CF76XX

EPROM-Based 8-Bit CMOS Microcontrollers

Devices included in this data sheet:

- CF7645
- CF7665
- CF7685

High Performance RISC CPU:

- Only 35 instructions to learn
- All single-cycle instructions (200 ns), except for program branches which are two-cycle
- Operating speed:
  - DC - 20 MHz clock input
  - DC - 200 ns instruction cycle

Device	Program Memory	Data Memory
CF7645	512	96
CF7665	1K	96
CF7685	2K	128

- Interrupt capability
- 12 special function hardware registers
- 8-level deep hardware stack
- Direct, Indirect and Relative addressing modes

Peripheral Features:

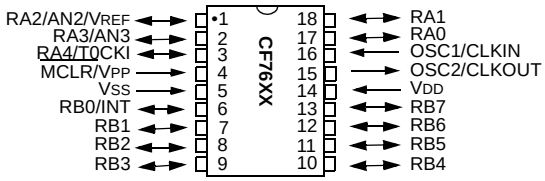
- 13 I/O pins with individual direction control
- High current sink/source for direct LED drive
- Timer0: 8-bit timer/counter with 8-bit programmable prescaler

Special Microcontroller Features:

- Power-on Reset (POR)
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Programmable code protection
- Power saving SLEEP mode
- Selectable oscillator options
- Serial in-circuit programming (via two pins)
- Four user programmable ID locations

Pin Diagrams

PDIP, SOIC



CMOS Technology:

- Low-power, high-speed CMOS EPROM technology
- Fully static design
- Operating Range 3.0V to 5.5V
- Temperature 0°C to 70°C
- Low power consumption
  - < 2.0 mA @ 5.0V, 4.0 MHz
  - 15 µA typical @ 3.0V, 32 kHz
  - < 1.0 µA typical standby current @ 3.0V, 0°C to 70°C

Table of Contents

NOTES:

1.0 GENERAL DESCRIPTION

The CF76XX devices are 18-Pin EPROM-based members of the versatile PICmicro® family of low-cost, high-performance, CMOS, fully-static, 8-bit microcontrollers.

All PICmicro microcontrollers employ an advanced RISC architecture. The CF76XX devices have enhanced core features, eight-level deep stack, and multiple internal and external interrupt sources. The separate instruction and data buses of the Harvard architecture allow a 14-bit wide instruction word with the separate 8-bit wide data. The two-stage instruction pipeline allows all instructions to execute in a single-cycle, except for program branches (which require two cycles). A total of 35 instructions (reduced instruction set) are available. Additionally, a large register set gives some of the architectural innovations used to achieve a very high performance.

CF76XX microcontrollers typically achieve a 2:1 code compression and a 4:1 speed improvement over other 8-bit microcontrollers in their class.

The CF7645 and CF7665 have 96 bytes of RAM. The CF7685 has 128 bytes of RAM. Each device has 13 I/O pins and an 8-bit timer/counter with an 8-bit programmable prescaler.

CF76XX devices have special features to reduce external components, thus reducing system cost, enhancing system reliability and reducing power consumption. There are four oscillator options, of which the single pin RC oscillator provides a low-cost solution, the LP oscillator minimizes power consumption, XT is a standard crystal, and the HS is for High Speed crystals. The SLEEP (power-down) mode offers power savings. The user can wake up the chip from SLEEP through several external and internal interrupts and reset.

A highly reliable Watchdog Timer with its own on-chip RC oscillator provides protection against software lock- up.

Table 1-1 shows the features of the CF76XX mid-range microcontroller families.

A simplified block diagram of the CF76XX is shown in Figure 3-1.

1.1 Applications

The CF76XX series fits perfectly in applications ranging from battery chargers to low-power remote sensors. The EPROM technology makes customization of application programs (detection levels, pulse generation, timers, etc.) extremely fast and convenient. The small footprint packages make this microcontroller series perfect for all applications with space limitations. Low-cost, low-power, high-performance, ease of use and I/O flexibility make the CF76XX very versatile.

TABLE 1-1: CF76XX FAMILY OF DEVICES

		CF7645	CF7665	CF7685
Clock	Maximum Frequency of Operation (MHz)	20	20	20
Memory	EPROM Program Memory (x14 words)	512	1K	2K
	Data Memory (bytes)	96	96	128
Peripherals	Timer Module(s)	TMR0	TMR0	TMRO
Features	Interrupt Sources	4	4	4
	I/O Pins	13	13	13
	Voltage Range (Volts)	3.0-5.5	3.0-5.5	3.0-5.5
	Packages	18-pin DIP,SOIC;	18-pin DIP, SOIC;	18-pin DIP, SOIC;

All CF76XX Family devices have Power-on Reset, selectable Watchdog Timer, selectable code protect and high I/O current capability. All CF76XX Family devices use serial programming with clock pin RB6 and data pin RB7.

2.0 CF76XX DEVICE VARIETIES

A variety of packaging options are available. Depending on application and production requirements, the proper device option can be selected using the information in the CF76XX Product Identification System section at the end of this data sheet. When placing orders, please use this page of the data sheet to specify the correct part number.

NOTES:

3.0 ARCHITECTURAL OVERVIEW

The high performance of the CF76XX family can be attributed to a number of architectural features commonly found in RISC microprocessors. To begin with, the CF76XX uses a Harvard architecture, in which, program and data are accessed from separate memories using separate busses. This improves bandwidth over traditional von Neumann architecture, where program and data are fetched from the same memory. Separating program and data memory further allows instructions to be sized differently than 8-bit wide data word. Instruction opcodes are 14-bits wide making it possible to have all single word instructions. A 14-bit wide program memory access bus fetches a 14-bit instruction in a single cycle. A two-stage pipeline overlaps fetch and execution of instructions. Consequently, all instructions (35) execute in a single-cycle (200 ns @ 20 MHz) except for program branches.

The CF7645 addresses 512 x 14 on-chip program memory. The CF7665 addresses 1K x 14 program memory. The CF7685 addresses 2K x 14 program memory. All program memory is internal.

The CF76XX can directly or indirectly address its register files or data memory. All special function registers including the program counter are mapped in the data memory. The CF76XX has an orthogonal (symmetrical) instruction set that makes it possible to carry out any operation on any register using any addressing mode. This symmetrical nature and lack of ‘special optimal situations’ make programming with the CF76XX simple yet efficient. In addition, the learning curve is reduced significantly.

The CF76XX devices contain an 8-bit ALU and working register. The ALU is a general purpose arithmetic unit. It performs arithmetic and Boolean functions between data in the working register and any register file.

The ALU is 8-bit wide and capable of addition, subtraction, shift and logical operations. Unless otherwise mentioned, arithmetic operations are two's complement in nature. In two-operand instructions, typically one operand is the working register (W register). The other operand is a file register or an immediate constant. In single operand instructions, the operand is either the W register or a file register.

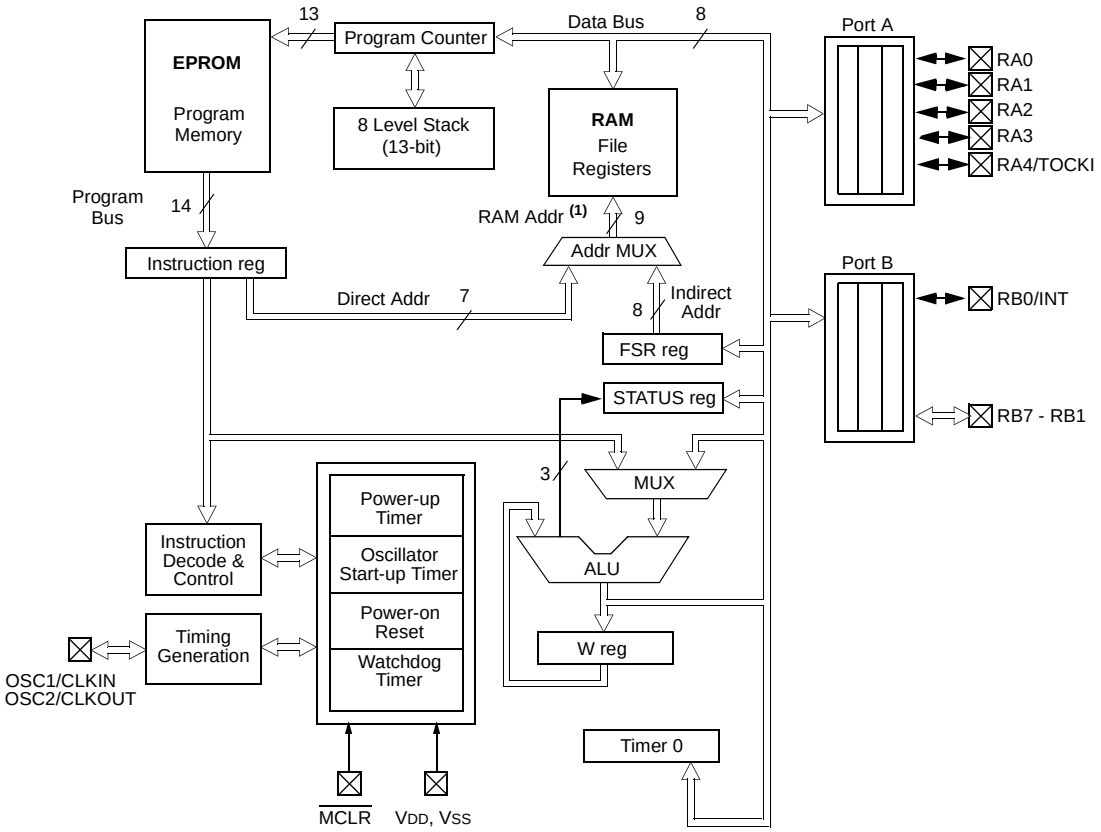
The W register is an 8-bit working register used for ALU operations. It is not an addressable register.

Depending on the instruction executed, the ALU may affect the values of the Carry (C), Digit Carry (DC), and Zero (Z) bits in the STATUS register. The C and DC bits operate as a Borrow and Digit Borrow out bit, respectively, bit in subtraction. See the SUBLW and SUBWF instructions for examples.

A simplified block diagram is shown in Figure 3-1, with a description of the device pins in Table 3-1.

FIGURE 3-1: BLOCK DIAGRAM

Device	Program Memory	Data Memory (RAM)
CF7645	512 x 14	96 x 8
CF7665	1K x 14	96 x 8
CF7685	2K x 14	128 x 8



Note 1: Higher order bits are from the STATUS register.

TABLE 3-1: CF76XX PINOUT DESCRIPTION

Name	DIP/ SOIC Pin #	I/O/P Type	Buffer Type	Description
OSC1/CLKIN	16	I	ST/CMOS	Oscillator crystal input/external clock source input.
OSC2/CLKOUT	15	O	—	Oscillator crystal output. Connects to crystal or resonator in crystal oscillator mode. In RC mode, OSC2 pin outputs CLKOUT, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate.
$\overline{MCLR}$/VPP	4	I/P	ST	Master clear (reset) input/programming voltage input. This pin is an active low reset to the device.
				PORTA is a bi-directional I/O port.
RA0	17	I/O	ST	
RA1	18	I/O	ST	
RA2	1	I/O	ST	
RA3	2	I/O	ST	
RA4/T0CKI	3	I/O	ST	Can be selected to be the clock input to the Timer0 timer/counter. Output is open drain type.
				PORTB is a bi-directional I/O port. PORTB can be software programmed for internal weak pull-up on all inputs.
RB0/INT	6	I/O	TTL/ST(1)	RB0/INT can also be selected as an external interrupt pin.
RB1	7	I/O	TTL	
RB2	8	I/O	TTL	
RB3	9	I/O	TTL	
RB4	10	I/O	TTL	Interrupt on change pin.
RB5	11	I/O	TTL	Interrupt on change pin.
RB6	12	I/O	TTL/ST(2)	Interrupt on change pin. Serial programming clock.
RB7	13	I/O	TTL/ST(2)	Interrupt on change pin. Serial programming data.
VSS	5	P	—	Ground reference for logic and I/O pins.
VDD	14	P	—	Positive supply for logic and I/O pins.

Legend: O = output I/O = input/output P = power
— = Not used I = Input ST = Schmitt Trigger input
TTL = TTL input

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.
Note 2: This buffer is a Schmitt Trigger input when used in serial programming mode.

3.1 Clocking Scheme/Instruction Cycle

The clock input (OSC1/CLKIN pin) is internally divided by four to generate four non-overlapping quadrature clocks namely Q1, Q2, Q3 and Q4. Internally, the program counter (PC) is incremented every Q1, the instruction is fetched from the program memory and latched into the instruction register in Q4. The instruction is decoded and executed during the following Q1 through Q4. The clocks and instruction execution flow is shown in Figure 3-2.

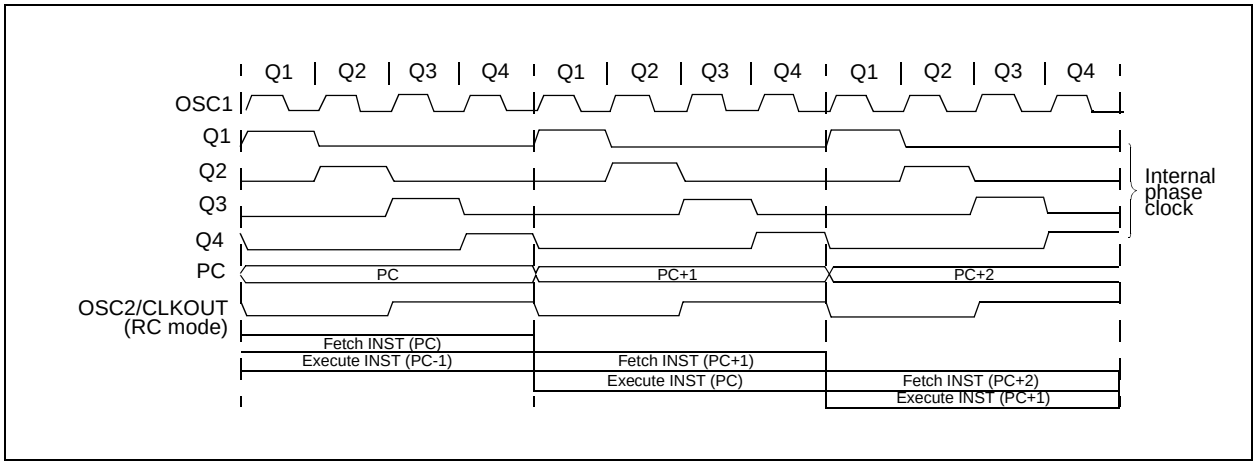
3.2 Instruction Flow/Pipelining

An “Instruction Cycle” consists of four Q cycles (Q1, Q2, Q3 and Q4). The instruction fetch and execute are pipelined such that fetch takes one instruction cycle while decode and execute takes another instruction cycle. However, due to the pipelining, each instruction effectively executes in one cycle. If an instruction causes the program counter to change (e.g., `GOTO`) then two cycles are required to complete the instruction (Example 3-1).

A fetch cycle begins with the program counter (PC) incrementing in Q1.

In the execution cycle, the fetched instruction is latched into the “Instruction Register (IR)” in cycle Q1. This instruction is then decoded and executed during the Q2, Q3, and Q4 cycles. Data memory is read during Q2 (operand read) and written during Q4 (destination write).

FIGURE 3-2: CLOCK/INSTRUCTION CYCLE



EXAMPLE 3-1: INSTRUCTION PIPELINE FLOW

```
1. MOVLW 55h
2. MOVWF PORTB
3. CALL  SUB_1
4. BSF   PORTA, BIT3
```

Fetch 1	Execute 1				
	Fetch 2	Execute 2			
		Fetch 3	Execute 3		
			Fetch 4	Flush	
				Fetch SUB_1	Execute SUB_1

All instructions are single cycle, except for any program branches. These take two cycles since the fetch instruction is “flushed” from the pipeline, while the new instruction is being fetched and then executed.

4.0 MEMORY ORGANIZATION

4.1 Program Memory Organization

The CF76XX has a 13-bit program counter capable of addressing an 8K x 14 program memory space. Only the first 512 x 14 (0000h - 01FFh) for the CF7645, 1K x 14 (0000h - 03FFh) for the CF7665 and 2K x 14 (0000h - 07FFh) for the CF7685 are physically implemented. Accessing a location above these boundaries will cause a wrap-around within the first 512 x 14 space (CF7645) or 1K x 14 space (CF7665) or 2K x 14 space (CF7685). The reset vector is at 0000h and the interrupt vector is at 0004h (Figure 4-1, Figure 4-2, Figure 4-3).

FIGURE 4-1: PROGRAM MEMORY MAP AND STACK FOR THE CF7645

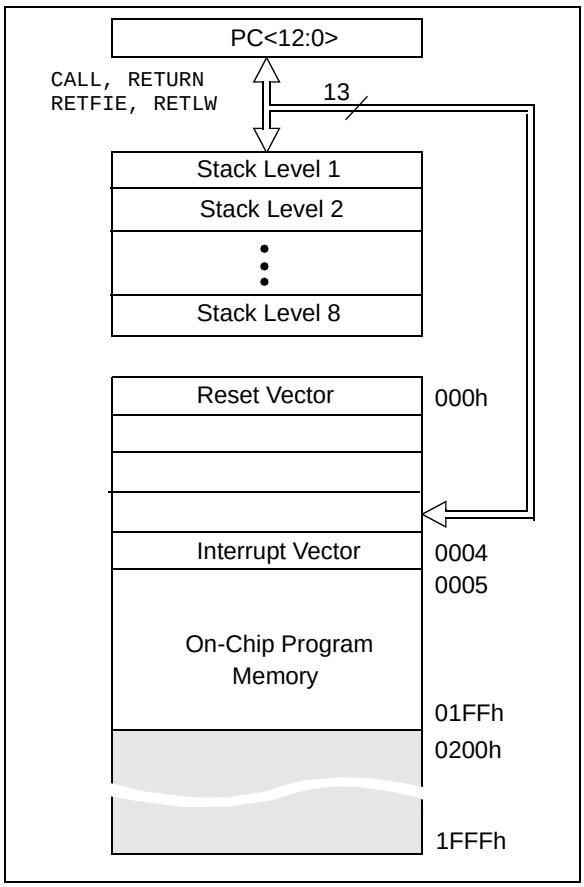


FIGURE 4-2: PROGRAM MEMORY MAP AND STACK FOR THE CF7665

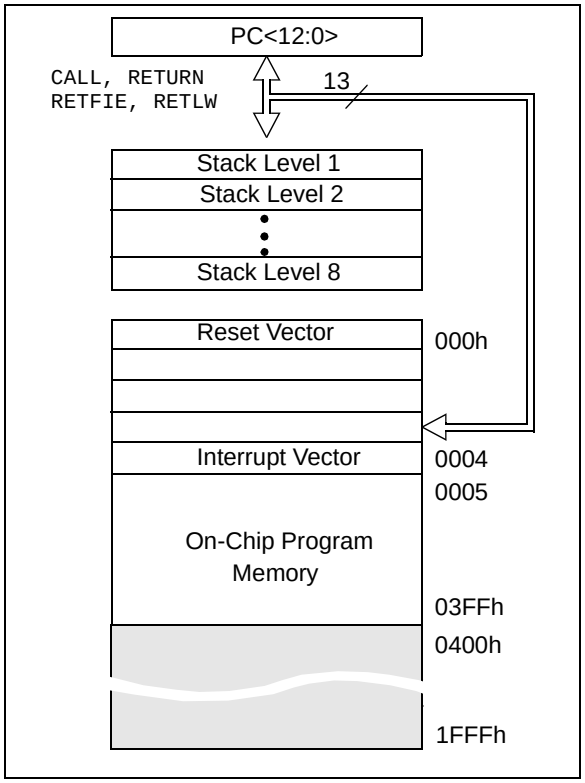
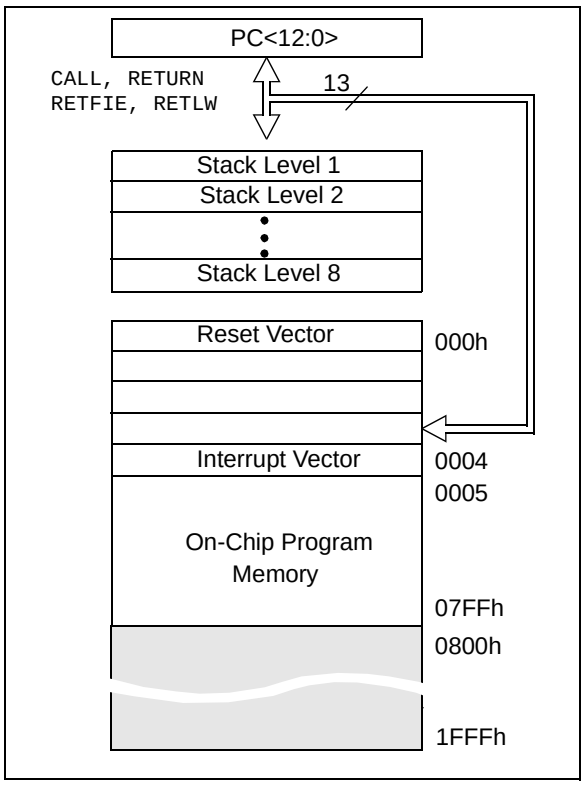


FIGURE 4-3: PROGRAM MEMORY MAP AND STACK FOR THE CF7685



4.2 Data Memory Organization

The data memory (Figure 4-4 and Figure 4-5) is partitioned into two banks, which contain the General Purpose Registers and the Special Function Registers. Bank 0 is selected when the RP0 bit is cleared. Bank 1 is selected when the RP0 bit (STATUS <5>) is set. The Special Function Registers are located in the first 32 locations of each bank. Register locations 20-7Fh (Bank0) on the CF7645 and 20-7Fh (Bank0) and A0-BFh (Bank1) on the CF7665 and CF7685 are General Purpose Registers implemented as static RAM. Some Special Purpose Registers are mapped in Bank 1.

Addresses F0h-FFh of bank1 are implemented as common ram and mapped back to addresses 70h-7Fh in bank0 on the CF7645/7665/7685.

4.2.1 GENERAL PURPOSE REGISTER FILE

The register file is organized as 96 x 8 in the CF7645 and CF7665 and 128 x 8 in the CF7685. Each is accessed either directly or indirectly through the File Select Register FSR (Section 4.4).

FIGURE 4-4: DATA MEMORY MAP FOR THE CF7645/CF7665

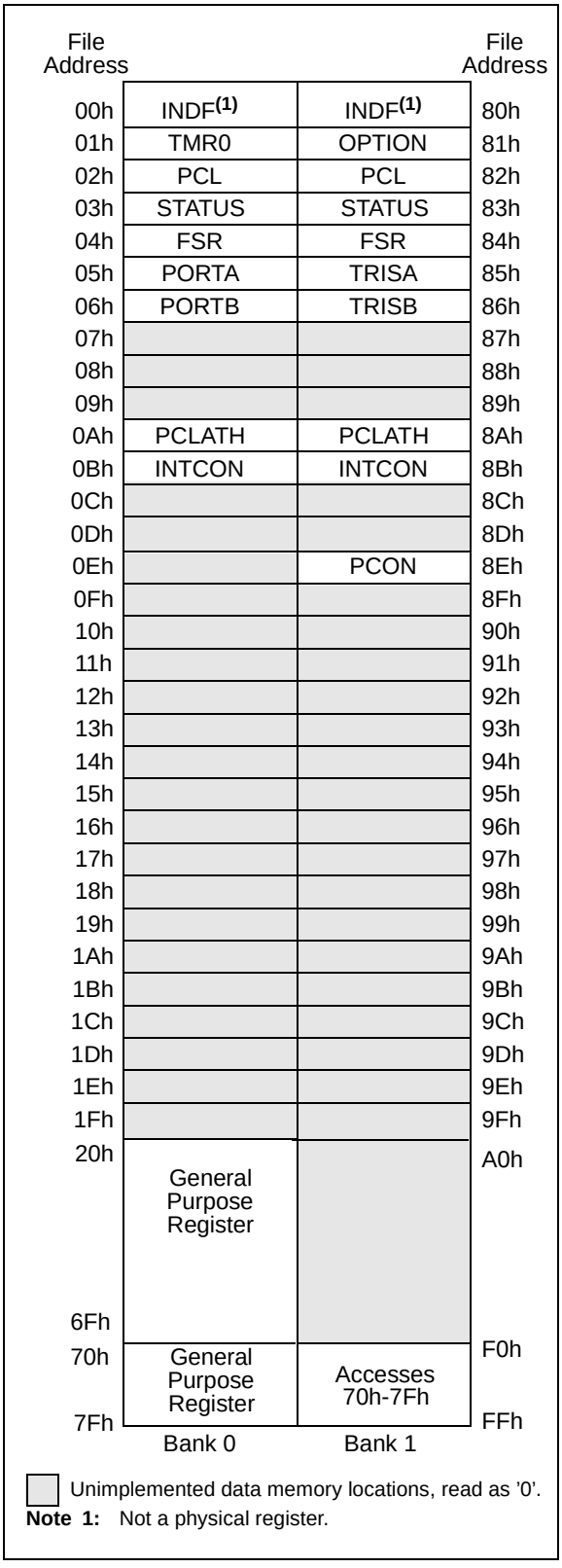
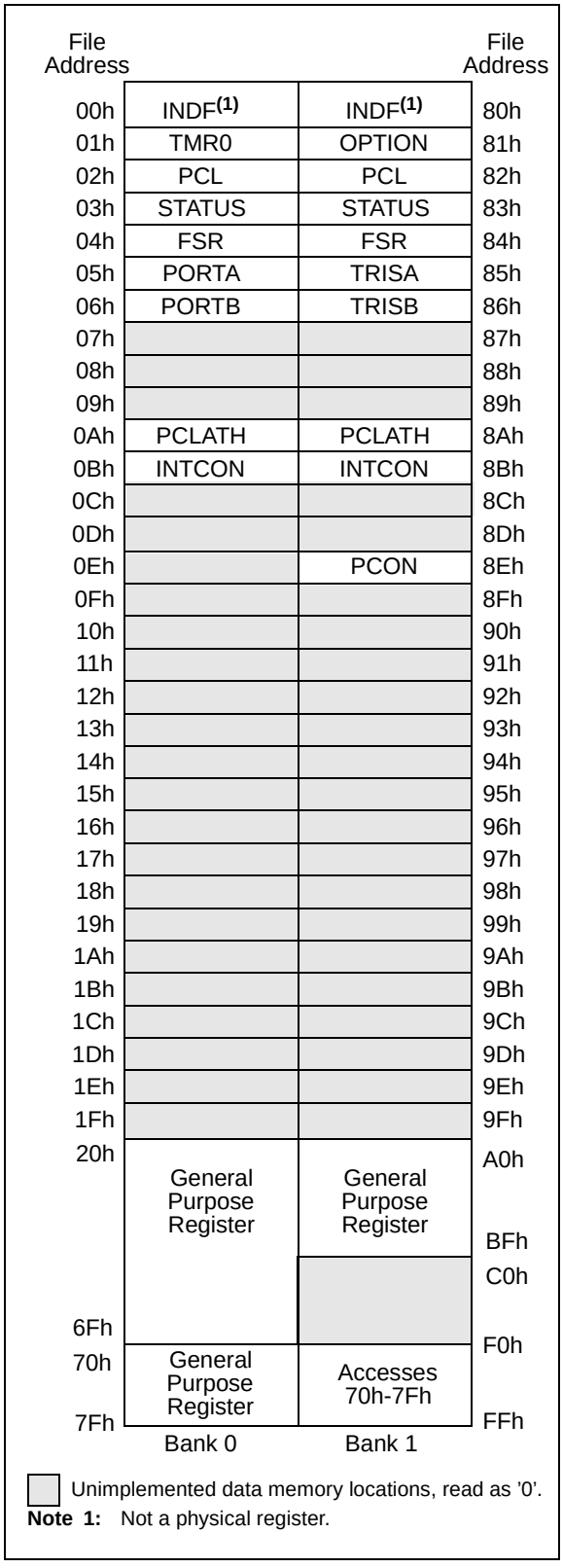


FIGURE 4-5: DATA MEMORY MAP FOR THE CF7685



4.2.2 SPECIAL FUNCTION REGISTERS

The Special Function Registers are registers used by the CPU and Peripheral functions for controlling the desired operation of the device (Table 4-1). These registers are 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 in this section. Those related to the operation of the peripheral features are described in the section of that peripheral feature.

