# PIC16C433
# Data Sheet

## 8-Bit CMOS Microcontroller with LIN bus Transceiver

**Trademarks**



DNV Certification, Inc. USA

ANSI • RAB QMS ACCREDITED

DNV

DNV MSC The Netherlands Accredited by the RvA

ISO 9001 / QS-9000 REGISTERED FIRM

*Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona in July 1999. The Company's quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, KEELOQ® code hopping devices, Serial EEPROMs and microperipheral products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001 certified.*

# PIC16C433

## 8-Bit CMOS Microcontroller with LIN bus Transceiver

### Devices Included in this Data Sheet:

- PIC16C433

### High-Performance RISC CPU:

- Only 35 single word instructions to learn
- All instructions are single cycle (400 ns) except for program branches which are two-cycle
- Operating speed: DC - 10 MHz clock input
  DC - 400 ns instruction cycle

| Device | Memory | |
|---|---|---|
| | Program | Data RAM |
| PIC16C433 | 2048 x 14 | 128 x 8 |

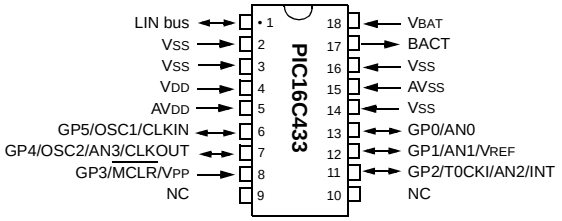
- 14-bit wide instructions
- 8-bit wide data path
- Interrupt capability
- Special function hardware registers
- 8-level deep hardware stack
- Direct, indirect and relative addressing modes for data and instructions

### Peripheral Features:

- Integrated LIN bus Transceiver
- Wake-up on bus activity
- 12V battery operation for Transceiver
- Thermal Shut-down for Transceiver
- Ground Loss Protection
- Four-channel, 8-bit A/D converter
- 8-bit real time clock/counter (TMR0) with 8-bit programmable prescaler

### PIN DIAGRAM

**PDIP, SOIC, Windowed CERDIP**




### Special Microcontroller Features:

- In-Circuit Serial Programming (ICSP™)
- Internal 4 MHz oscillator with programmable calibration
- Selectable clockout
- 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
- Interrupt-on-pin change (GP0, GP1, GP3)
- Internal pull-ups on I/O pins (GP0, GP1, GP3)
- Internal pull-up on $\overline{\text{MCLR}}$ pin
- Selectable oscillator options:
  - INTRC: Precision internal 4 MHz oscillator
  - EXTRC: External low-cost RC oscillator
  - XT: Standard crystal/resonator
  - HS: High speed crystal/resonator
  - LP: Power saving, low frequency crystal

### CMOS Technology:

- Low power, high speed CMOS EPROM/ HV-CMOS technology
- Fully static design
- Operating voltage range 4.5V to 5.5V
- Industrial and Extended temperature ranges
- Low power consumption
  < 2 mA @ 5V, 4 MHz
  < 1 µA typical standby current

## Table of Contents



## 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.

## 1.0 GENERAL DESCRIPTION

The PIC16C433 device is a low cost, high performance, CMOS, fully static, 8-bit microcontroller with integrated analog-to-digital (A/D) converter and an integrated LIN bus Transceiver.

The LIN physical layer is implemented in hardware with a voltage range from 0V to 18V, with a 40V transient capability. The LIN protocol is to be implemented in firmware, which enables flexibility with future revisions of the LIN protocol.

All PICmicro® microcontrollers employ an advanced RISC architecture. The PIC16C433 microcontroller has 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.

PIC16C433 microcontroller typically achieve a 2:1 code compression and a 4:1 speed improvement over other 8-bit microcontrollers in their class.

The PIC16C433 device has 128 bytes of RAM, 5 I/O pins and 1 input pin. In addition, a timer/counter is available. Also a 4-channel, high-speed, 8-bit A/D is provided. The 8-bit resolution is ideally suited for applications requiring low cost analog interface (i.e., thermostat control, pressure sensing, etc.)

The PIC16C433 device has special features to reduce external components, thus reducing cost, enhancing system reliability and reducing power consumption. The Power-on Reset (POR), Power-up Timer (PWRT), and Oscillator Start-up Timer (OST) eliminate the need for external RESET circuitry. There are five oscillator configurations to choose from, including INTRC precision internal oscillator mode and the power saving LP (Low Power) oscillator mode. Power saving SLEEP mode, Watchdog Timer and code protection features improve system cost, power and reliability. The SLEEP (power-down) feature provides a power saving mode. The user can wake-up the chip from SLEEP through several external and internal interrupts and RESETS.

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

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

### 1.1 Applications

The PIC16C433 microcontroller fits well in applications ranging from personal care appliances and security systems to low power remote transmitters/receivers. The EPROM technology makes customizing application programs (transmitter codes, appliance settings, receiver frequencies, etc.) extremely fast. The small footprint packages, for through hole or surface mounting, make this microcontroller series perfect for applications with space limitations. Low cost, low power, high performance, ease of use and I/O flexibility make the PIC16C433 series very versatile even in areas where no microcontroller use has been considered before (i.e., timer functions, replacement of "glue" logic and PLD's in larger systems, coprocessor applications).

### 1.2 Family and Upward Compatibility

The PIC16C433 product is code compatible with other members of the 14-bit PIC16CXXX families.

### 1.3 Development Support

The PIC16C433 device is supported by a full featured macro assembler, a software simulator, an in-circuit emulator, a low cost development programmer and a full featured programmer. A "C" compiler and fuzzy logic support tools are also available.

**NOTES:**

## 2.0 PIC16C433 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 PIC16C433 Product Identification System section at the end of this data sheet. When placing orders, please use that page of the data sheet to specify the correct part number.

For example, the PIC16C433 device “type” is indicated in the device number:

1. **C**, as in PIC16**C**433. These devices have EPROM type memory and operate over the standard voltage range.
2. **LC**, as in PIC16**LC**433. These devices have EPROM type memory and operate over an extended voltage range.

### 2.1 UV Erasable Devices

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

The UV erasable version can be erased and reprogrammed to any of the configuration modes. Microchip's PRO MATE® II device programmer supports the PIC16C433. Third party programmers also are available; refer to the Microchip Third Party Guide for a list of sources.

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

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

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

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

### 2.3 Quick-Turn-Programming (QTP) Devices

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

### 2.4 Serialized Quick-Turn Programming (SQTP℠) 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.

**NOTES:**

## 3.0 ARCHITECTURAL OVERVIEW

The high performance of the PIC16C433 family can be attributed to a number of architectural features commonly found in RISC microprocessors. To begin with, the PIC16C433 uses a Harvard architecture, in which program and data are accessed from separate memories using separate buses. This improves bandwidth over traditional von Neumann architecture in which program and data are fetched from the same memory using the same bus. Separating program and data buses also allow instructions to be sized differently than the 8-bit wide data word. Instruction opcodes are 14-bits wide making it possible to have all single word instructions. A 14-bit wide program memory access bus fetches a 14-bit instruction in a single instruction cycle. A two-stage pipeline overlaps fetch and execution of instructions (Example 3-1). Consequently, all instructions (35) execute in a single cycle (400 ns @ 10 MHz) except for program branches.

The PIC16C433 can directly or indirectly address its register files or data memory. All special function registers, including the program counter, are mapped in the data memory. The PIC16C433 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 PIC16C433 simple, yet efficient. In addition, the learning curve is reduced significantly.

PIC16C433 devices contain an 8-bit ALU and working register. The ALU is a general purpose arithmetic unit. It performs arithmetic and Boolean functions between the data in the working register and any register file.

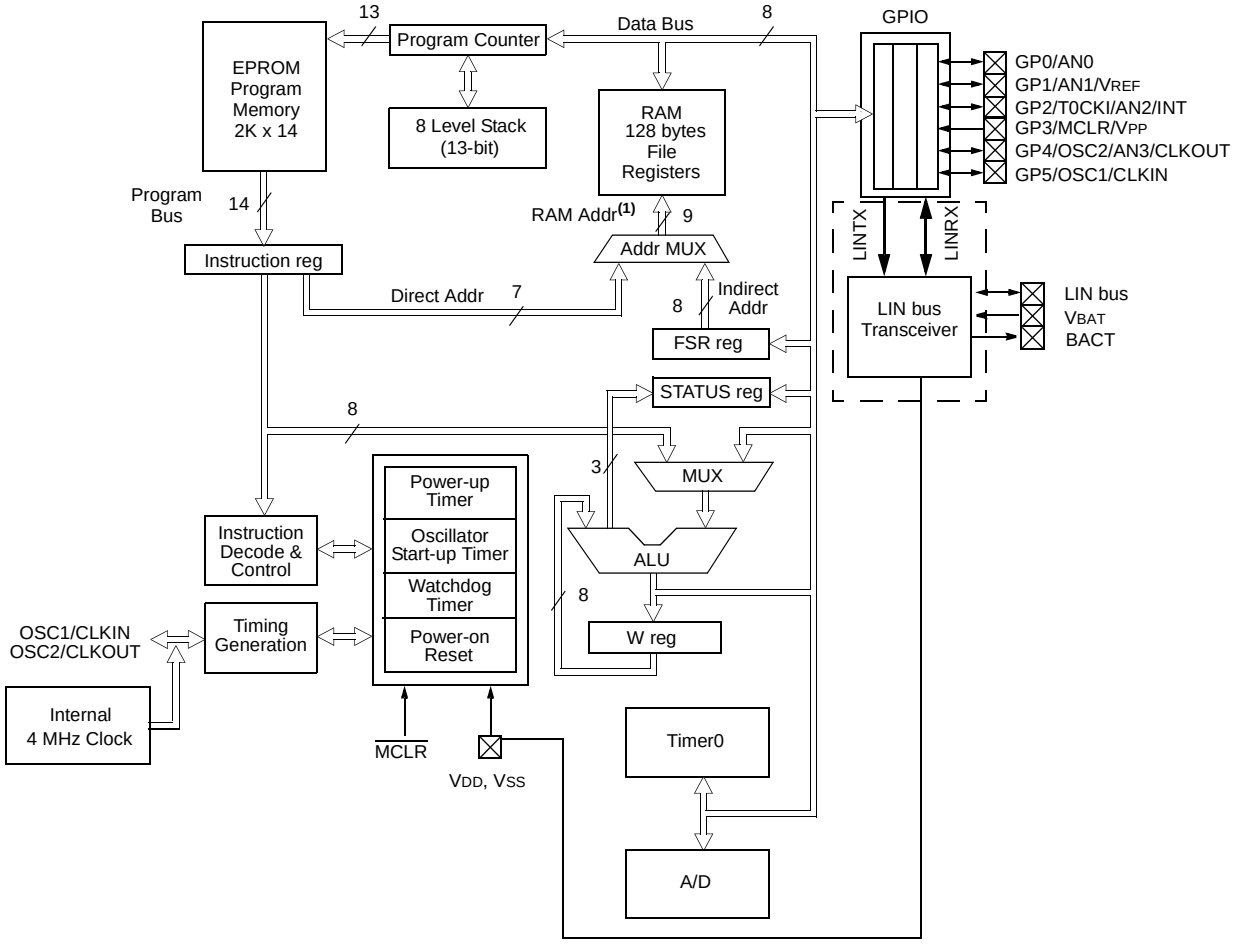
The ALU is 8-bits wide and capable of addition, subtraction, shift and logical operations. Unless otherwise mentioned, arithmetic operations are two's complement in nature. In two-operand instructions, typically one operand is the working register (W register). The other operand is a file register, or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

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

Depending on the instruction executed, the ALU may affect the values of the Carry (C), Digit Carry (DC), and Zero (Z) bits in the STATUS register. The C and DC bits operate as a $\overline{\text{borrow}}$ bit and a digit $\overline{\text{borrow}}$ out bit, respectively, in subtraction. See the `SUBLW` and `SUBWF` instructions for examples.

**FIGURE 3-1: PIC16C433 BLOCK DIAGRAM**

| Device | Program Memory | Data Memory (RAM) |
|---|---|---|
| PIC16C433 | 2K x 14 | 128 x 8 |




**Note 1:** Higher order bits are from the STATUS Register.

**TABLE 3-1: PIC16C433 PINOUT DESCRIPTION**

| Name | DIP Pin # | I/O/P Type | Buffer Type | Description |
|---|---|---|---|---|
| GP0/AN0 | 13 | I/O | TTL/ST | Bi-directional I/O port/serial programming data/analog input 0. Can be software programmed for internal weak pull-up and interrupt-on-pin change. This buffer is a Schmitt Trigger input when used in Serial Programming mode. |
| GP1/AN1/$V_{REF}$ | 12 | I/O | TTL/ST | Bi-directional I/O port/serial programming clock/analog input 1/ voltage reference. Can be software programmed for internal weak pull-up and interrupt-on-pin change. This buffer is a Schmitt Trigger input when used in Serial Programming mode. |
| GP2/T0CKI/AN2/INT | 11 | I/O | ST | Bi-directional I/O port/analog input 2. Can be configured as T0CKI or external interrupt. |
| GP3/$\overline{MCLR}$/$V_{PP}$ | 8 | I | TTL/ST | Input port/master clear (RESET) input/programming voltage input. When configured as $\overline{MCLR}$, this pin is an active low RESET to the device. Voltage on $\overline{MCLR}$/$V_{PP}$ must not exceed $V_{DD}$ during normal device operation. Can be software programmed for internal weak pull-up and interrupt-on-pin change. Weak pull-up always on if configured as $\overline{MCLR}$ . This buffer is Schmitt Trigger when in $\overline{MCLR}$ mode. |
| GP4/OSC2/AN3/CLKOUT | 7 | I/O | TTL | Bi-directional I/O port/oscillator crystal output/analog input 3. Connections to crystal or resonator in crystal oscillator mode (HS, XT and LP modes only, GPIO in other modes). In EXTRC and INTRC modes, the pin output can be configured to CLKOUT, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate. |
| GP5/OSC1/CLKIN | 6 | I/O | TTL/ST | Bi-directional IO port/oscillator crystal input/external clock source input (GPIO in INTRC mode only, OSC1 in all other oscillator modes). Schmitt trigger input for EXTRC oscillator mode. |
| LIN bus | 1 | I/O | HV/OD | High voltage bi-directional bus interface. |
| $V_{BAT}$ | 18 | P | — | Battery input voltage. |
| BACT | 17 | O | — | Bus activity output pin. |
| $V_{DD}$ | 4 | P | — | Positive supply for logic and I/O pins. |
| $V_{SS}$ | 2,3,14, 16 | P | — | Ground reference for logic and I/O pins. |
| $AV_{DD}$ | 5 | P | — | Analog positive supply. |
| $AV_{SS}$ | 15 | P | — | Analog ground. |

Legend: I = Input, O = Output, I/O = Input/Output, P = Power, — = not used, TTL = TTL input, ST = Schmitt Trigger input, OD = Open Drain

### 3.1 Clocking Scheme/Instruction Cycle

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

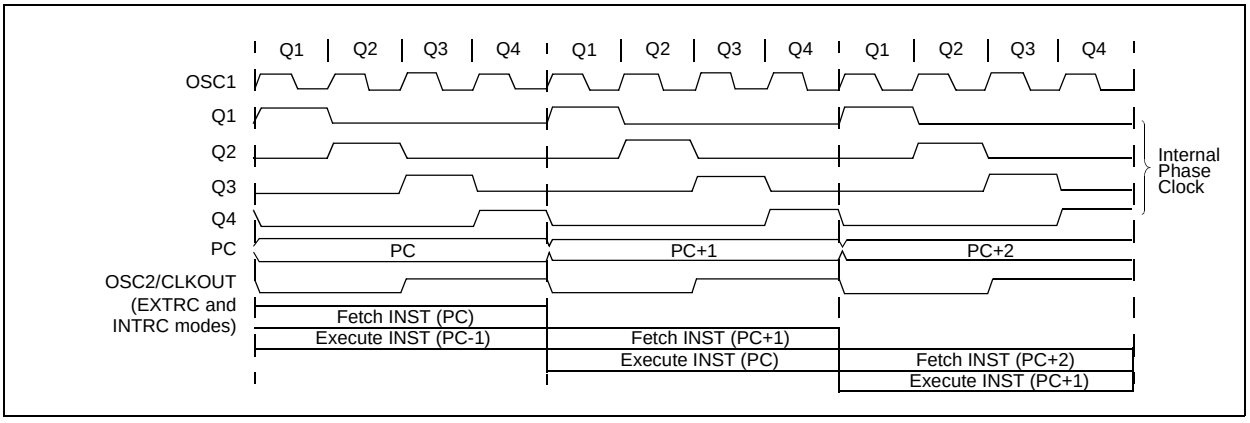
### 3.2 Instruction Flow/Pipelining

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

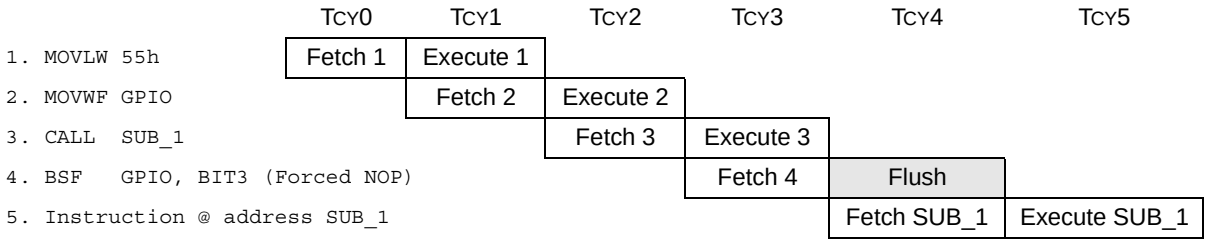
A fetch cycle begins with the program counter (PC) incrementing in Q1.

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 fetched instruction is “flushed” from the pipeline while the new instruction is being fetched and then executed.

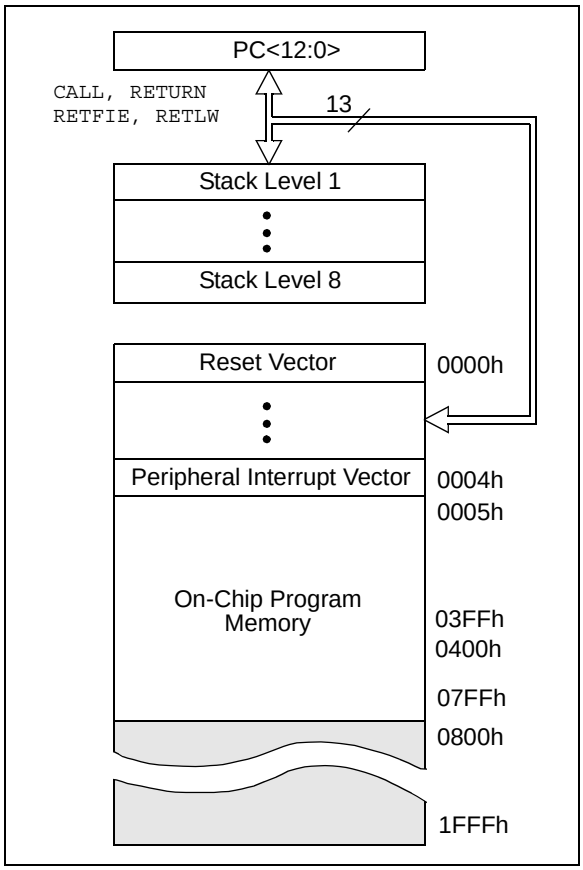
# 4.0 MEMORY ORGANIZATION

## 4.1 Program Memory Organization

The PIC16C433 has a 13-bit program counter capable of addressing an 8K x 14 program memory space.

For the PIC16C433 the first 2K x 14 (0000h-07FFh) is implemented. Accessing a location above the physically implemented address will cause a wraparound. The Reset Vector is at 0000h and the interrupt vector is at 0004h.

**FIGURE 4-1: PIC16C433 PROGRAM MEMORY MAP AND STACK**




## 4.2 Data Memory Organization

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

RP0 (STATUS<5>) = 1 → Bank 1

RP0 (STATUS<5>) = 0 → Bank 0

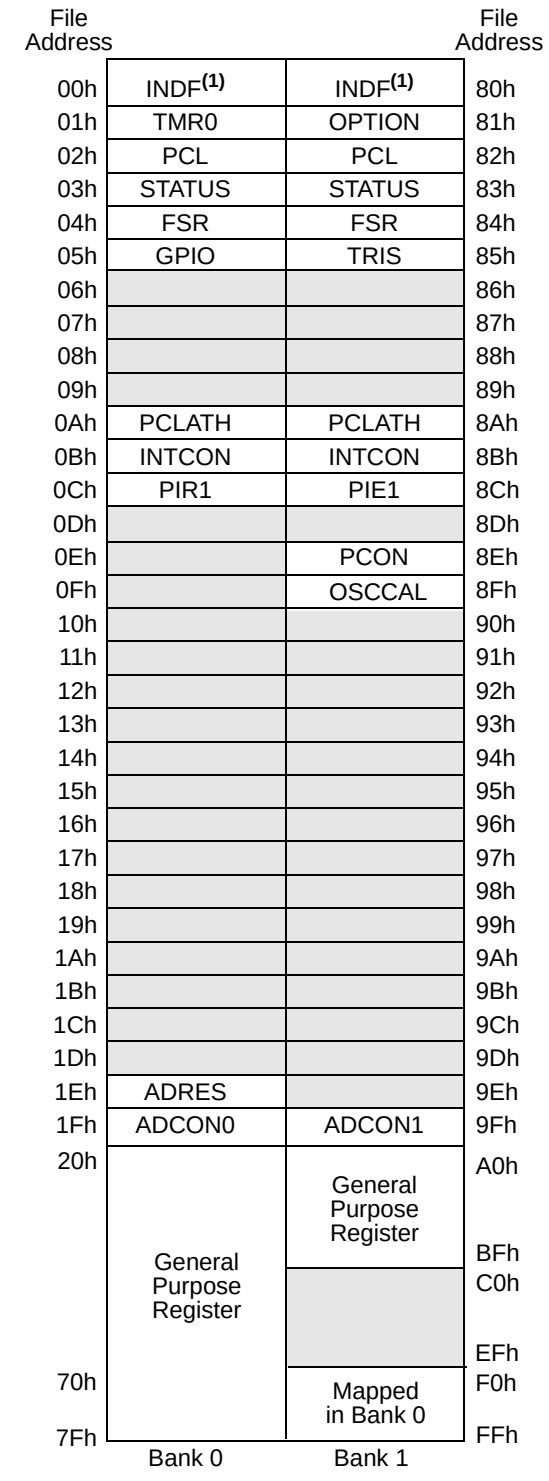
Each Bank extends up to 7Fh (128 bytes). The lower locations of each Bank are reserved for the Special Function Registers. Above the Special Function Registers are General Purpose Registers implemented as static RAM. Both Bank 0 and Bank 1 contain Special Function Registers. Some "high use" Special Function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

Also note that F0h through FFh on the PIC16C433 is mapped into Bank 0 registers 70h-7Fh as common RAM.

### 4.2.1 GENERAL PURPOSE REGISTER FILE

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

**FIGURE 4-2: PIC16C433 REGISTER FILE MAP**




Unimplemented data memory locations, read as '0'.

**Note 1:** Not a physical register.

### 4.2.2 SPECIAL FUNCTION REGISTERS

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

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

**TABLE 4-1: PIC16C433 SPECIAL FUNCTION REGISTER SUMMARY**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on Power-on Reset | Value on all other RESETS(3) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 0** | | | | | | | | | | | |
| 00h(1) | INDF | Addressing this location uses contents of FSR to address data memory (not a physical register) | | | | | | | | 0000 0000 | 0000 0000 |
| 01h | TMR0 | Timer0 module's register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 02h(1) | PCL | Program Counter's (PC) Least Significant Byte | | | | | | | | 0000 0000 | 0000 0000 |
| 03h(1) | STATUS | IRP(4) | RP1(4) | RP0 | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | 0001 1xxx | 000q quuu |
| 04h(1) | FSR | Indirect data memory address pointer | | | | | | | | xxxx xxxx | uuuu uuuu |
| 05h | GPIO | LINTX | LINRX | GP5 | GP4 | GP3 | GP2 | GP1 | GP0 | 11xx xxxx | 11uu uuuu |
| 06h | — | Unimplemented | | | | | | | | — | — |
| 07h | — | Unimplemented | | | | | | | | — | — |
| 08h | — | Unimplemented | | | | | | | | — | — |
| 09h | — | Unimplemented | | | | | | | | — | — |
| 0Ah(1,2) | PCLATH | — | — | — | Write Buffer for the upper 5 bits of the Program Counter | | | | | ---0 0000 | ---0 0000 |
| 0Bh(1) | INTCON | GIE | PEIE | T0IE | INTE | GPIE | T0IF | INTF | GPIF | 0000 000x | 0000 000u |
| 0Ch | PIR1 | — | ADIF | — | — | — | — | — | — | -0-- ---- | -0-- ---- |
| 0Dh | — | Unimplemented | | | | | | | | — | — |
| 0Eh | — | Unimplemented | | | | | | | | — | — |
| 0Fh | — | Unimplemented | | | | | | | | — | — |
| 10h | — | Unimplemented | | | | | | | | — | — |
| 11h | — | Unimplemented | | | | | | | | — | — |
| 12h | — | Unimplemented | | | | | | | | — | — |
| 13h | — | Unimplemented | | | | | | | | — | — |
| 14h | — | Unimplemented | | | | | | | | — | — |
| 15h | — | Unimplemented | | | | | | | | — | — |
| 16h | — | Unimplemented | | | | | | | | — | — |
| 17h | — | Unimplemented | | | | | | | | — | — |
| 18h | — | Unimplemented | | | | | | | | — | — |
| 19h | — | Unimplemented | | | | | | | | — | — |
| 1Ah | — | Unimplemented | | | | | | | | — | — |
| 1Bh | — | Unimplemented | | | | | | | | — | — |
| 1Ch | — | Unimplemented | | | | | | | | — | — |
| 1Dh | — | Unimplemented | | | | | | | | — | — |
| 1Eh | ADRES | A/D Result Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 1Fh | ADCON0 | ADCS1 | ADCS0 | reserved | CHS1 | CHS0 | GO/$\overline{DONE}$ | reserved | ADON | 0000 0000 | 0000 0000 |

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

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

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

**3:** Other (non power-up) RESETS include external RESET through $\overline{MCLR}$ and Watchdog Timer Reset.

**4:** The IRP and RP1 bits are reserved on the PIC16C433; always maintain these bits clear.

**TABLE 4-1: PIC16C433 SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

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

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

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

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

**3:** Other (non power-up) RESETS include external RESET through $\overline{MCLR}$ and Watchdog Timer Reset.

**4:** The IRP and RP1 bits are reserved on the PIC16C433; always maintain these bits clear.

4.2.2.1 STATUS Register

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

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

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

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

**Note 1:** Bits IRP and RP1 (STATUS<7:6>) are not used by the PIC16C433 and should be maintained clear. Use of these bits as general purpose R/W bits is NOT recommended, since this may affect upward compatibility with future products.

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

**REGISTER 4-1: STATUS REGISTER (ADDRESS 03h, 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, 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, always maintain this bit clear.

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

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

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

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

bit 0 **C:** Carry/$\overline{borrow}$ bit (`ADDWF`, `ADDLW`, `SUBLW`, `SUBWF` instructions)
`1` = A carry-out from the most significant bit of the result occurred
`0` = No carry-out from the most significant bit of the result occurred

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

| 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 |

4.2.2.2 OPTION Register

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

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

**REGISTER 4-2: OPTION REGISTER (ADDRESS 81h))**

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

bit 7 **$\overline{\text{GPPU}}$**: Weak Pull-up Enable bit
1 = Weak pull-ups disabled
0 = Weak pull-ups enabled (GP0, GP1, GP3)

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

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

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

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

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

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

| 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 various enable and flag bits for the TMR0 Register overflow, GPIO port change and external GP2/INT pin interrupts.

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

**REGISTER 4-3: INTCON REGISTER (ADDRESS 0Bh, 8Bh)**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-x |
|---|---|---|---|---|---|---|---|
| GIE | PEIE | T0IE | INTE | GPIE | T0IF | INTF | GPIF |
| bit7 | | | | | | | bit0 |

bit 7 **GIE:** Global Interrupt Enable bit
1 = Enables all unmasked interrupts
0 = Disables all interrupts

bit 6 **PEIE**: Peripheral Interrupt Enable bit
1 = Enables all unmasked peripheral interrupts
0 = Disables all peripheral interrupts

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

bit 4 **INTE**: INT External Interrupt Enable bit
1 = Enables the external interrupt on GP2/INT/T0CKI/AN2 pin
0 = Disables the external interrupt on GP2/INT/T0CKI/AN2 pin

bit 3 **GPIE**: GPIO Interrupt-on-Change Enable bit
1 = Enables the GPIO Interrupt-on-Change
0 = Disables the GPIO Interrupt-on-Change

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

bit 1 **INTF**: RB0/INT External Interrupt Flag bit
1 = The external interrupt on GP2/INT/T0CKI/AN2 pin occurred (must be cleared in software)
0 = The external interrupt on GP2/INT/T0CKI/AN2 pin did not occur

bit 0 **GPIF**: GPIO Interrupt-on-Change Flag bit
1 = GP0, GP1 or GP3 pins changed state (must be cleared in software)
0 = Neither GP0, GP1 nor GP3 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 |

4.2.2.4 PIE1 Register

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

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

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

| U-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | ADIE | — | — | — | — | — | — |
| bit7 | | | | | | | bit0 |

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

bit 6 **ADIE**: A/D Converter Interrupt Enable bit
1 = Enables the A/D interrupt
0 = Disables the A/D interrupt

bit 5-0 **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 |

4.2.2.5 PIR1 Register

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

| **Note:** | Interrupt flag bits get set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the global enable bit, GIE (INTCON<7>). User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. |
|---|---|

**REGISTER 4-5: PIR1 REGISTER (ADDRESS 0Ch))**

| U-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | ADIF | — | — | — | — | — | — |
| bit7 | | | | | | | bit0 |

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

bit 6 **ADIF**: A/D Converter Interrupt Flag bit
1 = An A/D conversion completed (must be cleared in software)
0 = The A/D conversion is not complete

bit 5-0 **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 |

4.2.2.6 PCON Register

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

**REGISTER 4-6: PCON REGISTER (ADDRESS 8Eh))**

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

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

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

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

| 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 |

4.2.2.7 OSCCAL Register

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

**REGISTER 4-7: OSCCAL REGISTER (ADDRESS 8Fh)**

| R/W-0 | R/W-1 | R/W-1 | R/W-1 | R/W-0 | R/W-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| CAL3 | CAL2 | CAL1 | CAL0 | CALFST | CALSLW | — | — |
| bit7 | | | | | | | bit0 |

bit 7-4 **CAL<3:0>:** Fine Calibration'

bit 3 **CALFST**: Calibration Fast
1 = Increase frequency
0 = No change

bit 2 **CALSLW**: Calibration Slow
1 = Decrease frequency
0 = No change

bit 1-0 **Unimplemented:** Read as '0'

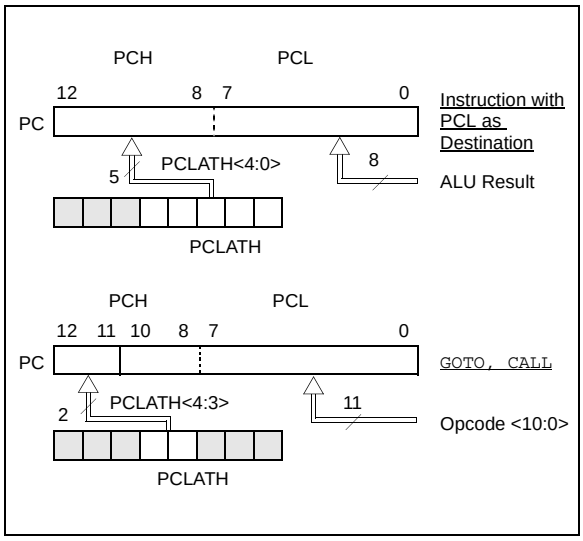
**Note:** If CALFST = 1 and CALSLW = 1, CALFST has precedence.

