

---

---

## 20-Pin Flash, 8-Bit Microcontrollers

---

---

### High-Performance RISC CPU:

- C Compiler Optimized Architecture
- Only 49 Instructions
- Operating Speed:
  - DC – 20 MHz clock input
  - 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

### Flexible Oscillator Structure:

- 16 MHz Internal Oscillator Block:
  - Factory calibrated to  $\pm 1\%$ , typical
  - Software selectable frequency range from 16 MHz to 31 kHz
- 31 kHz Low-Power Internal Oscillator
- Three External Clock modes up to 20 MHz

### Special Microcontroller Features:

- Operating Voltage Range:
  - 1.8V to 3.6V (PIC16LF1507)
  - 2.3V to 5.5V (PIC16F1507)
- Self-Programmable under Software Control
- Power-on Reset (POR)
- Power-up Timer (PWRT)
- Programmable Low-Power Brown-out Reset (LPBOR)
- Extended Watchdog Timer (WDT):
  - Programmable period from 1 ms to 256s
- Programmable Code Protection
- In-Circuit Serial Programming™ (ICSP™) via Two Pins
- Enhanced Low-Voltage Programming (LVP)
- In-Circuit Debug (ICD) via Two Pins
- Power-Saving Sleep mode:
  - Low-Power Sleep mode
  - Low-Power BOR (LPBOR)
- Integrated Temperature Indicator
- 128 Bytes High-Endurance Flash
  - 100,000 write Flash endurance (minimum)

### Memory:

- 2 Kwords Linear Program Memory Addressing
- 128 bytes Linear Data Memory Addressing
- High-Endurance Flash Data Memory (HEF)
  - 128 bytes if nonvolatile data storage
  - 100k erase/write cycles

### eXtreme Low-Power (XLP) Features (PIC16LF1507):

- Sleep Current:
  - 20 nA @ 1.8V, typical
- Watchdog Timer Current:
  - 260 nA @ 1.8V, typical
- Operating Current:
  - 30  $\mu$ A/MHz @ 1.8V, typical

### Peripheral Features:

- Analog-to-Digital Converter (ADC):
  - 10-bit resolution
  - 12 external channels
  - Two internal channels:
    - Fixed Voltage Reference
    - Temperature Indicator channel
  - Auto acquisition capability
  - Conversion available during Sleep
- Voltage Reference:
  - 1.024V Fixed Voltage Reference (FVR) with 1x, 2x and 4x Gain output levels
- 18 I/O Pins (1 Input-only Pin):
  - High current sink/source 25 mA/25 mA
  - Individually programmable weak pull-ups
  - Individually programmable Interrupt-on-Change (IOC) pins
- Timer0: 8-Bit Timer/Counter with 8-Bit Programmable Prescaler
- Enhanced Timer1:
  - 16-bit timer/counter with prescaler
  - External Gate Input mode
- Timer2: 8-Bit Timer/Counter with 8-Bit Period Register, Prescaler and Postscaler
- Four 10-bit PWM modules

# PIC16(L)F1507

## Peripheral Features (Continued):

- Two Configurable Logic Cell (CLC) modules:
  - 16 selectable input source signals
  - Four inputs per module
  - Software control of combinational/sequential logic/state/clock functions
- Numerically Controlled Oscillator (NCO):
  - 20-bit accumulator
  - 16-bit increment
  - True linear frequency control
  - High-speed clock input
- Selectable Output modes
  - Fixed Duty Cycle (FDC) mode
  - Pulse Frequency (PF) mode
- Complementary Waveform Generator (CWG):
  - Eight selectable signal sources
  - Selectable falling and rising edge dead-band control
  - Polarity control
  - Four auto-shutdown sources
  - Multiple input sources: PWM, CLC, NCO

## PIC12(L)F1501/PIC16(L)F150X FAMILY TYPES

Device	Data Sheet Index	Program Memory Flash (words)	Data SRAM (bytes)	I/O's <sup>(2)</sup>	10-bit ADC (ch)	Comparators	DAC	Timers (8/16-bit)	PWM	EUSART	MSSP (I <sup>2</sup> C/SPI)	CWG	CLC	NCO	Debug <sup>(1)</sup>	XLP
PIC12(L)F1501	(1)	1024	64	6	4	1	1	2/1	4	—	—	1	2	1	H	—
PIC16(L)F1503	(2)	2048	128	12	8	2	1	2/1	4	—	1	1	2	1	H	—
PIC16(L)F1507	(3)	2048	128	18	12	—	—	2/1	4	—	—	1	2	1	H	—
PIC16(L)F1508	(4)	4096	256	18	12	2	1	2/1	4	1	1	1	4	1	I/H	Y
PIC16(L)F1509	(4)	8192	512	18	12	2	1	2/1	4	1	1	1	4	1	I/H	Y

**Note 1:** Debugging Methods: (I) - Integrated on Chip; (H) - using Debug Header; (E) - using Emulation Header.

**2:** One pin is input-only.

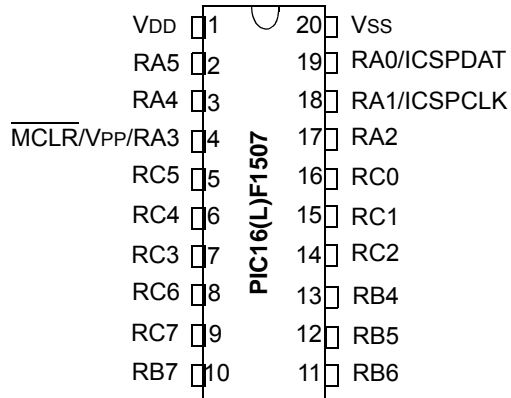
**Data Sheet Index:** (Unshaded devices are described in this document.)

- 1: DS40001615 [PIC12\(L\)F1501 Data Sheet, 8-Pin Flash, 8-bit Microcontrollers.](#)
- 2: DS40001607 [PIC16\(L\)F1503 Data Sheet, 14-Pin Flash, 8-bit Microcontrollers.](#)
- 3: DS40001586 [PIC16\(L\)F1507 Data Sheet, 20-Pin Flash, 8-bit Microcontrollers.](#)
- 4: DS40001609 [PIC16\(L\)F1508/9 Data Sheet, 20-Pin Flash, 8-bit Microcontrollers.](#)

**Note:** For other small form-factor package availability and marking information, please visit <http://www.microchip.com/packaging> or contact your local sales office.

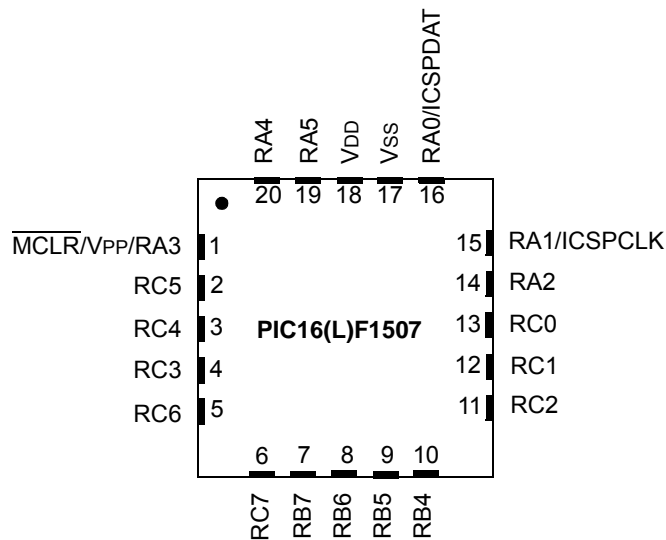
## PIN DIAGRAMS

### 20-pin PDIP, SOIC, SSOP



**Note:** See [Table 1](#) for location of all peripheral functions.

### 20-pin QFN, UQFN



**Note 1:** See [Table 1](#) for location of all peripheral functions.

**2:** It is recommended that the exposed bottom pad be connected to VSS.

# PIC16(L)F1507

## PIN ALLOCATION TABLE

TABLE 1: 20-PIN ALLOCATION TABLE (PIC16(L)F1507)

I/O	20-Pin PDIP/SOIC/SSOP	20-Pin QFN/UQFN	ADC	Reference	CWG	NCO	CLC	Timers	PWM	Interrupt	Pull-up	Basic
RA0	19	16	AN0	—	—	—	—	—	—	IOC	Y	ICSPDAT
RA1	18	15	AN1	VREF+	—	—	—	—	—	IOC	Y	ICSPCLK
RA2	17	14	AN2	—	CWG1FLT	—	CLC1	T0CKI	PWM3	INT/ IOC	Y	—
RA3	4	1	—	—	—	—	CLC1IN0	—	—	IOC	Y	MCLR VPP
RA4	3	20	AN3	—	—	—	—	T1G	—	IOC	Y	CLKOUT
RA5	2	19	—	—	—	NCO1CLK	—	T1CKI	—	IOC	Y	CLKIN
RB4	13	10	AN10	—	—	—	—	—	—	IOC	Y	—
RB5	12	9	AN11	—	—	—	—	—	—	IOC	Y	—
RB6	11	8	—	—	—	—	—	—	—	IOC	Y	—
RB7	10	7	—	—	—	—	—	—	—	IOC	Y	—
RC0	16	13	AN4	—	—	—	CLC2	—	—	—	—	—
RC1	15	12	AN5	—	—	NCO1	—	—	PWM4	—	—	—
RC2	14	11	AN6	—	—	—	—	—	—	—	—	—
RC3	7	4	AN7	—	—	—	CLC2IN0	—	PWM2	—	—	—
RC4	6	3	—	—	CWG1B	—	CLC2IN1	—	—	—	—	—
RC5	5	2	—	—	CWG1A	—	CLC1 <sup>(1)</sup>	—	PWM1	—	—	—
RC6	8	5	AN8	—	—	NCO1 <sup>(1)</sup>	—	—	—	—	—	—
RC7	9	6	AN9	—	—	—	CLC1IN1	—	—	—	—	—
VDD	1	18	—	—	—	—	—	—	—	—	—	VDD
VSS	20	17	—	—	—	—	—	—	—	—	—	VSS

Note 1: Alternate pin function selected with the APFCON (Register 11-1) register.

## TABLE OF CONTENTS

1.0	Device Overview .....	7
2.0	Enhanced Mid-Range CPU .....	11
3.0	Memory Organization .....	13
4.0	Device Configuration .....	37
5.0	Oscillator Module.....	42
6.0	Resets .....	51
7.0	Interrupts .....	59
8.0	Power-Down Mode (Sleep) .....	72
9.0	Watchdog Timer (WDT) .....	75
10.0	Flash Program Memory Control .....	79
11.0	I/O Ports .....	95
12.0	Interrupt-On-Change .....	108
13.0	Fixed Voltage Reference (FVR) .....	113
14.0	Temperature Indicator Module .....	115
15.0	Analog-to-Digital Converter (ADC) Module .....	117
16.0	Timer0 Module .....	131
17.0	Timer1 Module with Gate Control.....	134
18.0	Timer2 Module .....	145
19.0	Pulse-Width Modulation (PWM) Module .....	148
20.0	Configurable Logic Cell (CLC).....	154
21.0	Numerically Controlled Oscillator (NCO) Module .....	170
22.0	Complementary Waveform Generator (CWG) Module .....	177
23.0	In-Circuit Serial Programming™ (ICSP™) .....	189
24.0	Instruction Set Summary .....	191
25.0	Electrical Specifications.....	205
26.0	DC and AC Characteristics Graphs and Charts .....	227
27.0	Development Support.....	257
28.0	Packaging Information.....	261
Appendix A: Data Sheet Revision History.....		274
The Microchip Website .....		275
Customer Change Notification Service .....		275
Customer Support.....		275
Product Identification System .....		276

## 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](mailto:docerrors@microchip.com) or fax the **Reader Response Form** in the back of this data sheet to (480) 792-4150. We welcome your feedback.

### Most Current Data Sheet

To obtain the most up-to-date version of this data sheet, please register at our Worldwide 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., DS30000A is version A of document DS30000).

### Errata

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

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

- Microchip's Worldwide 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](http://www.microchip.com) to receive the most current information on all of our products.

## 1.0 DEVICE OVERVIEW

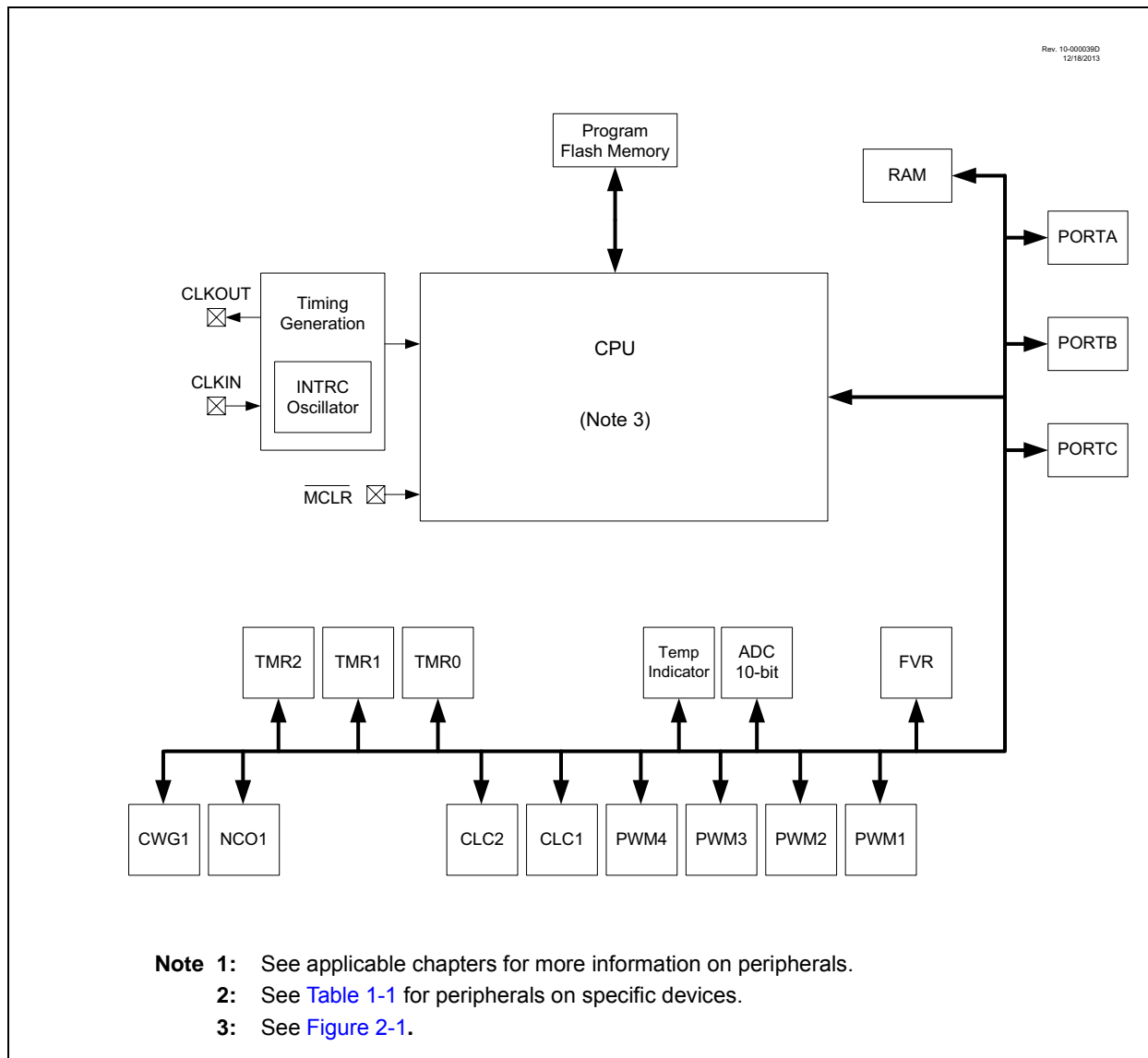
The block diagram of these devices are shown in [Figure 1-1](#), the available peripherals are shown in [Table 1-1](#), and the pinout descriptions are shown in [Table 1-2](#).

**TABLE 1-1: DEVICE PERIPHERAL SUMMARY**

Peripheral	PIC12(L)F1501	PIC16(L)F1503	PIC16(L)F1507	PIC16(L)F1508	PIC16(L)F1509
Analog-to-Digital Converter (ADC)	•	•	•	•	•
Complementary Wave Generator (CWG)	•	•	•	•	•
Digital-to-Analog Converter (DAC)	•	•		•	•
Enhanced Universal Synchronous/Asynchronous Receiver/Transmitter (EUSART)				•	•
Fixed Voltage Reference (FVR)	•	•	•	•	•
Numerically Controlled Oscillator (NCO)	•	•	•	•	•
Temperature Indicator	•	•	•	•	•
Comparators					
	C1	•	•	•	•
	C2		•	•	•
Configurable Logic Cell (CLC)					
	CLC1	•	•	•	•
	CLC2	•	•	•	•
	CLC3			•	•
	CLC4			•	•
Master Synchronous Serial Ports					
	MSSP1		•	•	•
PWM Modules					
	PWM1	•	•	•	•
	PWM2	•	•	•	•
	PWM3	•	•	•	•
	PWM4	•	•	•	•
Timers					
	Timer0	•	•	•	•
	Timer1	•	•	•	•
	Timer2	•	•	•	•

# PIC16(L)F1507

FIGURE 1-1: PIC16(L)F1507 BLOCK DIAGRAM





**TABLE 1-2: PIC16(L)F1507 PINOUT DESCRIPTION**

Name	Function	Input Type	Output Type	Description
RA0/AN0/ICSPDAT	RA0	TTL	CMOS	General purpose I/O.
	AN0	AN	—	A/D Channel input.
	ICSPDAT	ST	CMOS	ICSP™ Data I/O.
RA1/AN1/VREF+/ICSPCLK	RA1	TTL	CMOS	General purpose I/O.
	AN1	AN	—	A/D Channel input.
	VREF+	AN	—	A/D Positive Voltage Reference input.
	ICSPCLK	ST	—	Serial Programming Clock.
RA2/AN2/T0CKI/INT/PWM3/CLC1 <sup>(1)</sup> /CWG1FLT	RA2	ST	CMOS	General purpose I/O.
	AN2	AN	—	A/D Channel input.
	T0CKI	ST	—	Timer0 clock input.
	INT	ST	—	External interrupt.
	PWM3	—	CMOS	Pulse Width Module source output.
	CLC1	—	CMOS	Configurable Logic Cell source output.
RA3/CLC1IN0/VPP/MCLR	RA3	TTL	—	General purpose input.
	CLC1IN0	ST	—	Configurable Logic Cell source input.
	VPP	HV	—	Programming voltage.
	MCLR	ST	—	Master Clear with internal pull-up.
RA4/AN3/CLKOUT/T1G	RA4	TTL	CMOS	General purpose I/O.
	AN3	AN	—	A/D Channel input.
	CLKOUT	—	CMOS	Fosc/4 output.
	T1G	ST	—	Timer1 Gate input.
RA5/CLKIN/T1CKI/NC01CLK	RA5	TTL	CMOS	General purpose I/O.
	CLKIN	CMOS	—	External clock input (EC mode).
	T1CKI	ST	—	Timer1 clock input.
	NC01CLK	ST	—	Numerically Controlled Oscillator Clock source input.
RB4/AN10	RB4	TTL	CMOS	General purpose I/O.
	AN10	AN	—	A/D Channel input.
RB5/AN11	RB5	TTL	CMOS	General purpose I/O.
	AN11	AN	—	A/D Channel input.
RB6	RB6	TTL	CMOS	General purpose I/O.
RB7	RB7	TTL	CMOS	General purpose I/O.
RC0/AN4/CLC2	RC0	TTL	CMOS	General purpose I/O.
	AN4	AN	—	A/D Channel input.
	CLC2	—	CMOS	Configurable Logic Cell source output.
RC1/AN5/PWM4/NC01 <sup>(1)</sup>	RC1	TTL	CMOS	General purpose I/O.
	AN5	AN	—	A/D Channel input.
	PWM4	—	CMOS	Pulse Width Module source output.
	NC01	—	CMOS	Numerically Controlled Oscillator is source output.
RC2/AN6	RC2	TTL	CMOS	General purpose I/O.
	AN6	AN	—	A/D Channel input.

**Legend:** AN = Analog input or output    CMOS = CMOS compatible input or output    OD = Open Drain  
TTL = TTL compatible input    ST = Schmitt Trigger input with CMOS levels    I<sup>2</sup>C™ = Schmitt Trigger input with I<sup>2</sup>C levels  
HV = High Voltage    XTAL = Crystal

**Note 1:** Alternate pin function selected with the APFCON (Register 11-1) register.

# PIC16(L)F1507

**TABLE 1-2: PIC16(L)F1507 PINOUT DESCRIPTION (CONTINUED)**

Name	Function	Input Type	Output Type	Description
RC3/AN7/PWM2/CLC2IN0	RC3	TTL	CMOS	General purpose I/O.
	AN7	AN	—	A/D Channel input.
	PWM2	—	CMOS	Pulse Width Module source output.
	CLC2IN0	ST	—	Configurable Logic Cell source input.
RC4/CLC2IN1/CWG1B	RC4	TTL	CMOS	General purpose I/O.
	CLC2IN1	ST	—	Configurable Logic Cell source input.
	CWG1B	—	CMOS	CWG complementary output.
RC5/PWM1/CLC1 <sup>(1)</sup> /CWG1A	RC5	TTL	CMOS	General purpose I/O.
	PWM1	—	CMOS	PWM output.
	CLC1	—	CMOS	Configurable Logic Cell source output.
	CWG1A	—	CMOS	CWG primary output.
RC6/AN8/NCO1 <sup>(1)</sup>	RC6	TTL	CMOS	General purpose I/O.
	AN8	AN	—	A/D Channel input.
	NCO1	—	CMOS	Numerically Controlled Oscillator source output.
RC7/AN9/CLC1IN1	RC7	TTL	CMOS	General purpose I/O.
	AN8	AN	—	A/D Channel input.
	CLC1IN1	ST	—	Configurable Logic Cell source input.
VDD	VDD	Power	—	Positive supply.
VSS	VSS	Power	—	Ground reference.

**Legend:** AN = Analog input or output    CMOS = CMOS compatible input or output    OD = Open Drain  
TTL = TTL compatible input    ST = Schmitt Trigger input with CMOS levels    I<sup>2</sup>C™ = Schmitt Trigger input with I<sup>2</sup>C levels  
HV = High Voltage    XTAL = Crystal

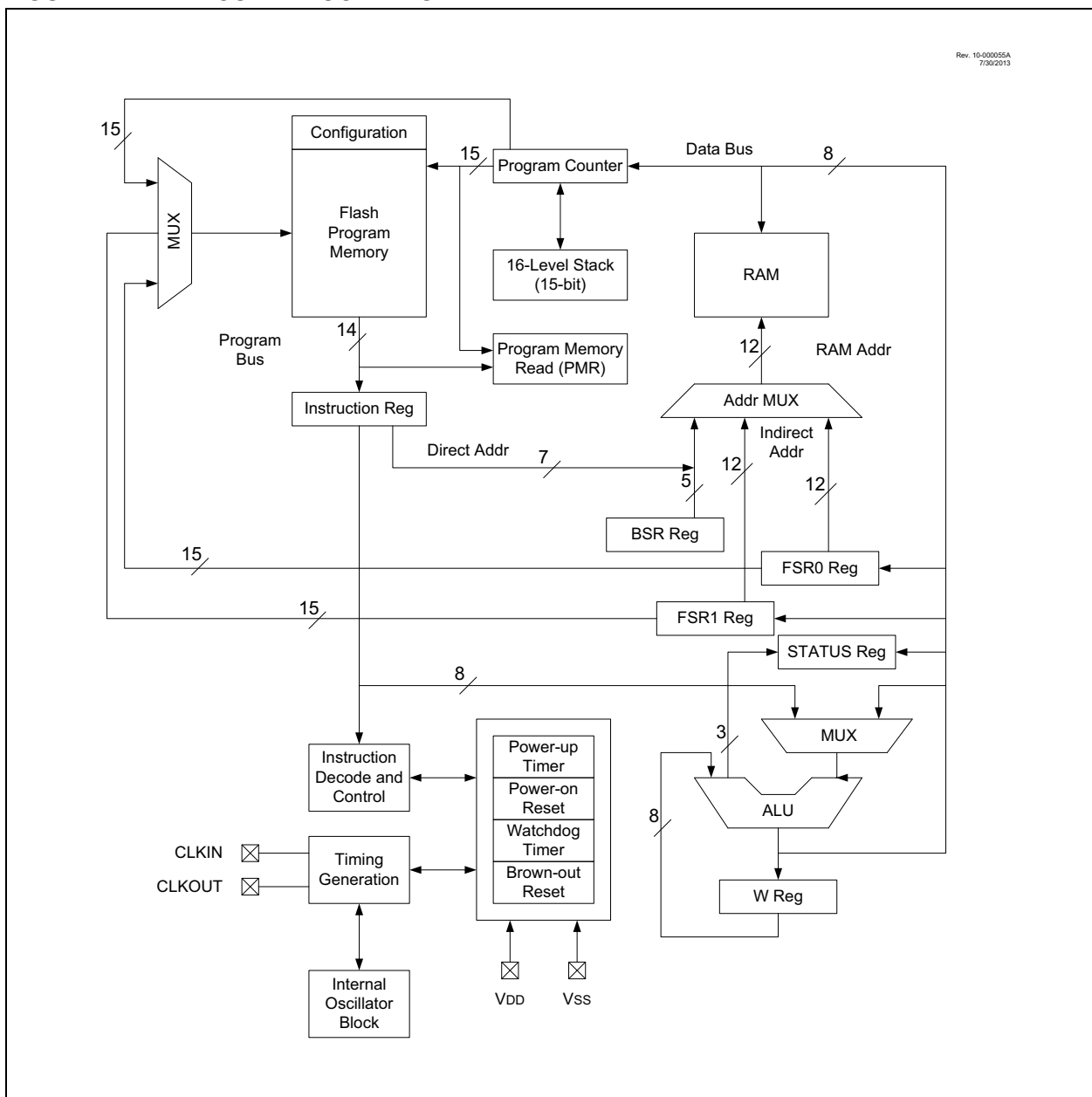
**Note 1:** Alternate pin function selected with the APFCON ([Register 11-1](#)) register.

## 2.0 ENHANCED MID-RANGE CPU

This family of devices contain 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

**FIGURE 2-1: CORE BLOCK DIAGRAM**



# PIC16(L)F1507

---

## 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 a hardware 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 3.5 “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 3.6 “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 24.0 “Instruction Set Summary”](#) for more details.

## 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 a 32K x 14 program memory space. [Table 3-1](#) shows the memory sizes implemented. 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 [Figure 3-1](#)).

### 3.2 High-Endurance Flash

This device has a 128 byte section of high-endurance program Flash memory (PFM) in lieu of data EEPROM. This area is especially well suited for nonvolatile data storage that is expected to be updated frequently over the life of the end product. See [Section 10.2 “Flash Program Memory Overview”](#) for more information on writing data to PFM. See [Section 3.2.1.2 “Indirect Read with FSR”](#) for more information about using the FSR registers to read byte data stored in PFM.

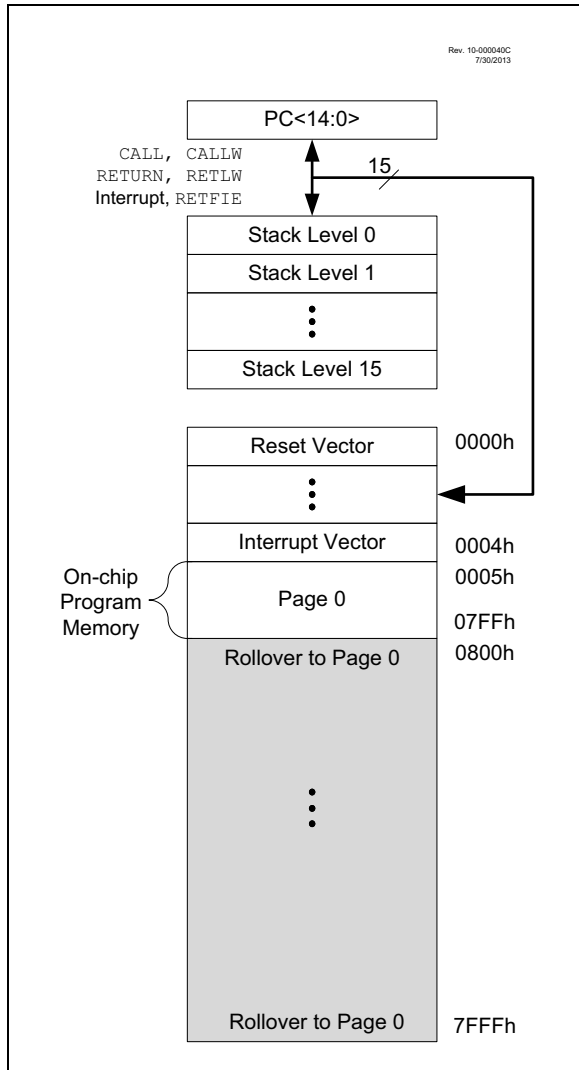
**TABLE 3-1: DEVICE SIZES AND ADDRESSES**

Device	Program Memory Space (Words)	Last Program Memory Address	High-Endurance Flash Memory Address Range <sup>(1)</sup>
PIC16LF1507 PIC16F1507	2,048	07FFh	0780h-07FFh

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

# PIC16(L)F1507

**FIGURE 3-1: PROGRAM MEMORY MAP AND STACK FOR PIC16(L)F1507**



## 3.2.1 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.

### 3.2.1.1 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.

## 3.2.1.2 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` operator will set bit<7> if a label points to a location in program memory.

### EXAMPLE 3-2: ACCESSING PROGRAM MEMORY VIA FSR

```
constants
  DW DATA0          ;First constant
  DW DATA1          ;Second constant
  DW DATA2
  DW DATA3
my_function
  ;... LOTS OF CODE...
  MOVLW  DATA_INDEX
  ADDLW  LOW constants
  MOVWF  FSR1L
  MOVLW  HIGH constants;MSb sets
                        automatically
  MOVWF  FSR1H
  BTFSC  STATUS, C    ;carry from ADDLW?
  INCF   FSR1h, f     ;yes
  MOVIW  0[FSR1]
;THE PROGRAM MEMORY IS IN W
```

# PIC16(L)F1507

## 3.3 Data Memory Organization

The data memory is partitioned in 32 memory banks with 128 bytes in a bank. Each bank consists of (Figure 3-2):

- 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 3.6 "Indirect Addressing"](#) for more information.

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

### 3.3.1 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 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



## 3.3.1.1 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 TO and 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 24.0 "Instruction Set Summary"](#)).

**Note 1:** The `C` and `DC` bits operate as Borrow and 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
—	—	—	TO	PD	Z	DC <sup>(1)</sup>	C <sup>(1)</sup>
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 **TO:** Time-Out bit  
 1 = After power-up, `CLRWDT` instruction or `SLEEP` instruction  
 0 = A WDT time-out occurred

bit 3 **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/Digit Borrow bit (`ADDWF`, `ADDLW`, `SUBLW`, `SUBWF` instructions)<sup>(1)</sup>  
 1 = A carry-out from the 4th low-order bit of the result occurred  
 0 = No carry-out from the 4th low-order bit of the result

bit 0 **C:** Carry/Borrow bit<sup>(1)</sup> (`ADDWF`, `ADDLW`, `SUBLW`, `SUBWF` instructions)<sup>(1)</sup>  
 1 = A carry-out from the Most Significant bit of the result occurred  
 0 = No carry-out from the Most Significant bit of the result occurred

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

# PIC16(L)F1507

## 3.3.2 SPECIAL FUNCTION REGISTER

The Special Function Registers 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.

## 3.3.3 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).

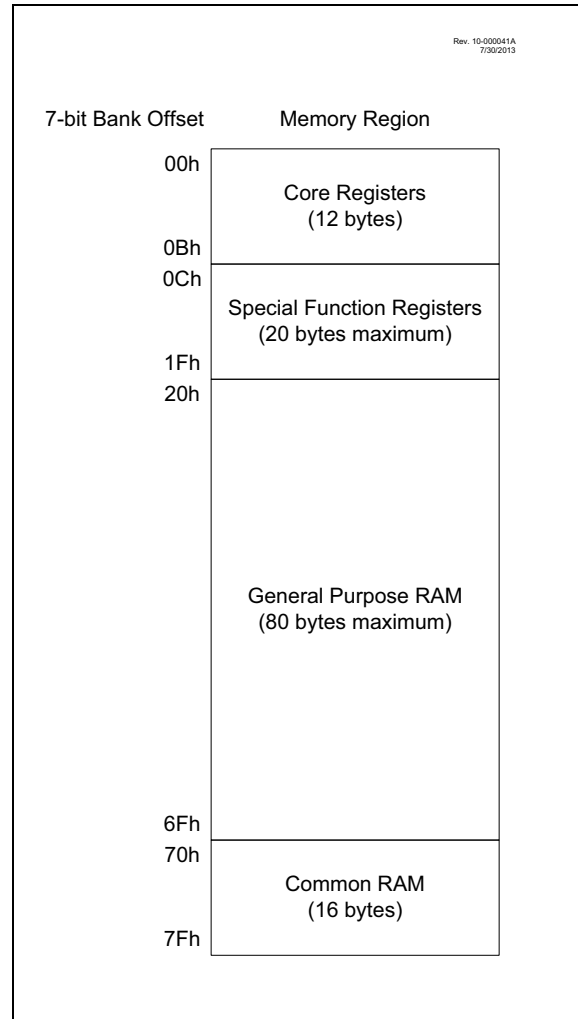
### 3.3.3.1 Linear Access to GPR

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 3.6.2 “Linear Data Memory”](#) for more information.

## 3.3.4 COMMON RAM

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

**FIGURE 3-2: BANKED MEMORY PARTITIONING**



### 3.3.5 DEVICE MEMORY MAPS

The memory maps for Bank 0 through Bank 31 are shown in the tables in this section.

**TABLE 3-3: PIC16(L)F1507 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	WPUA	28Ch	—	30Ch	—	38Ch	—
00Dh	PORTB	08Dh	TRISB	10Dh	LATB	18Dh	ANSELB	20Dh	WPUB	28Dh	—	30Dh	—	38Dh	—
00Eh	PORTC	08Eh	TRISC	10Eh	LATC	18Eh	ANSELC	20Eh	—	28Eh	—	30Eh	—	38Eh	—
00Fh	—	08Fh	—	10Fh	—	18Fh	—	20Fh	—	28Fh	—	30Fh	—	38Fh	—
010h	—	090h	—	110h	—	190h	—	210h	—	290h	—	310h	—	390h	—
011h	PIR1	091h	PIE1	111h	—	191h	PMADRL	211h	—	291h	—	311h	—	391h	IOCAP
012h	PIR2	092h	PIE2	112h	—	192h	PMADRH	212h	—	292h	—	312h	—	392h	IOCAN
013h	PIR3	093h	PIE3	113h	—	193h	PMDATL	213h	—	293h	—	313h	—	393h	IOCAF
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	WDTCN	117h	FVRCON	197h	VREGCON	217h	—	297h	—	317h	—	397h	—
018h	T1CON	098h	—	118h	—	198h	—	218h	—	298h	—	318h	—	398h	—
019h	T1GCON	099h	OSCCON	119h	—	199h	—	219h	—	299h	—	319h	—	399h	—
01Ah	TMR2	09Ah	OSCSTAT	11Ah	—	19Ah	—	21Ah	—	29Ah	—	31Ah	—	39Ah	—
01Bh	PR2	09Bh	ADRESL	11Bh	—	19Bh	—	21Bh	—	29Bh	—	31Bh	—	39Bh	—
01Ch	T2CON	09Ch	ADRESH	11Ch	—	19Ch	—	21Ch	—	29Ch	—	31Ch	—	39Ch	—
01Dh	—	09Dh	ADCON0	11Dh	APFCON	19Dh	—	21Dh	—	29Dh	—	31Dh	—	39Dh	—
01Eh	—	09Eh	ADCON1	11Eh	—	19Eh	—	21Eh	—	29Eh	—	31Eh	—	39Eh	—
01Fh	—	09Fh	ADCON2	11Fh	—	19Fh	—	21Fh	—	29Fh	—	31Fh	—	39Fh	—
020h	General Purpose Register 80 Bytes	0A0h	General Purpose Register 32 Bytes	120h	Unimplemented Read as '0'	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'
		0BFh 0C0h	Unimplemented Read as '0'												
06Fh	Common RAM	0EFh	Common RAM (Accesses 70h – 7Fh)	16Fh	Common RAM (Accesses 70h – 7Fh)	1EFh	Common RAM (Accesses 70h – 7Fh)	26Fh	Common RAM (Accesses 70h – 7Fh)	2EFh	Common RAM (Accesses 70h – 7Fh)	36Fh	Common RAM (Accesses 70h – 7Fh)	3EFh	Common RAM (Accesses 70h – 7Fh)
070h		0F0h		170h		1F0h		270h		2F0h		370h		3F0h	
07Fh		0FFh		17Fh		1FFh		27Fh		2FFh		37Fh		3FFh	

**Legend:** ■ = Unimplemented data memory locations, read as '0'

**TABLE 3-3: PIC16(L)F1507 MEMORY MAP (CONTINUED)**

BANK 8		BANK 9		BANK 10		BANK 11		BANK 12		BANK 13		BANK 14		BANK 15	
400h	Core Registers (Table 3-2)	480h	Core Registers (Table 3-2)	500h	Core Registers (Table 3-2)	580h	Core Registers (Table 3-2)	600h	Core Registers (Table 3-2)	680h	Core Registers (Table 3-2)	700h	Core Registers (Table 3-2)	780h	Core Registers (Table 3-2)
40Bh	—	48Bh	—	50Bh	—	58Bh	—	60Bh	—	68Bh	—	70Bh	—	78Bh	—
40Ch	—	48Ch	—	50Ch	—	58Ch	—	60Ch	—	68Ch	—	70Ch	—	78Ch	—
40Dh	—	48Dh	—	50Dh	—	58Dh	—	60Dh	—	68Dh	—	70Dh	—	78Dh	—
40Eh	—	48Eh	—	50Eh	—	58Eh	—	60Eh	—	68Eh	—	70Eh	—	78Eh	—
40Fh	—	48Fh	—	50Fh	—	58Fh	—	60Fh	—	68Fh	—	70Fh	—	78Fh	—
410h	—	490h	—	510h	—	590h	—	610h	—	690h	—	710h	—	790h	—
411h	—	491h	—	511h	—	591h	—	611h	PWM1DCL	691h	CWG1DBR	711h	—	791h	—
412h	—	492h	—	512h	—	592h	—	612h	PWM1DCH	692h	CWG1DBF	712h	—	792h	—
413h	—	493h	—	513h	—	593h	—	613h	PWM1CON	693h	CWG1CON0	713h	—	793h	—
414h	—	494h	—	514h	—	594h	—	614h	PWM2DCL	694h	CWG1CON1	714h	—	794h	—
415h	—	495h	—	515h	—	595h	—	615h	PWM2DCH	695h	CWG1CON2	715h	—	795h	—
416h	—	496h	—	516h	—	596h	—	616h	PWM2CON	696h	—	716h	—	796h	—
417h	—	497h	—	517h	—	597h	—	617h	PWM3DCL	697h	—	717h	—	797h	—
418h	—	498h	NCO1ACCL	518h	—	598h	—	618h	PWM3DCH	698h	—	718h	—	798h	—
419h	—	499h	NCO1ACCH	519h	—	599h	—	619h	PWM3CON	699h	—	719h	—	799h	—
41Ah	—	49Ah	NCO1ACCU	51Ah	—	59Ah	—	61Ah	PWM4DCL	69Ah	—	71Ah	—	79Ah	—
41Bh	—	49Bh	NCO1INCL	51Bh	—	59Bh	—	61Bh	PWM4DCH	69Bh	—	71Bh	—	79Bh	—
41Ch	—	49Ch	NCO1INCH	51Ch	—	59Ch	—	61Ch	PWM4CON	69Ch	—	71Ch	—	79Ch	—
41Dh	—	49Dh	—	51Dh	—	59Dh	—	61Dh	—	69Dh	—	71Dh	—	79Dh	—
41Eh	—	49Eh	NCO1CON	51Eh	—	59Eh	—	61Eh	—	69Eh	—	71Eh	—	79Eh	—
41Fh	—	49Fh	NCO1CLK	51Fh	—	59Fh	—	61Fh	—	69Fh	—	71Fh	—	79Fh	—
420h	Unimplemented Read as '0'	4A0h	Unimplemented Read as '0'	520h	Unimplemented Read as '0'	5A0h	Unimplemented Read as '0'	620h	Unimplemented Read as '0'	6A0h	Unimplemented Read as '0'	720h	Unimplemented Read as '0'	7A0h	Unimplemented Read as '0'
46Fh	—	4EFh	—	56Fh	—	5EFh	—	66Fh	—	6EFh	—	76Fh	—	7EFh	—
470h	Common RAM (Accesses 70h – 7Fh)	4F0h	Common RAM (Accesses 70h – 7Fh)	570h	Common RAM (Accesses 70h – 7Fh)	5F0h	Common RAM (Accesses 70h – 7Fh)	670h	Common RAM (Accesses 70h – 7Fh)	6F0h	Common RAM (Accesses 70h – 7Fh)	770h	Common RAM (Accesses 70h – 7Fh)	7F0h	Common RAM (Accesses 70h – 7Fh)
47Fh	—	4FFh	—	57Fh	—	5FFh	—	67Fh	—	6FFh	—	77Fh	—	7FFh	—
BANK 16		BANK 17		BANK 18		BANK 19		BANK 20		BANK 21		BANK 22		BANK 23	
800h	Core Registers (Table 3-2)	880h	Core Registers (Table 3-2)	900h	Core Registers (Table 3-2)	980h	Core Registers (Table 3-2)	A00h	Core Registers (Table 3-2)	A80h	Core Registers (Table 3-2)	B00h	Core Registers (Table 3-2)	B80h	Core Registers (Table 3-2)
80Bh	—	88Bh	—	90Bh	—	98Bh	—	A0Bh	—	A8Bh	—	B0Bh	—	B8Bh	—
80Ch	Unimplemented Read as '0'	88Ch	Unimplemented Read as '0'	90Ch	Unimplemented Read as '0'	98Ch	Unimplemented Read as '0'	A0Ch	Unimplemented Read as '0'	A8Ch	Unimplemented Read as '0'	B0Ch	Unimplemented Read as '0'	B8Ch	Unimplemented Read as '0'
86Fh	—	8EFh	—	96Fh	—	9EFh	—	A6Fh	—	A6Fh	—	B6Fh	—	BEFh	—
870h	Common RAM (Accesses 70h – 7Fh)	8F0h	Common RAM (Accesses 70h – 7Fh)	970h	Common RAM (Accesses 70h – 7Fh)	9F0h	Common RAM (Accesses 70h – 7Fh)	A70h	Common RAM (Accesses 70h – 7Fh)	A70h	Common RAM (Accesses 70h – 7Fh)	B70h	Common RAM (Accesses 70h – 7Fh)	BF0h	Common RAM (Accesses 70h – 7Fh)
87Fh	—	8FFh	—	97Fh	—	9FFh	—	A7Fh	—	A7Fh	—	B7Fh	—	BFh	—

Legend:  = Unimplemented data memory locations, read as '0'

**TABLE 3-3: PIC16(L)F1507 MEMORY MAP (CONTINUED)**

BANK 24		BANK 25		BANK 26		BANK 27		BANK 28		BANK 29		BANK 30		BANK 31	
C00h	Core Registers (Table 3-2)	C80h	Core Registers (Table 3-2)	D00h	Core Registers (Table 3-2)	D80h	Core Registers (Table 3-2)	E00h	Core Registers (Table 3-2)	E80h	Core Registers (Table 3-2)	F00h	Core Registers (Table 3-2)	F80h	Core Registers (Table 3-2)
C0Bh	—	C8Bh	—	D0Bh	—	D8Bh	—	E0Bh	—	E8Bh	—	F0Bh	—	F8Bh	—
C0Ch	—	C8Ch	—	D0Ch	—	D8Ch	—	E0Ch	—	E8Ch	—	F0Ch	—	F8Ch	—
C0Dh	—	C8Dh	—	D0Dh	—	D8Dh	—	E0Dh	—	E8Dh	—	F0Dh	—	F8Dh	—
C0Eh	—	C8Eh	—	D0Eh	—	D8Eh	—	E0Eh	—	E8Eh	—	F0Eh	—	F8Eh	—
C0Fh	—	C8Fh	—	D0Fh	—	D8Fh	—	E0Fh	—	E8Fh	—	F0Fh	—	F8Fh	—
C10h	—	C90h	—	D10h	—	D90h	—	E10h	—	E90h	—	F10h	—	F90h	—
C11h	—	C91h	—	D11h	—	D91h	—	E11h	—	E91h	—	F11h	—	F91h	—
C12h	—	C92h	—	D12h	—	D92h	—	E12h	—	E92h	—	F12h	—	F92h	—
C13h	—	C93h	—	D13h	—	D93h	—	E13h	—	E93h	—	F13h	—	F93h	—
C14h	—	C94h	—	D14h	—	D94h	—	E14h	—	E94h	—	F14h	—	F94h	—
C15h	—	C95h	—	D15h	—	D95h	—	E15h	—	E95h	—	F15h	—	F95h	—
C16h	—	C96h	—	D16h	—	D96h	—	E16h	—	E96h	—	F16h	—	F96h	—
C17h	—	C97h	—	D17h	—	D97h	—	E17h	—	E97h	—	F17h	—	F97h	—
C18h	—	C98h	—	D18h	—	D98h	—	E18h	—	E98h	—	F18h	—	F98h	—
C19h	—	C99h	—	D19h	—	D99h	—	E19h	—	E99h	—	F19h	—	F99h	—
C1Ah	—	C9Ah	—	D1Ah	—	D9Ah	—	E1Ah	—	E9Ah	—	F1Ah	—	F9Ah	—
C1Bh	—	C9Bh	—	D1Bh	—	D9Bh	—	E1Bh	—	E9Bh	—	F1Bh	—	F9Bh	—
C1Ch	—	C9Ch	—	D1Ch	—	D9Ch	—	E1Ch	—	E9Ch	—	F1Ch	—	F9Ch	—
C1Dh	—	C9Dh	—	D1Dh	—	D9Dh	—	E1Dh	—	E9Dh	—	F1Dh	—	F9Dh	—
C1Eh	—	C9Eh	—	D1Eh	—	D9Eh	—	E1Eh	—	E9Eh	—	F1Eh	—	F9Eh	—
C1Fh	—	C9Fh	—	D1Fh	—	D9Fh	—	E1Fh	—	E9Fh	—	F1Fh	—	F9Fh	—
C20h	Unimplemented Read as '0'	CA0h	Unimplemented Read as '0'	D20h	Unimplemented Read as '0'	DA0h	Unimplemented Read as '0'	E20h	Unimplemented Read as '0'	EA0h	Unimplemented Read as '0'	F20h	—	FA0h	—
C6Fh	—	CEFh	—	D6Fh	—	DEFh	—	E6Fh	—	EEFh	—	F6Fh	—	FEFh	—
C70h	Common RAM (Accesses 70h – 7Fh)	CF0h	Common RAM (Accesses 70h – 7Fh)	D70h	Common RAM (Accesses 70h – 7Fh)	DF0h	Common RAM (Accesses 70h – 7Fh)	E70h	Common RAM (Accesses 70h – 7Fh)	EF0h	Common RAM (Accesses 70h – 7Fh)	F70h	Common RAM (Accesses 70h – 7Fh)	FF0h	Common RAM (Accesses 70h – 7Fh)
CFFh	—	CFFh	—	D7Fh	—	DFh	—	E7Fh	—	EFFh	—	F7Fh	—	FFh	—

**Legend:** ■ = Unimplemented data memory locations, read as '0'.

See Table 3-3 for register mapping details

See Table 3-3 for register mapping details

# PIC16(L)F1507

**TABLE 3-3: PIC16(L)F1507 MEMORY MAP (CONTINUED)**

Bank 30		Bank 31		
F0Ch	—	F8Ch	Unimplemented Read as '0'	
F0Dh	—	FE3h		
F0Eh	—	FE4h		STATUS_SHAD
F0Fh	CLCDATA	FE5h		WREG_SHAD
F10h	CLC1CON	FE6h		BSR_SHAD
F11h	CLC1POL	FE7h		PCLATH_SHAD
F12h	CLC1SEL0	FE8h		FSR0L_SHAD
F13h	CLC1SEL1	FE9h		FSR0H_SHAD
F14h	CLC1GLS0	FEAh		FSR1L_SHAD
F15h	CLC1GLS1	FEBh		FSR1H_SHAD
F16h	CLC1GLS2	FECh	—	
F17h	CLC1GLS3	FEDh	STKPTR	
F18h	CLC2CON	FEEh	TOSL	
F19h	CLC2POL	FEFh	TOSH	
F1Ah	CLC2SEL0			
F1Bh	CLC2SEL1			
F1Ch	CLC2GLS0			
F1Dh	CLC2GLS1			
F1Eh	CLC2GLS2			
F1Fh	CLC2GLS3			
F20h	Unimplemented Read as '0'			
F6Fh				

**Legend:**   = Unimplemented data memory locations, read as '0'.

## 3.3.6 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	
<b>Bank 0-31</b>												
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	—	—	—	$\overline{TO}$	$\overline{PD}$	Z	DC	C	---1 1000	---q ruuu	
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	—	—	—	BSR<4:0>					---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	IOCFIE	TMR0IF	INTF	IOCFIF	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'.

