PIC16LF1902/3

28-Pin Flash-Based, 8-Bit CMOS MCUs with LCD Driver and XLP Technology

High-Performance RISC CPU

- C Compiler Optimized Architecture
- Only 49 Instructions
- Operating Speed:
  - DC – 20 MHz clock input @ 3.6V
  - DC – 16 MHz clock input @ 1.8V
  - DC – 200 ns instruction cycle
- Interrupt Capability with Automatic Context Saving
- 16-Level Deep Hardware Stack with Optional Overflow/Underflow Reset
- Direct, Indirect and Relative Addressing modes:
  - Two full 16-bit File Select Registers (FSRs)
  - FSRs can read program and data memory

Memory

- Up to 7 Kbytes Self-Write/Read Flash Program Memory Addressing
- Up to 256 Bytes Data Memory Addressing
- High-Endurance Flash Data Memory (HEF)
  - 128B of nonvolatile memory
  - 100K erase/write cycles

Flexible Oscillator Structure

- 16 MHz Internal Oscillator:
  - Accuracy to ± 3%, typical
  - Software selectable frequency range from 16 MHz to 31.25 kHz
- 31 kHz Low-Power Internal Oscillator
- Three External Clock modes up to 20 MHz
- Two-Speed Oscillator Start-up
- Low-Power RTC Implementation via LPT1OSC

Special Microcontroller Features

- Operating Voltage Range:
  - 1.8V-3.6V
- Self-Programmable under Software Control
- Power-on Reset (POR)
- Power-up Timer (PWRT)
- Low-Power Brown-Out Reset (LPBOR)
- Extended Watchdog Timer (WDT)
- In-Circuit Serial Programming™ (ICSP™) via Two Pins
- Enhanced Low-Voltage Programming (LVP)
- Programmable Code Protection
- Power-Saving Sleep mode

eXtreme Low-Power (XLP) Features (PIC16LF1902/3)

- Sleep Current
  - 30 nA @ 1.8V, typical
- Watchdog Timer Current:
  - 300 nA @ 1.8V, typical
- Secondary Oscillator: 500 nA @ 32 kHz,1.8V, typical

Analog Features

- Analog-to-Digital Converter (ADC):
  - 10-bit resolution, up to 11 channels
  - Conversion available during Sleep
  - Dedicated ADC RC oscillator
  - Fixed Voltage Reference (FVR) as channel
- Integrated Temperature Indicator
- Voltage Reference module:
  - Fixed Voltage Reference (FVR) with 1.024V and 2.048V output levels

Peripheral Highlights

- Up to 25 I/O Pins and 1 Input-only Pin:
  - High current 25 mA sink/source
  - Individually programmable weak pull-ups
  - Individually programmable interrupt-on-change (IOC) pins
- Integrated LCD Controller:
  - 19 segment pins and 72 total segments
  - Variable clock input
  - Contrast control
  - Internal voltage reference selections
- Timer0: 8-Bit Timer/Counter with 8-Bit Programmable Prescaler
- Enhanced Timer1:
  - 16-bit timer/counter with prescaler
  - External Gate Input mode
  - Dedicated low-power 32 kHz oscillator driver

PIC16LF1902/3 Family Types

Device	Data Sheet Index	Program Memory Flash (words)	Data SRAM (bytes)	High-Endurance Flash (bytes)	I/Os$^{(2)}$	10-bit ADC (ch)	Timers (8/16-bit)	EUSART	LCD			Debug$^{(1)}$	XLP
									Common Pins	Segment Pins	Total Segments		
PIC16LF1902	(1)	2048	128	128	25	11	1/1	—	4	19	72$^{(3)}$	H	Y
PIC16LF1903	(1)	4096	256	128	25	11	1/1	—	4	19	72$^{(3)}$	H	Y
PIC16LF1904	(2)	4096	256	128	36	14	1/1	1	4	29	116	I/H	Y
PIC16LF1906	(2)	8192	512	128	25	11	1/1	1	4	19	72$^{(3)}$	I/H	Y
PIC16LF1907	(2)	8192	512	128	36	14	1/1	1	4	29	116	I/H	Y

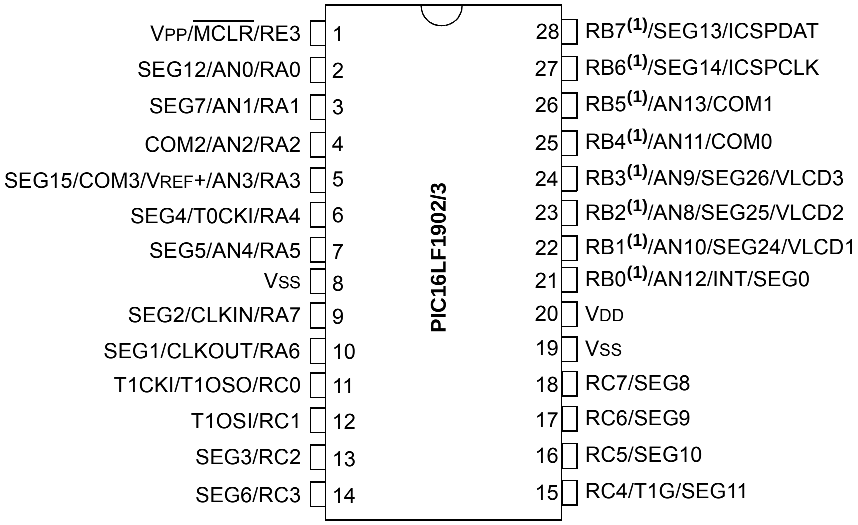
Note 1: I - Debugging, Integrated on Chip; H - Debugging, available using Debug Header.
2: One pin is input-only.
3: COM3 and SEG15 share a pin, so the total segments are limited to 72 for 28-pin devices.

Data Sheet Index: (Unshaded devices are described in this document.)

1: DS40001455 PIC16LF1902/1903 Data Sheet, 28-Pin Flash, 8-bit Microcontrollers.
2: DS40001569 PIC16LF1904/6/7 Data Sheet, 28/40/44-Pin Flash, 8-bit Microcontrollers.

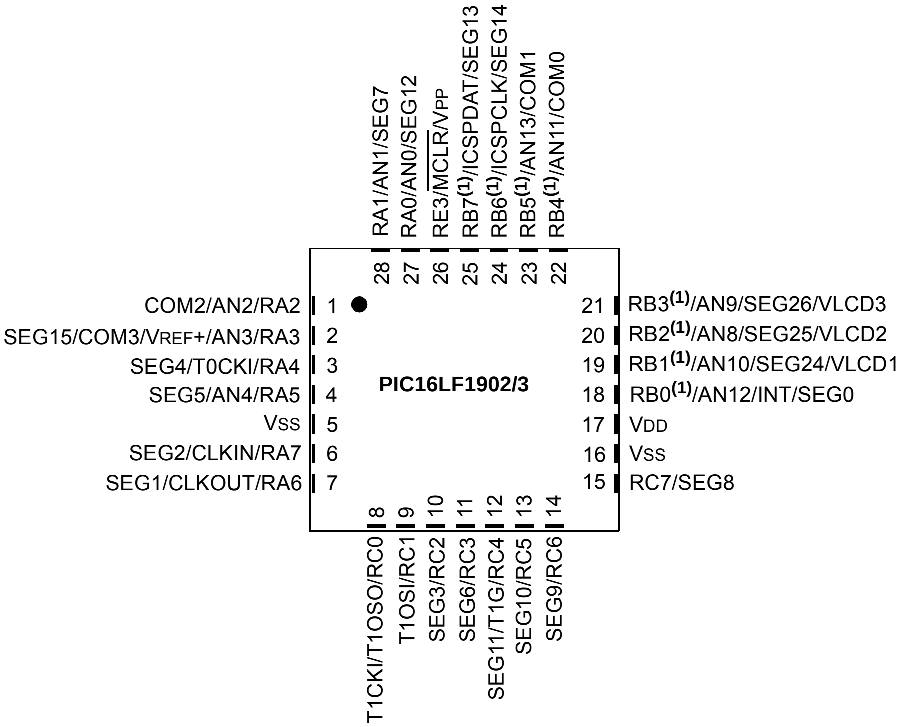
Pin Diagrams

FIGURE 1: 28-PIN PDIP, SOIC, SSOP


VPP/$\overline{\text{MCLR}}$/RE3 1
SEG12/AN0/RA0 2
SEG7/AN1/RA1 3
COM2/AN2/RA2 4
SEG15/COM3/VREF+/AN3/RA3 5
SEG4/T0CKI/RA4 6
SEG5/AN4/RA5 7
VSS 8
SEG2/CLKIN/RA7 9
SEG1/CLKOUT/RA6 10
T1CKI/T1OSO/RC0 11
T1OSI/RC1 12
SEG3/RC2 13
SEG6/RC3 14
PIC16LF1902/3
28 RB7$^{(1)}$/SEG13/ICSPDAT
27 RB6$^{(1)}$/SEG14/ICSPCLK
26 RB5$^{(1)}$/AN13/COM1
25 RB4$^{(1)}$/AN11/COM0
24 RB3$^{(1)}$/AN9/SEG26/VLCD3
23 RB2$^{(1)}$/AN8/SEG25/VLCD2
22 RB1$^{(1)}$/AN10/SEG24/VLCD1
21 RB0$^{(1)}$/AN12/INT/SEG0
20 VDD
19 VSS
18 RC7/SEG8
17 RC6/SEG9
16 RC5/SEG10
15 RC4/T1G/SEG11

Note 1: These pins have interrupt-on-change functionality.

FIGURE 2: 28-PIN UQFN


RA1/AN1/SEG7
RA0/AN0/SEG12
RE3/MCLR/VPP
RB7(1)/ICSPDAT/SEG13
RB6(1)/ICSPCLK/SEG14
RB5(1)/AN13/COM1
RB4(1)/AN11/COM0
28
27
26
25
24
23
22
COM2/AN2/RA2
SEG15/COM3/VREF+/AN3/RA3
SEG4/T0CKI/RA4
SEG5/AN4/RA5
VSS
SEG2/CLKIN/RA7
SEG1/CLKOUT/RA6
1
2
3
4
5
6
7
PIC16LF1902/3
21
20
19
18
17
16
15
RB3(1)/AN9/SEG26/VLCD3
RB2(1)/AN8/SEG25/VLCD2
RB1(1)/AN10/SEG24/VLCD1
RB0(1)/AN12/INT/SEG0
VDD
VSS
RC7/SEG8
8
9
10
11
12
13
14
T1CKI/T1OSO/RC0
T1OSI/RC1
SEG3/RC2
SEG6/RC3
SEG11/T1G/RC4
SEG10/RC5
SEG9/RC6

Note 1: These pins have interrupt-on-change functionality.

TABLE 1: 28-PIN ALLOCATION TABLE (PIC16LF1902/3)

I/O	28-Pin PDIP/ SOIC/SSOP	28-Pin UQFN	A/D	Timers	LCD	Interrupt	Pull-up	Basic
RA0	2	27	AN0	—	SEG12	—	—	—
RA1	3	28	AN1	—	SEG7	—	—	—
RA2	4	1	AN2	—	COM2	—	—	—
RA3	5	2	AN3/VREF+	—	SEG15/COM3	—	—	—
RA4	6	3	—	T0CKI	SEG4	—	—	—
RA5	7	4	AN4	—	SEG5	—	—	—
RA6	10	7	—	—	SEG1	—	—	CLKOUT
RA7	9	6	—	—	SEG2	—	—	CLKIN
RB0	21	18	AN12	—	SEG0	INT/IOC	Y	—
RB1	22	19	AN10	—	VLCD1/SEG24	IOC	Y	—
RB2	23	20	AN8	—	VLCD2/SEG25	IOC	Y	—
RB3	24	21	AN9	—	VLCD3/SEG26	IOC	Y	—
RB4	25	22	AN11	—	COM0	IOC	Y	—
RB5	26	23	AN13	—	COM1	IOC	Y	—
RB6	27	24	—	—	SEG14	IOC	Y	ICSPCLK
RB7	28	25	—	—	SEG13	IOC	Y	ICSPDAT
RC0	11	8	—	T1OSO/T1CKI	—	—	—	—
RC1	12	9	—	T1OSI	—	—	—	—
RC2	13	10	—	—	SEG3	—	—	—
RC3	14	11	—	—	SEG6	—	—	—
RC4	15	12	—	T1G	SEG11	—	—	—
RC5	16	13	—	—	SEG10	—	—	—
RC6	17	14	—	—	SEG9	—	—	—
RC7	18	15	—	—	SEG8	—	—	—
RE3	1	26	—	—	—	—	Y(1)	$\overline{\text{MCLR}}$/VPP
VDD	20	17	—	—	—	—	—	VDD
Vss	8,19	5,16	—	—	—	—	—	VSS

Note 1: Weak pull-up always enabled when $\overline{\text{MCLR}}$ is enabled, otherwise the pull-up is under user control.

Table of Contents

1.0 Device Overview 6
2.0 Enhanced Mid-Range CPU 10
3.0 Memory Organization 12
4.0 Device Configuration 33
5.0 Resets 38
6.0 Oscillator Module 46
7.0 Interrupts 55
8.0 Power-Down Mode (Sleep) 66
9.0 Watchdog Timer 68
10.0 Flash Program Memory Control 72
11.0 I/O Ports 88
12.0 Interrupt-On-Change 100
13.0 Fixed Voltage Reference (FVR) 103
14.0 Temperature Indicator Module 105
15.0 Analog-to-Digital Converter (ADC) Module 106
16.0 Timer0 Module 119
17.0 Timer1 Module with Gate Control 122
18.0 Liquid Crystal Display (LCD) Driver Module 133
19.0 In-Circuit Serial Programming™ (ICSP™) 166
20.0 Instruction Set Summary 169
21.0 Electrical Specifications 183
22.0 DC and AC Characteristics Graphs and Charts 199
23.0 Development Support 200
24.0 Packaging Information 204
Appendix A: Data Sheet Revision History 215
The Microchip Website 216
Customer Change Notification Service 216
Customer Support 216
Product Identification System 217

TO OUR VALUED CUSTOMERS

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

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at **docerrors@microchip.com**. We welcome your feedback.

Most Current Data Sheet

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

http://www.microchip.com

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000000A is version A of document DS30000000).

Errata

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

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

- Microchip's Worldwide Website; **http://www.microchip.com**
- Your local Microchip sales office (see last page)

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

Customer Notification System

Register on our website at **www.microchip.com** to receive the most current information on all of our products.

1.0 DEVICE OVERVIEW

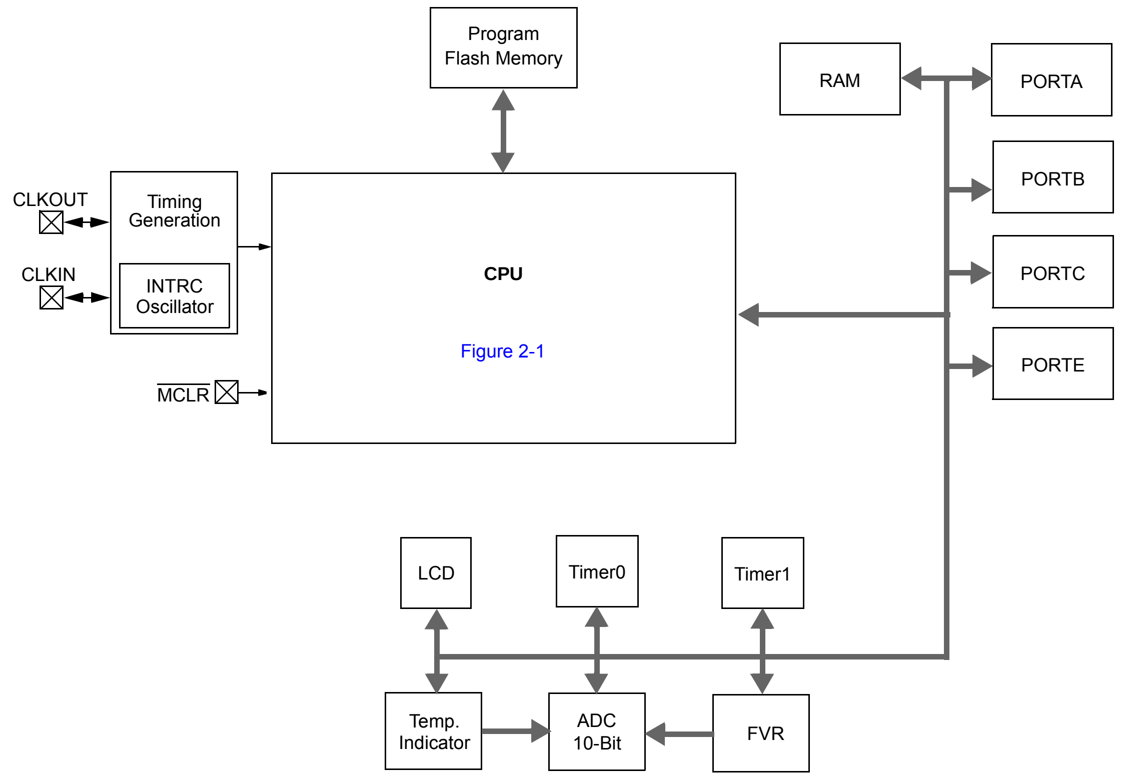
The PIC16LF1902/3 devices are described within this data sheet. They are available in 28-pin packages. Figure 1-1 shows a block diagram of the PIC16LF1902/3 devices. Table 1-2 shows the pinout descriptions.

Reference Table 1-1 for peripherals available per device.

TABLE 1-1: DEVICE PERIPHERAL SUMMARY

Peripheral		PIC16LF1902	PIC16LF1903
ADC		●	●
Fixed Voltage Reference (FVR)		●	●
LCD		●	●
Temperature Indicator		●	●
Timers			
	Timer0	●	●
	Timer1	●	●

FIGURE 1-1: PIC16LF1902/3 BLOCK DIAGRAM


Program
Flash Memory
RAM
PORTA
PORTB
PORTC
PORTE
CLKOUT
CLKIN
Timing
Generation
INTRC
Oscillator
CPU
Figure 2-1
MCLR
LCD
Timer0
Timer1
Temp.
Indicator
ADC
10-Bit
FVR

Note 1: See applicable chapters for more information on peripherals.

TABLE 1-2: PIC16LF1902/3 PINOUT DESCRIPTION

Name	Function	Input Type	Output Type	Description
RA0/AN0/SEG12	RA0	TTL	CMOS	General purpose I/O.
	AN0	AN	—	A/D Channel 0 input.
	SEG12	—	AN	LCD Analog output.
RA1/AN1/SEG7	RA1	TTL	CMOS	General purpose I/O.
	AN1	AN	—	A/D Channel 1 input.
	SEG7	—	AN	LCD Analog output.
RA2/AN2/COM2	RA2	TTL	CMOS	General purpose I/O.
	AN2	AN	—	A/D Channel 2 input.
	COM2	—	AN	LCD Analog output.
RA3/AN3/VREF+/COM3/SEG15	RA3	TTL	CMOS	General purpose I/O.
	AN3	AN	—	A/D Channel 3 input.
	VREF+	AN	—	A/D Voltage Reference input.
	COM3	—	AN	LCD Analog output.
	SEG15	—	AN	LCD Analog output.
RA4/T0CKI/SEG4	RA4	TTL	CMOS	General purpose I/O.
	T0CKI	ST	—	Timer0 clock input.
	SEG4	—	AN	LCD Analog output.
RA5/AN4/SEG5	RA5	TTL	CMOS	General purpose I/O.
	AN4	AN	—	A/D Channel 4 input.
	SEG5	—	AN	LCD Analog output.
RA6/CLKOUT/SEG1	RA6	TTL	CMOS	General purpose I/O.
	CLKOUT	—	CMOS	FOSC/4 output.
	SEG1	—	AN	LCD Analog output.
RA7/CLKIN/SEG2	RA7	TTL	CMOS	General purpose I/O.
	CLKIN	CMOS	—	External clock input (EC mode).
	SEG2	—	AN	LCD Analog output.
RB0/AN12/INT/SEG0	RB0	TTL	CMOS	General purpose I/O.
	AN12	AN	—	A/D Channel 12 input.
	INT	ST	—	External interrupt.
	SEG0	—	AN	LCD Analog output.
RB1(1)/AN10/SEG24/VLCD1	RB1	TTL	CMOS	General purpose I/O.
	AN10	AN	—	A/D Channel 10 input.
	SEG24	—	AN	LCD Analog output.
	VLCD1	AN	—	LCD analog input.
RB2(1)/AN8/SEG25/VLCD2	RB2	TTL	CMOS	General purpose I/O.
	AN8	AN	—	A/D Channel 8 input.
	SEG25	—	AN	LCD Analog output.
	VLCD2	AN	—	LCD analog input.
RB3(1)/AN9/SEG26/VLCD3	RB3	TTL	CMOS	General purpose I/O.
	AN9	AN	—	A/D Channel 9 input.
	SEG26	—	AN	LCD Analog output.
	VLCD3	AN	—	LCD analog input.

Legend:
AN = Analog input or output
TTL = TTL compatible input
HV = High Voltage
CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels
XTAL = Crystal
OD = Open-Drain
I^2C = Schmitt Trigger input with I^2C levels

Note 1: These pins have interrupt-on-change functionality.

TABLE 1-2: PIC16LF1902/3 PINOUT DESCRIPTION (CONTINUED)

Name	Function	Input Type	Output Type	Description
RB4(1)/AN11/COM0	RB4	TTL	CMOS	General purpose I/O.
	AN11	AN	—	A/D Channel 11 input.
	COM0	—	AN	LCD Analog output.
RB5(1)/AN13/COM1	RB5	TTL	CMOS	General purpose I/O.
	AN13	AN	—	A/D Channel 13 input.
	COM1	—	AN	LCD Analog output.
RB6(1)/ICSPCLK/SEG14	RB6	TTL	CMOS	General purpose I/O.
	ICSPCLK	ST	—	Serial Programming Clock.
	SEG14	—	AN	LCD Analog output.
RB7(1)/ICSPDAT/SEG13	RB7	TTL	CMOS	General purpose I/O.
	ICSPDAT	ST	CMOS	ICSP™ Data I/O.
	SEG13	—	AN	LCD Analog output.
RC0/T1OSO/T1CKI	RC0	TTL	CMOS	General purpose I/O.
	T1OSO	XTAL	XTAL	Timer1 oscillator connection.
	T1CKI	ST	—	Timer1 clock input.
RC1/T1OSI	RC1	TTL	CMOS	General purpose I/O.
	T1OSI	XTAL	XTAL	Timer1 oscillator connection.
RC2/SEG3	RC2	TTL	CMOS	General purpose I/O.
	SEG3	—	AN	LCD Analog output.
RC3/SEG6	RC3	TTL	CMOS	General purpose I/O.
	SEG6	—	AN	LCD Analog output.
RC4/T1G/SEG11	RC4	TTL	CMOS	General purpose I/O.
	T1G	XTAL	XTAL	Timer1 oscillator connection.
	SEG11	—	AN	LCD Analog output.
RC5/SEG10	RC5	TTL	CMOS	General purpose I/O.
	SEG10	—	AN	LCD Analog output.
RC6/SEG9	RC6	ST	CMOS	General purpose I/O.
	SEG9	—	AN	LCD Analog output.
RC7/SEG8	RC7	ST	CMOS	General purpose I/O.
	SEG8	—	AN	LCD Analog output.
RE3/$\overline{\text{MCLR}}$/VPP	RE3	TTL	CMOS	General purpose I/O.
	$\overline{\text{MCLR}}$	ST	—	Master Clear with internal pull-up.
	VPP	HV	—	Programming voltage.
VDD	VDD	Power	—	Positive supply.
VSS	VSS	Power	—	Ground reference.

Legend: AN = Analog input or output
TTL = TTL compatible input
HV = High Voltage
CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels
XTAL = Crystal
OD = Open-Drain
I^2C = Schmitt Trigger input with I^2C levels

Note 1: These pins have interrupt-on-change functionality.

2.0 ENHANCED MID-RANGE CPU

This family of devices contains an enhanced mid-range 8-bit CPU core. The CPU has 49 instructions. Interrupt capability includes automatic context saving. The hardware stack is 16 levels deep and has Overflow and Underflow Reset capability. Direct, Indirect, and Relative Addressing modes are available. Two File Select Registers (FSRs) provide the ability to read program and data memory.

- Automatic Interrupt Context Saving
- 16-level Stack with Overflow and Underflow
- File Select Registers
- Instruction Set

2.1 Automatic Interrupt Context Saving

During interrupts, certain registers are automatically saved in shadow registers and restored when returning from the interrupt. This saves stack space and user code. See **Section 7.5 "Automatic Context Saving"**, for more information.

2.2 16-level Stack with Overflow and Underflow

These devices have an external stack memory 15 bits wide and 16 words deep. A Stack Overflow or Underflow will set the appropriate bit (STKOVF or STKUNF) in the PCON register, and if enabled will cause a software Reset. See **Section 3.4 "Stack"** for more details.

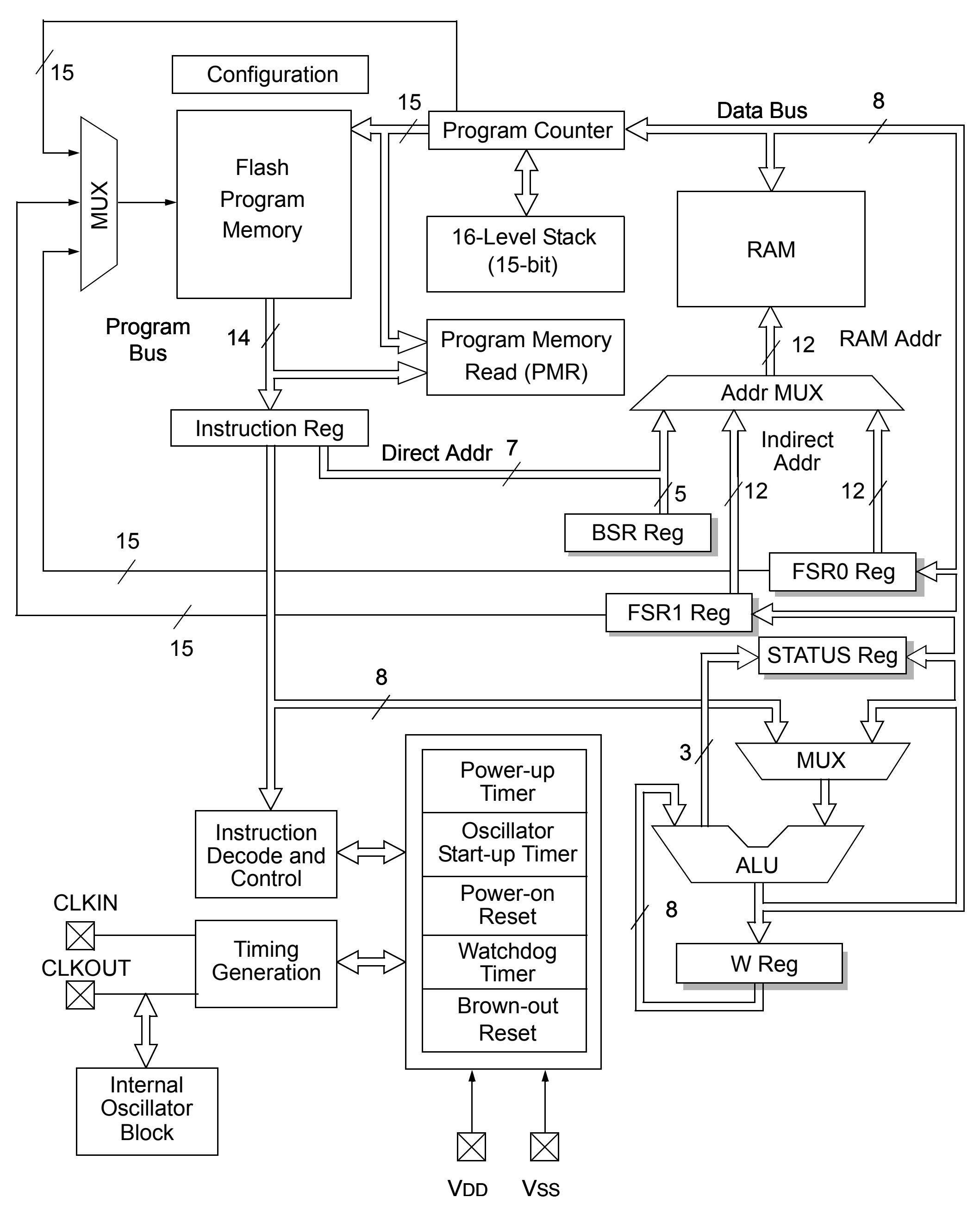
2.3 File Select Registers

There are two 16-bit File Select Registers (FSR). FSRs can access all file registers and program memory, which allows one Data Pointer for all memory. When an FSR points to program memory, there is one additional instruction cycle in instructions using INDF to allow the data to be fetched. General purpose memory can now also be addressed linearly, providing the ability to access contiguous data larger than 80 bytes. There are also new instructions to support the FSRs. See **Section 3.5 "Indirect Addressing"** for more details.

2.4 Instruction Set

There are 49 instructions for the enhanced mid-range CPU to support the features of the CPU. See **Section 20.0 "Instruction Set Summary"** for more details.

FIGURE 2-1: CORE BLOCK DIAGRAM


15
Configuration
15
Program Counter
Data Bus
8
MUX
Flash
Program
Memory
16-Level Stack
(15-bit)
RAM
Program
Bus
14
Program Memory
Read (PMR)
12
RAM Addr
Addr MUX
Instruction Reg
Direct Addr
7
Indirect
Addr
5
12
12
BSR Reg
15
FSR0 Reg
FSR1 Reg
15
STATUS Reg
8
3
MUX
Power-up
Timer
Oscillator
Start-up Timer
Power-on
Reset
Watchdog
Timer
Brown-out
Reset
Instruction
Decode and
Control
ALU
8
CLKIN
CLKOUT
Timing
Generation
W Reg
Internal
Oscillator
Block
VDD
VSS

3.0 MEMORY ORGANIZATION

These devices contain the following types of memory:

- Program Memory
  - Configuration Words
  - Device ID
  - User ID
  - Flash Program Memory
- Data Memory
  - Core Registers
  - Special Function Registers
  - General Purpose RAM
  - Common RAM

The following features are associated with access and control of program memory and data memory:

- PCL and PCLATH
- Stack
- Indirect Addressing

3.1 Program Memory Organization

The enhanced mid-range core has a 15-bit program counter capable of addressing 32K x 14 program memory space. Table 3-1 shows the memory sizes implemented for the PIC16LF1902/3 family. Accessing a location above these boundaries will cause a wrap-around within the implemented memory space. The Reset vector is at 0000h and the interrupt vector is at 0004h (see Figures 3-1, and 3-2).

TABLE 3-1: DEVICE SIZES AND ADDRESSES

Device	Program Memory Space (Words)	Last Program Memory Address	High-Endurance Flash Memory Address Range[1]
PIC16LF1902	2,048	07FFh	0780h-07FFh
PIC16LF1903	4,096	0FFFh	0F80h-0FFFh

Note 1: High-endurance Flash applies to low byte of each address in the range.

FIGURE 3-1: PROGRAM MEMORY MAP AND STACK FOR PIC16LF1902

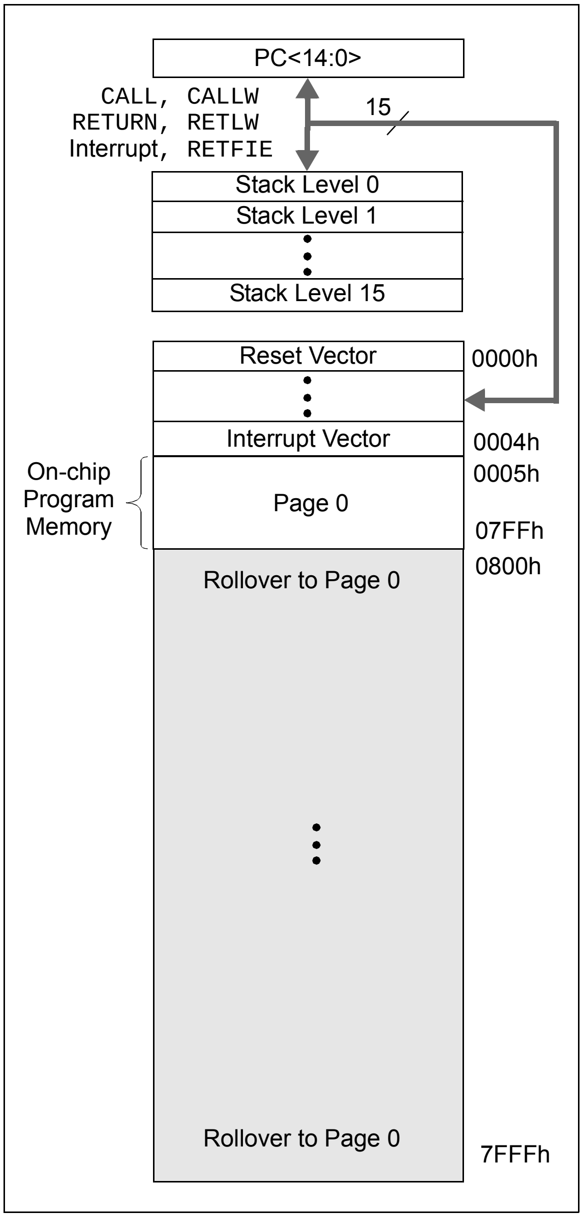
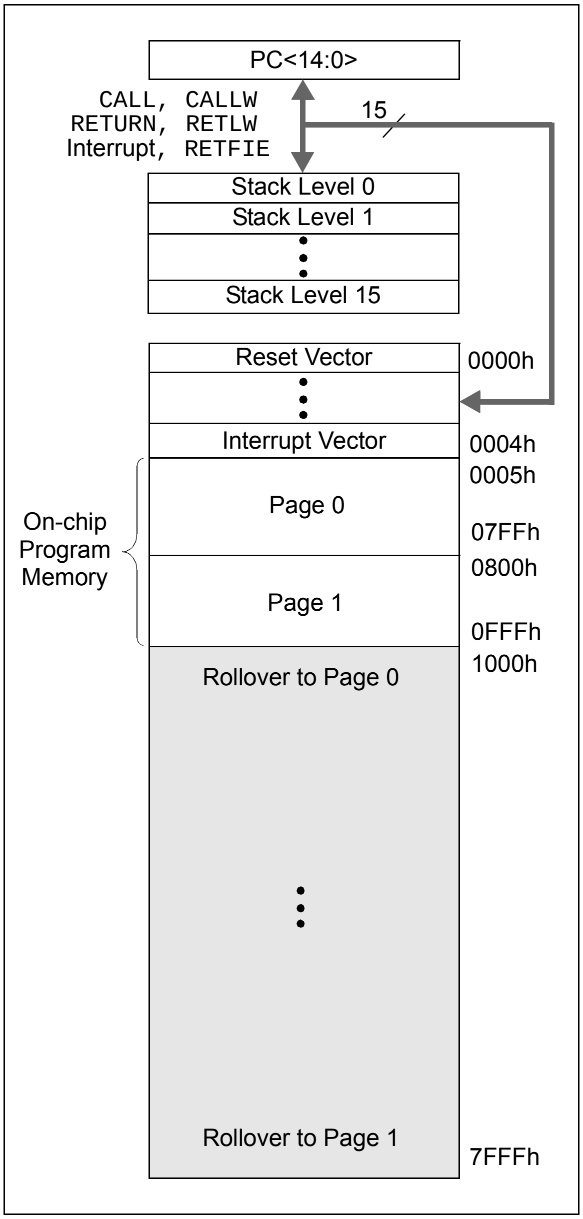

PC<14:0>
CALL, CALLW
RETURN, RETLW
Interrupt, RETFIE
15
Stack Level 0
Stack Level 1
Stack Level 15
Reset Vector
0000h
Interrupt Vector
0004h
0005h
On-chip Program Memory
Page 0
07FFh
0800h
Rollover to Page 0
Rollover to Page 0
7FFFh

FIGURE 3-2: PROGRAM MEMORY MAP AND STACK FOR PIC16LF1903


PC<14:0>
CALL, CALLW
RETURN, RETLW
Interrupt, RETFIE
15
Stack Level 0
Stack Level 1
Stack Level 15
Reset Vector
0000h
Interrupt Vector
0004h
0005h
On-chip Program Memory
Page 0
07FFh
0800h
Page 1
0FFFh
1000h
Rollover to Page 0
Rollover to Page 1
7FFFh

3.1.1 READING PROGRAM MEMORY AS DATA

There are two methods of accessing constants in program memory. The first method is to use tables of `RETLW` instructions. The second method is to set an FSR to point to the program memory.

3.1.1.1 `RETLW` Instruction

The `RETLW` instruction can be used to provide access to tables of constants. The recommended way to create such a table is shown in Example 3-1.

EXAMPLE 3-1: RETLW INSTRUCTION

```
constants
    BRW                   ;Add Index in W to
                          ;program counter to
                          ;select data
    RETLW DATA0           ;Index0 data
    RETLW DATA1           ;Index1 data
    RETLW DATA2
    RETLW DATA3


my_function
    ;… LOTS OF CODE…
    MOVLW      DATA_INDEX
    call constants
    ;… THE CONSTANT IS IN W
```

The `BRW` instruction makes this type of table very simple to implement. If your code must remain portable with previous generations of microcontrollers, then the `BRW` instruction is not available so the older table read method must be used.

3.1.1.2 Indirect Read with FSR

The program memory can be accessed as data by setting bit 7 of the FSRxH register and reading the matching INDFx register. The `MOVIW` instruction will place the lower eight bits of the addressed word in the W register. Writes to the program memory cannot be performed via the INDF registers. Instructions that access the program memory via the FSR require one extra instruction cycle to complete. Example 3-2 demonstrates accessing the program memory via an FSR.

The High directive will set bit<7> if a label points to a location in program memory.

EXAMPLE 3-2: ACCESSING PROGRAM MEMORY VIA FSR

```
constants
    RETLW DATA0        ;Index0 data
    RETLW DATA1        ;Index1 data
    RETLW DATA2
    RETLW DATA3
my_function
    ;… LOTS OF CODE…
    MOVLW   LOW constants
    MOVWF   FSR1L
    MOVLW   HIGH constants
    MOVWF   FSR1H
    MOVIW 0[FSR1]
;THE PROGRAM MEMORY IS IN W
```

3.2 Data Memory Organization

The data memory is partitioned in 32 memory banks with 128 bytes in a bank. Each bank consists of (Figure 3-3):

- 12 core registers
- 20 Special Function Registers (SFR)
- Up to 80 bytes of General Purpose RAM (GPR)
- 16 bytes of common RAM

The active bank is selected by writing the bank number into the Bank Select Register (BSR). Unimplemented memory will read as ‘`0`’. All data memory can be accessed either directly (via instructions that use the file registers) or indirectly via the two File Select Registers (FSR). See **Section 3.5 “Indirect Addressing”** for more information.

Data Memory uses a 12-bit address. The upper seven bits of the address define the Bank address and the lower five bits select the registers/RAM in that bank.

3.2.1 CORE REGISTERS

The core registers contain the registers that directly affect the basic operation. The core registers occupy the first 12 addresses of every data memory bank (addresses x00h/x08h through x0Bh/x8Bh). These registers are listed below in **Table 3-2**. For for detailed information, see **Table 3-4**.

TABLE 3-2: CORE REGISTERS

Addresses	BANKx
x00h or x80h	INDF0
x01h or x81h	INDF1
x02h or x82h	PCL
x03h or x83h	STATUS
x04h or x84h	FSR0L
x05h or x85h	FSR0H
x06h or x86h	FSR1L
x07h or x87h	FSR1H
x08h or x88h	BSR
x09h or x89h	WREG
x0Ah or x8Ah	PCLATH
x0Bh or x8Bh	INTCON

3.2.1.1 STATUS Register

The STATUS register, shown in Register 3-1, contains:

- the arithmetic status of the ALU
- the Reset status

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

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

It is recommended, therefore, that only `BCF`, `BSF`, `SWAPF` and `MOVWF` instructions are used to alter the STATUS register, because these instructions do not affect any Status bits. For other instructions not affecting any Status bits (Refer to **Section 20.0 "Instruction Set Summary"**).

> **Note:** The C and DC bits operate as $\overline{\mathrm{Borrow}}$ and $\overline{\mathrm{Digit\ Borrow}}$ out bits, respectively, in subtraction.

REGISTER 3-1: STATUS: STATUS REGISTER

U-0	U-0	U-0	R-1/q	R-1/q	R/W-0/u	R/W-0/u	R/W-0/u
—	—	—	$\overline{\mathrm{TO}}$	$\overline{\mathrm{PD}}$	Z	DC[1]	C[1]
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	q = Value depends on condition

bit 7-5 **Unimplemented:** Read as '`0`'

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

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

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

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

bit 0 **C:** Carry/$\overline{\mathrm{Borrow}}$ bit[1] (`ADDWF`, `ADDLW`, `SUBLW`, `SUBWF` instructions)[1]
`1` = A carry-out from the Most Significant bit of the result occurred
`0` = No carry-out from the Most Significant bit of the result occurred

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

3.2.2 SPECIAL FUNCTION REGISTER

The Special Function Registers (SFRs) are registers used by the application to control the desired operation of peripheral functions in the device. The Special Function Registers occupy the 20 bytes after the core registers of every data memory bank (addresses x0Ch/x8Ch through x1Fh/x9Fh). The registers associated with the operation of the peripherals are described in the appropriate peripheral chapter of this data sheet.

3.2.3 GENERAL PURPOSE RAM

There are up to 80 bytes of GPR in each data memory bank. The Special Function Registers occupy the 20 bytes after the core registers of every data memory bank (addresses x0Ch/x8Ch through x1Fh/x9Fh).

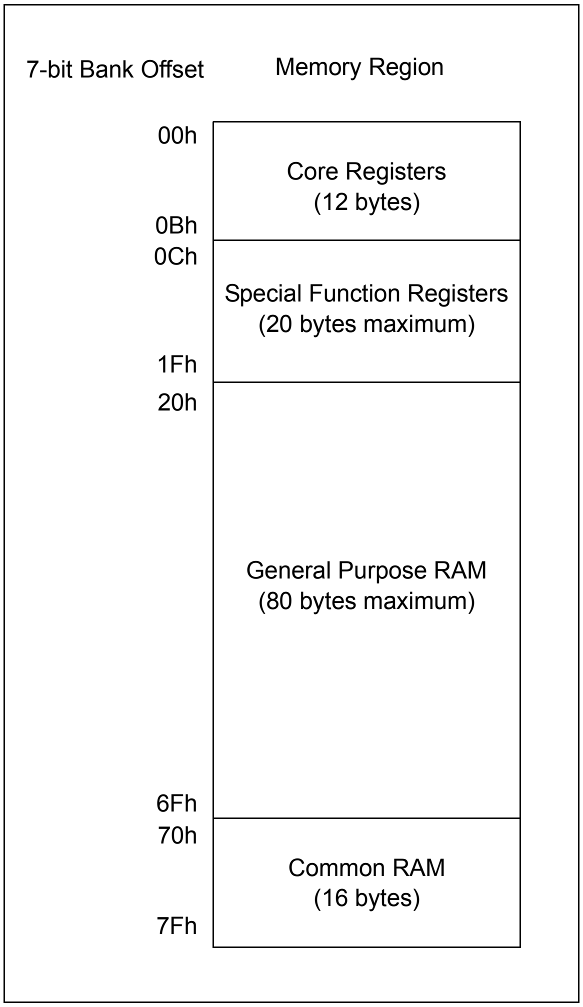
3.2.3.1 Linear Access to GPR

The general purpose RAM can be accessed in a non-banked method via the FSRs. This can simplify access to large memory structures. See **Section 3.5.2 "Linear Data Memory"** for more information.

3.2.4 COMMON RAM

There are 16 bytes of common RAM accessible from all banks.

FIGURE 3-3: BANKED MEMORY PARTITIONING


7-bit Bank Offset
Memory Region
00h
Core Registers
(12 bytes)
0Bh
0Ch
Special Function Registers
(20 bytes maximum)
1Fh
20h
General Purpose RAM
(80 bytes maximum)
6Fh
70h
Common RAM
(16 bytes)
7Fh

3.2.5 DEVICE MEMORY MAPS

The memory maps for PIC16LF1902 and PIC16LF1903 are as shown in **Table 3-3**.

TABLE 3-3: PIC16LF1902/3 MEMORY MAP

	BANK 0		BANK 1		BANK 2		BANK 3		BANK 4		BANK 5		BANK 6		BANK 7
000h	Core Registers (Table 3-2)	080h	Core Registers (Table 3-2)	100h	Core Registers (Table 3-2)	180h	Core Registers (Table 3-2)	200h	Core Registers (Table 3-2)	280h	Core Registers (Table 3-2)	300h	Core Registers (Table 3-2)	380h	Core Registers (Table 3-2)
00Bh		08Bh		10Bh		18Bh		20Bh		28Bh		30Bh		38Bh	
00Ch	PORTA	08Ch	TRISA	10Ch	LATA	18Ch	ANSELA	20Ch	—	28Ch	—	30Ch	—	38Ch	—
00Dh	PORTB	08Dh	TRISB	10Dh	LATB	18Dh	ANSELB	20Dh	WPUB	28Dh	—	30Dh	—	38Dh	—
00Eh	PORTC	08Eh	TRISC	10Eh	LATC	18Eh	—	20Eh	—	28Eh	—	30Eh	—	38Eh	—
00Fh	—	08Fh	—	10Fh	—	18Fh	—	20Fh	—	28Fh	—	30Fh	—	38Fh	—
010h	PORTE	090h	—	110h	—	190h	—	210h	WPUE	290h	—	310h	—	390h	—
011h	PIR1	091h	PIE1	111h	—	191h	PMADRL	211h	—	291h	—	311h	—	391h	—
012h	PIR2	092h	PIE2	112h	—	192h	PMADRH	212h	—	292h	—	312h	—	392h	—
013h	—	093h	—	113h	—	193h	PMDATL	213h	—	293h	—	313h	—	393h	—
014h	—	094h	—	114h	—	194h	PMDATH	214h	—	294h	—	314h	—	394h	IOCBP
015h	TMR0	095h	OPTION_REG	115h	—	195h	PMCON1	215h	—	295h	—	315h	—	395h	IOCBN
016h	TMR1L	096h	PCON	116h	BORCON	196h	PMCON2	216h	—	296h	—	316h	—	396h	IOCBF
017h	TMR1H	097h	WDTCON	117h	FVRCON	197h	—	217h	—	297h	—	317h	—	397h	—
018h	T1CON	098h	—	118h	—	198h	—	218h	—	298h	—	318h	—	398h	—
019h	T1GCON	099h	OSCCON	119h	—	199h	—	219h	—	299h	—	319h	—	399h	—
01Ah	—	09Ah	OSCSTAT	11Ah	—	19Ah	—	21Ah	—	29Ah	—	31Ah	—	39Ah	—
01Bh	—	09Bh	ADRESL	11Bh	—	19Bh	—	21Bh	—	29Bh	—	31Bh	—	39Bh	—
01Ch	—	09Ch	ADRESH	11Ch	—	19Ch	—	21Ch	—	29Ch	—	31Ch	—	39Ch	—
01Dh	—	09Dh	ADCON0	11Dh	—	19Dh	—	21Dh	—	29Dh	—	31Dh	—	39Dh	—
01Eh	—	09Eh	ADCON1	11Eh	—	19Eh	—	21Eh	—	29Eh	—	31Eh	—	39Eh	—
01Fh	—	09Fh	—	11Fh	—	19Fh	—	21Fh	—	29Fh	—	31Fh	—	39Fh	—
020h	General Purpose Register 96 Bytes	0A0h	General Purpose Register 32 Bytes	120h	General Purpose Register 80 Bytes$^{(1)}$	1A0h	Unimplemented Read as ‘0’	220h	Unimplemented Read as ‘0’	2A0h	Unimplemented Read as ‘0’	320h	Unimplemented Read as ‘0’	3A0h	Unimplemented Read as ‘0’
				13Fh											
			General Purpose Register 48 Bytes$^{(1)}$	140h											
06Fh		0EFh		16Fh		1EFh		26Fh		2EFh		36Fh		3EFh	
070h		0F0h	Accesses 70h – 7Fh	170h	Accesses 70h – 7Fh	1F0h	Accesses 70h – 7Fh	270h	Accesses 70h – 7Fh	2F0h	Accesses 70h – 7Fh	370h	Accesses 70h – 7Fh	3F0h	Accesses 70h – 7Fh
07Fh		0FFh		17Fh		1FFh		27Fh		2FFh		37Fh		3FFh	

Legend: ▨ = Unimplemented data memory locations, read as ‘0’.

Note 1: PIC16LF1903 only.

TABLE 3-3: PIC16LF1902/3 MEMORY MAP (CONTINUED)

Address	BANK 8	Address	BANK 9	Address	BANK 10	Address	BANK 11	Address	BANK 12	Address	BANK 13	Address	BANK 14
400h 40Bh	Core Registers (Table 3-2)	480h 48Bh	Core Registers (Table 3-2)	500h 50Bh	Core Registers (Table 3-2)	580h 58Bh	Core Registers (Table 3-2)	600h 60Bh	Core Registers (Table 3-2)	680h 68Bh	Core Registers (Table 3-2)	700h 70Bh	Core Registers (Table 3-2)
40Ch 46Fh	Unimplemented Read as ‘0’	48Ch 4EFh	Unimplemented Read as ‘0’	50Ch 56Fh	Unimplemented Read as ‘0’	58Ch 5EFh	Unimplemented Read as ‘0’	60Ch 66Fh	Unimplemented Read as ‘0’	68Ch 6EFh	Unimplemented Read as ‘0’	70Ch 76Fh	Unimplemented Read as ‘0’
470h 47Fh	Common RAM (Accesses 70h – 7Fh)	4F0h 4FFh	Common RAM (Accesses 70h – 7Fh)	570h 57Fh	Common RAM (Accesses 70h – 7Fh)	5F0h 5FFh	Common RAM (Accesses 70h – 7Fh)	670h 67Fh	Common RAM (Accesses 70h – 7Fh)	6F0h 6FFh	Common RAM (Accesses 70h – 7Fh)	770h 77Fh	Common RAM (Accesses 70h – 7Fh)

Address	BANK 16	Address	BANK 17	Address	BANK 18	Address	BANK 19	Address	BANK 20	Address	BANK 21	Address	BANK 22	Address	BANK 23
800h 80Bh	Core Registers (Table 3-2)Table 3-2	880h 88Bh	Core Registers (Table 3-2)	900h 90Bh	Core Registers (Table 3-2)	980h 98Bh	Core Registers (Table 3-2)	A00h A0Bh	Core Registers (Table 3-2)	A80h A8Bh	Core Registers (Table 3-2)	B00h B0Bh	Core Registers (Table 3-2)	B80h B8Bh	Core Registers (Table 3-2)
80Ch 86Fh	Unimplemented Read as ‘0’	88Ch 8EFh	Unimplemented Read as ‘0’	90Ch 96Fh	Unimplemented Read as ‘0’	98Ch 9EFh	Unimplemented Read as ‘0’	A0Ch A6Fh	Unimplemented Read as ‘0’	A8Ch AEFh	Unimplemented Read as ‘0’	B0Ch B6Fh	Unimplemented Read as ‘0’	B8Ch BEFh	Unimplemented Read as ‘0’
870h 87Fh	Common RAM (Accesses 70h – 7Fh)	8F0h 8FFh	Common RAM (Accesses 70h – 7Fh)	970h 97Fh	Common RAM (Accesses 70h – 7Fh)	9F0h 9FFh	Common RAM (Accesses 70h – 7Fh)	A70h A7Fh	Common RAM (Accesses 70h – 7Fh)	AF0h AFFh	Common RAM (Accesses 70h – 7Fh)	B70h B7Fh	Common RAM (Accesses 70h – 7Fh)	BF0h BFFh	Common RAM (Accesses 70h – 7Fh)

Address	BANK 24	Address	BANK 25	Address	BANK 26	Address	BANK 27	Address	BANK 28	Address	BANK 29	Address	BANK 30
C00h C0Bh	Core Registers (Table 3-2)	C80h C8Bh	Core Registers (Table 3-2)	D00h D0Bh	Core Registers (Table 3-2)	D80h D8Bh	Core Registers (Table 3-2)	E00h E0Bh	Core Registers (Table 3-2)	E80h E8Bh	Core Registers (Table 3-2)	F00h F0Bh	Core Registers (Table 3-2)
C0Ch C6Fh	Unimplemented Read as ‘0’	C8Ch CEFh	Unimplemented Read as ‘0’	D0Ch D6Fh	Unimplemented Read as ‘0’	D8Ch DEFh	Unimplemented Read as ‘0’	E0Ch E6Fh	Unimplemented Read as ‘0’	E8Ch EEFh	Unimplemented Read as ‘0’	F0Ch F6Fh	Unimplemented Read as ‘0’
C70h C7Fh	Common RAM (Accesses 70h – 7Fh)	CF0h CFFh	Common RAM (Accesses 70h – 7Fh)	D70h D7Fh	Common RAM (Accesses 70h – 7Fh)	DF0h DFFh	Common RAM (Accesses 70h – 7Fh)	E70h E7Fh	Common RAM (Accesses 70h – 7Fh)	EF0h EFFh	Common RAM (Accesses 70h – 7Fh)	F70h F7Fh	Common RAM (Accesses 70h – 7Fh)

Legend: ▨ = Unimplemented data memory locations, read as ‘0’

TABLE 3-3: PIC16LF1902/3 MEMORY MAP (CONTINUED)

	Bank 15
780h 78Bh	Core Registers (Table 3-2)
78Ch 790h	Unimplemented Read as ‘0’
791h	LCDCON
792h	LCDPS
793h	LCDREF
794h	LCDCST
795h	LCDRL
796h	—
797h	—
798h	LCDSE0
799h	LCDSE1
79Ah	—
79Bh	LCDSE3
79Ch 79Fh	Unimplemented Read as ‘0’
7A0h	LCDDATA0
7A1h	LCDDATA1
7A2h	—
7A3h	LCDDATA3
7A4h	LCDDATA4
7A5h	—
7A6h	LCDDATA6
7A7h	LCDDATA7
7A8h	—
7A9h	LCDDATA9
7AAh	LCDDATA10
7ABh	—
7ACh	LCDDATA12
7ADh	—
7AEh	—
7AFh	LCDDATA15
7B0h	—
7B1h	—
7B2h	LCDDATA18
7B3h	—
7B4h	—
7B5h	LCDDATA21
7B6h	—
7B7h	—
7B8h 7EFh	Unimplemented Read as ‘0’

	Bank 31
F80h F8Bh	Core Registers (Table 3-2)
F8Ch FE3h	Unimplemented Read as ‘0’
FE4h	STATUS_SHAD
FE5h	WREG_SHAD
FE6h	BSR_SHAD
FE7h	PCLATH_SHAD
FE8h	FSR0L_SHAD
FE9h	FSR0H_SHAD
FEAh	FSR1L_SHAD
FEBh	FSR1H_SHAD
FECh	—
FEDh	STKPTR
FEEh	TOSL
FEFh	TOSH
FF0h FFFh	Common RAM (Accesses 70h – 7Fh)

Legend: ▨ = Unimplemented data memory locations, read as ‘0’,

3.2.6 CORE FUNCTION REGISTERS SUMMARY

The Core Function registers listed in Table 3-4 can be addressed from any Bank.

TABLE 3-4: CORE FUNCTION REGISTERS SUMMARY

Addr	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
Bank 0-31											
x00h or x80h	INDF0	Addressing this location uses contents of FSR0H/FSR0L to address data memory (not a physical register)								`xxxx xxxx`	`uuuu uuuu`
x01h or x81h	INDF1	Addressing this location uses contents of FSR1H/FSR1L to address data memory (not a physical register)								`xxxx xxxx`	`uuuu uuuu`
x02h or x82h	PCL	Program Counter (PC) Least Significant Byte								`0000 0000`	`0000 0000`
x03h or x83h	STATUS	—	—	—	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Z	DC	C	`---1 1000`	`---q quuu`
x04h or x84h	FSR0L	Indirect Data Memory Address 0 Low Pointer								`0000 0000`	`uuuu uuuu`
x05h or x85h	FSR0H	Indirect Data Memory Address 0 High Pointer								`0000 0000`	`0000 0000`
x06h or x86h	FSR1L	Indirect Data Memory Address 1 Low Pointer								`0000 0000`	`uuuu uuuu`
x07h or x87h	FSR1H	Indirect Data Memory Address 1 High Pointer								`0000 0000`	`0000 0000`
x08h or x88h	BSR	—	—	—	BSR4	BSR3	BSR2	BSR1	BSR0	`---0 0000`	`---0 0000`
x09h or x89h	WREG	Working Register								`0000 0000`	`uuuu uuuu`
x0Ah or x8Ah	PCLATH	—	Write Buffer for the upper 7 bits of the Program Counter							`-000 0000`	`-000 0000`
x0Bh or x8Bh	INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	`0000 0000`	`0000 0000`

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

TABLE 3-5: SPECIAL FUNCTION REGISTER SUMMARY

Addr	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
Bank 0											
00Ch	PORTA	PORTA Data Latch when written: PORTA pins when read								`xxxx xxxx`	`uuuu uuuu`
00Dh	PORTB	PORTB Data Latch when written: PORTB pins when read								`xxxx xxxx`	`uuuu uuuu`
00Eh	PORTC	PORTC Data Latch when written: PORTC pins when read								`xxxx xxxx`	`uuuu uuuu`
00Fh	—	Unimplemented								—	—
010h	PORTE	—	—	—	—	RE3	—	—	—	`---- x---`	`---- u---`
011h	PIR1	TMR1GIF	ADIF	—	—		—	—	TMR1IF	`00-- ---0`	`0000 ---0`
012h	PIR2	—	—	—	—	—	LCDIF	—	—	`---- -0--`	`---- -0--`
013h	—	Unimplemented								—	—
014h	—	Unimplemented								—	—
015h	TMR0	Timer0 Module Register								`xxxx xxxx`	`uuuu uuuu`
016h	TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								`xxxx xxxx`	`uuuu uuuu`
017h	TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								`xxxx xxxx`	`uuuu uuuu`
018h	T1CON	TMR1CS1	TMR1CS0	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	—	TMR1ON	`0000 00-0`	`uuuu uu-u`
019h	T1GCON	TMR1GE	T1GPOL	T1GTM	T1GSPM	T1GGO/$\overline{\text{DONE}}$	T1GVAL	T1GSS1	T1GSS0	`0000 0x00`	`uuuu uxuu`
01Ah to 01Fh	—	Unimplemented								—	—
Bank 1											
08Ch	TRISA	PORTA Data Direction Register								`1111 1111`	`1111 1111`
08Dh	TRISB	PORTB Data Direction Register								`1111 1111`	`1111 1111`
08Eh	TRISC	PORTC Data Direction Register								`1111 1111`	`1111 1111`
08Fh	—	Unimplemented								—	—
090h	TRISE	—	—	—	—	—(2)	—	—	—	`---- 1---`	`---- 1---`
091h	PIE1	TMR1GIE	ADIE	—	—	—	—	—	TMR1IE	`00-- ---0`	`0000 ---0`
092h	PIE2	—	—	—	—	—	LCDIE	—	—	`---- -0--`	`---- -0--`
093h	—	Unimplemented								—	—
094h	—	Unimplemented								—	—
095h	OPTION_REG	$\overline{\text{WPUEN}}$	INTEDG	TMR0CS	TMR0SE	PSA	PS2	PS1	PS0	`1111 1111`	`1111 1111`
096h	PCON	STKOVF	STKUNF	—	$\overline{\text{RWDT}}$	$\overline{\text{RMCLR}}$	$\overline{\text{RI}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	`00-1 11qq`	`qq-q qquu`
097h	WDTCON	—	—	WDTPS4	WDTPS3	WDTPS2	WDTPS1	WDTPS0	SWDTEN	`--01 0110`	`--01 0110`
098h	—	Unimplemented								—	—
099h	OSCCON	—	IRCF3	IRCF2	IRCF1	IRCF0	—	SCS1	SCS0	`-011 1-00`	`-011 1-00`
09Ah	OSCSTAT	T1OSCR	—	OSTS	HFIOFR	—	—	LFIOFR	HFIOFS	`0-q0 --00`	`q-qq --0q`
09Bh	ADRESL	A/D Result Register Low								`xxxx xxxx`	`uuuu uuuu`
09Ch	ADRESH	A/D Result Register High								`xxxx xxxx`	`uuuu uuuu`
09Dh	ADCON0	—	CHS4	CHS3	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	ADON	`-000 0000`	`-000 0000`
09Eh	ADCON1	ADFM	ADCS2	ADCS1	ADCS0	—	—	ADPREF1	ADPREF0	`0000 ----`	`0000 ----`
09Fh	—	Unimplemented								—	—

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

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

2: Unimplemented, read as ‘1’.

TABLE 3-5: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

Addr	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
Bank 2											
10Ch	LATA	PORTA Data Latch								`xxxx xxxx`	`uuuu uuuu`
10Dh	LATB	PORTB Data Latch								`xxxx xxxx`	`uuuu uuuu`
10Eh	LATC	PORTC Data Latch								`xxxx xxxx`	`uuuu uuuu`
10Fh to 115h	—	Unimplemented								—	—
116h	BORCON	SBOREN	BORFS	—	—	—	—	—	BORRDY	`10-- ---q`	`uu-- ---u`
117h	FVRCON	FVREN	FVRRDY	TSEN	TSRNG	—	—	ADFVR1	ADFVR0	`0q00 --00`	`0q00 --00`
118h to 11Fh	—	Unimplemented								—	—
Bank 3											
18Ch	ANSELA	—	—	ANSA5	—	ANSA3	ANSA2	ANSA1	ANSA0	`--1- 1111`	`--11 1111`
18Dh	ANSELB	—	—	ANSB5	ANSB4	ANSB3	ANSB2	ANSB1	ANSB0	`--11 1111`	`--11 1111`
18Eh	—	Unimplemented								—	—
18Fh	—	Unimplemented								—	—
190h	—	Unimplemented								—	—
191h	PMADRL	Program Memory Address Register Low Byte								`0000 0000`	`0000 0000`
192h	PMADRH	—(2)	Program Memory Address Register High Byte							`1000 0000`	`1000 0000`
193h	PMDATL	Program Memory Read Data Register Low Byte								`xxxx xxxx`	`uuuu uuuu`
194h	PMDATH	—	—	Program Memory Read Data Register High Byte						`--xx xxxx`	`--uu uuuu`
195h	PMCON1	—(2)	CFGS	LWLO	FREE	WRERR	WREN	WR	RD	`1000 x000`	`1000 q000`
196h	PMCON2	Program Memory Control Register 2								`0000 0000`	`0000 0000`
197h to 19Fh	—	Unimplemented								—	—
Bank 4											
20Ch	—	Unimplemented								—	—
20Dh	WPUB	WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0	`1111 1111`	`1111 1111`
20Eh	—	Unimplemented								—	—
20Fh	—	Unimplemented								—	—
210h	WPUE	—	—	—	—	WPUE3	—	—	—	`---- 1---`	`---- 1---`
211h to 21Fh	—	Unimplemented								—	—
Bank 5											
28Ch — 29Fh	—	Unimplemented								—	—
Bank 6											
30Ch — 31Fh	—	Unimplemented								—	—

Legend: `x` = unknown, `u` = unchanged, `q` = value depends on condition, `-` = unimplemented, read as ‘`0`’, `r` = reserved.
Shaded locations are unimplemented, read as ‘`0`’.

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

2: Unimplemented, read as ‘`1`’.

TABLE 3-5: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

Addr	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
Bank 7											
38Ch — 393h	—	Unimplemented								—	—
394h	IOCBP	IOCBP7	IOCBP6	IOCBP5	IOCBP4	IOCBP3	IOCBP2	IOCBP1	IOCBP0	`0000 0000`	`0000 0000`
395h	IOCBN	IOCBN7	IOCBN6	IOCBN5	IOCBN4	IOCBN3	IOCBN2	IOCBN1	IOCBN0	`0000 0000`	`0000 0000`
396h	IOCBF	IOCBF7	IOCBF6	IOCBF5	IOCBF4	IOCBF3	IOCBF2	IOCBF1	IOCBF0	`0000 0000`	`0000 0000`
397h — 39Fh	—	Unimplemented								—	—
Bank 8-14											
x0Ch or x8Ch to x1Fh or x9Fh	—	Unimplemented								—	—
Bank 15											
78Ch — 790h	—	Unimplemented								—	—
791h	LCDCON	LCDEN	SLPEN	WERR	—	CS1	CS0	LMUX1	LMUX0	`000- 0011`	`000- 0011`
792h	LCDPS	WFT	BIASMD	LCDA	WA	LP3	LP2	LP1	LP0	`0000 0000`	`0000 0000`
793h	LCDREF	LCDIRE	—	LCDIRI	—	VLCD3PE	VLCD2PE	VLCD1PE	—	`0-0- 000-`	`0-0- 000-`
794h	LCDCST	—	—	—	—	—	LCDCST2	LCDCST1	LCDCST0	`---- -000`	`---- -000`
795h	LCDRL	LRLAP1	LRLAP0	LRLBP1	LRLBP0	—	LRLAT2	LRLAT1	LRLAT0	`0000 -000`	`0000 -000`
796h	—	Unimplemented								—	—
797h	—	Unimplemented								—	—
798h	LCDSE0	SE7	SE6	SE5	SE4	SE3	SE2	SE1	SE0	`0000 0000`	`uuuu uuuu`
799h	LCDSE1	SE15	SE14	SE13	SE12	SE11	SE10	SE9	SE8	`0000 0000`	`uuuu uuuu`
79Ah	—	Unimplemented								—	—
79Bh	LCDSE3	—	—	—	—	—	SE26	SE25	SE24	`---- -000`	`---- -uuu`
79Dh — 79Fh	—	Unimplemented								—	—
7A0h	LCDDATA0	SEG7 COM0	SEG6 COM0	SEG5 COM0	SEG4 COM0	SEG3 COM0	SEG2 COM0	SEG1 COM0	SEG0 COM0	`xxxx xxxx`	`uuuu uuuu`
7A1h	LCDDATA1	SEG15 COM0	SEG14 COM0	SEG13 COM0	SEG12 COM0	SEG11 COM0	SEG10 COM0	SEG9 COM0	SEG8 COM0	`xxxx xxxx`	`uuuu uuuu`
7A2h	—	Unimplemented								—	—
7A3h	LCDDATA3	SEG7 COM1	SEG6 COM1	SEG5 COM1	SEG4 COM1	SEG3 COM1	SEG2 COM1	SEG1 COM1	SEG0 COM1	`xxxx xxxx`	`uuuu uuuu`
7A4h	LCDDATA4	SEG15 COM1	SEG14 COM1	SEG13 COM1	SEG12 COM1	SEG11 COM1	SEG10 COM1	SEG9 COM1	SEG8 COM1	`xxxx xxxx`	`uuuu uuuu`
7A5h	—	Unimplemented								—	—
7A6h	LCDDATA6	SEG7 COM2	SEG6 COM2	SEG5 COM2	SEG4 COM2	SEG3 COM2	SEG2 COM2	SEG1 COM2	SEG0 COM2	`xxxx xxxx`	`uuuu uuuu`
7A7h	LCDDATA7	SEG15 COM2	SEG14 COM2	SEG13 COM2	SEG12 COM2	SEG11 COM2	SEG10 COM2	SEG9 COM2	SEG8 COM2	`xxxx xxxx`	`uuuu uuuu`
7A8h	—	Unimplemented								—	—
7A9h	LCDDATA9	SEG7 COM3	SEG6 COM3	SEG5 COM3	SEG4 COM3	SEG3 COM3	SEG2 COM3	SEG1 COM3	SEG0 COM3	`xxxx xxxx`	`uuuu uuuu`
7AAh	LCDDATA10	SEG15 COM3	SEG14 COM3	SEG13 COM3	SEG12 COM3	SEG11 COM3	SEG10 COM3	SEG9 COM3	SEG8 COM3	`xxxx xxxx`	`uuuu uuuu`

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

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

2: Unimplemented, read as '`1`'.

TABLE 3-5: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

Addr	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
Bank 15 (Continued)											
7ABh	—	Unimplemented								—	—
7ACh	LCDDATA12	—	—	—	—	—	SEG26 COM0	SEG25 COM0	SEG24 COM0	`---- -xxx`	`---- -uuu`
7ADh	—	Unimplemented								—	—
7AEh	—	Unimplemented								—	—
7AFh	LCDDATA15	—	—	—	—	—	SEG26 COM1	SEG25 COM1	SEG24 COM1	`---- -xxx`	`---- -uuu`
7B0h	—	Unimplemented								—	—
7B1h	—	Unimplemented								—	—
7B2h	LCDDATA18	—	—	—	—	—	SEG26 COM2	SEG25 COM2	SEG24 COM2	`---- -xxx`	`---- -uuu`
7B3h	—	Unimplemented								—	—
7B4h	—	Unimplemented								—	—
7B5h	LCDDATA21	—	—	—	—	—	SEG26 COM3	SEG25 COM3	SEG24 COM3	`---- -xxx`	`---- -uuu`
7B6h — 7EFh	—	Unimplemented								—	—
Bank 16-30											
x0Ch or x8Ch to x1Fh or x9Fh	—	Unimplemented								—	—
Bank 31											
F8Ch — FE3h	—	Unimplemented								—	—
FE4h	STATUS_SHAD	—	—	—	—	—	Z_SHAD	DC_SHAD	C_SHAD	`---- -xxx`	`---- -uuu`
FE5h	WREG_SHAD	Working Register Normal (Non-ICD) Shadow								`xxxx xxxx`	`uuuu uuuu`
FE6h	BSR_SHAD	—	—	—	Bank Select Register Normal (Non-ICD) Shadow					`---x xxxx`	`---u uuuu`
FE7h	PCLATH_SHAD	—	Program Counter Latch High Register Normal (Non-ICD) Shadow							`-xxx xxxx`	`uuuu uuuu`
FE8h	FSR0L_SHAD	Indirect Data Memory Address 0 Low Pointer Normal (Non-ICD) Shadow								`xxxx xxxx`	`uuuu uuuu`
FE9h	FSR0H_SHAD	Indirect Data Memory Address 0 High Pointer Normal (Non-ICD) Shadow								`xxxx xxxx`	`uuuu uuuu`
FEAh	FSR1L_SHAD	Indirect Data Memory Address 1 Low Pointer Normal (Non-ICD) Shadow								`xxxx xxxx`	`uuuu uuuu`
FEBh	FSR1H_SHAD	Indirect Data Memory Address 1 High Pointer Normal (Non-ICD) Shadow								`xxxx xxxx`	`uuuu uuuu`
FECh	—	Unimplemented								—	—
FEDh	STKPTR	—	—	—	Current Stack Pointer					`---1 1111`	`---1 1111`
FEEh	TOSL	Top of Stack Low byte								`xxxx xxxx`	`uuuu uuuu`
FEFh	TOSH	—	Top of Stack High byte							`-xxx xxxx`	`-uuu uuuu`

Legend: `x` = unknown, `u` = unchanged, `q` = value depends on condition, `-` = unimplemented, read as ‘`0`’, `r` = reserved. Shaded locations are unimplemented, read as ‘`0`’.

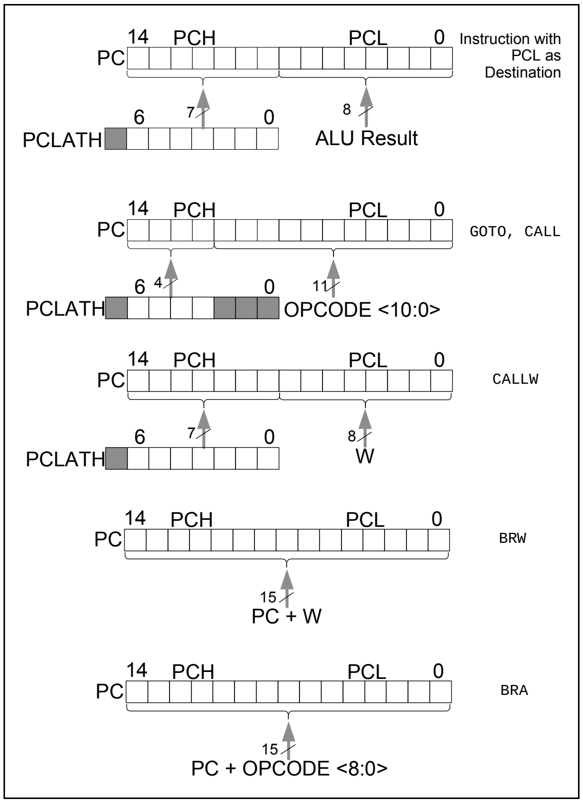
Note 1: These registers can be addressed from any bank.

2: Unimplemented, read as ‘`1`’.

3.3 PCL and PCLATH

The Program Counter (PC) is 15 bits wide. The low byte comes from the PCL register, which is a readable and writable register. The high byte (PC<14:8>) is not directly readable or writable and comes from PCLATH. On any Reset, the PC is cleared. Figure 3-4 shows the five situations for the loading of the PC.

FIGURE 3-4: LOADING OF PC IN DIFFERENT SITUATIONS



3.3.1 MODIFYING PCL

Executing any instruction with the PCL register as the destination simultaneously causes the Program Counter PC<14:8> bits (PCH) to be replaced by the contents of the PCLATH register. This allows the entire contents of the program counter to be changed by writing the desired upper seven bits to the PCLATH register. When the lower eight bits are written to the PCL register, all 15 bits of the program counter will change to the values contained in the PCLATH register and those being written to the PCL register.

