PIC14000

28-Pin Programmable Mixed Signal Controller

FEATURES

High-Performance RISC-like CPU core

- Based on PIC16C74 microcontroller
- Only 35 single word instructions to learn
- All single cycle instructions except for program branches which are two cycle
- Operating speed: DC - 20 MHz clock input
- 4096 x 14 on-chip EPROM program memory
- 192 x 8 general purpose registers (SRAM)
- 6 internal and 5 external interrupt sources
- 38 special function hardware registers
- Eight-level hardware stack

Analog Peripherals Features

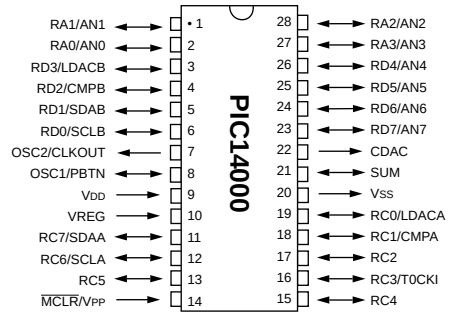
- Slope Analog-to-Digital (A/D) converter
  - Eight external input channels
    - Two channels with selectable input ranges of -0.3V to V_{DD} -2.0V or 0V to V_{DD} -1.5V
  - Seven internal input channels
  - Programmable A/D resolution, up to 16 bits
  - 16 ms maximum conversion time at maximum (16-bit) resolution and 4 MHz clock
  - 4-bit current DAC
  - Internal bandgap voltage reference
  - Factory calibrated with calibration constants stored in EPROM
  - Provisions for measuring energy in high frequency pulses (e.g. GSM cellular telephone)
- On-chip temperature sensor
- Voltage regulator control output for 5V operation
- Two multi-range DACs for programmable level/window detect or constant current/voltage charge control
- On-chip low voltage detector

Special Microcontroller Features

- Power-on Reset (POR), Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Multi-segment programmable code-protection
- Selectable oscillator options
  - Internal 4 MHz oscillator
  - External crystal oscillator
- Two pin serial in-system EPROM programming

PACKAGE TYPES

PDIP, SOIC, SSOP, Windowed CERDIP



Digital Peripherals Features

- 20 I/O pins with individual direction control
- High current sink/source for direct LED drive
- ADTMR: A/D counter, 16-bit counter with pre-load and capture
- TMR0: 8-bit timer/counter with 8-bit programmable prescaler
- I^2C^{TM} serial port compatible with:
  - ACCESS.bus™
  - SMBus (System Management Bus)

CMOS Technology

- Low-power, high-speed CMOS EPROM technology
- Fully static design
- Wide-operating voltage range (2.7V to 6.0V)
- Commercial and Industrial Temperature Range
- Low power dissipation (typical)
  - < 3 mA @5V, 4 MHz operating mode
  - < 200 μA @3V (Sleep mode: clocks stopped with analog circuits active)
  - < 5 μA @3V (Hibernate mode: clocks stopped and analog inactive)

APPLICATIONS

- Battery chargers
- Battery Capacity Monitoring
- Uninterruptable power supply controllers
- Power Management Controllers
- HVAC controllers
- Sensing and Data Acquisition

TABLE OF CONTENTS

To Our Valued Customers

We constantly strive to improve the quality of all our products and documentation. To this end, we recently converted to a new publishing software package which we believe will enhance our entire documentation process and product. As in any conversion process, information may have accidently been altered or deleted. We have spent an exceptional amount of time to ensure that these documents are correct. However, we realize that we may have missed a few things. If you find any information that is missing or appears in error, please use the reader response form in the back of this data sheet to inform us. We appreciate your assistance in making this a better document.

1.0 GENERAL DESCRIPTION

The PIC14000 is a low-cost, high-performance, CMOS, fully-static, mixed signal, factory calibrated, 8-bit microcontroller. Its features include medium to high resolution A/D conversion (10 to 16 bits), temperature sensing, closed loop charge control, serial communication, and low power operation.

The PIC14000 is based on the high-performance PIC16C74 core. It uses a RISC-like Harvard architecture CPU with separate 14-bit instruction and 8-bit data buses. A 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 are available. Additionally, a large register set is included.

PIC16/17 microcontrollers typically achieve a 2:1 code compression and a 4:1 speed improvement over other 8-bit microcontrollers.

Features:

The PIC14000 is a 28-pin device with these features:

- 4K of EPROM
- 192 bytes of RAM
- 20 I/O pins

The analog peripherals include:

- 8 external analog input channels, including 2 current sense input channels
- 6 internal analog input channels
- 2 charge-control channels with comparators
- A bandgap reference
- An internal temperature sensor
- A programmable current source for dynamic range control of the A/D conversion
- Two 8-bit logarithmic DACs

In addition, the I^2C serial port through a multiplexer supports two separate I^2C channels.

A special oscillator option allows either an internal 4 MHz oscillator or an external crystal oscillator. Using the internal 4 MHz oscillator requires no external components. Using the external oscillator option requires the PIC14000 to be in HS mode.

The PIC14000 contains two timers, WDT and TMR0. The Watchdog Timer (WDT) includes its own on-chip RC oscillator providing protection against software lock-up. TMR0 is a general purpose 8-bit timer/counter with an 8-bit prescaler. It may be clocked externally using the RC3/T0CKI pin.

Internal low-voltage detect circuit allows for tracking of voltage levels. Upon detecting the low voltage condition, the PIC14000 can be instructed to save its operating state then enter reset mode.

The internal band-gap reference is used for calibrating the measurements of the analog peripherals. The calibration factors are stored in EPROM and can be used to achieve high measurement accuracy.

Power savings modes are required for in-battery pack applications. The SLEEP and HIBERNATE modes offer different levels of power savings. The PIC14000 can wake up from these modes through interrupts or reset.

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

The PIC14000 fits perfectly in applications for battery charging, capacity monitoring, and data logging. The EPROM technology makes customization of application programs (battery characteristics, feature sets, etc.) extremely fast and convenient. The small footprint packages make this microcontroller based mixed signal device perfect for all applications with space limitations. Low-cost, low-power, high performance, ease of use and I/O flexibility make the PIC14000 very versatile in other applications such as temperature monitors/controllers and transducer compensators.

1.1 Family and Upward Compatibility

Code written for PIC16C6X/7X can be easily ported to the PIC14000 (see Appendix A).

1.2 Development Support

The PIC14000 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 DEVICE VARIETIES

A variety of frequency ranges and packaging options are available. The PIC14000 Product Selection System section at the end of this data sheet provides the devices options to be selected for your specific application and production requirements. When placing orders, please use the “PIC14000 Product Identification System” at the back of this data sheet to specify the correct part number.

2.1 UV Erasable Devices

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

The UV erasable version can be erased and reprogrammed to any of the configuration modes.

Note: Please note that erasing the device erases the pre-programmed calibration factors. Before erasing the device, read out the calibration information, store these values, and use them when programming the program memory. Please refer to the section on Program Memory for more details on these locations.

Microchip's PICSTART™ and PRO MATE™ programmers both support programming of the PIC14000. Third party programmers are also available, refer to Microchip’s Third Party Guide for a list of sources.

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 or small volume applications.

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

2.3 Quick-Turnaround-Production (QTP) Devices

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

2.4 Serialized Quick-Turnaround Production (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 PIC14000 addresses 4K x 14 program memory. All program memory is internal. The PIC14000 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 PIC14000 has an orthogonal 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 PIC14000 simple yet efficient. In addition, the learning curve is reduced significantly.

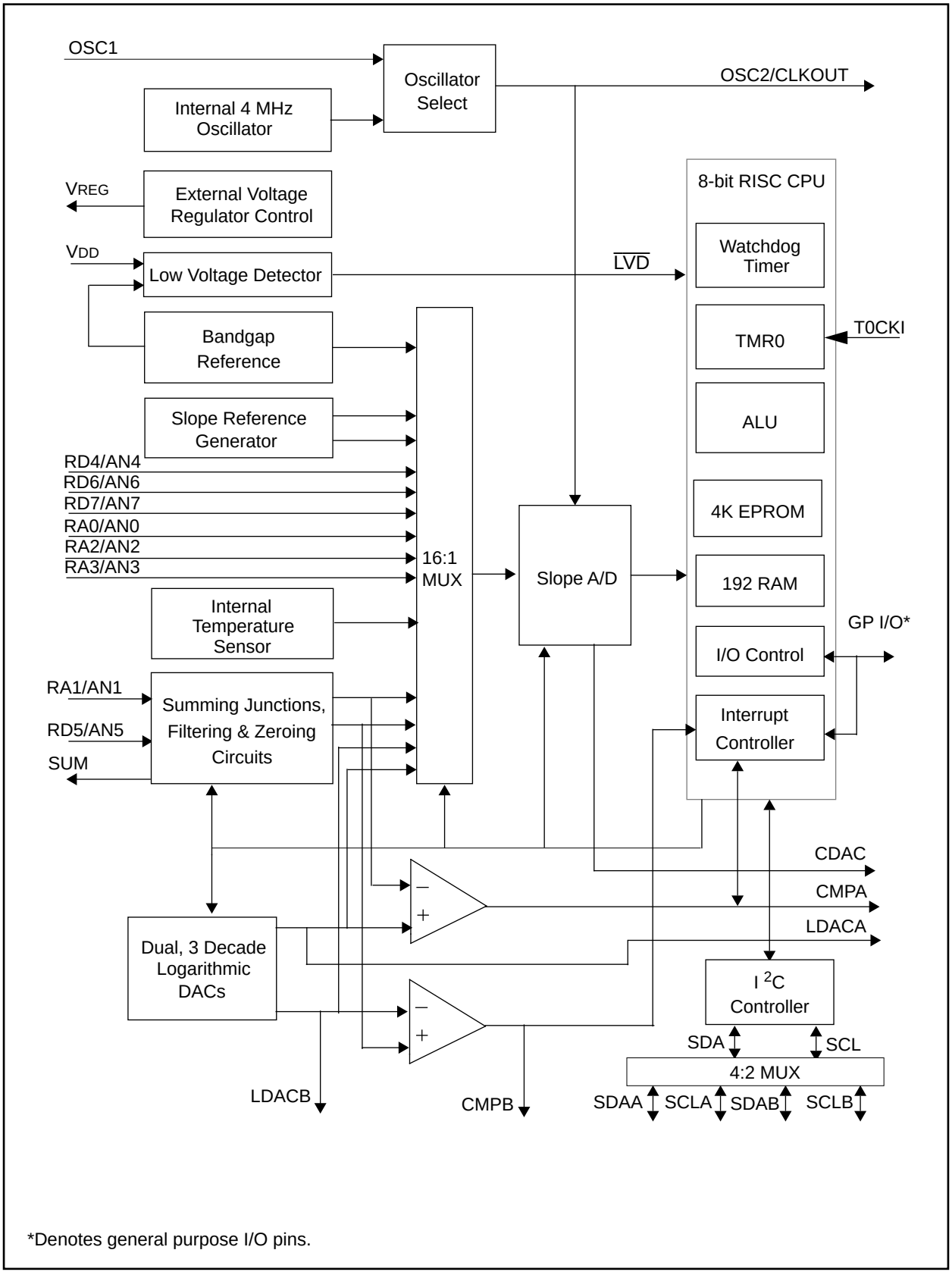
The PIC14000 contains an 8-bit ALU and working register. The ALU performs arithmetic and Boolean functions between data in the working register and any register file.

The ALU is capable of addition, subtraction, shift, and logical operations. Unless otherwise mentioned, arithmetic operations are two's complement. 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.

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

A simplified block diagram for the PIC14000 is shown in Figure 3-1, its corresponding pin description is shown in Table 3-1.

FIGURE 3-1: PIC14000 SIMPLIFIED BLOCK DIAGRAM



*Denotes general purpose I/O pins.

TABLE 3-1: PIN DESCRIPTIONS

PIN NAME	PIN NO.	I/O	PIN TYPE INPUT	PIN TYPE OUTPUT	DESCRIPTION
CDAC	22	O	-	AN	A/D ramp DAC current output. Normally connected to external capacitor to generate a linear voltage ramp.
RA0/AN0	2	I/O	AN/ST	CMOS	Analog input channel 0. This pin can also serve as a general-purpose I/O.
RA1/AN1	1	I/O	AN/ST	CMOS	Analog input channel 1. This pin has an internal summing junction and a zeroing network. If enabled, a +0.5V offset is added to the input voltage. Also contains a switch to disconnect the input voltage from the summing junction. This pin can also serve as a general-purpose I/O.
RA2/AN2	28	I/O	AN/ST	CMOS	Analog input channel 2. This pin can also serve as a general purpose digital I/O.
RA3/AN3	27	I/O	AN/ST	CMOS	Analog input channel 3. Can also serve as a general purpose digital I/O.
SUM	21	O	-	AN	AN1 summing junction output. This pin can be connected to an external capacitor for averaging small duration pulses.
RC0/LDACA	19	I/O-PU	ST	CMOS	LED direct-drive segment output or comparator A input / DAC output. This pin can also serve as a GPIO. If enabled, this pin has a weak internal pull-up to V_{DD}.
RC1/CMPA	18	I/O-PU	ST	CMOS	LED direct-drive output or comparator A output. This pin can also serve as a GPIO. If enabled, this pin has a weak internal pull-up to V_{DD}.
RC2	17	I/O-PU	ST	CMOS	LED direct-drive segment output. This pin can also serve as a GPIO. If enabled, this pin has a weak internal pull-up to V_{DD}
RC3/T0CKI	16	I/O-PU	ST	CMOS	LED direct-drive segment output. This pin can also serve as a GPIO, or an external clock input for Timer0. If enabled, this pin has a weak internal pull-up to V_{DD}.
RC4	15	I/O-PU	ST	CMOS	LED direct-drive segment output. This pin can also serve as a GPIO. If enabled, a change on this pin can cause a CPU interrupt. If enabled, this pin has a weak internal pull-up to V_{DD}.
RC5	13	I/O-PU	ST	CMOS	LED direct-drive segment output. This pin can also serve as a GPIO. If enabled, a change on this pin can cause a CPU interrupt. If enabled, this pin has a weak internal pull-up to V_{DD}.
RC6/SCLA	12	I/O	ST/SM	NPU/OC (No P-diode)	General purpose I/O. If enabled, is multiplexed as synchronous serial clock for I^2C interface. Also is the serial programming clock. If enabled, a change on this pin can cause a CPU interrupt. This pin has an N-channel pull-up device which can be disabled. Pull-ups are normally active. Programmable control for compatibility with SMBus voltage levels.
RC7/SDAA	11	I/O	ST/SM	NPU/OC (No P-diode)	General purpose I/O. If enabled, is multiplexed as synchronous serial data I/O for I^2C interface. Also is the serial programming data line. If enabled, a change on this pin can cause a CPU interrupt. This pin has an N-channel pull-up device which can be disabled. Pull-ups are normally active. Programmable control for compatibility with SM-BUS voltage levels.

TABLE 3-1: PIN DESCRIPTIONS (CONTINUED)

PIN NAME	PIN NO.	I/O	PIN TYPE INPUT	PIN TYPE OUTPUT	DESCRIPTION
RD0/SCLB	6	I/O	ST/SM	NPU/OC (No P-diode)	General purpose I/O. If enabled, is multiplexed as synchronous serial clock for I^2C interface. This pin has an N-channel pull-up device which can be disabled. Pull-ups are normally active. Programmable control for compatibility with SMBus voltage levels.
RD1/SDAB	5	I/O	ST/SM	NPU/OC (No P-diode)	General purpose I/O. If enabled, is multiplexed as synchronous serial data I/O for I^2C interface. This pin has an N-channel pull-up device which can be disabled. Pull-ups are normally active. Programmable control for compatibility with SMBus voltage levels.
RD2/CMPB	4	I/O-PU	AN/ST	CMOS	General purpose I/O or comparator B output.
RD3/LDACB	3	I/O-PU	AN/ST	CMOS	General purpose I/O or comparator B input / DAC output.
RD4/AN4	26	I/O	AN/ST	CMOS	Analog input channel 4. This pin can also serve as a general purpose digital I/O.
RD5/AN5	25	I/O	AN/ST	CMOS	Analog input channel 5. This pin has an internal summing junction and a zeroing network. If enabled, a +0.5V offset is added to the input voltage. Also contains a switch to disconnect the input voltage from the summing junction. This pin can also serve as a general-purpose I/O.
RD6/AN6	24	I/O	AN/ST	CMOS	Analog input channel 6. This pin can also serve as a general purpose digital I/O.
RD7/AN7	23	I/O	AN/ST	CMOS	Analog input channel 7. Can also serve as a general purpose digital I/O.
VREG	10	O	-	AN	This pin is an output to control the gate of an external N-FET for voltage regulation.
OSC1/PBTN	8	I-PU	ST	-	Input with weak pull-up resistor, can be used to generate an interrupt to the CPU on a falling edge, if in IN mode. Becomes oscillator input in HS mode. Connect to external crystal/resonator, or external oscillator.
OSC2/CLK-OUT	7	O	-	CMOS	Digital output in IN mode. Becomes oscillator output in HS mode. Connect to external crystal/resonator. Refer to Section 10.1 for oscillator configuration.
$\overline{MCLR}$/VPP	14	I/PWR	ST		Master clear (reset) input / programming voltage pin. This pin is an active low reset to the device.
V_{DD}	9	PWR			Positive supply connection
V_{SS}	20	GND			Return supply connection

Legend:

TYPE:	Definition:
TTL	TTL-compatible input
CMOS	CMOS-compatible input or output
ST	Schmitt trigger input, with CMOS levels
SM	SM-bus compatible input. VIL=0.6V, VIH=1.4V
OC	Open-collector output. An external pull-up resistor is required if this pin is used as an output.
NPU	N-channel pull-up. This pin will pull-up to approximately V_{DD} - 1.0V when outputting a logical '1'.
PU	Weak internal pull-up (10K-50K ohms)
No-P diode	No P-diode to V_{DD}. This pin may be pulled above the supply rail (to 6.0V maximum).
AN	Analog input or output

3.1 Clocking Scheme/Instruction Cycle

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

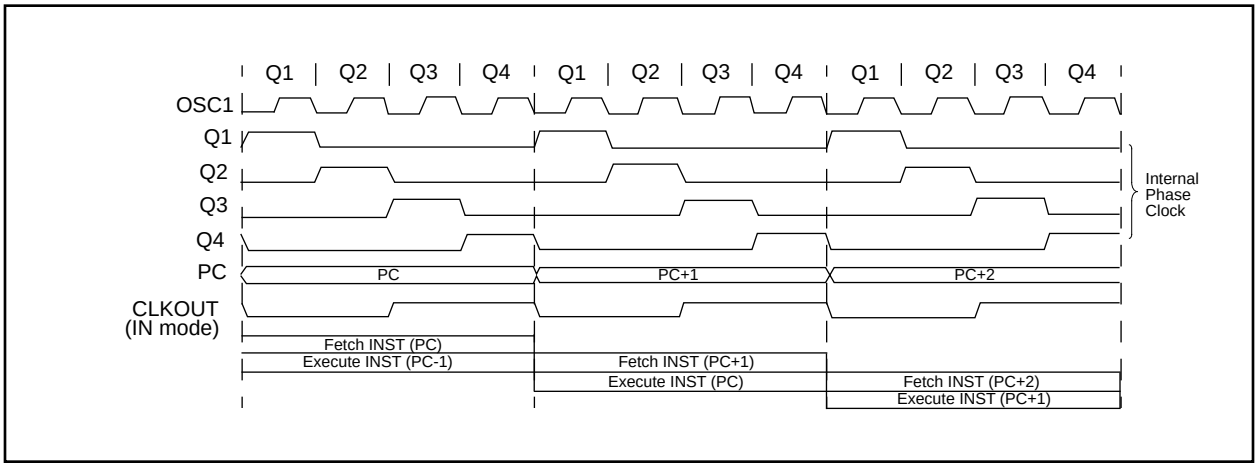
3.2 Instruction Flow/Pipelining

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

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

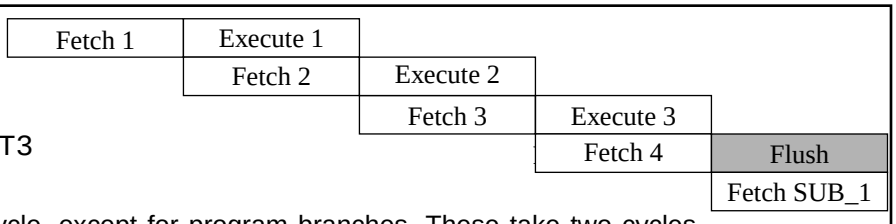
In the execution cycle, the fetched instruction is latched into the “Instruction Register (IR)” in cycle Q1. This instruction is then decoded and executed during the Q2, Q3, and Q4 cycles. Data memory is read during Q2 (operand read) and written during Q4 (destination write).

FIGURE 3-2: CLOCK/INSTRUCTION CYCLE



EXAMPLE 3-1: INSTRUCTION PIPELINE FLOW

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



All instructions are single cycle, except for 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.

NOTES:

4.0 MEMORY ORGANIZATION

4.1 Program Memory Organization

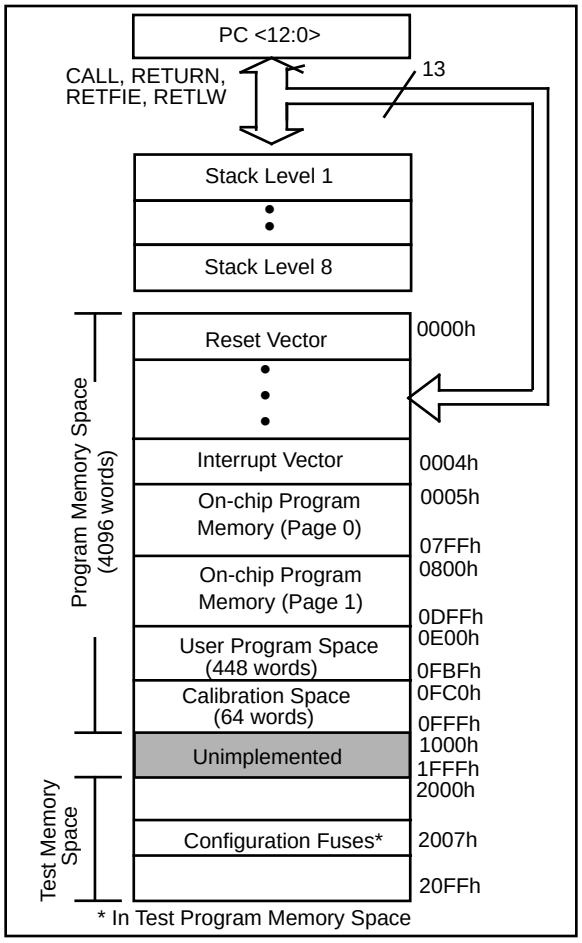
The PIC14000 family has a 13-bit program counter capable of addressing an 8K x 14 program memory space. Only the first 4K x 14 (0000-0FFFh) are physically implemented. Accessing a location above the physically implemented address will cause a wraparound. The reset vector is at 0000h and the interrupt vector is at 0004h (Figure 4-1).

The 4096 words of Program Memory space are divided into:

- Address Vectors (addr 0000h-0004h)
- Program Memory Page 0 (addr 0005h-07FFH)
- Program Memory Page 1 (addr 0800h-0DFFh)
- User Program Space (448 words, addr 0E00h-0FBFh)
- Calibration Space (64 words, addr 0FC0h-0FFFh)

Program code may reside in Page 0, Page 1, and User Program Space. PCLATH set to select Page 1 allows access to the User Program Space.

FIGURE 4-1: PIC14000 PROGRAM MEMORY MAP AND STACK



4.1.1 CALIBRATION SPACE

The Calibration Space is not used for instructions. This section stores constants and factors for the arithmetic calculations to calibrate the analog measurements.

TABLE 4-1: CALIBRATION DATA FORMATS

Address	Parameter	Symbol	Units	Min/Max	Format
0FC0h-0FC3h	Slope reference	K_{ref}	unit-less ratio	0.1/0.15	32-bit floating point**
0FC4h-0FC7h	Bandgap reference voltage	K_{bg}	Volts	1.0/1.5	32-bit floating point
0FC8h-0FCBh	Temperature sensor voltage	V_{thrm}	Volts	0.0/1.0	32-bit floating point
0FCCh-0FCFh	Temperature sensor voltage coefficient	K_{tc}	Volts/ degree Celsius	0.001/ 0.0010	32-bit floating point
0FD0h	Internal main oscillator frequency	F_{osc}	byte * 10KHz +3.0MHz		byte
0FD2h	WDT frequency (optional)	F_{wdt}	Hz	25-200	byte

** Microchip modified IEEE754 32-bit floating point format. Refer to application note AN575 for details.

TABLE 4-2: CALIBRATION CONSTANT ADDRESSES

Address	Data
0FC0	K_{ref} , exponent (eb)
0FC1	K_{ref} , mantissa high byte (f0)
0FC2	K_{ref} , mantissa middle byte (f1)
0FC3	K_{ref} , mantissa low byte (f2)
0FC4	K_{bg} , exponent (eb)
0FC5	K_{bg} , mantissa high byte (f0)
0FC6	K_{bg} , mantissa middle byte (f1)
0FC7	K_{bg} , mantissa low byte (f2)
0FC8	V_{thrm} , exponent (eb)
0FC9	V_{thrm} , mantissa high byte (f0)
0FCA	V_{thrm} , mantissa middle byte (f1)
0FCB	V_{thrm} , mantissa low byte (f2)
0FCC	K_{tc} , exponent (eb)
0FCD	K_{tc} , mantissa high byte (f0)
0FCE	K_{tc} , mantissa middle byte (f1)
0FCF	K_{tc} , mantissa low byte (f2)
0FD0	F_{in}, unsigned byte
0FD1	reserved
0FD2	F_{wdt}, unsigned byte (optional)
0FD3 - 0FFE	reserved
0FFE	calibration space checksum, low byte
0FFF	calibration space checksum, high byte

4.2 Data Memory Organization

The data memory (Figure 4-2) is partitioned into two Banks which contain the general purpose registers and the special function registers. Bank 0 is selected when the RP0 bit in the STATUS register is cleared. Bank 1 is selected when the RP0 bit in the STATUS register is set. Each Bank extends up to 7Fh (128 bytes). The first 32 locations of each Bank are reserved for the Special Function Registers. Several Special Function Registers are mapped in both Bank 0 and Bank 1. The General Purpose Registers, implemented as static RAM, are located from address 20h through 7Fh.

4.2.1 GENERAL PURPOSE REGISTER FILE

The register file is accessed either directly, or indirectly through the file select register FSR (Section 4.4).

FIGURE 4-2: REGISTER FILE MAP

File Address			
00	Indirect add.(*)	Indirect addr.(*)	80
01	TMR0	OPTION	81
02	PCL	PCL	82
03	STATUS	STATUS	83
04	FSR	FSR	84
05	PORTA	TRISA	85
06	RESERVED	RESERVED	86
07	PORTC	TRISC	87
08	PORTD	TRISD	88
09			89
0A	PCLATH	PCLATH	8A
0B	INTCON	INTCON	8B
0C	PIR1	PIE1	8C
0D			8D
0E	ADTMRL	PCON	8E
0F	ADTMRH	SLPCON	8F
10			90
11			91
12			92
13	I^2CBUF	I^2CADD	93
14	I^2CCON	I^2CSTAT	94
15	ADCAPL		95
16	ADCAPH		96
17			97
18			98
19			99
1A			9A
1B		LDACA	9B
1C		LDACB	9C
1D		CHGCON	9D
1E		MISC	9E
1F	ADCON0	ADCON1	9F
20	General Purpose Register (96 Bytes)	General Purpose Register (96 Bytes)	A0
7F			FF

* Not a physical register.
Shaded areas are unimplemented memory locations, read as '0's.

4.2.2 SPECIAL FUNCTION REGISTERS

The Special Function Registers are registers used by the CPU and Peripheral functions for controlling the desired operation of the device (Table 4-3). These registers are static RAM.

The special registers are classified into two sets. Special registers associated with the “core” functions are described in this section. Those registers related to the operation of the peripheral features are described in the section specific to that peripheral.

TABLE 4-3: SPECIAL REGISTERS FOR THE PIC14000

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
Bank0									
00 *	INDF (Indirect Address)	Addressing this location uses contents of the FSR to address data memory (not a physical register).							
01	TMR0	Timer0 data							
02*	PCL	Program Counter’s (PC’s) least significant byte							
03*	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
04*	FSR	Indirect data memory address pointer							
05	PORTA	PORTA data latch.							
06	Reserved	Reserved for emulation.							
07	PORTC	PORTC data latch							
08	PORTD	PORTD data latch							
09	Reserved								
0A	PCLATH	Buffered register for the upper 5 bits of the Program Counter (PC)							
0B	INTCON	GIE	PEIE	T0IE	r	r	T0IF	r	r
0C	PIR1	WUIF	-	-	PBIF	I^2CIF	RCIF	ADCIF	OVFIF
0D	Reserved								
0E	ADTMRL	A/D capture timer data least significant byte							
0F	ADTMRH	A/D capture timer data most significant byte							
10	Reserved								
11	Reserved								
12	Reserved								
13	I^2CBUF	I^2C Serial Port Receive Buffer/Transmit Register							
14	I^2CCON	WCOL	I^2COV	I^2CEN	CKP	I^2CM3	I^2CM2	I^2CM1	I^2CM0
15	ADCAPL	A/D capture latch least significant byte							
16	ADCAPH	A/D capture latch most significant byte							
17	Reserved								
18	Reserved								
19	Reserved								
1A	Reserved								
1B	Reserved								
1C	Reserved								
1D	Reserved								
1E	Reserved								
1F	ADCON0	ADCS3	ADCS2	ADCS1	ADCS0	-	AMUXOE	ADRST	ADZERO

Legend
— indicates unimplemented locations, read as ’0’ but cannot be overwritten
r indicates reserved locations, default is POR value and **should not be overwritten with any value**
Reserved indicates reserved register and **should not be overwritten with any value**
* indicates registers that can be addressed from either bank

TABLE 4-3: SPECIAL REGISTERS FOR THE PIC14000 (CONTINUED)

Address	Name	B7	B6	B5	B4	B3	B2	B1	B0
Bank1									
80 *	INDF (Indirect Address)	Addressing this location uses contents of FSR to address data memory (not a physical register).							
81	OPTION	$\overline{RCPU}$	r	TOCS	TOSE	PSA	PS2	PS1	PS0
82 *	PCL	Program Counter's (PC's) least significant byte							
83 *	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
84 *	FSR	Indirect data memory address pointer							
85	TRISA	PORTA Data Direction Register							
86	Reserved	Reserved for emulation							
87	TRISC	PORTC Data Direction Register							
88	TRISD	PORTD Data Direction Register							
89	Reserved								
8A	PCLATH	Buffered register for the upper 5 bits of the Program Counter (PC)							
8B	INTCON	GIE	PEIE	T0IE	r	r	T0IF	r	r
8C	PIE1	WUIE	-	-	PBIE	I^2CIE	RCIE	ADCIE	OVFIE
8D	Reserved								
8E	PCON	-	-	-	-	-	-	$\overline{POR}$	$\overline{LVD}$
8F	SLPCON	HIBEN	-	REFOFF	BIASOFF	OSCOFF	CWUOFF	TEMPOFF	ADOFF
90	Reserved								
91	Reserved								
92	Reserved								
93	I^2CADD	I^2C Synchronous Serial Port Address Register							
94	I^2CSTAT	-	-	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF
95	Reserved								
96	Reserved								
97	Reserved								
98	Reserved								
99	Reserved								
9A	Reserved								
9B	LDACA	LDASEL7	LDASEL6	LDASEL5	LDASEL4	LDASEL3	LDASEL2	LDASEL1	LDASEL0
9C	LDACB	LDBSEL7	LDBSEL6	LDBSEL5	LDBSEL4	LDBSEL3	LDBSEL2	LDBSEL1	LDBSEL0
9D	CHGCON	-	CCOMPB	CCBEN	CPOLB	-	CCOMPA	CCAEN	CPOLA
9E	MISC	SMHOG	SPGND	SPGND	I^2CSEL	SMBUS	INCLKEN	OSC2	OSC1
9F	ADCON1	ADDAC3	ADDAC2	ADDAC1	ADDAC0	ACFG3	ACFG2	ACFG1	ACFG0

Legend
— indicates unimplemented locations, read as '0' but cannot be overwritten
r indicates reserved locations, default is POR value and **should not be overwritten with any value**
Reserved indicates reserved register and **should not be overwritten with any value**
* indicates registers that can be addressed from either bank

4.2.2.1 STATUS REGISTER

The STATUS register (Address 03h or 83h) contains the arithmetic status of the ALU, the RESET status and the bank select bits for data memory.

The STATUS register (03h) can be the destination for any instruction, like any other register. If the STATUS register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. These bits are set or cleared. Furthermore, the $\overline{TO}$ and $\overline{PD}$ bits are not writable. Therefore, the result of an instruction with the STATUS register as destination may be different than intended.

For example, CLRF STATUS will clear the upper-three bits and set the Z bit. This leaves the status register as 000UU1UU (where U = unchanged).

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

Note 1: The IRP and RP1 bits (STATUS<7:6>) are not used by the PIC14000 and should be programmed as cleared. Use of these bits as general purpose R/W bits is NOT recommended, since this may affect upward compatibility with future products.

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

FIGURE 4-3: STATUS REGISTER

83h	B7	B6	B5	B4	B3	B2	B1	B0
STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
Read/Write	R/W	R/W	R/W	R	R	R/W	R/W	R/W
POR value FFh	0	0	0	1	1	X	X	X

Bit	Name	Function
B7	IRP	Not used. This bit should be programmed as ’0’. Use of this bit as a general purpose read/write bit is not recommended, since this may affect upward compatibility with future products.
B6	RP1	Not used. This bit should be programmed as ’0’. Use of this bit as a general purpose read/write bit is not recommended, since this may affect upward compatibility with future products.
B5	RP0	Register page select for direct addressing. 1 = Bank1 (80h - FFh) 0 = Bank0 (00h - 7Fh) Each page is 128 bytes. Only the RP0 bit is used.
B4	$\overline{TO}$	Time-out bit. 1 = After power-up and by the CLRWDT and SLEEP instruction. 0 = A watchdog timer time-out has occurred.
B3	$\overline{PD}$	Power down bit. 1 = After power-up or by a CLRWDT instruction. 0 = By execution of the SLEEP instruction.
B2	Z	Zero bit. 1 = The result of an arithmetic or logic operation is zero. 0 = The result of an arithmetic or logical operation is not zero.
B1	DC	Digit carry / $\overline{borrow}$ bit. For ADDWF and ADDLW instructions. 1 = A carry-out from the 4th low order bit of the result. 0 = No carry-out from the 4th low order bit of the result. Note: For Borrow, the polarity is reversed.
B0	C	Carry / $\overline{borrow}$ bit. For ADDWF and ADDLW instructions. 1 = A carry-out from the most significant bit of the result occurred. Note that 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. 0 = No carry-out from the most significant bit of the result. Note: For Borrow the polarity is reversed.

FIGURE 4-4: EVENTS AFFECTING $\overline{PD}$/$\overline{TO}$ STATUS BITS

Event	$\overline{TO}$	$\overline{PD}$	Remarks
Power-up	1	1	
WDT Time-out	0	U	No effect on $\overline{PD}$
SLEEP Instruction	1	0	Hibernate accessed via SLEEP instruction
CLRWDT	1	1	

TABLE 4-4: $\overline{PD}$/$\overline{TO}$ STATUS AFTER RESET

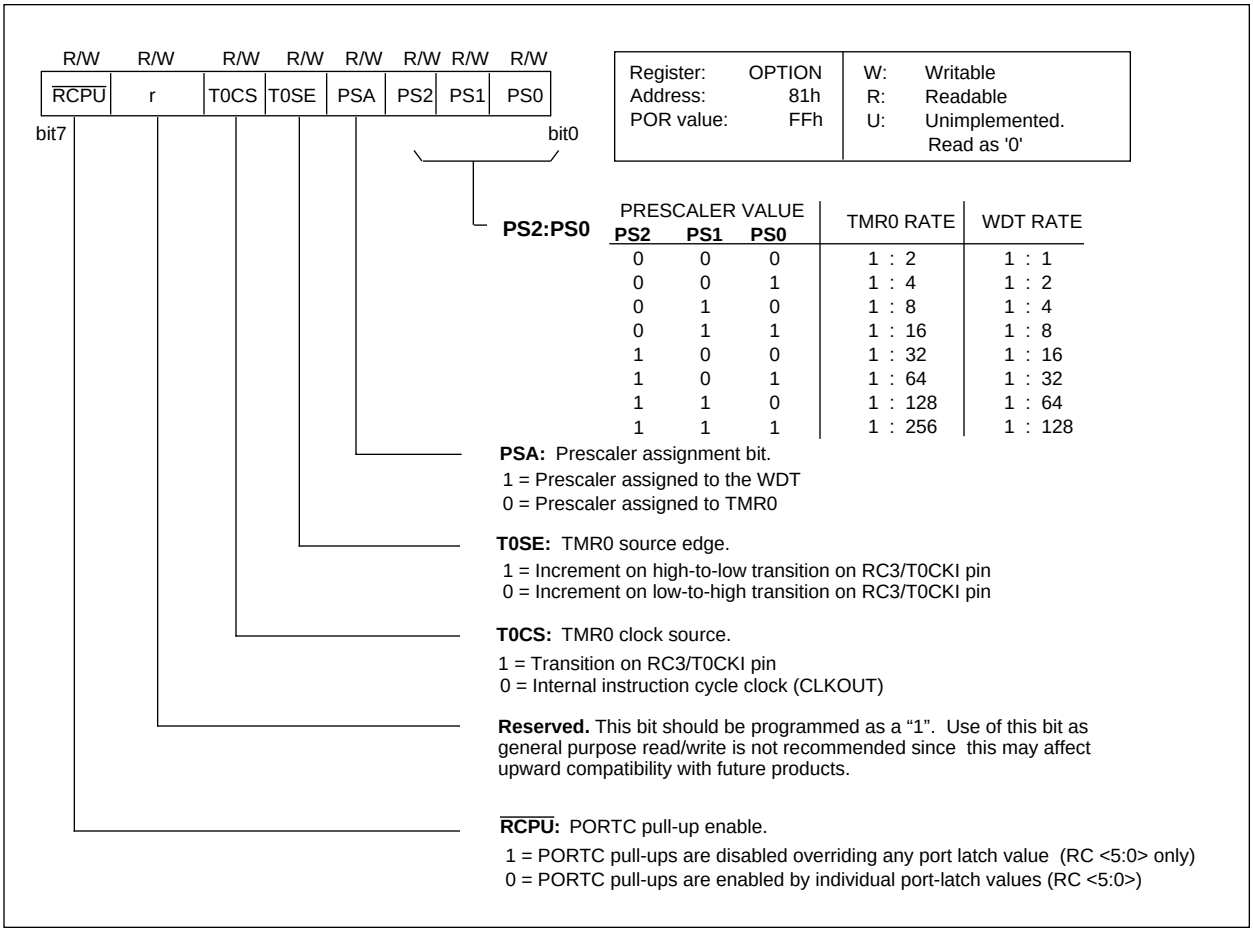
$\overline{TO}$	$\overline{PD}$	RESET was caused by
0	0	WDT Wake-up from SLEEP or Hibernate
0	1	WDT time-out (not during SLEEP)
U	0	$\overline{MCLR}$ wake-up from SLEEP or Hibernate
1	1	Power-up
U	U	$\overline{MCLR}$ reset during normal operation

4.2.2.2 OPTION REGISTER

The OPTION register (Address 81h) is a readable and writable register which contains various control bits to configure the TMR0/WDT prescaler, the external PBTN interrupt, TMR0, and the weak pull-ups on PORTC <5:0>. Bit6 is reserved in PIC14000.

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

FIGURE 4-5: OPTION REGISTER

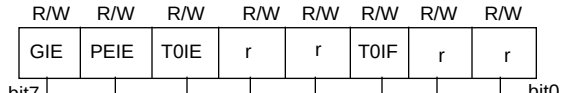


4.2.2.3 INTCON REGISTER

The INTCON Register is a readable and writable register which contains the various enable and flag bits for the Timer0 overflow, peripheral pin interrupts. Figure 4-6 shows the bits for the INTCON register for the PIC14000.

Note: The T0IF will be set by the specified condition even if the corresponding Interrupt Enable Bit is cleared (interrupt disabled) or the GIE bit is cleared (all interrupts disabled). Before enabling interrupt, clear the interrupt flag, to ensure that the program does not immediately branch to the peripheral interrupt service routine

FIGURE 4-6: INTCON REGISTER



Register:	INTCON	W:	Writable
Address:	0Bh or 8Bh	R:	Readable
POR value:	0000 000xb	U:	Unimplemented, read as '0'

Reserved. This bit should be programmed as '0'. Use of this bit as a general purpose read/write bit is not recommended, since this may affect upward compatibility with future products.

Reserved. This bit should be programmed as '0'. Use of this bit as a general purpose read/write bit is not recommended, since this may affect upward compatibility with future products.

T0IF: TMR0 overflow interrupt flag.

1 = The TMR0 has overflowed.
Must be cleared by software.
0 = TMR0 did not overflow.

Reserved. This bit should be programmed as '0'. Use of this bit as a general purpose read/write bit is not recommended, since this may affect upward compatibility with future products.

Reserved. This bit should be programmed as '0'. Use of this bit as a general purpose read/write bit is not recommended, since this may affect upward compatibility with future products.

