EPROM/ROM-Based 8-Bit Microcontroller

Devices Included in this Data Sheet:

- FM8P51E : EPROM device
- FM8P51 : Mask ROM device

FEATURES

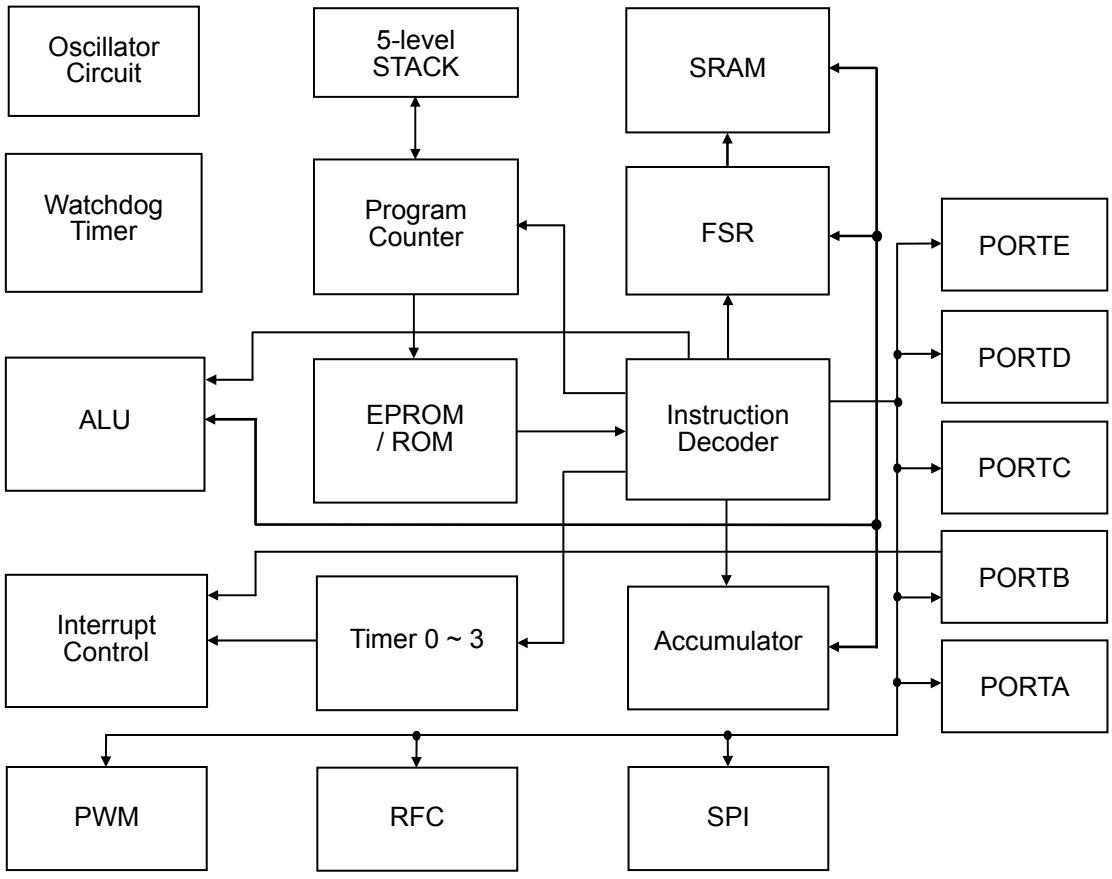
- Only 47 single word instructions
- All instructions are single cycle except for program branches which are two-cycle
- 13-bit wide instructions
- 8-bit wide data path
- 5-level deep hardware stack
- 4K x 13 bits on chip EPROM/ROM
- 140 x 8 bits on chip general purpose registers (SRAM)
- Operating speed: DC-20 MHz clock input
  DC-100 ns instruction cycle
- Direct, indirect addressing modes for data accessing
- One 8-bit real time clock/counter (Timer0) with 8-bit programmable prescaler
- Three 8-bit real time clock/counter (Timer1, Timer2, and Timer3) with period setting
- Internal Power-on Reset (POR)
- Built-in Low Voltage Detector (LVD) for Brown-out Reset (BOR)
- Power-up Reset Timer (PWRT) and Oscillator Start-up Timer(OST)
- On chip Watchdog Timer (WDT) with internal oscillator for reliable operation and soft-ware watch-dog enable/disable control
- Five I/O ports (38 I/O pins) with independent direction control
- 32 programmable pull-high input pins
- 12 wake-up pins
- 2 open-drain pins
- 2 R-option pins
- Clock output with high driving ability
- 19 channels of 15-bit resolution Resistor to Frequency Converter (RFC) output
- 2 channels of maximum 10-bit resolution Pulse Width Modulation (PWM) output
- Built-in 8-bit data comparator
- Seven internal interrupt source: Timer0 overflow, Timer1 match, Timer2 match, Timer3 match, SPI module (Receive & Transmit), RFC module; One external interrupt source: INT pin change
- Wake-up from SLEEP by port input change
- Power saving SLEEP mode
- Programmable Code Protection
- Built-in RC oscillator with external resistor
- Selectable oscillator options:
  - ERC: External Resistor/Capacitor Oscillator
  - ERIC: External Resistor/Internal Capacitor Oscillator
  - HF: High Frequency Crystal/Resonator Oscillator
  - LF: Low Frequency Crystal Oscillator
- Wide-operating voltage range:
  - EPROM : 2.3V to 5.5V
  - ROM : 2.3V to 5.5V

This datasheet contains new product information. Feeling Technology reserves the rights to modify the product specification without notice. No liability is assumed as a result of the use of this product. No rights under any patent accompany the sales of the product.

GENERAL DESCRIPTION

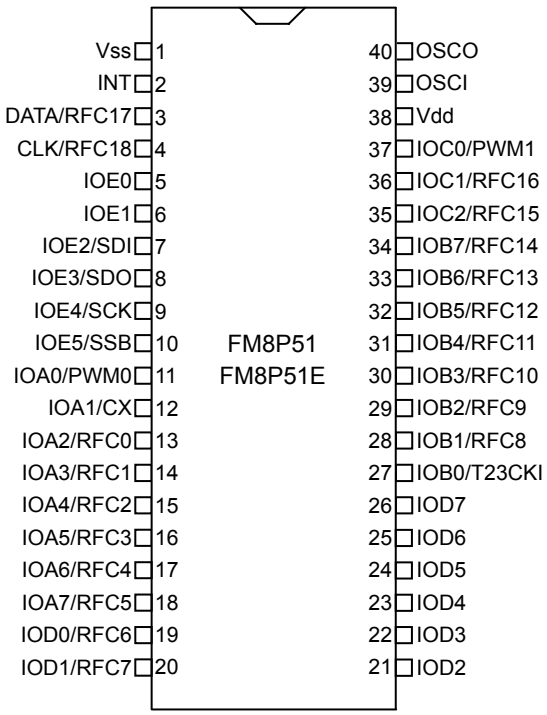
The FM8P51 series is a family of low-cost, high speed, high noise immunity, EPROM/ROM-based 8-bit CMOS microcontrollers. It employs a RISC architecture with only 47 instructions. All instructions are single cycle except for program branches which take two cycles. The easy to use and easy to remember instruction set reduces development time significantly.
The FM8P51 series consists of Power-on Reset (POR), Brown-out Reset (BOR), Power-up Reset Timer (PWRT), Oscillator Start-up Timer(OST), Watchdog Timer, EPROM/ROM, SRAM, tri-state I/O port, I/O pull-high/open-drain control, Power saving SLEEP mode, 4 real time programmable clock/counter, Interrupt, SPI, RFC, data compare, Wake-up from SLEEP mode, and Code Protection for EPROM products. There are four oscillator configurations to choose from, including the power-saving LP (Low Power) oscillator and cost saving RC oscillator.
The FM8P51 series address 4K×13 of program memory.
The FM8P51 series can directly or indirectly address its register files and data memory. All special function registers including the program counter are mapped in the data memory.

BLOCK DIAGRAM

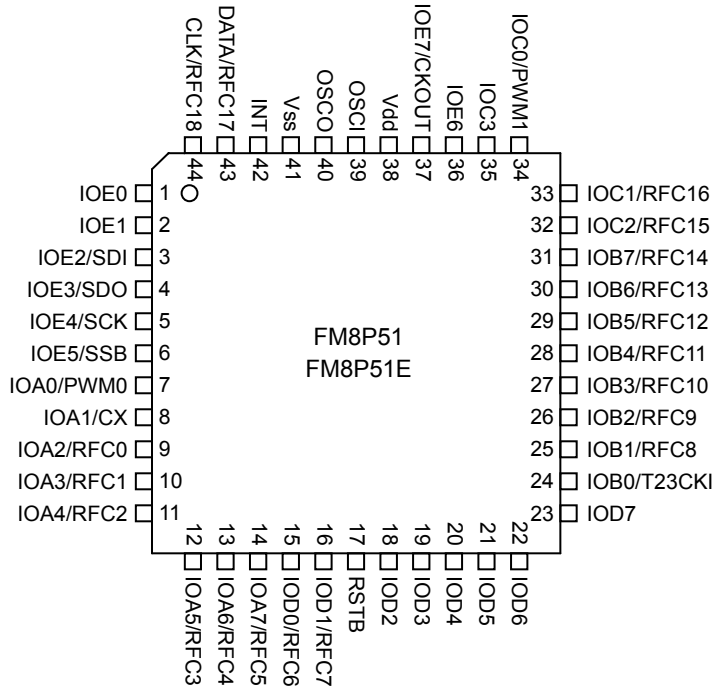


PIN CONNECTION

PDIP



QFP



PIN DESCRIPTIONS

Name	I/O	Description
IOA0 ~ IOA7	I/O	Bi-direction I/O port Software controlled pull-high for all pins
IOB0 ~ IOB7	I/O	Bi-direction I/O port with system wake-up function Software controlled pull-high for all pins
IOC0 ~ IOC2	I/O	Bi-direction I/O pins with driving ability selection
IOC3	I/O	Bi-direction I/O pin
CLK	I/O	By connecting IOC4 and IOC6 together IOC4 is a bi-direction I/O pin with system wake-up and software controlled pull-high function IOC6 is a bi-direction I/O pin with software controlled open-drain output Note : Both IOC4 and IOC6 should not be defined as output pins at the same time.
DATA	I/O	By connecting IOC5 and IOC7 together IOC5 is a bi-direction I/O pin with system wake-up and software controlled pull-high function IOC7 is a bi-direction I/O pin with software controlled open-drain output Note : Both IOC5 and IOC7 should not be defined as output pins at the same time.
IOD0 ~ IOD7	I/O	Bi-direction I/O port Software controlled pull-high for all pins IOD0 and IOD1 are R-option pins
IOE0 ~ IOE1	I/O	Bi-direction I/O pins with system wake-up function Software controlled pull-high for all pins
IOE2 ~ IOE5	I/O	Bi-direction I/O pins All pins can be pulled-high by software
IOE6 ~ IOE7	I/O	Bi-direction I/O pins
INT	I	External interrupt input triggered by falling edge
SDI	I	Serial data in for SPI
SDO	O	Serial data out for SPI
SCK	I/O	Serial clock for SPI
SSB	I	Slave select (active low) for SPI
PWM0 ~ PWM1	O	PWM output pins
RFC0 ~ RFC18	O	The RC oscillator network output of RFC module
CX	I	The RC oscillator network input of RFC module
OSCI	I	X'tal type: Oscillator crystal input RC type: Clock input of RC oscillator
OSCO	O	X'tal type: Oscillator crystal output. RC mode: Outputs with 1/4 the frequency of OSCI to denotes the instruction cycle rate
CKOUT	O	Oscillator frequency output with high driving ability and output phase selection
RSTB	I	System clear (RESET) input. This pin is an active low RESET to the device. Internal weak pull-high.
Vdd	-	Positive supply
Vss	-	Ground

Legend: I=input, O=output, I/O=input/output

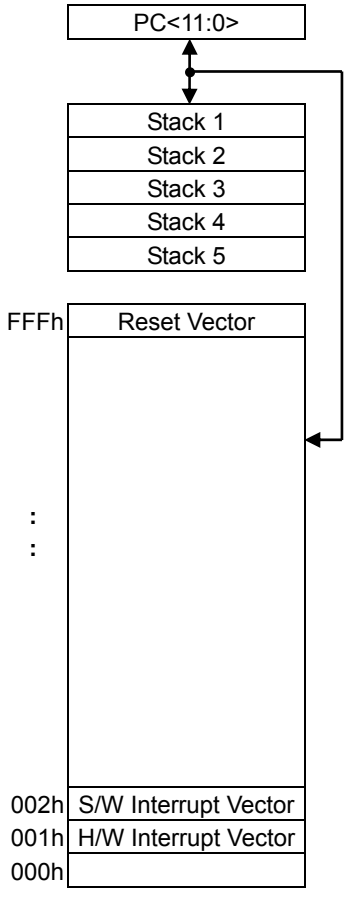
1.0 MEMORY ORGANIZATION

FM8P51 series memory is organized into program memory and data memory.

1.1 Program Memory Organization

The FM8P51 series have an 12-bit Program Counter capable of addressing a 4K×13 program memory space.
The RESET vector for the FM8P51 series is at FFFh.
The H/W interrupt vector is at 001h. And the S/W interrupt vector is at 002h.
FM8P51 series has program memory size greater than 1K words, but the CALL and GOTO instructions only have a 10-bit address range. This 10-bit address range allows a branch within a 1K program memory page size. To allow CALL and GOTO instructions to address the entire 4K program memory address range for FM8P51 series, there is another two bits to specify the program memory page. This paging bit comes from the PG<1:0> bits (STATUS<6:5>). When doing a CALL or GOTO instruction, the user must ensure that page bit PG<1:0> are programmed so that the desired program memory page is addressed. When one of the return instructions is executed, the entire 12-bit PC is POPed from the stack. Therefore, manipulation of the PG <1:0> is not required for the return instructions.

FIGURE 1.1: Program Memory Map and STACK



FM8P51 Series

1.2 Data Memory Organization

Data memory is composed of Special Function Registers and General Purpose Registers.
The General Purpose Registers are accessed either directly or indirectly through the FSR register.
The Special Function Registers are registers used by the CPU and peripheral functions to control the operation of the device.
In FM8P51 series, the data memory is partitioned into four banks. Switching between these banks requires the RP1 and RP0 bits in the FSR register to be configured for the desired bank.

TABLE 1.1: Registers File Map for FM8P51 Series

FSR<7:6> / Address	Description					
	0 0 Bank 0	0 1 Bank 1	1 0 Bank 2	1 1 Bank 3		
00h	INDF	Memory back to address in Bank 0				
01h	TMR0					
02h	PCL				N/A	OPTION
03h	STATUS					
04h	FSR					
05h	PORTA				05h	IOSTA
06h	PORTB				06h	IOSTB
07h	PORTC				07h	IOSTC
08h	PORTD				08h	IOSTD
09h	PORTE				09h	IOSTE
0Ah	SPIRCB	T23CON	PWMCON	RFCCON		
0Bh	SPITXB	TMR2	PW0DCL	RFCDL		
0Ch	SPISTAT	PR2	PW0DCH	RFCDH	0Ch	T1CON
0Dh	SPICON	TMR3	PW1DCL	CMPDX	0Dh	PHCON
0Eh	TMR1	PR3	PW1DCH	CMPDY	0Eh	PCON
0Fh	PR1			CMPSTAT	0Fh	INTEN
10h | 1Fh	General Purpose Registers	Memory back to address in Bank 0				
20h | 3Eh	General Purpose Registers	General Purpose Registers	General Purpose Registers	General Purpose Registers		
3Fh	INTFLAG	Memory back to address in Bank 0				

TABLE 1.2: The Registers Controlled by OPTION/OPTIONR or IOST/IOSTR Instructions

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
Unbanked									
N/A (r/w)	OPTION	/PHBCE	GIE			PSA	PS2	PS1	PS0
05h (r/w)	IOSTA	Port A I/O Control Register							
06h (r/w)	IOSTB	Port B I/O Control Register							
07h (r/w)	IOSTC	Port C I/O Control Register							
08h (r/w)	IOSTD	Port D I/O Control Register							
09h (r/w)	IOSTE	Port E I/O Control Register							
0Ch (r/w)	T1CON						T1ON	T1P1	T1P0
0Dh (r/w)	PHCON	HDC				/PHE	/PHD	/PHB	/PHA
0Eh (r/w)	PCON	LVDTE	ODE	WDTE	-	ROC	-	-	/WUE
0Fh (r/w)	INTEN	SPITXIE	RFCIE	T3IE	T2IE	T1IE	SPIRCIE	INTIE	T0IE

Legend: - = unimplemented, read as '0', * = unimplemented, read as '1'.

TABLE 1.3: Operational Registers Map

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
Unbanked									
00h (r/w)	INDF	Uses contents of FSR to address data memory (not a physical register)							
01h (r/w)	TMR0	8-bit real-time clock/counter							
02h (r/w)	PCL	Low order 8 bits of PC							
03h (r/w)	STATUS	GP	PG1	PG0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
04h (r/w)	FSR	RP1	RP0	Indirect data memory address pointer					
05h (r/w)	PORTA	IOA7	IOA6	IOA5	IOA4	IOA3	IOA2	IOA1	IOA0
06h (r/w)	PORTB	IOB7	IOB6	IOB5	IOB4	IOB3	IOB2	IOB1	IOB0
07h (r/w)	PORTC	IOC7	IOC6	IOC5	IOC4	IOC3	IOC2	IOC1	IOC0
08h (r/w)	PORTD	IOD7	IOD6	IOD5	IOD4	IOD3	IOD2	IOD1	IOD0
09h (r/w)	PORTE	IOE7	IOE6	IOE5	IOE4	IOE3	IOE2	IOE1	IOE0
Bank 0									
0Ah (r/w)	SPIRCB	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0
0Bh (r/w)	SPITXB	TX7	TX6	TX5	TX4	TX3	TX2	TX1	TX0
0Ch (r/w)	SPISTAT		TXBFIF	TXBF	TM1IF	SDOOD	SCKOD	RCBFIF	RCBF
0Dh (r/w)	SPICON	CKEDG	SPION	RCOV	SSE	SSEMOD	SPIM2	SPIM1	SPIM0
0Eh (r/w)	TMR1	TMR17	TMR16	TMR15	TMR14	TMR13	TMR12	TMR11	TMR10
0Fh (r/w)	PR1	PR17	PR16	PR15	PR14	PR13	PR12	PR11	PR10
Bank 1									
0Ah (r/w)	T23CON	-	-	-	T16	T3ON	T3CS	T2ON	T2CS
0Bh (r/w)	TMR2	TMR27	TMR26	TMR25	TMR24	TMR23	TMR22	TMR21	TMR20
0Ch (r/w)	PR2	PR27	PR26	PR25	PR24	PR23	PR22	PR21	PR20
0Dh (r/w)	TMR3	TMR37	TMR36	TMR35	TMR34	TMR33	TMR32	TMR31	TMR30
0Eh (r/w)	PR3	PR37	PR36	PR35	PR34	PR33	PR32	PR31	PR30
0Fh	-	Unimplemented, read as “0”s							
Bank 2									
0Ah (r/w)	PWMCON	-	-	-	-	-	PW1T3	PW1ON	PW0ON
0Bh (r/w)	PW0DCL	DC1	DC0	-	-	-	-	-	-
0Ch (r/w)	PW0DCH	DC9	DC8	DC7	DC6	DC5	DC4	DC3	DC2
0Dh (r/w)	PW1DCL	DC1	DC0	-	-	-	-	-	-
0Eh (r/w)	PW1DCH	DC9	DC8	DC7	DC6	DC5	DC4	DC3	DC2
0Fh	-	Unimplemented, read as “0”s							
Bank 3									
0Ah (r/w)	RFCCON	RFCON	START	-	RFCS4	RFCS3	RFCS2	RFCS1	RFCS0
0Bh (r)	RFCDL	RFCD7	RFCD6	RFCD5	RFCD4	RFCD3	RFCD2	RFCD1	RFCD0
0Ch (r)	RFCDH	RFCOV	RFCD14	RFCD13	RFCD12	RFCD11	RFCD10	RFCD9	RFCD8
0Dh (r/w)	CMPDX	DX7	DX6	DX5	DX4	DX3	DX2	DX1	DX0
0Eh (r/w)	CMPDY	DY7	DY6	DY5	DY4	DY3	DY2	DY1	DY0
0Fh (r)	CMPSTAT	-	-	-	-	CMPF3	CMPF2	CMPF1	CMPF0
Unbanked									
3Fh (r/w)	INTFLAG	SPITXIF	RFCIF	T3IF	T2IF	T1IF	SPIRCIF	INTIF	T0IF

Legend: - = unimplemented, read as ‘0’, * = unimplemented, read as ‘1’.