# PIC16(L)F1507

**TABLE 3-5: SPECIAL FUNCTION REGISTER SUMMARY**

Address	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
<b>Bank 0</b>											
00Ch	PORTA	—	—	RA5	RA4	RA3	RA2	RA1	RA0	--xx xxxx	--xx xxxx
00Dh	PORTB	RB7	RB6	RB5	RB4	—	—	—	—	xxxx ----	xxxx ----
00Eh	PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	xxxx xxxx	xxxx xxxx
010h	—	Unimplemented								—	—
011h	PIR1	TMR1GIF	ADIF	—	—	—	—	TMR2IF	TMR1IF	00-- --00	00-- --00
012h	PIR2	—	—	—	—	—	NCO1IF	—	—	---- -0--	---- -0--
013h	PIR3	—	—	—	—	—	—	CLC2IF	CLC1IF	---- --00	---- --00
014h	—	Unimplemented								—	—
015h	TMR0	Holding Register for the 8-bit Timer0 Count								xxxx xxxx	uuuu uuuu
016h	TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Count								xxxx xxxx	uuuu uuuu
017h	TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Count								xxxx xxxx	uuuu uuuu
018h	T1CON	TMR1CS<1:0>		T1CKPS<1:0>		—	$\overline{T1SYNC}$	—	TMR1ON	0000 -0-0	uuuu -u-u
019h	T1GCON	TMR1GE	T1GPOL	T1GTM	T1GSPM	T1GGO/DONE	T1GVAL	—	T1GSS0	0000 0x-0	uuuu ux-u
01Ah	TMR2	Timer2 Module Register								0000 0000	0000 0000
01Bh	PR2	Timer2 Period Register								1111 1111	1111 1111
01Ch	T2CON	—	T2OUTPS<3:0>				TMR2ON	T2CKPS<1:0>		-000 0000	-000 0000
01Dh to 01Fh	—	Unimplemented								—	—

**Bank 1**

08Ch	TRISA	—	—	TRISA5	TRISA4	— <sup>(2)</sup>	TRISA2	TRISA1	TRISA0	--11 1111	--11 1111
08Dh	TRISB	TRISB7	TRISB6	TRISB5	TRISB4	—	—	—	—	1111 ----	1111 ----
08Eh	TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	1111 1111	1111 1111
08Fh	—	Unimplemented								—	—
090h	—	Unimplemented								—	—
091h	PIE1	TMR1GIE	ADIE	—	—	—	—	TMR2IE	TMR1IE	00-- --00	00-- --00
092h	PIE2	—	—	—	—	—	NCO1IE	—	—	---- -0--	---- -0--
093h	PIE3	—	—	—	—	—	—	CLC2IE	CLC1IE	---- --00	---- --00
094h	—	Unimplemented								—	—
095h	OPTION_REG	$\overline{WPUEN}$	INTEDG	TMR0CS	TMR0SE	PSA	PS<2:0>			1111 1111	1111 1111
096h	PCON	STKOVF	STKUNF	—	RWD $\overline{T}$	$\overline{RMCLR}$	$\overline{R}$	$\overline{POR}$	$\overline{BOR}$	00-1 11q $\overline{q}$	q $\overline{q}$ -q q $\overline{q}$ uu
097h	WDTCON	—	—	WDTPS<4:0>				SWDTEN	—	--01 0110	--01 0110
098h	—	Unimplemented								—	—
099h	OSCCON	—	IRCF<3:0>			—	SCS<1:0>			-011 1-00	-011 1-00
09Ah	OSCSTAT	—	—	—	HFIOFR	—	—	LFIOFR	HFIOFS	---0 --00	---q --q $\overline{q}$
09Bh	ADRESL	ADC Result Register Low								xxxx xxxx	uuuu uuuu
09Ch	ADRESH	ADC Result Register High								xxxx xxxx	uuuu uuuu
09Dh	ADCON0	—	CHS<4:0>				$\overline{GO/DONE}$	ADON	—	-000 0000	-000 0000
09Eh	ADCON1	ADFM	ADCS<2:0>			—	—	ADPREF<1:0>		0000 --00	0000 --00
09Fh	ADCON2	TRIGSEL<3:0>				—	—	—	—	0000 ----	0000 ----

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

**Note 1:** PIC16F1507 only.

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



# PIC16(L)F1507

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

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets	
<b>Bank 2</b>												
10Ch	LATA	—	—	LATA5	LATA4	—	LATA2	LATA1	LATA0	--xx -xxx	--uu -uuu	
10Dh	LATB	LATB7	LATB6	LATB5	LATB4	—	—	—	—	xxxx ----	uuuu ----	
10Eh	LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	xxxx xxxx	uuuu uuuu	
10Fh to 115h	—	Unimplemented								—	—	
116h	BORCON	SBOREN	BORFS	—	—	—	—	—	BORRDY	10-- ---q	uu-- ---u	
117h	FVRCON	FVREN	FVRRDY	TSEN	TSRNG	—	—	ADFVR<1:0>		0q00 --00	0q00 --00	
118h to 11Ch	—	Unimplemented								—	—	
11Dh	APFCON	—	—	—	—	—	—	CLC1SEL	NCO1SEL	---- --00	---- --00	
11Eh	—	Unimplemented								—	—	
11Fh	—	Unimplemented								—	—	
<b>Bank 3</b>												
18Ch	ANSELA	—	—	—	ANSA4	—	ANSA2	ANSA1	ANSA0	---1 -111	---1 -111	
18Dh	ANSELB	—	—	ANSB5	ANSB4	—	—	—	—	--11 ----	--11 ----	
18Eh	ANSELC	ANSC7	ANSC6	—	—	ANSC3	ANSC2	ANSC1	ANSC0	11-- 1111	11-- 1111	
18Fh	—	Unimplemented								—	—	
190h	—	Unimplemented								—	—	
191h	PMADRL	Flash Program Memory Address Register Low Byte								0000 0000	0000 0000	
192h	PMADRH	— <sup>(2)</sup>	Flash Program Memory Address Register High Byte								1000 0000	1000 0000
193h	PMDATL	Flash Program Memory Read Data Register Low Byte								xxxx xxxx	uuuu uuuu	
194h	PMDATH	—	—	Flash Program Memory Read Data Register High Byte						--xx xxxx	--uu uuuu	
195h	PMCON1	— <sup>(2)</sup>	CFGS	LWLO	FREE	WRERR	WREN	WR	RD	1000 x000	1000 q000	
196h	PMCON2	Flash Program Memory Control Register 2								0000 0000	0000 0000	
197h	VREGCON <sup>(1)</sup>	—	—	—	—	—	—	VREGPM	Reserved	---- --01	---- --01	
198h to 19Fh	—	Unimplemented								—	—	

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

**Note 1:** PIC16F1507 only.

**Note 2:** Unimplemented, read as '1'.

# PIC16(L)F1507

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

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
<b>Bank 4</b>											
20Ch	WPUA	—	—	WPUA5	WPUA4	WPUA3	WPUA2	WPUA1	WPUA0	--11 1111	--11 1111
20Dh	WPUB	WPUB7	WPUB6	WPUB5	WPUB4	—	—	—	—	1111 ----	1111 ----
20Eh to 21Fh	—	Unimplemented								—	—
<b>Bank 5</b>											
28Ch to 29Fh	—	Unimplemented								—	—
<b>Bank 6</b>											
30Ch to 31Fh	—	Unimplemented								—	—
<b>Bank 7</b>											
38Ch to 390h	—	Unimplemented								—	—
391h	IOCAP	—	—	IOCAP5	IOCAP4	IOCAP3	IOCAP2	IOCAP1	IOCAP0	--00 0000	--00 0000
392h	IOCAN	—	—	IOCAN5	IOCAN4	IOCAN3	IOCAN2	IOCAN1	IOCAN0	--00 0000	--00 0000
393h	IOCAF	—	—	IOCAF5	IOCAF4	IOCAF3	IOCAF2	IOCAF1	IOCAF0	--00 0000	--00 0000
394h	IOCBP	IOCBP7	IOCBP6	IOCBP5	IOCBP4	—	—	—	—	0000 ----	0000 ----
395h	IOCBN	IOCBN7	IOCBN6	IOCBN5	IOCBN4	—	—	—	—	0000 ----	0000 ----
396h	IOCBF	IOCBF7	IOCBF6	IOCBF5	IOCBF4	—	—	—	—	0000 ----	0000 ----
397h to 39Fh	—	Unimplemented								—	—
<b>Bank 8</b>											
40Ch to 41Fh	—	Unimplemented								—	—
<b>Bank 9</b>											
48Ch to 497h	—	Unimplemented								—	—
498h	NCO1ACCL	NCO1ACC<7:0>								0000 0000	0000 0000
499h	NCO1ACCH	NCO1ACC<15:8>								0000 0000	0000 0000
49Ah	NCO1ACCU	NCO1ACC<19:16>								0000 0000	0000 0000
49Bh	NCO1INCL	NCO1INC<7:0>								0000 0001	0000 0001
49Ch	NCO1INCH	NCO1INC<15:8>								0000 0000	0000 0000
49Dh	—	Unimplemented								—	—
49Eh	NCO1CON	N1EN	N1OE	N1OUT	N1POL	—	—	—	N1PFM	0000 ---0	0000 ---0
49Fh	NCO1CLK	N1PWS<2:0>			—	—	—	N1CKS<1:0>		0000 --00	0000 --00

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

**Note 1:** PIC16F1507 only.

**Note 2:** Unimplemented, read as '1'.

# PIC16(L)F1507

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

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
<b>Bank 10</b>											
50Ch to 51Fh	—	Unimplemented								—	—
<b>Bank 11</b>											
58Ch to 59Fh	—	Unimplemented								—	—
<b>Bank 12</b>											
60Ch to 610h	—	Unimplemented								—	—
611h	PWM1DCL	PWM1DCL<7:6>		—	—	—	—	—	—	00-- ----	00-- ----
612h	PWM1DCH	PWM1DCH<7:0>								xxxx xxxx	uuuu uuuu
613h	PWM1CON0	PWM1EN	PWM1OE	PWM1OUT	PWM1POL	—	—	—	—	0000 ----	0000 ----
614h	PWM2DCL	PWM2DCL<7:6>		—	—	—	—	—	—	00-- ----	00-- ----
615h	PWM2DCH	PWM2DCH<7:0>								xxxx xxxx	uuuu uuuu
616h	PWM2CON0	PWM2EN	PWM2OE	PWM2OUT	PWM2POL	—	—	—	—	0000 ----	0000 ----
617h	PWM3DCL	PWM3DCL<7:6>		—	—	—	—	—	—	00-- ----	00-- ----
618h	PWM3DCH	PWM3DCH<7:0>								xxxx xxxx	uuuu uuuu
619h	PWM3CON0	PWM3EN	PWM3OE	PWM3OUT	PWM3POL	—	—	—	—	0000 ----	0000 ----
61Ah	PWM4DCL	PWM4DCL<7:6>		—	—	—	—	—	—	00-- ----	00-- ----
61Bh	PWM4DCH	PWM4DCH<7:0>								xxxx xxxx	uuuu uuuu
61Ch	PWM4CON0	PWM4EN	PWM4OE	PWM4OUT	PWM4POL	—	—	—	—	0000 ----	0000 ----
61Dh to 61Fh	—	Unimplemented								—	—
<b>Bank 13</b>											
68Ch to 690h	—	Unimplemented								—	—
691h	CWG1DBR	—	—	CWG1DBR<5:0>						--00 0000	--00 0000
692h	CWG1DBF	—	—	CWG1DBF<5:0>						--xx xxxx	--xx xxxx
693h	CWG1CON0	G1EN	G1OEB	G1OEA	G1POLB	G1POLA	—	—	G1CS0	0000 0--0	0000 0--0
694h	CWG1CON1	G1ASDLB<1:0>		G1ASDLA<1:0>		—	G1IS<2:0>			0000 -000	0000 -000
695h	CWG1CON2	G1ASE	G1ARSEN	—	—	—	—	G1ASDSFLT	G1ASDSCLC2	00-- --00	00-- --00
696h to 69Fh	—	Unimplemented								—	—

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

- Note 1:** PIC16F1507 only.  
**Note 2:** Unimplemented, read as '1'.

# PIC16(L)F1507

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

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
<b>Banks 14-29</b>											
x0Ch/ x8Ch — x1Fh/ x9Fh	—	Unimplemented								—	—
<b>Bank 30</b>											
F0Ch to F0Eh	—	Unimplemented								—	—
F0Fh	CLCDATA	—	—	—	—	—	—	MLC2OUT	MLC1OUT	---- --00	---- --00
F10h	CLC1CON	LC1EN	LC1OE	LC1OUT	LC1INTP	LC1INTN	LC1MODE<2:0>			0000 0000	0000 0000
F11h	CLC1POL	LC1POL	—	—	—	LC1G4POL	LC1G3POL	LC1G2POL	LC1G1POL	0--- xxxx	0--- uuuu
F12h	CLC1SEL0	—	LC1D2S<2:0>			—	LC1D1S<2:0>			-xxx -xxx	-uuu -uuu
F13h	CLC1SEL1	—	LC1D4S<2:0>			—	LC1D3S<2:0>			-xxx -xxx	-uuu -uuu
F14h	CLC1GLS0	LC1G1D4T	LC1G1D4N	LC1G1D3T	LC1G1D3N	LC1G1D2T	LC1G1D2N	LC1G1D1T	LC1G1D1N	xxxx xxxx	uuuu uuuu
F15h	CLC1GLS1	LC1G2D4T	LC1G2D4N	LC1G2D3T	LC1G2D3N	LC1G2D2T	LC1G2D2N	LC1G2D1T	LC1G2D1N	xxxx xxxx	uuuu uuuu
F16h	CLC1GLS2	LC1G3D4T	LC1G3D4N	LC1G3D3T	LC1G3D3N	LC1G3D2T	LC1G3D2N	LC1G3D1T	LC1G3D1N	xxxx xxxx	uuuu uuuu
F17h	CLC1GLS3	LC1G4D4T	LC1G4D4N	LC1G4D3T	LC1G4D3N	LC1G4D2T	LC1G4D2N	LC1G4D1T	LC1G4D1N	xxxx xxxx	uuuu uuuu
F18h	CLC2CON	LC2EN	LC2OE	LC2OUT	LC2INTP	LC2INTN	LC2MODE<2:0>			0000 0000	0000 0000
F19h	CLC2POL	LC2POL	—	—	—	LC2G4POL	LC2G3POL	LC2G2POL	LC2G1POL	0--- xxxx	0--- uuuu
F1Ah	CLC2SEL0	—	LC2D2S<2:0>			—	LC2D1S<2:0>			-xxx -xxx	-uuu -uuu
F1Bh	CLC2SEL1	—	LC2D4S<2:0>			—	LC2D3S<2:0>			-xxx -xxx	-uuu -uuu
F1Ch	CLC2GLS0	LC2G1D4T	LC2G1D4N	LC2G1D3T	LC2G1D3N	LC2G1D2T	LC2G1D2N	LC2G1D1T	LC2G1D1N	xxxx xxxx	uuuu uuuu
F1Dh	CLC2GLS1	LC2G2D4T	LC2G2D4N	LC2G2D3T	LC2G2D3N	LC2G2D2T	LC2G2D2N	LC2G2D1T	LC2G2D1N	xxxx xxxx	uuuu uuuu
F1Eh	CLC2GLS2	LC2G3D4T	LC2G3D4N	LC2G3D3T	LC2G3D3N	LC2G3D2T	LC2G3D2N	LC2G3D1T	LC2G3D1N	xxxx xxxx	uuuu uuuu
F1Fh	CLC2GLS3	LC2G4D4T	LC2G4D4N	LC2G4D3T	LC2G4D3N	LC2G4D2T	LC2G4D2N	LC2G4D1T	LC2G4D1N	xxxx xxxx	uuuu uuuu
F20h to F6Fh	—	Unimplemented								—	—

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

**Note 1:** PIC16F1507 only.

**Note 2:** Unimplemented, read as '1'.

# PIC16(L)F1507

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

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets	
<b>Bank 31</b>												
F8Ch — FE3h	—	Unimplemented								—	—	
FE4h	STATUS_ SHAD	—	—	—	—	—	Z_SHAD	DC_SHAD	C_SHAD	---- -xxx	---- -uuu	
FE5h	WREG_ SHAD	Working Register Shadow								xxxx xxxx	uuuu uuuu	
FE6h	BSR_ SHAD	—	—	—	Bank Select Register Shadow					---x xxxx	---u uuuu	
FE7h	PCLATH_ SHAD	—	Program Counter Latch High Register Shadow								-xxx xxxx	uuuu uuuu
FE8h	FSR0L_ SHAD	Indirect Data Memory Address 0 Low Pointer Shadow								xxxx xxxx	uuuu uuuu	
FE9h	FSR0H_ SHAD	Indirect Data Memory Address 0 High Pointer Shadow								xxxx xxxx	uuuu uuuu	
FEAh	FSR1L_ SHAD	Indirect Data Memory Address 1 Low Pointer Shadow								xxxx xxxx	uuuu uuuu	
FEBh	FSR1H_ SHAD	Indirect Data Memory Address 1 High Pointer 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, r = reserved. Shaded locations are unimplemented, read as '0'.

**Note 1:** PIC16F1507 only.

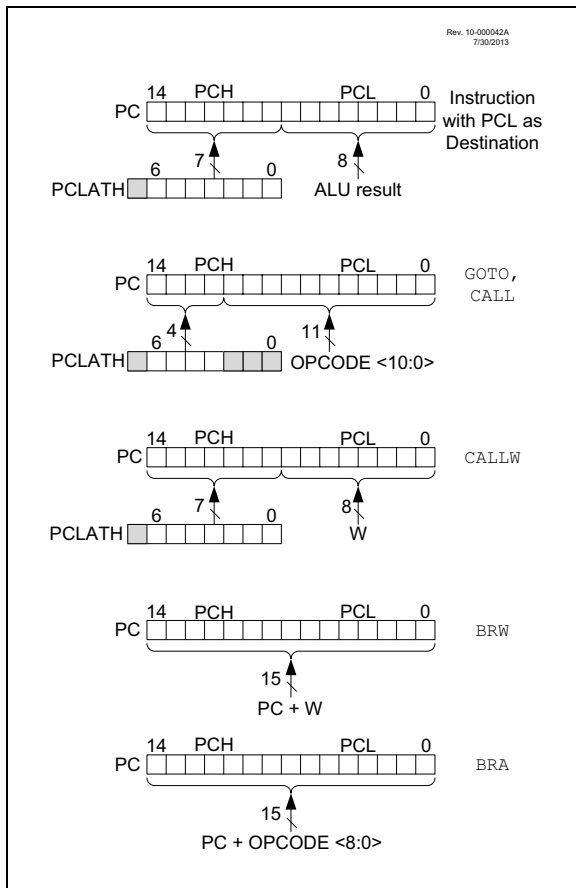
**Note 2:** Unimplemented, read as '1'.

# PIC16(L)F1507

## 3.4 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 3-3 shows the five situations for the loading of the PC.

**FIGURE 3-3: LOADING OF PC IN DIFFERENT SITUATIONS**



### 3.4.1 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.

### 3.4.2 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 Application Note AN556, "Implementing a Table Read" (DS00556).

### 3.4.3 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.

### 3.4.4 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.

## 3.5 Stack

All devices have a 16-level x 15-bit wide hardware stack (refer to Figures 3-4 through 3-7). 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 Words). 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 1:** 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.

### 3.5.1 ACCESSING THE STACK

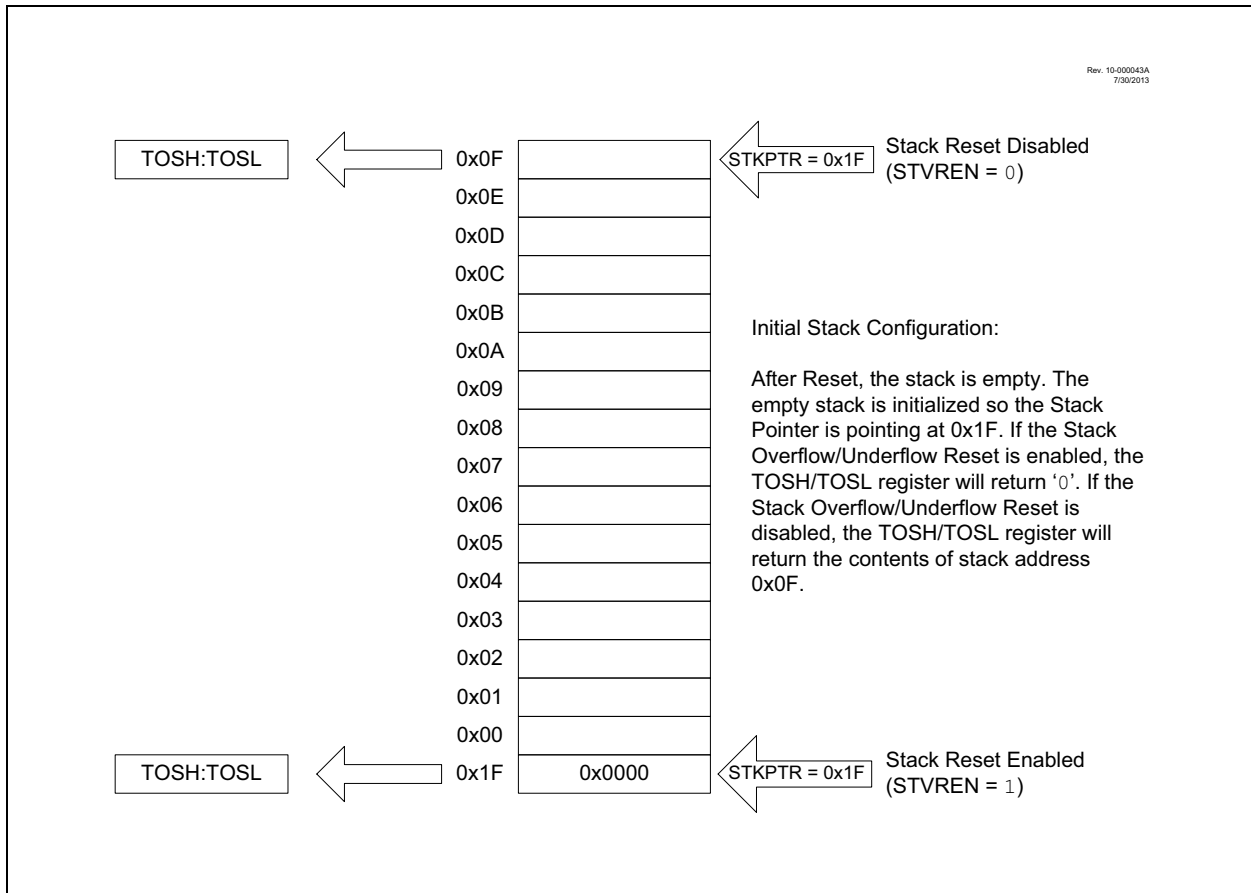
The stack is available through the `TOSH`, `TOSL` and `STKPTR` registers. `STKPTR` is the current value of the Stack Pointer. `TOSH:TOSL` register pair points to the TOP of the stack. Both registers are read/writable. `TOS` is split into `TOSH` and `TOSL` due to the 15-bit size of the PC. To access the stack, adjust the value of `STKPTR`, which will position `TOSH:TOSL`, then read/write to `TOSH:TOSL`. `STKPTR` is 5 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`.

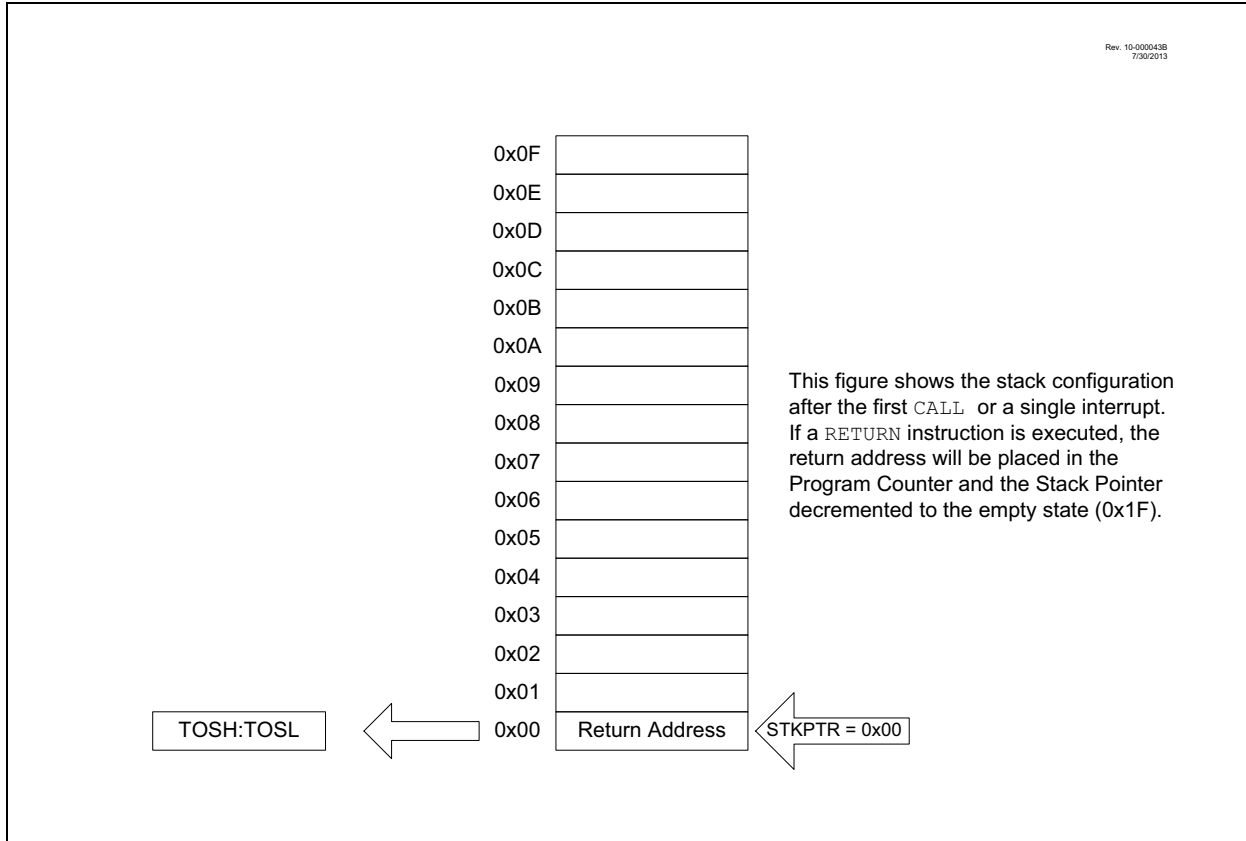
Reference Figure 3-4 through Figure 3-7 for examples of accessing the stack.

**FIGURE 3-4: ACCESSING THE STACK EXAMPLE 1**

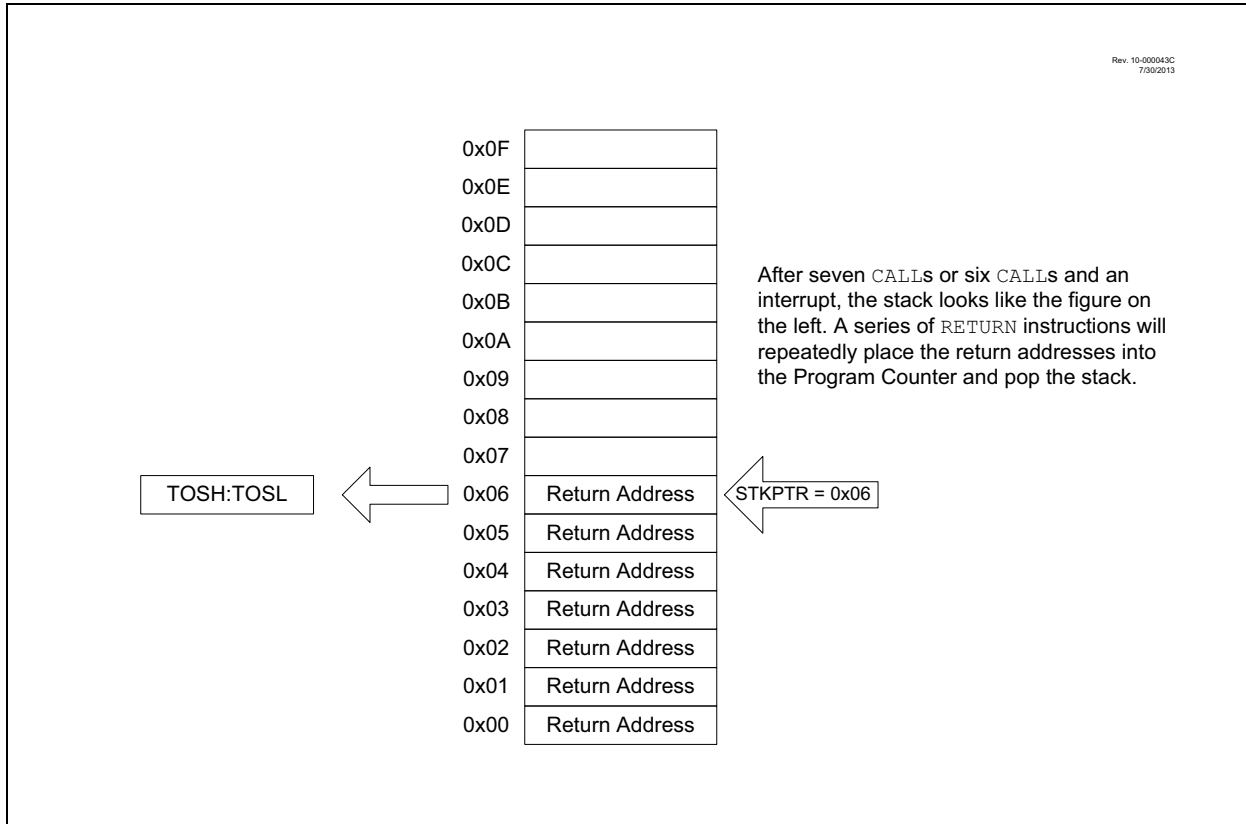


# PIC16(L)F1507

**FIGURE 3-5: ACCESSING THE STACK EXAMPLE 2**

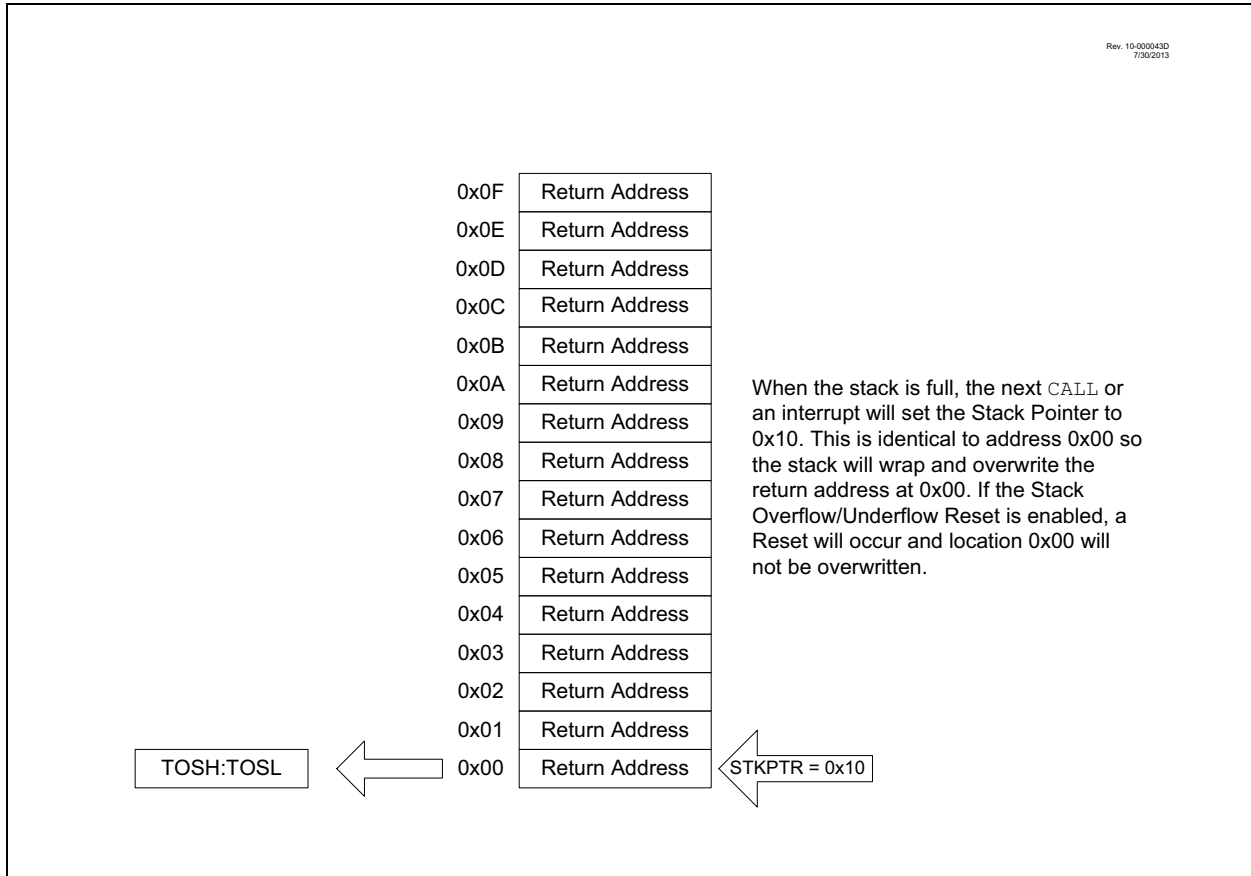


**FIGURE 3-6: ACCESSING THE STACK EXAMPLE 3**





**FIGURE 3-7: ACCESSING THE STACK EXAMPLE 4**



### 3.5.2 OVERFLOW/UNDERFLOW RESET

If the `STVREN` bit in Configuration Words 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.

### 3.6 Indirect Addressing

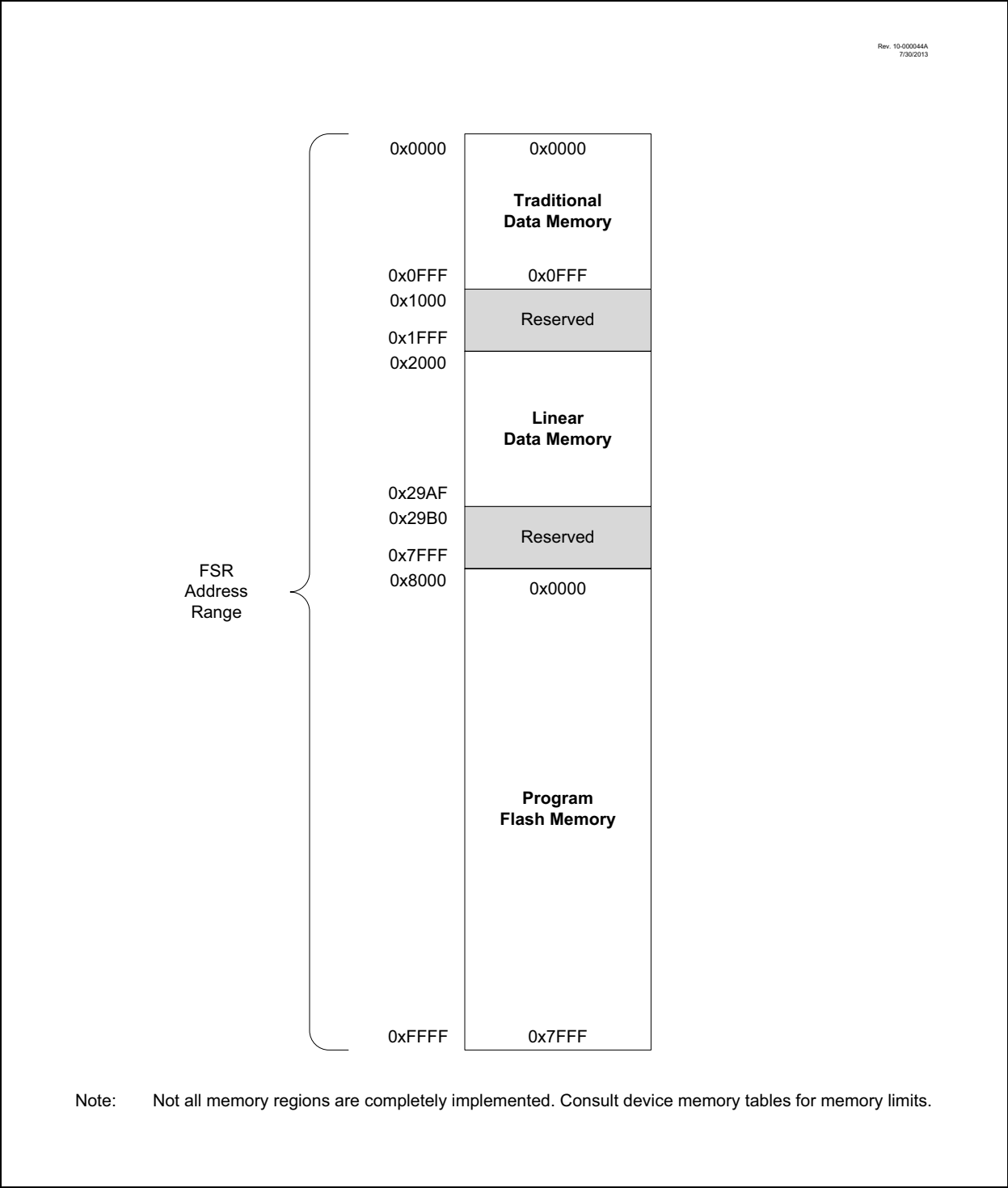
The `INDFn` registers are not physical registers. Any instruction that accesses an `INDFn` register actually accesses the register at the address specified by the File Select Registers (`FSR`). If the `FSRn` address specifies one of the two `INDFn` registers, the read will return '0' and the write will 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

# PIC16(L)F1507

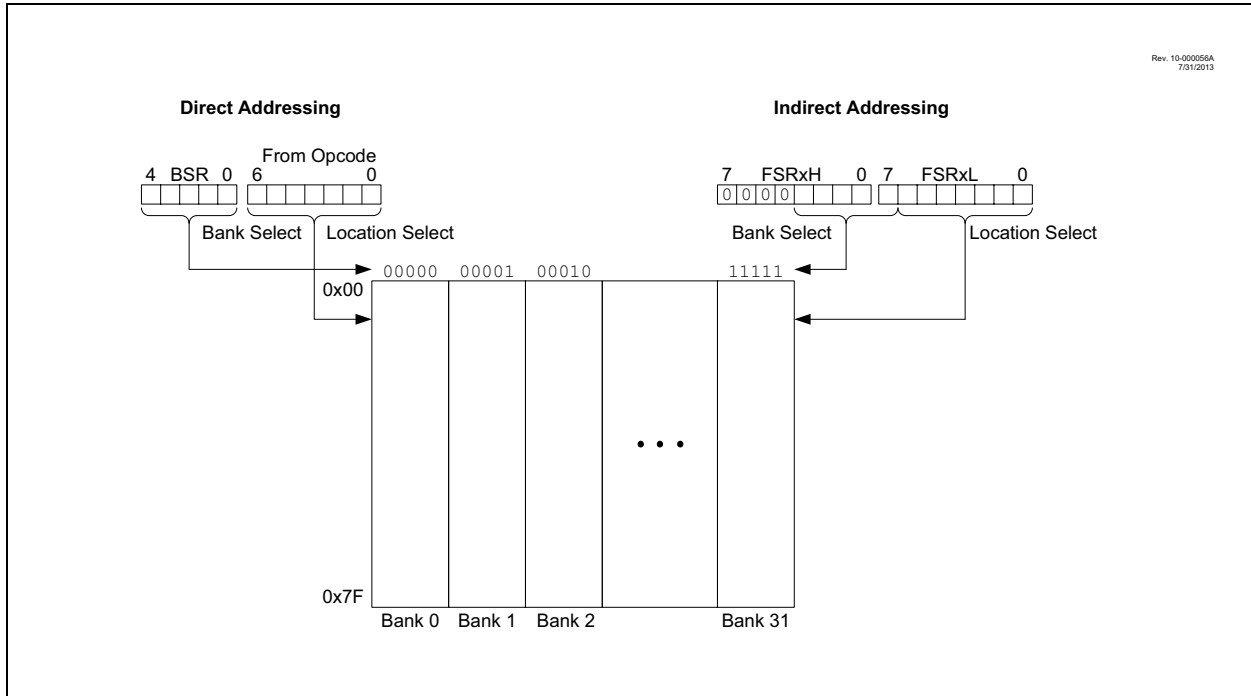
**FIGURE 3-8: INDIRECT ADDRESSING**



## 3.6.1 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 3-9: TRADITIONAL DATA MEMORY MAP**



# PIC16(L)F1507

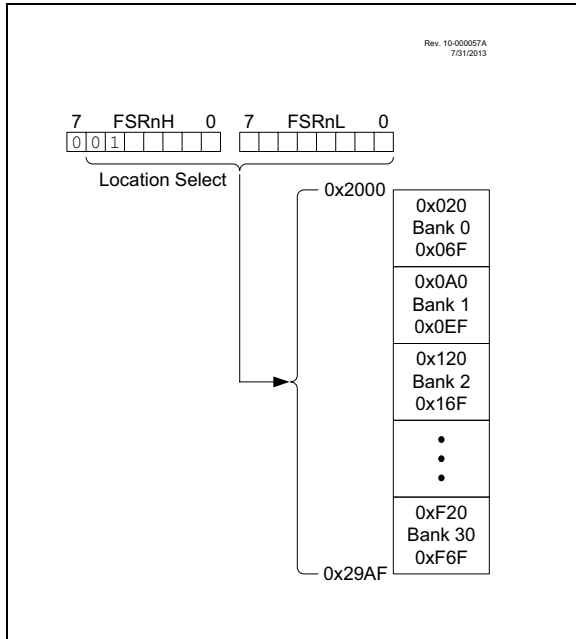
## 3.6.2 LINEAR DATA MEMORY

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.

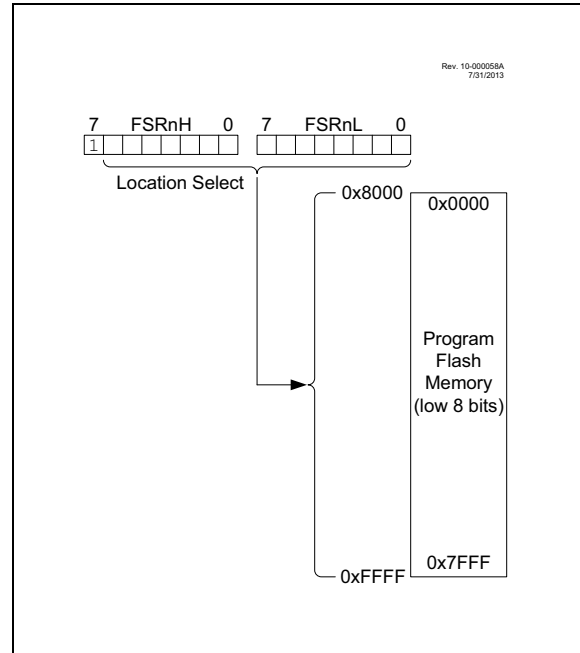
**FIGURE 3-10: LINEAR DATA MEMORY MAP**



## 3.6.3 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 3-11: PROGRAM FLASH MEMORY MAP**



## 4.0 DEVICE CONFIGURATION

Device configuration consists of Configuration Words, 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'.

# PIC16(L)F1507

## 4.2 Register Definitions: Configuration Words

### REGISTER 4-1: CONFIG1: CONFIGURATION WORD 1

U-1	U-1	R/P-1	R/P-1	R/P-1	U-1
—	—	CLKOUTEN	BOREN<1:0> <sup>(1)</sup>		—
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
$\overline{CP}$ <sup>(2)</sup>	MCLRE	$\overline{PWRT}$	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      **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<sup>(1)</sup>  
 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      **CP:** Code Protection bit<sup>(2)</sup>  
 1 = Program memory code protection is disabled  
 0 = Program memory code protection is enabled
- bit 6      **MCLRE:**  $\overline{MCLR}/VPP$  Pin Function Select bit  
If LVP bit = 1:  
             This bit is ignored.  
If LVP bit = 0:  
             1 =  $\overline{MCLR}/VPP$  pin function is  $\overline{MCLR}$ ; Weak pull-up enabled.  
             0 =  $\overline{MCLR}/VPP$  pin function is digital input;  $\overline{MCLR}$  internally disabled; Weak pull-up under control of WPUA3 bit.
- bit 5       **$\overline{PWRT}$ :** Power-Up Timer Enable bit  
 1 = PWRT disabled  
 0 = PWRT enabled
- bit 4-3      **WDTE<1:0>:** Watchdog Timer Enable bits  
 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  
 11 = ECH: External Clock, High-Power mode: on CLKIN pin  
 10 = ECM: External Clock, Medium Power mode: on CLKIN pin  
 01 = ECL: External Clock, Low-Power mode: on CLKIN pin  
 00 = INTOSC oscillator: I/O function on CLKIN pin

**Note 1:** Enabling Brown-out Reset does not automatically enable Power-up Timer.  
**Note 2:** Once enabled, code-protect can only be disabled by bulk erasing the device.

## REGISTER 4-2: CONFIG2: CONFIGURATION WORD 2

R/P-1	U-1	R/P-1	R/P-1	R/P-1	U-1
LVP <sup>(1)</sup>	—	$\overline{\text{LPBOR}}$	BORV <sup>(2)</sup>	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<sup>(1)</sup>  
                   1 = Low-voltage programming enabled  
                   0 = High-voltage on MCLR must be used for programming
- bit 12            **Unimplemented:** Read as '1'
- bit 11            **LPBOR:** Low-Power BOR Enable bit  
                   1 = Low-Power Brown-out Reset is disabled  
                   0 = Low-Power Brown-out Reset is enabled
- bit 10            **BORV:** Brown-Out Reset Voltage Selection bit<sup>(2)</sup>  
                   1 = Brown-out Reset voltage (**VBOR**), low trip point selected  
                   0 = Brown-out Reset voltage (**VBOR**), 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  
                   2 kW Flash memory (PIC16(L)F1507 only):  
                   11 = Write protection off  
                   10 = 000h to 1FFh write-protected, 200h to 7FFh may be modified  
                   01 = 000h to 3FFh write-protected, 400h to 7FFh may be modified  
                   00 = 000h to 7FFh write-protected, no addresses may be modified

- Note 1:** The LVP bit cannot be programmed to '0' when Programming mode is entered via LVP.  
**Note 2:** See **VBOR** parameter for specific trip point voltages.

# PIC16(L)F1507

---

## 4.3 Code Protection

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

### 4.3.1 PROGRAM MEMORY PROTECTION

The entire program memory space is protected from external reads and writes by the  $\overline{CP}$  bit in Configuration Words. When  $\overline{CP} = 0$ , 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 4.4 "Write Protection"](#) for more information.

## 4.4 Write Protection

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

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

## 4.5 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 "*PIC12(L)F1501/PIC16(L)F150X Memory Programming Specification*" (DS41573).



## 4.6 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.

## 4.7 Register Definitions: Device ID

### REGISTER 4-3: DEVID: DEVICE ID REGISTER

R	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

'1' = Bit is set

'0' = Bit is cleared

bit 13-5

**DEV<8:0>**: Device ID bits

Device	DEVID<13:0> Values	
	DEV<8:0>	REV<4:0>
PIC16LF1507	10 1101 110	x xxxxx
PIC16F1507	10 1101 000	x xxxxx

bit 4-0

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

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

# PIC16(L)F1507

---

## 5.0 OSCILLATOR MODULE

### 5.1 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 5-1](#) illustrates a block diagram of the oscillator module.

Clock sources can be supplied from an external clock or 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.

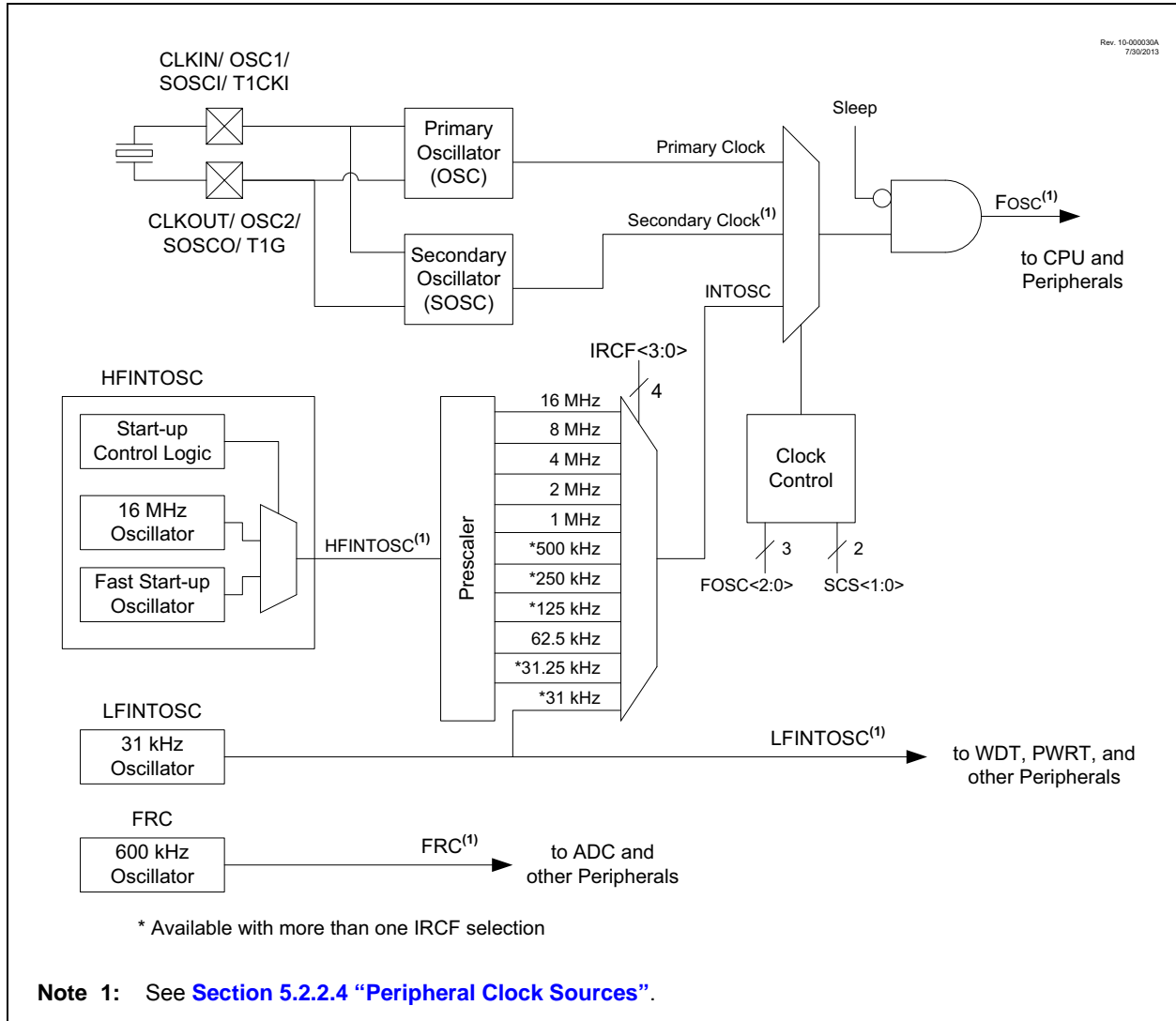
1. ECL – External Clock Low-Power mode (0 MHz to 0.5 MHz)
2. ECM – External Clock Medium Power mode (0.5 MHz to 4 MHz)
3. ECH – External Clock High-Power mode (4 MHz to 20 MHz)
4. INTOSC – Internal oscillator (31 kHz to 16 MHz)

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

The ECH, ECM, and ECL clock modes rely 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 5-1](#)). A wide selection of device clock frequencies may be derived from these two clock sources.

**FIGURE 5-1: SIMPLIFIED PIC® MCU CLOCK SOURCE BLOCK DIAGRAM**



# PIC16(L)F1507

## 5.2 Clock Source Types

Clock sources can be classified as external, internal or peripheral.

External clock sources rely on external circuitry for the clock source to function. Examples are: oscillator modules (ECH, ECM, ECL modes).

Internal clock sources are contained 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 (HFINTOSC) and the 31 kHz Low-Frequency Internal Oscillator (LFINTOSC).

The peripheral clock source is a nominal 600 kHz internal RC oscillator, FRC. The FRC is traditionally used with the ADC module, but is sometimes available to other peripherals. See [Section 5.2.2.4 “Peripheral Clock Sources”](#).

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 5.3 “Clock Switching”](#) for additional information.

### 5.2.1 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 Words 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 5.3 “Clock Switching”](#) for more information.

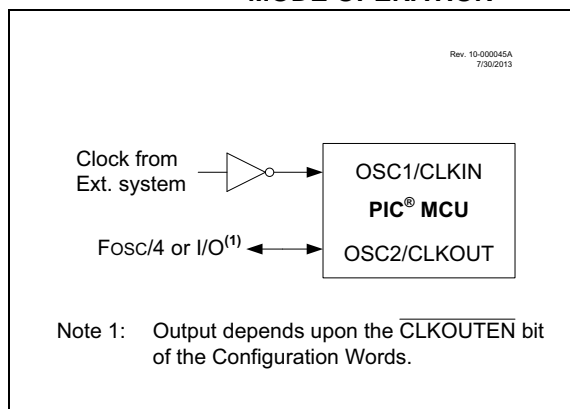
#### 5.2.1.1 EC Mode

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 OSC1 input. OSC2/CLKOUT is available for general purpose I/O or CLKOUT. [Figure 5-2](#) shows the pin connections for EC mode.

EC mode has three power modes to select from through the Fosc bits in the Configuration Words:

- ECH – High power, 4-20 MHz
- ECM – Medium power, 0.5-4 MHz
- ECL – Low power, 0-0.5 MHz

**FIGURE 5-2: EXTERNAL CLOCK (EC) MODE OPERATION**



## 5.2.2 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 Words 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 5.3 “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 CLKOUT pin is determined by the CLKOUTEN bit in Configuration Words.

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) operates at 31 kHz.

### 5.2.2.1 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 5-1](#)). The frequency derived from the HFINTOSC can be selected via software using the IRCF<3:0> bits of the OSCCON register. See [Section 5.2.2.6 “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> = 00, or
- Set the System Clock Source (SCS) bits of the OSCCON register to ‘1x’.

A fast start-up oscillator allows internal circuits to power-up and stabilize before switching to HFINTOSC.

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

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

### 5.2.2.2 LFINTOSC

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

The output of the LFINTOSC connects to a multiplexer (see [Figure 5-1](#)). Select 31 kHz, via software, using the IRCF<3:0> bits of the OSCCON register. See [Section 5.2.2.6 “Internal Oscillator Clock Switch Timing”](#) for more information. The LFINTOSC is also the frequency for the Power-up Timer (PWRT) and the 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> = 00, 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.

### 5.2.2.3 FRC

The FRC clock is an uncalibrated, nominal 600 kHz peripheral clock source.

The FRC is automatically turned on by the peripherals requesting the FRC clock.

The FRC clock continues to run during Sleep.

# PIC16(L)F1507

## 5.2.2.4 Peripheral Clock Sources

The clock sources described in this chapter and the Timer's are available to different peripherals. [Table 5-1](#) lists the clocks and timers available for each peripheral.

**TABLE 5-1: PERIPHERAL CLOCK SOURCES**

	FOSC	FRC	HFINTOSC	LFINTOSC	TMR0	TMR1	TMR2
ADC	•	•					
CLC	•	•	•	•	•	•	•
CWG	•		•				
NCO	•		•				
PWM	•						•
PWRT				•			
TMR0	•						
TMR1	•			•			
TMR2	•						
WDT				•			

## 5.2.2.5 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 postscaled output of the 16 MHz HFINTOSC and 31 kHz LFINTOSC connect to a multiplexer (see [Figure 5-1](#)). The Internal Oscillator Frequency Select bits IRCF<3:0> of the OSCCON register ([Register 5-1](#)) select the frequency output of the internal oscillators.