T0IE: TMR0 interrupt enable bit.

1 = Enables T0IF interrupt
0 = Disables T0IF interrupt

PEIE: Peripheral interrupt enable bit.

1 = Enables all un-masked peripheral interrupts
0 = Disables all peripheral interrupts

GIE: Global interrupt enable.

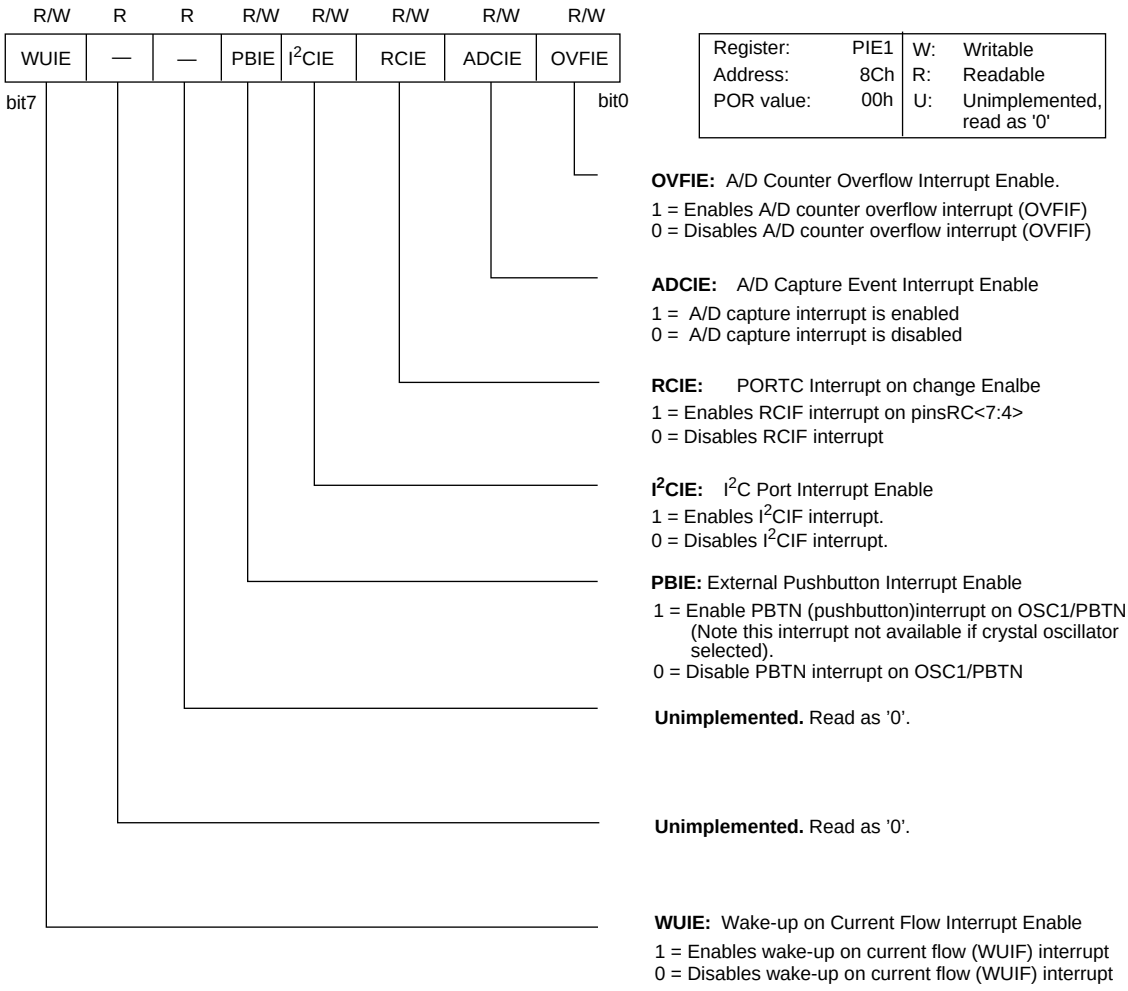
1 = Enables all un-masked interrupts
0 = Disables all interrupts

4.2.2.4 PIE1 REGISTER

This register contains the individual enable bits for the Peripheral interrupts including A/D capture event, I^2C serial port, PORTC change and A/D capture timer overflow.

Note: INTCON<6> must be enabled to enable any interrupt in PIE1.

FIGURE 4-7: PIE1 REGISTER

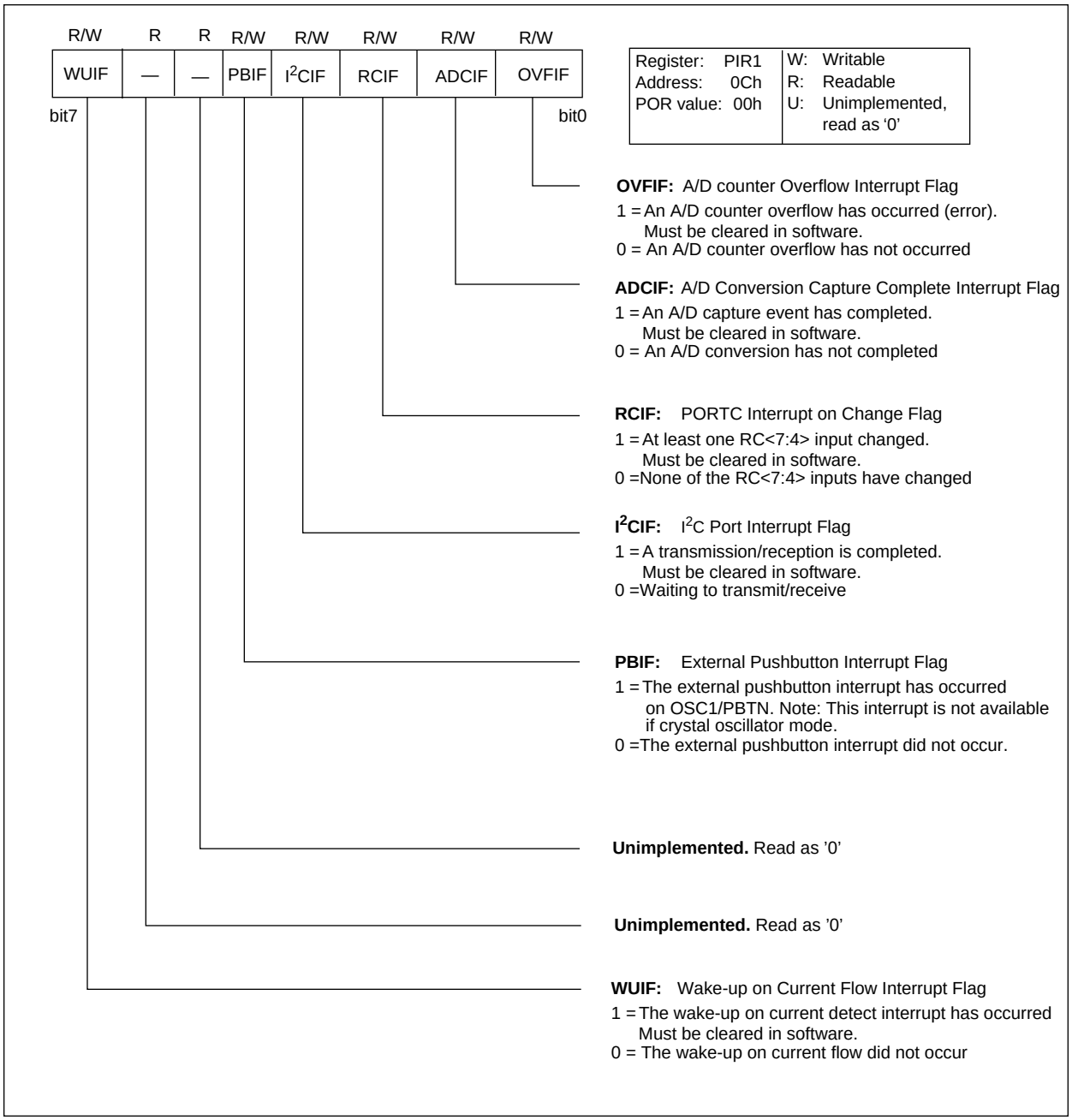


4.2.2.5 PIR1 REGISTER

This register contains the individual flag bits for the Peripheral interrupts (Figure 4-8).

Note: These bits will be set by the specified condition, even if the corresponding Interrupt Enable bit is cleared (interrupt disabled) or the GIE bit is cleared (all interrupts disabled). Before enabling an interrupt, the user may wish to clear the corresponding interrupt flag, to ensure that the program does not immediately branch to the Peripheral Interrupt service routine.

FIGURE 4-8: PIR1 REGISTER

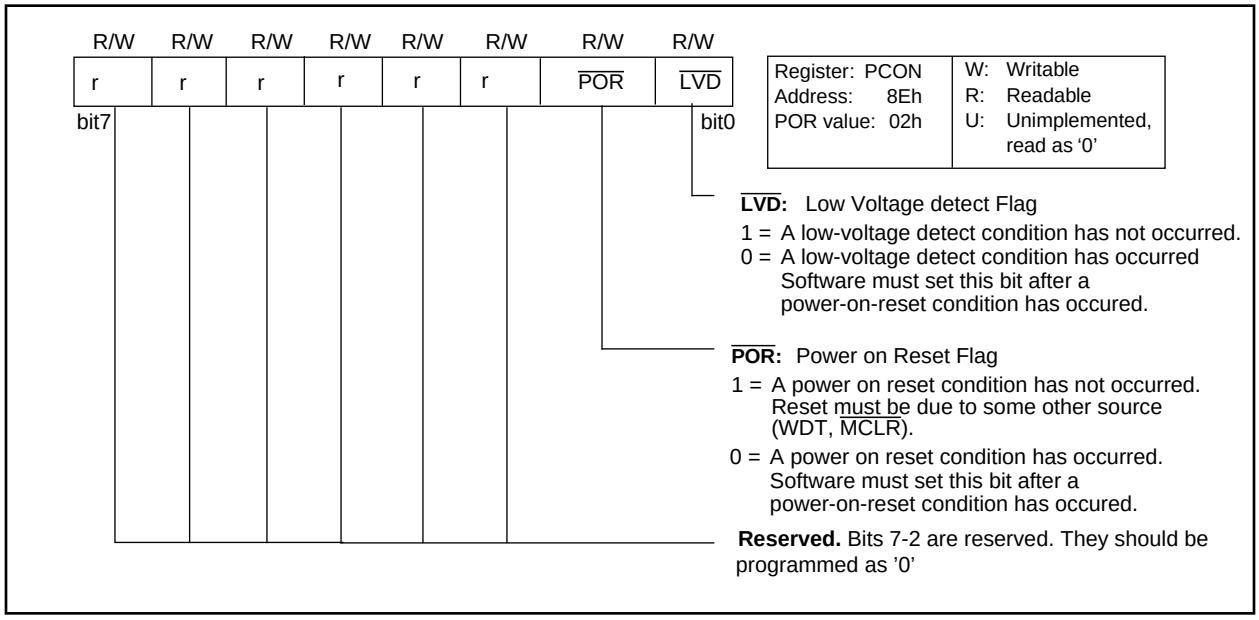


4.2.2.6 PCON REGISTER

The Power Control (PCON) register status contains 2 flag bits to allow differentiation between a Power-on Reset/low-voltage condition to an external $\overline{MCLR}$ reset, or WDT reset (Figure 4-9).

These bits are cleared on POR. The user must set these bits following POR. On a subsequent reset if POR is cleared, this is an indication that the reset was due to a power-on reset condition.

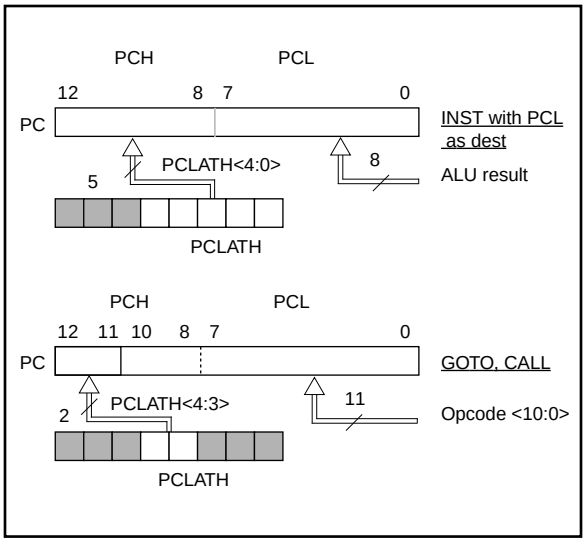
FIGURE 4-9: PCON REGISTER



4.3 PCL and PCLATH

The program counter (PC) is 13-bits wide. The low byte, PCL, is a readable and writable register. The high byte of the PC (PCH) is not directly readable or writable. PCLATH is a holding register for PC<12:8> where contents are transferred to the upper byte of the program counter. When PC is loaded with a new value during a CALL, GOTO or a write to PCL, the high bits of PC are loaded from PCLATH as shown in Figure 4-10.

FIGURE 4-10: LOADING OF PC IN DIFFERENT SITUATIONS



Note: On POR, the contents of the PCLATH register are unknown. The PCLATH should be initialized before a CALL, GOTO, or any instruction that modifies the PCL register is executed.

4.3.1 COMPUTED GOTO

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 “Table Read Using the PIC16CXX”(AN556).

4.3.2 STACK

The PIC140XX has an 8 deep x 13-bit wide hardware stack (Figure 4-1). 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 in the stack when a CALL instruction is executed or an interrupt is acknowledged. The stack is POPped in the event of a RETURN, RETLW or a RETFIE instruction execution. PCLATH is not affected by a “PUSH” or a “POP” operation.

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

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

Note 2: There are no instruction mnemonics called PUSH nor POP. These are actions that occur from the execution of the CALL, RETURN, RETLW, or RETFIE instructions, or the vectoring to an interrupt address

4.3.3 PROGRAM MEMORY PAGING

The PIC14000 has 4K of program memory, but the CALL and GOTO instructions only have a 11-bit address range. This 11-bit address range allows a branch within a 2K program memory page size. To allow CALL and GOTO instructions to address the entire 4K program memory address range, there must be another bit to specify the program memory page. This paging bit comes from the PCLATH<3> bit (Figure 4-10). When doing a CALL or GOTO instruction, the user must ensure that this page bit (PCLATH<3>) is programmed to the desired program memory page. If a CALL instruction (or interrupt) is executed, the entire 13-bit PC is pushed onto the stack. Therefore, manipulation of the PCLATH<3> is not required for the return instructions (which pops the PC from the stack).

Note: The PIC14000 ignores the PCLATH<4> bit, which is used for program memory pages 2 and 3 (1000h - 1FFFh). The use of PCLATH<4> as a general purpose read/write bit is not recommended since this may affect upward compatibility with future products.

Example 4-1 shows the calling of a subroutine in page 1 of the program memory. This example assumes that the PCLATH is saved and restored by the interrupt service routine (if interrupts are used).

EXAMPLE 4-1: CALL OF A SUBROUTINE IN PAGE 1 FROM PAGE 0

```
ORG 0X500
BSF     PCLATH, 3 ; Select page 1 (800h-FFFh)
CALL    SUB1_P1   ; Call subroutine in
        :         ; page 1 (800h-FFFh)
        :
        :
ORG     0X900
SUB1 P1 :         ; called subroutine
        :         ; page 1 (800h-FFFh)
        :
RETURN            ; return to page 0
                  ; (000h-7FFh)
```


4.4 Indirect Addressing, INDF and FSR Registers

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

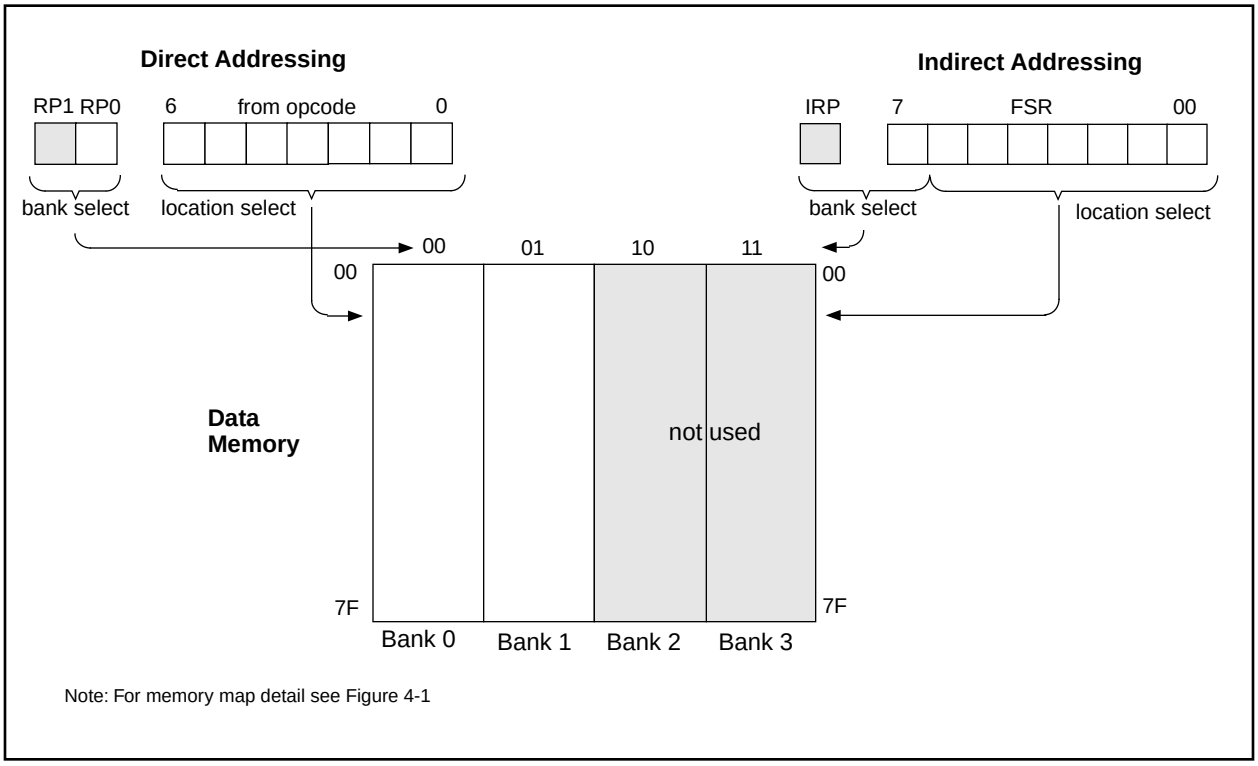
Indirect addressing is possible by using the INDF register. Any instruction using the INDF register actually accesses data pointed to by the file select register (FSR). Reading INDF itself indirectly will produce 00h. Writing to the INDF register indirectly results in a no-operation (although status bits may be affected). An effective 9-bit address is obtained by concatenating the 8-bit FSR register and the IRP bit (STATUS<7>), as shown in Figure 4-11. However, IRP is not used in the PIC14000.

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

EXAMPLE 4-2: INDIRECT ADDRESSING

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

FIGURE 4-11: INDIRECT/INDIRECT ADDRESSING



Note: For memory map detail see Figure 4-1

NOTES:

5.0 I/O PORTS

PIC14000 has three ports, PORTA, PORTC and PORTD, described in the following paragraphs. Generally, PORTA is used as the analog input port for measuring battery voltage, current and temperature. PORTC is used for general purpose I/O and for host communication. PORTD provides additional I/O lines. Four lines of PORTD may function as analog inputs. They may be used to measure individual cell voltages, for example, for rechargeable Lithium packs.

5.1 PORTA and TRISA

PORTA is a 4-bit wide port with data register located at location 05h and corresponding data direction register (TRISA) at 85h. PORTA can operate as either analog inputs for the internal A/D convertor or as general purpose digital I/O ports. These inputs are Schmitt-compatible when used as digital inputs, and have CMOS drivers as outputs. *For example, in a battery management application, the analog inputs can be connected to the battery voltage, battery current through an external sense resistor and external thermistor.*

PORTA pins are multiplexed with analog inputs. ADCFG<1:0> bits control whether these pins are analog or digital through the ADCON1 register (9Fh) as shown in Section 8.7. When configured to the digital mode, reading the PORTA register reads the status of the pins whereas writing to it will write to the port latch. When selected as an analog input, these pins will read as '0's.

Note:	On Power-on Reset, PORTA is configured as analog inputs

The TRISA register controls the direction of the PORTA pins, even when they are being used as analog inputs. The user must make sure to keep the pins configured as inputs when using them as analog inputs. A '1' in each location configures the corresponding port pin as an input. This register resets to all '1's, meaning all PORTA pins are initially inputs. The data register should be initialized prior to configuring the port as outputs. See Figure 5-2 and Figure 5-3.

PORTA inputs go through a Schmitt-compatible AND gate that is disabled when the input is in analog mode. Refer to Figure 5-1.

Note that bits RA7:RA4 are unimplemented and always read as '0'. Unused inputs should not be left floating to avoid leakage currents. All pins have input protection diodes to VDD and VSS.

EXAMPLE 5-1: INITIALIZING PORTA

```
CLRF  PORTA       ;Initialize PORTA by setting
                  ;output data latches
BSF   STATUS, RP0 ;Select Bank1
MOVLW 0x0F        ;Value used to initialize
                  ;data direction
MOVWF TRISA       ;Set RA<3:0> as inputs
                  ;RA<5:4> as outputs
                  ;TRISA<7:6> are always
                  ;read as '0'.
```

FIGURE 5-1: PORTA BLOCK DIAGRAM

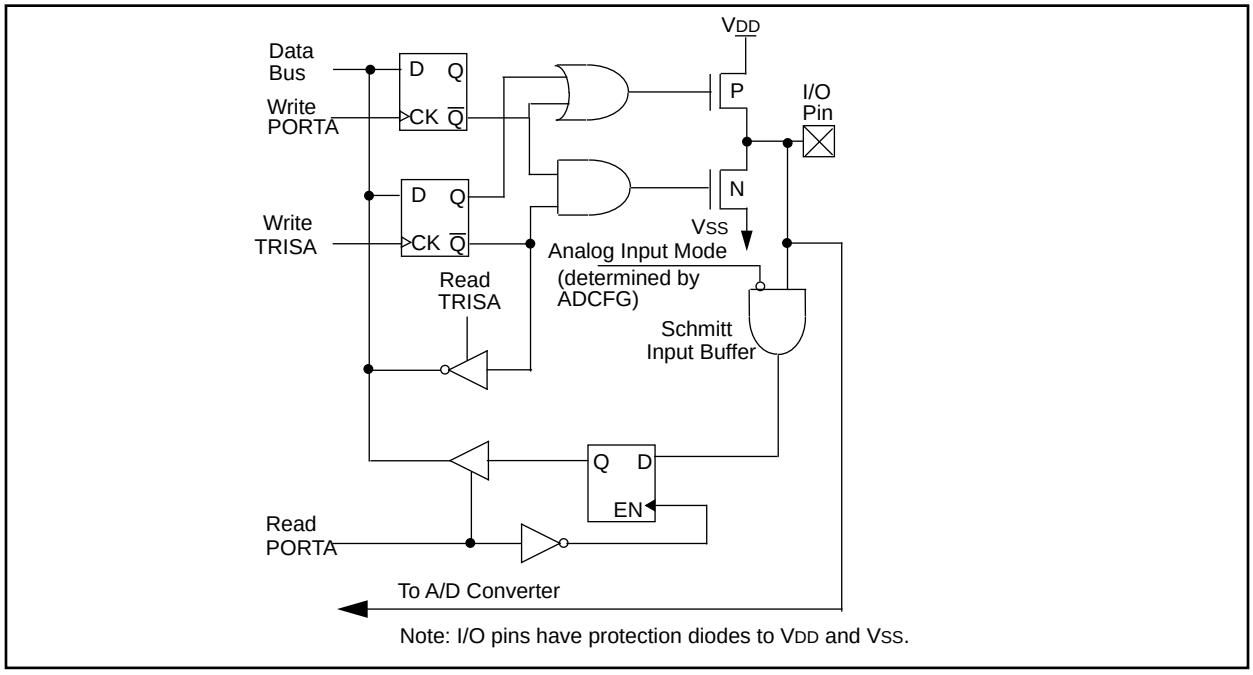


FIGURE 5-2: PORTA DATA REGISTER

05h	B7	B6	B5	B4	B3	B2	B1	B0
PORTA	-	-	-	-	RA3/AN3	RA2/AN2	RA1/AN1	RA0/AN0
Read/Write	U	U	U	U	R/W	R/W	R/W	R/W
POR value 0xh	0	0	0	0	X	X	X	X

Bit	Name	Function
B7-B4	-	Unimplemented. Reads as'0'.
B3	RA3/AN3	GPIO or analog input. Returns value on pin RA3/AN3 when used as digital input. When configured as analog input, reads as'0'.
B2	RA2/AN2	GPIO or analog input. Returns value on pin RA2/AN2 when used as digital input. When configured as analog input, reads as'0'. Can be connected to an external thermistor in a battery management application.
B1	RA1/AN1	GPIO or analog input. Returns value on pin RA1/AN1 when used as digital input. When configured as analog input, reads as'0'. Can be connected to an external current sense resistor in a battery management application. This pin has special input characteristics. See Table 3-1 for additional information.
B0	RA0/AN0	GPIO or analog input. Returns value on pin RA0/AN0 when used as digital input. When configured as analog input, reads as'0'. Normally connected to the battery voltage (through an external voltage divider if battery voltage exceeds 6.0V).

FIGURE 5-3: SUMMARY OF PORTA REGISTERS

Register Name	Function	Address	Power-on Reset Value
PORTA	PORTA pins when read PORTA data latch when written	05h	--xx xxxx[1]
TRISA	PORTA data direction register 0 = output, 1 = input	85h	--11 1111[1]
ADCON1	PORTA Analog or Digital configuration	9Fh (74/73) 88h (71)	0000 0000

Legend: x = unknown, - = unimplemented, read as a '0'. For reset values of registers in other reset situations refer to Table 14-7.
Note 1: The PIC16C71 does not have PORTA or TRISA bit 5, read as '0'.

5.2 PORTC and TRISC

PORTC is a 8-bit wide bidirectional port, with Schmitt trigger inputs, that serves the following functions depending on programming:

- Direct LED drive (PORTC<7:0>).
- I^2C communication lines (PORTC<7:6>), refer to Section 7.0 I^2C Serial Port.
- Interrupt on change function (PORTC<7:4>), discussed below and in Section 10.3 Interrupts.
- Hardware charge control for a constant current switching regulator (PORTC<1:0>). Refer to Section 9.5.
- Timer0 on RC3

The PORTC data register is located at location 06h and its data direction register (TRISC) is at 86h.

PORTC <5:0> have weak internal pull-ups (~100uA typical). A single control bit can turn on all the pull-ups. This is done by clearing bit $\overline{RCPU}$ (OPTION <7>). The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on power-on reset.

Four of the PORTC pins, RC<7:4> have an interrupt on change feature. Only pins configured as inputs can cause this interrupt to occur. In other words, any pin RC<7:4> configured as an output is excluded from the interrupt on change comparison. The input pins of RC<7:4> are compared with the old value latched on the last read of PORTC. The "mismatch" outputs of RC<7:4> are OR'ed together to assert the RCIF flag (PIR1 register <0>) and cause a CPU interrupt, if enabled.

Note: If the I^2C function is enabled, (I^2CCON <5>, address 14h), RC<7:6> are automatically excluded from the interrupt-on-change comparison.

This interrupt can wake the device up from SLEEP. The user, in the interrupt service routine, can clear the interrupt in one of two ways:

- Disable the interrupt by clearing RCIE (PIR1<3>) bit
- Read PORTC. This will end mismatch condition. Then, clear the RCIF bit.

A mismatch condition will continue to set the RCIF bit. Reading PORTC will end the mismatch condition, and allow the RCIF bit to be cleared.

If the charge control function is enabled, (bit CCAEN (CHGCON <1>) is set) the RC0/LDACA pin becomes the charge DAC analog output and should be connected to an external filter capacitor. Pin RC1/CMPA is the output from the charge control comparator and is used to switch an external power FET as part of a switching regulator.

Note:	Setting CCAEN changes the definition of RC0/LDACA and RC1/CMPA to their charge control functions, bypassing the PORTC data and TRISC register settings.

PORTC<7:6> also serve multiple functions. These pins act as the I^2C data and clock lines when the I^2C module is enabled. They also serve as the serial programming interface data and clock line for in-circuit programming of the EPROM.

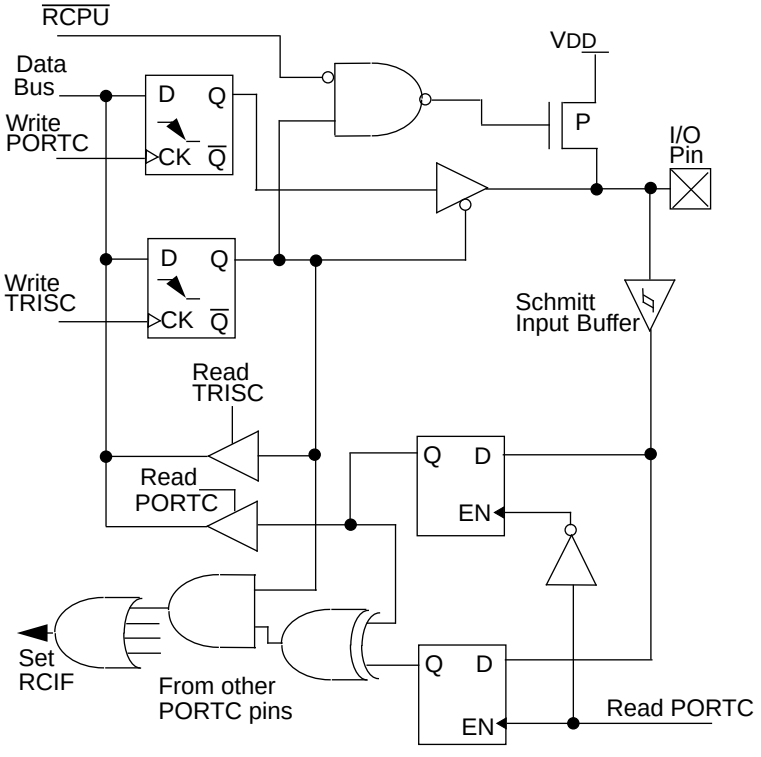
The TRISC register controls the direction of the RC pins. A '1' in each location configures the corresponding port pin as an input. This register resets to all '1's, meaning all PORTC pins are initially inputs. The data register should be initialized prior to configuring the port as outputs.

Unused inputs should not be left floating to avoid leakage currents. All pins have input protection diodes to VDD and VSS.

EXAMPLE 5-2: INITIALIZING PORTC

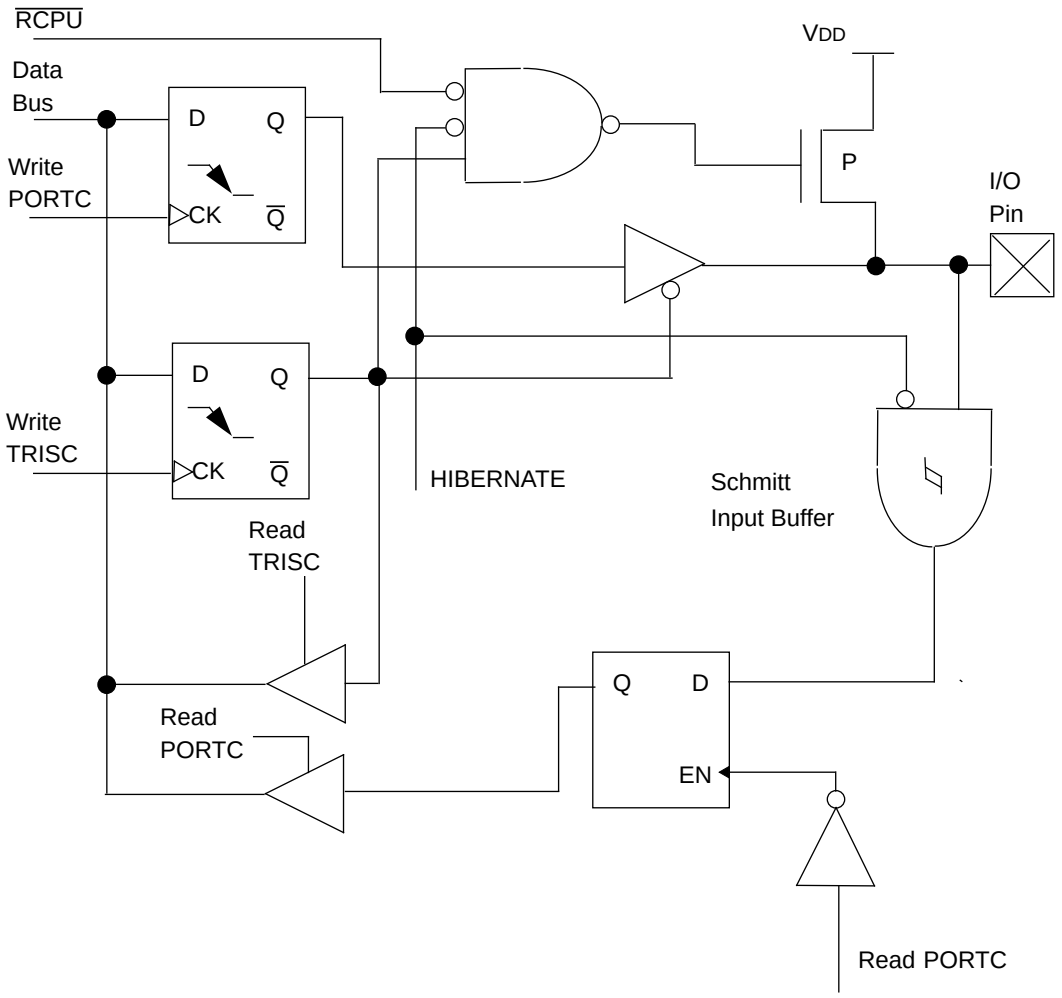
```
CLRF   PORTC        ;Initialize PORTC data
                    ;    latches before setting
                    ;    the data direction
                    ;    register
BSF    STATUS, RPO  ;Select Bank1
MOVLW  0xCF         ;Value used to initialize
                    ;data direction
MOVWF  TRISC        ;Set RC<3:0> as inputs
                    ;    RC<5:4> as outputs
                    ;    RC<7:6> as inputs
```

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



1. I/O pins have protection diodes to VDD and VSS.
2. Port Latch = '1' and TRISC = '1' enables weak pull-up if $\overline{RCPU}$ = '0' in OPTION register.
3. RC7/SDA, RC6/SCL have N-channel pull-ups to VDD which are disabled in I^2C mode.

FIGURE 5-5: BLOCK DIAGRAM OF PORTC <3:0> PINS



1. I/O pins have protection diodes to V_{DD} and V_{SS}.
2. Port Latch ='1' and TRISC ='1' enables weak pull-up if $\overline{RCPU}$ ='0' in OPTION register.
3. If the CCAEN bit in CHGCON is set to'1', RC0 becomes analog output LDACA, RC1 becomes CMPA, ignoring the PORTC<1:0> data, TRISC<1:0> register settings.

FIGURE 5-6: PORTC DATA REGISTER

07h	B7	B6	B5	B4	B3	B2	B1	B0
PORTC	RC7/SDA	RC6/SCL	RC5	RC4	RC3/TOCKI	RC2	RC1/CMPA	RC0/LDACA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR value xxh	x	x	x	x	x	x	x	x

Bit	Name	Function
B7	RC7/SDAA	Synchronous serial data I/O for I^2C interface. Also is the serial programming data line. This pin can also serve as a general purpose I/O. If enabled, a change on this pin can cause a CPU interrupt. This pin has an N-channel pull-up to VDD which is disabled in I^2C mode.
B6	RC6/SCLA	Synchronous serial clock for I^2C interface. Also is the serial programming clock. This pin can also serve as a general purpose I/O. If enabled, a change on this pin can cause a CPU interrupt. This pin has an N-channel pull-up to VDD which is disabled in I^2C mode.
B5	RC5	LED direct-drive output. This pin can also serve as a GPIO. If enabled, a change on this pin can cause a CPU interrupt. If enabled, this pin has a weak internal pull-up to VDD. The Option Register controls pull-up enable.
B4	RC4	LED direct-drive segment output. This pin can also serve as a GPIO. If enabled, a change on this pin can cause a CPU interrupt. If enabled, this pin has a weak internal pull-up to VDD. The Option Register controls pull-up enable.
B3	RC3/T0CKI	LED direct-drive segment output. This pin can also serve as a GPIO. If enabled, this pin has a weak internal pull-up to VDD. TOCKI is enabled as TMR0 clock via the OPTION register. The Option Register controls pull-up enable.
B2	RC2	LED direct-drive segment output. This pin can also serve as a GPIO. If enabled, this pin has a weak internal pull-up to VDD. The Option Register controls pull-up enable.
B1	RC1/CMPA	LED direct-drive segment output. This pin can also serve as a GPIO. As a charge controller, this pin can be used as part of a constant-current switching regulator. If enabled, this pin has a weak internal pull-up to VDD. The Option Register controls pull-up enable.
B0	RC0/LDACA	LED direct-drive segment output. This pin can also serve as a GPIO. As a charge controller, this pin can be used as part of a constant-current switching regulator. If enabled, this pin has a weak internal pull-up to VDD. The Option Register controls pull-up enable.

U= unimplemented. X = unknown.

5.2.1 TRISC PORTC DATA DIRECTION REGISTER

This register defines each pin of PORTC as either an input or output under software control. A '1' in each location configures the corresponding port pin as an input. This register resets to all '1's, meaning all PORTC pins are initially inputs. The data register should be initialized prior to configuring the port as outputs.

FIGURE 5-7: TRISC REGISTER

87h	B7	B6	B5	B4	B3	B2	B1	B0
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR value FFh	1	1	1	1	1	1	1	1

Bit	Name	Function
B7	TRISC7	Control direction on pin RC7/SDAA (has no effect if I^2C is enabled): 0 = pin is an output. 1 = pin is an input.
B6	TRISC6	Control direction on pin RC6/SCLA (has no effect if I^2C is enabled): 0 = pin is an output. 1 = pin is an input.
B5	TRISC5	Control direction on pin RC5: 0 = pin is an output. 1 = pin is an input.
B4	TRISC4	Control direction on pin RC4: 0 = pin is an output. 1 = pin is an input.
B3	TRISC3	Control direction on pin RC3: 0 = pin is an output. 1 = pin is an input.
B2	TRISC2	Control direction on pin RC2: 0 = pin is an output. 1 = pin is an input.
B1	TRISC1	Control direction on pin RC1/CMPA (has no effect if the charge control function is enabled): 0 = pin is an output. 1 = pin is an input.
B0	TRISC0	Control direction on pin RC0/LDACA (has no effect if the charge control function is enabled): 0 = pin is an output. 1 = pin is an input.

U= unimplemented. X = unknown.

5.3 PORTD and TRISD

PORTD is an 8-bit port that may be used for general purpose I/O. Four pins can be configured as analog inputs. For example, they could be used to measure individual cell voltages in rechargeable Lithium battery packs. Together with the analog channels of PORTA, 8 analog inputs are supplied.