2.0 FUNCTIONAL DESCRIPTIONS

2.1 Operational Registers

2.1.1 INDF (Indirect Addressing Register)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
00h (r/w)	INDF	Uses contents of FSR to address data memory (not a physical register)							

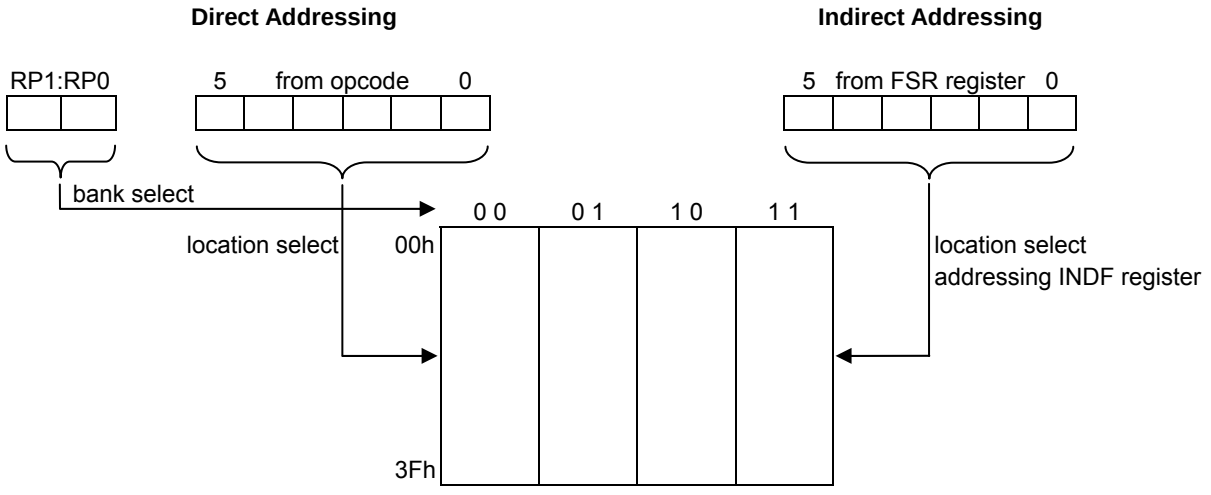
The INDF Register is not a physical register. Any instruction accessing the INDF register can actually access the register pointed by FSR Register. Reading the INDF register itself indirectly (FSR="0") will read 00h. Writing to the INDF register indirectly results in a no-operation (although status bits may be affected).
The bits 5-0 of FSR register are used to select up to 64 registers (address: 00h ~ 3Fh).
In FM8P51 series, the data memory is partitioned into four banks. Switching between these banks requires the RP1 and RP0 bits in the FSR register to be configured for the desired bank. The lower locations of each bank are reserved for the Special Function Registers. Above the Special Function Registers are General Purpose Registers. All Special Function Registers and some of General Purpose Registers from other banks are mirrored in bank 0 for code reduction and quicker access.

Accessed Bank	RP1:RP0
0	0 0
1	0 1
2	1 0
3	1 1

EXAMPLE 2.1: INDIRECT ADDRESSING

- Register file 38 contains the value 10h
- Register file 39 contains the value 0Ah
- Load the value 38 into the FSR Register
- A read of the INDF Register will return the value of 10h
- Increment the value of the FSR Register by one (@FSR=39h)
- A read of the INDR register now will return the value of 0Ah.

FIGURE 2.1: Direct/Indirect Addressing for FM8P51 Series



2.1.2 TMR0 (Time Clock/Counter register)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
01h (r/w)	TMR0	8-bit real-time clock/counter							

The Timer0 is a 8-bit timer/counter. The clock source of Timer0 can come from the instruction cycle clock.
The prescaler is assigned to Timer0 by clearing the PSA bit (OPTION<3>). In this case, the prescaler will be cleared when TMR0 register is written with a value.

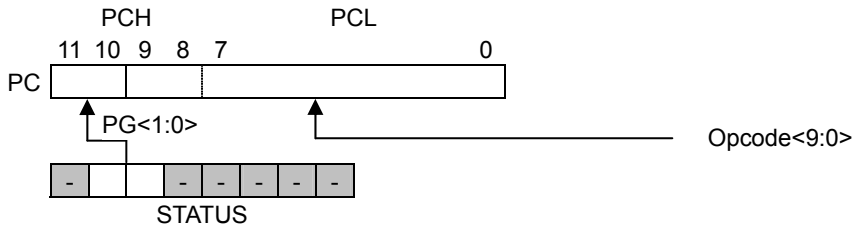
2.1.3 PCL (Low Bytes of Program Counter) & Stack

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
02h (r/w)	PCL	Low order 8 bits of PC							

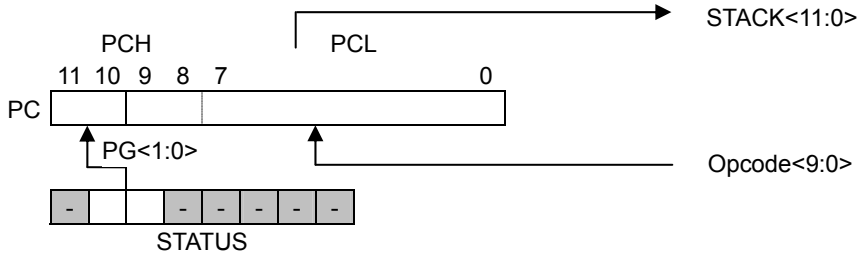
FM8P51 devices have a 12-bit wide Program Counter (PC) and five-level deep 12-bit hardware push/pop stack. The low byte of PC is called the PCL register. This register is readable and writable. The high byte of PC is called the PCH register. This register contains the PC<11:8> bits and is not directly readable or writable. All updates to the PCH register go through the PG<1:0> bits (STATUS<6:5>). As a program instruction is executed, the Program Counter will contain the address of the next program instruction to be executed. The PC value is increased by one, every instruction cycle, unless an instruction changes the PC.
For a GOTO instruction, the PC<9:0> is provided by the GOTO instruction word. The PC<11:10> is updated from the PG<1:0> bits (STATUS<6:5>). The PCL register is mapped to PC<7:0>.
For a CALL instruction, the PC<9:0> is provided by the CALL instruction word. The PC<11:10> is updated from the PG<1:0> bits (STATUS<6:5>). The next PC will be loaded (PUSHed) onto the top of STACK. The PCL register is mapped to PC<7:0>.
For a RETIA, RETFIE, or RETURN instruction, the PC are updated (POPed) from the top of STACK. The PCL register is mapped to PC<7:0>.
For any instruction where the PCL is the destination (excluding TBL instruction), the PC<7:0> is provided by the instruction word or ALU result, and the PC<9:8> will be cleared. The PC<11:10> will come from the PG<1:0> bits (STATUS<6:5>).
For TBL instruction, the PC<7:0> is provided by the ALU result, and the PC<9:8> are not changed. The PC<11:10> will come from the PG<1:0> bits (STATUS<6:5>).

FIGURE 2.2: Loading of PC in Different Situations

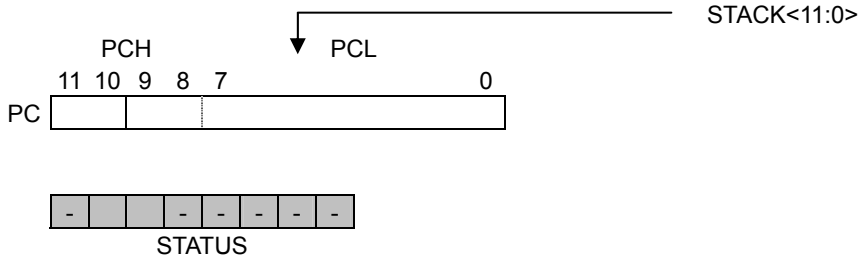
Situation 1: GOTO Instruction



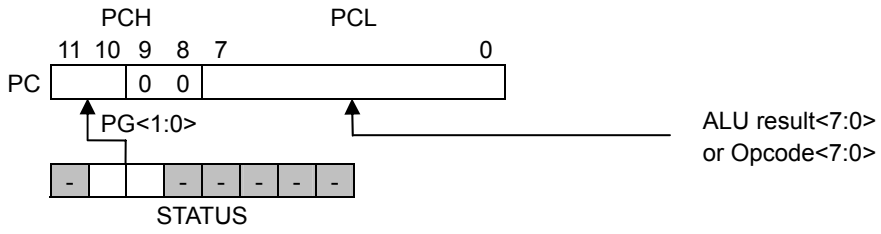
Situation 2: CALL Instruction



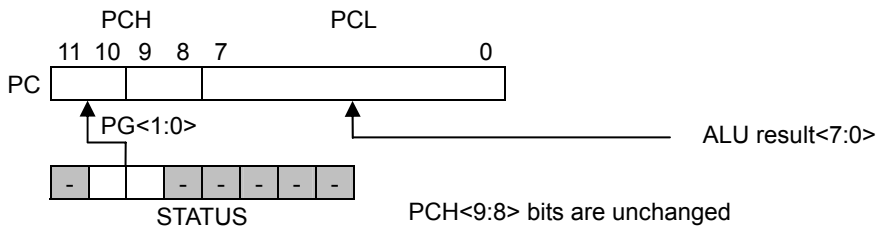
Situation 3: RETIA, RETFIE, or RETURN Instruction



Situation 4: Instruction with PCL as destination (excluding TBL instruction)



Situation 5: TBL Instruction



2.1.4 STATUS (Status Register)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
03h (r/w)	STATUS	GP	PG1	PG0	$\overline{TO}$	$\overline{PD}$	Z	DC	C

This register contains the arithmetic status of the ALU, the RESET status.
If the STATUS Register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. These bits are set or cleared according to the device logic. Furthermore, the $\overline{TO}$ and $\overline{PD}$ bits are not writable. Therefore, the result of an instruction with the STATUS Register as destination may be different than intended. For example, CLRR STATUS will clear the upper three bits and set the Z bit. This leaves the STATUS Register as 000u u1uu (where u = unchanged).

C : Carry/borrow bit.
ADDAR, ADDIA
= 1, a carry occurred.
= 0, a carry did not occur.
SUBAR, SUBIA
= 1, a borrow did not occur.
= 0, a borrow occurred.
Note : A subtraction is executed by adding the two's complement of the second operand. For rotate (RRR, RLR) instructions, this bit is loaded with either the high or low order bit of the source register.

DC : Half carry/half borrow bit.
ADDAR, ADDIA
= 1, a carry from the 4th low order bit of the result occurred.
= 0, a carry from the 4th low order bit of the result did not occur.
SUBAR, SUBIA
= 1, a borrow from the 4th low order bit of the result did not occur.
= 0, a borrow from the 4th low order bit of the result occurred.

Z : Zero bit.
= 1, the result of a logic operation is zero.
= 0, the result of a logic operation is not zero.

$\overline{PD}$: Power down flag bit.
= 1, after power-up or by the CLRWDT instruction.
= 0, by the SLEEP instruction.

$\overline{TO}$: Time overflow flag bit.
= 1, after power-up or by the CLRWDT or SLEEP instruction.
= 0, a watch-dog time overflow occurred.

PG1:PG0 : Program memory page select bits. Used for GOTO, CALL, or any instruction with PCL as destination.

PG1	PG0	Program Memory Page [Address]
0	0	Page 0 [000h~3FFh]
0	1	Page 1 [400h~7FFh]
1	0	Page 2 [800h~BFFh]
1	1	Page 3 [C00h~FFFh]

GP : General purpose read/write bit.

2.1.5 FSR (Indirect Data Memory Address Pointer)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
04h (r/w)	FSR	RP1	RP0	Indirect data memory address pointer					

Bit5:Bit0 : Select registers address in the indirect addressing mode. See 2.1.1 for detail description.

RP1:RP0 : These bits are used to switching the bank of four data memory banks. See 2.1.1 for detail description.

2.1.6 PORTA, PORTB, PORTC, PORTD & PORTE (Port Data Registers)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
05h (r/w)	PORTA	IOA7	IOA6	IOA5	IOA4	IOA3	IOA2	IOA1	IOA0
06h (r/w)	PORTB	IOB7	IOB6	IOB5	IOB4	IOB3	IOB2	IOB1	IOB0
07h (r/w)	PORTC	IOC7	IOC6	IOC5	IOC4	IOC3	IOC2	IOC1	IOC0
08h (r/w)	PORTD	IOD7	IOD6	IOD5	IOD4	IOD3	IOD2	IOD1	IOD0
09h (r/w)	PORTE	IOE7	IOE6	IOE5	IOE4	IOE3	IOE2	IOE1	IOE0

Reading the port (PORTA, PORTB, PORTC, PORTD, PORTE register) reads the status of the pins independent of the pin's input/output modes. Writing to these ports will write to the port data latch.

2.1.7 SPIRCB (SPI Receive Buffer Register) (Bank 0)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Ah (r/w)	SPIRCB	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0

RC7:RC0 : SPI receives data buffer. Once the 8-bits data have been received, the data in SPI shift register (SPISR) will be moved to the SPIRCB register.
The data must be read out before the next 8-bits data reception is completed if needed.
The RCBF flag is set when the data in SPISR is moved to the SPIRCB register, and cleared as the SPIRCB register reads.

2.1.8 SPITXB (SPI Transmit Buffer Register) (Bank 0)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Bh (r/w)	SPITXB	TX7	TX6	TX5	TX4	TX3	TX2	TX1	TX0

TX7:TX0 : SPI transmits data buffer. Once the first valid clock pulse appear on SCK pin, the data in SPITXB will be loaded into SPISR and start to shift in/out.
The new data must be written to SPITXB before the 8-bits data transmission is completed if needed.
The TXBF flag is set when the data in SPITXB is moved to the SPISR register, and cleared as the SPITXB register writes.

2.1.9 SPISTAT (SPI Status Register) (Bank 0)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Ch (r/w)	SPISTAT	-	TXBFIF	TXBF	TM1IF	SDOOD	SCKOD	RCBFIF	RCBF

RCBF : SPI receive buffer full flag. Set when the data in SPISR is moved to the SPIRCB register, reset by software or by reading SPIRCB register.
= 1, Receive complete, SPIRCB is full.
= 0, Receive not complete, SPIRCB is empty.

RCBFIF : SPI receive buffer full interrupt flag. Set when the data in SPISR is moved to the SPIRCB register, reset by software.
= 1, Receive complete, SPIRCB is full.
= 0, Receive not complete, SPIRCB is empty.

SCKOD : Open-drain control bit for SCK pin output
= 1, Open-drain enable.
= 0, Open-drain disable.

SDOOD : Open-drain control bit for SDO pin output
= 1, Open-drain enable.
= 0, Open-drain disable.

TM1IF : SPI receive complete interrupt flag in Timer 1 mode. Set when receiving complete, reset by software.
= 1, In Timer 1 mode, receiving complete.
= 0, In Timer 1 mode, receiving not complete yet.

TXBF : SPI transmit buffer empty flag. Set when the data in SPITXB is moved to the SPISR register, reset by software or by writing SPITXB register.
= 1, Transmit start, SPITXB is empty.
= 0, SPITXB is full.

TXBFIF : SPI transmit buffer empty interrupt flag. Set when the data in SPITXB is moved to the SPISR register, reset by software.
= 1, Transmit start, SPITXB is empty.
= 0, SPITXB is full.

2.1.10 SPICON (SPI Control Register) (Bank 0)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Dh (r/w)	SPICON	CKEDG	SPION	RCOV	SSE	SSEMOD	SPIM2	SPIM1	SPIM0

SPIM2:SPIM0 : SPI mode setting

SPIM2 : SPIM0	SSP Mode
0, 0, 0	SPI master mode, clock = Fosc/2
0, 0, 1	SPI master mode, clock = Fosc/4
0, 1, 0	SPI master mode, clock = Fosc/8
0, 1, 1	SPI master mode, clock = Fosc/16
1, 0, 0	SPI master mode, clock = Fosc/32
1, 0, 1	SPI slave mode, clock = SCK pin, SSB pin control enabled
1, 1, 0	SPI slave mode, clock = SCK pin, SSB pin control disabled
1, 1, 1	SPI master mode, clock = Timer1 output/2

SSEMOD : SSE bit control enable bit
= 1, Disable the SSE bit control. It means the SCK input/output directly.
= 0, Enable the SSE bit control. It means the SCK input/output will be inhibited if SSE = 0.

SSE : SPI shift register enable bit
= 1, Start to transmit/receive, and keep on “1” while the current byte is still being transmitted/received.
= 0, Reset by hardware as soon as the shifting is complete.
Note : this bit is a “don’t care” if SSEMOD = 1.

RCOV : SPI receive buffer overflow bit (only in slave mode)

= 1, A new byte is received while the SPIRCB register is still holding the previous data. In this case, the data in SPISR register will be ignored and lost.
= 0, Not overflow.

SPION : SPI module enable bit
= 1, Enable SPI module.
= 0, Disable SPI module.

CKEDG : Clock edge select bit
= 1, Data shifts out on falling edge of SCK, and shifts in on rising edge of SCK.
= 0, Data shifts in on rising edge of SCK, and shifts in on falling edge of SCK.

2.1.11 TMR1 (Timer 1 Register) (Bank 0)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Eh (r/w)	TMR1	TMR17	TMR16	TMR15	TMR14	TMR13	TMR12	TMR11	TMR10

TMR17:TMR10 : Timer 1 register and increase until the value matches to PR1 register, and then reset to “0”.

2.1.12 PR1 (Timer 1 Pulse-width Register) (Bank 0)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Fh (r/w)	PR1	PR17	PR16	PR15	PR14	PR13	PR12	PR11	PR10

PR17:PR10 : Timer 1 period register.

2.1.13 T23CON (Timer 2&3 Control Register) (Bank 1)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Ah (r/w)	T23CON	-	-	-	T16	T3ON	T3CS	T2ON	T2CS

T2CS : Timer 2 clock source selection bit
= 1, External clock input T23CKI pin.
= 0, Internal clock Fosc/4.

T2ON : Timer 2 module enable bit
= 1, Enable Timer 2 module.
= 0, Disable Timer 2 module.

T3CS : Timer 3 clock source selection bit
= 1, External clock input T23CKI pin.
= 0, Internal clock Fosc/4.

T3ON : Timer 3 module enable bit
= 1, Enable Timer 3 module.
= 0, Disable Timer 3 module.

T16 : 8-bit or 16-bit timer selection bit for Timer2 & Timer3
= 1, Timer2 & Timer3 are concatenated to form a 16-bit timer.
= 0, Timer2 & Timer3 are two 8-bit incrementing timers.