**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.

## 5.2.2.6 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 5-3](#)). 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:

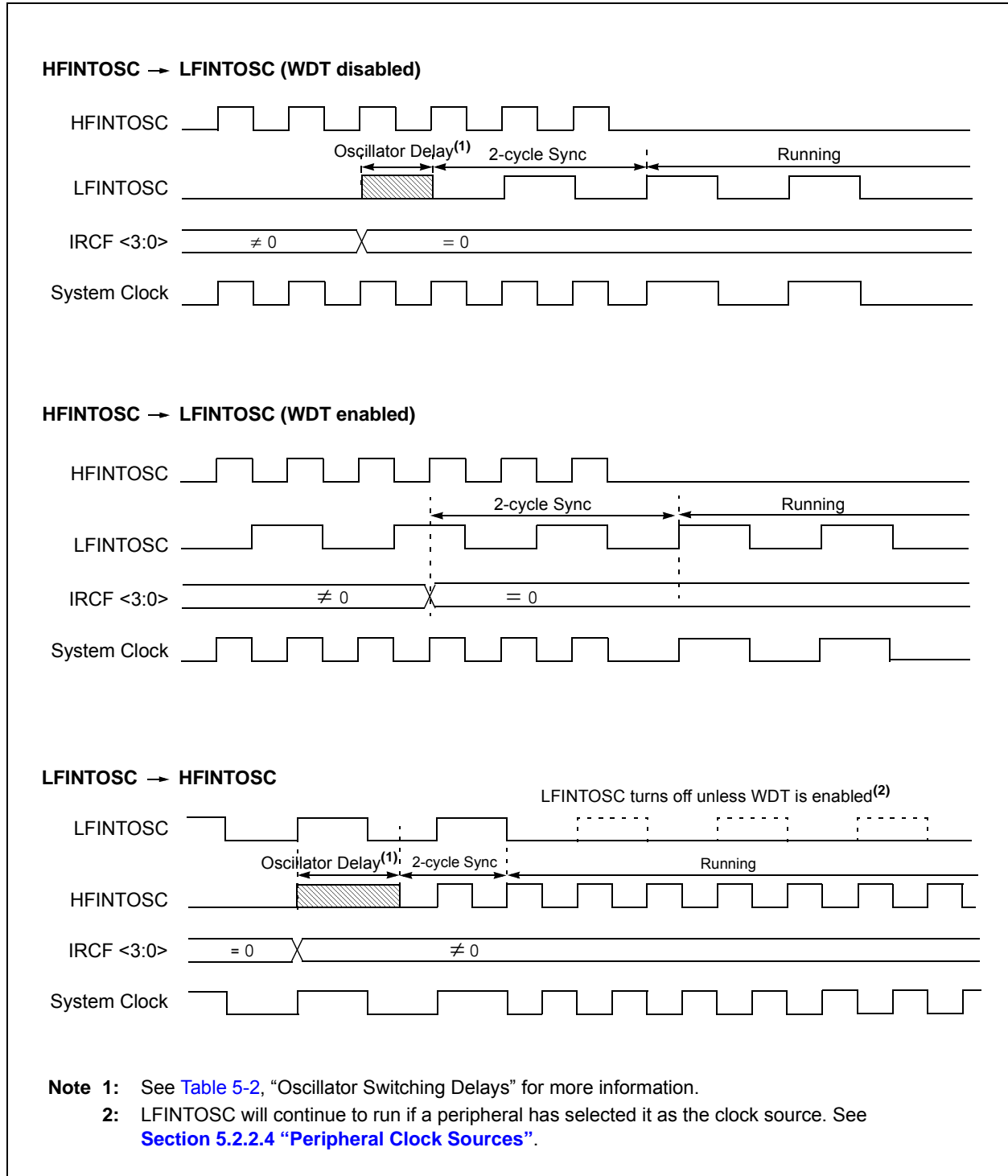
1. IRCF<3:0> bits of the OSCCON register are modified.
2. If the new clock is shut down, a clock start-up delay is started.
3. Clock switch circuitry waits for a falling edge of the current clock.
4. The current clock is held low and the clock switch circuitry waits for a rising edge in the new clock.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

See [Figure 5-3](#) 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 5-2](#).

Start-up delay specifications are located in [Table 25-8](#), "Oscillator Parameters".

**FIGURE 5-3: INTERNAL OSCILLATOR SWITCH TIMING**



# PIC16(L)F1507

## 5.3 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 Words
- Internal Oscillator Block (INTOSC)

### 5.3.1 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 Words.
- 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 5-2](#).

### 5.3.2 CLOCK SWITCHING BEFORE SLEEP

When clock switching from an old clock to a new clock is requested just prior to entering Sleep mode, it is necessary to confirm that the switch is complete before the `SLEEP` instruction is executed. Failure to do so may result in an incomplete switch and consequential loss of the system clock altogether. Clock switching is confirmed by monitoring the clock status bits in the OSCSTAT register. Switch confirmation can be accomplished by sensing that the ready bit for the new clock is set or the ready bit for the old clock is cleared. For example, when switching between the internal oscillator with the PLL and the internal oscillator without the PLL, monitor the PLLR bit. When PLLR is set, the switch to 32 MHz operation is complete. Conversely, when PPLR is cleared, the switch from 32 MHz operation to the selected internal clock is complete.

**TABLE 5-2: OSCILLATOR SWITCHING DELAYS**

Switch From	Switch To	Oscillator Delay
Any clock source	LFINTOSC	1 cycle of each clock source
	HFINTOSC	2 $\mu$ s (approx.)
	ECH, ECM, ECL	2 cycles



## 5.4 Register Definitions: Oscillator Control

**REGISTER 5-1: 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

1111 = 16 MHz
1110 = 8 MHz
1101 = 4 MHz
1100 = 2 MHz
1011 = 1 MHz
1010 = 500 kHz <sup>(1)</sup>
1001 = 250 kHz <sup>(1)</sup>
1000 = 125 kHz <sup>(1)</sup>
0111 = 500 kHz (default upon Reset)
0110 = 250 kHz
0101 = 125 kHz
0100 = 62.5 kHz
001x = 31.25 kHz
000x = 31 kHz LF

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

bit 1-0        **SCS<1:0>:** System Clock Select bits

1x = Internal oscillator block
01 = Reserved
00 = Clock determined by FOSC<1:0> in Configuration Words.

**Note 1:** Duplicate frequency derived from HFINTOSC.

# PIC16(L)F1507

**REGISTER 5-2: OSCSTAT: OSCILLATOR STATUS REGISTER**

U-0	U-0	U-0	R-0/q	U-0	U-0	R-0/q	R-0/q
—	—	—	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 7-5            **Unimplemented:** Read as '0'
- 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 is not stable, the Start-up Oscillator is driving INTOSC

**TABLE 5-3: 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>		49
OSCSTAT	—	—	—	HFIOFR	—	—	LFIOFR	HFIOFS	50

**Legend:** — = unimplemented location, read as '0'. Shaded cells are not used by clock sources.

**TABLE 5-4: 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	—	—	—	—	CLKOUTEN	BOREN<1:0>		—	38
	7:0	CP	MCLRE	PWRTE	WDTE<1:0>		—	FOSC<1:0>		

**Legend:** — = unimplemented location, read as '0'. Shaded cells are not used by clock sources.

## 6.0 RESETS

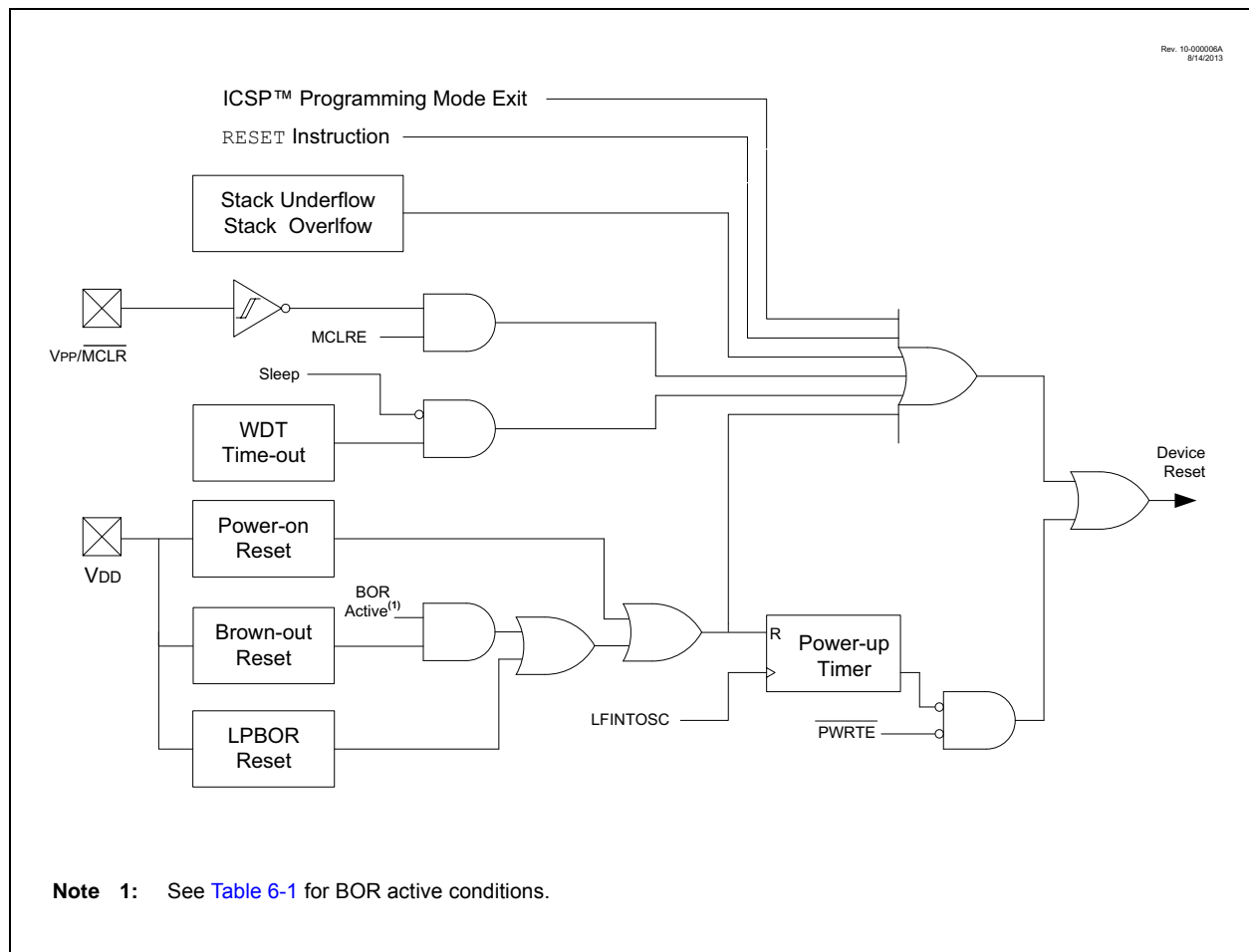
There are multiple ways to reset this device:

- 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 VDD 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 6-1.

**FIGURE 6-1: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT**



# PIC16(L)F1507

## 6.1 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 MCLR features can be used to extend the start-up period until all device operation conditions have been met.

### 6.1.1 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 PWRTE bit in Configuration Words.

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).

## 6.2 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 Words. 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 6-1](#) for more information.

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

A VDD noise rejection filter prevents the BOR from triggering on small events. If VDD falls below Vpor for a duration greater than parameter TBORDC, the device will reset. See [Figure 6-2](#) for more information.

**TABLE 6-1: BOR OPERATING MODES**

BOREN<1:0>	SBOREN	Device Mode	BOR Mode	Instruction Execution upon: Release of POR or Wake-up from Sleep
11	x	X	Active	Waits for BOR ready <sup>(1)</sup> (BORRDY = 1)
10	x	Awake	Active	Waits for BOR ready (BORRDY = 1)
		Sleep	Disabled	
01	1	X	Active	Waits for BOR ready <sup>(1)</sup> (BORRDY = 1)
	0	X	Disabled	Begins immediately (BORRDY = x)
00	x	X	Disabled	

**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.

### 6.2.1 BOR IS ALWAYS ON

When the BOREN bits of Configuration Words are programmed 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.

### 6.2.2 BOR IS OFF IN SLEEP

When the BOREN bits of Configuration Words are programmed 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.

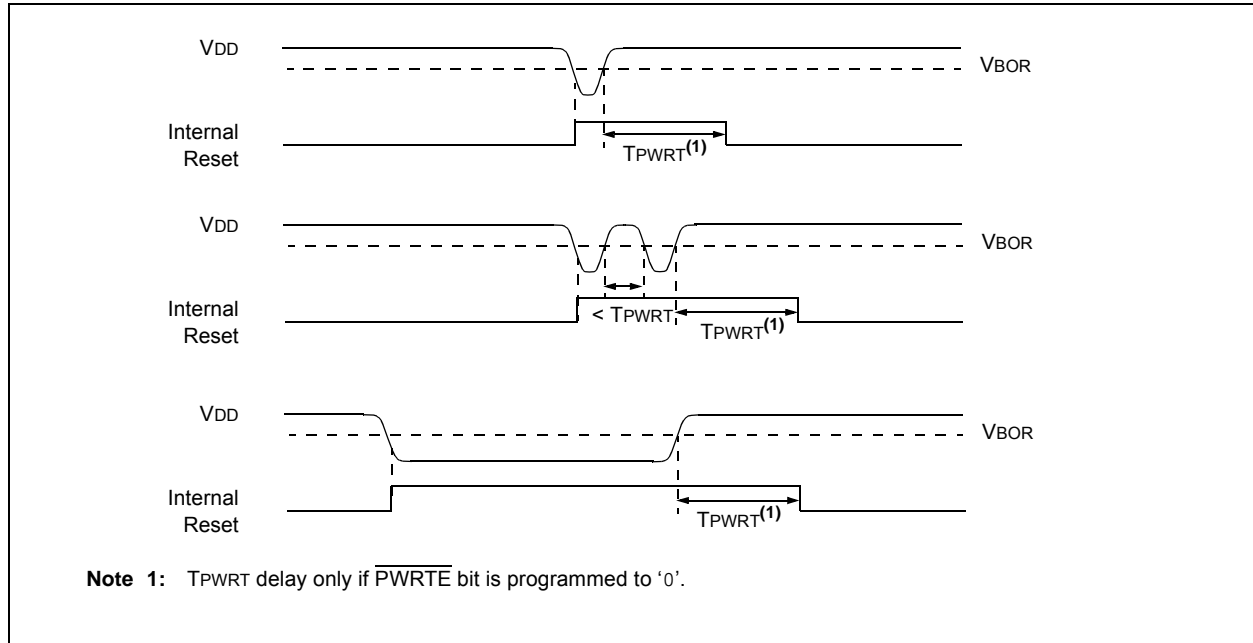
### 6.2.3 BOR CONTROLLED BY SOFTWARE

When the BOREN bits of Configuration Words are programmed 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 6-2: BROWN-OUT SITUATIONS**



## 6.3 Register Definitions: BOR Control

**REGISTER 6-1: 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 Words = 01:

- 1 = BOR Enabled
- 0 = BOR Disabled

If BOREN <1:0> in Configuration Words ≠ 01:

SBOREN is read/write, but has no effect on the BOR

bit 6 **BORFS:** Brown-Out Reset Fast Start bit<sup>(1)</sup>

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

If BOREN <1:0> = 11 (Always on) or BOREN <1:0> = 00 (Always off)

BORFS is Read/Write, but has no effect.

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

**Note 1:** BOREN <1:0> bits are located in Configuration Words.

# PIC16(L)F1507

## 6.4 Low-Power Brown-Out Reset (LPBOR)

The Low-Power Brown-out Reset (LPBOR) operates like the BOR to detect low voltage conditions on the VDD pin. When too low of a voltage is detected, the device is held in Reset. When this occurs, a register bit (BOR) is changed to indicate that a BOR Reset has occurred. The BOR bit in PCON is used for both BOR and the LPBOR. Refer to [Register 6-2](#).

The LPBOR voltage threshold (Lapboard) has a wider tolerance than the BOR (Vpor), but requires much less current (LPBOR current) to operate. The LPBOR is intended for use when the BOR is configured as disabled (BOREN = 00) or disabled in Sleep mode (BOREN = 10).

Refer to [Figure 6-1](#) to see how the LPBOR interacts with other modules.

### 6.4.1 ENABLING LPBOR

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

## 6.5 $\overline{\text{MCLR}}$

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 Words and the LVP bit of Configuration Words ([Table 6-2](#)).

**TABLE 6-2:  $\overline{\text{MCLR}}$  CONFIGURATION**

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

### 6.5.1 $\overline{\text{MCLR}}$ 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.

### 6.5.2 $\overline{\text{MCLR}}$ 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 11.3 “PORTA Registers”](#) for more information.

## 6.6 Watchdog Timer (WDT) Reset

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

## 6.7 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 6-4](#) for default conditions after a RESET instruction has occurred.

## 6.8 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 Words. See [Section 3.5.2 “Overflow/Underflow Reset”](#) for more information.

## 6.9 Programming Mode Exit

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

## 6.10 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  $\overline{\text{PWRT}}$  bit of Configuration Words.

## 6.11 Start-up Sequence

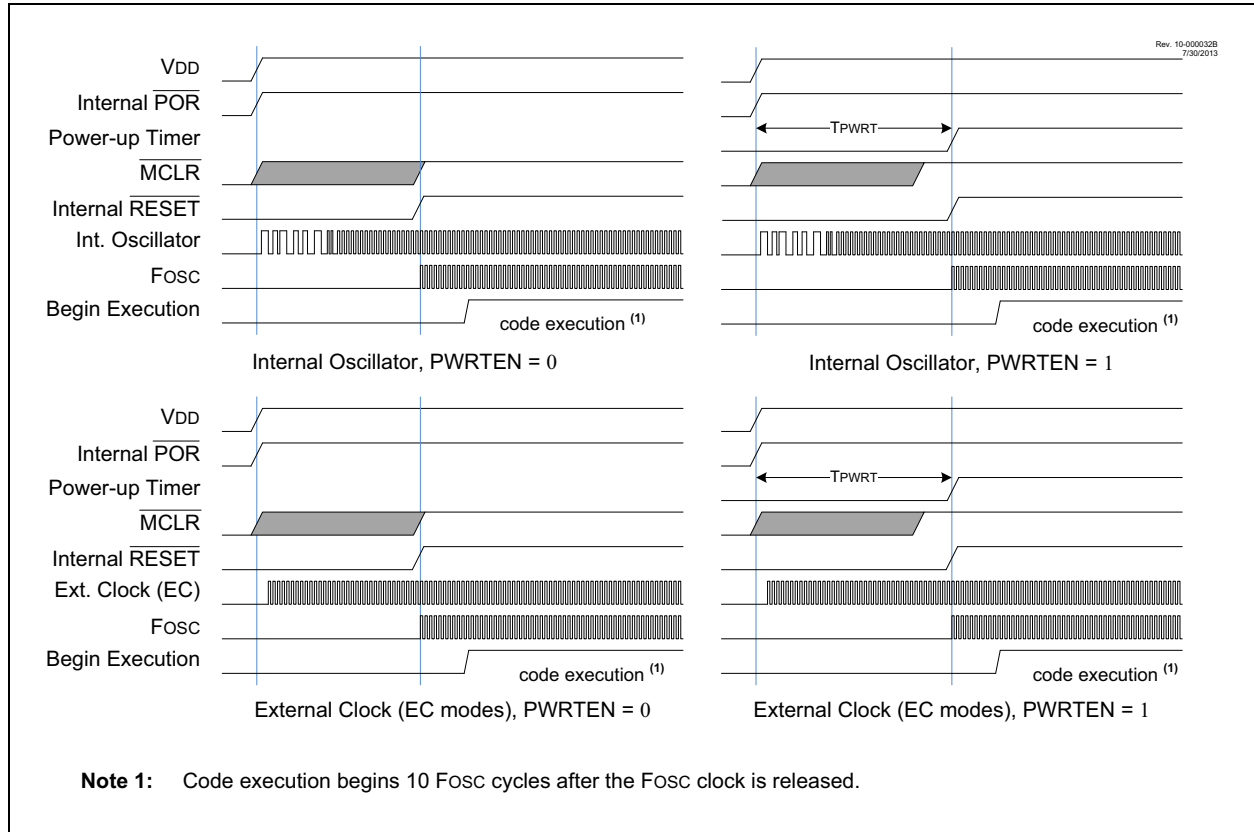
Upon the release of a POR or BOR, the following must occur before the device will begin executing:

1. Power-up Timer runs to completion (if enabled).
2.  $\overline{\text{MCLR}}$  must be released (if enabled).

The total time-out will vary based on oscillator configuration and Power-up Timer configuration. See [Section 5.0 “Oscillator Module”](#) for more information.

The Power-up Timer runs independently of  $\overline{\text{MCLR}}$  Reset. If  $\overline{\text{MCLR}}$  is kept low long enough, the Power-up Timer will expire. Upon bringing  $\overline{\text{MCLR}}$  high, the device will begin execution after 10 Fosc cycles (see [Figure 6-3](#)). This is useful for testing purposes or to synchronize more than one device operating in parallel.

**FIGURE 6-3: RESET START-UP SEQUENCE**



# PIC16(L)F1507

## 6.12 Determining the Cause of a Reset

Upon any Reset, multiple bits in the STATUS and PCON registers are updated to indicate the cause of the Reset. Table 6-3 and Table 6-4 show the Reset conditions of these registers.

**TABLE 6-3: RESET STATUS BITS AND THEIR SIGNIFICANCE**

STKOVF	STKUNF	RWDT	RMCLR	RI	POR	BOR	TO	PD	Condition
0	0	1	1	1	0	x	1	1	Power-on Reset
0	0	1	1	1	0	x	0	x	Illegal, $\overline{TO}$ is set on $\overline{POR}$
0	0	1	1	1	0	x	x	0	Illegal, $\overline{PD}$ is set on $\overline{POR}$
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	$\overline{MCLR}$ Reset during normal operation
u	u	u	0	u	u	u	1	0	$\overline{MCLR}$ 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 6-4: RESET CONDITION FOR SPECIAL REGISTERS**

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

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

**Note 1:** When the wake-up is due to an interrupt and the Global Interrupt 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.



## 6.13 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 ( $\overline{\text{RI}}$ )
- 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 6-2](#).

## 6.14 Register Definitions: Power Control

**REGISTER 6-2: 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

-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	<p><b>STKOVF:</b> Stack Overflow Flag bit</p> <p>1 = A Stack Overflow occurred</p> <p>0 = A Stack Overflow has not occurred or cleared by firmware</p>
bit 6	<p><b>STKUNF:</b> Stack Underflow Flag bit</p> <p>1 = A Stack Underflow occurred</p> <p>0 = A Stack Underflow has not occurred or cleared by firmware</p>
bit 5	<p><b>Unimplemented:</b> Read as '0'</p>
bit 4	<p><b><math>\overline{\text{RWDT}}</math>:</b> Watchdog Timer Reset Flag bit</p> <p>1 = A Watchdog Timer Reset has not occurred or set by firmware</p> <p>0 = A Watchdog Timer Reset has occurred (cleared by hardware)</p>
bit 3	<p><b><math>\overline{\text{RMCLR}}</math>:</b> MCLR Reset Flag bit</p> <p>1 = A <math>\overline{\text{MCLR}}</math> Reset has not occurred or set by firmware</p> <p>0 = A <math>\overline{\text{MCLR}}</math> Reset has occurred (cleared by hardware)</p>
bit 2	<p><b><math>\overline{\text{RI}}</math>:</b> RESET Instruction Flag bit</p> <p>1 = A RESET instruction has not been executed or set by firmware</p> <p>0 = A RESET instruction has been executed (cleared by hardware)</p>
bit 1	<p><b><math>\overline{\text{POR}}</math>:</b> Power-On Reset Status bit</p> <p>1 = No Power-on Reset occurred</p> <p>0 = A Power-on Reset occurred (must be set in software after a Power-on Reset occurs)</p>
bit 0	<p><b><math>\overline{\text{BOR}}</math>:</b> Brown-Out Reset Status bit</p> <p>1 = No Brown-out Reset occurred</p> <p>0 = A Brown-out Reset occurred (must be set in software after a Power-on Reset or Brown-out Reset occurs)</p>

# PIC16(L)F1507

**TABLE 6-5: 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	53
PCON	STKOVF	STKUNF	—	$\overline{RWDT}$	$\overline{RMCLR}$	$\overline{RI}$	$\overline{POR}$	$\overline{BOR}$	57
STATUS	—	—	—	$\overline{TO}$	$\overline{PD}$	Z	DC	C	17
WDTCON	—	—	WDTPS<4:0>					SWDTEN	77

**Legend:** — = unimplemented bit, reads as '0'. Shaded cells are not used by Resets.

**TABLE 6-6: SUMMARY OF CONFIGURATION WORD WITH RESETS**

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	—	—	—	—	$\overline{CLKOUTEN}$	BOREN<1:0>		—	38
	7:0	$\overline{CP}$	MCLRE	$\overline{PWRTE}$	WDTE<1:0>		—	FOSC<1:0>		
CONFIG2	13:8	—	—	LVP	$\overline{DEBUG}$	$\overline{LPBOR}$	BORV	STVREN	—	39
	7:0	—	—	—	—	—	—	WRT<1:0>		

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

## 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.

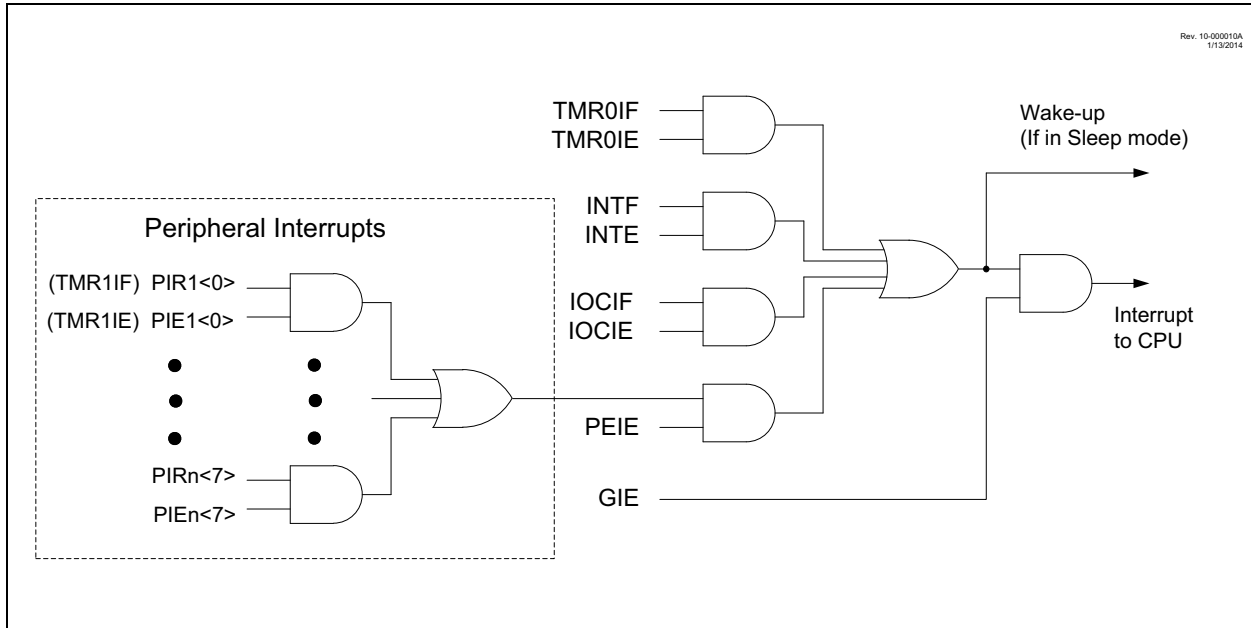
This chapter contains the following information for Interrupts:

- 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](#).

**FIGURE 7-1: INTERRUPT LOGIC**



# PIC16(L)F1507

---

## 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, PIE2 and PIE3 registers)

The INTCON, PIR1, PIR2 and PIR3 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.

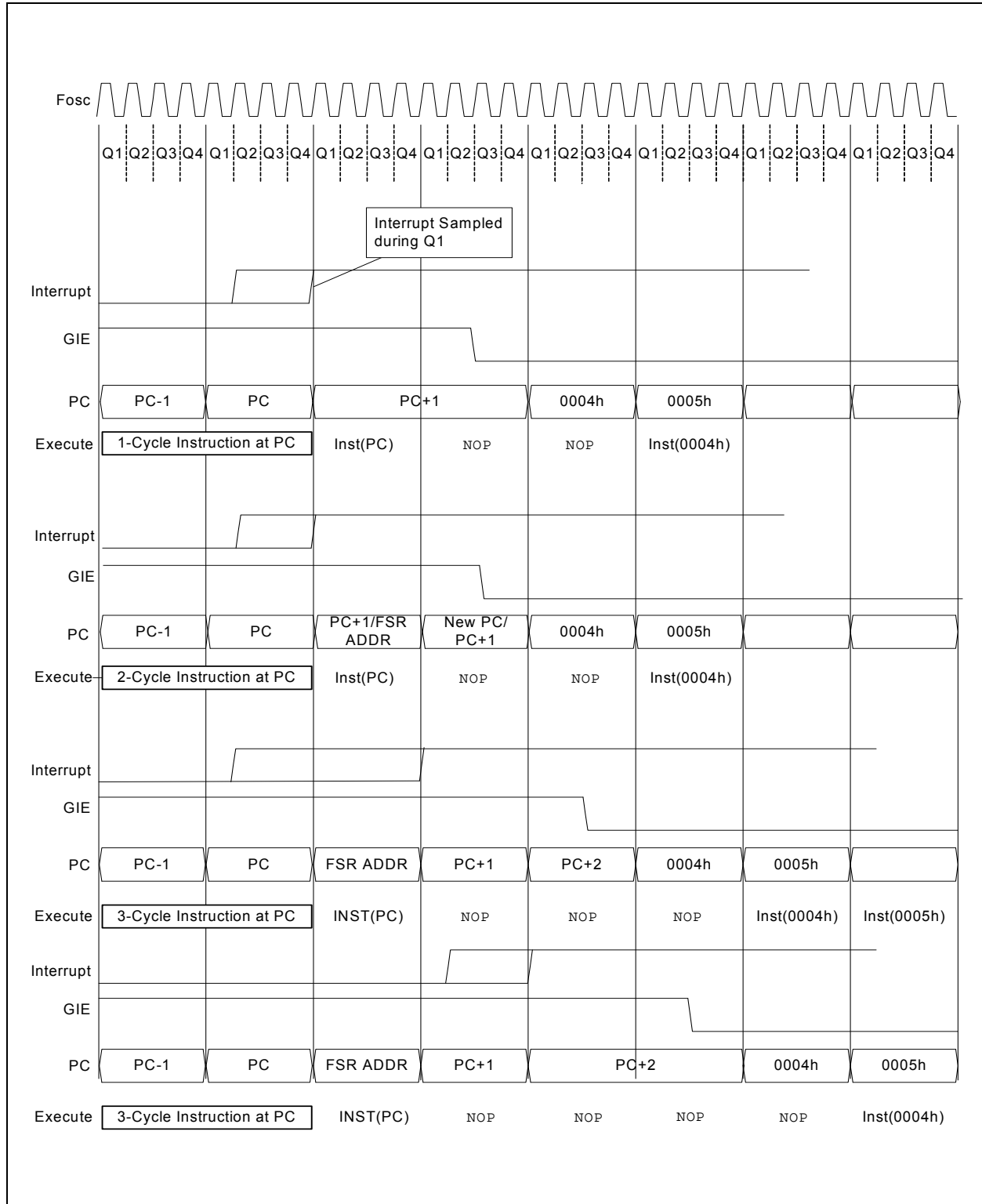
**Note 1:** Individual interrupt flag bits are set, regardless of the state of any other enable bits.

**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

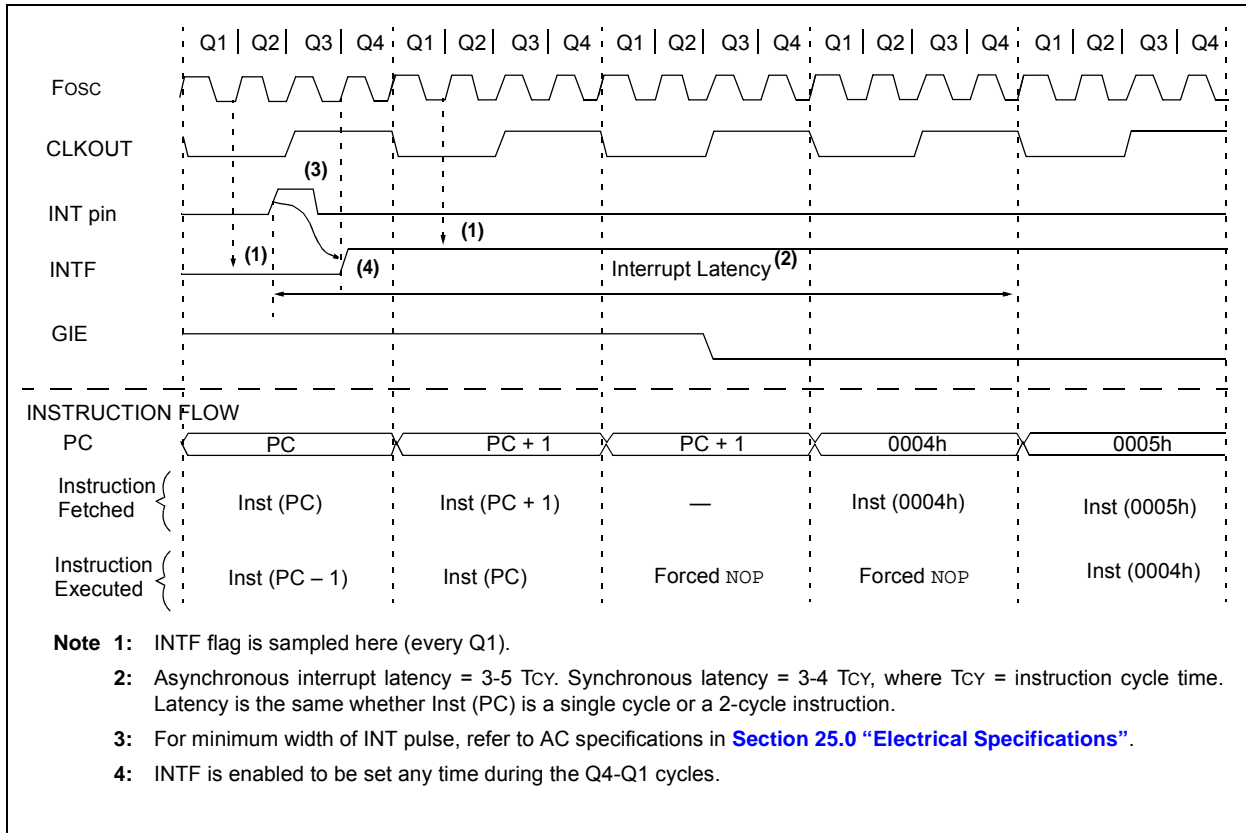
Interrupt latency is defined as the time from when the interrupt event occurs to the time code execution at the interrupt vector begins. The latency for synchronous interrupts is three or four instruction cycles. For asynchronous interrupts, the latency is three to five instruction cycles, depending on when the interrupt occurs. See [Figure 7-2](#) and [Figure 7-3](#) for more details.

**FIGURE 7-2: INTERRUPT LATENCY**



# PIC16(L)F1507

**FIGURE 7-3: INT PIN INTERRUPT TIMING**



## 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 [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  $\overline{TO}$  and  $\overline{PD}$ )
- 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.

# PIC16(L)F1507

## 7.6 Register Definitions: Interrupt Control

### 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 <sup>(1)</sup>	PEIE <sup>(2)</sup>	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF <sup>(3)</sup>
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<sup>(1)</sup>  
 1 = Enables all active interrupts  
 0 = Disables all interrupts
- bit 6      **PEIE:** Peripheral Interrupt Enable bit<sup>(2)</sup>  
 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 Enable bit  
 1 = Enables the interrupt-on-change  
 0 = Disables the interrupt-on-change
- 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<sup>(3)</sup>  
 1 = When at least one of the interrupt-on-change pins changed state  
 0 = None of the interrupt-on-change pins have changed state

- Note 1:** Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the Global Interrupt Enable bit, GIE of the INTCON register. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.
- 2:** Bit PEIE of the INTCON register must be set to enable any peripheral interrupt.
- 3:** The IOCIF Flag bit is read-only and cleared when all the interrupt-on-change flags in the IOCxF registers have been cleared by software.



## 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	R/W-0/0	R/W-0/0
TMR1GIE	ADIE	—	—	—	—	TMR2IE	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:** Analog-to-Digital Converter (ADC) Interrupt Enable bit  
1 = Enables the ADC interrupt  
0 = Disables the ADC interrupt
- bit 5-2    **Unimplemented:** Read as '0'
- bit 1      **TMR2IE:** TMR2 to PR2 Match Interrupt Enable bit  
1 = Enables the Timer2 to PR2 match interrupt  
0 = Disables the Timer2 to PR2 match interrupt
- bit 0      **TMR1IE:** Timer1 Overflow Interrupt Enable bit  
1 = Enables the Timer1 overflow interrupt  
0 = Disables the Timer1 overflow interrupt

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

# PIC16(L)F1507

---

## 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
—	—	—	—	—	NCO1IE	—	—
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 **NCO1IE:** Numerically Controlled Oscillator Interrupt Enable bit

1 = Enables the NCO interrupt

0 = Disables the NCO interrupt

bit 1-0 **Unimplemented:** Read as '0'**Note:** Bit PEIE of the INTCON register must be set to enable any peripheral interrupt.

## REGISTER 7-4: PIE3: PERIPHERAL INTERRUPT ENABLE REGISTER 3

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0
—	—	—	—	—	—	CLC2IE	CLC1IE
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      **Unimplemented:** Read as '0'

bit 1      **CLC2IE:** Configurable Logic Block 2 Interrupt Enable bit

1 = Enables the CLC 2 interrupt

0 = Disables the CLC 2 interrupt

bit 0      **CLC1IE:** Configurable Logic Block 1 Interrupt Enable bit

1 = Enables the CLC 1 interrupt

0 = Disables the CLC 1 interrupt

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

# PIC16(L)F1507

## REGISTER 7-5: PIR1: PERIPHERAL INTERRUPT REQUEST REGISTER 1

R/W-0/0	R/W-0/0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0
TMR1GIF	ADIF	—	—	—	—	TMR2IF	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:** ADC Interrupt Flag bit  
          1 = Interrupt is pending  
          0 = Interrupt is not pending
- bit 5-2    **Unimplemented:** Read as '0'
- bit 1      **TMR2IF:** Timer2 to PR2 Interrupt Flag bit  
          1 = Interrupt is pending  
          0 = Interrupt is not pending
- bit 0      **TMR1IF:** Timer1 Overflow Interrupt Flag bit  
          1 = Interrupt is pending  
          0 = Interrupt is not pending

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

## REGISTER 7-6: PIR2: PERIPHERAL INTERRUPT REQUEST REGISTER 2

U-0	U-0	U-0	U-0	U-0	R/W-0/0	U-0	U-0
—	—	—	—	—	NCO1IF	—	—
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      **NCO1IF:** Numerically Controlled Oscillator Flag bit

1 = Interrupt is pending

0 = Interrupt is not pending

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

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

# PIC16(L)F1507

## REGISTER 7-7: PIR3: PERIPHERAL INTERRUPT REQUEST REGISTER 3

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0
—	—	—	—	—	—	CLC2IF	CLC1IF
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      **Unimplemented:** Read as '0'

bit 1      **CLC2IF:** Configurable Logic Block 2 Interrupt Flag bit

1 = Interrupt is pending

0 = Interrupt is not pending

bit 0      **CLC1IF:** Configurable Logic Block 1 Interrupt Flag bit

1 = Interrupt is pending

0 = Interrupt is not pending

**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.

**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	IOCFIE	TMR0IF	INTF	IOCFIF	64
OPTION_REG	$\overline{\text{WPUEN}}$	INTEDG	TMR0CS	TMR0SE	PSA	PS<2:0>			133
PIE1	TMR1GIE	ADIE	—	—	—	—	TMR2IE	TMR1IE	65
PIE2	—	—	—	—	—	NCO1IE	—	—	66
PIE3	—	—	—	—	—	—	CLC2IE	CLC1IE	67
PIR1	TMR1GIF	ADIF	—	—	—	—	TMR2IF	TMR1IF	68
PIR2	—	—	—	—	—	NCO1IF	—	—	69
PIR3	—	—	—	—	—	—	CLC2IF	CLC1IF	70

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

# PIC16(L)F1507

## 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.  $\overline{PD}$  bit of the STATUS register is cleared.
3.  $\overline{TO}$  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. Timer1 and peripherals that operate from Timer1 continue operation in Sleep when the Timer1 clock source selected is:
  - LFINTOSC
  - T1CKI
7. ADC is unaffected, if the dedicated FRC oscillator is selected.
8. I/O ports maintain the status they had before `SLEEP` was executed (driving high, low or high-impedance).
9. 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
- CWG, NCO and CLC modules using HFINTOSC

I/O pins that are high-impedance inputs should be pulled to  $V_{DD}$  or  $V_{SS}$  externally to avoid switching currents caused by floating inputs.

Examples of internal circuitry that might be sourcing current include the FVR module. See [Section 13.0 “Fixed Voltage Reference \(FVR\)”](#) for more information on this module.

### 8.1 Wake-up from Sleep

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

1. External Reset input on  $\overline{MCLR}$  pin, 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 6.12 “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

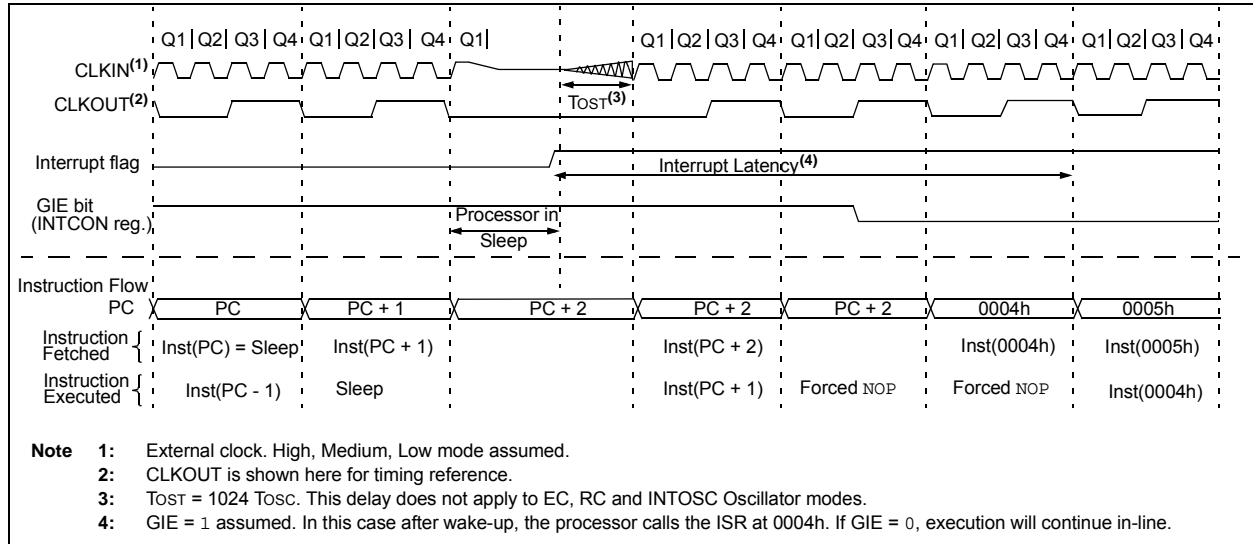
When global interrupts are disabled (GIE cleared) and any interrupt source has both its interrupt enable bit and interrupt flag bit set, one of the following will occur:

- 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
  - $\overline{TO}$  bit of the STATUS register will not be set
  - $\overline{PD}$  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
  - $\overline{TO}$  bit of the STATUS register will be set
  - $\overline{PD}$  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  $\overline{PD}$  bit. If the  $\overline{PD}$  bit is set, the `SLEEP` instruction was executed as a `NOP`.



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



## 8.2 Low-Power Sleep Mode

This device contains an internal Low Dropout (LDO) voltage regulator, which allows the device I/O pins to operate at voltages up to 5.5V while the internal device logic operates at a lower voltage. The LDO and its associated reference circuitry must remain active when the device is in Sleep mode.

Low-Power Sleep mode allows the user to optimize the operating current in Sleep. Low-Power Sleep mode can be selected by setting the VREGPM bit of the VREGCON register, putting the LDO and reference circuitry in a low-power state whenever the device is in Sleep.

### 8.2.1 SLEEP CURRENT VS. WAKE-UP TIME

In the Default Operating mode, the LDO and reference circuitry remain in the normal configuration while in Sleep. The device is able to exit Sleep mode quickly since all circuits remain active. In Low-Power Sleep mode, when waking up from Sleep, an extra delay time is required for these circuits to return to the normal configuration and stabilize.

The Low-Power Sleep mode is beneficial for applications that stay in Sleep mode for long periods of time. The Normal mode is beneficial for applications that need to wake from Sleep quickly and frequently.

### 8.2.2 PERIPHERAL USAGE IN SLEEP

Some peripherals that can operate in Sleep mode will not operate properly with the Low-Power Sleep mode selected. The LDO will remain in the Normal Power mode when those peripherals are enabled. The Low-Power Sleep mode is intended for use with these peripherals:

- Brown-out Reset (BOR)
- Watchdog Timer (WDT)
- External interrupt pin/Interrupt-on-change pins
- Timer1 (with external clock source)

The Complementary Waveform Generator (CWG), the Numerically Controlled Oscillator (NCO) and the Configurable Logic Cell (CLC) modules can utilize the HFINTOSC oscillator as either a clock source or as an input source. Under certain conditions, when the HFINTOSC is selected for use with the CWG, NCO or CLC modules, the HFINTOSC will remain active during Sleep. This will have a direct effect on the Sleep mode current.

Please refer to sections [Section 20.5 “Operation During Sleep”](#), [21.7 “Operation In Sleep”](#) and [22.10 “Operation During Sleep”](#) for more information.

**Note:** The PIC16LF1507 does not have a configurable Low-Power Sleep mode. PIC16LF1507 is an unregulated device and is always in the lowest power state when in Sleep, with no wake-up time penalty. This device has a lower maximum  $V_{DD}$  and I/O voltage than the PIC16F1507. See [Section 25.0 “Electrical Specifications”](#) for more information.

# PIC16(L)F1507

## 8.3 Register Definitions: Voltage Regulator Control

**REGISTER 8-1: VREGCON: VOLTAGE REGULATOR CONTROL REGISTER<sup>(1)</sup>**

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-1/1
—	—	—	—	—	—	VREGPM	Reserved
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            **Unimplemented:** Read as '0'  
bit 1              **VREGPM:** Voltage Regulator Power Mode Selection bit  
                    1 = Low-Power Sleep mode enabled in Sleep<sup>(2)</sup>  
                            Draws lowest current in Sleep, slower wake-up  
                    0 = Normal Power mode enabled in Sleep<sup>(2)</sup>  
                            Draws higher current in Sleep, faster wake-up  
bit 0              **Reserved:** Read as '1'. Maintain this bit set.

**Note 1:** PIC16F1507 only.  
**2:** See [Section 25.0 “Electrical Specifications”](#).

**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	IOCFIE	TMR0IF	INTF	IOCFIF	64
IOCAF	—	—	IOCAF5	IOCAF4	IOCAF3	IOCAF2	IOCAF1	IOCAF0	110
IOCAN	—	—	IOCAN5	IOCAN4	IOCAN3	IOCAN2	IOCAN1	IOCAN0	110
IOCAP	—	—	IOCAP5	IOCAP4	IOCAP3	IOCAP2	IOCAP1	IOCAP0	110
IOCBF	IOCBF7	IOCBF6	IOCBF5	IOCBF4	—	—	—	—	111
IOCBN	IOCBN7	IOCBN6	IOCBN5	IOCBN4	—	—	—	—	111
IOCBP	IOCBP7	IOCBP6	IOCBP5	IOCBP4	—	—	—	—	111
PIE1	TMR1GIE	ADIE	—	—	—	—	TMR2IE	TMR1IE	65
PIE2	—	—	—	—	—	NCO1IE	—	—	66
PIE3	—	—	—	—	—	—	CLC2IE	CLC1IE	67
PIR1	TMR1GIF	ADIF	—	—	—	—	TMR2IF	TMR1IF	68
PIR2	—	—	—	—	—	NCO1IF	—	—	69
PIR3	—	—	—	—	—	—	CLC2IF	CLC1IF	70
STATUS	—	—	—	$\overline{TO}$	$\overline{PD}$	Z	DC	C	17
WDTCON	—	—	WDTPS<4:0>					SWDTEN	77

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used in Power-Down mode.

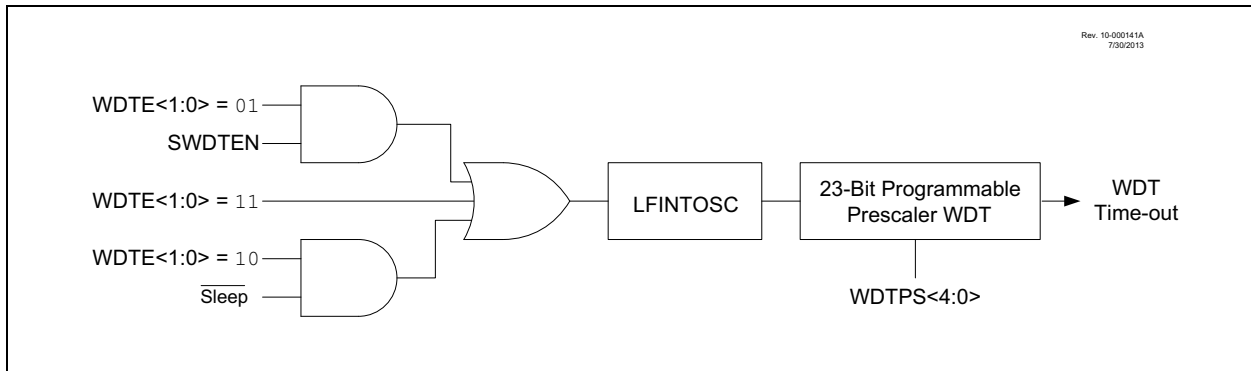
## 9.0 WATCHDOG TIMER (WDT)

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 (nominal)
- Multiple Reset conditions
- Operation during Sleep

**FIGURE 9-1: WATCHDOG TIMER BLOCK DIAGRAM**



# PIC16(L)F1507

## 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 25.0 “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 Words. See [Table 9-1](#).

### 9.2.1 WDT IS ALWAYS ON

When the WDTE bits of Configuration Words 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 Words 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 Words 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	X	Disabled
00	x	X	Disabled

**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 = INTOSC, EXTCLK	
Change INTOSC divider (IRCF bits)	Unaffected

## 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
- 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  $\overline{TO}$  and  $\overline{PD}$  bits in the STATUS register are changed to indicate the event. The  $\overline{RWDT}$  bit in the PCON register can also be used. See [Section 3.0 “Memory Organization”](#) for more information.

## 9.6 Register Definitions: Watchdog Timer Control

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

Bit Value = Prescale Rate

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

•  
•  
•

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

10010 = 1:8388608 ( $2^{23}$ ) (Interval 256s nominal)

10001 = 1:4194304 ( $2^{22}$ ) (Interval 128s nominal)

10000 = 1:2097152 ( $2^{21}$ ) (Interval 64s nominal)

01111 = 1:1048576 ( $2^{20}$ ) (Interval 32s nominal)

01110 = 1:524288 ( $2^{19}$ ) (Interval 16s nominal)

01101 = 1:262144 ( $2^{18}$ ) (Interval 8s nominal)

01100 = 1:131072 ( $2^{17}$ ) (Interval 4s nominal)

01011 = 1:65536 (Interval 2s nominal) (Reset value)

01010 = 1:32768 (Interval 1s nominal)

01001 = 1:16384 (Interval 512 ms nominal)

01000 = 1:8192 (Interval 256 ms nominal)

00111 = 1:4096 (Interval 128 ms nominal)

00110 = 1:2048 (Interval 64 ms nominal)

00101 = 1:1024 (Interval 32 ms nominal)

00100 = 1:512 (Interval 16 ms nominal)

00011 = 1:256 (Interval 8 ms nominal)

00010 = 1:128 (Interval 4 ms nominal)

00001 = 1:64 (Interval 2 ms nominal)

00000 = 1:32 (Interval 1 ms nominal)

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

If WDTE<1:0> = 1x:

This bit is ignored.

If WDTE<1:0> = 01:

1 = WDT is turned on

0 = WDT is turned off

If WDTE<1:0> = 00:

This bit is ignored.

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

# PIC16(L)F1507

**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>		49
PCON	STKOVF	STKUNF	—	RWD $\bar{T}$	RMCLR	RI	POR	BOR	57
STATUS	—	—	—	TO	PD	Z	DC	C	17
WDTCON	—	—	WDTPS<4:0>					SWDTEN	77

**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	—	—	—	—	CLKOUTEN	BOREN<1:0>		—	38
	7:0	CP	MCLRE	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 ( $\overline{CP}$  bit in Configuration Words) and write protection (WRT<1:0> bits in Configuration Words).

Code protection ( $\overline{CP} = 0$ )<sup>(1)</sup>, 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  $\overline{CP}$  bit of Configuration Words.

