



Welcome to [E-XFL.COM](https://www.e-xfl.com)

What is "[Embedded - Microcontrollers](#)"?

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

Applications of "[Embedded - Microcontrollers](#)"

Details

Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	10MHz
Connectivity	-
Peripherals	POR, WDT
Number of I/O	5
Program Memory Size	1.75KB (1K x 14)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	128 x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 5.5V
Data Converters	A/D 4x8b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	8-SOIC (0.209", 5.30mm Width)
Supplier Device Package	8-SOIJ
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic12c671-10i-sm

PIC12C67X

NOTES:

4.0 MEMORY ORGANIZATION

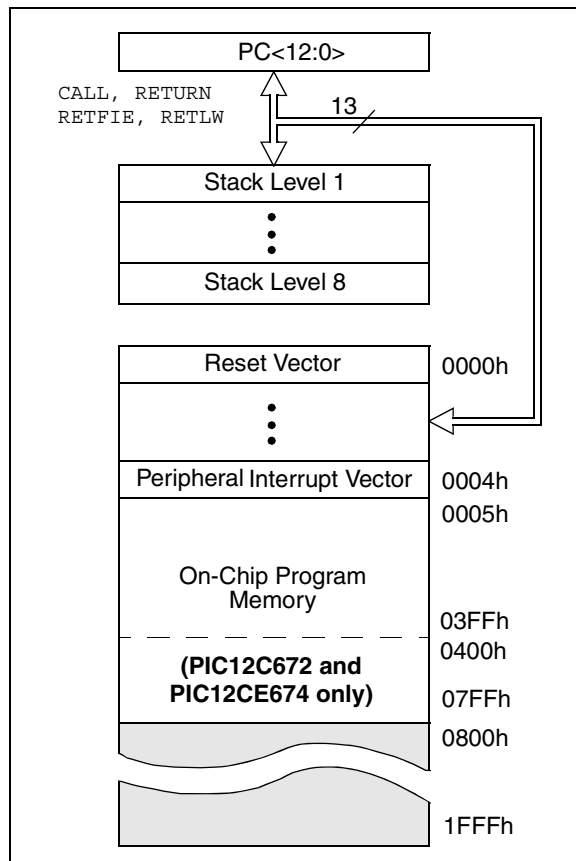
4.1 Program Memory Organization

The PIC12C67X has a 13-bit program counter capable of addressing an 8K x 14 program memory space.

For the PIC12C671 and the PIC12CE673, the first 1K x 14 (0000h-03FFh) is implemented.

For the PIC12C672 and the PIC12CE674, the first 2K x 14 (0000h-07FFh) is implemented. Accessing a location above the physically implemented address will cause a wraparound. The reset vector is at 0000h and the interrupt vector is at 0004h.

FIGURE 4-1: PIC12C67X PROGRAM MEMORY MAP AND STACK



4.2 Data Memory Organization

The data memory is partitioned into two banks, which contain the General Purpose Registers and the Special Function Registers. Bit RP0 is the bank select bit.

RP0 (STATUS<5>) = 1 → Bank 1

RP0 (STATUS<5>) = 0 → Bank 0

Each Bank extends up to 7Fh (128 bytes). The lower locations of each Bank are reserved for the Special Function Registers. Above the Special Function Registers are General Purpose Registers implemented as static RAM. Both Bank 0 and Bank 1 contain Special Function Registers. Some "high use" Special Function Registers from Bank 0 are mirrored in Bank 1 for code reduction and quicker access.

Also note that F0h through FFh on the PIC12C67X is mapped into Bank 0 registers 70h-7Fh as common RAM.

4.2.1 GENERAL PURPOSE REGISTER FILE

The register file can be accessed either directly or indirectly through the File Select Register FSR (Section 4.5).

PIC12C67X

FIGURE 4-2: PIC12C67X REGISTER FILE MAP

File Address			File Address
00h	INDF ⁽¹⁾	INDF ⁽¹⁾	80h
01h	TMR0	OPTION	81h
02h	PCL	PCL	82h
03h	STATUS	STATUS	83h
04h	FSR	FSR	84h
05h	GPIO	TRIS	85h
06h			86h
07h			87h
08h			88h
09h			89h
0Ah	PCLATH	PCLATH	8Ah
0Bh	INTCON	INTCON	8Bh
0Ch	PIR1	PIE1	8Ch
0Dh			8Dh
0Eh		PCON	8Eh
0Fh		OSCCAL	8Fh
10h			90h
11h			91h
12h			92h
13h			93h
14h			94h
15h			95h
16h			96h
17h			97h
18h			98h
19h			99h
1Ah			9Ah
1Bh			9Bh
1Ch			9Ch
1Dh			9Dh
1Eh	ADRES		9Eh
1Fh	ADCON0	ADCON1	9Fh
20h	General Purpose Register	General Purpose Register	A0h
			BFh
			C0h
			EFh
70h		Mapped in Bank 0	F0h
7Fh			FFh
Bank 0		Bank 1	

Unimplemented data memory locations, read as '0'.

Note 1:

Not a physical register.

4.2.2 SPECIAL FUNCTION REGISTERS

The Special Function Registers are registers used by the CPU and Peripheral Modules for controlling the desired operation of the device. These registers are implemented as static RAM.

The Special Function Registers can be classified into two sets (core and peripheral). Those registers associated with the “core” functions are described in this section, and those related to the operation of the peripheral features are described in the section of that peripheral feature.

PIC12C67X

TABLE 4-1: PIC12C67X SPECIAL FUNCTION REGISTER SUMMARY (CONT.)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other Resets ⁽³⁾
Bank 1											
80h ⁽¹⁾	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	0000 0000
81h	OPTION	GPPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111
82h ⁽¹⁾	PCL	Program Counter's (PC) Least Significant Byte								0000 0000	0000 0000
83h ⁽¹⁾	STATUS	IRP ⁽⁴⁾	RP1 ⁽⁴⁾	RP0	T0	PD	Z	DC	C	0001 1xxx	000q quuu
84h ⁽¹⁾	FSR	Indirect data memory address pointer								xxxx xxxx	uuuu uuuu
85h	TRIS	—	—	GPIO Data Direction Register						--11 1111	--11 1111
86h	—	Unimplemented								—	—
87h	—	Unimplemented								—	—
88h	—	Unimplemented								—	—
89h	—	Unimplemented								—	—
8Ah ^(1,2)	PCLATH	—	—	—	Write Buffer for the upper 5 bits of the PC					---0 0000	---0 0000
8Bh ⁽¹⁾	INTCON	GIE	PEIE	T0IE	INTE	GPIE	T0IF	INTF	GPIF	0000 000x	0000 000u
8Ch	PIE1	—	ADIE	—	—	—	—	—	—	-0-- ----	-0-- ----
8Dh	—	Unimplemented								—	—
8Eh	PCON	—	—	—	—	—	—	POR	—	---- --0-	---- --u-
8Fh	OSCCAL	CAL3	CAL2	CAL1	CAL0	CALFST	CALSLW	—	—	0111 00--	uuuu uu--
90h	—	Unimplemented								—	—
91h	—	Unimplemented								—	—
92h	—	Unimplemented								—	—
93h	—	Unimplemented								—	—
94h	—	Unimplemented								—	—
95h	—	Unimplemented								—	—
96h	—	Unimplemented								—	—
97h	—	Unimplemented								—	—
98h	—	Unimplemented								—	—
99h	—	Unimplemented								—	—
9Ah	—	Unimplemented								—	—
9Bh	—	Unimplemented								—	—
9Ch	—	Unimplemented								—	—
9Dh	—	Unimplemented								—	—
9Eh	—	Unimplemented								—	—
9Fh	ADCON1	—	—	—	—	—	PCFG2	PCFG1	PCFG0	---- -000	---- -000

