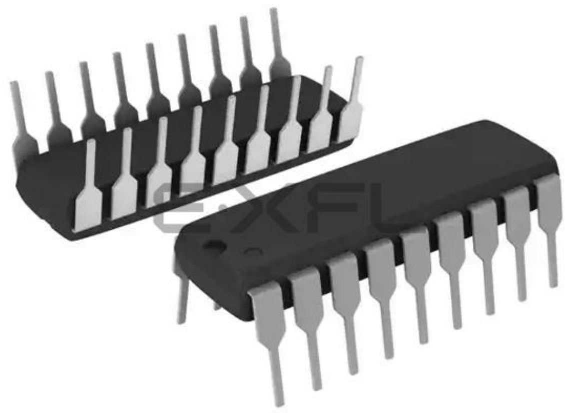


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	4MHz
Connectivity	-
Peripherals	POR, WDT
Number of I/O	13
Program Memory Size	1.75KB (1K x 14)
Program Memory Type	FLASH
EEPROM Size	64 x 8
RAM Size	68 x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 6V
Data Converters	-
Oscillator Type	External
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Through Hole
Package / Case	18-DIP (0.300", 7.62mm)
Supplier Device Package	18-PDIP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16lf84-04-p

4.0 MEMORY ORGANIZATION

There are two memory blocks in the PIC16F8X. These are the program memory and the data memory. Each block has its own bus, so that access to each block can occur during the same oscillator cycle.

The data memory can further be broken down into the general purpose RAM and the Special Function Registers (SFRs). The operation of the SFRs that control the "core" are described here. The SFRs used to control the peripheral modules are described in the section discussing each individual peripheral module.

The data memory area also contains the data EEPROM memory. This memory is not directly mapped into the data memory, but is indirectly mapped. That is, an indirect address pointer specifies the address of the data EEPROM memory to read/write. The 64 bytes of data EEPROM memory have the address range 0h-3Fh. More details on the EEPROM memory can be found in Section 7.0.

4.1 Program Memory Organization

The PIC16FXX has a 13-bit program counter capable of addressing an 8K x 14 program memory space. For the PIC16F83 and PIC16CR83, the first 512 x 14 (0000h-01FFh) are physically implemented (Figure 4-1). For the PIC16F84 and PIC16CR84, the first 1K x 14 (0000h-03FFh) are physically implemented (Figure 4-2). Accessing a location above the physically implemented address will cause a wrap-around. For example, for the PIC16F84 locations 20h, 420h, 820h, C20h, 1020h, 1420h, 1820h, and 1C20h will be the same instruction.

The reset vector is at 0000h and the interrupt vector is at 0004h.

FIGURE 4-1: PROGRAM MEMORY MAP AND STACK - PIC16F83/CR83

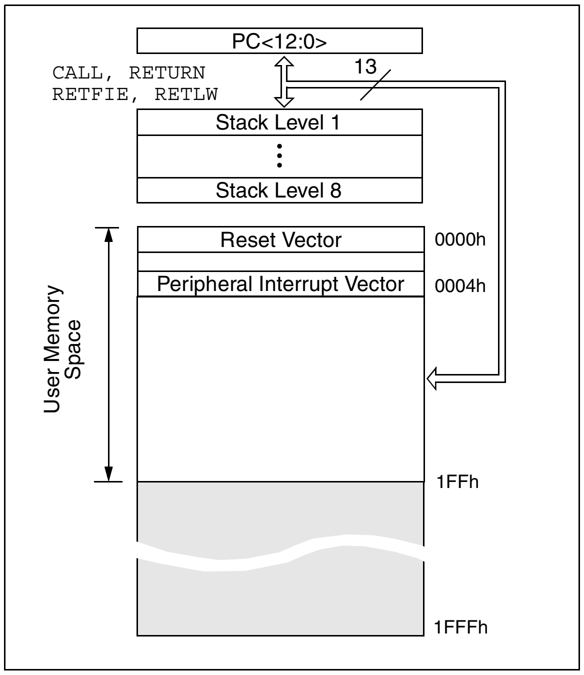
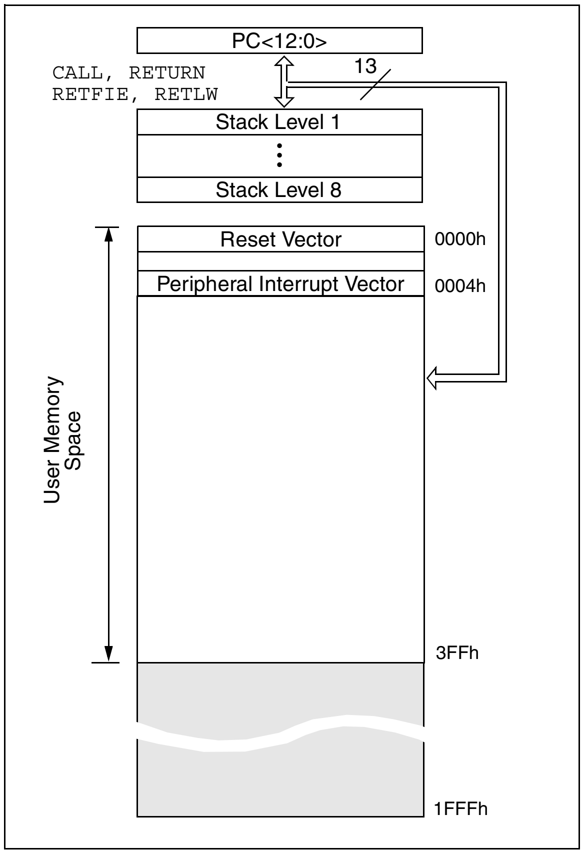


FIGURE 4-2: PROGRAM MEMORY MAP AND STACK - PIC16F84/CR84



4.5 Indirect Addressing; INDF and FSR Registers

The INDF register is not a physical register. Addressing INDF actually addresses the register whose address is contained in the FSR register (FSR is a *pointer*). This is indirect addressing.

EXAMPLE 4-1: INDIRECT ADDRESSING

- Register file 05 contains the value 10h
- Register file 06 contains the value 0Ah
- Load the value 05 into the FSR register
- A read of the INDF register will return the value of 10h
- Increment the value of the FSR register by one (FSR = 06)
- A read of the INDF register now will return the value of 0Ah.

Reading INDF itself indirectly (FSR = 0) will produce 00h. Writing to the INDF register indirectly results in a no-operation (although STATUS bits may be affected).

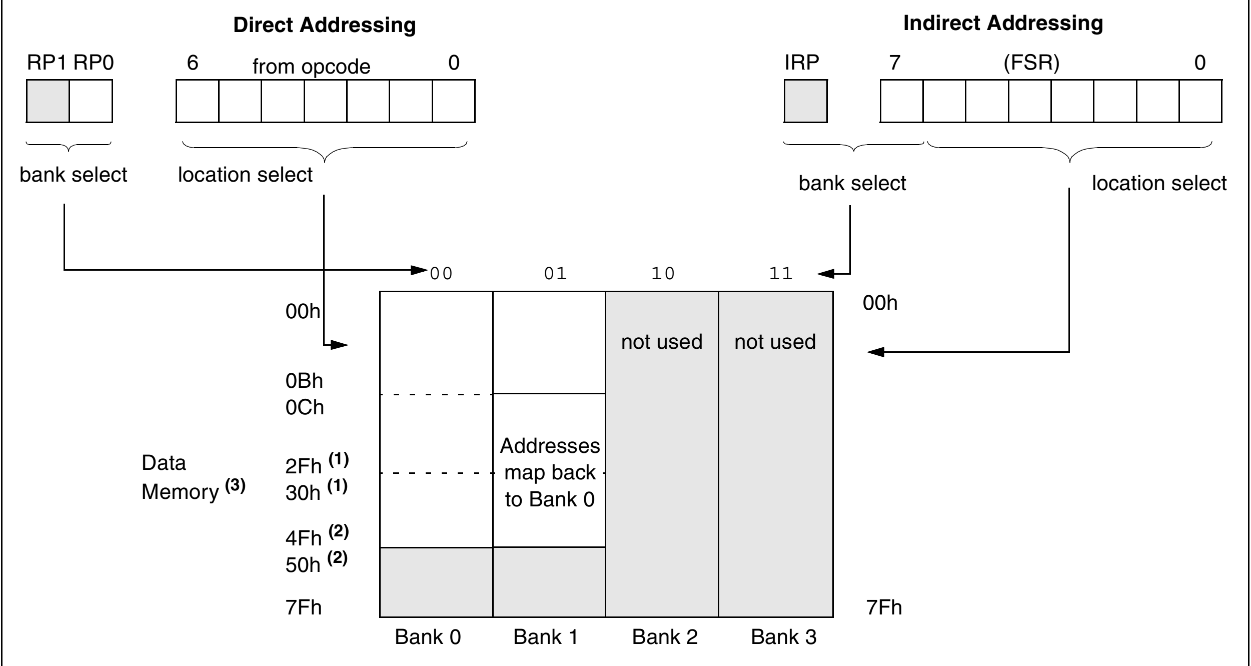
A simple program to clear RAM locations 20h-2Fh using indirect addressing is shown in Example 4-2.

