

**ABOV SEMICONDUCTOR
8-BIT SINGLE-CHIP MICROCONTROLLERS**

MC71PB204

USER'S MANUAL (Ver. 1.0)



TABLE OF CONTENTS

1. PRODUCT OVERVIEW	1
2. MEMORY ORGANIZATION	12
3. INTERRUPT.....	26
4. INSTRUCTION SET.....	30
5. CLOCK CIRCUIT	72
6. RESET AND POWER-DOWN MODE.....	78
7. I/O PORTS	84
8. WATCHDOG TIMER.....	94
9. 16-BIT TIMER 0 (8-BIT TIMER A/B).....	98
10. 6/8-BIT PWM.....	106
11. 12-BIT ANALOG TO DIGITAL CONVERTER	110
12. ELECTRICAL DATA.....	116
13. MECHANICAL DATA.....	126
14. MC71PB204 OTP	130
15. DEVELOPMENT TOOLS	132

1. PRODUCT OVERVIEW

1.1 KEY FEATURES

• CPU	GMC14 core (8-bit RISC CPU)
• Main Clock	Crystal/Ceramic: 0.4MHz – 12MHz Internal RC: 1MHz, 2MHz, 4MHz, 8MHz External RC: 1MHz – 8MHz
• ROM Capacity	4,096 x 14-bits (7,168-Byte)
• RAM Capacity	208 x 8-bits
• Instruction Set	35 single word instructions 14-bit wide instruction word
• Instruction Execution Times	167nS at 12MHz fx (main)
• I/O Port	I/O : 17 bits (20-pin package) I/O : 13 bits (16-pin package)
• Programmable Timer	One 16-bit timer/counter (shared with two 8-bit timer/counters)
• Watchdog Timer Function		
• PWM mode	6/8-bit selection
• A/D Converter	12-bit x 11-channel analog input
• ROM Option	LVR (2.6V, 3.0V, 4.0V) Oscillator selectable (Internal or External)
• Interrupt	External : external interrupt x 4 Internal : WDT interrupt, Timer 0/A interrupt, Timer B interrupt, PWM interrupt.
• Power Supply Voltage	2.4V to 5.5V at 4MHz 2.7V to 5.5V at 8MHz 4.0V to 5.5V at 12MHz
• Operating Temperature	– 20 °C to + 85 °C
• Power-Saving	Idle : only CPU clock stop Stop: System clock and CPU clock stop
• Package	20-pin DIP, 20-pin SOP 16-pin DIP, 16-pin SOP

1.2 Ordering Information

Device	ROM Size	RAM size	Package
MC71PB204D	4K words OTP	208 bytes	20 SOP
MC71PB204B	4K words OTP	208 bytes	20 DIP
MC71PB204M	4K words OTP	208 bytes	16 SOP
MC71PB204V	4K words OTP	208 bytes	16 DIP

1.3 BLOCK DIAGRAM

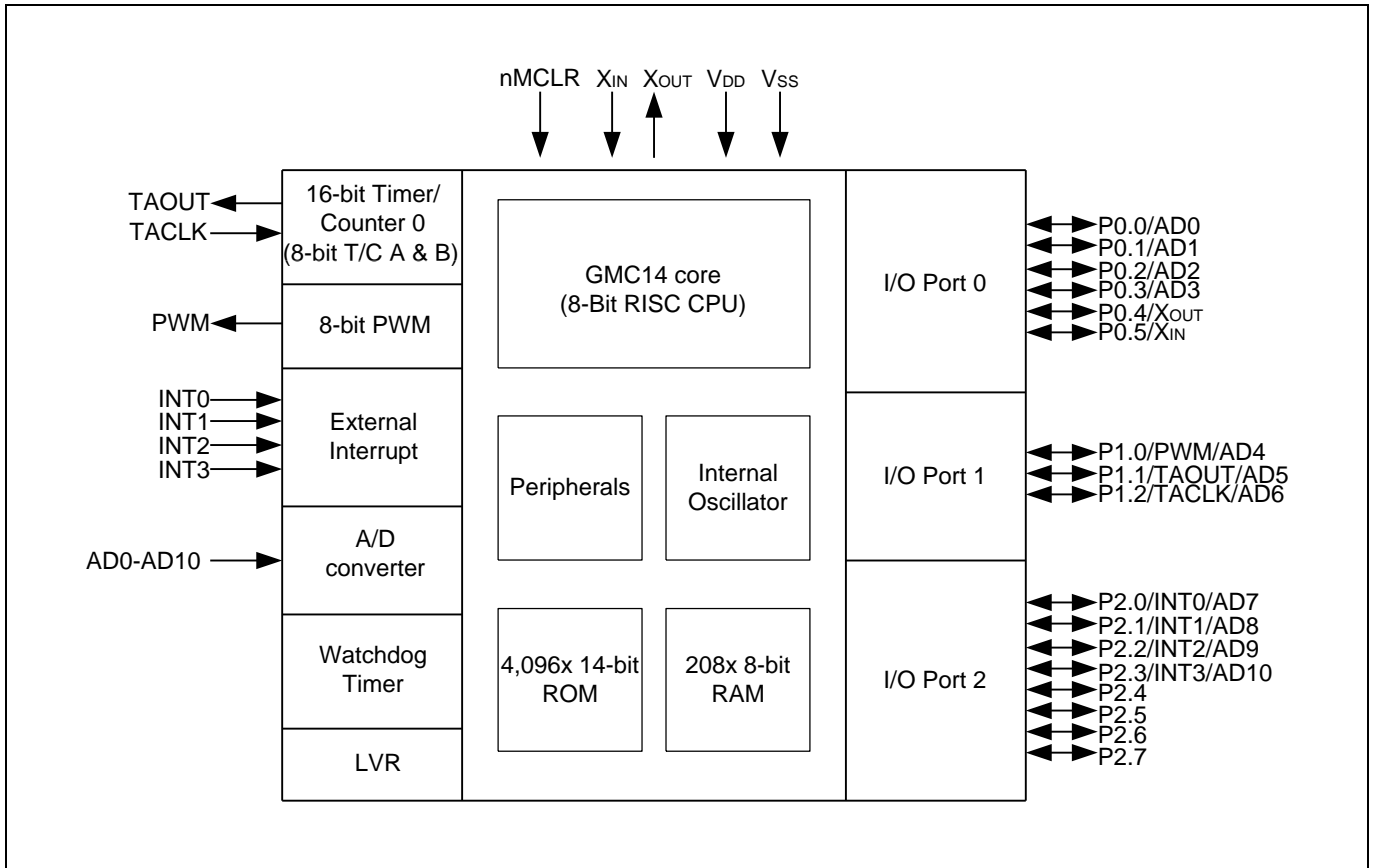


Figure 1-1. Block Diagram

1.4 PIN ASSIGNMENTS

1.4.1 20-PIN PACKAGE

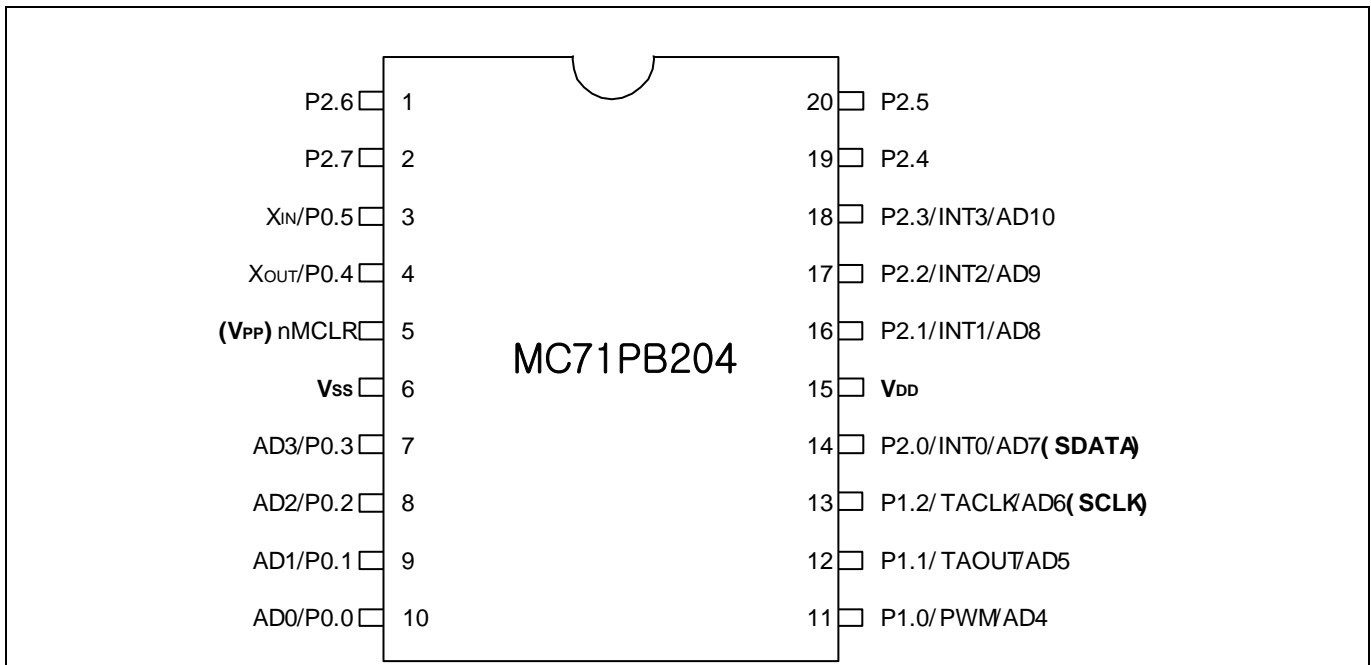


Figure 1-2. MC71PB204 Pin Assignments (20-Pin)

1.4.2 16-PIN PACKAGE

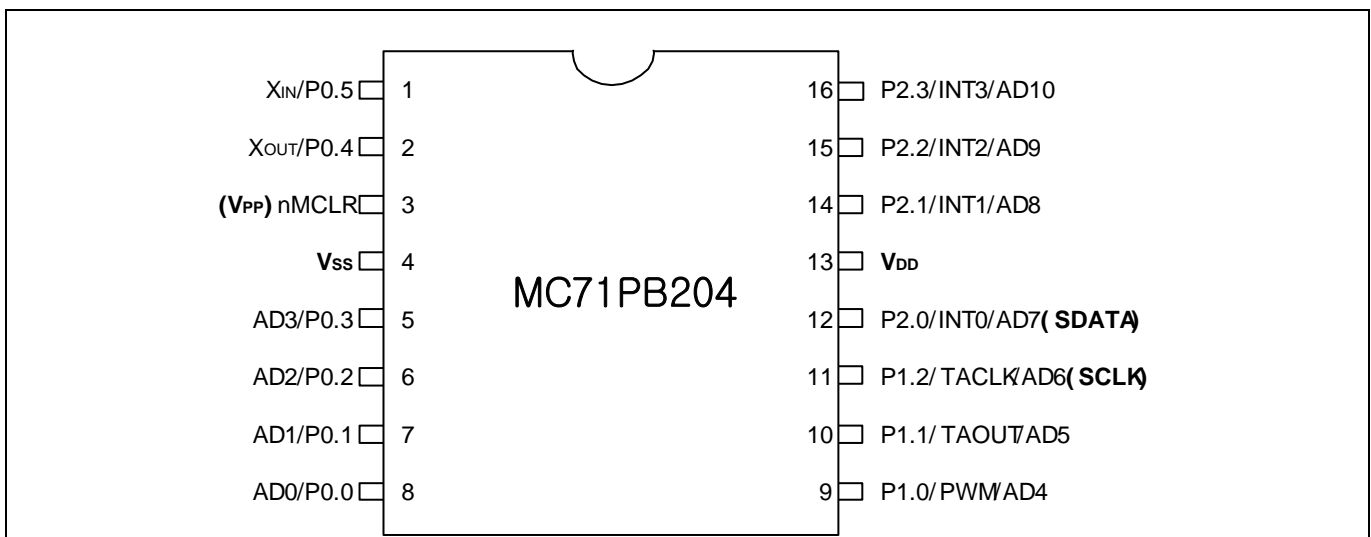


Figure 1-3. MC71PB204 Pin Assignments (16-Pin)

1.5 PIN DESCRIPTIONS

1.5.1 20-DIP(SOP) PACKAGE TYPE

Pin Names	I/O	Pin Description	After RESET	Alternative Functions
P0.0 P0.1 P0.2 P0.3	I/O	1-bit programmable I/O pin. Schmitt trigger input, Push-pull output, or Open-drain output port. Used as an input port, a Pull-up resistor can be programmed as 1-bit.	Input	AD0 AD1 AD2 AD3
P0.4 P0.5	I/O	1-bit programmable I/O pin. Schmitt trigger input, Push-pull output, or Open-drain output port. Used as an input port, a Pull-up resistor or Pull-down resistor can be programmed as 1-bit.	Input	X _{OUT} X _{IN}
P1.0 P1.1 P1.2	I/O	1-bit programmable I/O pin. Schmitt trigger input, Push-pull output, or Open-drain output port. Used as an input port, a Pull-up resistor can be programmed as 1-bit.	Input	PWM/AD4 TAOUT/AD5 TACLK/AD6
P2.0 P2.1 P2.2 P2.3	I/O	1-bit programmable I/O pin. Schmitt trigger Input, Push-pull output port. Used as an input port, a Pull-up resistor can be programmed as 1-bit.	Input	INT0/AD7 INT1/AD8 INT2/AD9 INT3/AD10
P2.4 - P2.7	I/O	1-bit programmable I/O pin. Schmitt trigger Input, Push-pull output, or Open-drain output port. Used as an input port, a Pull-up resistor can be programmed as 1-bit.	Input	-
PWM	I/O	6/8 bit high speed PWM output	Input	P1.0
TACLK	I/O	Timer 0/A external clock input	Input	P1.2
TAOUT	I/O	Timer 0/A data output	Input	P1.1
INT0 INT1 INT2 INT3	I/O	External interrupt input pins	Input	P2.0 P2.1 P2.2 P2.3
AD0 AD1 AD2 AD3 AD4 AD5 AD6 AD7 AD8	I/O	A/D Converter input	Input	P0.0 P0.1 P0.2 P0.3 P1.0 P1.1 P1.2 P2.0 P2.1

1.5.1 20-DIP(SOP) PACKAGE TYPE (CONTINUED)

Pin Names	I/O	Pin Description	After RESET	Alternative Functions
AD9 AD10	I/O	A/D Converter input	Input	P2.2 P2.3
nMCLR	I	System reset pin	Input	-
X _{IN} , X _{OUT}	-	Main oscillator pins	Output	P0.4, P0.5
V _{DD} , V _{SS}	-	Power input pins	-	-

1.5.2 16-DIP(SOP) PACKAGE TYPE

Pin Names	I/O	Pin Description	After RESET	Alternative Functions
P0.0 P0.1 P0.2 P0.3	I/O	1-bit programmable I/O pin. Schmitt trigger input, Push-pull output, or Open-drain output port. Used as an input port, a Pull-up resistor can be programmed as 1-bit.	Input	AD0 AD1 AD2 AD3
P0.4 P0.5	I/O	1-bit programmable I/O pin. Schmitt trigger input, Push-pull output, or Open-drain output port. Used as an input port, a Pull-up resistor or Pull-down resistor can be programmed as 1-bit.	Input	X _{OUT} X _{IN}
P1.0 P1.1 P1.2	I/O	1-bit programmable I/O pin. Schmitt trigger input, Push-pull output, or Open-drain output port. Used as an input port, a Pull-up resistor can be programmed as 1-bit.	Input	PWM/AD4 TAOUT/AD5 TACLK/AD6
P2.0 P2.1 P2.2 P2.3	I/O	1-bit programmable I/O pin. Schmitt trigger Input, Push-pull output port. Used as an input port, a Pull-up resistor can be programmed as 1-bit.	Input	INT0/AD7 INT1/AD8 INT2/AD9 INT3/AD10
PWM	I/O	6/8 bit high speed PWM output	Input	P1.0
TACLK	I/O	Timer 0/A external clock input	Input	P1.2
TAOUT	I/O	Timer 0/A data output	Input	P1.1
INT0 INT1 INT2 INT3	I/O	External interrupt input pins	Input	P2.0 P2.1 P2.2 P2.3
AD0 AD1 AD2 AD3 AD4 AD5 AD6 AD7 AD8 AD9 AD10	I/O	A/D Converter input	Input	P0.0 P0.1 P0.2 P0.3 P1.0 P1.1 P1.2 P2.0 P2.1 P2.2 P2.3
nMCLR	I	System reset pin	Input	-
X _{IN} , X _{OUT}	-	Main oscillator pins	Output	P0.4, P0.5
V _{DD} , V _{SS}	-	Power input pins	-	-

1.6 PIN CIRCUITS

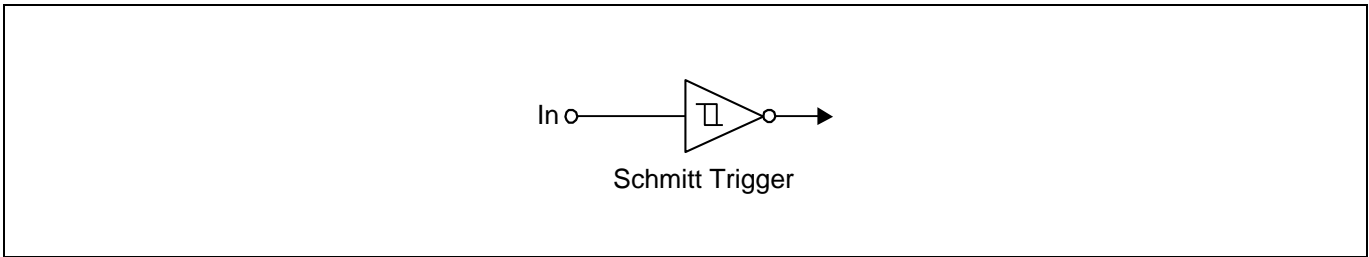


Figure 1-4. Pin Circuit Type 1 (nMCLR)

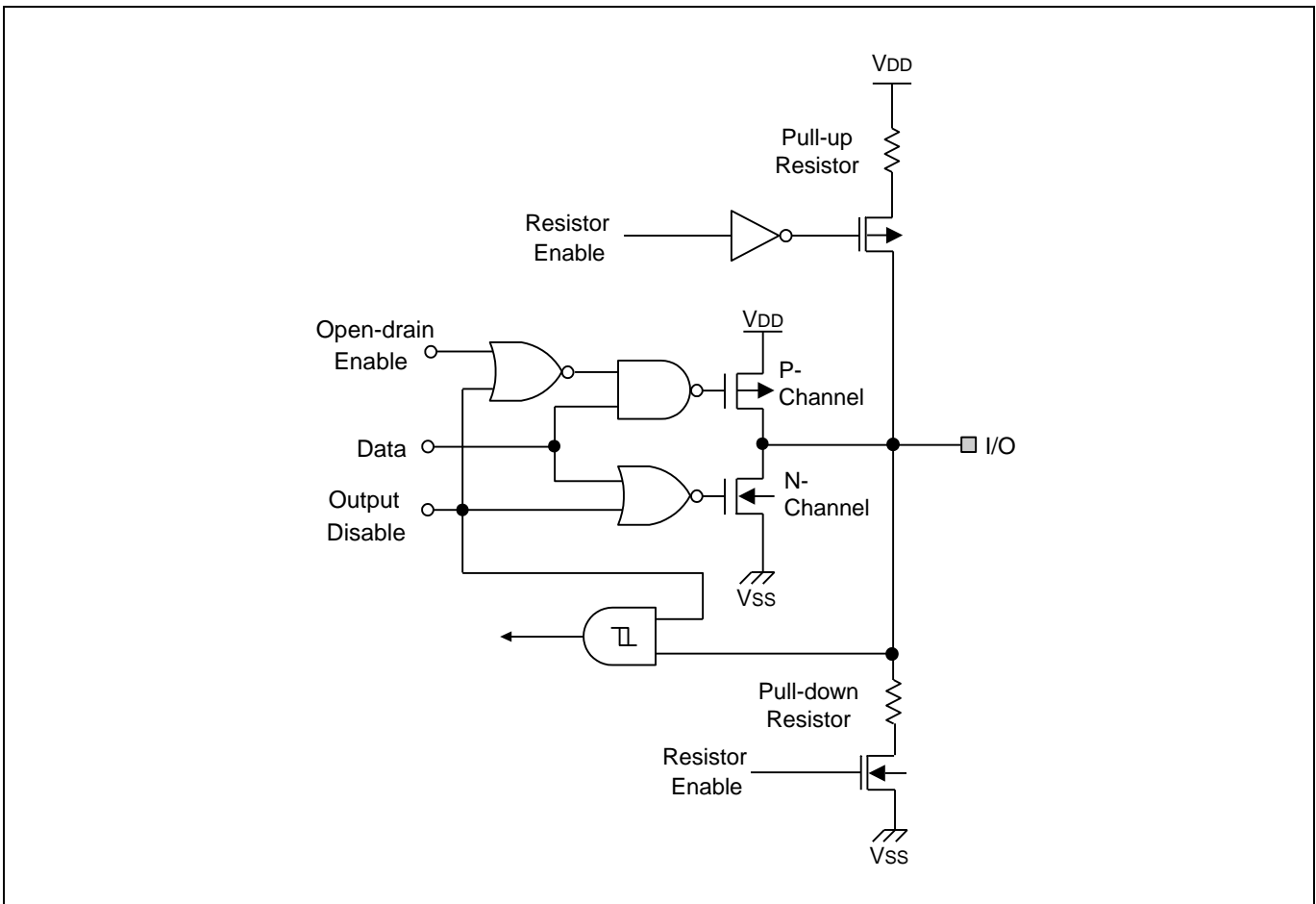


Figure 1-5. Pin Circuit Type 4-2 (P0.4/X_{OUT}, P0.5/X_{IN})

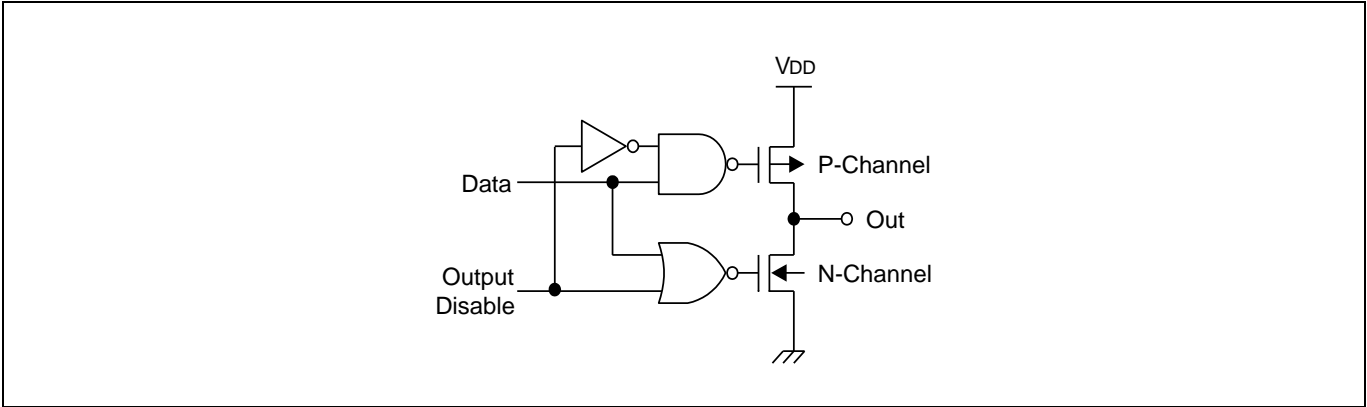


Figure 1-6. Pin Circuit Type 2

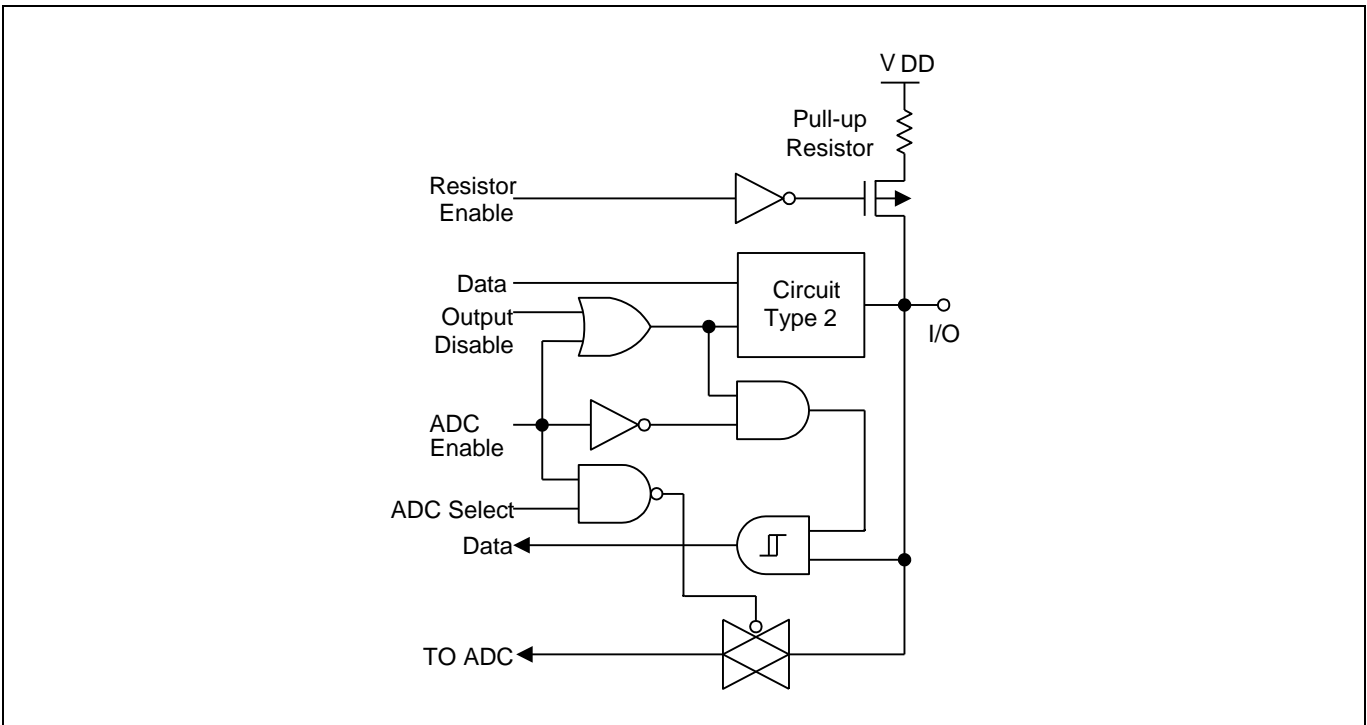


Figure 1-7. Pin Circuit Type 6-1 (P2.0-P2.3)

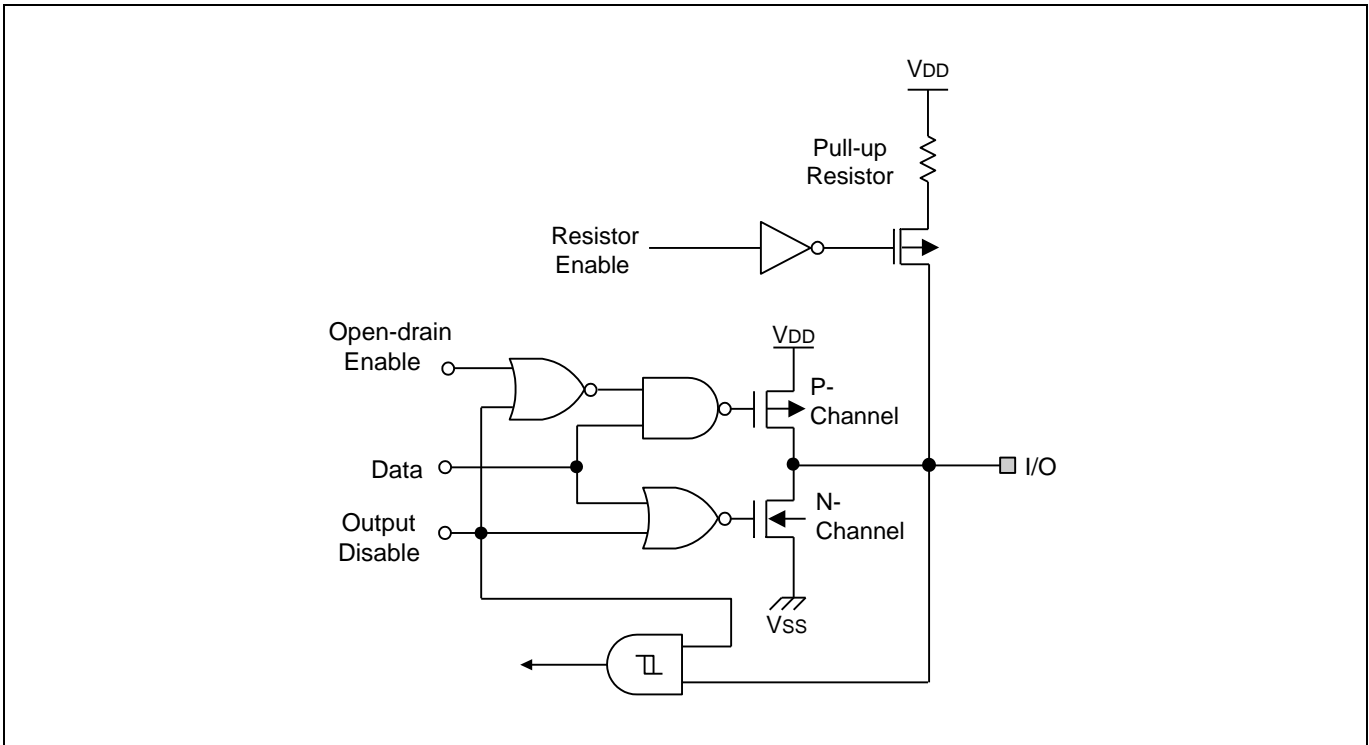


Figure 1-8. Pin Circuit Type 4-1 (P2.4-P2.7)

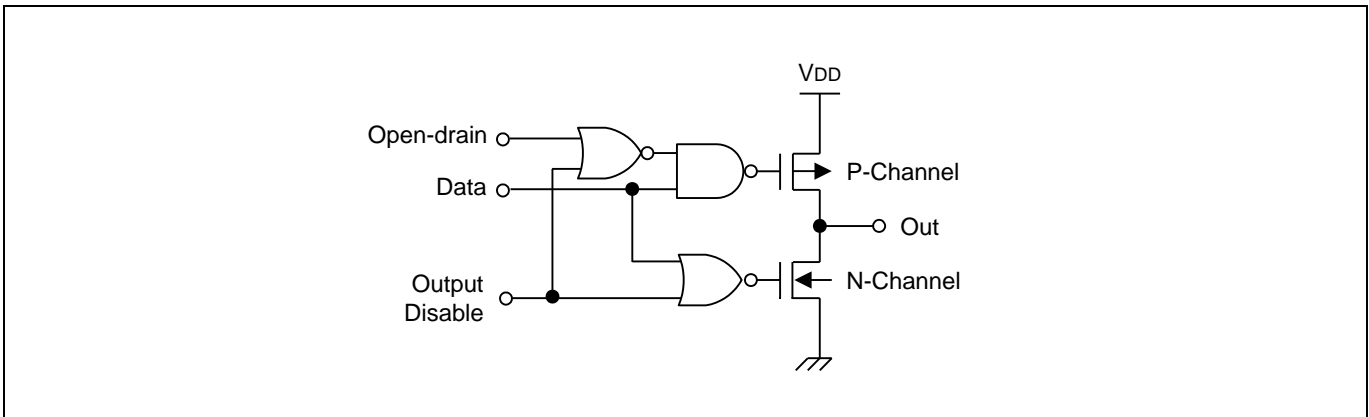


Figure 1-9. Pin Circuit Type 3

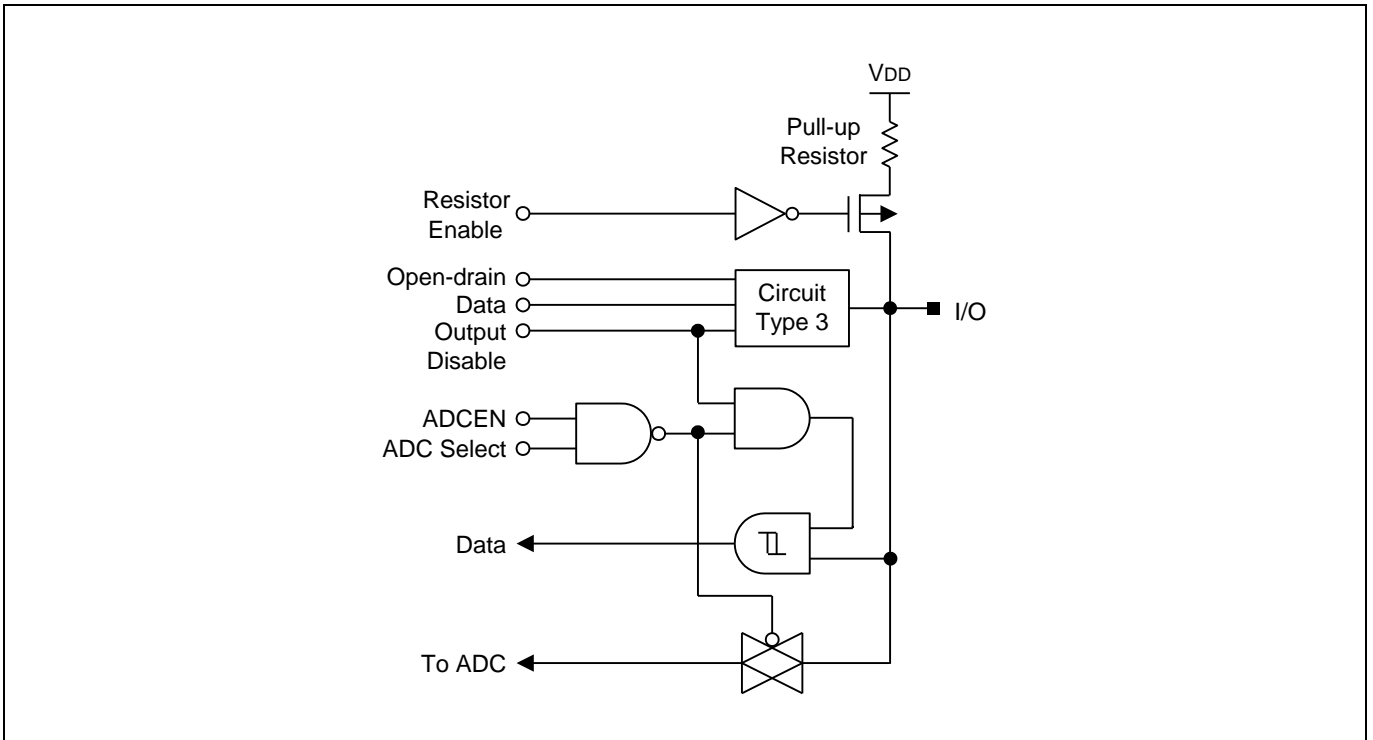


Figure 1-10. Pin Circuit Type 6-3 (P0.0-P0.3, P1.0-P1.2)

2. MEMORY ORGANIZATION

2.1 PROGRAM MEMORY ORGANIZATION

The GMC14 series have a 16-bit program counter capable of addressing a 64k x 14-bit program memory space. The reset vector is at 0000H, the interrupt vector at 0004H and the ROM Option at 001FH. The program memory size of the MC71PB204 is 4k words (4k x 14-bit, from the address 0000H to 0FFFH).