3.3.2 COMPUTED `GOTO`

A computed `GOTO` is accomplished by adding an offset to the program counter (`ADDWF PCL`). When performing 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 AN556, *“Implementing a Table Read”* (DS00556).

3.3.3 COMPUTED FUNCTION CALLS

A computed function `CALL` allows programs to maintain tables of functions and provide another way to execute state machines or look-up tables. When performing a table read using a computed function `CALL`, care should be exercised if the table location crosses a PCL memory boundary (each 256-byte block).

If using the `CALL` instruction, the PCH<2:0> and PCL registers are loaded with the operand of the `CALL` instruction. PCH<6:3> is loaded with PCLATH<6:3>.

The `CALLW` instruction enables computed calls by combining PCLATH and W to form the destination address. A computed `CALLW` is accomplished by loading the W register with the desired address and executing `CALLW`. The PCL register is loaded with the value of W and PCH is loaded with PCLATH.

3.3.4 BRANCHING

The branching instructions add an offset to the PC. This allows relocatable code and code that crosses page boundaries. There are two forms of branching, `BRW` and `BRA`. The PC will have incremented to fetch the next instruction in both cases. When using either branching instruction, a PCL memory boundary may be crossed.

If using `BRW`, load the W register with the desired unsigned address and execute `BRW`. The entire PC will be loaded with the address PC + 1 + W.

If using `BRA`, the entire PC will be loaded with PC + 1 +, the signed value of the operand of the `BRA` instruction.

3.4 Stack

All devices have a 16-level x 15-bit wide hardware stack (refer to Figures 3-3 and 3-3). The stack space is not part of either program or data space. The PC is PUSHed onto the stack when `CALL` or `CALLW` instructions are executed or an interrupt causes a branch. The stack is POPed in the event of a `RETURN`, `RETLW` or a `RETFIE` instruction execution. PCLATH is not affected by a PUSH or POP operation.

The stack operates as a circular buffer if the STVREN bit is programmed to '`0`' (Configuration Word 2). This means that after the stack has been PUSHed sixteen times, the seventeenth PUSH overwrites the value that was stored from the first PUSH. The eighteenth PUSH overwrites the second PUSH (and so on). The STKOVF and STKUNF flag bits will be set on an Overflow/Underflow, regardless of whether the Reset is enabled.

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

3.4.1 ACCESSING THE STACK

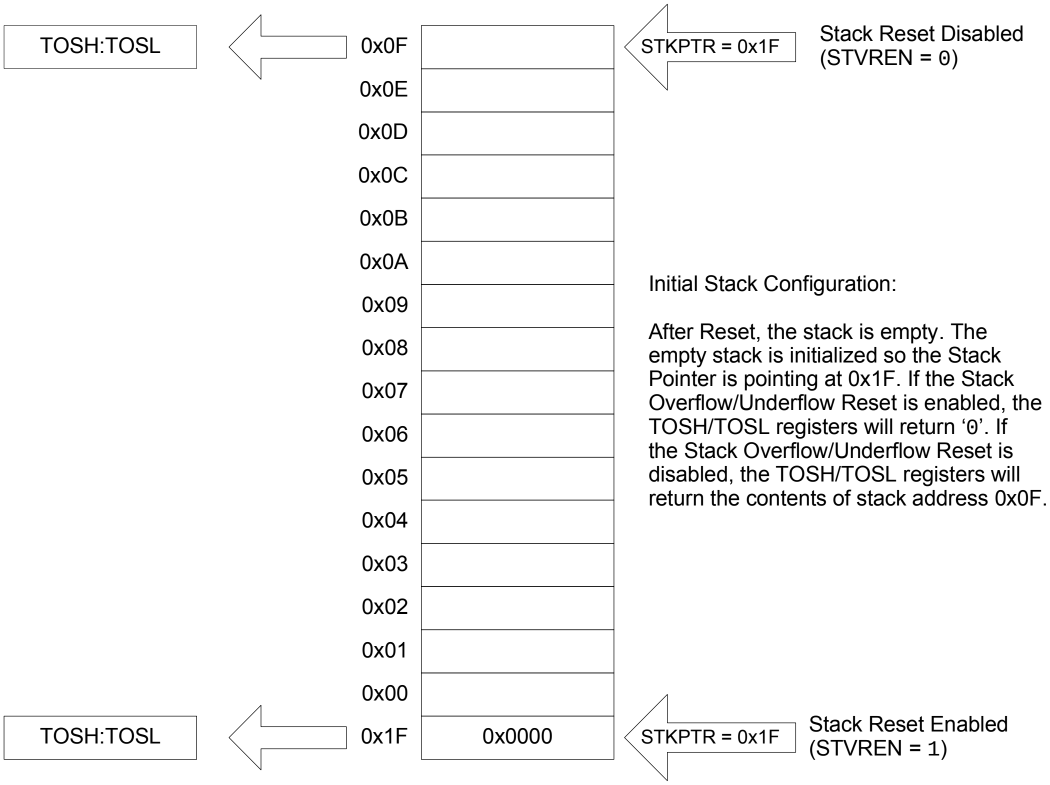
The stack is available through the TOSH, TOSL and STKPTR registers. STKPTR is the current value of the Stack Pointer. TOSH:TOSL register pair points to the TOP of the stack. Both registers are read/writable. TOS is split into TOSH and TOSL due to the 15-bit size of the PC. To access the stack, adjust the value of STKPTR, which will position TOSH:TOSL, then read/write to TOSH:TOSL. STKPTR is five bits to allow detection of overflow and underflow.

> **Note:** Care should be taken when modifying the STKPTR while interrupts are enabled.

During normal program operation, `CALL`, `CALLW` and Interrupts will increment STKPTR while `RETLW`, `RETURN`, and `RETFIE` will decrement STKPTR. At any time STKPTR can be inspected to see how much stack is left. The STKPTR always points at the currently used place on the stack. Therefore, a `CALL` or `CALLW` will increment the STKPTR and then write the PC, and a return will unload the PC and then decrement the STKPTR.

Reference Figure 3-5 through Figure 3-8 for examples of accessing the stack.

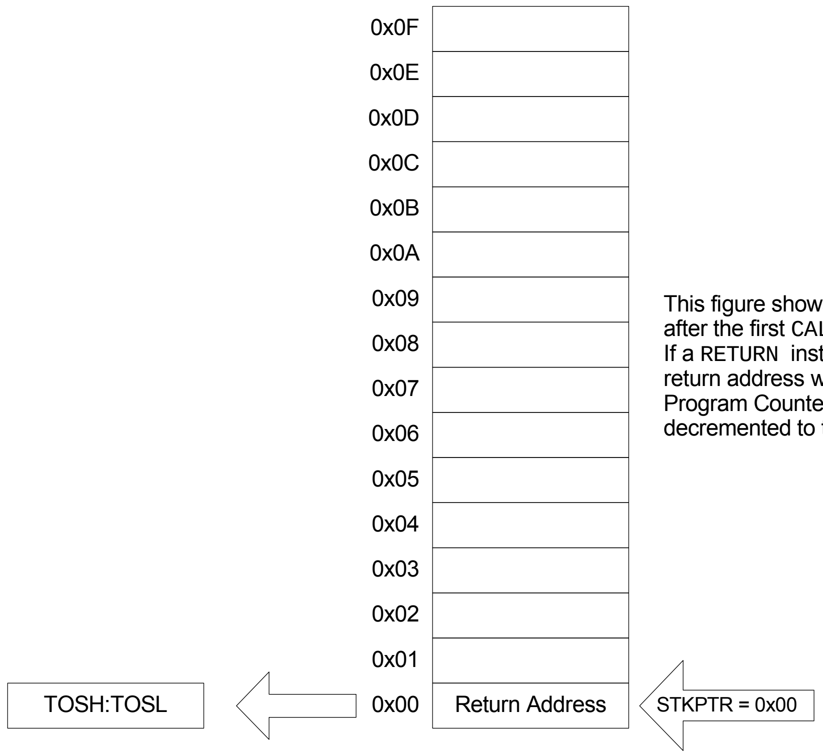
FIGURE 3-5: ACCESSING THE STACK EXAMPLE 1



Initial Stack Configuration:

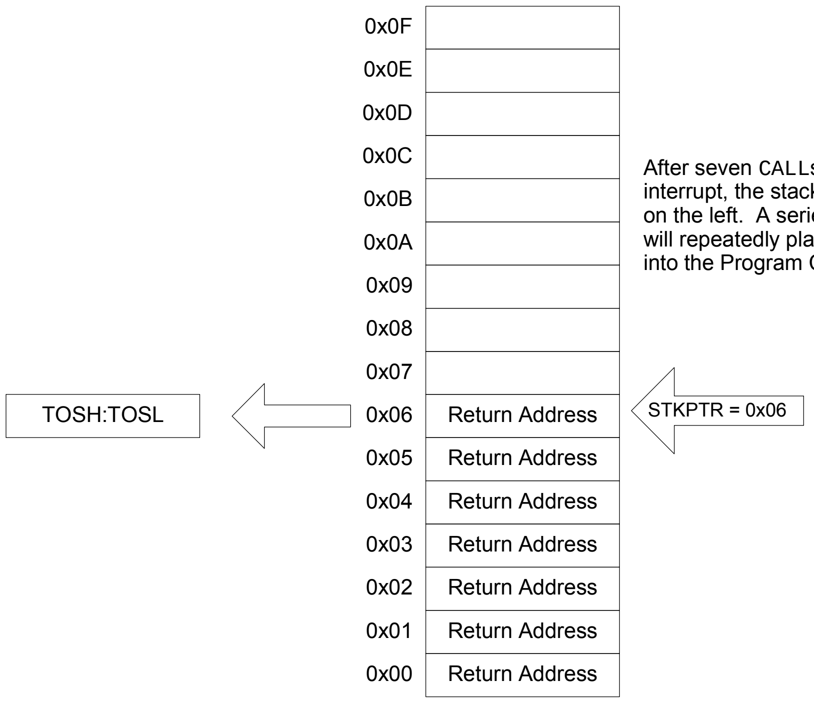
After Reset, the stack is empty. The empty stack is initialized so the Stack Pointer is pointing at 0x1F. If the Stack Overflow/Underflow Reset is enabled, the TOSH/TOSL registers will return '`0`'. If the Stack Overflow/Underflow Reset is disabled, the TOSH/TOSL registers will return the contents of stack address 0x0F.

FIGURE 3-6: ACCESSING THE STACK EXAMPLE 2


0x0F
0x0E
0x0D
0x0C
0x0B
0x0A
0x09
0x08
0x07
0x06
0x05
0x04
0x03
0x02
0x01
0x00
Return Address
TOSH:TOSL
STKPTR = 0x00

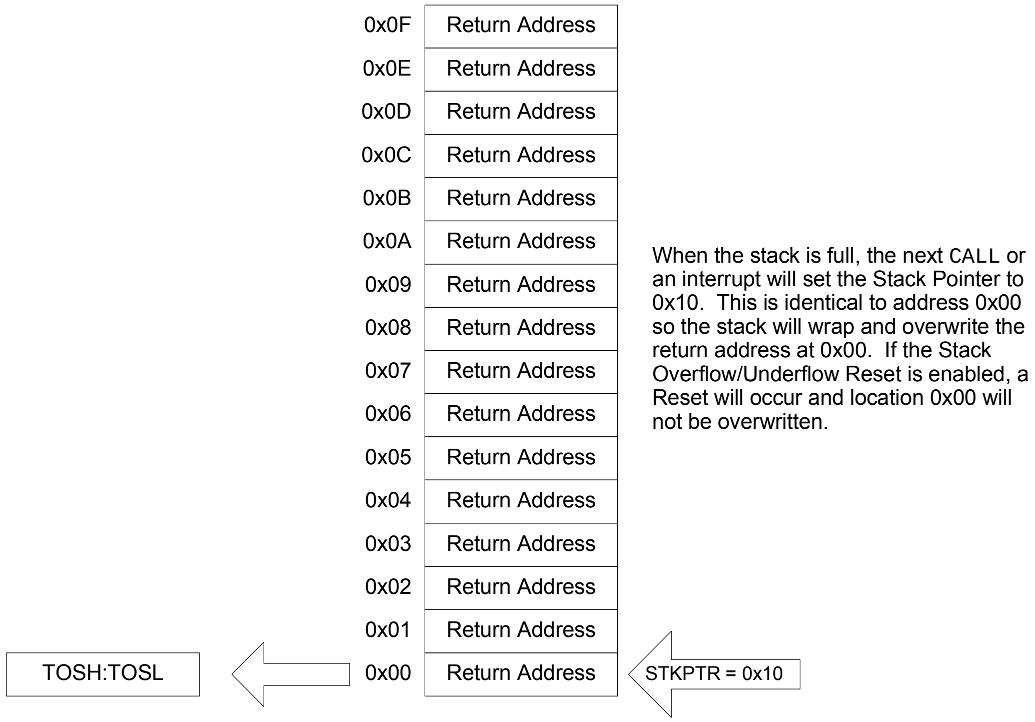
This figure shows the stack configuration after the first `CALL` or a single interrupt. If a `RETURN` instruction is executed, the return address will be placed in the Program Counter and the Stack Pointer decremented to the empty state (0x1F).

FIGURE 3-7: ACCESSING THE STACK EXAMPLE 3


0x0F
0x0E
0x0D
0x0C
0x0B
0x0A
0x09
0x08
0x07
0x06
Return Address
0x05
Return Address
0x04
Return Address
0x03
Return Address
0x02
Return Address
0x01
Return Address
0x00
Return Address
TOSH:TOSL
STKPTR = 0x06

After seven `CALL`s or six `CALL`s and an interrupt, the stack looks like the figure on the left. A series of `RETURN` instructions will repeatedly place the return addresses into the Program Counter and pop the stack.

FIGURE 3-8: ACCESSING THE STACK EXAMPLE 4


0x0F Return Address
0x0E Return Address
0x0D Return Address
0x0C Return Address
0x0B Return Address
0x0A Return Address
0x09 Return Address
0x08 Return Address
0x07 Return Address
0x06 Return Address
0x05 Return Address
0x04 Return Address
0x03 Return Address
0x02 Return Address
0x01 Return Address
0x00 Return Address
TOSH:TOSL
STKPTR = 0x10

When the stack is full, the next `CALL` or an interrupt will set the Stack Pointer to 0x10. This is identical to address 0x00 so the stack will wrap and overwrite the return address at 0x00. If the Stack Overflow/Underflow Reset is enabled, a Reset will occur and location 0x00 will not be overwritten.

3.4.2 OVERFLOW/UNDERFLOW RESET

If the STVREN bit in Configuration Word 2 is programmed to ‘`1`’, the device will be reset if the stack is PUSHed beyond the sixteenth level or POPed beyond the first level, setting the appropriate bits (STKOVF or STKUNF, respectively) in the PCON register.

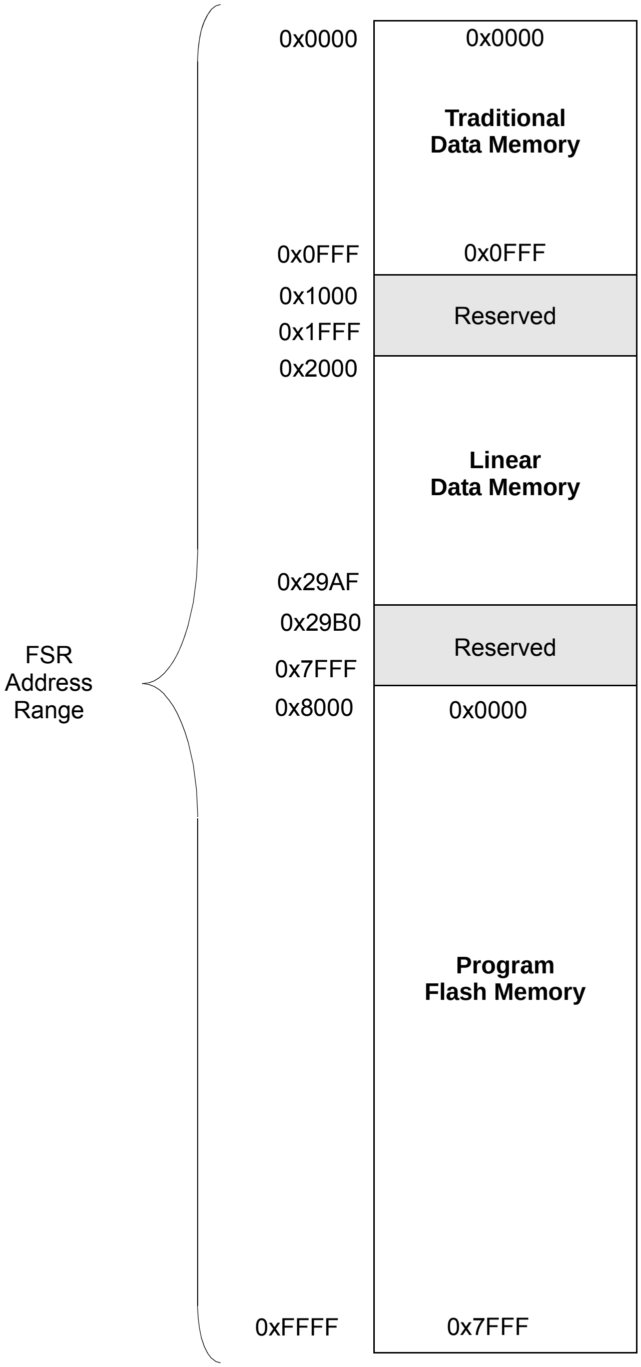
3.5 Indirect Addressing

The INDFn registers are not physical registers. Any instruction that accesses an INDFn register actually accesses the register at the address specified by the File Select Registers (FSR). If the FSRn address specifies one of the two INDFn registers, the read will return ‘`0`’ and the write will not occur (though Status bits may be affected). The FSRn register value is created by the pair FSRnH and FSRnL.

The FSR registers form a 16-bit address that allows an addressing space with 65536 locations. These locations are divided into three memory regions:

- Traditional Data Memory
- Linear Data Memory
- Program Flash Memory

FIGURE 3-9: INDIRECT ADDRESSING

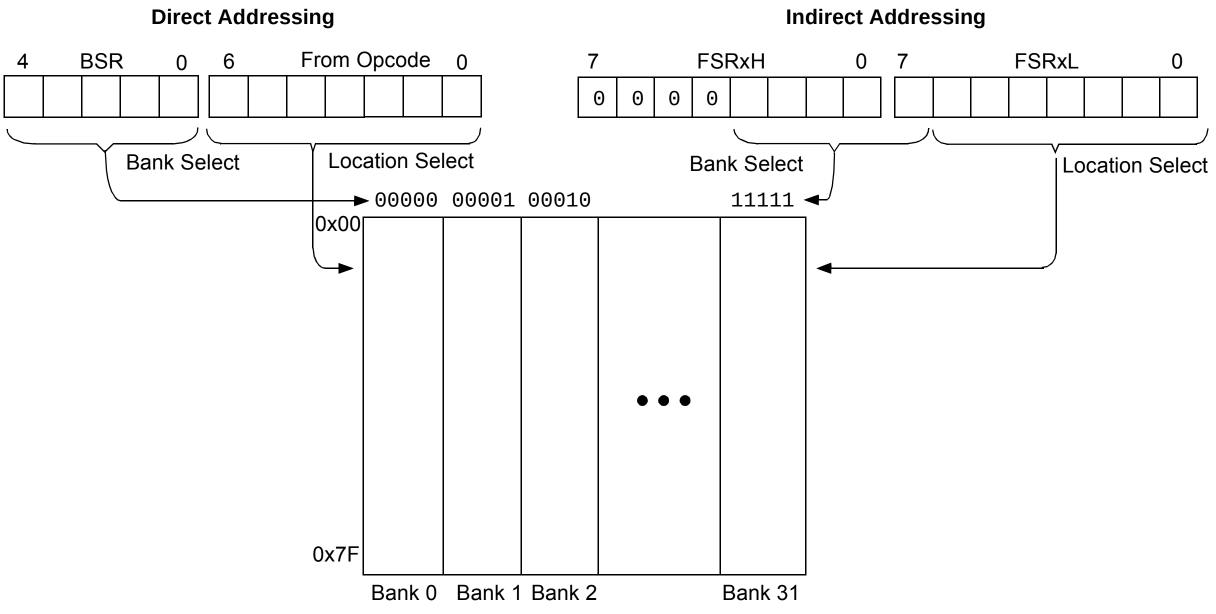

0x0000
0x0000
Traditional
Data Memory
0x0FFF
0x0FFF
0x1000
Reserved
0x1FFF
0x2000
Linear
Data Memory
0x29AF
0x29B0
Reserved
0x7FFF
FSR
Address
Range
0x8000
0x0000
Program
Flash Memory
0xFFFF
0x7FFF

Note: Not all memory regions are completely implemented. Consult device memory tables for memory limits.

3.5.1 TRADITIONAL DATA MEMORY

The traditional data memory is a region from FSR address 0x000 to FSR address 0xFFF. The addresses correspond to the absolute addresses of all SFR, GPR and common registers.

FIGURE 3-10: TRADITIONAL DATA MEMORY MAP


Direct Addressing
4 BSR 0
6 From Opcode 0
Bank Select
Location Select
Indirect Addressing
7 FSRxH 0
0 0 0 0
7 FSRxL 0
Bank Select
Location Select
00000 00001 00010
11111
0x00
0x7F
Bank 0
Bank 1
Bank 2
Bank 31

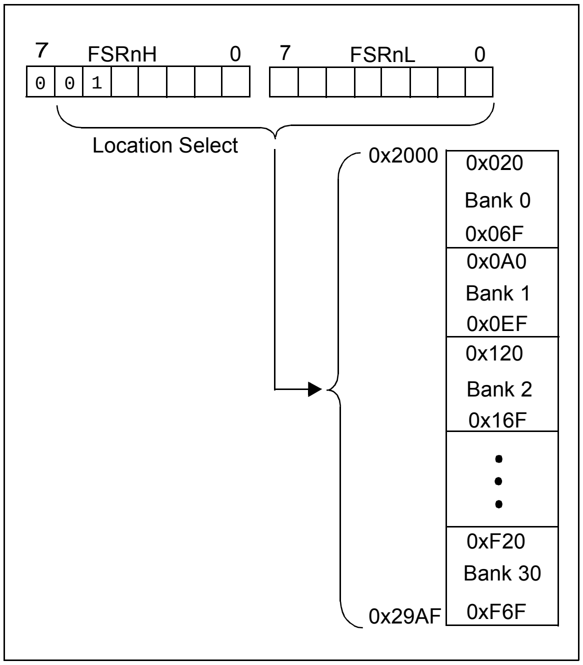
3.5.2 LINEAR DATA MEMORY

The linear data memory is the region from FSR address 0x2000 to FSR address 0x29AF. This region is a virtual region that points back to the 80-byte blocks of GPR memory in all the banks.

Unimplemented memory reads as 0x00. Use of the linear data memory region allows buffers to be larger than 80 bytes because incrementing the FSR beyond one bank will go directly to the GPR memory of the next bank.

The 16 bytes of common memory are not included in the linear data memory region.

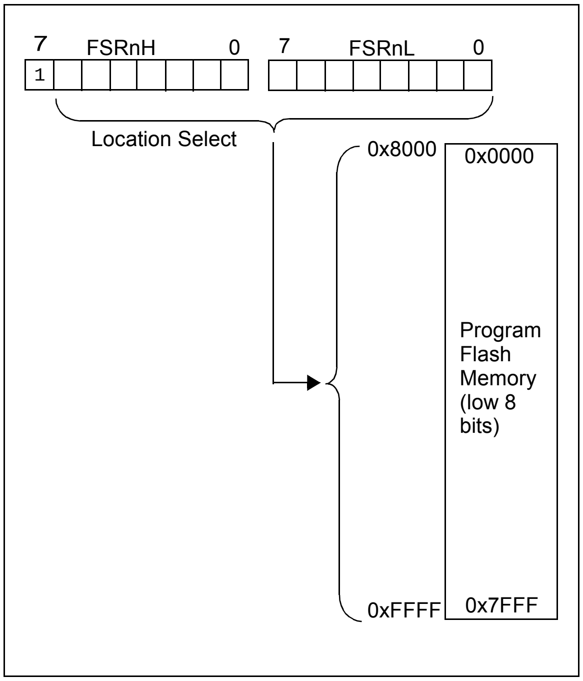
FIGURE 3-11: LINEAR DATA MEMORY MAP


7
FSRnH
0
7
FSRnL
0
0 0 1
Location Select
0x2000
0x29AF
0x020
Bank 0
0x06F
0x0A0
Bank 1
0x0EF
0x120
Bank 2
0x16F
0xF20
Bank 30
0xF6F

3.5.3 PROGRAM FLASH MEMORY

To make constant data access easier, the entire program Flash memory is mapped to the upper half of the FSR address space. When the MSB of FSRnH is set, the lower 15 bits are the address in program memory which will be accessed through INDF. Only the lower eight bits of each memory location is accessible via INDF. Writing to the program Flash memory cannot be accomplished via the FSR/INDF interface. All instructions that access program Flash memory via the FSR/INDF interface will require one additional instruction cycle to complete.

FIGURE 3-12: PROGRAM FLASH MEMORY MAP


7
FSRnH
0
7
FSRnL
0
1
Location Select
0x8000
0xFFFF
0x0000
Program Flash Memory (low 8 bits)
0x7FFF

4.0 DEVICE CONFIGURATION

Device Configuration consists of Configuration Word 1 and Configuration Word 2, Code Protection and Device ID.

4.1 Configuration Words

There are several Configuration Word bits that allow different oscillator and memory protection options. These are implemented as Configuration Word 1 at 8007h and Configuration Word 2 at 8008h.

Note: The $\overline{\text{DEBUG}}$ bit in Configuration Words is managed automatically by device development tools including debuggers and programmers. For normal device operation, this bit should be maintained as a ‘1’.

REGISTER 4-1: CONFIGURATION WORD 1

U-1	U-1	R/P-1	R/P-1	R/P-1	U-1
—	—	$\overline{\text{CLKOUTEN}}$	BOREN<1:0>		—
bit 13					bit 8

R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	U-1	R/P-1	R/P-1
$\overline{\text{CP}}$	MCLRE	$\overline{\text{PWRTE}}$	WDTE<1:0>		—	FOSC<1:0>	
bit 7							bit 0

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as ‘1’
‘0’ = Bit is cleared	‘1’ = Bit is set	-n = Value when blank or after Bulk Erase

bit 13-12	**Unimplemented:** Read as ‘1’
bit 11	**$\overline{\text{CLKOUTEN}}$:** Clock Out Enable bit 1 = CLKOUT function is disabled. I/O function on the CLKOUT pin. 0 = CLKOUT function is enabled on the CLKOUT pin
bit 10-9	**BOREN<1:0>:** Brown-out Reset Enable bits 11 = BOR enabled 10 = BOR enabled during operation and disabled in Sleep 01 = BOR controlled by SBOREN bit of the BORCON register 00 = BOR disabled
bit 8	**Unimplemented:** Read as ‘1’
bit 7	**$\overline{\text{CP}}$:** Code Protection bit 1 = Program memory code protection is disabled 0 = Program memory code protection is enabled
bit 6	**MCLRE:** $\overline{\text{MCLR}}$/VPP Pin Function Select bit If LVP bit = 1: This bit is ignored. If LVP bit = 0: 1 = $\overline{\text{MCLR}}$/VPP pin function is $\overline{\text{MCLR}}$; Weak pull-up enabled. 0 = $\overline{\text{MCLR}}$/VPP pin function is digital input; $\overline{\text{MCLR}}$ internally disabled; Weak pull-up under control of WPUE3 bit.
bit 5	**$\overline{\text{PWRTE}}$:** Power-up Timer Enable bit 1 = PWRT disabled 0 = PWRT enabled
bit 4-3	**WDTE<1:0>:** Watchdog Timer Enable bit 11 = WDT enabled 10 = WDT enabled while running and disabled in Sleep 01 = WDT controlled by the SWDTEN bit in the WDTCON register 00 = WDT disabled
bit 2	**Unimplemented:** Read as ‘1’
bit 1-0	**FOSC<1:0>:** Oscillator Selection bits 00 = INTOSC oscillator: I/O function on CLKIN pin 01 = ECL: External Clock, Low-Power mode (0-0.5 MHz): device clock supplied to CLKIN pin 10 = ECM: External Clock, Medium-Power mode (0.5-4 MHz): device clock supplied to CLKIN pin 11 = ECH: External Clock, High-Power mode (4-32 MHz): device clock supplied to CLKIN pin

REGISTER 4-2: CONFIGURATION WORD 2

R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	U-1
LVP[1]	$\overline{\text{DEBUG}}$[2]	$\overline{\text{LPBOR}}$	BORV[3]	STVREN	—
bit 13					bit 8

U-1	U-1	U-1	U-1	U-1	U-1	R/P-1	R/P-1
—	—	—	—	—	—	WRT<1:0>	
bit 7							bit 0

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as ‘1’
‘0’ = Bit is cleared	‘1’ = Bit is set	-n = Value when blank or after Bulk Erase

bit 13 **LVP:** Low-Voltage Programming Enable bit[1]
`1` = Low-voltage programming enabled
`0` = High-voltage on $\overline{\text{MCLR}}$ must be used for programming

bit 12 **$\overline{\text{DEBUG}}$:** In-Circuit Debugger Mode bit[2]
`1` = In-Circuit Debugger disabled, ICSPCLK and ICSPDAT are general purpose I/O pins
`0` = In-Circuit Debugger enabled, ICSPCLK and ICSPDAT are dedicated to the debugger

bit 11 **$\overline{\text{LPBOR}}$:** Low-Power BOR bit
`1` = Low-Power BOR is disabled
`0` = Low-Power BOR is enabled

bit 10 **BORV:** Brown-out Reset Voltage Selection bit[3]
`1` = Brown-out Reset voltage (V_{BOR}), low trip point selected
`0` = Brown-out Reset voltage (V_{BOR}) high trip point selected

bit 9 **STVREN:** Stack Overflow/Underflow Reset Enable bit
`1` = Stack Overflow or Underflow will cause a Reset
`0` = Stack Overflow or Underflow will not cause a Reset

bit 8-2 **Unimplemented:** Read as ‘`1`’

bit 1-0 **WRT<1:0>:** Flash Memory Self-Write Protection bits
<u>2 kW Flash memory (PIC16LF1902 only)</u>:
`11` = Write protection off
`10` = 000h to 1FFh write-protected, 200h to 7FFh may be modified by PMCON control
`01` = 000h to 3FFh write-protected, 400h to 7FFh may be modified by PMCON control
`00` = 000h to 7FFh write-protected, no addresses may be modified by PMCON control
<u>4 kW Flash memory (PIC16LF1903 only)</u>:
`11` = Write protection off
`10` = 000h to 1FFh write-protected, 200h to FFFh may be modified by PMCON control
`01` = 000h to 7FFh write-protected, 800h to FFFh may be modified by PMCON control
`00` = 000h to FFFh write-protected, no addresses may be modified by PMCON control

Note 1: The LVP bit cannot be programmed to ‘`0`’ when Programming mode is entered via LVP.
2: The $\overline{\text{DEBUG}}$ bit in Configuration Words is managed automatically by device development tools including debuggers and programmers. For normal device operation, this bit should be maintained as a ‘`1`’.
3: See V_{BOR} parameter for specific trip point voltages.

4.2 Code Protection

Code protection allows the device to be protected from unauthorized access. Program memory protection is controlled independently. Internal access to the program memory is unaffected by any code protection setting.

4.2.1 PROGRAM MEMORY PROTECTION

The entire program memory space is protected from external reads and writes by the $\overline{\text{CP}}$ bit in Configuration Word 1. When $\overline{\text{CP}}$ = `0`, external reads and writes of program memory are inhibited and a read will return all ‘`0`’s. The CPU can continue to read program memory, regardless of the protection bit settings. Writing the program memory is dependent upon the write protection setting. See **Section 4.3 “Write Protection”** for more information.

4.3 Write Protection

Write protection allows the device to be protected from unintended self-writes. Applications, such as boot loader software, can be protected while allowing other regions of the program memory to be modified.

The WRT<1:0> bits in Configuration Word 2 define the size of the program memory block that is protected.

4.4 User ID

Four memory locations (8000h-8003h) are designated as ID locations where the user can store checksum or other code identification numbers. These locations are readable and writable during normal execution. See **Section 10.4 “User ID, Device ID and Configuration Word Access”** for more information on accessing these memory locations. For more information on checksum calculation, see the **“***PIC16F193X/LF193X/PIC16F194X/LF194X/PIC16LF190X Memory Programming Specification***”** (DS41397).

4.5 Device ID and Revision ID

The memory location 8006h is where the Device ID and Revision ID are stored. The upper nine bits hold the Device ID. The lower five bits hold the Revision ID. See **Section 10.4 "User ID, Device ID and Configuration Word Access"** for more information on accessing these memory locations.

Development tools, such as device programmers and debuggers, may be used to read the Device ID and Revision ID.

REGISTER 4-3: DEVICEID: DEVICE ID REGISTER

R	R	R	R	R	R
DEV<8:3>					
bit 13					bit 8

R	R	R	R	R	R	R	R
DEV<2:0>			REV<4:0>				
bit 7							bit 0

Legend:

R = Readable bit
W = Writable bit
U = Unimplemented bit, read as '1'
u = Bit is unchanged
x = Bit is unknown
-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set
'0' = Bit is cleared
P = Programmable bit

bit 13-5 **DEV<8:0>:** Device ID bits

Device	DEVICEID<13:0> Values	
	DEV<8:0>	REV<4:0>
PIC16LF1902	`01 1100 001`	`x xxxx`
PIC16LF1903	`01 1100 000`	`x xxxx`

bit 4-0 **REV<4:0>:** Revision ID bits

These bits are used to identify the revision (see Table under DEV<8:0> above).

5.0 RESETS

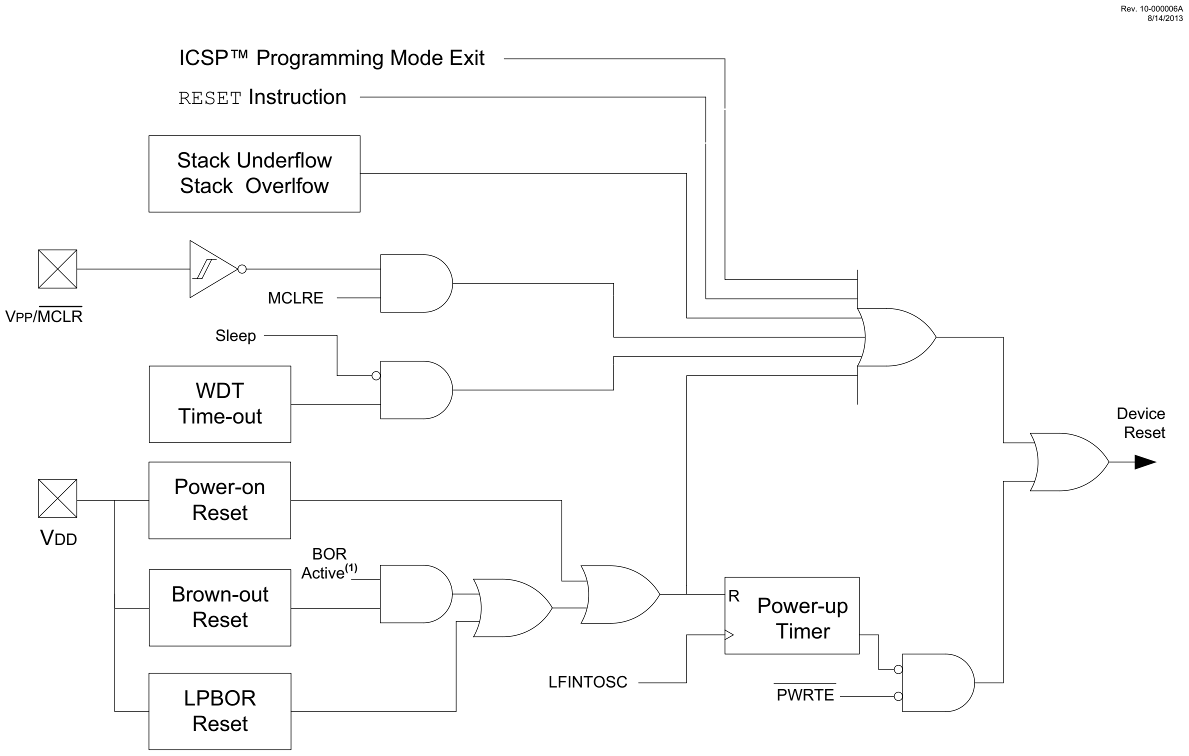
There are multiple ways to reset this device:

- Power-on Reset (POR)
- Brown-out Reset (BOR)
- Low-Power Brown-out Reset (LPBOR)
- $\overline{\text{MCLR}}$ Reset
- WDT Reset
- `RESET` instruction
- Stack Overflow
- Stack Underflow
- Programming mode exit

To allow V_{DD} to stabilize, an optional power-up timer can be enabled to extend the Reset time after a BOR or POR event.

A simplified block diagram of the On-Chip Reset Circuit is shown in Figure 5-1.

FIGURE 5-1: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT


Rev. 10-000006A
8/14/2013
ICSP™ Programming Mode Exit
RESET Instruction
Stack Underflow
Stack Overflow
VPP/MCLR
MCLRE
Sleep
WDT
Time-out
Power-on
Reset
VDD
BOR
Active(1)
Brown-out
Reset
LPBOR
Reset
LFINTOSC
R
Power-up
Timer
PWRTE
Device
Reset

Note 1: See Table 5-1 for BOR active conditions.

5.1 Power-on Reset (POR)

The POR circuit holds the device in Reset until VDD has reached an acceptable level for minimum operation. Slow rising VDD, fast operating speeds or analog performance may require greater than minimum VDD. The PWRT, BOR or $\overline{\text{MCLR}}$ features can be used to extend the start-up period until all device operation conditions have been met.

5.1.1 POWER-UP TIMER (PWRT)

The Power-up Timer provides a nominal 64 ms time-out on POR or Brown-out Reset.

The device is held in Reset as long as PWRT is active. The PWRT delay allows additional time for the VDD to rise to an acceptable level. The Power-up Timer is enabled by clearing the $\overline{\text{PWRTE}}$ bit in Configuration Word 1.

The Power-up Timer starts after the release of the POR and BOR.

For additional information, refer to Application Note AN607, *“Power-up Trouble Shooting”* (DS00607).

5.2 Brown-Out Reset (BOR)

The BOR circuit holds the device in Reset when VDD reaches a selectable minimum level. Between the POR and BOR, complete voltage range coverage for execution protection can be implemented.

The Brown-out Reset module has four operating modes controlled by the BOREN<1:0> bits in Configuration Word 1. The four operating modes are:

- BOR is always on
- BOR is off when in Sleep
- BOR is controlled by software
- BOR is always off

Refer to Table 5-1 for more information.

The Brown-out Reset voltage level is selectable by configuring the BORV bit in Configuration Word 2.

A VDD noise rejection filter prevents the BOR from triggering on small events. If VDD falls below VBOR for a duration greater than parameter TBORDC, the device will reset. See Figure 5-2 for more information.

TABLE 5-1: BOR OPERATING MODES

BOREN<1:0>	SBOREN	Device Mode	BOR Mode	Device Operation upon release of POR	Device Operation upon wake- up from Sleep
`11`	X	X	Active	Waits for BOR ready[1]	
`10`	X	Awake	Active	Waits for BOR ready	
		Sleep	Disabled		
`01`	`1`	X	Active	Waits for BOR ready[1]	
	`0`	X	Disabled	Begins immediately	
`00`	X	X	Disabled	Begins immediately	

Note 1: In these specific cases, “Release of POR” and “Wake-up from Sleep”, there is no delay in start-up. The BOR ready flag, (BORRDY = `1`), will be set before the CPU is ready to execute instructions because the BOR circuit is forced on by the BOREN<1:0> bits.

5.2.1 BOR IS ALWAYS ON

When the BOREN bits of Configuration Word 1 are set to ‘`11`’, the BOR is always on. The device start-up will be delayed until the BOR is ready and VDD is higher than the BOR threshold.

BOR protection is active during Sleep. The BOR does not delay wake-up from Sleep.

5.2.2 BOR IS OFF IN SLEEP

When the BOREN bits of Configuration Word 1 are set to ‘`10`’, the BOR is on, except in Sleep. The device start-up will be delayed until the BOR is ready and VDD is higher than the BOR threshold.

BOR protection is not active during Sleep. The device wake-up will be delayed until the BOR is ready.

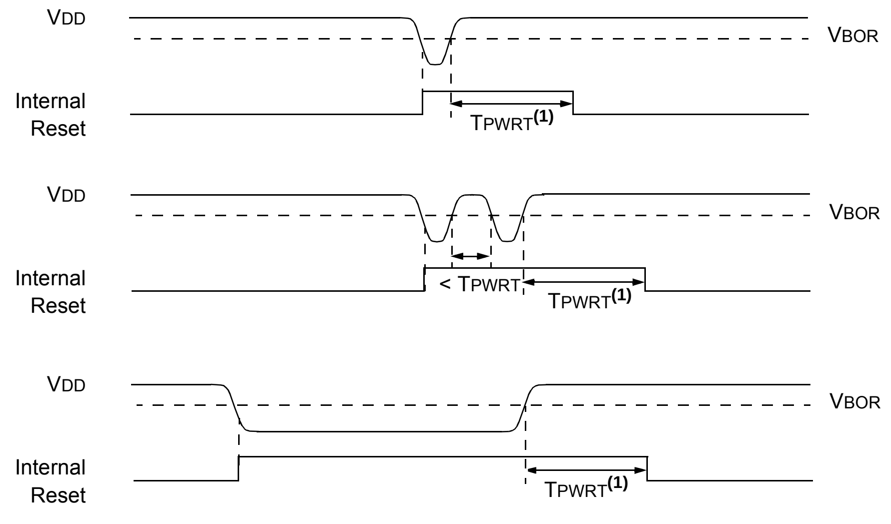
5.2.3 BOR CONTROLLED BY SOFTWARE

When the BOREN bits of Configuration Word 1 are set to ‘`01`’, the BOR is controlled by the SBOREN bit of the BORCON register. The device start-up is not delayed by the BOR ready condition or the VDD level.

BOR protection begins as soon as the BOR circuit is ready. The status of the BOR circuit is reflected in the BORRDY bit of the BORCON register.

BOR protection is unchanged by Sleep.

FIGURE 5-2: BROWN-OUT SITUATIONS



Note 1: TPWRT delay only if $\overline{\text{PWRTE}}$ bit is programmed to '`0`'.

REGISTER 5-1: BORCON: BROWN-OUT RESET CONTROL REGISTER

R/W-1/u	R/W-0/u	U-0	U-0	U-0	U-0	U-0	R-q/u
SBOREN	BORFS	—	—	—	—	—	BORRDY
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	q = Value depends on condition

bit 7 **SBOREN:** Software Brown-out Reset Enable bit

If BOREN <1:0> in Configuration Word 1 ≠ `01`:
SBOREN is read/write, but has no effect on the BOR.
If BOREN <1:0> in Configuration Word 1 = `01`:
`1` = BOR Enabled
`0` = BOR Disabled

bit 6 **BORFS:** Brown-out Reset Fast Start bit[1]

If BOREN<1:0> = `11` (Always on) or BOREN<1:0> = `00` (Always off)
BORFS is Read/Write, but has no effect.
If BOREN <1:0> = `10` (Disabled in Sleep) or BOREN<1:0> = `01` (Under software control):
`1` = Band gap is forced on always (covers sleep/wake-up/operating cases)
`0` = Band gap operates normally, and may turn off

bit 5-1 **Unimplemented:** Read as '`0`'

bit 0 **BORRDY:** Brown-out Reset Circuit Ready Status bit

`1` = The Brown-out Reset circuit is active
`0` = The Brown-out Reset circuit is inactive

5.3 Low-Power Brown-out Reset (LPBOR)

The Low-Power Brown-Out Reset (LPBOR) is an essential part of the Reset subsystem. Refer to Figure 5-1 to see how the BOR interacts with other modules.

The LPBOR is used to monitor the external VDD pin. When too low of a voltage is detected, the device is held in Reset. When this occurs, a register bit ($\overline{\text{BOR}}$) is changed to indicate that a BOR Reset has occurred. The same bit is set for both the BOR and the LPBOR. Refer to Register 5-2.

5.3.1 ENABLING LPBOR

The LPBOR is controlled by the $\overline{\text{LPBOR}}$ bit of Configuration Word 2. When the device is erased, the LPBOR module defaults to disabled.

5.3.1.1 LPBOR Module Output

The output of the LPBOR module is a signal indicating whether or not a Reset is to be asserted. This signal is to be OR'd together with the Reset signal of the BOR module to provide the generic $\overline{\text{BOR}}$ signal which goes to the PCON register and to the power control block.

5.4 $\overline{\text{MCLR}}$

The $\overline{\text{MCLR}}$ is an optional external input that can reset the device. The $\overline{\text{MCLR}}$ function is controlled by the MCLRE bit of Configuration Word 1 and the LVP bit of Configuration Word 2 (Table 5-2).

TABLE 5-2: $\overline{\text{MCLR}}$ CONFIGURATION

MCLRE	LVP	$\overline{\text{MCLR}}$
`0`	`0`	Disabled
`1`	`0`	Enabled
`x`	`1`	Enabled

5.4.1 $\overline{\text{MCLR}}$ ENABLED

When $\overline{\text{MCLR}}$ is enabled and the pin is held low, the device is held in Reset. The $\overline{\text{MCLR}}$ pin is connected to VDD through an internal weak pull-up.

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

Note: A Reset does not drive the $\overline{\text{MCLR}}$ pin low.

5.4.2 $\overline{\text{MCLR}}$ DISABLED

When $\overline{\text{MCLR}}$ is disabled, the pin functions as a general purpose input and the internal weak pull-up is under software control. See **Section 11.4 "PORTE Registers"** for more information.

5.5 Watchdog Timer (WDT) Reset

The Watchdog Timer generates a Reset if the firmware does not issue a `CLRWDT` instruction within the time-out period. The $\overline{\text{TO}}$ and $\overline{\text{PD}}$ bits in the STATUS register are changed to indicate the WDT Reset. See **Section 9.0 "Watchdog Timer"** for more information.

5.6 RESET Instruction

A `RESET` instruction will cause a device Reset. The $\overline{\text{RI}}$ bit in the PCON register will be set to '`0`'. See Table 5-4 for default conditions after a `RESET` instruction has occurred.

5.7 Stack Overflow/Underflow Reset

The device can reset when the Stack Overflows or Underflows. The STKOVF or STKUNF bits of the PCON register indicate the Reset condition. These Resets are enabled by setting the STVREN bit in Configuration Word 2. See **Section 5.7 "Stack Overflow/Underflow Reset"** for more information.

5.8 Programming Mode Exit

Upon exit of Programming mode, the device will behave as if a POR had just occurred.

5.9 Power-Up Timer

The Power-up Timer optionally delays device execution after a BOR or POR event. This timer is typically used to allow VDD to stabilize before allowing the device to start running.

The Power-up Timer is controlled by the $\overline{\text{PWRTE}}$ bit of Configuration Word 1.

5.10 Start-up Sequence

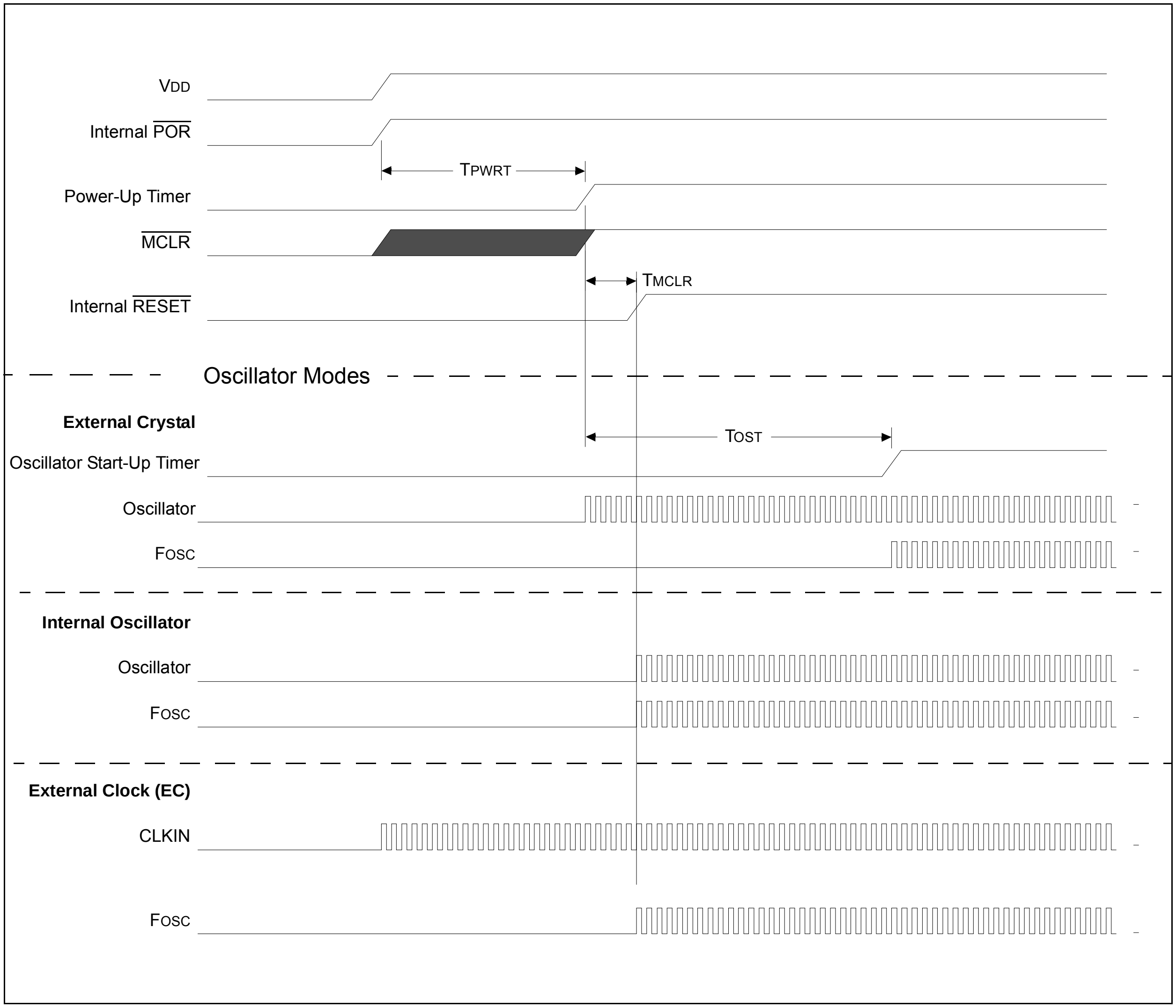
Upon the release of a POR or BOR, the following must occur before the device will begin executing:

1. Power-up Timer runs to completion (if enabled).
2. Oscillator start-up timer runs to completion (if required for oscillator source).
3. $\overline{\text{MCLR}}$ must be released (if enabled).

The total time-out will vary based on oscillator configuration and Power-up Timer configuration. See **Section 6.0 "Oscillator Module"** for more information.

The Power-up Timer and oscillator start-up timer run independently of $\overline{\text{MCLR}}$ Reset. If $\overline{\text{MCLR}}$ is kept low long enough, the Power-up Timer and oscillator start-up timer will expire. Upon bringing $\overline{\text{MCLR}}$ high, the device will begin execution immediately (see Figure 5-3). This is useful for testing purposes or to synchronize more than one device operating in parallel.

FIGURE 5-3: RESET START-UP SEQUENCE


VDD
Internal POR
TPWRT
Power-Up Timer
MCLR
TMCLR
Internal RESET
Oscillator Modes
External Crystal
TOST
Oscillator Start-Up Timer
Oscillator
FOSC
Internal Oscillator
Oscillator
FOSC
External Clock (EC)
CLKIN
FOSC

5.11 Determining the Cause of a Reset

Upon any Reset, multiple bits in the STATUS and PCON register are updated to indicate the cause of the Reset. Table 5-3 and Table 5-4 show the Reset conditions of these registers.

TABLE 5-3: RESET STATUS BITS AND THEIR SIGNIFICANCE

STKOVF	STKUNF	$\overline{\text{RWDT}}$	$\overline{\text{RMCLR}}$	$\overline{\text{RI}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Condition
`0`	`0`	`1`	`1`	`1`	`0`	`x`	`1`	`1`	Power-on Reset
`0`	`0`	`1`	`1`	`1`	`0`	`x`	`0`	`x`	Illegal, $\overline{\text{TO}}$ is set on $\overline{\text{POR}}$
`0`	`0`	`1`	`1`	`1`	`0`	`x`	`x`	`0`	Illegal, $\overline{\text{PD}}$ is set on $\overline{\text{POR}}$
`0`	`0`	`u`	`1`	`1`	`u`	`0`	`1`	`1`	Brown-out Reset
`u`	`u`	`0`	`u`	`u`	`u`	`u`	`0`	`u`	WDT Reset
`u`	`u`	`u`	`u`	`u`	`u`	`u`	`0`	`0`	WDT Wake-up from Sleep
`u`	`u`	`u`	`u`	`u`	`u`	`u`	`1`	`0`	Interrupt Wake-up from Sleep
`u`	`u`	`u`	`0`	`u`	`u`	`u`	`u`	`u`	$\overline{\text{MCLR}}$ Reset during normal operation
`u`	`u`	`u`	`0`	`u`	`u`	`u`	`1`	`0`	$\overline{\text{MCLR}}$ Reset during Sleep
`u`	`u`	`u`	`u`	`0`	`u`	`u`	`u`	`u`	RESET Instruction Executed
`1`	`u`	`u`	`u`	`u`	`u`	`u`	`u`	`u`	Stack Overflow Reset (STVREN = `1`)
`u`	`1`	`u`	`u`	`u`	`u`	`u`	`u`	`u`	Stack Underflow Reset (STVREN = `1`)

TABLE 5-4: RESET CONDITION FOR SPECIAL REGISTERS(2)

Condition	Program Counter	STATUS Register	PCON Register
Power-on Reset	0000h	`---1 1000`	`00-1 110x`
$\overline{\text{MCLR}}$ Reset during normal operation	0000h	`---u uuuu`	`uu-u 0uuu`
$\overline{\text{MCLR}}$ Reset during Sleep	0000h	`---1 0uuu`	`uu-u 0uuu`
WDT Reset	0000h	`---0 uuuu`	`uu-0 uuuu`
WDT Wake-up from Sleep	PC + 1	`---0 0uuu`	`uu-u uuuu`
Brown-out Reset	0000h	`---1 1uuu`	`00-1 11u0`
Interrupt Wake-up from Sleep	PC + 1(1)	`---1 0uuu`	`uu-u uuuu`
RESET Instruction Executed	0000h	`---u uuuu`	`uu-u u0uu`
Stack Overflow Reset (STVREN = `1`)	0000h	`---u uuuu`	`1u-u uuuu`
Stack Underflow Reset (STVREN = `1`)	0000h	`---u uuuu`	`u1-u uuuu`

Legend: `u` = unchanged, `x` = unknown, `-` = unimplemented bit, reads as ‘`0`’.

Note 1: When the wake-up is due to an interrupt and Global Enable bit (GIE) is set, the return address is pushed on the stack and PC is loaded with the interrupt vector (0004h) after execution of PC + 1.

2: If a Status bit is not implemented, that bit will be read as ‘`0`’.

5.12 Power Control (PCON) Register

The Power Control (PCON) register contains flag bits to differentiate between a:

- Power-on Reset ($\overline{\text{POR}}$)
- Brown-out Reset ($\overline{\text{BOR}}$)
- Reset Instruction Reset ($\overline{\text{RI}}$)
- $\overline{\text{MCLR}}$ Reset ($\overline{\text{RMCLR}}$)
- Watchdog Timer Reset ($\overline{\text{RWDT}}$)
- Stack Underflow Reset (STKUNF)
- Stack Overflow Reset (STKOVF)

The PCON register bits are shown in Register 5-2.

REGISTER 5-2: PCON: POWER CONTROL REGISTER

R/W/HS-0/q	R/W/HS-0/q	U-0	R/W/HC-1/q	R/W/HC-1/q	R/W/HC-1/q	R/W/HC-q/u	R/W/HC-q/u
STKOVF	STKUNF	—	$\overline{\text{RWDT}}$	$\overline{\text{RMCLR}}$	$\overline{\text{RI}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$
bit 7							bit 0

Legend:

HC = Bit is cleared by hardware

HS = Bit is set by hardware

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-m/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	q = Value depends on condition

bit 7	**STKOVF:** Stack Overflow Flag bit `1` = A Stack Overflow occurred `0` = A Stack Overflow has not occurred or set to ‘`0`’ by firmware
bit 6	**STKUNF:** Stack Underflow Flag bit `1` = A Stack Underflow occurred `0` = A Stack Underflow has not occurred or set to ‘`0`’ by firmware
bit 5	**Unimplemented:** Read as ‘`0`’
bit 4	**$\overline{\text{RWDT}}$:** Watchdog Timer Reset Flag bit `1` = A Watchdog Timer Reset has not occurred or set to ‘`1`’ by firmware `0` = A Watchdog Timer Reset has occurred (set to ‘`0`’ in hardware when a Watchdog Timer Reset)
bit 3	**$\overline{\text{RMCLR}}$:** $\overline{\text{MCLR}}$ Reset Flag bit `1` = A $\overline{\text{MCLR}}$ Reset has not occurred or set to ‘`1`’ by firmware `0` = A $\overline{\text{MCLR}}$ Reset has occurred (set to ‘`0`’ in hardware when a $\overline{\text{MCLR}}$ Reset occurs)
bit 2	**$\overline{\text{RI}}$:** RESET Instruction Flag bit `1` = A RESET instruction has not been executed or set to ‘`1`’ by firmware `0` = A RESET instruction has been executed (set to ‘`0`’ in hardware upon executing a RESET instruction)
bit 1	**$\overline{\text{POR}}$:** Power-on Reset Status bit `1` = No Power-on Reset occurred `0` = A Power-on Reset occurred (must be set in software after a Power-on Reset occurs)
bit 0	**$\overline{\text{BOR}}$:** Brown-out Reset Status bit `1` = No Brown-out Reset occurred `0` = A Brown-out Reset occurred (must be set in software after a Power-on Reset or Brown-out Reset occurs)

TABLE 5-5: SUMMARY OF REGISTERS ASSOCIATED WITH RESETS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
BORCON	SBOREN	BORFS	—	—	—	—	—	BORRDY	40
PCON	STKOVF	STKUNF	—	$\overline{\text{RWDT}}$	$\overline{\text{RMCLR}}$	$\overline{\text{RI}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	44
STATUS	—	—	—	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Z	DC	C	16
WDTCON	—	—	WDTPS<4:0>					SWDTEN	70

Legend: — = unimplemented location, read as ‘0’. Shaded cells are not used by Resets.

6.0 OSCILLATOR MODULE

6.1 Overview

The oscillator module has a wide variety of clock sources and selection features that allow it to be used in a wide range of applications while maximizing performance and minimizing power consumption. Figure 6-1 illustrates a block diagram of the oscillator module.

Clock sources can be supplied from external clock circuits. In addition, the system clock source can be supplied from one of two internal oscillators, with a choice of speeds selectable via software. Additional clock features include:

- Selectable system clock source between external or internal sources via software.
- Fast start-up oscillator allows internal circuits to power up and stabilize before switching to the 16 MHz HFINTOSC

The oscillator module can be configured in one of the following Clock modes:

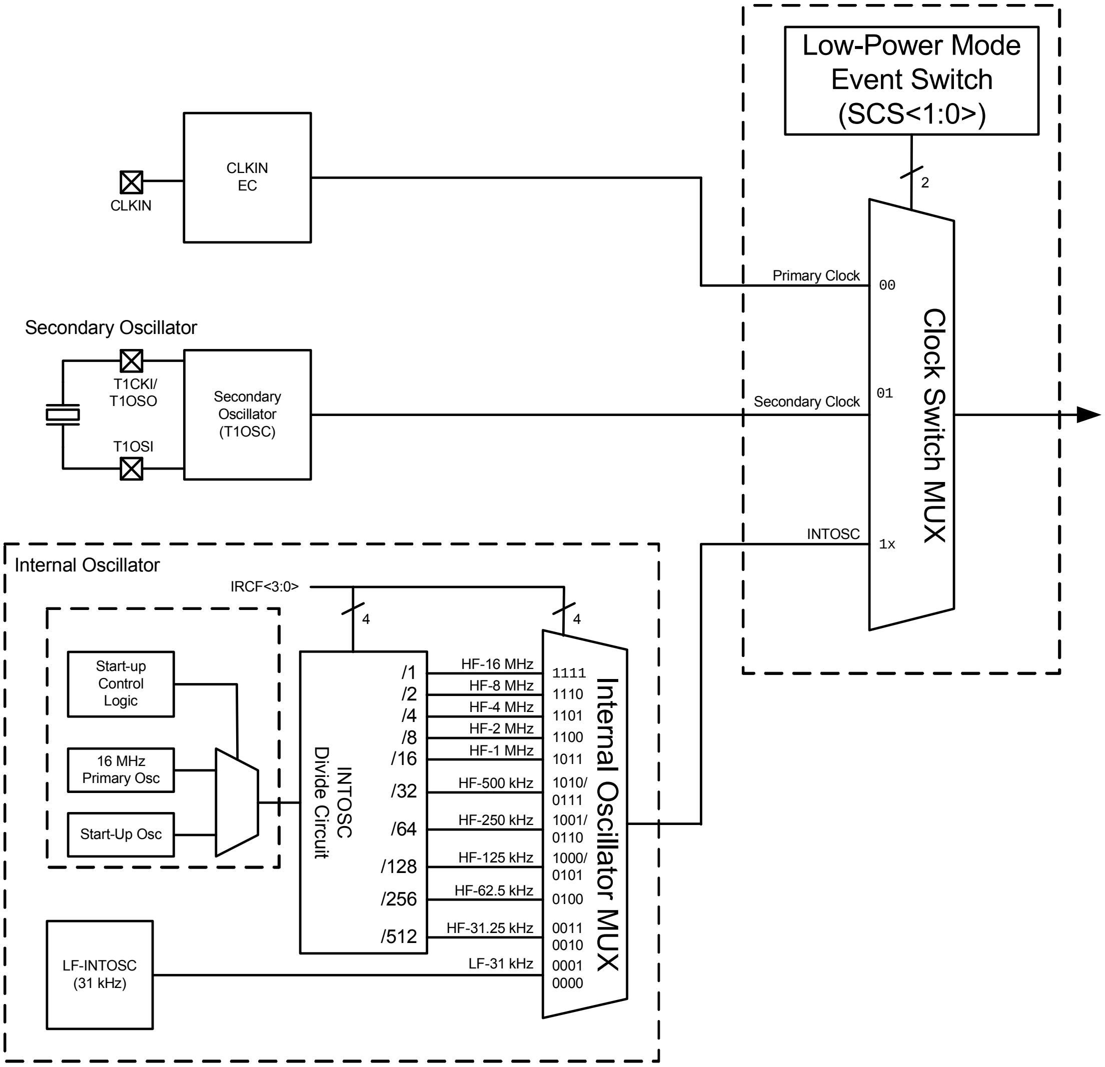
1. ECL – External Clock Low-Power mode (0 MHz to 0.5 MHz)
2. ECM – External Clock Medium-Power mode (0.5 MHz to 4 MHz)
3. ECH – External Clock High-Power mode (4 MHz to 32 MHz)
4. INTOSC – Internal oscillator (31 kHz to 16 MHz).

Clock Source modes are selected by the FOSC<1:0> bits in the Configuration Word 1. The FOSC bits determine the type of oscillator that will be used when the device is first powered.

The EC Clock mode relies on an external logic level signal as the device clock source.

The INTOSC internal oscillator block produces a low and high frequency clock source, designated LFINTOSC and HFINTOSC. (see Internal Oscillator Block, Figure 6-1). A wide selection of device clock frequencies may be derived from these two clock sources.

FIGURE 6-1: SIMPLIFIED PIC® MCU CLOCK SOURCE BLOCK DIAGRAM


Low-Power Mode
Event Switch
(SCS<1:0>)
2
CLKIN
CLKIN
EC
Primary Clock
00
Clock Switch MUX
Secondary Oscillator
T1CKI/
T1OSO
T1OSI
Secondary
Oscillator
(T1OSC)
Secondary Clock
01
INTOSC
1x
Internal Oscillator
IRCF<3:0>
4
4
Start-up
Control
Logic
16 MHz
Primary Osc
Start-Up Osc
INTOSC
Divide Circuit
/1
/2
/4
/8
/16
/32
/64
/128
/256
/512
HF-16 MHz
HF-8 MHz
HF-4 MHz
HF-2 MHz
HF-1 MHz
HF-500 kHz
HF-250 kHz
HF-125 kHz
HF-62.5 kHz
HF-31.25 kHz
LF-31 kHz
1111
1110
1101
1100
1011
1010/
0111
1001/
0110
1000/
0101
0100
0011
0010
0001
0000
Internal Oscillator MUX
LF-INTOSC
(31 kHz)

6.2 Clock Source Types

Clock sources can be classified as external or internal.

External clock sources rely on external circuitry for the clock source to function. An example is: oscillator module (EC mode) circuit.

Internal clock sources are contained internally within the oscillator module. The internal oscillator block has two internal oscillators that are used to generate the internal system clock sources: the 16 MHz High-Frequency Internal Oscillator and the 31 kHz Low-Frequency Internal Oscillator (LFINTOSC).

The system clock can be selected between external or internal clock sources via the System Clock Select (SCS) bits in the OSCCON register. See **Section 6.3 "Clock Switching"** for additional information.

6.2.1 EXTERNAL CLOCK SOURCES

An external clock source can be used as the device system clock by performing one of the following actions:

- Program the FOSC<1:0> bits in the Configuration Word 1 to select an external clock source that will be used as the default system clock upon a device Reset.
- Write the SCS<1:0> bits in the OSCCON register to switch the system clock source to:
  - Secondary oscillator during run-time, or
  - An external clock source determined by the value of the FOSC bits.

See **Section 6.3 "Clock Switching"**for more information.

6.2.1.1 EC Mode

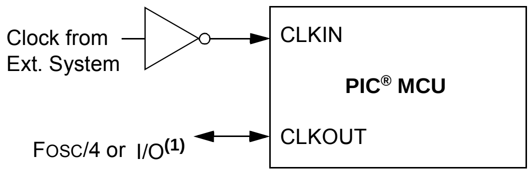
The External Clock (EC) mode allows an externally generated logic level signal to be the system clock source. When operating in this mode, an external clock source is connected to the CLKIN input. CLKOUT is available for general purpose I/O or CLKOUT. Figure 6-2 shows the pin connections for EC mode.

EC mode has three power modes to select from through Configuration Word 1:

- High power, 4-20 MHz (FOSC = `11`)
- Medium power, 0.5-4 MHz (FOSC = `10`)
- Low power, 0-0.5 MHz (FOSC = `01`)

There is no delay in operation after a Power-on Reset (POR) or wake-up from Sleep. Because the PIC® MCU design is fully static, stopping the external clock input will have the effect of halting the device while leaving all data intact. Upon restarting the external clock, the device will resume operation as if no time had elapsed.

FIGURE 6-2: EXTERNAL CLOCK (EC) MODE OPERATION



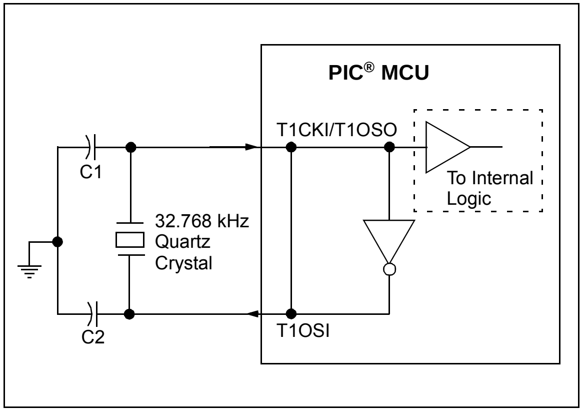
Note 1: Output depends upon $\overline{\text{CLKOUTEN}}$ bit of the Configuration Word 1.

6.2.1.2 Secondary Oscillator

The secondary oscillator is a separate crystal oscillator that is associated with the Timer1 peripheral. It is optimized for timekeeping operations with a 32.768 kHz crystal connected between the T1CKI/T1OSO and T1OSI device pins.

The secondary oscillator can be used as an alternate system clock source and can be selected during run-time using clock switching. Refer to **Section 6.3 "Clock Switching"** for more information.

FIGURE 6-3: QUARTZ CRYSTAL OPERATION (SECONDARY OSCILLATOR)



Note 1: Quartz crystal characteristics vary according to type, package and manufacturer. The user should consult the manufacturer data sheets for specifications and recommended application.

2: Always verify oscillator performance over the VDD and temperature range that is expected for the application.

3: For oscillator design assistance, reference the following Microchip Applications Notes:

- AN826, "*Crystal Oscillator Basics and Crystal Selection for rfPIC® and PIC® Devices*" (DS00826)
- AN849, "*Basic PIC® Oscillator Design*" (DS00849)
- AN943, "*Practical PIC® Oscillator Analysis and Design*" (DS00943)
- AN949, "*Making Your Oscillator Work*" (DS00949)
- TB097, "*Interfacing a Micro Crystal MS1V-T1K 32.768 kHz Tuning Fork Crystal to a PIC16F690/SS*" (DS91097)
- AN1288, "*Design Practices for Low-Power External Oscillators*" (DS01288)

6.2.2 INTERNAL CLOCK SOURCES

The device may be configured to use the internal oscillator block as the system clock by performing one of the following actions:

- Program the FOSC<1:0> bits in Configuration Word 1 to select the INTOSC clock source, which will be used as the default system clock upon a device Reset.
- Write the SCS<1:0> bits in the OSCCON register to switch the system clock source to the internal oscillator during run-time. See **Section 6.3 "Clock Switching"**for more information.

In **INTOSC** mode, CLKIN is available for general purpose I/O. CLKOUT is available for general purpose I/O or CLKOUT.

The function of the $\overline{\text{CLKOUT}}$ pin is determined by the state of the $\overline{\text{CLKOUTEN}}$ bit in Configuration Word 1.

The internal oscillator block has two independent oscillators that provides the internal system clock source.

1. The **HFINTOSC** (High-Frequency Internal Oscillator) is factory calibrated and operates at 16 MHz.
2. The **LFINTOSC** (Low-Frequency Internal Oscillator) is uncalibrated and operates at 31 kHz.

6.2.2.1 HFINTOSC

The High-Frequency Internal Oscillator (HFINTOSC) is a factory calibrated 16 MHz internal clock source.

The output of the HFINTOSC connects to a postscaler and multiplexer (see Figure 6-1). The frequency derived from the HFINTOSC can be selected via software using the IRCF<3:0> bits of the OSCCON register. See **Section 6.2.2.4 "Internal Oscillator Clock Switch Timing"** for more information.

The HFINTOSC is enabled by:

- Configure the IRCF<3:0> bits of the OSCCON register for the desired HF frequency, and
- FOSC<1:0> = `11`, or
- Set the System Clock Source (SCS) bits of the OSCCON register to '`1x`'.

The High-Frequency Internal Oscillator Ready bit (HFIOFR) of the OSCSTAT register indicates when the HFINTOSC is running and can be utilized.

The High-Frequency Internal Oscillator Status Stable bit (HFIOFS) of the OSCSTAT register indicates when the HFINTOSC is running within 0.5% of its final value.

6.2.2.2 LFINTOSC

The Low-Frequency Internal Oscillator (LFINTOSC) is an uncalibrated 31 kHz internal clock source.

The output of the LFINTOSC connects to a multiplexer (see Figure 6-1). Select 31 kHz, via software, using the IRCF<3:0> bits of the OSCCON register. See **Section 6.2.2.4 "Internal Oscillator Clock Switch Timing"** for more information. The LFINTOSC is also the frequency for the Power-up Timer (PWRT) and Watchdog Timer (WDT).

The LFINTOSC is enabled by selecting 31 kHz (IRCF<3:0> bits of the OSCCON register = `000`) as the system clock source (SCS bits of the OSCCON register = `1x`), or when any of the following are enabled:

- Configure the IRCF<3:0> bits of the OSCCON register for the desired LF frequency, and
- FOSC<1:0> = `01`, or
- Set the System Clock Source (SCS) bits of the OSCCON register to '`1x`'

Peripherals that use the LFINTOSC are:

- Power-up Timer (PWRT)
- Watchdog Timer (WDT)

The Low-Frequency Internal Oscillator Ready bit (LFIOFR) of the OSCSTAT register indicates when the LFINTOSC is running and can be utilized.

6.2.2.3 Internal Oscillator Frequency Selection

The system clock speed can be selected via software using the Internal Oscillator Frequency Select bits IRCF<3:0> of the OSCCON register.

The output of the 16 MHz HFINTOSC and 31 kHz LFINTOSC connects to a postscaler and multiplexer (see Figure 6-1). The Internal Oscillator Frequency Select bits IRCF<3:0> of the OSCCON register select the frequency output of the internal oscillators. One of the following frequencies can be selected via software:

- 16 MHz
- 8 MHz
- 4 MHz
- 2 MHz
- 1 MHz
- 500 kHz (Default after Reset)
- 250 kHz
- 125 kHz
- 62.5 kHz
- 31.25 kHz
- 31 kHz (LFINTOSC)

Note: Following any Reset, the IRCF<3:0> bits of the OSCCON register are set to '`0111`' and the frequency selection is set to 500 kHz. The user can modify the IRCF bits to select a different frequency.