EXAMPLE 4-2: HOW TO CLEAR RAM USING INDIRECT ADDRESSING

```
          movlw  0x20  ;initialize pointer
          movwf  FSR   ;  to RAM
NEXT      clrf   INDF  ;clear INDF register
          incf   FSR   ;inc pointer
          btfss  FSR,4 ;all done?
          goto   NEXT  ;NO, clear next
CONTINUE
          :            ;YES, continue
```

An effective 9-bit address is obtained by concatenating the 8-bit FSR register and the IRP bit (STATUS<7>), as shown in Figure 4-1. However, IRP is not used in the PIC16F8X.

FIGURE 4-1: DIRECT/INDIRECT ADDRESSING



Note 1: PIC16F83 and PIC16CR83 devices.
2: PIC16F84 and PIC16CR84 devices
3: For memory map detail see Figure 4-1.

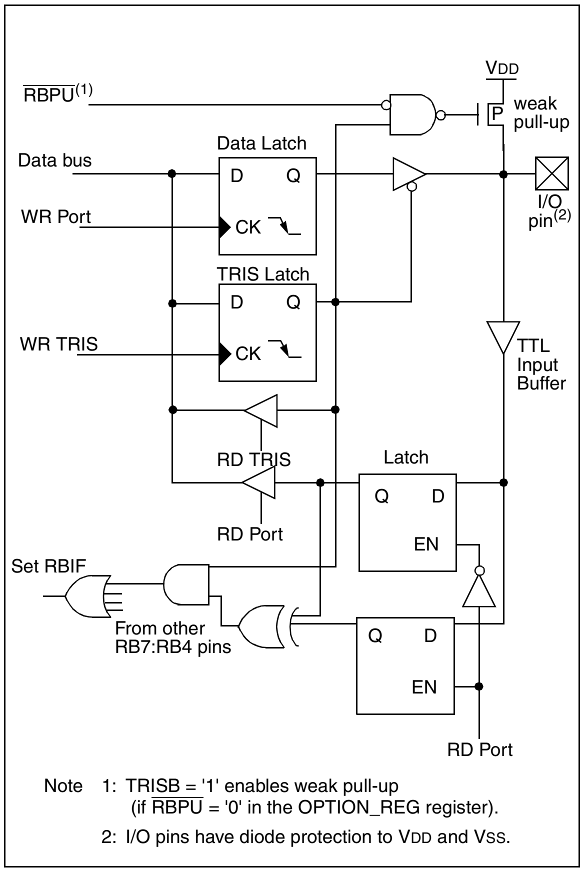
5.2 PORTB and TRISB Registers

PORTB is an 8-bit wide bi-directional port. The corresponding data direction register is TRISB. A '1' on any bit in the TRISB register puts the corresponding output driver in a hi-impedance mode. A '0' on any bit in the TRISB register puts the contents of the output latch on the selected pin(s).

Each of the PORTB pins have a weak internal pull-up. A single control bit can turn on all the pull-ups. This is done by clearing the $\overline{\text{RBPU}}$ (OPTION_REG<7>) bit. The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on a Power-on Reset.

Four of PORTB's pins, RB7:RB4, have an interrupt on change feature. Only pins configured as inputs can cause this interrupt to occur (i.e., any RB7:RB4 pin configured as an output is excluded from the interrupt on change comparison). The pins value in input mode are compared with the old value latched on the last read of PORTB. The "mismatch" outputs of the pins are OR'ed together to generate the RB port change interrupt.

FIGURE 5-3: BLOCK DIAGRAM OF PINS RB7:RB4



Note 1: TRISB = '1' enables weak pull-up (if $\overline{\text{RBPU}}$ = '0' in the OPTION_REG register).
2: I/O pins have diode protection to VDD and VSS.

This interrupt can wake the device from SLEEP. The user, in the interrupt service routine, can clear the interrupt in the following manner:

a) Read (or write) PORTB. This will end the mismatch condition.
b) Clear flag bit RBIF.

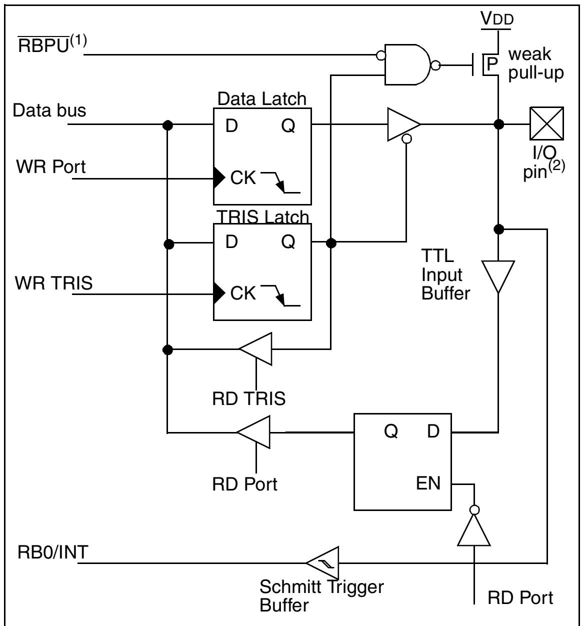
A mismatch condition will continue to set the RBIF bit. Reading PORTB will end the mismatch condition, and allow the RBIF bit to be cleared.

This interrupt on mismatch feature, together with software configurable pull-ups on these four pins allow easy interface to a key pad and make it possible for wake-up on key-depression (see AN552 in the Embedded Control Handbook).

Note 1: For a change on the I/O pin to be recognized, the pulse width must be at least T_{CY} (4/f_{OSC}) wide.

The interrupt on change feature is recommended for wake-up on key depression operation and operations where PORTB is only used for the interrupt on change feature. Polling of PORTB is not recommended while using the interrupt on change feature.

FIGURE 5-4: BLOCK DIAGRAM OF PINS RB3:RB0



Note 1: TRISB = '1' enables weak pull-up (if $\overline{\text{RBPU}}$ = '0' in the OPTION_REG register).
2: I/O pins have diode protection to VDD and VSS.

6.0 TIMER0 MODULE AND TMR0 REGISTER

The Timer0 module timer/counter has the following features:

- 8-bit timer/counter
- Readable and writable
- 8-bit software programmable prescaler
- Internal or external clock select
- Interrupt on overflow from FFh to 00h
- Edge select for external clock

Timer mode is selected by clearing the T0CS bit (OPTION_REG<5>). In timer mode, the Timer0 module (Figure 6-1) will increment every instruction cycle (without prescaler). If the TMR0 register is written, the increment is inhibited for the following two cycles (Figure 6-2 and Figure 6-3). The user can work around this by writing an adjusted value to the TMR0 register.