FIGURE 5-8: BLOCK DIAGRAM OF PORTD <7:4> PINS

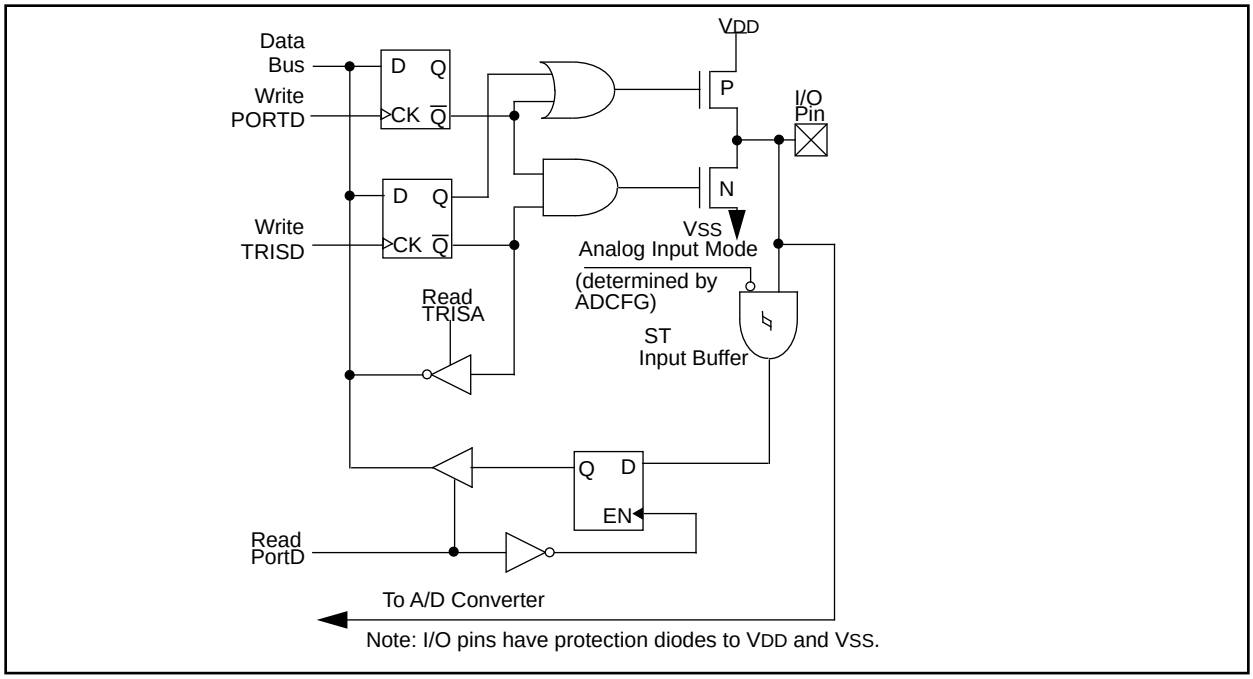
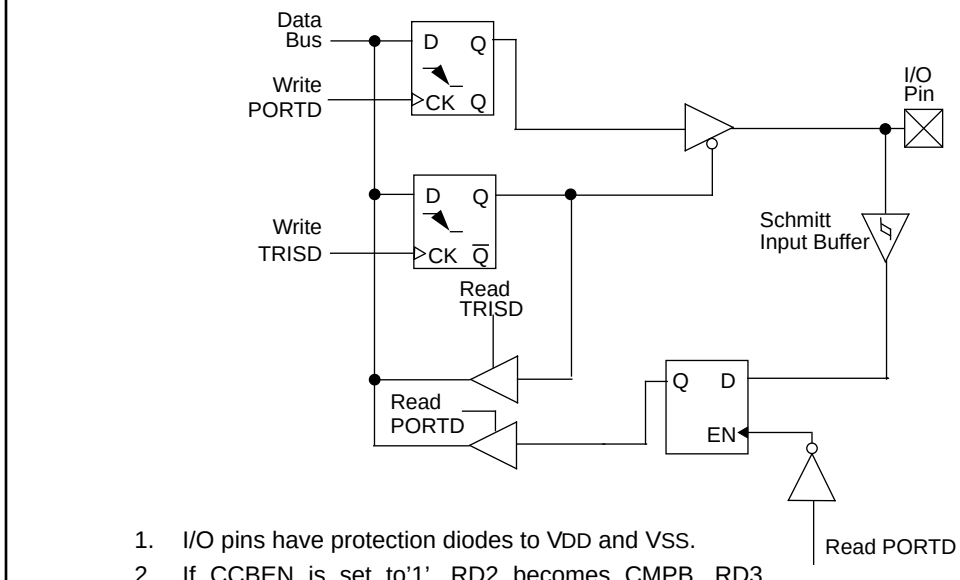


FIGURE 5-9: BLOCK DIAGRAM OF PORTD<3:0> PINS



1. I/O pins have protection diodes to VDD and VSS.
2. If CCBEN is set to'1', RD2 becomes CMPB, RD3 becomes LDACB, ignoring the PORTD<3:2> data, TRISD<1:0> register settings.

FIGURE 5-10: PORTD DATA REGISTER

08h	B7	B6	B5	B4	B3	B2	B1	B0
PORTD	RD7/AN7	RD6/AN6	RD5/AN5	RD4/AN4	RD3/LDACB	RD2/CMPB	RD1/SDAB	RD0/SCLB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR value xxh	x	x	x	x	x	x	x	x

Bit	Name	Function
B7	RD7/AN7	GPIO or analog input. Returns value on pin RD7/AN7 when used as digital input. When configured as analog input, reads as '0'.
B6	RD6/AN6	GPIO or analog input. Returns value on pin RD6/AN6 when used as digital input. When configured as analog input, reads as '0'. In a battery management application, connected to an external thermistor (channel B) for dual-pocket charge control.
B5	RD5/AN5	GPIO or analog input. Returns value on pin RD5/AN5 when used as digital input. When configured as analog input, reads as '0'. In a battery management application, connected to an external current sense resistor (channel B) for dual-pocket charge control. This pin has special input characteristics. See Table 3-1 for additional information.
B4	RD4/AN4	GPIO or analog input. Returns value on pin RD4/AN4 when used as digital input. When configured as analog input, reads as '0'. In a battery management application, connected to the battery voltage (channel B) for dual-pocket charge control.
B3	RD3/LDACB	This pin can serve as a GPIO. In a dual-pocket charge controller application, this pin can be used as part of a constant-current switching regulator for a 2nd battery pack.
B2	RD2/CMPB	This pin can serve as a GPIO. In a dual-pocket charge controller application, this pin can be used as part of a constant-current switching regulator for a 2nd battery pack.
B1	RD1/SDAB	Alternate synchronous serial data I/O for I^2C interface enabled by setting the I^2CSEL bit in the MISC register. This pin can also serve as a general purpose I/O. This pin has an N-channel pull-up which can be disabled.
B0	RD0/SCLB	Alternate synchronous serial clock for I^2C interface, enabled by setting the I^2CSEL bit in the MISC register. Also is the serial programming clock. This pin can also serve as a general purpose I/O. If enabled, a change on this pin can cause a CPU interrupt. This pin has an N-Channel pull-up which can be disabled.

Legend: U = unimplemented, read as '0'. x = unknown.

FIGURE 5-11: TRISD REGISTER

88h	B7	B6	B5	B4	B3	B2	B1	B0
TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR value FFh	1	1	1	1	1	1	1	1

Bit	Name	Function
B7	TRISD7	Control direction on pin RD7/AN7: 0 = pin is an output. 1 = pin is an input.
B6	TRISD6	Control direction on pin RD6/AN6: 0 = pin is an output. 1 = pin is an input.
B5	TRISD5	Control direction on pin RD5/AN5: 0 = pin is an output. 1 = pin is an input.
B4	TRISD4	Control direction on pin RD4/AN4: 0 = pin is an output. 1 = pin is an input.
B3	TRISD3	Control direction on pin RD3/LDACB (has no effect if the charge enable bit CCBEN is set to '1'): 0 = pin is an output. 1 = pin is an input.
B2	TRISD2	Control direction on pin RD2/CMPB (has no effect if the charge enable bit CCBEN is set to '1'): 0 = pin is an output. 1 = pin is an input.
B1	TRISD1	Control direction on pin RD1/SDAB: 0 = pin is an output. 1 = pin is an input.
B0	TRISD0	Control direction on pin RD0/SCLB: 0 = pin is an output. 1 = pin is an input.

TABLE 5-1: RESET CONDITION FOR REGISTERS

Register	Address	Power-on Reset	**$\overline{\text{MCLR}}$ reset during - normal operation - SLEEP WDT time-out during normal operation**	**Wake-up from SLEEP through interrupt Wake up from SLEEP through WDT time-out**
PORTD	08h	xxxx xxxx	uuuu uuuu	uuuu uuuu
TRISD	88h	1111 1111	1111 1111	uuuu uuuu

If the charge control function is enabled, (bit CCBEN (CHGCON <5>) is set) the RD3/LDACB pin becomes the charge DAC analog output and should be connected to an external filter capacitor. Pin RD2/CMPB is the output from the charge control comparator and is used to switch an external power FET as part of a switching regulator (Section 9.5).

Note:	Setting CCBEN changes the definition of RD3/LDACB and RD2/CMPB to their charge control functions, bypassing the PORTD data and TRISD register settings.

PORTD<1:0> also serve multiple functions. These pins act as the I^2C data and clock lines when the I^2C module is enabled.

The TRISD register controls the direction of the RD pins. A '1' in each location configures the corresponding port pin as an input. This register resets to all '1's, meaning all PORTD pins are initially inputs. The data register should be initialized prior to configuring the port as outputs.

Unused inputs should not be left floating to avoid leakage currents. All pins have input protection diodes to V_{DD} and V_{SS}.

EXAMPLE 5-3: INITIALIZING PORTD

```
CLRF   PORTD        ;Initialize PORTD data
                    ;    latches before setting
                    ;    the data direction
                    ;    register
BSF    STATUS, RPO  ;Select Bank1
MOVLW  0xFF         ;Value used to initialize
                    ;data direction
MOVWF  TRISD        ;Set RD<7:0> as inputs
```

5.4 I/O Programming Considerations

5.4.1 BI-DIRECTIONAL I/O PORTS

Reading the port register reads the values of the port pins. Writing to the port register writes the value to the port latch. Some instructions operate internally as read-modify-write. 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 PORTC will cause all eight bits of PORTC to be read into the CPU. Then the BSF operation takes place on bit5 and PORTC is written to the output latches. If another bit of PORTC is used as a bi-directional I/O pin (say bit0) and it is defined as an input at this time, the input signal present on the pin itself would be read into the CPU and re-written to the data latch of this particular pin, over-writing the previous content. As long as the pin stays in the input mode, no problem occurs. However, if bit0 is switched into output mode later on, the content of the data latch may now be unknown.

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 ("wire-or", "wire-and"). The resulting high output currents may damage the chip.

Example 5-4 shows the effect of two sequential read modify write instructions (ex. BCF, BSF, etc.) on an I/O Port.

EXAMPLE 5-4: READ MODIFY WRITE INSTRUCTIONS ON AN I/O PORT

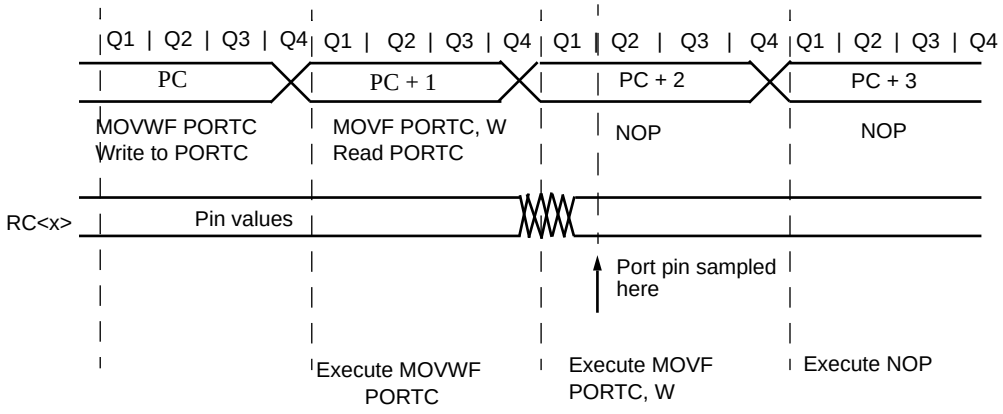
```
;Initial PORT settings:  PORTC<7:4> Inputs
;
;                        PORTC<3:0> Outputs
;PORTC<7:6> have external pull-up and are not
;connected to other circuitry
;
;                        PORT latch  PORT pins
;                        ----------  ----------

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


5.4.2 SUCCESSIVE OPERATIONS ON I/O PORTS

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

FIGURE 5-12: SUCCESSIVE I/O OPERATION

Example showing write to PORTC followed by immediate read. Some delays in settling may cause “old” Port data to be read, especially at higher clock frequencies. Data setup time = (0.25 Tcyc- Tpd), where Tcyc = instruction cycle time.



NOTES:

6.0 TIMER MODULES

The PIC14000 contains two general purpose timer modules, Timer0 (TMR0) and the Watchdog Timer (WDT). The A/D capture timer/counter is described in the A/D section.

The Timer0 module is identical to the Timer0 module of the PIC16C7X enhanced core products. It is an 8-bit overflow counter.

The Timer0 module has a programmable prescaler option. This prescaler can be assigned to either the Timer0 module or the Watchdog Timer (WDT). PSA (OPTION <3>) assigns the prescaler, and PS2:PS0 (OPTION <2:0>) determines the prescaler value. Timer0 can increment at the following rates: 1:1 (when prescaler assigned to Watchdog Timer), 1:2, 1:4, 1:8, 1:16, 1:32, 1:64, 1:128, 1:256.

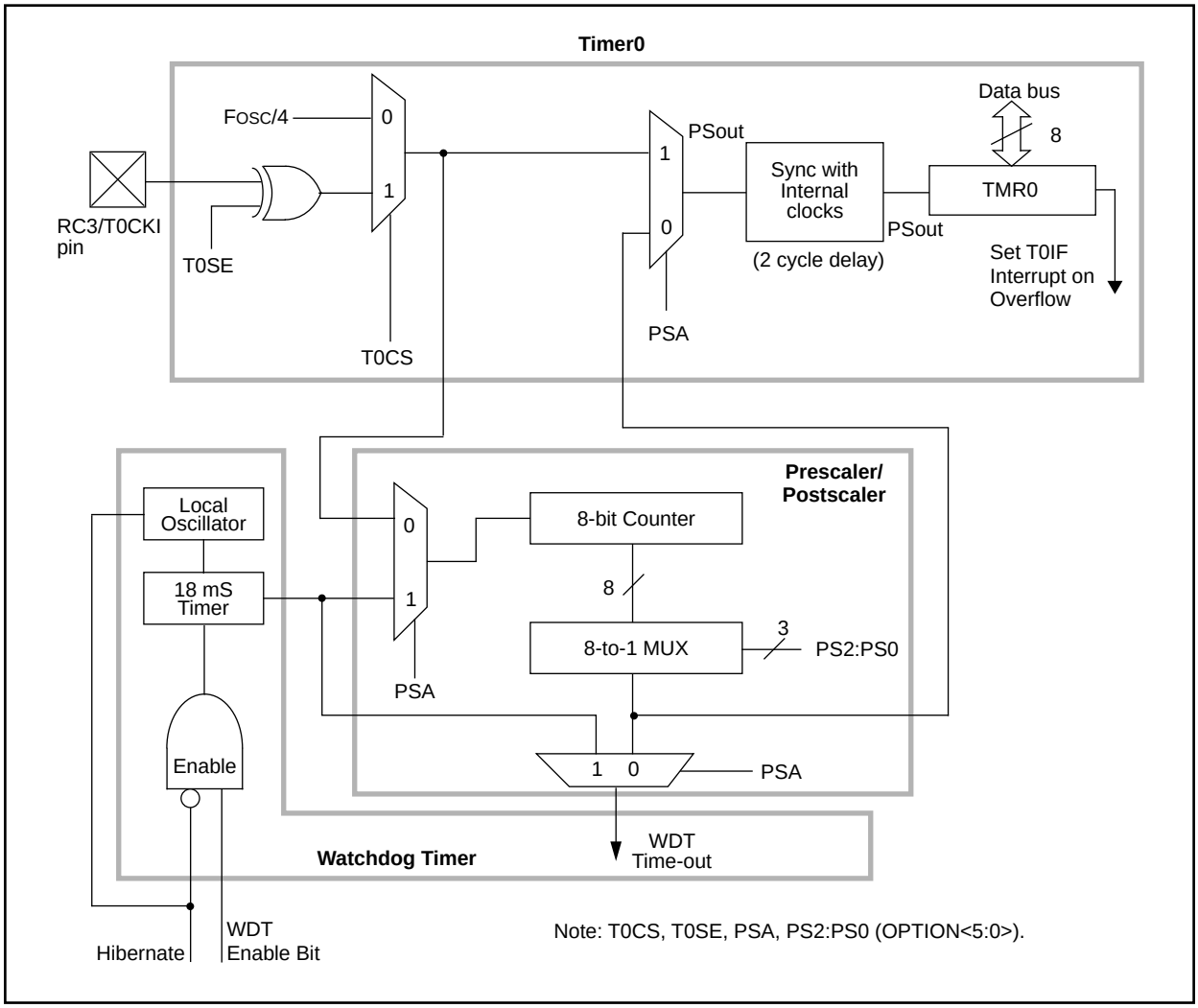
The Timer0 module has the following features:

- 8-bit timer
- Readable and writable (file address 01h)
- 8-bit software programmable prescaler
- Interrupt on overflow from FFh to 00h

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

The Timer0 module will increment every instruction cycle (without prescaler). If TMR0 is written, increment is inhibited for the following two cycles (Figure 6-2 and Figure 6-3. The user can compensate by writing an adjusted value to TMR0.

FIGURE 6-1: TIMER0 AND WATCHDOG TIMER BLOCK DIAGRAM



6.0.1 TIMER0 (TMR0) INTERRUPT

TMR0 interrupt is generated when the Timer0 module timer overflows from FFh to 00h. This overflow sets the T0IF bit. The interrupt can be masked by clearing bit T0IE (INTCON <5>). Flag bit T0IF (INTCON <2>) must be cleared in software by the TMR0 module interrupt service routine before re-enabling this interrupt. The Timer0 module interrupt cannot wake the processor from SLEEP since the timer is shut off during SLEEP. The timing of the TIMER0 interrupt is shown in Figure 6-4.

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

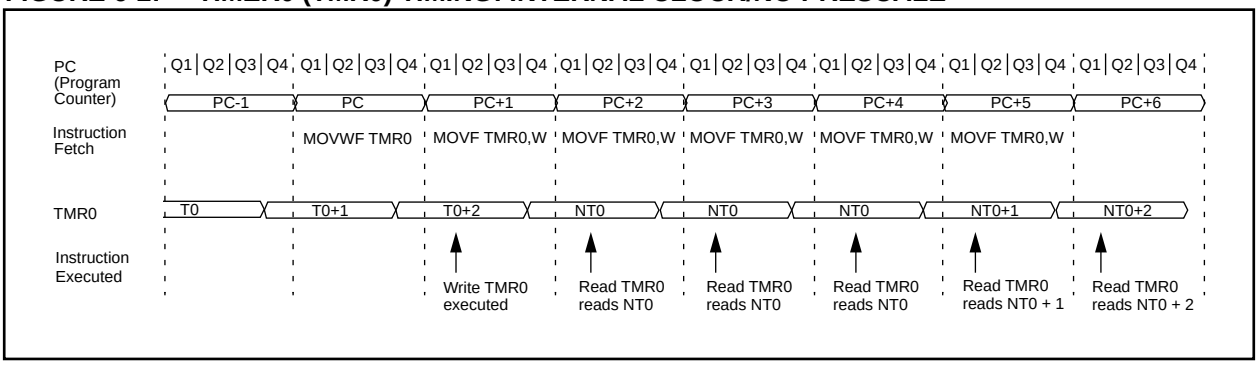


FIGURE 6-3: TIMER0 (TMR0) TIMING: INTERNAL CLOCK/PRESCALE 1:2

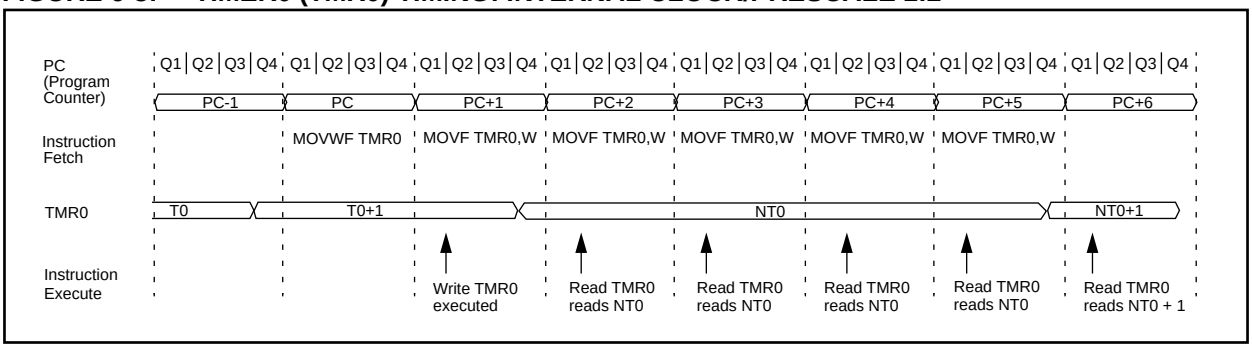
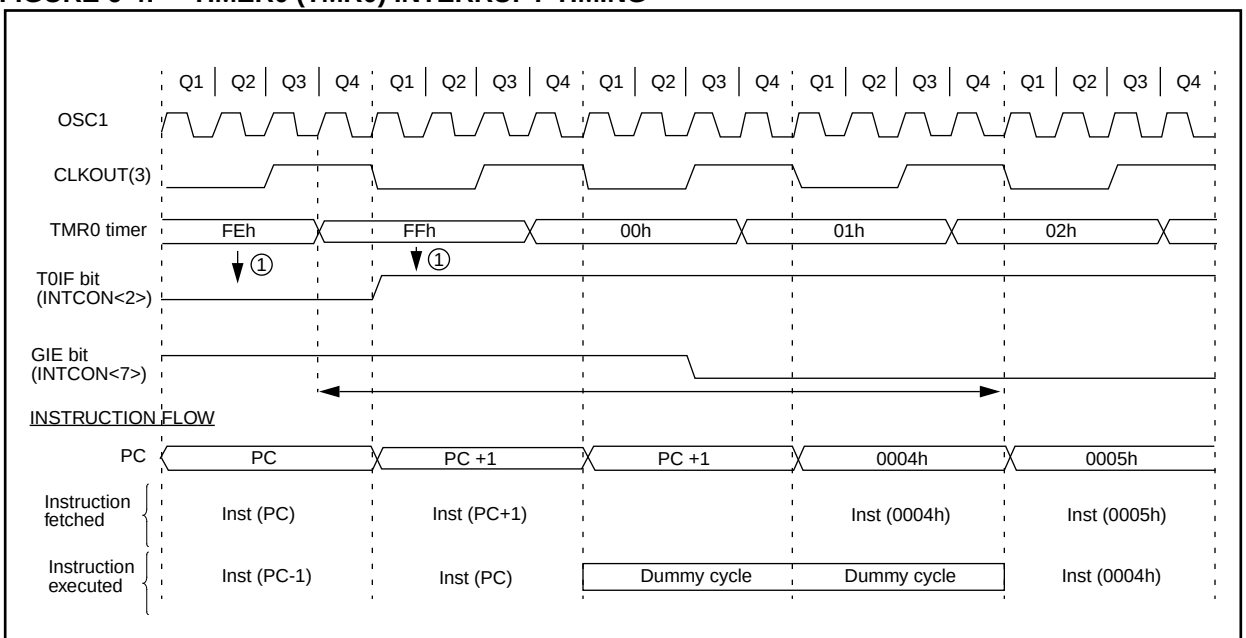


FIGURE 6-4: TIMER0 (TMR0) INTERRUPT TIMING



Note 1: T0IF interrupt flag is sampled here (every Q1).
2: Interrupt latency = 4Tcy where Tcy = instruction cycle time.
3: CLKOUT is available only in HS oscillator mode.

6.1 Using Timer0 with External Clock

When an external clock input is used for Timer0, it must meet certain requirements. The external clock requirement is due to internal phase clock (TOSC) synchronization. Also, there is a delay in the actual incrementing of TMR0 after synchronization.

6.1.1 EXTERNAL CLOCK SYNCHRONIZATION

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

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.

6.1.2 TIMER0 INCREMENT DELAY

Since the prescaler output is synchronized with the internal clocks, there is a small delay from the time the external clock edge occurs to the time the Timer0 module is actually incremented. Figure 6-5 shows the delay from the external clock edge to the timer incrementing.

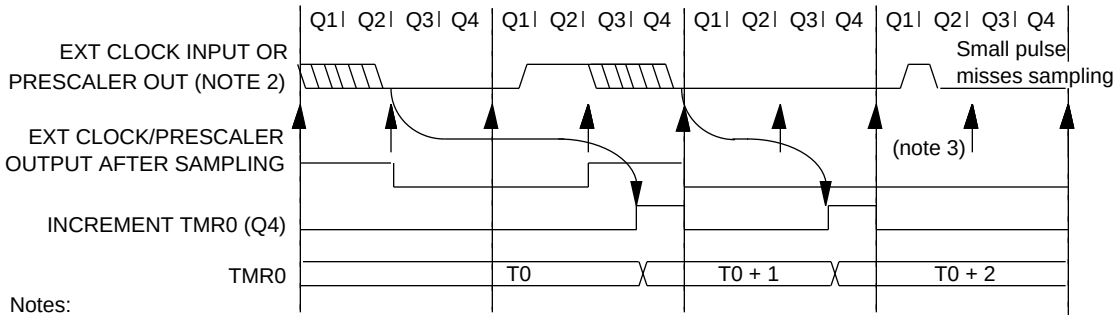
6.2 Prescaler

An 8-bit counter is available as a prescaler for the Timer0 module, or as a post-scaler for the Watchdog Timer (Figure 6-1). For simplicity, this counter is being referred to as “prescaler” throughout this data sheet. Note that there is only one prescaler available which is mutually exclusive between the Timer0 module and the Watchdog Timer. Thus, a prescaler assignment for the Timer0 module means that there is no prescaler for the Watchdog Timer, and vice-versa.

Bit PSA and PS2:PS0 (OPTION<3:0>) determine the prescaler assignment and prescale ratio.

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

FIGURE 6-5: TIMER0 TIMING WITH EXTERNAL CLOCK



Notes:

1. Delay from clock input change to TMR0 increment is 3 TOSC to 7 TOSC. (Duration of Q = TOSC). Therefore, the error in measuring the interval between two edges on TMR0 input = ± 4 tosc max.
2. External clock if no prescaler selected, Prescaler output otherwise.
3. The arrows indicate the points in time where sampling occurs.

6.2.1 SWITCHING PRESCALER ASSIGNMENT

The prescaler assignment is fully under software control, i.e., it can be changed “on the fly” during program execution. To avoid an unintended device RESET, the following instruction sequence (Example 6-1) must be executed when changing the prescaler assignment from TMR0 to WDT.

EXAMPLE 6-1: CHANGING PRESCALER (TMR0→WDT)

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

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

EXAMPLE 6-2: CHANGING PRESCALER (WDT→TMR0)

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

TABLE 6-1: SUMMARY OF TMR0 REGISTERS

Register Name	Function	Address	Power-on Reset Value
TMR0	Timer/counter register	01h	xxxx xxxx
OPTION	Configuration and prescaler assignment bits for TMR0. PIC.	81h	1111 1111
INTCON	TMR0 overflow interrupt flag and mask bits.	0Bh	0000 000x

Legend: x = unknown,

Note 1: For reset values of registers in other reset situations refer to Table 10-4.

TABLE 6-2: REGISTERS ASSOCIATED WITH TIMER0

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
01h	TMR0	TIMER0 TIMER/COUNTER							
0Bh/8Bh	INTCON	GIE	PEIE	T0IE	—	—	T0IF	—	—
81h	OPTION	$\overline{\text{RCPU}}$	—	T0CS	T0SE	PSA	PS2	PS1	PS0
87h	TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0

Legend: — = Unimplemented or reserved locations
Shaded boxes are not used by Timer0 module

7.0 INTER-INTEGRATED CIRCUIT SERIAL PORT (I^2C™)

The I^2C module is a serial interface useful for communicating with other peripheral or microcontroller devices. These peripheral devices may be serial EEPROMs, shift registers, display drivers, A/D converters, etc. The I^2C module is compatible with the following interface specifications:

- Inter-Integrated Circuit (I^2C)
- System Management Bus (SMBus)
- Access.bus™

Note: The I^2C module on PIC14000 only supports I^2C mode. This is different from the standard module used on the PIC16C7X family, which supports both I^2C and SPI modes. Caution should be exercised to avoid enabling SPI mode on the PIC14000

This section provides an overview of the Inter-IC(I^2C) bus. The I^2C bus is a two-wire serial interface developed by the Philips Corporation. The original specification, or standard mode, was for data transfers of up to 100 Kbps. An enhanced specification, or fast mode, supports data transmission up to 400 Kbps. Both standard mode and fast mode devices will inter-operate if attached to the same bus.

The I^2C interface employs a comprehensive protocol to ensure reliable transmission and reception of data. When transmitting data, one device is the “master” (generates the clock) while the other device(s) acts as the “slave”. All portions of the slave protocol are implemented in the I^2C module’s hardware, while portions of the master protocol will need to be addressed in the PIC14000 software. Table 7-1 defines some of the I^2C bus terminology. For additional information on the I^2C interface specification, please refer to the Philips Corporation document *“The I^2C-bus and how to use it”.* The order number for this document is 98-8080-575.

In the I^2C interface protocol each device has an address. When a master wishes to initiate a data transfer, it first transmits the address of the device that it wishes to talk to. All devices “listen” to see if this is their address. Within this address, a bit specifies if the master wishes to read from or write to the slave device. The master and slave are always in opposite modes (transmitter/receiver) of operation during a data transfer. They may operate in either of these two states:

- Master-transmitter and Slave-receiver
- Slave-transmitter and Master-receiver

In both cases the master generates the clock signal.

The output stages of the clock (SCL) and data (SDA) lines must have an open-drain or open-collector in order to perform the wired-AND function of the bus. External pull-up resistors are used to ensure a high level when no device is pulling the line down. The number of devices that may be attached to the I^2C bus is limited only by the maximum bus loading specification of 400 pF.

7.0.1 INITIATING AND TERMINATING DATA TRANSFER

During times of no data transfer (idle time), both the clock line (SCL) and the data line (SDA) are pulled high through the external pull-up resistors. The START and STOP determine the start and stop of data transmission. The START is defined as a high to low transition of SDA when SCL is high. The STOP is defined as a low to high transition of SDA when SCL is high. Figure 7-1 shows the START and STOP. The master generates these conditions for starting and terminating data transfer. Due to the definition of the START and STOP, when data is being transmitted the SDA line can only change state when the SCL line is low.

FIGURE 7-1: I^2C START AND STOP CONDITIONS

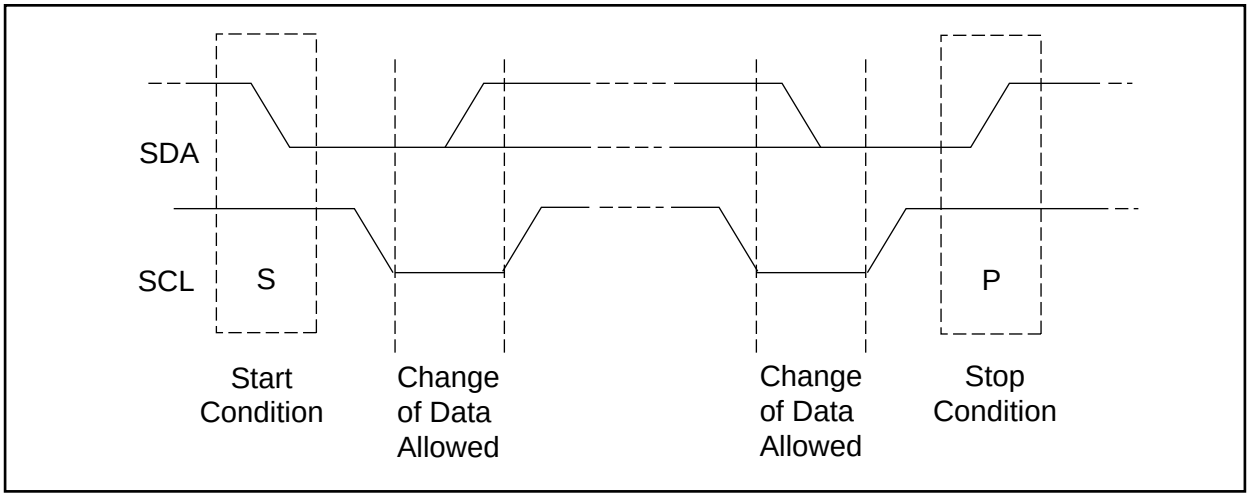


FIGURE 7-2: I^2CSTAT: I^2C PORT STATUS REGISTER

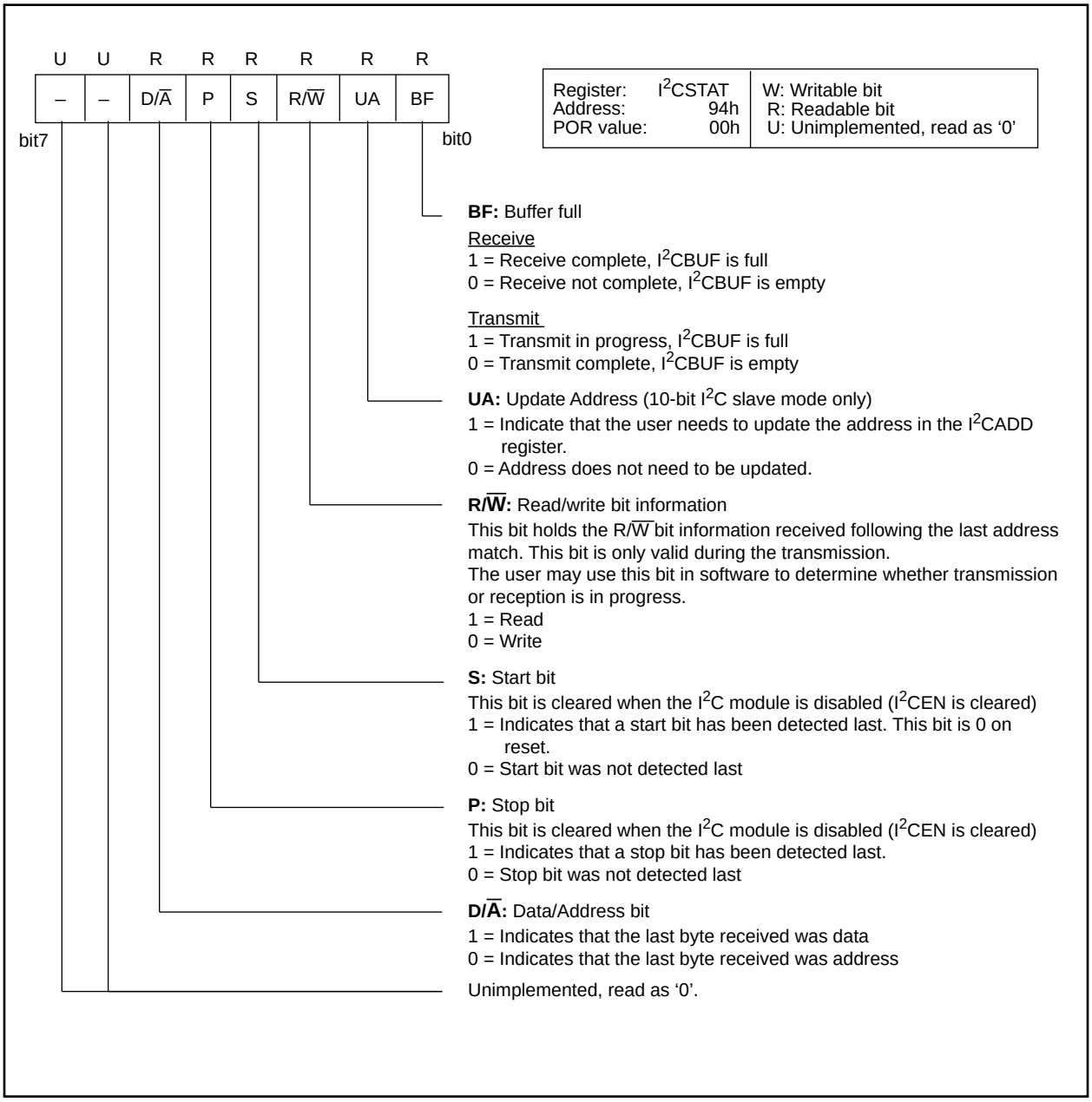


FIGURE 7-3: I²CCON: I²C PORT CONTROL REGISTER

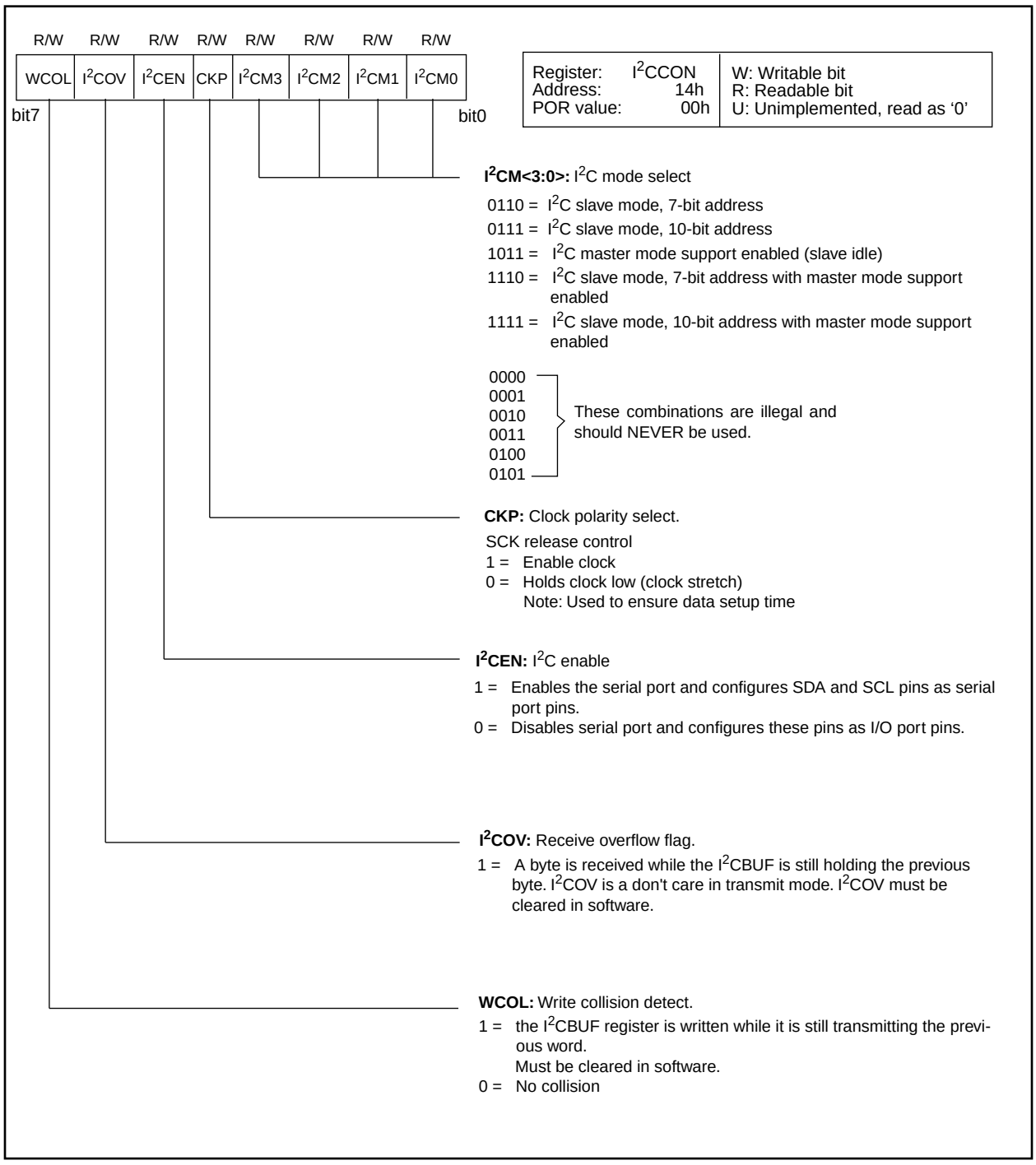


TABLE 7-1: I²C BUS TERMINOLOGY

Term	Description
Transmitter	The device that sends the data to the bus
Receiver	The device that receives the data from the bus
Master	The device which initiates the transfer, generates the clock, and terminates the transfer
Slave	The device addressed by a master
Multi-master	More than one master device in a system. These masters can attempt to control the bus at the same time without corrupting the message
Arbitration	Procedure that ensures that only one of the master devices will control the bus. This ensures that the transfer data does not get corrupted.
Synchronization	Procedure where the clock signals of two or more devices are synchronized.

FIGURE 7-4: I²C 7-BIT ADDRESS FORMAT

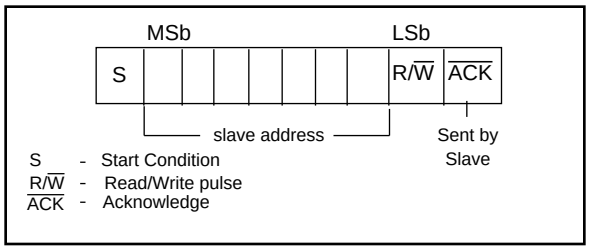
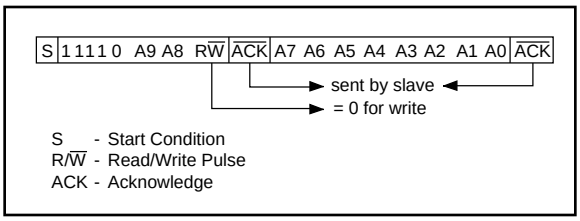


FIGURE 7-5: I²C 10-BIT ADDRESS FORMAT



7.0.2 ADDRESSING I²C DEVICES

There are two address formats. The simplest is the 7-bit address format with a R/$\overline{W}$ bit (Figure 7-4). The address is the most significant seven bits of the byte. For example when loading the I²CADD register, the least significant bit is a “don’t care”. The more complex is the 10-bit address with a R/$\overline{W}$ bit (Figure 7-5). For 10-bit address format, two bytes must be transmitted with the first five bits specifying this to be a 10-bit address.

7.0.3 TRANSFER ACKNOWLEDGE

All data must be transmitted per byte, with no limit to the number of bytes transmitted per data transfer. After each byte, the slave-receiver generates an acknowledge bit ($\overline{ACK}$). This is shown in Figure 7-6. When a slave-receiver doesn't acknowledge the slave address or received data, the master must abort the transfer. The slave must leave SDA high so that the master can generate the STOP (Figure 7-1).

If the master is receiving the data (master-receiver), it generates an acknowledge signal for each received byte of data, except for the last byte. To signal the end of data to the slave-transmitter, the master does not generate an acknowledge. The slave then releases the SDA line so the master can generate the STOP. The master can also generate the STOP during the acknowledge pulse for valid termination of data transfer.

If the slave needs to delay the transmission of the next byte, holding the SCL line low will force the master into a wait state. Data transfer continues when the slave releases the SCL line. This allows the slave to move the received data or fetch the data it needs to transfer before allowing the clock to start. This wait state technique can also be implemented at the bit level. Figure 7-7 shows a data transfer waveform.

Figure 7-8 and Figure 7-9 show master-transmitter and master-receiver data transfer sequences.

FIGURE 7-6: I^2C SLAVE-RECEIVER ACKNOWLEDGE

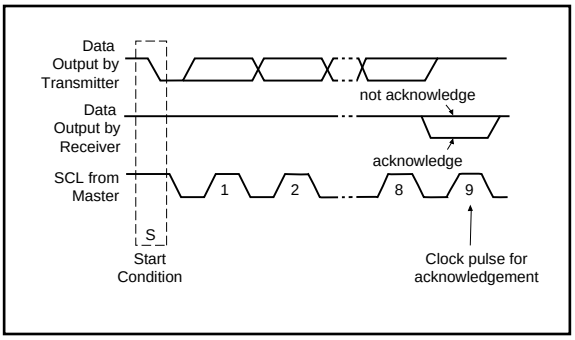
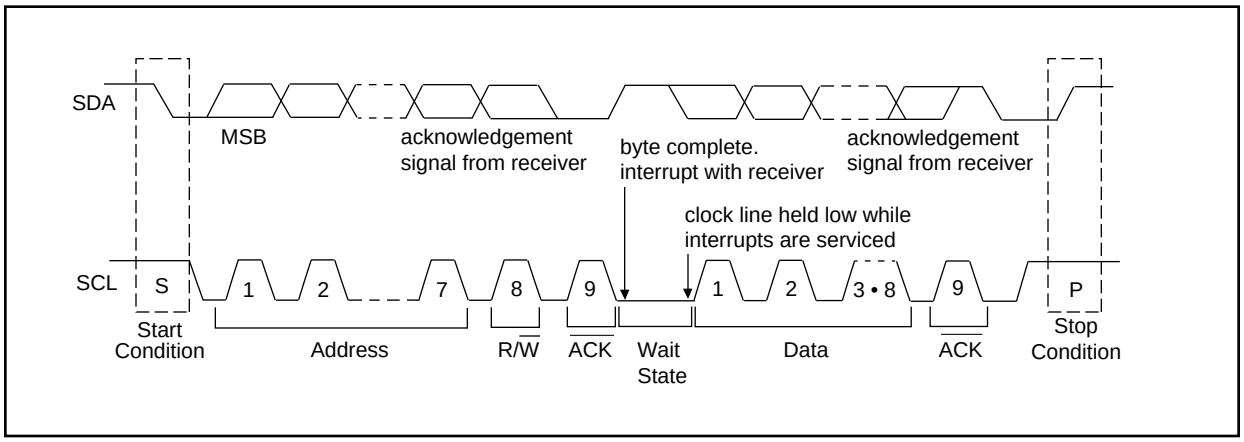


FIGURE 7-7: SAMPLE I^2C DATA TRANSFER



When a master does not wish to relinquish the bus (by generating a STOP condition), a repeated START (Sr) must be generated. This condition is identical to the START (SDA goes high-to-low while SCL is high), but occurs after a data transfer acknowledge pulse (not the bus-free state). This allows a master to send "commands" to the slave and then receive the requested information or to address a different slave device. This sequence is shown in Figure 7-10.

FIGURE 7-8: MASTER - TRANSMITTER SEQUENCE

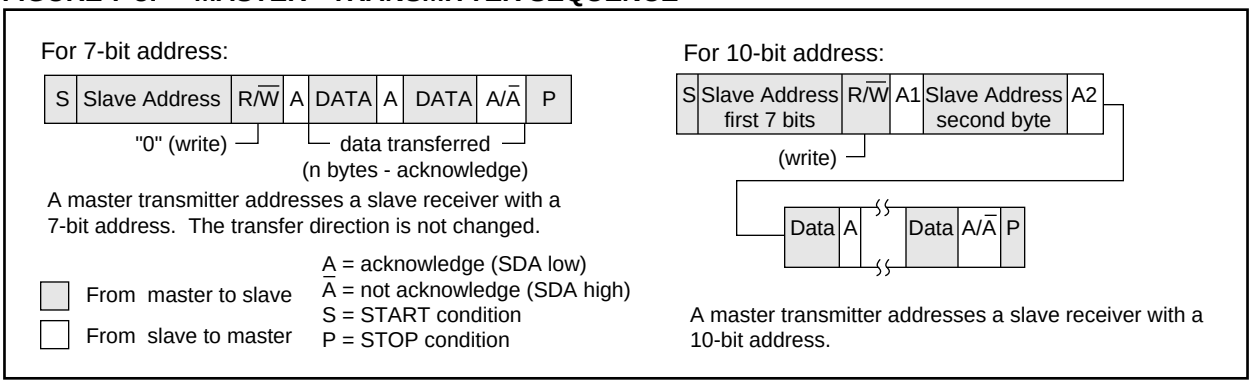


FIGURE 7-9: MASTER - RECEIVER SEQUENCE

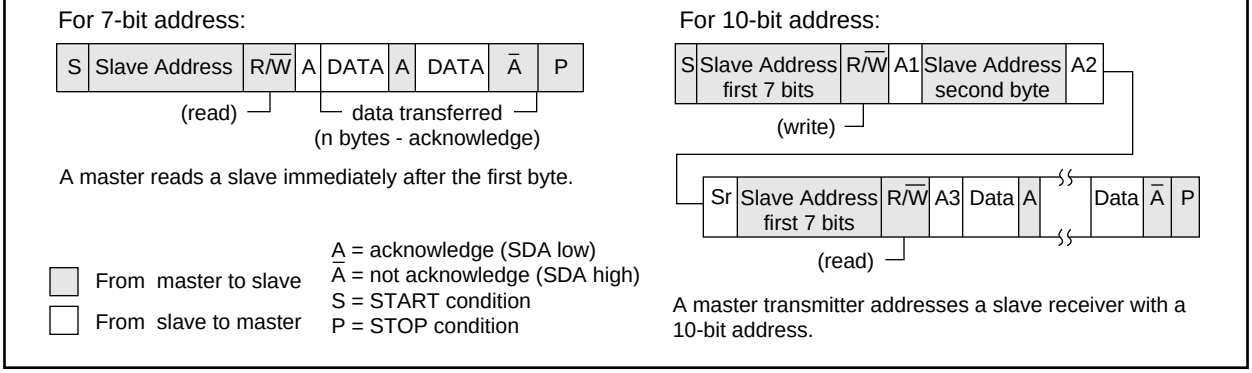
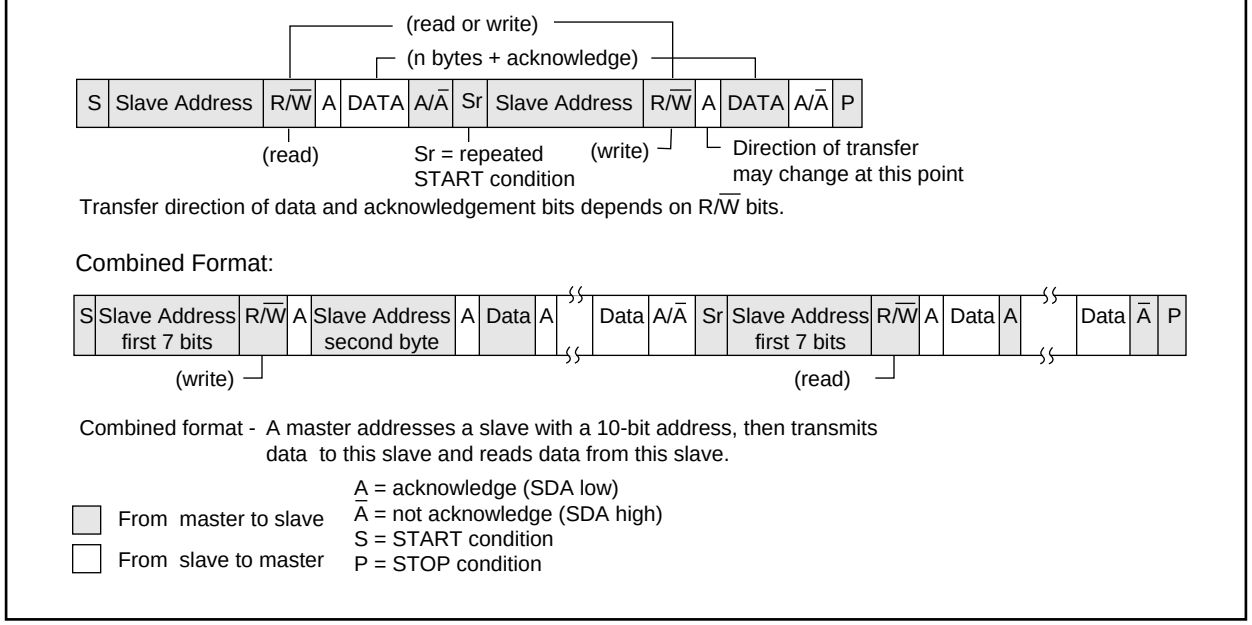


FIGURE 7-10: COMBINED FORMAT



7.0.4 MULTI-MASTER OPERATION

The I^2C protocol allows a system to have more than one master. This is called multi-master. When two or more masters try to transfer data at the same time, arbitration and synchronization occur.

7.0.4.1 ARBITRATION

Arbitration takes place on the SDA line, while the SCL line is high. The master which transmits a high when the other master transmits a low loses arbitration (Figure 7-11) and turns off its data output stage. A master which lost arbitrating can generate clock pulses until the end of the data byte where it lost arbitration. When the master devices are addressing the same device, arbitration continues into the data.

Masters that also incorporate the slave function, and have lost arbitration must immediately switch over to slave-receiver mode. This is because the winning master-transmitter may be addressing it.

Arbitration is not allowed between:

- A repeated START
- A STOP and a data bit
- A repeated START and a STOP

Care needs to be taken to ensure that these conditions do not occur.

7.0.4.2 CLOCK SYNCHRONIZATION

Clock synchronization occurs after the devices have started arbitration. This is performed using a wired-AND connection to the SCL line. A high to low transition on the SCL line causes the concerned devices to start counting off their low period. Once a device clock has gone low, it will hold the SCL line low until its SCL high state is reached. The low to high transition of this clock may not change the state of the SCL line, if another device clock is still within its low period. The SCL line is held low by the device with the longest low period. Devices with shorter low periods enter a high wait-state, until the SCL line comes high. When the SCL line comes high, all devices start counting off their high periods. The first device to complete its high period will pull the SCL line low. The SCA line high time is determined by the device with the shortest high period. This is shown in the Figure 7-12.

FIGURE 7-11: MULTI-MASTER ARBITRATION (2 MASTERS)

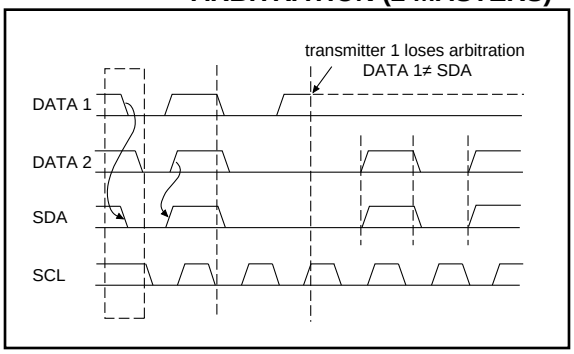


FIGURE 7-12: I^2C CLOCK SYNCHRONIZATION

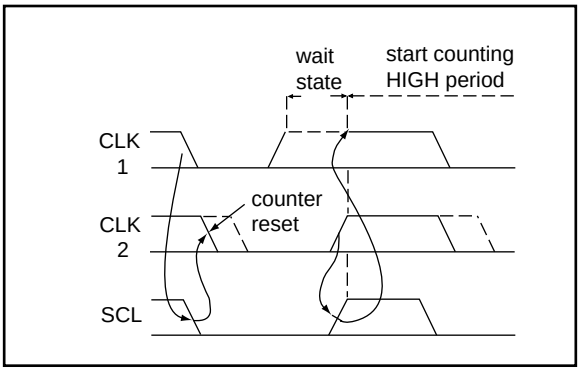
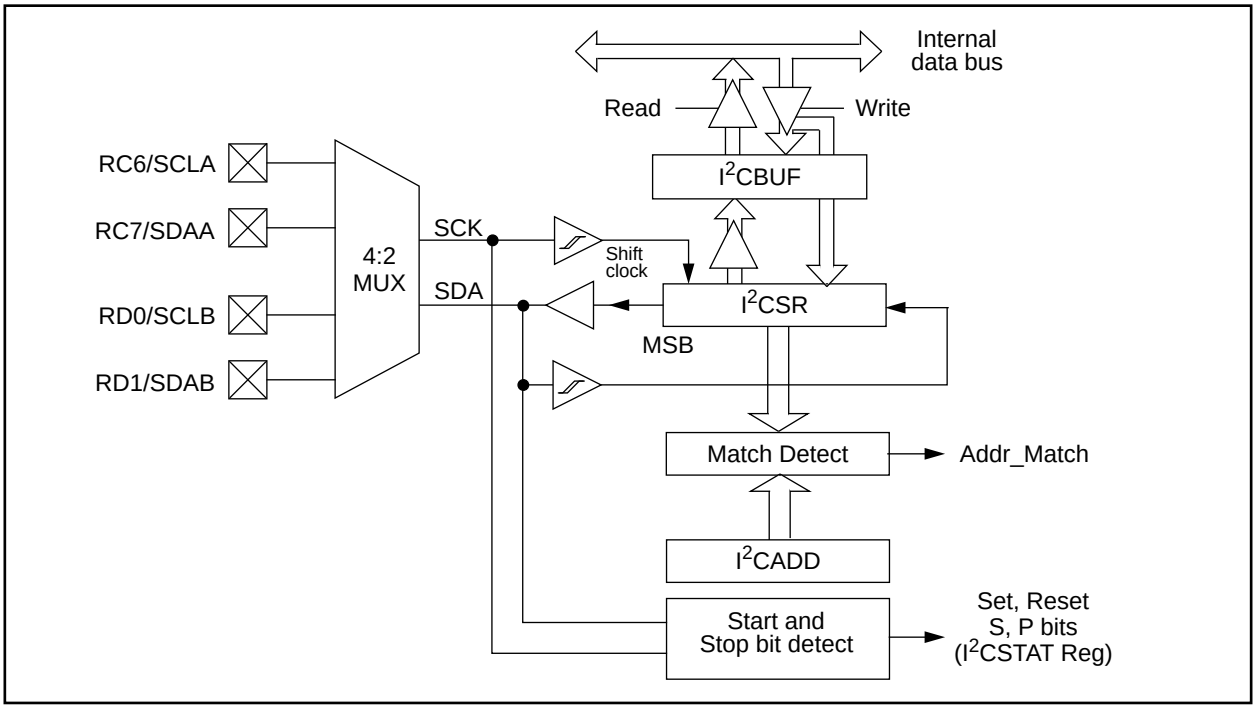


FIGURE 7-13: I^2C BLOCK DIAGRAM



7.1 I^2C Operation

The I^2C module in I^2C mode fully implements all slave functions, and provides support in hardware to facilitate software implementations of the master functions. The I^2C module implements the standard and fast mode specifications as well as 7-bit and 10-bit addressing. Two pins are used for data transfer. These are the RC6/SCLA pin, which is the I^2C clock, and the RC7/SDAA pin which acts as the I^2C data. The users must configure these pins as inputs or outputs through the TRISC<7:6> bits. A block diagram of the I^2C module in I^2C mode is shown in Figure 7-13. The I^2C module functions are enabled by setting the I^2C Enable (I^2CEN) bit in the I^2CCON register (14h, bit 5).

The I^2C module has five registers for I^2C operation. These are the:

- I^2C Control Register (I^2CCON)
- I^2C Status Register (I^2CSTAT)
- Serial Receive / Transmit Buffer (I^2CBUF)
- I^2C Shift Register (I^2CSR) - Not directly accessible
- Address Register (I^2CADD)

The I^2CCON register (14h) allows control of the I^2C operation. Four mode selection bits (I^2CCON<3:0>) allow one of the following I^2C modes to be selected:

- I^2C Slave mode (7-bit address)
- I^2C Slave mode (10-bit address)
- I^2C Slave mode (7-bit address), with master-mode support
- I^2C Slave mode (10-bit address), with master-mode support
- I^2C Master mode, slave is idle

Selection of any I^2C mode with the I^2CEN bit set, forces the SCL and SDA pins to be open collector, provided these pins are set to inputs through the TRISC bits.

The I^2CSTAT register gives the status of the data transfer. This information includes detection of a START or STOP bit, specifies if the received byte was data or address, if the next byte is the completion of 10-bit address, and if this will be a read or write data transfer. The I^2CSTAT register is read only.

The I^2CBUF is the register to which transfer data is written to or read from. The I^2CSR register shifts the data in or out of the device. In receive operations, the I^2CBUF and I^2CSR create a double buffered receiver. This allows reception of the next byte before reading the last byte of received data. When the complete byte is received, it is transferred to the I^2CBUF and the I^2CIF is set. If another complete byte is received before the I^2CBUF is read, a receiver overflow has occurred and the I^2COV bit (I^2CCON<6>) is set.

The I^2CADD register holds the slave address. In 10-bit mode, the user needs to write the high byte of the address (1 1 1 1 0 A9 A8 0). Following the high byte address match, the low byte of the address needs to be loaded (A7-A0).

7.1.1 SLAVE MODE

In slave mode, the SCL and SDA pins must be configured as inputs (TRISC <7:6> are set). The I^2C module will override the input state with the output data when required (slave-transmitter).

When an address is matched or the data transfer from an address match is received, the hardware automatically will generate the acknowledge ($\overline{ACK}$) pulse, and then load the I^2CBUF with the received value in the I^2CSR.

There are two conditions that will cause the I^2C module not to give this $\overline{ACK}$ pulse. These are if either (or both) occur:

- the Buffer Full (BF) bit was set before the transfer was received, or
- the Overflow (I^2COV) bit was set before the transfer was received.

In this case, the I^2CSR value is not loaded into the I^2CBUF, but the I^2CIF bit is set. Table 7-2 shows what happens when a data transfer byte is received, given the status of the BF and I^2COV bits. The shaded boxes show the conditions where user software did not properly clear the overflow condition. The BF flag is cleared by reading the I^2CBUF register while the I^2COV bit is cleared through software.

The SCL clock input must have a minimum high and low for proper operation. The high and low times of the I^2C specification as well as the requirement of the I^2C module is shown in the AC timing specifications.

TABLE 7-2: DATA TRANSFER RECEIVED BYTE ACTIONS

Status Bits as Data Transfer is Received		$I^2CSR \rightarrow I^2CBUF$	Generate $\overline{ACK}$ Pulse	Set I^2CIF bit (I^2C interrupt if enabled)
BF	I^2COV			
0	0	Yes	Yes	Yes
1	0	No	No	Yes
1	1	No	No	Yes
0	1	No	No	Yes

7.1.1.1 ADDRESSING

Once the I²C module has been enabled, the I²C waits for a START to occur. Following the START, the 8-bits are shifted into the I²CSR. All incoming bits are sampled with the rising edge of the clock (SCL) line. The I²CSR<7:1> is compared to the I²CADD register. The address is compared on the falling edge of the eighth clock (SCL) pulse. If the addresses match, and the BF and I²COV bits are clear, the following things happen:

- I²CSR loaded into I²CBUF
- Buffer Full (BF) bit is set
- $\overline{ACK}$ pulse is generated
- I²C Interrupt Flag (I²CIF) is set (interrupt is generated if enabled (I²CIE set) on falling edge of ninth SCL pulse.

In 10-bit address mode, two address bytes need to be received by the slave (Figure 7-5). The five most significant bits (MSbs) of the first address byte specify if this is a 10-bit address. The R/$\overline{W}$ bit (bit 0) must specify a write, so the slave device will received the second address byte. For a 10-bit address the first byte would equal '1 1 1 1 0 A9 A8 0', where A9 and A8 are the two MSbs of the address. The sequence of events for 10-bit address are as follows, with steps 7-9 for slave-transmitter:

1. Receive first (high) byte of address (I²CIF, BF and UA are set).
2. Update I²CADD with second (low) byte of address (clears UA and releases SCL line).
3. Read I²CBUF (clears BF) and clear I²CIF.
4. Receive second (low) byte of address (I²CIF, BF and UA are set).
5. Update I²CADD with first (high) byte of address (clears UA, if match releases SCL line).
6. Read I²CBUF (clears BF) and clear I²CIF
7. Receive Repeated START.
8. Receive first (high) byte of address (I²CIF and BF are set).
9. Read I²CBUF (clears BF) and clear I²CIF.

7.1.1.2 RECEPTION

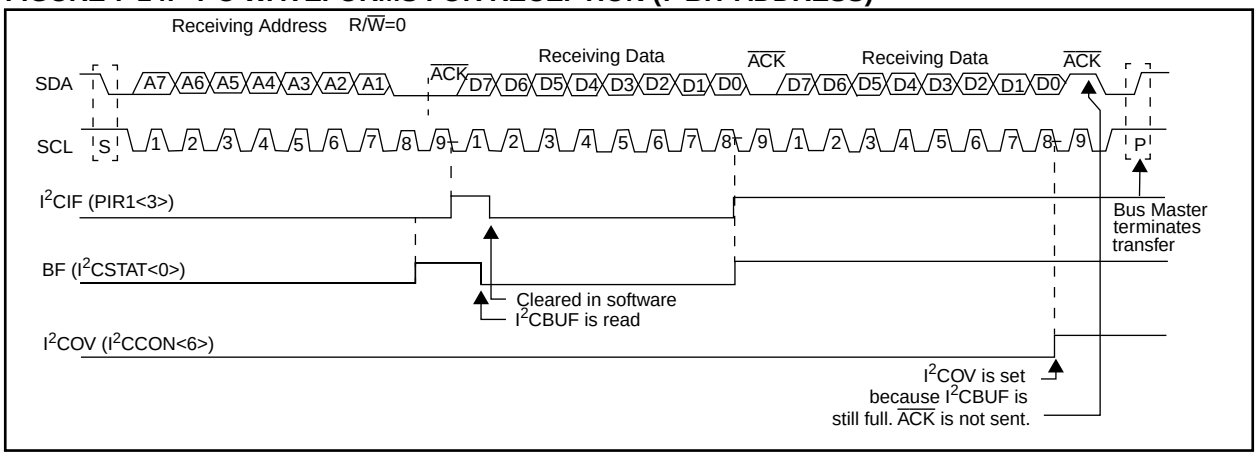
When the R/$\overline{W}$ bit of the address byte is clear and an address match occurs, the R/$\overline{W}$ bit of the I²CSTAT register is cleared. The received address is loaded into the I²CBUF.

When the address byte overflow condition exists then no acknowledge ($\overline{ACK}$) pulse is given. An overflow condition is defined as either the BF bit (I²CSTAT<0>) is set or the I²COV bit (I²CCON<6>) is set (Figure 7-14).

An I²CIF interrupt is generated for each data transfer byte. The I²CIF bit must be cleared in software, and the I²CSTAT register is used to determine the status of the byte. In master mode with slave enabled, three interrupt sources are possible. Reading BF, P and S will indicate the source of the interrupt.

Caution: BF is set after receipt of eight bits and automatically cleared after the I²CBUF is read. However, the flag is not actually cleared until receipt of the acknowledge pulse. Otherwise extra reads appear to be valid.

FIGURE 7-14: I²C WAVEFORMS FOR RECEPTION (7-BIT ADDRESS)



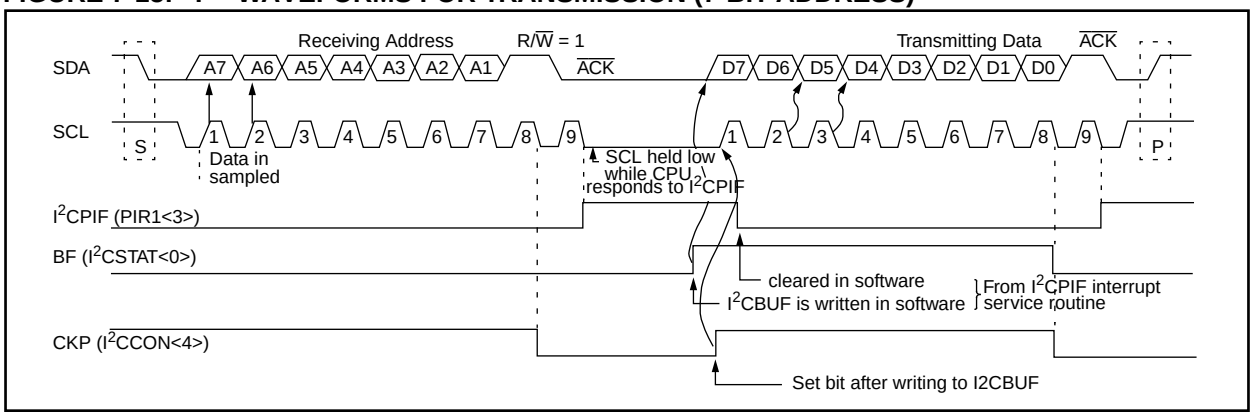
7.1.1.3 TRANSMISSION

When the R/$\overline{W}$ bit of the address byte is set and an address match occurs, the R/$\overline{W}$ bit of the I^2CSTAT register is set. The received address is loaded into the I^2CBUF The $\overline{ACK}$ pulse will be sent on the ninth bit, and the SCL pin is held low. The transmit data must be loaded into the I^2CBUF register, which also loads the I^2CSR register. Then the SCL pin should be enabled by setting the CKP bit (I^2CCON<4>). The eight data bits are shifted out on the falling edge of the SCL input. This ensures that the SDA signal is valid during the SCL high time (Figure 7-15).

A I^2CIF interrupt is generated for each data transfer byte. The I^2CIF bit must be cleared in software, and the I^2CSTAT register is used to determine the status of the byte. The I^2CIF bit is set on the falling edge of the ninth clock pulse.

As a slave-transmitter, the $\overline{ACK}$ pulse from the master-receiver is latched on the rising edge of the ninth SCL input pulse. If the SDA line was high (not $\overline{ACK}$), then the data transfer is complete. The slave then monitors for another occurrence of the START bit. If the SDA line was low ($\overline{ACK}$), the transmit data must be loaded into the I^2CBUF register, which also loads the I^2CSR register. Then the SCL pin should be enabled by setting the CKP bit (I^2CCON<4>).

FIGURE 7-15: I^{2C} WAVEFORMS FOR TRANSMISSION (7-BIT ADDRESS)



7.1.2 MASTER MODE

Master mode operation is supported by interrupt generation on the detection of the START and STOP. The STOP(P) and START(S) bits are cleared from a reset or when the I^2C module is disabled. Control of the I^2C bus may be taken when the P bit is set, or the bus is idle and both the S and P bits are cleared.

In master mode, the SCL and SDA lines are manipulated by changing the corresponding TRISC<7:6> bits to an output (cleared). The output level is always low, regardless of the value(s) in PORTC<7:6>. So when transmitting data, a “1” data bit must have the TRISC<7> bit set (input) and a “0” data bit must have the TRISC<7> bit cleared (output). The same scenario is true for the SCL line with the TRISC<6> bit.

The following events will cause the I^2C interrupt Flag (I^2CIF) to be set (I^2C interrupt if enabled):

- START
- STOP
- Data transfer byte transmitted/received

Master mode of operation can be done with either the slave mode idle ($I^2CM3...I^2CM0$ = 1011b) or with the slave active. When both master and slave modes are enabled, the software needs to differentiate the source(s) of the interrupt.

7.1.3 MULTI-MASTER MODE

In multi-master mode, the interrupt generation on the detection of the START and STOP allows the determination of when the bus is free. The STOP (P) and START (S) bits are cleared from a reset or when the I^2C module is disabled. Control of the I^2C bus may be taken when the P bit is set, or the bus is idle and both the S and P bits are cleared. When the bus is busy, enabling the I^2C interrupt will generate the interrupt when the STOP occurs.

In multi-master operation, the SDA line must be monitored to see if the signal level is the expected output level. This check only needs to be done when a high level is output. If a high level is expected and low level is present, the device needs to release the SDA and SCL lines (set TRISC<7:6>). There are two stages where this arbitration can be lost, these are:

- Address Transfer
- Data Transfer

When the slave logic is enabled, the slave continues to receive. If arbitration was lost during the address transfer stage, the device may being addressed. If addressed an $\overline{ACK}$ pulse will be generated. If arbitration was lost during the data transfer stage, the device will need to re-transfer the data at a later time.

TABLE 7-3: REGISTERS ASSOCIATED WITH I^2C OPERATION

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0B/8Bh	INTCON	GIE	PEIE	T0IE	r	r	T0IF	r	r
0Ch	PIR1	WUIF	—	—	PBIF	I^2CIF	RCIF	ADIF	OVFIF
8Ch	PIE1	WUIE	—	—	PBIE	I^2CIE	RCIE	ADIE	OVFIE
13h	I^2CBUF	I^2C Serial Port Receive Buffer/Transmit Register							
93h	I^2CADD	I^2C mode Synchronous Serial Port (I^2C mode) Address Register							
14h	I^2CCON	WCOL	I^2CON	I^2CEN	CKP	I^2CM3	I^2CM2	I^2CM1	I^2CM0
94h	I^2CSTAT	—	—	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF

Legend: — = Unimplemented location, read as ’0’
r indicates reserved locations, default is POR value and **should not be overwritten with any value**

Note: Shaded boxes are not used by the I^2C module.

FIGURE 7-16: OPERATION OF THE I^2C IN IDLE_MODE, RCV_MODE OR XMIT_MODE

