register							
$000A	Port C Attribute	00h	R	0		Port C Attribute register					
	(P_IOC_Attrib)		W	-		Port C Attribute register					
$000B	Port D Attribute	00h	R	0			Port D Attribute register				
	(P_IOD_Attrib)		W	-			Port D Attribute register				
$000C	Interrupt Flag 0	00h	R	ADIF	WDIF	0	0	IRQ3IF	IRQ2IF	IRQ1IF/ CAP3IF	IRQ0IF/ CAP2IF
	(P_INT_Flag0)		W	Write ‘1’ to clear corresponding interrupt flag(s)							
$000D	Interrupt Ctrl 0	00h	R	ADIE	WDIE	0	0	IRQ3IE	IRQ2IE	IRQ1IE/ CAP3IE	IRQ0IE/ CAP2IE
	(P_INT_Ctrl0)		W	ADIE	WDIE	-	-	IRQ3IE	IRQ2IE	IRQ1IE/ CAP3IE	IRQ0IE/ CAP2IE
$000E	Interrupt Flag 1	00h	R	CAP1IF	CAP0IF	0	0	T3OIF	T2OIF	T1OIF	T0OIF
	(P_INT_Flag1)		W	Write ‘1’ to clear corresponding interrupt flag(s)							
$000F	Interrupt Ctrl 1	00h	R	CAP1IE	CAP0IE	0	0	T3OIE	T2OIE	T1OIE	T0OIE
	(P_INT_Ctrl1)		W	CAP1IE	CAP0IE	-	-	T3OIE	T2OIE	T1OIE	T0OIE

Note: IRQ0and IRQ1 status/control bits are shared with CAP2and CAP3 status/control bits

Note: Port C bit[5:4] and Port D bit[4:3] are available on the SPMC65P2408A only, and they are illegal in the SPMC65P2404A.

Note: The reset values of Port A/B/C/D attribute depend on IOINIT setting ($7FE2.0)

2). $0010~$001E：Timer/PWM Configuration & Data

Address	Function	ERST value	R/W	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
$0010	Watchdog clear (P_WDT_Clr)	00h	R	0	0	0	0	0	0	0	0
			W	Write #$55h to clear Watchdog timer (C_WDT_Clr = #$55)							
$0011	Timer0-1 Ctrl0 (P_TMR0_1_Ctrl0)	00h	R	0	Timer1 Function Selection			0	-	Timer0 Function Selection	
			W	-	Timer1 Function Selection			-	-	Timer0 Function Selection	
$0012	Timer0-1 Ctrl1 (P_TMR0_1_Ctrl1)	00h	R	0	Timer1 Pre-scale Selection			0	Timer0 Pre-scale Selection		
			W	-	Timer1 Pre-Scale Selection			-	Timer0 Pre-scale Selection		
$0013	Timer0 Counter (P_TMR0_Count)	00h	R	Timer0 8-bit Counter Register							
	Timer0 Preload (P_TMR0_Preload)		W	Timer0 8-bit Preload Register							
	Compare0 (P_TMR0_Comp)	00h	R	Timer0 8-bit Counter Register							
			W	Timer0 8-bit Compare Value							
	Capture0 (P_TMR0_Cap)	00h	R	Timer0 8-bit Capture Width Value							
			W	Timer0 8-bit Preload Register							
$0015	Timer1 Counter (P_TMR1_Count)	00h	R	Timer1 8-bit or 16-bit Low Byte Counter Register							
	Timer1 Preload (P_TMR1_Preload)		W	Timer1 8-bit or 16-bit Low Byte Preload Register							
	Compare1 (P_TMR1_Comp)	00h	R	Timer1 8-bit or 16-bit Low Byte Counter Register							
			W	Timer1 8-bit or 16-bit Compare Low Byte Value							
	Capture1 (P_TMR1_Cap)	00h	R	Timer1 8-bit or 16-bit Capture Low Byte Width Value							
			W	Timer1 8-bit or 16-bit Low Byte Preload Register							
	PWM1 Period (P_TMR1_PWMPeriod)	00h	R	Timer1 12-bit PWM Period Low Bye Value							
			W	Timer1 12-bit PWM Period Low Bye Value							
$0016	Timer1 Counter High (P_TMR1_CountHi)	00h	R	Timer1 16-bit High Byte Counter Register							
	Timer1 Preload High (P_TMR1_PreloadHi)		W	Timer1 16-bit High Byte Preload Register							
	Compare1 High Byte (P_TMR1_CompHi)	00h	R	Timer1 16-bit High Byte Counter Register							
			W	Timer1 16-bit Compare High Byte Value							
	Capture1 High Byte (P_TMR1_CapHi)	00h	R	Timer1 16-bit Capture High Byte Width Value							
			W	Timer1 16-bit High Byte Preload Register							
	Capture1 Cycle (P_TMR1_CapCycle8)	00h	R	Timer1 8-bit Capture Cycle Value							
			W	Timer1 16-bit High Byte Preload Register							
	PWM1 Duty Period (P_TMR1_DutyPeriod)	00h	R	Timer1 12-bit PWM Duty High Byte Value				Timer1 12-bit PWM Period High Byte Value			
			W	Timer1 12-bit PWM Duty High Byte Value				Timer1 12-bit PWM Period High Byte Value			
$0017	PWM1 Duty (P_TMR1_PWMDuty)	00h	R	Timer1 12-bit PWM Duty Low Byte Value							
			W	Timer1 12-bit PWM Duty Low Byte Value							
$0018	Timer2-3 Ctrl0 (P_TMR2_3_Ctrl0)	00h	R	0	Timer3 Function Selection			0	0	Timer2 Function Selection	

Address	Function	ERST value	R/W	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
			W	-	Timer3 Function Selection			-	-	Timer2 Function Selection	
$0019	Timer2-3 Ctrl1 (P_TMR2_3_Ctrl1)	00h	R	0	Timer3 Pre-scale Selection			0	Timer2 Pre-scale Selection		
			W	-	Timer3 Pre-scale Selection			-	Timer2 Pre-scale Selection		
$001A	Timer2 Counter (P_TMR2_Count)	00h	R	Timer2 8-bit Counter Register							
	Timer2 Preload (P_TMR2_Preload)		W	Timer2 8-bit Preload Register							
	Capture2 (P_TMR2_Cap)	00h	R	Timer2 8-bit Capture Width Value							
			W	Timer2 8-bit Low Byte Preload Register							
$001C	Timer3 Counter (P_TMR3_Count)	00h	R	Timer3 8-bit or 16-bit Low Byte Counter Register							
	Timer3 Preload (P_TMR3_Preload)		W	Timer3 8-bit or 16-bit Low Byte Preload Register							
	Compare3 (P_TMR3_Comp)	00h	R	Timer3 8-bit or 16-bit Low Byte Counter Register							
			W	Timer3 8-bit or 16-bit Compare Low Byte Value							
	Capture3 (P_TMR3_Cap)	00h	R	Timer3 8-bit or 16-bit Capture Low Byte Width Value							
			W	Timer3 8-bit or 16-bit Low Byte Preload Register							
	PWM3 Period (P_TMR3_PWMPeriod)	00h	R	Timer3 12-bit PWM Period Low Bye Value							
			W	Timer3 12-bit PWM Period Low Bye Value							
$001D	Timer3 Counter High (P_TMR3_CountHi)	00h	R	Timer3 16-bit High Byte Counter Register							
	Timer3 Preload High (P_TMR3_PreloadHi)		W	Timer3 16-bit High Byte Preload Register							
	Compare3 High Byte (P_TMR3_CompHi)	00h	R	Timer3 16-bit High Byte Counter Register							
			W	Timer3 16-bit Compare High Byte Value							
	Capture3 High Byte (P_TMR3_CapHi)	00h	R	Timer3 16-bit Capture High Byte Width Value							
			W	Timer3 16-bit High Byte Preload Register							
	Capture3 Cycle (P_TMR3_CapCycle8)	00h	R	Timer3 8-bit Capture Cycle Value							
			W	Timer3 16-bit High Byte Preload Register							
	PWM3 Duty Period (P_TMR3_DutyPeriod)	00h	R	Timer3 12-bit PWM Duty High Byte Value				Timer3 12-bit PWM Period High Byte Value			
			W	Timer3 12-bit PWM Duty High Byte Value				Timer3 12-bit PWM Period High Byte Value			
$001E	PWM3 Duty (P_TMR3_PWMDuty)	00h	R	Timer3 12-bit PWM Duty Low Byte Value							
			W	Timer3 12-bit PWM Duty Low Byte Value							

3). $0026~002D: Interrupt and ADC control

Address	Function	ERST value	R/W	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
$0026	Interrupt Flag 2	00h	R	0	0	ITVALIF	0	UARTIF	SPIIF	0	0

	(P_INT_Flag2)		W	Write '1' to clear corresponding interrupt flag(s), but SPIIF cannot be clear.							
$0027	Interrupt Ctrl 2	00h	R	0	0	ITVALIE	0	0	0	0	0
	(P_INT_Ctrl2)		W	-	-	ITVALIE	-	-	-		
$0028	A/D Control 0	06h	R	ADEN	ADVRT	0	0	ADCS2	ADCS1	ADCS0	ADRDY
	(P_AD_Ctrl0)		W	ADEN	ADVRT	-	-	ADCS2	ADCS1	ADCS0	STARTB
$0029	A/D Input Channel Ctrl 1	00h	R	PCFG7	PCFG6	PCFG5	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0
	(P_AD_Ctrl1)		W	PCFG7	PCFG6	PCFG5	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0
$002A	A/D Control 2	00h	R	ADCE	0	ADS2	ADS1	ADS0	0	0	0
	(P_AD_Ctrl2)		W	ADCE	-	ADS2	ADS1	ADS0	-	-	-
$002B	A/D Data Hi	00h	R	ADR9	ADR8	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2
	(P_AD_DataHi)		W	-							
$002C	A/D Data Lo	00h	R	ADR1	ADR0	0	0	0	0	0	0
	(P_AD_DataLo)		W	-							
$002D	Buzzer Control	00h	R	INTIMS3	INTIMS2	INTIMS1	INTIMS0	BZFS3	BZFS2	BZFS1	BZFS0
	(P_BUZ_Ctrl)		W	INTIMS3	INTIMS2	INTIMS1	INTIMS0	BZFS3	BZFS2	BZFS1	BZFS0

Note: UARTIF and INTIMS3 are available on the SPMC65P2408A only.

4). $0030~$0036：Special control register with double write access

Address	Function	ERST value	R/W	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
$0030	System Ctrl	C0h	R	POR	ERST	LVR	0	WDR	IAR	0	0
	(P_SYS_Ctrl)		W	Write '1' to clear corresponding reset flag(s)						-	-
$0031	Mode Ctrl	00h	R	0	0	0	0	0	0	0	0
	(P_Mode_Ctrl)		W	Write #$5A to enter STOP mode. Write #$A5 to enter HALT mode. Write #$66 to reset all internal modules, except CPU.							
$0032	Watchdog Ctrl	F0h	R	SCKEN	WDS2	WDS1	WDS0	0	0	0	0
	(P_WDT_Ctrl)		W	SCKEN	WDS2	WDS1	WDS0	-	-	-	-
$0034	IRQ Option	00h	R	IRQ3ES	IRQM3	IRQ2ES	IRQM2	IRQ1ES / CAP3ES	IRQM1	IRQ0ES / CAP2ES	IRQM0
	(P_IRQ_Opt1)		W	IRQ3ES	IRQM3	IRQ2ES	IRQM2	IRQ1ES / CAP3ES	IRQM1	IRQ0ES / CAP2ES	IRQM0
$0035	Slew Rate Control	00h	R	0	0	0	0	0	0	0	SLOWE
	(P_IO_Opt)		W	-	-	-	-	-	-	-	SLOWE
$0036	LVR Option	00h	R	0	0	0	0	0	0	0	LVRV40
	(P_LVR_Opt)		W	-	-	-	-	-	-	-	LVRV40

Note: LVRV40 cannot be cleared by external reset.

5). $0038~$003C：SPI communication

Address	Function	ERST value	R/W	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
$0038	SPI Ctrl0	00h	R	SPIEN	MOD	SCKPHA	SCKPOL	SMS	SCKSEL2	SCKSEL1	SCKSEL0
	(P_SPI_Ctrl0)		W	SPIEN	MOD	SCKPHA	SCKPOL	SMS	SCKSEL2	SCKSEL1	SCKSEL0
$0039	SPI Ctrl1	02h	R	SMSEN	SWRST	0	0	0	0	SPISPCLK1	SPISPCLK0

	(P_SPI_Ctrl1)		W	SMSEN	SWRST	-	-	-	-	SPISPCLK1	SPISPCLK0
$003A	SPI TX RX Status (P_SPI_Status)	00h	R	SPIIF	SPIIEN	TXBF	0	0	0	0	BUFFull
			W	Write "1" to clear	SPIIEN	-	-	-	-	-	-
$003B	SPI Transmission DATA (P_SPI_TxData)	00h	R	0	0	0	0	0	0	0	0
			W	SPI data out							
$003C	SPI Reception DATA (P_SPI_RxData)	00h	R	SPI data in							
			W	-	-	-	-	-	-	-	-

6). $0046~$0049：UART communication (2408A Only)

Address	Function	ERST value	R/W	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
$0046	UART Control (P_UART_Ctrl)	00h	R	RXIE	TXIE	RXEN	TXEN	SOFTRST	STOPSEL	PSEL	PEN
			W	RXIE	TXIE	RXEN	TXEN	SOFTRST	STOPSEL	PSEL	PEN
$0047	UART Baud Rate Divider (P_UART_Baud)	00h	R	UARTBAUD7	UARTBAUD6	UARTBAUD5	UARTBAUD4	UARTBAUD3	UARTBAUD2	UARTBAUD1	UARTBAUD0
			W	UARTBAUD7	UARTBAUD6	UARTBAUD5	UARTBAUD4	UARTBAUD3	UARTBAUD2	UARTBAUD1	UARTBAUD0
$0048	UART STATUS (P_UART_Status)	00h	R	RXIF	TXIF	BUSY	0	0	OERR	PERR	FERR
			W	-	-	-	-	-	OERR	PERR	FERR
$0049	UART DATA (P_UART_Data)	00h	R	UART Received Data							
			W	UART Transmitted Data							

Note: UART function is available on the SPMC65P2408A only, and it is illegal in the SPMC65P2404A.

7). $0058~$005F：Port Data buffer

Address	Function	ERST value	R/W	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
$0058	Capture Control	00h	R	CAPOPT	0	CAPIP3	CAPIP2	CAPIP1	CAPIP0	CAP1ES	CAP0ES
	(P_CAP_Ctrl)		W	CAPOPT	-	CAPIP3	CAPIP2	CAPIP1	CAPIP0	CAP1ES	CAP0ES
$0059	Port A Data Buffer	00h	R	Port A Data Buffer							
	(P_IOA_Buf)		W	Port A Data Buffer							
$005A	Port B Data Buffer	00h	R	Port B Data Buffer							
	(P_IOB_Buf)		W	Port B Data Buffer							
$005B	Port C Data Buffer	00h	R	0		Port C Data Buffer					
	(P_IOC_Buf)		W	-		Port C Data Buffer					
$005C	Port D Data Buffer	00h	R	0			Port D Data Buffer				
	(P_IOD_Buf)		W	-			Port D Data Buffer				

Note: Port C bit[5:4] and Port D bit[4:3] are available on the SPMC65P2408A only, and they are illegal in the SPMC65P2404A.

5.2.4. Special Control Register

The registers located between $0030 and $0036 are related to system operation. Therefore, SPMC65P2404A/2408A provides a protective writing method for those registers. Those registers have to be double written. The processes of double write are formed with two consecutive writing cycles to the target address, which are between $0030 and $0036. Programmer has to double write to modify the content of control registers located between $0030 and $0036. The purposes of the double write process is to keep the content of the control registers correct and protect them from the data bus or address bus noise interfere

[Example] 5.2.2 Setting some significant registers located between $0030 and $0036 by double writing method

```
lda   #$FF                          ; Clear Reset flag
sta   P_SYS_Ctrl
sta   P_SYS_Ctrl
lda   #C_MODE_Reset                 ; reset all I/O
sta   P_MODE_Ctrl
sta   P_MODE_Ctrl
lda   #C_WDT_Div_16384:             ; WDI= Fslow(25KHz)/16384= 1.5Hz
sta   P_WDT_Ctrl
sta   P_WDT_Ctrl
lda   #C_IRQOpt1_IRQ0ES             ; INT0 is rising edge trigger, other are falling
sta   P_IRQ_Opt1
sta   P_IRQ_Opt1
lda   #$00                          ; set LVR= 2.5V
sta   P_LVR_Opt
sta   P_LVR_Opt
```

5.2.5. Device Configuration Register

The SPMC65P2404A/2408A provides three bytes of registers for system configuration. These three registers will be written at the same time when programmer programs the OTP ROM; thus, the chip will start at proper state after power-on or external reset. Please reference to section 5.13 for the detailed setting and related registers.

5.2.6. User Information Register

The SPMC65P2404/2408A provides 16 bytes of OTP memory for programmer to store their information. These 16 bytes range from $7FF0 to $7FFF.

Sunplus defines $7FF0~$7FF3 as the serial number and the remaining address as the free definition location which can record product in oration.

5.3. Clock Source

The SPMC65P2404A/2408A supports Crystal / Resonator, RC oscillator, or external clock sources, as shown in the following diagram. They can be selected by device configuration register P_MO ($7FE0). The detailed register setting of device configuration register is given in section 5.13.

Table 5.3.1 shows the resistor and capacitance value for RC oscillator. Programmer should use the suggested value to choose system clock. Note that when choosing Crystal / Resonator or external clock, the clock frequency will be divided by two after feeding into chip. It means if programmer wants to operate it at 8MHz, the crystal frequency or external clock frequency must be set to 16MHz.

[Table] 5.3.1 R-oscillator resistor vs. frequency table

Frequency (Hz), VDD=5.0V	Resistor (Ω), C=50PF
F_{SYS} = 135K	75k
F_{SYS} = 195K	51k
F_{SYS} = 480K	20k
F_{SYS} = 0.9M	10k
F_{SYS} = 1.6M	5.1k
F_{SYS} = 2.25M	3.3k

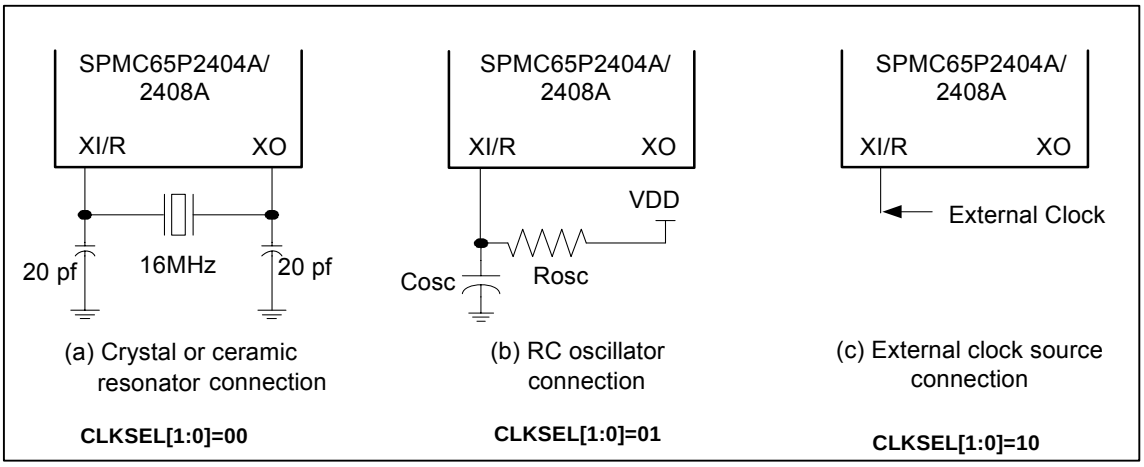


Figure 5.3-1 Three types of clock sources

5.4. Power Saving Mode

5.4.1. Introduction

To reduce the current consumption when system does not need to be active, STOP mode and HALT mode can be utilized. These two modes are able to reduce power consumption to save power. They also feature different wakeup time. Programmer has to choose suitable mode for applications. To transfer into STOP mode or HALT mode, programmer must double-write correspond value to Mode Control Register. For more information about STOP and HALT modes will be depicted in the next two sections. The Fig. 5.4-1 shows the mode transfer for SPMC65P2404A/2408A. The system always starts from reset and operates in normal mode at beginning. Programmer can transfer the operating mode to STOP or HALT mode from normal mode. After entering STOP or HALT mode, the system can only be allowed back to normal mode. For example, if programmer wants to change it from HALT mode to STOP mode, the system has to be changed from HALT mode to normal mode first and then switched from normal

mode to STOP mode. Same procedures are applied for switching from STOP mode to HALT mode.

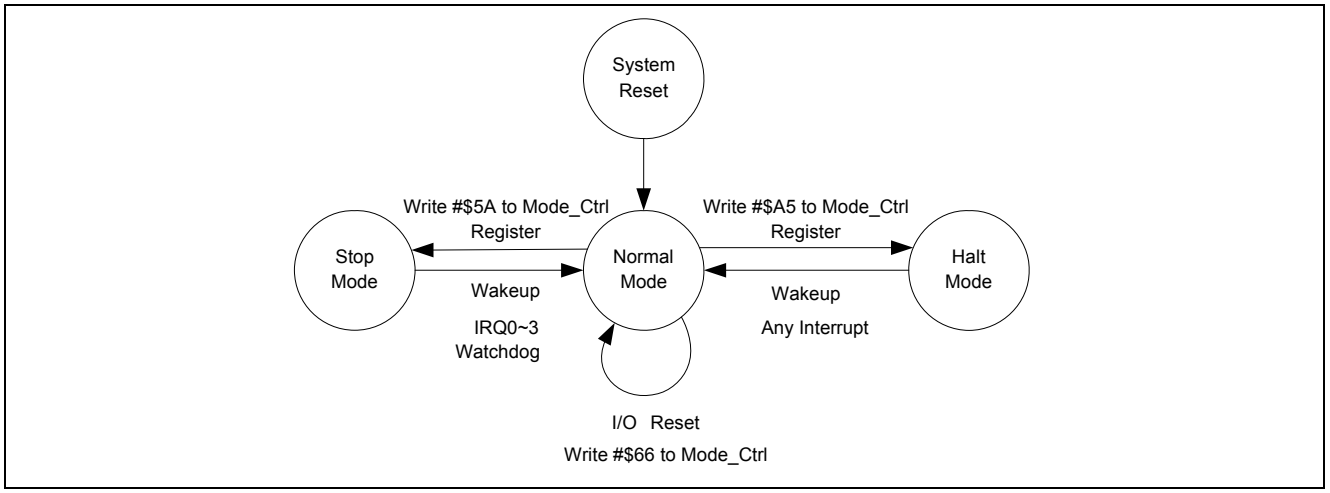


Figure 5.4-1 Power saving mode operation

5.4.2. STOP Mode

The STOP mode function will disable all system clocks, including the clock generation circuit. Once the system enters the STOP mode, only the activated external interrupt events (from I/Os) or watchdog interrupt event can recover the normal operation from STOP mode. Be aware that if programmer sets the watchdog interrupt event as wakeup source, the power will be larger than disabling the watchdog and setting wakeup source from external interrupt.

In order to wait for the clock source to be stable, it will delay for a certain time when system receives an interrupt event. This delay time is set at 40ms in typical and will range from 20ms to 60ms. After the clock is stable, if programmer enables the interrupt exception (clear the interrupt flag in status register by CLI), the program will jump into exception routine immediately; otherwise, the program will start from the next instruction of the STOP mode command. To confirm the interrupt event is able to wake the system up, the corresponding interrupt enabling bits must be set before entering the STOP mode.

If programmer uses the watchdog interrupt to recover the system, the on-chip watchdog RC oscillator clock must be activated before entering STOP mode. Besides, using watchdog to recover the system, programmer must set the watchdog interrupt rate properly to avoid system reset occurring at the clock stable time, as shown in Figure 5.4-3.

To enter STOP mode, programmer must write #$5A to mode control register (P_MODE_Ctrl) twice.

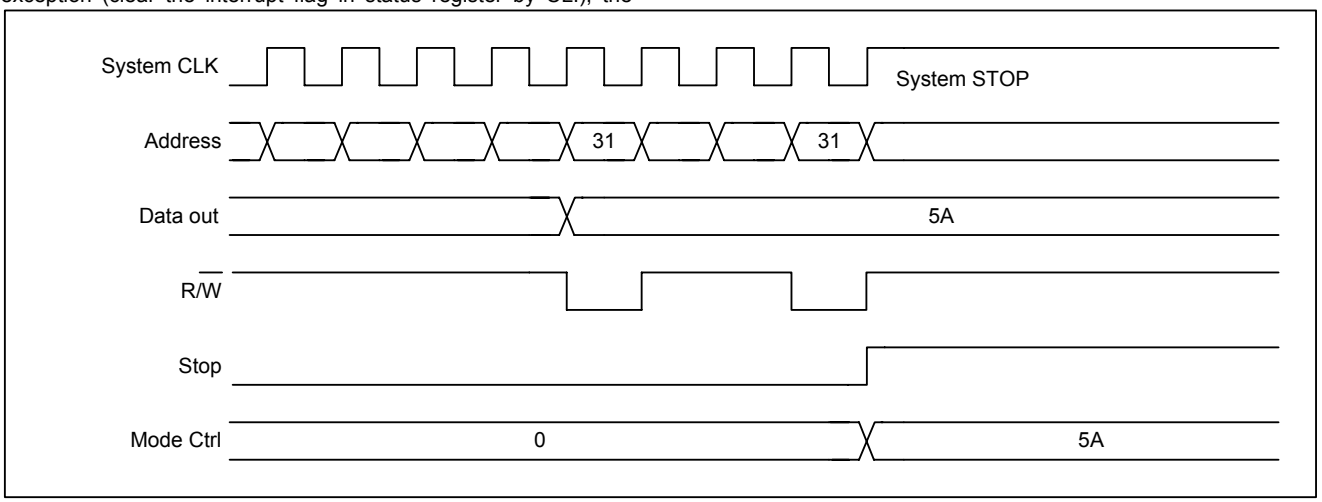


Figure 5.4-2 Enter STOP mode

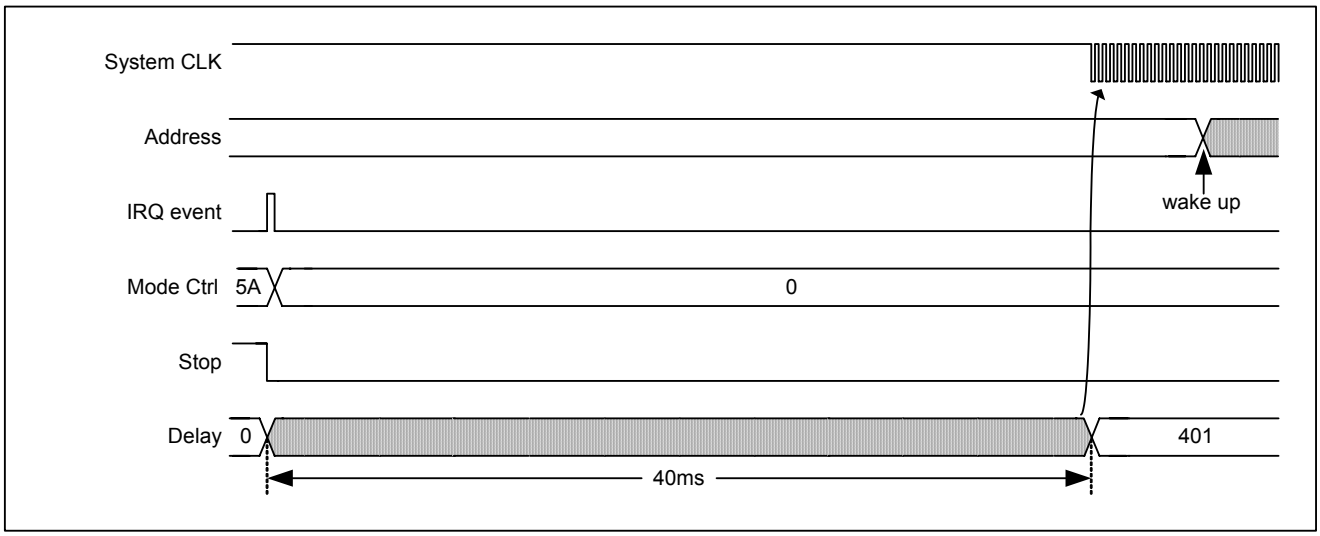


Figure 5.4-3 Leave STOP mode by external interrupt

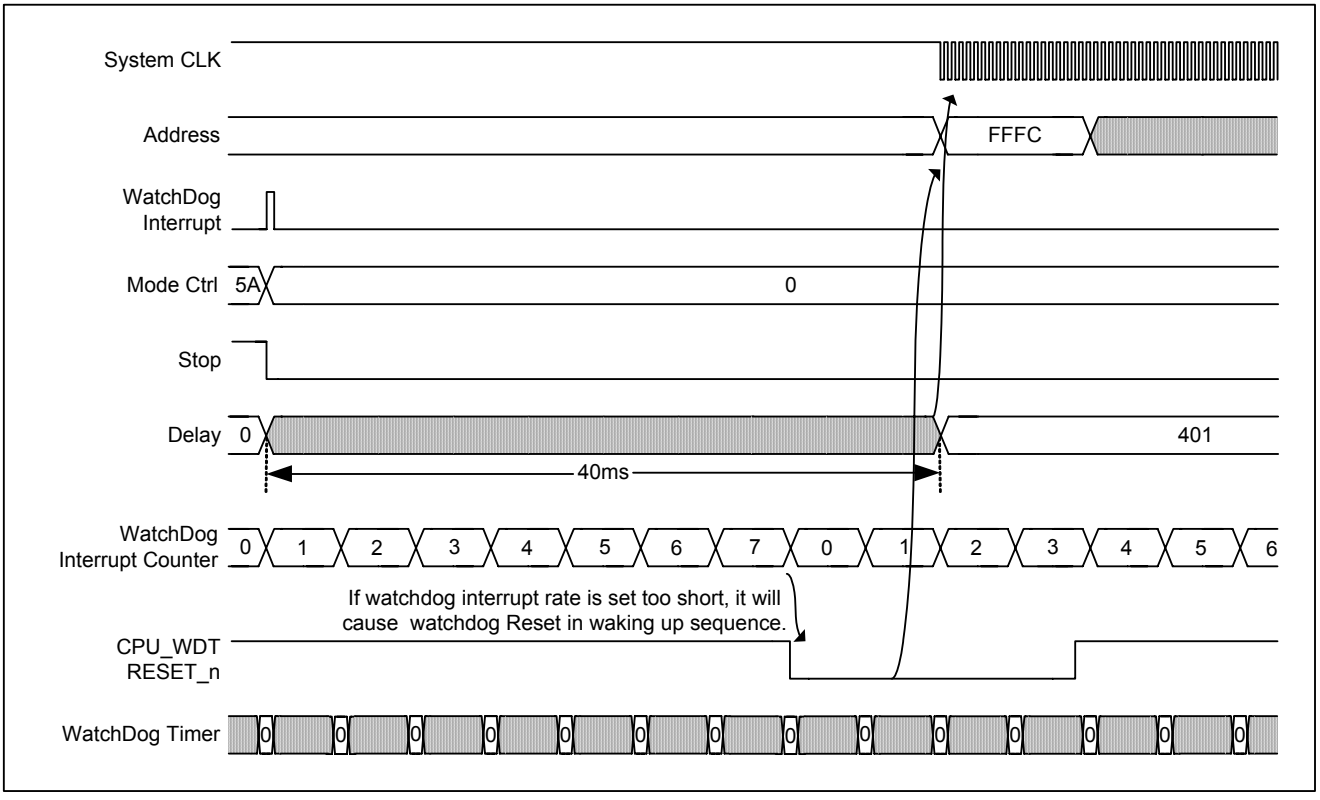


Figure 5.4-4 Leave STOP mode by watchdog with incorrect interrupt rate

5.4.3. HALT mode

The HALT mode will disable the CPU clock. The peripheral clock remains active so that the peripheral will keep running in HALT mode.

All interrupt sources will recover the normal operation from HALT mode. As STOP mode, if programmer enables the interrupt exception (clear the interrupt flag in status register by CLI), the program will jump into exception routine immediately; otherwise, the system will run from the next instruction of the HALT mode command immediately. In HALT mode, there is no delay time from the wakeup event to CPU active.

To enter HALT mode, programmer must write #$A5 to mode

control register (P_MODE_Ctrl) twice.