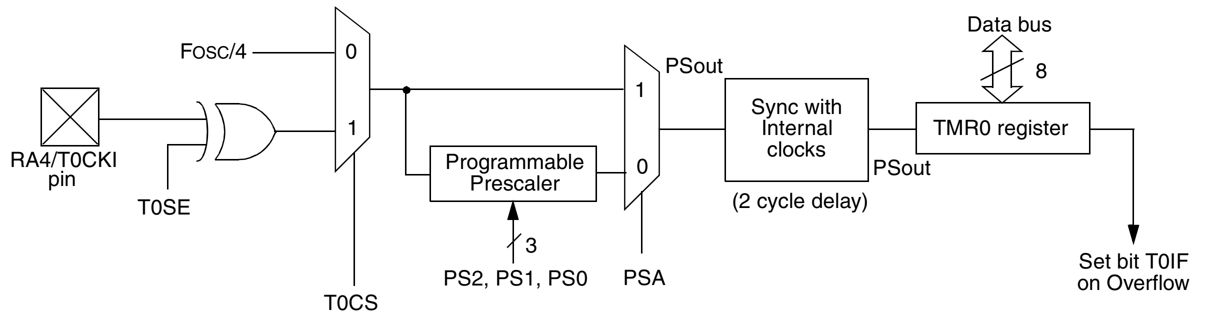
Counter mode is selected by setting the T0CS bit (OPTION_REG<5>). In this mode TMR0 will increment either on every rising or falling edge of pin RA4/T0CKI. The incrementing edge is determined by the T0 source edge select bit, T0SE (OPTION_REG<4>). Clearing bit T0SE selects the rising edge. Restrictions on the external clock input are discussed in detail in Section 6.2.

The prescaler is shared between the Timer0 Module and the Watchdog Timer. The prescaler assignment is controlled, in software, by control bit PSA (OPTION_REG<3>). Clearing bit PSA will assign the prescaler to the Timer0 Module. The prescaler is not readable or writable. When the prescaler (Section 6.3) is assigned to the Timer0 Module, the prescale value (1:2, 1:4, ..., 1:256) is software selectable.

6.1 TMR0 Interrupt

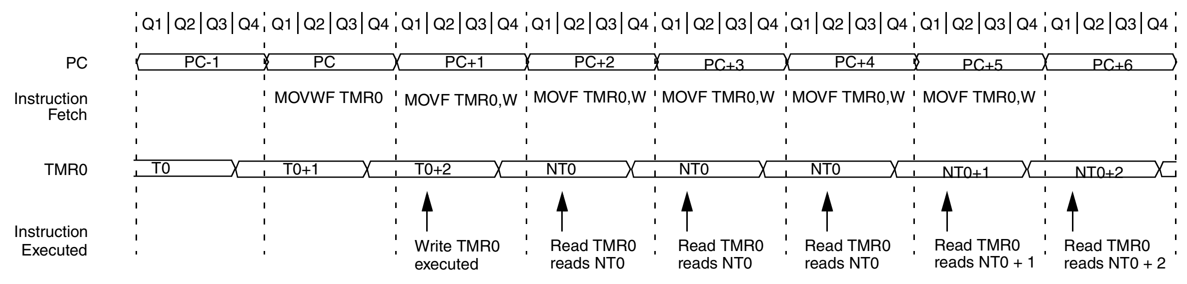
The TMR0 interrupt is generated when the TMR0 register overflows from FFh to 00h. This overflow sets the T0IF bit (INTCON<2>). The interrupt can be masked by clearing enable bit T0IE (INTCON<5>). The T0IF bit must be cleared in software by the Timer0 Module interrupt service routine before re-enabling this interrupt. The TMR0 interrupt (Figure 6-4) cannot wake the processor from SLEEP since the timer is shut off during SLEEP.

FIGURE 6-1: TMR0 BLOCK DIAGRAM



Note 1: Bits T0CS, T0SE, PS2, PS1, PS0 and PSA are located in the OPTION_REG register.
2: The prescaler is shared with the Watchdog Timer (Figure 6-6)

FIGURE 6-2: TMR0 TIMING: INTERNAL CLOCK/NO PRESCALER



6.3.1 SWITCHING PRESCALER ASSIGNMENT

The prescaler assignment is fully under software control (i.e., it can be changed “on the fly” during program execution).

Note:	To avoid an unintended device RESET, the following instruction sequence (Example 6-1) must be executed when changing the prescaler assignment from Timer0 to the WDT. This sequence must be taken even if the WDT is disabled. To change prescaler from the WDT to the Timer0 module use the sequence shown in Example 6-2.

EXAMPLE 6-1: CHANGING PRESCALER (TIMER0→WDT)

```
BCF      STATUS, RP0  ;Bank 0
CLRF     TMR0         ;Clear TMR0
                      ; and Prescaler
BSF      STATUS, RP0  ;Bank 1
CLRWDT                ;Clears WDT
MOVLW    b'xxxx1xxx'  ;Select new
MOVWF    OPTION_REG       ; prescale value
BCF      STATUS, RP0  ;Bank 0
```

EXAMPLE 6-2: CHANGING PRESCALER (WDT→TIMER0)

```
CLRWDT                ;Clear WDT and
                      ; prescaler
BSF      STATUS, RP0  ;Bank 1
MOVLW    b'xxxx0xxx'  ;Select TMR0, new
                      ; prescale value
                      ' and clock source
MOVWF    OPTION_REG       ;
BCF      STATUS, RP0  ;Bank 0
```

TABLE 6-1 REGISTERS ASSOCIATED WITH TIMER0

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other resets
01h	TMR0	Timer0 module’s register								xxxx xxxx	uuuu uuuu
0Bh	INTCON	GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 0000
81h	OPTION_REG	$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111
85h	TRISA	—	—	—	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	---1 1111	---1 1111

Legend: x = unknown, u = unchanged. - = unimplemented read as '0'. Shaded cells are not associated with Timer0.

8.10 Context Saving During Interrupts

During an interrupt, only the return PC value is saved on the stack. Typically, users wish to save key register values during an interrupt (e.g., W register and STATUS register). This is implemented in software.

Example 8-1 stores and restores the STATUS and W register's values. The User defined registers, W_TEMP and STATUS_TEMP are the temporary storage locations for the W and STATUS registers values.

Example 8-1 does the following:

a) Stores the W register.
b) Stores the STATUS register in STATUS_TEMP.
c) Executes the Interrupt Service Routine code.
d) Restores the STATUS (and bank select bit) register.
e) Restores the W register.

EXAMPLE 8-1: SAVING STATUS AND W REGISTERS IN RAM

```
PUSH   MOVWF   W_TEMP          ; Copy W to TEMP register,
       SWAPF   STATUS, W       ; Swap status to be saved into W
       MOVWF   STATUS_TEMP     ; Save status to STATUS_TEMP register
ISR    :                       :
       :                       ; Interrupt Service Routine
       :                       ;  should configure Bank as required
       :                       ;
POP    SWAPF   STATUS_TEMP, W  ; Swap nibbles in STATUS_TEMP register
                               ; and place result into W
       MOVWF   STATUS          ; Move W into STATUS register
                               ;   (sets bank to original state)
       SWAPF   W_TEMP, F       ; Swap nibbles in W_TEMP and place result in W_TEMP
       SWAPF   W_TEMP, W       ; Swap nibbles in W_TEMP and place result into W
```

8.11 Watchdog Timer (WDT)

The Watchdog Timer is a free running on-chip RC oscillator which does not require any external components. This RC oscillator is separate from the RC oscillator of the OSC1/CLKIN pin. That means that the WDT will run even if the clock on the OSC1/CLKIN and OSC2/CLKOUT pins of the device has been stopped, for example, by execution of a `SLEEP` instruction. During normal operation a WDT time-out generates a device RESET. If the device is in SLEEP mode, a WDT Wake-up causes the device to wake-up and continue with normal operation. The WDT can be permanently disabled by programming configuration bit WDTE as a '0' (Section 8.1).

8.11.1 WDT PERIOD

The WDT has a nominal time-out period of 18 ms, (with no prescaler). The time-out periods vary with temperature, VDD and process variations from part to part (see DC specs). If longer time-out periods are desired, a prescaler with a division ratio of up to 1:128 can be assigned to the WDT under software control by writing to the OPTION_REG register. Thus, time-out periods up to 2.3 seconds can be realized.