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

Shaded locations are unimplemented, read as '0'.

Note 1: These registers can be addressed from either bank.

2: The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<12:8> whose contents are transferred to the upper byte of the program counter.

3: Other (non power-up) resets include external reset through MCLR and Watchdog Timer Reset.

4: The IRP and RP1 bits are reserved on the PIC12C67X; always maintain these bits clear.

5: The SCL (GP7) and SDA (GP6) bits are unimplemented on the PIC12C671/672 and read as '0'.

PIC12C67X

4.2.2.2 OPTION REGISTER

The OPTION Register is a readable and writable register, which contains various control bits to configure the TMR0/WDT prescaler, the External INT Interrupt, TMR0 and the weak pull-ups on GPIO.

Note: To achieve a 1:1 prescaler assignment for the TMR0 register, assign the prescaler to the Watchdog Timer by setting bit PSA (OPTION<3>).

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
GPPU	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

bit 7: **GPPU**: Weak Pull-up Enable
1 = Weak pull-ups disabled
0 = Weak pull-ups enabled (GP0, GP1, GP3)

bit 6: **INTEDG**: Interrupt Edge
1 = Interrupt on rising edge of GP2/T0CKI/AN2/INT pin
0 = Interrupt on falling edge of GP2/T0CKI/AN2/INT pin

bit 5: **T0CS**: TMR0 Clock Source Select bit
1 = Transition on GP2/T0CKI/AN2/INT pin
0 = Internal instruction cycle clock (CLKOUT)

bit 4: **T0SE**: TMR0 Source Edge Select bit
1 = Increment on high-to-low transition on GP2/T0CKI/AN2/INT pin
0 = Increment on low-to-high transition on GP2/T0CKI/AN2/INT pin

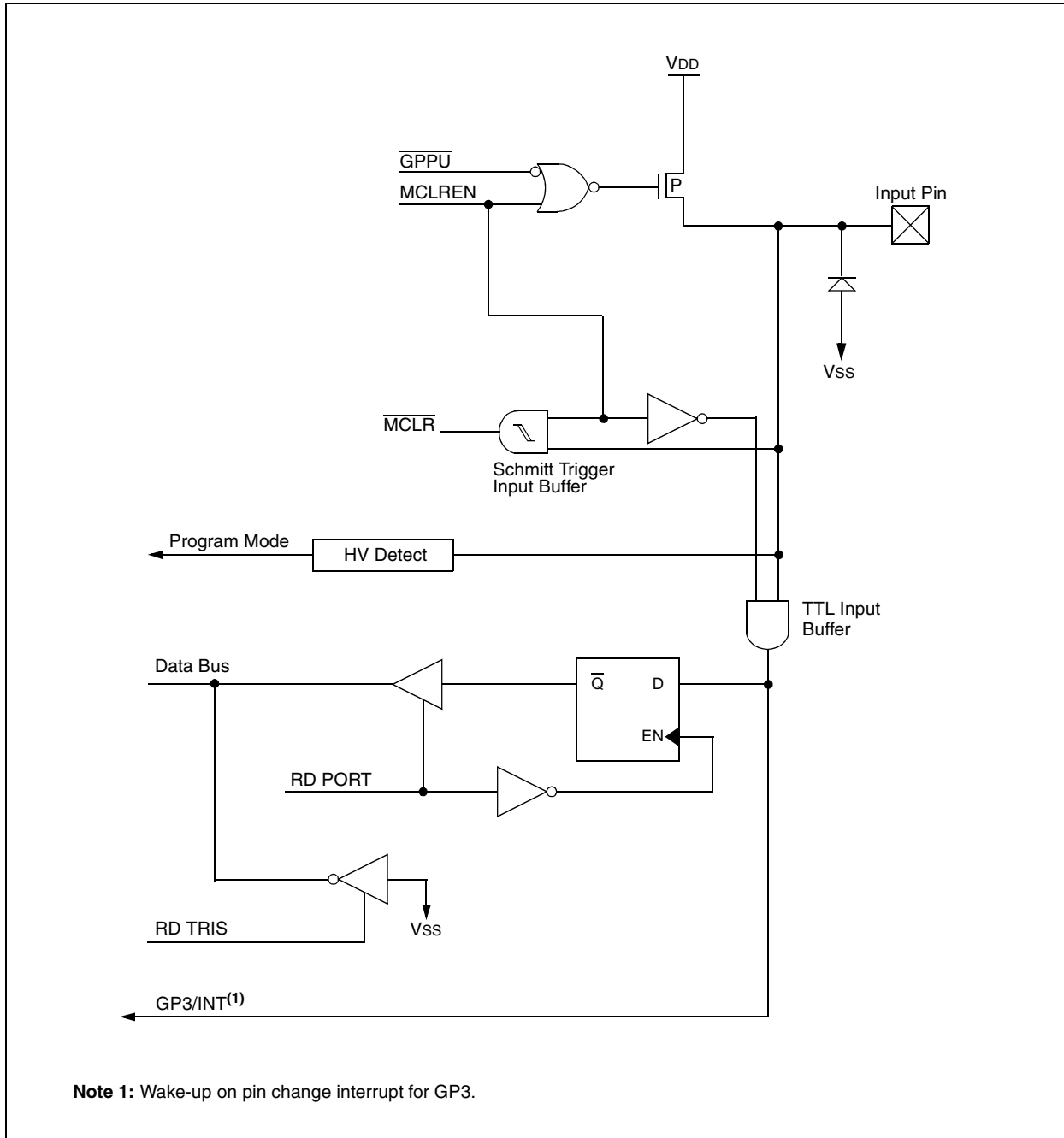
bit 3: **PSA**: Prescaler Assignment bit
1 = Prescaler is assigned to the WDT
0 = Prescaler is assigned to the Timer0 module