### 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)F1507	16	16

# PIC16(L)F1507

## 10.2.1 READING THE FLASH PROGRAM MEMORY

To read a program memory location, the user must:

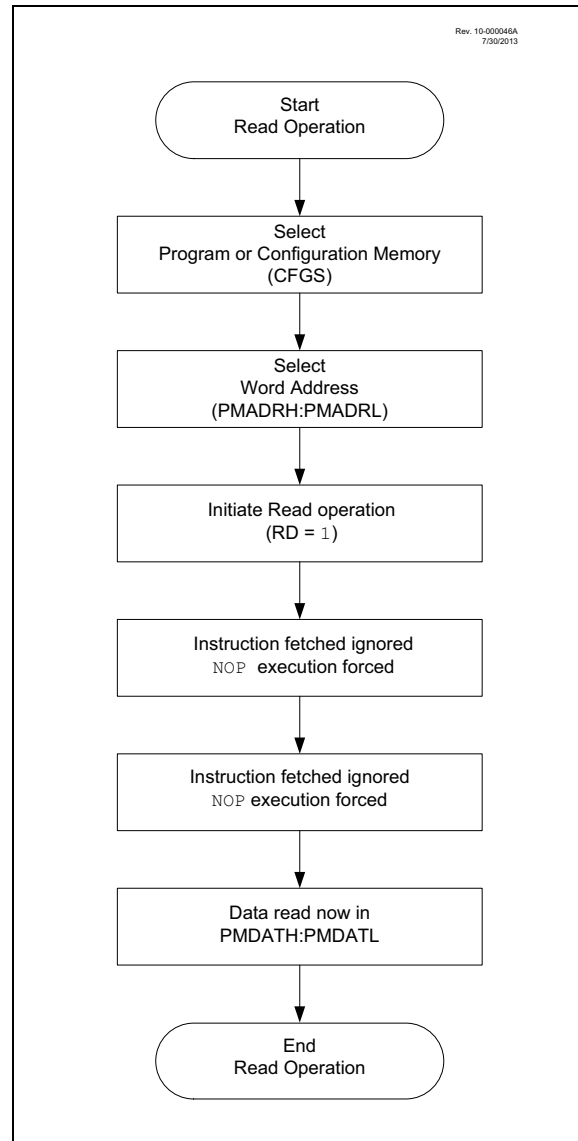
1. Write the desired address to the PMADRH:PMADRL register pair.
2. Clear the CFGS bit of the PMCON1 register.
3. Then, set control bit RD of the PMCON1 register.

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.

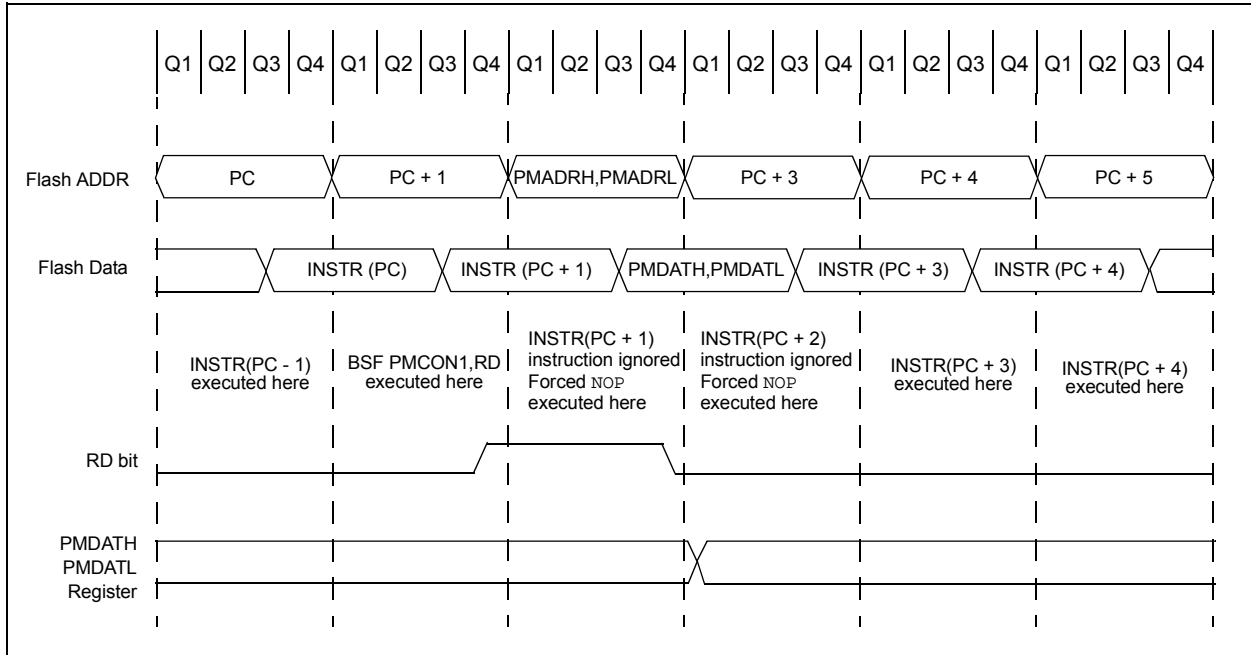
**Note:** The two instructions following a program memory read are required to be NOPs. This prevents the user from executing a 2-cycle instruction on the next instruction after the RD bit is set.

**FIGURE 10-1: FLASH PROGRAM MEMORY READ FLOWCHART**





**FIGURE 10-2: FLASH PROGRAM MEMORY READ CYCLE EXECUTION**



**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    ;
  MOVWF   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
    
```

# PIC16(L)F1507

## 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

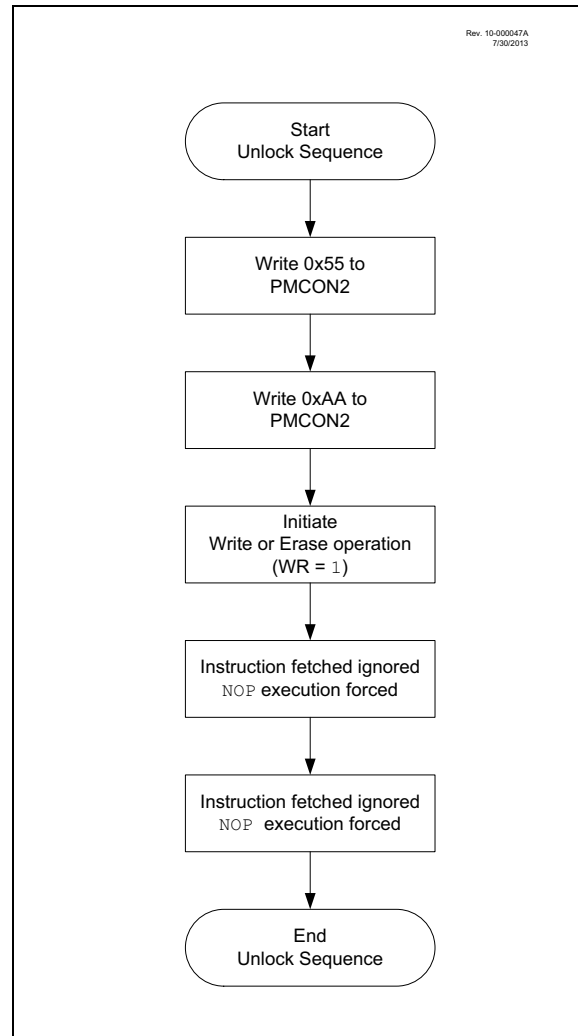
The unlock sequence consists of the following steps:

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**



## 10.2.3 ERASING FLASH PROGRAM MEMORY

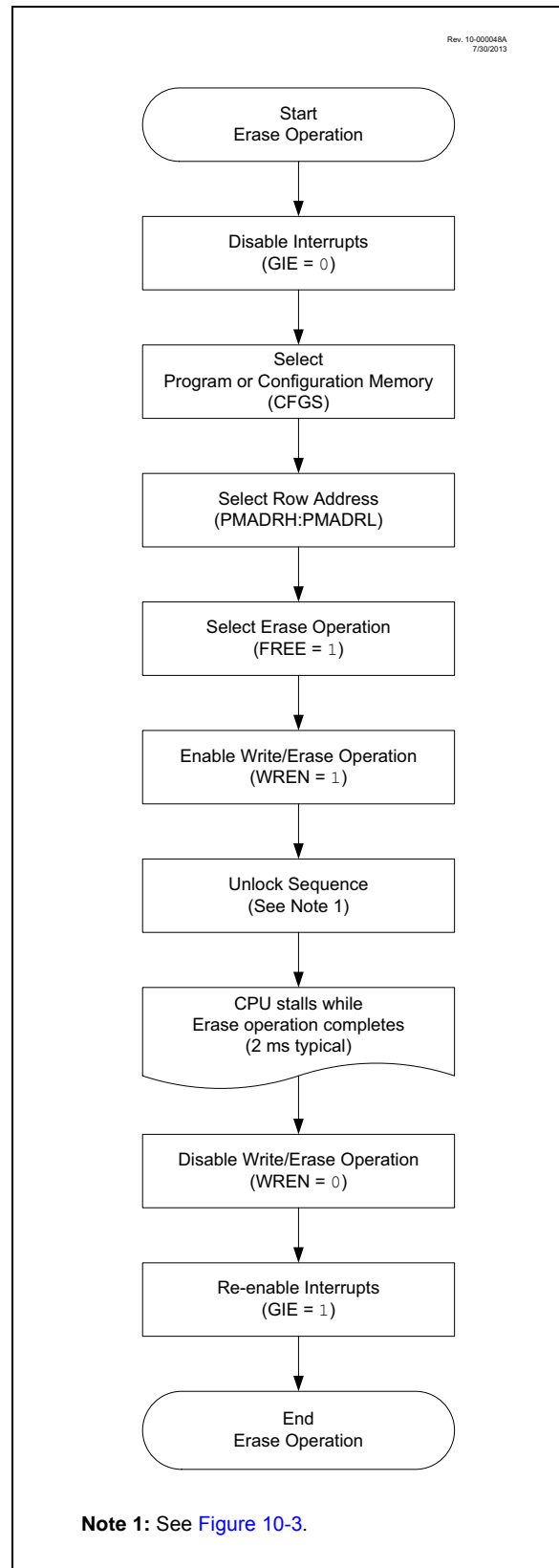
While executing code, program memory can only be erased by rows. To erase a row:

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 immediately following the WR bit set instruction. 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**



# PIC16(L)F1507

## 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,CFG5       ; Not configuration space
        BSF    PMCON1,FREE       ; Specify an erase operation
        BSF    PMCON1,WREN       ; Enable writes

        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
```

Required  
Sequence

## 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 16 write latches) for more details.

The write latches are aligned to the Flash row address boundary defined by the upper 10-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.

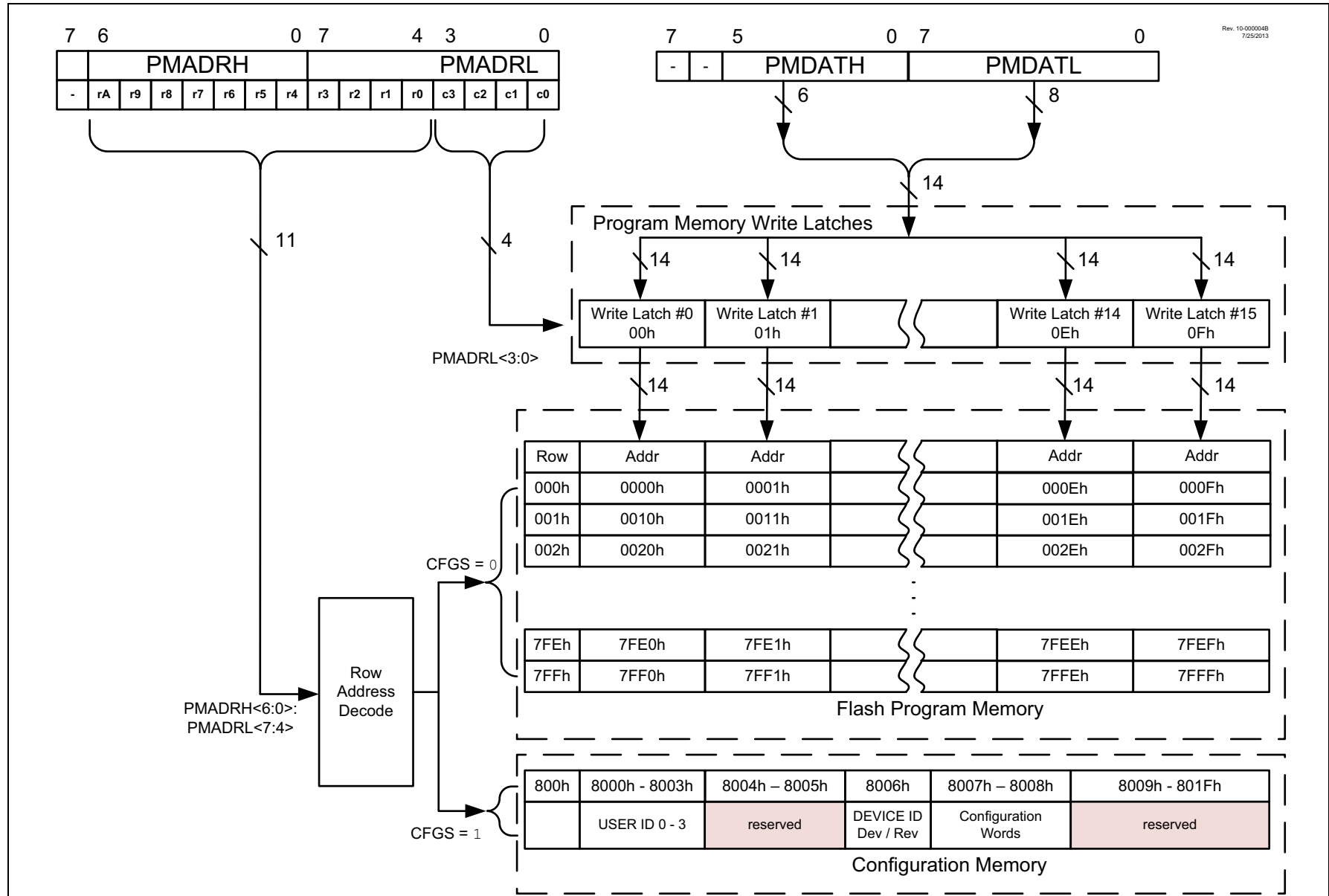
**Note:** The special unlock sequence is required to load a write latch with data or initiate a Flash programming operation. If the unlock sequence is interrupted, writing to the latches or program memory will not be initiated.

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.

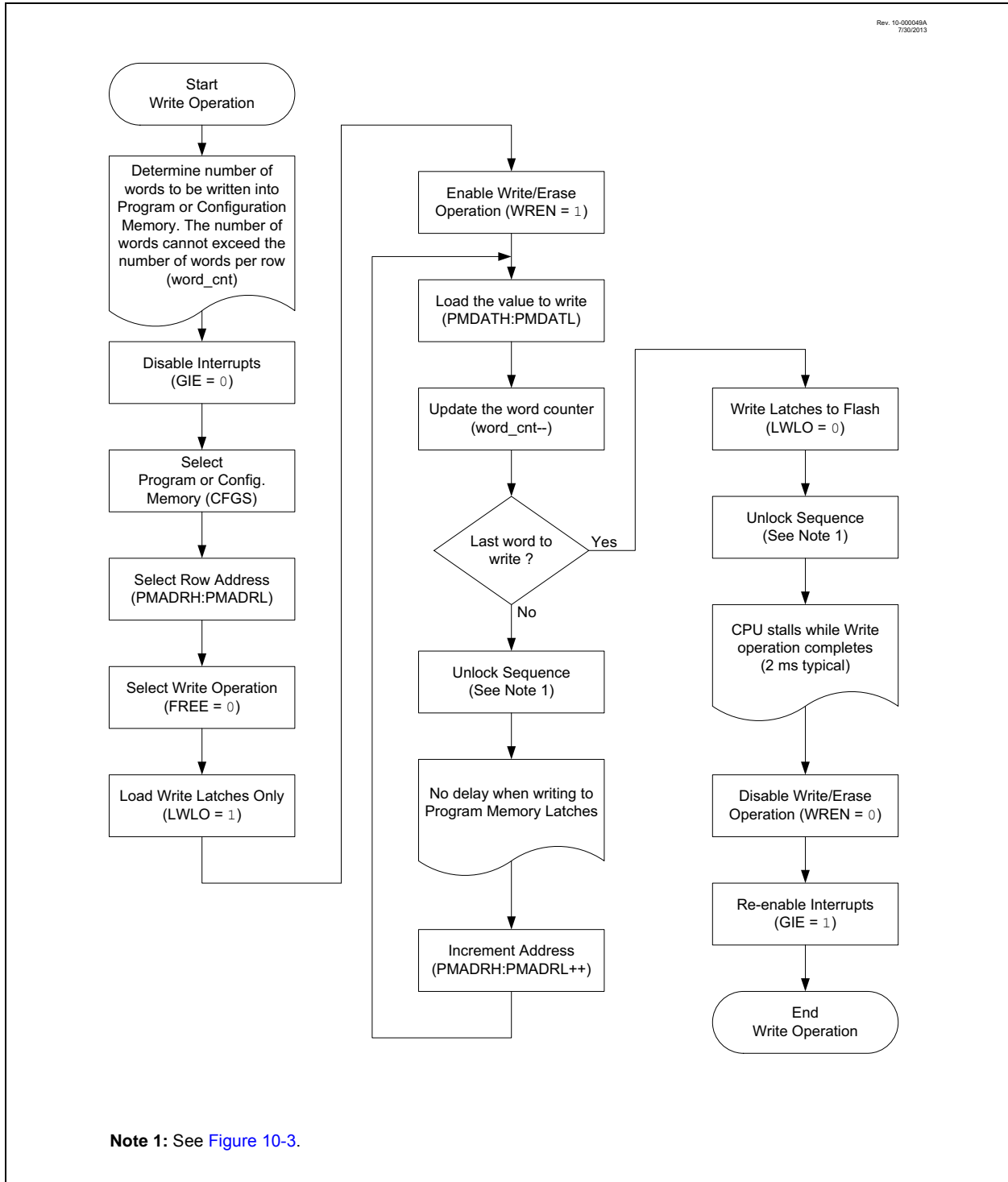
**Note:** The program memory write latches are reset to the blank state (0x3FFF) at the completion of every write or erase operation. As a result, it is not necessary to load all the program memory write latches. Unloaded latches will remain in the blank state.

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 16 WRITE LATCHES**



**FIGURE 10-6: FLASH MEMORY WRITE FLOWCHART**



# PIC16(L)F1507

## EXAMPLE 10-3: WRITING TO FLASH PROGRAM MEMORY (16 WRITE LATCHES)

```
; This write routine assumes the following:
; 1. 32 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,CFGSS    ; 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  0x0F         ; Check if we're on the last of 16 addresses
      ANDLW  0x0F         ;
      BTFSC  STATUS,Z     ; Exit if last of 16 words,
      GOTO   START_WRITE  ;

      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

      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 is complete
                          ; after write processor continues with 3rd instruction

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

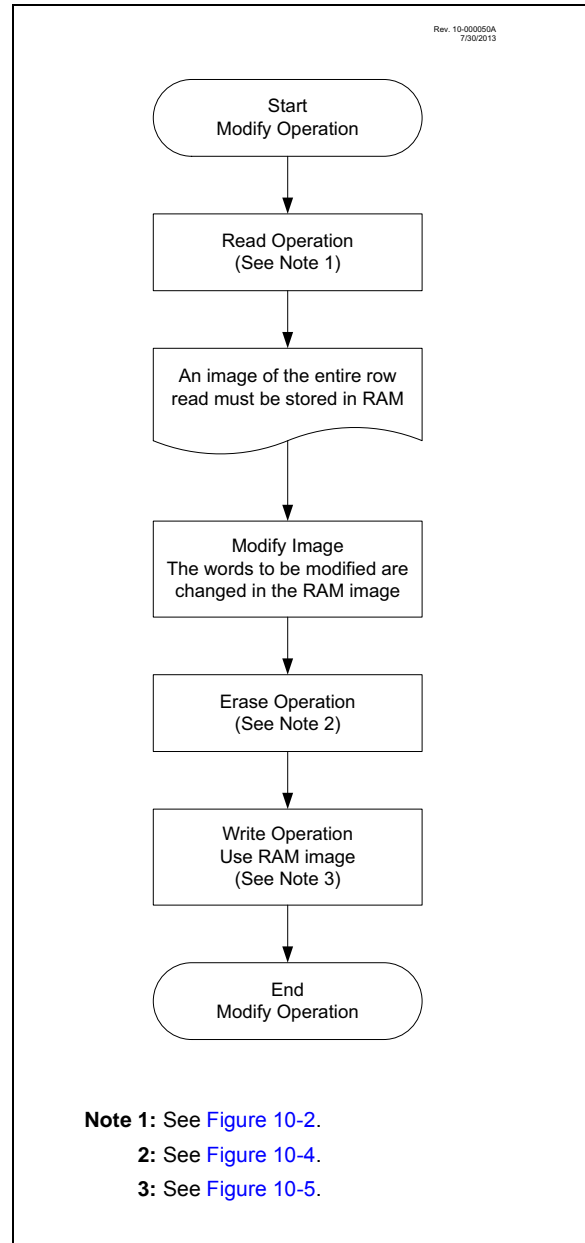


## 10.3 Modifying Flash Program Memory

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**



# PIC16(L)F1507

## 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  $CFG5 = 1$  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 PMDATA:PMDATL register pair is cleared, reading back '0's.

**TABLE 10-2: USER ID, DEVICE ID AND CONFIGURATION WORD ACCESS (CFG5 = 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

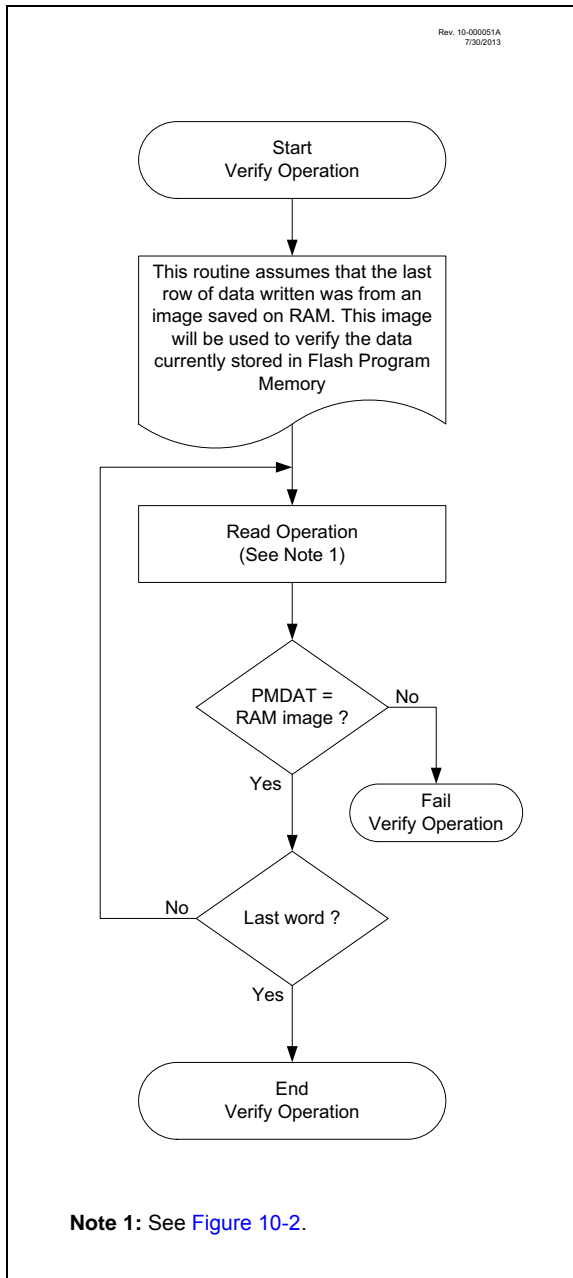
BSF     PMCON1,CFG5      ; Select Configuration Space
BCF     INTCON,GIE       ; Disable interrupts
BSF     PMCON1,RD        ; Initiate read
NOP     ; Executed (See Figure 10-2)
NOP     ; Ignored (See Figure 10-2)
BSF     INTCON,GIE       ; Restore interrupts

MOVF    PMDATL,W         ; Get LSB of word
MOVWF   PROG_DATA_LO    ; Store in user location
MOVF    PMDATA,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**



# PIC16(L)F1507

## 10.6 Register Definitions: Flash Program Memory Control

### 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
—(1)	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

**Note 1:** Unimplemented, read as '1'.

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

U-1	R/W-0/0	R/W-0/0	R/W/HC-0/0	R/W/HC-x/q <sup>(2)</sup>	R/W-0/0	R/S/HC-0/0	R/S/HC-0/0
— <sup>(1)</sup>	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<sup>(3)</sup>  
 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 a 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).

# PIC16(L)F1507

**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
INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	64
PMCON1	_(1)	CFGS	LWLO	FREE	WRERR	WREN	WR	RD	93
PMCON2	Program Memory Control Register 2								94
PMADRL	PMADRL<7:0>								92
PMADRH	_(1)	PMADRH<6:0>							92
PMDATL	PMDATL<7:0>								92
PMDATH	—	—	PMDATH<5:0>						92

**Legend:** — = unimplemented location, read as '0'. Shaded cells are not used by Flash program memory.

**Note 1:** Unimplemented, read as '1'.

**TABLE 10-4: SUMMARY OF CONFIGURATION WORD WITH RESETS**

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	—	—	—	—	CLKOUTEN	BOREN<1:0>		—	38
	7:0	CP	MCLRE	PWRTE	WDTE<1:0>		—	FOSC<1:0>		
CONFIG2	13:8	—	—	LVP	DEBUG	LPBOR	BORV	STVREN	—	39
	7:0	—	—	—	—	—	WRT<1:0>			

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

## 11.0 I/O PORTS

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)

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.

**TABLE 11-1: PORT AVAILABILITY PER DEVICE**

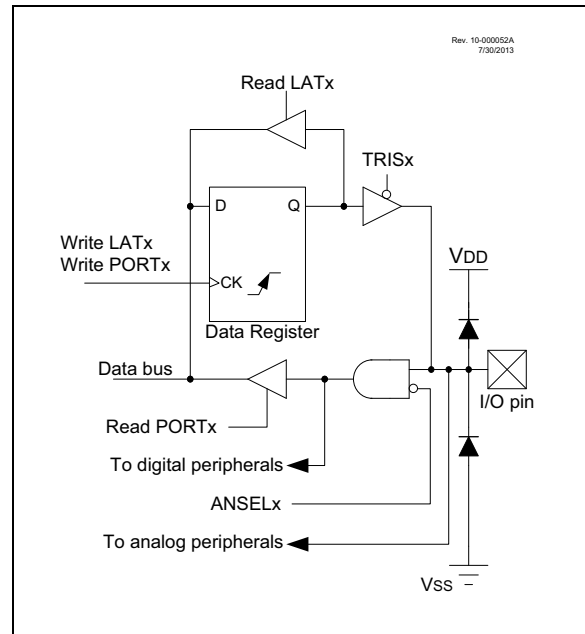
Device	PORTA	PORTB	PORTC
PIC16(L)F1507	•	•	•

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

A write operation to the LATx register has the same effect as a write to the corresponding PORTx register. A read of the LATx register reads of the values held in the I/O PORT latches, while a read of the PORTx 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**



# PIC16(L)F1507

## 11.1 Alternate Pin Function

The Alternate Pin Function Control (APFCON) register is used to steer specific peripheral input and output functions between different pins. The APFCON register is shown in [Register 11-1](#). For this device family, the following functions can be moved between different pins.

- CLC1
- NCO1

These bits have no effect on the values of any TRIS register. PORT and TRIS overrides will be routed to the correct pin. The unselected pin will be unaffected.

## 11.2 Register Definitions: Alternate Pin Function Control

### REGISTER 11-1: APFCON: ALTERNATE PIN FUNCTION CONTROL REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0
—	—	—	—	—	—	CLC1SEL	NCO1SEL
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      **Unimplemented:** Read as '0'bit 1      **CLC1SEL:** Pin Selection bit

1 = CLC1 function is on RC5

0 = CLC1 function is on RA2

bit 0      **NCO1SEL:** Pin Selection bit

1 = NCO1 function is on RC6

0 = NCO1 function is on RA1



## 11.3 PORTA Registers

### 11.3.1 DATA REGISTER

PORTA is a 6-bit wide, bidirectional port. The corresponding data direction register is TRISA (Register 11-3). 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 an I/O port.

Reading the PORTA register (Register 11-2) 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).

### 11.3.2 DIRECTION CONTROL

The TRISA register (Register 11-3) 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'.

### 11.3.3 ANALOG CONTROL

The ANSELA register (Register 11-5) 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.

#### EXAMPLE 11-1: INITIALIZING PORTA

```
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
```

### 11.3.4 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 and comparator 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 below in Table 11-2.

**TABLE 11-2: PORTA OUTPUT PRIORITY**

Pin Name	Function Priority <sup>(1)</sup>
RA0	ICSPDAT RA0
RA1	RA1
RA2	CLC1 <sup>(2)</sup> PWM3 RA2
RA3	None
RA4	CLKOUT RA4
RA5	RA5

- Note 1:** Priority listed from highest to lowest.  
**Note 2:** Default pin (see APFCON register).

# PIC16(L)F1507

## 11.4 Register Definitions: PORTA

### REGISTER 11-2: PORTA: PORTA REGISTER

U-0	U-0	R/W-x/x	R/W-x/x	R-x/x	R/W-x/x	R/W-x/x	R/W-x/x
—	—	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 7-6                      **Unimplemented:** Read as '0'  
bit 5-0                      **RA<5:0>:** PORTA I/O Value bits<sup>(1)</sup>  
                                    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 11-3: TRISA: PORTA TRI-STATE REGISTER

U-0	U-0	R/W-1/1	R/W-1/1	U-1	R/W-1/1	R/W-1/1	R/W-1/1
—	—	TRISA5	TRISA4	— <sup>(1)</sup>	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 7-6                      **Unimplemented:** Read as '0'  
bit 5-4                      **TRISA<5:4>:** PORTA Tri-State Control bit  
                                    1 = PORTA pin configured as an input (tri-stated)  
                                    0 = PORTA pin configured as an output  
bit 3                              **Unimplemented:** Read as '1'  
bit 2-0                      **TRISA<2:0>:** PORTA Tri-State Control bit  
                                    1 = PORTA pin configured as an input (tri-stated)  
                                    0 = PORTA pin configured as an output

**Note 1:** Unimplemented, read as '1'.

## REGISTER 11-4: LATA: PORTA DATA LATCH REGISTER

U-0	U-0	R/W-x/u	R/W-x/u	U-0	R/W-x/u	R/W-x/u	R/W-x/u
—	—	LATA5	LATA4	—	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 7-6      **Unimplemented:** Read as '0'
- bit 5-4      **LATA<5:4>:** RA<5:4> Output Latch Value bits<sup>(1)</sup>
- bit 3        **Unimplemented:** Read as '0'
- bit 2-0      **LATA<2:0>:** RA<2:0> Output Latch Value bits<sup>(1)</sup>

**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 11-5: ANSALA: PORTA ANALOG SELECT REGISTER

U-0	U-0	U-0	R/W-1/1	U-0	R/W-1/1	R/W-1/1	R/W-1/1
—	—	—	ANSA4	—	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 7-5      **Unimplemented:** Read as '0'
- bit 4        **ANSA4:** Analog Select between Analog or Digital Function on pins RA4, respectively
  - 1 = Analog input. Pin is assigned as analog input<sup>(1)</sup>. Digital input buffer disabled.
  - 0 = Digital I/O. Pin is assigned to port or digital special function.
- bit 3        **Unimplemented:** Read as '0'
- bit 2-0      **ANSA<2:0>:** Analog Select between Analog or Digital Function on pins RA<2:0>, respectively
  - 1 = Analog input. Pin is assigned as analog input<sup>(1)</sup>. Digital input buffer disabled.
  - 0 = Digital I/O. Pin is assigned to port or digital special function.

**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.

# PIC16(L)F1507

**REGISTER 11-6: WPUA: WEAK PULL-UP PORTA 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
—	—	WPUA5	WPUA4	WPUA3	WPUA2	WPUA1	WPUA0
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                      **WPUA<5:0>:** Weak Pull-up Register bits<sup>(3)</sup>  
                                    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.  
**Note 2:** The weak pull-up device is automatically disabled if the pin is configured as an output.  
**Note 3:** For the WPUA3 bit, when MCLRE = 1, weak pull-up is internally enabled, but not reported here.

**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	—	—	—	ANSA4	—	ANSA2	ANSA1	ANSA0	99
APFCON	—	—	—	—	T1GSEL	—	CLC1SEL	NCO1SEL	96
LATA	—	—	LATA5	LATA4	—	LATA2	LATA1	LATA0	99
OPTION_REG	$\overline{\text{WPUEN}}$	INTEDG	TMR0CS	TMR0SE	PSA	PS<2:0>			133
PORTA	—	—	RA5	RA4	RA3	RA2	RA1	RA0	98
TRISA	—	—	TRISA5	TRISA4	— <sup>(1)</sup>	TRISA2	TRISA1	TRISA0	98
WPUA	—	—	WPUA5	WPUA4	WPUA3	WPUA2	WPUA1	WPUA0	100

**Legend:** x = unknown, u = unchanged, — = unimplemented locations read as '0'. Shaded cells are not used by PORTA.  
**Note 1:** Unimplemented, read as '1'.

**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>		—	38
	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.

## 11.5 PORTB Registers

### 11.5.1 DATA REGISTER

PORTB is a 4-bit wide, bidirectional port. The corresponding data direction register is TRISB (Register 11-8). Setting a TRISB bit (= 1) will make the corresponding PORTB pin an input (i.e., disable the output driver). Clearing a TRISB bit (= 0) will make the corresponding PORTB pin an output (i.e., enables output driver and puts 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 11-7) 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).

### 11.5.2 DIRECTION CONTROL

The TRISB register (Register 11-8) 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'.

### 11.5.3 ANALOG CONTROL

The ANSELB register (Register 11-10) 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 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 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.

### 11.5.4 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 functions, such as ADC and comparator 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 below in Table 11-5.

**TABLE 11-5: PORTB OUTPUT PRIORITY**

Pin Name	Function Priority <sup>(1)</sup>
RB4	RB4
RB5	RB5
RB6	RB6
RB7	RB7

- Note 1:** Priority listed from highest to lowest.  
**2:** Default pin (see APFCON register).

# PIC16(L)F1507

## 11.6 Register Definitions: PORTB

### REGISTER 11-7: PORTB: PORTB REGISTER

R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	U-0	U-0	U-0	U-0
RB7	RB6	RB5	RB4	—	—	—	—
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            **RB<7:4>**: PORTB I/O Value bits<sup>(1)</sup>  
1 = Port pin is  $\geq V_{IH}$   
0 = Port pin is  $\leq V_{IL}$

bit 3-0            **Unimplemented**: Read as '0'

**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: TRISB: PORTB TRI-STATE REGISTER

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	U-0	U-0	U-0	U-0
TRISB7	TRISB6	TRISB5	TRISB4	—	—	—	—
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            **RB<7:4>**: PORTB Tri-State Control bits  
1 = PORTB pin configured as an input (tri-stated)  
0 = PORTB pin configured as an output

bit 3-0            **Unimplemented**: Read as '0'

## REGISTER 11-9: LATB: PORTB DATA LATCH REGISTER

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	U-0	U-0	U-0	U-0
LATB7	LATB6	LATB5	LATB4	—	—	—	—
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      **LATB<7:4>**: RB<7:4> Output Latch Value bits<sup>(1)</sup>

bit 3-0      **Unimplemented**: Read as '0'

**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 11-10: ANSELB: PORTB ANALOG SELECT REGISTER

U-0	U-0	R/W-1/1	R/W-1/1	U-0	U-0	U-0	U-0
—	—	ANSB5	ANSB4	—	—	—	—
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 5-4      **ANSB<5:4>**: Analog Select between Analog or Digital Function on pins RB<5:4>, respectively  
 1 = Analog input. Pin is assigned as analog input<sup>(1)</sup>. Digital input buffer disabled.  
 0 = Digital I/O. Pin is assigned to port or digital special function.

bit 3-0      **Unimplemented**: Read as '0'

**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.

# PIC16(L)F1507

## REGISTER 11-11: WPUB: WEAK PULL-UP PORTB REGISTER<sup>(1),(2)</sup>

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	U-0	U-0	U-0	U-0
WPUB7	WPUB6	WPUB5	WPUB4	—	—	—	—
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                      **WPUB<7:4>**: Weak Pull-up Register bits

1 = Pull-up enabled

0 = Pull-up disabled

bit 3-0                      **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.

**Note 2:** The weak pull-up device is automatically disabled if the pin is 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	—	—	—	—	103
APFCON	—	—	—	—	—	—	CLC1SEL	NCO1SEL	96
LATB	LATB7	LATB6	LATB5	LATB4	—	—	—	—	103
OPTION_REG	$\overline{\text{WPUEN}}$	INTEDG	TMR0CS	TMR0SE	PSA	PS<2:0>			133
PORTB	RB7	RB6	RB5	RB4	—	—	—	—	102
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	—	—	—	—	102
WPUB	WPUB7	WPUB6	WPUB5	WPUB4	—	—	—	—	104

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

**Note 1:** Unimplemented, read as '1'.

**TABLE 11-7: SUMMARY OF CONFIGURATION WORD WITH PORTB**

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>		—	38
	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 PORTB.



## 11.7 PORTC Registers

### 11.7.1 DATA REGISTER

PORTC is a 8-bit wide, bidirectional port. The corresponding data direction register is TRISC (Register 11-13). Setting a TRISC bit (= 1) will make the corresponding PORTC pin an input (i.e., disable the output driver). 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 11-12) 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).

### 11.7.2 DIRECTION CONTROL

The TRISC register (Register 11-13) 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'.

### 11.7.3 ANALOG CONTROL

The ANSEL register (Register 11-15) is used to configure the Input mode of an I/O pin to analog. Setting the appropriate ANSEL 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 ANSEL 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 ANSEL 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.

### 11.7.4 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 11-8.

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 output priority list. 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 the output priority list.

**TABLE 11-8: PORTC OUTPUT PRIORITY**

Pin Name	Function Priority <sup>(1)</sup>
RC0	CLC2 RC0
RC1	NCO1 <sup>(2)</sup> PWM4 RC1
RC2	RC2
RC3	PWM2 RC3
RC4	CWG1B RC4
RC5	CWG1A CLC1 <sup>(3)</sup> PWM1 RC5
RC6	NCO1 <sup>(3)</sup> RC6
RC7	RC7

- Note 1:** Priority listed from highest to lowest.  
**Note 2:** Default pin (see APFCON register).  
**Note 3:** Alternate pin (see APFCON register).

# PIC16(L)F1507

## 11.8 Register Definitions: PORTC

### REGISTER 11-12: 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 = Port pin is  $\geq V_{IH}$   
0 = Port pin is  $\leq V_{IL}$

### REGISTER 11-13: 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 = PORTC pin configured as an input (tri-stated)  
0 = PORTC pin configured as an output

### REGISTER 11-14: 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<sup>(1)</sup>

**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 11-15: ANSELC: PORTC ANALOG SELECT REGISTER

R/W-1/1	R/W-1/1	U-0	U-0	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
ANSC7	ANSC6	—	—	ANSC3	ANSC2	ANSC1	ANSC0
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      **ANSC<7:6>**: Analog Select between Analog or Digital Function on pins RC<7:6>, respectively  
 1 = Analog input. Pin is assigned as analog input<sup>(1)</sup>. Digital input buffer disabled.  
 0 = Digital I/O. Pin is assigned to port or digital special function.
- bit 5-4      **Unimplemented**: Read as '0'
- bit 3-0      **ANSC<3:0>**: Analog Select between Analog or Digital Function on pins RC<3:0>, respectively  
 1 = Analog input. Pin is assigned as analog input<sup>(1)</sup>. Digital input buffer disabled.  
 0 = Digital I/O. Pin is assigned to port or digital special function.

**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 11-9: 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
ANSELC	ANSC7	ANSC6	—	—	ANSC3	ANSC2	ANSC1	ANSC0	<a href="#">107</a>
LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	<a href="#">106</a>
PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	<a href="#">106</a>
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	<a href="#">106</a>

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

## 12.0 INTERRUPT-ON-CHANGE

The PORTA and 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 port pin, or combination of port 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 port 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 port pin, a rising edge detector and a falling edge detector are present. To enable a pin to detect a rising edge, the associated bit of the IOCxP register is set. To enable a pin to detect a falling edge, the associated bit of the IOCxN register is set.

A pin can be configured to detect rising and falling edges simultaneously by setting both associated bits of the IOCxP and IOCxN registers, respectively.

## 12.3 Interrupt Flags

The IOCAFx and IOCBFx bits located in the IOCAF and IOCBF registers, respectively, are status flags that correspond to the interrupt-on-change pins of the associated port. 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 IOCAFx and IOCBFx bits.

### 12.4 Clearing Interrupt Flags

The individual status flags, (IOCAFx and 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: CLEARING INTERRUPT FLAGS (PORTA EXAMPLE)

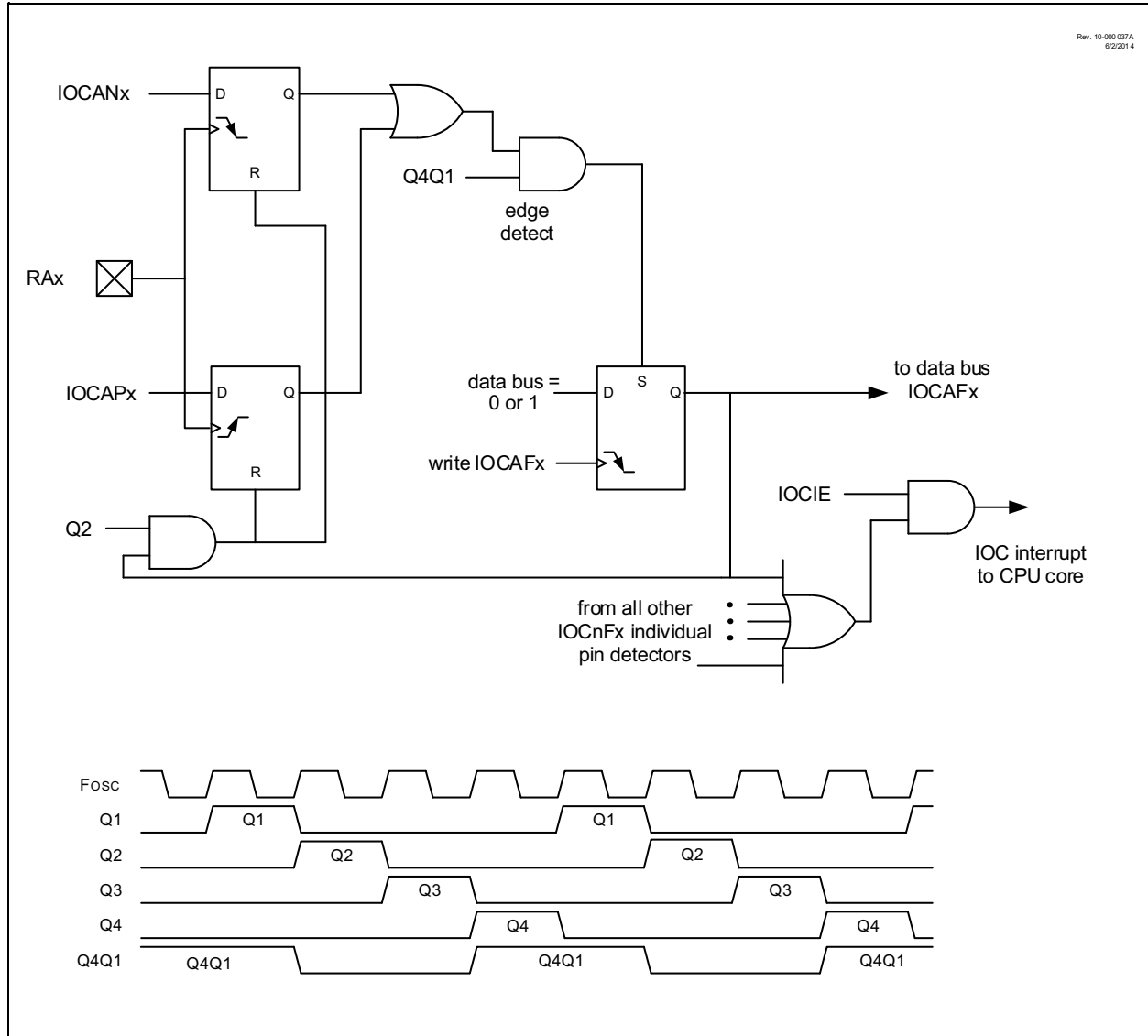
```
MOVLW 0xff
XORWF IOCAF, W
ANDWF IOCAF, F
```

### 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 IOCxF register will be updated prior to the first instruction executed out of Sleep.

**FIGURE 12-1: INTERRUPT-ON-CHANGE BLOCK DIAGRAM (PORTA EXAMPLE)**



# PIC16(L)F1507

## 12.6 Register Definitions: Interrupt-on-Change Control

### REGISTER 12-1: IOCAP: INTERRUPT-ON-CHANGE PORTA POSITIVE EDGE REGISTER

U-0	U-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
—	—	IOCAP5	IOCAP4	IOCAP3	IOCAP2	IOCAP1	IOCAP0
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                      **IOCAP<5:0>:** Interrupt-on-Change PORTA Positive Edge Enable bits  
1 = Interrupt-on-Change enabled on the pin for a positive going edge. IOCAF<sub>x</sub> bit and IOCIF flag will be set upon detecting an edge.  
0 = Interrupt-on-Change disabled for the associated pin.

### REGISTER 12-2: IOCAN: INTERRUPT-ON-CHANGE PORTA NEGATIVE EDGE REGISTER

U-0	U-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
—	—	IOCAN5	IOCAN4	IOCAN3	IOCAN2	IOCAN1	IOCAN0
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                      **IOCAN<5:0>:** Interrupt-on-Change PORTA Negative Edge Enable bits  
1 = Interrupt-on-Change enabled on the pin for a negative going edge. IOCAF<sub>x</sub> bit and IOCIF flag will be set upon detecting an edge.  
0 = Interrupt-on-Change disabled for the associated pin.

### REGISTER 12-3: IOCAF: INTERRUPT-ON-CHANGE PORTA FLAG REGISTER

U-0	U-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
—	—	IOCAF5	IOCAF4	IOCAF3	IOCAF2	IOCAF1	IOCAF0
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 7-6                      **Unimplemented:** Read as '0'

bit 5-0                      **IOCAF<5:0>:** Interrupt-on-Change PORTA Flag bits  
1 = An enabled change was detected on the associated pin.  
Set when IOCAP<sub>x</sub> = 1 and a rising edge was detected on RAX, or when IOCAN<sub>x</sub> = 1 and a falling edge was detected on RAX.  
0 = No change was detected, or the user cleared the detected change.

## REGISTER 12-4: IOCBP: INTERRUPT-ON-CHANGE PORTB POSITIVE EDGE REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	U-0	U-0	U-0
IOCBP7	IOCBP6	IOCBP5	IOCBP4	—	—	—	—
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                      **IOCBP<7:4>**: Interrupt-on-Change PORTB Positive Edge Enable bits  
1 = Interrupt-on-Change enabled on the pin for a positive going edge. IOCBFx bit and IOCIF flag will be set upon detecting an edge.  
0 = Interrupt-on-Change disabled for the associated pin.  
bit 3-0                      **Unimplemented**: Read as '0'

## REGISTER 12-5: IOCBN: INTERRUPT-ON-CHANGE PORTB NEGATIVE EDGE REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	U-0	U-0	U-0
IOCBN7	IOCBN6	IOCBN5	IOCBN4	—	—	—	—
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                      **IOCBN<7:4>**: Interrupt-on-Change PORTB Negative Edge Enable bits  
1 = Interrupt-on-Change enabled on the pin for a negative going edge. IOCBFx bit and IOCIF flag will be set upon detecting an edge.  
0 = Interrupt-on-Change disabled for the associated pin.  
bit 3-0                      **Unimplemented**: Read as '0'

## REGISTER 12-6: IOCBF: INTERRUPT-ON-CHANGE PORTB FLAG REGISTER

R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	U-0	U-0	U-0	U-0
IOCBF7	IOCBF6	IOCBF5	IOCBF4	—	—	—	—
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 7-4                      **IOCBF<7:4>**: Interrupt-on-Change PORTB 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.  
bit 3-0                      **Unimplemented**: Read as '0'

# PIC16(L)F1507

**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
ANSELA	—	—	—	ANSA4	—	ANSA2	ANSA1	ANSA0	99
INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	64
IOCAF	—	—	IOCAF5	IOCAF4	IOCAF3	IOCAF2	IOCAF1	IOCAF0	110
IOCAN	—	—	IOCAN5	IOCAN4	IOCAN3	IOCAN2	IOCAN1	IOCAN0	110
IOCAP	—	—	IOCAP5	IOCAP4	IOCAP3	IOCAP2	IOCAP1	IOCAP0	110
IOCBF	IOCBF7	IOCBF6	IOCBF5	IOCBF4	—	—	—	—	111
IOCBN	IOCBN7	IOCBN6	IOCBN5	IOCBN4	—	—	—	—	111
IOCBP	IOCBP7	IOCBP6	IOCBP5	IOCBP4	—	—	—	—	111
TRISA	—	—	TRISA5	TRISA4	— <sup>(1)</sup>	TRISA2	TRISA1	TRISA0	98
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	—	—	—	—	102

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

**Note 1:** Unimplemented, read as '1'.



## 13.0 FIXED VOLTAGE REFERENCE (FVR)

The Fixed Voltage Reference (FVR) is a stable voltage reference, independent of  $V_{DD}$ , with a nominal output level ( $V_{FVR}$ ) of 1.024V. The output of the FVR can be configured to supply a reference voltage to the following:

- ADC input channel

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

### 13.1 Independent Gain Amplifier

The output of the FVR supplied to the peripherals, (listed above), is routed through a programmable gain amplifier. Each amplifier can be programmed for a gain of 1x, 2x or 4x, to produce the three possible voltage levels.

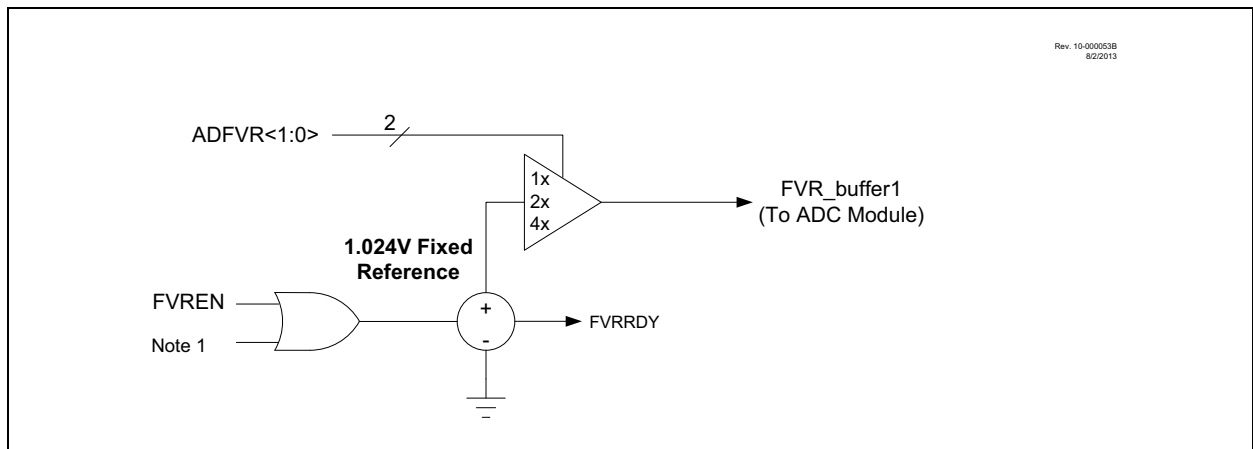
The ADFVR<1:0> bits of the FVRCON register are used to enable and configure the gain amplifier settings for the reference supplied to the ADC module. Reference [Section 15.0 “Analog-to-Digital Converter \(ADC\) Module”](#) for additional information.

To minimize current consumption when the FVR is disabled, the FVR buffers should be turned off by clearing the Buffer Gain Selection bits.

### 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 the FVR Stabilization Period characterization graph, [Figure 26-48](#).

**FIGURE 13-1: VOLTAGE REFERENCE BLOCK DIAGRAM**



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

Peripheral	Conditions	Description
HFINTOSC	FOSC<2:0> = 010 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.
LDO	All PIC16F1507 devices, when VREGPM = 1 and not in Sleep	The device runs off of the Low-Power Regulator when in Sleep mode.

# PIC16(L)F1507

## 13.3 Register Definitions: FVR Control

**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 <sup>(1)</sup>	FVRRDY <sup>(2)</sup>	TSEN <sup>(3)</sup>	TSRNG <sup>(3)</sup>	—	—	ADFVR<1:0> <sup>(1)</sup>	
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<sup>(1)</sup>  
           1 = Fixed Voltage Reference is enabled  
           0 = Fixed Voltage Reference is disabled
- bit 6      **FVRRDY:** Fixed Voltage Reference Ready Flag bit<sup>(2)</sup>  
           1 = Fixed Voltage Reference output is ready for use  
           0 = Fixed Voltage Reference output is not ready or not enabled
- bit 5      **TSEN:** Temperature Indicator Enable bit<sup>(3)</sup>  
           1 = Temperature Indicator is enabled  
           0 = Temperature Indicator is disabled
- bit 4      **TSRNG:** Temperature Indicator Range Selection bit<sup>(3)</sup>  
           1 =  $V_{OUT} = V_{DD} - 4V_T$  (High Range)  
           0 =  $V_{OUT} = V_{DD} - 2V_T$  (Low Range)
- bit 3-2    **Unimplemented:** Read as '0'
- bit 1-0    **ADFVR<1:0>:** ADC FVR Buffer Gain Selection bit<sup>(1)</sup>  
           11 = ADC FVR Buffer Gain is 4x, with output voltage = 4x VFVR (4.096V nominal)<sup>(4)</sup>  
           10 = ADC FVR Buffer Gain is 2x, with output voltage = 2x VFVR (2.048V nominal)<sup>(4)</sup>  
           01 = ADC FVR Buffer Gain is 1x, with output voltage = 1x VFVR (1.024V nominal)  
           00 = ADC FVR Buffer is off

- Note 1:** To minimize current consumption when the FVR is disabled, the FVR buffers should be turned off by clearing the Buffer Gain Selection bits.
- 2:** FVRRDY is always '1' for the PIC16F1507 devices.
- 3:** See [Section 14.0 "Temperature Indicator Module"](#) for additional information.
- 4:** Fixed Voltage Reference output cannot exceed VDD.

**TABLE 13-2: SUMMARY OF REGISTERS ASSOCIATED WITH THE 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	—	—	ADFVR<1:0>		<a href="#">114</a>

**Legend:** Shaded cells are unused by the Fixed Voltage Reference module.

## 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  $-40^{\circ}\text{C}$  and  $+85^{\circ}\text{C}$ . 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: $V_{OUT}$ RANGES

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

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

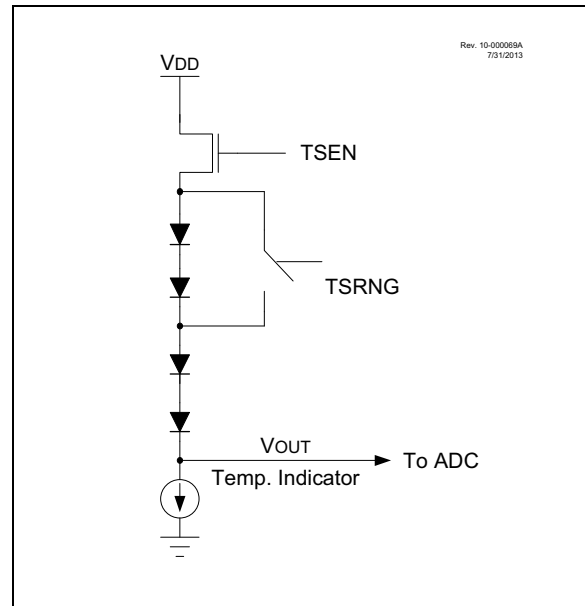
The temperature sense circuit is integrated with the Fixed Voltage Reference (FVR) module. See Section 13.0 "Fixed Voltage Reference (FVR)" for more information.

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  $V_{DD}$  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 $V_{DD}$

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,  $V_{DD}$ , must be high enough to ensure that the temperature circuit is correctly biased.

Table 14-1 shows the recommended minimum  $V_{DD}$  vs. range setting.

TABLE 14-1: RECOMMENDED  $V_{DD}$  VS. RANGE

Min. $V_{DD}$ , TSRNG = 1	Min. $V_{DD}$ , 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  $200\ \mu\text{s}$  after the ADC input multiplexer is connected to the temperature indicator output before the conversion is performed. In addition, the user must wait  $200\ \mu\text{s}$  between sequential conversions of the temperature indicator output.

# PIC16(L)F1507

---

---

**TABLE 14-2: SUMMARY OF REGISTERS ASSOCIATED WITH THE TEMPERATURE INDICATOR**

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	—	—	ADFVR<1:0>		<a href="#">114</a>

**Legend:** Shaded cells are unused by the temperature indicator module.

## 15.0 ANALOG-TO-DIGITAL CONVERTER (ADC) MODULE

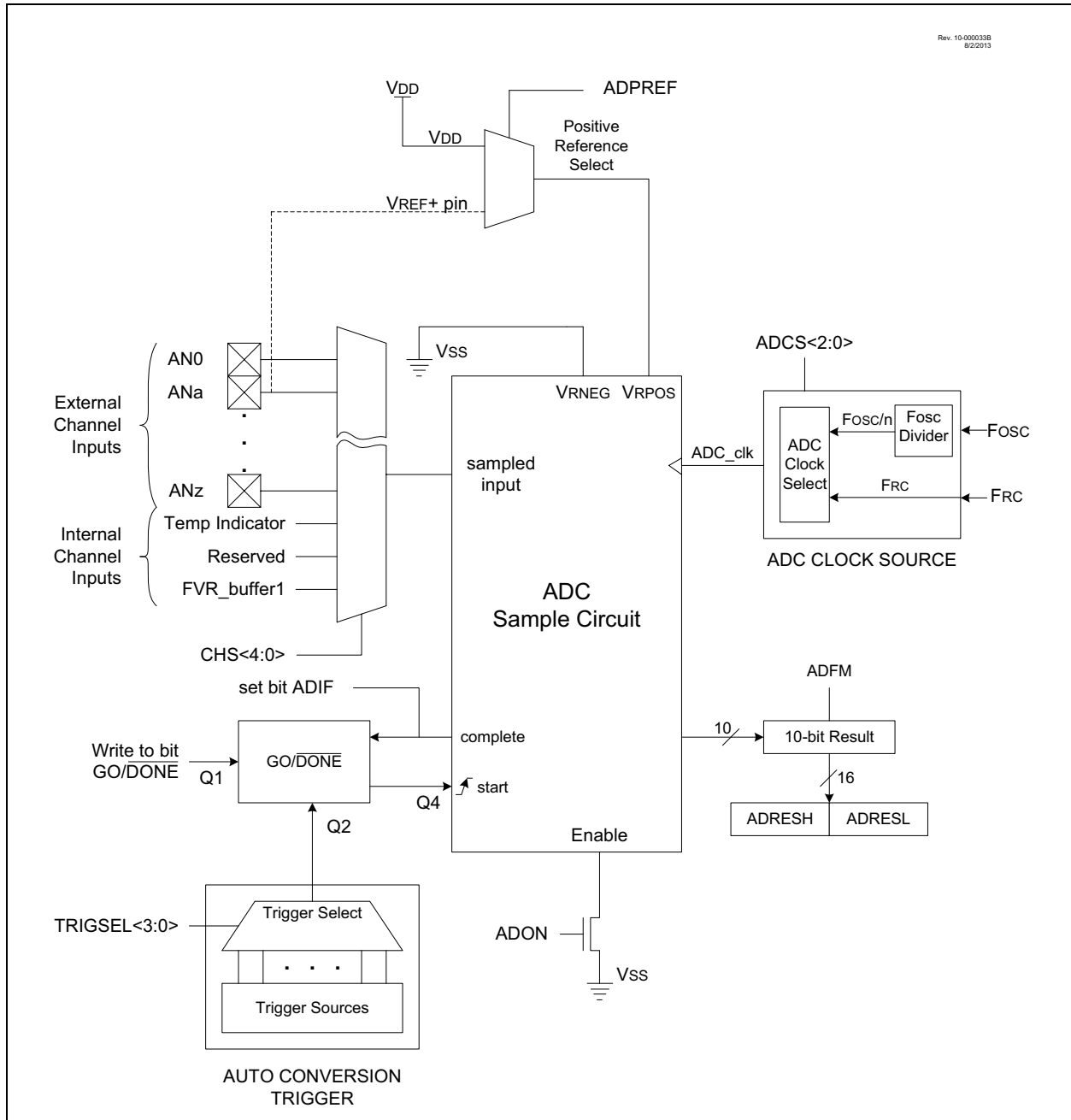
The Analog-to-Digital Converter (ADC) allows conversion of an analog input signal to a 10-bit binary representation of that signal. This device uses analog inputs, which are multiplexed into a single sample and hold circuit. The output of the sample and hold is connected to the input of the converter. 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**



# PIC16(L)F1507

## 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 14 channel selections available:

- AN<11:0> pins
- Temperature Indicator
- FVR\_buffer1

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

When changing channels, a delay (TACQ) is required before starting the next conversion. Refer to [Section 15.2.6 “ADC Conversion Procedure”](#) for more information.

### 15.1.3 ADC VOLTAGE REFERENCE

The ADC module uses a positive and a negative voltage reference. The positive reference is labeled ref+ and the negative reference is labeled ref-.

The positive voltage reference (ref+) is selected by the ADPREF bits in the ADCON1 register. The positive voltage reference source can be:

- VREF+ pin
- VDD

The negative voltage reference (ref-) source is:

- VSS

### 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 (internal RC 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 ADC conversion requirements in [Section 25.0 “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.

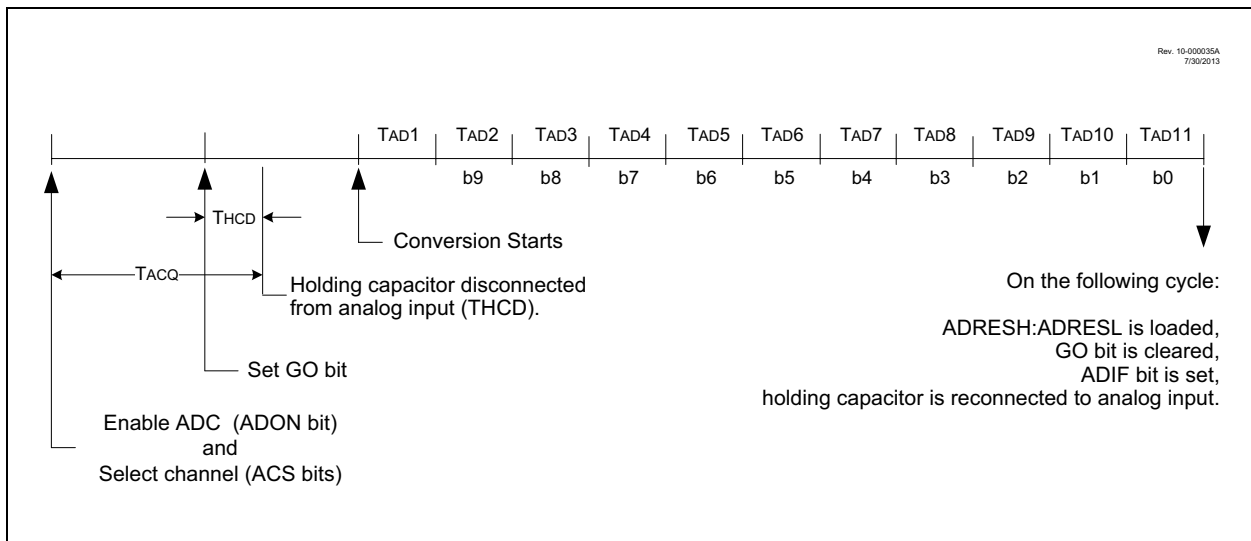
**TABLE 15-1: ADC CLOCK PERIOD (TAD) Vs. DEVICE OPERATING FREQUENCIES**

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	125 ns	250 ns	500 ns	2.0 μs
Fosc/4	100	200 ns	250 ns	500 ns	1.0 μs	4.0 μs
Fosc/8	001	400 ns	500 ns	1.0 μs	2.0 μs	8.0 μs
Fosc/16	101	800 ns	1.0 μs	2.0 μs	4.0 μs	16.0 μs
Fosc/32	010	1.6 μs	2.0 μs	4.0 μs	8.0 μs	32.0 μs
Fosc/64	110	3.2 μs	4.0 μs	8.0 μs	16.0 μs	64.0 μs
FRC	x11	1.0-6.0 μs	1.0-6.0 μs	1.0-6.0 μs	1.0-6.0 μs	1.0-6.0 μs

**Legend:** Shaded cells are outside of recommended range.

**Note:** The TAD period when using the FRC clock source can fall within a specified range, (see TAD parameter). The TAD period when using the Fosc-based clock source can be configured for a more precise TAD period. 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**



# PIC16(L)F1507

## 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.

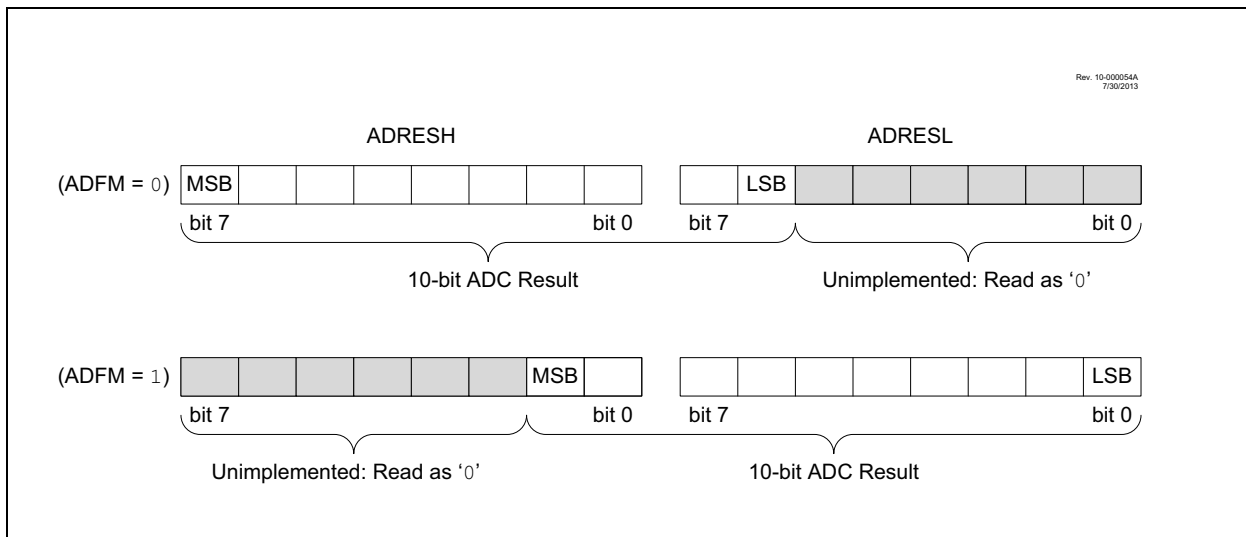
This interrupt can be generated while the device is operating or while in Sleep. If the device is in Sleep, the interrupt will wake-up the device. Upon waking from Sleep, the next instruction following the `SLEEP` instruction is always executed. If the user is attempting to wake-up from Sleep and resume 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 ADC 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 ADC CONVERSION RESULT FORMAT**





## 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/DONE bit of the `ADCON0` register to a '1' will start the Analog-to-Digital conversion.

**Note:** The GO/DONE bit should not be set in the same instruction that turns on the ADC. Refer to [Section 15.2.6 “ADC Conversion Procedure”](#).

### 15.2.2 COMPLETION OF A CONVERSION

When the conversion is complete, the ADC module will:

- Clear the GO/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/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. Performing the ADC conversion during Sleep can reduce system noise. 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 AUTO-CONVERSION TRIGGER

The auto-conversion trigger allows periodic ADC measurements without software intervention. When a rising edge of the selected source occurs, the GO/DONE bit is set by hardware.

The auto-conversion trigger source is selected with the TRIGSEL<3:0> bits of the `ADCON2` register.

Using the auto-conversion trigger does not assure proper ADC timing. It is the user's responsibility to ensure that the ADC timing requirements are met.

See [Table 15-2](#) for auto-conversion sources.

**TABLE 15-2: AUTO-CONVERSION SOURCES**

Source Peripheral	Signal Name
Timer0	T0_overflow
Timer1	T1_overflow
Timer2	T2_match
CLC1	LC1_out
CLC2	LC2_out

# PIC16(L)F1507

## 15.2.6 ADC 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)
  - Disable weak pull-ups either globally (Refer to the OPTION\_REG register) or individually (Refer to the appropriate WPUx 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<sup>(1)</sup>
4. Wait the required acquisition time<sup>(2)</sup>.
5. Start conversion by setting the GO/DONE bit.
6. Wait for ADC conversion to complete by one of the following:
  - Polling the GO/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 15.4 “ADC Acquisition Requirements”](#).

## EXAMPLE 15-1: ADC CONVERSION