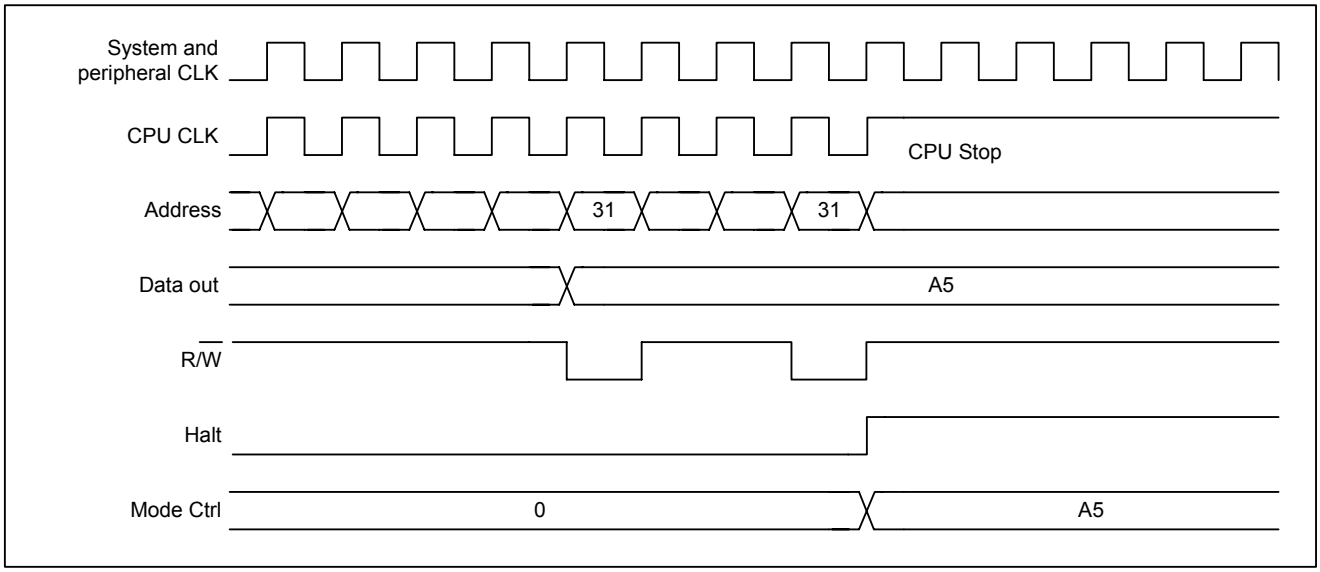


Figure 5.4-5 Enter HALT mode

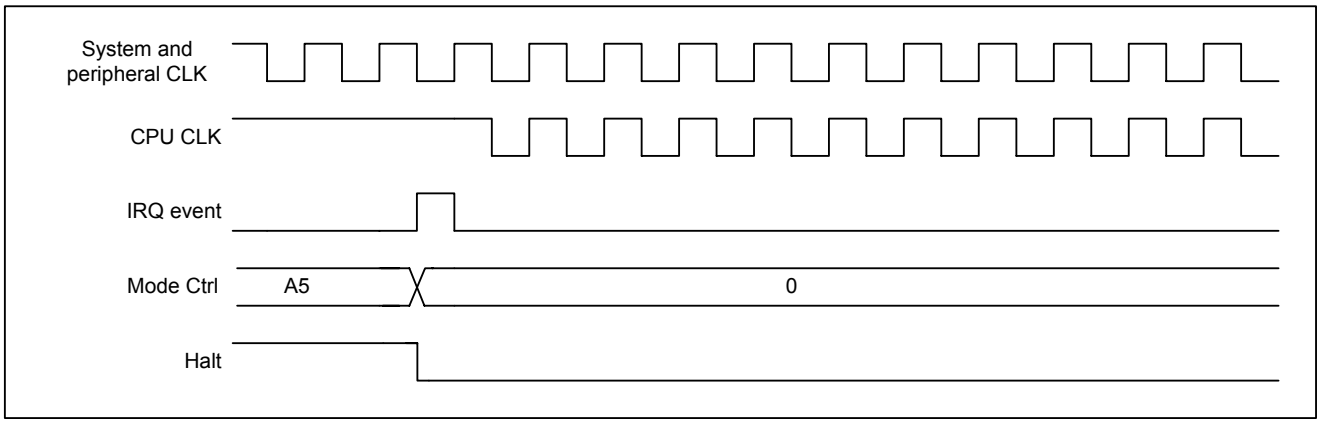


Figure 5.4-6 Leave HALT mode from interrupt

1). Mode Control Register (P_MODE_Ctrl, $0031)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	Mode_Ctrl7	Mode_Ctrl6	Mode_Ctrl5	Mode_Ctrl4	Mode_Ctrl3	Mode_Ctrl2	Mode_Ctrl1	Mode_Ctrl0
ACCESS	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
DEFAULT	0	0	0	0	0	0	0	0

Note: This byte must be double-write.

Bit [7:0]: **Mode ctrl** [7:0]: Operation mode control register

Read:

Data is always #$00

Write (twice):

#$5A = enter STOP mode

#$A5 = enter HALT mode

#$66 = reset all internal modules, except CPU

[Example] 5.4.1 Let MCU enter STOP mode

```
lda     #C_MODE_STOP                    ; STOP command
```

```
    sta     P_MODE_Ctrl
    sta     P_MODE_Ctrl
```

5.5. Interrupt

5.5.1. Introduction

The SPMC65P2404A/2408A provides seven types of interrupt sources with two interrupt levels. The seven types of interrupt sources are external interrupt, timer interrupt, watchdog interrupt, ADC Interrupt, real-time interrupt, communication interrupt and capture interrupt. The two interrupt levels are non-maskable interrupt and normal interrupt. They will be discussed in the following sections.

5.5.2. External Interrupt

There are four external interrupt sources in SPMC65P2404A/2408A. They are IRQ0, IRQ1, IRQ2 or IRQ3, which correspond to I/O port PB4, PB5, PD0 and PD1. These IRQ signals are combined with status and control registers to generate the maskable interrupt events to CPU.

For all external IRQ channels, each IRQ channel has individual interrupt control and status bits. Once an external interrupt takes place, the flag will be set and keep until programmer's program clears the flag. Besides setting corresponding interrupt control registers, programmer still needs to use 'CLI' instruction to clear the interrupt flag in status register in CPU.

The interrupt request signal will be generated after interrupt is enabled. Each IRQx has a register used to set the trigger mode of the interrupt event. The trigger mode can be selected as either edge trigger mode or level trigger mode. When the interrupt channel is enabled with edge trigger mode, an active transition edge on the external interrupt inputs will generate an interrupt. If the channel is enabled with level trigger mode, the active level of the external interrupt inputs will set the interrupt event until the active level condition is removed.

5.5.3. Non Maskable Interrupt

One of the four external interrupts can be configured into the source of non-maskable interrupt by NMI selection register ($7FE3). Except the setting on NMI selection register, the configuration and enable/control of the NMI interrupt source is the same as normal external interrupt.

When an external interrupt source is set into NMI interrupt, it can be used to trigger the NMI interrupt routine regardless of whether CPU is running. In other words, even the CPU is running an NMI interrupt routine, another NMI interrupt trigger signal will still interrupt current routine and start a new one. By this feature, external interrupt source that is set into NMI interrupt can only use edge trigger mode rather than level trigger mode. In other words, level trigger is illegal to NMI interrupt. This is to prevent overflow on the stack memory.

5.5.4. Peripheral Interrupt

Except the external interrupt, there are still six interrupt sources in SPMC65P2404A/2408A - Timer Interrupt, Watchdog Interrupt, ADC Interrupt, Real-time Interrupt, Communication Interrupt and Capture Interrupt. These interrupts have individual status (occurred or not) and control (enable or not) registers. In general, once an interrupt event occurs, the corresponding flag bit will be set. If the related interrupt control bit is set to enable interrupt, an interrupt request signal will be generated and will be dealt by service routine. If the related interrupt control bit is disabled, programmer still can observe the corresponding flag bit, but no interrupt request signal will be generated. The interrupt flag bits must be cleared in the interrupt service routine to prevent program from deadlock in interrupt service routine. With any instruction, interrupts pending during the previous instruction is served.

The timer interrupt, real-time interrupt, watchdog interrupt, ADC interrupt, SPI, UART and capture interrupt will be described in corresponding section.

[Table] 5.5.1 Interrupt source list

Source		Interrupt flag register	Interrupt control register	Source		Interrupt flag register	Interrupt control register
Timer Overflow	Timer0	T0OIF($000E.0)	T0OIE($000F.0)	External INT Input	IRQ0	IRQ0IF($000C.0)	IRQ0IE($000D.0)
	Timer1	T1OIF($000E.1)	T1OIE($000F.1)		IRQ1	IRQ1IF($000C.1)	IRQ1IE($000D.1)
	Timer2	T2OIF($000E.2)	T2OIE($000F.2)		IRQ2	IRQ2IF($000C.2)	IRQ2IE($000D.2)

	Timer3	T3OIF($000E.3)	T3OIE($000F.3)		IRQ3	IRQ3IF($000C.3)	IRQ3IE($000D.3)
Capture	Capture0	CAP0IF($000E.6)	CAP0IE($000F.6)	Analog	ADC	ADIF($000C.7)	ADIE($000D.7)
	Capture1	CAP1IF($000E.7)	CAP1IE($000F.7)				
	Capture2	CAP2IF($000C.0)	CAP2IE($000D.0)				
	Capture3	CAP3IF($000C.1)	CAP3IE($000D.1)	Communication	SPI	SPIIF($0026.2) R SPIIF($003A.7) R/W	SPIIE($003A.6)
					UART (2408A only)	RXIF($0048.7) TXIF($0048.6)	RXIE($0046.7) TXIE($0046.6)
Watchdog		WDIF($000C.6)	WDIE($000D.6)	Interval Timer		ITVALIF($0026.5)	ITVALIE($0027.5)

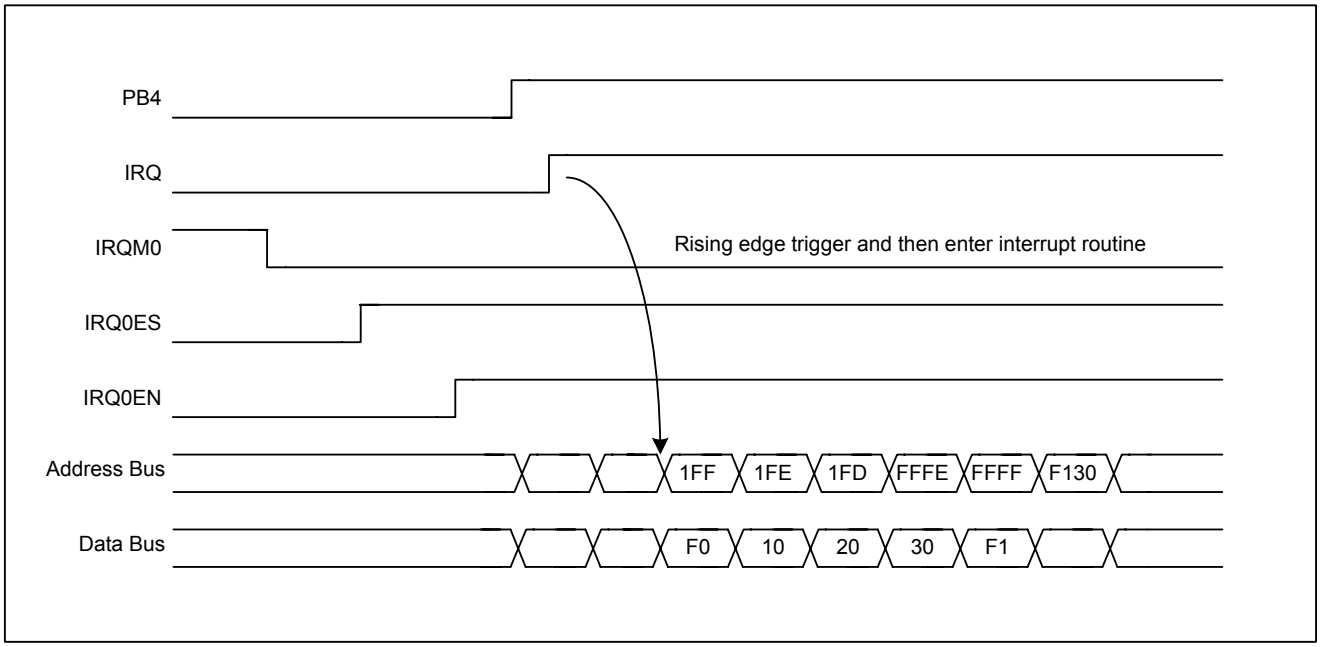


Figure 5.5-1 Interrupt triggered by IRQ0(PB4)

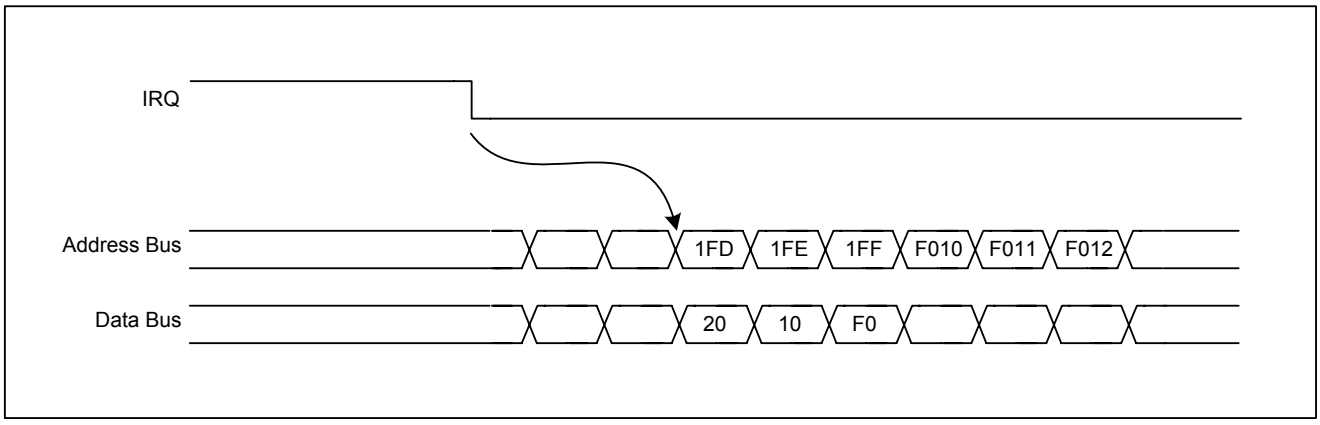


Figure 5.5-2 Leave interrupt routine

5.5.5. Interrupt register

1). Interrupt Flag Register0 (P_INT_Flag0, $000C)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	ADIF	WDIF	0	0	IRQ3IF	IRQ2IF	IRQ1IF/ CAP3IF	IRQ0IF/ CAP2IF
ACCESS	R/W	R/W	-	-	R/W	R/W	R/W	R/W
DEFAULT	0	0	0	0	0	0	0	0

This flag is cleared by writing the corresponding bit by “1”.

Bit 7 **ADIF**: A/D conversion interrupt flag
0 = no event
1 = event has occurred

Bit 6 **WDIF**: Watchdog interrupt flag
0 = no event
1 = event has occurred

Bit [5:4] Reserved

Bit 3 **IRQ3IF**: IRQ3 interrupt flag
0 = no event
1 = event has occurred

.

Bit 2 **IRQ2IF**: IRQ2 interrupt flag
0 = no event
1 = event has occurred

Bit 1 **IRQ1IF/CAP3IF**: IRQ1/CAP3 interrupt flag
0 = no event
1 = event has occurred
When capture3 is enabled, this bit shows the flag of capture input. Otherwise it shows the flag of interrupt1

Bit 0 **IRQ0IF/CAP2IF**: IRQ0/CAP2 interrupt flag
0 = no event
1 = event has occurred
When capture2 is enabled, this bit shows the flag of capture input. Otherwise it shows the flag of interrupt0

2). Interrupt Flag Register1 (P_INT_Flag1, $000E)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	CAP1IF	CAP0IF	-	-	T3OIF	T2OIF	T1OIF	T0OIF
ACCESS	R/W	R/W	-	-	R/W	R/W	R/W	R/W
DEFAULT	0	0	0	0	0	0	0	0

This flag is cleared by writing the corresponding bit by “1”.

Bit 7 **CAP1IF**: Capture1 interrupt flag
0 = no event
1 = event has occurred

Bit 6 **CAP0IF**: Capture0 interrupt flag
0 = no event
1 = event has occurred

Bit [5:4] Reserved

Bit 3 **T3OIF**: Timer3 overflow interrupt flag
0 = no event
1 = event has occurred

Bit 2 **T2OIF**: Timer2 overflow interrupt flag
0 = no event
1 = event has occurred

Bit 1 **T1OIF**: Timer1 overflow interrupt flag
0 = no event
1 = event has occurred

Bit 0 **T0OIF**: Timer0 overflow interrupt flag
0 = no event
1 = event has occurred

3). Interrupt Flag Register2 (P_INT_Flag2, $0026)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	-	ITVALIF	-	UARTIF	SPIIF	-	-
ACCESS	-	-	R/W	-	R	R	-	-
DEFAULT	0	0	0	0	0	0	0	0

This flag is cleared by writing the corresponding bit by “1”

Bit [7:6] Reserved

Bit 5 **ITVALIF**: Interval timer overflow interrupt flag

0 = no event

1 = event has occurred

Bit 4 Reserved

Bit 3 **UARTIF**: UART interrupt flag

0 = no event

1 = event has occurred

Bit 2 **SPIIF**: SPI interrupt flag

0 = no event

1 = event has occurred

The SPIIF can't be cleared at this register. It must be cleared at SPI status register (P_SPI_Status, $3A)

Bit [1:0] Reserved

4). Interrupt Control Register0 (P_INT_Ctrl0, $000D)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	ADIE	WDIE	-	-	IRQ3IE	IRQ2IE	IRQ1IE/ CAP3IE	IRQ0IE/ CAP2IE
ACCESS	R/W	R/W	-	-	R/W	R/W	R/W	R/W
DEFAULT	0	0	0	0	0	0	0	0

Bit 7 **ADIE**: A/D converter interrupt enable bit

0 = interrupt disable

1 = interrupt enable

Bit 6 **WDIE**: Watchdog interrupt enable bit

0 = interrupt disable

1 = interrupt enable

Bit [5:4] Reserved

Bit 3 **IRQ3IE**: IRQ3 interrupt enable bit

0 = interrupt disable

1 = interrupt enable

Bit 2 **IRQ2IE**: IRQ2 interrupt enable bit

0 = interrupt disable

1 = interrupt enable

Bit 1 **IRQ1IE**/ **CAP3IE**: IRQ1/CAP3 interrupt enable bit

0 = interrupt disable

1 = interrupt enable

When capture3 is enabled, this bit controls the capture3 interrupt. Otherwise it enables the interrupt1

Bit 0 **IRQ0IE**/ **CAP2IE**: IRQ0/CAP2 interrupt enable bit

0 = interrupt disable

1 = interrupt enable

When capture2 is enabled, this bit controls the capture2 interrupt. Otherwise it enables the interrupt0

5). Interrupt Control Register1 (P_INT_Ctrl1, $000F)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	CAP1IE	CAP0IE	-	-	T3OIE	T2OIE	T1OIE	T0OIE
ACCESS	R/W	R/W	-	-	R/W	R/W	R/W	R/W
DEFAULT	0	0	0	0	0	0	0	0

Bit 7 **CAP1IE**: Capture1 interrupt enable bit

0 = interrupt disable

1 = interrupt enable

Bit 6 **CAP0IE**: Capture0 interrupt enable bit

0 = interrupt disable

1 = interrupt enable

Bit [5:4] Reserved

Bit 3 **T3OIE**: Timer3 overflow interrupt enable bit

0 = interrupt disable

1 = interrupt enable

Bit 2 **T2OIE**: Timer2 overflow interrupt enable bit

0 = interrupt disable

1 = interrupt enable

Bit 1 **T1OIE**: Timer1 overflow interrupt enable bit

0 = interrupt disable

1 = interrupt enable

Bit 0 **T0OIE**: Timer0 overflow interrupt enable bit
0 = interrupt disable
1 = interrupt enable

6). Interrupt Control Register2 (P_INT_Ctrl2, $0027)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	-	ITVALIE	-	-	-	-	-
ACCESS	-	-	R/W	-	-	-	-	-
DEFAULT	0	0	0	0	0	0	0	0

Bit [7:6] Reserved

Bit 6 **ITVALIE**: Interval timer overflow interrupt enable bit
0 = interrupt disable
1 = interrupt enable

Bit [4:0] Reserved

7). NMI Selection Register (P_NMI, $7FE3)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	-	-	-	-	NMIS2	NMIS1	NMIS0
ACCESS	R	R	R	R	R	R	R	R
DEFAULT	1	1	1	1	1	1	1	1

Note: This byte can be found out of the FortisIDE mask option.

Bit [7:3] Reserved

Bit [2:0] **NMIS** [2:0]: Non-maskable interrupt source control bits
111 = disable
110 = reserved
101 = reserved
100 = reserved
011 = PD1 (INT3) is the NMI source
010 = PD0 (INT2) is the NMI source
001 = PB5 (INT1) is the NMI source
000 = PB4 (INT0) is the NMI source

8). IRQ Option1 Register (P_IRQ_Opt1, $0034)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	IRQ3ES	IRQM3	IRQ2ES	IRQM2	IRQ1ES/ CAP3ES	IRQM1	IRQ0ES/ CAP2ES	IRQM0
ACCESS	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
DEFAULT	0	0	0	0	0	0	0	0

Note: This byte must be double-write.

Bit 7 **IRQ3ES**: Edge select of IRQ3
IRQM3=”0”
0 = falling edge trigger
1 = rising edge trigger
IRQM3=”1”
0 = low level trigger
1 = high level trigger

Bit 6 **IRQM3**: IRQ3 trigger mode select bit
0 = edge trigger
1 = level trigger

Bit 5 **IRQ2ES**: Edge select of IRQ2
IRQM2=”0”
0 = falling edge trigger
1 = rising edge trigger
IRQM2=”1”
0 = low level trigger
1 = high level trigger

Bit 4 **IRQM2**: IRQ2 trigger mode select bit
0 = edge trigger
1 = level trigger

Bit 3 **IRQ1ES/CAP3ES**: Edge select of IRQ2/CAP3ES
IRQM1="0"
0 = falling edge trigger
1 = rising edge trigger
IRQM1="1"
0 = low level trigger
1 = high level trigger
When capture3 is enabled, this bit set the edge of capture input. Otherwise it set the edge of interrupt1.
1=falling edge clear counter
0=rising edge clear counter

Bit 2 **IRQM1**: IRQ1 trigger mode select bit
0 = edge trigger
1 = level trigger

Bit 1 **IRQ0ES/CAP2ES**: Edge select of IRQ0/CAP2ES
IRQM0="0"
0 = falling edge trigger
1 = rising edge trigger
IRQM0="1"
0 = low level trigger
1 = high level trigger
When capture2 is enabled, this bit set the edge of capture input. Otherwise it set the edge of interrupt0.
1 = falling edge clear counter
0 = rising edge clear counter

Bit 0 **IRQM0**: IRQ0 trigger mode select bit
0 = edge trigger
1 = level trigger

[Example] 5.5.1 Enable IRQ0 interrupt using rising edge trigger

```
set     P_IRQ_Opt1, CB_IRQOpt1_IRQ0ES          ; INT0 is rising edge trigger
set     P_IRQ_Opt1, CB_IRQOpt1_IRQ0ES
lda     #$FF
sta     P_INT_Flag0                            ; clear INT request flag
set     P_INT_Ctrl0,CB_INT_IRQ0IE              ; enable IRQ0 INT (INT0)
```

5.6. RESET

5.6.1. Introduction

There are five types of reset resources for the system, Power-On Reset (POR), External Reset (ERST), Low Voltage Reset (LVR), Watchdog Timer Reset (WDR), and Illegal Address Reset (IAR). These reset sources can be concluded as external events and internal events. The external events are coming from power line or external trigger event. The internal events come from the program exceptions or internal software reset event. Table 5.6.1 shows the affected region for each reset source.

5.6.2. Power-On Reset (POR)

A POR is generated when VDD is rising from 0v. When VDD rises to an acceptable level (~1.45V), the power on reset circuit will starts a power-on sequence. An internal counter will count about 40ms waiting for stable power, then another 40ms to wait for stable clock. After that, the system will operate in target speed and the system starts to activate.

The POR will reset whole chip and the registers. To take advantage of the POR, just connect the RESETB pin directly (or through the resistor) to VDD. This will eliminate external RC components usually needed to create a power-on reset.

5.6.3. External Reset (ERST)

The SPMC65P2404A/2408A provides an external pin to force the system returning to the initial status. The RESETB pin is used to connect an RC circuit, shown in Figure 5.6-1 Reset Circuit. This pin is a low active signal. When the RESETB pin falls below 0.3 x VDD, system will be forced to enter reset state.

The external reset pulse width must be larger than 200ns at least. Any pulse shorter than 200ns will be filtered and taken no effect on system. If a reset pulse that is long enough to take effect, the reset will be extended to 40ms + (1024 x System clock), as shown in Figure 5.6-3.

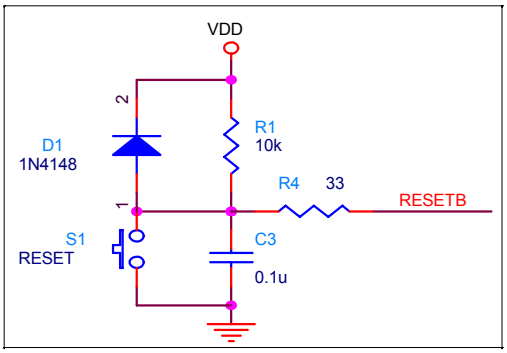


Figure 5.6-1 Reset circuit

Note:

1. The diode D helps discharge the capacitor quickly when VDD powers down.
2. R1 < 40KΩ is recommended to ensure that voltage drop across R does not violate the device's electrical specification.
3. R4 = 33Ω to 1KΩ will limit any current flowing into RESETB to prevent ESD.

5.6.4. Low Voltage Reset (LVR)

The on-chip Low Voltage Reset (LVR) circuitry forces the system entering reset state when the MCU voltage falls below the specific LVR trigger voltage. This function prevents MCU from working at an invalid operating voltage range.

A device configuration register (**$7FE0.2**) is used to enable or disable this function. If this function is enabled, the LVR circuit will monitor power level while chip is operating. If the power is lower than the specific level for a specific period, 1024 cycles of system clock, the system reset will be triggered. After the low power event is ended, the reset cycles of LVR will be extended with additional 1024 cycles of system clock.

Programmer can select low voltage reset level through LVRV40 (P_LVR_Opt, $0036.0). If LVRV40 is set to 0, the low voltage reset level is 2.5V, else it is 4.0V.

In general, the P_SYS_Ctrl will not be cleared by LVR. In addition, the LVR event will be ignored while in POR reset cycle or in RESETB reset cycle.

The LVR will be disabled if the system is in STOP mode. It means if the power level drops too low in STOP mode, it will trigger a POR directly rather than a LVR.

5.6.5. Watchdog Timer Reset (WDR)

On-chip watchdog circuitry makes the device entering reset when MCU goes into unknown state without any watchdog clearance. This function prevents the MCU to be stuck in an abnormal condition. The WDT can be disabled or enabled through device configuration register (P_MO, $7FE0). The internal reset of WDT will be generated by a time-out event of the WDT automatically when watchdog is enabled.

This reset signal uses the time base of WDT using the independent watchdog RC oscillator circuit and further dividing it by eight (wds[2:0] timing times 8). This signal will reset the CPU and restart the program, but no peripheral circuits will be reset. To avoid a WDT time-out reset, programmer has to write #$55 to P_WDT_Clr periodically. If a reset signal is generated, it will also clear the WDT counter to restart the WDT, but the WDT reset flag will not be cleared.

Please refer to chapter 5.12.1 for detail operations of Watchdog timer.

5.6.6. Illegal Address Reset (IAR)

The SPMC65P2404A/2408A provides an Illegal Address Reset to prevent system from entering illegal address. An illegal address is defined as if an instruction op-code is fetched from neither an address in the working area nor an address in the stack area, or a write command is sent to the OTP ROM area. When an illegal address occurs, the system will trigger a reset to CPU and force CPU back to the beginning point of the program, but no peripheral circuit is reset.

[Table] 5.6.1 Affected region for each reset source.

	POR	ERST	Software	LVR	WDR	IAR
CPU reset	Yes	Yes	No	Yes	Yes	Yes
Peripheral reset	Yes	Yes	Yes	Yes	No	No
P_IOA_Attrib	vvvvvvvv	vvvvvvvv	vvvvvvvv	vvvvvvvv	xxxxxxxx	xxxxxxxx
P_IOB_Attrib	vvvvvvvv	vvvvvvvv	vvvvvvvv	vvvvvvvv	xxxxxxxx	xxxxxxxx
P_IOC_Attrib	vvvvvvvv	vvvvvvvv	vvvvvvvv	vvvvvvvv	xxxxxxxx	xxxxxxxx
P_IOD_Attrib	vvvvvvvv	vvvvvvvv	vvvvvvvv	vvvvvvvv	xxxxxxxx	xxxxxxxx
P_SYS_Ctrl	100-00-0	x1x-xx-x	xxx-xx-x	xx1-xx-x	xxx-1x-x	xxx-x1-x
P_LVR_Opt	-------0	-------x	-------x	-------x	-------x	-------x