TABLE 4-1: SPECIAL REGISTERS FOR THE CF76XX STATUS REGISTER

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR Reset	Value on all other resets(1)
Bank 0											
00h	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								xxxx xxxx	xxxx xxxx
01h	TMR0	Timer0 Module’s Register								xxxx xxxx	uuuu uuuu
02h	PCL	Program Counter's (PC) Least Significant Byte								0000 0000	0000 0000
03h	STATUS	IRP(2)	RP1(2)	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	000q quuu
04h	FSR	Indirect data memory address pointer								xxxx xxxx	uuuu uuuu
05h	PORTA	—	—	—	RA4	RA3	RA2	RA1	RA0	---x 0000	---u 0000
06h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx	uuuu uuuu
07h-09h	Unimplemented									—	—
0Ah	PCLATH	—	—	—	Write buffer for upper 5 bits of program counter					---0 0000	---0 0000
0Bh	INTCON	GIE	—	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0-00 000x	0000 000u
0Ch-1Fh	Unimplemented									—	—
Bank 1											
80h	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								xxxx xxxx	xxxx xxxx
81h	OPTION	$\overline{RBPU}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111
82h	PCL	Program Counter's (PC) Least Significant Byte								0000 0000	0000 0000
83h	STATUS	IRP(2)	RP1(2)	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	000q quuu
84h	FSR	Indirect data memory address pointer								xxxx xxxx	uuuu uuuu
85h	TRISA	—	—	—	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	---1 1111	---1 1111
86h	TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111	1111 1111
87h-89h	Unimplemented									—	—
8Ah	PCLATH	—	—	—	Write buffer for upper 5 bits of program counter					---0 0000	---0 0000
8Bh	INTCON	GIE	—	T0IE	INTE	RBIE	T0IF	INTF	RBIF	000 000x	0000 000u
8Ch-8Dh	Unimplemented									—	—
8Eh	PCON	—	—	—	—	—	—	$\overline{POR}$	—	---- --0-	---- --u-
8Fh-9Fh	Unimplemented									—	—

Legend: — = Unimplemented locations read as ‘0’, u = unchanged, x = unknown,
q = value depends on condition, shaded = unimplemented

Note 1: Other (non power-up) resets include $\overline{MCLR}$ reset, and Watchdog Timer Reset during normal operation.

2: IRP & RP1 bits are reserved, always maintain these bits clear.

3: Bit 6 of INTCON register is reserved for future use. Always maintain this bit as clear.

The STATUS register, shown in Register 4-1, contains the arithmetic status of the ALU, the RESET status and the bank select bits for data memory.

The STATUS register can be the destination for any instruction, like any other register. If the STATUS register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. These bits are set or cleared according to the device logic. Furthermore, the $\overline{TO}$ and $\overline{PD}$ bits are not writable. Therefore, the result of an instruction with the STATUS register as destination may be different than intended.

For example, `CLRF STATUS` will clear the upper-three bits and set the Z bit. This leaves the status register as `000uu1uu` (where u = unchanged).

It is recommended, therefore, that only `BCF`, `BSF`, `SWAPF` and `MOVWF` instructions are used to alter the STATUS register, because these instructions do not affect any status bit. For other instructions, not affecting any status bits, see the “Instruction Set Summary”.

Note 1:	The IRP and RP1 bits (STATUS<7:6>) are not used by the CF76XX and should be programmed as '0'. Use of these bits as general purpose R/W bits is NOT recommended, since this may affect upward compatibility with future products.
2:	The C and DC bits operate as a $\overline{\text{Borrow}}$ and $\overline{\text{Digit Borrow}}$ out bit, respectively, in subtraction. See the `SUBLW` and `SUBWF` instructions for examples.

REGISTER 4-1: STATUS REGISTER (ADDRESS 03H OR 83H)

Reserved	Reserved	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
							bit0

R = Readable bit
W = Writable bit
U = Unimplemented bit, read as '0'
- n = Value at POR
- x = Unknown at POR

bit 7: **IRP**: Register Bank Select bit (used for indirect addressing)
1 = Bank 2, 3 (100h - 1FFh)
0 = Bank 0, 1 (00h - FFh)
The IRP bit is reserved on the CF76XX; always maintain this bit clear.

bit 6-5: **RP<1:0>**: Register Bank Select bits (used for direct addressing)
01 = Bank 1 (80h - FFh)
00 = Bank 0 (00h - 7Fh)
Each bank is 128 bytes. The RP1 bit is reserved on the CF76XX; always maintain this bit clear.

bit 4: **$\overline{TO}$**: Time-out bit
1 = After power-up, CLRWDT instruction, or SLEEP instruction
0 = A WDT time-out occurred

bit 3: **$\overline{PD}$**: Power-down bit
1 = After power-up or by the CLRWDT instruction
0 = By execution of the SLEEP instruction

bit 2: **Z**: Zero bit
1 = The result of an arithmetic or logic operation is zero
0 = The result of an arithmetic or logic operation is not zero

bit 1: **DC**: Digit carry/$\overline{borrow}$ bit (ADDWF, ADDLW, SUBLW, SUBWF instructions)(for $\overline{borrow}$ the polarity is reversed)
1 = A carry-out from the 4th low order bit of the result occurred
0 = No carry-out from the 4th low order bit of the result

bit 0: **C**: Carry/$\overline{borrow}$ bit (ADDWF, ADDLW, SUBLW, SUBWF instructions)
1 = A carry-out from the most significant bit of the result occurred
0 = No carry-out from the most significant bit of the result occurred
Note: For $\overline{borrow}$ the polarity is reversed. A subtraction is executed by adding the two's complement of the second operand. For rotate (RRF, RLF) instructions, this bit is loaded with either the high or low order bit of the source register.

4.2.2.1 OPTION REGISTER

The OPTION register is a readable and writable register, which contains various control bits to configure the TMR0/WDT prescaler, the external RB0/INT interrupt, TMR0 and the weak pull-ups on PORTB.

Note: To achieve a 1:1 prescaler assignment for TMR0, assign the prescaler to the WDT (PSA = 1).

REGISTER 4-2: OPTION REGISTER (ADDRESS 81H)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0
bit7							bit0

R = Readable bit
W = Writable bit
U = Unimplemented bit, read as '0'
- n = Value at POR reset
- x = Unknown at POR reset

bit 7: **$\overline{\text{RBPU}}$**: PORTB Pull-up Enable bit
1 = PORTB pull-ups are disabled
0 = PORTB pull-ups are enabled by individual port latch values

bit 6: **INTEDG**: Interrupt Edge Select bit
1 = Interrupt on rising edge of RB0/INT pin
0 = Interrupt on falling edge of RB0/INT pin

bit 5: **T0CS**: TMR0 Clock Source Select bit
1 = Transition on RA4/T0CKI pin
0 = Internal instruction cycle clock (CLKOUT)

bit 4: **T0SE**: TMR0 Source Edge Select bit
1 = Increment on high-to-low transition on RA4/T0CKI pin
0 = Increment on low-to-high transition on RA4/T0CKI pin

bit 3: **PSA**: Prescaler Assignment bit
1 = Prescaler is assigned to the WDT
0 = Prescaler is assigned to the Timer0 module

bit 2-0: **PS<2:0>**: Prescaler Rate Select bits

Bit Value	TMR0 Rate	WDT Rate
000	1 : 2	1 : 1
001	1 : 4	1 : 2
010	1 : 8	1 : 4
011	1 : 16	1 : 8
100	1 : 32	1 : 16
101	1 : 64	1 : 32
110	1 : 128	1 : 64
111	1 : 256	1 : 128

4.2.2.2 INTCON REGISTER

The INTCON register is a readable and writable register, which contains the various enable and flag bits for all interrupt sources except the comparator module.

Note:	Interrupt flag bits get set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the global enable bit, GIE (INTCON<7>).

REGISTER 4-3: INTCON REGISTER (ADDRESS 0BH OR 8BH)

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE	-	T0IE	INTE	RBIE	T0IF	INTF	RBIF
bit7							bit0

R = Readable bit
W = Writable bit
U = Unimplemented bit, read as '0'
- n = Value at POR reset
- x = Unknown at POR reset

bit 7: **GIE:** Global Interrupt Enable bit
1 = Enables all un-masked interrupts
0 = Disables all interrupts

bit 6: Reserved for future use - always maintain this bit clear.

bit 5: **T0IE**: TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 interrupt
0 = Disables the TMR0 interrupt

bit 4: **INTE**: RB0/INT External Interrupt Enable bit
1 = Enables the RB0/INT external interrupt
0 = Disables the RB0/INT external interrupt

bit 3: **RBIE**: RB Port Change Interrupt Enable bit
1 = Enables the RB port change interrupt
0 = Disables the RB port change interrupt

bit 2: **T0IF**: TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software)
0 = TMR0 register did not overflow

bit 1: **INTF**: RB0/INT External Interrupt Flag bit
1 = The RB0/INT external interrupt occurred (must be cleared in software)
0 = The RB0/INT external interrupt did not occur

bit 0: **RBIF**: RB Port Change Interrupt Flag bit
1 = When at least one of the RB<7:4> pins changed state (must be cleared in software)
0 = None of the RB<7:4> pins have changed state

4.2.2.3 PCON REGISTER

The PCON register contains flag bits to differentiate between a Power-on Reset, an external $\overline{\text{MCLR}}$ reset, or WDT reset.

REGISTER 4-4: PCON REGISTER (ADDRESS 8Eh)

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	—	—	—	$\overline{\text{POR}}$	—
bit7							bit0

R = Readable bit
W = Writable bit
U = Unimplemented bit, read as '0'
- n = Value at POR reset
- x = Unknown at POR reset

bit 7-2: **Unimplemented:** Read as '0'

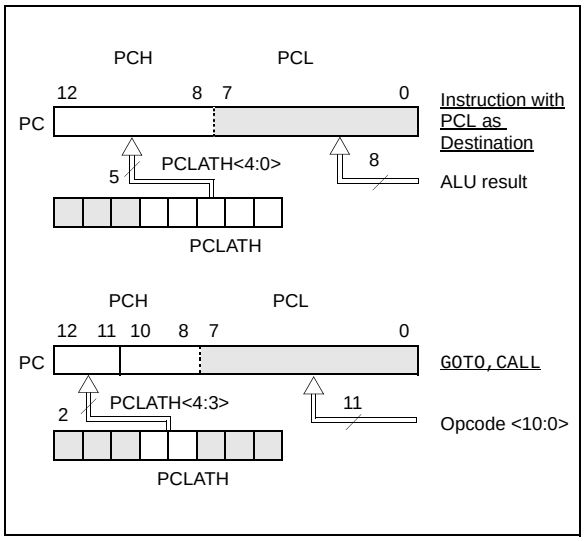
bit 1: **$\overline{\text{POR}}$**: Power-on Reset Status bit
1 = No Power-on Reset occurred
0 = A Power-on Reset occurred (must be set in software after a Power-on Reset occurs)

bit 0: **Unimplemented:** Read as '0'

4.3 PCL and PCLATH

The program counter (PC) is 13-bits wide. The low byte comes from the PCL register, which is a readable and writable register. The high byte (PC<12:8>) is not directly readable or writable and comes from PCLATH. On any reset, the PC is cleared. Figure 4-5 shows the two situations for the loading of the PC. The upper example in the figure shows how the PC is loaded on a write to PCL (PCLATH<4:0> → PCH). The lower example in the figure shows how the PC is loaded during a CALL or GOTO instruction (PCLATH<4:3> → PCH).

FIGURE 4-5: LOADING OF PC IN DIFFERENT SITUATIONS



4.3.1 COMPUTED GOTO

A computed GOTO is accomplished by adding an offset to the program counter (ADDWF PCL). When doing a table read using a computed GOTO method, care should be exercised if the table location crosses a PCL memory boundary (each 256 byte block). Refer to the application note, *"Implementing a Table Read"* (AN556).

4.3.2 STACK

The CF76XX family has an 8-level deep x 13-bit wide hardware stack (Figure 4-2 and Figure 4-3). The stack space is not part of either program or 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 POPed 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 was stored from the first push. The tenth push overwrites the second push (and so on).

Note 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.

4.4 Indirect Addressing, INDF and FSR Registers

The INDF register is not a physical register. Addressing the INDF register will cause indirect addressing.

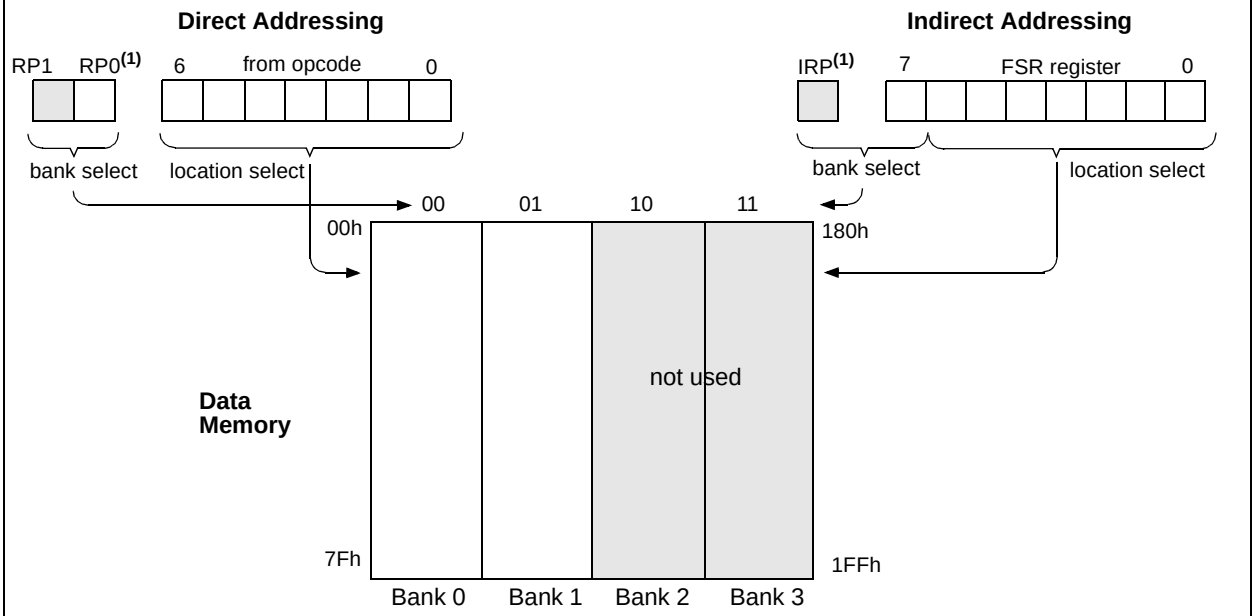
Indirect addressing is possible by using the INDF register. Any instruction using the INDF register actually accesses data pointed to by the File Select Register (FSR). Reading INDF itself indirectly will produce 00h. Writing to the INDF register indirectly results in a no-operation (although status bits may be affected). An effective 9-bit address is obtained by concatenating the 8-bit FSR register and the IRP bit (STATUS<7>), as shown in Figure 4-6. However, IRP is not used in the CF76XX.

A simple program to clear RAM location 20h-7Fh using indirect addressing is shown in Example 4-1.

EXAMPLE 4-1: INDIRECT ADDRESSING

```
          movlw  0x20    ;initialize pointer
          movwf  FSR     ;to RAM
NEXT      clrf   INDF    ;clear INDF register
          incf   FSR     ;inc pointer
          btfss  FSR,7   ;all done?
          goto   NEXT    ;no clear next
                         ;yes continue
CONTINUE:
```

FIGURE 4-6: DIRECT/INDIRECT ADDRESSING CF76XX



For memory map detail see (Figure 4-4 and 4-5).

Note 1: The RP1 and IRP bits are reserved; always maintain these bits clear.

NOTES:

5.0 I/O PORTS

The CF76XX has two ports, PORTA and PORTB. Some pins for these I/O ports are multiplexed with an alternate function for the peripheral features on the device. In general, when a peripheral is enabled, that pin may not be used as a general purpose I/O pin.

5.1 PORTA and TRISA Registers

PORTA is a 5-bit wide latch. RA4 is a Schmitt Trigger input and an open drain output. Port RA4 is multiplexed with the T0CKI clock input. All other RA port pins have Schmitt Trigger input levels and full CMOS output drivers. All pins have data direction bits (TRIS registers), which can configure these pins as input or output.

A '1' in the TRISA register puts the corresponding output driver in a hi- impedance mode. A '0' in the TRISA register puts the contents of the output latch on the selected pin(s).

Reading the PORTA register reads the status of the pins, whereas writing to it will write to the port latch. All write operations are read-modify-write operations. So a write to a port implies that the port pins are first read, then this value is modified and written to the port data latch.

FIGURE 5-1: BLOCK DIAGRAM OF RA1:RA0 PINS

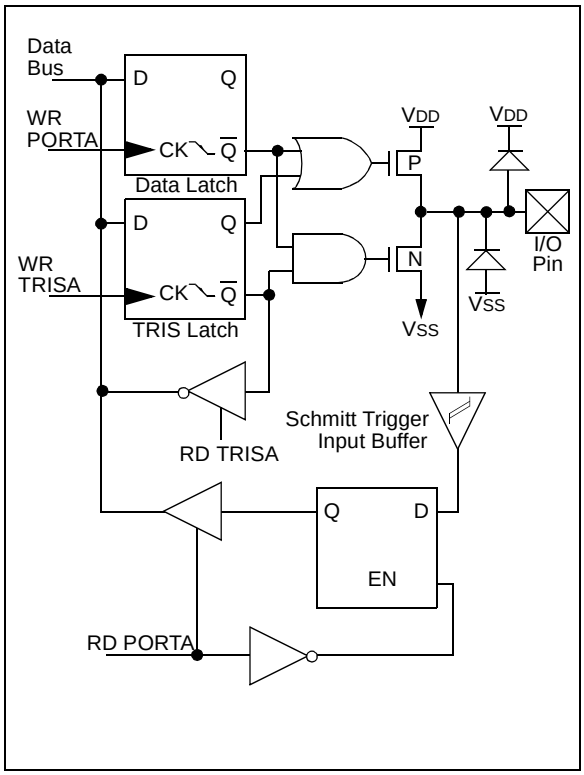


FIGURE 5-2: BLOCK DIAGRAM OF RA2 PIN

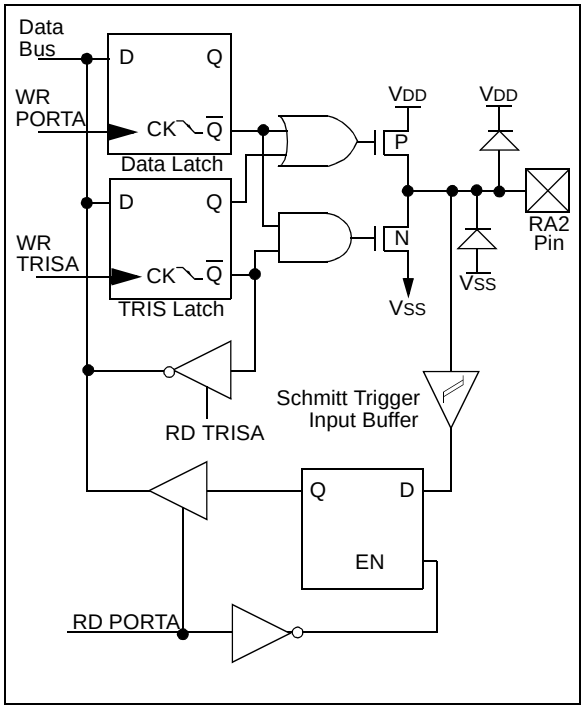


TABLE 5-1: PORTA FUNCTIONS

Name	Bit #	Buffer Type	Function
RA0	bit0	ST	Input/output
RA1	bit1	ST	Input/output
RA2	bit2	ST	Input/output
RA3	bit3	ST	Input/output
RA4/T0CKI	bit4	ST	Input/output or external clock input for TMR0.

Legend: ST = Schmitt Trigger input

TABLE 5-2: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on All Other Resets
05h	PORTA	—	—	—	RA4	RA3	RA2	RA1	RA0	---x 0000	---u 0000
85h	TRISA	—	—	—	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	---1 1111	---1 1111

Legend: — = Unimplemented locations, read as '0', u = unchanged, x = unknown

Note: Shaded bits are not used by PORTA.

FIGURE 5-3: BLOCK DIAGRAM OF RA3 PIN

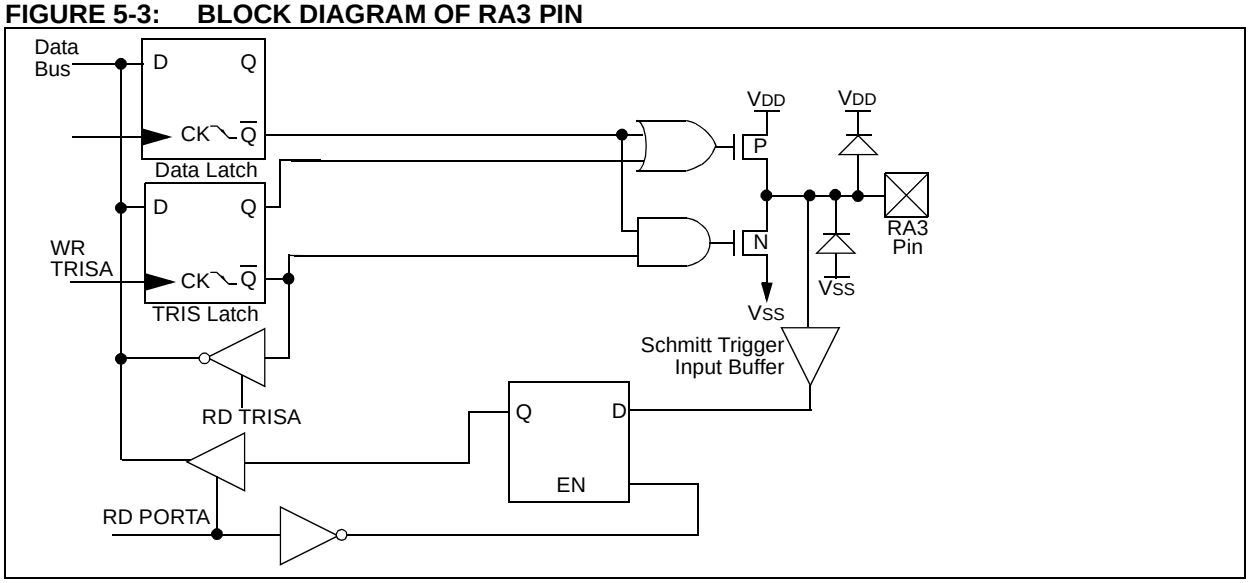
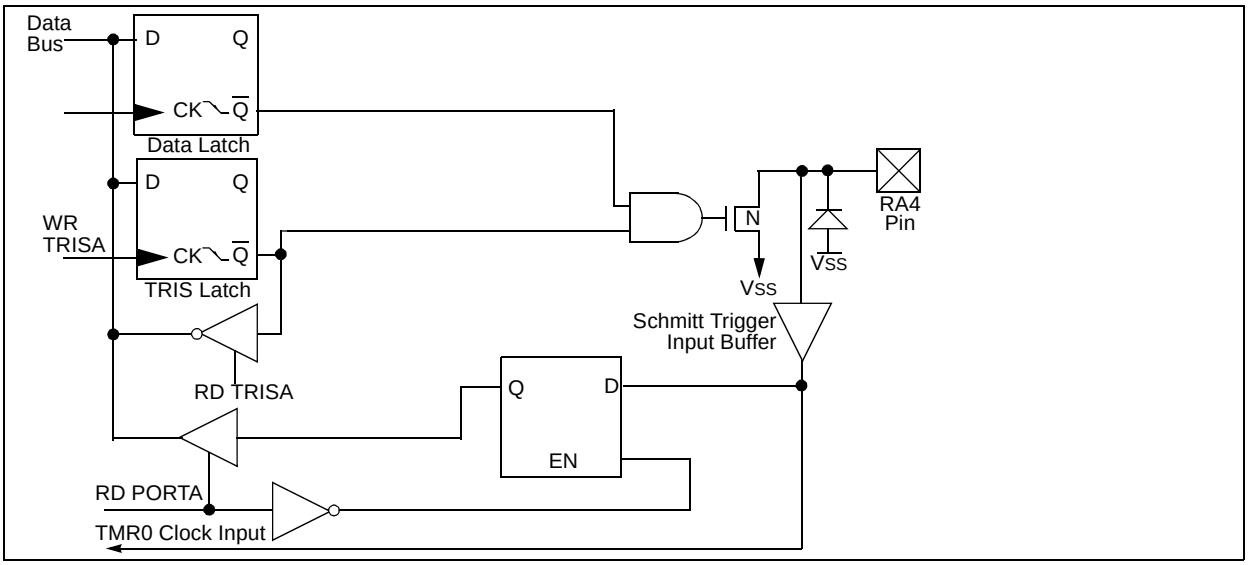


FIGURE 5-4: BLOCK DIAGRAM OF RA4 PIN



5.2 PORTB and TRISB Registers

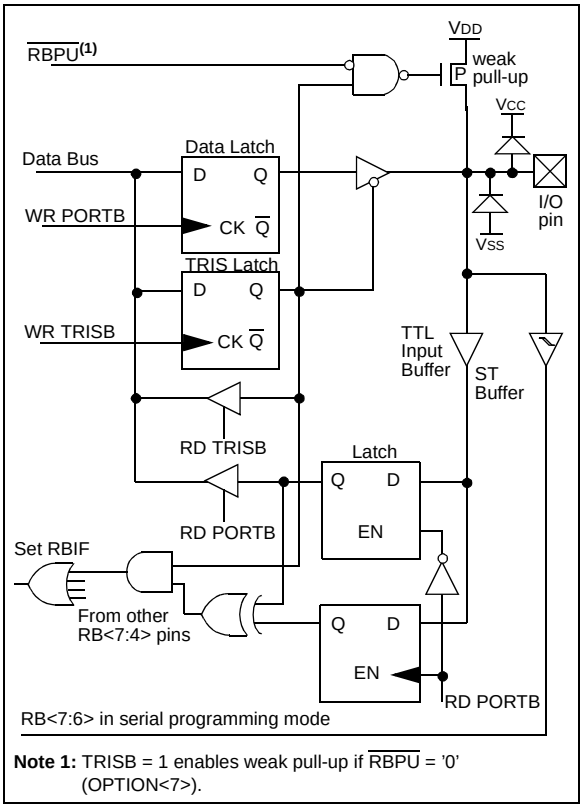
PORTB is an 8-bit wide, bi-directional port. The corresponding data direction register is TRISB. A '1' in the TRISB register puts the corresponding output driver in a high impedance mode. A '0' in the TRISB register puts the contents of the output latch on the selected pin(s).