```
;This code block configures the ADC
;for polling, Vdd and Vss references, FRC
;oscillator and AN0 input.
;
;Conversion start & polling for completion
;are included.
;
BANKSEL    ADCON1    ;
MOVLW     B'11110000' ;Right justify, FRC
                                ;oscillator
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    WPUA     ;
BCF       WPUA,0    ;Disable weak
                                pull-up on RA0
BANKSEL    ADCON0    ;
MOVLW     B'00000001' ;Select channel AN0
MOVWF     ADCON0    ;Turn ADC On
CALL     SampleTime ;Acquisition 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
```

## 15.3 Register Definitions: ADC Control

**REGISTER 15-1: ADCON0: ADC CONTROL REGISTER 0**

U-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
—	CHS<4:0>					GO/DONE	ADON
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 6-2    **CHS<4:0>:** Analog Channel Select bits
  - 00000 = AN0
  - 00001 = AN1
  - 00010 = AN2
  - 00011 = AN3
  - 00100 = AN4
  - 00101 = AN5
  - 00110 = AN6
  - 00111 = AN7
  - 01000 = AN8
  - 01001 = AN9
  - 01010 = AN10
  - 01011 = AN11
  - 01100 = Reserved. No channel connected.
  - 
  - 
  - 
  - 11100 = Reserved. No channel connected.
  - 11101 = Temperature Indicator<sup>(1)</sup>
  - 11110 = Reserved. No channel connected.
  - 11111 = FVR (Fixed Voltage Reference) Buffer 1 Output<sup>(2)</sup>
- bit 1      **GO/DONE:** ADC Conversion Status bit
  - 1 = ADC conversion cycle in progress. Setting this bit starts an ADC conversion cycle.  
This bit is automatically cleared by hardware when the ADC conversion has completed.
  - 0 = ADC 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 14.0 “Temperature Indicator Module”](#) for more information.  
**Note 2:** See [Section 13.0 “Fixed Voltage Reference \(FVR\)”](#) for more information.

# PIC16(L)F1507

## REGISTER 15-2: ADCON1: ADC 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:** ADC 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>:** ADC Conversion Clock Select bits  
 000 = Fosc/2  
 001 = Fosc/8  
 010 = Fosc/32  
 011 = FRC (clock supplied from an internal RC oscillator)  
 100 = Fosc/4  
 101 = Fosc/16  
 110 = Fosc/64  
 111 = FRC (clock supplied from an internal RC oscillator)
- bit 3-2      **Unimplemented:** Read as '0'
- bit 1-0      **ADPREF<1:0>:** ADC Positive Voltage Reference Configuration bits  
 00 = VRPOS is connected to VDD  
 01 = Reserved  
 10 = VRPOS is connected to external VREF+ pin<sup>(1)</sup>  
 11 = Reserved

**Note 1:** When selecting the VREF+ pin as the source of the positive reference, be aware that a minimum voltage specification exists. See [Section 25.0 "Electrical Specifications"](#) for details.

## REGISTER 15-3: ADCON2: ADC CONTROL REGISTER 2

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	U-0	U-0	U-0
TRIGSEL<3:0> <sup>(1)</sup>				—	—	—	—
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      **TRIGSEL<3:0>**: Auto-Conversion Trigger Selection bits<sup>(1)</sup>

0000	=	No auto-conversion trigger selected
0001	=	Reserved
0010	=	Reserved
0011	=	Timer0 – T0_overflow <sup>(2)</sup>
0100	=	Timer1 – T1_overflow <sup>(2)</sup>
0101	=	Timer2 – T2_match
0110	=	Reserved
0111	=	Reserved
1000	=	CLC1 – LC1_out
1001	=	CLC2 – LC2_out
1010	=	Reserved
1011	=	Reserved
1100	=	Reserved
1101	=	Reserved
1110	=	Reserved
1111	=	Reserved

bit 3-0      **Unimplemented**: Read as '0'

**Note 1:** This is a rising edge sensitive input for all sources.

**2:** Signal also sets its corresponding interrupt flag.

# PIC16(L)F1507

## REGISTER 15-4: 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 15-5: 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 15-6: 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 15-7: 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

# PIC16(L)F1507

## 15.4 ADC 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 ( $R_S$ ) and the internal sampling switch ( $R_{SS}$ ) impedance directly affect the time required to charge the capacitor CHOLD. The sampling switch ( $R_{SS}$ ) impedance varies over the device voltage ( $V_{DD}$ ), refer to Figure 15-4. **The maximum recommended impedance for analog sources is 10 k $\Omega$ .** As the

source impedance is decreased, the acquisition time may be decreased. After the analog input channel is selected (or changed), an ADC 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 VDD*

$$\begin{aligned}T_{ACQ} &= \text{Amplifier Settling Time} + \text{Hold Capacitor Charging Time} + \text{Temperature Coefficient} \\ &= T_{AMP} + T_C + T_{COFF} \\ &= 2\mu s + T_C + [(Temperature - 25^\circ C)(0.05\mu s/^\circ C)]\end{aligned}$$

*The value for  $T_C$  can be approximated with the following equations:*

$$V_{APPLIED} \left( 1 - \frac{1}{(2^{n+1}) - 1} \right) = V_{CHOLD} \quad ;[1] V_{CHOLD} \text{ charged to within } 1/2 \text{ lsb}$$

$$V_{APPLIED} \left( 1 - e^{\frac{-T_C}{RC}} \right) = V_{CHOLD} \quad ;[2] V_{CHOLD} \text{ charge response to } V_{APPLIED}$$

$$V_{APPLIED} \left( 1 - e^{\frac{-T_C}{RC}} \right) = V_{APPLIED} \left( 1 - \frac{1}{(2^{n+1}) - 1} \right) \quad ;\text{combining [1] and [2]}$$

*Note: Where  $n$  = number of bits of the ADC.*

*Solving for  $T_C$ :*

$$\begin{aligned}T_C &= -CHOLD(R_{IC} + R_{SS} + R_S) \ln(1/2047) \\ &= -12.5pF(1k\Omega + 7k\Omega + 10k\Omega) \ln(0.0004885) \\ &= 1.72\mu s\end{aligned}$$

*Therefore:*

$$\begin{aligned}T_{ACQ} &= 2\mu s + 1.72\mu s + [(50^\circ C - 25^\circ C)(0.05\mu s/^\circ C)] \\ &= 4.97\mu s\end{aligned}$$

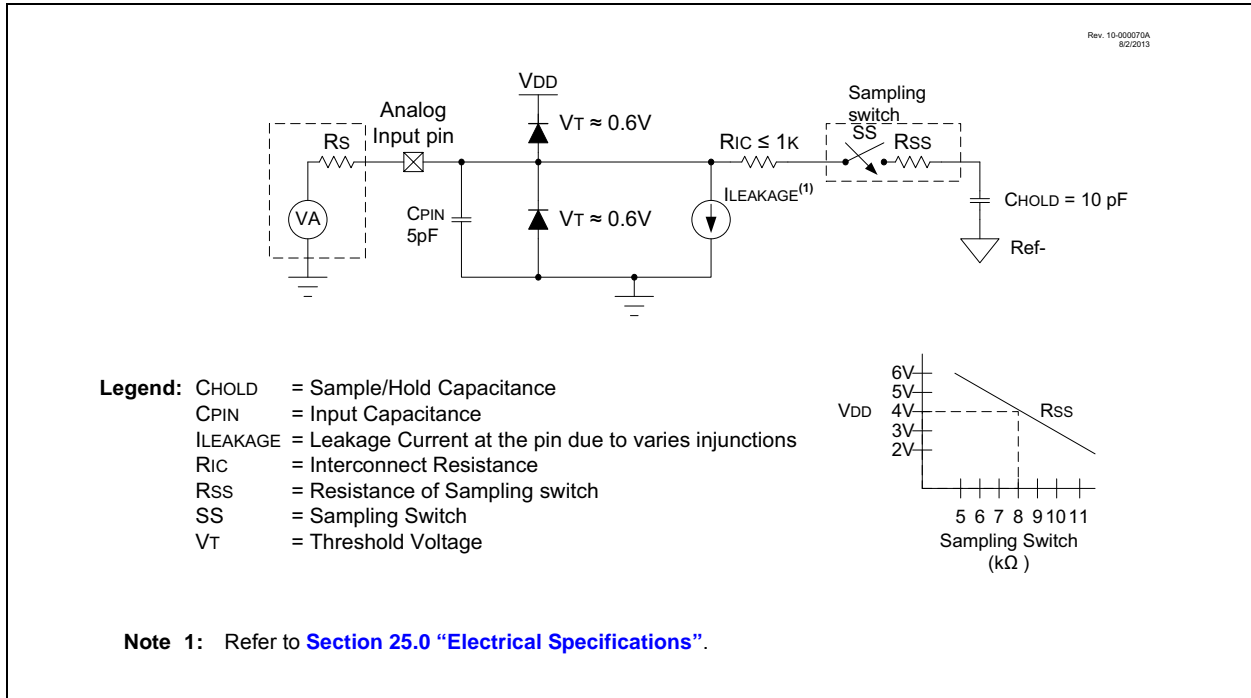
**Note 1:** The reference voltage ( $V_{RPOS}$ ) has no effect on the equation, since it cancels itself out.

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

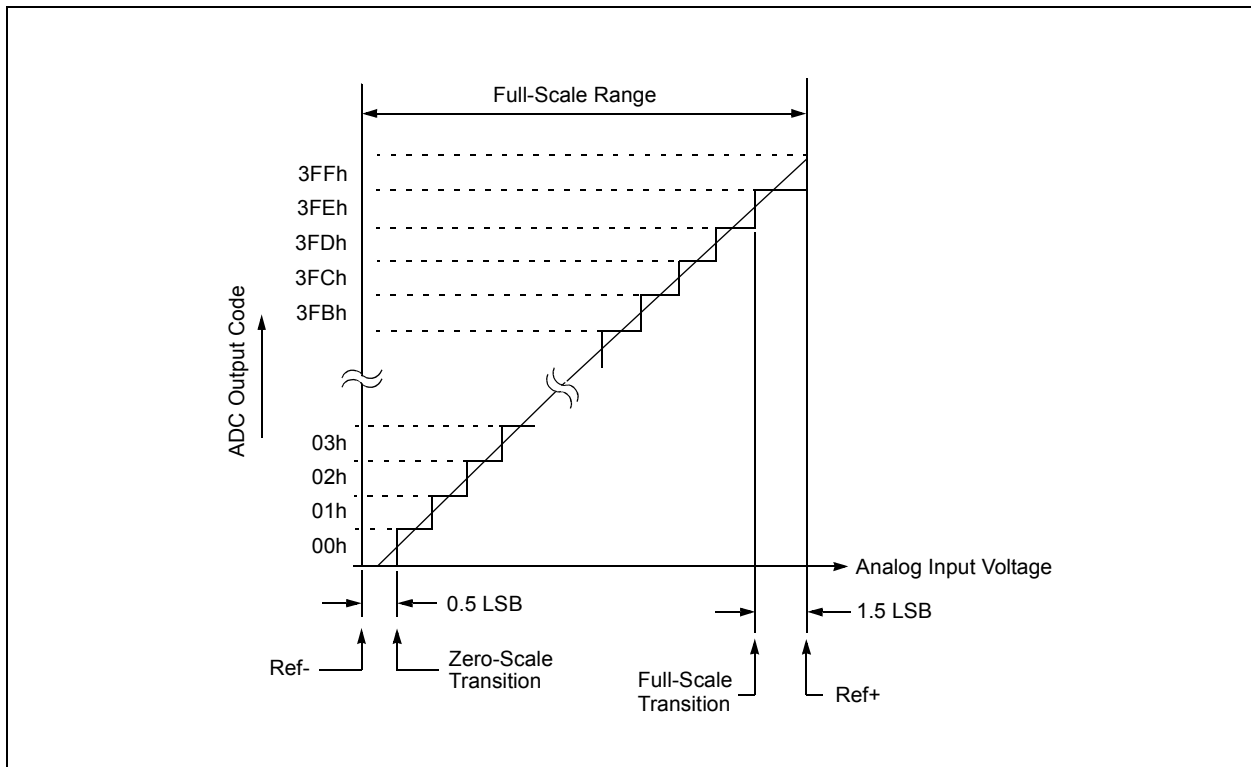
**3:** The maximum recommended impedance for analog sources is 10 k $\Omega$ . This is required to meet the pin leakage specification.



**FIGURE 15-4: ANALOG INPUT MODEL**



**FIGURE 15-5: ADC TRANSFER FUNCTION**



# PIC16(L)F1507

**TABLE 15-3: 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	—	CHS<4:0>					GO/DONE	ADON	123
ADCON1	ADFM	ADCS<2:0>			—	—	ADPREF<1:0>		124
ADCON2	TRIGSEL<3:0>				—	—	—	—	125
ADRESH	ADC Result Register High								126, 127
ADRESL	ADC Result Register Low								126, 127
ANSELA	—	—	—	ANSA4	—	ANSA2	ANSA1	ANSA0	99
ANSELB	—	—	ANSB5	ANSB4	—	—	—	—	103
ANSELC	ANSC7	ANSC6	—	—	ANSC3	ANSC2	ANSC1	ANSC0	107
INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	64
PIE1	TMR1GIE	ADIE	—	—	—	—	TMR2IE	TMR1IE	65
PIR1	TMR1GIF	ADIF	—	—	—	—	TMR2IF	TMR1IF	68
TRISA	—	—	TRISA5	TRISA4	— <sup>(1)</sup>	TRISA2	TRISA1	TRISA0	98
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	—	—	—	—	102
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	106
FVRCON	FVREN	FVRRDY	TSEN	TSRNG	—	—	ADFVR<1:0>		114

**Legend:** x = unknown, u = unchanged, — = unimplemented read as '0', α = value depends on condition. Shaded cells are not used for ADC module.

**Note 1:** Unimplemented, read as '1'.

## 16.0 TIMER0 MODULE

The Timer0 module is an 8-bit timer/counter with the following features:

- 8-bit timer/counter register (TMR0)
- 3-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.

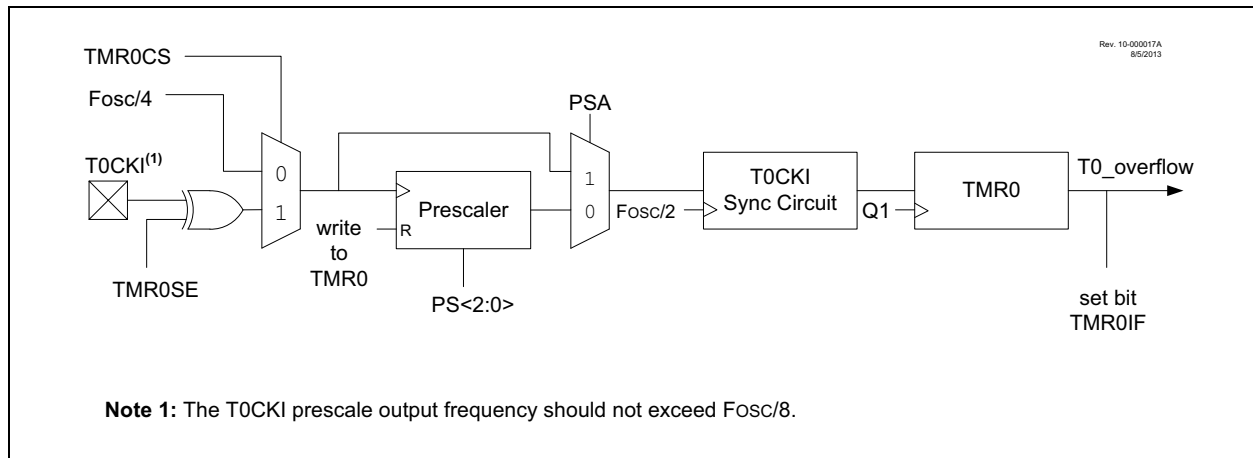
#### 16.1.2 8-BIT COUNTER MODE

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: TIMER0 BLOCK DIAGRAM**



# PIC16(L)F1507

---

## 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 25.0 “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 Register Definitions: Option 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
ADCON2	TRIGSEL<3:0>				—	—	—	—	125
INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	64
OPTION_REG	$\overline{\text{WPUEN}}$	INTEDG	TMR0CS	TMR0SE	PSA	PS<2:0>			133
TMR0	Holding Register for the 8-bit Timer0 Count								131*
TRISA	—	—	TRISA5	TRISA4	— <sup>(1)</sup>	TRISA2	TRISA1	TRISA0	98

**Legend:** — = Unimplemented location, read as '0'. Shaded cells are not used by the Timer0 module.

\* Page provides register information.

**Note 1:** Unimplemented, read as '1'.

# PIC16(L)F1507

## 17.0 TIMER1 MODULE WITH GATE CONTROL

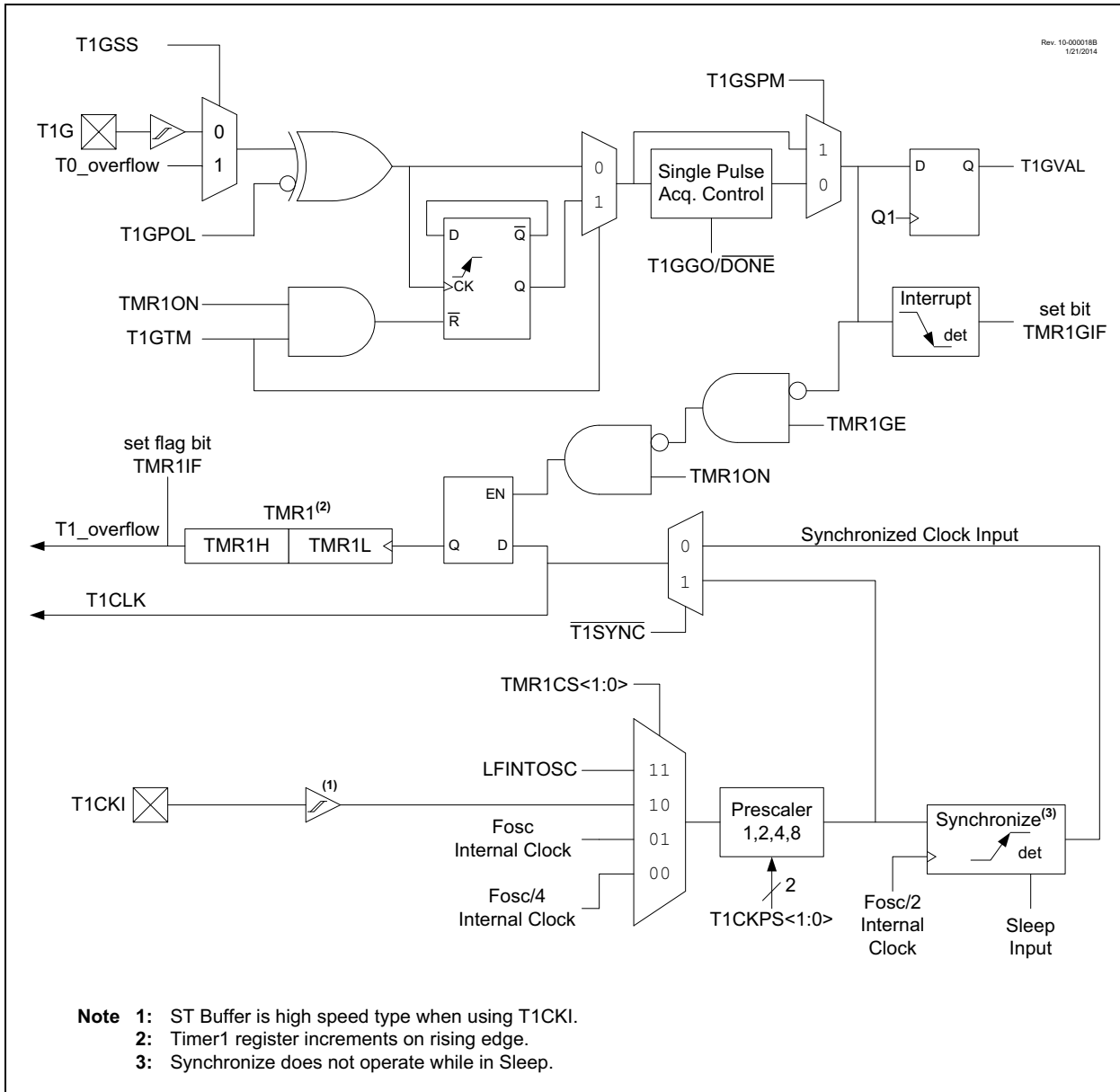
The Timer1 module is a 16-bit timer/counter with the following features:

- 16-bit timer/counter register pair (TMR1H:TMR1L)
- Programmable internal or external clock source
- 2-bit prescaler
- Optionally synchronized comparator out
- Multiple Timer1 gate (count enable) sources

- Interrupt on overflow
- Wake-up on overflow (external clock, Asynchronous mode only)
- ADC Auto-Conversion Trigger(s)
- 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**



## 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> 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 sources may be used:

- Asynchronous event on the T1G pin to Timer1 gate
- C1 or C2 comparator input 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. The external clock source can be synchronized to the microcontroller system clock or it can run asynchronously.

**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**

TMR1CS<1:0>	Clock Source
11	LFINTOSC
10	External Clocking on T1CKI Pin
01	System Clock (FOSC)
00	Instruction Clock (FOSC/4)

# PIC16(L)F1507

## 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 Operation in Asynchronous Counter Mode

If control bit  $\overline{T1SYNC}$  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 17.4.1 “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.

### 17.4.1 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.

## 17.5 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.

### 17.5.1 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

### 17.5.2 TIMER1 GATE SOURCE SELECTION

Timer1 gate source selections are shown in [Table 17-4](#). Source selection is controlled by the T1GSS<1:0> 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
0	Timer1 Gate pin (T1G)
1	Overflow of Timer0 (T0_overflow) (TMR0 increments from FFh to 00h)

**Note 1:** Optionally synchronized comparator output.



## 17.5.2.1 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.

## 17.5.2.2 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.

## 17.5.3 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.

<b>Note:</b> Enabling Toggle mode at the same time as changing the gate polarity may result in indeterminate operation.
---

## 17.5.4 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/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/DONE bit will automatically be cleared. No other gate events will be allowed to increment Timer1 until the T1GGO/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/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.

## 17.5.5 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).

## 17.5.6 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).

# PIC16(L)F1507

## 17.6 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.

## 17.7 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
- $\overline{T1SYNC}$  bit of the T1CON register must be set
- TMR1CS bits of the T1CON register must be configured

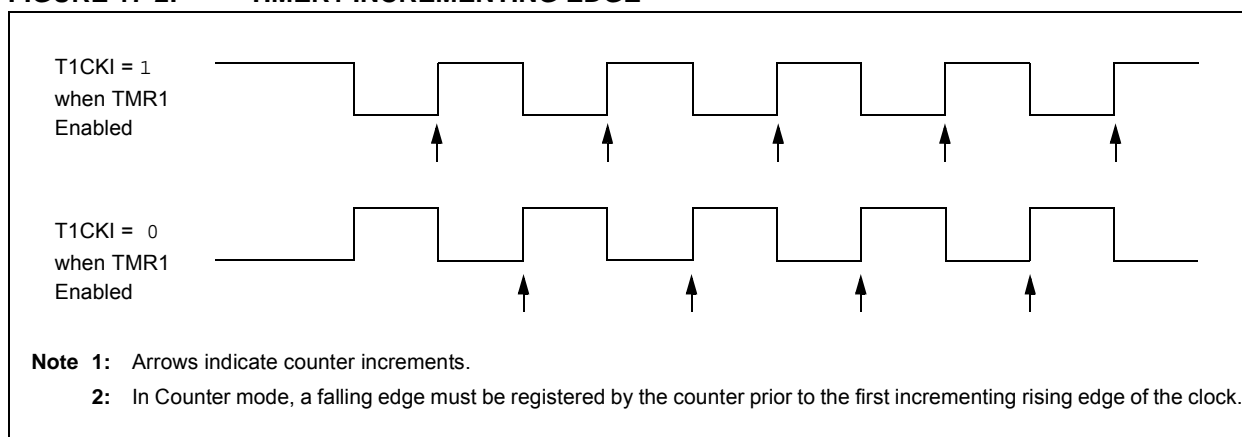
The device will wake-up on an overflow and execute the next instructions. If the GIE bit of the INTCON register is set, the device will call the Interrupt Service Routine.

Timer1 oscillator will continue to operate in Sleep regardless of the  $\overline{T1SYNC}$  bit setting.

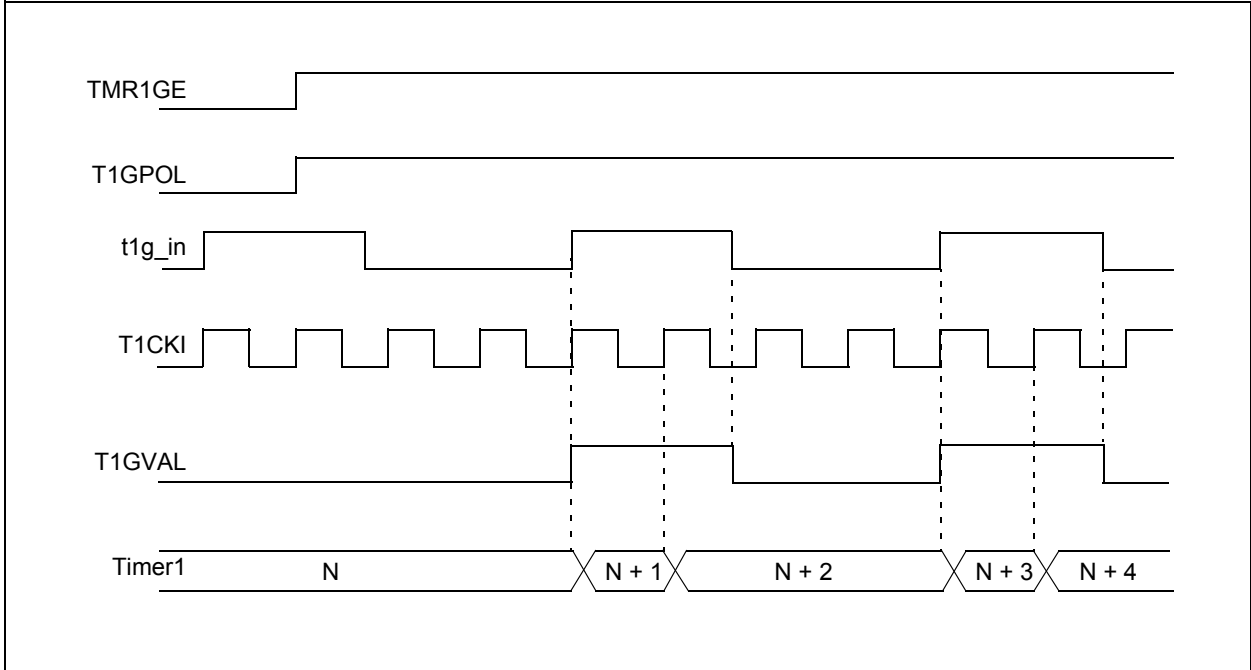
### 17.7.1 ALTERNATE PIN LOCATIONS

This module incorporates I/O pins that can be moved to other locations with the use of the alternate pin function register, APFCON. To determine which pins can be moved and what their default locations are upon a Reset, see [Section 11.1 “Alternate Pin Function”](#) for more information.

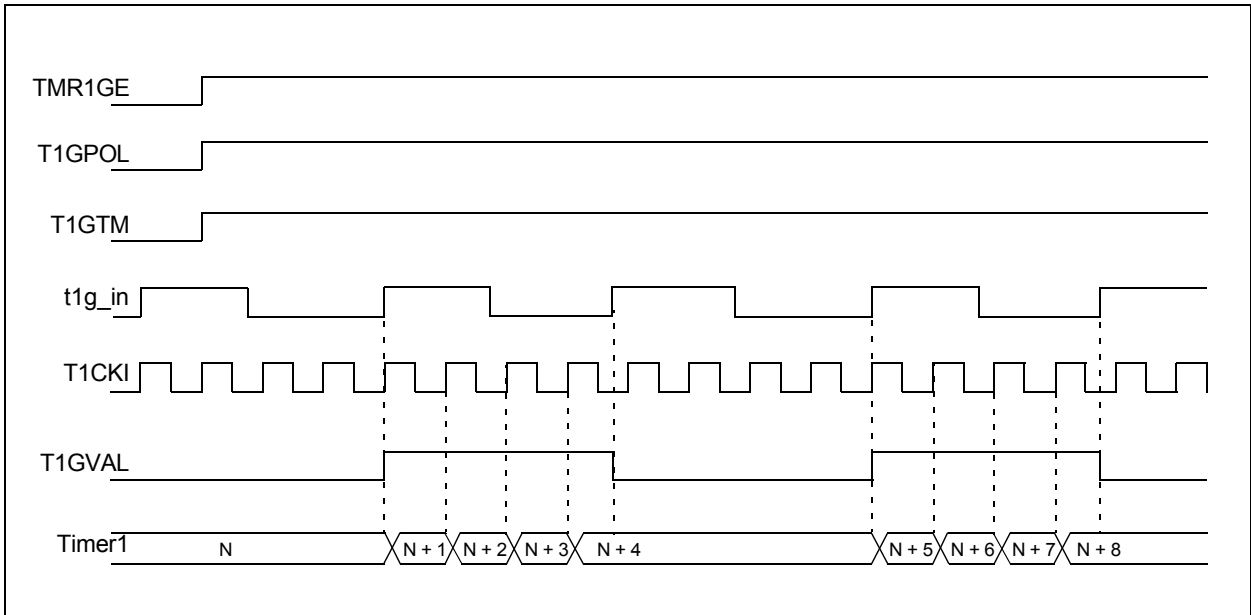
**FIGURE 17-2: TIMER1 INCREMENTING EDGE**



**FIGURE 17-3: TIMER1 GATE ENABLE MODE**

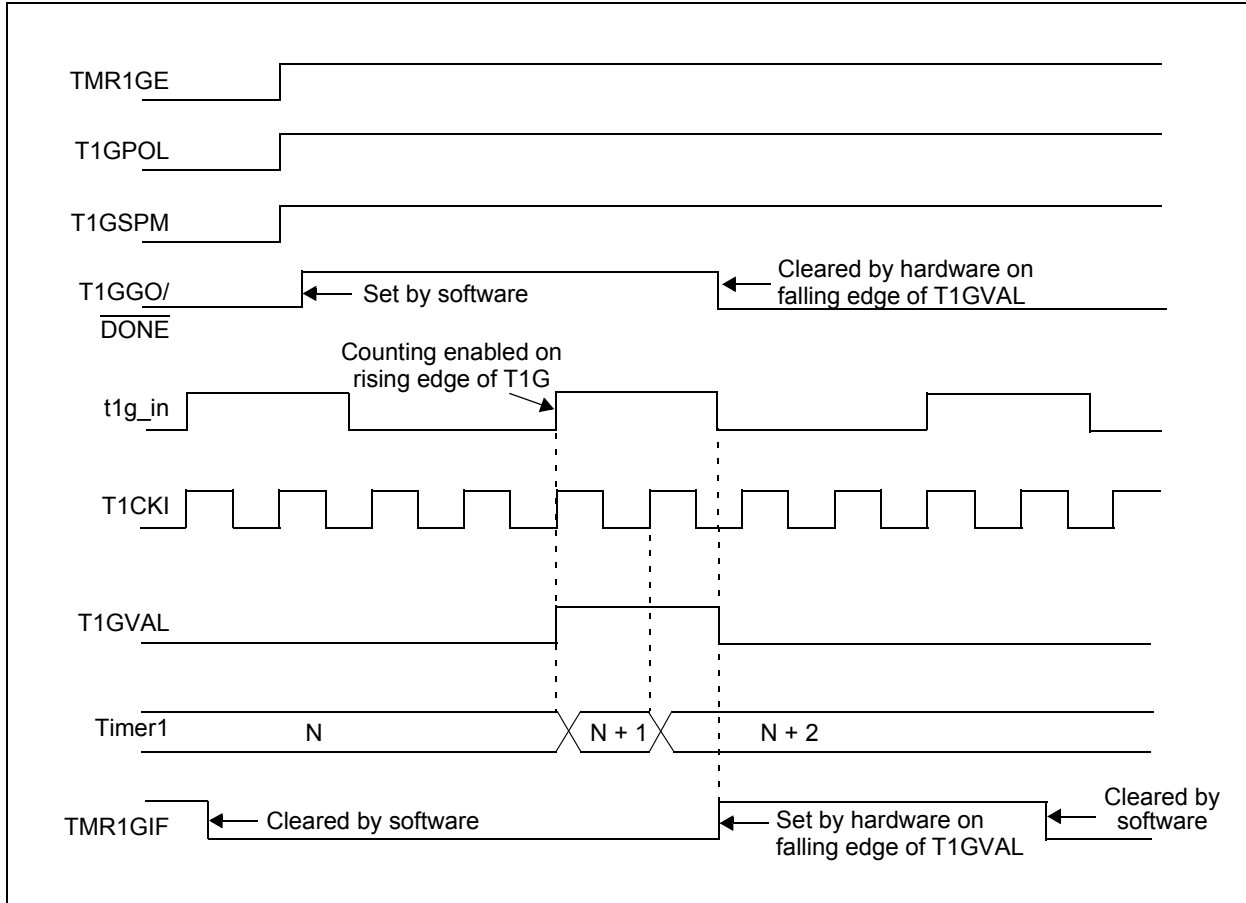


**FIGURE 17-4: TIMER1 GATE TOGGLE MODE**

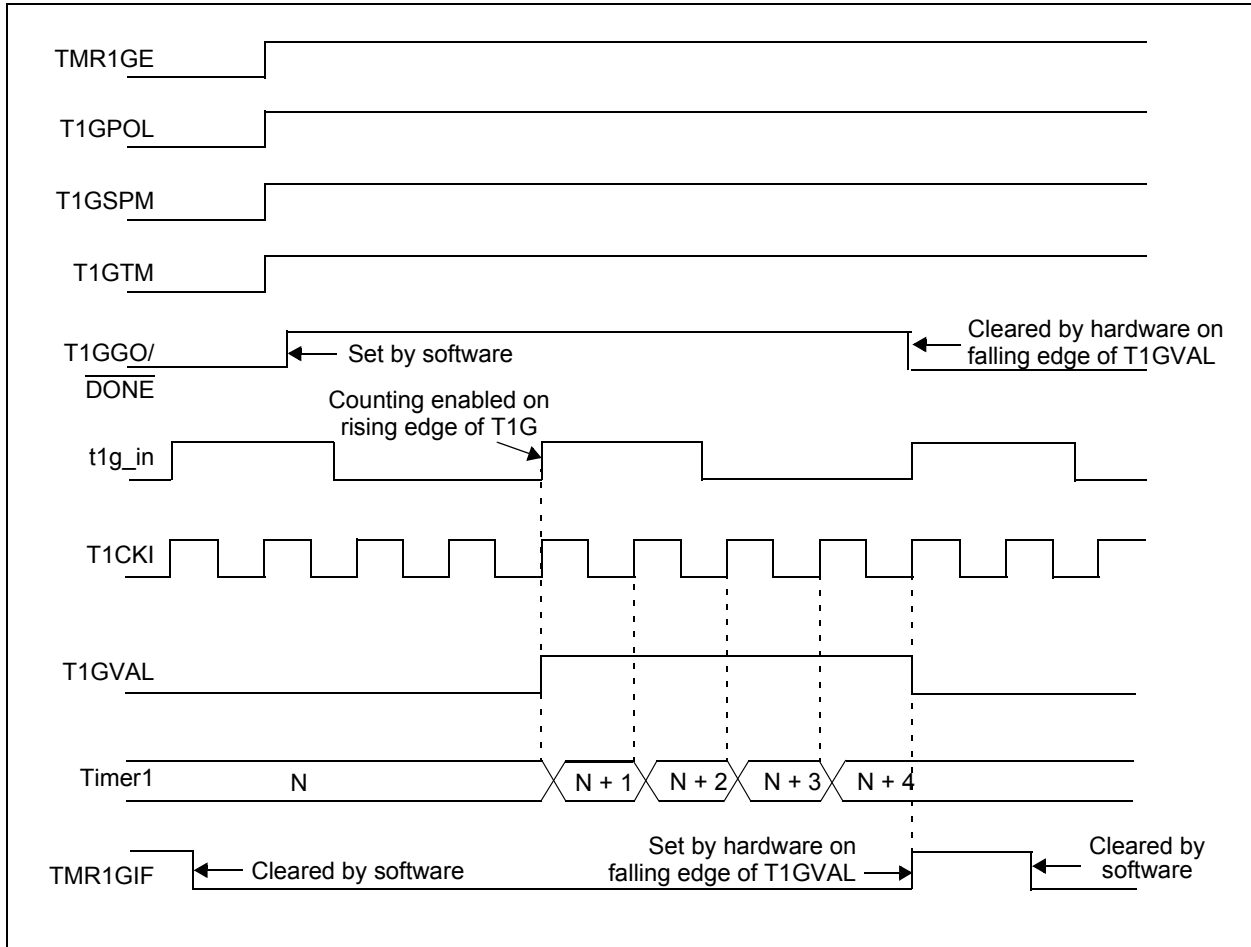


# PIC16(L)F1507

**FIGURE 17-5: TIMER1 GATE SINGLE-PULSE MODE**



**FIGURE 17-6: TIMER1 GATE SINGLE-PULSE AND TOGGLE COMBINED MODE**



# PIC16(L)F1507

## 17.8 Register Definitions: Timer1 Control

### REGISTER 17-1: T1CON: TIMER1 CONTROL REGISTER

R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	U-0	R/W-0/u	U-0	R/W-0/u
TMR1CS<1:0>	T1CKPS<1:0>	—	$\overline{T1SYNC}$	—	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 = Timer1 clock source is LFINTOSC

10 = Timer1 clock source is T1CKI pin (on the rising edge)

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 **Unimplemented**: Read as '0'

bit 2  **$\overline{T1SYNC}$** : Timer1 Synchronization Control bit

1 = Do not synchronize asynchronous clock input

0 = Synchronize asynchronous clock input with system clock (Fosc)

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

bit 0 **TMR1ON**: Timer1 On bit

1 = Enables Timer1

0 = Stops Timer1 and clears Timer1 gate flip-flop

## 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	U-0	R/W-0/u
TMR1GE	T1GPOL	T1GTM	T1GSPM	T1GGO/ DONE	T1GVAL	—	T1GSS
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/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 Value Status bit  
Indicates the current state of the Timer1 gate that could be provided to TMR1H:TMR1L.  
Unaffected by Timer1 Gate Enable (TMR1GE).
- bit 1      **Unimplemented:** Read as '0'
- bit 0      **T1GSS:** Timer1 Gate Source Select bit  
1 = Timer0 overflow output (T0\_output)  
0 = Timer1 gate pin (T1G)

# PIC16(L)F1507

**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
ANSELA	—	—	—	ANSA4	—	ANSA2	ANSA1	ANSA0	99
INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	64
PIE1	TMR1GIE	ADIE	—	—	—	—	TMR2IE	TMR1IE	65
PIR1	TMR1GIF	ADIF	—	—	—	—	TMR2IF	TMR1IF	68
TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Count								138*
TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Count								138*
TRISA	—	—	TRISA5	TRISA4	— <sup>(1)</sup>	TRISA2	TRISA1	TRISA0	98
T1CON	TMR1CS<1:0>		T1CKPS<1:0>		—	T1SYNC	—	TMR1ON	142
T1GCON	TMR1GE	T1GPOL	T1GTM	T1GSPM	T1GGO/ DONE	T1GVAL	—	T1GSS0	143

**Legend:** — = unimplemented location, read as '0'. Shaded cells are not used by the Timer1 module.

\* Page provides register information.

**Note 1:** Unimplemented, read as '1'.



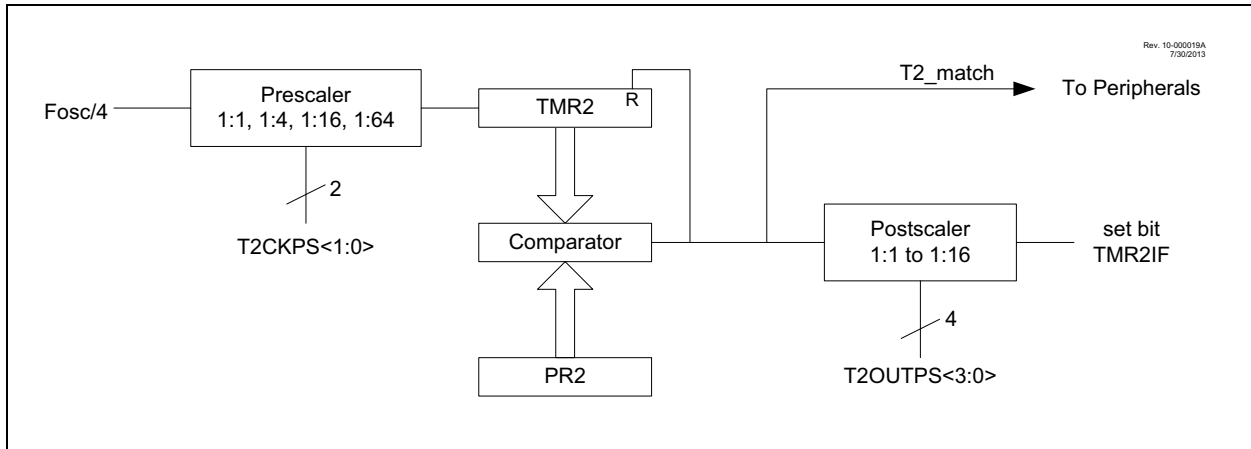
## 18.0 TIMER2 MODULE

The Timer2 module incorporates the following features:

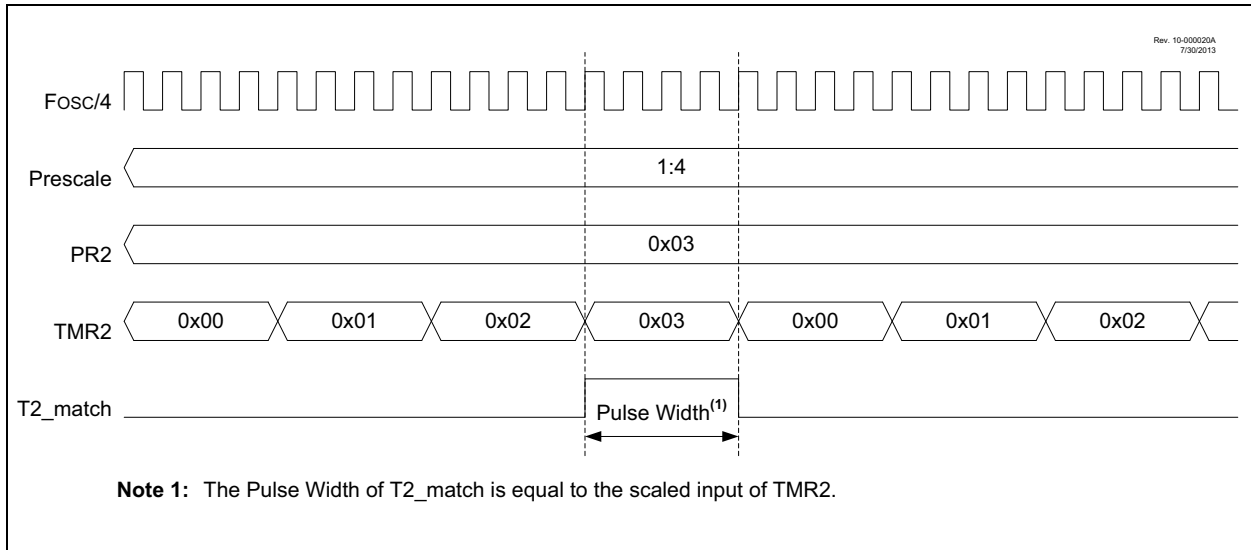
- 8-bit Timer and Period registers (TMR2 and PR2, respectively)
- Readable and writable (both registers)
- Software programmable prescaler (1:1, 1:4, 1:16, and 1:64)
- Software programmable postscaler (1:1 to 1:16)
- Interrupt on TMR2 match with PR2

See [Figure 18-1](#) for a block diagram of Timer2.

**FIGURE 18-1: TIMER2 BLOCK DIAGRAM**



**FIGURE 18-2: TIMER2 TIMING DIAGRAM**



# PIC16(L)F1507

## 18.1 Timer2 Operation

The clock input to the Timer2 module is the system instruction clock ( $F_{osc}/4$ ).

TMR2 increments from 00h on each clock edge.

A 4-bit counter/prescaler on the clock input allows direct input, divide-by-4 and divide-by-16 prescale options. These options are selected by the prescaler control bits,  $T2CKPS<1:0>$  of the T2CON register. The value of TMR2 is compared to that of the Period register, PR2, on each clock cycle. When the two values match, the comparator generates a match signal as the timer output. This signal also resets the value of TMR2 to 00h on the next cycle and drives the output counter/postscaler (see Section 18.2 “Timer2 Interrupt”).

The TMR2 and PR2 registers are both directly readable and writable. The TMR2 register is cleared on any device Reset, whereas the PR2 register initializes to FFh. Both the prescaler and postscaler counters are cleared on the following events:

- a write to the TMR2 register
- a write to the T2CON register
- Power-on Reset (POR)
- Brown-out Reset (BOR)
- $\overline{MCLR}$  Reset
- Watchdog Timer (WDT) Reset
- Stack Overflow Reset
- Stack Underflow Reset
- RESET Instruction

**Note:** TMR2 is not cleared when T2CON is written.

## 18.2 Timer2 Interrupt

Timer2 can also generate an optional device interrupt. The Timer2 output signal (T2\_match) provides the input for the 4-bit counter/postscaler. This counter generates the TMR2 match interrupt flag which is latched in TMR2IF of the PIR1 register. The interrupt is enabled by setting the TMR2 Match Interrupt Enable bit, TMR2IE of the PIE1 register.

A range of 16 postscale options (from 1:1 through 1:16 inclusive) can be selected with the postscaler control bits,  $T2OUTPS<3:0>$ , of the T2CON register.

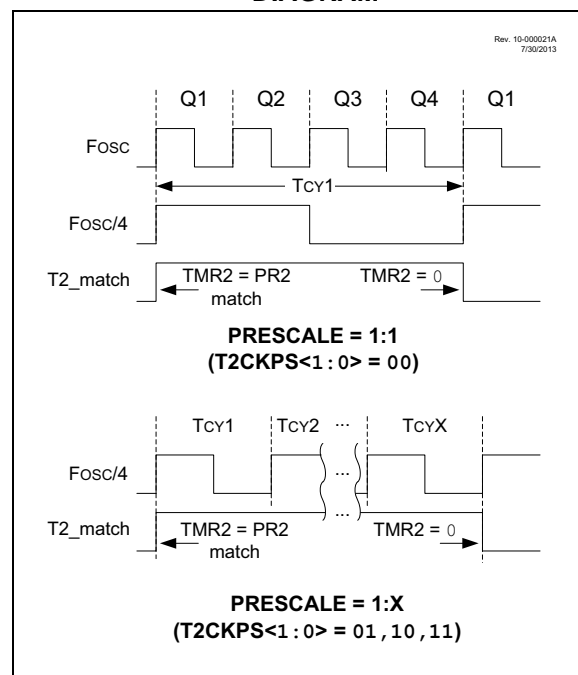
## 18.3 Timer2 Output

The output of TMR2 is T2\_match. T2\_match is available to the following peripherals:

- Configurable Logic Cell (CLC)
- Numerically Controlled Oscillator (NCO)
- Pulse Width Modulator (PWM)

The T2\_match signal is synchronous with the system clock. Figure 18-3 shows two examples of the timing of the T2\_match signal relative to  $F_{osc}$  and prescale value,  $T2CKPS<1:0>$ . The upper diagram illustrates 1:1 prescale timing and the lower diagram, 1:X prescale timing.

**FIGURE 18-3: T2\_MATCH TIMING DIAGRAM**



## 18.4 Timer2 Operation During Sleep

Timer2 cannot be operated while the processor is in Sleep mode. The contents of the TMR2 and PR2 registers will remain unchanged while the processor is in Sleep mode.

## 18.5 Register Definitions: Timer2 Control

**REGISTER 18-1: T2CON: TIMER2 CONTROL REGISTER**

U-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
—	T2OUTPS<3:0>			TMR2ON	T2CKPS<1:0>		
bit 7							bit 0

<b>Legend:</b>		
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       **T2OUTPS<3:0>:** Timer2 Output Postscaler Select bits
  - 0000 = 1:1 Postscaler
  - 0001 = 1:2 Postscaler
  - 0010 = 1:3 Postscaler
  - 0011 = 1:4 Postscaler
  - 0100 = 1:5 Postscaler
  - 0101 = 1:6 Postscaler
  - 0110 = 1:7 Postscaler
  - 0111 = 1:8 Postscaler
  - 1000 = 1:9 Postscaler
  - 1001 = 1:10 Postscaler
  - 1010 = 1:11 Postscaler
  - 1011 = 1:12 Postscaler
  - 1100 = 1:13 Postscaler
  - 1101 = 1:14 Postscaler
  - 1110 = 1:15 Postscaler
  - 1111 = 1:16 Postscaler
- bit 2           **TMR2ON:** Timer2 On bit
  - 1 = Timer2 is on
  - 0 = Timer2 is off
- bit 1-0       **T2CKPS<1:0>:** Timer2 Clock Prescale Select bits
  - 00 = Prescaler is 1
  - 01 = Prescaler is 4
  - 10 = Prescaler is 16
  - 11 = Prescaler is 64

**TABLE 18-1: SUMMARY OF REGISTERS ASSOCIATED WITH TIMER2**

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	IOCFIE	TMR0IF	INTF	IOCFIF	64
PIE1	TMR1GIE	ADIE	—	—	—	—	TMR2IE	TMR1IE	65
PIR1	TMR1GIF	ADIF	—	—	—	—	TMR2IF	TMR1IF	68
PR2	Timer2 Module Period Register								145*
T2CON	—	T2OUTPS<3:0>			TMR2ON	T2CKPS<1:0>			147
TMR2	Holding Register for the 8-bit TMR2 Count								145*

**Legend:** — = unimplemented location, read as '0'. Shaded cells are not used for Timer2 module.  
 \* Page provides register information.

# PIC16(L)F1507

## 19.0 PULSE-WIDTH MODULATION (PWM) MODULE

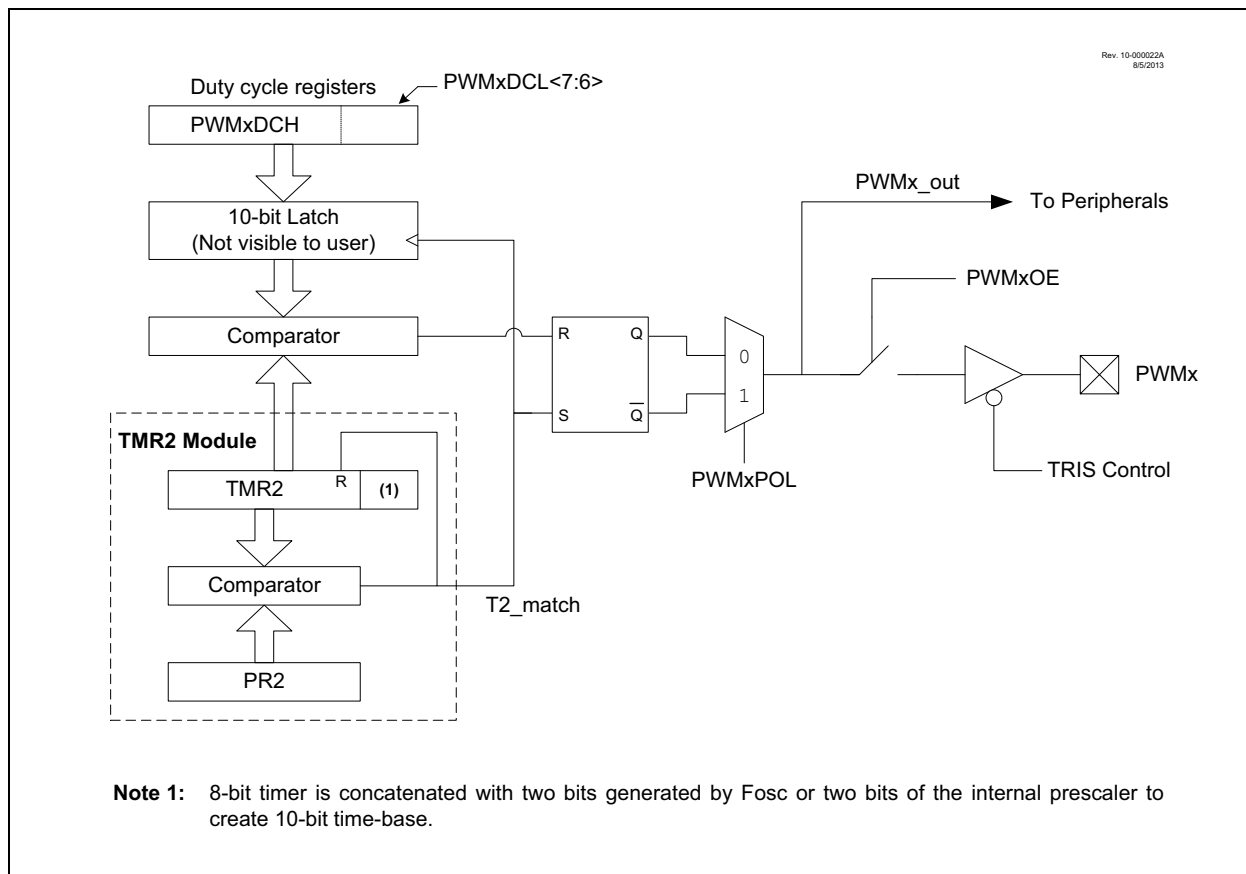
The PWM module generates a Pulse-Width Modulated signal determined by the duty cycle, period, and resolution that are configured by the following registers:

- PR2
- T2CON
- PWMxDCH
- PWMxDCL
- PWMxCON

Figure 19-1 shows a simplified block diagram of PWM operation.

For a step-by-step procedure on how to set up this module for PWM operation, refer to [Section 19.1.9 “Setup for PWM Operation using PWMx Pins”](#).

**FIGURE 19-1: SIMPLIFIED PWM BLOCK DIAGRAM**



## 19.1 PWMx Pin Configuration

All PWM outputs are multiplexed with the PORT data latch. The user must configure the pins as outputs by clearing the associated TRIS bits.

**Note:** Clearing the PWMxOE bit will relinquish control of the PWMx pin.

### 19.1.1 FUNDAMENTAL OPERATION

The PWM module produces a 10-bit resolution output. Timer2 and PR2 set the period of the PWM. The PWMxDCL and PWMxDCH registers configure the duty cycle. The period is common to all PWM modules, whereas the duty cycle is independently controlled.

**Note:** The Timer2 postscaler is not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output.

All PWM outputs associated with Timer2 are set when TMR2 is cleared. Each PWMx is cleared when TMR2 is equal to the value specified in the corresponding PWMxDCH (8 MSb) and PWMxDCL<7:6> (2 LSb) registers. When the value is greater than or equal to PR2, the PWM output is never cleared (100% duty cycle).

**Note:** The PWMxDCH and PWMxDCL registers are double buffered. The buffers are updated when Timer2 matches PR2. Care should be taken to update both registers before the timer match occurs.

### 19.1.2 PWM OUTPUT POLARITY

The output polarity is inverted by setting the PWMxPOL bit of the PWMxCON register.

### 19.1.3 PWM PERIOD

The PWM period is specified by the PR2 register of Timer2. The PWM period can be calculated using the formula of [Equation 19-1](#).

#### EQUATION 19-1: PWM PERIOD

$$PWM\ Period = [(PR2) + 1] \cdot 4 \cdot TOSC \cdot (TMR2\ Prescale\ Value)$$

**Note:**  $TOSC = 1/FOSC$

When TMR2 is equal to PR2, the following three events occur on the next increment cycle:

- TMR2 is cleared
- The PWM output is active. (Exception: When the PWM duty cycle = 0%, the PWM output will remain inactive.)
- The PWMxDCH and PWMxDCL register values are latched into the buffers.

**Note:** The Timer2 postscaler has no effect on the PWM operation.

### 19.1.4 PWM DUTY CYCLE

The PWM duty cycle is specified by writing a 10-bit value to the PWMxDCH and PWMxDCL register pair. The PWMxDCH register contains the eight MSBs and the PWMxDCL<7:6>, the two LSbs. The PWMxDCH and PWMxDCL registers can be written to at any time.

[Equation 19-2](#) is used to calculate the PWM pulse width.

[Equation 19-3](#) is used to calculate the PWM duty cycle ratio.

#### EQUATION 19-2: PULSE WIDTH

$$Pulse\ Width = (PWMxDCH:PWMxDCL<7:6>) \cdot TOSC \cdot (TMR2\ Prescale\ Value)$$

**Note:**  $TOSC = 1/FOSC$

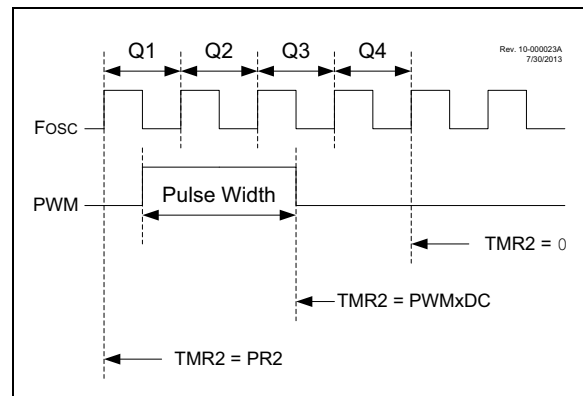
#### EQUATION 19-3: DUTY CYCLE RATIO

$$Duty\ Cycle\ Ratio = \frac{(PWMxDCH:PWMxDCL<7:6>)}{4(PR2 + 1)}$$

The 8-bit timer TMR2 register is concatenated with the two Least Significant bits of  $1/FOSC$ , adjusted by the Timer2 prescaler to create the 10-bit time base. The system clock is used if the Timer2 prescaler is set to 1:1.

[Figure 19-2](#) shows a waveform of the PWM signal when the duty cycle is set for the smallest possible pulse.

**FIGURE 19-2: PWM OUTPUT**



# PIC16(L)F1507

## 19.1.5 PWM RESOLUTION

The resolution determines the number of available duty cycles for a given period. For example, a 10-bit resolution will result in 1024 discrete duty cycles, whereas an 8-bit resolution will result in 256 discrete duty cycles.

The maximum PWM resolution is ten bits when PR2 is 255. The resolution is a function of the PR2 register value as shown by [Equation 19-4](#).

### EQUATION 19-4: PWM RESOLUTION

$$Resolution = \frac{\log[4(PR2 + 1)]}{\log(2)} \text{ bits}$$

**Note:** If the pulse width value is greater than the period the assigned PWM pin(s) will remain unchanged.

**TABLE 19-1: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS (Fosc = 20 MHz)**

PWM Frequency	0.31 kHz	4.88 kHz	19.53 kHz	78.12 kHz	156.3 kHz	208.3 kHz
Timer Prescale	64	4	1	1	1	1
PR2 Value	0xFF	0xFF	0xFF	0x3F	0x1F	0x17
Maximum Resolution (bits)	10	10	10	8	7	6.6

**TABLE 19-2: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS (Fosc = 8 MHz)**

PWM Frequency	0.31 kHz	4.90 kHz	19.61 kHz	76.92 kHz	153.85 kHz	200.0 kHz
Timer Prescale	64	4	1	1	1	1
PR2 Value	0x65	0x65	0x65	0x19	0x0C	0x09
Maximum Resolution (bits)	8	8	8	6	5	5

## 19.1.6 OPERATION IN SLEEP MODE

In Sleep mode, the TMR2 register will not increment and the state of the module will not change. If the PWMx pin is driving a value, it will continue to drive that value. When the device wakes up, TMR2 will continue from its previous state.

## 19.1.7 CHANGES IN SYSTEM CLOCK FREQUENCY

The PWM frequency is derived from the system clock frequency (Fosc). Any changes in the system clock frequency will result in changes to the PWM frequency. Refer to [Section 19.1.1](#) for additional details.

## 19.1.8 EFFECTS OF RESET

Any Reset will force all ports to Input mode and the PWM registers to their Reset states.

## 19.1.9 SETUP FOR PWM OPERATION USING PWMx PINS

The following steps should be taken when configuring the module for PWM operation using the PWMx pins:

1. Disable the PWMx pin output driver(s) by setting the associated TRIS bit(s).
2. Clear the PWMxCON register.
3. Load the PR2 register with the PWM period value.
4. Clear the PWMxDCH register and bits <7:6> of the PWMxDCL register.
5. Configure and start Timer2:
  - Clear the TMR2IF interrupt flag bit of the PIR1 register. See note below.
  - Configure the T2CKPS bits of the T2CON register with the Timer2 prescale value.
  - Enable Timer2 by setting the TMR2ON bit of the T2CON register.
6. Enable PWM output pin and wait until Timer2 overflows, TMR2IF bit of the PIR1 register is set. See note below.
7. Enable the PWMx pin output driver(s) by clearing the associated TRIS bit(s) and setting the PWMxOE bit of the PWMxCON register.
8. Configure the PWM module by loading the PWMxCON register with the appropriate values.

**Note 1:** In order to send a complete duty cycle and period on the first PWM output, the above steps must be followed in the order given. If it is not critical to start with a complete PWM signal, then move Step 8 to replace Step 4.

**2:** For operation with other peripherals only, disable PWMx pin outputs.

# PIC16(L)F1507

## 19.2 Register Definitions: PWM Control

### REGISTER 19-1: PWMxCON: PWM CONTROL REGISTER

R/W-0/0	R/W-0/0	R-0/0	R/W-0/0	U-0	U-0	U-0	U-0