Note: **v:** depends on P_SECU.IOINIT setting, **x:** no effect

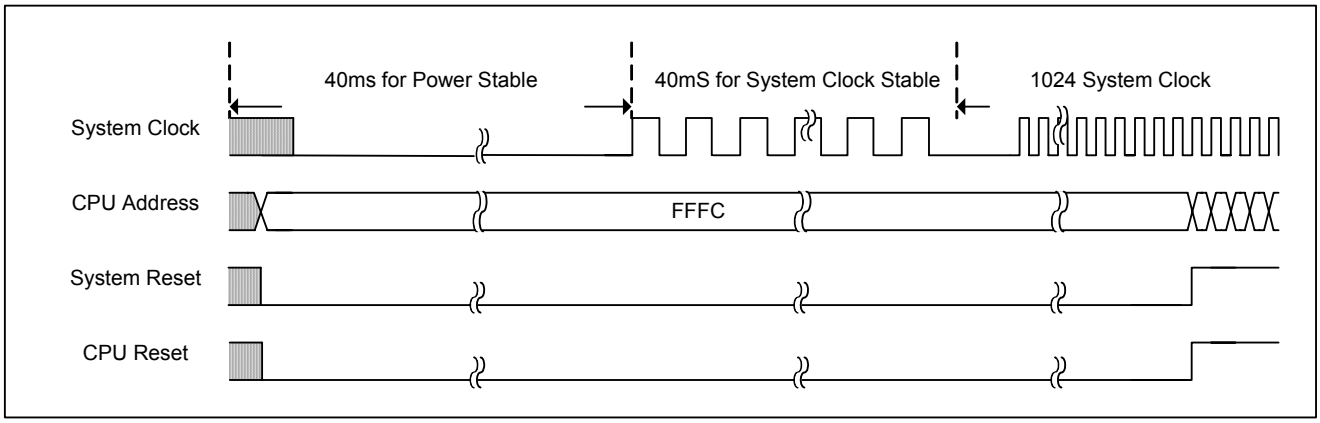


Figure 5.6-2 Power-on reset sequence

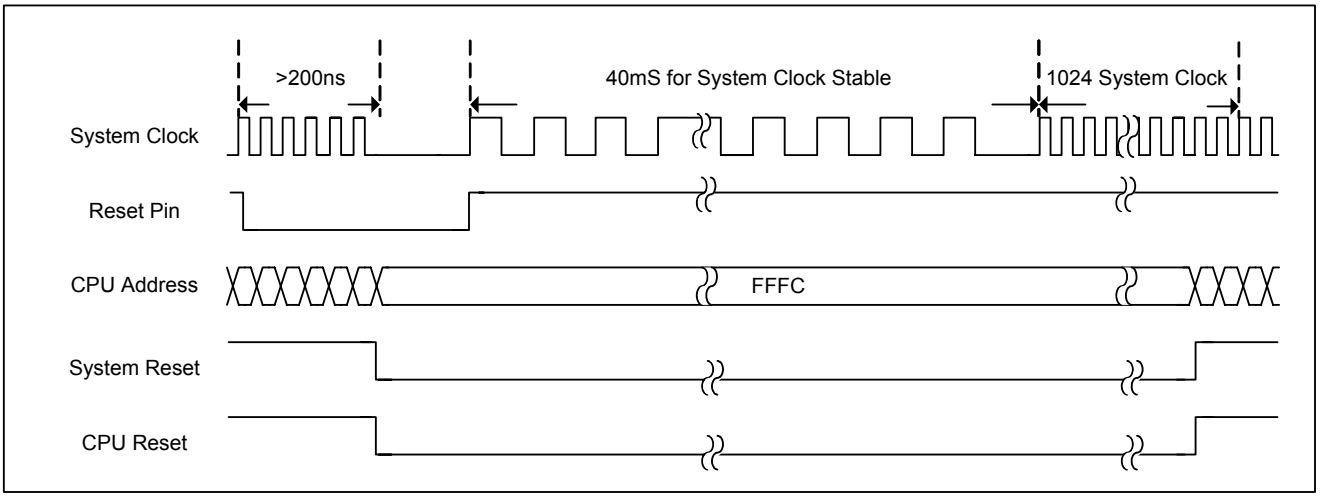


Figure 5.6-3 External reset sequence

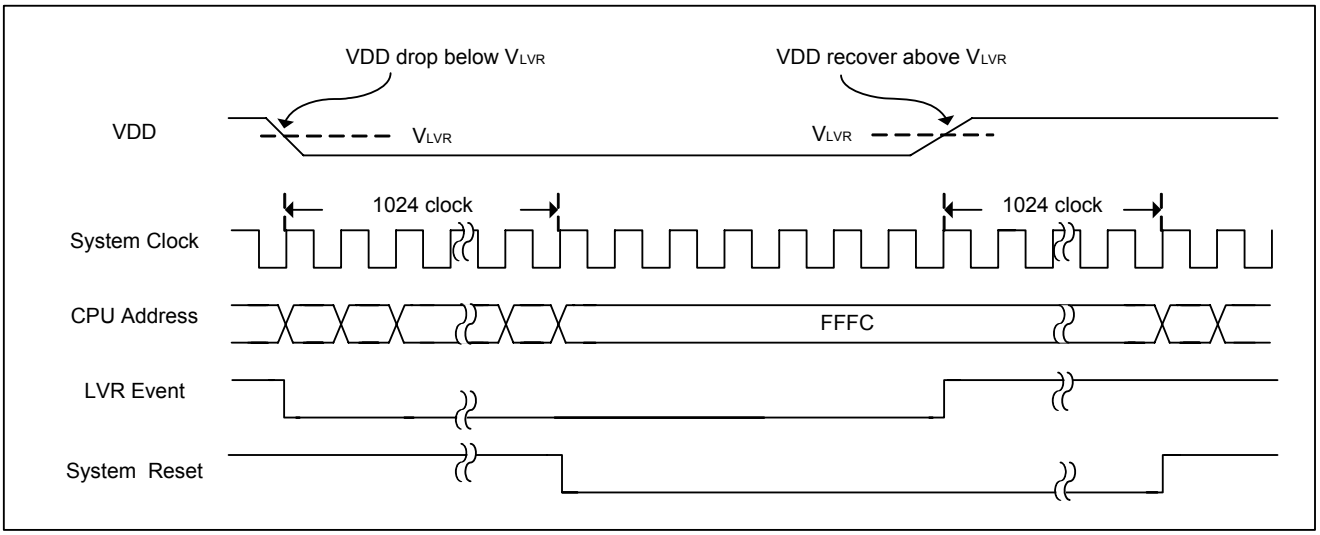


Figure 5.6-4 LVR reset sequence

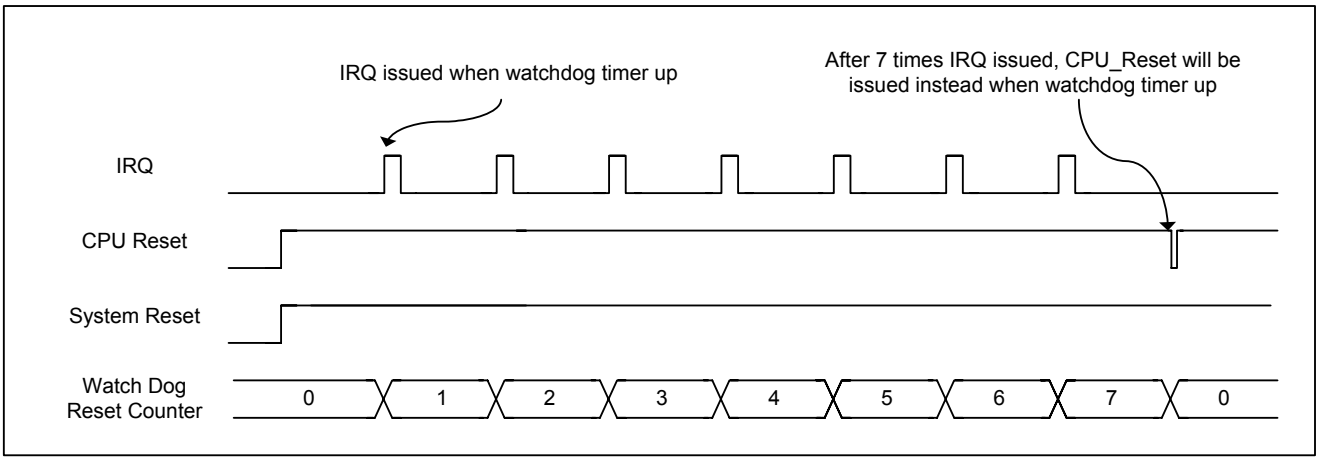


Figure 5.6-5 Watchdog reset sequence

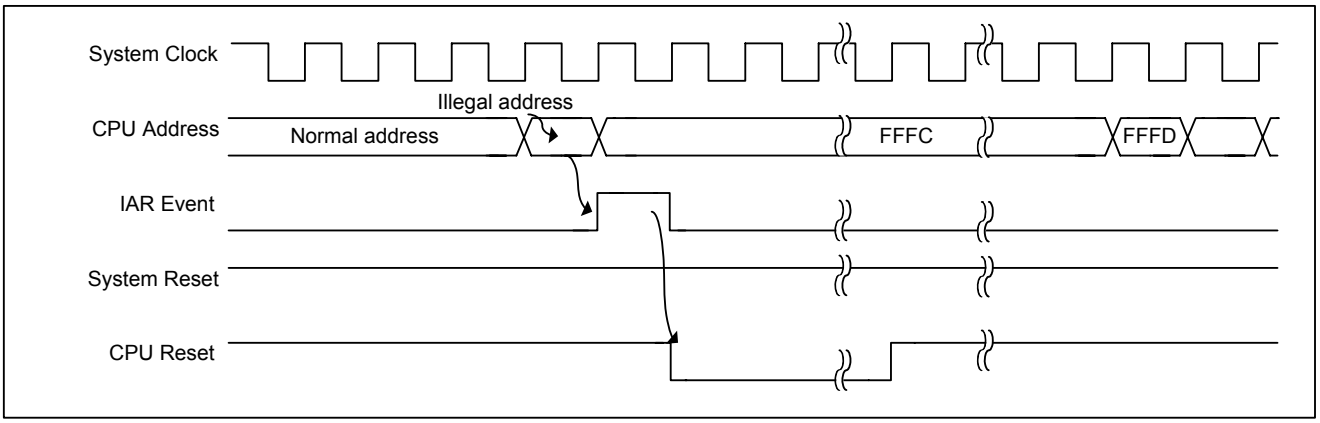


Figure 5.6-6 IAR reset sequence

1). System Control Register (P_SYS_Ctrl, $0030)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	POR	ERST	LVR	-	WDR	IAR	-	-
ACCESS	R/W	R/W	R/W	-	R/W	R/W	-	-
DEFAULT	1	0	0	0	0	0	0	0

Note: This byte must be double-write.

Bit 7 **POR**: Power on reset flag
0 = no POR
1 = POR is occurred
It is used to indicate the reset cycle is generated by power on reset. The POR will reset the whole internal blocks.

Bit 6 **ERST**: External reset flag
0 = no ERST
1 = ERST is occurred
It is used to indicate the reset cycle is generated by external reset input RESETB. The external reset will reset the whole internal blocks, except this register and P_LVR_Opt register.

Bit 5 **LVR**: Low voltage reset flag
0 = no LVR
1 = LVR is occurred
It is used to indicate the reset cycle generated by low voltage reset with period in case of the LVR being enabled by option. The LVR will reset the whole internal blocks, except this register.

Bit 4 Reserved

Bit 3 **WDR**: Watchdog timer reset flag
0 = no WDR
1 = WDR is occurred
It is used to indicate the reset cycle is generated by WDT overflow reset in case of the WDT being enabled by option. The WDT reset will reset CPU, but not whole internal blocks.

Bit 2 **IAR**: Illegal address reset flag
0 = no IAR
1 = IAR is occurred
It is used to indicate that the reset cycle is generated by IAR in case of the exception being detected. The IAR will reset CPU, but not whole internal blocks.

Bit 1 Reserved

Note: This flag is cleared by writing the corresponding bit by "1".

2). LVR Option Register (P_LVR_Opt, $0036)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	-	-	-	-	-	-	LVRV40
ACCESS	-	-	-	-	-	-	-	R/W
DEFAULT	0	0	0	0	0	0	0	0

Note: This byte must be double-write.

Bit [7:1] Reserved

Bit 0 **LVRV40**: LVR level select
0 = the LVR level is 2.5V
1 = the LVR level is 4.0V

Note: This select bit can only be set once and will be cleared only when Power-on reset occurred.

[Example] 5.6.1 Enable low voltage reset on condition that operation voltage is bellow 4V.

```
set   P_LVR_Opt,C_LVR_V40          ;set LVR=4.0V
set   P_LVR_Opt,C_LVR_V40
```

[Example] 5.6.2 Save the value of P_SYS_Ctrl register and then clear it

```
lda   P_SYS_Ctrl                   ; Read out reset flag
sta   G_MWorkReg1
lda   #$FF                         ; Clear Reset flag
sta   P_SYS_Ctrl
sta   P_SYS_Ctrl
```

5.7. I/O PORTS

5.7.1. Introduction

The SPMC65P2404A /2408A has four ports, Port A, Port B, Port C and Port D. These port pins may be multiplexed with an alternate function for the peripheral features on the device. In general, when an initial reset state, all ports are used as a general purpose input port. These I/O structures contain four parts: buffer, data, direction and attribution registers. Each corresponding bit in these ports should be given a value.

The setting rules are as follows:

- The direction setting determines whether this pin is an input or an output.
- The attribute setting gives a feature to the pin, float / not float.

- The buffer register setting affects the initial content of the pin. For input, it also determines the pull-high or pull-low setting. For output, it determines the output value. In addition, the bit-operation to the I/O port should be used for buffer register avoiding error setting.
- The data register is used to read the value on the port, which can be different when programmer sets the port to input pull-high/pull- low. Writing to this register will also write to the buffer register.

[Table] 5.7.1 I/O configurations

Direction (P_IOX_Dir)	Attribution (P_IOX_Attrib)	Buffer (P_IOX_Buf)	Function	Description
0	0	0	Pull Low (WPD)	Input with pull-low
0	0	1	Pull High (WPU)	Input with pull-high
1	0	1	Output High	Output Data
1	0	0	Output Low	Output Data
X	1	X	Float	Input with float

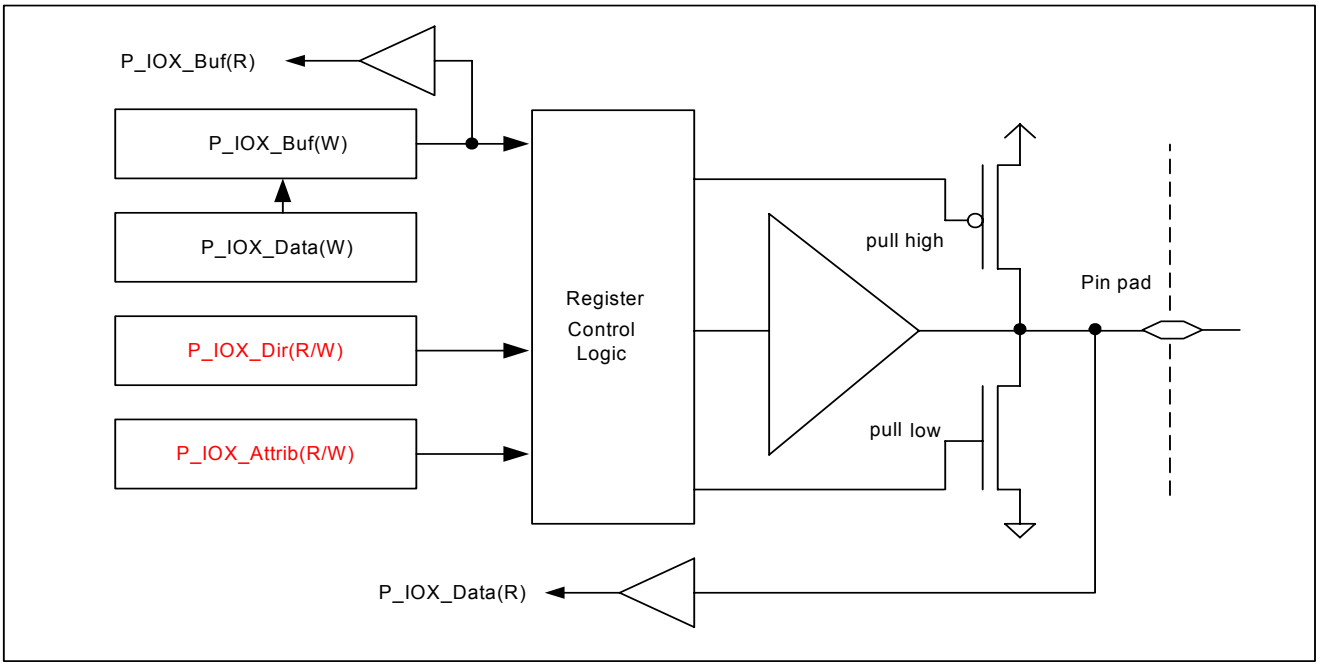


Figure 5.7-1 Block diagram of I/O port

5.7.2. Port A

Port A is an 8-bit bi-directional I/O port. The I/O Port A has 8 programmable I/Os that are controlled by data register P_IOA_Buf, direction control register P_IOA_Dir, and attribution register P_IOA_Attrib. P_IOA_Buf is used to store the data contents for output. To read the real I/O value, programmer has to read P_IOA_Data. Each pin of Port A is shared with A/D converter pin. The default function of Port A is general purpose I/O.

[Table] 5.7.2 Port A function list

Port A Pin	BIT#	Shared function
PA0	Bit0	Input/Output or AN0 analog input
PA1	Bit1	Input/Output or AN1 analog input

PA2	Bit2	Input/Output or AN2 analog input
PA3	Bit3	Input/Output or AN3 analog input
PA4	Bit4	Input/Output or AN4 analog input
PA5	Bit5	Input/Output or AN5 analog input
PA6	Bit6	Input/Output or AN6 analog input
PA7	Bit7	Input/Output or AN7 analog input

1). Port A IO Register (P_IOA_Data, $0000)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	P_IOA_Data							
ACCESS	R/W							
DEFAULT	00h							

Bit [7:0] **P_IOA_Data**: Port A value. Writing to this register will write to P_IOA_Buf.

2). Port A Buffer Register (P_IOA_Buf, $0059)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	P_IOA_Buf							
ACCESS	R/W							
DEFAULT	00h							

Bit [7:0] **P_IOA_Buf**: Port A buffer register

3). Port A Direction Register (P_IOA_Dir, $0004)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	P_IOA_Dir							
ACCESS	R/W							
DEFAULT	00h							

Bit [7:0] **P_IOA_Dir**: Port A direction register

0 = input

1 = output

4). Port A Attribution Register (P_IOA_Attrib, $0008)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	P_IOA_Attrib							
ACCESS	R/W							
DEFAULT	00h							

Bit [7:0] **P_IOA_Attrib**: Port A attribution register

0 = not float

1 = input with float

[Example] 5.7.1 Set Port A[7:0] as output with low data.

```
lda     #$FF                    ; store accumulator with $FF
sta     P_IOA_Dir
lda     #$00
sta     P_IOA_Attrib
sta     P_IOA_Data
```

[Example] 5.7.2 Set Port A[7:0] as Input with pulling Low.

```
lda     #$00                    ; store accumulator with $00
sta     P_IOA_Dir
sta     P_IOA_Attrib
sta     P_IOA_Data
```

5.7.3. Port B

Port B is an 8-bit bi-directional I/O port. The I/O Port B has 8 programmable I/Os, controlled by buffer register P_IOB_Buf, direction control register P_IOB_Dir, and attribution register P_IOB_Attrib. P_IOB_Buf is used to store the data contents for output. Reading P_IOB_Data will get the real IO value. In addition, Port B is multiplexed with various special functions. After reset, the default setting for port B is used as general I/O ports.

PB6 and PB7 also support slew rate control function. This function is controlled by Slew rate control register (P_IO_Opt, $0035). When slew rate enable bit is set to '1', the digital output on PB6 and PB7 will delay on falling edge with approximate 250ns which depended on system clock (F_{SYS}). The benefit of slew rate function is to prevent MCU from EMI when long distant transmission.

[Table] 5.7.3 Port B function list

Port B Pin	BIT	Shared function
PB0	Bit0	Input/Output or Capture0 input (Timer0) or external event counter input (Timer0)
PB1	Bit1	Input/Output or Capture1 input (Timer1) or external event counter input (Timer1)
PB2	Bit2	Input/Output or Timer0 compare output
PB3	Bit3	Input/Output or Timer1 compare output or Timer1 PWM output
PB4	Bit4	Input/Output or external interrupt 0 input or Capture2 input (Timer2) or external event counter input (Timer2)
PB5	Bit5	Input/Output or external interrupt 1 input or Capture3 input (Timer3) or external event counter input (Timer3)
PB6	Bit6	Input/Output (slew rate control) or buzzer output
PB7	Bit7	Input/Output (slew rate control) or ADC top reference voltage

1). Port B Data Register (P_IOB_ Data, $0001)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	P_IOB_ Data							
ACCESS	R/W							
DEFAULT	00h							

Bit [7:0] **P_IOB_ Data**: Port B Data value. Write to this register will write P_IOB_Buf.

2). Port B Buffer Register (P_IOB_Buf, $005A)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	P_IOB_Buf							
ACCESS	R/W							
DEFAULT	00h							

Bit [7:0] **P_IOB_Buf**: Port B buffer register

3). Port B Direction Register (P_IOB_Dir, $0005)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	P_IOB_Dir							
ACCESS	R/W							
DEFAULT	00h							

Bit [7:0] **P_IOB_Dir**: Port B direction register

0 = input

1 = output

4). Port B Attribution Register (P_IOB_Attrib, $0009)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	P_IOB_Attrib							
ACCESS	R/W							
DEFAULT	00h							

Bit [7:0] **P_IOB_Attrib**: Port B attribution register

0 = not float

1 = input with float

5). Slew Rate Control Register (P_IO_Opt, $0035)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	-	-	-	-	-	-	SLOWE
ACCESS	-	-	-	-	-	-	-	R/W
DEFAULT	0	0	0	0	0	0	0	0

Note: This byte must be double-write.

Bit [7:1] Reserved

Bit 0 **SLOWE**: Slew rate controls enable selection bit

1 = the slew rate level of PB[7:6] is slow

0 = the slew rate level of PB[7:6] is normal.

(Without slew rate control)

[Example] 5.7.3 Set Port B[3:0] as output with low data, Port B[7:4] as input with pulling high.

```
lda     #$0F                    ; store accumulator with $0F
sta     P_IOB_Dir
lda     #$00
sta     P_IOB_Attrib
lda     #$F0
sta     P_IOB_Data
```

[Example] 5.7.4 Set Port B[7:0] as input with float.

```
lda     #$FF                    ; store accumulator with $FF
sta     P_IOB_Attrib
```

5.7.4. Port C

The I/O Port C on the SPMC65P2404A/2408A has 4/6 programmable I/Os, controlled by buffer register P_IOC_Buf, direction control register P_IOC_Dir, and attribution register P_IOC_Attrib. P_IOC_Buf is used to store the data contents for I/O port. P_IOC_Data can be used to read the real value on the port. Port C is multiplexed with SPI and UART functions (UART is only for SPMC65P2408A). After reset, the default setting for port C is used as general I/O ports.

[Table] 5.7.4 Port C function list

Port C Pin	BIT	Shared function
PC0	Bit0	Input/Output or SPI slave select signal
PC1	Bit1	Input/Output or SPI clock output or Clock input
PC2	Bit2	Input/Output or SPI data input
PC3	Bit3	Input/Output or SPI data output
PC4	Bit4	Input/Output or UART Transmit data output (2408A Only)
PC5	Bit5	Input/Output or UART receive data input (2408A Only)

1). Port C Data Register (P_IOC_Data, $0002)

BIT	-	-	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	P_IOC_Data							
ACCESS	R/W							
DEFAULT	00h							

Bit [5:0] **P_IOC_Data**: Port C data value. Write to this register will write to P_IOC_Buf.

2). Port C Buffer Register (P_IOC_Buf, $005B)

BIT	-	-	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	P_IOC_Buf							
ACCESS	R/W							
DEFAULT	00h							

Bit [5:0] **P_IOC_Buf**: Port C buffer register

3). Port C Direction Register (P_IOC_Dir, $0006)

BIT	-	-	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	P_IOC_Dir							
ACCESS	R/W							
DEFAULT	00h							

Bit [5:0] **P_IOC_Dir**: Port C direction register

0 = input

1 = output

4). Port C Attribution Register (P_IOC_Attrib, $000A)

BIT	-	-	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	P_IOC_Attrib							
ACCESS	R/W							
DEFAULT	00h							

Bit [5:0] **P_IOC_Attrib**: Port C attribution register

0 = not float

1 = input with float

[Example] 5.7.5 Set Port C[5:0] as output with low data

```
lda   #$3F                    ; store accumulator with $3F
sta   P_IOC_Dir
lda   #$00
sta   P_IOC_Attrib
sta   P_IOC_Data
```

[Example] 5.7.6 Set Port C[5:0] as input with float.

```
lda   #$3F                    ; store accumulator with $3F
sta   P_IOC_Attrib
```

5.7.5. Port D

The I/O Port D on the SPMC65P2404A/2408A have 3/5 programmable I/Os that are controlled by buffer register P_IOD_Buf, direction control register P_IOD_Dir, and attribution register P_IOD_Attrib. P_IOD_Buf is used to store the data contents for output. P_IOD_Data can be used to read the real value on the port. Port D is multiplexed with various special functions. After reset, the default setting for port D is used as general I/O ports.

Table 4.4 Port D function list

Port D Pin	BIT	Shared function
PD0	Bit0	Input/Output or external interrupt 2 input
PD1	Bit1	Input/Output or external interrupt 3 input
PD2	Bit2	Input/Output or Timer3 compare output or Timer3 PWM output
PD3	Bit3	Input/Output or Timer2 compare output (2408A only)

PD4	Bit4	Input/Output or no shared function (2408A only)

1). Port D Data Register (P_IOD_Data, $0003)

BIT	-	-	-	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	P_IOD_Data							
ACCESS	R/W							
DEFAULT	00h							

Bit [4:0] **P_IOD_Data**: Port D data value. Writing to this register will write to P_IOD_Buf.

2). Port D Buffer Register (P_IOD_Buf, $005C)

BIT	-	-	-	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	P_IOD_Buf							
ACCESS	R/W							
DEFAULT	00h							

Bit [4:0] **P_IOD_Buf**: Port D buffer register

3). Port D Direction Register (P_IOD_Dir, $0007)

BIT	-	-	-	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	P_IOD_Dir							
ACCESS	R/W							
DEFAULT	00h							

Bit [4:0] **P_IOD_Dir**: Port D direction register

0 = input

1 = output

4). Port D Attribution Register (P_IOD_Attrib, $000B)

BIT	-	-	-	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	P_IOD_Attrib							
ACCESS	R/W							
DEFAULT	00h							

Bit [4:0] **P_IOD_Attrib**: Port D attribute register

0 = not float

1 = input with float

[Example] 5.7.7 Set Port D[4:0] as output with high data.

```
lda     #$1F                    ; store accumulator with $1F
sta     P_IOD_Dir
lda     #$00
sta     P_IOD_Attrib
```

```
lda     #$1F
sta     P_IOD_Data
```

5.8. Timer Module

5.8.1. Introduction

Both SPMC65P2404A and SPMC65P2408A are equipped with 4 channels of Timers. Timer0 and Timer2 are 8-bit timers. Timer1 and Timer3 are 16-bit timers. All of these four are up-count timers. Timer1 and Timer3 contain two powerful CCP (Capture/Compare/PWM) functions, controlled by corresponding control registers. These functions can be easily configured. Each timer's function summary is shown as below:

[Table] 5.8.1 Summary of timer function for SPMC65P2408A

	Timer/Event Counter		Capture		Compare		PWM
	8 bit	16 bit	8 bit	16 bit	8 bit	16 bit	
Timer 0	YES	None	Width	None	YES	None	None
Timer 1	YES	YES	Width/Cycle	Width	YES	YES	12 bit
Timer 2	YES	None	Width	None	YES	None	None
Timer 3	YES	YES	Width/Cycle	Width	YES	YES	12 bit

Note: Timer 2 on the SPMC65P2404A cannot operate as 8-bit compare mode.

5.8.2. Timer0

Timer0 is an 8-bit Timer. This timer can be set to operate in internal clock with pre-scalar or external clock input. When the timer's counter rollovers from 255 to 0, an overflow interrupt will be generated to set the Timer0 interrupt flag and the timer's counter register will be reloaded with pre-loaded value.

Figure 5.8-1 shows a simplified block diagram of the 8-bit timer. Figure 5.8-2 show the timer overflow and interrupt position.

The Timer0 module has the following features:

- Readable and writable
- Dedicated 8-stage pre-scalar timer
- Clock source selectable to be external or internal
- Interrupt-on-overflow from #$FF to #$00
- Supports 8-bit capture function
- Supports 8-bit compare function

[Example] 5.8.1 Formula for 8-bit timer0

$$f_{To} = \frac{F_{sys}}{Timer\_prescaler \times (256 - Timer\_preload\_value)}$$

f_{To} : Timer overflow frequency

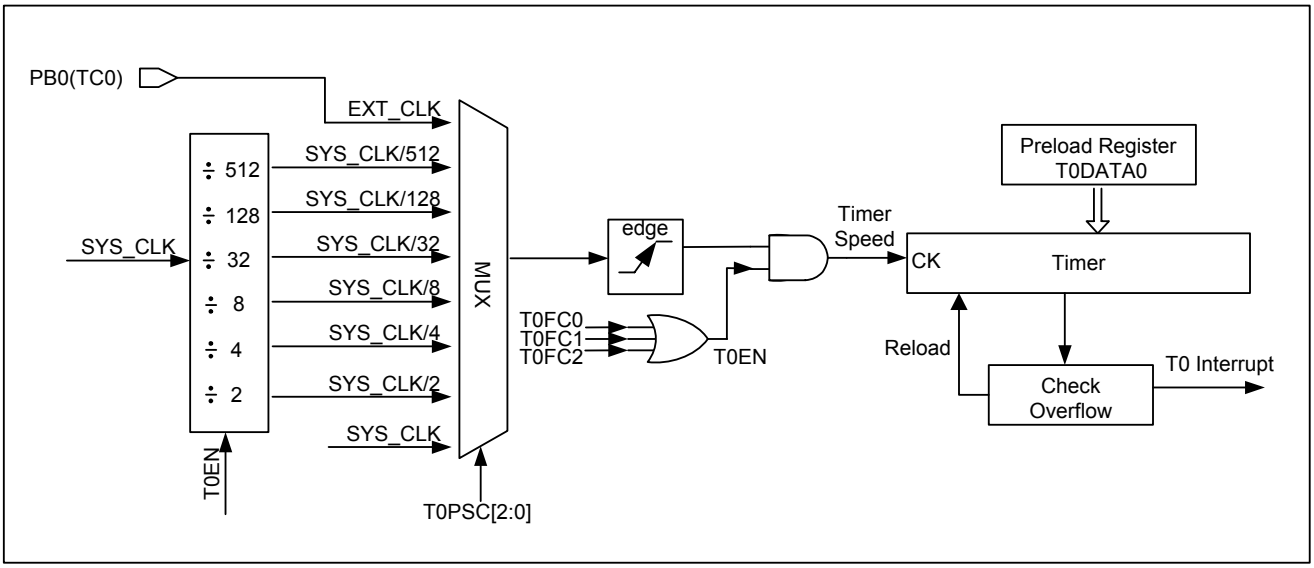


Figure 5.8-1 8-Bit timer block diagram

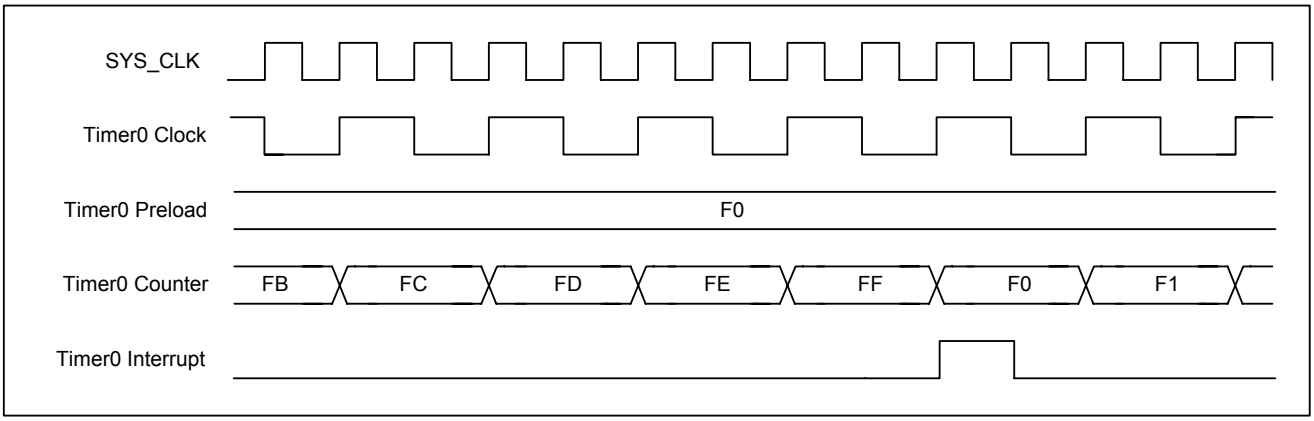


Figure 5.8-2 8-bit timer overflow

1). Timer0_1 Control Register0 (P_TMR0_1_Ctrl0, $0011)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	T1FC2	T1FC1	T1FC0	-	-	T0FC1	T0FC0
ACCESS	-	R/W	R/W	R/W	-	-	R/W	R/W
DEFAULT	0	0	0	0	0	0	0	0

Bit 7 Reserved

Bit [6:4] **T1FC**[2：0]: Timer 1 function configuration bits

111 = 12-bit PWM

110 = 16-bit capture (width)

101 = 16-bit compare

100 = 16-bit timer

011 = 8-bit capture (width, cycle)

010 = 8-bit compare

001 = 8-bit timer

000 = disable

Bit [2:3] Reserved

Bit [1:0] **T0FC**[1：0]: Timer0 function configuration bits

11 = 8-bit capture (width)

10 = 8-bit compare

01 = 8-bit timer

00 = disable

2). Timer0_1 Control Register1 (P_TMR0_1_Ctrl1, $0012)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	T1PSC2	T1PSC1	T1PSC0	-	T0PSC2	T0PSC1	T0PSC0
ACCESS	-	R/W	R/W	R/W	-	R/W	R/W	R/W
DEFAULT	0	0	0	0	0	0	0	0

Bit 7 Reserved

Bit [6:4] **T1PSC**[2：0]: Timer1 pre-scale configuration bits

111 = external event

110 = F_{SYS} ÷ 512

101 = F_{SYS} ÷ 128

100 = F_{SYS} ÷ 32

011 = F_{SYS} ÷ 8

010 = F_{SYS} ÷ 4

001 = F_{SYS} ÷ 2

000 = F_{SYS}

Bit 3 Reserved

Bit [2:0] **T0PSC**[2：0]: Timer0 pre-scale configuration bits

111 = external event

110 = F_{SYS} ÷ 512

101 = F_{SYS} ÷ 128

100 = F_{SYS} ÷ 32

011 = F_{SYS} ÷ 8

010 = F_{SYS} ÷ 4

001 = F_{SYS} ÷ 2

000 = F_{SYS}

F_{SYS}：Frequency of system

3). Timer0 Count Register (P_TMR0_Count, $0013)

		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
8-bit Timer	W	T0PLV_7	T0PLV_6	T0PLV_5	T0PLV_4	T0PLV_3	T0PLV_2	T0PLV_1	T0PLV_0
	R	T0R_7	T0R_6	T0R_5	T0R_4	T0R_3	T0R_2	T0R_1	T0R_0
8-bit Compare	W	T0COV_7	T0COV_6	T0COV_5	T0COV_4	T0COV_3	T0COV_2	T0COV_1	T0COV_0
	R	T0R_7	T0R_6	T0R_5	T0R_4	T0R_3	T0R_2	T0R_1	T0R_0
8-bit Capture	W	T0PLV_7	T0PLV_6	T0PLV_5	T0PLV_4	T0PLV_3	T0PLV_2	T0PLV_1	T0PLV_0
	R	T0CWV_7	T0CWV_6	T0CWV_5	T0CWV_4	T0CWV_3	T0CWV_2	T0CWV_1	T0CWV_0
DEFAULT		0	0	0	0	0	0	0	0

Bit [7:0] **P_TMR0_Count**[7:0]: Timer0 data register0

8-bit timer mode:

Write: Timer0 pre-load value T0PLV_[7:0]

Read: Timer0 register T0R_[7:0]

8-bit capture mode:

Write: Timer0 pre-load value T0PLV_[7:0]

Read: Timer0 capture width value T0CWV_[7:0]

8-bit compare mode:

Write: Timer0 compare value T0COV_[7:0]

Read: Timer0 register T0R_[7:0]

[Example] 5.8.2 Set Timer0 as 8-bit timer operation and generate 1ms overflow.

```
lda   #6                        ; Before starting timer, set Timer0 counter initial value first
sta   P_TMR0_Preload
lda   #C_T0FCS_Div_32           ; Set Timer0 clock source is Fsys/32
sta   P_TMR0_1_Ctrl1
lda   #C_T08B_Timer             ; Set Timer0 is 8-bit timer
sta   P_TMR0_1_Ctrl0
lda   #6                        ; Set Timer0 preload counter= 256-6= 250
sta   P_TMR0_Preload            ; Fsys(8MHz)/32/250= 1KHz(1ms)
```

[Example] 5.8.3 Set Timer0 as 8-bit compare operation

```
lda   #156                    ; Before starting timer, set Timer0 counter initial value first
sta   P_TMR0_Preload
lda   #C_T0FCS_Div_128        ; Set Timer0 clock source is Fsys/128
sta   P_TMR0_1_Ctrl1
lda   #C_T08B_COMP            ; Set Timer0 is 8-bit compare output
sta   P_TMR0_1_Ctrl0
lda   #156                    ; Set Timer0 preload counter= 256-156= 100
sta   P_TMR0_Preload          ; Fsys(8MHz)/128/100= 625Hz on PB2
```

5.8.3. Timer1

Timer1 can be used as a 16-bit timer or 8-bit timer. When it is used as an 8-bit timer, its function is the same as Timer0. If it is configured into 16-bit timer mode, a 16-bit timer counter is increased from reload value to 65535. When timer rollovers from 65535 to 0, it would trigger a timer overflow interrupt, and the timer's counter register will be reloaded with pre-load value.

