



**MICROCHIP**

# PIC12C67X

## 8-Pin, 8-Bit CMOS Microcontroller with A/D Converter

### Devices included in this Data Sheet:

PIC12C671 and PIC12C672 are 8-bit microcontrollers with 8-bit A/D Converter packaged in 8-lead packages. They are based on the 14-bit PIC16/17 architecture.

### High-Performance RISC CPU:

- Only 35 single word instructions to learn
- All instructions are single cycle (1  $\mu$ s) except for program branches which are two-cycle
- Operating speed: DC - 10 MHz clock input  
DC - 1  $\mu$ s instruction cycle

Device	EPROM	RAM
PIC12C671	1024 x 14	128 x 8
PIC12C672	2048 x 14	128 x 8

- 14-bit wide instructions
- 8-bit wide data path
- Interrupt capability
- Special function hardware registers
- Eight-level deep hardware stack
- Direct, indirect and relative addressing modes for data and instructions
- Internal 4 MHz oscillator with programmable calibration
- Selectable clockout
- In-circuit serial programming
- 4-channel 8-bit analog-to-digital converter

### Peripheral Features:

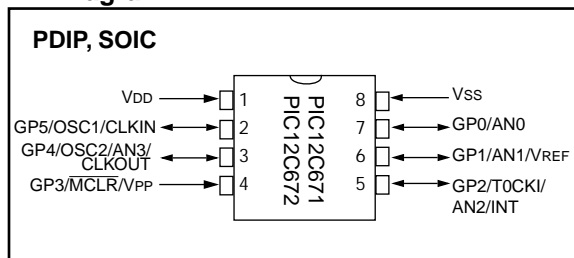
- 8-bit real time clock/counter (TMR0) with 8-bit programmable prescaler
- Power-On Reset (POR)
- Power-up Timer (PWRT) and Oscillator Start-up Timer (OSC)
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Programmable code-protection
- Power saving SLEEP mode
- Interrupt on pin change (GP0, GP1, GP3)
- Internal pull-ups on I/O pins (GP0, GP1, GP3)

- Selectable oscillator options:
  - INTRC: Precision internal 4 MHz oscillator
  - EXTRC: External low-cost RC oscillator
  - XT: Standard crystal/resonator
  - HS: High speed crystal/resonator
  - LP: Power saving, low frequency crystal
- Internal pull-up on MCLR pin

### CMOS Technology:

- Low power, high speed CMOS EPROM technology
- Fully static design
- Wide operating voltage range:
  - Commercial: 2.5V to 5.5V
  - Industrial: 2.5V to 5.5V
  - Extended: 4.5V to 5.5V
- Low power consumption
  - < 2 mA @ 5V, 4 MHz
  - 15  $\mu$ A typical @ 3V, 32 KHz
  - < 1  $\mu$ A typical standby current

### Pin Diagram



# PIC12C67X

---

---

## Table of Contents

1.0	General Description.....	3
2.0	PIC12C67X Device Varieties .....	5
3.0	Architectural Overview .....	7
4.0	Memory Organization .....	11
5.0	I/O Ports .....	23
6.0	Timer0 Module .....	25
7.0	Analog-to-Digital Converter (A/D) Module.....	31
8.0	Special Features of the CPU.....	39
9.0	Instruction Set Summary .....	55
10.0	Development Support.....	69
11.0	Electrical Characteristics for PIC12C67X .....	73
12.0	DC and AC Characteristics - PIC12C67X .....	89
13.0	Packaging Information.....	93
Appendix A: Compatibility .....		97
Index.....		101
PIC12C67X Product Identification System.....		109

## To Our Valued Customers

We constantly strive to improve the quality of all our products and documentation. We have spent an exceptional amount of time to ensure that these documents are correct. However, we realize that we may have missed a few things. If you find any information that is missing or appears in error, please use the reader response form in the back of this data sheet to inform us. We appreciate your assistance in making this a better document.

## 1.0 GENERAL DESCRIPTION

The PIC12C67X device is a low-cost, high-performance, CMOS, fully-static, 8-bit microcontroller with integrated analog-to-digital (A/D) converter, in the PIC12CXXX Microcontroller family.

All PIC16/17 microcontrollers employ an advanced RISC architecture. The PIC12C67X microcontrollers 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.

PIC12C67X microcontrollers typically achieve a 2:1 code compression and a 4:1 speed improvement over other 8-bit microcontrollers in their class.

The PIC12C67X devices have 128 bytes of RAM and 6 I/O pins. In addition a timer/counter is available. Also a 4-channel high-speed 8-bit A/D is provided. The 8-bit resolution is ideally suited for applications requiring low-cost analog interface, e.g. thermostat control, pressure sensing, etc.

The PIC12C67X device has special features to reduce external components, thus reducing cost, enhancing system reliability and reducing power consumption. The PIC12C67X products are equipped with special features that reduce system cost and power requirements. The Power-On Reset (POR), Power-up Timer (PWRT), and Oscillator Start-up Timer (OST) eliminate the need for external reset circuitry. There are five oscillator configurations to choose from, including INTRC precision internal oscillator mode and the power-saving LP (Low Power) oscillator. Power saving SLEEP mode, Watchdog Timer and code protection features improve system cost, power and reliability. The SLEEP (power-down) feature provides a power saving mode. The user can wake up the chip from SLEEP through several external and internal interrupts and resets.

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

A UV erasable windowed package version is ideal for code development while the cost-effective One-Time-Programmable (OTP) version is suitable for production in any volume. The customer can take full advantage of Microchip's price leadership in OTP microcontrollers while benefiting from the OTP's flexibility.

The PIC12C67X device fits perfectly in applications ranging from security and remote sensors to appliance control and automotive. The EPROM technology makes customization of application programs (transmitter codes, motor speeds, receiver frequencies, 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 PIC12C67X very versatile even in areas where no microcontroller use has been considered before (e.g. timer functions, communications and coprocessor applications).

### 1.1 Family and Upward Compatibility

The PIC12C67X products are compatible with other members of the PIC16CXXX family.

### 1.2 Development Support

The PIC12C67X device is supported by a full-featured macro assembler, a software simulator, an in-circuit emulator, a low-cost development programmer and a full-featured programmer. A "C" compiler and fuzzy logic support tools are also available.

# PIC12C67X

TABLE 1-1: PIC12CXXX FAMILY OF DEVICES

		PIC12C508	PIC12C509	PIC12C671	PIC12C672
<b>Clock</b>	Maximum Frequency of Operation (MHz)	4	4	10	10
<b>Memory</b>	EPROM Program Memory	512 x 12	1024 x 12	1024 x 14	2048 x 14
	Data Memory (bytes)	25	41	128	128
<b>Peripherals</b>	Timer Module(s)	TMR0	TMR0	TMR0	TMR0
	A/D Converter (8-bit) Channels	—	—	4	4
<b>Features</b>	Wake-up from SLEEP on pin change	Yes	Yes	Yes	Yes
	Interrupt Sources	—	—	4	4
	I/O Pins	5	5	5	5
	Input Pins	1	1	1	1
	Internal Pull-ups	Yes	Yes	Yes	Yes
	Voltage Range (Volts)	2.5-5.5	2.5-5.5	2.5-5.5	2.5-5.5
	In-Circuit Serial Programming	Yes	Yes	Yes	Yes
	Number of Instructions	33	33	35	35
	Packages	8-pin DIP, JW, SOIC	8-pin DIP, JW, SOIC	8-pin DIP, JW, SOIC	8-pin DIP, JW, SOIC

All PIC12CXXX devices have Power-on Reset, selectable Watchdog Timer, selectable code protect and high I/O current capability. All PIC12CXXX devices use serial programming with data pin GP0 and clock pin GP1.

## 2.0 PIC12C67X DEVICE VARIETIES

A variety of frequency ranges and packaging options are available. Depending on application and production requirements, the proper device option can be selected using the information in the PIC12C67X Product Identification System section at the end of this data sheet. When placing orders, please use that page of the data sheet to specify the correct part number.

For the PIC12C67X, there are two device "types" as indicated in the device number:

1. **C**, as in PIC12**C**671. These devices have EPROM type memory and operate over the standard voltage range.
2. **LC**, as in PIC12**LC**671. These devices have EPROM type memory and operate over an extended voltage range.

### 2.1 UV Erasable Devices

The UV erasable version, offered in windowed package, is optimal for prototype development and pilot programs.

The UV erasable version can be erased and reprogrammed to any of the configuration modes. Microchip's PICSTART<sup>®</sup> Plus and PRO MATE<sup>®</sup> programmers both support the PIC12C67X. Third party programmers also are available; refer to the Microchip Third Party Guide for a list of sources.

**CAUTION: Calibration values must be read before erasing.**

### 2.2 One-Time-Programmable (OTP) Devices

The availability of OTP devices is especially useful for customers who need the flexibility for frequent code updates and small volume applications.

The OTP devices, packaged in plastic packages, permit the user to program them once. In addition to the program memory, the configuration bits must also be programmed.

### 2.3 Quick-Turnaround-Production (QTP) Devices

Microchip offers a QTP Programming Service for factory production orders. This service is made available for users who choose not to program a medium to high quantity of units and whose code patterns have stabilized. The devices are identical to the OTP devices but with all EPROM locations and configuration options already programmed by the factory. Certain code and prototype verification procedures apply before production shipments are available. Please contact your local Microchip Technology sales office for more details.

### 2.4 Serialized Quick-Turnaround Production (SQTP<sup>SM</sup>) Devices

Microchip offers a unique programming service where a few user-defined locations in each device are programmed with different serial numbers. The serial numbers may be random, pseudo-random, or sequential.

Serial programming allows each device to have a unique number which can serve as an entry-code, password, or ID number.

# PIC12C67X

---

NOTES:

## 3.0 ARCHITECTURAL OVERVIEW

The high performance of the PIC12C67X family can be attributed to a number of architectural features commonly found in RISC microprocessors. To begin with, the PIC12C67X uses a Harvard architecture, in which program and data are accessed from separate memories using separate buses. This improves bandwidth over traditional von Neumann architecture in which program and data are fetched from the same memory using the same bus. Separating program and data buses also allow instructions to be sized differently than the 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 (Example 3-1). Consequently, all instructions (35) execute in a single cycle (1  $\mu$ s @ 4 MHz) except for program branches.

The table below lists program memory (EPROM) and data memory (RAM) for each PIC12C67X device.

Device	Program Memory	Data Memory
PIC12C671	1K x 14	128 x 8
PIC12C672	2K x 14	128 x 8

The PIC12C67X 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 PIC12C67X 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 PIC12C67X simple yet efficient. In addition, the learning curve is reduced significantly.

PIC12C67X devices contain an 8-bit ALU and working register. The ALU is a general purpose arithmetic unit. It performs arithmetic and Boolean functions between the data in the working register and any register file.

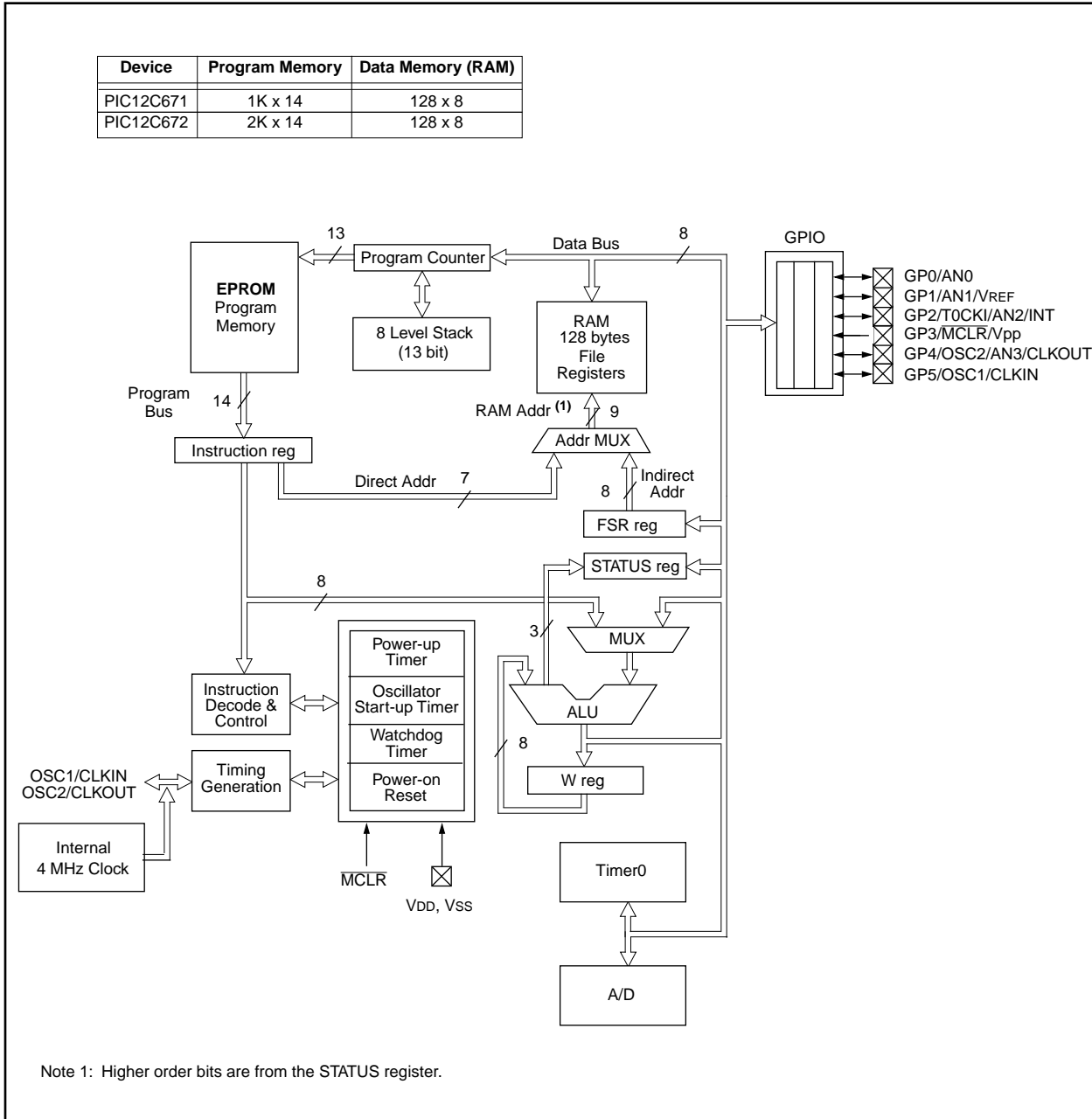
The ALU is 8-bits wide and capable of addition, subtraction, shift and logical operations. Unless otherwise mentioned, arithmetic operations are two's complement in nature. In two-operand instructions, typically one operand is the working register (W register). The other operand is a file register or 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 bit and a digit borrow out bit, respectively, in subtraction. See the `SUBLW` and `SUBWF` instructions for examples.

# PIC12C67X

FIGURE 3-1: PIC12C67X BLOCK DIAGRAM





**TABLE 3-1: PIC12C67X PINOUT DESCRIPTION**

<b>Name</b>	<b>DIP Pin #</b>	<b>SOIC Pin #</b>	<b>I/O/P Type</b>	<b>Buffer Type</b>	<b>Description</b>
GP0/AN0	7	7	I/O	TTL/ST	Bi-directional I/O port/serial programming data/analog input 0. Can be software programmed for internal weak pull-up and interrupt on pin change. This buffer is a Schmitt Trigger input when used in serial program-

# PIC12C67X

## 3.1 Clocking Scheme/Instruction Cycle

The clock input (from OSC1) 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.

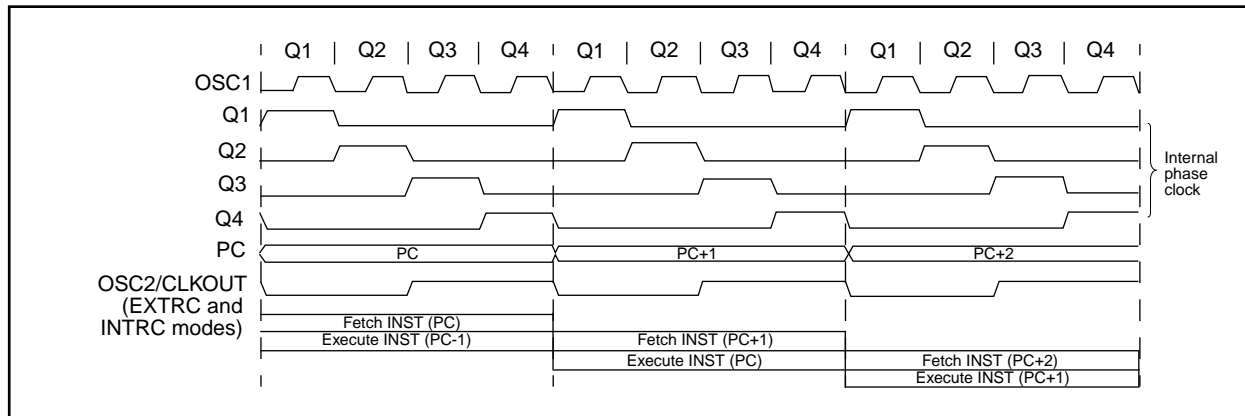
## 3.2 Instruction Flow/Pipelining

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).

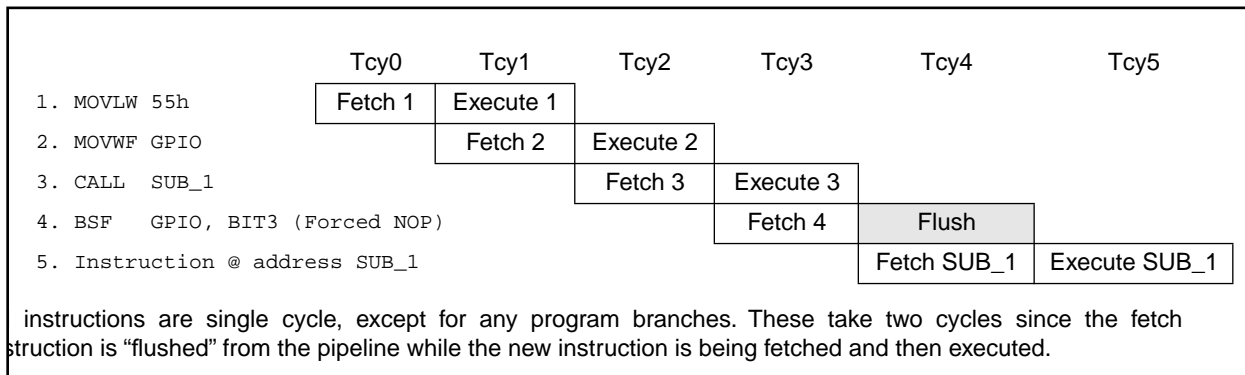
A fetch cycle begins with the program counter (PC) incrementing in Q1.

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**



## 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 the first 1K x 14 (0000h-03FFh) is implemented.

For the PIC12C672, the first 2K x 14 (0000h-07FFh) is implemented. Accessing a location above

# PIC12C67X

**FIGURE 4-2: PIC12C67X REGISTER FILE MAP**

File Address			File Address
00h	INDF <sup>(1)</sup>	INDF <sup>(1)</sup>	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			F0h
7Fh			Mapped in Bank 0
	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.

**TABLE 4-1: PIC12C67X SPECIAL FUNCTION REGISTER SUMMARY**

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 <sup>(3)</sup>
<b>Bank 0</b>											
00h <sup>(1)</sup>	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	0000 0000
01h	TMR0	Timer0 module's register								xxxx xxxx	uuuu uuuu
02h <sup>(1)</sup>	PCL	Program Counter's (PC) Least Significant Byte								0000 0000	0000 0000
03h <sup>(1)</sup>	STATUS	IRP <sup>(4)</sup>	RP1 <sup>(4)</sup>	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	000q quuu
04h <sup>(1)</sup>	FSR	Indirect data memory address pointer								xxxx xxxx	uuuu uuuu
05h	GPIO	—	—	GP5	GP4	GP3	GP2	GP1	GP0	--xx xxxx	--uu uuuu
06h	—	Unimplemented								—	—
07h	—	Unimplemented								—	—
08h	—	Unimplemented								—	—
09h	—	Unimplemented								—	—
0Ah <sup>(1,2)</sup>	PCLATH	—	—	—	Write Buffer for the upper 5 bits of the Program Counter					---0 0000	---0 0000
0Bh <sup>(1)</sup>	INTCON	GIE	PEIE	TOIE	INTE	GPIE	TOIF	INTF	GPIF	0000 000x	0000 000u
0Ch	PIR1	—	ADIF	—	—	—	—	—	—	-0-- ----	-0-- ----
0Dh	—	Unimplemented								—	—
0Eh	—	Unimplemented								—	—
0Fh	—	Unimplemented								—	—
10h	—	Unimplemented								—	—
11h	—	Unimplemented								—	—
12h	—	Unimplemented								—	—
13h	—	Unimplemented								—	—
14h	—	Unimplemented								—	—
15h	—	Unimplemented								—	—
16h	—	Unimplemented								—	—
17h	—	Unimplemented								—	—
18h	—	Unimplemented								—	—
19h	—	Unimplemented								—	—
1Ah	—	Unimplemented								—	—
1Bh	—	Unimplemented								—	—
1Ch	—	Unimplemented								—	—
1Dh	—	Unimplemented								—	—
1Eh	ADRES	A/D Result Register								xxxx xxxx	uuuu uuuu
1Fh	ADCON0	ADCS1	ADCS0	r	CHS1	CHS0	GO/DONE	r	ADON	0000 0000	0000 0000

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

# 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 <sup>(3)</sup>
<b>Bank 1</b>											
80h <sup>(1)</sup>	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 <sup>(1)</sup>	PCL	Program Counter's (PC) Least Significant Byte								0000 0000	0000 0000
83h <sup>(1)</sup>	STATUS	IRP <sup>(4)</sup>	RP1 <sup>(4)</sup>	RP0	T0	PD	Z	DC	C	0001 1xxx	000q quuu
84h <sup>(1)</sup>	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 <sup>(1,2)</sup>	PCLATH	—	—	—	Write Buffer for the upper 5 bits of the PC					---0 0000	---0 0000
8Bh <sup>(1)</sup>	INTCON	GIE	PEIE	T0IE	INTE	GPIE	TOIF	INTF	GPIF	0000 000x	0000 000x
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', r = reserved.  
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.

## 4.2.2.1 STATUS REGISTER

The STATUS register, shown in Figure 4-3, 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, as with 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  $\overline{\text{TO}}$  and  $\overline{\text{PD}}$  bits are not writable. Therefore, the result of an instruction with the STATUS register as destination may be different than intended.

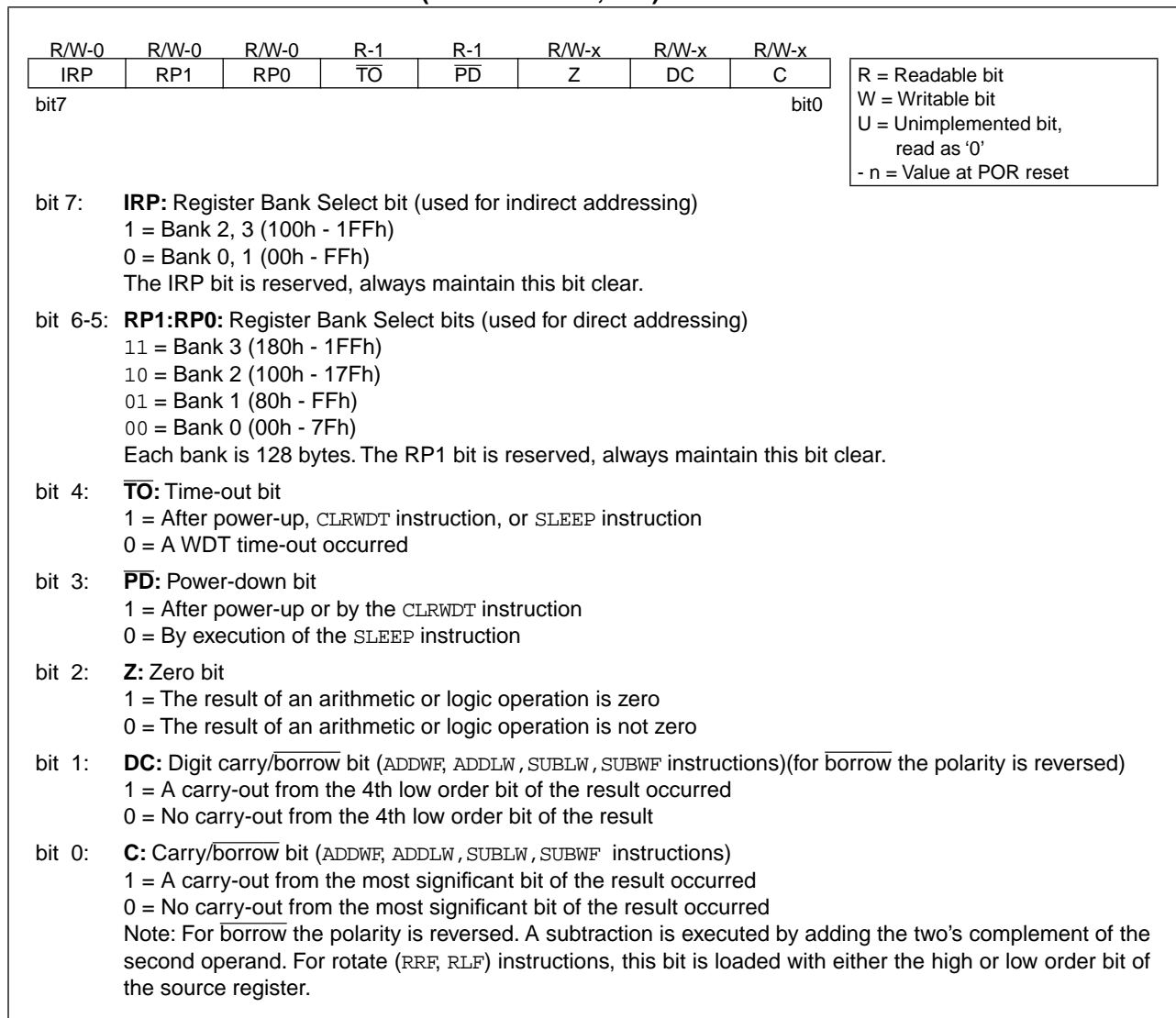
For example, `CLRF STATUS` will clear the upper-three bits and set the Z bit. This leaves the STATUS register as `000u u1uu` (where u = unchanged).

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 the Z, C or DC bits from the STATUS register. For other instructions, not affecting any status bits, see the "Instruction Set Summary."

**Note 1:** Bits IRP and RP1 (STATUS<7:6>) are not used by the PIC12C67X and should be maintained clear. Use of these bits as general purpose R/W bits is NOT recommended, since this may affect upward compatibility with future products.

**Note 2:** The C and DC bits operate as a borrow and digit borrow bit, respectively, in subtraction. See the `SUBLW` and `SUBWF` instructions for examples.

**FIGURE 4-3: STATUS REGISTER (ADDRESS 03h, 83h)**



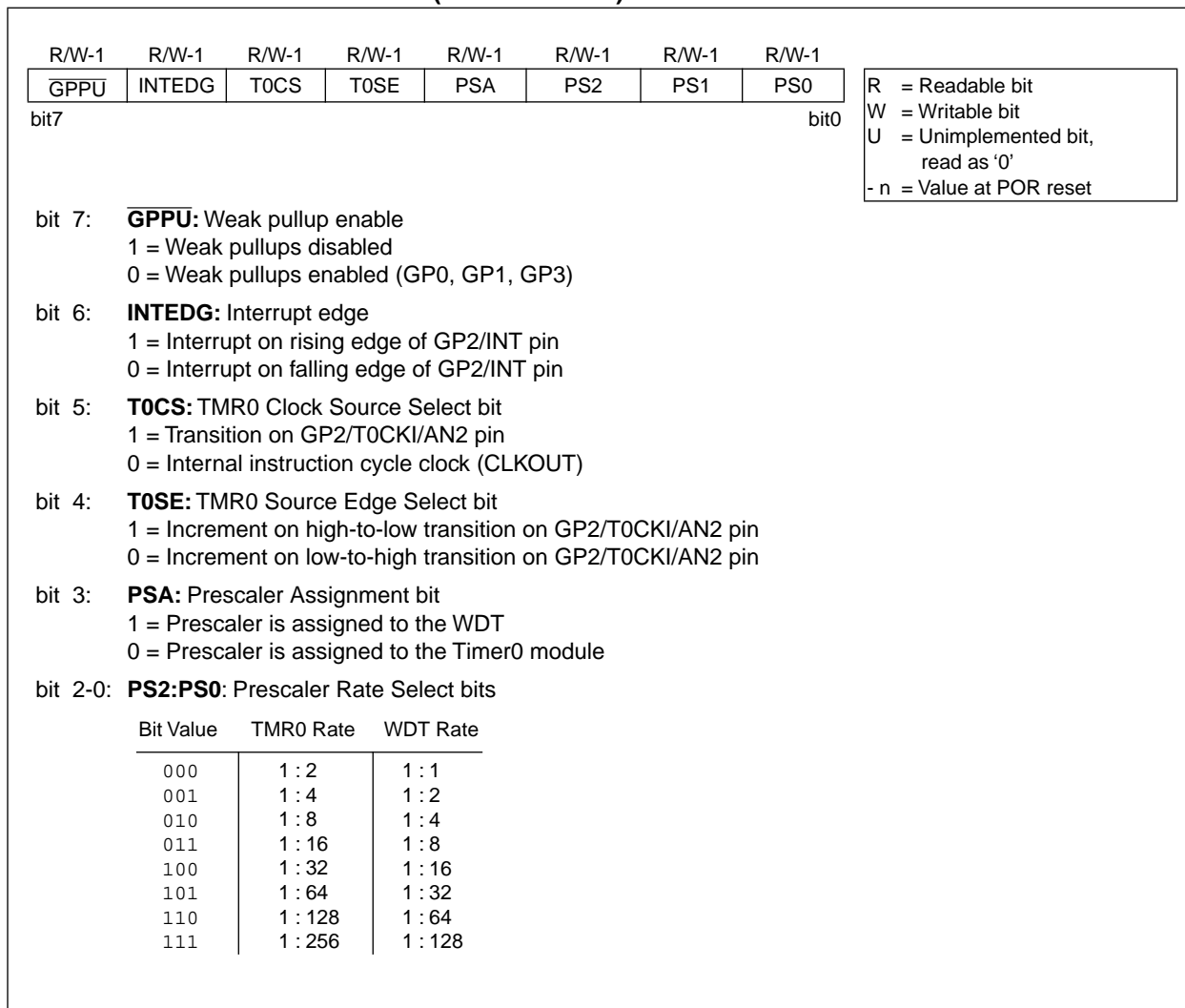
# PIC12C67X

## 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>).

**FIGURE 4-4: OPTION REGISTER (ADDRESS 81h)**



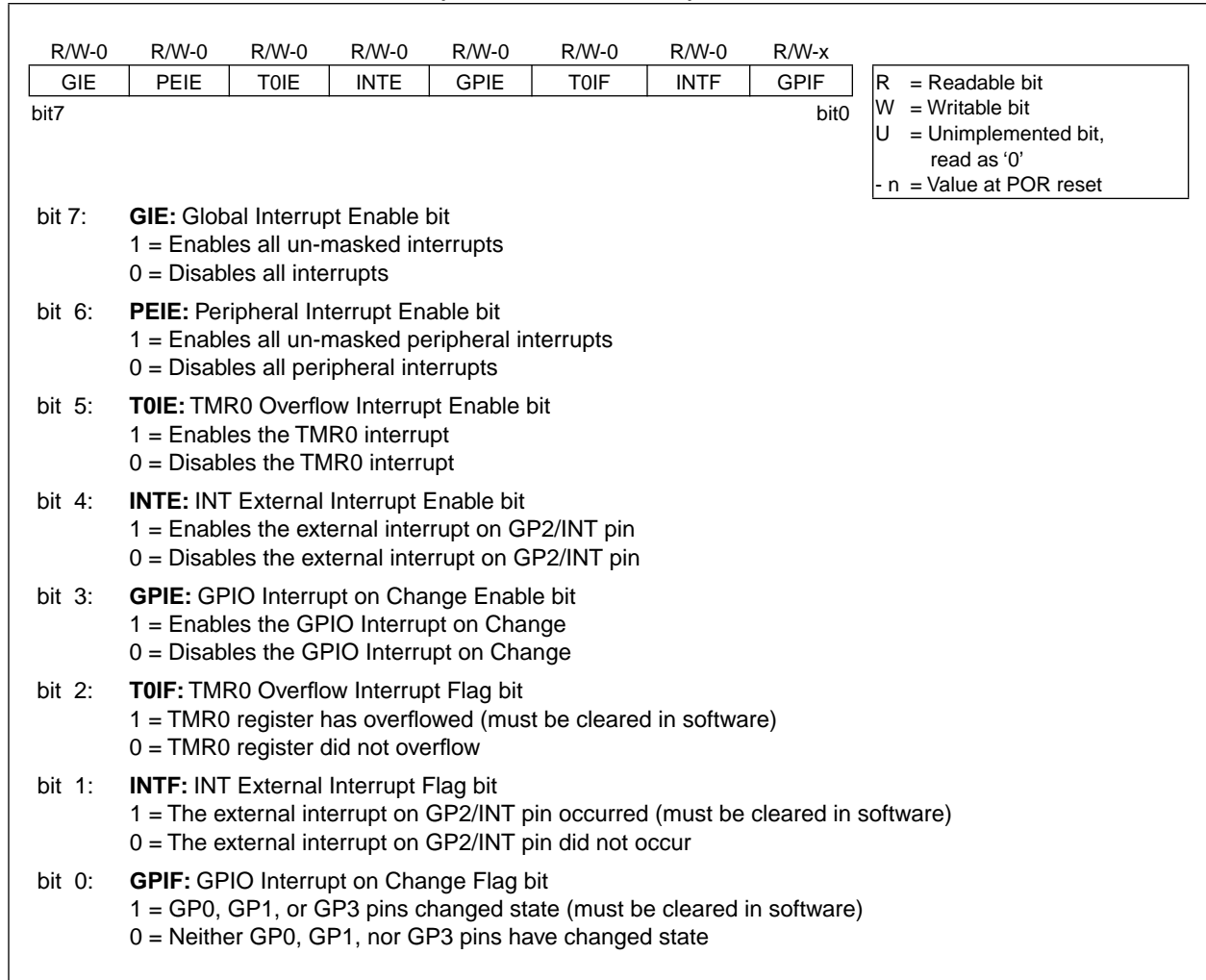


## 4.2.2.3 INTCON REGISTER

The INTCON Register is a readable and writable register which contains various enable and flag bits for the TMR0 register overflow, GPIO Port change and External GP2/INT Pin interrupts.