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

PIC12C67X

FIGURE 5-3: BLOCK DIAGRAM OF GP3/MCLR/VPP PIN



6.0 EEPROM PERIPHERAL OPERATION

The PIC12CE673 and PIC12CE674 each have 16 bytes of EEPROM data memory. The EEPROM memory has an endurance of 1,000,000 erase/write cycles and a data retention of greater than 40 years. The EEPROM data memory supports a bi-directional 2-wire bus and data transmission protocol. These two-wires are serial data (SDA) and serial clock (SCL), that are mapped to bit6 and bit7, respectively, of the GPIO register (SFR 06h). Unlike the GP0-GP5 that are connected to the I/O pins, SDA and SCL are only connected to the internal EEPROM peripheral. For most applications, all that is required is calls to the following functions:

```
; Byte_Write: Byte write routine
;   Inputs: EEPROM Address    EEADDR
;           EEPROM Data      EEDATA
;   Outputs: Return 01 in W if OK, else
;           return 00 in W
;
; Read_Current: Read EEPROM at address
;               currently held by EE device.
;   Inputs: NONE
;   Outputs: EEPROM Data      EEDATA
;           Return 01 in W if OK, else
;           return 00 in W
;
; Read_Random: Read EEPROM byte at supplied
;               address
;   Inputs: EEPROM Address    EEADDR
;   Outputs: EEPROM Data      EEDATA
;           Return 01 in W if OK,
;           else return 00 in W
```

The code for these functions is available on our web site (www.microchip.com). The code will be accessed by either including the source code FL67XINC.ASM or by linking FLASH67X.ASM. FLASH67X.INC provides external definition to the calling program.

6.0.1 SERIAL DATA

SDA is a bi-directional pin used to transfer addresses and data into and data out of the device.

For normal data transfer, SDA is allowed to change only during SCL low. Changes during SCL high are reserved for indicating the START and STOP conditions.

6.0.2 SERIAL CLOCK

This SCL signal is used to synchronize the data transfer from and to the EEPROM.

6.1 Bus Characteristics

The following **bus protocol** is to be used with the EEPROM data memory. In this section, the term “processor” is used to denote the portion of the PIC12C67X that interfaces to the EEPROM via software.

- Data transfer may be initiated only when the bus is not busy.

During data transfer, the data line must remain stable whenever the clock line is HIGH. Changes in the data line while the clock line is HIGH will be interpreted as a START or STOP condition.

Accordingly, the following bus conditions have been defined (Figure 6-3).

6.1.1 BUS NOT BUSY (A)

Both data and clock lines remain HIGH.

6.1.2 START DATA TRANSFER (B)

A HIGH to LOW transition of the SDA line while the clock (SCL) is HIGH determines a START condition. All commands must be preceded by a START condition.

6.1.3 STOP DATA TRANSFER (C)

A LOW to HIGH transition of the SDA line while the clock (SCL) is HIGH determines a STOP condition. All operations must be ended with a STOP condition.

6.1.4 DATA VALID (D)

The state of the data line represents valid data when, after a START condition, the data line is stable for the duration of the HIGH period of the clock signal.

The data on the line must be changed during the LOW period of the clock signal. There is one bit of data per clock pulse.

Each data transfer is initiated with a START condition and terminated with a STOP condition. The number of the data bytes transferred between the START and STOP conditions is determined by the available data EEPROM space.