PWMxEN	PWMxOE	PWMxOUT	PWMxPOL	—	—	—	—
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      **PWMxEN:** PWM Module Enable bit  
1 = PWM module is enabled  
0 = PWM module is disabled
- bit 6      **PWMxOE:** PWM Module Output Enable bit  
1 = Output to PWMx pin is enabled  
0 = Output to PWMx pin is disabled
- bit 5      **PWMxOUT:** PWM Module Output Value bit
- bit 4      **PWMxPOL:** PWMx Output Polarity Select bit  
1 = PWM output is active-low  
0 = PWM output is active-high
- bit 3-0    **Unimplemented:** Read as '0'



## REGISTER 19-2: PWMxDCH: PWM DUTY CYCLE HIGH BITS

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
PWMxDCH<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 **PWMxDCH<7:0>**: PWM Duty Cycle Most Significant bits  
These bits are the MSBs of the PWM duty cycle. The two LSBs are found in the PWMxDCL register.

## REGISTER 19-3: PWMxDCL: PWM DUTY CYCLE LOW BITS

R/W-x/u	R/W-x/u	U-0	U-0	U-0	U-0	U-0	U-0
PWMxDCL<7:6>		—	—	—	—	—	—
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 **PWMxDCL<7:6>**: PWM Duty Cycle Least Significant bits  
These bits are the LSBs of the PWM duty cycle. The MSBs are found in the PWMxDCH register.

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

## TABLE 19-3: SUMMARY OF REGISTERS ASSOCIATED WITH PWM

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
PR2	Timer2 module Period Register								145*
PWM1CON	PWM1EN	PWM1OE	PWM1OUT	PWM1POL	—	—	—	—	152
PWM1DCH	PWM1DCH<7:0>								153
PWM1DCL	PWM1DCL<7:6>		—	—	—	—	—	—	153
PWM2CON	PWM2EN	PWM2OE	PWM2OUT	PWM2POL	—	—	—	—	152
PWM2DCH	PWM2DCH<7:0>								153
PWM2DCL	PWM2DCL<7:6>		—	—	—	—	—	—	153
PWM3CON	PWM3EN	PWM3OE	PWM3OUT	PWM3POL	—	—	—	—	152
PWM3DCH	PWM3DCH<7:0>								153
PWM3DCL	PWM3DCL<7:6>		—	—	—	—	—	—	153
PWM4CON	PWM4EN	PWM4OE	PWM4OUT	PWM4POL	—	—	—	—	152
PWM4DCH	PWM4DCH<7:0>								153
PWM4DCL	PWM4DCL<7:6>		—	—	—	—	—	—	153
T2CON	—	T2OUTPS<3:0>				TMR2ON	T2CKPS<1:0>		147
TMR2	Timer2 module Register								145*
TRISA	—	—	TRISA5	TRISA4	— <sup>(1)</sup>	TRISA2	TRISA1	TRISA0	98
TRISC	—	—	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	106

**Legend:** — = Unimplemented locations, read as '0', u = unchanged, x = unknown. Shaded cells are not used by the PWM.

\* Page provides register information.

**Note 1:** Unimplemented, read as '1'.

# PIC16(L)F1507

## 20.0 CONFIGURABLE LOGIC CELL (CLC)

The Configurable Logic Cell (CLCx) provides programmable logic that operates outside the speed limitations of software execution. The logic cell takes up to 16 input signals, and through the use of configurable gates, reduces the 16 inputs to four logic lines that drive one of eight selectable single-output logic functions.

Input sources are a combination of the following:

- I/O pins
- Internal clocks
- Peripherals
- Register bits

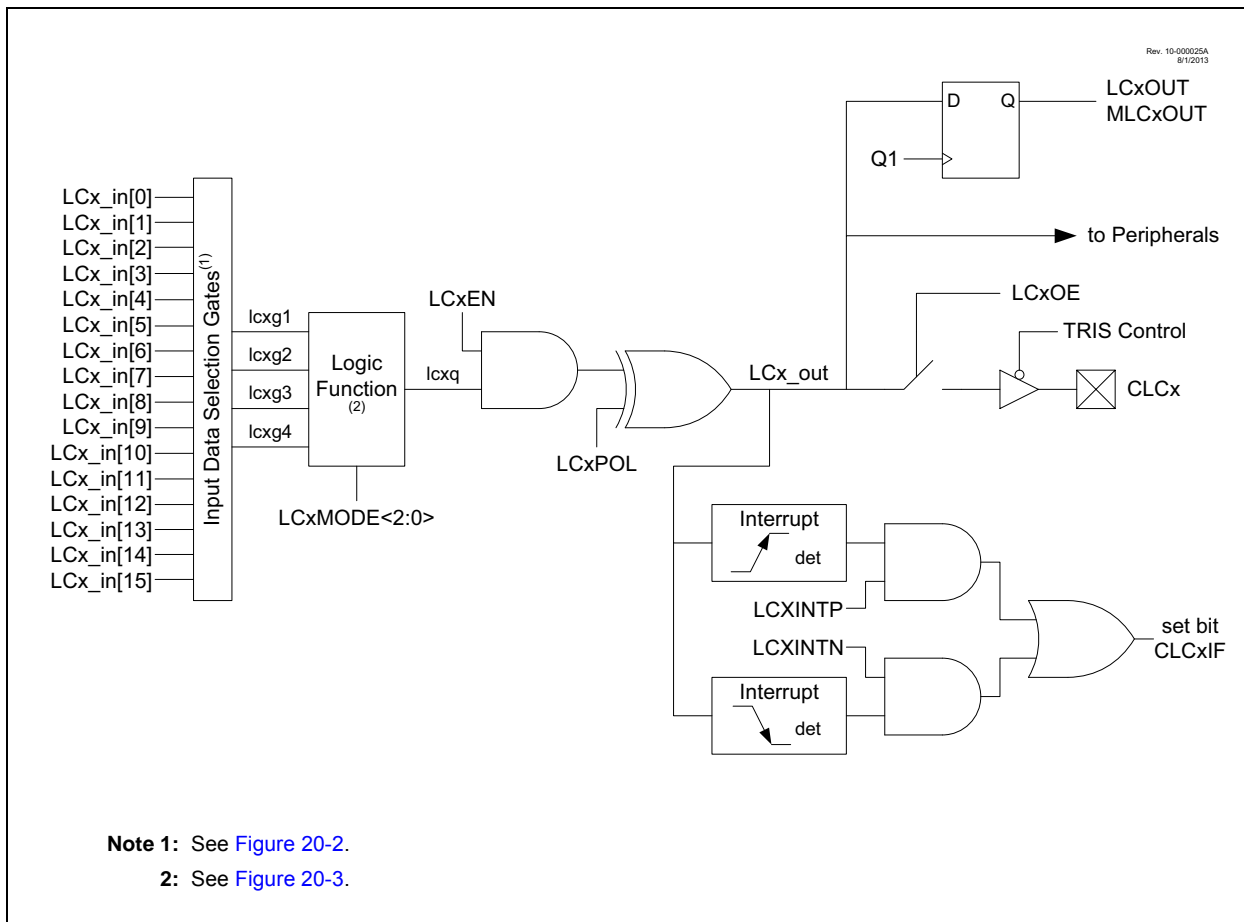
The output can be directed internally to peripherals and to an output pin.

Refer to [Figure 20-1](#) for a simplified diagram showing signal flow through the CLCx.

Possible configurations include:

- Combinatorial Logic
  - AND
  - NAND
  - AND-OR
  - AND-OR-INVERT
  - OR-XOR
  - OR-XNOR
- Latches
  - S-R
  - Clocked D with Set and Reset
  - Transparent D with Set and Reset
  - Clocked J-K with Reset

**FIGURE 20-1: CONFIGURABLE LOGIC CELL BLOCK DIAGRAM**



## 20.1 CLCx Setup

Programming the CLCx module is performed by configuring the four stages in the logic signal flow. The four stages are:

- Data selection
- Data gating
- Logic function selection
- Output polarity

Each stage is setup at run time by writing to the corresponding CLCx Special Function Registers. This has the added advantage of permitting logic reconfiguration on-the-fly during program execution.

### 20.1.1 DATA SELECTION

There are 16 signals available as inputs to the configurable logic. Four 8-input multiplexers are used to select the inputs to pass on to the next stage. The 16 inputs to the multiplexers are arranged in groups of four. Each group is available to two of the four multiplexers, in

each case, paired with a different group. This arrangement makes possible selection of up to two from a group without precluding a selection from another group.

Data selection is through four multiplexers as indicated on the left side of [Figure 20-2](#). Data inputs in the figure are identified by a generic numbered input name.

[Table 20-1](#) correlates the generic input name to the actual signal for each CLC module. The columns labeled lcx1 through lcx4 indicate the MUX output for the selected data input. D1S through D4S are abbreviations for the MUX select input codes: LCxD1S<2:0> through LCxD4S<2:0>, respectively. Selecting a data input in a column excludes all other inputs in that column.

Data inputs are selected with CLCxSEL0 and CLCxSEL1 registers ([Register 20-3](#) and [Register 20-5](#), respectively).

**Note:** Data selections are undefined at power-up.

**TABLE 20-1: CLCx DATA INPUT SELECTION**

Data Input	lcx1 D1S	lcx2 D2S	lcx3 D3S	lcx4 D4S	CLC 1	CLC 2
LCx_in[0]	000	—	—	100	CLC1IN0	CLC2IN0
LCx_in[1]	001	—	—	101	CLC1IN1	CLC2IN1
LCx_in[2]	010	—	—	110	Reserved	Reserved
LCx_in[3]	011	—	—	111	Reserved	Reserved
LCx_in[4]	100	000	—	—	Fosc	Fosc
LCx_in[5]	101	001	—	—	T0_overflow	T0_overflow
LCx_in[6]	110	010	—	—	T1_overflow	T1_overflow
LCx_in[7]	111	011	—	—	T2_match	T2_match
LCx_in[8]	—	100	000	—	LC1_out	LC1_out
LCx_in[9]	—	101	001	—	LC2_out	LC2_out
LCx_in[10]	—	110	010	—	Reserved	Reserved
LCx_in[11]	—	111	011	—	Reserved	Reserved
LCx_in[12]	—	—	100	000	NCO1_out	LFINTOSC
LCx_in[13]	—	—	101	001	HFINTOSC	FRC
LCx_in[14]	—	—	110	010	PWM3_out	PWM1_out
LCx_in[15]	—	—	111	011	PWM4_out	PWM2_out

# PIC16(L)F1507

## 20.1.2 DATA GATING

Outputs from the input multiplexers are directed to the desired logic function input through the data gating stage. Each data gate can direct any combination of the four selected inputs.

**Note:** Data gating is undefined at power-up.

The gate stage is more than just signal direction. The gate can be configured to direct each input signal as inverted or non-inverted data. Directed signals are ANDed together in each gate. The output of each gate can be inverted before going on to the logic function stage.

The gating is in essence a 1-to-4 input AND/NAND/OR/NOR gate. When every input is inverted and the output is inverted, the gate is an OR of all enabled data inputs. When the inputs and output are not inverted, the gate is an AND of all enabled inputs.

Table 20-2 summarizes the basic logic that can be obtained in gate 1 by using the gate logic select bits. The table shows the logic of four input variables, but each gate can be configured to use less than four. If no inputs are selected, the output will be zero or one, depending on the gate output polarity bit.

**TABLE 20-2: DATA GATING LOGIC**

CLCxGLS0	LCxG1POL	Gate Logic
0x55	1	AND
0x55	0	NAND
0xAA	1	NOR
0xAA	0	OR
0x00	0	Logic 0
0x00	1	Logic 1

It is possible (but not recommended) to select both the true and negated values of an input. When this is done, the gate output is zero, regardless of the other inputs, but may emit logic glitches (transient-induced pulses). If the output of the channel must be zero or one, the recommended method is to set all gate bits to zero and use the gate polarity bit to set the desired level.

Data gating is configured with the logic gate select registers as follows:

- Gate 1: CLCxGLS0 ([Register 20-5](#))
- Gate 2: CLCxGLS1 ([Register 20-6](#))
- Gate 3: CLCxGLS2 ([Register 20-7](#))
- Gate 4: CLCxGLS3 ([Register 20-8](#))

Register number suffixes are different than the gate numbers because other variations of this module have multiple gate selections in the same register.

Data gating is indicated in the right side of [Figure 20-2](#). Only one gate is shown in detail. The remaining three gates are configured identically with the exception that the data enables correspond to the enables for that gate.

## 20.1.3 LOGIC FUNCTION

There are eight available logic functions including:

- AND-OR
- OR-XOR
- AND
- S-R Latch
- D Flip-Flop with Set and Reset
- D Flip-Flop with Reset
- J-K Flip-Flop with Reset
- Transparent Latch with Set and Reset

Logic functions are shown in [Figure 20-3](#). Each logic function has four inputs and one output. The four inputs are the four data gate outputs of the previous stage. The output is fed to the inversion stage and from there to other peripherals, an output pin, and back to the CLCx itself.

## 20.1.4 OUTPUT POLARITY

The last stage in the configurable logic cell is the output polarity. Setting the LCxPOL bit of the CLCxCON register inverts the output signal from the logic stage. Changing the polarity while the interrupts are enabled will cause an interrupt for the resulting output transition.

## 20.1.5 CLCx SETUP STEPS

The following steps should be followed when setting up the CLCx:

- Disable CLCx by clearing the LCxEN bit.
- Select desired inputs using CLCxSEL0 and CLCxSEL1 registers (See [Table 20-3](#)).
- Clear any associated ANSEL bits.
- Set all TRIS bits associated with inputs.
- Clear all TRIS bits associated with outputs.
- Enable the chosen inputs through the four gates using CLCxGLS0, CLCxGLS1, CLCxGLS2, and CLCxGLS3 registers.
- Select the gate output polarities with the LCxPOLy bits of the CLCxPOL register.
- Select the desired logic function with the LCxMODE<2:0> bits of the CLCxCON register.
- Select the desired polarity of the logic output with the LCxPOL bit of the CLCxPOL register. (This step may be combined with the previous gate output polarity step).
- If driving a device, set the LCxOE bit in the CLCxCON register and also clear the TRIS bit corresponding to that output.
- If interrupts are desired, configure the following bits:
  - Set the LCxINTP bit in the CLCxCON register for rising event.
  - Set the LCxINTN bit in the CLCxCON register or falling event.
  - Set the CLCxIE bit of the associated PIE registers.
  - Set the GIE and PEIE bits of the INTCON register.
- Enable the CLCx by setting the LCxEN bit of the CLCxCON register.

## 20.2 CLCx Interrupts

An interrupt will be generated upon a change in the output value of the CLCx when the appropriate interrupt enables are set. A rising edge detector and a falling edge detector are present in each CLC for this purpose.

The CLCxIF bit of the associated PIR registers will be set when either edge detector is triggered and its associated enable bit is set. The LCxINTP enables rising edge interrupts and the LCxINTN bit enables falling edge interrupts. Both are located in the CLCxCON register.

To fully enable the interrupt, set the following bits:

- LCxON bit of the CLCxCON register
- CLCxIE bit of the associated PIE registers
- LCxINTP bit of the CLCxCON register (for a rising edge detection)
- LCxINTN bit of the CLCxCON register (for a falling edge detection)
- PEIE and GIE bits of the INTCON register

The CLCxIF bit of the associated PIR registers, must be cleared in software as part of the interrupt service. If another edge is detected while this flag is being cleared, the flag will still be set at the end of the sequence.

## 20.3 Output Mirror Copies

Mirror copies of all LCxCON output bits are contained in the CLCxDATA register. Reading this register reads the outputs of all CLCs simultaneously. This prevents any reading skew introduced by testing or reading the CLCxOUT bits in the individual CLCxCON registers.

## 20.4 Effects of a Reset

The CLCxCON register is cleared to zero as the result of a Reset. All other selection and gating values remain unchanged.

## 20.5 Operation During Sleep

The CLC module operates independently from the system clock and will continue to run during Sleep, provided that the input sources selected remain active.

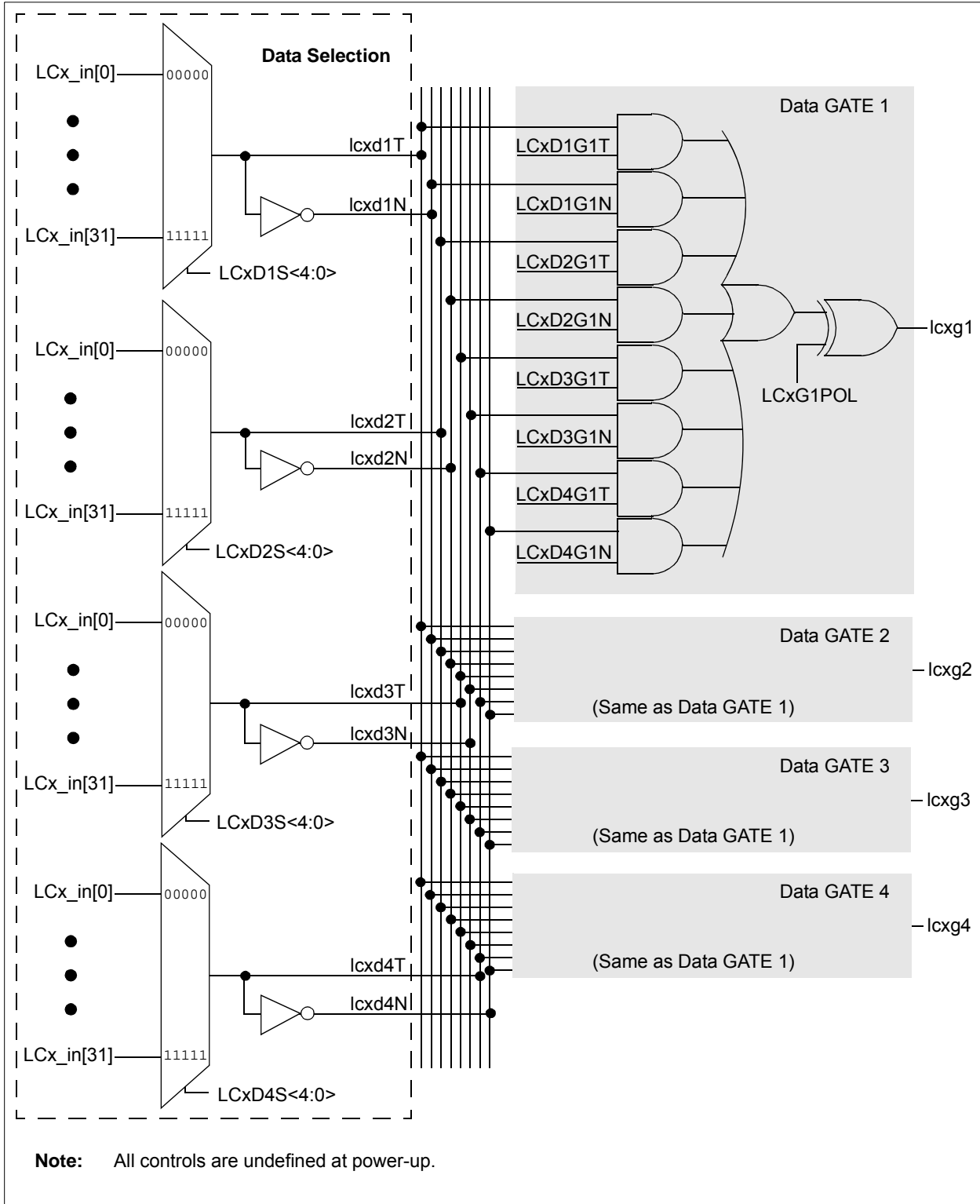
The HFINTOSC remains active during Sleep when the CLC module is enabled and the HFINTOSC is selected as an input source, regardless of the system clock source selected.

In other words, if the HFINTOSC is simultaneously selected as the system clock and as a CLC input source, when the CLC is enabled, the CPU will go idle during Sleep, but the CLC will continue to operate and the HFINTOSC will remain active.

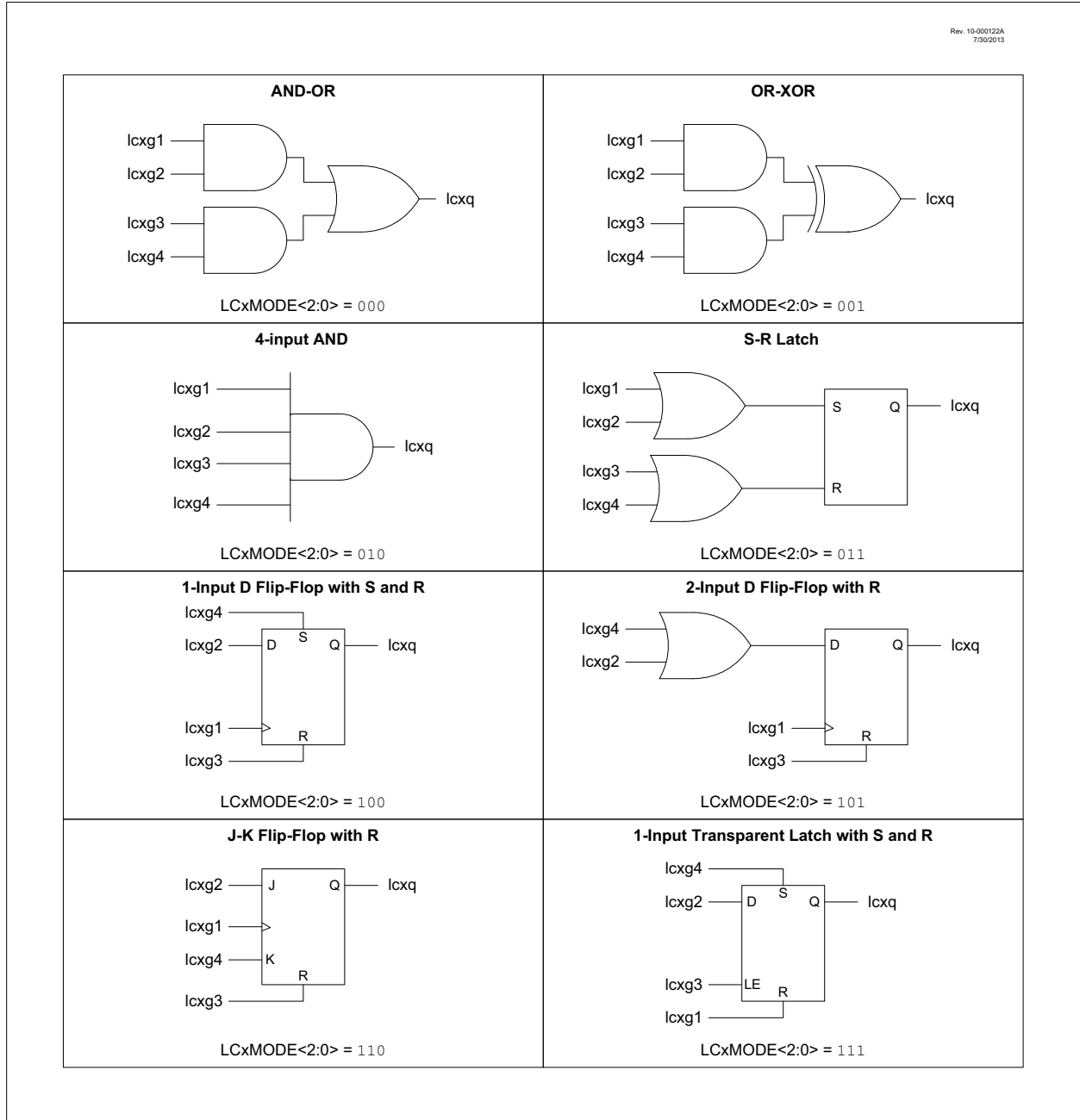
This will have a direct effect on the Sleep mode current.

# PIC16(L)F1507

**FIGURE 20-2: INPUT DATA SELECTION AND GATING**



**FIGURE 20-3: PROGRAMMABLE LOGIC FUNCTIONS**



# PIC16(L)F1507

## 20.6 Register Definitions: CLC Control

### REGISTER 20-1: CLCxCON: CONFIGURABLE LOGIC CELL CONTROL REGISTER

R/W-0/0	R/W-0/0	R-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
LCxEN	LCxOE	LCxOUT	LCxINTP	LCxINTN	LCxMODE<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      **LCxEN:** Configurable Logic Cell Enable bit  
1 = Configurable logic cell is enabled and mixing input signals  
0 = Configurable logic cell is disabled and has logic zero output
- bit 6      **LCxOE:** Configurable Logic Cell Output Enable bit  
1 = Configurable logic cell port pin output enabled  
0 = Configurable logic cell port pin output disabled
- bit 5      **LCxOUT:** Configurable Logic Cell Data Output bit  
Read-only: logic cell output data, after LCxPOL; sampled from lcx\_out wire.
- bit 4      **LCxINTP:** Configurable Logic Cell Positive Edge Going Interrupt Enable bit  
1 = CLCxIF will be set when a rising edge occurs on lcx\_out  
0 = CLCxIF will not be set
- bit 3      **LCxINTN:** Configurable Logic Cell Negative Edge Going Interrupt Enable bit  
1 = CLCxIF will be set when a falling edge occurs on lcx\_out  
0 = CLCxIF will not be set
- bit 2-0    **LCxMODE<2:0>:** Configurable Logic Cell Functional Mode bits  
111 = Cell is 1-input transparent latch with S and R  
110 = Cell is J-K flip-flop with R  
101 = Cell is 2-input D flip-flop with R  
100 = Cell is 1-input D flip-flop with S and R  
011 = Cell is S-R latch  
010 = Cell is 4-input AND  
001 = Cell is OR-XOR  
000 = Cell is AND-OR



## REGISTER 20-2: CLCxPOL: SIGNAL POLARITY CONTROL REGISTER

R/W-0/0	U-0	U-0	U-0	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
LCxPOL	—	—	—	LCxG4POL	LCxG3POL	LCxG2POL	LCxG1POL
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      **LCxPOL:** LCOOUT Polarity Control bit  
1 = The output of the logic cell is inverted  
0 = The output of the logic cell is not inverted
- bit 6-4    **Unimplemented:** Read as '0'
- bit 3      **LCxG4POL:** Gate 4 Output Polarity Control bit  
1 = The output of gate 4 is inverted when applied to the logic cell  
0 = The output of gate 4 is not inverted
- bit 2      **LCxG3POL:** Gate 3 Output Polarity Control bit  
1 = The output of gate 3 is inverted when applied to the logic cell  
0 = The output of gate 3 is not inverted
- bit 1      **LCxG2POL:** Gate 2 Output Polarity Control bit  
1 = The output of gate 2 is inverted when applied to the logic cell  
0 = The output of gate 2 is not inverted
- bit 0      **LCxG1POL:** Gate 1 Output Polarity Control bit  
1 = The output of gate 1 is inverted when applied to the logic cell  
0 = The output of gate 1 is not inverted

# PIC16(L)F1507

## REGISTER 20-3: CLCxSELO: MULTIPLEXER DATA 1 AND 2 SELECT REGISTER

U-0	R/W-x/u	R/W-x/u	R/W-x/u	U-0	R/W-x/u	R/W-x/u	R/W-x/u
—	LCxD2S<2:0> <sup>(1)</sup>			—	LCxD1S<2:0> <sup>(1)</sup>		
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 6-4       **LCxD2S<2:0>:** Input Data 2 Selection Control bits<sup>(1)</sup>
- 111 = LCx\_in[11] is selected for lcx2
  - 110 = LCx\_in[10] is selected for lcx2
  - 101 = LCx\_in[9] is selected for lcx2
  - 100 = LCx\_in[8] is selected for lcx2
  - 011 = LCx\_in[7] is selected for lcx2
  - 010 = LCx\_in[6] is selected for lcx2
  - 001 = LCx\_in[5] is selected for lcx2
  - 000 = LCx\_in[4] is selected for lcx2
- bit 3           **Unimplemented:** Read as '0'
- bit 2-0       **LCxD1S<2:0>:** Input Data 1 Selection Control bits<sup>(1)</sup>
- 111 = LCx\_in[7] is selected for lcx1
  - 110 = LCx\_in[6] is selected for lcx1
  - 101 = LCx\_in[5] is selected for lcx1
  - 100 = LCx\_in[4] is selected for lcx1
  - 011 = LCx\_in[3] is selected for lcx1
  - 010 = LCx\_in[2] is selected for lcx1
  - 001 = LCx\_in[1] is selected for lcx1
  - 000 = LCx\_in[0] is selected for lcx1

**Note 1:** See [Table 20-1](#) for signal names associated with inputs.

## REGISTER 20-4: CLCxSEL1: MULTIPLEXER DATA 3 AND 4 SELECT REGISTER

U-0	R/W-x/u	R/W-x/u	R/W-x/u	U-0	R/W-x/u	R/W-x/u	R/W-x/u
—	LCxD4S<2:0> <sup>(1)</sup>			—	LCxD3S<2:0> <sup>(1)</sup>		
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 6-4 **LCxD4S<2:0>:** Input Data 4 Selection Control bits<sup>(1)</sup>

111 = LCx\_in[3] is selected for lcx4

110 = LCx\_in[2] is selected for lcx4

101 = LCx\_in[1] is selected for lcx4

100 = LCx\_in[0] is selected for lcx4

011 = LCx\_in[15] is selected for lcx4

010 = LCx\_in[14] is selected for lcx4

001 = LCx\_in[13] is selected for lcx4

000 = LCx\_in[12] is selected for lcx4

bit 3 **Unimplemented:** Read as '0'

bit 2-0 **LCxD3S<2:0>:** Input Data 3 Selection Control bits<sup>(1)</sup>

111 = LCx\_in[15] is selected for lcx3

110 = LCx\_in[14] is selected for lcx3

101 = LCx\_in[13] is selected for lcx3

100 = LCx\_in[12] is selected for lcx3

011 = LCx\_in[11] is selected for lcx3

010 = LCx\_in[10] is selected for lcx3

001 = LCx\_in[9] is selected for lcx3

000 = LCx\_in[8] is selected for lcx3

**Note 1:** See [Table 20-1](#) for signal names associated with inputs.

# PIC16(L)F1507

## REGISTER 20-5: CLCxGLS0: GATE 1 LOGIC SELECT 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
LCxG1D4T	LCxG1D4N	LCxG1D3T	LCxG1D3N	LCxG1D2T	LCxG1D2N	LCxG1D1T	LCxG1D1N
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      **LCxG1D4T:** Gate 1 Data 4 True (non-inverted) bit  
1 = lcx4T is gated into lcxg1  
0 = lcx4T is not gated into lcxg1
- bit 6      **LCxG1D4N:** Gate 1 Data 4 Negated (inverted) bit  
1 = lcx4N is gated into lcxg1  
0 = lcx4N is not gated into lcxg1
- bit 5      **LCxG1D3T:** Gate 1 Data 3 True (non-inverted) bit  
1 = lcx3T is gated into lcxg1  
0 = lcx3T is not gated into lcxg1
- bit 4      **LCxG1D3N:** Gate 1 Data 3 Negated (inverted) bit  
1 = lcx3N is gated into lcxg1  
0 = lcx3N is not gated into lcxg1
- bit 3      **LCxG1D2T:** Gate 1 Data 2 True (non-inverted) bit  
1 = lcx2T is gated into lcxg1  
0 = lcx2T is not gated into lcxg1
- bit 2      **LCxG1D2N:** Gate 1 Data 2 Negated (inverted) bit  
1 = lcx2N is gated into lcxg1  
0 = lcx2N is not gated into lcxg1
- bit 1      **LCxG1D1T:** Gate 1 Data 1 True (non-inverted) bit  
1 = lcx1T is gated into lcxg1  
0 = lcx1T is not gated into lcxg1
- bit 0      **LCxG1D1N:** Gate 1 Data 1 Negated (inverted) bit  
1 = lcx1N is gated into lcxg1  
0 = lcx1N is not gated into lcxg1

## REGISTER 20-6: CLCxGLS1: GATE 2 LOGIC SELECT 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
LCxG2D4T	LCxG2D4N	LCxG2D3T	LCxG2D3N	LCxG2D2T	LCxG2D2N	LCxG2D1T	LCxG2D1N
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      **LCxG2D4T:** Gate 2 Data 4 True (non-inverted) bit  
1 = lcx4T is gated into lcxg2  
0 = lcx4T is not gated into lcxg2
- bit 6      **LCxG2D4N:** Gate 2 Data 4 Negated (inverted) bit  
1 = lcx4N is gated into lcxg2  
0 = lcx4N is not gated into lcxg2
- bit 5      **LCxG2D3T:** Gate 2 Data 3 True (non-inverted) bit  
1 = lcx3T is gated into lcxg2  
0 = lcx3T is not gated into lcxg2
- bit 4      **LCxG2D3N:** Gate 2 Data 3 Negated (inverted) bit  
1 = lcx3N is gated into lcxg2  
0 = lcx3N is not gated into lcxg2
- bit 3      **LCxG2D2T:** Gate 2 Data 2 True (non-inverted) bit  
1 = lcx2T is gated into lcxg2  
0 = lcx2T is not gated into lcxg2
- bit 2      **LCxG2D2N:** Gate 2 Data 2 Negated (inverted) bit  
1 = lcx2N is gated into lcxg2  
0 = lcx2N is not gated into lcxg2
- bit 1      **LCxG2D1T:** Gate 2 Data 1 True (non-inverted) bit  
1 = lcx1T is gated into lcxg2  
0 = lcx1T is not gated into lcxg2
- bit 0      **LCxG2D1N:** Gate 2 Data 1 Negated (inverted) bit  
1 = lcx1N is gated into lcxg2  
0 = lcx1N is not gated into lcxg2

# PIC16(L)F1507

## REGISTER 20-7: CLCxGLS2: GATE 3 LOGIC SELECT 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
LCxG3D4T	LCxG3D4N	LCxG3D3T	LCxG3D3N	LCxG3D2T	LCxG3D2N	LCxG3D1T	LCxG3D1N
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      **LCxG3D4T:** Gate 3 Data 4 True (non-inverted) bit  
           1 = lcx4T is gated into lcxg3  
           0 = lcx4T is not gated into lcxg3
- bit 6      **LCxG3D4N:** Gate 3 Data 4 Negated (inverted) bit  
           1 = lcx4N is gated into lcxg3  
           0 = lcx4N is not gated into lcxg3
- bit 5      **LCxG3D3T:** Gate 3 Data 3 True (non-inverted) bit  
           1 = lcx3T is gated into lcxg3  
           0 = lcx3T is not gated into lcxg3
- bit 4      **LCxG3D3N:** Gate 3 Data 3 Negated (inverted) bit  
           1 = lcx3N is gated into lcxg3  
           0 = lcx3N is not gated into lcxg3
- bit 3      **LCxG3D2T:** Gate 3 Data 2 True (non-inverted) bit  
           1 = lcx2T is gated into lcxg3  
           0 = lcx2T is not gated into lcxg3
- bit 2      **LCxG3D2N:** Gate 3 Data 2 Negated (inverted) bit  
           1 = lcx2N is gated into lcxg3  
           0 = lcx2N is not gated into lcxg3
- bit 1      **LCxG3D1T:** Gate 3 Data 1 True (non-inverted) bit  
           1 = lcx1T is gated into lcxg3  
           0 = lcx1T is not gated into lcxg3
- bit 0      **LCxG3D1N:** Gate 3 Data 1 Negated (inverted) bit  
           1 = lcx1N is gated into lcxg3  
           0 = lcx1N is not gated into lcxg3

## REGISTER 20-8: CLCxGLS3: GATE 4 LOGIC SELECT 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
LCxG4D4T	LCxG4D4N	LCxG4D3T	LCxG4D3N	LCxG4D2T	LCxG4D2N	LCxG4D1T	LCxG4D1N
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	<b>LCxG4D4T:</b> Gate 4 Data 4 True (non-inverted) bit 1 = lcx4T is gated into lcxg4 0 = lcx4T is not gated into lcxg4
bit 6	<b>LCxG4D4N:</b> Gate 4 Data 4 Negated (inverted) bit 1 = lcx4N is gated into lcxg4 0 = lcx4N is not gated into lcxg4
bit 5	<b>LCxG4D3T:</b> Gate 4 Data 3 True (non-inverted) bit 1 = lcx3T is gated into lcxg4 0 = lcx3T is not gated into lcxg4
bit 4	<b>LCxG4D3N:</b> Gate 4 Data 3 Negated (inverted) bit 1 = lcx3N is gated into lcxg4 0 = lcx3N is not gated into lcxg4
bit 3	<b>LCxG4D2T:</b> Gate 4 Data 2 True (non-inverted) bit 1 = lcx2T is gated into lcxg4 0 = lcx2T is not gated into lcxg4
bit 2	<b>LCxG4D2N:</b> Gate 4 Data 2 Negated (inverted) bit 1 = lcx2N is gated into lcxg4 0 = lcx2N is not gated into lcxg4
bit 1	<b>LCxG4D1T:</b> Gate 4 Data 1 True (non-inverted) bit 1 = lcx1T is gated into lcxg4 0 = lcx1T is not gated into lcxg4
bit 0	<b>LCxG4D1N:</b> Gate 4 Data 1 Negated (inverted) bit 1 = lcx1N is gated into lcxg4 0 = lcx1N is not gated into lcxg4

# PIC16(L)F1507

---

## REGISTER 20-9: CLCDATA: CLC DATA OUTPUT

U-0	U-0	U-0	U-0	U-0	U-0	R-0	R-0
—	—	—	—	—	—	MLC2OUT	MLC1OUT
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 Unimplemented: Read as '0'

bit 1 MLC2OUT: Mirror copy of LC2OUT bit

bit 0 MLC1OUT: Mirror copy of LC1OUT bit



**TABLE 20-3: SUMMARY OF REGISTERS ASSOCIATED WITH CLCx**

Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Register on Page
ANSELA	—	—	—	ANSA4	—	ANSA2	ANSA1	ANSA0	99
ANSELB	—	—	ANSB5	ANSB4	—	—	—	—	103
ANSELC	ANSC7	ANSC6	—	—	ANSC3	ANSC2	ANSC1	ANSC0	107
CLC1CON	LC1EN	LC1OE	LC1OUT	LC1INTP	LC1INTN	LC1MODE<2:0>			160
CLCDATA	—	—	—	—	—	MLC3OUT	MLC2OUT	MLC1OUT	168
CLC1GLS0	LC1G1D4T	LC1G1D4N	LC1G1D3T	LC1G1D3N	LC1G1D2T	LC1G1D2N	LC1G1D1T	LC1G1D1N	164
CLC1GLS1	LC1G2D4T	LC1G2D4N	LC1G2D3T	LC1G2D3N	LC1G2D2T	LC1G2D2N	LC1G2D1T	LC1G2D1N	165
CLC1GLS2	LC1G3D4T	LC1G3D4N	LC1G3D3T	LC1G3D3N	LC1G3D2T	LC1G3D2N	LC1G3D1T	LC1G3D1N	166
CLC1GLS3	LC1G4D4T	LC1G4D4N	LC1G4D3T	LC1G4D3N	LC1G4D2T	LC1G4D2N	LC1G4D1T	LC1G4D1N	167
CLC1POL	LC1POL	—	—	—	LC1G4POL	LC1G3POL	LC1G2POL	LC1G1POL	161
CLC1SEL0	—	LC1D2S<2:0>			—	LC1D1S<2:0>			162
CLC1SEL1	—	LC1D4S<2:0>			—	LC1D3S<2:0>			163
CLC2CON	LC2EN	LC2OE	LC2OUT	LC2INTP	LC2INTN	LC2MODE<2:0>			160
CLC2GLS0	LC2G1D4T	LC2G1D4N	LC2G1D3T	LC2G1D3N	LC2G1D2T	LC2G1D2N	LC2G1D1T	LC2G1D1N	164
CLC2GLS1	LC2G2D4T	LC2G2D4N	LC2G2D3T	LC2G2D3N	LC2G2D2T	LC2G2D2N	LC2G2D1T	LC2G2D1N	165
CLC2GLS2	LC2G3D4T	LC2G3D4N	LC2G3D3T	LC2G3D3N	LC2G3D2T	LC2G3D2N	LC2G3D1T	LC2G3D1N	166
CLC2GLS3	LC2G4D4T	LC2G4D4N	LC2G4D3T	LC2G4D3N	LC2G4D2T	LC2G4D2N	LC2G4D1T	LC2G4D1N	167
CLC2POL	LC2POL	—	—	—	LC2G4POL	LC2G3POL	LC2G2POL	LC2G1POL	161
CLC2SEL0	—	LC2D2S<2:0>			—	LC2D1S<2:0>			162
CLC2SEL1	—	LC2D4S<2:0>			—	LC2D3S<2:0>			163
INTCON	GIE	PEIE	TMR0IE	INTE	IOCF	TMR0IF	INTF	IOCF	64
PIE3	—	—	—	—	—	—	CLC2IE	CLC1IE	67
PIR3	—	—	—	—	—	—	CLC2IF	CLC1IF	70
TRISA	—	—	TRISA5	TRISA4	— <sup>(1)</sup>	TRISA2	TRISA1	TRISA0	98
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	—	—	—	—	102
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	106

**Legend:** — = unimplemented read as '0'. Shaded cells are not used for CLC module.

**Note 1:** Unimplemented, read as '1'.

# PIC16(L)F1507

## 21.0 NUMERICALLY CONTROLLED OSCILLATOR (NCO) MODULE

The Numerically Controlled Oscillator (NCOx) module is a timer that uses the overflow from the addition of an increment value to divide the input frequency. The advantage of the addition method over simple counter driven timer is that the resolution of division does not vary with the divider value. The NCOx is most useful for applications that require frequency accuracy and fine resolution at a fixed duty cycle.