The IRCF<3:0> bits of the OSCCON register allow duplicate selections for some frequencies. These duplicate choices can offer system design trade-offs. Lower power consumption can be obtained when changing oscillator sources for a given frequency. Faster transition times can be obtained between frequency changes that use the same oscillator source.

6.2.2.4 Internal Oscillator Clock Switch Timing

When switching between the HFINTOSC and the LFINTOSC, the new oscillator may already be shut down to save power (see Figure 6-4). If this is the case, there is a delay after the IRCF<3:0> bits of the OSCCON register are modified before the frequency selection takes place. The OSCSTAT register will reflect the current active status of the HFINTOSC and LFINTOSC oscillators. The sequence of a frequency selection is as follows:

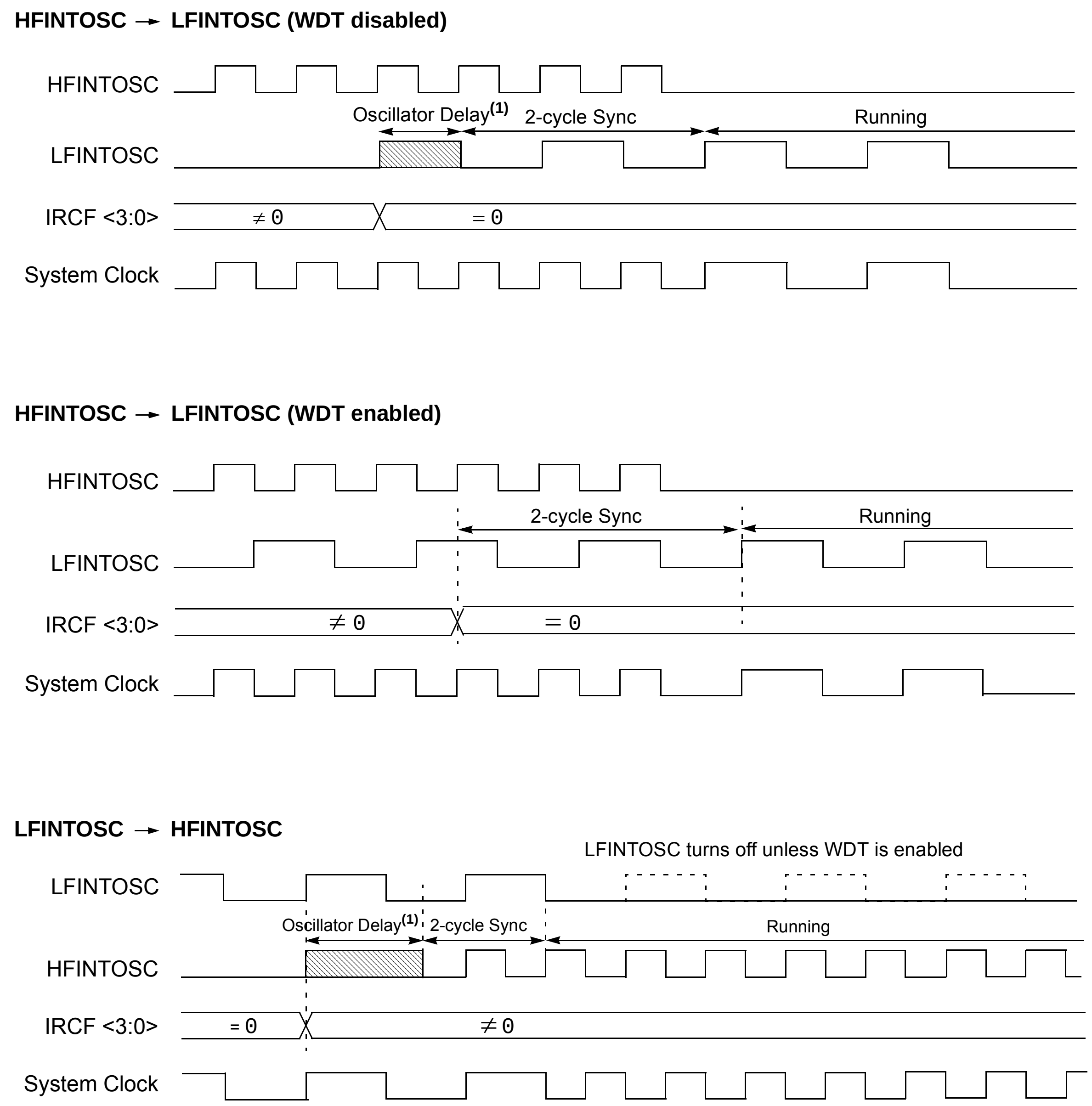
1. IRCF<3:0> bits of the OSCCON register are modified.
2. If the new clock is shut down, a clock start-up delay is started.
3. Clock switch circuitry waits for a falling edge of the current clock.
4. The current clock is held low and the clock switch circuitry waits for a rising edge in the new clock.
5. The new clock is now active.
6. The OSCSTAT register is updated as required.
7. Clock switch is complete.

See Figure 6-4 for more details.

If the internal oscillator speed is switched between two clocks of the same source, there is no start-up delay before the new frequency is selected. Clock switching time delays are shown in Table 6-2.

Start-up delay specifications are located in the oscillator tables of **Section 21.0 "Electrical Specifications"**.

FIGURE 6-4: INTERNAL OSCILLATOR SWITCH TIMING


HFINTOSC → LFINTOSC (WDT disabled)
HFINTOSC
Oscillator Delay(1)
2-cycle Sync
Running
LFINTOSC
IRCF <3:0>
≠ 0
= 0
System Clock
HFINTOSC → LFINTOSC (WDT enabled)
HFINTOSC
2-cycle Sync
Running
LFINTOSC
IRCF <3:0>
≠ 0
= 0
System Clock
LFINTOSC → HFINTOSC
LFINTOSC turns off unless WDT is enabled
LFINTOSC
Oscillator Delay(1)
2-cycle Sync
Running
HFINTOSC
IRCF <3:0>
= 0
≠ 0
System Clock

Note 1: See Table 6-1 for more information.

TABLE 6-1: OSCILLATOR SWITCHING DELAYS

Switch From	Switch To	Oscillator Delay
Any clock source	LFINTOSC	1 cycle of each clock source
	HFINTOSC	2 μs (approx.)
	ECH, ECM, ECL	2 cycles
	Secondary Oscillator	1024 Secondary Oscillator Cycles

6.3 Clock Switching

The system clock source can be switched between external and internal clock sources via software using the System Clock Select (SCS) bits of the OSCCON register. The following clock sources can be selected using the SCS bits:

- Default system oscillator determined by FOSC bits in Configuration Word 1
- Secondary oscillator 32 kHz crystal
- Internal Oscillator Block (INTOSC)

6.3.1 SYSTEM CLOCK SELECT (SCS) BITS

The System Clock Select (SCS) bits of the OSCCON register selects the system clock source that is used for the CPU and peripherals.

- When the SCS bits of the OSCCON register = `00`, the system clock source is determined by value of the FOSC<1:0> bits in the Configuration Word 1.
- When the SCS bits of the OSCCON register = `01`, the system clock source is the secondary oscillator.
- When the SCS bits of the OSCCON register = `1x`, the system clock source is chosen by the internal oscillator frequency selected by the IRCF<3:0> bits of the OSCCON register. After a Reset, the SCS bits of the OSCCON register are always cleared.

When switching between clock sources, a delay is required to allow the new clock to stabilize. These oscillator delays are shown in Table 6-2.

6.3.2 OSCILLATOR START-UP TIME-OUT STATUS (OSTS) BIT

The Oscillator Start-up Time-out Status (OSTS) bit of the OSCSTAT register indicates whether the system clock is running from the external clock source, as defined by the FOSC<1:0> bits in the Configuration Word 1, or from the internal clock source.

6.3.3 SECONDARY OSCILLATOR

The secondary oscillator is a separate crystal oscillator associated with the Timer1 peripheral. It is optimized for timekeeping operations with a 32.768 kHz crystal connected between the T1OSI and T1CKI/T1OSO device pins.

The secondary oscillator is enabled using the T1OSCEN control bit in the T1CON register. See **Section 17.0 “Timer1 Module with Gate Control”** for more information about the Timer1 peripheral.

6.3.4 SECONDARY OSCILLATOR READY (T1OSCR) BIT

The user must ensure that the secondary oscillator is ready to be used before it is selected as a system clock source. The Secondary Oscillator Ready (T1OSCR) bit of the OSCSTAT register indicates whether the secondary oscillator is ready to be used. After the T1OSCR bit is set, the SCS bits can be configured to select the secondary oscillator.

6.4 Oscillator Control Registers

REGISTER 6-1: OSCCON: OSCILLATOR CONTROL REGISTER

U-0	R/W-0/0	R/W-1/1	R/W-1/1	R/W-1/1	U-0	R/W-0/0	R/W-0/0
—	IRCF<3:0>				—	SCS<1:0>	
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7 **Unimplemented:** Read as ‘`0`’

bit 6-3 **IRCF<3:0>:** Internal Oscillator Frequency Select bits
`000x` = 31 kHz LF
`001x` = 31.25 kHz
`0100` = 62.5 kHz
`0101` = 125 kHz
`0110` = 250 kHz
`0111` = 500 kHz (default upon Reset)
`1000` = 125 kHz[1]
`1001` = 250 kHz[1]
`1010` = 500 kHz[1]
`1011` = 1 MHz
`1100` = 2 MHz
`1101` = 4 MHz
`1110` = 8 MHz
`1111` = 16 MHz

bit 2 **Unimplemented:** Read as ‘`0`’

bit 1-0 **SCS<1:0>:** System Clock Select bits
`1x` = Internal oscillator block
`01` = Secondary oscillator
`00` = Clock determined by FOSC<1:0> in Configuration Word 1.

Note 1: Duplicate frequency derived from HFINTOSC.

REGISTER 6-2: OSCSTAT: OSCILLATOR STATUS REGISTER

R-1/q	U-0	R-q/q	R-0/q	U-0	U-0	R-0/0	R-0/q
T1OSCR	—	OSTS	HFIOFR	—	—	LFIOFR	HFIOFS
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	q = Conditional

bit 7 **T1OSCR:** Timer1 Oscillator Ready bit
If T1OSCEN = `1`:
`1` = Timer1 oscillator is ready
`0` = Timer1 oscillator is not ready
If T1OSCEN = 0:
`1` = Timer1 clock source is always ready

bit 6 **Unimplemented:** Read as ‘`0`’

bit 5 **OSTS:** Oscillator Start-up Time-out Status bit
`1` = Running from the external clock source (EC)
`0` = Running from an internal oscillator (FOSC<1:0> = `00`)

bit 4 **HFIOFR:** High-Frequency Internal Oscillator Ready bit
`1` = HFINTOSC is ready
`0` = HFINTOSC is not ready

bit 3-2 **Unimplemented:** Read as ‘`0`’

bit 1 **LFIOFR:** Low-Frequency Internal Oscillator Ready bit
`1` = LFINTOSC is ready
`0` = LFINTOSC is not ready

bit 0 **HFIOFS:** High-Frequency Internal Oscillator Stable bit
`1` = HFINTOSC 16 MHz oscillator is stable and is driving the INTOSC
`0` = HFINTOSC 16 MHz oscillator is not stable, the Start-up Oscillator is driving INTOSC

TABLE 6-2: SUMMARY OF REGISTERS ASSOCIATED WITH CLOCK SOURCES

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
OSCCON	—	IRCF<3:0>				—	SCS<1:0>		53
OSCSTAT	T1OSCR	—	OSTS	HFIOFR	—	—	LFIOFR	HFIOFS	54
T1CON	TMR1CS<1:0>		T1CKPS<1:0>		T1OSCEN	$\overline{\text{T1SYNC}}$	—	TMR1ON	130

Legend: — = unimplemented location, read as ‘`0`’. Shaded cells are not used by clock sources.

TABLE 6-3: SUMMARY OF CONFIGURATION WORD WITH CLOCK SOURCES

Name	Bits	Bit -/7	Bit -/6	Bit 13/5	Bit 12/4	Bit 11/3	Bit 10/2	Bit 9/1	Bit 8/0	Register on Page
CONFIG1	13:8	—	—	—	—	$\overline{\text{CLKOUTEN}}$	BOREN<1:0>		—	34
	7:0	$\overline{\text{CP}}$	MCLRE	$\overline{\text{PWRTE}}$	WDTE<1:0>		—	FOSC<1:0>		

Legend: — = unimplemented location, read as ‘`0`’. Shaded cells are not used by clock sources.

7.0 INTERRUPTS

The interrupt feature allows certain events to preempt normal program flow. Firmware is used to determine the source of the interrupt and act accordingly. Some interrupts can be configured to wake the MCU from Sleep mode.

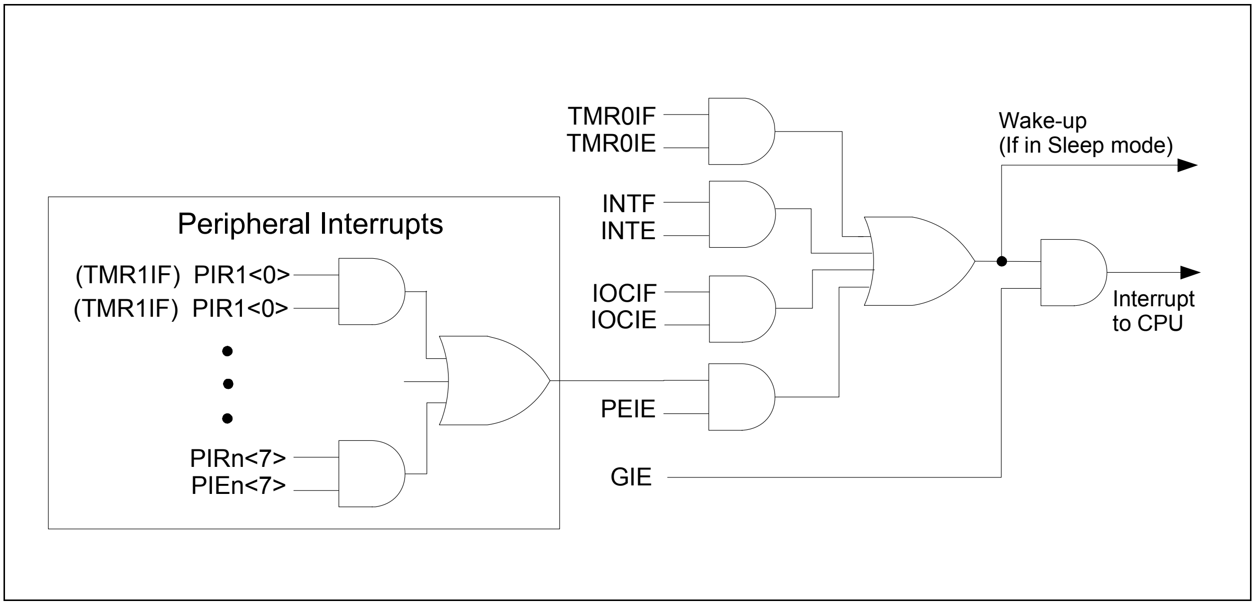
This chapter contains the following information for Interrupts:

- Operation
- Interrupt Latency
- Interrupts During Sleep
- INT Pin
- Automatic Context Saving

Many peripherals produce Interrupts. Refer to the corresponding chapters for details.

A block diagram of the interrupt logic is shown in Figure 7.1 and Figure 7.1.

FIGURE 7-1: INTERRUPT LOGIC


TMR0IF
TMR0IE
Wake-up
(If in Sleep mode)
Peripheral Interrupts
INTF
INTE
(TMR1IF) PIR1<0>
(TMR1IF) PIR1<0>
IOCIF
IOCIE
Interrupt
to CPU
PEIE
PIRn<7>
PIEn<7>
GIE

7.1 Operation

Interrupts are disabled upon any device Reset. They are enabled by setting the following bits:

- GIE bit of the INTCON register
- Interrupt Enable bit(s) for the specific interrupt event(s)
- PEIE bit of the INTCON register (if the Interrupt Enable bit of the interrupt event is contained in the PIE1 and PIE2 registers)

The INTCON, PIR1 and PIR2 registers record individual interrupts via interrupt flag bits. Interrupt flag bits will be set, regardless of the status of the GIE, PEIE and individual interrupt enable bits.

The following events happen when an interrupt event occurs while the GIE bit is set:

- Current prefetched instruction is flushed
- GIE bit is cleared
- Current Program Counter (PC) is pushed onto the stack
- Critical registers are automatically saved to the shadow registers (See **Section 7.5 “Automatic Context Saving”**)
- PC is loaded with the interrupt vector 0004h

The firmware within the Interrupt Service Routine (ISR) should determine the source of the interrupt by polling the interrupt flag bits. The interrupt flag bits must be cleared before exiting the ISR to avoid repeated interrupts. Because the GIE bit is cleared, any interrupt that occurs while executing the ISR will be recorded through its interrupt flag, but will not cause the processor to redirect to the interrupt vector.

The `RETFIE` instruction exits the ISR by popping the previous address from the stack, restoring the saved context from the shadow registers and setting the GIE bit.

For additional information on a specific interrupt’s operation, refer to its peripheral chapter.

Note 1: Individual interrupt flag bits are set, regardless of the state of any other enable bits.

2: All interrupts will be ignored while the GIE bit is cleared. Any interrupt occurring while the GIE bit is clear will be serviced when the GIE bit is set again.

7.2 Interrupt Latency

Interrupt latency is defined as the time from when the interrupt event occurs to the time code execution at the interrupt vector begins. The latency for synchronous interrupts is three or four instruction cycles. For asynchronous interrupts, the latency is three to five instruction cycles, depending on when the interrupt occurs. See Figure 7-2 and Figure 7.3 for more details.

FIGURE 7-2: INTERRUPT LATENCY

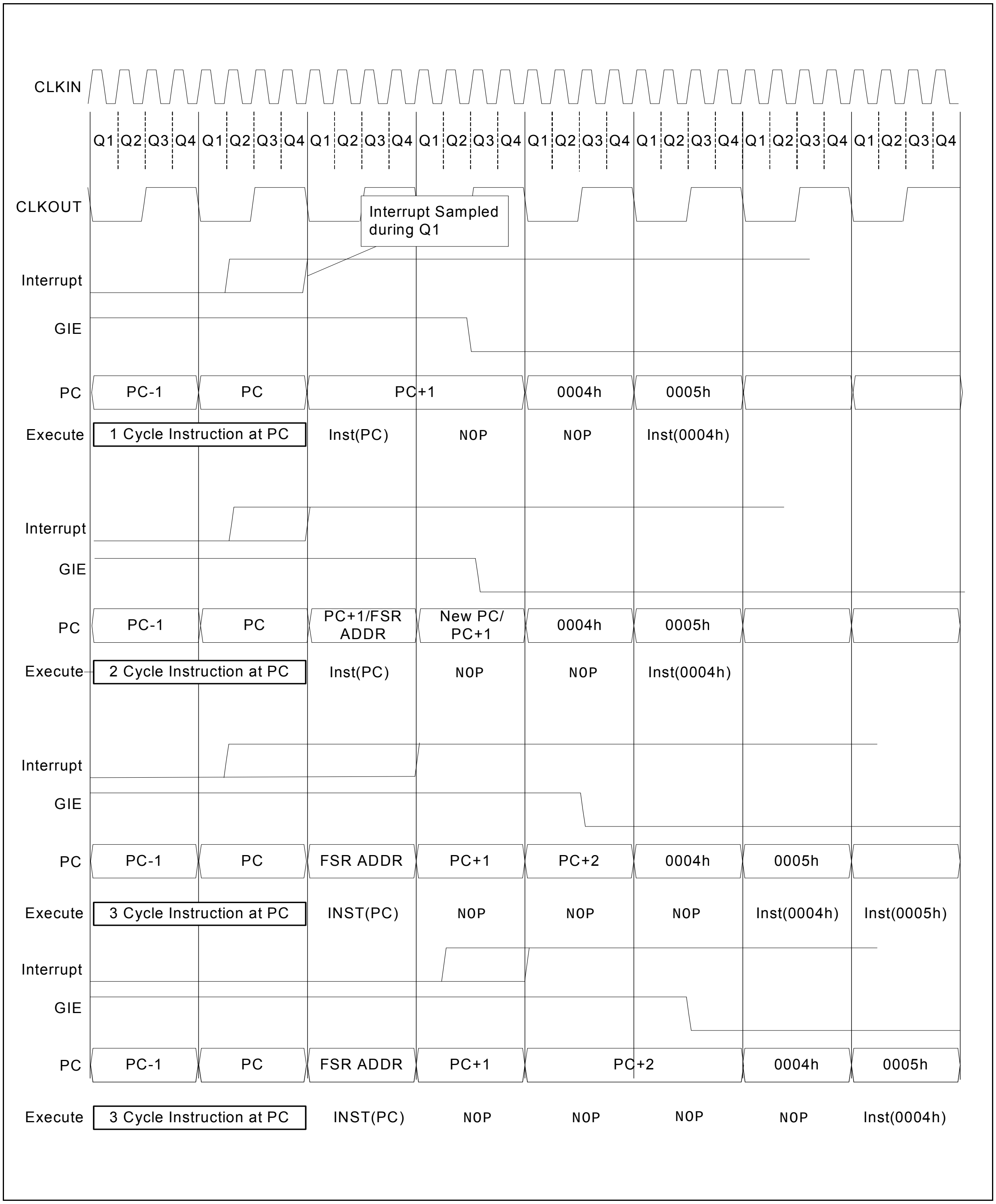
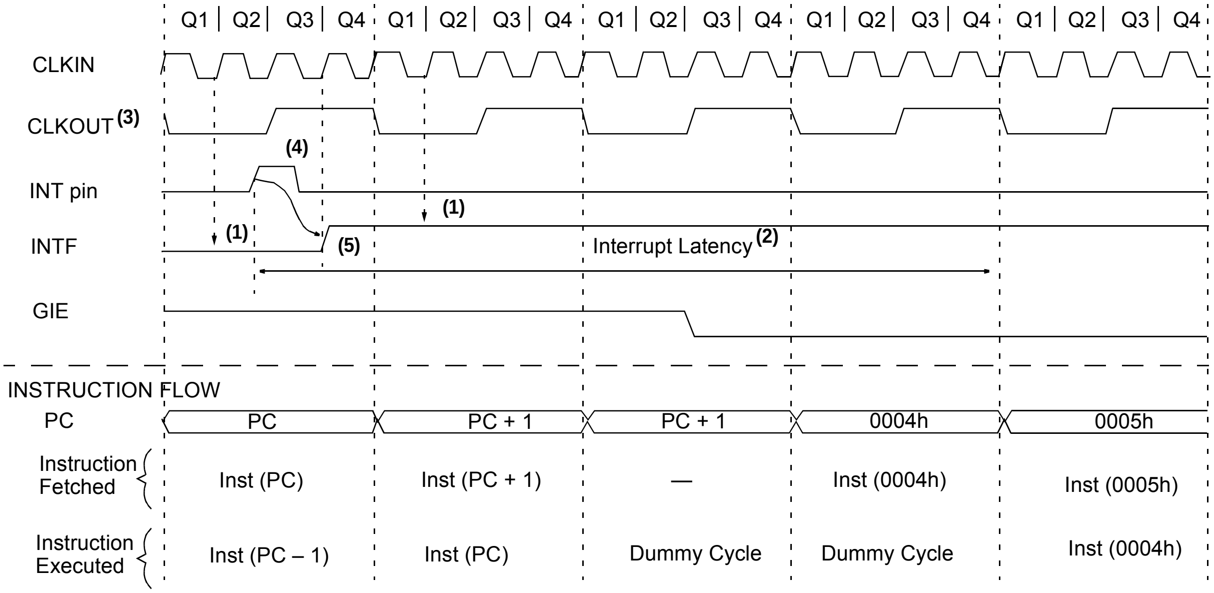

CLKIN
Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4
CLKOUT
Interrupt Sampled during Q1
Interrupt
GIE
PC
PC-1
PC
PC+1
0004h
0005h
Execute
1 Cycle Instruction at PC
Inst(PC)
NOP
NOP
Inst(0004h)
Interrupt
GIE
PC
PC-1
PC
PC+1/FSR ADDR
New PC/ PC+1
0004h
0005h
Execute
2 Cycle Instruction at PC
Inst(PC)
NOP
NOP
Inst(0004h)
Interrupt
GIE
PC
PC-1
PC
FSR ADDR
PC+1
PC+2
0004h
0005h
Execute
3 Cycle Instruction at PC
INST(PC)
NOP
NOP
NOP
Inst(0004h)
Inst(0005h)
Interrupt
GIE
PC
PC-1
PC
FSR ADDR
PC+1
PC+2
0004h
0005h
Execute
3 Cycle Instruction at PC
INST(PC)
NOP
NOP
NOP
NOP
Inst(0004h)

FIGURE 7-3: INT PIN INTERRUPT TIMING


Q1 Q2 Q3 Q4
CLKIN
CLKOUT(3)
INT pin
INTF
GIE
(4)
(1)
(1)
(5)
Interrupt Latency(2)
INSTRUCTION FLOW
PC
PC
PC + 1
PC + 1
0004h
0005h
Instruction Fetched
Inst (PC)
Inst (PC + 1)
—
Inst (0004h)
Inst (0005h)
Instruction Executed
Inst (PC – 1)
Inst (PC)
Dummy Cycle
Dummy Cycle
Inst (0004h)

Note 1: INTF flag is sampled here (every Q1).

2: Asynchronous interrupt latency = 3-5 TCY. Synchronous latency = 3-4 TCY, where TCY = instruction cycle time. Latency is the same whether Inst (PC) is a single cycle or a 2-cycle instruction.

3: CLKOUT not available in all Oscillator modes.

4: For minimum width of INT pulse, refer to AC specifications in **Section 21.0 "Electrical Specifications"**.

5: INTF is enabled to be set any time during the Q4-Q1 cycles.

7.3 Interrupts During Sleep

Some interrupts can be used to wake from Sleep. To wake from Sleep, the peripheral must be able to operate without the system clock. The interrupt source must have the appropriate Interrupt Enable bit(s) set prior to entering Sleep.

On waking from Sleep, if the GIE bit is also set, the processor will branch to the interrupt vector. Otherwise, the processor will continue executing instructions after the `SLEEP` instruction. The instruction directly after the `SLEEP` instruction will always be executed before branching to the ISR. Refer to the **Section 8.0 "Power-Down Mode (Sleep)"** for more details.

7.4 INT Pin

The INT pin can be used to generate an asynchronous edge-triggered interrupt. This interrupt is enabled by setting the INTE bit of the INTCON register. The INTEDG bit of the OPTION_REG register determines on which edge the interrupt will occur. When the INTEDG bit is set, the rising edge will cause the interrupt. When the INTEDG bit is clear, the falling edge will cause the interrupt. The INTF bit of the INTCON register will be set when a valid edge appears on the INT pin. If the GIE and INTE bits are also set, the processor will redirect program execution to the interrupt vector.

7.5 Automatic Context Saving

Upon entering an interrupt, the return PC address is saved on the stack. Additionally, the following registers are automatically saved in the Shadow registers:

- W register
- STATUS register (except for $\overline{\text{TO}}$ and $\overline{\text{PD}}$)
- BSR register
- FSR registers
- PCLATH register

Upon exiting the Interrupt Service Routine, these registers are automatically restored. Any modifications to these registers during the ISR will be lost. If modifications to any of these registers are desired, the corresponding Shadow register should be modified and the value will be restored when exiting the ISR. The Shadow registers are available in Bank 31 and are readable and writable. Depending on the user's application, other registers may also need to be saved.

7.6 Interrupt Control Registers

7.6.1 INTCON REGISTER

The INTCON register is a readable and writable register, which contains the various enable and flag bits for TMR0 register overflow, interrupt-on-change and external INT pin interrupts.

> **Note:** Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the Global Enable bit, GIE of the INTCON register. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

REGISTER 7-1: INTCON: INTERRUPT CONTROL REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R-0/0
GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7	**GIE:** Global Interrupt Enable bit `1` = Enables all active interrupts `0` = Disables all interrupts
bit 6	**PEIE:** Peripheral Interrupt Enable bit `1` = Enables all active peripheral interrupts `0` = Disables all peripheral interrupts
bit 5	**TMR0IE:** Timer0 Overflow Interrupt Enable bit `1` = Enables the Timer0 interrupt `0` = Disables the Timer0 interrupt
bit 4	**INTE:** INT External Interrupt Enable bit `1` = Enables the INT external interrupt `0` = Disables the INT external interrupt
bit 3	**IOCIE:** Interrupt-on-Change Interrupt Enable bit `1` = Enables the interrupt-on-change interrupt `0` = Disables the interrupt-on-change interrupt
bit 2	**TMR0IF:** Timer0 Overflow Interrupt Flag bit `1` = TMR0 register has overflowed `0` = TMR0 register did not overflow
bit 1	**INTF:** INT External Interrupt Flag bit `1` = The INT external interrupt occurred `0` = The INT external interrupt did not occur
bit 0	**IOCIF:** Interrupt-on-Change Interrupt Flag bit `1` = When at least one of the interrupt-on-change pins changed state `0` = None of the interrupt-on-change pins have changed state

7.6.2 PIE1 REGISTER

The PIE1 register contains the interrupt enable bits, as shown in Register 7-2.

Note: Bit PEIE of the INTCON register must be set to enable any peripheral interrupt.

REGISTER 7-2: PIE1: PERIPHERAL INTERRUPT ENABLE REGISTER 1

R/W-0/0	R/W-0/0	U-0	U-0	U-0	U-0	U-0	R/W-0/0
TMR1GIE	ADIE	—	—	—	—	—	TMR1IE
bit 7							bit 0

Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7	**TMR1GIE:** Timer1 Gate Interrupt Enable bit `1` = Enables the Timer1 Gate Acquisition interrupt `0` = Disables the Timer1 Gate Acquisition interrupt
bit 6	**ADIE:** A/D Converter (ADC) Interrupt Enable bit `1` = Enables the ADC interrupt `0` = Disables the ADC interrupt
bit 5-1	**Unimplemented:** Read as ‘`0`’
bit 0	**TMR1IE:** Timer1 Overflow Interrupt Enable bit `1` = Enables the Timer1 overflow interrupt `0` = Disables the Timer1 overflow interrupt

7.6.3 PIE2 REGISTER

The PIE2 register contains the interrupt enable bits, as shown in Register 7-3.

Note: Bit PEIE of the INTCON register must be set to enable any peripheral interrupt.

REGISTER 7-3: PIE2: PERIPHERAL INTERRUPT ENABLE REGISTER 2

U-0	U-0	U-0	U-0	U-0	R/W-0/0	U-0	U-0
—	—	—	—	—	LCDIE	—	—
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-3	**Unimplemented:** Read as ‘`0`’
bit 2	**LCDIE:** LCD Module Interrupt Enable bit `1` = Enables the LCD module interrupt `0` = Disables the LCD module interrupt
bit 1-0	**Unimplemented:** Read as ‘`0`’

7.6.4 PIR1 REGISTER

The PIR1 register contains the interrupt flag bits, as shown in Register 7-4.

Note: Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the Global Enable bit, GIE, of the INTCON register. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

REGISTER 7-4: PIR1: PERIPHERAL INTERRUPT REQUEST REGISTER 1

R/W-0/0	R/W-0/0	U-0	U-0	U-0	U-0	U-0	R/W-0/0
TMR1GIF	ADIF	—	—	—	—	—	TMR1IF
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7	**TMR1GIF:** Timer1 Gate Interrupt Flag bit `1` = Interrupt is pending `0` = Interrupt is not pending
bit 6	**ADIF:** A/D Converter Interrupt Flag bit `1` = Interrupt is pending `0` = Interrupt is not pending
bit 5-1	**Unimplemented:** Read as ‘`0`’
bit 0	**TMR1IF:** Timer1 Overflow Interrupt Flag bit `1` = Interrupt is pending `0` = Interrupt is not pending

7.6.5 PIR2 REGISTER

The PIR2 register contains the interrupt flag bits, as shown in Register 7-5.

Note:	Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the Global Enable bit, GIE, of the INTCON register. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

REGISTER 7-5: PIR2: PERIPHERAL INTERRUPT REQUEST REGISTER 2

U-0	U-0	U-0	U-0	U-0	R/W-0/0	U-0	U-0
—	—	—	—	—	LCDIF	—	—
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-3	**Unimplemented:** Read as ‘0’
bit 2	**LCDIF:** LCD Module Interrupt Flag bit 1 = Interrupt is pending 0 = Interrupt is not pending
bit 1-0	**Unimplemented:** Read as ‘0’

TABLE 7-1: SUMMARY OF REGISTERS ASSOCIATED WITH INTERRUPTS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	60
OPTION_REG	$\overline{\text{WPUEN}}$	INTEDG	TMR0CS	TMR0SE	PSA	PS<2:0>			121
PIE1	TMR1GIE	ADIE	—	—	—	—	—	TMR1IE	61
PIE2	—	—	—	—	—	LCDIE	—	—	62
PIR1	TMR1GIF	ADIF	—	—	—	—	—	TMR1IF	63
PIR2	—	—	—	—	—	LCDIF	—	—	64

Legend: — = unimplemented location, read as ‘0’. Shaded cells are not used by interrupts.

8.0 POWER-DOWN MODE (SLEEP)

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

Upon entering Sleep mode, the following conditions exist:

1. WDT will be cleared but keeps running, if enabled for operation during Sleep.
2. $\overline{\text{PD}}$ bit of the STATUS register is cleared.
3. $\overline{\text{TO}}$ bit of the STATUS register is set.
4. CPU clock is disabled.
5. 31 kHz LFINTOSC is unaffected and peripherals that operate from it may continue operation in Sleep.
6. Secondary oscillator is unaffected and peripherals that operate from it may continue operation in Sleep.
7. ADC is unaffected, if the dedicated FRC clock is selected.
8. Capacitive Sensing oscillator is unaffected.
9. I/O ports maintain the status they had before `SLEEP` was executed (driving high, low or high-impedance).
10. Resets other than WDT are not affected by Sleep mode.

Refer to individual chapters for more details on peripheral operation during Sleep.

To minimize current consumption, the following conditions should be considered:

- I/O pins should not be floating
- External circuitry sinking current from I/O pins
- Internal circuitry sourcing current from I/O pins
- Current draw from pins with internal weak pull-ups
- Modules using 31 kHz LFINTOSC
- Modules using Secondary oscillator

I/O pins that are high-impedance inputs should be pulled to VDD or VSS externally to avoid switching currents caused by floating inputs.

Examples of internal circuitry that might be sourcing current include the FVR module. See **13.0 “Fixed Voltage Reference (FVR)”** for more information.

8.1 Wake-up from Sleep

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

1. External Reset input on $\overline{\text{MCLR}}$ pin, if enabled
2. BOR Reset, if enabled
3. POR Reset
4. Watchdog Timer, if enabled
5. Any external interrupt
6. Interrupts by peripherals capable of running during Sleep (see individual peripheral for more information)

The first three events will cause a device Reset. The last three events are considered a continuation of program execution. To determine whether a device Reset or wake-up event occurred, refer to **Section 5.11, Determining the Cause of a Reset**.

When the `SLEEP` instruction is being executed, the next instruction (PC + 1) is prefetched. For the device to wake-up through an interrupt event, the corresponding interrupt enable bit must be enabled. Wake-up will occur regardless of the state of the GIE bit. If the GIE bit is disabled, the device continues execution at the instruction after the `SLEEP` instruction. If the GIE bit is enabled, the device executes the instruction after the `SLEEP` instruction, the device will then call the Interrupt Service Routine. 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.

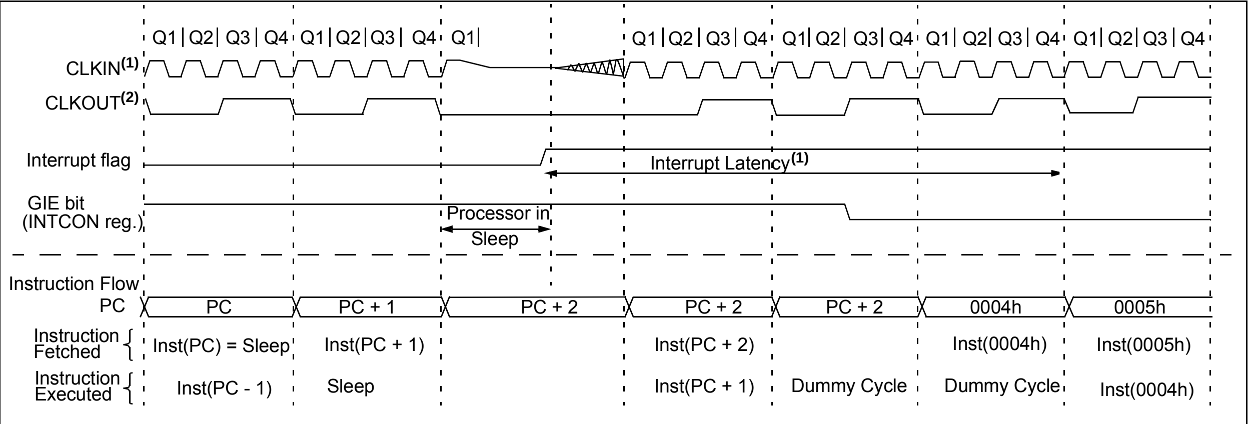
8.1.1 WAKE-UP USING INTERRUPTS

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

- If the interrupt occurs **before** the execution of a `SLEEP` instruction
  - `SLEEP` instruction will execute as a `NOP`.
  - WDT and WDT prescaler will not be cleared
  - $\overline{\text{TO}}$ bit of the STATUS register will not be set
  - $\overline{\text{PD}}$ bit of the STATUS register will not be cleared.
- If the interrupt occurs **during or after** the execution of a `SLEEP` instruction
  - `SLEEP` instruction will be completely executed
  - Device will immediately wake-up from Sleep
  - WDT and WDT prescaler will be cleared
  - $\overline{\text{TO}}$ bit of the STATUS register will be set
  - $\overline{\text{PD}}$ bit of the STATUS register will be cleared.

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

FIGURE 8-1: WAKE-UP FROM SLEEP THROUGH INTERRUPT


Q1 | Q2 | Q3 | Q4 | Q1 | Q2 | Q3 | Q4 | Q1 |
Q1 | Q2 | Q3 | Q4 | Q1 | Q2 | Q3 | Q4 | Q1 | Q2 | Q3 | Q4 | Q1 | Q2 | Q3 | Q4
CLKIN(1)
CLKOUT(2)
Interrupt flag
Interrupt Latency(1)
GIE bit (INTCON reg.)
Processor in Sleep
Instruction Flow
PC
PC
PC + 1
PC + 2
PC + 2
PC + 2
0004h
0005h
Instruction Fetched
Inst(PC) = Sleep
Inst(PC + 1)
Inst(PC + 2)
Inst(0004h)
Inst(0005h)
Instruction Executed
Inst(PC - 1)
Sleep
Inst(PC + 1)
Dummy Cycle
Dummy Cycle
Inst(0004h)

Note 1: GIE = `1` assumed. In this case after wake-up, the processor calls the ISR at 0004h. If GIE = `0`, execution will continue in-line.

TABLE 8-1: SUMMARY OF REGISTERS ASSOCIATED WITH POWER-DOWN MODE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	60
IOCBF	IOCBF7	IOCBF6	IOCBF5	IOCBF4	IOCBF3	IOCBF2	IOCBF1	IOCBF0	101
IOCBN	IOCBN7	IOCBN6	IOCBN5	IOCBN4	IOCBN3	IOCBN2	IOCBN1	IOCBN0	101
IOCBP	IOCBP7	IOCBP6	IOCBP5	IOCBP4	IOCBP3	IOCBP2	IOCBP1	IOCBP0	101
PIE1	TMR1GIE	ADIE	—	—	—	—	—	TMR1IE	61
PIE2	—	—	—	—	—	LCDIE	—	—	62
PIR1	TMR1GIF	ADIF	—	—	—	—	—	TMR1IF	63
PIR2	—	—	—	—	—	LCDIF	—	—	64
STATUS	—	—	—	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Z	DC	C	16
WDTCON	—	—	WDTPS<4:0>					SWDTEN	70

Legend: — = unimplemented location, read as ‘`0`’. Shaded cells are not used in Power-Down mode.

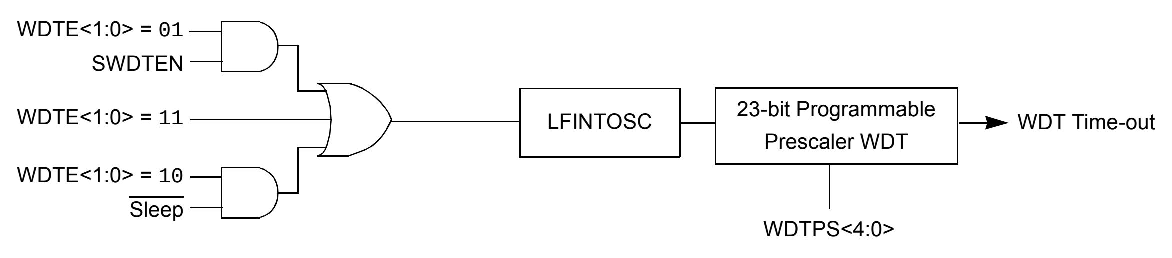
9.0 WATCHDOG TIMER

The Watchdog Timer is a system timer that generates a Reset if the firmware does not issue a `CLRWDT` instruction within the time-out period. The Watchdog Timer is typically used to recover the system from unexpected events.

The WDT has the following features:

- Independent clock source
- Multiple operating modes
  - WDT is always on
  - WDT is off when in Sleep
  - WDT is controlled by software
  - WDT is always off
- Configurable time-out period is from 1 ms to 256 seconds (typical)
- Multiple Reset conditions
- Operation during Sleep

FIGURE 9-1: WATCHDOG TIMER BLOCK DIAGRAM


WDTE<1:0> = 01
SWDTEN
WDTE<1:0> = 11
WDTE<1:0> = 10
Sleep
LFINTOSC
23-bit Programmable Prescaler WDT
WDT Time-out
WDTPS<4:0>

9.1 Independent Clock Source

The WDT derives its time base from the 31 kHz LFINTOSC internal oscillator. Time intervals in this chapter are based on a nominal interval of 1 ms. See **Section 21.0 “Electrical Specifications”** for the LFINTOSC tolerances.

9.2 WDT Operating Modes

The Watchdog Timer module has four operating modes controlled by the WDTE<1:0> bits in Configuration Word 1. See Table 9-1.

9.2.1 WDT IS ALWAYS ON

When the WDTE bits of Configuration Word 1 are set to ‘`11`’, the WDT is always on.

WDT protection is active during Sleep.

9.2.2 WDT IS OFF IN SLEEP

When the WDTE bits of Configuration Word 1 are set to ‘`10`’, the WDT is on, except in Sleep.

WDT protection is not active during Sleep.

9.2.3 WDT CONTROLLED BY SOFTWARE

When the WDTE bits of Configuration Word 1 are set to ‘`01`’, the WDT is controlled by the SWDTEN bit of the WDTCON register.

WDT protection is unchanged by Sleep. See Table 9-1 for more details.

TABLE 9-1: WDT OPERATING MODES

WDTE<1:0>	SWDTEN	Device Mode	WDT Mode
`11`	X	X	Active
`10`	X	Awake	Active
		Sleep	Disabled
`01`	`1`	X	Active
	`0`		Disabled
`00`	X	X	Disabled

9.3 Time-Out Period

The WDTPS bits of the WDTCON register set the time-out period from 1 ms to 256 seconds (nominal). After a Reset, the default time-out period is two seconds.

9.4 Clearing the WDT

The WDT is cleared when any of the following conditions occur:

- Any Reset
- `CLRWDT` instruction is executed
- Device enters Sleep
- Device wakes up from Sleep
- Oscillator fail event
- WDT is disabled

See Table 9-2 for more information.

9.5 Operation During Sleep

When the device enters Sleep, the WDT is cleared. If the WDT is enabled during Sleep, the WDT resumes counting.

When the device exits Sleep, the WDT is cleared again.

When a WDT time-out occurs while the device is in Sleep, no Reset is generated. Instead, the device wakes up and resumes operation. The $\overline{\text{TO}}$ and $\overline{\text{PD}}$ bits in the STATUS register are changed to indicate the event. See **Section 3.0 “Memory Organization”** and STATUS register (**Register 3-1**) for more information.

TABLE 9-2: WDT CLEARING CONDITIONS

Conditions	WDT
WDTE<1:0> = `00`	Cleared
WDTE<1:0> = `01` and SWDTEN = `0`	
WDTE<1:0> = `10` and enter Sleep	
`CLRWDT` Command	
Oscillator Fail Detected	
Exit Sleep + System Clock = T1OSC, EXTRC, INTOSC, EXTCLK	
Change INTOSC divider (IRCF bits)	Unaffected

9.6 Watchdog Control Register

REGISTER 9-1: WDTCON: WATCHDOG TIMER CONTROL REGISTER

U-0	U-0	R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1	R/W-1/1	R/W-0/0
—	—	WDTPS<4:0>					SWDTEN
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-m/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-6 **Unimplemented:** Read as ‘0’

bit 5-1 **WDTPS<4:0>:** Watchdog Timer Period Select bits(1)

Bit Value = Prescale Rate

`00000` = 1:32 (Interval 1 ms nominal)
`00001` = 1:64 (Interval 2 ms nominal)
`00010` = 1:128 (Interval 4 ms nominal)
`00011` = 1:256 (Interval 8 ms nominal)
`00100` = 1:512 (Interval 16 ms nominal)
`00101` = 1:1024 (Interval 32 ms nominal)
`00110` = 1:2048 (Interval 64 ms nominal)
`00111` = 1:4096 (Interval 128 ms nominal)
`01000` = 1:8192 (Interval 256 ms nominal)
`01001` = 1:16384 (Interval 512 ms nominal)
`01010` = 1:32768 (Interval 1s nominal)
`01011` = 1:65536 (Interval 2s nominal) (Reset value)
`01100` = 1:131072 (2^{17}) (Interval 4s nominal)
`01101` = 1:262144 (2^{18}) (Interval 8s nominal)
`01110` = 1:524288 (2^{19}) (Interval 16s nominal)
`01111` = 1:1048576 (2^{20}) (Interval 32s nominal)
`10000` = 1:2097152 (2^{21}) (Interval 64s nominal)
`10001` = 1:4194304 (2^{22}) (Interval 128s nominal)
`10010` = 1:8388608 (2^{23}) (Interval 256s nominal)

`10011` = Reserved. Results in minimum interval (1:32)
•
•
•
`11111` = Reserved. Results in minimum interval (1:32)

bit 0 **SWDTEN:** Software Enable/Disable for Watchdog Timer bit

If WDTE<1:0> = `00`:
This bit is ignored.
If WDTE<1:0> = `01`:
`1` = WDT is turned on
`0` = WDT is turned off
If WDTE<1:0> = `1x`:
This bit is ignored.

Note 1: Times are approximate. WDT time is based on 31 kHz LFINTOSC.

TABLE 9-3: SUMMARY OF REGISTERS ASSOCIATED WITH WATCHDOG TIMER

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
OSCCON	—	IRCF<3:0>				—	SCS<1:0>		53
STATUS	—	—	—	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Z	DC	C	16
WDTCON	—	—	WDTPS<4:0>					SWDTEN	70

Legend: x = unknown, u = unchanged, – = unimplemented locations read as ‘0’. Shaded cells are not used by Watchdog Timer.

TABLE 9-4: SUMMARY OF CONFIGURATION WORD WITH WATCHDOG TIMER

Name	Bits	Bit -/7	Bit -/6	Bit 13/5	Bit 12/4	Bit 11/3	Bit 10/2	Bit 9/1	Bit 8/0	Register on Page
CONFIG1	13:8	—	—	—	—	$\overline{\text{CLKOUTEN}}$	BOREN<1:0>		—	34
	7:0	$\overline{\text{CP}}$	MCLRE	$\overline{\text{PWRTE}}$	WDTE<1:0>		—	FOSC<1:0>		

Legend: — = unimplemented location, read as ‘0’. Shaded cells are not used by Watchdog Timer.

10.0 FLASH PROGRAM MEMORY CONTROL

The Flash program memory is readable and writable during normal operation over the full VDD range. Program memory is indirectly addressed using Special Function Registers (SFRs). The SFRs used to access program memory are:

- PMCON1
- PMCON2
- PMDATL
- PMDATH
- PMADRL
- PMADRH

When accessing the program memory, the PMDATH:PMDATL register pair forms a 2-byte word that holds the 14-bit data for read/write, and the PMADRH:PMADRL register pair forms a 2-byte word that holds the 15-bit address of the program memory location being read.

The write time is controlled by an on-chip timer. The write/erase voltages are generated by an on-chip charge pump rated to operate over the operating voltage range of the device.

The Flash Program Memory can be protected in two ways; by code protection ($\overline{\text{CP}}$ bit in Configuration Word 1) and write protection (WRT<1:0> bits in Configuration Word 2).

Code protection ($\overline{\text{CP}}$ = `0`)[(1)], disables access, reading and writing, to the Flash Program Memory via external device programmers. Code protection does not affect the self-write and erase functionality. Code protection can only be reset by a device programmer performing a Bulk Erase to the device, clearing all Flash Program Memory, Configuration bits and User IDs.

Write protection prohibits self-write and erase to a portion or all of the Flash Program Memory as defined by the bits WRT<1:0>. Write protection does not affect a device programmers ability to read, write or erase the device.

Note 1: Code protection of the entire Flash Program Memory array is enabled by clearing the $\overline{\text{CP}}$ bit of Configuration Word 1.

10.1 PMADRL and PMADRH Registers

The PMADRH:PMADRL register pair can address up to a maximum of 32K words of program memory. When selecting a program address value, the MSB of the address is written to the PMADRH register and the LSB is written to the PMADRL register.

10.1.1 PMCON1 AND PMCON2 REGISTERS

PMCON1 is the control register for Flash Program Memory accesses.

Control bits RD and WR initiate read and write, respectively. These bits cannot be cleared, only set, in software. They are cleared by hardware at completion of the read or write operation. The inability to clear the WR bit in software prevents the accidental, premature termination of a write operation.

The WREN bit, when set, will allow a write operation to occur. On power-up, the WREN bit is clear. The WRERR bit is set when a write operation is interrupted by a Reset during normal operation. In these situations, following Reset, the user can check the WRERR bit and execute the appropriate error handling routine.

The PMCON2 register is a write-only register. Attempting to read the PMCON2 register will return all '`0`'s.

To enable writes to the program memory, a specific pattern (the unlock sequence), must be written to the PMCON2 register. The required unlock sequence prevents inadvertent writes to the program memory write latches and Flash Program Memory.

10.2 Flash Program Memory Overview

It is important to understand the Flash Program Memory structure for erase and programming operations. Flash Program Memory is arranged in rows. A row consists of a fixed number of 14-bit program memory words. A row is the minimum size that can be erased by user software.

After a row has been erased, the user can reprogram all or a portion of this row. Data to be written into the program memory row is written to 14-bit wide data write latches. These write latches are not directly accessible to the user, but may be loaded via sequential writes to the PMDATH:PMDATL register pair.

Note: If the user wants to modify only a portion of a previously programmed row, then the contents of the entire row must be read and saved in RAM prior to the erase. Then, new data and retained data can be written into the write latches to reprogram the row of Flash Program Memory. However, any unprogrammed locations can be written without first erasing the row. In this case, it is not necessary to save and rewrite the other previously programmed locations.

See Table 10-1 for Erase Row size and the number of write latches for Flash Program Memory.

TABLE 10-1: FLASH MEMORY ORGANIZATION BY DEVICE

Device	Row Erase (words)	Write Latches (words)
PIC16(L)F1902	32	32
PIC16(L)F1903		

10.2.1 READING THE FLASH PROGRAM MEMORY

To read a program memory location, the user must:

1. Write the desired address to the PMADRH:PMADRL register pair.
2. Clear the CFGS bit of the PMCON1 register.
3. Then, set control bit RD of the PMCON1 register.

Once the read control bit is set, the program memory Flash controller will use the second instruction cycle to read the data. This causes the second instruction immediately following the “`BSF PMCON1,RD`” instruction to be ignored. The data is available in the very next cycle, in the PMDATH:PMDATL register pair; therefore, it can be read as two bytes in the following instructions.

PMDATH:PMDATL register pair will hold this value until another read or until it is written to by the user.

Note: The two instructions following a program memory read are required to be `NOP`s. This prevents the user from executing a 2-cycle instruction on the next instruction after the RD bit is set.

FIGURE 10-1: FLASH PROGRAM MEMORY READ FLOWCHART

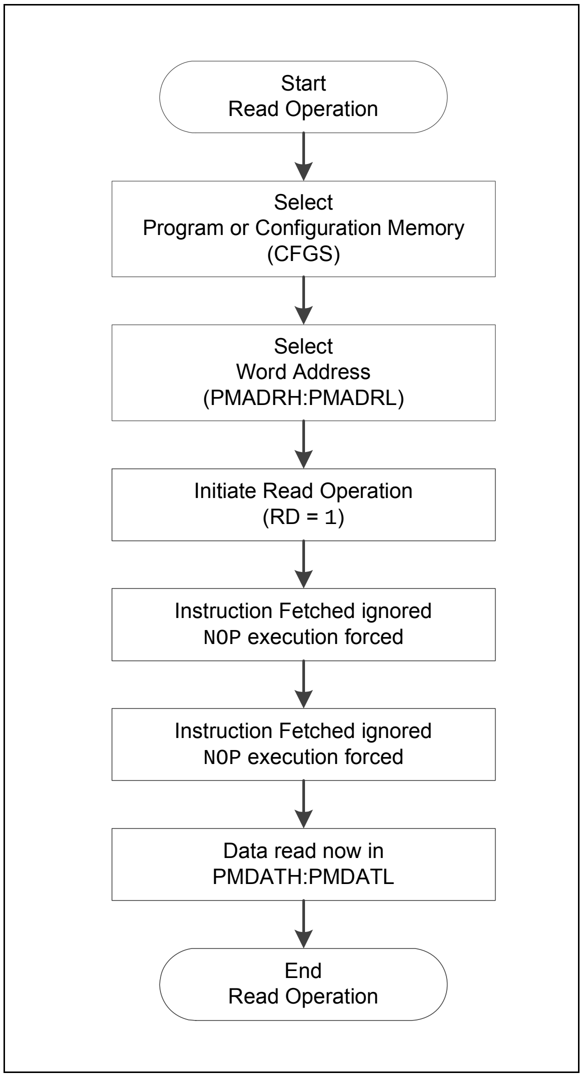
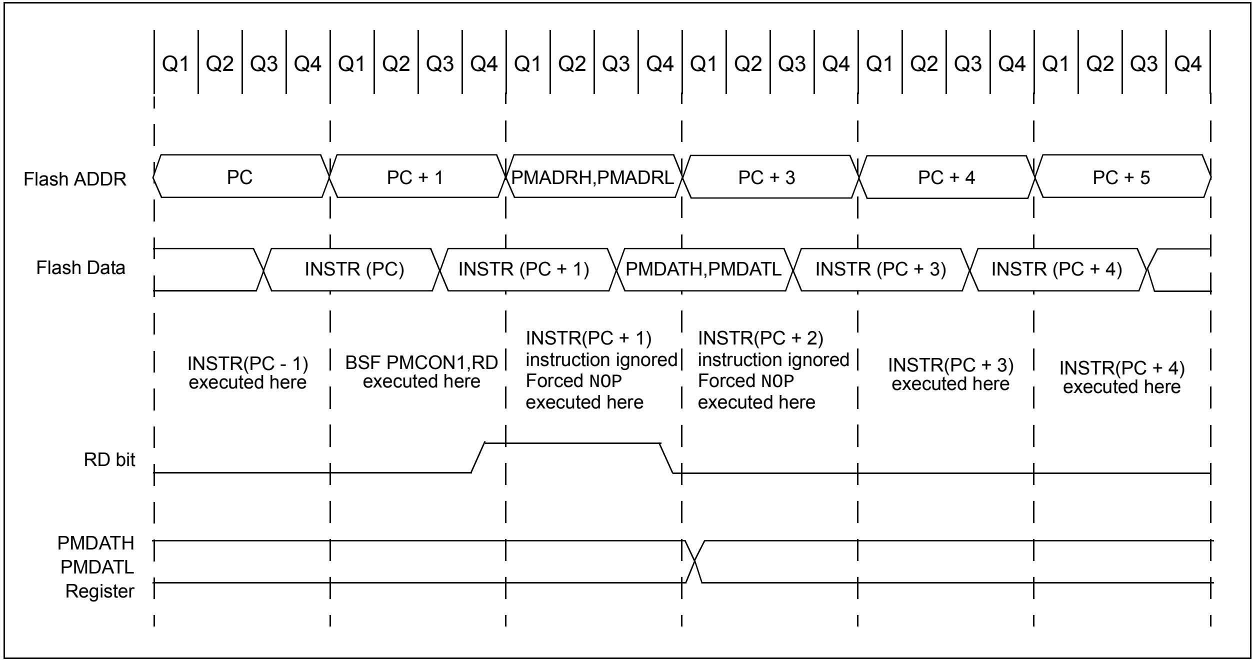

Start
Read Operation
Select
Program or Configuration Memory
(CFGS)
Select
Word Address
(PMADRH:PMADRL)
Initiate Read Operation
(RD = 1)
Instruction Fetched ignored
NOP execution forced
Instruction Fetched ignored
NOP execution forced
Data read now in
PMDATH:PMDATL
End
Read Operation

FIGURE 10-2: FLASH PROGRAM MEMORY READ CYCLE EXECUTION


Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4
Flash ADDR
PC
PC + 1
PMADRH,PMADRL
PC + 3
PC + 4
PC + 5
Flash Data
INSTR (PC)
INSTR (PC + 1)
PMDATH,PMDATL
INSTR (PC + 3)
INSTR (PC + 4)
INSTR(PC - 1) executed here
BSF PMCON1,RD executed here
INSTR(PC + 1) instruction ignored Forced NOP executed here
INSTR(PC + 2) instruction ignored Forced NOP executed here
INSTR(PC + 3) executed here
INSTR(PC + 4) executed here
RD bit
PMDATH
PMDATL
Register

EXAMPLE 10-1: FLASH PROGRAM MEMORY READ

```
* This code block will read 1 word of program
* memory at the memory address:
   PROG_ADDR_HI : PROG_ADDR_LO
*   data will be returned in the variables;
*   PROG_DATA_HI, PROG_DATA_LO

    BANKSEL   PMADRL            ; Select Bank for PMCON registers
    MOVLW     PROG_ADDR_LO      ;
    MOVWF     PMADRL            ; Store LSB of address
    MOVLW     PROG_ADDR_HI      ;
    MOVWL     PMADRH            ; Store MSB of address

    BCF       PMCON1,CFGS       ; Do not select Configuration Space
    BSF       PMCON1,RD         ; Initiate read
    NOP                         ; Ignored (Figure 10-2)
    NOP                         ; Ignored (Figure 10-2)

    MOVF      PMDATL,W          ; Get LSB of word
    MOVWF     PROG_DATA_LO      ; Store in user location
    MOVF      PMDATH,W          ; Get MSB of word
    MOVWF     PROG_DATA_HI      ; Store in user location
```

10.2.2 FLASH MEMORY UNLOCK SEQUENCE

The unlock sequence is a mechanism that protects the Flash Program Memory from unintended self-write programming or erasing. The sequence must be executed and completed without interruption to successfully complete any of the following operations:

- Row Erase
- Load program memory write latches
- Write of program memory write latches to program memory
- Write of program memory write latches to User IDs

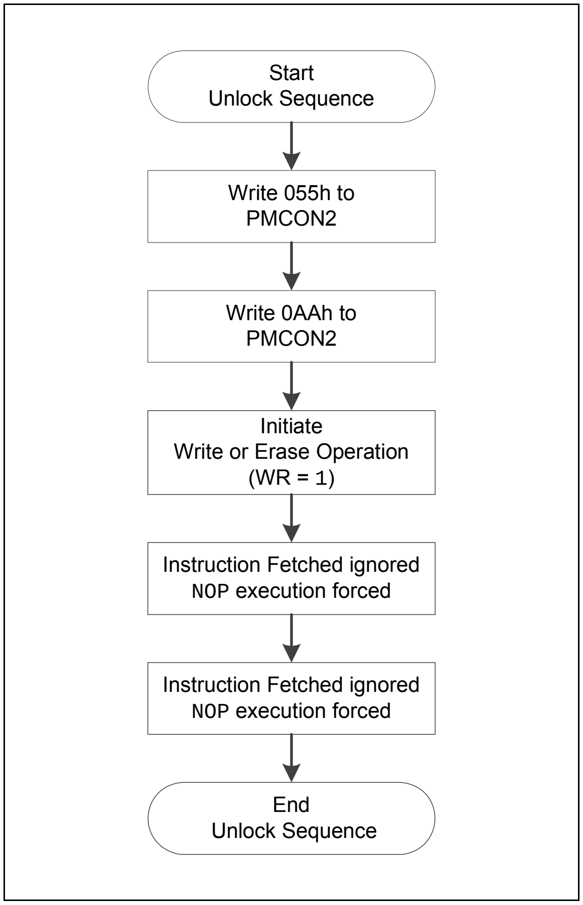
The unlock sequence consists of the following steps:

1. Write 55h to PMCON2
2. Write AAh to PMCON2
3. Set the WR bit in PMCON1
4. `NOP` instruction
5. `NOP` instruction

Once the WR bit is set, the processor will always force two `NOP` instructions. When an Erase Row or Program Row operation is being performed, the processor will stall internal operations (typical 2 ms), until the operation is complete and then resume with the next instruction. When the operation is loading the program memory write latches, the processor will always force the two `NOP` instructions and continue uninterrupted with the next instruction.

Since the unlock sequence must not be interrupted, global interrupts should be disabled prior to the unlock sequence and re-enabled after the unlock sequence is completed.

FIGURE 10-3: FLASH PROGRAM MEMORY UNLOCK SEQUENCE FLOWCHART


Start
Unlock Sequence
Write 055h to
PMCON2
Write 0AAh to
PMCON2
Initiate
Write or Erase Operation
(WR = 1)
Instruction Fetched ignored
NOP execution forced
Instruction Fetched ignored
NOP execution forced
End
Unlock Sequence

10.2.3 ERASING FLASH PROGRAM MEMORY

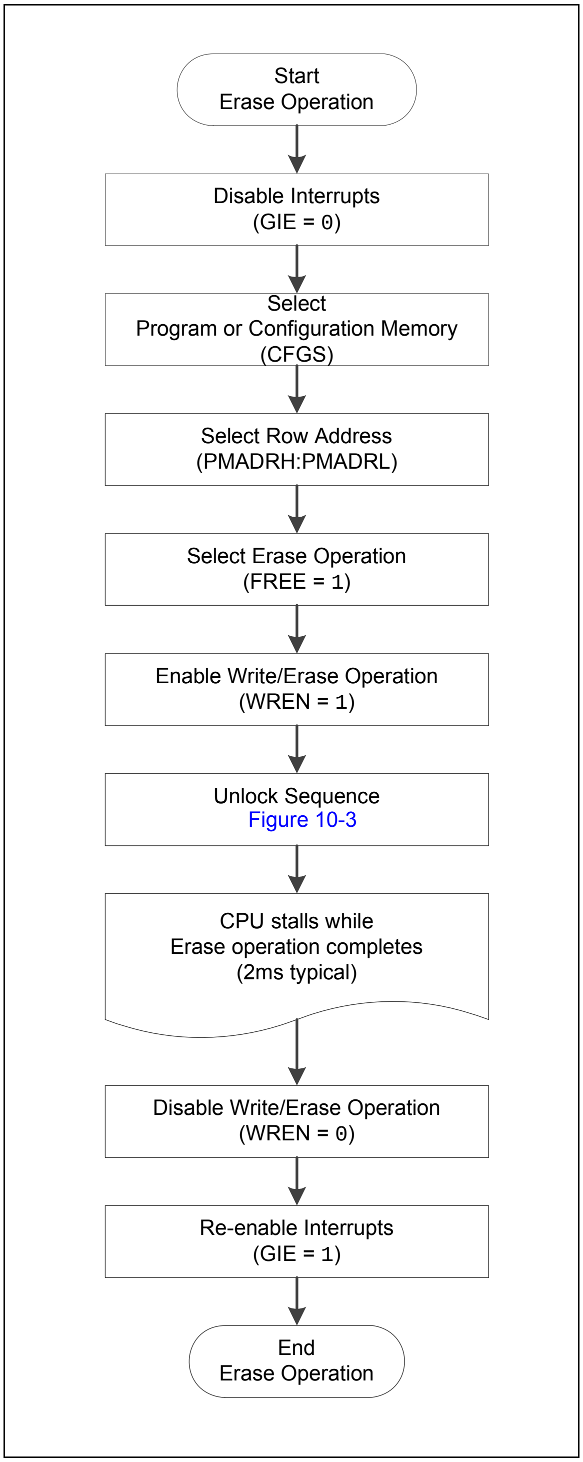
While executing code, program memory can only be erased by rows. To erase a row:

1. Load the PMADRH:PMADRL register pair with any address within the row to be erased.
2. Clear the CFGS bit of the PMCON1 register.
3. Set the FREE and WREN bits of the PMCON1 register.
4. Write 55h, then AAh, to PMCON2 (Flash programming unlock sequence).
5. Set control bit WR of the PMCON1 register to begin the erase operation.

See Example 10-2.

After the “`BSF PMCON1,WR`” instruction, the processor requires two cycles to set up the erase operation. The user must place two `NOP` instructions after the WR bit is set. The processor will halt internal operations for the typical 2 ms erase time. This is not Sleep mode as the clocks and peripherals will continue to run. After the erase cycle, the processor will resume operation with the third instruction after the PMCON1 `WRITE` instruction.

FIGURE 10-4: FLASH PROGRAM MEMORY ERASE FLOWCHART


Start
Erase Operation
Disable Interrupts
(GIE = 0)
Select
Program or Configuration Memory
(CFGS)
Select Row Address
(PMADRH:PMADRL)
Select Erase Operation
(FREE = 1)
Enable Write/Erase Operation
(WREN = 1)
Unlock Sequence
Figure 10-3
CPU stalls while
Erase operation completes
(2ms typical)
Disable Write/Erase Operation
(WREN = 0)
Re-enable Interrupts
(GIE = 1)
End
Erase Operation

EXAMPLE 10-2: ERASING ONE ROW OF PROGRAM MEMORY

```
; This row erase routine assumes the following:
; 1. A valid address within the erase row is loaded in ADDRH:ADDRL
; 2. ADDRH and ADDRL are located in shared data memory 0x70 - 0x7F (common RAM)


        BCF        INTCON,GIE     ; Disable ints so required sequences will execute properly
        BANKSEL    PMADRL
        MOVF       ADDRL,W        ; Load lower 8 bits of erase address boundary
        MOVWF      PMADRL
        MOVF       ADDRH,W        ; Load upper 6 bits of erase address boundary
        MOVWF      PMADRH
        BCF        PMCON1,CFGS    ; Not configuration space
        BSF        PMCON1,FREE    ; Specify an erase operation
        BSF        PMCON1,WREN    ; Enable writes
```

Required Sequence

```
        MOVLW      55h            ; Start of required sequence to initiate erase
        MOVWF      PMCON2         ; Write 55h
        MOVLW      0AAh           ;
        MOVWF      PMCON2         ; Write AAh
        BSF        PMCON1,WR      ; Set WR bit to begin erase
        NOP                       ; NOP instructions are forced as processor starts
        NOP                       ; row erase of program memory.
```

```
                                  ;
                                  ; The processor stalls until the erase process is complete
                                  ; after erase processor continues with 3rd instruction

        BCF        PMCON1,WREN    ; Disable writes
        BSF        INTCON,GIE     ; Enable interrupts
```

10.2.4 WRITING TO FLASH PROGRAM MEMORY

Program memory is programmed using the following steps:

1. Load the address in PMADRH:PMADRL of the row to be programmed.
2. Load each write latch with data.
3. Initiate a programming operation.
4. Repeat steps 1 through 3 until all data is written.

Before writing to program memory, the word(s) to be written must be erased or previously unwritten. Program memory can only be erased one row at a time. No automatic erase occurs upon the initiation of the write.

Program memory can be written one or more words at a time. The maximum number of words written at one time is equal to the number of write latches. See Figure 10-5 (row writes to program memory with 32 write latches) for more details.

The write latches are aligned to the Flash row address boundary defined by the upper ten bits of PMADRH:PMADRL, (PMADRH<6:0>:PMADRL<7:5>) with the lower five bits of PMADRL, (PMADRL<4:0>) determining the write latch being loaded. Write operations do not cross these boundaries. At the completion of a program memory write operation, the data in the write latches is reset to contain 0x3FFF.

The following steps should be completed to load the write latches and program a row of program memory. These steps are divided into two parts. First, each write latch is loaded with data from the PMDATH:PMDATL using the unlock sequence with LWLO = `1`. When the last word to be loaded into the write latch is ready, the LWLO bit is cleared and the unlock sequence executed. This initiates the programming operation, writing all the latches into Flash Program Memory.

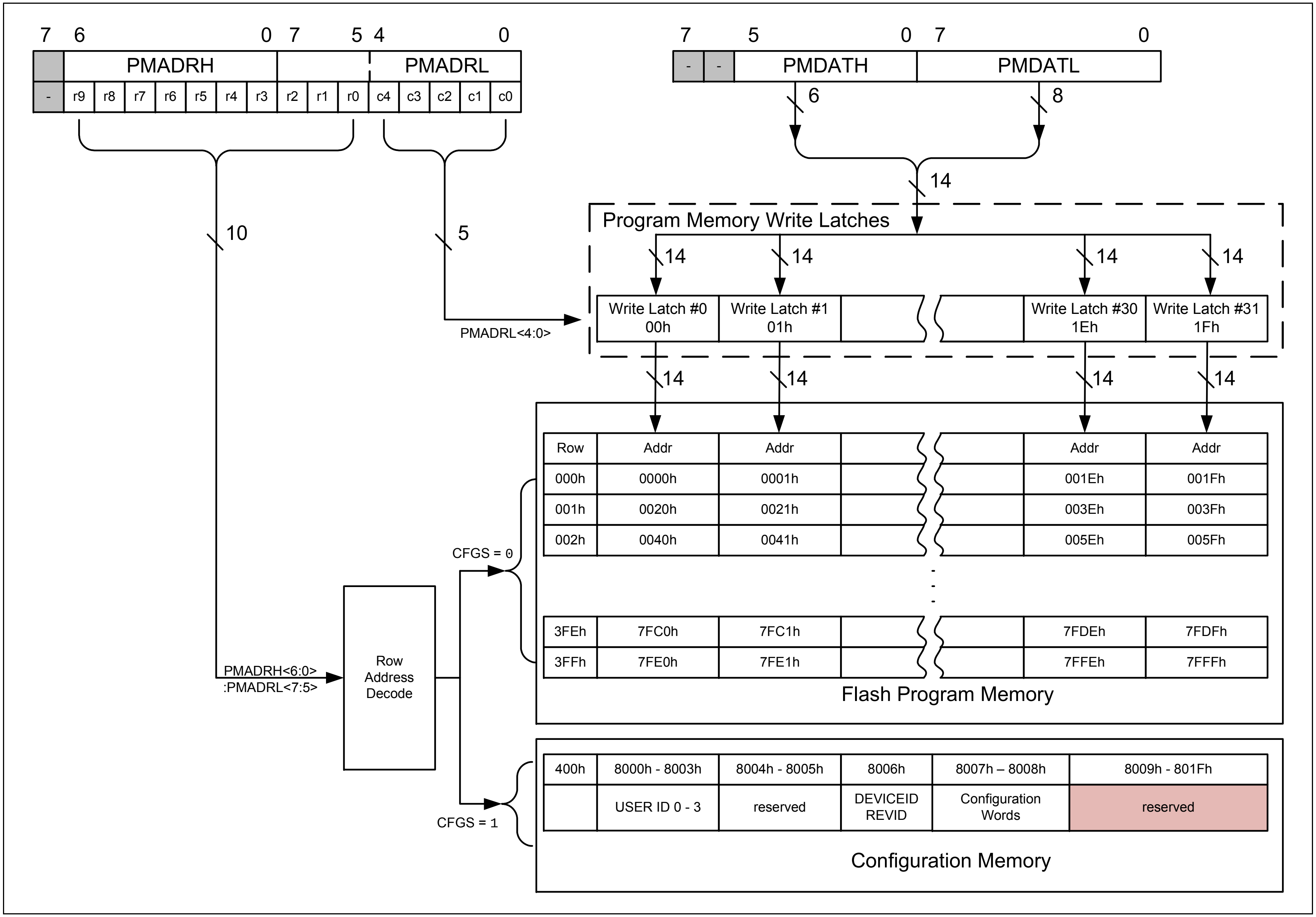
Note:	The special unlock sequence is required to load a write latch with data or initiate a Flash programming operation. If the unlock sequence is interrupted, writing to the latches or program memory will not be initiated.

1. Set the WREN bit of the PMCON1 register.
2. Clear the CFGS bit of the PMCON1 register.
3. Set the LWLO bit of the PMCON1 register. When the LWLO bit of the PMCON1 register is ‘`1`’, the write sequence will only load the write latches and will not initiate the write to Flash Program Memory.
4. Load the PMADRH:PMADRL register pair with the address of the location to be written.
5. Load the PMDATH:PMDATL register pair with the program memory data to be written.
6. Execute the unlock sequence (**Section 10.2.2 “Flash Memory Unlock Sequence”**). The write latch is now loaded.
7. Increment the PMADRH:PMADRL register pair to point to the next location.
8. Repeat steps 5 through 7 until all but the last write latch has been loaded.
9. Clear the LWLO bit of the PMCON1 register. When the LWLO bit of the PMCON1 register is ‘`0`’, the write sequence will initiate the write to Flash Program Memory.
10. Load the PMDATH:PMDATL register pair with the program memory data to be written.
11. Execute the unlock sequence (**Section 10.2.2 “Flash Memory Unlock Sequence”**). The entire program memory latch content is now written to Flash Program Memory.