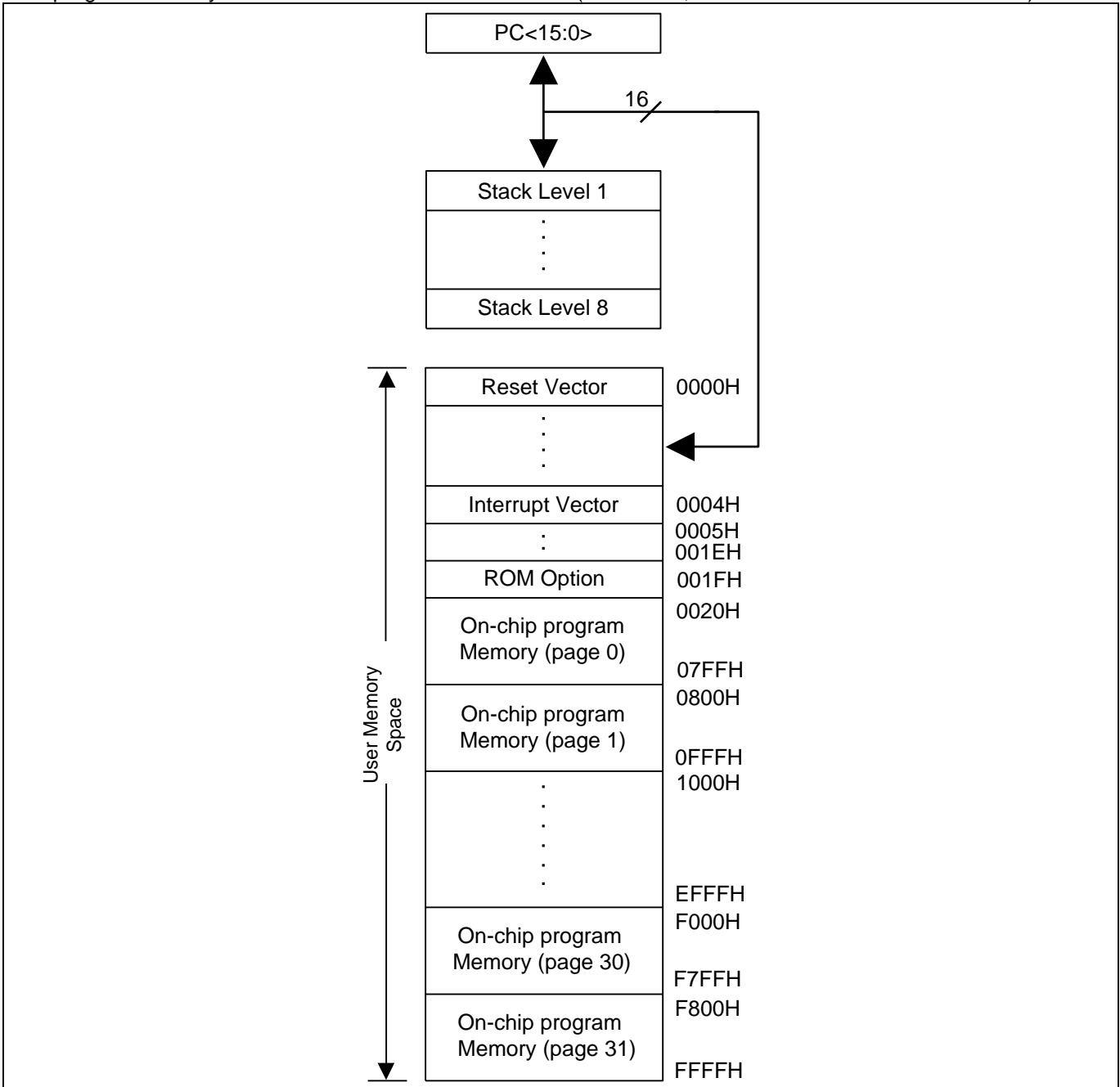


Figure 2-1. Program Memory Map and Stack

2.1.1. ROM OPTION

The ROM Option is start-condition of the chip. The ROM Option address is 001FH. The default value of the ROM Option is 3FFFH (LVR disable, selected crystal/ceramic oscillator).

For example, if you program as below:

```
CODE 0x1F      ;ROM option area
DW    0x3FBE   ;Disable LVR, Select Crystal/Ceramic OSC
```

ROM OPTION

ROM Address: 001FH

	.13	.12	.11	.10	.9	.8	.7	.6	.5	.4	.3	.2	.1	.0	
MSB	1	1	1	1	1	LVR EN	LVRS			OSCS			1	LSB	

! To avoid malfunction, You must set undefined bits.

Bit13-9	Not available for the MC71PB204	
LVREN	LVR Enable/Disable Selection Bit	0: Enable 1: Disable
LVRS	LVR Level Selection Bits	1100: 2.6 V 0111: 3.0 V 0100: 4.0 V
OSCS	Oscillator Selection Bits	000: External RC 001: Internal RC; 4MHz 010: Internal RC; 2MHz 011: Internal RC; 1MHz 100: Internal RC; 8MHz 111: Crystal/Ceramic oscillator
Bit0	Not available for the MC71PB204	

NOTES:

1. The value of unused bits <0> and <9:13> is don't care.
2. When LVR is enabled, LVR level must be set to appropriate value, not default value.

2.1.2 PC AND PCLATH REGISTERS

The Program Counter (PC) is 16-bit wide. The lower bits (PC<7:0>) come from PCL register, which is a readable and writable register. The upper bits (PC<15:8>) are not directly readable (or writable), but are indirectly writable through the PCLATH register. On any reset, (the upper bits of) the PC is (will be) cleared.

Figure 2-2 shows the two situations for PC loading. The upper example in the figure 2-2 shows how the PC is loaded by writing to PCL (PCLATH<7:0> → PCH). The lower example in the figure 2-2 shows how the PC is loaded during a CALL or GOTO instruction (PCLATH<7:3> → PCH).

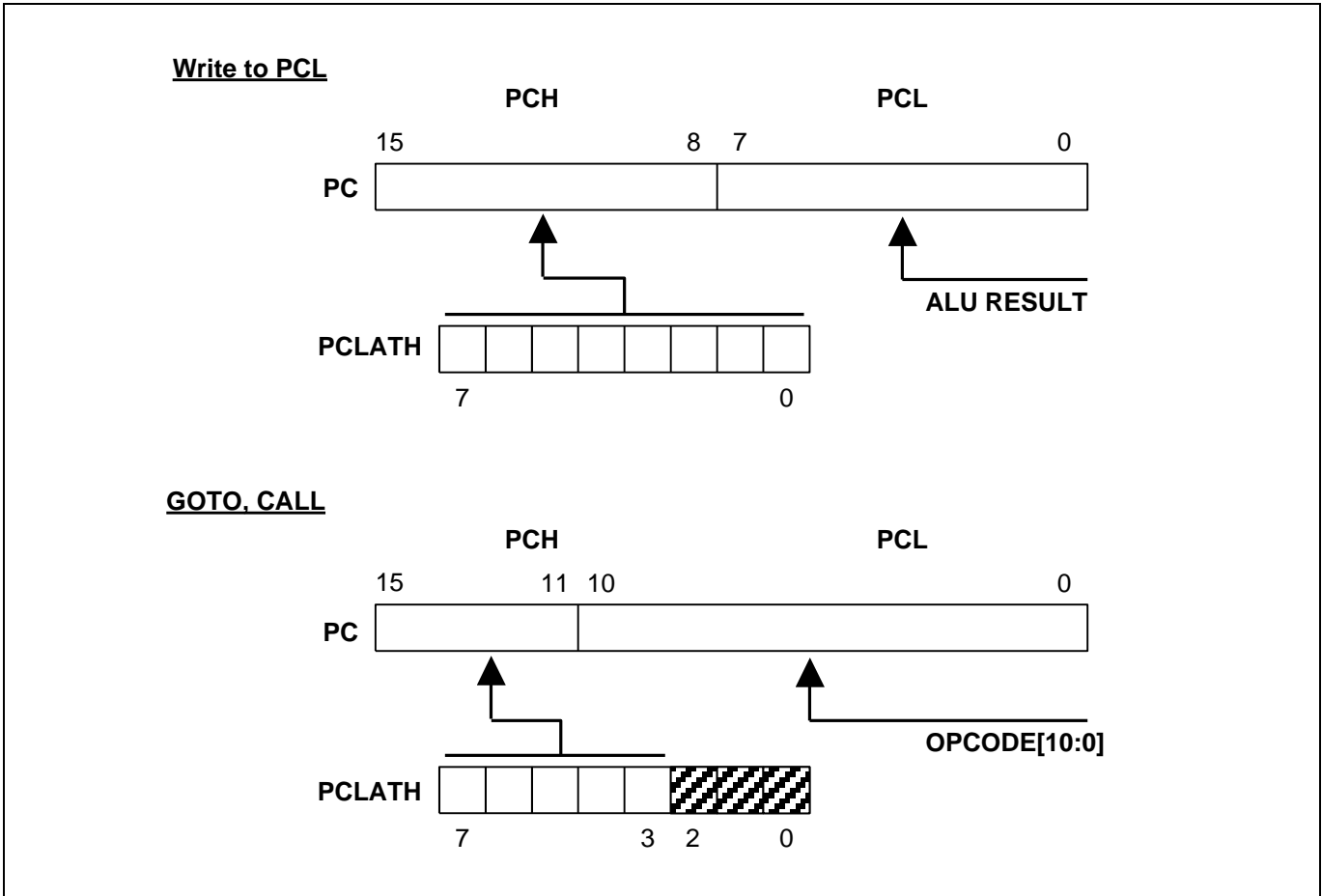


Figure 2-2. Loading of PC in different situations

A computed GOTO is accomplished by adding an offset to the program counter (ADDWF PCL). When reading a table using a computed GOTO method, pay attention if the table location crosses a PCL memory boundary (each 256 byte block).

2.1.3 PROGRAM MEMORY PAGING

The GMC14 series devices are capable of addressing a continuous 64k words block of program memory. The CALL and GOTO instructions provide only 11 bits of address to allow branching within any 2k words program memory page. When doing a CALL or GOTO instruction the upper 5 bits of the address are provided by PCLATH<7:3>. When doing a CALL or GOTO instruction, the user must ensure that the page select bits are programmed so that the desired program memory page is addressed. If RETURN, RETLW, or RETFIE instructions are executed, the entire 16-bit PC is popped from the stack. Therefore, manipulation of the PCLATH<7:3> bits is not required for the return instructions (which pops the address from the stack).

NOTE: Because the MC71PB204 use only PCLATH<3:0> bit, the PCLATH<7:4> bits should be always logic '0000b'.

2.2 STACK

The GMC14 series has an 8 level depth x 16-bit width hardware stack. The stack space is neither part of program nor data space and the stack pointer is not readable or writable. The PC is pushed onto the stack when a CALL instruction is executed or an interrupt causes a branch. The stack is popped in the event of a RETURN, RETLW or a RETFIE instruction execution. PCLATH is not affected by a push or pop operation. The stack operates as a circular buffer. This means that after the stack has been pushed eight times, the ninth push overwrites the value that stored from the first push. The tenth push overwrites the second push (and so on).

NOTES:

1. There are no STATUS bits to indicate stack overflow or stack underflow conditions.
2. There are no instructions/mnemonics called push or pop. These are actions that occur from the execution of the CALL, RETURN, RETLW, and RETFIE instructions, or the vectoring to an interrupt address.

2.3 DATA MEMORY

The data memory for the MC71PB204 is partitioned onto three banks which contain the general purpose registers and the special function registers. Bank 0 is selected when the RP1:RP0 bits in status register are 00b, bank 1 when RP1:RP0 are 01b, bank2 when RP1:RP0 are 10b.

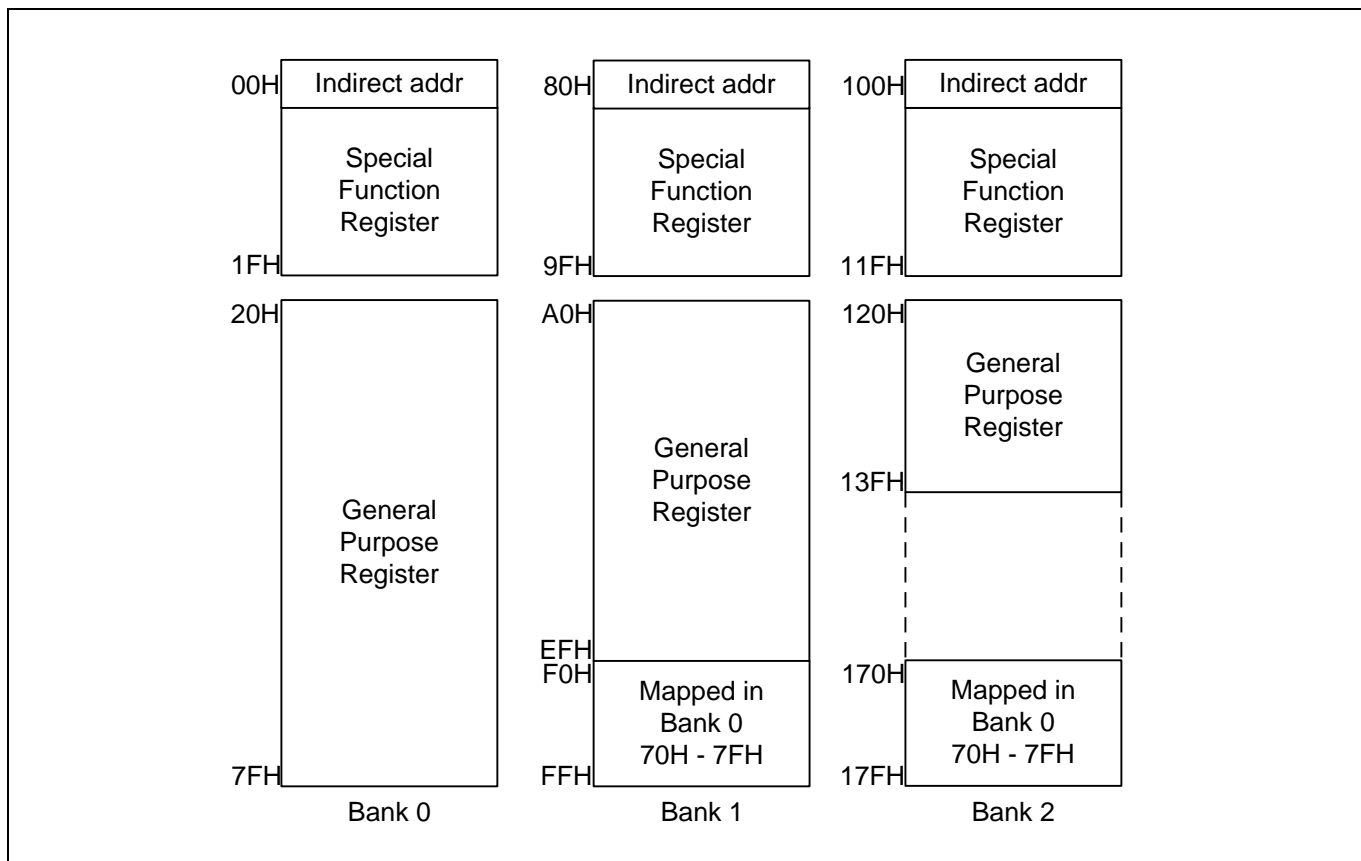


Figure 2-3. Register File Organization (MC71PB204)

The lower locations of each bank are reserved for the special function registers. The upper locations of each bank are general purpose registers implemented as static RAM. All three banks contain special function registers. Some of the special function registers are mirrored in other banks for code reduction and quicker access.

The register file can be accessed either directly, or indirectly through the File Select Register (FSR). Refer to 2.3.3 addressing mode.

2.3.1 GENERAL PURPOSE REGISTER

The size of the MC71PB204's general purpose register is 208 bytes (20H~7FH, A0H~EFH and 120H~13FH). The following general purpose registers are not physically implemented:

- 0F0H-0FFH of Bank 1
- 170H-17FH of Bank 2

These locations are used for common access across banks.

2.3.2 SPECIAL FUNCTION REGISTER

There are 33 bytes of special function register. Some of the special function registers are visible on any of three memory banks. As is shown in figure below;

00H	CONFIG	CONFIG	CONFIG
01H	CPUCLK	-	-
02H	PCL	PCL	PCL
03H	STATUS	STATUS	STATUS
04H	FSR	FSR	FSR
05H	ADMR	PWMSCR	-
06H	ADRRH	PWMCR	-
07H	ADRRL	PWMDR	-
08H	WTSCR	-	-
09H	WTCR	-	-
0AH	PCLATH	PCLATH	PCLATH
0BH	INTCON	-	-
0CH	IPND	-	-
0DH	TSCRA	-	-
0EH	TCRA	-	-
0FH	TDRA	-	-
10H	TSCRB	-	-
11H	TCRB	-	-
12H	TDRB	-	-
13H	-	-	-
14H	-	-	-
15H	DDR0H	-	-
16H	DDR0L	-	-
17H	PUR0	-	-
18H	DDR1	-	-
19H	DDR2H	-	-
1AH	DDR2L	-	-
1BH	EINT2	-	-
1CH	EPND1	-	-
1DH	P0	-	-
1EH	P1	-	-
1FH	P2	-	-
20H	RAM Memory	RAM Memory	RAM Memory
7FH	Space	Space	Space
	Bank 0	Bank 1	Bank 2

Figure 2-4. Data memory map

The special function registers are the registers used by the cpu and peripheral modules for controlling the desired operation of the device. These registers are implemented as static RAM. The special function registers can be classified into two sets (core and peripheral). The special function registers associated with the “core” functions are described below. The rest of special function registers are described in the corresponding peripheral section.

2.3.2.1 CONFIG Register

The CONFIG register contains configuration bits, which defines additional the MC71PB204 core features.

To change it's contents, the FSR register should be cleared first, and then a particular instruction using indirect addressing mode should be executed.

CONFIG REGISTER (CONFIG)

00H, 80H, 100H

	7	6	5	4	3	2	1	0	
CONFIG	-	-	-	-	-	-	DEC	INC	(Initial value : - - - - -00)
Read/Write	-	-	-	-	-	-	R/W	R/W	

Bit7-2	Not available for the MC71PB204	
DEC	Indirect addressing mode with post decrement FSR contents	0: Post FSR decrement disabled 1: Post FSR decrement enabled
INC	Indirect addressing mode with post increment FSR contents	0: Post FSR increment disabled 1: Post FSR increment enabled

NOTE: Both DEC and INC bits set disable auto increment/decrement function.

2.3.2.2 STATUS Register

The STATUS register contains the arithmetic status of the ALU and the bank selection bits for data memory. The STATUS register can be the destination for any instruction, as with any other register. If the STATUS register is the destination for an instruction that affects the Z, DC or C bits, then writing to these three bits is disabled. These bits are set or cleared according to the device logic. Furthermore, the nTO and nPD bits are not writable. Therefore, the result of an instruction with the STATUS register as destination may be different from intended.

STATUS REGISTER (STATUS)

03H, 83H, 103H

	7	6	5	4	3	2	1	0	
STATUS	IRP	RP1	RP0	nTO	nPD	Z	DC	C	(Initial value : NOTE)
Read/Write	R/W	R/W	R/W	R	R	R/W	R/W	R/W	

IRP	Register Bank Selection Bit (used for indirect addressing)	0: Bank 0, 1 (00H – 0FFH) 1: Bank 2, 3 (100H – 1FFH)
RP1 RP0	Register Bank Selection Bits (used for direct addressing)	00: Bank 0 01: Bank 1 10: Bank 2 01: Bank 3; (Not used for the MC71PB204)
nTO	Time-out Bit	0: A WDT time-out occurred 1: After power-up, CLRWDT instruction, or SLEEP instruction.
nPD	Power-down Bit	0: By execution of the SLEEP instruction 1: After power-up or by the CLRWDT instruction
Z	Zero Bit	0: The result of an arithmetic or logic operation is not zero 1: The result of an arithmetic or logic operation in zero
DC	Digit Carry/nBorrow Bit	0: No carry-out from the 4 th low order bit of the result occurred 1: A carry-out from the 4 th low order bit of the result occurred
C	Carry/nBorrow Bit	0: No carry-out from the most significant bit of the result occurred 1: A carry-out from the most significant bit of the result occurred

NOTE: Refer to the Table 6-4 in the Chapter 6. RESET AND POWER DOWN for the initial value of STATUS register.

2.3.2.3 Internal Interrupt Control Register (INTCON)

The INTCON register is able to select enable or disable global interrupt, watchdog timer interrupt, PWM interrupt, timer A interrupt, and timer B interrupt.

INTERNAL INTERRUPT CONTROL REGISTER (INTCON)

0BH

	7	6	5	4	3	2	1	0	
INTCON	GIE	WTIE	PWMIE	TAIE	TBIE	-	-	-	(Initial value : 0000 0 - - -)
Read/Write	R/W	R/W	R/W	R/W	R/W	-	-	-	

GIE	Global Interrupt Enable Bit	0: Disable all interrupt 1: Enable all un-masked interrupts
WTIE	Watchdog Timer Interrupt Enable Bit	0: Disable interrupt 1: Enable interrupt
PWMIE	PWM Interrupt Enable Bit	0: Disable interrupt 1: Enable interrupt
TAIE	Timer 0/A Interrupt Enable Bit	0: Disable interrupt 1: Enable interrupt
TBIE	Timer B Interrupt Enable Bit	0: Disable interrupt 1: Enable interrupt
Bit2-0	Not available for the MC71PB204	

2.3.2.4 Internal Interrupt Pending Register (IPND)

The IPND register is a readable and writable register, which contains various pending bits for internal interrupt.

INTERNAL INTERRUPT PENDING REGISTER (IPND)

0CH

	7	6	5	4	3	2	1	0	
IPND	-	WT PND	PWM PND	TA PND	TB PND	-	-	-	(Initial value : - 000 0 - - -)
Read/Write	-	R/W	R/W	R/W	R/W	-	-	-	

Bit7	Not available for the MC71PB204	
WTPND	Watchdog Timer Interrupt Pending Bit	0: interrupt request is not pending (when read); pending bit clear when write 0 1: interrupt request is pending (when read)
PWMPND	PWM Interrupt Pending Bit	0: interrupt request is not pending (when read); pending bit clear when write 0 1: interrupt request is pending (when read)
TAPND	Timer 0/A Interrupt Pending Bit	0: interrupt request is not pending (when read); pending bit clear when write 0 1: interrupt request is pending (when read)
TBPND	Timer B Interrupt Pending Bit	0: interrupt request is not pending (when read); pending bit clear when write 0 1: interrupt request is pending (when read)
Bit2-0	Not available for the MC71PB204	

2.3.2.5 Special Function Register's Map

Table 2-1. BANK0 Register's Map

Register Name	Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Hex								
CONFIG	00H	–	–	–	–	–	–	DEC	INC
CPUCLK	01H	IDLE		–	–	–	–	CCLK	
PCL	02H	Program counter (PC) Least Significant Byte							
STATUS	03H	IRP	RP1	RP0	nTO	nPD	Z	DC	C
FSR	04H	Indirect data memory address pointer							
ADMR	05H	SSBIT	EOC	ADCLK		ADCH			
ADDRH	06H	A/D Converter Data Register, High Byte							
ADDRL	07H	A/D Converter Data Register, Low Byte							
WTSCR	08H	WTFUN				WT3C	WTCS		WTCC
WTCR	09H	8-Bit Watchdog Timer Counter Register							
PCLATH	0AH	–	–	–	Write buffer upper 5 bits of PC				
INTCON	0BH	GIE	WTIE	PWMIE	TAIE	TBIE	–	–	–
IPND	0CH	–	WT PND	PWM PND	TA PND	TB PND	–	–	–
TSCRA	0DH	–	–	TARL	TACE	TACS		T0MOD	
TCRA	0EH	Timer 0/A Counter Register							
TDRA	0FH	Timer 0/A Data Register							
TSCRB	10H	–	–	TBRL	TBCE	TBCS		–	
TCRB	11H	Timer B Counter Register							
TDRB	12H	Timer B Data Register							
13H -14H are not mapped									
DDR0H	15H	P05		P05PD	–	P04		P04PD	–
DDR0L	16H	P03		P02		P01		P00	
PUR0	17H	–	PUR06	PUR05	PUR04	PUR03	PUR02	PUR01	PUR00
DDR1	18H	P12		P11			P10		
DDR2H	19H	P27		P26		P25		P24	
DDR2L	1AH	P23		P22		P21		P20	
EINT2	1BH	INT3		INT2		INT1		INT0	
EPND2	1CH	–	–	–	–	PND3	PND2	PND1	PND0
P0	1DH	Port 0 Data Register							
P1	1EH	Port 1 Data Register							
P2	1FH	Port 2 Data Register							

NOTE: A dash (‘–’) means that the bit is neither used nor mapped.

Table 2-2. BANK1 Register's Map

Register Name	Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Hex								
CONFIG	80H	-	-	-	-	-	-	DEC	INC
81H is not mapped									
PCL	82H	Program counter (PC) Least Significant Byte							
STATUS	83H	IRP	RP1	RP0	nTO	nPD	Z	DC	C
FSR	84H	Indirect data memory address pointer							
PWMSCR	85H	PWMICS		-	-	PWMD	PWMC	PWME	-
PWMCR	86H	PWM Counter Register							
PWMDR	87H	PWM Data Register							
88H - 89H are not mapped									
PCLATH	8AH	-	-	-	With buffer upper 5 bits of PC				
8BH - 9FH are not mapped									

NOTE: A dash ('-') means that the bit is neither used nor mapped.

Table 2-3. BANK2 Register's Map

Register Name	Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Hex								
CONFIG	100H	-	-	-	-	-	-	DEC	INC
101H is not mapped									
PCL	102H	Program counter (PC) Least Significant Byte							
STATUS	103H	IRP	RP1	RP0	nTO	nPD	Z	DC	C
FSR	104H	Indirect data memory address pointer							
105H - 109H are not mapped									
PCLATH	10AH	-	-	-	With buffer upper 5 bits of PC				
10BH - 11FH are not mapped									

NOTE: A dash ('-') means that the bit is neither used nor mapped.

2.3.3 ADDRESSING MODES

The MC71PB204 supports two addressing modes: direct or indirect. In Direct Addressing, the 9-bit direct address is concatenated from RP [1:0] bits of STATUS (03H) register and a 7LSB of instruction word. Indirect addressing is possible by using the CONFIG (00H) register. Any instruction using CONFIG (00H) register actually accesses data pointed by the File Select Register (FSR (04H)). The 9-bit address is concatenated from IRP bit from STATUS (03H) register and 8 bits of FSR (04H) register. Both Direct and indirect addressing modes are shown in figure below.