Reading PORTB register reads the status of the pins, whereas writing to it will write to the port latch. All write operations are read-modify-write operations. So a write to a port implies that the port pins are first read, then this value is modified and written to the port data latch.

Each of the PORTB pins has a weak internal pull-up (≈200 μA typical). A single control bit can turn on all the pull-ups. This is done by clearing the $\overline{\text{RBPU}}$ (OPTION<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 Power-on Reset.

Four of PORTB's pins, RB<7:4>, have an interrupt on change feature. Only pins configured as inputs can cause this interrupt to occur (e.g., any RB<7:4> pin configured as an output is excluded from the interrupt on change comparison). The input pins (of RB<7:4>) are compared with the old value latched on the last read of PORTB. The "mismatch" outputs of RB<7:4> are OR'ed together to generate the RBIF interrupt (flag latched in INTCON<0>).

FIGURE 5-5: BLOCK DIAGRAM OF RB<7:4> PINS



Note 1: TRISB = 1 enables weak pull-up if $\overline{\text{RBPU}}$ = '0' (OPTION<7>).

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

a) Any read or write of PORTB. This will end the mismatch condition.

b) Clear flag bit RBIF.

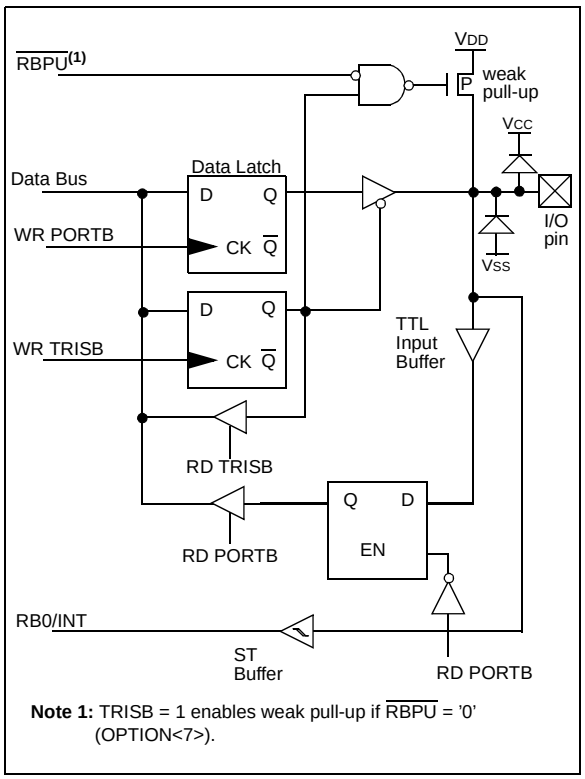
A mismatch condition will continue to set flag bit RBIF. Reading PORTB will end the mismatch condition, and allow flag bit RBIF to be cleared.

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, "Implementing Wake-Up on Key Strokes.)

Note:	If a change on the I/O pin should occur when the read operation is being executed (start of the Q2 cycle), then the RBIF interrupt flag may not get set.

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 5-6: BLOCK DIAGRAM OF RB<3:0> PINS



Note 1: TRISB = 1 enables weak pull-up if $\overline{\text{RBPU}}$ = '0' (OPTION<7>).

TABLE 5-3: PORTB FUNCTIONS

Name	Bit #	Buffer Type	Function
RB0/INT	bit0	TTL/ST(1)	Input/output or external interrupt input. Internal software programmable weak pull-up.
RB1	bit1	TTL	Input/output pin. Internal software programmable weak pull-up.
RB2	bit2	TTL	Input/output pin. Internal software programmable weak pull-up.
RB3	bit3	TTL	Input/output pin. Internal software programmable weak pull-up.
RB4	bit4	TTL	Input/output pin (with interrupt on change). Internal software programmable weak pull-up.
RB5	bit5	TTL	Input/output pin (with interrupt on change). Internal software programmable weak pull-up.
RB6	bit6	TTL/ST(2)	Input/output pin (with interrupt on change). Internal software programmable weak pull-up. Serial programming clock pin.
RB7	bit7	TTL/ST(2)	Input/output pin (with interrupt on change). Internal software programmable weak pull-up. Serial programming data pin.

Legend: ST = Schmitt Trigger, TTL = TTL input
Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.
Note 2: This buffer is a Schmitt Trigger input when used in serial programming mode.

TABLE 5-4: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on All Other Resets
06h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx	uuuu uuuu
86h	TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111	1111 1111
81h	OPTION	$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111

Note: Shaded bits are not used by PORTB.
u = unchanged
x = unknown

5.3 I/O Programming Considerations

5.3.1 BI-DIRECTIONAL I/O PORTS

Any instruction which writes, operates internally as a read followed by a write operation. The BCF and BSF instructions, for example, read the register into the CPU, execute the bit operation and write the result back to the register. Caution must be used when these instructions are applied to a port with both inputs and outputs defined. For example, a BSF operation on bit5 of PORTB will cause all eight bits of PORTB to be read into the CPU. Then the BSF operation takes place on bit5 and PORTB is written to the output latches. If another bit of PORTB is used as a bidirectional I/O pin (e.g., bit0) and it is defined as an input at this time, the input signal present on the pin itself would be read into the CPU and re-written to the data latch of this particular pin, overwriting the previous content. As long as the pin stays in the input mode, no problem occurs. However, if bit0 is switched into output mode later on, the content of the data latch may now be unknown.

Reading the port register, reads the values of the port pins. Writing to the port register writes the value to the port latch. When using read modify write instructions (ex. BCF, BSF, etc.) on a port, the value of the port pins is read, the desired operation is done to this value, and this value is then written to the port latch.

Example 5-1 shows the effect of two sequential read-modify-write instructions (ex., BCF, BSF, etc.) on an I/O port.

A pin actively outputting a Low or High should not be driven from external devices at the same time in order to change the level on this pin ("wired-or", "wired-and"). The resulting high output currents may damage the chip.

EXAMPLE 5-1: READ-MODIFY-WRITE INSTRUCTIONS ON AN I/O PORT

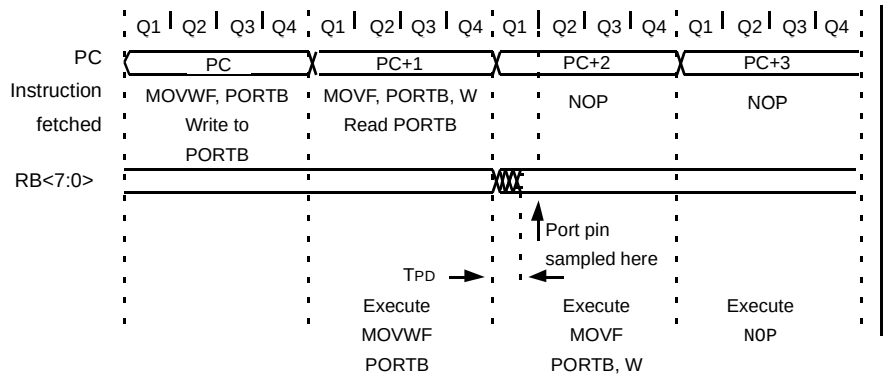
```
; Initial PORT settings:  PORTB<7:4> Inputs
;
;                         PORTB<3:0> Outputs
; PORTB<7:6> have external pull-up and are not
; connected to other circuitry
;
;                         PORT latch  PORT pins
;                         ----------  ----------

    BCF PORTB, 7        ; 01pp pppp   11pp pppp
    BCF PORTB, 6        ; 10pp pppp   11pp pppp
    BSF STATUS,RP0      ;
    BCF TRISB, 7        ; 10pp pppp   11pp pppp
    BCF TRISB, 6        ; 10pp pppp   10pp pppp
;
; Note that the user may have expected the pin
; values to be 00pp pppp. The 2nd BCF caused
; RB7 to be latched as the pin value (High).
```

5.3.2 SUCCESSIVE OPERATIONS ON I/O PORTS

The actual write to an I/O port happens at the end of an instruction cycle, whereas for reading, the data must be valid at the beginning of the instruction cycle (Figure 5-7). Therefore, care must be exercised if a write followed by a read operation is carried out on the same I/O port. The sequence of instructions should be such to allow the pin voltage to stabilize (load dependent) before the next instruction which causes that file to be read into the CPU is executed. Otherwise, the previous state of that pin may be read into the CPU rather than the new state. When in doubt, it is better to separate these instructions with a NOP or another instruction not accessing this I/O port.

FIGURE 5-7: SUCCESSIVE I/O OPERATION



Note:

This example shows write to PORTB followed by a read from PORTB.

Note that:

data setup time = (0.25 TCY - TPD) where TCY = instruction cycle and TPD = propagation delay of Q1 cycle to output valid.

Therefore, at higher clock frequencies, a write followed by a read may be problematic.

NOTES:

6.0 TIMER0 MODULE

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

Figure 6-1 is a simplified block diagram of the Timer0 module.

Timer mode is selected by clearing the T0CS bit (OPTION<5>). In timer mode, the TMR0 will increment every instruction cycle (without prescaler). If Timer0 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 TMR0.

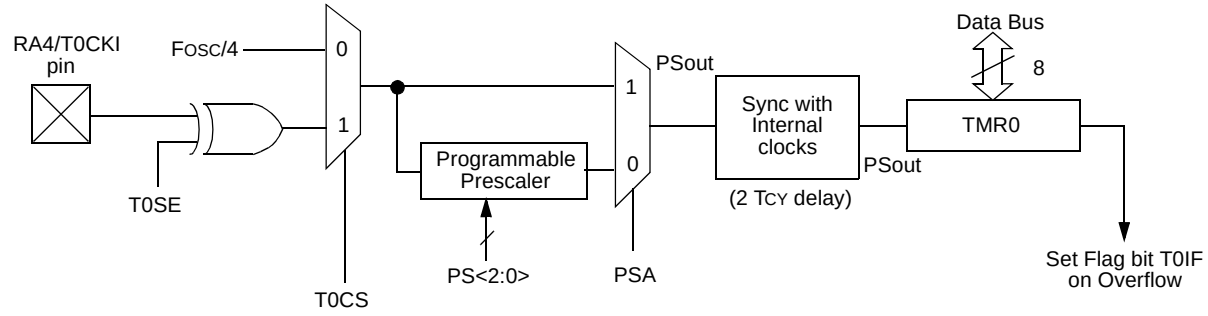
Counter mode is selected by setting the T0CS bit. In this mode, Timer0 will increment either on every rising or falling edge of pin RA4/T0CKI. The incrementing edge is determined by the source edge (T0SE) control bit (OPTION<4>). Clearing the T0SE bit selects the rising edge. Restrictions on the external 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 the control bit PSA (OPTION<3>). Clearing the PSA bit will assign the prescaler to Timer0. The prescaler is not readable or writable. When the prescaler is assigned to the Timer0 module, prescale value of 1:2, 1:4, ..., 1:256 are selectable. Section 6.3 details the operation of the prescaler.

6.1 TIMER0 Interrupt

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

FIGURE 6-1: TIMER0 BLOCK DIAGRAM



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

FIGURE 6-2: TIMER0 (TMR0) TIMING: INTERNAL CLOCK/NO PRESCALER

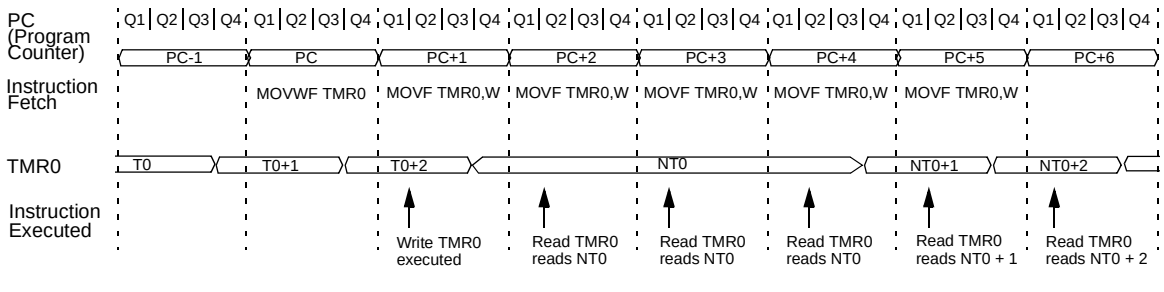


FIGURE 6-3: TIMER0 TIMING: INTERNAL CLOCK/PRESCALE 1:2

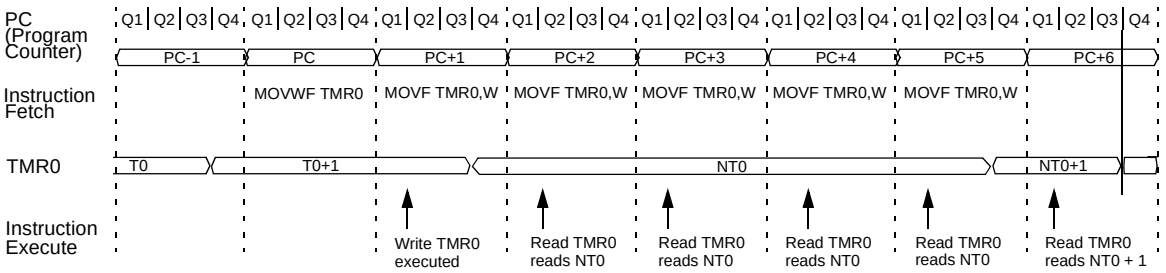
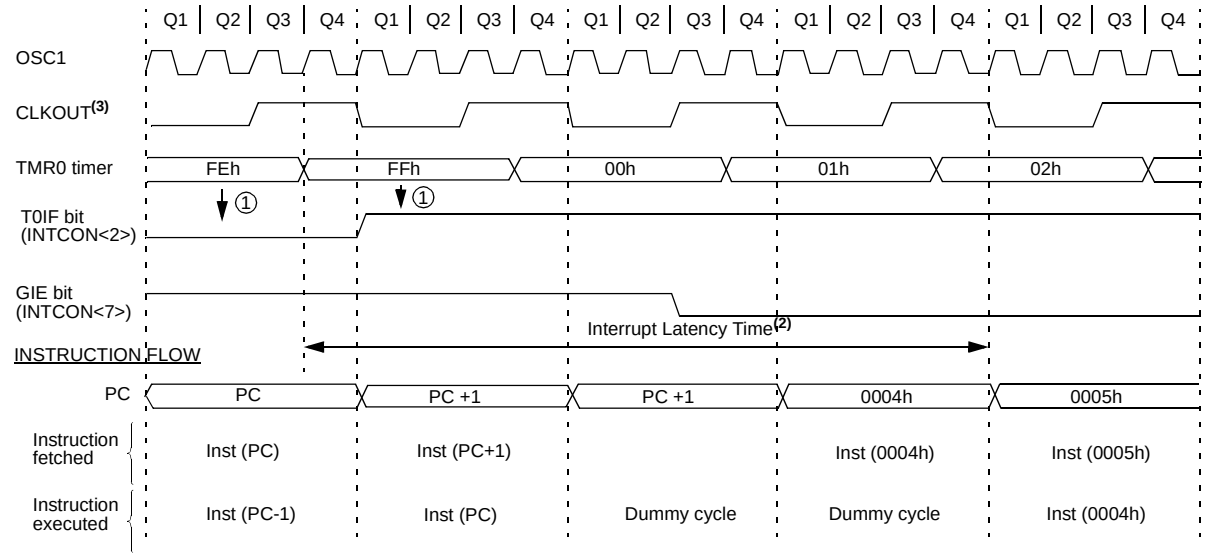


FIGURE 6-4: TIMER0 INTERRUPT TIMING



Note 1: T0IF interrupt flag is sampled here (every Q1).
Note 2: Interrupt latency = 3TCY, where TCY = instruction cycle time.
Note 3: CLKOUT is available only in RC oscillator mode.

6.2 Using Timer0 with External Clock

When an external clock input is used for Timer0, it must meet certain requirements. The external clock requirement is due to internal phase clock (TOSC) synchronization. Also, there is a delay in the actual incrementing of Timer0 after synchronization.

6.2.1 EXTERNAL CLOCK SYNCHRONIZATION

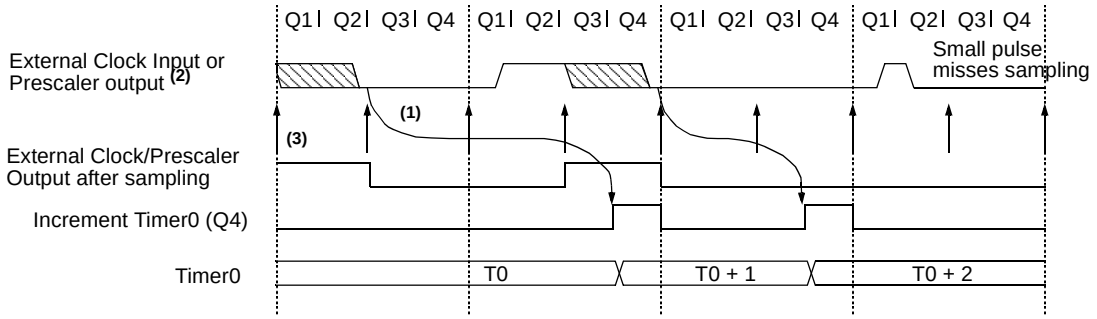
When no prescaler is used, the external clock input is the same as the prescaler output. The synchronization of T0CKI with the internal phase clocks is accomplished by sampling the prescaler output on the Q2 and Q4 cycles of the internal phase clocks (Figure 6-5). Therefore, it is necessary for T0CKI to be high for at least 2TOSC (and a small RC delay of 20 ns) and low for at least 2TOSC (and a small RC delay of 20 ns). Refer to the electrical specification of the desired device.

When a prescaler is used, the external clock input is divided by the asynchronous ripple-counter type prescaler, so that the prescaler output is symmetrical. For the external clock to meet the sampling requirement, the ripple-counter must be taken into account. Therefore, it is necessary for T0CKI to have a period of at least 4TOSC (and a small RC delay of 40 ns) divided by the prescaler value. The only requirement on T0CKI high and low time is that they do not violate the minimum pulse width requirement of 10 ns. Refer to parameters 40, 41 and 42 in the electrical specification of the desired device.

6.2.2 TIMER0 INCREMENT DELAY

Since the prescaler output is synchronized with the internal clocks, there is a small delay from the time the external clock edge occurs to the time the TMR0 is actually incremented. Figure 6-5 shows the delay from the external clock edge to the timer incrementing.

FIGURE 6-5: TIMER0 TIMING WITH EXTERNAL CLOCK



Note 1: Delay from clock input change to Timer0 increment is 3Tosc to 7Tosc. (Duration of Q = Tosc). Therefore, the error in measuring the interval between two edges on Timer0 input = ±4Tosc max.
Note 2: External clock if no prescaler selected, Prescaler output otherwise.
Note 3: The arrows indicate the points in time where sampling occurs.

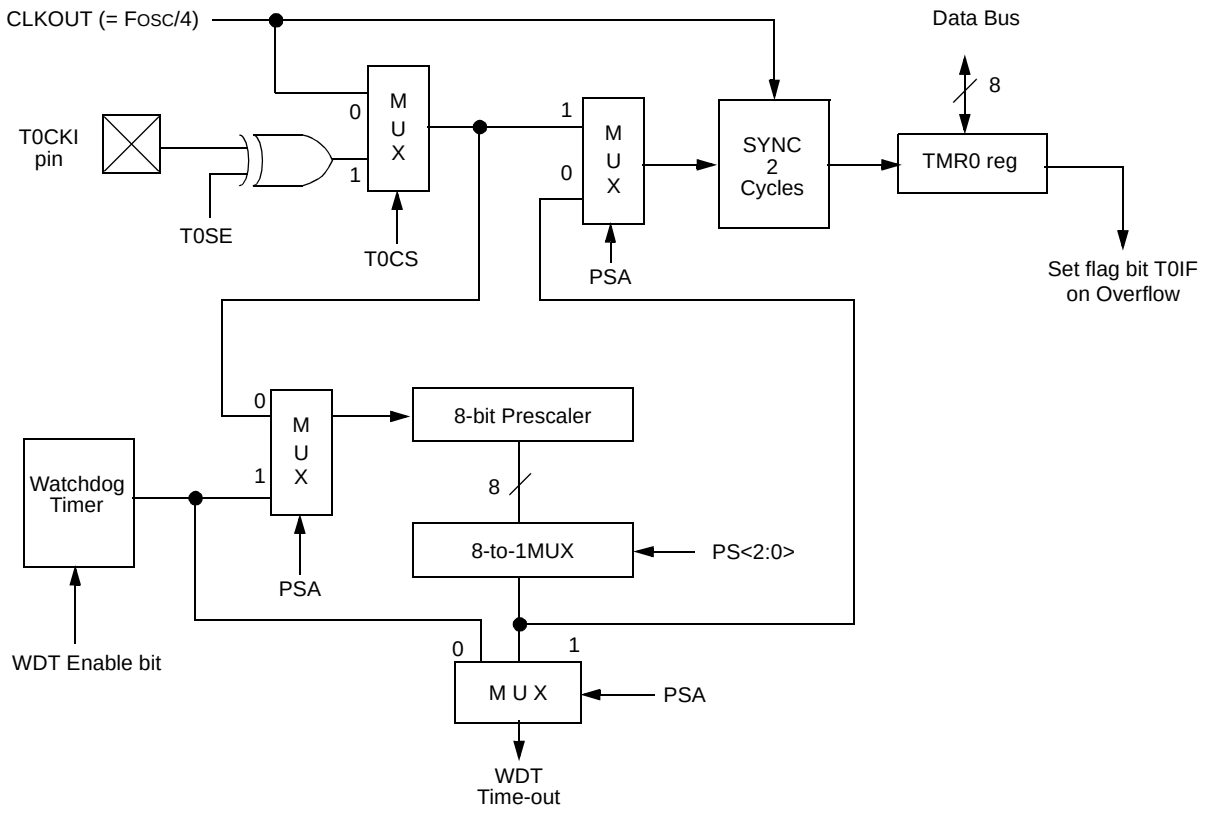
6.3 Prescaler

An 8-bit counter is available as a prescaler for the Timer0 module, or as a postscaler for the Watchdog Timer, respectively (Figure 6-6). For simplicity, this counter is being referred to as “prescaler” throughout this data sheet. Note that there is only one prescaler available which is mutually exclusive between the Timer0 module and the Watchdog Timer. Thus, a prescaler assignment for the Timer0 module means that there is no prescaler for the Watchdog Timer, and vice-versa.

The PSA and PS<2:0> bits (OPTION<3:0>) determine the prescaler assignment and prescale ratio.

When assigned to the Timer0 module, all instructions writing to the TMR0 register (e.g., `CLRF 1`, `MOVWF 1`, `BSF 1,x`....etc.) will clear the prescaler. When assigned to WDT, a `CLRWDT` instruction will clear the prescaler along with the Watchdog Timer. The prescaler is not readable or writable.

FIGURE 6-6: BLOCK DIAGRAM OF THE TIMER0/WDT PRESCALER



Note: T0SE, T0CS, PSA, PS<2:0> are bits in the OPTION register.

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). To avoid an unintended device RESET, the following instruction sequence (Example 6-1) must be executed when changing the prescaler assignment from Timer0 to WDT.

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

```
 1.BCF    STATUS, RP0  ;Skip if already in
                       ; Bank 0
 2.CLRWDT              ;Clear WDT
 3.CLRF   TMR0         ;Clear TMR0 & Prescaler
 4.BSF    STATUS, RP0  ;Bank 1
 5.MOVLW  '00101111'b; ;These 3 lines (5, 6, 7)
 6.MOVWF  OPTION       ; are required only if
                       ; desired PS<2:0> are
 7.CLRWDT              ; 000 or 001
 8.MOVLW  '00101xxx'b  ;Set Postscaler to
 9.MOVWF  OPTION       ; desired WDT rate
10.BCF    STATUS, RP0  ;Return to Bank 0
```

To change prescaler from the WDT to the TMR0 module, use the sequence shown in Example 6-2. This precaution must be taken even if the WDT is disabled.

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

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

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 POR	Value on All Other Resets
01h	TMR0	Timer0 module register								xxxx xxxx	uuuu uuuu
0Bh/8Bh	INTCON	GIE	—	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
81h	OPTION	$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111
85h	TRISA	—	—	—	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	---1 1111	---1 1111

Legend: — = Unimplemented locations, read as ‘0’.