Bit7:BIT5 : Not used. Read as “0”s

2.1.14 TMR2 (Timer 2 Register) (Bank 1)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Bh (r/w)	TMR2	TMR27	TMR26	TMR25	TMR24	TMR23	TMR22	TMR21	TMR20

TMR27:TMR20 : Timer 2 register and increase until the value matches to PR2 register, and then reset to “0”.

2.1.15 PR2 (Timer 2 Pulse-width Register) (Bank 1)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Ch (r/w)	PR2	PR27	PR26	PR25	PR24	PR23	PR22	PR21	PR20

PR27:PR20 : Timer 2 period register.

2.1.16 TMR3 (Timer 3 Register) (Bank 1)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Dh (r/w)	TMR3	TMR37	TMR36	TMR35	TMR34	TMR33	TMR32	TMR31	TMR30

TMR37:TMR30 : Timer 3 register and increase until the value matches to PR3 register, and then reset to “0”.

2.1.17 PR3 (Timer 3 Pulse-width Register) (Bank 1)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Eh (r/w)	PR3	PR37	PR36	PR35	PR34	PR33	PR32	PR31	PR30

PR37:PR30 : Timer 3 period register.

2.1.18 PWMCON (PWM Control Register) (Bank 2)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Ah (r/w)	PWMCON	-	-	-	-	-	PW1T3	PW1ON	PW0ON

PW0ON : PWM0 module enable bit
= 1, Enable PWM0 output, IOA0/PWM0 pin is configured to PWM0 pin.
= 0, Disable PWM0 output, IOA0/PWM0 pin is configured to IOA0 pin.

PW1ON : PWM1 module enable bit
= 1, Enable PWM1 output, IOC0/PWM1 pin is configured to PWM1 pin.
= 0, Disable PWM1 output, IOC0/PWM1 pin is configured to IOC0 pin.

PW1T3 : PWM1 time base selection bit
= 1, the PWM1 time base is determined by TMR3 and PR3.
= 0, the PWM1 time base is determined by TMR2 and PR2.

Bit7:bit3 : Not used. Read as “0”s.

2.1.19 PW0DCL (PWM0 Duty Cycle Register Low Byte) (Bank 2)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Bh (r/w)	PW0DCL	DC1	DC0	-	-	-	-	-	-

Bit5:bit0 : Not used. Read as “0”s.

DC1:DC0 : The low bits of PWM0 duty cycle.

2.1.20 PW0DCH (PWM0 Duty Cycle Register High Byte) (Bank 2)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Ch (r/w)	PW0DCH	DC9	DC8	DC7	DC6	DC5	DC4	DC3	DC2

DC9:DC2 : The high byte of PWM0 duty cycle.

2.1.21 PW1DCL (PWM1 Duty Cycle Register Low Byte) (Bank 2)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Dh (r/w)	PW1DCL	DC1	DC0	-	-	-	-	-	-

Bit5:bit0 : Not used. Read as “0”s.

DC1:DC0 : The low bits of PWM1 duty cycle.

2.1.22 PW1DCH (PWM1 Duty Cycle Register High Byte) (Bank 2)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Eh (r/w)	PW1DCH	DC9	DC8	DC7	DC6	DC5	DC4	DC3	DC2

DC9:DC2 : The high byte of PWM1 duty cycle.

2.1.23 RFCCON (RFC Control Register) (Bank 3)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Ah (r/w)	RFCCON	RFCON	START	-	RFCS4	RFCS3	RFCS2	RFCS1	RFCS0

RFCS4:RFCS0 : Select one the RFC oscillation network of RFCx (x = 0 to 18). The selected IOxx/RFCx pin will be configured as RFCx output pin if RFCON = 1. Other IOxx/RFCx pins will still behave as port pins. If RFCON = 0, all IOxx/RFCx pins will still behave as port pins.
= 0,0,0,0,0, IOA2/RFC0 pin is configured to RFC0 pin.
= 0,0,0,0,1, IOA3/RFC1 pin is configured to RFC1 pin.
= 0,0,0,1,0, IOA4/RFC2 pin is configured to RFC2 pin.
= 0,0,0,1,1, IOA5/RFC3 pin is configured to RFC3 pin.
= 0,0,1,0,0, IOA6/RFC4 pin is configured to RFC4 pin.
= 0,0,1,0,1, IOA7/RFC5 pin is configured to RFC5 pin.
= 0,0,1,1,0, IOD0/RFC6 pin is configured to RFC6 pin.
= 0,0,1,1,1, IOD1/RFC7 pin is configured to RFC7 pin.
= 0,1,0,0,0, IOB1/RFC8 pin is configured to RFC8 pin.
= 0,1,0,0,1, IOB2/RFC9 pin is configured to RFC9 pin.
= 0,1,0,1,0, IOB3/RFC10 pin is configured to RFC10 pin.
= 0,1,0,1,1, IOB4/RFC11 pin is configured to RFC11 pin.

= 0,1,1,0,0, IOB5/RFC12 pin is configured to RFC12 pin.
= 0,1,1,0,1, IOB6/RFC13 pin is configured to RFC13 pin.
= 0,1,1,1,0, IOB7/RFC14 pin is configured to RFC14 pin.
= 0,1,1,1,1, IOC2/RFC15 pin is configured to RFC15 pin.
= 1,0,0,0,0, IOC1/RFC16 pin is configured to RFC16 pin.
= 1,0,0,0,1, DATA/RFC17 pin is configured to RFC17 pin.
= 1,0,0,1,0, CLK/RFC18 pin is configured to RFC18 pin.

Bit5 : Not used. Read as "0".

START : RFC counter enable bit
= 1, RFC counter start to convert.
= 0, Stop the RFC conversion, reset by hardware when conversion is finished or by software.

RFCON : RFC module enable bit
= 1, Enable RFC module, the selected IOxx/RFCx pin is configured to RFCx pin, and the IOA1/CX pin is configured to CX pin.
= 0, Disable RFC module, all the IOxx/RFCx pins are configured to IOXX pin, and the IOA1/CX pin is configured to IOA1 pin.

2.1.24 RFCDL (RFC Data Register Low Byte) (Bank 3)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Bh (r)	RFCDL	RFCD7	RFCD6	RFCD5	RFCD4	RFCD3	RFCD2	RFCD1	RFCD0

RFCD7:RFCD0 : The low byte of RFC conversion result.

2.1.25 RFCDH (RFC Data Register High Byte) (Bank 3)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Ch (r)	RFCDH	RFCOV	RFCD14	RFCD13	RFCD12	RFCD11	RFCD10	RFCD9	RFCD8

RFCD14:RFCD8 : The high byte of RFC conversion result.

RFCOV : RFC counter overflow flag. Set when RFC counter overflow, reset by RFC counter reset.
= 1, Overflow.
= 0, Not overflow.

2.1.26 CMPDX (Data X Register Data Comparator) (Bank 3)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Dh (r/w)	CMPDX	DX7	DX6	DX5	DX4	DX3	DX2	DX1	DX0

DX7:DX0 : The data X of data comparator, this data will be compared to CMPDY.

2.1.27 CMPDY (Data Y Register for Data Comparator) (Bank 3)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Eh (r/w)	CMPDY	DY7	DY6	DY5	DY4	DY3	DY2	DY1	DY0

DY7:DY0 : The data Y of data comparator, this data will be compared to CMPDX.

2.1.28 CMPDY (Data Comparator Status Register) (Bank 3)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Fh (r)	CMPSTAT	-	-	-	-	CMPF3	CMPF2	CMPF1	CMPF0

CMPF4:CMPF0 : the error number of the compared result (0 ~ 8) of data comparator

Bit7:bit4 : Not used. Read as “0”s.

2.1.29 INTFLAG (Interrupt Status Register)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
3Fh (r/w)	INTFLAG	SPITXIF	RFCIF	T3IF	T2IF	T1IF	SPIRCIF	INTIF	T0IF

T0IF : Timer0 overflow interrupt flag. Set when Timer0 overflows, reset by software.

INTIF : External INT pin interrupt flag. Set by falling edge on INT pin, reset by software.

SPIRCIF : SPI receive module interrupt flag. Set when SPI receiver buffer is full (SPI data transmission complete), reset by software.

T1IF : Timer1 match interrupt flag. Set when TMR1 register matches to PR1 register, reset by software.

T2IF : Timer2 match interrupt flag. Set when TMR2 register matches to PR2 register, reset by software.

T3IF : Timer3 match interrupt flag. Set when TMR3 register matches to PR3 register, reset by software.

RFCIF : RFC module interrupt flag. Set when RFC conversion is completed, reset by software.

SPITXIF : SPI transmit module interrupt flag. Set when SPI transmit buffer is empty (SPI data transmission start), reset by software.

2.1.30 ACC (Accumulator)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
N/A (r/w)	ACC	Accumulator							

Accumulator is an internal data transfer, or instruction operand holding. It can not be addressed.

2.1.31 OPTION Register

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
N/A (r/w)	OPTION	/PHBCE	GIE			PSA	PS2	PS1	PS0

Accessed by OPTION/OPTIONR instructions.

By executing the OPTION instruction, the contents of the ACC Register will be transferred to the OPTION Register.
By executing the OPTIONR instruction, user can read this register into ACC.
The OPTION Register contains various control bits to configure the Timer0/WDT prescaler, Timer0, pull-high, and interrupt.
The OPTION Register are set all “1”s except GIE bit after any reset.

PS2:PS0 : Prescaler rate select bits.

PS2:PS0	Timer0 Rate	WDT Rate
0 0 0	1:2	1:1
0 0 1	1:4	1:2
0 1 0	1:8	1:4
0 1 1	1:16	1:8
1 0 0	1:32	1:16
1 0 1	1:64	1:32
1 1 0	1:128	1:64
1 1 1	1:256	1:128

PSA : Prescaler assign bit.
= 1, WDT (watch-dog timer).
= 0, TMR0 (Timer0).

GIE : Global interrupt enable bit.
= 0, Disable all interrupts. For wake-up from SLEEP mode through an interrupt event, the device will continue execution at the instruction after the SLEEP instruction.
= 1, Enable all un-masked interrupts. For wake-up from SLEEP mode through an interrupt event, the device will branch to the interrupt address (001h).
Note : 1. The GIE bit is not writable bit. This bit is only set by ENI or RETFIE instructions, and cleared by DISI instruction or entering into interrupt subroutine.
2. When an interrupt event occur with the GIE bit and its corresponding interrupt enable bit are all set, the GIE bit will be cleared by hardware to disable any further interrupts. The RETFIE instruction will exit the interrupt routine and set the GIE bit to re-enable interrupt.

/PHBCE : = 0, Enable the internal pull-high of IOB0~ IOB7, IOC4~IOC5, and IOE0~IOE5 pins.
= 1, Disable the internal pull-high of IOB0~ IOB7, IOC4~IOC5, and IOE0~IOE5 pins.
Note : /PHB, /PHE are “AND” gating with /PHBCE, that is each one written “0” will enable pull-high.

2.1.32 IOSTA, IOSTB, IOSTC, IOSTD & IOSTE (Port I/O Control Registers)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
05h (r/w)	IOSTA	Port A I/O Control Register							
06h (r/w)	IOSTB	Port B I/O Control Register							
07h (r/w)	IOSTC	Port C I/O Control Register							
08h (r/w)	IOSTD	Port D I/O Control Register							
09h (r/w)	IOSTE	Port E I/O Control Register							

Accessed by IOST/IOSTR instructions.

The Port I/O Control Registers are loaded with the contents of the ACC Register by executing the IOST R (05h~09h) instruction. By executing the IOSTR instruction, user can read these registers into ACC.
A ‘1’ from a IOST Register bit puts the corresponding output driver in hi-impedance state (input mode). A ‘0’ enables the output buffer and puts the contents of the output data latch on the selected pins (output mode).
The IOST Registers are set all “1”s (output drivers disabled) upon RESET.

2.1.33 T1CON (Timer 1 Control Register)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Ch (r/w)	T1CON						T1ON	T1P1	T1P0

Accessed by IOST/IOSTR instructions.

T1P1:T1P0 : Timer 1 prescaler bits.

T1P1 : T1P0	Prescaler Rate
0, 0	1 : 1
0, 1	1 : 4
1, 0	1 : 8
1, 1	1 : 16

T1ON : Timer 1 module enable bit
= 1, Enable Timer 1 module.
= 0, Disable Timer 1 module.

2.1.34 PHCON (Pull-high Control Register)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Dh (r/w)	PHCON	HDC				/PHE	/PHD	/PHB	/PHA

Accessed by IOST/IOSTR instructions.

/PHA : = 0, Enable the internal pull-high of IOA0~ IOA7 pins.
= 1, Disable the internal pull-high of IOA0~ IOA7 pins.

/PHB : = 0, Enable the internal pull-high of IOB0~ IOB7 pins.
= 1, Disable the internal pull-high of IOB0~ IOB7 pins.
Note : /PHB is “AND” gating with /PHBCE, that is each one written “0” will enable pull-high.

/PHD : = 0, Enable the internal pull-high of IOD0~ IOD7 pins.
= 1, Disable the internal pull-high of IOD0~ IOD7 pins.

/PHE : = 0, Enable the internal pull-high of IOE0~ IOE5 pins.
= 1, Disable the internal pull-high of IOE0~ IOE5 pins.
Note : /PHE is “AND” gating with /PHBCE, that is each one written “0” will enable pull-high.

HDC : Driving ability enable bit of IOC0~IOC2 pins.
= 0, Normal driving ability of IOC0~IOC2 pins.
= 1, Reduce the driving ability of IOC0~IOC2 pins.

2.1.35 PCON (Power Control Register)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Eh (r/w)	PCON	LVDTE	ODE	WDTE	-	ROC	-	-	/WUE

Accessed by IOST/IOSTR instructions.

/WUE : Input change wake-up function of IOB0~IOB7, IOC4~IOC5, and IOE0~IOE1 pins enable bit.
= 0, Enable the input change wake-up function.
= 1, Disable the input change wake-up function.

Bit2:Bit1 : Not used. Read as “0”s.

ROC : R-option function of IOD0 and IOD1 pins enable bit.
= 0, Disable the R-option function.
= 1, Enable the R-option function. In this case, if a 430KΩ external resistor is connected/disconnected to Vss, the status of IOD0 (IOD1) is read as “0”/”1”.

Bit4 : Not used. Read as “1”.

WDTE : WDT (watch-dog timer) enable bit.
= 0, Disable WDT.
= 1, Enable WDT.

ODE : Open-drain function of IOC6 and IOC7 pins enable bit.
= 0, Disable the open-drain function for both IOC6 and IOC7 pins.
= 1, Enable the open-drain function for both IOC6 and IOC7 pins.

LVDTE : LVDT (low voltage detector) enable bit.
= 0, Disable LVDT.
= 1, Enable LVDT.

2.1.36 INTEN (Interrupt Mask Register)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
0Fh (r/w)	INTEN	SPITXIE	RFCIE	T3IE	T2IE	T1IE	SPIRCIE	INTIE	T0IE

Accessed by IOST/IOSTR instructions.

T0IE : Timer0 overflow interrupt enable bit.
= 0, Disable the Timer0 overflow interrupt.
= 1, Enable the Timer0 overflow interrupt.

INTIE : External INT pin interrupt enable bit.
= 0, Disable the External INT pin interrupt.
= 1, Enable the External INT pin interrupt.

SPIRCIE : SPI receive module interrupt enable bit.
= 0, Disable the SPI receive module interrupt.
= 1, Enable the SPI receive module interrupt.

T1IE : Timer1 match interrupt enable bit.
= 0, Disable the Timer1 match interrupt.
= 1, Enable the Timer1 match interrupt.

T2IE : Timer2 match interrupt enable bit.
= 0, Disable the Timer2 match interrupt.
= 1, Enable the Timer2 match interrupt.

T3IE : Timer3 match interrupt enable bit.
= 0, Disable the Timer3 match interrupt.
= 1, Enable the Timer3 match interrupt.

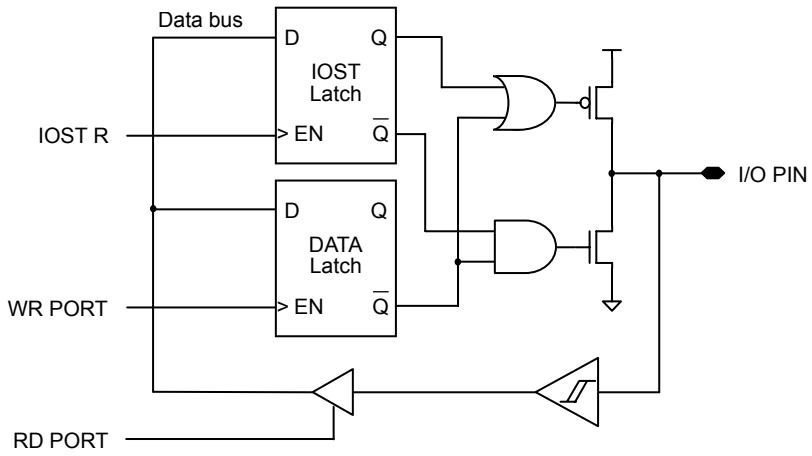
RFCIE : RFC module interrupt enable bit.
= 0, Disable the RFC module interrupt.
= 1, Enable the RFC module interrupt.

SPITXIE : SPI transmit module interrupt enable bit.
= 0, Disable the SPI transmit module interrupt.
= 1, Enable the SPI transmit module interrupt.

2.2 I/O Ports

Port A, port B, port C, port D and port E are bi-directional tri-state I/O ports.
All I/O pins (IOA, IOB, IOC, IOD and IOE) have data direction control registers (IOSTA, IOSTB, IOSTC, IOSTD, and IOSTE) which can configure these pins as output or input.
IOA<7:0>, IOB<7:0>, IOC<5:4>, IOD<7:0> and IOE<5:0> have corresponding pull-high control bits (/PHBCE, /PHA, /PHB, /PHD, and /PHE bits) to enable the weak internal pull-high. The weak pull-high is automatically turned off when the pin is configured as an output pin.
IOC6 and IOC7 have an open-drain control bit (ODE, PCON<6>) to enable the open-drain output when these pins are configured to be an output pin.
IOD0 and IOD1 are the R-option pins enabled by setting the ROC bit (PCON<3>). When the R-option function is used, it is recommended that IOD0 and IOD1 are used as output pins, and read the status of IOD0 and IOD1 before these pins are configured to be an output pin.
IOB0~IOB7, IOC4~IOC5, and IOE0~IOE1 also provide the input change interrupt/wake-up function which are enabled by clear /WUE bit (PCON<0>). The input change interrupt/wake-up function is automatically turned off when the pin is configured as an output pin.

FIGURE 2.3: Block Diagram of I/O PINs



Pull-high/R-option is not shown in the figure

2.3 Timer0/WDT & Prescler

2.3.1 Timer0

The Timer0 is a 8-bit timer/counter. The clock source of Timer0 comes from the internal clock.
The timer0 register (TMR0) will increment every instruction cycle (without prescaler). If TMR0 register is written, the increment is inhibited for the following two cycles.