Note:	The program memory write latches are reset to the blank state (0x3FFF) at the completion of every write or erase operation. As a result, it is not necessary to load all the program memory write latches. Unloaded latches will remain in the blank state.

An example of the complete write sequence is shown in Example 10-3. The initial address is loaded into the PMADRH:PMADRL register pair; the data is loaded using indirect addressing.

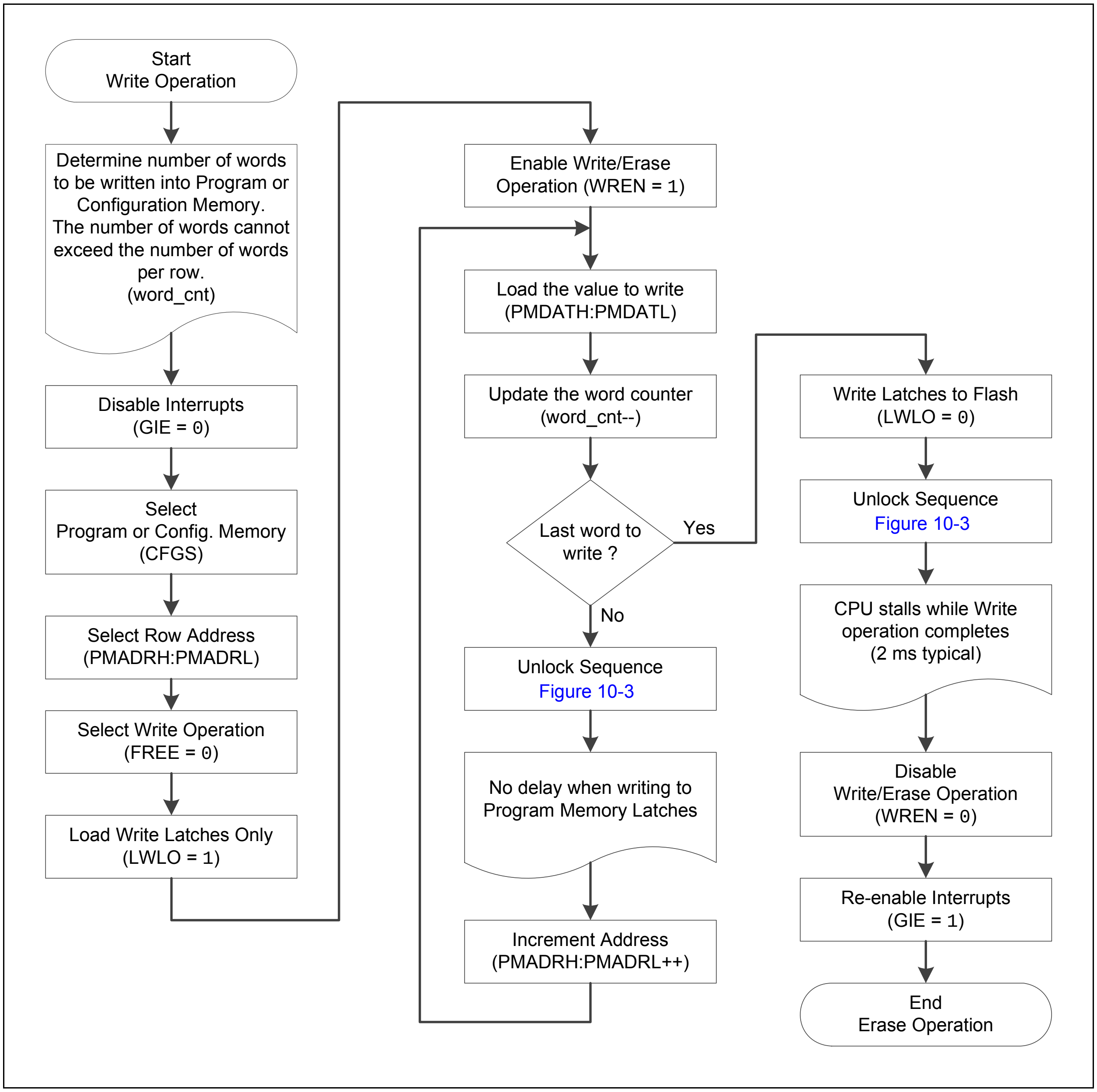
FIGURE 10-5: BLOCK WRITES TO FLASH PROGRAM MEMORY WITH 32 WRITE LATCHES


7 6 0 7 5 4 0
PMADRH
PMADRL
- r9 r8 r7 r6 r5 r4 r3 r2 r1 r0 c4 c3 c2 c1 c0
10
5
PMADRL<4:0>
PMADRH<6:0>
:PMADRL<7:5>
Row Address Decode
CFGS = 0
CFGS = 1
7 5 0 7 0
- -
PMDATH
PMDATL
6
8
14
Program Memory Write Latches
14
14
14
14
Write Latch #0 00h
Write Latch #1 01h
Write Latch #30 1Eh
Write Latch #31 1Fh
14
14
14
14
Flash Program Memory
Configuration Memory

Row	Addr	Addr		Addr	Addr
000h	0000h	0001h		001Eh	001Fh
001h	0020h	0021h		003Eh	003Fh
002h	0040h	0041h		005Eh	005Fh
3FEh	7FC0h	7FC1h		7FDEh	7FDFh
3FFh	7FE0h	7FE1h		7FFEh	7FFFh

400h	8000h - 8003h	8004h - 8005h	8006h	8007h – 8008h	8009h - 801Fh
	USER ID 0 - 3	reserved	DEVICEID REVID	Configuration Words	reserved

FIGURE 10-6: FLASH PROGRAM MEMORY WRITE FLOWCHART


Start
Write Operation
Determine number of words to be written into Program or Configuration Memory. The number of words cannot exceed the number of words per row. (word_cnt)
Disable Interrupts (GIE = 0)
Select Program or Config. Memory (CFGS)
Select Row Address (PMADRH:PMADRL)
Select Write Operation (FREE = 0)
Load Write Latches Only (LWLO = 1)
Enable Write/Erase Operation (WREN = 1)
Load the value to write (PMDATH:PMDATL)
Update the word counter (word_cnt--)
Last word to write ?
Yes
No
Unlock Sequence Figure 10-3
No delay when writing to Program Memory Latches
Increment Address (PMADRH:PMADRL++)
Write Latches to Flash (LWLO = 0)
Unlock Sequence Figure 10-3
CPU stalls while Write operation completes (2 ms typical)
Disable Write/Erase Operation (WREN = 0)
Re-enable Interrupts (GIE = 1)
End
Erase Operation

EXAMPLE 10-3: WRITING TO FLASH PROGRAM MEMORY

```
; This write routine assumes the following:
; 1. 64 bytes of data are loaded, starting at the address in DATA_ADDR
; 2. Each word of data to be written is made up of two adjacent bytes in DATA_ADDR,
; stored in little endian format
; 3. A valid starting address (the least significant bits = 00000) is loaded in ADDRH:ADDRL
; 4. ADDRH and ADDRL are located in shared data memory 0x70 - 0x7F (common RAM)
;
        BCF         INTCON,GIE      ; Disable ints so required sequences will execute properly
        BANKSEL     PMADRH          ; Bank 3
        MOVF        ADDRH,W         ; Load initial address
        MOVWF       PMADRH          ;
        MOVF        ADDRL,W         ;
        MOVWF       PMADRL          ;
        MOVLW       LOW DATA_ADDR   ; Load initial data address
        MOVWF       FSR0L           ;
        MOVLW       HIGH DATA_ADDR  ; Load initial data address
        MOVWF       FSR0H           ;
        BCF         PMCON1,CFGS     ; Not configuration space
        BSF         PMCON1,WREN     ; Enable writes
        BSF         PMCON1,LWLO     ; Only Load Write Latches
LOOP
        MOVIW       FSR0++          ; Load first data byte into lower
        MOVWF       PMDATL          ;
        MOVIW       FSR0++          ; Load second data byte into upper
        MOVWF       PMDATH          ;

        MOVF        PMADRL,W        ; Check if lower bits of address are '00000'
        XORLW       0x1F            ; Check if we're on the last of 32 addresses
        ANDLW       0x1F            ;
        BTFSC       STATUS,Z        ; Exit if last of 32 words,
        GOTO        START_WRITE     ;
```

Required Sequence

```
        MOVLW       55h             ; Start of required write sequence:
        MOVWF       PMCON2          ; Write 55h
        MOVLW       0AAh            ;
        MOVWF       PMCON2          ; Write AAh
        BSF         PMCON1,WR       ; Set WR bit to begin write
        NOP                         ; NOP instructions are forced as processor
                                    ; loads program memory write latches
        NOP                         ;
```

```
        INCF        PMADRL,F        ; Still loading latches Increment address
        GOTO        LOOP            ; Write next latches

START_WRITE
        BCF         PMCON1,LWLO     ; No more loading latches - Actually start Flash program
                                    ; memory write
```

Required Sequence

```
        MOVLW       55h             ; Start of required write sequence:
        MOVWF       PMCON2          ; Write 55h
        MOVLW       0AAh            ;
        MOVWF       PMCON2          ; Write AAh
        BSF         PMCON1,WR       ; Set WR bit to begin write
        NOP                         ; NOP instructions are forced as processor writes
                                    ; all the program memory write latches simultaneously
        NOP                         ; to program memory.
```

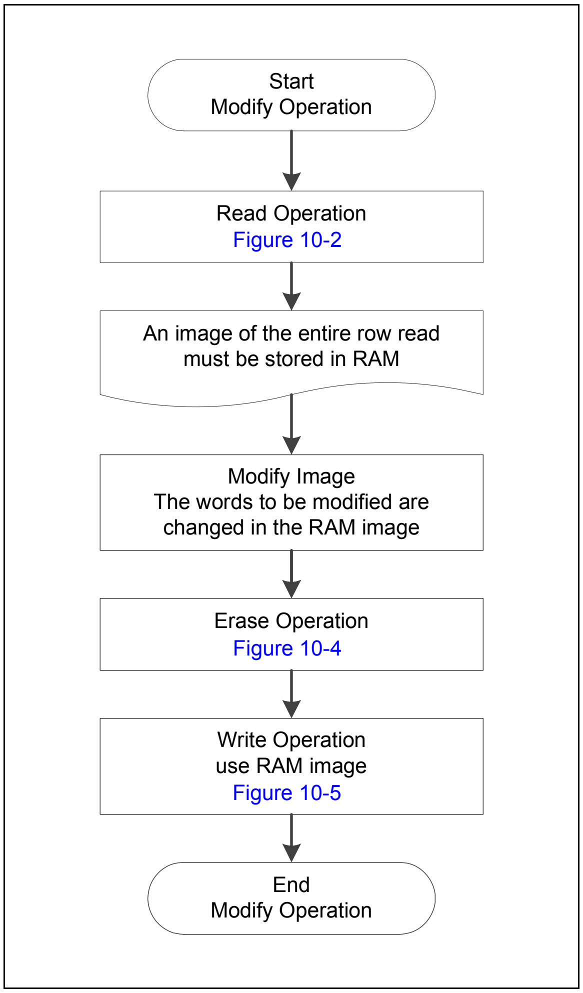
```
                                    ; After NOPs, the processor
                                    ; stalls until the self-write process in complete
                                    ; after write processor continues with 3rd instruction
        BCF         PMCON1,WREN     ; Disable writes
        BSF         INTCON,GIE      ; Enable interrupts
```

10.3 Modifying Flash Program Memory

When modifying existing data in a program memory row, and data within that row must be preserved, it must first be read and saved in a RAM image. Program memory is modified using the following steps:

1. Load the starting address of the row to be modified.
2. Read the existing data from the row into a RAM image.
3. Modify the RAM image to contain the new data to be written into program memory.
4. Load the starting address of the row to be rewritten.
5. Erase the program memory row.
6. Load the write latches with data from the RAM image.
7. Initiate a programming operation.

FIGURE 10-7: FLASH PROGRAM MEMORY MODIFY FLOWCHART


Start
Modify Operation
Read Operation
Figure 10-2
An image of the entire row read must be stored in RAM
Modify Image
The words to be modified are changed in the RAM image
Erase Operation
Figure 10-4
Write Operation
use RAM image
Figure 10-5
End
Modify Operation

10.4 User ID, Device ID and Configuration Word Access

Instead of accessing program memory, the User ID's, Device ID/Revision ID and Configuration Words can be accessed when CFGS = `1` in the PMCON1 register. This is the region that would be pointed to by PC<15> = `1`, but not all addresses are accessible. Different access may exist for reads and writes. Refer to Table 10-2.

When read access is initiated on an address outside the parameters listed in Table 10-2, the PMDATH:PMDATL register pair is cleared, reading back '`0`'s.

TABLE 10-2: USER ID, DEVICE ID AND CONFIGURATION WORD ACCESS (CFGS = 1)

Address	Function	Read Access	Write Access
8000h-8003h	User IDs	Yes	Yes
8006h	Device ID/Revision ID	Yes	No
8007h-8008h	Configuration Words 1 and 2	Yes	No

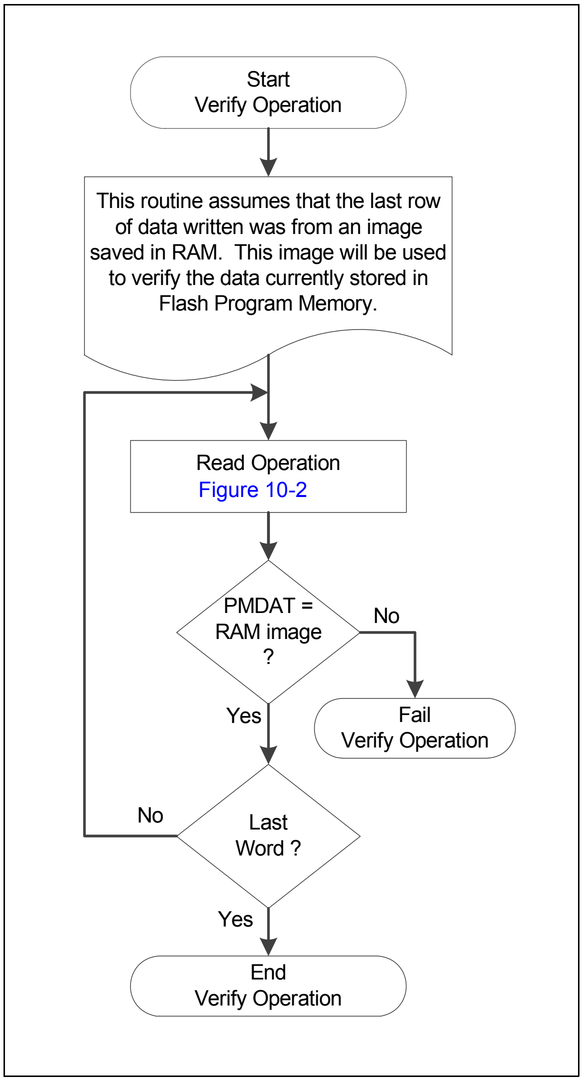
EXAMPLE 10-4: CONFIGURATION WORD AND DEVICE ID ACCESS

```
* This code block will read 1 word of program memory at the memory address:
*   PROG_ADDR_LO (must be 00h-08h) data will be returned in the variables;
*   PROG_DATA_HI, PROG_DATA_LO

    BANKSEL  PMADRL           ; Select correct Bank
    MOVLW    PROG_ADDR_LO     ;
    MOVWF    PMADRL           ; Store LSB of address
    CLRF     PMADRH           ; Clear MSB of address

    BSF      PMCON1,CFGS      ; Select Configuration Space
    BCF      INTCON,GIE       ; Disable interrupts
    BSF      PMCON1,RD        ; Initiate read
    NOP                       ; Executed (See Figure 10-2)
    NOP                       ; Ignored (See Figure 10-2)
    BSF      INTCON,GIE       ; Restore interrupts

    MOVF     PMDATL,W         ; Get LSB of word
    MOVWF    PROG_DATA_LO     ; Store in user location
    MOVF     PMDATH,W         ; Get MSB of word
    MOVWF    PROG_DATA_HI     ; Store in user location
```

10.5 Write Verify

It is considered good programming practice to verify that program memory writes agree with the intended value. Since program memory is stored as a full page then the stored program memory contents are compared with the intended data stored in RAM after the last write is complete.

FIGURE 10-8: FLASH PROGRAM MEMORY VERIFY FLOWCHART


Start
Verify Operation
This routine assumes that the last row of data written was from an image saved in RAM. This image will be used to verify the data currently stored in Flash Program Memory.
Read Operation
Figure 10-2
PMDAT = RAM image ?
No
Fail
Verify Operation
Yes
Last Word ?
No
Yes
End
Verify Operation

10.6 Flash Program Memory Control Registers

REGISTER 10-1: PMDATL: PROGRAM MEMORY DATA LOW BYTE REGISTER

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
PMDAT<7:0>							
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0	**PMDAT<7:0>**: Read/write value for Least Significant bits of program memory

REGISTER 10-2: PMDATH: PROGRAM MEMORY DATA HIGH BYTE REGISTER

U-0	U-0	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
—	—	PMDAT<13:8>					
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-6	**Unimplemented:** Read as ‘0’
bit 5-0	**PMDAT<13:8>**: Read/write value for Most Significant bits of program memory

REGISTER 10-3: PMADRL: PROGRAM MEMORY ADDRESS LOW BYTE REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
PMADR<7:0>							
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0	**PMADR<7:0>**: Specifies the Least Significant bits for program memory address

REGISTER 10-4: PMADRH: PROGRAM MEMORY ADDRESS HIGH BYTE REGISTER

U-1	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	PMADR<14:8>						
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7	**Unimplemented:** Read as ‘1’
bit 6-0	**PMADR<14:8>**: Specifies the Most Significant bits for program memory address

REGISTER 10-5: PMCON1: PROGRAM MEMORY CONTROL 1 REGISTER

U-1[1]	R/W-0/0	R/W-0/0	R/W/HC-0/0	R/W/HC-x/q[2]	R/W-0/0	R/S/HC-0/0	R/S/HC-0/0
—	CFGS	LWLO	FREE	WRERR	WREN	WR	RD
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
S = Bit can only be set	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	HC = Bit is cleared by hardware

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

bit 6 **CFGS:** Configuration Select bit
1 = Access configuration, User ID and Device ID registers
0 = Access Flash Program Memory

bit 5 **LWLO:** Load Write Latches Only bit[3]
1 = Only the addressed program memory write latch is loaded/updated on the next WR command
0 = The addressed program memory write latch is loaded/updated and a write of all program memory write latches will be initiated on the next WR command

bit 4 **FREE:** Program Flash Erase Enable bit
1 = Performs an erase operation on the next WR command (hardware cleared upon completion)
0 = Performs an write operation on the next WR command

bit 3 **WRERR:** Program/Erase Error Flag bit
1 = Condition indicates an improper program or erase sequence attempt or termination (bit is set automatically on any set attempt (write '1') of the WR bit).
0 = The program or erase operation completed normally.

bit 2 **WREN:** Program/Erase Enable bit
1 = Allows program/erase cycles
0 = Inhibits programming/erasing of program Flash

bit 1 **WR:** Write Control bit
1 = Initiates a program Flash program/erase operation.
The operation is self-timed and the bit is cleared by hardware once operation is complete.
The WR bit can only be set (not cleared) in software.
0 = Program/erase operation to the Flash is complete and inactive.

bit 0 **RD:** Read Control bit
1 = Initiates a program Flash read. Read takes one cycle. RD is cleared in hardware. The RD bit can only be set (not cleared) in software.
0 = Does not initiate a program Flash read.

Note 1: Unimplemented bit, read as '1'.
2: The WRERR bit is automatically set by hardware when a program memory write or erase operation is started (WR = 1).
3: The LWLO bit is ignored during a program memory erase operation (FREE = 1).

REGISTER 10-6: PMCON2: PROGRAM MEMORY CONTROL 2 REGISTER

W-0/0	W-0/0	W-0/0	W-0/0	W-0/0	W-0/0	W-0/0	W-0/0
Program Memory Control Register 2							
bit 7							bit 0

Legend:

R = Readable bit
W = Writable bit
U = Unimplemented bit, read as ‘0’
S = Bit can only be set
x = Bit is unknown
-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set
‘0’ = Bit is cleared

bit 7-0	**Flash Memory Unlock Pattern bits** To unlock writes, a 55h must be written first, followed by an AAh, before setting the WR bit of the PMCON1 register. The value written to this register is used to unlock the writes. There are specific timing requirements on these writes.

TABLE 10-3: SUMMARY OF REGISTERS ASSOCIATED WITH FLASH PROGRAM MEMORY

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
PMCON1	—(1)	CFGS	LWLO	FREE	WRERR	WREN	WR	RD	86
PMCON2	Program Memory Control Register 2								87
PMADRL	PMADRL<7:0>								85
PMADRH	—(1)	PMADRH<6:0>							85
PMDATL	PMDATL<7:0>								85
PMDATH	—	—	PMDATH<5:0>						85
INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	60

Legend: — = unimplemented location, read as ‘`0`’. Shaded cells are not used by Flash Program Memory module.

Note 1: Unimplemented, read as ‘`1`’.

TABLE 10-4: SUMMARY OF CONFIGURATION WORD WITH FLASH PROGRAM MEMORY

Name	Bits	Bit -/7	Bit -/6	Bit 13/5	Bit 12/4	Bit 11/3	Bit 10/2	Bit 9/1	Bit 8/0	Register on Page
CONFIG1	13:8	—	—	—	—	$\overline{\text{CLKOUTEN}}$	BOREN<1:0>		—	34
	7:0	$\overline{\text{CP}}$	MCLRE	$\overline{\text{PWRTE}}$	WDTE<1:0>		—	FOSC<1:0>		
CONFIG2	13:8	—	—	LVP	$\overline{\text{DEBUG}}$	$\overline{\text{LPBOR}}$	BORV	STVREN	—	35
	7:0	—	—	—	—	—	—	WRT<1:0>		

Legend: — = unimplemented location, read as ‘`0`’. Shaded cells are not used by clock sources.

11.0 I/O PORTS

In general, when a peripheral is enabled on a port pin, that pin cannot be used as a general purpose output. However, the pin can still be read.

Each port has three standard registers for its operation. These registers are:

- TRISx registers (data direction)
- PORTx registers (reads the levels on the pins of the device)
- LATx registers (output latch)

Some ports may have one or more of the following additional registers. These registers are:

- ANSELx (analog select)
- WPUx (weak pull-up)

TABLE 11-1: PORT AVAILABILITY PER DEVICE

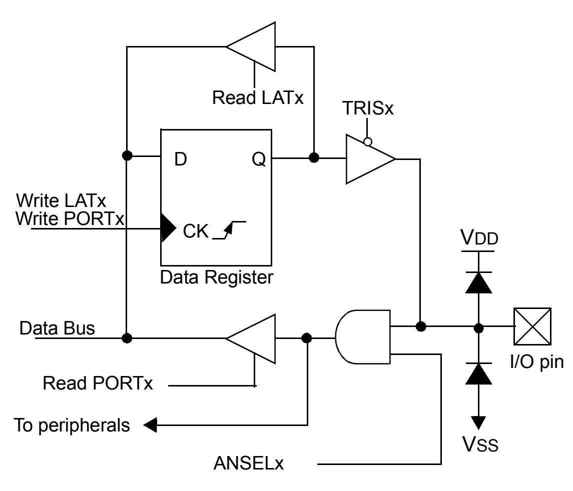
Device	PORTA	PORTB	PORTC	PORTE
PIC16LF1902/3	●	●	●	●

The Data Latch (LATA register) is useful for read-modify-write operations on the value that the I/O pins are driving.

A write operation to the LATA register has the same effect as a write to the corresponding PORTA register. A read of the LATA register reads of the values held in the I/O PORT latches, while a read of the PORTA register reads the actual I/O pin value.

Ports that support analog inputs have an associated ANSELx register. When an ANSEL bit is set, the digital input buffer associated with that bit is disabled. Disabling the input buffer prevents analog signal levels on the pin between a logic high and low from causing excessive current in the logic input circuitry. A simplified model of a generic I/O port, without the interfaces to other peripherals, is shown in Figure 11-1.

FIGURE 11-1: GENERIC I/O PORT OPERATION



EXAMPLE 11-1: INITIALIZING PORTA

```
; This code example illustrates
; initializing the PORTA register. The
; other ports are initialized in the same
; manner.

BANKSEL  PORTA        ;
CLRF     PORTA        ;Init PORTA
BANKSEL  LATA         ;Data Latch
CLRF     LATA         ;
BANKSEL  ANSELA       ;
CLRF     ANSELA       ;digital I/O
BANKSEL  TRISA        ;
MOVLW    B'00111000'  ;Set RA<5:3> as inputs
MOVWF    TRISA        ;and set RA<2:0> as
                      ;outputs
```

11.1 PORTA Registers

PORTA is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISA (Register 11-2). Setting a TRISA bit (= `1`) will make the corresponding PORTA pin an input (i.e., disable the output driver). Clearing a TRISA bit (= `0`) will make the corresponding PORTA pin an output (i.e., enables output driver and puts the contents of the output latch on the selected pin). The exception is RA3, which is input only and its TRIS bit will always read as ‘`1`’. Example 11-1 shows how to initialize PORTA.

Reading the PORTA register (Register 11-1) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch (LATA).

The TRISA register (Register 11-2) controls the PORTA pin output drivers, even when they are being used as analog inputs. The user should ensure the bits in the TRISA register are maintained set when using them as analog inputs. I/O pins configured as analog input always read ‘`0`’.

11.1.1 ANSELA REGISTER

The ANSELA register (Register 11-4) is used to configure the Input mode of an I/O pin to analog. Setting the appropriate ANSELA bit high will cause all digital reads on the pin to be read as ‘`0`’ and allow analog functions on the pin to operate correctly.

The state of the ANSELA bits has no effect on digital output functions. A pin with TRIS clear and ANSEL set will still operate as a digital output, but the Input mode will be analog. This can cause unexpected behavior when executing read-modify-write instructions on the affected port.

Note: The ANSELA bits default to the Analog mode after Reset. To use any pins as digital general purpose or peripheral inputs, the corresponding ANSEL bits must be initialized to ‘`0`’ by user software.

11.1.2 PORTA FUNCTIONS AND OUTPUT PRIORITIES

Each PORTA pin is multiplexed with other functions. The pins, their combined functions and their output priorities are shown in Table 11-2.

When multiple outputs are enabled, the actual pin control goes to the peripheral with the highest priority.

Analog input functions, such as ADC, comparator and CapSense inputs, are not shown in the priority lists. These inputs are active when the I/O pin is set for Analog mode using the ANSELx registers. Digital output functions may control the pin when it is in Analog mode with the priority shown in Table 11-2.

TABLE 11-2: PORTA OUTPUT PRIORITY

Pin Name	Function Priority[1]
RA0	SEG12 (LCD) AN0 RA0
RA1	SEG7 AN1 RA1
RA2	COM2 AN2 RA2
RA3	VREF+ COM3 SEG15 AN3 RA3
RA4	SEG4 T0CKI RA4
RA5	SEG6 AN5 RA5
RA6	CLKOUT SEG1 RA6
RA7	CLKIN SEG2 RA7

Note 1: Priority listed from highest to lowest.

REGISTER 11-1: PORTA: PORTA REGISTER

R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R-x/x	R/W-x/x	R/W-x/x	R/W-x/x
RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0
bit 7							bit 0

Legend:

R = Readable bit
W = Writable bit
U = Unimplemented bit, read as ‘0’
u = Bit is unchanged
x = Bit is unknown
-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set
‘0’ = Bit is cleared

bit 7-0 **RA<7:0>**: PORTA I/O Value bits(1)
`1` = Port pin is $\geq V_{IH}$
`0` = Port pin is $\leq V_{IL}$

Note 1: Writes to PORTA are actually written to corresponding LATA register. Reads from PORTA register is return of actual I/O pin values.

REGISTER 11-2: TRISA: PORTA TRI-STATE REGISTER

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R-1/1	R/W-1/1	R/W-1/1	R/W-1/1
TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0
bit 7							bit 0

Legend:

R = Readable bit
W = Writable bit
U = Unimplemented bit, read as ‘0’
u = Bit is unchanged
x = Bit is unknown
-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set
‘0’ = Bit is cleared

bit 7-4 **TRISA<7:4>:** PORTA Tri-State Control bits
`1` = PORTA pin configured as an input (tri-stated)
`0` = PORTA pin configured as an output

bit 3 **TRISA3:** RA3 Port Tri-State Control bit
This bit is always ‘`1`’ as RA3 is an input only

bit 2-0 **TRISA<2:0>:** PORTA Tri-State Control bits
`1` = PORTA pin configured as an input (tri-stated)
`0` = PORTA pin configured as an output

REGISTER 11-3: LATA: PORTA DATA LATCH REGISTER

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
LATA7	LATA6	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0
bit 7							bit 0

Legend:

R = Readable bit
W = Writable bit
U = Unimplemented bit, read as ‘0’
u = Bit is unchanged
x = Bit is unknown
-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set
‘0’ = Bit is cleared

bit 7-4 **LATA<7:0>**: PORTA Output Latch Value bits(1)

Note 1: Writes to PORTA are actually written to corresponding LATA register. Reads from PORTA register is return of actual I/O pin values.

REGISTER 11-4: ANSELA: PORTA ANALOG SELECT REGISTER

U-0	U-0	R/W-1/1	U-0	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
—	—	ANSA5	—	ANSA3	ANSA2	ANSA1	ANSA0
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-6 **Unimplemented:** Read as ‘`0`’

bit 5 **ANSA5**: Analog Select between Analog or Digital Function on pins RA5, respectively
`0` = Digital I/O. Pin is assigned to port or digital special function.
`1` = Analog input. Pin is assigned as analog input**(1)**. Digital input buffer disabled.

bit 4 **Unimplemented:** Read as ‘`0`’

bit 3-0 **ANSA<3:0>**: Analog Select between Analog or Digital Function on pins RA<3:0>, respectively
`0` = Digital I/O. Pin is assigned to port or digital special function.
`1` = Analog input. Pin is assigned as analog input**(1)**. Digital input buffer disabled.

Note 1: When setting a pin to an analog input, the corresponding TRIS bit must be set to Input mode in order to allow external control of the voltage on the pin.

TABLE 11-3: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
ANSELA	—	—	ANSA5	—	ANSA3	ANSA2	ANSA1	ANSA0	91
LATA	LATA7	LATA6	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	90
OPTION_REG	$\overline{\text{WPUEN}}$	INTEDG	TMR0CS	TMR0SE	PSA	PS<2:0>			121
PORTA	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0	90
TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	90

Legend: x = unknown, u = unchanged, – = unimplemented locations read as ‘`0`’. Shaded cells are not used by PORTA.

TABLE 11-4: SUMMARY OF CONFIGURATION WORD WITH PORTA

Name	Bits	Bit -/7	Bit -/6	Bit 13/5	Bit 12/4	Bit 11/3	Bit 10/2	Bit 9/1	Bit 8/0	Register on Page
CONFIG1	13:8	—	—	—	—	$\overline{\text{CLKOUTEN}}$	BOREN<1:0>		—	34
	7:0	$\overline{\text{CP}}$	MCLRE	$\overline{\text{PWRTE}}$	WDTE<1:0>		—	FOSC<1:0>		

Legend: — = unimplemented location, read as ‘`0`’. Shaded cells are not used by PORTA.

11.2 PORTB Registers

PORTB is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISB (Register 11-6). Setting a TRISB bit (= `1`) will make the corresponding PORTB pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISB bit (= `0`) will make the corresponding PORTB pin an output (i.e., enable the output driver and put the contents of the output latch on the selected pin). Example 11-1 shows how to initialize an I/O port.

Reading the PORTB register (Register 11-5) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch (LATB).

The TRISB register (Register 11-6) controls the PORTB pin output drivers, even when they are being used as analog inputs. The user should ensure the bits in the TRISB register are maintained set when using them as analog inputs. I/O pins configured as analog input always read ‘`0`’.

11.2.1 ANSELB REGISTER

The ANSELB register (Register 11-8) is used to configure the Input mode of an I/O pin to analog. Setting the appropriate ANSELB bit high will cause all digital reads on the pin to be read as ‘`0`’ and allow analog functions on the pin to operate correctly.

The state of the ANSELB bits has no effect on digital output functions. A pin with TRIS clear and ANSELB set will still operate as a digital output, but the Input mode will be analog. This can cause unexpected behavior when executing read-modify-write instructions on the affected port.

Note:	The ANSELB bits default to the Analog mode after Reset. To use any pins as digital general purpose or peripheral inputs, the corresponding ANSEL bits must be initialized to ‘`0`’ by user software.

11.2.2 PORTB FUNCTIONS AND OUTPUT PRIORITIES

Each PORTB pin is multiplexed with other functions. The pins, their combined functions and their output priorities are shown in Table 11-5.

When multiple outputs are enabled, the actual pin control goes to the peripheral with the highest priority.

Analog input and some digital input functions are not included in the list below. These input functions can remain active when the pin is configured as an output. Certain digital input functions override other port functions and are included in Table 11-5.

TABLE 11-5: PORTB OUTPUT PRIORITY

Pin Name	Function Priority[1]
RB0	SEG0 AN12 INT IOC RB0
RB1	SEG24 AN10 VLCD1 IOC RB1
RB2	SEG25 AN8 VLCD2 IOC RB2
RB3	SEG26 AN9 VLCD3 IOC RB3
RB4	COM0 AN11 IOC RB4
RB5	COM1 AN13 IOC RB5
RB6	SEG14 IOC RB6
RB7	SEG13 IOC RB7

Note 1: Priority listed from highest to lowest.

REGISTER 11-5: PORTB: PORTB REGISTER

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0 **RB<7:0>**: PORTB General Purpose I/O Pin bits(1)
1 = Port pin is ≥ V_{IH}
0 = Port pin is ≤ V_{IL}

Note 1: Writes to PORTB are actually written to corresponding LATB register. Reads from PORTB register is return of actual I/O pin values.

REGISTER 11-6: TRISB: PORTB TRI-STATE REGISTER

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0 **TRISB<7:0>:** PORTB Tri-State Control bits
1 = PORTB pin configured as an input (tri-stated)
0 = PORTB pin configured as an output

REGISTER 11-7: LATB: PORTB DATA LATCH REGISTER

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0 **LATB<7:0>**: PORTB Output Latch Value bits(1)

Note 1: Writes to PORTB are actually written to corresponding LATB register. Reads from PORTB register is return of actual I/O pin values.

REGISTER 11-8: ANSELB: PORTB ANALOG SELECT REGISTER

U-0	U-0	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
—	—	ANSB5	ANSB4	ANSB3	ANSB2	ANSB1	ANSB0
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-6 **Unimplemented:** Read as ‘`0`’

bit 5-0 **ANSB<5:0>**: Analog Select between Analog or Digital Function on pins RB<5:0>, respectively
`0` = Digital I/O. Pin is assigned to port or digital special function.
`1` = Analog input. Pin is assigned as analog input[1]. Digital input buffer disabled.

Note 1: When setting a pin to an analog input, the corresponding TRIS bit must be set to Input mode in order to allow external control of the voltage on the pin.

REGISTER 11-9: WPUB: WEAK PULL-UP PORTB REGISTER

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0 **WPUB<7:0>**: Weak Pull-up Register bits
`1` = Pull-up enabled
`0` = Pull-up disabled

Note 1: Global $\overline{\text{WPUEN}}$ bit of the OPTION_REG register must be cleared for individual pull-ups to be enabled.
2: The weak pull-up device is automatically disabled if the pin is in configured as an output.

TABLE 11-6: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
ANSELB	—	—	ANSB5	ANSB4	ANSB3	ANSB2	ANSB1	ANSB0	94
LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	93
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	93
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	93
WPUB	WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0	94

Legend: x = unknown, u = unchanged, - = unimplemented locations read as ‘`0`’. Shaded cells are not used by PORTB.

11.3 PORTC Registers

PORTC is an 8-bit wide bidirectional port. The corresponding data direction register is TRISC (Register 11-6). Setting a TRISC bit (= `1`) will make the corresponding PORTC pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISC bit (= `0`) will make the corresponding PORTC pin an output (i.e., enable the output driver and put the contents of the output latch on the selected pin). Example 11-1 shows how to initialize an I/O port.

Reading the PORTC register (Register 11-5) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch (LATC).

The TRISC register (Register 11-6) controls the PORTC pin output drivers, even when they are being used as analog inputs. The user should ensure the bits in the TRISC register are maintained set when using them as analog inputs. I/O pins configured as analog input always read ‘`0`’.

11.3.1 PORTC FUNCTIONS AND OUTPUT PRIORITIES

Each PORTC pin is multiplexed with other functions. The pins, their combined functions and their output priorities are shown in Table 11-7.

When multiple outputs are enabled, the actual pin control goes to the peripheral with the highest priority.

Analog input and some digital input functions are not included in the list below. These input functions can remain active when the pin is configured as an output. Certain digital input functions override other port functions and are included in Table 11-7.

TABLE 11-7: PORTC OUTPUT PRIORITY

Pin Name	Function Priority[1]
RC0	SOSCO (T1OSCO) T1CKI RC0
RC1	SOSC1 (T1OSCI) RC1
RC2	SEG2 RC2
RC3	SEG6 RC3
RC4	SEG11 T1G RC4
RC5	SEG10 RC5
RC6	SEG9 RC6
RC7	SEG8 RC7

Note 1: Priority listed from highest to lowest.

REGISTER 11-10: PORTC: PORTC REGISTER

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0 **RC<7:0>**: PORTC General Purpose I/O Pin bits[1]
`1` = Port pin is $\geq$ VIH
`0` = Port pin is $\leq$ VIL

Note 1: Writes to PORTC are actually written to corresponding LATC register. Reads from PORTC register is return of actual I/O pin values.

REGISTER 11-11: TRISC: PORTC TRI-STATE REGISTER

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0 **TRISC<7:0>:** PORTC Tri-State Control bits[1]
`1` = PORTC pin configured as an input (tri-stated)
`0` = PORTC pin configured as an output

REGISTER 11-12: LATC: PORTC DATA LATCH REGISTER

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0 **LATC<7:0>**: PORTC Output Latch Value bits[1]

Note 1: Writes to PORTC are actually written to corresponding LATC register. Reads from PORTC register is return of actual I/O pin values.

TABLE 11-8: SUMMARY OF REGISTERS ASSOCIATED WITH PORTC

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	93
PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	93
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	93

Legend: x = unknown, u = unchanged, - = unimplemented locations read as ‘`0`’. Shaded cells are not used by PORTC.

11.4 PORTE Registers

RE3 is input only, and also functions as $\overline{\text{MCLR}}$. The $\overline{\text{MCLR}}$ feature can be disabled via a configuration fuse. RE3 also supplies the programming voltage. The TRIS bit for RE3 (TRISE3) always reads '1'.

11.4.1 PORTE FUNCTIONS AND OUTPUT PRIORITIES

No output priorities, RE3 is an input only pin.

REGISTER 11-13: PORTE: PORTE REGISTER

U-0	U-0	U-0	U-0	R-x/u	U-0	U-0	U-0
—	—	—	—	RE3	—	—	—
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	

bit 7-4	**Unimplemented**: Read as '0'
bit 3	**RE3**: PORTE Input Pin bit 1 = Port pin is > V_{IH} 0 = Port pin is < V_{IL}
bit 2-0	**Unimplemented**: Read as '0'

REGISTER 11-14: TRISE: PORTE TRI-STATE REGISTER

U-0	U-0	U-0	U-0	U-1[1]	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	

bit 7-4	**Unimplemented**: Read as '0'
bit 3	**Unimplemented**: Read as '1'
bit 2-0	**Unimplemented**: Read as '0'

Note 1: Unimplemented, read as '1'.

REGISTER 11-15: WPUE: WEAK PULL-UP PORTE REGISTER

U-0	U-0	U-0	U-0	R/W-1/1	U-0	U-0	U-0
—	—	—	—	WPUE3	—	—	—
bit 7							bit 0

Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-4 **Unimplemented:** Read as ‘`0`’

bit 3 **WPUE:** Weak Pull-up Register bit
`1` = Pull-up enabled
`0` = Pull-up disabled

bit 2-0 **Unimplemented:** Read as ‘`0`’

Note 1: Global $\overline{\text{WPUEN}}$ bit of the OPTION_REG register must be cleared for individual pull-ups to be enabled.

2: The weak pull-up device is automatically disabled if the pin is in configured as an output.

TABLE 11-9: SUMMARY OF REGISTERS ASSOCIATED WITH PORTE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
ADCON0	—	CHS<4:0>					GO/$\overline{\text{DONE}}$	ADON	112
PORTE	—	—	—	—	RE3	—	—	—	98
TRISE	—	—	—	—	—[1]	—	—	—	98
WPUE	—	—	—	—	WPUE3	—	—	—	99

Legend: x = unknown, u = unchanged, – = unimplemented locations read as ‘`0`’. Shaded cells are not used by PORTE.

Note 1: Unimplemented, read as ‘`1`’.

12.0 INTERRUPT-ON-CHANGE

The PORTB pins can be configured to operate as Interrupt-on-Change (IOC) pins. An interrupt can be generated by detecting a signal that has either a rising edge or a falling edge. Any individual PORTB pin, or combination of PORTB pins, can be configured to generate an interrupt. The interrupt-on-change module has the following features:

- Interrupt-on-Change enable (Master Switch)
- Individual pin configuration
- Rising and falling edge detection
- Individual pin interrupt flags

Figure 12-1 is a block diagram of the IOC module.

12.1 Enabling the Module

To allow individual PORTB pins to generate an interrupt, the IOCIE bit of the INTCON register must be set. If the IOCIE bit is disabled, the edge detection on the pin will still occur, but an interrupt will not be generated.

12.2 Individual Pin Configuration

For each PORTB pin, a rising edge detector and a falling edge detector are present. To enable a pin to detect a rising edge, the associated IOCBPx bit of the IOCBP register is set. To enable a pin to detect a falling edge, the associated IOCBNx bit of the IOCBN register is set.

A pin can be configured to detect rising and falling edges simultaneously by setting both the IOCBPx bit and the IOCBNx bit of the IOCBP and IOCBN registers, respectively.

12.3 Interrupt Flags

The IOCBFx bits located in the IOCBF register are status flags that correspond to the interrupt-on-change pins of PORTB. If an expected edge is detected on an appropriately enabled pin, then the status flag for that pin will be set, and an interrupt will be generated if the IOCIE bit is set. The IOCIF bit of the INTCON register reflects the status of all IOCBFx bits.

12.4 Clearing Interrupt Flags

The individual status flags, (IOCBFx bits), can be cleared by resetting them to zero. If another edge is detected during this clearing operation, the associated status flag will be set at the end of the sequence, regardless of the value actually being written.

In order to ensure that no detected edge is lost while clearing flags, only AND operations masking out known changed bits should be performed. The following sequence is an example of what should be performed.

EXAMPLE 12-1:

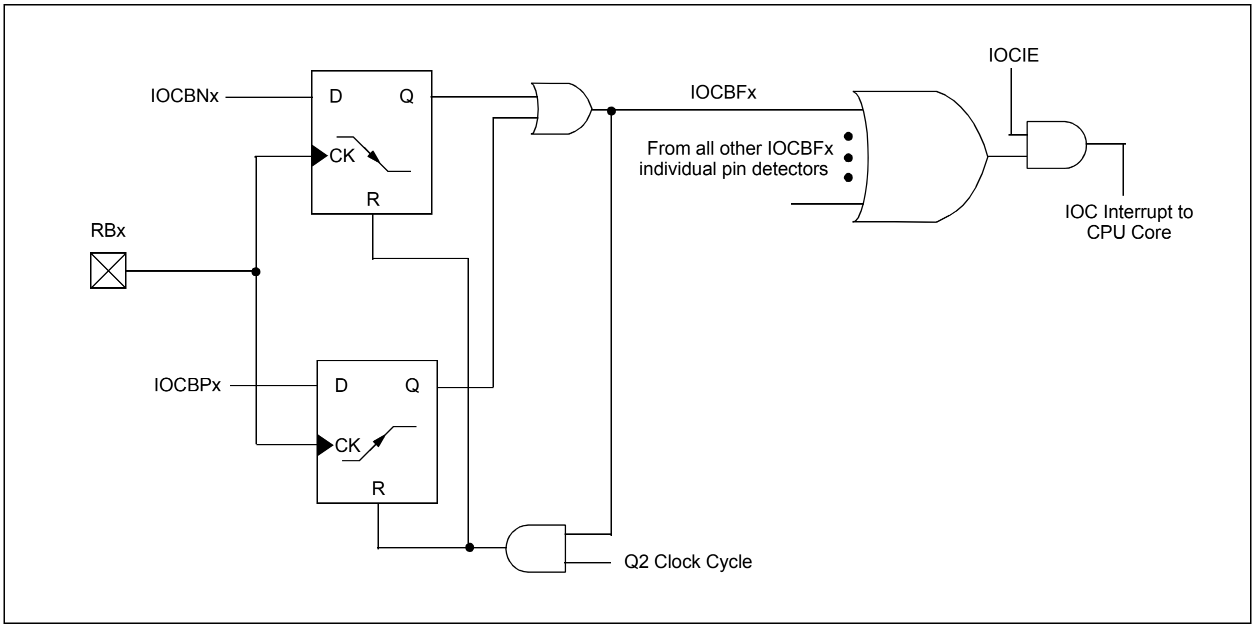
```
MOVLW  0xff
XORWF  IOCBF, W
ANDWF  IOCBF, F
```

12.5 Operation in Sleep

The interrupt-on-change interrupt sequence will wake the device from Sleep mode, if the IOCIE bit is set.

If an edge is detected while in Sleep mode, the IOCBF register will be updated prior to the first instruction executed out of Sleep.

FIGURE 12-1: INTERRUPT-ON-CHANGE BLOCK DIAGRAM


IOCIE
IOCBNx
D
Q
IOCBFx
CK
From all other IOCBFx individual pin detectors
R
IOC Interrupt to CPU Core
RBx
IOCBPx
D
Q
CK
R
Q2 Clock Cycle

REGISTER 12-1: IOCBP: INTERRUPT-ON-CHANGE POSITIVE EDGE REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
IOCBP7	IOCBP6	IOCBP5	IOCBP4	IOCBP3	IOCBP2	IOCBP1	IOCBP0
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0 **IOCBP<7:0>:** Interrupt-on-Change Positive Edge Enable bits

`1` = Interrupt-on-Change enabled on the pin for a positive going edge. Associated Status bit and interrupt flag will be set upon detecting an edge.

`0` = Interrupt-on-Change disabled for the associated pin.

REGISTER 12-2: IOCBN: INTERRUPT-ON-CHANGE NEGATIVE EDGE REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
IOCBN7	IOCBN6	IOCBN5	IOCBN4	IOCBN3	IOCBN2	IOCBN1	IOCBN0
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0 **IOCBN<7:0>:** Interrupt-on-Change Negative Edge Enable bits

`1` = Interrupt-on-Change enabled on the pin for a negative going edge. Associated Status bit and interrupt flag will be set upon detecting an edge.

`0` = Interrupt-on-Change disabled for the associated pin.

REGISTER 12-3: IOCBF: INTERRUPT-ON-CHANGE FLAG REGISTER

R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0
IOCBF7	IOCBF6	IOCBF5	IOCBF4	IOCBF3	IOCBF2	IOCBF1	IOCBF0
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	HS - Bit is set in hardware

bit 7-0 **IOCBF<7:0>:** Interrupt-on-Change Flag bits

`1` = An enabled change was detected on the associated pin.
Set when IOCBPx = `1` and a rising edge was detected on RBx, or when IOCBNx = `1` and a falling edge was detected on RBx.

`0` = No change was detected, or the user cleared the detected change.

TABLE 12-1: SUMMARY OF REGISTERS ASSOCIATED WITH INTERRUPT-ON-CHANGE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
ANSELB	—	—	ANSB5	ANSB4	ANSB3	ANSB2	ANSB1	ANSB0	94
INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	60
IOCBF	IOCBF7	IOCBF6	IOCBF5	IOCBF4	IOCBF3	IOCBF2	IOCBF1	IOCBF0	101
IOCBN	IOCBN7	IOCBN6	IOCBN5	IOCBN4	IOCBN3	IOCBN2	IOCBN1	IOCBN0	101
IOCBP	IOCBP7	IOCBP6	IOCBP5	IOCBP4	IOCBP3	IOCBP2	IOCBP1	IOCBP0	101
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	93

Legend: — = unimplemented location, read as ‘`0`’. Shaded cells are not used by interrupt-on-change.

13.0 FIXED VOLTAGE REFERENCE (FVR)

The Fixed Voltage Reference, or FVR, is a stable voltage reference, independent of VDD, with 1.024V or 2.048V selectable output levels. The output of the FVR can be configured as the FVR input channel on the ADC.

The FVR can be enabled by setting the FVREN bit of the FVRCON register.

13.1 Independent Gain Amplifiers

The output of the FVR supplied to the ADC is routed through two independent programmable gain amplifiers. Each amplifier can be configured to amplify the reference voltage by 1x or 2x, to produce the two possible voltage levels.

The ADFVR<1:0> bits of the FVRCON register are used to enable and configure the gain amplifier settings for the reference supplied to the ADC module. Reference **Section 15.0 “Analog-to-Digital Converter (ADC) Module”** for additional information.

13.2 FVR Stabilization Period

When the Fixed Voltage Reference module is enabled, it requires time for the reference and amplifier circuits to stabilize. Once the circuits stabilize and are ready for use, the FVRRDY bit of the FVRCON register will be set. See **Section 21.0 “Electrical Specifications”** for the minimum delay requirement.

FIGURE 13-1: VOLTAGE REFERENCE BLOCK DIAGRAM

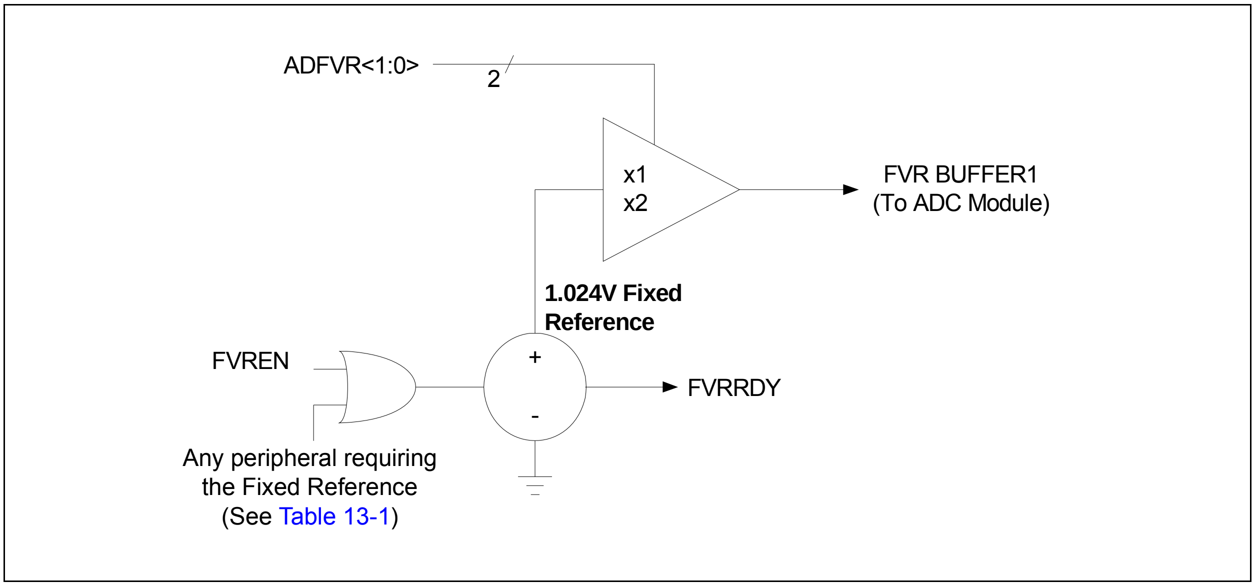


TABLE 13-1: PERIPHERALS REQUIRING THE FIXED VOLTAGE REFERENCE (FVR)

Peripheral	Conditions	Description
HFINTOSC	FOSC<2:0> = `100` and IRCF<3:0> = `000x`	INTOSC is active and device is not in Sleep.
BOR	BOREN<1:0> = `11`	BOR always enabled.
	BOREN<1:0> = `10` and BORFS = `1`	BOR disabled in Sleep mode, BOR Fast Start enabled.
	BOREN<1:0> = `01` and BORFS = `1`	BOR under software control, BOR Fast Start enabled.

13.3 FVR Control Registers

REGISTER 13-1: FVRCON: FIXED VOLTAGE REFERENCE CONTROL REGISTER

R/W-0/0	R-q/q	R/W-0/0	R/W-0/0	U-0	U-0	R/W-0/0	R/W-0/0
FVREN	FVRRDY[1]	TSEN	TSRNG	—	—	ADFVR<1:0>	
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	q = Value depends on condition

bit 7 **FVREN:** Fixed Voltage Reference Enable bit
`0` = Fixed Voltage Reference is disabled
`1` = Fixed Voltage Reference is enabled

bit 6 **FVRRDY:** Fixed Voltage Reference Ready Flag bit[1]
`0` = Fixed Voltage Reference output is not ready or not enabled
`1` = Fixed Voltage Reference output is ready for use

bit 5 **TSEN:** Temperature Indicator Enable bit
`0` = Temperature Indicator is disabled
`1` = Temperature Indicator is enabled

bit 4 **TSRNG:** Temperature Indicator Range Selection bit
`0` = $V_{OUT} = V_{DD} - 2V_T$ (Low Range)
`1` = $V_{OUT} = V_{DD} - 4V_T$ (High Range)

bit 3-2 **Unimplemented:** Read as ‘`0`’

bit 1-0 **ADFVR<1:0>:** ADC Fixed Voltage Reference Selection bit
`00` = ADC Fixed Voltage Reference Peripheral output is off.
`01` = ADC Fixed Voltage Reference Peripheral output is 1x (1.024V)
`10` = ADC Fixed Voltage Reference Peripheral output is 2x (2.048V)[2]
`11` = Reserved

Note 1: FVRRDY will output the true state of the band gap.
2: Fixed Voltage Reference output cannot exceed V_{DD}.

TABLE 13-2: SUMMARY OF REGISTERS ASSOCIATED WITH FIXED VOLTAGE REFERENCE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
FVRCON	FVREN	FVRRDY	TSEN	TSRNG	—	—	ADFVR1	ADFVR0	104

Legend: Shaded cells are not used with the Fixed Voltage Reference.

14.0 TEMPERATURE INDICATOR MODULE

This family of devices is equipped with a temperature circuit designed to measure the operating temperature of the silicon die. The circuit's range of operating temperature falls between of -40°C and +85°C. The output is a voltage that is proportional to the device temperature. The output of the temperature indicator is internally connected to the device ADC.

The circuit may be used as a temperature threshold detector or a more accurate temperature indicator, depending on the level of calibration performed. A one-point calibration allows the circuit to indicate a temperature closely surrounding that point. A two-point calibration allows the circuit to sense the entire range of temperature more accurately. Reference Application Note AN1333, "*Use and Calibration of the Internal Temperature Indicator*" (DS01333) for more details regarding the calibration process.

14.1 Circuit Operation

Figure 14-1 shows a simplified block diagram of the temperature circuit. The proportional voltage output is achieved by measuring the forward voltage drop across multiple silicon junctions.

Equation 14-1 describes the output characteristics of the temperature indicator.

EQUATION 14-1: VOUT RANGES

High Range: $V_{OUT} = V_{DD} - 4V_T$

Low Range: $V_{OUT} = V_{DD} - 2V_T$

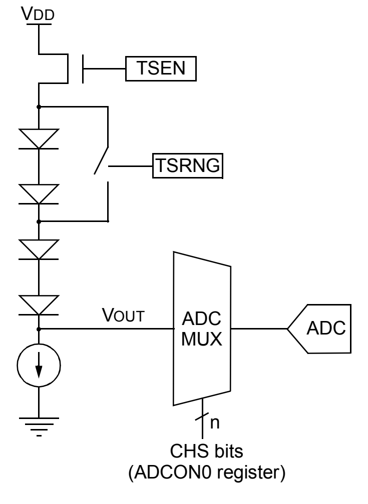
The temperature sense circuit is integrated with the Fixed Voltage Reference (FVR) module. See **Section 13.0 "Fixed Voltage Reference (FVR)"** for more information.

The circuit is enabled by setting the TSEN bit of the FVRCON register. When disabled, the circuit draws no current.

The circuit operates in either high or low range. The high range, selected by setting the TSRNG bit of the FVRCON register, provides a wider output voltage. This provides more resolution over the temperature range, but may be less consistent from part to part. This range requires a higher bias voltage to operate and thus, a higher VDD is needed.

The low range is selected by clearing the TSRNG bit of the FVRCON register. The low range generates a lower voltage drop and thus, a lower bias voltage is needed to operate the circuit. The low range is provided for low voltage operation.

FIGURE 14-1: TEMPERATURE CIRCUIT DIAGRAM



14.2 Minimum Operating VDD vs. Minimum Sensing Temperature

When the temperature circuit is operated in low range, the device may be operated at any operating voltage that is within specifications.

When the temperature circuit is operated in high range, the device operating voltage, VDD, must be high enough to ensure that the temperature circuit is correctly biased.

Table 14-1 shows the recommended minimum VDD vs. range setting.

TABLE 14-1: RECOMMENDED VDD VS. RANGE

Min. VDD, TSRNG = 1	Min. VDD, TSRNG = 0
3.6V	1.8V

14.3 Temperature Output

The output of the circuit is measured using the internal Analog-to-Digital converter. A channel is reserved for the temperature circuit output. Refer to **Section 15.0 "Analog-to-Digital Converter (ADC) Module"** for detailed information.

14.4 ADC Acquisition Time

To ensure accurate temperature measurements, the user must wait at least 200 μs after the ADC input multiplexer is connected to the temperature indicator output before the conversion is performed. In addition, the user must wait 200 μs between sequential conversions of the temperature indicator output.

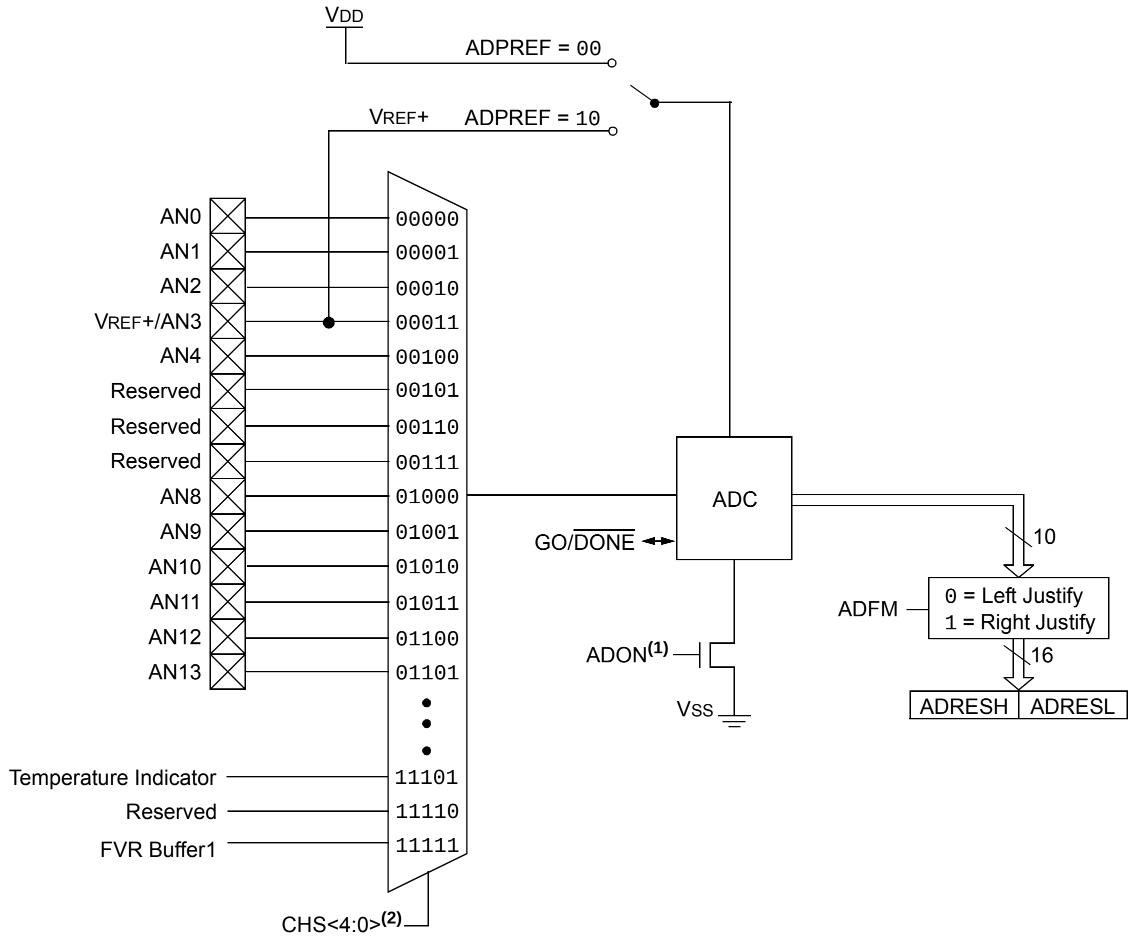
15.0 ANALOG-TO-DIGITAL CONVERTER (ADC) MODULE

The Analog-to-Digital Converter (ADC) allows conversion of an analog input signal to a 10-bit binary representation of that signal. This device uses analog inputs, which are multiplexed into a single sample and hold circuit. The output of the sample and hold is connected to the input of the converter. The converter generates a 10-bit binary result via successive approximation and stores the conversion result into the ADC result registers (ADRESH:ADRESL register pair). Figure 15-1 shows the block diagram of the ADC.

The ADC voltage reference is software selectable to be either internally generated or externally supplied.

The ADC can generate an interrupt upon completion of a conversion. This interrupt can be used to wake-up the device from Sleep.

FIGURE 15-1: ADC BLOCK DIAGRAM


VDD
ADPREF = 00
VREF+
ADPREF = 10
AN0
AN1
AN2
VREF+/AN3
AN4
Reserved
Reserved
Reserved
AN8
AN9
AN10
AN11
AN12
AN13
Temperature Indicator
Reserved
FVR Buffer1
00000
00001
00010
00011
00100
00101
00110
00111
01000
01001
01010
01011
01100
01101
11101
11110
11111
CHS<4:0>(2)
ADC
GO/DONE
ADON(1)
VSS
10
ADFM
0 = Left Justify
1 = Right Justify
16
ADRESH
ADRESL

Note 1: When ADON = 0, all multiplexer inputs are disconnected.

2: See ADCON0 register (Example 15-1) for detailed analog channel selection per device.

15.1 ADC Configuration

When configuring and using the ADC the following functions must be considered:

- Port configuration
- Channel selection
- ADC voltage reference selection
- ADC conversion clock source
- Interrupt control
- Result formatting

15.1.1 PORT CONFIGURATION

The ADC can be used to convert both analog and digital signals. When converting analog signals, the I/O pin should be configured for analog by setting the associated TRIS and ANSEL bits. Refer to **Section 11.0 "I/O Ports"** for more information.

Note: Analog voltages on any pin that is defined as a digital input may cause the input buffer to conduct excess current.

15.1.2 CHANNEL SELECTION

There are up to 11 channel selections available:

- AN<13:0> pins
- Temperature Indicator
- FVR (Fixed Voltage Reference) Output

Refer to **Section 13.0 "Fixed Voltage Reference (FVR)"** and **Section 14.0 "Temperature Indicator Module"** for more information on these channel selections.

The CHS bits of the ADCON0 register determine which channel is connected to the sample and hold circuit.

When changing channels, a delay is required before starting the next conversion. Refer to **Section 15.2 "ADC Operation"** for more information.

15.1.3 ADC VOLTAGE REFERENCE

The ADPREF bits of the ADCON1 register provides control of the positive voltage reference. The positive voltage reference can be:

- VREF+ pin
- VDD

15.1.4 CONVERSION CLOCK

The source of the conversion clock is software selectable via the ADCS bits of the ADCON1 register. There are seven possible clock options:

- FOSC/2
- FOSC/4
- FOSC/8
- FOSC/16
- FOSC/32
- FOSC/64
- FRC (dedicated internal oscillator)

The time to complete one bit conversion is defined as TAD. One full 10-bit conversion requires 11.5 TAD periods as shown in Figure 15-2.

For correct conversion, the appropriate TAD specification must be met. Refer to the A/D conversion requirements in **Section 21.0 "Electrical Specifications"** for more information. Table 15-1 gives examples of appropriate ADC clock selections.

Note: Unless using the FRC, any changes in the system clock frequency will change the ADC clock frequency, which may adversely affect the ADC result.

TABLE 15-1: ADC CLOCK PERIOD (TAD) VS. DEVICE OPERATING FREQUENCIES

ADC Clock Period (TAD)		Device Frequency (FOSC)				
ADC Clock Source	**ADCS<2:0>**	**20 MHz**	**16 MHz**	**8 MHz**	**4 MHz**	**1 MHz**
FOSC/2	000	100 ns(2)	125 ns(2)	250 ns(2)	500 ns(2)	2.0 μs
FOSC/4	100	200 ns(2)	250 ns(2)	500 ns(2)	1.0 μs	4.0 μs
FOSC/8	001	400 ns(2)	0.5 μs(2)	1.0 μs	2.0 μs	8.0 μs(3)
FOSC/16	101	800 ns	1.0 μs	2.0 μs	4.0 μs	16.0 μs(3)
FOSC/32	010	1.6 μs	2.0 μs	4.0 μs	8.0 μs(3)	32.0 μs(3)
FOSC/64	110	3.2 μs	4.0 μs	8.0 μs(3)	16.0 μs(3)	64.0 μs(3)
FRC	x11	1.0-6.0 μs(1,4)	1.0-6.0 μs(1,4)	1.0-6.0 μs(1,4)	1.0-6.0 μs(1,4)	1.0-6.0 μs(1,4)

Legend: Shaded cells are outside of recommended range.

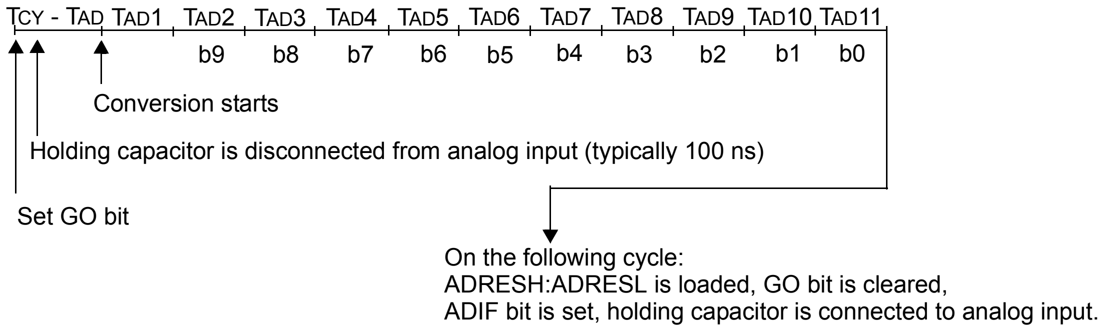
Note 1: The FRC source has a typical TAD time of 1.6 μs for VDD.

2: These values violate the minimum required TAD time.

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

4: The ADC clock period (TAD) and total ADC conversion time can be minimized when the ADC clock is derived from the system clock FOSC. However, the FRC clock source must be used when conversions are to be performed with the device in Sleep mode.

FIGURE 15-2: ANALOG-TO-DIGITAL CONVERSION TAD CYCLES


TCY - TAD
TAD1
TAD2
TAD3
TAD4
TAD5
TAD6
TAD7
TAD8
TAD9
TAD10
TAD11
b9
b8
b7
b6
b5
b4
b3
b2
b1
b0
Conversion starts
Holding capacitor is disconnected from analog input (typically 100 ns)
Set GO bit
On the following cycle:
ADRESH:ADRESL is loaded, GO bit is cleared,
ADIF bit is set, holding capacitor is connected to analog input.

15.1.5 INTERRUPTS

The ADC module allows for the ability to generate an interrupt upon completion of an Analog-to-Digital conversion. The ADC Interrupt Flag is the ADIF bit in the PIR1 register. The ADC Interrupt Enable is the ADIE bit in the PIE1 register. The ADIF bit must be cleared in software.

> **Note 1:** The ADIF bit is set at the completion of every conversion, regardless of whether or not the ADC interrupt is enabled.
>
> **2:** The ADC operates during Sleep only when the FRC oscillator is selected.

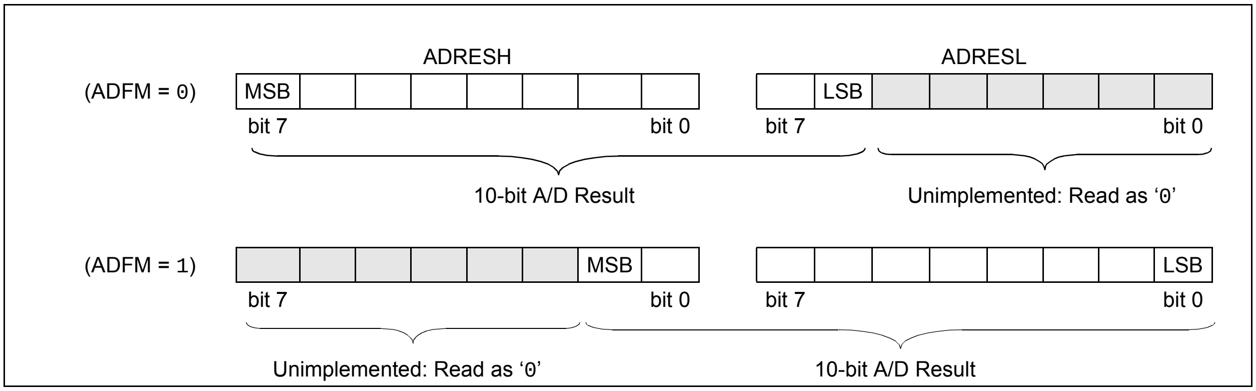
This interrupt can be generated while the device is operating or while in Sleep. If the device is in Sleep, the interrupt will wake-up the device. Upon waking from Sleep, the next instruction following the `SLEEP` instruction is always executed. If the user is attempting to wake-up from Sleep and resume in-line code execution, the GIE and PEIE bits of the INTCON register must be disabled. If the GIE and PEIE bits of the INTCON register are enabled, execution will switch to the Interrupt Service Routine.

15.1.6 RESULT FORMATTING

The 10-bit A/D conversion result can be supplied in two formats, left justified or right justified. The ADFM bit of the ADCON1 register controls the output format.

Figure 15-3 shows the two output formats.

FIGURE 15-3: 10-BIT A/D CONVERSION RESULT FORMAT


ADRESH
ADRESL
(ADFM = 0)
MSB
LSB
bit 7
bit 0
bit 7
bit 0
10-bit A/D Result
Unimplemented: Read as '0'
(ADFM = 1)
MSB
LSB
bit 7
bit 0
bit 7
bit 0
Unimplemented: Read as '0'
10-bit A/D Result

15.2 ADC Operation

15.2.1 STARTING A CONVERSION

To enable the ADC module, the ADON bit of the ADCON0 register must be set to a ‘1’. Setting the GO/$\overline{\text{DONE}}$ bit of the ADCON0 register to a ‘1’ will start the Analog-to-Digital conversion.

> **Note:** The GO/$\overline{\text{DONE}}$ bit should not be set in the same instruction that turns on the ADC. Refer to **Section 15.2.5 “A/D Conversion Procedure”**.

15.2.2 COMPLETION OF A CONVERSION

When the conversion is complete, the ADC module will:

- Clear the GO/$\overline{\text{DONE}}$ bit
- Set the ADIF Interrupt Flag bit
- Update the ADRESH and ADRESL registers with new conversion result

15.2.3 TERMINATING A CONVERSION

If a conversion must be terminated before completion, the GO/$\overline{\text{DONE}}$ bit can be cleared in software. The ADRESH and ADRESL registers will be updated with the partially complete Analog-to-Digital conversion sample. Incomplete bits will match the last bit converted.

> **Note:** A device Reset forces all registers to their Reset state. Thus, the ADC module is turned off and any pending conversion is terminated.

15.2.4 ADC OPERATION DURING SLEEP

The ADC module can operate during Sleep. This requires the ADC clock source to be set to the FRC option. When the FRC clock source is selected, the ADC waits one additional instruction before starting the conversion. This allows the `SLEEP` instruction to be executed, which can reduce system noise during the conversion. If the ADC interrupt is enabled, the device will wake-up from Sleep when the conversion completes. If the ADC interrupt is disabled, the ADC module is turned off after the conversion completes, although the ADON bit remains set.

When the ADC clock source is something other than FRC, a `SLEEP` instruction causes the present conversion to be aborted and the ADC module is turned off, although the ADON bit remains set.