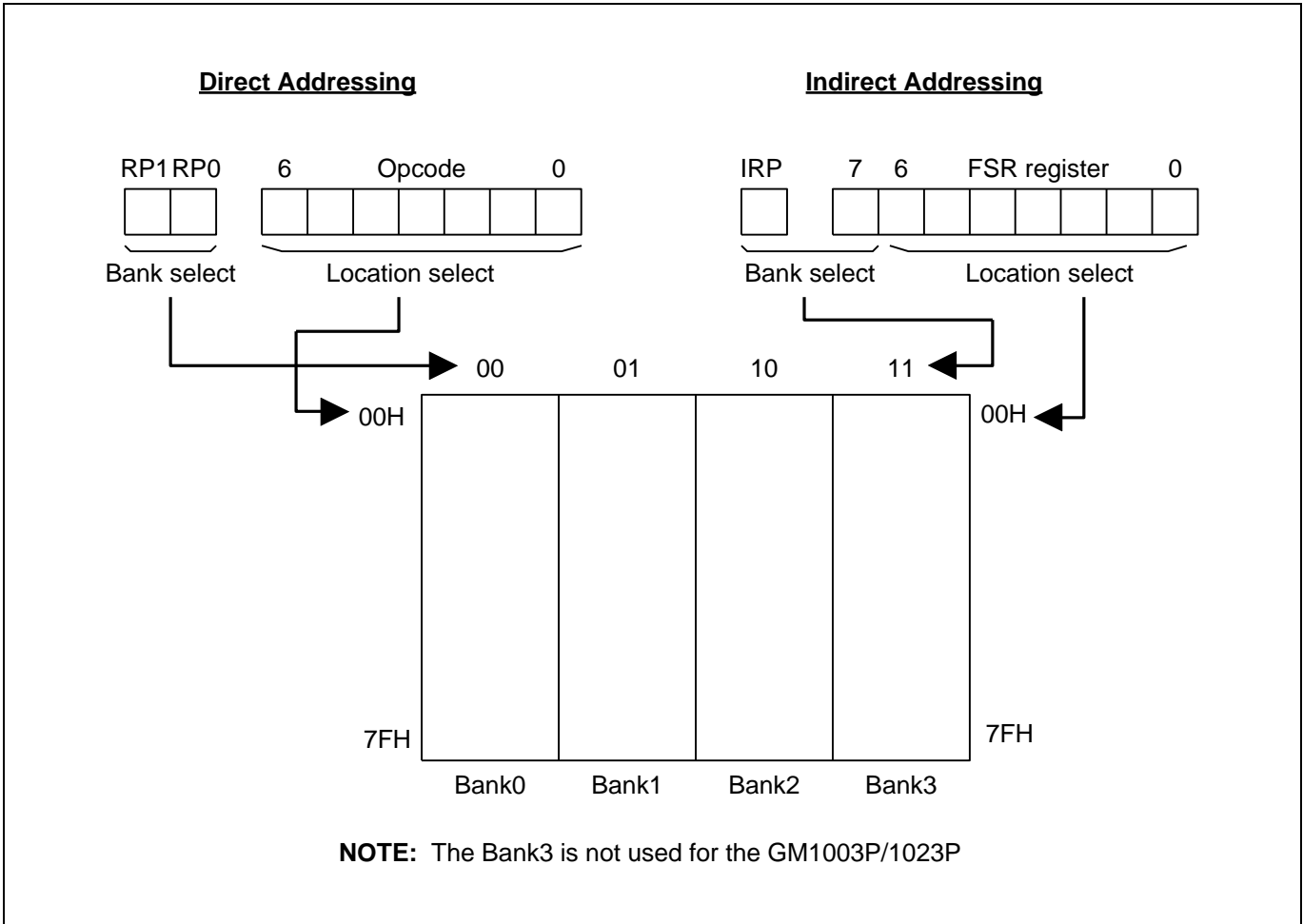


Figure 2-5. Direct/Indirect addressing

2.3.3.1. Indirect Addressing Mode

Indirect addressing mode is applied when the instruction point directly to the CONFIG (00H) register. Any instruction pointing directly the CONFIG (00H) register as a source/destination, actually accesses data pointed by the FSR (file select register, 04H).

In indirect addressing mode, user can select three of supported indirect addressing mode:

- Simple indirect: the indirect address comes from concatenated IRP and FSR.
- Indirect with post increment: the content of FSR register is post incremented, after execution of any instruction using indirect addressing mode.
- Indirect with post decrement: the content of FSR register is post decremented, after execution of any instruction using indirect addressing mode.

A simple program to clear RAM locations 20H-2FH using indirect addressing is shown in Example 2-1.

EXAMPLE 2-1: INDIRECT ADDRESSING

```

                MOVLW    0x20        ;initialize pointer
                MOVWF    FSR         ;to RAM
NEXT:          CLRf     CONFIG      ;clear CONFIG register
                INCF     FSR,F       ;inc pointer
                BTFSS   FSR,4       ;all done?
                GOTO    NEXT        ;no clear next
CONTINUE:     :                   ;yes continue
    
```

3. INTERRUPT

3.1 INTERRUPT STRUCTURE

The MC71PB204 has eight interrupt sources:

- Four external interrupts (Port 2.0~Port 2.3)
- Watchdog timer interrupt
- PWM interrupt
- Timer 0/A underflow interrupt
- Timer B underflow interrupt

The interrupt vector address is located at 0004H of ROM address area. Please be careful not to overwrite any of the stored vector addresses.

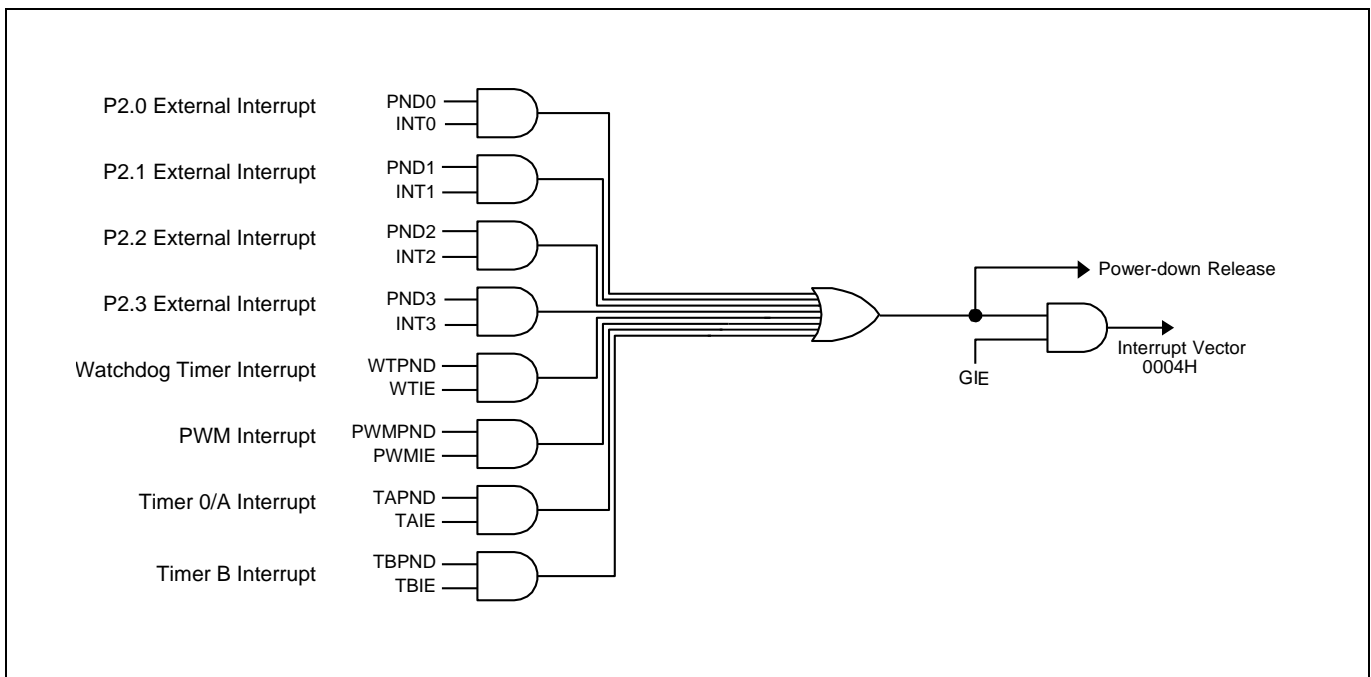


Figure 3-1. Interrupt Structure

The interrupt control registers, INTCON and EINT2 have enable bit of individual interrupt, and INTCON has global interrupt enable bit. The interrupt pending registers, IPND and EPND2 record individual interrupt requests in corresponding bits.

A global interrupt enable bit, GIE (INTCON<7>) enables (if set) all un-masked interrupts or disables (if cleared) all interrupts. When GIE bit is set, and an interrupt's pending bit and interrupt enable bit are set, the interrupt will vector immediately. Individual interrupts can be disabled through their corresponding enable bits in the INTCON or EINT2.

The watchdog timer interrupt, PWM interrupt, timer 0/A underflow interrupt, and timer B underflow interrupt control bits are contained in the INTCON register and their corresponding interrupt pending bits are in the IPND register.

Four external interrupts are contained in the EINT2 register and their corresponding interrupt pending bits are in the EPND2 register.

When an interrupt is responded to, the GIE bit is cleared to disable any further interrupts, the return address is pushed onto the stack and the PC is loaded with vector address 0004H. At the interrupt service routine, the source(s) of the interrupt can be determined by polling the interrupt pending bits. The interrupt pending bit(s) must be cleared in software before re-enabling interrupts to avoid recursive interrupts.

3.2 SAVING KEY REGISTERS DURING AN INTERRUPT SERVICE

During an interrupt, only the return PC value is saved on the stack. Typically, users may wish to save key registers like W register or STATUS register during an interrupt. This will have to be implemented in software. Ex 3-1) shows how to store and restore the STATUS, W, and PCLATH registers. The register, W_TEMP, must be defined in each bank at the same offset from the bank base address.

The example:

- 1) Stores the W register.
- 2) Stores the STATUS register in bank 0.
- 3) Stores the PCLATH register.
- 4) Executes the Interrupt Service Routine code.
- 5) Restores the PCLATH register.
- 6) Restores the STATUS and W registers.

EX 3-1) SAVING STATUS, W, AND PCLATH REGISTERS IN RAM

```

MOVWF  W_TEMP          ;Copy W to TEMP register, could be bank one or zero
SWAPF  STATUS, W       ;Swap status to be saved into W
CLRF   STATUS           ;bank 0, regardless of current bank, Clears IRP, RP1, RP0
MOVWF  STATUS_TEMP     ;Save status to bank zero STATUS_TEMP register
MOVF   PCLATH, W       ;Only required if using pages 1, 2 and/or 3
MOVWF  PCLATH_TEMP     ;Save PCLATH into W
CLRF   PCLATH          ;Page zero, regardless of current page
:
:(Interrupt Service Routine)
:
MOVF   PCLATH_TEMP, W  ;Restore PCLATH
MOVWF  PCLATH          ;Move W into PCLATH
SWAPF  STATUS_TEMP, W ;Swap STATUS_TEMP register into W
                          ;(sets bank to original state)
MOVWF  STATUS          ;Move W into STATUS register
SWAPF  W_TEMP, F       ;Swap W_TEMP
SWAPF  W_TEMP, W       ;Swap W_TEMP into W

```

NOTES

4. INSTRUCTION SET

Each MC71PB204 instruction has 14-bit word length divided into an OPCODE, which specifies the instruction type and operands. The instruction set is grouped into the three basic categories:

- Byte-oriented operations
- Bit-oriented operations
- Literal and control operations

Figure below shows three general formats that the instruction can have.

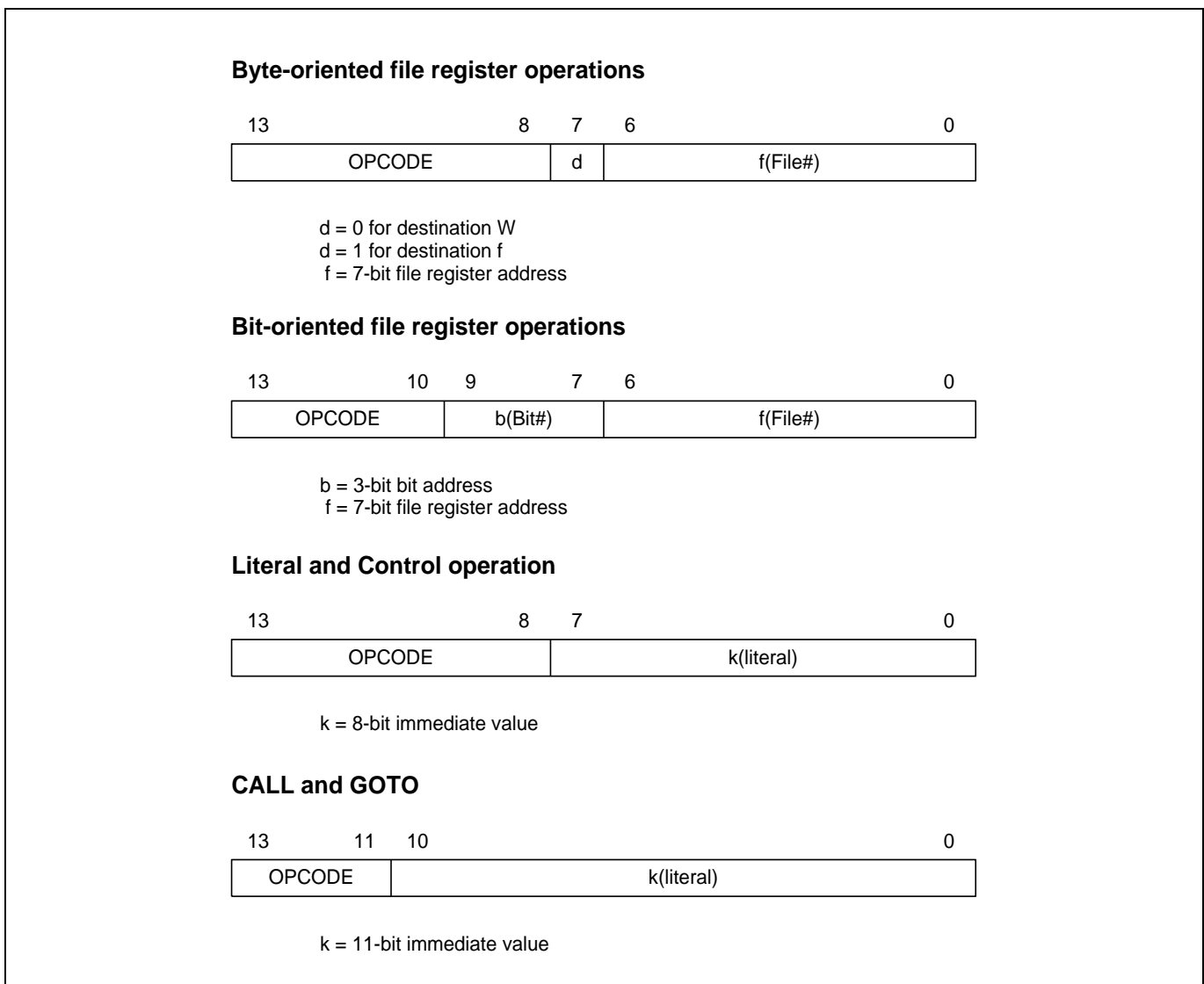


Figure 4-1. General Format of Instructions

All instructions are executed within 2 Clock cycles. Except the instructions using indirect addressing mode which are executed within 4 CLK periods (two instruction cycles).

Table 4-1. Opcode Filed Descriptions

Field	Description
f	Register file address (0x00 to 0x7F)
W	Working register (accumulator)
b	Bit address within an 8-bit file register
k	Literal field, constant data or label
x	Don't care location (=0 or 1)
d	Destination selected; d = 0: store result in W, d = 1: store result in file register f. Default is d = 1
label	Label name
TOS	Top of Stack
PC	Program Counter
PCLATH	Program Counter High Latch
GIE	Global Interrupt Enable bit
WDTCNT	Watchdog Timer Counter
nTO	Time-out bit
nPD	Power-down bit
destination	Destination either the W register or the specified register file location

4.1 INSTRUCTION SET SUMMARY

4.1.1 BYTE-ORIENTED INSTRUCTION

Table 4-2. Byte-Oriented Operations

Mnemonic, operands	Description	14-bit opcode				Status	Cycles	
		MSB		LSB				
ADDWF	f, d	Add W and f	00	0111	d f f f	f f f f	C, DC, Z	2
	d	Add W and (FSR)	00	0111	d000	0000	C, DC, Z	4
ANDWF	f, d	AND W and f	00	0101	d f f f	f f f f	Z	2
	d	AND W and (FSR)	00	0101	d000	0000	Z	4
CLRF	f	Clear f	00	0001	1 f f f	f f f f	Z	2
		Clear (FSR)	00	0001	1000	0000	Z	4
CLRWF	f, d	Add W and f	00	0001	0 x x x	x x x x	Z	2
COMF	f, d	Complement f	00	1001	d f f f	f f f f	Z	2
	d	Complement (FSR)	00	1001	d000	0000	Z	4
DECF	f, d	Decrement f	00	0011	d f f f	f f f f	Z	2
	d	Decrement (FSR)	00	0011	d000	0000	Z	4
DECFSZ	f, d	Decrement f, Skip if 0	00	1011	d f f f	f f f f	-	2
	d	Decrement (FSR), Skip if 0	00	1011	d000	0000	-	4
INCF	f, d	Increment f	00	1010	d f f f	f f f f	Z	2
	d	Increment (FSR)	00	1010	d000	0000	Z	4
INCFSZ	f, d	Increment f, Skip if 0	00	1111	d f f f	f f f f	-	2
	d	Increment (FSR), Skip if 0	00	1111	d000	0000	-	4
IORWF	f, d	Inclusive OR W with f	00	0100	d f f f	f f f f	Z	2
	d	Inclusive OR W with (FSR)	00	0100	d000	0000	Z	4
MOVF	f, d	Move f	00	1000	d f f f	f f f f	Z	2
	d	Move (FSR)	00	1000	d000	0000	Z	4
MOVWF	f, d	Move W to f	00	0000	1 f f f	f f f f	-	2
		Move W to (FSR)	00	0000	1000	0000	-	4
NOP		No Operation	00	0000	0 x x 0	0000	-	2
RLF	f, d	Rotate Left f through Carry	00	1101	d f f f	f f f f	C	2
	d	Rotate Left (FSR) through Carry	00	1101	d000	0000	C	4
RRF	f, d	Rotate Right f through Carry	00	1100	d f f f	f f f f	C	2
	d	Rotate Right (FSR) through Carry	00	1100	d000	0000	C	4
SUBWF	f, d	Subtract W from f	00	0010	d f f f	f f f f	C, DC, Z	2
	d	Subtract W from (FSR)	00	0010	d000	0000	C, DC, Z	2
SWAPF	f, d	Swap nibbles in f	00	1110	d f f f	f f f f	-	2
	d	Swap nibbles in (FSR)	00	1110	d000	0000	-	2
XORWF	f, d	Exclusive OR W with f	00	0110	d f f f	f f f f	Z	2
	d	Exclusive OR W with (FSR)	00	0110	d000	0000	Z	4

4.1.2 BIT-ORIENTED INSTRUCTION

Table 4-3. Bit-Oriented Operations

Mnemonic, operands	Description	14-bit opcode				Status	Cycles	
		MSB		LSB				
BCF	f, b	Bit Clear f	0 1	00 b b	b f f f	f f f f	-	2
	b	Bit Clear (FSR)	0 1	00 b b	b 0 0 0	0 0 0 0	-	4
BSF	f, b	Bit Set f	0 1	01 b b	b f f f	f f f f	-	2
	b	Bit Set (FSR)	0 1	01 b b	b 0 0 0	0 0 0 0	-	4
BTFSC	f, b	Bit Test f, Skip if Clear	0 1	10 b b	b f f f	f f f f	-	2
	b	Bit Test (FSR), Skip if Clear	0 1	10 b b	b 0 0 0	0 0 0 0	-	4
BTFSS	f, b	Bit Test f, Skip if Set	0 1	11 b b	b f f f	f f f f	-	2
	b	Bit Test (FSR), Skip if Set	0 1	11 b b	b 0 0 0	0 0 0 0	-	2

4.1.3 LITERAL AND CONTROL OPERATIONS

Table 4-4. Literal and Control Operations

Mnemonic, operands	Description	14-bit opcode				Status	Cycles	
		MSB		LSB				
ADDLW	imm	Add literal and f	1 1	1 1 1 x	k k k k	k k k k	C, DC, Z	2
ANDLW	imm	Add literal and f	1 1	1 0 0 1	k k k k	k k k k	Z	2
CALL	imm	Call subroutine	1 0	0 k k k	k k k k	k k k k	-	4
CLRWDT	-	Clear Watchdog Timer	0 0	0 0 0 0	0 1 1 0	0 1 0 0	nTO, nPD	2
GOTO	imm	Go to address	1 0	1 k k k	k k k k	k k k k	-	4
IORLW	imm	Inclusive OR literal with W	1 1	1 0 0 0	k k k k	k k k k	Z	2
MOVLW	imm	Move literal to W	1 1	0 0 x x	k k k k	k k k k	-	2
RETFIE	-	Return from Interrupt	0 0	0 0 0 0	0 0 0 0	1 0 0 1	-	4
RETLW	imm	Return with literal in W	1 1	0 1 x x	k k k k	k k k k	-	4
RETURN	-	Return from subroutine	0 0	0 0 0 0	0 0 0 0	1 0 0 0	-	4
SLEEP	-	Go into standby mode	0 0	0 0 0 0	0 1 1 0	0 0 1 1	nTO, nPD	2
SUBLW	imm	Subtract W from literal	1 1	1 1 0 x	k k k k	k k k k	C, DC, Z	2
XORLW	imm	Exclusive OR literal with W	1 1	1 0 1 0	k k k k	k k k k	Z	2

4.2 INSTRUCTION DESCRIPTION

4.2.1 ADDLW – ADD LITERAL AND W

Operands: $0 \leq \text{imm} (k) \leq 255$

Operation: $W \leftarrow W + \text{imm} (k)$

Status Affected: C, DC, Z

Description: The contents of W register are added to the eight bit immediate data 'imm' and the result is placed in the W register.

Encoding:

1 1	1 1 1 x	k k k k	k k k k
-----	---------	---------	---------

Cycles: DIR : 2

Example: ADDLW 0x15

Before Instruction

W = 0x10

After Instruction

W = 0x25

4.2.2 ADDWF – ADD W AND F

Operands: $0 \leq f \leq 127$
 $d \in [0, 1]$

Operation: destination $\leftarrow W + f$
 destination $\leftarrow W + FSR$

Status Affected: C, DC, Z

Description: ADDWF instruction add contents of the W register with register specified by 'f' operand. If 'd' is 0 the result is stored in the W register. If 'd' is 1 the result is stored back in register 'f'.

Encoding:

0 0	0 1 1 1	d f f f	f f f f
0 0	0 1 1 1	d 0 0 0	0 0 0 0

Cycles: DIR : 2
 INDIR : 4

Example: ADDWF FSR, 0

Before Instruction

W = 0x17
 FSR = 0xC2

After Instruction

W = 0xD9
 FSR = 0XC2

4.2.3 ANDLW – AND LITERAL WITH W

Operands: $0 \leq \text{imm} (k) \leq 255$

Operation: $W \leftarrow W \text{ and } \text{imm} (k)$

Status Affected: Z

Description: The contents of W register are AND'ed with the eight bit immediate data 'imm'. The result is placed in the W

Encoding:

1 1	1 0 0 1	k k k k	k k k k
-----	---------	---------	---------

Cycles: DIR : 2

Example: ANDLW 0x5F

Before Instruction

W = **0xA3**

After Instruction

W = **0x03**

4.2.4 ANDWF – AND W WITH F

Operands: $0 \leq f \leq 127$
 $d \in [0, 1]$

Operation: destination \leftarrow W and f
 destination \leftarrow W and FSR

Status Affected: Z

Description: ANDWF instruction AND the W register with register specified by 'f' operand. The result is stored in W or back in 'f' register respectively to the 'd' value.

Encoding:

0 0	0 1 0 1	d f f f	f f f f
0 0	0 1 0 1	d 0 0 0	0 0 0 0

Cycles: DIR : 2
 INDIR : 4

Example: ANDWF FSR, 1

Before Instruction

W = 0x17
 FSR = 0xC2

After Instruction

W = 0X17
 FSR = 0x02

4.2.5 BCF – BIT CLEAR F

Operands: $0 \leq f \leq 127$
 $0 \leq b \leq 7$

Operation: $f.b \leq 0$
 $FSR.b \leq 0$

Status Affected: -

Description: Bit 'b' in register 'f' is cleared.

Encoding:

0 1	0 0 b b	b f f f	f f f f
-----	---------	---------	---------

0 1	0 0 b b	b 0 0 0	0 0 0 0
-----	---------	---------	---------

Cycles: DIR : 2
 INDIR : 4

Example: BCF Flag_Buf, 1

Before Instruction

Flag_Buf = 0x17

After Instruction

Flag_Buf = 0x16

4.2.6 BSF – BIT SET F

Operands: $0 \leq f \leq 127$
 $0 \leq b \leq 7$

Operation: $f.b \leq 1$
 $FSR.b \leq 1$

Status Affected: -

Description: Bit 'b' in register 'f' is cleared.

Encoding:

0 1	0 1 b b	b f f f	f f f f
0 1	0 1 b b	b 0 0 0	0 0 0 0

Cycles: DIR : 2
 INDIR : 4

Example: BSF Flag_Buf, 7

Before Instruction

Flag_Buf = 0x17

After Instruction

Flag_Buf = 0x97

4.2.7 BTFSC – BIT TEST, SKIP IF CLEAR

Operands: $0 \leq f \leq 127$
 $0 \leq b \leq 7$

Operation: skip if, f.b = 0
 skip if, FSR.b = 0

Status Affected: -

Description: Check the 'b' bit in 'f' register and skip next instruction when 'b' is '0'. If bit 'b' is '0' then the next instruction fetched during the current instruction execution is discarded, and a NOP is executed instead, making this a two-cycle instruction.

Encoding:

0 1	1 0 b b	b f f f	f f f f
-----	---------	---------	---------

0 1	1 0 b b	b 0 0 0	0 0 0 0
-----	---------	---------	---------

Cycles: DIR : 2
 INDIR : 4

Example:
 aaa: BTFSC Flag_Buf, 1
 bbb: GOTO ccc
 ddd:

Before Instruction

PC = address aaa

After Instruction

If Flag_Buf <1> = 0,
 PC = address ddd

If Flag_Buf <1> = 1,
 PC = address bbb

4.2.8 BTFSS – BIT TEST, SKIP IF SET

Operands: $0 \leq f \leq 127$
 $0 \leq b \leq 7$

Operation: skip if f.b = 1
 skip if FSR.b = 1

Status Affected: -

Description: If bit 'b' in register 'f' is '1' then the next instruction is skipped. If bit 'b' is '0' then the next instruction fetched during the current instruction execution is discarded, and a NOP is executed instead, making this a two-cycle instruction.

Encoding:

0 1	1 1 b b	b f f f	f f f f
0 1	1 1 b b	b 0 0 0	0 0 0 0

Cycles: DIR : 2
 INDIR : 4

Example: aaa: BTFSC Flag_Buf, 1
 bbb: GOTO ccc
 ddd:

Before Instruction

PC = address aaa

After Instruction

If Flag_Buf<1> = 0,
 PC = address bbb

If Flag_Buf<1> = 1,
 PC = address ddd

4.2.9 CALL – CALL SUBROUTINE

Operands: $0 \leq \text{imm} (k) \leq 2047$

Operation: TOS \leftarrow PC + 1,
PC [10:0] \leftarrow imm (k),
PC [12:11] \leftarrow PCLATH [4:3]

Status Affected: -

Description: Call Subroutine. First, return address PC+1 is pushed onto the stack. The eleven bit immediate address is loaded into PC [10:0]. The upper bits of the PC are loaded from PCLATH. CALL is two-cycle instruction.

Encoding:

1 0	0 k k k	k k k k	k k k k
-----	---------	---------	---------

Cycles: 4

Example: aaa: CALL bbb

Before Instruction

PC = address aaa

After Instruction

PC = address bbb
SP = address aaa+1

4.2.10 CLRF – CLEAR F

Operands: $0 \leq f \leq 127$

Operation: $f \leq 0x00$
 $FSR \leq 0x00$

Status Affected: Z

Description: The contents of register 'f' is cleared and the Z bit in STATUS register is set

Encoding:

0 0	0 0 0 1	1 f f f	f f f f
0 0	0 0 0 1	1 0 0 0	0 0 0 0

Cycles: DIR : 2
 INDIR : 4

Example: CLRF Flag_Buf

Before Instruction

Flag_Buf = 0xAA

After Instruction

Flag_Buf = 0x00

4.2.11 CLRW – CLEAR W

Operands: -

Operation: W <= 0x00

Status Affected: Z

Description: The contents of working register W is cleared and the Z bit in STATUS register is set

Encoding:

0 0	0 0 0 1	0 x x x	x x x x
-----	---------	---------	---------

Cycles: 2

Example: CLRW

Before Instruction

W = 0xAA

After Instruction

W = 0x00

4.2.12 CLRWDT – CLEAR WATCHDOG TIMER

Operands: -

Operation: WDCNT <= Clear

Status Affected: nTO, nPD

Description: CLRWDT instruction resets the WDCNT. Status bits nTO and nPD are set.

Encoding:

0 0	0 0 0 0	0 1 1 0	0 1 0 0
-----	---------	---------	---------

Cycles: 2

Example: CLRWDT

Before Instruction

WDCNT = ?

After Instruction

WDCNT = 0x00
nTO = 1
nPD = 1

4.2.13 COMF – COMPLEMENT F

Operands: $0 \leq f \leq 127$
 $d \in [0, 1]$

Operation: destination \leftarrow f
 destination \leftarrow FSR

Status Affected: Z

Description: The contents of register 'f' is complemented and transferred to the destination W or 'f' depending of the 'd'.

Encoding:

0 0	1 0 0 1	d f f f	f f f f
0 0	1 0 0 1	d 0 0 0	0 0 0 0

Cycles: DIR : 2
 INDIR : 4

Example: COMF REG1, 0

Before Instruction

REG1 = 0x13

After Instruction

REG1 = 0x13
 W = 0xEC

4.2.14 DECF – DECREMENT F

Operands: $0 \leq f \leq 127$
 $d \in [0, 1]$

Operation: destination $\leq f - 1$
 destination $\leq FSR - 1$

Status Affected: Z

Description: Decrement register 'f'. The result of operation is stored in the destination W or 'f' depending of the 'd' value.

Encoding:

0 0	0 0 1 1	d f f f	f f f f
0 0	0 0 1 1	d 0 0 0	0 0 0 0

Cycles: DIR : 2
 INDIR : 4

Example: DECF REG1, 1

Before Instruction

REG1	=	0x1
Z	=	0

After Instruction

REG1	=	0x0
Z	=	1

4.2.15 DECFSZ – DECREMENT F, SKIP IF 0

Operands: $0 \leq f \leq 127$
 $d \in [0, 1]$

Operation: destination $\leq f - 1$, Skip if result = 0
 destination $\leq \text{FSR} - 1$, Skip if result = 0

Status Affected: -

Description: The contents of register 'f' are decremented and stored in the destination. If the result of operation is 0, the next instruction, which is already fetched, is discarded and NOP is executed instead making it two-cycle instruction.

Encoding:

0 0	1 0 1 1	d f f f	f f f f
0 0	1 0 1 1	d 0 0 0	0 0 0 0

Cycles: DIR : 2
 INDIR : 4

Example:

aaa:	DECFSZ	REG1, 1
	GOTO	aaa
bbb:		

Before Instruction

PC = address aaa

After Instruction

REG1 = REG1 - 1

If REG1 = 0,
 PC = address bbb

If REG1 \neq 0,
 PC = address aaa + 1

4.2.16 GOTO – UNCONDITIONAL BRANCH

Operands: $0 \leq \text{imm (k)} \leq 2047$

Operation: PC [10:0] \leftarrow imm (k)
PC [12:11] \leftarrow PCLATH [4:3]

Status Affected: -

Description: *GOTO* is an unconditional branch. The eleven bit immediate value is loaded into PC bits [10:0]. The upper bits of PC are loaded from PCLATH [4:3]. *GOTO* is a two-cycle instruction

Encoding:

1 0	1 k k k	k k k k	k k k k
-----	---------	---------	---------

Cycles: 4

Example: GOTO Loop

After Instruction

PC = address Loop

4.2.17 INCF– INCREMENT F

Operands: $0 \leq f \leq 127$
 $d \in [0, 1]$

Operation: destination $\leftarrow f + 1$
 destination $\leftarrow \text{FSR} + 1$

Status Affected: Z

Description: The contents of register 'f' are incremented. The result of operation is stored in the W register or 'f', depending of the 'd' value.

Encoding:

0 0	1 0 1 0	d f f f	f f f f
-----	---------	---------	---------

0 0	1 0 1 0	d 0 0 0	0 0 0 0
-----	---------	---------	---------

Cycles: DIR : 2
 INDIR : 4

Example: INCF REG1, 1

Before Instruction

REG1	=	0xFF
Z	=	0

After Instruction

REG1	=	0x0
Z	=	1

4.2.18 INCFSZ – INCREMENT F, SKIP IF 0

Operands: $0 \leq f \leq 127$
 $d \in [0, 1]$

Operation: destination $\leq f + 1$, Skip if result = 0
 destination $\leq FSR + 1$, Skip if result = 0

Status Affected: -

Description: The contents of register 'f' are incremented and stored in the destination. 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.