Note: Shaded bits are not used by TMR0 module.
u = unchanged
x = unknown

NOTES:

7.0 SPECIAL FEATURES OF THE CPU

Special circuits to deal with the needs of real time applications are what sets a microcontroller apart from other processors. The CF76XX family has a host of such features intended to maximize system reliability, minimize cost through elimination of external components, provide power saving operating modes and offer code protection.

These are:

1. OSC selection
2. Reset
   Power-on Reset (POR)
   Power-up Timer (PWRT)
   Oscillator Start-Up Timer (OST)
3. Interrupts
4. Watchdog Timer (WDT)
5. SLEEP
6. Code protection
7. ID Locations
8. In-circuit serial programming

The CF76XX devices have a Watchdog Timer, which is controlled by configuration bits. It runs off its own RC oscillator for added reliability. There are two timers that offer necessary delays on power-up. One is the Oscillator Start-up Timer (OST), intended to keep the chip in reset until the crystal oscillator is stable. The other is the Power-up Timer (PWRT), which provides a fixed delay of 72 ms (nominal) on power-up only, designed to keep the part in reset while the power supply stabilizes. There is also circuitry to reset the device if a brown-out occurs, which provides at least a 72 ms reset. With these three functions on-chip, most applications need no external reset circuitry.

The SLEEP mode is designed to offer a very low current power-down mode. The user can wake-up from SLEEP through external reset, Watchdog Timer wake-up or through an interrupt. Several oscillator options are also made available to allow the part to fit the application. The RC oscillator option saves system cost, while the LP crystal option saves power. A set of configuration bits are used to select various options.

7.1 Configuration Bits

The configuration bits can be programmed (read as '0') or left unprogrammed (read as '1') to select various device configurations. These bits are mapped in program memory location 2007h.

The user will note that address 2007h is beyond the user program memory space. In fact, it belongs to the special test/configuration memory space (2000h – 3FFFh), which can be accessed only during programming.

FIGURE 7-1: CONFIGURATION WORD

CP1	CP0(2)	CP1	CP0(2)	CP1	CP0(2)	—	—	CP1	CP0(2)	$\overline{\text{PWRTE}}$(1)	WDTE	F0SC1	F0SC0
bit13													bit0

CONFIG Address
REGISTER: 2007h

bit 13-8, 5-4: **CP<1:0>:** Code protection bit pairs(2)
Code protection for 2K program memory
11 = Program memory code protection off
10 = 0400h-07FFh code protected
01 = 0200h-07FFh code protected
00 = 0000h-07FFh code protected
Code protection for 1K program memory
11 = Program memory code protection off
10 = Program memory code protection off
01 = 0200h-03FFh code protected
00 = 0000h-03FFh code protected
Code protection for 0.5K program memory
11 = Program memory code protection off
10 = Program memory code protection off
01 = Program memory code protection off
00 = 0000h-01FFh code protected

bit 7: **Unimplemented**: Read as '1'

bit 6: **RESERVED: Do Not Use**: (1)

bit 3: **$\overline{\text{PWRTE}}$**: Power-up Timer Enable bit (1, 3)
1 = PWRT disabled
0 = PWRT enabled

bit 2: **WDTE**: Watchdog Timer Enable bit
1 = WDT enabled
0 = WDT disabled

bit 1-0: **FOSC1:FOSC0**: Oscillator Selection bits
11 = RC oscillator
10 = HS oscillator
01 = XT oscillator
00 = LP oscillator

Note 1: All of the CP<1:0> pairs have to be given the same value to enable the code protection scheme listed.

7.2 Oscillator Configurations

7.2.1 OSCILLATOR TYPES

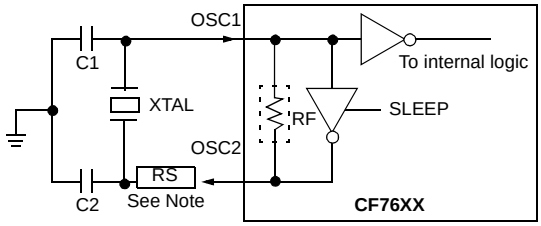
The CF76XX devices can be operated in four different oscillator options. The user can program two configuration bits (FOSC1 and FOSC0) to select one of these four modes:

- LP Low Power Crystal
- XT Crystal/Resonator
- HS High Speed Crystal/Resonator
- RC Resistor/Capacitor

7.2.2 CRYSTAL OSCILLATOR / CERAMIC RESONATORS

In XT, LP or HS modes, a crystal or ceramic resonator is connected to the OSC1 and OSC2 pins to establish oscillation (Figure 7-2). The CF76XX oscillator design requires the use of a parallel cut crystal. Use of a series cut crystal may give a frequency out of the crystal manufacturers specifications. When in XT, LP or HS modes, the device can have an external clock source to drive the OSC1 pin (Figure 7-3).

FIGURE 7-2: CRYSTAL OPERATION (OR CERAMIC RESONATOR) (HS, XT OR LP OSC CONFIGURATION)



See Table 7-1 and Table 7-2 for recommended values of C1 and C2.

Note: A series resistor may be required for AT strip cut crystals.

FIGURE 7-3: EXTERNAL CLOCK INPUT OPERATION (HS, XT OR LP OSC CONFIGURATION)

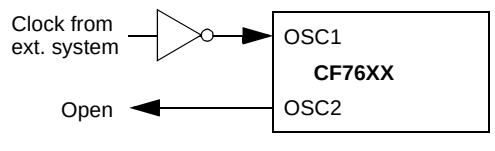


TABLE 7-1: CAPACITOR SELECTION FOR CERAMIC RESONATORS

Ranges Characterized:			
Mode	**Freq**	**OSC1(C1)**	**OSC2(C2)**
XT	455 kHz 2.0 MHz 4.0 MHz	22 - 100 pF 15 - 68 pF 15 - 68 pF	22 - 100 pF 15 - 68 pF 15 - 68 pF
HS	8.0 MHz 16.0 MHz	10 - 68 pF 10 - 22 pF	10 - 68 pF 10 - 22 pF
Higher capacitance increases the stability of the oscillator but also increases the start-up time. These values are for design guidance only. Since each resonator has its own characteristics, the user should consult the resonator manufacturer for appropriate values of external components.			

TABLE 7-2: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR

Mode	Freq	OSC1(C1)	OSC2(C2)
LP	32 kHz 200 kHz	68 - 100 pF 15 - 30 pF	68 - 100 pF 15 - 30 pF
XT	100 kHz 2 MHz 4 MHz	68 - 150 pF 15 - 30 pF 15 - 30 pF	150 - 200 pF 15 - 30 pF 15 - 30 pF
HS	8 MHz 10 MHz 20 MHz	15 - 30 pF 15 - 30 pF 15 - 30 pF	15 - 30 pF 15 - 30 pF 15 - 30 pF
Higher capacitance increases the stability of the oscillator but also increases the start-up time. These values are for design guidance only. Rs may be required in HS mode as well as XT mode to avoid overdriving crystals with low drive level specification. Since each crystal has its own characteristics, the user should consult the crystal manufacturer for appropriate values of external components.			

7.2.3 EXTERNAL CRYSTAL OSCILLATOR CIRCUIT

Either a prepackaged oscillator can be used or a simple oscillator circuit with TTL gates can be built. Prepackaged oscillators provide a wide operating range and better stability. A well-designed crystal oscillator will provide good performance with TTL gates. Two types of crystal oscillator circuits can be used; one with series resonance or one with parallel resonance.

Figure 7-4 shows implementation of a parallel resonant oscillator circuit. The circuit is designed to use the fundamental frequency of the crystal. The 74AS04 inverter performs the 180° phase shift that a parallel oscillator requires. The 4.7 kΩ resistor provides the negative feedback for stability. The 10 kΩ potentiometers bias the 74AS04 in the linear region. This could be used for external oscillator designs.

FIGURE 7-4: EXTERNAL PARALLEL RESONANT CRYSTAL OSCILLATOR CIRCUIT

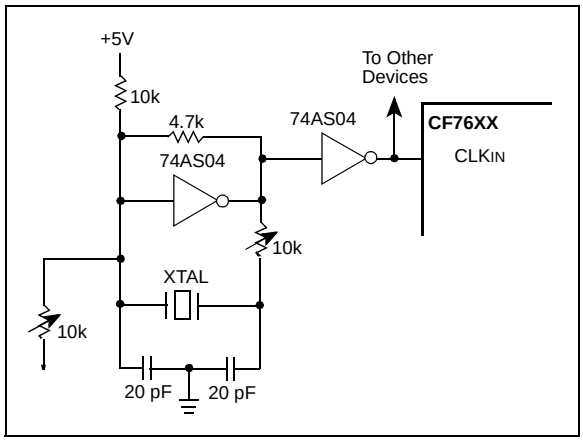
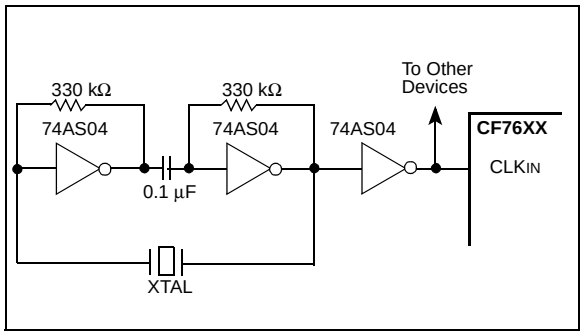


Figure 7-5 shows a series resonant oscillator circuit. This circuit is also designed to use the fundamental frequency of the crystal. The inverter performs a 180° phase shift in a series resonant oscillator circuit. The 330 kΩ resistors provide the negative feedback to bias the inverters in their linear region.

FIGURE 7-5: EXTERNAL SERIES RESONANT CRYSTAL OSCILLATOR CIRCUIT



7.2.4 RC OSCILLATOR

For timing insensitive applications the “RC” device option offers additional cost savings. The RC oscillator frequency is a function of the supply voltage, the resistor (Rext) and capacitor (Cext) values, and the operating temperature. In addition to this, the oscillator frequency will vary from unit to unit due to normal process parameter variation. Furthermore, the difference in lead frame capacitance between package types will also affect the oscillation frequency, especially for low Cext values. The user also needs to take into account variation due to tolerance of external R and C components used. Figure 7-6 shows how the R/C combination is connected to the CF76XX. For Rext values below 2.2 kΩ, the oscillator operation may become unstable or stop completely. For very high Rext values (e.g., 1 MΩ), the oscillator becomes sensitive to noise, humidity and leakage. Thus, we recommend to keep Rext between 3 kΩ and 100 kΩ.

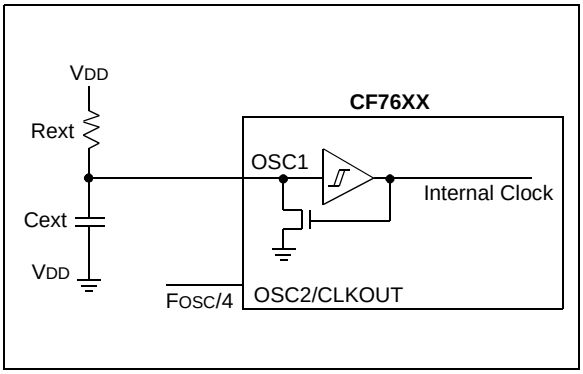
Although the oscillator will operate with no external capacitor (Cext = 0 pF), we recommend using values above 20 pF for noise and stability reasons. With no or small external capacitance, the oscillation frequency can vary dramatically due to changes in external capacitances, such as PCB trace capacitance or package lead frame capacitance.

See Section 10.0 for RC frequency variation from part to part due to normal process variation. The variation is larger for larger R (since leakage current variation will affect RC frequency more for large R) and for smaller C (since variation of input capacitance will affect RC frequency more).

See Section 10.0 for variation of oscillator frequency due to VDD for given Rext/Cext values, as well as frequency variation due to operating temperature for given R, C and VDD values.

The oscillator frequency, divided by 4, is available on the OSC2/CLKOUT pin, and can be used for test purposes or to synchronize other logic (Figure 3-2 for waveform).

FIGURE 7-6: RC OSCILLATOR MODE



7.3 Reset

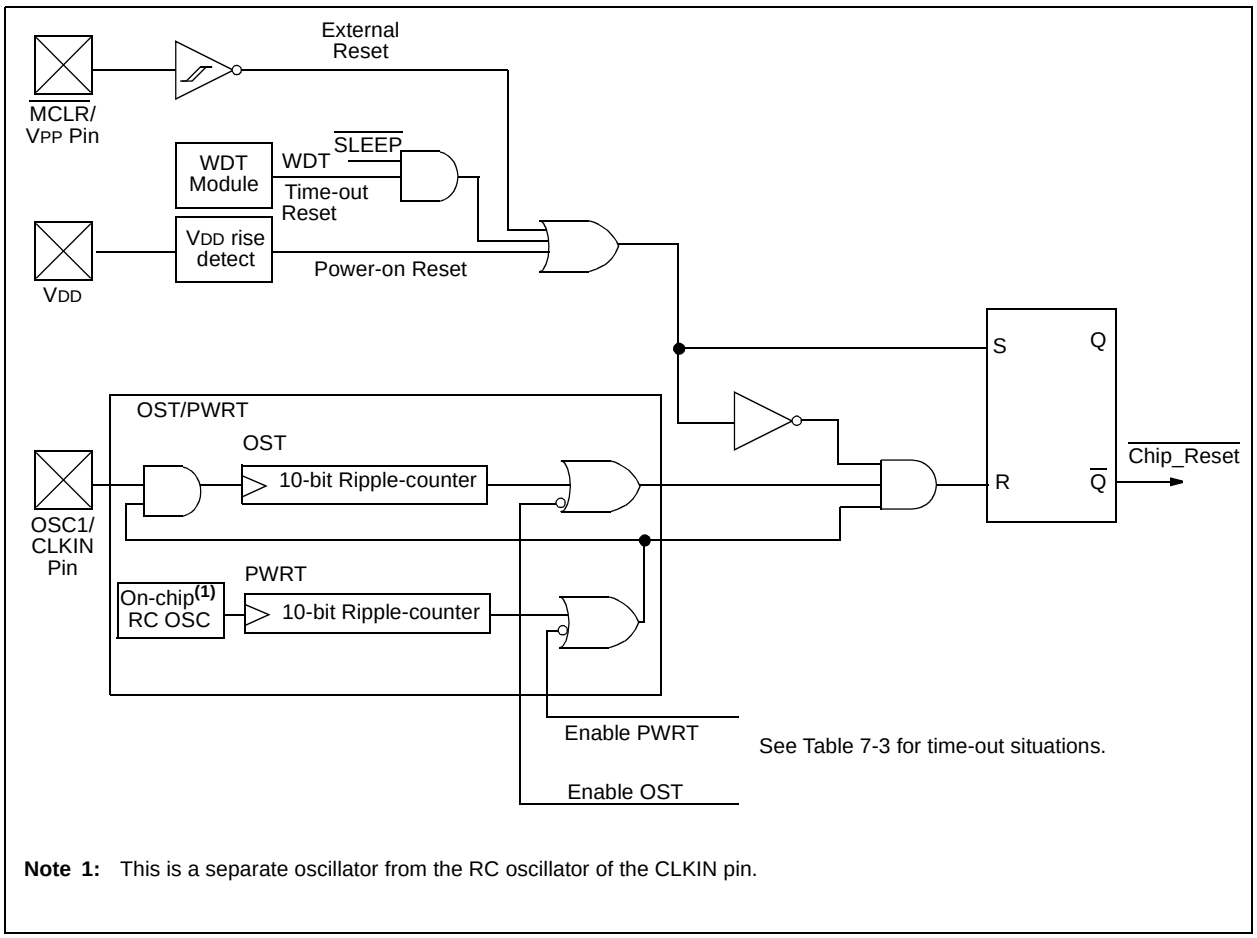
The CF76XX differentiates between various kinds of reset:

a) Power-on reset (POR)
b) $\overline{MCLR}$ reset during normal operation
c) $\overline{MCLR}$ reset during SLEEP
d) WDT reset (normal operation)
e) WDT wake-up (SLEEP)

Some registers are not affected in any reset condition Their status is unknown on POR and unchanged in any other reset. Most other registers are reset to a “reset state” on Power-on reset, $\overline{MCLR}$ reset, WDT reset and $\overline{MCLR}$ reset during SLEEP. They are not affected by a WDT wake-up, since this is viewed as the resumption of normal operation. $\overline{TO}$ and $\overline{PD}$ bits are set or cleared differently in different reset situations as indicated in Table 7-4. These bits are used in software to determine the nature of the reset. See Table 7-7 for a full description of reset states of all registers.

A simplified block diagram of the on-chip reset circuit is shown in Figure .

The $\overline{MCLR}$ reset path has a noise filter to detect and ignore small pulses. Simplified Block Diagram of On-chip Reset Circuit



Note 1: This is a separate oscillator from the RC oscillator of the CLKIN pin.

7.4 Power-on Reset (POR), Power-up Timer (PWRT), and Oscillator Start-up Timer (OST)

7.4.1 POWER-ON RESET (POR)

The on-chip POR circuit holds the chip in reset until VDD has reached a high enough level for proper operation. To take advantage of the POR, just tie the MCLR pin through a resistor to VDD. This will eliminate external RC components usually needed to create Power-on Reset. A maximum rise time for VDD is required. See Electrical Specifications for details.

The POR circuit does not produce an internal reset when VDD declines.

When the device starts normal operation (exits the reset condition), device operating parameters (voltage, frequency, temperature, etc.) must be met to ensure operation. If these conditions are not met, the device must be held in reset until the operating conditions are met.

For additional information, refer to Application Note AN607, “Power-up Trouble Shooting”.

7.4.2 POWER-UP TIMER (PWRT)

The Power-up Timer provides a fixed 72 ms (nominal) time-out on power-up only, from POR or Brown-out Reset. The Power-up Timer operates on an internal RC oscillator. The chip is kept in reset as long as PWRT is active. The PWRT delay allows the VDD to rise to an acceptable level. A configuration bit, $\overline{\text{PWRTE}}$ can disable (if set) or enable (if cleared or programmed) the Power-up Timer. The Power-Up Time delay will vary from chip to chip and due to VDD, temperature and process variation. See DC parameters for details.

7.4.3 OSCILLATOR START-UP TIMER (OST)

The Oscillator Start-Up Timer (OST) provides a 1024 oscillator cycle (from OSC1 input) delay after the PWRT delay is over. This ensures that the crystal oscillator or resonator has started and stabilized.

The OST time-out is invoked only for XT, LP and HS modes and only on power-on reset or wake-up from sleep.

7.4.4 TIME-OUT SEQUENCE

On power-up the time-out sequence is as follows: First PWRT time-out is invoked after POR has expired. Then OST is activated. The total time-out will vary based on oscillator configuration and $\overline{\text{PWRTE}}$ bit status. For example, in RC mode with $\overline{\text{PWRTE}}$ bit erased (PWRT disabled), there will be no time-out at all. Figure 7-7, Figure 7-8 and Figure 7-9 depict time-out sequences.

Since the time-outs occur from the POR pulse, if $\overline{\text{MCLR}}$ is kept low long enough, the time-outs will expire. Then bringing $\overline{\text{MCLR}}$ high will begin execution immediately (see Figure 7-8). This is useful for testing purposes or to synchronize more than one CF76XX device operating in parallel.

Table 7-6 shows the reset conditions for some special registers, while Table 7-7 shows the reset conditions for all the registers.

7.4.5 POWER CONTROL (PCON)/ STATUS REGISTER

The power control/status register, PCON (address 8Eh), has one bit.

Bit0 is Reserved.

Bit1 is $\overline{\text{POR}}$ (Power-on-reset). It is a ‘0’ on power-on-reset and unaffected otherwise. The user must write a ‘1’ to this bit following a power-on-reset. On a subsequent reset, if $\overline{\text{POR}}$ is ‘0’, it will indicate that a power-on-reset must have occurred (VDD may have gone too low).

TABLE 7-3: TIME-OUT IN VARIOUS SITUATIONS

Oscillator Configuration	Power-up		Wake-up from SLEEP
	$\overline{\text{PWRTE}}$ = 0	$\overline{\text{PWRTE}}$ = 1	
XT, HS, LP	72 ms + 1024 TOSC	1024 TOSC	1024 TOSC
RC	72 ms	—	—

TABLE 7-4: STATUS/PCON BITS AND THEIR SIGNIFICANCE

$\overline{POR}$	$\overline{TO}$	$\overline{PD}$	
0	1	1	Power-on-reset
0	0	X	Illegal, $\overline{TO}$ is set on $\overline{POR}$
0	X	0	Illegal, $\overline{PD}$ is set on $\overline{POR}$
1	0	u	WDT Reset
1	0	0	WDT Wake-up
1	u	u	$\overline{MCLR}$ reset during normal operation
1	1	0	$\overline{MCLR}$ reset during SLEEP

Legend: u = unchanged, x = unknown

TABLE 7-5: SUMMARY OF REGISTERS ASSOCIATED WITH BROWN-OUT

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR Reset	Value on all other resets(1)
83h	STATUS				$\overline{TO}$	$\overline{PD}$				0001 1xxx	000q quuu
8Eh	PCON	—	—	—	—	—	—	$\overline{POR}$	—	---- --0-	---- --u-

Legend: u = unchanged, x = unknown, - = unimplemented bit, reads as '0', q = value depends on condition.

Note 1: Other (non-power-up) resets include $\overline{MCLR}$ reset and Watchdog Timer Reset during normal operation.

TABLE 7-6: INITIALIZATION CONDITION FOR SPECIAL REGISTERS

Condition	Program Counter	STATUS Register	PCON Register
Power-on Reset	000h	0001 1xxx	---- --0-
$\overline{MCLR}$ reset during normal operation	000h	000u uuuu	---- --u-
$\overline{MCLR}$ reset during SLEEP	000h	0001 0uuu	---- --u-
WDT reset	000h	0000 uuuu	---- --u-
WDT Wake-up	PC + 1	uuu0 0uuu	---- --u-
Interrupt Wake-up from SLEEP	PC + 1(1)	uuu1 0uuu	---- --u-

Legend: u = unchanged, x = unknown, - = unimplemented bit, reads as '0'.

Note 1: When the wake-up is due to an interrupt and global enable bit, GIE is set, the PC is loaded with the interrupt vector (0004h) after execution of PC+1.

TABLE 7-7: INITIALIZATION CONDITION FOR REGISTERS

Register	Address	Power-on Reset	• $\overline{MCLR}$ Reset during normal operation • MCLR Reset during SLEEP • WDT Reset	• Wake up from SLEEP through interrupt • Wake up from SLEEP through WDT time-out
W	-	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF	00h	-	-	-
TMR0	01h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCL	02h	0000 0000	0000 0000	PC + 1(2)
STATUS	03h	0001 1xxx	000q quuu(3)	uuuq quuu(3)
FSR	04h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA	05h	---x xxxx	---u uuuu	---u uuuu
PORTB	06h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCLATH	0Ah	---0 0000	---0 0000	---u uuuu
INTCON	0Bh	0-00 000x	0000 000u	uuuu uqqq(1)
OPTION	81h	1111 1111	1111 1111	uuuu uuuu
TRISA	85h	---1 1111	---1 1111	---u uuuu
TRISB	86h	1111 1111	1111 1111	uuuu uuuu
PCON	8Eh	---- --0-	---- --u(1-3)-	---- --u-

Legend: u = unchanged, x = unknown, - = unimplemented bit, reads as '0', q = value depends on condition.

Note 1: One or more bits in INTCON will be affected to cause wake up.

Note 2: When the wake-up is due to an interrupt and the GIE bit is set, the PC is loaded with the interrupt vector (0004h).

Note 3: See Table 7-6 for reset value for specific condition.