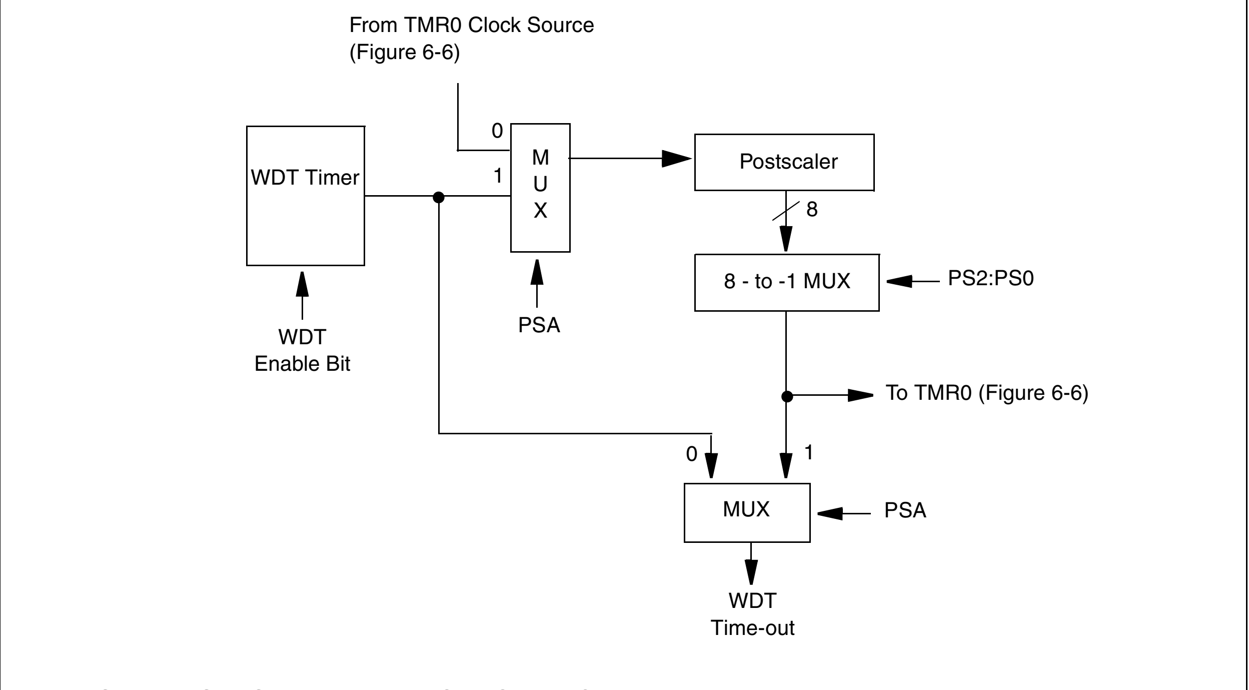
The `CLRWDT` and `SLEEP` instructions clear the WDT and the postscaler (if assigned to the WDT) and prevent it from timing out and generating a device RESET condition.

The $\overline{\text{TO}}$ bit in the STATUS register will be cleared upon a WDT time-out.

8.11.2 WDT PROGRAMMING CONSIDERATIONS

It should also be taken into account that under worst case conditions (VDD = Min., Temperature = Max., max. WDT prescaler) it may take several seconds before a WDT time-out occurs.

FIGURE 8-18: WATCHDOG TIMER BLOCK DIAGRAM



Note: PSA and PS2:PS0 are bits in the OPTION_REG register.

TABLE 8-7 SUMMARY OF REGISTERS ASSOCIATED WITH THE WATCHDOG TIMER

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other resets
2007h	Config. bits	(2)	(2)	(2)	(2)	PWRTE(1)	WDTE	FOSC1	FOSC0	(2)	
81h	OPTION_ REG	$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111

Legend: x = unknown. Shaded cells are not used by the WDT.
Note 1: See Figure 8-1 and Figure 8-2 for operation of the PWRTE bit.
2: See Figure 8-1, Figure 8-2 and Section 8.13 for operation of the Code and Data protection bits.

IORWF — Inclusive OR W with f

Syntax:	[*label*] IORWF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(W) .OR. (f) $\rightarrow$ (destination)
Status Affected:	Z
Encoding:	`00` `0100` `dfff` `ffff`
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'.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example

```
IORWF        RESULT, 0
```

Before Instruction
RESULT = 0x13
W = 0x91
After Instruction
RESULT = 0x13
W = 0x93
Z = 1

MOVLW — Move Literal to W

Syntax:	[*label*] MOVLW k
Operands:	$0 \le k \le 255$
Operation:	k $\rightarrow$ (W)
Status Affected:	None
Encoding:	`11` `00xx` `kkkk` `kkkk`
Description:	The eight bit literal 'k' is loaded into W register. The don't cares will assemble as 0's.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process data	Write to W

Example

```
MOVLW   0x5A
```

After Instruction
W = 0x5A

MOVF — Move f

Syntax:	[*label*] MOVF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) $\rightarrow$ (destination)
Status Affected:	Z
Encoding:	`00` `1000` `dfff` `ffff`
Description:	The contents of register f is moved to a destination dependant upon the status of d. If d = 0, destination is W register. If d = 1, the destination is file register f itself. d = 1 is useful to test a file register since status flag Z is affected.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example

```
MOVF    FSR,  0
```

After Instruction
W = value in FSR register
Z = 1

MOVWF — Move W to f

Syntax:	[*label*] MOVWF f
Operands:	$0 \le f \le 127$
Operation:	(W) $\rightarrow$ (f)
Status Affected:	None
Encoding:	`00` `0000` `1fff` `ffff`
Description:	Move data from W register to register 'f'.
Words:	1
Cycles:	1

Q Cycle Activity:

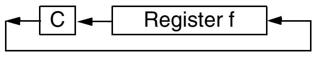
Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write register 'f'

Example

```
MOVWF   OPTION_REG
```

Before Instruction
OPTION = 0xFF
W = 0x4F
After Instruction
OPTION = 0x4F
W = 0x4F

RLF	**Rotate Left f through Carry**
Syntax:	[*label*] RLF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	See description below
Status Affected:	C
Encoding:	`00` `1101` `dfff` `ffff`
Description:	The contents of register 'f' are rotated one bit to the left through the Carry Flag. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is stored back in register 'f'.
Words:	1
Cycles:	1



Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example `RLF REG1,0`

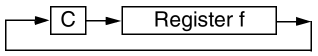
Before Instruction

```
REG1  =  1110 0110
C     =  0
```

After Instruction

```
REG1  =  1110 0110
W     =  1100 1100
C     =  1
```

RRF	**Rotate Right f through Carry**
Syntax:	[*label*] RRF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	See description below
Status Affected:	C
Encoding:	`00` `1100` `dfff` `ffff`
Description:	The contents of register 'f' are rotated one bit to the right through the Carry Flag. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is placed back in register 'f'.
Words:	1
Cycles:	1



Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process data	Write to destination

Example `RRF REG1,0`

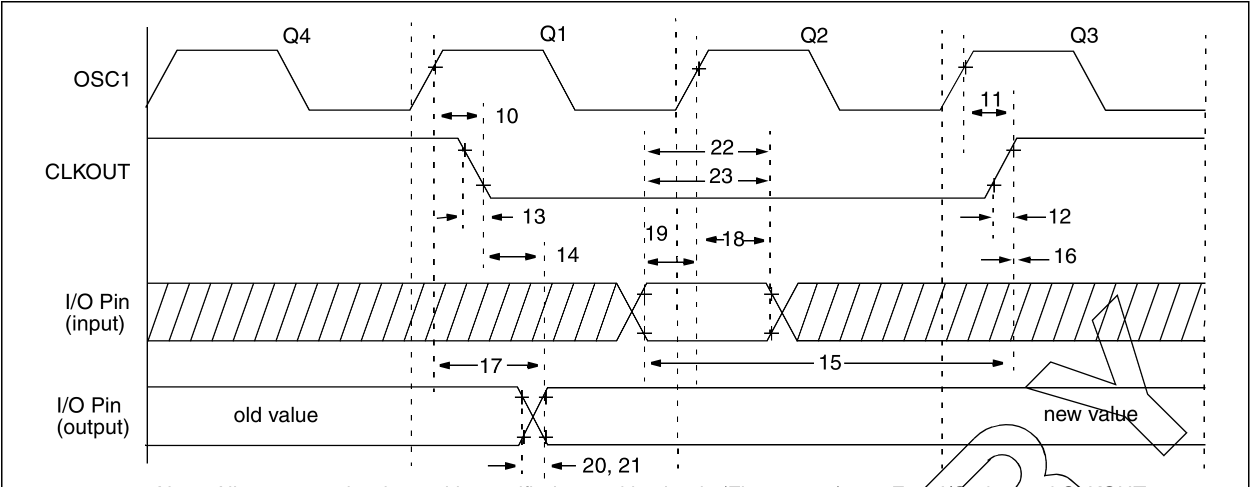
Before Instruction

```
REG1  =  1110 0110
C     =  0
```

After Instruction

```
REG1  =  1110 0110
W     =  0111 0011
C     =  0
```

FIGURE 10-4: CLKOUT AND I/O TIMING



Note: All tests must be done with specified capacitive loads (Figure 10-2) 50 pF on I/O pins and CLKOUT.