**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>).

**FIGURE 4-5: INTCON REGISTER (ADDRESS 0Bh, 8Bh)**



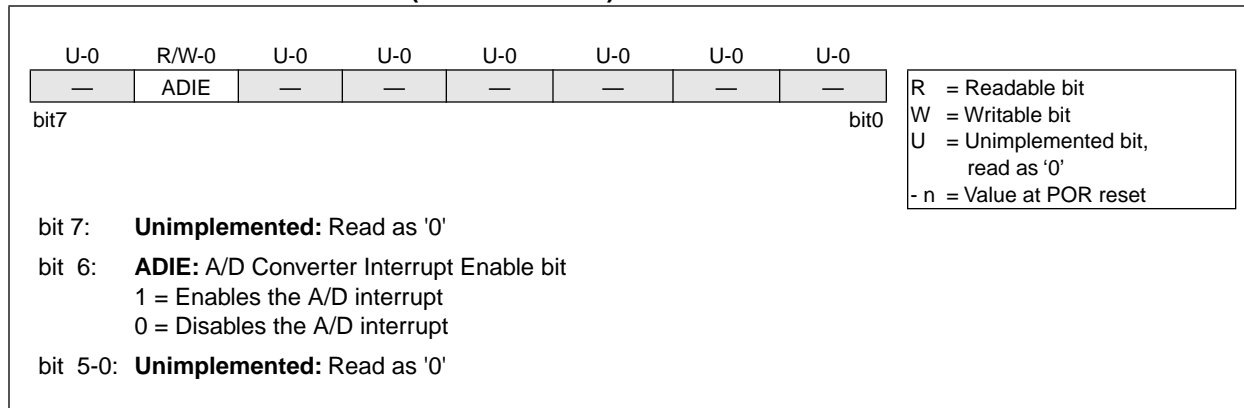
# PIC12C67X

## 4.2.2.4 PIE1 REGISTER

This register contains the individual enable bits for the Peripheral interrupts.

**Note:** Bit PEIE (INTCON<6>) must be set to enable any peripheral interrupt.

**FIGURE 4-6: PIE1 REGISTER (ADDRESS 8Ch)**

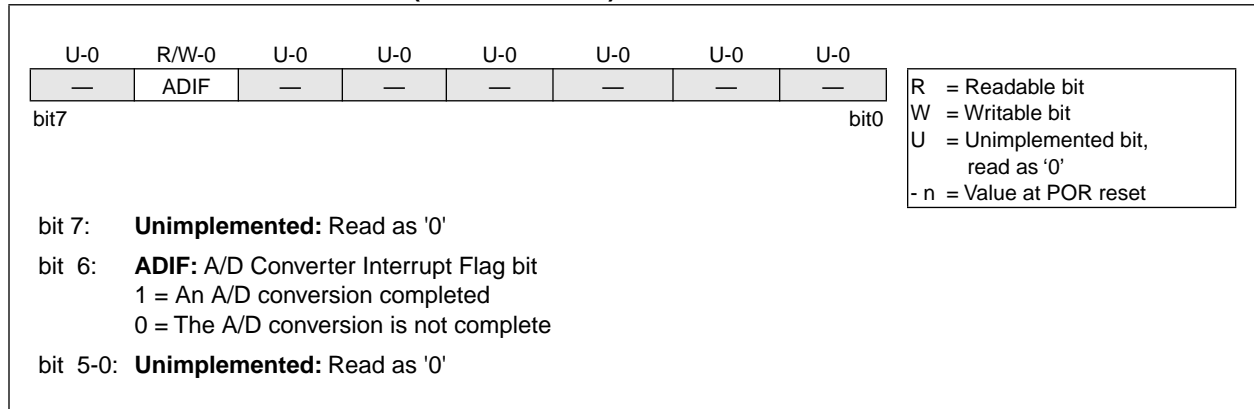


## 4.2.2.5 PIR1 REGISTER

This register contains the individual flag bits for the Peripheral interrupts.

**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>). User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

**FIGURE 4-7: PIR1 REGISTER (ADDRESS 0Ch)**

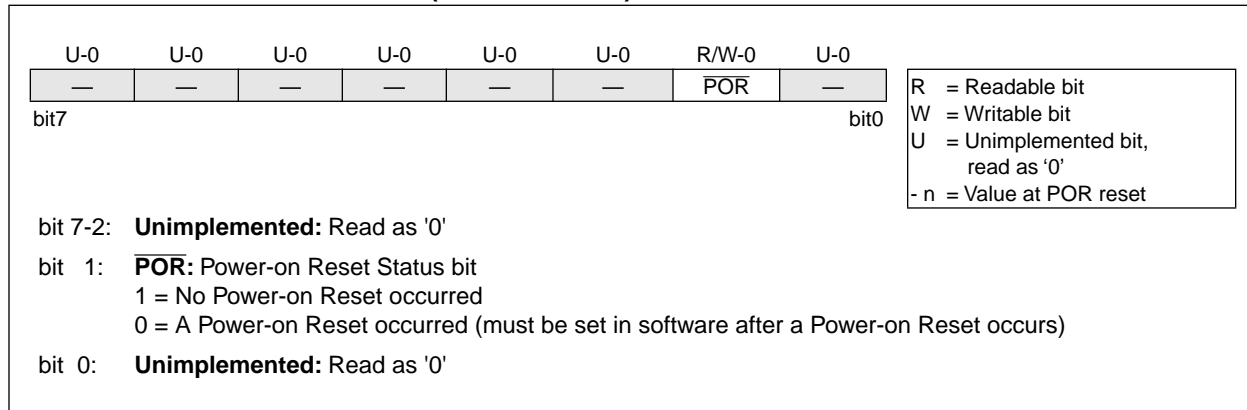


# PIC12C67X

## 4.2.2.6 PCON REGISTER

The Power Control (PCON) register contains a flag bit to allow differentiation between a Power-on Reset (POR), an external  $\overline{\text{MCLR}}$  Reset, and WDT Reset.

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



## 4.2.2.7 OSCCAL REGISTER

The Oscillator Calibration (OSCCAL) register is used to calibrate the internal 4 MHz oscillator. It contains four bits for fine calibration and two other bits to either increase or decrease frequency.

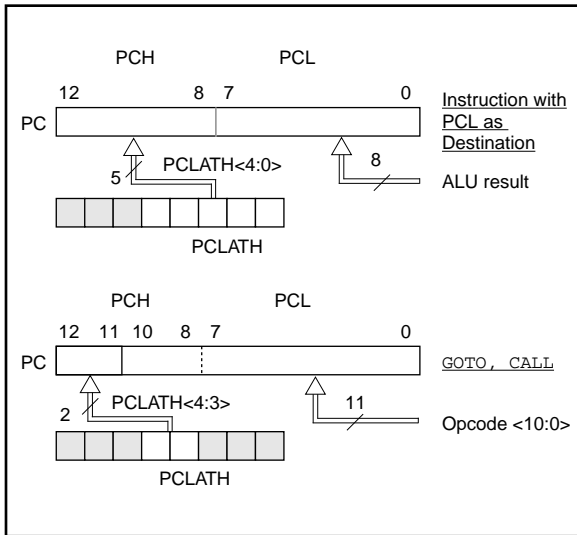
**FIGURE 4-9: OSCCAL REGISTER (ADDRESS 8FU0**

# PIC12C67X

## 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 4-10 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 4-10: 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 PIC12C67X family has an 8 level deep x 13-bit wide hardware stack. 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.

**Note 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 Program Memory Paging

The PIC12C67X ignores both paging bits PCLATH<4:3>, which are used to access program memory when more than one page is available. The use of PCLATH<4:3> as general purpose read/write bits for the PIC12C67X is not recommended since this may affect upward compatibility with future products.

## 4.5 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 the register pointed to by the File Select Register, FSR. Reading the INDF register itself indirectly (FSR = '0') will read 00h. Writing to the INDF register indirectly results in a no-operation (although status bits may be affected). An effective 9-bit address is obtained by concatenating the 8-bit FSR register and the IRP bit (STATUS<7>), as shown in Figure 4-11. However, IRP is not used in the PIC12C67X.

# PIC12C67X

---

NOTES:



## 5.0 I/O PORT

As with any other register, the I/O register can be written and read under program control. However, read instructions (e.g., `MOVF GPIO, W`) always read the I/O pins independent of the pin's input/output modes. On RESET, all I/O ports are defined as input (inputs are at hi-impedance) since the I/O control registers are all set.

### 5.1 GPIO

GPIO is an 8-bit I/O register. Only the low order 6 bits are used (GP5:GP0). Bits 7 and 6 are unimplemented and read as '0's. Please note that GP3 is an input only pin. The configuration word can set several I/O's to alternate functions. When acting as alternate functions the pins will read as '0' during port read. Pins GP0, GP1, and GP3 can be configured with weak pull-ups and also with interrupt on change. The interrupt on change and weak pull-up functions are not pin selectable. If pin 4 is configured as  $\overline{MCLR}$ , the weak pull-up is always on. Interrupt on change for this pin is not set and GP3 will read as '0'. Interrupt on change is enabled by setting `INTCON<3>`. Note that external oscillator use overrides the GPIO functions on GP4 and GP5.

### 5.2 TRIS Register

This register controls the data direction for GPIO. A '1' from a TRIS register bit puts the corresponding output driver in a hi-impedance mode. A '0' puts the contents of the output data latch on the selected pins, enabling the output buffer. The exceptions are GP3 which is input only and its TRIS bit will always read as '1'.

**Note:** A read of the ports reads the pins, not the output data latches. That is, if an output driver on a pin is enabled and driven high, but the external system is holding it low, a read of the port will indicate that the pin is low.

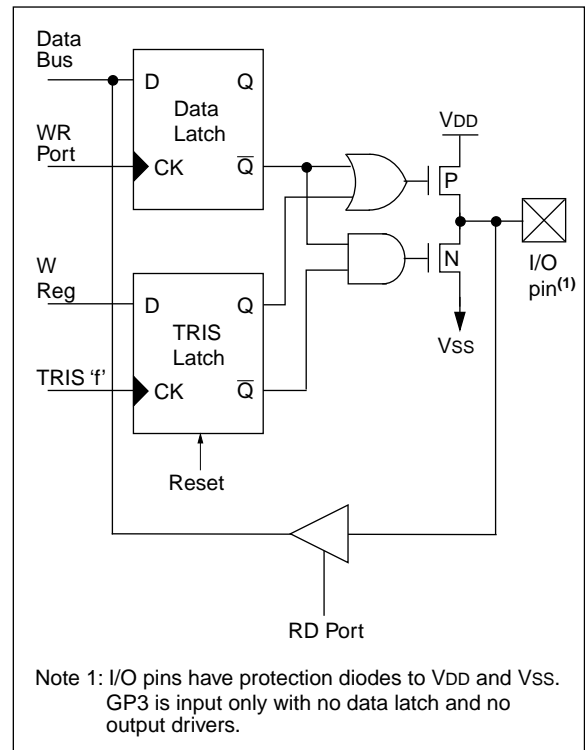
Upon reset, the TRIS register is all '1's, making all pins inputs.

### 5.3 I/O Interfacing

The equivalent circuit for an I/O port pin is shown in Figure 5-2. All port pins, except GP3 which is input only, may be used for both input and output operations. For input operations these ports are non-latching. Any input must be present until read by an input instruction (e.g., `MOVF GPIO, W`). The outputs are latched and remain unchanged until the output latch is rewritten. To use a port pin as output, the corresponding direction control bit in TRIS must be cleared (= 0). For use as an input, the corresponding TRIS bit must be set. Any I/O pin (except GP3) can be programmed individually as input or output.

**Note:** On a Power-on Reset, GP0, GP1, GP2, GP4 are configured as analog inputs and read as '0'.

**FIGURE 5-1: EQUIVALENT CIRCUIT FOR A SINGLE I/O PIN**



**TABLE 5-1: SUMMARY OF PORT REGISTERS**

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
85h											

# PIC12C67X

## 5.4 I/O Programming Considerations

### 5.4.1 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 GPIO will cause all eight bits of GPIO to be read into the CPU. Then the `BSF` operation takes place on bit5 and GPIO is written to the output latches. If another bit of GPIO is used as a bi-directional 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 rewritten 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 to an output, 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 on an I/O port.

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

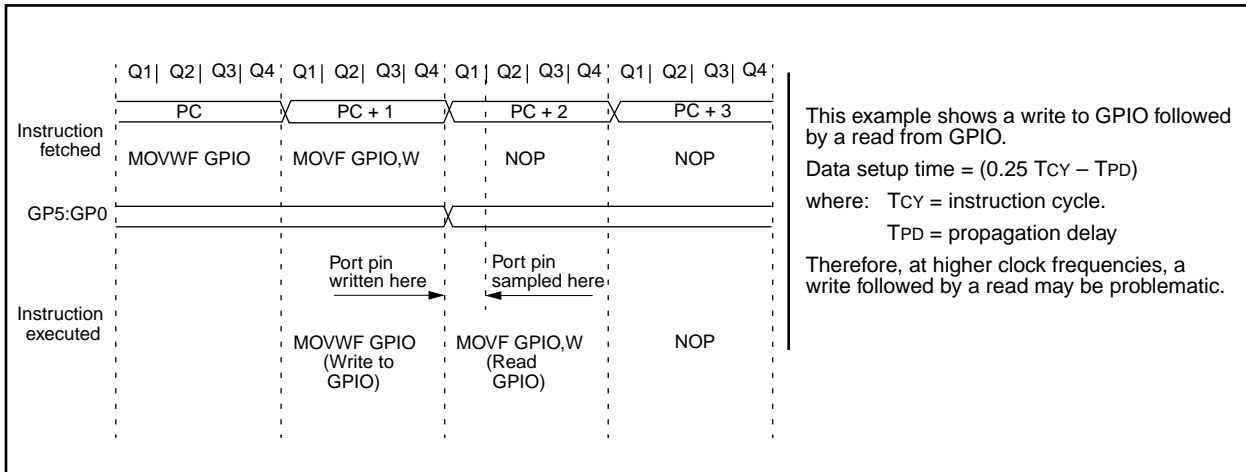
```

;Initial GPIO Settings
; GPIO<5:3> Inputs
; GPIO<2:0> Outputs
;
;
;           GPIO latch  GPIO pins
;           -----  -----
BCF  GPIO, 5  ;--01 -ppp  --11 pppp
BCF  GPIO, 4  ;--10 -ppp  --11 pppp
MOVLW 007h   ;
TRIS GPIO    ;--10 -ppp  --11 pppp
;
;Note that the user may have expected the pin
;values to be --00 pppp. The 2nd BCF caused
;GP5 to be latched as the pin value (High).

```

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.

FIGURE 5-2: SUCCESSIVE I/O OPERATION



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

Counter mode is selected by setting bit T0CS (OPTION<5>). In counter mode, Timer0 will increment either on every rising or falling edge of pin RA4/T0CKI. The incrementing edge is determined by the Timer0 Source Edge Select bit T0SE (OPTION<4>). Clearing

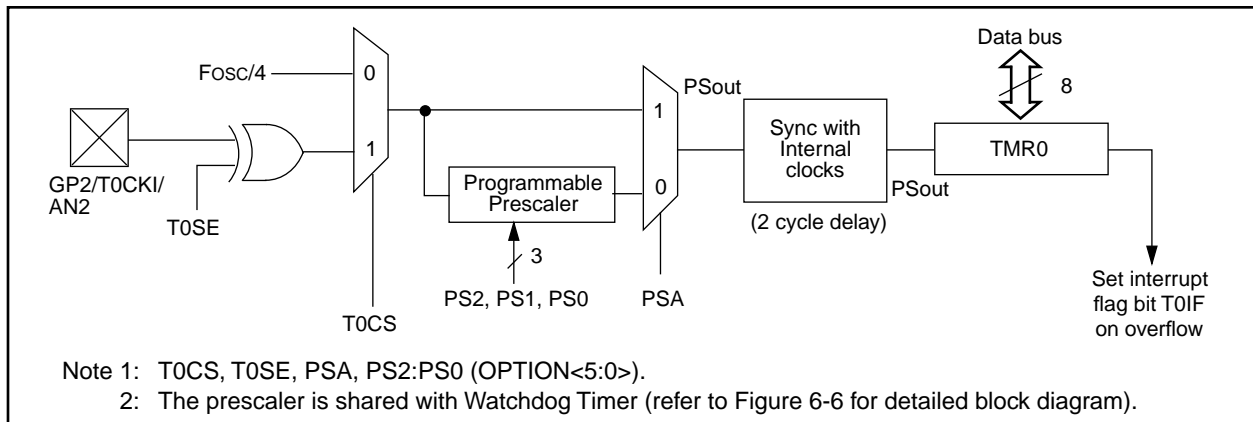
bit T0SE selects the rising edge. Restrictions on the external clock input are discussed in detail in Section 6.2.

The prescaler is mutually exclusively shared between the Timer0 module and the Watchdog Timer. The prescaler assignment is controlled in software by control bit PSA (OPTION<3>). Clearing bit PSA will assign the prescaler to the Timer0 module. The prescaler is not readable or writable. When the prescaler is assigned to the Timer0 module, prescale values of 1:2, 1:4, ..., 1:256 are selectable. Section 6.3 details the operation of the prescaler.

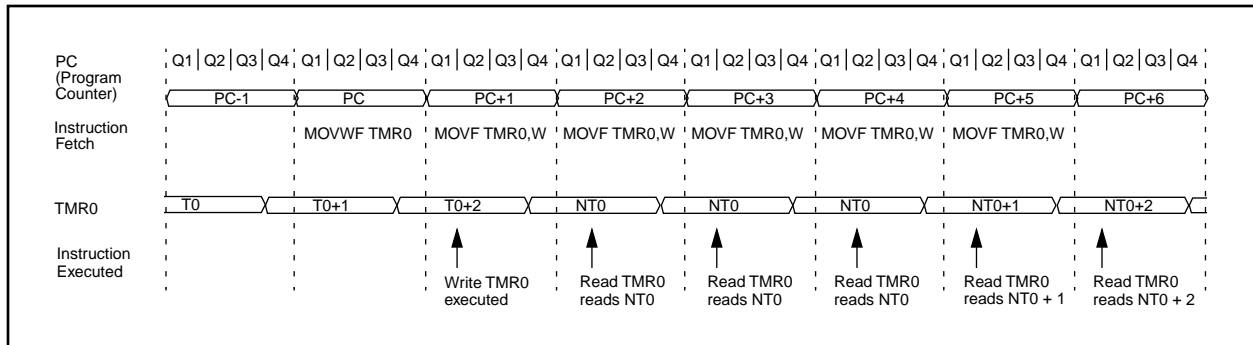
### 6.1 Timer0 Interrupt

The TMR0 interrupt is generated when the TMR0 register overflows from FFh to 00h. This overflow sets bit T0IF (INTCON<2>). The interrupt can be masked by clearing bit T0IE (INTCON<5>). Bit T0IF must be cleared in software by the Timer0 module interrupt service routine before re-enabling this interrupt. The TMR0 interrupt cannot awaken 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**

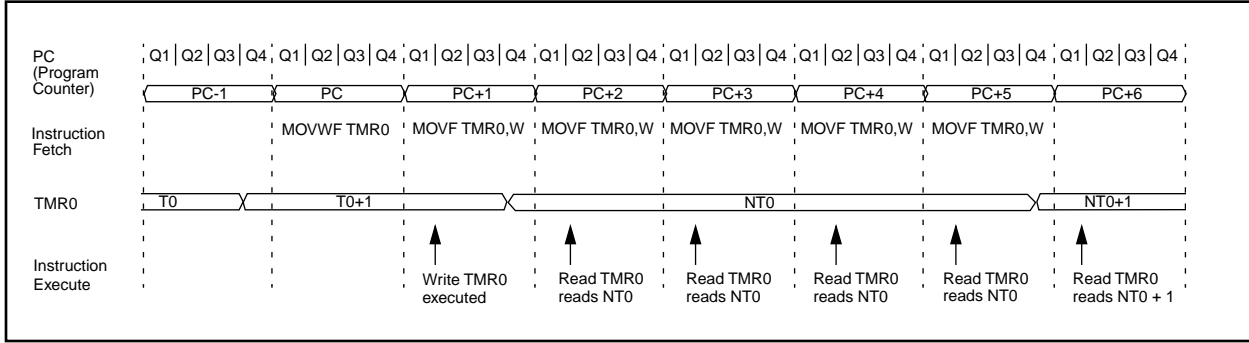


**FIGURE 6-2: TIMER0 TIMING: INTERNAL CLOCK/NO PRESCALE**

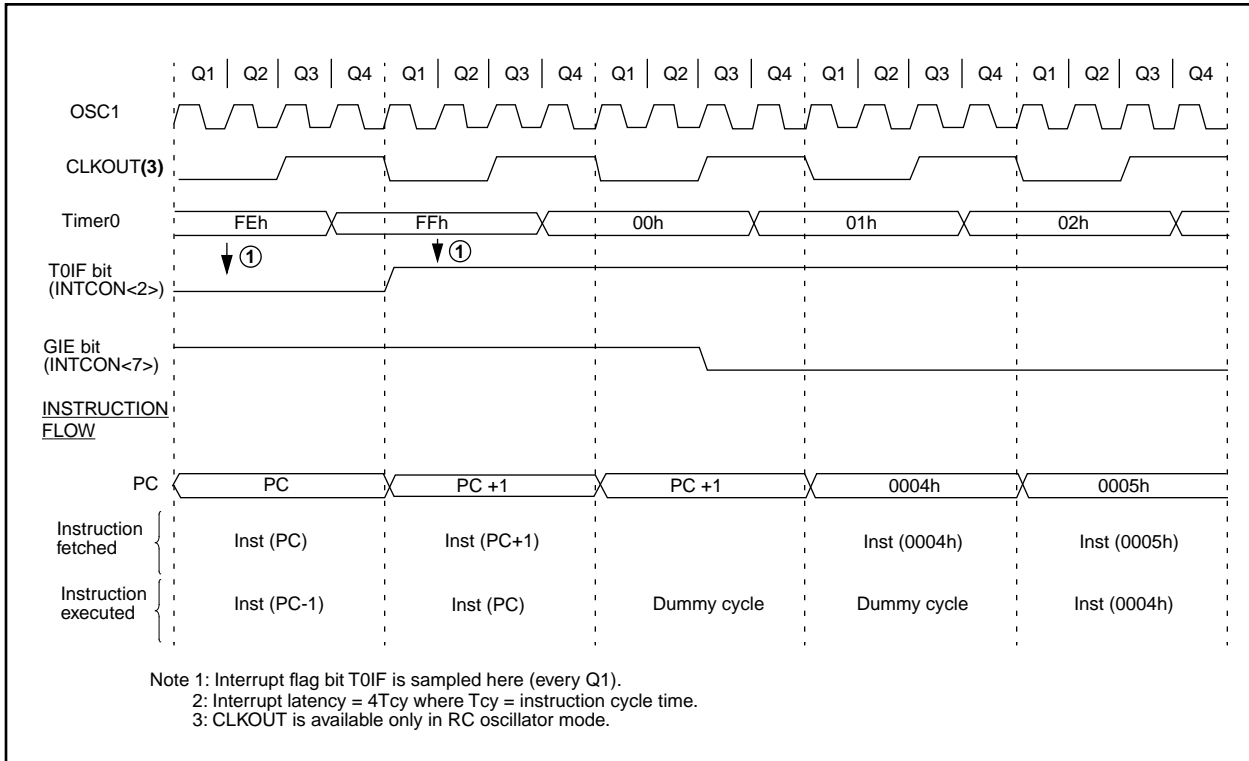


# PIC12C67X

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



**FIGURE 6-4: TIMER0 INTERRUPT TIMING**



## 6.2 Using Timer0 with an External Clock

When an external clock input is used for Timer0, it must meet certain requirements. The requirements ensure the external clock can be synchronized with the internal phase clock (TOSC). 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.

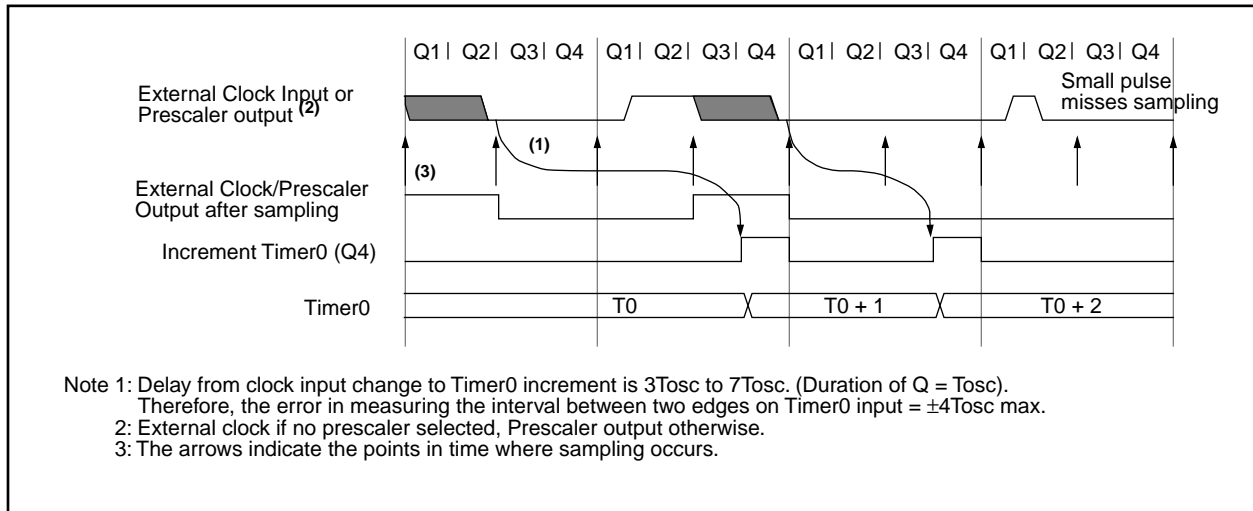
When a prescaler is used, the external clock input is divided by the asynchronous ripple-counter type pres-

caler 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 TMR0 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 Timer0 module 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**



# PIC12C67X

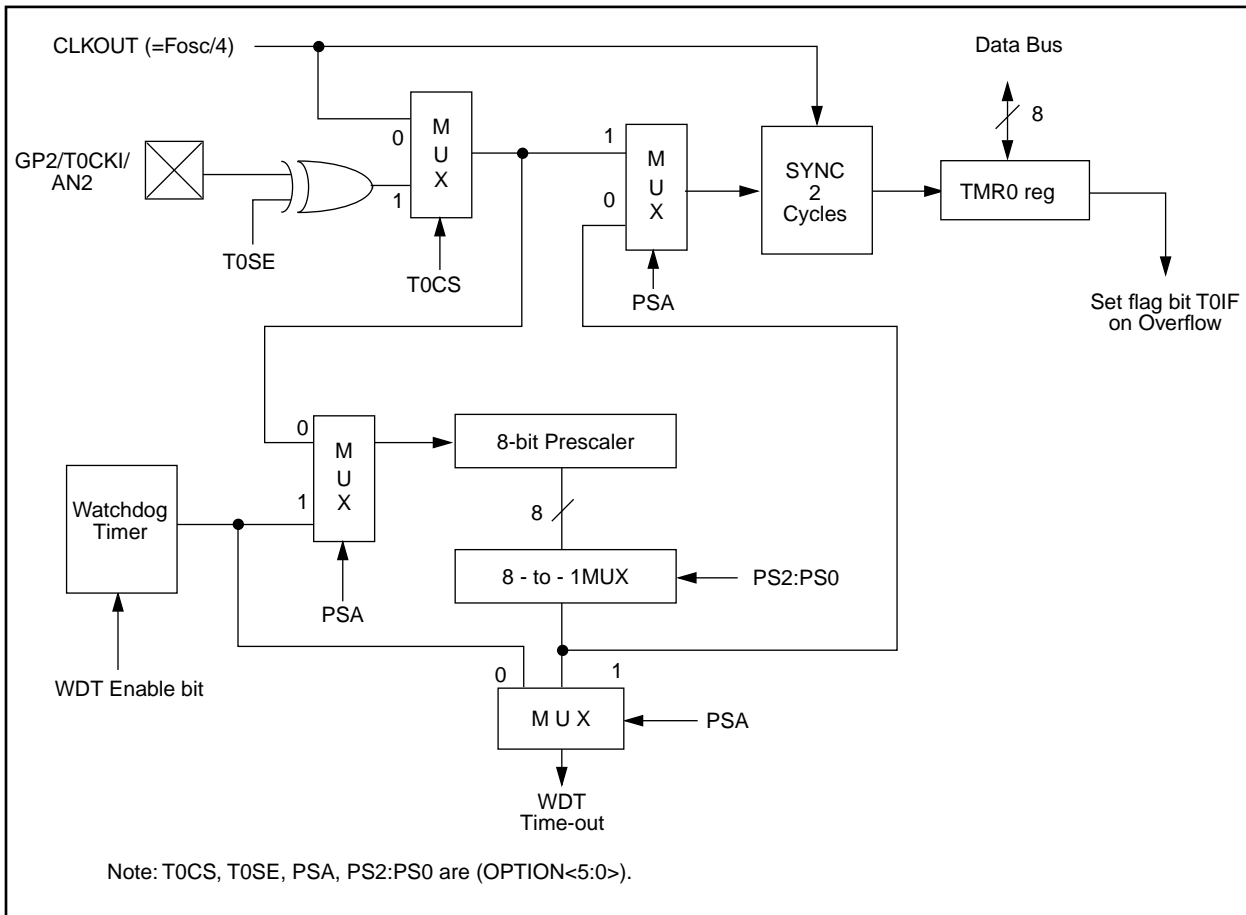
## 6.3 Prescaler

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 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 PS2:PS0 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**



## 6.3.1 SWITCHING PRESCALER ASSIGNMENT

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

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

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

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

To change prescaler from the WDT to the Timer0 module use the sequence shown in Example 6-2.

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

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

**TABLE 6-1: REGISTERS ASSOCIATED WITH TIMER0**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on all other Resets
01h	TMR0	Timer0 module's register								xxxx xxxx	uuuu uuuu
0Bh/8Bh	INTCON	GIE	PEIE	TOIE	INTE	GPIE	TOIF	INTF	GPIF	0000 000x	0000 000u
81h	OPTION	$\overline{\text{GPPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111
85h	TRIS	—	—	TRIS5	TRIS4	TRIS3	TRIS2	TRIS1	TRIS0	--11 1111	--11 1111

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

# PIC12C67X

---

NOTES:



## 7.0 ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

The analog-to-digital (A/D) converter module has four analog inputs.

The A/D allows conversion of an analog input signal to a corresponding 8-bit digital number (refer to Application Note AN546 for use of A/D Converter). The output of the sample and hold is the input into the converter, which generates the result via successive approximation. The analog reference voltage is software selectable to either the device's positive supply voltage (VDD) or the voltage level on the GP1/AN1/VREF pin. The A/D converter has a unique feature of being able to operate while the device is in SLEEP mode.

The A/D module has three registers. These registers are:

- A/D Result Register (ADRES)
- A/D Control Register 0 (ADCON0)
- A/D Control Register 1 (ADCON1)

The ADCON0 register, shown in Figure 7-1, controls the operation of the A/D module. The ADCON1 register, shown in Figure 7-2, configures the functions of the port pins. The port pins can be configured as analog inputs (GP1 can also be a voltage reference) or as digital I/O.