FIGURE 7-7: TIME-OUT SEQUENCE ON POWER-UP (MCLR NOT TIED TO VDD): CASE 1

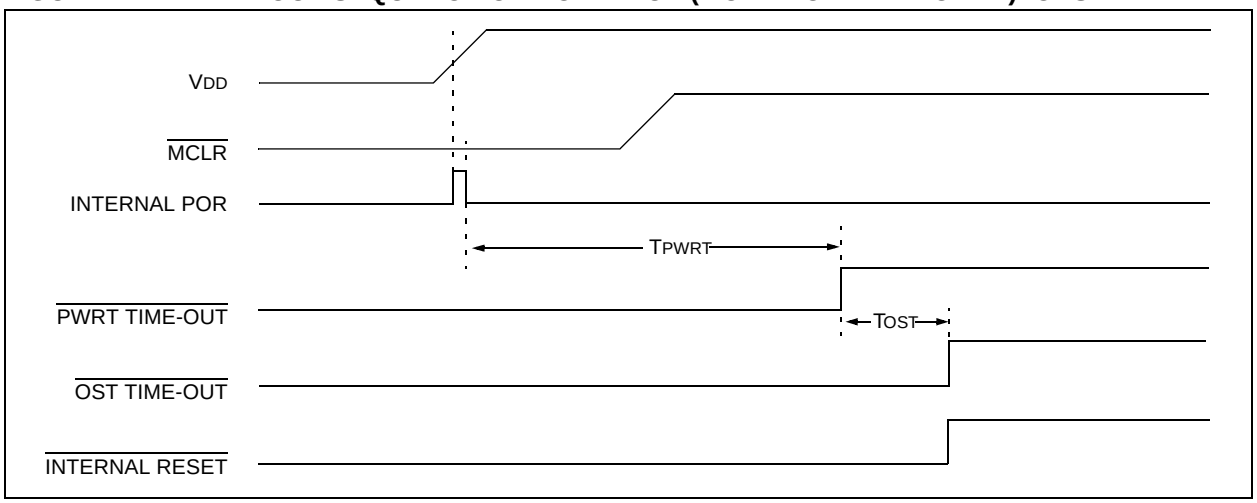


FIGURE 7-8: TIME-OUT SEQUENCE ON POWER-UP (MCLR NOT TIED TO VDD): CASE 2

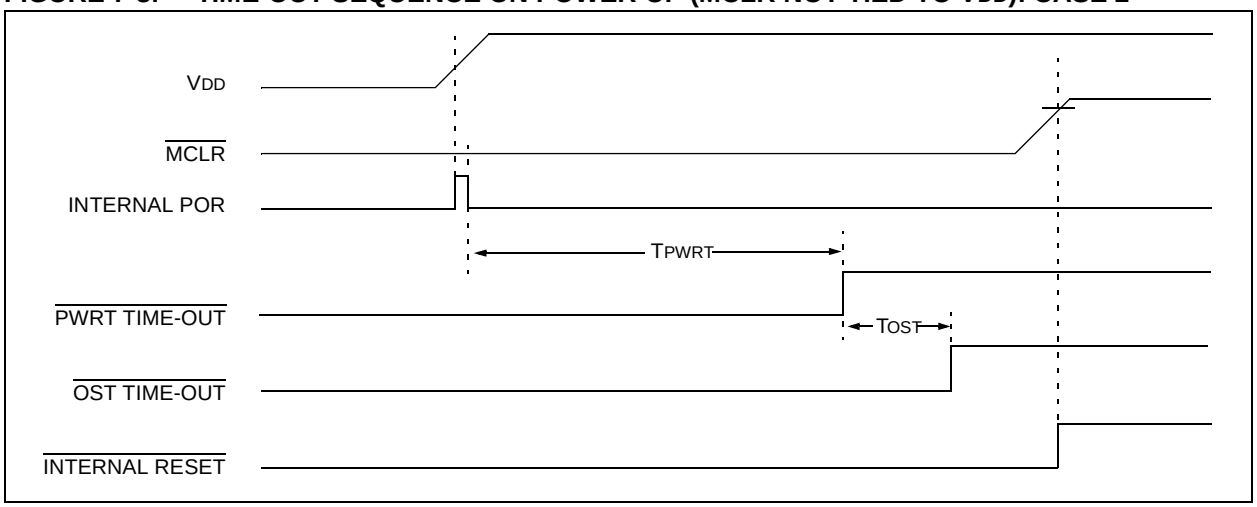


FIGURE 7-9: TIME-OUT SEQUENCE ON POWER-UP (MCLR TIED TO VDD)

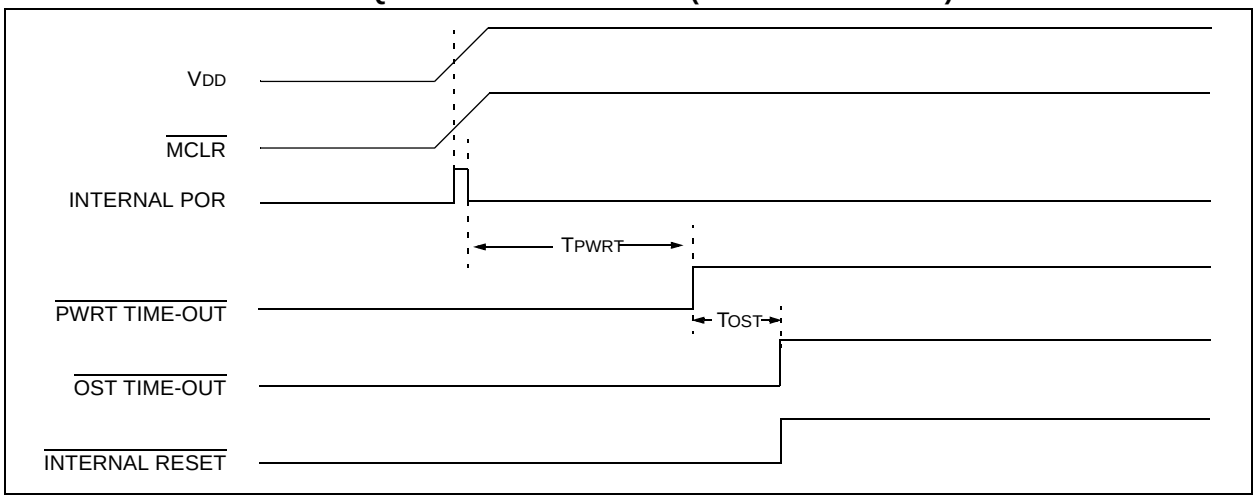
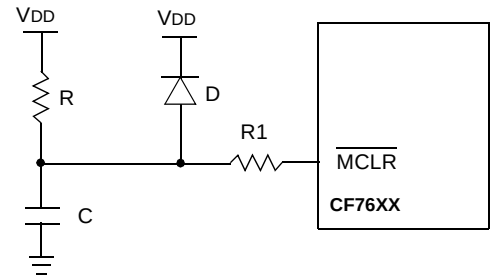


FIGURE 7-10: EXTERNAL POWER-ON RESET CIRCUIT (FOR SLOW VDD POWER-UP)



Note 1: External power-on reset circuit is required only if VDD power-up slope is too slow. The diode D helps discharge the capacitor quickly when VDD powers down.

Note 2: < 40 kΩ is recommended to make sure that voltage drop across R does not violate the device's electrical specification.

Note 3: R1 = 100Ω to 1 kΩ will limit any current flowing into MCLR from external capacitor C in the event of MCLR/VPP pin breakdown due to Electrostatic Discharge (ESD) or Electrical Overstress (EOS).

7.5 Interrupts

The CF76XX has 4 sources of interrupt:

- External interrupt RB0/INT
- TMR0 overflow interrupt
- PORTB change interrupts (pins RB<7:4>)

The interrupt control register (INTCON) records individual interrupt requests in flag bits. It also has individual and global interrupt enable bits.

A global interrupt enable bit, GIE (INTCON<7>) enables (if set) all un-masked interrupts or disables (if cleared) all interrupts. Individual interrupts can be disabled through their corresponding enable bits in INTCON register. GIE is cleared on reset.

The "return from interrupt" instruction, RETFIE, exits interrupt routine, as well as sets the GIE bit, which re-enable RB0/INT interrupts.

The INT pin interrupt, the RB port change interrupt and the TMR0 overflow interrupt flags are contained in the INTCON register.

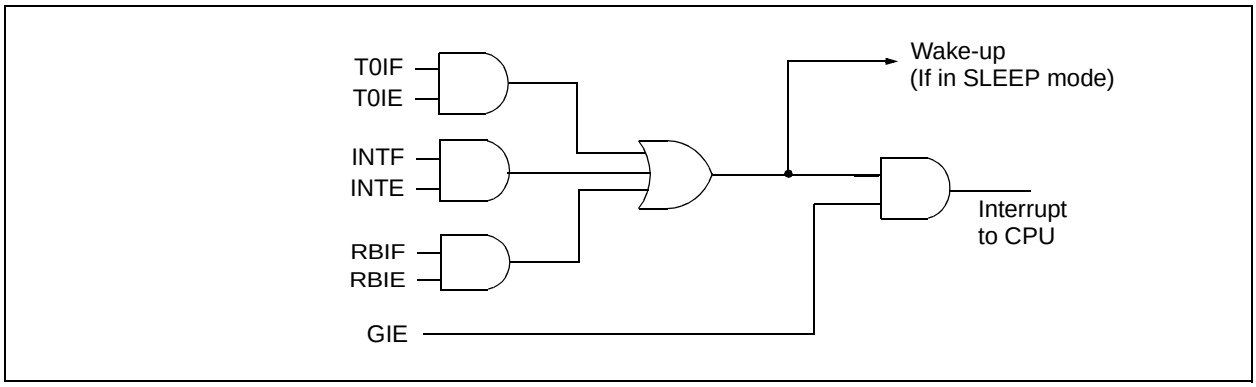
When an interrupt is responded to, the GIE is cleared to disable any further interrupt, the return address is pushed into the stack and the PC is loaded with 0004h. Once in the interrupt service routine, the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt flag bit(s) must be cleared in software before re-enabling interrupts to avoid RB0/INT recursive interrupts.

For external interrupt events, such as the INT pin or PORTB change interrupt, the interrupt latency will be three or four instruction cycles. The exact latency depends when the interrupt event occurs (Figure 7-12). The latency is the same for one or two cycle instructions. Once in the interrupt service routine, the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt flag bit(s) must be cleared in software before re-enabling interrupts to avoid multiple interrupt requests.

Note 1: Individual interrupt flag bits are set regardless of the status of their corresponding mask bit or the GIE bit.

Note 2: When an instruction that clears the GIE bit is executed, any interrupts that were pending for execution in the next cycle are ignored. The CPU will execute a NOP in the cycle immediately following the instruction which clears the GIE bit. The interrupts which were ignored are still pending to be serviced when the GIE bit is set again.

FIGURE 7-11: INTERRUPT LOGIC



7.5.1 RB0/INT INTERRUPT

External interrupt on RB0/INT pin is edge triggered, either rising if INTEDG bit (OPTION<6>) is set, or falling, if INTEDG bit is clear. When a valid edge appears on the RB0/INT pin, the INTF bit (INTCON<1>) is set. This interrupt can be disabled by clearing the INTE control bit (INTCON<4>). The INTF bit must be cleared in software in the interrupt service routine before re-enabling this interrupt. The RB0/INT interrupt can wake-up the processor from SLEEP, if the INTE bit was set prior to going into SLEEP. The status of the GIE bit decides whether or not the processor branches to the interrupt vector following wake-up. See Section 7.8 for details on SLEEP and Figure 7-14 for timing of wake-up from SLEEP through RB0/INT interrupt.

7.5.2 TMR0 INTERRUPT

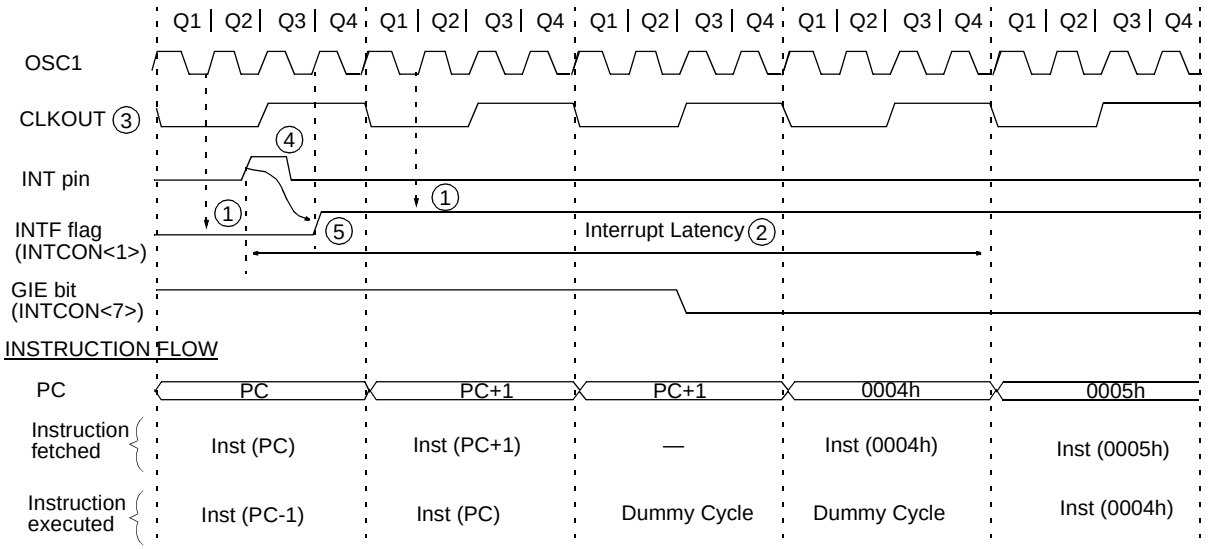
An overflow (FFh $\rightarrow$ 00h) in the TMR0 register will set the T0IF (INTCON<2>) bit. The interrupt can be enabled/disabled by setting/clearing T0IE (INTCON<5>) bit. For operation of the Timer0 module, see Section 6.0.

7.5.3 PORTB INTERRUPT

An input change on PORTB <7:4> sets the RBIF (INTCON<0>) bit. The interrupt can be enabled/disabled by setting/clearing the RBIE (INTCON<4>) bit. For operation of PORTB (Section 5.2).

Note:	If a change on the I/O pin should occur when the read operation is being executed (start of the Q2 cycle), then the RBIF interrupt flag may not get set.

FIGURE 7-12: INT PIN INTERRUPT TIMING



Note 1: INTF flag is sampled here (every Q1).
2: Asynchronous interrupt latency = 3-4 TCY. Synchronous latency = 3 TCY, where TCY = instruction cycle time. Latency is the same whether Inst (PC) is a single cycle or a 2-cycle instruction.
3: CLKOUT is available only in RC oscillator mode.
4: For minimum width of INT pulse, refer to AC specs.
5: INTF is enabled to be set anytime during the Q4-Q1 cycles.

TABLE 7-8: SUMMARY OF INTERRUPT REGISTERS

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR Reset	Value on all other resets(1)
0Bh	INTCON	GIE	-	T0IE	INTE	RBIE	T0IF	INTF	RBIF	- 000x	0000 000u

Note 1: Other (non power-up) resets include $\overline{\text{MCLR}}$ reset, Brown-out Reset and Watchdog Timer Reset during normal operation.

7.6 Context Saving During Interrupts

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

Example 7-1 stores and restores the STATUS and W registers. The user register, W_TEMP, must be defined in both banks and must be defined at the same offset from the bank base address (i.e., W_TEMP is defined at 0x20 in Bank 0 and it must also be defined at 0xA0 in Bank 1). The user register, STATUS_TEMP, must be defined in Bank 0. The Example 7-1:

- Stores the W register
- Stores the STATUS register in Bank 0
- Executes the ISR code
- Restores the STATUS (and bank select bit register)
- Restores the W register

EXAMPLE 7-1: SAVING THE STATUS AND W REGISTERS IN RAM

```
MOVWF   W_TEMP          ;copy W to temp register,
                        ;could be in either bank
SWAPF   STATUS,W        ;swap status to be saved into W
BCF     STATUS,RP0      ;change to bank 0 regardless
                        ;of current bank
MOVWF   STATUS_TEMP     ;save status to bank 0
                        ;register
    :
    :   (ISR)
    :
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
```

7.7 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 CLKIN pin. That means that the WDT will run, even if the clock on the OSC1 and OSC2 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 time-out causes the device to wake-up and continue with normal operation. The WDT can be permanently disabled by programming the configuration bit WDTE as clear (Section 7.1).

7.7.1 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 register. Thus, time-out periods up to 2.3 seconds can be realized.

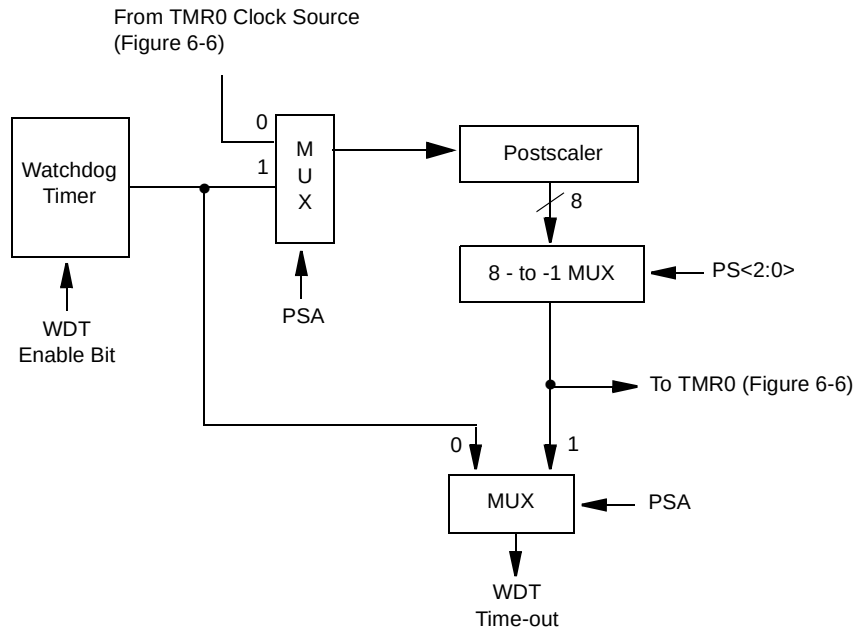
The CLRWDT and SLEEP instructions clear the WDT and the postscaler, if assigned to the WDT, and prevent it from timing out and generating a device RESET.

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

7.7.2 WDT PROGRAMMING CONSIDERATIONS

It should also be taken in 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 7-13: WATCHDOG TIMER BLOCK DIAGRAM



Note: T0SE, T0CS, PSA, PS<2:0> are bits in the OPTION register.

TABLE 7-9: SUMMARY OF WATCHDOG TIMER REGISTERS

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR Reset	Value on all other Resets
2007h	Config. bits	—	—	CP1	CP0	$\overline{\text{PWRTE}}$	WDTE	FOSC1	FOSC0	—	—
81h	OPTION	$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111

Legend: Shaded cells are not used by the Watchdog Timer.

Note: – = Unimplemented location, read as “0”
+ = Reserved for future use

7.8 Power-Down Mode (SLEEP)

The Power-down mode is entered by executing a SLEEP instruction.

If enabled, the Watchdog Timer will be cleared but keeps running, the $\overline{PD}$ bit in the STATUS register is cleared, the $\overline{TO}$ bit is set, and the oscillator driver is turned off. The I/O ports maintain the status they had, before SLEEP was executed (driving high, low, or hi-impedance).

For lowest current consumption in this mode, all I/O pins should be either at V_{DD} or V_{SS} with no external circuitry drawing current from the I/O pin and the comparators and V_{REF} should be disabled. I/O pins that are hi-impedance inputs should be pulled high or low externally to avoid switching currents caused by floating inputs. The T0CKI input should also be at V_{DD} or V_{SS} for lowest current consumption. The contribution from on chip pull-ups on PORTB should be considered.

The $\overline{MCLR}$ pin must be at a logic high level (V_{IHMC}).

Note:	It should be noted that a RESET generated by a WDT time-out does not drive $\overline{MCLR}$ pin low.

7.8.1 WAKE-UP FROM SLEEP

The device can wake-up from SLEEP through one of the following events:

1. External reset input on $\overline{MCLR}$ pin
2. Watchdog Timer Wake-up (if WDT was enabled)
3. Interrupt from RB0/INT pin or RB Port change.

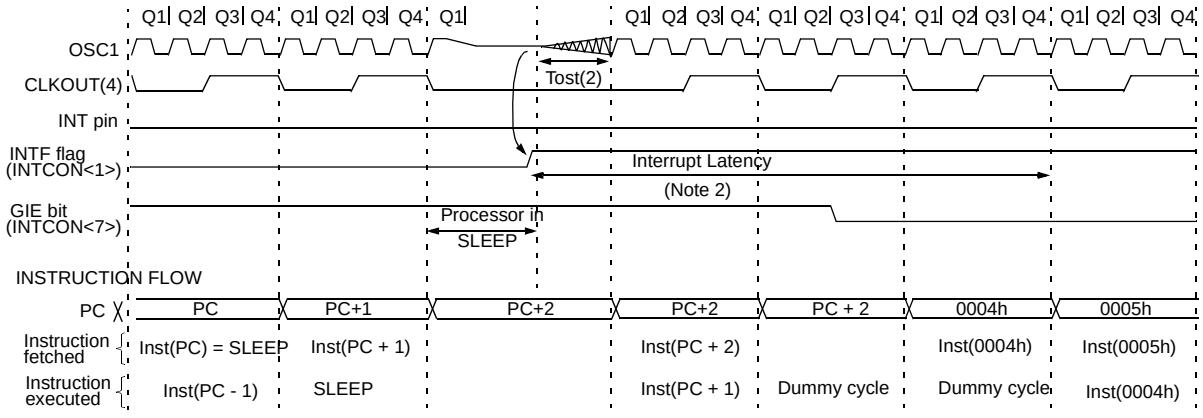
The first event will cause a device reset. The two latter events are considered a continuation of program execution. The $\overline{TO}$ and $\overline{PD}$ bits in the STATUS register can be used to determine the cause of device reset. $\overline{PD}$ bit, which is set on power-up, is cleared when SLEEP is invoked. $\overline{TO}$ bit is cleared if WDT wake-up occurred.

When the SLEEP instruction is being executed, the next instruction (PC + 1) is pre-fetched. For the device to wake-up through an interrupt event, the corresponding interrupt enable bit must be set (enabled). Wake-up is regardless of the state of the GIE bit. If the GIE bit is clear (disabled), the device continues execution at the instruction after the SLEEP instruction. If the GIE bit is set (enabled), the device executes the instruction after the SLEEP instruction and then branches to the interrupt address (0004h). In cases where the execution of the instruction following SLEEP is not desirable, the user should have an NOP after the SLEEP instruction.

Note:	If the global interrupts are disabled (GIE is cleared), but any interrupt source has both its interrupt enable bit and the corresponding interrupt flag bits set, the device will immediately wake-up from sleep. The sleep instruction is completely executed.

The WDT is cleared when the device wakes-up from sleep, regardless of the source of wake-up.

FIGURE 7-14: WAKE-UP FROM SLEEP THROUGH INTERRUPT



Note 1: XT, HS or LP oscillator mode assumed.
Note 2: $T_{OST} = 1024T_{OSC}$ (drawing not to scale) This delay will not be there for RC osc mode.
Note 3: GIE = '1' assumed. In this case, after wake- up, the processor jumps to the interrupt routine. If GIE = '0', execution will continue in-line.
Note 4: CLKOUT is not available in these osc modes, but shown here for timing reference.

7.9 Code Protection

If the code protection bit(s) have not been programmed, the on-chip program memory can be read out for verification purposes.

7.10 ID Locations

Four memory locations (2000h-2003h) are designated as ID locations where the user can store checksum or other code-identification numbers. These locations are not accessible during normal execution, but are readable and writable during program/verify. Only the least significant 4 bits of the ID locations are used.

7.11 In-Circuit Serial Programming

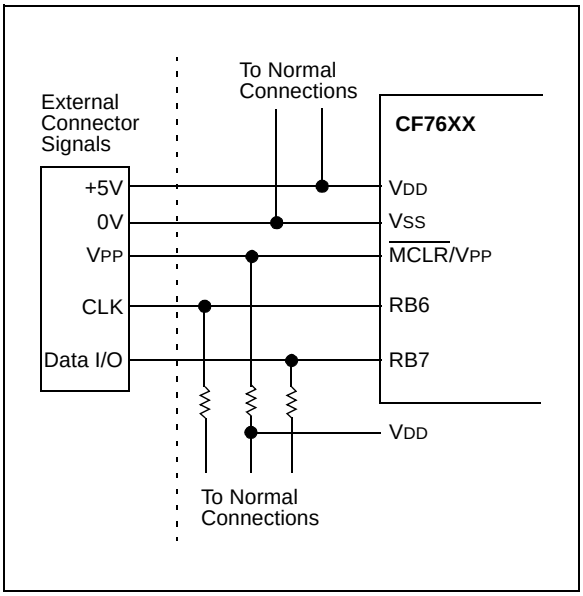
The CF76XX microcontrollers can be serially programmed while in the end application circuit. This is simply done with two lines for clock and data and three other lines for power, ground, and the programming voltage. This allows customers to manufacture boards with unprogrammed devices, and then program the microcontroller just before shipping the product. This also allows the most recent firmware or a custom firmware to be programmed.

The device is placed into a program/verify mode by holding the RB6 and RB7 pins low, while raising the $\overline{MCLR}$ (V_{PP}) pin from V_{IL} to V_{IHH} (see programming specification). RB6 becomes the programming clock and RB7 becomes the programming data. Both RB6 and RB7 are Schmitt Trigger inputs in this mode.

After reset, to place the device into programming/verify mode, the program counter (PC) is at location 00h. A 6-bit command is then supplied to the device. Depending on the command, 14-bits of program data are then supplied to or from the device, depending if the command was a load or a read. For complete details of serial programming, please refer to the CF76XX Programming Specification (#DS30228).

A typical in-circuit serial programming connection is shown in Figure 7-15.

FIGURE 7-15: TYPICAL IN-CIRCUIT SERIAL PROGRAMMING CONNECTION



8.0 INSTRUCTION SET SUMMARY

Each CF76XX instruction is a 14-bit word divided into an OPCODE which specifies the instruction type and one or more operands which further specify the operation of the instruction. The CF76XX instruction set summary in Table 8-2 lists **byte-oriented**, **bit-oriented**, and **literal and control** operations. Table 8-1 shows the opcode field descriptions.

For **byte-oriented** instructions, 'f' represents a file register designator and 'd' represents a destination designator. The file register designator specifies which file register is to be used by the instruction.

The destination designator specifies where the result of the operation is to be placed. If 'd' is zero, the result is placed in the W register. If 'd' is one, the result is placed in the file register specified in the instruction.

For **bit-oriented** instructions, 'b' represents a bit field designator which selects the number of the bit affected by the operation, while 'f' represents the number of the file in which the bit is located.

For **literal and control** operations, 'k' represents an eight or eleven bit constant or literal value.

TABLE 8-1: OPCODE FIELD 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) The assembler will generate code with x = 0. It is the recommended form of use for compatibility with all Microchip software tools.
d	Destination select; 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
WDT	Watchdog Timer/Counter
$\overline{TO}$	Time-out bit
$\overline{PD}$	Power-down bit
dest	Destination either the W register or the specified register file location
[ ]	Options
( )	Contents
$\rightarrow$	Assigned to
< >	Register bit field
$\in$	In the set of
italics	User defined term (font is courier)

The instruction set is highly orthogonal and is grouped into three basic categories:

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

All instructions are executed within one single instruction cycle, unless a conditional test is true or the program counter is changed as a result of an instruction. In this case, the execution takes two instruction cycles with the second cycle executed as a NOP. One instruction cycle consists of four oscillator periods. Thus, for an oscillator frequency of 4 MHz, the normal instruction execution time is 1 µs. If a conditional test is true or the program counter is changed as a result of an instruction, the instruction execution time is 2 µs.

Table 8-1 lists the instructions recognized by the MPASM assembler.

Figure 8-1 shows the three general formats that the instructions can have.

Note: To maintain upward compatibility with future PICmicro® products, do not use the OPTION and TRIS instructions.

All examples use the following format to represent a hexadecimal number:

0xhh

where h signifies a hexadecimal digit.