TABLE 10-4 CLKOUT AND I/O TIMING REQUIREMENTS

Parameter No.	Sym	Characteristic		Min	Typ†	Max	Units	Conditions
10	TosH2ckL	OSC1↑ to CLKOUT↓	PIC16F8X	—	15	30 *	ns	Note 1
10A			PIC16LF8X	—	15	120 *	ns	Note 1
11	TosH2ckH	OSC1↑ to CLKOUT↑	PIC16F8X	—	15	30 *	ns	Note 1
11A			PIC16LF8X	—	15	120 *	ns	Note 1
12	TckR	CLKOUT rise time	PIC16F8X	—	15	30 *	ns	Note 1
12A			PIC16LF8X	—	15	100 *	ns	Note 1
13	TckF	CLKOUT fall time	PIC16F8X	—	15	30 *	ns	Note 1
13A			PIC16LF8X	—	15	100 *	ns	Note 1
14	TckL2ioV	CLKOUT ↓ to Port out valid		—	—	0.5TCY +20 *	ns	Note 1
15	TioV2ckH	Port in valid before	PIC16F8X	0.30TCY + 30 *	—	—	ns	Note 1
		CLKOUT ↑	PIC16LF8X	0.30TCY + 80 *	—	—	ns	Note 1
16	TckH2ioI	Port in hold after CLKOUT ↑		0 *	—	—	ns	Note 1
17	TosH2ioV	OSC1↑ (Q1 cycle) to	PIC16F8X	—	—	125 *	ns	
		Port out valid	PIC16LF8X	—	—	250 *	ns	
18	TosH2ioI	OSC1↑ (Q2 cycle) to Port input invalid (I/O in hold time)	PIC16F8X	10 *	—	—	ns	
			PIC16LF8X	10 *	—	—	ns	
19	TioV2osH	Port input valid to OSC1↑ (I/O in setup time)	PIC16F8X	-75 *	—	—	ns	
			PIC16LF8X	-175 *	—	—	ns	
20	TioR	Port output rise time	PIC16F8X	—	10	35 *	ns	
20A			PIC16LF8X	—	10	70 *	ns	
21	TioF	Port output fall time	PIC16F8X	—	10	35 *	ns	
21A			PIC16LF8X	—	10	70 *	ns	
22	Tinp	INT pin high	PIC16F8X	20 *	—	—	ns	
22A		or low time	PIC16LF8X	55 *	—	—	ns	
23	Trbp	RB7:RB4 change INT	PIC16F8X	TOSC §	—	—	ns	
23A		high or low time	PIC16LF8X	TOSC §	—	—	ns	

* These parameters are characterized but not tested.

† Data in "Typ" column is at 5.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

§ By design

Note 1: Measurements are taken in RC Mode where CLKOUT output is 4 x TOSC.

11.4 DC CHARACTERISTICS: PIC16CR84, PIC16CR83 (Commercial, Industrial) PIC16LCR84, PIC16LCR83 (Commercial, Industrial)

DC Characteristics All Pins Except Power Supply Pins			**Standard Operating Conditions (unless otherwise stated)** Operating temperature 0°C ≤ TA ≤ +70°C (commercial); -40°C ≤ TA ≤ +85°C (industrial). Operating voltage VDD range as described in DC spec Section 11.1 and Section 11.2.				
Parameter No.	**Sym**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
		Capacitive Loading Specs on Output Pins					
D100	COSC2	OSC2 pin	—	—	15	pF	In XT, HS and LP modes when external clock is used to drive OSC1.
D101	CIO	All I/O pins and OSC2 (RC mode)	—	—	50	pF	
		Data EEPROM Memory					
D120	ED	Endurance	1M	10M	—	E/W	25°C at 5V
D121	VDRW	VDD for read/write	VMIN	—	6.0	V	VMIN = Minimum operating voltage
D122	TDEW	Erase/Write cycle time	—	10	20*	ms	

* These parameters are characterized but not tested.

† Data in “Typ” column is at 5.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

TABLE 11-2 TIMING PARAMETER SYMBOLOGY

The timing parameter symbols have been created following one of the following formats:

1. TppS2ppS
2. TppS

T			
F	Frequency	T	Time

Lowercase symbols (pp) and their meanings:

pp			
2	to	os,osc	OSC1
ck	CLKOUT	ost	oscillator start-up timer
cy	cycle time	pwrt	power-up timer
io	I/O port	rbt	RBx pins
inp	INT pin	t0	T0CKI
mc	$\overline{\text{MCLR}}$	wdt	watchdog timer

Uppercase symbols and their meanings:

S			
F	Fall	P	Period
H	High	R	Rise
I	Invalid (Hi-impedance)	V	Valid
L	Low	Z	High Impedance

FIGURE 11-1: PARAMETER MEASUREMENT INFORMATION

All timings are measure between high and low measurement points as indicated in the figures below.

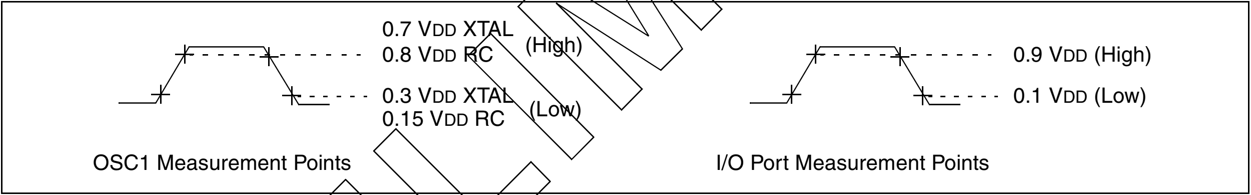
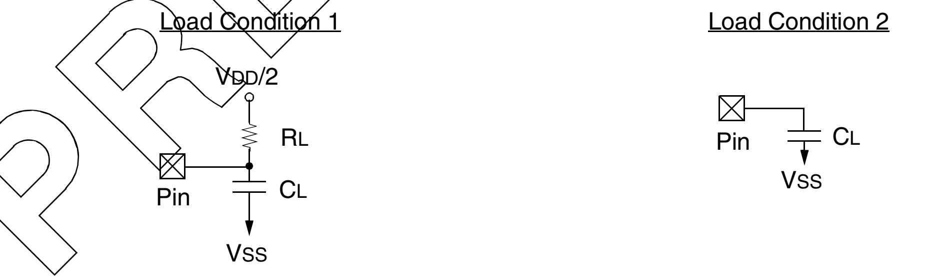


FIGURE 11-2: LOAD CONDITIONS



RL = 464Ω

CL = 50 pF for all pins except OSC2.

15 pF for OSC2 output.