PIC12C67X

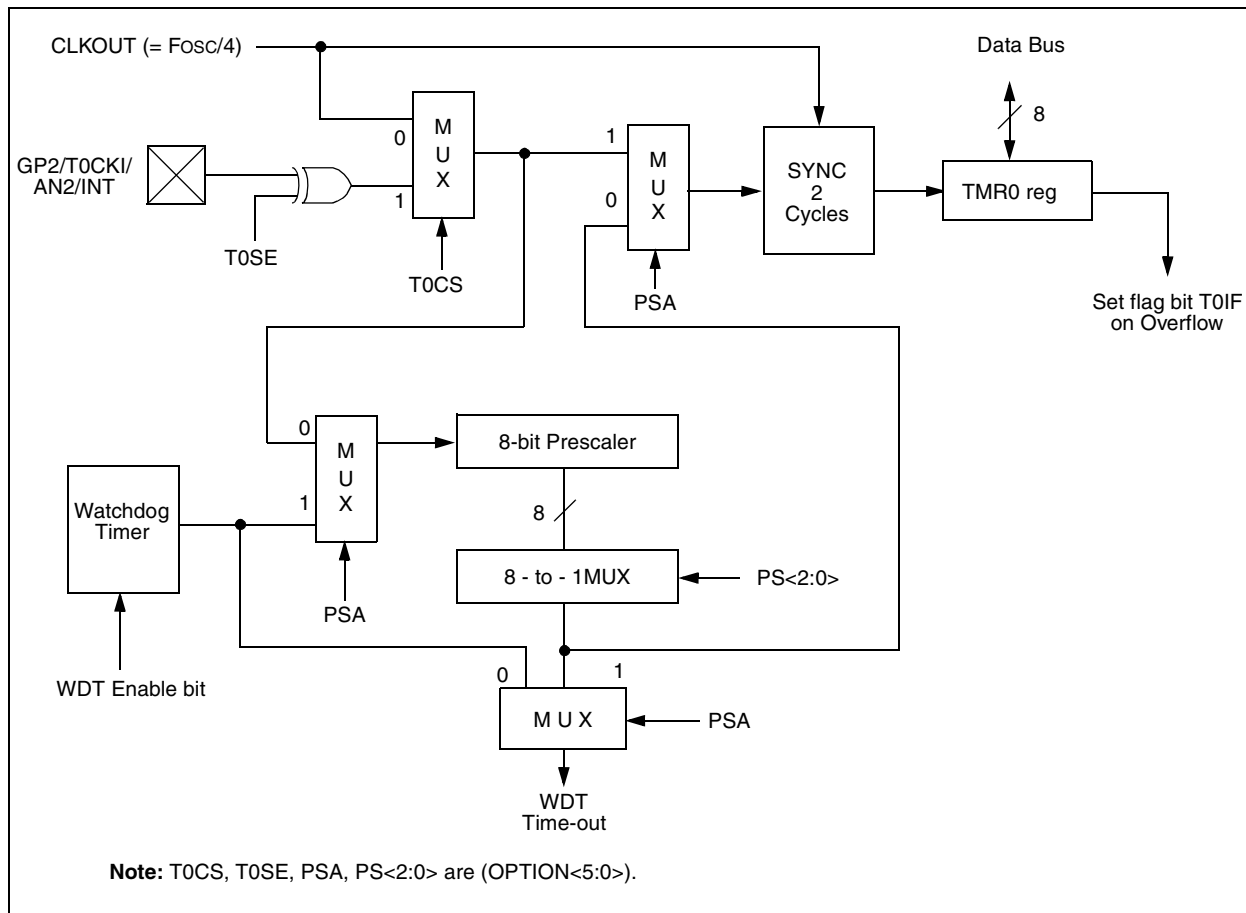
7.3 Prescaler

An 8-bit counter is available as a prescaler for the Timer0 module, or as a postscaler for the Watchdog Timer, respectively (Figure 7-6). 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 exclusively shared 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 (i.e., `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 7-6: BLOCK DIAGRAM OF THE TIMER0/WDT PRESCALER

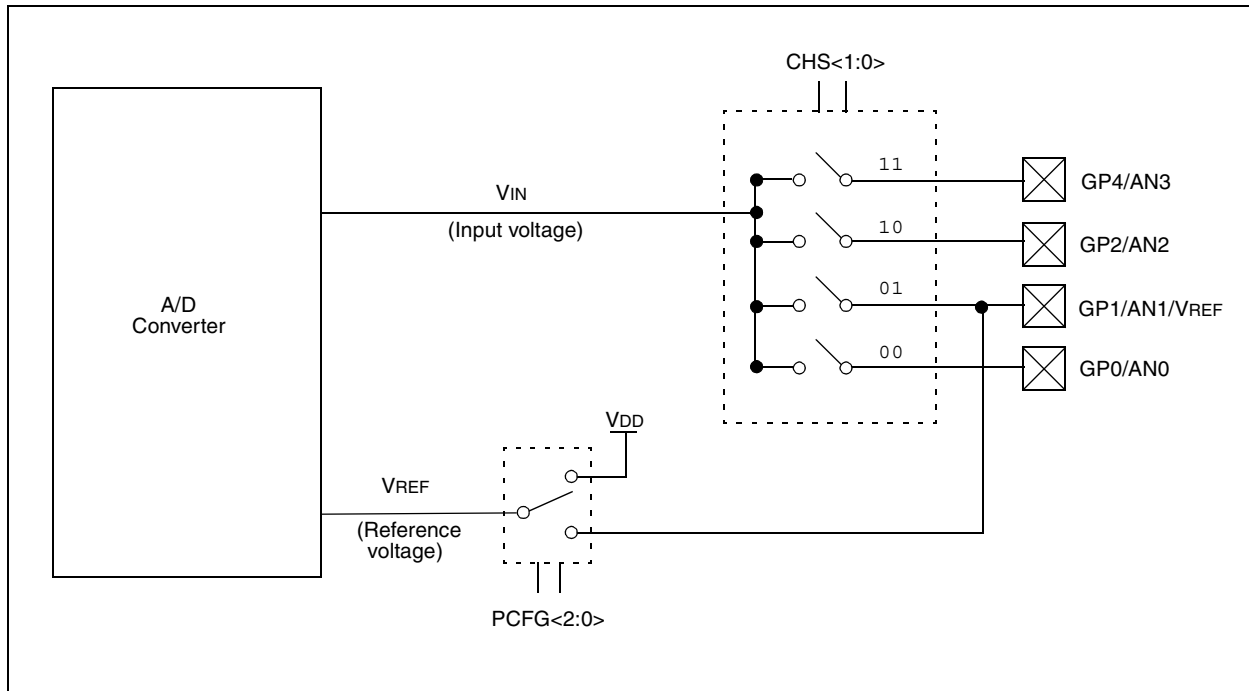


The ADRES Register contains the result of the A/D conversion. When the A/D conversion is complete, the result is loaded into the ADRES register, the $\overline{\text{GO/DONE}}$ bit (ADCON0<2>) is cleared, and A/D interrupt flag bit ADIF (PIE1<6>) is set. The block diagrams of the A/D module are shown in Figure 8-1.

After the A/D module has been configured as desired, the selected channel must be acquired before the conversion is started. The analog input channels must have their corresponding TRIS bits selected as an input. To determine sample time, see Section 8.1. After this acquisition time has elapsed, the A/D conversion can be started. The following steps should be followed for doing an A/D conversion:

1. Configure the A/D module:
 - Configure analog pins / voltage reference / and digital I/O (ADCON1 and TRIS)
 - Select A/D input channel (ADCON0)
 - Select A/D conversion clock (ADCON0)
 - Turn on A/D module (ADCON0)
2. Configure A/D interrupt (if desired):
 - Clear ADIF bit
 - Set ADIE bit
 - Set GIE bit
3. Wait the required acquisition time.
4. Start conversion:
 - Set $\overline{\text{GO/DONE}}$ bit (ADCON0)
5. Wait for A/D conversion to complete, by either:
 - Polling for the $\overline{\text{GO/DONE}}$ bit to be cleared
- OR
- Waiting for the A/D interrupt
6. Read A/D Result Register (ADRES), clear bit ADIF if required.
7. For the next conversion, go to step 1, step 2 or step 3 as required. The A/D conversion time per bit is defined as T_{AD} . A minimum wait of $2T_{AD}$ is required before next acquisition starts.

FIGURE 8-1: A/D BLOCK DIAGRAM



PIC12C67X

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

9.4.1 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 a Power-on Reset. A maximum rise time for VDD is specified. See Electrical Specifications for details.

When the device starts normal operation (exits the reset condition), device operating parameters (voltage, frequency, temperature, ...) 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."

9.4.2 POWER-UP TIMER (PWRT)

The Power-up Timer provides a fixed 72 ms nominal time-out on power-up only, from the POR. The Power-up Timer operates on an internal RC oscillator. The chip is kept in reset as long as the PWRT is active. The PWRT's time delay allows VDD to rise to an acceptable level. A configuration bit is provided to enable/disable the PWRT.

The power-up time delay will vary from chip to chip due to VDD, temperature and process variation. See Table 11-4.

9.4.3 OSCILLATOR START-UP TIMER (OST)

The Oscillator Start-up Timer (OST) provides 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.

9.4.4 TIME-OUT SEQUENCE

On power-up, the Time-out Sequence is as follows: first, PWRT time-out is invoked after the POR time delay has expired; then, OST is activated. The total time-out will vary, based on oscillator configuration and the status of the PWRT. For example, in RC mode with the PWRT disabled, there will be no time-out at all. Figure 9-7, Figure 9-8, and Figure 9-9 depict time-out sequences on power-up.