FIGURE 8-1: GENERAL FORMAT FOR INSTRUCTIONS

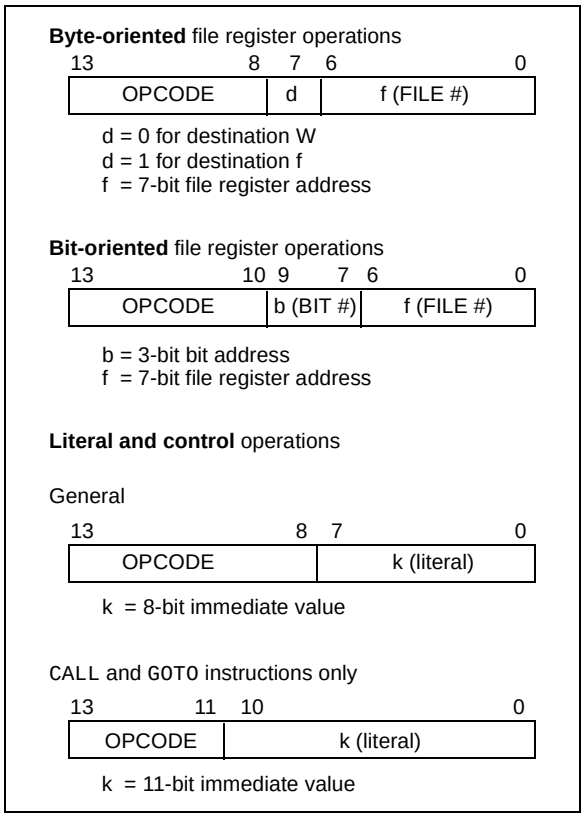


TABLE 8-2: CF76XX INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	14-Bit Opcode MSb LSb	Status Affected	Notes
BYTE-ORIENTED FILE REGISTER OPERATIONS						
ADDWF	**f, d**	Add W and f	1	00 0111 dfff ffff	C,DC,Z	1,2
ANDWF	**f, d**	AND W with f	1	00 0101 dfff ffff	Z	1,2
CLRF	**f**	Clear f	1	00 0001 lfff ffff	Z	2
CLRW	**-**	Clear W	1	00 0001 0000 0011	Z	
COMF	**f, d**	Complement f	1	00 1001 dfff ffff	Z	1,2
DECF	**f, d**	Decrement f	1	00 0011 dfff ffff	Z	1,2
DECFSZ	**f, d**	Decrement f, Skip if 0	1(2)	00 1011 dfff ffff		1,2,3
INCF	**f, d**	Increment f	1	00 1010 dfff ffff	Z	1,2
INCFSZ	**f, d**	Increment f, Skip if 0	1(2)	00 1111 dfff ffff		1,2,3
IORWF	**f, d**	Inclusive OR W with f	1	00 0100 dfff ffff	Z	1,2
MOVF	**f, d**	Move f	1	00 1000 dfff ffff	Z	1,2
MOVWF	**f**	Move W to f	1	00 0000 lfff ffff		
NOP	**-**	No Operation	1	00 0000 0xx0 0000		
RLF	**f, d**	Rotate Left f through Carry	1	00 1101 dfff ffff	C	1,2
RRF	**f, d**	Rotate Right f through Carry	1	00 1100 dfff ffff	C	1,2
SUBWF	**f, d**	Subtract W from f	1	00 0010 dfff ffff	C,DC,Z	1,2
SWAPF	**f, d**	Swap nibbles in f	1	00 1110 dfff ffff		1,2
XORWF	**f, d**	Exclusive OR W with f	1	00 0110 dfff ffff	Z	1,2
BIT-ORIENTED FILE REGISTER OPERATIONS						
BCF	**f, b**	Bit Clear f	1	01 00bb bfff ffff		1,2
BSF	**f, b**	Bit Set f	1	01 01bb bfff ffff		1,2
BTFSC	**f, b**	Bit Test f, Skip if Clear	1 (2)	01 10bb bfff ffff		3
BTFSS	**f, b**	Bit Test f, Skip if Set	1 (2)	01 11bb bfff ffff		3
LITERAL AND CONTROL OPERATIONS						
ADDLW	**k**	Add literal and W	1	11 111x kkkk kkkk	C,DC,Z	
ANDLW	**k**	AND literal with W	1	11 1001 kkkk kkkk	Z	
CALL	**k**	Call subroutine	2	10 0kkk kkkk kkkk		
CLRWDT	**-**	Clear Watchdog Timer	1	00 0000 0110 0100	$\overline{TO}$,$\overline{PD}$	
GOTO	**k**	Go to address	2	10 1kkk kkkk kkkk		
IORLW	**k**	Inclusive OR literal with W	1	11 1000 kkkk kkkk	Z	
MOVLW	**k**	Move literal to W	1	11 00xx kkkk kkkk		
RETFIE	**-**	Return from interrupt	2	00 0000 0000 1001		
RETLW	**k**	Return with literal in W	2	11 01xx kkkk kkkk		
RETURN	**-**	Return from Subroutine	2	00 0000 0000 1000		
SLEEP	**-**	Go into standby mode	1	00 0000 0110 0011	$\overline{TO}$,$\overline{PD}$	
SUBLW	**k**	Subtract W from literal	1	11 110x kkkk kkkk	C,DC,Z	
XORLW	**k**	Exclusive OR literal with W	1	11 1010 kkkk kkkk	Z	

Note 1: When an I/O register is modified as a function of itself (e.g., `MOVF PORTB, 1`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

Note 2: If this instruction is executed on the TMR0 register (and, where applicable, d = 1), the prescaler will be cleared if assigned to the Timer0 Module.

Note 3: If Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

8.1 Instruction Descriptions

ADDLW **Add Literal and W**

Syntax: [*label*] ADDLW k

Operands: $0 \le k \le 255$

Operation: (W) + k → (W)

Status Affected: C, DC, Z

Encoding:

11	111x	kkkk	kkkk

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

Words: 1

Cycles: 1

Example
```
ADDLW   0x15
```
Before Instruction
W = 0x10
After Instruction
W = 0x25

ANDLW **AND Literal with W**

Syntax: [*label*] ANDLW k

Operands: $0 \le k \le 255$

Operation: (W) .AND. (k) → (W)

Status Affected: Z

Encoding:

11	1001	kkkk	kkkk

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

Words: 1

Cycles: 1

Example
```
ANDLW   0x5F
```
Before Instruction
W = 0xA3
After Instruction
W = 0x03

ADDWF **Add W and f**

Syntax: [*label*] ADDWF f,d

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

Operation: (W) + (f) → (dest)

Status Affected: C, DC, Z

Encoding:

00	0111	dfff	ffff

Description: Add the contents of the W register with register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.

Words: 1

Cycles: 1

Example
```
ADDWF   FSR, 0
```
Before Instruction
W = 0x17
FSR = 0xC2
After Instruction
W = 0xD9
FSR = 0xC2

ANDWF **AND W with f**

Syntax: [*label*] ANDWF f,d

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

Operation: (W) .AND. (f) → (dest)

Status Affected: Z

Encoding:

00	0101	dfff	ffff

Description: AND the W register with register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.

Words: 1

Cycles: 1

Example
```
ANDWF   FSR, 1
```
Before Instruction
W = 0x17
FSR = 0xC2
After Instruction
W = 0x17
FSR = 0x02

BCF **Bit Clear f**

Syntax: [*label*] BCF f,b

Operands: 0 ≤ f ≤ 127
0 ≤ b ≤ 7

Operation: 0 → (f<b>)

Status Affected: None

Encoding:

01	00bb	bfff	ffff

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

Words: 1

Cycles: 1

Example

```
BCF     FLAG_REG, 7
```

Before Instruction
FLAG_REG = 0xC7
After Instruction
FLAG_REG = 0x47

BSF **Bit Set f**

Syntax: [*label*] BSF f,b

Operands: 0 ≤ f ≤ 127
0 ≤ b ≤ 7

Operation: 1 → (f<b>)

Status Affected: None

Encoding:

01	01bb	bfff	ffff

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

Words: 1

Cycles: 1

Example

```
BSF     FLAG_REG,   7
```

Before Instruction
FLAG_REG = 0x0A
After Instruction
FLAG_REG = 0x8A

BTFSC **Bit Test, Skip if Clear**

Syntax: [*label*] BTFSC f,b

Operands: 0 ≤ f ≤ 127
0 ≤ b ≤ 7

Operation: skip if (f<b>) = 0

Status Affected: None

Encoding:

01	10bb	bfff	ffff

Description: If bit 'b' in register 'f' is '0', 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.

Words: 1

Cycles: 1(2)

Example

```
HERE    BTFSC   FLAG,1
FALSE   GOTO    PROCESS_CO
TRUE    •       DE
        •
        •
```

Before Instruction
PC = address HERE
After Instruction
if FLAG<1> = 0,
PC = address TRUE
if FLAG<1>=1,
PC = address FALSE

BTFSS **Bit Test f, Skip if Set**

Syntax: [*label*] BTFSS f,b

Operands: 0 ≤ f ≤ 127
0 ≤ b < 7

Operation: skip if (f<b>) = 1

Status Affected: None

Encoding:

01	11bb	bfff	ffff

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

Words: 1

Cycles: 1(2)

Example

```
HERE    BTFSS   FLAG,1
FALSE   GOTO    PROCESS_CO
TRUE    •       DE
        •
        •
```

Before Instruction
PC = address HERE
After Instruction
if FLAG<1> = 0,
PC = address FALSE
if FLAG<1> = 1,
PC = address TRUE

CALL **Call Subroutine**

Syntax: [*label*] CALL k

Operands: 0 ≤ k ≤ 2047

Operation: (PC)+ 1→ TOS,
k → PC<10:0>,
(PCLATH<4:3>) → PC<12:11>

Status Affected: None

Encoding:

10	0kkk	kkkk	kkkk

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

Words: 1

Cycles: 2

Example

```
HERE    CALL
                THER
                E
```

Before Instruction
PC = Address HERE
After Instruction
PC = Address THERE
TOS = Address HERE+1

CLRF **Clear f**

Syntax: [*label*] CLRF f

Operands: 0 ≤ f ≤ 127

Operation: 00h → (f)
1 → Z

Status Affected: Z

Encoding:

00	0001	1fff	ffff

Description: The contents of register 'f' are cleared and the Z bit is set.

Words: 1

Cycles: 1

Example

```
CLRF    FLAG_REG
```

Before Instruction
FLAG_REG = 0x5A
After Instruction
FLAG_REG = 0x00
Z = 1

CLRW **Clear W**

Syntax: [*label*] CLRW

Operands: None

Operation: 00h → (W)
1 → Z

Status Affected: Z

Encoding:

00	0001	0000	0011

Description: W register is cleared. Zero bit (Z) is set.

Words: 1

Cycles: 1

Example
```
CLRW
```
Before Instruction
W = 0x5A
After Instruction
W = 0x00
Z = 1

CLRWDT **Clear Watchdog Timer**

Syntax: [*label*] CLRWDT

Operands: None

Operation: 00h → WDT
0 → WDT prescaler,
1 → $\overline{TO}$
1 → $\overline{PD}$

Status Affected: $\overline{TO}$, $\overline{PD}$

Encoding:

00	0000	0110	0100

Description: CLRWDT instruction resets the Watchdog Timer. It also resets the prescaler of the WDT. Status bits $\overline{TO}$ and $\overline{PD}$ are set.

Words: 1

Cycles: 1

Example
```
CLRWDT
```
Before Instruction
WDT counter = ?
After Instruction
WDT counter = 0x00
WDT prescaler = 0
$\overline{TO}$ = 1
$\overline{PD}$ = 1

COMF **Complement f**

Syntax: [*label*] COMF f,d

Operands: 0 ≤ f ≤ 127
d ∈ [0,1]

Operation: ($\bar{f}$) → (dest)

Status Affected: Z

Encoding:

00	1001	dfff	ffff

Description: The contents of register 'f' are complemented. If 'd' is 0, the result is stored in W. If 'd' is 1, the result is stored back in register 'f'.

Words: 1

Cycles: 1

Example
```
COMF    REG1,0
```
Before Instruction
REG1 = 0x13
After Instruction
REG1 = 0x13
W = 0xEC

DECF **Decrement f**

Syntax: [*label*] DECF f,d

Operands: 0 ≤ f ≤ 127
d ∈ [0,1]

Operation: (f) - 1 → (dest)

Status Affected: Z

Encoding:

00	0011	dfff	ffff

Description: Decrement register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.

Words: 1

Cycles: 1

Example
```
DECF    CNT, 1
```
Before Instruction
CNT = 0x01
Z = 0
After Instruction
CNT = 0x00
Z = 1

DECFSZ **Decrement f, Skip if 0**

Syntax: [*label*] DECFSZ f,d

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

Operation: (f) - 1 → (dest); skip if result = 0

Status Affected: None

Encoding:

00	1011	dfff	ffff

Description: The contents of register 'f' are decremented. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'.
If the result is 0, the next instruction, which is already fetched, is discarded. A NOP is executed instead making it a two-cycle instruction.

Words: 1

Cycles: 1(2)

Example

```
HERE     DECFSZ   CNT, 1
         GOTO     LOOP
CONTINUE •
         •
         •
```

Before Instruction
PC = address HERE
After Instruction
CNT = CNT - 1
if CNT = 0,
PC = address CONTINUE
if CNT ≠ 0,
PC = address HERE+1

GOTO **Unconditional Branch**

Syntax: [*label*] GOTO k

Operands: $0 \le k \le 2047$

Operation: k → PC<10:0>
PCLATH<4:3> → PC<12:11>

Status Affected: None

Encoding:

10	1kkk	kkkk	kkkk

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.

Words: 1

Cycles: 2

Example

```
GOTO THERE
```

After Instruction
PC = Address THERE

INCF **Increment f**

Syntax: [*label*] INCF f,d

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

Operation: (f) + 1 → (dest)

Status Affected: Z

Encoding:

00	1010	dfff	ffff

Description: The contents of register 'f' are incremented. 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

Example

```
INCF    CNT, 1
```

Before Instruction
CNT = 0xFF
Z = 0
After Instruction
CNT = 0x00
Z = 1

INCFSZ **Increment f, Skip if 0**

Syntax: [*label*] INCFSZ f,d

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

Operation: (f) + 1 → (dest), skip if result = 0

Status Affected: None

Encoding:

00	1111	dfff	ffff

Description: The contents of register 'f' are incremented. If 'd' is 0 the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'.
If the result is 0, the next instruction, which is already fetched, is discarded. A NOP is executed instead making it a two-cycle instruction.

Words: 1

Cycles: 1(2)

Example

```
HERE     INCFSZ     CNT,  1
         GOTO       LOOP
CONTINUE •
         •
         •
```

Before Instruction
PC = address HERE
After Instruction
CNT = CNT + 1
if CNT= 0,
PC = address CONTINUE
if CNT≠ 0,
PC = address HERE +1

IORLW **Inclusive OR Literal with W**

Syntax: [*label*] IORLW k

Operands: $0 \le k \le 255$

Operation: (W) .OR. k → (W)

Status Affected: Z

Encoding:

11	1000	kkkk	kkkk

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

Words: 1

Cycles: 1

Example

```
IORLW   0x35
```

Before Instruction
W = 0x9A
After Instruction
W = 0xBF
Z = 1

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) → (dest)

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

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 ≤ k ≤ 255

Operation: k → (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

Example
```
MOVLW   0x5A
```
After Instruction
```
W  =   0x5A
```

MOVF **Move f**

Syntax: [*label*] MOVF f,d

Operands: 0 ≤ f ≤ 127
d ∈ [0,1]

Operation: (f) → (dest)

Status Affected: Z

Encoding:

00	1000	dfff	ffff

Description: The contents of register f is moved to a destination dependent 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

Example
```
MOVF    FSR, 0
```
After Instruction
```
W  = value in FSR register
Z  = 1
```

MOVWF **Move W to f**

Syntax: [*label*] MOVWF f

Operands: 0 ≤ f ≤ 127

Operation: (W) → (f)

Status Affected: None

Encoding:

00	0000	1fff	ffff

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

Words: 1

Cycles: 1

Example
```
MOVWF       OPTION
```
Before Instruction
```
OPTION  =   0xFF
W       =   0x4F
```
After Instruction
```
OPTION  =   0x4F
W       =   0x4F
```

NOP **No Operation**

Syntax: [*label*] NOP

Operands: None

Operation: No operation

Status Affected: None

Encoding:

00	0000	0xx0	0000

Description: No operation.

Words: 1

Cycles: 1

Example
```
NOP
```

OPTION	Load Option Register
Syntax:	[*label*] OPTION
Operands:	None
Operation:	(W) → OPTION
Status Affected:	None
Encoding:	00 | 0000 | 0110 | 0010
Description:	The contents of the W register are loaded in the OPTION register. This instruction is supported for code compatibility with PIC16C5X products. Since OPTION is a readable/writable register, the user can directly address it.
Words:	1
Cycles:	1
Example	**To maintain upward compatibility with future PICmicro® products, do not use this instruction.**

RETFIE	Return from Interrupt
Syntax:	[*label*] RETFIE
Operands:	None
Operation:	TOS → PC, 1 → GIE
Status Affected:	None
Encoding:	00 | 0000 | 0000 | 1001
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>). This is a two-cycle instruction.
Words:	1
Cycles:	2
Example	`RETFIE` After Interrupt PC = TOS GIE = 1

RETLW	Return with Literal in W
Syntax:	[*label*] RETLW k
Operands:	0 ≤ k ≤ 255
Operation:	k → (W); TOS → PC
Status Affected:	None
Encoding:	11 | 01xx | kkkk | kkkk
Description:	The W register is loaded with the eight bit literal 'k'. The program counter is loaded from the top of the stack (the return address). This is a two-cycle instruction.
Words:	1
Cycles:	2
Example	CALL TABLE;W contains table ;offset value • ;W now has table value TABLE • • ADDWF PC ;W = offset RETLW k1 ;Begin table RETLW k2 ; • • • RETLW kn ; End of table Before Instruction W = 0x07 After Instruction W = value of k8

RETURN	Return from Subroutine
Syntax:	[*label*] RETURN
Operands:	None
Operation:	TOS → PC
Status Affected:	None
Encoding:	00 | 0000 | 0000 | 1000
Description:	Return from subroutine. The stack is POPed and the top of the stack (TOS) is loaded into the program counter. This is a two cycle instruction.
Words:	1
Cycles:	2
Example	`RETURN` After Interrupt PC = TOS

RLF **Rotate Left f through Carry**

Syntax: [*label*] RLF f,d

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

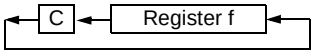
Operation: See description below

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

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

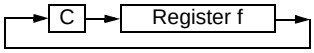
Operation: See description below

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

Example
```
RRF     REG1,
        0
```
Before Instruction
```
REG1 = 1110 0110
C    = 0
```
After Instruction
```
REG1 = 1110 0110
W    = 0111 0011
C    = 0
```

SLEEP

Syntax: [*label*] SLEEP

Operands: None

Operation: 00h → WDT,
0 → WDT prescaler,
1 → $\overline{TO}$,
0 → $\overline{PD}$

Status Affected: $\overline{TO}$, $\overline{PD}$

Encoding:

00	0000	0110	0011

Description: The power-down status bit, $\overline{PD}$ is cleared. Time-out status bit, $\overline{TO}$ is set. Watchdog Timer and its prescaler are cleared.
The processor is put into SLEEP mode with the oscillator stopped. See Section 7.8 for more details.

Words: 1

Cycles: 1

Example:
```
SLEE
P
```

SUBLW **Subtract W from Literal**

Syntax: [*label*] SUBLW k

Operands: $0 \le k \le 255$

Operation: k - (W) → (W)

Status Affected: C, DC, Z

Encoding:

11	110x	kkkk	kkkk

Description: The W register is subtracted (2's complement method) from the eight bit literal 'k'. The result is placed in the W register.

Words: 1

Cycles: 1

Example 1: SUBLW 0x02

Before Instruction

W = 1
C = ?

After Instruction

W = 1
C = 1; result is positive

Example 2: Before Instruction

W = 2
C = ?

After Instruction

W = 0
C = 1; result is zero

Example 3: Before Instruction

W = 3
C = ?

After Instruction

W = 0xFF
C = 0; result is negative

SUBWF **Subtract W from f**

Syntax: [*label*] SUBWF f,d

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

Operation: (f) - (W) → (dest)

Status Affected: C, DC, Z

Encoding:

00	0010	dfff	ffff

Description: Subtract (2's complement method) W register from register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.

Words: 1

Cycles: 1

Example 1: SUBW REG1,1
F

Before Instruction

REG1 = 3
W = 2
C = ?

After Instruction

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

Example 2: Before Instruction

REG1 = 2
W = 2
C = ?

After Instruction

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

Example 3: Before Instruction

REG1 = 1
W = 2
C = ?

After Instruction

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

SWAPF **Swap Nibbles in f**

Syntax: [*label*] SWAPF f,d

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

Operation: (f<3:0>) → (dest<7:4>),
(f<7:4>) → (dest<3:0>)

Status Affected: None

Encoding:

00	1110	dfff	ffff

Description: The upper and lower nibbles of register 'f' are exchanged. If 'd' is 0, the result is placed in W register. If 'd' is 1, the result is placed in register 'f'.

Words: 1

Cycles: 1

Example

```
SWAPF REG, 0
```

Before Instruction

REG1 = 0xA5

After Instruction

REG1 = 0xA5
W = 0x5A

TRIS **Load TRIS Register**

Syntax: [*label*] TRIS f

Operands: $5 \le f \le 7$

Operation: (W) → TRIS register f;

Status Affected: None

Encoding:

00	0000	0110	0fff

Description: The instruction is supported for code compatibility with the PIC16C5X products. Since TRIS registers are readable and writable, the user can directly address them.

Words: 1

Cycles: 1

Example

To maintain upward compatibility with future PICmicro® products, do not use this instruction.

XORLW **Exclusive OR Literal with W**

Syntax: [*label*] XORLW k

Operands: $0 \le k \le 255$

Operation: (W) .XOR. k → (W)

Status Affected: Z

Encoding:

11	1010	kkkk	kkkk

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

Words: 1

Cycles: 1

Example

```
XORL 0xAF
W
```

Before Instruction

W = 0xB5

After Instruction

W = 0x1A

XORWF **Exclusive OR W with f**

Syntax: [*label*] XORWF f,d

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

Operation: (W) .XOR. (f) → (dest)

Status Affected: Z

Encoding:

00	0110	dfff	ffff

Description: Exclusive OR the contents of the W register with register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.

Words: 1

Cycles: 1

Example

```
XORW REG 1
F
```

Before Instruction

REG = 0xAF
W = 0xB5

After Instruction

REG = 0x1A
W = 0xB5

NOTES:

9.0 ELECTRICAL SPECIFICATIONS

Absolute Maximum Ratings †

Ambient Temperature under bias ..... 0° to +70°C

Storage Temperature ..... -65° to +150°C

Voltage on any pin with respect to V_{SS} (except V_{DD} and $\overline{MCLR}$) ..... -0.6V to V_{DD} +0.6V

Voltage on V_{DD} with respect to V_{SS} ..... 0 to +7.5V

Voltage on $\overline{MCLR}$ with respect to V_{SS} (Note 2) ..... 0 to +14V

Voltage on RA4 with respect to V_{SS} 8.5V

Total power Dissipation (Note 1) 1.0W

Maximum Current out of V_{SS} pin ..... 300 mA

Maximum Current into V_{DD} pin ..... 250 mA

Input Clamp Current, I_{IK} (V_I <0 or V_I> V_{DD}) ..... ±20 mA

Output Clamp Current, I_{OK} (V_O <0 or V_O>V_{DD}) ..... ±20 mA

Maximum Output Current sunk by any I/O pin ..... 25 mA

Maximum Output Current sourced by any I/O pin ..... 25 mA

Maximum Current sunk by PORTA and PORTB ..... 200 mA

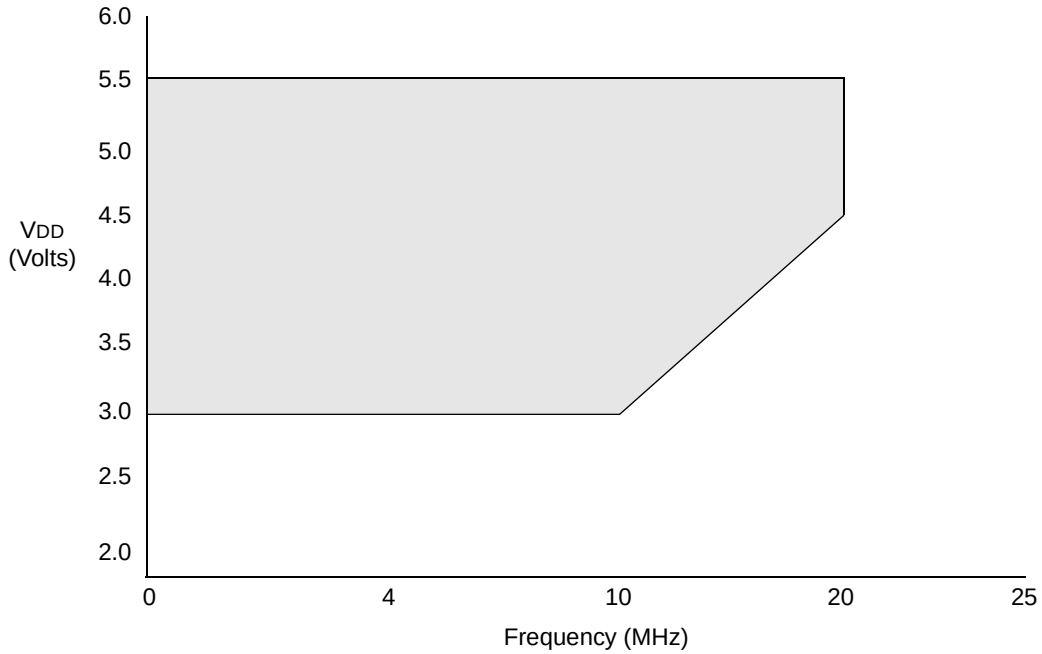
Maximum Current sourced by PORTA and PORTB ..... 200 mA

Note 1: Power dissipation is calculated as follows: $P_{DIS} = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD}-V_{OH}) \times I_{OH}\} + \sum(Vol \times I_{OL})$

2: Voltage spikes below V_{SS} at the $\overline{MCLR}$ pin, inducing currents greater than 80 mA, may cause latch-up. Thus, a series resistor of 50-100Ω should be used when applying a "low" level to the $\overline{MCLR}$ pin rather than pulling this pin directly to V_{SS}.

† **NOTICE**: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

FIGURE 9-1: CF76XX VOLTAGE-FREQUENCY GRAPH, 0°C ≤ TA ≤ +70°C



Note 1: The shaded region indicates the permissible combinations of voltage and frequency.

2: The maximum rated speed of the part limits the permissible combinations of voltage and frequency. Please reference the Product Identification System section for the maximum rated speed of the parts.