Encoding:

0 0	1 1 1 1	d f f f	f f f f
0 0	1 1 1 1	d 0 0 0	0 0 0 0

Cycles: DIR : 2
 INDIR : 4

Example: aaa: INCFSZ REG1, 1
 GOTO Loop
 bbb:

Before Instruction

PC = address aaa

After Instruction

REG1 = REG1 + 1

If REG1 = 0,
PC = address bbb

If REG1 \neq 0,
PC = address aaa + 1

4.2.19 IORLW – INCLUSIVE OR LITERAL WITH W

Operands: $0 \leq \text{imm (k)} \leq 255$

Operation: $W \leftarrow W \text{ OR } \text{imm (k)}$

Status Affected: Z

Description: The contents of the W register are OR'ed with the eight bit literal 'k'. The result is placed in the W register.

Encoding:

1 1	1 0 0 0	k k k k	k k k k
-----	---------	---------	---------

Cycles: 2

Example: IORLW 0x35

Before Instruction

W = **0x9A**

After Instruction

W = **0XBF**

Z = **0**

4.2.20 IORWF – INCLUSIVE OR W WITH F

Operands: $0 \leq f \leq 127$
 $d \in [0, 1]$

Operation: destination \leftarrow W OR f
 destination \leftarrow W OR FSR

Status Affected: Z

Description: Inclusive OR the W register with register 'f'. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is placed back in register 'f'.

Encoding:

0 0	0 1 0 0	d f f f	f f f f
0 0	0 1 0 0	d 0 0 0	0 0 0 0

Cycles: DIR : 2
 INDIR : 4

Example: IORWF REG1

Before Instruction

REG1 = 0x13
 W = 0x9A

After Instruction

REG1 = 0X13
 W = 0x93
 Z = 1

4.2.21 MOVLW – MOVE LITERAL TO W

Operands: $0 \leq \text{imm} (k) \leq 255$

Operation: $W \leftarrow \text{imm} (k)$

Status Affected: -

Description: The eight bit immediate data 'imm' is loaded into W register

Encoding:

1	1	0	0	x	x	k	k	k	k	k	k	k	k
---	---	---	---	---	---	---	---	---	---	---	---	---	---

Cycles: 2

Example: MOVLW 0XAA

After Instruction

W = 0xAA

4.2.22 MOVF – MOVE F

Operands: $0 \leq f \leq 127$
 $d \in [0, 1]$

Operation: destination \leftarrow f
 destination \leftarrow FSR

Status Affected: Z

Description: The contents of register f are moved to destination dependent of 'd' value.

Encoding:

0 0	1 0 0 0	d f f f	f f f f
0 0	1 0 0 0	d 0 0 0	0 0 0 0

Cycles: DIR : 2
 INDIR : 4

Example: MOVF REG1, 0

After Instruction

W = value in REG1 register
Z = 1

4.2.23 MOVWF – MOVE W TO F

Operands: $0 \leq f \leq 127$

Operation: $f \leftarrow W$
 $FSR \leftarrow W$

Status Affected: -

Description: Move data from W register to register 'f'.

Encoding:

00	0000	d f f f	f f f f
----	------	---------	---------

00	0000	d 0 0 0	0 0 0 0
----	------	---------	---------

Cycles: DIR : 2
 INDIR : 4

Example: MOVWF REG1

Before Instruction

REG1 = 0x00
 W = 0x9A

After Instruction

REG1 = 0x9A
 W = 0x9A

4.2.24 NOP – NO OPERATION

Operands: -

Operation: -

Status Affected: -

Description: No operation.

Encoding:

0 0	0 0 0 0	0 x x 0	0 0 0 0
-----	---------	---------	---------

Cycles: 2

Example: NOP

4.2.25 RETFIE – RETURN FROM INTERRUPT

Operands: -

Operation: PC \leq TOS
GIE \leq 1

Status Affected: -

Description: Return from interrupt. Stack is POPed and Top of Stack (TOS) is loaded in the PC. Interrupts are enabled by setting Global Interrupt Enable bit, GIE (INTCON(7)).

Encoding:

00	0000	0000	1001
----	------	------	------

Cycles: 4

Example: RETFIE

After Instruction

PC	=	TOS
GIE	=	1

4.2.26 RETLW – RETURN WITH LITERAL IN W

Operands: $0 \leq \text{imm} (k) \leq 255$

Operation: PC \leq TOS
W \leq imm (k)

Status Affected: -

Description: The W register is loaded with eight bit immediate data 'imm'. The program counter is loaded with the return address from the top of stack.

Encoding:

1 1	0 1 x x	k k k k	k k k k
-----	---------	---------	---------

Cycles: 4

Example: CALL TABLE

```
TABLE ADDWF    PC
      RETLWR0
      RETLWR1
      RETLWR2
```

Before Instruction

W = 0x02

After Instruction

W = value of R3

4.2.27 RETURN – RETURN FROM SUBROUTINE

Operands: -

Operation: PC <= TOS

Status Affected: -

Description: Return from subroutine. The stack is POPed and the top of the stack is loaded into the PC.

Encoding:

0 0	0 0 0 0	0 0 0 0	1 0 0 0
-----	---------	---------	---------

Cycles: 4

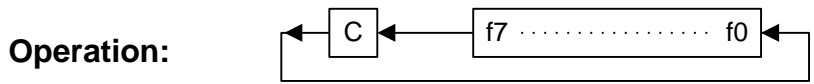
Example: RETURN

After Instruction

PC = TOS

4.2.28 RLF – ROTATE LEFT F THROUGH CARRY

Operands: $0 \leq f \leq 127$
 $d \in [0, 1]$



Status Affected: C

Description: The contents of register 'f' is rotated one bit to the left through the Carry Flag. Result is stored in to destination respectively to the 'd' value.

Encoding:

0 0	1 1 0 1	d f f f	f f f f
0 0	1 1 0 1	d 0 0 0	0 0 0 0

Cycles: DIR : 2
 INDIR : 4

Example: RLF REG1, 0

Before Instruction

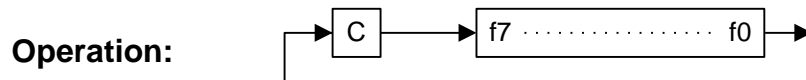
REG1	=	1110 0110
C	=	0

After Instruction

REG1	=	1110 0110
W	=	1100 1100
C	=	1

4.2.29 RRF – ROTATE RIGHT F THROUGH CARRY

Operands: $0 \leq f \leq 127$
 $d \in [0, 1]$



Status Affected: C

Description: The contents of register 'f' are rotated one bit to the right through the Carry Flag. Result is stored in to destination respectively to the 'd' value.

Encoding:

00	1100	d f f f	f f f f
----	------	---------	---------

00	1100	d 0 0 0	0 0 0 0
----	------	---------	---------

Cycles: DIR : 2
 INDIR : 4

Example: RRF REG1, 0

Before Instruction

REG1	=	1110 0110
C	=	0

After Instruction

REG1	=	1110 0110
W	=	0111 0011
C	=	0

4.2.30 SLEEP

Operands: -

Operands: WDCNT <= Clear

Status Affected: nTO, nPD

Description: Status bits nTO is set and nPD is cleared. WDCNT is cleared. It will be in "STOP" mode when CPUCLK.7-.6 value is '10', and the 'SLEEP' instruction is executed. It will be in "IDLE" mode when CPUCLK.7-.6 value is any values except for '10', and the 'SLEEP' instruction is executed.

Encoding:

0 0	1 1 0 0	0 1 1 0	0 0 1 1
-----	---------	---------	---------

Cycles: 2

Example: SLEEP

4.2.31 SUBLW – SUBTRACT W FROM LITERAL

Operands: $0 \leq \text{imm}(k) \leq 255$

Operands: $W \leftarrow \text{imm}(k) - W$

Status Affected: C, DC, Z

Description: The W register is subtracted (2' complement method) from the eight bit immediate data 'imm'. The result is stored in W register.

Encoding:

1 1	1 1 0 x	k k k k	k k k k
-----	---------	---------	---------

Cycles: 2

4.2.31 SUBLW – SUBTRACT W FROM LITERAL(CONTINUED)

Example:

SUBLW0x02

Before Instruction

W = 1
 C = ?
 Z = ?

After Instruction

W = 1
 C = 1; result is positive
 Z = 0

Before Instruction

W = 2
 C = ?
 Z = ?

After Instruction

W = 0
 C = 1; result is zero
 Z = 1

Before Instruction

W = 3
 C = ?
 Z = ?

After Instruction

W = 0xFF
 C = 0; result is negative
 Z = 0

4.2.32 SUBWF – SUBTRACT W FROM F

Operands: $0 \leq f \leq 127$
 $d \in [0, 1]$

Operands: destination $\leftarrow f - W$
destination $\leftarrow FSR - W$

Status Affected: C, DC, Z

Description: Subtract (2's complement method) W register from register 'f'. Result is stored in destination respectively to the value of 'd'.

Encoding:

00	0010	d f f f	f f f f
----	------	---------	---------

00	0010	d 0 0 0	0 0 0 0
----	------	---------	---------

Cycles: DIR : 2
INDIR : 4

4.2.32 SUBWF – SUBTRACT W FROM F(CONTINUED)

Example:

SUBWF REG1, 1

Before Instruction

REG1 = 3
W = 2
C = ?
Z = ?

After Instruction

REG1 = 1
W = 2
C = 1; result is positive
Z = 0

Before Instruction

REG1 = 2
W = 2
C = ?
Z = ?

After Instruction

REG1 = 0
W = 2
C = 1; result is zero
Z = 1

Before Instruction

REG1 = 1
W = 2
C = ?
Z = ?

After Instruction

REG1 = 0xFF
W = 2
C = 0; result is negative
Z = 0

4.2.33 SWAPF – SWAP NIBBLES IN F

Operands: $0 \leq f \leq 127$
 $d \in [0, 1]$

Operands: destination [7:4] \leftarrow f [3:0],
 destination [3:0] \leftarrow f [7:4]

Status Affected: -

Description: The upper and lower nibbles of register 'f' are exchanged. Result is stored in destination respectively to the value of 'd'.

Encoding:

0 0	1 1 1 0	d f f f	f f f f
0 0	1 1 1 0	d 0 0 0	0 0 0 0

Cycles: DIR : 2
 INDIR : 4

Example: SWAPFREG1

Before Instruction

REG1 = 0xA5

After Instruction

REG1 = 0xA5
 W = 0x5A

4.2.34 XORLW – EXCLUSIVE OR LITERAL WITH W

Operands: $0 \leq \text{imm (k)} \leq 255$

Operands: $W \leq \text{imm (k)} \text{ XOR } W$

Status Affected: Z

Description: The Exclusive OR of the contents of W register and 8 bit immediate data is stored in W.

Encoding:

1 1	1 0 1 0	k k k k	k k k k
-----	---------	---------	---------

Cycles: 2

Example: XORLW 0xAF

Before Instruction

W = **0xB5**

After Instruction

W = **0x1A**

4.2.35 XORWF – EXCLUSIVE OR W WITH F

Operands: $0 \leq f \leq 127$
 $d \in [0, 1]$

Operands: destination \leftarrow W XOR f
 destination \leftarrow W XOR FSR

Status Affected: Z

Description: Exclusive OR the W register with register 'f'. Result is stored in destination respectively to the value of 'd'.

Encoding:

0 0	0 1 1 0	d f f f	f f f f
-----	---------	---------	---------

0 0	0 1 1 0	d 0 0 0	0 0 0 0
-----	---------	---------	---------

Cycles: DIR : 2
 INDIR : 4

Example: XORWF REG1

Before Instruction

REG1	=	0xAF
W	=	0xB5

After Instruction

REG1	=	0x1A
W	=	0xB5

NOTES

5. CLOCK CIRCUIT

The MC71PB204 microcontroller has an oscillator circuit: a main clock circuit. The CPU and peripheral hardware is operated on system clock frequency supplied through this circuit. The maximum CPU clock frequency of the MC71PB204 is determined by CPUCLK register settings.

5.1 SYSTEM CLOCK CIRCUIT

The system clock circuit has the following components:

- External crystal, ceramic resonator, or an external clock source
- Internal RC or External RC oscillation source
- Oscillator stop and wake-up functions
- Programmable frequency divider for the CPU clock (f_{xx} divided by 1, 2, 8, or 16)
- CPU clock control register, CPUCLK

5.1.1 MAIN OSCILLATOR CIRCUITS

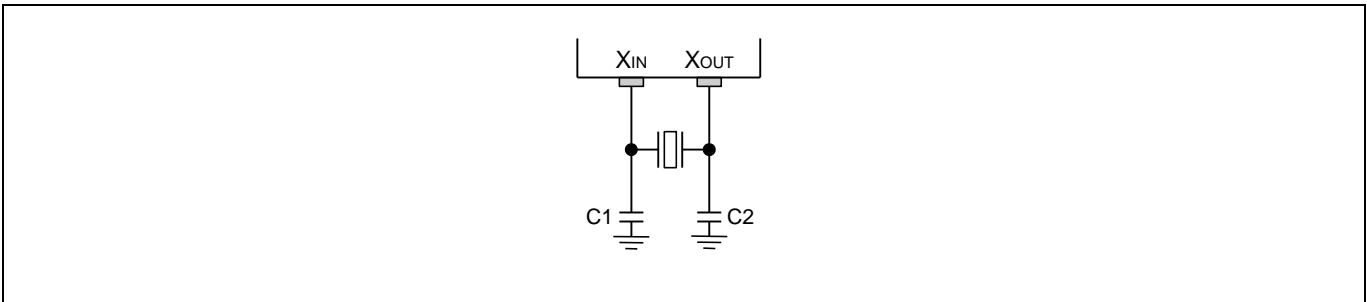


Figure 5-1. Crystal/Ceramic Oscillator (fx)

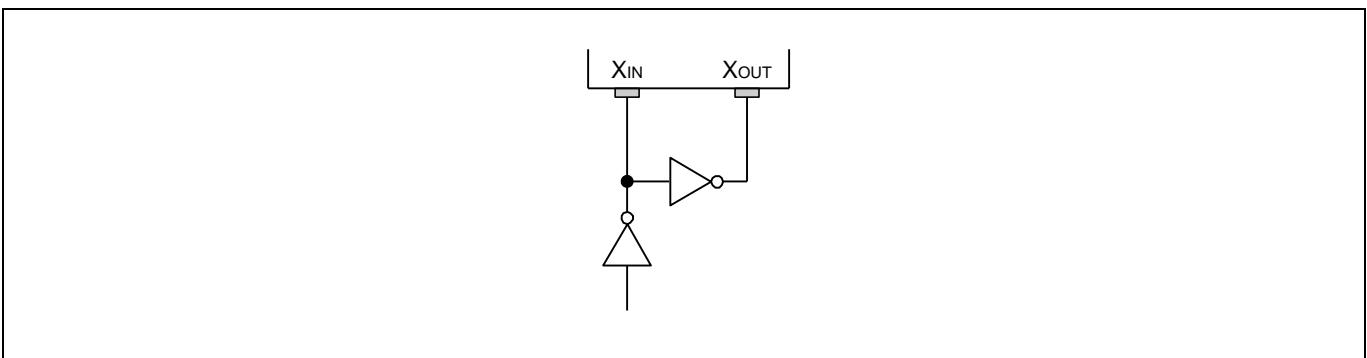


Figure 5-2. External Oscillator (fx)

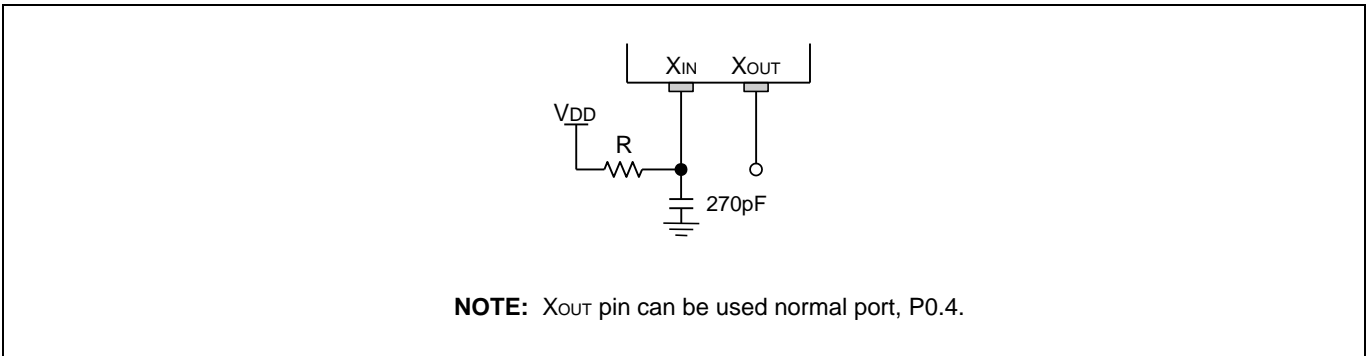


Figure 5-3. External RC Oscillator (fx)

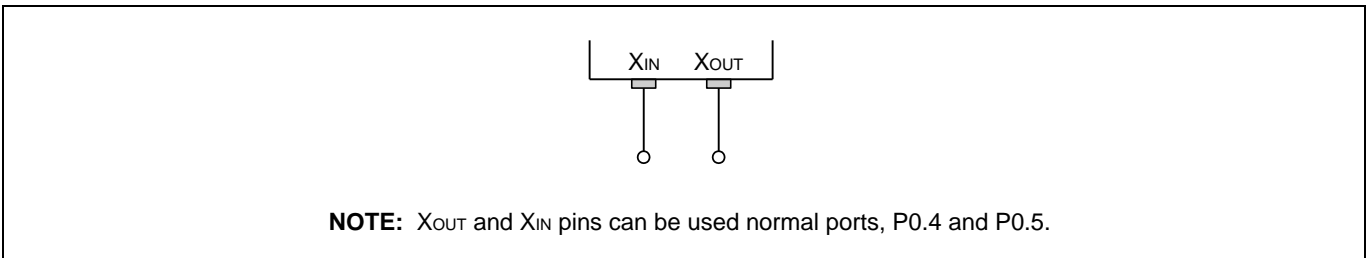
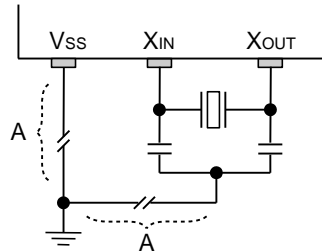


Figure 5-4. Internal RC Oscillator (fx)

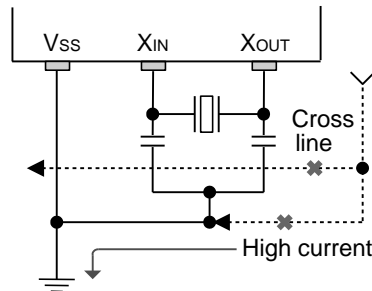
NOTE: fx means main oscillator clock.

5.1.2 EXAMPLE OF INCORRECT OSCILLATOR CONNECTION



NOTE: Mount the oscillator as close as possible to pins, and "A" line should not be too long.

Figure 5-5. Incorrect oscillator connection (1)



NOTES:

1. Signal line should not cross the oscillator circuit.
2. High current should not flow through ground line of oscillator.

Figure 5-6. Incorrect oscillator connection (2)

5.1.3 CLOCK STATUS DURING POWER-DOWN MODES

The two power-down modes, Stop mode and Idle mode, affect the system clock as follows:

- In Stop mode, the main oscillator is halted. Stop mode is released by a reset operation or an external interrupt and the oscillator is started.
- In Idle mode, the internal clock signal is gated to the CPU, but not to interrupt structure and timer/counter. Idle mode is released by a reset or by an external or internal interrupt.

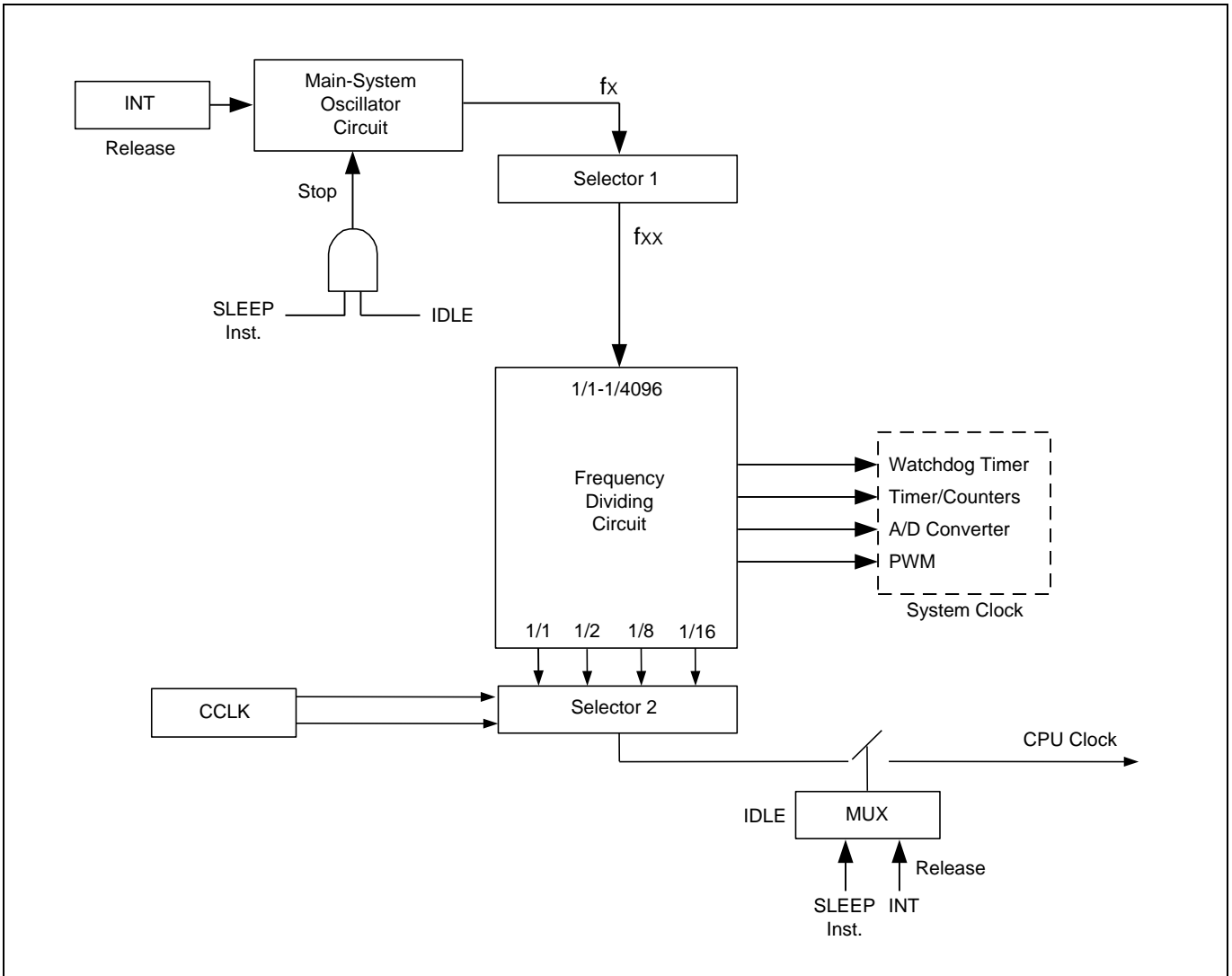


Figure 5-7. System Clock Circuit Diagram

5.1.4 CPU CLOCK CONTROL REGISTER

CPU CLOCK CONTROL REGISTER (CPUCLK)

01H

	7	6	5	4	3	2	1	0	
CPUCLK	IDLE		-	-	-	-	CCLK		(Initial value : 00 - - - -11)
Read/Write	R/W	R/W	-	-	-	-	R/W	R/W	

IDLE	IDLE Mode	10: Enable STOP Others : Enable IDLE; (CPU clock is stop)
Bit5-2	Not available for the MC71PB204	
CCLK	CPU Clock (System Clock) Selection Bits(note)	00: fxx 01: fxx/2 10: fxx/8 11: fxx/16

NOTES:

1. After reset the slowest clock (divided by 16) is selected as the system clock. To select faster clock speed, load the appropriate values to CPUCLK<0> and CPUCLK<1>.
2. It will be in "STOP" mode when CPUCLK<7:6> value is '10b' and the "SLEEP" inst. is executed.
3. It will be in "IDLE" mode when CPUCLK<7:6> value is not '10b' and the "SLEEP" inst. is executed.

NOTES

6. RESET AND POWER-DOWN MODE

6.1 RESET

The MC71PB204 has four resets.

- Power on reset
- Low voltage reset (LVR)
- nMCLR reset
- Watchdog timer overflow reset

Under the descriptions are about initial value according to each reset mentioned above;

- The Table 6-1~6-3 shows the register initial values. The values are initial values by power on reset, LVR, nMCLR, and watchdog timer overflow reset.
- The W register's initial value by power on reset, LVR, nMCLR, and watchdog timer overflow reset is "00000000b".
- The RAM (general purpose registers)'s initial value is undefined by the power on reset and retentive on the LVR, nMCLR or watchdog timer overflow reset.

Table 6-1. BANK0 Register and Initial Values

Register Name	Address (Hex)	Initial Value (bit)							
		7	6	5	4	3	2	1	0
CONFIG	00H	–	–	–	–	–	–	0	0
CPUCLK	01H	0	0	–	–	–	–	1	1
PCL	02H	0	0	0	0	0	0	0	0
STATUS	03H	Refer to the Table 6-4							
FSR	04H	x	x	x	x	x	x	x	x
ADMR	05H	0	0	0	0	0	0	0	0
ADDRH	06H	x	x	x	x	x	x	x	x
ADDRL	07H	x	x	x	x	–	–	–	–
WTSCR	08H	0	0	0	0	0	1	1	0
WTCR	09H	0	0	0	0	0	0	0	0
PCLATH	0AH	–	–	–	0	0	0	0	0
INTCON	0BH	0	0	0	0	0	–	–	–
IPND	0CH	–	0	0	0	0	–	–	–
TSCRA	0DH	–	–	0	0	1	1	1	0
TCRA	0EH	1	1	1	1	1	1	1	1
TDRA	0FH	1	1	1	1	1	1	1	1
TSCRB	10H	–	–	0	0	1	1	1	–
TCRB	11H	1	1	1	1	1	1	1	1
TDRB	12H	1	1	1	1	1	1	1	1
13H-14H are not mapped									
DDR0H	15H	0	0	0	–	0	0	0	–
DDR0L	16H	0	0	0	0	0	0	0	0
PUR0	17H	–	0	0	0	0	0	0	0
DDR1	18H	0	0	0	0	0	0	0	0
DDR2H	19H	0	0	0	0	0	0	0	0
DDR2L	1AH	0	0	0	0	0	0	0	0
EINT2	1BH	0	0	0	0	0	0	0	0
EPND2	1CH	–	–	–	–	0	0	0	0
P0	1DH	0	0	0	0	0	0	0	0
P1	1EH	0	0	0	0	0	0	0	0
P2	1FH	0	0	0	0	0	0	0	0

NOTES:

1. An 'x' means that the bit value is undefined following reset.
2. A dash (–) means that the bit is neither used nor mapped, but the bit is read as '0'.

Table 6-2. BANK1 Register and Initial Values

Register Name	Address (Hex)	Initial Value (bit)							
		7	6	5	4	3	2	1	0
CONFIG	80H	–	–	–	–	–	–	0	0
81H is not mapped									
PCL	82H	0	0	0	0	0	0	0	0
STATUS	83H	Refer to the Table 6-4							
FSR	84H	x	x	x	x	x	x	x	x
PWMSCR	85H	0	0	–	–	0	0	0	–
PWMCR	86H	0	0	0	0	0	0	0	0
PWMDR	87H	0	0	0	0	0	0	0	0
88H-89H are not mapped									
PCLATH	8AH	–	–	–	0	0	0	0	0
8BH-9FH are not mapped									

NOTES:

1. An 'x' means that the bit value is undefined following reset.
2. A dash ('–') means that the bit is neither used nor mapped, but the bit is read as '0'.

Table 6-3. BANK2 Register and Initial Values

Register Name	Address (Hex)	Initial Value (bit)							
		7	6	5	4	3	2	1	0
CONFIG	100H	–	–	–	–	–	–	0	0
101H is not mapped									
PCL	102H	0	0	0	0	0	0	0	0
STATUS	103H	Refer to the Table 6-4							
FSR	104H	x	x	x	x	x	x	x	x
105H-109H are not mapped									
PCLATH	10AH	–	–	–	0	0	0	0	0
10BH-11FH are not mapped									

NOTES:

1. An 'x' means that the bit value is undefined following reset.
2. A dash ('–') means that the bit is neither used nor mapped, but the bit is read as '0'.

Table 6-4. STATUS Register's Initial Values

STATUS Register's Initial Values	Descriptions
0001 1000b	By power on reset
0000 0000b	By watchdog timer overflow during Sleep mode
0000 1000b	By watchdog timer overflow during Operating mode

6.1.1 RECOMMENDED nMCLR PIN CIRCUIT

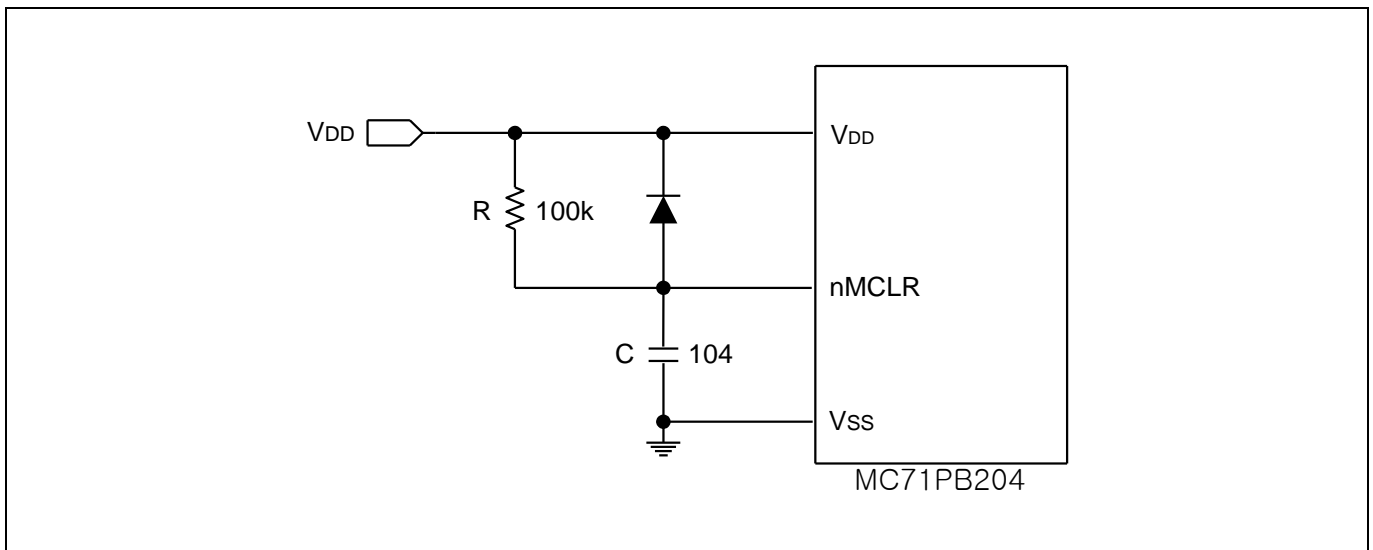


Figure 6-1. Recommended nMCLR Pin Circuit

6.2 POWER-DOWN MODE

6.2.1 STOP MODE

Stop mode is invoked by the instruction SLEEP when the IDLE (CPUCLK<7:6>) = "10b". In Stop mode, the operation of the CPU and all peripherals is halted. That is, the on-chip main oscillator stops and the current consumption is reduced. All system functions stop when the clock "freezes", but data stored in the internal register file is retained. Stop mode can be released through one of two ways; by a reset or by external interrupts.

```

EX)  MOVLW    b'00111111'
      ANDWF   CPUCLK, F
      MOVLW   b'10000000'      ;enable STOP (CPUCLK<7:6> = "10")
      IORWF   CPUCLK, F

      SLEEP   ;STOP Mode
      NOP
      NOP
      NOP
  
```

NOTE: Do not use stop mode if you are using an external clock source because X_{IN} input must be restricted internally to V_{SS} to reduce current leakage.