11.5 Timing Diagrams and Specifications

FIGURE 11-3: EXTERNAL CLOCK TIMING

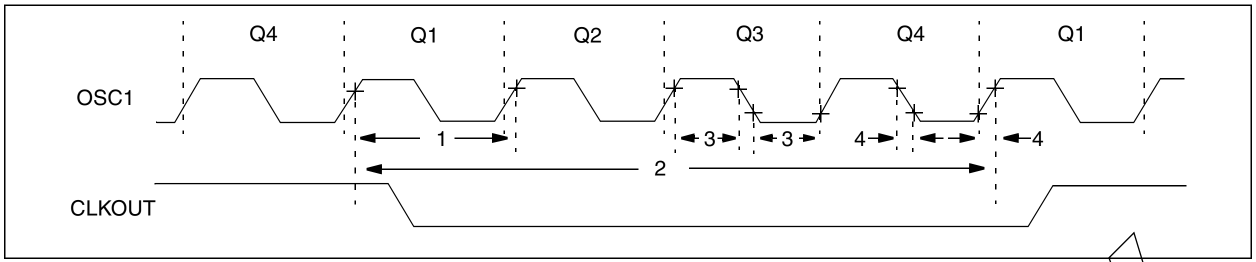


TABLE 11-3 EXTERNAL CLOCK TIMING REQUIREMENTS

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions	
	FOSC	**External CLKIN Frequency**(1)	DC	—	2	MHz	XT, RC osc	PIC16LCR8X-04
			DC	—	4	MHz	XT, RC osc	PIC16CR8X-04
			DC	—	10	MHz	HS osc	PIC16CR8X-10
			DC	—	200	kHz	LP osc	PIC16LCR8X-04
		Oscillator Frequency(1)	DC	—	2	MHz	RC osc	PIC16LCR8X-04
			DC	—	4	MHz	RC osc	PIC16CR8X-04
			0.1	—	2	MHz	XT osc	PIC16LCR8X-04
			0.1	—	4	MHz	XT osc	PIC16CR8X-04
			1.0	—	10	MHz	HS osc	PIC16CR8X-10
			DC	—	200	kHz	LP osc	PIC16LCR8X-04
1	TOSC	**External CLKIN Period**(1)	500	—	—	ns	XT, RC osc	PIC16LCR8X-04
			250	—	—	ns	XT, RC osc	PIC16CR8X-04
			100	—	—	ns	HS osc	PIC16CR8X-10
			5.0	—	—	μs	LP osc	PIC16LCR8X-04
		Oscillator Period(1)	500	—	—	ns	RC osc	PIC16LCR8X-04
			250	—	—	ns	RC osc	PIC16CR8X-04
			500	—	10,000	ns	XT osc	PIC16LCR8X-04
			250	—	10,000	ns	XT osc	PIC16CR8X-04
			100	—	1,000	ns	HS osc	PIC16CR8X-10
			5.0	—	—	μs	LP osc	PIC16LCR8X-04
2	TCY	**Instruction Cycle Time**(1)	0.4	4/FOSC	DC	μs		
3	TOSL, TOSH	**Clock in (OSC1) High or Low Time**	60 *	—	—	ns	XT osc	PIC16LCR8X-04
			50 *	—	—	ns	XT osc	PIC16CR8X-04
			2.0 *	—	—	μs	LP osc	PIC16LCR8X-04
			35 *	—	—	ns	HS osc	PIC16CR8X-10
4	TOSR, TOSF	**Clock in (OSC1) Rise or Fall Time**	25 *	—	—	ns	XT osc	PIC16CR8X-04
			50 *	—	—	ns	LP osc	PIC16LCR8X-04
			15 *	—	—	ns	HS osc	PIC16CR8X-10

* These parameters are characterized but not tested.

† Data in "Typ" column is at 5.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: Instruction cycle period (TCY) equals four times the input oscillator time base period. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at "min." values with an external clock applied to the OSC1 pin.
When an external clock input is used, the "Max." cycle time limit is "DC" (no clock) for all devices.