15.2.5 A/D CONVERSION PROCEDURE

This is an example procedure for using the ADC to perform an Analog-to-Digital conversion:

1. Configure Port:
    - Disable pin output driver (Refer to the TRIS register)
    - Configure pin as analog (Refer to the ANSEL register)
2. Configure the ADC module:
    - Select ADC conversion clock
    - Configure voltage reference
    - Select ADC input channel
    - Turn on ADC module
3. Configure ADC interrupt (optional):
    - Clear ADC interrupt flag
    - Enable ADC interrupt
    - Enable peripheral interrupt
    - Enable global interrupt(1)
4. Wait the required acquisition time(2).
5. Start conversion by setting the GO/$\overline{\text{DONE}}$ bit.
6. Wait for ADC conversion to complete by one of the following:
    - Polling the GO/$\overline{\text{DONE}}$ bit
    - Waiting for the ADC interrupt (interrupts enabled)
7. Read ADC Result.
8. Clear the ADC interrupt flag (required if interrupt is enabled).

Note 1: The global interrupt can be disabled if the user is attempting to wake-up from Sleep and resume in-line code execution.

2: Refer to **Section 15.3 “A/D Acquisition Requirements”**.

EXAMPLE 15-1: A/D CONVERSION

```
;This code block configures the ADC
;for polling, Vdd and Vss references, Frc
;clock and AN0 input.
;
;Conversion start & polling for completion
; are included.
;
BANKSEL   ADCON1      ;
MOVLW     B’11110000’ ;Right justify, Frc
                      ;clock
MOVWF     ADCON1      ;Vdd and Vss Vref
BANKSEL   TRISA       ;
BSF       TRISA,0     ;Set RA0 to input
BANKSEL   ANSEL       ;
BSF       ANSEL,0     ;Set RA0 to analog
BANKSEL   ADCON0      ;
MOVLW     B’00000001’ ;Select channel AN0
MOVWF     ADCON0      ;Turn ADC On
CALL      SampleTime  ;Acquisiton delay
BSF       ADCON0,ADGO ;Start conversion
BTFSC     ADCON0,ADGO ;Is conversion done?
GOTO      $-1         ;No, test again
BANKSEL   ADRESH      ;
MOVF      ADRESH,W    ;Read upper 2 bits
MOVWF     RESULTHI    ;store in GPR space
BANKSEL   ADRESL      ;
MOVF      ADRESL,W    ;Read lower 8 bits
MOVWF     RESULTLO    ;Store in GPR space
```

15.2.6 ADC REGISTER DEFINITIONS

The following registers are used to control the operation of the ADC.

REGISTER 15-1: ADCON0: A/D CONTROL REGISTER 0

<table>
<tr><td>U-0</td><td>R/W-0/0</td><td>R/W-0/0</td><td>R/W-0/0</td><td>R/W-0/0</td><td>R/W-0/0</td><td>R/W-0/0</td><td>R/W-0/0</td></tr>
<tr><td>—</td><td colspan="5">CHS<4:0></td><td>GO/$\overline{\text{DONE}}$</td><td>ADON</td></tr>
<tr><td colspan="4">bit 7</td><td colspan="4">bit 0</td></tr>
</table>

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7 **Unimplemented:** Read as ‘`0`’

bit 6-2 **CHS<4:0>:** Analog Channel Select bits
`00000` = AN0
`00001` = AN1
`00010` = AN2
`00011` = AN3
`00100` = AN4
`00101` = Reserved. No channel connected.
`00110` = Reserved. No channel connected.
`00111` = Reserved. No channel connected.
`01000` = AN8
`01001` = AN9
`01010` = AN10
`01011` = AN11
`01100` = AN12
`01101` = AN13
`01110` = Reserved. No channel connected.
•
•
•
`11100` = Reserved. No channel connected.
`11101` = Temperature Indicator(2)
`11110` = Reserved. No channel connected.
`11111` = FVR (Fixed Voltage Reference) Buffer 1 Output(1)

bit 1 **GO/$\overline{\text{DONE}}$:** A/D Conversion Status bit
`1` = A/D conversion cycle in progress. Setting this bit starts an A/D conversion cycle.
This bit is automatically cleared by hardware when the A/D conversion has completed.
`0` = A/D conversion completed/not in progress

bit 0 **ADON:** ADC Enable bit
`1` = ADC is enabled
`0` = ADC is disabled and consumes no operating current

Note 1: See **Section 13.0 “Fixed Voltage Reference (FVR)”** for more information.
2: See **Section 14.0 “Temperature Indicator Module”** for more information.

REGISTER 15-2: ADCON1: A/D CONTROL REGISTER 1

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	U-0	R/W-0/0	R/W-0/0
ADFM	ADCS<2:0>			—	—	ADPREF<1:0>	
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7 **ADFM:** A/D Result Format Select bit

`1` = Right justified. Six Most Significant bits of ADRESH are set to ‘`0`’ when the conversion result is loaded.

`0` = Left justified. Six Least Significant bits of ADRESL are set to ‘`0`’ when the conversion result is loaded.

bit 6-4 **ADCS<2:0>:** A/D Conversion Clock Select bits

`000` = FOSC/2
`001` = FOSC/8
`010` = FOSC/32
`011` = FRC (clock supplied from a dedicated RC oscillator)
`100` = FOSC/4
`101` = FOSC/16
`110` = FOSC/64
`111` = FRC (clock supplied from a dedicated RC oscillator)

bit 3-2 **Unimplemented:** Read as ‘`0`’

bit 1-0 **ADPREF<1:0>:** A/D Positive Voltage Reference Configuration bits

`00` = VREF+ is connected to VDD
`01` = Reserved
`10` = VREF+ is connected to external VREF+ pin(1)
`11` = Reserved

Note 1: When selecting the FVR or the VREF+ pin as the source of the positive reference, be aware that a minimum voltage specification exists. See **Section 21.0 “Electrical Specifications”** for details.

REGISTER 15-3: ADRESH: ADC RESULT REGISTER HIGH (ADRESH) ADFM = `0`

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
ADRES<9:2>							
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0	**ADRES<9:2>**: ADC Result Register bits Upper eight bits of 10-bit conversion result

REGISTER 15-4: ADRESL: ADC RESULT REGISTER LOW (ADRESL) ADFM = `0`

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
ADRES<1:0>		—	—	—	—	—	—
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-6	**ADRES<1:0>**: ADC Result Register bits Lower two bits of 10-bit conversion result
bit 5-0	**Reserved**: Do not use.

REGISTER 15-5: ADRESH: ADC RESULT REGISTER HIGH (ADRESH) ADFM = 1

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
—	—	—	—	—	—	ADRES<9:8>	
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-2	**Reserved**: Do not use.
bit 1-0	**ADRES<9:8>**: ADC Result Register bits Upper two bits of 10-bit conversion result

REGISTER 15-6: ADRESL: ADC RESULT REGISTER LOW (ADRESL) ADFM = 1

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
ADRES<7:0>							
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0	**ADRES<7:0>**: ADC Result Register bits Lower eight bits of 10-bit conversion result

15.3 A/D Acquisition Requirements

For the ADC to meet its specified accuracy, the charge holding capacitor (CHOLD) must be allowed to fully charge to the input channel voltage level. The Analog Input model is shown in Figure 15-4. The source impedance (RS) and the internal sampling switch (RSS) impedance directly affect the time required to charge the capacitor CHOLD. The sampling switch (RSS) impedance varies over the device voltage (VDD), refer to Figure 15-4. **The maximum recommended impedance for analog sources is 10 kΩ.** As the source impedance is decreased, the acquisition time may be decreased. After the analog input channel is selected (or changed), an A/D acquisition must be done before the conversion can be started. To calculate the minimum acquisition time, Equation 15-1 may be used. This equation assumes that 1/2 LSb error is used (1,024 steps for the ADC). The 1/2 LSb error is the maximum error allowed for the ADC to meet its specified resolution.

EQUATION 15-1: ACQUISITION TIME EXAMPLE

Assumptions: $Temperature = 50°C$ *and external impedance of* $10k\Omega$ $5.0V$ V_{DD}

$$\begin{aligned} T_{ACQ} &= \text{Amplifier Settling Time} + \text{Hold Capacitor Charging Time} + \text{Temperature Coefficient} \\ &= T_{AMP} + T_C + T_{COFF} \\ &= 2\mu s + T_C + [(Temperature - 25°C)(0.05\mu s/°C)] \end{aligned}$$

The value for T_C *can be approximated with the following equations:*

$$V_{APPLIED}\left(1 - \frac{1}{(2^{n+1}) - 1}\right) = V_{CHOLD} \qquad ;[1]\ V_{CHOLD} \text{ charged to within 1/2 lsb}$$

$$V_{APPLIED}\left(1 - e^{\frac{-T_C}{RC}}\right) = V_{CHOLD} \qquad ;[2]\ V_{CHOLD} \text{ charge response to } V_{APPLIED}$$

$$V_{APPLIED}\left(1 - e^{\frac{-T_C}{RC}}\right) = V_{APPLIED}\left(1 - \frac{1}{(2^{n+1}) - 1}\right) \qquad ;\text{combining [1] and [2]}$$

Note: Where n = number of bits of the ADC.

Solving for T_C:

$$\begin{aligned} T_C &= -C_{HOLD}(R_{IC} + R_{SS} + R_S)\ \ln(1/2047) \\ &= -12.5pF(1k\Omega + 7k\Omega + 10k\Omega)\ \ln(0.0004885) \\ &= 1.715\mu s \end{aligned}$$

Therefore:

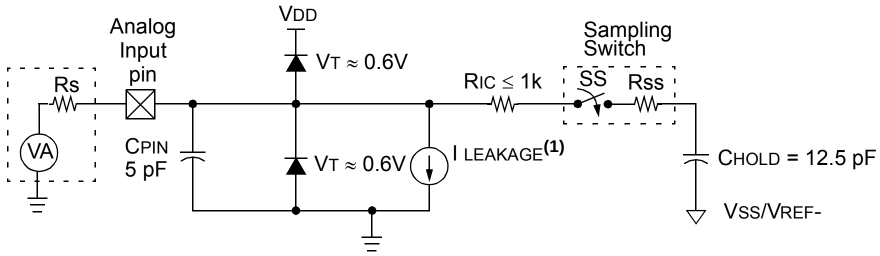
$$\begin{aligned} T_{ACQ} &= 2\mu s + 1.715\mu s + [(50°C - 25°C)(0.05\mu s/°C)] \\ &= 4.96\mu s \end{aligned}$$

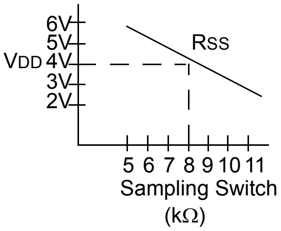
Note 1: The reference voltage (VREF) has no effect on the equation, since it cancels itself out.

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

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

FIGURE 15-4: ANALOG INPUT MODEL


Analog Input pin
VDD
VT ≈ 0.6V
Sampling Switch
Rs
RIC ≤ 1k
SS
Rss
VA
CPIN 5 pF
VT ≈ 0.6V
I LEAKAGE(1)
CHOLD = 12.5 pF
VSS/VREF-



Legend:
CHOLD = Sample/Hold Capacitance
CPIN = Input Capacitance
I LEAKAGE = Leakage current at the pin due to various junctions
RIC = Interconnect Resistance
RSS = Resistance of Sampling Switch
SS = Sampling Switch
VT = Threshold Voltage

Note 1: Refer to **Section 21.0 "Electrical Specifications"**.

FIGURE 15-5: ADC TRANSFER FUNCTION

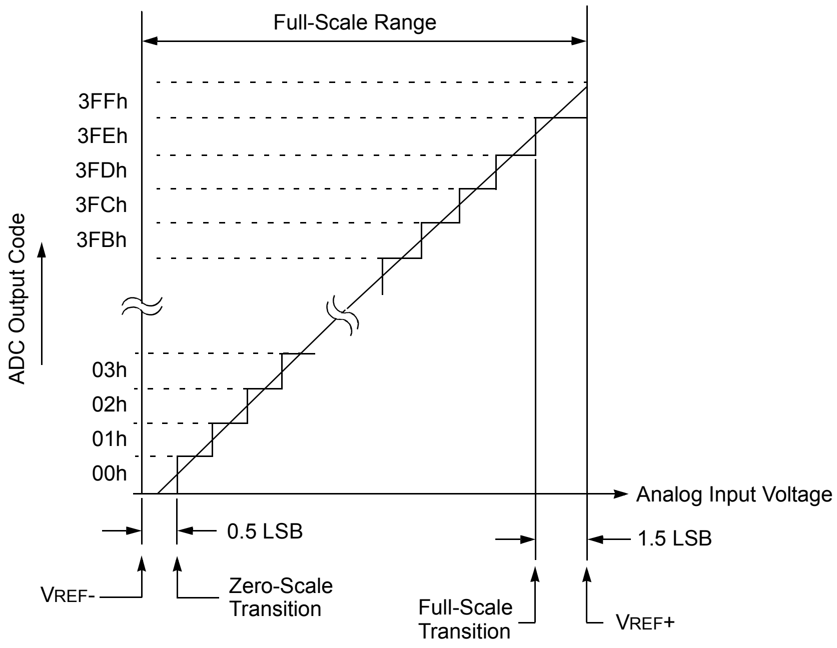

Full-Scale Range
3FFh
3FEh
3FDh
3FCh
3FBh
ADC Output Code
03h
02h
01h
00h
Analog Input Voltage
0.5 LSB
1.5 LSB
VREF-
Zero-Scale Transition
Full-Scale Transition
VREF+

TABLE 15-2: SUMMARY OF REGISTERS ASSOCIATED WITH ADC

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
ADCON0	—	CHS4	CHS3	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	ADON	112
ADCON1	ADFM	ADCS2	ADCS1	ADCS0	—	—	ADPREF1	ADPREF0	113
ADRESH	A/D Result Register High								114, 115
ADRESL	A/D Result Register Low								114, 115
ANSELA	—	—	ANSA5	—	ANSA3	ANSA2	ANSA1	ANSA0	91
ANSELB	—	—	ANSB5	ANSB4	ANSB3	ANSB2	ANSB1	ANSB0	94
INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	60
PIE1	TMR1GIE	ADIE	—	—	—	—	—	TMR1IE	61
PIR1	TMR1GIF	ADIF	—	—	—	—	—	TMR1IF	63
TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	90
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	93
FVRCON	FVREN	FVRRDY	TSEN	TSRNG	—	—	ADFVR1	ADFVR0	104

Legend: x = unknown, u = unchanged, – = unimplemented read as ‘`0`’, q = value depends on condition. Shaded cells are not used for ADC module.

16.0 TIMER0 MODULE

The Timer0 module is an 8-bit timer/counter with the following features:

- 8-bit timer/counter register (TMR0)
- 8-bit prescaler (independent of Watchdog Timer)
- Programmable internal or external clock source
- Programmable external clock edge selection
- Interrupt on overflow
- TMR0 can be used to gate Timer1

Figure 16-1 is a block diagram of the Timer0 module.

16.1 Timer0 Operation

The Timer0 module can be used as either an 8-bit timer or an 8-bit counter.

16.1.1 8-BIT TIMER MODE

The Timer0 module will increment every instruction cycle, if used without a prescaler. 8-Bit Timer mode is selected by clearing the TMR0CS bit of the OPTION_REG register.

When TMR0 is written, the increment is inhibited for two instruction cycles immediately following the write.

Note: The value written to the TMR0 register can be adjusted, in order to account for the two instruction cycle delay when TMR0 is written.

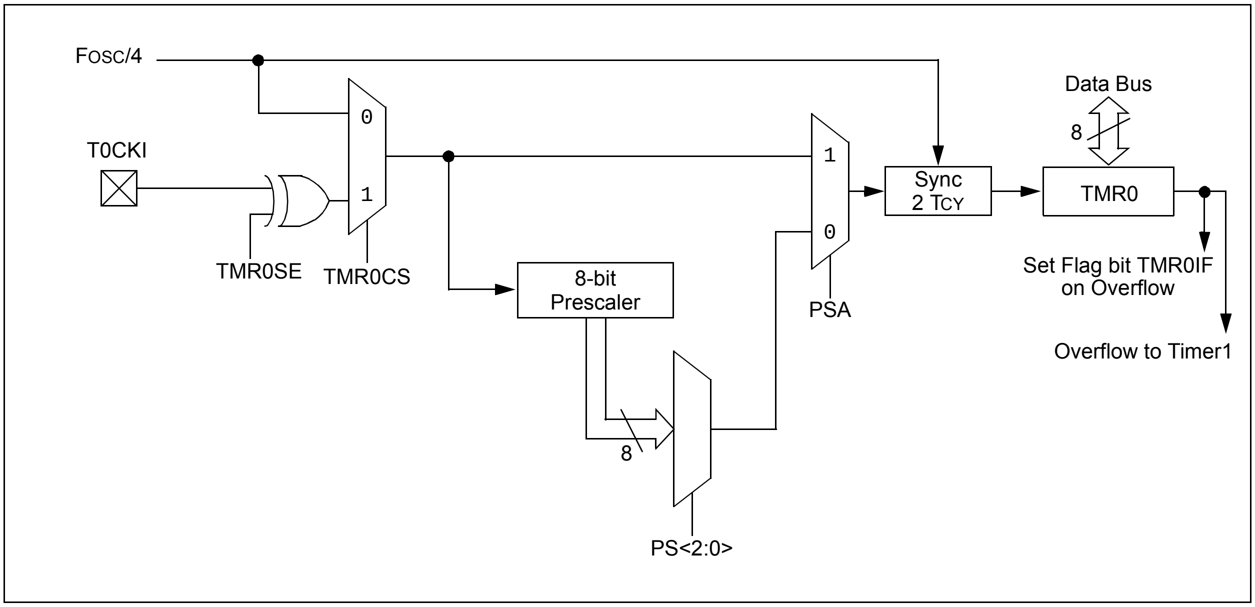
16.1.2 8-BIT COUNTER MODE

In 8-Bit Counter mode, the Timer0 module will increment on every rising or falling edge of the T0CKI pin.

8-Bit Counter mode using the T0CKI pin is selected by setting the TMR0CS bit in the OPTION_REG register to ‘1’.

The rising or falling transition of the incrementing edge for either input source is determined by the TMR0SE bit in the OPTION_REG register.

FIGURE 16-1: BLOCK DIAGRAM OF THE TIMER0



16.1.3 SOFTWARE PROGRAMMABLE PRESCALER

A software programmable prescaler is available for exclusive use with Timer0. The prescaler is enabled by clearing the PSA bit of the OPTION_REG register.

Note:	The Watchdog Timer (WDT) uses its own independent prescaler.

There are eight prescaler options for the Timer0 module ranging from 1:2 to 1:256. The prescale values are selectable via the PS<2:0> bits of the OPTION_REG register. In order to have a 1:1 prescaler value for the Timer0 module, the prescaler must be disabled by setting the PSA bit of the OPTION_REG register.

The prescaler is not readable or writable. All instructions writing to the TMR0 register will clear the prescaler.

16.1.4 TIMER0 INTERRUPT

Timer0 will generate an interrupt when the TMR0 register overflows from FFh to 00h. The TMR0IF interrupt flag bit of the INTCON register is set every time the TMR0 register overflows, regardless of whether or not the Timer0 interrupt is enabled. The TMR0IF bit can only be cleared in software. The Timer0 interrupt enable is the TMR0IE bit of the INTCON register.

Note:	The Timer0 interrupt cannot wake the processor from Sleep since the timer is frozen during Sleep.

16.1.5 8-BIT COUNTER MODE SYNCHRONIZATION

When in 8-Bit Counter mode, the incrementing edge on the T0CKI pin must be synchronized to the instruction clock. Synchronization can be accomplished by sampling the prescaler output on the Q2 and Q4 cycles of the instruction clock. The high and low periods of the external clocking source must meet the timing requirements as shown in **Section 21.0 “Electrical Specifications”**.

16.1.6 OPERATION DURING SLEEP

Timer0 cannot operate while the processor is in Sleep mode. The contents of the TMR0 register will remain unchanged while the processor is in Sleep mode.

16.2 Option and Timer0 Control Register

REGISTER 16-1: OPTION_REG: OPTION REGISTER

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
$\overline{\text{WPUEN}}$	INTEDG	TMR0CS	TMR0SE	PSA	PS<2:0>		
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7 **$\overline{\text{WPUEN}}$:** Weak Pull-up Enable bit

`1` = All weak pull-ups are disabled (except $\overline{\text{MCLR}}$, if it is enabled)
`0` = Weak pull-ups are enabled by individual WPUx latch values

bit 6 **INTEDG:** Interrupt Edge Select bit

`1` = Interrupt on rising edge of INT pin
`0` = Interrupt on falling edge of INT pin

bit 5 **TMR0CS:** Timer0 Clock Source Select bit

`1` = Transition on T0CKI pin
`0` = Internal instruction cycle clock (FOSC/4)

bit 4 **TMR0SE:** Timer0 Source Edge Select bit

`1` = Increment on high-to-low transition on T0CKI pin
`0` = Increment on low-to-high transition on T0CKI pin

bit 3 **PSA:** Prescaler Assignment bit

`1` = Prescaler is not assigned to the Timer0 module
`0` = Prescaler is assigned to the Timer0 module

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

Bit Value	Timer0 Rate
`000`	1 : 2
`001`	1 : 4
`010`	1 : 8
`011`	1 : 16
`100`	1 : 32
`101`	1 : 64
`110`	1 : 128
`111`	1 : 256

TABLE 16-1: SUMMARY OF REGISTERS ASSOCIATED WITH TIMER0

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	60
OPTION_REG	$\overline{\text{WPUEN}}$	INTEDG	TMR0CS	TMR0SE	PSA	PS<2:0>			121
TMR0	Timer0 Module Register								119*
TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	90

Legend: — = Unimplemented location, read as ‘`0`’. Shaded cells are not used by the Timer0 module.
* Page provides register information.

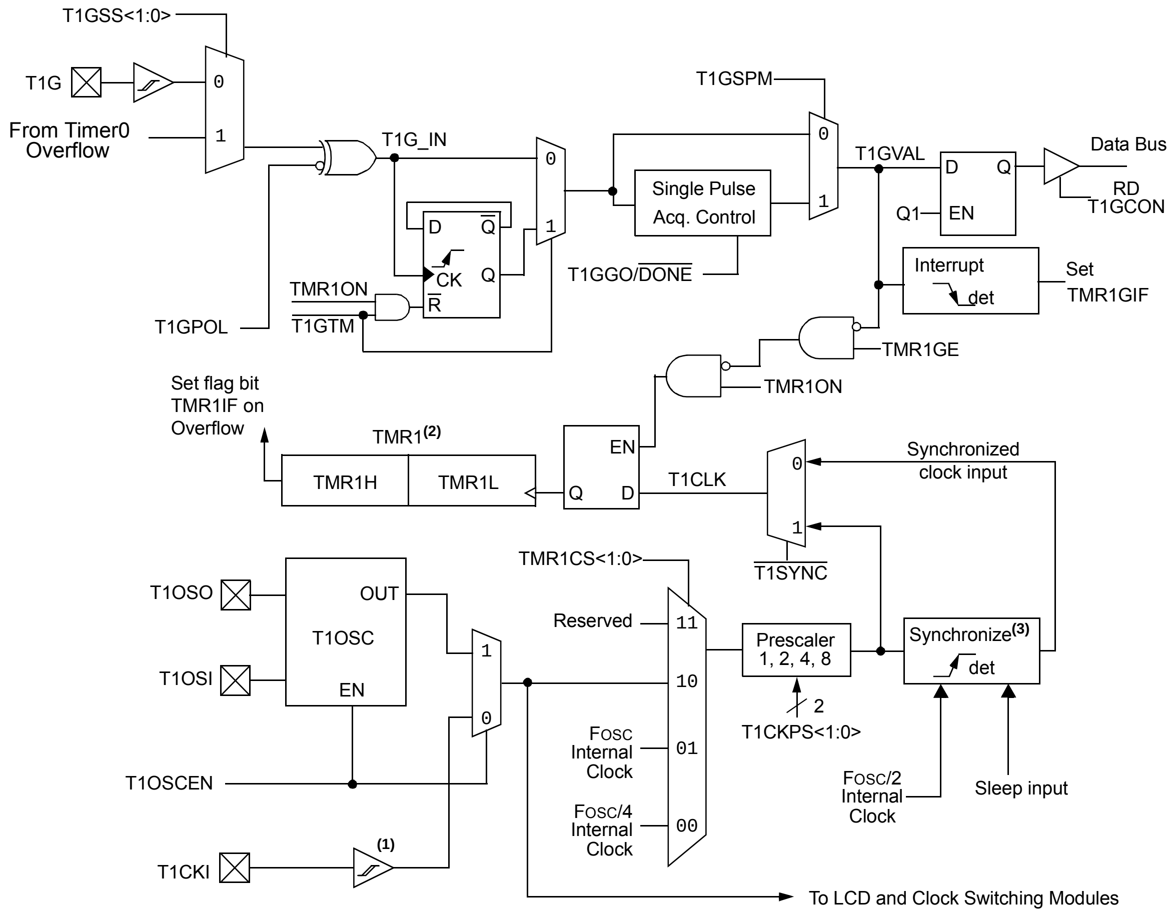
17.0 TIMER1 MODULE WITH GATE CONTROL

The Timer1 module is a 16-bit timer/counter with the following features:

- 16-bit timer/counter register pair (TMR1H:TMR1L)
- Programmable internal or external clock source
- 2-bit prescaler
- Dedicated 32 kHz oscillator circuit
- Multiple Timer1 gate (count enable) sources
- Interrupt on overflow
- Wake-up on overflow (external clock, Asynchronous mode only)
- Selectable Gate Source Polarity
- Gate Toggle mode
- Gate Single-pulse mode
- Gate Value Status
- Gate Event Interrupt

Figure 17-1 is a block diagram of the Timer1 module.

FIGURE 17-1: TIMER1 BLOCK DIAGRAM


T1GSS<1:0>
T1G
0
1
From Timer0 Overflow
T1G_IN
T1GSPM
0
1
0
1
T1GVAL
D
Q
Q1
EN
Data Bus
RD T1GCON
Single Pulse Acq. Control
D
Q̅
CK
Q
R̅
TMR1ON
T1GTM
T1GPOL
T1GGO/DONE
Interrupt det
Set TMR1GIF
TMR1GE
TMR1ON
Set flag bit TMR1IF on Overflow
TMR1(2)
TMR1H
TMR1L
EN
Q
D
T1CLK
0
1
Synchronized clock input
T1SYNC
TMR1CS<1:0>
T1OSO
T1OSC
OUT
EN
T1OSI
1
0
Reserved
11
10
01
00
Fosc Internal Clock
Fosc/4 Internal Clock
Prescaler 1, 2, 4, 8
2
T1CKPS<1:0>
Synchronize(3) det
Fosc/2 Internal Clock
Sleep input
T1OSCEN
T1CKI
(1)
To LCD and Clock Switching Modules

Note 1: ST Buffer is high speed type when using T1CKI.
2: Timer1 register increments on rising edge.
3: Synchronize does not operate while in Sleep.

17.1 Timer1 Operation

The Timer1 module is a 16-bit incrementing counter which is accessed through the TMR1H:TMR1L register pair. Writes to TMR1H or TMR1L directly update the counter.

When used with an internal clock source, the module is a timer and increments on every instruction cycle. When used with an external clock source, the module can be used as either a timer or counter and increments on every selected edge of the external source.

Timer1 is enabled by configuring the TMR1ON and TMR1GE bits in the T1CON and T1GCON registers, respectively. Table 17-1 displays the Timer1 enable selections.

TABLE 17-1: TIMER1 ENABLE SELECTIONS

TMR1ON	TMR1GE	Timer1 Operation
`0`	`0`	Off
`0`	`1`	Off
`1`	`0`	Always On
`1`	`1`	Count Enabled

17.2 Clock Source Selection

The TMR1CS<1:0> and T1OSCEN bits of the T1CON register are used to select the clock source for Timer1. Table 17-2 displays the clock source selections.

17.2.1 INTERNAL CLOCK SOURCE

When the internal clock source is selected the TMR1H:TMR1L register pair will increment on multiples of FOSC as determined by the Timer1 prescaler.

When the FOSC internal clock source is selected, the Timer1 register value will increment by four counts every instruction clock cycle. Due to this condition, a 2 LSB error in resolution will occur when reading the Timer1 value. To utilize the full resolution of Timer1, an asynchronous input signal must be used to gate the Timer1 clock input.

The following asynchronous source may be used:

- Asynchronous event on the T1G pin to Timer1 gate

17.2.2 EXTERNAL CLOCK SOURCE

When the external clock source is selected, the Timer1 module may work as a timer or a counter.

When enabled to count, Timer1 is incremented on the rising edge of the external clock input T1CKI or the capacitive sensing oscillator signal. Either of these external clock sources can be synchronized to the microcontroller system clock or they can run asynchronously.

When used as a timer with a clock oscillator, an external 32.768 kHz crystal can be used in conjunction with the dedicated internal oscillator circuit.

Note: In Counter mode, a falling edge must be registered by the counter prior to the first incrementing rising edge after any one or more of the following conditions:

- Timer1 enabled after POR
- Write to TMR1H or TMR1L
- Timer1 is disabled
- Timer1 is disabled (TMR1ON = `0`) when T1CKI is high then Timer1 is enabled (TMR1ON=`1`) when T1CKI is low.

TABLE 17-2: CLOCK SOURCE SELECTIONS

TMR1CS1	TMR1CS0	T1OSCEN	Clock Source
`0`	`0`	`x`	Instruction Clock (FOSC/4)
`0`	`1`	`x`	System Clock (FOSC)
`1`	`0`	`0`	External Clocking on T1CKI Pin
`1`	`0`	`1`	Osc. Circuit on T1OSI/T1OSO Pins
`1`	`1`	`x`	Reserved

17.3 Timer1 Prescaler

Timer1 has four prescaler options allowing 1, 2, 4 or 8 divisions of the clock input. The T1CKPS bits of the T1CON register control the prescale counter. The prescale counter is not directly readable or writable; however, the prescaler counter is cleared upon a write to TMR1H or TMR1L.

17.4 Timer1 Oscillator

A dedicated low-power 32.768 kHz oscillator circuit is built-in between pins T1OSI (input) and T1OSO. This internal circuit is to be used in conjunction with an external 32.768 kHz crystal.

The oscillator circuit is enabled by setting the T1OSCEN bit of the T1CON register. The oscillator will continue to run during Sleep.

Note: The oscillator requires a start-up and stabilization time before use. Thus, T1OSCEN should be set and a suitable delay observed prior to enabling Timer1.

17.5 Timer1 Operation in Asynchronous Counter Mode

If control bit $\overline{\text{T1SYNC}}$ of the T1CON register is set, the external clock input is not synchronized. The timer increments asynchronously to the internal phase clocks. If the external clock source is selected then the timer will continue to run during Sleep and can generate an interrupt on overflow, which will wake-up the processor. However, special precautions in software are needed to read/write the timer (see **Section 17.5.1 "Reading and Writing Timer1 in Asynchronous Counter Mode"**).

Note: When switching from synchronous to asynchronous operation, it is possible to skip an increment. When switching from asynchronous to synchronous operation, it is possible to produce an additional increment.

17.5.1 READING AND WRITING TIMER1 IN ASYNCHRONOUS COUNTER MODE

Reading TMR1H or TMR1L while the timer is running from an external asynchronous clock will ensure a valid read (taken care of in hardware). However, the user should keep in mind that reading the 16-bit timer in two 8-bit values itself, poses certain problems, since the timer may overflow between the reads.

For writes, it is recommended that the user simply stop the timer and write the desired values. A write contention may occur by writing to the timer registers, while the register is incrementing. This may produce an unpredictable value in the TMR1H:TMR1L register pair.

17.6 Timer1 Gate

Timer1 can be configured to count freely or the count can be enabled and disabled using Timer1 gate circuitry. This is also referred to as Timer1 Gate Enable.

Timer1 gate can also be driven by multiple selectable sources.

17.6.1 TIMER1 GATE ENABLE

The Timer1 Gate Enable mode is enabled by setting the TMR1GE bit of the T1GCON register. The polarity of the Timer1 Gate Enable mode is configured using the T1GPOL bit of the T1GCON register.

When Timer1 Gate Enable mode is enabled, Timer1 will increment on the rising edge of the Timer1 clock source. When Timer1 Gate Enable mode is disabled, no incrementing will occur and Timer1 will hold the current count. See Figure 17-3 for timing details.

TABLE 17-3: TIMER1 GATE ENABLE SELECTIONS

T1CLK	T1GPOL	T1G	Timer1 Operation
↑	`0`	`0`	Counts
↑	`0`	`1`	Holds Count
↑	`1`	`0`	Holds Count
↑	`1`	`1`	Counts

17.6.2 TIMER1 GATE SOURCE SELECTION

The Timer1 gate source can be selected from one of four different sources. Source selection is controlled by the T1GSS bits of the T1GCON register. The polarity for each available source is also selectable. Polarity selection is controlled by the T1GPOL bit of the T1GCON register.

TABLE 17-4: TIMER1 GATE SOURCES

T1GSS	Timer1 Gate Source
`00`	Timer1 Gate Pin
`01`	Overflow of Timer0 (TMR0 increments from FFh to 00h)

17.6.2.1 T1G Pin Gate Operation

The T1G pin is one source for Timer1 gate control. It can be used to supply an external source to the Timer1 gate circuitry.

17.6.2.2 Timer0 Overflow Gate Operation

When Timer0 increments from FFh to 00h, a low-to-high pulse will automatically be generated and internally supplied to the Timer1 gate circuitry.

17.6.3 TIMER1 GATE TOGGLE MODE

When Timer1 Gate Toggle mode is enabled, it is possible to measure the full-cycle length of a Timer1 gate signal, as opposed to the duration of a single level pulse.

The Timer1 gate source is routed through a flip-flop that changes state on every incrementing edge of the signal. See Figure 17-4 for timing details.

Timer1 Gate Toggle mode is enabled by setting the T1GTM bit of the T1GCON register. When the T1GTM bit is cleared, the flip-flop is cleared and held clear. This is necessary in order to control which edge is measured.

Note: Enabling Toggle mode at the same time as changing the gate polarity may result in indeterminate operation.

17.6.4 TIMER1 GATE SINGLE-PULSE MODE

When Timer1 Gate Single-Pulse mode is enabled, it is possible to capture a single pulse gate event. Timer1 Gate Single-Pulse mode is first enabled by setting the T1GSPM bit in the T1GCON register. Next, the T1GGO/$\overline{\text{DONE}}$ bit in the T1GCON register must be set. The Timer1 will be fully enabled on the next incrementing edge. On the next trailing edge of the pulse, the T1GGO/$\overline{\text{DONE}}$ bit will automatically be cleared. No other gate events will be allowed to increment Timer1 until the T1GGO/$\overline{\text{DONE}}$ bit is once again set in software. See Figure 17-5 for timing details.

If the Single Pulse Gate mode is disabled by clearing the T1GSPM bit in the T1GCON register, the T1GGO/$\overline{\text{DONE}}$ bit should also be cleared.

Enabling the Toggle mode and the Single-Pulse mode simultaneously will permit both sections to work together. This allows the cycle times on the Timer1 gate source to be measured. See Figure 17-6 for timing details.

17.6.5 TIMER1 GATE VALUE STATUS

When Timer1 gate value status is utilized, it is possible to read the most current level of the gate control value. The value is stored in the T1GVAL bit in the T1GCON register. The T1GVAL bit is valid even when the Timer1 gate is not enabled (TMR1GE bit is cleared).

17.6.6 TIMER1 GATE EVENT INTERRUPT

When Timer1 Gate Event Interrupt is enabled, it is possible to generate an interrupt upon the completion of a gate event. When the falling edge of T1GVAL occurs, the TMR1GIF flag bit in the PIR1 register will be set. If the TMR1GIE bit in the PIE1 register is set, then an interrupt will be recognized.

The TMR1GIF flag bit operates even when the Timer1 gate is not enabled (TMR1GE bit is cleared).

17.7 Timer1 Interrupt

The Timer1 register pair (TMR1H:TMR1L) increments to FFFFh and rolls over to 0000h. When Timer1 rolls over, the Timer1 interrupt flag bit of the PIR1 register is set. To enable the interrupt on rollover, you must set these bits:

- TMR1ON bit of the T1CON register
- TMR1IE bit of the PIE1 register
- PEIE bit of the INTCON register
- GIE bit of the INTCON register

The interrupt is cleared by clearing the TMR1IF bit in the Interrupt Service Routine.

Note: The TMR1H:TMR1L register pair and the TMR1IF bit should be cleared before enabling interrupts.

17.8 Timer1 Operation During Sleep

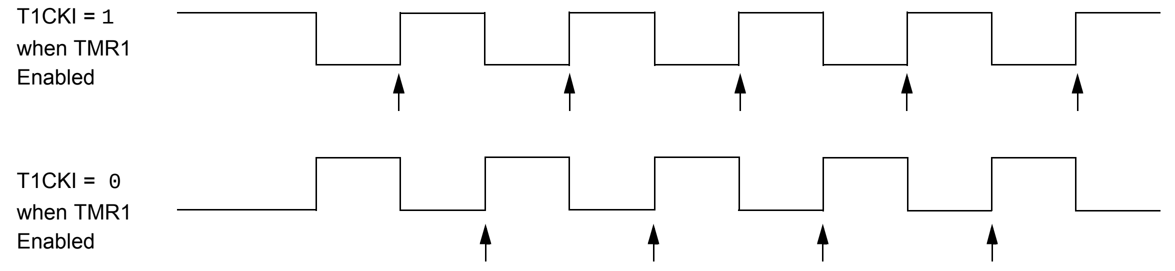
Timer1 can only operate during Sleep when setup in Asynchronous Counter mode. In this mode, an external crystal or clock source can be used to increment the counter. To set up the timer to wake the device:

- TMR1ON bit of the T1CON register must be set
- TMR1IE bit of the PIE1 register must be set
- PEIE bit of the INTCON register must be set
- $\overline{\text{T1SYNC}}$ bit of the T1CON register must be set
- TMR1CS bits of the T1CON register must be configured
- T1OSCEN bit of the T1CON register must be configured

The device will wake-up on an overflow and execute the next instructions. If the GIE bit of the INTCON register is set, the device will call the Interrupt Service Routine.

Timer1 oscillator will continue to operate in Sleep regardless of the $\overline{\text{T1SYNC}}$ bit setting.

FIGURE 17-2: TIMER1 INCREMENTING EDGE



Note 1: Arrows indicate counter increments.

2: In Counter mode, a falling edge must be registered by the counter prior to the first incrementing rising edge of the clock.

FIGURE 17-3: TIMER1 GATE ENABLE MODE

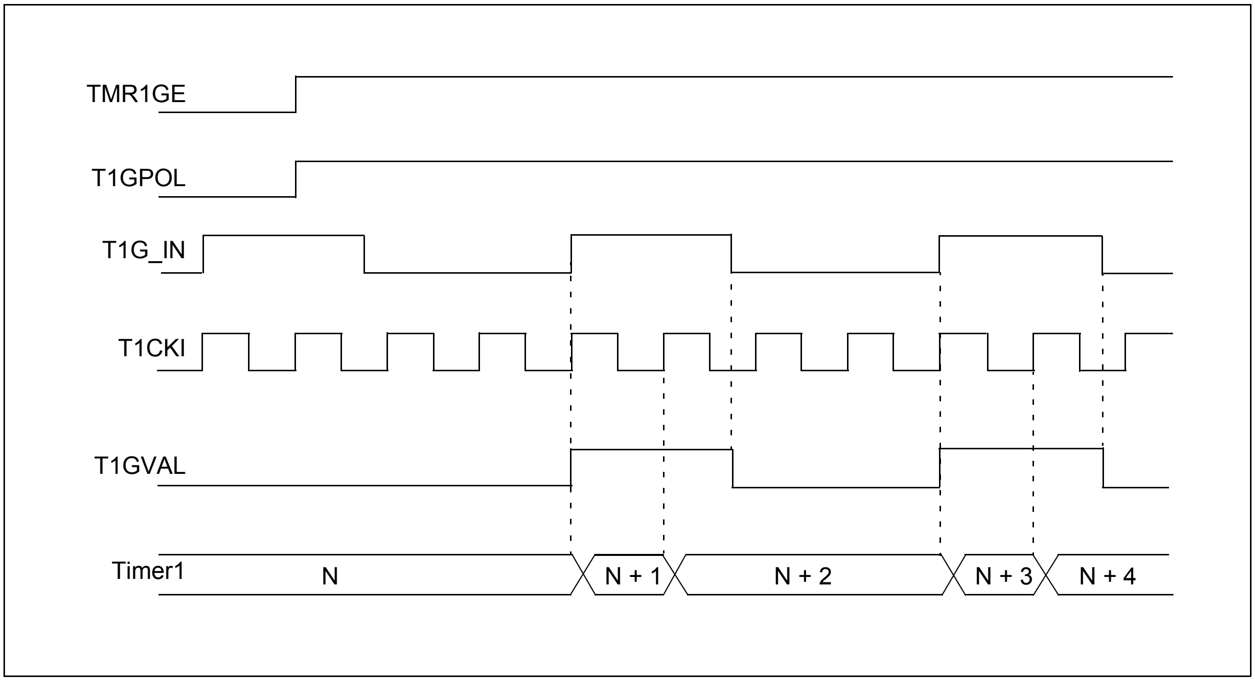

TMR1GE
T1GPOL
T1G_IN
T1CKI
T1GVAL
Timer1
N
N + 1
N + 2
N + 3
N + 4

FIGURE 17-4: TIMER1 GATE TOGGLE MODE

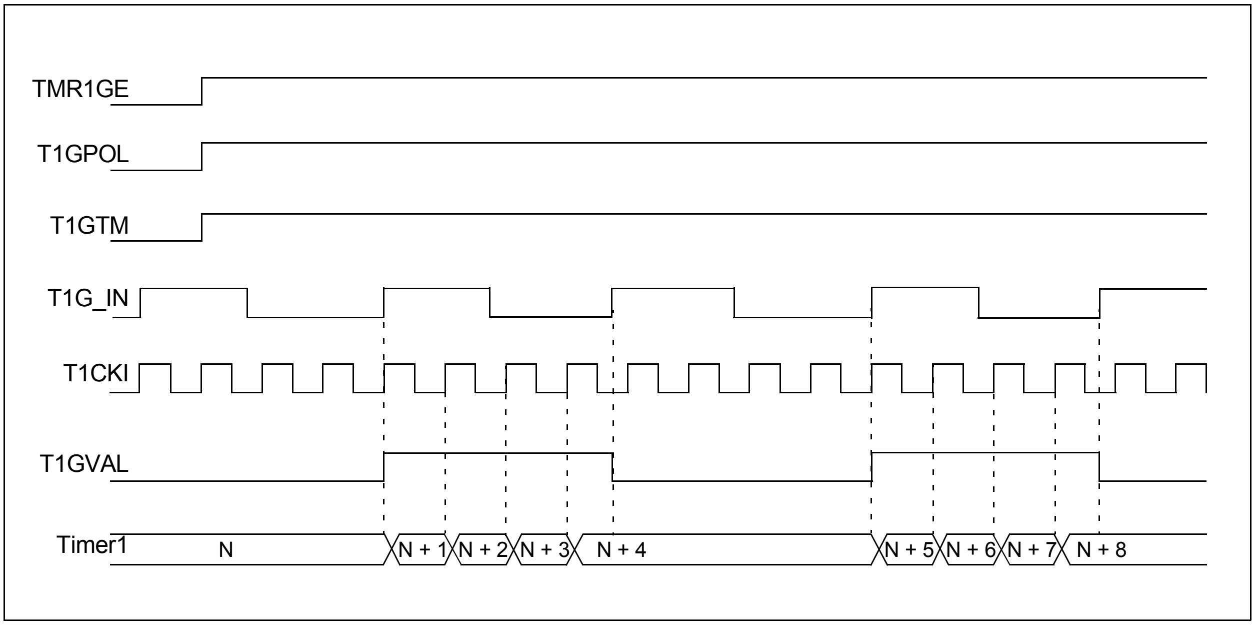

TMR1GE
T1GPOL
T1GTM
T1G_IN
T1CKI
T1GVAL
Timer1
N
N + 1
N + 2
N + 3
N + 4
N + 5
N + 6
N + 7
N + 8

FIGURE 17-5: TIMER1 GATE SINGLE-PULSE MODE

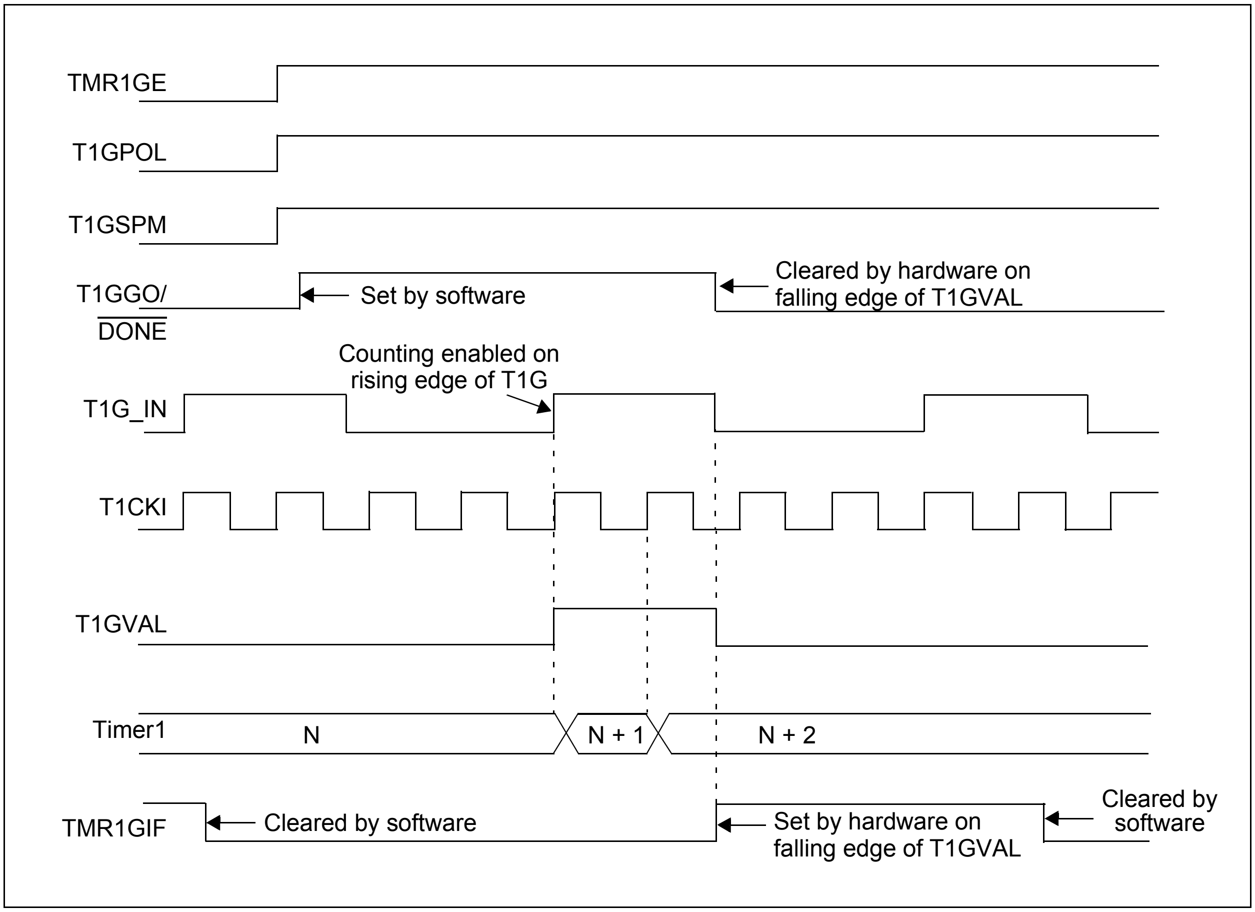
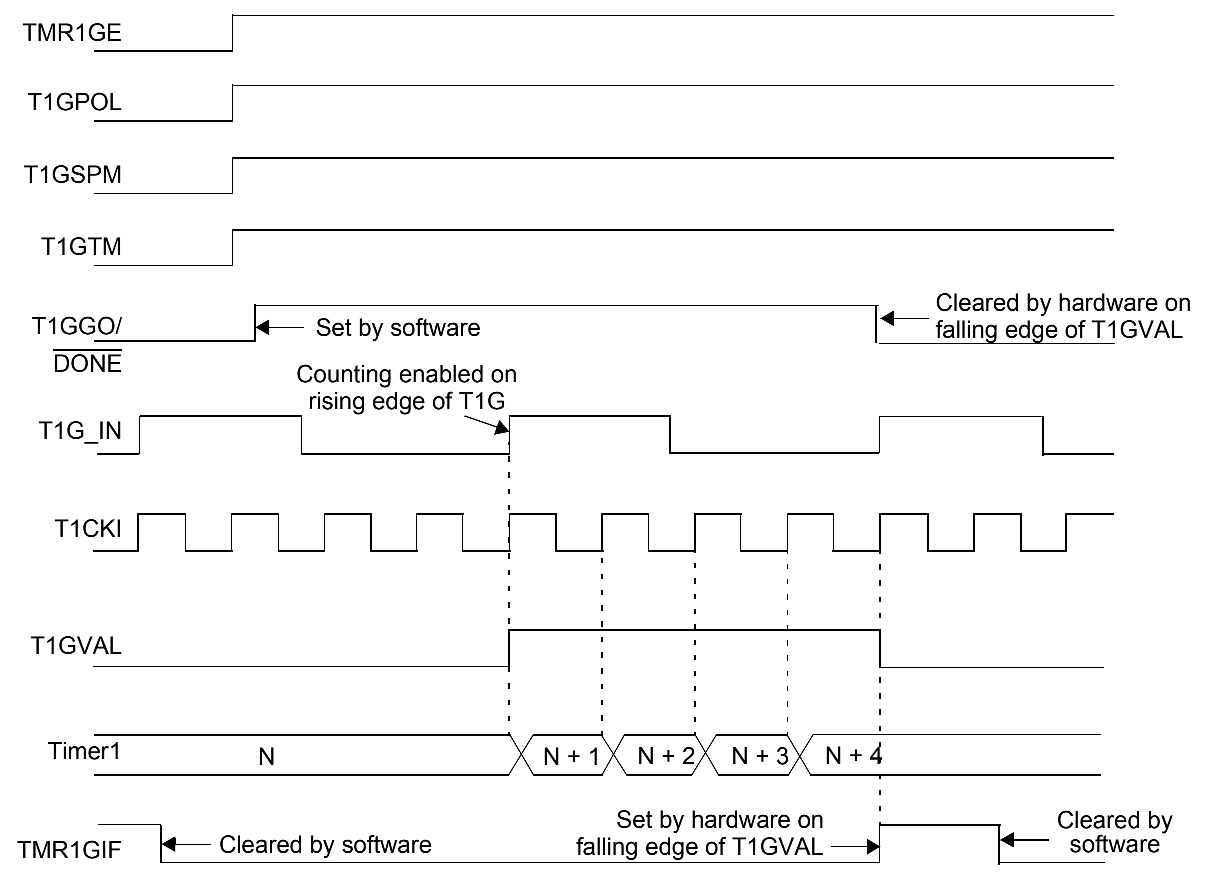

TMR1GE
T1GPOL
T1GSPM
T1GGO/DONE
Set by software
Cleared by hardware on falling edge of T1GVAL
Counting enabled on rising edge of T1G
T1G_IN
T1CKI
T1GVAL
Timer1
N
N + 1
N + 2
TMR1GIF
Cleared by software
Set by hardware on falling edge of T1GVAL
Cleared by software

FIGURE 17-6: TIMER1 GATE SINGLE-PULSE AND TOGGLE COMBINED MODE


TMR1GE
T1GPOL
T1GSPM
T1GTM
T1GGO/$\overline{\text{DONE}}$
Set by software
Cleared by hardware on falling edge of T1GVAL
Counting enabled on rising edge of T1G
T1G_IN
T1CKI
T1GVAL
Timer1
N
N + 1
N + 2
N + 3
N + 4
TMR1GIF
Cleared by software
Set by hardware on falling edge of T1GVAL
Cleared by software

17.9 Timer1 Control Register

The Timer1 Control register (T1CON), shown in Register 17-1, is used to control Timer1 and select the various features of the Timer1 module.

REGISTER 17-1: T1CON: TIMER1 CONTROL REGISTER

R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	U-0	R/W-0/u
TMR1CS<1:0>		T1CKPS<1:0>		T1OSCEN	$\overline{\text{T1SYNC}}$	—	TMR1ON
bit 7							bit 0

Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-6 **TMR1CS<1:0>:** Timer1 Clock Source Select bits

`11` = Reserved
`10` = Timer1 clock source is pin or oscillator:
If T1OSCEN = `0`:
External clock from T1CKI pin (on the rising edge)
If T1OSCEN = `1`:
Crystal oscillator on T1OSI/T1OSO pins
`01` = Timer1 clock source is system clock (FOSC)
`00` = Timer1 clock source is instruction clock (FOSC/4)

bit 5-4 **T1CKPS<1:0>:** Timer1 Input Clock Prescale Select bits

`11` = 1:8 Prescale value
`10` = 1:4 Prescale value
`01` = 1:2 Prescale value
`00` = 1:1 Prescale value

bit 3 **T1OSCEN:** LP Oscillator Enable Control bit

`1` = Dedicated Timer1 oscillator circuit enabled
`0` = Dedicated Timer1 oscillator circuit disabled

bit 2 **$\overline{\text{T1SYNC}}$:** Timer1 External Clock Input Synchronization Control bit

TMR1CS<1:0> = `1X`
`1` = Do not synchronize external clock input
`0` = Synchronize external clock input with system clock (FOSC)

TMR1CS<1:0> = `0X`
This bit is ignored. Timer1 uses the internal clock when TMR1CS<1:0> = `1X`.

bit 1 **Unimplemented:** Read as ‘`0`’

bit 0 **TMR1ON:** Timer1 On bit

`1` = Enables Timer1
`0` = Stops Timer1
Clears Timer1 gate flip-flop

17.10 Timer1 Gate Control Register

The Timer1 Gate Control register (T1GCON), shown in Register 17-2, is used to control Timer1 gate.

REGISTER 17-2: T1GCON: TIMER1 GATE CONTROL REGISTER

R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R/W/HC-0/u	R-x/x	R/W-0/u	R/W-0/u
TMR1GE	T1GPOL	T1GTM	T1GSPM	T1GGO/$\overline{\text{DONE}}$	T1GVAL	T1GSS<1:0>	
bit 7							bit 0

Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	HC = Bit is cleared by hardware

bit 7 **TMR1GE:** Timer1 Gate Enable bit

If TMR1ON = 0:
This bit is ignored
If TMR1ON = 1:
`1` = Timer1 counting is controlled by the Timer1 gate function
`0` = Timer1 counts regardless of Timer1 gate function

bit 6 **T1GPOL:** Timer1 Gate Polarity bit

`1` = Timer1 gate is active-high (Timer1 counts when gate is high)
`0` = Timer1 gate is active-low (Timer1 counts when gate is low)

bit 5 **T1GTM:** Timer1 Gate Toggle Mode bit

`1` = Timer1 Gate Toggle mode is enabled
`0` = Timer1 Gate Toggle mode is disabled and toggle flip-flop is cleared
Timer1 gate flip-flop toggles on every rising edge.

bit 4 **T1GSPM:** Timer1 Gate Single-Pulse Mode bit

`1` = Timer1 gate Single-Pulse mode is enabled and is controlling Timer1 gate
`0` = Timer1 gate Single-Pulse mode is disabled

bit 3 **T1GGO/$\overline{\text{DONE}}$:** Timer1 Gate Single-Pulse Acquisition Status bit

`1` = Timer1 gate single-pulse acquisition is ready, waiting for an edge
`0` = Timer1 gate single-pulse acquisition has completed or has not been started

bit 2 **T1GVAL:** Timer1 Gate Current State bit

Indicates the current state of the Timer1 gate that could be provided to TMR1H:TMR1L.
Unaffected by Timer1 Gate Enable (TMR1GE).

bit 1-0 **T1GSS<1:0>:** Timer1 Gate Source Select bits

`00` = Timer1 gate pin
`01` = Timer0 overflow output
`10` = Reserved
`11` = Reserved

TABLE 17-5: SUMMARY OF REGISTERS ASSOCIATED WITH TIMER1

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	60
PIE1	TMR1GIE	ADIE	—	—	—	—	—	TMR1IE	61
PIR1	TMR1GIF	ADIF	—	—	—	—	—	TMR1IF	63
TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								126*
TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								126*
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	96
T1CON	TMR1CS1	TMR1CS0	T1CKPS<1:0>		T1OSCEN	$\overline{\text{T1SYNC}}$	—	TMR1ON	130
T1GCON	TMR1GE	T1GPOL	T1GTM	T1GSPM	T1GGO/$\overline{\text{DONE}}$	T1GVAL	T1GSS1	T1GSS0	131

Legend: — = unimplemented location, read as ‘0’. Shaded cells are not used by the Timer1 module.
* Page provides register information.

18.0 LIQUID CRYSTAL DISPLAY (LCD) DRIVER MODULE

The Liquid Crystal Display (LCD) Driver module generates the timing control to drive a static or multiplexed LCD panel. In the PIC16LF1902/3 device, the module drives the panels of up to four commons and up to 72 total segments. The LCD module also provides control of the LCD pixel data.

The LCD Driver module supports:

- Direct driving of LCD panel
- Three LCD clock sources with selectable prescaler
- Up to four common pins:
  - Static (1 common)
  - 1/2 multiplex (2 commons)
  - 1/3 multiplex (3 commons)
  - 1/4 multiplex (4 commons)
- 19 Segment pins
- Static, 1/2 or 1/3 LCD Bias

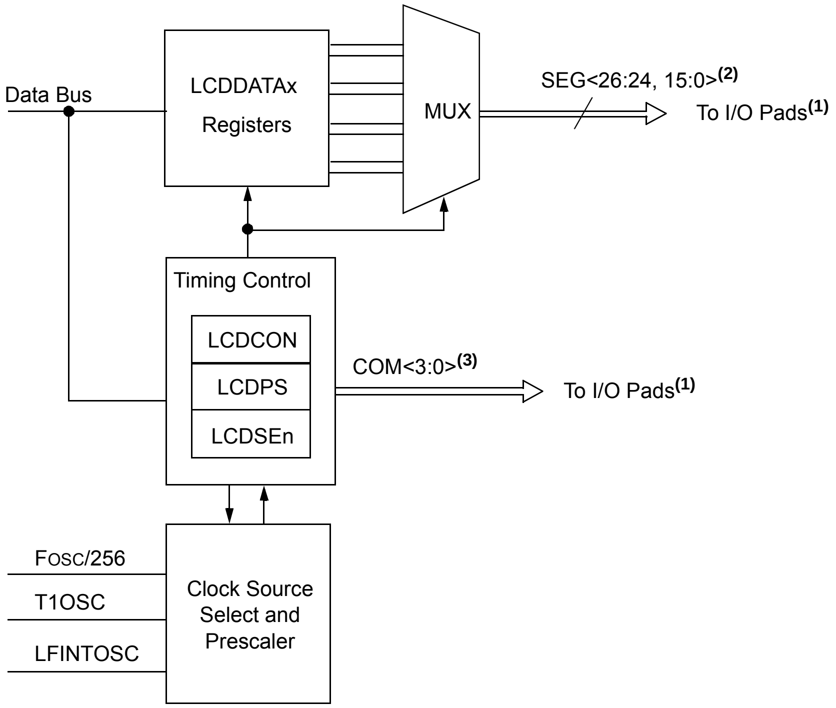
Note: COM3 and SEG15 share the same physical pin on the PIC16LF1902/3, therefore SEG15 is not available when using 1/4 multiplex displays.

18.1 LCD Registers

The module contains the following registers:

- LCD Control register (LCDCON)
- LCD Phase register (LCDPS)
- LCD Reference Ladder register (LCDRL)
- LCD Contrast Control register (LCDCST)
- LCD Reference Voltage Control register (LCDREF)
- Up to 3 LCD Segment Enable registers (LCDSEn)
- Up to 12 LCD data registers (LCDDATAn)

FIGURE 18-1: LCD DRIVER MODULE BLOCK DIAGRAM


Data Bus
LCDDATAx Registers
MUX
SEG<26:24, 15:0>(2)
To I/O Pads(1)
Timing Control
LCDCON
LCDPS
LCDSEn
COM<3:0>(3)
To I/O Pads(1)
FOSC/256
T1OSC
LFINTOSC
Clock Source Select and Prescaler

Note 1: These are not directly connected to the I/O pads, but may be tri-stated, depending on the configuration of the LCD module.

2: COM3 and SEG15 share the same physical pin, therefore SEG15 is not available when using 1/4 multiplex displays.

TABLE 18-1: LCD SEGMENT AND DATA REGISTERS

Device	# of LCD Registers	
	Segment Enable	Data
PIC16LF1902/3	3	12

The LCDCON register (Register 18-1) controls the operation of the LCD Driver module. The LCDPS register (Register 18-2) configures the LCD clock source prescaler and the type of waveform; Type-A or Type-B. The LCDSEn registers (Register 18-5) configure the functions of the port pins.

The following LCDSEn registers are available:

- LCDSE0 SE<7:0>
- LCDSE1 SE<15:8>
- LCDSE3 SE<26:24>

Once the module is initialized for the LCD panel, the individual bits of the LCDDATAn registers are cleared/set to represent a clear/dark pixel, respectively:

- LCDDATA0 SEG<7:0>COM0
- LCDDATA1 SEG<15:8>COM0
- LCDDATA3 SEG<7:0>COM1
- LCDDATA4 SEG<15:8>COM1
- LCDDATA6 SEG<7:0>COM2
- LCDDATA7 SEG<15:8>COM2
- LCDDATA9 SEG<7:0>COM3
- LCDDATA10 SEG<15:8>COM3
- LCDDATA12 SEG<26:24>COM0
- LCDDATA15 SEG<26:24>COM1
- LCDDATA18 SEG<26:24>COM2
- LCDDATA21 SEG<26:24>COM3

As an example, LCDDATAn is detailed in Register 18-6.

Once the module is configured, the LCDEN bit of the LCDCON register is used to enable or disable the LCD module. The LCD panel can also operate during Sleep by clearing the SLPEN bit of the LCDCON register.

REGISTER 18-1: LCDCON: LIQUID CRYSTAL DISPLAY (LCD) CONTROL REGISTER

R/W-0/0	R/W-0/0	R/C-0/0	U-0	R/W-0/0	R/W-0/0	R/W-1/1	R/W-1/1
LCDEN	SLPEN	WERR	—	CS<1:0>		LMUX<1:0>	
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	C = Only clearable bit

bit 7 **LCDEN:** LCD Driver Enable bit
`1` = LCD Driver module is enabled
`0` = LCD Driver module is disabled

bit 6 **SLPEN:** LCD Driver Enable in Sleep Mode bit
`1` = LCD Driver module is disabled in Sleep mode
`0` = LCD Driver module is enabled in Sleep mode

bit 5 **WERR:** LCD Write Failed Error bit
`1` = LCDDATAn register written while the WA bit of the LCDPS register = `0` (must be cleared in software)
`0` = No LCD write error

bit 4 **Unimplemented:** Read as ‘`0`’

bit 3-2 **CS<1:0>:** Clock Source Select bits
`00` = FOSC/256
`01` = T1OSC (Timer1)
`1x` = LFINTOSC (31 kHz)

bit 1-0 **LMUX<1:0>:** Commons Select bits

LMUX<1:0>	Multiplex	Maximum Number of Pixels PIC16LF1902/3	Bias
`00`	Static (COM0)	19	Static
`01`	1/2 (COM<1:0>)	38	1/2 or 1/3
`10`	1/3 (COM<2:0>)	57	1/2 or 1/3
`11`	1/4 (COM<3:0>)	72[(1)]	1/3

Note 1: On these devices, COM3 and SEG15 are shared on one pin, limiting the device from driving 72 segments.

REGISTER 18-2: LCDPS: LCD PHASE REGISTER

R/W-0/0	R/W-0/0	R-0/0	R-0/0	R/W-0/0	R/W-0/0	R/W-1/1	R/W-1/1
WFT	BIASMD	LCDA	WA	LP<3:0>			
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	C = Only clearable bit

bit 7 **WFT:** Waveform Type bit

`1` = Type-B phase changes on each frame boundary
`0` = Type-A phase changes within each common type

bit 6 **BIASMD:** Bias Mode Select bit

When LMUX<1:0> = `00`:
`0` = Static Bias mode (do not set this bit to ‘`1`’)

When LMUX<1:0> = `01`:
`1` = 1/2 Bias mode
`0` = 1/3 Bias mode

When LMUX<1:0> = `10`:
`1` = 1/2 Bias mode
`0` = 1/3 Bias mode

When LMUX<1:0> = `11`:
`0` = 1/3 Bias mode (do not set this bit to ‘`1`’)

bit 5 **LCDA:** LCD Active Status bit

`1` = LCD Driver module is active
`0` = LCD Driver module is inactive

bit 4 **WA:** LCD Write Allow Status bit

`1` = Writing to the LCDDATAn registers is allowed
`0` = Writing to the LCDDATAn registers is not allowed

bit 3-0 **LP<3:0>:** LCD Prescaler Selection bits

`1111` = 1:16
`1110` = 1:15
`1101` = 1:14
`1100` = 1:13
`1011` = 1:12
`1010` = 1:11
`1001` = 1:10
`1000` = 1:9
`0111` = 1:8
`0110` = 1:7
`0101` = 1:6
`0100` = 1:5
`0011` = 1:4
`0010` = 1:3
`0001` = 1:2
`0000` = 1:1

REGISTER 18-3: LCDREF: LCD REFERENCE VOLTAGE CONTROL REGISTER

R/W-0/0	U-0	R/W-0/0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	U-0
LCDIRE	—	LCDIRI	—	VLCD3PE	VLCD2PE	VLCD1PE	—
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	C = Only clearable bit

bit 7 **LCDIRE:** LCD Internal Reference Enable bit
`1` = Internal LCD Reference is enabled and connected to the Internal Contrast Control circuit
`0` = Internal LCD Reference is disabled

bit 6 **Unimplemented:** Read as ‘`0`’

bit 5 **LCDIRI:** LCD Internal Reference Ladder Idle Enable bit
Allows the Internal FVR buffer to shut down when the LCD Reference Ladder is in power mode ‘B’
`1` = When the LCD Reference Ladder is in power mode ‘B’, the LCD Internal FVR buffer is disabled.
`0` = The LCD Internal FVR Buffer ignores the LCD Reference Ladder Power mode.

bit 4 **Unimplemented:** Read as ‘`0`’

bit 3 **VLCD3PE:** VLCD3 Pin Enable bit
`1` = The VLCD3 pin is connected to the internal bias voltage LCDBIAS3[(1)]
`0` = The VLCD3 pin is not connected

bit 2 **VLCD2PE:** VLCD2 Pin Enable bit
`1` = The VLCD2 pin is connected to the internal bias voltage LCDBIAS2[(1)]
`0` = The VLCD2 pin is not connected

bit 1 **VLCD1PE:** VLCD1 Pin Enable bit
`1` = The VLCD1 pin is connected to the internal bias voltage LCDBIAS1[(1)]
`0` = The VLCD1 pin is not connected

bit 0 **Unimplemented:** Read as ‘`0`’

Note 1: Normal pin controls of TRISx and ANSELx are unaffected.

REGISTER 18-4: LCDCST: LCD CONTRAST CONTROL REGISTER

U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	—	LCDCST<2:0>		
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	C = Only clearable bit

bit 7-3 **Unimplemented:** Read as ‘`0`’

bit 2-0 **LCDCST<2:0>:** LCD Contrast Control bits
Selects the resistance of the LCD contrast control resistor ladder

Bit Value = Resistor ladder
`000` = Minimum Resistance (Maximum contrast). Resistor ladder is shorted.
`001` = Resistor ladder is at 1/7th of maximum resistance
`010` = Resistor ladder is at 2/7th of maximum resistance
`011` = Resistor ladder is at 3/7th of maximum resistance
`100` = Resistor ladder is at 4/7th of maximum resistance
`101` = Resistor ladder is at 5/7th of maximum resistance
`110` = Resistor ladder is at 6/7th of maximum resistance
`111` = Resistor ladder is at maximum resistance (Minimum contrast).

REGISTER 18-5: LCDSEn: LCD SEGMENT ENABLE REGISTERS

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
SEn	SEn	SEn	SEn	SEn	SEn	SEn	SEn
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0	**SEn:** Segment Enable bits `1` = Segment function of the pin is enabled `0` = I/O function of the pin is enabled

REGISTER 18-6: LCDDATAn: LCD DATA REGISTERS

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
SEGx-COMy	SEGx-COMy	SEGx-COMy	SEGx-COMy	SEGx-COMy	SEGx-COMy	SEGx-COMy	SEGx-COMy
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-0	**SEGx-COMy:** Pixel On bits `1` = Pixel on (dark) `0` = Pixel off (clear)

18.2 LCD Clock Source Selection

The LCD module has three possible clock sources:

- FOSC/256
- T1OSC
- LFINTOSC

The first clock source is the system clock divided by 256 (FOSC/256). This divider ratio is chosen to provide about 1 kHz output when the system clock is 8 MHz. The divider is not programmable. Instead, the LCD prescaler bits LP<3:0> of the LCDPS register are used to set the LCD frame clock rate.

The second clock source is the T1OSC. This also gives about 1 kHz when a 32.768 kHz crystal is used with the Timer1 oscillator. To use the Timer1 oscillator as a clock source, the T1OSCEN bit of the T1CON register should be set.

The third clock source is the 31 kHz LFINTOSC, which provides approximately 1 kHz output.

The second and third clock sources may be used to continue running the LCD while the processor is in Sleep.

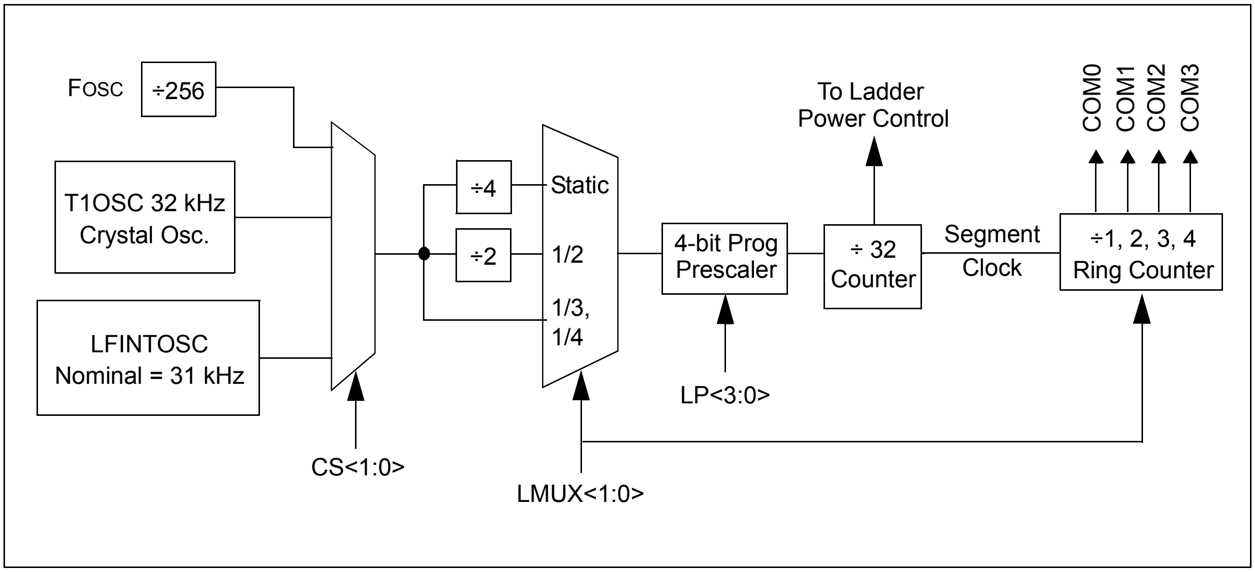
Using bits CS<1:0> of the LCDCON register can select any of these clock sources.

18.2.1 LCD PRESCALER

A 4-bit counter is available as a prescaler for the LCD clock. The prescaler is not directly readable or writable; its value is set by the LP<3:0> bits of the LCDPS register, which determine the prescaler assignment and prescale ratio.

The prescale values are selectable from 1:1 through 1:16.