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

**FIGURE 4-3: 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 PIC16C433 family has an 8-level deep x 13-bit wide hardware stack. The stack space is not part of either program or data space and the stack pointer is not readable or writable. The PC is PUSHed onto the stack when a CALL instruction is executed, or an interrupt causes a branch. The stack is POPed in the event of a RETURN, RETLW or a RETFIE instruction execution. PCLATH is not affected by a PUSH or POP operation.

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

| | |
|---|---|
| **Note 1:** | There are no status bits to indicate stack overflow or stack underflow conditions. |
| **2:** | There are no instructions/mnemonics called PUSH or POP. These are actions that occur from the execution of the CALL, RETURN, RETLW, and RETFIE instructions, or the vectoring to an interrupt address. |

## 4.4 Program Memory Paging

The PIC16C433 ignores both paging bits PCLATH<4:3>, which are used to access program memory when more than one page is available. The use of PCLATH<4:3> as general purpose read/write bits for the PIC16C433 is not recommended, since this may affect upward compatibility with future products.

### 4.5 Indirect Addressing, INDF and FSR Registers

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

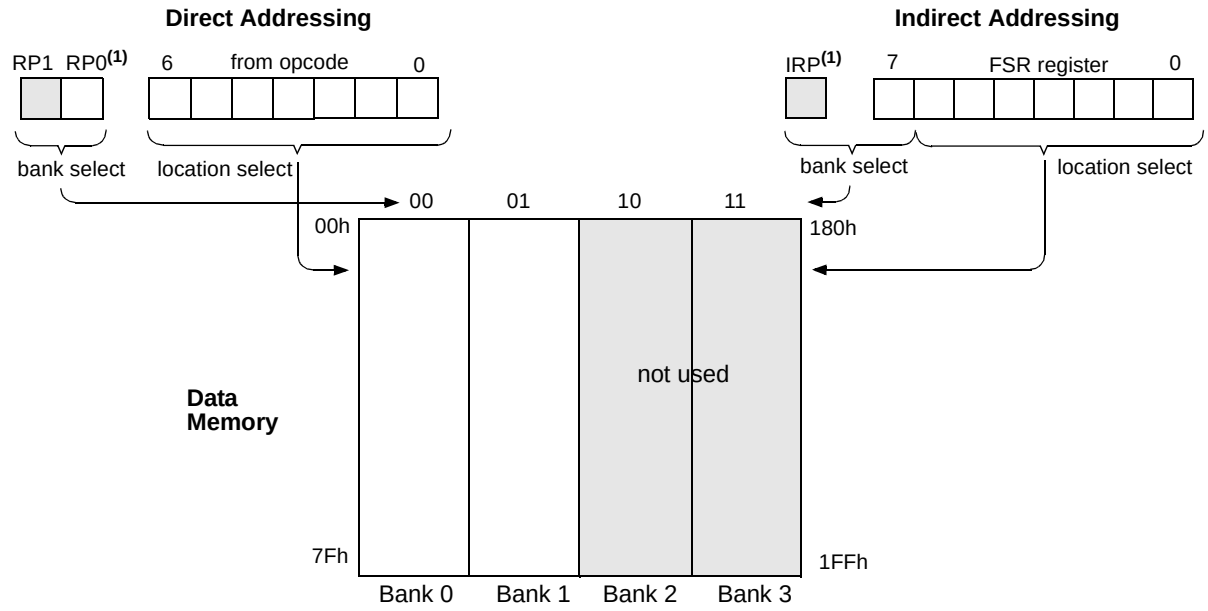
Any instruction using the INDF register actually accesses the register pointed to by the File Select Register, FSR. Reading the INDF register itself, indirectly (FSR = '0') will read 00h. Writing to the INDF register, indirectly results in a no-operation (although status bits may be affected). An effective 9-bit address is obtained by concatenating the 8-bit FSR register and the IRP bit (STATUS<7>), as shown in Figure 4-4. However, IRP is not used in the PIC16C433.