Features of the NCOx include:

- 16-bit increment function
- Fixed Duty Cycle (FDC) mode
- Pulse Frequency (PF) mode
- Output pulse width control
- Multiple clock input sources
- Output polarity control
- Interrupt capability

Figure 21-1 is a simplified block diagram of the NCOx module.

### 21.1 NCOx Operation

The NCOx operates by repeatedly adding a fixed value to an accumulator. Additions occur at the input clock rate. The accumulator will overflow with a carry periodically, which is the raw NCOx output (NCO\_overflow). This effectively reduces the input clock by the ratio of the addition value to the maximum accumulator value. See Equation 21-1.

The NCOx output can be further modified by stretching the pulse or toggling a flip-flop. The modified NCOx output is then distributed internally to other peripherals and optionally output to a pin. The accumulator overflow also generates an interrupt (NCO\_interrupt).

The NCOx period changes in discrete steps to create an average frequency. This output depends on the ability of the receiving circuit (i.e., CWG or external resonant converter circuitry) to average the NCOx output to reduce uncertainty.

#### 21.1.1 NCOx CLOCK SOURCES

Clock sources available to the NCOx include:

- HFINTOSC
- FOSC
- LC1\_out
- CLKIN pin

The NCOx clock source is selected by configuring the NxCCKS<2:0> bits in the NCOxCLK register.

#### EQUATION 21-1:

$$F_{OVERFLOW} = \frac{NCO \text{ Clock Frequency} \times \text{Increment Value}}{2^n}$$

*n* = Accumulator width in bits

#### 21.1.2 ACCUMULATOR

The accumulator is a 20-bit register. Read and write access to the accumulator is available through three registers:

- NCOxACCL
- NCOxACCH
- NCOxACCU

#### 21.1.3 ADDER

The NCOx adder is a full adder, which operates independently from the system clock. The addition of the previous result and the increment value replaces the accumulator value on the rising edge of each input clock.

#### 21.1.4 INCREMENT REGISTERS

The increment value is stored in two 8-bit registers making up a 16-bit increment. In order of LSB to MSB they are:

- NCOxINCL
- NCOxINCH

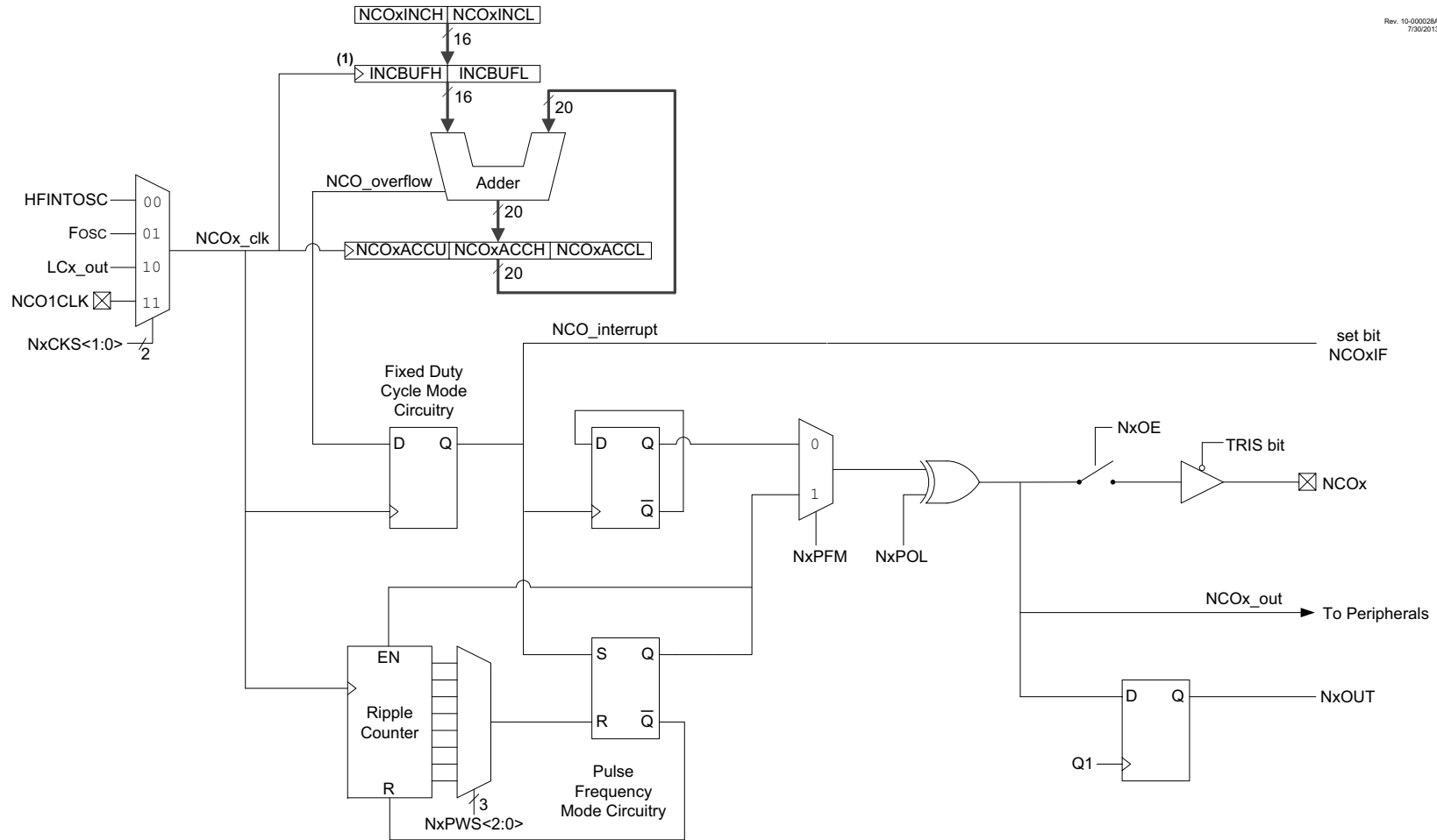
When the NCO module is enabled, the NCOxINCH should be written first, then the NCOxINCL register. Writing to the NCOxINCL register initiates the increment buffer registers to be loaded simultaneously on the second rising edge of the NCOx\_clk signal.

The registers are readable and writable. The increment registers are double-buffered to allow value changes to be made without first disabling the NCOx module.

When the NCO module is disabled, the increment buffers are loaded immediately after a write to the increment registers.

**Note:** The increment buffer registers are not user-accessible.

**FIGURE 21-1: NUMERICALLY CONTROLLED OSCILLATOR (NCOx) MODULE SIMPLIFIED BLOCK DIAGRAM**



**Note 1:** The increment registers are double-buffered to allow for value changes to be made without first disabling the NCO module. The full increment value is loaded into the buffer registers on the second rising edge of the  $NCOx\_clk$  signal that occurs immediately after a write to  $NCOxINCL$  register. The buffers are not user-accessible and are shown here for reference.

Rev. 10-000025A  
7/20/2013

# PIC16(L)F1507

---

## 21.2 Fixed Duty Cycle (FDC) Mode

In Fixed Duty Cycle (FDC) mode, every time the accumulator overflows (NCO\_overflow), the output is toggled. This provides a 50% duty cycle, provided that the increment value remains constant. For more information, see [Figure 21-2](#).

The FDC mode is selected by clearing the NxPFM bit in the NCOxCON register.

## 21.3 Pulse Frequency (PF) Mode

In Pulse Frequency (PF) mode, every time the accumulator overflows (NCO\_overflow), the output becomes active for one or more clock periods. Once the clock period expires, the output returns to an inactive state. This provides a pulsed output.

The output becomes active on the rising clock edge immediately following the overflow event. For more information, see [Figure 21-2](#).

The value of the active and inactive states depends on the polarity bit, NxPOL in the NCOxCON register.

The PF mode is selected by setting the NxPFM bit in the NCOxCON register.

### 21.3.1 OUTPUT PULSE WIDTH CONTROL

When operating in PF mode, the active state of the output can vary in width by multiple clock periods. Various pulse widths are selected with the NxPWS<2:0> bits in the NCOxCLK register.

When the selected pulse width is greater than the accumulator overflow time frame, the output of the NCOx operation is indeterminate.

## 21.4 Output Polarity Control

The last stage in the NCOx module is the output polarity. The NxPOL bit in the NCOxCON register selects the output polarity. Changing the polarity while the interrupts are enabled will cause an interrupt for the resulting output transition.

The NCOx output can be used internally by source code or other peripherals. Accomplish this by reading the NxOUT (read-only) bit of the NCOxCON register.

The NCOx output signal is available to the following peripherals:

- CLC
- CWG

## 21.5 Interrupts

When the accumulator overflows (NCO\_overflow), the NCOx Interrupt Flag bit, NCOxIF, of the PIRx register is set. To enable the interrupt event (NCO\_interrupt), the following bits must be set:

- NxEN bit of the NCOxCON register
- NCOxIE bit of the PIEx register
- PEIE bit of the INTCON register
- GIE bit of the INTCON register

The interrupt must be cleared by software by clearing the NCOxIF bit in the Interrupt Service Routine.

## 21.6 Effects of a Reset

All of the NCOx registers are cleared to zero as the result of a Reset.

## 21.7 Operation In Sleep

The NCO module operates independently from the system clock and will continue to run during Sleep, provided that the clock source selected remains active.

The HFINTOSC remains active during Sleep when the NCO module is enabled and the HFINTOSC is selected as the clock source, regardless of the system clock source selected.

In other words, if the HFINTOSC is simultaneously selected as the system clock and the NCO clock source, when the NCO is enabled, the CPU will go idle during Sleep, but the NCO will continue to operate and the HFINTOSC will remain active.

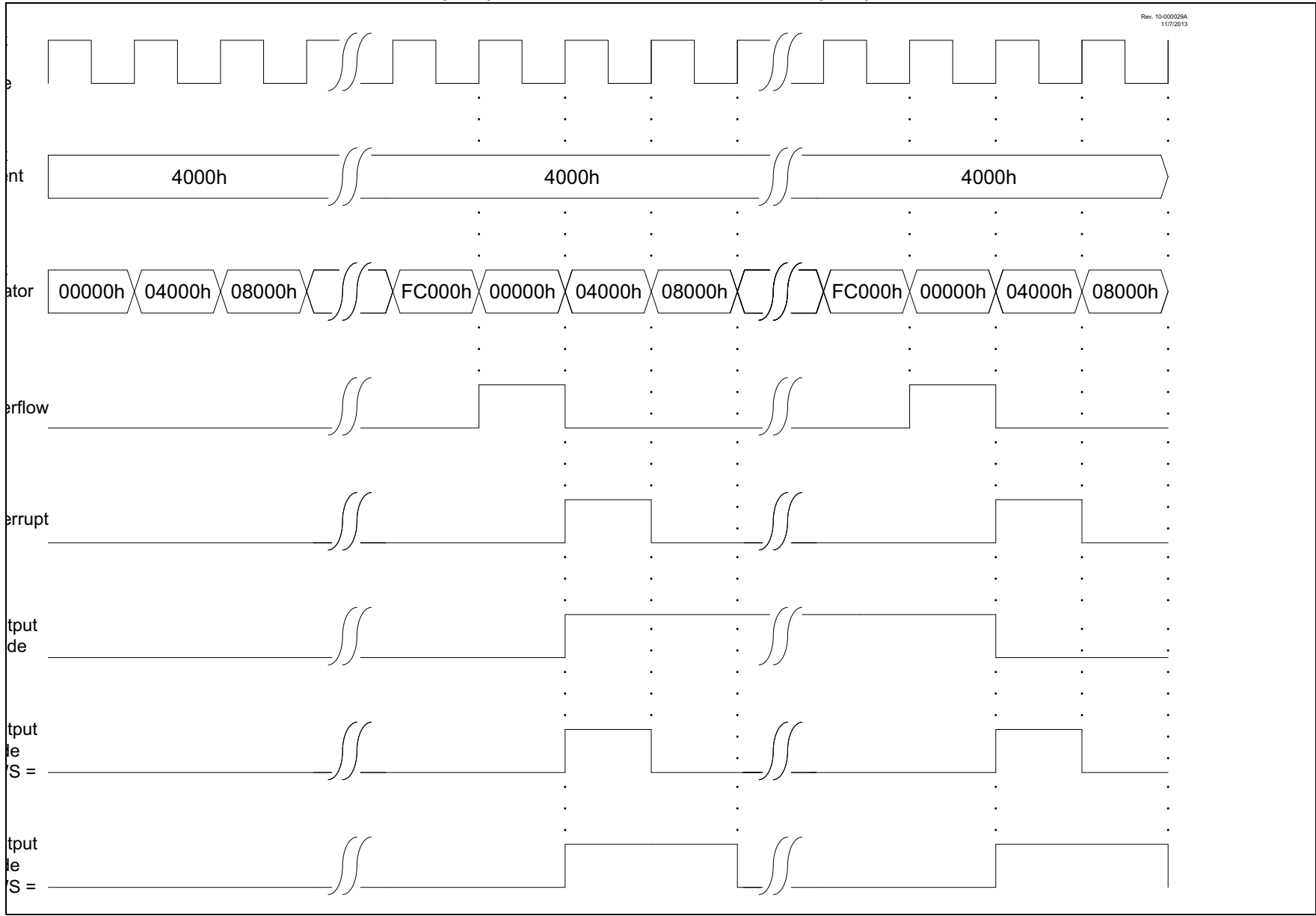
This will have a direct effect on the Sleep mode current.

## 21.8 Alternate Pin Locations

This module incorporates I/O pins that can be moved to other locations with the use of the alternate pin function register, APFCON. To determine which pins can be moved and what their default locations are upon a Reset, see [Section 11.1 “Alternate Pin Function”](#) for more information.

FIGURE 21-2: NCO – FIXED DUTY CYCLE (FDC) AND PULSE FREQUENCY MODE (PFM) OUTPUT OPERATION DIAGRAM

Rev. 10-000029A  
11/17/2013



# PIC16(L)F1507

## 21.9 Register Definitions: NCOx Control Registers

### REGISTER 21-1: NCOxCON: NCOx CONTROL REGISTER

R/W-0/0	R/W-0/0	R-0/0	R/W-0/0	U-0	U-0	U-0	R/W-0/0
NxEN	NxOE	NxOUT	NxPOL	—	—	—	NxPFM
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            **NxEN:** NCOx Enable bit  
1 = NCOx module is enabled  
0 = NCOx module is disabled
- bit 6            **NxOE:** NCOx Output Enable bit  
1 = NCOx output pin is enabled  
0 = NCOx output pin is disabled
- bit 5            **NxOUT:** NCOx Output bit  
1 = NCOx output is high  
0 = NCOx output is low
- bit 4            **NxPOL:** NCOx Polarity bit  
1 = NCOx output signal is active low (inverted)  
0 = NCOx output signal is active high (non-inverted)
- bit 3-1         **Unimplemented:** Read as '0'
- bit 0            **NxPFM:** NCOx Pulse Frequency Mode bit  
1 = NCOx operates in Pulse Frequency mode  
0 = NCOx operates in Fixed Duty Cycle mode

### REGISTER 21-2: NCOxCLK: NCOx INPUT CLOCK CONTROL REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	U-0	U-0	U-0	R/W-0/0	R/W-0/0
NxPWS<2:0> <sup>(1, 2)</sup>			—	—	—	NxCKS<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 7-5         **NxPWS<2:0>:** NCOx Output Pulse Width Select bits<sup>(1, 2)</sup>  
111 = 128 NCOx clock periods  
110 = 64 NCOx clock periods  
101 = 32 NCOx clock periods  
100 = 16 NCOx clock periods  
011 = 8 NCOx clock periods  
010 = 4 NCOx clock periods  
001 = 2 NCOx clock periods  
000 = 1 NCOx clock periods
- bit 4-2         **Unimplemented:** Read as '0'
- bit 1-0         **NxCKS<1:0>:** NCOx Clock Source Select bits  
11 = NCO1CLK pin  
10 = LC1\_out  
01 = Fosc  
00 = HFINTOSC (16 MHz)

**Note 1:** NxPWS applies only when operating in Pulse Frequency mode.  
**2:** If NCOx pulse width is greater than NCO\_overflow period, operation is indeterminate.

## REGISTER 21-3: NCOxACCL: NCOx ACCUMULATOR REGISTER – LOW BYTE

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
NCOxACC<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      **NCOxACC<7:0>**: NCOx Accumulator, Low Byte

## REGISTER 21-4: NCOxACCH: NCOx ACCUMULATOR REGISTER – HIGH BYTE

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
NCOxACC<15:8>							
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 7-0      **NCOxACC<15:8>**: NCOx Accumulator, High Byte

## REGISTER 21-5: NCOxACCU: NCOx ACCUMULATOR REGISTER – UPPER BYTE

U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	NCOxACC<19:16>			
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 3-0      **NCOxACC<19:16>**: NCOx Accumulator, Upper Byte

# PIC16(L)F1507

**REGISTER 21-6: NCOxINCL: NCOx INCREMENT REGISTER – LOW BYTE<sup>(1)</sup>**

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-1/1
NCOxINC<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                  **NCOxINC<7:0>**: NCOx Increment, Low Byte

**Note 1:** Write the NCOxINCH register first, then the NCOxINCL register. See [21.1.4 “Increment Registers”](#) for more information.

**REGISTER 21-7: NCOxINCH: NCOx INCREMENT REGISTER – HIGH BYTE<sup>(1)</sup>**

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
NCOxINC<15:8>							
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 7-0                  **NCOxINC<15:8>**: NCOx Increment, High Byte

**Note 1:** Write the NCOxINCH register first, then the NCOxINCL register. See [21.1.4 “Increment Registers”](#) for more information.

**TABLE 21-1: SUMMARY OF REGISTERS ASSOCIATED WITH NCOx**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
APFCON	—	—	—	—	—	—	CLC1SEL	NCO1SEL	96
INTCON	GIE	PEIE	TMR0IE	INTE	IOCFIE	TMR0IF	INTF	IOCFIF	64
NCO1ACCCH	NCO1ACC<15:8>								175
NCO1ACCCL	NCO1ACC<7:0>								175
NCO1ACCU	—				NCO1ACC<19:16>				175
NCO1CLK	N1PWS<2:0>			—	—	—	N1CKS<1:0>		174
NCO1CON	N1EN	N1OE	N1OUT	N1POL	—	—	—	N1PFM	174
NCO1INCH	NCO1INC<15:8>								176
NCO1INCL	NCO1INC<7:0>								176
PIE2	—	—	—	—	—	NCO1IE	—	—	66
PIR2	—	—	—	—	—	NCO1IF	—	—	69
TRISA	—	—	TRISA5	TRISA4	— <sup>(1)</sup>	TRISA2	TRISA1	TRISA0	98
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	106

**Legend:** x = unknown, u = unchanged, — = unimplemented read as '0', q = value depends on condition. Shaded cells are not used for NCOx module.

**Note 1:** Unimplemented, read as '1'.



## 22.0 COMPLEMENTARY WAVEFORM GENERATOR (CWG) MODULE

The Complementary Waveform Generator (CWG) produces a complementary waveform with dead-band delay from a selection of input sources.

The CWG module has the following features:

- Selectable dead-band clock source control
- Selectable input sources
- Output enable control
- Output polarity control
- Dead-band control with independent 6-bit rising and falling edge dead-band counters
- Auto-shutdown control with:
  - Selectable shutdown sources
  - Auto-restart enable
  - Auto-shutdown pin override control

### 22.1 Fundamental Operation

The CWG generates two output waveforms from the selected input source.

The off-to-on transition of each output can be delayed from the on-to-off transition of the other output, thereby, creating a time delay immediately where neither output is driven. This is referred to as dead time and is covered in [Section 22.5 “Dead-Band Control”](#). A typical operating waveform, with dead band, generated from a single input signal is shown in [Figure 22-2](#).

It may be necessary to guard against the possibility of circuit faults or a feedback event arriving too late or not at all. In this case, the active drive must be terminated before the Fault condition causes damage. This is referred to as auto-shutdown and is covered in [Section 22.9 “Auto-Shutdown Control”](#).

### 22.2 Clock Source

The CWG module allows the following clock sources to be selected:

- Fosc (system clock)
- HFINTOSC (16 MHz only)

The clock sources are selected using the G1CS0 bit of the CWGxCON0 register ([Register 22-1](#)).

## 22.3 Selectable Input Sources

The CWG generates the output waveforms from the input sources in [Table 22-1](#).

**TABLE 22-1: SELECTABLE INPUT SOURCES**

Source Peripheral	Signal Name
PWM1	PWM1_out
PWM2	PWM2_out
PWM3	PWM3_out
PWM4	PWM4_out
NCO1	NCO1_out
CLC1	LC1_out

The input sources are selected using the GxIS<2:0> bits in the CWGxCON1 register ([Register 22-2](#)).

## 22.4 Output Control

Immediately after the CWG module is enabled, the complementary drive is configured with both CWGxA and CWGxB drives cleared.

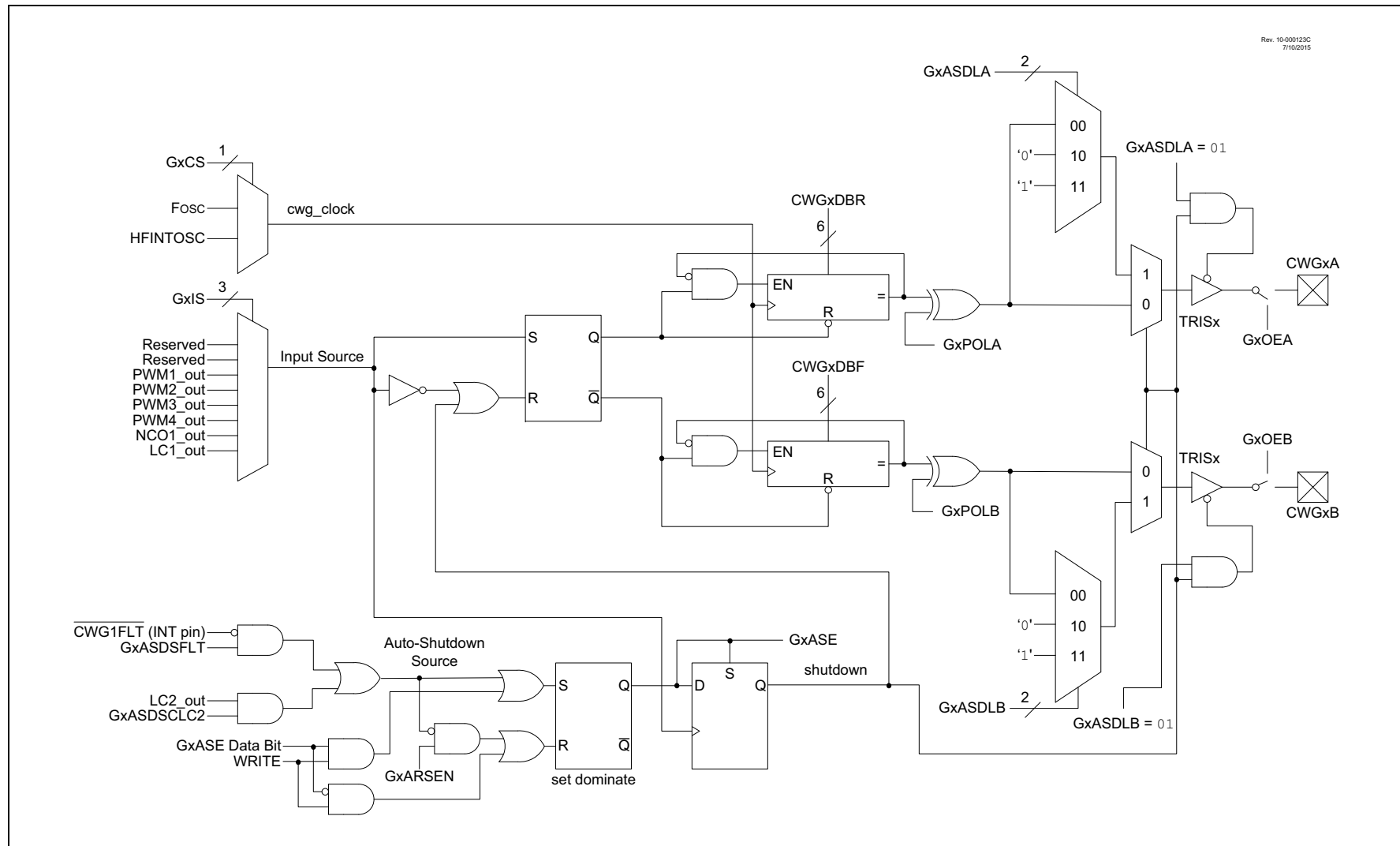
### 22.4.1 OUTPUT ENABLES

Each CWG output pin has individual output enable control. Output enables are selected with the GxOEA and GxOEB bits of the CWGxCON0 register. When an output enable control is cleared, the module asserts no control over the pin. When an output enable is set, the override value or active PWM waveform is applied to the pin per the port priority selection. The output pin enables are dependent on the module enable bit, GxEN. When GxEN is cleared, CWG output enables and CWG drive levels have no effect.

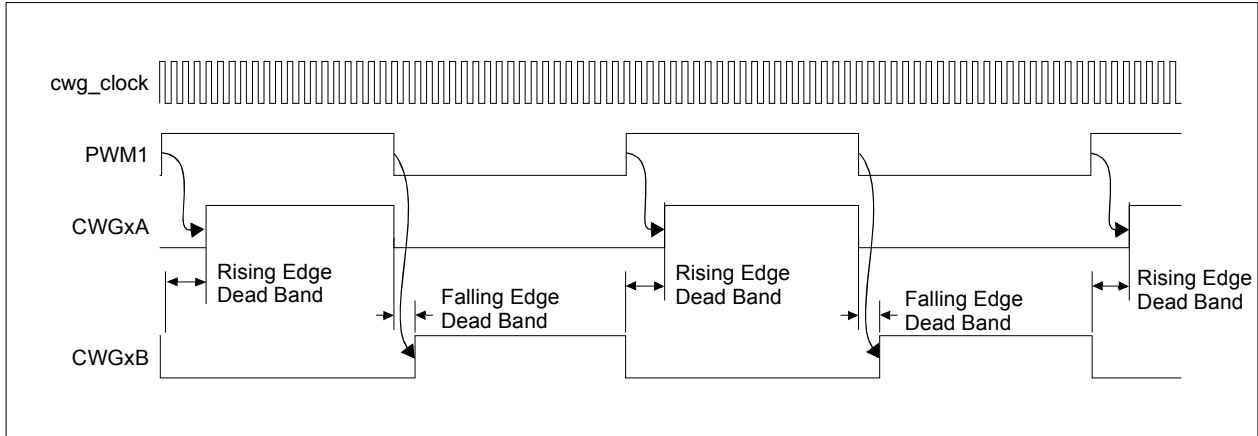
### 22.4.2 POLARITY CONTROL

The polarity of each CWG output can be selected independently. When the output polarity bit is set, the corresponding output is active-high. Clearing the output polarity bit configures the corresponding output as active-low. However, polarity does not affect the override levels. Output polarity is selected with the GxPOLA and GxPOLB bits of the CWGxCON0 register.

FIGURE 22-1: SIMPLIFIED CWG BLOCK DIAGRAM



**FIGURE 22-2: TYPICAL CWG OPERATION WITH PWM1 (NO AUTO-SHUTDOWN)**



## 22.5 Dead-Band Control

Dead-band control provides for non-overlapping output signals to prevent shoot-through current in power switches. The CWG contains two 6-bit dead-band counters. One dead-band counter is used for the rising edge of the input source control. The other is used for the falling edge of the input source control.

Dead band is timed by counting CWG clock periods from zero up to the value in the rising or falling dead-band counter registers. See CWGxDBR and CWGxDBF registers ([Register 22-4](#) and [Register 22-5](#), respectively).

## 22.6 Rising Edge Dead Band

The rising edge dead-band delays the turn-on of the CWGxA output from when the CWGxB output is turned off. The rising edge dead-band time starts when the rising edge of the input source signal goes true. When this happens, the CWGxB output is immediately turned off and the rising edge dead-band delay time starts. When the rising edge dead-band delay time is reached, the CWGxA output is turned on.

The CWGxDBR register sets the duration of the dead-band interval on the rising edge of the input source signal. This duration is from 0 to 64 counts of dead band.

Dead band is always counted off the edge on the input source signal. A count of 0 (zero), indicates that no dead band is present.

If the input source signal is not present for enough time for the count to be completed, no output will be seen on the respective output.

## 22.7 Falling Edge Dead Band

The falling edge dead band delays the turn-on of the CWGxB output from when the CWGxA output is turned off. The falling edge dead-band time starts when the falling edge of the input source goes true. When this happens, the CWGxA output is immediately turned off and the falling edge dead-band delay time starts. When the falling edge dead-band delay time is reached, the CWGxB output is turned on.

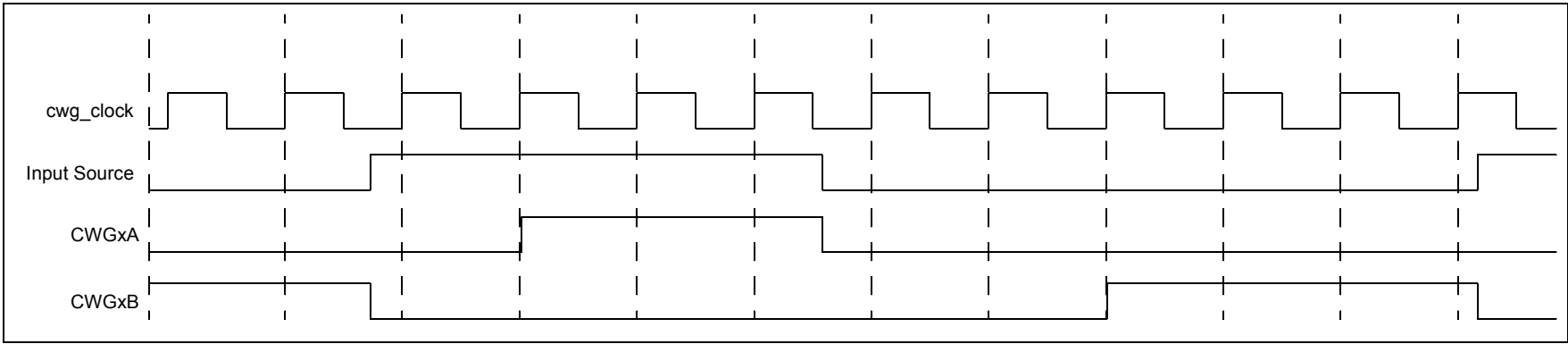
The CWGxDBF register sets the duration of the dead-band interval on the falling edge of the input source signal. This duration is from 0 to 64 counts of dead band.

Dead band is always counted off the edge on the input source signal. A count of 0 (zero), indicates that no dead band is present.

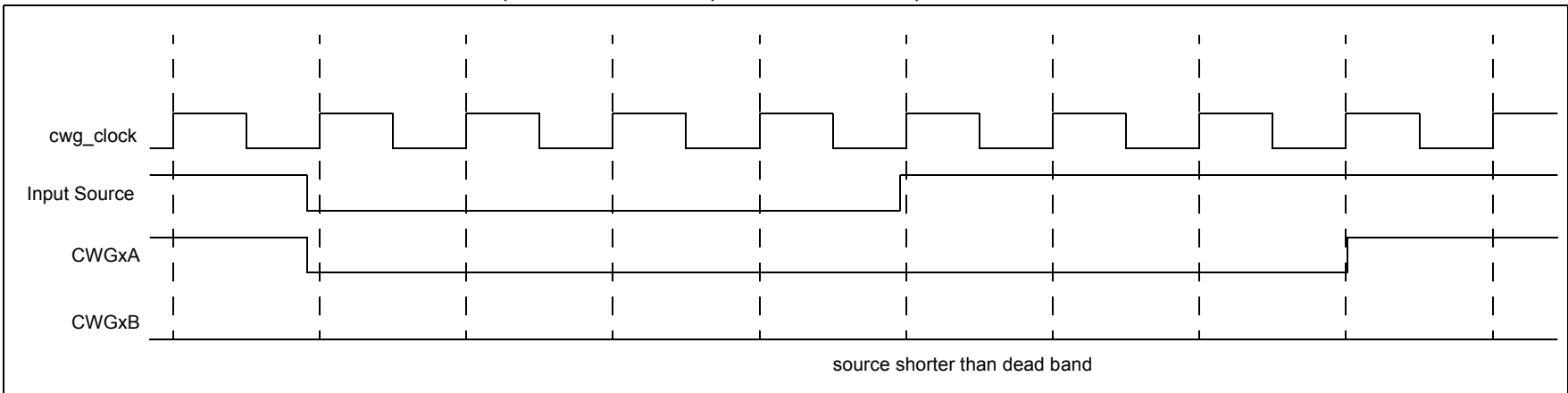
If the input source signal is not present for enough time for the count to be completed, no output will be seen on the respective output.

Refer to [Figure 22-3](#) and [Figure 22-4](#) for examples.

**FIGURE 22-3: DEAD-BAND OPERATION, CWGxDBR = 01H, CWGxDBF = 02H**



**FIGURE 22-4: DEAD-BAND OPERATION, CWGxDBR = 03H, CWGxDBF = 04H, SOURCE SHORTER THAN DEAD BAND**



## 22.8 Dead-Band Uncertainty

When the rising and falling edges of the input source triggers the dead-band counters, the input may be asynchronous. This will create some uncertainty in the dead-band time delay. The maximum uncertainty is equal to one CWG clock period. Refer to [Equation 22-1](#) for more detail.

### EQUATION 22-1: DEAD-BAND UNCERTAINTY

$$T_{DEADBAND\_UNCERTAINTY} = \frac{1}{F_{cwg\_clock}}$$

Example:

$$F_{cwg\_clock} = 16 \text{ MHz}$$

Therefore:

$$\begin{aligned} T_{DEADBAND\_UNCERTAINTY} &= \frac{1}{F_{cwg\_clock}} \\ &= \frac{1}{16 \text{ MHz}} \\ &= 62.5 \text{ ns} \end{aligned}$$

## 22.9 Auto-Shutdown Control

Auto-shutdown is a method to immediately override the CWG output levels with specific overrides that allow for safe shutdown of the circuit. The shutdown state can be either cleared automatically or held until cleared by software.

### 22.9.1 SHUTDOWN

The shutdown state can be entered by either of the following two methods:

- Software generated
- External Input

#### 22.9.1.1 Software Generated Shutdown

Setting the GxASE bit of the CWGxCON2 register will force the CWG into the shutdown state.

When auto-restart is disabled, the shutdown state will persist as long as the GxASE bit is set.

When auto-restart is enabled, the GxASE bit will clear automatically and resume operation on the next rising edge event. See [Figure 22-6](#).

#### 22.9.1.2 External Input Source

External shutdown inputs provide the fastest way to safely suspend CWG operation in the event of a Fault condition. When any of the selected shutdown inputs goes active, the CWG outputs will immediately go to the selected override levels without software delay. Any combination of two input sources can be selected to cause a shutdown condition. The sources are:

- CLC2 – LC2\_out
- CWG1FLT

Shutdown inputs are selected in the CWGxCON2 register. ([Register 22-3](#)).

**Note:** Shutdown inputs are level sensitive, not edge sensitive. The shutdown state cannot be cleared, except by disabling auto-shutdown, as long as the shutdown input level persists.

## 22.10 Operation During Sleep

The CWG module operates independently from the system clock and will continue to run during Sleep, provided that the clock and input sources selected remain active.

The HFINTOSC remains active during Sleep, provided that the CWG module is enabled, the input source is active, and the HFINTOSC is selected as the clock source, regardless of the system clock source selected.

In other words, if the HFINTOSC is simultaneously selected as the system clock and the CWG clock source, when the CWG is enabled and the input source is active, the CPU will go idle during Sleep, but the CWG will continue to operate and the HFINTOSC will remain active.

This will have a direct effect on the Sleep mode current.

## 22.11 Configuring the CWG

The following steps illustrate how to properly configure the CWG to ensure a synchronous start:

1. Ensure that the TRIS control bits corresponding to CWGxA and CWGxB are set so that both are configured as inputs.
2. Clear the GxEN bit, if not already cleared.
3. Set desired dead-band times with the CWGxDBR and CWGxDBF registers.
4. Setup the following controls in CWGxCON2 auto-shutdown register:
  - Select desired shutdown source.
  - Select both output overrides to the desired levels (this is necessary even if not using auto-shutdown because start-up will be from a shutdown state).
  - Set the GxASE bit and clear the GxARSEN bit.
5. Select the desired input source using the CWGxCON1 register.
6. Configure the following controls in CWGxCON0 register:
  - Select desired clock source.
  - Select the desired output polarities.
  - Set the output enables for the outputs to be used.
7. Set the GxEN bit.
8. Clear TRIS control bits corresponding to CWGxA and CWGxB to be used to configure those pins as outputs.
9. If auto-restart is to be used, set the GxARSEN bit and the GxASE bit will be cleared automatically. Otherwise, clear the GxASE bit to start the CWG.

### 22.11.1 PIN OVERRIDE LEVELS

The levels driven to the output pins, while the shutdown input is true, are controlled by the GxASDLA and GxASDLB bits of the CWGxCON1 register ([Register 22-3](#)). GxASDLA controls the CWG1A override level and GxASDLB controls the CWG1B override level. The control bit logic level corresponds to the output logic drive level while in the shutdown state. The polarity control does not apply to the override level.

### 22.11.2 AUTO-SHUTDOWN RESTART

After an auto-shutdown event has occurred, there are two ways to have resume operation:

- Software controlled
- Auto-restart

The restart method is selected with the GxARSEN bit of the CWGxCON2 register. Waveforms of software controlled and automatic restarts are shown in [Figure 22-5](#) and [Figure 22-6](#).

#### 22.11.2.1 Software Controlled Restart

When the GxARSEN bit of the CWGxCON2 register is cleared, the CWG must be restarted after an auto-shutdown event by software.

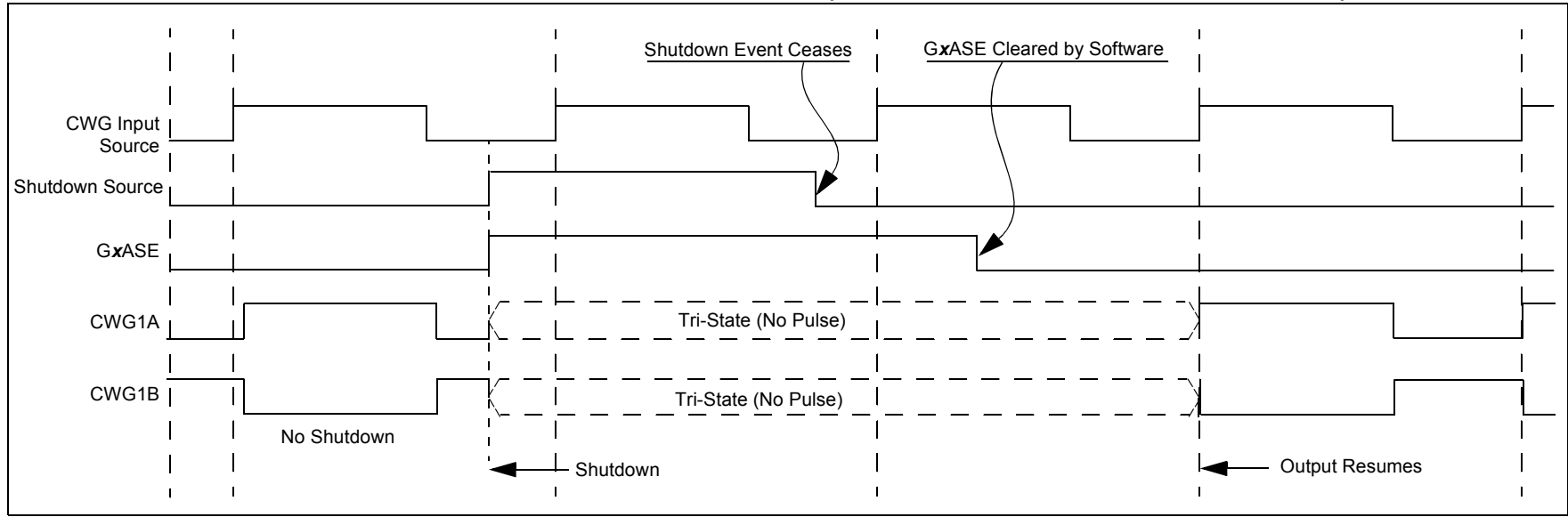
Clearing the shutdown state requires all selected shutdown inputs to be low, otherwise the GxASE bit will remain set. The overrides will remain in effect until the first rising edge event after the GxASE bit is cleared. The CWG will then resume operation.

#### 22.11.2.2 Auto-Restart

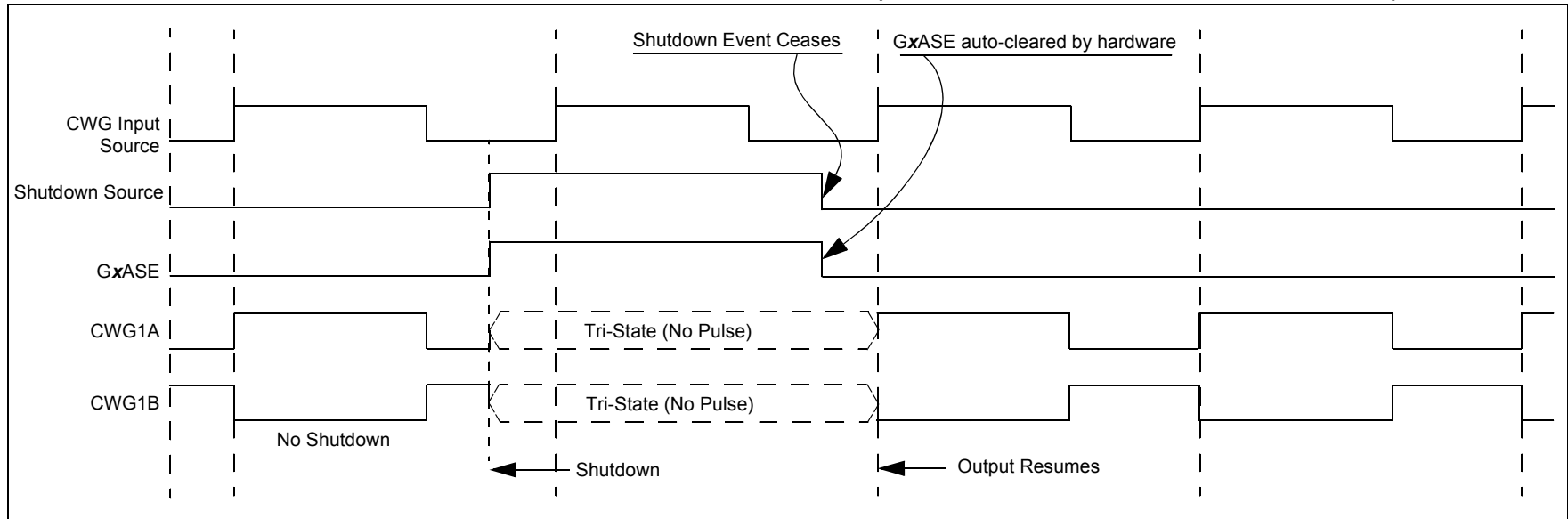
When the GxARSEN bit of the CWGxCON2 register is set, the CWG will restart from the auto-shutdown state automatically.

The GxASE bit will clear automatically when all shutdown sources go low. The overrides will remain in effect until the first rising edge event after the GxASE bit is cleared. The CWG will then resume operation.

**FIGURE 22-5: SHUTDOWN FUNCTIONALITY, AUTO-RESTART DISABLED ( $GxARSEN = 0, GxASDLA = 01, GxASDLB = 01$ )**



**FIGURE 22-6: SHUTDOWN FUNCTIONALITY, AUTO-RESTART ENABLED ( $GxARSEN = 1, GxASDLA = 01, GxASDLB = 01$ )**



# PIC16(L)F1507

## 22.12 Register Definitions: CWG Control

### REGISTER 22-1: CWGxCON0: CWG CONTROL REGISTER 0

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	U-0	R/W-0/0
GxEN	GxOEB	GxOEA	GxPOLB	GxPOLA	—	—	GxCS0
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      **GxEN:** CWGx Enable bit  
1 = Module is enabled  
0 = Module is disabled
- bit 6      **GxOEB:** CWGxB Output Enable bit  
1 = CWGxB is available on appropriate I/O pin  
0 = CWGxB is not available on appropriate I/O pin
- bit 5      **GxOEA:** CWGxA Output Enable bit  
1 = CWGxA is available on appropriate I/O pin  
0 = CWGxA is not available on appropriate I/O pin
- bit 4      **GxPOLB:** CWGxB Output Polarity bit  
1 = Output is inverted polarity  
0 = Output is normal polarity
- bit 3      **GxPOLA:** CWGxA Output Polarity bit  
1 = Output is inverted polarity  
0 = Output is normal polarity
- bit 2-1    **Unimplemented:** Read as '0'
- bit 0      **GxCS0:** CWGx Clock Source Select bit  
1 = HFINTOSC  
0 = Fosc



## REGISTER 22-2: CWGxCON1: CWG CONTROL REGISTER 1

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	U-0	R/W-0/0	R/W-0/0	R/W-0/0
GxASDLB<1:0>		GxASDLA<1:0>		—	GxIS<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	q = Value depends on condition

- bit 7-6      **GxASDLB<1:0>**: CWGx Shutdown State for CWGxB  
 When an auto shutdown event is present (GxASE = 1):  
 11 = CWGxB pin is driven to '1', regardless of the setting of the GxPOLB bit.  
 10 = CWGxB pin is driven to '0', regardless of the setting of the GxPOLB bit.  
 01 = CWGxB pin is tri-stated  
 00 = CWGxB pin is driven to its inactive state after the selected dead-band interval. GxPOLB still will control the polarity of the output.
- bit 5-4      **GxASDLA<1:0>**: CWGx Shutdown State for CWGxA  
 When an auto shutdown event is present (GxASE = 1):  
 11 = CWGxA pin is driven to '1', regardless of the setting of the GxPOLA bit.  
 10 = CWGxA pin is driven to '0', regardless of the setting of the GxPOLA bit.  
 01 = CWGxA pin is tri-stated  
 00 = CWGxA pin is driven to its inactive state after the selected dead-band interval. GxPOLA still will control the polarity of the output.
- bit 3      **Unimplemented**: Read as '0'
- bit 2-0      **GxIS<2:0>**: CWGx Input Source Select bits  
 111 = CLC1 – LC1\_out  
 110 = NCO1 – NCO1\_out  
 101 = PWM4 – PWM4\_out  
 100 = PWM3 – PWM3\_out  
 011 = PWM2 – PWM2\_out  
 010 = PWM1 – PWM1\_out  
 001 = Reserved  
 000 = Reserved

# PIC16(L)F1507

## REGISTER 22-3: CWGxCON2: CWG CONTROL REGISTER 2

R/W-0/0	R/W-0/0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0
GxASE	GxARSEN	—	—	—	—	GxASDSFLT	GxASDSCLC2