FIGURE 18-2: LCD CLOCK GENERATION


FOSC
÷256
T1OSC 32 kHz
Crystal Osc.
LFINTOSC
Nominal = 31 kHz
CS<1:0>
÷4
÷2
Static
1/2
1/3,
1/4
LMUX<1:0>
4-bit Prog
Prescaler
LP<3:0>
÷ 32
Counter
To Ladder
Power Control
Segment
Clock
÷1, 2, 3, 4
Ring Counter
COM0
COM1
COM2
COM3

18.3 LCD Bias Voltage Generation

The LCD module can be configured for one of three bias types:

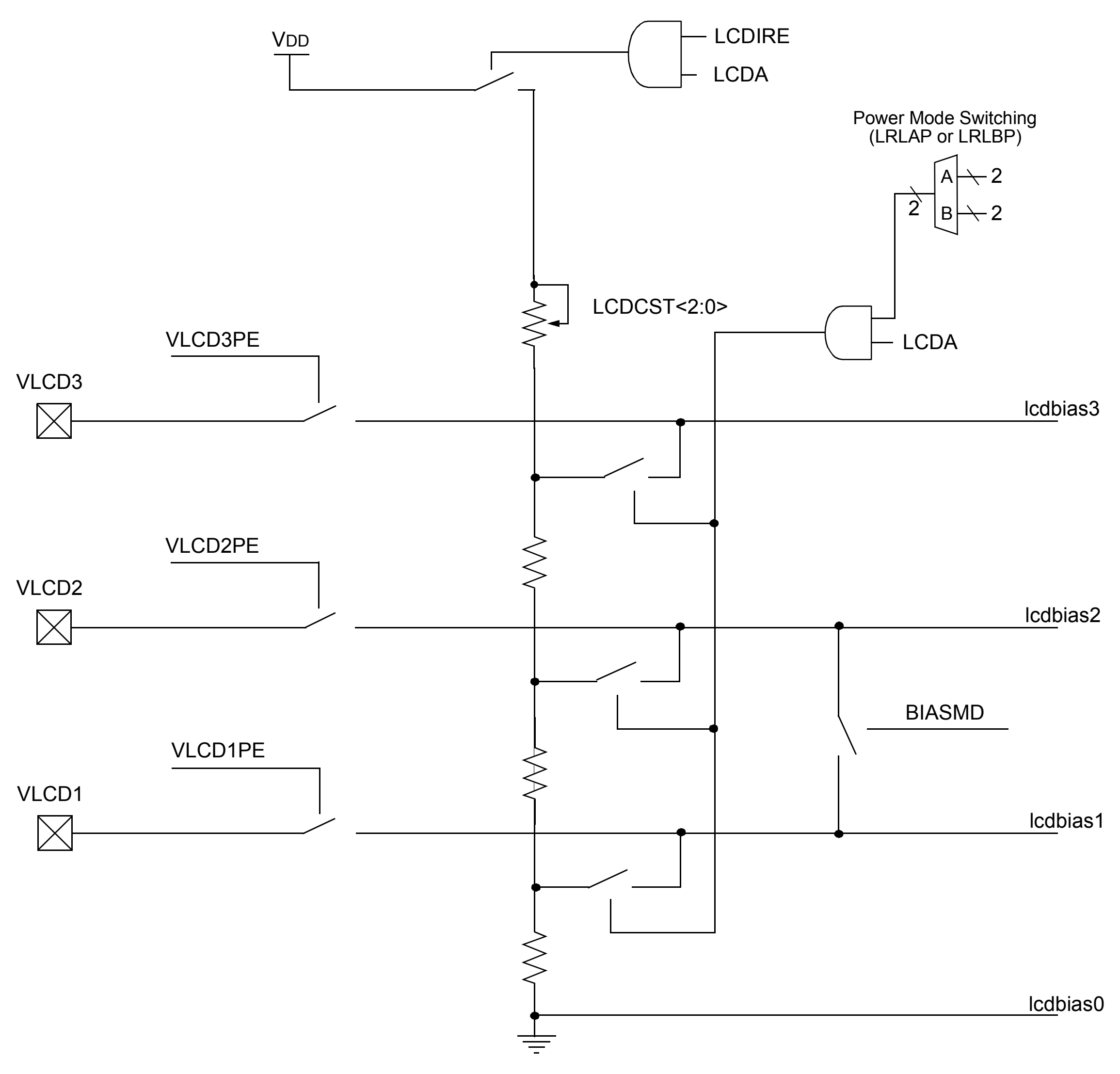
- Static Bias (2 voltage levels: VSS and VLCD)
- 1/2 Bias (3 voltage levels: VSS, 1/2 VLCD and VLCD)
- 1/3 Bias (4 voltage levels: VSS, 1/3 VLCD, 2/3 VLCD and VLCD)

TABLE 18-2: LCD BIAS VOLTAGES

	Static Bias	1/2 Bias	1/3 Bias
LCD Bias 0	VSS	VSS	VSS
LCD Bias 1	—	1/2 VDD	1/3 VDD
LCD Bias 2	—	1/2 VDD	2/3 VDD
LCD Bias 3	VLCD3	VLCD3	VLCD3

So that the user is not forced to place external components and use up to three pins for bias voltage generation, internal contrast control and an internal reference ladder are provided internally to the PIC16LF1902/3. Both of these features may be used in conjunction with the external VLCD<3:1> pins, to provide maximum flexibility. Refer to Figure 18-3.

FIGURE 18-3: LCD BIAS VOLTAGE GENERATION BLOCK DIAGRAM


VDD
LCDIRE
LCDA
Power Mode Switching
(LRLAP or LRLBP)
A
B
2
2
2
LCDCST<2:0>
LCDA
VLCD3PE
VLCD3
lcdbias3
VLCD2PE
VLCD2
lcdbias2
BIASMD
VLCD1PE
VLCD1
lcdbias1
lcdbias0

18.4 LCD Bias Internal Reference Ladder

The internal reference ladder can be used to divide the LCD bias voltage two or three equally spaced voltages that will be supplied to the LCD segment pins. To create this, the reference ladder consists of three matched resistors. Refer to Figure 18-3.

18.4.1 BIAS MODE INTERACTION

When in 1/2 Bias mode (BIASMD = `1`), then the middle resistor of the ladder is shorted out so that only two voltages are generated. The current consumption of the ladder is higher in this mode, with the one resistor removed.

TABLE 18-3: LCD INTERNAL LADDER POWER MODES (1/3 BIAS)

Power Mode	Nominal Resistance of Entire Ladder	Nominal IDD
Low	3 Mohm	1 µA
Medium	300 kohm	10 µA
High	30 kohm	100 µA

18.4.2 POWER MODES

The internal reference ladder may be operated in one of three power modes. This allows the user to trade off LCD contrast for power in the specific application. The larger the LCD glass, the more capacitance is present on a physical LCD segment, requiring more current to maintain the same contrast level.

Three different power modes are available, LP, MP and HP. The internal reference ladder can also be turned off for applications that wish to provide an external ladder or to minimize power consumption. Disabling the internal reference ladder results in all of the ladders being disconnected, allowing external voltages to be supplied.

Whenever the LCD module is inactive (LCDA = `0`), the internal reference ladder will be turned off.

18.4.3 AUTOMATIC POWER MODE SWITCHING

As an LCD segment is electrically only a capacitor, current is drawn only during the interval where the voltage is switching. To minimize total device current, the LCD internal reference ladder can be operated in a different power mode for the transition portion of the duration. This is controlled by the LCDRL Register (Register 18-7).

The LCDRL register allows switching between two power modes, designated 'A' and 'B'. 'A' Power mode is active for a programmable time, beginning at the time when the LCD segments transition. 'B' Power mode is the remaining time before the segments or commons change again. The LRLAT<2:0> bits select how long, if any, that the 'A' Power mode is active. Refer to Figure 18-4.

To implement this, the 5-bit prescaler used to divide the 32 kHz clock down to the LCD controller's 1 kHz base rate is used to select the power mode.

FIGURE 18-4: LCD INTERNAL REFERENCE LADDER POWER MODE SWITCHING DIAGRAM – TYPE A

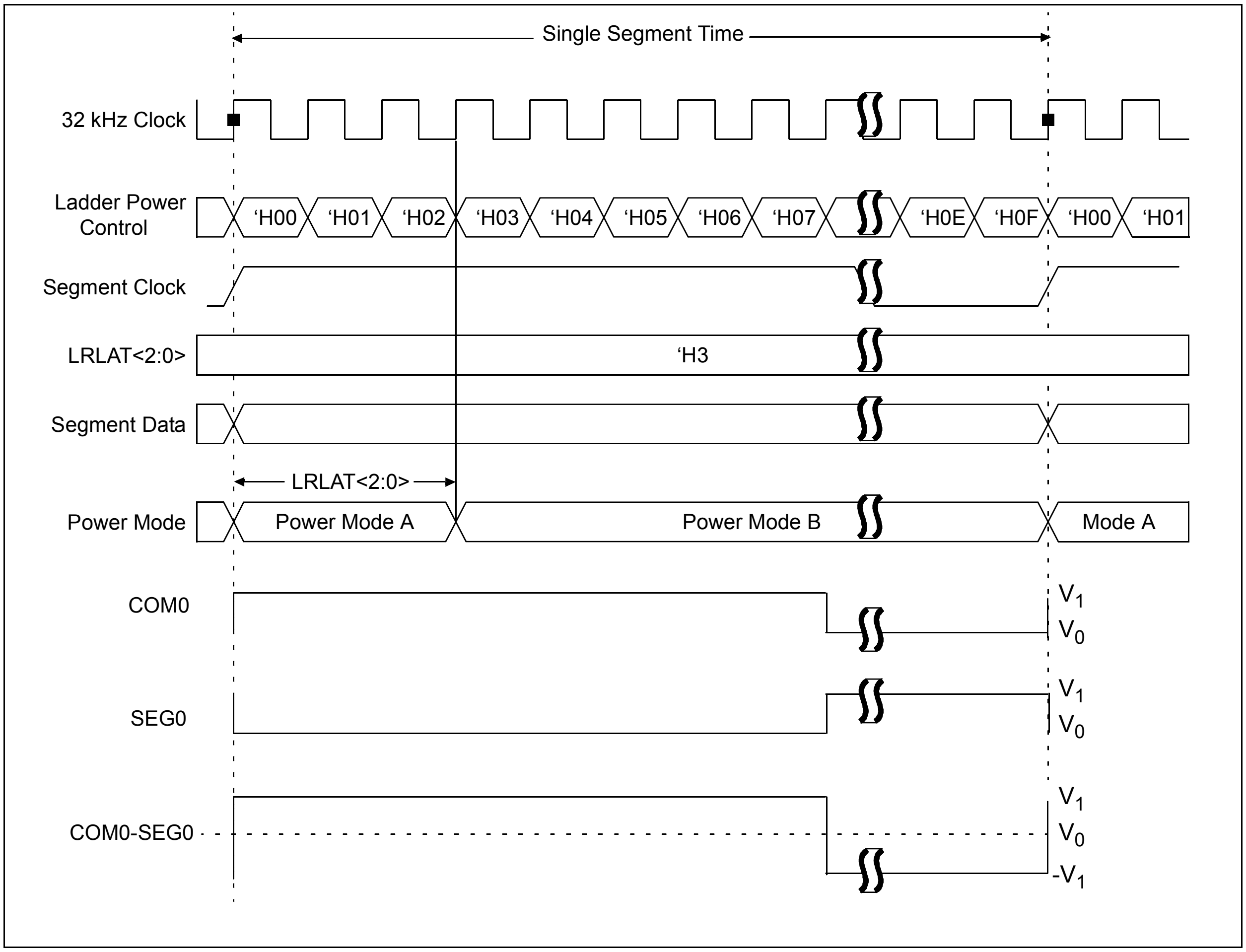

Single Segment Time
32 kHz Clock
Ladder Power Control
'H00
'H01
'H02
'H03
'H04
'H05
'H06
'H07
'H0E
'H0F
'H00
'H01
Segment Clock
LRLAT<2:0>
'H3
Segment Data
LRLAT<2:0>
Power Mode
Power Mode A
Power Mode B
Mode A
COM0
SEG0
COM0-SEG0
V1
V0
V1
V0
V1
V0
-V1

FIGURE 18-5: LCD INTERNAL REFERENCE LADDER POWER MODE SWITCHING DIAGRAM – TYPE A WAVEFORM (1/2 MUX, 1/2 BIAS DRIVE)

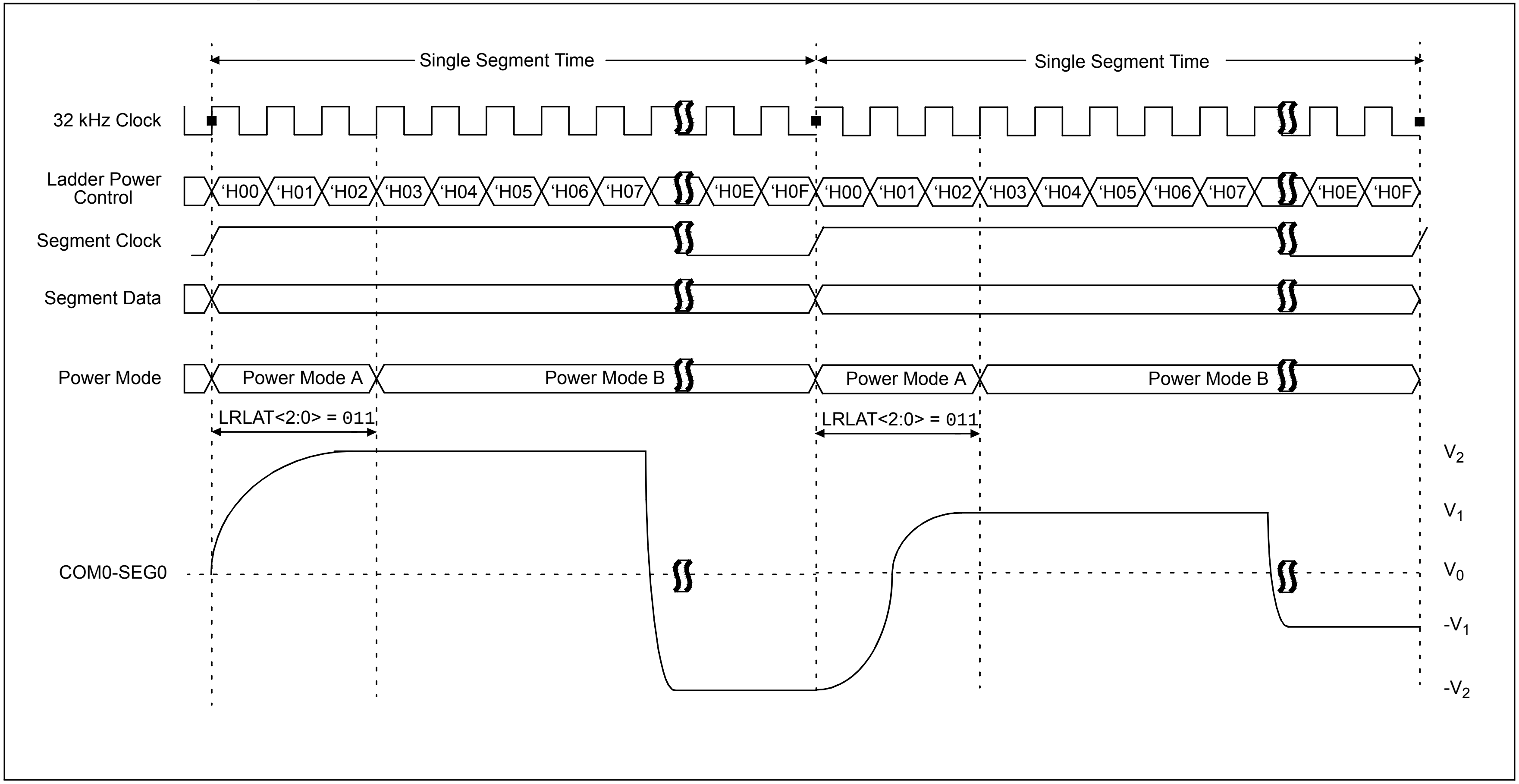
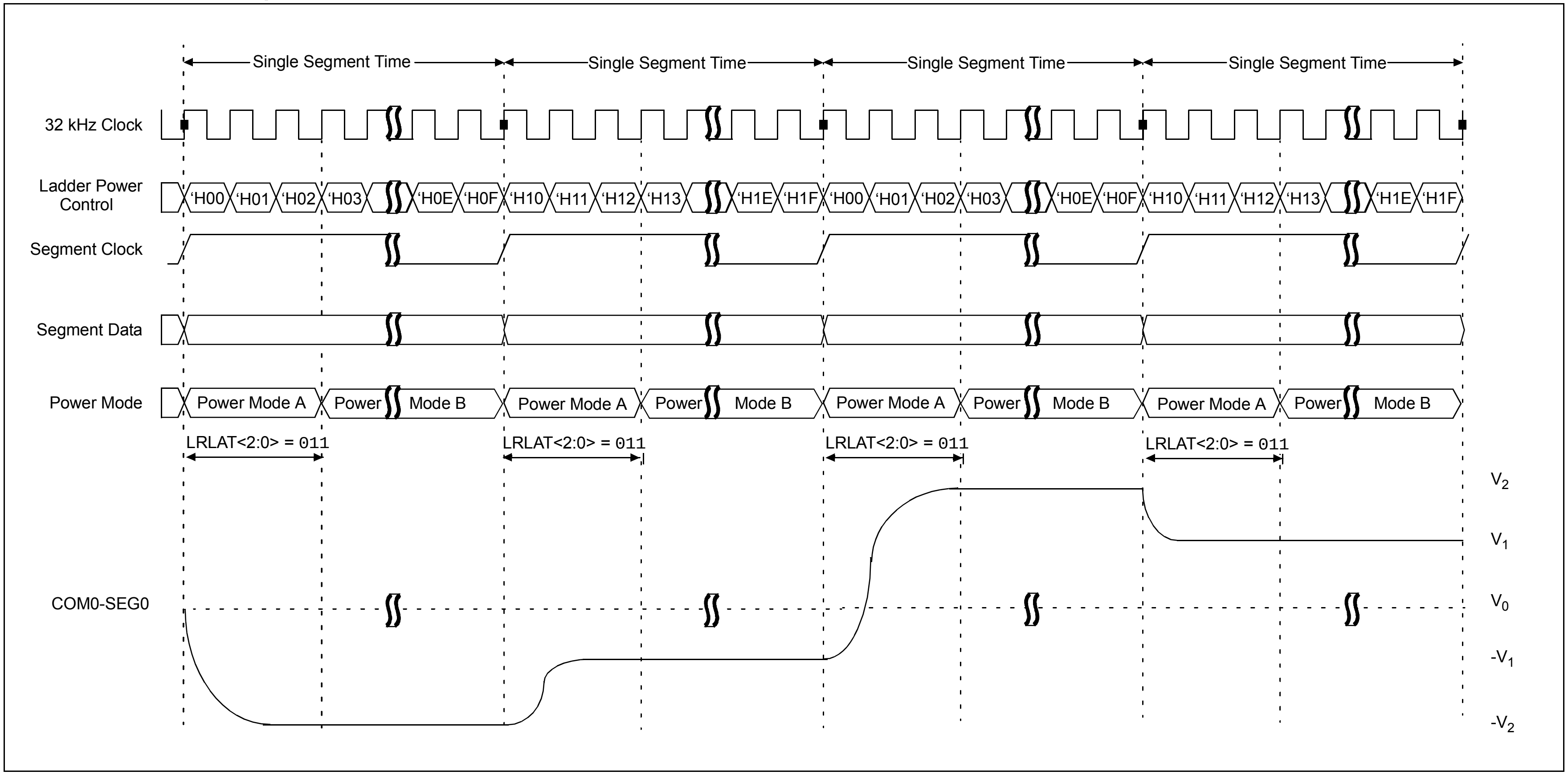

Single Segment Time
Single Segment Time
32 kHz Clock
Ladder Power Control
'H00
'H01
'H02
'H03
'H04
'H05
'H06
'H07
'H0E
'H0F
Segment Clock
Segment Data
Power Mode
Power Mode A
Power Mode B
LRLAT<2:0> = 011
COM0-SEG0
V_2
V_1
V_0
$-V_1$
$-V_2$

FIGURE 18-6: LCD INTERNAL REFERENCE LADDER POWER MODE SWITCHING DIAGRAM – TYPE B WAVEFORM (1/2 MUX, 1/2 BIAS DRIVE)


Single Segment Time
Single Segment Time
Single Segment Time
Single Segment Time
32 kHz Clock
Ladder Power Control
'H00
'H01
'H02
'H03
'H0E
'H0F
'H10
'H11
'H12
'H13
'H1E
'H1F
Segment Clock
Segment Data
Power Mode
Power Mode A
Power Mode B
LRLAT<2:0> = 011
COM0-SEG0
V_2
V_1
V_0
$-V_1$
$-V_2$

REGISTER 18-7: LCDRL: LCD REFERENCE LADDER CONTROL REGISTERS

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
LRLAP<1:0>		LRLBP<1:0>		—	LRLAT<2:0>		
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-6 **LRLAP<1:0>:** LCD Reference Ladder A Time Power Control bits
During Time interval A (Refer toFigure 18-4):
`00` = Internal LCD Reference Ladder is powered down and unconnected
`01` = Internal LCD Reference Ladder is powered in Low-Power mode
`10` = Internal LCD Reference Ladder is powered in Medium-Power mode
`11` = Internal LCD Reference Ladder is powered in High-Power mode

bit 5-4 **LRLBP<1:0>:** LCD Reference Ladder B Time Power Control bits
During Time interval B (Refer to Figure 18-4):
`00` = Internal LCD Reference Ladder is powered down and unconnected
`01` = Internal LCD Reference Ladder is powered in Low-Power mode
`10` = Internal LCD Reference Ladder is powered in Medium-Power mode
`11` = Internal LCD Reference Ladder is powered in High-Power mode

bit 3 **Unimplemented:** Read as ‘`0`’

bit 2-0 **LRLAT<2:0>:** LCD Reference Ladder A Time Interval Control bits
Sets the number of 32 kHz clocks that the A Time Interval Power mode is active

For type A waveforms (WFT = `0`):

`000` = Internal LCD Reference Ladder is always in ‘B’ Power mode
`001` = Internal LCD Reference Ladder is in ‘A’ Power mode for 1 clock and ‘B’ Power mode for 15 clocks
`010` = Internal LCD Reference Ladder is in ‘A’ Power mode for 2 clocks and ‘B’ Power mode for 14 clocks
`011` = Internal LCD Reference Ladder is in ‘A’ Power mode for 3 clocks and ‘B’ Power mode for 13 clocks
`100` = Internal LCD Reference Ladder is in ‘A’ Power mode for 4 clocks and ‘B’ Power mode for 12 clocks
`101` = Internal LCD Reference Ladder is in ‘A’ Power mode for 5 clocks and ‘B’ Power mode for 11 clocks
`110` = Internal LCD Reference Ladder is in ‘A’ Power mode for 6 clocks and ‘B’ Power mode for 10 clocks
`111` = Internal LCD Reference Ladder is in ‘A’ Power mode for 7 clocks and ‘B’ Power mode for 9 clocks

For type B waveforms (WFT = `1`):

`000` = Internal LCD Reference Ladder is always in ‘B’ Power mode.
`001` = Internal LCD Reference Ladder is in ‘A’ Power mode for 1 clock and ‘B’ Power mode for 31 clocks
`010` = Internal LCD Reference Ladder is in ‘A’ Power mode for 2 clocks and ‘B’ Power mode for 30 clocks
`011` = Internal LCD Reference Ladder is in ‘A’ Power mode for 3 clocks and ‘B’ Power mode for 29 clocks
`100` = Internal LCD Reference Ladder is in ‘A’ Power mode for 4 clocks and ‘B’ Power mode for 28 clocks
`101` = Internal LCD Reference Ladder is in ‘A’ Power mode for 5 clocks and ‘B’ Power mode for 27 clocks
`110` = Internal LCD Reference Ladder is in ‘A’ Power mode for 6 clocks and ‘B’ Power mode for 26 clocks
`111` = Internal LCD Reference Ladder is in ‘A’ Power mode for 7 clocks and ‘B’ Power mode for 25 clocks

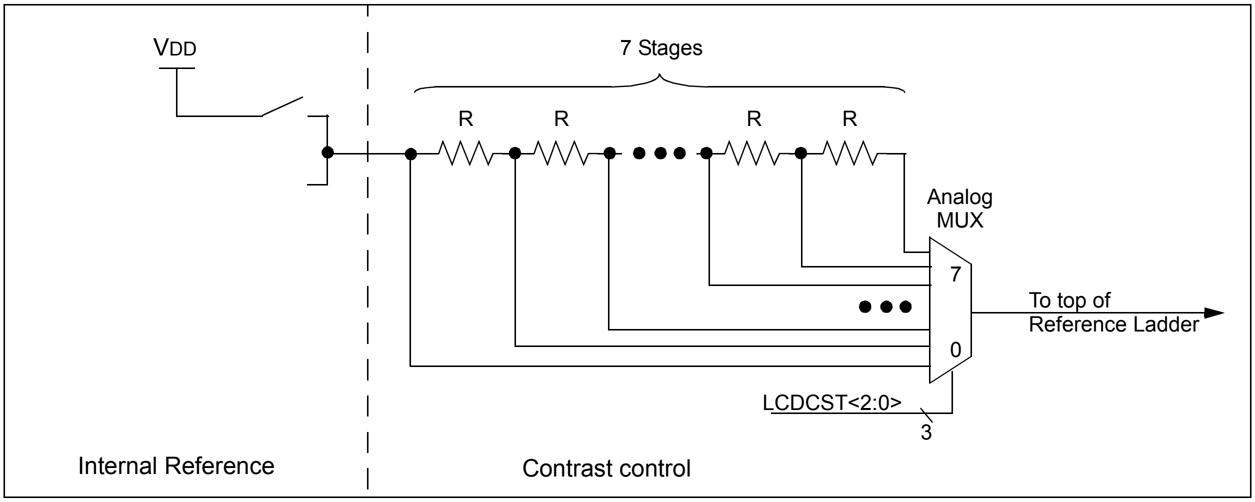
18.4.4 CONTRAST CONTROL

The LCD contrast control circuit consists of a 7-tap resistor ladder, controlled by the LCDCST bits. Refer to Figure 18-7.

The contrast control circuit is used to decrease the output voltage of the signal source by a total of approximately 10%, when LCDCST = `111`.

Whenever the LCD module is inactive (LCDA = `0`), the contrast control ladder will be turned off (open).

FIGURE 18-7: INTERNAL REFERENCE AND CONTRAST CONTROL BLOCK DIAGRAM



18.4.5 INTERNAL REFERENCE

Under firmware control, an internal reference for the LCD bias voltages can be enabled. When enabled, the source of this voltage can be VDD. When no internal reference is selected, the LCD contrast control circuit is disabled and LCD bias must be provided externally.

Whenever the LCD module is inactive (LCDA = `0`), the internal reference will be turned off.

When the internal reference is enabled and the Fixed Voltage Reference is selected, the LCDIRI bit can be used to minimize power consumption by tying into the LCD reference ladder automatic power mode switching. When LCDIRI = `1` and the LCD reference ladder is in Power mode 'B', the LCD internal FVR buffer is disables.

Note: The LCD module automatically turns on the Fixed Voltage Reference when needed.

18.4.6 VLCD<3:1> PINS

The VLCD<3:1> pins provide the ability for an external LCD bias network to be used instead of the internal ladder. Use of the VLCD<3:1> pins does not prevent use of the internal ladder. Each VLCD pin has an independent control in the LCDREF register (Register 18-3), allowing access to any or all of the LCD Bias signals. This architecture allows for maximum flexibility in different applications

For example, the VLCD<3:1> pins may be used to add capacitors to the internal reference ladder, increasing the drive capacity.

For applications where the internal contrast control is insufficient, the firmware can choose to only enable the VLCD3 pin, allowing an external contrast control circuit to use the internal reference divider.

18.5 LCD Multiplex Types

The LCD driver module can be configured into one of four multiplex types:

- Static (only COM0 is used)
- 1/2 multiplex (COM<1:0> are used)
- 1/3 multiplex (COM<2:0> are used)
- 1/4 multiplex (COM<3:0> are used)

The LMUX<1:0> bit setting of the LCDCON register decides which of the LCD common pins are used (see Table 18-4 for details).

If the pin is a digital I/O, the corresponding TRIS bit controls the data direction. If the pin is a COM drive, then the TRIS setting of that pin is overridden.

TABLE 18-4: COMMON PIN USAGE

Multiplex	LMUX <1:0>	COM3	COM2	COM1	COM1
Static	`00`	Unused	Unused	Unused	Active
1/2	`01`	Unused	Unused	Active	Active
1/3	`10`	Unused	Active	Active	Active
1/4	`11`	Active	Active	Active	Active

18.6 Segment Enables

The LCDSEn registers are used to select the pin function for each segment pin. The selection allows each pin to operate as either an LCD segment driver or as one of the pin’s alternate functions. To configure the pin as a segment pin, the corresponding bits in the LCDSEn registers must be set to ‘`1`’.

If the pin is a digital I/O, the corresponding TRIS bit controls the data direction. Any bit set in the LCDSEn registers overrides any bit settings in the corresponding TRIS register.

Note: On a Power-on Reset, these pins are configured as normal I/O, not LCD pins.

18.7 Pixel Control

The LCDDATAx registers contain bits which define the state of each pixel. Each bit defines one unique pixel.

Register 18-6 shows the correlation of each bit in the LCDDATAx registers to the respective common and segment signals.

Any LCD pixel location not being used for display can be used as general purpose RAM.

18.8 LCD Frame Frequency

The rate at which the COM and SEG outputs change is called the LCD frame frequency.

TABLE 18-5: FRAME FREQUENCY FORMULAS

Multiplex	Frame Frequency[2] =
Static	Clock source[1]/(4 x (LCD Prescaler) x 32 x 1))
1/2	Clock source[1]/(2 x (LCD Prescaler) x 32 x 2))
1/3	Clock source[1]/(1 x (LCD Prescaler) x 32 x 3))
1/4	Clock source[1]/(1 x (LCD Prescaler) x 32 x 4))

Note 1: Clock source is FOSC/256, T1OSC or LFINTOSC.

2: See Figure 18-2.

TABLE 18-6: APPROXIMATE FRAME FREQUENCY (IN Hz) USING FOSC @ 8 MHz, TIMER1 @ 32.768 kHz OR LFINTOSC

LP<3:0>	Static	1/2	1/3	1/4
2	122	122	162	122
3	81	81	108	81
4	61	61	81	61
5	49	49	65	49
6	41	41	54	41
7	35	35	47	35

18.9 LCD Waveform Generation

LCD waveforms are generated so that the net AC voltage across the dark pixel should be maximized and the net AC voltage across the clear pixel should be minimized. The net DC voltage across any pixel should be zero.

The COM signal represents the time slice for each common, while the SEG contains the pixel data.

The pixel signal (COM-SEG) will have no DC component and it can take only one of the two RMS values. The higher RMS value will create a dark pixel and a lower RMS value will create a clear pixel.

As the number of commons increases, the delta between the two RMS values decreases. The delta represents the maximum contrast that the display can have.

The LCDs can be driven by two types of waveform: Type-A and Type-B. In Type-A waveform, the phase changes within each common type, whereas in Type-B waveform, the phase changes on each frame boundary. Thus, Type-A waveform maintains 0 VDC over a single frame, whereas Type-B waveform takes two frames.

Note 1: If Sleep has to be executed with LCD Sleep disabled (LCDCON<SLPEN> is `‘1’`), then care must be taken to execute Sleep only when VDC on all the pixels is `‘0’`.

2: When the LCD clock source is FOSC/256, if Sleep is executed, irrespective of the LCDCON<SLPEN> setting, the LCD immediately goes into Sleep. Thus, take care to see that VDC on all pixels is `‘0’` when Sleep is executed.

Figure 18-8 through Figure 18-18 provide waveforms for static, half-multiplex, 1/3-multiplex and 1/4-multiplex drives for Type-A and Type-B waveforms.

FIGURE 18-8: TYPE-A/TYPE-B WAVEFORMS IN STATIC DRIVE

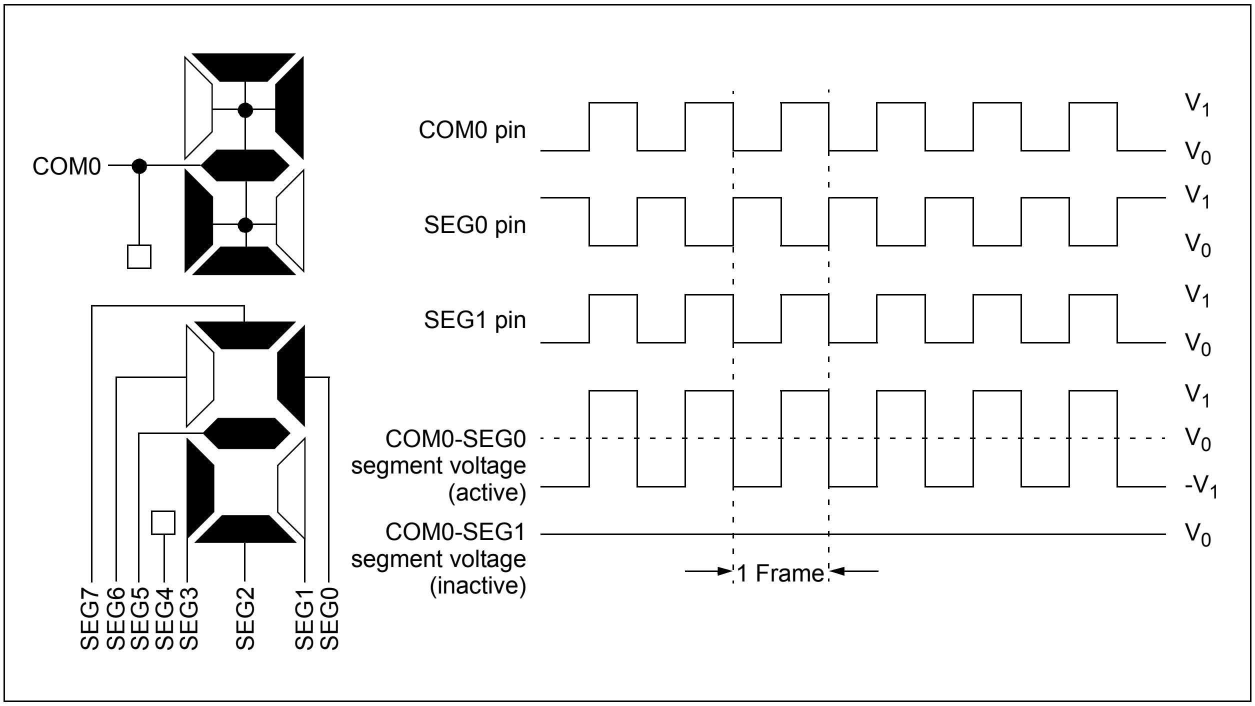
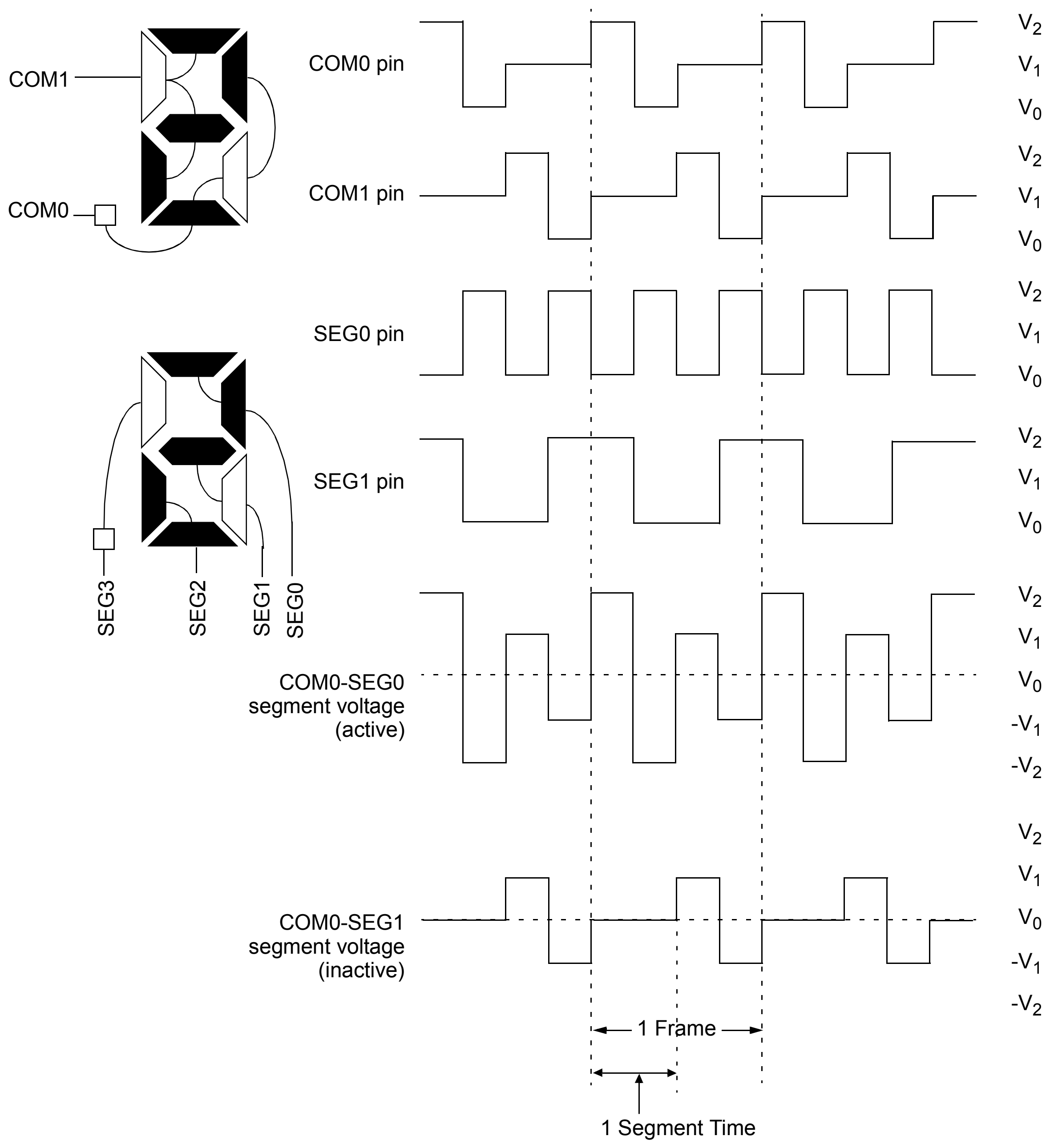
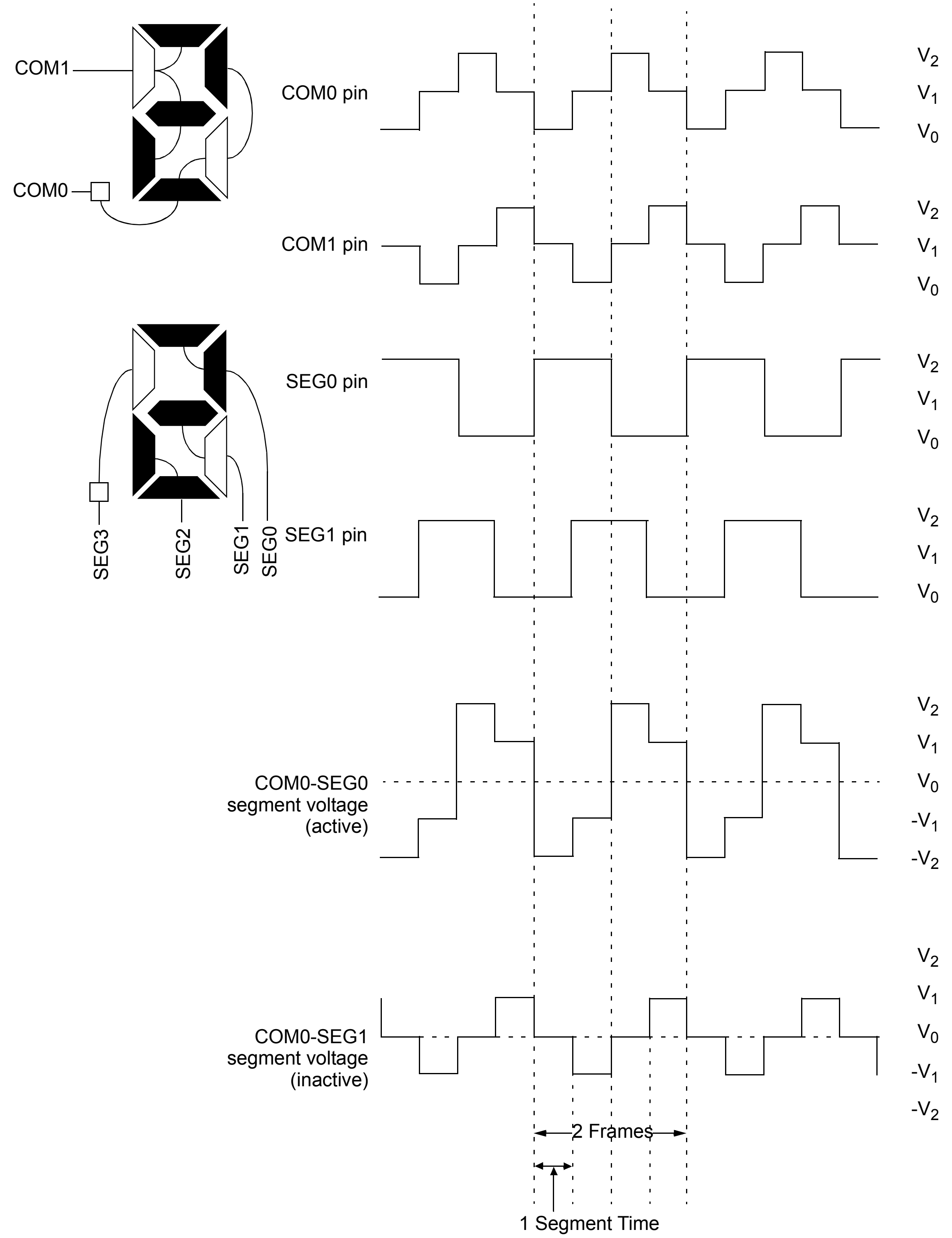


FIGURE 18-9: TYPE-A WAVEFORMS IN 1/2 MUX, 1/2 BIAS DRIVE


COM1
COM0
SEG3
SEG2
SEG1
SEG0
COM0 pin
COM1 pin
SEG0 pin
SEG1 pin
COM0-SEG0 segment voltage (active)
COM0-SEG1 segment voltage (inactive)
V_2
V_1
V_0
$-V_1$
$-V_2$
1 Frame
1 Segment Time

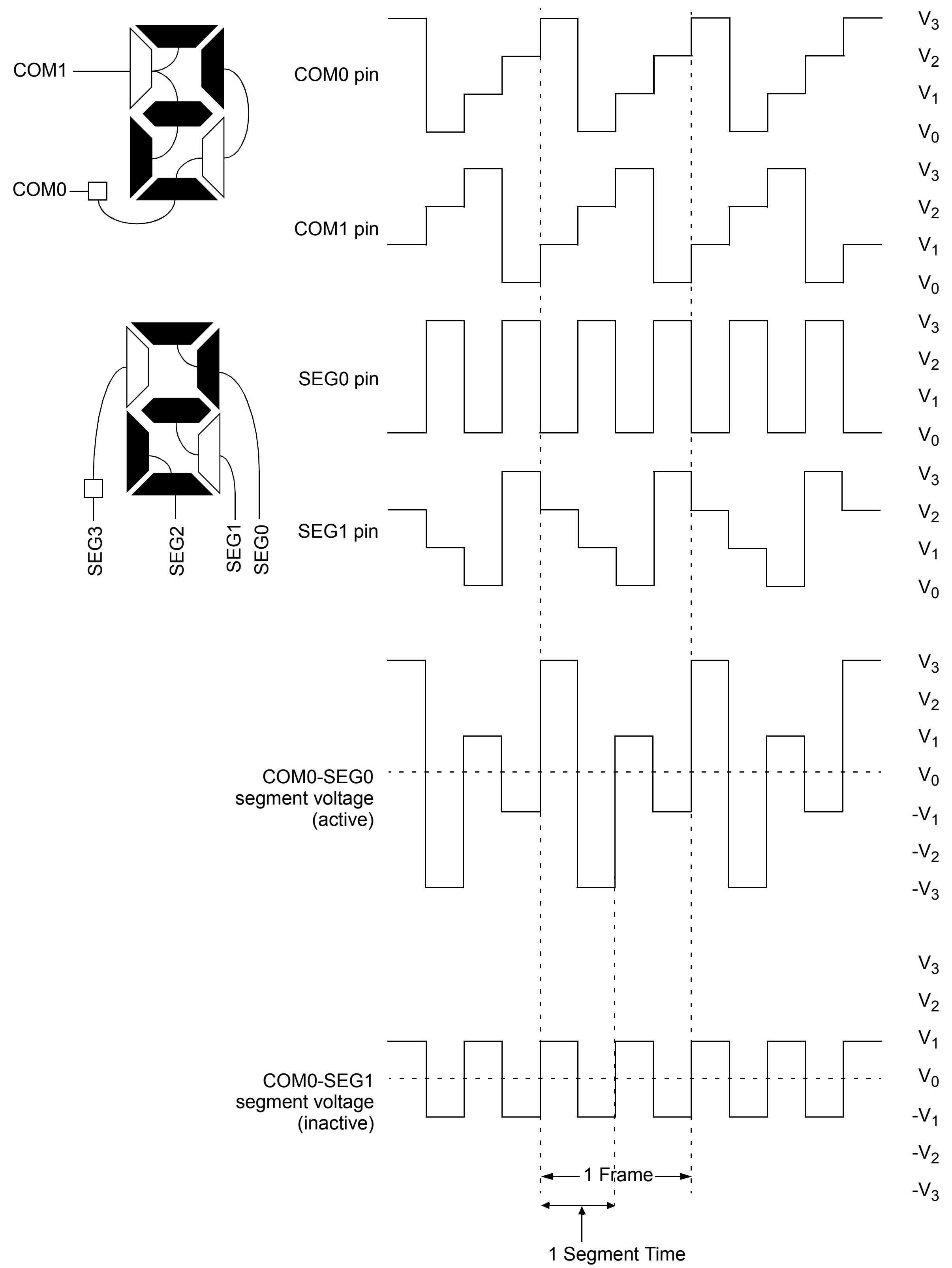
Note: 1 Frame = 2 single segment times.

FIGURE 18-10: TYPE-B WAVEFORMS IN 1/2 MUX, 1/2 BIAS DRIVE


COM1
COM0
COM0 pin
COM1 pin
SEG0 pin
SEG1 pin
SEG3
SEG2
SEG1
SEG0
COM0-SEG0
segment voltage
(active)
COM0-SEG1
segment voltage
(inactive)
V2
V1
V0
-V1
-V2
2 Frames
1 Segment Time

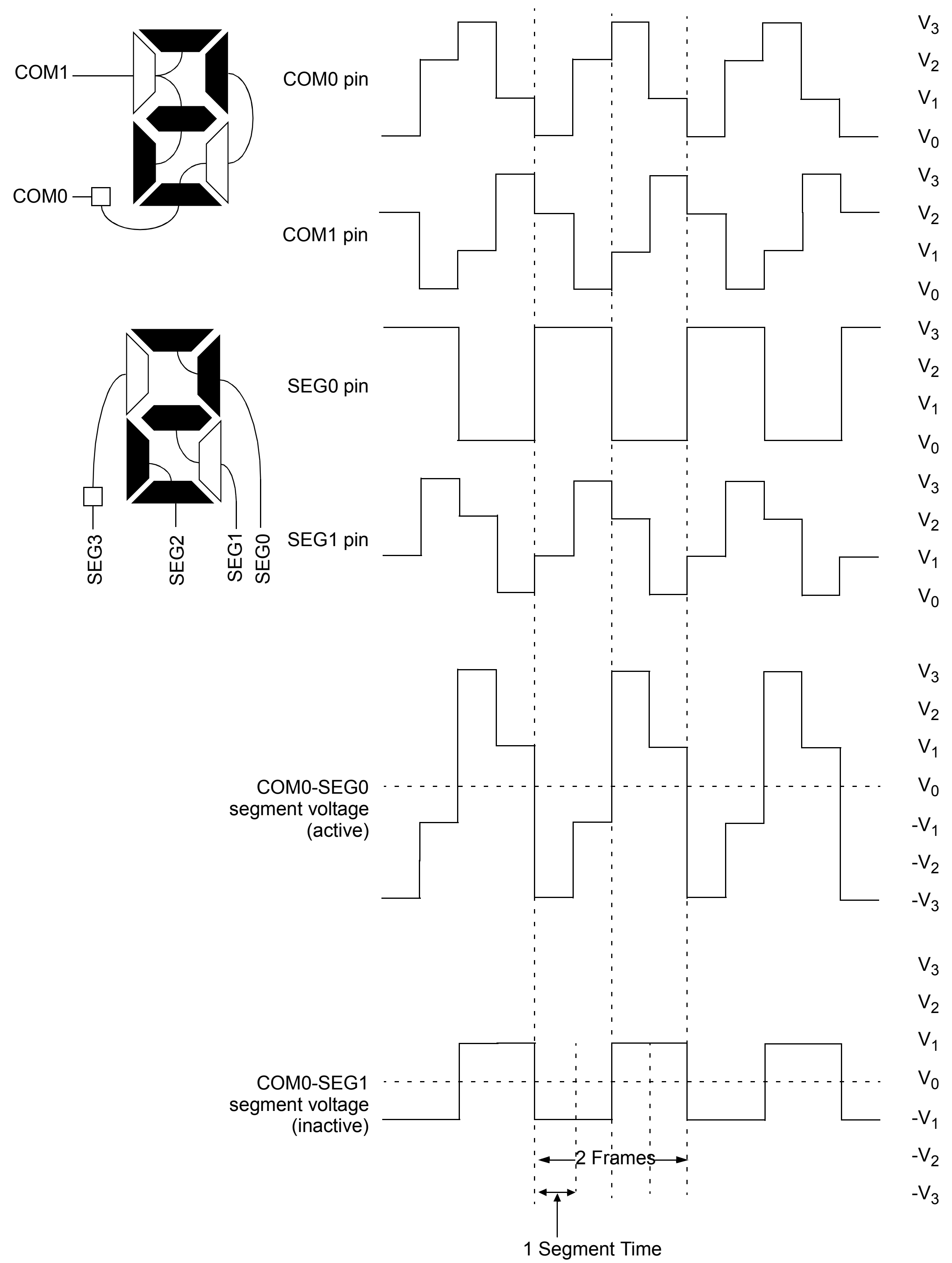
Note: 1 Frame = 2 single segment times.

FIGURE 18-11: TYPE-A WAVEFORMS IN 1/2 MUX, 1/3 BIAS DRIVE


COM1
COM0
COM0 pin
COM1 pin
SEG0 pin
SEG1 pin
SEG3
SEG2
SEG1
SEG0
COM0-SEG0 segment voltage (active)
COM0-SEG1 segment voltage (inactive)
V3
V2
V1
V0
-V1
-V2
-V3
1 Frame
1 Segment Time

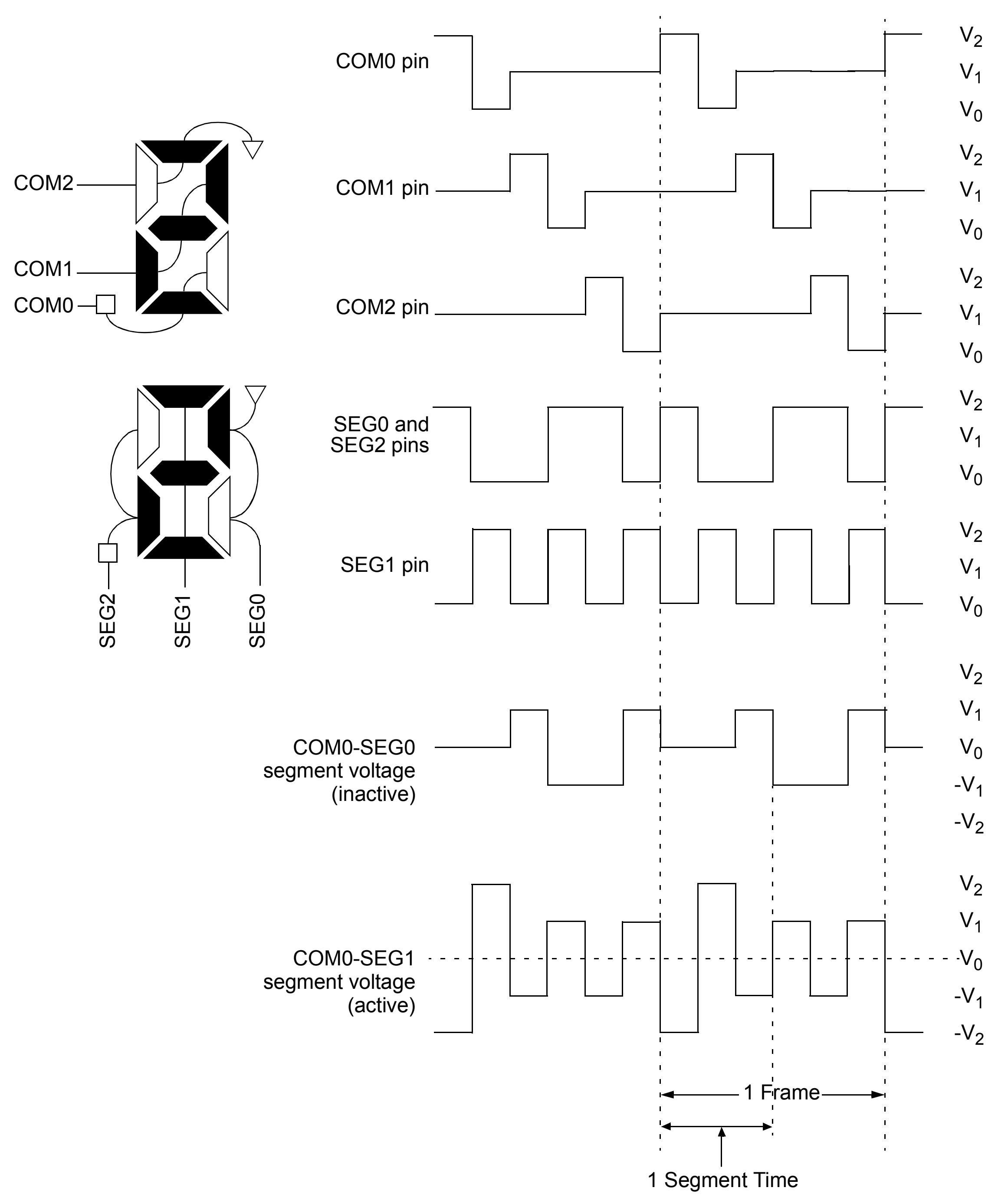
Note: 1 Frame = 2 single segment times.

FIGURE 18-12: TYPE-B WAVEFORMS IN 1/2 MUX, 1/3 BIAS DRIVE


COM1
COM0
COM0 pin
COM1 pin
SEG0 pin
SEG1 pin
SEG3
SEG2
SEG1
SEG0
V3
V2
V1
V0
COM0-SEG0 segment voltage (active)
-V1
-V2
-V3
COM0-SEG1 segment voltage (inactive)
2 Frames
1 Segment Time

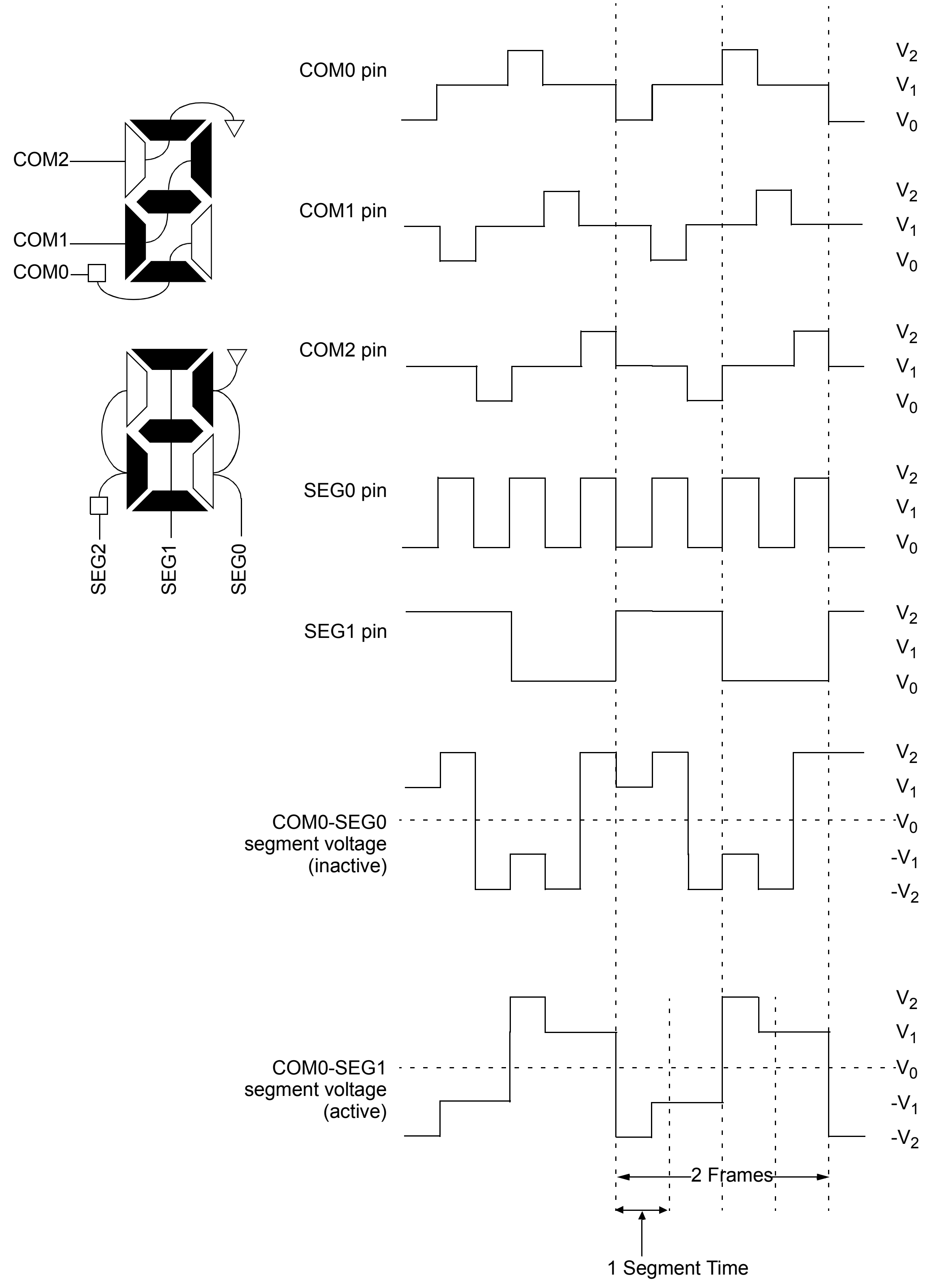
Note: 1 Frame = 2 single segment times.

FIGURE 18-13: TYPE-A WAVEFORMS IN 1/3 MUX, 1/2 BIAS DRIVE


COM2
COM1
COM0
SEG2
SEG1
SEG0
COM0 pin
COM1 pin
COM2 pin
SEG0 and SEG2 pins
SEG1 pin
COM0-SEG0 segment voltage (inactive)
COM0-SEG1 segment voltage (active)
V2
V1
V0
-V1
-V2
1 Frame
1 Segment Time

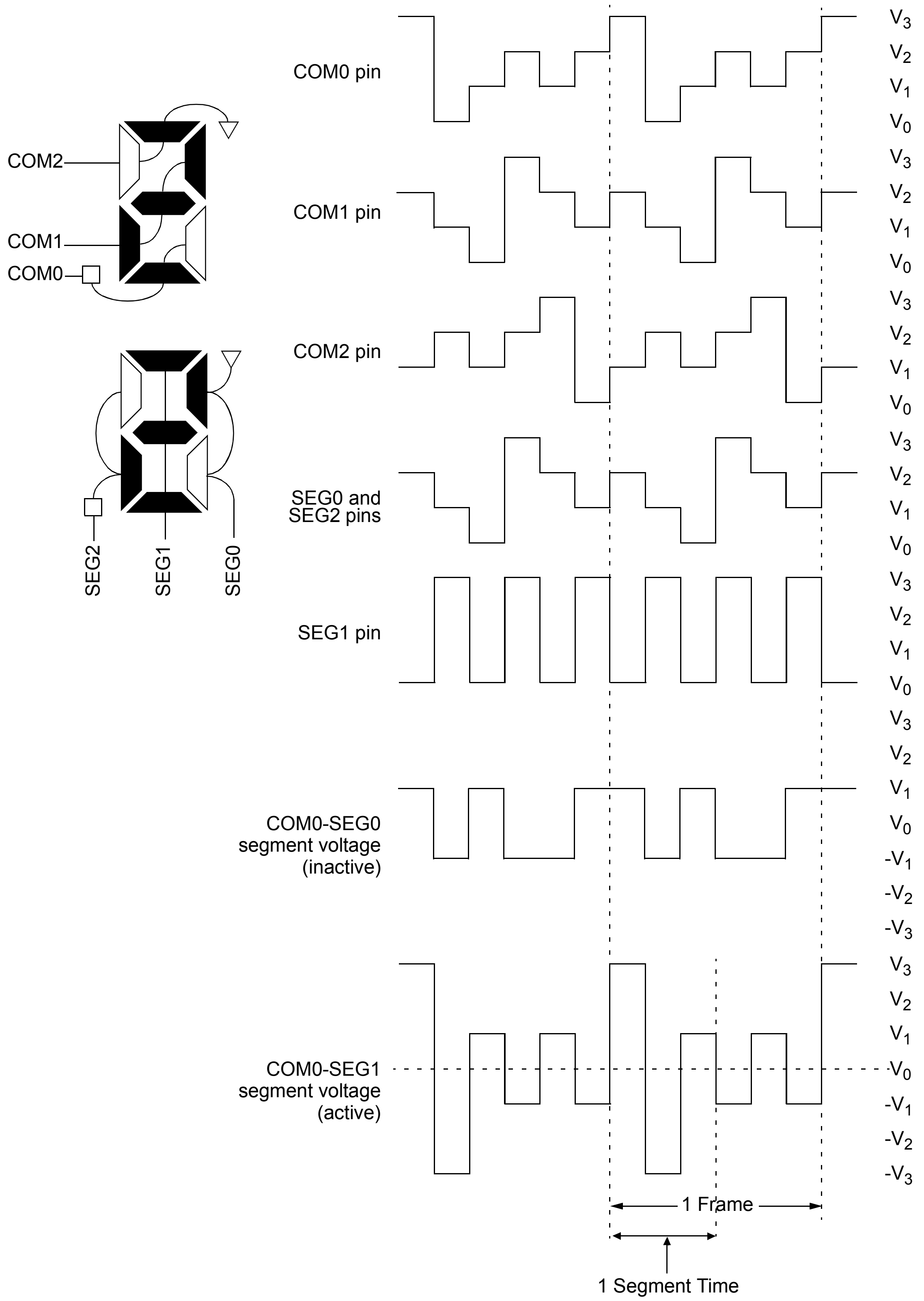
Note: 1 Frame = 2 single segment times.

FIGURE 18-14: TYPE-B WAVEFORMS IN 1/3 MUX, 1/2 BIAS DRIVE


COM2
COM1
COM0
SEG2
SEG1
SEG0
COM0 pin
COM1 pin
COM2 pin
SEG0 pin
SEG1 pin
COM0-SEG0 segment voltage (inactive)
COM0-SEG1 segment voltage (active)
V_2
V_1
V_0
$-V_1$
$-V_2$
2 Frames
1 Segment Time

Note: 1 Frame = 2 single segment times.

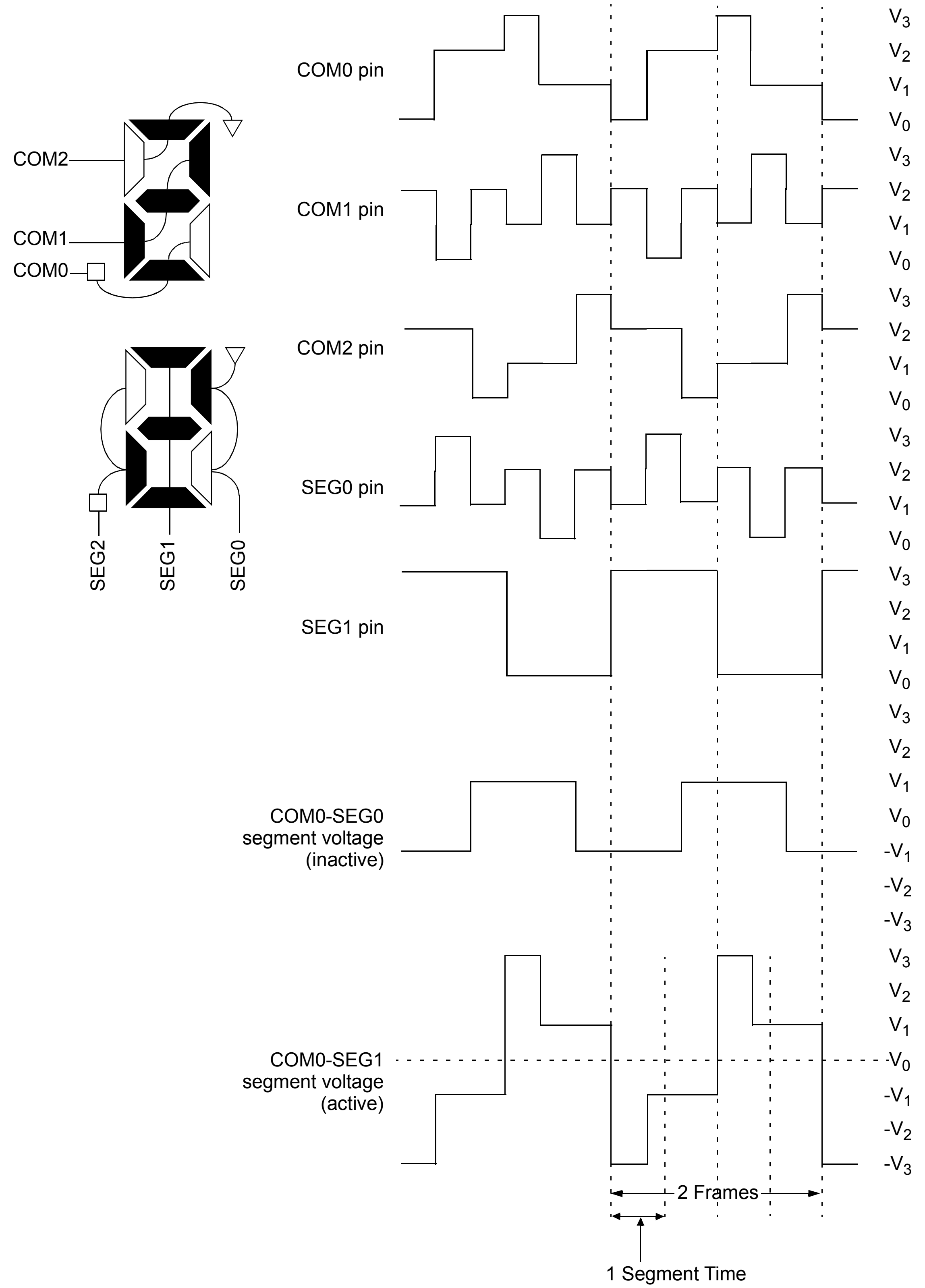
FIGURE 18-15: TYPE-A WAVEFORMS IN 1/3 MUX, 1/3 BIAS DRIVE



COM0 pin
COM1 pin
COM2 pin
SEG0 and SEG2 pins
SEG1 pin
COM0-SEG0 segment voltage (inactive)
COM0-SEG1 segment voltage (active)
COM2
COM1
COM0
SEG2
SEG1
SEG0
V3
V2
V1
V0
-V1
-V2
-V3
1 Frame
1 Segment Time

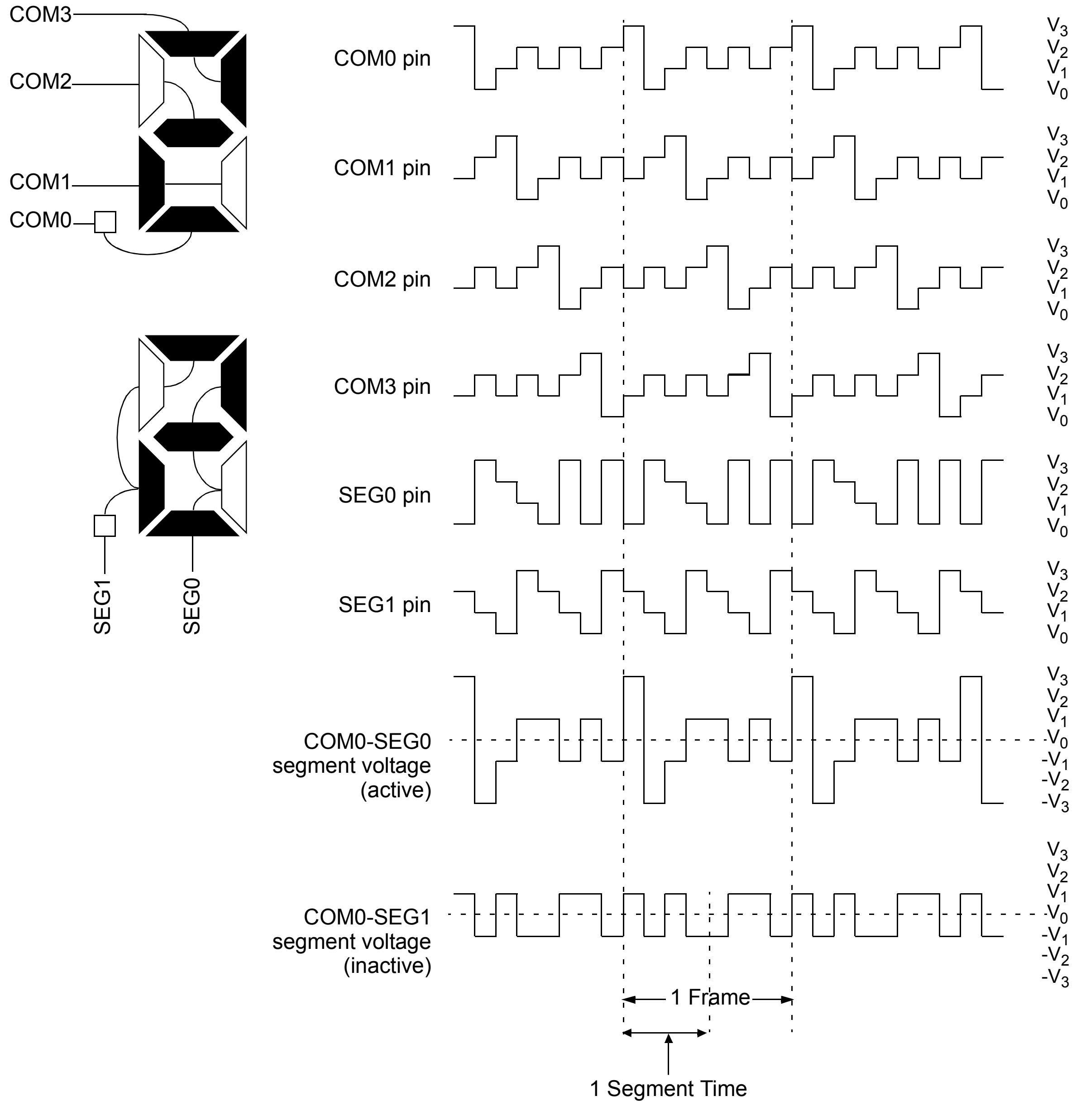
Note: 1 Frame = 2 single segment times.

FIGURE 18-16: TYPE-B WAVEFORMS IN 1/3 MUX, 1/3 BIAS DRIVE


COM0 pin
COM1 pin
COM2 pin
SEG0 pin
SEG1 pin
COM0-SEG0 segment voltage (inactive)
COM0-SEG1 segment voltage (active)
COM2
COM1
COM0
SEG2
SEG1
SEG0
V_3
V_2
V_1
V_0
$-V_1$
$-V_2$
$-V_3$
2 Frames
1 Segment Time

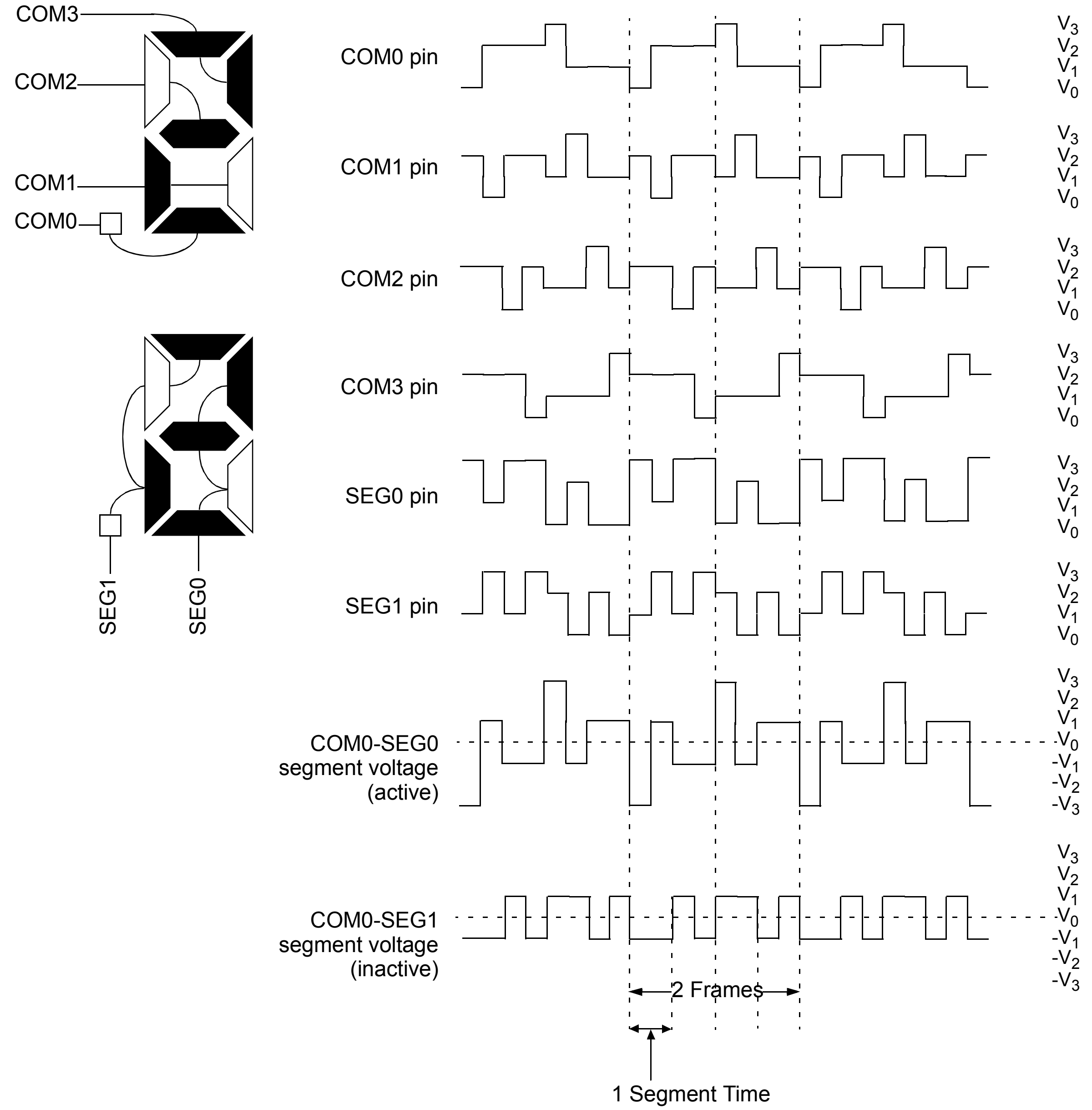
Note: 1 Frame = 2 single segment times.