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,F   ;inc pointer
        btfss   FSR,4   ;all done?
        goto    NEXT    ;no clear next
CONTINUE
        :               ;yes continue
```

**FIGURE 4-4: DIRECT/INDIRECT ADDRESSING**




For register file map detail see Figure 4-2.

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

**NOTES:**

## 5.0 I/O PORT

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

### 5.1 GPIO

GPIO is an 8-bit I/O register. Only the low order 6 bits are used (GP<5:0>). Bits 6 and 7 (LINTX and LINRX, respectively) are used by the LIN bus transceiver peripheral. Please note that GP3 is an input only pin. The configuration word can set several I/O's to alternate functions. When acting as alternate functions, the pins will read as '0' during port read. Pins GP0, GP1 and GP3 can be configured with weak pull-ups and also with interrupt-on-change. The interrupt-on-change and weak pull-up functions are not pin selectable. If pin 4 (GP3), is configured as MCLR, a weak pull-up is always on. Interrupt-on-change for this pin is not set and GP3 will read as '0'. Interrupt-on-change is enabled by setting bit GPIE, INTCON<3>. Note that external oscillator use overrides the GPIO functions on GP4 and GP5.

### 5.2 TRIS Register

This register controls the data direction for GPIO. A '1' from a TRIS Register bit puts the corresponding output driver in a hi-impedance mode. A '0' puts the contents of the output data latch on the selected pins, enabling the output buffer. The exceptions are GP3, which is input only and its TRIS bit will always read as '1', while GP6 and GP7 TRIS bits will read as '0'.

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

Upon RESET, the TRIS Register is all '1's, making all pins inputs.

TRIS for pins GP4 and GP5 is forced to a '1', where appropriate. Writes to TRIS <5:4> will have an effect in EXTRC and INTRC oscillator modes only. When GP4 is configured as CLKOUT, changes to TRIS<4> will have no effect.

### 5.3 I/O Interfacing

The equivalent circuit for an I/O port pin is shown in Figure 5-1 through Figure 5-5. All port pins, except GP3, which is input only, may be used for both input and output operations. For input operations, these ports are non-latching. Any input must be present until read by an input instruction (i.e., `MOVF GPIO,W`). The outputs are latched and remain unchanged until the output latch is rewritten. To use a port pin as output, the corresponding direction control bit in TRIS must be cleared (= 0). For use as an input, the corresponding TRIS bit must be set. Any I/O pin (except GP3) can be programmed individually as input or output.

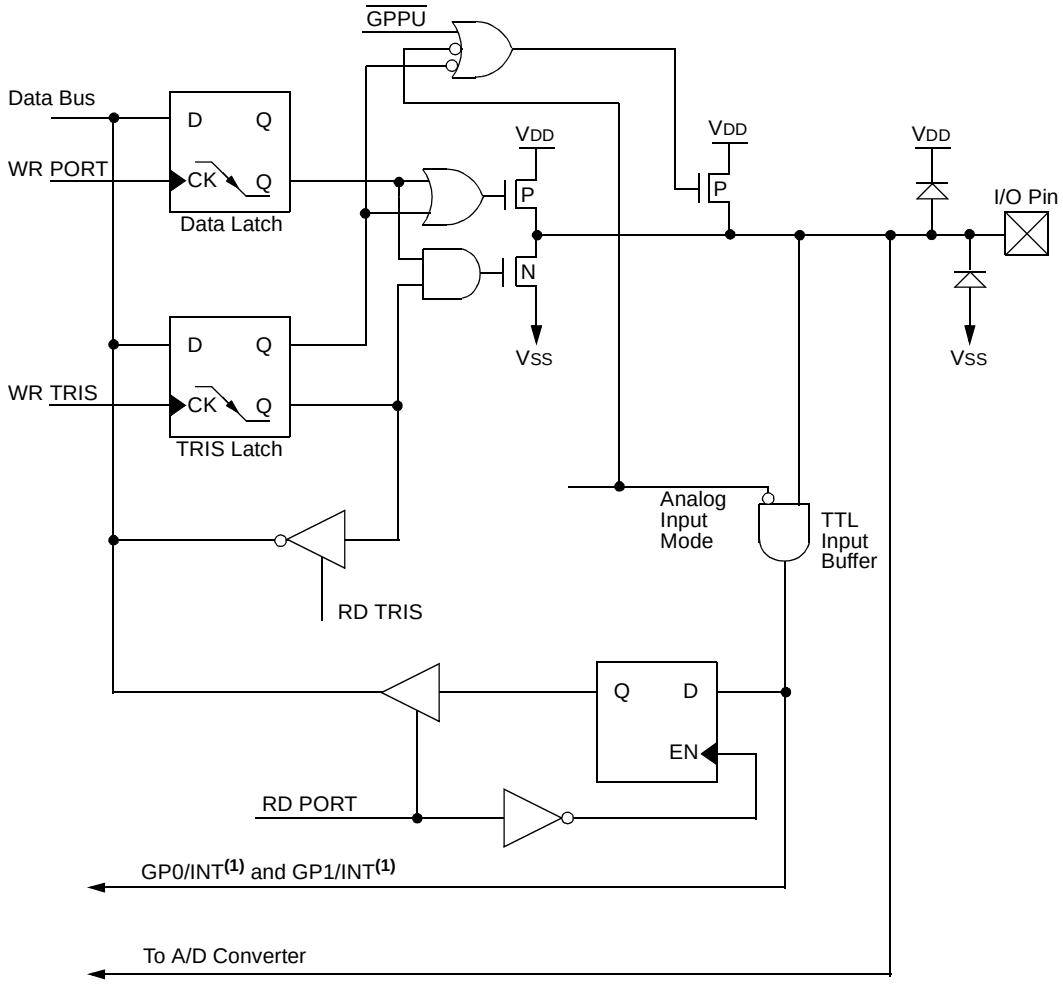
Port pins LINTX and LINRX are used for the LIN bus transceiver. These port pins are not available externally on the package. Users should avoid writing to pins GP6 (SDA) and GP7 (SCL), when not communicating with the LIN bus transceiver.

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

### 5.4 The BACT Pin

The BACT pin can be used to wake-up the device from SLEEP upon bus activity. In order to wake the bus up, the BACT pin has to be tied to either GP0, GP1 or GP2. These pins can be configured to wake the PIC16C433 upon a change. The BACT pin reflects the communication on the LIN bus. This pin is not latched.

**FIGURE 5-1: BLOCK DIAGRAM OF GP0/AN0 AND GP1/AN1/VREF PIN**




**Note 1:** Wake-up on pin change interrupts for GP0 and GP1.

**FIGURE 5-2: BLOCK DIAGRAM OF GP2/T0CKI/AN2/INT PIN**

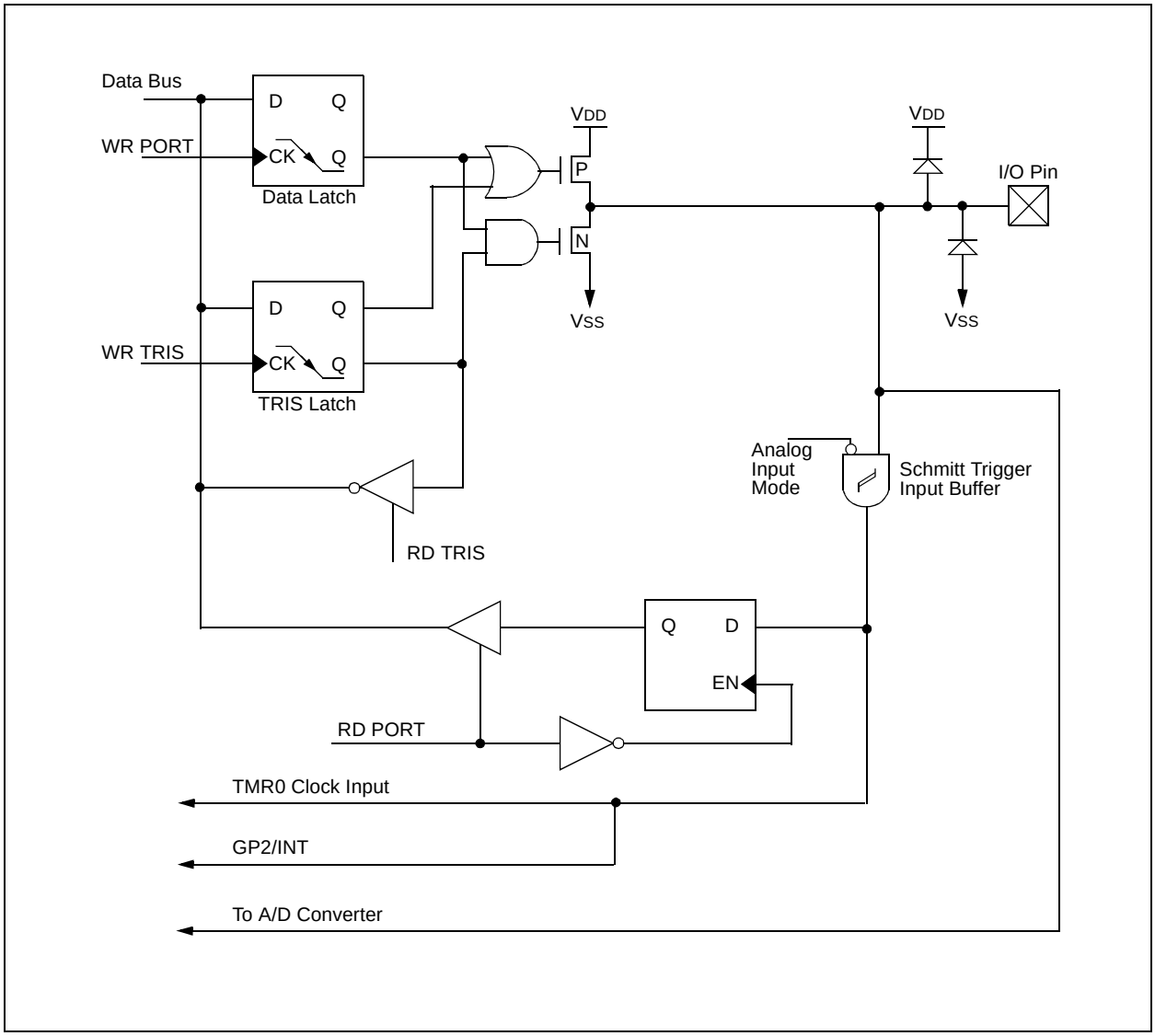
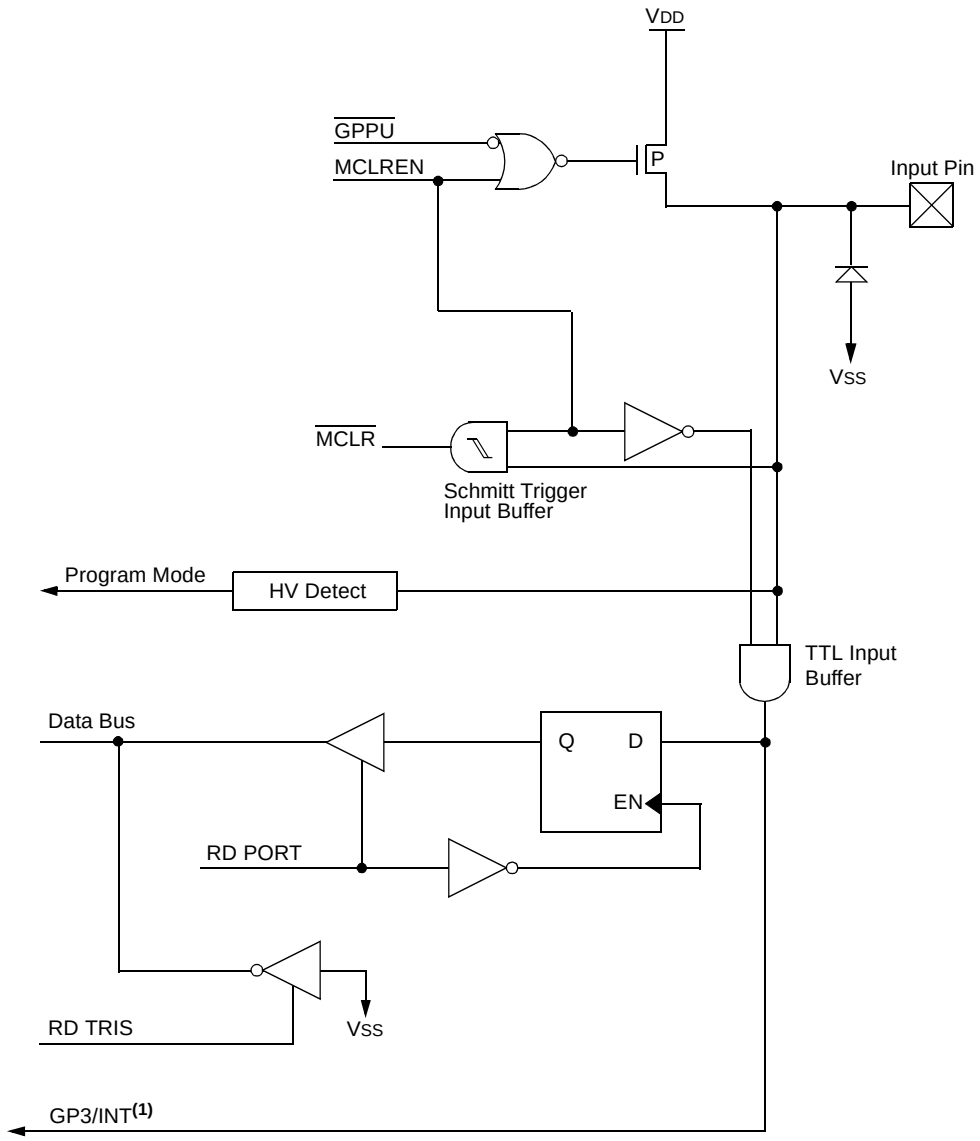


**FIGURE 5-3: BLOCK DIAGRAM OF GP3/$\overline{\text{MCLR}}$/VPP PIN**




**Note 1:** Wake-up on pin change interrupt for GP3.

**FIGURE 5-4: BLOCK DIAGRAM OF GP4/OSC2/AN3/CLKOUT PIN**

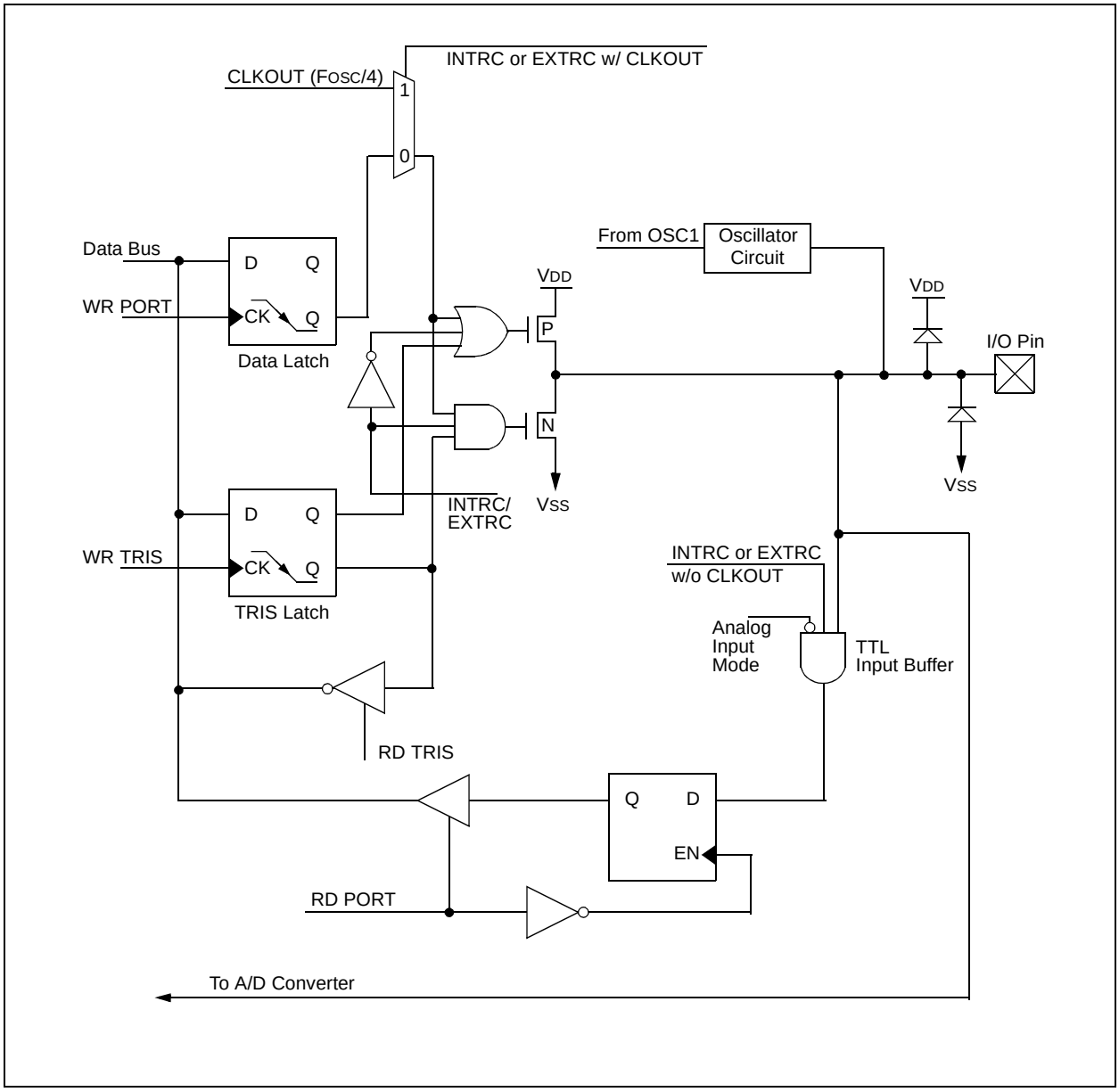


**FIGURE 5-5: BLOCK DIAGRAM OF GP5/OSC1/CLKIN PIN**

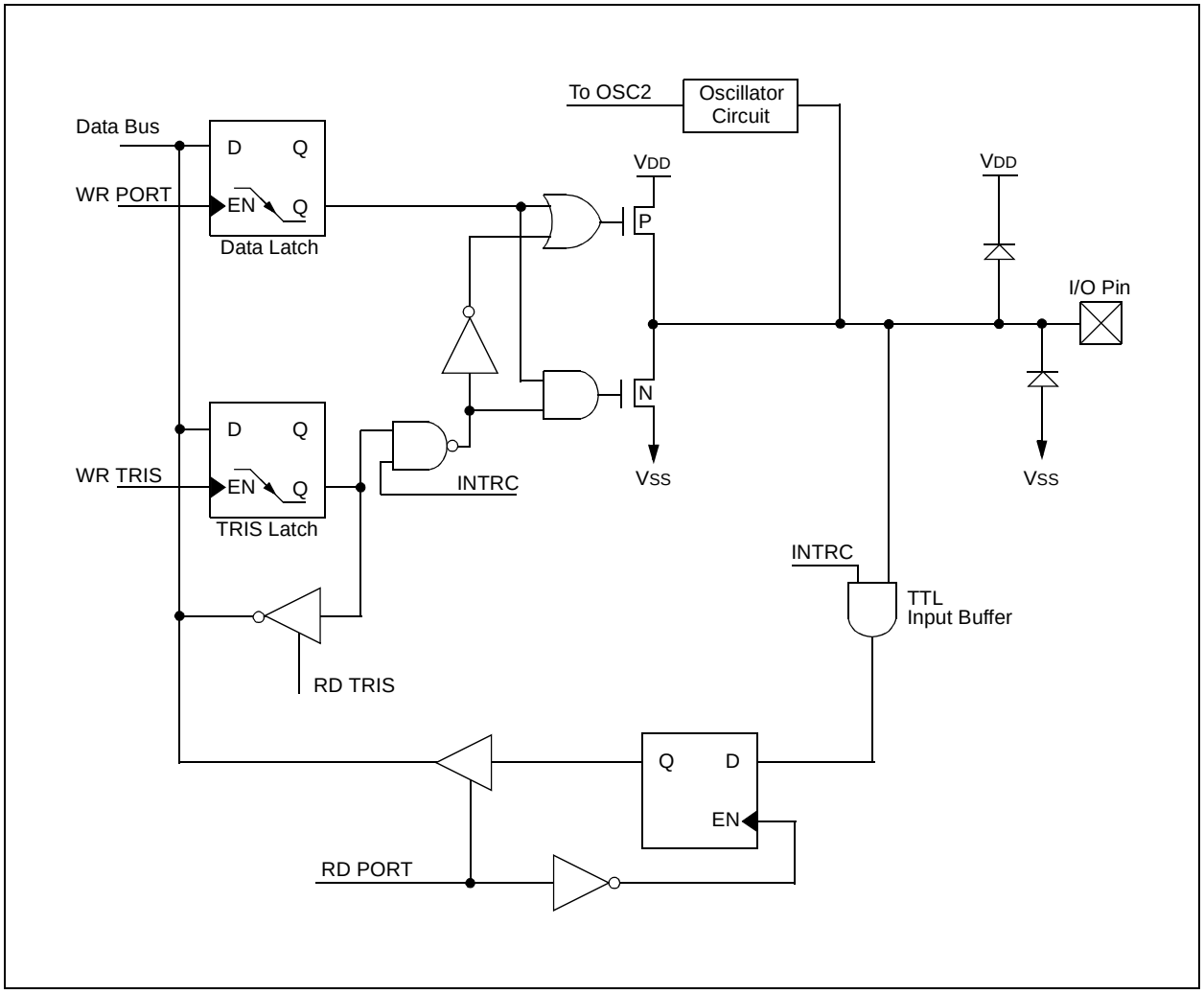



**TABLE 5-1: SUMMARY OF PORT REGISTERS**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on Power-on Reset | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 85h | TRIS | — | — | GPIO Data Direction Register | | | | | | --11 1111 | --11 1111 |
| 81h | OPTION | $\overline{\text{GPPU}}$ | INTEDG | T0CS | T0SE | PSA | PS2 | PS1 | PS0 | 1111 1111 | 1111 1111 |
| 03h | STATUS | IRP(1) | RP1(1) | RP0 | $\overline{\text{TO}}$ | $\overline{\text{PD}}$ | Z | DC | C | 0001 1xxx | 000q quuu |
| 05h | GPIO | LINTX | LINRX | GP5 | GP4 | GP3 | GP2 | GP1 | GP0 | 11xx xxxx | 11uu uuuu |

Legend: Shaded cells not used by Port Registers, read as '0', — = unimplemented, read as '0', x = unknown, u = unchanged, q = see tables in Section 9.4 for possible values.

**Note 1:** The IRP and RP1 bits are reserved on the PIC16C433; always maintain these bits clear.

## 5.5 I/O Programming Considerations

### 5.5.1 BI-DIRECTIONAL I/O PORTS

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

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

Example 5-1 shows the effect of two sequential read-modify-write instructions on an I/O port.

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

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

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

**NOTES:**

## 6.0 LIN bus TRANSCEIVER

The PIC16C433 has an integrated LIN bus transceiver, which allows the microcontroller to communicate via a LIN bus. The LIN bus protocol is handled by the microcontroller. The conversion from 5V signal to LIN bus signals is handled by the transceiver

### 6.1 The LIN bus Protocol

The LIN bus protocol is not described within this document. For further information regarding the LIN bus protocol, please refer to www.lin-subbus.de.

### 6.2 LIN bus Interfacing

The LIN bus will be transmitted by toggling the LINTX bit in the LININTF register. Data transmitted on the LIN bus is read by checking the LINRX bit in the PORTA register.

For a LIN bus Slave software implementation, please refer to AN729.

### 6.3 LIN bus Hardware Interface

Figure 6-1shows how to implement a hardware LIN bus interface for the PIC16C433.

An external 45V zener diode between $V_{BB}$ and ground, with a 500Ω resistor in series with the battery supply and the $V_{BB}$ pin, protects the PIC16C433 from power transients.

An external reverse battery blocking diode is used to provide polarity protection.

### 6.4 Thermal Shut-down

In thermal shut-down, the LIN bus output is disabled instantaneously. The output transistor is turned off, regardless of the input level at pin LINTX bit and only a limited current can flow into the receiver connected to the LIN bus pin.

| **Note:** | TLINRX must be set to '1' at all times. |
|---|---|

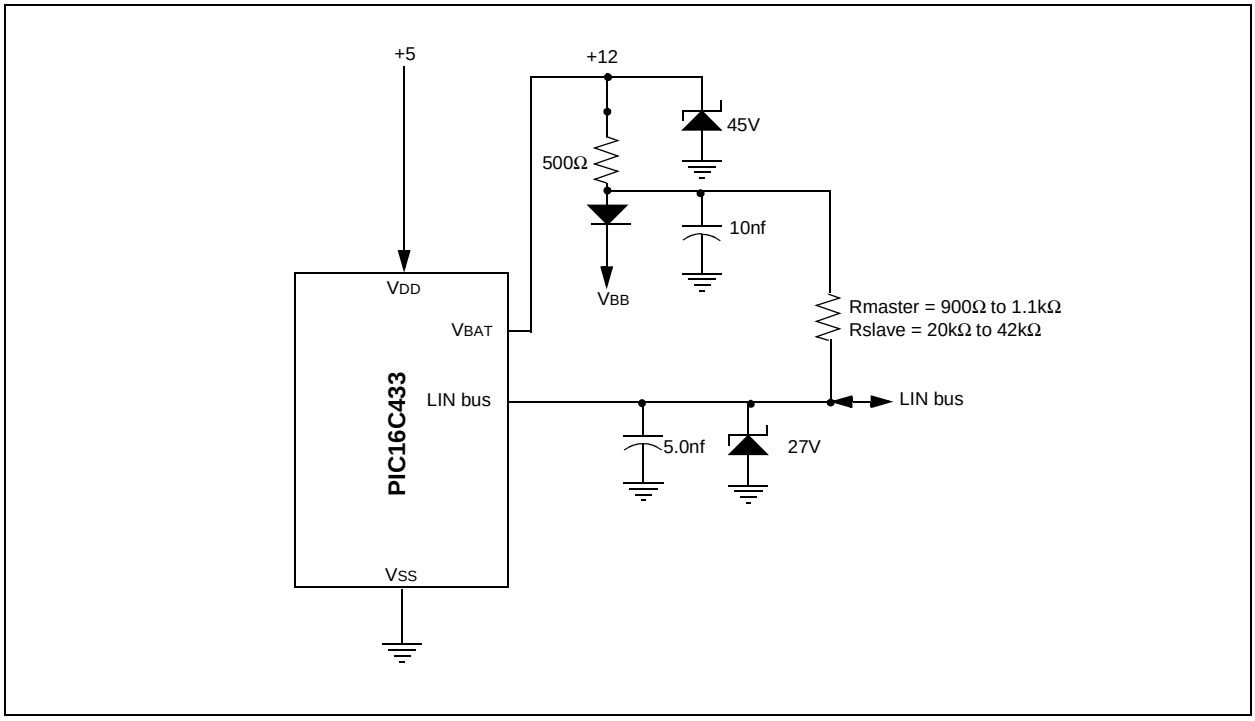
### 6.5 Wake-up from SLEEP upon Bus Activity

The PIC16C433 can wake-up from SLEEP upon bus activity in the following way:

1. Connect BACT to GP2/T0CKI/AN2/INT.

The BACT signal is a mirror of the LIN bus. This signal can be routed to GP2/T0CKI/AN2/INT pin. The GP2/T0CKI/AN2/INT interrupt wakes up the device from SLEEP.

**FIGURE 6-1: TYPICAL LIN bus APPLICATION**



**NOTES:**

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

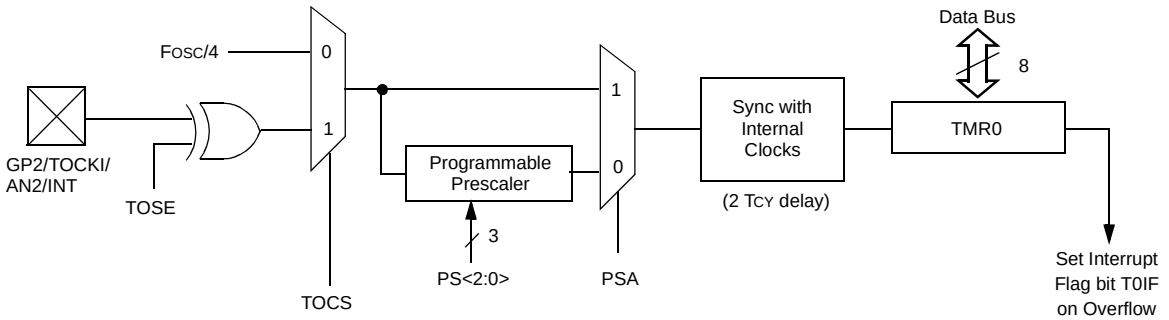
Counter mode is selected by setting bit T0CS (OPTION<5>). In Counter mode, Timer0 will increment either on every rising or falling edge of pin RA4/T0CKI. The incrementing edge is determined by the bit T0SE (OPTION<4>). Clearing bit T0SE selects the rising edge. Restrictions on the external clock input are discussed in detail in Section 7.2.

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

### 7.1 Timer0 Interrupt

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

**FIGURE 7-1: TIMER0 BLOCK DIAGRAM**




**Note 1:** TOCS, TOSE, PSA, PS<2:0> (OPTION<5:0>).
**2:** The prescaler is shared with Watchdog Timer (refer to Figure 7-6 for detailed block diagram).

**FIGURE 7-2: TIMER0 TIMING: INTERNAL CLOCK/NO PRESCALE**

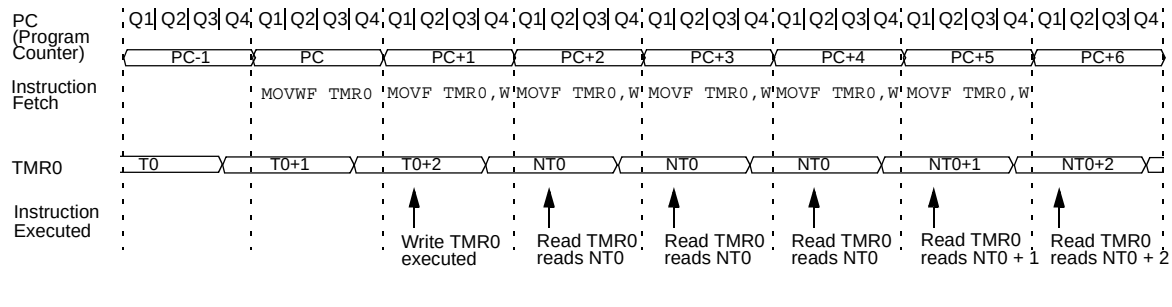


**FIGURE 7-3: TIMER0 TIMING: INTERNAL CLOCK/PRESCALE 1:2**

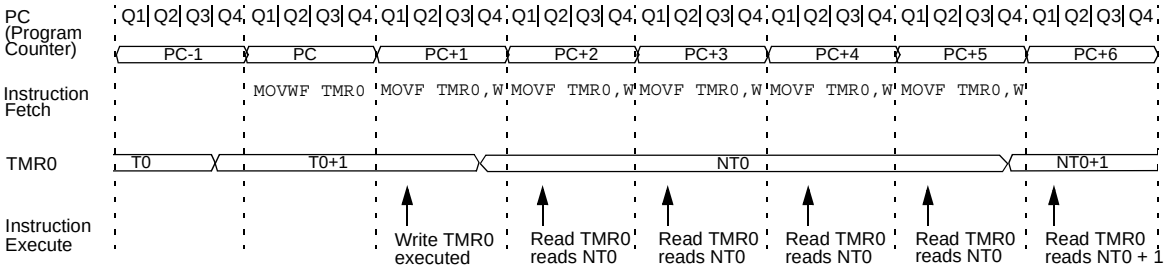
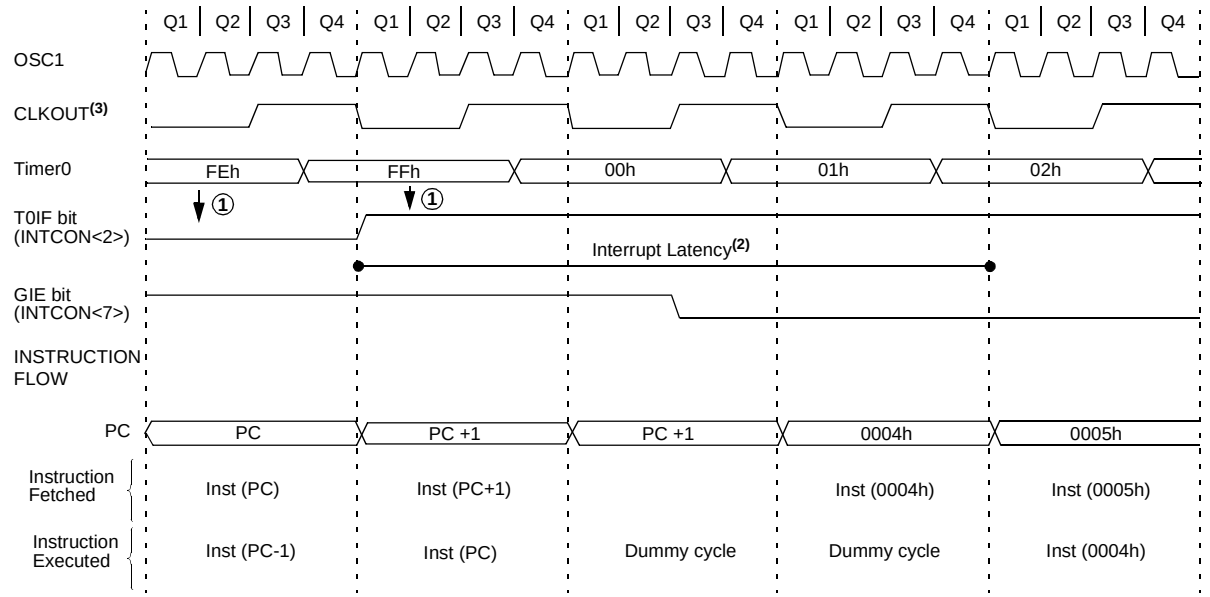



**FIGURE 7-4: TIMER0 INTERRUPT TIMING**




**Note 1:** Interrupt flag bit T0IF is sampled here (every Q1).

**2:** Interrupt latency = 3TCY where TCY = instruction cycle time.

**3:** CLKOUT is available only in the INTRC and EXTRC oscillator modes.

## 7.2 Using Timer0 with an External Clock

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

### 7.2.1 EXTERNAL CLOCK SYNCHRONIZATION

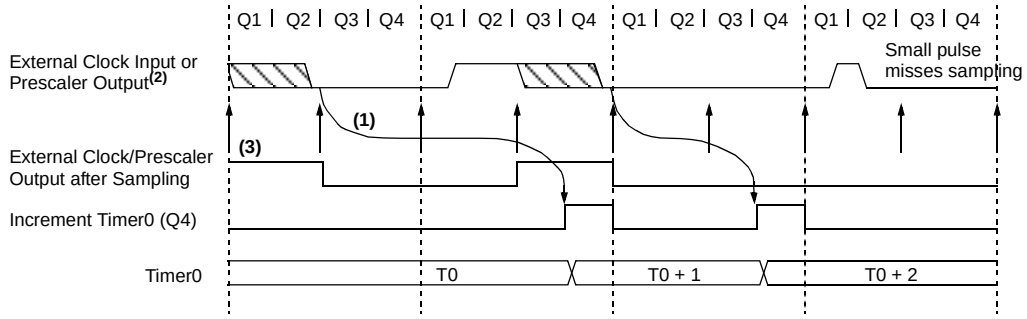
When no prescaler is used, the external clock input is used as the clock source. The synchronization of T0CKI with the internal phase clocks is accomplished by sampling the prescaler output on the Q2 and Q4 cycles of the internal phase clocks (Figure 7-5). Therefore, it is necessary for T0CKI to be high 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 TMR0 INCREMENT DELAY

Since the prescaler output is synchronized with the internal clocks, there is a small delay from the time the external clock edge occurs to the time the Timer0 module is actually incremented. Figure 7-5 shows the delay from the external clock edge to the timer incrementing.

**FIGURE 7-5: TIMER0 TIMING WITH EXTERNAL CLOCK**




**Note 1:** Delay from clock input change to Timer0 increment is 3TOSC to 7TOSC (Duration of Q = TOSC). Therefore, the error in measuring the interval between two edges on Timer0 input = ±4TOSC max.

**2:** External clock if no prescaler selected; prescaler output otherwise.

**3:** The arrows indicate the points in time where sampling occurs.

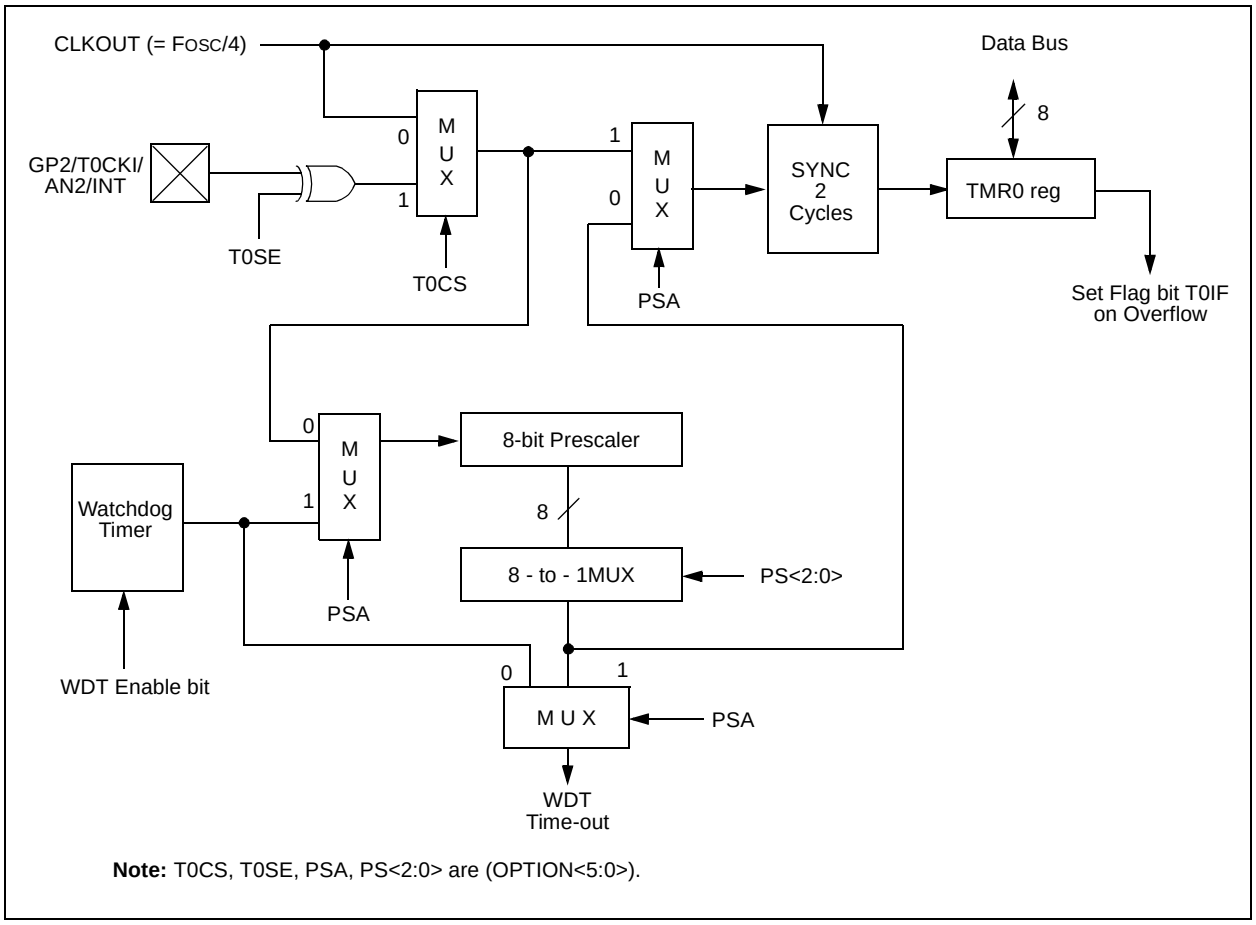
### 7.3 Prescaler

An 8-bit counter is available as a prescaler for the Timer0 module, or as a postscaler for the Watchdog Timer, respectively (Figure 7-6). For simplicity, this counter is being referred to as “prescaler” throughout this data sheet. Note that there is only one prescaler available which is mutually exclusively shared between the Timer0 module and the Watchdog Timer. Thus, a prescaler assignment for the Timer0 module means that there is no prescaler for the Watchdog Timer, and vice-versa.

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

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

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




**Note:** T0CS, T0SE, PSA, PS<2:0> are (OPTION<5:0>).

### 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).

**Note:** To avoid an unintended device RESET, the following instruction sequence (shown in Example 7-1) must be executed when changing the prescaler assignment from Timer0 to the WDT. This sequence must be followed even if the WDT is disabled.

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

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

To change prescaler from the WDT to the Timer0 module, use the sequence shown in Example 7-2.

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

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

**TABLE 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 | PEIE | T0IE | INTE | GPIE | T0IF | INTF | GPIF | 0000 000x | 0000 000u |
| 81h | OPTION | $\overline{\text{GPPU}}$ | INTEDG | T0CS | T0SE | PSA | PS2 | PS1 | PS0 | 1111 1111 | 1111 1111 |
| 85h | TRIS | — | — | TRIS5 | TRIS4 | TRIS3 | TRIS2 | TRIS1 | TRIS0 | --11 1111 | --11 1111 |

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

**NOTES:**

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

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

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

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

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

The ADCON0 Register, shown in Figure 8-1, controls the operation of the A/D module. The ADCON1 Register, shown in Figure 8-2, configures the functions of the port pins. The port pins can be configured as analog inputs (GP1 can also be a voltage reference) or as digital I/O.

| | |
|---|---|
| **Note 1:** | If the port pins are configured as analog inputs (RESET condition), reading the port (`MOVF GPIO,W`) results in reading '0's. |
| **2:** | Changing ADCON1 Register can cause the GPIF and INTF flags to be set in the INTCON Register. These interrupts should be disabled prior to modifying ADCON1. |

**REGISTER 8-1: ADCON0 REGISTER (ADDRESS 1Fh)**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| ADCS1 | ADCS0 | reserved | CHS1 | CHS0 | GO/$\overline{\text{DONE}}$ | reserved | ADON |
| bit7 | | | | | | | bit0 |

bit 7-6 **ADCS<1:0>:** A/D Conversion Clock Select bits
00 = $F_{OSC}$/2
01 = $F_{OSC}$/8
10 = $F_{OSC}$/32
11 = $F_{RC}$ (clock derived from an RC oscillation)

bit 5 **Reserved**

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

bit 2 **GO/$\overline{\text{DONE}}$:** A/D Conversion Status bit

If ADON = 1:
1 = A/D conversion in progress (setting this bit starts the A/D conversion)
0 = A/D conversion not in progress (this bit is automatically cleared by hardware when the A/D conversion is complete)

bit 1 **Reserved**

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

| 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 |

**REGISTER 8-2: ADCON1 REGISTER (ADDRESS 9Fh**

| U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | PCFG2 | PCFG1 | PCFG0 |
| bit7 | | | | | | | bit0 |

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

bit 1-0 **PCFG<2:0>:** A/D Port Configuration Control bits

| PCFG<2:0> | GP4 | GP2 | GP1 | GP0 | VREF |
|---|---|---|---|---|---|
| 000(1) | A | A | A | A | VDD |
| 001 | A | A | VREF | A | GP1 |
| 010 | D | A | A | A | VDD |
| 011 | D | A | VREF | A | GP1 |
| 100 | D | D | A | A | VDD |
| 101 | D | D | VREF | A | GP1 |
| 110 | D | D | D | A | VDD |
| 111 | D | D | D | D | VDD |

A = Analog Input

D = Digital I/O

**Note 1:** Value on RESET.

**2:** Any instruction that reads a pin configured as an analog input will read a '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 |

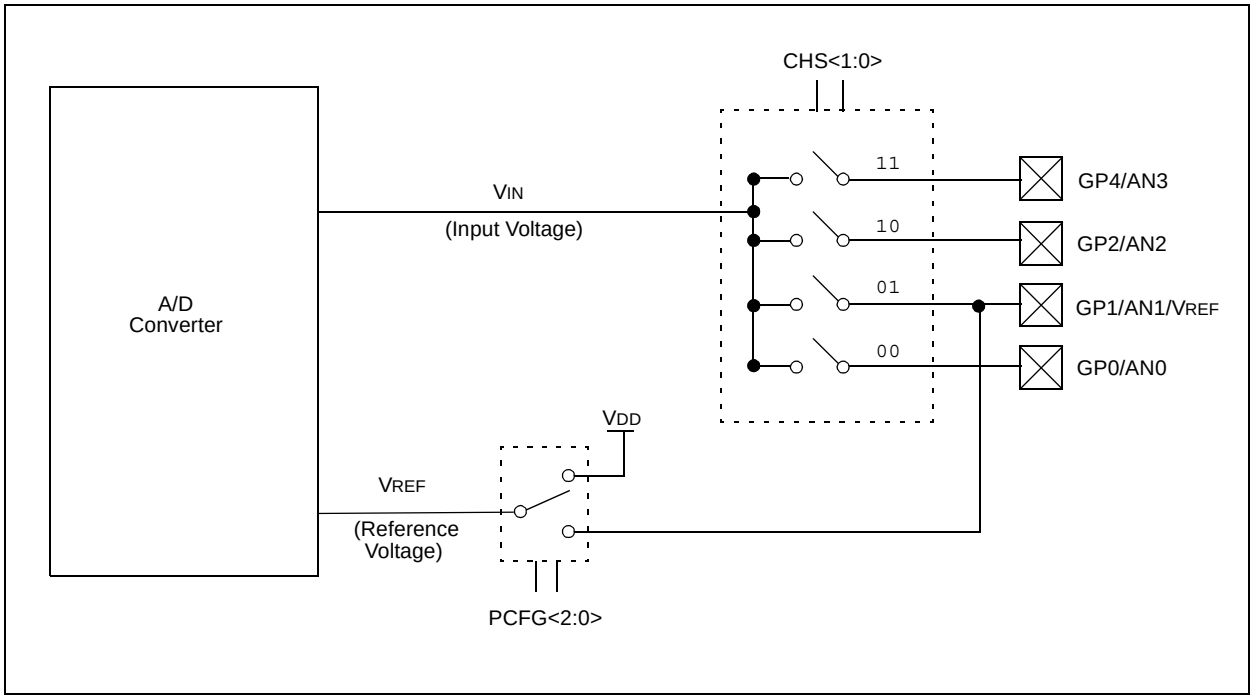
The ADRES Register contains the result of the A/D conversion. When the A/D conversion is complete, the result is loaded into the ADRES register, the GO/$\overline{\text{DONE}}$ bit (ADCON0<2>) is cleared and A/D interrupt flag bit ADIF (PIE1<6>) is set. The block diagrams of the A/D module are shown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

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

1. Configure the A/D module:
   - Configure analog pins/voltage reference/ and digital I/O (ADCON1 and TRIS)
   - Select A/D input channel (ADCON0)
   - Select A/D conversion clock (ADCON0)
   - Turn on A/D module (ADCON0)
2. Configure A/D interrupt (if desired):
   - Clear ADIF bit
   - Set ADIE bit
   - Set GIE bit
3. Wait the required acquisition time.
4. Start conversion:
   - Set GO/$\overline{\text{DONE}}$ bit (ADCON0)
5. Wait for A/D conversion to complete, by either:
   - Polling for the GO/$\overline{\text{DONE}}$ bit to be cleared

   OR
   - Waiting for the A/D interrupt
6. Read A/D Result Register (ADRES), clear bit ADIF if required.
7. For the next conversion, go to step 1, step 2, or step 3, as required. The A/D conversion time per bit is defined as TAD. A minimum wait of 2TAD is required before next acquisition starts.

**FIGURE 8-1: A/D BLOCK DIAGRAM**



### 8.1 A/D Sampling Requirements

For the A/D converter to meet its specified accuracy, the charge holding capacitor (CHOLD) must be allowed to fully charge to the input channel voltage level. The analog input model is shown in Figure 8-2. The source impedance (RS) and the internal sampling switch (RSS) impedance directly affect the time required to charge the capacitor CHOLD. The sampling switch (RSS) impedance varies over the device voltage (VDD), see Figure 8-2. **The maximum recommended impedance for analog sources is 10 kΩ.** After the analog input channel is selected (changed), this acquisition must be done before the conversion can be started.

To calculate the minimum acquisition time, Equation 8-1 may be used. This equation assumes that 1/2 LSb error is used (512 steps for the A/D). The 1/2 LSb error is the maximum error allowed for the A/D to meet its specified resolution.

**EQUATION 8-1: A/D MINIMUM CHARGING TIME**

$V_{HOLD} = (V_{REF} - (V_{REF}/512)) \cdot (1 - e^{(-Tc/C_{HOLD}(R_{IC} + R_{SS} + R_S))})$

or

$Tc = -(51.2\ pF)(1\ k\Omega + R_{SS} + R_S)\ \ln(1/511)$

Example 8-1 shows the calculation of the minimum required acquisition time TACQ. This calculation is based on the following system assumptions.

Rs = 10 kΩ

1/2 LSb error

VDD = 5V → Rss = 7 kΩ

Temperature (system max.) = 50°C

VHOLD = 0 @ t = 0

**Note 1:** The reference voltage (VREF) has no effect on the equation, since it cancels itself out.

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

**3:** The maximum recommended impedance for analog sources is 10 kΩ. This is required to meet the pin leakage specification.

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

**EXAMPLE 8-1: CALCULATING THE MINIMUM REQUIRED SAMPLE TIME**

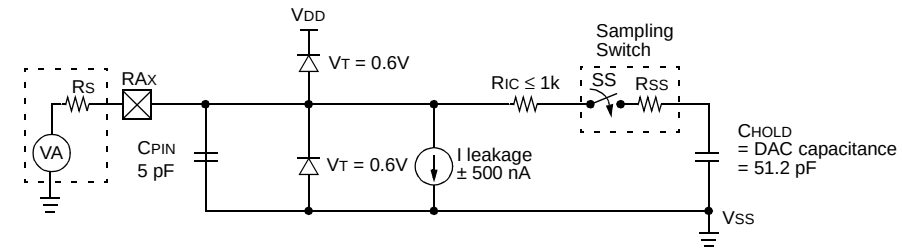
TACQ = Internal Amplifier Settling Time +
Holding Capacitor Charging Time +
Temperature Coefficient

TACQ = 5 µs + Tc + [(Temp - 25°C)(0.05 µs/°C)]

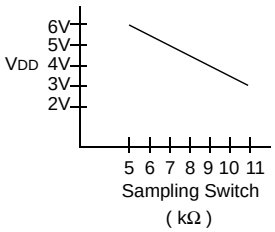
Tc = -CHOLD (RIC + RSS + RS) ln(1/512)
-51.2 pF (1 kΩ + 7 kΩ + 10 kΩ) ln(0.0020)
-51.2 pF (18 kΩ) ln(0.0020)
-0.921 µs (-6.2146)
5.724 µs

TACQ = 5 µs + 5.724 µs + [(50°C - 25°C)(0.05 µs/°C)]
10.724 µs + 1.25 µs
11.974 µs

**FIGURE 8-2: ANALOG INPUT MODEL**




Legend: CPIN = input capacitance
VT = threshold voltage
I leakage = leakage current at the pin due to various junctions
RIC = interconnect resistance
SS = sampling switch
CHOLD = sample/hold capacitance (from DAC)



### 8.2 Selecting the A/D Conversion Clock

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

- 2TOSC
- 8TOSC
- 32TOSC
- Internal ADC RC oscillator

For correct A/D conversions, the A/D conversion clock (TAD) must be selected to ensure a minimum TAD time of 1.6 µs. If the minimum TAD time of 1.6 µs can not be obtained, TAD should be ≤ 8 µs for preferred operation.

Table 8-1 shows the resultant TAD times derived from the device operating frequencies and the A/D clock source selected.

### 8.3 Configuring Analog Port Pins

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

The A/D operation is independent of the state of the CHS<2:0> bits and the TRIS bits.

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

**2:** Analog levels on any pin that is defined as a digital input (including the AN<3:0> pins) may cause the input buffer to consume current that is out of the devices specification.

**TABLE 8-1: TAD vs. DEVICE OPERATING FREQUENCIES**

| AD Clock Source (TAD) | | Device Frequency | | |
|---|---|---|---|---|
| **Operation** | **ADCS<1:0>** | **4 MHz** | **1.25 MHz** | **333.33 kHz** |
| 2TOSC | 00 | 500 ns(2) | 1.6 µs | 6 µs |
| 8TOSC | 01 | 2.0 µs | 6.4 µs | 24 µs(3) |
| 32TOSC | 10 | 8.0 µs | 25.6 µs(3) | 96 µs(3) |
| Internal ADC RC Oscillator(5) | 11 | 2 - 6 µs(1,4) | 2 - 6 µs(1,4) | 2 - 6 µs(1) |

**Note 1:** The RC source has a typical TAD time of 4 µs.
**2:** These values violate the minimum required TAD time.
**3:** For faster conversion times, the selection of another clock source is recommended.
**4:** While in RC mode, with device frequency above 1 MHz, conversion accuracy is out of specification.
**5:** For extended voltage devices (LC), please refer to Electrical Specifications section.

## 8.4 A/D Conversions

Example 8-2 shows how to perform an A/D conversion. The GPIO pins are configured as analog inputs. The analog reference (VREF) is the device VDD. The A/D interrupt is enabled and the A/D conversion clock is FRC. The conversion is performed on the GP0 channel.

| **Note:** | The GO/$\overline{\text{DONE}}$ bit should **NOT** be set in the same instruction that turns on the A/D. |
|---|---|

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

**EXAMPLE 8-2: DOING AN A/D CONVERSION**

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

## 8.5 A/D Operation During SLEEP

The A/D module can operate during SLEEP mode. This requires that the A/D clock source be set to RC (ADCS<1:0> = 11). When the RC clock source is selected, the A/D module waits one instruction cycle before starting the conversion. This allows the SLEEP instruction to be executed, which eliminates all digital switching noise from the conversion. When the conversion is completed, the GO/DONE bit will be cleared, and the result loaded into the ADRES Register. If the A/D interrupt is enabled, the device will wake-up from SLEEP. If the A/D interrupt is not enabled, the A/D module will then be turned off, although the ADON bit will remain set.

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

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