Since the time-outs occur from the POR pulse, if MCLR is kept low long enough, the time-outs will expire. Then bringing MCLR high will begin execution immediately (Figure 9-9). This is useful for testing purposes or to synchronize more than one PIC12C67X device operating in parallel.

9.4.5 POWER CONTROL (PCON)/STATUS REGISTER

The Power Control/Status Register, PCON (address 8Eh), has one bit. See Register 4-6 for register.

Bit1 is $\overline{\text{POR}}$ (Power-on Reset). It is cleared on a Power-on Reset and is unaffected otherwise. The user sets this bit following a Power-on Reset. On subsequent resets, if POR is '0', it will indicate that a Power-on Reset must have occurred.

TABLE 9-4: TIME-OUT IN VARIOUS SITUATIONS

Oscillator Configuration	Power-up		Wake-up from SLEEP
	$\overline{\text{PWRT}} = 0$	$\overline{\text{PWRT}} = 1$	
XT, HS, LP	72 ms + 1024Tosc	1024Tosc	1024Tosc
INTRC, EXTRC	72 ms	—	—

TABLE 9-5: STATUS/PCON BITS AND THEIR SIGNIFICANCE

POR	TO	PD	
0	1	1	Power-on Reset
0	0	x	Illegal, $\overline{\text{TO}}$ is set on $\overline{\text{POR}}$
0	x	0	Illegal, $\overline{\text{PD}}$ is set on $\overline{\text{POR}}$
1	0	u	WDT Reset
1	0	0	WDT Wake-up
1	u	u	$\overline{\text{MCLR}}$ Reset during normal operation
1	1	0	$\overline{\text{MCLR}}$ Reset during SLEEP or interrupt wake-up from SLEEP

Legend: u = unchanged, x = unknown.

PIC12C67X

9.5 Interrupts

There are four sources of interrupt:

Interrupt Sources
TMR0 Overflow Interrupt
External Interrupt GP2/INT pin
GPIO Port Change Interrupts (pins GP0, GP1, GP3)
A/D Interrupt

The Interrupt Control Register (INTCON) records individual interrupt requests in flag bits. It also has individual and global interrupt enable bits.

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

A global interrupt enable bit, GIE (INTCON<7>), enables (if set) all un-masked interrupts or disables (if cleared) all interrupts. When bit GIE is enabled and an interrupt's flag bit and mask bit are set, the interrupt will vector immediately. Individual interrupts can be disabled through their corresponding enable bits in various registers. Individual interrupt flag bits are set, regardless of the status of their corresponding mask bit or the GIE bit. The GIE bit is cleared on reset.

The "return-from-interrupt" instruction, `RETFIE`, exits the interrupt routine, as well as sets the GIE bit, which re-enables interrupts.

The GP2/INT, GPIO port change interrupt and the TMR0 overflow interrupt flags are contained in the INTCON register.

The peripheral interrupt flag ADIF, is contained in the Special Function Register PIR1. The corresponding interrupt enable bit is contained in Special Function Register PIE1, and the peripheral interrupt enable bit is contained in Special Function Register INTCON.

When an interrupt is responded to, the GIE bit is cleared to disable any further interrupt, the return address is pushed onto 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 repeated interrupts.

For external interrupt events, such as GPIO change interrupt, the interrupt latency will be three or four instruction cycles. The exact latency depends on when the interrupt event occurs (Figure 9-14). The latency is the same for one or two cycle instructions. Individual interrupt flag bits are set, regardless of the status of their corresponding mask bit or the GIE bit.

FIGURE 9-13: INTERRUPT LOGIC

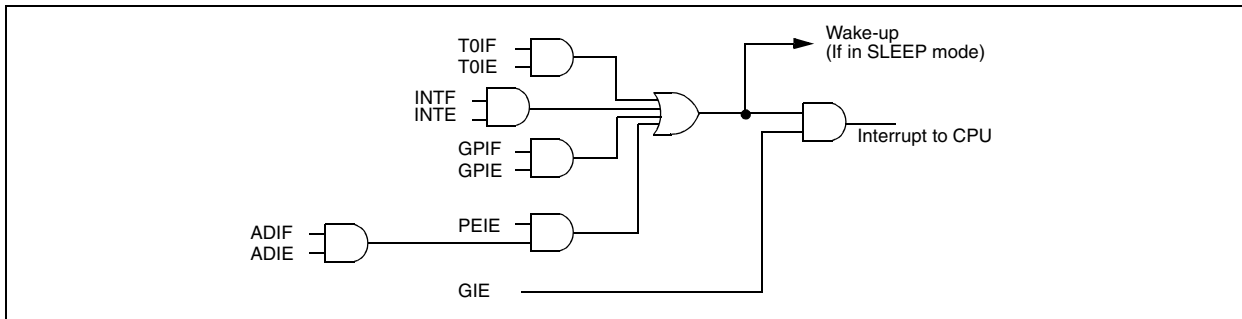
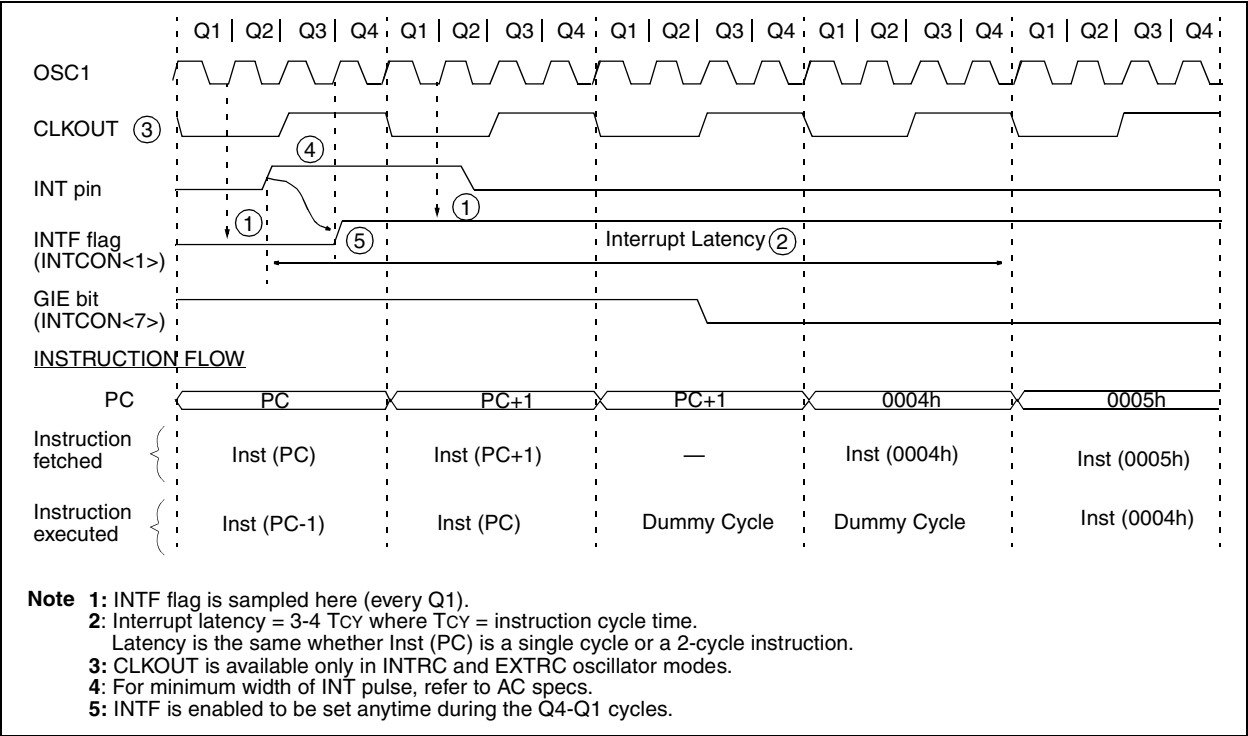
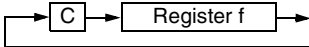