2.3.2 Watchdog Timer (WDT)

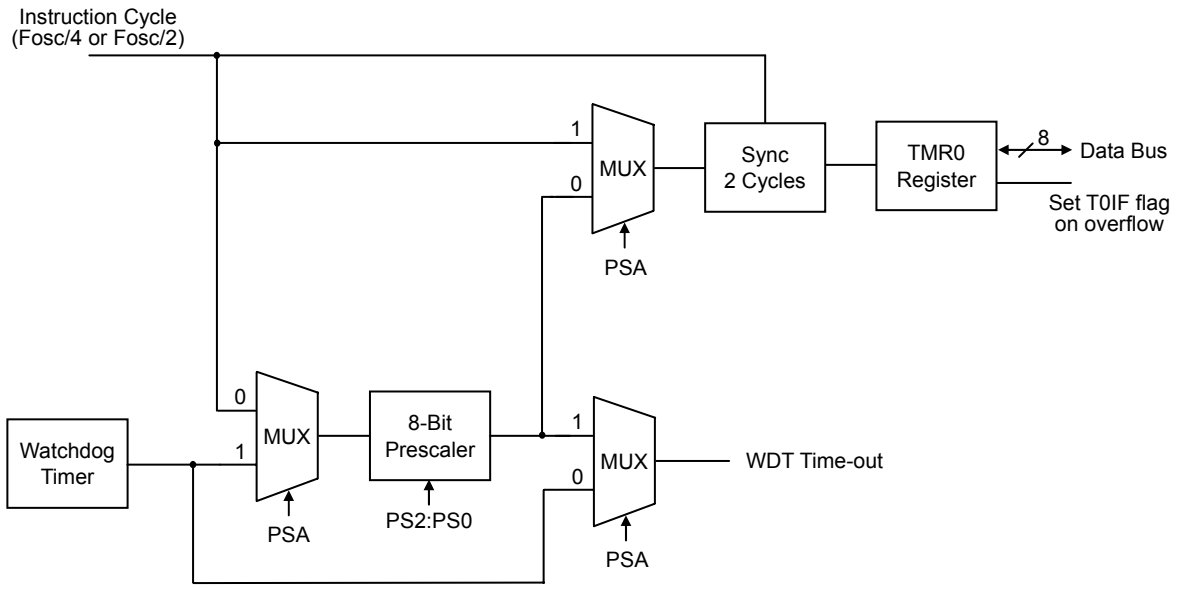
The Watchdog Timer (WDT) is a free running on-chip RC oscillator which does not require any external components. So the WDT will still run even if the clock on the OSCI and OSCO pins is turned off, such as in SLEEP mode. During normal operation or in SLEEP mode, a WDT time-out will cause the device reset and the $\overline{TO}$ bit (STATUS<4>) will be cleared.
The WDT can be disabled by clearing the control bit WDTE (PCON<5>) to “0”.
The WDT has a nominal time-out period of 18 ms (without prescaler). If a longer time-out period is desired, a

prescaler with a division ratio of up to 1:128 can be assigned to the WDT controlled by the OPTION register. Thus, the longest time-out period is approxmately 2.3 seconds if reset delay time is set to 18ms.
The CLRWDT instruction clears the WDT and the prescaler, if assigned to the WDT, and prevents it from timing out and generating a device reset.
The SLEEP instruction resets the WDT and the prescaler, if assigned to the WDT. This gives the maximum SLEEP time before a WDT Wake-up Reset.

2.3.3 Prescaler

An 8-bit counter (down counter) is available as a prescaler for the Timer0, or as a postscaler for the Watchdog Timer (WDT). Note that the prescaler may be used by either the Timer0 module or the WDT, but not both. Thus, a prescaler assignment for the Timer0 means that there is no prescaler for the WDT, and vice-versa.
The PSA bit (OPTION<3>) determines prescaler assignment. The PS<2:0> bits (OPTION<2:0>) determine prescaler ratio.
When the prescaler is assigned to the Timer0 module, all instructions writing to the TMR0 register will clear the prescaler. When it is assigned to WDT, a CLRWDT instruction will clear the prescaler along with the WDT.
The prescaler is neither readable nor writable. On a RESET, the prescaler contains all '1's.
To avoid an unintended device reset, CLRWDT or CLRR TMR0 instructions must be executed when changing the prescaler assignment from Timer0 to the WDT, and vice-versa.

FIGURE 2.4: Block Diagram of The Timer0/WDT Prescaler



2.4 Timer1

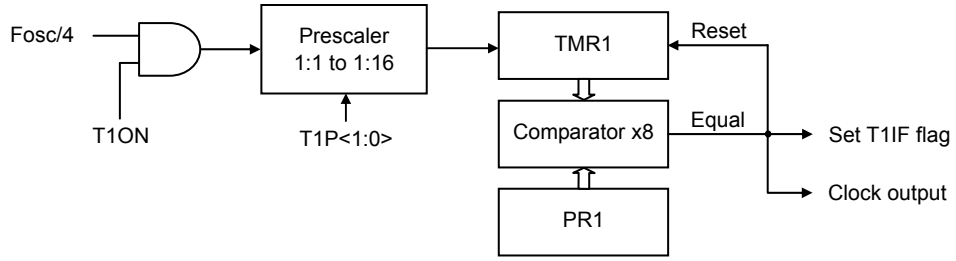
The Timer1 is a 8-bit clock counter with a programmable prescaler and a 8-bit period register (PR1). It also can be as a baud rate clock generator for the SPI module. The clock source of Timer1 comes from the internal clock (Fosc/4). The option of Timer1 prescaler (1:1, 1:4, 1:8, 1:16) is defined by T1P1:T1P0 (T1CON<1:0>) bits. **The prescaler is cleared when a value is written to TMR1 or T1CON register, and during any kind of reset as well.**
The timer increments from 00h until it equals the period register (PR1). It then resets to 00h at the next increment cycle. The timer interrupt flag (T1IF) is set when the timer rollover to 00h.
The timer also has a corresponding interrupt enable bit (T1IE). The timer interrupt can be enabled/disabled by setting/clearing this bit.
The timer s can be turned on and off under software control. When the timer on control bit (T1ON, T1CON<2>) is set, the timer increments from the clock source. When T1ON is cleared, the timer is turned off and cannot cause the

timer interrupt flag to be set.

TABLE 2.1: Timer 1 Prescaler Rate

T1P1 : T1P0	Prescaler Rate
0, 0	1 : 1
0, 1	1 : 4
1, 0	1 : 8
1, 1	1 : 16

FIGURE 2.5: Block Diagram of The Timer1



2.5 Timer2 & Timer3

Timer2 and Timer3 are two 8-bit incrementing timers, each with an 8-bit period register (PR2 and PR3, respectively) and separate overflow interrupt flags. Timer2 and Timer3 can operate either as timers (increment on internal clock, Fosc/4), or as counters (increment on falling edge of external clock on pin IOB0/T23CKI). They are also software configurable to operate as a single 16-bit timer/counter. These timers are also used as the time base for the PWM (Pulse Width Modulation) modules.

2.5.1 Timer2 & Timer3 in 8-Bit Mode

Both Timer2 and Timer3 will operate in 8-bit mode when the T16 (T23CON<4>) bit is clear. These two timers can be independently configured to increment from the internal clock (Fosc/4), or from an external clock source on IOB0/T23CKI pin. The timer clock source in configured by the TxCS bit (x = 2 for Timer2 (T23CON<0>), x = 3 for Timer3 (T23CON<2>)). When TxCS is set, the clock source is the IOB0/T23CKI pin and the counters will increment on every falling edge of the IOB0/T23CKI pin.
The timer increments from 00h until it equals the period register (PRx, x = 2 for Timer2, x = 3 for Timer3). It then resets to 00h at the next increment cycle. The timer interrupt flag is set when the timer rollover to 00h. Timer2 and Timer3 have individual interrupt flag bits (T2IF and T3IF).
Each timer also has a corresponding interrupt enable bit (T2IE and T3IE). The timer interrupt can be enabled/disabled by setting/clearing this bit.
The timer s can be turned on and off under software control. When the timer on control bit (TxON, T23CON<1> and T23CON<3>) is set, the timer increments from the clock source. When TxON is cleared, the timer is turned off and cannot cause the timer interrupt flag to be set.
When TxCS is set, the clock source is the IOB0/T23CKI pin, and the counter will increment on every falling edge on the IOB0/T23CKI pin. The T23CKI input is synchronized with internal phase clocks. This cause a delay from the time a falling edge appears on T23CKI to the time TMR2 or TMR3 is actually incremented.

2.5.2 Timer2 & Timer3 in 16-bit Mode

To select 16-bit mode, set the T16 bit. In this mode, TMR2 and TMR3 are concatenated to from a 16-bit timer (TMR3:TMR2). The 16-bit timer increments until it matches the 16-bit period register (PR3:PR2). On the following timer clock, the timer value is reset to 0000h, and the T2IF bit is set.
When selecting the clock source for the 16-bit timer, the T2CS bit control the entire 16-bit timer and T3CS is a “don’t

care". When T2CS is clear, the timer increments from the internal clock (Fosc/4). When T2CS is set, the timer increments on every falling edge of the IOB0/T23CKI pin.
For the 16-bit timer to increment, both T2ON and T3ON bits must be set.

TABLE 2.2: Turning on 16-Bit Timer

T16	T3ON	T2ON	Result
1	1	1	16-bit timer (TMR3:TMR2) ON
1	0	1	16-bit timer ON, only TMR2 increments
1	x	0	16-bit timer OFF
0	1	1	Timers in 8-bit mode

When T2CS is set, the 16-bit TMR3:TMR2 increments on the falling edge of clock input T23CKI. The T23CKI input is synchronized with internal phase clocks. This cause a delay from the time a falling edge appears on T23CKI to the time TMR3:TMR2 is actually incremented.

FIGURE 2.6: Timer2 and Timer3 in Two 8-bit Timer/Counter Mode

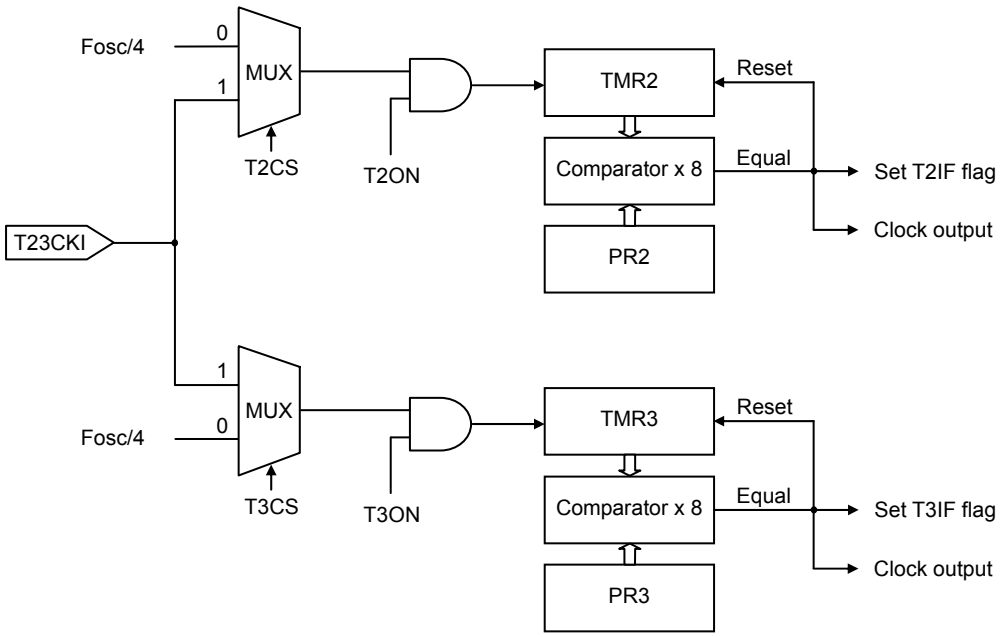
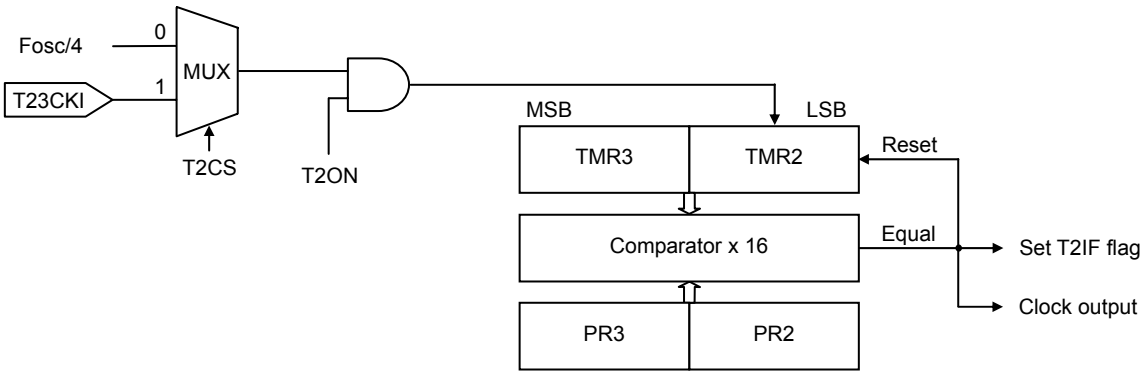


FIGURE 2.7: Timer2 and Timer3 in 16-bit Timer/Counter Mode



2.6 PWM (Pulse Width Modulation) Module

Two high speed pulse width modulation (PWM) outputs are provided. The PWM0 output uses Timer2 as its time base, the PWM1 may be configured to use Timer2 or Timer3 as the time base. The PWM outputs are on the IOA0/PWM0, and IOC0/PWM1 pins.
Each PWM output has a maximum resolution of 10-bits. The duty cycle of the output can vary from 0% to 100%. The user needs to set the PW0ON bit (PWMCON<0>) to enable the PWM output. When PW0ON bit is set, the IOA0/PWM0 pin is configured as PWM0 output and forced as an output, irrespective of the data direct bit (IOSTA<0>). When the PW0ON is clear, the pin behaves as a port pin.
Similarly, the PW1ON bit (PWMCON<1>) controls the configuration of the IOC0/PWM1 pin.

2.6.1 PWM Periods

The period of PWM0 output is determined by Timer2 and its period register (PR2). The period of the PWM1 output can be software configured to use either Timer2 or Timer3 as the time base. For PWM1, when PW1T3 bit (PWMCON<2>) is clear, the time base is determined by TMR2 and PR2, and when PW1T3 bit is set, the time base is determined by TMR3 and PR3.
Running two different PWM outputs on two different timers allows different periods. Running all PWMs from Timer2 allows the best use of resources by freeing Timer3 to operate as an 8-bit timer. Timer2 and Timer3 cannot be used as a 16-bit timer if any PWM is being used.

The PWM periods can be calculated as follows:

Period of PWM0 = [(PR2) + 1] x 4Tosc

Period of PWM1 = [(PR2) + 1] x 4Tosc or
[(PR3) + 1] x 4Tosc

The duty cycle of PWMx is determined by the 10-bit value DCx<9:0>. The upper 8-bits are from register PWxDCH and lower2-bits are from PWxDCL<7:6> (PWxDCH:PWxDCL<7:6>).

The PWMx duty cycle is as follows:

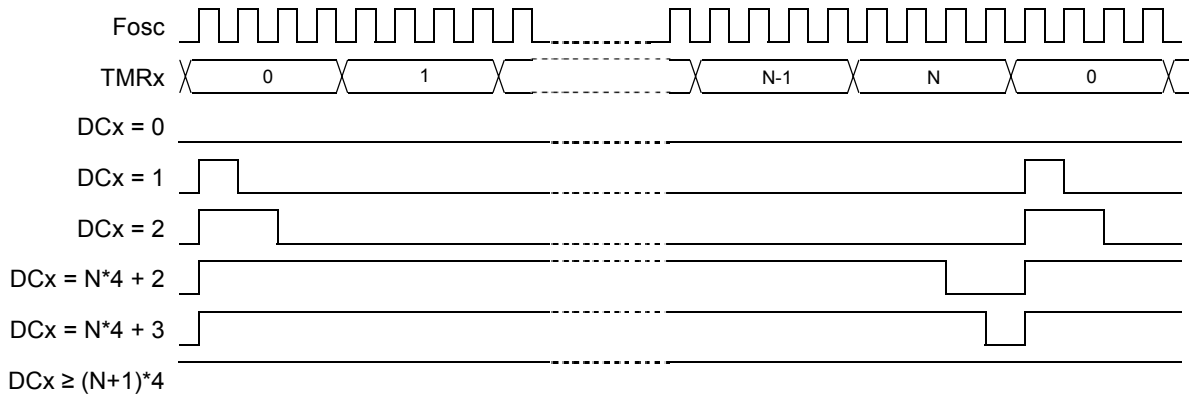
PWMx Duty Cycle = (DCx) x Tosc
Where DCx represents the 10-bit value from PWxDCH:PWxDCL.

If DCx = 0, then the duty cycle is zero. If PRx = PWxDCH, then the PWM output will be low for one to four Tosc (depending on the state of the PWxDCL<7:6> bits). For a duty cycle to be 100%, the PWxDCH value must be greater then the PRx value.
The duty cycle registers for both PWM outputs are double buffered. When the user writes to these registers, they are stored in master latches. When TMR2 (or TMR3) overflows and a new PWM period beings, the master latch values are transferred to the slave latches.

2.6.2 PWM Interrupts

The PWM modules make use of the TMR2 and/or TMR3 interrupts. A timer interrupt is generated when TMR2 or TMR3 equals its period register and on the following increment is cleared to zero. This interrupt also marks the beginning of a PWM cycle. The user can write new duty cycle values before the timer rollover. The TMR2 interrupt is latched into the T2IF bit and the TMR3 interrupt is latched into the T3IF bit. These flags must be cleared in software.

FIGURE 2.8: The Output Waveform of PWM Driver (PRx = N)



2.7 SPI(Serial Peripheral Interface) Module

The Serial Port Interface (SPI) Module is a serial interface useful communicating with other peripheral or microcontroller device.
The SPI mode allows 8-bit of data to be synchronously transmitted and received simultaneously. To accomplish communication, typically three pins are used:

1. Serial Clock (SCK)
2. Serial Data In (SDI)
3. Serial Data Output (SDO)

Additionally a fourth pin may be used when in a slave mode of operation:

4. Slave Select (SSB)

The SPI consists of a transmit/receive shift register (SPISR), a receive buffer register (SPIRCB), and a transmit buffer register (SPITXB). The SPISR shifts the data in and out of the device, MSB first. Once the first valid clock pulse appear on SCK pin (controlled by SSE (SPICON<4>) and SSEMOD (SPICON<3>) bits), data in SPITXB will be loaded into SPISR and start to shift in/out, then transmit buffer empty detect bit TXBF (SPISTAT<5>), and interrupt flag bits SPITXIF (INTFLAG<7>) and TXBFIF (SPISTAT<6>) are set. Once the 8-bits of data have been received, the data in SPISR will be moved to the SPIRCB register, then receive buffer full detect bit RCBF (SPISTAT<0>), and interrupt flag bits SPIRCIF (INTFLAG<2>) and RCBFIF (SPISTAT<1>) are set.
If FM8P51 is a master controller, it sends clock through the SCK pin. A couple of 8-bit data are transmitted and received at the same time. And if FM8P51 is defined as a slave, its SCK pin could be programmed as an input pin. Data will continue to be shifted based on both the clock rate and the selected edge.
When the application S/W is expecting to transmit valid data, the SPITXB should be write before the last byte of data have been transmitted completely. Buffer empty bit TXBF indicates when SPISR has been loaded with the data of SPITXB (reception/transmission start). The TXBF bit is cleared by software or by writting SPITXB register. And the TXBF bit may be ignored if the SPI is only a receiver.
Also when the application S/W is expecting to receive valid data, the SPIRCB should be read before the next byte of data have been received completely. Buffer full bit RCBF indicates when SPIRCB has been loaded with the received data (reception/transmission is complete). The RCBF bit is cleared by software or by reading SPIRCB register. And the RCBF bit may be ignored if the SPI is only a transmitter.
Generally the SPI interrupt is used to determine when the transmission/reception has started/completed, the SPIRCB/SPITXB must be read and/or written. If the interrupt method is not going to be used, then S/W polling RCBF and/or TXBF bits is needed.
If the SPI is only going to receive, the TXBF flag could be ignored.