9.1 DC CHARACTERISTICS: CF76XX (Commercial)

DC CHARACTERISTICS			Standard Operating Conditions (unless otherwise stated) Operating temperature 0°C ≤ TA ≤ +70°C for commercial				
Param No.	**Sym**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
D001	VDD	Supply Voltage	3.0	–	5.5	V	See Figures 12-1 through 12-5
D002	VDR	RAM Data Retention Voltage (Note 1)	–	1.5*	–	V	Device in SLEEP mode
D003	VPOR	VDD start voltage to ensure Power-on Reset	–	Vss	–	V	See section on power-on reset for details
D004	SVDD	VDD rise rate to ensure Power-on Reset	0.05*	–	–	V/ms	See section on power-on reset for details
D010	IDD	Supply Current (Note 2)	–	1.8	3.3	mA	FOSC = 4 MHz, VDD = 5.5V, WDT disabled, XT mode, (Note 4)*
			–	35	70	µA	FOSC = 32 kHz, VDD = 4.0V, WDT disabled, LP mode
			–	9.0	20	mA	FOSC = 20 MHz, VDD = 5.5V, WDT disabled, HS mode
D020	IPD	Power Down Current (Note 3)	–	1.0	2.5	µA	VDD=4.0V, WDT disabled

* These parameters are characterized but not tested.

Note 1: Data in "Typ" column is at 5.0V, 25°C, unless otherwise stated. These parameters are for design guidance only and are not tested.

2: This is the limit to which VDD can be lowered in SLEEP mode without losing RAM data.

3: The supply current is mainly a function of the operating voltage and frequency. Other factors such as I/O pin loading and switching rate, oscillator type, internal code execution pattern, and temperature also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail to rail; all I/O pins tri-stated, pulled to VDD,
MCLR = VDD; WDT enabled/disabled as specified.

4: The power down current in SLEEP mode does not depend on the oscillator type. Power down current is measured with the part in SLEEP mode, with all I/O pins in hi-impedance state and tied to VDD or VSS.

5: For RC osc configuration, current through Rext is not included. The current through the resistor can be estimated by the formula: Ir = VDD/2Rext (mA) with Rext in kΩ.

6: The Δ current is the additional current consumed when this peripheral is enabled. This current should be added to the base IDD or IPD measurement.

9.2 DC CHARACTERISTICS: CF76XX (Commercial)

DC CHARACTERISTICS			Standard Operating Conditions (unless otherwise stated) Operating temperature 0°C ≤ TA ≤ +70°C for commercial Operating voltage VDD range as described in table below.				
Param. No.	**Sym**	**Characteristic**	**Min**	**Typ†**	**Max**	**Unit**	**Conditions**
	VIL	**Input Low Voltage**					
		I/O ports					
D030		with TTL buffer	VSS	-	0.8V 0.15VDD	V	VDD = 4.5V to 5.5V otherwise
D031		with Schmitt Trigger input	VSS		0.2VDD	V	
D032		MCLR, RA4/T0CKI,OSC1 (in RC mode)	Vss	-	0.2VDD	V	Note2
D033		OSC1 (in XT and HS)	Vss	-	0.3VDD	V	
		OSC1 (in LP)	Vss	-	0.6VDD-1.0	V	
	VIH	**Input High Voltage**					
		I/O ports		-			
D040		with TTL buffer	2.0V .25VDD + 0.8V	-	VDD VDD	V	VDD = 4.5V to 5.5V otherwise
D041		with Schmitt Trigger input	0.8VDD		VDD		
D042		MCLR RA4/T0CKI	0.8VDD	-	VDD	V	
D043		OSC1 (XT, HS and LP)	0.7VDD	-	VDD	V	
D043A		OSC1 (in RC mode)	0.9VDD				Note2
D070	IPURB	PORTB weak pull-up current	50	200	400	µA	VDD = 5.0V, VPIN = VSS
		Input Leakage Current (2, 3)					
	IIL	I/O ports (Except PORTA)			±1.0	µA	VSS ≤ VPIN ≤ VDD, pin at hi-impedance
D060		PORTA	-	-	±0.5	µA	Vss ≤ VPIN ≤ VDD, pin at hi-impedance
D061		RA4/T0CKI	-	-	±1.0	µA	Vss ≤ VPIN ≤ VDD
D063		OSC1, MCLR	-	-	±5.0	µA	Vss ≤ VPIN ≤ VDD, XT, HS and LP osc configuration
	VOL	**Output Low Voltage**					
D080		I/O ports	-	-	0.6	V	IOL=8.5 mA, VDD=4.5V, -0° to +70°C
D083		OSC2/CLKOUT (RC only)	-	-	0.6	V	IOL=1.6 mA, VDD=4.5V, -0° to +70°C
	VOH	**Output High Voltage (3)**					
D090		I/O ports (Except RA4)	VDD-0.7 VDD-0.7	-	-	V V	IOH=-3.0 mA, VDD=4.5V, -0° to +70°C
D092		OSC2/CLKOUT (RC only)	VDD-0.7 VDD-0.7	-	-	V V	IOH=-1.3 mA, VDD=4.5V, -0° to +70°C
*D150	VOD	**Open-Drain High Voltage**			8.5*		RA4 pin CF76XX
		Capacitive Loading Specs on Output Pins					
D100	COSC2	OSC2 pin			15	pF	In XT, HS and LP modes when external clock used to drive OSC1.
D101	CIO	All I/O pins/OSC2 (in RC mode)			50	pF	

* These parameters are characterized but not tested.

Note 1: Data in "Typ" column is at 5.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

2: In RC oscillator configuration, the OSC1 pin is a Schmitt Trigger input. It is not recommended that the CF76XX be driven with external clock in RC mode.

3: The leakage current on the MCLR pin is strongly dependent on applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

4: Negative current is defined as coming out of the pin.

9.3 Timing Parameter Symbology

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

1. TppS2ppS
2. TppS

T			
F	Frequency	T	Time

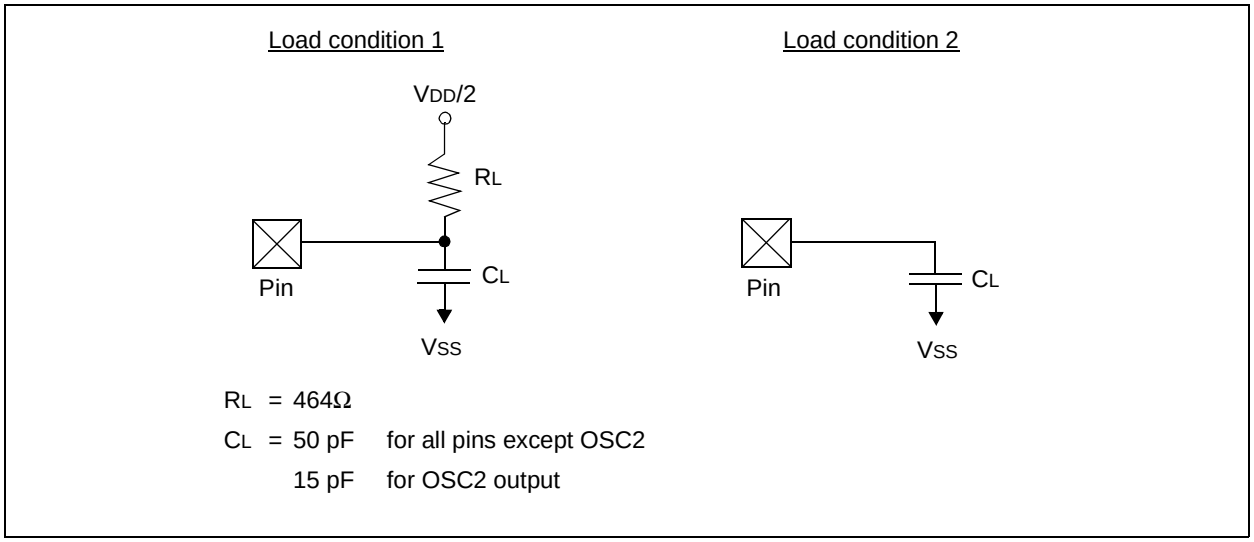
Lowercase subscripts (pp) and their meanings:

pp			
ck	CLKOUT	osc	OSC1
io	I/O port	t0	T0CKI
mc	$\overline{\text{MCLR}}$		

Uppercase letters and their meanings:

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

FIGURE 9-2: LOAD CONDITIONS



R_L = 464Ω

C_L = 50 pF for all pins except OSC2

15 pF for OSC2 output

9.4 Timing Diagrams and Specifications

FIGURE 9-3: EXTERNAL CLOCK TIMING

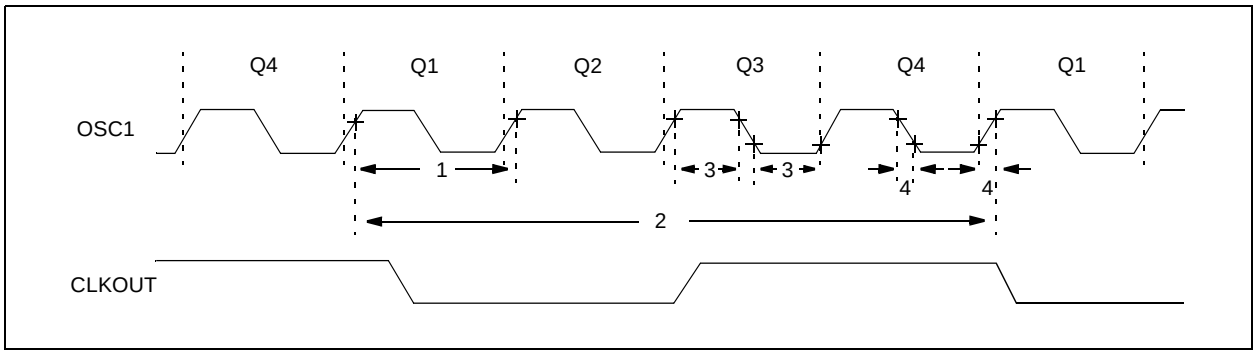


TABLE 9-1: EXTERNAL CLOCK TIMING REQUIREMENTS

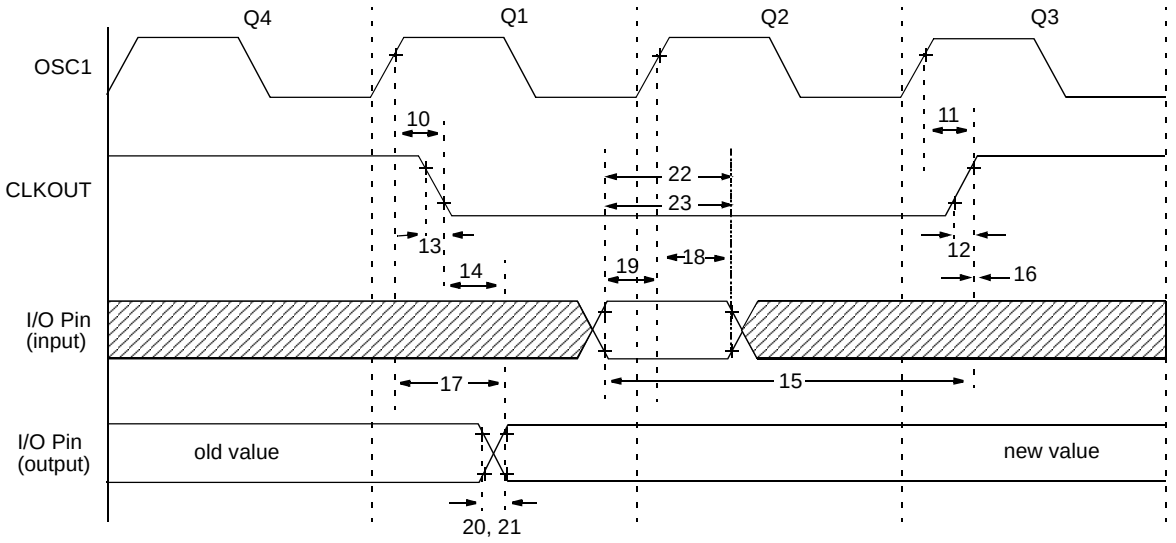
Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
1A	FOSC	External CLKIN Frequency (Note 1)	DC	—	4	MHz	XT and RC osc mode, VDD=5.0V
			DC	—	20	MHz	HS osc mode
			DC	—	200	kHz	LP osc mode
		Oscillator Frequency (Note 1)	DC	—	4	MHz	RC osc mode, VDD=5.0V
			0.1	—	4	MHz	XT osc mode
			1	—	20	MHz	HS osc mode
			DC	–	200	kHz	LP osc mode
1	TOSC	External CLKIN Period (Note 1)	250	—	—	ns	XT and RC osc mode
			50	—	—	ns	HS osc mode
			5	—	—	µs	LP osc mode
		Oscillator Period (Note 1)	250	—	—	ns	RC osc mode
			250	—	10,000	ns	XT osc mode
			50	—	1,000	ns	HS osc mode
			5	—	—	µs	LP osc mode
2	TCY	Instruction Cycle Time (Note 1)	1.0	FOSC/4	DC	µs	TCYS=FOSC/4
3*	TosL, TosH	External Clock in (OSC1) High or Low Time	100*	—	—	ns	XT oscillator, TOSC L/H duty cycle
			2*	—	—	µs	LP oscillator, TOSC L/H duty cycle
			20*	—	—	ns	HS oscillator, TOSC L/H duty cycle
4*	TosR, TosF	External Clock in (OSC1) Rise or Fall Time	25*	—	—	ns	XT oscillator
			50*	—	—	ns	LP oscillator
			15*	—	—	ns	HS oscillator

* 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 9-4: CLKOUT AND I/O TIMING



Note: All tests must be done with specified capacitance loads (Figure 9-2) 50 pF on I/O pins and CLKOUT.

TABLE 9-2: CLKOUT AND I/O TIMING REQUIREMENTS

Parameter #	Sym	Characteristic	Min	Typ†	Max	Units
10*	TosH2ckL	OSC1↑ to CLKOUT↓ (1)	— —	75 —	200 400	ns ns
11*	TosH2ckH	OSC1↑ to CLKOUT↑ (1)	— —	75 —	200 400	ns ns
12*	TckR	CLKOUT rise time (1)	— —	35 —	100 200	ns ns
13*	TckF	CLKOUT fall time (1)	— —	35 —	100 200	ns ns
14*	TckL2ioV	CLKOUT ↓ to Port out valid (1)	—	—	20	ns
15*	TioV2ckH	Port in valid before CLKOUT ↑ (1)	TOSC +200 ns TOSC +400 ns	— —	— —	ns ns
16*	TckH2ioI	Port in hold after CLKOUT ↑ (1)	0	—	—	ns
17*	TosH2ioV	OSC1↑ (Q1 cycle) to Port out valid	— —	50	150 300	ns ns
18*	TosH2ioI	OSC1↑ (Q2 cycle) to Port input invalid (I/O in hold time)	100 200	— —	— —	ns ns
19*	TioV2osH	Port input valid to OSC1↑ (I/O in setup time)	0	—	—	ns
20*	TioR	Port output rise time	— —	10 —	40 80	ns ns
21*	TioF	Port output fall time	— —	10 —	40 80	ns ns
22*	Tinp	RB0/INT pin high or low time	25 40	— —	— —	ns ns
23	Trbp	RB<7:4> change interrupt high or low time	TCY	—	—	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.

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

FIGURE 9-5: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING

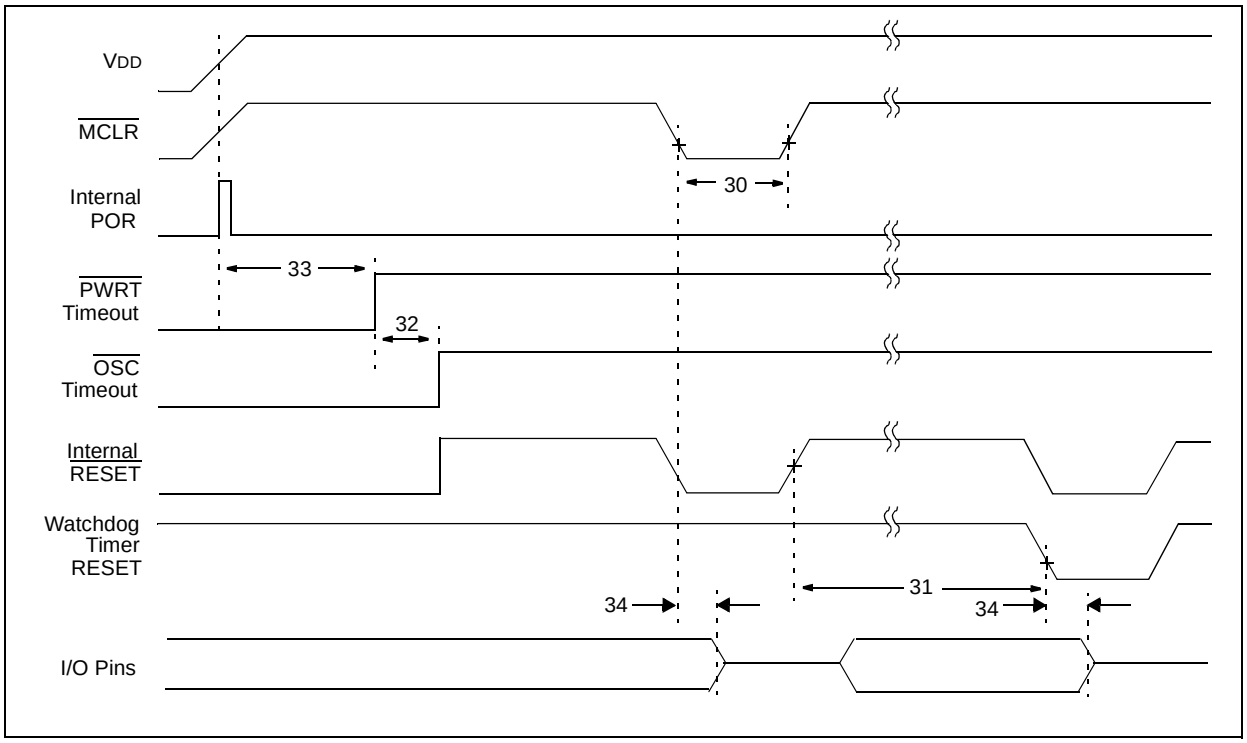


TABLE 9-3: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER REQUIREMENTS

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
30	TmcL	$\overline{\text{MCLR}}$ Pulse Width (low)	2000	—	—	ns	-0° to +70°C
31	Twdt	Watchdog Timer Time-out Period (No Prescaler)	7*	18	33*	ms	VDD = 5.0V, -0° to +70°C
32	Tost	Oscillation Start-up Timer Period	—	1024 TOSC	—	—	TOSC = OSC1 period
33	Tpwrt	Power-up Timer Period	28*	72	132*	ms	VDD = 5.0V, -0° to +70°C
34	TIOZ	I/O hi-impedance from $\overline{\text{MCLR}}$ low		—	2.0	µs	

* 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.

FIGURE 9-6: TIMER0 CLOCK TIMING

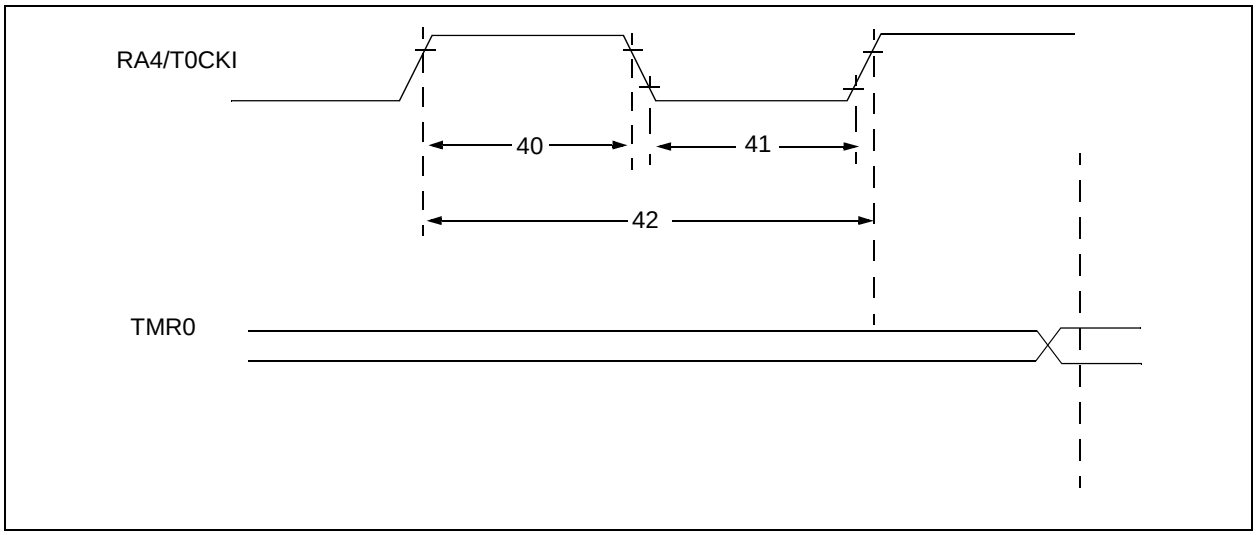


TABLE 9-4: TIMER0 CLOCK REQUIREMENTS

Parameter No.	Sym	Characteristic		Min	Typ†	Max	Units	Conditions
40	Tt0H	T0CKI High Pulse Width	No Prescaler	0.5 TCY + 20*	—	—	ns	
			With Prescaler	10*	—	—	ns	
41	Tt0L	T0CKI Low Pulse Width	No Prescaler	0.5 TCY + 20*	—	—	ns	
			With Prescaler	10*	—	—	ns	
42	Tt0P	T0CKI Period		$\frac{T_{CY} + 40}{N}$*	—	—	ns	N = prescale value (1, 2, 4, ..., 256)

* 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.

10.0 DEVICE CHARACTERIZATION INFORMATION

The graphs and tables provided in this section are for design guidance and are not tested. In some graphs or tables, the data presented is outside specified operating range (e.g., outside specified VDD range). This is for information only and devices will operate properly only within the specified range.

The data presented in this section is a statistical summary of data collected on units from different lots over a period of time. “Typical” represents the mean of the distribution, while “max” or “min” represents (mean + 3σ) and (mean – 3σ) respectively, where σ is standard deviation.

FIGURE 10-1: IDD VS. FREQUENCY (XT MODE, VDD = 5.5V)

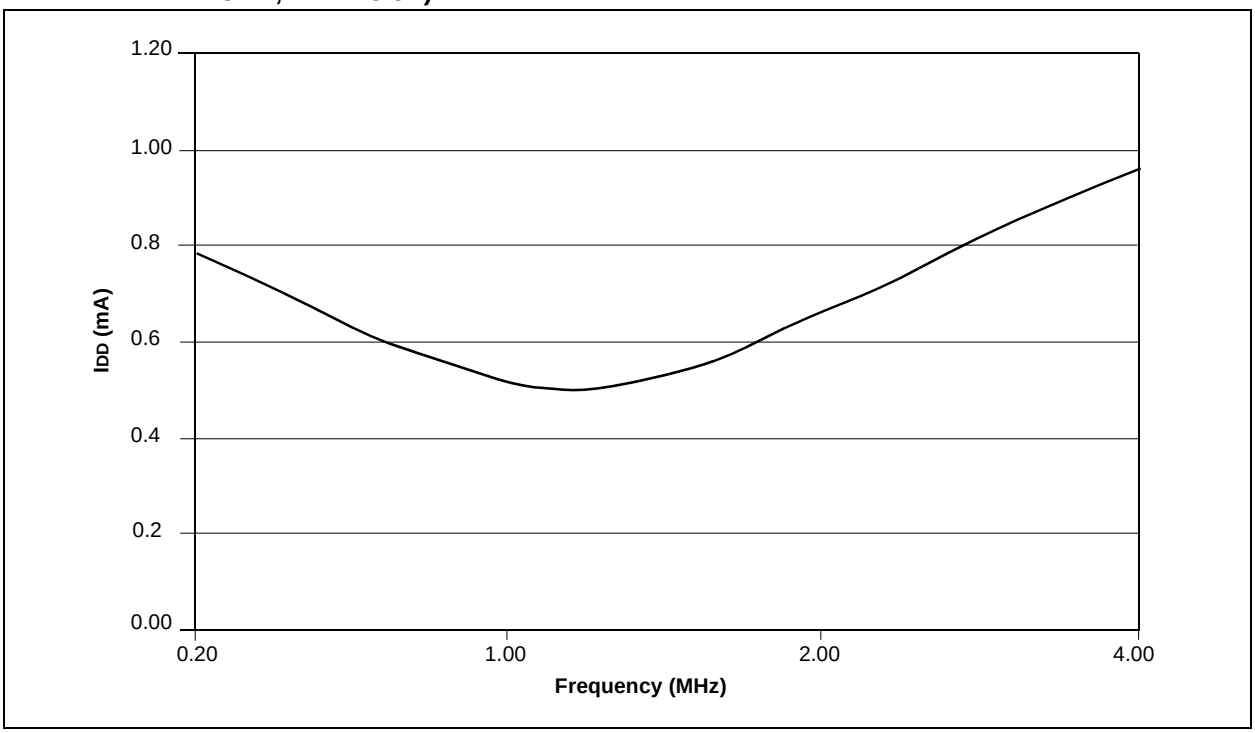


FIGURE 10-2: CF7685 I_{PD} vs. V_{DD} (WDT Disable)