The Stop mode can be released through one of the following events:

1. External reset input on nMCLR pin.
2. External interrupt from INT0~INT3 pins.

6.2.2 IDLE MODE

Idle mode is invoked by the instruction SLEEP when the IDLE (CPUCLK<7:6>) = "00b", "01b" or "11b". In idle mode, CPU operations are halted while some peripherals remain active. During idle mode, the internal clock signal is gated away from the CPU, but all peripherals remain active. Port pins retain the mode (input or output) they had at the time idle mode was entered.

```

EX)  MOVLW    b'11000000'      ;enable IDLE (CPUCLK<7:6> = "00", "01" or "11")
      IORWF   CPUCLK, F

      SLEEP   ;IDLE Mode
      NOP
      NOP
      NOP
  
```

The Idle mode can be released through one of the following events:

1. External reset input on nMCLR pin.
2. External interrupt from INT0~INT3 pins
3. Internal interrupt on Watchdog timer, Timer A, Timer B or PWM.

NOTES

7. I/O PORTS

The MC71PB204 microcontroller has three bit-programmable I/O ports, P0-P2. The CPU accesses ports by writing or reading port register directly.

7.1 PORT CONFIGURATION

7.1.1 PORT 0 CONFIGURATION

- 1-bit programmable I/O port.
- Schmitt trigger input, push-pull, or open-drain output mode can be selected by software.
- A pull-up resistor (P0.0-P0.5) or pull-down resistor (P0.4-P0.5) can be specified in 1-bit.
- P0.0-P0.3 can be used as AD0-AD3.
- P0.4, P0.5 can be used as X_{OUT} , X_{IN} .

7.1.2 PORT 1 CONFIGURATION

- 1-bit programmable I/O port.
- Schmitt trigger input, push-pull, or open-drain output mode can be selected by software.
- A pull-up resistor can be specified in 1-bit.
- P1.0-P1.2 can be used as AD4-AD6, PWM, TAOOUT, and TACLK.

7.1.3 PORT 2 CONFIGURATION

- 1-bit programmable I/O port.
- P2.0-P2.3 are schmitt trigger input, push-pull output mode can be selected by software.
- P2.4-P2.7 are schmitt trigger input, push-pull, or open-drain output mode selected by software.
NOTE: P2.4-P2.7 are not on 16-pin package.
- A pull-up resistor can be specified in 1-bit.
- P2.0-P2.3 can be used as AD7-AD10 and INT0-INT3.

7.2 PORT REGISTERS

7.2.1 PORT 0 DATA REGISTER

- P0 at location 1DH, Bank0.
- A reset clears the P0 data register to "00H".

7.2.2 PORT 1 DATA REGISTER

- P1 at location 1EH, Bank0.
- A reset clears the P1 data register to "00H".

7.2.3 PORT 2 DATA REGISTER

- P2 at location 1FH, Bank0.
- A reset clears the P2 data register to “00H”.

7.2.4 PORT 0 DATA DIRECTION REGISTER HIGH BYTE (DDR0H)

A reset clears the DDR0H register to ‘00H’, makes P0.5-P0.4 pins input mode, and disables pull-down resistor.

You can use DDR0H register setting to select input (with or without pull-down and pull-up) or output mode (open-drain or push-pull).

If you want to use X_{IN} and X_{OUT} , the OSCS (ROM Option<3:1>) must select to Crystal/ceramic oscillator mode (111b). If you want to use P0.5 and P0.4, the OSCS (ROM Option<3:1>) must select to Internal RC mode (001b, 010b, 011b, 100b).

DDR0H — Port 0 Data Direction Register High Byte

15H

	7	6	5	4	3	2	1	0	
DDR0H	P05	P05PD	-	P04	P04PD	-	(Initial value : 000- 000-)		
Read/Write	R/W	R/W	R/W	-	R/W	R/W	R/W	-	

P05	P0.5/X_{IN}	00: Schmitt trigger input mode 01: Output mode, open-drain 10: Not available 11: Output mode, push-pull
P05PD	P0.5 Pull-down Enable Bit	0: Disable Pull-down resistor 1: Enable Pull-down resistor
Bit4	Not available for the MC71PB204	
P04	P0.4/X_{OUT}	00: Schmitt trigger input mode 01: Output mode, open-drain 10: Not available 11: Output mode, push-pull
P04PD	P0.4 Pull-down Enable Bit	0: Disable Pull-down resistor 1: Enable Pull-down resistor
Bit0	Not available for the MC71PB204	

7.2.5 PORT 0 DATA DIRECTION REGISTER LOW BYTE (DDR0L)

A reset clears the DDR0L register to '00H', makes P0.3-P0.0 pins input mode.

You can use DDR0L register setting to select input or output mode (open-drain or push-pull) and enable alternative functions.

When programming the port, please remember that any alternative peripheral I/O function that defined by the DDR0L register must also be enabled in the associated peripheral module.

DDR0L — Port 0 Data Direction Register Low Byte

16H

	7	6	5	4	3	2	1	0	
DDR0L	P03		P02		P01		P00		(Initial value : 0000 0000)
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P03	P0.3/AD3	00: Schmitt trigger input mode 01: Output mode, open-drain 10: Alternative function (AD3) 11: Output mode, push-pull
P02	P0.2/AD2	00: Schmitt trigger input mode 01: Output mode, open-drain 10: Alternative function (AD2) 11: Output mode, push-pull
P01	P0.1/AD1	00: Schmitt trigger input mode 01: Output mode, open-drain 10: Alternative function (AD1) 11: Output mode, push-pull
P00	P0.0/AD0	00: Schmitt trigger input mode 01: Output mode, open-drain 10: Alternative function (AD0) 11: Output mode, push-pull

7.2.6 PORT 0 PULL-UP RESISTOR ENABLE REGISTER (PUR0)

Using the PUR0 register, you can configure pull-up resistors to individual P0.5-P0.0 pins and P1.2 pin.

PUR0 — Port 0 Pull-up Resistor Enable Register

17H

	7	6	5	4	3	2	1	0	
PUR0	PUR12	-	PUR05	PUR04	PUR03	PUR02	PUR01	PUR00	(Initial value : 0-00 0000)
Read/Write	R/W	-	R/W	R/W	R/W	R/W	R/W	R/W	

PUR12	P1.2 Pull-up Resistor Enable Bit	0: Disable pull-up resistor 1: Enable pull-up resistor
Bit6	Not available for the MC71PB204	
PUR05	P0.5 Pull-up Resistor Enable Bit	0: Disable pull-up resistor 1: Enable pull-up resistor
PUR04	P0.4 Pull-up Resistor Enable Bit	0: Disable pull-up resistor 1: Enable pull-up resistor
PUR03	P0.3 Pull-up Resistor Enable Bit	0: Disable pull-up resistor 1: Enable pull-up resistor
PUR02	P0.2 Pull-up Resistor Enable Bit	0: Disable pull-up resistor 1: Enable pull-up resistor
PUR01	P0.1 Pull-up Resistor Enable Bit	0: Disable pull-up resistor 1: Enable pull-up resistor
PUR00	P0.0 Pull-up Resistor Enable Bit	0: Disable pull-up resistor 1: Enable pull-up resistor

NOTE: A pull-up resistor of port 0 and P1.2 is automatically disabled only when the corresponding pin is selected as push-pull output or alternative function.

7.2.7 PORT 1 DATA DIRECTION REGISTER (DDR1)

A reset clears the DDR1 register to '00H', makes P1.2-P1.0 pins input mode.

You can use DDR1 register setting to select input (with or without pull-up) or output mode (open-drain or push-pull) and enable alternative functions.

When programming the port, please remember that any alternative peripheral I/O function that defined by the DDR1 register must also be enabled in the associated peripheral module.

DDR1 — Port 1 Data Direction Register

18H

	7	6	5	4	3	2	1	0	
DDR1	P12		P11			P10			(Initial value : 0000 0000)
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P12	P1.2/AD6/TACLK	00: Schmitt trigger input mode (TACLK) 01: Output mode, open-drain 10: Alternative function (AD6) 11: Output mode, push-pull
P11	P1.1/AD5/TAOUT	000: Schmitt trigger input mode 001: Schmitt trigger input mode; Pull-up 01x: Alternative function (AD5) 100: Output mode, push-pull 101: Output mode, open-drain 11x: Alternative function (TAOUT)
P10	P1.0/AD4/PWM	000: Schmitt trigger input mode 001: Schmitt trigger input mode; Pull-up 01x: Alternative function (AD4) 100: Output mode, push-pull 101: Output mode, open-drain 11x: Alternative function (PWM)

NOTE: If you use pull-up resistor of Port1.2, must be set to PUR12 (PUR0<7>).

7.2.8 PORT 2 DATA DIRECTION REGISTER HIGH BYTE (DDR2H)

A reset clears the DDR2H register to '00H', makes P2.7-P2.4 pins input pull-up mode. You can use DDR2H register setting to select input (with or without pull-up) or output mode (open-drain or push-pull).

DDR2H — Port 2 Data Direction Register High Byte

19H

	7	6	5	4	3	2	1	0	
DDR2H	P27		P26		P25		P24		(Initial value : 0000 0000)
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

P27	P2.7	00: Schmitt trigger input mode; Pull-up 01: Schmitt trigger input mode 10: Output mode, open-drain 11: Output mode, push-pull
P26	P2.6	00: Schmitt trigger input mode; Pull-up 01: Schmitt trigger input mode 10: Output mode, open-drain 11: Output mode, push-pull
P25	P2.5	00: Schmitt trigger input mode; Pull-up 01: Schmitt trigger input mode 10: Output mode, open-drain 11: Output mode, push-pull
P24	P2.4	00: Schmitt trigger input mode; Pull-up 01: Schmitt trigger input mode 10: Output mode, open-drain 11: Output mode, push-pull

NOTE: P2.7-P2.4 are not on 16-pin package.

7.2.9 PORT 2 DATA DIRECTION REGISTER LOW BYTE (DDR2L)

A reset clears the DDR2L register to '00H', makes P2.3-P2.0 pins input pull-up mode. You can use DDR2L register setting to select input (with or without pull-up) or push-pull output mode and enable alternative functions.

When programming the port, please remember that any alternative peripheral I/O function that defined by the DDR2L register must also be enabled in the associated peripheral module.

DDR2L — Port 2 Data Direction Register Low Byte

1AH

	7	6	5	4	3	2	1	0	
DDR2L	P23		P22		P21		P20		(Initial value : 0000 0000)
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

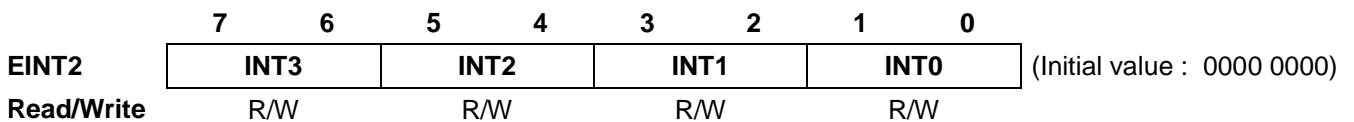
P23	P2.3/AD10/INT3	00: Shmitt trigger input mode; Pull-up 01: Shmitt trigger input mode 10: Alternative function (AD10) 11: Output mode, push-pull
P22	P2.2/AD9/INT2	00: Shmitt trigger input mode; Pull-up 01: Shmitt trigger input mode 10: Alternative function (AD9) 11: Output mode, push-pull
P21	P2.1/AD8/INT1	00: Shmitt trigger input mode; Pull-up 01: Shmitt trigger input mode 10: Alternative function (AD8) 11: Output mode, push-pull
P20	P2.0/AD7/INT0	00: Shmitt trigger input mode; Pull-up 01: Shmitt trigger input mode 10: Alternative function (AD7) 11: Output mode, push-pull

7.2.10 PORT 2 EXTERNAL INTERRUPT REGISTER (EINT2)

A reset clears the EINT2 register to '00H', disables INT3-INT0 interrupt. You can use EINT2 register setting to select Disable interrupt or Enable interrupt (on falling, rising, or both edge).

EINT2 — Port 2 External Interrupt Register

1BH



INT3	P2.3/INT3 External Interrupt Enable Bits	00: Disable Interrupt 01: Enable Interrupt by falling edge 10: Enable Interrupt by rising edge 11: Enable Interrupt by falling and rising edge
INT2	P2.2/INT2 External Interrupt Enable Bits	00: Disable Interrupt 01: Enable Interrupt by falling edge 10: Enable Interrupt by rising edge 11: Enable Interrupt by falling and rising edge
INT1	P2.1/INT1 External Interrupt Enable Bits	00: Disable Interrupt 01: Enable Interrupt by falling edge 10: Enable Interrupt by rising edge 11: Enable Interrupt by falling and rising edge
INT0	P2.0/INT0 External Interrupt Enable Bits	00: Disable Interrupt 01: Enable Interrupt by falling edge 10: Enable Interrupt by rising edge 11: Enable Interrupt by falling and rising edge

7.2.11 PORT 2 EXTERNAL INTERRUPT PENDING REGISTER (EPND2)

The EPND2 register lets you check for interrupt pending conditions and clear the pending condition when the interrupt service routine has been initiated. The application program detects Interrupt requests by polling the EPND2 register.

When the CPU acknowledges the interrupt request, application software must clear the pending condition by writing "0" to the corresponding EPND2 bit.

EPND2 — Port 2 External Interrupt Pending Register

1CH

	7	6	5	4	3	2	1	0	
EPND2	-	-	-	-	PND3	PND2	PND1	PND0	(Initial value : - - - - 0000)
Read/Write	-	-	-	-	R/W	R/W	R/W	R/W	

Bit7-4	Not available for the MC71PB204	
PND3	P2.3/INT3 External Interrupt Pending Bit	0: Interrupt request is not pending (when read); pending bit clear when write 0 1: Interrupt request is pending (when read)
PND2	P2.2/INT2 External Interrupt Pending Bit	0: Interrupt request is not pending (when read); pending bit clear when write 0 1: Interrupt request is pending (when read)
PND1	P2.1/INT1 External Interrupt Pending Bit	0: Interrupt request is not pending (when read); pending bit clear when write 0 1: Interrupt request is pending (when read)
PND0	P2.0/INT0 External Interrupt Pending Bit	0: Interrupt request is not pending (when read); pending bit clear when write 0 1: Interrupt request is pending (when read)

NOTES

8. WATCHDOG TIMER

The MC71PB204 microcontroller has a watchdog timer

It is used in two different ways:

- As a clock source to watchdog timer to provide an automatic reset mechanism in the event of a system malfunction.
- As a signal of the end of the required oscillation stabilization interval after a reset or stop mode release.

8.1 WATCHDOG TIMER REGISTER

8.1.1 WATCHDOG TIMER STATUS AND CONTROL REGISTER (WTSCR)

After reset, the WTSCR register is '06H'. This enables the watchdog function and selects a watchdog timer clock frequency of fxx/4096. To disable the watchdog function, you must write the signature code "1010b" to the WTFUN(WTSCR<7:4>).

WTSCR — Watchdog Timer Status and Control Register

08H

	7	6	5	4	3	2	1	0	
WTSCR	WTFUN				WT3C	WTCS		WTCC	(Initial value : 0000 0110)
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

WTFUN	Watchdog Timer Function Disable Code (for System Reset) Bits	1010: Disable watchdog timer function Others: Enable watchdog timer function
WT3C	Watchdog Timer 3-bit Counter Clear Bit	0: No effect 1: Clear watchdog timer 3-bit counter (Automatically cleared to "0" after being cleared watchdog timer counter)
WTCS	Watchdog Timer Clock Selection Bits	00: fxx/16 01: fxx/128 10: fxx/1024 11: fxx/4096
WTCC	Watchdog Timer Counter Clear Bit	0: No effect 1: Clear the watchdog timer counter value (Automatically cleared to "0" after being cleared watchdog timer counter)

8.1.2 WATCHDOG TIMER BLOCK DIAGRAM

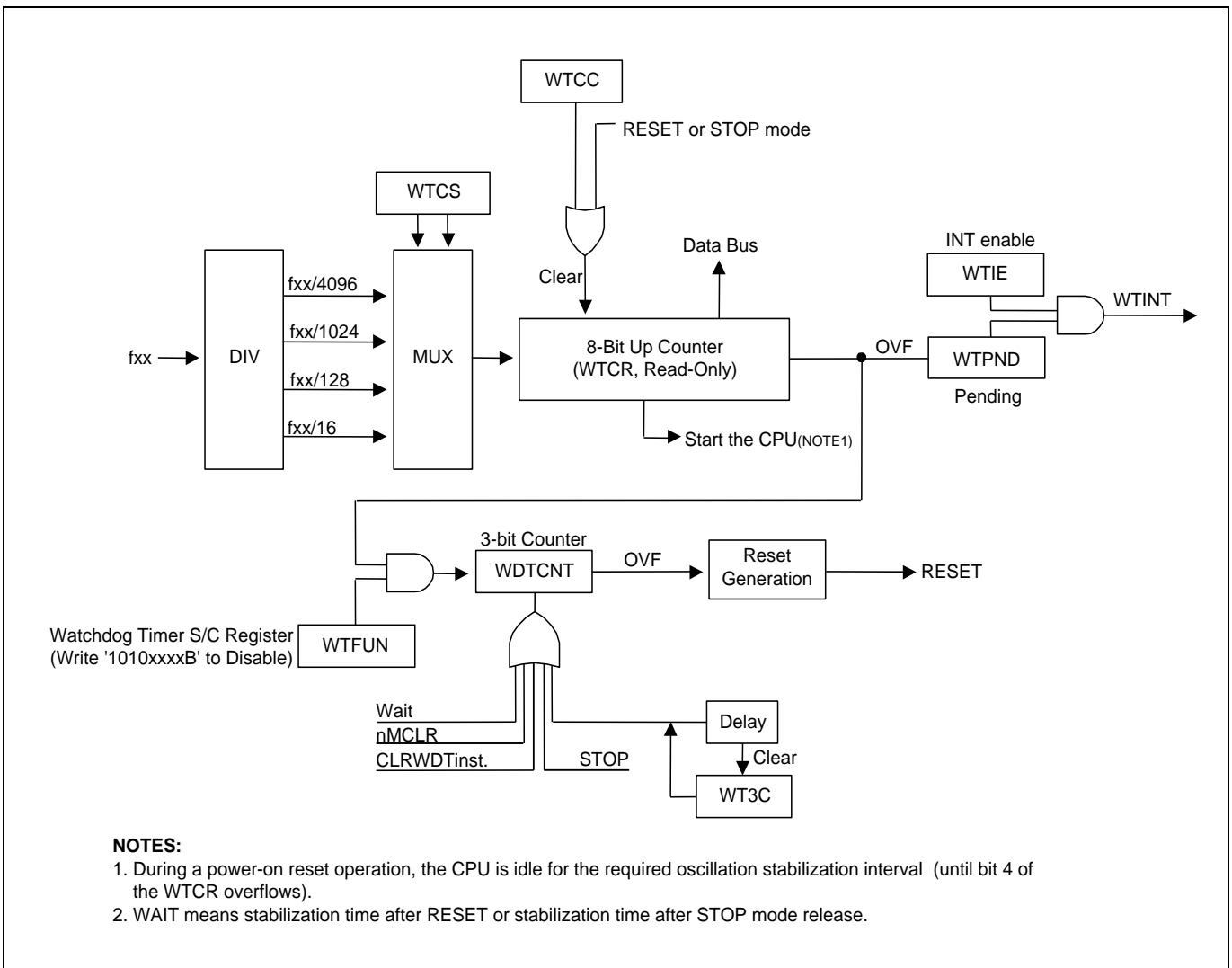


Figure 8-1. Watchdog Timer Block Diagram

NOTES

9. 16-BIT TIMER 0 (8-BIT TIMER A/B)

The 16-bit timer 0 is used in one 16-bit timer or two 8-bit timers mode. When T0MOD is set to “1”, it is one 16-bit timer mode. When T0MOD is cleared to “0”, the timer 0 is used as two 8-bit timers.

- One 16-bit timer mode (Timer 0)
- Two 8-bit timers mode (Timer A/B)

9.1 ONE 16-BIT TIMER MODE (TIMER 0)

Timer 0 has the following functional components:

- Clock frequency divider (fx divided by 512, 256, 64, 8, or 1) with multiplexer
- External clock input pin, TACLK
- 16-bit down counter (TCRA, TCRB) and 16-bit reference data register (TDRA, TDRB)
- Timer 0 status and control register (TSCRA)
- Timer 0 interrupt generation

9.1.1 FUNCTION DESCRIPTION

TAOUT pin is toggled and a timer 0 interrupt is generated when the down counter (TCRA/TCRB) is underflow.

For example, you write the value 11H and 17H to TDRA and TDRB, respectively, and 37H to TSCRA. The counter will decrement from 1117H to 0000H, and then occur underflow. At this point, the timer 0 interrupt request is pending, the value 1117H is reloaded to the down counter (TCRA/TCRB), and counting resumes.

The timer 0 interrupt pending bit must be cleared in the interrupt sub-routine by writing “0” to the TAPND (IPND<4>).

9.1.2 16-BIT TIMER 0 REGISTER

9.1.2.1 Timer A Status and Control Register (TSCRA)

After reset, the TSCRA register is '0EH'. This disables counting operation, selects an input clock frequency of fxx/512, and selects two 8-bit timers mode(Timer A/B). To select one 16-bit timer mode, T0MOD (TSCRA<0>) should be set to "1".

To enable the timer 0 interrupt, you must set "1" to TAIE (INTCON<4>).

When the timer 0 interrupt sub-routine is serviced, the pending condition must be cleared by software writing "0" to the timer 0 interrupt pending bit, TAPND (IPND<4>).

TSCRA — Timer A status and control register

0DH

	7	6	5	4	3	2	1	0	
TSCRA	-	-	TARL	TACE	TACS			T0MOD	(Initial value : - -00 1110)
Read/Write	-	-	R/W	R/W	R/W	R/W	R/W	R/W	

Bit7-6	Not available for the MC71PB204	
TARL	Timer 0 Counter Reload Bit	0: No effect 1: Reload timer data to the Timer 0 Counter
TACE	Timer 0 Count Enable Bit	0: Disable counting operation 1: Enable counting operation (Timer data of TDRA register is reloaded to TCRA register)
TACS	Timer 0 Clock Selection Bits	000: Not available 001: TACLK (external clock) 010: Not available 011: fxx/1 (system clock) 100: fxx/8 101: fxx/64 110: fxx/256 111: fxx/512
T0MOD	Timer 0 Mode Selection Bit	0: Two 8-bit timers mode (Timer A/B) 1: One 16-bit timer mode (Timer 0)

9.1.2.2 16-Bit Timer 0 Block Diagram (T0MOD=1)

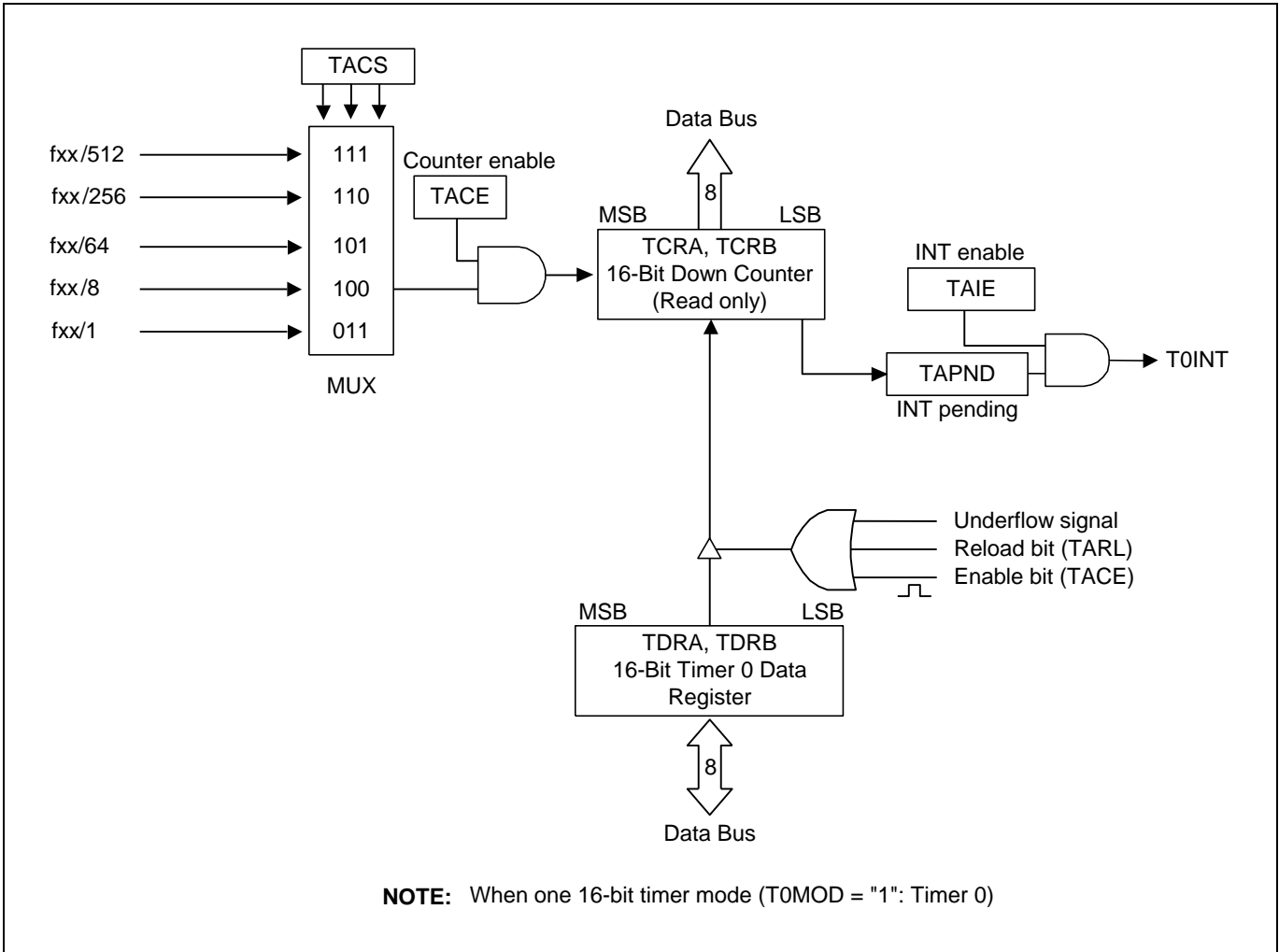


Figure 9-1. 16-Bit Timer 0 Block Diagram

9.2 TWO 8-BIT TIMERS MODE (TIMER A/B)

Timer A has the following functional components:

- Clock frequency divider (fx divided by 512, 256, 64, 8, or 1, and TACLK: External clock) with multiplexer
- 8-bit down counter (TCRA) and 8-bit reference data register (TDRA)
- Timer A status and control register (TSCRA)
- Timer A interrupt generation

Timer B has the following functional components:

- Clock frequency divider (fx divided by 512, 256, 64, 8, or 1) with multiplexer
- 8-bit down counter (TCRB) and 8-bit reference data register (TDRB)
- Timer B status and control register (TSCRB)
- Timer B interrupt generation

9.2.1 FUNCTION DESCRIPTION

9.2.1.1 8-bit Timer A

TAOUT is toggled and a timer A interrupt is generated when the down counter (TCRA) is underflow.

For example, you write the value 25H to TDRA, and 36H to TSCRA. The counter will decrement from 25H to 00H, and then occur underflow. At this point, the timer A interrupt request is pending, the value 25H is reloaded to the down counter (TCRA), and counting resumes.

The timer A interrupt pending bit must be cleared in the interrupt sub-routine by writing "0" to the TAPND (IPND<4>).

9.2.1.2 8-bit Timer B

A timer B interrupt is generated when the down counter (TCRB) is underflow.

For example, you write the value 81H to TDRB, and 36H to TSCRB. The counter will decrement from 81H to 00H, and then occur underflow. At this point, the timer B interrupt request is pending, the value 81H is reloaded to the down counter (TCRB), and counting resumes.

The timer B interrupt pending bit must be cleared in the interrupt sub-routine by writing "0" to the TBPND (IPND<3>).

9.2.2 8-BIT TIMER A REGISTER

9.2.2.1 Timer A Status and Control Register (TSCRA)

After reset, the TSCRA register is '0EH'. This disables counting operation, selects an input clock frequency of fxx/512, and selects two 8-bit timers mode(Timer A/B).

To enable the timer A interrupt, you must set "1" to TAIE (INTCON<4>).

When the timer A interrupt sub-routine is serviced, the pending condition must be cleared by software writing "0" to the timer A interrupt pending bit, TAPND (IPND<4>).

TSCRA — Timer A status and control register

0DH

	7	6	5	4	3	2	1	0	
TSCRA	-	-	TARL	TACE	TACS			TOMOD	(Initial value : - -00 1110)
Read/Write	-	-	R/W	R/W	R/W	R/W	R/W	R/W	

Bit7-6	Not available for the MC71PB204	
TARL	Timer A Counter Reload Bit	0: No effect 1: Reload timer data to the Timer A Counter
TACE	Timer A Count Enable Bit	0: Disable counting operation 1: Enable counting operation (Timer data of TDRA register is reloaded to TCRA register)
TACS	Timer A Clock Selection Bits	000: Not available 001: TACLK (external clock) 010: Not available 011: fxx/1 (system clock) 100: fxx/8 101: fxx/64 110: fxx/256 111: fxx/512
TOMOD	Timer 0 Mode Selection Bit	0: Two 8-bit timers mode (Timer A/B) 1: One 16-bit timer mode (Timer 0)

9.2.2.2 8-Bit Timer A Block Diagram (T0MOD=0)

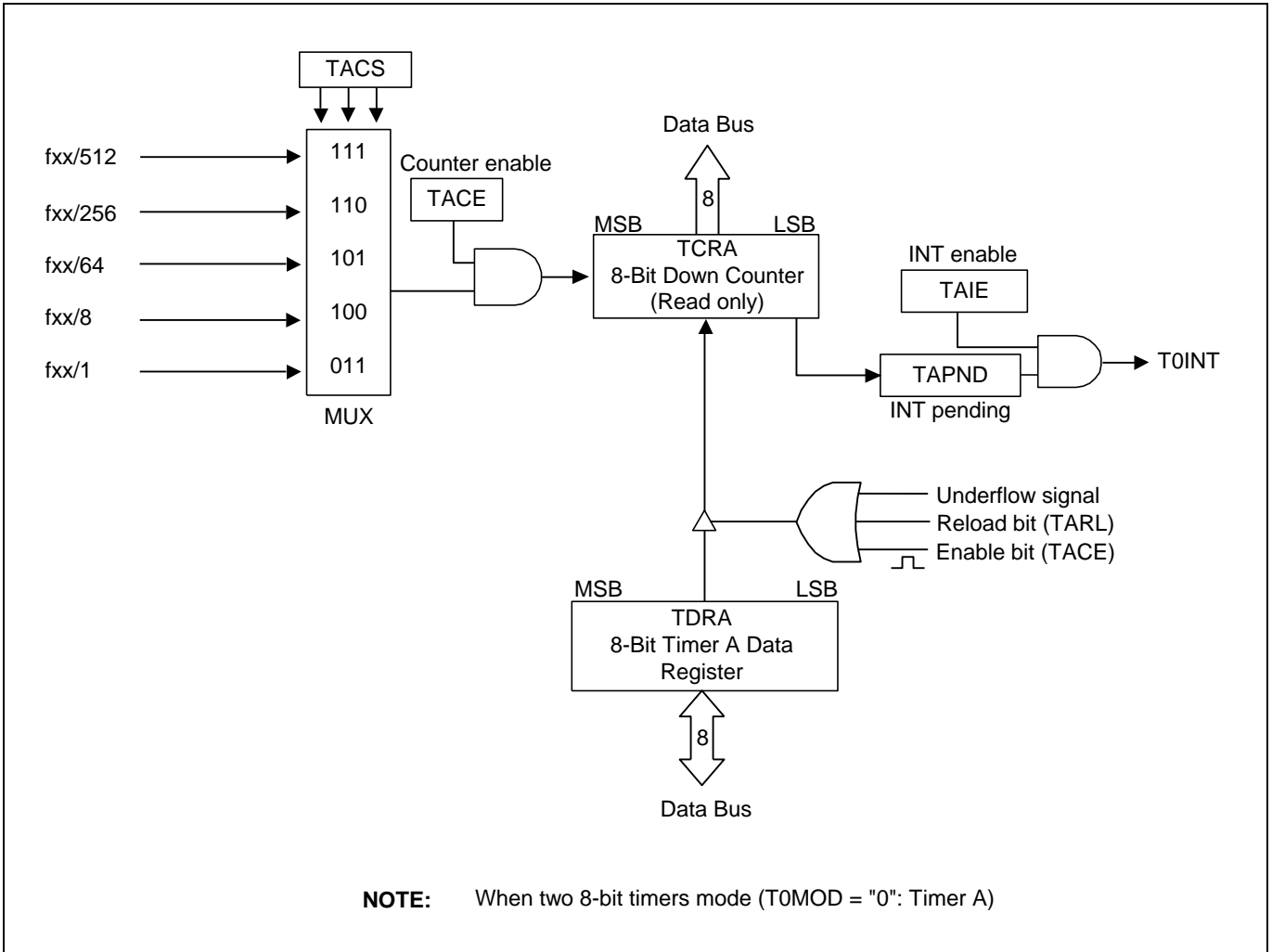


Figure 9-2. 8-Bit Timer A Block Diagram

9.2.3 8-BIT TIMER B REGISTER

9.2.3.1 Timer B Status and Control Register (TSCRB)

After reset, the TSCRB register is '0EH'. This disables counting operation and selects an input clock frequency of fxx/512. You must write "0" to T0MOD (TSCRA<0>).