```
IDLE_MODE (7-bit):
if (Addr_match)                 {       Set interrupt;
                                        if (R/W = 1)    {       Send ACK = 0;
                                                                set XMIT_MODE;
                                                        }
                                        else if (R/W = 0) set RCV_MODE;
                                }
```

```
RCV_MODE:
if ((I2CBUF=Full) OR (I2COV = 1))
        {       Set I2COV;
                Do not acknowledge;
        }
else    {       transfer I2CSR → I2CBUF;
                send ACK = 0;
        }
Receive 8-bits in I2CSR;
Set interrupt;
```

```
XMIT_MODE:
While ((I2CBUF = Empty) AND (CKP=0)) Hold SCL Low;
Send byte;
Set interrupt;
if (ACK Received = 1)           {       End of transmission;
                                        Go back to IDLE_MODE;
                                }
else if (ACK Received = 0)   Go back to XMIT_MODE;
```

```
IDLE_MODE (10-Bit):
If (High_byte_addr_match AND (R/W = 0))
        {       PRIOR_ADDR_MATCH = FALSE;
                Set interrupt;
                if ((I2CBUF = Full) OR ((I2COV = 1))
                        {       Set I2COV;
                                Do not acknowledge;
                        }
                else    {       Set UA = 1;
                                Send ACK = 0;
                                While (I2CADD not updated) Hold SCL low;
                                Clear UA = 0;
                                Receive Low_addr_byte;
                                Set interrupt;
                                Set UA = 1;
                                If (Low_byte_addr_match)
                                        {       PRIOR_ADDR_MATCH = TRUE;
                                                Send ACK = 0;
                                                while (I2CADD not updated) Hold SCL low;
                                                Clear UA = 0;
                                                Set RCV_MODE;
                                        }
                        }
        }
else if (High_byte_addr_match AND (R/W = 1)
        {       if (PRIOR_ADDR_MATCH)
                        {       send ACK = 0;
                                set XMIT_MODE;
                        }
        else PRIOR_ADDR_MATCH = FALSE;
        }
```

7.1.4 SMBus™ AND ACCESS.bus™ CONSIDERATIONS

PIC14000 is compliant with the SMBus specification published by Intel® and ACCESS.bus specifications. Some key points to note regarding the differences between the two bus specifications and how it pertains to the PIC14000 hardware are listed below:

- SMBus has fixed input voltage thresholds. PIC14000 I/O buffers have programmable levels that can be selected to be compatible with both SMBus threshold levels via the SMBus and SPGND bits in the MISC register.
- PIC14000 IOL levels and VOL levels have been set to meet the worst-case of both the SMBus and ACCESS bus specifications. Specifically, VOL = 0.4V and IOL = 6 mA.
- A mechanism to stretch the I^2C clock time has been implemented to support SMBus slave transactions. The SMHOG bit in the MISC register allows hardware to automatically force and hold the I^2C clock line low when a data byte has been received. This prevents the SMBus master from overflowing the receive buffer in instances where the microcontroller may be to busy servicing higher priority tasks to respond to a I^2C module interrupt. Or, if the microcontroller is in SLEEP mode and needs time to wake-up and respond to the I^2C interrupt

7.1.4.1 SMHOG STATE MACHINE