| **Note:** | For the A/D module to operate in SLEEP, the A/D clock source must be set to RC (ADCS<1:0> = 11). To perform an A/D conversion in SLEEP, the GO/DONE bit must be set, followed by the SLEEP instruction. |
|---|---|

## 8.6 A/D Accuracy/Error

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

The maximum pin leakage current is specified in the Device Data Sheet electrical specification, parameter #D060.

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

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

## 8.7 Effects of a RESET

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

## 8.8 Connection Considerations

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

| **Note:** | For the PIC16C433, care must be taken when using the GP4 pin in A/D conversions due to its proximity to the OSC1 pin. |
|---|---|

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

## 8.9 Transfer Function

The ideal transfer function of the A/D converter is as follows: the first transition occurs when the analog input voltage ($V_{AIN}$) is 1 LSb (or Analog $V_{REF}$/256) (Figure 8-3).

**FIGURE 8-3: A/D TRANSFER FUNCTION**

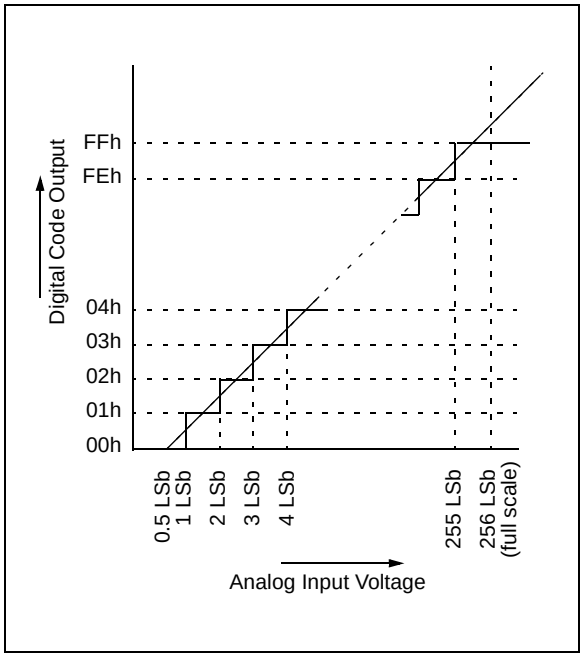


**FIGURE 8-4: FLOW CHART OF A/D OPERATION**

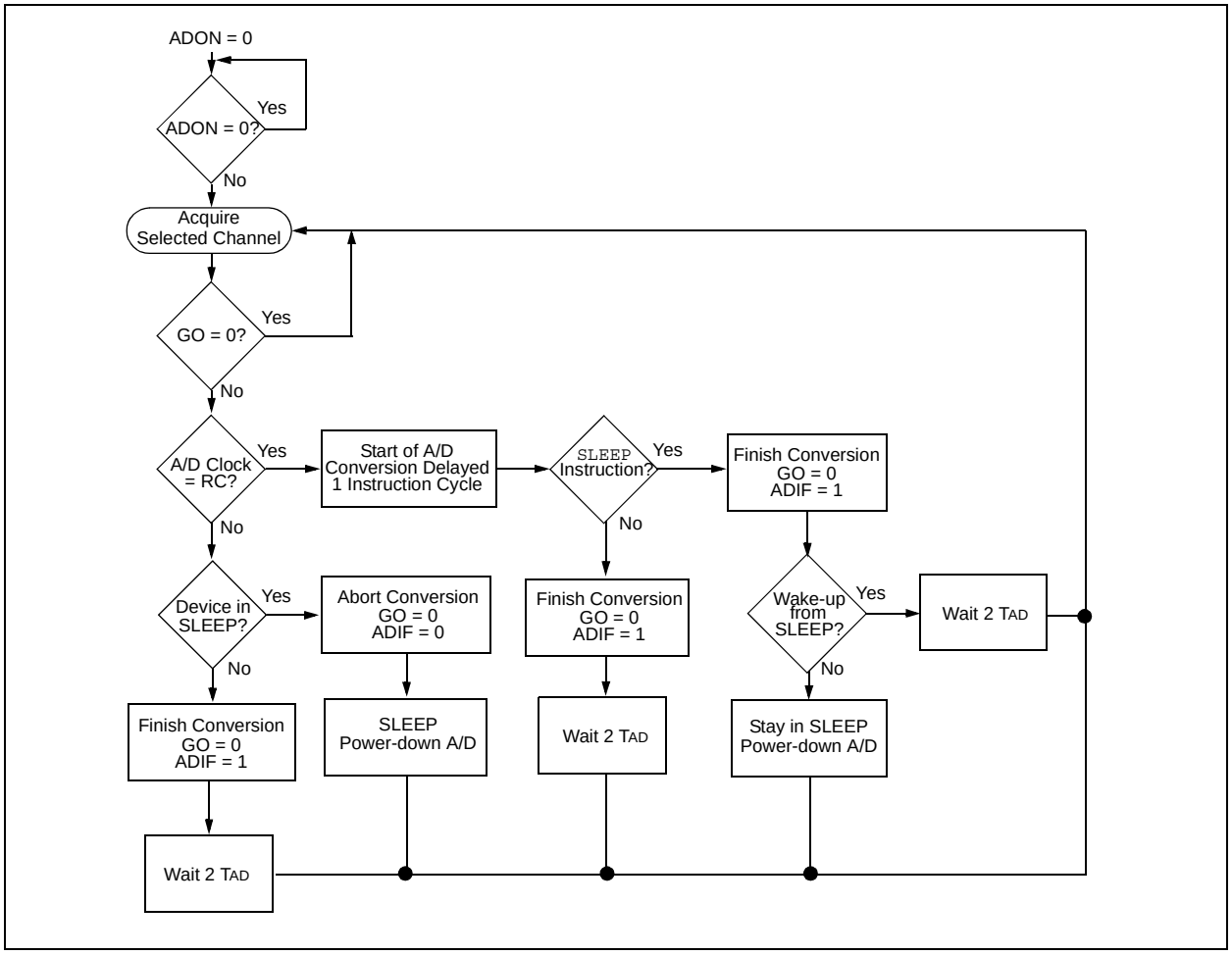



**TABLE 8-2: SUMMARY OF A/D REGISTERS**

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on Power-on Reset | Value on all other RESETS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0Bh/8Bh | INTCON(1) | GIE | PEIE | T0IE | INTE | GPIE | T0IF | INTF | GPIF | 0000 000x | 0000 000u |
| 0Ch | PIR1 | — | ADIF | — | — | — | — | — | — | -0-- ---- | -0-- ---- |
| 8Ch | PIE1 | — | ADIE | — | — | — | — | — | — | -0-- ---- | -0-- ---- |
| 1Eh | ADRES | A/D Result Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 1Fh | ADCON0 | ADCS1 | ADCS0 | reserved | CHS1 | CHS0 | GO/$\overline{\text{DONE}}$ | reserved | ADON | 0000 0000 | 0000 0000 |
| 9Fh | ADCON1 | — | — | — | — | — | PCFG2 | PCFG1 | PCFG0 | ---- -000 | ---- -000 |
| 05h | GPIO | LINTX | LINRX | GP5 | GP4 | GP3 | GP2 | GP1 | GP0 | 11xx xxxx | 11uu uuuu |
| 85h | TRIS | — | — | TRIS5 | TRIS4 | TRIS3 | TRIS2 | TRIS1 | TRIS0 | --11 1111 | --11 1111 |

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used for A/D conversion.
**Note 1:** These registers can be addressed from either bank.

## 9.0 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 PIC16C433 device has a host of such features intended to maximize system reliability, minimize cost through elimination of external components, provide power saving operating modes and offer code protection. These are:

- Oscillator Selection
- RESET
  - Power-on Reset (POR)
  - Power-up Timer (PWRT)
  - Oscillator Start-up Timer (OST)
- Interrupts
- Watchdog Timer (WDT)
- SLEEP
- Code Protection
- ID Locations
- In-Circuit Serial Programming

The PIC16C433 has a Watchdog Timer, which can be shut-off only through configuration bits. It runs off its own RC oscillator for added reliability. There are two timers that offer necessary delays on power-up. One is the Oscillator Start-up Timer (OST), intended to keep the chip in RESET until the crystal oscillator is stable. The other is the Power-up Timer (PWRT), which provides a fixed delay of 72 ms (nominal) on power-up only, designed to keep the part in RESET while the power supply stabilizes. With these two timers on-chip, most applications need no external RESET circuitry.

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

### 9.1 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.

**REGISTER 9-1: CONFIGURATION WORD**

| CP1 | CP0 | CP1 | CP0 | CP1 | CP0 | MCLRE | CP1 | CP0 | $\overline{\text{PWRTE}}$ | WDTE | FOSC2 | FOSC1 | FOSC0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| bit13 | | | | | | | | | | | | | bit0 |

Register: CONFIG
Address 2007h

bit 13-8, 6-5: **CP<1:0>:** Code Protection bit pairs(1)
11 = Code protection off
10 = Locations 400h through 7FEh code protected
01 = Locations 200h through 7FEh code protected
00 = All memory is code protected

bit 7: **MCLRE:** Master Clear Reset Enable bit
1 = Master Clear enabled
0 = Master Clear disabled

bit 4: **$\overline{\text{PWRTE}}$:** Power-up Timer Enable bit
1 = PWRT disabled
0 = PWRT enabled

bit 3: **WDTE:** Watchdog Timer Enable bit
1 = WDT enabled
0 = WDT disabled

bit 2-0: **FOSC<2:0>:** Oscillator Selection bits
111 = EXTRC, clockout on OSC2
110 = EXTRC, OSC2 is I/O
101 = INTRC, clockout on OSC2
100 = INTRC, OSC2 is I/O
011 = Invalid selection
010 = HS oscillator
001 = XT oscillator
000 = LP oscillator

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

## 9.2 Oscillator Configurations

### 9.2.1 OSCILLATOR TYPES

The PIC16C433 can be operated in seven different oscillator modes. The user can program three configuration bits (FOSC<2:0>) to select one of these seven modes:

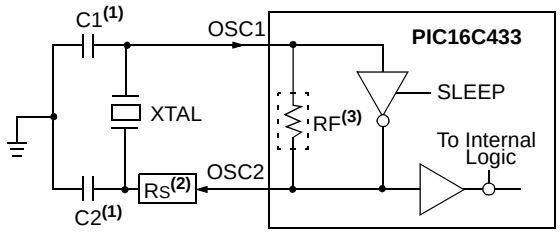
- LP: Low Power Crystal
- HS: High Speed Crystal/Resonator
- XT: Crystal/Resonator
- INTRC*: Internal 4 MHz Oscillator
- EXTRC*: External Resistor/Capacitor

*Can be configured to support CLKOUT

### 9.2.2 CRYSTAL OSCILLATOR/CERAMIC RESONATORS

In XT, HS or LP modes, a crystal or ceramic resonator is connected to the GP5/OSC1/CLKIN and GP4/OSC2 pins to establish oscillation (Figure 9-1). The PIC16C433 oscillator design requires the use of a parallel cut crystal. Use of a series cut crystal may give a frequency out of the crystal manufacturers specifications. When in XT, HS or LP modes, the device can have an external clock source drive the GP5/OSC1/CLKIN pin (Figure 9-2).

**FIGURE 9-1: CRYSTAL OPERATION OR CERAMIC RESONATOR (XT, HS OR LP OSC CONFIGURATION)**




**Note 1:** See Capacitor Selection tables for recommended values of C1 and C2.
**2:** A series resistor (RS) may be required for AT strip cut crystals.
**3:** RF varies with the oscillator mode selected (approx. value = 10 MΩ).

**FIGURE 9-2: EXTERNAL CLOCK INPUT OPERATION (XT, HS OR LP OSC CONFIGURATION)**

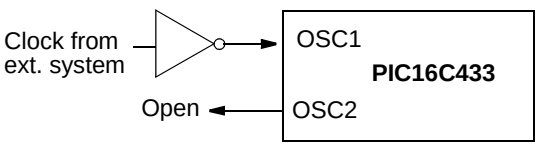



**TABLE 9-1: CAPACITOR SELECTION FOR CERAMIC RESONATORS - PIC16C433**

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

These values are for design guidance only. Since each resonator has its own characteristics, the user should consult the resonator manufacturer for appropriate values of external components.

**TABLE 9-2: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR - PIC16C433**

| Osc Type | Resonator Freq | Cap. Range C1 | Cap. Range C2 |
|---|---|---|---|
| LP | 32 kHz(1) | 15 pF | 15 pF |
| | 100 kHz | 15-30 pF | 30-47 pF |
| | 200 kHz | 15-30 pF | 15-82 pF |
| XT | 100 kHz | 15-30 pF | 200-300 pF |
| | 200 kHz | 15-30 pF | 100-200 pF |
| | 455 kHz | 15-30 pF | 15-100 pF |
| | 1 MHz | 15-30 pF | 15-30 pF |
| | 2 MHz | 15-30 pF | 15-30 pF |
| | 4 MHz | 15-47 pF | 15-47 pF |
| HS | 4 MHz | 15-30 pF | 15-30 pF |
| | 8 MHz | 15-30 pF | 15-30 pF |
| | 10 MHz | 15-30 pF | 15-30 pF |

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

These values are for design guidance only. RS may be required in HS mode, as well as XT mode, to avoid overdriving crystals with low drive level specification. Since each crystal has its own characteristics, the user should consult the crystal manufacturer for appropriate values of external components.



### 9.2.3 EXTERNAL CRYSTAL OSCILLATOR CIRCUIT

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

Figure 9-3 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-degree 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 circuit could be used for external oscillator designs.

**FIGURE 9-3: EXTERNAL PARALLEL RESONANT CRYSTAL OSCILLATOR CIRCUIT**

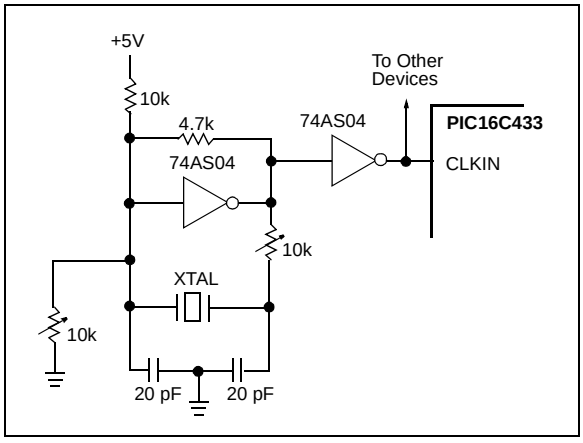
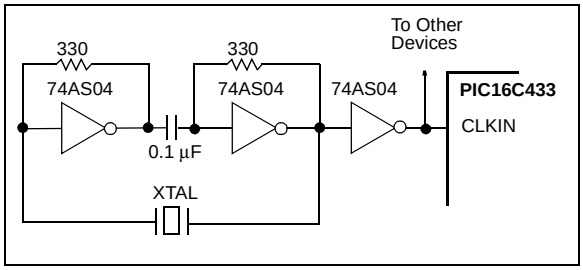



Figure 9-4 shows a series resonant oscillator circuit. This circuit is also designed to use the fundamental frequency of the crystal. The inverter performs a 180-degree phase shift in a series resonant oscillator circuit. The 330 Ω resistors provide the negative feedback to bias the inverters in their linear region.

**FIGURE 9-4: EXTERNAL SERIES RESONANT CRYSTAL OSCILLATOR CIRCUIT**




### 9.2.4 EXTERNAL RC OSCILLATOR

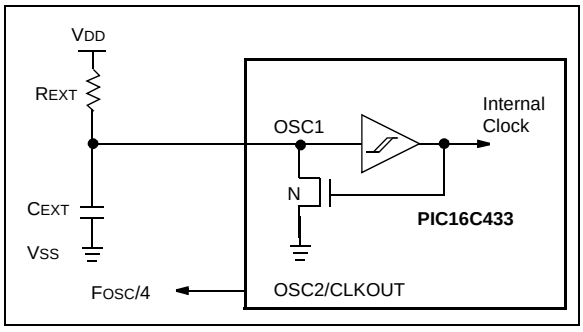
For timing insensitive applications, the RC device option offers additional cost savings. The RC oscillator frequency is a function of the supply voltage, the resistor (REXT) and capacitor (CEXT) values, and the operating temperature. In addition to this, the oscillator frequency will vary from unit to unit, due to normal process parameter variation. Furthermore, the difference in lead frame capacitance between package types will also affect the oscillation frequency, especially for low CEXT values. The user also needs to take into account variation due to tolerance of external R and C components used.

Figure 9-5 shows how the R/C combination is connected to the PIC16C433. For REXT values below 2.2 kΩ, the oscillator operation may become unstable or stop completely. For very high REXT values (i.e., 1 MΩ), the oscillator becomes sensitive to noise, humidity and leakage. Thus, we recommend keeping REXT between 3 kΩ and 100 kΩ.

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

The variation is greater for larger R (since leakage current variation will affect RC frequency more for large R) and for smaller C (since variation of input capacitance will affect RC frequency more).

**FIGURE 9-5: EXTERNAL RC OSCILLATOR MODE**



### 9.2.5 INTERNAL 4 MHz RC OSCILLATOR

The internal RC oscillator provides a fixed 4 MHz (nominal) system clock at $V_{DD}$ = 5V and 25°C. See Section 13.0 for information on variation over voltage and temperature.

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

OSCCAL, when written to with the calibration value, will "trim" the internal oscillator to remove process variation from the oscillator frequency. Bits <7:4>, CAL<3:0> are used for fine calibration, while bit3, CALFST, and bit2, CALSLW, are used for more coarse adjustment. Adjusting CAL<3:0> from `0000` to `1111` yields a higher clock speed. Set CALFST = `1` for greater increase in frequency, or set CALSLW = `1` for greater decrease in frequency. Note that bits 1 and 0 of OSCCAL are unimplemented and should be written as 0, when modifying OSCCAL for compatibility with future devices.

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

### 9.2.6 CLKOUT

The PIC16C433 can be configured to provide a clock out signal (CLKOUT) on pin 3, when the configuration word address (2007h) is programmed with FOSC2, FOSC1, and FOSC0, equal to `101` for INTRC or `111` for EXTRC. The oscillator frequency, divided by 4, can be used for test purposes or to synchronize other logic.

## 9.3 RESET

The PIC16C433 differentiates between various kinds of RESET:

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

Some registers are not affected in any RESET condition; their status is unknown on POR and unchanged in any other RESET. Most other registers are reset to a "RESET state" on Power-on Reset (POR), $\overline{MCLR}$ Reset, WDT Reset, and $\overline{MCLR}$ Reset during SLEEP. They are not affected by a WDT Wake-up, which is viewed as the resumption of normal operation. The $\overline{TO}$ and $\overline{PD}$ bits are set or cleared differently in different RESET situations, as indicated in Table 9-5. These bits are used in software to determine the nature of the RESET. See Table 9-6 for a full description of RESET states of all registers.

A simplified block diagram of the On-Chip Reset circuit is shown in Figure 9-6.

The PIC16C433 has a $\overline{MCLR}$ noise filter in the $\overline{MCLR}$ Reset path. The filter will detect and ignore small pulses.

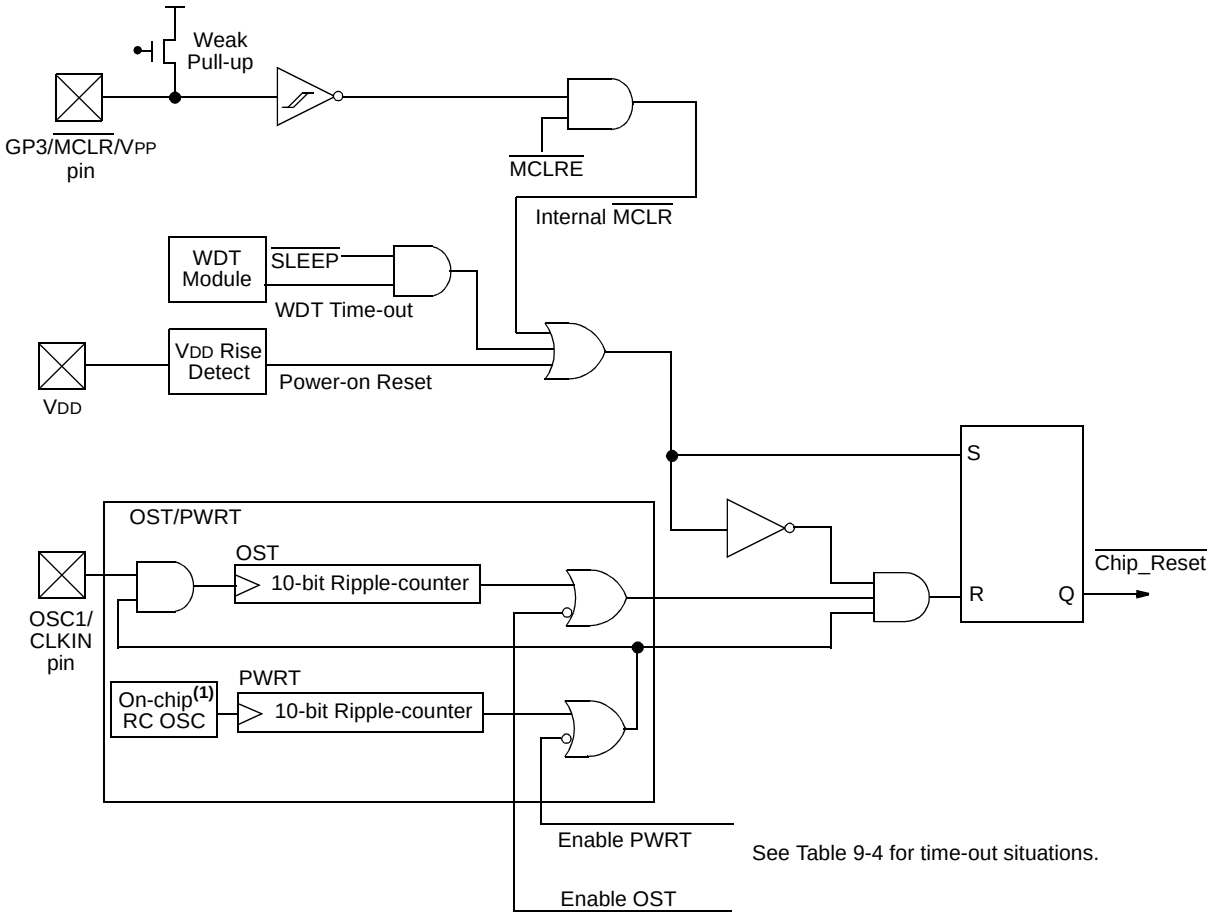
It should be noted that a WDT Reset does not drive $\overline{MCLR}$ pin low.

When $\overline{MCLR}$ is asserted, the state of the OSC1/CLKIN and CLKOUT/OSC2 pins are as follows:

**TABLE 9-3: CLKIN/$\overline{CLKOUT}$ PIN STATES WHEN MCLR ASSERTED**

| Oscillator Mode | OSC1/CLKIN Pin | OSC2/CLKOUT Pin |
|---|---|---|
| EXTRC, CLKOUT on OSC2 | OSC1 pin is tri-stated and driven by external circuit | OSC2 pin is driven low |
| EXTRC, OSC2 is I/O | OSC1 pin is tri-stated and driven by external circuit | OSC2 pin is tri-state input |
| INTRC, CLKOUT on OSC2 | OSC1 pin is tri-state input | OSC2 pin is driven low |
| INTRC, OSC2 is I/O | OSC1 pin is tri-state input | OSC2 pin is tri-state input |

**FIGURE 9-6: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT**




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

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

### 9.4.1 POWER-ON RESET (POR)

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

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

For additional information, refer to Application Note AN607, "*Power-up Trouble Shooting*."

### 9.4.2 POWER-UP TIMER (PWRT)

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

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

### 9.4.3 OSCILLATOR START-UP TIMER (OST)

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

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

### 9.4.4 TIME-OUT SEQUENCE

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

Since the time-outs occur from the POR pulse, if MCLR is kept low long enough, the time-outs will expire. Then bringing MCLR high will begin execution immediately (Figure 9-9). This is useful for testing purposes, or to synchronize more than one PIC16C433 device operating in parallel.

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

The Power Control/Status Register, PCON (address 8Eh), has one bit.

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

**TABLE 9-4: TIME-OUT IN VARIOUS SITUATIONS**

| Oscillator Configuration | Power-up | | Wake-up from SLEEP |
|---|---|---|---|
| | PWRTE = 0 | PWRTE = 1 | |
| XT, HS, LP | 72 ms + 1024TOSC | 1024TOSC | 1024TOSC |
| INTRC, EXTRC | 72 ms | — | — |

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

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

Legend: u = unchanged, x = unknown

**TABLE 9-6: RESET CONDITION FOR SPECIAL REGISTERS**

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

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

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

**TABLE 9-7: INITIALIZATION CONDITIONS FOR ALL REGISTERS**

| Register | Power-on Reset | $\overline{MCLR}$ Reset WDT Reset | Wake-up via WDT or Interrupt |
|---|---|---|---|
| W | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| INDF | 0000 0000 | 0000 0000 | 0000 0000 |
| TMR0 | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| PCL | 0000 0000 | 0000 0000 | PC + 1(2) |
| STATUS | 0001 1xxx | 000q quuu(3) | uuuq quuu(3) |
| FSR | xxxx xxxx | uuuu uuuu | uuuu uuuu |
| GPIO | 11xx xxxx | 11uu uuuu | 11uu uuuu |
| PCLATH | ---0 0000 | ---0 0000 | ---u uuuu |
| INTCON | 0000 000x | 0000 000u | uuuu uqqq(1) |
| PIR1 | -0-- ---- | -0-- ---- | -q-- ----(4) |
| ADCON0 | 0000 0000 | 0000 0000 | uuuu uquu(5) |
| OPTION | 1111 1111 | 1111 1111 | uuuu uuuu |
| TRIS | --11 1111 | --11 1111 | --uu uuuu |
| PIE1 | -0-- ---- | -0-- ---- | -u-- ---- |
| PCON | ---- --0- | ---- --u- | ---- --u- |
| OSCCAL | 0111 00-- | uuuu uu-- | uuuu uu-- |
| ADCON1 | ---- -000 | ---- -000 | ---- -uuu |

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

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

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

**3:** See Table 9-5 for RESET value for specific condition.

**4:** If wake-up was due to A/D completing then bit 6 = 1, all other interrupts generating a wake-up will cause bit 6 = u.

**5:** If wake-up was due to A/D completing then bit 3 = 0, all other interrupts generating a wake-up will cause bit 3 = u.

**FIGURE 9-7: TIME-OUT SEQUENCE ON POWER-UP (MCLR NOT TIED TO VDD): CASE 1**

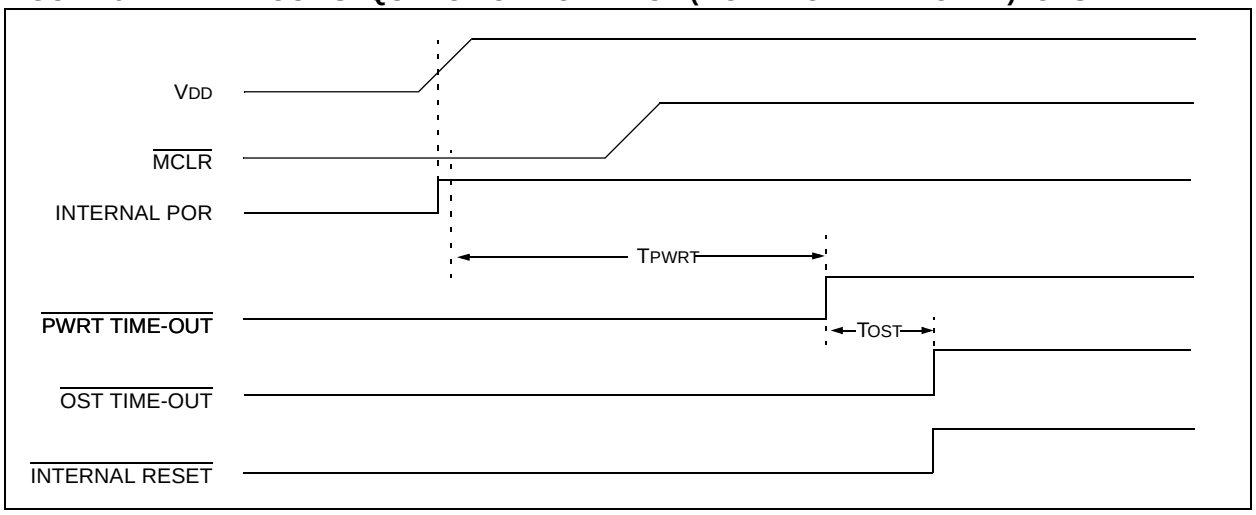



**FIGURE 9-8: TIME-OUT SEQUENCE ON POWER-UP (MCLR NOT TIED TO VDD): CASE 2**

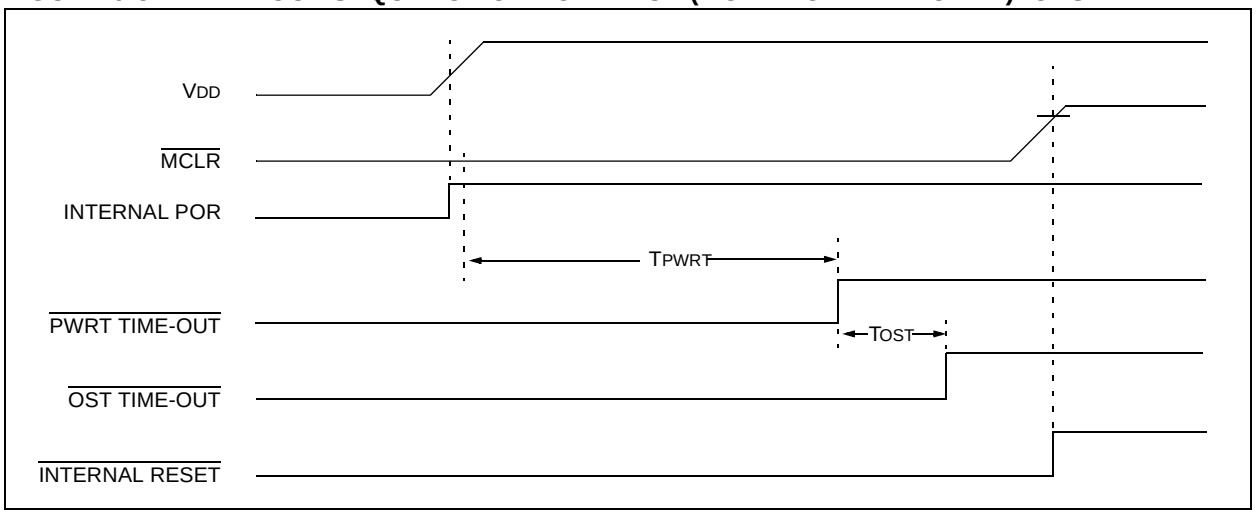



**FIGURE 9-9: TIME-OUT SEQUENCE ON POWER-UP (MCLR TIED TO VDD)**

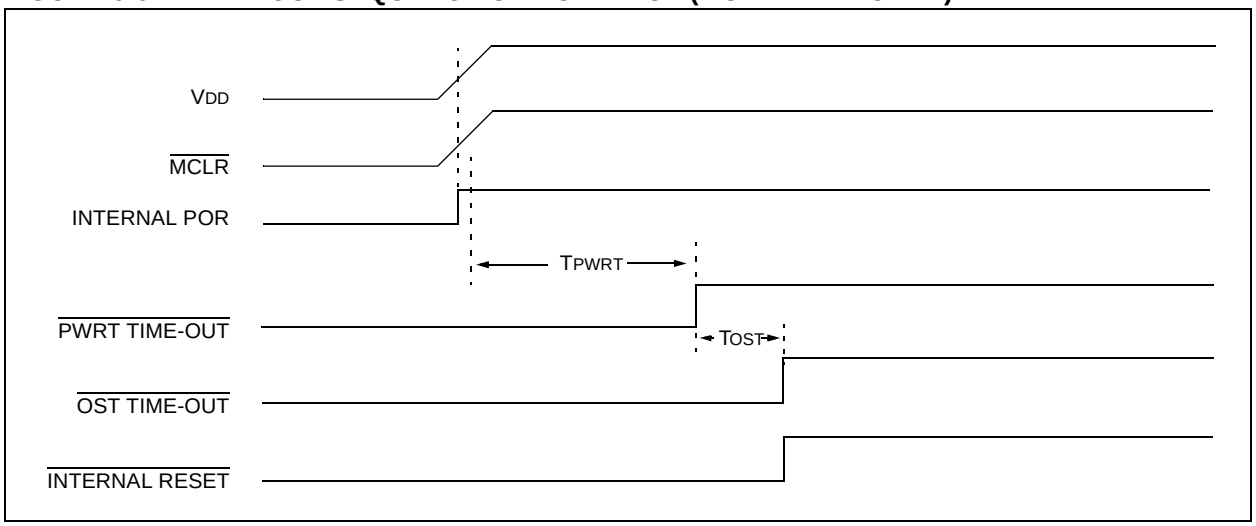
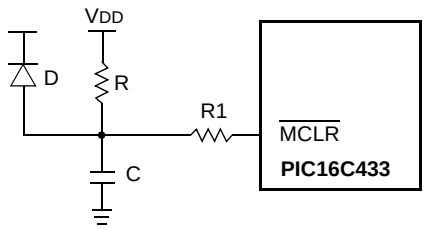


**FIGURE 9-10: EXTERNAL POWER-ON RESET CIRCUIT (FOR SLOW VDD POWER-UP)**

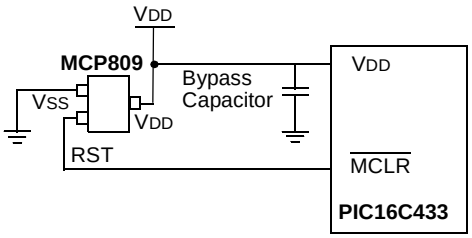



**Note 1:** External Power-on Reset circuit is required only if VDD power-up slope is too slow. The diode D helps discharge the capacitor quickly when VDD powers down.

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

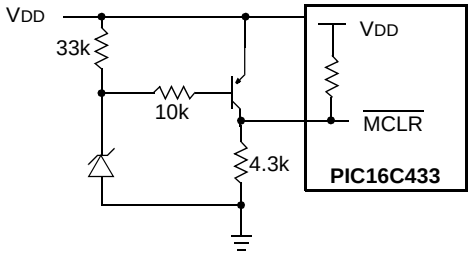
**3:** R1 = 100Ω to 1 kΩ will limit any current flowing into MCLR from external capacitor C, in the event of MCLR/VPP pin breakdown due to Electrostatic Discharge (ESD), or Electrical Overstress (EOS).

**FIGURE 9-11: EXTERNAL BROWN-OUT PROTECTION CIRCUIT 3**




This brown-out protection circuit employs Microchip Technology's MCP809 microcontroller supervisor. The MCP8XX and MCP1XX families of supervisors provide push-pull and open collector outputs, with both high and low active RESET pins. There are 7 different trip point selections to accommodate 5V and 3V systems.

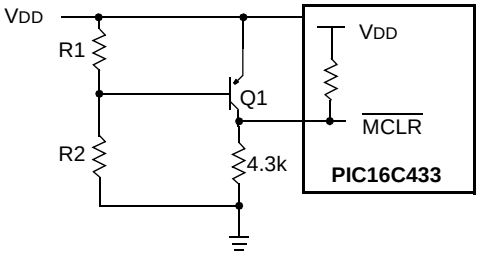
**FIGURE 9-12: EXTERNAL BROWN-OUT PROTECTION CIRCUIT 1**




**Note 1:** This circuit will activate RESET when VDD goes below (Vz + 0.7V), where Vz = Zener voltage.

**2:** Resistors should be adjusted for the characteristics of the transistor.

**FIGURE 9-13: EXTERNAL BROWN-OUT PROTECTION CIRCUIT 2**




**Note 1:** This brown-out circuit is less expensive, albeit less accurate. Transistor Q1 turns off when VDD is below a certain level, such that:

$$V_{DD} \cdot \frac{R1}{R1 + R2} = 0.7V$$

**2:** Resistors should be adjusted for the characteristics of the transistor.

## 9.5 Interrupts

There are four sources of interrupt:

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

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

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

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

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

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

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

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

For external interrupt events, such as GPIO change interrupt, the interrupt latency will be three or four instruction cycles. The exact latency depends on when the interrupt event occurs (Figure 9-15). The latency is the same for one or two cycle instructions. Individual interrupt flag bits are set, regardless of the status of their corresponding mask bit or the GIE bit.

**FIGURE 9-14: INTERRUPT LOGIC**

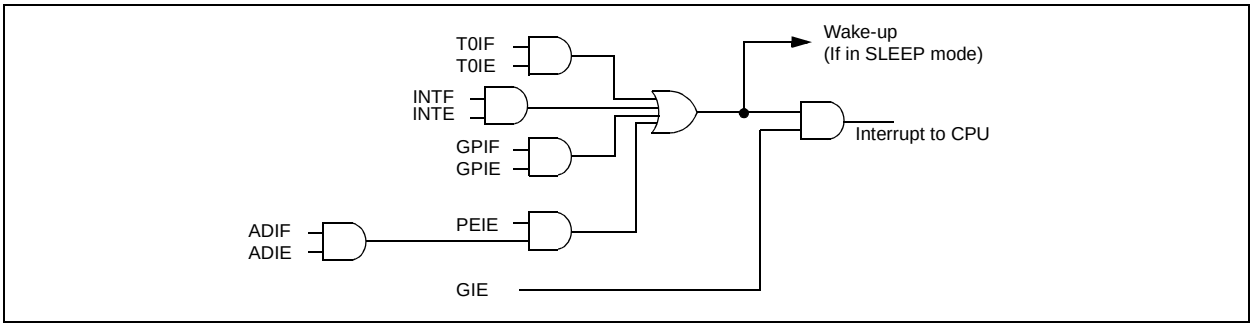
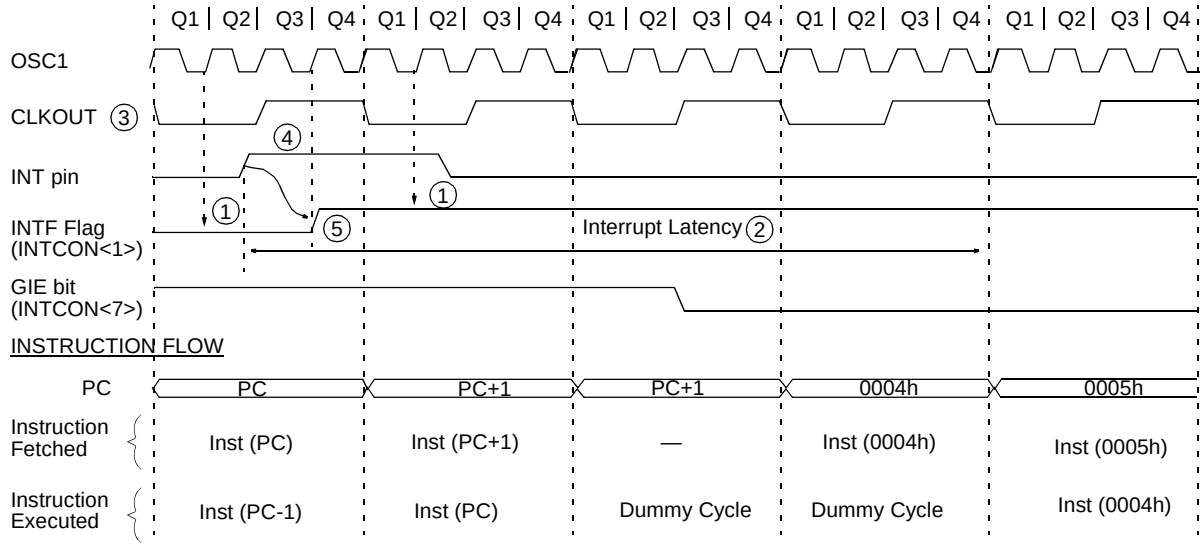


**FIGURE 9-15: INT PIN INTERRUPT TIMING**




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

### 9.5.1 TMR0 INTERRUPT

An overflow (FFh → 00h) in the TMR0 register will set flag bit T0IF (INTCON<2>). The interrupt can be enabled/disabled by setting/clearing enable bit T0IE (INTCON<5>) (see Section 7.0). The flag bit T0IF (INTCON<2>) will be set, regardless of the state of the enable bits. If used, this flag must be cleared in software.

### 9.5.2 INT INTERRUPT

External interrupt on GP2/INT pin is edge triggered; either rising if bit INTEDG (OPTION<6>) is set, or falling, if the INTEDG bit is clear. When a valid edge appears on the GP2/INT pin, flag bit INTF (INTCON<1>) is set. This interrupt can be disabled by clearing enable bit INTE (INTCON<4>). Flag bit INTF must be cleared in software in the Interrupt Service Routine before re-enabling this interrupt. The INT interrupt can wake-up the processor from SLEEP, if bit INTE was set prior to going into SLEEP. The status of global interrupt enable bit GIE decides whether or not the processor branches to the interrupt vector following wake-up. See Section 9.8 for details on SLEEP mode.

### 9.5.3 GPIO INTCON CHANGE

An input change on GP3, GP1 or GP0 sets flag bit GPIF (INTCON<0>). The interrupt can be enabled/disabled by setting/clearing enable bit GPIE (INTCON<3>) (Section 5.1) . This flag bit GPIF (INTCON<0>) will be set, regardless of the state of the enable bits. If used, this flag must be cleared in software.

## 9.6 Context Saving During Interrupts

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

Example 9-1 shows the storing and restoring of the STATUS and W registers. The register, W_TEMP, must be defined in both banks and must be defined at the same offset from the bank base address (i.e., if W_TEMP is defined at 0x20 in bank 0, it must also be defined at 0xA0 in bank 1).

Example 9-2 shows the saving and restoring of STATUS and W using RAM locations 0x70 - 0x7F. W_TEMP is defined at 0x70 and STATUS_TEMP is defined at 0x71.

The example:

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

**EXAMPLE 9-1: SAVING STATUS AND W REGISTERS USING GENERAL PURPOSE RAM (0x20 - 0x6F)**