Note that since this chip equips an 8-bit CPU and the data bus width is 8-bit, program cannot access 16-bit data simultaneously. In order to overcome this limitation in 16-bit timer/counter mode, the timer MSB (Most Significant Byte) data register is designed with extra read/write buffer, which is shown in Figure 5.8-3. Programmer must read the LSB byte first so that the MSB byte is buffered automatically. This buffered value remains unchanged until MSB byte is read. After program reads the MSB byte, the 16-bit read sequence is completed. Moreover, programmer must write the MSB byte first and then the LSB byte. The MSB byte will be buffered automatically. This buffered value remains unchanged until the LSB byte is written. The MSB byte and LSB byte will be written into corresponding registers simultaneously. After the LSB byte is written, the 16-bit write sequence is completed.

Moreover, Timer1 shares control register with Timer0. Please refer to Timer0's register for Timer1's control.

The Timer1 module has the following features:

- Readable and writable
- Dedicated 8-stage pre-scalar timer
- Clock source selectable to be external or internal
- Selectable 8-bit/16-bit timer mode
- Interrupt-on-overflow from #$FF to #$00 in 8-bit mode and #$FFFF to #$0000 in 16-bit mode
- Supports 8-bit/16-bit capture function
- Supports 8-bit/16-bit compare function
- Supports 12-bit PWM function

Figure 5.8-3 shows the 16-bit timer access method.
Figure 5.8-4 shows a simplified block diagram of the 16-bit timer.
Figure 5.8-5 shows the timer overflow and interrupt position.

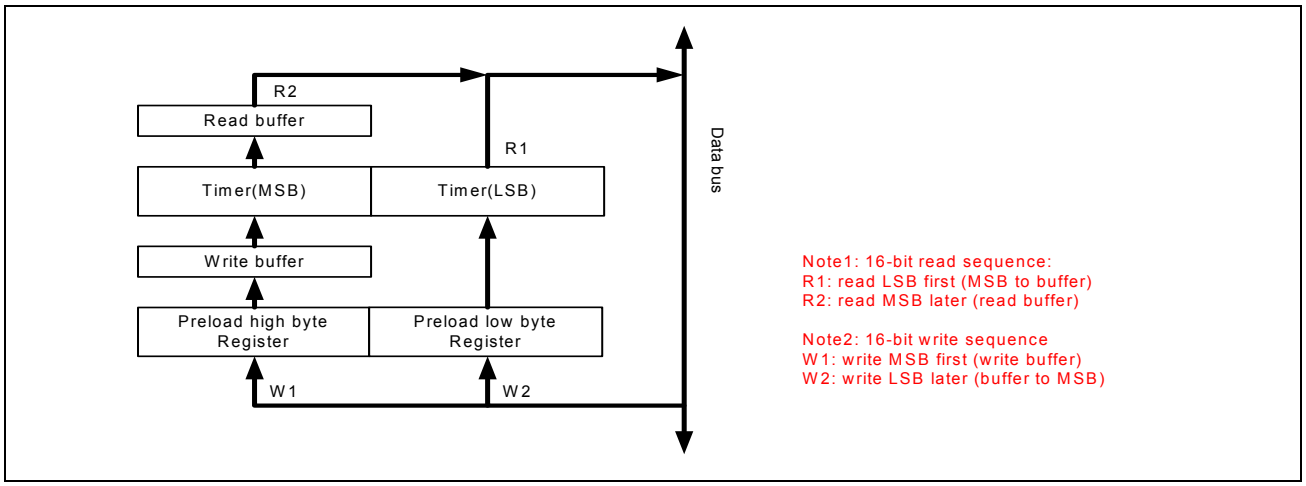


Figure 5.8-3 16-Bit timer access method diagram

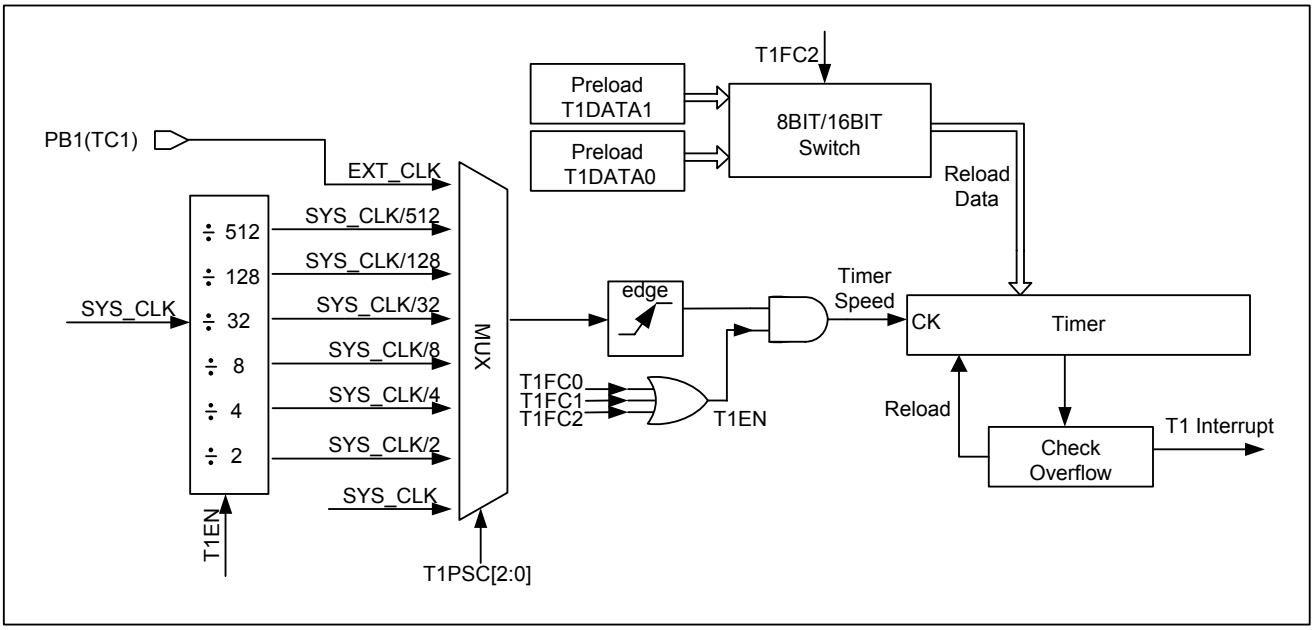


Figure 5.8-4 16-bit timer block diagram

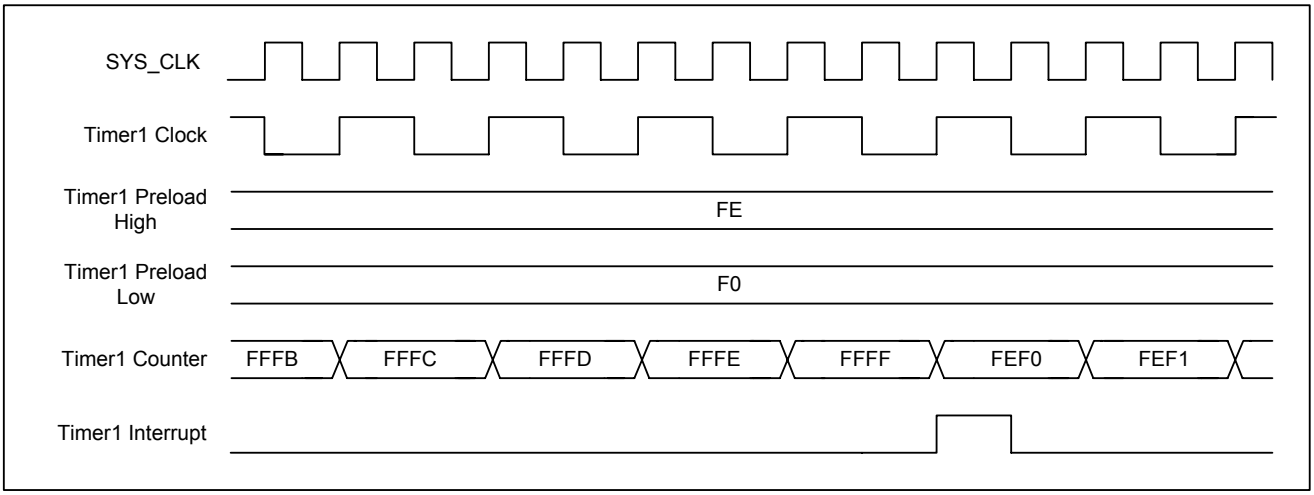


Figure 5.8-5 16-bit timer overflow

4). Timer1 Count Register (P_TMR1_Count, $0015)

Bit		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
8/16-bit Timer	W	T1PLV_7	T1PLV_6	T1PLV_5	T1PLV_4	T1PLV_3	T1PLV_2	T1PLV_1	T1PLV_0
	R	T1R_7	T1R_6	T1R_5	T1R_4	T1R_3	T1R_2	T1R_1	T1R_0
8/16-bit Compare	W	T1COV_7	T1COV_6	T1COV_5	T1COV_4	T1COV_3	T1COV_2	T1COV_1	T1COV_0
	R	T1R_7	T1R_6	T1R_5	T1R_4	T1R_3	T1R_2	T1R_1	T1R_0
8/16-bit Capture	W	T1PLV_7	T1PLV_6	T1PLV_5	T1PLV_4	T1PLV_3	T1PLV_2	T1PLV_1	T1PLV_0
	R	T1CWV_7	T1CWV_6	T1CWV_5	T1CWV_4	T1CWV_3	T1CWV_2	T1CWV_1	T1CWV_0
12-bit PWM	W	T1PPV_7	T1PPV_6	T1PPV_5	T1PPV_4	T1PPV_3	T1PPV_2	T1PPV_1	T1PPV_0
	R	T1PPV_7	T1PPV_6	T1PPV_5	T1PPV_4	T1PPV_3	T1PPV_2	T1PPV_1	T1PPV_0
DEFAULT		0	0	0	0	0	0	0	0

Bit [7:0] **P_TMR1_Count** [7:0]: Timer1 data register0

8/16-bit timer mode:

Write: Timer1 pre-load value T1PLV_[7:0]

Read: Timer1 register T1R_[7:0]

8/16-bit compare mode:

Write: Timer1 compare value T1COV_[7:0]

Read: Timer1 register T1R_[7:0]

8/16-bit capture mode:

Write: Timer1 pre-load value T1PLV_[7:0]

Read: Timer1 capture width value T1CWV_[7:0]

12-bit PWM mode:

Write: Timer1 PWM period value T1PPV_[7:0]

Read: Timer1 PWM period value T1PPV_[7:0]

5). Timer1 Count Register (P_TMR1_CountHi, $0016)

Bit		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
16-bit Timer	W	T1PLV_15	T1PLV_14	T1PLV_13	T1PLV_12	T1PLV_11	T1PLV_10	T1PLV_9	T1PLV_8
	R	T1R_15	T1R_14	T1R_13	T1R_12	T1R_11	T1R_10	T1R_9	T1R_8
16-bit Compare	W	T1COV_15	T1COV_14	T1COV_13	T1COV_12	T1COV_11	T1COV_10	T1COV_9	T1COV_8
	R	T1R_15	T1R_14	T1R_13	T1R_12	T1R_11	T1R_10	T1R_9	T1R_8
16-bit Capture	W	T1PLV_15	T1PLV_14	T1PLV_13	T1PLV_12	T1PLV_11	T1PLV_10	T1PLV_9	T1PLV_8
	R	T1CWV_15	T1CWV_14	T1CWV_13	T1CWV_12	T1CWV_11	T1CWV_10	T1CWV_9	T1CWV_8
8-bit Capture	W	T1PLV_15	T1PLV_14	T1PLV_13	T1PLV_12	T1PLV_11	T1PLV_10	T1PLV_9	T1PLV_8
	R	T1CCV_7	T1CCV_6	T1CCV_5	T1CCV_4	T1CCV_3	T1CCV_2	T1CCV_1	T1CCV_0

12-bit PWM	W	T1PDV_11	T1PDV_10	T1PDV_9	T1PDV_8	T1PPV_11	T1PPV_10	T1PPV_9	T1PPV_8
	R	T1PDV_11	T1PDV_10	T1PDV_9	T1PDV_8	T1PPV_11	T1PPV_10	T1PPV_9	T1PPV_8
DEFAULT		0	0	0	0	0	0	0	0

Bit [7:4] **P_TMR1_CountHi** [7:4]: Timer1 data register1

16-bit timer mode:

Write: Timer1 pre-load value T1PLV_[15:12]

Read: Timer1 register T1R_[15:12]

16-bit compare mode:

Write: Timer1 compare value T1COV_[15:12]

Read: Timer1 register T1R_[15:12]

16-bit capture mode:

Write: Timer1 pre-load value T1PLV_[15:12]

Read: Timer1 capture width value T1CWV_[15:12]

8-bit capture mode:

Write: Timer1 pre-load value T1PLV_[15:12]

Read: Timer1 capture cycle value T1CCV_[7:4]

12-bit PWM mode:

Write: Timer1 PWM duty value T1PDV_ [11:8]

Read: Timer1 PWM duty value T1PDV_ [11:8]

Bit [3:0] **P_TMR1_CountHi** [3:0]: Timer1 data register1

16-bit timer mode:

Write: Timer1 pre-load value T1PLV_[11:8]

Read: Timer1 register T1R_[11:8]

16-bit compare mode:

Write: Timer1 compare value T1COV_[11:8]

Read: Timer1 register T1R_[11:8]

16-bit capture mode:

Write: Timer1 pre-load value T1PLV_[11:8]

Read: Timer1 capture width value T1CWV_[11:8]

8-bit capture mode:

Write: Timer1 pre-load value T1PLV_[11:8]

Read: Timer1 capture cycle value T1CCV_[3:0]

12-bit PWM mode:

Write: Timer1 PWM period value T1PPV_ [11:8]

Read: Timer1 PWM period value T1PPV_ [11:8]

6). Timer1 Low Byte Duty Register (P_TMR1_PWMDuty, $0017)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	T1PDV_7	T1PDV_6	T1PDV_5	T1PDV_4	T1PDV_3	T1PDV_2	T1PDV_1	T1PDV_0
ACCESS	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
DEFAULT	0	0	0	0	0	0	0	0

Bit [7:0] **P_TMR1_PWMDuty** [7:0]: Timer1 data register2

12-bit PWM mode:

Write: Timer1 PWM duty value T1PDV_[7:0]

Read: Timer1 PWM duty value T1PDV_[7:0]

[Example] 5.8.4 Set Timer1 as 16-bit timer operation and generate 10ms overflow.

```
    lda   #$FD                          ; Before starting timer, set Timer1 counter initial value first
    sta   P_TMR1_PreloadHi              ; Set high byte initial value first
    lda   #143
    sta   P_TMR1_Preload                ; Set low byte initial value later
    lda   #C_T1FCS_Div_128              ; Set Timer1 clock source is Fsys/128
    sta   P_TMR0_1_Ctrl1
    lda   #C_T116B_Timer                ; Set Timer1 is 16-bit timer
    sta   P_TMR0_1_Ctrl0

    lda   #$FD                        ; 1'st set Timer1 preload high byte counter= 2x 256= 512
    sta   P_TMR1_PreloadHi
    lda   #143                          ; 2'nd set Timer1 preload low byte counter= 256-143= 113
    sta   P_TMR1_Preload                ; Fsys(8Mhz)/128/625= 100Hz(10ms)
```

5.8.4. Timer2

In SPMC65P2408, Timer2 has same function as Timer0. In SPMC65P2404A, it does not provide compare function for Timer2. In SPMC65P2404A, the timer2 has same function as Timer0 except the compare function.

The Timer2 module has the following features:

- Readable and writable
- Dedicated 8-stage pre-scalar timer
- Clock source selectable to be external or internal
- Interrupt-on-overflow from #$FF to #$00
- Supports 8-bit capture function
- Supports 8-bit compare function (SPMC65P2408A only)

7). Timer2_3 Control Register0 (P_TMR2_3_Ctrl0, $0018)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	T3FC2	T3FC1	T3FC0	-	-	T2FC1	T2FC0
ACCESS	-	R/W	R/W	R/W	-	-	R/W	R/W
DEFAULT	0	0	0	0	0	0	0	0

Bit 7 Reserved

Bit [6:4] **T3FC** [2:0]: Timer3 function configuration bits

111 = 12-bit PWM

110 = 16-bit capture (width)

101 = 16-bit compare

100 = 16-bit timer

011 = 8-bit capture (width, cycle)

010 = 8-bit compare

001 = 8-bit timer

000 = disable

Bit [3:2] Reserved

Bit [1:0] **T2FC**[1：0]: Timer2 function configuration bits

11 = 8-bit capture (width)

10 = 8-bit compare (2404A reserved, 2408A only)

01 = 8-bit timer

00 = disable

8). Timer2_3 Control Register1 (P_TMR2_3_Ctrl1, $0019)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	T3PSC2	T3PSC1	T3PSC0	-	T2PSC2	T2PSC1	T2PSC0
ACCESS	-	R/W	R/W	R/W	-	R/W	R/W	R/W
DEFAULT	0	0	0	0	0	0	0	0

Bit 7 Reserved

Bit [6:4] **T3PSC**[2：0]: Timer3 pre-scale configuration bits

111 = external event

110 = F_{SYS} ÷ 512

101 = F_{SYS} ÷ 128

100 = F_{SYS} ÷ 32

011 = F_{SYS} ÷ 8

010 = F_{SYS} ÷ 4

001 = F_{SYS} ÷ 2

000 = F_{SYS}

Bit 3 Reserved

Bit [2:0] **T2PSC**[2:0]: Timer2 pre-scale configuration bits

111 = external event

110 = F_{SYS} ÷ 512

101 = F_{SYS} ÷ 128

100 = F_{SYS} ÷ 32

011 = F_{SYS} ÷ 8

010 = F_{SYS} ÷ 4

001 = F_{SYS} ÷ 2

000 = F_{SYS}

F_{SYS}: frequency of clock source

9). Timer2 Count Register (P_TMR2_Count, $001A)

Bit		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
8-bit	W	T2PLV_7	T2PLV_6	T21PLV_5	T2PLV_4	T2PLV_3	T2PLV_2	T2PLV_1	T2PLV_0
Timer	R	T2R_7	T2R_6	T2R_5	T2R_4	T2R_3	T2R_2	T2R_1	T2R_0

8-bit Capture	W	T2PLV_7	T2PLV_6	T2PLV_5	T2PLV_4	T2PLV_3	T2PLV_2	T2PLV_1	T2PLV_0
	R	T2CWV_7	T2CWV_6	T2CWV_5	T2CWV_4	T2CWV_3	T2CWV_2	T2CWV_1	T2CWV_0
DEFAULT		0	0	0	0	0	0	0	0

Bit [7:0] **P_TMR2_Count** [7:0]: Timer2 data register0

8-bit timer mode:

Write: Timer1 pre-load value T2PLV_[7:0]

Read: Timer1 register T2R_[7:0]

8-bit Capture mode:

Write: Timer1 pre-load value T2PLV_[7:0]

Read: Timer1 capture width value T2CWV_[7:0]

5.8.5. Timer3

Timer3 has exactly same function as Timer1. It also shares the control registers with Timer2. Please refer to Timer2's control registers for it.

It has the following features:

- Readable and writable
- Dedicated 8-stage pre-scalar timer
- Clock source selectable to be external or internal
- Selectable 8-bit/16-bit timer mode
- Interrupt-on-overflow from #$FF to #$00 in 8-bit mode and #$FFFF to #$0000 in 16-bit mode
- Supports 8-bit/16-bit capture function
- Supports 8-bit/16-bit compare function
- Supports 12-bit PWM function

10). Timer3 Count Register (P_TMR3_ Count, $001C)

Bit		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
8/16-bit Timer	W	T3PLV_7	T3PLV_6	T3PLV_5	T3PLV_4	T3PLV_3	T3PLV_2	T3PLV_1	T3PLV_0
	R	T3R_7	T3R_6	T3R_5	T3R_4	T3R_3	T3R_2	T3R_1	T3R_0
8/16-bit Compare	W	T3COV_7	T3COV_6	T3COV_5	T3COV_4	T3COV_3	T3COV_2	T3COV_1	T3COV_0
	R	T3R_7	T3R_6	T3R_5	T3R_4	T3R_3	T3R_2	T3R_1	T3R_0
8/16-bit Capture	W	T3PLV_7	T3PLV_6	T3PLV_5	T3PLV_4	T3PLV_3	T3PLV_2	T3PLV_1	T3PLV_0
	R	T3CWV_7	T3CWV_6	T3CWV_5	T3CWV_4	T3CWV_3	T3CWV_2	T3CWV_1	T3CWV_0
12-bit PWM	W	T3PPV_7	T3PPV_6	T3PPV_5	T3PPV_4	T3PPV_3	T3PPV_2	T3PPV_1	T3PPV_0
	R	T3PPV_7	T3PPV_6	T3PPV_5	T3PPV_4	T3PPV_3	T3PPV_2	T3PPV_1	T3PPV_0
DEFAULT		0	0	0	0	0	0	0	0

Bit [7:0] **P_TMR3_ Count** [7:0]: Timer3 data register0

8/16-bit timer mode:

Write: Timer3 pre-load value T3PLV_[7:0]

Read: Timer3 register T3R_[7:0]

8/16-bit compare mode:

Write: Timer3 compare value T3COV_[7:0]

Read: Timer3 register T3R_[7:0]

8/16-bit capture mode:

Write: Timer3 pre-load value T3PLV_[7:0]

Read: Timer3 capture width value T3CWV_[7:0]

12-bit PWM mode:

Write: Timer3 PWM period value T3PPV_[7:0]

Read: Timer3 PWM period value T3PPV_[7:0]

11). Timer3 Count Register (P_TMR3_ CountHi, $001D)

Bit		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
16-bit Timer	W	T3PLV_15	T3PLV_14	T3PLV_13	T3PLV_12	T3PLV_11	T3PLV_10	T3PLV_9	T3PLV_8
	R	T3R_15	T3R_14	T3R_13	T3R_12	T3R_11	T3R_10	T3R_9	T3R_8

16-bit Compare	W	T3COV_15	T3COV_14	T3COV_13	T3COV_12	T3COV_11	T3COV_10	T3COV_9	T3COV_8
	R	T3R_15	T3R_14	T3R_13	T3R_12	T3R_11	T3R_10	T3R_9	T3R_8
16-bit Capture	W	T3PLV_15	T3PLV_14	T3PLV_13	T3PLV_12	T3PLV_11	T3PLV_10	T3PLV_9	T3PLV_8
	R	T3CWV_15	T3CWV_14	T3CWV_13	T3CWV_12	T3CWV_11	T3CWV_10	T3CWV_9	T3CWV_8
8-bit Capture	W	T3PLV_15	T3PLV_14	T3PLV_13	T3PLV_12	T3PLV_11	T3PLV_10	T3PLV_9	T3PLV_8
	R	T3CCV_7	T3CCV_6	T3CCV_5	T3CCV_4	T3CCV_3	T3CCV_2	T3CCV_1	T3CCV_0
12-bit PWM	W	T3PDV_11	T3PDV_10	T3PDV_9	T3PDV_8	T3PPV_11	T3PPV_10	T3PPV_9	T3PPV_8
	R	T3PDV_11	T3PDV_10	T3PDV_9	T3PDV_8	T3PPV_11	T3PPV_10	T3PPV_9	T3PPV_8
DEFAULT		0	0	0	0	0	0	0	0

Bit [7:4] **P_TMR3_ CountHi** [7:4]: Timer3 data register1

16-bit timer mode:

Write: Timer3 pre-load value T3PLV_[15:12]

Read: Timer3 register T3R_[15:12]

16-bit compare mode:

Write: Timer3 compare value T3COV_[15:12]

Read: Timer3 register T3R_[15:12]

16-bit capture mode:

Write: Timer3 pre-load value T3PLV_[15:12]

Read: Timer3 capture width value T3CWV_[15:12]

8-bit capture mode:

Write: Timer3 pre-load value T3PLV_[15:12]

Read: Timer3 capture cycle value T3CCV_[15:12]

12-bit PWM mode:

Write: Timer3 PWM duty value T3PDV_ [11:8]

Read: Timer3 PWM duty value T3PDV_ [11:8]

Bit [3:0] **P_TMR3_ CountHi** [3:0]: Timer3 data register1

16-bit timer mode:

Write: Timer1 pre-load value T3PLV_[11:8]

Read: Timer3 register T3R_[11:8]

16-bit compare mode:

Write: Timer3 compare value T3COV_[11:8]

Read: Timer3 register T3R_[11:8]

16-bit capture mode:

Write: Timer3 pre-load value T3PLV_[11:8]

Read: Timer3 capture width value T3CWV_[11:8]

8-bit capture mode:

Write: Timer3 pre-load value T3PLV_[11:8]

Read: Timer3 capture cycle value T3CCV_[11:8]

12-bit PWM mode:

Write: Timer3 PWM period value T3PPV_ [11:8]

Read: Timer3 PWM period value T3PPV_ [11:8]

12). Timer3 Count Register (P_TMR3_PWMDuty, $001E)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	T3PDV_7	T3PDV_6	T3PDV_5	T3PDV_4	T3PDV_3	3PDV_2	T3PDV_1	T3PDV_0
ACCESS	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
DEFAULT	0	0	0	0	0	0	0	0

Bit [7:0] **P_TMR3_PWMDuty** [7:0]: Timer3 data register2
12-bit PWM mode:
Write: Timer3 PWM duty value T3PDV_[7:0]
Read: Timer3 PWM duty value T3PDV_[7:0]

5.9. Capture/Compare/PWM (CCP) Function

SPMC65P2404A/2408A provides powerful CCP functions. These CCP functions are combined with timer modules. Within different length of the timer's counter bits, each timer of SPMC65P2404A/2408A provides different CCP functions. In general, there are five CCP modes - 8-bit compare mode, 16-bit compare mode, 8-bit capture mode, 16-bit capture mode, and 12-bit PWM mode.

[Table] 5.9.1 CCP mode – timer resource

Function	Timer Resource
8-bit compare	Timer0, Timer1, Timer3, Timer2 (2408 only)
16-bit compare	Timer1, Timer3
8-bit capture	Timer0, Timer1, Timer2, Timer3
16-bit capture	Timer1, Timer3
12-bit PWM	Timer1, Timer3

5.9.1. 8-Bit Compare Mode

The 8-bit compare mode is supported by Timer0, Timer1, Timer2 (2408 only), and Timer3. Take Timer0 as an example, when Timer0 is set to 8-bit compare mode, the compare data can be set by P_TMR0_Comp ($0013). Timer0 uses the compare data as the starting number and generates the overflow interrupt when timer goes overflow. The compare module toggles the output level of PB2 when the overflow interrupt is generated. This sequence will keep running till the timer is stopped. In other words, the duty of compare output is fixed to 50% and the frequency of the compare output is half of the timer overflow rate. The compare operation is shown in figure 5.9-1.

[Example] 5.9.1 Formula for 8-bit compare output

$$f_{Comp} = \frac{F_{sys}}{Timer\_prescaler \times (256 - Timer\_Compare\_value)} \times \frac{1}{2}$$

f_{Comp} : Compare output frequency

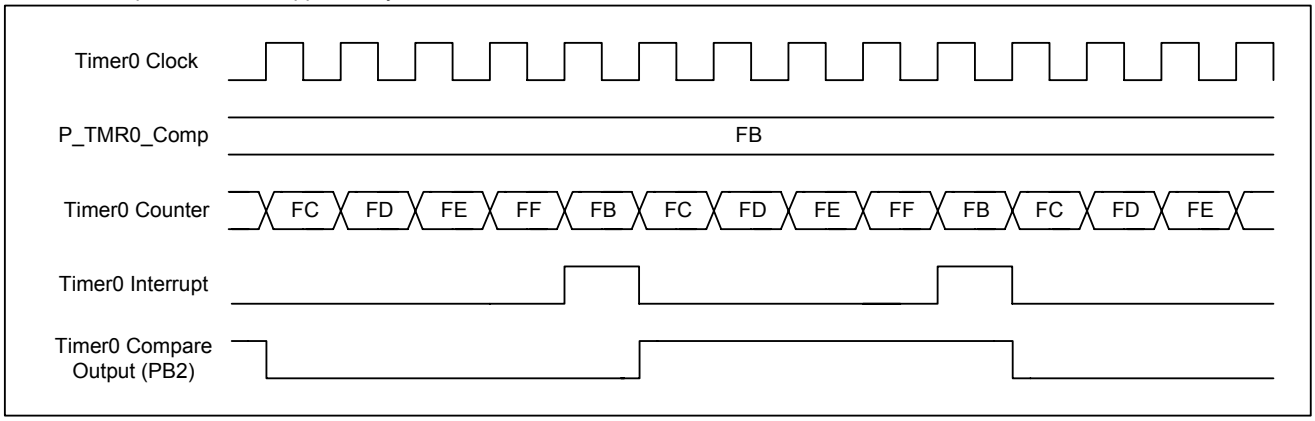


Figure 5.9-1 8-bit compare waveform

5.9.2. 16-Bit Compare Mode

The 16-bit compare mode is supported by Timer1 and Timer3. Using Timer1 as an example, the compare data can be set by P_TMR1_Comp ($0015) and P_TMR1_CompHi ($0016) when Timer1 is set to 16-bit compare mode. The Timer1 uses the compare data as the starting number, and generates the overflow interrupt when timer goes overflow. The compare module toggles the output level of PB3 when the overflow interrupt is generated. This sequence will keep running till the timer is stopped. In other words, the duty of compare output is fixed to 50% and the

frequency of the compare output is half of the timer overflow rate.
The compare operation is shown in figure 5.9-2.

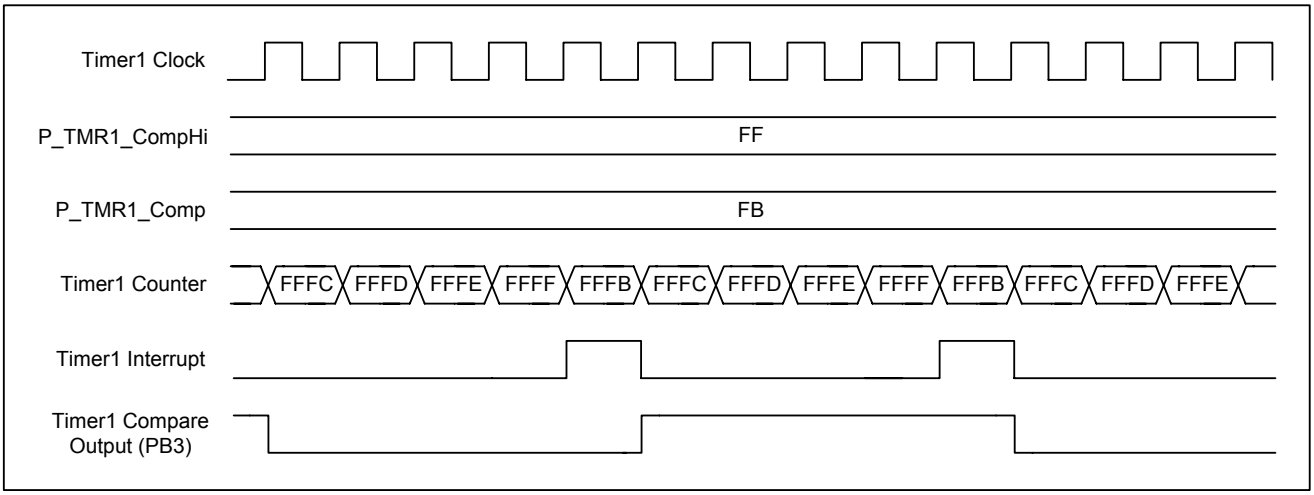


Figure 5.9-2 16-bit compare Waveform

[Example] 5.9.2 Set Timer1 as 16-bit compare operation and generate compared output

```
lda   #$FD                          ; Before starting timer, set Timer1 counter initial value first
sta   P_TMR1_PreloadHi              ; Set high byte initial value first
lda   #143
sta   P_TMR1_Preload                ; Set low byte initial value later
lda   #C_T1FCS_Div_128              ; Set Timer1 clock source is Fsys/128
sta   P_TMR0_1_Ctrl
lda   #C_T116B_COMP                 ; Set Timer1 is 16-bit compare output
sta   P_TMR0_1_Ctrl0
lda   #$FD                  ; 1'st set Timer1 preload high byte counter= 2x 256= 512
sta   P_TMR1_PreloadHi
lda   #143                          ; 2'nd set Timer1 preload low byte counter= 256-143= 113
sta   P_TMR1_Preload                ; Fsys(8Mhz)/128/625= 100Hz(10ms)on PB3
```

5.9.3. 8-Bit Capture Mode

All of the four timers in SPMC65P2404A/2408A can be used for 8-bit capture mode. When timer is in an 8-bit capture mode, Capture input pin must be set as input mode and then acts as the capture event input. The capture polarity and interrupt evoke polarity are selected by capture control register (P_CAP_Ctrl, $0058).

There is still a difference in 8-bit Capture mode between Timer0/Timer2 and Timer1/Timer3. The Time0 and Timer2 are 8-bit timers, so they can only be used for pulse width measurement since they only have one byte of register to store capture data. The Timer1 and Timer3 are 16-bit timers so that they can be used to measure both width and cycle of a pulse. The 8-bit capture operation is shown in figure 5.9-3.