```
While SMHOG = 1 do
     do
          if SMHOG = 0 then exit
(while)        ; if I C int. pending do noth-
ing until it's serviced
     until I2CIF = 0
     do
          if SMHOG = 0 then exit
(while)        ; wait for I2C interrupt
     until I2CIF =1
     do
          do
               if SMHOG = 0 then exit (while)
           until (SCL = 0 or I2CIF = 0)
           if I2CIF = 1 then
                do
                      hold SCL low
                until I2CIF = 0 or SMHOG=0
            if SMHOG = 0 then exit (while)
      forever
while end
```


8.0 ANALOG MODULES FOR A/D CONVERSION

8.1 Overview

PIC14000 includes analog components to create a precision slope A/D converter that is used to translate battery voltage, current and temperature into digital values for both battery monitoring and charging control. A slope conversion method is especially suited to applications in which a relatively lengthy time may be taken for conversion (several milliseconds) to obtain the benefits of noise reduction through signal averaging. This can lead to greater resolutions and accuracies than achievable with a successive-approximation converter for the same cost. PIC14000 uses a digital integration method to eliminate many of the inaccuracies and complexity associated with an analog integrator.

The slope A/D converter (Figure 8-1) includes:

- Comparator
- 4-bit current DAC
- 16-channel analog mux
- 16-bit A/D capture timer

The 16 analog channels can be assigned to:

- External voltage or external reference
- Temperature (internal)
- Temperature (external)
- External current
- Internal bandgap reference
- SREFHI
- SREFLO
- Charge/wake-up controller DAC outputs

For a battery management application, external voltage would be the battery voltage and the external current would be the charge/discharge current of the battery pack.

Each channel is converted independently by means of a slope conversion method using a single precision comparator. The current DAC feeds an external 0.1 µF (nominal) capacitor to generate the ramp voltage used in the conversion.

8.2 Conversion Process

These are the steps to perform data conversion:

- Set ADRST (ADCON0<2>), which stops the timer and discharges the ramp capacitor to ground.
- Be sure to set ADRST for a minimum of 200 µs.
- After conversion takes place, reset ADRST through software, it will allow the capture timer to begin counting and the ramp capacitor to begin charging.
- The capture timer is an up-counter, and must be reset by software to 0000h before each conversion.
- When the ramp voltage exceeds the analog input, the comparator output changes from high to low.
- This transition causes a capture event and copies the current A/D timer value into the 16-bit capture register.
- An interrupt is generated to the CPU if enabled.

Note: The A/D timer continues to run following a capture event.

The maximum A/D timer count is 65,536. Its frequency is fixed to the oscillator frequency, as determined by the on-chip crystal or IN oscillator. At a 4 MHz oscillation frequency, the maximum conversion time is 16.38 ms for a full count. A typical conversion should complete before full-count is reached. A timer overflow flag is set once the timer rolls over (FFFFh to 0000h), and an interrupt is sent to the CPU, if enabled.

End-user calibration is greatly simplified or eliminated by making use of the on-chip EPROM. Internal component values are measured at factory final test and stored in the memory for use by the application firmware.

Periodic conversion cycles should be performed on the bandgap reference to compensate for A/D component drift. Measurements for the reference voltage count are equated to the voltage value stored into EPROM during calibration. All other channel measurements are compensated for by ratioing the actual count with the bandgap count any multiplying by the bandgap voltage value stored in EPROM. In addition, a bias/zeroing network is provided on chip for the current sense input. The result is, the zero point of the current provides very high accuracies at low current values, where most needed. Since all measurements are relative to the reference, offset voltages inherent in the comparator are cancelled out. The combination of slope-conversion with automatic zeroing, plus factory calibration allow A/D resolutions of 16-bits with accuracies exceeding 12 bits.

Most of the analog components used in the conversion and the A/D timer clock are automatically disabled during idle periods for maximum power savings. Several other power-saving modes can be enabled via software and/or hardware control (Section 10.7).

8.3 A/D Capture Timer (ADTMR) Module

The A/D capture timer (ADTMR) is used as the reference counter for the A/D conversion(s) and is comprised of a 16-bit up timer, which is incremented every oscillator cycle. ADTMR is reset to 0000h by a power-up reset; otherwise the software must reset it after each conversion. A separate 16-bit capture register (ADCAP) is used to capture the ADTMR count if an A/D capture event occurs (see below). Both the A/D timer and capture register are readable and writable. The low byte of the A/D timer (ADTMRL) is accessed at location 0Eh while the high byte (ADTMRH) is accessed at location 0Fh. Similarly, the low byte of the A/D capture register (ADCAP) is accessed at location 15h, and the high byte is located at 16h.

Caution: Reading or writing the ADTMR register during an A/D conversion cycle can produce unpredictable results and is not recommended.

The correct sequence for writing the ADTMR register is as follows:

Action	Result
1. Write the HI byte first.	This stops the timer from counting
2. Write the LOW byte last.	After it is written, the timer will start counting automatically.

Reversing this order will yield unpredictable results.

During conversion one of two events will occur:

1. capture event, **or**
2. timer overflow

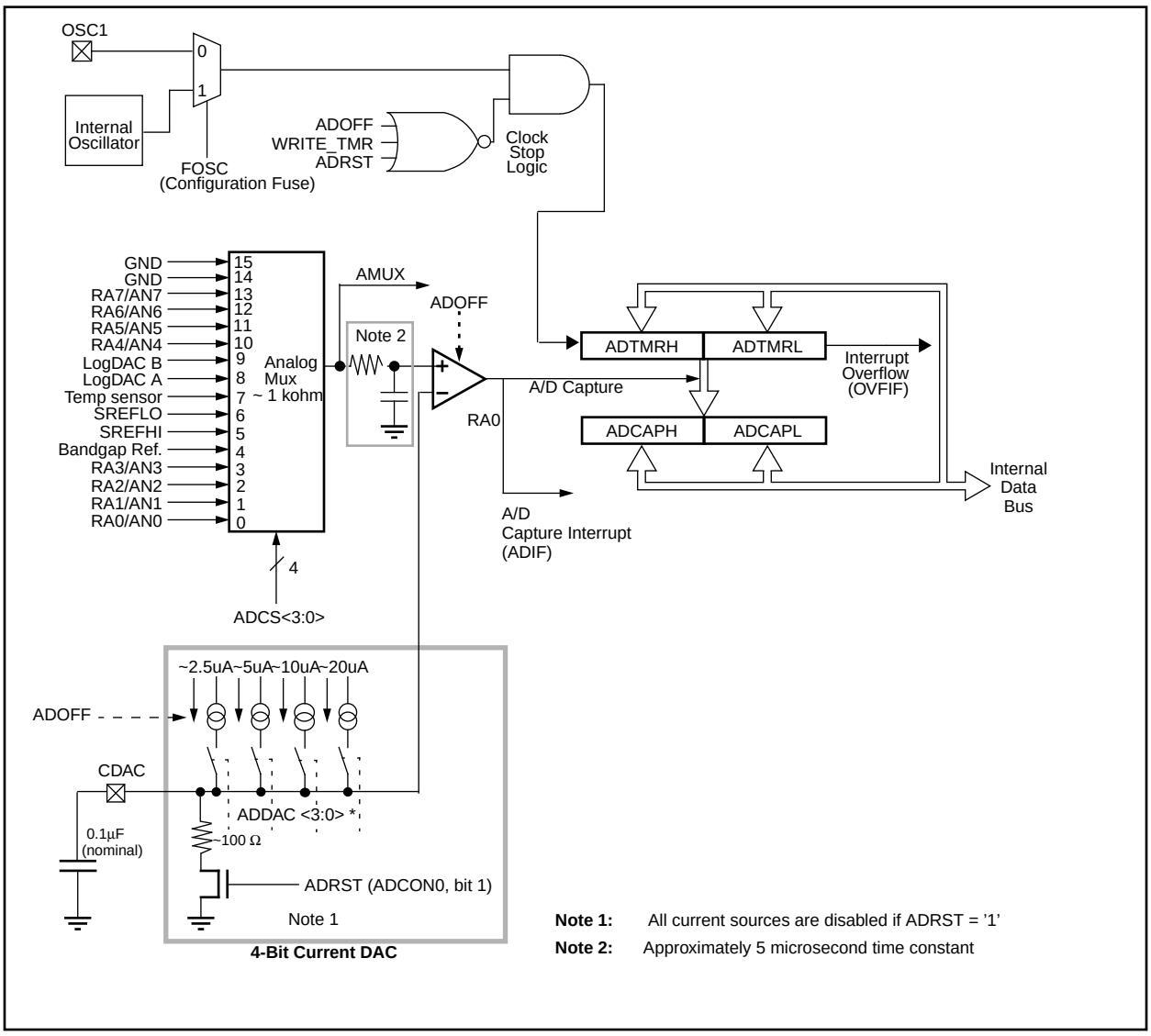
In a capture event, the comparator trips when the slope voltage on the CDAC output exceeds the input voltage, causing the comparator output to transition from high to low. This causes a transfer of the current timer count to the capture register and setting of the ADCIF flag (PIR1<1>). A CPU interrupt will be generated if bit ADCIE (PIE1<1>) is programmed to '1' (interrupt enabled). In addition, the Global Interrupt Enable (GIE, INTCON<7>) must also be set. Software is responsible for clearing the ADCIF flag prior to the next conversion cycle. Note that this interrupt can only occur once per conversion cycle.

In a capture timer overflow condition, no valid capture event occurs (comparator does not trip). In this case, the timer rolls over from FFFFh to 0000h, and a capture overflow flag (OVFIF) is asserted (PIR1<0>). The timer continues to increment following a timer overflow. A CPU interrupt can be generated if bit OVFIE (PIE1<0>) is programmed to '1' (interrupt enabled). In addition, the Global Interrupt Enable (GIE, INTCON<7>) must also be set. Software is responsible for clearing the OVFIF flag prior to the next conversion cycle.

To initiate a conversion you must do three things:

1. Set the ADRST bit (ADCON0<1>) in software. The hardware responds by stopping the A/D timer (ADTMR) and discharging the external capacitor connected to the CDAC pin. The timing of the reset is determined by the software to allow enough time for the ramp capacitor to fully discharge. The minimum reset pulse width requirement for a 0.1uF ramp capacitor is 200 micro-seconds.
2. Reset ADTMR in software. ADTMR will not count while ADRST is set.
3. Reset the ADRST bit to low (0). This internal hold/discharge is released allowing the capture time to count and the ramp capacitor to begin charging. Figure 8-2 shows a typical A/D conversion cycle.

FIGURE 8-1: A/D BLOCK DIAGRAM



Note 1: All current sources are disabled if ADRST = '1'

Note 2: Approximately 5 microsecond time constant

FIGURE 8-2: EXAMPLE A/D CONVERSION CYCLE

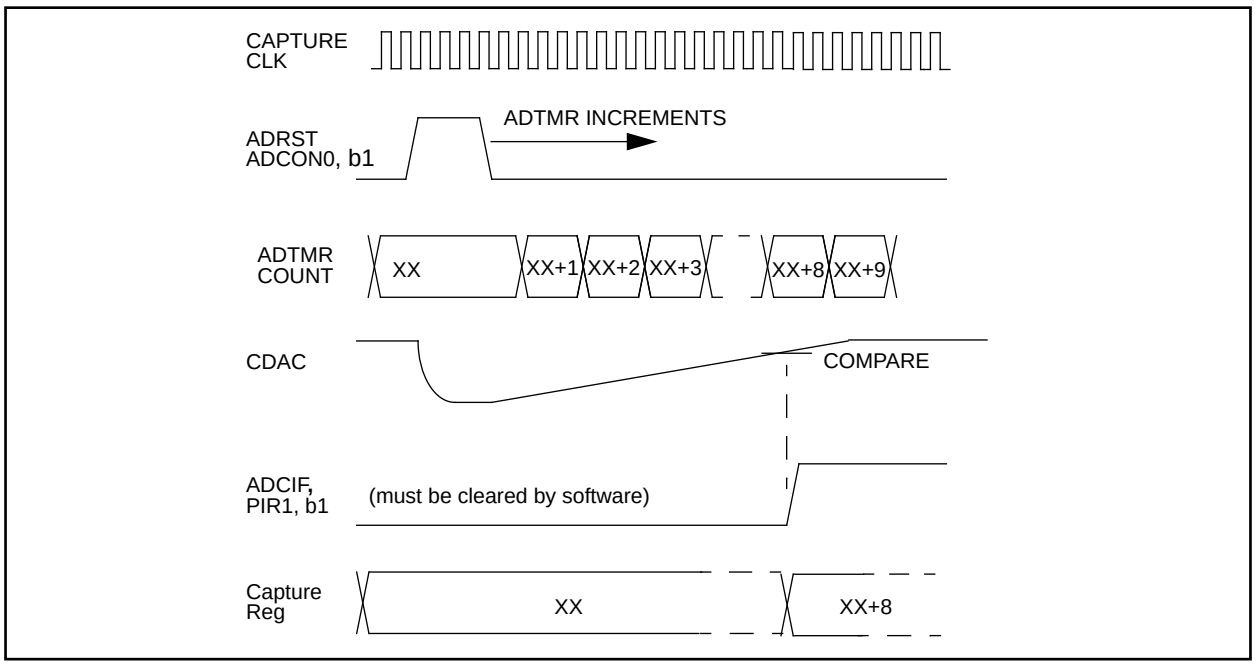


FIGURE 8-3: A/D CAPTURE TIMER (LOW BYTE)

0Eh	B7	B6	B5	B4	B3	B2	B1	B0
ADTMRL	b7	b6	b5	b4	b3	b2	b1	b0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR value 00h	0	0	0	0	0	0	0	0

FIGURE 8-4: A/D CAPTURE TIMER (HIGH BYTE)

0Fh	B7	B6	B5	B4	B3	B2	B1	B0
ADTMRH	b15	b14	b13	b12	b11	b10	b9	b8
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR value 00h	0	0	0	0	0	0	0	0

FIGURE 8-5: A/D CAPTURE REGISTER (LOW BYTE)

15h	B7	B6	B5	B4	B3	B2	B1	B0
ADCAPL	b7	b6	b5	b4	b3	b2	b1	b0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR value 00h	0	0	0	0	0	0	0	0

FIGURE 8-6: A/D CAPTURE REGISTER (HIGH BYTE)

16h	B7	B6	B5	B4	B3	B2	B1	B0
ADCAPH	b15	b14	b13	b12	b11	b10	b9	b8
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR value 00h	0	0	0	0	0	0	0	0

Legend: U= unimplemented. X = unknown.

8.4 A/D Comparator

A precision comparator is the heart of the slope A/D converter. The positive terminal of the comparator is connected to the output of an analog mux. The negative terminal is connected to the external 0.1 μF (nominal) ramp capacitor. An RC low-pass filter is connected between the output of the analog mux and the comparator input. The nominal time-constant for the RC filter is 5 μs.

8.5 Analog Mux

A total of 15 channels are internally multiplexed to the single A/D comparator positive input. Four configuration bits ADCS<3:0> (ADCON0< 7:4>) select the channel to be converted. These channels may be assigned to external thermistor, voltage, current, internal thermistor, internal bandgap voltage, charge/wake-up control DAC outputs, SREFHI and SREFLO from the slope reference divider, and ground. Additional channels are available depending on system requirements. Refer to Table 8-1.

TABLE 8-1: A/D CHANNEL SELECT DECODE

ADCS(3:0)				A/D CHANNEL
0	0	0	0	RA0/AN0 pin
0	0	0	1	RA1/AN1 pin
0	0	1	0	RA2/AN2 pin
0	0	1	1	RA3/AN3 pin
0	1	0	0	Bandgap voltage (internal)
0	1	0	1	Slope reference SREFHI (internal)
0	1	1	0	Slope reference SREFLO (internal)
0	1	1	1	Internal temperature sensor
1	0	0	0	Charge control/wake-up detect logDAC A output
1	0	0	1	Charge control/wake-up detect logDAC B output
1	0	1	0	RD4/AN4 pin
1	0	1	1	RD5/AN5 pin
1	1	0	0	RD6/AN6 pin
1	1	0	1	RD7/AN7 pin
1	1	1	0	Tied to analog ground
1	1	1	1	Tied to analog ground

For example in a dual pocket battery charger application, the following pins can be assigned to measure voltages as follows:

- RA0/AN0 pin (Battery A voltage)
- RA1/AN1 pin (Current sense A voltage)
- RA2/AN2 pin (External thermistor A voltage)
- RD4/AN4 pin (Battery B voltage)
- RD5/AN5 pin (Current sense B voltage)
- RD6/AN6 pin (External thermistor B voltage)

8.6 Programmable Slope Control DAC

Four configuration bits ADDAC<3:0> (ADCON1 <7:4>) are used to control a 4-bit current DAC for generating the comparison slope voltage to the A/D comparator. It allows slope compensation for voltage, frequency and capacitor tolerance variations. It also ensures that the dynamic range of the inputs is not compromised by the slope voltage. The current values range from 0 to 37.5 μA (nominal) in 2.5 μA increments. The intermediate values of the 4-bit DAC current source are as follows:

TABLE 8-2: A/D CDAC CURRENT DAC OUTPUT DECODE

ADDAC(3:0)				DAC CURRENT OUTPUT
0	0	0	0	OFF - all current sources disabled
0	0	0	1	2.5 μA
0	0	1	0	5 μA
0	0	1	1	7.5 μA
0	1	0	0	10 μA
0	1	0	1	12.5 μA
0	1	1	0	15 μA
0	1	1	1	17.5 μA
1	0	0	0	20 μA
1	0	0	1	22.5 μA
1	0	1	0	25 μA
1	0	1	1	27.5 μA
1	1	0	0	30 μA
1	1	0	1	32.5 μA
1	1	1	0	35 μA
1	1	1	1	37.5 μA

The 4-bit DAC output is tied to the CDAC pin and is used to charge an external 0.1uF capacitor for establishing the slope voltage to the A/D comparator. (Refer to Figure 8-1.) This capacitor should have a low voltage-coefficient for optimum results. The CDAC output must be discharged at the beginning of each conversion cycle by asserting bit ADRST (ADCON0 <1>) for at least 200 μs to allow a complete discharge. Asserting bit ADRST temporarily disables the ramp DAC current sources internally. Current flow begins with the de-assertion of ADRST. The DAC output current will be stable no more than 10 μs after de-asserting ADRST.

8.7 A/D Control Registers

Two A/D control registers are provided on PIC14000 to control the conversion process. These are ADCON0 (1Fh) and ADCON1 (9Fh). Both registers are readable and writable.

TABLE 8-3: A/D CONTROL AND STATUS REGISTER 1

1Fh	B7	B6	B5	B4	B3	B2	B1	B0
ADCON0	ADCON1 ADCS3	ADCS2	ADCS1	ADCS0	-	AMUXOE	ADRST	ADZERO
Read/Write	R/W	R/W	R/W	R/W	U	R/W	R/W	R/W
POR value 02h	0	0	0	0	0	0	1	0

Bit	Name	Function
B7-B4	ADCS3 ADCS2 ADCS1 ADCS0	A/D Channel Selects. Refer to the Table 8-1 for decoding.
B3	-	Unimplemented. Read as '0'.
B2	AMUXOE	Analog Mux Output Enable 1 = Tie AMux Output to AN0 pin 0 = AN0 pin normal
B1	ADRST	A/D Reset Control Bit 1 = Reset the A/D Capture Timer, discharge CDAC capacitor 0 = Normal operation (A/D running)
B0	ADZERO	A/D Zero Select Control. 1 = Enable zeroing operation on current sense input. 0 = Disable zeroing operation on current sense input.

TABLE 8-4: A/D CONTROL AND STATUS REGISTER 2

9Fh	B7	B6	B5	B4	B3	B2	B1	B0
ADCON1	ADDAC3	ADDAC2	ADDAC1	ADDAC0	ACFG3	ACFG2	ACFG1	ACFG0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR value 00h	0	0	0	0	0	0	0	0

Bit	Name	Function
B7-B4	ADDAC3 ADDAC2 ADDAC1 ADDAC0	A/D Current DAC Selects. Refer to Table 8-2 for decoding.
B3-B2	ACFG3 ACFG2	PORTD Configuration Selects (See Table 8-5 for decoding)
B1-B0	ACFG1 ACFG0	PORTA Configuration Selects (See Table 8-5 for decoding)

TABLE 8-5: PORTA AND PORTD CONFIGURATION SELECT DECODE

ACFG<1:0>	RA0/AN0	RA1/AN1	RA2/AN2	RA3/AN3
ACFG<3:2>	RD4/AN4	RD5/AN5	RD6/AN6	RD7/AN7
0 0	A	A	A	A
0 1	A	A	A	D
1 0	A	A	D	D
1 1	D	D	D	D

8.8 A/D Speed, Resolution and Capacitor Selection

The conversion time for the A/D converter on the PIC14000 can be calculated using the equation:

Conversion Time = (1/Fosc) x 2(N bits of resolution)
where Fosc is the oscillator frequency

N is the numbers of bits resolution desired

Therefore at 4MHz, the conversion time for 16 bits is 16.384 msec. Conversely, it is 256 µsec for 10 bits.

The PIC14000 analog peripherals form a slope voltage converter. Choosing the correct ramp capacitor for the CDAC pin (pin 22) is required to achieve the desired resolution and conversion time. The equation for selecting the ramp capacitor value is:

Capacitor = (conversion time in seconds) X (current DAC in amps) / (full scale in volts)

Table 8-6 provides example capacitor values for the desired A/D resolution, conversion time, and full scale voltage measurement.

TABLE 8-6: RAMP CAPACITOR SELECTION (EXAMPLES FOR FULL SCALE OF 3.5V AND 1.5V)

A/D Resolution (Bits)	Conversion Time (Seconds)	Full Scale (Volts)	A/D Current DAC (µamps)	Ramp Cap (Farads)	Ramp Capacitor Nearest Standard Value
16	0.016384	3.5	25	1.17E-07	.1uF
14	0.004096	3.5	25	2.93E-08	.022uF
12	0.001024	3.5	25	7.31E-09	6800pF
16	0.016384	1.5	25	2.73E-07	.022uF
14	0.004096	1.5	25	6.83E-08	6800pF
12	0.001024	1.5	25	1.71E-08	1500pF

Note: Assumes FOSC of 4MHz and current DAC value of 25 µA.

8.9 Precision Voltage Reference

The bandgap reference circuit is used to generate a precision voltage reference accurate to within 1% after calibration. The output of the bandgap reference is used to reference the A/D and the low-voltage detector. The bandgap reference is channel 4 of the analog mux and is selected using configuration bits ADCS<3:0>.

8.10 Current Reference

PIC14000 contains a current reference that generates a 1uA (nominal) current reference accurate to within 25% absolute, and less than 4% over temperature and voltage. The output of the current reference is used to generate the current sources for the comparators, DACs, temperature sensor, and the current sense biasing network.

Refer to Section 9.0 for additional details of temperature and current sense bias network.

9.0 OTHER ANALOG MODULES

PIC14000 has additional analog hardware modules to optimize performance for battery management applications. These include:

- bandgap voltage reference (refer to Section 8.9)
- current reference (refer to Section 8.10)
- charge controller/current flow detector
- internal temperature sensor
- voltage regulator control
- supply voltage divider

The analog circuitry can be shut off during idle periods for power savings. Refer to Section 10.7 for additional information on the PIC14000 power-down modes.

9.1 Current Bias and Zeroing Network

Current is measured at the RA1/BATI input by connecting an external sense resistor in series with the battery. For example, in a battery pack exhibiting charge and discharge currents of ±5A and using a 0.05 Ω sense resistor, results in voltages of -0.25V to +0.25V at the pin. A current source and resistor are used to bias this voltage into a range usable by the comparator (0.5V nominal bias). The nominal value of the bias current source is 5 μA and the resistor is 100 kΩ.

Note: The minimum voltage permissible at the RA1/BATI pin is -0.3V. The input protection diode(s) will begin to turn on beyond -0.3V, introducing significant error in the current measurement. Under no conditions should the pin voltage fall below -0.5V. The suggested system design is to orient the battery polarity so that a negative voltage occurs during charge, not discharge cycles.

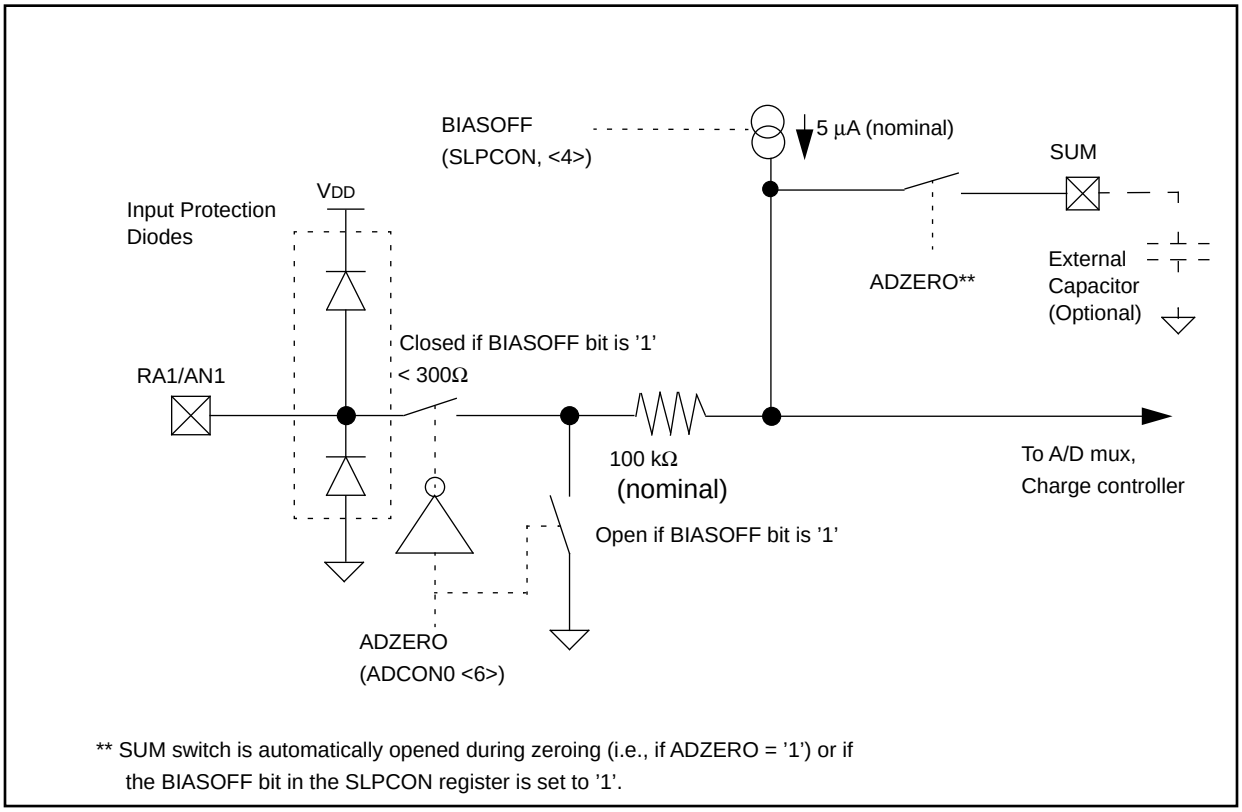
A current zeroing technique is used to increase accuracy of the current measurement. Two matched pass gates are used in the zeroing process. One gate disconnects the current sense input from the bias network. The second gate grounds the input. This simulates a zero current condition. An A/D reading is taken (zeroed). Subsequent readings are calculated relative to this zero count from the A/D. This “zeroing” of the current provides very high accuracies at low current values, where it is most needed.

For capturing short duration current pulses, such as in GSM digital cellular phone applications, an optional filter capacitor may be connected to the SUM pin to ground. This forms an RC network with the internal 100 kΩ (nominal) bias resistor to act as a DC averaging filter. The capacitor size can be adjusted for the target time constant. A switch is included between the zeroing/bias network and the SUM pin. This switch is closed during A/D sampling periods and is automatically opened during the zeroing operation (i.e., if ADZERO = ’1’). If not required in the system, this pin should be left no-connected (floating).

The current bias source can be turned off by setting the BIASOFF bit (SLPCON register 8Fh <4>) to ’1’. This also forces the ZERO and SUM switches open, so the AN1/RA1 pin can continue to be used as a general-purpose analog input.

Figure 9-1 shows the current bias and zero network in simplified form.

FIGURE 9-1: CURRENT BIAS AND ZEROING NETWORK



** SUM switch is automatically opened during zeroing (i.e., if ADZERO = '1') or if the BIASOFF bit in the SLPCON register is set to '1'.

9.2 Slope Reference Divider

A bandgap reference and amplifier with a resistor divider is used to compensate for the zero offset of the A/D converter. There may be a slight time lag (less than 10 µs) between de-assertion of the internal reset condition and the beginning of the ramp capacitor charge. Firmware must be able to determine this time lag (offset) and compensate for A/D measurements. The voltage divider provides two tap points for this purpose, named SREFHI and SREFLO. The value of SREFHI is approximately 9 times the value of SREFLO. The value of SREFHI is approximately 1.23V and is used for the slope detect upper limit in the A/D conversions. The trip point SREFLO is approximately 0.14V and is used for the slope detect lower trip point. By converting the SREFHI and SREFLO slope detect points the slope ramp offset can be calculated. The firmware can then use this information to compensate the A/D and optimize accuracy. This offset delay is subtracted from subsequent conversion outputs to arrive at the true digital count corresponding to the analog input voltage.

9.3 Internal Temperature Sensor

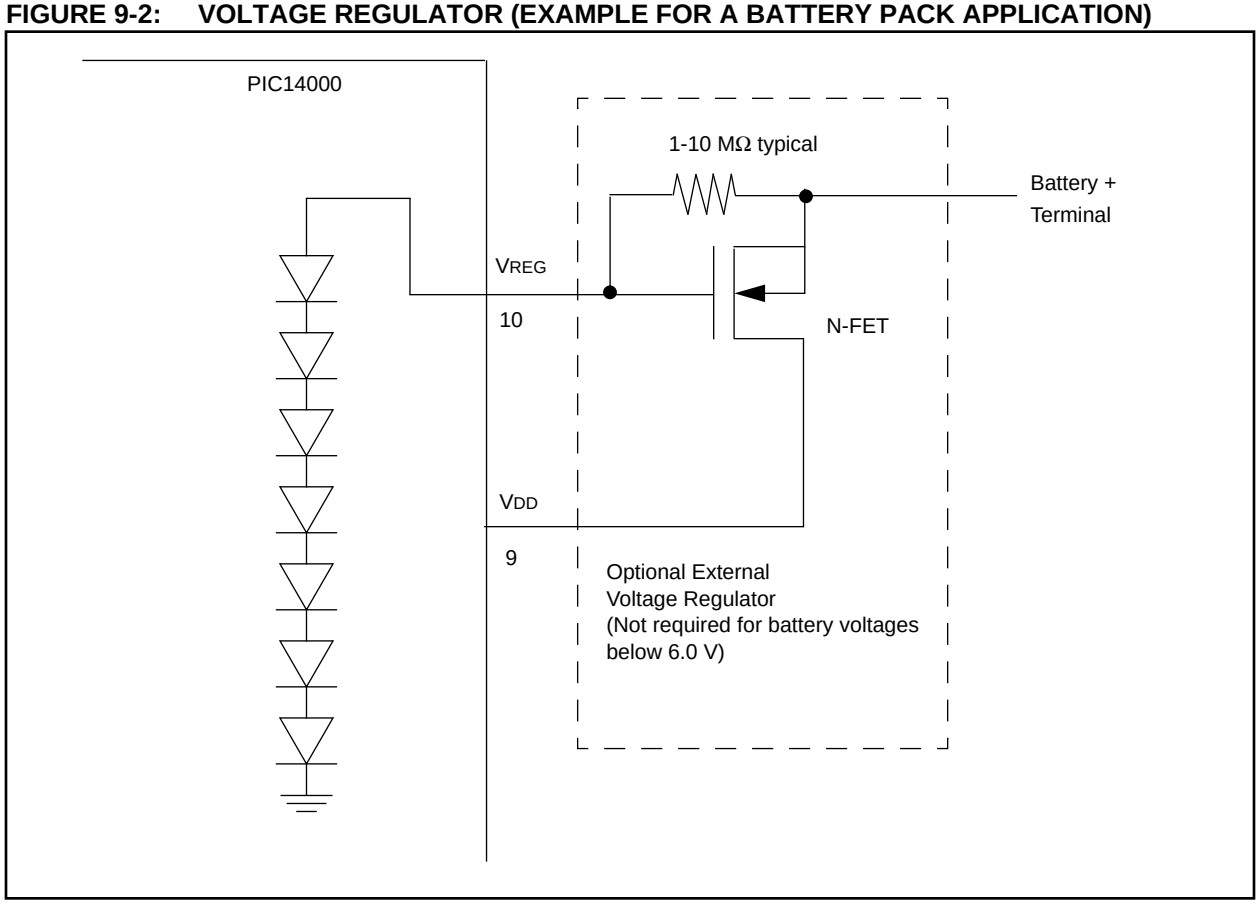
An internal temperature sensor will be used as one input to the A/D. The temperature sensor is able to measure resolutions to 0.1°C and absolute temperature to +/- 2.5°C over the 0° to 70°C temperature range. If more precise temperature monitoring is necessary, an external thermistor should be used.

Note: In most cases it is recommended that an external temperature sensor (thermistor) be used for fast charge control. The external thermistor should be affixed directly to the battery pack for reducing temperature response time and improving accuracy.

9.4 Voltage Regulator Output

For systems where the battery voltage (during charging) exceeds the maximum VDD limit of the device (6.0V), a control for an inexpensive external voltage regulator (FET + resistor) is included to reduce system cost. The VREG output pin can be connected to an external N-channel FET, which is in series with the battery voltage and the VDD pin. Its nominal output voltage is 6 V. This will provide a VDD of about 5 V, after the voltage drop across the FET. Figure 9-2 shows a typical circuit connection of the voltage regulator.

FIGURE 9-2: VOLTAGE REGULATOR (EXAMPLE FOR A BATTERY PACK APPLICATION)



9.5 Charge Controller / Current Flow Detector

PIC14000 includes much of the hardware required to form a constant-current switching regulator for battery charging. This circuit can also be used to sense current flow and generate an interrupt to the CPU. This interrupt is able to wake the device from SLEEP mode as described in Section 10.7.1. The circuit is comprised of two DACs and comparator channels A and B. Channel A is reserved for the charge control function, while both channels are used for the current flow detector to detect both positive and negative current. Each DAC output is tied to one of the comparators as shown in Figure 9-7. Two registers LDACA (9Bh) and LDACB (9Ch) are used to select the DAC output voltages.

To enable hardware charge control, the CCAEN bit (CHGCON<1>) must be set. Setting CCAEN to '1' causes the pins RC0/LDACA and RC1/CMPA to assume their charge control functions. In this situation, pin RC0/LDACA becomes the analog output from DAC "A" and is normally connected to an external filter capacitor. Pin RC1/CMPA becomes the output from the charge control comparator and is used to switch an external power FET to control the battery charge current. The charge controller also includes a comparator that examines the current sense voltage (after the bias network). During charging, the circuit acts to make the current sense voltage equal to the charge DAC output voltage effectively controlling the amount of charging current seen by the battery. The status of the charge control/current-flow detect comparators can be read via the CCOMPA and CCOMPB bits (CHGCON<2:6>). These are read-only bits and writes to these locations have no effect.

Note:	The CCAEN bit does not affect the charge DAC voltages nor comparator. These circuits are disabled by setting the CWUOFF bit in the SLPCON register (8Fh).

The two DACs are built using two resistor ladders, current source and analog multiplexers. One of the resistor ladders is used for a coarse adjustment of the output voltage. The second ladder is used to fine tune the output from the first ladder. The DAC voltage granularity and range varies depending on the value of the LDxSEL<7:0> bits in LDACA and LDACB registers. LDxSEL<7:3> selects the output from the coarse ladder according to Table 9-1, while LDxSEL<2:0> are for the fine-tune adjustment (Table 9-2).

The coarse resistor ladder is matched to the current sense bias resistor so that the center point of the ladder is approximately equal to zero current flow. This allows the DAC to control or monitor both charge and discharge current flow. This ladder is comprised of 32 taps, and is divided into two regions. The first region

is defined for controlling trickle or topping charge rates and has a range of ±50 mV. The resolution in this region is 5 mV and the range is ±50mV. This corresponds to currents of 100 mA resolution up to 1A with an external 0.05 Ω sense resistor. The second region is defined for fast charge applications. The resolution here is 50 mV (1A) with a maximum range of ±0.35V (+/- 7A, with 0.05 Ω resistor).

The fine-tune resistor ladder has 8 taps, and is used to divide the buffered output voltage from the coarse ladder. This yields a minimum DAC voltage resolution of approximately 0.714 mV or 14.3 mA with a 0.05 Ω sense resistor.

Note: The current granularity and ranges mentioned assume an external 0.05 ohm sense resistor. These values will change depending on the actual sense resistor value used.

TABLE 9-1: DIGITAL DAC DECODE (COURSE ADJUST)

LDxSEL<7:3>					NOMINAL OUTPUT VOLTAGE RANGE (V)	SENSE CURRENT RANGE (with 0.05 ohm sense resistor)-mA
0	0	0	0	0	0.5000 - 0.5050	0 - 100
0	0	0	0	1	0.5050 - 0.5100	100 - 200
0	0	0	1	0	0.5100 - 0.5150	200 - 300
0	0	0	1	1	0.5150 - 0.5200	300 - 400
0	0	1	0	0	0.5200 - 0.5250	400 - 500
0	0	1	0	1	0.5250 - 0.5300	500 - 600
0	0	1	1	0	0.5300 - 0.5350	600 - 700
0	0	1	1	1	0.5350 - 0.5400	700 - 800
0	1	0	0	0	0.5400 - 0.5450	800 - 900
0	1	0	0	1	0.5450 - 0.5500	900 - 1000
0	1	0	1	0	0.5500 - 0.6000	1000 - 2000
0	1	0	1	1	0.6000 - 0.6500	2000 - 3000
0	1	1	0	0	0.6500 - 0.7000	3000 - 4000
0	1	1	0	1	0.7000 - 0.7500	4000 - 5000
0	1	1	1	0	0.7500 - 0.8000	5000 - 6000
0	1	1	1	1	0.8000 - 0.8500	6000 - 7000
1	0	0	0	0	0.4950 - 0.5000	0 - (100)
1	0	0	0	1	0.4900 - 0.4950	(100) - (200)
1	0	0	1	0	0.4850 - 0.4900	(200) - (300)
1	0	0	1	1	0.4800 - 0.4850	(300) - (400)
1	0	1	0	0	0.4750 - 0.4800	(400) - (500)
1	0	1	0	1	0.4700 - 0.4750	(500) - (600)
1	0	1	1	0	0.4650 - 0.4700	(600) - (700)
1	0	1	1	1	0.4600 - 0.4650	(700) - (800)
1	1	0	0	0	0.4550 - 0.4600	(800) - (900)
1	1	0	0	1	0.4500 - 0.4550	(900) - (1000)
1	1	0	1	0	0.4000 - 0.4500	(1000) - (2000)
1	1	0	1	1	0.3500 - 0.4000	(2000) - (3000)
1	1	1	0	0	0.3000 - 0.3500	(3000) - (4000)
1	1	1	0	1	0.2500 - 0.3000	(4000) - (5000)
1	1	1	1	0	0.2000 - 0.2500	(5000) - (6000)
1	1	1	1	1	0.1500 - 0.2000	(6000) - (7000)

TABLE 9-2: DIGITAL DAC DECODE (FINE ADJUST)

LDxSEL<2:0>			FRACTIONAL VALUE OF THE COARSE RANGE
0	0	0	1/8
0	0	1	1/4
0	1	0	3/8
0	1	1	1/2
1	0	0	5/8
1	0	1	3/4
1	1	0	7/8
1	1	1	MAXIMUM

For example, if a topping off current of 340 mA was required, LDxSEL<7:4> would be set to a value of '00011010' binary. This yields a coarse range of 300-400 mA. The fine-tune setting selects 3/8 times the coarse range maximum or approximately 37.5 mA.

Two polarity bits are provided in the CHGCON register that allow the comparator outputs to be inverted under firmware control. This allows the same comparators to be used as both a charge control output and for the current flow detector, for generating a positive triggered interrupt.

FIGURE 9-3: DAC TRANSFER FUNCTION

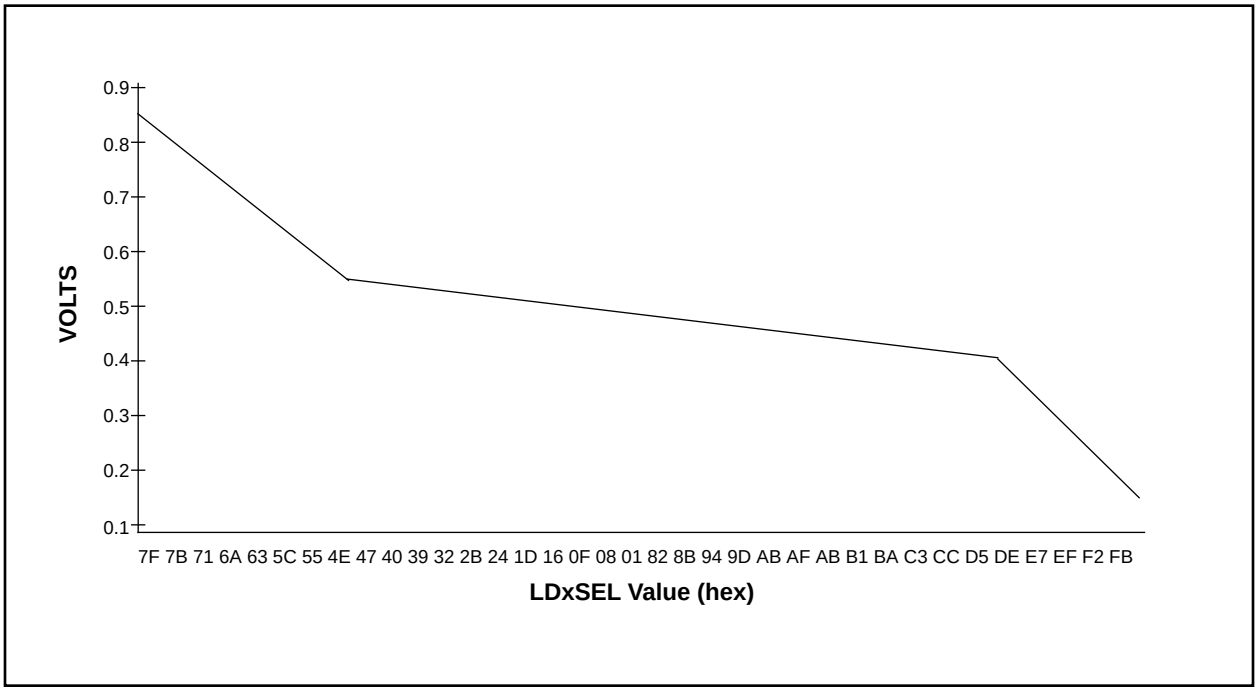


FIGURE 9-4: CHARGE/CURRENT FLOW DETECT CONTROL REGISTER

9Dh	B7	B6	B5	B4	B3	B2	B1	B0
CHGCON	U	CCOMPB	CCBEN	CPOLB	U	CCOMPA	CCAEN	CPOLA
Read/Write	-	R	R/W	R/W	-	R	R/W	R/W
POR value 00h	0	0	0	0	0	0	0	0

Bit	Name	Function
B7	-	Unimplemented. Read as '0'.
B6	CCOMPB	Charge Control Comparator Output B. Reading this bit returns the status of the charge control/wake-up comparator B output before the inverter stage. Writes to this bit have no effect.
B5	CCBEN	Charge Control Function Enable Bit. (Channel B) 1 = Charge Control is Enabled. RD2/CMPB and RD3/LDACB are used to control switching regulator. 0 = Charge Control is Disabled (default). RD2/CMPB and RD3/LDACB assume normal PORTD function.
B4	CPOLB	Charge Control Polarity Bit B 1 = Invert the output from the charge/wake-up comparator B. 0 = Do not invert the output from the charge/wake-up comparator B (default)
B3	-	Unimplemented. Read as '0'.
B2	CCOMPA	Charge Control Comparator Output A. Reading this bit returns the status of the charge control/wake-up comparator A output before the inverter stage. Writes to this bit have no effect.
B1	CCAEN	Charge Control Function Enable Bit. (Channel A) 1 = Charge Control is Enabled. RC0/LDACA and RC1/CMPA are used to control switching regulator. 0 = Charge Control is Disabled (default). RC0/LDACA and RC1/CMPA assume normal PORTC function.
B0	CPOLA	Charge Control Polarity Bit A 1 = Invert the output from the charge/wake-up comparator A 0 = Do not invert the output from the charge/wake-up comparator A (default)

FIGURE 9-5: LDACA REGISTER

9Bh	B7	B6	B5	B4	B3	B2	B1	B0
LDACA	LDASEL7	LDASEL6	LDASEL5	LDASEL4	LDASEL3	LDASEL2	LDASEL1	LDASEL0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR value 00h	0	0	0	0	0	0	0	0

Bit	Name	Function
B7-B0	LDASEL7 LDASEL6 LDASEL5 LDASEL4 LDASEL3 LDASEL2 LDASEL1 LDASEL0	DAC A Voltage Select Bits. See Table 9-1 and Table 9-2 for decoding.

FIGURE 9-6: LDACB REGISTER

9Ch	B7	B6	B5	B4	B3	B2	B1	B0
LDACB	LDBSEL7	LDBSEL6	LDBSEL5	LDBSEL4	LDBSEL3	LDBSEL2	LDBSEL1	LDBSEL0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR value 00h	0	0	0	0	0	0	0	0

Bit	Name	Function
B7-B0	LDBSEL7 LDBSEL6 LDBSEL5 LDBSEL4 LDBSEL3 LDBSEL2 LDBSEL1 LDBSEL0	DAC B Voltage Select Bits. See Table 9-1 and Table 9-2 for decoding.

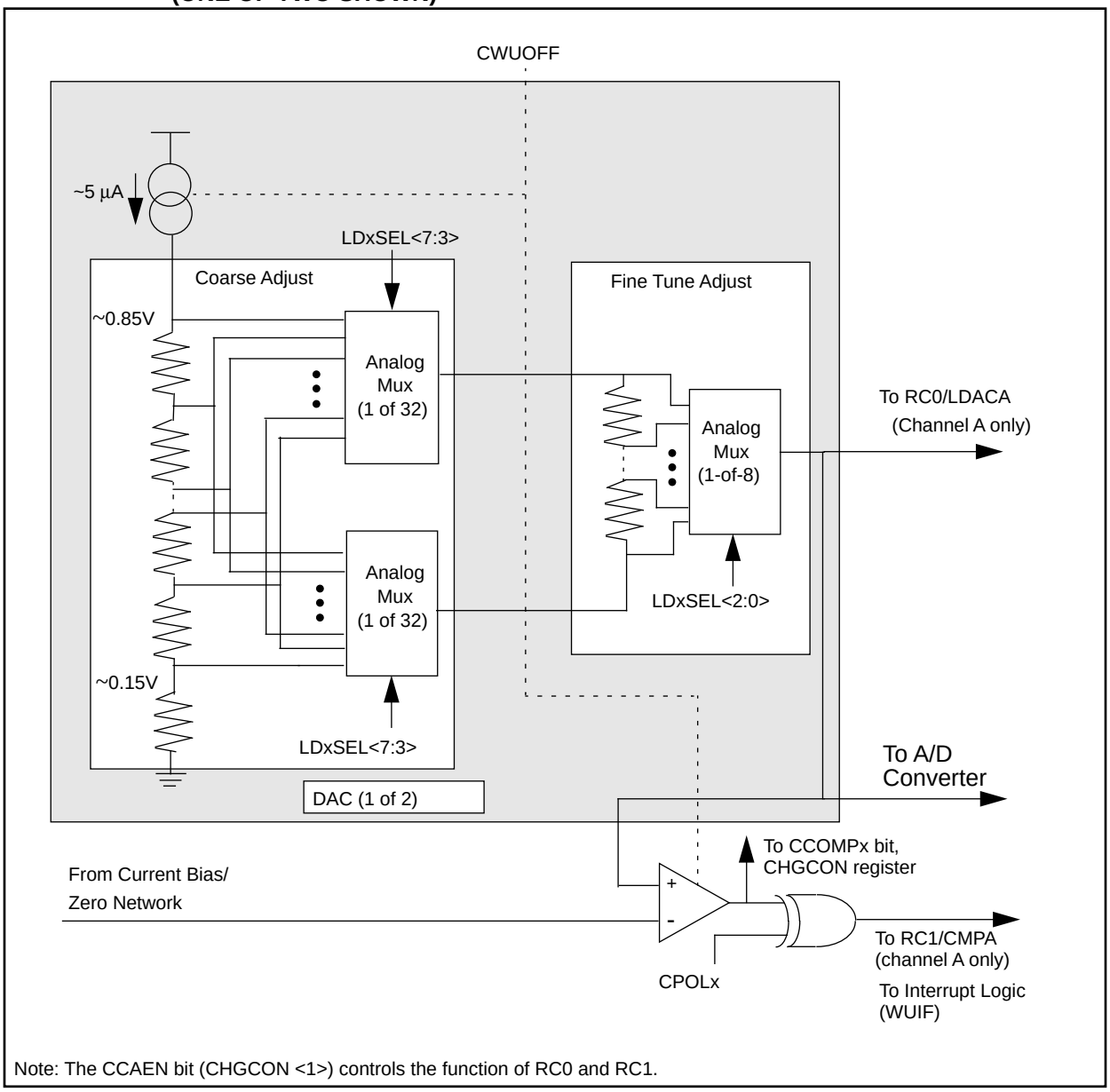
9.5.1 WAKE-UP ON CURRENT DETECT

The charge controller/current flow detector can also be used to detect current flow during SLEEP or idle modes and cause an interrupt to force a system wake-up. The two DACs are used to adjust the current wake-up threshold by programming the LDxSEL<7:4> bits in the LDACA (9Bh) and LDACB (9Ch) registers according to Table 9-1 and Table 9-2. By programming the two DACs to detect opposite polarities, both positive and negative current flow can cause an interrupt. Either DAC/comparator pair can cause a CPU interrupt. The interrupt flag (WUIF) is located in the PIR1<7> and is enabled by the WUIE bit in the PIE1 register. The enable bit must be set to '1' to enable the CPU interrupt. The outputs of the two comparators can be read (CCOMPA, CCOMPB) to determine which of the two detectors caused the interrupt.

Note: The wake-up on current flow feature can be used even though charging is not used in the system. In this case, the CCAEN bit (CHGCON<1>) would always remain cleared to '0'.

When using this feature, the CWUOFF and BIASOFF bits must be cleared to '0'. This enables charge controller/wake-up DACs, comparators and the current sense bias source.

FIGURE 9-7: CHARGE CONTROL/CURRENT FLOW DETECT BLOCK DIAGRAM (ONE OF TWO SHOWN)



Note: The CCAEN bit (CHGCON <1>) controls the function of RC0 and RC1.

10.0 SPECIAL FEATURES OF THE CPU

What sets apart a microcontroller from other processors are special circuits to deal with the needs of real time applications. PIC14000 has a host of such features intended to maximize system reliability, minimize cost through elimination of external components, provide power saving operating modes and offer code protection. These are:

1. OSC (oscillator) selection
   - Crystal/resonator
   - Internal oscillator
2. Reset options
   - Power-on Reset
   - Power-up Timer
   - Oscillator Start-up Timer
3. Interrupts
4. Watchdog Timer
5. SLEEP and HIBERNATE modes
6. Code protection
7. In-circuit serial programming

These features will be described in the following sections.

10.1 Oscillator Configurations

PIC14000 can be operated with two different oscillator options. The user can program a configuration fuse (FUSES<0>) to select one of these:

- HS High Speed Crystal/Ceramic Resonator (FOSC ='0')
- IN Internal oscillator (FOSC ='1')(Default)

The FOSC fuse bit is located in configuration word 2007h.

10.1.1 INTERNAL OSCILLATOR CIRCUIT

PIC14000 includes an internal oscillator option that offers additional cost and board-space savings. No external components are required. The nominal operating frequency is 4 MHz, with an absolute frequency tolerance of +/- 20%. Frequency drift over voltage and temperature must remain below 5%. The frequency is measured and stored into the calibration space in EPROM (address 0FD0h).

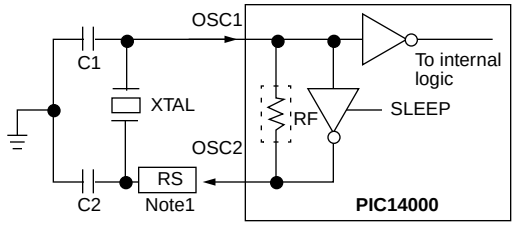
By selecting IN mode (FOSC fuse bit = '1'), OSC1/PBTN becomes a digital input (with weak internal pull-up resistor) and can be read via bit MISC<0>. Writes to this location have no effect. The OSC1/PBTN input is capable of generating an interrupt to the CPU if enabled (Section 10.5). Also, OSC2 pin becomes a digital output for general purpose use and is accessed via MISC <1>. Writes to bit7 directly affects the OSC2 pin. Reading this bit returns the contents of the output latch. The MISC register format is shown in Figure 10-5.

The OSC2 pin also outputs the IN oscillator frequency, divided-by-four, via INCLKEN (MISC <2>).

10.1.2 CRYSTAL OSCILLATOR / CERAMIC RESONATOR

In HS mode, a crystal or ceramic resonator is connected to the OSC1 and OSC2 pins to establish oscillation. A parallel cut crystal is required. Use of a series cut crystal may give a frequency outside of the crystal manufacturer's specifications. When in HS mode, the device can have an external clock source to drive the OSC1 pin.

FIGURE 10-1: CRYSTAL/CERAMIC RESONATOR OPERATION (HS OSC CONFIGURATION)



See Table 10-1 and Table 10-2 for recommended values of C1 and C2.

Note 1: A series resistor may be required for AT strip cut crystals.

FIGURE 10-2: EXTERNAL CLOCK INPUT OPERATION (HS OSC CONFIGURATION)

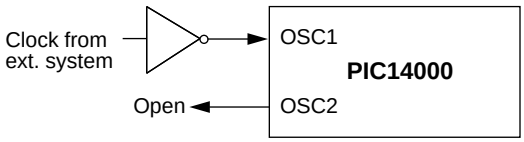


TABLE 10-1: CERAMIC RESONATORS

Mode	Freq	OSC1	OSC2
HS	4.0MHz	15 - 68 pF	15 - 68 pF
	8.0MHz	10 - 68 pF	10 - 68 pF
	16.0MHz	10 - 22 pF	10 - 22 pF

Note: Recommended values of C1 and C2 are identical to the ranges tested table.

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

Resonators Used:

4.0MHz	Murata Erie CSA4.00MG	+/-.5%
8.0MHz	Murata Erie CSA8.00MT	+/-.5%
16.0MHz	Murata Erie CSA16.00MX	+/-.5%

All resonators used did not have built-in capacitors.

TABLE 10-2: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR

Mode	Freq	OSC1	OSC2
HS	4MHz	15 - 33 pF	15 - 33 pF
	8MHz	15 - 47 pF	15 - 47 pF
	20MHz	15 - 47 pF	15 - 47 pF

Note: Higher capacitance increases the stability of oscillator but also increases the start-up time. These values are for design guidance only. Rs may be required in HS 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.

§ For V_{DD} > 4.5V, C1 = C2 ≈ 30pf is recommended.

10.1.3 EXTERNAL CRYSTAL OSCILLATOR CIRCUIT

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

Figure 10-3 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 potentiometer biases the 74AS04 in the linear region. This could be used for external oscillator designs.

FIGURE 10-3: EXTERNAL PARALLEL RESONANT CRYSTAL OSCILLATOR CIRCUIT

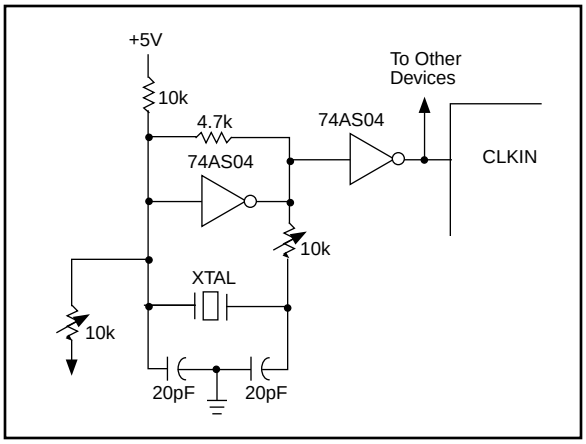
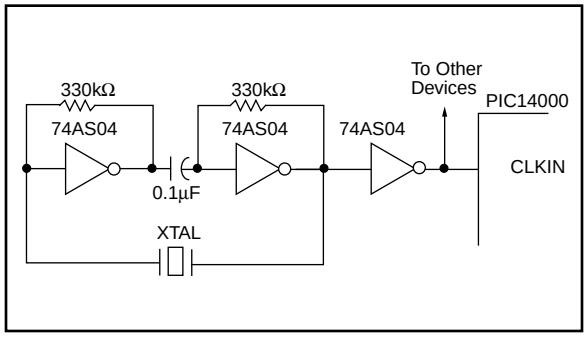


Figure 10-4 shows a series resonant oscillator circuit. This circuit is also designed to use the fundamental frequency of the crystal. The inverter performs a 180-degree phase shift in a series resonant oscillator circuit. The 330 kΩ resistors provide the negative feedback to bias the inverters in their linear region.

FIGURE 10-4: EXTERNAL SERIES RESONANT CRYSTAL OSCILLATOR CIRCUIT



10.2 Reset Operation

PIC14000 differentiates between various kinds of reset:

- Power-on/low-voltage detect reset ($\overline{POR}$)
- $\overline{MCLR}$ reset during normal operation
- $\overline{MCLR}$ reset during SLEEP
- WDT time-out during normal operation
- WDT time-out during SLEEP

Some registers are not affected in any reset condition; their status is unknown on $\overline{POR}$ and unchanged in any other reset. Most other registers are reset to a defined state on power-on reset ($\overline{POR}$), on $\overline{MCLR}$ or WDT reset during normal operation, and on $\overline{MCLR}$ reset during SLEEP. They are not affected by a WDT reset during SLEEP, since this reset is viewed as the resumption of normal operation. $\overline{TO}$ and $\overline{PD}$ bits are set or cleared differently in different reset situations as indicated in Table 10-3. These bits are used in software to determine the nature of reset.

It is recommended that a checksum comparison at $\overline{POR}$ to ensure integrity of the data space contents.

A simplified block diagram of the on-chip reset circuit is shown in Figure 10-6.

FIGURE 10-5: MISC REGISTER

9Eh	B7	B6	B5	B4	B3	B2	B1	B0
MISC	SMHOG	SPGNDB	SPGNDA	I^2CSEL	SMBUS	INCLKEN	OSC2	OSC1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R
POR value 00h	0	0	0	0	0	0	0	X

Bit	Name	Function
B7	SMHOG	SMHOG enable 1 = Stretch I^2C CLK signal (hold low) when receive data buffer is full (refer to Section 7.1.4). For pausing I^2C transfers while preventing interruptions of A/D conversions. 0 = Disable I^2C CLK stretch. (default)
B6	SPGNDB	Serial Port Ground Select 1 = PORTD<1:0> ground reference is the RD5/AN5 pin. 0 = PORTD<1:0> ground reference is Vss.
B5	SPGNDA	Serial Port Ground Select 1 = PORTC<7:6> ground reference is the RA1/AN1 pin. 0 = PORTC<7:6> ground reference is Vss.
B4	I^2CSEL	I^2C Port select Bit. 1 = PORTD<1:0> are used as the I^2C clock and data lines. 0 = PORTC<7:6> are used as the I^2C clock and data lines.
B3	SMBus	SMBus-Compatibility Select 1 = SMBus compatibility mode is enabled. PORTC<7:6> have SMBus-compatible input thresholds. 0 = SMBus-compatibility is disabled (default). PORTC<7:6> have Schmitt trigger input thresholds.
B2	INCLKEN	Oscillator Output Select 1 = Output IN oscillator signal on OSC2 pin. 0 = Disconnect IN oscillator signal from OSC2 pin. (default)
B1	OSC2	OSC2 output port bit (available in IN mode only). Writes to this location affect the OSC2 pin in IN mode. Reads return the value of the output latch.
B0	OSC1	OSC1 input port bit (available in IN mode only). Reads from this location return the status of the OSC1 pin in IN mode. Writes have no effect.

TABLE 10-3: STATUS BITS AND THEIR SIGNIFICANCE

$\overline{POR}$	$\overline{TO}$	$\overline{PD}$	Meaning
0	1	1	Power-On Reset
0	0	X	Illegal, $\overline{TO}$ is set on $\overline{POR}$
0	X	0	Illegal, $\overline{PD}$ is set on $\overline{POR}$
1	0	1	WDT reset during normal operation
1	0	0	WDT time-out wakeup from sleep
1	1	1	$\overline{MCLR}$ reset during normal operation
1	1	0	$\overline{MCLR}$ reset during SLEEP or HIB, or interrupt wake-up from SLEEP or HIB.

10.3 Low-Voltage Detector

PIC14000 contains an integrated low-voltage detector that uses a precision voltage reference. The supply voltage is divided by two and compared to the bandgap reference output (approximately 1.23V). If the supply voltage (V_{DD}) falls below Vtrip- for greater than 1 us (nominal), then the low-voltage detector will cause the $\overline{LVD}$ bit in register PCON (8Eh) to be reset. This bit must be read by software

The approximate values of the low-voltage detector trip points are as follows:

- Vtrip- = 2.55V +/- 0.1V
- Vtrip+ = 2.60V +/- 0.1V
- Hysteresis Vtrip-/Vtrip+ = 50 mV

10.4 Power-Up Timer (PWRT) and Oscillator Start-up Timer (OST)

10.4.1 V_{DD} RISE DETECTOR

A V_{DD} rise detector is included to ensure a clean reset under fast V_{DD} ramps where the bandgap reference (and low-voltage detector) may not have enough time to start up and stabilize. A $\overline{POR}$ pulse is generated on-chip when V_{DD} rise is detected (in the range of 2.0-2.2V). To take advantage of the $\overline{POR}$, tie $\overline{MCLR}$ directly (or through a resistor) to V_{DD}. This circuit has a maximum V_{DD} slew rate specification (see electrical specifications). However, this circuit, when used in combination with the DC low-voltage detector, effectively guarantee a proper reset regardless of the V_{DD} ramp time.

The V_{DD} rise detector does not produce an internal reset when V_{DD} declines.

10.4.2 POWER-UP TIMER (PWRT)

The Power-up Timer provides a fixed 72 ms (nominal) time-out on power-up only, from $\overline{POR}$. The power-up timer operates from a local internal oscillator. The chip is kept in reset as long as PWRT is active. The PWRT delay allows the V_{DD} to rise to an acceptable level. A configuration fuse, PWRTE, can disable (if set, or unprogrammed) or enable (if cleared, or programmed) the power-up timer.

The power-up timer delay will vary from chip to chip and due to V_{DD} and temperature.

10.4.3 OSCILLATOR START-UP TIMER (OST)

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

The OST time-out provides an 8-cycle delay with IN mode only after a Power-on Reset (POR) or wake-up from SLEEP.

10.4.4 TIMEOUT SEQUENCE

On power-up the time-out sequence is as follows: First the PWRT time-out is invoked after $\overline{POR}$ has expired. Then OST is activated in HS (crystal oscillator) mode. The total time-out will vary based on the oscillator configuration and PWRTE fuse status. For example, in IN mode, with PWRTE fuse unprogrammed (PWRT disabled), there will be no time-out delay at all. Figure 13-4 depicts the power-on reset time-out sequences.

Table 10-4 shows the reset conditions for some special registers, while Table 10-5 shows the reset conditions for all registers.

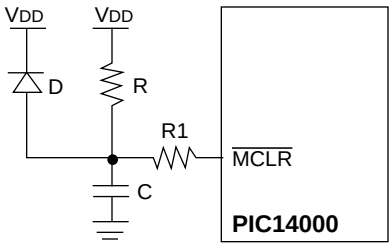
TABLE 10-4: RESET CONDITION FOR SPECIAL REGISTERS

	PCL Addr: 02h	STATUS Addr: 03h	PCON Addr: 8Eh
Power-on Reset	000h	0001 1xxx	0--- --0x
$\overline{MCLR}$ reset during normal operation	000h	0001 1uuu	u--- --ux
$\overline{MCLR}$ reset during SLEEP	000h	0001 0uuu	u--- --ux
WDT reset during normal operation	000h	0000 1uuu	u--- --ux
WDT during SLEEP	PC + 1	uuu0 0uuu	u--- --ux
Interrupt wake-up from SLEEP	PC + 1 (1)	uuu1 0uuu	u--- --ux

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

(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).

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



1. External power-on reset circuit is required only if V_{DD} power-up slope is too slow. The diode D helps discharge the capacitor quickly when V_{DD} powers down.
2. R < 40 KΩ is recommended to make sure that voltage drop across R does not exceed 0.2V (max leakage current spec on $\overline{MCLR}$ pin is 5 μA). A larger voltage drop will degrade VIH level on $\overline{MCLR}$ pin.
3. R1 = 100 Ω to 1 KΩ will limit any current flowing into $\overline{MCLR}$ from external capacitor C in the event of $\overline{MCLR}$ pin breakdown due to ESD or EOS.

TABLE 10-5: RESET CONDITION FOR REGISTERS

Register	Address	Power-on Reset	MCLR reset during - normal operation - SLEEP WDT time-out during normal operation	Wake-up from SLEEP through interrupt Wake up from SLEEP through WDT time-out
W	-	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF	00h/80h	-	-	-
TMR0	01h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCL	02h/82h	0000h	0000h	PC + 1 (2)
STATUS	03h/83h	0001 1xxx	000? ?uuu (3)	uuu? ?uuu (3)
FSR	04h/84h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA	05h	---- xxxx	---- uuuu	---- uuuu
PORTC	07h	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCLATH	0Ah/8Ah	---0 0000	---0 0000	---u uuuu
INTCON	0Bh/8Bh	0000 000x	0000 000u	uuuu uuuu (1)
PIR1	0Ch	0000 0000	0000 0000	uuuu uuuu (1)
ADTMRL	0Eh	0000 0000	0000 0000	uuuu uuuu
ADTMRH	0Fh	0000 0000	0000 0000	uuuu uuuu
I2CBUF	13h	xxxx xxxx	uuuu uuuu	uuuu uuuu
I2CCON	14h	0000 0000	0000 0000	uuuu uuuu
ADCAPL	15h	0000 0000	0000 0000	uuuu uuuu
ADCAPH	16h	0000 0000	0000 0000	uuuu uuuu
ADCON0	1Fh	0000 0010	0000 0010	uuuu uuuu
OPTION	81h	1111 1111	1111 1111	uuuu uuuu
TRISA	85h	---- 1111	---- 1111	---- uuuu
TRISC	87h	1111 1111	1111 1111	uuuu uuuu
PIE1	8Ch	0000 0000	0000 0000	uuuu uuuu
PCON	8Eh	---- --00	---- --uu	---- --uu
SLPCON	8Fh	0011 1111	0011 1111	uuuu uuuu
I2CADD	93h	0000 0000	0000 0000	uuuu uuuu
I2CSTAT	94h	--00 0000	--00 0000	--uu uuuu
LDACA	9Bh	0000 0000	0000 0000	uuuu uuuu
LDACB	9Ch	0000 0000	0000 0000	uuuu uuuu
CHGCON	9Dh	0x00 0x00	0x00 0x00	uuuu uuuu
MISC	9Eh	0000 000x	0000 000x	uuuu uuuu
ADCON1	9Fh	0000 0000	0000 0000	uuuu uuuu

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

10.5 Interrupts

PIC14000 has several sources of interrupt:

- External interrupt from OSC1/PBTN pin
- I^2C port interrupt
- PORTC change interrupt on change (pins RC<7:4> only)
- Timer0 overflow
- A/D capture timer overflow
- A/D converter capture event (conversion-complete)
- Wake-up on current detect

This section addresses the external and Timer0 interrupts only. Refer to the appropriate sections for description of the serial port, wake-up and A/D interrupts.

The interrupt control register (INTCON, address 0Bh or 8Bh) records individual interrupt requests in flag bits. It also has individual and global enable bits. The peripheral interrupt flags reside in the PIR1 register (0Ch). Peripheral interrupt enable interrupts are contained in the PIE1 register (8Ch).

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

When an interrupt is serviced, the GIE is cleared to disable any further interrupt, the return address is pushed onto the stack and the PC is loaded with 0004h, the interrupt vector. For external interrupt events, such as the I^2C interrupt, the interrupt latency will be 3 or 4 instruction cycles. The exact latency depends when the interrupt event occurs. The latency is the same for 1 or 2 cycle instructions. Once in the interrupt service routine the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt flag bit(s) must be cleared in software before re-enabling interrupts to avoid infinite interrupt requests. Individual interrupt flag bits are set regardless of the status of their corresponding mask bit or the GIE bit to allow polling.

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

Note 1: The individual interrupt flags will be set by the specified condition even though the corresponding interrupt enable bit is cleared (interrupt disabled) or the GIE bit is cleared (all interrupts disabled).

Note 2: If an interrupt occurs while the Global Interrupt Enable (GIE) bit is being cleared, the GIE bit may unintentionally be re-enabled by the user's Interrupt Service Routine (the `RETFIE` instruction). The events that would cause this to occur are:

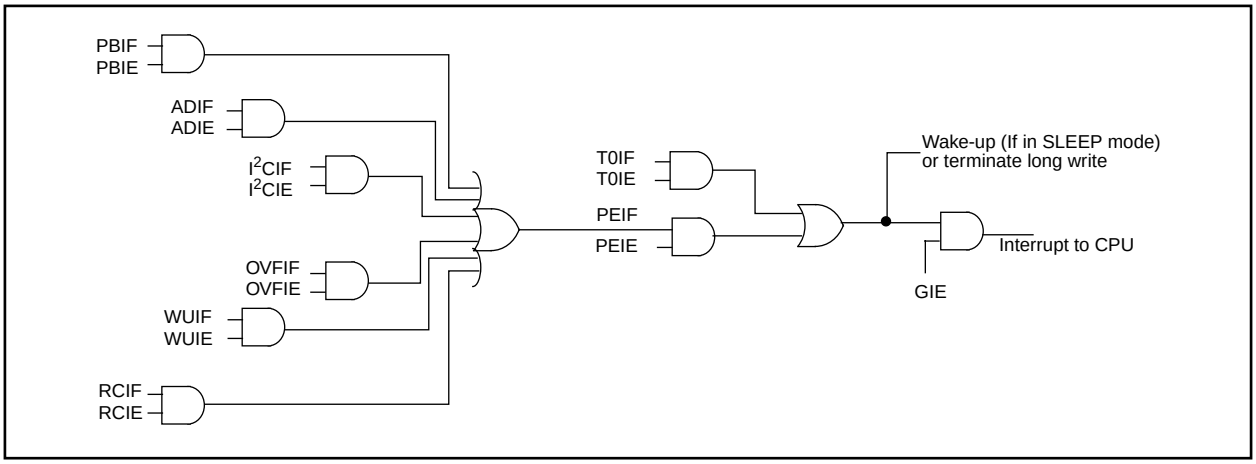
1. An instruction clears the GIE bit while an interrupt is acknowledged.
2. The program branches to the interrupt vector and executes the Interrupt Service Routine.
3. The interrupt service routine completes with the execution of the `RETFIE` instruction. This causes the GIE bit to be set (enables interrupts), and the program returns to the instruction after the one which was meant to disable interrupts.

The method to ensure that interrupts are globally disabled is:

1. Ensure that the GIE bit was cleared by the instruction, as shown in the following code:

```
LOOP: BCF   INTCON,GIE ; Disable Global Interrupts
      BTFSC INTCON,GIE ; Global Interrupts Disabled?
      GOTO  LOOP       ; No, try again
      :                ; Yes, continue with program
                       ;     flow
```

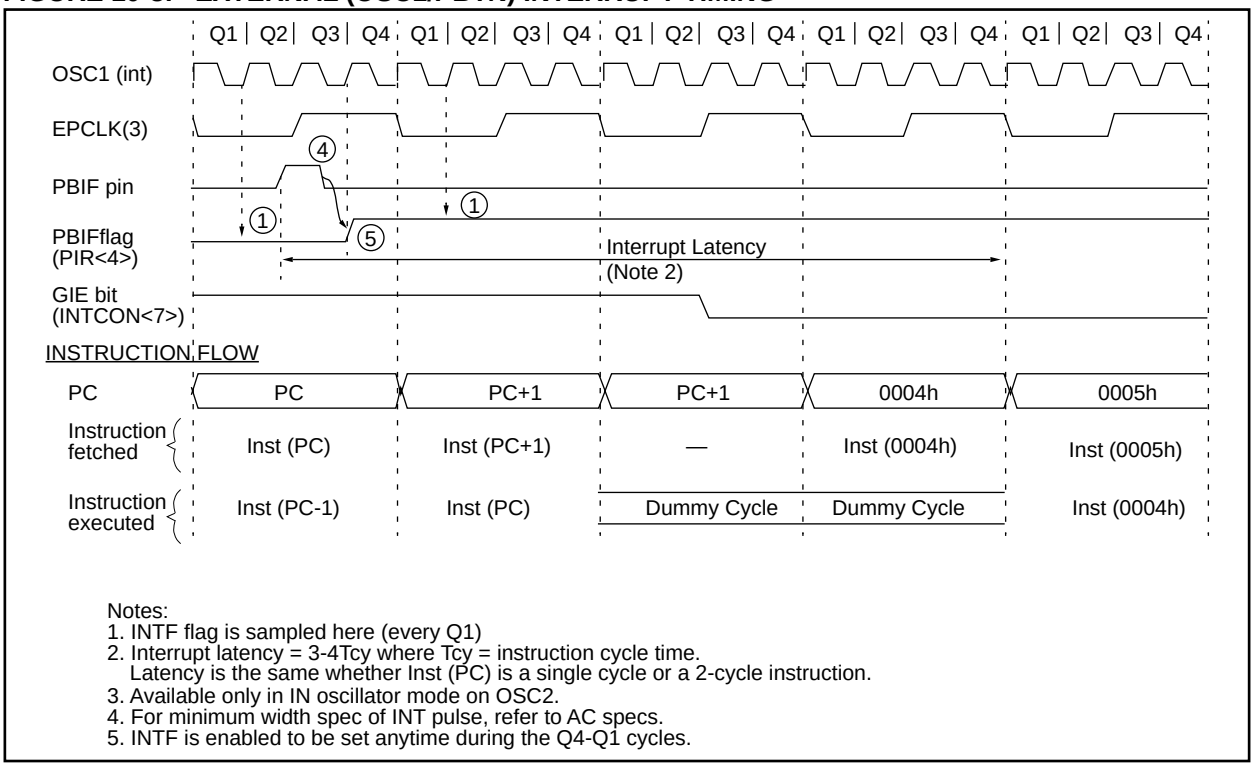
FIGURE 10-7: INTERRUPT LOGIC SCHEMATIC



10.5.1 EXTERNAL INTERRUPT

An external interrupt can be generated via the OSC1/PBTN pin if IN mode is enabled. A weak pull-up is also enabled on this pin if IN mode is enabled. These features are unavailable in HS (crystal/resonator) mode. This interrupt is falling edge triggered. When a valid edge appears on OSC1/PBTN pin, the PBIF bit is set (PIR1<4>). This interrupt can be disabled by clearing the PBIE control bit (PIE1 <4>). The PBIF bit must be cleared in software in the interrupt service routine before re-enabling the interrupt. The PBTN interrupt can wake up the processor from SLEEP if the PBIE bit is set (interrupt enabled) prior to going into SLEEP mode. The status of the GIE bit determines whether or not the processor branches to the interrupt vector following wake-up. The timing of the external interrupt is shown in Figure 10-8.

FIGURE 10-8: EXTERNAL (OSC1/PBTN) INTERRUPT TIMING



Notes:
1. INTF flag is sampled here (every Q1)
2. Interrupt latency = 3-4Tcy where Tcy = instruction cycle time.
   Latency is the same whether Inst (PC) is a single cycle or a 2-cycle instruction.
3. Available only in IN oscillator mode on OSC2.
4. For minimum width spec of INT pulse, refer to AC specs.
5. INTF is enabled to be set anytime during the Q4-Q1 cycles.

10.5.2 TIMER0 INTERRUPT

An overflow (FFh -> 00h) in Timer0 will set the T0IF (INTCON <2>) flag. T0IE (INTCON <5>) controls the interrupt.

10.5.3 PORTC INTERRUPT ON CHANGE

An input change on PORTC <7:4> sets the RCIF (PIR1<2>) bit. RCIE (PIE1 <2>) controls the interrupt. For operation of PORTC, refer to Section 5.2.

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

10.5.4 CONTEXT SWITCHING 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 for example, W register and Status register. Example 10-1 is an example that shows saving registers in RAM.

EXAMPLE 10-1: SAVING W REGISTER AND STATUS IN RAM

```
push:   MOVWF   TEMP_W          ; Copy W to temp register
        SWAPF   STATUS,W        ; Swap status to be saved into W
        BCF     STATUS,RP0      ; Select bank 0
        MOVWF   TEMP_STAT       ; Save status to temp_stat regis-
ter
        :                       ;
        :       (ISR)           ;
        :
pop:    SWAPF   TEMP_STAT, W    ; Swap temp_stat reg into W (sets
bank
                                ; to original state)
        MOVWF   STATUS          ; Move W into status register
        SWAPF   TEMP_W, F       ; Do not want to
        SWAPF   TEMP_W, W       ; affect Z-bit
```


10.6 Watchdog Timer (WDT)

The watchdog timer is realized as a free running on-chip RC oscillator which does not require any external components. This RC oscillator is separate from the IN oscillator used to generate the CPU and A/D clocks. That means that the WDT will run even if the clock has been stopped, for example, by execution of a SLEEP instruction. Refer to Section 10.7.1 for more information.

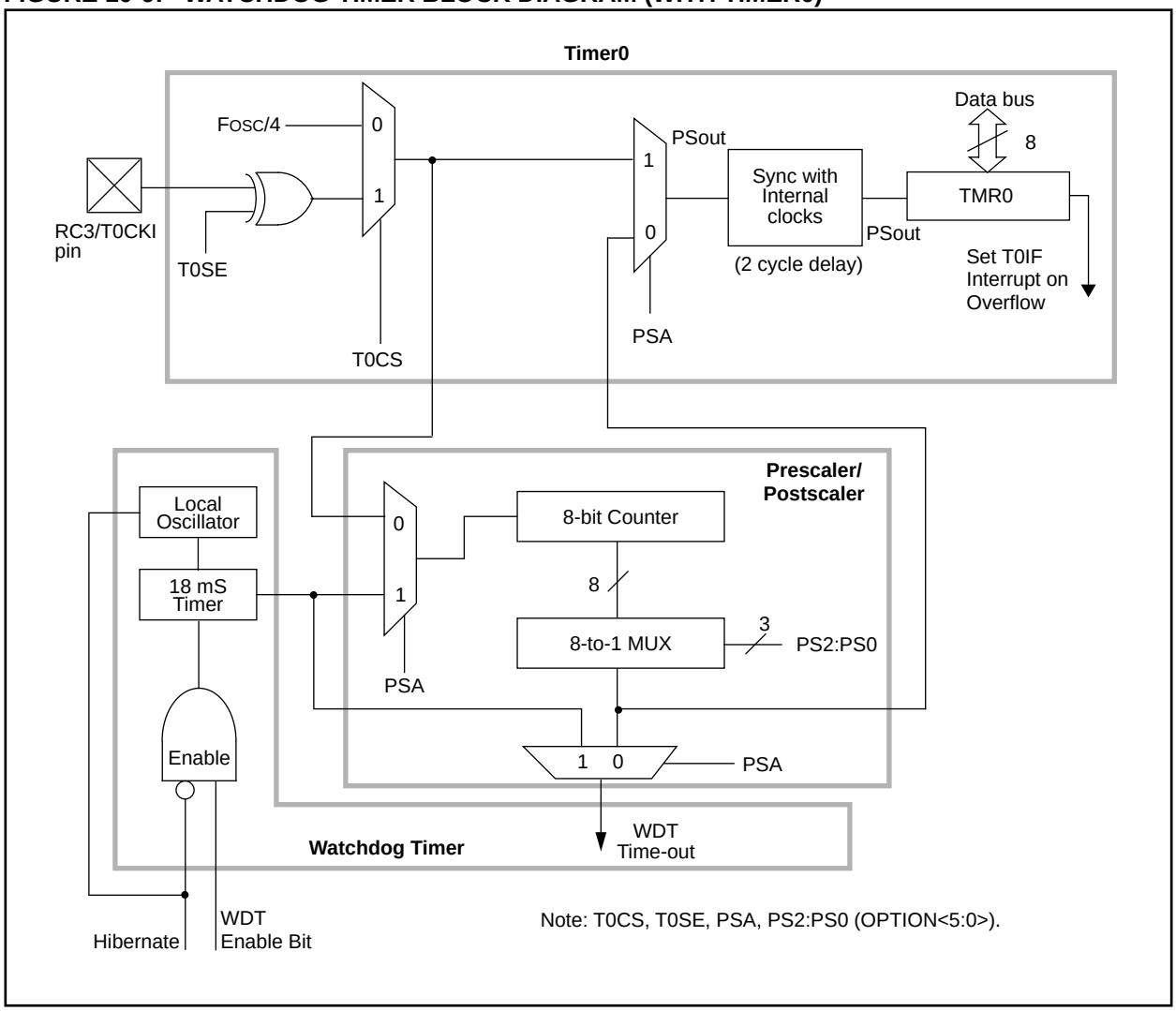
During normal operation, a WDT time-out generates a device RESET. If the device is in SLEEP mode, a WDT time-out causes the device to wake-up and continue with normal operation.

The WDT can be permanently disabled by programming the configuration fuse WDTE as a '0'. Its oscillator can be shut down to conserve battery power by entering HIBERNATE Mode. Refer to Section 10.7.3 for more information on HIBERNATE mode.

CAUTION: Beware of disabling WDT if software routines require exiting based on WDT reset. For example, the MCU will not exit HIBERNATE mode based on WDT reset.

A block diagram of the watchdog timer is shown in Figure 10-9. It should be noted that a RESET generated by the WDT time-out does not drive $\overline{\text{MCLR}}$ low.

FIGURE 10-9: WATCHDOG TIMER BLOCK DIAGRAM (WITH TIMER0)



10.6.1 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 registers. Thus, time-out periods up to 2.3 seconds can be realized. The CLRWDT and SLEEP instructions clear the WDT and the prescaler, if assigned to the WDT, and prevent it from timing out and generating a device RESET.

The $\overline{TO}$ bit in the status register will be cleared upon a watchdog timer time-out.

10.6.2 WDT PROGRAMMING CONSIDERATIONS

It should also be taken into account that under worst-case conditions (minimum VDD, maximum temperature, maximum WDT prescaler) it may take several seconds before a WDT time-out occurs.

10.7 Power Management Options

PIC14000 has several power management options to prolong battery lifetime. The SLEEP instruction halts the CPU and turn off the on-chip oscillator. The CPU can be in SLEEP mode, yet the A/D converter can continue to run. Several bits are included in the SLPCON register (8Fh) to control power to analog modules, including the current bias source, charge controller, temperature sensor and A/D converter. For even lower power, a HIBERNATE mode which halts all on-chip activity, can be selected. A summary of the power management options is contained in Table 10-6.

TABLE 10-6: SUMMARY OF POWER MANAGEMENT OPTIONS

Function	Estimated Current*	Summary
CPU clock	Frequency dependent	OFF during SLEEP mode, ON otherwise.
Main Oscillator	Frequency dependent	ON if NOT in SLEEP mode. In SLEEP mode, controlled by OSCOFF bit, SLPCON<3>
Watchdog Timer	~ 5 μA	Controlled by WDTE, 2007h<2> and HIBEN, SLPCON <7>
Temperature sensor	100 μA	Controlled by TEMPOFF, SLPCON<1>
Low-voltage Detector	< 50 μA	Controlled by REFOFF, SLPCON<5>
Charge Controller/Current Flow Detector	50-100 μA	Controlled by CWUOFF, SLPCON<2>
A/D Comparator	100 μA	Controlled by ADOFF, SLPCON<0>
A/D Current DAC	Depends on ADDAC<3:0>. Anywhere from 0 - 40 μA.	Controlled by ADOFF, SLPCON<0>
Slope Reference Divider	~100 μA	Controlled by ADOFF, SLPCON<0>
Current Bias Network	~5 μA	Controlled by BIASOFF, SLPCON<4>.
A/D Capture Clock	< 500 μA at 4 MHz.	Automatically stops when overflow occurs. Requires ADRST to restart. Also controlled by ADOFF, SLPCON<0>
Bandgap Reference	~ 15 μA	Controlled by REFOFF, SLPCON< 5>
Bias Generator (for current sources)	< 5 μA	Controlled by REFOFF, SLPCON<5>
Voltage Regulator Control	Depends on external components.	Always ON. Does not consume power if unconnected.
Power On Reset	< 5 μA	Always ON

Note: All circuits except voltage regulator and Power On Reset are OFF in HIBERNATE mode.

* Estimated current is preliminary information only.

10.7.1 SLEEP MODE

The SLEEP mode is entered by executing a SLEEP instruction.

If SLEEP mode is enabled, the WDT will be cleared but keep running. The $\overline{PD}$ bit in the STATUS register is cleared, the $\overline{TO}$ bit is set, and on-chip oscillators are shut off, except the WDT RC oscillator, which continues to run. The I/O ports maintain the status they had before the SLEEP command was executed (driving high, low, or high-impedance). The IN or HS oscillator will continue to run if bit OSCOFF in the SLPCON register is cleared.

It is an option while in SLEEP mode to leave the on-chip oscillator running. This option allows an A/D conversion to continue while the CPU is in SLEEP mode. The CPU clocks are stopped in this condition to preserve power. The operation of the on-chip oscillator during SLEEP is controlled by the OSCOFF bit in the SLPCON <4>. Setting this bit to '1' allows the oscillator to continue to run. This bit is only active in SLEEP mode.

For lowest power consumption in this mode, all I/O pins should be either at V_{DD} or V_{SS} with no external circuitry drawing current from the I/O pin. I/O pins that are high-impedance inputs should be pulled high or low externally to avoid leakage currents caused by floating inputs. The $\overline{MCLR}$ pin must be at a logic high level (V_{IH}). The contribution from any on-chip pull-up resistors should be considered.

10.7.2 WAKE-UP FROM SLEEP

The PIC14000 can wake up from SLEEP through one of the following events:

1. External reset input on $\overline{MCLR}$ pin
2. Watchdog Timer time-out (if WDT is enabled)
3. Interrupt from OSC1/PBTN pin
4. RC<7:4> port change
5. TMR0 overflow.

The following peripheral interrupts can cause wake-up from SLEEP:

1. I^2C (serial port) start/stop bit detect interrupt.

2. Wake-up on current flow interrupt.

3. A/D conversion complete (capture event) interrupt.

4. A/D timer overflow interrupt.

Note: For an A/D interrupt to occur, bit OSCOFF must be cleared to allow the on-chip oscillator to continue to run during SLEEP mode. Similarly, in order to use the wake-up on current detect interrupt to exit SLEEP mode, bit CWUOFF must be cleared to keep the charge controller/current flow detector powered.

An external reset on $\overline{MCLR}$ pin causes a device reset. The other wake-up events are considered a continuation of program execution. The $\overline{TO}$ and $\overline{PD}$ bits in the STATUS register can be used to determine the cause of device reset. 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 a wake-up).

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. Wake-up occurs regardless of the state of bit GIE. If bit GIE is clear, the device continues execution at the instruction after the SLEEP instruction. If bit GIE is set, 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.

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

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

10.7.3 HIBERNATE MODE

HIBERNATE mode is identical to SLEEP mode with the exception that the WDT (Watchdog Timer) is forced off.

The HIBERNATE mode is entered by executing a SLEEP instruction with bit HIBEN in the SLPCON register set.

Exiting HIBERNATE mode occurs in the same way as exiting SLEEP mode, except that exiting due to a watchdog reset cannot occur.

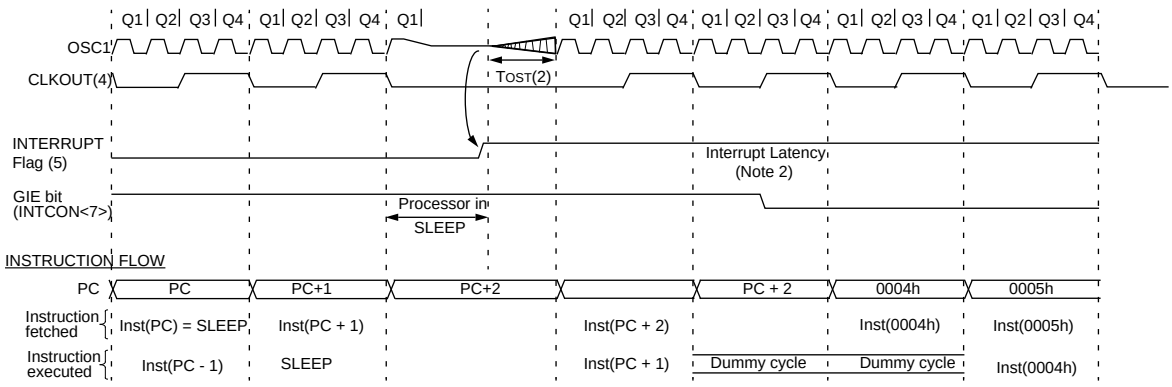
HIBERNATE mode allows power consumption to be reduced to a minimum (typically a few microamps) during periods of non-operation. This allows for reduced current consumption in battery operated systems that may undergo long storage periods (for example, in warehouse after manufacture). This mode may be desirable if a battery continues to be discharged below the end-of-discharge voltage, to minimize battery drain as much as possible, until a recharge cycle can be performed.

FIGURE 10-10: SLPCON REGISTER

8Fh	B7	B6	B5	B4	B3	B2	B1	B0
SLPCON	HIBEN	-	REFOFF	BIASOFF	OSCOFF	CWUOFF	TEMPOFF	ADOFF
Read/Write	R/W	U	R/W	R/W	R/W	R/W	R/W	R/W
POR value 1Fh	0	0	1	1	1	1	1	1

Bit	Name	Function
B7	HIBEN	Hibernate Mode Select 1 = Hibernate mode enable 0 = Normal operating mode
B6	-	Unimplemented. Read as '0'
B5	REFOFF	References Power Control 1 = The Bandgap reference, Low voltage detect, and bias generator is off. 0 = The Bandgap reference is on.
B4	BIASOFF	Summing Junction Current Bias Source Power Control 1 = The current bias source is off. The AN1/BATI input can continue to function as either an analog or digital input. 0 = The current bias source is ON. The signal at the AN1/BATI input is level shifted by approximately 0.5V.
B3	OSCOFF	Internal Oscillator ON/OFF bit during SLEEP mode. 1 = The internal IN or crystal oscillator is disabled during SLEEP mode. 0 = The internal IN or crystal oscillator is running during SLEEP mode for A/D conversions to continue. NOTE: This bit is effective only in SLEEP mode.
B2	CWUOFF	Charge Controller/Current Flow Detect Power Control 1 = The charge controller/current flow detector circuits are OFF. 0 = The charge controller/current flow detector circuits are ON.
B1	TEMPOFF	On-chip Temperature Sensor Power Control 1 = The temperature sensor is OFF. 0 = The temperature sensor is ON.
B0	ADOFF	A/D Module Power Control 1 = The A/D module power (comparator, current DAC, slope reference) is OFF 0 = The A/D module power is ON.

FIGURE 10-11: WAKE-UP FROM SLEEP THROUGH INTERRUPT



Note 1: HS oscillator mode assumed.
2: TOST = 1024 TOSC (drawing not to scale). This delay will be 8 TOSC for IN osc mode.
3: GIE = 1 assumed. In this case after wake up processor jumps to interrupt routine. If GIE = 0, execution will continue in line.
4: CLKOUT is not available in these osc modes, but shown here for timing reference.
5: Interrupt flag can be from any source. Refer to Section 10.5 for sources.

10.8 Code Protection

The code in the program memory can be protected by programming the code protect fuses. When code protected, the contents of the program memory cannot be read out. In code-protected mode, the configuration word (2007h) will not be scrambled, allowing reading of all configuration bits (fuse(s)).

10.8.1 CODE PROTECTION FUSES

Three separate code protect fuse bits are provided on PIC14000. They are CPP (program space), CPU (user space), and CPC (calibration space). The code protection fuses are part of the configuration fuse word at location 2007h as defined in Figure 10-12. The contents of all segments can be read. Protected user and program segments cannot be read or written (programmed). Protected calibration space can be read, but not written.

Note that address 2007h is beyond the user program memory space. It resides in the special test/configuration memory space (2000h - 3FFFh), which can be accessed only during programming. The code protection fuses cannot be erased, once programmed.

FIGURE 10-12: CONFIGURATION FUSE WORD

2007h	B13-B8	B7	B6	B5	B4	B3	B2	B1	B0
FUSES	CP<5:0>	CPC	N/A	CPU	CPP	PWRTE	WDTE	N/A	FOSC
Read/Write	R/W	R/W	Reserved	R/W	R/W	R/W	R/W	Reserved	R/W
Erased value	1	1	1	1	1	1	1	1	1

Bit	Name	Function
B13-B8	CP<5:0>	Alternate Code Protection Bits (Reserved).
B7	CPC	Calibration Space Code Protection Bit 1 = Calibration space is readable and programmable. 0 = Calibration space is write protected (fuse is programmed).
B6	N/A	Reserved
B5	CPU	User Space Code Protection Bit 1 = User space is readable and programmable. 0 = User space is read/write protected (bit is programmed).
B4	CPP	Program Space Code Protection Bit 1 = Program space is readable and programmable. 0 = Program space is read/write protected (bit is programmed).
B3	PWRTE	Power-up Timer Enable Bit. 0 = Power-up timer is enabled. 1 = Power-up timer is disabled (unprogrammed).
B2	WDTE	Watchdog Timer Enable Bit 1 = WDT is enabled. (unprogrammed) 0 = WDT is disabled.
B1	N/A	Reserved
B0	FOSC	Oscillator Selection Bit 0 = HS oscillator (crystal/resonator) 1 = IN oscillator (default unprogrammed)

10.9 In-Circuit Serial Programming

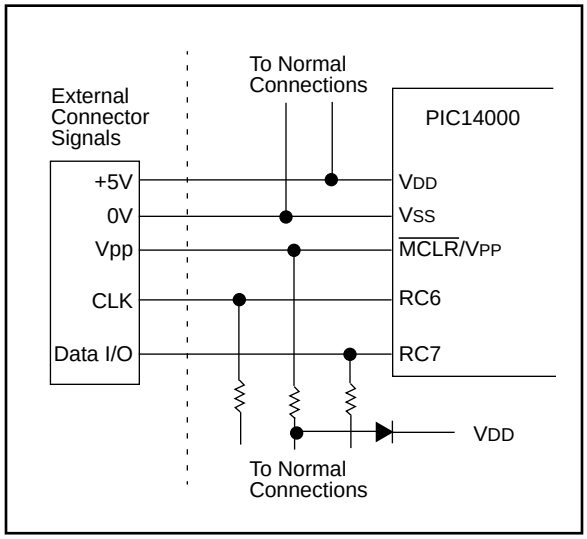
PIC14000 can be serially programmed while in the end application circuit. This is simply done with two lines for clock and data, and three other lines for power, ground and the programming voltage. This allows customers to manufacture boards with unprogrammed devices, and then program the microcontroller just before shipping the product. This allows the most recent firmware or a custom firmware to be programmed.

The device is placed into a program/verify mode by holding the RC6/SCL and RC7/SDA pins low while raising the $\overline{MCLR}$ (V_{PP}) pin from V_{IL} to V_{IH}. RC6 then becomes the programming clock and RC7 becomes the programmed data. Both RC6 and RC7 are Schmitt trigger inputs in this mode.

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

A typical in-system serial programming connection is shown in Figure 10-13.

FIGURE 10-13: TYPICAL IN-SYSTEM SERIAL PROGRAMMING CONNECTION



NOTES:

11.0 INSTRUCTION SET SUMMARY

The PIC14000's instruction set is the same as PIC16CXX. Each 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 instruction set summary in Table 11-2 lists byte-oriented, bit-oriented, and literal and control operations. Table 11-1 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 utilized by the instruction.

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

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

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

TABLE 11-1: OPCODE FIELD DESCRIPTIONS

Field	Description
f	Register file address (0x00 to 0x7F)
w	Working register (accumulator)
b	Bit address within an 8 bit file register
k	Literal field, constant data or label
x	Don't care location (= 0 or 1) The assembler will generate code with x = 0. It is the recommended form of use for compatibility with all software tools.
d	Destination select; d = 0: store result in W, d = 1: store result in file register f. Default is d = 1
label	Label name
TOS	Top of Stack
PC	Program Counter
PCLATH	Program Counter High Latch
GIE	Global Interrupt Enable Bit
WDT	Watchdog Timer Counter
TO	Time-out Bit
PD	Power-down Bit
dest	Destination either the W register or the specified register file location
[ ]	Options
( )	Contents
→	Assigned to
< >	Register bit field
∈	In the set of
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 11-2 lists the instructions recognized by the MPASM assembler.

Figure 11-1 shows the three general formats that the instructions can have.

Note: To maintain upward compatibility with future PIC16CXX 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 11-1: GENERAL FORMAT FOR INSTRUCTIONS

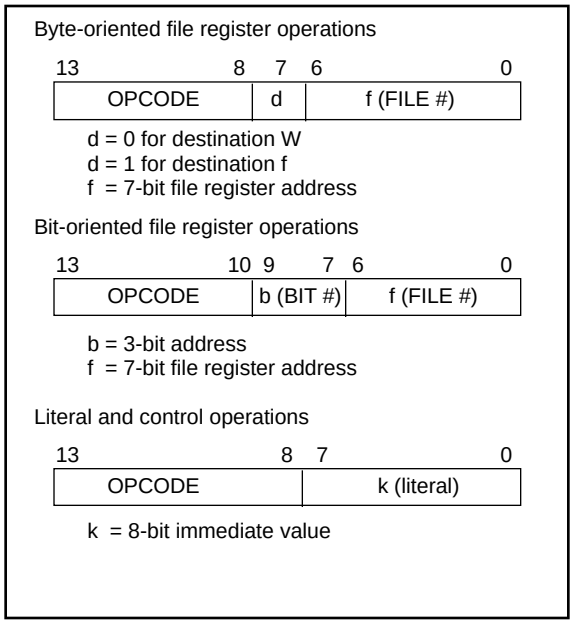


TABLE 11-2: PIC14000 INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	14-Bit Opcode msb			lsb	Status Affected	Notes
ADDWF		Add W and f	1					C,DC,Z	1,2
ANDWF	**f, d**	AND W and f	1	00	0111	dfff	ffff	Z	1,2
CLRF	**f, d**	Clear f	1	00	0101	dfff	ffff	Z	2
CLRW	**f**	Clear W	1	00	0001	lfff	ffff	Z	
COMF	**-**	Complement f	1	00	0001	0xxx	xxxx	Z	1,2
DECF	**f, d**	Decrement f	1	00	1001	dfff	ffff	Z	1,2
DECFSZ	**f, d**	Decrement f, Skip if 0	1(2)	00	0011	dfff	ffff		1,2,3
INCF	**f, d**	Increment f	1	00	1011	dfff	ffff	Z	1,2
INCFSZ	**f, d**	Increment f, Skip if 0	1(2)	00	1010	dfff	ffff		1,2,3
IORWF	**f, d**	Inclusive OR W and f	1	00	1111	dfff	ffff	Z	1,2
MOVF	**f, d**	Move f	1	00	0100	dfff	ffff	Z	1,2
MOVWF	**f**	Move W to f	1	00	1000	dfff	ffff	Z	1,2
NOP	**-**	No Operation	1	00	0000	lfff	ffff		
RLF	**f, d**	Rotate left through carry	1	00	0000	0xx0	0000		
RRF	**f, d**	Rotate right f through carry	1	00	1101	dfff	ffff	C	1,2
SUBWF	**f, d**	Subtract W from f	1	00	1100	dfff	ffff	C	1,2
SWAPF	**f, d**	Swap nibbles in f	1	00	0010	dfff	ffff	C,DC,Z	1,2
XORWF	**f, d**	Exclusive OR W and f	1	00	1110	dfff	ffff		1,2
								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 to W	1	11	111x	kkkk	kkkk	C,DC,Z	
ANDLW	**k**	AND literal to 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 to W	1	11	1000	kkkk	kkkk	Z	
MOVLW	**-**	Move literal to W	1	11	00xx	kkkk	kkkk		
RETFIE	**k**	Return from interrupt	2	00	0000	0000	1001		
RETLW	**-**	Return with literal in W	2	11	01xx	kkkk	kkkk		
RETURN	**-**	Return from subroutine	2	00	0000	0000	1000		
SLEEP	**k**	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		Excl. OR literal to W	1	11	1010	kkkk	kkkk	Z	

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

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

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.

11.1 Instruction Descriptions

ADDLW **Add Literal to 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 and 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 to 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 to 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, 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    BTFSC   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 **Subroutine Call**

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: Subroutine call. 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 Register**

Syntax: [label] CLRW

Operands: None

Operation: 00h → (W)
1 → Z

Status Affected: Z

Encoding:

00	0001	0XXX	XXXX

Description: W registered 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 prescale = 0
$\overline{TO}$ = 1
$\overline{PD}$ = 1

COMF **Complement f**

Syntax: [label] COMF f,d

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

Operation: $(\bar{f}) \rightarrow$ (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: $0 \le f \le 127$
$d \in [0,1]$

Operation: (f)-1 → (dest)

Status Affected: Z

Encoding:

00	0011	dfff	ffff

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

Words: 1

Cycles: 1

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

DECFSZ **Decrement f, skip if 0**

Syntax: [label] DECFSZ f,d

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

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

Status Affected: None

Encoding:

00	1011	dfff	ffff

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

Words: 1

Cycles: 1(2)

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

GOTO **Unconditional Branch**

Syntax: [label] GOTO k

Operands: $0 \le k \le 2047$

Operation: $k \rightarrow PC<10:0>$
$(PCLATH<4:3>) \rightarrow PC<12:11>$

Status Affected: None

Encoding:

10	1kkk	kkkk	kkkk

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

Words: 1

Cycles: 2

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

INCF **Increment f**

Syntax: [label] INCF f,d

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

Operation: (f) + 1 → (dest)

Status Affected: Z

Encoding:

00	1010	dfff	ffff

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

Words: 1

Cycles: 1

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

INCFSZ **Increment f, skip if 0**

Syntax: [label] INCFSZ f,d

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

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

Status Affected: None

Encoding:

00	1111	dfff	ffff

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

Words: 1

Cycles: 1(2)

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

IORLW **Inclusive OR Literal with W**

Syntax: [label] IORLW k

Operands: $0 \le k \le 255$

Operation: (W) .OR. (k) → (W)

Status Affected: Z

Encoding:

11	1000	kkkk	kkkk

Description: The contents of the W register 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

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) → (W)

Status Affected: $\overline{Z}$

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

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

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 is moved to destination 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

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
```