FIGURE 2.9: SPI Block Diagram

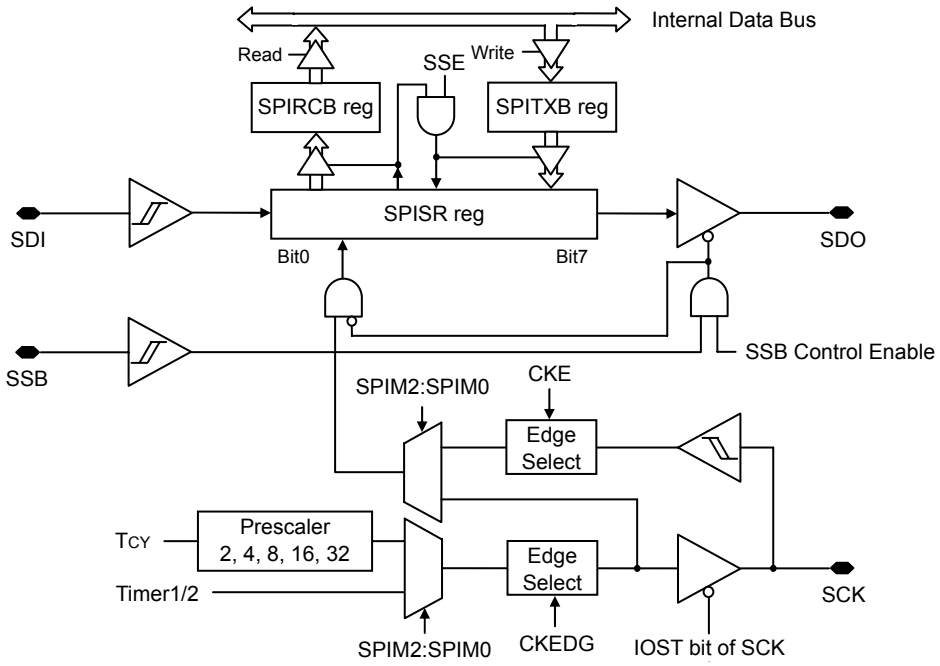


TABLE 2.3: SPI Mode Setting

SPIM2 : SPIM0	SPI Mode
0,0,0	SPI master mode, clock = Fosc/2
0,0,1	SPI master mode, clock = Fosc/4
0,1,0	SPI master mode, clock = Fosc/8
0,1,1	SPI master mode, clock = Fosc/16
1,0,0	SPI master mode, clock = Fosc/32
1,0,1	SPI slave mode, clock = SCK pin, SSB pin control enabled
1,1,0	SPI slave mode, clock = SCK pin, SSB pin control disabled
1,1,1	SPI master mode, clock = Timer1 output/2

TABLE 2.4: The Description of SPI SCK Control Bit

CKEDG	= 1, Data shifts out on falling edge of SCK, and shifts in on rising edge of SCK = 0, Data shifts in on rising edge of SCK, and shifts in on falling edge of SCK

2.7.1 Master Mode

2.7.1.1 Master Mode with SSE Control (SSEMOD = 0)

In this master mode, the data is transmitted/received as soon as the SPI shift register enable bit SSE (SPICON<4>) bit is setting to “1” by S/W. The data in SPITXB will be loaded into SPISR at the same time and start to shift in/out, then transmit buffer empty detect bit (TXBF), and interrupt flag bits (SPITXIF, TXBFIF) are set. And then user could write the next byte data to SPITXB register before the 8-bit data transmission is completed if needed. The SSE bit will be kept in “1” if the communication is still undergoing. And the SSE bit will be cleared by hardware while the shifting is completed. Once the 8-bits of data have been received, the data in SPISR will be moved to the SPIRCB register, then buffer full detect bit (RCBF), interrupt flag bits (SPIRCIF, RCBFIF) are set. And then user could read out the SPIRCB register before next 8-bit data transmission is completed if needed.

How to transmit/receive data in this master mode:

1. Enable SPI function by setting the SPION (SPICON<6>) bit.
2. Decide the transmission rate and source by programming SPIM2:SPIM0 (SPICON<2:0>) bits.
3. Write the data that you want to transmit to SPITXB register if needed.
4. Set SSE (SPICON<4>) bit to start transmit.
5. When the 8-bit data transmission starts, both of the SPITXIF and TXBFIF interrupt flags will set to 1. Besides, both of these bits are cleared by software. The TXBF flag also will be set to “1”, cleared by software or by writting data to SPITXB register.
6. Write next byte data to SPITXB register before this byte transmission being finished if needed.
7. When the 8-bit data transmission is over, the SSE bit will be reset to “0” by hardware. Therefore, if user want to transmit/receive another 8-bit data, user must set SSE bit to “1” again.
8. When the 8-bit data transmission is completed, both of the SPIRCIF and RCBFIF interrupt flags will set to 1. Besides, both of these bits are cleared by software. The RCBF flag also will be set to “1”, cleared by software or by reading out SPIRCB register.
9. Read out the SPIRCB register before next byte transmission being finished if needed.

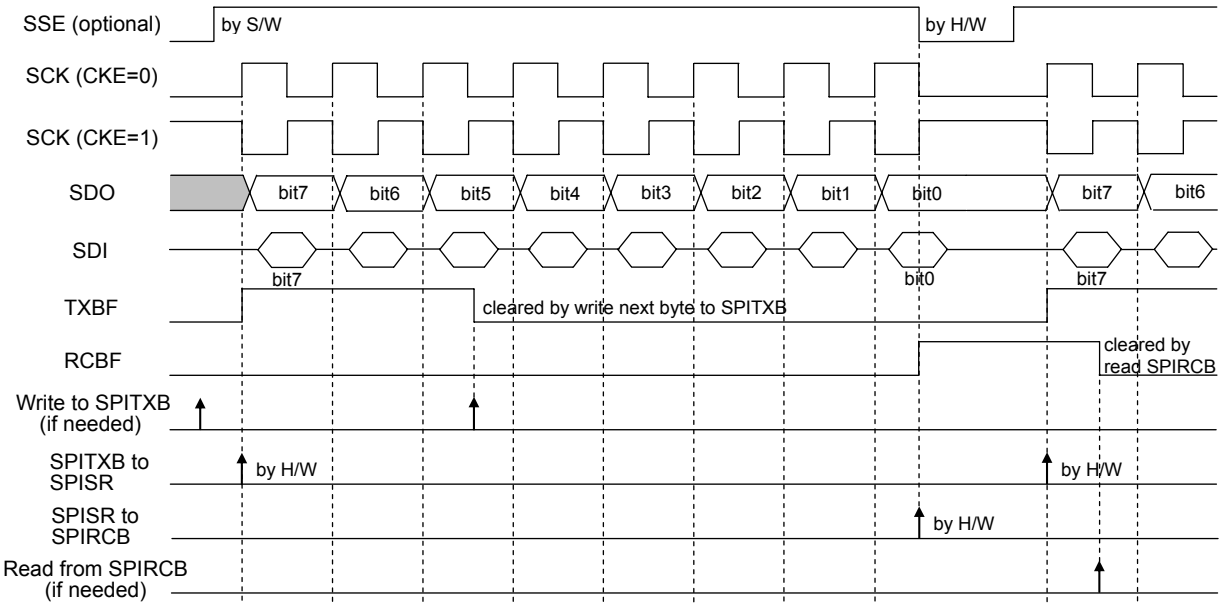
2.7.1.2 Master Mode without SSE Control (SSEMOD = 1)

In this master mode, the data is transmitted/received as soon as write data to SPITXB register by S/W. The data in SPITXB will be loaded into SPISR at the same time and start to shift in/out, then transmit buffer empty detect bit (TXBF), and interrupt flag bits (SPITXIF, TXBFIF) are set. And then user could write the next byte data to SPITXB register before the 8-bit data transmission is completed if next byte transmission is needed. If the next byte data is not written into SPITXB, the communication will be stopped after the 8-bit data transmission is completed. Once the 8-bits of data have been received, the data in SPISR will be moved to the SPIRCB register, then buffer full detect bit (RCBF), interrupt flag bits (SPIRCIF, RCBFIF) are set. And then user could read out the SPIRCB register before next 8-bit data transmission is completed if needed.

How to transmit/receive data in this master mode:

1. Enable SPI function by setting the SPION (SPICON<6>) bit.
2. Decide the transmission rate and source by programming SPIM2:SPIM0 (SPICON<2:0>) bits.
3. Write the data that you want to transmit to SPITXB register to start transmit.
4. When the 8-bit data transmission starts, both of the SPITXIF and TXBFIF interrupt flags will set to 1. Besides, both of these bits are cleared by software. The TXBF flag also will be set to “1”, cleared by software or by writting data to SPITXB register.
5. Write next byte data to SPITXB register before this byte transmission being finished if next byte transmission is needed.
6. When the 8-bit data transmission is completed, both of the SPIRCIF and RCBFIF interrupt flags will set to 1. Besides, both of these bits are cleared by software. The RCBF flag also will be set to “1”, cleared by software or by reading out SPIRCB register.
7. Read out the SPIRCB register before next byte transmission being finished if needed.

FIGURE 2.10: SPI Mode Timing (Master Mode)



2.7.2 Slave Mode

2.7.2.1 Slave Mode with SSE Control (SSEMOD = 0)

In this slave mode, the data is transmitted and received as the external clock pulses appear on SCK pin. Once the SPI shift register enable bit SSE (SPICON<4>) has been set to “1”, data in SPITXB will be loaded into SPISR and start to shift in/out, then transmit buffer empty detect bit (TXBF), and interrupt flag bits (SPITXIF, TXBFIF) are set. And then user could write the next byte data to SPITXB register before the 8-bit data transmission is completed if needed. Once the 8-bits of data have been received, the data in SPISR will be moved to the SPIRCB register, then buffer full detect bit (RCBF), interrupt flag bits (SPIRCIF, RCBFIF) are set. And then user could read out the SPIRCB register before next 8-bit data transmission is completed if needed.
The SSB pin allows a synchronous slave mode. The SPI must be in slave mode with SSB pin control enabled (SPICON<2:0> = 101). When the SSB pin is low, transmission and reception are enabled and the SDO pin is driven. When the SSB pin goes high, the SDO pin is no longer driven, even if in the middle of transmitted byte, and becomes a floating output. External pull-up/pull-down resistors may be desirable, depending on the application. When the SPI module resets, the bit counter is forced to 0. This can be done by forcing the SSB pin to high level or clearing the SPION bit (SPICON<6>).

How to transmit/receive data in this slave mode:

1. Enable SPI function by setting the SPION (SPICON<6>) bit.
2. Enable/disable the SSB pin control by programming SPIM2:SPIM0 (SPICON<2:0>) bits.
3. Write the data that you want to transmit to SPITXB register if needed.
4. Set SSE (SPICON<4>) bit and wait the external clock pulses appear on SCK pin to start transmit.
5. When the 8-bit data transmission starts, both of the SPITXIF and TXBFIF interrupt flags will set to 1. Besides, both of these bits are cleared by software. The TXBF flag also will be set to “1”, cleared by software or by writting data to SPITXB register.
6. Write next new data to SPITXB register before this byte transmission being finished if needed.
7. When the 8-bit data transmission is over, the SSE bit will be reset to “0” by hardware. Therefore, if user want to transmit/receive another 8-bit data, user must set SSE bit to “1” again before next clock pulse appearing SCK pin.
8. When the 8-bit data transmission is completed, both of the SPIRCIF and RCBFIF interrupt flags will set to 1.

Besides, both of these bits are cleared by software. The RCBF flag also will be set to “1”, cleared by software or by reading out SPIRCB register.

9. Read out the SPIRCB register before next byte transmission being finished if needed.

2.7.2.2 Slave Mode without SSE Control (SSEMOD = 1)

In this slave mode, the data is transmitted and received as the external clock pulses appear on SCK pin. Once the clock pulse appear on SCK pin, data in SPITXB will be loaded into SPISR and start to shift in/out, then transmit buffer empty detect bit (TXBF), and interrupt flag bits (SPITXIF, TXBFIF) are set. And then user could write the next byte data to SPITXB register before the 8-bit data transmission is completed if needed. Once the 8-bits of data have been received, the data in SPISR will be moved to the SPIRCB register, then buffer full detect bit (RCBF), interrupt flag bits (SPIRCIF, RCBFIF) are set. And then user could read out the SPIRCB register before next 8-bit data transmission is completed if needed.
The SSB pin allows a synchronous slave mode. The SPI must be in slave mode with SSB pin control enabled (SPICON<2:0> = 101). When the SSB pin is low, transmission and reception are enabled and the SDO pin is driven. When the SSB pin goes high, the SDO pin is no longer driven, even if in the middle of transmitted byte, and becomes a floating output. External pull-up/pull-down resistors may be desirable, depending on the application. When the SPI module resets, the bit counter is forced to 0. This can be done by forcing the SSB pin to high level or clearing the SPION bit (SPICON<6>).

How to transmit/receive data in this slave mode:

1. Enable SPI function by setting the SPION (SPICON<6>) bit.
2. Enable/disable the SSB pin control by programming SPIM2:SPIM0 (SPICON<2:0>) bits.
3. Write the data that you want to transmit to SPITXB register if needed.
4. Wait the external clock pulses appear on SCK pin to start transmit.
5. When the 8-bit data transmission starts, both of the SPITXIF and TXBFIF interrupt flags will set to 1. Besides, both of these bits are cleared by software. The TXBF flag also will be set to “1”, cleared by software or by writting data to SPITXB register.
6. Write next new data to SPITXB register before this byte transmission being finished if needed.
7. When the 8-bit data transmission is completed, both of the SPIRCIF and RCBFIF interrupt flags will set to 1. Besides, both of these bits are cleared by software. The RCBF flag also will be set to “1”, cleared by software or by reading out SPIRCB register.
8. Read out the SPIRCB register before next byte transmission being finished if needed.

FIGURE 2.11: SPI Mode Timing (Slave Mode, with SSB control enabled)

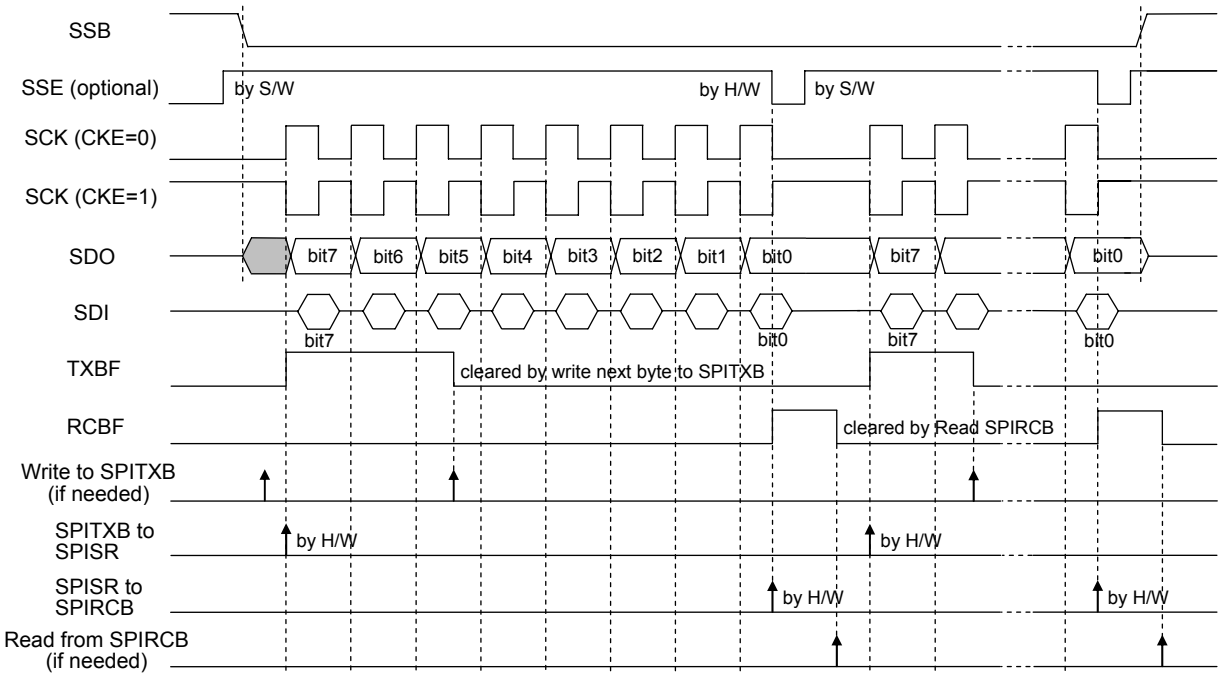
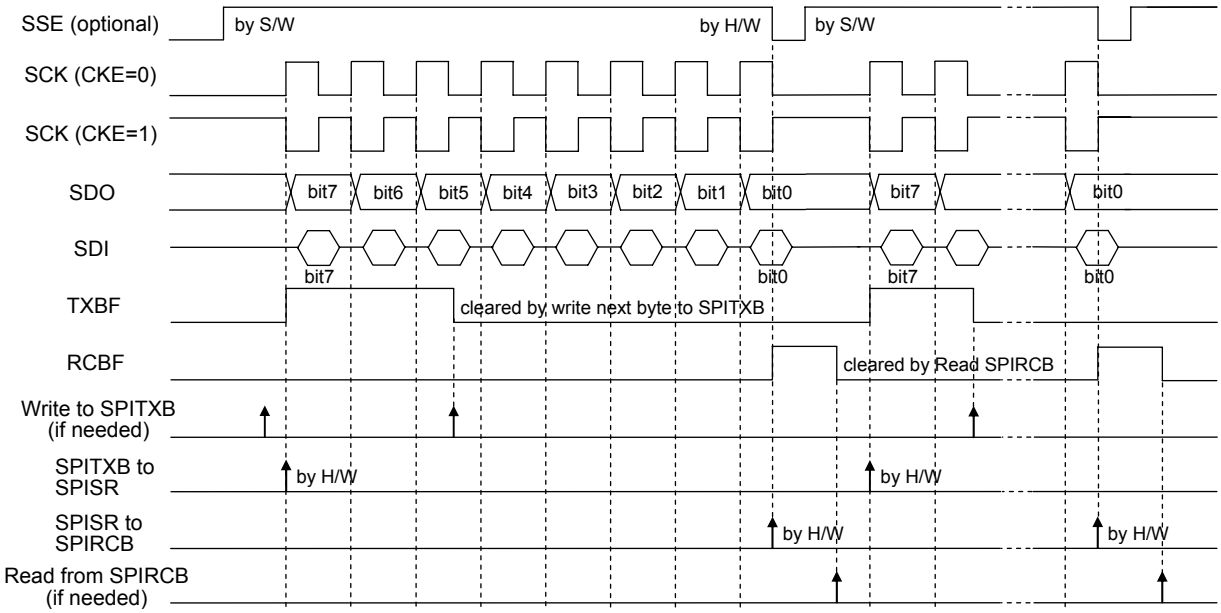


FIGURE 2.12: SPI Mode Timing (Slave Mode, with SSB control disabled)



2.8 Interrupts

The FM8P51 series has up to eight sources of interrupt:
1. External interrupt INT pin.
2. TMR0 overflow interrupt.
3. TMR1 match interrupt.
4. TMR2 match interrupt.
5. TMR3 match interrupt.
6. SPI receive module interrupt.
7. SPI transmit module interrupt.
8. RFC module interrupt.