The capture module also provides a hold function for programmer to catch input signal once. By setting the P_CAP_Ctrl.CAPOPT to 1, the input signal will only be captured once. After first capture, the data will be held till programmer reading them out. Once programmer reads the capture data, the input signal will be captured and hold again. This feature can be used to measure single pulse width/cycle.

[Example] 5.9.3 Formula for 8-bit capture using Timer0

```
Capture Width=(P_TMR0_Cap +1) x Timer Pre-scale
```

[Example] 5.9.4 Formula for 8-bit capture using Timer1

```
Capture Width=(P_TMR1_Cap +1) x Timer Pre-scale
Capture Cycle=(P_TMR1_CapCycle8 +1) x Timer Pre-scale
```

13). Capture Control Register (P_CAP_Ctrl, $0058)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	CAPOPT	-	CAPIP3	CAP1P2	CAPIP1	CAP1P0	CAP1ES	CAP0ES
ACCESS	R/W	-	R/W	R/W	R/W	R/W	R/W	R/W
DEFAULT	0	0	0	0	0	0	0	0

Bit 7: **CAPOPT**: Capture data hold option bit.
1 = hold capture data
0 = update captured date if new data is received

Bit 6: Reserved

Bit 5: **CAPIP3**: Capture3 interrupt evoke polarity selection bit
1 = be same with CAP3ES edge setting
0 = be opposite to CAP3ES edge setting

Bit 4: **CAPIP2**: Capture2 interrupt evoke polarity selection bit
1 = be same with CAP2ES edge setting
0 = be opposite to CAP2ES edge setting

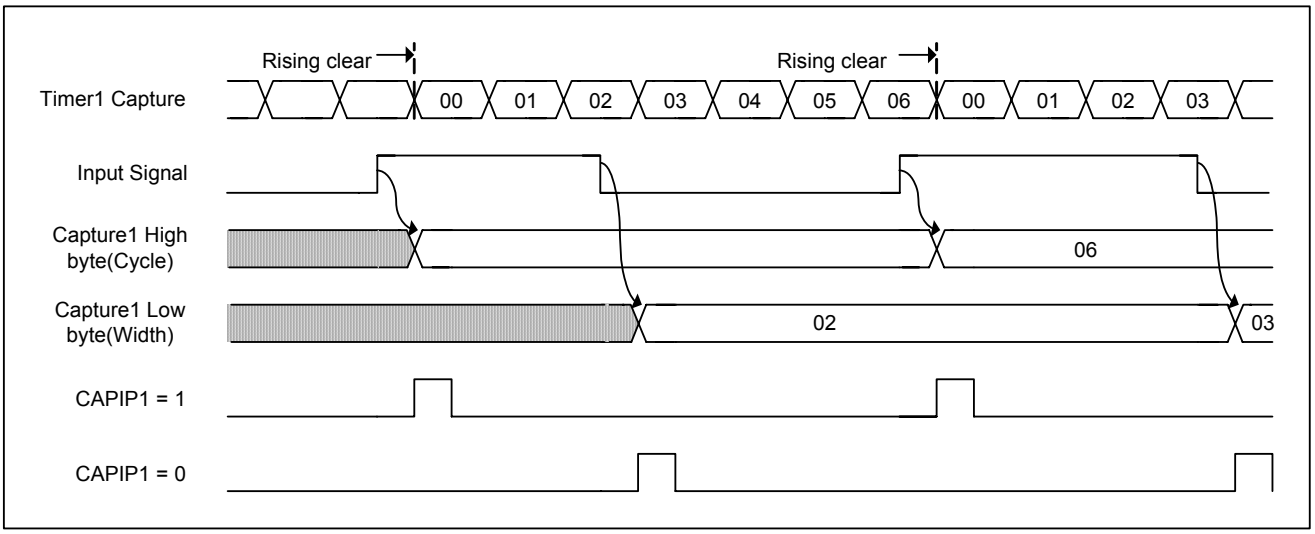
Bit 3: **CAPIP1**: Capture1 interrupt evoke polarity selection bit
1 = be same with CAP1ES edge setting
0 = be opposite to CAP1ES edge setting

Bit 2: **CAPIP0**: Capture0 interrupt evoke polarity selection bit
1 = be same with CAP0ES edge setting
0 = be opposite to CAP0ES edge setting

Bit 1: **CAP1ES**: Polarity edge selection bit in Capture1 of Timer1
1 =falling edge clear counter
0 =rising edge clear counter

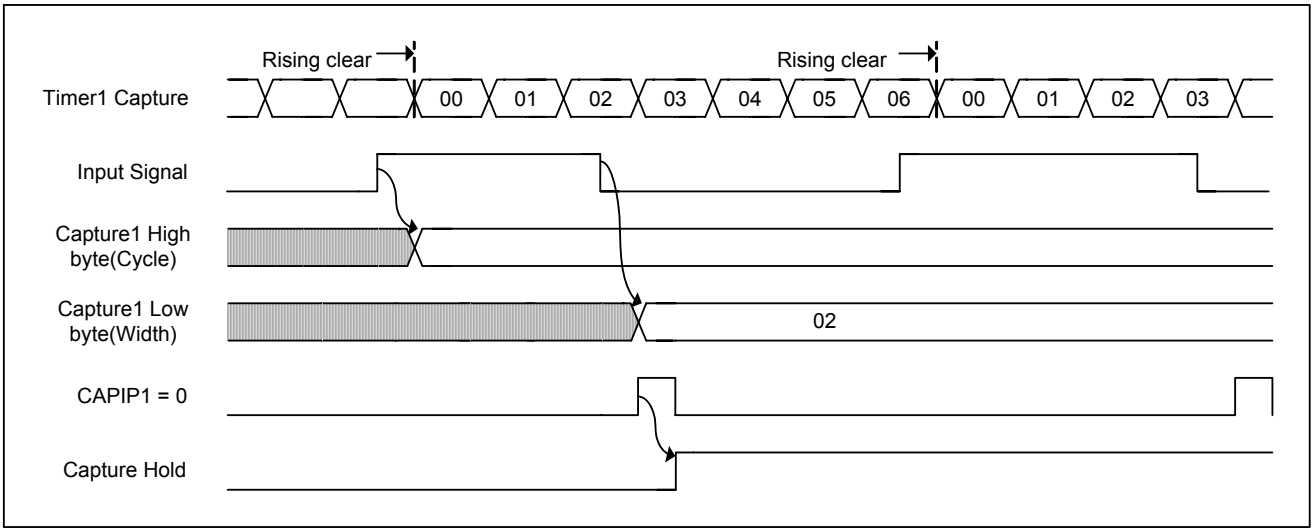
Bit 0: **CAP0ES**: Polarity edge selection bit in Capture0 of Timer0
1 = falling edge clear counter
0 = rising edge clear counter

Note: CAP3ES and CAP2ES are combined with IRQ option in $34 since they share input pin with IRQ1 and IRQ0. Please refer to Section 5.5 Interrupt for this setting.



Note : Timer0/Timer2 does not have capture high byte

Figure 5.9-3 8-Bit capture waveform



Note : Timer0/Timer2 does not have capture high byte

Figure 5.9-4 8-Bit capture and hold waveform

[Example] 5.9.5 Set Timer0 as 8-bit capture operation.

```
lda   #C_T0FCS_Div_512              ; Set Timer0 clock source is Fsys/512(15.6KHz~61Hz measurement)
sta   P_TMR0_1_Ctrl1
lda   #C_T08B_CAP                   ; Set Timer0 is 8-bit capture
sta   P_TMR0_1_Ctrl0

lda   #(C_CAP_IP0+C_CAP0_ES)        ; Falling edge sample data, falling edge CAP0 int evoke.
sta   P_CAP_Ctrl                    ; => Input pulse low width measurement on PB0
```

5.9.4. 16-bit Capture Mode

The Timer1 and Timer3 support 16-bit wide capture mode. The 16-bit capture operation is the same as 8-bit capture, except timer registers can be counted to 16 bit-wide and only the pulse width can be measured.

[Example] 5.9.6 Formula for 16-bit capture using Timer1

```
Capture Width=((P_TMR1_CapHi, P_TMR1_Cap)+1) x Timer Pre-scale
```

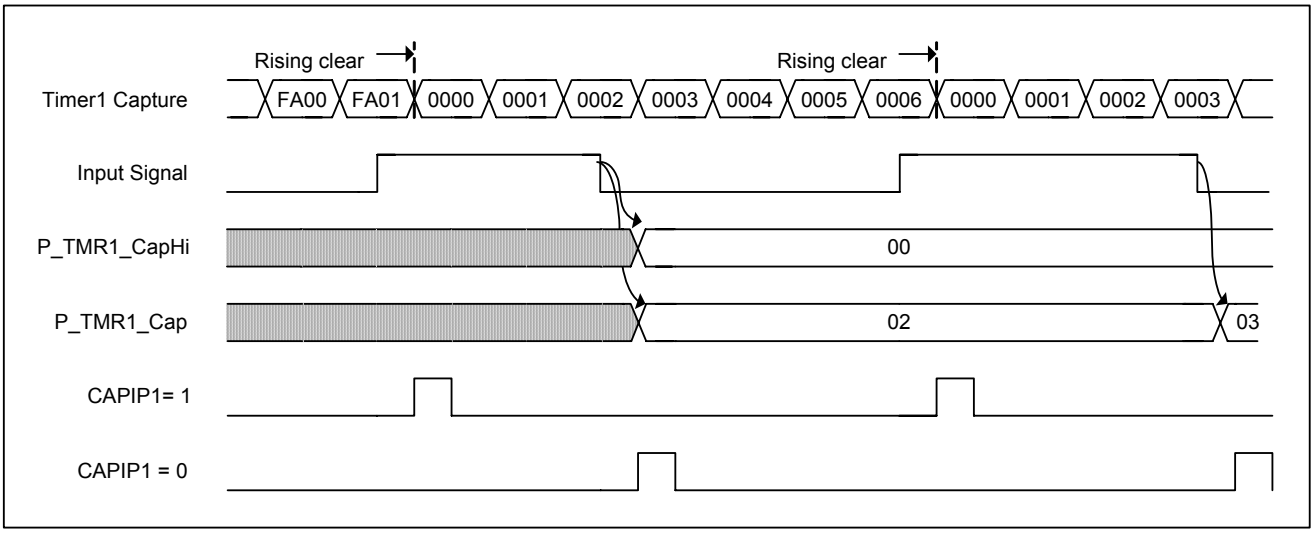


Figure 5.9-5 16-bit capture waveform

[Example] 5.9.7 Set Timer1 as 16-bit capture operation.

```
lda   #C_T1FCS_Div_128              ; Set Timer1 clock source is Fsys/128(62.5KHz~1Hz measurement)
sta   P_TMR0_1_Ctrl1
lda   #C_T116B_CAP                  ; Set Timer1 is 16-bit capture
sta   P_TMR0_1_Ctrl0

lda   (C_CAP_IP1+C_CAP1_ES)         ; Falling edge sample data, falling edge CAP1 int evoke.
sta   P_CAP_Ctrl                    ; => Input pulse low width measurement on PB1
```

5.9.5. 12-bit PWM Mode

SPMC65P2404A/2408A provides two high-speed PWM (Pulse Width Modulation) modules that are shared with Timer1 and Timer3. Here we will take timer1 as an example to explain how to set registers in PWM mode. When Timer1 function configuration registers are set as PWM, the PB2 will become the PWM output pin automatically. The PWM module can output a 12-bit resolution PWM waveform. The registers for setting PWM period are composed of the value in P_TMR1_DutyPeriod[3:0] and P_TMR1_PWM Period[7:0] and the PWM duty is composed of the value in P_TMR1_DutyPeriod[7:4] and P_TMR1_PWMDuty [7:0]. The higher 4-bit period value P_TMR1_DutyPeriod[3:0] should be written first before writing the lower 8-bit period value to P_TMR1_PWMPeriod[7:0]. So it is for the setting of duty value, programmer has to set P_TMR1_DutyPeriod[7:4] first before setting P_TMR1_PWMDuty[7:0]. This way is the same as the 16-bit timer writing method.

The following table shows the relation of resolution and frequency. It supports programmer to choose PWM frequency with suit resolution.

[Table] 5.9.2 Resolution vs. PWM frequency

Resolution	PWM Frequency		
	Pre-Scale= = F_{sys} ÷ 1	Pre-Scale= = F_{sys}÷ 2	Pre-Scale= = F_{sys}÷ 4
12-Bit operation	1.95KHZ	975HZ	487HZ
11-Bit operation	3.9 KHZ	1.95KHZ	975HZ
10-Bit operation	7.8 KHZ	3.9 KHZ	1.95KHZ
9-Bit operation	15.6 KHZ	7.8 KHZ	3.9 KHZ

Note: System operation at 8MHz

[Example] 5.9.8 Formula for 12-bit PWM using Timer1

PWM Period = (P_TMR1_DutyPeriod [3:0]: P_TMR1_PWMPeriod [7:0]) x Timer Pre-scale

PWM Duty = (P_TMR1_DutyPeriod [7:4]: P_TMR1_PWMDuty [7:0]) x Timer Pre-scale

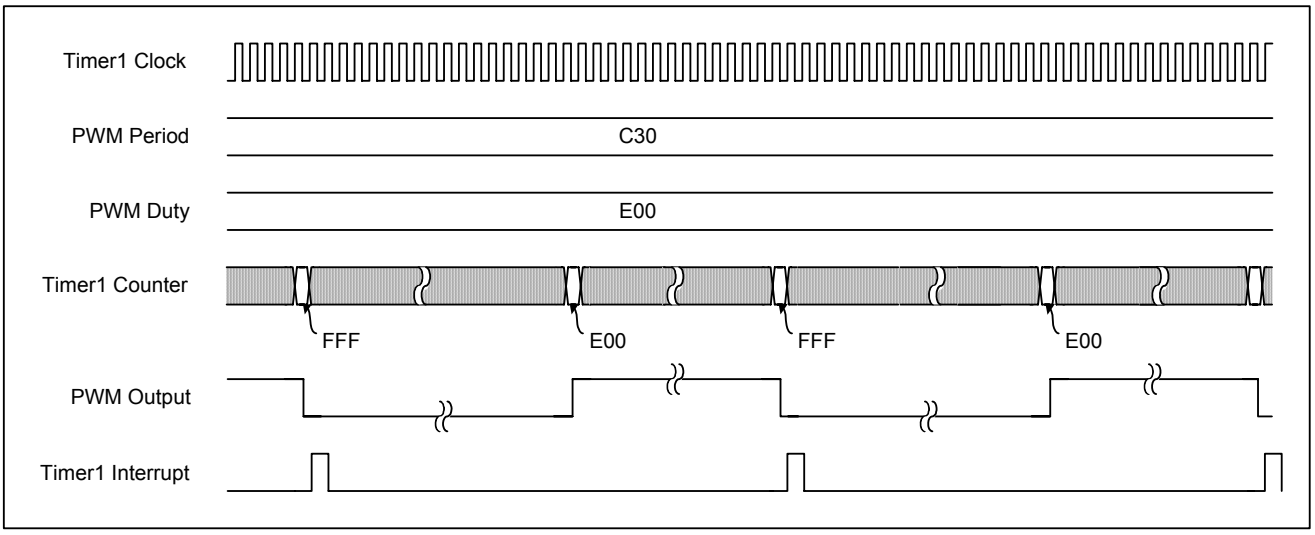


Figure 5.9-6 12-Bit PWM operating waveform

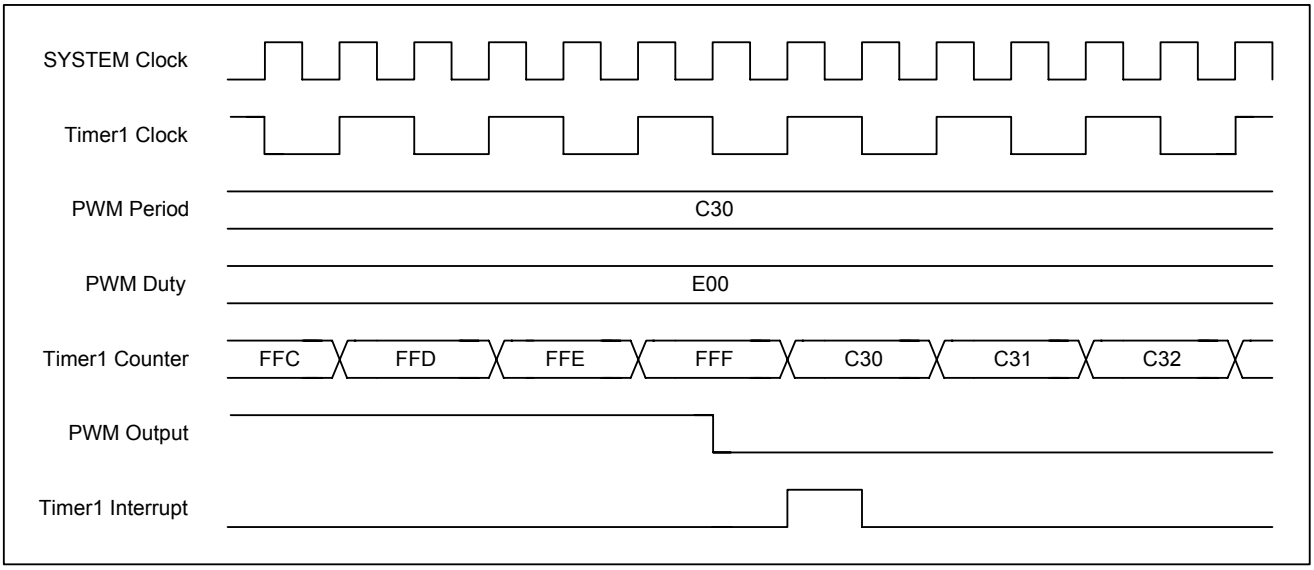


Figure 5.9-7 12-Bit PWM reload waveform

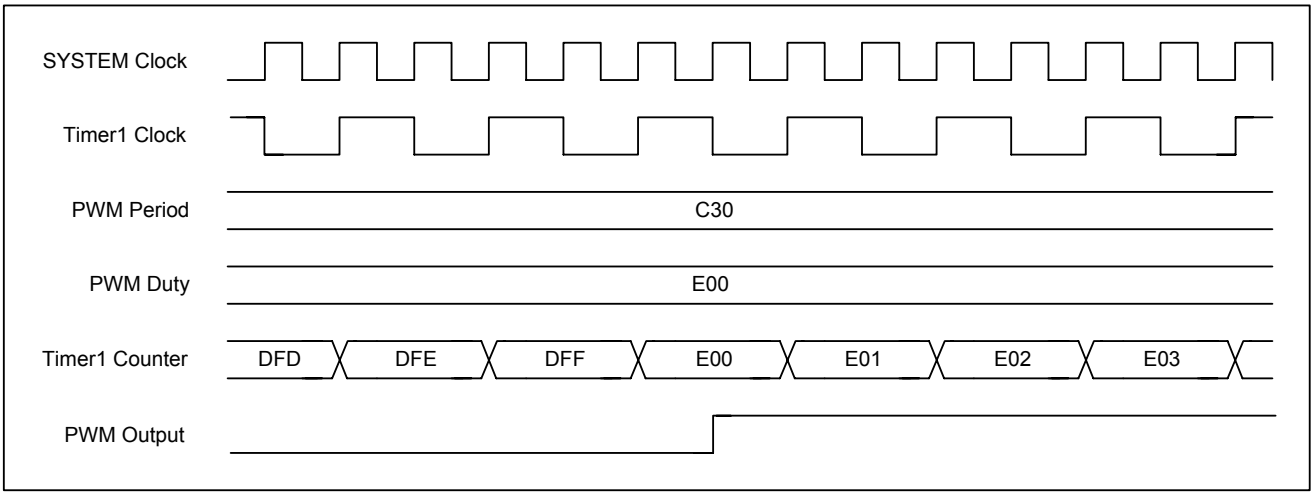


Figure 5.9-8 12-Bit PWM transient waveform

[Example] 5.9.9 Set Timer1 as 12-bit PWM operation and generate PWM output on PB3.

```
    lda   #$70                      ; Before starting timer, set Timer1 counter initial value first
    sta   P_TMR1_DutyPeriod         ; Set duty value
    lda   #$00
    sta   P_TMR1_PWMPeriod          ; Set Period value
    lda   #$FF
    sta   P_TMR1_PWMDuty
    lda   #C_T1FCS_Div_8            ; Set Timer1 clock source is Fsys/8
    sta   P_TMR0_1_Ctrl1
    lda   #C_T112B_PWM              ; Set Timer1 is 12-bit PWM
    sta   P_TMR0_1_Ctrl0


    lda   #$70
```

```
sta   P_TMR1_DutyPeriod          ; PWM duty $7FF, PWM period $1000
lda   #$00
sta   P_TMR1_PWMPeriod
lda   #$FF
sta   P_TMR1_PWMDuty             ; PWM output 244Hz 50% duty ratio on PB3
```

5.10. Analog Module

5.10.1. A/D Converter

5.10.1.1. Introduction

SPMC65P2404A/2408A is embedded with a 10-bit 8-channel ADC. It is used for many applications such as touch panel, battery power detection, and etc. For speech record, an external AGC is needed. The channel inputs of ADC are shared with PA bit7-bit0 , PB bit7 is shared with top reference voltage.

P_AD_Ctrl0, P_AD_Ctrl1 and P_AD_Ctrl2 are the control registers for A/D converter. The P_AD_Ctrl0 controls the setting of A/D converter module. The analog reference voltage can be selected from external voltage input on PB7 or internal voltage VDD via ADVRT bit in P_AD_Ctrl0 register. The converter speed of the ADC can be selected from P_AD_Ctrl0. Be careful to choose A/D clock speed, it should be lower than 1.4MHZ to satisfy A/D hardware requirement. P_AD_Ctrl1 is used to configure PA[7:0] pin as analog pin for A/D or digital pin for I/O. Programmer has to set P_AD_Ctrl1 first before using the I/O port as A/D input pin; otherwise, the A/D value will be incorrect. The P_AD_Ctrl2 is used to select current channel for conversion.

The processing of conversion is started when the ADRDY bit is written to “0”. When the conversion is success, the ADRDY bit will set to “1” automatically.

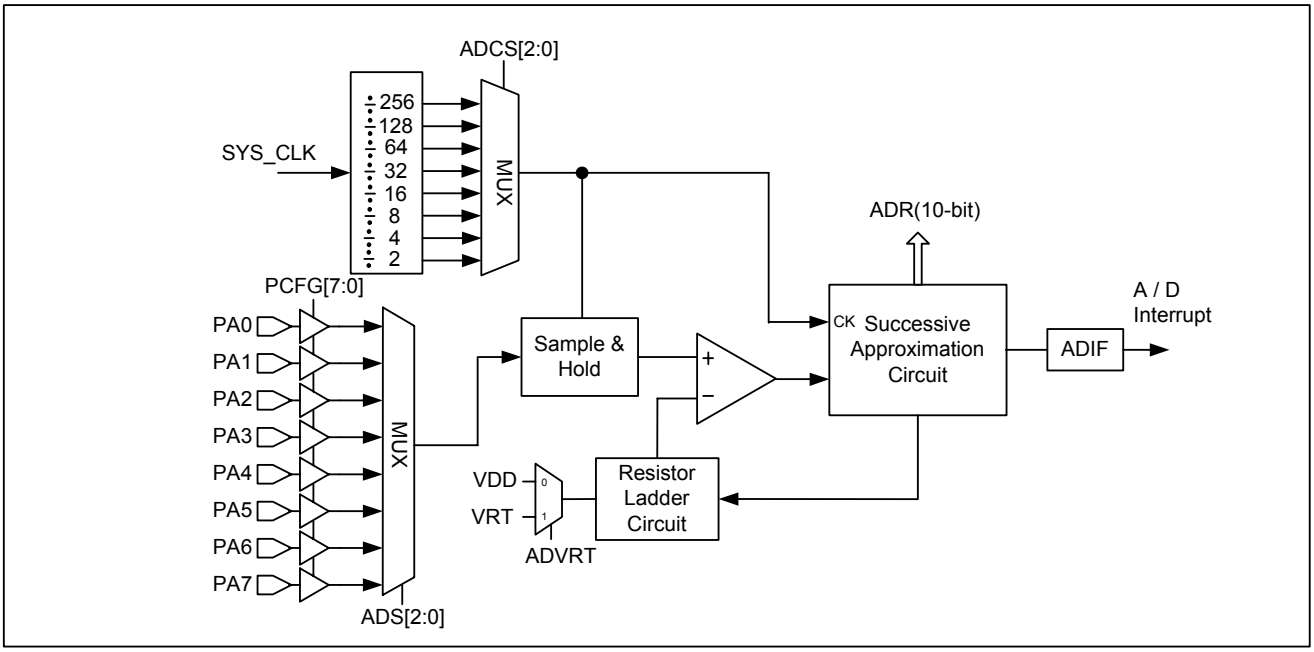


Figure 5.10-1 A/D converter block diagram

5.10.1.2. ADC Register

1). ADC Control Register 0 (P_AD_Ctrl0, $0028)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	ADEN	ADVRT	0	0	ADCS2	ADCS1	ADCS0	ADRDY
ACCESS	R/W	R/W	-	-	R/W	R/W	R/W	R/W
DEFAULT	0	0	0	0	0	1	1	0

Bit 7 **ADEN**: ADC enable bit

0 = ADC function is disabled

1 = ADC function is enabled

Bit 6 **ADVRT**: ADC top reference voltage source selection

1 = external voltage (PB7) as top reference voltage

0 = Vdd as top reference voltage

Bit [5:4] Reserved

Bit [3:1] **ADCS**[2:0] : ADC clock selection bits

111 = F_{SYS} ÷ 256

110 = F_{SYS} ÷ 128

101 = F_{SYS} ÷ 64

100 = F_{SYS} ÷ 32

011 = F_{SYS} ÷ 16

010 = F_{SYS} ÷ 8

001 = F_{SYS} ÷ 4

000 = F_{SYS} ÷ 2

F_{SYS}: frequency of system clock

Bit 0 **ADRDY/STARTB**: ADC ready or start control bit

Read: ADC ready status bit

1: ADC is ready and waiting for next conversion.

0: ADC is busy, ie. ADC is converting data.

Write: ADC start bit

1: no effect

0: ADC start conversion

2). ADC Control Register 1 (P_AD_Ctrl1, $29)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	PCFG7	PCFG6	PCFG5	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0
ACCESS	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
DEFAULT	0	0	0	0	0	0	0	0

Bit [7:0]: **PCFG**: ADC channel configuration control bits

These bits are used to configure A/D channel as analog pins of digital pins.

PCFG7:

1 = analog input (AN7)

0 = digital input (PA7)

PCFG6:

1 = analog input (AN6)

0 = digital input (PA6)

PCFG5:

1 = analog input (AN5)

0 = digital input (PA5)

PCFG4:

1 = analog input (AN4)

0 = digital input (PA4)

PCFG3:

1 = analog input (AN3)

0 = digital input (PA3)

PCFG2:

1 = analog input (AN2)

0 = digital input (PA2)

PCFG1:

1 = analog input (AN1)

0 = digital input (PA1)

PCFG0:

1 = analog input (AN0)

0 = digital input (PA0)

3). ADC Control Register 2 (P_AD_Ctrl2, $002A)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	ADCE	0	ADS2	ADS1	ADS0	0	0	0
ACCESS	R/W	-	R/W	R/W	R/W	-	-	-
DEFAULT	0	0	0	0	0	0	0	0

Bit 7: **ADCE**: ADC power control bit. This bit is used to enable the bias circuit of AD converter.

1 = Enable ADC power

0 = Disable ADC poewr

Bit 6: Reserved

Bit [5:3]: ADC current conversion channel selection bits

000 = select channel 0 (AN0)

001 = select channel 1 (AN1)
010 = select channel 2 (AN2)
011 = select channel 3 (AN3)
100 = select channel 4 (AN4)
101 = select channel 5 (AN5)
110 = select channel 6 (AN6)
111 = select channel 7 (AN7)
Otherwise = reserved

Bit [2:0]: Reserved

4). ADC Result Register High byte (P_AD_DataHi, $002B)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	P_AD_DataHi							
ACCESS	R/W							
DEFAULT	00h							

Bit [7:0] **P_AD_DataHi**: These bits are the most significant 8 bits of converted data.

5). ADC Result Register Low byte (P_AD_DataLo, $002C)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	P_AD_DataLo		-	-	-	-	-	-
ACCESS	R/W		-	-	-	-	-	-
DEFAULT	0	0	-	-	-	-	-	-

Bit [7:6] **P_AD_DataLo**: These bits are the least significant 2 bits of converted data.

Bit [5:0] Reserved

5.10.1.3. ADC Converting Flow

Following figure shows the process of AD module operation.

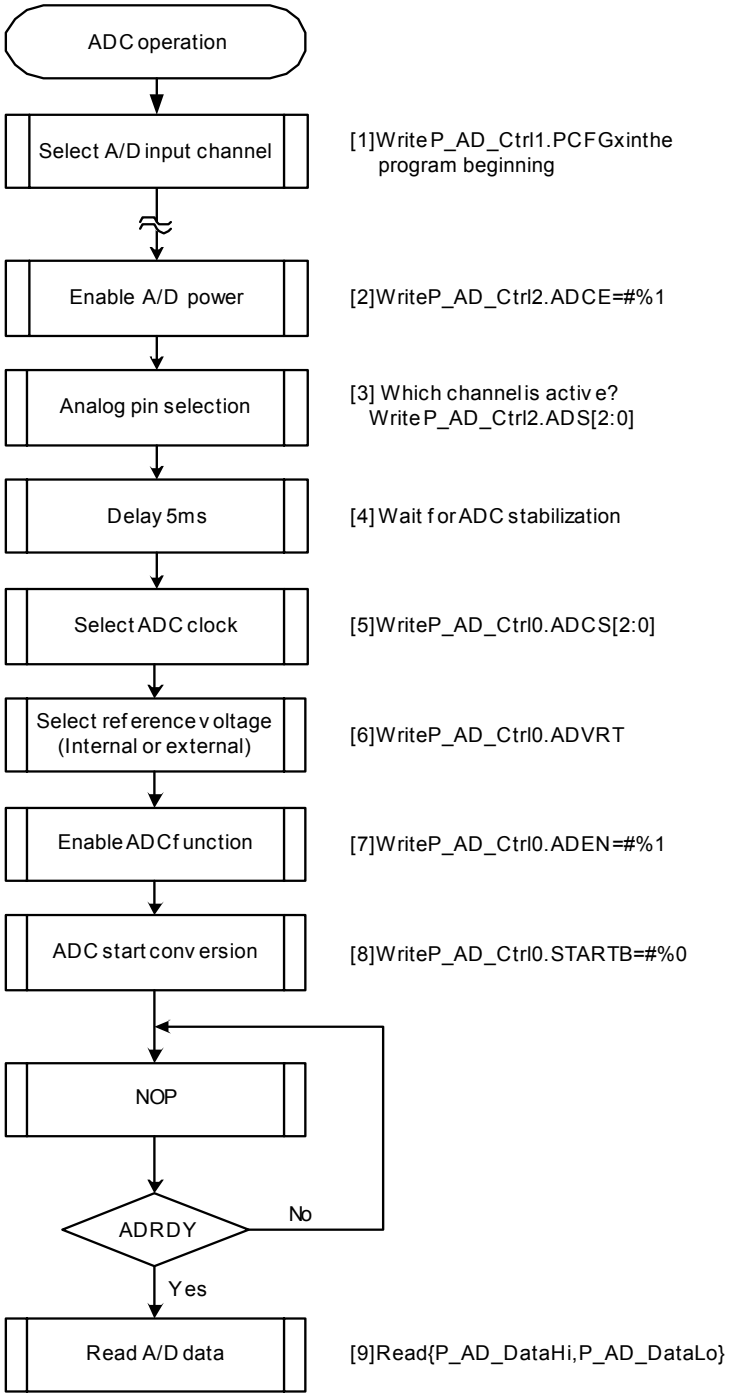


Figure 15.2 Process of A/D module operation

[Example] 5.10.1: Enable A/D converter

```
lda     #C_AD_Pin0                      ; PA0 is analog input
sta     P_AD_Ctrl1
lda     #(C_AD_CE+ C_AD_Ch0)            ; Select AN0
sta     P_AD_Ctrl2                      ; select channel 0
jsr     F_Delay5ms                      ; Delay 5.0 msec
lda     #(C_AD_EN + C_AD_CS_8)
sta     P_AD_Ctrl0                      ; ADC enable, ADC clock = Fsys / 8
lda     P_AD_Ctrl0
and     #11111110B                      ; Start convert
sta     P_AD_Ctrl0
```

5.11. Communication Module

5.11.1. SPI (Serial Peripheral Interface)

5.11.1.1. Introduction

The SPMC65P2404A/2408A devices include 4-pin SPI module. The SPI is a high-speed synchronous serial I/O that allows a serial bit stream to be transmitted out or received into the device at a programmable transferring rate. The SPI supports full-duplex synchronous transfer between a master device and a slave device. SPMC65P2404A/2408A supports both master and slave modes. The parameters such as operation mode, clock frequency, clock phase, and clock polarity are programmer programmable. The SPI module has the following features:

- Four external pins:
  - SDO: data output pin (shared with PC3)
  - SDI: data input pin (shared with PC2)
  - SCK: clock input/output pin (shared with PC1)
  - SSB: Slave select pin (shared with PC0)
- Supports full-duplex synchronous transfer
- Two operation modes: master and slave
- Baud rate: programmable transfer rate / Max. 2Mbps at 8MHz CPU clock
- Data word length: 8-bit
- Programmable clock phase and clock polarity settings
- Selectable data strobe time: input data bit sampled at the middle/end of data output time
- SPI TX/RX buffer is only one byte
- Improves noise immunity with sampling option
- Following is a function diagram of SPI module.