RETFIE — **Return from Interrupt**

Syntax: [label] RETFIE

Operands: None

Operation: TOS → PC,
1 → GIE;

Status Affected: None

Encoding:

00	0000	0000	1001

Description: Return from Interrupt. Stack is popped and Top of Stack (TOS) is loaded in the PC. Interrupts are enabled by setting the Global Interrupt Enable (GIE) bit. GIE is the global interrupt enable bit (INTCON<7>). This is a two cycle instruction.

Words: 1

Cycles: 2

Example

```
RETFIE
```

After Interrupt
PC = TOS
GIE = 1

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 PIC16CXX products, do not use this instruction.

RETLW — **Return Literal to W**

Syntax: [label] RETLW k

Operands: 0 ≤ k ≤ 255

Operation: k → W; TOS → PC;

Status Affected: None

Encoding:

11	01XX	kkkk	kkkk

Description: The W register is loaded with the eight bit literal 'k'. The program counter is loaded from the top of the stack (the return address). This is a two cycle instruction.

Words: 1

Cycles: 2

Example

```
CALL TABLE  ;W contains table
            ;offset value
•           ;W now has table value
•
•
TABLE ADDWF PC   ;W = offset
      RETLW k1   ;Begin table
      RETLW k2   ;
      •
      •
      •
      RETLW kn   ; End of table
```

Before Instruction
W = 0x07
After Instruction
W = value of k7

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

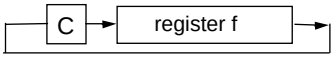
Operation: See description below

Status Affected: C

Encoding:

00	1100	dfff	ffff

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



Words: 1

Cycles: 1

Example

```
RRF          REG1,0
```

Before Instruction

```
REG1   =   11100110
C      =   0
```

After Instruction

```
REG1   =   11100110
W      =   01110011
C      =   1
```

RLF **Rotate Left f through Carry**

Syntax: [label] RLF f,d

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

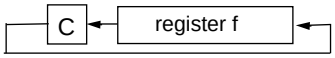
Operation: See description below

Status Affected: C

Encoding:

00	1101	dfff	ffff

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



Words: 1

Cycles: 1

Example

```
RLF          REG1,0
```

Before Instruction

```
REG1   =   11100110
C      =   0
```

After Instruction

```
REG1   =   11100110
W      =   11001100
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. See Section 10.7 for more details.

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 = FF
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 methodize 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 = FF
W = 2
C = 0; result is negative

SWAPF **Swap f**

Syntax: [label] SWAPF f,d

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

Operation: f<0:3> → d<4:7>,
f<4:7> → d<0:3>

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