INTFLAG is the interrupt flag register that recodes the interrupt requests in the relative flags.
A global interrupt enable bit, GIE (OPTION<6>), enables (if set) all un-masked interrupts or disables (if cleared) all interrupts. Individual interrupts can be enabled/disabled through their corresponding enable bits in INTEN register regardless of the status of the GIE bit.
When an interrupt event occur with the GIE bit and its corresponding interrupt enable bit are all set, the GIE bit will be cleared by hardware to disable any further interrupts, and the next instruction will be fetched from address 001h.
The interrupt flag bits must be cleared by software before re-enabling GIE bit to avoid recursive interrupts.
Executing the ENI instruction will set the GIE bit, and executing the DISI instruction will clear the GIE bit.
The RETFIE instruction exits the interrupt routine and set the GIE bit to re-enable interrupt.
The flag bit in INTFLAG register is set by interrupt event regardless of the status of its mask bit. Reading the INTFLAG register will be the logic AND of INTFLAG and INTEN.
When an interrupt is generated by the INT instruction, the next instruction will be fetched from address 002h.

2.8.1 External INT Interrupt

External interrupt on INT pin is falling edge triggered.
When a falling edge appears on the INT pin the flag bit INTIF (INTFLAG<1>) is set. This interrupt can be disables by clearing INTIE bit (INTEN<1>).
The INT0 pin interrupt can wake-up the system from SLEEP condition, if bit INT0IE was set before going to SLEEP. If GIE bit was set, the program will execute interrupt service routine after wake-up; or if GIE bit was cleared, the program will execute next PC after wake-up.

2.8.2 Timer0 Interrupt

An overflow (FFh → 00h) in the TMR0 register will set the flag bit T0IF (INTFLAG<0>).
This interrupt can be disabled by clearing T0IE bit (INTEN<0>).

2.8.3 Timer1 Interrupt

An match condition (TMR1 = PR1) in the TMR1 register will set the flag bits T1IF (INTFLAG<3>) and TM1IF (SPISTAT<4>).
If user wants to clear the T1IF bit, user needs to clear the TM1IF bit first, and then the T1IF bit can be cleared by software.
This interrupt can be disabled by clearing T1IE bit (INTEN<3>).

2.8.4 Timer2 Interrupt

In 8-bit mode, an match condition (TMR2 = PR2) in the TMR2 register will set the flag bit T2IF (INTFLAG<4>).
In 16-bit mode, an match condition (TMR3:TMR2 = PR3:PR2) in the TMR3 and TMR2 register will set the flag bit T2IF (INTFLAG<4>).
This interrupt can be disabled by clearing T2IE bit (INTEN<4>).

2.8.5 Timer3 Interrupt

In 8-bit mode, an match condition (TMR3 = PR3) in the TMR1 register will set the flag bit T3IF (INTFLAG<5>).
In 16-bit mode, the Timer3 interrupt will be disabled.
This interrupt can be disabled by clearing T3IE bit (INTEN<5>).

2.8.6 SPI Receive Module Interrupt

After one byte of SPI transmission is completed, both the flag bit SPIRCIF (INTFLAG<2>) and RCBFIF (SPISTAT<1>) will be set.
If user wants to clear the SPIRCIF bit, user needs to clear the RCBFIF bit first, and then the SPIRCIF bit can be cleared by software.
This interrupt can be disabled by clearing SPIRCIE bit (INTEN<2>).

2.8.7 SPI Transmit Module Interrupt

After one byte of SPI transmission start, both the flag bit SPITXIF (INTFLAG<7>) and TXBFIF (SPISTAT<6>) will be set.
If user wants to clear the SPITXIF bit, user needs to clear the TXBFIF bit first, and then the SPITXIF bit can be cleared by software.
This interrupt can be disabled by clearing SPITXIE bit (INTEN<7>).

2.8.8 RFC Module Interrupt

After RFC conversion is finished, the RFCIF flag (INTFLAG<6>) will be set.
This interrupt can be disabled by clearing RFCIE bit (INTEN<6>).

2.9 Resistor to Frequency Converter (RFC)

The Resistor to Frequency Converter (RFC) can compare nineteen different sensors with the reference resistor separately.
This RFC contains nighteen external pins:
CX : the oscillation schemmit trigger input (IOA1/CX pin).
RFC0 ~ RFC18: the resistor/sensor output pin 0 ~ 18 (IOA2/RFC0, IOA3/RFC1, IOA4/RFC2, IOA5/RFC3, IOA6/RFC4, IOA7/RFC5, IOD0/RFC6, IOD1/RFC7, IOB1/RFC8, IOB2/RFC9, IOB3/RFC10, IOB4/RFC11, IOB5/RFC12, IOB6/RFC13, IOB7/RFC14, IOC2/RFC15, IOC1/RFC16, DATA/RFC17, and CLK/RFC18 pins)

FIGURE 2.13: The Block Diagram of RFC

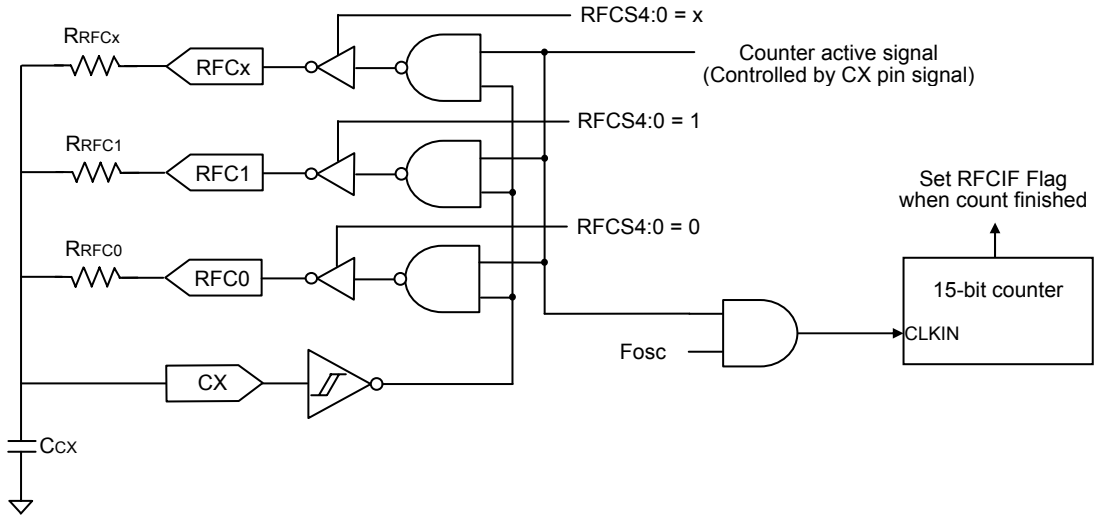


TABLE 2.5: The Description of RFC Control Bits

RFCS4:RFCS0	Select one the RFC oscillation network of RFCx (x = 0 to 18). The selected IOxx/RFCx pin will be configured as RFCx output pin if RFCON = 1. Other IOxx/RFCx pins will still behave as port pins. If RFCON = 0, all IOxx/RFCx pins will still behave as port pins.
START	= 1, RFC counter start to convert. Reset by hardware after conversion is finished. = 0, Stop the RFC conversion
RFCON	= 1, Enable RFC module, the selected IOXX/RFCx pin is configured to RFCx pin, an the IOA1/CX pin is configured to CX pin. = 0, Disable RFC module, all the IOxx/RFCx pins are configured to IOxx pin, and the IOA1/CX pin is configured to IOA1 pin.

2.9.1 RC Oscillator Network

The RFC circuitry may build up 19 RC oscillation networks through RFC0 to RFC18 and CX pins with external resistors. Only one RC oscillation network may be active at a time. When the oscillation network is built up, the count active pulse will be generated by the oscillation network and transferred to the 15-bit counter through the CX pin. It will then enable or disable the 15-bit counter in order to count the oscillation clock. **The 15-bit RFC counter is cleared when a value is written to RFCCON register, RFCON bit is cleared, and during any kind of reset as well.**

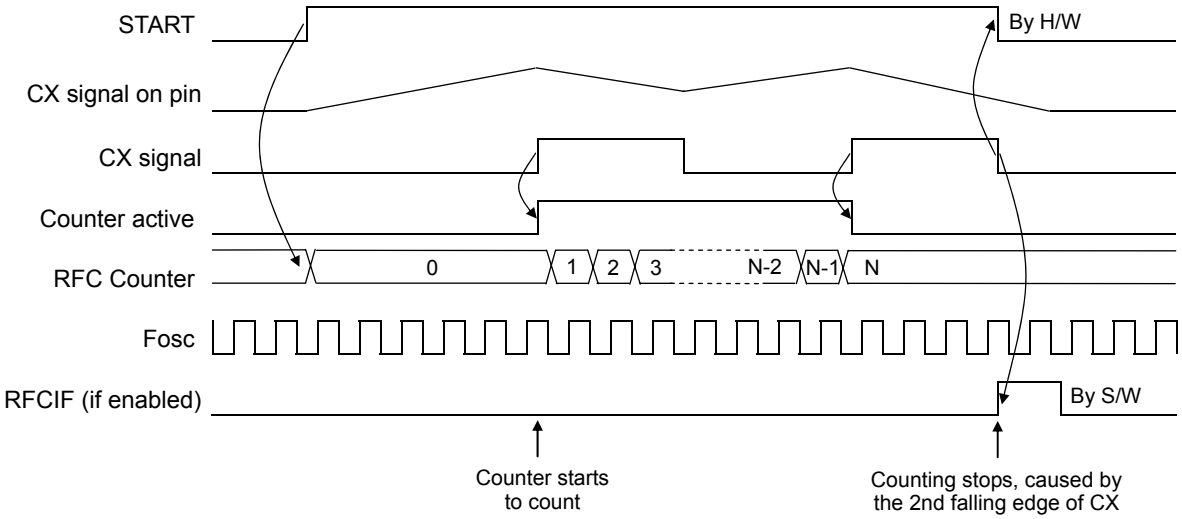
How to built the RC oscillation network:
1. Connect the resistor and capacitor on RFCx (x = 0 to 18, if needed) and CX pins.
2. Switch all of the needed RFCx and CX pins to input mode.

3. Enable the RFC module by set the RFCON bit.
4. Select one of RFCx pins by RFCS4:RFCS0 bits to enable the output pin for RC networks respectively. The selected RFCx will output low at this time. Other RFCx pins will become of a tri-state type.
5. Set START bit to enable the RC oscillation network and 15-bit counter. The RC oscillation network will not operate if this bit has not been set. After conversion is finished, the START bit will be cleared by hardware and the RFCIF flag will be set (if enable).

2.9.2 Enable/Disable the Counter by CX Signal

In this module, CX pin is the signal to control the counter period and the clock source of the counter comes from the internal system clock (Fosc).
The counter will start to count after the first rising edge signal applied on the CX pin after the RFCON bit (RFCCON<7>) is set. Once the second rising edge is applied to the CX pin after the counter is enabled, the counter will stop counting. And after the second falling edge is applied to the CX pin, the RFC block will clear the RFCON bit and set the RFC interrupt flag RFCIF bit (INTFLAG<6>) if RFCIE bit is set.
User also can polling the RFCON or RFCIF bit to check if the conversion is finished.

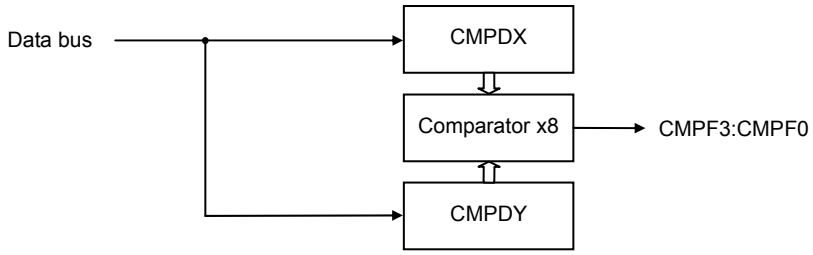
FIGURE 2.14: The Sample of the RFC Counter Controlled by the CX Pin



2.10 Data Comparator

The data comparator can compare two different data in compared data register (CMPDX, CMPDY), and output the number of different/error bits between CMPDX and CMPDY into CMPF3:CMPF0 bits (CMPSTAT<3:0>).
The data comparator will auto-compare the data of CMPDX to the data of CMPDY whenever any one of (CMPDX, CMPDY) is changed.
If user write 00h to CMPDX, and write one 8-bit data to CMPDY, then CMPF3:CMPF0 means the number of "1" of the 8-bit data. Similarly, if user write FFh to CMPDX, and write one 8-bit data to CMPDY, then CMPF3:CMPF0 means the number of "0" of the 8-bit data.

FIGURE 2.15: The Block Diagram of Data Comparator



2.11 Power-down Mode (SLEEP)

Power-down mode is entered by executing a SLEEP instruction.
When SLEEP instruction is executed, the $\overline{PD}$ bit (STATUS<3>) is cleared, the $\overline{TO}$ bit is set, the watchdog timer will be cleared and keeps running, and the oscillator driver is turned off.
All I/O pins maintain the status they had before the SLEEP instruction was executed.

2.11.1 Wake-up from SLEEP Mode

The device can wake-up from SLEEP mode through one of the following events:
1. RSTB reset.
2. WDT time-out reset (if enabled).
3. Input change wake-up.

External RSTB reset and WDT time-out reset will cause a device reset. The $\overline{PD}$ and $\overline{TO}$ bits can be used to determine the cause of device reset. The $\overline{PD}$ bit is set on power-up and is cleared when SLEEP instruction is executed. The $\overline{TO}$ bit is cleared if a WDT time-out occurred.
Any one of the wake-up pins is set to "0", the device will wake-up and continue execution at the instruction after the SLEEP instruction. In this case, before entering SLEEP mode, the wake-up function of trigger sources (IOB0~IOB7, IOC4~IOC5, and IOE0~IOE1) should be selected (e.g. input pin) and enabled (e.g. pull-high, wake-up control (/WUE bit, PCON<0>)).
The system wake-up delay time is 18ms plus 128 oscillator cycle time.

2.12 Reset

FM8P51 devices may be RESET in one of the following ways:

1. Power-on Reset (POR)
2. Brown-out Reset (BOR)
3. RSTB Pin Reset
4. WDT time-out Reset

Some registers are not affected in any RESET condition. Their status is unknown on Power-on Reset and unchanged in any other RESET. Most other registers are reset to a “reset state” on Power-on Reset, RSTB or WDT Reset.

A Power-on RESET pulse is generated on-chip when Vdd rise is detected. To use this feature, the user merely ties the RSTB pin to Vdd.

On-chip Low Voltage Detector (LVD) places the device into reset when Vdd is below a fixed voltage. This ensures that the device does not continue program execution outside the valid operation Vdd range. Brown-out RESET is typically used in AC line or heavy loads switched applications.

A RSTB or WDT Wake-up from SLEEP also results in a device RESET, and not a continuation of operation before SLEEP.

The $\overline{TO}$ and $\overline{PD}$ bits (STATUS<4:3>) are set or cleared depending on the different reset conditions.

2.12.1 Power-up Reset Timer(PWRT)

The Power-up Reset Timer provides a nominal 18/4.5/288/72ms delay after Power-on Reset (POR), Brown-out Reset (BOR), RSTB Reset or WDT time-out Reset. The device is kept in reset state as long as the PWRT is active. The PWDT delay will vary from device to device due to Vdd, temperature, and process variation.

2.12.2 Oscillator Start-up Timer(OST)

The OST timer provides a 16 oscillator cycle delay (from OSCI input) after the PWRT delay (18/4.5/288/72ms) is over. This delay ensures that the X’tal oscillator or resonator has started and stabilized. The device is kept in reset state as long as the OST is active.

This counter only starts incrementing after the amplitude of the OSCI signal reaches the oscillator input thresholds.

2.12.3 Reset Sequence

When Power-on Reset (POR), Brown-out Reset (BOR), RSTB Reset or WDT time-out Reset is detected, the reset sequence is as follows:

1. The reset latch is set and the PWRT & OST are cleared.
2. When the internal POR, BOR, RSTB Reset or WDT time-out Reset pulse is finished, then the PWRT begins counting.
3. After the PWRT time-out, the OST is activated.
4. And after the OST delay is over, the reset latch will be cleared and thus end the on-chip reset signal.

The totally system reset delay time is 18/4.5/288/72ms plus 16 oscillator cycle time.

FIGURE 2.16: Simplified Block Diagram of on-chip Reset Circuit

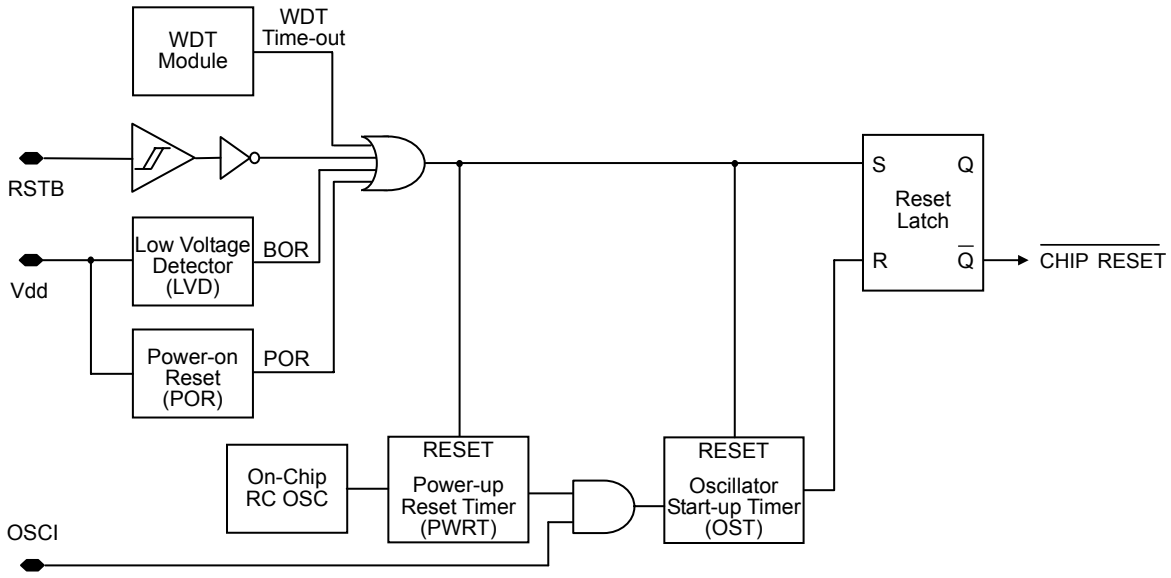


TABLE 2.6: Reset Conditions for All Registers

Register	Address	Power-on Reset Brown-out Reset	RSTB Reset	WDT Reset
ACC	N/A	xxxx xxxx	uuuu uuuu	uuuu uuuu
OPTION	N/A	1011 1111	1011 1111	1u11 1111
IOSTA	05h	1111 1111	1111 1111	1111 1111
IOSTB	06h	1111 1111	1111 1111	1111 1111
IOSTC	07h	1111 1111	1111 1111	1111 1111
IOSTD	08h	1111 1111	1111 1111	1111 1111
IOSTE	09h	A: 1111 1111 B: 0011 1111	A: 1111 1111 B: 0011 1111	A: 1111 1111 B: 0011 1111
T1CON	0Ch	0000 0000	0000 0000	0000 0000
PHCON	0Dh	1111 1111	1111 1111	1111 1111
PCON	0Eh	101- 0--1	101- 0--1	101- 0--1
INTEN	0Fh	-000 0000	-000 0000	-000 0000
INDF	00h, unbanked	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR0	01h, unbanked	0000 0000	0000 0000	0000 0000
PCL	02h, unbanked	1111 1111	1111 1111	1111 1111
STATUS	03h, unbanked	0001 1xxx	000# #uuu	000# #uuu
FSR	04h, unbanked	00xx xxxx	00uu uuuu	00uu uuuu
PORTA	05h, unbanked	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTB	06h, unbanked	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTC	07h, unbanked	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTD	08h, unbanked	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTE	09h, unbanked	A: xxxx xxxx B: --xx xxxx	A: uuuu uuuu B: --uu uuuu	A: uuuu uuuu B: --uu uuuu
SPIRCB	0Ah, bank 0	xxxx xxxx	uuuu uuuu	uuuu uuuu

SPITXB	0Bh, bank 0	xxxx xxxx	uuuu uuuu	uuuu uuuu
SPISTAT	0Ch, bank 0	0000 0000	0000 0000	0000 0000
SPICON	0Dh, bank 0	0000 0000	0000 0000	0000 0000
TMR1	0Eh, bank 0	0000 0000	0000 0000	0000 0000
PR1	0Fh, bank 0	1111 1111	1111 1111	1111 1111
T23CON	0Ah, bank 1	---0 0000	---0 0000	---0 0000
TMR2	0Bh, bank 1	0000 0000	0000 0000	0000 0000
PR2	0Ch, bank 1	1111 1111	1111 1111	1111 1111
TMR3	0Dh, bank 1	0000 0000	0000 0000	0000 0000
PR3	0Eh, bank 1	1111 1111	1111 1111	1111 1111
PWMCON	0Ah, bank 2	---- -000	---- -000	---- -000
PW0DCL	0Bh, bank 2	00-- ----	00-- ----	00-- ----
PW0DCH	0Ch, bank 2	0000 0000	0000 0000	0000 0000
PW1DCL	0Dh, bank 2	00-- ----	00-- ----	00-- ----
PW1DCH	0Eh, bank 2	0000 0000	0000 0000	0000 0000
RFCCON	0Ah, bank 3	000- 0000	000- 0000	000- 0000
RFCDL	0Bh, bank 3	0000 0000	0000 0000	0000 0000
RFCDH	0Ch, bank 3	0000 0000	0000 0000	0000 0000
CMPDX	0Dh, bank 3	0000 0000	0000 0000	0000 0000
CMPDY	0Eh, bank 3	0000 0000	0000 0000	0000 0000
CMPSTAT	0Fh, bank 3	---- 0000	---- 0000	---- 0000
INTFLAG	3Fh, unbanked	0000 0000	0000 0000	0000 0000
General Purpose Registers		xxxx xxxx	uuuu uuuu	uuuu uuuu

Legend: u = unchanged, x = unknown, - = unimplemented,
= refer to the following table for possible values.