```
MOVWF    W_TEMP           ;Copy W to TEMP register, could be bank one or zero
SWAPF    STATUS,W         ;Swap status to be saved into W
BCF      STATUS,RP0       ;Change to bank zero, regardless of current bank
MOVWF    STATUS_TEMP      ;Save status to bank zero STATUS_TEMP register
:
:(ISR)
:
SWAPF    STATUS_TEMP,W    ;Swap STATUS_TEMP register into W
                          ;(sets bank to original state)
MOVWF    STATUS           ;Move W into STATUS register
SWAPF    W_TEMP,F         ;Swap W_TEMP
SWAPF    W_TEMP,W         ;Swap W_TEMP into W
RETFIE                    ;Return from interrupt
```

**EXAMPLE 9-2: SAVING STATUS AND W REGISTERS USING SHARED RAM (0x70 - 0x7F)**

```
MOVWF    W_TEMP           ;Copy W to TEMP register (bank independent)
MOVF     STATUS,W         ;Move STATUS register into W
MOVWF    STATUS_TEMP      ;Save contents of STATUS register
:
:(ISR)
:
MOVF     STATUS_TEMP,W    ;Retrieve copy of STATUS register
MOVWF    STATUS           ;Restore pre-isr STATUS register contents
SWAPF    W_TEMP,F         ;
SWAPF    W_TEMP,W         ;Restore pre-isr W register contents
RETFIE                    ;Return from interrupt
```

## 9.7 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 OSC1/CLKIN pin. That means that the WDT will run, even if the clock on the OSC1/CLKIN and OSC2/CLKOUT pins of the device has been stopped, for example, by execution of a SLEEP instruction. During normal operation, a WDT time-out generates a device RESET (Watchdog Timer Reset). If the device is in SLEEP mode, a WDT time-out causes the device to wake-up and continue with normal operation (Watchdog Timer Wake-up). The WDT can be permanently disabled by clearing configuration bit WDTE (Section 9.1).

### 9.7.1 WDT PERIOD

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

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

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

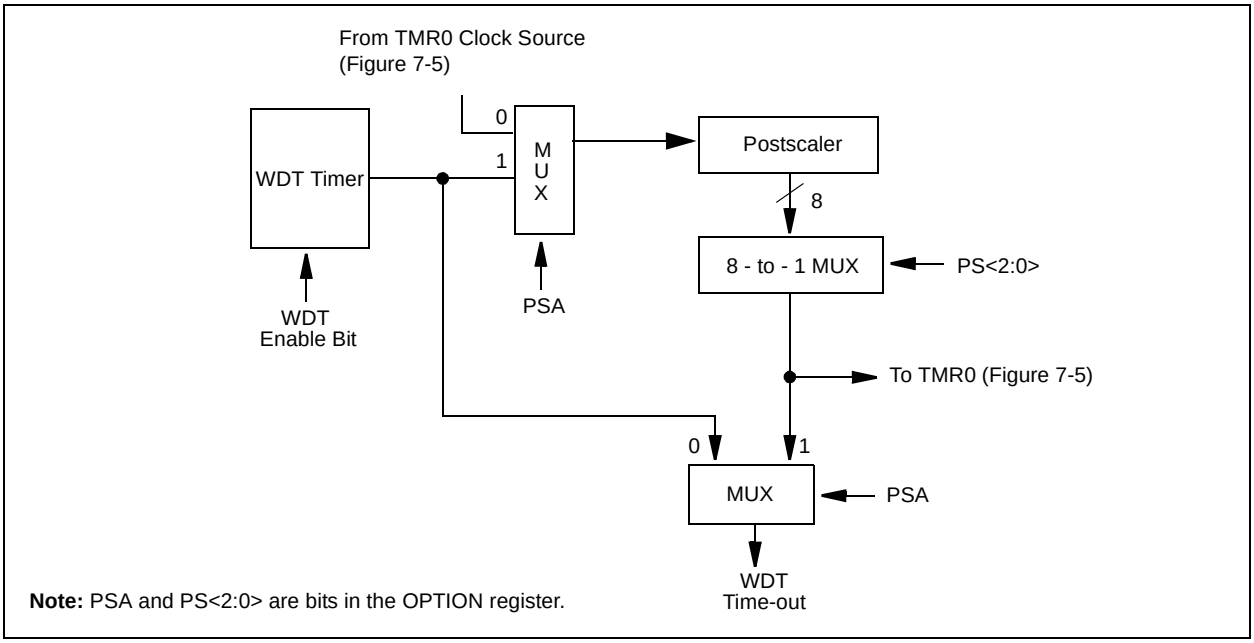
### 9.7.2 WDT PROGRAMMING CONSIDERATIONS

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

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

See Example 7-1 and Example 7-2 for changing prescaler between WDT and Timer0.

**FIGURE 9-16: WATCHDOG TIMER BLOCK DIAGRAM**




**Note:** PSA and PS<2:0> are bits in the OPTION register.

**TABLE 9-8: SUMMARY OF WATCHDOG TIMER REGISTERS**

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

Legend: Shaded cells are not used by the Watchdog Timer.
**Note 1:** See Register 9-1 for operation of these bits. Not all CP0 and CP1 bits are shown.

## 9.8 Power-down Mode (SLEEP)

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

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

For lowest current consumption in this mode, place all I/O pins at either VDD or VSS, ensure no external circuitry is drawing current from the I/O pin, power-down the A/D, and disable external clocks. Pull all I/O pins that are hi-impedance inputs, high or low externally, to avoid switching currents caused by floating inputs. The T0CKI input, if enabled, should also be at VDD or VSS, for lowest current consumption. The contribution from on-chip pull-ups on GPIO should be considered.

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

### 9.8.1 WAKE-UP FROM SLEEP

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

1. External RESET input on $\overline{MCLR}$ pin.
2. Watchdog Timer Wake-up (if WDT was enabled).
3. GP2/INT interrupt, interrupt GPIO port change or some peripheral interrupts.
4. LIN bus activity (connect BACT to GP2/T0CKI/AN2/INT pin).

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

The following peripheral interrupt can wake the device from SLEEP:

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

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

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

### 9.8.2 WAKE-UP USING INTERRUPTS

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

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

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

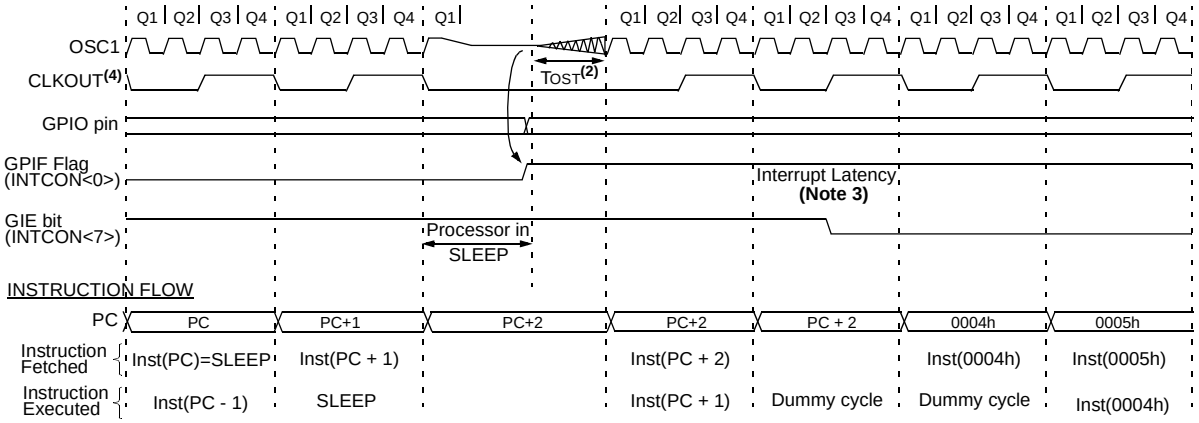
To ensure that the WDT is cleared, a CLRWDT instruction should be executed before a SLEEP instruction.

### 9.8.3 WAKE-UP FROM SLEEP UPON BUS ACTIVITY

The PIC16C433 can be woken up upon bus activity on the LIN bus. This is done by connecting the BACT pin with either GP0, GP1 or GP2. The pin which will be connected to the BACT pin has to be configured to wake the microcontroller up from SLEEP.

**FIGURE 9-17: WAKE-UP FROM SLEEP THROUGH INTERRUPT**




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

## 9.9 Program Verification/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. |
|---|---|

## 9.10 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. It is recommended that only the 4 least significant bits of the ID location are used.

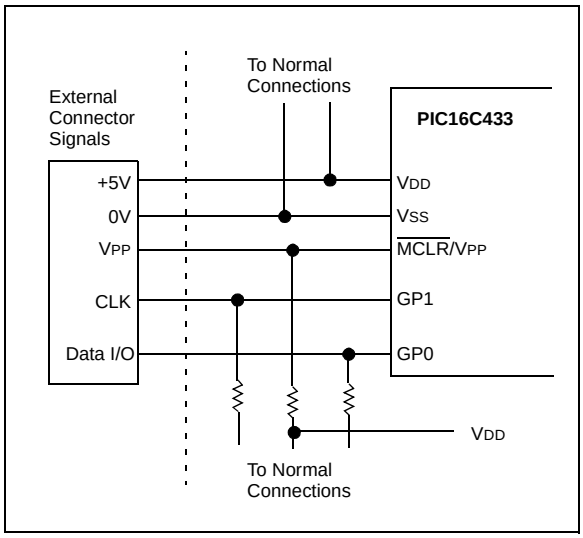
## 9.11 In-Circuit Serial Programming

PIC16C433 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 GP1 and GP0 pins low, while raising the MCLR (VPP) pin from VIL to VIHH (see programming specification). GP1 (clock) becomes the programming clock and GP0 (data) becomes the programming data. Both GP0 and GP1 are Schmitt Trigger inputs in this mode.

After RESET, and if the device is placed 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 PIC16C433 Programming Specifications.

**FIGURE 9-18: TYPICAL IN-CIRCUIT SERIAL PROGRAMMING CONNECTION**



**NOTES:**

## 10.0 INSTRUCTION SET SUMMARY

Each PIC16C433 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 PIC16C433 instruction set summary in Table 10-2 lists **byte-oriented**, **bit-oriented**, and **literal and control** operations. Table 10-1 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 address 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 10-1: 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)<br>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,<br>d = 1: store result in file register f.<br>Default is d = 1. |
| label | Label name |
| TOS | Top-of-Stack |
| PC | Program Counter |
| PCLATH | Program Counter High Latch |
| GIE | Global Interrupt Enable bit |
| WDT | Watchdog Timer/Counter |
| $\overline{TO}$ | Time-out bit |
| $\overline{PD}$ | Power-down bit |
| dest | Destination either the W register or the specified register file location |
| [ ] | Options |
| ( ) | Contents |
| → | Assigned to |
| < > | Register bit field |
| ∈ | In the set of |
| *italics* | User defined term (font is courier) |

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

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

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

Table 10-2 lists the instructions recognized by the MPASM assembler.

Figure 10-1 shows the three general formats that the instructions can have.

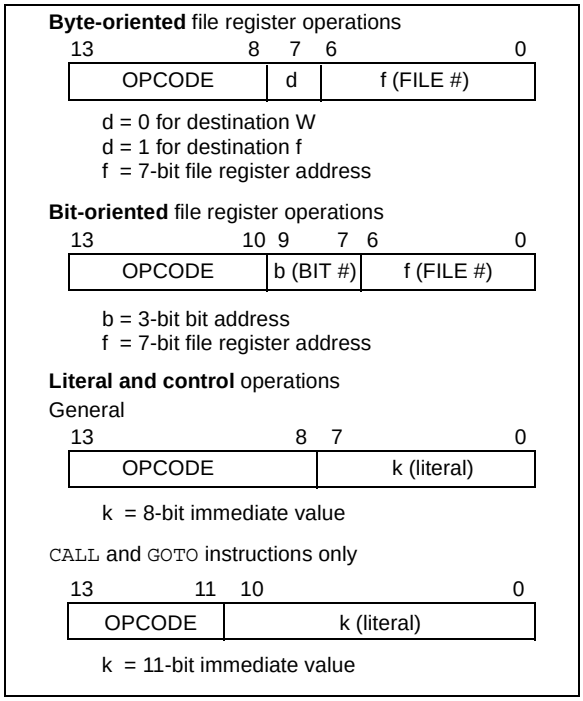
**Note:** To maintain upward compatibility with future PIC16C433 products, do not use the OPTION and TRIS instructions.

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

0xhh

where h signifies a hexadecimal digit.

**FIGURE 10-1: GENERAL FORMAT FOR INSTRUCTIONS**



## 10.1 Special Function Registers as Source/Destination

The PIC16C433's orthogonal instruction set allows read and write of all file registers, including special function registers. There are some special situations specified in the following sections the user should be aware of.

### 10.1.1 STATUS AS DESTINATION

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

### 10.1.2 TRIS AS DESTINATION

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

### 10.1.3 PCL AS SOURCE OR DESTINATION

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

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

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

### 10.1.4 BIT MANIPULATION

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

**TABLE 10-2: INSTRUCTION SET SUMMARY**

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

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

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

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

## 10.2 Instruction Descriptions

**ADDLW** **Add Literal and W**

Syntax: [ *label* ] ADDLW k

Operands: $0 \le k \le 255$

Operation: (W) + k → (W)

Status Affected: C, DC, Z

Encoding:

| | | | |
|---|---|---|---|
| 11 | 111x | kkkk | kkkk |

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

Words: 1

Cycles: 1

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

**ANDLW** **And Literal with W**

Syntax: [ *label* ] ANDLW k

Operands: $0 \le k \le 255$

Operation: (W) .AND. (k) → (W)

Status Affected: Z

Encoding:

| | | | |
|---|---|---|---|
| 11 | 1001 | kkkk | kkkk |

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

Words: 1

Cycles: 1

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

**ADDWF** **Add W and f**

Syntax: [ *label* ] ADDWF f,d

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

Operation: (W) + (f) → (dest)

Status Affected: C, DC, Z

Encoding:

| | | | |
|---|---|---|---|
| 00 | 0111 | dfff | ffff |

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

Words: 1

Cycles: 1

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

**ANDWF** **AND W with f**

Syntax: [ *label* ] ANDWF f,d

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

Operation: (W) .AND. (f) → (dest)

Status Affected: Z

Encoding:

| | | | |
|---|---|---|---|
| 00 | 0101 | dfff | ffff |

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

Words: 1

Cycles: 1

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

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

Syntax: [ *label* ] BCF f,b

Operands: 0 ≤ f ≤ 127
0 ≤ b ≤ 7

Operation: 0 → (f<b>)

Status Affected: None

Encoding:

| 01 | 00bb | bfff | ffff |
|---|---|---|---|

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

Words: 1

Cycles: 1

Example

```
BCF     FLAG_REG, 7
```

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

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

Syntax: [ *label* ] BSF f,b

Operands: 0 ≤ f ≤ 127
0 ≤ b ≤ 7

Operation: 1 → (f<b>)

Status Affected: None

Encoding:

| 01 | 01bb | bfff | ffff |
|---|---|---|---|

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

Words: 1

Cycles: 1

Example

```
BSF     FLAG_REG,   7
```

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

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

Syntax: [ *label* ] BTFSC f,b

Operands: 0 ≤ f ≤ 127
0 ≤ b ≤ 7

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

Status Affected: None

Encoding:

| 01 | 10bb | bfff | ffff |
|---|---|---|---|

Description: If bit 'b' in register 'f' is '0', then the next instruction is skipped.
If bit 'b' is '0', then the next instruction fetched during the current instruction execution is discarded, and a NOP is executed instead, making this a 2 cycle instruction.

Words: 1

Cycles: 1(2)

Example

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

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

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

Syntax: [ *label* ] BTFSS f,b

Operands: 0 ≤ f ≤ 127
0 ≤ b < 7

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

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 01 | 11bb | bfff | ffff |

Description: If bit 'b' in register 'f' is '1', then the next instruction is skipped.
If bit 'b' is '1', then the next instruction fetched during the current instruction execution is discarded and a NOP is executed instead, making this a 2-cycle instruction.

Words: 1

Cycles: 1(2)

Example

```
HERE    BTFSS   FLAG,1
FALSE   GOTO    PROCESS_CODE
TRUE    •
        •
        •
```

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

**CALL** **Call Subroutine**

Syntax: [ *label* ] CALL k

Operands: 0 ≤ k ≤ 2047

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

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 10 | 0kkk | kkkk | kkkk |

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

Words: 1

Cycles: 2

Example

```
HERE    CALL   THERE
```

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

**CLRF** **Clear f**

Syntax: [ *label* ] CLRF f

Operands: 0 ≤ f ≤ 127

Operation: 00h → (f)
1 → Z

Status Affected: Z

Encoding:

| | | | |
|---|---|---|---|
| 00 | 0001 | 1fff | ffff |

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

Words: 1

Cycles: 1

Example

```
CLRF    FLAG_REG
```

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

**CLRW** **Clear W**

Syntax: [ *label* ] CLRW

Operands: None

Operation: 00h → (W)
1 → Z

Status Affected: Z

Encoding:

| | | | |
|---|---|---|---|
| 00 | 0001 | 0000 | 0011 |

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

Words: 1

Cycles: 1

Example

```
CLRW
```

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

**CLRWDT** **Clear Watchdog Timer**

Syntax: [ *label* ] CLRWDT

Operands: None

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

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

Encoding:

| | | | |
|---|---|---|---|
| 00 | 0000 | 0110 | 0100 |

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

Words: 1

Cycles: 1

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

**COMF** **Complement f**

Syntax: [ *label* ] COMF f,d

Operands: 0 ≤ f ≤ 127
d ∈ [0,1]

Operation: ($\bar{f}$) → (dest)

Status Affected: Z

Encoding:

| | | | |
|---|---|---|---|
| 00 | 1001 | dfff | ffff |

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

Words: 1

Cycles: 1

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

**DECF** **Decrement f**

Syntax: [ *label* ] DECF f,d

Operands: 0 ≤ f ≤ 127
d ∈ [0,1]

Operation: (f) - 1 → (dest)

Status Affected: Z

Encoding:

| | | | |
|---|---|---|---|
| 00 | 0011 | dfff | ffff |

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

Words: 1

Cycles: 1

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

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

Syntax: [ *label* ] DECFSZ f,d

Operands: 0 ≤ f ≤ 127
d ∈ [0,1]

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

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 00 | 1011 | dfff | ffff |

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

Words: 1

Cycles: 1(2)

Example
```
HERE     DECFSZ   CNT, 1
         GOTO     LOOP
CONTINUE •
         •
         •
```
Before Instruction
PC = address HERE
After Instruction
CNT = CNT - 1
if CNT = 0,
PC = address CONTINUE
if CNT ¼ 0,
PC = address HERE+1

**GOTO** **Unconditional Branch**

Syntax: [ *label* ] GOTO k

Operands: 0 ≤ k ≤ 2047

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

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 10 | 1kkk | kkkk | kkkk |

Description: GOTO is an unconditional branch. The eleven-bit immediate value is loaded into PC bits <10:0>. The upper bits of PC are loaded from PCLATH<4:3>. GOTO is a two-cycle instruction.

Words: 1

Cycles: 2

Example
```
GOTO THERE
```
After Instruction
PC = Address THERE

**INCF** **Increment f**

Syntax: [ *label* ] INCF f,d

Operands: 0 ≤ f ≤ 127
d ∈ [0,1]

Operation: (f) + 1 → (dest)

Status Affected: Z

Encoding:

| | | | |
|---|---|---|---|
| 00 | 1010 | dfff | ffff |

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

Words: 1

Cycles: 1

Example
```
INCF CNT, 1
```
Before Instruction
CNT = 0xFF
Z = 0
After Instruction
CNT = 0x00
Z = 1

**INCFSZ** **Increment f, Skip if 0**

Syntax: [ *label* ] INCFSZ f,d

Operands: 0 ≤ f ≤ 127
d ∈ [0,1]

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

Status Affected: None

Encoding:

| | | | |
|---|---|---|---|
| 00 | 1111 | dfff | ffff |

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

Words: 1

Cycles: 1(2)

Example
```
HERE     INCFSZ   CNT, 1
         GOTO     LOOP
CONTINUE •
         •
         •
```
Before Instruction
PC = address HERE
After Instruction
CNT = CNT + 1
if CNT= 0,
PC = address CONTINUE
if CNT≠ 0,
PC = address HERE +1

**IORLW** **Inclusive OR Literal with W**

Syntax: [ *label* ] IORLW k

Operands: 0 ≤ k ≤ 255

Operation: (W) .OR. k → (W)

Status Affected: Z

Encoding:

| | | | |
|---|---|---|---|
| 11 | 1000 | kkkk | kkkk |

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

Words: 1

Cycles: 1

Example
```
IORLW 0x35
```
Before Instruction
W = 0x9A
After Instruction
W = 0xBF
Z = 1

**IORWF** **Inclusive OR W with f**

Syntax: [ *label* ] IORWF f,d

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

Operation: (W) .OR. (f) → (dest)

Status Affected: Z

Encoding:

| 00 | 0100 | dfff | ffff |
|---|---|---|---|

Description: Inclusive OR the W register with register 'f'. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'.

Words: 1

Cycles: 1

Example
```
IORWF         RESULT, 0
```
Before Instruction
RESULT = 0x13
W = 0x91
After Instruction
RESULT = 0x13
W = 0x93
Z = 1

**MOVF** **Move f**

Syntax: [ *label* ] MOVF f,d

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

Operation: (f) → (dest)

Status Affected: Z

Encoding:

| 00 | 1000 | dfff | ffff |
|---|---|---|---|

Description: The contents of register f are 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
```
MOVF     FSR, 0
```
After Instruction
W = value in FSR register
Z = 1

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

Syntax: [ *label* ] MOVLW k

Operands: $0 \le k \le 255$

Operation: k → (W)

Status Affected: None

Encoding:

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

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

Words: 1

Cycles: 1

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

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

Syntax: [ *label* ] MOVWF f

Operands: $0 \le f \le 127$

Operation: (W) → (f)

Status Affected: None

Encoding:

| 00 | 0000 | 1fff | ffff |
|---|---|---|---|

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

Words: 1

Cycles: 1

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

**NOP** — **No Operation**

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

**OPTION** — **Load Option Register**

| | |
|---|---|
| Syntax: | [ *label* ] OPTION |
| Operands: | None |
| Operation: | (W) → OPTION |
| Status Affected: | None |
| Encoding: | 00 \| 0000 \| 0110 \| 0010 |
| Description: | The contents of the W register are loaded in the OPTION register. This instruction is supported for code compatibility with PIC16C5X products. Since OPTION is a readable/writable register, the user can directly address it. |
| Words: | 1 |
| Cycles: | 1 |
| Example | **To maintain upward compatibility with future PIC16C433 products, do not use this instruction.** |

**RETFIE** — **Return from Interrupt**

| | |
|---|---|
| Syntax: | [ *label* ] RETFIE |
| Operands: | None |
| Operation: | TOS → PC,<br>1 → GIE |
| Status Affected: | None |
| Encoding: | 00 \| 0000 \| 0000 \| 1001 |
| Description: | Return from Interrupt. Stack is POPed and Top-of-Stack (TOS) is loaded in the PC. Interrupts are enabled by setting Global Interrupt Enable bit, GIE (INTCON<7>). This is a two-cycle instruction. |
| Words: | 1 |
| Cycles: | 2 |
| Example | RETFIE<br>After Interrupt<br>PC = TOS<br>GIE = 1 |

**RETLW** — **Return with Literal in W**

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

**RETURN** **Return from Subroutine**

Syntax: [ *label* ] RETURN

Operands: None

Operation: TOS → PC

Status Affected: None

Encoding:

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

Description: Return from subroutine. The stack is POPed and the top of the stack (TOS) is loaded into the program counter. This is a two-cycle instruction.

Words: 1

Cycles: 2

Example