**Note:** If the port pins are configured as analog inputs (reset condition), reading the port (MOVF GP,W) results in reading '0's.

**FIGURE 7-1: ADCON0 REGISTER (ADDRESS 1Fh)**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADCS1	ADCS0	r	CHS1	CHS0	GO/DONE	r	ADON
bit7							bit0

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

bit 7-6: **ADCS1:ADCS0:** A/D Conversion Clock Select bits  
00 = FOSC/2  
01 = FOSC/8  
10 = FOSC/32  
11 = FRC (clock derived from an RC oscillation)

bit 5: **Reserved**

bit 4-3: **CHS1:CHS0:** Analog Channel Select bits  
00 = channel 0, (GP0/AN0)  
01 = channel 1, (GP1/AN1)  
10 = channel 2, (GP2/AN2)  
11 = channel 3, (GP4/AN3)

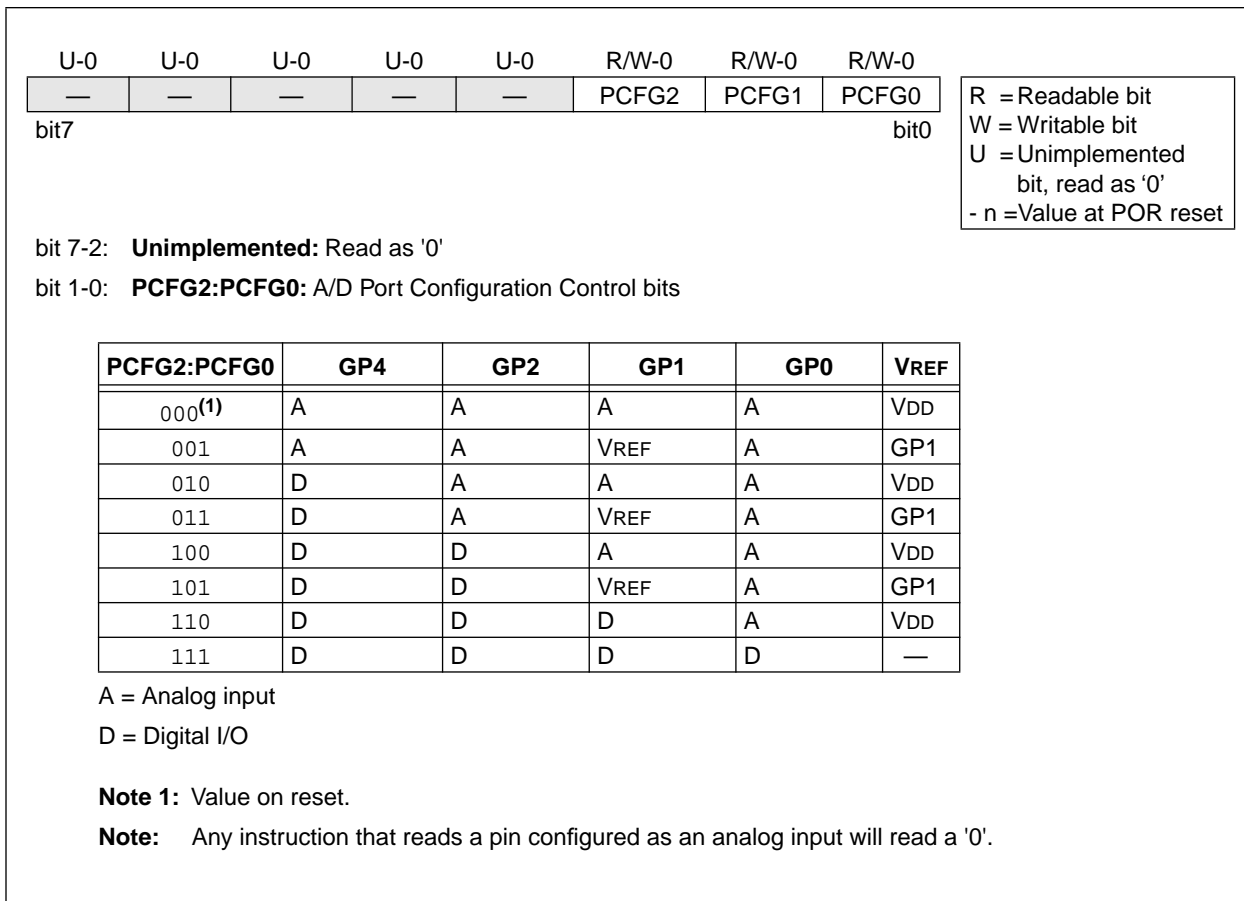
bit 2: **GO/DONE:** A/D Conversion Status bit  
If ADON = 1  
1 = A/D conversion in progress (setting this bit starts the A/D conversion)  
0 = A/D conversion not in progress (This bit is automatically cleared by hardware when the A/D conversion is complete)

bit 1: **Reserved**

bit 0: **ADON:** A/D On bit  
1 = A/D converter module is operating  
0 = A/D converter module is shutoff and consumes no operating current

# PIC12C67X

**FIGURE 7-2: ADCON1 REGISTER (ADDRESS 9Fh)**

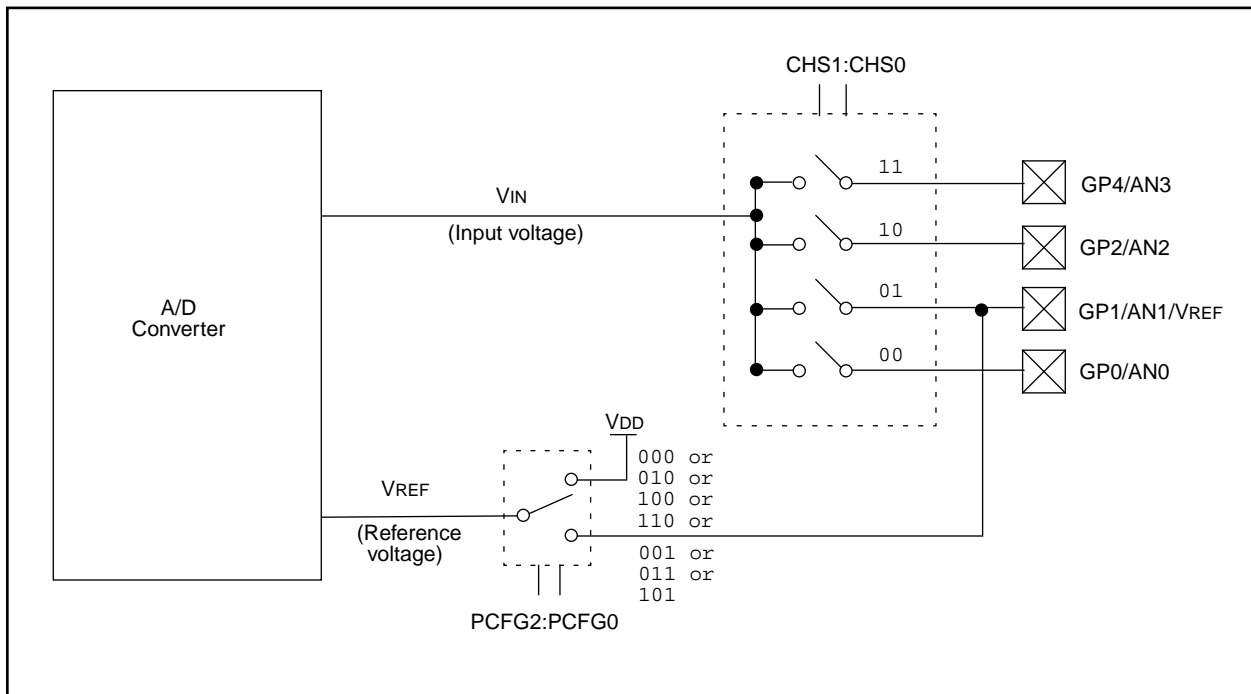


The ADRES register contains the result of the A/D conversion. When the A/D conversion is complete, the result is loaded into the ADRES register, the GO/DONE bit (ADCON0<2>) is cleared, and A/D interrupt flag bit ADIF (PIE1<6>) is set. The block diagrams of the A/D module are shown in Figure 7-3.

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 7.1. 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)
  - 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 GO/DONE bit (ADCON0)
5. Wait for A/D conversion to complete, by either:
  - Polling for the 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 next conversion, go to step 1 or step 2 as required. The A/D conversion time per bit is defined as TAD. A minimum wait of 2TAD is required before next acquisition starts.

**FIGURE 7-3: A/D BLOCK DIAGRAM**



# PIC12C67X

## 7.1 A/D Sampling Requirements

For the A/D converter to meet its specified accuracy, the charge holding capacitor (CHOLD) must be allowed to fully charge to the input channel voltage level. The analog input model is shown in Figure 7-4. The source impedance ( $R_s$ ) and the internal sampling switch ( $R_{ss}$ ) impedance directly affect the time required to charge the capacitor CHOLD. The sampling switch ( $R_{ss}$ ) impedance varies over the device voltage ( $V_{DD}$ ), see Figure 7-4. **The maximum recommended impedance for analog sources is 10 k $\Omega$ .** After the analog input channel is selected (changed) this acquisition must be done before the conversion can be started.

To calculate the minimum acquisition time, Equation 7-1 may be used. This equation assumes that 1/2 LSB error is used (512 steps for the A/D). The 1/2 LSB error is the maximum error allowed for the A/D to meet its specified resolution.

### EQUATION 7-1: A/D MINIMUM CHARGING TIME

$$V_{HOLD} = (V_{REF} - (V_{REF}/512)) \cdot (1 - e^{(-T_c/CHOLD)(R_{IC} + R_{SS} + R_s)})$$

or

$$T_c = -(51.2 \text{ pF})(1 \text{ k}\Omega + R_{SS} + R_s) \ln(1/511)$$

Example 7-1 shows the calculation of the minimum required acquisition time  $T_{ACQ}$ . This calculation is based on the following system assumptions.

$R_s = 10 \text{ k}\Omega$

1/2 LSB error

$V_{DD} = 5V \rightarrow R_{SS} = 7 \text{ k}\Omega$

Temp (system max.) = 50°C

$V_{HOLD} = 0 @ t = 0$

**Note 1:** The reference voltage ( $V_{REF}$ ) has no effect on the equation, since it cancels itself out.

**Note 2:** The charge holding capacitor (CHOLD) is not discharged after each conversion.

**Note 3:** The maximum recommended impedance for analog sources is 10 k $\Omega$ . This is required to meet the pin leakage specification.

**Note 4:** After a conversion has completed, a 2.0 TAD delay must complete before acquisition can begin again. During this time the holding capacitor is not connected to the selected A/D input channel.

### EXAMPLE 7-1: CALCULATING THE MINIMUM REQUIRED SAMPLE TIME

$T_{ACQ} =$  Amplifier Settling Time +  
Holding Capacitor Charging Time +  
Temperature Coefficient

$$T_{ACQ} = 5 \mu\text{s} + T_c + [(Temp - 25^\circ\text{C})(0.05 \mu\text{s}/^\circ\text{C})]$$

$$T_c = -CHOLD (R_{IC} + R_{SS} + R_s) \ln(1/512)$$

$$-51.2 \text{ pF} (1 \text{ k}\Omega + 7 \text{ k}\Omega + 10 \text{ k}\Omega) \ln(0.0020)$$

$$-51.2 \text{ pF} (18 \text{ k}\Omega) \ln(0.0020)$$

$$-0.921 \mu\text{s} (-6.2146)$$

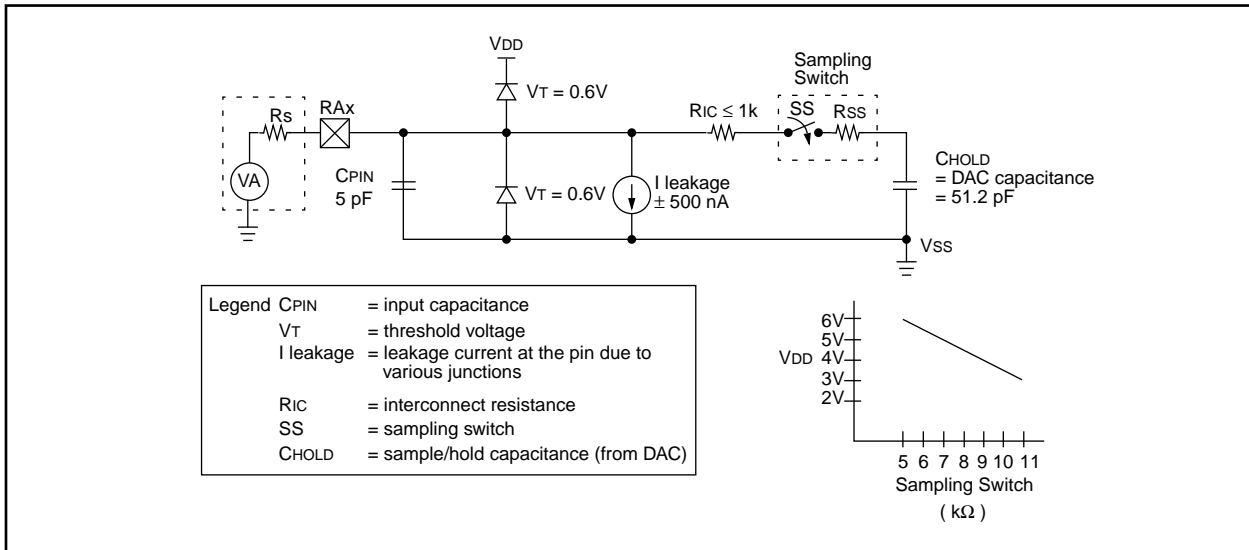
$$5.724 \mu\text{s}$$

$$T_{ACQ} = 5 \mu\text{s} + 5.724 \mu\text{s} + [(50^\circ\text{C} - 25^\circ\text{C})(0.05 \mu\text{s}/^\circ\text{C})]$$

$$10.724 \mu\text{s} + 1.25 \mu\text{s}$$

$$11.974 \mu\text{s}$$

FIGURE 7-4: ANALOG INPUT MODEL



## 7.2 Selecting the A/D Conversion Clock

The A/D conversion time per bit is defined as  $T_{AD}$ . The A/D conversion requires 9.5  $T_{AD}$  per 8-bit conversion. The source of the A/D conversion clock is software selected. The four possible options for  $T_{AD}$  are:

- 2Tosc
- 8Tosc
- 32Tosc
- Precision internal 4 MHz oscillator

For correct A/D conversions, the A/D conversion clock ( $T_{AD}$ ) must be selected to ensure a minimum  $T_{AD}$  time of 1.6  $\mu$ s.

Table 7-1 shows the resultant  $T_{AD}$  times derived from the device operating frequencies and the A/D clock source selected.

## 7.3 Configuring Analog Port Pins

The ADCON1 and TRIS registers control the operation of the A/D port pins. The port pins that are desired as analog inputs must have their corresponding TRIS bits set (input). If the TRIS bit is cleared (output), the digital output level ( $V_{OH}$  or  $V_{OL}$ ) will be converted.

The A/D operation is independent of the state of the CHS2:CHS0 bits and the TRIS bits.

**Note 1:** When reading the port register, all pins configured as analog input channel will read as cleared (a low level). Pins configured as digital inputs, will convert an analog input. Analog levels on a digitally configured input will not affect the conversion accuracy.

**Note 2:** Analog levels on any pin that is defined as a digital input (including the AN3:AN0 pins), may cause the input buffer to consume current that is out of the devices specification.

**TABLE 7-1:  $T_{AD}$  vs. DEVICE OPERATING FREQUENCIES**

AD Clock Source ( $T_{AD}$ )		Device Frequency		
Operation	ADCS1:ADCS0	4 MHz	1.25 MHz	333.33 kHz
2Tosc	00	500 ns <sup>(2)</sup>	1.6 $\mu$ s	6 $\mu$ s
8Tosc	01	2.0 $\mu$ s	6.4 $\mu$ s	24 $\mu$ s <sup>(3)</sup>
32Tosc	10	8.0 $\mu$ s	25.6 $\mu$ s <sup>(3)</sup>	96 $\mu$ s <sup>(3)</sup>
Internal ADC RC Oscillator <sup>(5)</sup>	11	2 - 6 $\mu$ s <sup>(1,4)</sup>	2 - 6 $\mu$ s <sup>(1,4)</sup>	2 - 6 $\mu$ s <sup>(1)</sup>

Note 1: The RC source has a typical  $T_{AD}$  time of 4  $\mu$ s.

2: These values violate the minimum required  $T_{AD}$  time.

3: For faster conversion times, the selection of another clock source is recommended.

4: While in RC mode, with device frequency above 1 MHz, conversion accuracy is out of specification.

5: For extended voltage devices (LC), please refer to Electrical Specifications section.

# PIC12C67X

---

## 7.4 A/D Conversions

Example 7-2 show how to perform an A/D conversion. The GP pins are configured as analog inputs. The analog reference (VREF) is the device VDD. The A/D interrupt is enabled, and the A/D conversion clock is FRC. The conversion is performed on the GP0 channel.

**Note:** The GO/DONE bit should **NOT** be set in the same instruction that turns on the A/D.

Clearing the GO/DONE bit during a conversion will abort the current conversion. The ADRES register will NOT be updated with the partially completed A/D conversion sample. That is, the ADRES register will continue to contain the value of the last completed conversion (or the last value written to the ADRES register). After the A/D conversion is aborted, a 2TAD wait is required before the next acquisition is started. After this 2TAD wait, an acquisition is automatically started on the selected channel.

### EXAMPLE 7-2: DOING AN A/D CONVERSION

```
BSF    STATUS, RP0        ; Select Page 1
CLRF   ADCON1             ; Configure A/D inputs
BSF    PIE1, ADIE         ; Enable A/D interrupts
BCF    STATUS, RP0        ; Select Page 0
MOVLW  0xC1               ; RC Clock, A/D is on, Channel 0 is selected
MOVWF  ADCON0             ;
BCF    PIR1, ADIF         ; Clear A/D interrupt flag bit
BSF    INTCON, PEIE       ; Enable peripheral interrupts
BSF    INTCON, GIE        ; Enable all interrupts
;
; Ensure that the required sampling time for the selected input channel has elapsed.
; Then the conversion may be started.
;
BSF    ADCON0, GO         ; Start A/D Conversion
:      ; The ADIF bit will be set and the GO/DONE bit
:      ; is cleared upon completion of the A/D Conversion.
```

## 7.5 A/D Operation During Sleep

The A/D module can operate during SLEEP mode. This requires that the A/D clock source be set to RC (ADCS1:ADCS0 = 11). When the RC clock source is selected, the A/D module waits one instruction cycle before starting the conversion. This allows the SLEEP instruction to be executed, which eliminates all digital switching noise from the conversion. When the conversion is completed the GO/DONE bit will be cleared, and the result loaded into the ADRES register. If the A/D interrupt is enabled, the device will wake-up from SLEEP. If the A/D interrupt is not enabled, the A/D module will then be turned off, although the ADON bit will remain set.

When the A/D clock source is another clock option (not RC), a SLEEP instruction will cause the present conversion to be aborted and the A/D module to be turned off, though the ADON bit will remain set.

Turning off the A/D places the A/D module in its lowest current consumption state.

**Note:** For the A/D module to operate in SLEEP, the A/D clock source must be set to RC (ADCS1:ADCS0 = 11). To perform an A/D conversion in SLEEP, the GO/DONE bit must be set, followed by the SLEEP instruction.

## 7.6 A/D Accuracy/Error

The overall accuracy of the A/D is less than  $\pm 1$  LSb for  $V_{DD} = 5V \pm 10\%$  and the analog  $V_{REF} = V_{DD}$ . This overall accuracy includes offset error, full scale error, and integral error. The A/D converter is guaranteed to be monotonic. The resolution and accuracy may be less when either the analog reference ( $V_{DD}$ ) is less than 5.0V or when the analog reference ( $V_{REF}$ ) is less than  $V_{DD}$ .

The maximum pin leakage current is  $\pm 5 \mu A$ .

In systems where the device frequency is low, use of the A/D RC clock is preferred. At moderate to high frequencies,  $T_{AD}$  should be derived from the device oscillator.  $T_{AD}$  must not violate the minimum and should be  $\leq 8 \mu s$  for preferred operation. This is because  $T_{AD}$ , when derived from  $T_{OSC}$ , is kept away from on-chip phase clock transitions. This reduces, to a large extent, the effects of digital switching noise. This is not possible with the RC derived clock. The loss of accuracy due to digital switching noise can be significant if many I/O pins are active.

In systems where the device will enter SLEEP mode after the start of the A/D conversion, the RC clock source selection is required. In this mode, the digital noise from the modules in SLEEP are stopped. This method gives high accuracy.

## 7.7 Effects of a RESET

A device reset forces all registers to their reset state. This forces the A/D module to be turned off, and any conversion is aborted. The value that is in the ADRES register is not modified for a Power-on Reset. The ADRES register will contain unknown data after a Power-on Reset.

## 7.8 Connection Considerations

If the input voltage exceeds the rail values ( $V_{SS}$  or  $V_{DD}$ ) by greater than 0.2V, then the accuracy of the conversion is out of specification.

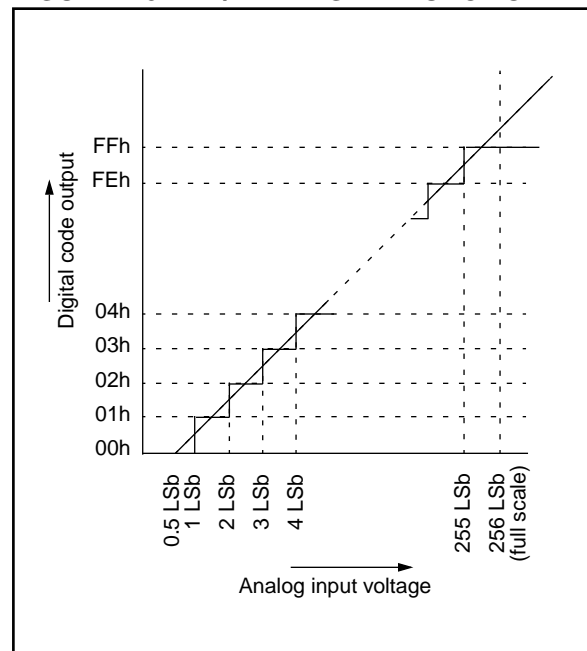
**Note:** For the PIC12C67X, care must be taken when using the GP4 pin in A/D conversions due to its proximity to the OSC1 pin.

An external RC filter is sometimes added for anti-aliasing of the input signal. The R component should be selected to ensure that the total source impedance is kept under the 10 k $\Omega$  recommended specification. Any external components connected (via hi-impedance) to an analog input pin (capacitor, zener diode, etc.) should have very little leakage current at the pin.

## 7.9 Transfer Function

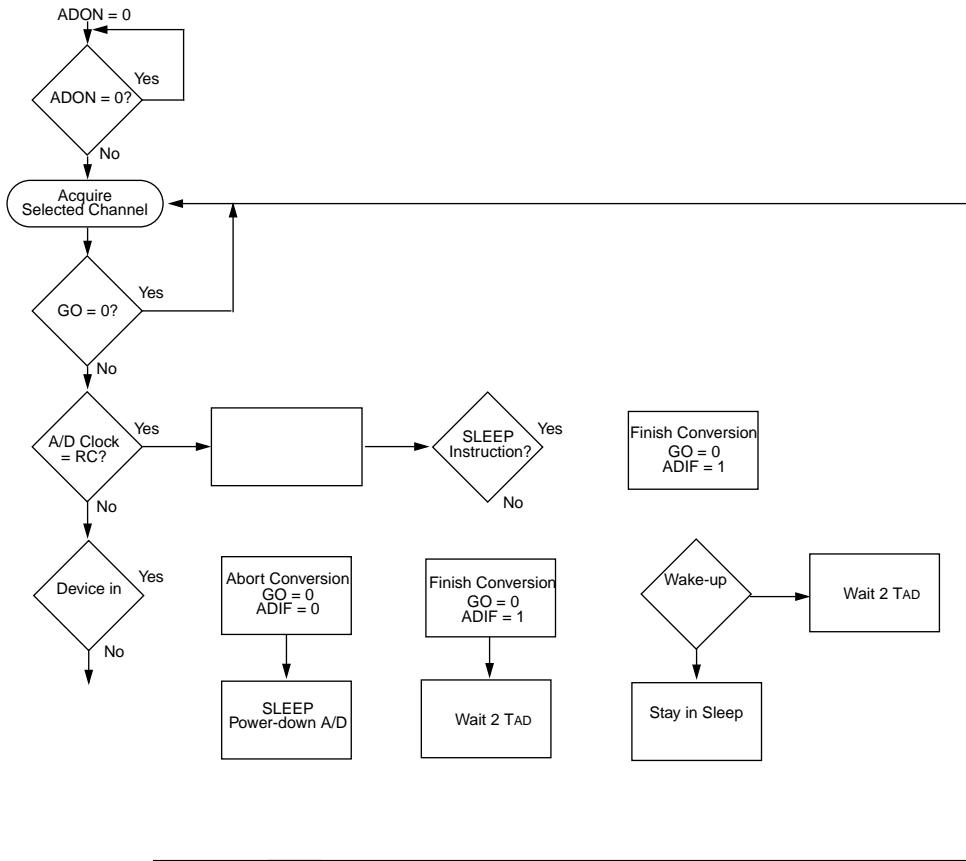
The ideal transfer function of the A/D converter is as follows: the first transition occurs when the analog input voltage ( $V_{AIN}$ ) is 1 LSb (or Analog  $V_{REF} / 256$ ) (Figure 7-5).

**FIGURE 7-5: A/D TRANSFER FUNCTION**



# PIC12C67X

**FIGURE 7-6: FLOWCHART OF A/D OPERATION**



**TABLE 7-2: SUMMARY OF A/D REGISTERS**

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 <sup>(1)</sup>
0Bh/8Bh	INTCON	GIE	PEIE	T0IE	INTE	GPIE	T0IF	INTF	GPIF	0000 000x	0000 000u
0Ch	PIR1	—	ADIF	—	—	—	—	—	—	-0-- ----	-0-- ----
8Ch	PIE1	—	ADIE	—	—	—	—	—	—	-0-- ----	-0-- ----
1Eh	ADRES	A/D Result Register								xxxx xxxx	uuuu uuuu
1Fh	ADCON0	ADCS1	ADCS0	r	CHS1	CHS0	GO/DONE	r	ADON	0000 0000	0000 0000
9Fh	ADCON1	—	—	—	—	—	PCFG2	PCFG1	PCFG0	---- -000	---- -000
05h	GPI0	—	—	GP5	GP4	GP3	GP2	GP1	GP0	--xx xxxx	--uu uuuu
85h	TRIS	—	—	TRIS5	TRIS4	TRIS3	TRIS2	TRIS1	TRIS0	--11 1111	--11 1111

Legend: x = unknown, u = unchanged, - = unimplemented read as '0', r = reserved. Shaded cells are not used for A/D conversion.

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



## 8.0 SPECIAL FEATURES OF THE CPU

What sets a microcontroller apart from other processors are special circuits to deal with the needs of real-time applications. The PIC12C67X 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:

- Oscillator selection
- Reset
  - Power-on Reset (POR)
  - Power-up Timer (PWRT)
  - Oscillator Start-up Timer (OST)
- Interrupts
- Watchdog Timer (WDT)
- SLEEP
- Code protection
- ID locations
- In-circuit serial programming

The PIC12C67X has a Watchdog Timer which can be shut off only through 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. With these two timers on-chip, most applications need no external reset circuitry.

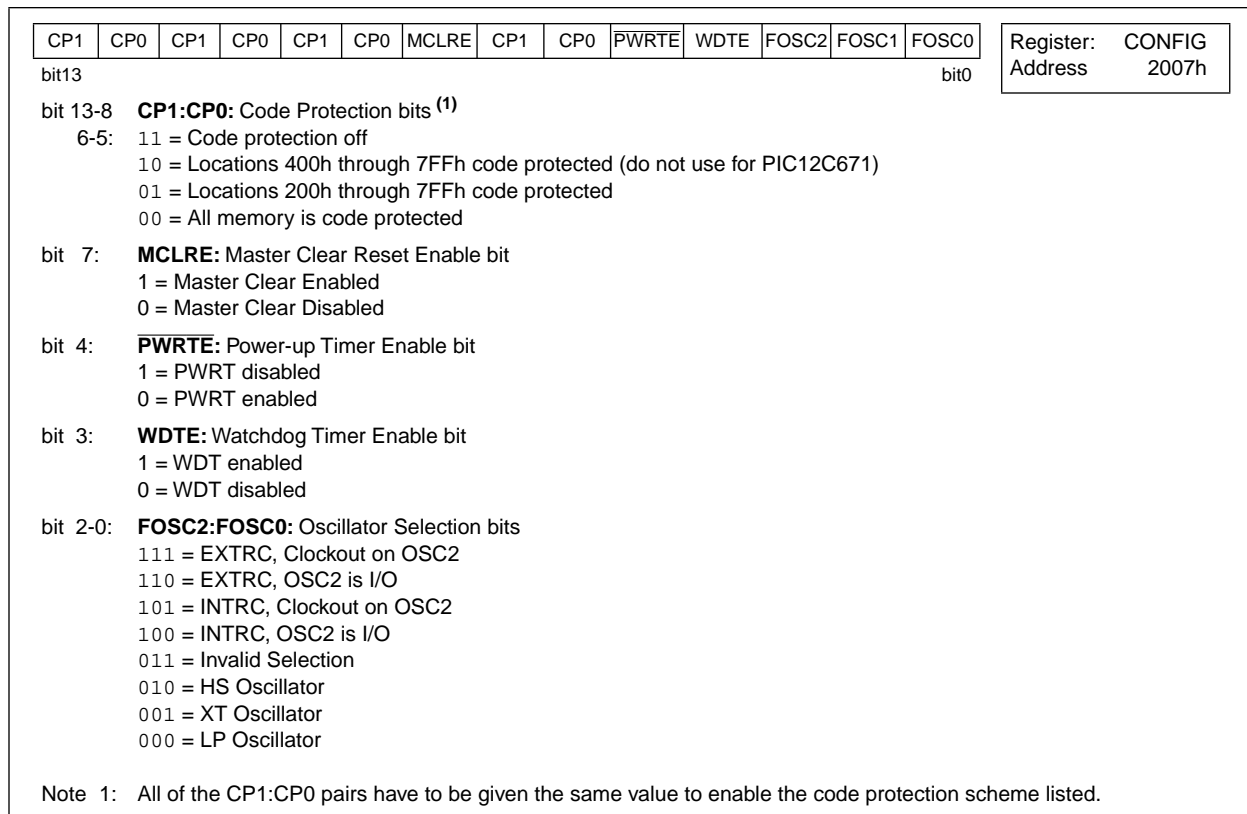
SLEEP mode is designed to offer a very low current power-down mode. The user can wake-up from SLEEP through external reset, Watchdog Timer Wake-up, or through an interrupt. Several oscillator options are also made available to allow the part to fit the application. The EXTRC oscillator option saves system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

### 8.1 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 8-1: CONFIGURATION WORD**



# PIC12C67X

## 8.2 Oscillator Configurations

### 8.2.1 OSCILLATOR TYPES

The PIC12C67X can be operated in seven different oscillator modes. The user can program three configuration bits (FOSC2:FOSC0) to select one of these seven modes:

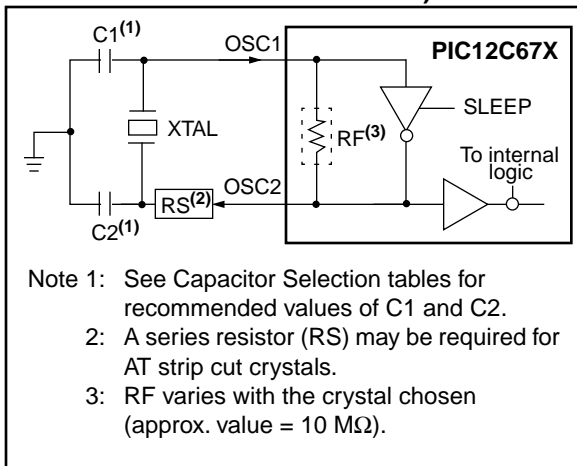
- LP: Low Power Crystal
- HS: High Speed Crystal Resonator
- XT: Crystal/Resonator
- INTRC\*: Internal 4 MHz Oscillator
- EXTRC\*: External Resistor/Capacitor

\*Can be configured to support CLKOUT

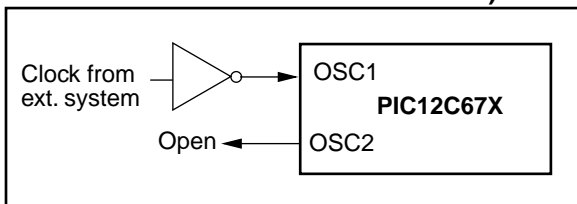
### 8.2.2 CRYSTAL OSCILLATOR / CERAMIC RESONATORS

In XT, HS or LP modes, a crystal or ceramic resonator is connected to the GP5/OSC1/CLKIN and GP4/OSC2 pins to establish oscillation (Figure 8-2). The PIC12C67X 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, HS or LP modes, the device can have an external clock source drive the GP5/OSC1/CLKIN pin (Figure 8-3).

**FIGURE 8-2: CRYSTAL OPERATION (OR CERAMIC RESONATOR) (XT, HS OR LP OSC CONFIGURATION)**



**FIGURE 8-3: EXTERNAL CLOCK INPUT OPERATION (XT, HS OR LP OSC CONFIGURATION)**



**TABLE 8-1: CAPACITOR SELECTION FOR CERAMIC RESONATORS - PIC12C67X**

Osc Type	Resonator Freq	Cap. Range C1	Cap. Range C2
XT	455 kHz	22-100 pF	22-100 pF
	2.0 MHz	15-68 pF	15-68 pF
	4.0 MHz	15-68 pF	15-68 pF
HS	4.0 MHz	15-68 pF	15-68 pF
	8.0 MHz	10-68 pF	10-68 pF
	10.0 MHz	10-22 pF	10-22 pF

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 8-2: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR - PIC12C67X**

Osc Type	Resonator Freq	Cap. Range C1	Cap. Range C2
LP	32 kHz <sup>(1)</sup>	15 pF	15 pF
	100 kHz	15-30 pF	30-47 pF
	200 kHz	15-30 pF	15-30 pF
XT	100 kHz	15-30 pF	200-300 pF
	200 kHz	15-30 pF	100-200 pF
	455 kHz	15-30 pF	15-100 pF
	1 MHz	15-30 pF	15-30 pF
	2 MHz	15-30 pF	15-30 pF
	4 MHz	15-47 pF	15-47 pF
HS	4 MHz	15-30 pF	15-30 pF
	8 MHz	15-30 pF	15-30 pF
	10 MHz	15-30 pF	15-30 pF

Note 1: For VDD > 4.5V, C1 = C2 ≈ 30 pF is recommended.

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.

## 8.2.3 EXTERNAL CRYSTAL OSCILLATOR CIRCUIT

Either a prepackaged oscillator or a simple oscillator circuit with TTL gates can be used as an external crystal oscillator circuit. 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 parallel resonance, or one with series resonance.

Figure 8-4 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-degree phase shift that a parallel oscillator requires. The 4.7 k $\Omega$  resistor provides the negative feedback for stability. The 10 k $\Omega$  potentiometers bias the 74AS04 in the linear region. This circuit could be used for external oscillator designs.

**FIGURE 8-4: EXTERNAL PARALLEL RESONANT CRYSTAL OSCILLATOR CIRCUIT**

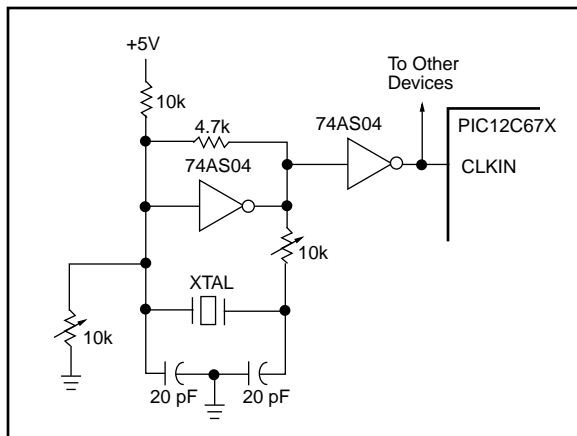
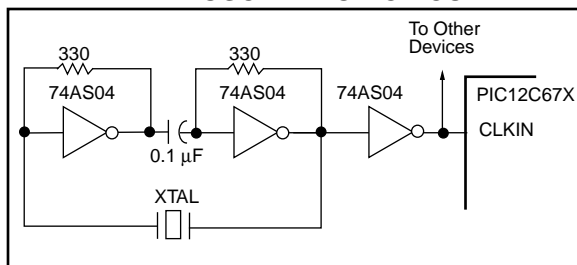


Figure 8-5 shows a series resonant oscillator circuit. This circuit is also designed to use the fundamental frequency of the crystal. The inverter performs a 180-degree phase shift in a series resonant oscillator circuit. The 330  $\Omega$  resistors provide the negative feedback to bias the inverters in their linear region.

**FIGURE 8-5: EXTERNAL SERIES RESONANT CRYSTAL OSCILLATOR CIRCUIT**



## 8.2.4 EXTERNAL 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 ( $R_{ext}$ ) and capacitor ( $C_{ext}$ ) 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  $C_{ext}$  values. The user also needs to take into account variation due to tolerance of external R and C components used.

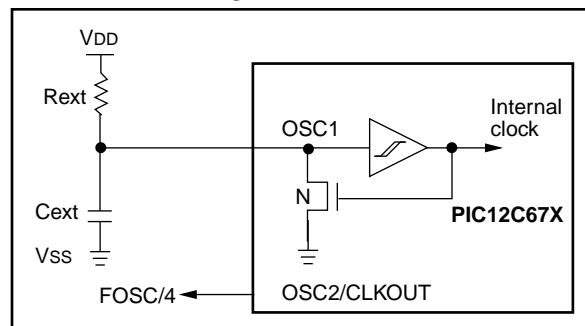
Figure 8-6 shows how the R/C combination is connected to the PIC12C67X. For  $R_{ext}$  values below 2.2 k $\Omega$ , the oscillator operation may become unstable, or stop completely. For very high  $R_{ext}$  values (e.g., 1 M $\Omega$ ) the oscillator becomes sensitive to noise, humidity and leakage. Thus, we recommend keeping  $R_{ext}$  between 3 k $\Omega$  and 100 k $\Omega$ .

Although the oscillator will operate with no external capacitor ( $C_{ext} = 0$  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.

The Electrical Specifications sections show 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).

Also, see the Electrical Specifications sections for variation of oscillator frequency due to  $V_{DD}$  for given  $R_{ext}/C_{ext}$  values as well as frequency variation due to operating temperature for given R, C, and  $V_{DD}$  values.

**FIGURE 8-6: EXTERNAL RC OSCILLATOR MODE**



# PIC12C67X

## 8.2.5 INTERNAL 4 MHz RC OSCILLATOR

The internal RC oscillator provides a fixed 4 MHz (nominal) system clock at  $V_{DD} = 5V$  and  $25^{\circ}C$ , see "Electrical Specifications" section for information on variation over voltage and temperature.

In addition, a calibration instruction is programmed into the last address of the program memory which contains the calibration value for the internal RC oscillator. This value is programmed as a `RETLW XX` instruction where `XX` is the calibration value. In order to retrieve the calibration value, issue a `CALL YY` instruction where `YY` is the last location in program memory (03FFh for the PIC12C671, 07FFh for the PIC12C672). Control will be returned to the user's program with the calibration value loaded into the `W` register. The program should then perform a `MOVWF OSCCAL` instruction to load the value into the internal RC oscillator trim register.

`OSCCAL`, when written to with the calibration value, will "trim" the internal oscillator to remove process variation from the oscillator frequency. Bits `<7:4>`, `CAL3-CAL0` are used for fine calibration while bit 3, `CALFST`, and bit 2, `CALSLW` are used for more coarse adjustment. Adjusting `CAL3-0` from 0000 to 1111 yields a higher clock speed. Set `CALFST = 1` for greater increase in frequency or set `CALSLW = 1` for greater decrease in frequency. Note that bits 1 and 0 of `OSCCAL` are unimplemented and should be written as 0 when modifying `OSCCAL` for compatibility with future devices.

**Note:** Please note that erasing the device will also erase the pre-programmed internal calibration value for the internal oscillator. The calibration value must be saved prior to erasing the part.

## 8.2.6 CLKOUT

The PIC12C67X can be configured to provide a clock out signal `CLKOUT` on pin 3 when the configuration word address (2007h) is programmed with `FOSC2`, `FOSC1`, `FOSC0` equal to 101 for `INTRC` or 111 for `EXTRC`. The oscillator frequency, divided by 4 can be used for test purposes or to synchronize other logic.

## 8.3 Reset

The PIC12C67X differentiates between various kinds of reset:

- Power-on Reset (POR)
- $\overline{MCLR}$  reset during normal operation
- $\overline{MCLR}$  reset during SLEEP
- WDT Reset (normal operation)

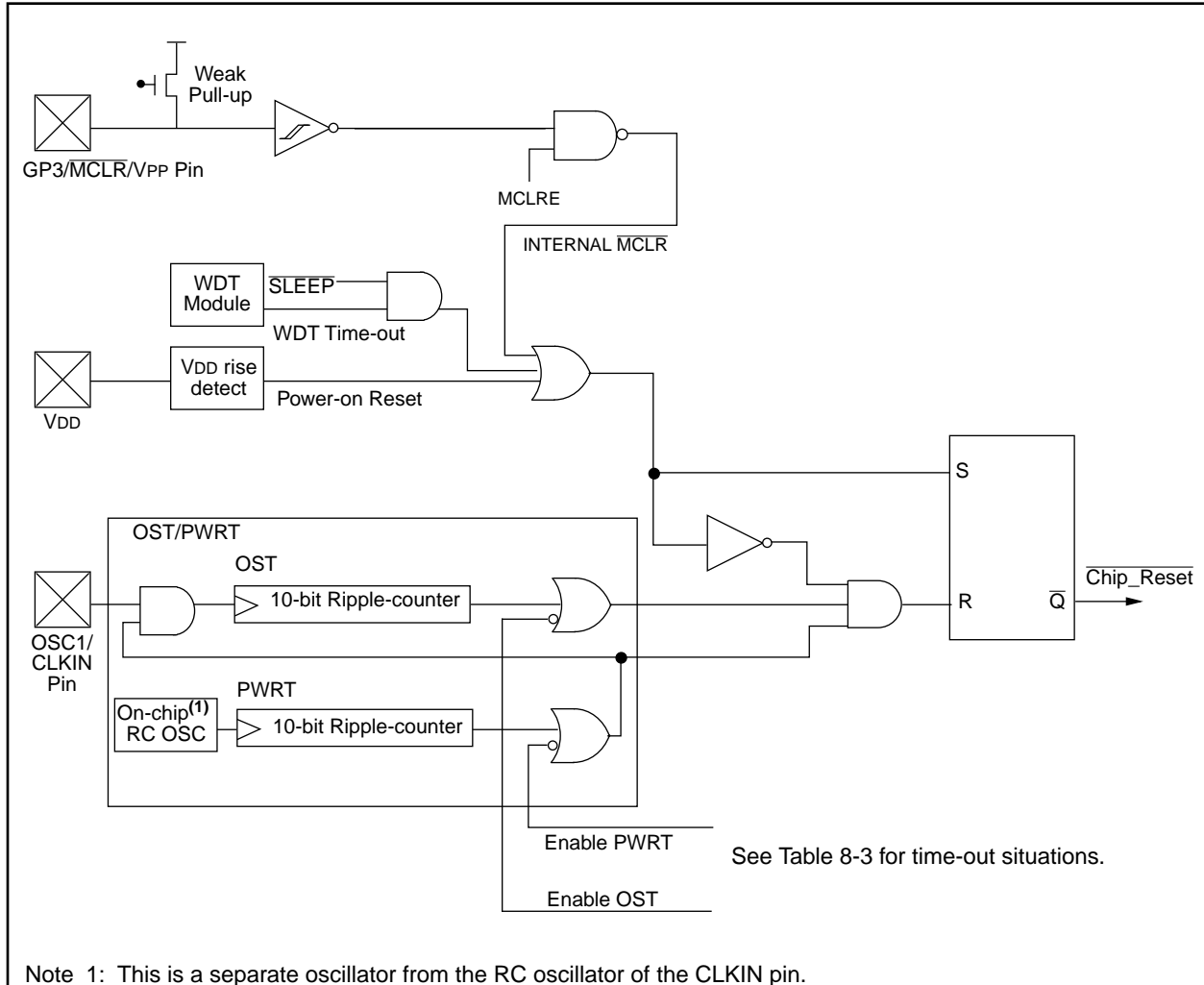
Some registers are not affected in any reset condition; their status is unknown on POR and unchanged in any other reset. Most other registers are reset to a "reset state" on Power-on Reset (POR), on the  $\overline{MCLR}$  and WDT Reset, on  $\overline{MCLR}$  reset during SLEEP. They are not affected by a WDT Wake-up, which is viewed as the resumption of normal operation. The  $\overline{TO}$  and  $\overline{PD}$  bits are set or cleared differently in different reset situations as indicated in Table 8-4. These bits are used in software to determine the nature of the reset. See Table 8-5 for a full description of reset states of all registers.

A simplified block diagram of the on-chip reset circuit is shown in Figure 8-7.

The PIC12C67X has a  $\overline{MCLR}$  noise filter in the  $\overline{MCLR}$  reset path. The filter will detect and ignore small pulses.

It should be noted that a WDT Reset does not drive  $\overline{MCLR}$  pin low.

**FIGURE 8-7: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT**



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

# PIC12C67X

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

### 8.4.1 POWER-ON RESET (POR)

A Power-on Reset pulse is generated on-chip when VDD rise is detected (in the range of 1.5V - 2.1V). To take advantage of the POR, just tie the MCLR pin directly (or 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."

### 8.4.2 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 DC parameters for details.

### 8.4.3 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.

### 8.4.4 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 8-8, Figure 8-9, and Figure 8-10 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 8-9). This is useful for testing purposes or to synchronize more than one PIC12C67X device operating in parallel.

Table 8-5 shows the reset conditions for all the registers.

### 8.4.5 POWER CONTROL/STATUS REGISTER (PCON)

The power control/status register, PCON (address 8Eh) has one bit. See Figure 4-8 for register.

Bit1 is  $\overline{POR}$  (Power-on Reset). It is cleared on a Power-on Reset and is unaffected otherwise. The user set 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 8-3: TIME-OUT IN VARIOUS SITUATIONS

Oscillator Configuration	Power-up		Wake-up from SLEEP
	PWRT = 0	PWRT = 1	
XT, HS, LP	72 ms + 1024T <sub>OSC</sub>	1024T <sub>OSC</sub>	1024T <sub>OSC</sub>
INTRC, EXTRC	72 ms	—	—

TABLE 8-4: STATUS BITS AND THEIR SIGNIFICANCE

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

**TABLE 8-5: RESET CONDITION FOR SPECIAL REGISTERS**

Condition	Program Counter	STATUS Register	PCON Register
Power-on Reset	000h	0001 1xxx	---- --0-
MCLR Reset during normal operation	000h	0001 1uuu	---- --u-
MCLR Reset during SLEEP	000h	0001 0uuu	---- --u-
WDT Reset during normal operation	000h	0000 1uuu	---- --u-
WDT Wake-up from SLEEP	PC + 1	uuu0 0uuu	---- --u-
Interrupt wake-up from SLEEP	PC + 1 <sup>(1)</sup>	uuu1 0uuu	---- --u-

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

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

**TABLE 8-6: INITIALIZATION CONDITIONS FOR ALL REGISTERS**

Register	Power-on Reset	MCLR Resets WDT Reset	Wake-up via WDT or Interrupt
W	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF	0000 0000	0000 0000	0000 0000
TMR0	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCL	0000 0000	0000 0000	PC + 1 <sup>(2)</sup>
STATUS	0001 1xxx	000q quuu <sup>(3)</sup>	uuuq quuu <sup>(3)</sup>
FSR	xxxx xxxx	uuuu uuuu	uuuu uuuu
GPIO	--xx xxxx	--uu uuuu	--uu uuuu
PCLATH	---0 0000	---0 0000	---u uuuu
INTCON	0000 000x	0000 000u	uuuu uuuu <sup>(1)</sup>
PIR1	-0-- ----	-0-- ----	-u-- ---- <sup>(1)</sup>
ADCON0	0000 0000	0000 0000	uuuu uuuu
OPTION	1111 1111	1111 1111	uuuu uuuu
TRIS	--11 1111	--11 1111	--uu uuuu
PIE1	-0-- 0000	-0-- 0000	-u-- uuuu
PCON	---- --0-	---- --u-	---- --u-
OSCCAL	0111 00--	uuuu uu--	uuuu uu--
ADCON1	---- -000	---- -000	---- -uuu

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

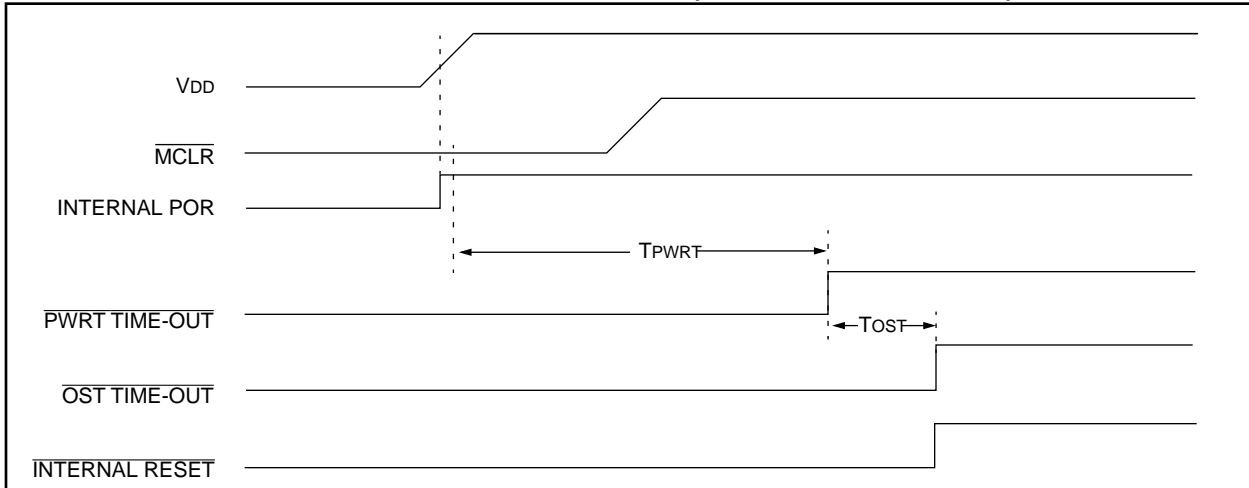
Note 1: One or more bits in INTCON and PIR1 will be affected (to cause wake-up).

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).

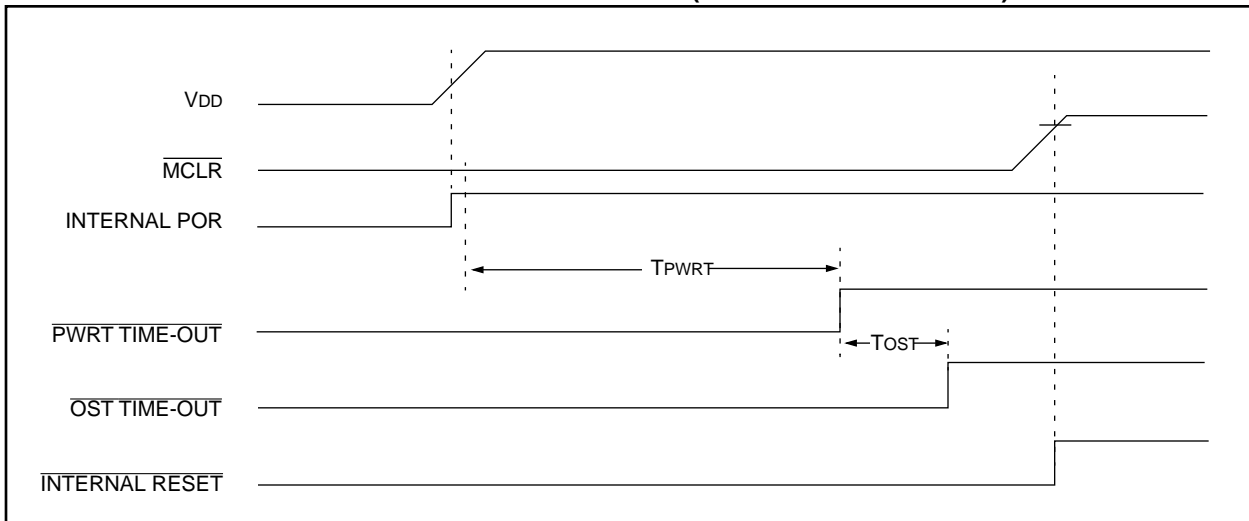
3: See Table 8-5 for reset value for specific condition.

# PIC12C67X

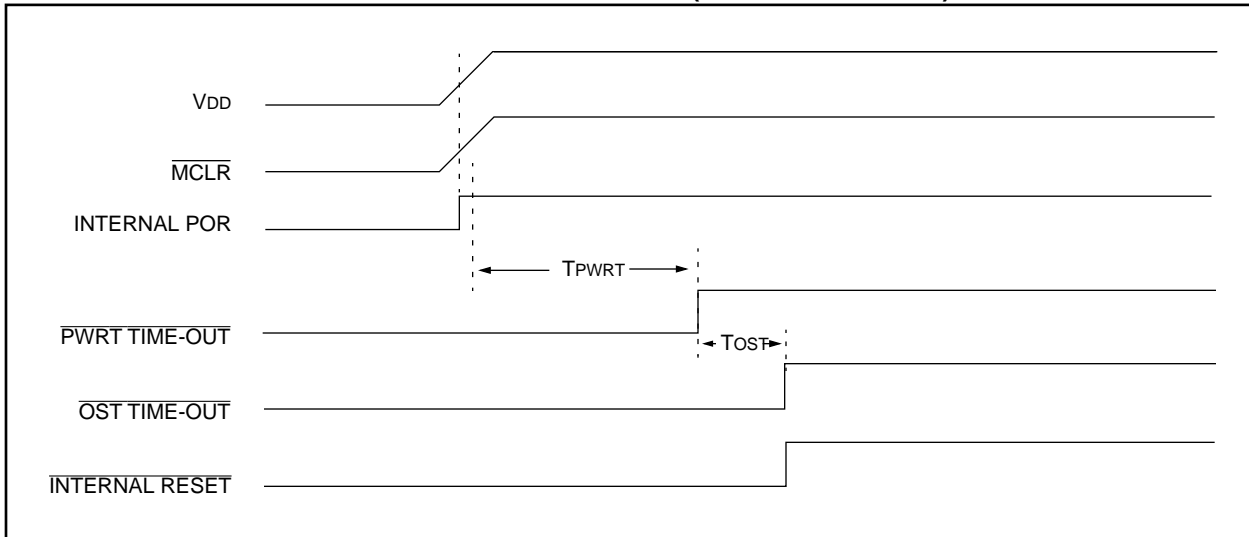
**FIGURE 8-8: TIME-OUT SEQUENCE ON POWER-UP ( $\overline{\text{MCLR}}$  NOT TIED TO  $V_{\text{DD}}$ ): CASE 1**



**FIGURE 8-9: TIME-OUT SEQUENCE ON POWER-UP ( $\overline{\text{MCLR}}$  NOT TIED TO  $V_{\text{DD}}$ ): CASE 2**

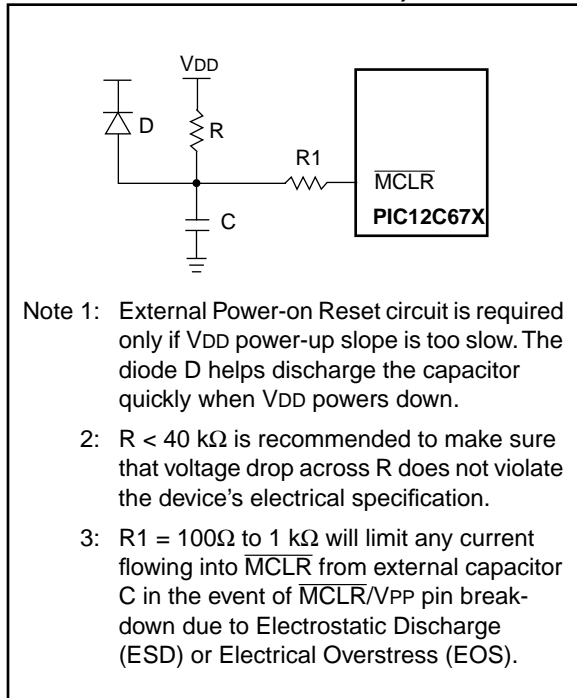


**FIGURE 8-10: TIME-OUT SEQUENCE ON POWER-UP ( $\overline{\text{MCLR}}$  TIED TO  $V_{\text{DD}}$ )**

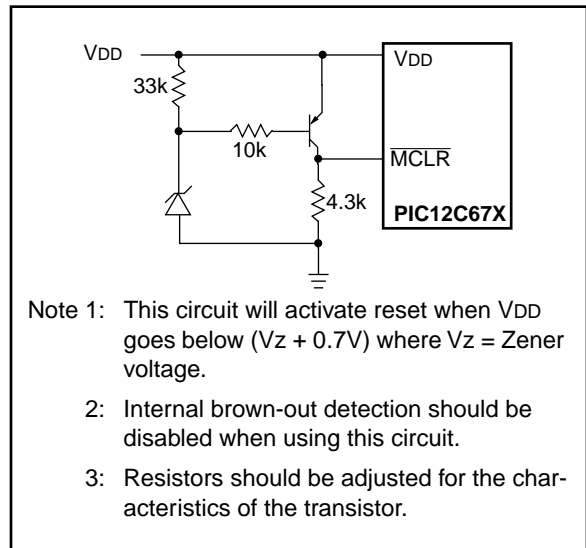




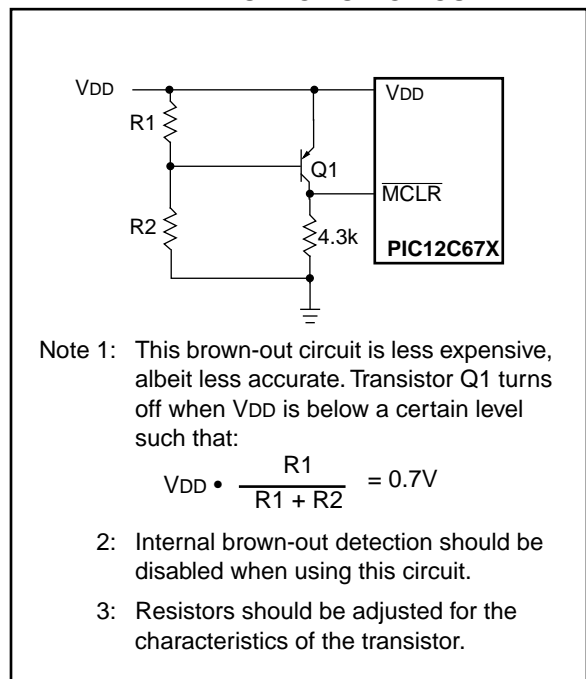
**FIGURE 8-11: EXTERNAL POWER-ON RESET CIRCUIT (FOR SLOW VDD POWER-UP)**



**FIGURE 8-12: EXTERNAL BROWN-OUT PROTECTION CIRCUIT 1**



**FIGURE 8-13: EXTERNAL BROWN-OUT PROTECTION CIRCUIT 2**



# PIC12C67X

## 8.5 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 bits are set regardless of the status of 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.

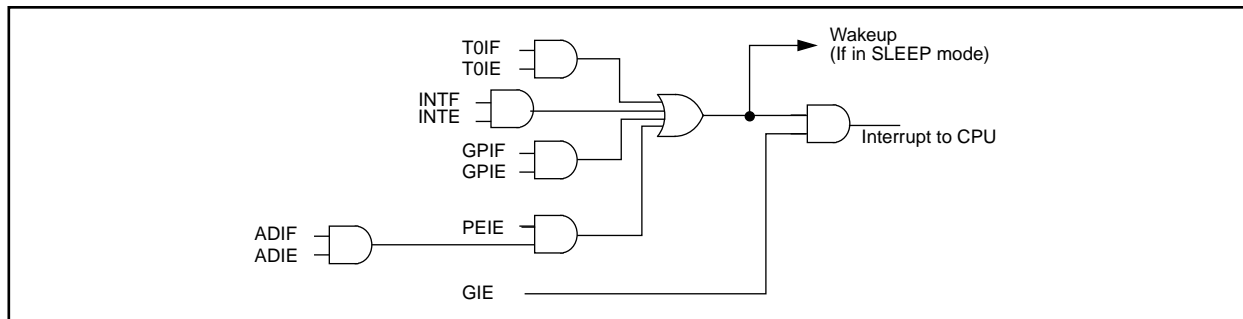
The GP2/INT, GPIO port change interrupt and the TMR0 overflow interrupt flags are contained in the INTCON register.

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.

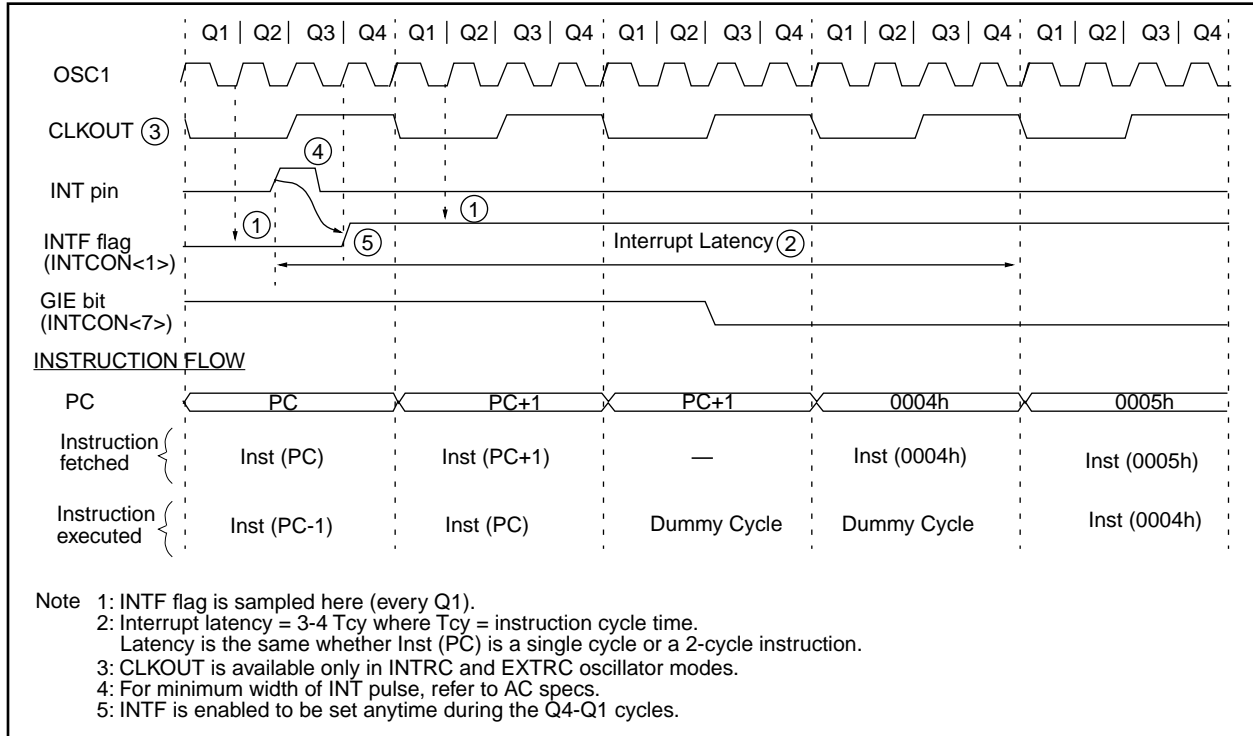
When an interrupt is responded to, the GIE bit is cleared to disable any further interrupt, the return address is pushed onto the stack and the PC is loaded with 0004h. Once in the interrupt service routine the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt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

For external interrupt events, such as GPIO change interrupt, the interrupt latency will be three or four instruction cycles. The exact latency depends when the interrupt event occurs (Figure 8-15). 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 8-14: INTERRUPT LOGIC



**FIGURE 8-15: INT PIN INTERRUPT TIMING**



# PIC12C67X

---

## 8.5.1 TMR0 INTERRUPT

An overflow (FFh → 00h) in the TMR0 register will set flag bit T0IF (INTCON<2>). The interrupt can be enabled/disabled by setting/clearing enable bit T0IE (INTCON<5>). (Section 6.0)

## 8.5.2 INT INTERRUPT

External interrupt on GP2/INT pin is edge triggered: either rising if bit INTEDG (OPTION<6>) is set, or falling, if the INTEDG bit is clear. When a valid edge appears on the GP2/INT pin, flag bit INTF (INTCON<1>) is set. This interrupt can be disabled by clearing enable bit INTE (INTCON<4>). Flag bit INTF must be cleared in software in the interrupt service routine before re-enabling this interrupt. The INT interrupt can wake-up the processor from SLEEP, if bit INTE was set prior to going into SLEEP. The status of global interrupt enable bit GIE decides whether or not the processor branches to the interrupt vector following wake-up. See Section 8.8 for details on SLEEP mode.

## 8.5.3 GPIO INTCON CHANGE

An input change on GP3, GP1 or GP0 sets flag bit GPIF (INTCON<0>). The interrupt can be enabled/disabled by setting/clearing enable bit GPIE (INTCON<3>). (Section 5.2)

## 8.6 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 i.e., W register and STATUS register. This will have to be implemented in software.

Example 8-1 store and restore the STATUS and W registers. The register, W\_TEMP, must be defined in both banks and must be defined at the same offset from the bank base address (i.e., if W\_TEMP is defined at 0x20 in bank 0, it must also be defined at 0xA0 in bank 1).

The example:

- a) Stores the W register.
- b) Stores the STATUS register in bank 0.
- c) Executes the ISR code.
- d) Restores the STATUS register (and bank select bit).
- e) Restores the W register.

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

```
MOVWF    W_TEMP          ;Copy W to TEMP register, could be bank one or zero
SWAPF    STATUS,W        ;Swap status to be saved into W
BCF      STATUS,RP0      ;Change to bank zero, regardless of current bank
MOVWF    STATUS_TEMP     ;Save status to bank zero STATUS_TEMP 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
```

## 8.7 Watchdog Timer (WDT)

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

### 8.7.1 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 early and generating a premature device `RESET` condition.

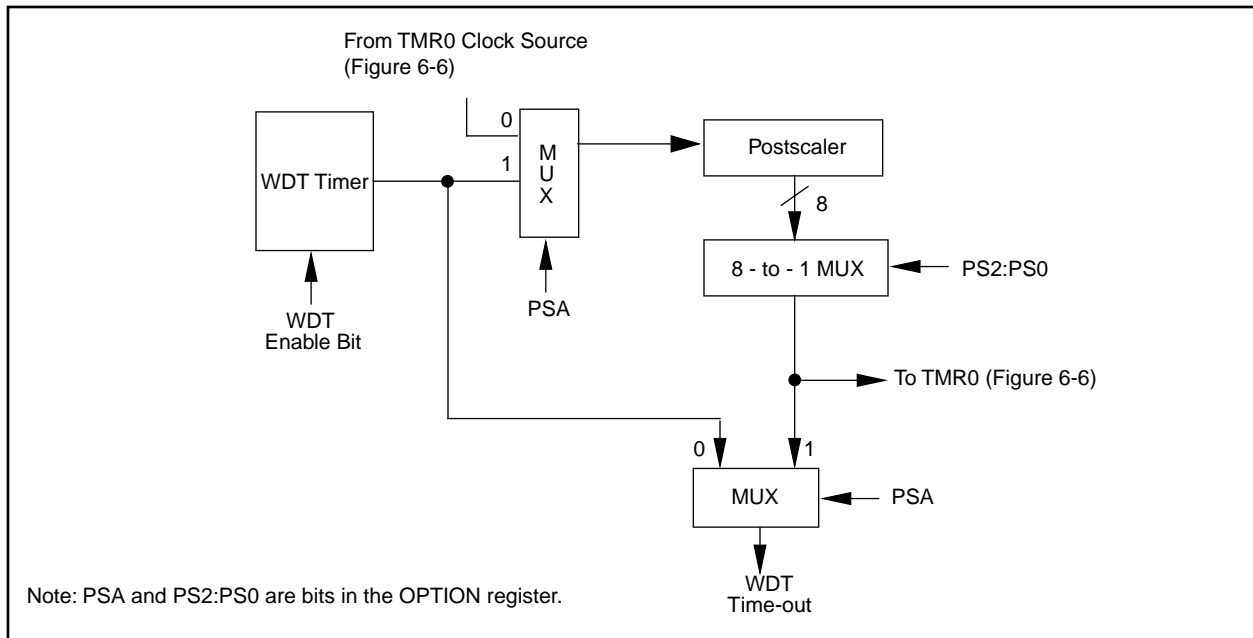
The  $\overline{TO}$  bit in the `STATUS` register will be cleared upon a Watchdog Timer time-out.

### 8.7.2 WDT PROGRAMMING CONSIDERATIONS

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

**Note:** When the prescaler is assigned to the WDT, always execute a `CLRWDT` instruction before changing the prescale value, otherwise a WDT reset may occur.

**FIGURE 8-16: WATCHDOG TIMER BLOCK DIAGRAM**



**FIGURE 8-17: SUMMARY OF WATCHDOG TIMER REGISTERS**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
2007h	Config. bits <sup>(1)</sup>	MCLRE	CP1	CP0	PWRTE	WDTE	FOSC2	FOSC1	FOSC0
81h	OPTION	$\overline{GPPU}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0

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

Note 1: See Figure 8-1 for operation of these bits. Not all `CP0` and `CP1` bits are shown.

# PIC12C67X

---

## 8.8 Power-down Mode (SLEEP)

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 (STATUS<3>) is cleared, the  $\overline{TO}$  (STATUS<4>) bit is set, and the oscillator driver is turned off. The I/O ports maintain the status they had, before the `SLEEP` instruction was executed (driving high, low, or hi-impedance).

For lowest current consumption in this mode, place all I/O pins at either  $V_{DD}$ , or  $V_{SS}$ , ensure no external circuitry is drawing current from the I/O pin, power-down the A/D, disable external clocks. Pull all I/O pins, that are hi-impedance inputs, high or low externally to avoid switching currents caused by floating inputs. The  $\overline{TOCKI}$  input if enabled should also be at  $V_{DD}$  or  $V_{SS}$  for lowest current consumption. The contribution from on-chip pull-ups on GPIO should be considered.

The  $\overline{MCLR}$  pin if enabled must be at a logic high level (VIHMC).

### 8.8.1 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. GP2/INT interrupt, interrupt GPIO port change, or some Peripheral Interrupts.

External  $\overline{MCLR}$  Reset will cause a device reset. All other events are considered a continuation of program execution and cause a "wake-up". The  $\overline{TO}$  and  $\overline{PD}$  bits in the STATUS register can be used to determine the cause of device reset. The  $\overline{PD}$  bit, which is set on power-up, is cleared when `SLEEP` is invoked. The  $\overline{TO}$  bit is cleared if a WDT time-out occurred (and caused wake-up).

The following peripheral interrupt can wake the device from `SLEEP`:

1. A/D conversion (when A/D clock source is RC).

Other peripherals can not generate interrupts since during `SLEEP`, no on-chip Q clocks are present.

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 a `NOP` after the `SLEEP` instruction.

### 8.8.2 WAKE-UP USING INTERRUPTS

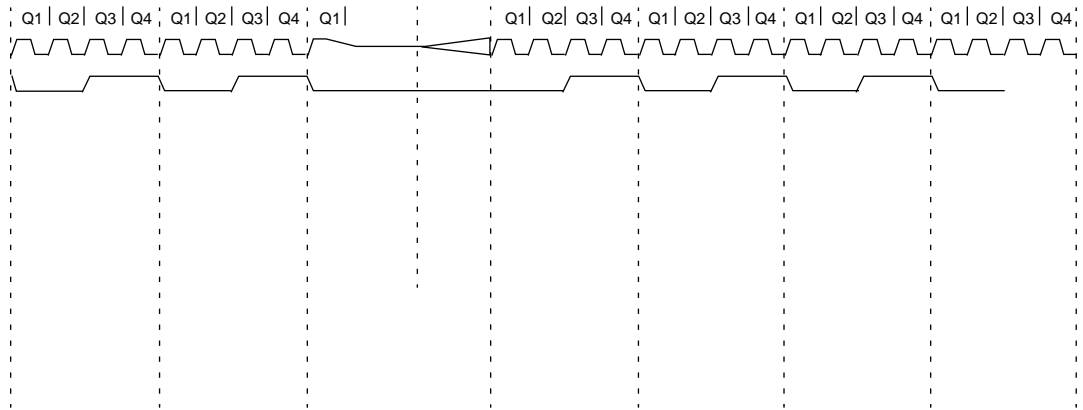
When global interrupts are disabled (GIE cleared) and any interrupt source has both its interrupt enable bit and interrupt flag bit set, one of the following will occur:

- If the interrupt occurs **before** the the execution of a `SLEEP` instruction, the `SLEEP` instruction will complete as a `NOP`. Therefore, the WDT and WDT postscaler will not be cleared, the  $\overline{TO}$  bit will not be set and  $\overline{PD}$  bits will not be cleared.
- If the interrupt occurs **during or after** the execution of a `SLEEP` instruction, the device will immediately wake up from sleep. The `SLEEP` instruction will be completely executed before the wake-up. Therefore, the WDT and WDT postscaler will be cleared, the  $\overline{TO}$  bit will be set and the  $\overline{PD}$  bit will be cleared.

Even if the flag bits were checked before executing a `SLEEP` instruction, it may be possible for flag bits to become set before the `SLEEP` instruction completes. To determine whether a `SLEEP` instruction executed, test the  $\overline{PD}$  bit. If the  $\overline{PD}$  bit is set, the `SLEEP` instruction was executed as a `NOP`.

To ensure that the WDT is cleared, a `CLRWDT` instruction should be executed before a `SLEEP` instruction.

**FIGURE 8-18: WAKE-UP FROM SLEEP THROUGH INTERRUPT**



# PIC12C67X

---

NOTES:



## 9.0 INSTRUCTION SET SUMMARY

Each PIC12C67X 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 PIC12C67X instruction set summary in Table 9-2 lists **byte-oriented**, **bit-oriented**, and **literal and control** operations. Table 9-1 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 9-1: 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
T $\bar{O}$	Time-out bit
P $\bar{D}$	Power-down bit
dest	Destination either the W register or the specified register file location
[ ]	Options
( )	Contents
→	Assigned to
< >	Register bit field
∈	In the set of
<i>italics</i>	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  $\mu$ s. If a conditional test is true or the program counter is changed as a result of an instruction, the instruction execution time is 2  $\mu$ s.

Table 9-2 lists the instructions recognized by the MPASM assembler.

Figure 9-1 shows the three general formats that the instructions can have.

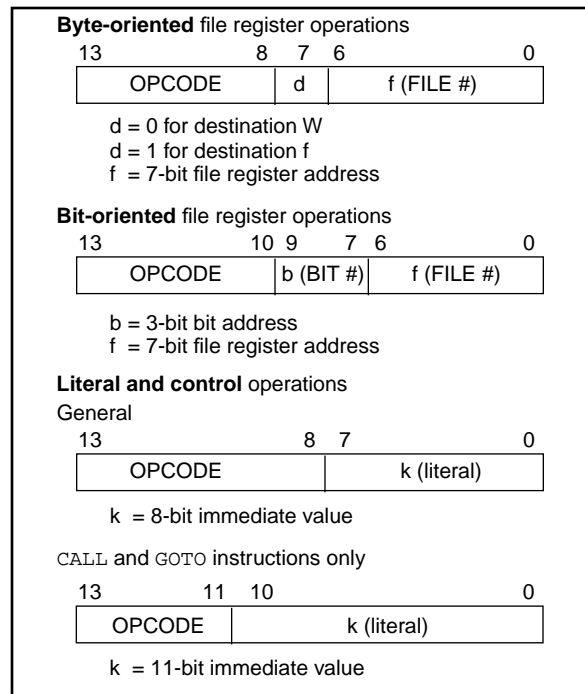
**Note:** To maintain upward compatibility with future PIC12C67X products, do not use the `OPTION` and `TRIS` instructions.

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

0xhh

where h signifies a hexadecimal digit.

**FIGURE 9-1: GENERAL FORMAT FOR INSTRUCTIONS**



# PIC12C67X

---

## 9.1 Special Function Registers as Source/Destination

The PIC12C67X's orthogonal instruction set allows read and write of all file registers, including special function registers. There are some special situations the user should be aware of:

### 9.1.1 STATUS AS DESTINATION

If an instruction writes to STATUS, the Z, C and DC bits may be set or cleared as a result of the instruction and overwrite the original data bits written. For example, executing `CLRF STATUS` will clear register STATUS, and then set the Z bit leaving `0000 0100b` in the register.

### 9.1.2 TRIS AS DESTINATION

Bit 3 of the TRIS register always reads as a '1' since GP3 is an input only pin. This fact can affect some read-modify-write operations on the TRIS register.

### 9.1.3 PCL AS SOURCE OR DESTINATION

Read, write or read-modify-write on PCL may have the following results:

Read PC: PCL → dest  
Write PCL: PCLATH → PCH;  
8-bit destination value → PCL  
Read-Modify-Write: PCL → ALU operand  
PCLATH → PCH;  
8-bit result → PCL

Where PCH = program counter high byte (not an addressable register), PCLATH = Program counter high holding latch, dest = destination, WREG or f.

### 9.1.4 BIT MANIPULATION

All bit manipulation instructions are done by first reading the entire register, operating on the selected bit and writing the result back (read-modify-write). The user should keep this in mind when operating on special function registers, such as ports.

**TABLE 9-2: INSTRUCTION SET SUMMARY**

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

# PIC12C67X

## 9.2 Instruction Descriptions

<b>ADDLW</b>	<b>Add Literal and W</b>				
Syntax:	[ <i>label</i> ] ADDLW k				
Operands:	$0 \leq k \leq 255$				
Operation:	(W) + k → (W)				
Status Affected:	C, DC, Z				
Encoding:	<table border="1"> <tr> <td>11</td> <td>111x</td> <td>kkkk</td> <td>kkkk</td> </tr> </table>	11	111x	kkkk	kkkk
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	<pre>ADDLW 0x15</pre> <p>Before Instruction W = 0x10</p> <p>After Instruction W = 0x25</p>				

<b>ANDLW</b>	<b>And Literal with W</b>				
Syntax:	[ <i>label</i> ] ANDLW k				
Operands:	$0 \leq k \leq 255$				
Operation:	(W) .AND. (k) → (W)				
Status Affected:	Z				
Encoding:	<table border="1"> <tr> <td>11</td> <td>1001</td> <td>kkkk</td> <td>kkkk</td> </tr> </table>	11	1001	kkkk	kkkk
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	<pre>ANDLW 0x5F</pre> <p>Before Instruction W = 0xA3</p> <p>After Instruction W = 0x03</p>				

<b>ADDWF</b>	<b>Add W and f</b>				
Syntax:	[ <i>label</i> ] ADDWF f,d				
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$				
Operation:	(W) + (f) → (dest)				
Status Affected:	C, DC, Z				
Encoding:	<table border="1"> <tr> <td>00</td> <td>0111</td> <td>dfff</td> <td>ffff</td> </tr> </table>	00	0111	dfff	ffff
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	<pre>ADDWF FSR, 0</pre> <p>Before Instruction W = 0x17 FSR = 0xC2</p> <p>After Instruction W = 0xD9 FSR = 0xC2</p>				

<b>ANDWF</b>	<b>AND W with f</b>				
Syntax:	[ <i>label</i> ] ANDWF f,d				
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$				
Operation:	(W) .AND. (f) → (dest)				
Status Affected:	Z				
Encoding:	<table border="1"> <tr> <td>00</td> <td>0101</td> <td>dfff</td> <td>ffff</td> </tr> </table>	00	0101	dfff	ffff
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	<pre>ANDWF FSR, 1</pre> <p>Before Instruction W = 0x17 FSR = 0xC2</p> <p>After Instruction W = 0x17 FSR = 0x02</p>				

**BCF**                    **Bit Clear f**

---

Syntax:                [ *label* ] BCF    f,b

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

Operation:             $0 \rightarrow (f<b>)$

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
    
```

**BTFSC**                **Bit Test, Skip if Clear**

---

Syntax:                [ *label* ] BTFSC   f,b

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

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 2 cycle instruction.

Words:                1

Cycles:               1(2)

Example

```

HERE       BTFSC   FLAG, 1
FALSE      GOTO    PROCESS_CODE
TRUE       .
             .
    
```

```

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

**BSF**                    **Bit Set f**

---

Syntax:                [ *label* ] BSF    f,b

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

Operation:             $1 \rightarrow (f<b>)$

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
    
```

# PIC12C67X

## **BTFS** Bit Test f, Skip if Set

Syntax: [ *label* ] BTFS f,b

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

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 2 cycle instruction.

Words: 1

Cycles: 1(2)

Example

```

    HERE    BTFS    FLAG,1
    FALSE   GOTO   PROCESS_CODE
    TRUE    .
           .
           .
    
```

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:  $0 \leq k \leq 2047$

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   THERE
    
```

Before Instruction  
 PC = Address HERE

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

## **CLRF** Clear f

Syntax: [ *label* ] CLRF f

Operands:  $0 \leq f \leq 127$

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	0xxx	xxxx
----	------	------	------

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

**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

**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

**DECFSZ**      **Decrement f, Skip if 0**

Syntax:      `[ label ] DECFSZ f,d`

Operands:      `0 ≤ f ≤ 127`  
`d ∈ [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

# PIC12C67X

## GOTO Unconditional Branch

Syntax: [ *label* ] GOTO *k*  
 Operands:  $0 \leq k \leq 2047$   
 Operation:  $k \rightarrow PC<10:0>$   
            $PCLATH<4:3> \rightarrow PC<12:11>$   
 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

## INCFSZ Increment f, Skip if 0

Syntax: [ *label* ] INCFSZ *f,d*  
 Operands:  $0 \leq f \leq 127$   
            $d \in [0,1]$   
 Operation:  $(f) + 1 \rightarrow (\text{dest})$ , 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

## INCF Increment f

Syntax: [ *label* ] INCF *f,d*  
 Operands:  $0 \leq f \leq 127$   
            $d \in [0,1]$   
 Operation:  $(f) + 1 \rightarrow (\text{dest})$   
 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

## IORLW Inclusive OR Literal with W

Syntax: [ *label* ] IORLW *k*  
 Operands:  $0 \leq k \leq 255$   
 Operation:  $(W) .OR. k \rightarrow (W)$   
 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:                 $0 \leq f \leq 127$   
 $d \in [0,1]$

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

**MOVF**                    **Move f**

---

Syntax:                    [ *label* ] MOVF f,d

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

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

**MOVLW**                  **Move Literal to W**

---

Syntax:                    [ *label* ] MOVLW k

Operands:                 $0 \leq k \leq 255$

Operation:                 $k \rightarrow (W)$

Status Affected:        None

Encoding:                

11	00xx	kkkk	kkkk
----	------	------	------

Description:             The eight bit literal 'k' is loaded into W register. The don't cares will assemble as 0's.

Words:                    1

Cycles:                    1

Example                    MOVLW              0x5A

After Instruction

W = 0x5A

**MOVWF**                  **Move W to f**

---

Syntax:                    [ *label* ] MOVWF f

Operands:                 $0 \leq f \leq 127$

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

# PIC12C67X

## NOP No Operation

Syntax:	[ <i>label</i> ] NOP				
Operands:	None				
Operation:	No operation				
Status Affected:	None				
Encoding:	<table border="1"> <tr> <td>00</td> <td>0000</td> <td>0xx0</td> <td>0000</td> </tr> </table>	00	0000	0xx0	0000
00	0000	0xx0	0000		
Description:	No operation.				
Words:	1				
Cycles:	1				
Example	NOP				

## RETFIE Return from Interrupt

Syntax:	[ <i>label</i> ] RETFIE				
Operands:	None				
Operation:	TOS → PC, 1 → GIE				
Status Affected:	None				
Encoding:	<table border="1"> <tr> <td>00</td> <td>0000</td> <td>0000</td> <td>1001</td> </tr> </table>	00	0000	0000	1001
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	<pre>RETFIE</pre> <p>After Interrupt</p> <pre>PC = TOS GIE = 1</pre>				

OPTION	Load Option Register				
Syntax:	[ <i>label</i> ] OPTION				
Operands:	None				
Operation:	(W) → OPTION				
Status Affected:	None				
Encoding:	<table border="1"> <tr> <td>00</td> <td>0000</td> <td>0110</td> <td>0010</td> </tr> </table>	00	0000	0110	0010
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	<p><b>To maintain upward compatibility with future PIC12C67X products, do not use this instruction.</b></p>				

## RETLW Return with Literal in W

Syntax:	[ <i>label</i> ] RETLW k				
Operands:	0 ≤ k ≤ 255				
Operation:	k → (W); TOS → PC				
Status Affected:	None				
Encoding:	<table border="1"> <tr> <td>11</td> <td>01xx</td> <td>kkkk</td> <td>kkkk</td> </tr> </table>	11	01xx	kkkk	kkkk
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	<pre>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</pre> <p>Before Instruction</p> <pre>W = 0x07</pre> <p>After Instruction</p> <pre>W = value of k8</pre>				

## RETURN Return from Subroutine

Syntax: [ *label* ] RETURN  
 Operands: None  
 Operation: TOS → PC  
 Status Affected: None  
 Encoding: 

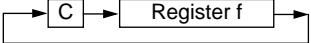
00	0000	0000	1000
----	------	------	------

  
 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

## RRF Rotate Right f through Carry

Syntax: [ *label* ] RRF f,d  
 Operands:  $0 \leq f \leq 127$   
 $d \in [0,1]$   
 Operation: See description below  
 Status Affected: C  
 Encoding: 

00	1100	dfff	ffff
----	------	------	------

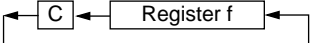
  
 Description: The contents of register 'f' are rotated one bit to the right through the Carry Flag. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is placed back in register 'f'.  
  
 Words: 1  
 Cycles: 1  
 Example: RRF REG1,0

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

## RLF Rotate Left f through Carry

Syntax: [ *label* ] RLF f,d  
 Operands:  $0 \leq f \leq 127$   
 $d \in [0,1]$   
 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

## 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.  
 Words: 1  
 Cycles: 1  
 Example: SLEEP

# PIC12C67X

## SUBLW Subtract W from Literal

Syntax: [ *label* ] SUBLW *k*

Operands:  $0 \leq k \leq 255$

Operation:  $k - (W) \rightarrow (W)$

Status: C, DC, Z

Affected:

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

Operation:  $(f) - (W) \rightarrow (\text{dest})$

Status: C, DC, Z

Affected:

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: SUBWF REG1, 1

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:         $0 \leq f \leq 127$   
                     $d \in [0,1]$   
Operation:        (f<3:0>)  $\rightarrow$  (dest<7:4>),  
                    (f<7:4>)  $\rightarrow$  (dest<3:0>)  
Status Affected: None  
Encoding:         00

# PIC12C67X

---

NOTES:

## 10.0 DEVELOPMENT SUPPORT

### 10.1 Development Tools

The PICmicro™ microcontrollers are supported with a full range of hardware and software development tools:

- PICMASTER/PICMASTER CE Real-Time In-Circuit Emulator
- ICEPIC Low-Cost PIC16C5X and PIC16CXXX In-Circuit Emulator
- PRO MATE® II Universal Programmer
- PICSTART® Plus Entry-Level Prototype Programmer
- PICDEM-1 Low-Cost Demonstration Board
- PICDEM-2 Low-Cost Demonstration Board
- PICDEM-3 Low-Cost Demonstration Board
- MPASM Assembler
- MPLAB™ SIM Software Simulator
- MPLAB-C (C Compiler)
- Fuzzy Logic Development System (*fuzzyTECH*®-MP)

### 10.2 PICMASTER: High Performance Universal In-Circuit Emulator with MPLAB IDE

The PICMASTER Universal In-Circuit Emulator is intended to provide the product development engineer with a complete microcontroller design tool set for all microcontrollers in the PIC12CXXX, PIC14C00, PIC16C5X, PIC16CXXX and PIC17CXX families. PICMASTER is supplied with the MPLAB™ Integrated Development Environment (IDE), which allows editing, "make" and download, and source debugging from a single environment.

Interchangeable target probes allow the system to be easily reconfigured for emulation of different processors. The universal architecture of the PICMASTER allows expansion to support all new Microchip microcontrollers.

The PICMASTER Emulator System has been designed as a real-time emulation system with advanced features that are generally found on more expensive development tools. The PC compatible 386 (and higher) machine platform and Microsoft Windows® 3.x environment were chosen to best make these features available to you, the end user.

A CE compliant version of PICMASTER is available for European Union (EU) countries.

### 10.3 ICEPIC: Low-Cost PIC16CXXX In-Circuit Emulator

ICEPIC is a low-cost in-circuit emulator solution for the Microchip PIC16C5X and PIC16CXXX families of 8-bit OTP microcontrollers.

ICEPIC is designed to operate on PC-compatible machines ranging from 286-AT® through Pentium™ based machines under Windows 3.x environment. ICEPIC features real time, non-intrusive emulation.

### 10.4 PRO MATE II: Universal Programmer

The PRO MATE II Universal Programmer is a full-featured programmer capable of operating in stand-alone mode as w

## 10.6 PICDEM-1 Low-Cost PIC16/17 Demonstration Board

The PICDEM-1 is a simple board which demonstrates the capabilities of several of Microchip's microcontrollers. The microcontrollers supported are: PIC16C5X (PIC16C54 to PIC16C58A), PIC16C61, PIC16C62X, PIC16C71, PIC16C8X, PIC17C42, PIC17C43 and PIC17C44. All necessary hardware and software is included to run basic demo programs. The users can program the sample microcontrollers provided with the PICDEM-1 board, on a PRO MATE II or PICSTART-Plus programmer, and easily test firmware. The user can also connect the PICDEM-1 board to the PICMASTER emulator and download the firmware to the emulator for testing. Additional prototype area is available for the user to build some additional hardware and connect it to the microcontroller socket(s). Some of the features include an RS-232 interface, a potentiometer for simulated analog input, push-button switches and eight LEDs connected to PORTB.

## 10.7 PICDEM-2 Low-Cost PIC16CXX Demonstration Board

The PICDEM-2 is a simple demonstration board that supports the PIC16C62, PIC16C64, PIC16C65, PIC16C73 and PIC16C74 microcontrollers. All the necessary hardware and software is included to run the basic demonstration programs. The user can program the sample microcontrollers provided with the PICDEM-2 board, on a PRO MATE II programmer or PICSTART-Plus, and easily test firmware. The PICMASTER emulator may also be used with the PICDEM-2 board to test firmware. Additional prototype area has been provided to the user for adding additional hardware and connecting it to the microcontroller socket(s). Some of the features include a RS-232 interface, push-button switches, a potentiometer for simulated analog input, a Serial EEPROM to demonstrate usage of the I<sup>2</sup>C bus and separate headers for connection to an LCD module and a keypad.

## 10.8 PICDEM-3 Low-Cost PIC16CXXX Demonstration Board

The PICDEM-3 is a simple demonstration board that supports the PIC16C923 and PIC16C924 in the PLCC package. It will also support future 44-pin PLCC microcontrollers with a LCD Module. All the necessary hardware and software is included to run the basic demonstration programs. The user can program the sample microcontrollers provided with the PICDEM-3 board, on a PRO MATE II programmer or PICSTART Plus with an adapter socket, and easily test firmware. The PICMASTER emulator may also be used with the PICDEM-3 board to test firmware. Additional prototype area has been provided to the user for adding hardware and connecting it to the microcontroller socket(s). Some of the features include

an RS-232 interface, push-button switches, a potentiometer for simulated analog input, a thermistor and separate headers for connection to an external LCD module and a keypad. Also provided on the PICDEM-3 board is an LCD panel, with 4 commons and 12 segments, that is capable of displaying time, temperature and day of the week. The PICDEM-3 provides an additional RS-232 interface and Windows 3.1 software for showing the demultiplexed LCD signals on a PC. A simple serial interface allows the user to construct a hardware demultiplexer for the LCD signals.

## 10.9 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:

- A full featured editor
- Three operating modes
  - editor
  - emulator
  - simulator
- A project manager
- Customizable tool bar and key mapping
- A status bar with project information
- Extensive on-line help

MPLAB allows you to:

- Edit your source files (either assembly or 'C')
- One touch assemble (or compile) and download to PIC16/17 tools (automatically updates all project information)
- Debug using:
  - source files
  - absolute listing file
- Transfer data dynamically via DDE (soon to be replaced by OLE)
- Run up to four emulators on the same PC

The ability to use MPLAB with Microchip's simulator allows a consistent platform and the ability to easily switch from the low cost simulator to the full featured emulator with minimal retraining due to development tools.

## 10.10 Assembler (MPASM)

The MPASM Universal Macro Assembler is a PC-hosted symbolic assembler. It supports all microcontroller series including the PIC12C5XX, PIC14000, PIC16C5X, PIC16CXXX, and PIC17CXX families.

MPASM offers full featured Macro capabilities, conditional assembly, and several source and listing formats. It generates various object code formats to support Microchip's development tools as well as third party programmers.

MPASM allows full symbolic debugging from PICMASTER, Microchip's Universal Emulator System.



MPASM has the following features to assist in developing software for specific use applications.

- Provides translation of Assembler source code to object code for all Microchip microcontrollers.
- Macro assembly capability.
- Produces all the files (Object, Listing, Symbol, and special) required for symbolic debug with Microchip's emulator systems.
- Supports Hex (default), Decimal and Octal source and listing formats.

MPASM provides a rich directive language to support programming of the PIC16/17. Directives are helpful in making the development of your assemble source code shorter and more maintainable.

## 10.11 Software Simulator (MPLAB-SIM)

The MPLAB-SIM Software Simulator allows code development in a PC host environment. It allows the user to simulate the PIC16/17 series microcontrollers on an instruction level. On any given instruction, the user may examine or modify any of the data areas or provide external stimulus to any of the pins. The input/output radix can be set by the user and the execution can be performed in; single step, execute until break, or in a trace mode.

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

## 10.12 C Compiler (MPLAB-C)

The MPLAB-C Code Development System is a complete 'C' compiler and integrated development environment for Microchip's PIC16/17 family of microcontrollers. The compiler provides powerful integration capabilities and ease of use not found with other compilers.

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

## 10.13 Fuzzy Logic Development System (fuzzyTECH-MP)

*fuzzyTECH-MP* fuzzy logic development tool is available in two versions - a low cost introductory version, MP Explorer, for designers to gain a comprehensive working knowledge of fuzzy logic system design; and a full-featured version, *fuzzyTECH-MP*, edition for implementing more complex systems.

Both versions include Microchip's *fuzzyLAB™* demonstration board for hands-on experience with fuzzy logic systems implementation.

## 10.14 MP-DriveWay™ – Application Code Generator

MP-DriveWay is an easy-to-use Windows-based Application Code Generator. With MP-DriveWay you can visually configure all the peripherals in a PIC16/17 device and, with a click of the mouse, generate all the initialization and many functional code modules in C language. The output is fully compatible with Microchip's MPLAB-C C compiler. The code produced is highly modular and allows easy integration of your own code. MP-DriveWay is intelligent enough to maintain your code through subsequent code generation.

## 10.15 SEEVAL® Evaluation and Programming System

The SEEVAL SEEPROM Designer's Kit supports all Microchip 2-wire and 3-wire Serial EEPROMs. The kit includes everything necessary to read, write, erase or program special features of any Microchip SEEPROM product including Smart Serials™ and secure serials. The Total Endurance™ Disk is included to aid in trade-off analysis and reliability calculations. The total kit can significantly reduce time-to-market and result in an optimized system.

## 10.16 KEELOQ® Evaluation and Programming Tools

KEELOQ evaluation and programming tools support Microchips HCS Secure Data Products. The HCS evaluation kit includes an LCD display to show changing codes, a decoder to decode transmissions, and a programming interface to program test transmitters.

# PIC12C67X

TABLE 10-1: DEVELOPMENT TOOLS FROM MICROCHIP

	PIC12C5XX	PIC14000	PIC16C5X	PIC16CXXX	PIC16C6X	PIC16C7XX	PIC16C8X	PIC16C9XX	PIC17C4X	PIC17C75X	24CXX 25CXX 93CXX	HCS200 HCS300 HCS301
Emulator Products	PICMASTER <sup>®</sup> / PICMASTER-CE In-Circuit Emulator	✓	✓	✓	✓	✓	✓	✓	✓	Available 3097		
	ICEPIC Low-Cost In-Circuit Emulator	✓	✓	✓	✓	✓	✓		✓			
Software Tools	MPLAB <sup>™</sup> Integrated Development Environment		✓	✓	✓	✓	✓	✓	✓	✓		
	MPLAB <sup>™</sup> C Compiler	✓	✓	✓	✓	✓	✓	✓	✓	✓		
	fuzzyTECH <sup>®</sup> -MP Explorer/Edition Fuzzy Logic Dev. Tool	✓	✓	✓	✓	✓	✓	✓	✓			
	MP-DriveWay <sup>™</sup> Applications Code Generator	✓	✓	✓	✓	✓	✓	✓	✓	✓		
	Total Endurance <sup>™</sup> Software Model										✓	
Programmers	PICSTART <sup>®</sup> Lite Ultra Low-Cost Dev. Kit		✓			✓	✓					
	PICSTART <sup>®</sup> Plus Low-Cost Universal Dev. Kit	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	PRO MATE <sup>®</sup> II Universal Programmer	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	KEELOQ <sup>®</sup> Programmer											
	SEEVAL <sup>®</sup> Designers Kit										✓	
Demo Boards	PICDEM-1		✓	✓			✓		✓			
	PICDEM-2					✓						
	PICDEM-3							✓				
KEELOQ <sup>®</sup> Evaluation Kit											✓	

## 11.0 ELECTRICAL CHARACTERISTICS FOR PIC12C67X

### Absolute Maximum Ratings †

Ambient temperature under bias .....	-40° to +125°C
Storage temperature .....	-65°C to +150°C
Voltage on any pin with respect to VSS (except VDD and MCLR).....	-0.3V to (VDD + 0.3V)
Voltage on VDD with respect to VSS .....	0 to +7.5V
Voltage on MCLR with respect to VSS (Note 2).....	0 to +14V
Total power dissipation (Note 1).....	700 mW
Maximum current out of VSS pin .....	200 mA
Maximum current into VDD pin .....	150 mA
Input clamp current, I <sub>IK</sub> (V <sub>I</sub> < 0 or V <sub>I</sub> > VDD).....	± 20 mA
Output clamp current, I <sub>OK</sub> (V <sub>O</sub> < 0 or V <sub>O</sub> > VDD).....	± 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 GPIO pins combined.....	100 mA
Maximum current sourced by GPIO pins combined.....	100 mA

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

† 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.

**TABLE 11-1: CROSS REFERENCE OF DEVICE SPECS FOR OSCILLATOR CONFIGURATIONS AND FREQUENCIES OF OPERATION (COMMERCIAL DEVICES)**

OSC	PIC12C671-04 PIC12C672-04	PIC12C671-10 PIC12C672-10	PIC12LC671-04 PIC12LC672-04	PIC12C671/JW PIC12C672/JW
INTRC	VDD: 3.0V to 5.5V IDD: 5 mA max. at 5.5V IPD: 21 $\mu$ A max. at 4V Freq: 4 MHz max.	VDD: 3.0V to 5.5V IDD: 2.7 mA typ. at 5.5V IPD: 1.5 $\mu$ A typ. at 4V Freq: 4 MHz max.	VDD: 2.5V to 5.5V IDD: 2.0 mA typ. at 2.5V IPD: 0.9 $\mu$ A typ. at 2.5V Freq: 4 MHz max.	VDD: 3.0V to 5.5V IDD: 5 mA max. at 5.5V IPD: 21 $\mu$ A max. at 4V Freq: 4 MHz max.
EXTRC	VDD: 3.0V to 5.5V IDD: 5 mA max. at 5.5V IPD: 21 $\mu$ A max. at 4V Freq: 4 MHz max.	VDD: 3.0V to 5.5V IDD: 2.7 mA typ. at 5.5V IPD: 1.5 $\mu$ A typ. at 4V Freq: 4 MHz max.	VDD: 2.5V to 5.5V IDD: 2.0 mA typ. at 2.5V IPD: 0.9 $\mu$ A typ. at 2.5V Freq: 4 MHz max.	VDD: 3.0V to 5.5V IDD: 5 mA max. at 5.5V IPD: 21 $\mu$ A max. at 4V Freq: 4 MHz max.
XT	VDD: 3.0V to 5.5V IDD: 5 mA max. at 5.5V IPD: 21 $\mu$ A max. at 4V Freq: 4 MHz max.	VDD: 3.0V to 5.5V IDD: 2.7 mA typ. at 5.5V IPD: 1.5 $\mu$ A typ. at 4V Freq: 4 MHz max.	VDD: 2.5V to 5.5V IDD: 2.0 mA typ. at 2.5V IPD: 0.9 $\mu$ A typ. at 2.5V Freq: 4 MHz max.	VDD: 3.0V to 5.5V IDD: 5 mA max. at 5.5V IPD: 21 $\mu$ A max. at 4V Freq: 4 MHz max.
HS	VDD: 4.5V to 5.5V IDD: 13.5 mA typ. at 5.5V IPD: 1.5 $\mu$ A typ. at 4.5V Freq: 4 MHz max.	VDD: 4.5V to 5.5V IDD: 30 mA max. at 5.5V IPD: 1.5 $\mu$ A typ. at 4.5V Freq: 10 MHz max.	Do not use in HS mode	VDD: 3.0V to 5.5V IDD: 30 mA max. at 5.5V IPD: 1.5 $\mu$ A typ. at 4.5V Freq: 10 MHz max.
LP	VDD: 3.0V to 5.5V IDD: 52.5 $\mu$ A typ. at 32 kHz, 4.0V IPD: 0.9 $\mu$ A typ. at 4.0V Freq: 200 kHz max.	Do not use in LP mode	VDD: 2.5V to 5.5V IDD: 48 $\mu$ A max. at 32 kHz, 2.5V IPD: 5.0 $\mu$ A max. at 2.5V Freq: 200 kHz max.	VDD: 2.5V to 5.5V IDD: 48 $\mu$ A max. at 32 kHz, 2.5V IPD: 5.0 $\mu$ A max. at 2.5V Freq: 200 kHz max.

The shaded sections indicate oscillator selections which are tested for functionality, but not for MIN/MAX specifications. It is recommended that the user select the device type that ensures the specifications required.

**11.1 DC Characteristics:** PIC12C671-04 (Commercial, Industrial, Extended)<sup>(5)</sup>  
 PIC12C671-10 (Commercial, Industrial, Extended)<sup>(5)</sup>  
 PIC12C672-04 (Commercial, Industrial, Extended)<sup>(5)</sup>  
 PIC12C672-10 (Commercial, Industrial, Extended)<sup>(5)</sup>

DC CHARACTERISTICS		Standard Operating Conditions (unless otherwise specified)					
		Operating temperature 0°C ≤ TA ≤ +70°C (commercial) -40°C ≤ TA ≤ +85°C (industrial) -40°C ≤ TA ≤ +125°C (extended)					
Param. No.	Characteristic	Sym	Min	Typ†	Max	Units	Conditions
D001 D001A	Supply Voltage	VDD	3.0 4.5	- -	5.5 5.5	V V	XT, INTRC, EXTRC and LP osc configuration HS osc configuration
D002*	RAM Data Retention Voltage (Note 1)	VDR	-	1.5	-	V	Device in SLEEP mode
D003	VDD start voltage to ensure internal Power-on Reset signal	VPOR	-	VSS	-	V	See section on Power-on Reset for details
D004*	VDD rise rate to ensure internal Power-on Reset signal	SVDD	0.05	-	-	V/ms	See section on Power-on Reset for details
D010 D010C D013	Supply Current (Note 2)	IDD	-	2.7 7.7 13.5	5 5 30	mA mA mA	XT, EXTRC osc configuration (PIC12C67X-04) FOSC = 4 MHz, VDD = 5.5V (Note 4) INTRC osc configuration FOSC = 4 MHz, VDD = 5.5V HS osc configuration (PIC12C67X-10) FOSC = 20 MHz, VDD = 5.5V
D020 D021 D021A D021B	Power-down Current (Note 3)	IPD	-	5.5 1.5 1.5 1.5	-	μA μA μA μA	VDD = 4.0V, WDT enabled, -40°C to +85°C VDD = 4.0V, WDT disabled, 0°C to +70°C VDD = 4.0V, WDT disabled, -40°C to +85°C VDD = 4.0V, WDT disabled, -40°C to +125°C

\* These parameters are characterized but not tested.

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

2: 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 tristated, pulled to VDD

MCLR = VDD; WDT enabled/disabled as specified.

3: 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 and VSS.

4: For EXTRC osc configurations, current through Rext is not included. The current through the resistor can be estimated by the formula  $I_r = V_{DD}/2R_{ext}$  (mA) with Rext in kOhm.

5: **Extended operating range is Advance Information for this device.**

# PIC12C67X

## 11.2 DC Characteristics: PIC12LC671-04 (Commercial, Industrial) PIC12LC672-04 (Commercial, Industrial)

DC CHARACTERISTICS		Standard Operating Conditions (unless otherwise specified)						
		Operating temperature 0°C ≤ TA ≤ +70°C (commercial) -40°C ≤ TA ≤ +85°C (industrial)						
Param No.	Characteristic	Sym	Min	Typ†	Max	Units	Conditions	
D001	Supply Voltage	VDD	2.5	-	5.5	V	XT, INTRC, EXTRC and LP osc configuration (DC - 4 MHz)	
D002*	RAM Data Retention Voltage (Note 1)	VDR	-	1.5	-	V	Device in SLEEP mode	
D003	VDD start voltage to ensure internal Power-on Reset signal	VPOR	-	VSS	-	V	See section on Power-on Reset for details	
D004*	VDD rise rate to ensure internal Power-on Reset signal	SVDD	0.05	-	-	V/ms	See section on Power-on Reset for details	
D010	Supply Current (Note 2)	IDD	-	2.0	3.8	mA	XT, EXTRC osc configuration FOSC = 4 MHz, VDD = 3.0V (Note 4)	
D010B				2.0	3.8	mA		INTRC osc configuration FOSC = 4 MHz, VDD = 3.0V
D010A				22.5	48	μA		LP osc configuration FOSC = 32 kHz, VDD = 3.0V, WDT disabled
D020	Power-down Current (Note 3)	IPD	-	5.5	-	μA	VDD = 3.0V, WDT enabled, -40°C to +85°C	
D021				0.9	-	μA	VDD = 3.0V, WDT disabled, 0°C to +70°C	
D021A				0.9	-	μA	VDD = 3.0V, WDT disabled, -40°C to +85°C	

\* These parameters are characterized but not tested.

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

2: 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 tristated, pulled to VDD

MCLR = VDD; WDT enabled/disabled as specified.

3: 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 and VSS.

4: For EXTRC osc configuration, current through Rext is not included. The current through the resistor can be estimated by the formula  $I_r = V_{DD}/2R_{ext}$  (mA) with Rext in kOhm.

- 11.3 DC Characteristics: PIC12C671-04 (Commercial, Industrial, Extended)<sup>(4)</sup>  
 PIC12C671-10 (Commercial, Industrial, Extended)<sup>(4)</sup>  
 PIC12C672-04 (Commercial, Industrial, Extended)<sup>(4)</sup>  
 PIC12C672-10 (Commercial, Industrial, Extended)<sup>(4)</sup>  
 PIC12LC671-04 (Commercial, Industrial)  
 PIC12LC672-04 (Commercial, Industrial)

Standard Operating Conditions (unless otherwise specified)							
DC CHARACTERISTICS							
Operating temperature 0°C ≤ TA ≤ +70°C (commercial) -40°C ≤ TA ≤ +85°C (industrial) -40°C ≤ TA ≤ +125°C (extended)							
Operating voltage VDD range as described in DC spec Section 11.1 and Section 11.2.							
Param No.	Characteristic	Sym	Min	Typ †	Max	Units	Conditions
D030	<b>Input Low Voltage</b> I/O ports with TTL buffer	VIL	VSS	-	0.5V	V	
D031	with Schmitt Trigger buffer		VSS	-	0.2VDD	V	
D032	MCLR, GP2/T0CKI/AN2/INT (in RC mode)		VSS	-	0.2VDD	V	
D033	OSC1 (in XT, HS and LP)		VSS	-	0.3VDD	V	Note1
D040	<b>Input High Voltage</b> I/O ports with TTL buffer	VIH	2.0	-	VDD	V	4.5 ≤ VDD ≤ 5.5V
D040A			0.8VDD	-	VDD	V	For VDD > 5.5V or VDD < 4.5V
D041	with Schmitt Trigger buffer		0.8VDD	-	VDD	V	For entire VDD range
D042	MCLR, GP2/T0CKI/AN2/INT		0.8VDD	-	VDD	V	
D042A	OSC1 (XT, HS and LP)		0.7VDD	-	VDD	V	Note1
D043	OSC1 (in EXTRC mode)		0.9VDD	-	VDD	V	
D070	GPIO weak pull-up current	IPUR	50	250	400	µA	VDD = 5V, VPIN = VSS
D060	<b>Input Leakage Current</b> (Notes 2, 3) I/O ports	IIL	-	-	±1	µA	VSS ≤ VPIN ≤ VDD, Pin at hi-impedance
D061	MCLR, RA4/T0CKI		-	-	±5	µA	VSS ≤ VPIN ≤ VDD
D063	OSC1		-	-	±5	µA	VSS ≤ VPIN ≤ VDD, XT, HS and LP osc configuration
D080	<b>Output Low Voltage</b> I/O ports	VOL	-	-	0.6	V	IOL = 8.5 mA, VDD = 4.5V, -40°C to +85°C
D080A			-	-	0.6	V	IOL = 7.0 mA, VDD = 4.5V, -40°C to +125°C
D083	OSC2/CLKOUT		-	-	0.6	V	IOL = 1.6 mA, VDD = 4.5V, -40°C to +85°C
D083A			-	-	0.6	V	IOL = 1.2 mA, VDD = 4.5V, -40°C to +125°C

† 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: In EXTRC oscillator configuration, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended that the PIC12C67X be driven with external clock in RC mode.
- 2: The leakage current on the MCLR pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.
- 3: Negative current is defined as coming out of the pin.
- 4: **Extended operating range is Advance Information for this device.**

# PIC12C67X

Standard Operating Conditions (unless otherwise specified)							
<b>DC CHARACTERISTICS</b> Operating temperature $0^{\circ}\text{C} \leq T_A \leq +70^{\circ}\text{C}$ (commercial) $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ (industrial) $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ (extended) Operating voltage $V_{DD}$ range as described in DC spec Section 11.1 and Section 11.2.							
Param No.	Characteristic	Sym	Min	Typ †	Max	Units	Conditions
D090	<b>Output High Voltage</b> I/O ports (Note 3)	$V_{OH}$	$V_{DD} - 0.7$	-	-	V	$I_{OH} = -3.0\text{ mA}$ , $V_{DD} = 4.5\text{V}$ , $-40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$
D090A			$V_{DD} - 0.7$	-	-	V	$I_{OH} = -2.5\text{ mA}$ , $V_{DD} = 4.5\text{V}$ , $-40^{\circ}\text{C}$ to $+125^{\circ}\text{C}$
D092	OSC2/CLKOUT		$V_{DD} - 0.7$	-	-	V	$I_{OH} = -1.3\text{ mA}$ , $V_{DD} = 4.5\text{V}$ , $-40^{\circ}\text{C}$ to $+85^{\circ}\text{C}$
D092A			$V_{DD} - 0.7$	-	-	V	$I_{OH} = -1.0\text{ mA}$ , $V_{DD} = 4.5\text{V}$ , $-40^{\circ}\text{C}$ to $+125^{\circ}\text{C}$
D100	<b>Capacitive Loading Specs on Output Pins</b> OSC2 pin	$C_{osc2}$	-	-	15	pF	In XT, HS and LP modes when external clock is used to drive OSC1.
D101	All I/O pins and OSC2	$C_{IO}$	-	-	50	pF	

† 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: In EXTRC oscillator configuration, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended that the PIC12C67X be driven with external clock in RC mode.
- 2: The leakage current on the MCLR pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.
- 3: Negative current is defined as coming out of the pin.
- 4: **Extended operating range is Advance Information for this device.**

PRELIMINARY



## 11.4 Timing Parameter Symbology

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

1. TppS2ppS
2. TppS
3. TCC:ST (I<sup>2</sup>C specifications only)
4. Ts (I<sup>2</sup>C specifications only)

<b>T</b>			
F	Frequency	T	Time

Lowercase letters (pp) and their meanings:

<b>pp</b>			
cc	CCP1	osc	OSC1
ck	CLKOUT	rd	RD
cs	CS	rw	RD or WR
di	SDI	sc	SCK
do	SDO	ss	SS
dt	Data in	t0	T0CKI
io	I/O port	t1	T1CKI
mc	MCLR	wr	WR

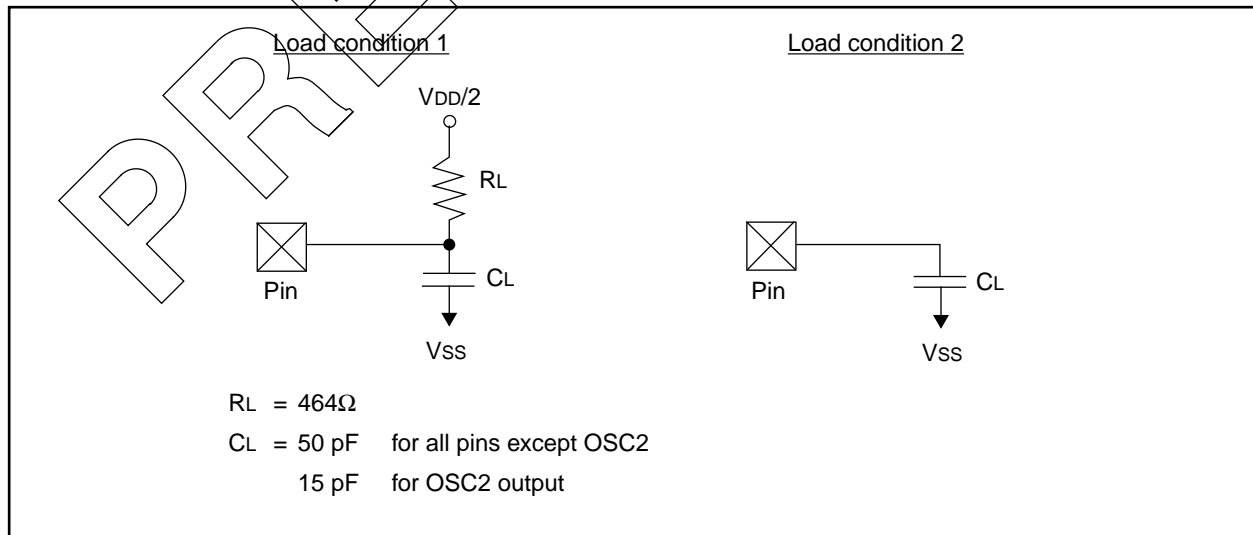
Uppercase letters and their meanings:

<b>S</b>			
F	Fall	P	Period
H	High	R	Rise
I	Invalid (Hi-impedance)	V	Valid
L	Low	Z	Hi-impedance
<b>I<sup>2</sup>C only</b>		High	High
AA	output access	Low	Low
BUF	Bus free		

TCC:ST (I<sup>2</sup>C specifications only)

<b>CC</b>			
HD	Hold	SU	Setup
<b>ST</b>			
DAT	DATA input hold	STO	STOP condition
STA	START condition		

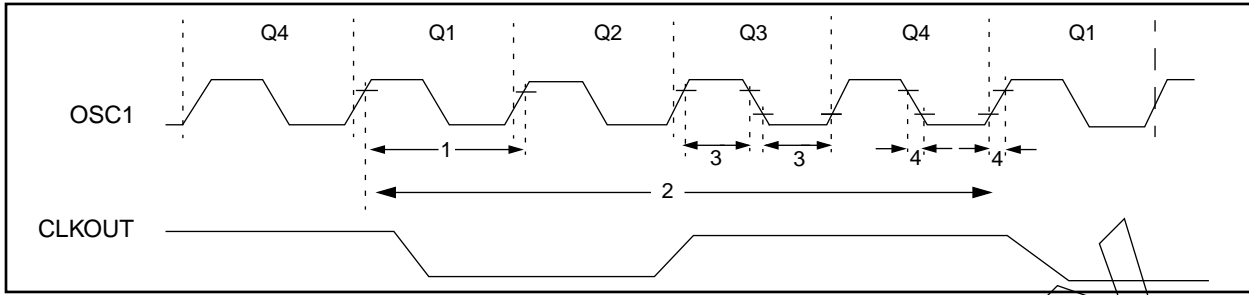
**FIGURE 11-1: LOAD CONDITIONS**



# PIC12C67X

## 11.5 Timing Diagrams and Specifications

**FIGURE 11-2: EXTERNAL CLOCK TIMING**



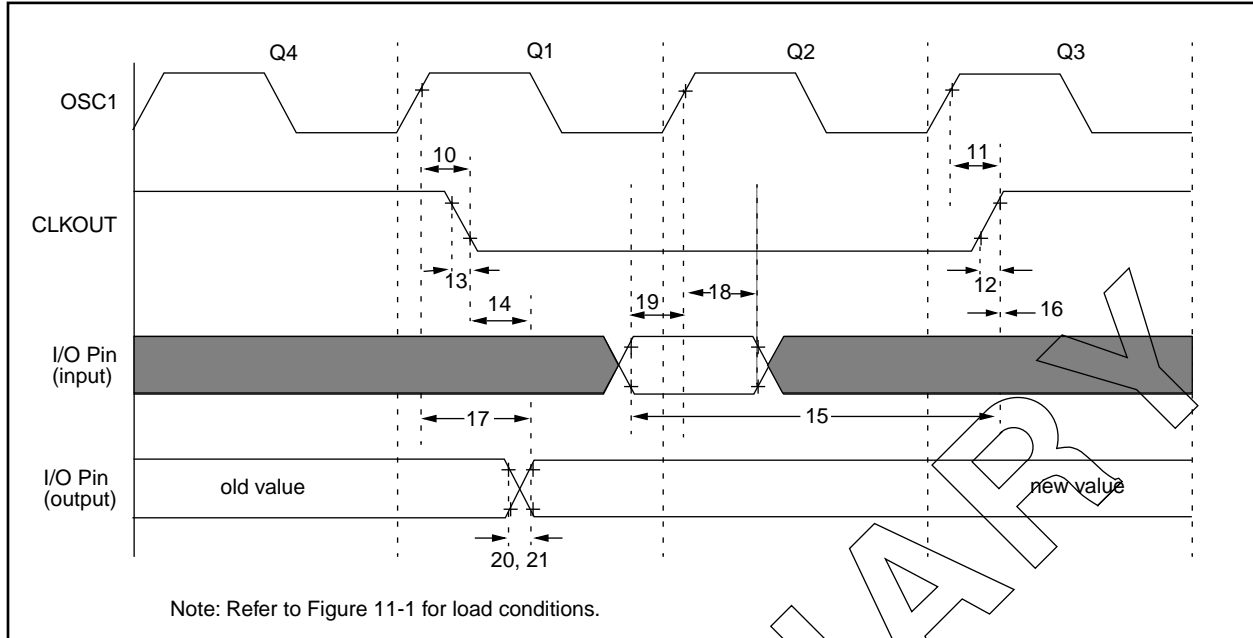
**TABLE 11-2: CLOCK TIMING REQUIREMENTS**

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
	Fos	<b>External CLKIN Frequency (Note 1)</b>	DC	—	4	MHz	XT and EXTRC osc mode
			DC	—	4	MHz	HS osc mode (PIC12C67X-04)
			DC	—	200	kHz	LP osc mode
		<b>Oscillator Frequency (Note 1)</b>	DC	—	4	MHz	EXTRC osc mode
			.455	—	4	MHz	XT osc mode
1	Tosc	<b>External CLKIN Period (Note 1)</b>	4	—	4	MHz	HS osc mode (PIC12C67X-04)
			4	—	10	MHz	HS osc mode (PIC12C67X-10)
			5	—	200	kHz	LP osc mode
			250	—	—	ns	XT and EXTRC osc mode
		<b>Oscillator Period (Note 1)</b>	250	—	—	ns	EXTRC osc mode
250	—	10,000	ns	XT osc mode			
250	—	250	ns	HS osc mode (PIC12C67X-04)			
100	—	250	ns	HS osc mode (PIC12C67X-10)			
5	—	—	μs	LP osc mode			
2	Tcy	<b>Instruction Cycle Time (Note 1)</b>	200	—	DC	ns	Tcy = 4/FOSC
3	TosL, TosH	<b>External Clock in (OSC1) High or Low Time</b>	50	—	—	ns	XT oscillator
			2.5	—	—	μs	LP oscillator
			10	—	—	ns	HS oscillator
4	TosR, TosF	<b>External Clock in (OSC1) Rise or Fall Time</b>	—	—	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.

**FIGURE 11-3: CLKOUT AND I/O TIMING**



**TABLE 11-3: CLKOUT AND I/O TIMING REQUIREMENTS**

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
10*	TosH2ckL	OSC1↑ to CLKOUT↓	—	15	30	ns	Note 1
11*	TosH2ckH	OSC1↑ to CLKOUT↑	—	15	30	ns	Note 1
12*	TckR	CLKOUT rise time	—	5	15	ns	Note 1
13*	TckF	CLKOUT fall time	—	5	15	ns	Note 1
14*	TckL2ioV	CLKOUT ↓ to Port out valid	—	—	0.5T <sub>CY</sub> + 20	ns	Note 1
15*	TioV2ckH	Port in valid before CLKOUT ↑	0.25T <sub>CY</sub> + 25	—	—	ns	Note 1
16*	TckH2ioI	Port in hold after CLKOUT ↑	0	—	—	ns	Note 1
17*	TosH2ioV	OSC1↑ (Q1 cycle) to Port out valid	—	—	80 - 100	ns	
18*	TosH2ioI	OSC1↑ (Q2 cycle) to Port input invalid (I/O in hold time)	TBD	—	—	ns	
19*	TioV2osH	Port input valid to OSC1↑ (I/O in setup time)	TBD	—	—	ns	
20*	TioR	Port output rise time	—	10	25	ns	
		PIC12LC67X	—	—	60	ns	
21*	TioF	Port output fall time	—	10	25	ns	
		PIC12LC67X	—	—	60	ns	
22††*	Tihp	INT pin high or low time	20	—	—	ns	
23††*	Trbp	GPIO change INT high or low time	20	—	—	ns	

\* These parameters are characterized but not tested.

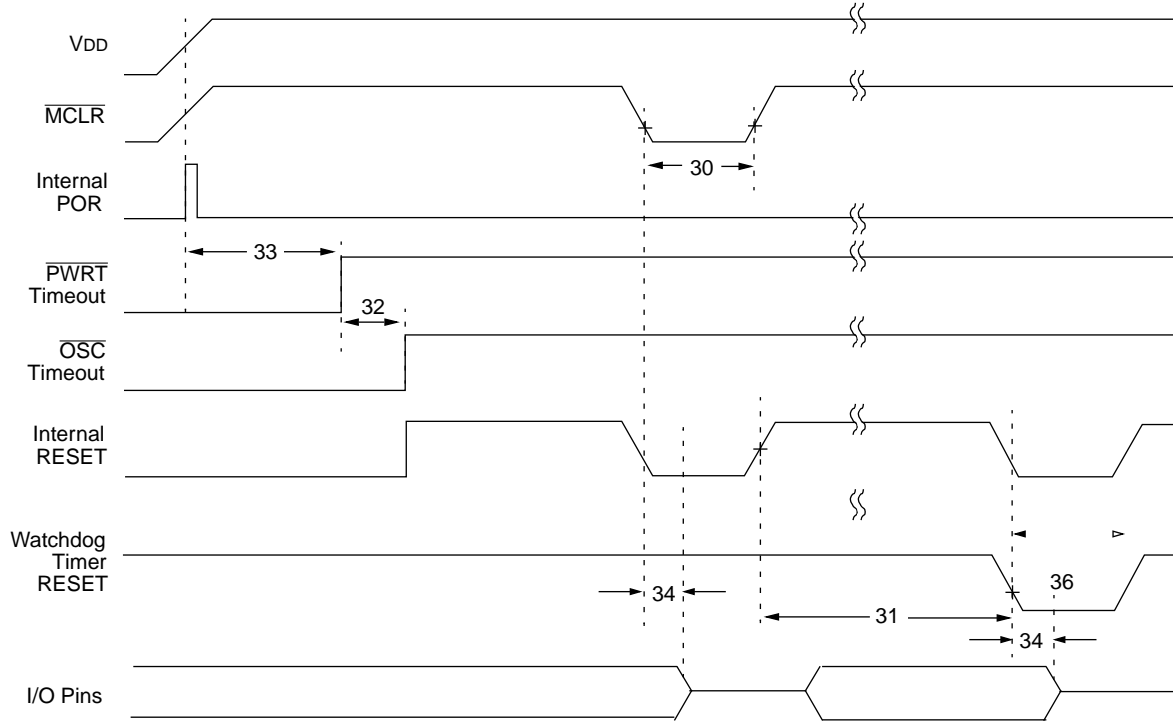
† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

†† These parameters are asynchronous events not related to any internal clock edges.

Note 1: Measurements are taken in EXTRC and INTRC modes where CLKOUT output is 4 x T<sub>osc</sub>.

# PIC12C67X

**FIGURE 11-4: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, AND POWER-UP TIMER TIMING**



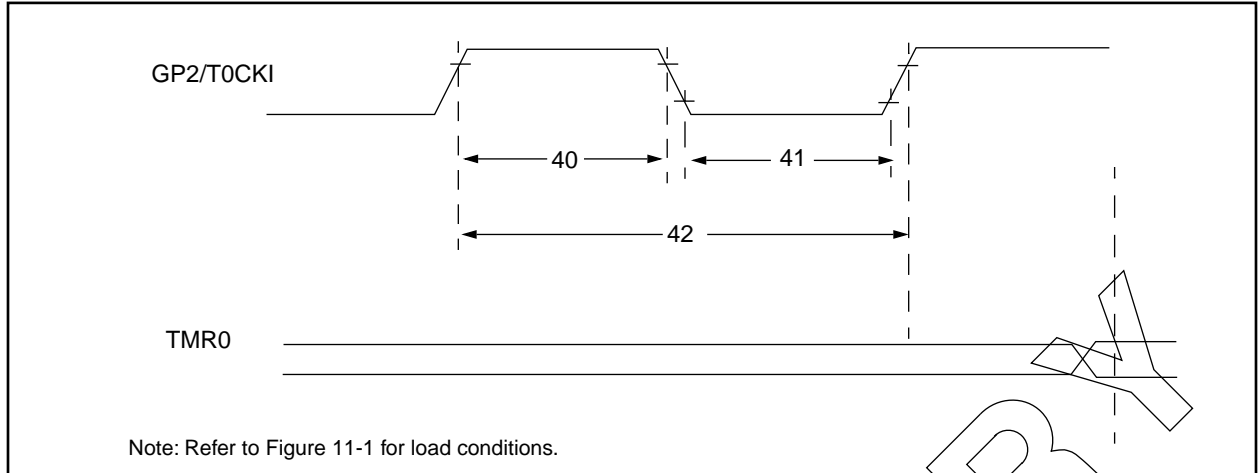
**TABLE 11-4: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER**

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
30	Tmcl	MCLR Pulse Width (low)	2	—	—	μs	VDD = 5V, -40°C to +125°C
31*	Twdt	Watchdog Timer Time-out Period (No Prescaler)	7	18	33	ms	VDD = 5V, -40°C to +125°C
32	Tost	Oscillation Start-up Timer Period	—	1024TOSC	—	—	TOSC = OSC1 period
33*	Tpwrt	Power up Timer Period	28	72	132	ms	VDD = 5V, -40°C to +125°C
34	Tioz	I/O Hi-impedance from MCLR Low or Watchdog Timer Reset	—	—	2.1	μs	

\* These parameters are characterized but not tested.

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**FIGURE 11-5: TIMER0 CLOCK TIMINGS**



**TABLE 11-5: TIMER0 CLOCK REQUIREMENTS**

Param No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
40	Tt0H	T0CKI High Pulse Width	No Prescaler	$0.5TCY + 20^*$	—	—	ns
			With Prescaler	$10^*$	—	—	ns
41	Tt0L	T0CKI Low Pulse Width	No Prescaler	$0.5TCY + 20^*$	—	—	ns
			With Prescaler	$10^*$	—	—	ns
42	Tt0P	T0CKI Period	Greater of: $20\mu s$ or $TCY + 40^*$ N		—	—	ns N = prescale value (1, 2, 4, ..., 256)
48	Tcke2tmr1	Delay from external clock edge to timer increment	$2Tosc$	—	$7Tosc$	—	

\* These parameters are characterized but not tested.

† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

# PIC12C67X

**TABLE 11-6: GPIO PULL-UP RESISTOR RANGES**

VDD (Volts)	Temperature (°C)	Min	Typ	Max	Units
GP0/GP1					
2.5	-40	50K	62K	84K	kΩ
	25	63K	73K	84K	kΩ
	85	63K	80K	84K	kΩ
	125	63K	81K	84K	kΩ
5.5	-40	19K	22K	27K	kΩ
	25	25K	27K	31K	kΩ
	85	28K	33K	40K	kΩ
	125	31K	36K	43K	kΩ
GP3					
2.5	-40	490K	710K	965K	kΩ
	25	610K	890K	1.2M	kΩ
	85	769K	1.1M	1.5M	kΩ
	125	810K	1.2M	1.6M	kΩ
5.5	-40	310K	410K	510K	kΩ
	25	360K	470K	580K	kΩ
	85	430K	550K	670K	kΩ
	125	465K	585K	715K	kΩ

\* These parameters are characterized but not tested.

PRELIMINARY

**TABLE 11-7: A/D CONVERTER CHARACTERISTICS:**  
**PIC12C671-04 (COMMERCIAL, INDUSTRIAL, EXTENDED<sup>(3)</sup>)**  
**PIC12C671-10 (COMMERCIAL, INDUSTRIAL, EXTENDED<sup>(3)</sup>)**  
**PIC12C672-04 (COMMERCIAL, INDUSTRIAL, EXTENDED<sup>(3)</sup>)**  
**PIC12C672-10 (COMMERCIAL, INDUSTRIAL, EXTENDED<sup>(3)</sup>)**

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
	NR	Resolution	—	—	8-bits	—	$V_{REF} = V_{DD} = 5.12V, V_{SS} \leq AIN \leq V_{REF}$
	NINT	Integral error	—	—	less than $\pm 1$ LSb	—	$V_{REF} = V_{DD} = 5.12V, V_{SS} \leq AIN \leq V_{REF}$
	NDIF	Differential error	—	—	less than $\pm 1$ LSb	—	$V_{REF} = V_{DD} = 5.12V, V_{SS} \leq AIN \leq V_{REF}$
	NFS	Full scale error	—	—	less than $\pm 1$ LSb	—	$V_{REF} = V_{DD} = 5.12V, V_{SS} \leq AIN \leq V_{REF}$
	NOFF	Offset error	—	—	less than $\pm 1$ LSb	—	$V_{REF} = V_{DD} = 5.12V, V_{SS} \leq AIN \leq V_{REF}$
	—	Monotonicity	—	guaranteed	—	—	$V_{SS} \leq AIN \leq V_{REF}$
	VREF	Reference voltage	3.0V	—	$V_{DD} + 0.3$	—	
	VAIN	Analog input voltage	$V_{SS} - 0.3$	—	$V_{REF} + 0.3$	—	
	ZAIN	Recommended impedance of analog voltage source	—	—	10.0	k $\Omega$	
	IAD	A/D conversion current (VDD)	—	180	—	$\mu A$	Average current consumption when A/D is on. (Note 1)
	IREF	VREF input current (Note 2)	—	—	1 10	mA $\mu A$	During sampling All other times

\* These parameters are characterized but not tested.

† 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: When A/D is off, it will not consume any current other than minor leakage current. The power-down current spec includes any such leakage from the A/D module.

2: VREF current is from GP1 pin or VDD pin, whichever is selected as reference input.

3: **Extended operating range is Advance Information for this device.**

# PIC12C67X

**TABLE 11-8: A/D CONVERTER CHARACTERISTICS:**  
**PIC12LC671-04 (COMMERCIAL, INDUSTRIAL, EXTENDED<sup>(4)</sup>)**  
**PIC12LC672-04 (COMMERCIAL, INDUSTRIAL, EXTENDED<sup>(4)</sup>)**

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
	NR	Resolution	—	—	8-bits	—	VREF = VDD = 3.0V (Note 1)
	NINT	Integral error	—	—	less than ±1 LSb	—	VREF = VDD = 3.0V (Note 1)
	NDIF	Differential error	—	—	less than ±1 LSb	—	VREF = VDD = 3.0V (Note 1)
	NFS	Full scale error	—	—	less than ±1 LSb	—	VREF = VDD = 3.0V (Note 1)
	NOFF	Offset error	—	—	less than ±1 LSb	—	VREF = VDD = 3.0V (Note 1)
	—	Monotonicity	—	guaranteed	—	—	VSS ≤ AIN ≤ VREF
	VREF	Reference voltage	3.0V	—	VDD + 0.3	V	
	VAIN	Analog input voltage	VSS - 0.3	—	VREF + 0.3	V	
	ZAIN	Recommended impedance of analog voltage source	—	—	10.0	kΩ	
	IAD	A/D conversion current (VDD)	—	90	—	μA	Average current consumption when A/D is on. (Note 2)
	IREF	VREF input current (Note 3)	—	—	1 10	mA μA	During sampling All other times

\* These parameters are characterized but not tested.

† 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: These specifications apply if VREF = 3.0V and if VDD ≥ 3.0V. VIN must be between VSS and VREF

2: When A/D is off, it will not consume any current other than minor leakage current. The power-down current spec includes any such leakage from the A/D module.

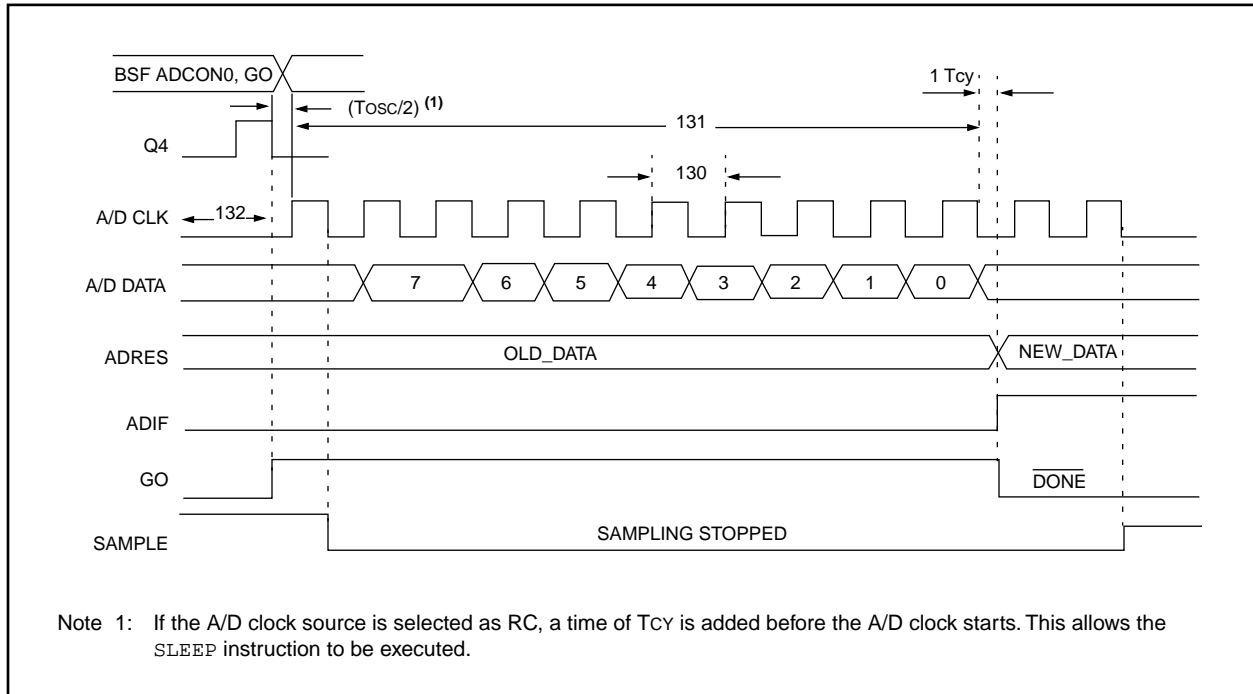
3: VREF current is from GP1 pin or VDD pin, whichever is selected as reference input.

4: **Extended operating range is Advance Information for this device.**

PRELIMINARY



**FIGURE 11-6: A/D CONVERSION TIMING**



**TABLE 11-9: A/D CONVERSION REQUIREMENTS**

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
130	TAD	A/D clock period	1.6 2.0		— —	$\mu s$ $\mu s$	$V_{REF} \geq 3.0V$ $V_{REF}$ full range
130	TAD	A/D Internal RC Oscillator source	3.0 2.0	6.0 4.0	9.0 6.0	$\mu s$ $\mu s$	ADCS1:ADCS0 = 11 (RC oscillator source) PIC12LC67X, $V_{DD} = 3.0V$ PIC12C67X
131	TCNV	Conversion time (not including S/H time). Note 1	—	9.5TAD	—	—	
132	TACQ	Acquisition time	Note 2	20	—	$\mu s$	

\* These parameters are characterized but not tested.

† 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: ADRES register may be read on the following  $T_{CY}$  cycle.

# PIC12C67X

---

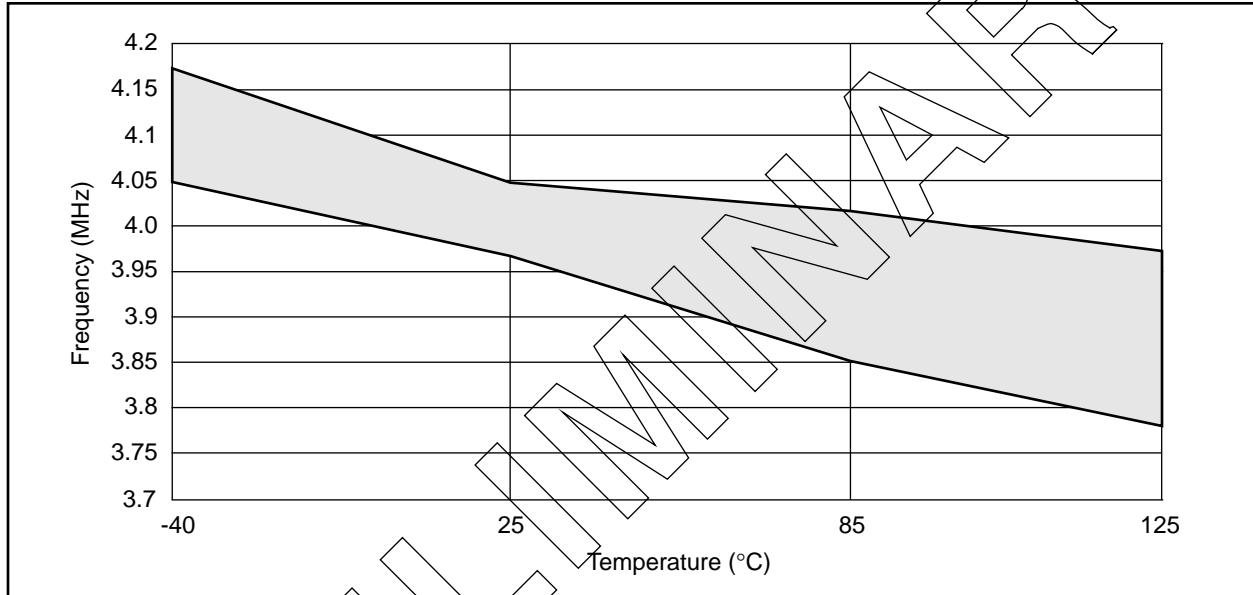
NOTES:

## 12.0 DC AND AC CHARACTERISTICS - PIC12C67X

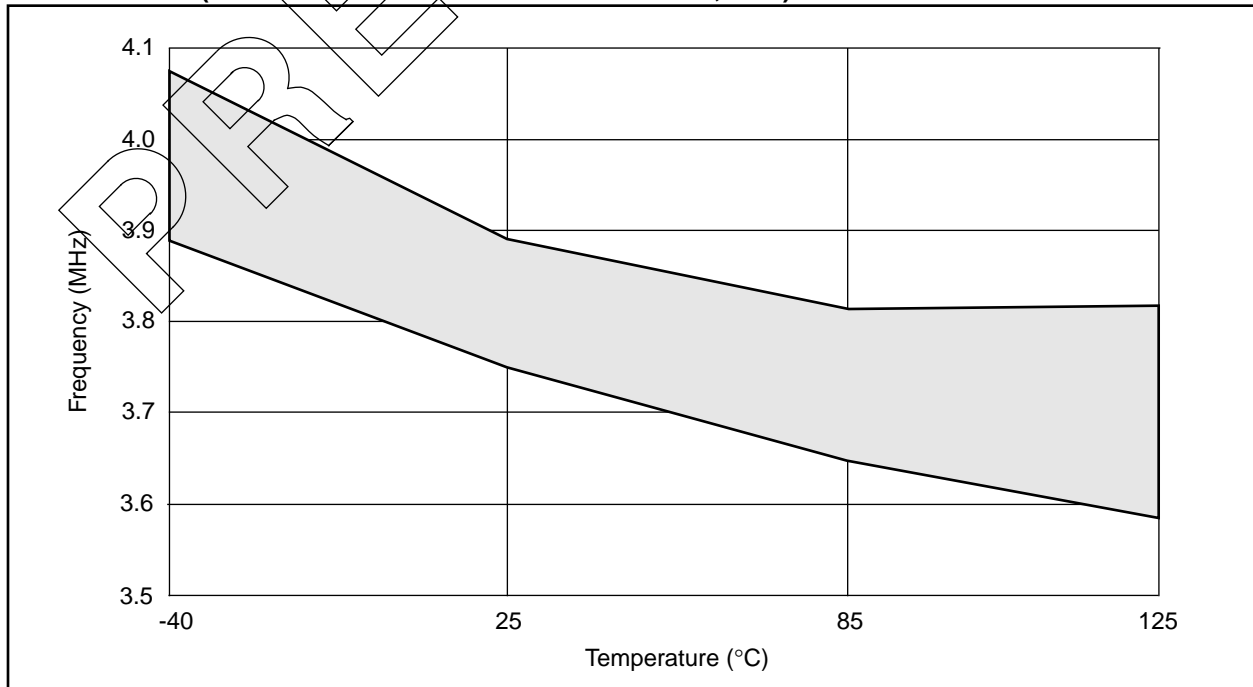
The graphs and tables provided in this section are for design guidance and are not tested or guaranteed. In some graphs or tables the data presented are outside specified operating range (e.g., outside specified VDD range). This is for information only and devices will operate properly only within the 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 + 3 $\sigma$ ) and (mean - 3 $\sigma$ ) respectively, where  $\sigma$  is standard deviation.

**FIGURE 12-1: CALIBRATED INTERNAL RC FREQUENCY RANGE VS. TEMPERATURE (VDD = 5.0V)  
(INTERNAL RC IS CALIBRATED TO 25°C, 5.0V)**



**FIGURE 12-2: CALIBRATED INTERNAL RC FREQUENCY RANGE VS. TEMPERATURE (VDD = 3.0V)  
(INTERNAL RC IS CALIBRATED TO 25°C, 5.0V)**



# PIC12C67X

FIGURE 12-3: INTERNAL RC FREQUENCY VS. CALIBRATION VALUE ( $V_{DD} = 5.5V$ )

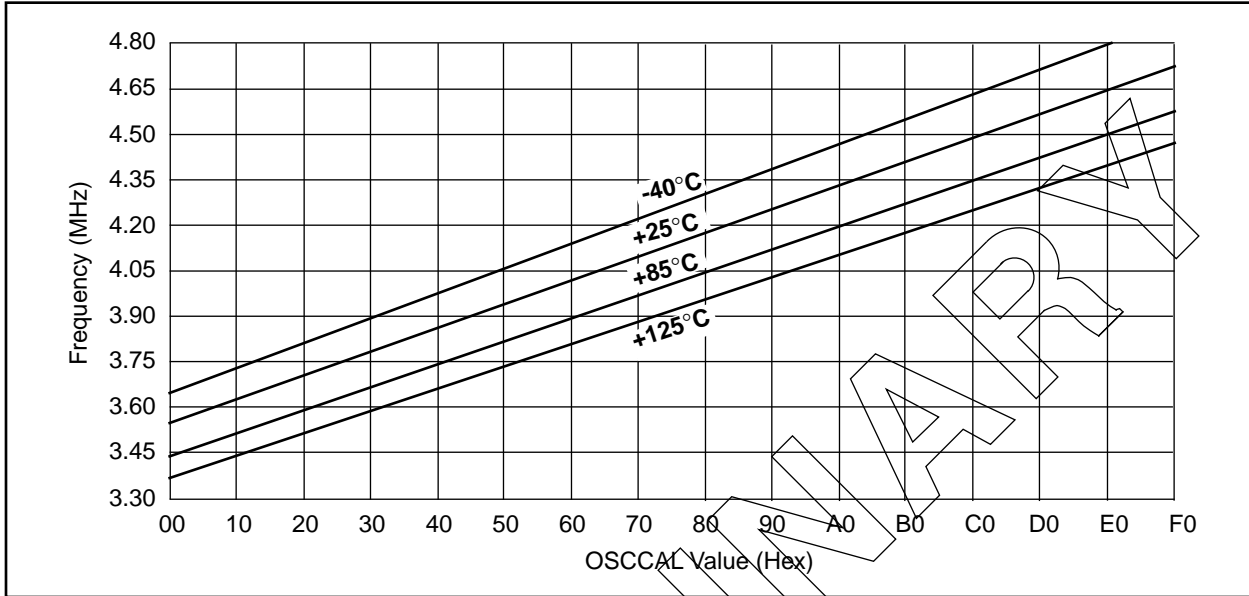
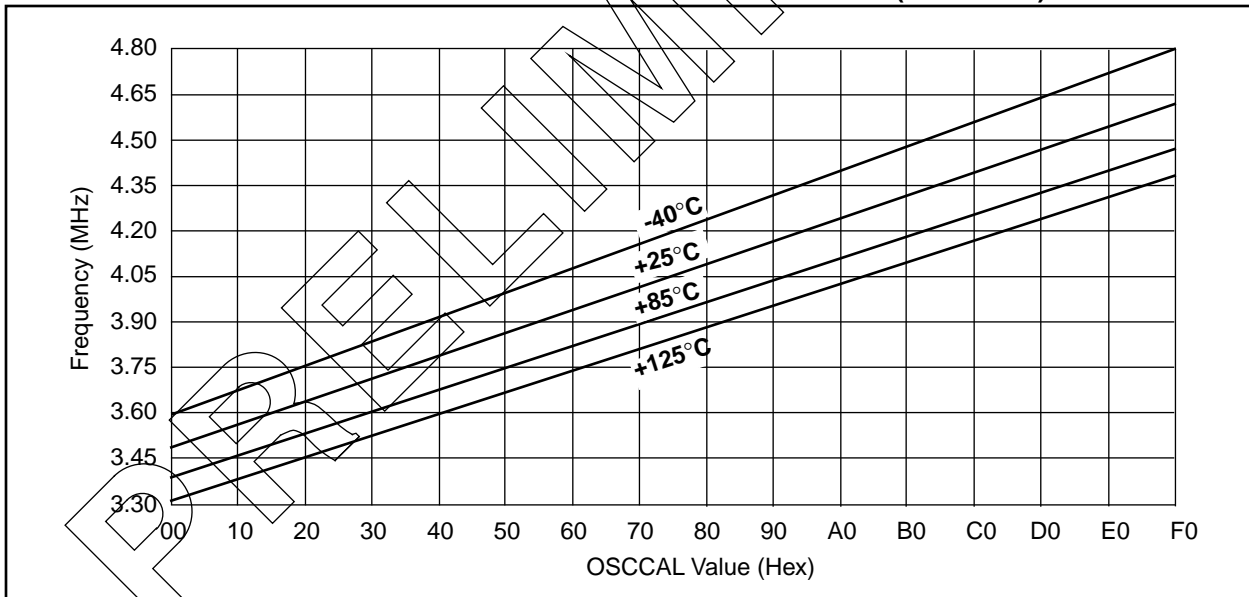


FIGURE 12-4: INTERNAL RC FREQUENCY VS. CALIBRATION VALUE ( $V_{DD} = 3.5V$ )



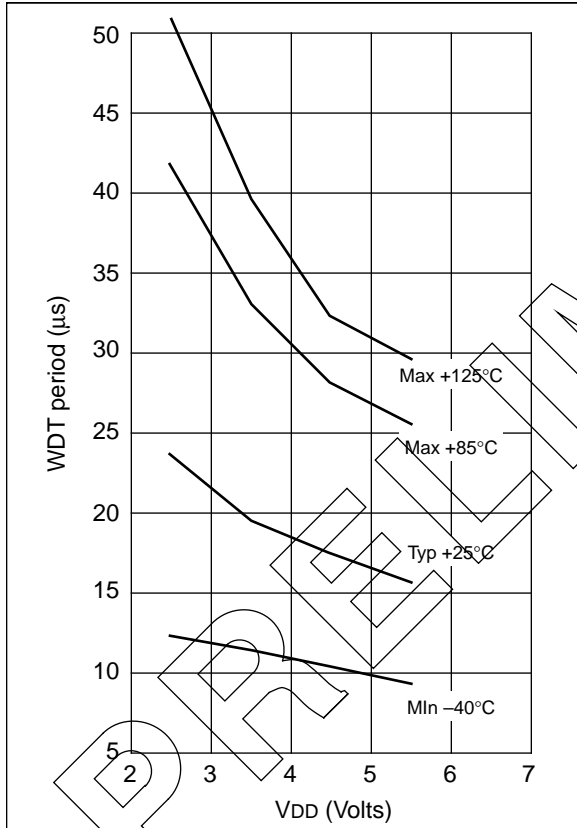
# PIC12C67X

**TABLE 12-1: DYNAMIC I<sub>DD</sub> (TYPICAL) - WDT ENABLED, 25°C**

Oscillator	Frequency	V <sub>DD</sub> = 2.5V	V <sub>DD</sub> = 5.5V
External RC	4 MHz	250 μA*	620 μA*
Internal RC	4 MHz	420 μA	1.1 mA
XT	4 MHz	251 μA	775 μA
LP	32 KHz	7 μA	37 μA

\*Does not include current through external R&C.

**FIGURE 12-5: WDT TIMER TIME-OUT PERIOD vs. V<sub>DD</sub>**



# PIC12C67X

FIGURE 12-6:  $I_{OH}$  vs.  $V_{OH}$ ,  $V_{DD} = 2.5\text{ V}$

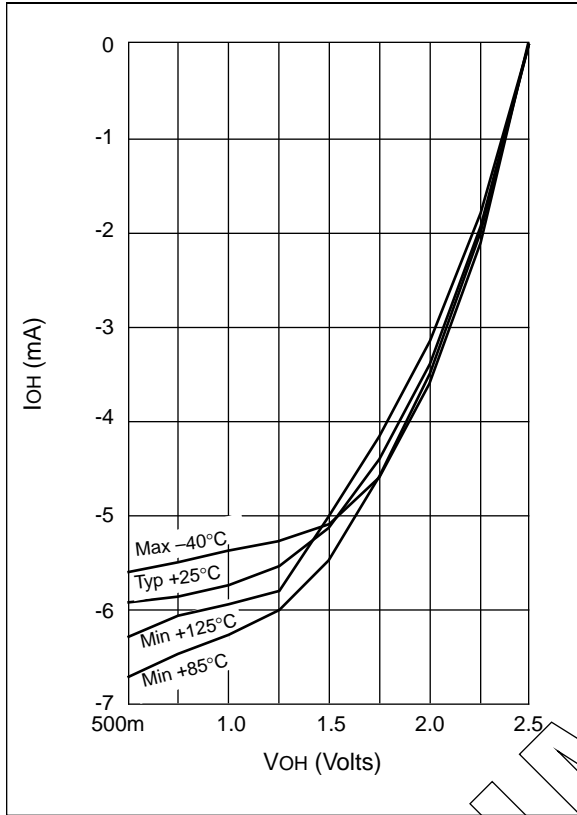


FIGURE 12-8:  $I_{OL}$  vs.  $V_{OL}$ ,  $V_{DD} = 2.5\text{ V}$

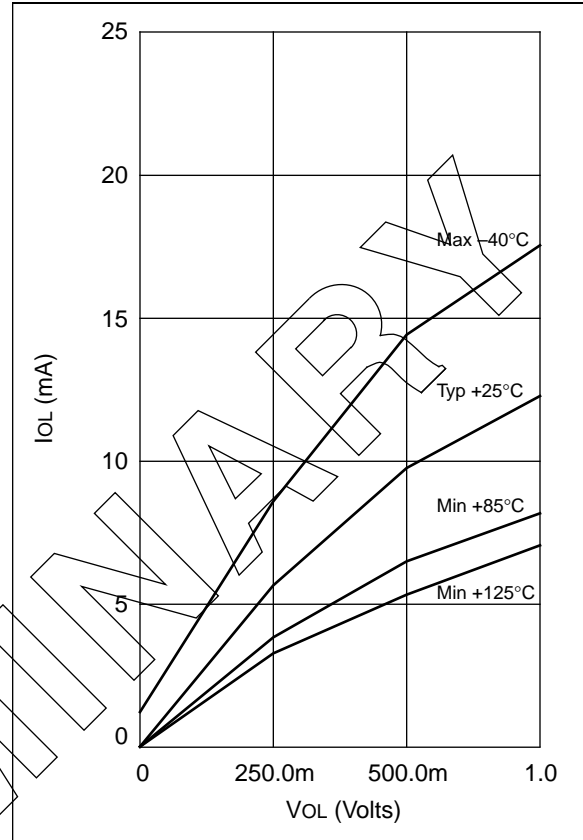


FIGURE 12-7:  $I_{OH}$  vs.  $V_{OH}$ ,  $V_{DD} = 5.5\text{ V}$

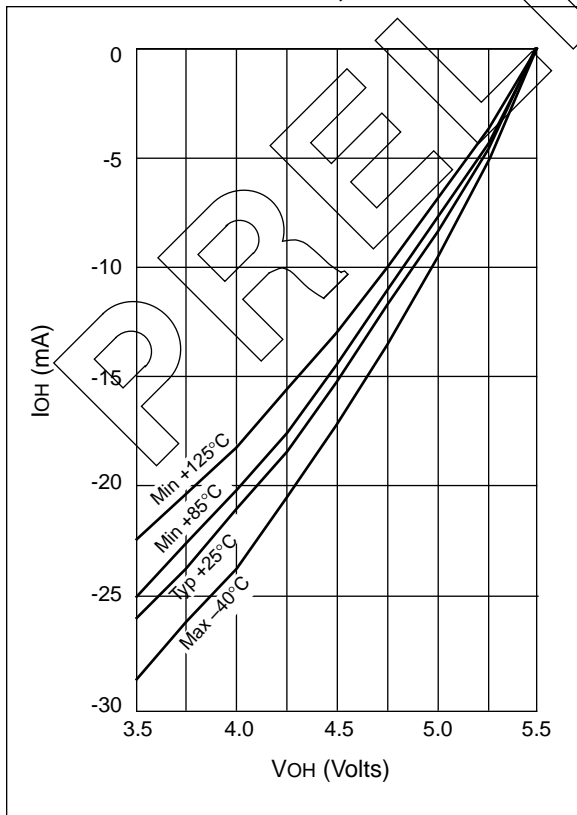
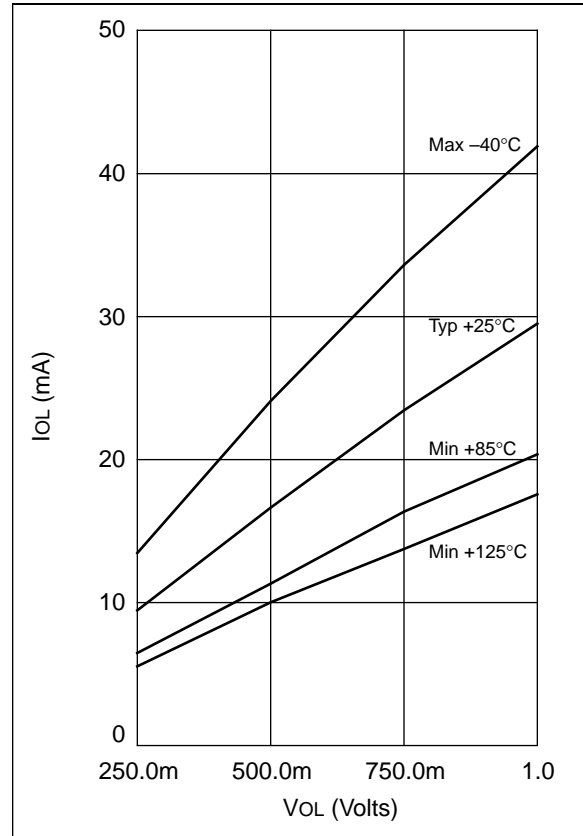


FIGURE 12-9:  $I_{OL}$  vs.  $V_{OL}$ ,  $V_{DD} = 5.5\text{ V}$



## 13.0 PACKAGING INFORMATION

### 13.1 Package Marking Information

#### 8-Lead PDIP (300 mil)



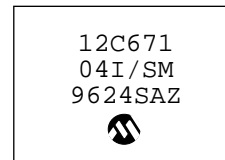
#### Example



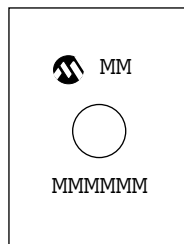
#### 8-Lead SOIC (208 mil)



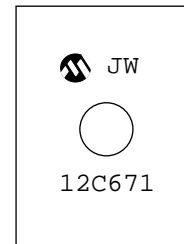
#### Example



#### 8-Lead Windowed Ceramic Side Brazed (300 mil)



#### Example



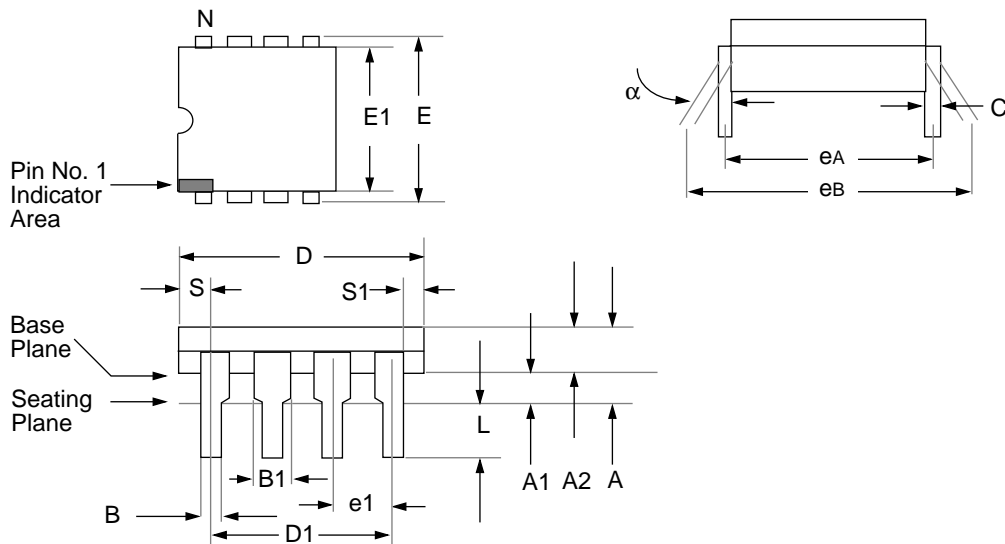
<b>Legend:</b> MM...M	Microchip part number information
XX...X	Customer specific information*
AA	Year code (last 2 digits of calendar year)
BB	Week code (week of January 1 is week '01')
C	Facility code of the plant at which wafer is manufactured
	C = Chandler, Arizona, U.S.A.,
	S = Tempe, Arizona, U.S.A.
	T = Tempe, Arizona, U.S.A.
D	Mask revision number
E	Assembly code of the plant or country of origin in which part was assembled

**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.

# PIC12C67X

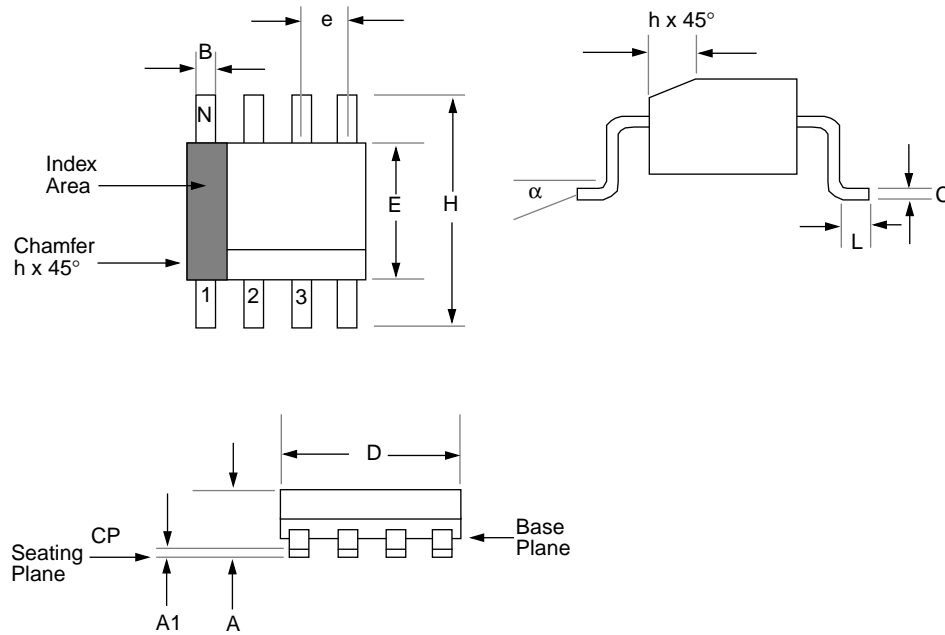
## 13.2 8-Lead Plastic Dual In-line (300 mil)



Package Group: Plastic Dual In-Line (PLA)						
Symbol	Millimeters			Inches		
	Min	Max	Notes	Min	Max	Notes
$\alpha$	0°	10°		0°	10°	
A	–	4.064		–	0.160	
A1	0.381	–		0.015	–	
A2	3.048	3.810		0.120	0.150	
B	0.355	0.559		0.014	0.022	
B1	1.397	1.651		0.055	0.065	
C	0.203	0.381	Typical	0.008	0.015	Typical
D	9.017	10.922		0.355	0.430	
D1	7.620	7.620	Reference	0.300	0.300	Reference
E	7.620	8.255		0.300	0.325	
E1	6.096	7.112		0.240	0.280	
e1	2.489	2.591	Typical	0.098	0.102	Typical
eA	7.620	7.620	Reference	0.300	0.300	Reference
eB	7.874	9.906		0.310	0.390	
L	3.048	3.556		0.120	0.140	
N	8	8		8	8	
S	0.889	–		0.035	–	
S1	0.254	–		0.010	–	



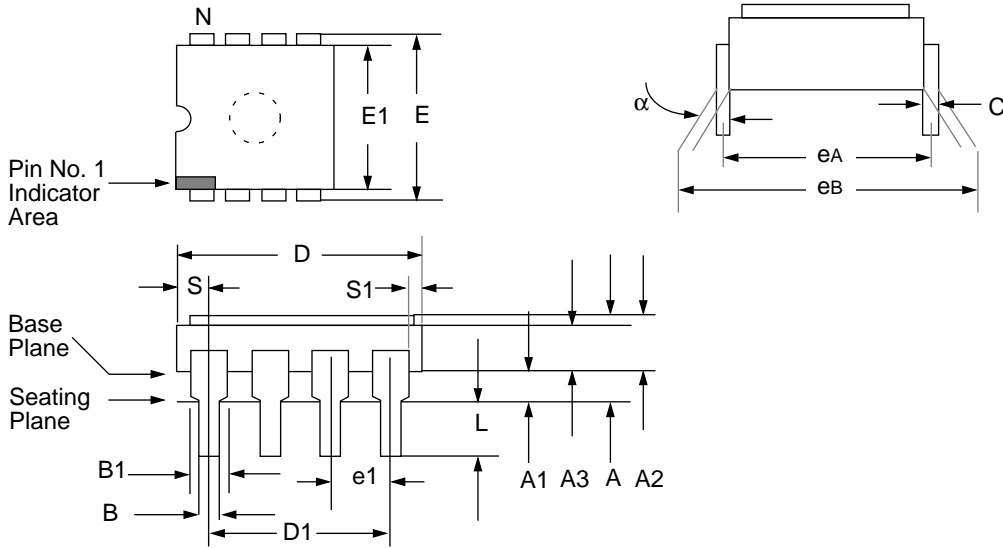
## 13.3 8-Lead Plastic Surface Mount (SOIC - Medium, 208 mil Body)



Package Group: Plastic SOIC (SM)						
Symbol	Millimeters			Inches		
	Min	Max	Notes	Min	Max	Notes
α	0°	8°		0°	8°	
A	1.778	2.00		0.070	0.079	
A1	0.101	0.249		0.004	0.010	
B	0.355	0.483		0.014	0.019	
C	0.190	0.249		0.007	0.010	
D	5.080	5.334		0.200	0.210	
E	5.156	5.411		0.203	0.213	
e	1.270	1.270	Reference	0.050	0.050	Reference
H*	7.670	8.103		0.302	0.319	
h	0.381	0.762		0.015	0.030	
L	0.508	1.016		0.020	0.040	
N	14	14		14	14	
CP	—	0.102		—	0.004	

# PIC12C67X

## 13.4 8-Lead Ceramic Side Brazed Dual In-Line with Window (JW) (300 mil)



Package Group: Ceramic Side Brazed Dual In-Line (CER)

Symbol	Millimeters			Inches		
	Min	Max	Notes	Min	Max	Notes
$\alpha$	0°	10°		0°	10°	
A	3.937	5.030		0.155	0.198	
A1	0.635	1.143		0.025	0.045	
A2	2.921	3.429		0.115	0.135	
A3	1.778	2.413		0.070	0.095	
B	0.406	0.508		0.016	0.020	
B1	1.371	1.371	Typical	0.054	0.054	Typical
C	0.228	0.305	Typical	0.009	0.012	Typical
D	13.004	13.412		0.512	0.528	
D1	7.416	7.824	BSC	0.292	0.308	BSC
E	7.569	8.230		0.298	0.324	
E1	7.112	7.620		0.280	0.300	
e1	2.540	2.540	Typical	0.100	0.100	Typical
eA	7.620	7.620	BSC	0.300	0.300	BSC
eB	7.620	9.652		0.300	0.380	
L	3.302	4.064		0.130	0.160	
S	2.540	3.048		0.100	0.120	
S1	0.127	—		0.005	—	

## APPENDIX A: COMPATIBILITY

To convert code written for PIC16C5X to PIC12C67X, the user should take the following steps:

1. Remove any program memory page select operations (PA2, PA1, PA0 bits) for `CALL`, `GOTO`.
2. Revisit any computed jump operations (write to PC or add to PC, etc.) to make sure page bits are set properly under the new scheme.
3. Eliminate any data memory page switching. Redefine data variables to reallocate them.
4. Verify all writes to `STATUS`, `OPTION`, and `FSR` registers since these have changed.
5. Change reset vector to 0000h.