FIGURE 12-2: TYPICAL RC OSCILLATOR FREQUENCY vs. VDD, CEXT = 20 pF

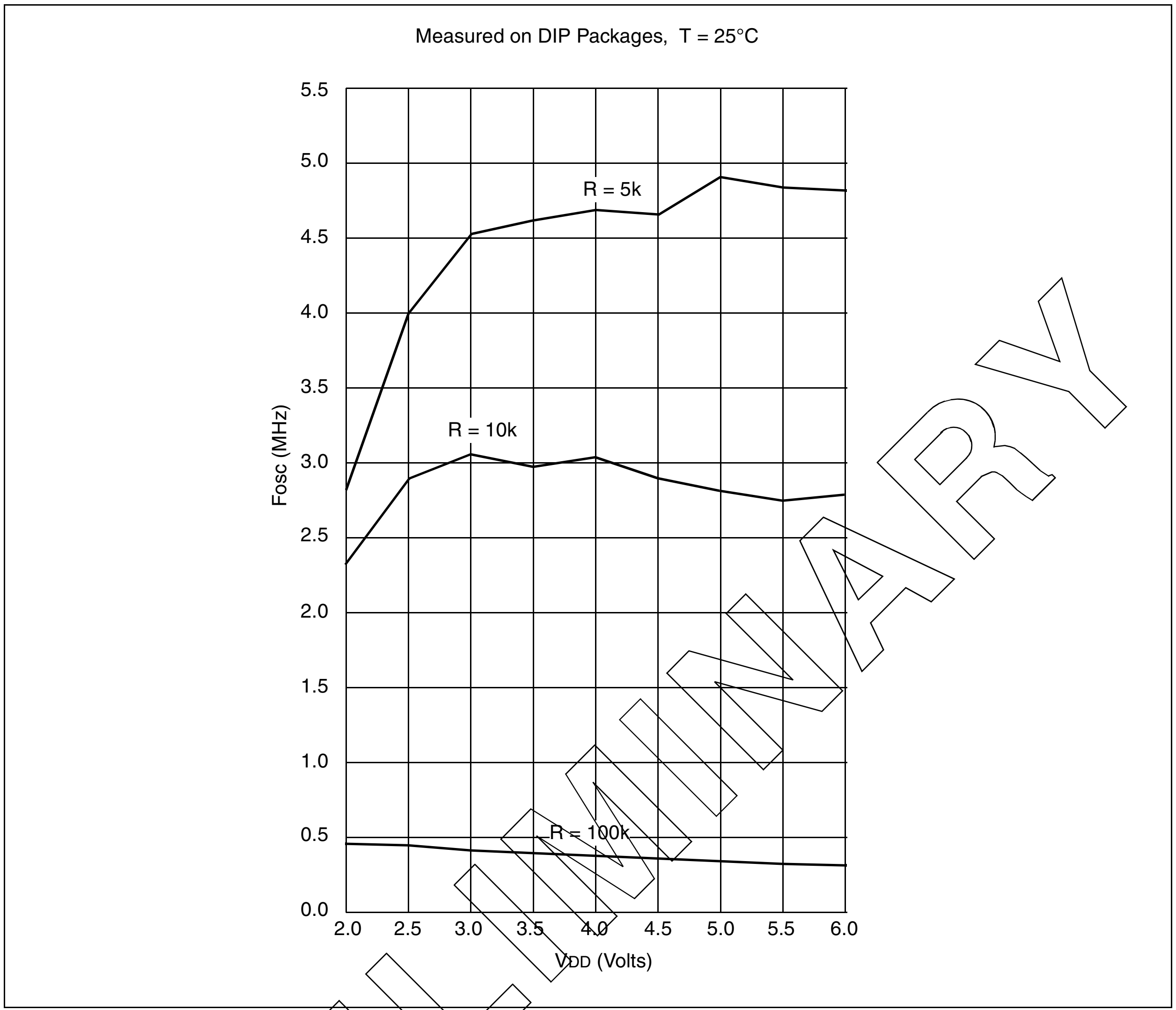


FIGURE 12-3: TYPICAL RC OSCILLATOR FREQUENCY vs. VDD, CEXT = 100 pF

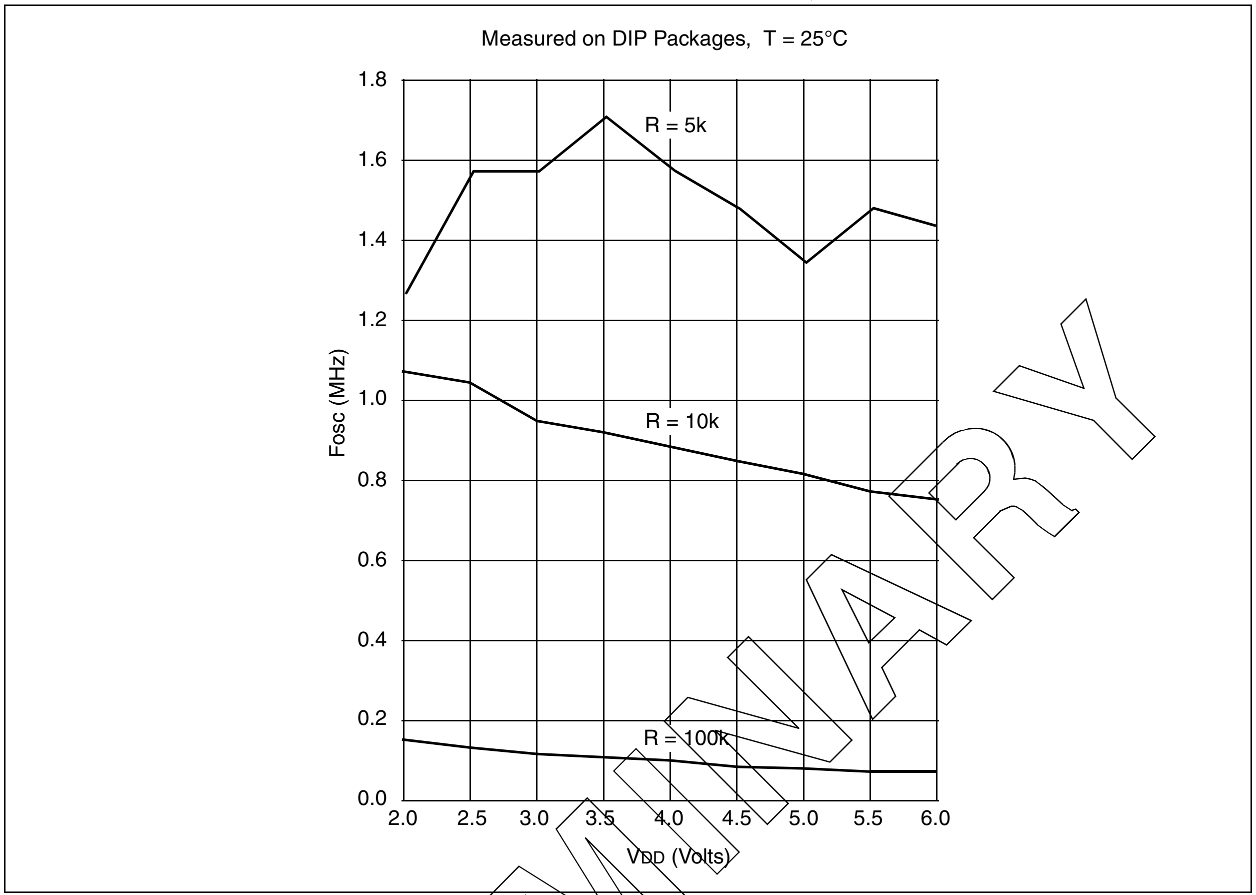


FIGURE 12-4: TYPICAL RC OSCILLATOR FREQUENCY vs. VDD, CEXT = 300 pF

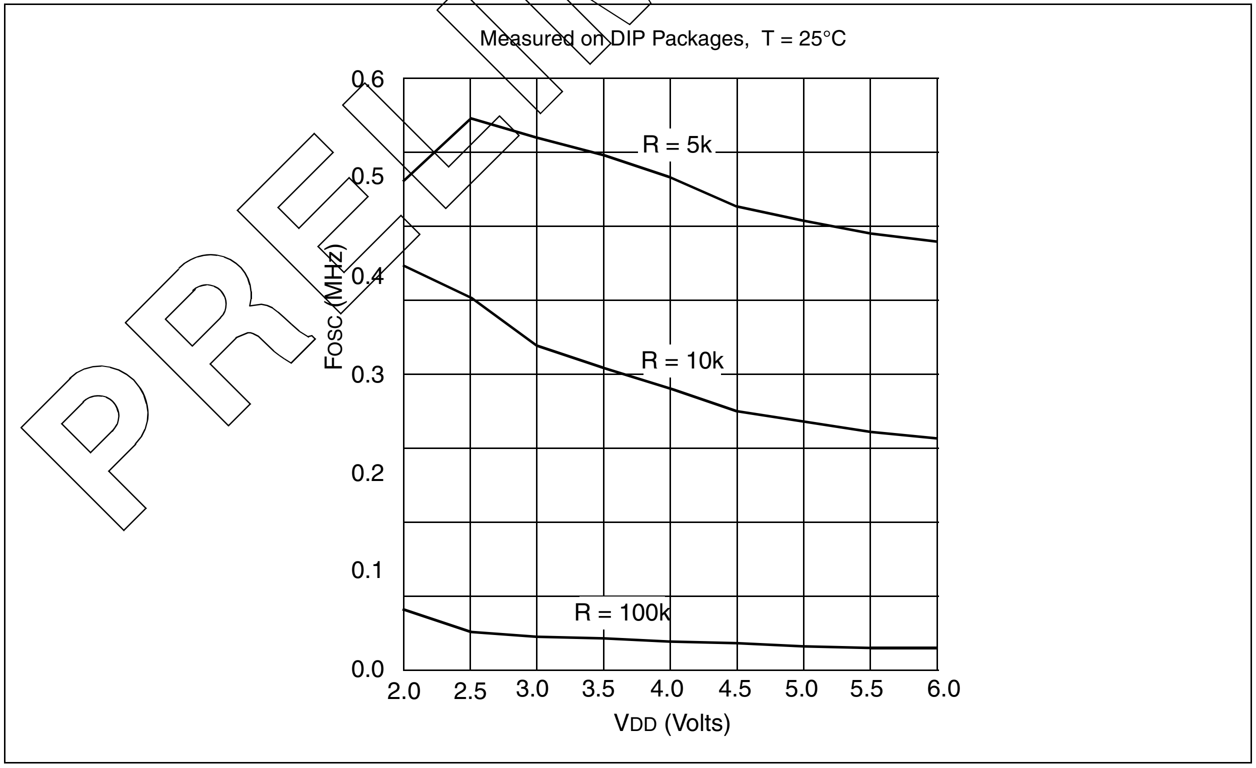


FIGURE 12-5: TYPICAL IPD vs. VDD, WATCHDOG DISABLED

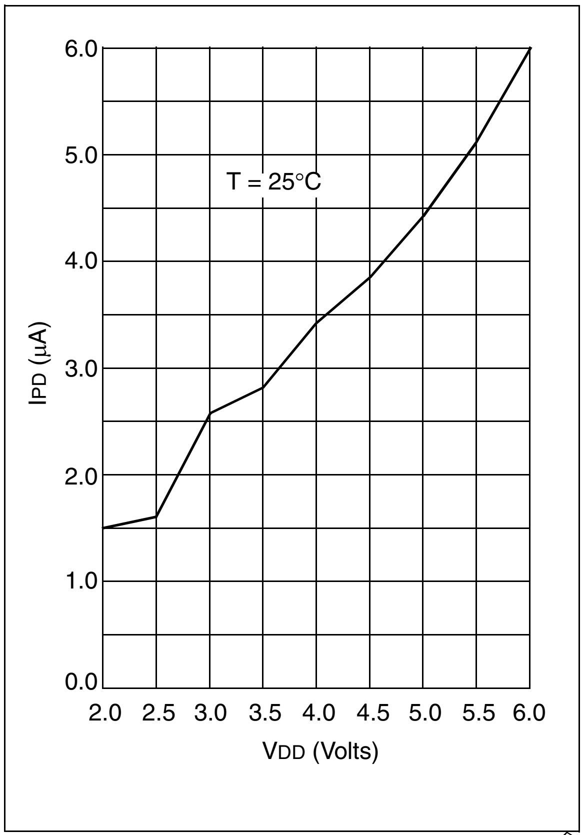


FIGURE 12-6: TYPICAL IPD vs. VDD, WATCHDOG ENABLED

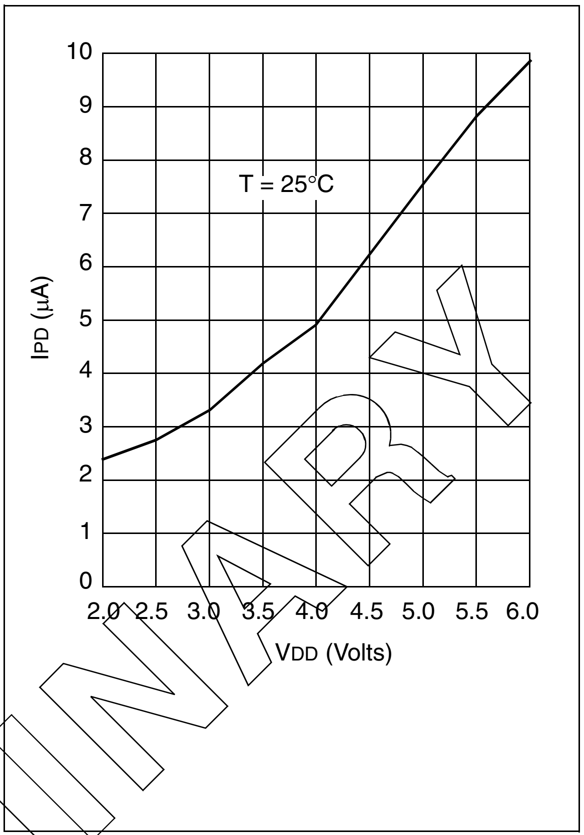
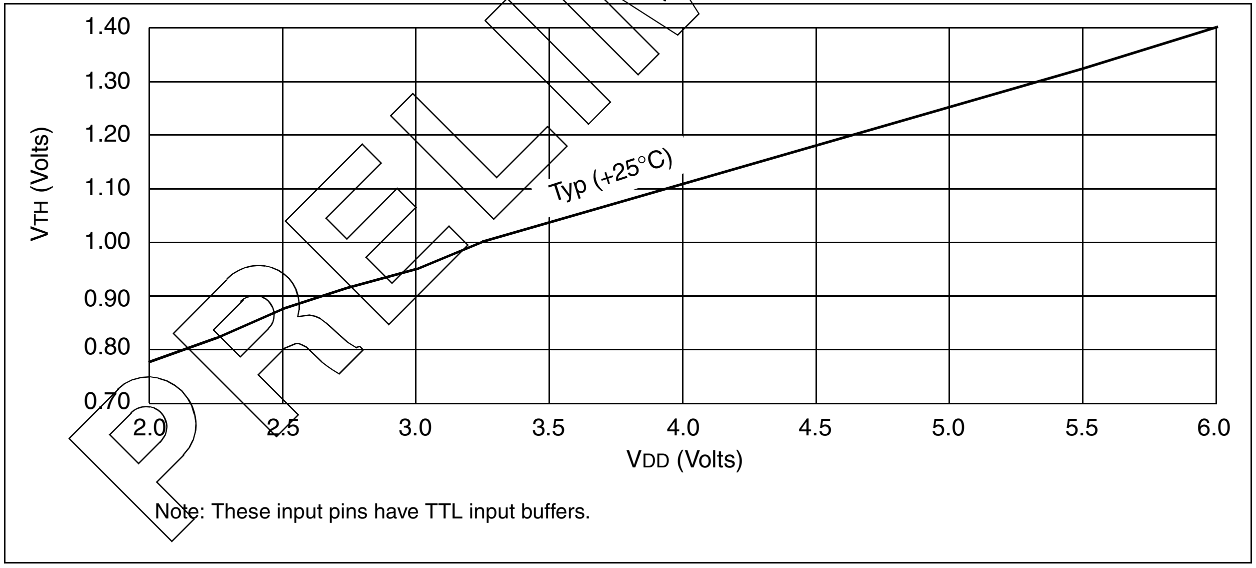
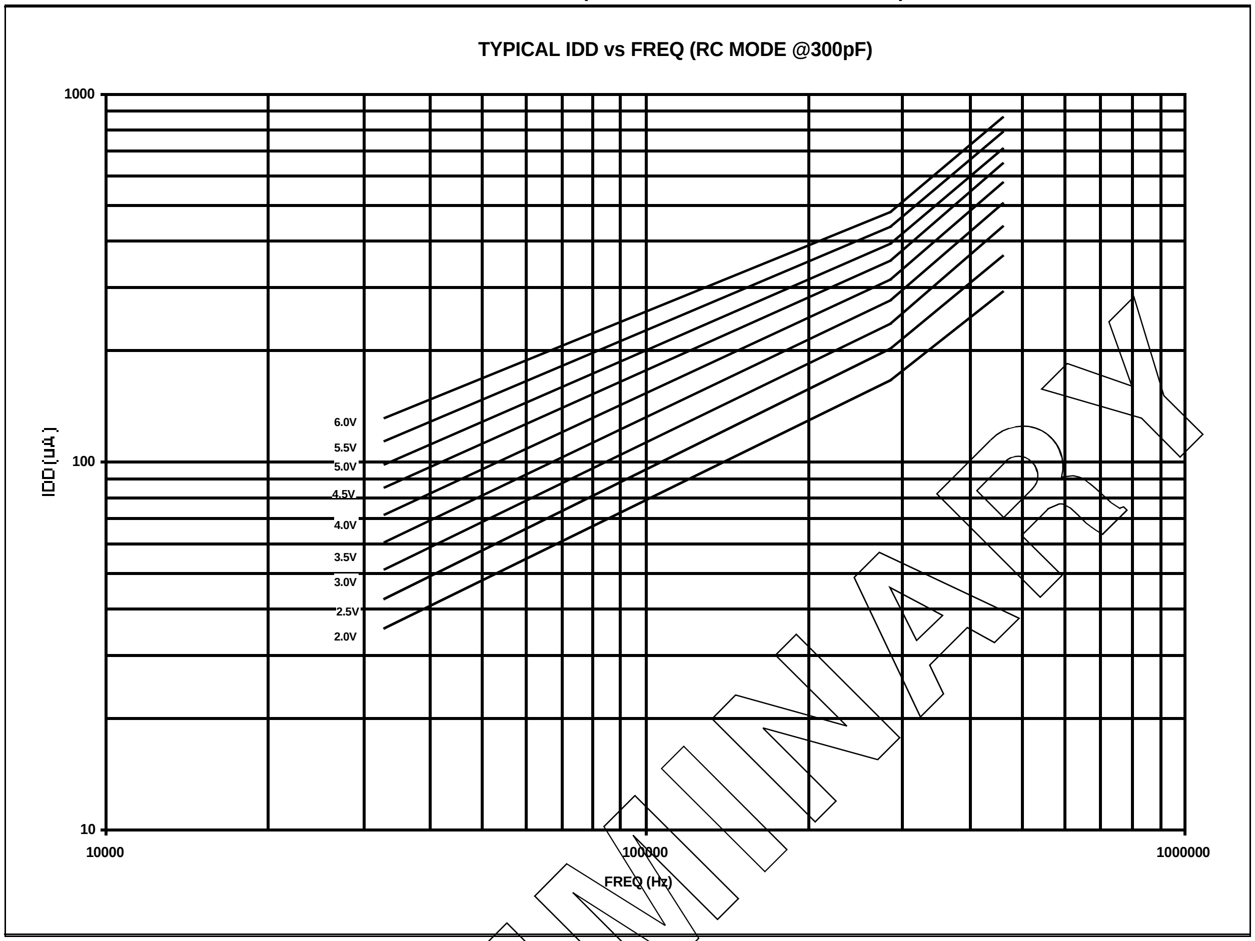


FIGURE 12-7: VTH (INPUT THRESHOLD VOLTAGE) OF I/O PINS vs. VDD



Note: These input pins have TTL input buffers.

FIGURE 12-12: TYPICAL IDD vs. FREQUENCY (RC MODE @300PF, 25°C)



READER RESPONSE

It is our intention to provide you with the best documentation possible to ensure successful use of your Microchip product. If you wish to provide your comments on organization, clarity, subject matter, and ways in which our documentation can better serve you, please FAX your comments to the Technical Publications Manager at (602) 786-7578.

Please list the following information, and use this outline to provide us with your comments about this Data Sheet.

To: Technical Publications Manager Total Pages Sent

RE: Reader Response

From: Name ______________________________

Company ______________________________

Address ______________________________

City / State / ZIP / Country ______________________________

Telephone: (_______) _________ - _________ FAX: (______) _________ - _________

Application (optional):

Would you like a reply?____Y ____N

Device: **PIC16F8X** Literature Number: **DS30430D**

Questions:

1. What are the best features of this document?

____

____

2. How does this document meet your hardware and software development needs?

____

____

3. Do you find the organization of this data sheet easy to follow? If not, why?

____

____

4. What additions to the data sheet do you think would enhance the structure and subject?

____

____

5. What deletions from the data sheet could be made without affecting the overall usefulness?

____

____

6. Is there any incorrect or misleading information (what and where)?

____

____

7. How would you improve this document?

____

____

8. How would you improve our software, systems, and silicon products?

____

____

ISBN: 9781620769300

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
═ ISO/TS 16949 ═

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.