```
RETURN
```

After Interrupt
PC = TOS

**RRF** **Rotate Right f through Carry**

Syntax: [ *label* ] RRF f,d

Operands: 0 ≤ f ≤ 127
d ∈ [0,1]

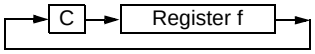
Operation: See description below

Status Affected: C

Encoding:

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

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




Words: 1

Cycles: 1

Example

```
RRF         REG1,0
```

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

**RLF** **Rotate Left f through Carry**

Syntax: [ *label* ] RLF f,d

Operands: 0 ≤ f ≤ 127
d ∈ [0,1]

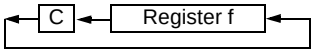
Operation: See description below

Status Affected: C

Encoding:

| | | | |
|---|---|---|---|
| 00 | 1101 | dfff | ffff |

Description: The contents of register 'f' are rotated one bit to the left through the Carry Flag. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is stored back in register 'f'.




Words: 1

Cycles: 1

Example

```
RLF         REG1,0
```

Before Instruction
REG1 = 1110 0110
C = 0
After Instruction
REG1 = 1110 0110
W = 1100 1100
C = 1

**SLEEP**

Syntax: [ *label* ] SLEEP

Operands: None

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

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

Encoding:

| | | | |
|---|---|---|---|
| 00 | 0000 | 0110 | 0011 |

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

Words: 1

Cycles: 1

Example:

```
SLEEP
```

**SUBLW** **Subtract W from Literal**

Syntax: [ *label* ] SUBLW k

Operands: $0 \le k \le 255$

Operation: k - (W) → (W)

Status Affected: C, DC, Z

Encoding:

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

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

Words: 1

Cycles: 1

Example 1:
```
SUBLW   0x02
```
Before Instruction
W = 1
C = ?
After Instruction
W = 1
C = 1; result is positive

Example 2: Before Instruction
W = 2
C = ?
After Instruction
W = 0
C = 1; result is zero

Example 3: Before Instruction
W = 3
C = ?
After Instruction
W = 0xFF
C = 0; result is negative

**SUBWF** **Subtract W from f**

Syntax: [ *label* ] SUBWF f,d

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

Operation: (f) - (W) → (dest)

Status Affected: C, DC, Z

Encoding:

| 00 | 0010 | dfff | ffff |
|---|---|---|---|

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

Words: 1

Cycles: 1

Example 1:
```
SUBWF   REG1,1
```
Before Instruction
REG1 = 3
W = 2
C = ?
After Instruction
REG1 = 1
W = 2
C = 1; result is positive

Example 2: Before Instruction
REG1 = 2
W = 2
C = ?
After Instruction
REG1 = 0
W = 2
C = 1; result is zero

Example 3: Before Instruction
REG1 = 1
W = 2
C = ?
After Instruction
REG1 = 0xFF
W = 2
C = 0; result is negative

**SWAPF** **Swap Nibbles in f**

Syntax: [ *label* ] SWAPF f,d

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

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

Status Affected: None

Encoding:

| 00 | 1110 | dfff | ffff |
|---|---|---|---|

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

Words: 1

Cycles: 1

Example

```
SWAPF REG,   0
```

Before Instruction

REG1 = 0xA5

After Instruction

REG1 = 0xA5
W = 0x5A

**TRIS** **Load TRIS Register**

Syntax: [ *label* ] TRIS f

Operands: $5 \le f \le 7$

Operation: (W) → TRIS register f;

Status Affected: None

Encoding:

| 00 | 0000 | 0110 | 0fff |
|---|---|---|---|

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

Words: 1

Cycles: 1

Example

**To maintain upward compatibility with future PIC16C433 products, do not use this instruction.**

**XORLW** **Exclusive OR Literal with W**

Syntax: [ *label* ] XORLW k

Operands: $0 \le k \le 255$

Operation: (W) .XOR. k → (W)

Status Affected: Z

Encoding:

| 11 | 1010 | kkkk | kkkk |
|---|---|---|---|

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

Words: 1

Cycles: 1

Example

```
XORLW   0xAF
```

Before Instruction

W = 0xB5

After Instruction

W = 0x1A

**XORWF** **Exclusive OR W with f**

Syntax: [ *label* ] XORWF f,d

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

Operation: (W) .XOR. (f) → (dest)

Status Affected: Z

Encoding:

| 00 | 0110 | dfff | ffff |
|---|---|---|---|

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

Words: 1

Cycles: 1

Example