To enable the timer B interrupt, you must write "1" to TBIE (INTCON<3>).

When the timer B interrupt sub-routine is serviced, the pending condition must be cleared by software writing "0" to the timer B interrupt pending bit, TBPND (IPND<3>).

TSCRB — Timer B status and control register

10H

	7	6	5	4	3	2	1	0	
TSCRB	-	-	TBRL	TBCE	TBCS			-	(Initial value : --00 111-)
Read/Write	-	-	R/W	R/W	R/W	R/W	R/W	-	

Bit7-6	Not available for the MC71PB204	
TBRL	Timer B Counter Reload Bit	0: No effect 1: Reload timer data to the Timer B Counter
TBCE	Timer B Count Enable Bit	0: Disable counting operation 1: Enable counting operation (Timer data of TDRB register is reloaded to TCRB register)
TBCS	Timer B Clock Selection Bits	000: Not available 001: Not available 010: Not available 011: fxx/1 (system clock) 100: fxx/8 101: fxx/64 110: fxx/256 111: fxx/512
Bit0	Not available for the MC71PB204	

9.2.3.2 8-Bit Timer B Block Diagram (T0MOD=0)

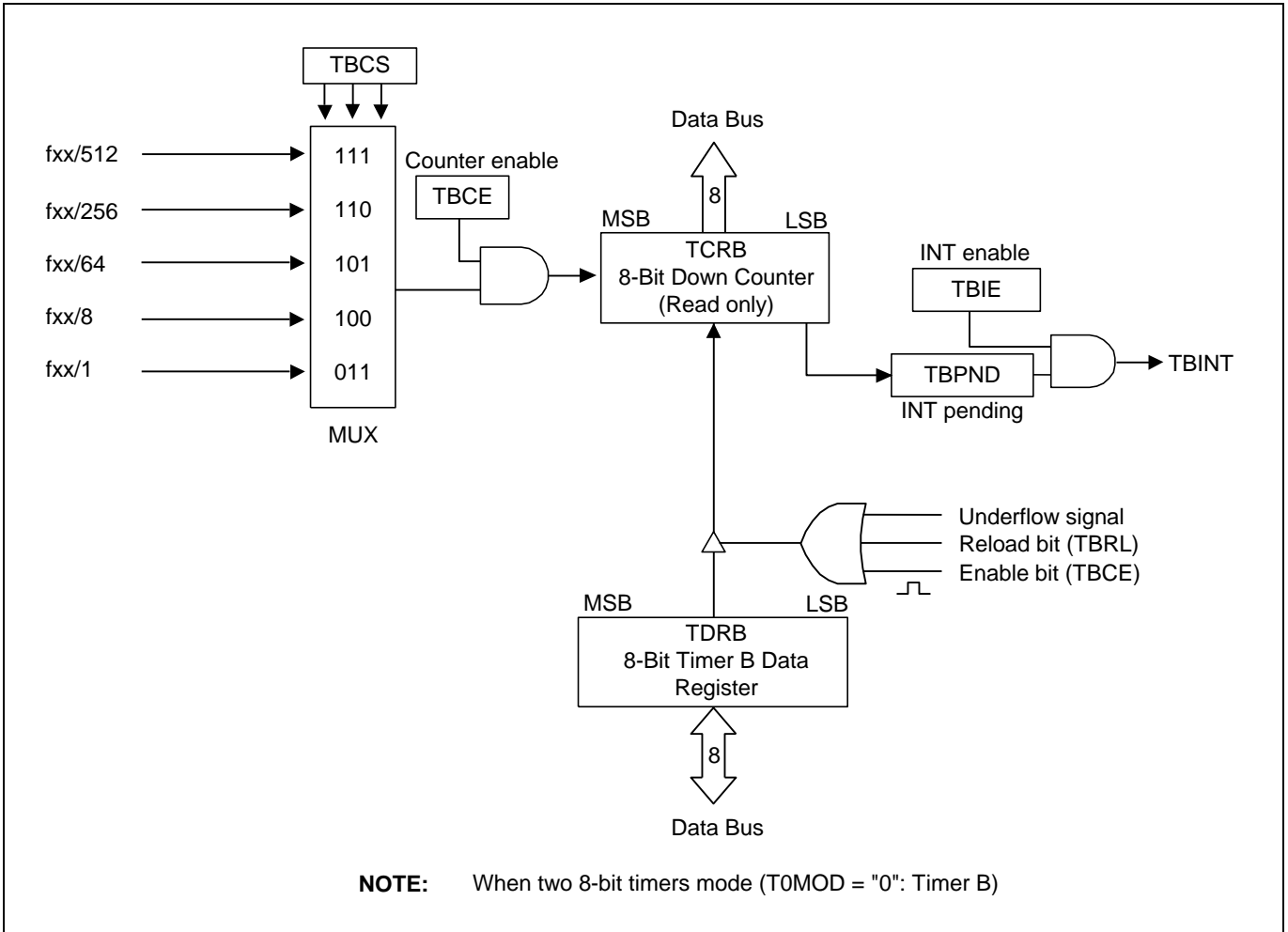


Figure 9-3. 8-Bit Timer B Block Diagram

10. 6/8-BIT PWM

The MC71PB204 microcontroller has a 6/8-bit PWM. The PWM status and control register, PWMSCR, is used to select the input clock frequency, to clear PWM counter, to reload PWM data, and to enable or disable the PWM counter. It is located in bank 1 at address 85h, and is readable/writable.

To enable the PWM interrupt (PWMIE), you must write "1" to PWMIE (INTCON<5>) and PWME (PWSCR<1>). To detect an interrupt pending condition when PWMIE is disabled, the application program polls pending bit, PWMPND (IPND<5>). If it is detected, it means a PWM interrupt is pending. When the PWM interrupt sub-routine is serviced, the pending condition must be cleared by software writing "0" to the PWM interrupt pending bit, PWMPND (IPND<5>).

10.1 PWM REGISTER

10.1.1 PWM STATUS AND CONTROL REGISTER (PWMSCR)

A reset clears PWMSCR to '00H'. This selects an input clock frequency of $f_{xx}/1$ and disables counting operation. So, if you want to use the PWM, you must write "1" to PWME (PWMSCR<1>) and write "11xb" to P10(DDR1<2:0>).

When the PWM interrupt sub-routine is serviced, the pending condition must be cleared by software writing "0" to the PWM interrupt pending bit, PWMPND (IPND<5>).

PWMSCR — PWM Status And Control Register

85H

	7	6	5	4	3	2	1	0	
PWMSCR	PWMICS		-	-	PWMD	PWMC	PWME	-	(Initial value : 00- - 000-)
Read/Write	R/W	R/W	-	-	R/W	R/W	R/W	-	

PWMICS	PWM Input Clock Selection Bits	00: $f_{xx}/1$ 01: $f_{xx}/2$ 10: $f_{xx}/8$ 11: $f_{xx}/64$
Bit5-4	Not available for the MC71PB204	
PWMD	PWM Data Reload interval Selection Bit	0: Reload from 8-bit up counter overflow 1: Reload from 6-bit up counter overflow
PWMC	PWM Counter Clear Bit	0: No effect 1: Clear the PWM Counter (when write, automatically cleared to "0")
PWME	PWM Counter Enable Bit	0: Stop Counter 1: Start (Resume Countering)
Bit0	Not available for the MC71PB204	

10.2 PWM FUNCTION DESCRIPTION

The PWM output signal toggles to low level whenever the lower 6 bits of counter matches the PWM data register (PWMDR<7:2>). If the value in the PWMDR<7:2> register is not zero, an overflow of the lower 6 bits of counter causes the PWM output to toggle to high level. In this way, the reference value written to the PWMDR<7:2> determines the module's base duty cycle.

The value in the upper 2 bits of counter is compared with the extension setting in the two extension bits of PWMDATA register (PWMDR<1:0>). These upper 2 bits of counter value, together with extension logic and the PWMDR<1:0> are used to "stretch" the duty cycle of the PWM output. The "stretch" value is one extra clock period at specific intervals, or cycles.

If PWM clock is 4MHz, for example, the PWMDR<1:0> is '01b', the 3rd cycle will be one pulse longer than the other 3 cycles. If the base duty cycle is 50 %, the duty of the 3rd cycle will be "stretched" to approximately 51% duty. For example, if you write '10b' to the PWMDR<1:0>, all even-numbered cycles will be one pulse longer. If you write '11b' to the PWMDR<1:0>, all cycles will be stretched to one pulse except the 1st cycle. PWM output goes to an output buffer and then to the corresponding PWM output pin. In this way, you can obtain high output resolution at high frequencies.

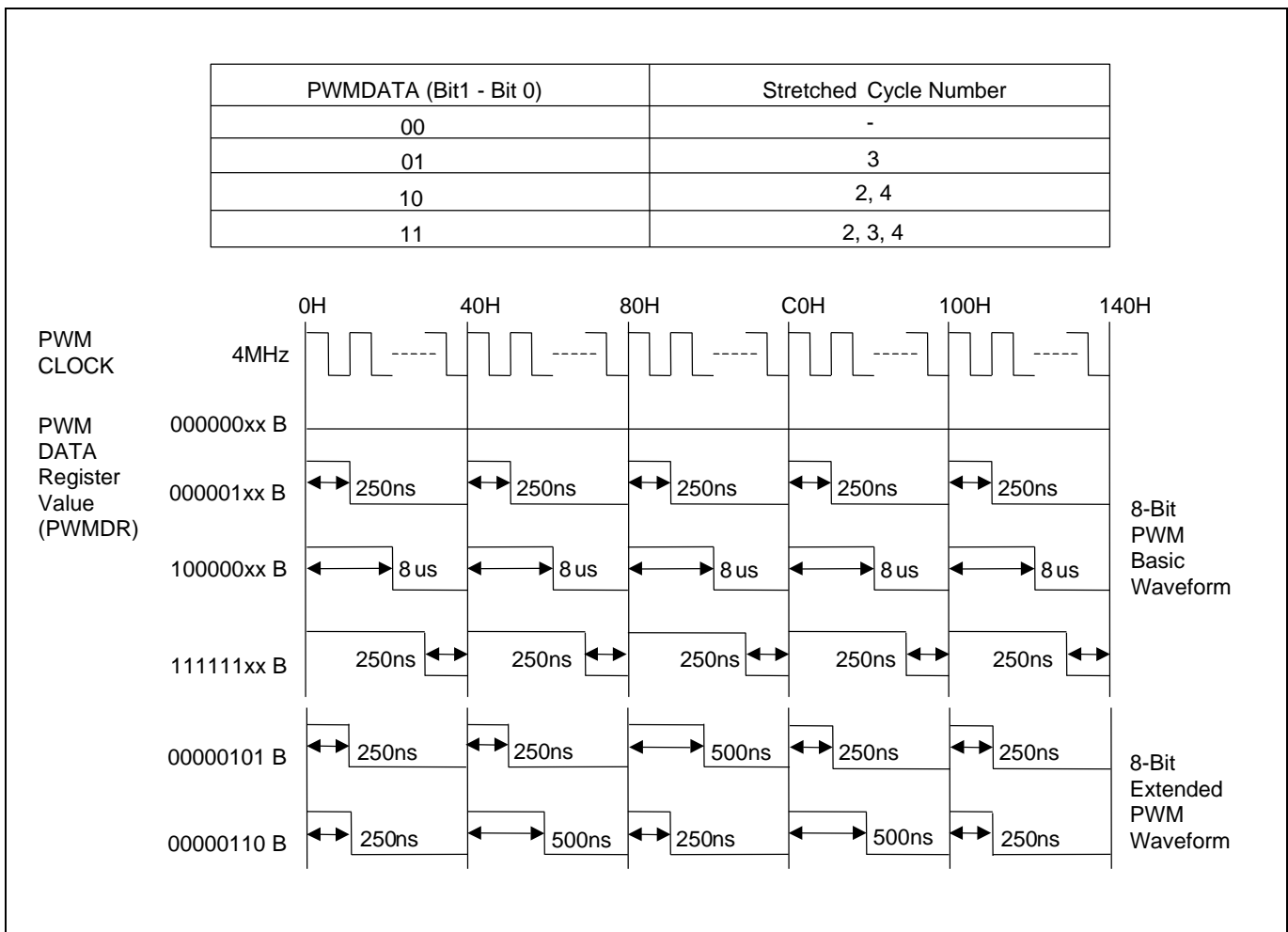


Figure 10-1. PWM Waveform

10.3 PWM FUNCTION BLOCK DIAGRAM

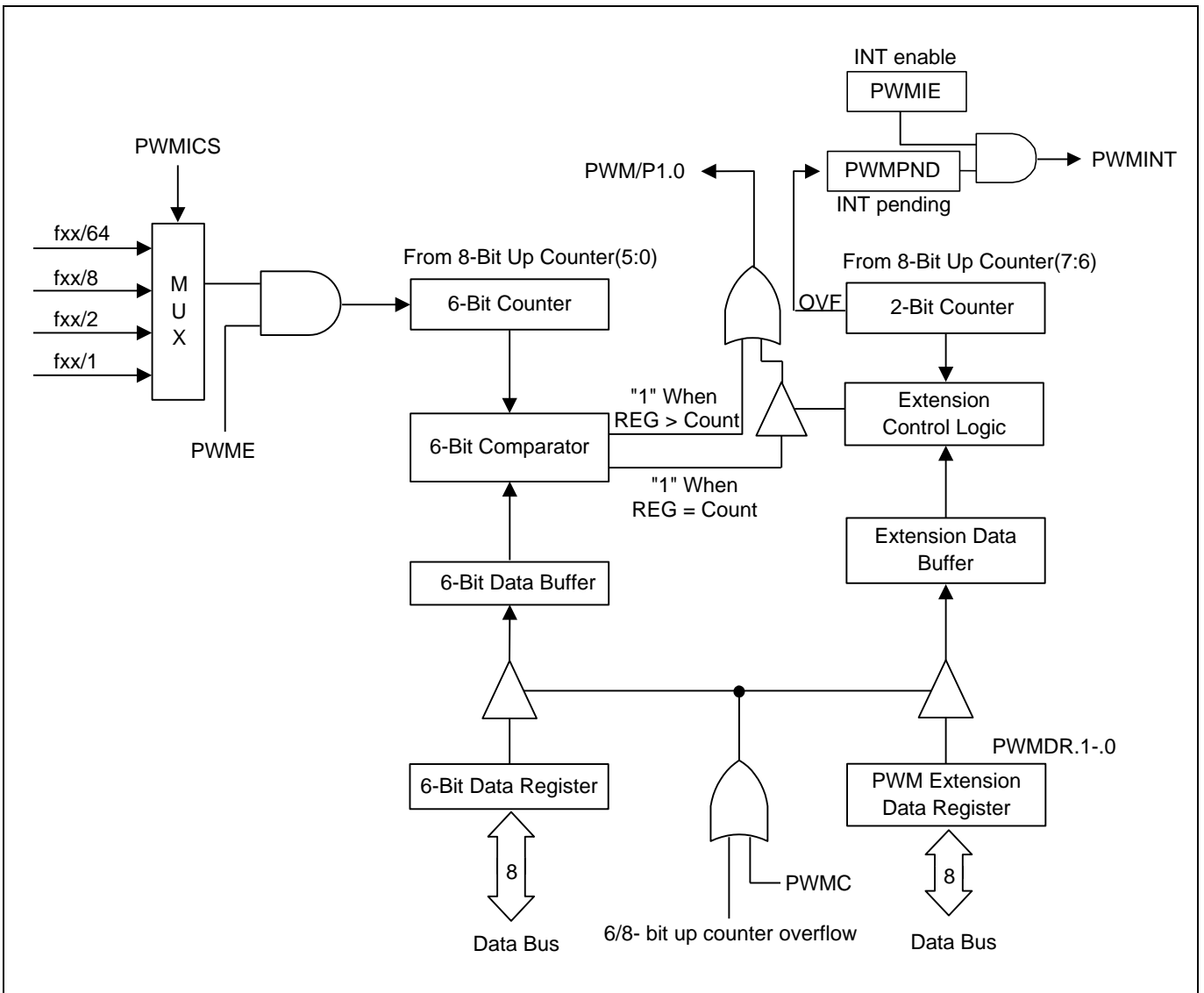


Figure 10-2. PWM Circuit Diagram

11. 12-BIT ANALOG TO DIGITAL CONVERTER

The 12-bit A/D converter (ADC) module uses successive approximation logic to convert analog levels entering at one of the eleven input channels to equivalent 12-bit digital values. The analog input level must lie between the AV_{REF} and AV_{SS} values. The A/D converter has the analog comparator with successive approximation logic, D/A converter logic (resistor string type), A/D mode register (ADMR), eleven multiplexed analog data input pins (AD0-AD10), and 12-bit A/D conversion data output register (ADDRH/ADDRL).

11.1 FUNCTION DESCRIPTION

To initiate an analog-to-digital conversion procedure, at first you must set P0.0-P0.3, P1.0-P1.2, or P2.0-P2.3 with alternative function to enable ADC analog input. And you write the channel selection data in the A/D mode register (ADMR) to select one of the eleven analog input pins (AD0-AD10) and set the conversion start/stop bit, SSBIT (ADMR<7>). The readable-writable ADMR register is located in address 05H. The pins not used for ADC can be used for normal I/O.

During a normal conversion, ADC logic initially sets the successive approximation register with 800H (the approximate half-way point of a 12-bit register). This register is then updated automatically during each conversion step. The successive approximation block performs 12-bit conversions for one input channel at a time. You can dynamically select different channels by manipulating the channel selection bits, ADCH (ADMR<3:0>). To start the A/D conversion, you should set the start/stop bit, SSBIT (ADMR<7>). When a conversion is completed, the end-of-conversion bit, EOC (ADMR<6>) is automatically set to 1 and the result is dumped into the ADDRH/ADDRL register where it can be read. Then the A/D converter enters an idle state. The EOC bit is cleared when SSBIT is set. Remember to read the contents of ADDRH/ADDRL before another conversion starts. Otherwise, the previous result will be overwritten by the next conversion result.

NOTE

Because the A/D converter has no sample-and-hold circuitry, it is very important that fluctuation of the analog level at the AD0-AD10 input pins during a conversion procedure be kept to an absolute minimum. Any change in the input level, perhaps due to noise, will invalidate the result.

If the chip enters to STOP or IDLE mode in conversion process, there will be a leakage current path in A/D block. You must use STOP or IDLE mode after ADC operation is finished.

11.2 A/D CONVERTER REGISTERS

11.2.1 A/D MODE REGISTER (ADMR)

After reset, the start/stop bit is turned off. You can select only one analog input channel at a time. Other analog input pins (AD0-AD10) can be selected dynamically by manipulating the ADCH(ADMR<3:0>). And the pins not used for analog input can be used for normal I/O function.

ADMR — A/D Mode Register

05H

	7	6	5	4	3	2	1	0	
ADMR	SSBIT	EOC	ADCLK		ADCH				(Initial value : 0000 0000)
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

SSBIT	Start or Stop Bit	0: Stop operation 1: Start operation (the EOC bit is cleared)
EOC	End of Conversion Bit	0: Conversion not complete 1: Conversion complete
ADCLK	A/D Clock Selection Bits	00: fxx/1 01: fxx/2 10: fxx/4 11: fxx/8
ADCH	A/D Input Pin Selection Bits	0000: AD0 0001: AD1 0010: AD2 0011: AD3 0100: AD4 0101: AD5 0110: AD6 0111: AD7 1000: AD8 1001: AD9 1010: AD10 Others: Not available

11.2.2 A/D CONVERTER DATA REGISTER (ADDRH/ADDRL)

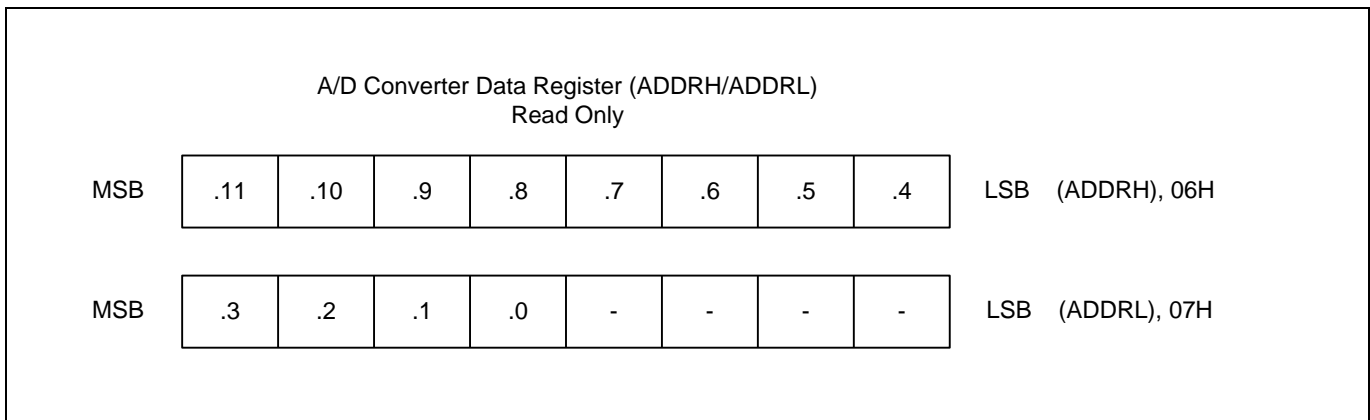


Figure 11-1. A/D Converter Data Register (ADDRH/ADDRL)

11.3 CONVERSION TIMING

The A/D conversion process requires 4 steps (4 clock edges) to convert each bit and 10 clocks to set-up A/D conversion. Therefore, total of 58 clocks are required to complete a 12-bit conversion: When f_{xx}/8 is selected for conversion clock with a 12 MHz f_{xx} clock frequency, one clock cycle is 0.66 μs. Each bit conversion requires 4 clocks, the conversion rate is calculated as follows:

$$4 \text{ clocks/bit} \times 12 \text{ bits} + \text{set-up time} = 58 \text{ clocks}, 58 \text{ clock} \times 0.66 \mu\text{s} = 38.7 \mu\text{s at } 1.5 \text{ MHz (12 MHz/8)}$$

NOTE: The A/D converter needs at least 25 μs for conversion time.

11.4 INTERNAL REFERENCE VOLTAGE LEVELS

In the ADC function block, the analog input voltage level is compared to the reference voltage. The analog input level must be remained within the range AV_{SS} to AV_{REF} .

Different reference voltage levels are generated internally along the resistor tree during the analog conversion process for each conversion step. The reference voltage level for the first conversion bit is always $1/2 AV_{REF}$.

11.5 A/D CONVERTER BLOCK DIAGRAM

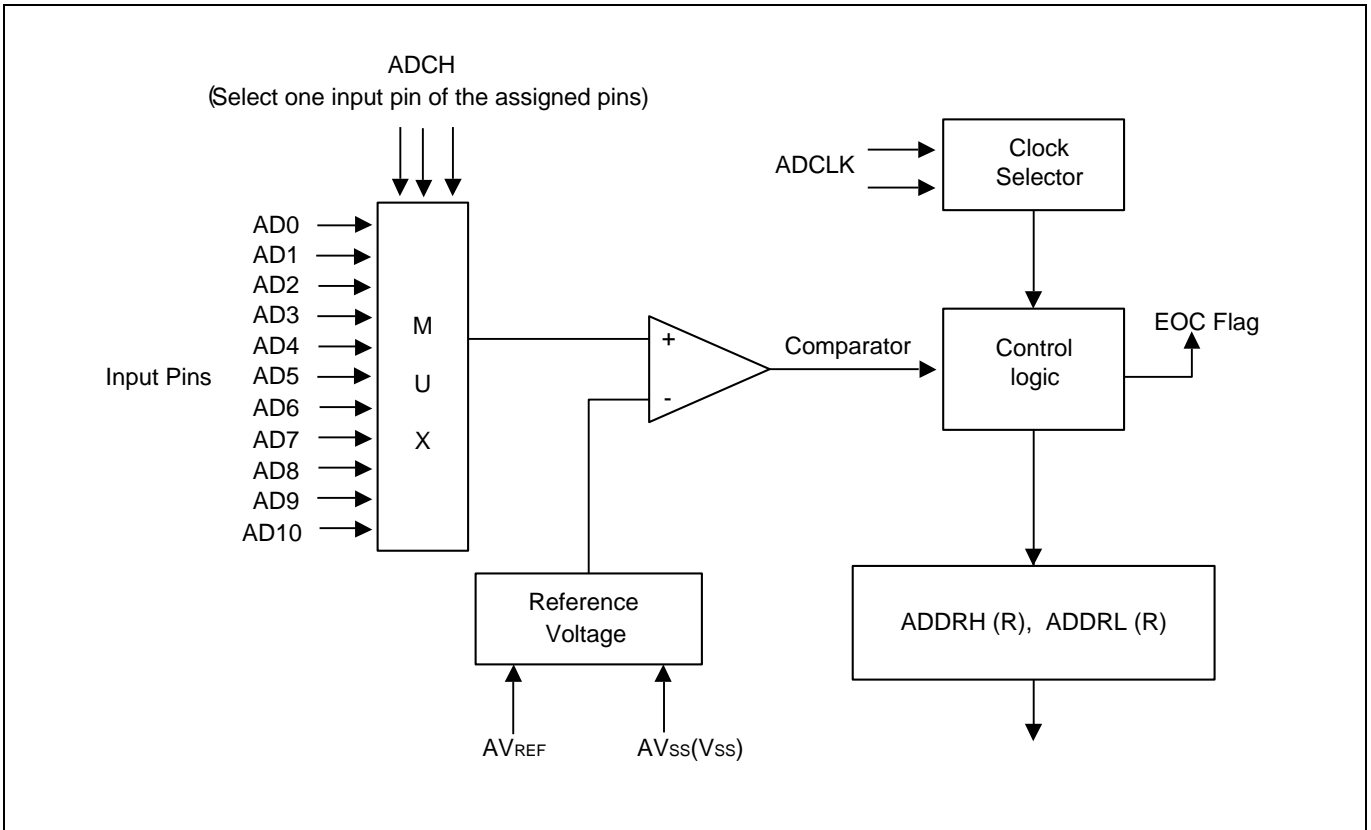


Figure 11-2. A/D Converter Block Diagram

11.6 RECOMMENDED A/D CONVERTER CIRCUIT

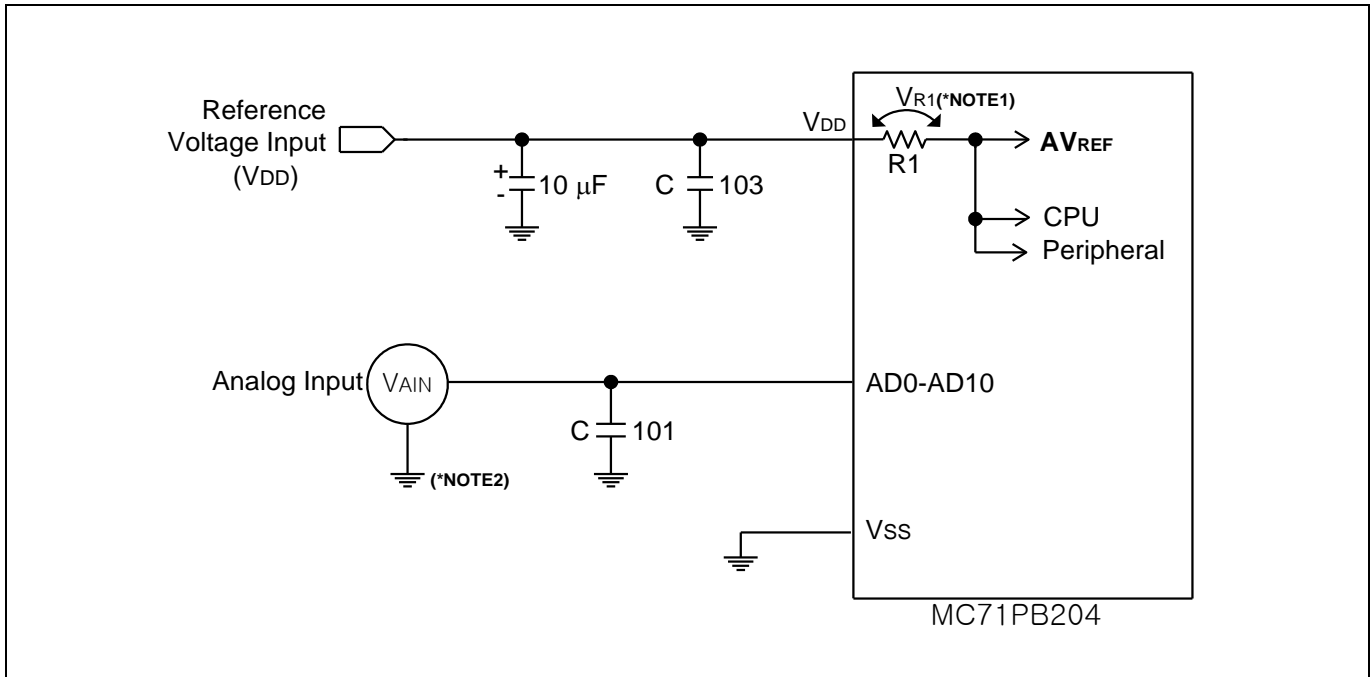


Figure 11-3. Recommended A/D Converter Circuit

NOTES:

1. The voltage supplied to AV_{REF} is dropped as much as "V_{R1}".
2. Lay out the GND of V_{AIN} as close as possible to the power source.