TABLE 2.7: $\overline{TO}$ / $\overline{PD}$ Status after Reset

$\overline{TO}$	$\overline{PD}$	RESET was caused by
1	1	Power-on Reset
1	1	Brown-out reset
u	u	RSTB Reset during normal operation
1	0	RSTB Reset during SLEEP
0	1	WDT Reset during normal operation
0	0	WDT Wake-up during SLEEP

Legend: u = unchanged

TABLE 2.8: Events Affecting $\overline{TO}$ / $\overline{PD}$ Status Bits

Event	$\overline{TO}$	$\overline{PD}$
Power-on	1	1
WDT Time-Out	0	u
SLEEP instruction	1	0
CLRWDT instruction	1	1

Legend: u = unchanged

2.13 Hexadecimal Convert to Decimal (HCD)

Decimal format is another number format for FM8P51 series. When the content of the data memory has been assigned as decimal format, it is necessary to convert the results to decimal format after the execution of ALU instructions. When the decimal converting operation is processing, all of the operand data (including the contents of the data memory (RAM), accumulator (ACC), immediate data, and look-up table) should be in the decimal format, or the results of conversion will be incorrect.
Instruction DAA can convert the ACC data from hexadecimal to decimal format after any addition operation and restored to ACC.
The conversion operation is illustrated in example 2.2.

EXAMPLE 2.2: DAA CONVERSION

```
MOVIA   90h      ;Set immediate data = decimal format number “90” (ACC ← 90h)
MOVAR   30h      ;Load immediate data “90” to data memory address 30H
MOVIA   10h      ;Set immediate data = decimal format number “10” (ACC ← 10h)
ADDAR   30h, 0   ;Contents of the data memory address 30H and ACC are binary-added
                 ;the result loads to the ACC (ACC ← A0h, C ← 0)
DAA              ;Convert the content of ACC to decimal format, and restored to ACC
                 ;The result in the ACC is “00” and the carry bit C is “1”. This represents the
                 ;decimal number “100”
```

Instruction DAS can convert the ACC data from hexadecimal to decimal format after any subtraction operation and restored to ACC.
The conversion operation is illustrated in example 2.3.

EXAMPLE 2.3: DAS CONVERSION

```
MOVIA   10h      ;Set immediate data = decimal format number “10” (ACC ← 10h)
MOVAR   30h      ;Load immediate data “10” to data memory address 30H
MOVIA   20h      ;Set immediate data = decimal format number “20” (ACC ← 20h)
SUBAR   30h, 0   ;Contents of the data memory address 30H and ACC are binary-subtracted
                 ;the result loads to the ACC (ACC ← F0h, C ← 0)
DAS              ;Convert the content of ACC to decimal format, and restored to ACC
                 ;The result in the ACC is “90” and the carry bit C is “0”. This represents the
                 ;decimal number “ -10”
```

2.14 Oscillator Configurations

FM8P51 series can be operated in four different oscillator modes. Users can program two configuration bits (Fosc<1:0>) to select the appropriate modes:

- LF: Low Frequency Crystal Oscillator
- HF: High Frequency Crystal/Resonator Oscillator
- ERIC: External Resistor/Internal Capacitor Oscillator
- ERC: External Resistor/Capacitor Oscillator

In LF, or HF modes, a crystal or ceramic resonator in connected to the OSCI and OSCO pins to establish oscillation. When in LF, or HF modes, the devices can have an external clock source drive the OSCI pin.
The ERC/ERIC device option offers additional cost savings for timing insensitive applications. The RC oscillator frequency is a function of the supply voltage, the resistor (Rext) and capacitor (Cext), the operating temperature, and the process parameter.

FIGURE 2.17: HF, or LF Oscillator Modes (Crystal Operation or Ceramic Resonator)

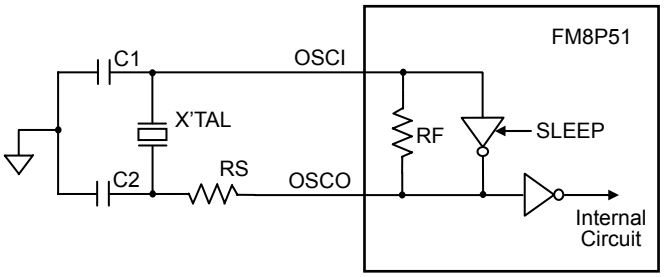


FIGURE 2.18: HF, or LF Oscillator Modes (External Clock Input Operation)

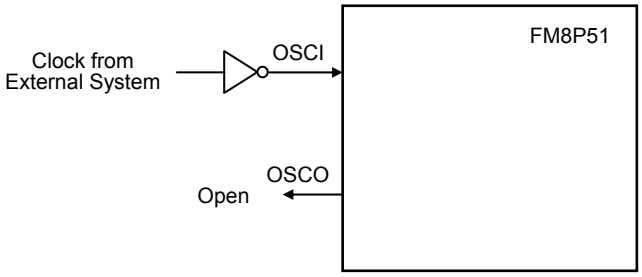


FIGURE 2.19: ERC Oscillator Mode (External RC Oscillator)

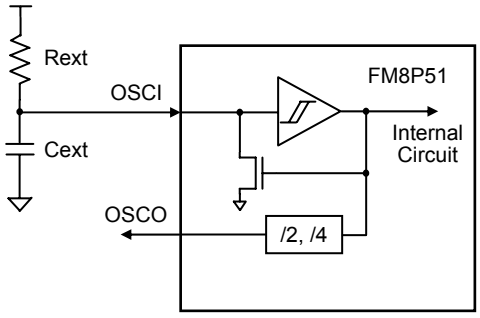
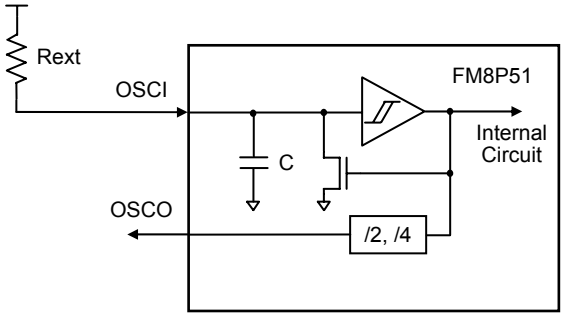


FIGURE 2.20: ERIC Oscillator Mode (External R, Internal C Oscillator)



2.15 Configurations Word

TABLE 2.9: Configurations Word 0

Name	Description
Fosc	Oscillator Selection Bits = 1, 1 → ERC mode (external R & C) (default) = 1, 0 → ERIC mode (external R & internal C) = 0, 1 → HF mode = 0, 0 → LF mode
WDTEN	Watchdog Timer Enable Bit = 1, WDT enabled (default) = 0, WDT disabled
PROTECT	Code Protection Bit = 1, EPROM code protection off (default) = 0, EPROM code protection on
LVDT	Low Voltage Detector Selection Bit = 1, 1 → disable (default) = 0, 1 → enable, LVDT voltage = 2.0V = 0, 0 → enable, LVDT voltage = 3.6V
OSCD	Instruction Period Selection Bit = 1, four oscillator periods (default) = 0, two oscillator periods
CKOUT	IOE7/CKOUT Selection Bits = 1, 1 → enable IOE7 (default) = 0, 1 → enable CKOUT, same phase of OSCI = 0, 0 → enable CKOUT, opposite phase of OSCI
SUT	Reset Delay Time Selection Bits = 1, 1 → 18 ms (default) = 1, 0 → 4.5 ms = 0, 1 → 288 ms = 0, 0 → 72 ms
TYPE	Type Selection Bit = 1, A type (44-pin) is selected (default) = 0, B type (40-pin) is selected

TABLE 2.10: Configurations Word 1

Name	Description
PMOD	Power Mode Selection Bits = 1 → Non-power saving (default) = 0 → Power saving
DEL	SDI Input Delay Time Selection Bits = 1, 1 → 0ns (default) = 0, 1 → 50ns = 1, 0 → 100ns

3.0 INSTRUCTION SET

Mnemonic, Operands		Description	Operation	Cycles	Status Affected
BCR	**R, bit**	Clear bit in R	0 → R<b>	1	-
BSR	**R, bit**	Set bit in R	1 → R<b>	1	-
BTRSC	**R, bit**	Test bit in R, Skip if Clear	Skip if R<b> = 0	1/2 (1)	-
BTRSS	**R, bit**	Test bit in R, Skip if Set	Skip if R<b> = 1	1/2 (1)	-
NOP		No Operation	No operation	1	-
CLRWDT		Clear Watchdog Timer	00h → WDT, 00h → WDT prescaler	1	$\overline{TO}$, $\overline{PD}$
OPTION		Load OPTION register	ACC → OPTION	1	-
OPTIONR		Read OPTION register	OPTION → ACC	1	-
SLEEP		Go into power-down mode	00h → WDT, 00h → WDT prescaler	1	$\overline{TO}$, $\overline{PD}$
DAA		Adjust ACC's data format from HEX to DEC after any addition operation	ACC(hex) → ACC(dec)	1	C
DAS		Adjust ACC's data format from HEX to DEC after any subtraction operation	ACC(hex) → ACC(dec)	1	-
TBL		Table look-up	PC<7:0> + ACC → PC<7:0> PC<9:8> unchanged PG<1:0> → PC<11:10>	1	C, DC, Z
INT		S/W interrupt	PC + 1 → Top of Stack, 002h → PC	2	-
ENI		Set GIE bit	1 → GIE	1	-
DISI		Clear GIE bit	0 → GIE	1	-
RETURN		Return from subroutine	Top of Stack → PC	2	-
RETFIE		Return from interrupt, set GIE bit	Top of Stack → PC, 1 → GIE	2	-
CLRA		Clear ACC	00h → ACC	1	Z
IOST	**R**	Load IOST register	ACC → IOST register	1	-
IOSTR	**R**	Read IOST register	IOST register → ACC	1	-
CLRR	**R**	Clear R	00h → R	1	Z
MOVAR	**R**	Move ACC to R	ACC → R	1	-
MOVR	**R, d**	Move R	R → dest	1	Z
DECR	**R, d**	Decrement R	R - 1 → dest	1	Z
DECRSZ	**R, d**	Decrement R, Skip if 0	R - 1 → dest, Skip if result = 0	1/2 (1)	-
INCR	**R, d**	Increment R	R + 1 → dest	1	Z
INCRSZ	**R, d**	Increment R, Skip if 0	R + 1 → dest, Skip if result = 0	1/2 (1)	-
ADDAR	**R, d**	Add ACC and R	R + ACC → dest	1	C, DC, Z
SUBAR	**R, d**	Subtract ACC from R	R - ACC → dest	1	C, DC, Z
ADCAR	**R, d**	Add ACC and R with Carry	R + ACC + C → dest	1	C, DC, Z
SBCAR	**R, d**	Subtract ACC from R with Carry	R + $\overline{ACC}$ + C → dest	1	C, DC, Z
ANDAR	**R, d**	AND ACC with R	ACC and R → dest	1	Z

IORAR	**R, d**	Inclusive OR ACC with R	ACC or R → dest	1	Z
XORAR	**R, d**	Exclusive OR ACC with R	R xor ACC → dest	1	Z
COMR	**R, d**	Complement R	$\overline{R}$ → dest	1	Z
RLR	**R, d**	Rotate left f through Carry	R<7> → C, R<6:0> → dest<7:1>, C → dest<0>	1	C
RRR	**R, d**	Rotate right f through Carry	C → dest<7>, R<7:1> → dest<6:0>, R<0> → C	1	C
SWAPR	**R, d**	Swap R	R<3:0> → dest<7:4>, R<7:4> → dest<3:0>	1	-
MOVIA	**I**	Move Immediate to ACC	I → ACC	1	-
ADDIA	**I**	Add ACC and Immediate	I + ACC → ACC	1	C, DC, Z
SUBIA	**I**	Subtract ACC from Immediate	I - ACC → ACC	1	C, DC, Z
ANDIA	**I**	AND Immediate with ACC	ACC and I → ACC	1	Z
IORIA	**I**	OR Immediate with ACC	ACC or I → ACC	1	Z
XORIA	**I**	Exclusive OR Immediate to ACC	ACC xor I → ACC	1	Z
RETIA	**I**	Return, place Immediate in ACC	I → ACC, Top of Stack → PC	2	-
CALL	**I**	Call subroutine	PC + 1 → Top of Stack, I → PC<9:0> PG<1:0> → PC<11:10>	2	-
GOTO	**I**	Unconditional branch	I → PC<9:0> PG<1:0> → PC<11:10>	2	-

Note: 1. 2 cycles for skip, else 1 cycle

2. bit : Bit address within an 8-bit register R
   R : Register address (00h to 3Fh)
   I : Immediate data
   ACC : Accumulator
   d : Destination select;
   =0 (store result in ACC)
   =1 (store result in file register R)
   dest : Destination
   PC : Program Counter
   PG : Program Memory Page Select Bits
   WDT : Watchdog Timer Counter
   GIE : Global interrupt enable bit
   $\overline{TO}$: Time-out bit
   $\overline{PD}$: Power-down bit
   C : Carry bit
   DC : Digital carry bit
   Z : Zero bit

ADCAR	**Add ACC and R with Carry**
Syntax:	ADCAR R, d
Operands:	$0 \leq R \leq 63$ $d \in [0,1]$
Operation:	R + ACC + C → dest
Status Affected:	C, DC, Z
Description:	Add the contents of the ACC register and register 'R' with Carry. If 'd' is 0 the result is stored in the ACC register. If 'd' is '1' the result is stored back in register 'R'.
Cycles:	1

ADDAR	**Add ACC and R**
Syntax:	ADDAR R, d
Operands:	$0 \leq R \leq 63$ $d \in [0,1]$
Operation:	ACC + R → dest
Status Affected:	C, DC, Z
Description:	Add the contents of the ACC register and register 'R'. If 'd' is 0 the result is stored in the ACC register. If 'd' is '1' the result is stored back in register 'R'.
Cycles:	1

ADDIA	**Add ACC and Immediate**
Syntax:	ADDIA I
Operands:	$0 \leq I \leq 255$
Operation:	ACC + I → ACC
Status Affected:	C, DC, Z
Description:	Add the contents of the ACC register with the 8-bit immediate 'I'. The result is placed in the ACC register.
Cycles:	1

ANDAR	**AND ACC and R**
Syntax:	ANDAR R, d
Operands:	$0 \leq R \leq 63$ $d \in [0,1]$
Operation:	ACC and R → dest
Status Affected:	Z
Description:	The contents of the ACC register are AND'ed with register 'R'. If 'd' is 0 the result is stored in the ACC register. If 'd' is '1' the result is stored back in register 'R'.
Cycles:	1

ANDIA	**AND Immediate with ACC**
Syntax:	ANDIA I
Operands:	$0 \leq I \leq 255$
Operation:	ACC AND I → ACC
Status Affected:	Z
Description:	The contents of the ACC register are AND'ed with the 8-bit immediate 'I'. The result is placed in the ACC register.
Cycles:	1

BCR	**Clear Bit in R**
Syntax:	BCF R, b
Operands:	0≤R≤63
	0≤b≤7
Operation:	0 → R<b>
Status Affected:	None
Description:	Clear bit 'b' in register 'R'.
Cycles:	1

BSR	**Set Bit in R**
Syntax:	BSR R, b
Operands:	0≤R≤63
	0≤b≤7
Operation:	1 → R<b>
Status Affected:	None
Description:	Set bit 'b' in register 'R'.
Cycles:	1

BTRSC	**Test Bit in R, Skip if Clear**
Syntax:	BTRSC R, b
Operands:	0≤R≤63
	0≤b≤7
Operation:	Skip if R<b> = 0
Status Affected:	None
Description:	If bit 'b' in register 'R' is 0 then the next instruction is skipped. If bit 'b' is 0 then next instruction fetched during the current instruction execution is discarded, and a NOP is executed instead making this a 2-cycle instruction..
Cycles:	1(2)

BTRSS	**Test Bit in R, Skip if Set**
Syntax:	BTRSS R, b
Operands:	0≤R≤63
	0≤b≤7
Operation:	Skip if R<b> = 1
Status Affected:	None
Description:	If bit 'b' in register 'R' is '1' then the next instruction is skipped. If bit 'b' is '1', then the next instruction fetched during the current instruction execution, is discarded and a NOP is executed instead, making this a 2-cycle instruction.
Cycles:	1(2)

CALL	**Subroutine Call**
Syntax:	CALL I
Operands:	0≤I≤1023
Operation:	PC +1 → Top of Stack; I → PC<9:0> PG<1:0> → PC<11:10>
Status Affected:	None
Description:	Subroutine call. First, return address (PC+1) is pushed onto the stack. The 10-bit immediate address is loaded into PC bits <9:0>. CALL is a two-cycle instruction.
Cycles:	2

CLRA	**Clear ACC**
Syntax:	CLRA
Operands:	None
Operation:	00h → ACC; 1 → Z
Status Affected:	Z
Description:	The ACC register is cleared. Zero bit (Z) is set.
Cycles:	1

CLRR	**Clear R**
Syntax:	CLRR R
Operands:	$0 \leq R \leq 63$
Operation:	00h → R; 1 → Z
Status Affected:	Z
Description:	The contents of register 'R' are cleared and the Z bit is set.
Cycles:	1

CLRWDT	**Clear Watchdog Timer**
Syntax:	CLRWDT
Operands:	None
Operation:	00h → WDT; 00h → WDT prescaler (if assigned); 1 → $\overline{TO}$; 1 → $\overline{PD}$
Status Affected:	$\overline{TO}$, $\overline{PD}$
Description:	The CLRWDT instruction resets the WDT. It also resets the prescaler, if the prescaler is assigned to the WDT and not Timer0. Status bits $\overline{TO}$ and $\overline{PD}$ are set.
Cycles:	1