```
XORWF   REG  1
```

Before Instruction

REG = 0xAF
W = 0xB5

After Instruction

REG = 0x1A
W = 0xB5

**NOTES:**

## 11.0 DEVELOPMENT SUPPORT

The PICmicro® 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 for PIC16F87X
- 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
  - KEELOQ® Demonstration Board

### 11.1 MPLAB Integrated Development Environment Software

The MPLAB IDE software brings an ease of software development previously unseen in the 8-bit microcontroller market. 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 PICmicro 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.

### 11.2 MPASM Assembler

The MPASM assembler is a full-featured universal macro assembler for all PICmicro MCU’s.

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.

### 11.3 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.

### 11.4 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.

### 11.5 MPLAB SIM Software Simulator

The MPLAB SIM software simulator allows code development in a PC-hosted environment by simulating the PICmicro 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.

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

The MPLAB ICE universal in-circuit emulator is intended to provide the product development engineer with a complete microcontroller design tool set for PICmicro 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 reconfigured for emulation of different processors. The universal architecture of the MPLAB ICE in-circuit emulator allows expansion to support new PICmicro 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.

### 11.7 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.

### 11.8 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 PIC16F87X and can be used to develop for this and other PICmicro microcontrollers from the PIC16CXXX family. The MPLAB ICD utilizes the in-circuit debugging capability built into the PIC16F87X. This feature, along with Microchip's In-Circuit Serial Programming™ protocol, offers cost-effective in-circuit FLASH 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.

### 11.9 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 PICmicro devices. It can also set code protection in this mode.

### 11.10 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 PICmicro 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.

### 11.11 PICDEM 1 Low Cost PICmicro 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.

### 11.12 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 $I^2C$™ bus and separate headers for connection to an LCD module and a keypad.

### 11.13 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.

### 11.14 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.

### 11.15 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 11-1: DEVELOPMENT TOOLS FROM MICROCHIP**

| | | PIC12CXXX | PIC14000 | PIC16C5X | PIC16C6X | PIC16CXXX | PIC16F62X | PIC16C7X | PIC16C7XX | PIC16C8X | PIC16F8XX | PIC16C9XX | PIC17C4X | PIC17C7XX | PIC18CXX2 | 24CXX/ 25CXX/ 93CXX | HCSXXX | MCRFXXX | MCP2510 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Software Tools | MPLAB® Integrated Development Environment | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | | | |
| | MPLAB® C17 C Compiler | | | | | | | | | | | | ✓ | ✓ | | | | | |
| | MPLAB® C18 C Compiler | | | | | | | | | | | | | | ✓ | | | | |
| | MPASM™ Assembler/ MPLINK™ Object Linker | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | |
| Emulators | 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.
† Development tool is available on select devices.

**NOTES:**

## 12.0 ELECTRICAL SPECIFICATIONS FOR PIC16C433

**Absolute Maximum Ratings †**

Ambient temperature under bias ..... –40° to +125°C

Storage temperature ..... –65°C to +150°C

Voltage on any pin with respect to $V_{SS}$ (except $V_{DD}$, $\overline{MCLR}$, LIN and BACT) ..... –0.3V to ($V_{DD}$ + 0.3V)

Voltage on $V_{DD}$ with respect to $V_{SS}$ ..... 0 to +7.0V

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

Total power dissipation **(Note 1)** ..... 700 mW

Maximum current out of $V_{SS}$ pin ..... 200 mA

Maximum current into $V_{DD}$ pin ..... 150 mA

Input clamp current, $I_{IK}$ ($V_I$ < 0 or $V_I$ > $V_{DD}$) ..... ± 20 mA

Output clamp current, $I_{OK}$ ($V_O$ < 0 or $V_O$ > $V_{DD}$) ..... ± 20 mA

Maximum output current sunk by any I/O pin (except LIN and BACT) ..... 25 mA

Maximum output current sunk by LIN and BACT ..... 200 mA

Maximum output current sourced by any I/O pin ..... 25 mA

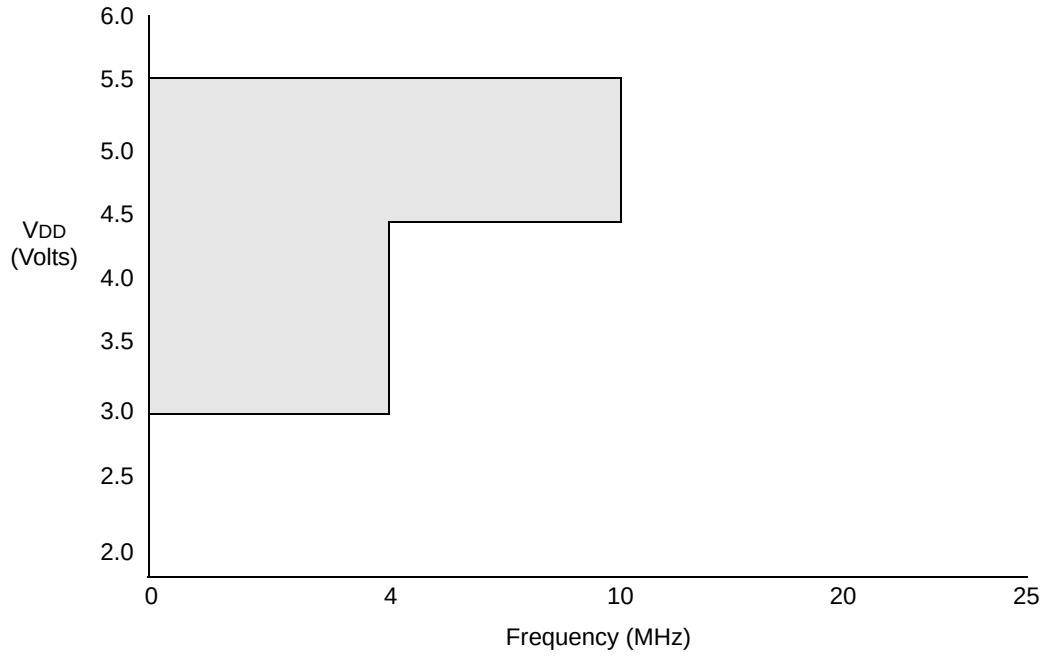
Maximum current sunk by GPIO pins combined ..... 100 mA

Maximum current sourced by GPIO pins combined ..... 100 mA

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

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

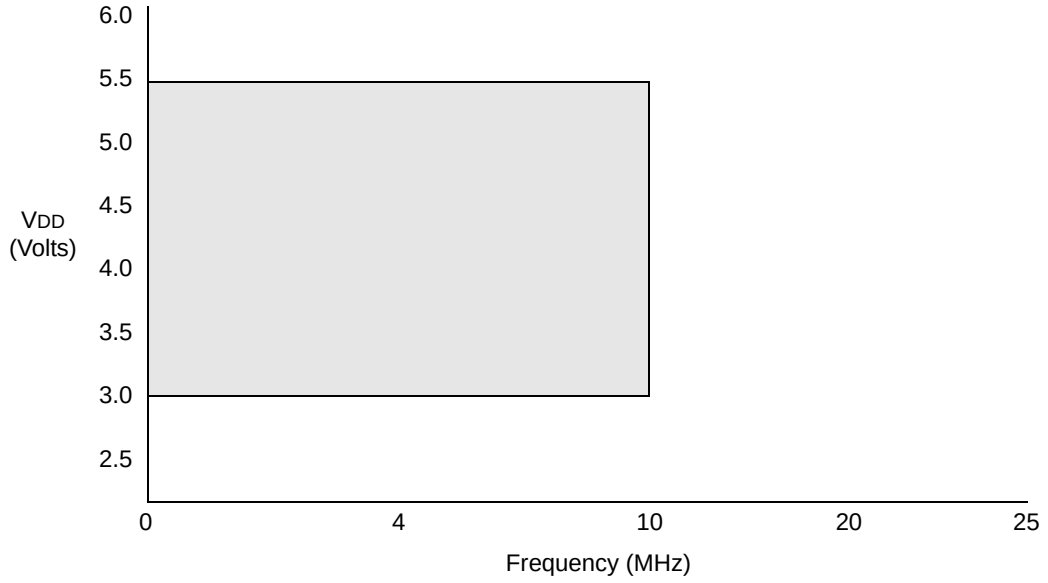
**FIGURE 12-1: PIC16C433 VOLTAGE-FREQUENCY GRAPH, -40°C ≤ TA < 0°C, +85°C <TA ≤ +125°C**




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

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

**FIGURE 12-2: PIC16C433 VOLTAGE-FREQUENCY GRAPH, 0°C ≤ TA ≤ +85°C**




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

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

## 12.1 DC Characteristics

| PIC16C433 (Industrial, Extended) | | | Standard Operating Conditions (unless otherwise specified)<br>Operating Temperature –40°C ≤ TA ≤ +85°C for industrial<br>–40°C ≤ TA ≤ +125°C for extended | | | | |
|---|---|---|---|---|---|---|---|
| **Parm No.** | **Sym** | **Characteristic** | **Min** | **Typ†** | **Max** | **Units** | **Conditions** |
| D001 | VDD | **Supply Voltage** | 3.0 | | 5.5 | V | |
| D001A | VBAT | **Operating Battery Voltage** | 8.0 | 13.8 | 18 | V | |
| D002 | VDR | **RAM Data Retention Voltage (Note 1)** | | 1.5* | | V | Device in SLEEP mode |
| D003 | VPOR | **VDD Start Voltage to ensure Power-on Reset** | | VSS | | V | See section on Power-on Reset for details |
| D004 | SVDD | **VDD Rise Rate to ensure Power-on Reset** | 0.05* | | | V/ms | See section on Power-on Reset for details |
| D010 | IDD | **Supply Current (Note 2)** | — | 1.2 | 2.5 | mA | FOSC = 4MHz, VDD = 3.0V<br>XT and EXTRC mode **(Note 3)** |
| D010C | | | — | 1.2 | 2.5 | mA | FOSC = 4MHz, VDD = 3.0V<br>INTRC mode **(Note 5)** |
| | | | — | 2.2 | 8 | mA | FOSC = 10MHz, VDD = 5.5V<br>HS mode |
| D010A | | | — | 19 | 29 | µA | FOSC = 32kHz, VDD = 3.0V, WDT disabled<br>LP mode, Commercial Temperature |
| | | | — | 19 | 37 | µA | FOSC = 32kHz, VDD = 3.0V, WDT disabled<br>LP mode, Industrial Temperature |
| | | | — | 32 | 60 | µA | FOSC = 32kHz, VDD = 3.0V, WDT disabled<br>LP mode, Extended Temperature |
| D020 | IPD | **Power-down Current (Note 4)** | — | 0.25 | 6 | µA | VDD = 3.0V, Commercial, WDT disabled |
| D021 | | | — | 0.25 | 7 | µA | VDD = 3.0V, Industrial, WDT disabled |
| D021B | | | — | 2 | 14 | µA | VDD = 3.0V, Extended, WDT disabled |
| | | | — | 0.5 | 8 | µA | VDD = 5.5V, Commercial, WDT disabled |
| | | | — | 0.8 | 9 | µA | VDD = 5.5V, Industrial, WDT disabled |
| | | | — | 3 | 16 | µA | VDD = 5.5V, Extended, WDT disabled |
| D022 | ΔIWDT | **Watchdog Timer Current** | — | 2.2 | 5 | µA | VDD = 3.0V, Commercial |
| | | | — | 2.2 | 6 | µA | VDD = 3.0V, Industrial |
| | | | — | 4 | 11 | µA | VDD = 3.0V, Extended |
| D028 | ΔIEE | **Supply Current (Note 2)**<br>During read/write to EEPROM peripheral | — | 0.1 | 0.2 | mA | FOSC = 4MHz, VDD = 5.5V, SCL = 400kHz |
| | FOSC | **LP Oscillator Operating Frequency** | 0 | | 200 | kHz | All temperatures |
| | | **INTRC/EXTRC Oscillator Operating Frequency** | — | | 4(5) | MHz | All temperatures |
| | | **XT Oscillator Operating Frequency** | 0 | | 4 | MHz | All temperatures |
| | | **HS Oscillator Operating Frequency** | 0 | | 10 | MHz | All temperatures |

* These parameters are characterized but not tested.

† Data in Typical ("Typ") column is based on characterization results at 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 bus loading, oscillator type, bus rate, internal code execution pattern and temperature, also have an impact on the current consumption.

a) The test conditions for all IDD measurements in active operation mode are: OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VSS, T0CKI = VDD, MCLR = VDD; WDT disabled.

b) For standby current measurements, the conditions are the same, except that the device is in SLEEP mode.

**3:** For EXTRC osc configuration, current through REXT is not included. The current through the resistor can be estimated by the formula:
Ir = VDD/2REXT (mA) with REXT in kOhm.

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

**5:** INTRC calibration value is for 4MHz nominal at 5V, 25°C.

## 12.2 DC Characteristics: PIC16C433 (Industrial, Extended) and PIC16LC433 (Industrial)

| DC CHARACTERISTICS | | | Standard Operating Conditions (unless otherwise specified)<br>Operating temperature –40°C ≤ TA ≤ +85°C (industrial)<br>–40°C ≤ TA ≤ +125°C (extended)<br>Operating voltage VDD range as described in DC spec Section 12.1. | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Characteristic** | **Sym** | **Min** | **Typ†** | **Max** | **Units** | **Conditions** |
| | **Input Low Voltage** | | | | | | |
| | I/O ports | VIL | | | | | |
| D030 | with TTL buffer | | VSS | — | 0.8V | V | For 4.5V ≤ VDD ≤ 5.5V |
| | | | VSS | — | 0.15VDD | V | otherwise |
| D031 | with Schmitt Trigger buffer | | VSS | — | 0.2VDD | V | |
| D032 | $\overline{MCLR}$, GP2/T0CKI/AN2/INT (in EXTRC mode) | | VSS | — | 0.2VDD | V | |
| D033 | OSC1 (in EXTRC mode) | | VSS | — | 0.2VDD | | **(Note 1)** |
| D033 | OSC1 (in XT, HS, and LP) | | VSS | — | 0.3VDD | V | **(Note 1)** |
| | **Input High Voltage** | | | | | | |
| | I/O ports | VIH | | — | | | |
| D040 | with TTL buffer | | 2.0V | — | VDD | V | 4.5V ≤ VDD ≤ 5.5V |
| D040A | | | 0.25VDD + 0.8V | — | VDD | V | otherwise |
| D041 | with Schmitt Trigger buffer | | 0.8VDD | — | VDD | V | For entire VDD range |
| D042 | $\overline{MCLR}$, GP2/T0CKI/AN2/INT | | 0.8VDD | — | VDD | V | |
| D042A | OSC1 (XT, HS, and LP) | | 0.7VDD | — | VDD | V | **(Note 1)** |
| D043 | OSC1 (in EXTRC mode) | | 0.9VDD | — | VDD | V | |
| | **Input Leakage Current (Notes 2, 3)** | | | | | | |
| D060 | I/O ports | IIL | — | — | ±1 | µA | VSS ≤ VPIN ≤ VDD, pin at hi-impedance |
| D061 | GP3/$\overline{MCLR}$ **(Note 5)** | | | | ±30 | µA | VSS ≤ VPIN ≤ VDD |
| D061A | GP3 **(Note 6)** | | | | ±5 | µA | VSS ≤ VPIN ≤ VDD |
| D062 | GP2/T0CKI | | — | — | ±5 | µA | VSS ≤ VPIN ≤ VDD |
| D063 | OSC1 | | — | — | ±5 | µA | VSS ≤ VPIN ≤ VDD, XT, HS, and LP osc configuration |
| D070 | GPIO weak pull-up current **(Note 4)** | IPUR | 50 | 250 | 400 | µA | VDD = 5V, VPIN = VSS |
| | $\overline{MCLR}$ pull-up current | — | — | — | 30 | µA | VDD = 5V, VPIN = VSS |
| | **Output Low Voltage** | | | | | | |
| D080 | I/O ports | VOL | — | — | 0.6 | V | IOL = 8.5 mA, VDD = 4.5V, –40°C to +85°C |
| D080A | | | — | — | 0.6 | V | IOL = 7.0 mA, VDD = 4.5V, –40°C to +125°C |
| D083 | OSC2/CLKOUT | | — | — | 0.6 | V | IOL = 1.6 mA, VDD = 4.5V, –40°C to +85°C |
| D083A | | | — | — | 0.6 | V | IOL = 1.2 mA, VDD = 4.5V, –40°C to +125°C |

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

**Note 1:** In EXTRC oscillator configuration, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended that the PIC16C433 be driven with external clock in RC mode.

**2:** The leakage current on the $\overline{MCLR}$ pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

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

**4:** Does not include GP3. For GP3 see parameters D061 and D061A.

**5:** This spec. applies to GP3/$\overline{MCLR}$ configured as external $\overline{MCLR}$ and GP3/$\overline{MCLR}$ configured as input with internal pull-up enabled.

**6:** This spec. applies when GP3/$\overline{MCLR}$ is configured as an input with pull-up disabled. The leakage current of the $\overline{MCLR}$ circuit is higher than the standard I/O logic.

| DC CHARACTERISTICS | | Standard Operating Conditions (unless otherwise specified)<br>Operating temperature –40°C ≤ TA ≤ +85°C (industrial)<br>–40°C ≤ TA ≤ +125°C (extended)<br>Operating voltage VDD range as described in DC spec Section 12.1. | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Characteristic** | **Sym** | **Min** | **Typ†** | **Max** | **Units** | **Conditions** |
| | **Output High Voltage** | | | | | | |
| D090 | I/O ports **(Note 3)** | VOH | VDD - 0.7 | — | — | V | IOH = -3.0 mA, VDD = 4.5V, –40°C to +85°C |
| D090A | | | VDD - 0.7 | — | — | V | IOH = -2.5 mA, VDD = 4.5V, –40°C to +125°C |
| D092 | OSC2/CLKOUT | | VDD - 0.7 | — | — | V | IOH = 1.3 mA, VDD = 4.5V, –40°C to +85°C |
| D092A | | | VDD - 0.7 | — | — | V | IOH = 1.0 mA, VDD = 4.5V, –40°C to +125°C |
| | **Capacitive Loading Specs on Output Pins** | | | | | | |
| D100 | OSC2 pin | COSC2 | — | — | 15 | pF | In XT and LP modes when external clock is used to drive OSC1 |
| D101 | All I/O pins | CIO | — | — | 50 | pF | |

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

**Note 1:** In EXTRC oscillator configuration, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended that the PIC16C433 be driven with external clock in RC mode.

**2:** The leakage current on the $\overline{MCLR}$ pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

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

**4:** Does not include GP3. For GP3 see parameters D061 and D061A.

**5:** This spec. applies to GP3/$\overline{MCLR}$ configured as external $\overline{MCLR}$ and GP3/$\overline{MCLR}$ configured as input with internal pull-up enabled.

**6:** This spec. applies when GP3/$\overline{MCLR}$ is configured as an input with pull-up disabled. The leakage current of the $\overline{MCLR}$ circuit is higher than the standard I/O logic.

**TABLE 12-1: LIN bus TRANSCEIVER OPERATING SPECIFICATIONS**

Operating Conditions: $V_{DD}$ range as described in Table 12-1, -40°C < $T_A$ < +125°C.

| Param No. | Characteristics | Sym | Min | Typ | Max | Units | Comments |
|---|---|---|---|---|---|---|---|
| D313 | $V_{DD}$ Quiescent Operating Current | $I_{DD\_LIN}$ | - | - | 1 | mA | |
| D314 | $V_{BAT}$ Low Power Current | $I_{BAT}$ | | | 50 | µA | |

* These parameters are characterized but not tested.

**TABLE 12-2: LIN bus TRANSCEIVER INTERFACE SPECIFICATIONS**

Operating Conditions: $V_{DD}$ range as described in Table 12-1, -40°C < $T_A$ < +125°C.

| Param No. | Characteristics | Sym | Min | Typ | Max | Units | Comments |
|---|---|---|---|---|---|---|---|
| D315 | Low Level Output Current | $I_{OL\_LIN\_DOMINANT}$ | 40 | - | 200 | mA | $V_{BUS}$ = 12V |
| D316 | High Level Output Leakage Current | $I_{OH\_LIN}$ | -20 | | 20 | µA | $V_{BUS} \geq V_{BAT}$; $V_{BUS}$ < 40V |
| D317 | Low Level Output Current, Open Ground | $I_{OH\_LIN\_REVERS}$ | -1 | | 1 | mA | |
| D318 | Low Level Input Voltage | $V_{IL\_LIN}$ | -8 | | 0.4$V_{BAT}$ | V | Dominant State |
| D319 | High Level Input Voltage | $V_{IH\_LIN}$ | 0.6$V_{BAT}$ | | 18 | V | Recessive State |
| D320 | Input Hysteresis | $V_{HYS\_LIN}$ | 0.05$V_{BAT}$ | | 0.1$V_{BAT}$ | V | $V_{IH\_LIN}$ - $V_{IL\_LIN}$ |
| D321 | Short Circuit Current Limit | $I_{SC\_LIN}$ | 0.05 | | 200 | mA | |
| D322 | High Level Output Voltage | $V_{OH\_LIN}$ | 0.8$V_{BAT}$ | | | V | |
| D323 | Low Level Output Voltage | $V_{OL\_LIN}$ | | | 0.2$V_{BAT}$ | V | |

* These parameters are characterized but not tested.

## 12.3 Timing Parameter Symbology

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

1. TppS2ppS
2. TppS
3. TCC:ST ($I^2C$ specifications only)
4. Ts ($I^2C$ specifications only)

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

Lowercase letters (pp) and their meanings:

| | | | |
|---|---|---|---|
| **pp** | | | |
| cc | CCP1 | osc | OSC1 |
| ck | CLKOUT | rd | $\overline{RD}$ |
| cs | $\overline{CS}$ | rw | $\overline{RD}$ or $\overline{WR}$ |
| di | SDI | sc | SCK |
| do | SDO | ss | $\overline{SS}$ |
| dt | Data in | t0 | T0CKI |
| io | I/O port | t1 | T1CKI |
| mc | $\overline{MCLR}$ | wr | $\overline{WR}$ |

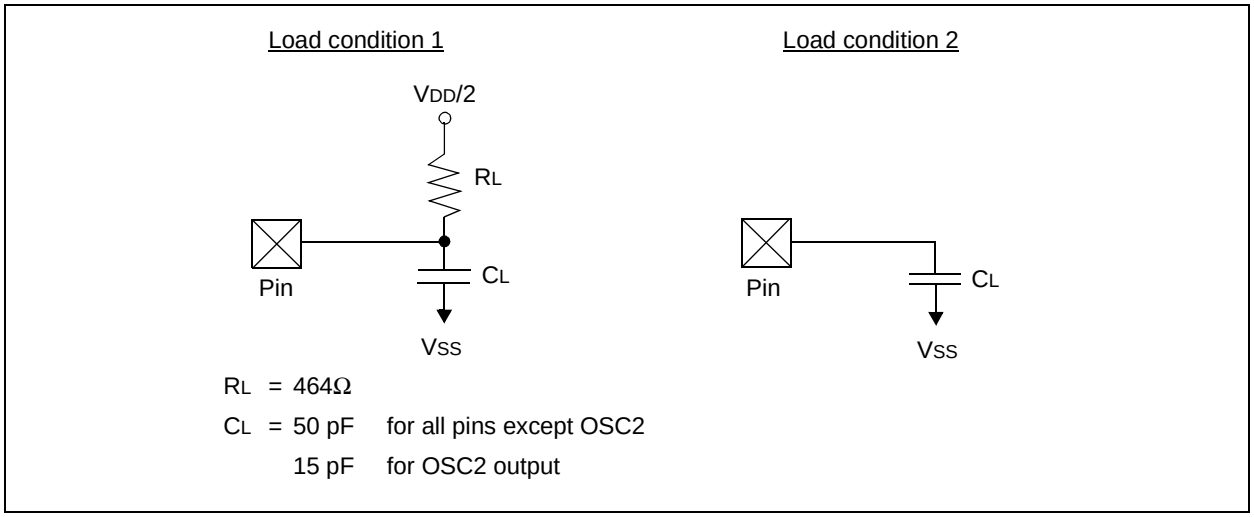
Uppercase letters and their meanings:

| | | | |
|---|---|---|---|
| **S** | | | |
| F | Fall | P | Period |
| H | High | R | Rise |
| I | Invalid (Hi-impedance) | V | Valid |
| L | Low | Z | Hi-impedance |
| **$I^2C$ only** | | | |
| AA | output access | High | High |
| BUF | Bus free | Low | Low |

TCC:ST ($I^2C$ specifications only)

| | | | |
|---|---|---|---|
| **CC** | | | |
| HD | Hold | SU | Setup |
| **ST** | | | |
| DAT | DATA input hold | STO | STOP condition |
| STA | START condition | | |

**FIGURE 12-3: LOAD CONDITIONS**



## 12.4 Timing Diagrams and Specifications

**FIGURE 12-4: EXTERNAL CLOCK TIMING**

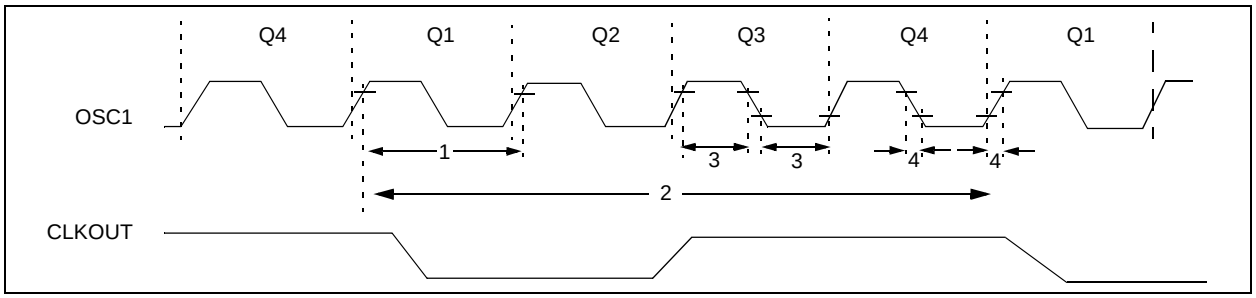



**TABLE 12-3: CLOCK TIMING REQUIREMENTS**

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

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

**Note 1:** Instruction cycle period (TCY) equals four times the input oscillator time-base period. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at "Min." values with an external clock applied to the OSC1/CLKIN pin.

When an external clock input is used, the "Max." cycle time limit is "DC" (no clock) for all devices. OSC2 is disconnected (has no loading) for the PIC16C433.

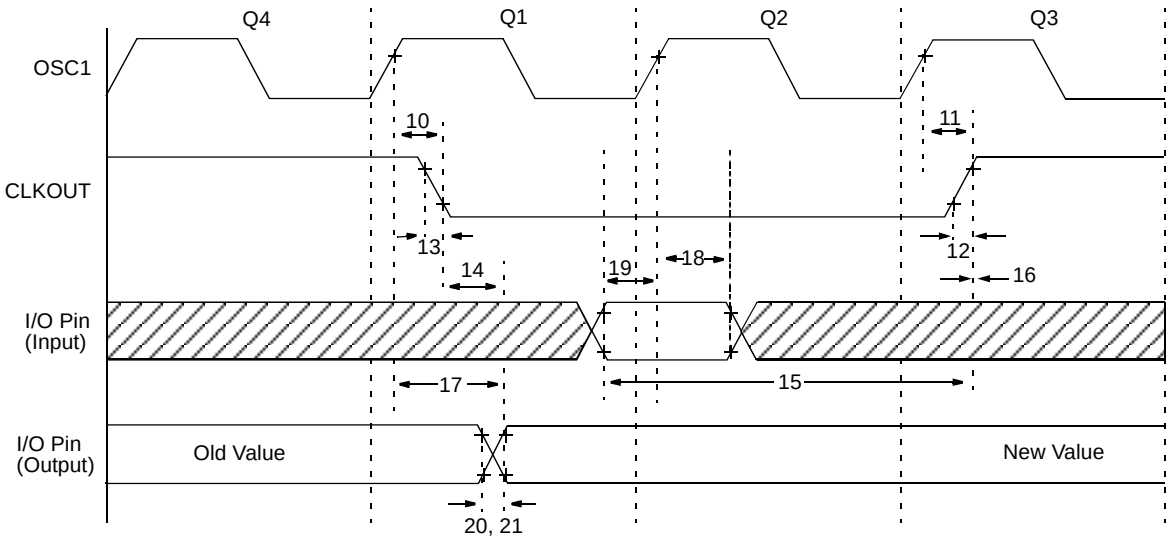
**TABLE 12-4: CALIBRATED INTERNAL RC FREQUENCIES - PIC16C433**

| AC Characteristics | | **Standard Operating Conditions (unless otherwise specified)**<br>Operating Temperature –40°C ≤ TA ≤ +85°C (industrial)<br>Operating Voltage VDD range is described in Section 10.1 | | | | | |
|---|---|---|---|---|---|---|---|
| **Parameter No.** | **Sym** | **Characteristic** | **Min*** | **Typ(1)** | **Max*** | **Units** | **Conditions** |
| | | Internal Calibrated RC Frequency | 3.65 | 4.00 | 4.28 | MHz | VDD = 5.0V |

* These parameters are characterized but not tested.

**Note 1:** Data in the Typical ("Typ") column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**FIGURE 12-5: CLKOUT AND I/O TIMING**




**Note:** Refer to Figure 12-3 for load conditions.

**TABLE 12-5: CLKOUT AND I/O TIMING REQUIREMENTS**

| Param No. | Sym | Characteristic | | Min | Typ† | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| 10* | TosH2ckL | OSC1↑ to CLKOUT↓ | | — | 75 | 200 | ns | **(Note 1)** |
| 11* | TosH2ckH | OSC1↑ to CLKOUT↑ | | — | 75 | 200 | ns | **(Note 1)** |
| 12* | TckR | CLKOUT rise time | | — | 35 | 100 | ns | **(Note 1)** |
| 13* | TckF | CLKOUT fall time | | — | 35 | 100 | ns | **(Note 1)** |
| 14* | TckL2ioV | CLKOUT ↓ to Port out valid | | — | — | 0.5TCY + 20 | ns | **(Note 1)** |