NOTES

12. ELECTRICAL DATA

In this chapter, MC71PB204 electrical characteristics are presented in tables and graphs. The information is arranged in the following order:

- Absolute maximum ratings
- D.C. electrical characteristics
- A.C. electrical characteristics
- Data retention supply voltage in stop mode
- Input/output capacitance
- LVR (Low Voltage Reset) electrical characteristics
- A/D converter electrical characteristics
- Main clock oscillator characteristics
- Main oscillator stabilization time
- External RC oscillation characteristics
- Internal RC oscillation characteristics

12.1 ELECTRICAL CHARACTERISTICS

12.1.1 ABSOLUTE MAXIMUM RATINGS

($T_A = 25^\circ\text{C}$)

Parameter	Symbol	Rating	Units	Conditions
Supply Voltage	V_{DD}	- 0.3 to + 6.5	V	–
Input Voltage	V_{IN}	- 0.3 to $V_{DD} + 0.3$	V	Ports 0 – 2
Output Voltage	V_{OUT}	- 0.3 to $V_{DD} + 0.3$	V	–
Output Current High	I_{OUTH}	- 15	mA	One I/O pin active
		- 60		All I/O pins active
Output Current Low	I_{OUTL}	+ 30 (Peak value)	mA	One I/O pin active
		+ 100 (Peak value)		Total pin current for ports
Operating Temperature	T_{OPR}	- 20 to + 85	$^\circ\text{C}$	–
Storage Temperature	T_{STG}	- 65 to + 150	$^\circ\text{C}$	–

12.1.2 D.C. ELECTRICAL CHARACTERISTICS

($T_A = -20^\circ\text{C}$ to $+85^\circ\text{C}$, $V_{DD} = 2.4\text{ V}$ to 5.5 V)

Parameter	Symbol	Min	Typ.	Max	Units	Conditions
Operating Voltage	V_{DD}	2.4	–	5.5	V	$f_x = 0.4 - 4.0\text{MHz}$
		2.7	–	5.5		$f_x = 0.4 - 8.0\text{MHz}$
		4.0	–	5.5		$f_x = 0.4 - 12.0\text{MHz}$
Input High Voltage	V_{IH1}	$0.8V_{DD}$	–	V_{DD}	V	nMCLR, Ports 0 – 2
	V_{IH2}	$V_{DD}-0.1$	–	V_{DD}		X_{IN} , X_{OUT}
Input Low Voltage	V_{IL1}	V_{SS}	–	$0.2V_{DD}$	V	nMCLR, Ports 0 – 2
	V_{IL2}	V_{SS}	–	0.1		X_{IN} , X_{OUT}
Output High Voltage	V_{OH}	$V_{DD} - 1.0$	–	–	V	$V_{DD} = 4.5\text{ V}$ to 5.5 V All output ports ; $I_{OH} = -2\text{ mA}$
Output Low Voltage	V_{OL1}	–	–	2.0	V	$V_{DD} = 4.5\text{ V}$ to 5.5 V All output ports except V_{OL2} ; $I_{OL} = 15\text{ mA}$
	V_{OL2}	–	–	2.0	V	$V_{DD} = 4.5\text{ V}$ to 5.5 V P0.0-P0.3, P1 ; $I_{OL} = 25\text{ mA}$

12.1.2 D.C. ELECTRICAL CHARACTERISTICS (CONCLUDED)

($T_A = -20\text{ }^\circ\text{C}$ to $+85\text{ }^\circ\text{C}$, $V_{DD} = 2.4\text{ V}$ to 5.5 V)

Parameter	Symbol	Min	Typ	Max	Units	Conditions		
Input high Leakage Current	I_{LIH1}	–	–	3	uA	$V_I = V_{DD}$ All input pins except for I_{LIH2}		
	I_{LIH2}	–	–	20		$V_I = V_{DD}$ X_{IN}, X_{OUT}		
Output High Leakage Current	I_{LOH}	–	–	3	uA	$V_O = V_{DD}$ All output pins		
Input Low Leakage Current	I_{LIL1}	–	–	-3	uA	$V_I = 0\text{ V}$ All input pins except for I_{LIH2}		
	I_{LIL2}	–	–	-20		$V_I = 0\text{ V}$ X_{IN}, X_{OUT}		
Output Low Leakage Current	I_{LOL}	–	–	-3	uA	$V_O = 0\text{ V}$ All output pins		
Pull-Up Resistors	R_{PU}	25	47	100	k Ω	$V_{DD} = 5\text{ V}$	$V_I = 0\text{ V}; T_A = 25\text{ }^\circ\text{C}$, Ports 0 – 2	
		50	95	200		$V_{DD} = 3\text{ V}$		
Pull-Down Resistors	R_{PD}	25	47	100	k Ω	$V_{DD} = 5\text{ V}$	$V_I = 0\text{ V}; T_A = 25\text{ }^\circ\text{C}$ P0.4, P0.5	
		50	95	200		$V_{DD} = 3\text{ V}$		
Oscillator Feed Back Resistor	R_{OSC}	300	600	1500	k Ω	$V_{DD} = 5\text{ V}$	$T_A = 25\text{ }^\circ\text{C}$ $X_{IN} = V_{DD}, X_{OUT} = 0\text{ V}$	
		600	1200	3000		$V_{DD} = 3\text{ V}$		
Supply Current (1)	I_{OPE}	–	7.0	14.0	mA	12.0 MHz	Operation mode: $V_{DD} = 5\text{ V} \pm 10\%$	$C1 =$ $C2 =$ 22pF
			3.0	6.0		4.0 MHz		
			1.5	3.0		4.0 MHz		
	1.0		2.6	mA	12.0 MHz	Idle mode: $V_{DD} = 5\text{ V} \pm 10\%$	$C1 =$ $C2 =$ 22pF	
	0.8		1.6		4.0 MHz			
	0.2		0.6		4.0 MHz			
	I_{STOP}		uA	0.5	5.0	$V_{DD} = 5\text{ V} \pm 10\%$		Stop mode: $T_A = 25\text{ }^\circ\text{C}$
				0.5	3.0			
				–	15.0	$V_{DD} = 5\text{ V} \pm 10\%$		Stop mode: $T_A = -20\text{ }^\circ\text{C}$ to $70\text{ }^\circ\text{C}$
				–	6.0		$V_{DD} = 3\text{ V} \pm 10\%$	

NOTES:

- Supply current does not include current drawn through internal pull-up resistors, PWM, and ADC.
- I_{STOP} is current when main clock stops.

12.1.3 A.C ELECTRICAL CHARACTERISTICS

($T_A = -20^{\circ}C + 85^{\circ}C$, $V_{DD} = 2.4V$ to $5.5V$)

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
Interrupt input, High, Low width	t_{INTH} , t_{INTL}	200	–	–	ns	All interrupt $V_{DD} = 5V$
nMCLR input Low width	t_{RSL}	10	–	–	μs	Input $V_{DD} = 5V$

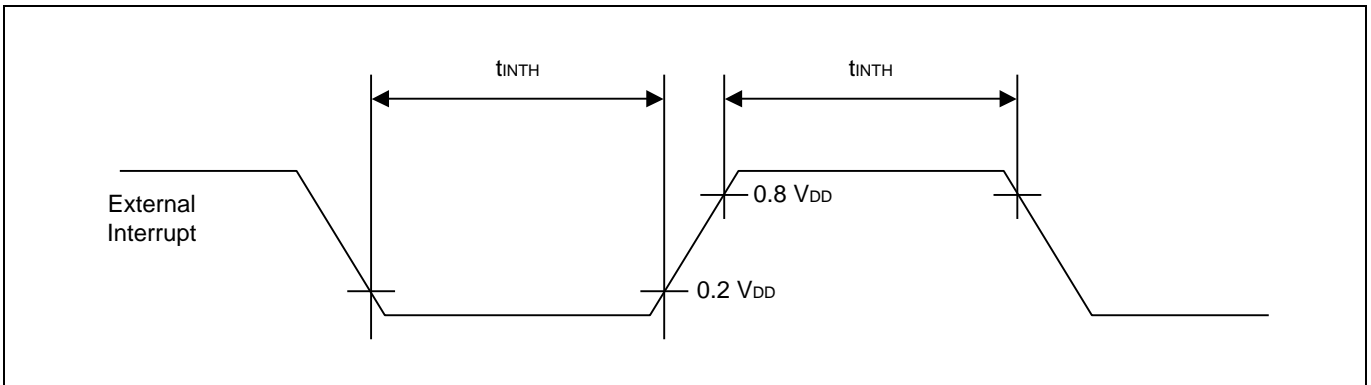


Figure 12-1. Input Timing for External Interrupts

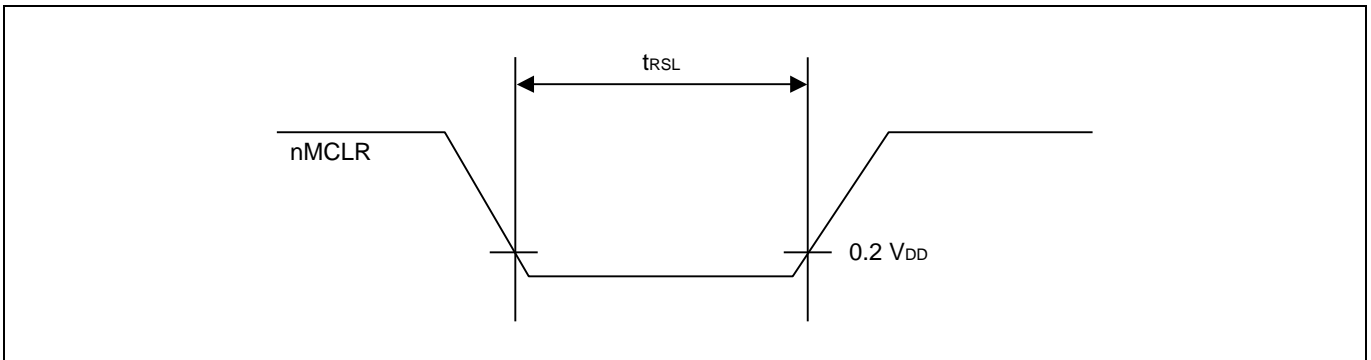


Figure 12-2. Input Timing for RESET

12.1.4 INPUT/OUTPUT CAPACITANCE

($T_A = -20^{\circ}C$ to $+85^{\circ}C$, $V_{DD} = 0V$)

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
Input capacitance	C_{IN}	–	–	10	pF	$f = 1MHz$; unmeasured pins are connected to V_{SS}
Output capacitance	C_{OUT}					
I/O capacitance	C_{IO}					

12.1.5 DATA RETENTION SUPPLY VOLTAGE IN STOP MODE

($T_A = -20\text{ }^\circ\text{C}$ to $+85\text{ }^\circ\text{C}$, $V_{DD} = 2.4\text{ V}$ to 5.5 V)

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
Data retention supply voltage	V_{DDDR}	2.4	–	5.5	V	–
Data retention supply current	I_{DDDR}	–	–	1	μA	$V_{DDDR} = 2.4\text{ V}$ ($T_A = 25\text{ }^\circ\text{C}$) Stop mode

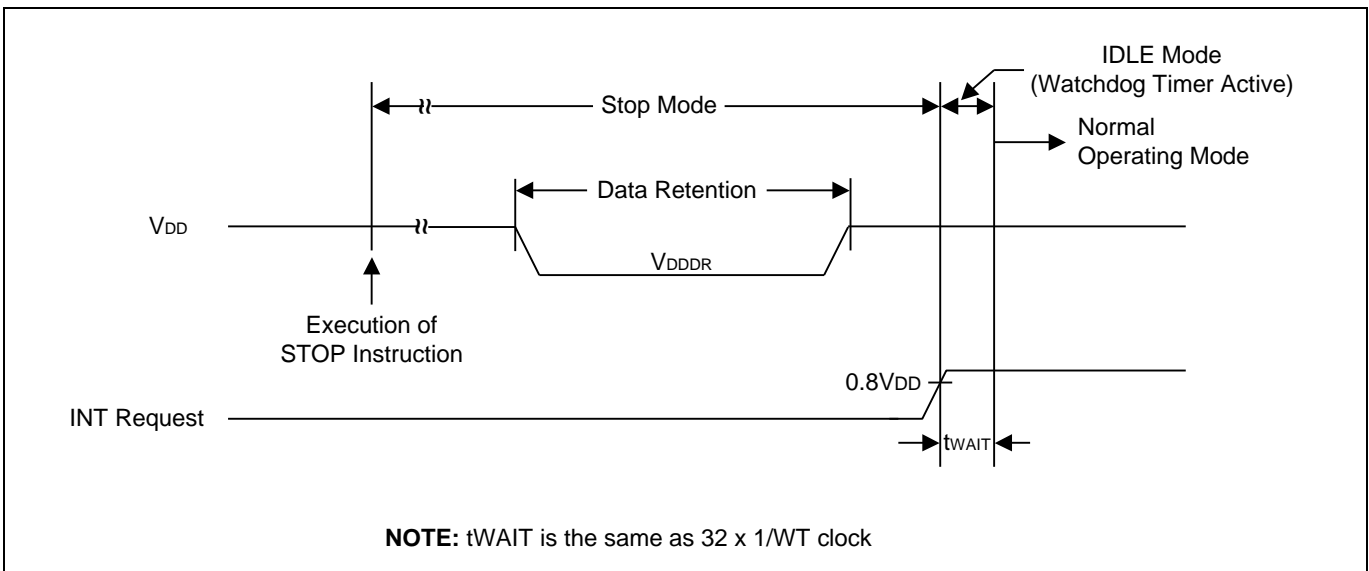


Figure 12-3. Stop Mode Release Timing When Initiated by an Interrupt

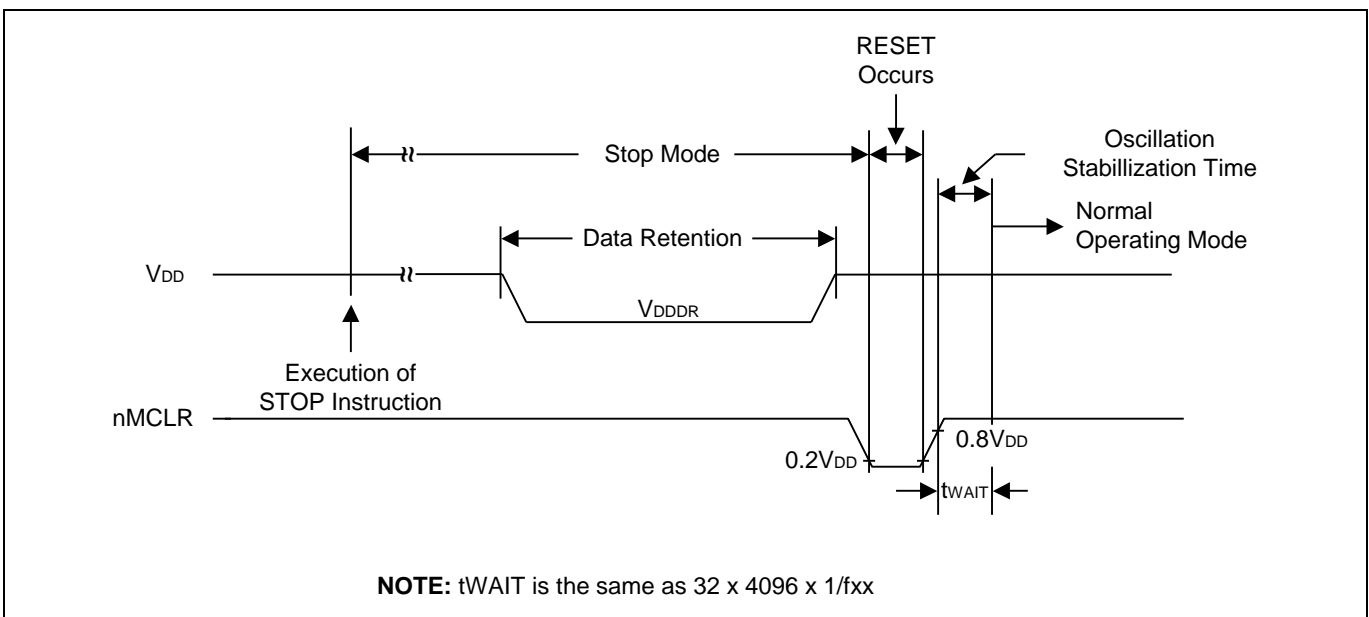


Figure 12-4. Stop Mode Release Timing When Initiated by a RESET

12.1.6 LVR (LOW VOLTAGE RESET) ELECTRICAL CHARACTERISTICS

($T_A = -20\text{ }^\circ\text{C}$ to $+85\text{ }^\circ\text{C}$, $V_{DD} = 2.4\text{ V}$ to 5.5 V)

Parameter	Symbol	Min	Typ	Max	Unit	Condition
LVR voltage	VLVR	2.4 2.7 3.6	2.6 3.0 4.0	2.8 3.3 4.4	V	–
V_{DD} voltage rising time	t_R	10	–	(note2)	μS	–
V_{DD} voltage off time	t_{OFF}	0.5	–	–	S	–
Hysteresys voltage of LVR	ΔV	–	10	100	mV	–
Current consumption	ILVR	–	45	80	μA	$V_{DD} = 3\text{V}$

NOTES:

1. The current of LVR circuit is consumed when LVR is enabled by "ROM Option".
2. $2^{16}/f_x$ (= 6.55 ms at $f_x = 10\text{ MHz}$)

12.1.7 A/D CONVERTER ELECTRICAL CHARACTERISTICS

($T_A = -20\text{ }^\circ\text{C}$ to $+85\text{ }^\circ\text{C}$, $V_{DD} = 2.7\text{ V}$ to 5.5 V)

Parameter	Symbol	Min	Typ	Max	Units	Conditions
A/D converting Resolution	–	–	12	–	bits	–
Integral Linearity Error	ILE	–	–	± 3	LSB	AVREF = 5.12V(NOTE), $V_{SS} = 0\text{V}$, $T_A = +25\text{ }^\circ\text{C}$
Differential Linearity Error	DLE	–	–	± 2		
Offset Error of Top	EOT	–	± 1	± 3		
Offset Error of Bottom	EOB	–	± 1	± 3		
A/D conversion time	t_{CON}	25	–	–	μs	–
Analog input voltage	V_{IAN}	V_{SS}	–	V_{DD}	V	–
Analog input impedance	R_{AN}	2	1000	–	$\text{M}\Omega$	$V_{DD} = 5\text{ V}$
Analog input current	I_{ADIN}	–	–	10	μA	$V_{DD} = 5\text{ V}$
Analog block current	I_{ADC}	–	1	3	mA	$V_{DD} = 5\text{V}$
		–	0.5	1.5	mA	$V_{DD} = 3\text{V}$

NOTE: Refer to the figure 11-3 in the page 113.

12.1.8 MAIN CLOCK OSCILLATOR CHARACTERISTICS

($T_A = -20\text{ }^\circ\text{C}$ to $+85\text{ }^\circ\text{C}$, $V_{DD} = 2.4\text{ V}$ to 5.5 V)

Oscillator	Parameter	Min	Typ	Max	Units	Condition
Crystal	Main oscillation frequency	0.4	–	4	MHz	2.4 V – 5.5 V
		0.4	–	8		2.7 V – 5.5 V
		0.4	–	12		4.0 V – 5.5 V
Ceramic Oscillator	Main oscillation frequency	0.4	–	4	MHz	2.4 V – 5.5 V
		0.4	–	8		2.7 V – 5.5 V
		0.4	–	12		4.0 V – 5.5 V
External Clock	X_{IN} input frequency	0.4	–	4	MHz	2.4 V – 5.5 V
		0.4	–	8		2.7 V – 5.5 V
		0.4	–	12		4.0 V – 5.5 V

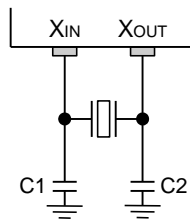


Figure 12-5. Crystal/Ceramic Oscillator

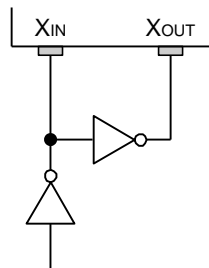


Figure 12-6. External Clock

12.1.9 MAIN OSCILLATION STABILIZATION TIME

($T_A = -20\text{ }^\circ\text{C}$ to $+85\text{ }^\circ\text{C}$, $V_{DD} = 2.4\text{ V}$ to 5.5 V)

Oscillator	Min	Typ	Max	Unit	Condition
Crystal	–	–	60	ms	$f_x > 1\text{ MHz}$
Ceramic	–	–	10	ms	Oscillation stabilization occurs when V_{DD} is equal to the minimum oscillator voltage range.
External clock	41.7	–	1250	ns	X_{IN} input High and Low width (t_{XH} , t_{XL})

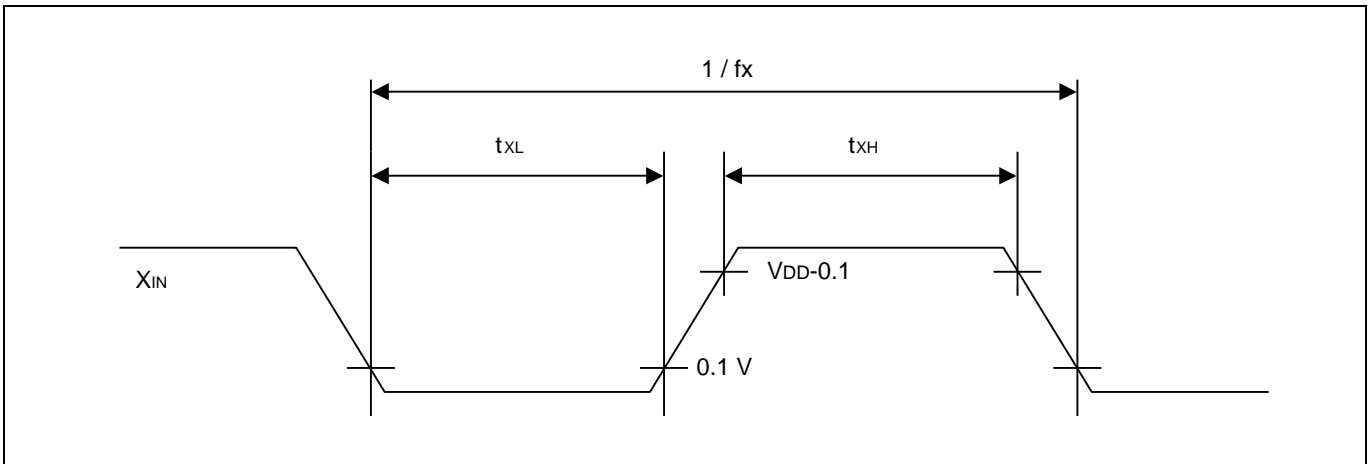


Figure 12-7. Clock Timing Measurement at X_{IN}

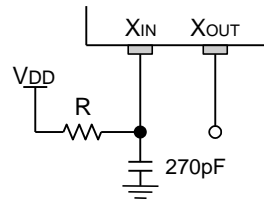
12.1.10 EXTERNAL RC OSCILLATION CHARACTERISTICS

($T_A = -10\text{ }^\circ\text{C}$ to $+70\text{ }^\circ\text{C}$, $V_{DD} = 2.4\text{ V}$ to 5.5 V)

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
RC oscillator frequency Range (1)	fERC	1	–	8	MHz	$T_A = 25\text{ }^\circ\text{C}$
Accuracy of RC Oscillation (2)	ACCERC	– 6	–	+ 6	%	$V_{DD} = 3.3\text{ V}$, $T_A = 25\text{ }^\circ\text{C}$
		– 12	–	+ 12		$V_{DD} = 3.3\text{ V}$, $T_A = -10\text{ }^\circ\text{C}$ to $+70\text{ }^\circ\text{C}$
RC oscillator setup time (3)	tSUERC	–	–	10	msec	$T_A = 25\text{ }^\circ\text{C}$

NOTES:

- The external resistor is connected between V_{DD} and X_{IN} pin and the 270pF capacitor is connected between X_{IN} and V_{SS} pin. (X_{OUT} pin can be used normal port, P0.4)
The frequency is adjusted by external resistor.
- The min/max frequencies are within the range of RC OSC frequency (1MHz to 8MHz)
- Data based on characterization results, not tested in production



NOTE: X_{OUT} pin can be used normal port, P0.4.

Figure 12-8. External RC Oscillator

12.1.11 INTERNAL RC OSCILLATION CHARACTERISTICS

($T_A = -10\text{ }^\circ\text{C}$ to $+70\text{ }^\circ\text{C}$, $V_{DD} = 2.4\text{ V}$ to 5.5 V)

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
RC oscillator frequency (1)	f _{IRC}	6.4	8	9.6	MHz	$V_{DD} = 3.3\text{V}$, $T_A = 25\text{ }^\circ\text{C}$
		6	8	10		$V_{DD} = 3.3\text{V}$, $T_A = -10\text{ }^\circ\text{C}$ to $70\text{ }^\circ\text{C}$
Clock duty ratio	T _{OD}	40	50	60	%	—
RC oscillator setup time (2)	t _{SUIRC}	—	—	10	msec	$T_A = 25\text{ }^\circ\text{C}$

NOTES:

1. Data based on characterization results, not tested in production
2. X_{IN} and X_{OUT} pins can be used normal ports, P0.4 and P0.5.

12.1.12 OPERATING VOLTAGE RANGE

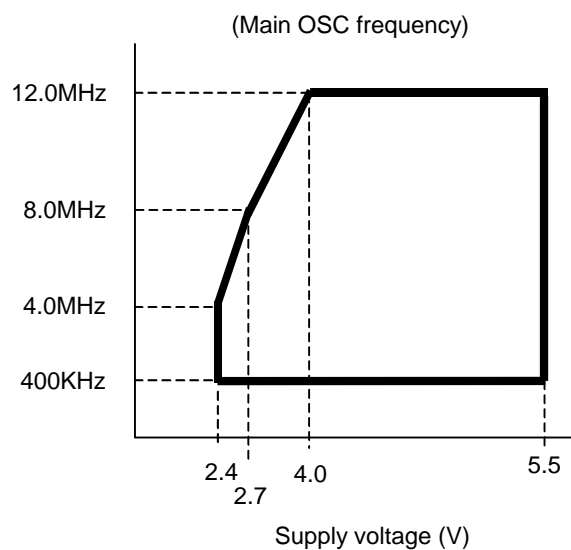


Figure 12-9. Operating Voltage Range

NOTES

13. MECHANICAL DATA

The MC71PB204 microcontroller is currently available in 20-DIP/20-SOP and 16-DIP/16-SOP package.

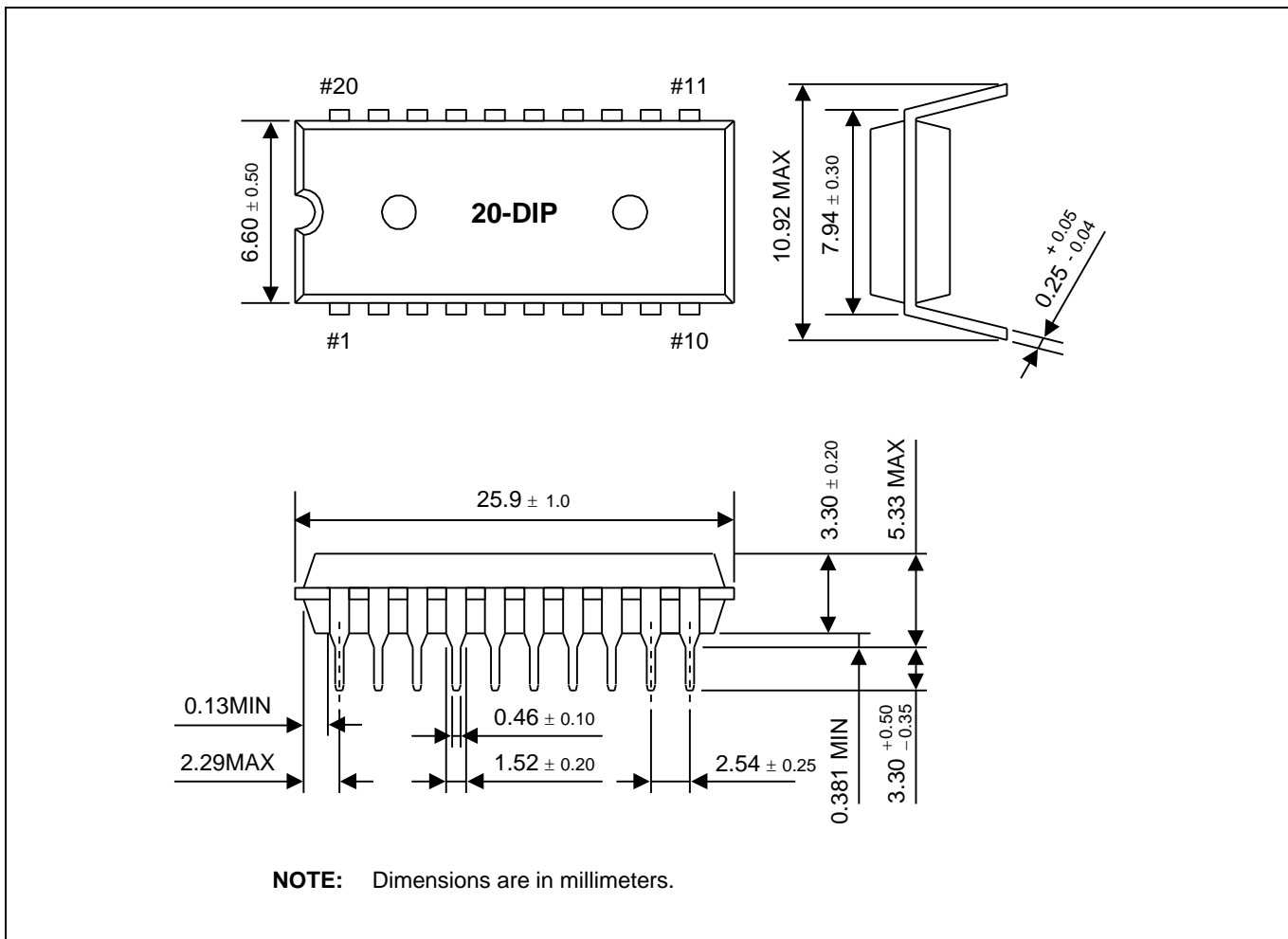
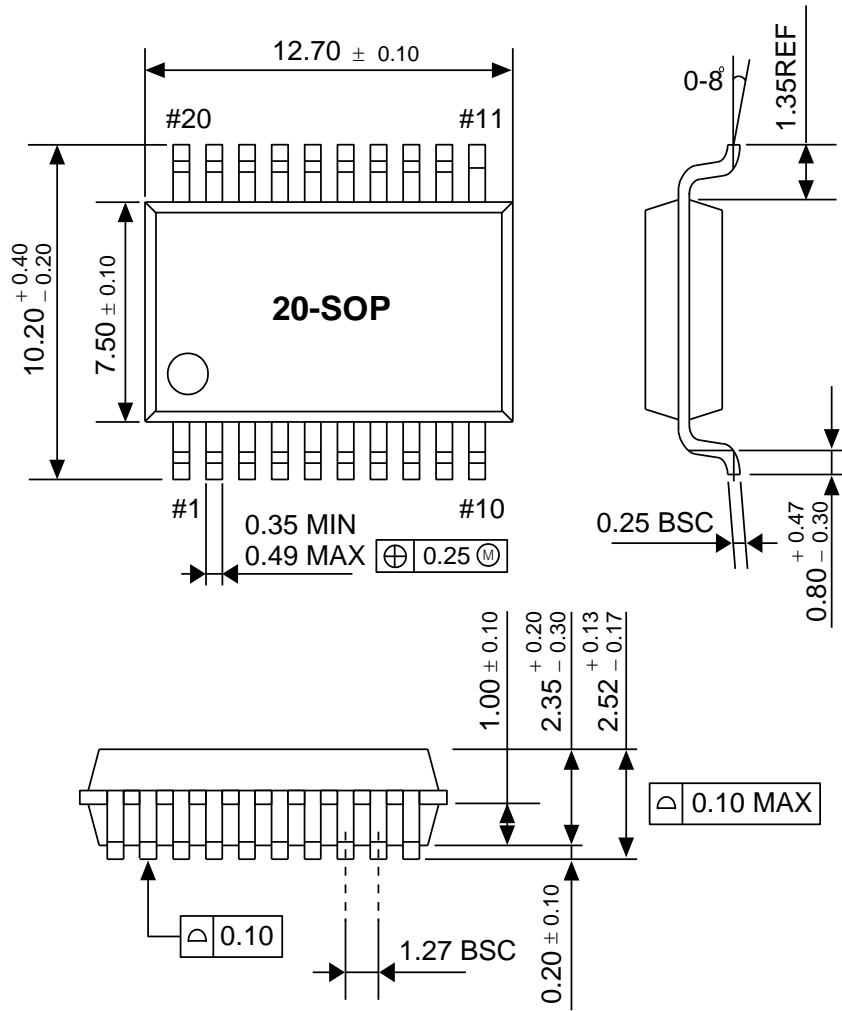
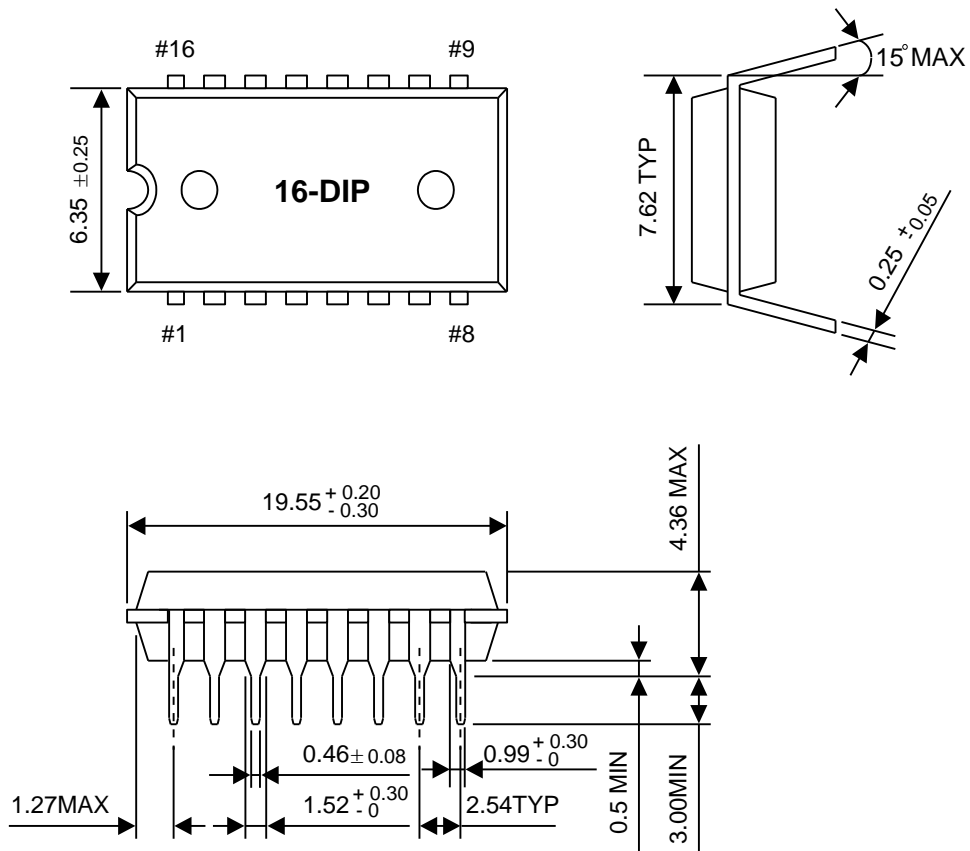


Figure 13-1. 20-DIP Package Dimensions



NOTE: Dimensions are in millimeters.