FIGURE 18-17: TYPE-A WAVEFORMS IN 1/4 MUX, 1/3 BIAS DRIVE


COM3
COM2
COM1
COM0
SEG1
SEG0
COM0 pin
COM1 pin
COM2 pin
COM3 pin
SEG0 pin
SEG1 pin
COM0-SEG0 segment voltage (active)
COM0-SEG1 segment voltage (inactive)
V_3
V_2
V_1
V_0
$-V_1$
$-V_2$
$-V_3$
1 Frame
1 Segment Time

Note: 1 Frame = 2 single segment times.

FIGURE 18-18: TYPE-B WAVEFORMS IN 1/4 MUX, 1/3 BIAS DRIVE


COM3
COM2
COM1
COM0
SEG1
SEG0
COM0 pin
COM1 pin
COM2 pin
COM3 pin
SEG0 pin
SEG1 pin
COM0-SEG0 segment voltage (active)
COM0-SEG1 segment voltage (inactive)
V3
V2
V1
V0
-V1
-V2
-V3
2 Frames
1 Segment Time

Note: 1 Frame = 2 single segment times.

18.10 LCD Interrupts

The LCD module provides an interrupt in two cases. An interrupt when the LCD controller goes from active to inactive controller. An interrupt also provides unframe boundaries for Type B waveform. The LCD timing generation provides an interrupt that defines the LCD frame timing.

18.10.1 LCD INTERRUPT ON MODULE SHUTDOWN

An LCD interrupt is generated when the module completes shutting down (LCDA goes from ‘`1`’ to ‘`0`’).

18.10.2 LCD FRAME INTERRUPTS

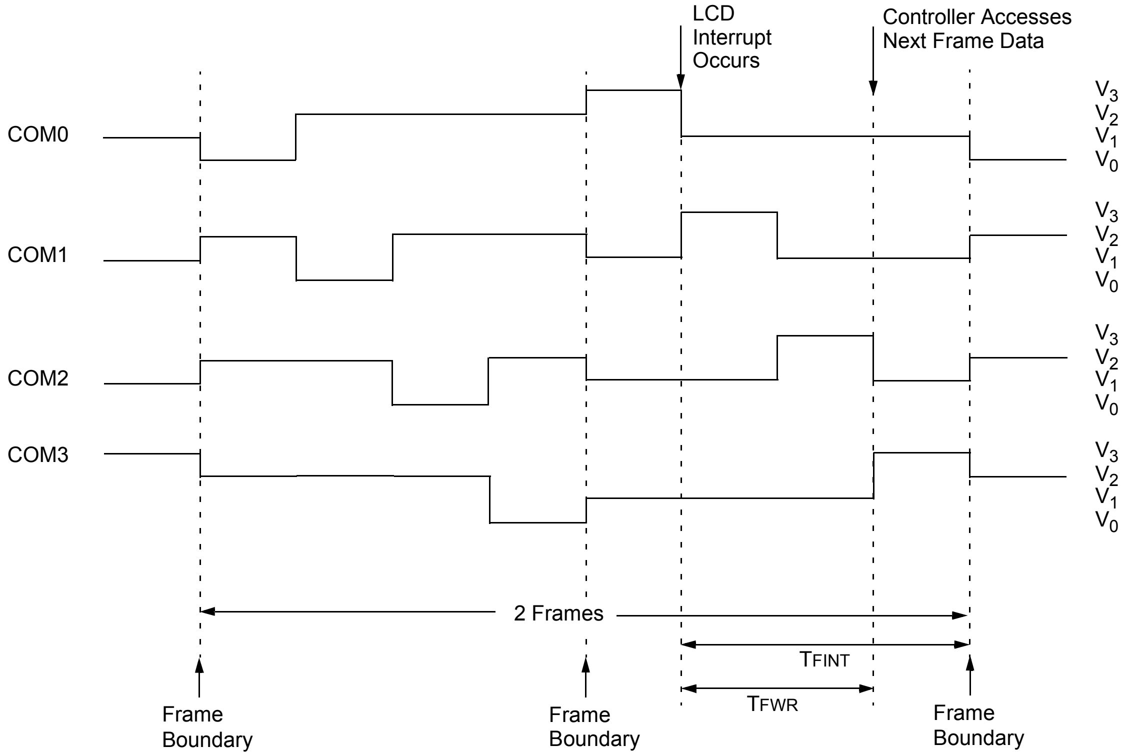
A new frame is defined to begin at the leading edge of the COM0 common signal. The interrupt will be set immediately after the LCD controller completes accessing all pixel data required for a frame. This will occur at a fixed interval before the frame boundary (TFINT), as shown in Figure 18-19. The LCD controller will begin to access data for the next frame within the interval from the interrupt to when the controller begins to access data after the interrupt (TFWR). New data must be written within TFWR, as this is when the LCD controller will begin to access the data for the next frame.

When the LCD driver is running with Type-B waveforms and the LMUX<1:0> bits are not equal to ‘`00`’ (static drive), there are some additional issues that must be addressed. Since the DC voltage on the pixel takes two frames to maintain zero volts, the pixel data must not change between subsequent frames. If the pixel data were allowed to change, the waveform for the odd frames would not necessarily be the complement of the waveform generated in the even frames and a DC component would be introduced into the panel. Therefore, when using Type-B waveforms, the user must synchronize the LCD pixel updates to occur within a subframe after the frame interrupt.

To correctly sequence writing while in Type-B, the interrupt will only occur on complete phase intervals. If the user attempts to write when the write is disabled, the WERR bit of the LCDCON register is set and the write does not occur.

Note:	The LCD frame interrupt is not generated when the Type-A waveform is selected and when the Type-B with no multiplex (static) is selected.

FIGURE 18-19: WAVEFORMS AND INTERRUPT TIMING IN QUARTER-DUTY CYCLE DRIVE (EXAMPLE – TYPE-B, NON-STATIC)


LCD Interrupt Occurs
Controller Accesses Next Frame Data
COM0
COM1
COM2
COM3
V_3
V_2
V_1
V_0
2 Frames
TFINT
TFWR
Frame Boundary
Frame Boundary
Frame Boundary

$T_{FWR} = T_{FRAME}/2*(LMUX<1:0> + 1) + T_{CY}/2$

$T_{FINT} = (T_{FWR}/2 - (2\ T_{CY} + 40\text{ ns})) \rightarrow \text{minimum} = 1.5(T_{FRAME}/4) - (2\ T_{CY} + 40\text{ ns})$

$(T_{FWR}/2 - (1\ T_{CY} + 40\text{ ns})) \rightarrow \text{maximum} = 1.5(T_{FRAME}/4) - (1\ T_{CY} + 40\text{ ns})$

18.11 Operation During Sleep

The LCD module can operate during Sleep. The selection is controlled by bit SLPEN of the LCDCON register. Setting the SLPEN bit allows the LCD module to go to Sleep. Clearing the SLPEN bit allows the module to continue to operate during Sleep.

If a `SLEEP` instruction is executed and SLPEN = `1`, the LCD module will cease all functions and go into a very low-current Consumption mode. The module will stop operation immediately and drive the minimum LCD voltage on both segment and common lines. Figure 18-20 shows this operation.

The LCD module can be configured to operate during Sleep. The selection is controlled by bit SLPEN of the LCDCON register. Clearing SLPEN and correctly configuring the LCD module clock will allow the LCD module to operate during Sleep. Setting SLPEN and correctly executing the LCD module shutdown will disable the LCD module during Sleep and save power.

If a `SLEEP` instruction is executed and SLPEN = `1`, the LCD module will immediately cease all functions, drive the outputs to Vss and go into a very low-current mode. The `SLEEP` instruction should only be executed after the LCD module has been disabled and the current cycle completed, thus ensuring that there are no DC voltages on the glass. To disable the LCD module, clear the LCDEN bit. The LCD module will complete the disabling process after the current frame, clear the LCDA bit and optionally cause an interrupt.

The steps required to properly enter Sleep with the LCD disabled are:

- Clear LCDEN
- Wait for LCDA = `0` either by polling or by interrupt
- Execute `SLEEP`

If SLPEN = `0` and `SLEEP` is executed while the LCD module clock source is FOSC/4, then the LCD module will halt with the pin driving the last LCD voltage pattern. Prolonged exposure to a fixed LCD voltage pattern will cause damage to the LCD glass. To prevent LCD glass damage, either perform the proper LCD module shutdown prior to Sleep, or change the LCD module clock to allow the LCD module to continue operation during Sleep.

If a `SLEEP` instruction is executed and SLPEN = `0` and the LCD module clock is either T1OSC or LFINTOSC, the module will continue to display the current contents of the LCDDATA registers. While in Sleep, the LCD data cannot be changed. If the LCDIE bit is set, the device will wake from Sleep on the next LCD frame boundary. The LCD module current consumption will not decrease in this mode; however, the overall device power consumption will be lower due to the shutdown of the CPU and other peripherals.

Table 18-7 shows the status of the LCD module during a Sleep while using each of the three available clock sources.

Note: When the LCDEN bit is cleared, the LCD module will be disabled at the completion of frame. At this time, the port pins will revert to digital functionality. To minimize power consumption due to floating digital inputs, the LCD pins should be driven low using the PORT and TRIS registers.

If a `SLEEP` instruction is executed and SLPEN = `0`, the module will continue to display the current contents of the LCDDATA registers. To allow the module to continue operation while in Sleep, the clock source must be either the LFINTOSC or T1OSC external oscillator. While in Sleep, the LCD data cannot be changed. The LCD module current consumption will not decrease in this mode; however, the overall consumption of the device will be lower due to shut down of the core and other peripheral functions.

Table 18-7 shows the status of the LCD module during Sleep while using each of the three available clock sources:

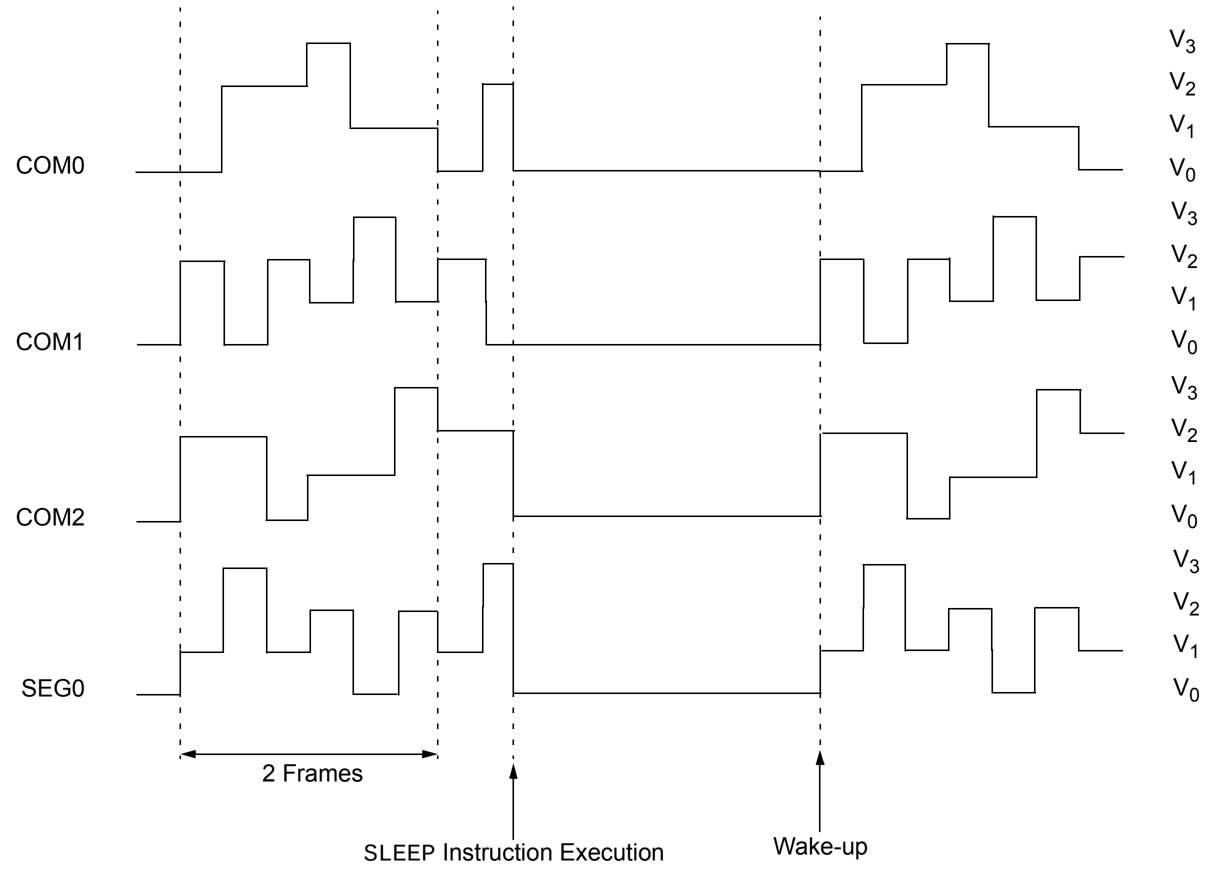
TABLE 18-7: LCD MODULE STATUS DURING SLEEP

Clock Source	SLPEN	Operational During Sleep
T1OSC	`0`	Yes
	`1`	No
LFINTOSC	`0`	Yes
	`1`	No
FOSC/4	`0`	No
	`1`	No

Note: The LFINTOSC or external T1OSC oscillator must be used to operate the LCD module during Sleep.

If LCD interrupts are being generated (Type-B waveform with a multiplex mode not static) and LCDIE = `1`, the device will awaken from Sleep on the next frame boundary.

FIGURE 18-20: SLEEP ENTRY/EXIT WHEN SLPEN = 1


COM0
COM1
COM2
SEG0
V_3
V_2
V_1
V_0
2 Frames
SLEEP Instruction Execution
Wake-up

18.12 Configuring the LCD Module

The following is the sequence of steps to configure the LCD module.

1. Select the frame clock prescale using bits LP<3:0> of the LCDPS register.
2. Configure the appropriate pins to function as segment drivers using the LCDSEn registers.
3. Configure the LCD module for the following using the LCDCON register:
    - Multiplex and Bias mode, bits LMUX<1:0>
    - Timing source, bits CS<1:0>
    - Sleep mode, bit SLPEN
4. Write initial values to pixel data registers, LCDDATA0 through LCDDATA21.
5. Clear LCD Interrupt Flag, LCDIF bit of the PIR2 register and if desired, enable the interrupt by setting bit LCDIE of the PIE2 register.
6. Configure bias voltages by setting the LCDRL, LCDREF and the associated ANSELx registers as needed.
7. Enable the LCD module by setting bit LCDEN of the LCDCON register.

18.13 Disabling the LCD Module

To disable the LCD module, write all '`0`'s to the LCDCON register.

18.14 LCD Current Consumption

When using the LCD module the current consumption consists of the following three factors:

- Oscillator Selection
- LCD Bias Source
- Capacitance of the LCD segments

The current consumption of just the LCD module can be considered negligible compared to these other factors.

18.14.1 OSCILLATOR SELECTION

The current consumed by the clock source selected must be considered when using the LCD module. See **Section 21.0 "Electrical Specifications"** for oscillator current consumption information.

18.14.2 LCD BIAS SOURCE

The LCD bias source, internal or external, can contribute significantly to the current consumption. Use the highest possible resistor values while maintaining contrast to minimize current.

18.14.3 CAPACITANCE OF THE LCD SEGMENTS

The LCD segments which can be modeled as capacitors which must be both charged and discharged every frame. The size of the LCD segment and its technology determines the segment's capacitance.

TABLE 18-8: SUMMARY OF REGISTERS ASSOCIATED WITH LCD OPERATION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	60
LCDCON	LCDEN	SLPEN	WERR	—	CS1	CS0	LMUX<1:0>		135
LCDCST	—	—	—	—	—	LCDCST<2:0>			138
LCDDATA0	SEG7 COM0	SEG6 COM0	SEG5 COM0	SEG4 COM0	SEG3 COM0	SEG2 COM0	SEG1 COM0	SEG0 COM0	139
LCDDATA1	SEG15 COM0	SEG14 COM0	SEG13 COM0	SEG12 COM0	SEG11 COM0	SEG10 COM0	SEG9 COM0	SEG8 COM0	139
LCDDATA3	SEG7 COM1	SEG6 COM1	SEG5 COM1	SEG4 COM1	SEG3 COM1	SEG2 COM1	SEG1 COM1	SEG0 COM1	139
LCDDATA4	SEG15 COM1	SEG14 COM1	SEG13 COM1	SEG12 COM1	SEG11 COM1	SEG10 COM1	SEG9 COM1	SEG8 COM1	139
LCDDATA6	SEG7 COM2	SEG6 COM2	SEG5 COM2	SEG4 COM2	SEG3 COM2	SEG2 COM2	SEG1 COM2	SEG0 COM2	139
LCDDATA7	SEG15 COM2	SEG14 COM2	SEG13 COM2	SEG12 COM2	SEG11 COM2	SEG10 COM2	SEG9 COM2	SEG8 COM2	139
LCDDATA9	SEG7 COM3	SEG6 COM3	SEG5 COM3	SEG4 COM3	SEG3 COM3	SEG2 COM3	SEG1 COM3	SEG0 COM3	139
LCDDATA10	SEG15 COM3	SEG14 COM3	SEG13 COM3	SEG12 COM3	SEG11 COM3	SEG10 COM3	SEG9 COM3	SEG8 COM3	139
LCDDATA12	—	—	—	—	—	SEG26 COM0	SEG25 COM0	SEG24 COM0	139
LCDDATA15	—	—	—	—	—	SEG26 COM1	SEG25 COM1	SEG24 COM1	139
LCDDATA18	—	—	—	—	—	SEG26 COM2	SEG25 COM2	SEG24 COM2	139
LCDDATA21	—	—	—	—	—	SEG26 COM3	SEG25 COM3	SEG24 COM3	139
LCDPS	WFT	BIASMD	LCDA	WA	LP<3:0>				136
LCDREF	LCDIRE	—	LCDIRI	—	VLCD3PE	VLCD2PE	VLCD1PE	—	137
LCDRL	LRLAP<1:0>		LRLBP<1:0>		—	LRLAT<2:0>			146
LCDSE0	SE<7:0>								139
LCDSE1	SE<15:8>								139
LCDSE3	—	—	—	—	—	SE<26:24>			139
PIE2	—	—	—	—	—	LCDIE	—	—	62
PIR2	—	—	—	—	—	LCDIF	—	—	64
T1CON	TMR1CS1	TMR1CS0	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	—	TMR1ON	130

Legend: — = unimplemented location, read as ‘`0`’. Shaded cells are not used by the LCD module.

19.0 IN-CIRCUIT SERIAL PROGRAMMING™ (ICSP™)

ICSP™ programming allows customers to manufacture circuit boards with unprogrammed devices. Programming can be done after the assembly process allowing the device to be programmed with the most recent firmware or a custom firmware. Five pins are needed for ICSP™ programming:

- ICSPCLK
- ICSPDAT
- $\overline{\text{MCLR}}$/VPP
- VDD
- VSS

In Program/Verify mode the Program Memory, User IDs and the Configuration Words are programmed through serial communications. The ICSPDAT pin is a bidirectional I/O used for transferring the serial data and the ICSPCLK pin is the clock input. For more information on ICSP™ refer to the **“***PIC16F193X/LF193X/PIC16F194X/LF194X/PIC16LF190X Memory Programming Specification***”** (DS41397).

19.1 High-Voltage Programming Entry Mode

The device is placed into High-Voltage Programming Entry mode by holding the ICSPCLK and ICSPDAT pins low then raising the voltage on $\overline{\text{MCLR}}$/VPP to VIHH.

19.2 Low-Voltage Programming Entry Mode

The Low-Voltage Programming Entry mode allows the PIC® Flash MCUs to be programmed using VDD only, without high voltage. When the LVP bit of Configuration Word 2 is set to ‘`1`’, the low-voltage ICSP programming entry is enabled. To disable the Low-Voltage ICSP mode, the LVP bit must be programmed to ‘`0`’.

Entry into the Low-Voltage Programming Entry mode requires the following steps:

1. $\overline{\text{MCLR}}$ is brought to VIL.
2. A 32-bit key sequence is presented on ICSPDAT, while clocking ICSPCLK.

Once the key sequence is complete, $\overline{\text{MCLR}}$ must be held at VIL for as long as Program/Verify mode is to be maintained.

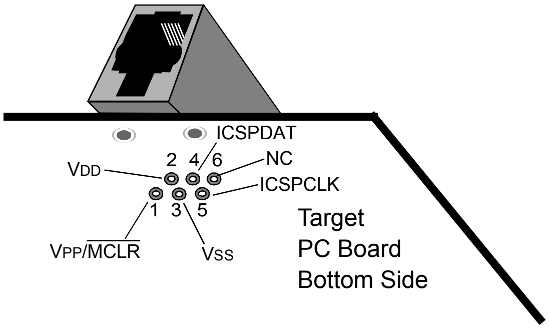
If low-voltage programming is enabled (LVP = `1`), the $\overline{\text{MCLR}}$ Reset function is automatically enabled and cannot be disabled. See **Section 5.3 “Low-Power Brown-out Reset (LPBOR)”** for more information.

The LVP bit can only be reprogrammed to ‘`0`’ by using the High-Voltage Programming mode.

19.3 Common Programming Interfaces

Connection to a target device is typically done through an ICSP™ header. A commonly found connector on development tools is the RJ-11 in the 6P6C (6-pin, 6 connector) configuration. See Figure 19-1.

FIGURE 19-1: ICD RJ-11 STYLE CONNECTOR INTERFACE



Pin Description*

1 = VPP/$\overline{\text{MCLR}}$

2 = VDD Target

3 = VSS (ground)

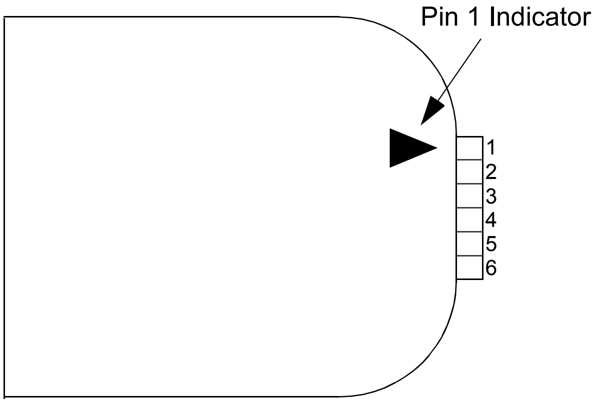
4 = ICSPDAT

5 = ICSPCLK

6 = No Connect

Another connector often found in use with the PICkit™ programmers is a standard 6-pin header with 0.1 inch spacing. Refer to Figure 19-2.

FIGURE 19-2: PICkit™ STYLE CONNECTOR INTERFACE



Pin Description*

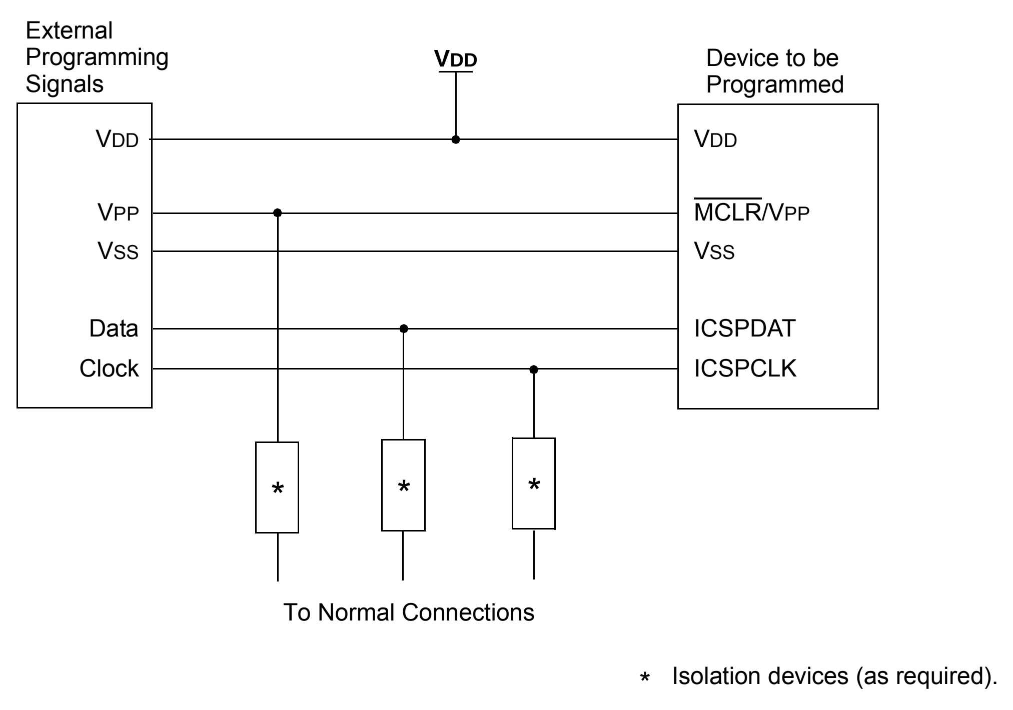
1 = VPP/$\overline{\text{MCLR}}$
2 = VDD Target
3 = VSS (ground)
4 = ICSPDAT
5 = ICSPCLK
6 = No Connect

* The 6-pin header (0.100" spacing) accepts 0.025" square pins.

For additional interface recommendations, refer to your specific device programmer manual prior to PCB design.

It is recommended that isolation devices be used to separate the programming pins from other circuitry. The type of isolation is highly dependent on the specific application and may include devices such as resistors, diodes, or even jumpers. See Figure 19-3 for more information.

FIGURE 19-3: TYPICAL CONNECTION FOR ICSP™ PROGRAMMING


External
Programming
Signals
VDD
Device to be
Programmed
VDD
VPP
VSS
Data
Clock
VDD
MCLR/VPP
VSS
ICSPDAT
ICSPCLK
*
*
*
To Normal Connections
* Isolation devices (as required).

20.0 INSTRUCTION SET SUMMARY

Each PIC16 instruction is a 14-bit word containing the operation code (opcode) and all required operands. The opcodes are broken into three broad categories.

- Byte Oriented
- Bit Oriented
- Literal and Control

The literal and control category contains the most varied instruction word format.

Table 20-3 lists the instructions recognized by the MPASM™ assembler.

All instructions are executed within a single instruction cycle, with the following exceptions, which may take two or three cycles:

- Subroutine takes two cycles (`CALL`, `CALLW`)
- Returns from interrupts or subroutines take two cycles (`RETURN`, `RETLW`, `RETFIE`)
- Program branching takes two cycles (`GOTO`, `BRA`, `BRW`, `BTFSS`, `BTFSC`, `DECFSZ`, `INCSFZ`)
- One additional instruction cycle will be used when any instruction references an indirect file register and the file select register is pointing to program memory.

One instruction cycle consists of four oscillator cycles; for an oscillator frequency of 4 MHz, this gives a nominal instruction execution rate of 1 MHz.

All instruction examples use the format ‘`0xhh`’ to represent a hexadecimal number, where ‘h’ signifies a hexadecimal digit.

20.1 Read-Modify-Write Operations

Any instruction that specifies a file register as part of the instruction performs a Read-Modify-Write (R-M-W) operation. The register is read, the data is modified, and the result is stored according to either the instruction, or the destination designator ‘d’. A read operation is performed on a register even if the instruction writes to that register.

TABLE 20-1: OPCODE FIELD DESCRIPTIONS

Field	Description
f	Register file address (0x00 to 0x7F)
W	Working register (accumulator)
b	Bit address within an 8-bit file register
k	Literal field, constant data or label
x	Don’t care location (= `0` or `1`). The assembler will generate code with x = `0`. It is the recommended form of use for compatibility with all Microchip software tools.
d	Destination select; d = `0`: store result in W, d = `1`: store result in file register f. Default is d = `1`.
n	FSR or INDF number. (0-1)
mm	Pre-post increment-decrement mode selection

TABLE 20-2: ABBREVIATION DESCRIPTIONS

Field	Description
PC	Program Counter
$\overline{\text{TO}}$	Time-out bit
C	Carry bit
DC	Digit carry bit
Z	Zero bit
$\overline{\text{PD}}$	Power-down bit

FIGURE 20-1: GENERAL FORMAT FOR INSTRUCTIONS

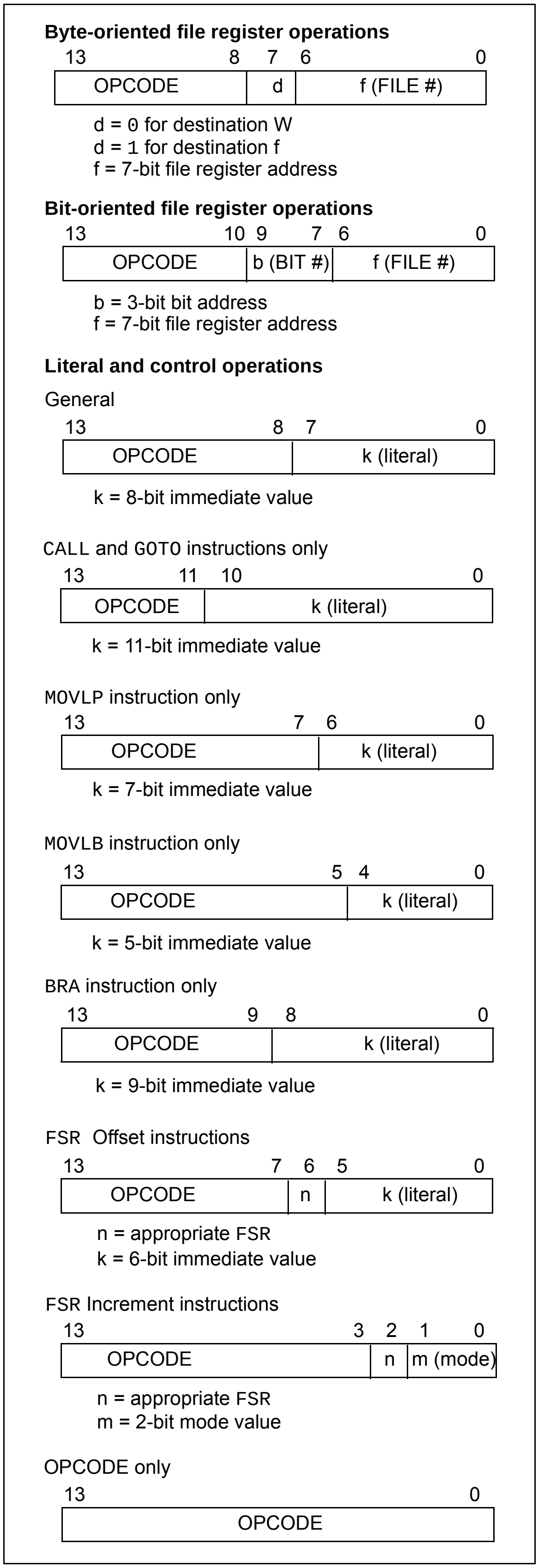

Byte-oriented file register operations
13 8 7 6 0
OPCODE
d
f (FILE #)
d = 0 for destination W
d = 1 for destination f
f = 7-bit file register address
Bit-oriented file register operations
13 10 9 7 6 0
OPCODE
b (BIT #)
f (FILE #)
b = 3-bit bit address
f = 7-bit file register address
Literal and control operations
General
13 8 7 0
OPCODE
k (literal)
k = 8-bit immediate value
CALL and GOTO instructions only
13 11 10 0
OPCODE
k (literal)
k = 11-bit immediate value
MOVLP instruction only
13 7 6 0
OPCODE
k (literal)
k = 7-bit immediate value
MOVLB instruction only
13 5 4 0
OPCODE
k (literal)
k = 5-bit immediate value
BRA instruction only
13 9 8 0
OPCODE
k (literal)
k = 9-bit immediate value
FSR Offset instructions
13 7 6 5 0
OPCODE
n
k (literal)
n = appropriate FSR
k = 6-bit immediate value
FSR Increment instructions
13 3 2 1 0
OPCODE
n
m (mode)
n = appropriate FSR
m = 2-bit mode value
OPCODE only
13 0
OPCODE

TABLE 20-3: PIC16LF1902/3 ENHANCED INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	14-Bit Opcode				Status Affected	Notes
				MSb			LSb		
BYTE-ORIENTED FILE REGISTER OPERATIONS									
ADDWF	f, d	Add W and f	1	`00`	`0111`	`dfff`	`ffff`	C, DC, Z	2
ADDWFC	f, d	Add with Carry W and f	1	`11`	`1101`	`dfff`	`ffff`	C, DC, Z	2
ANDWF	f, d	AND W with f	1	`00`	`0101`	`dfff`	`ffff`	Z	2
ASRF	f, d	Arithmetic Right Shift	1	`11`	`0111`	`dfff`	`ffff`	C, Z	2
LSLF	f, d	Logical Left Shift	1	`11`	`0101`	`dfff`	`ffff`	C, Z	2
LSRF	f, d	Logical Right Shift	1	`11`	`0110`	`dfff`	`ffff`	C, Z	2
CLRF	f	Clear f	1	`00`	`0001`	`lfff`	`ffff`	Z	2
CLRW	–	Clear W	1	`00`	`0001`	`0000`	`00xx`	Z	
COMF	f, d	Complement f	1	`00`	`1001`	`dfff`	`ffff`	Z	2
DECF	f, d	Decrement f	1	`00`	`0011`	`dfff`	`ffff`	Z	2
INCF	f, d	Increment f	1	`00`	`1010`	`dfff`	`ffff`	Z	2
IORWF	f, d	Inclusive OR W with f	1	`00`	`0100`	`dfff`	`ffff`	Z	2
MOVF	f, d	Move f	1	`00`	`1000`	`dfff`	`ffff`	Z	2
MOVWF	f	Move W to f	1	`00`	`0000`	`1fff`	`ffff`		2
RLF	f, d	Rotate Left f through Carry	1	`00`	`1101`	`dfff`	`ffff`	C	2
RRF	f, d	Rotate Right f through Carry	1	`00`	`1100`	`dfff`	`ffff`	C	2
SUBWF	f, d	Subtract W from f	1	`00`	`0010`	`dfff`	`ffff`	C, DC, Z	2
SUBWFB	f, d	Subtract with Borrow W from f	1	`11`	`1011`	`dfff`	`ffff`	C, DC, Z	2
SWAPF	f, d	Swap nibbles in f	1	`00`	`1110`	`dfff`	`ffff`		2
XORWF	f, d	Exclusive OR W with f	1	`00`	`0110`	`dfff`	`ffff`	Z	2
BYTE ORIENTED SKIP OPERATIONS									
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	`00`	`1011`	`dfff`	`ffff`		1, 2
INCFSZ	f, d	Increment f, Skip if 0	1(2)	`00`	`1111`	`dfff`	`ffff`		1, 2
BIT-ORIENTED FILE REGISTER OPERATIONS									
BCF	f, b	Bit Clear f	1	`01`	`00bb`	`bfff`	`ffff`		2
BSF	f, b	Bit Set f	1	`01`	`01bb`	`bfff`	`ffff`		2
BIT-ORIENTED SKIP OPERATIONS									
BTFSC	f, b	Bit Test f, Skip if Clear	1 (2)	`01`	`10bb`	`bfff`	`ffff`		1, 2
BTFSS	f, b	Bit Test f, Skip if Set	1 (2)	`01`	`11bb`	`bfff`	`ffff`		1, 2
LITERAL OPERATIONS									
ADDLW	k	Add literal and W	1	`11`	`1110`	`kkkk`	`kkkk`	C, DC, Z	
ANDLW	k	AND literal with W	1	`11`	`1001`	`kkkk`	`kkkk`	Z	
IORLW	k	Inclusive OR literal with W	1	`11`	`1000`	`kkkk`	`kkkk`	Z	
MOVLB	k	Move literal to BSR	1	`00`	`0000`	`001k`	`kkkk`		
MOVLP	k	Move literal to PCLATH	1	`11`	`0001`	`1kkk`	`kkkk`		
MOVLW	k	Move literal to W	1	`11`	`0000`	`kkkk`	`kkkk`		
SUBLW	k	Subtract W from literal	1	`11`	`1100`	`kkkk`	`kkkk`	C, DC, Z	
XORLW	k	Exclusive OR literal with W	1	`11`	`1010`	`kkkk`	`kkkk`	Z	

Note 1: If the Program Counter (PC) is modified, or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

2: If this instruction addresses an INDF register and the MSb of the corresponding FSR is set, this instruction will require one additional instruction cycle.

TABLE 20-3: PIC16LF1902/3 ENHANCED INSTRUCTION SET (CONTINUED)

Mnemonic, Operands		Description	Cycles	14-Bit Opcode MSb LSb	Status Affected	Notes
		CONTROL OPERATIONS				
BRA	k	Relative Branch	2	`11 001k kkkk kkkk`		
BRW	–	Relative Branch with W	2	`00 0000 0000 1011`		
CALL	k	Call Subroutine	2	`10 0kkk kkkk kkkk`		
CALLW	–	Call Subroutine with W	2	`00 0000 0000 1010`		
GOTO	k	Go to address	2	`10 1kkk kkkk kkkk`		
RETFIE	k	Return from interrupt	2	`00 0000 0000 1001`		
RETLW	k	Return with literal in W	2	`11 0100 kkkk kkkk`		
RETURN	–	Return from Subroutine	2	`00 0000 0000 1000`		
		INHERENT OPERATIONS				
CLRWDT	–	Clear Watchdog Timer	1	`00 0000 0110 0100`	$\overline{\text{TO}}$, $\overline{\text{PD}}$	
NOP	–	No Operation	1	`00 0000 0000 0000`		
OPTION	–	Load OPTION_REG register with W	1	`00 0000 0110 0010`		
RESET	–	Software device Reset	1	`00 0000 0000 0001`		
SLEEP	–	Go into Standby mode	1	`00 0000 0110 0011`	$\overline{\text{TO}}$, $\overline{\text{PD}}$	
TRIS	f	Load TRIS register with W	1	`00 0000 0110 0fff`		
		C-COMPILER OPTIMIZED				
ADDFSR	n, k	Add Literal k to FSRn	1	`11 0001 0nkk kkkk`		
MOVIW	n mm	Move Indirect FSRn to W with pre/post inc/dec modifier, mm	1	`00 0000 0001 0nmm`	Z	**2, 3**
	k[n]	Move INDFn to W, Indexed Indirect.	1	`11 1111 0nkk kkkk`	Z	**2**
MOVWI	n mm	Move W to Indirect FSRn with pre/post inc/dec modifier, mm	1	`00 0000 0001 1nmm`		**2, 3**
	k[n]	Move W to INDFn, Indexed Indirect.	1	`11 1111 1nkk kkkk`		**2**

Note 1: If the Program Counter (PC) is modified, or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

2: If this instruction addresses an INDF register and the MSb of the corresponding FSR is set, this instruction will require one additional instruction cycle.

3: See Table in the MOVIW and MOVWI instruction descriptions.

20.2 Instruction Descriptions

ADDFSR	**Add Literal to FSRn**
Syntax:	[*label*] ADDFSR FSRn, k
Operands:	-32 ≤ k ≤ 31 n ∈ [0, 1]
Operation:	FSR(n) + k → FSR(n)
Status Affected:	None
Description:	The signed 6-bit literal ‘k’ is added to the contents of the FSRnH:FSRnL register pair. FSRn is limited to the range 0000h - FFFFh. Moving beyond these bounds will cause the FSR to wrap-around.

ADDLW	**Add literal and W**
Syntax:	[*label*] ADDLW k
Operands:	0 ≤ k ≤ 255
Operation:	(W) + k → (W)
Status Affected:	C, DC, Z
Description:	The contents of the W register are added to the 8-bit literal ‘k’ and the result is placed in the W register.

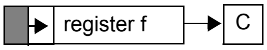
ADDWF	**Add W and f**
Syntax:	[*label*] ADDWF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(W) + (f) → (destination)
Status Affected:	C, DC, Z
Description:	Add the contents of the W register with register ‘f’. If ‘d’ is ‘0’, the result is stored in the W register. If ‘d’ is ‘1’, the result is stored back in register ‘f’.

ADDWFC	**ADD W and CARRY bit to f**
Syntax:	[*label*] ADDWFC f {,d}
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(W) + (f) + (C) → dest
Status Affected:	C, DC, Z
Description:	Add W, the Carry flag and data memory location ‘f’. If ‘d’ is ‘0’, the result is placed in W. If ‘d’ is ‘1’, the result is placed in data memory location ‘f’.

ANDLW	**AND literal with W**
Syntax:	[*label*] ANDLW k
Operands:	0 ≤ k ≤ 255
Operation:	(W) .AND. (k) → (W)
Status Affected:	Z
Description:	The contents of W register are AND’ed with the 8-bit literal ‘k’. The result is placed in the W register.

ANDWF	**AND W with f**
Syntax:	[*label*] ANDWF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(W) .AND. (f) → (destination)
Status Affected:	Z
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’.

ASRF	**Arithmetic Right Shift**
Syntax:	[*label*] ASRF f {,d}
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(f<7>)→ dest<7> (f<7:1>) → dest<6:0>, (f<0>) → C,
Status Affected:	C, Z
Description:	The contents of register ‘f’ are shifted one bit to the right through the Carry flag. The MSb remains unchanged. If ‘d’ is ‘0’, the result is placed in W. If ‘d’ is ‘1’, the result is stored back in register ‘f’.



BCF	**Bit Clear f**
Syntax:	[*label*] BCF f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b \leq 7$
Operation:	`0` → (f<b>)
Status Affected:	None
Description:	Bit ‘b’ in register ‘f’ is cleared.

BRA	**Relative Branch**
Syntax:	[*label*] BRA label [*label*] BRA $+k
Operands:	$-256 \leq label - PC + 1 \leq 255$ $-256 \leq k \leq 255$
Operation:	(PC) + 1 + k → PC
Status Affected:	None
Description:	Add the signed 9-bit literal ‘k’ to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 1 + k. This instruction is a 2-cycle instruction. This branch has a limited range.

BRW	**Relative Branch with W**
Syntax:	[*label*] BRW
Operands:	None
Operation:	(PC) + (W) → PC
Status Affected:	None
Description:	Add the contents of W (unsigned) to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 1 + (W). This instruction is a 2-cycle instruction.

BSF	**Bit Set f**
Syntax:	[*label*] BSF f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b \leq 7$
Operation:	`1` → (f<b>)
Status Affected:	None
Description:	Bit ‘b’ in register ‘f’ is set.

BTFSC	**Bit Test f, Skip if Clear**
Syntax:	[*label*] BTFSC f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b \leq 7$
Operation:	skip if (f<b>) = `0`
Status Affected:	None
Description:	If bit ‘b’ in register ‘f’ is ‘`1`’, the next instruction is executed. If bit ‘b’, in register ‘f’, is ‘`0`’, the next instruction is discarded, and a `NOP` is executed instead, making this a 2-cycle instruction.

BTFSS	**Bit Test f, Skip if Set**
Syntax:	[*label*] BTFSS f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b < 7$
Operation:	skip if (f<b>) = `1`
Status Affected:	None
Description:	If bit ‘b’ in register ‘f’ is ‘`0`’, the next instruction is executed. If bit ‘b’ is ‘`1`’, then the next instruction is discarded and a `NOP` is executed instead, making this a 2-cycle instruction.

CALL	**Call Subroutine**
Syntax:	[*label*] CALL k
Operands:	0 ≤ k ≤ 2047
Operation:	(PC)+ 1→ TOS, k → PC<10:0>, (PCLATH<6:3>) → PC<14:11>
Status Affected:	None
Description:	Call Subroutine. First, return address (PC + 1) is pushed onto the stack. The 11-bit immediate address is loaded into PC bits <10:0>. The upper bits of the PC are loaded from PCLATH. `CALL` is a 2-cycle instruction.

CALLW	**Subroutine Call With W**
Syntax:	[*label*] CALLW
Operands:	None
Operation:	(PC) +1 → TOS, (W) → PC<7:0>, (PCLATH<6:0>) → PC<14:8>
Status Affected:	None
Description:	Subroutine call with W. First, the return address (PC + 1) is pushed onto the return stack. Then, the contents of W is loaded into PC<7:0>, and the contents of PCLATH into PC<14:8>. `CALLW` is a 2-cycle instruction.

CLRF	**Clear f**
Syntax:	[*label*] CLRF f
Operands:	0 ≤ f ≤ 127
Operation:	00h → (f) `1` → Z
Status Affected:	Z
Description:	The contents of register ‘f’ are cleared and the Z bit is set.

CLRW	**Clear W**
Syntax:	[*label*] CLRW
Operands:	None
Operation:	00h → (W) `1` → Z
Status Affected:	Z
Description:	W register is cleared. Zero bit (Z) is set.

CLRWDT	**Clear Watchdog Timer**
Syntax:	[*label*] CLRWDT
Operands:	None
Operation:	00h → WDT `0` → WDT prescaler, `1` → $\overline{TO}$ `1` → $\overline{PD}$
Status Affected:	$\overline{TO}$, $\overline{PD}$
Description:	`CLRWDT` instruction resets the Watchdog Timer. It also resets the prescaler of the WDT. Status bits $\overline{TO}$ and $\overline{PD}$ are set.

COMF	**Complement f**
Syntax:	[*label*] COMF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [`0`,`1`]
Operation:	$(\bar{f})$ → (destination)
Status Affected:	Z
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’.

DECF	**Decrement f**
Syntax:	[*label*] DECF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [`0`,`1`]
Operation:	(f) - 1 → (destination)
Status Affected:	Z
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’.

DECFSZ	**Decrement f, Skip if 0**
Syntax:	[*label*] DECFSZ f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(f) - 1 → (destination); skip if result = 0
Status Affected:	None
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 ‘1’, the next instruction is executed. If the result is ‘0’, then a NOP is executed instead, making it a 2-cycle instruction.

GOTO	**Unconditional Branch**
Syntax:	[*label*] GOTO k
Operands:	0 ≤ k ≤ 2047
Operation:	k → PC<10:0> PCLATH<6:3> → PC<14:11>
Status Affected:	None
Description:	GOTO is an unconditional branch. The 11-bit immediate value is loaded into PC bits <10:0>. The upper bits of PC are loaded from PCLATH<4:3>. GOTO is a 2-cycle instruction.

INCF	**Increment f**
Syntax:	[*label*] INCF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(f) + 1 → (destination)
Status Affected:	Z
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’.

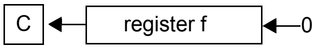
INCFSZ	**Increment f, Skip if 0**
Syntax:	[*label*] INCFSZ f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(f) + 1 → (destination), skip if result = 0
Status Affected:	None
Description:	The contents of register ‘f’ are incremented. If ‘d’ is ‘0’, the result is placed in the W register. If ‘d’ is ‘1’, the result is placed back in register ‘f’. If the result is ‘1’, the next instruction is executed. If the result is ‘0’, a NOP is executed instead, making it a 2-cycle instruction.

IORLW	**Inclusive OR literal with W**
Syntax:	[*label*] IORLW k
Operands:	0 ≤ k ≤ 255
Operation:	(W) .OR. k → (W)
Status Affected:	Z
Description:	The contents of the W register are OR’ed with the 8-bit literal ‘k’. The result is placed in the W register.

IORWF	**Inclusive OR W with f**
Syntax:	[*label*] IORWF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(W) .OR. (f) → (destination)
Status Affected:	Z
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’.

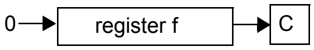
LSLF Logical Left Shift

Syntax:	[*label*] LSLF f {,d}
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(f<7>) → C (f<6:0>) → dest<7:1> 0 → dest<0>
Status Affected:	C, Z
Description:	The contents of register ‘f’ are shifted one bit to the left through the Carry flag. A ‘0’ is shifted into the LSb. If ‘d’ is ‘0’, the result is placed in W. If ‘d’ is ‘1’, the result is stored back in register ‘f’.



LSRF Logical Right Shift

Syntax:	[*label*] LSRF f {,d}
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	0 → dest<7> (f<7:1>) → dest<6:0>, (f<0>) → C,
Status Affected:	C, Z
Description:	The contents of register ‘f’ are shifted one bit to the right through the Carry flag. A ‘0’ is shifted into the MSb. If ‘d’ is ‘0’, the result is placed in W. If ‘d’ is ‘1’, the result is stored back in register ‘f’.



MOVF Move f

Syntax:	[*label*] MOVF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(f) → (dest)
Status Affected:	Z
Description:	The contents of register f is moved to a destination dependent upon the status of d. If d = 0, destination is W register. If d = 1, the destination is file register f itself. d = 1 is useful to test a file register since status flag Z is affected.
Words:	1
Cycles:	1
Example:	`MOVF FSR, 0` After Instruction W = value in FSR register Z = 1

MOVIW Move INDFn to W

Syntax:	[*label*] MOVIW ++FSRn [*label*] MOVIW --FSRn [*label*] MOVIW FSRn++ [*label*] MOVIW FSRn-- [*label*] MOVIW k[FSRn]
Operands:	n ∈ [`0`,`1`] mm ∈ [`00`,`01`, `10`, `11`] -32 ≤ k ≤ 31
Operation:	INDFn → W Effective address is determined by • FSR + 1 (preincrement) • FSR - 1 (predecrement) • FSR + k (relative offset) After the Move, the FSR value will be either: • FSR + 1 (all increments) • FSR - 1 (all decrements) • Unchanged
Status Affected:	Z

Mode	Syntax	mm
Preincrement	++FSRn	`00`
Predecrement	--FSRn	`01`
Postincrement	FSRn++	`10`
Postdecrement	FSRn--	`11`

Description:	This instruction is used to move data between W and one of the indirect registers (INDFn). Before/after this move, the pointer (FSRn) is updated by pre/post incrementing/decrementing it. **Note:** The INDFn registers are not physical registers. Any instruction that accesses an INDFn register actually accesses the register at the address specified by the FSRn. FSRn is limited to the range 0000h - FFFFh. Incrementing/decrementing it beyond these bounds will cause it to wrap-around.

MOVLB Move literal to BSR

Syntax:	[*label*] MOVLB k
Operands:	0 ≤ k ≤ 15
Operation:	k → BSR
Status Affected:	None
Description:	The 5-bit literal 'k' is loaded into the Bank Select Register (BSR).

MOVLP Move literal to PCLATH

Syntax:	[*label*] MOVLP k
Operands:	0 ≤ k ≤ 127
Operation:	k → PCLATH
Status Affected:	None
Description:	The 7-bit literal 'k' is loaded into the PCLATH register.

MOVLW Move literal to W

Syntax:	[*label*] MOVLW k
Operands:	0 ≤ k ≤ 255
Operation:	k → (W)
Status Affected:	None
Description:	The 8-bit literal 'k' is loaded into W register. The "don't cares" will assemble as '`0`'s.
Words:	1
Cycles:	1
<u>Example:</u>	`MOVLW 0x5A` After Instruction W = 0x5A

MOVWF Move W to f

Syntax:	[*label*] MOVWF f
Operands:	0 ≤ f ≤ 127
Operation:	(W) → (f)
Status Affected:	None
Description:	Move data from W register to register 'f'.
Words:	1
Cycles:	1
<u>Example:</u>	`MOVWF OPTION_REG` Before Instruction OPTION_REG = 0xFF W = 0x4F After Instruction OPTION_REG = 0x4F W = 0x4F

MOVWI	Move W to INDFn
Syntax:	[*label*] MOVWI ++FSRn [*label*] MOVWI --FSRn [*label*] MOVWI FSRn++ [*label*] MOVWI FSRn-- [*label*] MOVWI k[FSRn]
Operands:	n ∈ [`0,1`] mm ∈ [`00,01`, `10`, `11`] $-32 \le k \le 31$
Operation:	W → INDFn Effective address is determined by • FSR + 1 (preincrement) • FSR - 1 (predecrement) • FSR + k (relative offset) After the Move, the FSR value will be either: • FSR + 1 (all increments) • FSR - 1 (all decrements) Unchanged
Status Affected:	None

Mode	Syntax	mm
Preincrement	++FSRn	`00`
Predecrement	--FSRn	`01`
Postincrement	FSRn++	`10`
Postdecrement	FSRn--	`11`

Description:	This instruction is used to move data between W and one of the indirect registers (INDFn). Before/after this move, the pointer (FSRn) is updated by pre/post incrementing/decrementing it. **Note:** The INDFn registers are not physical registers. Any instruction that accesses an INDFn register actually accesses the register at the address specified by the FSRn. FSRn is limited to the range 0000h - FFFFh. Incrementing/decrementing it beyond these bounds will cause it to wrap-around. The increment/decrement operation on FSRn WILL NOT affect any Status bits.

NOP	No Operation
Syntax:	[*label*] NOP
Operands:	None
Operation:	No operation
Status Affected:	None
Description:	No operation.
Words:	1
Cycles:	1
<u>Example:</u>	`NOP`

OPTION	Load OPTION_REG Register with W
Syntax:	[*label*] OPTION
Operands:	None
Operation:	(W) → OPTION_REG
Status Affected:	None
Description:	Move data from W register to OPTION_REG register.
Words:	1
Cycles:	1
<u>Example:</u>	`OPTION` Before Instruction OPTION_REG = 0xFF W = 0x4F After Instruction OPTION_REG = 0x4F W = 0x4F

RESET	Software Reset
Syntax:	[*label*] RESET
Operands:	None
Operation:	Execute a device Reset. Resets the nRI flag of the PCON register.
Status Affected:	None
Description:	This instruction provides a way to execute a hardware Reset by software.

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

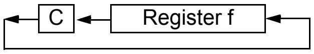
After Interrupt
PC = TOS
GIE = `1`

RETLW	**Return with literal in W**
Syntax:	[*label*] RETLW k
Operands:	0 ≤ k ≤ 255
Operation:	k → (W); TOS → PC
Status Affected:	None
Description:	The W register is loaded with the 8-bit literal ‘k’. The program counter is loaded from the top of the stack (the return address). This is a 2-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 k8

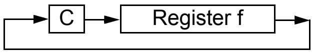
RETURN	**Return from Subroutine**
Syntax:	[*label*] RETURN
Operands:	None
Operation:	TOS → PC
Status Affected:	None
Description:	Return from subroutine. The stack is POPed and the top of the stack (TOS) is loaded into the program counter. This is a 2-cycle instruction.

RLF	**Rotate Left f through Carry**
Syntax:	[*label*] RLF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [`0`,`1`]
Operation:	See description below
Status Affected:	C
Description:	The contents of register ‘f’ are rotated one bit to the left through the Carry flag. If ‘d’ is ‘`0`’, the result is placed in the W register. If ‘d’ is ‘`1`’, the result is stored back in register ‘f’. 
Words:	1
Cycles:	1
Example:	`RLF     REG1,0`

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

RRF Rotate Right f through Carry

Syntax:	[*label*] RRF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	See description below
Status Affected:	C
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’.



SLEEP Enter Sleep mode

Syntax:	[*label*] SLEEP
Operands:	None
Operation:	00h → WDT, 0 → WDT prescaler, 1 → $\overline{TO}$, 0 → $\overline{PD}$
Status Affected:	$\overline{TO}$, $\overline{PD}$
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.

SUBLW Subtract W from literal

Syntax:	[*label*] SUBLW k
Operands:	0 ≤ k ≤ 255
Operation:	k - (W) → (W)
Status Affected:	C, DC, Z
Description:	The W register is subtracted (2’s complement method) from the 8-bit literal ‘k’. The result is placed in the W register.

C = 0	W > k
C = 1	W ≤ k
DC = 0	W<3:0> > k<3:0>
DC = 1	W<3:0> ≤ k<3:0>

SUBWF Subtract W from f

Syntax:	[*label*] SUBWF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(f) - (W) → (destination)
Status Affected:	C, DC, Z
Description:	Subtract (2’s complement method) W register from register ‘f’. If ‘d’ is ‘0’, the result is stored in the W register. If ‘d’ is ‘1’, the result is stored back in register ‘f.

C = 0	W > f
C = 1	W ≤ f
DC = 0	W<3:0> > f<3:0>
DC = 1	W<3:0> ≤ f<3:0>

SUBWFB Subtract W from f with Borrow

Syntax:	SUBWFB f {,d}
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(f) – (W) – ($\overline{B}$) → dest
Status Affected:	C, DC, Z
Description:	Subtract W and the BORROW flag (CARRY) from register ‘f’ (2’s complement method). If ‘d’ is ‘0’, the result is stored in W. If ‘d’ is ‘1’, the result is stored back in register ‘f’.

SWAPF	Swap Nibbles in f
Syntax:	[*label*] SWAPF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [`0`,`1`]
Operation:	(f<3:0>) → (destination<7:4>), (f<7:4>) → (destination<3:0>)
Status Affected:	None
Description:	The upper and lower nibbles of register ‘f’ are exchanged. If ‘d’ is ‘`0`’, the result is placed in the W register. If ‘d’ is ‘`1`’, the result is placed in register ‘f’.

TRIS	Load TRIS Register with W
Syntax:	[*label*] TRIS f
Operands:	5 ≤ f ≤ 7
Operation:	(W) → TRIS register ‘f’
Status Affected:	None
Description:	Move data from W register to TRIS register. When ‘f’ = 5, TRISA is loaded. When ‘f’ = 6, TRISB is loaded. When ‘f’ = 7, TRISC is loaded.

XORLW	Exclusive OR literal with W
Syntax:	[*label*] XORLW k
Operands:	0 ≤ k ≤ 255
Operation:	(W) .XOR. k → (W)
Status Affected:	Z
Description:	The contents of the W register are XOR’ed with the 8-bit literal ‘k’. The result is placed in the W register.

XORWF	Exclusive OR W with f
Syntax:	[*label*] XORWF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [`0`,`1`]
Operation:	(W) .XOR. (f) → (destination)
Status Affected:	Z
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’.

21.0 ELECTRICAL SPECIFICATIONS

Absolute Maximum Ratings(†)

Ambient temperature under bias	-40°C to +125°C
Storage temperature	-65°C to +150°C
Voltage on pins with respect to VSS	
on VDD	-0.3V to +4.0V
on $\overline{\text{MCLR}}$	-0.3V to +9.0V
on all other pins	-0.3V to (VDD + 0.3V)
Total power dissipation(1)	800 mW
Maximum current	
out of VSS pin	
-40°C ≤ TA ≤ +85°C for industrial	350 mA
-40°C ≤ TA ≤ +125°C for extended	95 mA
into VDD pin	
-40°C ≤ TA ≤ +85°C for industrial	250 mA
-40°C ≤ TA ≤ +125°C for extended	70 mA
Clamp current, IK (VPIN < 0 or VPIN > VDD)	± 20 mA
Maximum output current	
sunk by any I/O pin	50 mA
sourced by any I/O pin	50 mA

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

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

21.1 Standard Operating Conditions

The standard operating conditions for any device are defined as:

Operating Voltage:	VDDMIN ≤ VDD ≤ VDDMAX
Operating Temperature:	TA_MIN ≤ TA ≤ TA_MAX

VDD — Operating Supply Voltage(2)

VDDMIN (FOSC ≤ 16 MHz)	+1.8V
VDDMIN (FOSC ≤ 20 MHz)	+2.3V
VDDMAX	+3.6V

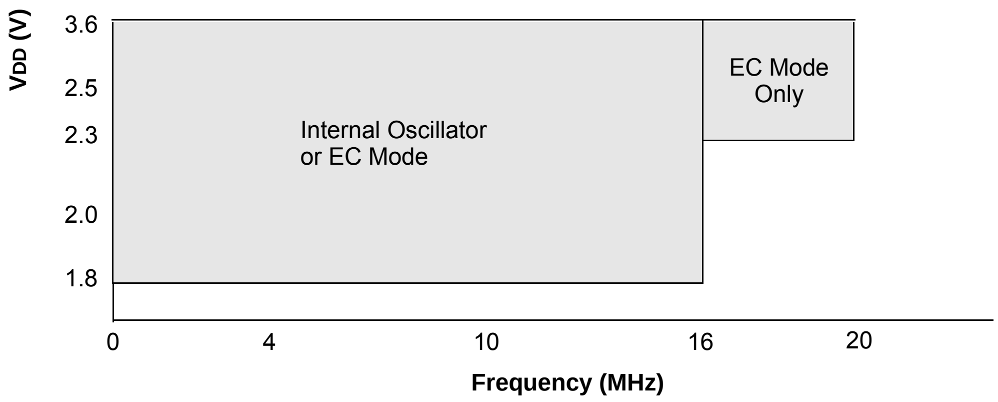
TA — Operating Ambient Temperature Range

Industrial Temperature	
TA_MIN	-40°C
TA_MAX	+85°C
Extended Temperature	
TA_MIN	-40°C
TA_MAX	+125°C

Note 1: Maximum current rating requires even load distribution across I/O pins. Maximum current rating may be limited by the device package power dissipation characterizations, see Table 21-6: “Thermal Characteristics” to calculate device specifications.

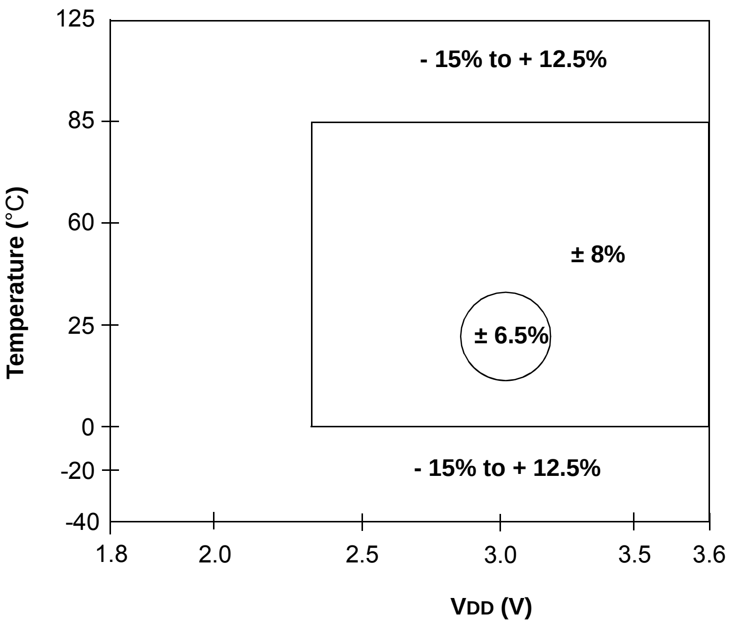
2: See Parameter D001, DS Characteristics: Supply Voltage.

FIGURE 21-1: VOLTAGE FREQUENCY GRAPH, -40°C ≤ T_A ≤ +125°C



Note 1: The shaded region indicates the permissible combinations of voltage and frequency.
2: Refer to Table 21-7 for each Oscillator mode's supported frequencies.

FIGURE 21-2: HFINTOSC FREQUENCY ACCURACY OVER DEVICE V_{DD} AND TEMPERATURE



21.2 DC Characteristics

TABLE 21-1: SUPPLY VOLTAGE

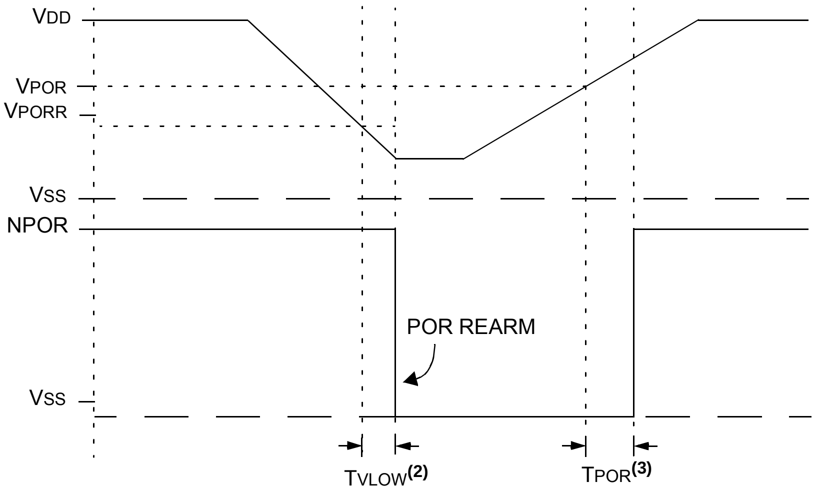
PIC16LF1902/3			**Standard Operating Conditions (unless otherwise stated)**				
Param. No.	**Sym.**	**Characteristic**	**Min.**	**Typ†**	**Max.**	**Units**	**Conditions**
D001	VDD	**Supply Voltage**	1.8	—	3.6	V	FOSC ≤ 16 MHz
			2.3	—	3.6	V	FOSC ≤ 20 MHz (EC mode)
D002*	VDR	**RAM Data Retention Voltage**[(1)]	1.5	—	—	V	Device in Sleep mode
D002A*	VPOR*	**Power-on Reset Release Voltage**	1.54	1.64	1.74	V	
D002B*	VPORR*	**Power-on Reset Rearm Voltage**	—	1.7	—	V	Device in Sleep mode
D003	VADFVR	**Fixed Voltage Reference Voltage for ADC, Initial Accuracy**	6 7 7 8	— — — —	4 4 6 6	%	1.024V, VDD ≥ 1.8V, 85°C 1.024V, VDD ≥ 1.8V, 125°C 2.048V, VDD ≥ 2.5V, 85°C 2.048V, VDD ≥ 2.5V, 125°C
D003A	VCDAFVR	**Fixed Voltage Reference Voltage for Comparator and DAC, Initial Accuracy**	7 8 8 9	— — — —	5 5 7 7	%	1.024V, VDD ≥ 1.8V, 85°C 1.024V, VDD ≥ 1.8V, 125°C 2.048V, VDD ≥ 2.5V, 85°C 2.048V, VDD ≥ 2.5V, 125°C
D003B	VLCDFVR	**Fixed Voltage Reference Voltage for LCD Bias, Initial Accuracy**	9 9.5	— —	9 9	%	3.072V, VDD ≥ 3.6V, 85°C 3.072V, VDD ≥ 3.6V, 125°C
D003C*	TCVFVR	**Temperature Coefficient, Fixed Voltage Reference**	—	-130	—	ppm/°C	
D003D*	ΔVFVR/ ΔVIN	**Line Regulation, Fixed Voltage Reference**	—	0.270	—	%/V	
D004*	SVDD	**VDD Rise Rate** to ensure internal Power-on Reset signal	0.05	—	—	V/ms	See **Section 5.1 “Power-on Reset (POR)”** for details.

* These parameters are characterized but not tested.

† Data in “Typ” column is at 3.3V, 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.

FIGURE 21-3: POR AND POR REARM WITH SLOW RISING VDD



Note 1: When NPOR is low, the device is held in Reset.

2: TPOR 1 µs typical.

3: TVLOW 2.7 µs typical.

TABLE 21-2: SUPPLY CURRENT (IDD)(1,2)

PIC16LF1902/3			Standard Operating Conditions (unless otherwise stated)				
Param. No.	**Device Characteristics**	**Min.**	**Typ†**	**Max.**	**Units**	**Conditions**	
						VDD	**Note**
D010		—	58	75	μA	1.8	FOSC = 1 MHz EC Oscillator mode High-Power mode
		—	115	140	μA	3.0	
		—	133	176	μA	3.6	
D011		—	130	200	μA	1.8	FOSC = 4 MHz EC Oscillator mode High-Power mode
		—	245	300	μA	3.0	
		—	290	350	μA	3.6	
D012		—	218	275	μA	1.8	FOSC = 500 kHz HFINTOSC mode
		—	283	375	μA	3.0	
		—	314	395	μA	3.6	
D013		—	233	325	μA	1.8	FOSC = 1 MHz HFINTOSC mode
		—	309	425	μA	3.0	
		—	347	475	μA	3.6	
D014		—	305	360	μA	1.8	FOSC = 4 MHz HFINTOSC mode
		—	433	520	μA	3.0	
		—	500	600	μA	3.6	
D015		—	395	480	μA	1.8	FOSC = 8 MHz HFINTOSC mode
		—	600	720	μA	3.0	
		—	700	850	μA	3.6	
D016		—	567	670	μA	1.8	FOSC = 16 MHz HFINTOSC mode
		—	915	1100	μA	3.0	
		—	1087	1300	μA	3.6	
D017		—	2.7	7.2	μA	1.8	FOSC = 31 kHz LFINTOSC mode -40°C ≤ TA ≤ +125°C
		—	4.5	9.7	μA	3.0	
		—	5.2	12.0	μA	3.6	
D017A		—	2.7	6.5	μA	1.8	FOSC = 31 kHz LFINTOSC mode -40°C ≤ TA ≤ +85°C
		—	4.5	9.0	μA	3.0	
		—	5.2	11.0	μA	3.6	
D018		—	2.4	6.7	μA	1.8	FOSC = 32 kHz EC Oscillator mode, Low-Power mode -40°C ≤ TA ≤ +125°C
		—	4.2	9.2	μA	3.0	
		—	4.8	11.5	μA	3.6	
D018A		—	2.4	6.0	μA	1.8	FOSC = 32 kHz EC Oscillator mode, Low-Power mode -40°C ≤ TA ≤ +85°C
		—	4.2	8.5	μA	3.0	
		—	4.8	10.5	μA	3.6	

† Data in “Typ” column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: The test conditions for all IDD measurements in active operation mode are: OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD; $\overline{\text{MCLR}}$ = VDD; WDT disabled.

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.

3: FVR and BOR are disabled.

TABLE 21-3: POWER-DOWN CURRENTS (IPD)

PIC16LF1902			Standard Operating Conditions (unless otherwise stated)					
Param. No.	**Device Characteristics**	**Min.**	**Typ†**	**Max. +85°C**	**Max. +125°C**	**Units**	**Conditions**	
							VDD	**Note**
	Power-down Base Current (IPD)[2]							
D023		—	0.15	1.0	3.0	μA	1.8	WDT, BOR, FVR and T1OSC disabled, all Peripherals Inactive
		—	0.16	2.0	4.0	μA	3.0	
		—	0.65	3.0	5.0	μA	3.6	
D024		—	0.27	2.0	4.0	μA	1.8	WDT Current **(Note 1)**
		—	0.56	3.0	5.0	μA	3.0	
		—	0.75	4.0	6.0	μA	3.6	
D025		—	17.5	31	35	μA	1.8	FVR current
		—	17.7	33	38	μA	3.0	
		—	17.8	35	41	μA	3.6	
D026		—	0.15	2.3	3.56	μA	3.0	LPBOR current
		—	0.21	3.4	4.70	μA	3.6	
D027		—	7.0	10	12	μA	3.0	BOR Current
		—	7.5	12	14	μA	3.6	
D028		—	0.50	2.0	4.0	μA	1.8	T1OSC Current
		—	0.60	3.0	5.0	μA	3.0	
		—	0.70	4.0	6.0	μA	3.6	
D029		—	0.40	2.0	4.0	μA	1.8	ADC Current **(Note 1, Note 3)**, no conversion in progress
		—	0.70	3.0	5.0	μA	3.0	
		—	0.90	4.0	6.0	μA	3.6	
D030		—	—	250	—	μA	1.8	ADC Current **(Note 1, Note 3)**, conversion in progress
		—	—	250	—	μA	3.0	
		—	—	250	—	μA	3.6	
D031	LCD Bias Ladder							
	Low power	—	1	2	6	μA	1.8	
	Medium Power	—	10	13	21	μA	3.0	
	High Power	—	100	111	120	μA	3.6	

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Legend: TBD = To Be Determined

Note 1: The peripheral current is the sum of the base IDD or IPD and the additional current consumed when this peripheral is enabled. The peripheral Δ current can be determined by subtracting the base IDD or IPD current from this limit. Max values should be used when calculating total current consumption.

2: 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 high-impedance state and tied to VDD.

3: A/D oscillator source is FRC.