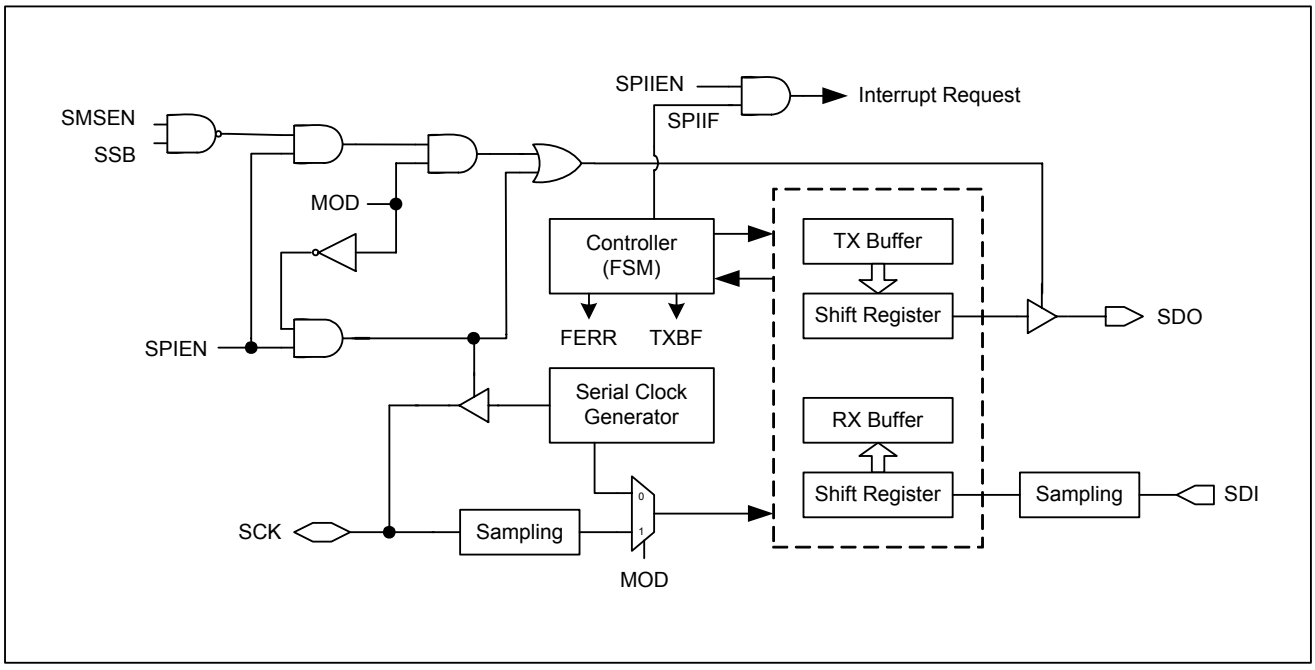


Figure 13.1 SPI structure

5.11.1.2. SPI Operation

5.11.1.3. Master Mode

In master mode operation, the shifting clock (SCK) is generated by SPI module. There are two control bits controlling the clock phase (SCKPHA) and polarity (SCKPOL) in the P_SPI_Ctrl0 register. The transmission starts immediately when data is written to the P_SPI_TxData register. In addition, the SDI pin (PC2) can be programmed as high-impedance if programmer wants to receive the data from slave device. SPMC65P2404A/2408A supports receiving and transmitting interrupt for SPI module. The interruption can be enabled simultaneously by setting SPIIEN bit to '1' in P_SPI_Status register. Similarly, SPIIF bit has to be cleared after receiving or transmitting interrupt service routine execution.

After one byte data is written in P_SPI_TxData register, the data is latched into its internal transmission buffer. If the shift register is

empty, the data will be loaded to the shift register and start transmitting at the next SCLK phase. On the other hand, if the shift register is busy in shifting data (TXBF flag is set in P_SPI_Status register), the new data will not be loaded until the present byte has been shifted out.

The SPI shifts the data from MSB to LSB through the SDO pin. The 8-bit data is shifted out after eight SCK cycles. At the same time, the data is also shifted in through SDI pin. When each 8-bit transfer is completed, the SPIIF bit in P_SPI_Status register will be set; also, a SPI interrupt will be generated if the SPIIEN bit is set to '1' in P_SPI_Status register.

In contrast, while SPI interface is received one byte successfully, the received data will be latched into received buffer. At that time, SPIIF bit in P_SPI_Status register will be set and a SPI interrupt will be issued to CPU if the SPIIEN bit in the P_SPI_Status register is set.

The following diagram depicts the timing scheme on SPI master mode for different operation types (polarity control bit equals "1" or "0", phase control bit equals "1" or "0", and sample strobe control bit equals "1" or "0").

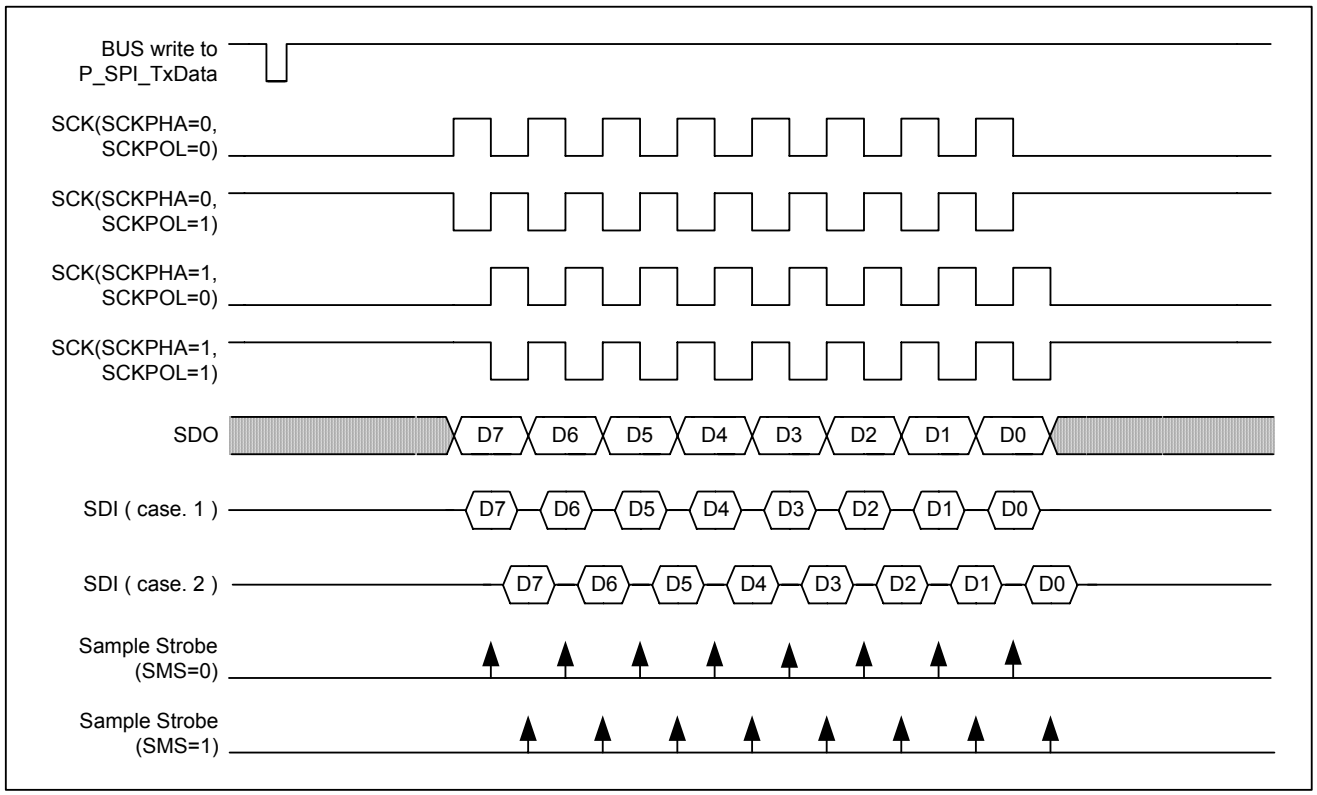


Figure 5.11-1 SPI master mode timing

5.11.1.4. Slave Mode

In slave mode, the shifting clock SCK comes from external SPI master so that the transmission starts from the first external SCK event. To transmit data, programmer should write the data to its transmitting buffer before the first SCK coming from the master. Both master and slave devices must be programmed with the same SCK phase and polarity to transmit and receive data.

If the clock phase bit (SCKPHA) is "1", the first data bit to be shifted out starts right after the command written to P_SPI_TxData register. If the clock phase bit (SCKPHA) is "0", the first data bit to be shifted will start after first SCK edge.

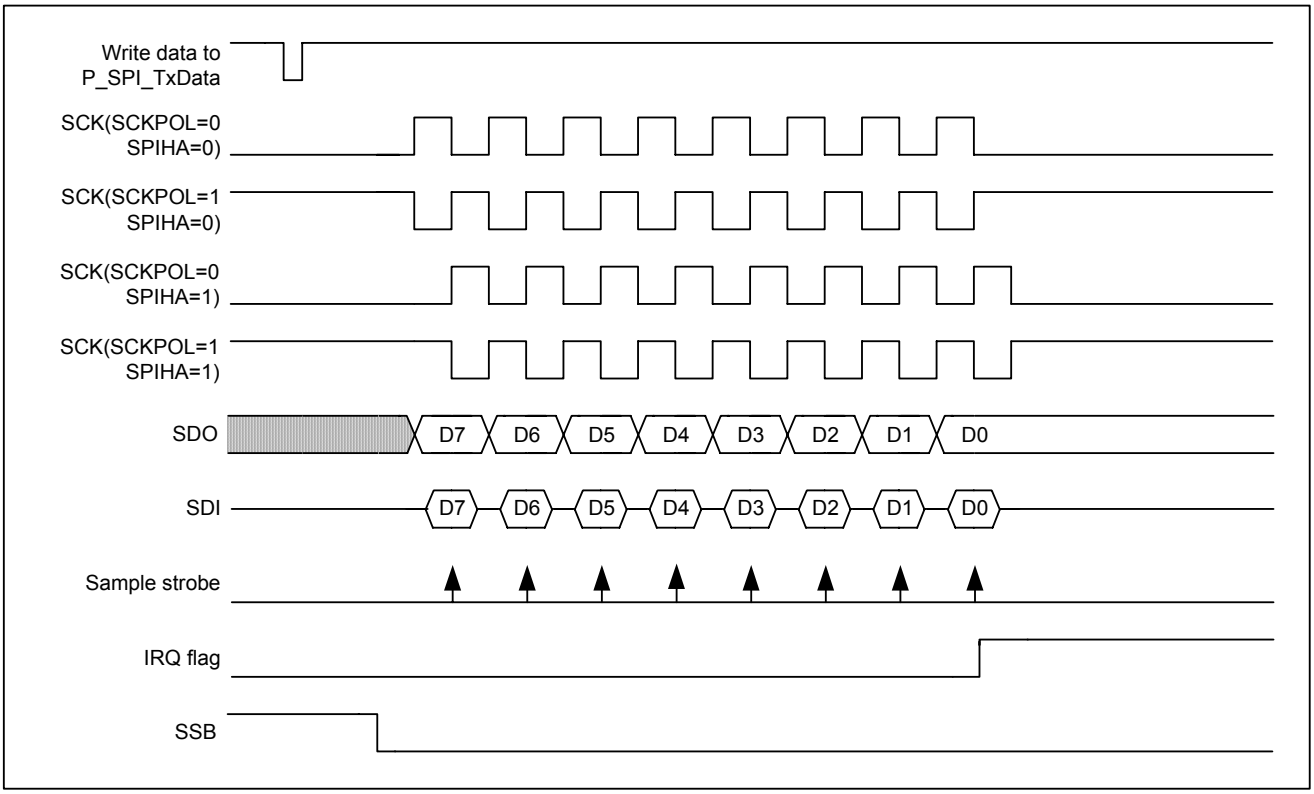


Figure 5.11-2 SPI slave mode timing

5.11.1.5. SPI Register

1). SPI Control Register0 (P_SPI_Ctrl0, $0038)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	SPIEN	MOD	SCKPHA	SCKPOL	SMS	SCKSEL2	SCKSEL1	SCKSEL0
ACCESS	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
DEFAULT	0	0	0	0	0	0	0	0

Bit 7 **SPIEN**: SPI enable bit. Once this bit is set to 1, PC[3:0] becomes SPI interface.
1 = enable SPI
0 = disable SPI

Bit 6 **MOD**: SPI mode
1 = slave mode
0 = master mode

Bit 5 **SCKPHA**: SPI clock phase. SPI clock phase select, see SPI master mode timing

Bit 4 **SCKPOL**: SPI clock polarity. SPI clock polarity select, see SPI master mode timing

Bit 3 **SMS**: Sample Mode selection bit for master mode
1 = input data bit sampled at the end of data output time
0 = input data bit sampled at the middle of data output time

Bit [2:0] **SCKSEL** [2:0]: Master mode clock selection bit
111 = F_{SYS} ÷ 128
110 = F_{SYS} ÷ 128
101 = F_{SYS} ÷ 128
100 = F_{SYS} ÷ 64
011 = F_{SYS} ÷ 32
010 = F_{SYS} ÷ 16
001 = F_{SYS} ÷ 8
000 = F_{SYS} ÷ 4
F_{SYS} : frequency of system clock

2). SPI Control Register1 (P_SPI_Ctrl1, $0039)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	SMSEN	SWRST	-	-	-	-	SPISPCLK1	SPISPCLK0
ACCESS	R/W	R/W	-	-	-	-	R/W	R/W
DEFAULT	0	0	0	0	0	0	1	0

Bit 7 **SMSEN**: SPI slave mode selection input

1 = PC0 becomes SSB input pin

0 = PC0 is GPIO

SSB: slave mode selection, low active

Bit 6 **SWRST**: SPI software reset

Write: 1 = generate one pulse to reset SPI module except register setting

0 = no action.

Read: always 0

Bit [5:2] Reserved

Bit [1:0] **SPISPCLK** [1:0]: sampling clock selection bits

11 = F_{SYS} ÷ 4

10 = F_{SYS} ÷ 2

01 = F_{SYS}

00 = no sampling

F_{SYS} : frequency of system clock

Note: The purpose of sampling clock is to prevent received data from glitch noise, but lower sampling clock would affect the speed of communication

3). SPI State Status Register (P_SPI_Status, $003A)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	SPIIF	SPIIEN	TXBF	-	-	-	-	BUFFull
ACCESS	R/W	R/W	R	-	-	-	-	R
DEFAULT	0	0	0	0	0	0	0	0

Bit 7 **SPIIF**: SPI interrupt flag

Read:

1 = SPI interrupt flag is active

0 = no flag

Write:

1 = clear flag

0 = no action.

Bit 6 **SPIIEN**: SPI interrupt enable bit

1 = enable

0 = disable

Bit 5 **TXBF**: Transmission buffer full flag.

1 = transmission buffer is full.

0 = transmission buffer is empty

Bit [4:1] Reserved

Bit 0 **BUFFull**: Buffer full and overwrite

1 = overwrite

0 = buffer is under normal condition

4). SPI Transmission Buffer Register0 (P_SPI_TxData, $003B)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	SPITXDATA7	SPITXDATA6	SPITXDATA5	SPITXDATA4	SPITXDATA3	SPITXDATA2	SPITXDATA1	SPITXDATA0
ACCESS	W	W	W	W	W	W	W	W
DEFAULT	0	0	0	0	0	0	0	0

Bit [7:0] **SPITXDATA**: SPI transmit data

Read: always is #$00

Write: transmission data

5). SPI Receive Buffer Register0 (P_SPI_RxData, $003C)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	SPIRXDATA7	SPIRXDATA6	SPIRXDATA5	SPIRXDATA4	SPIRDATA3	SPIRXDATA2	SPIRXDATA1	SPIRXDATA0
ACCESS	R	R	R	R	R	R	R	R
DEFAULT	0	0	0	0	0	0	0	0

Bit [7:0] **SPIRXDATA**: SPI receive data

Read: SPI receive data

Write: no action

[Example] 5.11.1 SPI formula for sampling clock selection and I/O setting

1. SPI clock formula

Sampling clock must follow the formula below in order to ensure SPI can get data correctly.

Sampling clock > 4 x SPI clock (Master mode)

System clock > 4 x SPI clock x SPISPCLK (Slave mode)

For example:

SPMC65P2404A as Master device

SPI Clock = 1MHz(PC1), F_{SYS} = 8MHz

Sampling clock > 4 x 1MHz = 4MHz

Sampling clock must be greater than 4MHz to ensure the receiving data correctly.

So, sampling clock selection bits (SPISPCLK [1:0]) may set to 1.

Sampling clock = 8MHz / 1 = 8Mhz

Port C setting status when SPI operation in Master mode

PC0(SS): Output

PC1(SCK): Output

PC2(SDI): Input with float

PC3(SDO): Output

Port C setting status when SPI operation in Slave mode

PC0(SS): Input with float

PC1(SCK): Input with float

PC2(SDI): Input with float

PC3(SDO): Input with float

[Example] 5.11.2 SPI operation on master mode (initial)

```
        lda     #00000100B                      ; Set PC2 as input float for SDI
        sta     P_IOC_Attrib
        lda     #(C_SPI_INTIF+C_SPI_INTEN)      ; Enable SPI INT, clear SPI INT flag
        sta     P_SPI_Status
        lda     #(C_SPI_SWRST+C_SPISPC_Div_4)   ; Software reset SPI, sample clock= Fsys/4(must >= 4* SPI clock)
        sta     P_SPI_Ctrl1
        lda     #(C_SPI_EN+C_SPICS_Div_128)     ; Enable SPI, master mode, Clock= Fsys(8MHz)/128= 62.5KHz
        sta     P_SPI_Ctrl0
```

[Example] 5.11.3 SPI operation on master mode (transmit data)

```
L_TestSPIL7:
        set     P_SPI_Status, CB_SPI_INTIF      ; clear SPI INT flag
        lda     #$55
        sta     P_SPI_TxData                    ; Send data
L_TestSPIL71:
        lda     P_SPI_Status
        and     #C_SPI_INTIF                    ; SPI send finish ?
        beq     L_TestSPIL71                    ; no
```

[Example] 5.11.4 SPI operation on master mode (receive data)

```
L_TestSPIL5:
        lda     P_SPI_Status
        and     #C_SPI_INTIF                    ; SPI send INT ?
        beq     L_TestSPIL5                     ; no
        nop
        lda     P_SPI_RxData                    ; read data
        sta     G_MWorkReg1
```

[Example] 5.11.5 SPI operation on slave mode (initial)

```
        lda     #00001111B                      ; Set PC0~3 as input float for slave mode
        sta     P_IOC_Attrib
        lda     #(C_SPI_EN+C_SPI_MOD)           ; Enable SPI,slaver mode, Clock= Fsys(8MHz)/128= 62.5KHz
        sta     P_SPI_Ctrl0
        lda     #(C_SPI_SMSEN+C_SPI_SWRST+C_SPISPC_Div_4)
                                                ; Software reset SPI,sample clock= Fsys/4(sample clock >4* SPI clock)
        sta     P_SPI_Ctrl1
        lda     #(C_SPI_INTIF+C_SPI_INTEN)      ; Enable SPI INT,clear SPI INT flag
        sta     P_SPI_Status
```

5.11.2. UART (Universal Asynchronous Receiver/Transceiver) (2408A Only)

5.11.2.1. Introduction

The SPMC65P2408A includes a serial communications interface module. The UART module supports digital communications between CPU and other asynchronous peripherals that use the standard non-return-to-zero (NRZ) format. The UART receiver and transmitter are double-buffered, and each has its own separate enable and interrupt bits. Both can be operated independently or

simultaneously in the full-duplex mode. The bit rate ranged from 2400 bps to 38400 is programmable through an 8-bit baud-select register.

To ensure data integrity, the UART has error check flag for parity, overrun, and framing errors. The parity error checks if the parity matches the setting of UART. The framing error checks if the data stops at correct status. If the transmission speed is too fast that programmer cannot read the received data out before another byte of data is received, an overrun error will be set. Programmer can read the flag bit to check the transmission status.

Features of the UART module include:

- Two external pins:
  - RXD: data reception pin (shared with PC5)
  - TXD: data transmission pin (shared with PC4)
- Provides standard asynchronous, full-duplex communication
- Programmable trans-receive baud rate
- Parity can be even, odd or disabled for generation and detection
- Stop bit width can be 1 or 2 bits
- Supports transmitting interrupt
- Supports receiving interrupt
- High noise rejection for bit receiving (majority decision of 3 consecutive samples in the middle of received bit time)
- Framing, Parity error detection during reception.
- Overrun detection
- Programmable baud rate from 2400 bps to 38400 bps @ Fsys=8MHz

Figure 5.11-3 shows the block diagram of UART.

Figure 5.11-4 and figure 5.11-5 shows the operation of UART.

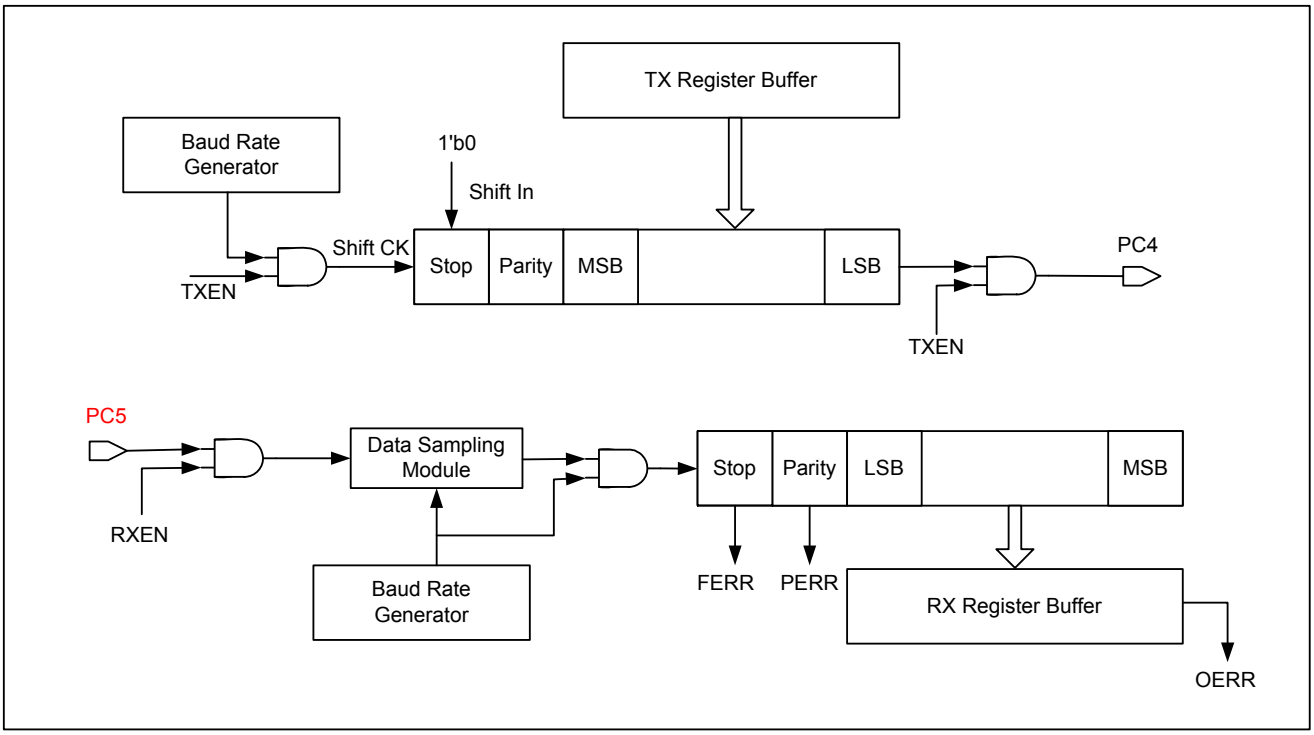


Figure 5.11-3 Block diagram of UART

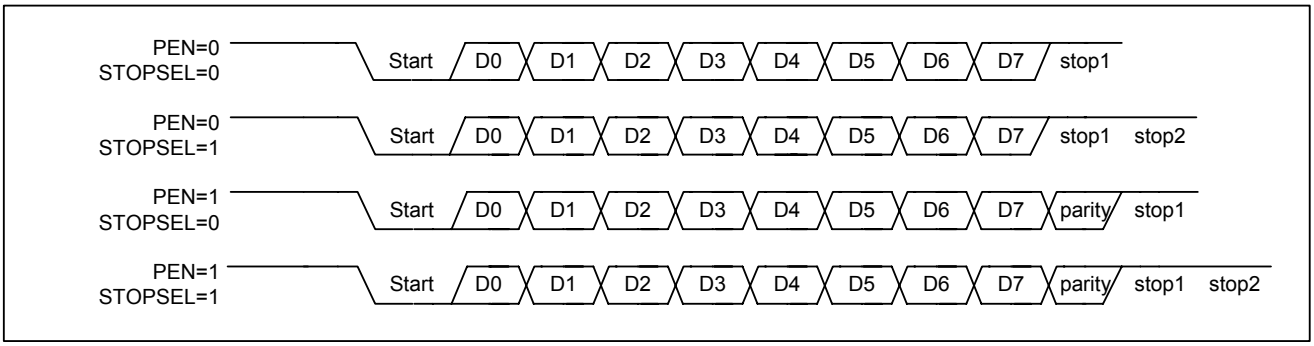


Figure 5.11-4 Data format of UART

5.11.2.2. UART Operation

There exists a baud rate register and a 8-bit timer to generate the baud rate. Each times the timer increments from its maximum count (255), a clock is sent to the baud rate circuit. The clock is through divide-by-16 counter to generate the baud rate. The timer is reloaded automatically the value in baud rate register.

Baud Rate= F_{SYS} / [16 x (256 –P_UART_Baud)]

The content in baud rate register is taken as an 8-bit unsigned number. To derive the required baud rate register values from a known baud rate, use the equation and refer to table 14.2:

P_UART_Baud =256 - F_{SYS} / (16 x Baud Rate)

The UART begins transmitting after the first rollover of the divide-by-16 counter after the software writes to the P_UART_Data register. The UART transmits data on the TXD pin in the following order: start bit, 8 data bits (LSB first), parity bit (parity enable mode only), stop bit. The TXIF bit in P_UART_Status register is set after 2 CPU clock cycles when the stop bit is transmitted. The TXIF bit is cleared automatically after the software writes to the P_UART_Data register.

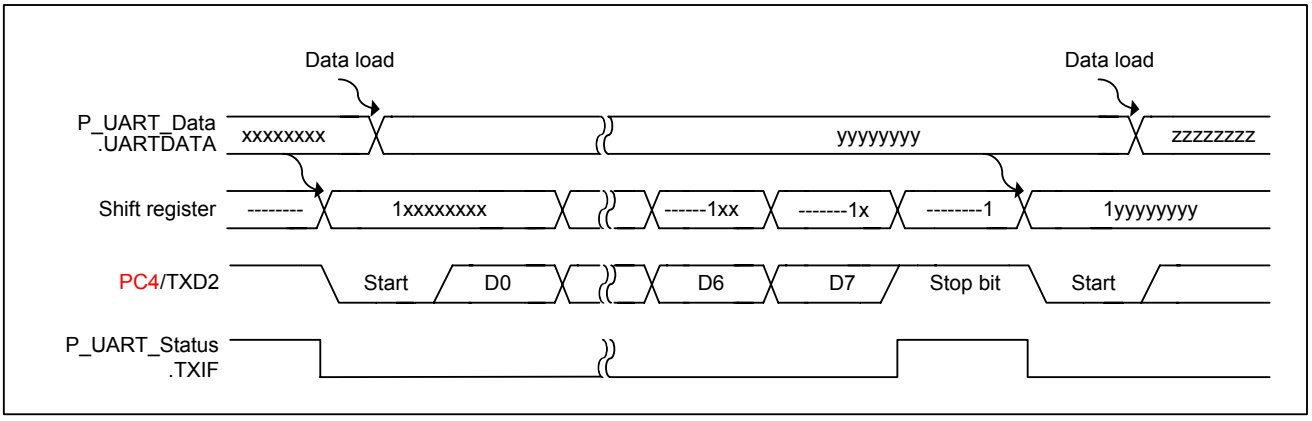


Figure 5.11-5 Transmit buffer empty

Reception begins at the falling edge of a start bit received on RXD pin, when enabled by the RXEN bit in P_UART_Ctrl register. For this purpose, RXD is sampled 16 times per bit for any baud rate. When a falling edge of a start bit is detected, the divide-by-16 counter used to generate the receiving clock is reset to align the counter rollover to the bit boundaries.

For noise rejection, the serial port establishes the content of each received bit by a majority decision of 3 consecutive samples in the middle of each bit time. This is especially true for the start bit. If the falling edge on RXD is not verified by a majority decision of 3 consecutive samples logic low level, then the serial port stops reception and waits for another falling edge on RXD.

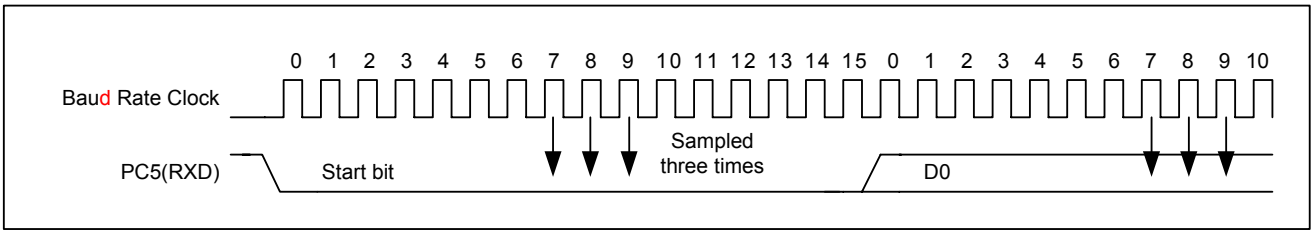


Figure 5.11-6 UART data sampling method

After receiving the stop bit, the UART module writes the received byte to the P_UART_Data register and set the RXIF and RXBF bit. The serial port then waits for another high-to-low transition on the RXD pin.

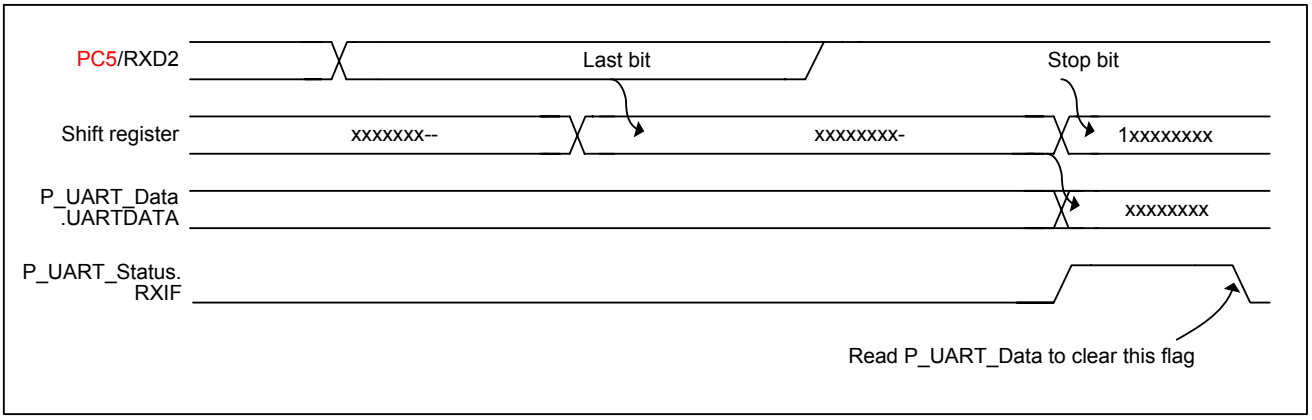


Figure 5.11-7 RX buffer full

If the received byte is not read out before the next reception finished, the data will be over-written by the new received. In every reception session, RXBF is checked after receiving the stop bit. If the RXBF bit is set, the OE will be set to record this overrun error event. Remarkably, the OE will be cleared automatically if the error check success in the following session.

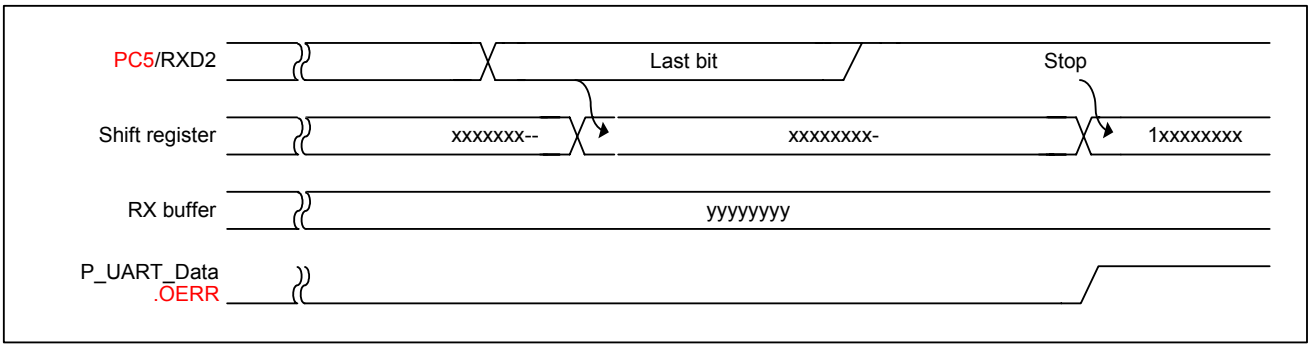


Figure 5.11-8 Overrun error timing

The parity and frame check is used for improving the reliability of reception. The parity can be even or odd according to the configuration of P_UART_Ctrl.PSEL. The parity check is performed after receiving parity bit if P_UART_Ctrl.PEN is enabled. The PE bit will be set if any parity error. Please refer to **Figure 5.11-9** for timing diagram. The Stop Bit is the part of the UART data formation. If the reception session fails to receive Stop Bit, the integrity of the data frame is lost. The FE bit is set to record this frame error event. **Figure 5.11-10** shows the frame error timing. Remarkably, PE and FE will be clear automatically if the error checks success in the following session, respectively.