| 15* | TioV2ckH | Port in valid before CLKOUT ↑ | | TOSC + 200 | — | — | ns | **(Note 1)** |
| 16* | TckH2ioI | Port in hold after CLKOUT ↑ | | 0 | — | — | ns | **(Note 1)** |
| 17* | TosH2ioV | OSC1↑ (Q1 cycle) to Port out valid | | — | 50 | 150 | ns | |
| 18* | TosH2ioI | OSC1↑ (Q2 cycle) to Port input invalid (I/O in hold time) | PIC16**C**433 | 100 | — | — | ns | |
| 18A* | | | PIC16**LC**433 | 200 | — | — | ns | |
| 19* | TioV2osH | Port input valid to OSC1↑ (I/O in setup time) | | 0 | — | — | ns | |
| 20* | TioR | Port output rise time | PIC16**C**433 | — | 10 | 40 | ns | |
| 20A* | | | PIC16**LC**433 | — | — | 80 | ns | |
| 21* | TioF | Port output fall time | PIC16**C**433 | — | 10 | 40 | ns | |
| 21A* | | | PIC16**LC**433 | — | — | 80 | ns | |
| 22††* | Tinp | GP2/INT pin high or low time | | TCY | — | — | ns | |
| 23††* | Trbp | GP0/GP1/GP3 change INT high or low time | | TCY | — | — | ns | |

* These parameters are characterized but not tested.

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

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

**Note 1:** Measurements are taken in EXTRC and INTRC modes where CLKOUT output is 4 x TOSC.

**FIGURE 12-6: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, AND POWER-UP TIMER TIMING**

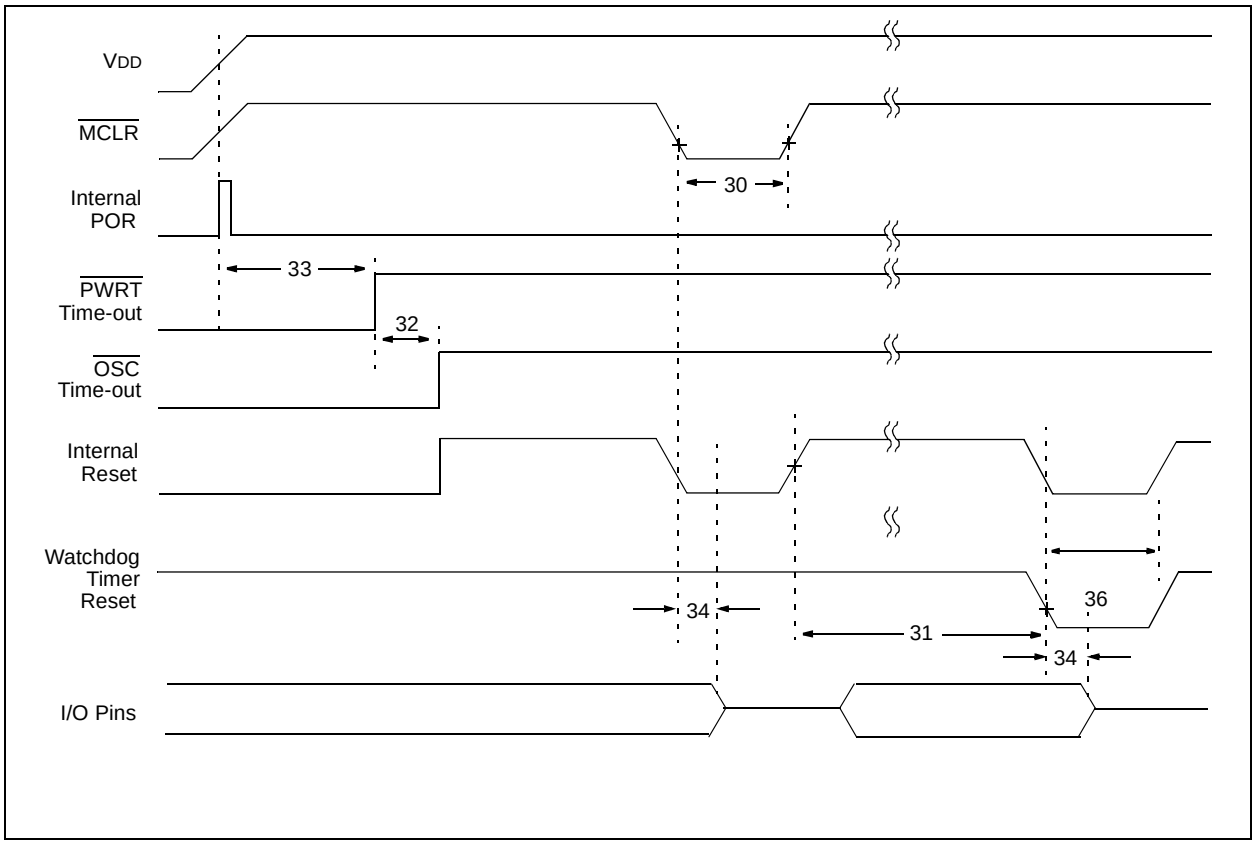


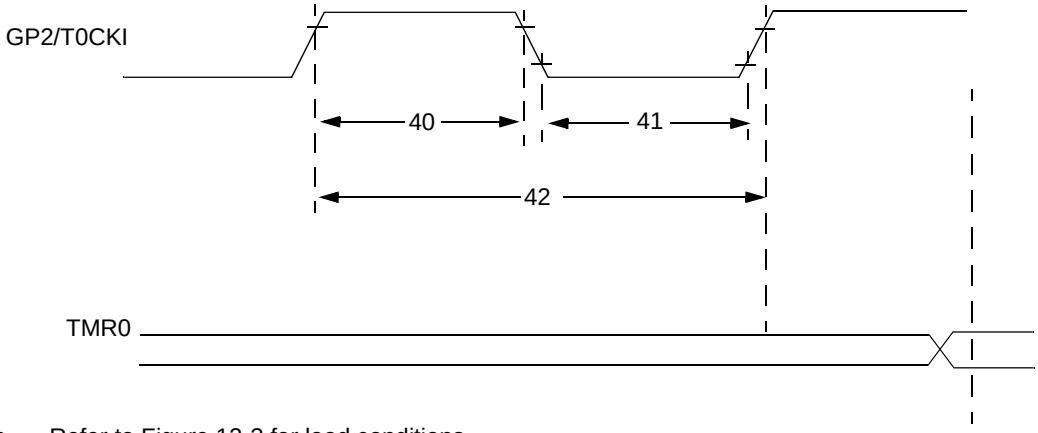

**TABLE 12-6: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER**

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

* These parameters are characterized but not tested.

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

**FIGURE 12-7: TIMER0 CLOCK TIMINGS**




**Note:** Refer to Figure 12-3 for load conditions.

**TABLE 12-7: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS**

| Param No. | Sym | Characteristic | | Min | Typ† | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| 40* | Tt0H | T0CKI High Pulse Width | No Prescaler | 0.5TCY + 20 | — | — | ns | Must also meet parameter 42 |
| | | | With Prescaler | 10 | — | — | ns | |
| 41* | Tt0L | T0CKI Low Pulse Width | No Prescaler | 0.5TCY + 20 | — | — | ns | Must also meet parameter 42 |
| | | | With Prescaler | 10 | — | — | ns | |
| 42* | Tt0P | T0CKI Period | No Prescaler | TCY + 40 | — | — | ns | |
| | | | With Prescaler | Greater of: 20 or $\frac{T_{CY} + 40}{N}$ | — | — | ns | N = prescale value (2, 4,..., 256) |
| 48 | TCKE2tmr1 | Delay from external clock edge to timer increment | | 2TOSC | — | 7TOSC | — | |

* These parameters are characterized but not tested.

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

**TABLE 12-8: GPIO PULL-UP RESISTOR RANGES**

| VDD (Volts) | Temperature (°C) | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| GP0/GP1 | | | | | |
| 2.5 | –40 | 38K | 42K | 63K | Ω |
| | 25 | 42K | 48K | 63K | Ω |
| | 85 | 42K | 49K | 63K | Ω |
| | 125 | 50K | 55K | 63K | Ω |
| 5.5 | –40 | 15K | 17K | 20K | Ω |
| | 25 | 18K | 20K | 23K | Ω |
| | 85 | 19K | 22K | 25K | Ω |
| | 125 | 22K | 24K | 28K | Ω |
| GP3 | | | | | |
| 2.5 | –40 | 285K | 346K | 417K | Ω |
| | 25 | 343K | 414K | 532K | Ω |
| | 85 | 368K | 457K | 532K | Ω |
| | 125 | 431K | 504K | 593K | Ω |
| 5.5 | –40 | 247K | 292K | 360K | Ω |
| | 25 | 288K | 341K | 437K | Ω |
| | 85 | 306K | 371K | 448K | Ω |
| | 125 | 351K | 407K | 500K | Ω |

* These parameters are characterized but not tested.

**TABLE 12-9: LIN bus AC CHARACTERISTICS**

| Symbol | Parameter | Min. | Typ. | Max. | Unit | Note |
|---|---|---|---|---|---|---|
| $\lvert dV/dt \rvert$ | Slope rising and falling edges | 1 | 2 | | V/µs | |
| $T_{trans\_pd}$ | Propagation delay of transmitter | | | 4 | µs | $T_{trans\_pd} = max(T_{trans\_pdr}$ or $T_{trans\_pdf})$ |
| $T_{rec\_pd}$ | Propagation delay of receiver | | | 6 | µs | $T_{rec\_pd} = max(T_{rec\_pdr}$ or $T_{rec\_pdf})$ |
| $T_{rec\_sym}$ | Symmetry of receiver propagation delay rising edge w.r.t. falling edge | -2 | | 2 | µs | $T_{rec\_sym} = T_{rec\_pdf} - T_{rec\_pdr})$ |
| $T_{trans\_sym}$ | Symmetry of transmitter propagation delay rising edge w.r.t. falling edge | -2 | | 2 | µs | $T_{trans\_sym} = T_{trans\_pdf} - T_{rans\_pdr})$ |

**TABLE 12-10: LIN THERMAL CHARACTERISTICS**

| Symbol | Parameter | Typ. | Max. | Unit | Note |
|---|---|---|---|---|---|
| $\Theta_{recovery}$ | Recovery Temperature | +135 | | °C | Information Parameter |
| $\Theta_{shutdown}$ | Shut-down Temperature | +155 | | °C | Information Parameter |
| $T_{THERM}$ | Short Circuit Recovery Time | | 1.5 | ms | Information Parameter |

**FIGURE 12-8: TIMING DIAGRAM**

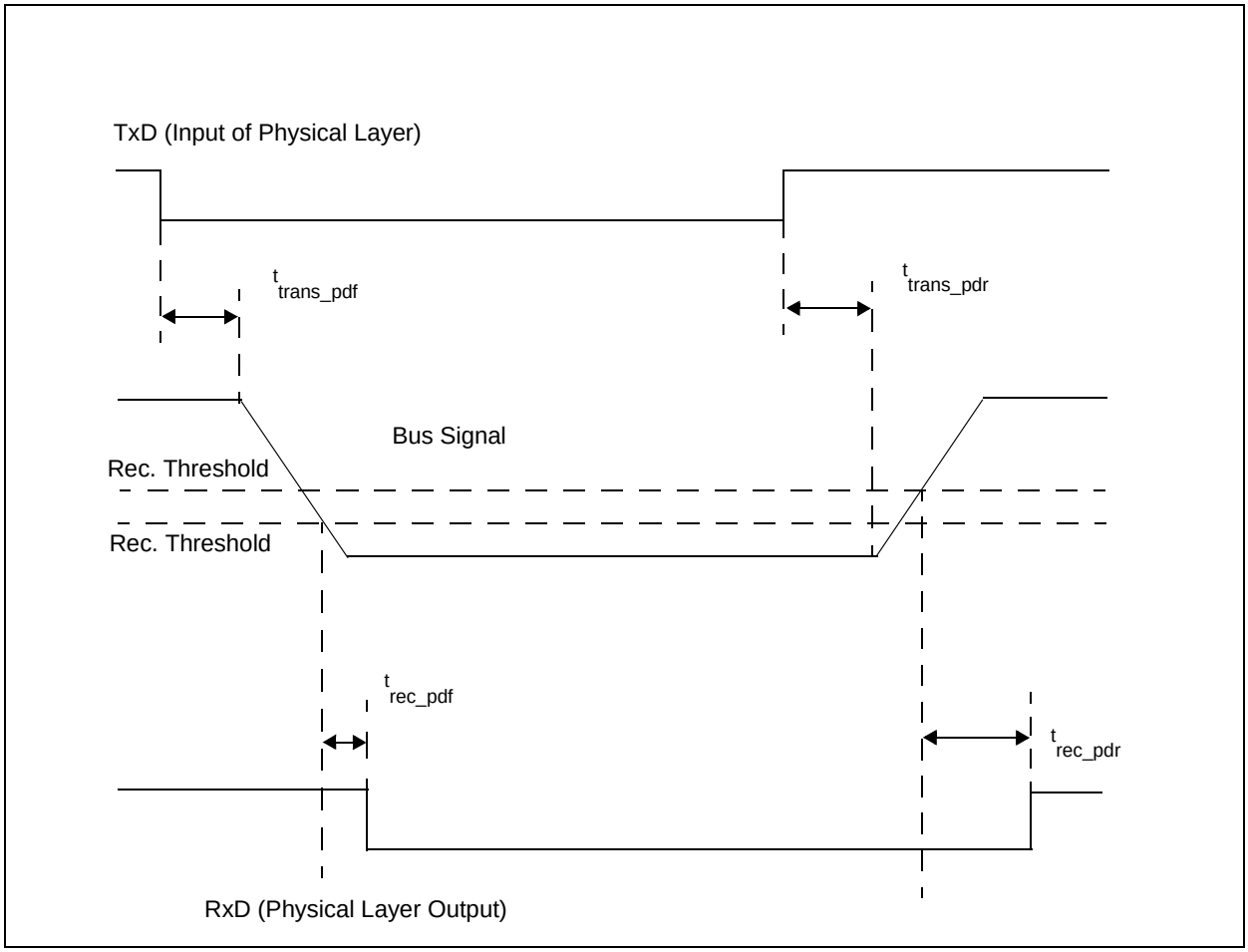


**TABLE 12-11: A/D CONVERTER CHARACTERISTICS: PIC16C433**

| Param No. | Sym | Characteristic | | Min | Typ† | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| A01 | NR | Resolution | | — | — | 8-bits | bit | VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF |
| A02 | EABS | Total absolute error | | — | — | < ±1 | LSb | VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF |
| A03 | EIL | Integral linearity error | | — | — | < ±1 | LSb | VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF |
| A04 | EDL | Differential linearity error | | — | — | < ±1 | LSb | VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF |
| A05 | EFS | Full scale error | | — | — | < ±1 | LSb | VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF |
| A06 | EOFF | Offset error | | — | — | < ±1 | LSb | VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF |
| A10 | — | Monotonicity | | — | guaranteed **(Note 3)** | — | — | VSS ≤ VAIN ≤ VREF |
| A20 | VREF | Reference voltage | | 2.5V | — | VDD + 0.3 | V | |
| A25 | VAIN | Analog input voltage | | VSS - 0.3 | — | VREF + 0.3 | V | |
| A30 | ZAIN | Recommended impedance of analog voltage source | | — | — | 10.0 | kΩ | |
| A40 | IAD | A/D conversion current (VDD) | PIC16**C**433 | — | 180 | — | µA | Average current consumption when A/D is on **(Note 1)** |
| | | | PIC16**LC**433 | — | 90 | — | µA | |
| A50 | IREF | VREF input current **(Note 2)** | | 10 | — | 1000 | µA | During VAIN acquisition. Based on differential of VHOLD to VAIN to charge CHOLD, see Section 8.1 |
| | | | | — | — | 10 | µA | During A/D Conversion cycle |

* These parameters are characterized but not tested.

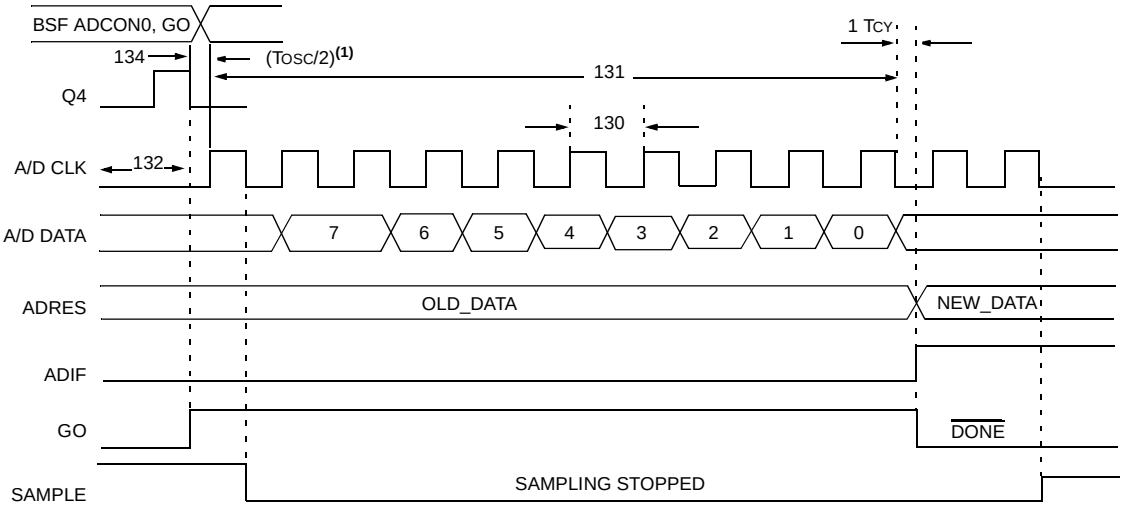
† Data in "Typ" column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note 1:** When A/D is off, it will not consume any current other than minor leakage current. The power-down current spec includes any such leakage from the A/D module.

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

**3:** The A/D conversion result never decreases with an increase in the input voltage, and has no missing codes.

**FIGURE 12-9: A/D CONVERSION TIMING**




**Note 1:** If the A/D clock source is selected as RC, a time of TCY is added before the A/D clock starts. This allows the SLEEP instruction to be executed.

**TABLE 12-12: A/D CONVERSION REQUIREMENTS**

| Param No. | Sym | Characteristic | | Min | Typ† | Max | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| 130 | TAD | A/D clock period | PIC16**C**433 | 1.6 | — | — | µs | TOSC based, VREF ≥ 3.0V |
| | | | PIC16**LC**433 | 2.0 | — | — | µs | TOSC based, VREF full range |
| | | | PIC16**C**433 | 2.0 | 4.0 | 6.0 | µs | A/D RC mode |
| | | | PIC16**C**433 | 3.0 | 6.0 | 9.0 | µs | A/D RC mode |
| 131 | TCNV | Conversion time (not including S/H time) **(Note 1)** | | 11 | — | 11 | Tad | |
| 132 | TACQ | Acquisition time | | **(Note 2)** | 20 | — | µs | |
| | | | | 5* | — | — | µs | The minimum time is the amplifier setting time. This may be used if the "new" input voltage has not changed by more than 1 LSb (i.e., 20.0 mV @ 5.12V) from the last sampled voltage (as stated on CHOLD). |
| 134 | TGO | Q4 to A/D clock start | | — | TOSC/2 § | — | — | If the A/D clock source is selected as RC, a time of TCY is added before the A/D clock starts. This allows the SLEEP instruction to be executed. |
| 135 | TSWC | Switching from convert → sample time | | 1.5 § | — | — | Tad | |

* 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.

§ This specification ensured by design.

**Note 1:** ADRES register may be read on the following TCY cycle.

**2:** See Section 8.1 for minimum conditions.

**NOTES:**

## 13.0 DC AND AC CHARACTERISTICS - PIC16C433

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

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

**FIGURE 13-1: CALIBRATED INTERNAL RC FREQUENCY RANGE VS. TEMPERATURE (VDD = 5.0V) (INTERNAL RC IS CALIBRATED TO 25°C, 5.0V)**

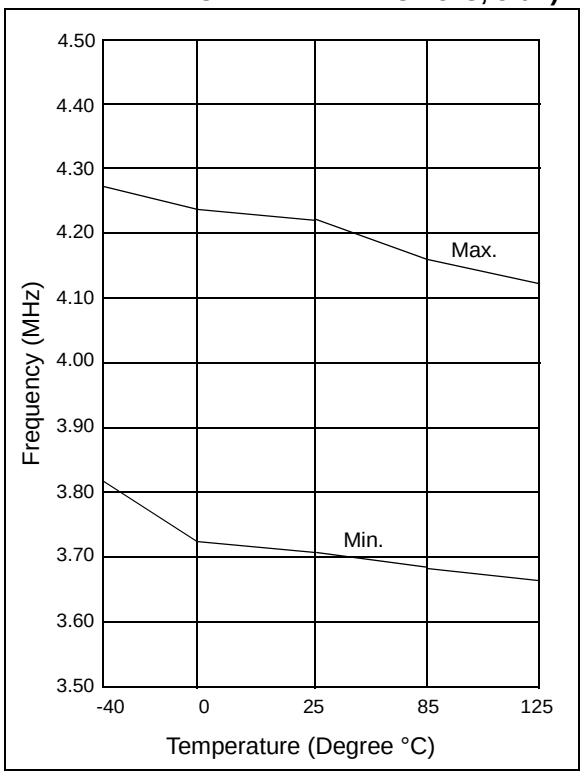



**FIGURE 13-2: CALIBRATED INTERNAL RC FREQUENCY RANGE VS. TEMPERATURE (VDD = 2.5V) (INTERNAL RC IS CALIBRATED TO 25°C, 5.0V)**

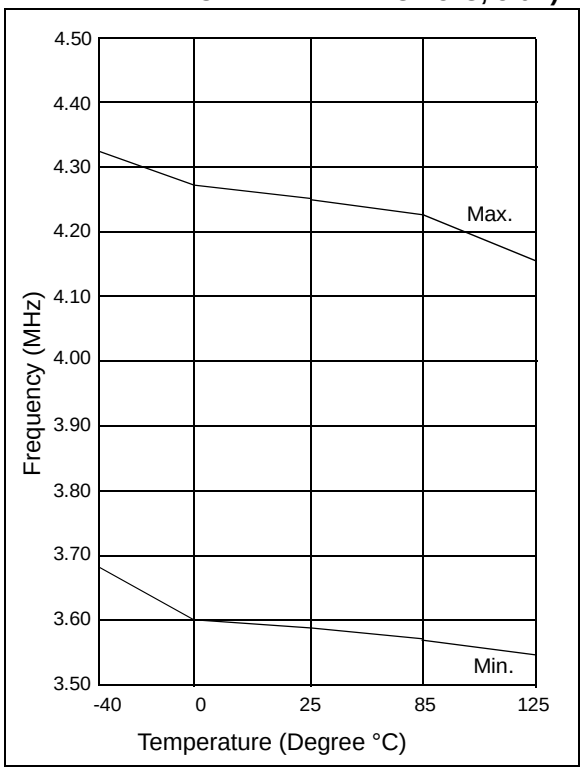


**TABLE 13-1: DYNAMIC IDD (TYPICAL) - WDT ENABLED, 25°C**

| Oscillator | Frequency | VDD = 2.5V | VDD = 5.5V |
|---|---|---|---|
| External RC | 4 MHz | 400 µA* | 900 µA* |
| Internal RC | 4 MHz | 400 µA | 900 µA |
| XT | 4 MHz | 400 µA | 900 µA |
| LP | 32 kHz | 15 µA | 60 µA |

*Does not include current through external R&C.

**FIGURE 13-3: WDT TIMER TIME-OUT PERIOD vs. VDD**

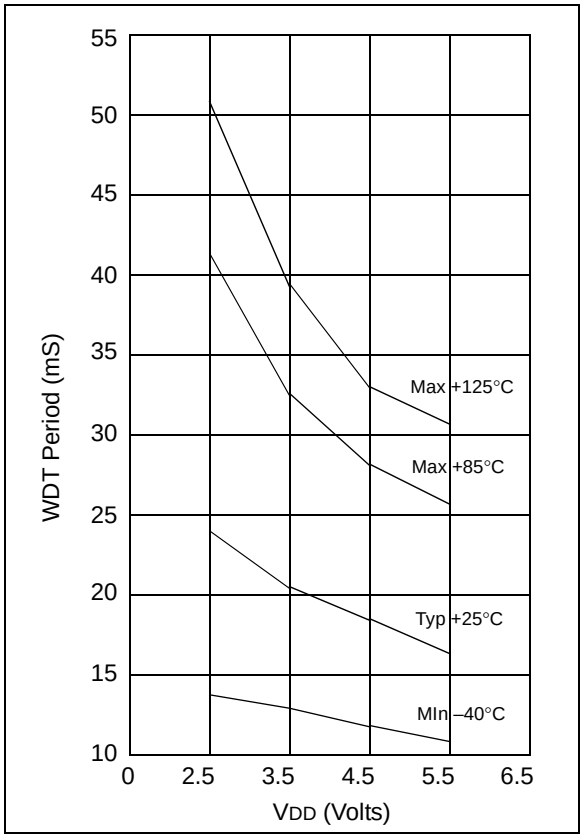



**FIGURE 13-4: IOH vs. VOH, VDD = 2.5V**

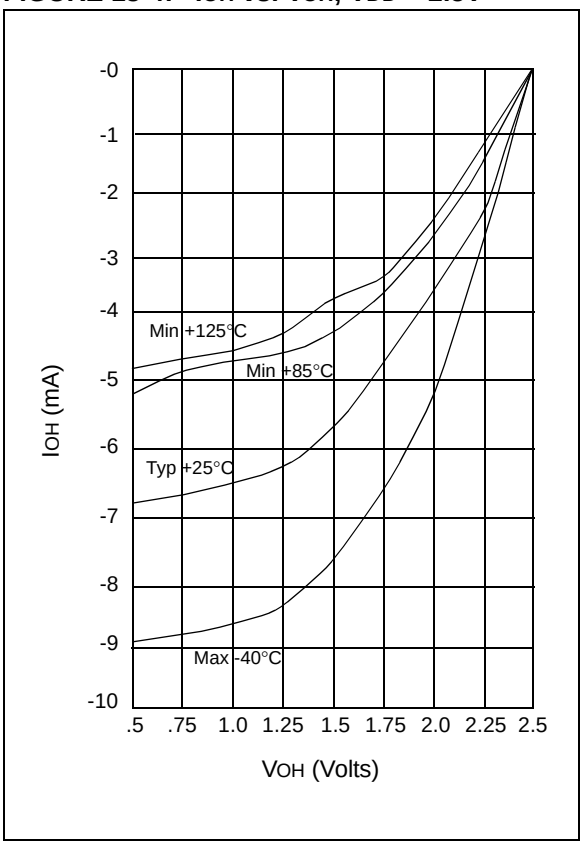


**FIGURE 13-5: IOH vs. VOH, VDD = 3.5V**

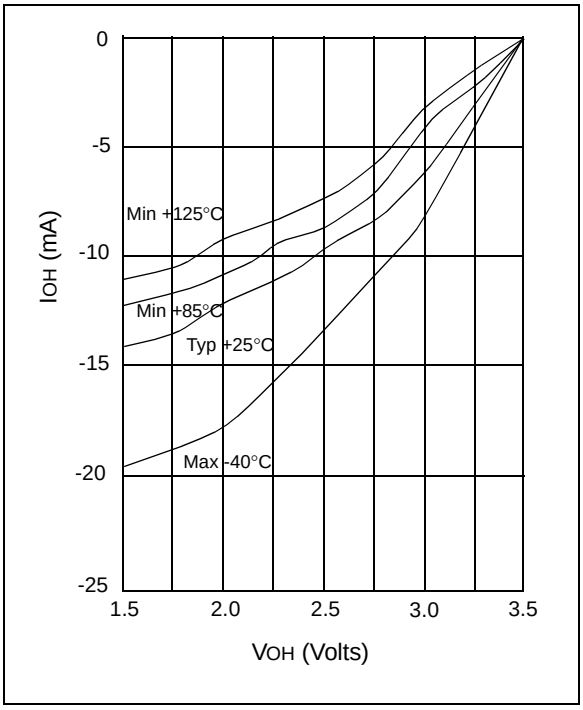



**FIGURE 13-6: IOH vs. VOH, VDD = 5.5V**

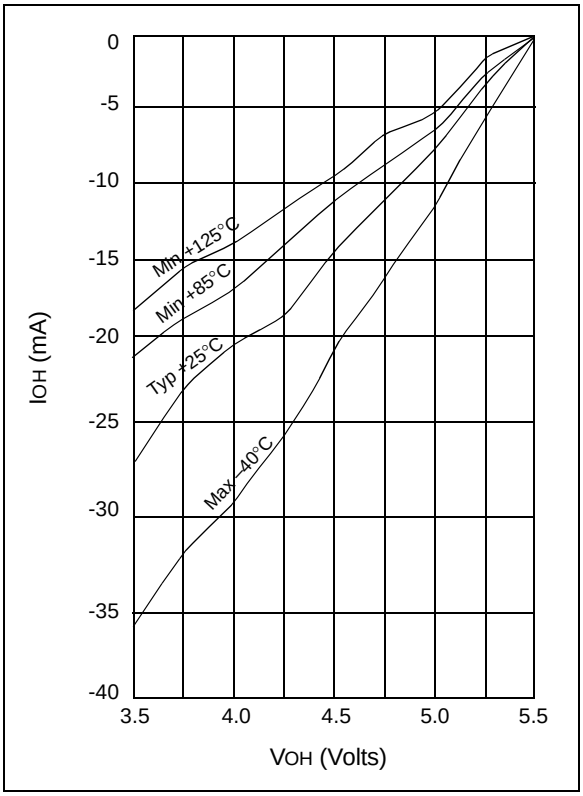



**FIGURE 13-7: IOL vs. VOL, VDD = 2.5V**

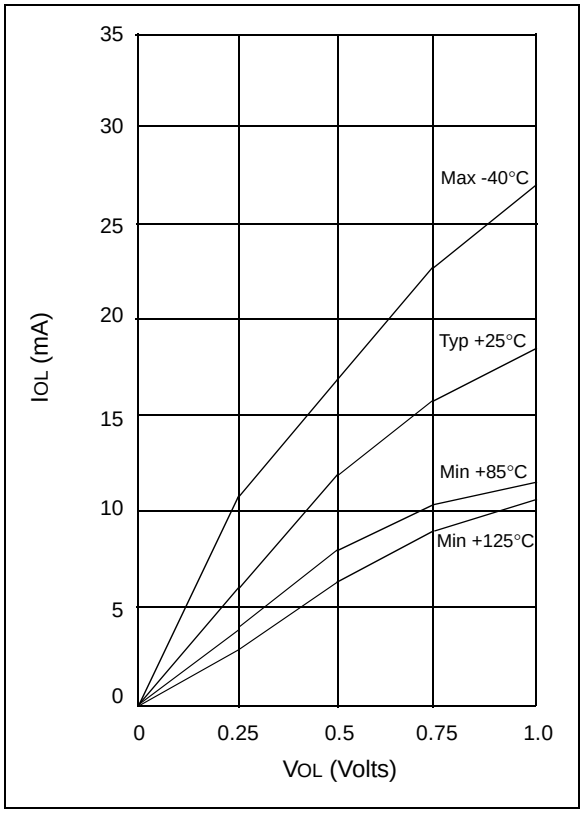



**FIGURE 13-8: IOL vs. VOL, VDD = 3.5V**

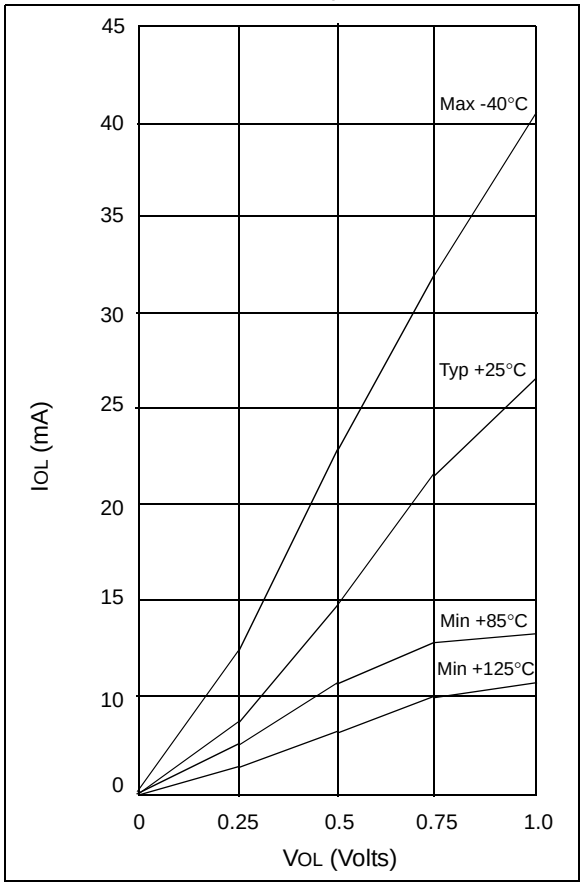


**FIGURE 13-9: IOL vs. VOL, VDD = 5.5V**

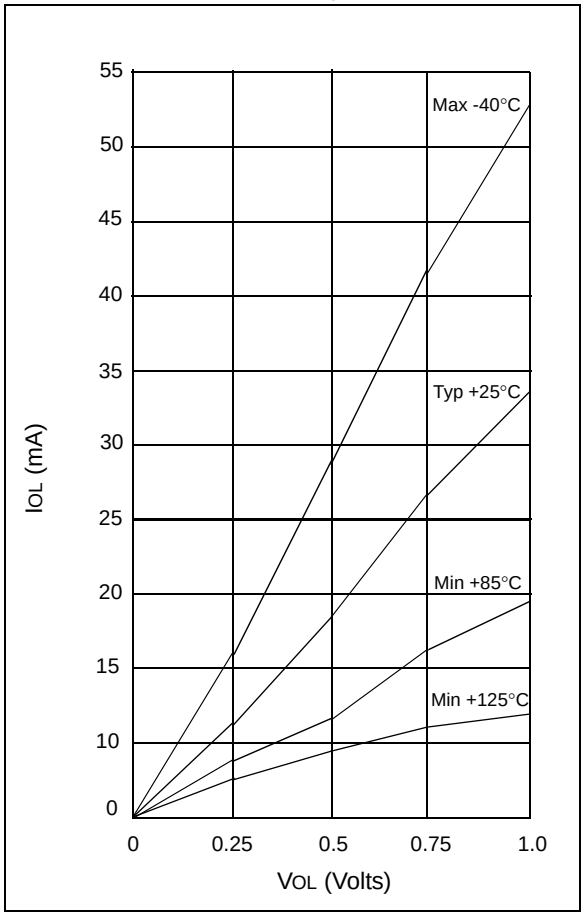



**FIGURE 13-10: VTH (INPUT THRESHOLD VOLTAGE) OF GPIO PINS vs. VDD**

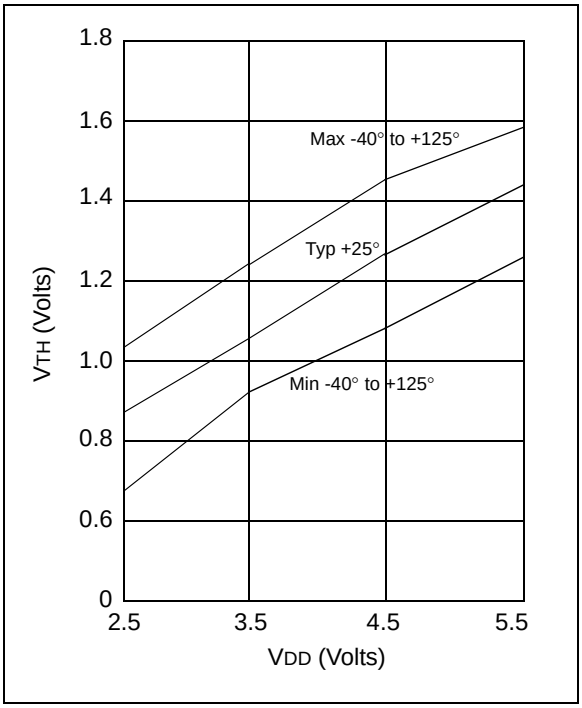
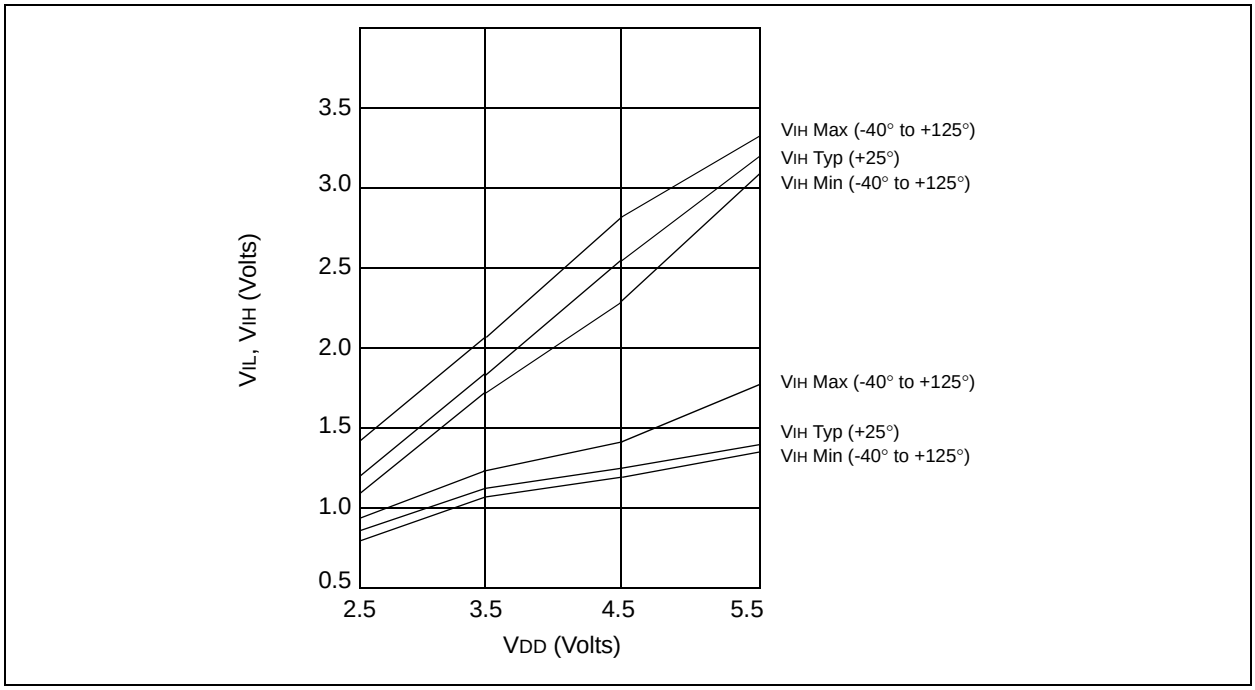


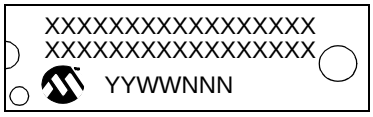

**FIGURE 13-11: VIL, VIH OF NMCLR AND T0CKI vs. VDD**



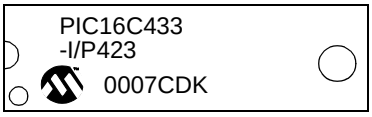
## 14.0 PACKAGING INFORMATION

### 14.1 Package Marking Information

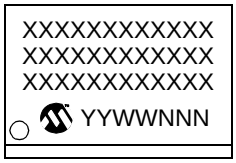
18-Lead PDIP




Example

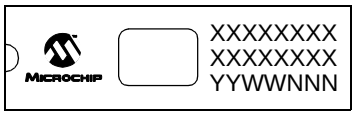



18-Lead SOIC (.300")

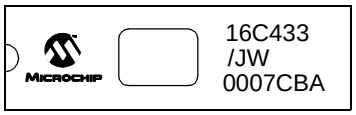



Example


18-Lead CERDIP Windowed




Example

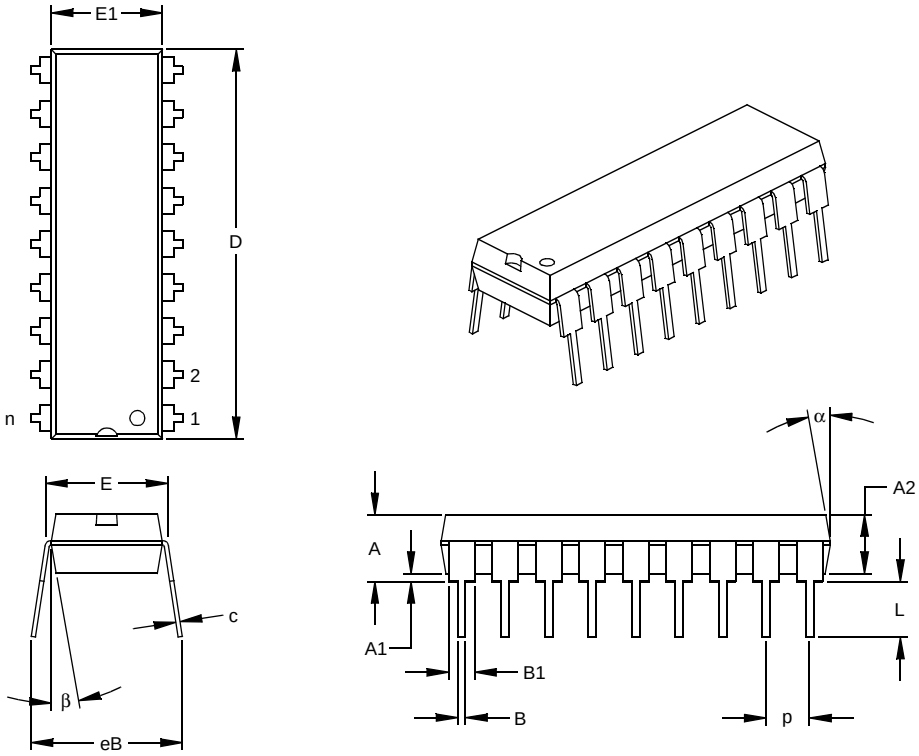



| | | |
|---|---|---|
| **Legend:** | XX...X | Customer specific information* |
| | YY | Year code (last 2 digits of calendar year) |
| | WW | Week code (week of January 1 is week '01') |
| | NNN | Alphanumeric traceability code |

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

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

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

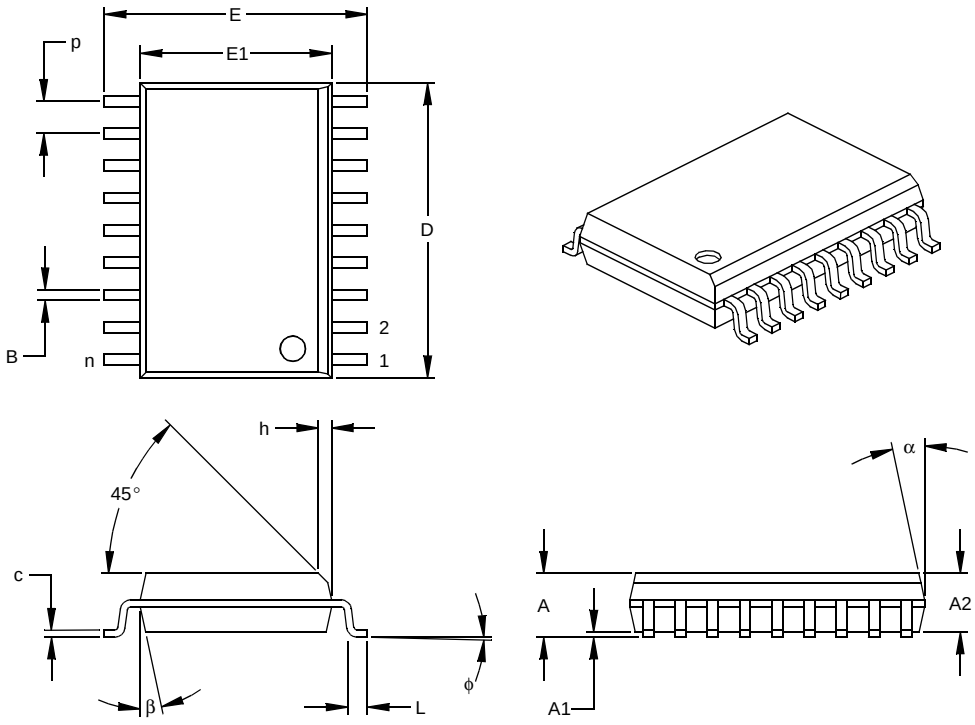



| Units | | INCHES* | | | MILLIMETERS | | |
|---|---|---|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 18 | | | 18 | |
| Pitch | p | | .100 | | | 2.54 | |
| Top to Seating Plane | A | .140 | .155 | .170 | 3.56 | 3.94 | 4.32 |
| Molded Package 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

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

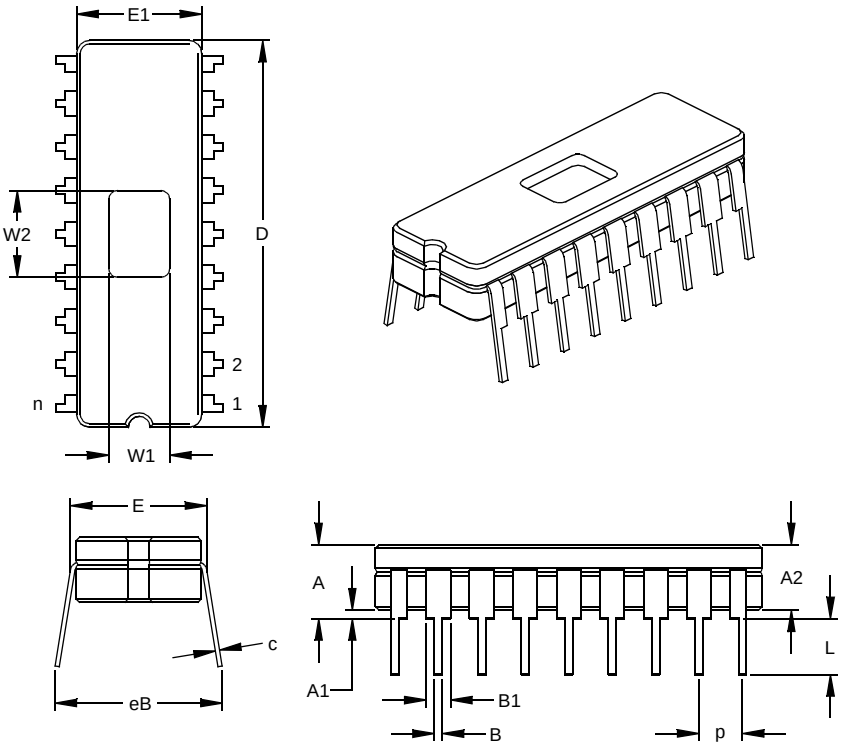



| Units | | INCHES* | | | MILLIMETERS | | |
|---|---|---|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX | MIN | NOM | MAX |
| Number of Pins | n | | 18 | | | 18 | |
| Pitch | p | | .050 | | | 1.27 | |
| Overall Height | A | .093 | .099 | .104 | 2.36 | 2.50 | 2.64 |
| Molded Package Thickness | 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 | f | 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

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




| 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

## APPENDIX A: COMPATIBILITY

To convert code written for PIC16C5X to PIC16C433, 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.

**NOTES:**

## INDEX

## ON-LINE SUPPORT

Microchip provides on-line support on the Microchip World Wide Web (WWW) site.

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

### Connecting to the Microchip Internet Web Site

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

**www.microchip.com**

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

**ftp://ftp.microchip.com**

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

- Latest Microchip Press Releases
- Technical Support Section with Frequently Asked Questions
- Design Tips
- Device Errata
- Job Postings
- Microchip Consultant Program Member Listing
- Links to other useful web sites related to Microchip Products
- Conferences for products, Development Systems, technical information and more
- Listing of seminars and events

### Systems Information and Upgrade Hot Line

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

1-800-755-2345 for U.S. and most of Canada, and

1-480-792-7302 for the rest of the world.

013001

## 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 Data Sheet.

To: Technical Publications Manager Total Pages Sent

RE: Reader Response

From: Name _____

Company _____

Address _____

City / State / ZIP / Country _____

Telephone: (_____) _____ - _____ FAX: (_____) _____ - _____

Application (optional):

Would you like a reply? ___Y ___N

Device: **PIC16C433** Literature Number: **DS41139A**

Questions:

1. What are the best features of this document?

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

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

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

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

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

7. How would you improve this document?

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

## PRODUCT IDENTIFICATION SYSTEM

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

| PART NO. | X | /XX | XXX |
|---|---|---|---|
| Device | Temperature Range | Package | Pattern |

| | | |
|---|---|---|
| Device | PIC16C433<br>PIC16LC433<br>PIC16C433T (Tape & reel for SOIC only)<br>PIC16LC433T (Tape & reel for SOIC only) | |
| Temperature Range | I<br>E | = -40°C to +85°C<br>= -40°C to +125°C |
| Package | P<br>JW*<br>SM | = PDIP<br>= Windowed CERDIP<br>= SOIC |
| Pattern | Special Requirements | |

**Examples:**

a) PIC16C433-I/P = Industrial temp., PDIP, 4 MHz - 10 MHz, normal $V_{DD}$ limits

b) PIC16C433-E/P = Extended temp., PDIP, 4 MHz - 10 MHz, normal $V_{DD}$ 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:

1. Your local Microchip sales office
2. The Microchip Corporate Literature Center U.S. FAX: (480) 792-7277
3. The Microchip Worldwide Site (www.microchip.com)

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

**New Customer Notification System**

Register on our web site (www.microchip.com/cn) to receive the most current information on our products.

# 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: 480-792-7627
Web Address: http://www.microchip.com

**Rocky Mountain**
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7966 Fax: 480-792-7456

**Atlanta**
500 Sugar Mill Road, Suite 200B
Atlanta, GA 30350
Tel: 770-640-0034 Fax: 770-640-0307

**Austin**
Analog Product Sales
8303 MoPac Expressway North
Suite A-201
Austin, TX 78759
Tel: 512-345-2030 Fax: 512-345-6085

**Boston**
2 Lan Drive, Suite 120
Westford, MA 01886
Tel: 978-692-3848 Fax: 978-692-3821

**Boston**
Analog Product Sales
Unit A-8-1 Millbrook Tarry Condominium
97 Lowell Road
Concord, MA 01742
Tel: 978-371-6400 Fax: 978-371-0050

**Chicago**
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 630-285-0071 Fax: 630-285-0075

**Dallas**
4570 Westgrove Drive, Suite 160
Addison, TX 75001
Tel: 972-818-7423 Fax: 972-818-2924

**Dayton**
Two Prestige Place, Suite 130
Miamisburg, OH 45342
Tel: 937-291-1654 Fax: 937-291-9175

**Detroit**
Tri-Atria Office Building
32255 Northwestern Highway, Suite 190
Farmington Hills, MI 48334
Tel: 248-538-2250 Fax: 248-538-2260

**Los Angeles**
18201 Von Karman, Suite 1090
Irvine, CA 92612
Tel: 949-263-1888 Fax: 949-263-1338

**Mountain View**
Analog Product Sales
1300 Terra Bella Avenue
Mountain View, CA 94043-1836
Tel: 650-968-9241 Fax: 650-967-1590

**New York**
150 Motor Parkway, Suite 202
Hauppauge, NY 11788
Tel: 631-273-5305 Fax: 631-273-5335

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

**Toronto**
6285 Northam Drive, Suite 108
Mississauga, Ontario L4V 1X5, Canada
Tel: 905-673-0699 Fax: 905-673-6509

## ASIA/PACIFIC

**Australia**
Microchip Technology Australia Pty Ltd
Suite 22, 41 Rawson Street
Epping 2121, NSW
Australia
Tel: 61-2-9868-6733 Fax: 61-2-9868-6755

**China - Beijing**
Microchip Technology Beijing Office
Unit 915
New China Hong Kong Manhattan Bldg.
No. 6 Chaoyangmen Beidajie
Beijing, 100027, No. China
Tel: 86-10-85282100 Fax: 86-10-85282104

**China - Shanghai**
Microchip Technology Shanghai Office
Room 701, Bldg. B
Far East International Plaza
No. 317 Xian Xia Road
Shanghai, 200051
Tel: 86-21-6275-5700 Fax: 86-21-6275-5060

**Hong Kong**
Microchip Asia Pacific
RM 2101, Tower 2, Metroplaza
223 Hing Fong Road
Kwai Fong, N.T., Hong Kong
Tel: 852-2401-1200 Fax: 852-2401-3431

**India**
Microchip Technology Inc.
India Liaison Office
Divyasree Chambers
1 Floor, Wing A (A3/A4)
No. 11, O'Shaugnessey Road
Bangalore, 560 025, India
Tel: 91-80-2290061 Fax: 91-80-2290062

**Japan**
Microchip Technology Intl. Inc.
Benex S-1 6F
3-18-20, Shinyokohama
Kohoku-Ku, Yokohama-shi
Kanagawa, 222-0033, Japan
Tel: 81-45-471- 6166 Fax: 81-45-471-6122

## ASIA/PACIFIC (continued)

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

**Singapore**
Microchip Technology Singapore Pte Ltd.
200 Middle Road
#07-02 Prime Centre
Singapore, 188980
Tel: 65-334-8870 Fax: 65-334-8850

**Taiwan**
Microchip Technology Taiwan
11F-3, No. 207
Tung Hua North Road
Taipei, 105, Taiwan
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139

## EUROPE

**Denmark**
Microchip Technology Denmark ApS
Regus Business Centre
Lautrup hoj 1-3
Ballerup DK-2750 Denmark
Tel: 45 4420 9895 Fax: 45 4420 9910

**France**
Arizona Microchip Technology SARL
Parc d'Activite du Moulin de Massy
43 Rue du Saule Trapu
Batiment A - ler Etage
91300 Massy, France
Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79

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

**Germany**
Analog Product Sales
Lochhamer Strasse 13
D-82152 Martinsried, Germany
Tel: 49-89-895650-0 Fax: 49-89-895650-22

**Italy**
Arizona Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Taurus 1 V. Le Colleoni 1
20041 Agrate Brianza
Milan, Italy
Tel: 39-039-65791-1 Fax: 39-039-6899883

**United Kingdom**
Arizona Microchip Technology Ltd.
505 Eskdale Road
Winnersh Triangle
Wokingham
Berkshire, England RG41 5TU
Tel: 44 118 921 5869 Fax: 44-118 921-5820

01/30/01