COMR	**Complement R**
Syntax:	COMR R, d
Operands:	$0 \leq R \leq 63$ $d \in [0,1]$
Operation:	$\overline{R}$ → dest
Status Affected:	Z
Description:	The contents of register 'R' are complemented. If 'd' is 0 the result is stored in the ACC register. If 'd' is 1 the result is stored back in register 'R'.
Cycles:	1

DAA	**Adjust ACC's data format from HEX to DEC**
Syntax:	DAA
Operands:	None
Operation:	ACC(hex) → ACC(dec)
Status Affected:	C
Description:	Convert the ACC data from hexadecimal to decimal format after any addition operation and restored to ACC.
Cycles:	1

DAS	**Adjust ACC's data format from HEX to DEC**
Syntax:	DAS
Operands:	None
Operation:	ACC(hex) → ACC(dec)
Status Affected:	None
Description:	Convert the ACC data from hexadecimal to decimal format after any subtraction operation and restored to ACC.
Cycles:	1

DECR	**Decrement R**
Syntax:	DECR R, d
Operands:	$0 \leq R \leq 63$ $d \in [0,1]$
Operation:	R - 1 → dest
Status Affected:	Z
Description:	Decrement register 'R'. If 'd' is 0 the result is stored in the ACC register. If 'd' is 1 the result is stored back in register 'R'.
Cycles:	1

DECRSZ	**Decrement R, Skip if 0**
Syntax:	DECRSZ R, d
Operands:	$0 \leq R \leq 63$ $d \in [0,1]$
Operation:	R - 1 → dest; skip if result =0
Status Affected:	None
Description:	The contents of register 'R' are decremented. If 'd' is 0 the result is placed in the ACC register. If 'd' is 1 the result is placed back in register 'R'. If the result is 0, the next instruction, which is already fetched, is discarded and a NOP is executed instead making it a two-cycle instruction.
Cycles:	1(2)

DISI	**Clear GIE bit**
Syntax:	DISI
Operands:	None
Operation:	0 → GIE
Status Affected:	None
Description:	Disable interrupt.
Cycles:	1

ENI	**Set GIE bit**
Syntax:	ENI
Operands:	None
Operation:	1 → GIE
Status Affected:	None
Description:	Enable interrupt.
Cycles:	1

GOTO	**Unconditional Branch**
Syntax:	GOTO I
Operands:	$0 \leq I \leq 1023$
Operation:	I → PC<9:0> PG<1:0> → PC<11:10>
Status Affected:	None
Description:	GOTO is an unconditional branch. The 10-bit immediate value is loaded into PC bits <9:0>. GOTO is a two-cycle instruction.
Cycles:	2

INCR	**Increment R**
Syntax:	INCR R, d
Operands:	$0 \leq R \leq 63$ $d \in [0,1]$
Operation:	R + 1 → dest
Status Affected:	Z
Description:	The contents of register 'R' are incremented. If 'd' is 0 the result is placed in the ACC register. If 'd' is 1 the result is placed back in register 'R'.
Cycles:	1

INCRSZ	**Increment R, Skip if 0**
Syntax:	INCRSZ R, d
Operands:	$0 \leq R \leq 63$ $d \in [0,1]$
Operation:	R + 1 → dest, skip if result = 0
Status Affected:	None
Description:	The contents of register 'R' are incremented. If 'd' is 0 the result is placed in the ACC register. If 'd' is the result is placed back in register 'R'. If the result is 0, then the next instruction, which is already fetched, is discarded and a NOP is executed instead making it a two-cycle instruction.
Cycles:	1(2)

INT	**S/W Interrupt**
Syntax:	INT
Operands:	None
Operation:	PC + 1 → Top of Stack, 002h → PC
Status Affected:	None
Description:	Interrupt subroutine call. First, return address (PC+1) is pushed onto the stack. The address 002h is loaded into PC bits <10:0>.
Cycles:	2

IORAR	**OR ACC with R**
Syntax:	IORAR R, d
Operands:	$0 \leq R \leq 63$ $d \in [0,1]$
Operation:	ACC or R → dest
Status Affected:	Z
Description:	Inclusive OR the ACC register with register 'R'. If 'd' is 0 the result is placed in the ACC register. If 'd' is 1 the result is placed back in register 'R'.
Cycles:	1

IORIA	**OR Immediate with ACC**
Syntax:	IORIA I
Operands:	$0 \leq I \leq 255$
Operation:	ACC or I → ACC
Status Affected:	Z
Description:	The contents of the ACC register are OR'ed with the 8-bit immediate 'I'. The result is placed in the ACC register.
Cycles:	1

IOST	**Load IOST Register**
Syntax:	IOST R
Operands:	R = 5,6,7,8,9,12,13,14 or 15
Operation:	ACC → IOST register R
Status Affected:	None
Description:	IOST register 'R' (5,6,7,8,9,12,13,14 or 15) is loaded with the contents of the ACC register.
Cycles:	1

IOSTR	**Read IOST Register**
Syntax:	IOSTR R
Operands:	R = 5,6,7,8, 9, C, D, E or F
Operation:	IOST register R → ACC
Status Affected:	None
Description:	The ACC register is loaded with the contents of IOST register 'R' (5,6,7,8,9,12,13,14 or 15).
Cycles:	1

MOVAR	**Move ACC to R**
Syntax:	MOVAR R
Operands:	$0 \leq R \leq 63$
Operation:	ACC → R
Status Affected:	None
Description:	Move data from the ACC register to register 'R'.
Cycles:	1

MOVIA	**Move Immediate to ACC**
Syntax:	MOVIA I
Operands:	$0 \leq I \leq 255$
Operation:	I → ACC
Status Affected:	None
Description:	The 8-bit immediate 'I' is loaded into the ACC register. The don't cares will assemble as 0s.
Cycles:	1

MOVR	**Move R**
Syntax:	MOVR R, d
Operands:	$0 \leq R \leq 63$ $d \in [0,1]$
Operation:	R → dest
Status Affected:	Z
Description:	The contents of register 'R' is moved to destination 'd'. If 'd' is 0, destination is the ACC register. If 'd' is 1, the destination is file register 'R'. 'd' is 1 is useful to test a file register since status flag Z is affected.
Cycles:	1

NOP	**No Operation**
Syntax:	NOP
Operands:	None
Operation:	No operation
Status Affected:	None
Description:	No operation.
Cycles:	1

OPTION	**Load OPTION Register**
Syntax:	OPTION
Operands:	None
Operation:	ACC → OPTION
Status Affected:	None
Description:	The content of the ACC register is loaded into the OPTION register.
Cycles:	1

OPTIONR	**Read OPTION Register**
Syntax:	OPTION
Operands:	None
Operation:	OPTION → ACC
Status Affected:	None
Description:	The content of the OPTION register is loaded into the ACC register.
Cycles:	1

RETFIE	**Return from Interrupt, Set 'GIE' Bit**
Syntax:	RETFIE
Operands:	None
Operation:	Top of Stack → PC
Status Affected:	None
Description:	The program counter is loaded from the top of the stack (the return address). The 'GIE' bit is set to 1. This is a two-cycle instruction.
Cycles:	2

RETIA	**Return with Immediate in ACC**
Syntax:	RETIA I
Operands:	$0 \leq I \leq 255$
Operation:	I → ACC; Top of Stack → PC
Status Affected:	None
Description:	The ACC register is loaded with the 8-bit immediate 'I'. The program counter is loaded from the top of the stack (the return address). This is a two-cycle instruction.
Cycles:	2

RETURN	**Return from Subroutine**
Syntax:	RETURN
Operands:	None
Operation:	Top of Stack → PC
Status Affected:	None
Description:	The program counter is loaded from the top of the stack (the return address). This is a two-cycle instruction.
Cycles:	2

RLR	**Rotate Left f through Carry**
Syntax:	RLR R, d
Operands:	$0 \leq R \leq 63$ $d \in [0,1]$
Operation:	R<7> → C; R<6:0> → dest<7:1>; C → dest<0>
Status Affected:	C
Description:	The contents of register 'R' are rotated one bit to the left through the Carry Flag. If 'd' is 0 the result is placed in the ACC register. If 'd' is 1 the result is stored back in register 'R'.
Cycles:	1

RRR	**Rotate Right f through Carry**
Syntax:	RRR R, d
Operands:	$0 \leq R \leq 63$ $d \in [0,1]$
Operation:	C → dest<7>; R<7:1> → dest<6:0>; R<0> → C
Status Affected:	C
Description:	The contents of register 'R' are rotated one bit to the right through the Carry Flag. If 'd' is 0 the result is placed in the ACC register. If 'd' is 1 the result is placed back in register 'R'.
Cycles:	1

SLEEP	**Enter SLEEP Mode**
Syntax:	SLEEP
Operands:	None
Operation:	00h → WDT; 00h → WDT prescaler; 1 → $\overline{TO}$; 0 → $\overline{PD}$
Status Affected:	$\overline{TO}$, $\overline{PD}$
Description:	Time-out status bit ($\overline{TO}$) is set. The power-down status bit ($\overline{PD}$) is cleared. The WDT and its prescaler are cleared. The processor is put into SLEEP mode.
Cycles:	1

SBCAR	**Subtract ACC from R with Carry**
Syntax:	SBCAR R, d
Operands:	$0 \leq R \leq 63$ $d \in [0,1]$
Operation:	R + $\overline{ACC}$ + C → dest
Status Affected:	C, DC, Z
Description:	Add the 2's complement data of the ACC register from register 'R' with Carry. If 'd' is 0 the result is stored in the ACC register. If 'd' is 1 the result is stored back in register 'R'.
Cycles:	1

SUBAR	**Subtract ACC from R**
Syntax:	SUBAR R, d
Operands:	$0 \leq R \leq 63$ $d \in [0,1]$
Operation:	R - ACC → dest
Status Affected:	C, DC, Z
Description:	Subtract (2's complement method) the ACC register from register 'R'. If 'd' is 0 the result is stored in the ACC register. If 'd' is 1 the result is stored back in register 'R'.
Cycles:	1

SUBIA	**Subtract ACC from Immediate**
Syntax:	SUBIA I
Operands:	$0 \leq I \leq 255$
Operation:	I - ACC → ACC
Status Affected:	C, DC, Z
Description:	Subtract (2's complement method) the ACC register from the 8-bit immediate 'I'. The result is placed in the ACC register.
Cycles:	1

SWAPR	**Swap nibbles in R**
Syntax:	SWAPR R, d
Operands:	$0 \leq R \leq 63$ $d \in [0,1]$
Operation:	R<3:0> → dest<7:4>; R<7:4> → dest<3:0>
Status Affected:	None
Description:	The upper and lower nibbles of register 'R' are exchanged. If 'd' is 0 the result is placed in ACC register. If 'd' is 1 the result in placed in register 'R'.
Cycles:	1

TBL	**Table Look-up**
Syntax:	TBL
Operands:	None
Operation:	PC<7:0> + ACC → PC<7:0> PC<9:8> unchanged PG<1:0> → PC<11:10>
Status Affected:	C, DC, Z
Description:	Operate with RETIA to look-up table
Cycles:	1

XORAR	**Exclusive OR ACC with R**
Syntax:	XORAR R, d
Operands:	$0 \leq R \leq 63$ $d \in [0,1]$
Operation:	ACC xor R → dest
Status Affected:	Z
Description:	Exclusive OR the contents of the ACC register with register 'R'. If 'd' is 0 the result is stored in the ACC register. If 'd' is 1 the result is stored back in register 'R'.
Cycles:	1

XORIA	Exclusive OR Immediate with ACC
Syntax:	XORIA I
Operands:	$0 \leq I \leq 255$
Operation:	ACC xor I → ACC
Status Affected:	Z
Description:	The contents of the ACC register are XOR'ed with the 8-bit immediate 'I'. The result is placed in the ACC register.
Cycles:	1

4.0 ABSOLUTE MAXIMUM RATINGS

Ambient Operating Temperature	0℃ to +70℃
Store Temperature	-65℃ to +150℃
DC Supply Voltage (Vdd)	0V to +6.0V
Input Voltage with respect to Ground (Vss)	-0.3V to (Vdd + 0.3)V

5.0 OPERATING CONDITIONS

DC Supply Voltage	+2.3V to +5.5V
Operating Temperature	0℃ to +70℃

6.0 ELECTRICAL CHARACTERISTICS

6.1 ELECTRICAL CHARACTERISTICS of FM8P51E

To be defined

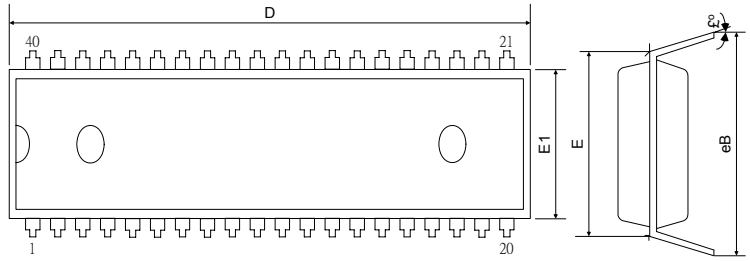
6.2 ELECTRICAL CHARACTERISTICS of FM8P51

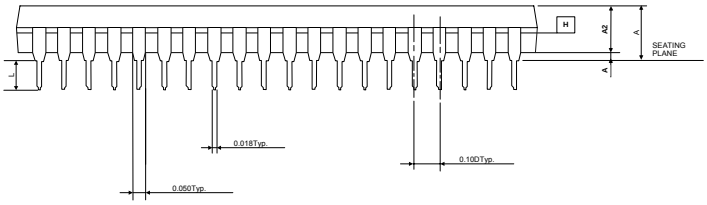
Under Operating Conditions, at four clock instruction cycles and WDT & LVDT are disabled

Sym	Description	Conditions	Min.	Typ.	Max.	Unit
F_{HF}	X'tal oscillation range	HF mode, Vdd=5V	1		20	MHz
		HF mode, Vdd=3V				
F_{LF}	X'tal oscillation range	LF mode, Vdd=5V	32		1000	KHz
		LF mode, Vdd=3V				
F_{ERIC}	RC oscillation range	ERIC mode, Vdd=5V	DC			MHZ
		ERIC mode, Vdd=3V	DC		10	
F_{ERC}	RC oscillation range	ERC mode, Vdd=5V	DC			MHz
		ERC mode, Vdd=3V	DC		10	
V_{IH}	Input high voltage	I/O ports, Vdd=5V	2.0			V
		RSTB pin, Vdd=5V	3.8			
		I/O ports, Vdd=3V	1.6			
		RSTB pin, Vdd=3V	2.4			
V_{IL}	Input low voltage	I/O ports, Vdd=5V			1.0	V
		RSTB pin, Vdd=5V			1.0	
		I/O ports, Vdd=3V			0.6	
		RSTB pin, Vdd=3V			0.6	
V_{OH1}	Output high voltage (IOC0,IOC1,IOC2)	I_{OH}=9mA, Vdd=5V, HDC = 1	2.4			V
		I_{OH}=12mA, Vdd=5V, HDC = 0	2.4			V
V_{OH2}	Output high voltage (other output pins)	I_{OH}=12mA, Vdd=5V	2.4			V
V_{OL1}	Output low voltage (IOC0,IOC1,IOC2)	I_{OL}=5.4mA, Vdd=5V, HDC = 1			0.4	V
		I_{OL}=8mA, Vdd=5V, HDC = 0			0.4	V
V_{OL2}	Output low voltage (other output pins)	I_{OL}=8mA, Vdd=5V			0.4	V
I_{PH}	Pull-high current	Input pin at Vss, Vdd=5V		-53		uA
I_{WDT}	WDT current	Vdd=5V		5.8		uA
		Vdd=3V		0.9		
T_{WDT}	WDT period	Vdd=5V				mS
		Vdd=3V				
I_{LVDT}	LVDT current	Vdd=5V				uA
		Vdd=3V				
I_{SB}	Power down current	Sleep mode, Vdd=5V, WDT enable		7	10	uA
		Sleep mode, Vdd=5V, WDT disable		1.2		
		Sleep mode, Vdd=3V, WDT enable		1.5		
		Sleep mode, Vdd=3V, WDT disable		0.6		
I_{DD}	Operating current	2MHz, Vdd = 5V				mA
		32KHz, Vdd = 3V	15	20	30	uA

2.0 PACKAGE DIMENSION

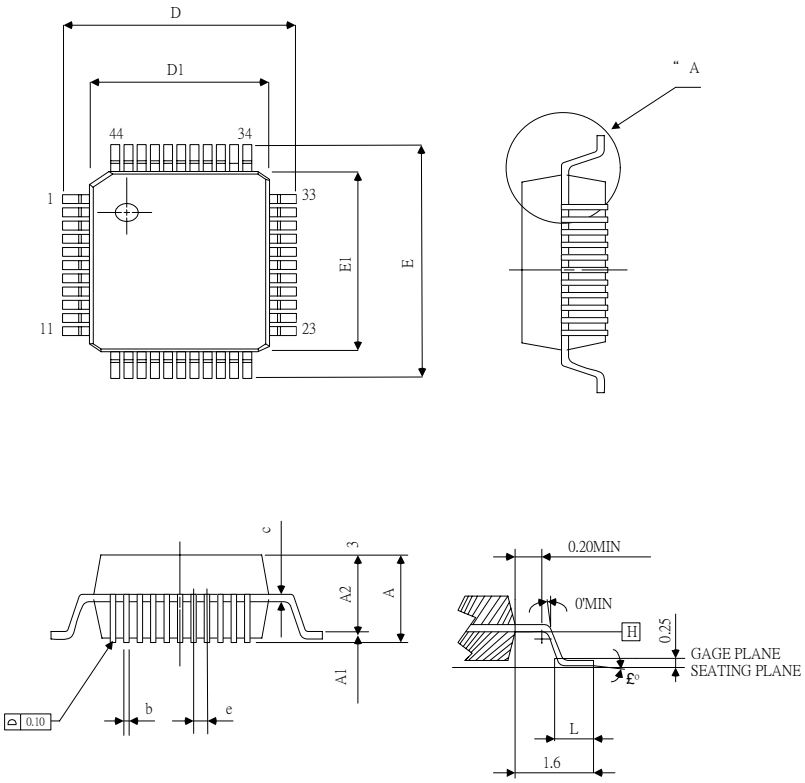
7.1 40-PIN PDIP 600mil





Symbols	Dimension In Inches		
	Min	Nom	Max
A	-	-	0.220
A1	0.015	-	-
A2	0.150	0.155	0.160
D	2.055	2.060	2.070
E		0.600 BSC	
E1	0.540	0.545	0.550
L	0.115	0.130	0.150
eB	0.630	0.650	0.670
θ°	0°	7°	15°

7.2 44-PIN QFP



Symbols	Dimension In Millimeters		
	Min	Nom	Max
A	-	-	2.7
A1	0.25	-	0.50
A2	1.9	2.0	2.2
b		0.3 (TYP.)	
D	13.00	13.20	13.40
D1	9.9	10.00	10.10
E	13.00	13.20	13.40
E1	9.9	10.00	10.10
L	0.73	0.88	0.93
e		0.80 (TYP.)	
θ°	0°	-	7°
C	0.1	0.15	0.2

8.0 ORDERING INFORMATION

OTP Type MCU	Package Type	Pin Count	Package Size
FM8P51EP	PDIP	40	600 mil
FM8P51EF	QFP	44	10mm x 10mm

Mask Type MCU	Package Type	Pin Count	Package Size
FM8P51P	PDIP	40	600 mil
FM8P51F	QFP	44	10mm x 10mm