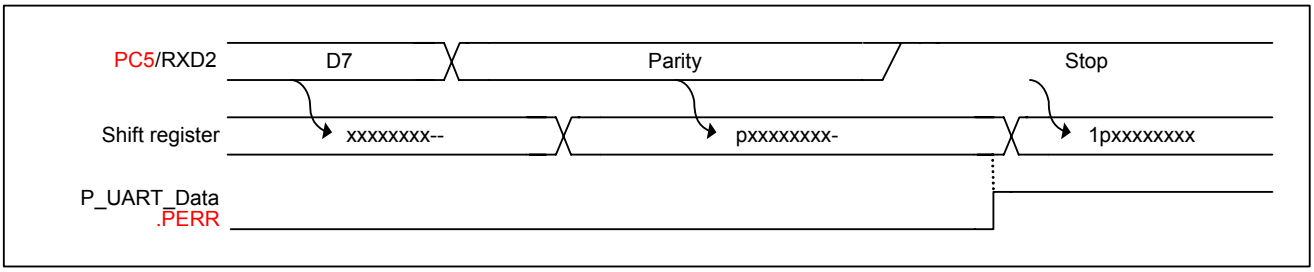


Figure 5.11-9 Parity error timing

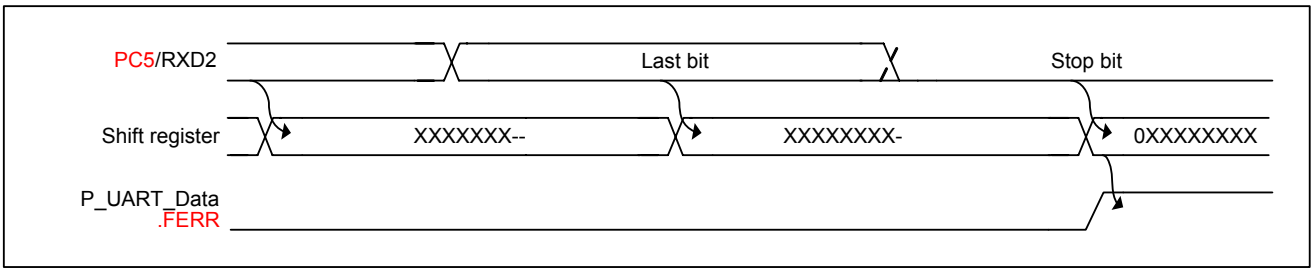


Figure 5.11-10 Frame error timing

5.11.2.3. UART Register

6). UART Control Register(P_UART_Ctrl, $0046)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	RXIE	TXIE	RXEN	TXEN	SOFTRST	STOPSEL	PSEL	PEN
ACCESS	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
DEFAULT	0	0	0	0	0	0	0	0

Bit 7 **RXIE**: Receive interrupt enable bit
1 = enable receive interrupt request
0 = disable receive interrupt request

Bit 6 **TXIE**: Transmit interrupt enable bit
1 = enable transmit interrupt request
0 = disable transmit interrupt request

Bit 5 **RXEN**: UART receive function enable bit
1 = enable receive interrupt request
0 = disable receive interrupt request

Bit 4 **TXEN**: UART transmit function enable bit
1 = enable UART transmit function
0 = disable UART transmit function

Bit 3 **SOFTRST**: Software reset
Write:
1 = reset all UART module
0 = no action

Bit 2 **STOPSEL**: Stop bit length selection bit
1 = 2 stop bits
0 = 1 stop bit

Bit 1 **PSEL**: Parity type selection bit
1 = even parity
0 = odd parity

Bit 0 **PEN**: Parity enable bit
1 = enable parity check or generation
0 = disable parity check or generation

7). UART BAUD Register (P_UART_Baud, $0047)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	UARTBAUD7	UARTBAUD6	UARTBAUD5	UARTBAUD4	UARTBAUD3	UARTBAUD2	UARTBAUD1	UARTBAUD0
ACCESS	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
DEFAULT	0	0	0	0	0	0	0	0

Bit [7:0] **UARTBAUD[7:0]**: UART baud rate divisor

Baud Rate= F_{SYS} / [16 x (256 –P_UART_Baud)]

The formula for baud rate can be derived from below formula:

P_UART_Baud =256 - F_{SYS} / (16 x Baud Rate)

Table 14.2 Baud rate vs. P_UART_baud register value

Baud Rate (bps)	Suggestion value at Fsys=8MHz
2400	#$30
4800	#$98
9600	#$CC
19200	#$E6
38400	#$F3

8). UART Status Register (P_UART_Status, $0048)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	RXIF	TXIF	BUSY	-	-	OERR	PERR	FERR
ACCESS	R	R	R	-	-	R	R	R
DEFAULT	0	0	0	0	0	0	0	0

Bit 7 **RXIF**: Receive interrupt flag bit

1 = RX data is ready (full)

0 = RX data is not ready

Bit 6 **TXIF**: Transmit interrupt flag bit

1 = TX buffer is ready (empty)

0 = TX buffer is full

Bit 5 **BUSY**: UART transmission in progress flag bit

1 = transmission is progress

0 = transmission is finished and UART is waiting for next transmission.

Bit [4:3] Reserved

Bit 2 **OERR**: Overrun error flag bit

Read:

1 = overrun error occurs

0 = no overrun error

Bit 1 **PERR**: Parity error flag bit

Read:

1 = parity error occurs

0 = no Parity error

Bit 0 **FERR**: Frame error flag bit

Read:

1 = frame error occurs

0 = no frame error

9). UART DATA Register (P_UART_Data, $0049)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	UARTDATA7	UARTDATA6	UARTDATA5	UARTDATA4	UARTDATA3	UARTDATA2	UARTDATA1	UARTDATA0
ACCESS	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
DEFAULT	0	0	0	0	0	0	0	0

Bit 7 **UARTDATA[7:0]**: UART data register

Read: get received data

Write: push the transmission data to buffer

[Example] 5.11.6 UART initializes

```
        lda     #00000000B
        sta     P_IOC_Data
        sta     P_IOC_Attrib
        lda     #11011111B                      ; PC5(Rx) as input,PC4(Tx) as output
        sta     P_IOC_Dir
        lda     #C_UART_SOFTRST                 ; Reset UART
        sta     P_UART_Ctrl
        lda     #(C_UART_RXEN+C_UART_TXEN+C_UART_PSEL+C_UART_PEN)
                                                ; Enable parity check, even parity
        sta     P_UART_Ctrl
        lda     #$CC                            ; UART Baud Rate= 9600 @8MHz
        sta     P_UART_Baud
        lda     #(C_UART_OERR+C_UART_PERR+C_UART_FERR)
                                                ; Clear error flags
        sta     P_UART_Status
```

5.12. Other On-Chip Peripherals

5.12.1. Watchdog

The purpose of a watchdog timer is preventing the system from entering a software dead lock loop. A software dead lock loop usually is generated by a program logic error or external interference.

The watchdog circuit contains an up-counter which uses a 25KHz clock as its clock source. The 25KHz clock, so called SLOWCK, is generated from an internal RC oscillator. Within a certain period, watchdog counter must be cleared. If the watchdog timer is not cleared for a setting time, watchdog circuit will issue an interrupt to CPU. The watchdog keeps running and issuing interrupt periodically if CPU does not answer the interrupt. After eight interrupts are issued, a reset signal will apply to CPU to reset the program counter and status registers in CPU to restart the program. This system protects the system from incorrect code execution by launching a CPU reset when the watchdog timer overflows. To prevent watchdog overflow, programmer's program must clear the watchdog timer period. For SPMC65P2404A/2408A devices, watchdog function can be enabled or disabled by Device Configuration Register (P_MO, $7FE0). The Watchdog Timer Block Diagram is shown in figure Fig. 5.12-1.

The watchdog can also be used to wake from STOP mode. Through Watchdog Control Register (P_WDT_Ctrl, $0032), programmer can keep watchdog enabled when system enters STOP mode. Programmer can use this function to wake up system period. But as discussed in section 5.4.2, the watchdog interrupt frequency has to set properly to avoid watchdog reset occurs when system is waiting for clock source to be stable.

Feature:

- 8 stage selectable interrupt rates ranged from 195Hz to 1.5Hz
- Resets CPU after 8 interrupt signals are issued
- Can be used as a wakeup source in STOP mode
- Hardware watchdog enable/disable selected from device configuration registers.

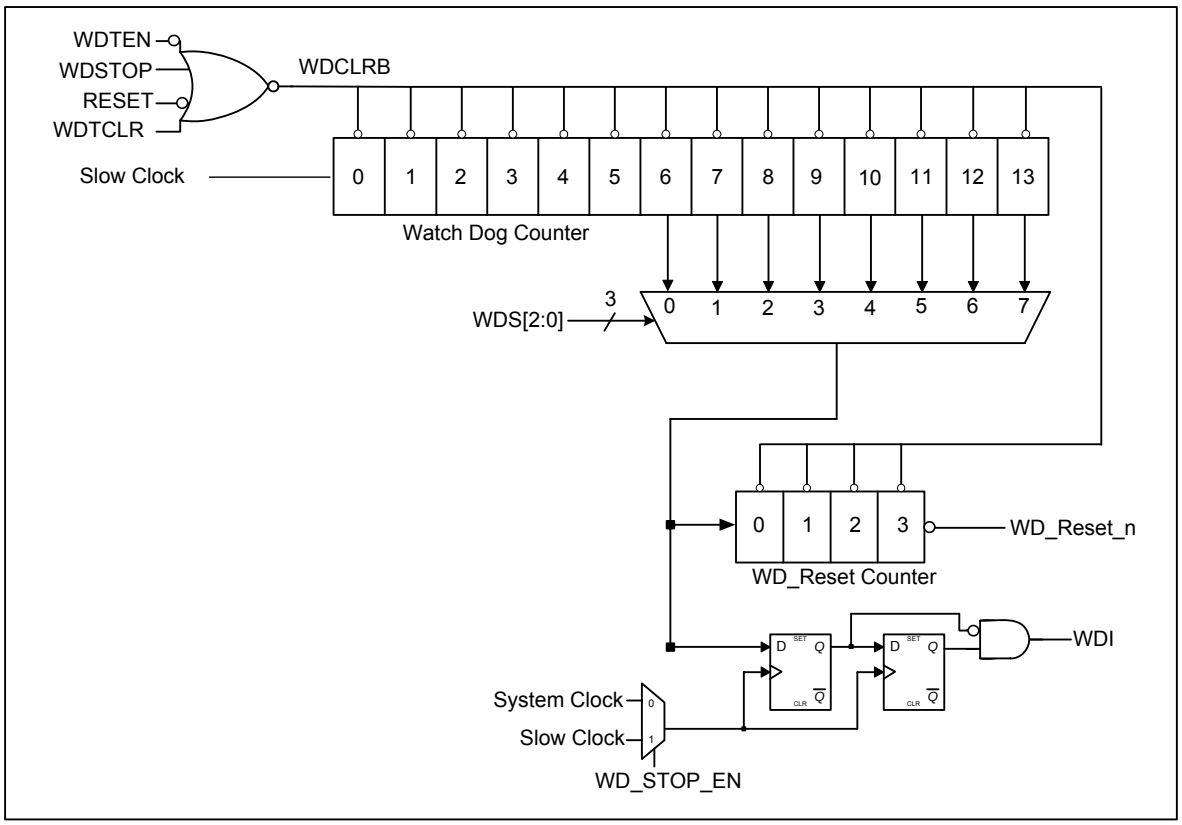


Figure 5.12-1 Watchdog timer block diagram

1). Watchdog Control Register (P_WDT_Ctrl, $0032)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	SCKEN	WDS2	WDS1	WDS0	-	-	-	-
ACCESS	R/W	R/W	R/W	R/W				
DEFAULT	1	1	1	1	0	0	0	0

Bit 7 **SCKEN**: Slow clock enable bit in stop mode

0 = disable in stop mode

1 = enable in stop mode

Bit [6:4] **WDS** [2:0]: Select bits of watchdog interrupt rate

000 = F_{SLOW}÷128

001 = F_{SLOW}÷256

010 = F_{SLOW}÷512

011 = F_{SLOW}÷1024

100 = F_{SLOW}÷2048

101 = F_{SLOW}÷4096

110 = F_{SLOW}÷8192

111 = F_{SLOW}÷16384

F_{SLOW}: internal build-in RC oscillator, typical frequency is 25kHz.

WDS[2:0]	Watch Dog Reset (Hz)	Watch Dog Interrupt Clock (Hz)
000	195/8	195
001	97/8	97
010	48/8	48
011	24/8	24
100	12/8	12
101	6/8	6
110	3/8	3
111	1.5/8	1.5

2). Watchdog Clear Register (P_WDT_Clr, $0010)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	P_WDT_Clr							
ACCESS	W							
DEFAULT	00h							

Bit [7:0] **P_WDT_Clr**: Watchdog clear register

It is used to clear watchdog timer by writing "#$55" to this register.

[Example] 5.12.1 Enable watchdog function

```
lda   #C_WDT_Div_16384          ; WDI= Fslow(25KHz)/16384= 1.5Hz
sta   P_WDT_Ctrl
sta   P_WDT_Ctrl
lda   #$FF
sta   P_INT_Flag0               ; Clear INT request flag
set   P_INT_Ctrl0,CB_INT_WDIE   ; Enable WDT INT.
```

5.12.2. Time Base Interval Timer

SPMC65P2404A/2408A provides a 22-bit time base Interval timer with 15 selectable frequency (7 for SPMC65P2404A) options. The interrupt frequency of this Interval Timer can be set in P_BUZ_Ctrl register, as shown in table 5.11.1. If interval timer interrupt enable bit is set to '1', it will cause interruption every interval timer period. This interval timer can be used as a constant timing source

without using a normal timer. Detailed register setting is shown in next page.

[Table] 5.12.1 Time base interval timer list (In this table, the system clock is F_{SYS} = 8MHz.)

INTIMS [3:0]	Time Base Interval Timer period	
	Divisor	F_{T0}= F_{SYS}
0000	-	Function disable
0001	2^7=128	16us
0010	2^8=256	32us
0011	2^9=512	64us
0100	2^{10}=1024	128us
0101	2^{11}=2048	256us
0110	2^{12}=4096	512us
0111	2^{13}=8192	1.024ms
1000	2^{14}=16384	2.048ms (SPMC65P2408A only)
1001	2^{15}=32768	4.096ms (SPMC65P2408A only)
1010	2^{16}=65535	8.192ms (SPMC65P2408Aonly)
1011	2^{17}=131072	16.384ms (SPMC65P2408A only)
1100	2^{18}=262144	32.768ms (SPMC65P2408A only)
1101	2^{19}=524288	65.535ms (SPMC65P2408A only)
1110	2^{21}	262.144ms (SPMC65P2408A only)
1111	2^{23}	1s (SPMC65P2408A only)

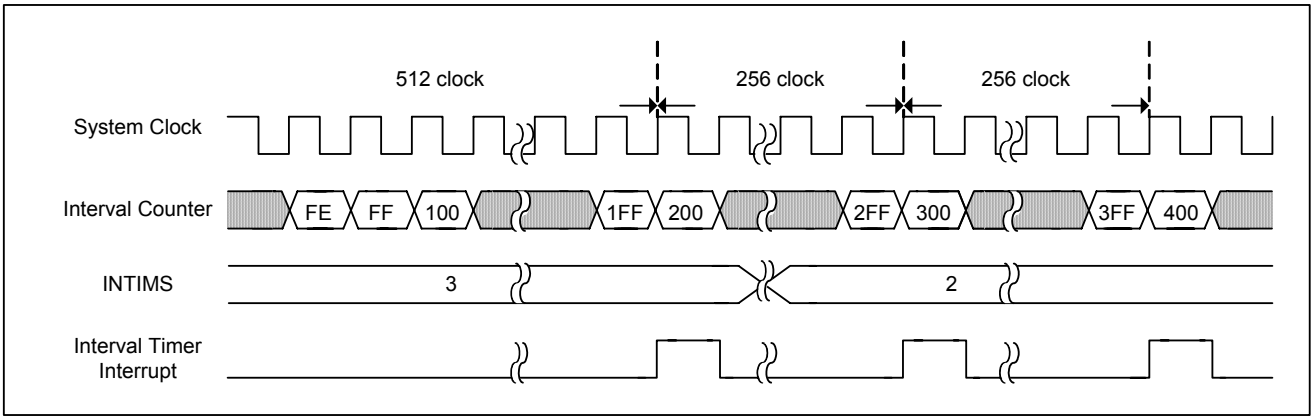


Figure 5.12-2 Time base interval timer operating waveform

[Example] 5.12.2 Generate 512us interrupt using time base interval timer.

```
lda   #C_TBASE_Div_4k                ; Store $60 to P_BUZ_Ctrl register
sta   P_BUZ_Ctrl
set   P_INT_Ctrl2, CB_INT_ITVALIE    ; Enable Interval Timer interrupt
```

5.12.3. Buzzer

A 50% duty pulse can be output using the buzzer output circuit, which is useful for buzzer drive. Driver output from pin PB6. The buzzer output frequency can be selected by setting BZFS[3:0] in P_BUZ_Ctrl register.

Table 10.1 Buzzer output list (In this table, the system clock is F_{SYS} = 8MHz.)

BZFS[3:0]	BZO (Buzzer output) rate

	Divisor	F_{T0}= F_{SYS} / Divisor
0000	-	Function disable
0001	64	125k
0010	128	62.5k
0011	256	31.25k
0100	512	15.625k
0101	1024	7.8125k
0110	2048	3.906k
0111	4096	1.953k
1000	8192	0.976k
1001	4	2M
1010	8	1M
1011	16	500k
1100	32	250k
1101	32	250k
11110	32	250k
1111	32	250k

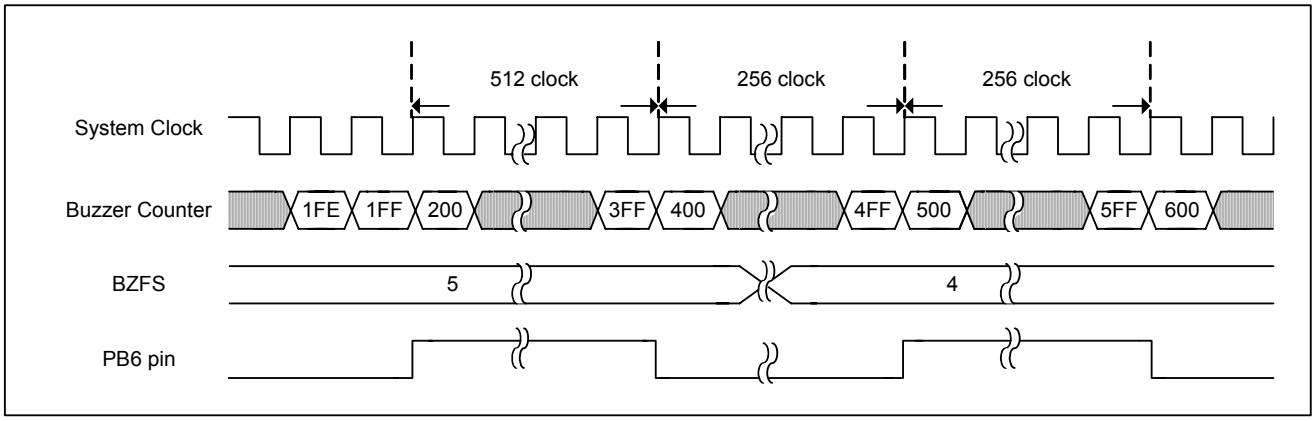


Figure 5.12-3 Buzzer operating waveform

1). Buzzer Control Register (P_BUZ_Ctrl, $002D)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	INTIMS3	INTIMS2	INTIMS1	INTIMS0	BZFS3	BZFS2	BZFS1	BZFS0
ACCESS	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
DEFAULT	0	0	0	0	0	0	0	0

Bit [7:4] INTIMS[3:0]: Interval timer frequencies selection bits

0000 = function disable

0001 = F_{SYS}÷128

0010 = F_{SYS}÷256

0011 = F_{SYS}÷512

0100 = F_{SYS}÷1024

0101 = F_{SYS}÷2048

0110 = F_{SYS}÷4096

0111 = F_{SYS}÷8192

1000 = F_{SYS}÷16384 (SPMC65P2408A only)

1001 = F_{SYS}÷32768 (SPMC65P2408A only)

1010 = F_{SYS}÷65535 (SPMC65P2408A only)

1011 = F_{SYS}÷131072 (SPMC65P2408A only)

1100 = F_{SYS}÷262144 (SPMC65P2408A only)

1101 = F_{SYS}÷524288 (SPMC65P2408A only)

1110 = $F_{SYS} \div 2^{21}$ (SPMC65P2408 only)
1111 = $F_{SYS} \div 2^{23}$ (SPMC65P2408 only)

Bit [3:0] BZFS[3:0]: Buzzer frequencies select bits
0000 = Disable
0001 = FSYS/4
0010 = FSYS/128
0011 = FSYS/256
0100 = FSYS/512
0101 = FSYS/1024
0110 = FSYS/2048
0111 = FSYS/4096
1000 = FSYS/8192
1001 = FSYS/4
1010 = FSYS/8
1011 = FSYS/16
1100 = FSYS/32
1101 = FSYS/32
1110 = FSYS/32
1111 = FSYS/32

[Example] 5.12.3 Generate pulse output using buzzer function, which period is 128us and duty is 50%.

```
lda     #C_BUZ_Div_1k                   ; store $05 to accumulator
sta     P_BUZ_Ctrl
```

5.13. Device Configuration Register

5.13.1.1. Introduction

The Device Configuration registers are used to setup the operation condition. They have to be programmed within the OTP code from OTP Writer. The locations for the registers are in $7FE0, $7FE2, and $7FE3. The first byte ($7FE0) is used to configure clock source, LVR enable and watchdog enable. The second byte ($7FE2) is used to set security control for RC oscillator output and GPIO initial setting . Once the security bit is set, the OTP code cannot be R/W from writer. The third byte ($7FE3) is used to select the Non Maskable Interrupt source.

5.13.1.2. Device Configuration Register

1). Device Configuration Register (P_MO, $7FE0)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	-	-	CLKSEL1	CLKSEL0	LVREN	WDTEN	-
ACCESS	R	R	R	R	R	R	R	R
DEFAULT	1	1	1	0	0	0	0	1

Bit [7:5] Reserved
Bit [4:3] **CLKSEL** [1:0] : Clock source select bits
11 = reserved
10 = external clock
01 = RC oscillator
00 =crystal or resonator oscillator
Bit 2 **LVREN**: Low voltage reset enable bit
0 = LVR is disabled
1 = LVR is enabled
Bit 1 **WDTEN**: Watchdog enable bit
0 = WDT is disabled
1 = WDT is enabled
Bit 0 Reserved

2). Secure Register (P_SECU, $7FE2)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	-	-	-	-	-	RCOUT	IOINIT
ACCESS	R	R	R	R	R	R	R	R
DEFAULT	1	1	1	1	1	1	1	1

Bit [7:2] Reserved
Bit 1 **RCOUT**: RC oscillator output enable bit
1 = clock output at XO pin.
0 = no output
Bit 0 **IOINIT**: GPIO initial setting selection bit
1 = all of GPIO float initially

0 = all of GPIO pull low initially.

3). NMI Selection Register (P_NMI, $7FE3)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	-	-	-	-	NMIS2	NMIS1	NMIS0
ACCESS	R	R	R	R	R	R	R	R
DEFAULT	1	1	1	1	1	1	1	1

Bit [7:3] Reserved

Bit [2:0] **NMIS** [2:0]: Non-maskable Interrupt source control bits

111 = disable

110 = reserved

101 = reserved

100 = Reserved

011 = PD1 (INT3) is the NMI source

010 = PD0 (INT2) is the NMI source

001 = PB5 (INT1) is the NMI source

000 = PB4 (INT0) is the NMI source

5.14. Alphabetical List of Instruction Set

NO.	MNEMONIC	OP CODE	BYTE NO	CYCLE NO	OPERATION	FLAG NV-BDIZC
1.	ADC #dd	69	2	2	Add to accumulator with carry. A ← (A) + (M) + C If D-flag set to 1, the ADC performs decimal operation.	NV--D-ZC
2.	ADC aa	65	2	3		
3.	ADC aa, X	75	2	4		
4.	ADC aaaa	6D	3	4		
5.	ADC aaaa,X	7D	3	4		
6.	ADC aaaa,Y	79	3	4		
7.	ADC (aa,X)	61	2	6		
8.	ADC (aa), Y	71	2	5		
9.	AND #dd	29	2	2	And memory data with accumulator. A ← (A) ^ (M)	N-----Z-
10.	AND aa	25	2	3		
11.	AND aa, X	35	2	4		
12.	AND aaaa	2D	3	4		
13.	AND aaaa,X	3D	3	4		
14.	AND aaaa,Y	39	3	4		
15.	AND (aa,X)	21	2	6		
16.	AND (aa), Y	31	2	5		
17.	ASL A	0A	1	2	Arithmetic Shift Left	N-----ZC
18.	ASL aa	06	2	5		
19.	ASL aa,X	16	2	6		
20.	ASL aaaa	0E	3	6		
21.	ASL aaaa,X	1E	3	7		
22.	BCC ??	90	2	2*	Branch if carry bit clear If (C) = 0, then pc ← (pc) + ??	--------
23.	BCS ??	B0	2	2*	Branch if carry bit set If (C) = 1, then pc ← (pc) + ??	--------
24.	BEQ ??	F0	2	2*	Branch if equal If (Z) = 1, then pc ← (pc) + ??	--------
25.	BIT aa	24	2	3	Test bit in memory with accumulator Z ← (A) ^ (M), N ← (M_7), V ← (M_6)	NV----Z-
26.	BIT aaaa	2C	3	4		
27.	BMI ??	30	2	2*	Branch if minus If (N) = 1, then pc ← (pc) + ??	--------
28.	BNE ??	D0	2	2*	Branch if not equal If (Z) = 0, then pc ← (pc) + ??	--------
29.	BPL ??	10	2	2*	Branch if plus If (N) = 0, then pc ← (pc) + ??	--------
30.	BVC ??	50	2	2*	Branch if overflow bit clear If (V) = 0, then pc ← (pc) + ??	--------
31.	BVS ??	70	2	2*	Branch if overflow bit set If (V) = 1, then pc ← (pc) + ??	--------
32.	CLC	18	1	2	Clear C-flag : C ← “0”	-------0
33.	CLD	D8	1	2	Clear D-flag : D ← “0”	----0---
34.	CLI	58	1	2	Clear I-flag : I ← “0”	-----0--
35.	CLR aa,0	0F	2	5	Bit clear	--------

NO.	MNEMONIC	OP CODE	BYTE NO	CYCLE NO	OPERATION	FLAG NV-BDIZC
36.	CLR aa,1	1F	2	5	(M.bit) ← “0”	
37.	CLR aa,2	2F	2	5		
38.	CLR aa,3	3F	2	5		
39.	CLR aa,4	4F	2	5		
40.	CLR aa,5	5F	2	5		
41.	CLR aa,6	6F	2	5		
42.	CLR aa,7	7F	2	5		
43.	CLV	B8	1	2	Clear V-flag : V ← “0”	-0------
44.	CMP #dd	C9	2	2	Compare memory data with accumulator,	
45.	CMP aa	C5	2	3	(A) – (M)	
46.	CMP aa, X	D5	2	4		
47.	CMP aaaa	CD	3	4		
48.	CMP aaaa,X	DD	3	4		N-----ZC
49.	CMP aaaa,Y	D9	3	4		
50.	CMP (aa,X)	C1	2	6		
51.	CMP (aa), Y	D1	2	5		
52.	CPX #dd	E0	2	2	Compare memory data with X-register,	
53.	CPX aa	E4	2	3	(X) – (M)	N-----ZC
54.	CPX aaaa	EC	3	4		
55.	CPY #dd	C0	2	2	Compare memory data with Y-register,	
56.	CPY aa	C4	2	3	(Y) – (M)	N-----ZC
57.	CPY aaaa	CC	3	4		
58.	DEC aa	C6	2	5	Decrement	
59.	DEC aa, X	D6	2	6	M ← (M) - 1	
60.	DEC aaaa	CE	3	6		
61.	DEC aaaa,X	DE	3	7		N-----Z-
62.	DEX	CA	1	2		
63.	DEY	88	1	2		
64.	EOR #dd	49	2	2	Exclusive OR	
65.	EOR aa	45	2	3	A ← (A) ⊕ (M)	
66.	EOR aa, X	55	2	4		
67.	EOR aaaa	4D	3	4		
68.	EOR aaaa,X	5D	3	4		N-----Z-
69.	EOR aaaa,Y	59	3	4		
70.	EOR (aa,X)	41	2	6		
71.	EOR (aa), Y	51	2	5		
72.	INC aa	E6	2	5	Increment	
73.	INC aa, X	F6	2	6	M ← (M) + 1	
74.	INC aaaa	EE	3	6		N-----Z-
75.	INC aaaa,X	FE	3	7		
76.	INV aa,0	87	2	5	Bit inverse	--------
77.	INV aa,1	97	2	5	(M.bit) ← ~(M.bit)	
78.	INV aa,2	A7	2	5		
79.	INV aa,3	B7	2	5		

NO.	MNEMONIC	OP CODE	BYTE NO	CYCLE NO	OPERATION	FLAG NV-BDIZC
80.	INV aa,4	C7	2	5		
81.	INV aa,5	D7	2	5		
82.	INV aa,6	E7	2	5		
83.	INV aa,7	F7	2	5		
84.	INX	E8	1	2	X ← X + 1	N-----Z-
85.	INY	C8	1	2	Y ← Y + 1	N-----Z-
86.	JMP aaaa	4C	3	3	Unconditional jump	--------
87.	JMP (aaaa)	6C	3	5	Pc ← jump address	
88.	JSR aaaa	20	3	6	Jump to subroutine (sp) ← (pc_H), sp ← sp – 1, (sp) ← (pc_L), sp ← sp – 1, pc ← aaaa	--------
89.	LDA #dd	A9	2	2	Load accumulator	N-----Z-
90.	LDA aa	A5	2	3	A ← (M)	
91.	LDA aa, X	B5	2	4		
92.	LDA aaaa	AD	3	4		
93.	LDA aaaa,X	BD	3	4		
94.	LDA aaaa,Y	B9	3	4		
95.	LDA (aa,X)	A1	2	6		
96.	LDA (aa), Y	B1	2	5		
97.	LDX #dd	A2	2	2	Load X-register	N-----Z-
98.	LDX aa	A6	2	3	X ← (M)	
99.	LDX aa, Y	B6	2	4		
100.	LDX aaaa	AE	3	4		
101.	LDX aaaa,Y	BE	3	4		
102.	LDY #dd	A0	2	2	Load Y-register	N-----Z-
103.	LDY aa	A4	2	3	Y ← (M)	
104.	LDY aa, X	B4	2	4		
105.	LDY aaaa	AC	3	4		
106.	LDY aaaa,X	BC	3	4		
107.	LSR A	4A	1	2	Logical shift right	N-----ZC
108.	LSR aa	46	2	5		
109.	LSR aa, X	56	2	6		
110.	LSR aaaa	4E	3	6		
111.	LSR aaaa,X	5E	3	7		
112.	NOP	EA	1	2	No operation	--------
113.	ORA #dd	09	2	2	Logical OR	N-----Z-
114.	ORA aa	05	2	3	A ← (A) v (M)	
115.	ORA aa, X	15	2	4		
116.	ORA aaaa	0D	3	4		
117.	ORA aaaa,X	1D	3	4		
118.	ORA aaaa,Y	19	3	4		
119.	ORA (aa,X)	01	2	6		
120.	ORA (aa), Y	11	2	5		
121.	PHA	48	1	3	(sp) ← A, sp ← sp - 1	--------

NO.	MNEMONIC	OP CODE	BYTE NO	CYCLE NO	OPERATION	FLAG NV-BDIZC
122.	PHP	08	1	3	(sp) ← P status, sp ← sp -1	
123.	PLA	68	1	4	sp ← sp +1, A ← (sp)	--------
124.	PLP	28	1	4	Sp ← sp +1, P status ← (sp)	restored
125.	ROL A	2A	1	2	Rotate left through carry	N-----ZC
126.	ROL aa	26	2	5		
127.	ROL aa, X	36	2	6		
128.	ROL aaaa	2E	3	6		
129.	ROL aaaa,X	3E	3	7		
130.	ROR A	6A	1	2	Rotate right through carry	N-----ZC
131.	ROR aa	66	2	5		
132.	ROR aa, X	76	2	6		
133.	ROR aaaa	6E	3	6		
134.	ROR aaaa,X	7E	3	7		
135.	RTI	40	1	6	Return from interrupt Sp ← sp + 1, P status ← (sp), sp ← sp + 1, pc_L ← (sp), sp ← sp +1, pc_H ← (sp)	restored
136.	RTS	60	1	6	Return from subroutine Sp ← sp + 1, pc_L ← (sp), sp ← sp +1, pc_H ← (sp)	--------
137.	SBC #dd	E9	2	2	Subtract with carry A ← (A) – (M) - ~(C)	NV----ZC
138.	SBC aa	E5	2	3		
139.	SBC aa, X	F5	2	4		
140.	SBC aaaa	ED	3	4		
141.	SBC aaaa,X	FD	3	4		
142.	SBC aaaa,Y	F9	3	4		
143.	SBC (aa,X)	E1	2	6		
144.	SBC (aa), Y	F1	2	5		
145.	SEC	38	1	2	Set C-flag：C ← “1”	-------1
146.	SED	F8	1	2	Set D-flag：D ← “1”	----1---
147.	SEI	78	1	2	Set I-flag：I ← “1”	-----1--
148.	SET aa,0	8F	2	5	Bit set (M.bit) ← “1”	--------
149.	SET aa,1	9F	2	5		
150.	SET aa,2	AF	2	5		
151.	SET aa,3	BF	2	5		
152.	SET aa,4	CF	2	5		
153.	SET aa,5	DF	2	5		
154.	SET aa,6	EF	2	5		
155.	SET aa,7	FF	2	5		
156.	STA aa	85	2	3	Store accumulator in memory (M) ← A	--------
157.	STA aa, X	95	2	4		
158.	STA aaaa	8D	3	4		
159.	STA aaaa,X	9D	3	5		
160.	STA aaaa,Y	99	3	5		
161.	STA (aa,X)	81	2	6		

NO.	MNEMONIC	OP CODE	BYTE NO	CYCLE NO	OPERATION	FLAG NV-BDIZC
162.	STA (aa), Y	91	2	6		
163.	STX aa	86	2	3	Store X-register in memory	
164.	STX aa, Y	96	2	4	(M) ← X	--------
165.	STX aaaa	8E	3	4		
166.	STY aa	84	2	3	Store Y-register in memory	
167.	STY aa, X	94	2	4	(M) ← Y	--------
168.	STY aaaa	8C	3	4		
169.	TAX	AA	1	2	Transfer accumulator to X-register : X ← A	N-----Z-
170.	TAY	A8	1	2	Transfer accumulator to Y-register : Y ← A	N-----Z-
171.	TST aa,0	07	2	3	Bit test	
172.	TST aa,1	17	2	3	Z ← ~(M.bit)	
173.	TST aa,2	27	2	3		
174.	TST aa,3	37	2	3		
175.	TST aa,4	47	2	3		------Z-
176.	TST aa,5	57	2	3		
177.	TST aa,6	67	2	3		
178.	TST aa,7	77	2	3		
179.	TSX	BA	1	2	Transfer sp to X-register : X ← sp	N-----Z-
180.	TXA	8A	1	2	Transfer X-register to accumulator : A ← X	N-----Z-
181.	TXS	9A	1	2	Transfer X-register to sp : sp ← X	N-----Z-
182.	TYA	98	1	2	Transfer Y-register to accumulator : A ← Y	N-----Z-

* means the instruction cycle adds 1 if branch occurs.