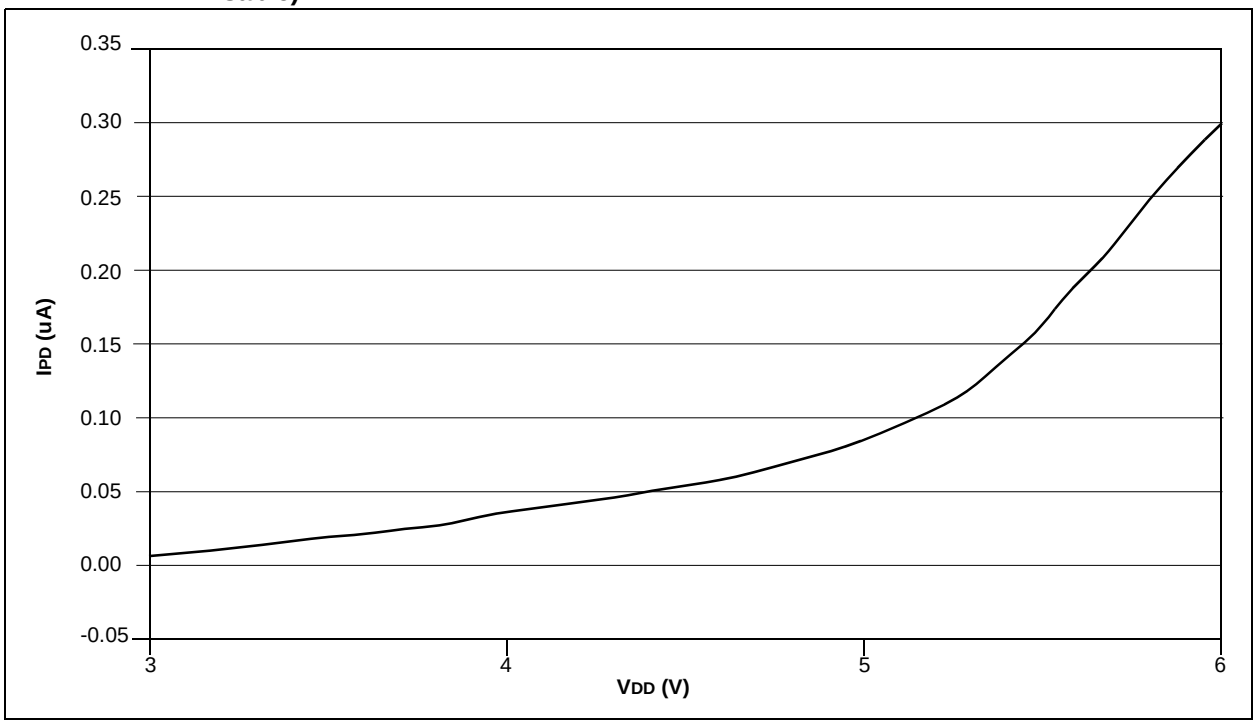


FIGURE 10-3: IDD vs. VDD (XT OSC 4MHz)

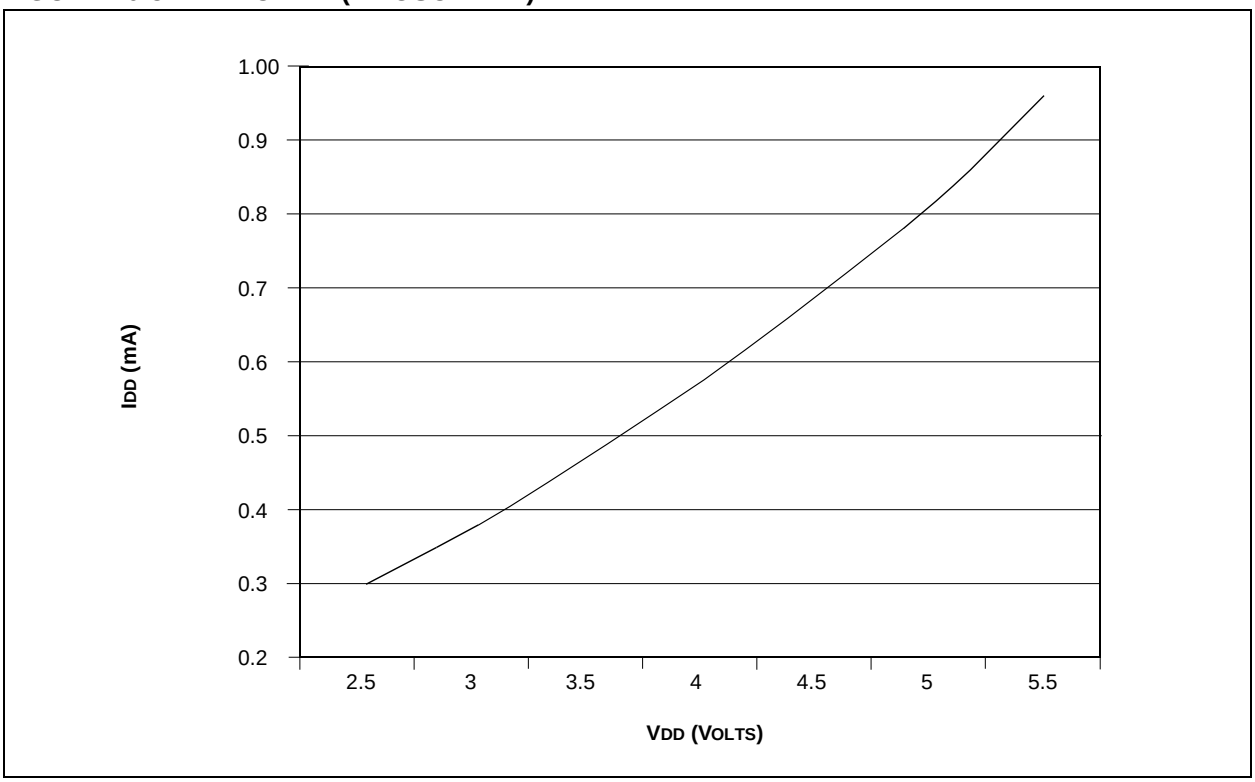


FIGURE 10-4: IOL vs. VOL, VDD = 3.0V

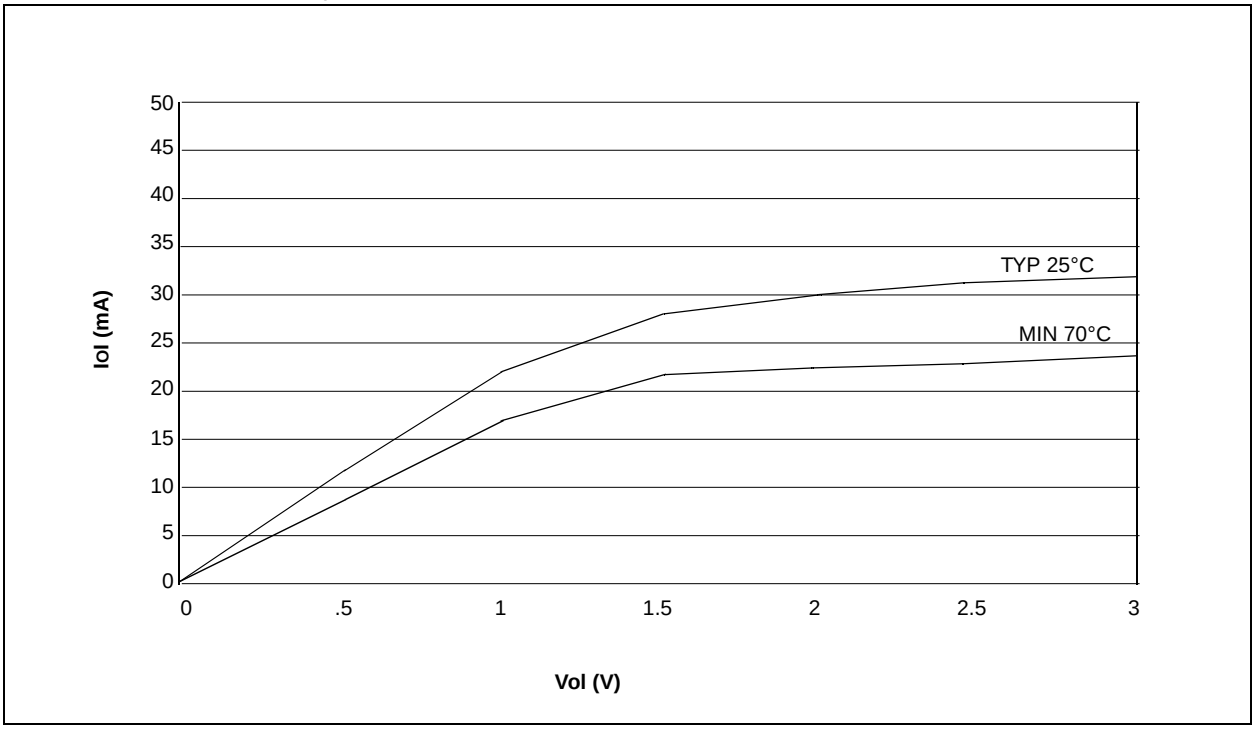


FIGURE 10-5: IOH VS. VOH, VDD = 3.0V)

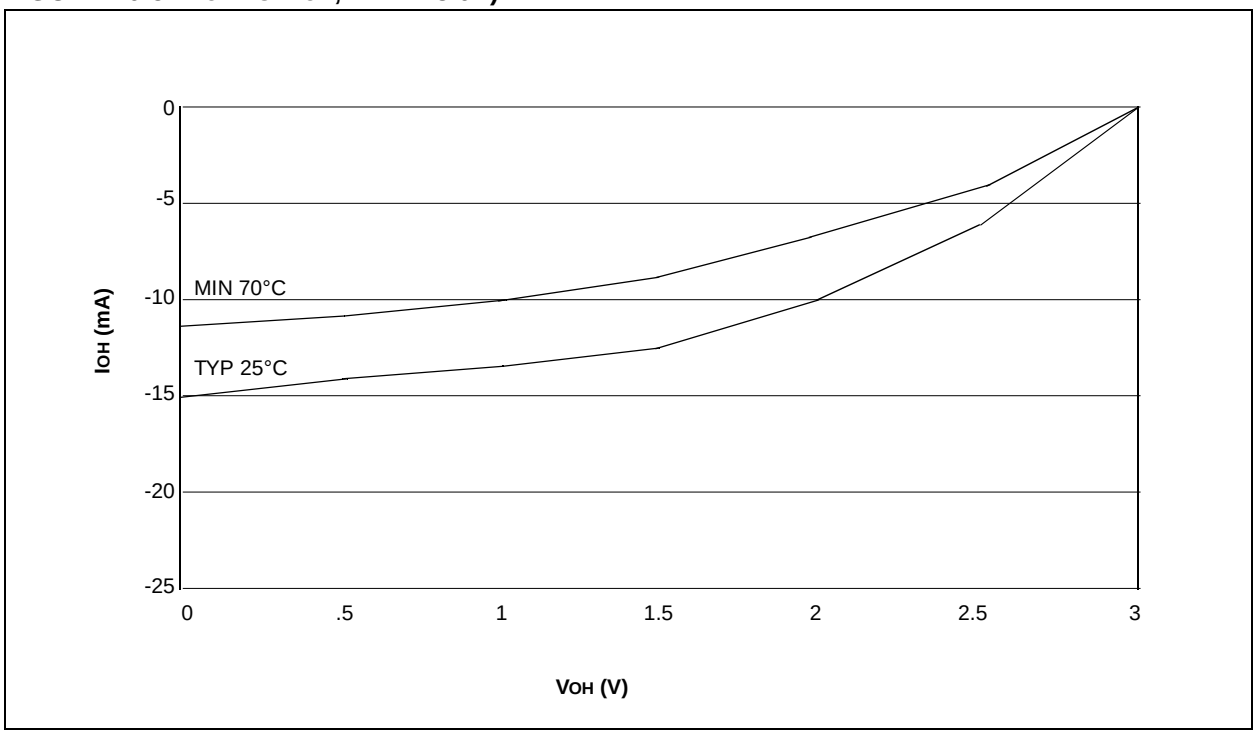


FIGURE 10-6: IOl VS. VOL, VDD = 5.5V)

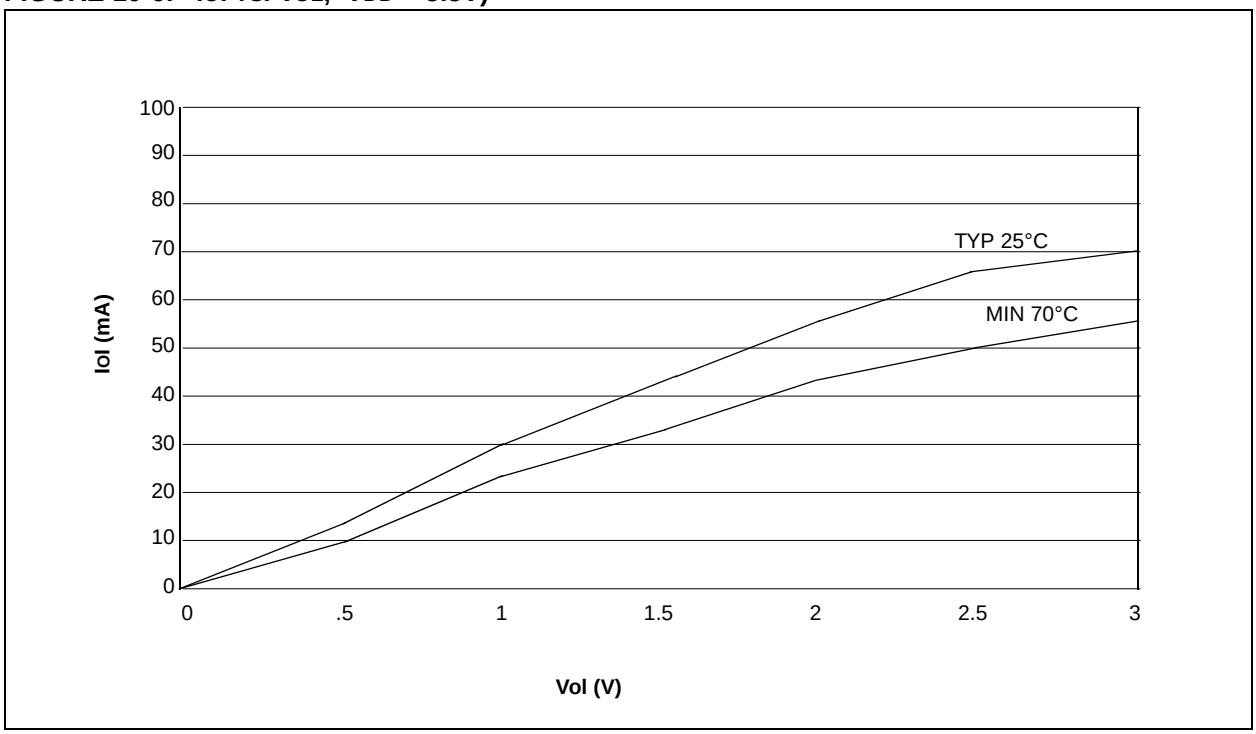
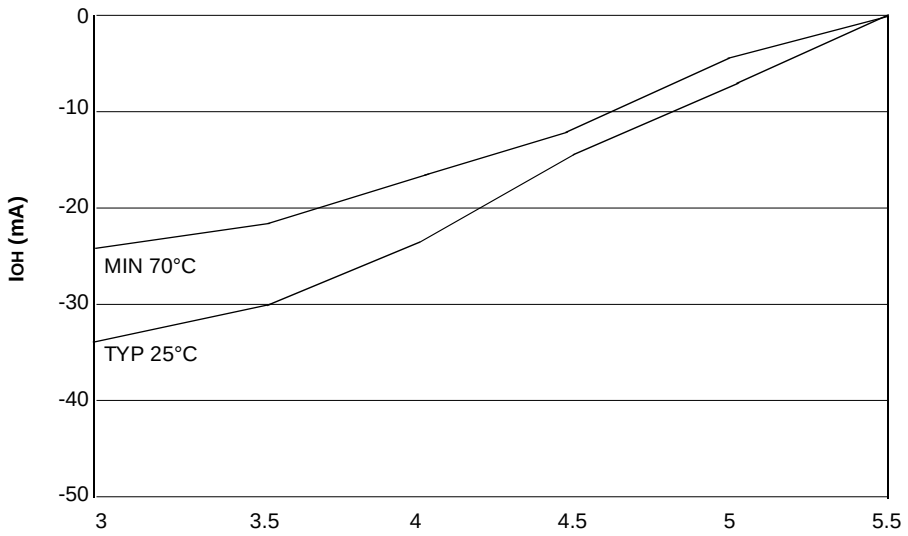


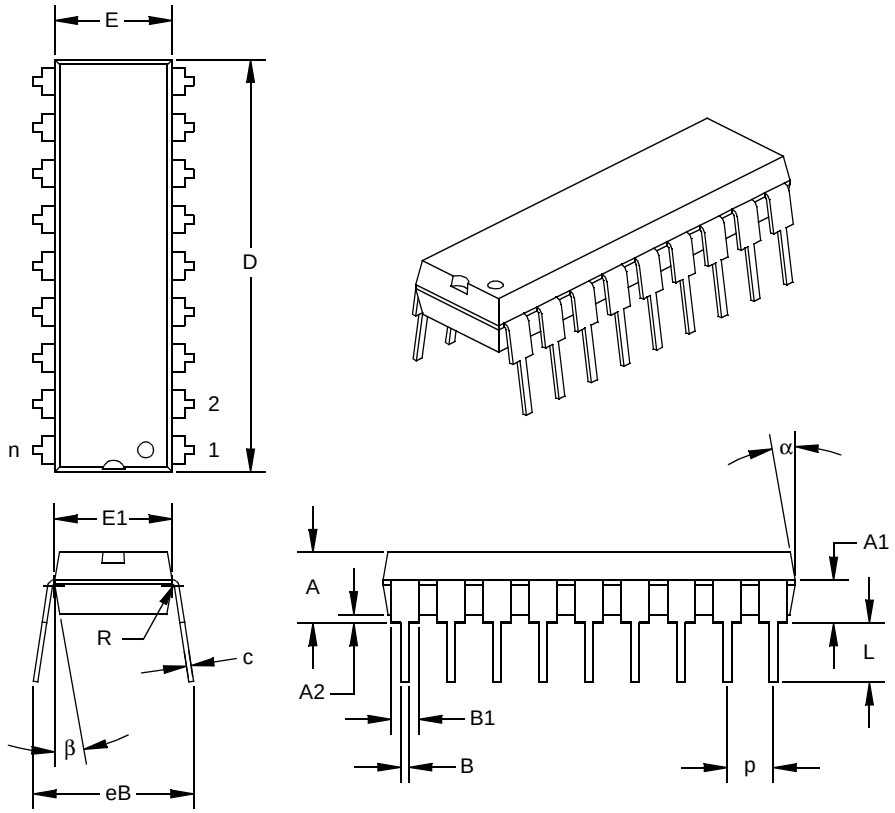
FIGURE 10-7: I_{OH} VS. V_{OH}, V_{DD} = 5.5V)



NOTES:

11.0 PACKAGING INFORMATION

Package Type: K04-007 18-Lead Plastic Dual In-line (P) – 300 mil



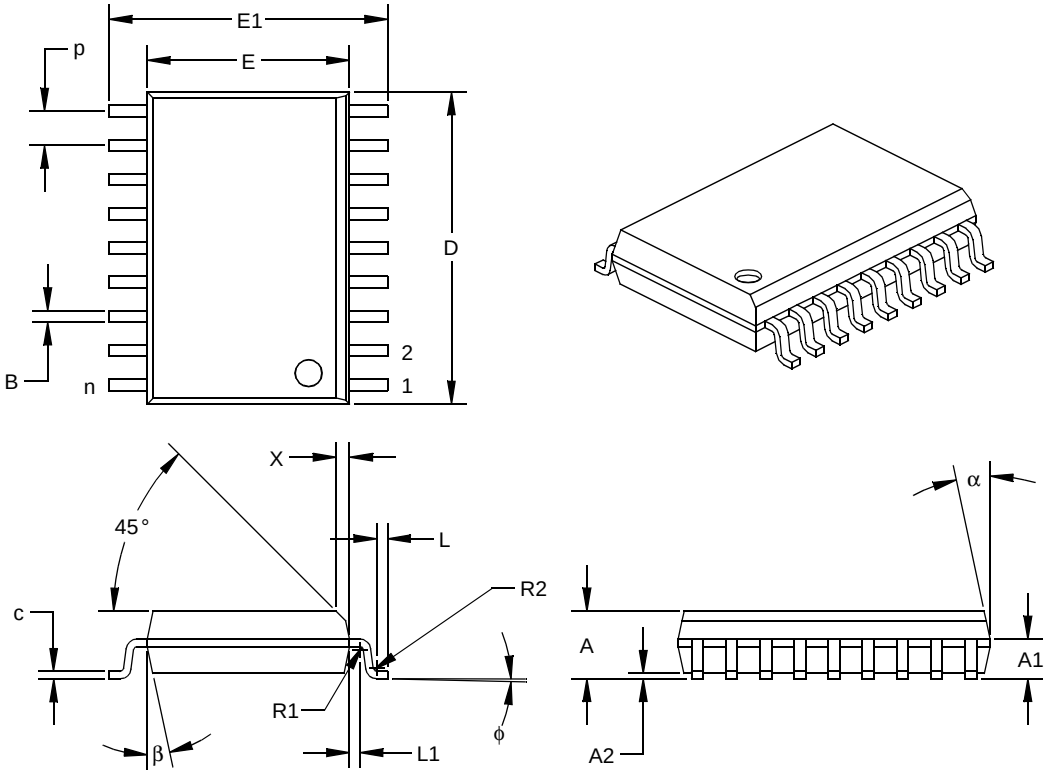
Units		INCHES*			MILLIMETERS		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
PCB Row Spacing			0.300			7.62	
Number of Pins	n		18			18	
Pitch	p		0.100			2.54	
Lower Lead Width	B	0.013	0.018	0.023	0.33	0.46	0.58
Upper Lead Width	B1†	0.055	0.060	0.065	1.40	1.52	1.65
Shoulder Radius	R	0.000	0.005	0.010	0.00	0.13	0.25
Lead Thickness	c	0.005	0.010	0.015	0.13	0.25	0.38
Top to Seating Plane	A	0.110	0.155	0.155	2.79	3.94	3.94
Top of Lead to Seating Plane	A1	0.075	0.095	0.115	1.91	2.41	2.92
Base to Seating Plane	A2	0.000	0.020	0.020	0.00	0.51	0.51
Tip to Seating Plane	L	0.125	0.130	0.135	3.18	3.30	3.43
Package Length	D‡	0.890	0.895	0.900	22.61	22.73	22.86
Molded Package Width	E‡	0.245	0.255	0.265	6.22	6.48	6.73
Radius to Radius Width	E1	0.230	0.250	0.270	5.84	6.35	6.86
Overall Row Spacing	eB	0.310	0.349	0.387	7.87	8.85	9.83
Mold Draft Angle Top	α	5	10	15	5	10	15
Mold Draft Angle Bottom	β	5	10	15	5	10	15

* Controlling Parameter.

† Dimension “B1” does not include dam-bar protrusions. Dam-bar protrusions shall not exceed 0.003” (0.076 mm) per side or 0.006” (0.152 mm) more than dimension “B1.”

‡ Dimensions “D” and “E” do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.010” (0.254 mm) per side or 0.020” (0.508 mm) more than dimensions “D” or “E.”

Package Type: K04-051 18-Lead Plastic Small Outline (SO) – Wide, 300 mil



Units		INCHES*			MILLIMETERS		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Pitch	p		0.050			1.27	
Number of Pins	n		18			18	
Overall Pack. Height	A	0.093	0.099	0.104	2.36	2.50	2.64
Shoulder Height	A1	0.048	0.058	0.068	1.22	1.47	1.73
Standoff	A2	0.004	0.008	0.011	0.10	0.19	0.28
Molded Package Length	D‡	0.450	0.456	0.462	11.43	11.58	11.73
Molded Package Width	E‡	0.292	0.296	0.299	7.42	7.51	7.59
Outside Dimension	E1	0.394	0.407	0.419	10.01	10.33	10.64
Chamfer Distance	X	0.010	0.020	0.029	0.25	0.50	0.74
Shoulder Radius	R1	0.005	0.005	0.010	0.13	0.13	0.25
Gull Wing Radius	R2	0.005	0.005	0.010	0.13	0.13	0.25
Foot Length	L	0.011	0.016	0.021	0.28	0.41	0.53
Foot Angle	ϕ	0	4	8	0	4	8
Radius Centerline	L1	0.010	0.015	0.020	0.25	0.38	0.51
Lead Thickness	c	0.009	0.011	0.012	0.23	0.27	0.30
Lower Lead Width	B†	0.014	0.017	0.019	0.36	0.42	0.48
Mold Draft Angle Top	α	0	12	15	0	12	15
Mold Draft Angle Bottom	β	0	12	15	0	12	15

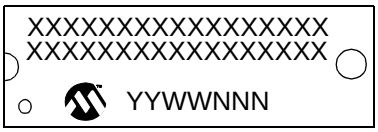
* Controlling Parameter.

† Dimension “B” does not include dam-bar protrusions. Dam-bar protrusions shall not exceed 0.003” (0.076 mm) per side or 0.006” (0.152 mm) more than dimension “B.”

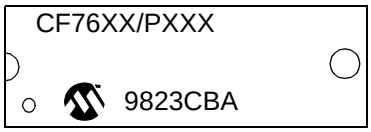
‡ Dimensions “D” and “E” do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.010” (0.254 mm) per side or 0.020” (0.508 mm) more than dimensions “D” or “E.”

11.1 Package Marking Information

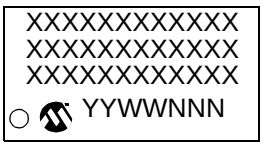
18-Lead PDIP



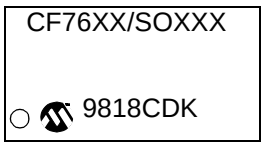
Example



18-Lead SOIC (.300")



Example



Legend:	XX...X	Customer specific information*
	YY	Year code (last 2 digits of calendar year)
	WW	Week code (week of January 1 is week '01')
	NNN	Alphanumeric traceability code

Note: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line thus limiting the number of available characters for customer specific information.

***** Standard OTP marking consists of Microchip part number, year code, week code, facility code, mask rev#, and assembly code. For OTP marking beyond this, certain price adders apply. Please check with your Microchip Sales Office. For QTP devices, any special marking adders are included in QTP price.

NOTES:

INDEX

T

W

X

PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.

PART NO.	/XX	XXX
Device	Package	Pattern

Device	CF76XX
Package	P = PDIP SO = SOIC (Gull Wing, 300 mil body)
Pattern	QTP, SQTP, (factory specified) or Special Requirements . Blank for OTP and Windowed devices.

Examples:

a) CF7665/P = Commercial Temp.,
PDiP package,
normal V_{DD} limits

b) CF7645/SO = Commercial Temp.,
SOIC package,
Normal V_{DD} limits

Sales and Support

Data Sheets
Products supported by a preliminary Data Sheet may have an errata sheet describing minor operational differences and recommended workarounds. To determine if an errata sheet exists for a particular device, please contact your local Microchip sales office.

Please specify which device, revision of silicon and Data Sheet (include Literature #) you are using.

NOTES: