

EPROM-Based 8-Bit CMOS Microcontrollers

Devices Included in this Data Sheet:

Referred to collectively as PIC16C55X.

- PIC16C554
- PIC16C557
- PIC16C558

High Performance RISC CPU:

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

| Device | Program Memory | Data Memory |
|-----------|----------------|-------------|
| PIC16C554 | 512 | 80 |
| PIC16C557 | 2 K | 128 |
| PIC16C558 | 2 K | 128 |

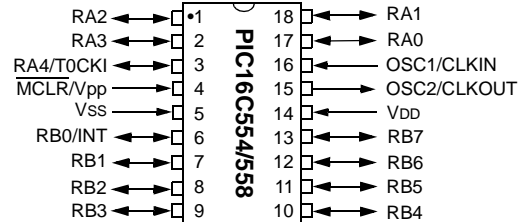
- Interrupt capability
- 16-18 special function hardware registers
- 8-level deep hardware stack
- Direct, Indirect and Relative Addressing modes

Peripheral Features:

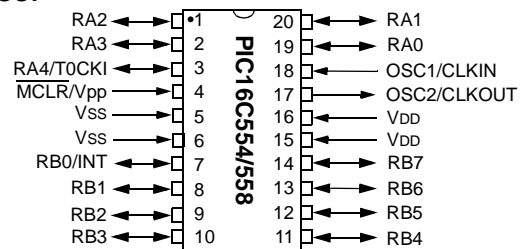
- 13-22 I/O pins with individual direction control
 - Pull-up resistors on PORTB
- High current sink/source for direct LED drive
- Timer0: 8-bit timer/counter with 8-bit programmable prescaler

Pin Diagram

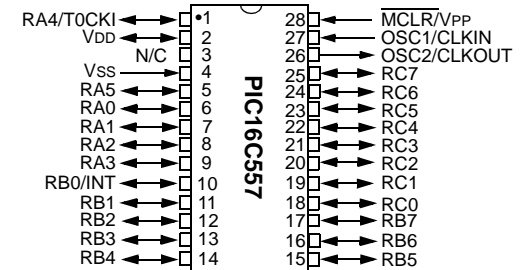
PDIP, SOIC, Windowed Cerdip



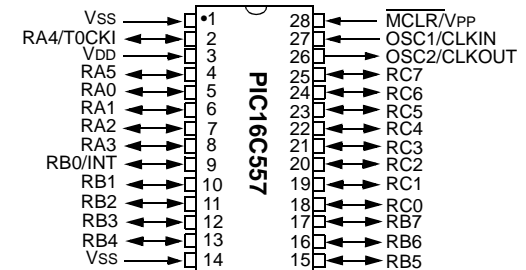
SSOP



PDIP, SOIC, Windowed Cerdip



SSOP



PIC16C55X

Special Microcontroller Features:

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

Note: For additional information on enhancements, see Appendix A

CMOS Technology:

- Low power, high speed CMOS EPROM technology
- Fully static design
- Wide operating voltage range
 - 2.5V to 5.5V
- Commercial, Industrial and Extended temperature range
- Low power consumption
 - < 2.0 mA @ 5.0V, 4.0 MHz
 - 15 μ A typical 3.0V, 32 kHz
 - < 1.0 μ A typical standby current @ 3.0V

Device Differences

| Device | Voltage Range | Oscillator |
|-----------|---------------|------------|
| PIC16C554 | 2.5 - 5.5 | (Note 1) |
| PIC16C557 | 2.5 - 5.5 | (Note 1) |
| PIC16C558 | 2.5 - 5.5 | (Note 1) |

Note 1: If you change from this device to another device, please verify oscillator characteristics in your application.

Table of Contents

| | | |
|--|-----------------------------------|-----|
| 1.0 | General Description..... | 5 |
| 2.0 | PIC16C55X Device Varieties | 7 |
| 3.0 | Architectural Overview | 9 |
| 4.0 | Memory Organization | 13 |
| 5.0 | I/O Ports | 23 |
| 6.0 | Special Features of the CPU | 31 |
| 7.0 | Timer0 Module | 47 |
| 8.0 | Instruction Set Summary | 53 |
| 9.0 | Development Support..... | 67 |
| 10.0 | Electrical Specifications..... | 73 |
| 11.0 | Packaging Information..... | 87 |
| Appendix A: | Enhancements..... | 97 |
| Appendix B: | Compatibility | 97 |
| Index | | 99 |
| On-Line Support..... | | 101 |
| Systems Information and Upgrade Hot Line | | 101 |
| Reader Response | | 102 |
| Product Identification System | | 103 |

TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at docerrors@mail.microchip.com or fax the **Reader Response Form** in the back of this data sheet to (480) 792-4150. We welcome your feedback.

Most Current Data Sheet

To obtain the most up-to-date version of this data sheet, please register at our Worldwide Web site at:

<http://www.microchip.com>

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000A is version A of document DS30000).

Errata

An errata sheet, describing minor operational differences from the data sheet and recommended workarounds, may exist for current devices. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please check with one of the following:

- Microchip's Worldwide Web site; <http://www.microchip.com>
- Your local Microchip sales office (see last page)
- The Microchip Corporate Literature Center; U.S. FAX: (480) 792-7277

When contacting a sales office or the literature center, please specify which device, revision of silicon and data sheet (include literature number) you are using.

Customer Notification System

Register on our web site at www.microchip.com/cn to receive the most current information on all of our products.

PIC16C55X

NOTES:

1.0 GENERAL DESCRIPTION

The PIC16C55X are 18, 20 and 28-Pin EPROM-based members of the versatile PIC16CXX family of low cost, high performance, CMOS, fully-static, 8-bit microcontrollers.

All PIC[®] microcontrollers employ an advanced RISC architecture. The PIC16C55X 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.

PIC16C55X microcontrollers typically achieve a 2:1 code compression and a 4:1 speed improvement over other 8-bit microcontrollers in their class.

The PIC16C554 has 80 bytes of RAM. The PIC16C557 and PIC16C558 have 128 bytes of RAM. The PIC16C554 and PIC16C558 have 13 I/O pins and an 8-bit timer/counter with an 8-bit programmable prescaler. The PIC16C557 has 22 I/O pins and an 8-bit timer/counter with an 8-bit programmable prescaler.

PIC16C55X devices have special features to reduce external components, thus reducing cost, enhancing system reliability and reducing power consumption. There are four oscillator options, of which the single pin RC oscillator provides a low cost solution, the LP oscillator minimizes power consumption, XT is a standard crystal, and the HS is for high speed crystals. The SLEEP (power-down) mode offers power saving. The user can wake-up the chip from SLEEP through several external and internal interrupts and RESET.

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

A UV-erasable Cerdip packaged version is ideal for code development while the cost effective One-Time Programmable (OTP) version is suitable for production in any volume.

Table 1-1 shows the features of the PIC16C55X mid-range microcontroller families.

A simplified block diagram of the PIC16C55X is shown in Figure 3-1.

The PIC16C55X series fit perfectly in applications ranging from motor control to low power remote sensors. The EPROM technology makes customization of application programs (detection levels, pulse generation, timers, etc.) extremely fast and convenient. The small footprint packages make this microcontroller series perfect for all applications with space limitations. Low cost, low power, high performance, ease of use and I/O flexibility make the PIC16C55X very versatile.

1.1 Family and Upward Compatibility

Users familiar with the family of microcontrollers will realize that this is an enhanced version of the architecture. Please refer to Appendix A for a detailed list of enhancements. Code written for can be easily ported to PIC16C55X family of devices (Appendix B).

The PIC16C55X family fills the niche for users wanting to migrate up from the family and not needing various peripheral features of other members of the PIC16XX mid-range microcontroller family.

1.2 Development Support

The PIC16C55X family 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.

PIC16C55X

TABLE 1-1: PIC16C55X FAMILY OF DEVICES

| | | PIC16C554 | PIC16C557 | PIC16C558 |
|---|--------------------------------------|----------------------------------|----------------------------------|---------------------------|
| Clock | Maximum Frequency of Operation (MHz) | 20 | 20 | 20 |
| | | | | |
| Memory | EPROM Program Memory (x14 words) | 512 | 2K | 2K |
| | Data Memory (bytes) | 80 | 128 | 128 |
| Peripherals | Timer Module(s) | TMR0 | TMR0 | TMR0 |
| Features | Interrupt Sources | 3 | 3 | 3 |
| | I/O Pins | 13 | 22 | 13 |
| | Voltage Range (Volts) | 2.5-5.5 | 2.5-5.5 | 2.5-5.5 |
| | Brown-out Reset | — | — | — |
| | Packages | 18-pin DIP, SOIC; 20-pin SSOP | 28-pin DIP, SOIC; 28-pin SSOP | 18-pin DIP, SOIC, SSOP |
| All PIC [®] Family devices have Power-on Reset, selectable Watchdog Timer, selectable code protect and high I/O current capability. All PIC16C55X Family devices use serial programming with clock pin RB6 and data pin RB7. | | | | |

2.0 PIC16C55X 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 PIC16C55X Product Identification System section at the end of this data sheet. When placing orders, please use this page of the data sheet to specify the correct part number.

2.1 UV Erasable Devices

The UV erasable version, offered in CERDIP package, is optimal for prototype development and pilot programs. This version can be erased and reprogrammed to any of the oscillator modes.

Microchip's PICSTART[®] and PROMATE[®] programmers both support programming of the PIC16C55X.

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. 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 Microchip Technology sales office for more details.

2.4 Serialized Quick-Turnaround Production (SQTPSM) 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.

PIC16C55X

NOTES:

3.0 ARCHITECTURAL OVERVIEW

The high performance of the PIC16C55X family can be attributed to a number of architectural features commonly found in RISC microprocessors. To begin with, the PIC16C55X uses a Harvard architecture in which program and data are accessed from separate memories using separate busses. This improves bandwidth over traditional von Neumann architecture where program and data are fetched from the same memory. Separating program and data memory further allows instructions to be sized differently from 8-bit wide data words. Instruction opcodes are 14-bit wide making it possible to have all single word instructions. A 14-bit wide program memory access bus fetches a 14-bit instruction in a single cycle. A two-stage pipeline overlaps fetch and execution of instructions. Consequently, all instructions (35) execute in a single-cycle (200 ns @ 20 MHz) except for program branches. The table below lists the memory (EPROM and RAM).

| Device | Program Memory (EPROM) | Data Memor (RAM) |
|-----------|------------------------|------------------|
| PIC16C554 | 512 | 80 |
| PIC16C557 | 2 K | 128 |
| PIC16C558 | 2 K | 128 |

The PIC16C554 addresses 512 x 14 on-chip program memory. The PIC16C557 and PIC16C558 addresses 2 K x 14 program memory. All program memory is internal.

The PIC16C55X can directly or indirectly address its register files or data memory. All special function registers, including the program counter, are mapped into the data memory. The PIC16C55X 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 PIC16C55X simple yet efficient. In addition, the learning curve is reduced significantly.

The PIC16C55X devices contain an 8-bit ALU and working register. The ALU is a general purpose arithmetic unit. It performs arithmetic and Boolean functions between data in the working register and any register file.

The ALU is 8-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 and Digit Borrow out bit, respectively, in subtraction. See the `SUBLW` and `SUBWF` instructions for examples.

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

PIC16C55X

FIGURE 3-1: BLOCK DIAGRAM

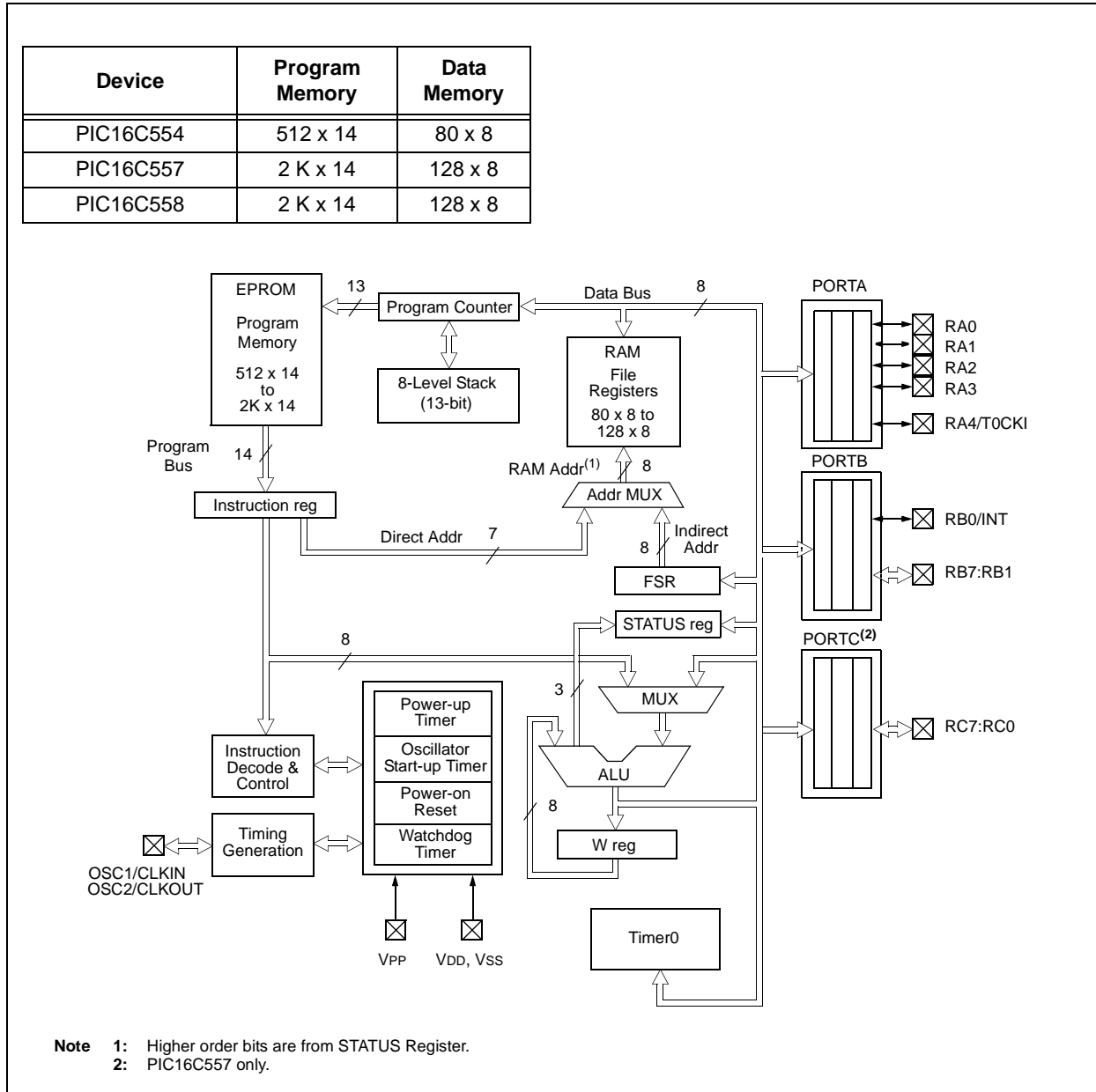


TABLE 3-1: PIC16C55X PINOUT DESCRIPTION

| Name | Pin Number | | | Pin Type | Buffer Type | Description |
|--------------------|------------|------|-------|----------|-----------------------|--|
| | PDIP | SOIC | SSOP | | | |
| OSC1/CLKIN | 16 | 16 | 18 | I | ST/CMOS | Oscillator crystal input/external clock source output. |
| OSC2/CLKOUT | 15 | 15 | 17 | O | — | Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode. In RC mode, OSC2 pin outputs CLKOUT which has 1/4 the frequency of OSC1, and denotes the instruction cycle rate. |
| MCLR/VPP | 4 | 4 | 4 | I/P | ST | Master clear (Reset) input/programming voltage input. This pin is an active low RESET to the device. |
| RA0 | 17 | 17 | 19 | I/O | ST | Bi-directional I/O port |
| RA1 | 18 | 18 | 20 | I/O | ST | Bi-directional I/O port |
| RA2 | 1 | 1 | 1 | I/O | ST | Bi-directional I/O port |
| RA3 | 2 | 2 | 2 | I/O | ST | Bi-directional I/O port |
| RA4/T0CKI | 3 | 3 | 3 | I/O | ST | Bi-directional I/O port or external clock input for TMR0. Output is open drain type. |
| RB0/INT | 6 | 6 | 7 | I/O | TTL/ST ⁽¹⁾ | Bi-directional I/O port can be software programmed for internal weak pull-up. RB0/INT can also be selected as an external interrupt pin. |
| RB1 | 7 | 7 | 8 | I/O | TTL | Bi-directional I/O port can be software programmed for internal weak pull-up. |
| RB2 | 8 | 8 | 9 | I/O | TTL | Bi-directional I/O port can be software programmed for internal weak pull-up. |
| RB3 | 9 | 9 | 10 | I/O | TTL | Bi-directional I/O port can be software programmed for internal weak pull-up. |
| RB4 | 10 | 10 | 11 | I/O | TTL | Bi-directional I/O port can be software programmed for internal weak pull-up. Interrupt-on-change pin. |
| RB5 | 11 | 11 | 12 | I/O | TTL | Bi-directional I/O port can be software programmed for internal weak pull-up. Interrupt-on-change pin. |
| RB6 | 12 | 12 | 13 | I/O | TTL/ST ⁽²⁾ | Bi-directional I/O port can be software programmed for internal weak pull-up. Interrupt-on-change pin. Serial programming clock. |
| RB7 | 13 | 13 | 14 | I/O | TTL/ST ⁽²⁾ | Bi-directional I/O port can be software programmed for internal weak pull-up. Interrupt-on-change pin. Serial programming data. |
| RC0 ⁽³⁾ | 18 | 18 | 18 | I/O | TTL | Bi-directional I/O port input buffer. |
| RC1 ⁽³⁾ | 19 | 19 | 19 | I/O | TTL | Bi-directional I/O port input buffer. |
| RC2 ⁽³⁾ | 20 | 20 | 20 | I/O | TTL | Bi-directional I/O port input buffer. |
| RC3 ⁽³⁾ | 21 | 21 | 21 | I/O | TTL | Bi-directional I/O port input buffer. |
| RC4 ⁽³⁾ | 22 | 22 | 22 | I/O | TTL | Bi-directional I/O port input buffer. |
| RC5 ⁽³⁾ | 23 | 23 | 23 | I/O | TTL | Bi-directional I/O port input buffer. |
| RC6 ⁽³⁾ | 24 | 24 | 24 | I/O | TTL | Bi-directional I/O port input buffer. |
| RC7 ⁽³⁾ | 25 | 25 | 25 | I/O | TTL | Bi-directional I/O port input buffer. |
| VSS | 5 | 5 | 5,6 | P | — | Ground reference for logic and I/O pins. |
| VDD | 14 | 14 | 15,16 | P | — | Positive supply for logic and I/O pins. |

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

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.
 2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.
 3: PIC16C557 only.

PIC16C55X

3.1 Clocking Scheme/Instruction Cycle

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

3.2 Instruction Flow/Pipelining

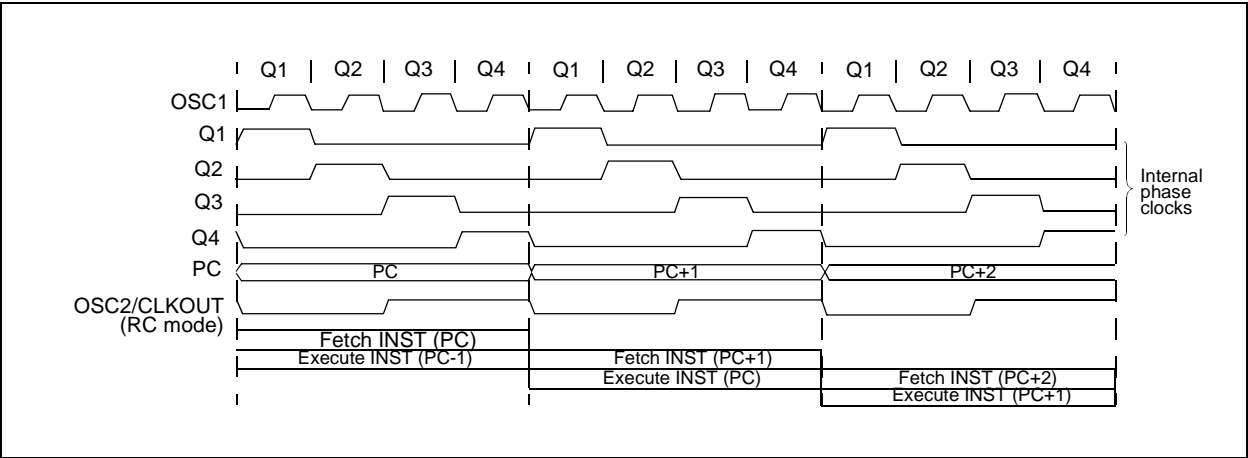
An “Instruction Cycle” consists of four Q cycles (Q1, Q2, Q3 and Q4). The instruction fetch and execute are pipelined such that fetch takes one instruction cycle

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).

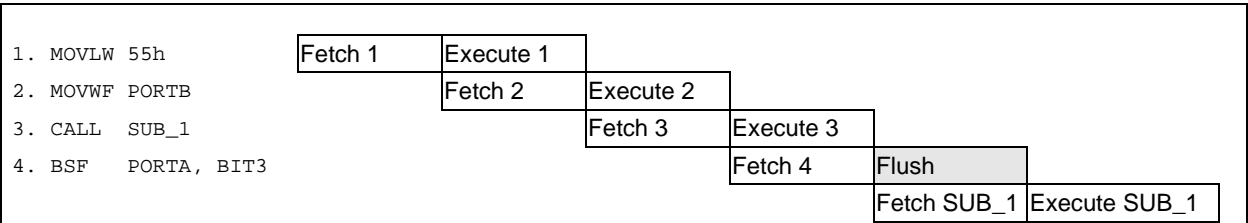
A fetch cycle begins with the program counter (PC) incrementing in Q1.

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



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

4.0 MEMORY ORGANIZATION

4.1 Program Memory Organization

The PIC16C55X has a 13-bit program counter capable of addressing an 8 K x 14 program memory space. Only the first 512 x 14 (0000h - 01FFh) for the PIC16C554 and 2K x 14 (0000h - 07FFh) for the PIC16C557 and PIC16C558 are physically implemented. Accessing a location above these boundaries will cause a wrap-around within the first 512 x 14 spaces in the PIC16C554, or 2K x 14 space of the PIC16C558 and PIC16C557. The RESET vector is at 0000h and the interrupt vector is at 0004h (Figure 4-1, Figure 4-2).

FIGURE 4-1: PROGRAM MEMORY MAP AND STACK FOR THE PIC16C554

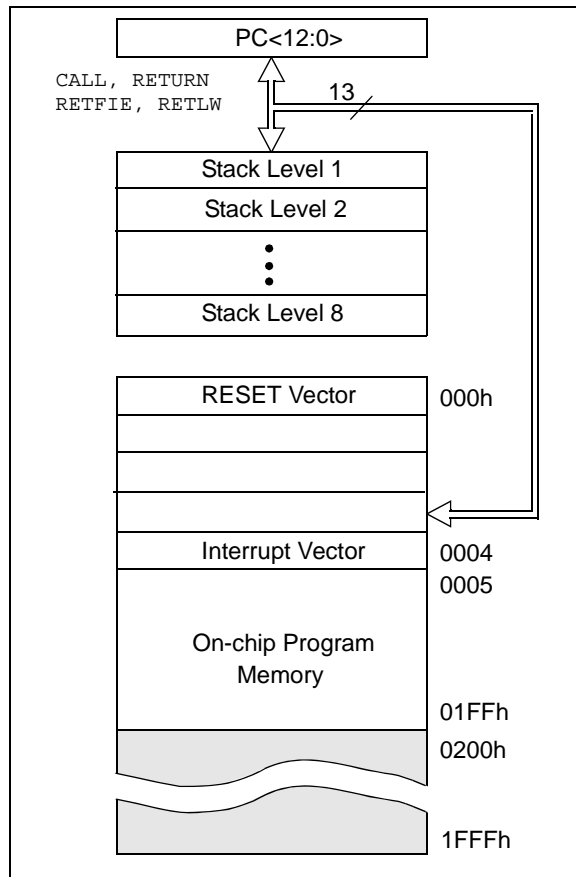
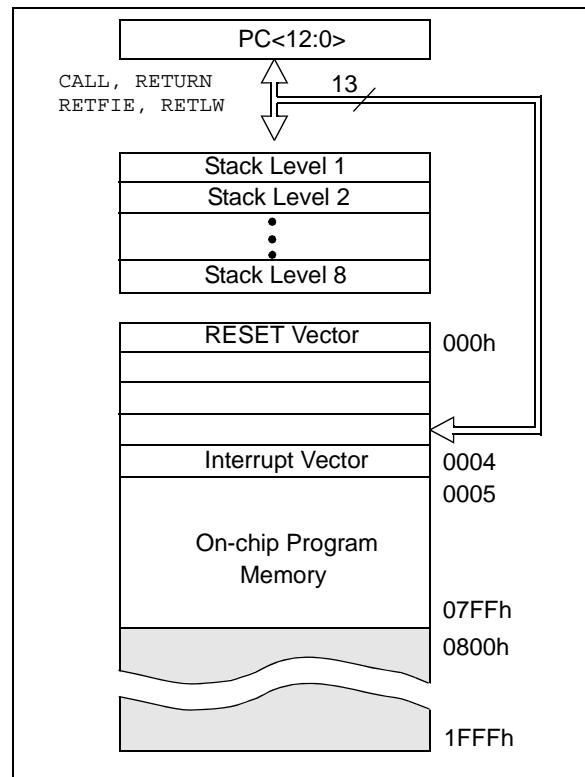


FIGURE 4-2: PROGRAM MEMORY MAP AND STACK FOR THE PIC16C557 AND PIC16C558



4.2 Data Memory Organization

The data memory (Figure 4-3 through Figure 4-5) is partitioned into two banks which contain the General Purpose Registers (GPR) and the Special Function Registers (SFR). Bank 0 is selected when the RP0 bit (STATUS <5>) is cleared. Bank 1 is selected when the RP0 bit is set. The Special Function Registers are located in the first 32 locations of each Bank. Register locations 20-6Fh (Bank 0) on the PIC16C554 and 20-7Fh (Bank 0) and A0-BFh (Bank 1) on the PIC16C558 and PIC16C557 are General Purpose Registers implemented as static RAM. Some special purpose registers are mapped in Bank 1.

4.2.1 GENERAL PURPOSE REGISTER FILE

The register file is organized as 80 x 8 in the PIC16C554 and 128 x 8 in the PIC16C557 and PIC16C558. Each can be accessed either directly or indirectly through the File Select Register, FSR (Section 4.4).

PIC16C55X

FIGURE 4-3: DATA MEMORY MAP FOR THE PIC16C554

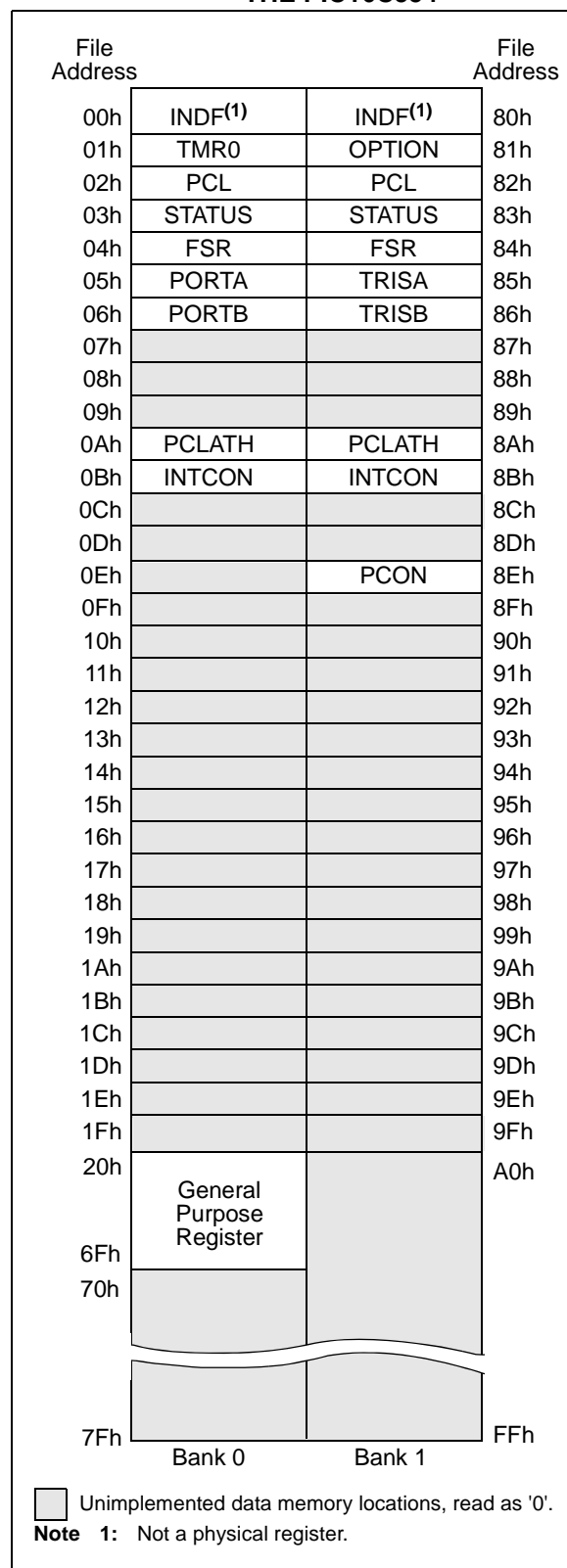


FIGURE 4-4: DATA MEMORY MAP FOR THE PIC16C557

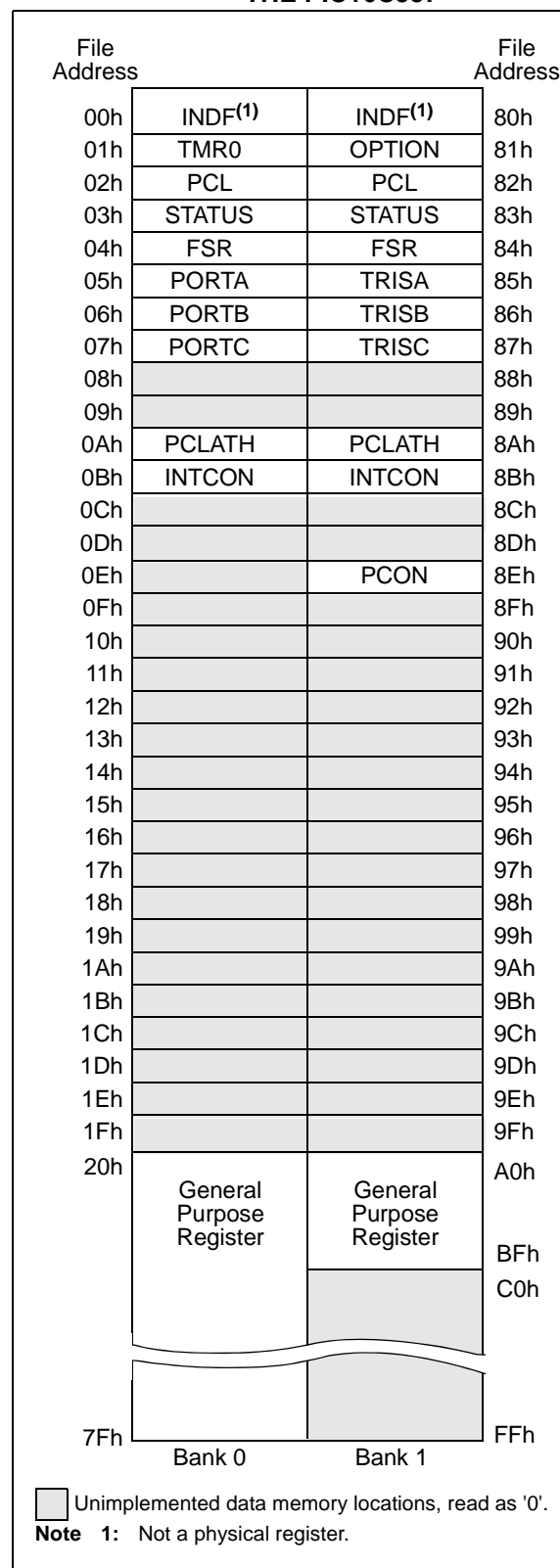


FIGURE 4-5: DATA MEMORY MAP FOR THE PIC16C558

| File Address | | | File Address |
|---|--------------------------|--------------------------|--------------|
| 00h | INDF ⁽¹⁾ | INDF ⁽¹⁾ | 80h |
| 01h | TMR0 | OPTION | 81h |
| 02h | PCL | PCL | 82h |
| 03h | STATUS | STATUS | 83h |
| 04h | FSR | FSR | 84h |
| 05h | PORTA | TRISA | 85h |
| 06h | PORTB | TRISB | 86h |
| 07h | | | 87h |
| 08h | | | 88h |
| 09h | | | 89h |
| 0Ah | PCLATH | PCLATH | 8Ah |
| 0Bh | INTCON | INTCON | 8Bh |
| 0Ch | | | 8Ch |
| 0Dh | | | 8Dh |
| 0Eh | | PCON | 8Eh |
| 0Fh | | | 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 | | | 9Eh |
| 1Fh | | | 9Fh |
| 20h | General Purpose Register | General Purpose Register | A0h |
| | | | BFh |
| | | | C0h |
| 7Fh | Bank 0 | Bank 1 | FFh |
| | | | |
| <div><div></div> Unimplemented data memory locations, read as '0'.</div> <div>Note 1: Not a physical register.</div> | | | |

4.2.2 SPECIAL FUNCTION REGISTERS

The Special Function Registers are registers used by the CPU and peripheral functions for controlling the desired operation of the device (Table 4-1). These registers are static RAM.

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

PIC16C55X

TABLE 4-1: SPECIAL REGISTERS FOR THE PIC16C55X

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR Reset | Detail on Page: |
|---------|----------------------|--|--------------------|--------|--|-----------------|--------|------------------|-----------|--------------------|-----------------|
| Bank 0 | | | | | | | | | | | |
| 00h | INDF | Addressing this location uses contents of FSR to address data memory (not a physical register) | | | | | | | | xxxx xxxx | 21 |
| 01h | TMR0 | Timer0 Module's Register | | | | | | | | xxxx xxxx | 47 |
| 02h | PCL | Program Counter's (PC) Least Significant Byte | | | | | | | | 0000 0000 | 21 |
| 03h | STATUS | IRP ⁽²⁾ | RP1 ⁽²⁾ | RP0 | \overline{TO} | \overline{PD} | Z | DC | C | 0001 1xxx | 17 |
| 04h | FSR | Indirect data memory address pointer | | | | | | | | xxxx xxxx | 21 |
| 05h | PORTA | — | — | — | RA4 | RA3 | RA2 | RA1 | RA0 | ---x xxxx | 23 |
| 06h | PORTB | RB7 | RB6 | RB5 | RB4 | RB3 | RB2 | RB1 | RB0 | xxxx xxxx | 25 |
| 07h | PORTC ⁽⁴⁾ | RC7 | RC6 | RC5 | RC4 | RC3 | RC2 | RC1 | RC0 | xxxx xxxx | 27 |
| 08h | — | Unimplemented | | | | | | | | — | — |
| 09h | — | Unimplemented | | | | | | | | — | — |
| 0Ah | PCLATH | — | — | — | Write buffer for upper 5 bits of program counter | | | | ---0 0000 | 21 | |
| 0Bh | INTCON | GIE | (3) | TOIE | INTE | RBIE | TOIF | INTF | RBIF | 0000 000x | 19 |
| 0Ch | — | Unimplemented | | | | | | | | — | — |
| 0Dh-1Eh | — | Unimplemented | | | | | | | | — | — |
| 1Fh | — | Unimplemented | | | | | | | | — | — |
| Bank 1 | | | | | | | | | | | |
| 80h | INDF | Addressing this location uses contents of FSR to address data memory (not a physical register) | | | | | | | | xxxx xxxx | 21 |
| 81h | OPTION | \overline{RBPU} | INTEDG | T0CS | T0SE | PSA | PS2 | PS1 | PS0 | 1111 1111 | 18 |
| 82h | PCL | Program Counter's (PC) Least Significant Byte | | | | | | | | 0000 0000 | 21 |
| 83h | STATUS | — | — | RP0 | \overline{TO} | \overline{PD} | Z | DC | C | 0001 1xxx | 17 |
| 84h | FSR | Indirect data memory address pointer | | | | | | | | xxxx xxxx | 21 |
| 85h | TRISA | — | — | — | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | ---1 1111 | 23 |
| 86h | TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 1111 1111 | 25 |
| 87h | TRISC ⁽⁴⁾ | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 1111 1111 | 27 |
| 88h | — | Unimplemented | | | | | | | | — | — |
| 89h | — | Unimplemented | | | | | | | | — | — |
| 8Ah | PCLATH | — | — | — | Write buffer for upper 5 bits of program counter | | | | ---0 0000 | 21 | |
| 8Bh | INTCON | GIE | (3) | TOIE | INTE | RBIE | TOIF | INTF | RBIF | 0000 000x | 19 |
| 8Ch | — | Unimplemented | | | | | | | | — | — |
| 8Dh | — | Unimplemented | | | | | | | | — | — |
| 8Eh | PCON | — | — | — | — | — | — | \overline{POR} | — | ---- --0- | 20 |
| 8Fh-9Eh | — | Unimplemented | | | | | | | | — | — |
| 9Fh | — | Unimplemented | | | | | | | | — | — |

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

- Note** 1: Other (non Power-up) Resets include \overline{MCLR} Reset and Watchdog Timer Reset during normal operation.
 2: IRP & RP1 bits are reserved, always maintain these bits clear.
 3: Bit 6 of INTCON register is reserved for future use. Always maintain this bit as clear.
 4: PIC16C557 only.

4.2.2.1 STATUS Register

The STATUS register, shown in Figure 4-2, contains the arithmetic status of the ALU, the RESET status and the bank select bits for data memory.

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

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

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

Note 1: The IRP and RP1 bits (STATUS<7:6>) are not used by the PIC16C55X and should be programmed as '0'. Use of these bits as general purpose R/W bits is NOT recommended, since this may affect upward compatibility with future products.

2: The C and DC bits operate as a Borrow and Digit Borrow out bit, respectively, in subtraction. See the `SUBLW` and `SUBWF` instructions for examples.

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

| Reserved | Reserved | R/W-0 | R-1 | R-1 | R/W-x | R/W-x | R/W-x |
|----------|----------|-------|-----------------|-----------------|-------|-------|-------|
| IRP | RP1 | RP0 | \overline{TO} | \overline{PD} | Z | DC | C |
| bit7 | | | | | | | bit0 |

- bit 7 IRP:** Register Bank Select bit (used for Indirect addressing)
 1 = Bank 2, 3 (100h - 1FFh)
 0 = Bank 0, 1 (00h - FFh)
 The IRP bit is reserved on the PIC16C55X, always maintain this bit clear
- bit 6-5 RP1:RP0:** Register Bank Select bits (used for Direct addressing)
 11 = Bank 3 (180h - 1FFh)
 10 = Bank 2 (100h - 17Fh)
 01 = Bank 1 (80h - FFh)
 00 = Bank 0 (00h - 7Fh)
 Each bank is 128 bytes. The RP1 bit is reserved on the PIC16C55X, always maintain this bit clear.
- bit 4 \overline{TO} :** Timeout bit
 1 = After power-up, `CLRWDT` instruction, or `SLEEP` instruction
 0 = A WDT timeout occurred
- bit 3 \overline{PD} :** Power-down bit
 1 = After power-up or by the `CLRWDT` instruction
 0 = By execution of the `SLEEP` instruction
- bit 2 Z:** Zero bit
 1 = The result of an arithmetic or logic operation is zero
 0 = The result of an arithmetic or logic operation is not zero
- bit 1 DC:** Digit carry/borrow bit (`ADDWF`, `ADDLW`, `SUBLW`, `SUBWF` instructions) (for borrow the polarity is reversed)
 1 = A carry-out from the 4th low order bit of the result occurred
 0 = No carry-out from the 4th low order bit of the result
- bit 0 C:** Carry/borrow bit (`ADDWF`, `ADDLW`, `SUBLW`, `SUBWF` instructions)
 1 = A carry-out from the Most Significant bit of the result occurred
 0 = No carry-out from the Most Significant bit of the result occurred

Note 1: For borrow the polarity is reversed. A subtraction is executed by adding the two's complement of the second operand. For rotate (`RRF`, `RLF`) instructions, this bit is loaded with either the high or low order bit of the source register.

Legend:

| | | |
|--------------------------|------------------|------------------------------------|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| - n = Value at POR reset | '1' = Bit is set | '0' = Bit is cleared |
| | | x = Bit is unknown |

PIC16C55X

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 RB0/INT interrupt, TMR0 and the weak pull-ups on PORTB.

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

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

| R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|-------|--------|-------|-------|-------|-------|-------|-------|
| RBPU | INTEDG | T0CS | T0SE | PSA | PS2 | PS1 | PS0 |
| bit7 | | | | | | | bit0 |

- bit 7 **RBPU:** PORTB Pull-up Enable bit
1 = PORTB pull-ups are disabled
0 = PORTB pull-ups are enabled by individual port latch values
- bit 6 **INTEDG:** Interrupt Edge Select bit
1 = Interrupt on rising edge of RB0/INT pin
0 = Interrupt on falling edge of RB0/INT pin
- bit 5 **T0CS:** TMR0 Clock Source Select bit
1 = Transition on RA4/T0CKI pin
0 = Internal instruction cycle clock (CLKOUT)
- bit 4 **T0SE:** TMR0 Source Edge Select bit
1 = Increment on high-to-low transition on RA4/T0CKI pin
0 = Increment on low-to-high transition on RA4/T0CKI pin
- bit 3 **PSA:** Prescaler Assignment bit
1 = Prescaler is assigned to the WDT
0 = Prescaler is assigned to the Timer0 module
- bit 2-0 **PS2:PS0:** Prescaler Rate Select bits

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

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

- n = Value at POR reset

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

4.2.2.3 INTCON Register

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

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

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

| R/W-0 | Reserved | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-x |
|-------|----------|-------|-------|-------|-------|-------|-------|
| GIE | — | TOIE | INTE | RBIE | TOIF | INTF | RBIF |
| bit7 | | | | | | | bit0 |

- bit 7 **GIE:** Global Interrupt Enable bit
1 = Enables all un-masked interrupts
0 = Disables all interrupts
- bit 6 **Reserved:** For future use. Always maintain this bit clear.
- bit 5 **TOIE:** TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 interrupt
0 = Disables the TMR0 interrupt
- bit 4 **INTE:** RB0/INT External Interrupt Enable bit
1 = Enables the RB0/INT external interrupt
0 = Disables the RB0/INT external interrupt
- bit 3 **RBIE:** RB Port Change Interrupt Enable bit
1 = Enables the RB port change interrupt
0 = Disables the RB port change interrupt
- bit 2 **TOIF:** TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software)
0 = TMR0 register did not overflow
- bit 1 **INTF:** RB0/INT External Interrupt Flag bit
1 = The RB0/INT external interrupt occurred (must be cleared in software)
0 = The RB0/INT external interrupt did not occur
- bit 0 **RBIF:** RB Port Change Interrupt Flag bit
1 = When at least one of the RB7:RB4 pins changed state (must be cleared in software)
0 = None of the RB7:RB4 pins have changed state

Legend:

| | | |
|--------------------------|------------------|--|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| - n = Value at POR reset | '1' = Bit is set | '0' = Bit is cleared x = Bit is unknown |

PIC16C55X

4.2.2.4 PCON Register

The PCON register contains a flag bit to differentiate between a Power-on Reset, an external MCLR Reset or WDT Reset. See Section 6.3 and Section 6.4 for detailed RESET operation.

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

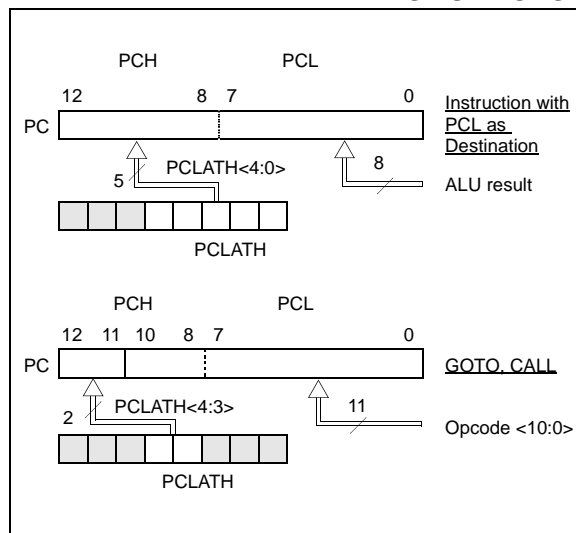
| | | | | | | | | |
|---------|---------------------------------------|-----|-----|-----|-----|-----|-------|------|
| | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0 | U-0 |
| | — | — | — | — | — | — | POR | — |
| bit7 | | | | | | | | bit0 |
| bit 7-2 | Unimplemented: Read as '0' | | | | | | | |
| bit 1 | POR: Power-on Reset status bit | | | | | | | |
| | 1 = No Power-on Reset occurred | | | | | | | |
| | 0 = Power-on Reset occurred | | | | | | | |
| bit 0 | Unimplemented: Read as '0' | | | | | | | |

| | | | |
|--------------------------|------------------|------------------------------------|--------------------|
| Legend: | | | |
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' | |
| - n = Value at POR reset | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

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 bits (PC<12:8>) are not directly readable or writable and come from PCLATH. On any RESET, the PC is cleared. Figure 4-6 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 Figure 4-6 shows how the PC is loaded during a CALL or GOTO instruction (PCLATH<4:3> → PCH).

FIGURE 4-6: 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 PIC16C55X family has an 8-level deep x 13-bit wide hardware stack (Figure 4-1 and Figure 4-2). The stack space is not part of either program or data space and the stack pointer is not readable or writable. The PC is PUSHed onto the stack when a CALL instruction is executed or an interrupt causes a branch. The stack is POPed in the event of a RETURN, RETLW or a RETFIE instruction execution. PCLATH is not affected by a PUSH or POP operation.

The stack operates as a circular buffer. This means that after the stack has been PUSHed eight times, the ninth push overwrites the value that was stored from the first push. The tenth push overwrites the second push (and so on).

Note 1: There are no status bits to indicate stack overflow or stack underflow conditions.

2: There are no instructions mnemonics called PUSH or POP. These are actions that occur from the execution of the CALL, RETURN, RETLW and RETFIE instructions, or vectoring to an interrupt address.

4.4 Indirect Addressing, INDF and FSR Registers

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

Indirect addressing is possible by using the INDF register. Any instruction using the INDF register actually accesses data pointed to by the file select register (FSR). Reading INDF itself indirectly will produce 00h. Writing to the INDF register indirectly results in a no-operation (although status bits may be affected). An effective 9-bit address is obtained by concatenating the 8-bit FSR register and the IRP bit (STATUS<7>), as shown in Figure 4-7. However, IRP is not used in the PIC16C55X.

A simple program to clear RAM locations 20h-2Fh using indirect addressing is shown in Example 4-1.

EXAMPLE 4-1: INDIRECT ADDRESSING

```

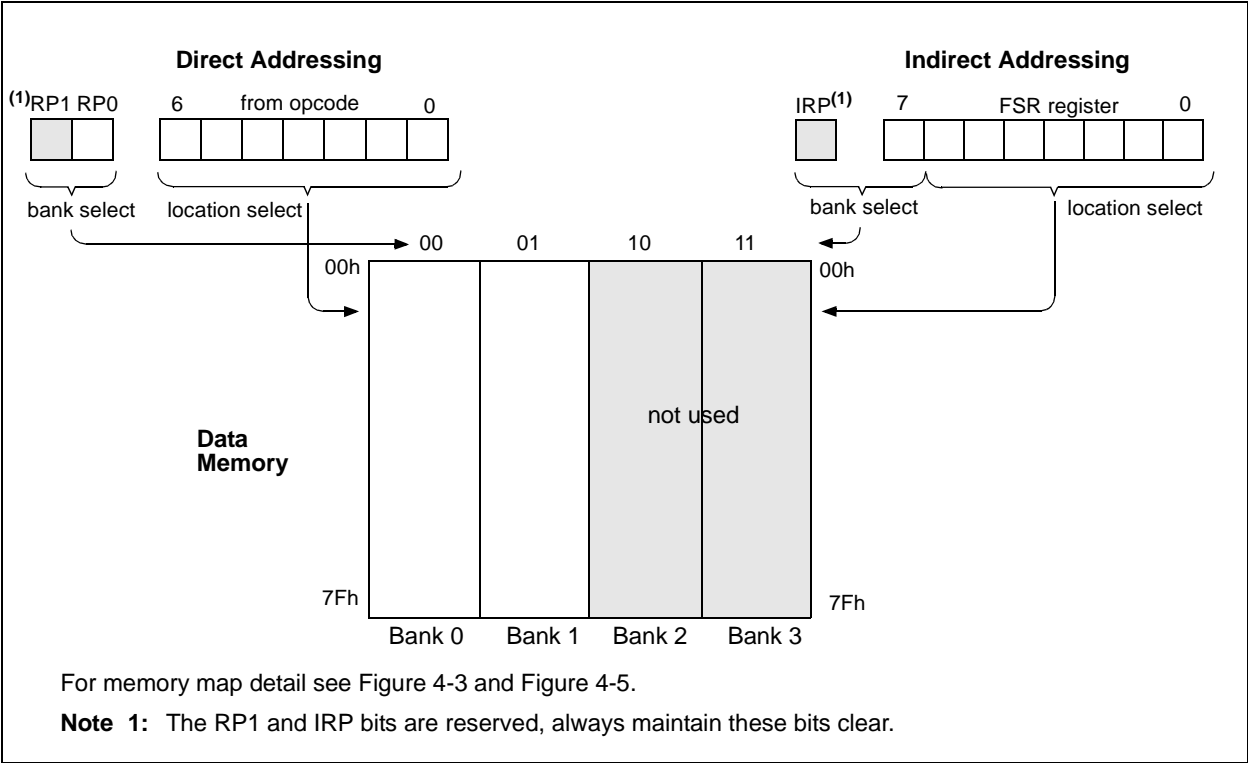
movlw 0x20 ;initialize pointer
movwf FSR ;to RAM
NEXT   clrf INDF ;clear INDF register
       incf FSR ;inc pointer
       btfss FSR,4 ;all done?
       goto NEXT ;no clear next
                           ;yes continue

```

CONTINUE:

PIC16C55X

FIGURE 4-7: DIRECT/INDIRECT ADDRESSING PIC16C55X



5.0 I/O PORTS

The PIC16C554 and PIC16C558 have two ports, PORTA and PORTB. The PIC16C557 has three ports, PORTA, PORTB and PORTC.

5.1 PORTA and TRISA Registers

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

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

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

Note 1: On RESET, the TRISA register is set to all inputs.

FIGURE 5-2: BLOCK DIAGRAM OF RA4 PIN

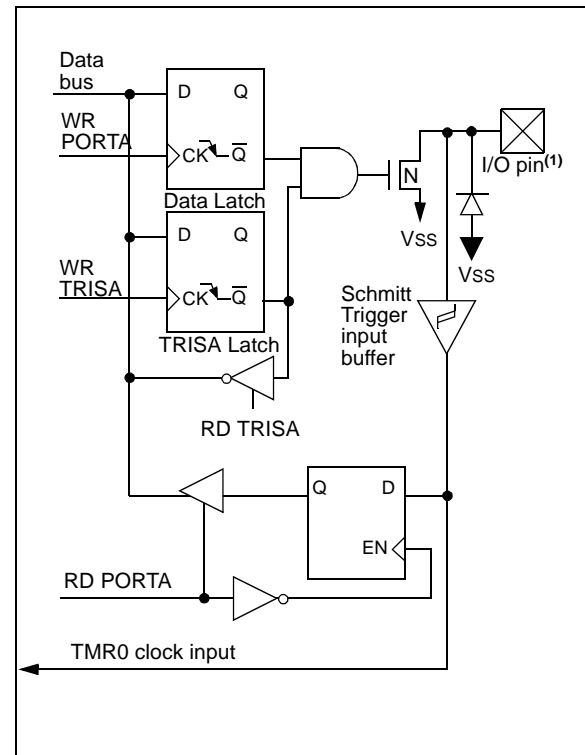
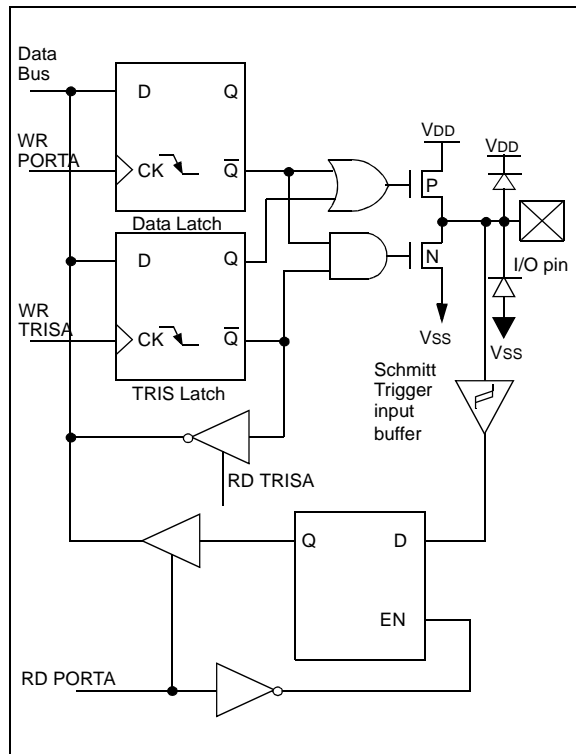


FIGURE 5-1: BLOCK DIAGRAM OF PORT PINS RA<3:0>



PIC16C55X

TABLE 5-1: PORTA FUNCTIONS

| Name | Bit # | Buffer Type | Function |
|-----------|-------|-------------|--|
| RA0 | Bit 0 | ST | Bi-directional I/O port. |
| RA1 | Bit 1 | ST | Bi-directional I/O port. |
| RA2 | Bit 2 | ST | Bi-directional I/O port. |
| RA3 | Bit 3 | ST | Bi-directional I/O port. |
| RA4/T0CKI | Bit 4 | ST | Bi-directional I/O port or external clock input for TMR0. Output is open drain type. |

Legend: ST = Schmitt Trigger input

TABLE 5-2: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA

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

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

Note 1: Shaded bits are not used by PORTA.

5.2 PORTB and TRISB Registers

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

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

Each of the PORTB pins has a weak internal pull-up ($\approx 200 \mu\text{A}$ typical). A single control bit can turn on all the pull-ups. This is done by clearing the $\overline{\text{RBP}}\text{U}$ (OPTION<7>) bit. The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on Power-on Reset.

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

latched in INTCON<0>). This interrupt can wake the device from SLEEP. The user, in the interrupt service routine, can clear the interrupt in the following manner:

- Any read or write of PORTB (this will end the mismatch condition)
- Clear flag bit RBIF

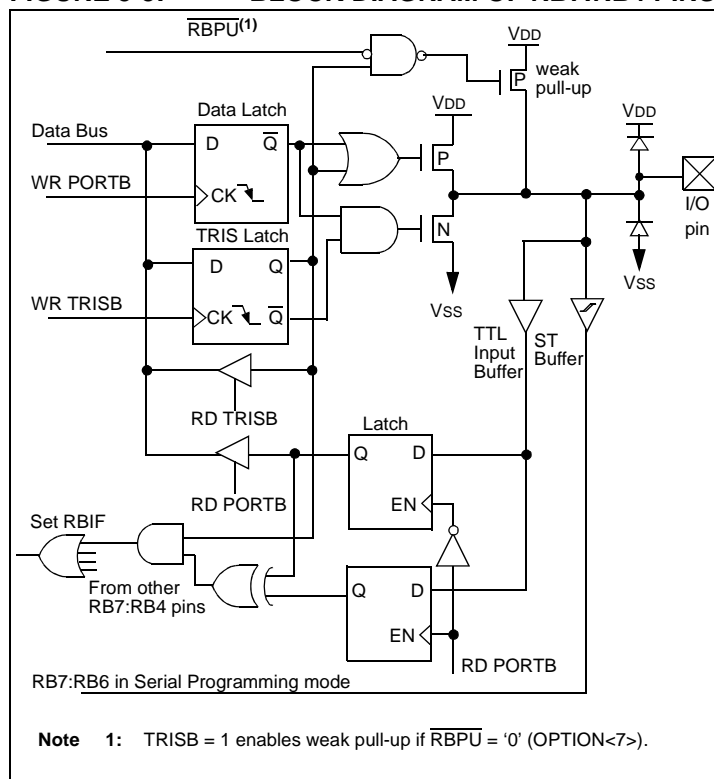
A mismatch condition will continue to set flag bit RBIF. Reading PORTB will end the mismatch condition, and allow flag bit RBIF to be cleared.

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

Note 1: If a change on the I/O pin should occur when the read operation is being executed (start of the Q2 cycle), then the RBIF interrupt flag may not get set.

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

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



The diagram illustrates the internal circuitry of the I/O pin driver for PORTB. It shows the interaction between the Data Bus, WR PORTB, WR TRISB, RD TRISB, RD PORTB, and RB0/INT signals. Key components include the Data Latch, TRIS Latch, RD PORTB, RD PORTB, and RD PORTB. The circuit also features a weak pull-up transistor connected to VDD and a pull-down transistor connected to VSS. The I/O pin is shown with its connection to VDD and VSS through diodes.

| Name | Bit # | Buffer Type | Function |
|---------|-------|-----------------------|--|
| RB0/INT | Bit 0 | TTL/ST ⁽¹⁾ | Bi-directional I/O port. Internal software programmable weak pull-up. |
| RB1 | Bit 1 | TTL | Bi-directional I/O port. Internal software programmable weak pull-up. |
| RB2 | Bit 2 | TTL | Bi-directional I/O port. Internal software programmable weak pull-up. |
| RB3 | Bit 3 | TTL | Bi-directional I/O port. Internal software programmable weak pull-up. |
| RB4 | Bit 4 | TTL | Bi-directional I/O port (with interrupt-on-change). Internal software programmable weak pull-up. |
| RB5 | Bit 5 | TTL | Bi-directional I/O port (with interrupt-on-change). Internal software programmable weak pull-up. |
| RB6 | Bit 6 | TTL/ST ⁽²⁾ | Bi-directional I/O port (with interrupt-on-change). Internal software programmable weak pull-up. Serial programming clock pin. |
| RB7 | Bit 7 | TTL/ST ⁽²⁾ | Bi-directional I/O port (with interrupt-on-change). Internal software programmable weak pull-up. Serial programming data pin. |

Note

- 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.
- 2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR | Value on All Other RESETS |
|----------|--------|---------------------------------|----------|--------|--------|--------|--------|--------|--------|--------------|---------------------------|
| 06h | PORTB | RB7 | RB6 | RB5 | RB4 | RB3 | RB2 | RB1 | RB0 | xxxx xxxx | uuuu uuuu |
| 86h | TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 1111 1111 | 1111 1111 |
| 81h | OPTION | $\overline{\text{RBP}}\text{U}$ | INTEDG | T0CS | T0SE | PSA | PS2 | PS1 | PS0 | 1111 1111 | 1111 1111 |
| 0BH, 8BH | INTCON | GIE | Reserved | T0IE | INTE | BRIE | T0IF | INTF | RBIF | 0000 000x | 0000 000x |

Note 1: Shaded bits are not used by PORTB.

5.3 PORTC and TRISC Registers⁽¹⁾

PORTC is a 8-bit wide latch. All pins have data direction bits (TRIS registers) which can configure these pins as input or output.

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

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

FIGURE 5-5: BLOCK DIAGRAM OF PORT PINS RC<7:0>

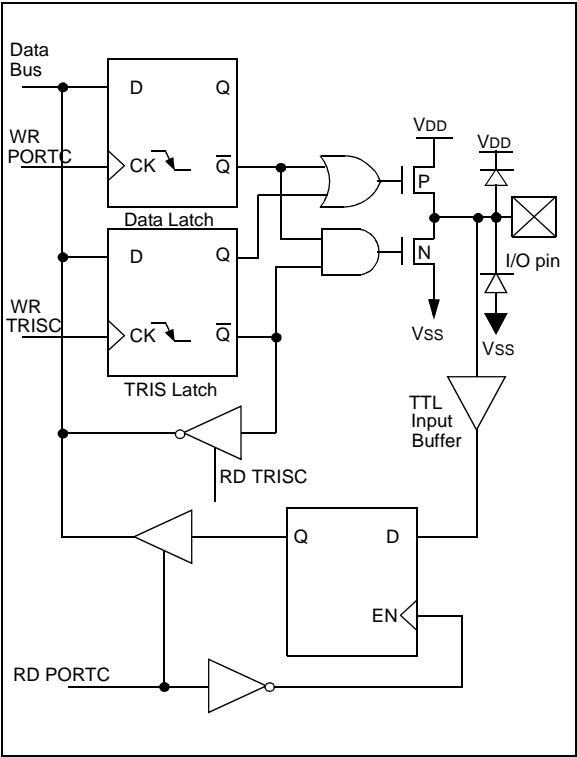


TABLE 5-5: PORTC FUNCTIONS

| Name | Bit # | Buffer Type | Function |
|------|-------|-------------|--------------------------|
| RC0 | Bit 0 | TTL | Bi-directional I/O port. |
| RC1 | Bit 1 | TTL | Bi-directional I/O port. |
| RC2 | Bit 2 | TTL | Bi-directional I/O port. |
| RC3 | Bit 3 | TTL | Bi-directional I/O port. |
| RC4 | Bit 4 | TTL | Bi-directional I/O port. |
| RC5 | Bit 5 | TTL | Bi-directional I/O port. |
| RC6 | Bit 6 | TTL | Bi-directional I/O port. |
| RC7 | Bit 7 | TTL | Bi-directional I/O port. |

Legend: ST = Schmitt Trigger, TTL = TTL input

TABLE 5-6: SUMMARY OF REGISTERS ASSOCIATED WITH PORTC AND TRISC

| 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 |
|---------|-------|--------|--------|--------|--------|--------|--------|--------|--------|--------------|---------------------------|
| 07h | PORTC | RC7 | RC6 | RC5 | RC4 | RC3 | RC2 | RC1 | RC0 | xxxx xxxx | uuuu uuuu |
| 87h | TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 1111 1111 | 1111 1111 |

Legend: x = unknown, u = unchanged
Note 1: PIC16C557 ONLY.

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 `PORTB` will cause all eight bits of `PORTB` to be read into the CPU. Then the `BSF` operation takes place on bit5 and `PORTB` is written to the output latches. If another bit of `PORTB` is used as a bi-directional I/O pin (e.g., bit 0) and it is defined as an input at this time, the input signal present on the pin itself would be read into the CPU and re-written to the data latch of this particular pin, overwriting the previous content. As long as the pin stays in the Input mode, no problem occurs. However, if bit 0 is switched into Output mode later on, the content of the data latch may now be unknown.

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

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

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

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

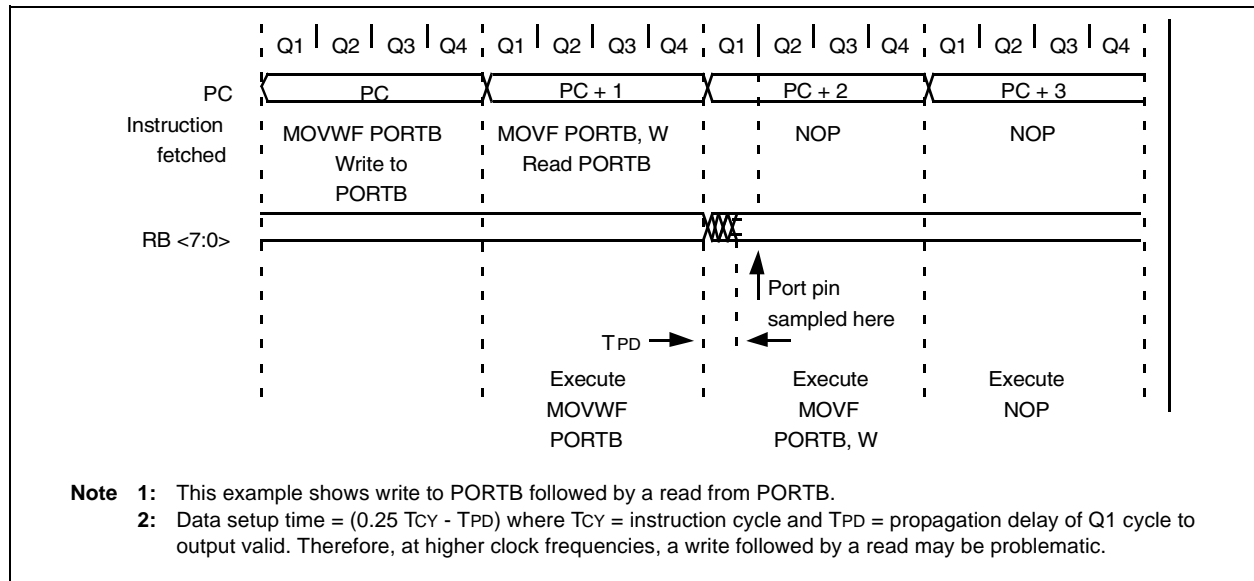
```
; Initial PORT settings: PORTB<7:4> Inputs
;
;                               PORTB<3:0> Outputs
; PORTB<7:6> have external pull-up and are
; not connected to other circuitry
;
;                               PORT latch PORT pins
;                               -----
;

BCF PORTB, 7    ; 01pp pppp 11pp pppp
BCF PORTB, 6    ; 10pp pppp 11pp pppp
BSF STATUS, RP0 ;
BCF TRISB, 7    ; 10pp pppp 11pp pppp
BCF TRISB, 6    ; 10pp pppp 10pp pppp
```

5.4.2 SUCCESSIVE OPERATIONS ON I/O PORTS

The actual write to an I/O port happens at the end of an instruction cycle, whereas for reading, the data must be valid at the beginning of the instruction cycle, as shown in Figure 5-6. Therefore, care must be exercised if a write followed by a read operation is carried out on the same I/O port. The sequence of instructions should be such to allow the pin voltage to stabilize (load dependent) before the next instruction which causes that file to be read into the CPU is executed. Otherwise, the previous state of that pin may be read into the CPU rather than the new state. When in doubt, it is better to separate these instructions with an NOP or another instruction not accessing this I/O port.

FIGURE 5-6: SUCCESSIVE I/O OPERATION



PIC16C55X

NOTES:

6.0 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 PIC16C55X family has a host of such features intended to maximize system reliability, minimize cost through elimination of external components, provide power saving operating modes and offer code protection.

These are:

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

The PIC16C55X has a Watchdog Timer which is controlled by configuration bits. It runs off its own RC oscillator for added reliability. There are two timers that offer necessary delays on power-up. One is the Oscillator Start-up Timer (OST), which is 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 functions on-chip, most applications need no external RESET circuitry.

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

6.1 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.

PIC16C55X

REGISTER 6-1: CONFIGURATION WORD

| | | | | | | | | | | | | | |
|--------|-----|-----|-----|-----|-----|---|----------|-----|-----|-------|------|-------|-------|
| CP1 | CP0 | CP1 | CP0 | CP1 | CP0 | — | Reserved | CP1 | CP0 | PWRTE | WDTE | FOSC1 | FOSC0 |
| bit 13 | | | | | | | | | | | | | bit 0 |

bit 13-8 **CP<1:0>**: Code protection bits⁽¹⁾
 bit 5-4 11 = Program Memory code protection off
 10 = 0400h - 07FFh code protected
 01 = 0200h - 07FFh code protected
 11 = 0000h - 07FFh code protected

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

bit 6 **Reserved**: Do not use

bit 3 **PWRTE**: Power-up Timer Enable bit
 1 = PWRT disabled
 0 = PWRT enabled

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

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

Note 1: All of the CP1:CP0 pairs have to be given the same value to enable the code protection scheme listed.

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

- n = Value at POR reset

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

6.2 Oscillator Configurations

6.2.1 OSCILLATOR TYPES

The PIC16C55X can be operated in four different oscillator options. The user can program two configuration bits (FOSC1 and FOSC0) to select one of these four modes:

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

6.2.2 CRYSTAL OSCILLATOR / CERAMIC RESONATORS

In XT, LP or HS modes a crystal or ceramic resonator is connected to the OSC1 and OSC2 pins to establish oscillation (Figure 6-1). The PIC16C55X oscillator design requires the use of a parallel cut crystal. Use of a series cut crystal may give a frequency out of the crystal manufacturers specifications. When in XT, LP or HS modes, the device can have an external clock source to drive the OSC1 pin (Figure 6-2).

FIGURE 6-1: CRYSTAL OPERATION (OR CERAMIC RESONATOR) (HS, XT OR LP OSC CONFIGURATION)

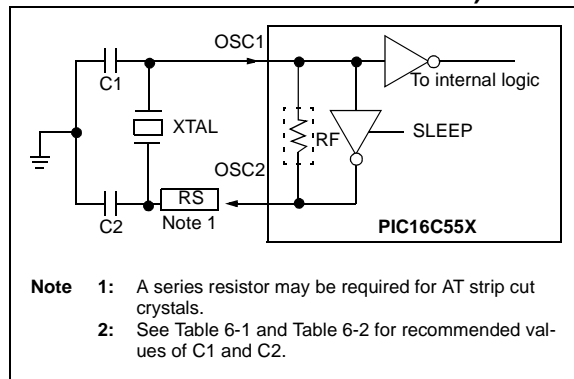


FIGURE 6-2: EXTERNAL CLOCK INPUT OPERATION (HS, XT OR LP OSC CONFIGURATION)

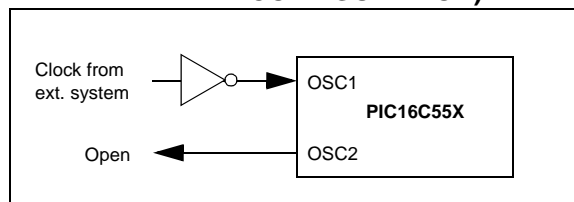


TABLE 6-1: CAPACITOR SELECTION FOR CERAMIC RESONATORS (PRELIMINARY)

| Ranges Characterized: | | | |
|-----------------------|----------|-------------|-------------|
| Mode | Freq | OSC1(C1) | OSC2(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 | 8.0 MHz | 10 - 68 pF | 10 - 68 pF |
| | 16.0 MHz | 10 - 22 pF | 10 - 22 pF |

Note 1: Higher capacitance increases the stability of the oscillator but also increases the start-up time. These values are for design guidance only. Since each resonator has its own characteristics, the user should consult with the resonator manufacturer for appropriate values of external components.

TABLE 6-2: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR (PRELIMINARY)

| Mode | Freq | OSC1(C1) | OSC2(C2) |
|------|---------|-------------|--------------|
| LP | 32 kHz | 68 - 100 pF | 68 - 100 pF |
| | 200 kHz | 15 - 30 pF | 15 - 30 pF |
| XT | 100 kHz | 68 - 150 pF | 150 - 200 pF |
| | 2 MHz | 15 - 30 pF | 15 - 30 pF |
| | 4 MHz | 15 - 30 pF | 15 - 30 pF |
| HS | 8 MHz | 15 - 30 pF | 15 - 30 pF |
| | 10 MHz | 15 - 30 pF | 15 - 30 pF |
| | 20 MHz | 15 - 30 pF | 15 - 30 pF |

Note 1: Higher capacitance increases the stability of the oscillator but also increases the start-up time. These values are for design guidance only. Rs may be required in HS mode as well as XT mode to avoid over-driving crystals with low-drive level specification. Since each crystal has its own characteristics, the user should consult with the crystal manufacturer for appropriate values of external components.

6.2.3 EXTERNAL CRYSTAL OSCILLATOR CIRCUIT

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

Figure 6-3 shows implementation of a parallel resonant oscillator circuit. The circuit is designed to use the fundamental frequency of the crystal. The 74AS04 inverter performs the 180° phase shift that a parallel oscillator requires. The 4.7 kΩ resistor provides the negative feedback for stability. The 10 kΩ potentiometers bias the 74AS04 in the linear region. This could be used for external oscillator designs.

FIGURE 6-3: EXTERNAL PARALLEL RESONANT CRYSTAL OSCILLATOR CIRCUIT

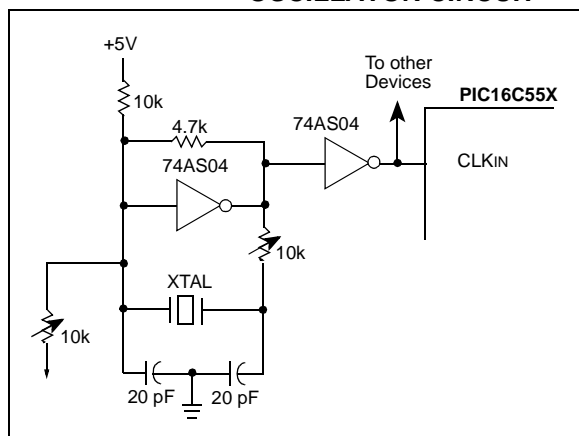
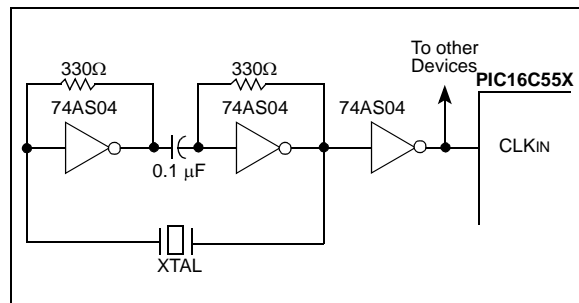


Figure 6-4 shows a series resonant oscillator circuit. This circuit is also designed to use the fundamental frequency of the crystal. The inverter performs a 180° phase shift in a series resonant oscillator circuit. The 330Ω resistors provide the negative feedback to bias the inverters in their linear region.

FIGURE 6-4: EXTERNAL SERIES RESONANT CRYSTAL OSCILLATOR CIRCUIT



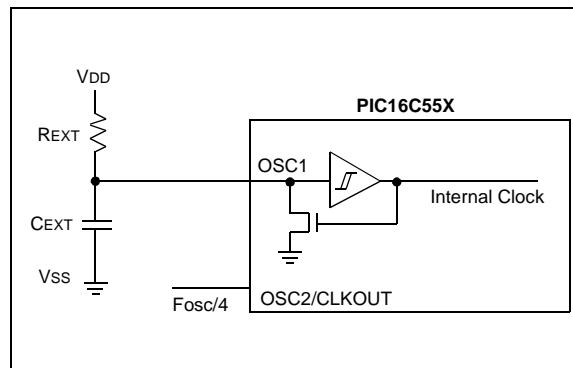
6.2.4 RC OSCILLATOR

For timing insensitive applications the "RC" device option offers additional cost savings. The RC oscillator frequency is a function of the supply voltage, the resistor (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 6-5 shows how the R/C combination is connected to the PIC16C55X. For R_{EXT} values below 2.2 kΩ, the oscillator operation may become unstable, or stop completely. For very high R_{EXT} values (e.g., 1 MΩ), the oscillator becomes sensitive to noise, humidity and leakage. Thus, we recommend to keep R_{EXT} between 3 kΩ and 100 kΩ.

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

FIGURE 6-5: RC OSCILLATOR MODE



6.3 RESET

The PIC16C55X differentiates between various kinds of RESET:

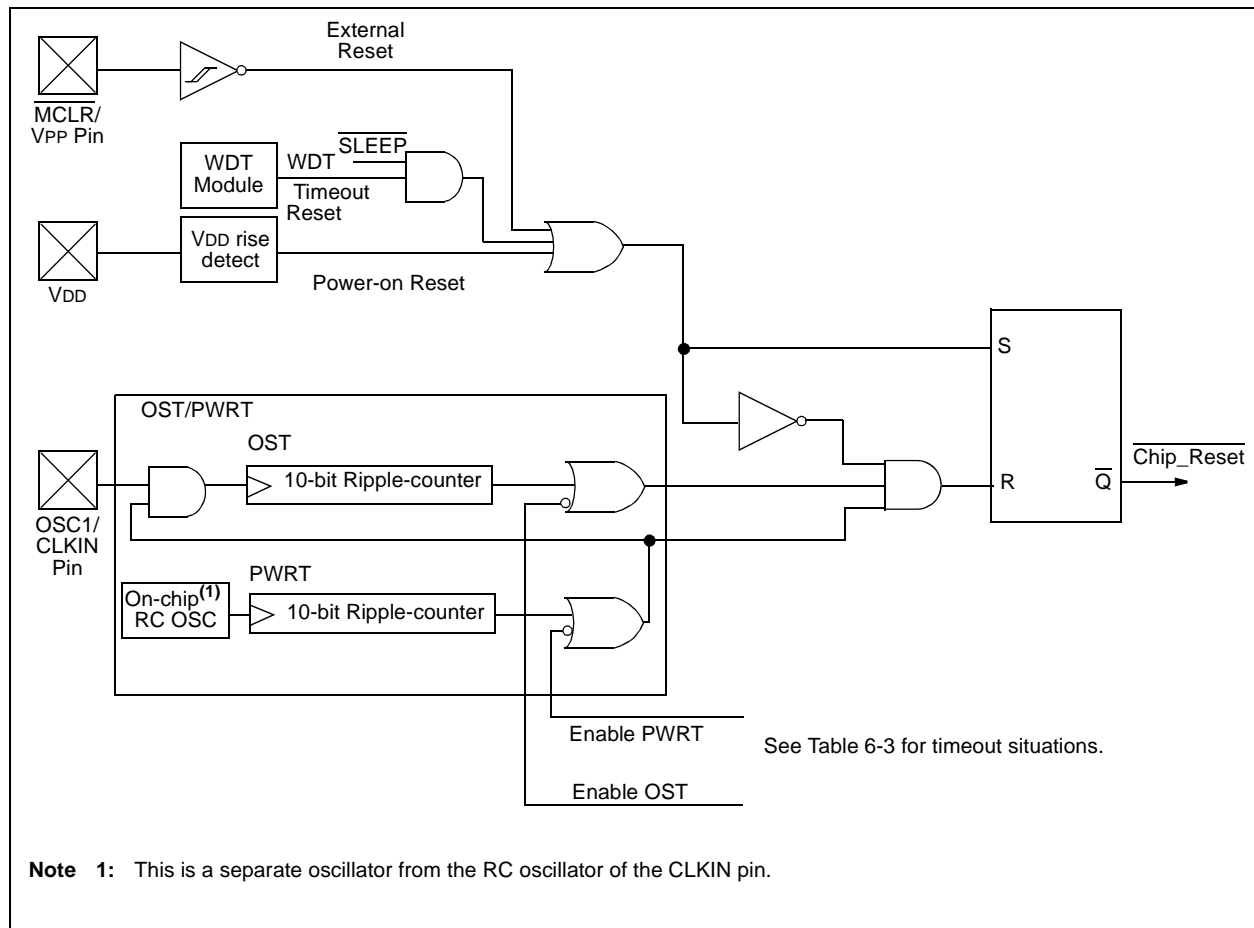
- Power-on Reset (POR)
- $\overline{\text{MCLR}}$ Reset during normal operation
- $\overline{\text{MCLR}}$ Reset during SLEEP
- WDT Reset (normal operation)
- WDT wake-up (SLEEP)

Some registers are not affected in any RESET condition; their status is unknown on POR and unchanged in any other RESET. Most other registers are reset to a "RESET state" on Power-on Reset, on $\overline{\text{MCLR}}$ or WDT Reset and on $\overline{\text{MCLR}}$ Reset during SLEEP. They are not affected by a WDT wake-up, since this is viewed as the resumption of normal operation. $\overline{\text{TO}}$ and $\overline{\text{PD}}$ bits are set or cleared differently in different RESET situations as indicated in Table 6-4. These bits are used in software to determine the nature of the RESET. See Table 6-6 for a full description of RESET states of all registers.

A simplified block diagram of the on-chip RESET circuit is shown in Figure 6-6.

The $\overline{\text{MCLR}}$ Reset path has a noise filter to detect and ignore small pulses. See Table 10-3 for pulse width specification.

FIGURE 6-6: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT



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

6.4.1 POWER-ON RESET (POR)

A Power-on Reset pulse is generated on-chip when VDD rise is detected (in the range of 1.6V – 1.8V). To take advantage of the POR, just tie the $\overline{\text{MCLR}}$ pin through a resistor to VDD. This will eliminate external RC components usually needed to create Power-on Reset. A maximum rise time for VDD is required. See Electrical Specifications for details.

The POR circuit does not produce internal RESET when VDD declines.

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

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

6.4.2 POWER-UP TIMER (PWRT)

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

6.4.3 OSCILLATOR START-UP TIMER (OST)

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

The OST timeout is invoked only for XT, LP and HS modes and only on Power-on Reset or wake-up from SLEEP.

6.4.4 TIMEOUT SEQUENCE

On power-up, the timeout sequence is as follows: First PWRT timeout is invoked after POR has expired, then OST is activated. The total timeout will vary based on oscillator configuration and $\overline{\text{PWRTE}}$ bit status. For example, in RC mode with $\overline{\text{PWRTE}}$ bit erased (PWRT disabled), there will be no timeout at all. Figure 6-7, Figure 6-8 and Figure 6-9 depict timeout sequences.

Since the timeouts occur from the POR pulse, if $\overline{\text{MCLR}}$ is kept low long enough, the timeouts will expire. Then bringing $\overline{\text{MCLR}}$ high will begin execution immediately (see Figure 6-8). This is useful for testing purposes or to synchronize more than one PIC16C55X device operating in parallel.

Table 6-5 shows the RESET conditions for some special registers, while Table 6-6 shows the RESET conditions for all the registers.

6.4.5 POWER CONTROL/STATUS REGISTER (PCON)

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

TABLE 6-3: TIMEOUT IN VARIOUS SITUATIONS

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

TABLE 6-4: STATUS BITS AND THEIR SIGNIFICANCE

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

PIC16C55X

TABLE 6-5: INITIALIZATION CONDITION FOR SPECIAL REGISTERS

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

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

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

TABLE 6-6: INITIALIZATION CONDITION FOR REGISTERS

| Register | Address | Power-on Reset | MCLR Reset during normal operation MCLR Reset during SLEEP WDT Reset | Wake-up from SLEEP through interrupt Wake-up from SLEEP through WDT timeout |
|----------------------|---------|----------------|--|--|
| W | — | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| INDF | 00h | — | — | — |
| TMR0 | 01h | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| PCL | 02h | 0000 0000 | 0000 0000 | PC + 1 ⁽²⁾ |
| STATUS | 03h | 0001 1xxx | 000q quuu ⁽³⁾ | uuuq quuu ⁽³⁾ |
| FSR | 04h | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| PORTA | 05h | ---x xxxx | ---u uuuu | ---u uuuu |
| PORTB | 06h | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| PORTC ⁽⁴⁾ | 06h | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| PCLATH | 0Ah | ---0 0000 | ---0 0000 | ---u uuuu |
| INTCON | 0Bh | 0000 000x | 0000 000u | uuuu uuuu ⁽¹⁾ |
| OPTION | 81h | 1111 1111 | 1111 1111 | uuuu uuuu |
| TRISA | 85h | ---1 1111 | ---1 1111 | ---u uuuu |
| TRISB | 86h | 1111 1111 | 1111 1111 | uuuu uuuu |
| TRISC ⁽⁴⁾ | 86h | 1111 1111 | 1111 1111 | uuuu uuuu |
| PCON | 8Eh | ---- --0- | ---- --u- | ---- --u- |

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

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

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 6-5 for RESET value for specific condition.

4: PIC16C557 only.

FIGURE 6-7: TIMEOUT SEQUENCE ON POWER-UP ($\overline{\text{MCLR}}$ NOT TIED TO V_{DD}): CASE 1

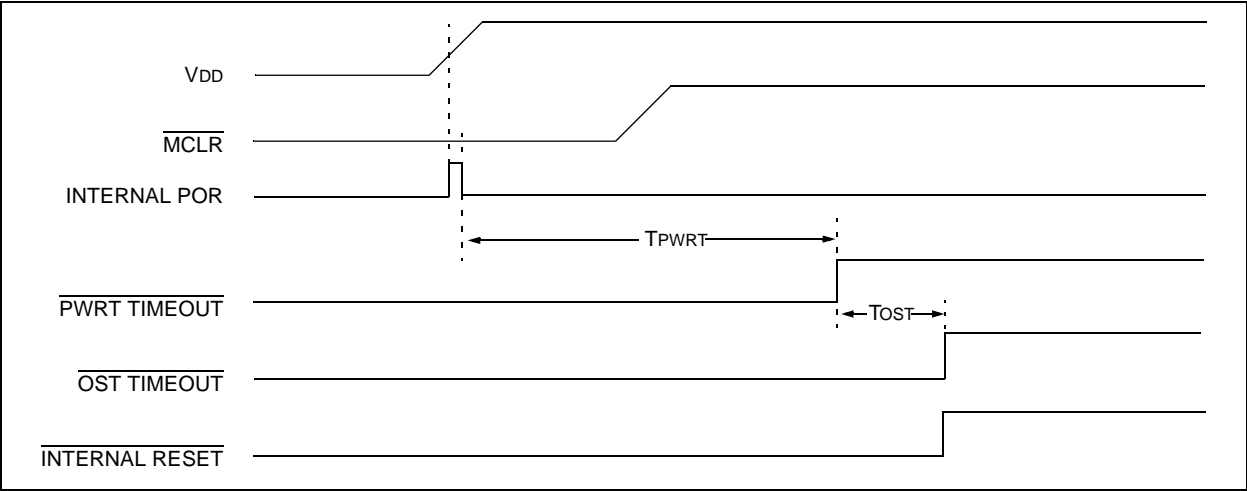
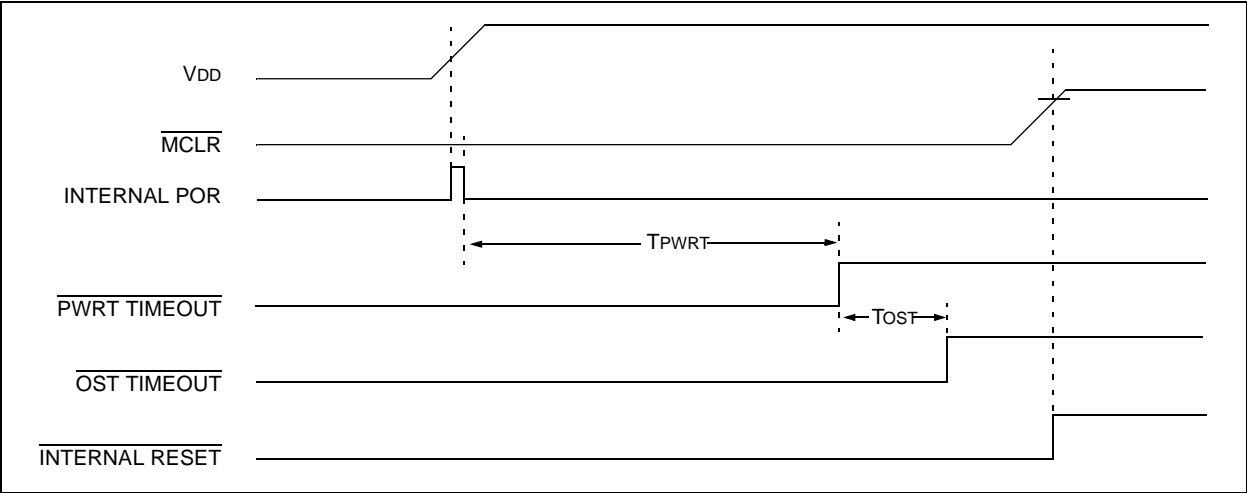


FIGURE 6-8: TIMEOUT SEQUENCE ON POWER-UP ($\overline{\text{MCLR}}$ NOT TIED TO V_{DD}): CASE 2



PIC16C55X

FIGURE 6-9: TIMEOUT SEQUENCE ON POWER-UP ($\overline{\text{MCLR}}$ TIED TO V_{DD}): CASE 3

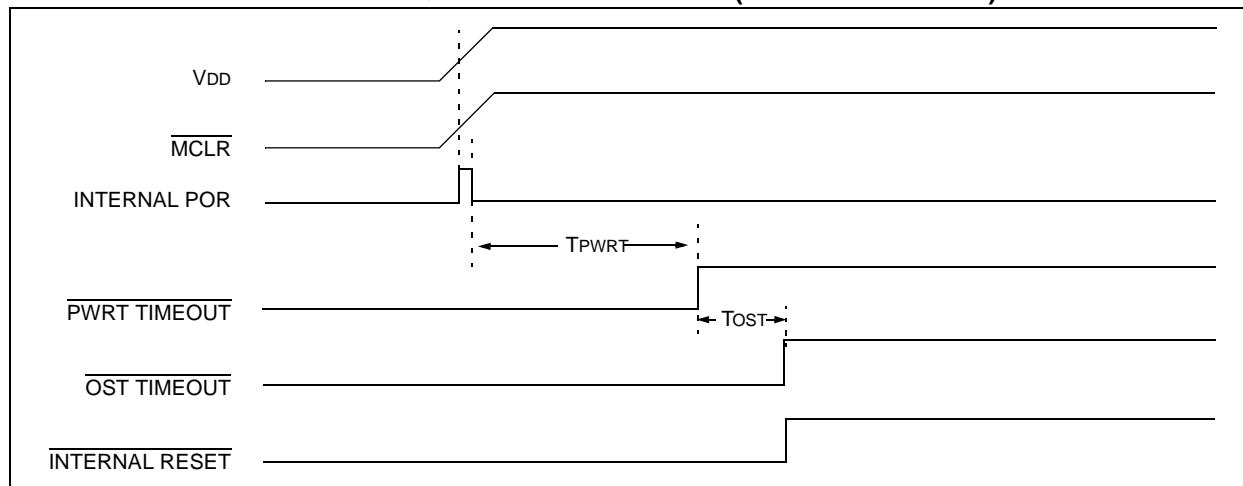
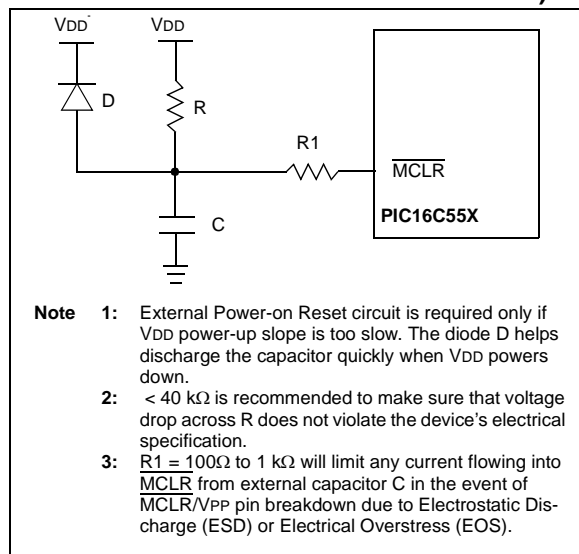


FIGURE 6-10: EXTERNAL POWER-ON RESET CIRCUIT (FOR SLOW V_{DD} POWER-UP)



6.5 Interrupts

The PIC16C55X has 3 sources of interrupt:

- External interrupt RB0/INT
- TMR0 overflow interrupt
- PORTB change interrupts (pins RB7:RB4)

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

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

The “Return from Interrupt” instruction, *RETFIE*, exits the interrupt routine as well as sets the GIE bit, which re-enables RB0/INT interrupts.

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

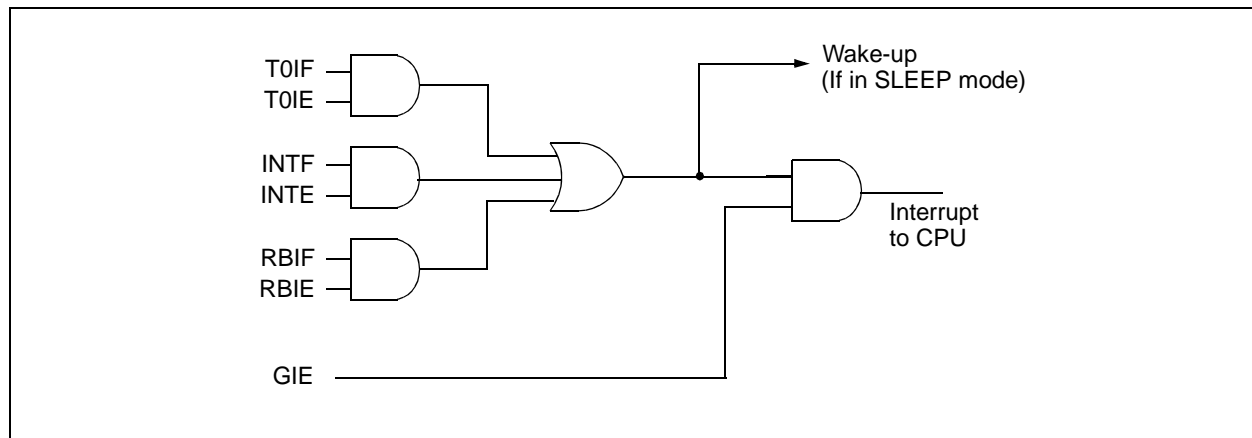
When an interrupt is responded to, the GIE is cleared to disable any further interrupt, the return address is pushed into the stack and the PC is loaded with 0004h. Once in the interrupt service routine the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt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

For external interrupt events, such as the INT pin or PORTB change interrupt, the interrupt latency will be three or four instruction cycles. The exact latency depends when the interrupt event occurs (Figure 6-12). The latency is the same for one or two cycle instructions. Once in the interrupt service routine, the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt flag bit(s) must be cleared in software before re-enabling interrupts to avoid multiple interrupt requests. Individual interrupt flag bits are set regardless of the status of their corresponding mask bit or the GIE bit.

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

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

FIGURE 6-11: INTERRUPT LOGIC



PIC16C55X

6.5.1 RB0/INT INTERRUPT

An external interrupt on RB0/INT pin is edge triggered: either rising if INTEDG bit (OPTION<6>) is set, or falling if INTEDG bit is clear. When a valid edge appears on the RB0/INT pin, the INTF bit (INTCON<1>) is set. This interrupt can be disabled by clearing the INTE control bit (INTCON<4>). The INTF bit must be cleared in software in the interrupt service routine before re-enabling this interrupt. The RB0/INT interrupt can wake-up the processor from SLEEP, if the INTE bit was set prior to going into SLEEP. The status of the GIE bit decides whether or not the processor branches to the interrupt vector following wake-up. See Section 6.8 for details on SLEEP and Figure 6-14 for timing of wake-up from SLEEP through RB0/INT interrupt.

6.5.2 TMR0 INTERRUPT

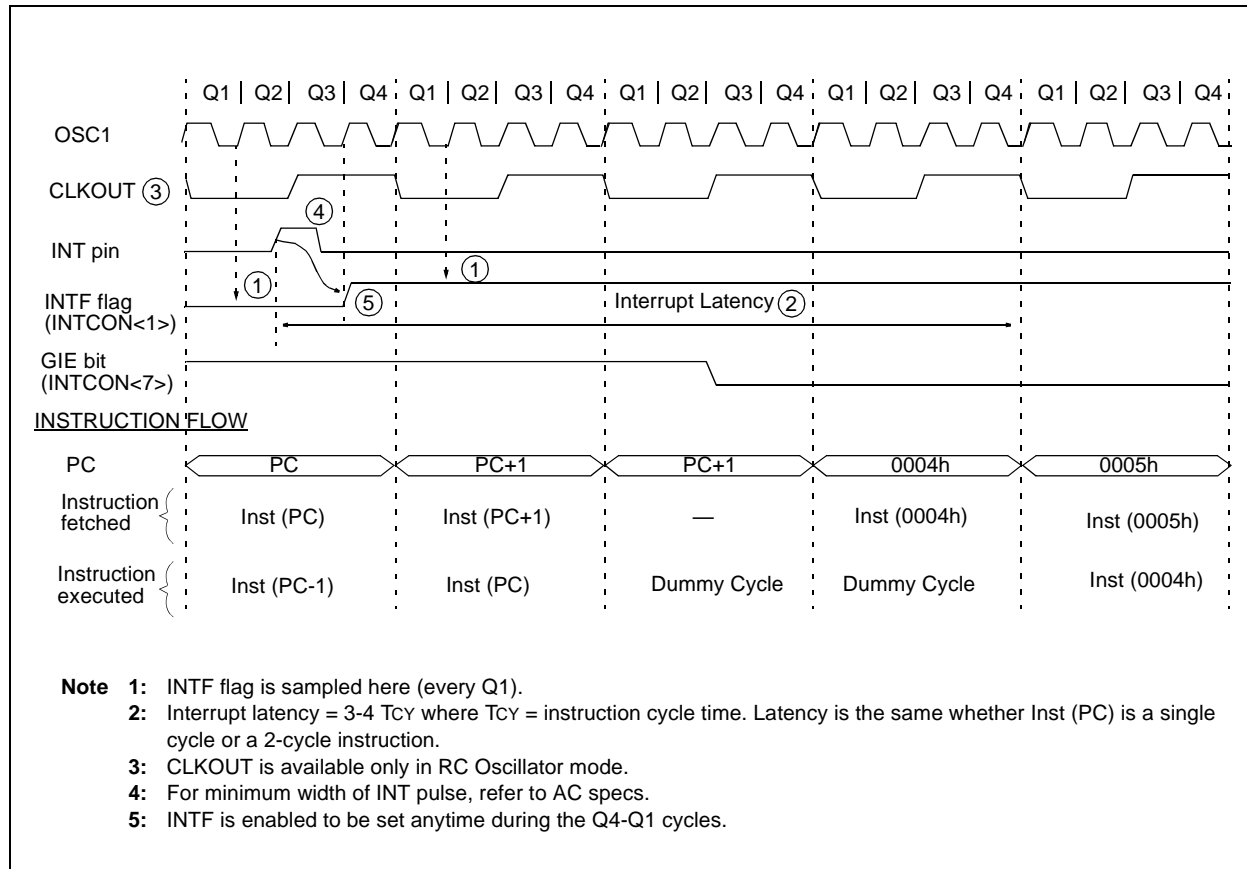
An overflow (FFh → 00h) in the TMR0 register will set the T0IF (INTCON<2>) bit. The interrupt can be enabled/disabled by setting/clearing T0IE (INTCON<5>) bit. For operation of the Timer0 module, see Section 7.0.

6.5.3 PORTB INTERRUPT

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

Note: If a change on the I/O pin should occur when the read operation is being executed (start of the Q2 cycle), then the RBIF interrupt flag may get set.

FIGURE 6-12: INT PIN INTERRUPT TIMING



6.6 Context Saving During Interrupts

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

Example 6-1 stores and restores the STATUS and W registers. The user register, W_TEMP, must be defined in both banks and must be defined at the same offset from the bank base address (i.e., W_TEMP is defined at 0x20 in Bank 0 and it must also be defined at 0xA0 in Bank 1). The user register, STATUS_TEMP, must be defined in Bank 0. The Example 6-1:

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

EXAMPLE 6-1: SAVING THE STATUS AND W REGISTERS IN RAM

```

MOVWF  W_TEMP      ;copy W to TEMP
                    ;register, could be in
                    ;either bank
SWAPF  STATUS,W     ;swap STATUS to be
                    ;saved into W
BCF     STATUS,RP0   ;change to bank0
                    ;regardless of
                    ;current bank
MOVWF  STATUS_TEMP  ;save STATUS to bank0
                    ;register
:
:
:
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

```

6.7 Watchdog Timer (WDT)

The Watchdog Timer is a free running on-chip RC oscillator which does not require any external components. This RC oscillator is separate from the RC oscillator of the CLKIN pin. That means that the WDT will run, even if the clock on the OSC1 and OSC2 pins of the device has been stopped, for example, by execution of a SLEEP instruction. During normal operation, a WDT timeout generates a device RESET. If the device is in SLEEP mode, a WDT timeout causes the device to wake-up and continue with normal operation. The WDT can be permanently disabled by programming the configuration bit WDTE as clear (Section 6.1).

6.7.1 WDT PERIOD

The WDT has a nominal timeout period of 18 ms, (with no prescaler). The timeout periods vary with temperature, VDD and process variations from part-to-part (see DC specs). If longer timeout 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, timeout periods up to 2.3 seconds can be realized.

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

The \overline{TO} bit in the STATUS register will be cleared upon a Watchdog Timer timeout.

6.7.2 WDT PROGRAMMING CONSIDERATIONS

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

PIC16C55X

FIGURE 6-13: WATCHDOG TIMER BLOCK DIAGRAM

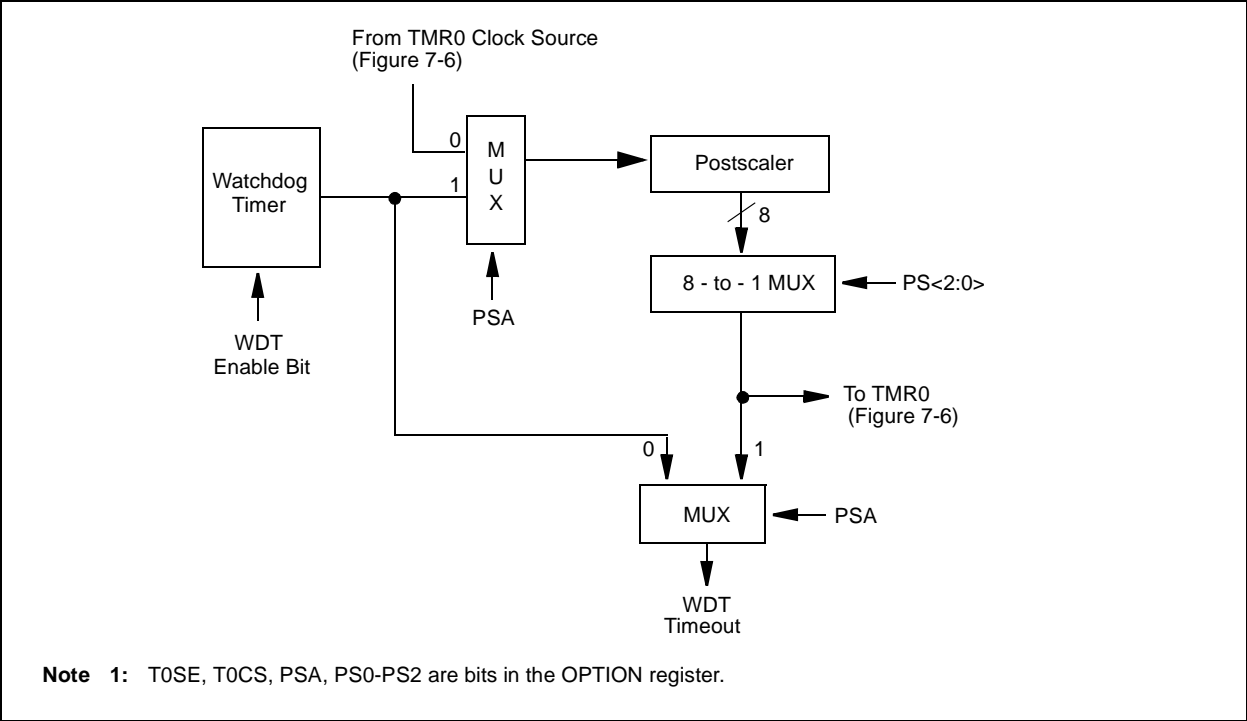


TABLE 6-7: SUMMARY OF WATCHDOG TIMER REGISTERS

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR | Value on all other RESETS |
|---------|--------------|-------|----------|-------|-------|-------|-------|-------|-------|--------------|---------------------------|
| 2007h | Config. bits | — | Reserved | CP1 | CP0 | PWRTE | WDTE | FOSC1 | FOSC0 | | |
| 81h | OPTION | RBP1 | INTEDG | T0CS | T0SE | PSA | PS2 | PS1 | PS0 | 1111 1111 | 1111 1111 |

Legend: x = unknown, u = unchanged, q = value depends on condition, — = unimplemented, read as '0'.
Shaded cells are not used by the Watchdog Timer.

6.8 Power-Down Mode (SLEEP)

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

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

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

The MCLR pin must be at a logic high level (VIHMC).

Note: It should be noted that a RESET generated by a WDT timeout does not drive MCLR pin low.

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

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

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

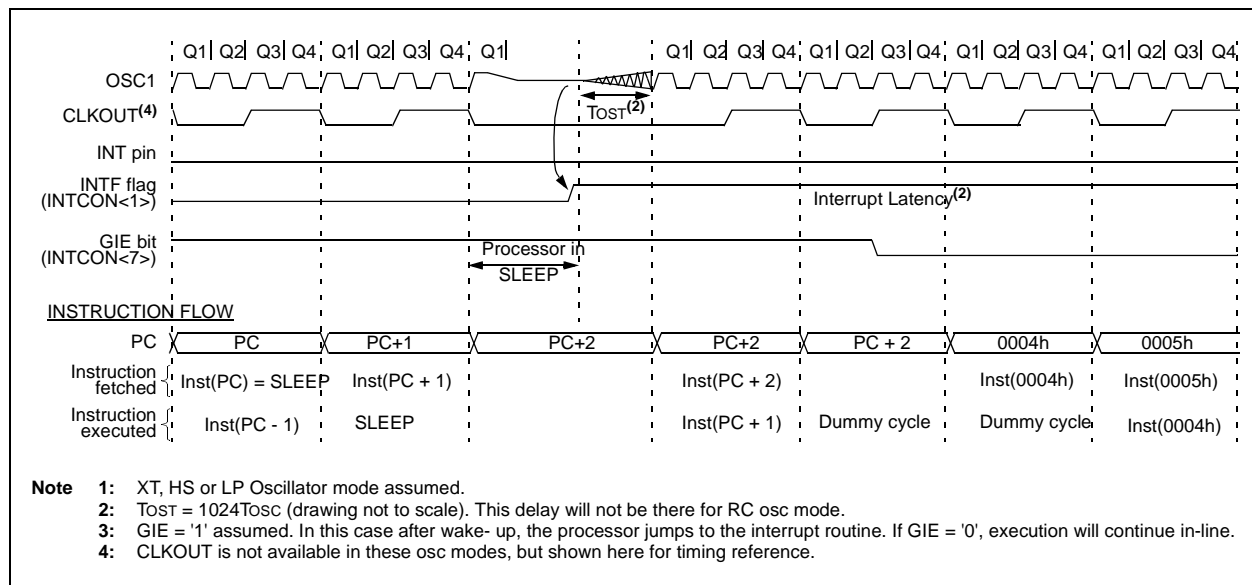
6.8.1 WAKE-UP FROM SLEEP

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

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

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

FIGURE 6-14: WAKE-UP FROM SLEEP THROUGH INTERRUPT



PIC16C55X

6.9 Code Protection

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

Note: Microchip does not recommend code protecting windowed devices.

6.10 ID Locations

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

6.11 In-Circuit Serial Programming™

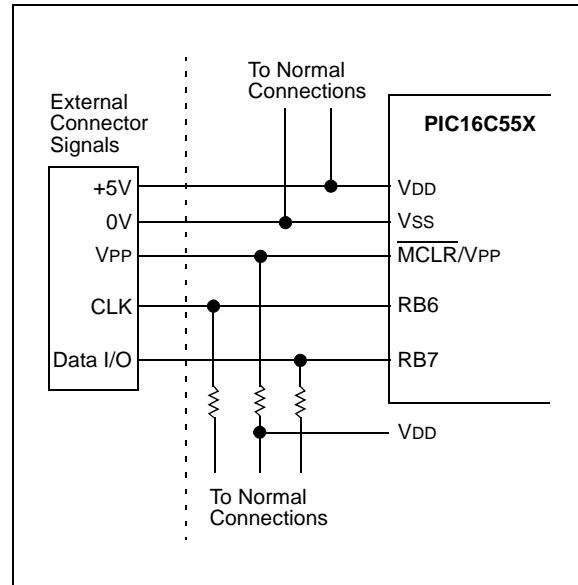
The PIC16C55X microcontrollers can be serially programmed while in the end application circuit. This is simply done with two lines for clock and data, and three other lines for power, ground, and the programming voltage. This allows customers to manufacture boards with unprogrammed devices, and then program the microcontroller just before shipping the product. This also allows the most recent firmware or a custom firmware to be programmed.

The device is placed into a Program/Verify mode by holding the RB6 and RB7 pins low while raising the MCLR (VPP) pin from VIL to VIH (see programming specification). RB6 becomes the programming clock and RB7 becomes the programming data. Both RB6 and RB7 are Schmitt Trigger inputs in this mode.

After RESET, to place the device into Programming/Verify mode, the program counter (PC) is at location 00h. A 6-bit command is then supplied to the device. Depending on the command, 14 bits of program data are then supplied to or from the device, depending if the command was a load or a read. For complete details of serial programming, please refer to the PIC16C6X/7X Programming Specifications (Literature #DS30228).

A typical in-circuit serial programming connection is shown in Figure 6-15.

FIGURE 6-15: TYPICAL IN-CIRCUIT SERIAL PROGRAMMING CONNECTION



7.0 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 7-1 is a simplified block diagram of the Timer0 module.

Timer mode is selected by clearing the T0CS bit (OPTION<5>). In Timer mode, the TMR0 will increment every instruction cycle (without prescaler). If Timer0 is written, the increment is inhibited for the following two cycles (Figure 7-2 and Figure 7-3). The user can work around this by writing an adjusted value to TMR0.

Counter mode is selected by setting the T0CS bit. In this mode Timer0 will increment either on every rising or falling edge of pin RA4/T0CKI. The incrementing edge is determined by the source edge (T0SE) control

bit (OPTION<4>). Clearing the T0SE bit selects the rising edge. Restrictions on the external clock input are discussed in detail in Section 7.2.

The prescaler is shared between the Timer0 module and the Watchdog Timer. The prescaler assignment is controlled in software by the control bit PSA (OPTION<3>). Clearing the PSA bit will assign the prescaler to Timer0. The prescaler is not readable or writable. When the prescaler is assigned to the Timer0 module, prescale value of 1:2, 1:4, ..., 1:256 are selectable. Section 7.3 details the operation of the prescaler.

7.1 TIMER0 Interrupt

Timer0 interrupt is generated when the TMR0 register timer/counter overflows from FFh to 00h. This overflow sets the T0IF bit. The interrupt can be masked by clearing the T0IE bit (INTCON<5>). The T0IF bit (INTCON<2>) must be cleared in software by the Timer0 module interrupt service routine before re-enabling this interrupt. The Timer0 interrupt cannot wake the processor from SLEEP since the timer is shut off during SLEEP. See Figure 7-4 for Timer0 interrupt timing.

FIGURE 7-1: TIMER0 BLOCK DIAGRAM

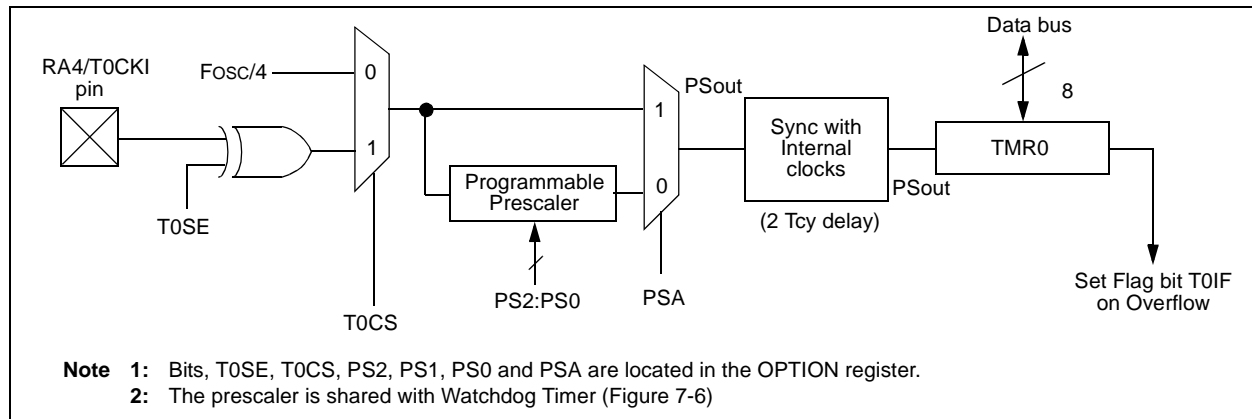
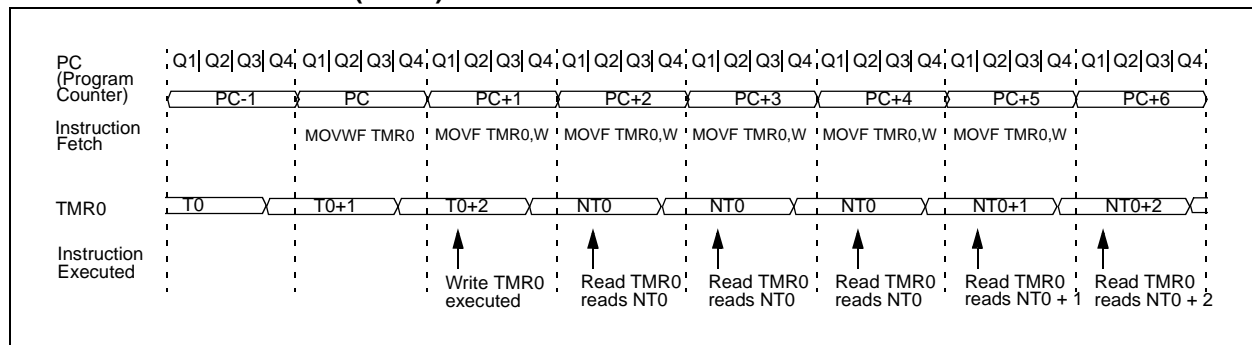


FIGURE 7-2: TIMER0 (TMR0) TIMING: INTERNAL CLOCK/NO PRESCALER



PIC16C55X

FIGURE 7-3: TIMER0 TIMING: INTERNAL CLOCK/PRESCALE 1:2

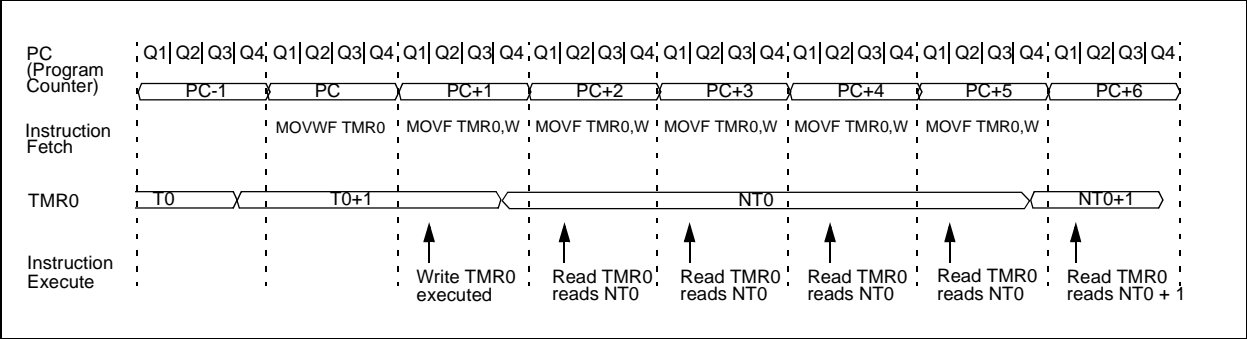
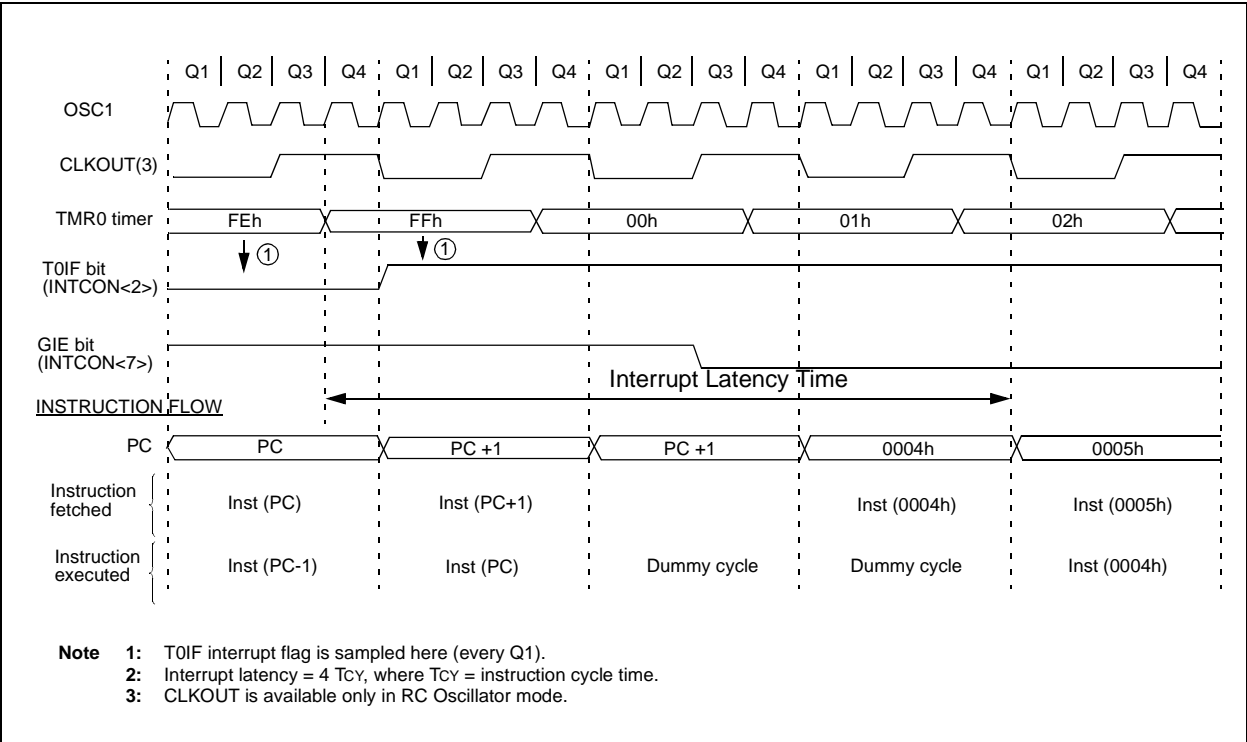


FIGURE 7-4: TIMER0 INTERRUPT TIMING



7.2 Using Timer0 with External Clock

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

7.2.1 EXTERNAL CLOCK SYNCHRONIZATION

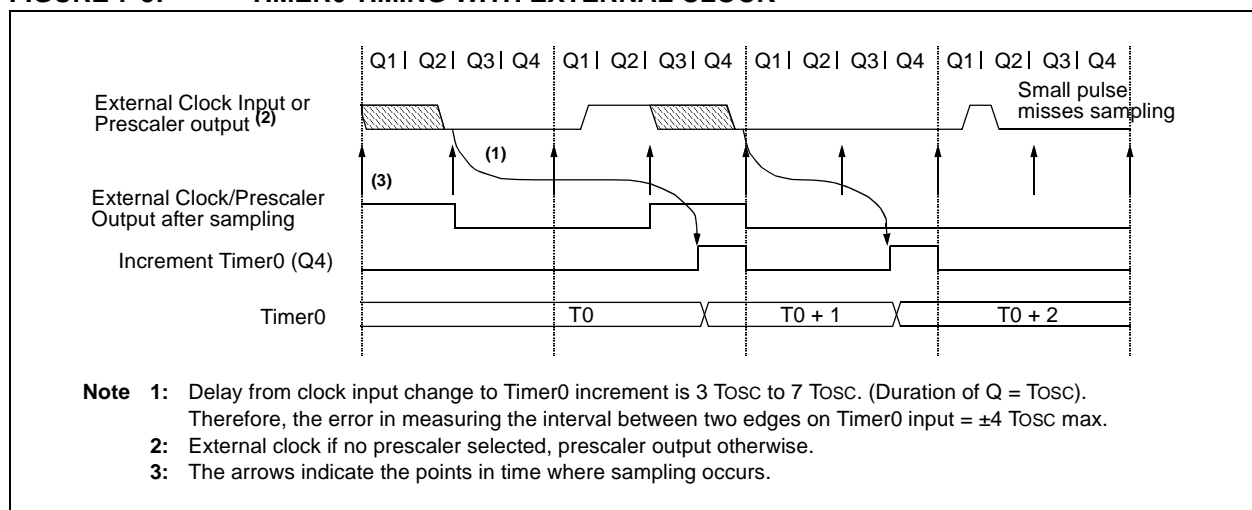
When no prescaler is used, the external clock input is the same as the prescaler output. The synchronization of T0CKI with the internal phase clocks is accomplished by sampling the prescaler output on the Q2 and Q4 cycles of the internal phase clocks (Figure 7-5). Therefore, it is necessary for T0CKI to be high for at least 2Tosc (and a small RC delay of 20 ns) and low for at least 2Tosc (and a small RC delay of 20 ns). Refer to the electrical specification of the desired device.

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

7.2.2 TIMER0 INCREMENT DELAY

Since the prescaler output is synchronized with the internal clocks, there is a small delay from the time the external clock edge occurs to the time the TMR0 is actually incremented. Figure 7-5 shows the delay from the external clock edge to the timer incrementing.

FIGURE 7-5: TIMER0 TIMING WITH EXTERNAL CLOCK



7.3 Prescaler

An 8-bit counter is available as a prescaler for the Timer0 module, or as a postscaler for the Watchdog Timer, respectively (Figure 7-6). For simplicity, this counter is being referred to as “prescaler” throughout this data sheet.

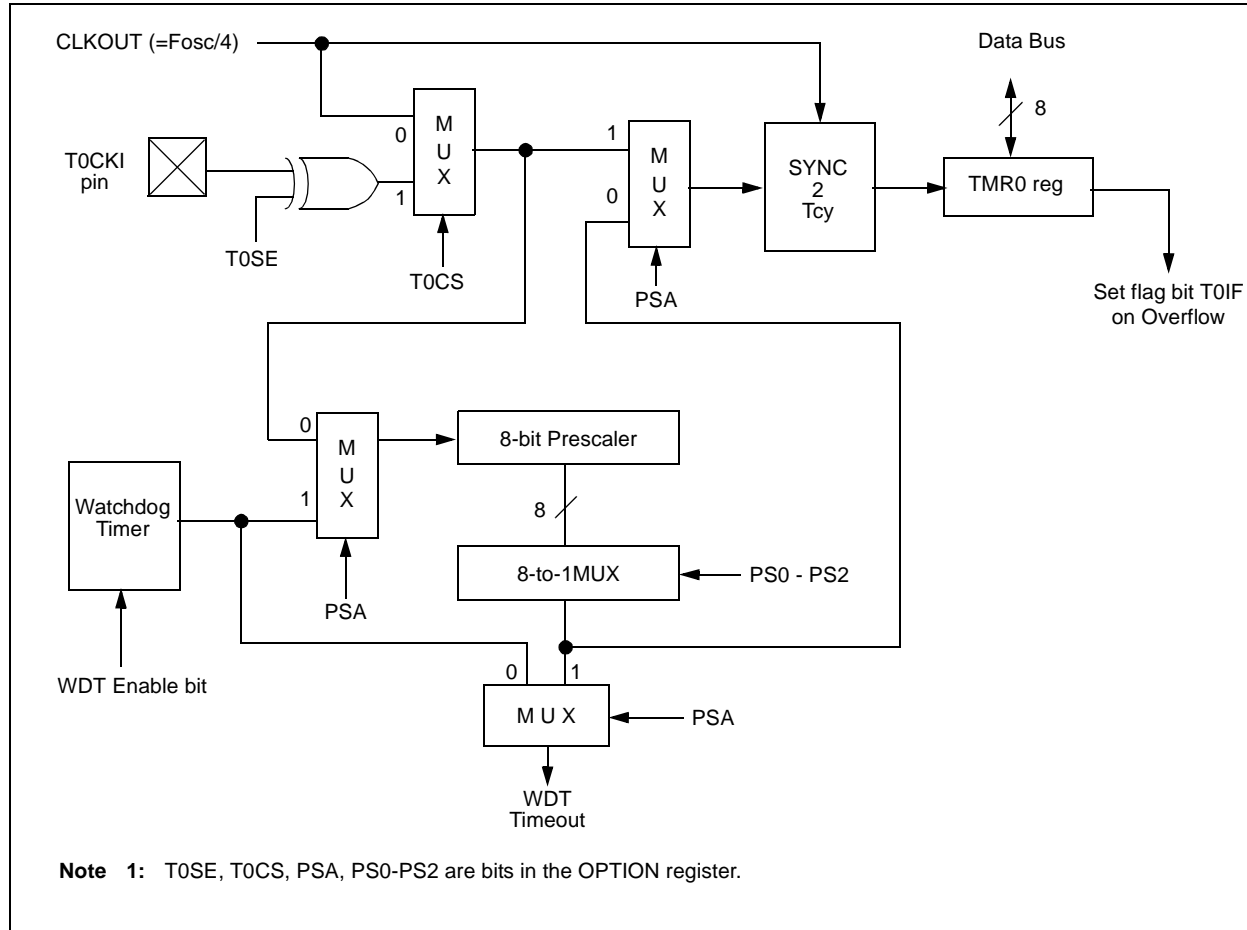
Note: There is only one prescaler available which is mutually exclusive between the Timer0 module and the Watchdog Timer. Thus, a prescaler assignment for the Timer0 module means that there is no prescaler for the Watchdog Timer, and vice-versa.

The PSA and 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 CLRWDI instruction will clear the prescaler along with the Watchdog Timer. The prescaler is not readable or writable.

PIC16C55X

FIGURE 7-6: BLOCK DIAGRAM OF THE TIMER0/WDT PRESCALER



7.3.1 SWITCHING PRESCALER ASSIGNMENT

The prescaler assignment is fully under software control (i.e., it can be changed “on the fly” during program execution). To avoid an unintended device RESET, the following instruction sequence (Example 7-1) must be executed when changing the prescaler assignment from Timer0 to WDT. Lines 5-7 are required only if the desired postscaler rate is 1:1 (PS<2:0> = 000) or 1:2 (PS<2:0> = 001).

EXAMPLE 7-1: CHANGING PRESCALER (TIMER0→WDT)

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

To change prescaler from the WDT to the TMR0 module use the sequence shown in Example 7-2. This precaution must be taken even if the WDT is disabled.

EXAMPLE 7-2: CHANGING PRESCALER (WDT→TIMER0)

```
CLRWDTClear WDT and
        ;prescaler
BSF     STATUS, RP0
MOVLW   b'xxx0xxx' ;Select TMR0, new
                        ;prescale value and
                        ;clock source
MOVWF   OPTION
BCF     STATUS, RP0
```

TABLE 7-1: 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 | Reserved | TOIE | INTE | RBIE | TOIF | INTF | RBIF | 0000 000x | 0000 000x |
| 81h | OPTION | RBPU | INTEDG | T0CS | T0SE | PSA | PS2 | PS1 | PS0 | 1111 1111 | 1111 1111 |
| 85h | TRISA | — | — | — | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | ---1 1111 | ---1 1111 |

Legend: — = Unimplemented locations, read as '0',
Note 1: Shaded bits are not used by TMR0 module.

PIC16C55X

NOTES:

8.0 INSTRUCTION SET SUMMARY

Each PIC16C55X 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 PIC16C55X instruction set summary in Table 8-2 lists **byte-oriented**, **bit-oriented**, and **literal and control** operations. Table 8-1 shows the opcode field descriptions.

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

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

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

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

TABLE 8-1: OPCODE FIELD DESCRIPTIONS

| Field | Description |
|----------------|---|
| f | Register file address (0x00 to 0x7F) |
| W | Working register (accumulator) |
| b | Bit address within an 8-bit file register |
| k | Literal field, constant data or label |
| x | Don't care location (= 0 or 1) The assembler will generate code with x = 0. It is the recommended form of use for compatibility with all Microchip software tools. |
| d | Destination select; d = 0: store result in W, d = 1: store result in file register f. Default is d = 1 |
| label | Label name |
| TOS | Top of Stack |
| PC | Program Counter |
| PCLATH | Program Counter High Latch |
| GIE | Global Interrupt Enable bit |
| WDT | Watchdog Timer/Counter |
| TO | Timeout bit |
| PD | 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 μs. If a conditional test is true or the program counter is changed as a result of an instruction, the instruction execution time is 2 μs.

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

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

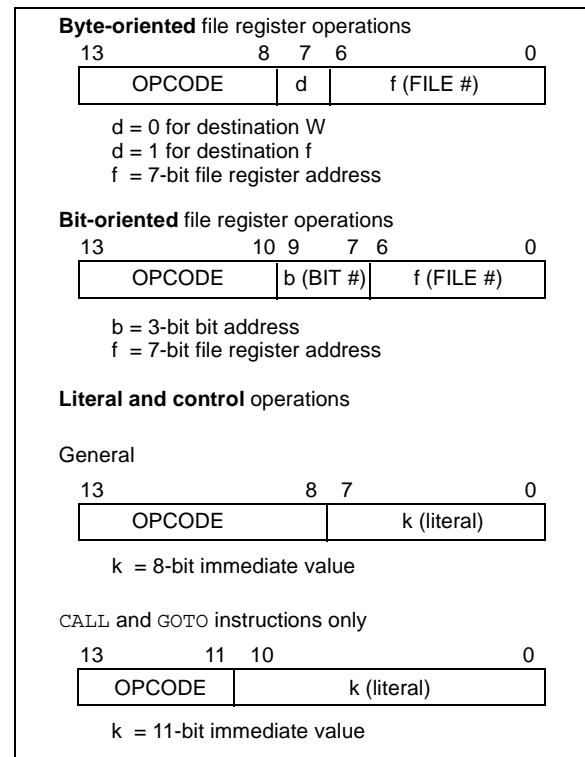
Note: To maintain upward compatibility with future PIC® MCU products, do not use the `OPTION` and `TRIS` instructions.

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

0xhh

where h signifies a hexadecimal digit.

FIGURE 8-1: GENERAL FORMAT FOR INSTRUCTIONS



PIC16C55X

TABLE 8-2: PIC16C55X INSTRUCTION SET

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

- Note 1:** When an I/O register is modified as a function of itself (e.g., `MOVF PORTB, 1`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.
- Note 2:** If this instruction is executed on the TMR0 register (and, where applicable, d = 1), the prescaler will be cleared if assigned to the Timer0 Module.
- Note 3:** If Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a NOP.

8.1 Instruction Descriptions

ADDLW Add Literal and W

| | | | | | |
|------------------|---|------|------|------|------|
| Syntax: | [<i>label</i>] ADDLW k | | | | |
| Operands: | $0 \leq k \leq 255$ | | | | |
| Operation: | $(W) + k \rightarrow (W)$ | | | | |
| 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 | ADDLW 0x15 Before Instruction W = 0x10 After Instruction W = 0x25 | | | | |

ANDLW AND Literal with W

| | | | | | |
|------------------|---|------|------|------|------|
| Syntax: | [<i>label</i>] ANDLW k | | | | |
| Operands: | $0 \leq k \leq 255$ | | | | |
| Operation: | (W) .AND. (k) \rightarrow (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 | ANDLW 0x5F Before Instruction W = 0xA3 After Instruction W = 0x03 | | | | |

ADDWF Add W and f

| | | | | | | | | | | | | | |
|------------------|--|------|------|------|------|---|------|---|---|------|-----|---|------|
| Syntax: | [<i>label</i>] ADDWF <i>f</i> , <i>d</i> | | | | | | | | | | | | |
| Operands: | $0 \leq f \leq 127$ $d \in [0,1]$ | | | | | | | | | | | | |
| Operation: | $(W) + (f) \rightarrow (dest)$ | | | | | | | | | | | | |
| 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</p> <table><tr><td>W</td><td>=</td><td>0x17</td></tr><tr><td>FSR</td><td>=</td><td>0xC2</td></tr></table> <p>After Instruction</p> <table><tr><td>W</td><td>=</td><td>0xD9</td></tr><tr><td>FSR</td><td>=</td><td>0xC2</td></tr></table> | W | = | 0x17 | FSR | = | 0xC2 | W | = | 0xD9 | FSR | = | 0xC2 |
| W | = | 0x17 | | | | | | | | | | | |
| FSR | = | 0xC2 | | | | | | | | | | | |
| W | = | 0xD9 | | | | | | | | | | | |
| FSR | = | 0xC2 | | | | | | | | | | | |

ANDWF AND W with f

| | | | | | |
|------------------|--|------|------|------|------|
| Syntax: | [<i>label</i>] ANDWF <i>f</i> , <i>d</i> | | | | |
| Operands: | $0 \leq f \leq 127$ $d \in [0,1]$ | | | | |
| Operation: | (W) .AND. (f) \rightarrow (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 | ANDWF FSR, 1 Before Instruction W = 0x17 FSR = 0xC2 After Instruction W = 0x17 FSR = 0x02 | | | | |

PIC16C55X

BCF Bit Clear f

| | | | | | |
|------------------|---|------|------|------|------|
| Syntax: | [<i>label</i>] BCF f,b | | | | |
| Operands: | $0 \leq f \leq 127$ $0 \leq b \leq 7$ | | | | |
| Operation: | $0 \rightarrow (f)$ | | | | |
| Status Affected: | None | | | | |
| Encoding: | <table border="1"><tr><td>01</td><td>00bb</td><td>bfff</td><td>ffff</td></tr></table> | 01 | 00bb | bfff | ffff |
| 01 | 00bb | bfff | ffff | | |
| Description: | Bit 'b' in register 'f' is cleared. | | | | |
| Words: | 1 | | | | |
| Cycles: | 1 | | | | |
| Example | BCF FLAG_REG, 7 | | | | |

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

BSF Bit Set f

| | | | | |
|------------------|--|-----------|------|------|
| Syntax: | [<i>label</i>] BSF f,b | | | |
| Operands: | $0 \leq f \leq 127$ $0 \leq b \leq 7$ | | | |
| Operation: | $1 \rightarrow (f)$ | | | |
| Status Affected: | None | | | |
| Encoding: | 01 | 01bb | bfff | ffff |
| Description: | Bit 'b' in register 'f' is set. | | | |
| Words: | 1 | | | |
| Cycles: | 1 | | | |
| Example | BSF | FLAG_REG, | 7 | |

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

BTFSC Bit Test, Skip if Clear

| | | | | | |
|------------------|--|------|------|------|------|
| Syntax: | [<i>label</i>] BTFSC f,b | | | | |
| Operands: | $0 \leq f \leq 127$ $0 \leq b \leq 7$ | | | | |
| Operation: | skip if (f) = 0 | | | | |
| Status Affected: | None | | | | |
| Encoding: | <table><tr><td>01</td><td>10bb</td><td>bfff</td><td>ffff</td></tr></table> | 01 | 10bb | bfff | ffff |
| 01 | 10bb | bfff | ffff | | |
| Description: | If bit 'b' in register 'f' is '0' then the next instruction is skipped. If bit 'b' is '1' then the next instruction fetched during the current instruction execution is discarded, and a NOP is executed instead, making this a two-cycle instruction. | | | | |

Words: 1
Cycles: 1(2)
Example

| | | |
|-------|-------|--------------|
| HERE | 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

| BTFSS | | Bit Test f, Skip if Set | | | | | | | | | | | | | | | | | | |
|------------------|---|-------------------------|------|--|----|------|-------|--------|-------|------|--------------|------|---|--|--|---|--|--|---|--|
| Syntax: | [<i>label</i>] BTFSS f,b | | | | | | | | | | | | | | | | | | | |
| Operands: | $0 \leq f \leq 127$ $0 \leq b < 7$ | | | | | | | | | | | | | | | | | | | |
| Operation: | skip if (f) = 1 | | | | | | | | | | | | | | | | | | | |
| Status Affected: | None | | | | | | | | | | | | | | | | | | | |
| Encoding: | <table border="1"><tr><td>01</td><td>11bb</td><td>bfff</td><td>ffff</td></tr></table> | | | | 01 | 11bb | bfff | ffff | | | | | | | | | | | | |
| 01 | 11bb | bfff | ffff | | | | | | | | | | | | | | | | | |
| Description: | If bit 'b' in register 'f' is '1' then the next instruction is skipped. If bit 'b' is '1', then the next instruction fetched during the current instruction execution, is discarded and a NOP is executed instead, making this a two-cycle instruction. | | | | | | | | | | | | | | | | | | | |
| Words: | 1 | | | | | | | | | | | | | | | | | | | |
| Cycles: | 1(2) | | | | | | | | | | | | | | | | | | | |
| Example | <table><tr><td>HERE</td><td>BTFSS</td><td>FLAG,1</td></tr><tr><td>FALSE</td><td>GOTO</td><td>PROCESS_CODE</td></tr><tr><td>TRUE</td><td>.</td><td></td></tr><tr><td></td><td>.</td><td></td></tr><tr><td></td><td>.</td><td></td></tr></table> Before Instruction PC = address HERE After Instruction if FLAG<1> = 0, PC = address FALSE if FLAG<1> = 1, PC = address TRUE | | | | | HERE | BTFSS | FLAG,1 | FALSE | GOTO | PROCESS_CODE | TRUE | . | | | . | | | . | |
| HERE | BTFSS | FLAG,1 | | | | | | | | | | | | | | | | | | |
| FALSE | GOTO | PROCESS_CODE | | | | | | | | | | | | | | | | | | |
| TRUE | . | | | | | | | | | | | | | | | | | | | |
| | . | | | | | | | | | | | | | | | | | | | |
| | . | | | | | | | | | | | | | | | | | | | |

| CALL | | Call Subroutine | | | | | | | |
|------------------|---|-----------------|------|--|--|----|------|------|------|
| Syntax: | [<i>label</i>] CALL k | | | | | | | | |
| Operands: | 0 ≤ k ≤ 2047 | | | | | | | | |
| Operation: | (PC)+ 1 → TOS, k → PC<10:0>, (PCLATH<4:3>) → PC<12:11> | | | | | | | | |
| Status Affected: | None | | | | | | | | |
| Encoding: | <table border="1"><tr><td>10</td><td>0kkk</td><td>kkkk</td><td>kkkk</td></tr></table> | | | | | 10 | 0kkk | kkkk | kkkk |
| 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: | [<i>label</i>] CLRF f | | | | | | |
| Operands: | $0 \leq f \leq 127$ | | | | | | |
| Operation: | 00h \rightarrow (f) 1 \rightarrow Z | | | | | | |
| Status Affected: | Z | | | | | | |
| Encoding: | <table border="1"><tr><td>00</td><td>0001</td><td>1fff</td><td>ffff</td></tr></table> | | | 00 | 0001 | 1fff | ffff |
| 00 | 0001 | 1fff | ffff | | | | |
| Description: | The contents of register 'f' are cleared and the Z bit is set. | | | | | | |
| Words: | 1 | | | | | | |
| Cycles: | 1 | | | | | | |
| Example | <pre>CLRF FLAG_REG</pre> <p>Before Instruction FLAG_REG=0x5A</p> <p>After Instruction FLAG_REG=0x00 Z =1</p> | | | | | | |

PIC16C55X

| CLRW | Clear W | | | | |
|------------------|--|------|------|------|------|
| Syntax: | [<i>label</i>] CLRW | | | | |
| Operands: | None | | | | |
| Operation: | 00h → (W) 1 → Z | | | | |
| Status Affected: | Z | | | | |
| Encoding: | <table><tr><td>00</td><td>0001</td><td>0000</td><td>0011</td></tr></table> | 00 | 0001 | 0000 | 0011 |
| 00 | 0001 | 0000 | 0011 | | |
| Description: | W register is cleared. Zero bit (Z) is set. | | | | |
| Words: | 1 | | | | |
| Cycles: | 1 | | | | |
| Example | CLRW Before Instruction W = 0x5A After Instruction W = 0x00 Z = 1 | | | | |

| COMF | Complement f | | | | |
|------------------|--|------|------|------|------|
| Syntax: | [<i>label</i>] COMF f,d | | | | |
| Operands: | $0 \leq f \leq 127$ $d \in [0,1]$ | | | | |
| Operation: | $(\bar{f}) \rightarrow (\text{dest})$ | | | | |
| Status Affected: | Z | | | | |
| Encoding: | <table><tr><td>00</td><td>1001</td><td>dfff</td><td>ffff</td></tr></table> | 00 | 1001 | dfff | ffff |
| 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 | | | | |

| CLRWDWT | Clear Watchdog Timer | | | | |
|------------------|---|------|------|------|------|
| Syntax: | [<i>label</i>] CLRWDWT | | | | |
| Operands: | None | | | | |
| Operation: | 00h → WDT 0 → WDT prescaler, 1 → \overline{TO} 1 → \overline{PD} | | | | |
| Status Affected: | \overline{TO} , \overline{PD} | | | | |
| Encoding: | <table><tr><td>00</td><td>0000</td><td>0110</td><td>0100</td></tr></table> | 00 | 0000 | 0110 | 0100 |
| 00 | 0000 | 0110 | 0100 | | |
| Description: | CLRWDWT 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 | <div>CLRWDWT</div> <div>Before Instruction</div> <div>WDT counter = ?</div> <div>After Instruction</div> <div>WDT counter = 0x00</div> <div>WDT prescaler = 0</div> <div>\overline{TO} = 1</div> <div>\overline{PD} = 1</div> | | | | |

| DECF | Decrement f | | | | |
|------------------|--|------|------|------|------|
| Syntax: | [<i>label</i>] DECF f,d | | | | |
| Operands: | $0 \leq f \leq 127$ $d \in [0,1]$ | | | | |
| Operation: | $(f) - 1 \rightarrow (\text{dest})$ | | | | |
| Status Affected: | Z | | | | |
| Encoding: | <table><tr><td>00</td><td>0011</td><td>dfff</td><td>ffff</td></tr></table> | 00 | 0011 | dfff | ffff |
| 00 | 0011 | dfff | ffff | | |
| Description: | Decrement register 'f'. If 'd' is 0 the result is stored in the W register. If 'd' is 1 the result is stored back in register 'f'. | | | | |
| Words: | 1 | | | | |
| Cycles: | 1 | | | | |
| Example | DECF CNT, 1 Before Instruction CNT = 0x01 Z = 0 After Instruction CNT = 0x00 Z = 1 | | | | |

DECFSZ Decrement f, Skip if 0

Syntax: `[label] DECFSZ f,d`

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

Operation: $(f) - 1 \rightarrow (\text{dest}); \quad \text{skip if result} = 0$

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 \neq 0,
PC = address HERE+1

GOTO Unconditional Branch

Syntax: `[label] GOTO k`

Operands: $0 \leq k \leq 2047$

Operation: $k \rightarrow \text{PC}<10:0>$
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

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

PIC16C55X

| INCSZ | Increment f, Skip if 0 | | | | | | | | | | | | | | | |
|------------------|---|--------|-------|--------|------|------|------|----------|---|--|--|---|--|--|---|--|
| Syntax: | [<i>label</i>] INCSZ 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: | <table><tr><td>00</td><td>1111</td><td>dfff</td><td>ffff</td></tr></table> | 00 | 1111 | dfff | ffff | | | | | | | | | | | |
| 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 | <table><tr><td>HERE</td><td>INCSZ</td><td>CNT, 1</td></tr><tr><td></td><td>GOTO</td><td>LOOP</td></tr><tr><td>CONTINUE</td><td>.</td><td></td></tr><tr><td></td><td>.</td><td></td></tr><tr><td></td><td>.</td><td></td></tr></table> <p>Before Instruction</p> <p>PC = address HERE</p> <p>After Instruction</p> <p>CNT = CNT + 1</p> <p>if CNT = 0,</p> <p>PC = address CONTINUE</p> <p>if CNT \neq 0,</p> <p>PC = address HERE +1</p> | HERE | INCSZ | CNT, 1 | | GOTO | LOOP | CONTINUE | . | | | . | | | . | |
| HERE | INCSZ | CNT, 1 | | | | | | | | | | | | | | |
| | GOTO | LOOP | | | | | | | | | | | | | | |
| CONTINUE | . | | | | | | | | | | | | | | | |
| | . | | | | | | | | | | | | | | | |
| | . | | | | | | | | | | | | | | | |

| IORLW | | Inclusive OR Literal with W | | | | | | | |
|------------------|---|-----------------------------|------|--|--|----|------|------|------|
| Syntax: | [<i>label</i>] IORLW k | | | | | | | | |
| Operands: | $0 \leq k \leq 255$ | | | | | | | | |
| Operation: | (W) .OR. k \rightarrow (W) | | | | | | | | |
| Status Affected: | Z | | | | | | | | |
| Encoding: | <table border="1"><tr><td>11</td><td>1000</td><td>kkkk</td><td>kkkk</td></tr></table> | | | | | 11 | 1000 | kkkk | kkkk |
| 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: | [<i>label</i>] IORWF f,d | | | | |
| Operands: | $0 \leq f \leq 127$ $d \in [0,1]$ | | | | |
| Operation: | (W) .OR. (f) \rightarrow (dest) | | | | |
| Status Affected: | Z | | | | |
| Encoding: | <table border="1"><tr><td>00</td><td>0100</td><td>dfff</td><td>ffff</td></tr></table> | 00 | 0100 | dfff | ffff |
| 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 | | | | |

| MOVLW | Move Literal to W | | | | |
|------------------|--|------|------|------|------|
| Syntax: | [<i>label</i>] MOVLW k | | | | |
| Operands: | 0 ≤ k ≤ 255 | | | | |
| Operation: | k → (W) | | | | |
| Status Affected: | None | | | | |
| Encoding: | <table><tr><td>11</td><td>00xx</td><td>kkkk</td><td>kkkk</td></tr></table> | 11 | 00xx | kkkk | kkkk |
| 11 | 00xx | kkkk | kkkk | | |
| Description: | The eight bit literal 'k' is loaded into W register. The don't cares will assemble as 0's. | | | | |
| Words: | 1 | | | | |
| Cycles: | 1 | | | | |
| Example | MOVLW 0x5A | | | | |
| | After Instruction | | | | |
| | W = 0x5A | | | | |

| MOVF | Move f | | | | |
|------------------|---|------|------|------|------|
| Syntax: | [<i>label</i>] MOVF f,d | | | | |
| Operands: | $0 \leq f \leq 127$ $d \in [0,1]$ | | | | |
| Operation: | (f) \rightarrow (dest) | | | | |
| Status Affected: | Z | | | | |
| Encoding: | <table><tr><td>00</td><td>1000</td><td>dfff</td><td>ffff</td></tr></table> | 00 | 1000 | dfff | ffff |
| 00 | 1000 | dfff | ffff | | |
| Description: | The contents of register f is moved to a destination dependant upon the status of d. If d = 0, destination is W register. If d = 1, the destination is file register f itself. d = 1 is useful to test a file register since status flag Z is affected. | | | | |
| Words: | 1 | | | | |
| Cycles: | 1 | | | | |
| Example | <pre>MOVF FSR, 0</pre> <p>After Instruction</p> <p>W = value in FSR register</p> <p>Z = 1</p> | | | | |

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

| NOP | | No Operation | | | |
|------------------|----------------------|--------------|------|------|--|
| Syntax: | [<i>label</i>] NOP | | | | |
| Operands: | None | | | | |
| Operation: | No operation | | | | |
| Status Affected: | None | | | | |
| Encoding: | 00 | 0000 | 0xx0 | 0000 | |
| Description: | No operation. | | | | |
| Words: | 1 | | | | |
| Cycles: | 1 | | | | |
| Example | NOP | | | | |

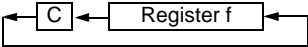
| OPTION | Load Option Register | | | | |
|--|---|--|------|------|------|
| Syntax: | [<i>label</i>] OPTION | | | | |
| Operands: | None | | | | |
| Operation: | (W) → OPTION | | | | |
| Status Affected: | None | | | | |
| Encoding: | <table><tr><td>00</td><td>0000</td><td>0110</td><td>0010</td></tr></table> | 00 | 0000 | 0110 | 0010 |
| 00 | 0000 | 0110 | 0010 | | |
| Description: | <p>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.</p> | | | | |
| Words: | 1 | | | | |
| Cycles: | 1 | | | | |
| Example | <table><tr><td>To maintain upward compatibility with future PIC MCU products, do not use this instruction.</td></tr></table> | To maintain upward compatibility with future PIC MCU products, do not use this instruction. | | | |
| To maintain upward compatibility with future PIC MCU products, do not use this instruction. | | | | | |

PIC16C55X

| 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 | RETFIE | | | | | | | | |
| | After Interrupt | | | | | | | | |
| | PC = TOS | | | | | | | | |
| | GIE = 1 | | | | | | | | |

| RETURN | | Return from Subroutine | | | | | | |
|------------------|--|------------------------|------|--|----|------|------|------|
| Syntax: | [<i>label</i>] RETURN | | | | | | | |
| Operands: | None | | | | | | | |
| Operation: | TOS → PC | | | | | | | |
| Status Affected: | None | | | | | | | |
| Encoding: | <table border="1"><tr><td>00</td><td>0000</td><td>0000</td><td>1000</td></tr></table> | | | | 00 | 0000 | 0000 | 1000 |
| 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 | | | | | | | |

| RETLW | | Return with Literal in W | | | | | | | |
|------------------|---|--------------------------|------|--|--|----|------|------|------|
| Syntax: | [<i>label</i>] RETLW k | | | | | | | | |
| Operands: | $0 \leq k \leq 255$ | | | | | | | | |
| Operation: | $k \rightarrow (W)$; TOS \rightarrow PC | | | | | | | | |
| Status Affected: | None | | | | | | | | |
| Encoding: | <table><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 . . . ADDWF PC ;W = offset RETLW k1 ;Begin table RETLW k2 ; . . . RETLW kn ; End of table</pre> <p>Before Instruction</p> <p>W = 0x07</p> <p>After Instruction</p> <p>W = value of k8</p> | | | | | | | | |

| RLF | Rotate Left f through Carry | | | | |
|------------------|--|------|------|------|------|
| Syntax: | [<i>label</i>] RLF <i>f</i> , <i>d</i> | | | | |
| Operands: | $0 \leq f \leq 127$ $d \in [0,1]$ | | | | |
| Operation: | See description below | | | | |
| Status Affected: | C | | | | |
| Encoding: | <table><tr><td>00</td><td>1101</td><td>dfff</td><td>ffff</td></tr></table> | 00 | 1101 | dfff | ffff |
| 00 | 1101 | dfff | ffff | | |
| Description: | <p>The contents of register 'f' are rotated one bit to the left through the Carry Flag. If 'd' is 0 the result is placed in the W register. If 'd' is 1 the result is stored back in register 'f'.</p>  | | | | |
| Words: | 1 | | | | |
| Cycles: | 1 | | | | |
| Example | <pre>RLF REG1,0</pre> <p>Before Instruction</p> <pre>REG1 = 1110 0110 C = 0</pre> <p>After Instruction</p> <pre>REG1 = 1110 0110 W = 1100 1100 C = 1</pre> | | | | |

RRF Rotate Right f through Carry

Syntax: [*label*] RRF *f*,*d*

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

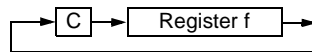
Operation: See description below

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

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. Timeout status bit, \overline{TO} is set. Watchdog Timer and its prescaler are cleared. The processor is put into SLEEP mode with the oscillator stopped. See Section 6.8 for more details.

Words: 1

Cycles: 1

Example: SLEEP

SUBLW Subtract W from Literal

Syntax: [*label*] SUBLW *k*

Operands: $0 \leq k \leq 255$

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

Status Affected: C, DC, Z

Encoding:

| | | | |
|----|------|------|------|
| 11 | 110x | kkkk | kkkk |
|----|------|------|------|

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

Words: 1

Cycles: 1

Example 1: SUBLW 0x02

Before Instruction

W = 1
C = ?

After Instruction

W = 1
C = 1; result is positive

Example 2:

Before Instruction

W = 2
C = ?

After Instruction

W = 0
C = 1; result is zero

Example 3:

Before Instruction

W = 3
C = ?

After Instruction

W = 0xFF
C = 0; result is negative

PIC16C55X

SUBWF Subtract W from f

| | | | | | |
|--------------|--|------|------|------|------|
| Syntax: | [<i>label</i>] 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: | <table><tr><td>00</td><td>0010</td><td>dfff</td><td>ffff</td></tr></table> | 00 | 0010 | dfff | ffff |
| 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: | [<i>label</i>] SWAPF f,d | | | |
| Operands: | $0 \leq f \leq 127$ $d \in [0,1]$ | | | |
| Operation: | $(f<3:0>) \rightarrow (\text{dest}<7:4>),$ $(f<7:4>) \rightarrow (\text{dest}<3:0>)$ | | | |
| Status Affected: | None | | | |
| Encoding: | 00 | 1110 | dfff | ffff |
| Description: | The upper and lower nibbles of register 'f' are exchanged. If 'd' is 0 the result is placed in W register. If 'd' is 1 the result is placed in register 'f'. | | | |
| Words: | 1 | | | |
| Cycles: | 1 | | | |
| Example | SWAPF REG, 0 | | | |

Before Instruction

REG1 = 0xA5

After Instruction

REG1 = 0xA5
W = 0x5A

| TRIS | Load TRIS Register | | | | |
|--|---|--|------|------|------|
| Syntax: | [<i>label</i>] TRIS f | | | | |
| Operands: | $5 \leq f \leq 7$ | | | | |
| Operation: | (W) → TRIS register f; | | | | |
| Status Affected: | None | | | | |
| Encoding: | <table><tr><td>00</td><td>0000</td><td>0110</td><td>0fff</td></tr></table> | 00 | 0000 | 0110 | 0fff |
| 00 | 0000 | 0110 | 0fff | | |
| Description: | The instruction is supported for code compatibility with the PIC16C5X products. Since TRIS registers are readable and writable, the user can directly address them. | | | | |
| Words: | 1 | | | | |
| Cycles: | 1 | | | | |
| Example | <table><tr><td>To maintain upward compatibility with future PIC MCU products, do not use this instruction.</td></tr></table> | To maintain upward compatibility with future PIC MCU products, do not use this instruction. | | | |
| To maintain upward compatibility with future PIC MCU products, do not use this instruction. | | | | | |

XORLW Exclusive OR Literal with W

| | | | | | |
|------------------|--|------|------|------|------|
| Syntax: | [<i>label</i>] XORLW k | | | | |
| Operands: | $0 \leq k \leq 255$ | | | | |
| Operation: | (W) .XOR. k \rightarrow (W) | | | | |
| Status Affected: | Z | | | | |
| Encoding: | <table border="1"><tr><td>11</td><td>1010</td><td>kkkk</td><td>kkkk</td></tr></table> | 11 | 1010 | kkkk | kkkk |
| 11 | 1010 | kkkk | kkkk | | |
| Description: | The contents of the W register are XOR'ed with the eight bit literal 'k'. The result is placed in the W register. | | | | |
| Words: | 1 | | | | |
| Cycles: | 1 | | | | |
| Example: | <pre>XORLW 0xAF</pre> <p>Before Instruction</p> <p>W = 0xB5</p> <p>After Instruction</p> <p>W = 0x1A</p> | | | | |

XORWF Exclusive OR W with f

| | | | | | |
|------------------|---|------|------|------|------|
| Syntax: | [<i>label</i>] XORWF f,d | | | | |
| Operands: | $0 \leq f \leq 127$ $d \in [0,1]$ | | | | |
| Operation: | (W) .XOR. (f) \rightarrow (dest) | | | | |
| Status Affected: | Z | | | | |
| Encoding: | <table border="1"><tr><td>00</td><td>0110</td><td>dfff</td><td>ffff</td></tr></table> | 00 | 0110 | dfff | ffff |
| 00 | 0110 | dfff | ffff | | |
| Description: | Exclusive OR the contents of the W register with register 'f'. If 'd' is 0 the result is stored in the W register. If 'd' is 1 the result is stored back in register 'f'. | | | | |
| Words: | 1 | | | | |
| Cycles: | 1 | | | | |
| Example | <pre>XORWF REG 1</pre> <p>Before Instruction</p> <p>REG = 0xAF W = 0xB5</p> <p>After Instruction</p> <p>REG = 0x1A W = 0xB5</p> | | | | |

PIC16C55X

NOTES:

9.0 DEVELOPMENT SUPPORT

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

- Integrated Development Environment
 - MPLAB® IDE Software
- Assemblers/Compilers/Linkers
 - MPASM™ Assembler
 - MPLAB C17 and MPLAB C18 C Compilers
 - MPLINK™ Object Linker/
MPLIB™ Object Librarian
- Simulators
 - MPLAB SIM Software Simulator
- Emulators
 - MPLAB ICE 2000 In-Circuit Emulator
 - ICEPIC™ In-Circuit Emulator
- In-Circuit Debugger
 - MPLAB ICD
- Device Programmers
 - PRO MATE® II Universal Device Programmer
 - PICSTART® Plus Entry-Level Development Programmer
- Low Cost Demonstration Boards
 - PICDEM™ 1 Demonstration Board
 - PICDEM 2 Demonstration Board
 - PICDEM 3 Demonstration Board
 - PICDEM 17 Demonstration Board
 - KEELQ® Demonstration Board

9.1 MPLAB Integrated Development Environment Software

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

- An interface to debugging tools
 - simulator
 - programmer (sold separately)
 - emulator (sold separately)
 - in-circuit debugger (sold separately)
- A full-featured editor
- A project manager
- Customizable toolbar and key mapping
- A status bar
- On-line help

The MPLAB IDE allows you to:

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

The ability to use MPLAB IDE with multiple debugging tools allows users to easily switch from the cost-effective simulator to a full-featured emulator with minimal retraining.

9.2 MPASM Assembler

The MPASM assembler is a full-featured universal macro assembler for all PIC MCUs.

The MPASM assembler has a command line interface and a Windows shell. It can be used as a stand-alone application on a Windows 3.x or greater system, or it can be used through MPLAB IDE. The MPASM assembler generates relocatable object files for the MPLINK object linker, Intel® standard HEX files, MAP files to detail memory usage and symbol reference, an absolute LST file that contains source lines and generated machine code, and a COD file for debugging.

The MPASM assembler features include:

- Integration into MPLAB IDE projects.
- User-defined macros to streamline assembly code.
- Conditional assembly for multi-purpose source files.
- Directives that allow complete control over the assembly process.

9.3 MPLAB C17 and MPLAB C18 C Compilers

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

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

9.4 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK object linker combines relocatable objects created by the MPASM assembler and the MPLAB C17 and MPLAB C18 C compilers. It can also link relocatable objects from pre-compiled libraries, using directives from a linker script.

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

The MPLINK object linker features include:

- Integration with MPASM assembler and MPLAB C17 and MPLAB C18 C compilers.
- Allows all memory areas to be defined as sections to provide link-time flexibility.

The MPLIB object librarian features include:

- Easier linking because single libraries can be included instead of many smaller files.
- Helps keep code maintainable by grouping related modules together.
- Allows libraries to be created and modules to be added, listed, replaced, deleted or extracted.

9.5 MPLAB SIM Software Simulator

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

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

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

The MPLAB ICE universal in-circuit emulator is intended to provide the product development engineer with a complete microcontroller design tool set for PIC microcontrollers (MCUs). Software control of the MPLAB ICE in-circuit emulator is provided by the MPLAB Integrated Development Environment (IDE), which allows editing, building, downloading and source debugging from a single environment.

The MPLAB ICE 2000 is a full-featured emulator system with enhanced trace, trigger and data monitoring features. Interchangeable processor modules allow the system to be easily re configured for emulation of different processors. The universal architecture of the MPLAB ICE in-circuit emulator allows expansion to support new PIC microcontrollers.

The MPLAB ICE in-circuit 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 platform and Microsoft® Windows environment were chosen to best make these features available to you, the end user.

9.7 ICEPIC In-Circuit Emulator

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

9.8 MPLAB ICD In-Circuit Debugger

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

9.9 PRO MATE II Universal Device Programmer

The PRO MATE II universal device programmer is a full-featured programmer, capable of operating in Stand-alone mode, as well as PC-hosted mode. The PRO MATE II device programmer is CE compliant.

The PRO MATE II device programmer has programmable VDD and VPP supplies, which allow it to verify programmed memory at VDD min and VDD max for maximum reliability. It has an LCD display for instructions and error messages, keys to enter commands and a modular detachable socket assembly to support various package types. In Stand-alone mode, the PRO MATE II device programmer can read, verify, or program PIC devices. It can also set code protection in this mode.

9.10 PICSTART Plus Entry Level Development Programmer

The PICSTART Plus development programmer is an easy-to-use, low cost, prototype programmer. It connects to the PC via a COM (RS-232) port. MPLAB Integrated Development Environment software makes using the programmer simple and efficient.

The PICSTART Plus development programmer supports all PIC devices with up to 40 pins. Larger pin count devices, such as the PIC16C92X and PIC17C76X, may be supported with an adapter socket. The PICSTART Plus development programmer is CE compliant.

9.11 PICDEM 1 Low Cost PIC MCU Demonstration Board

The PICDEM 1 demonstration board 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 user can program the sample microcontrollers provided with the PICDEM 1 demonstration board on a PRO MATE II device programmer, or a PICSTART Plus development programmer, and easily test firmware. The user can also connect the PICDEM 1 demonstration board to the MPLAB ICE in-circuit emulator and download the firmware to the emulator for testing. A 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.

9.12 PICDEM 2 Low Cost PIC16CXX Demonstration Board

The PICDEM 2 demonstration board 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 demonstration board on a PRO MATE II device programmer, or a PICSTART Plus development programmer, and easily test firmware. The MPLAB ICE in-circuit emulator may also be used with the PICDEM 2 demonstration board to test firmware. A 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 I²C™ bus and separate headers for connection to an LCD module and a keypad.

9.13 PICDEM 3 Low Cost PIC16CXXX Demonstration Board

The PICDEM 3 demonstration board 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 an 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 demonstration board on a PRO MATE II device programmer, or a PICSTART Plus development programmer with an adapter socket, and easily test firmware. The MPLAB ICE in-circuit emulator may also be used with the PICDEM 3 demonstration board to test firmware. A prototype area has been provided to the user for adding 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 thermistor and separate headers for connection to an external LCD module and a keypad. Also provided on the PICDEM 3 demonstration board is a LCD panel, with 4 commons and 12 segments, that is capable of displaying time, temperature and day of the week. The PICDEM 3 demonstration board provides an additional RS-232 interface and Windows 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.

9.14 PICDEM 17 Demonstration Board

The PICDEM 17 demonstration board is an evaluation board that demonstrates the capabilities of several Microchip microcontrollers, including PIC17C752, PIC17C756A, PIC17C762 and PIC17C766. All necessary hardware is included to run basic demo programs, which are supplied on a 3.5-inch disk. A programmed sample is included and the user may erase it and program it with the other sample programs using the PRO MATE II device programmer, or the PICSTART Plus development programmer, and easily debug and test the sample code. In addition, the PICDEM 17 demonstration board supports downloading of programs to and executing out of external FLASH memory on board. The PICDEM 17 demonstration board is also usable with the MPLAB ICE in-circuit emulator, or the PICMASTER emulator and all of the sample programs can be run and modified using either emulator. Additionally, a generous prototype area is available for user hardware.

9.15 KEELoQ Evaluation and Programming Tools

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

TABLE 9-1: DEVELOPMENT TOOLS FROM MICROCHIP

| | PIC12CXXX | PIC14000 | PIC16C5X | PIC16C6X | PIC16CXX | PIC16C7X | PIC16C7XX | PIC16C8X | PIC16F8XX | PIC16G9XX | PIC17C4X | PIC17C7XX | PIC18CXX2 | PIC18FXX | 24CXX/ 25CXX/ 93CXX | HCXXX | MCRFXXX | MCP2510 |
|---------------------------|---|----------|----------|----------|----------|----------|-----------|----------|-----------|-----------|----------|-----------|-----------|----------|---------------------------|-------|---------|---------|
| Software Tools | MPLAB® Integrated Development Environment | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | |
| | MPLAB® C17 C Compiler | | | | | | | | | | ✓ | ✓ | ✓ | | | | | |
| | MPLAB® C18 C Compiler | | | | | | | | | | | | ✓ | ✓ | ✓ | ✓ | | |
| Emulators | MPASM™ Assembler/ MPLINK™ Object Linker | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | | |
| | MPLAB® ICE In-Circuit Emulator | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | | | |
| | ICEPIC™ In-Circuit Emulator | ✓ | | ✓ | ✓ | | ✓ | ✓ | | ✓ | | | | | | | | |
| Debugger | MPLAB® ICD In-Circuit Debugger | | | ✓ | ✓ | ✓ | ✓ | | ✓ | | | | | ✓ | | | | |
| Programmers | PICSTART® Plus Entry Level Development Programmer | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | | |
| | PRO MATE® II Universal Device Programmer | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | |
| Demo Boards and Eval Kits | PICDEM™ 1 Demonstration Board | | ✓ | | | † | | ✓ | | | ✓ | | | | | | | |
| | PICDEM™ 2 Demonstration Board | | | | † | † | | | | | | | ✓ | | | | | |
| | PICDEM™ 3 Demonstration Board | | | | | | | | | ✓ | | | | | | | | |
| | PICDEM™ 14A Demonstration Board | ✓ | | | | | | | | | | | | | | | | |
| | PICDEM™ 17 Demonstration Board | | | | | | | | | | | ✓ | | | | | | |
| | KEELOQ® Evaluation Kit | | | | | | | | | | | | | | | ✓ | | |
| | KEELOQ® Transponder Kit | | | | | | | | | | | | | | | ✓ | | |
| | microID™ Programmer's Kit | | | | | | | | | | | | | | | | ✓ | |
| | 125 kHz microID™ Developer's Kit | | | | | | | | | | | | | | | | ✓ | |
| | 125 kHz Anticollision microID™ Developer's Kit | | | | | | | | | | | | | | | | ✓ | |
| | 13.56 MHz Anticollision microID™ Developer's Kit | | | | | | | | | | | | | | | | ✓ | |
| | MCP2510 CAN Developer's Kit | | | | | | | | | | | | | | | | ✓ | ✓ |

* Contact the Microchip Technology Inc. web site at www.microchip.com for information on how to use the MPLAB® ICD In-Circuit Debugger (DV164001) with PIC16C62, 63, 64, 65, 72, 73, 74, 76, 77.

** Contact Microchip Technology Inc. for availability date.

PIC16C55X

NOTES:

10.0 ELECTRICAL SPECIFICATIONS

Absolute Maximum Ratings †

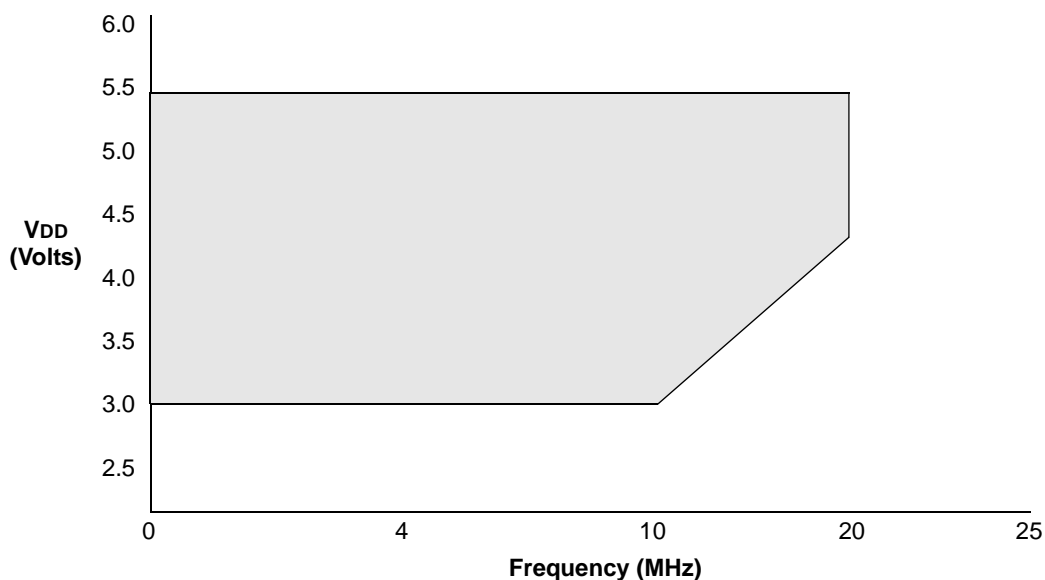
| | |
|---|--------------------|
| Ambient Temperature under bias | -40° to +125°C |
| Storage Temperature | -65° to +150°C |
| Voltage on any pin with respect to VSS (except VDD and $\overline{\text{MCLR}}$) | -0.6V to VDD +0.6V |
| Voltage on VDD with respect to VSS | 0 to +7.5V |
| Voltage on $\overline{\text{MCLR}}$ with respect to VSS..... | 0 to +14V |
| Total power Dissipation (Note 1) | 1.0W |
| Maximum Current out of VSS pin | 300 mA |
| Maximum Current into VDD pin | 250 mA |
| Input Clamp Current, I _{IK} (V _I < 0 or V _I > VDD) | ±20 mA |
| Output Clamp Current, I _{OK} (V _O < 0 or V _O > 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 PORTA, PORTB and PORTC | 200 mA |
| Maximum Current sourced by PORTA, PORTB and PORTC | 200 mA |

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

† **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.

PIC16C55X

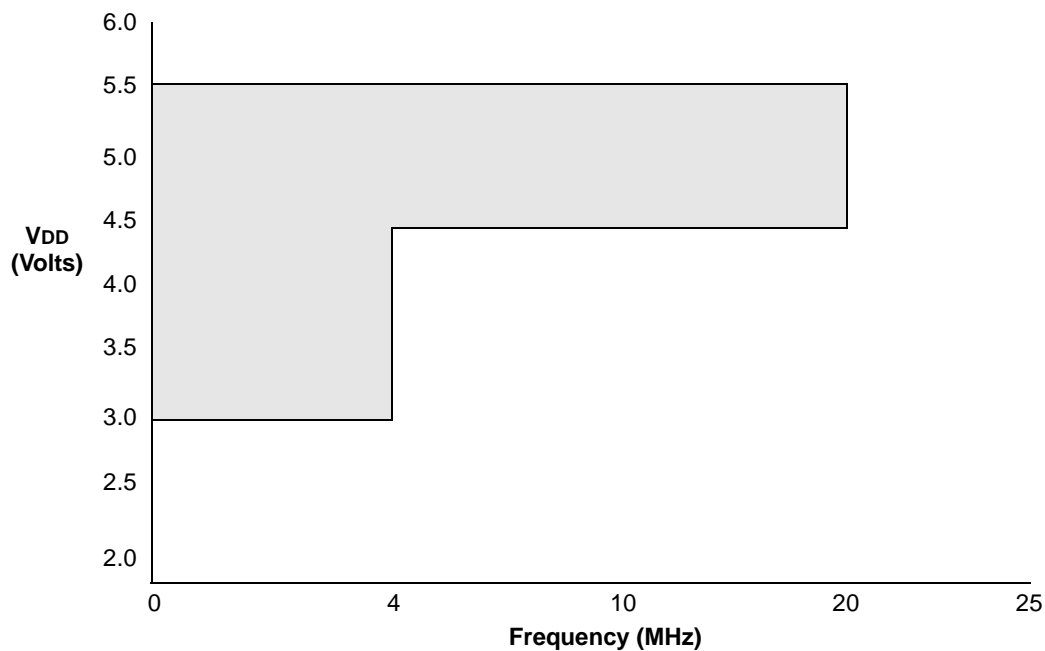
FIGURE 10-1: VOLTAGE-FREQUENCY GRAPH, $0^{\circ}\text{C} \leq T_A \leq +70^{\circ}\text{C}$ (COMMERCIAL TEMPS)



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

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

FIGURE 10-2: VOLTAGE-FREQUENCY GRAPH, $-40^{\circ}\text{C} \leq T_A < 0^{\circ}\text{C}$, $+70^{\circ}\text{C} < T_A \leq +125^{\circ}\text{C}$ (OUTSIDE OF COMMERCIAL TEMPS)



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

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

FIGURE 10-3: VOLTAGE-FREQUENCY GRAPH, $0^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$

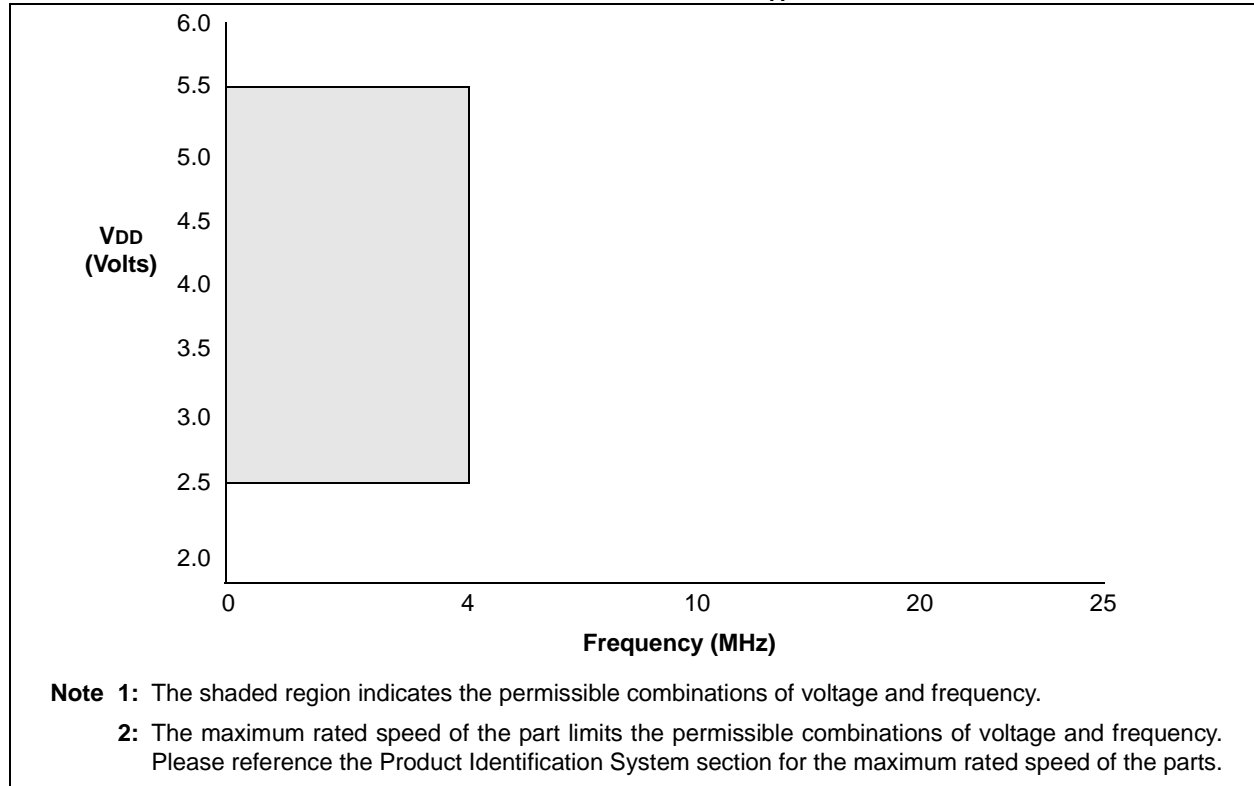
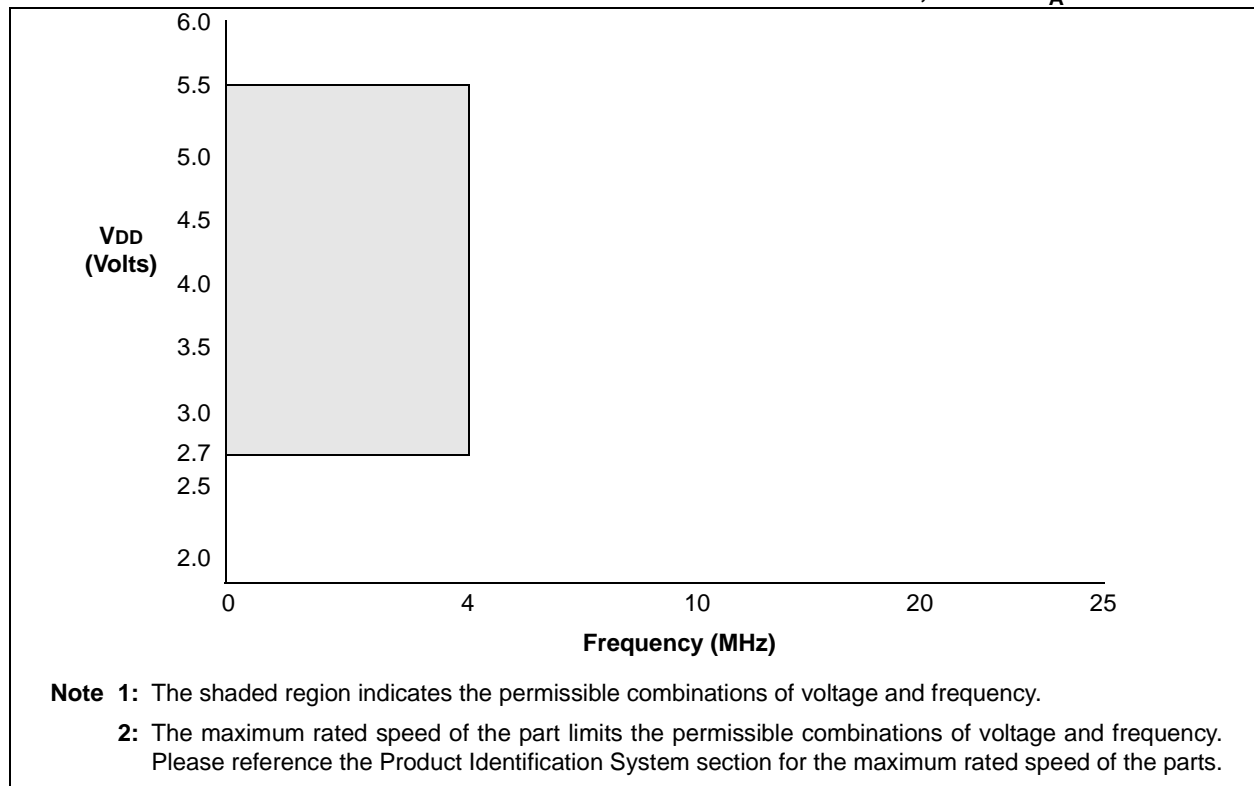


FIGURE 10-4: PIC16LC554/557/558 VOLTAGE-FREQUENCY GRAPH, $-40^{\circ}\text{C} \leq T_A \leq 0^{\circ}\text{C}$



PIC16C55X

10.1 DC Characteristics: PIC16C55X-04 (Commercial, Industrial, Extended) PIC16C55X-20 (Commercial, Industrial, Extended) HCS1365-04 (Commercial, Industrial, Extended)

| DC Characteristics <div> Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial and $0^{\circ}\text{C} \leq T_A \leq +70^{\circ}\text{C}$ for commercial and $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ for extended </div> | | | | | | | |
|---|------|---|------------|--------|------------|--------|--|
| Param No. | Sym | Characteristic | Min | Typ† | Max | Units | Conditions |
| D001 | VDD | Supply Voltage | | | | | |
| | | 16LC55X | 3.0 2.5 | — | 5.5 5.5 | V | XT and RC osc configuration LP osc configuration |
| D001 D001A | | 16C55X | 3.0 4.5 | — — | 5.5 5.5 | V V | XT, RC and LP osc configuration HS osc configuration |
| D002 | VDR | RAM Data Retention Voltage⁽¹⁾ | — | 1.5* | — | V | Device in SLEEP mode |
| D003 | VPOR | VDD Start Voltage to ensure Power-on Reset | — | VSS | — | V | See Section 6.4, Power-on Reset for details |
| D004 | SVDD | VDD Rise Rate to ensure Power-on Reset | 0.05* | — | — | V/ms | See Section 6.4, Power-on Reset for details |
| D010 D010A D010 D010A D013 | IDD | Supply Current⁽²⁾ | | | | | |
| | | 16LC55X | — | 1.4 | 2.5 | mA | XT and RC osc configuration Fosc = 2.0 MHz, VDD = 3.0V, WDT disabled ⁽⁴⁾ |
| | | | — | 26 | 53 | μA | LP osc configuration Fosc = 32 kHz, VDD = 3.0V, WDT disabled |
| | | 16C55X | — | 1.8 | 3.3 | mA | XT and RC osc configuration Fosc = 4 MHz, VDD = 5.5V, WDT disabled ⁽⁴⁾ |
| | | | — | 35 | 70 | μA | LP osc configuration, PIC16C55X-04 only Fosc = 32 kHz, VDD = 4.0V, WDT disabled |
| | | | — | 9.0 | 20 | mA | HS osc configuration Fosc = 20 MHz, VDD = 5.5V, WDT disabled |

* 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.

Note 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 configured as input, pulled to VDD,

MCLR = VDD; WDT enabled/disabled as specified.

Note 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 configured as input and tied to VDD or VSS.

Note 4: For RC 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 kΩ.

Note 5: The Δ current is the additional current consumed when this peripheral is enabled. This current should be added to the base IDD or IPD measurement.

10.1 DC Characteristics: PIC16C55X-04 (Commercial, Industrial, Extended) PIC16C55X-20 (Commercial, Industrial, Extended) HCS1365-04 (Commercial, Industrial, Extended)

| DC Characteristics | | Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial and 0°C ≤ TA ≤ +70°C for commercial and -40°C ≤ TA ≤ +125°C for extended | | | | | |
|--------------------|-------------------|---|-----|------|-----|-------|---------------------------------|
| Param No. | Sym | Characteristic | Min | Typ† | Max | Units | Conditions |
| D020 | IPD | Power-Down Current⁽³⁾ | | | | | |
| | | 16LC55X | — | 0.7 | 2 | μA | VDD = 3.0V, WDT disabled |
| | | 16C55X | — | 1.0 | 2.5 | μA | VDD = 4.0V, WDT disabled |
| | ΔI _{WDT} | WDT Current⁽⁵⁾ | | | | | |
| | | 16LC55X | — | 6.0 | 15 | μA | VDD = 3.0V |
| | | 16C55X | — | 6.0 | 20 | μA | VDD = 4.0V (+85°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 configured as input, 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 configured as input and tied to VDD or VSS.

4: For RC 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 kΩ.

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

PIC16C55X

10.2 DC Characteristics: PIC16C55X (Commercial, Industrial, Extended) PIC16LC55X(Commercial, Industrial, Extended)

| Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for industrial and $0^{\circ}\text{C} \leq T_A \leq +70^{\circ}\text{C}$ for commercial and $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ for automotive Operating voltage V_{DD} range as described in DC spec Table 10-1 | | | | | | | |
|--|-----------------|---|------------------------------------|------|------------------------------------|--------|--|
| DC Characteristics | | | | | | | |
| Param. No. | Sym | Characteristic | Min | Typ† | Max | Unit | Conditions |
| D030 | V _{IL} | Input Low Voltage | | | | | |
| | | I/O ports | | | | | |
| | | with TTL buffer | V _{SS} | — | 0.8V 0.15 V _{DD} | V | V _{DD} = 4.5V to 5.5V otherwise |
| | | with Schmitt Trigger input | V _{SS} | — | 0.2 V _{DD} | V | |
| | | MCLR, RA4/T0CKI, OSC1 (in RC mode) | V _{SS} | — | 0.2 V _{DD} | V | (Note1) |
| D033 | | OSC1 (in XT* and HS) | V _{SS} | — | 0.3 V _{DD} | V | |
| | | OSC1 (in LP*) | V _{SS} | — | 0.6 V _{DD} -1.0 | V | |
| D040 | V _{IH} | Input High Voltage | | | | | |
| | | I/O ports | | | | | |
| | | with TTL buffer | 2.0V 0.8 + 0.25 V _{DD} | — | V _{DD} V _{DD} | V V | V _{DD} = 4.5V to 5.5V otherwise |
| | | with Schmitt Trigger input | 0.8V | — | V _{DD} | V | |
| | | MCLR RA4/T0CKI | 0.8 V _{DD} | — | V _{DD} | V | |
| | | OSC1 (XT*, HS and LP*) | 0.7 V _{DD} | — | V _{DD} | V | |
| D043A | | OSC1 (in RC mode) | 0.9 V _{DD} | — | | | (Note1) |
| D070 | IPURB | PORTB weak pull-up current | 50 | 200 | 400 | μA | V _{DD} = 5.0V, V _{PIN} = V _{SS} |
| D060 | I _{IL} | Input Leakage Current⁽²⁾⁽³⁾ | | | | | |
| | | I/O ports (Except PORTA) | | | ±1.0 | μA | V _{SS} ≤ V _{PIN} ≤ V _{DD} , pin at hi-impedance |
| | | PORTA | — | — | ±0.5 | μA | V _{SS} ≤ V _{PIN} ≤ V _{DD} , pin at hi-impedance |
| | | RA4/T0CKI | — | — | ±1.0 | μA | V _{SS} ≤ V _{PIN} ≤ V _{DD} |
| | | OSC1, MCLR | — | — | ±5.0 | μA | V _{SS} ≤ V _{PIN} ≤ V _{DD} , XT, HS and LP osc configuration |
| D080 | V _{OL} | Output Low Voltage | | | | | |
| | | I/O ports | — | — | 0.6 | V | I _{OL} =8.5 mA, V _{DD} =4.5V, -40° to +85°C |
| | | | — | — | 0.6 | V | I _{OL} =7.0 mA, V _{DD} =4.5V, +125°C |
| | | OSC2/CLKOUT | — | — | 0.6 | V | I _{OL} =1.6 mA, V _{DD} =4.5V, -40° to +85°C |
| | | (RC only) | — | — | 0.6 | V | I _{OL} =1.2 mA, V _{DD} =4.5V, +125°C |
| D090 | V _{OH} | Output High Voltage⁽³⁾ | | | | | |
| | | I/O ports (Except RA4) | V _{DD} -0.7 | — | — | V | I _{OH} =-3.0 mA, V _{DD} =4.5V, -40° to +85°C |

* These parameters are characterized but not tested.

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

- Note 1:** In RC oscillator configuration, the OSC1 pin is a Schmitt Trigger input. It is not recommended that the PIC16C55X be driven with external clock in RC mode.
- Note 2:** The leakage current on the MCLR pin is strongly dependent on applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.
- Note 3:** Negative current is defined as coming out of the pin.

10.2 DC Characteristics: PIC16C55X (Commercial, Industrial, Extended) PIC16LC55X(Commercial, Industrial, Extended) (Continued)

| DC Characteristics | | Standard Operating Conditions (unless otherwise stated) | | | | | |
|---|-------|--|---------|------|-----|------|--|
| | | Operating temperature -40°C ≤ TA ≤ +85°C for industrial and 0°C ≤ TA ≤ +70°C for commercial and -40°C ≤ TA ≤ +125°C for automotive Operating voltage VDD range as described in DC spec Table 10-1 | | | | | |
| Param. No. | Sym | Characteristic | Min | Typ† | Max | Unit | Conditions |
| D092 | | OSC2/CLKOUT | VDD-0.7 | — | — | V | IOH=-2.5 mA, VDD=4.5V, +125°C |
| | | | VDD-0.7 | — | — | V | IOH=-1.3 mA, VDD=4.5V, -40° to +85°C |
| | | (RC only) | VDD-0.7 | — | — | V | IOH=-1.0 mA, VDD=4.5V, +125°C |
| * | VOD | Open-Drain High Voltage | | | 10* | V | RA4 pin |
| Capacitive Loading Specs on Output Pins | | | | | | | |
| D100 | COSC2 | OSC2 pin | | | 15 | pF | In XT, HS and LP modes when external clock used to drive OSC1. |
| D101 | CIO | All I/O pins/OSC2 (in RC mode) | | | 50 | pF | |

* These parameters are characterized but not tested.

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

- Note** 1: In RC oscillator configuration, the OSC1 pin is a Schmitt Trigger input. It is not recommended that the PIC16C55X be driven with external clock in RC mode.
- 2: The leakage current on the MCLR pin is strongly dependent on applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.
- 3: Negative current is defined as coming out of the pin.

PIC16C55X

10.3 Timing Parameter Symbolology

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

- 1. TppS2ppS
- 2. TppS

| | | |
|----------|-----------|------|
| T | | |
| F | Frequency | T |
| | | Time |

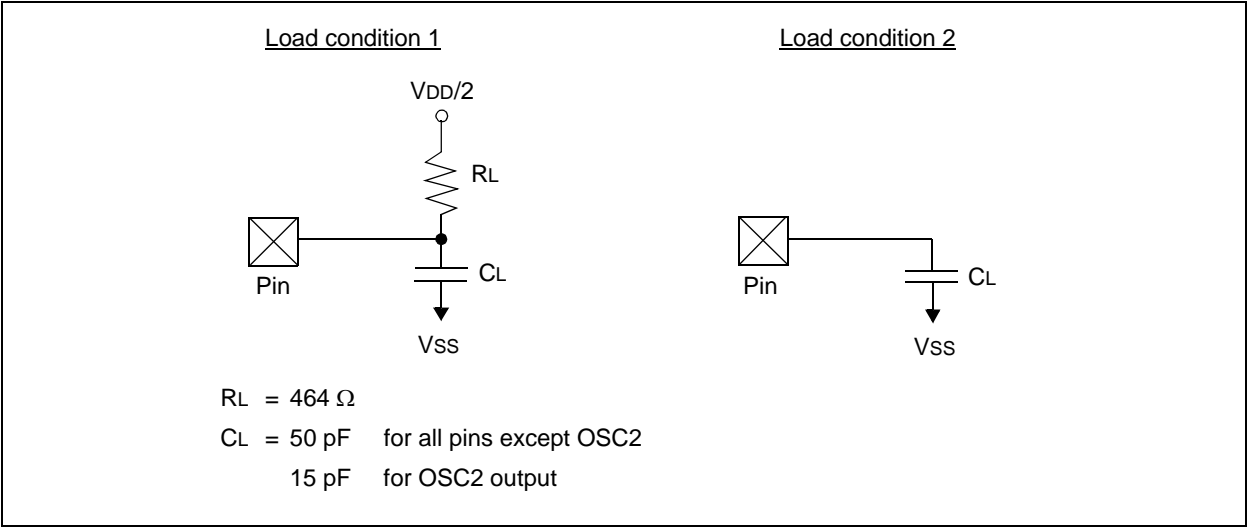
Lowercase subscripts (pp) and their meanings:

| | | |
|-----------|----------|-------|
| pp | | |
| ck | CLKOUT | os |
| io | I/O port | t0 |
| mc | MCLR | T0CKI |

Uppercase letters and their meanings:

| | | |
|----------|------------------------|--------------|
| S | | |
| F | Fall | P |
| H | High | R |
| I | Invalid (Hi-impedance) | V |
| L | Low | Z |
| | | Hi-impedance |

FIGURE 10-5: LOAD CONDITIONS



10.4 Timing Diagrams and Specifications

FIGURE 10-6: EXTERNAL CLOCK TIMING

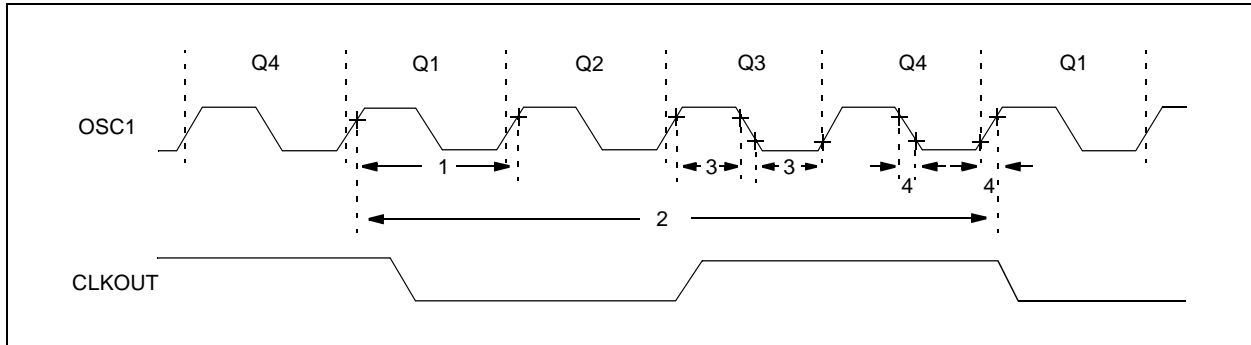


TABLE 10-1: EXTERNAL CLOCK TIMING REQUIREMENTS

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

* These parameters are characterized but not tested.

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

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

PIC16C55X

FIGURE 10-7: CLKOUT AND I/O TIMING

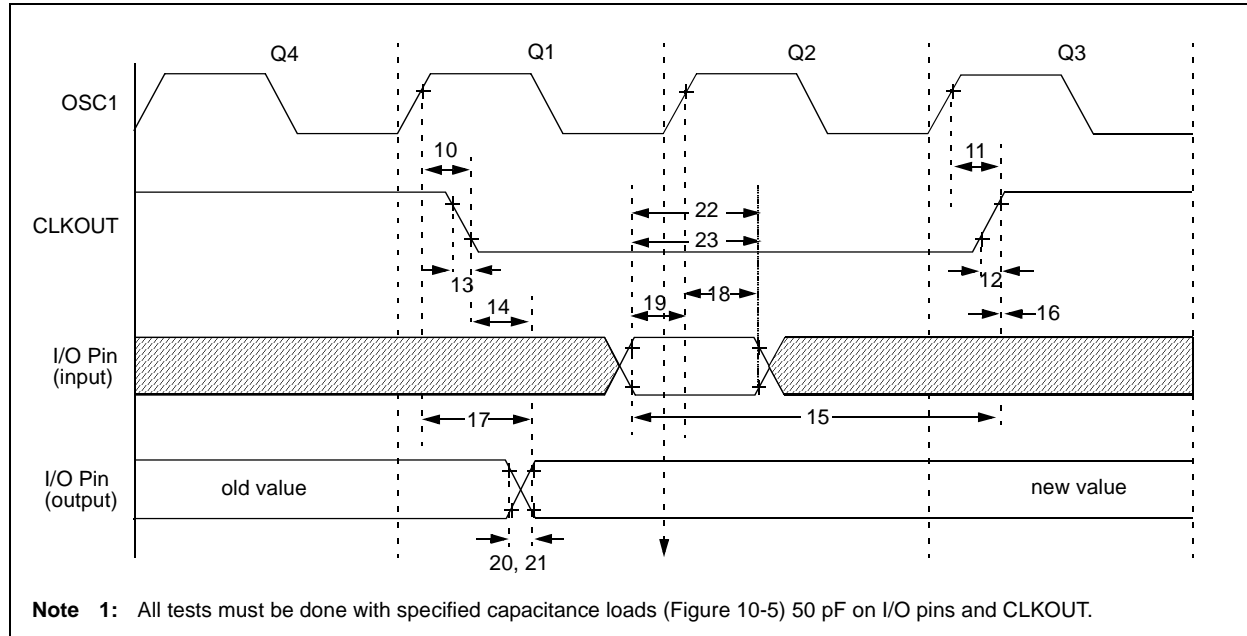


TABLE 10-2: CLKOUT AND I/O TIMING REQUIREMENTS

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

* These parameters are characterized but not tested.

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

Note 1: Measurements are taken in RC mode where CLKOUT output is 4 x Tosc.

PIC16C55X

FIGURE 10-8: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING

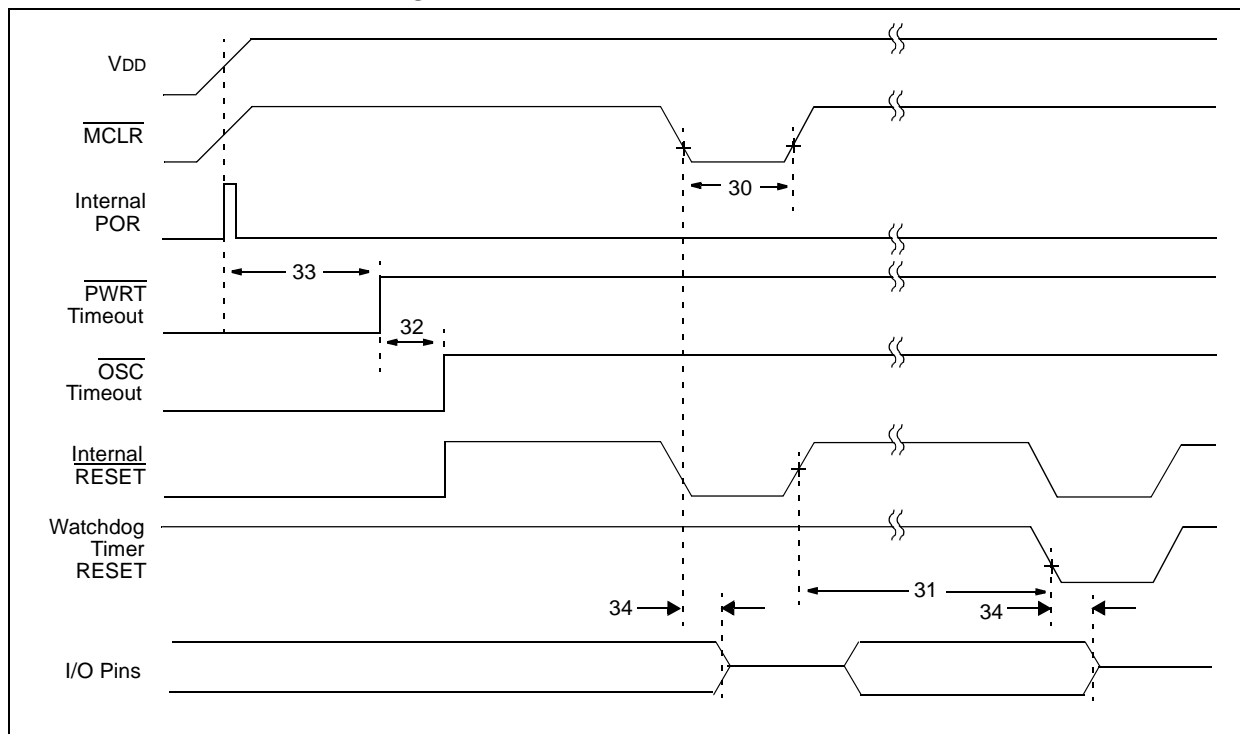


TABLE 10-3: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER REQUIREMENTS

| Param No. | Sym | Characteristic | Min | Typ† | Max | Units | Conditions |
|-----------|-------|--|------|-----------|------|-------|---------------------------|
| 30 | Tmcl | MCLR Pulse Width (low) | 2000 | — | — | ns | -40° to +85°C |
| 31 | Twdt | Watchdog Timer Timeout Period (No Prescaler) | 7* | 18 | 33* | ms | VDD = 5.0V, -40° to +85°C |
| 32 | Tost | Oscillation Start-up Timer Period | — | 1024 TOSC | — | — | TOSC = OSC1 period |
| 33 | Tpwrt | Power-up Timer Period | 28* | 72 | 132* | ms | VDD = 5.0V, -40° to +85°C |
| 34 | Tioz | I/O hi-impedance from MCLR low | — | — | 2.0* | μs | |

* These parameters are characterized but not tested.

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

FIGURE 10-9: TIMER0 CLOCK TIMING

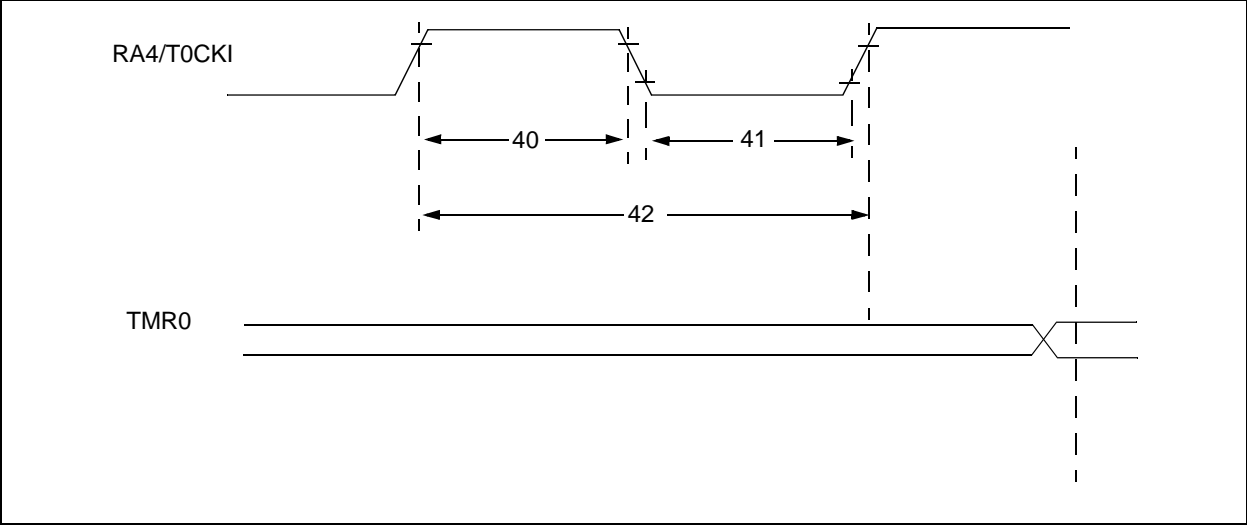


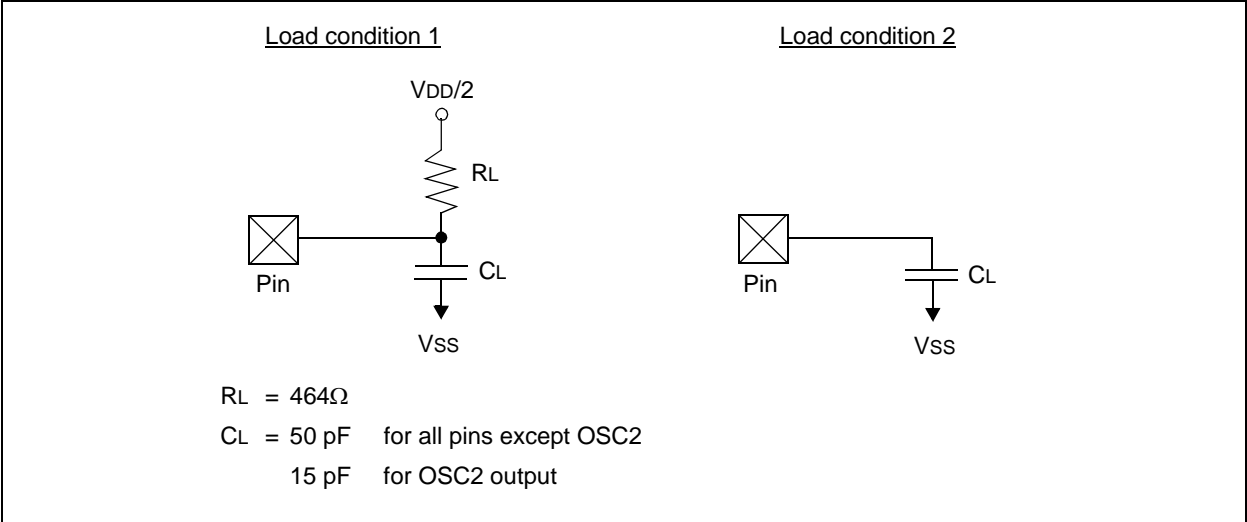
TABLE 10-4: TIMER0 CLOCK REQUIREMENTS

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

* These parameters are characterized but not tested.

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

FIGURE 10-10: LOAD CONDITIONS



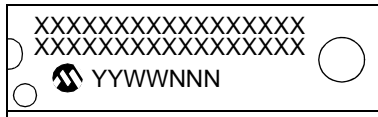
PIC16C55X

NOTES:

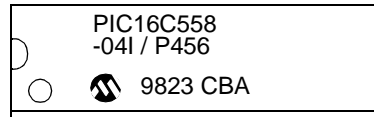
11.0 PACKAGING INFORMATION

11.1 Package Marking Information

18-Lead PDIP



Example



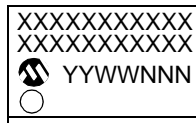
28-Lead PDIP



Example



20-Lead SSOP



Example



28-Lead SSOP



Example



| | | |
|----------------|--------|--|
| Legend: | XX...X | Customer-specific information |
| | Y | Year code (last digit of calendar year) |
| | YY | Year code (last 2 digits of calendar year) |
| | WW | Week code (week of January 1 is week '01') |
| | NNN | Alphanumeric traceability code |
| | (e3) | Pb-free JEDEC designator for Matte Tin (Sn) |
| | * | This package is Pb-free. The Pb-free JEDEC designator (e3) can be found on the outer packaging for this package. |

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.

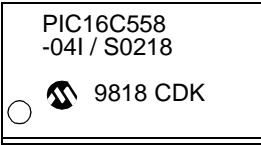
PIC16C55X

Package Marking Information (Cont'd)

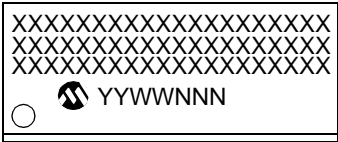
18-Lead SOIC (.300")



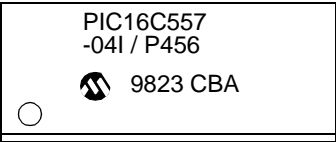
Example



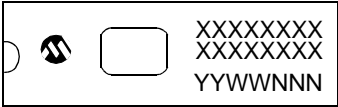
28-Lead SOIC (.300")



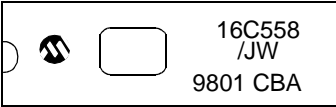
Example



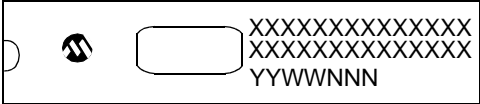
18-Lead Cerdip Windowed



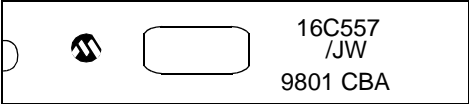
Example



28-Lead Cerdip Windowed

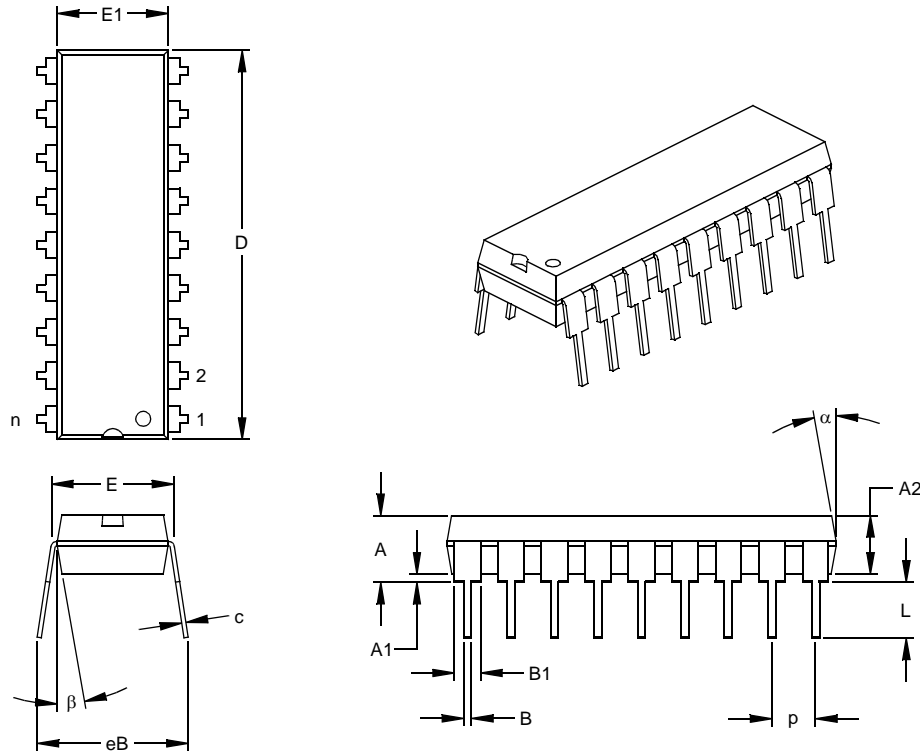


Example



18-Lead Plastic Dual In-line (P) – 300 mil (PDIP)

Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



| Units | | INCHES* | | | MILLIMETERS | | |
|----------------------------|------|---------|------|------|-------------|-------|-------|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 18 | | | 18 | |
| Pitch | p | | .100 | | | 2.54 | |
| Top to Seating Plane | A | .140 | .155 | .170 | 3.56 | 3.94 | 4.32 |
| Molded Package Thickness | A2 | .115 | .130 | .145 | 2.92 | 3.30 | 3.68 |
| Base to Seating Plane | A1 | .015 | | | 0.38 | | |
| Shoulder to Shoulder Width | E | .300 | .313 | .325 | 7.62 | 7.94 | 8.26 |
| Molded Package Width | E1 | .240 | .250 | .260 | 6.10 | 6.35 | 6.60 |
| Overall Length | D | .890 | .898 | .905 | 22.61 | 22.80 | 22.99 |
| Tip to Seating Plane | L | .125 | .130 | .135 | 3.18 | 3.30 | 3.43 |
| Lead Thickness | c | .008 | .012 | .015 | 0.20 | 0.29 | 0.38 |
| Upper Lead Width | B1 | .045 | .058 | .070 | 1.14 | 1.46 | 1.78 |
| Lower Lead Width | B | .014 | .018 | .022 | 0.36 | 0.46 | 0.56 |
| Overall Row Spacing | § eB | .310 | .370 | .430 | 7.87 | 9.40 | 10.92 |
| Mold Draft Angle Top | α | 5 | 10 | 15 | 5 | 10 | 15 |
| Mold Draft Angle Bottom | β | 5 | 10 | 15 | 5 | 10 | 15 |

* Controlling Parameter

§ Significant Characteristic

Notes:

Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

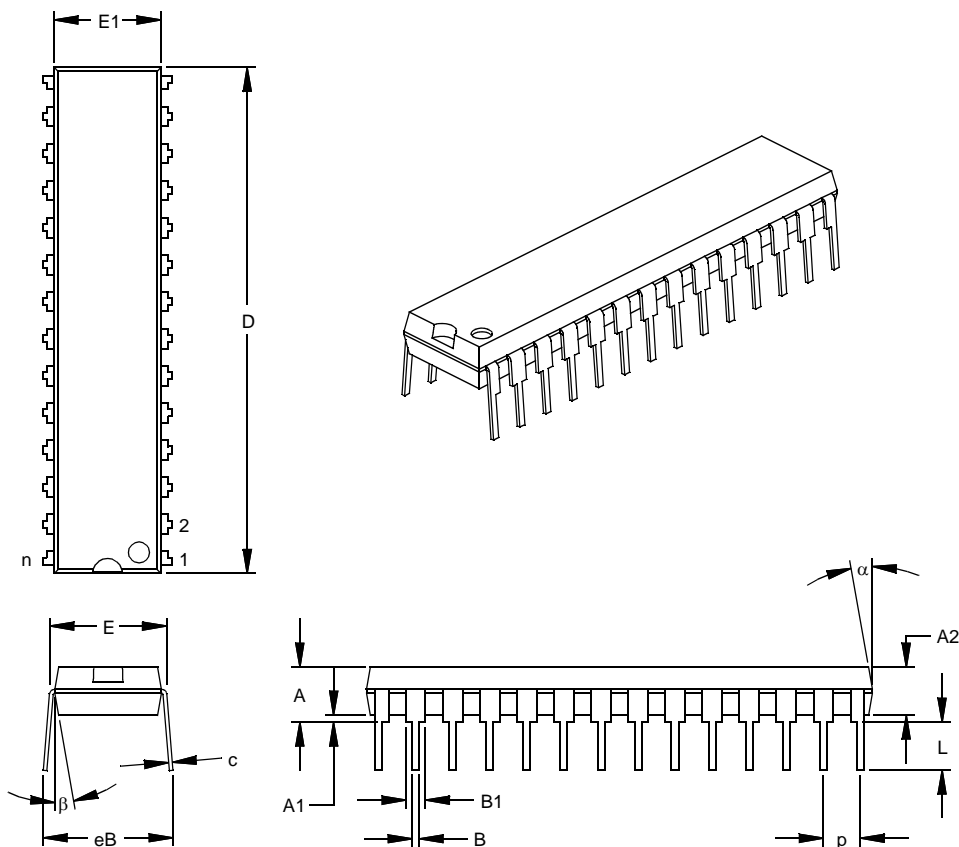
JEDEC Equivalent: MS-001

Drawing No. C04-007

PIC16C55X

28-Lead Skinny Plastic Dual In-line (SP) – 300 mil (PDIP)

Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



| Units | | INCHES* | | | MILLIMETERS | | |
|----------------------------|------|---------|-------|-------|-------------|-------|-------|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 28 | | | 28 | |
| Pitch | p | | .100 | | | 2.54 | |
| Top to Seating Plane | A | .140 | .150 | .160 | 3.56 | 3.81 | 4.06 |
| Molded Package Thickness | A2 | .125 | .130 | .135 | 3.18 | 3.30 | 3.43 |
| Base to Seating Plane | A1 | .015 | | | 0.38 | | |
| Shoulder to Shoulder Width | E | .300 | .310 | .325 | 7.62 | 7.87 | 8.26 |
| Molded Package Width | E1 | .275 | .285 | .295 | 6.99 | 7.24 | 7.49 |
| Overall Length | D | 1.345 | 1.365 | 1.385 | 34.16 | 34.67 | 35.18 |
| Tip to Seating Plane | L | .125 | .130 | .135 | 3.18 | 3.30 | 3.43 |
| Lead Thickness | c | .008 | .012 | .015 | 0.20 | 0.29 | 0.38 |
| Upper Lead Width | B1 | .040 | .053 | .065 | 1.02 | 1.33 | 1.65 |
| Lower Lead Width | B | .016 | .019 | .022 | 0.41 | 0.48 | 0.56 |
| Overall Row Spacing | § eB | .320 | .350 | .430 | 8.13 | 8.89 | 10.92 |
| Mold Draft Angle Top | α | 5 | 10 | 15 | 5 | 10 | 15 |
| Mold Draft Angle Bottom | β | 5 | 10 | 15 | 5 | 10 | 15 |

* Controlling Parameter

§ Significant Characteristic

Notes:

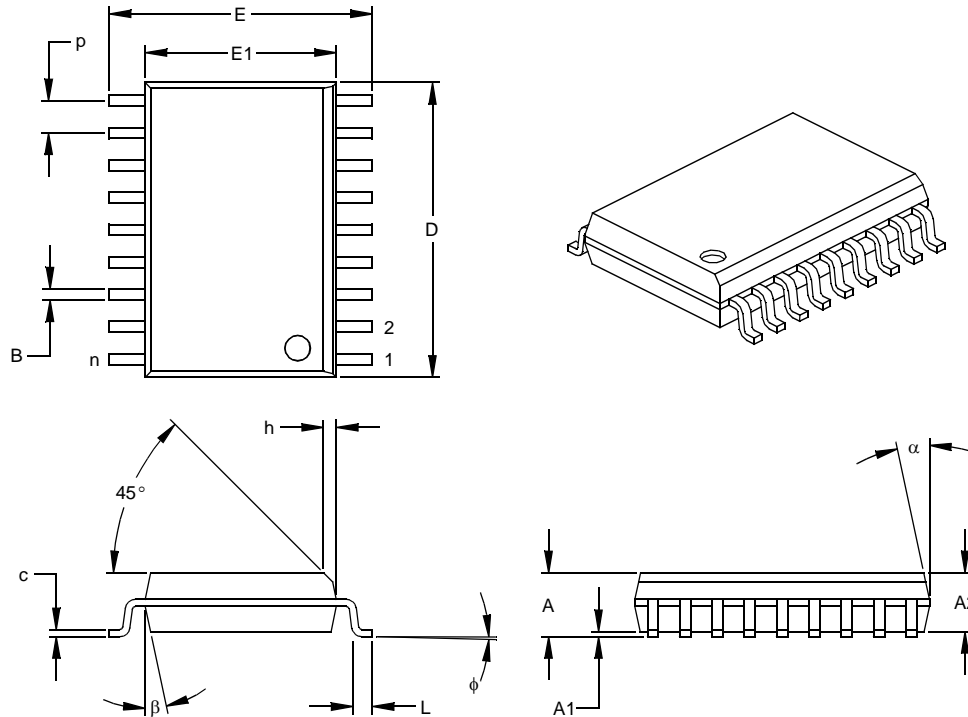
Dimension D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC Equivalent: MO-095

Drawing No. C04-070

18-Lead Plastic Small Outline (SO) – Wide, 300 mil (SOIC)

Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



| Units | | INCHES* | | | MILLIMETERS | | |
|--------------------------|----|---------|------|------|-------------|-------|-------|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 18 | | | 18 | |
| Pitch | P | | .050 | | | 1.27 | |
| Overall Height | A | .093 | .099 | .104 | 2.36 | 2.50 | 2.64 |
| Molded Package Thickness | A2 | .088 | .091 | .094 | 2.24 | 2.31 | 2.39 |
| Standoff § | A1 | .004 | .008 | .012 | 0.10 | 0.20 | 0.30 |
| Overall Width | E | .394 | .407 | .420 | 10.01 | 10.34 | 10.67 |
| Molded Package Width | E1 | .291 | .295 | .299 | 7.39 | 7.49 | 7.59 |
| Overall Length | D | .446 | .454 | .462 | 11.33 | 11.53 | 11.73 |
| Chamfer Distance | h | .010 | .020 | .029 | 0.25 | 0.50 | 0.74 |
| Foot Length | L | .016 | .033 | .050 | 0.41 | 0.84 | 1.27 |
| Foot Angle | φ | 0 | 4 | 8 | 0 | 4 | 8 |
| Lead Thickness | c | .009 | .011 | .012 | 0.23 | 0.27 | 0.30 |
| Lead Width | B | .014 | .017 | .020 | 0.36 | 0.42 | 0.51 |
| Mold Draft Angle Top | α | 0 | 12 | 15 | 0 | 12 | 15 |
| Mold Draft Angle Bottom | β | 0 | 12 | 15 | 0 | 12 | 15 |

* Controlling Parameter

§ Significant Characteristic

Notes:

Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

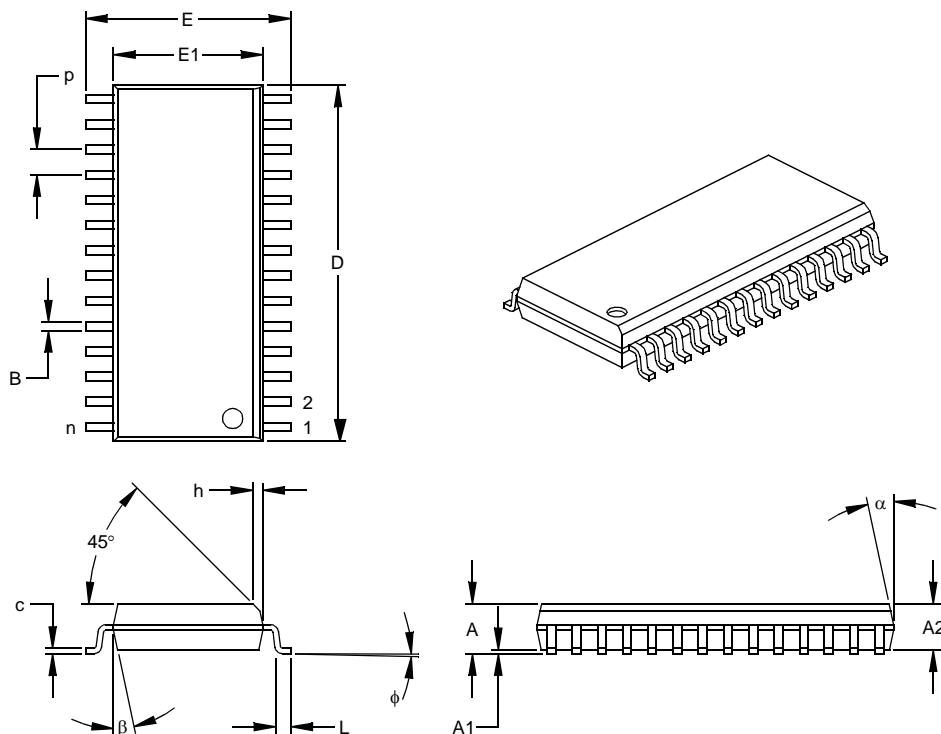
JEDEC Equivalent: MS-013

Drawing No. C04-051

PIC16C55X

28-Lead Plastic Small Outline (SO) – Wide, 300 mil (SOIC)

Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



| Units | | INCHES* | | | MILLIMETERS | | |
|--------------------------|----|---------|------|------|-------------|-------|-------|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 28 | | | 28 | |
| Pitch | p | | .050 | | | 1.27 | |
| Overall Height | A | .093 | .099 | .104 | 2.36 | 2.50 | 2.64 |
| Molded Package Thickness | A2 | .088 | .091 | .094 | 2.24 | 2.31 | 2.39 |
| Standoff § | A1 | .004 | .008 | .012 | 0.10 | 0.20 | 0.30 |
| Overall Width | E | .394 | .407 | .420 | 10.01 | 10.34 | 10.67 |
| Molded Package Width | E1 | .288 | .295 | .299 | 7.32 | 7.49 | 7.59 |
| Overall Length | D | .695 | .704 | .712 | 17.65 | 17.87 | 18.08 |
| Chamfer Distance | h | .010 | .020 | .029 | 0.25 | 0.50 | 0.74 |
| Foot Length | L | .016 | .033 | .050 | 0.41 | 0.84 | 1.27 |
| Foot Angle Top | φ | 0 | 4 | 8 | 0 | 4 | 8 |
| Lead Thickness | c | .009 | .011 | .013 | 0.23 | 0.28 | 0.33 |
| Lead Width | B | .014 | .017 | .020 | 0.36 | 0.42 | 0.51 |
| Mold Draft Angle Top | α | 0 | 12 | 15 | 0 | 12 | 15 |
| Mold Draft Angle Bottom | β | 0 | 12 | 15 | 0 | 12 | 15 |

* Controlling Parameter

§ Significant Characteristic

Notes:

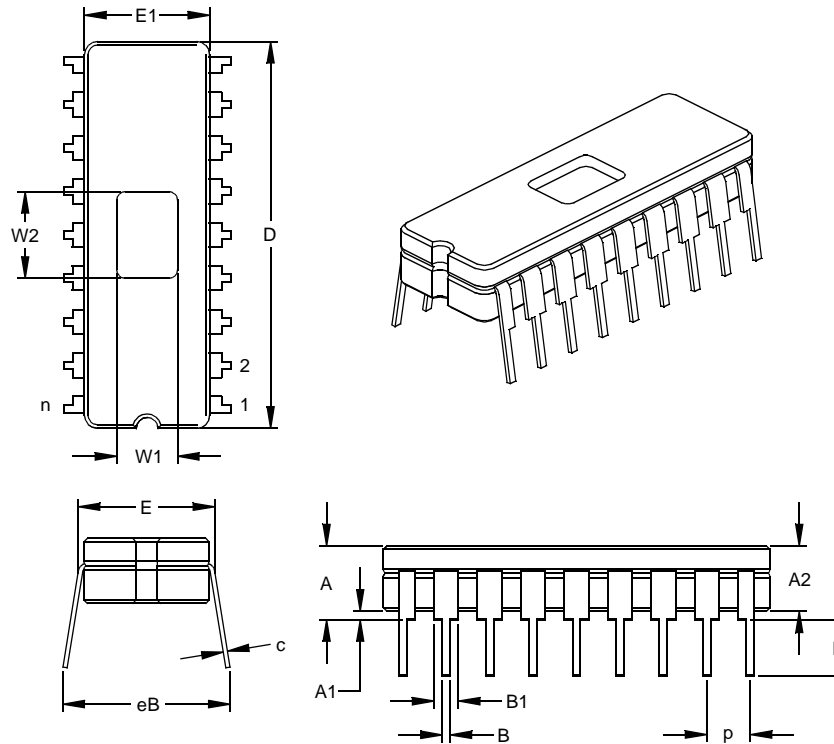
Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC Equivalent: MS-013

Drawing No. C04-052

18-Lead Ceramic Dual In-line with Window (JW) – 300 mil (CERDIP)

Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



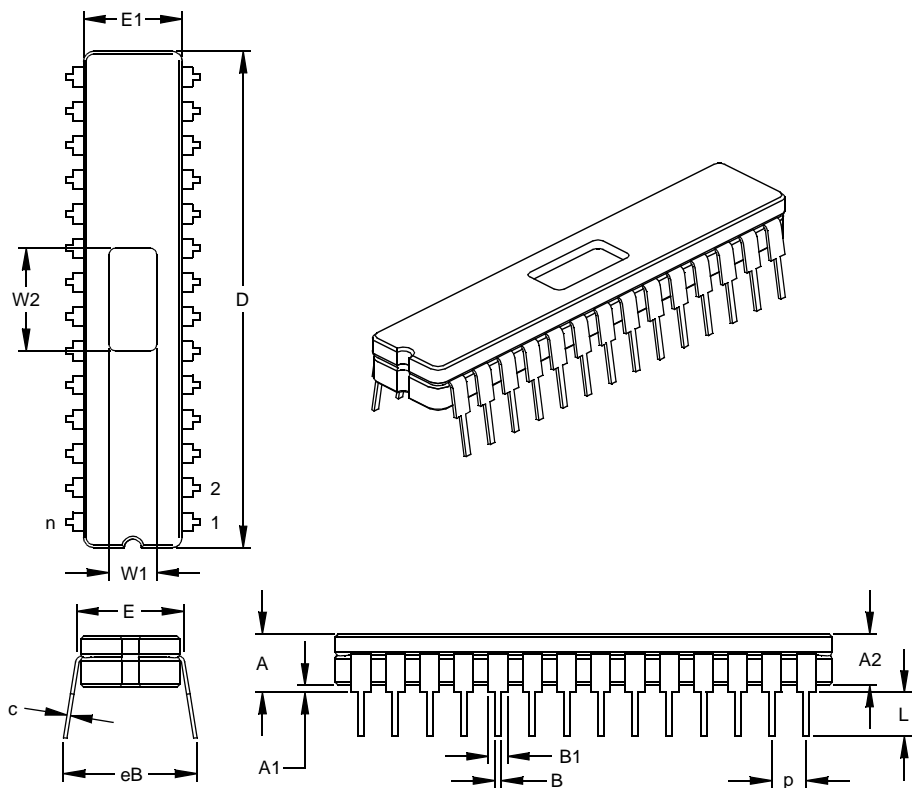
| Units | | INCHES* | | | MILLIMETERS | | |
|----------------------------|--------|---------|------|------|-------------|-------|-------|
| Dimension | Limits | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 18 | | | 18 | |
| Pitch | p | | .100 | | | 2.54 | |
| Top to Seating Plane | A | .170 | .183 | .195 | 4.32 | 4.64 | 4.95 |
| Ceramic Package Height | A2 | .155 | .160 | .165 | 3.94 | 4.06 | 4.19 |
| Standoff | A1 | .015 | .023 | .030 | 0.38 | 0.57 | 0.76 |
| Shoulder to Shoulder Width | E | .300 | .313 | .325 | 7.62 | 7.94 | 8.26 |
| Ceramic Pkg. Width | E1 | .285 | .290 | .295 | 7.24 | 7.37 | 7.49 |
| Overall Length | D | .880 | .900 | .920 | 22.35 | 22.86 | 23.37 |
| Tip to Seating Plane | L | .125 | .138 | .150 | 3.18 | 3.49 | 3.81 |
| Lead Thickness | c | .008 | .010 | .012 | 0.20 | 0.25 | 0.30 |
| Upper Lead Width | B1 | .050 | .055 | .060 | 1.27 | 1.40 | 1.52 |
| Lower Lead Width | B | .016 | .019 | .021 | 0.41 | 0.47 | 0.53 |
| Overall Row Spacing | § eB | .345 | .385 | .425 | 8.76 | 9.78 | 10.80 |
| Window Width | W1 | .130 | .140 | .150 | 3.30 | 3.56 | 3.81 |
| Window Length | W2 | .190 | .200 | .210 | 4.83 | 5.08 | 5.33 |

* Controlling Parameter
 § Significant Characteristic
 JEDEC Equivalent: MO-036
 Drawing No. C04-010

PIC16C55X

28-Lead Ceramic Dual In-line with Window (JW) – 300 mil (CERDIP)

Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



| Units | | INCHES* | | | MILLIMETERS | | |
|----------------------------|------|---------|-------|-------|-------------|-------|-------|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 28 | | | 28 | |
| Pitch | p | | .100 | | | 2.54 | |
| Top to Seating Plane | A | .170 | .183 | .195 | 4.32 | 4.64 | 4.95 |
| Ceramic Package Height | A2 | .155 | .160 | .165 | 3.94 | 4.06 | 4.19 |
| Standoff | A1 | .015 | .023 | .030 | 0.38 | 0.57 | 0.76 |
| Shoulder to Shoulder Width | E | .300 | .313 | .325 | 7.62 | 7.94 | 8.26 |
| Ceramic Pkg. Width | E1 | .285 | .290 | .295 | 7.24 | 7.37 | 7.49 |
| Overall Length | D | 1.430 | 1.458 | 1.485 | 36.32 | 37.02 | 37.72 |
| Tip to Seating Plane | L | .135 | .140 | .145 | 3.43 | 3.56 | 3.68 |
| Lead Thickness | c | .008 | .010 | .012 | 0.20 | 0.25 | 0.30 |
| Upper Lead Width | B1 | .050 | .058 | .065 | 1.27 | 1.46 | 1.65 |
| Lower Lead Width | B | .016 | .019 | .021 | 0.41 | 0.47 | 0.53 |
| Overall Row Spacing | § eB | .345 | .385 | .425 | 8.76 | 9.78 | 10.80 |
| Window Width | W1 | .130 | .140 | .150 | 3.30 | 3.56 | 3.81 |
| Window Length | W2 | .290 | .300 | .310 | 7.37 | 7.62 | 7.87 |

* Controlling Parameter

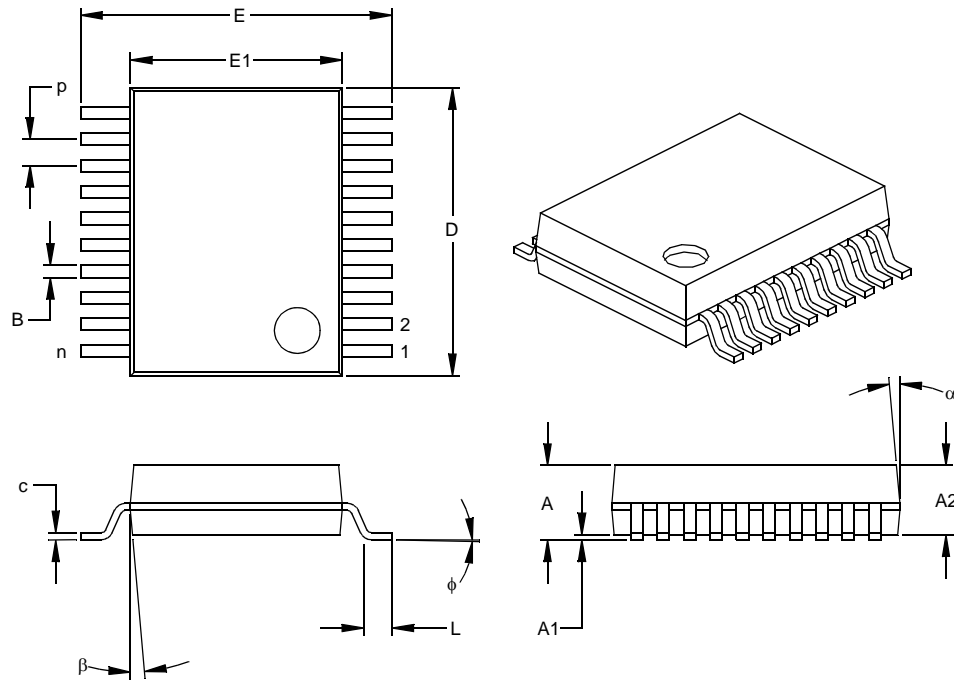
§ Significant Characteristic

JEDEC Equivalent: MO-058

Drawing No. C04-080

20-Lead Plastic Shrink Small Outline (SS) – 209 mil, 5.30 mm (SSOP)

Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



| Units | | INCHES* | | | MILLIMETERS | | |
|--------------------------|----|---------|------|------|-------------|--------|--------|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 20 | | | 20 | |
| Pitch | p | | .026 | | | 0.65 | |
| Overall Height | A | .068 | .073 | .078 | 1.73 | 1.85 | 1.98 |
| Molded Package Thickness | A2 | .064 | .068 | .072 | 1.63 | 1.73 | 1.83 |
| Standoff § | A1 | .002 | .006 | .010 | 0.05 | 0.15 | 0.25 |
| Overall Width | E | .299 | .309 | .322 | 7.59 | 7.85 | 8.18 |
| Molded Package Width | E1 | .201 | .207 | .212 | 5.11 | 5.25 | 5.38 |
| Overall Length | D | .278 | .284 | .289 | 7.06 | 7.20 | 7.34 |
| Foot Length | L | .022 | .030 | .037 | 0.56 | 0.75 | 0.94 |
| Lead Thickness | c | .004 | .007 | .010 | 0.10 | 0.18 | 0.25 |
| Foot Angle | φ | 0 | 4 | 8 | 0.00 | 101.60 | 203.20 |
| Lead Width | B | .010 | .013 | .015 | 0.25 | 0.32 | 0.38 |
| Mold Draft Angle Top | α | 0 | 5 | 10 | 0 | 5 | 10 |
| Mold Draft Angle Bottom | β | 0 | 5 | 10 | 0 | 5 | 10 |

* Controlling Parameter

§ Significant Characteristic

Notes:

Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

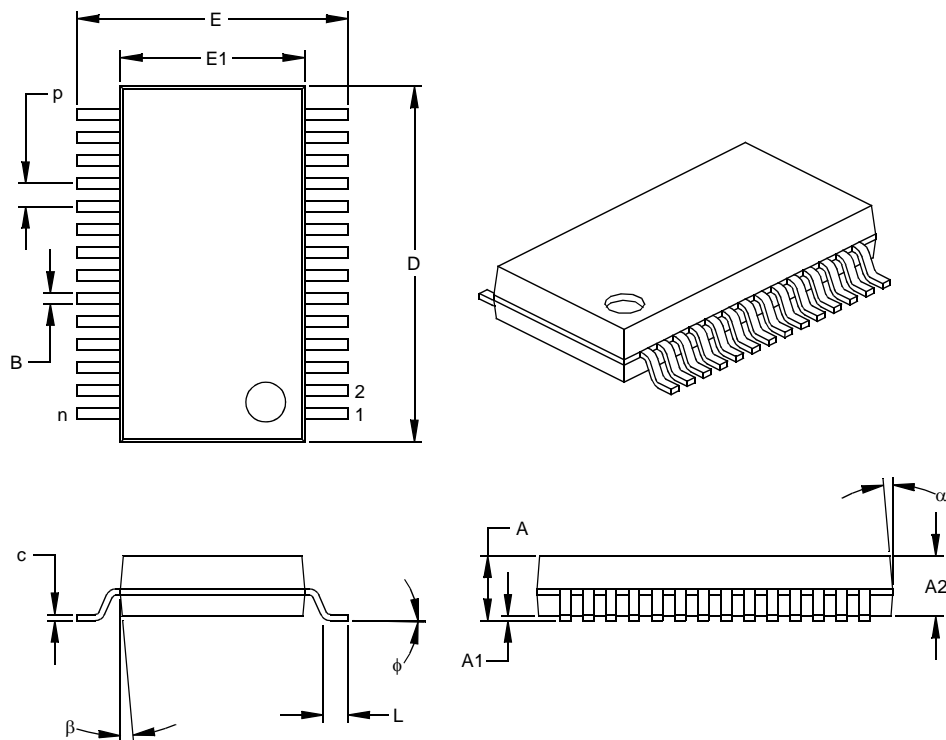
JEDEC Equivalent: MO-150

Drawing No. C04-072

PIC16C55X

28-Lead Plastic Shrink Small Outline (SS) – 209 mil, 5.30 mm (SSOP)

Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



| Units | | INCHES | | | MILLIMETERS* | | |
|--------------------------|----|--------|------|------|--------------|--------|--------|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 28 | | | 28 | |
| Pitch | p | | .026 | | | 0.65 | |
| Overall Height | A | .068 | .073 | .078 | 1.73 | 1.85 | 1.98 |
| Molded Package Thickness | A2 | .064 | .068 | .072 | 1.63 | 1.73 | 1.83 |
| Standoff § | A1 | .002 | .006 | .010 | 0.05 | 0.15 | 0.25 |
| Overall Width | E | .299 | .309 | .319 | 7.59 | 7.85 | 8.10 |
| Molded Package Width | E1 | .201 | .207 | .212 | 5.11 | 5.25 | 5.38 |
| Overall Length | D | .396 | .402 | .407 | 10.06 | 10.20 | 10.34 |
| Foot Length | L | .022 | .030 | .037 | 0.56 | 0.75 | 0.94 |
| Lead Thickness | c | .004 | .007 | .010 | 0.10 | 0.18 | 0.25 |
| Foot Angle | φ | 0 | 4 | 8 | 0.00 | 101.60 | 203.20 |
| Lead Width | B | .010 | .013 | .015 | 0.25 | 0.32 | 0.38 |
| Mold Draft Angle Top | α | 0 | 5 | 10 | 0 | 5 | 10 |
| Mold Draft Angle Bottom | β | 0 | 5 | 10 | 0 | 5 | 10 |

* Controlling Parameter

§ Significant Characteristic

Notes:

Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC Equivalent: MS-150

Drawing No. C04-073

APPENDIX A: ENHANCEMENTS

The following are the list of enhancements over the PIC16C5X microcontroller family:

1. Instruction word length is increased to 14 bits. This allows larger page sizes both in program memory (4K now as opposed to 512 before) and register file (up to 128 bytes now versus 32 bytes before).
2. A PC high latch register (PCLATH) is added to handle program memory paging. PA2, PA1, PA0 bits are removed from STATUS register.
3. Data memory paging is slightly redefined. STATUS register is modified.
4. Four new instructions have been added: RETURN, RETFIE, ADDLW, and SUBLW. Two instructions TRIS and OPTION are being phased out although they are kept for compatibility with PIC16C5X.
5. OPTION and TRIS registers are made addressable.
6. Interrupt capability is added. Interrupt vector is at 0004h.
7. Stack size is increased to 8 deep.
8. RESET vector is changed to 0000h.
9. RESET of all registers is revised. Three different RESET (and wake-up) types are recognized. Registers are reset differently.
10. Wake-up from SLEEP through interrupt is added.
11. Two separate timers, Oscillator Start-up Timer (OST) and Power-up Timer (PWRT) are included for more reliable power-up. These timers are invoked selectively to avoid unnecessary delays on power-up and wake-up.
12. PORTB has weak pull-ups and interrupt-on-change feature.
13. Timer0 clock input, T0CKI pin is also a port pin (RA4/T0CKI) and has a TRIS bit.
14. FSR is made a full 8-bit register.
15. "In-circuit programming" is made possible. The user can program PIC16C55X devices using only five pins: VDD, VSS, VPP, RB6 (clock) and RB7 (data in/out).
16. PCON status register is added with a Power-on Reset (POR) status bit.
17. Code protection scheme is enhanced such that portions of the program memory can be protected, while the remainder is unprotected.
18. PORTA inputs are now Schmitt Trigger inputs.

APPENDIX B: COMPATIBILITY

To convert code written for PIC16C5X to PIC16C55X, 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.

APPENDIX C: REVISION HISTORY

Revision E (January 2013)

Added a note to each package outline drawing.

PIC16C55X

NOTES:

INDEX

A

| | |
|------------------------------|----|
| ADDLW Instruction | 55 |
| ADDWF Instruction | 55 |
| ANDLW Instruction | 55 |
| ANDWF Instruction | 55 |
| Architectural Overview | 9 |
| Assembler | |
| MPASM Assembler | 67 |

B

| | |
|--------------------------|----|
| BCF Instruction | 56 |
| Block Diagram | |
| TIMER0 | 47 |
| TMR0/WDT PRESCALER | 50 |
| BSF Instruction | 56 |
| BTFSC Instruction | 56 |
| BTFSS Instruction | 57 |

C

| | |
|---|----|
| CALL Instruction | 57 |
| Clocking Scheme/Instruction Cycle | 12 |
| CLRF Instruction | 57 |
| CLRW Instruction | 58 |
| CLRWDW Instruction | 58 |
| Code Protection | 46 |
| COMF Instruction | 58 |
| Configuration Bits | 31 |

D

| | |
|--------------------------------|----|
| Data Memory Organization | 13 |
| DECF Instruction | 58 |
| DECFSZ Instruction | 59 |
| Development Support | 67 |

E

| | |
|---|----|
| Errata | 3 |
| External Crystal Oscillator Circuit | 34 |

G

| | |
|-------------------------------------|----|
| General purpose Register File | 13 |
| GOTO Instruction | 59 |

I

| | |
|---|----|
| I/O Ports | 23 |
| I/O Programming Considerations | 28 |
| ICEPIC In-Circuit Emulator | 68 |
| ID Locations | 46 |
| INCF Instruction | 59 |
| INCFSSZ Instruction | 60 |
| In-Circuit Serial Programming | 46 |
| Indirect Addressing, INDF and FSR Registers | 21 |
| Instruction Flow/Pipelining | 12 |
| Instruction Set | |
| ADDLW | 55 |
| ADDWF | 55 |
| ANDLW | 55 |
| ANDWF | 55 |
| BCF | 56 |
| BSF | 56 |
| BTFSC | 56 |
| BTFSS | 57 |
| CALL | 57 |
| CLRF | 57 |

| | |
|-------------------------------|----|
| CLRW | 58 |
| CLRWDW | 58 |
| COMF | 58 |
| DECF | 58 |
| DECFSZ | 59 |
| GOTO | 59 |
| INCF | 59 |
| INCFSSZ | 60 |
| IORLW | 60 |
| IORWF | 60 |
| MOVF | 61 |
| MOVLW | 60 |
| MOVWF | 61 |
| NOP | 61 |
| OPTION | 61 |
| RETFIE | 62 |
| RETLW | 62 |
| RETURN | 62 |
| RLF | 62 |
| RRF | 63 |
| SLEEP | 63 |
| SUBLW | 63 |
| SUBWF | 64 |
| SWAPF | 64 |
| TRIS | 64 |
| XORLW | 65 |
| XORWF | 65 |
| Instruction Set Summary | 53 |
| INT Interrupt | 42 |
| INTCON Register | 19 |
| Interrupts | 41 |
| IORLW Instruction | 60 |
| IORWF Instruction | 60 |

K

| | |
|---|----|
| KEELOQ Evaluation and Programming Tools | 70 |
|---|----|

M

| | |
|---|----|
| MOVF Instruction | 61 |
| MOVLW Instruction | 60 |
| MOVWF Instruction | 61 |
| MPLAB C17 and MPLAB C18 C Compilers | 67 |
| MPLAB ICD In-Circuit Debugger | 69 |
| MPLAB ICE High Performance Universal In-Circuit Emulator with MPLAB IDE | 68 |
| MPLAB Integrated Development Environment Software | 67 |
| MPLINK Object Linker/MPLIB Object Librarian | 68 |

N

| | |
|-----------------------|----|
| NOP Instruction | 61 |
|-----------------------|----|

O

| | |
|---|----|
| One-Time-Programmable (OTP) Devices | 7 |
| OPTION Instruction | 61 |
| OPTION Register | 18 |
| Oscillator Configurations | 33 |
| Oscillator Start-up Timer (OST) | 36 |

P

| | |
|---|----|
| PCL and PCLATH | 21 |
| PCON Register | 20 |
| PICDEM 1 Low Cost PIC MCU Demonstration Board | 69 |
| PICDEM 17 Demonstration Board | 70 |
| PICDEM 2 Low Cost PIC16CXX Demonstration Board | 69 |
| PICDEM 3 Low Cost PIC16CXXX Demonstration Board | 70 |

PIC16C55X

| | |
|---|--------|
| PICSTART Plus Entry Level Development Programmer | 69 |
| Port RB Interrupt | 42 |
| PORTA | 23 |
| PORTB | 25, 27 |
| Power Control/Status Register (PCON) | 37 |
| Power-Down Mode (SLEEP) | 45 |
| Power-On Reset (POR) | 36 |
| Power-up Timer (PWRT) | 36 |
| Prescaler | 49 |
| PRO MATE II Universal Device Programmer | 69 |
| Program Memory Organization | 13 |

Q

| | |
|---|---|
| Quick-Turnaround-Production (QTP) Devices | 7 |
|---|---|

R

| | |
|--------------------------|----|
| RC Oscillator | 34 |
| Reset | 35 |
| RETFIE Instruction | 62 |
| RETLW Instruction | 62 |
| RETURN Instruction | 62 |
| RLF Instruction | 62 |
| RRF Instruction | 63 |

S

| | |
|---|----|
| Serialized Quick-Turnaround-Production (SQTP) Devices ... | 7 |
| SLEEP Instruction | 63 |
| Software Simulator (MPLAB SIM) | 68 |
| Special Features of the CPU | 31 |
| Special Function Registers | 15 |
| Stack | 21 |
| Status Register | 17 |
| SUBLW Instruction | 63 |
| SUBWF Instruction | 64 |
| SWAPF Instruction | 64 |

T

| | |
|--|--------|
| Timer0 | |
| TIMER0 | 47 |
| TIMER0 (TMR0) Interrupt | 47 |
| TIMER0 (TMR0) Module | 47 |
| TMR0 with External Clock | 49 |
| Timer1 | |
| Switching Prescaler Assignment | 51 |
| Timing Diagrams and Specifications | 81 |
| TMR0 Interrupt | 42 |
| TRIS Instruction | 64 |
| TRISA | 23 |
| TRISB | 25, 27 |

W

| | |
|----------------------------|----|
| Watchdog Timer (WDT) | 43 |
| WWW, On-Line Support | 3 |

X

| | |
|-------------------------|----|
| XORLW Instruction | 65 |
| XORWF Instruction | 65 |

THE MICROCHIP WEB SITE

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

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

CUSTOMER CHANGE NOTIFICATION SERVICE

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

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

CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

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

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

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

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 (480) 792-4150.

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

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:

Literature Number: DS40143E

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 document easy to follow? If not, why?

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

5. What deletions from the document 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?

PRODUCT IDENTIFICATION SYSTEM

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

| <u>PART NO.</u> | <u>X</u> | <u>/XX</u> | <u>XXX</u> |
|-------------------|--|------------|------------|
| Device | Temperature Range | Package | Pattern |
| Device | PIC17C756: Standard VDD range PIC17C756T: (Tape and Reel) PIC17LC756: Extended VDD range | | |
| Temperature Range | - = 0°C to +70°C I = -40°C to +85°C | | |
| Package | CL = Windowed LCC PT = TQFP L = PLCC | | |
| Pattern | QTP, SQTP, ROM Code (factory specified) or Special Requirements. Blank for OTP and Windowed devices. | | |

Examples:

- PIC17C756-16L Commercial Temp., PLCC package, 16 MHz, normal VDD limits
- PIC17LC756-08/PT Commercial Temp., TQFP package, 8MHz, extended VDD limits
- PIC17C756-33I/PT Industrial Temp., TQFP package, 33 MHz, normal VDD limits

* JW Devices are UV erasable and can be programmed to any device configuration. JW Devices meet the electrical requirement of each oscillator type.

Sales and Support

Data Sheets

Products supported by a preliminary Data Sheet may have an errata sheet describing minor operational differences and recommended workarounds. To determine if an errata sheet exists for a particular device, please contact one of the following:

- Your local Microchip sales office
- The Microchip Worldwide Site (www.microchip.com)

PIC16C55X

NOTES:

NOTES:

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights.

Trademarks

The Microchip name and logo, the Microchip logo, dsPIC, FlashFlex, KEELOQ, KEELOQ logo, MPLAB, PIC, PICmicro, PICSTART, PIC³² logo, rPIC, SST, SST Logo, SuperFlash and UNI/O are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

FilterLab, Hampshire, HI-TECH C, Linear Active Thermistor, MTP, SEEVAL and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Silicon Storage Technology is a registered trademark of Microchip Technology Inc. in other countries.

Analog-for-the-Digital Age, Application Maestro, BodyCom, chipKIT, chipKIT logo, CodeGuard, dsPICDEM, dsPICDEM.net, dsPICworks, dsSPEAK, ECAN, ECONOMONITOR, FanSense, HI-TIDE, In-Circuit Serial Programming, ICSP, Mindi, MiWi, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, mTouch, Omniscent Code Generation, PICC, PICC-18, PICDEM, PICDEM.net, PICkit, PICtail, REAL ICE, rLAB, Select Mode, SQL, Serial Quad I/O, Total Endurance, TSHARC, UniWinDriver, WiperLock, ZENA and Z-Scale are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

GestIC and ULPP are registered trademarks of Microchip Technology Germany II GmbH & Co. & KG, a subsidiary of Microchip Technology Inc., in other countries.

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

© 1996-2013, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.



Printed on recycled paper.

ISBN: 9781620769737

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

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



Worldwide Sales and Service

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
<http://www.microchip.com/support>
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Cleveland
Independence, OH
Tel: 216-447-0464
Fax: 216-447-0643

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Farmington Hills, MI
Tel: 248-538-2250
Fax: 248-538-2260

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

Santa Clara
Santa Clara, CA
Tel: 408-961-6444
Fax: 408-961-6445

Toronto
Mississauga, Ontario,
Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8569-7000
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

China - Chongqing
Tel: 86-23-8980-9588
Fax: 86-23-8980-9500

China - Hangzhou
Tel: 86-571-2819-3187
Fax: 86-571-2819-3189

China - Hong Kong SAR
Tel: 852-2943-5100
Fax: 852-2401-3431

China - Nanjing
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

China - Qingdao
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

China - Shanghai
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

China - Shenyang
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

China - Shenzhen
Tel: 86-755-8864-2200
Fax: 86-755-8203-1760

China - Wuhan
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

China - Xian
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

China - Xiamen
Tel: 86-592-2388138
Fax: 86-592-2388130

China - Zhuhai
Tel: 86-756-3210040
Fax: 86-756-3210049

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

India - Pune
Tel: 91-20-2566-1512
Fax: 91-20-2566-1513

Japan - Osaka
Tel: 81-6-6152-7160
Fax: 81-6-6152-9310

Japan - Tokyo
Tel: 81-3-6880-3770
Fax: 81-3-6880-3771

Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Malaysia - Kuala Lumpur
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

Malaysia - Penang
Tel: 60-4-227-8870
Fax: 60-4-227-4068

Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Hsin Chu
Tel: 886-3-5778-366
Fax: 886-3-5770-955

Taiwan - Kaohsiung
Tel: 886-7-213-7828
Fax: 886-7-330-9305

Taiwan - Taipei
Tel: 886-2-2508-8600
Fax: 886-2-2508-0102

Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

UK - Wokingham
Tel: 44-118-921-5869
Fax: 44-118-921-5820

11/29/12



**Стандарт
Электрон
Связь**

Мы молодая и активно развивающаяся компания в области поставок электронных компонентов. Мы поставляем электронные компоненты отечественного и импортного производства напрямую от производителей и с крупнейших складов мира.

Благодаря сотрудничеству с мировыми поставщиками мы осуществляем комплексные и плановые поставки широчайшего спектра электронных компонентов.

Собственная эффективная логистика и склад в обеспечивает надежную поставку продукции в точно указанные сроки по всей России.

Мы осуществляем техническую поддержку нашим клиентам и предпродажную проверку качества продукции. На все поставляемые продукты мы предоставляем гарантию .

Осуществляем поставки продукции под контролем ВП МО РФ на предприятия военно-промышленного комплекса России , а также работаем в рамках 275 ФЗ с открытием отдельных счетов в уполномоченном банке. Система менеджмента качества компании соответствует требованиям ГОСТ ISO 9001.

Минимальные сроки поставки, гибкие цены, неограниченный ассортимент и индивидуальный подход к клиентам являются основой для выстраивания долгосрочного и эффективного сотрудничества с предприятиями радиоэлектронной промышленности, предприятиями ВПК и научно-исследовательскими институтами России.

С нами вы становитесь еще успешнее!

Наши контакты:

Телефон: +7 812 627 14 35

Электронная почта: sales@st-electron.ru

Адрес: 198099, Санкт-Петербург,
Промышленная ул, дом № 19, литера Н,
помещение 100-Н Офис 331