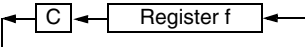
FIGURE 9-14: INT PIN INTERRUPT TIMING



RETURN		Return from Subroutine				
Syntax:	[<i>label</i>] 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					

RRF		Rotate Right f through Carry							
Syntax:	[<i>label</i>] RRF f,d								
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$								
Operation:	See description below								
Status Affected:	C								
Encoding:	<table border="1"><tr><td>00</td><td>1100</td><td>dfff</td><td>ffff</td></tr></table>					00	1100	dfff	ffff
00	1100	dfff	ffff						
Description:	<p>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'.</p> 								
Words:	1								
Cycles:	1								
Example	RRF								

Before Instruction
REG1 = 1110 0110
C = 0
After Instruction
REG1 = 1110 0110
W = 0111 0011
C = 0

RLF	Rotate Left f through Carry															
Syntax:	[<i>label</i>] RLF f,d															
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$															
Operation:	See description below															
Status Affected:	C															
Encoding:	<table><tr><td>00</td><td>1101</td><td>dfff</td><td>ffff</td></tr></table>	00	1101	dfff	ffff											
00	1101	dfff	ffff													
Description:	<p>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'.</p> 															
Words:	1															
Cycles:	1															
Example	<pre>RLF REG1, 0</pre> <p>Before Instruction</p> <table><tr><td>REG1</td><td>=</td><td>1110 0110</td></tr><tr><td>C</td><td>=</td><td>0</td></tr></table> <p>After Instruction</p> <table><tr><td>REG1</td><td>=</td><td>1110 0110</td></tr><tr><td>W</td><td>=</td><td>1100 1100</td></tr><tr><td>C</td><td>=</td><td>1</td></tr></table>	REG1	=	1110 0110	C	=	0	REG1	=	1110 0110	W	=	1100 1100	C	=	1
REG1	=	1110 0110														
C	=	0														
REG1	=	1110 0110														
W	=	1100 1100														
C	=	1														

SLEEP								
Syntax:	[<i>label</i>]	SLEEP						
Operands:	None							
Operation:	00h → WDT, 0 → WDT prescaler, 1 → \overline{TO} , 0 → PD							
Status Affected:	\overline{TO} , \overline{PD}							
Encoding:	<table><tr><td>00</td><td>0000</td><td>0110</td><td>0011</td></tr></table>				00	0000	0110	0011
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.							
Words:	1							
Cycles:	1							
Example:	SLEEP							

SWAPF		Swap Nibbles in f							
Syntax:	[<i>label</i>] SWAPF f,d								
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$								
Operation:	(f<3:0>) → (dest<7:4>), (f<7:4>) → (dest<3:0>)								
Status Affected:	None								
Encoding:	<table border="1"><tr><td>00</td><td>1110</td><td>dfff</td><td>ffff</td></tr></table>					00	1110	dfff	ffff
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:	[<i>label</i>] TRIS f									
Operands:	$5 \leq f \leq 7$									
Operation:	(W) → TRIS register f;									
Status Affected:	None									
Encoding:	<table border="1"><tr><td>00</td><td>0000</td><td>0110</td><td>0fff</td></tr></table>						00	0000	0110	0fff
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	<table border="1"><tr><td>To maintain upward compatibility with future PIC12C67X products, do not use this instruction.</td></tr></table>						To maintain upward compatibility with future PIC12C67X products, do not use this instruction.			
To maintain upward compatibility with future PIC12C67X products, do not use this instruction.										

XORLW	Exclusive OR Literal with W				
Syntax:	[<i>label</i>] XORLW k				
Operands:	0 ≤ k ≤ 255				
Operation:	(W) .XOR. k → (W)				
Status Affected:	Z				
Encoding:	<table><tr><td>11</td><td>1010</td><td>kkkk</td><td>kkkk</td></tr></table>	11	1010	kkkk	kkkk
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:	XORLW 0xAF Before Instruction W = 0xB5 After Instruction W = 0x1A				

XORWF

Exclusive OR W with f

Syntax:

[*label*] XORWF f,d

Operands:

0 ≤ f ≤ 127
d ∈ [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

XORWF REG 1

Before Instruction

REG = 0xAF

W = 0xB5

After Instruction

REG = 0x1A

W = 0xB5

11.0 DEVELOPMENT SUPPORT

The PIC® microcontrollers are supported with a full range of hardware and software development tools:

- Integrated Development Environment
 - MPLAB® IDE Software
- Assemblers/Compilers/Linkers
 - MPASM Assembler
 - MPLAB-C17 and MPLAB-C18 C Compilers
 - MPLINK/MPLIB Linker/Librarian
- Simulators
 - MPLAB-SIM Software Simulator
- Emulators
 - MPLAB-ICE Real-Time In-Circuit Emulator
 - PICMASTER®/PICMASTER-CE In-Circuit Emulator
 - ICEPIC™
- In-Circuit Debugger
 - MPLAB-ICD for PIC16F877
- Device Programmers
 - PRO MATE® II Universal Programmer
 - PICSTART® Plus Entry-Level Prototype Programmer
- Low-Cost Demonstration Boards
 - SIMICE
 - PICDEM-1
 - PICDEM-2
 - PICDEM-3
 - PICDEM-17
 - SEEVAL®
 - KEELOQ®

11.1 MPLAB Integrated Development Environment Software

The MPLAB IDE software brings an ease of software development previously unseen in the 8-bit microcontroller market. MPLAB is a Windows®-based application which contains:

- Multiple functionality
 - editor
 - simulator
 - programmer (sold separately)
 - emulator (sold separately)
- A full featured editor
- A project manager
- Customizable tool bar and key mapping
- A status bar
- On-line help

MPLAB allows you to:

- Edit your source files (either assembly or 'C')
- One touch assemble (or compile) and download to PIC MCU tools (automatically updates all project information)
- Debug using:
 - source files
 - absolute listing file
 - object code

The ability to use MPLAB with Microchip's simulator, MPLAB-SIM, allows a consistent platform and the ability to easily switch from the cost-effective simulator to the full featured emulator with minimal retraining.