6. ELECTRICAL CHARACTERISTICS

6.1. Absolute Maximum Rating (VSS = 0)

Characteristics	Symbol	Min.	Typ.	Max.	Unit	Condition
Voltage rating on VDD	VDD	-0.3	-	6.0	V	
Voltage rating on input	V_{IN}	-0.3	-	VDD +0.3	V	
Current into Vdd Pin	I_{VDD}	-	-	100	mA	
Current out of Vss Pin	I_{VSS}	-	-	120	mA	
Current soured by each I/O port	I_{OHR}	-	-	15	mA	
Current sunk by each I/O port	I_{OLR}	-	-	20	mA	
Power dissipation	PD	-	-	350	mW	Ta=85℃
Storage temperature	T_{STR}	-55	-	125	ºC	

Note: Stresses beyond those given in the Absolute Maximum Rating table may cause operational errors or damage to the device. For normal operational conditions see AC/DC Electrical Characteristics.

6.2. Recommended Operating Conditions

Characteristics	Symbol	Min.	Typ.	Max.	Unit	Condition
Operating supply voltage	VDD	3.0	-	5.5	V	When LVR is disabled
	VDD	LVR	-	5.5	V	When LVR is enabled
System clock	F_{SYS}	200K	-	8.0M	Hz	VDD = 3.0~5.5V
		200K	-	4.0M		VDD = 2.4~5.5V
Operating ambient temperature	T_{OPR}	-40	-	85	ºC	

Note: F_{SYS}= ½ x Fosc, Fosc: Input crystal frequency.

6.3. DC/AC Electrical Characteristics (VDD = 5.0V, T_A = -40ºC~85ºC)

6.3.1. Item Definition

Symbol	Definition	Symbol	Definition
V_{IH}	Input high voltage	I_{OH}	Output high current (source)
V_{IL}	Input low voltage	I_{OL}	Output low current (sink)
V_{TH}	Input threshold voltage	I_Z	Input leakage current
I_P	Pull-up/down current		

6.3.2. PIN Attribute Description

Mnemonic	Description	Symbol	Min.	Typ.	Max.	Unit	Condition
VDD,VSS	Supply current in Normal mode	I_{DD}	-	10.0	15.0	mA	F_{OSC} = 16.0MHz @ VDD = 5.5V
	Supply current in HALT mode	I_{HALT}		8.0	12.0	mA	VDD = 5.5V
	Supply current in STOP mode	I_{PD}	-	8.0	15	μA	VDD = 5.5V
XI,XO	Oscillation stabilizing time	t_{ST}	-	1024	-	F_{SYS}	If crystal mode selected
	RC clock frequency deviation	DFv	-15	-	+15	%	If RC mode selected
	External capacitance	C_{RC}	20	-	100	pf	If RC mode selected
	External resistor	R_{RC}	2.0	-	-	kΩ	If RC mode selected
	High level clock pulse width	t_H	31.25	-	-	ns	If external clock selected
	Low level clock pulse width	t_L	31.25	-	-	ns	If external clock selected
PA[7:0],	Input:	V_{IH}	3.5	-	-	V	

Mnemonic	Description	Symbol	Min.	Typ.	Max.	Unit	Condition
PB[5:0], PC[3:0], PD[2:0],	a. Schmitt trigger input	V_{IL}	-	-	1.4	V	
	b. Pull-up/down/floating option	V_{TH}	-	0.5	-	V	
	Output:	I_{OH}	4.0	-	-	mA	V_{OH} = 4.5V
	a. LED driving capability	I_{OL}	10.0	-	-	mA	V_{OL} = 0.5V
	b. 4mA/10mA driving output.	I_P	-	50	-	µA	V_{IN} = VDD or VSS
PB[7:6]	Input:	V_{IH}	3.5	-	-	V	
	a. Schmitt trigger input	V_{IL}	-	-	1.4	V	
	b. Pull-up/down/floating option	V_{TH}	-	0.5	-	V	
	Output:	I_{OH}	4.0	-	-	mA	V_{OH} = 4.5V
	a. LED driving capability	I_{OL}	20.0	-	-	mA	V_{OL} = 0.5V
	b. 4mA/20mA driving output.	I_P	-	50	-	µA	V_{IN} = VDD or VSS
All Port	I/O Port input Hi-Z leakage	I_Z	-	-	10	µA	
RESETB	Input:	V_{IH}	3.5	-	-	V	
	a. Schmitt trigger input	V_{IL}	-	-	1.4	V	
	b. Pull-Up input	V_{TH}	-	0.5	-	V	
	Output: none	I_P	-	125	-	µA	V_{IN} = VSS
		t_W	1.0	-	-	µS	Input minimum width

6.4. Analog Interface Electrical Characteristics (VDD = 5.0V, T_A = -40ºC~85ºC)

Mnemonic	Description		Symbol	Min.	Typ.	Max.	Unit	Condition
10-bit A/D Converter	Resolution		N_{R_AD}			10	Bit	
	Top Reference Voltage		V_{RT}	3		VDD	V	If PB7 act as VRT
	Top Reference Voltage Supply Current		I_{RT}			800	uA	If PB7 act as VRT
	Analog Input Voltage		V_{AIN}	0		VDD/V_{RT}	V	
	Conversion Rate		F_{AD}			100	KHz	VDD=5.0V (Note 1)
	Analog Input Impedance		R_{AIN}					(Note 2)
	Accuracy	Integral Linearity Error	E_{INL_AD}			±4.0	LSB	
		Differential Linearity Error	E_{DNL_AD}			±4.0	LSB	
		Zero Offset Error	E_{ZOE_AD}			±1.5	LSB	
		Full Scale Error	E_{FSE_AD}			±2.5	LSB	
		Total Error	E_{ALL_AD}			±4.0	LSB	
8-bit A/D Converter	Resolution		N_{R_AD}			8	Bit	
	Top Reference Voltage		V_{RT}	3		VDD	V	If PB7 act as VRT
	Top Reference Voltage Supply Current		I_{RT}			800	uA	If PB7 act as VRT
	Analog Input Voltage		V_{AIN}	0		VDD/V_{RT}	V	
	Conversion Rate		F_{AD}			100	KHz	VDD=5.0V (Note 1)
	Analog Input Impedance		R_{AIN}					(Note 2)
	Accuracy	Integral Linearity Error	E_{INL_AD}			±1.0	LSB	
		Differential Linearity Error	E_{DNL_AD}			±1.0	LSB	
		Zero Offset Error	E_{ZOE_AD}			±1.0	LSB	
		Full Scale Error	E_{FSE_AD}			±1.0	LSB	
		Total Error	E_{ALL_AD}			±1.0	LSB	
Low Voltage Reset	Low voltage reset if LV40=0		V_{LVR}	2.3	2.5	2.7	V	
	Low voltage reset if LV40=1		V_{LVR}	3.68	4.0	4.32	V	

Note 1: Due to the ADC's sample capacitance, input impedance, and external input circuitry, there will be a settling time required for the sample capacitor to assume the measured input signal voltage. The ADC specification for minimum settling time is 1.5uSec, which is more restrictive in most cases.

Note 2:

(1) The A/D Converter used for the SPMC65P2404A/2408A contains a sample hold circuit as illustrated below to fetch analog input voltage into the sample hold capacitor after activation A/D conversion. (Note this is a simplified representation of the ADC circuit in sampling mode)

(2) For the reason, if the output impedance of the external circuit for the analog input is high, analog input voltage might not stabilize within the analog input sampling period (1.5uSec). Therefore, it is recommended to keep the output impedance of the external circuit low.

(3) Note that if the impedance cannot be kept low, it is recommended to connect an external capacitor of about 0.01uF to 0.1uF for the analog input pin.

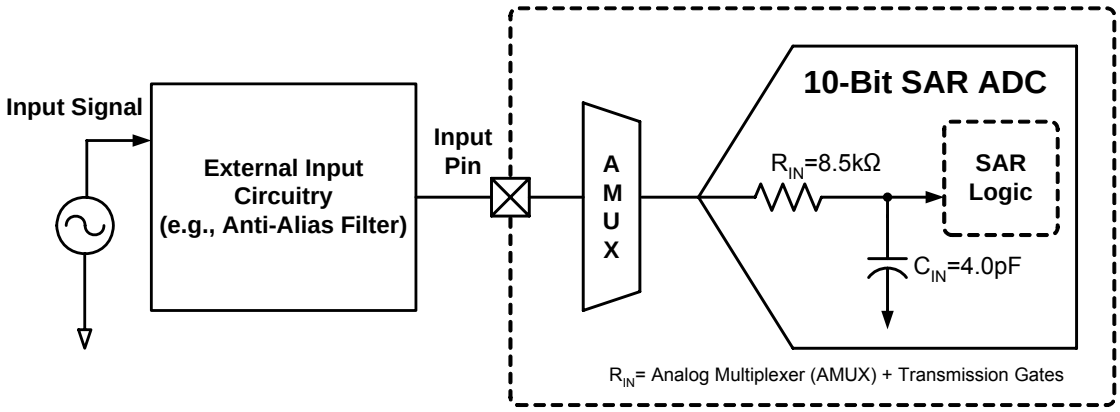


Fig. 1. Equivalent ADC Circuit for Estimating Settling Time

7. PACKAGE/PAD LOCATIONS

7.1. PAD Assignment and Locations

Please contact Sunplus sales representatives for more information.

7.2. Ordering Information

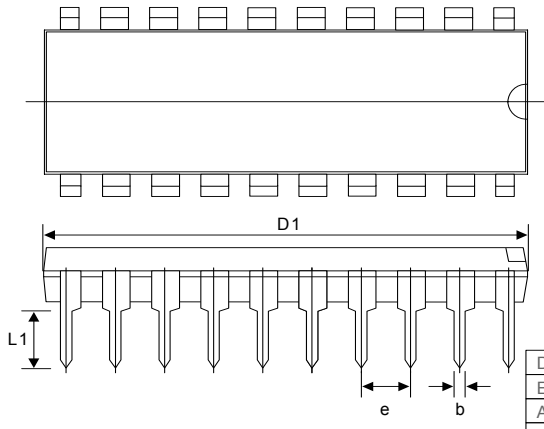
Product Number	Package Type
SPMC65P2404A-NnnV-PS07	Package form – SOP 20 (300mil)
SPMC65P2404A-NnnV-PS08	Package form – SOP 28 (300mil)
SPMC65P2404A-NnnV-PD05	Package form – PDIP 20 (300mil)
SPMC65P2404A-NnnV-PD08	Package form – PDIP 28 (300mil)
SPMC65P2408A-NnnV-PS05	Package form – SOP 28 (300mil)
SPMC65P2408A-NnnV-PS08	Package form – SOP 32 (445mil)
SPMC65P2408A-NnnV-PD08	Package form – PDIP 28 (300mil)
SPMC65P2408A-NnnV-PD10	Package form – PDIP 32 (600mil)

Note1: Code number is assigned for customer.

Note2: Code number (N = A - Z or 0 - 9, nn = 00 - 99); version (V = A - Z).

7.3. Package Information

7.3.1. PDIP 20 (300mil)



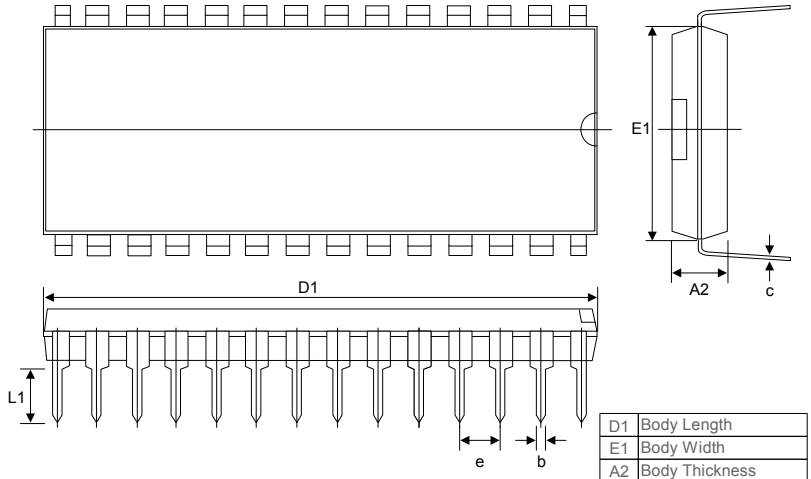
Body Size			Lead Size			
D1	E1	A2	L1	b	c	e
1020	250	130	115	18	10	100

All units are in mil. 1mil = 25.4μm

D1	Body Length
E1	Body Width
A2	Body Thickness
L1	Lead Length
b	Lead Width
c	Lead Thickness
e	Lead Pitch

PDIP-20-300

7.3.2. PDIP 28 (300mil)



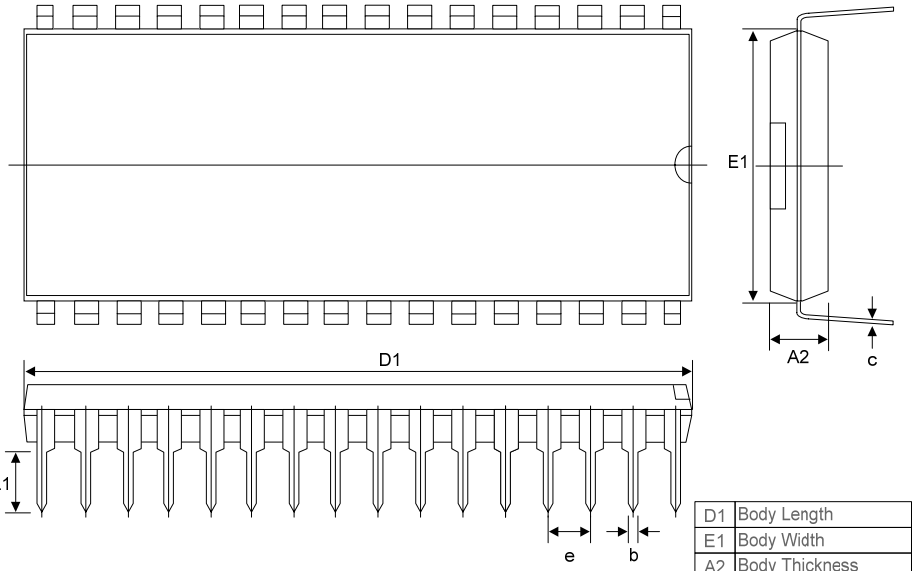
Body Size			Lead Size			
D1	E1	A2	L1	b	c	e
1388	290	130	115	18	10	100

All units are in mil. 1mil = 25.4μm

D1	Body Length
E1	Body Width
A2	Body Thickness
L1	Lead Length
b	Lead Width
c	Lead Thickness
e	Lead Pitch

PDIP-28-300

7.3.3. PDIP 32 (600mil)



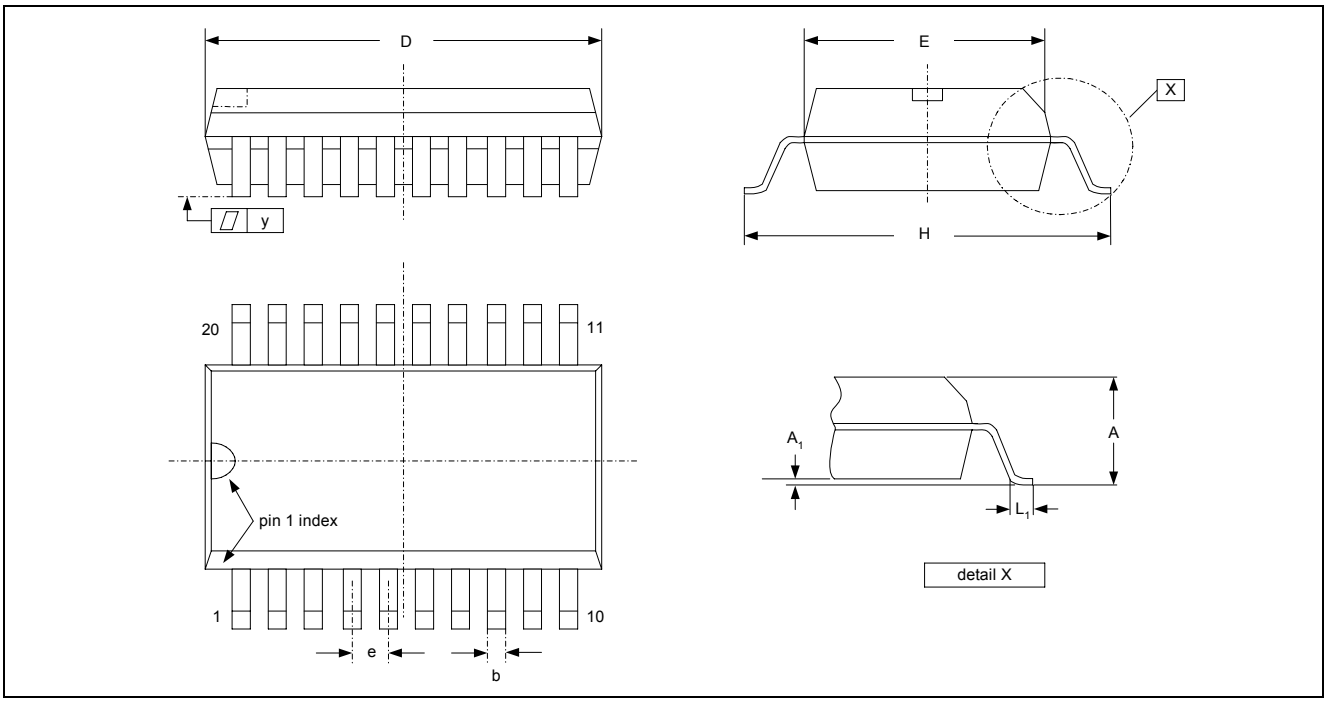
Body Size			Lead Size			
D1	E1	A2	L1	b	c	e
1650	550	150	100	18	10	100

All units are in mil. 1mil = 25.4μm

D1	Body Length
E1	Body Width
A2	Body Thickness
L1	Lead Length
b	Lead Width
c	Lead Thickness
e	Lead Pitch

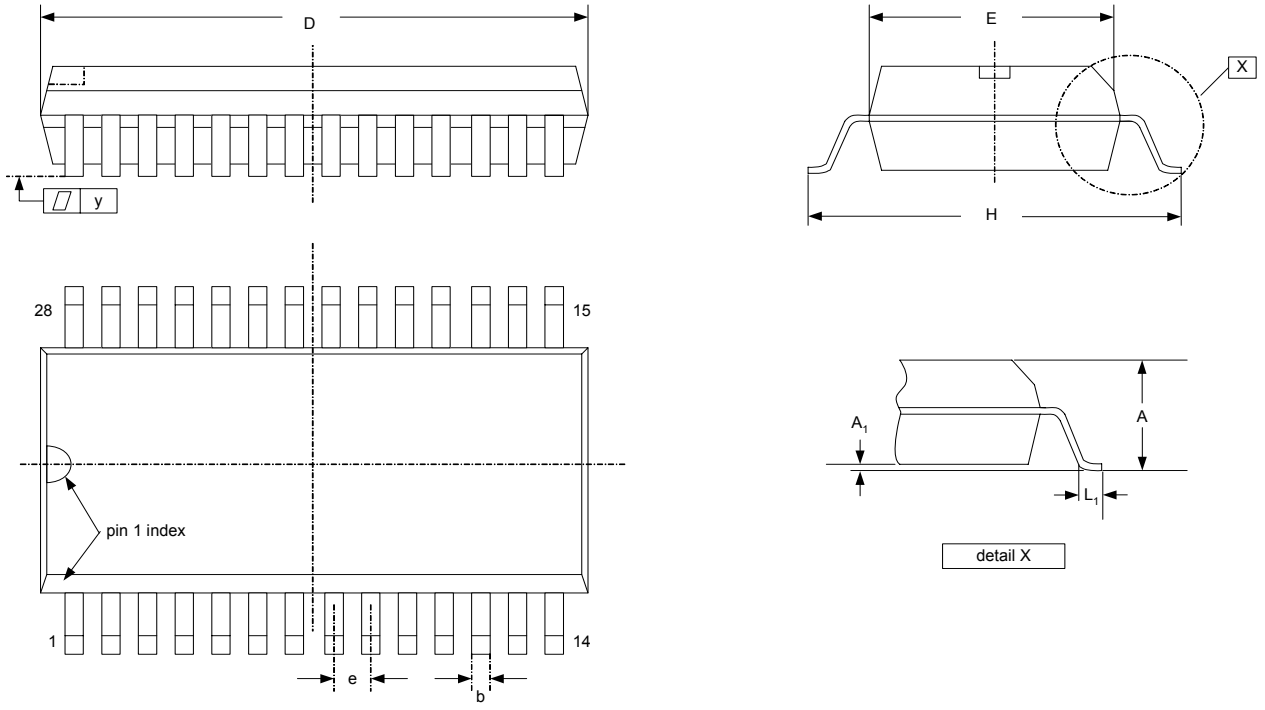
PDIP-32-600

7.3.4. SOP 20 (300mil)



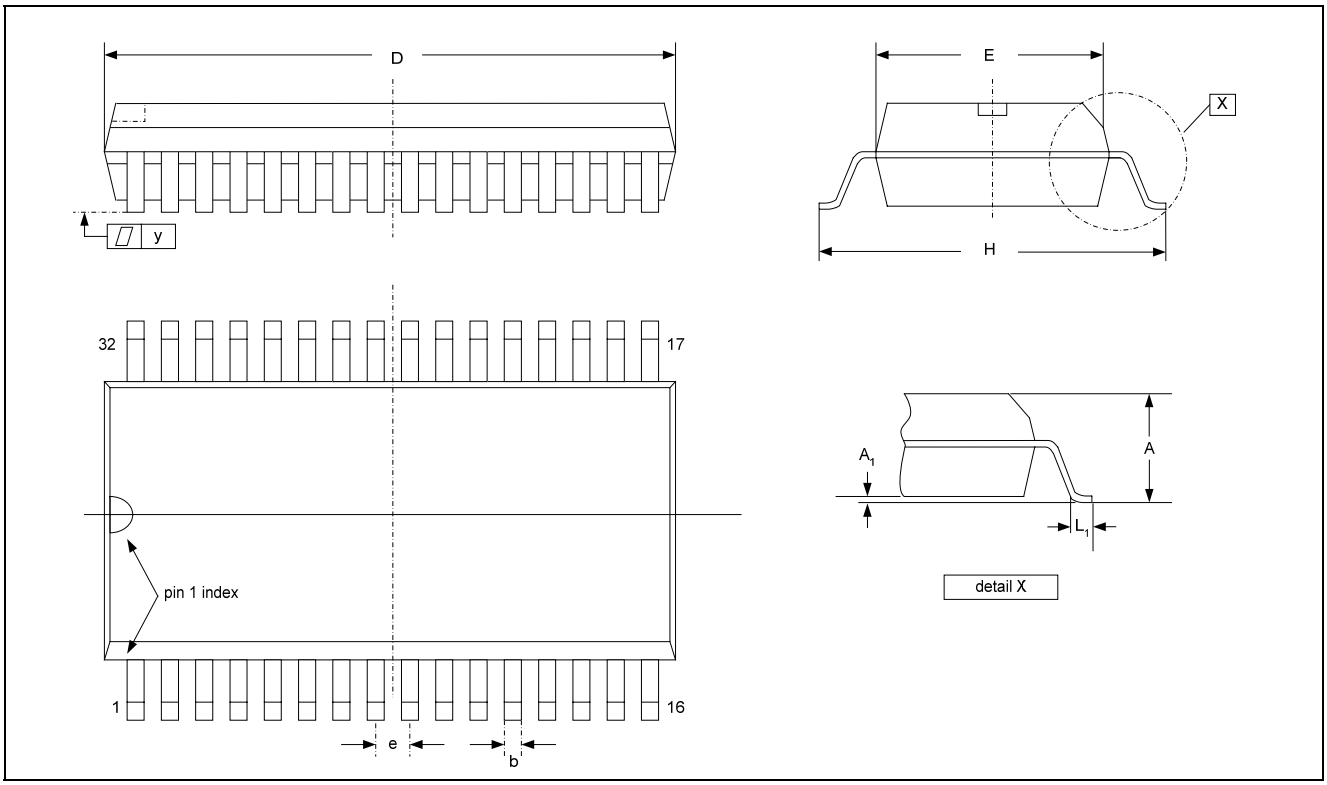
Symbol	Dimension in inch		
	Min.	Typ.	Max.
A	0.093	-	0.104
A_1	0.004	-	0.012
B	-	0.016	-
D	0.496	-	0.508
E	0.291	-	0.299
e	-	0.050	-
H	0.394	-	0.419
L_1	0.016	-	0.050
y	-	-	0.004

7.3.5. SOP 28 (300mil)



Symbol	Dimension in inch		
	Min.	Typ.	Max.
A	0.093	-	0.104
A1	0.004	-	0.012
b	-	0.016	-
D	0.697	-	0.713
E	0.291	-	0.299
e	-	0.050	-
H	0.394	-	0.419
L1	0.016	-	0.050
y	-	-	0.004

7.3.6. SOP 32 (445mil)



Symbol	Dimension in inch		
	Min.	Typ.	Max.
A	-	-	0.120
A1	0.004	-	0.014
b	-	0.016	-
D	0.799	-	0.815
E	0.437	-	0.450
e	-	0.05	-
H	0.530	-	0.580
L1	0.016	-	0.050
y	-	-	0.004

7.4. Storage Condition and Period for Package

Package	Moisture sensitivity level	Max. Reflow temperature	Floor life storage condition	Dry pack
SOP	LEVEL 3	220 +5/-0℃	168Hrs @ ≦30℃/ 60% R.H.	No
PDIP	LEVEL 3 (Reference)	220 +5/-0℃ (Reference)	N/A	No

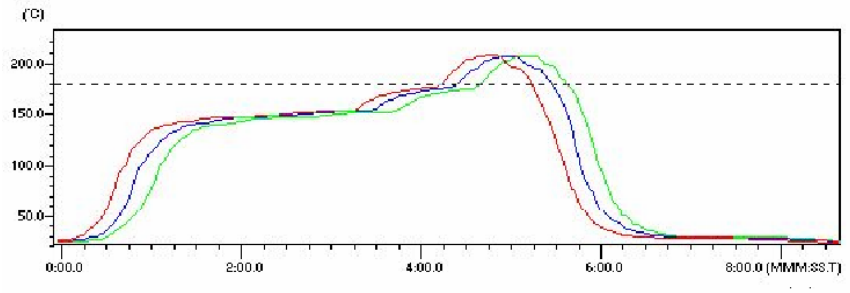
Note1: Please refer to IPC/JEDEC standard J-STD-020A and EIA JEDEC stand JFSD22-A112

Note2: or refer to the "CAUTION Note" on dry pack bag.

7.5. Recommended SMT Temperature Profile

This "Recommended" temperature profile is a rough guideline for SMT process reference. Most of SUNPLUS lead-frame base product choice Matte Tin and Sn/Bi for plating recipe. For PPF (Pre-Plated Frame) product with 63/37 solder paste, we recommend 240℃~245℃ for peak temperature.

Recommended Reflow Profile for 63/37 Solder Paste or Cu lead frame

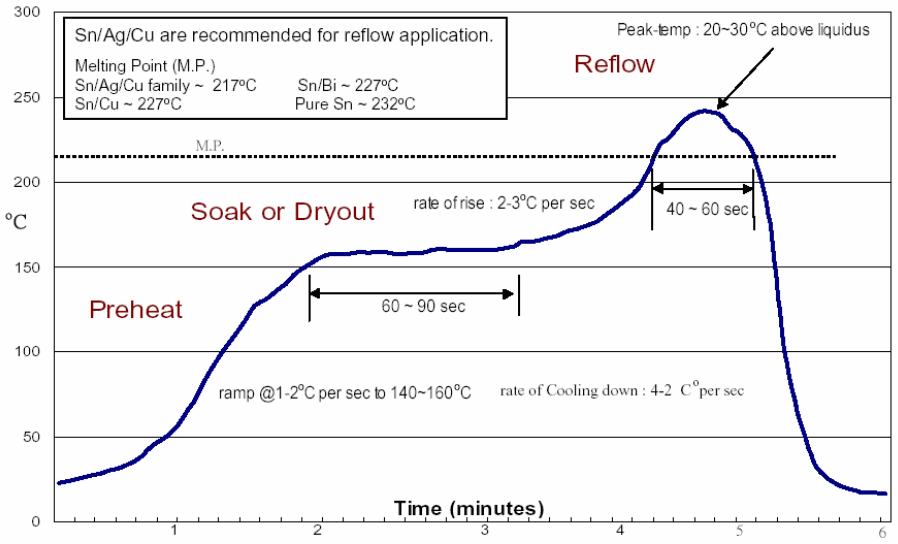


This profile is designed for use with Sn63 or Sn62 and can serve as a general guideline in establishing a reflow profile.

Reflow Profile:

- Heating-up @1~3°C per sec to 140°C
- Preheat @ 140-150°C for 120 ~ 160 sec
- Ramp @ 2~3 °C per sec to peak temperature (220 ~ 225 °C), Temperature over 183°C for 45~ 75 sec
- Cooling down to room temperature @4~2°C per sec to avoid undesired intermatalic compound layer.

Recommended Reflow Profile for Lead-free Solder Paste or PPF lead frame



8. DISCLAIMER

The information appearing in this publication is believed to be accurate.

Integrated circuits sold by Sunplus Technology are covered by the warranty and patent indemnification provisions stipulated in the terms of sale only. SUNPLUS makes no warranty, express, statutory implied or by description regarding the information in this publication or regarding the freedom of the described chip(s) from patent infringement. FURTHERMORE, SUNPLUS MAKES NO WARRANTY OF MERCHANTABILITY OR FITNESS FOR ANY PURPOSE. SUNPLUS reserves the right to halt production or alter the specifications and prices at any time without notice. Accordingly, the reader is cautioned to verify that the data sheets and other information in this publication are current before placing orders. Products described herein are intended for use in normal commercial applications. Applications involving unusual environmental or reliability requirements, e.g. military equipment or medical life support equipment, are specifically not recommended without additional processing by SUNPLUS for such applications. Please note that application circuits illustrated in this document are for reference purposes only.

9. REVISION HISTORY

Date	Revision #	Description	Page
MAR. 03, 2005	1.0	Original	97
MAR. 13, 2005	1.1	Second	97
Nov. 11, 2005	1.2	Third	97