Figure 13-2. 20-SOP Package Dimensions



NOTE: Dimensions are in millimeters.

Figure 13-3. 16-DIP-300A Package Dimensions

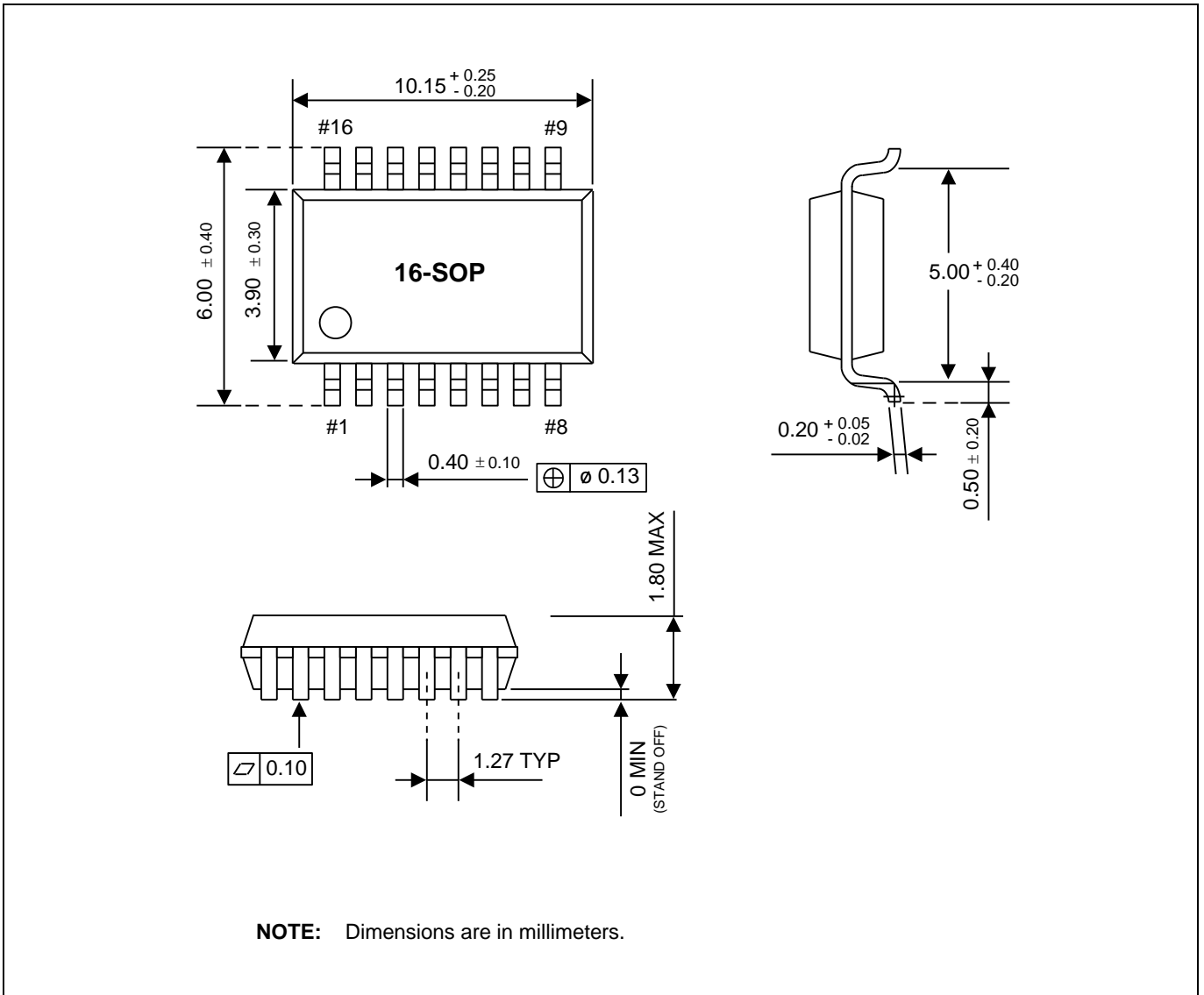


Figure 13-4. 16-SOP-375 Package Dimensions

14. MC71PB204 OTP

The MC71PB204 single-chip microcontroller is OTP (One Time Programmable). The OTP (EPROM) is accessed by serial data format.

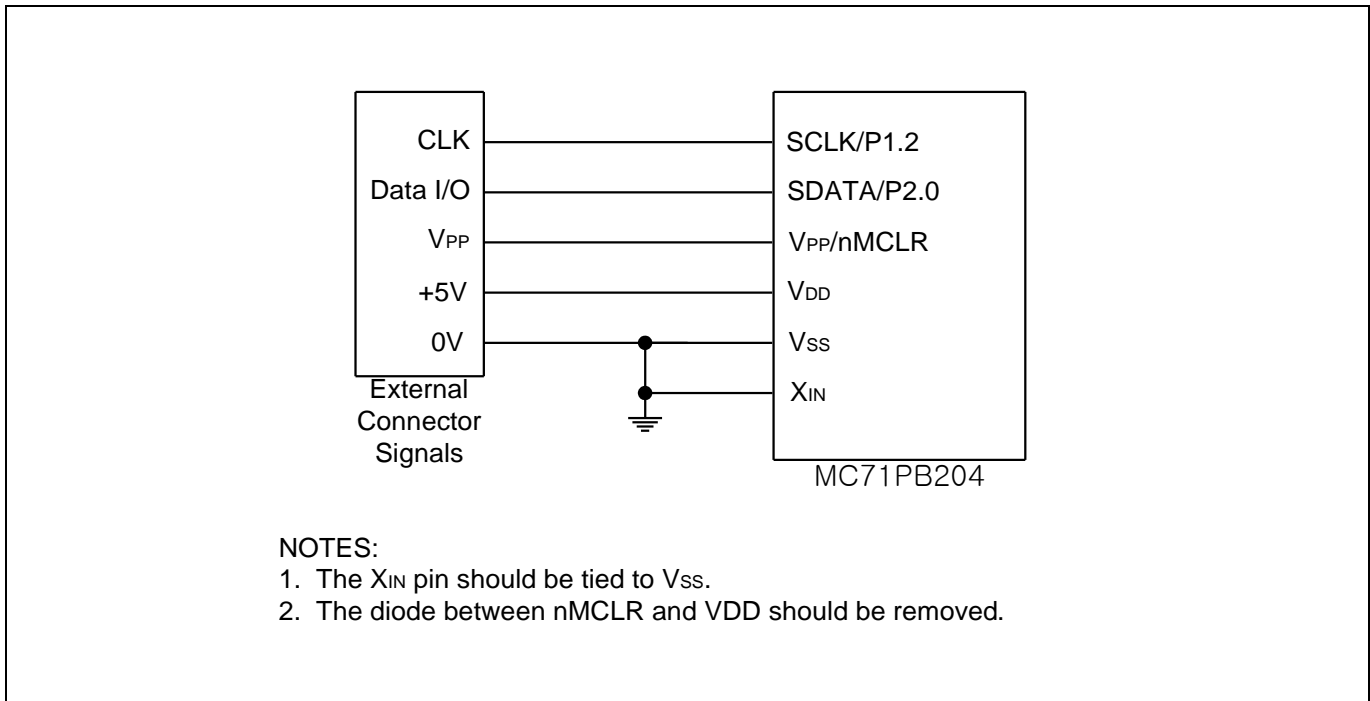


Figure 14-1. OTP Programming Circuit

Table 14-1. Descriptions of Pins Used to Read/Write the EPROM

Main Chip Pin Name	During Programming			
	Pin Name	Pin No.	I/O	Function
P1.2	SCLK	13(11)	I	Serial clock pin. Input only pin.
P2.0	SDATA	14(12)	I/O	Serial data pin. Output port when reading and input port when writing. Can be assigned as a Input/push-pull output port.
nMCLR	V _{PP}	5(3)	I	Power supply pin for EPROM cell writing (indicates that OTP enters into the writing mode). When 12.0 V is applied, OTP is in writing mode and when 5 V is applied, OTP is in reading mode (option).
V _{DD} /V _{SS}	V _{DD} /V _{SS}	15(13)/6(4)	–	Logic power supply pin. V _{DD} should be tied to +5 V during programming.

NOTES:

1. The X_{IN} pin should be tied to V_{SS}.
2. The diode between nMCLR and V_{DD} should be removed.
3. Parentheses indicate pin number for 16-pin package.

15. DEVELOPMENT TOOLS

The GMTTOOLS™ development suite provides a simple and comprehensive environment to develop and debug firmware for GSSP microcontrollers. It runs on Windows operating system, consisting of an assembler (GMASM), a linker (GMLINK), a hardware-assisted debugger (HAD), an evaluation board and a Windows-based debugging software.

15.1 WINDOWS DEBUG SOFTWARE

The Window-based debugger has two different modes: software simulator mode and hardware emulator mode. Those two modes can be used to validate pre-silicon software in simulation mode and to do real-time debugging of developed software inside silicon in the emulation mode. Once loaded, the program may be observed in Source Window, run at full-speed, single stepped by machine or C level instructions, or stopped at any of the breakpoints. The key features of debug software are Processor execution control, Read-Write all processor contents, unlimited number of software breakpoints and hardware execution breakpoints in program and data memory.

15.2 HARDWARE ASSISTED DEBUGGER (HAD)

Hardware Assisted Debugger (HAD) is a hardware adapter managing the communication between the debug engine within the microcontroller and the PC.

15.3 GMASM ASSEMBLER

GMASM is a parts of gmutils software. It includes gmasm, gmlink, and gmlib. Each tool is intended to be an open source replacement for a corresponding Microchip™ tools. GMASM is full featured powerful macro processor and generates fully relocated and mapped listings and support complex expressions involving assembler variables, numbers, labels and strings. GMASM supports all GMC14 products from GSSP.

15.4 GMLINK

The GMLINK linker allows users to easily relocate their ROM image in memory. It supports both COD or COFF object formats.

15.5 EVALUATION BOARDS

Evaluation boards are available for all GMC14 series microcontrollers. All required evaluation system cables and adapters are included in the device-specific evaluation board.

15.6 HOST REQUIREMENTS

A Pentium class computer with minimum 32 MB of memory, 10 MB of free space on Hard Disk, CD-ROM drive, RS232 serial port, and Windows® 98/Me or Windows® NT/2000/XP operating system are required.

15.7 MC71PB204 DEBUGGER SYSTEM

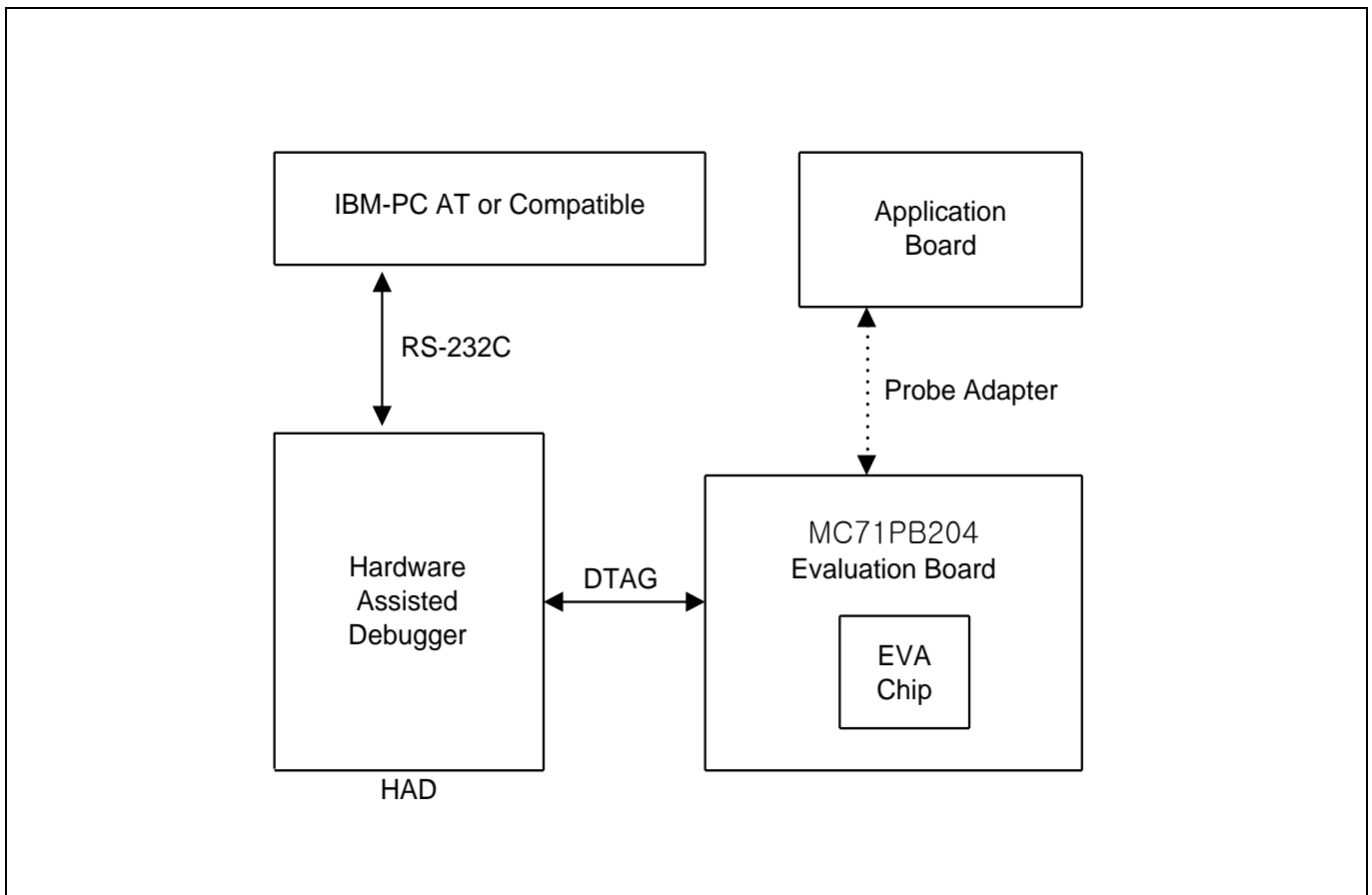


Figure 15-1. MC71PB204 Debugger System

15.8 MC71PB204 EVALUATION BOARD

The MC71PB204 evaluation board is used for the MC71PB204 microcontroller. It is supported by the HAD development system.

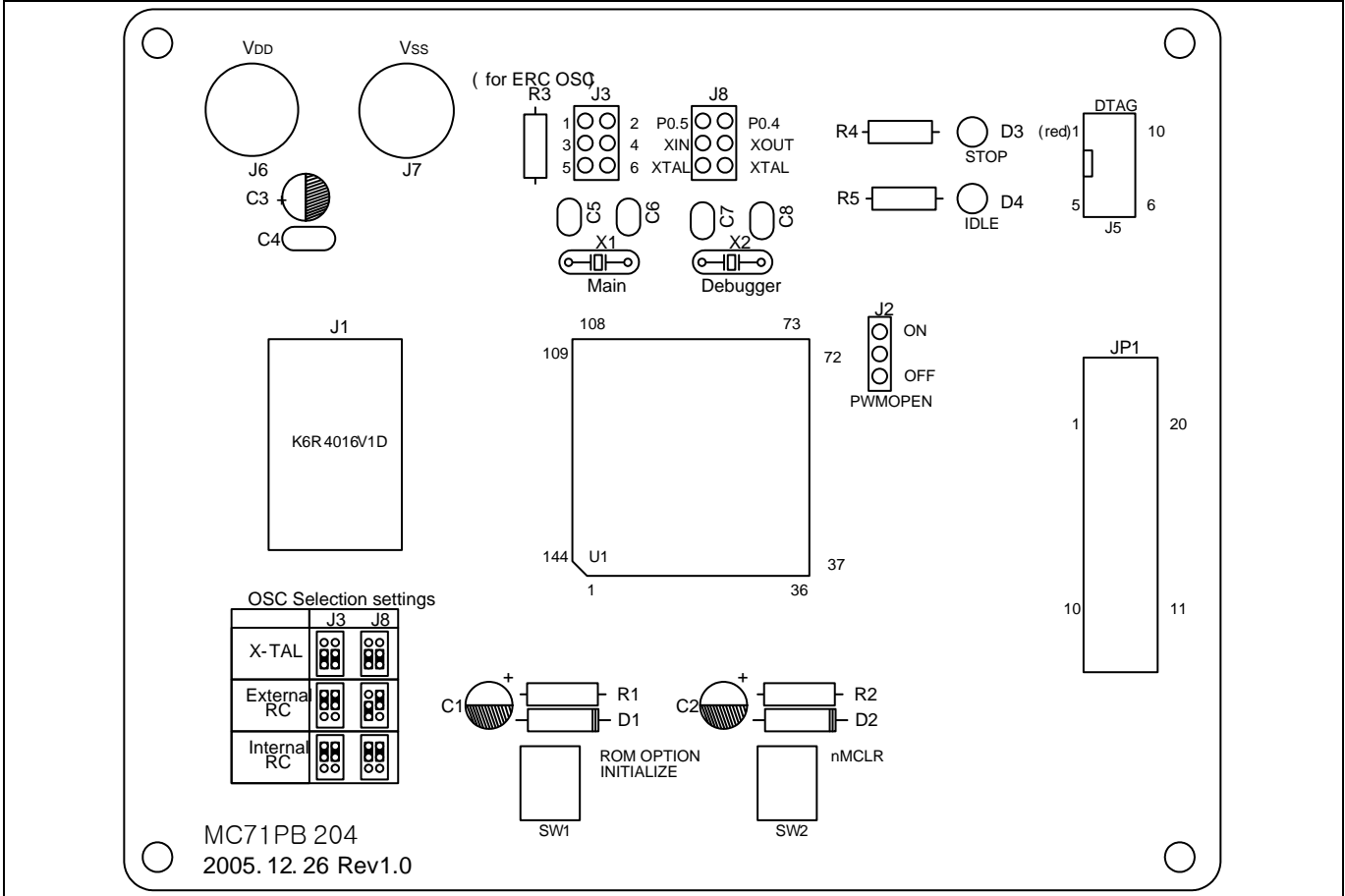


Figure 15-2. MC71PB204 Evaluation Board Configuration

Table 15-1. 10pin-DTAG(J5) Pin Description

Pin Number	Pin Name	Description	Pin Number	Pin Name	Description
1	DCK	Clock signal	6	GND	Signal ground
2	DDI	Data input	7	VDD	Power supply
3	N.C.	Not connected	8	VDD	Power supply
4	N.C.	Not connected	9	N.C.	Not connected
5	DDO	Data output	10	GND	Signal ground

Table 15-2. Main Oscillation Selection Settings for MC71PB204 Evaluation Board

Main OSC Selection Settings	Operating Mode	Description
<p>J3: 1, 2 closed; 3, 4 open; 5, 6 open. J8: 1, 2 open; 3, 4 closed; 5, 6 closed.</p>	Crystal/Ceramic Oscillator	The ROM option<3:1> value of Crystal/Ceramic OSC is 111b. The P0.4 and P0.5 are used for X _{OUT} and X _{IN} . The 'X1' is an x-tal for the main Crystal/Ceramic.
<p>J3: 1, 2 closed; 3, 4 open; 5, 6 open. J8: 1, 2 open; 3, 4 closed; 5, 6 open.</p>	External RC Oscillation	The ROM option<3:1> value of External RC OSC is 000b. The external resistor (R3) is connected between VDD and X _{IN} (P0.5). The P0.4 (X _{OUT}) is used for normal port. How to use the External RC OSC mode: ① Set “X-tal OSC mode” by J3 and J8 ② Download “External RC mode program code” ③ If a dialog box is shown during the download step, Push “Enter” key ④ Change to “External RC mode” by J3 and J8 ⑤ Download again ⑥ Repeat step “⑤” when download “External RC mode program code” NOTE: The external resistor, 'R3' is needed
<p>J3: 1, 2 closed; 3, 4 open; 5, 6 open. J8: 1, 2 open; 3, 4 closed; 5, 6 open.</p>	Internal RC Oscillation	The ROM option<3:1> value of Internal RC OSC is 001b, 010b, 011b, or 100b. The P0.4 (X _{OUT}) and P0.5 (X _{IN}) are used for normal port. How to use the Internal RC OSC mode: ① Set “X-tal OSC mode” by J3 and J8 ② Download “Internal RC mode program code” ③ Change to “Internal RC mode” by J3 and J8

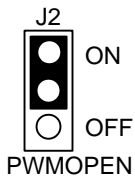
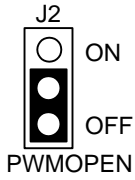
Table 15-3. The EVA Debugger Clock('X2') for MC71PB204 Evaluation Board

EVA Debugger Clock	Description
“X2”	The 'X2' is an x-tal for the EVA debugger clock. The maximum clock frequency is 16MHz. NOTES: 1. The main clock frequency should be set under the EVA debugger clock frequency. 2. The EVA debugger clock is applied when you make a new project of the debugger for the MC71PB204.

Table 15-4. The 'SW1' for MC71PB204 ROM Option Initialize

ROM Option Initialize	Description
"SW1"	<p>The 'SW1' is used when you want to initialize the ROM option. If it is pushed, the ROM option area (001FH of ROM address) has the default value, 3FFFH (disabled LVR, selected X-tal OSC).</p> <p>How to change the "External RC mode" to "X-TAL mode":</p> <ol style="list-style-type: none"> ① Set "X-tal OSC mode" by J3 and J8 ② Push "SW1" ③ Download "X-tal OSC mode program code"

Table 15-5. PWM_OPEN Setting

PWM_OPEN Setting	Description
	In the PWM mode, the PWM Pulse maintains when the debugger is in stop operation or single step operation.
	In the PWM mode, the PWM Pulse stops when the debugger is in stop operation or single step operation.

GREEN LED ('D4')

The Green LED is ON when the evaluation chip is in idle mode or stop mode.

RED LED ('D3')

The Red LED is ON when the evaluation chip is in stop mode.

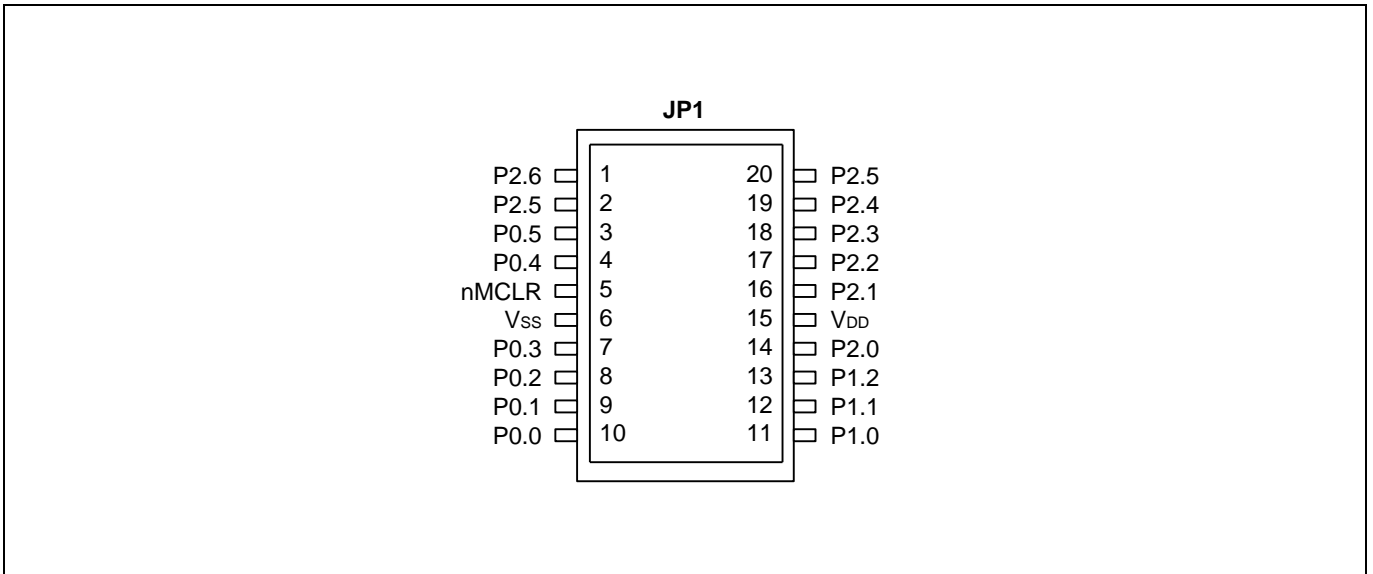


Figure 15-3. Connector 'JP1' for MC71PB204 Evaluation Board

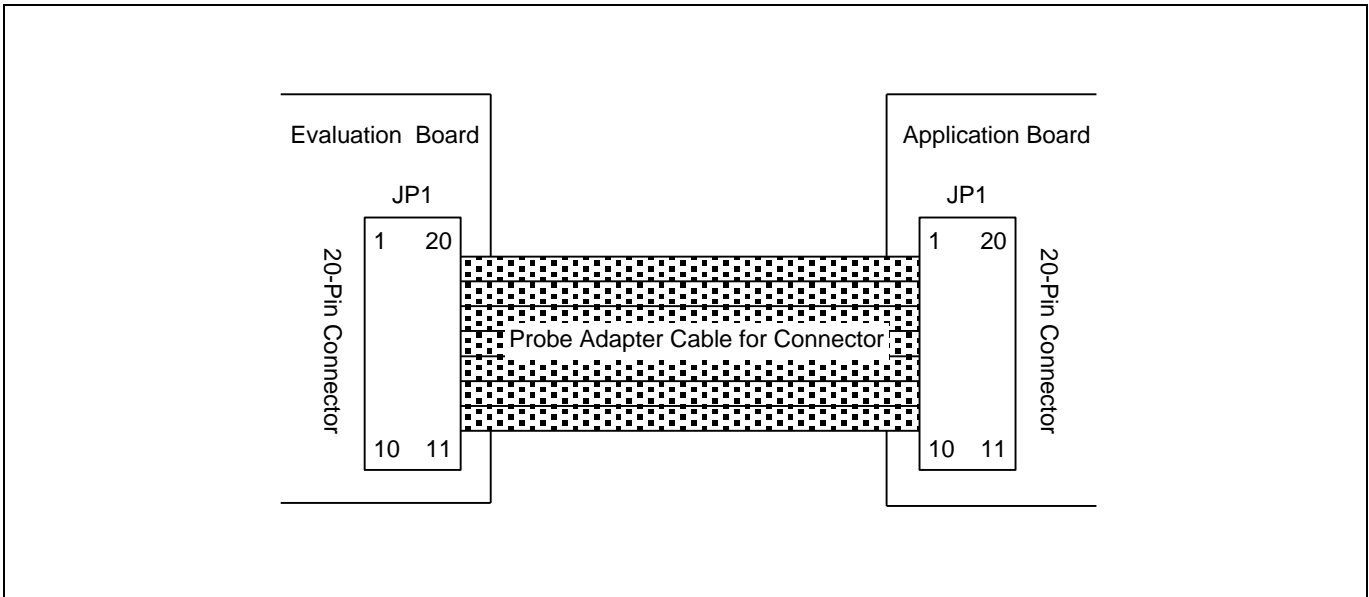


Figure 15-4. MC71PB204 Probe Adapter for 20-DIP, 20-SOP Package

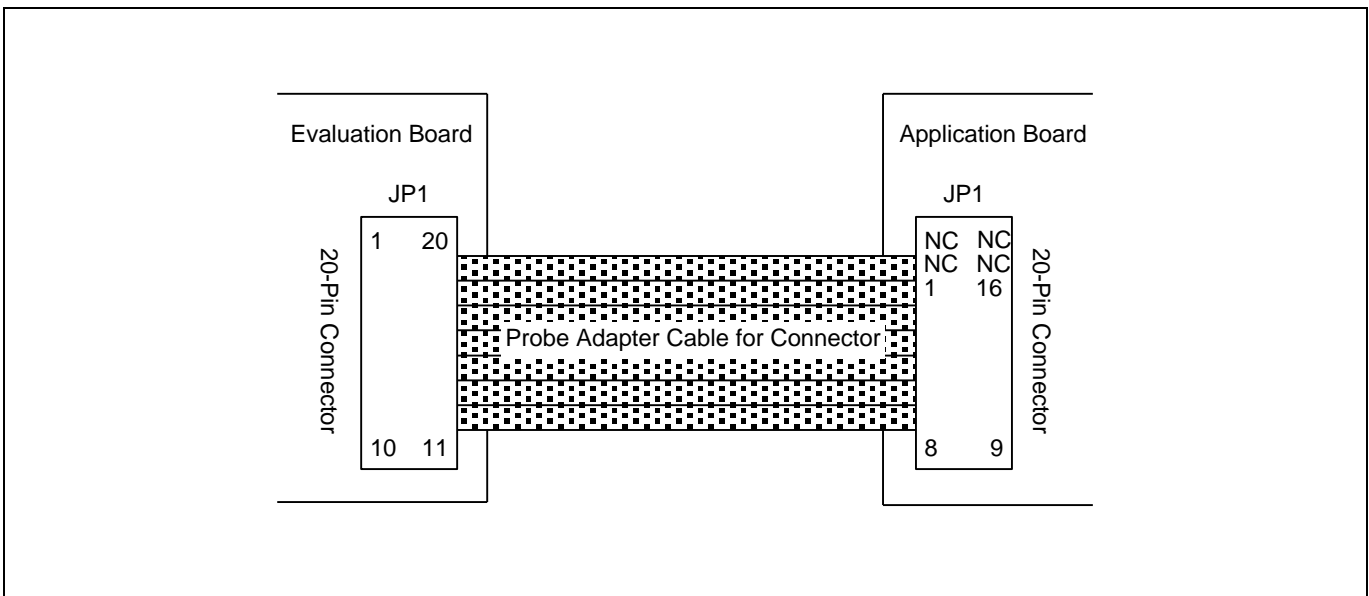


Figure 15-5. MC71PB204 Probe Adapter for 16-DIP, 16-SOP Package

NOTES