11.2 MPASM Assembler

MPASM is a full featured universal macro assembler for all PIC MCUs. It can produce absolute code directly in the form of HEX files for device programmers, or it can generate relocatable objects for MPLINK.

MPASM has a command line interface and a Windows shell and can be used as a standalone application on a Windows 3.x or greater system. MPASM generates relocatable object files, Intel standard HEX files, MAP files to detail memory usage and symbol reference, an absolute LST file which contains source lines and generated machine code, and a COD file for MPLAB debugging.

MPASM features include:

- MPASM and MPLINK are integrated into MPLAB projects.
- MPASM allows user defined macros to be created for streamlined assembly.
- MPASM allows conditional assembly for multi purpose source files.
- MPASM directives allow complete control over the assembly process.

11.3 MPLAB-C17 and MPLAB-C18 C Compilers

The MPLAB-C17 and MPLAB-C18 Code Development Systems are complete ANSI 'C' compilers and integrated development environments for Microchip's PIC17CXXX and PIC18CXXX family of microcontrollers, respectively. These compilers provide powerful integration capabilities and ease of use not found with other compilers.

For easier source level debugging, the compilers provide symbol information that is compatible with the MPLAB IDE memory display.

11.4 MPLINK/MPLIB Linker/Librarian

MPLINK is a relocatable linker for MPASM and MPLAB-C17 and MPLAB-C18. It can link relocatable objects from assembly or C source files along with pre-compiled libraries using directives from a linker script.

MPLIB is a librarian for pre-compiled code to be used with MPLINK. When a routine from a library is called from another source file, only the modules that contains that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications. MPLIB manages the creation and modification of library files.

MPLINK features include:

- MPLINK works with MPASM and MPLAB-C17 and MPLAB-C18.
- MPLINK allows all memory areas to be defined as sections to provide link-time flexibility.

MPLIB features include:

- MPLIB makes linking easier because single libraries can be included instead of many smaller files.
- MPLIB helps keep code maintainable by grouping related modules together.
- MPLIB commands allow libraries to be created and modules to be added, listed, replaced, deleted, or extracted.

11.5 MPLAB-SIM Software Simulator

The MPLAB-SIM Software Simulator allows code development in a PC host environment by simulating the PIC series microcontrollers on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a file or user-defined key press to any of the pins. The execution can be performed in single step, execute until break, or trace mode.

MPLAB-SIM fully supports symbolic debugging using MPLAB-C17 and MPLAB-C18 and MPASM. The Software Simulator offers the flexibility to develop and debug code outside of the laboratory environment making it an excellent multi-project software development tool.

11.6 MPLAB-ICE High Performance Universal In-Circuit Emulator with MPLAB IDE

The MPLAB-ICE Universal In-Circuit Emulator is intended to provide the product development engineer with a complete microcontroller design tool set for PIC microcontrollers (MCUs). Software control of MPLAB-ICE is provided by the MPLAB Integrated Development Environment (IDE), which allows editing, “make” and download, and source debugging from a single environment.

Interchangeable processor modules allow the system to be easily reconfigured for emulation of different processors. The universal architecture of the MPLAB-ICE allows expansion to support new PIC microcontrollers.

The MPLAB-ICE Emulator System has been designed as a real-time emulation system with advanced features that are generally found on more expensive devel-

opment tools. The PC platform and Microsoft® Windows 3.x/95/98 environment were chosen to best make these features available to you, the end user.

MPLAB-ICE 2000 is a full-featured emulator system with enhanced trace, trigger, and data monitoring features. Both systems use the same processor modules and will operate across the full operating speed range of the PIC MCU.

11.7 PICMASTER/PICMASTER CE

The PICMASTER system from Microchip Technology is a full-featured, professional quality emulator system. This flexible in-circuit emulator provides a high-quality, universal platform for emulating Microchip 8-bit PIC microcontrollers (MCUs). PICMASTER systems are sold worldwide, with a CE compliant model available for European Union (EU) countries.

11.8 ICEPIC

ICEPIC is a low-cost in-circuit emulation solution for the Microchip Technology PIC16C5X, PIC16C6X, PIC16C7X, and PIC16CXXX families of 8-bit one-time-programmable (OTP) microcontrollers. The modular system can support different subsets of PIC16C5X or PIC16CXXX products through the use of interchangeable personality modules or daughter boards. The emulator is capable of emulating without target application circuitry being present.

11.9 MPLAB-ICD In-Circuit Debugger

Microchip's In-Circuit Debugger, MPLAB-ICD, is a powerful, low-cost run-time development tool. This tool is based on the flash PIC16F877 and can be used to develop for this and other PIC microcontrollers from the PIC16CXXX family. MPLAB-ICD utilizes the In-Circuit Debugging capability built into the PIC16F87X. This feature, along with Microchip's In-Circuit Serial Programming protocol, offers cost-effective in-circuit flash programming and debugging from the graphical user interface of the MPLAB Integrated Development Environment. This enables a designer to develop and debug source code by watching variables, single-stepping and setting break points. Running at full speed enables testing hardware in real-time. The MPLAB-ICD is also a programmer for the flash PIC16F87X family.

11.10 PRO MATE II Universal Programmer

The PRO MATE II Universal Programmer is a full-featured programmer capable of operating in stand-alone mode as well as PC-hosted mode. PRO MATE II is CE compliant.