TABLE 21-4: I/O PORTS

DC CHARACTERISTICS			Standard Operating Conditions (unless otherwise stated)				
Param No.	**Sym.**	**Characteristic**	**Min.**	**Typ†**	**Max.**	**Units**	**Conditions**
	VIL	**Input Low Voltage**					
		I/O PORT:					
D032		with TTL buffer	—	—	0.15 VDD	V	1.8V ≤ VDD ≤ 3.6V
D033		with Schmitt Trigger buffer	—	—	0.2 VDD	V	1.8V ≤ VDD ≤ 3.6V
D034		$\overline{\text{MCLR}}$, OSC1	—	—	0.2 VDD	V	
	VIH	**Input High Voltage**					
		I/O ports:					
D040		with TTL buffer	0.25 VDD + 0.8	—	—	V	1.8V ≤ VDD ≤ 3.6V
D041		with Schmitt Trigger buffer	0.8 VDD	—	—	V	1.8V ≤ VDD ≤ 3.6V
D042		$\overline{\text{MCLR}}$	0.8 VDD	—	—	V	
	IIL	**Input Leakage Current(2)**					
D060		I/O ports	—	± 5 ± 5	± 125 ± 1000	nA nA	VSS ≤ VPIN ≤ VDD, Pin at high-impedance @ 85°C 125°C
D061		$\overline{\text{MCLR}}$(3)	—	± 50	± 200	nA	VSS ≤ VPIN ≤ VDD @ 85°C
	IPUR	**Weak Pull-up Current**					
D070*			25	100	200	μA	VDD = 3.3V, VPIN = VSS
	VOL	**Output Low Voltage**					
D080		I/O ports	—	—	0.6	V	IOL = 6mA, VDD = 3.3V IOL = 1.8mA, VDD = 1.8V
	VOH	**Output High Voltage**					
D090		I/O ports	VDD - 0.7	—	—	V	IOH = 3mA, VDD = 3.3V IOH = 1mA, VDD = 1.8V
		Capacitive Loading Specs on Output Pins					
D101*	CIO	All I/O pins	—	—	50	pF	

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: Negative current is defined as current sourced by the pin.

TABLE 21-5: MEMORY PROGRAMMING REQUIREMENTS

DC CHARACTERISTICS			**Standard Operating Conditions (unless otherwise stated)**				
Param. No.	**Sym.**	**Characteristic**	**Min.**	**Typ†**	**Max.**	**Units**	**Conditions**
		Program Memory High Voltage Programming Specifications					
D110	VIHH	Voltage on $\overline{\text{MCLR}}$/VPP/RE3 pin	8.0	—	9.0	V	**(Note 2)**
D111	IDDVPP	Programming/Erase Current on VPP, High Voltage Programming	—	—	10	mA	
D112		VDD for Bulk Erase	2.7	—	VDD max.	V	
D113	VPEW	VDD for Write or Row Erase	VDD min.	—	VDD max.	V	
D114	IPPPGM	Programming/Erase Current on VPP, Low Voltage Programming	—	—	1.0	mA	
D115	IDDPGM	Programming/Erase Current on VDD, High or Low Voltage Programming	—		5.0	mA	
		Program Flash Memory					
D121	EP	Cell Endurance	1K	10K	—	E/W	-40°C to +85°C (**Note 1**)
D122	VPR	VDD for Read	VDD min.	—	VDD max.	V	
D123	TIW	Self-timed Write Cycle Time	—	2	2.5	ms	
D124	TRETD	Characteristic Retention	40	—	—	Year	Provided no other specifications are violated

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: Self-write and Block Erase.

2: Required only if single-supply programming is disabled.

TABLE 21-6: THERMAL CONSIDERATIONS

Standard Operating Conditions (unless otherwise stated)					
Param. No.	**Sym.**	**Characteristic**	**Typ.**	**Units**	**Conditions**
TH01	θJA	Thermal Resistance Junction to Ambient	60	°C/W	28-pin SPDIP package
			80	°C/W	28-pin SOIC package
			90	°C/W	28-pin SSOP package
			27.5	°C/W	28-pin UQFN 4x4mm package
TH02	θJC	Thermal Resistance Junction to Case	31.4	°C/W	28-pin SPDIP package
			24	°C/W	28-pin SOIC package
			24	°C/W	28-pin SSOP package
			24	°C/W	28-pin UQFN 4x4mm package
TH03	TJMAX	Maximum Junction Temperature	150	°C	
TH04	PD	Power Dissipation	—	W	PD = PINTERNAL + PI/O
TH05	PINTERNAL	Internal Power Dissipation	—	W	PINTERNAL = IDD x VDD(1)
TH06	PI/O	I/O Power Dissipation	—	W	PI/O = Σ (IOL * VOL) + Σ (IOH * (VDD - VOH))
TH07	PDER	Derated Power	—	W	PDER = PDMAX (TJ - TA)/θJA(2)

Note 1: IDD is current to run the chip alone without driving any load on the output pins.

2: TA = Ambient Temperature

3: TJ = Junction Temperature

21.3 AC Characteristics

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

1. TppS2ppS
2. TppS

T			
F	Frequency	T	Time

Lowercase letters (pp) and their meanings:

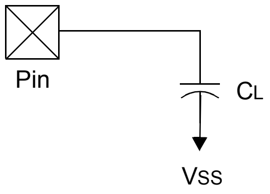
pp			
cc	CCP1	osc	OSC1
ck	CLKOUT	rd	$\overline{\text{RD}}$
cs	$\overline{\text{CS}}$	rw	$\overline{\text{RD}}$ or $\overline{\text{WR}}$
di	SDI	sc	SCK
do	SDO	ss	$\overline{\text{SS}}$
dt	Data in	t0	T0CKI
io	I/O PORT	t1	T1CKI
mc	$\overline{\text{MCLR}}$	wr	$\overline{\text{WR}}$

Uppercase letters and their meanings:

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

FIGURE 21-4: LOAD CONDITIONS

Load Condition



Legend: C_L = 50 pF for all pins, 15 pF for OSC2 output

TABLE 21-7: CLOCK OSCILLATOR TIMING REQUIREMENTS

Standard Operating Conditions (unless otherwise stated)

Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions
OS01	FOSC	External CLKIN Frequency(1)	DC DC DC	— — —	0.5 4 20	MHz MHz MHz	External Clock (ECL) External Clock (ECM) External Clock (ECH)
OS02	TOSC	External CLKIN Period(1)	50	—	∞	ns	External Clock (EC)
OS03	TCY	Instruction Cycle Time(1)	200	TCY	DC	ns	TCY = 4/FOSC

* These parameters are characterized but not tested.

† Data in “Typ” column is at 3.0V, 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 CLKIN pin. When an external clock input is used, the “max” cycle time limit is “DC” (no clock) for all devices.

TABLE 21-8: OSCILLATOR PARAMETERS

Standard Operating Conditions (unless otherwise stated)

Param. No.	Sym.	Characteristic	Freq. Tolerance	Min.	Typ†	Max.	Units	Conditions
OS08	HFOSC	Internal Calibrated HFINTOSC Frequency(2)	±8% ±6.5%	— —	16 16	— —	MHz MHz	0°C ≤ TA ≤ +85°C VDD = 3.0V at +25°C
OS10A*	TIOSC ST	HFINTOSC 16 MHz Oscillator Wake-up from Sleep Start-up Time	— —	— —	5 5	15 15	μs μs	VDD = 2.0V, -40°C to +85°C VDD = 3.0V, -40°C to +85°C

* These parameters are characterized but not tested.

† Data in “Typ” column is at 3.0V, 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.

2: To ensure these oscillator frequency tolerances, VDD and VSS must be capacitively decoupled as close to the device as possible. 0.1 μF and 0.01 μF values in parallel are recommended.

FIGURE 21-5: CLKOUT AND I/O TIMING

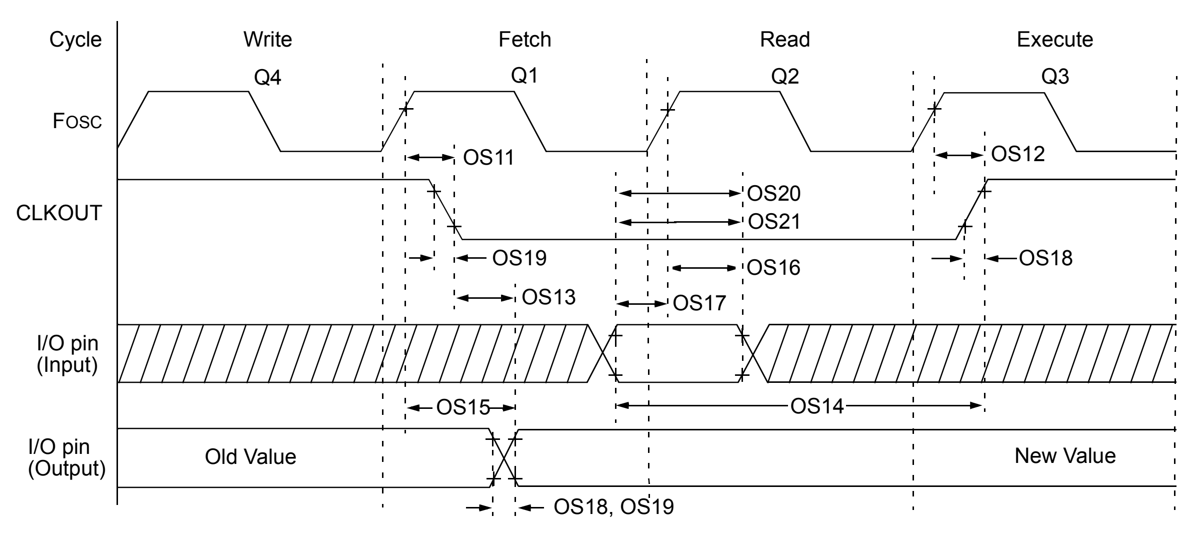
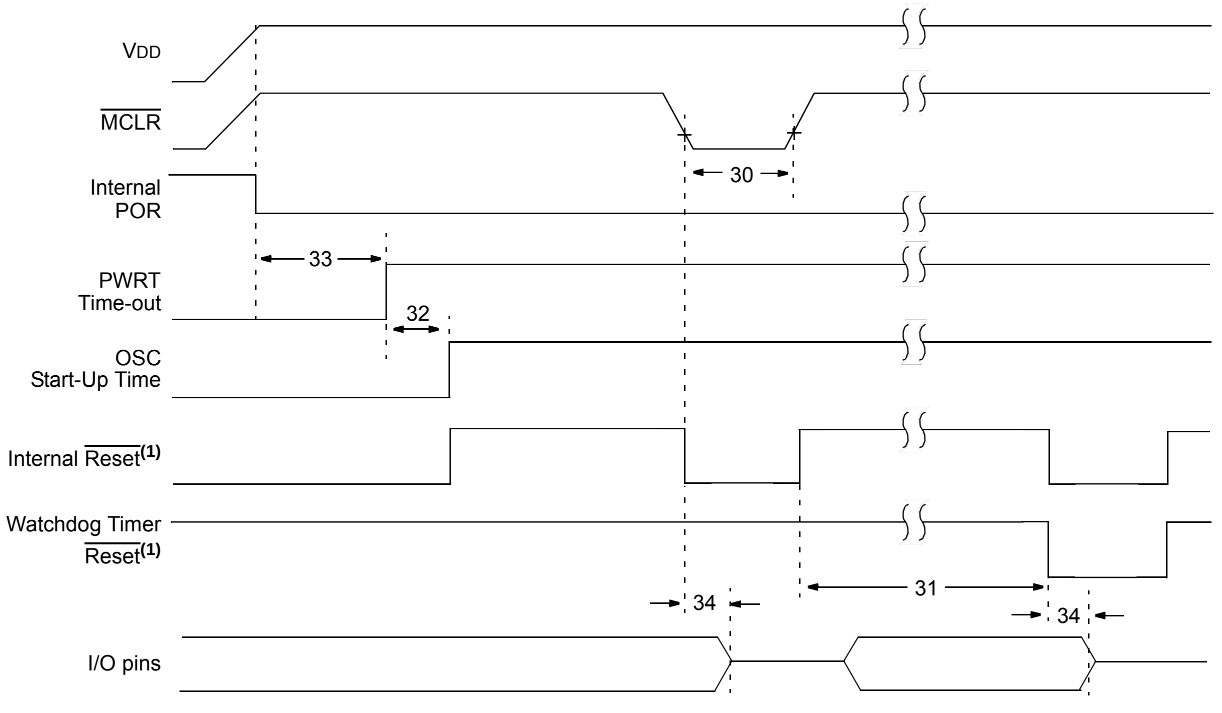

Cycle
Write
Fetch
Read
Execute
Q4
Q1
Q2
Q3
FOSC
CLKOUT
I/O pin (Input)
I/O pin (Output)
OS11
OS12
OS20
OS21
OS19
OS16
OS18
OS13
OS17
OS15
OS14
Old Value
New Value
OS18, OS19

TABLE 21-9: CLKOUT AND I/O TIMING PARAMETERS

Standard Operating Conditions (unless otherwise stated)							
Param No.	**Sym.**	**Characteristic**	**Min.**	**Typ†**	**Max.**	**Units**	**Conditions**
OS11	TosH2ckL	FOSC↑ to CLKOUT↓ (1)	—	—	70	ns	VDD = 3.0V
OS12	TosH2ckH	FOSC↑ to CLKOUT↑ (1)	—	—	72	ns	VDD = 3.0V
OS13	TckL2ioV	CLKOUT↓ to Port out valid(1)	—	—	20	ns	
OS14	TioV2ckH	Port input valid before CLKOUT↑(1)	TOSC + 200 ns	—	—	ns	
OS15	TosH2ioV	FOSC↑ (Q1 cycle) to Port out valid	—	50	70*	ns	VDD = 3.0V
OS16	TosH2ioI	FOSC↑ (Q2 cycle) to Port input invalid (I/O in hold time)	50	—	—	ns	VDD = 3.0V
OS17	TioV2osH	Port input valid to FOSC↑ (Q2 cycle) (I/O in setup time)	20	—	—	ns	
OS18	TioR	Port output rise time	— —	40 15	72 32	ns	VDD = 3.0V VDD = 2.0V
OS19	TioF	Port output fall time	— —	28 15	55 30	ns	VDD = 2.0V VDD = 3.0V
OS20*	Tinp	INT pin input high or low time	25	—	—	ns	
OS21*	Tioc	Interrupt-on-change new input level time	25	—	—	ns	

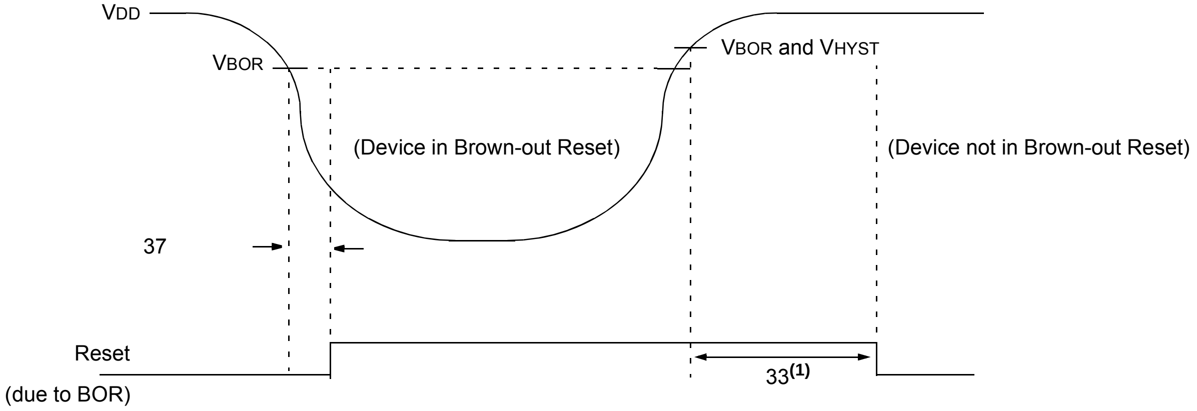
* These parameters are characterized but not tested.
† Data in “Typ” column is at 3.0V, 25°C unless otherwise stated.
Note 1: Measurements are taken in EC mode where CLKOUT output is 4 x TOSC.

FIGURE 21-6: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING


VDD
MCLR
Internal POR
PWRT Time-out
OSC Start-Up Time
Internal Reset(1)
Watchdog Timer Reset(1)
I/O pins
30
33
32
34
31
34

Note 1: Asserted low.

FIGURE 21-7: BROWN-OUT RESET TIMING AND CHARACTERISTICS


VDD
VBOR
VBOR and VHYST
(Device in Brown-out Reset)
(Device not in Brown-out Reset)
37
Reset
(due to BOR)
33(1)

Note 1: 64 ms delay only if $\overline{\text{PWRTE}}$ bit in the Configuration Word register is programmed to '`0`'.
2 ms delay if $\overline{\text{PWRTE}}$ = `0` and VREGEN = `1`.

FIGURE 21-8: MINIMUM PULSE WIDTH FOR LPBOR DETECTION

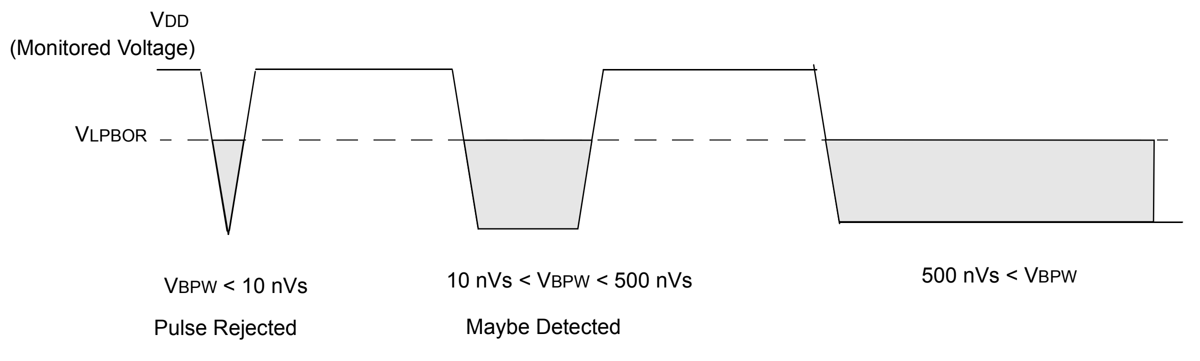

VDD
(Monitored Voltage)
VLPBOR
VBPW < 10 nVs
Pulse Rejected
10 nVs < VBPW < 500 nVs
Maybe Detected
500 nVs < VBPW

TABLE 21-10: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER AND BROWN-OUT RESET PARAMETERS

Standard Operating Conditions (unless otherwise stated)

Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions
30	TMCL	$\overline{\text{MCLR}}$ Pulse Width (low)	2 5	— —	— —	μs μs	VDD = 3.0V, -40°C to +85°C VDD = 3.0V
31	FWDTLP	Low Frequency Internal Oscillator Frequency	19	33	52	kHz	
32	TOST	Oscillator Start-up Timer Period(1)	—	1024	—	Tosc	**(Note 2)**
33*	TPWRT	Power-up Timer Period, $\overline{\text{PWRTE}}$ = `0`	—	2048	—	Tosc	Clocked by LFINTOSC
34*	TIOZ	I/O High-Impedance from $\overline{\text{MCLR}}$ Low or Watchdog Timer Reset	—	—	2.0	μs	
35	VBOR	Brown-out Reset Voltage: BORV = `0` BORV = `1`	2.55 1.80	2.70 1.90	2.85 2.05	V V	
35A*	VHYST	Brown-out Reset Hysteresis	25 —	50 —	75 100	mV mV	-40°C to +85°C -40°C to +125°C
35B*	TBORDC	Brown-out Reset DC Response Time	1 —	3 —	5 10	μs μs	VDD ≤ VBOR, -40°C to +85°C VDD ≤ VBOR
35C	TBORAC	Brown-out Reset AC Response Time	—	100	—	ns	Transient Response immunity for a noise spike that goes from VDD to VSS and back with 10 ns rise and fall times. Guidance only.
36	TFVRS	Fixed Voltage Reference Turn-on Time	—	—	5	μs	Turn on to specified stability
37	VLPBOR	Low-Power Brown-out Reset Voltage	1.85	1.95	2.10	V	-40°C to +85°C
38*	VZPHYST	Zero-Power Brown-out Reset Hysteresis	0	25	60	mV	-40°C to +85°C
39*	TZPBPW	Zero-Power Brown-out Reset AC Response Time for BOR detection	10	—	500	nVs	VDD ≤ VBOR, -40°C to +85°C

* These parameters are characterized but not tested.

† Data in “Typ” column is at 3.0V, 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.

2: Period of the slower clock.

3: To ensure these voltage tolerances, VDD and VSS must be capacitively decoupled as close to the device as possible. 0.1 μF and 0.01 μF values in parallel are recommended.

FIGURE 21-9: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS

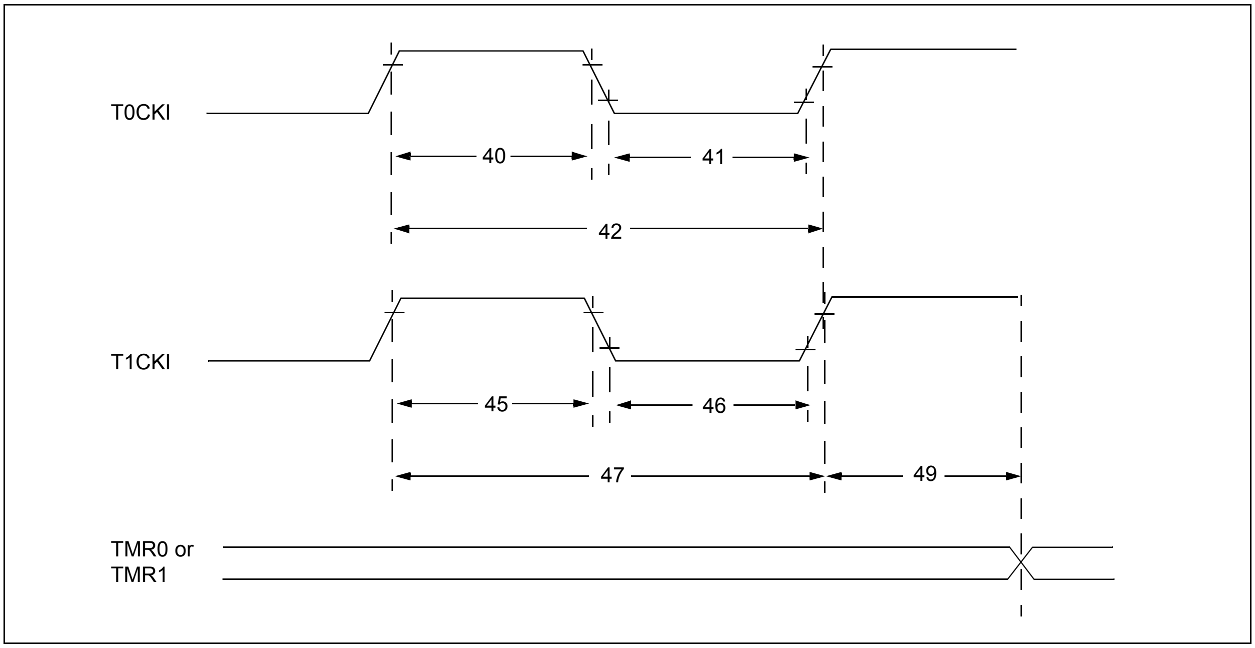


TABLE 21-11: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS

Standard Operating Conditions (unless otherwise stated)

Param No.	Sym.	Characteristic		Min.	Typ†	Max.	Units	Conditions
40*	TT0H	T0CKI High Pulse Width	No Prescaler	0.5 TCY + 20	—	—	ns	
			With Prescaler	10	—	—	ns	
41*	TT0L	T0CKI Low Pulse Width	No Prescaler	0.5 TCY + 20	—	—	ns	
			With Prescaler	10	—	—	ns	
42*	TT0P	T0CKI Period		Greater of: 20 or $\frac{T_{CY} + 40}{N}$	—	—	ns	N = prescale value (2, 4, ..., 256)
45*	TT1H	T1CKI High Time	Synchronous, No Prescaler	0.5 TCY + 20	—	—	ns	
			Synchronous, with Prescaler	15	—	—	ns	
			Asynchronous	30	—	—	ns	
46*	TT1L	T1CKI Low Time	Synchronous, No Prescaler	0.5 TCY + 20	—	—	ns	
			Synchronous, with Prescaler	15	—	—	ns	
			Asynchronous	30	—	—	ns	
47*	TT1P	T1CKI Input Period	Synchronous	Greater of: 30 or $\frac{T_{CY} + 40}{N}$	—	—	ns	N = prescale value (1, 2, 4, 8)
			Asynchronous	60	—	—	ns	
48	FT1	Timer1 Oscillator Input Frequency Range (oscillator enabled by setting bit T1OSCEN)		32.4	32.768	33.1	kHz	
49*	TCKEZTMR1	Delay from External Clock Edge to Timer Increment		2 TOSC	—	7 TOSC	—	Timers in Sync mode

* These parameters are characterized but not tested.

† Data in “Typ” column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

TABLE 21-12: PIC16LF1902/3 CONVERTER (ADC) CHARACTERISTICS$^{(1,2,3)}$

Operating Conditions (unless otherwise stated)
VDD = 3.0V, TA = 25°C

Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions
AD01	NR	Resolution	—	—	10	bit	
AD02	EIL	Integral Error	—	±1	±1.7	LSb	VREF = 3.0V
AD03	EDL	Differential Error	—	±1	±1	LSb	No missing codes VREF = 3.0V
AD04	EOFF	Offset Error	—	±1	±2.5	LSb	VREF = 3.0V
AD05	EGN	Gain Error	—	±1	±2.0	LSb	VREF = 3.0V
AD06	VREF	Reference Voltage	1.8	—	VDD	V	VREF = (VRPOS - VRNEG)
AD07	VAIN	Full-Scale Range	VSS	—	VREF	V	
AD08	ZAIN	Recommended Impedance of Analog Voltage Source	—	—	10	kΩ	Can go higher if external 0.01μF capacitor is present on input pin.

* These parameters are characterized but not tested.

† Data in “Typ” column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: Total Absolute Error includes integral, differential, offset and gain errors.

2: The ADC conversion result never decreases with an increase in the input voltage and has no missing codes.

3: See **Section 22.0 “DC and AC Characteristics Graphs and Charts”** for operating characterization.

TABLE 21-13: PIC16LF1902/3 A/D CONVERSION REQUIREMENTS

Standard Operating Conditions (unless otherwise stated)

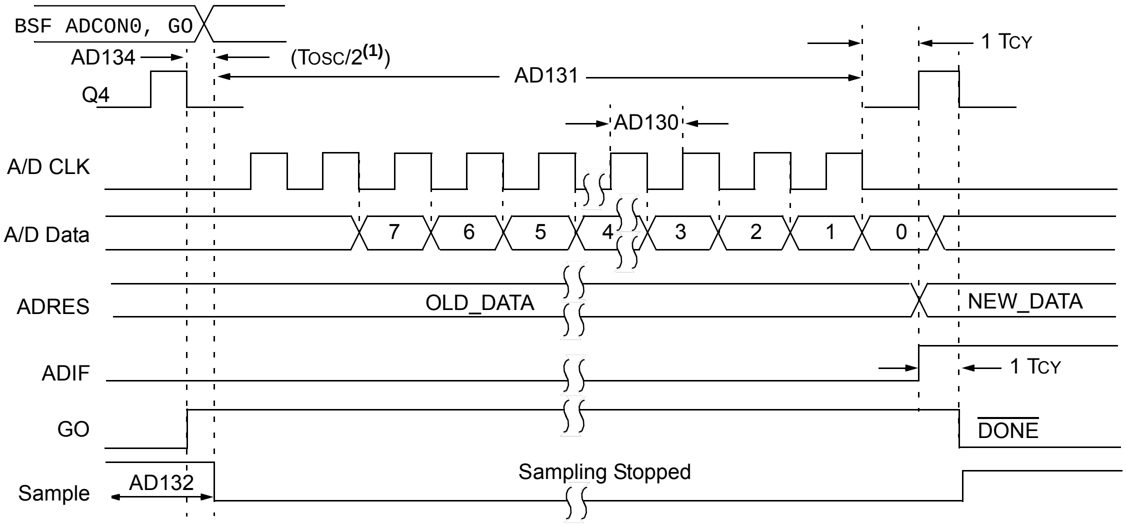
Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions
AD130*	TAD	ADC Clock Period (TADC)	1.0	—	6.0	μs	FOSC-based
		ADC Internal FRC Oscillator Period (TFRC)	1.0	2.0	6.0	μs	ADCS<2:0> = x11 (ADC FRC mode)
AD131	TCNV	Conversion Time (not including Acquisition Time)$^{(1)}$	—	11	—	TAD	Set GO/$\overline{\text{DONE}}$ bit to conversion complete
AD132*	TACQ	Acquisition Time	—	5.0	—	μs	
AD133*	THCD	Holding Capacitor Disconnect Time	— —	1/2 TAD 1/2 TAD + 1TCY	— —		FOSC-based ADCS<2:0> = x11 (ADC FRC mode)

* These parameters are characterized but not tested.

† Data in “Typ” column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

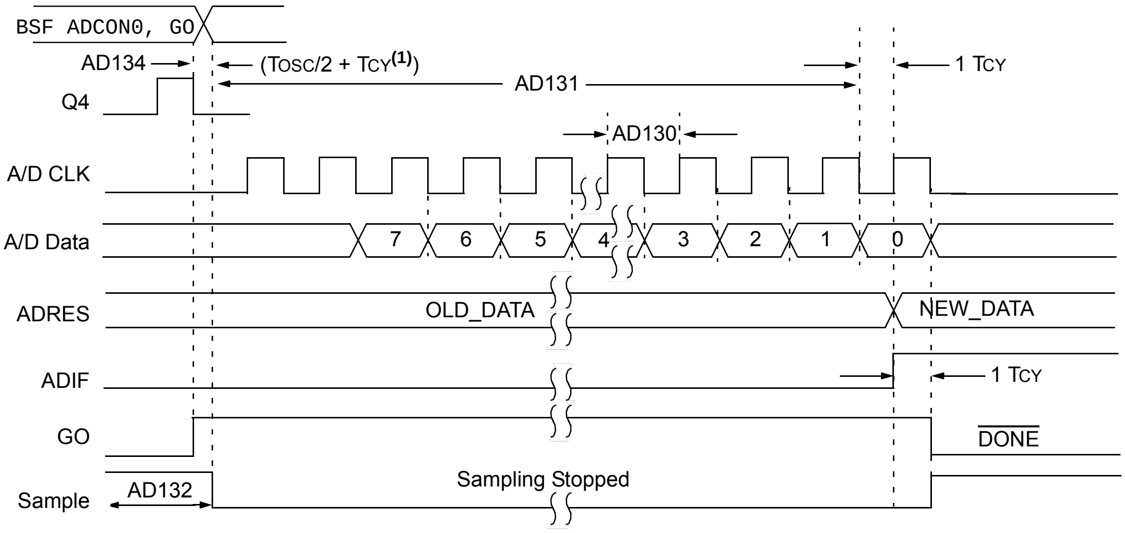
Note 1: The ADRES register may be read on the following TCY cycle.

FIGURE 21-10: PIC16LF1902/3 A/D CONVERSION TIMING (NORMAL MODE)


BSF ADCON0, GO
AD134
(TOSC/2(1))
AD131
1 TCY
Q4
AD130
A/D CLK
A/D Data
7
6
5
4
3
2
1
0
ADRES
OLD_DATA
NEW_DATA
ADIF
1 TCY
GO
DONE
Sample
AD132
Sampling Stopped

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

FIGURE 21-11: PIC16LF1902/3 A/D CONVERSION TIMING (SLEEP MODE)


BSF ADCON0, GO
AD134
(TOSC/2 + TCY(1))
AD131
1 TCY
Q4
AD130
A/D CLK
A/D Data
7
6
5
4
3
2
1
0
ADRES
OLD_DATA
NEW_DATA
ADIF
1 TCY
GO
DONE
Sample
AD132
Sampling Stopped

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

22.0 DC AND AC CHARACTERISTICS GRAPHS AND CHARTS

Graphs and charts are not available at this time.

23.0 DEVELOPMENT SUPPORT

The PIC® microcontrollers (MCU) and dsPIC® digital signal controllers (DSC) are supported with a full range of software and hardware development tools:

- Integrated Development Environment
  - MPLAB® X IDE Software
- Compilers/Assemblers/Linkers
  - MPLAB XC Compiler
  - MPASM™ Assembler
  - MPLINK™ Object Linker/ MPLIB™ Object Librarian
  - MPLAB Assembler/Linker/Librarian for Various Device Families
- Simulators
  - MPLAB X SIM Software Simulator
- Emulators
  - MPLAB REAL ICE™ In-Circuit Emulator
- In-Circuit Debuggers/Programmers
  - MPLAB ICD 3
  - PICkit™ 3
- Device Programmers
  - MPLAB PM3 Device Programmer
- Low-Cost Demonstration/Development Boards, Evaluation Kits and Starter Kits
- Third-party development tools

23.1 MPLAB X Integrated Development Environment Software

The MPLAB X IDE is a single, unified graphical user interface for Microchip and third-party software, and hardware development tool that runs on Windows®, Linux and Mac OS® X. Based on the NetBeans IDE, MPLAB X IDE is an entirely new IDE with a host of free software components and plug-ins for high-performance application development and debugging. Moving between tools and upgrading from software simulators to hardware debugging and programming tools is simple with the seamless user interface.

With complete project management, visual call graphs, a configurable watch window and a feature-rich editor that includes code completion and context menus, MPLAB X IDE is flexible and friendly enough for new users. With the ability to support multiple tools on multiple projects with simultaneous debugging, MPLAB X IDE is also suitable for the needs of experienced users.

Feature-Rich Editor:

- Color syntax highlighting
- Smart code completion makes suggestions and provides hints as you type
- Automatic code formatting based on user-defined rules
- Live parsing

User-Friendly, Customizable Interface:

- Fully customizable interface: toolbars, toolbar buttons, windows, window placement, etc.
- Call graph window

Project-Based Workspaces:

- Multiple projects
- Multiple tools
- Multiple configurations
- Simultaneous debugging sessions

File History and Bug Tracking:

- Local file history feature
- Built-in support for Bugzilla issue tracker

23.2 MPLAB XC Compilers

The MPLAB XC Compilers are complete ANSI C compilers for all of Microchip’s 8, 16, and 32-bit MCU and DSC devices. These compilers provide powerful integration capabilities, superior code optimization and ease of use. MPLAB XC Compilers run on Windows, Linux or MAC OS X.

For easy source level debugging, the compilers provide debug information that is optimized to the MPLAB X IDE.

The free MPLAB XC Compiler editions support all devices and commands, with no time or memory restrictions, and offer sufficient code optimization for most applications.

MPLAB XC Compilers include an assembler, linker and utilities. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. MPLAB XC Compiler uses the assembler to produce its object file. Notable features of the assembler include:

- Support for the entire device instruction set
- Support for fixed-point and floating-point data
- Command-line interface
- Rich directive set
- Flexible macro language
- MPLAB X IDE compatibility

23.3 MPASM Assembler

The MPASM Assembler is a full-featured, universal macro assembler for PIC10/12/16/18 MCUs.

The MPASM Assembler generates relocatable object files for the MPLINK Object Linker, Intel® standard HEX files, MAP files to detail memory usage and symbol reference, absolute LST files that contain source lines and generated machine code, and COFF files for debugging.

The MPASM Assembler features include:

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

23.4 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK Object Linker combines relocatable objects created by the MPASM Assembler. It can link relocatable objects from precompiled libraries, using directives from a linker script.

The MPLIB Object Librarian manages the creation and modification of library files of precompiled code. When a routine from a library is called from a source file, only the modules that contain that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications.

The object linker/library features include:

- Efficient linking of single libraries instead of many smaller files
- Enhanced code maintainability by grouping related modules together
- Flexible creation of libraries with easy module listing, replacement, deletion and extraction

23.5 MPLAB Assembler, Linker and Librarian for Various Device Families

MPLAB Assembler produces relocatable machine code from symbolic assembly language for PIC24, PIC32 and dsPIC DSC devices. MPLAB XC Compiler uses the assembler to produce its object file. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. Notable features of the assembler include:

- Support for the entire device instruction set
- Support for fixed-point and floating-point data
- Command-line interface
- Rich directive set
- Flexible macro language
- MPLAB X IDE compatibility

23.6 MPLAB X SIM Software Simulator

The MPLAB X SIM Software Simulator allows code development in a PC-hosted environment by simulating the PIC MCUs and dsPIC DSCs on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a comprehensive stimulus controller. Registers can be logged to files for further run-time analysis. The trace buffer and logic analyzer display extend the power of the simulator to record and track program execution, actions on I/O, most peripherals and internal registers.

The MPLAB X SIM Software Simulator fully supports symbolic debugging using the MPLAB XC Compilers, and the MPASM and MPLAB Assemblers. The software simulator offers the flexibility to develop and debug code outside of the hardware laboratory environment, making it an excellent, economical software development tool.

23.7 MPLAB REAL ICE In-Circuit Emulator System

The MPLAB REAL ICE In-Circuit Emulator System is Microchip's next generation high-speed emulator for Microchip Flash DSC and MCU devices. It debugs and programs all 8, 16 and 32-bit MCU, and DSC devices with the easy-to-use, powerful graphical user interface of the MPLAB X IDE.

The emulator is connected to the design engineer's PC using a high-speed USB 2.0 interface and is connected to the target with either a connector compatible with in-circuit debugger systems (RJ-11) or with the new high-speed, noise tolerant, Low-Voltage Differential Signal (LVDS) interconnection (CAT5).

The emulator is field upgradable through future firmware downloads in MPLAB X IDE. MPLAB REAL ICE offers significant advantages over competitive emulators including full-speed emulation, run-time variable watches, trace analysis, complex breakpoints, logic probes, a ruggedized probe interface and long (up to three meters) interconnection cables.

23.8 MPLAB ICD 3 In-Circuit Debugger System

The MPLAB ICD 3 In-Circuit Debugger System is Microchip's most cost-effective, high-speed hardware debugger/programmer for Microchip Flash DSC and MCU devices. It debugs and programs PIC Flash microcontrollers and dsPIC DSCs with the powerful, yet easy-to-use graphical user interface of the MPLAB IDE.

The MPLAB ICD 3 In-Circuit Debugger probe is connected to the design engineer's PC using a high-speed USB 2.0 interface and is connected to the target with a connector compatible with the MPLAB ICD 2 or MPLAB REAL ICE systems (RJ-11). MPLAB ICD 3 supports all MPLAB ICD 2 headers.

23.9 PICkit 3 In-Circuit Debugger/ Programmer

The MPLAB PICkit 3 allows debugging and programming of PIC and dsPIC Flash microcontrollers at a most affordable price point using the powerful graphical user interface of the MPLAB IDE. The MPLAB PICkit 3 is connected to the design engineer's PC using a full-speed USB interface and can be connected to the target via a Microchip debug (RJ-11) connector (compatible with MPLAB ICD 3 and MPLAB REAL ICE). The connector uses two device I/O pins and the Reset line to implement in-circuit debugging and In-Circuit Serial Programming™ (ICSP™).

23.10 MPLAB PM3 Device Programmer

The MPLAB PM3 Device Programmer is a universal, CE compliant device programmer with programmable voltage verification at V_{DDMIN} and V_{DDMAX} for maximum reliability. It features a large LCD display (128 x 64) for menus and error messages, and a modular, detachable socket assembly to support various package types. The ICSP cable assembly is included as a standard item. In Stand-Alone mode, the MPLAB PM3 Device Programmer can read, verify and program PIC devices without a PC connection. It can also set code protection in this mode. The MPLAB PM3 connects to the host PC via an RS-232 or USB cable. The MPLAB PM3 has high-speed communications and optimized algorithms for quick programming of large memory devices, and incorporates an MMC card for file storage and data applications.

23.11 Demonstration/Development Boards, Evaluation Kits, and Starter Kits

A wide variety of demonstration, development and evaluation boards for various PIC MCUs and dsPIC DSCs allows quick application development on fully functional systems. Most boards include prototyping areas for adding custom circuitry and provide application firmware and source code for examination and modification.

The boards support a variety of features, including LEDs, temperature sensors, switches, speakers, RS-232 interfaces, LCD displays, potentiometers and additional EEPROM memory.

The demonstration and development boards can be used in teaching environments, for prototyping custom circuits and for learning about various microcontroller applications.

In addition to the PICDEM™ and dsPICDEM™ demonstration/development board series of circuits, Microchip has a line of evaluation kits and demonstration software for analog filter design, KEELOQ® security ICs, CAN, IrDA®, PowerSmart battery management, SEEVAL® evaluation system, Sigma-Delta ADC, flow rate sensing, plus many more.

Also available are starter kits that contain everything needed to experience the specified device. This usually includes a single application and debug capability, all on one board.

Check the Microchip web page (www.microchip.com) for the complete list of demonstration, development and evaluation kits.

23.12 Third-Party Development Tools

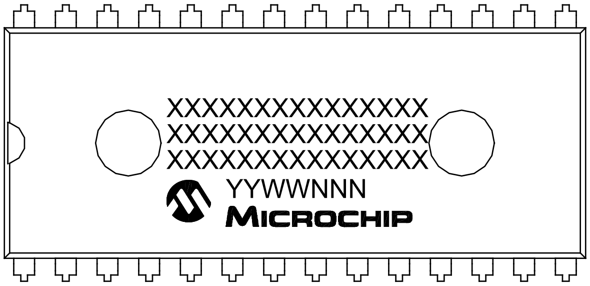
Microchip also offers a great collection of tools from third-party vendors. These tools are carefully selected to offer good value and unique functionality.

- Device Programmers and Gang Programmers from companies, such as SoftLog and CCS
- Software Tools from companies, such as Gimpel and Trace Systems
- Protocol Analyzers from companies, such as Saleae and Total Phase
- Demonstration Boards from companies, such as MikroElektronika, Digilent® and Olimex
- Embedded Ethernet Solutions from companies, such as EZ Web Lynx, WIZnet and IPLogika®

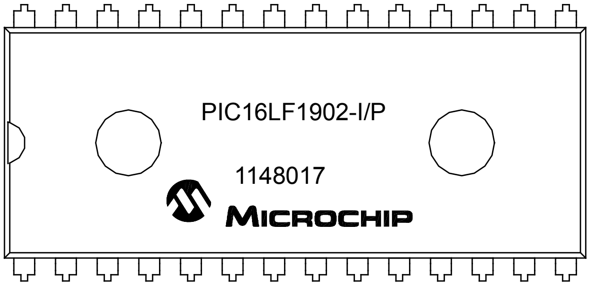
24.0 PACKAGING INFORMATION

24.1 Package Marking Information

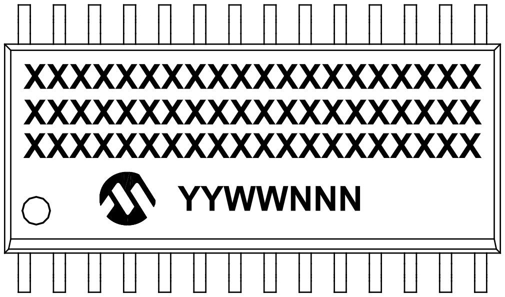
28-Lead PDIP (600 mil)


XXXXXXXXXXXXXXX
XXXXXXXXXXXXXXX
XXXXXXXXXXXXXXX
YYWWNNN
MICROCHIP

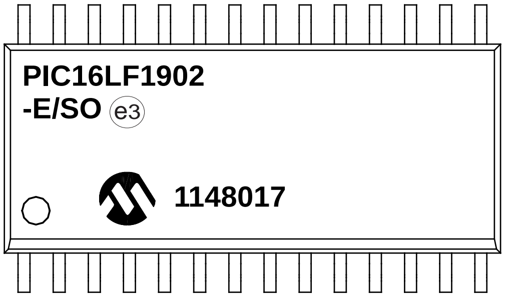
Example


PIC16LF1902-I/P
1148017
MICROCHIP

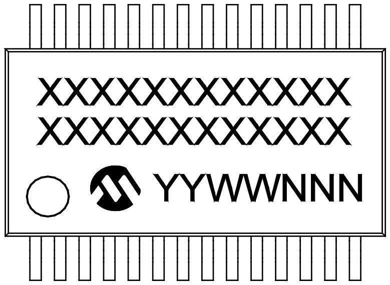
28-Lead SOIC (7.50 mm)


XXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXX
YYWWNNN

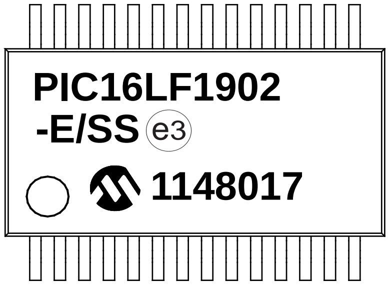
Example


PIC16LF1902
-E/SO e3
1148017

28-Lead SSOP (5.30 mm)


XXXXXXXXXXXX
XXXXXXXXXXXX
YYWWNNN

Example

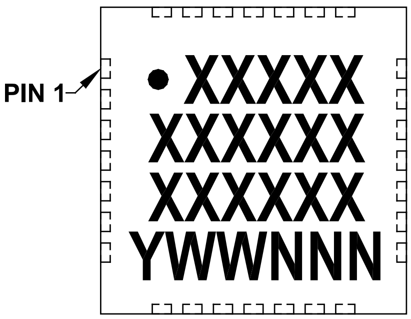

PIC16LF1902
-E/SS e3
1148017

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

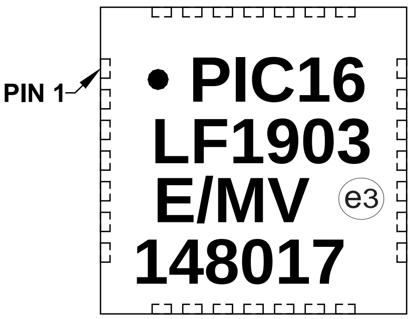
Note: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.

24.2 Package Marking Information

28-Lead UQFN (4x4x0.5 mm)



Example



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

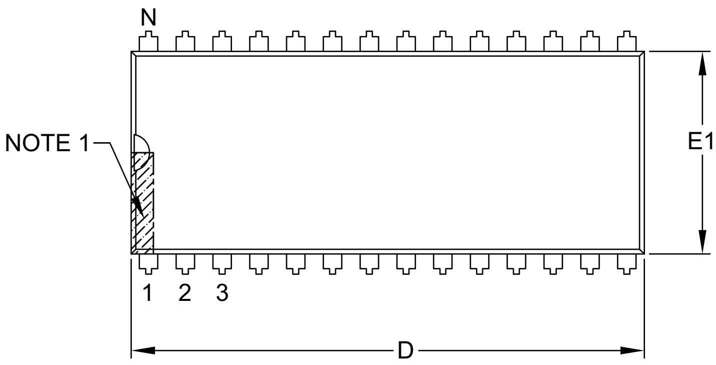
Note: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.

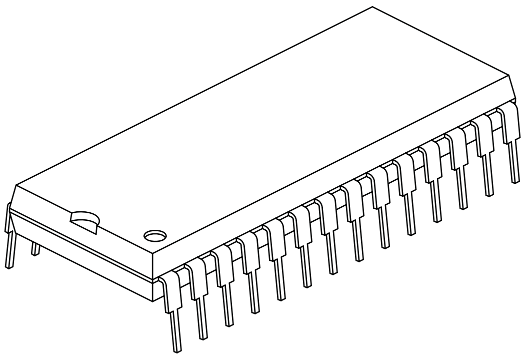
24.3 Package Details

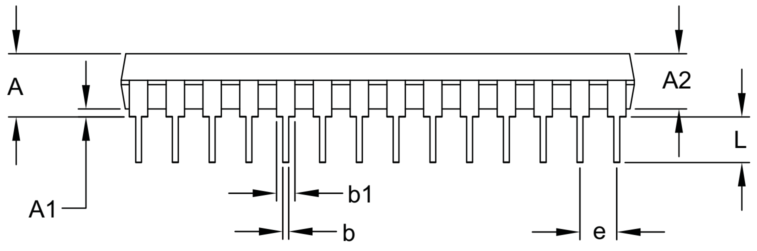
The following sections give the technical details of the packages.

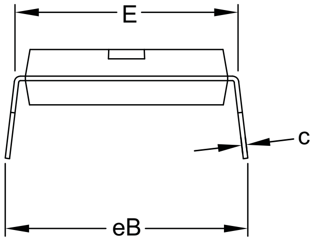
28-Lead Plastic Dual In-Line (P) – 600 mil Body [PDIP]

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









	Units	INCHES		
	Dimension Limits	MIN	NOM	MAX
Number of Pins	N	28		
Pitch	e	.100 BSC		
Top to Seating Plane	A	–	–	.250
Molded Package Thickness	A2	.125	–	.195
Base to Seating Plane	A1	.015	–	–
Shoulder to Shoulder Width	E	.590	–	.625
Molded Package Width	E1	.485	–	.580
Overall Length	D	1.380	–	1.565
Tip to Seating Plane	L	.115	–	.200
Lead Thickness	c	.008	–	.015
Upper Lead Width	b1	.030	–	.070
Lower Lead Width	b	.014	–	.022
Overall Row Spacing §	eB	–	–	.700

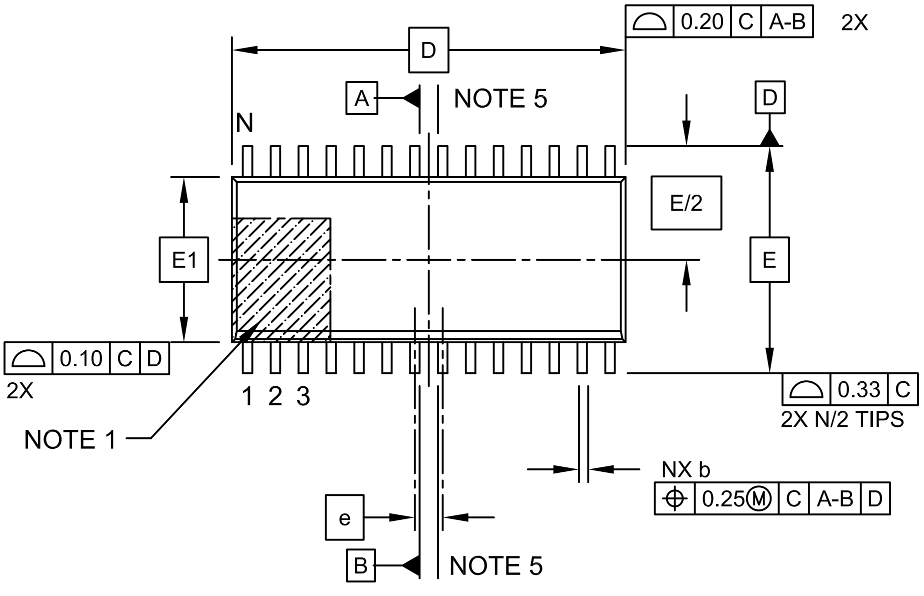
Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

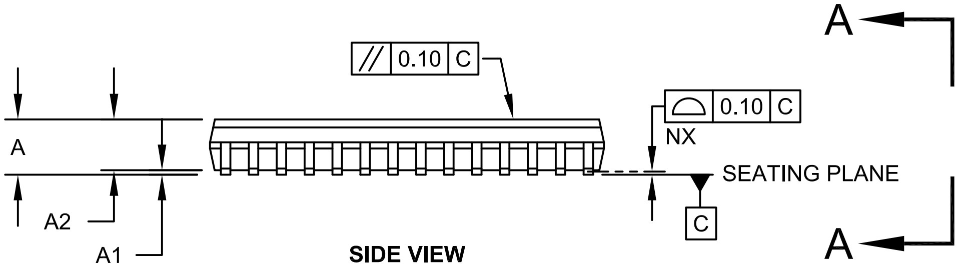
Microchip Technology Drawing C04-079B

28-Lead Plastic Small Outline (SO) - Wide, 7.50 mm Body [SOIC]

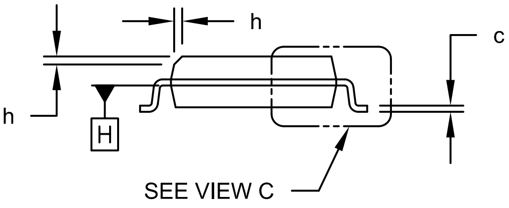
Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging


D
⌓ 0.20 C A-B
2X
A
NOTE 5
N
D
E/2
E1
E
⌓ 0.10 C D
2X
1 2 3
⌓ 0.33 C
2X N/2 TIPS
NOTE 1
NX b
⌖ 0.25Ⓜ C A-B D
e
B
NOTE 5

TOP VIEW


A
// 0.10 C
⌓ 0.10 C
NX
SEATING PLANE
A
A2
A1
C
A

SIDE VIEW

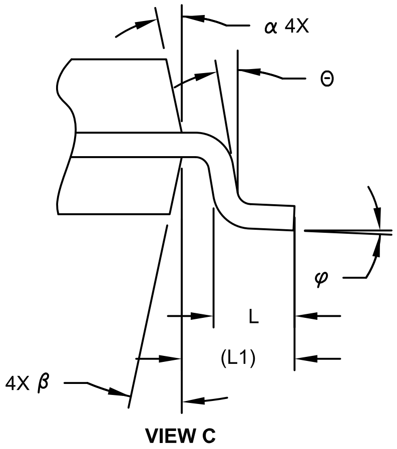

h
c
h
H
SEE VIEW C

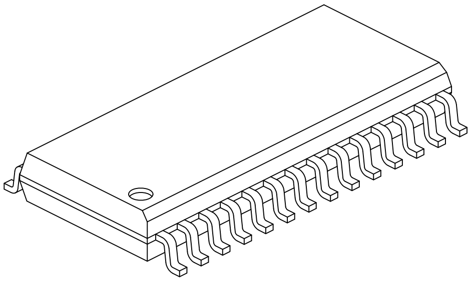
VIEW A-A

Microchip Technology Drawing C04-052C Sheet 1 of 2

28-Lead Plastic Small Outline (SO) - Wide, 7.50 mm Body [SOIC]

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





		MILLIMETERS		
Units				
Dimension Limits		MIN	NOM	MAX
Number of Pins	N	28		
Pitch	e	1.27 BSC		
Overall Height	A	-	-	2.65
Molded Package Thickness	A2	2.05	-	-
Standoff §	A1	0.10	-	0.30
Overall Width	E	10.30 BSC		
Molded Package Width	E1	7.50 BSC		
Overall Length	D	17.90 BSC		
Chamfer (Optional)	h	0.25	-	0.75
Foot Length	L	0.40	-	1.27
Footprint	L1	1.40 REF		
Lead Angle	Θ	0°	-	-
Foot Angle	φ	0°	-	8°
Lead Thickness	c	0.18	-	0.33
Lead Width	b	0.31	-	0.51
Mold Draft Angle Top	α	5°	-	15°
Mold Draft Angle Bottom	β	5°	-	15°

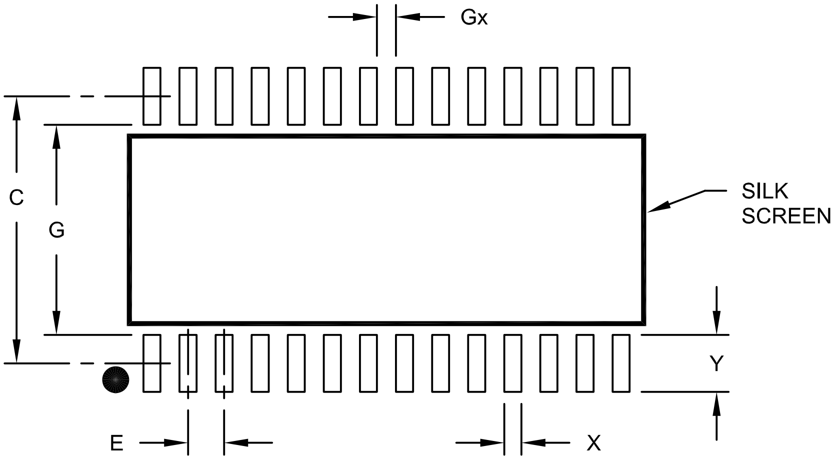
Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic
3. Dimension D does not include mold flash, protrusions or gate burrs, which shall not exceed 0.15 mm per end. Dimension E1 does not include interlead flash or protrusion, which shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.
5. Datums A & B to be determined at Datum H.

Microchip Technology Drawing C04-052C Sheet 2 of 2

28-Lead Plastic Small Outline (SO) - Wide, 7.50 mm Body [SOIC]

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



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E	1.27 BSC		
Contact Pad Spacing	C		9.40	
Contact Pad Width (X28)	X			0.60
Contact Pad Length (X28)	Y			2.00
Distance Between Pads	Gx	0.67		
Distance Between Pads	G	7.40		

Notes:

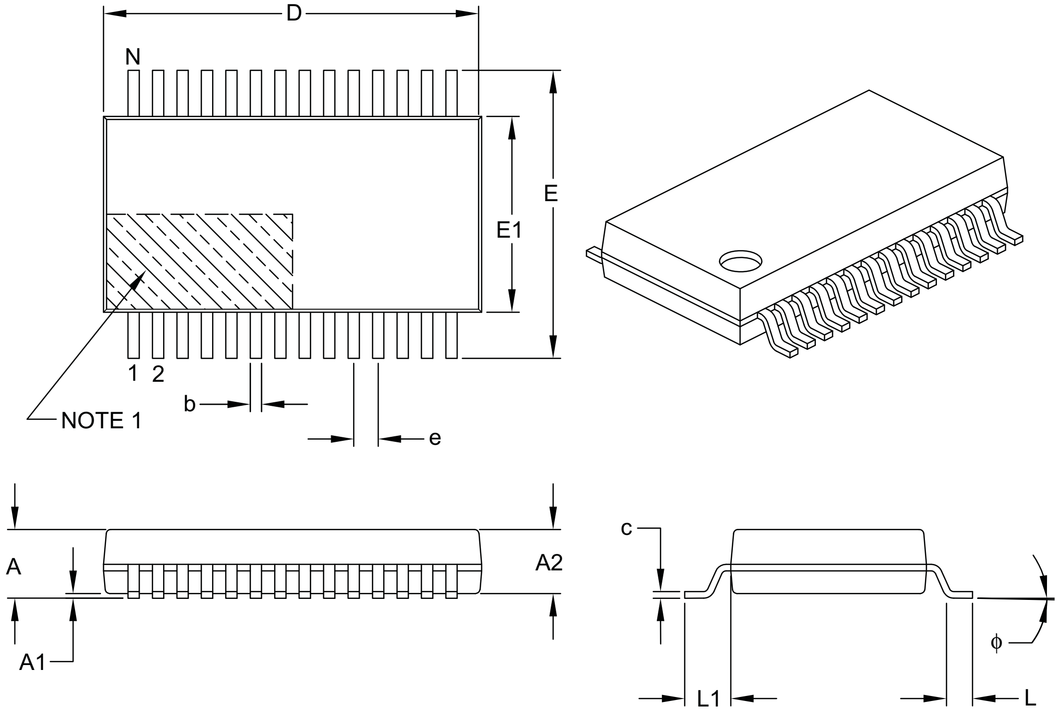
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2052A

28-Lead Plastic Shrink Small Outline (SS) – 5.30 mm Body [SSOP]

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



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		28	
Pitch	e		0.65 BSC	
Overall Height	A	–	–	2.00
Molded Package Thickness	A2	1.65	1.75	1.85
Standoff	A1	0.05	–	–
Overall Width	E	7.40	7.80	8.20
Molded Package Width	E1	5.00	5.30	5.60
Overall Length	D	9.90	10.20	10.50
Foot Length	L	0.55	0.75	0.95
Footprint	L1		1.25 REF	
Lead Thickness	c	0.09	–	0.25
Foot Angle	ϕ	0°	4°	8°
Lead Width	b	0.22	–	0.38

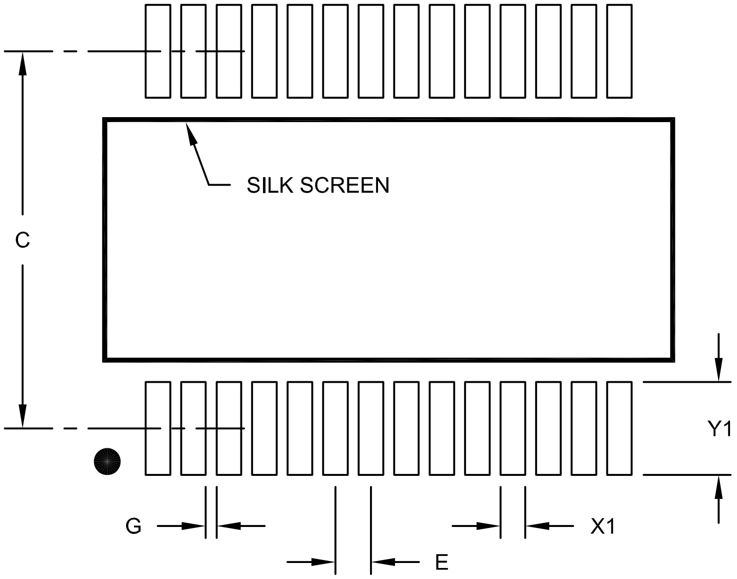
Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.20 mm per side.
3. Dimensioning and tolerancing per ASME Y14.5M.
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-073B

28-Lead Plastic Shrink Small Outline (SS) - 5.30 mm Body [SSOP]

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



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E	0.65 BSC		
Contact Pad Spacing	C		7.20	
Contact Pad Width (X28)	X1			0.45
Contact Pad Length (X28)	Y1			1.75
Distance Between Pads	G	0.20		

Notes:

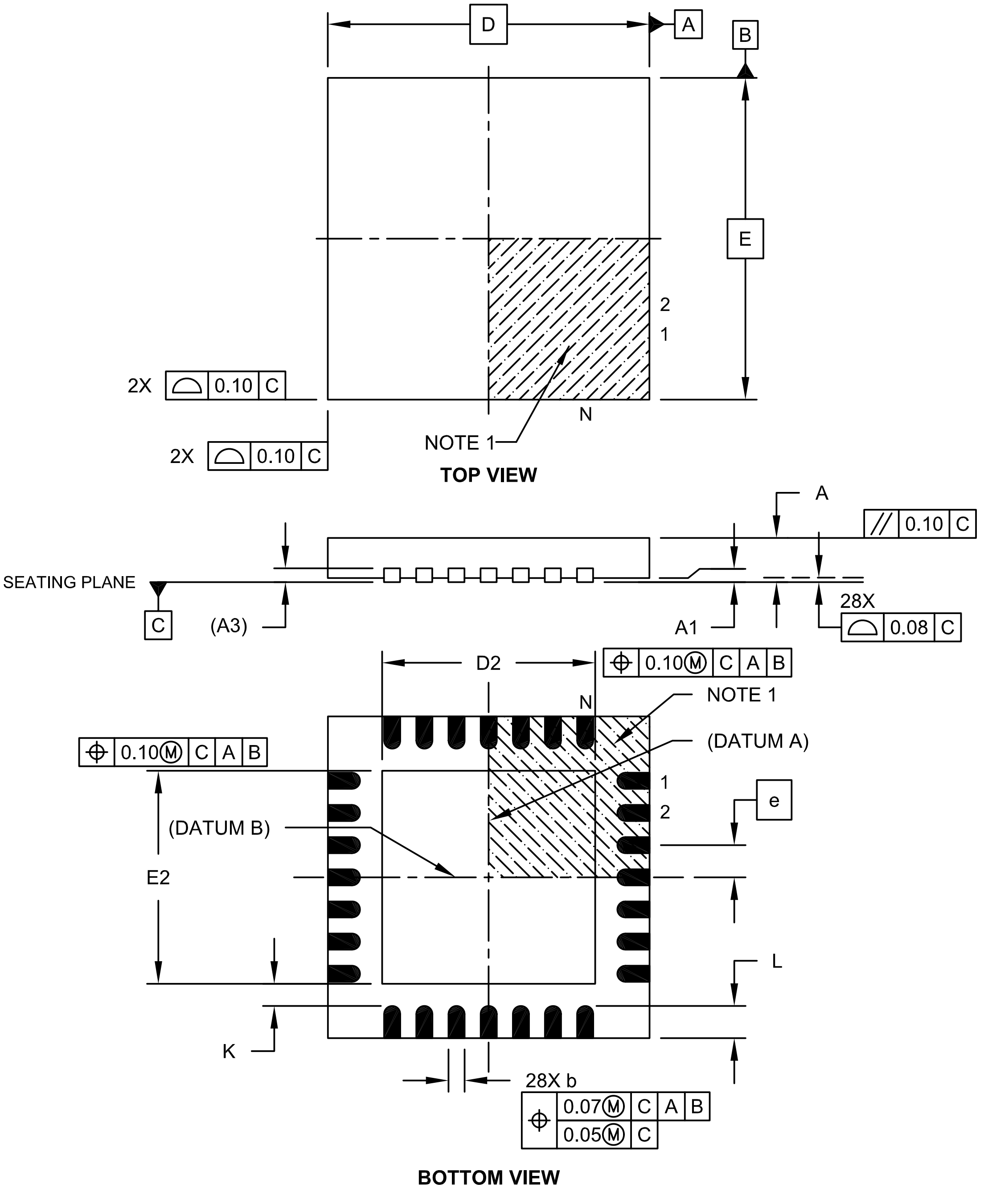
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2073A

28-Lead Plastic Ultra Thin Quad Flat, No Lead Package (MV) – 4x4x0.5 mm Body [UQFN]

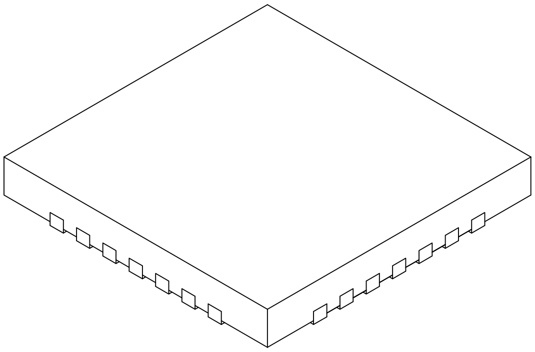
Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging


D
A
B
E
2
1
N
2X 0.10 C
2X 0.10 C
NOTE 1
TOP VIEW
A
0.10 C
SEATING PLANE
C
(A3)
A1
28X
0.08 C
D2
0.10Ⓜ C A B
N
NOTE 1
(DATUM A)
0.10Ⓜ C A B
1
2
e
(DATUM B)
E2
L
K
28X b
0.07Ⓜ C A B
0.05Ⓜ C
BOTTOM VIEW

Microchip Technology Drawing C04-152A Sheet 1 of 2

28-Lead Plastic Ultra Thin Quad Flat, No Lead Package (MV) – 4x4x0.5 mm Body [UQFN]

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



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		28	
Pitch	e		0.40 BSC	
Overall Height	A	0.45	0.50	0.55
Standoff	A1	0.00	0.02	0.05
Contact Thickness	A3		0.127 REF	
Overall Width	E		4.00 BSC	
Exposed Pad Width	E2	2.55	2.65	2.75
Overall Length	D		4.00 BSC	
Exposed Pad Length	D2	2.55	2.65	2.75
Contact Width	b	0.15	0.20	0.25
Contact Length	L	0.30	0.40	0.50
Contact-to-Exposed Pad	K	0.20	-	-

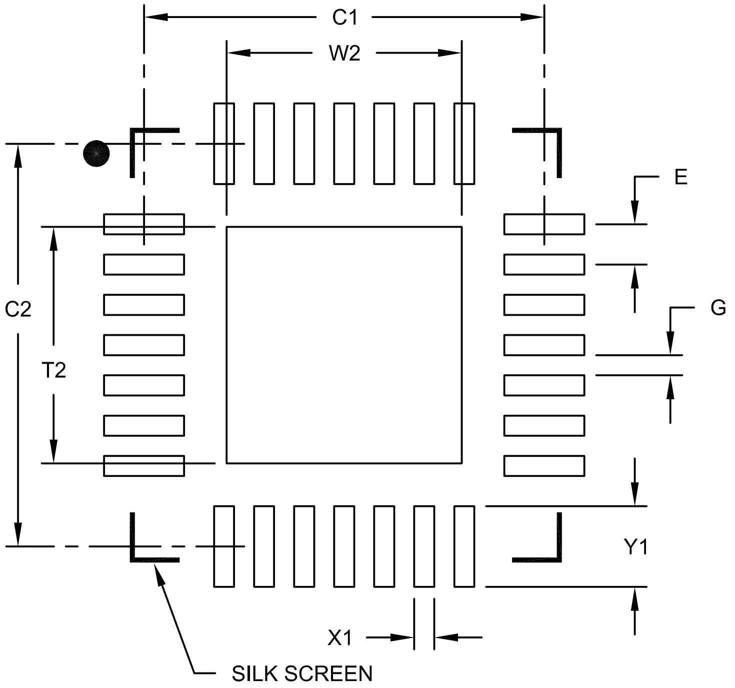
Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated.
3. Dimensioning and tolerancing per ASME Y14.5M.
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-152A Sheet 2 of 2

28-Lead Ultra Thin Plastic Quad Flat, No Lead Package (MV) - 4x4 mm Body [UQFN] With 0.40 mm Contact Length

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



RECOMMENDED LAND PATTERN

		Units	MILLIMETERS		
Dimension Limits			MIN	NOM	MAX
Contact Pitch	E		0.40 BSC		
Optional Center Pad Width	W2				2.35
Optional Center Pad Length	T2				2.35
Contact Pad Spacing	C1			4.00	
Contact Pad Spacing	C2			4.00	
Contact Pad Width (X28)	X1				0.20
Contact Pad Length (X28)	Y1				0.80
Distance Between Pads	G		0.20		

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2152A

APPENDIX A: DATA SHEET REVISION HISTORY

Revision A (01/2011)

Original release.

Revision B (04/2011)

Revised Sections: Flexible Oscillator Structure; Low-Power Features; Electrical Specifications; Changed ULPBOR to LPBOR.

Revision C (07/2014)

Updated Example 3-2, Register 4-2 and Table 5-1; Updated section 6, Oscillator Module, and section 9, Watchdog Timer; Updated table 10-3 and Figure 18-7; Removed Figure 19-1; Updated section 21, Electrical Specifications; Other minor corrections.

Revision D (04/2015)

Updated Equation 15-1; Figures 5-1, 18-7, 21-2, and 21-8; Register 4-2; Sections 4.1, 18.4.5, and 21.0; and Table 5-1, 9-2, 10-4, 21-1, and 21-10.

Revision E (01/2016)

Added ‘Memory’ section; Updated ‘PIC16LF1902/3 Family Types’ table; Other minor corrections.

Revision F (05/2016)

Updated Figure 21-2. Removed parameter 34A in Table 21-10. Removed Table 18-7, LCD Worksheet. Other minor corrections.

THE MICROCHIP WEBSITE

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

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

CUSTOMER CHANGE NOTIFICATION SERVICE

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

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

CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

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

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

Technical support is available through the website at: http://www.microchip.com/support

PRODUCT IDENTIFICATION SYSTEM

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

PART NO.	[X][(1)]	-	X	/XX	XXX
Device	**Tape and Reel Option**		**Temperature Range**	**Package**	**Pattern**

Device:	PIC16LF1902, PIC16LF1903	
Tape and Reel Option:	Blank T	= Standard packaging (tube or tray) = Tape and Reel[(1)]
Temperature Range:	I E	= -40°C to +85°C (Industrial) = -40°C to +125°C (Extended)
Package:	MV P SO SS	= UQFN (4x4x0.5) = PDIP = SOIC = SSOP
Pattern:	QTP, SQTP, Code or Special Requirements (blank otherwise)	

Examples:

a) PIC16LF1902T - I/MV 301
Tape and Reel,
Industrial temperature,
UQFN package,
QTP pattern #301

b) PIC16LF1903 - I/P
Industrial temperature
PDIP package

c) PIC16LF1903 - E/SS
Extended temperature,
SSOP package

Note 1: Tape and Reel identifier only appears in the catalog part number description. This identifier is used for ordering purposes and is not printed on the device package. Check with your Microchip Sales Office for package availability with the Tape and Reel option.

NOTES:

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

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

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

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

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

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

Trademarks

The Microchip name and logo, the Microchip logo, AnyRate, dsPIC, FlashFlex, flexPWR, Heldo, JukeBlox, KeeLoq, KeeLoq logo, Kleer, LANCheck, LINK MD, MediaLB, MOST, MOST logo, MPLAB, OptoLyzer, PIC, PICSTART, PIC32 logo, RightTouch, SpyNIC, SST, SST Logo, SuperFlash and UNI/O are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

ClockWorks, The Embedded Control Solutions Company, ETHERSYNCH, Hyper Speed Control, HyperLight Load, IntelliMOS, mTouch, Precision Edge, and QUIET-WIRE are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, BodyCom, chipKIT, chipKIT logo, CodeGuard, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, EtherGREEN, In-Circuit Serial Programming, ICSP, Inter-Chip Connectivity, JitterBlocker, KleerNet, KleerNet logo, MiWi, motorBench, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, PureSilicon, RightTouch logo, REAL ICE, Ripple Blocker, Serial Quad I/O, SQI, SuperSwitcher, SuperSwitcher II, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

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

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

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

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

© 2011-2016, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

ISBN: 978-1-5224-0546-7

Worldwide Sales and Service

AMERICAS

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

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

Austin, TX
Tel: 512-257-3370

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

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

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

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

Detroit
Novi, MI
Tel: 248-848-4000

Houston, TX
Tel: 281-894-5983

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

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

New York, NY
Tel: 631-435-6000

San Jose, CA
Tel: 408-735-9110

Canada - Toronto
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon

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

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

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

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

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

China - Dongguan
Tel: 86-769-8702-9880

China - Hangzhou
Tel: 86-571-8792-8115
Fax: 86-571-8792-8116

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

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

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

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

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

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

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

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

ASIA/PACIFIC

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

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

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

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

India - Pune
Tel: 91-20-3019-1500

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

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

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

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

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

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

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

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

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

Taiwan - Kaohsiung
Tel: 886-7-213-7828

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

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

EUROPE

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

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

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

Germany - Dusseldorf
Tel: 49-2129-3766400

Germany - Karlsruhe
Tel: 49-721-625370

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

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

Italy - Venice
Tel: 39-049-7625286

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

Poland - Warsaw
Tel: 48-22-3325737

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

Sweden - Stockholm
Tel: 46-8-5090-4654

UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820

07/14/15