```
SWAP  REG,   0
F
```

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 PIC16CXX 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

12.0 DEVELOPMENT SUPPORT

12.1 Development Tools

The PIC14000 are supported with a full range of hardware and software development tools:

- PICMASTER™ Real-Time In-Circuit Emulator
- PRO MATE™ Universal Programmer
- MPASM Assembler
- MPSIM Software Simulator
- C Compiler (MP-C)
- Fuzzy logic development system (fuzzyTECH®–MP)

12.2 PICMASTER: High Performance Universal In-Circuit Emulator

The PICMASTER Universal In-Circuit Emulator provides the product development engineer with a complete microcontroller design tool set for all microcontrollers in the PIC14000, PIC16C5X, PIC16CXX, and PIC17CXX families. A PICMASTER System configuration is shown in Figure 12-1.

Interchangeable target probes allow the system to be easily reconfigured for emulation of different processors. The universal architecture of the PICMASTER allows expansion to support all new PIC14000, PIC16C5X, PIC16CXX and PIC17CXX microcontrollers.

The Emulator System is designed to operate on PC compatible 386 and better machines. The development software runs in the Microsoft® Windows® 3.x environment, allowing the operator access to a wide range of supporting software and accessories.

The PICMASTER has been designed as a real-time emulation system with advanced features generally found on more expensive development tools. The AT platform and Windows 3.x environment was chosen to best make these features available to you, the end user.

The PICMASTER Emulator Universal System consists primarily of four major components:

- Host-Interface Card
- Emulator Control Pod
- Target-Specific Emulator Probe
- PC Host Emulation Control Software

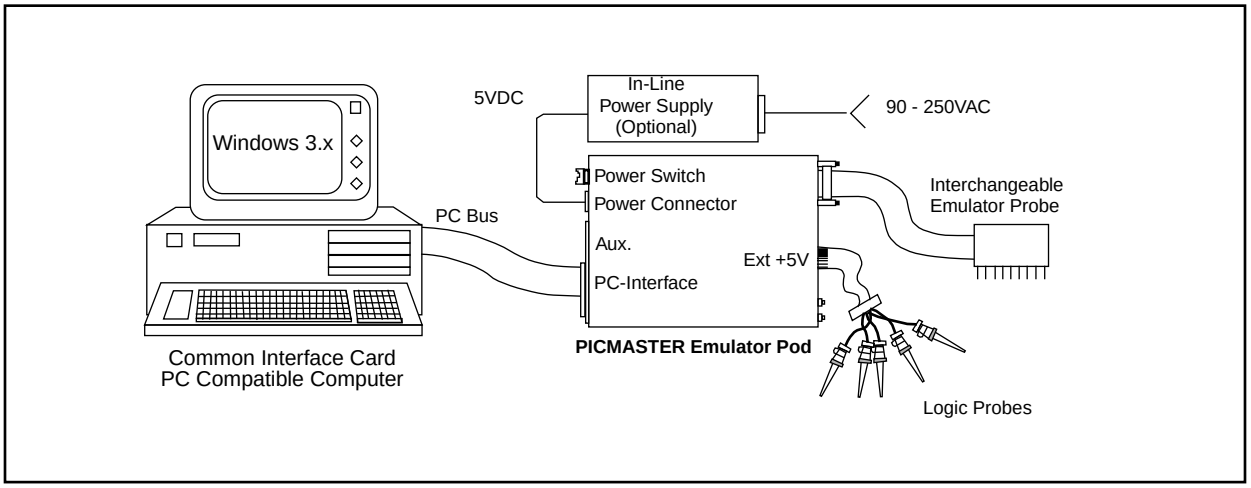
The Windows 3.x operating system allows the developer to take full advantage of the many powerful features and functions of the PICMASTER system.

PICMASTER emulation can operate in one window, while a text editor is running in a second window.

PC-Host emulation control software takes full advantage of Dynamic Data Exchange (DDE), a feature of Windows 3.x. DDE allows data to be dynamically transferred between two or more Windows programs. With this feature, data collected with PICMASTER can be automatically transferred to a spreadsheet or database program for further analysis.

Under Windows 3.x, two or more PICMASTER emulators can be run simultaneously on the same PC making development of multi-microcontroller systems possible (e.g., a system containing a PIC16CXX processor and a PIC17CXX processor).

FIGURE 12-1: PICMASTER SYSTEM CONFIGURATION



12.3 PRO MATE: Universal Programmer

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

The PRO MATE has programmable V_{DD} and V_{PP} supplies which allows it to verify programmed memory at V_{DD} min and V_{DD} max for maximum reliability. It has an LCD display for displaying error messages, keys to enter commands and a modular detachable socket assembly to support various package types. In stand-alone mode the PRO MATE can read, verify or program the PIC14000. It can also set fuse configuration and code-protect bits in this mode.

In PC-hosted mode, the PRO MATE connects to the PC via one of the COM (RS-232) ports. PC based user-interface software makes using the programmer simple and efficient. The user interface is full-screen and menu-based. Full screen display and editing of data, easy selection of fuse configuration and part type, easy selection of V_{DD} min, V_{DD} max and V_{PP} levels, load and store to and from disk files (Intel® hex format) are some of the features of the software. Essential commands such as read, verify, program and blank check can be issued from the screen. Additionally, serial programming support is possible where each part is programmed with a different serial number, sequential or random.

The PRO MATE has a modular “programming socket module.” Different socket modules are required for different processor types and/or package types.

12.4 Assembler (MPASM)

The MPASM Cross Assembler is a PC-hosted symbolic assembler. It supports all microcontroller series including the PIC14000, PIC16C5X, PIC16CXX, and PIC17CXX families.

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

MPASM allows full symbolic debugging from the Microchip Universal Emulator System (PICMASTER).

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

- Provides translation of Assembler source code to object code for all Microchip microcontrollers.
- Macro Assembly Capability
- Provides Object files, Listing files, Symbol files and special files required for debugging with one of the Microchip Emulator systems.
- Supports Hex (default), Decimal and Octal source and listing formats.

MPASM provides a full feature directive language represented by four basic classes of directives:

- **Data Directives** control the allocation of memory and provide a way to refer to data items symbolically, by meaningful names.
- **Listing Directives** control the MPASM listing display. They allow the specification of titles and sub-titles, page ejects and other listing control.
- **Control Directives** permit sections of conditionally assembled code.
- **Macro Directives** control the execution and data allocation within macro body definitions.

12.5 Software Simulator (MPSIM)

The MPSIM Software Simulator allows code development in a PC host environment. It allows the user to simulate the PIC16/17 series microcontrollers on an instruction level. On any given instruction, the user may examine or modify any of the data areas or provide external stimulus to any of the pins. The input/output radix can be set by the user and the execution can be performed in single step, execute until break or in a trace mode. MPSIM fully supports symbolic debugging using MP-C and MPASM. The Software Simulator offers the low cost flexibility to develop and debug code outside of the laboratory environment making it an excellent multi-project software development tool.

12.6 C Compiler (MP-C)

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

For easier source level debugging, the compiler provides symbol information that is compatible with the PICMASTER Universal Emulator memory display (emulator software versions 1.13 and later).

The MP-C Code Development System is supplied directly by Byte Craft Limited of Waterloo, Ontario, Canada. If you have any questions, please contact your regional Microchip FAE (see Worldwide Sales & Service at the end of data sheet), or Microchip technical support personnel at (602) 786-7627.

12.7 Fuzzy Logic Development System (*fuzzy*TECH-MP)

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

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

12.8 Development Systems

For convenience, the development tools are packaged into comprehensive systems as listed in Table 12-1.

TABLE 12-1: DEVELOPMENT SYSTEM PACKAGES

Item	Name	System Description
1.	PICMASTER System	PICMASTER In-Circuit Emulator with PRO MATE Programmer, Assembler, Software Simulator, Samples, and your choice of Target Probe,
2.	PRO MATE System	PRO MATE Universal Programmer, full featured stand-alone or PC-hosted programmer, Assembler, Simulator

NOTES:

13.0 ELECTRICAL CHARACTERISTICS FOR PIC14000
ABSOLUTE MAXIMUM RATINGS †

Ambient temperature under bias ..... -55°C to+ 125°C

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

Voltage on any pin with respect to V_{SS} (except V_{DD} and $\overline{MCLR}$) ..... -0.5V to V_{DD} +0.6V

Voltage on V_{DD} with respect to V_{SS} ..... 0 to +6.0 V

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

Total power Dissipation (Note 1) ..... 1.0 W

Maximum Current out of V_{SS} pin 300mA

Maximum Current into V_{DD} pin 250mA

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

Output clamp current, I_{OK} (V0 <0 or V0>V_{DD}) ..... ±20mA

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

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

Maximum Current sunk by PORTA, PORTC, and PORTD(combined) 200mA

Maximum Current sourced by PORTA, PORTC, and PORTE (combined) 200mA

Maximum Current sunk by PORTC and PORTD (combined) 200mA

Maximum Current sourced by PORTC and PORTD (combined) 200mA

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

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

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

13.1 DC Characteristics: PIC14000

DC CHARACTERISTICS						**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ + 125°C for automotive, -40°C ≤ TA ≤ + 85°C for industrial and 0°C ≤ TA ≤ +70°C for commercial Operating voltage VDD = 2.7V to 6.0V
Characteristic	**Sym**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
Supply Voltage	VDD	2.7	-	6.0	V	IN osc configuration
		4.5	-	5.5	V	HS osc configuration
RAM Data Retention Voltage (Note 1)	VDR	-	1.5	-	V	Device in SLEEP mode
VDD start voltage to guarantee Power-On Reset	VPOR	-	VSS	-	V	See section on power-on reset for details
VDD rise rate to guarantee Power-On Reset	SVDD	0.05*	-	-	V/ms	See section on power-on reset for details
Supply Current (Note 2, 5)	IDD	-	TBD	TBD	mA	IN osc configuration FOSC = 4 MHz, VDD = 5.0V (Note 4)
			TBD	TBD	mA	FOSC = 4MHz, VDD = 3.0V
		-	TBD	TBD	mA	HS osc configuration
						FOSC = 20 MHz, VDD = 5.5V
Power Down Current (Note 3, 5)	IPD	-	TBD	TBD	µA	VDD=4.0V, WDT enabled,-40°C to +85°C
		-	TBD	TBD	µA	VDD=4.0V, WDT disabled,-0°C to +70°C
		-	TBD	TBD	µA	VDD=4.0V, WDT disabled,-40°C to +85°C
		-	TBD	TBD	µA	VDD=4.0V, WDT disabled,-40°C to +125°C

* These parameters are characterized but not tested.

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

Note 1: This is the limit to which VDD can be lowered in SLEEP mode without losing RAM data.

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

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

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

5: Timer1 oscillator (when enabled) adds approximately 20 µA to the specification. This value is from characterization and is for design guidance only. This is not tested.

13.2 DC Characteristics: PIC14000

DC CHARACTERISTICS						**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ + 125°C for automotive, -40°C ≤ TA ≤ + 85°C for industrial and 0°C ≤ TA ≤ +70°C for commercial Operating voltage VDD range as described in DC specifications and Table 13-1
Characteristic	**Sym**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
Input Low Voltage						
I/O ports	VIL					
with Schmitt Trigger buffer		VSS	-	0.2VDD	V	
$\overline{MCLR}$, OSC1 (in IN mode)		Vss	-	0.2VDD	V	
OSC1 (in HS)		Vss	-	0.3VDD	V	Note1
Input High Voltage						
I/O ports	VIH		-			
with Schmitt Trigger buffer		0.85 VDD	-	VDD	V	For entire VDD range
PORTC weak pull-up current	IPURB	50	200	†400	µA	VDD = 5V, VPIN = VSS
Input Leakage Current (Notes 2,3)						
I/O ports	IIL			±1	µA	Vss ≤ VPIN ≤ VDD, Pin at hi-impedance
$\overline{MCLR}$				±5	µA	Vss ≤ VPIN ≤ VDD
OSC1				±5	µA	Vss ≤ VPIN ≤ VDD
Output Low Voltage						
I/O ports	VOL	-	-	0.6	V	IOL = 8.5mA, VDD-4.5V, -40°C to +85°C
CDAC		-		TBD	V	TBD
OSC2/CLKOUT (HS osc configuration)		-	-	0.6	V	IOL = 1.6mA, VDD-4.5V, -40°C to +85°C
IN OSC		-	-	TBD	V	IOL = 1.6mA, VDD-4.5V, -40°C to +85°C
Output High Voltage						
I/O ports (Note 3)	VOH	VDD-0.7	-	-	V	IOH = -3.0mA, VDD=4.5V, -40°C to +85°C
RC6, RC7, RD0		TBD	-	-	V	IOH = -3.0mA, VDD=4.5V, -40°C to +85°C
CDAC		TBD	-	-	V	TBD
OSC2/CLKOUT (HS osc configuration)		VDD-0.7	-	-	V	IOH = -1.3mA, VDD=4.5V, -40°C to +85°C
IN OSC		TBD	-	-	V	IOH = -1.3mA, VDD=4.5V, -40°C to +85°C
Capacitive Loading Specs on Output Pins						
OSC2 pin	COSC2			15	pF	In HS mode when external clock is used to drive OSC1.
All I/O pins and OSC2 (in IN mode)	CIO			50	pF	
SCL, SDA in I^2C mode	Cb			400	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 IN oscillator configuration, the OSC1 pin is a Schmitt trigger input.

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. All I/O ports except RC6, RC7, RD0.

4: The user may use better of the two specs.

13.3 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 subscripts (pp) and their meanings:

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

Uppercase letters and their meanings:

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

13.4 Timing Diagrams and Specifications

FIGURE 13-1: EXTERNAL CLOCK TIMING

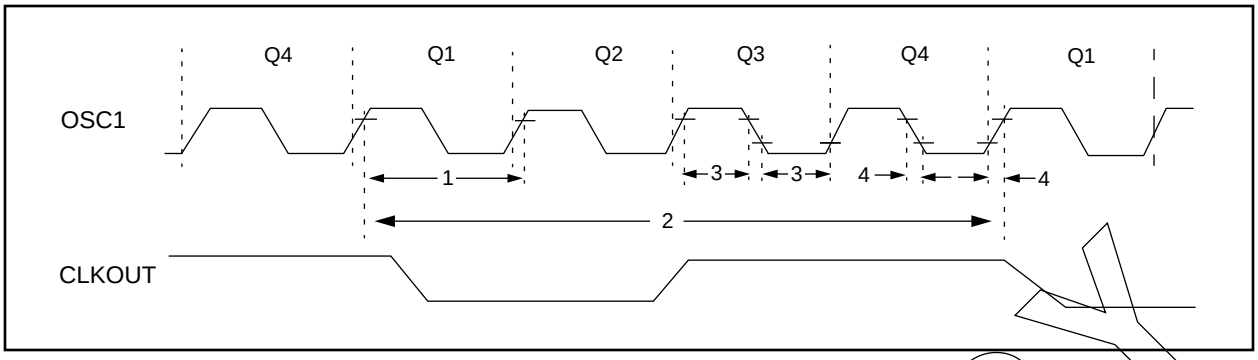


TABLE 13-1: EXTERNAL CLOCK TIMING REQUIREMENTS

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
	FOSC	External CLKIN Frequency (Note 1)	DC	—	4	MHz	HS osc mode (PIC14000-04)
			DC	—	20	MHz	HS osc mode (PIC14000-20)
		Oscillator Frequency (Note 1)	4	—	4	MHz	HS osc mode(PIC14000-04)
			4	—	10	MHz	HS osc mode (PIC14000-10)
			4	—	20	MHz	HS osc mode (PIC14000-20)
1	TOSC	External CLKIN Period (Note 1)	250	—	—	ns	HS osc mode (PIC14000-04)
			100	—	—	ns	HS osc mode (PIC14000-10)
			50	—	—	ns	HS osc mode (PIC14000-20)
		Oscillator Period (Note 1)	250	—	250	ns	HS osc mode (PIC14000-04)
			100	—	250	ns	HS osc mode (PIC14000-10)
			50	—	250	ns	HS osc mode (PIC14000-20)
2	TCY	Instruction Cycle Time (Note 1)	200	—	DC	ns	TCY = 4/FOSC
3	TOSL, TOSH	Clock in (OSC1) High or Low Time	10	—	—	ns	HS oscillator
4	TOSR, TOSF	Clock in (OSC1) Rise or Fall Time	—	—	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 pin.
When an external clock input is used, the "Max." cycle time limit is "DC" (no clock) for all devices.

FIGURE 13-2: LOAD CONDITIONS

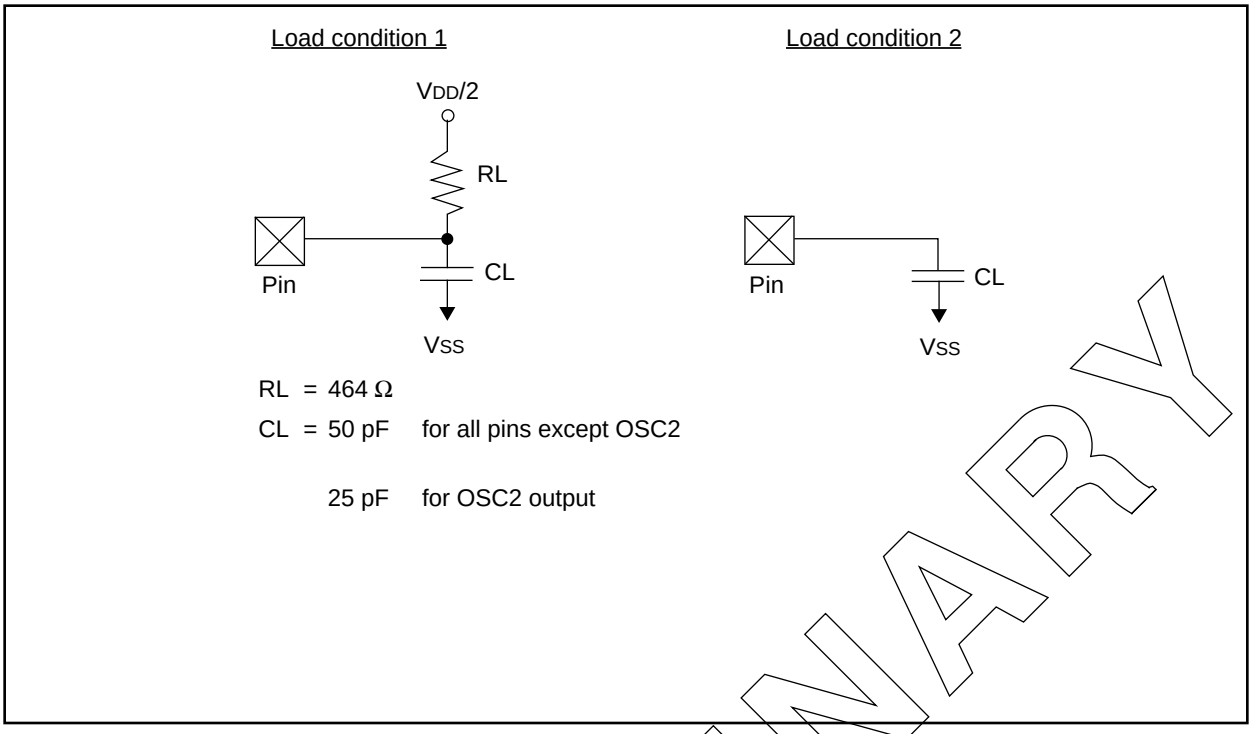
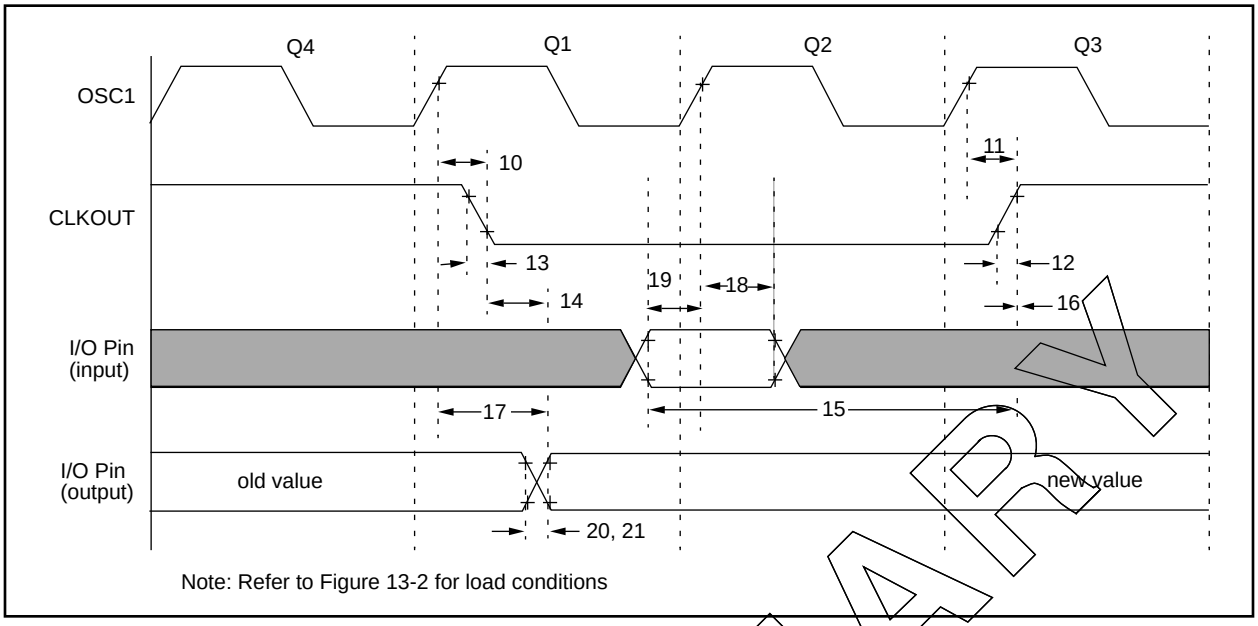


FIGURE 13-3: CLKOUT AND I/O TIMING



Note: Refer to Figure 13-2 for load conditions

TABLE 13-2: CLKOUT AND I/O TIMING REQUIREMENTS

Parameter No.	Sym	Characteristic	Min	Typ †	Max	Units	Conditions
10	TosH2ckL	OSC1↑ to CLKOUT↓	—	15	30	ns	Note 1
11	TosH2ckH	OSC1↑ to CLKOUT↑	—	15	30	ns	Note 1
12	TckR	CLKOUT rise time	—	5	15	ns	Note 1
13	TckF	CLKOUT fall time	—	5	15	ns	Note 1
14	TckL2ioV	CLKOUT ↓ to Port out valid	—	—	0.5TCY+20	ns	Note 1
15	TioV2ckH	Port in valid before CLKOUT ↑	0.25 TCY+25	—	—	ns	Note 1
16	TckH2ioI	Port in hold after CLKOUT ↑	0	—	—	ns	Note 1
17	TosH2ioV	OSC1↑ (Q1 cycle) to Port out valid	—	—	80 - 100	ns	
18	TosH2ioI	OSC1↑ (Q2 cycle) to Port input invalid (I/O in hold time)	TBD	—	—	ns	
19	TioV2osH	Port input valid to OSC1↑ (I/O in setup time)	TBD	—	—	ns	
20	TioR	Port output rise time	—	10	25	ns	
21	TioF	Port output fall time	—	10	25	ns	
22††	Tinp	PBTN pin high or low time	20	—	—	ns	IN mode only
23††	Trbp	RC<7:4> change INT high or low time	20	—	—	ns	

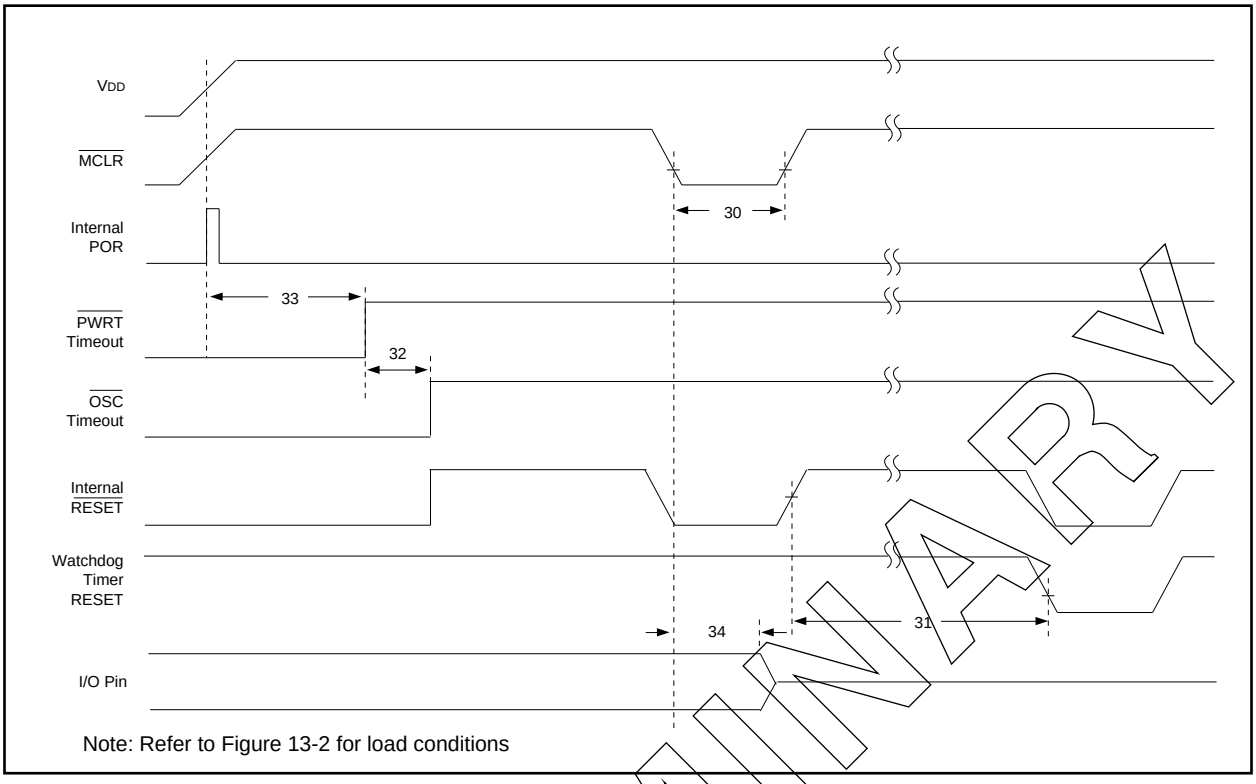
* These parameters are characterized but not tested.

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

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

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

FIGURE 13-4: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER (HS MODE) AND POWER-UP TIMER TIMING



Note: Refer to Figure 13-2 for load conditions

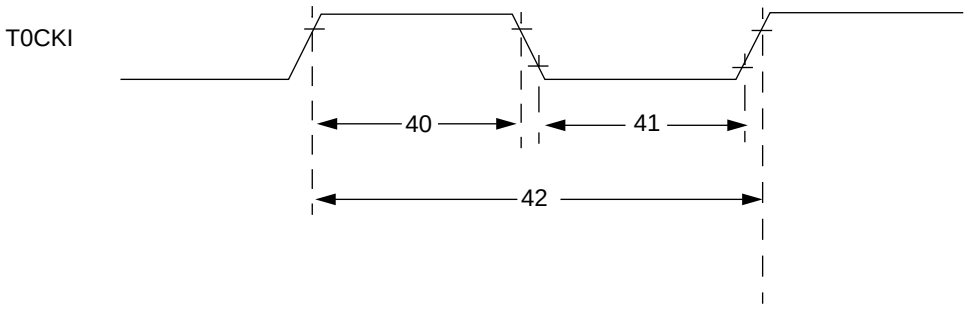
TABLE 13-3: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER REQUIREMENTS

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
30	TmcL	MCLR Pulse Width (low)	100	—	—	ns	VDD = 5V, -40°C to +125°C
31	Twdt	Watchdog Timer Timeout Period (No Prescaler)	7*	18	33*	ms	VDD = 5V, -40°C to +125°C
32	Tost	Oscillation Start-up Timer Period		1024 TOSC 8 TOSC		ms ms	TOSC = OSC1 period IN osc mode
33	Tpwrt	Power up Timer Period	28*	72	132*	ms	VDD = 5V, -40°C to +125°C
34	TIOZ	I/O High Impedance from MCLR Low			100	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.

FIGURE 13-5: TIMER0 CLOCK TIMINGS



Note: Refer to Figure 13-2 for load conditions.

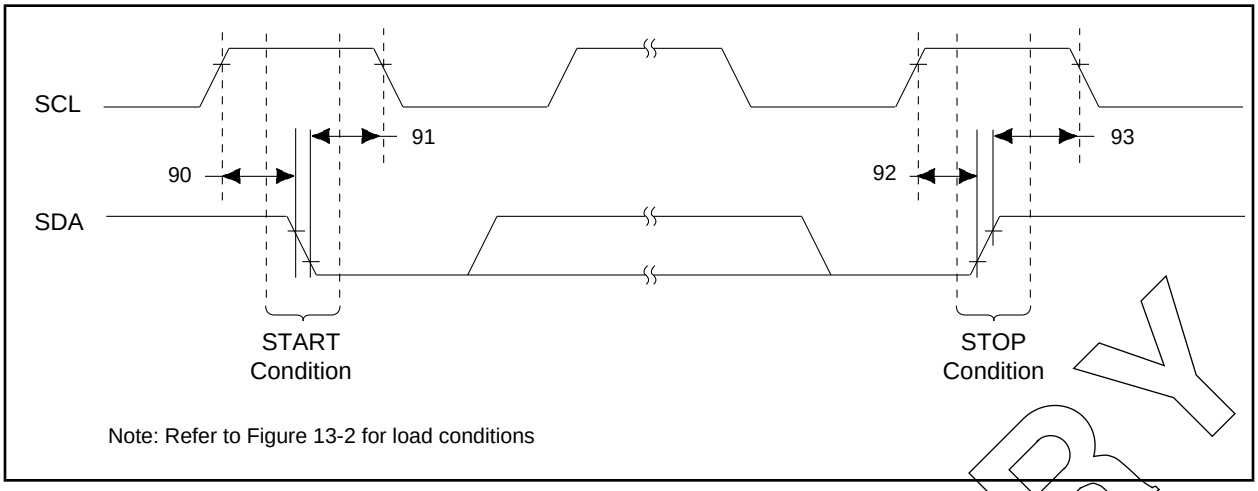
TABLE 13-4: TIMER0 CLOCK REQUIREMENTS

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

* These parameters are characterized but not tested.

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

FIGURE 13-6: I²C BUS START/STOP BITS TIMING

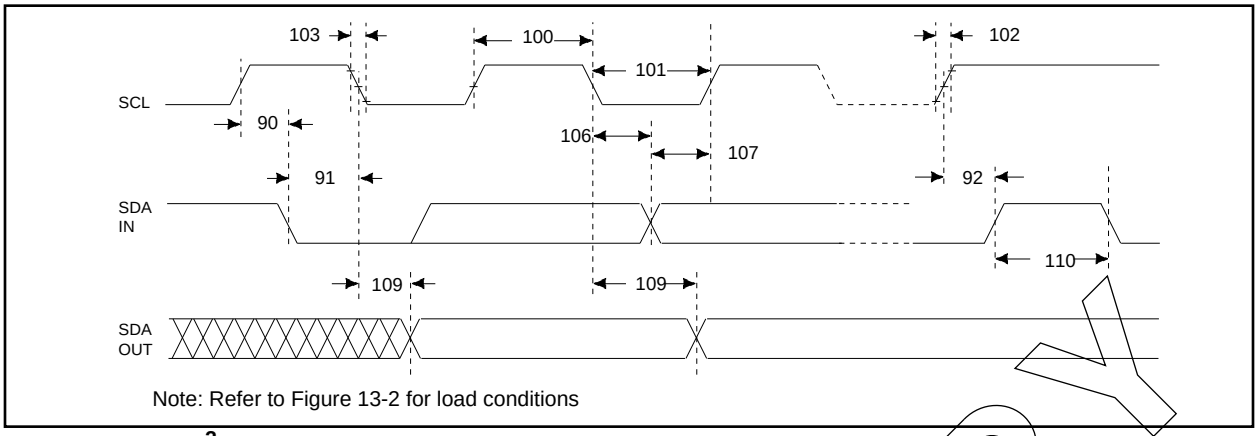


Note: Refer to Figure 13-2 for load conditions

TABLE 13-5: I²C BUS START/STOP BITS REQUIREMENTS

Parameter No.	Sym	Characteristic		Min	Typ	Max	Units	Conditions
90	TSU:STA	START condition	100 KHZ mode	4700	—	—	ns	Only relevant for repeated START condition
		Setup time	400 KHz mode	600	—	—		
91	THD:STA	START condition	100 KHz mode	4000	—	—	ns	After this period the first clock pulse is generated
		Hold time	400 KHz mode	600	—	—		
92	TSU:STO	STOP condition	100 KHZ mode	4700	—	—	ns	
		Setup time	400 KHz mode	600	—	—		
93	THD:STO	STOP condition	100 KHz mode	4000	—	—	ns	
		Hold time	400 KHz mode	600	—	—		

FIGURE 13-7: I²C BUS DATA TIMING



Note: Refer to Figure 13-2 for load conditions

TABLE 13-8: I²C BUS DATA REQUIREMENTS

Parameter No.	Sym	Characteristic		Min	Max	Units	Conditions
100	THIGH	Clock high time	100 kHz mode	4.0	—	µs	PIC14000 must operate at a minimum of 1.5 MHz
			400 kHz mode	0.6	—	µs	PIC14000 must operate at a minimum of 10 MHz
			I²C Module	1.5 TCY	—		
101	TLOW	Clock low time	100 kHz mode	4.7	—	µs	PIC14000 must operate at a minimum of 1.5 MHz
			400 kHz mode	1.3	—	µs	PIC14000 must operate at a minimum of 10 MHz
			I²C Module	1.5 TCY	—		
102	TR	SDA and SCL rise time	100 kHz mode	—	1000	ns	
			400 kHz mode	20+0.1 C_b	300	ns	C_b is specified to be from 10-400 pF
103	TF	SDA and SCL fall time	100 kHz mode	—	300	ns	
			400 kHz mode	20+0.1 C_b	300	ns	C_b is specified to be from 10-400 pF
90	TSU:STA	START condition setup time	100 kHz mode	4.7	—	µs	Only relevant for repeated START condition
			400 kHz mode	0.6	—	µs	
91	THD:STA	START condition hold time	100 kHz mode	4.0	—	µs	After this period the first clock pulse is generated
			400 kHz mode	0.6	—	µs	
106	THD:DAT	Data input hold time	100 kHz mode	0	—	ns	
			400 kHz mode	0	0.9	µs	
107	TSU:DAT	Data input setup time	100 kHz mode	250	—	ns	Note 2
			400 kHz mode	100	—	ns	
92	TSU:STO	STOP condition setup time	100 kHz mode	4.7	—	µs	
			400 kHz mode	0.6	—	µs	
109	TAA	Output valid from clock	100 kHz mode	—	3500	ns	Note 1
			400 kHz mode	—	—	ns	
110	TBUF	Bus free time	100 kHz mode	4.7	—	µs	Time the bus must be free before a new transmission can start
			400 kHz mode	1.3	—	µs	
	C_b	Bus capacitive loading		—	400	pF	

Note 1: As a transmitter, the device must provide this internal minimum delay time to bridge the undefined region (min. 300 ns) of the falling edge of SCL to avoid unintended generation of STARTs or STOPs.

2: A fast-mode I²C-bus device can be used in a standard-mode I²C-bus system, but the requirement tSU:DAT≥250ns must then be met. This will automatically be the case if the device does not stretch the LOW period of the SCL signal. If such a device does stretch the LOW period of the SCL signal, it must output the next data bit to the SDA line TR max.+tSU:DAT=1000+250=1250 ns (according to the standard-mode I²C bus specification) before the SCL line is released.

14.0 ANALOG SPECIFICATIONS

The following parameters will be provided on the following analog peripherals.

- Comparator
- 4-bit current DAC
- Bandgap voltage reference
- Digital-to-analog converter
- Thermistor
- Slope reference

High accuracy in analog-to-digital conversion can be achieved through proper component and current DAC selection. Please refer to the Section 8.0 "Analog Modules for A/D Conversion" for information on the analog input channels.

15.0 DC AND AC CHARACTERISTICS GRAPHS AND TABLES FOR PIC14000

FIGURE 15-1: TYPICAL I_{PD} VS V_{DD} WATCHDOG TIMER DISABLED 25°C

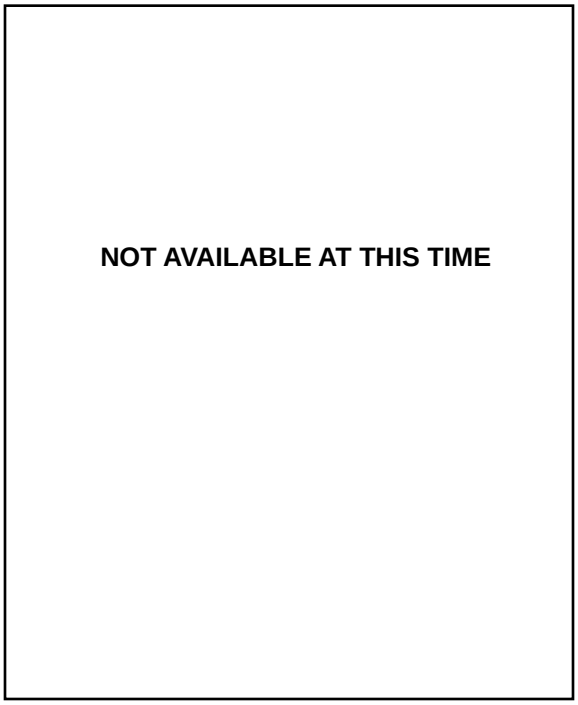


FIGURE 15-2: TYPICAL I_{PD} VS V_{DD} WATCHDOG TIMER ENABLED 25°C

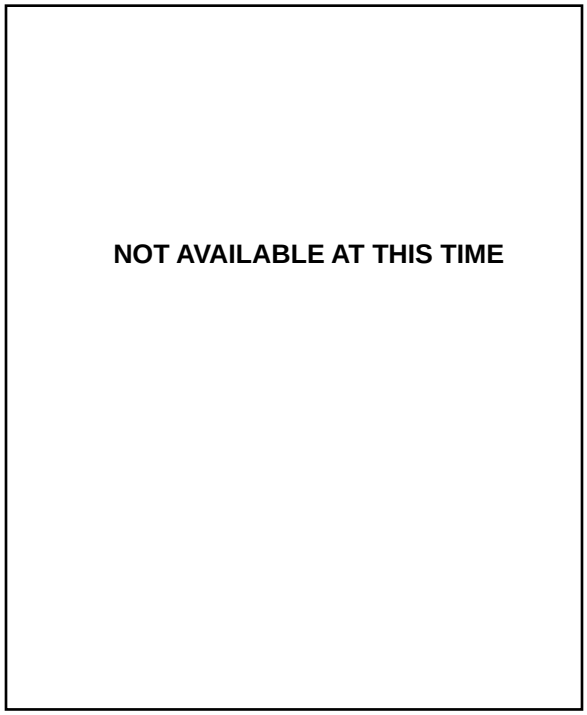


FIGURE 15-3: V_{TH} (INPUT THRESHOLD VOLTAGE) OF I/O PINS vs V_{DD}

NOT AVAILABLE AT THIS TIME

FIGURE 15-4: VTH (INPUT THRESHOLD VOLTAGE) OF OSC1 INPUT (IN HS MODE) vs V_{DD}

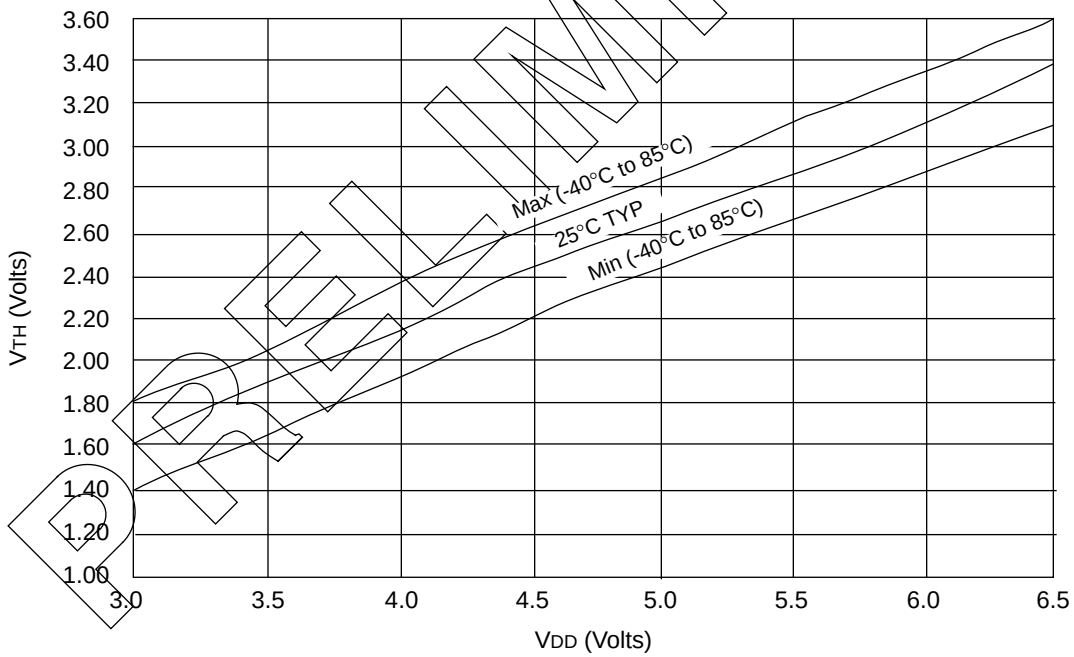


FIGURE 15-5: TYPICAL IDD VS FREQ (EXT CLOCK, 25°C)

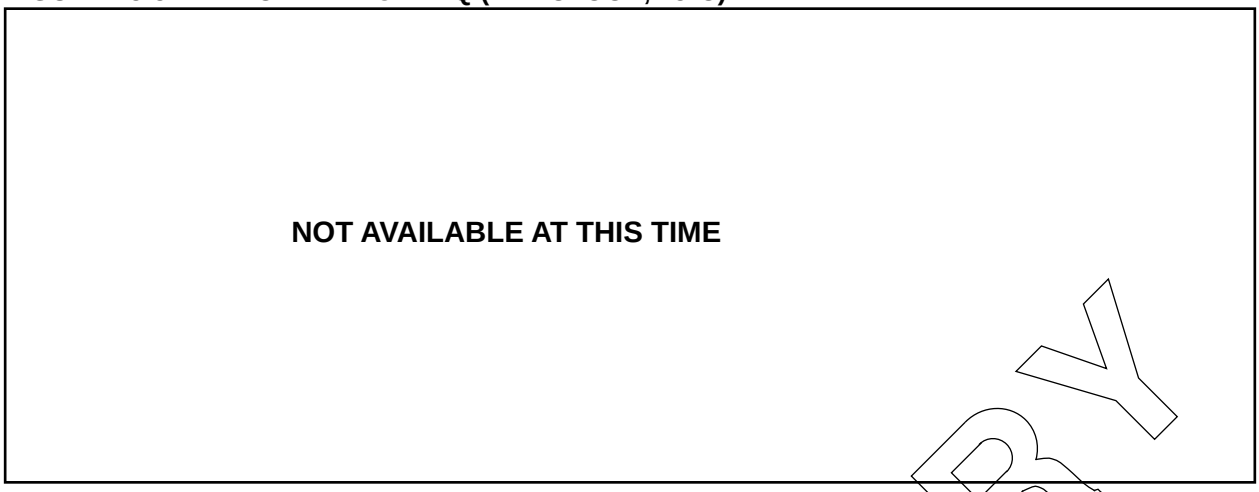


FIGURE 15-6: MAXIMUM, IDD VS FREQ (EXT CLOCK, -40° TO +85°C)

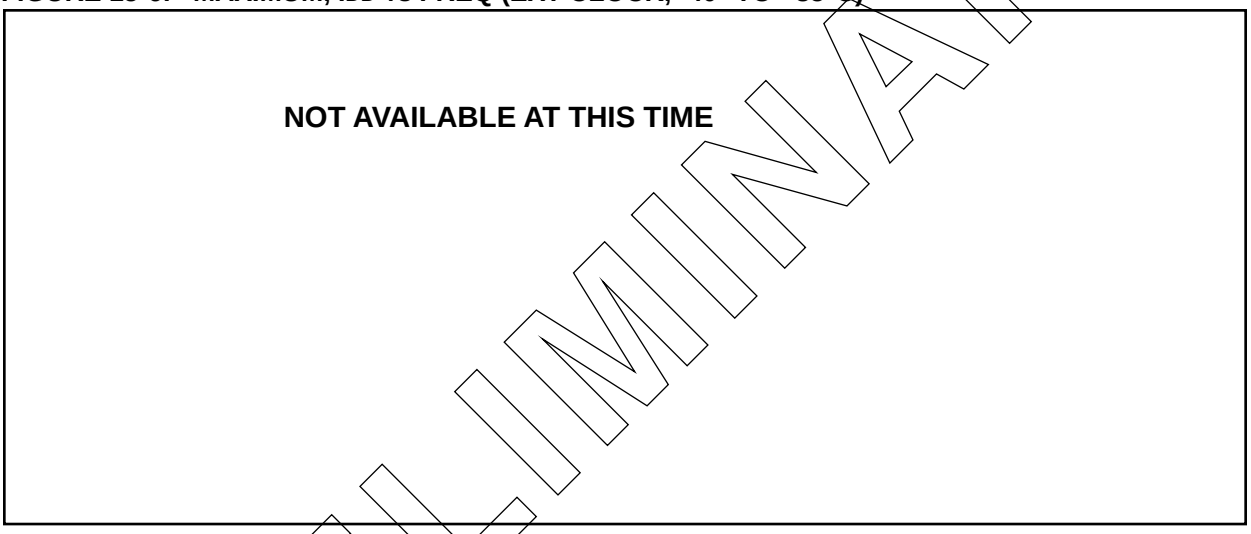


FIGURE 15-7: MAXIMUM IDD VS FREQ WITH A/D OFF (EXT CLOCK, -55° TO +125°C)

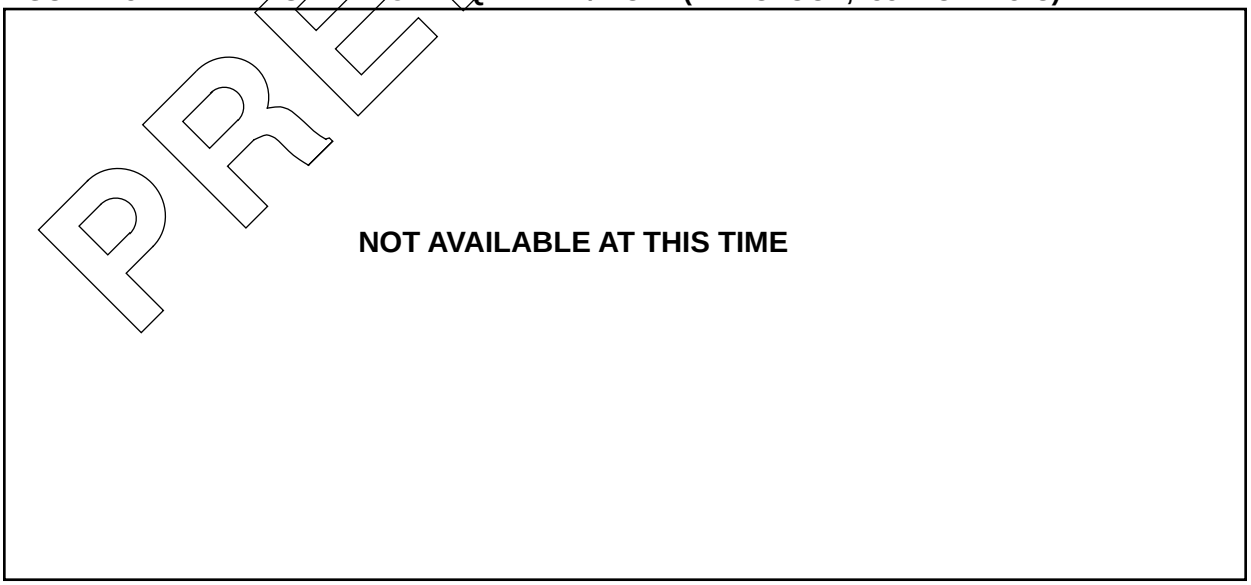


FIGURE 15-8: WDT TIMER TIME-OUT PERIOD VS VDD

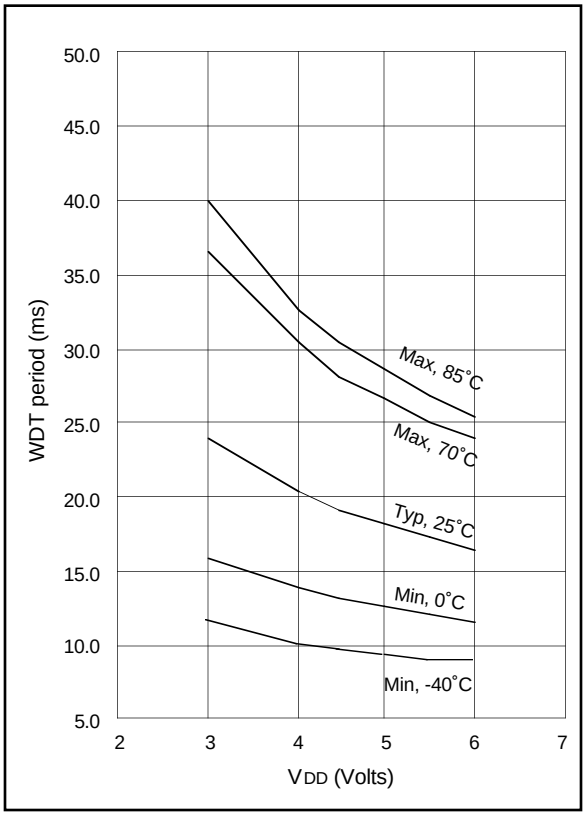


FIGURE 15-9: TRANSCONDUCTANCE (GM) OF HS OSCILLATOR VS VDD

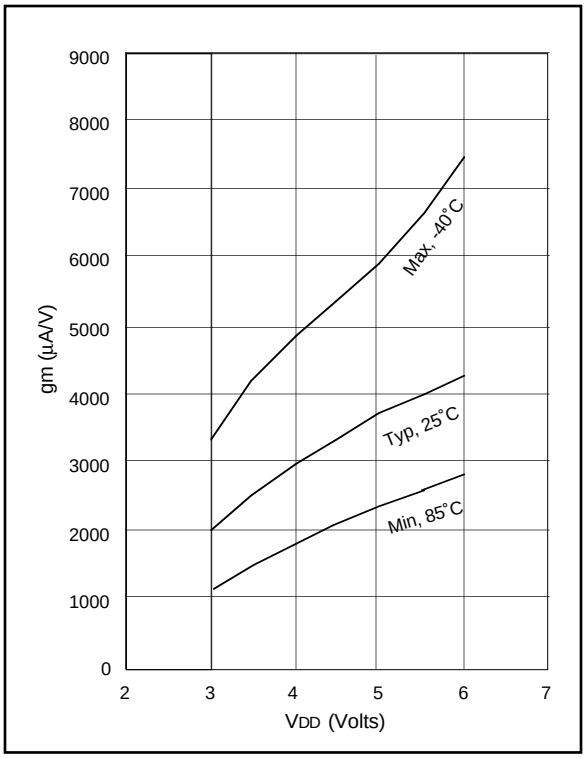


FIGURE 15-10: IOH VS VOH, VDD = 3V*

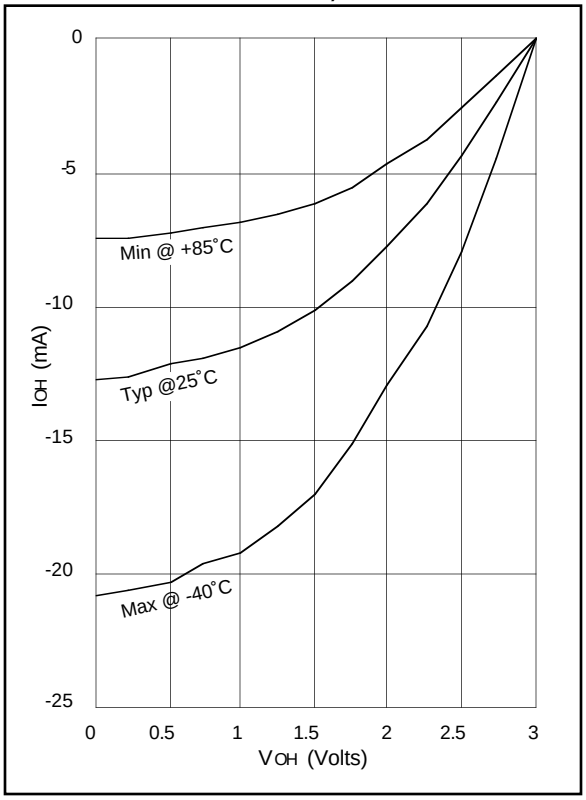
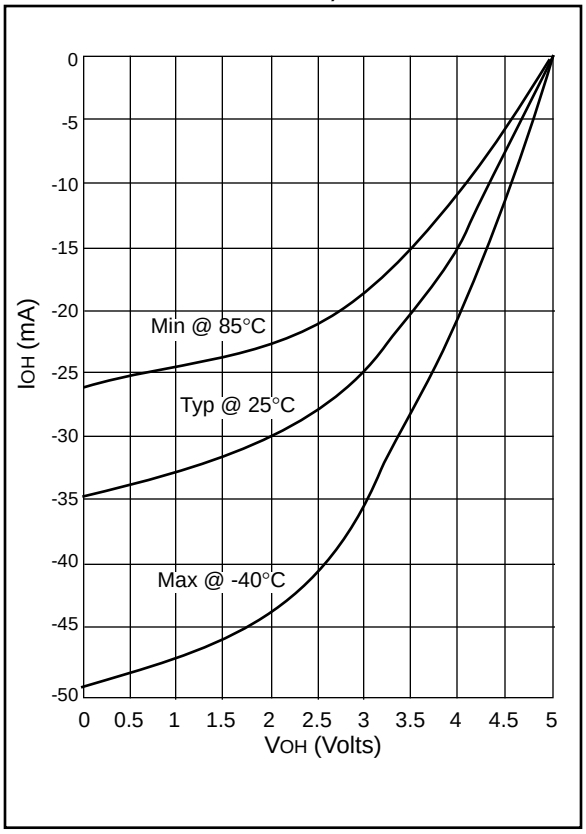


FIGURE 15-11: IOH VS VOH, VDD = 5V*



***NOTE:** All pins except RC6, RC7, RD0, RD1,OSC2

FIGURE 15-12: IOL VS VOL, VDD = 3V*

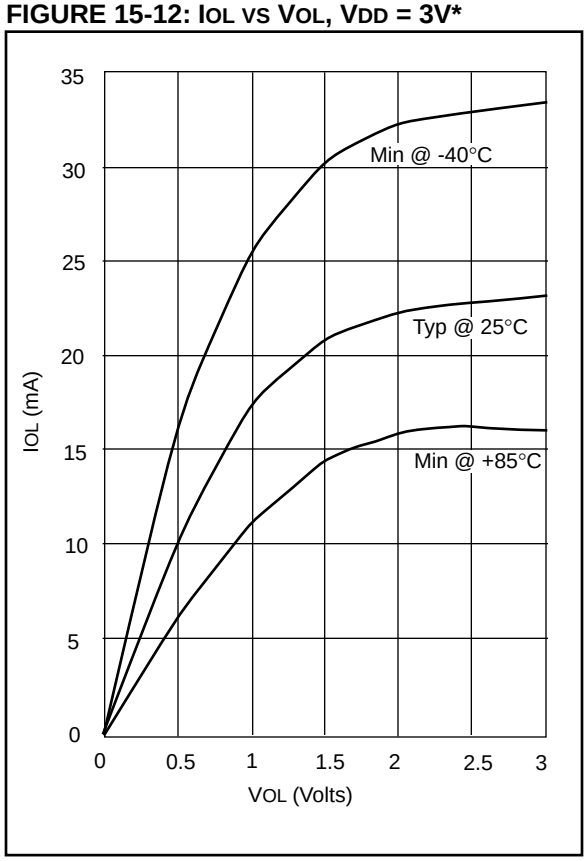
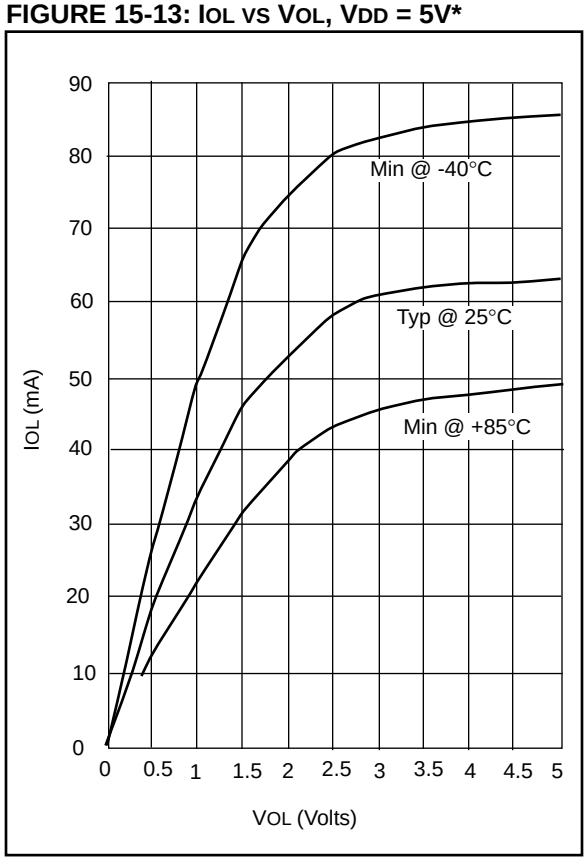


FIGURE 15-13: IOL VS VOL, VDD = 5V*



***NOTE:** All pins except OSC2

NOTES:

APPENDIX A: DIFFERENCES BETWEEN PIC14000 AND PIC16C74

The following is a list of differences between the PIC14000 and the PIC16C74.

Package Size

- The PIC14000 is a 28-pin device while the PIC16C74 is in 40- and 44-pin packages. The PIC14000 includes only Ports A, C and D. PORTB is used by the PIC16C02 emulation master device. This address is reserved for emulation use.

Clock Oscillator

- PIC14000 provides support for HS and IN oscillation options. The IN oscillator option on the PIC14000 uses internal RC components with a fixed nominal frequency of 4 MHz.

Peripherals

- PIC14000 does not include the USART, CCP register, Timer1 and Timer2. The registers associated with these modules are eliminated.
- PIC14000 supports the I^2C protocol only. SPI protocol is not supported. Registers associated with SPI protocol exist but are not used. The module should ONLY be programmed as an I^2C interface.
- The I^2C interface pins are located on PORTC, bits 7 and 6. These pins are also used as the serial programming interface.
- The slope A/D converter on the PIC14000 is completely different from the successive approximation A/D on the PIC16C74. Refer to the A/D Converter section for additional details.
- The PIC14000 includes a bandgap voltage reference, low-voltage detector and other analog modules not found on the PIC16C74. Refer to Section 8.0 for a complete list of analog modules on the PIC14000.
- PIC14000 provides hardware support for controlling two external switching regulators for battery charging. A logarithmic DAC and comparator is required for each socket. This circuit can also be used as a current flow detector to exit from SLEEP mode.
- The PORTB interrupt-on-change has been relocated to RC<7:4>.
- The external interrupt feature has been moved from RB0 to the OSC1/PBTN pin. This interrupt is available only in IN mode.
- Timer0 external clock option (T0CKI) is moved to RC3 pin.
- PORTC electrical characteristics are different from the PIC16C74. Refer to Section 13.0 of the PIC14000 data sheet and Section 18.0 in the PIC16C74 data sheet for details.

Program and Data Memory

- The PIC14000 program memory space has been redefined from the PIC16C74. The 4K of implemented memory has been divided into a program space and a 512 word user/calibration space. Code protect can be enabled for each space independently.

Power Management

- SLEEP/power-down modes have been enhanced on the PIC14000. Control bits have been added to the SLPCON register (1Fh) to enable power down of various analog modules. A new mode "Hibernate" has been added. Refer to Section 10.7 for details.
- HIBERNATE mode disables the Watchdog Timer allowing for software to disable the WDT. Please refer to the section on the Watchdog Timer for additional details.

Differences between PIC14000 and PIC16C74 in the Register File Map

This section details the differences in the register file map between the PIC14000 and the PIC16C74. Porting any code developed on the PIC16C74 to the PIC14000 must take these differences into account. Refer to Figure 4-2 for names of the specific PIC14000 registers. In some cases the PIC14000 register names are different from the PIC16C74 registers. Note that all memory locations specified at UNIMPLEMENTED are read as 0's. The differences between the PIC16C74 and PIC14000 register files are:

File Address	PIC16C74	PIC14000
06	PORTB	RESERVED
09	PORTE	UNIMPLEMENTED
0D	PIR2	UNIMPLEMENTED
0E	TMR1L	ADTMRL
0F	TMR1H	ADTMRH
10	T1CON	UNIMPLEMENTED
11	TMR2	UNIMPLEMENTED
12	T2CON	UNIMPLEMENTED
13	SSPBUF	I2CBUF
14	SSPCON	I2CCON
15	CCPR1L	ADCAPL
16	CCPR1H	ADCAPH
17	CCP1CON	UNIMPLEMENTED
18	RCSTA	UNIMPLEMENTED
19	TXREG	UNIMPLEMENTED
1A	RCREG	UNIMPLEMENTED
1B	CCPR2L	UNIMPLEMENTED
1C	CCPR2H	UNIMPLEMENTED
1D	CCP2CON	UNIMPLEMENTED
1E	ADRES	UNIMPLEMENTED
86	TRIS B	RESERVED
89	TRIS E	UNIMPLEMENTED
8D	PIE2	UNIMPLEMENTED
8F	UNIMPLEMENTED	SLPCON
92	PR2	UNIMPLEMENTED
93	SSPADD	I2CADD
94	SSPSTAT	I2CSTAT
98	TXSTA	UNIMPLEMENTED
99	SPBRG	UNIMPLEMENTED
9B	UNIMPLEMENTED	LDACA
9C	UNIMPLEMENTED	LDACB
9D	UNIMPLEMENTED	CHGCON
9E	UNIMPLEMENTED	MISC

Differences in Special Registers between PIC14000 and PIC16C74

This section details the differences in the special registers between the PIC14000 and the PIC16C74. Porting any code developed on the PIC16C74 to the PIC14000 must take these differences into account. In cases where registers have the same name but the bits have different functions, the differences in the individual bits are detailed. In cases where the register name and functionality has changed, only the register name is detailed. Refer to Table 4-3 for the definitions of the PIC14000 registers.

Address	Bit	PIC16C74	PIC14000
03	BIT7	IRP	RESERVED
	BIT6	RP1	RESERVED
06		PORTB	RESERVED
08		PORTD	RESERVED
09		PORTE	RESERVED
0B	BIT4	INTE	RESERVED
	BIT1	INTF	RESERVED
	BIT0	RBIF	RESERVED
0C	BIT7	PSPIF	WUIF
	BIT6	ADIF	NOT USED
	BIT5	RCIF	NOT USED
	BIT4	TXIF	PBIF
	BIT3	SSPIF	I2CIF
	BIT2	CCPIF	RCIF
	BIT1	TMR2IF	ADCIF
	BIT0	TMR1IF	OVFIF

LIST OF EXAMPLES

LIST OF FIGURES

LIST OF TABLES

CONNECTING TO MICROCHIP BBS

Connect worldwide to the Microchip BBS using the CompuServe® communications network. In most cases a local call is your only expense. The Microchip BBS connection does not use CompuServe membership services, therefore **you do not need CompuServe membership to join Microchip's BBS**.

There is **no charge** for connecting to the BBS, except for a toll charge to the CompuServe access number, where applicable. You do not need to be a CompuServe member to take advantage of this connection (you never actually log in to CompuServe).

The procedure to connect will vary slightly from country to country. Please check with your local CompuServe agent for details if you have a problem. CompuServe service allows multiple users at baud rates up to 14400 bps.

The following connect procedure applies in most locations:

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

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

NETWORK, depress **< ENTER↵ >**
and follow CompuServe's directions.

For voice information (or calling from overseas), you may call (614) 457-1550 for your local CompuServe number.

READER RESPONSE

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

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

To: Technical Publications Manager Total Pages Sent

RE: Reader Response

From: Name _____

Company _____

Address _____

City / State / ZIP / Country _____

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

Application (optional):

Would you like a reply? ___Y ___N

Device: PIC14000 Literature Number: DS40122A

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 usefullness?

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

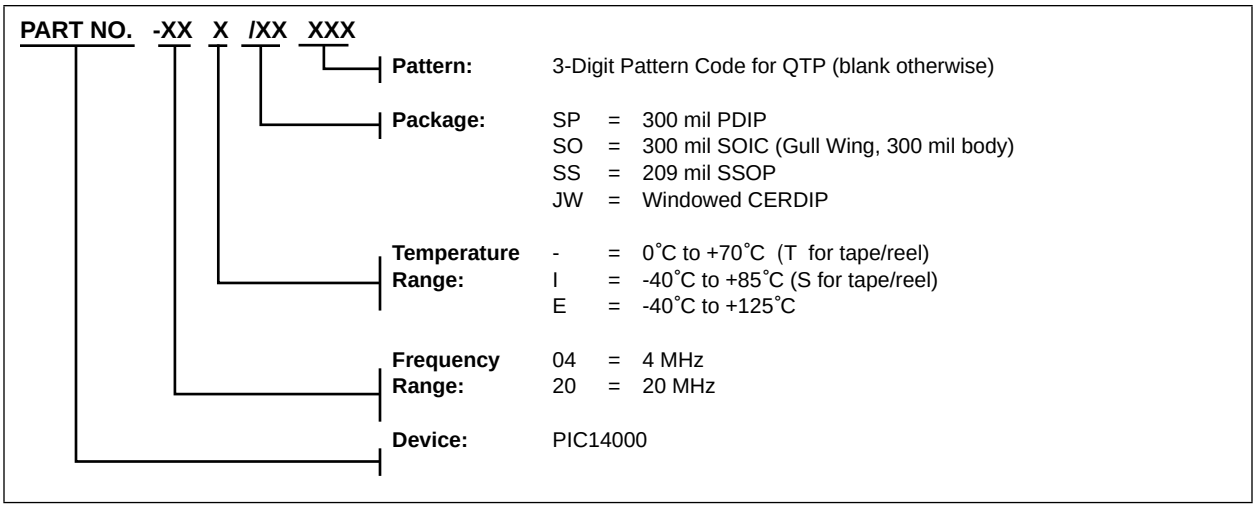
7. How would you improve this document?

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

NOTES:

PIC14000 Product Identification System

To order or to obtain information (e.g., on pricing or delivery), please use the listed part numbers, and refer to the factory or the listed sales offices.



AMERICAS

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

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

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

Chicago
Microchip Technology Inc.
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 708 285-0071 Fax: 708 285-0075

Dallas
Microchip Technology Inc.
14651 Dallas Parkway, Suite 816
Dallas, TX 75240-8809
Tel: 214 991-7177 Fax: 214 991-8588

Dayton
Microchip Technology Inc.
35 Rockridge Road
Englewood, OH 45322
Tel: 513 832-2543 Fax: 513 832-2841

Los Angeles
Microchip Technology Inc.
18201 Von Karman, Suite 455
Irvine, CA 92715
Tel: 714 263-1888 Fax: 714 263-1338

AMERICAS (continued)

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

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

ASIA/PACIFIC

Hong Kong
Microchip Technology
Unit No. 3002-3004, Tower 1
Metroplaza
223 Hing Fong Road
Kwai Fong, N.T. Hong Kong
Tel: 852 2 401 1200 Fax: 852 2 401 3431

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

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

EUROPE

United Kingdom
Arizona Microchip Technology Ltd.
Unit 6, The Courtyard
Meadow Bank, Furlong Road
Bourne End, Buckinghamshire SL8 5AJ
Tel: 44 0 1628 851077 Fax: 44 0 1628 850259

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

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

Italy
Arizona Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Pegaso Ingresso No. 2
Via Paracelso 23, 20041
Agrate Brianza (MI) Italy
Tel: 39 039 689 9939 Fax: 39 039 689 9883

JAPAN

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

9/22/95

MICROCHIP

Printed in the USA, 9/95