The PRO MATE II has programmable VDD and VPP supplies which allows it to verify programmed memory at VDD min and VDD max for maximum reliability. It has an LCD display for instructions and error messages, keys to enter commands and a modular detachable socket assembly to support various package types. In

PIC12C67X

12.6 Timing Diagrams and Specifications

FIGURE 12-5: EXTERNAL CLOCK TIMING

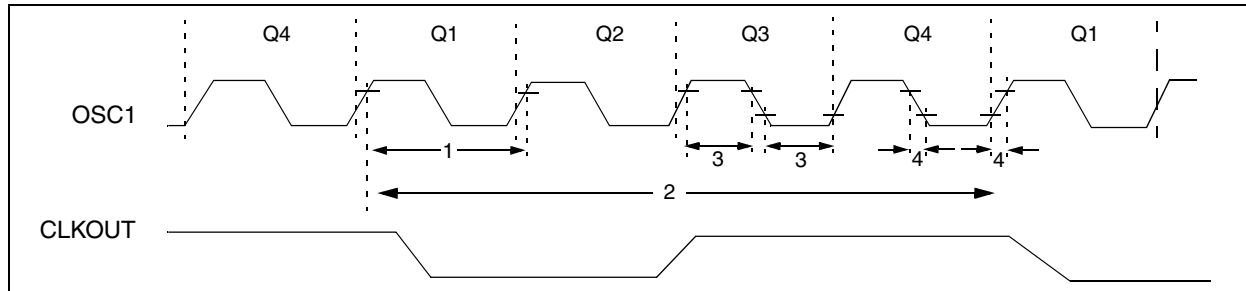


TABLE 12-1: CLOCK TIMING REQUIREMENTS

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
	FOSC	External CLKIN Frequency (Note 1)	DC	—	4	MHz	XT and EXTRC osc mode
			DC	—	4	MHz	HS osc mode (PIC12CE67X-04)
			DC	—	10	MHz	HS osc mode (PIC12CE67X-10)
			DC	—	200	kHz	LP osc mode
		Oscillator Frequency (Note 1)	DC	—	4	MHz	EXTRC osc mode
			.455	—	4	MHz	XT osc mode
			4	—	4	MHz	HS osc mode (PIC12CE67X-04)
			4	—	10	MHz	HS osc mode (PIC12CE67X-10)
1	TOSC	External CLKIN Period (Note 1)	250	—	—	ns	XT and EXTRC osc mode
			250	—	—	ns	HS osc mode (PIC12CE67X-04)
			100	—	—	ns	HS osc mode (PIC12CE67X-10)
			5	—	—	μs	LP osc mode
		Oscillator Period (Note 1)	250	—	—	ns	EXTRC osc mode
			250	—	10,000	ns	XT osc mode
			250	—	250	ns	HS osc mode (PIC12CE67X-04)
			100	—	250	ns	HS osc mode (PIC12CE67X-10)
2	TCY	Instruction Cycle Time (Note 1)	400	—	DC	ns	LP osc mode
			—	—	—	—	—
			—	—	—	—	—
			—	—	—	—	—
3	TosL, TosH	External Clock in (OSC1) High or Low Time	50	—	—	ns	XT oscillator
			2.5	—	—	μs	LP oscillator
			10	—	—	ns	HS oscillator
4	TosR, TosF	External Clock in (OSC1) Rise or Fall Time	—	—	25	ns	XT oscillator
			—	—	50	ns	LP oscillator
			—	—	15	ns	HS oscillator

† Data in "Typ" column is at 5V, 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/CLKIN pin.

When an external clock input is used, the "Max." cycle time limit is "DC" (no clock) for all devices. OSC2 is disconnected (has no loading) for the PIC12C67X.

PIC12C67X

NOTES:

PIC12C67X

I

I/O Interfacing	25
I/O Ports	25
I/O Programming Considerations	31
ID Locations	53
INCF Instruction	76
INCFSZ Instruction	76
In-Circuit Serial Programming	53, 67
INDF Register	14, 23
Indirect Addressing	23
Initialization Conditions for All Registers	59
Instruction Cycle	10
Instruction Flow/Pipelining	10
Instruction Format	69
Instruction Set	
ADDLW	72
ADDWF	72
ANDLW	72
ANDWF	72
BCF	73
BSF	73
BTFSC	73
BTFSS	74
CALL	74
CLRF	74
CLRW	74
CLRWDI	75
COMF	75
DECF	75
DECFSZ	75
GOTO	76
INCF	76
INCFSZ	76
IORLW	76
IORWF	77
MOVF	77
MOVLW	77
MOVWF	77
NOP	78
OPTION	78
RETFIE	78
RETLW	78
RETURN	79
RLF	79
RRF	79
SLEEP	79
SUBLW	80
SUBWF	80
SWAPF	81
TRIS	81
XORLW	81
XORWF	81
Section	69
INTCON Register	17
INTEDG bit	16
Internal Sampling Switch (Rss) Impedance	48
Interrupts	53
A/D	62
GP2/INT	62
GPIO Port	62
Section	62
TMR0	64
TMR0 Overflow	62
IORLW Instruction	76
IORWF Instruction	77
IRP bit	15

K

KeeLoq® Evaluation and Programming Tools	86
--	----

L

Loading of PC	22
---------------------	----

M

MCLR	56, 59
Memory	
Data Memory	11
Program Memory	11
Register File Map - PIC12CE67X	12
MOVF Instruction	77
MOVLW Instruction	77
MOVWF Instruction	77
MPLAB Integrated Development Environment Software	83

N

NOP Instruction	78
-----------------------	----

O

Opcode	69
OPTION Instruction	78
OPTION Register	16
Orthogonal	7
OSC selection	53
OSCCAL Register	21
Oscillator	
EXTRC	58
HS	58
INTRC	58
LP	58
XT	58
Oscillator Configurations	54
Oscillator Types	
EXTRC	54
HS	54
INTRC	54
LP	54
XT	54

P

Package Marking Information	115
Packaging Information	115
Paging, Program Memory	22
PCL	70
PCL Register	13, 14, 22
PCLATH	59
PCLATH Register	13, 14, 22
PCON Register	20, 58
PD bit	15, 56
PICDEM-1 Low-Cost PIC MCU Demo Board	85
PICDEM-2 Low-Cost PIC16CXX Demo Board	85
PICDEM-3 Low-Cost PIC16CXXX Demo Board	85
PICSTART® Plus Entry Level Development System	85
PIE1 Register	18
Pinout Description - PIC12CE67X	9
PIR1 Register	19
POP	22
POR	58
Oscillator Start-up Timer (OST)	53, 58
Power Control Register (PCON)	58
Power-on Reset (POR)	53, 58, 59
Power-up Timer (PWRT)	53, 58
Power-Up-Timer (PWRT)	58
Time-out Sequence	58
Time-out Sequence on Power-up	60
TO	56
Power	56

THE MICROCHIP WEB SITE

Microchip provides online support via our WWW site at www.microchip.com. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at www.microchip.com. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or field application engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at: <http://microchip.com/support>