# PIC12C67X

---

NOTES:

## INDEX

### A

#### A/D

Accuracy/Error .....	39
ADCON0 Register .....	33
ADIF bit .....	35
Analog Input Model Block Diagram .....	36
Analog-to-Digital Converter .....	33
Configuring Analog Port Pins .....	37
Configuring the Interrupt .....	35
Configuring the Module .....	35
Connection Considerations .....	39
Conversion Clock .....	37
Conversions .....	38
Converter Characteristics .....	87
Delays .....	36
Effects of a Reset .....	39
Equations .....	36
Flowchart of A/D Operation .....	40
GO/DONE bit .....	35
Internal Sampling Switch (R <sub>ss</sub> ) Impedence .....	36
Operation During Sleep .....	39
Sampling Requirements .....	36
Sampling Time .....	36
Source Impedence .....	36
Time Delays .....	36
Transfer Function .....	39
Absolute Maximum Ratings .....	75
ADDLW Instruction .....	60
ADDWF Instruction .....	60
ADIE bit .....	18
ADIF bit .....	19
ADRES Register .....	13, 33, 35
ALU .....	7
ANDLW Instruction .....	60
ANDWF Instruction .....	60
Application Notes	
AN546 .....	33
AN556 .....	22
Architecture	
Harvard .....	7
Overview .....	7
von Neumann .....	7
Assembler	
MPASM Assembler .....	72
<b>B</b>	
BCF Instruction .....	61
Bit Manipulation .....	58
Block Diagrams	
Analog Input Model .....	36
On-Chip Reset Circuit .....	45
Timer0 .....	27
Timer0/WDT Prescaler .....	30
Watchdog Timer .....	53
BSF Instruction .....	61
BTFSC Instruction .....	61
BTFSS Instruction .....	62
<b>C</b>	
C bit .....	15
CAL0 bit .....	21
CAL1 bit .....	21
CAL2 bit .....	21
CAL3 bit .....	21

CALFST bit .....	21
CALL Instruction .....	62
CALSLW bit .....	21
Carry bit .....	7
Clocking Scheme .....	10
CLRF Instruction .....	62
CLRW Instruction .....	62
CLRWDI Instruction .....	63
Code Examples	
Changing Prescaler (Timer0 to WDT) .....	31
Changing Prescaler (WDT to Timer0) .....	31
Indirect Addressing .....	23
Code Protection .....	41, 55
COMF Instruction .....	63
Computed GOTO .....	22
Configuration Bits .....	41
<b>D</b>	
DC bit .....	15
DC Characteristics	
PIC12C671 .....	77
PIC12C672 .....	77
DECF Instruction .....	63
DECFSZ Instruction .....	63
Development Support .....	3, 71
Development Tools .....	71
Diagrams - See Block Diagrams	
Digit Carry bit .....	7
Direct Addressing .....	23
<b>E</b>	
Electrical Characteristics	
PIC12C67X .....	75
External Brown-out Protection Circuit .....	49
External Power-on Reset Circuit .....	49
<b>F</b>	
Family of Devices	
PIC12CXXX .....	4
Features .....	1
FSR Register .....	13, 14, 23
Fuzzy Logic Dev. System ( <i>fuzzyTECH</i> <sup>®</sup> -MP) .....	73
<b>G</b>	
General Description .....	3
GIE bit .....	50
GOTO Instruction .....	64
GPIF bit .....	52
GPIO .....	25, 47
GPIO Register .....	13
GPPU bit .....	16
<b>I</b>	
I/O Interfacing .....	25
I/O Ports .....	25
I/O Programming Considerations .....	26
ICEPIC Low-Cost PIC16CXXX In-Circuit Emulator .....	71
ID Locations .....	41
INCF Instruction .....	64
INCFSZ Instruction .....	64
In-Circuit Serial Programming .....	41, 55
INDF Register .....	14, 23
Indirect Addressing .....	23
Initialization Conditions for All Registers .....	47
Instruction Cycle .....	10
Instruction Flow/Pipelining .....	10
Instruction Format .....	57

# PIC12C67X

Instruction Set		PIC12C67X	12
ADDLW	60	MOV Instruction	65
ADDWF	60	MOVLW Instruction	65
ANDLW	60	MOVWF Instruction	65
ANDWF	60	MP-DriveWay™ - Application Code Generator	73
BCF	61	MPLAB™ C	73
BSF	61	MPLAB™ Integrated Development Environment Software	72
BTFSC	61		
BTFSS	62	<b>N</b>	
CALL	62	NOP Instruction	66
CLRF	62		
CLRW	62	<b>O</b>	
CLRWDT	63	Opcode	57
COMF	63	OPTION Instruction	66
DECF	63	OPTION Register	16
DECFSZ	63	Orthogonal	7
GOTO	64	OSC selection	41
INCF	64	OSCCAL Register	21
INCFSZ	64	Oscillator	
IORLW	64	EXTRC	46
IORWF	65	HS	46
MOVF	65	INTRC	46
MOVLW	65	LP	46
MOVWF	65	XT	46
NOP	66	Oscillator Configurations	42
OPTION	66	Oscillator Types	
RETFIE	66	EXTRC	42
RETLW	66	HS	42
RETURN	67	INTRC	42
RLF	67	LP	42
RRF	67	XT	42
SLEEP	67		
SUBLW	68	<b>P</b>	
SUBWF	68	Package Marking Information	95
SWAPF	69	Packaging Information	95
TRIS	69	Paging, Program Memory	22
XORLW	69	PCL	58
XORWF	69	PCL Register	13, 14, 22
Section	57	PCLATH	47
INTCON Register	17	PCLATH Register	13, 14, 22
INTEDG bit	16	PCON Register	20, 46
Internal Sampling Switch (Rss) Impedence	36	$\overline{PD}$ bit	15, 44
Interrupts	41	PIC12C67X DC and AC Characteristics	91
A/D	50	PICDEM-1 Low-Cost PIC16/17 Demo Board	72
GP2/INT	50	PICDEM-2 Low-Cost PIC16CXX Demo Board	72
GPIO Port	50	PICDEM-3 Low-Cost PIC16CXXX Demo Board	72
Section	50	PICMASTER® In-Circuit Emulator	71
TMR0	52	PICSTART® Plus Entry Level Development System	71
TMR0 Overflow	50	PIE1 Register	18
IORLW Instruction	64	Pinout Description	
IORWF Instruction	65	PIC12C67X	9
IRP bit	15	PIR1 Register	19
<b>K</b>		POP	22
KeelLoq® Evaluation and Programming Tools	73	POR	46
<b>L</b>		Oscillator Start-up Timer (OST)	41, 46
Loading of PC	22	Power Control Register (PCON)	46
<b>M</b>		Power-on Reset (POR)	41, 46, 47
$\overline{MCLR}$	44, 47	Power-up Timer (PWRT)	41, 46
Memory		Power-Up-Timer (PWRT)	46
Data Memory	11	Time-out Sequence	46
Program Memory	11	Time-out Sequence on Power-up	48
Program Memory Map		TO	44
PIC12C67X	11	$\overline{POR}$ bit	20
Register File Map		Power	44
		Power-down Mode (SLEEP)	54
		Prescaler, Switching Between Timer0 and WDT	31
		PRO MATE® II Universal Programmer	71

Product Identification System - PIC12C67X .....	115	Interrupt .....	27
Program Branches .....	7	Interrupt Timing .....	28
Program Memory		Prescaler .....	30
Paging .....	22	Prescaler Block Diagram .....	30
Program Memory Map		Section .....	27
PIC12C67X .....	11	Switching Prescaler Assignment .....	31
Program Verification .....	55	Synchronization .....	29
PS0 bit .....	16	T0CKI .....	29
PS1 bit .....	16	T0IF .....	52
PS2 bit .....	16	Timing .....	27
PSA bit .....	16	TMR0 Interrupt .....	52
PUSH .....	22	Timing Diagrams	
<b>R</b>		A/D Conversion .....	89
RC Oscillator .....	43	CLKOUT and I/O .....	83
Read Modify Write .....	26	External Clock Timing .....	82
Read-Modify-Write .....	26	Time-out Sequence .....	48
Register File .....	11	Timer0 .....	27, 85
Registers		Timer0 Interrupt Timing .....	28
Map		Timer0 with External Clock .....	29
PIC12C67X .....	12	Wake-up from Sleep via Interrupt .....	55
Reset Conditions .....	47	$\overline{TO}$ bit .....	15
Reset .....	41, 44	TOSE bit .....	16
Reset Conditions for Special Registers .....	47	TRIS Instruction .....	69
RETFIE Instruction .....	66	TRIS Register .....	14, 25
RETLW Instruction .....	66	Two's Complement .....	
RETURN Instruction .....	67		
RLF Instruction .....	67		
RP0 bit .....	11, 15		
RP1 bit .....	15		
RRF Instruction .....	67		
<b>S</b>			
SEEVAL <sup>®</sup> Evaluation and Programming System .....	73		
Services			
One-Time-Programmable (OTP) .....	5		
Quick-Turnaround-Production (QTP) .....	5		
Serialized Quick-Turnaround Production (SQTP) .....	5		
SFR .....	58		
SFR As Source/Destination .....	58		
SLEEP .....	41, 44		
SLEEP Instruction .....	67		
Software Simulator (MPLAB <sup>™</sup> SIM) .....	73		
Special Features of the CPU .....	41		
Special Function Register			
PIC12C67X .....	13		
Special Function Registers .....	58		
Special Function Registers, Section .....	12		
Stack .....	22		
Overflows .....	22		
Underflow .....	22		
STATUS Register .....	15		
SUBLW Instruction .....	68		
SUBWF Instruction .....	68		
SWAPF Instruction .....	69		
<b>T</b>			
T0CS bit .....	16		
TAD .....	37		
Timer0			
RTCC .....	47		
Timers			
Timer0			
Block Diagram .....	27		
External Clock .....	29		
External Clock Timing .....	29		
Increment Delay .....	29		

# PIC12C67X

## LIST OF EXAMPLES

Example 3-1: Instruction Pipeline Flow .....	10
Example 4-1: Indirect Addressing .....	23
Example 5-1: Read-Modify-Write Instructions on an I/O Port .....	26
Example 6-1: Changing Prescaler (Timer0→WDT) .....	31
Example 6-2: Changing Prescaler (WDT→Timer0) .....	31
Example 7-1: Calculating the Minimum Required Sample Time.....	36
Example 7-2: Doing an A/D Conversion .....	38
Example 8-1: Saving STATUS and W Registers in RAM.....	52

## LIST OF FIGURES

Figure 3-1: PIC12C67X Block Diagram .....	8
Figure 3-2: Clock/Instruction Cycle .....	10
Figure 4-1: PIC12C67X Program Memory Map and Stack.....	11
Figure 4-2: PIC12C67X Register File Map.....	12
Figure 4-3: Status Register (Address 03h, 83h).....	15
Figure 4-4: OPTION Register (Address 81h) .....	16
Figure 4-5: INTCON Register (Address 0Bh, 8Bh) .....	17
Figure 4-6: PIE1 Register (Address 8Ch) .....	18
Figure 4-7: PIR1 Register (Address 0Ch) .....	19
Figure 4-8: PCON Register (Address 8Eh) .....	20
Figure 4-9: OSCCAL Register (Address 8Fh).....	21
Figure 4-10: Loading of PC In Different Situations.....	22
Figure 4-11: Direct/Indirect Addressing.....	23
Figure 5-1: Equivalent Circuit for a Single I/O Pin.....	25
Figure 5-2: Successive I/O Operation .....	26
Figure 6-1: Timer0 Block Diagram .....	27
Figure 6-2: Timer0 Timing: Internal Clock/ No Prescale .....	27
Figure 6-3: Timer0 Timing: Internal Clock/ Prescale 1:2.....	28
Figure 6-4: Timer0 Interrupt Timing .....	28
Figure 6-5: Timer0 Timing with External Clock .....	29
Figure 6-6: Block Diagram of the Timer0/ WDT Prescaler .....	30
Figure 7-1: ADCON0 Register (Address 1Fh).....	33
Figure 7-2: ADCON1 Register (Address 9Fh).....	34
Figure 7-3: A/D Block Diagram .....	35
Figure 7-4: Analog Input Model.....	36
Figure 7-5: A/D Transfer Function.....	39
Figure 7-6: Flowchart of A/D Operation .....	40
Figure 8-1: Configuration Word.....	41
Figure 8-2: Crystal Operation (or Ceramic Resonator) (XT, HS or LP OSC Configuration) .....	42
Figure 8-3: External Clock Input Operation (XT, HS or LP OSC Configuration) .....	42
Figure 8-4: External Parallel Resonant Crystal Oscillator Circuit.....	43
Figure 8-5: External Series Resonant Crystal Oscillator Circuit.....	43
Figure 8-6: External RC Oscillator Mode .....	43
Figure 8-7: Simplified Block Diagram of On-chip Reset Circuit .....	45
Figure 8-8: Time-out Sequence on Power-up (MCLR not Tied to VDD): Case 1 .....	48
Figure 8-9: Time-out Sequence on Power-up (MCLR Not Tied To VDD): Case 2.....	48
Figure 8-10: Time-out Sequence on Power-up (MCLR Tied to VDD).....	48
Figure 8-11: External Power-on Reset Circuit (for Slow VDD Power-up).....	49

Figure 8-12: External Brown-out Protection Circuit 1 .....	49
Figure 8-13: External Brown-out Protection Circuit 2 .....	49
Figure 8-14: Interrupt Logic .....	50
Figure 8-15: INT PIN Interrupt Timing .....	51
Figure 8-16: Watchdog Timer Block Diagram.....	53
Figure 8-17: Summary of Watchdog Timer Registers .....	53
Figure 8-18: Wake-up from Sleep Through Interrupt.....	55
Figure 8-19: Typical In-Circuit Serial Programming Connection .....	55
Figure 9-1: General Format for Instructions.....	57
Figure 11-1: Load Conditions .....	81
Figure 11-2: External Clock Timing .....	82
Figure 11-3: CLKOUT and I/O Timing .....	83
Figure 11-4: Reset, Watchdog Timer, Oscillator Start-Up Timer, and Power-Up Timer Timing.....	84
Figure 11-5: Timer0 Clock Timings.....	85
Figure 11-6: A/D Conversion Timing .....	89
Figure 12-1: Calibrated Internal RC Frequency Range vs. Temperature (VDD = 5.0V) (Internal RC is Calibrated to 25°C, 5.0V) ...	91
Figure 12-2: Calibrated Internal RC Frequency Range vs. Temperature (VDD = 3.0V) (Internal RC is Calibrated to 25°C, 5.0V) ...	91
Figure 12-3: Internal RC Frequency vs. Calibration Value (VDD = 5.5V).....	92
Figure 12-4: Internal RC Frequency vs. Calibration Value (VDD = 3.5V).....	92
Figure 12-5: WDT Timer Time-out Period vs. VDD .....	93
Figure 12-6: IOH vs. VOH, VDD = 2.5 V .....	94
Figure 12-7: IOH vs. VOH, VDD = 5.5 V .....	94
Figure 12-8: IOL vs. VOL, VDD = 2.5 V .....	94
Figure 12-9: IOL vs. VOL, VDD = 5.5 V .....	94

## LIST OF TABLES

Table 1-1: PIC12CXXX Family of Devices .....	4
Table 3-1: PIC12C67X Pinout Description .....	9
Table 4-1: PIC12C67X Special Function Register Summary .....	13
Table 5-1: Summary of Port Registers .....	25
Table 6-1: Registers Associated with Timer0 .....	31
Table 7-1: TAD vs. Device Operating Frequencies .....	37
Table 7-2: Summary of A/D Registers .....	40
Table 8-1: Capacitor Selection For Ceramic Resonators - PIC12C67X .....	42
Table 8-2: Capacitor Selection For Crystal Oscillator - PIC12C67X.....	42
Table 8-3: Time-out in Various Situations .....	46
Table 8-4: Status Bits and Their Significance.....	46
Table 8-5: Reset Condition for Special Registers.....	47
Table 8-6: Initialization Conditions for all registers .....	47
Table 9-1: Opcode Field Descriptions .....	57
Table 9-2: Instruction Set Summary .....	59
Table 10-1: Development Tools From Microchip.....	74
Table 11-1: Cross Reference of Device Specs for Oscillator Configurations and Frequencies of Operation (Commercial Devices) .....	76
Table 11-2: Clock Timing Requirements .....	82
Table 11-3: CLKOUT and I/O Timing Requirements .....	83
Table 11-4: Reset, Watchdog Timer, Oscillator Start-up Timer, Power-up Timer .....	84
Table 11-5: Timer0 Clock Requirements .....	85
Table 11-6: GPIO Pull-up Resistor Ranges.....	86



Table 11-7: A/D Converter Characteristics:  
PIC12C671-04 (Commercial, Industrial,  
Automotive<sup>(3)</sup>)  
PIC12C671-10 (Commercial, Industrial,  
Automotive<sup>(3)</sup>)  
PIC12C672-04 (Commercial, Industrial,  
Automotive<sup>(3)</sup>)  
PIC12C672-10 (Commercial, Industrial,  
Automotive<sup>(3)</sup>) ..... 87

Table 11-8: A/D Converter Characteristics:  
PIC12LC671-04 (Commercial, Industrial,  
Automotive<sup>(4)</sup>)  
PIC12LC672-04 (Commercial, Industrial,  
Automotive<sup>(4)</sup>) ..... 88

Table 11-9: A/D Conversion Requirements ..... 89

Table 12-1: Dynamic IDD (Typical) -  
WDT Enabled, 25°C ..... 93

# PIC12C67X

---

NOTES:

## ON-LINE SUPPORT

Microchip provides two methods of on-line support. These are the Microchip BBS and the Microchip World Wide Web (WWW) site.

Use Microchip's Bulletin Board Service (BBS) to get current information and help about Microchip products. Microchip provides the BBS communication channel for you to use in extending your technical staff with microcontroller and memory experts.

To provide you with the most responsive service possible, the Microchip systems team monitors the BBS, posts the latest component data and software tool updates, provides technical help and embedded systems insights, and discusses how Microchip products provide project solutions.

The web site, like the BBS, is used by Microchip as a means to make files and information easily available to customers. To view the site, the user must have access to the Internet and a web browser, such as Netscape or Microsoft Explorer. Files are also available for FTP download from our FTP site.

### Connecting to the Microchip Internet Web Site

The Microchip web site is available by using your favorite Internet browser to attach to:

**[www.microchip.com](http://www.microchip.com)**

The file transfer site is available by using an FTP service to connect to:

**<ftp://ftp.futureone.com/pub/microchip>**

The web site and file transfer site provide a variety of services. Users may download files for the latest Development Tools, Data Sheets, Application Notes, User's Guides, Articles and Sample Programs. A variety of Microchip specific business information is also available, including listings of Microchip sales offices, distributors and factory representatives. Other data available for consideration is:

- Latest Microchip Press Releases
- Technical Support Section with Frequently Asked Questions
- Design Tips
- Device Errata
- Job Postings
- Microchip Consultant Program Member Listing
- Links to other useful web sites related to Microchip Products

### Connecting to the Microchip BBS

Connect worldwide to the Microchip BBS using either the Internet or the CompuServe® communications network.

#### Internet:

You can telnet or ftp to the Microchip BBS at the address: **[mchippbbs.microchip.com](telnet://mchippbbs.microchip.com)**

#### CompuServe Communications Network:

When using the BBS via the Compuserve Network, in most cases, a local call is your only expense. The Microchip BBS connection does not use CompuServe membership services, therefore you do not need CompuServe membership to join Microchip's BBS. There is no charge for connecting to the Microchip BBS.

The procedure to connect will vary slightly from country to country. Please check with your local CompuServe agent for details if you have a problem. CompuServe service allow multiple users various baud rates depending on the local point of access.

The following connect procedure applies in most locations.

1. Set your modem to 8-bit, No parity, and One stop (8N1). This is not the normal CompuServe setting which is 7E1.
2. Dial your local CompuServe access number.
3. Depress the **<Enter>** key and a garbage string will appear because CompuServe is expecting a 7E1 setting.
4. Type +, depress the **<Enter>** key and "Host Name:" will appear.
5. Type MCHIPBBS, depress the **<Enter>** key and you will be connected to the Microchip BBS.

In the United States, to find the CompuServe phone number closest to you, set your modem to 7E1 and dial (800) 848-4480 for 300-2400 baud or (800) 331-7166 for 9600-14400 baud connection. After the system responds with "Host Name:", type NETWORK, depress the **<Enter>** key and follow CompuServe's directions.

For voice information (or calling from overseas), you may call (614) 723-1550 for your local CompuServe number.

Microchip regularly uses the Microchip BBS to distribute technical information, application notes, source code, errata sheets, bug reports, and interim patches for Microchip systems software products. For each SIG, a moderator monitors, scans, and approves or disapproves files submitted to the SIG. No executable files are accepted from the user community in general to limit the spread of computer viruses.

### Systems Information and Upgrade Hot Line

The Systems Information and Upgrade Line provides system users a listing of the latest versions of all of Microchip's development systems software products. Plus, this line provides information on how customers can receive any currently available upgrade kits. The Hot Line Numbers are:

- 1-800-755-2345 for U.S. and most of Canada, and
- 1-602-786-7302 for the rest of the world.

970301

**Trademarks:** The Microchip name, logo, PIC, PICSTART, PICMASTER and PRO MATE are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries. *FlexROM*, MPLAB and *fuzzyLAB*, are trademarks and SQTP is a service mark of Microchip in the U.S.A.

*fuzzyTECH* is a registered trademark of Inform Software Corporation. IBM, IBM PC-AT are registered trademarks of International Business Machines Corp. Pentium is a trademark of Intel Corporation. Windows is a trademark and MS-DOS, Microsoft Windows are registered trademarks of Microsoft Corporation. CompuServe is a registered trademark of CompuServe Incorporated.

All other trademarks mentioned herein are the property of their respective companies.

# PIC12C67X

---

---

## READER RESPONSE

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

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

To: Technical Publications Manager Total Pages Sent  
RE: Reader Response  
From: Name \_\_\_\_\_  
Company \_\_\_\_\_  
Address \_\_\_\_\_  
City / State / ZIP / Country \_\_\_\_\_  
Telephone: (\_\_\_\_\_) \_\_\_\_\_ - \_\_\_\_\_ FAX: (\_\_\_\_\_) \_\_\_\_\_ - \_\_\_\_\_

Application (optional):

Would you like a reply? \_\_\_Y \_\_\_N

Device: **PIC12C67X** Literature Number: **DS30561A**

Questions:

1. What are the best features of this document?

---

---

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

---

---

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

---

---

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

---

---

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

---

---

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

---

---

7. How would you improve this document?

---

---

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

---

---

## PIC12C67X PRODUCT IDENTIFICATION SYSTEM

PART NO.	-XX	X	/XX	XXX		Examples	
					<b>Pattern:</b>	Special Requirements	a) PIC12C671-04/P Commercial Temp., PDIP Package, 4 MHz, normal VDD limits
					<b>Package:</b>	SM = 208 mil SOIC P = 300 mil PDIP JW = 300 mil Windowed Ceramic Side Brazed	b) PIC12C671-04I/SM Industrial Temp., SOIC package, 4 MHz, normal VDD limits
					<b>Temperature Range:</b>	- = 0°C to +70°C I = -40°C to +85°C E = -40°C to +125°C	
					<b>Frequency Range:</b>	04 = 4 MHz 10 = 10 MHz	c) PIC12C671-04I/P Industrial Temp., PDIP package, 4 MHz, normal VDD limits
					<b>Device</b>	PIC12C671 PIC12C672 PIC12C671T (Tape & reel for SOIC only) PIC12C672T (Tape & reel for SOIC only) PIC12LC671 PIC12LC672 PIC12LC671T (Tape & reel for SOIC only) PIC12LC672T (Tape & reel for SOIC only)	

Please contact your local sales office for exact ordering procedures.

### Sales and Support

Products supported by a preliminary Data SU1 3T

# PIC12C67X

---

NOTES:

NOTES:

---

---

# WORLDWIDE SALES & SERVICE

---

---

## AMERICAS

### Corporate Office

Microchip Technology Inc.  
2355 West Chandler Blvd.  
Chandler, AZ 85224-6199  
Tel: 602-786-7200 Fax: 602-786-7277  
Technical Support: 602 786-7627  
Web: <http://www.microchip.com>

### Atlanta

Microchip Technology Inc.  
500 Sugar Mill Road, Suite 200B  
Atlanta, GA 30350  
Tel: 770-640-0034 Fax: 770-640-0307

### Boston

Microchip Technology Inc.  
5 Mount Royal Avenue  
Marlborough, MA 01752  
Tel: 508-480-9990 Fax: 508-480-8575

### Chicago

Microchip Technology Inc.  
333 Pierce Road, Suite 180  
Itasca, IL 60143  
Tel: 630-285-0071 Fax: 630-285-0075

### Dallas

Microchip Technology Inc.  
14651 Dallas Parkway, Suite 816  
Dallas, TX 75240-8809  
Tel: 972-991-7177 Fax: 972-991-8588

### Dayton

Microchip Technology Inc.  
Two Prestige Place, Suite 150  
Miamisburg, OH 45342  
Tel: 937-291-1654 Fax: 937-291-9175

### Los Angeles

Microchip Technology Inc.  
18201 Von Karman, Suite 1090  
Irvine, CA 92612  
Tel: 714-263-1888 Fax: 714-263-1338

### New York

Microchip Technology Inc.  
150 Motor Parkway, Suite 416  
Hauppauge, NY 11788  
Tel: 516-273-5305 Fax: 516-273-5335

### San Jose

Microchip Technology Inc.  
2107 North First Street, Suite 590  
San Jose, CA 95131  
Tel: 408-436-7950 Fax: 408-436-7955

### Toronto

Microchip Technology Inc.  
5925 Airport Road, Suite 200  
Mississauga, Ontario L4V 1W1, Canada  
Tel: 905-405-6279 Fax: 905-405-6253

## ASIA/PACIFIC

### Hong Kong

Microchip Asia Pacific  
RM 3801B, Tower Two  
Metroplaza  
223 Hing Fong Road  
Kwai Fong, N.T., Hong Kong  
Tel: 852-2-401-1200 Fax: 852-2-401-3431

### India

Microchip Technology India  
No. 6, Legacy, Convent Road  
Bangalore 560 025, India  
Tel: 91-80-299-4036 Fax: 91-80-559-9840

### Korea

Microchip Technology Korea  
168-1, Youngbo Bldg. 3 Floor  
Samsung-Dong, Kangnam-Ku  
Seoul, Korea  
Tel: 82-2-554-7200 Fax: 82-2-558-5934

### Shanghai

Microchip Technology  
RM 406 Shanghai Golden Bridge Bldg.  
2077 Yan'an Road West, Hongjiao District  
Shanghai, PRC 200335  
Tel: 86-21-6275-5700  
Fax: 86 21-6275-5060

### Singapore

Microchip Technology Taiwan  
Singapore Branch  
200 Middle Road  
#10-03 Prime Centre  
Singapore 188980  
Tel: 65-334-8870 Fax: 65-334-8850

### Taiwan, R.O.C

Microchip Technology Taiwan  
10F-1C 207  
Tung Hua North Road  
Taipei, Taiwan, ROC  
Tel: 886 2-717-7175 Fax: 886-2-545-0139

## EUROPE

### United Kingdom

Arizona Microchip Technology Ltd.  
Unit 6, The Courtyard  
Meadow Bank, Furlong Road  
Bourne End, Buckinghamshire SL8 5AJ  
Tel: 44-1628-851077 Fax: 44-1628-850259

### France

Arizona Microchip Technology SARL  
Zone Industrielle de la Bonde  
2 Rue du Buisson aux Fraises  
91300 Massy, France  
Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79

### Germany

Arizona Microchip Technology GmbH  
Gustav-Heinemann-Ring 125  
D-81739 München, Germany  
Tel: 49-89-627-144 0 Fax: 49-89-627-144-44

### Italy

Arizona Microchip Technology SRL  
Centro Direzionale Colleone  
Palazzo Taurus 1 V. Le Colleoni 1  
20041 Agrate Brianza  
Milan, Italy  
Tel: 39-39-6899939 Fax: 39-39-6899883

## JAPAN

Microchip Technology Intl. Inc.  
Benex S-1 6F  
3-18-20, Shin Yokohama  
Kohoku-Ku, Yokohama  
Kanagawa 222 Japan  
Tel: 81-4-5471-6166 Fax: 81-4-5471-6122

3/24/97



**MICROCHIP**

All rights reserved. © 1997, Microchip Technology Incorporated, USA.

---

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip's products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights. The Microchip logo and name are registered trademarks of Microchip Technology Inc. All rights reserved. All other trademarks mentioned herein are the property of their respective companies.