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      **GxASE:** Auto-Shutdown Event Status bit  
1 = An auto-shutdown event has occurred  
0 = No auto-shutdown event has occurred
- bit 6      **GxARSEN:** Auto-Restart Enable bit  
1 = Auto-restart is enabled  
0 = Auto-restart is disabled
- bit 5-2    **Unimplemented:** Read as '0'
- bit 1      **GxASDSFLT:** CWG Auto-shutdown on FLT Enable bit  
1 = Shutdown when  $\overline{\text{CWG1FLT}}$  input is low  
0 =  $\overline{\text{CWG1FLT}}$  input has no effect on shutdown
- bit 0      **GxASDSCLC2:** CWG Auto-shutdown on CLC2 Enable bit  
1 = Shutdown when CLC2 output (LC2\_out) is high  
0 = CLC2 output has no effect on shutdown

## REGISTER 22-4: CWGxDBR: COMPLEMENTARY WAVEFORM GENERATOR (CWGx) RISING DEAD-BAND COUNT 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
—	—	CWXDBR<5:0>					
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 7-6      **Unimplemented:** Read as '0'

bit 5-0      **CWXDBR<5:0>:** Complementary Waveform Generator (CWGx) Rising Counts

11 1111 = 63-64 counts of dead band

11 1110 = 62-63 counts of dead band

•  
•  
•

00 0010 = 2-3 counts of dead band

00 0001 = 1-2 counts of dead band

00 0000 = 0 counts of dead band

## REGISTER 22-5: CWGxDBF: COMPLEMENTARY WAVEFORM GENERATOR (CWGx) FALLING DEAD-BAND COUNT 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
—	—	CWXDBF<5:0>					
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 7-6      **Unimplemented:** Read as '0'

bit 5-0      **CWXDBF<5:0>:** Complementary Waveform Generator (CWGx) Falling Counts

11 1111 = 63-64 counts of dead band

11 1110 = 62-63 counts of dead band

•  
•  
•

00 0010 = 2-3 counts of dead band

00 0001 = 1-2 counts of dead band

00 0000 = 0 counts of dead band. Dead-band generation is bypassed.

# PIC16(L)F1507

**TABLE 22-2: SUMMARY OF REGISTERS ASSOCIATED WITH CWG**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
ANSELA	—	—	—	ANSA4	—	ANSA2	ANSA1	ANSA0	99
CWG1CON0	G1EN	G1OEB	G1OEA	G1POLB	G1POLA	—	—	G1CS0	184
CWG1CON1	G1ASDLB<1:0>		G1ASDLA<1:0>		—	—	G1IS<1:0>		185
CWG1CON2	G1ASE	G1ARSEN	—	—	—	—	G1ASDSFLT	G1ASDSCLC2	186
CWG1DBF	—	—	CWG1DBF<5:0>						187
CWG1DBR	—	—	CWG1DBR<5:0>						187
TRISA	—	—	TRISA5	TRISA4	— <sup>(1)</sup>	TRISA2	TRISA1	TRISA0	98
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	106

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

**Note 1:** Unimplemented, read as '1'.

## 23.0 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
- $\overline{\text{MCLR}}/\text{VPP}$
- 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 “PIC12(L)F1501/PIC16(L)F150X Memory Programming Specification” (DS41573).

### 23.1 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  $\overline{\text{MCLR}}/\text{VPP}$  to  $V_{\text{IH}}$ .

### 23.2 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 Words is set to ‘1’, the ICSP Low-Voltage Programming Entry mode 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  $V_{\text{IL}}$ .
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  $V_{\text{IL}}$  for as long as Program/Verify mode is to be maintained.

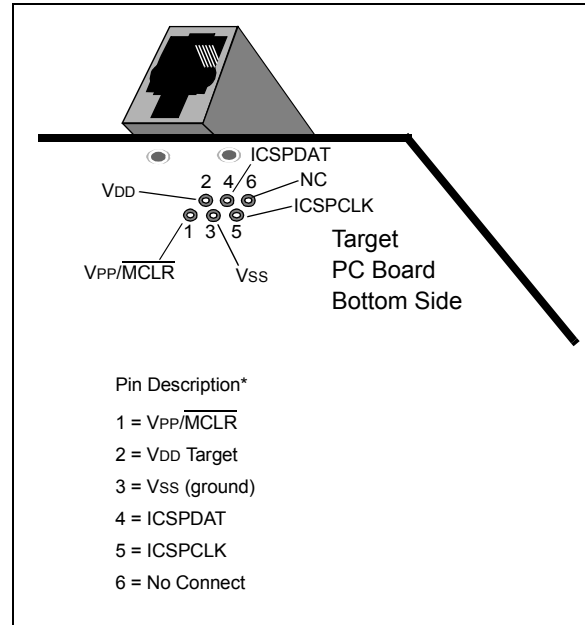
If low-voltage programming is enabled ( $\text{LVP} = 1$ ), the  $\overline{\text{MCLR}}$  Reset function is automatically enabled and cannot be disabled. See [Section 6.5 “MCLR”](#) for more information.

The LVP bit can only be reprogrammed to ‘0’ by using the High-Voltage Programming mode.

## 23.3 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-conductor) configuration. See [Figure 23-1](#).

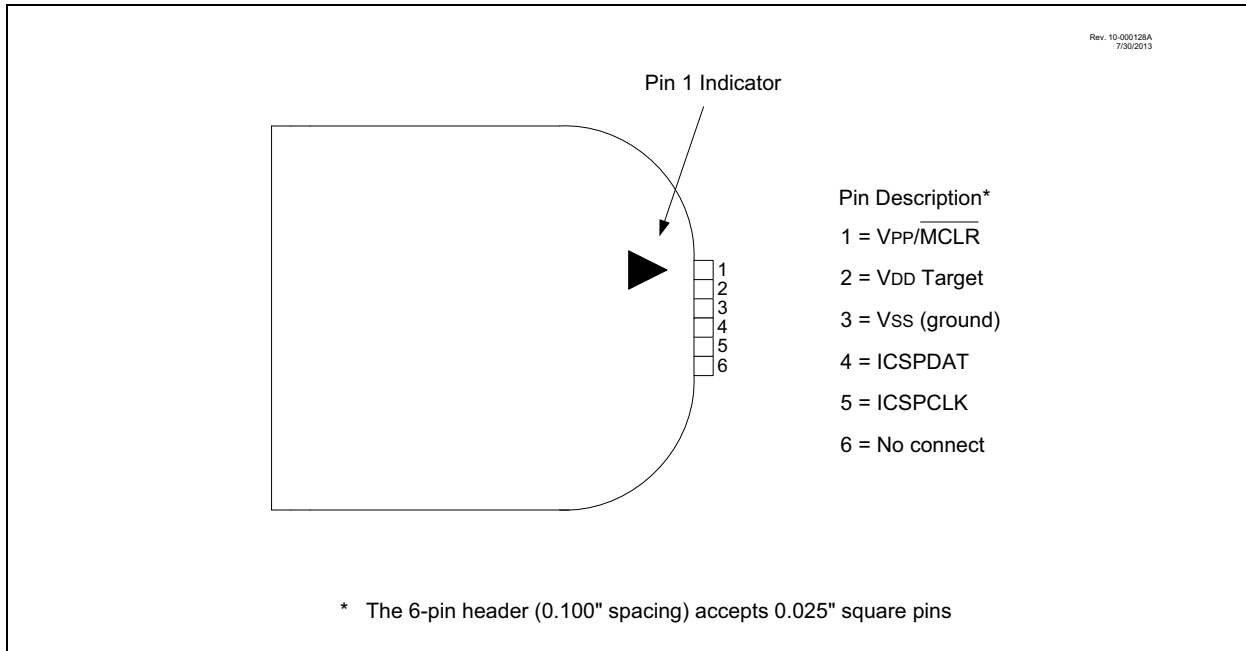
**FIGURE 23-1: ICD RJ-11 STYLE CONNECTOR INTERFACE**



Another connector often found in use with the PICkit™ programmers is a standard 6-pin header with 0.1 inch spacing. Refer to [Figure 23-2](#).

# PIC16(L)F1507

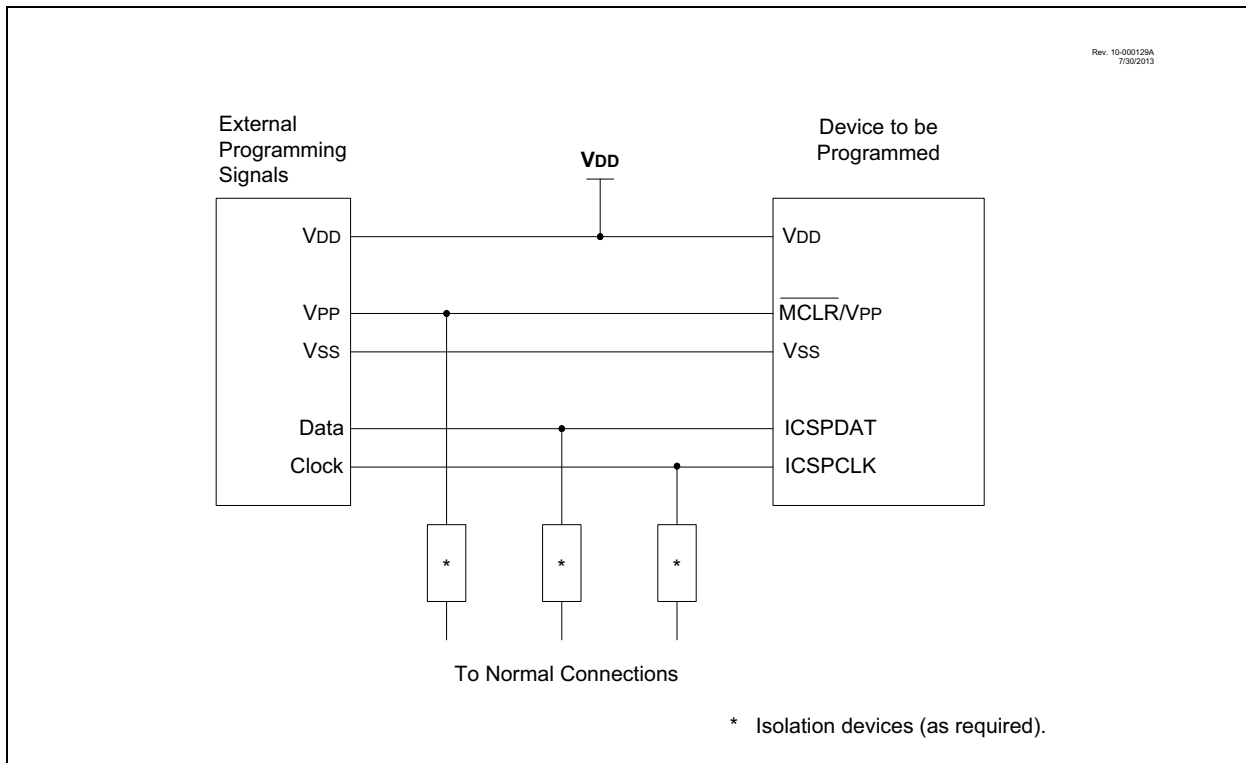
**FIGURE 23-2: PICKIT™ PROGRAMMER STYLE CONNECTOR INTERFACE**



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 23-3](#) for more information.

**FIGURE 23-3: TYPICAL CONNECTION FOR ICSP™ PROGRAMMING**



## 24.0 INSTRUCTION SET SUMMARY

Each 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 24-3 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 4 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.

## 24.1 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 24-1: OPCODE FIELD DESCRIPTIONS**

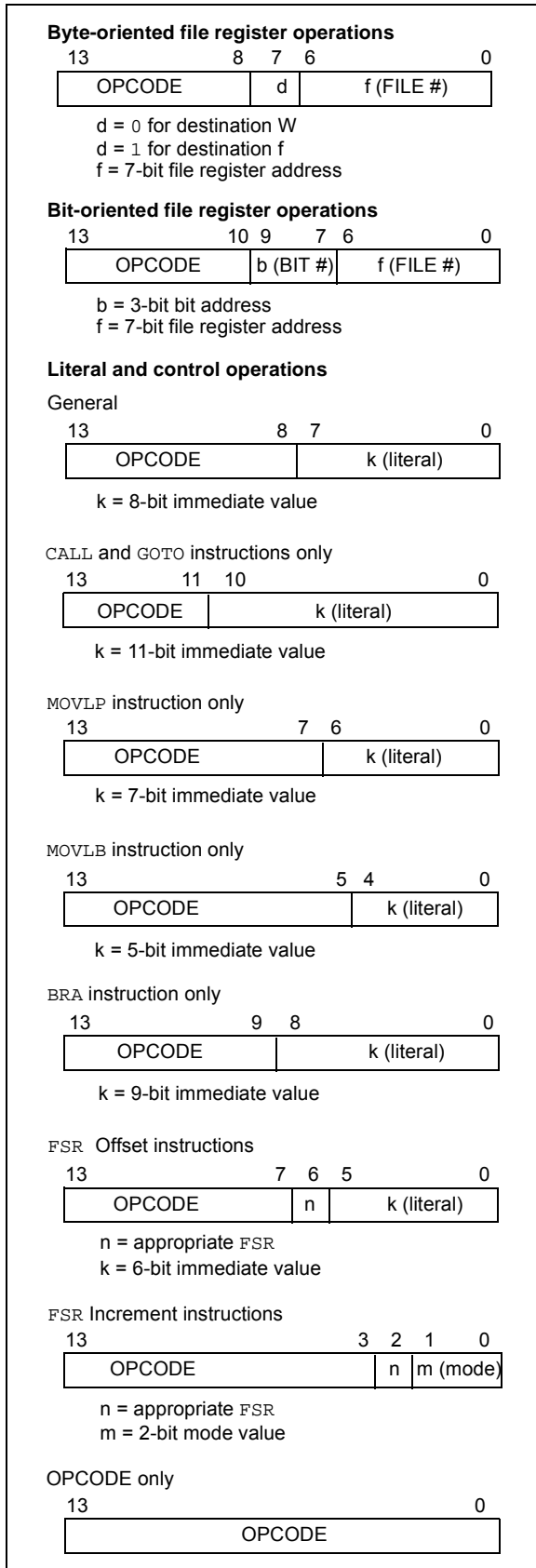
Field	Description
f	Register file address (0x00 to 0x7F)
W	Working register (accumulator)
b	Bit address within an 8-bit file register
k	Literal field, constant data or label
x	Don't care location (= 0 or 1). The assembler will generate code with x = 0. It is the recommended form of use for compatibility 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 24-2: ABBREVIATION DESCRIPTIONS**

Field	Description
PC	Program Counter
$\overline{TO}$	Time-Out bit
C	Carry bit
DC	Digit Carry bit
Z	Zero bit
$\overline{PD}$	Power-Down bit

# PIC16(L)F1507

**FIGURE 24-1: GENERAL FORMAT FOR INSTRUCTIONS**





**TABLE 24-3: ENHANCED MID-RANGE INSTRUCTION SET**

Mnemonic, Operands	Description	Cycles	14-Bit Opcode				Status Affected	Notes	
			MSb	LSb					
<b>BYTE-ORIENTED FILE REGISTER OPERATIONS</b>									
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	1fff	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
<b>BYTE ORIENTED SKIP OPERATIONS</b>									
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
<b>BIT-ORIENTED FILE REGISTER OPERATIONS</b>									
BCF	f, b	Bit Clear f	1	01	00bb	bfff	ffff		2
BSF	f, b	Bit Set f	1	01	01bb	bfff	ffff		2
<b>BIT-ORIENTED SKIP OPERATIONS</b>									
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
<b>LITERAL OPERATIONS</b>									
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.

# PIC16(L)F1507

**TABLE 24-3: ENHANCED MID-RANGE INSTRUCTION SET (CONTINUED)**

Mnemonic, Operands	Description	Cycles	14-Bit Opcode			Status Affected	Notes		
			MSb		LSb				
<b>CONTROL OPERATIONS</b>									
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		
<b>INHERENT OPERATIONS</b>									
CLRWDT	–	Clear Watchdog Timer	1	00	0000	0110	0100	$\overline{TO}$ , $\overline{PD}$	
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	$\overline{TO}$ , $\overline{PD}$	
TRIS	f	Load TRIS register with W	1	00	0000	0110	0fff		
<b>C-COMPILER OPTIMIZED</b>									
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 kkkk	Z	<b>2, 3</b>
	k[n]	Move INDFn to W, Indexed Indirect.	1	11	1111	0nkk	1nmm	Z	<b>2</b>
MOVWI	n mm	Move W to Indirect FSRn with pre/post inc/dec modifier, mm	1	00	0000	0001	kkkk		<b>2, 3</b>
	k[n]	Move W to INDFn, Indexed Indirect.	1	11	1111	1nkk			<b>2</b>

**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 *NOPE*.

**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.

## 24.2 Instruction Descriptions

### ADDFSR Add Literal to FSRn

**Syntax:** [ *label* ] ADDFSR FSRn, k

**Operands:**  $-32 \leq k \leq 31$   
 $n \in [0, 1]$

**Operation:**  $FSR(n) + k \rightarrow FSR(n)$

**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:**  $0 \leq k \leq 255$

**Operation:**  $(W) + k \rightarrow (W)$

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

**Operation:**  $(W) + (f) \rightarrow (\text{destination})$

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

**Operation:**  $(W) + (f) + (C) \rightarrow \text{dest}$

**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:**  $0 \leq k \leq 255$

**Operation:**  $(W) .AND. (k) \rightarrow (W)$

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

**Operation:**  $(W) .AND. (f) \rightarrow (\text{destination})$

**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

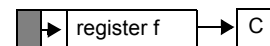
**Syntax:** [ *label* ] ASRF f {,d}

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

**Operation:**  $(f < 7 >) \rightarrow \text{dest} < 7 >$   
 $(f < 7 : 1 >) \rightarrow \text{dest} < 6 : 0 >$ ,  
 $(f < 0 >) \rightarrow C$ ,

**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'.



# PIC16(L)F1507

---

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

---

Syntax:                [ *label* ] BCF   f,b

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

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

Status Affected:    None

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

**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.

**BRA**                    **Relative Branch**

---

Syntax:                [ *label* ] BRA   label  
                         [ *label* ] BRA   \$+k

Operands:             $-256 \leq \text{label} - \text{PC} + 1 \leq 255$   
                          $-256 \leq k \leq 255$

Operation:             $(\text{PC}) + 1 + k \rightarrow \text{PC}$

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  $\text{PC} + 1 + k$ . This instruction is a 2-cycle instruction. This branch has a limited range.

**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.

**BRW**                    **Relative Branch with W**

---

Syntax:                [ *label* ] BRW

Operands:            None

Operation:             $(\text{PC}) + (W) \rightarrow \text{PC}$

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  $\text{PC} + 1 + (W)$ . 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:             $1 \rightarrow (f<b>)$

Status Affected:    None

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

<b>CALL</b>	<b>Call Subroutine</b>
Syntax:	[ <i>label</i> ] CALL k
Operands:	$0 \leq k \leq 2047$
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.

<b>CLRWDT</b>	<b>Clear Watchdog Timer</b>
Syntax:	[ <i>label</i> ] 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.

<b>CALLW</b>	<b>Subroutine Call With W</b>
Syntax:	[ <i>label</i> ] 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.

<b>COMF</b>	<b>Complement f</b>
Syntax:	[ <i>label</i> ] COMF f,d
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$
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'.

<b>CLRF</b>	<b>Clear f</b>
Syntax:	[ <i>label</i> ] CLRF f
Operands:	$0 \leq f \leq 127$
Operation:	00h → (f) 1 → Z
Status Affected:	Z
Description:	The contents of register 'f' are cleared and the Z bit is set.

<b>DECF</b>	<b>Decrement f</b>
Syntax:	[ <i>label</i> ] DECF f,d
Operands:	$0 \leq f \leq 127$ $d \in [0,1]$
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'.

<b>CLRW</b>	<b>Clear W</b>
Syntax:	[ <i>label</i> ] CLRW
Operands:	None
Operation:	00h → (W) 1 → Z
Status Affected:	Z
Description:	W register is cleared. Zero bit (Z) is set.

# PIC16(L)F1507

---

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

---

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

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

**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:**       $0 \leq k \leq 2047$

**Operation:**       $k \rightarrow PC<10:0>$   
                     $PCLATH<6:3> \rightarrow PC<14:11>$

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

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

**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:**       $0 \leq k \leq 255$

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

**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**

---

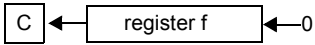
Syntax:                    `[label] LSLF f{,d}`

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

Operation:                 $(f<7>) \rightarrow C$   
 $(f<6:0>) \rightarrow \text{dest}<7:1>$   
 $0 \rightarrow \text{dest}<0>$

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**

---

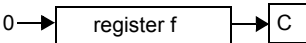
Syntax:                    `[label] LSRF f{,d}`

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

Operation:                 $0 \rightarrow \text{dest}<7>$   
 $(f<7:1>) \rightarrow \text{dest}<6:0>$ ,  
 $(f<0>) \rightarrow C$ ,

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

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

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

# PIC16(L)F1507

## MOVIW Move INDFn to W

Syntax: [ *label* ] MOVIW ++FSRn  
 [ *label* ] MOVIW --FSRn  
 [ *label* ] MOVIW FSRn++  
 [ *label* ] MOVIW FSRn--  
 [ *label* ] MOVIW k[FSRn]

Operands:  $n \in [0,1]$   
 $mm \in [00,01, 10, 11]$   
 $-32 \leq k \leq 31$

Operation: INDFn  $\rightarrow$  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:  $0 \leq k \leq 31$

Operation:  $k \rightarrow$  BSR

Status Affected: None

Description: The 5-bit literal 'k' is loaded into the Bank Select Register (BSR).

## MOVLPL Move literal to PCLATH

Syntax: [ *label* ] MOVLPL k

Operands:  $0 \leq k \leq 127$

Operation:  $k \rightarrow$  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:  $0 \leq k \leq 255$

Operation:  $k \rightarrow$  (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

Example: MOVLW 0x5A  
 After Instruction  
 W = 0x5A

## MOVWF Move W to f

Syntax: [ *label* ] MOVWF f

Operands:  $0 \leq f \leq 127$

Operation: (W)  $\rightarrow$  (f)

Status Affected: None

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

Words: 1

Cycles: 1

Example: 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 ≤ k ≤ 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

Example:                                  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.

**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.

# PIC16(L)F1507

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

---

Syntax:            [ *label* ] RETFIE

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

**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.

**RETLW**      **Return with literal in W**

---

Syntax:            [ *label* ] RETLW *k*

Operands:         $0 \leq k \leq 255$

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*

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

---

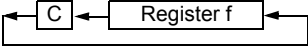
Syntax:            [ *label* ] RLF *f*,*d*

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

Operation:        See description below

Status Affected:   C

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

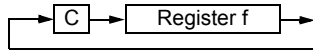
**Syntax:** [ *label* ] RRF f,d

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

**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:**  $0 \leq k \leq 255$

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

**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	$W \leq k$
DC = 0	$W\langle 3:0 \rangle > k\langle 3:0 \rangle$
DC = 1	$W\langle 3:0 \rangle \leq k\langle 3:0 \rangle$

## SUBWF Subtract W from f

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

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

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

**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	$W \leq f$
DC = 0	$W\langle 3:0 \rangle > f\langle 3:0 \rangle$
DC = 1	$W\langle 3:0 \rangle \leq f\langle 3:0 \rangle$

## SUBWFB Subtract W from f with Borrow

**Syntax:** SUBWFB f {,d}

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

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

**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'.

# PIC16(L)F1507

---

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

---

Syntax:      [ *label* ] SWAPF f,d

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

Operation:    ( $f<3:0>$ )  $\rightarrow$  ( $\text{destination}<7:4>$ ),  
              ( $f<7:4>$ )  $\rightarrow$  ( $\text{destination}<3:0>$ )

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'.

## **XORLW**      **Exclusive OR literal with W**

---

Syntax:      [ *label* ] XORLW k

Operands:     $0 \leq k \leq 255$

Operation:    (W) .XOR. k  $\rightarrow$  (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.

## **TRIS**      **Load TRIS Register with W**

---

Syntax:      [ *label* ] TRIS f

Operands:     $5 \leq f \leq 7$

Operation:    (W)  $\rightarrow$  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.

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

---

Syntax:      [ *label* ] XORWF f,d

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

Operation:    (W) .XOR. (f)  $\rightarrow$  (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'.

## 25.0 ELECTRICAL SPECIFICATIONS

### 25.1 Absolute Maximum Ratings<sup>(†)</sup>

Ambient temperature under bias .....	-40°C to +125°C
Storage temperature .....	-65°C to +150°C
Voltage on pins with respect to V <sub>SS</sub>	
on V <sub>DD</sub> pin	
PIC16F1507 .....	-0.3V to +6.5V
PIC16LF1507 .....	-0.3V to +4.0V
on $\overline{\text{MCLR}}$ pin .....	-0.3V to +9.0V
on all other pins .....	-0.3V to (V <sub>DD</sub> + 0.3V)
Maximum current	
on V <sub>SS</sub> pin <sup>(1)</sup>	
-40°C ≤ T <sub>A</sub> ≤ +85°C .....	250 mA
+85°C ≤ T <sub>A</sub> ≤ +125°C .....	85 mA
on V <sub>DD</sub> pin <sup>(1)</sup>	
-40°C ≤ T <sub>A</sub> ≤ +85°C .....	250 mA
+85°C ≤ T <sub>A</sub> ≤ +125°C .....	85 mA
Sunk by any standard I/O pin .....	50 mA
Sourced by any standard I/O pin .....	50 mA
Clamp current, I <sub>K</sub> (V <sub>PIN</sub> < 0 or V <sub>PIN</sub> > V <sub>DD</sub> ) .....	±20 mA
Total power dissipation <sup>(2)</sup> .....	800 mW

**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 25-6](#) to calculate device specifications.

**2:** 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.

# PIC16(L)F1507

## 25.2 Standard Operating Conditions

The standard operating conditions for any device are defined as:

Operating Voltage:  $V_{DDMIN} \leq V_{DD} \leq V_{DDMAX}$

Operating Temperature:  $T_{A\_MIN} \leq T_A \leq T_{A\_MAX}$

### V<sub>DD</sub> — Operating Supply Voltage<sup>(1)</sup>

#### PIC16LF1507

V<sub>DDMIN</sub> (F<sub>osc</sub> ≤ 16 MHz)..... +1.8V

V<sub>DDMIN</sub> (16 MHz < F<sub>osc</sub> ≤ 20 MHz) ..... +2.5V

V<sub>DDMAX</sub> ..... +3.6V

#### PIC16F1507

V<sub>DDMIN</sub> (F<sub>osc</sub> ≤ 16 MHz)..... +2.3V

V<sub>DDMIN</sub> (16 MHz < F<sub>osc</sub> ≤ 20 MHz) ..... +2.5V

V<sub>DDMAX</sub> ..... +5.5V

### T<sub>A</sub> — Operating Ambient Temperature Range

#### Industrial Temperature

T<sub>A\\_MIN</sub> ..... -40°C

T<sub>A\\_MAX</sub> ..... +85°C

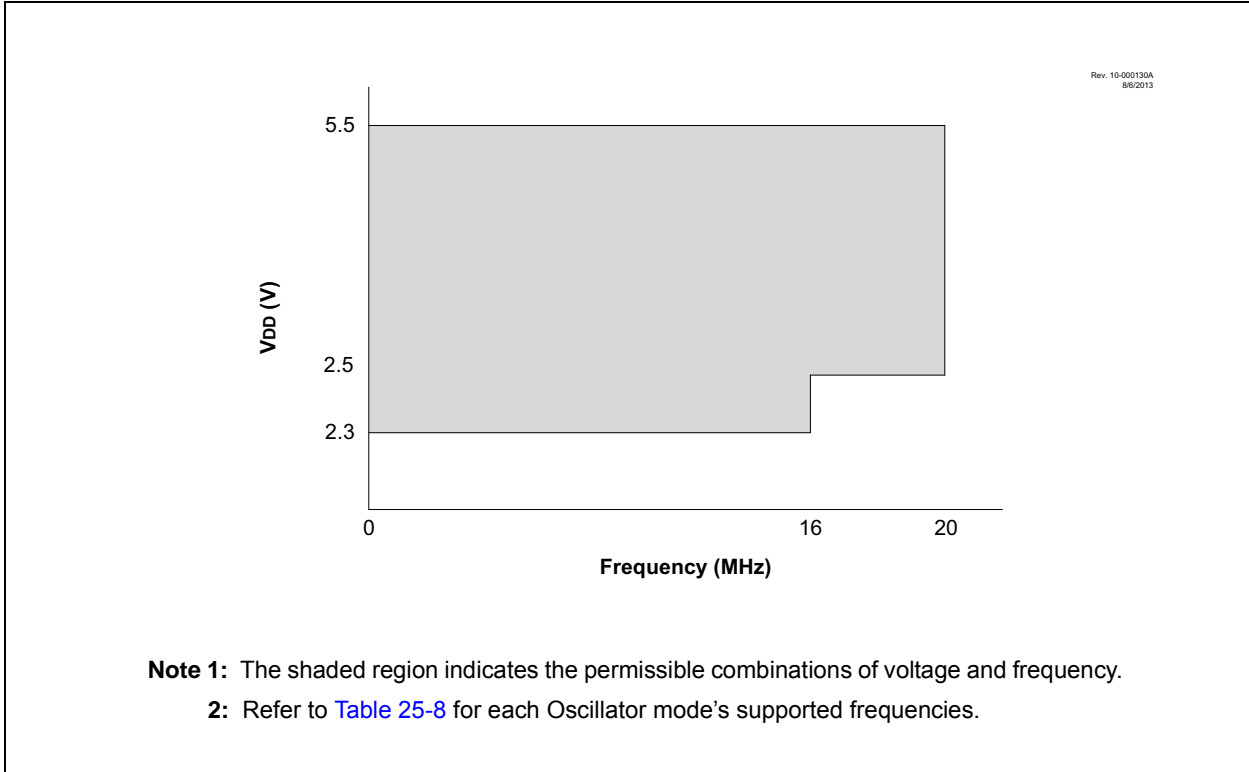
#### Extended Temperature

T<sub>A\\_MIN</sub> ..... -40°C

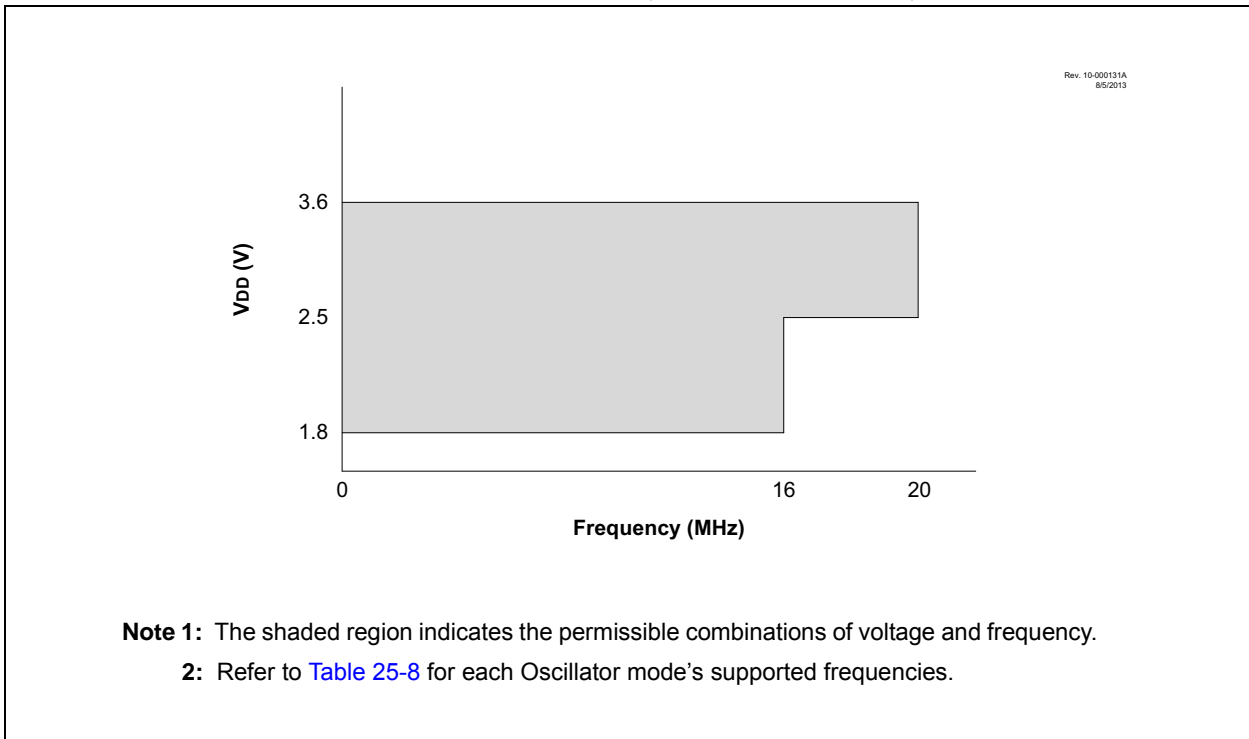
T<sub>A\\_MAX</sub> ..... +125°C

**Note 1:** See Parameter [D001](#), DC Characteristics: Supply Voltage.

**FIGURE 25-1: VOLTAGE FREQUENCY GRAPH,  $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ , PIC16F1507 ONLY**



**FIGURE 25-2: VOLTAGE FREQUENCY GRAPH,  $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ , PIC16LF1507 ONLY**



# PIC16(L)F1507

## 25.3 DC Characteristics

TABLE 25-1: SUPPLY VOLTAGE

PIC16LF1507		Standard Operating Conditions (unless otherwise stated)					
PIC16F1507							
Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions
D001	VDD	<b>Supply Voltage</b>					
			VDDMIN 1.8 2.5	— —	VDDMAX 3.6 3.6	V V	FOSC ≤ 16 MHz FOSC ≤ 20 MHz
D001			2.3 2.5	— —	5.5 5.5	V V	FOSC ≤ 16 MHz FOSC ≤ 20 MHz
D002*	VDR	<b>RAM Data Retention Voltage<sup>(1)</sup></b>					
			1.5	—	—	V	Device in Sleep mode
D002*			1.7	—	—	V	Device in Sleep mode
D002A*	VPOR	<b>Power-on Reset Release Voltage<sup>(2)</sup></b>					
			—	1.6	—	V	
D002A*			—	1.6	—	V	
D002B*	VPORR*	<b>Power-on Reset Rearm Voltage<sup>(2)</sup></b>					
			—	0.8	—	V	
D002B*			—	1.5	—	V	
D003	VFVR	<b>Fixed Voltage Reference Voltage</b>					
		1x gain (1.024V nominal)					VDD ≥ 2.5V, -40°C ≤ TA ≤ +85°C
		2x gain (2.048V nominal)	-4	—	+4	%	VDD ≥ 2.5V, -40°C ≤ TA ≤ +85°C
		4x gain (4.096V nominal)	-3	—	+7	%	VDD ≥ 4.75V, -40°C ≤ TA ≤ +85°C
D004*	SVDD	<b>VDD Rise Rate<sup>(2)</sup></b>	0.05	—	—	V/ms	Ensures that the Power-on Reset signal is released properly.

\* These parameters are characterized but not tested.

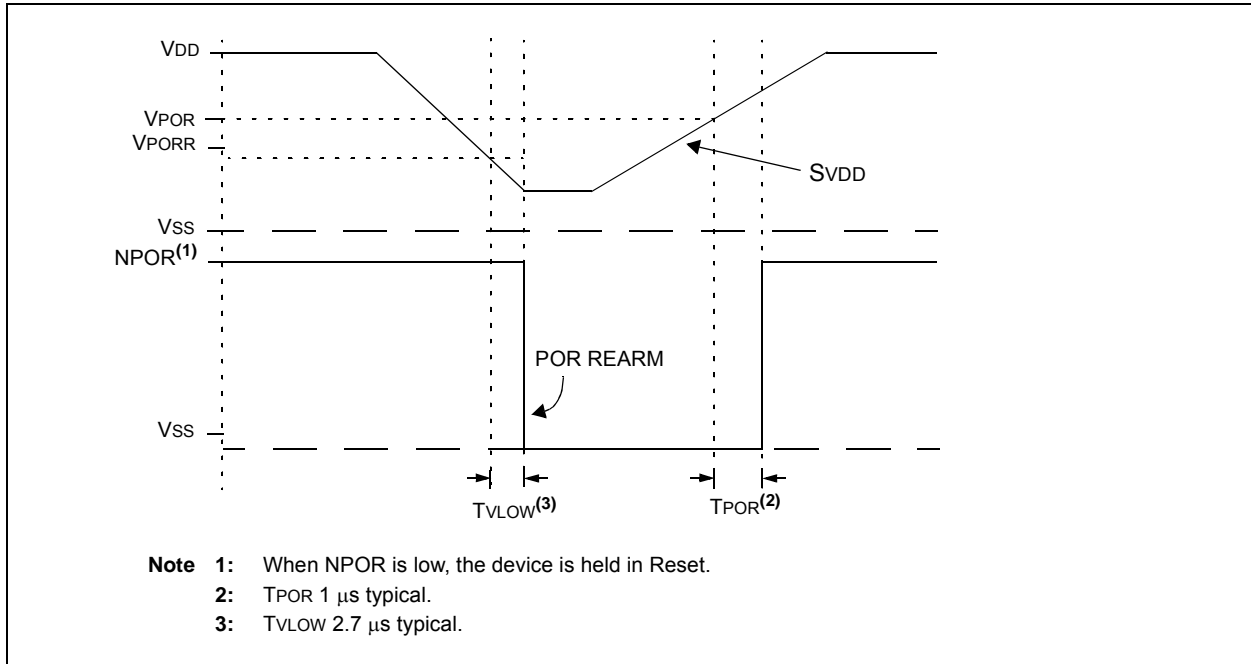
† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

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

**Note 2:** See [Figure 25-3](#), POR and POR REARM with Slow Rising VDD.



**FIGURE 25-3: POR AND POR REARM WITH SLOW RISING V<sub>DD</sub>**



# PIC16(L)F1507

TABLE 25-2: SUPPLY CURRENT (IDD)<sup>(1,2)</sup>

PIC16LF1507		Standard Operating Conditions (unless otherwise stated)					
PIC16F1507							
Param. No.	Device Characteristics	Min.	Typ†	Max.	Units	Conditions	
						VDD	Note
D013		—	30	65	μA	1.8	Fosc = 1 MHz, External Clock (ECM), Medium-Power mode
		—	55	100	μA	3.0	
D013		—	65	110	μA	2.3	Fosc = 1 MHz, External Clock (ECM), Medium-Power mode
		—	85	140	μA	3.0	
		—	115	190	μA	5.0	
D014		—	115	190	μA	1.8	Fosc = 4 MHz, External Clock (ECM), Medium-Power mode
		—	210	310	μA	3.0	
D014		—	180	270	μA	2.3	Fosc = 4 MHz, External Clock (ECM), Medium-Power mode
		—	240	365	μA	3.0	
		—	295	460	μA	5.0	
D015		—	3.2	12	μA	1.8	Fosc = 31 kHz, LFINTOSC, -40°C ≤ Ta ≤ +85°C
		—	5.4	20	μA	3.0	
D015		—	13	28	μA	2.3	Fosc = 31 kHz, LFINTOSC, -40°C ≤ Ta ≤ +85°C
		—	15	30	μA	3.0	
		—	17	36	μA	5.0	
D016		—	215	360	μA	1.8	Fosc = 500 kHz, HFINTOSC
		—	275	480	μA	3.0	
D016		—	270	450	μA	2.3	Fosc = 500 kHz, HFINTOSC
		—	300	500	μA	3.0	
		—	350	620	μA	5.0	
D017*		—	410	660	μA	1.8	Fosc = 8 MHz, HFINTOSC
		—	630	970	μA	3.0	
D017*		—	530	750	μA	2.3	Fosc = 8 MHz, HFINTOSC
		—	660	1100	μA	3.0	
		—	730	1200	μA	5.0	
D018		—	600	940	μA	1.8	Fosc = 16 MHz, HFINTOSC
		—	970	1400	μA	3.0	
D018		—	780	1200	μA	2.3	Fosc = 16 MHz, HFINTOSC
		—	1000	1550	μA	3.0	
		—	1090	1700	μA	5.0	

\* 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 test conditions for all IDD measurements in active operation mode are: OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VSS; 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.

**TABLE 25-2: SUPPLY CURRENT (IDD)<sup>(1,2)</sup> (CONTINUED)**

PIC16LF1507		Standard Operating Conditions (unless otherwise stated)					
PIC16F1507							
Param. No.	Device Characteristics	Min.	Typ†	Max.	Units	Conditions	
						VDD	Note
D019A		—	1030	1500	μA	3.0	Fosc = 20 MHz, External Clock (ECH), High-Power mode
D019A		—	1060	1600	μA	3.0	Fosc = 20 MHz, External Clock (ECH), High-Power mode
		—	1220	1800	μA	5.0	
D019B		—	6	16	μA	1.8	Fosc = 32 kHz, External Clock (ECL), Low-Power mode
		—	8	22	μA	3.0	
D019B		—	13	28	μA	2.3	Fosc = 32 kHz, External Clock (ECL), Low-Power mode
		—	15	31	μA	3.0	
		—	16	36	μA	5.0	
D019C		—	19	35	μA	1.8	Fosc = 500 kHz, External Clock (ECL), Low-Power mode
		—	32	55	μA	3.0	
D019C		—	31	52	μA	2.3	Fosc = 500 kHz, External Clock (ECL), Low-Power mode
		—	38	65	μA	3.0	
		—	44	74	μA	5.0	

\* 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 test conditions for all IDD measurements in active operation mode are: OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VSS; 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.

# PIC16(L)F1507

**TABLE 25-3: POWER-DOWN CURRENTS (I<sub>PD</sub>)<sup>(1,2)</sup>**

PIC16LF1507		Operating Conditions: (unless otherwise stated) Low-Power Sleep Mode						
PIC16F1507		Low-Power Sleep Mode, VREGPM = 1						
Param. No.	Device Characteristics	Min.	Typ†	Max. +85°C	Max. +125°C	Units	Conditions	
							VDD	Note
D022	Base I <sub>PD</sub>	—	0.020	1.0	8.0	μA	1.8	WDT, BOR, FVR and SOSC disabled, all Peripherals inactive
		—	0.025	2.0	9.0	μA	3.0	
D022	Base I <sub>PD</sub>	—	0.25	3.0	10	μA	2.3	WDT, BOR, FVR and SOSC disabled, all Peripherals inactive, Low-Power Sleep mode
		—	0.30	4.0	12	μA	3.0	
		—	0.40	6.0	15	μA	5.0	
D022A	Base I <sub>PD</sub>	—	9.8	16	18	μA	2.3	WDT, BOR, FVR and SOSC disabled, all Peripherals inactive, Normal-Power Sleep mode, VREGPM = 0
		—	10.3	18	20	μA	3.0	
		—	11.5	21	26	μA	5.0	
D023		—	0.26	2.0	9.0	μA	1.8	WDT Current
		—	0.44	3.0	10	μA	3.0	
D023		—	0.43	6.0	15	μA	2.3	WDT Current
		—	0.53	7.0	20	μA	3.0	
		—	0.64	8.0	22	μA	5.0	
D023A		—	15	28	30	μA	1.8	FVR Current
		—	18	30	33	μA	3.0	
D023A		—	18	33	35	μA	2.3	FVR Current
		—	19	35	37	μA	3.0	
		—	20	37	39	μA	5.0	
D024		—	6.0	17	20	μA	3.0	BOR Current
D024		—	7.0	17	30	μA	3.0	BOR Current
		—	8.0	20	40	μA	5.0	
D24A		—	0.1	4.0	10	μA	3.0	LPBOR Current
D24A		—	0.35	5.0	14	μA	3.0	LPBOR Current
		—	0.45	8.0	17	μA	5.0	

\* 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 peripheral Δ current can be determined by subtracting the base I<sub>PD</sub> 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 V<sub>SS</sub>.
- 3: ADC clock source is FRC.

**TABLE 25-3: POWER-DOWN CURRENTS (I<sub>PD</sub>)<sup>(1,2)</sup> (CONTINUED)**

PIC16LF1507		Operating Conditions: (unless otherwise stated) Low-Power Sleep Mode						
PIC16F1507		Low-Power Sleep Mode, VREGPM = 1						
Param. No.	Device Characteristics	Min.	Typ†	Max. +85°C	Max. +125°C	Units	Conditions	
							V <sub>DD</sub>	Note
D026		—	0.11	1.5	9.0	μA	1.8	ADC Current ( <b>Note 3</b> ), No conversion in progress
		—	0.12	2.7	10	μA	3.0	
D026		—	0.30	4.0	11	μA	2.3	ADC Current ( <b>Note 3</b> ), No conversion in progress
		—	0.35	5.0	13	μA	3.0	
		—	0.45	8.0	16	μA	5.0	
D026A*		—	250	—	—	μA	1.8	ADC Current ( <b>Note 3</b> ), Conversion in progress
		—	250	—	—	μA	3.0	
D026A*		—	280	—	—	μA	2.3	ADC Current ( <b>Note 3</b> ), Conversion in progress
		—	280	—	—	μA	3.0	
		—	280	—	—	μA	5.0	

\* 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 peripheral Δ current can be determined by subtracting the base I<sub>PD</sub> current from this limit. Max. values should be used when calculating total current consumption.
- Note 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 V<sub>SS</sub>.
- Note 3:** ADC clock source is FRC.

# PIC16(L)F1507

**TABLE 25-4: I/O PORTS**

Standard Operating Conditions (unless otherwise stated)							
Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions
D030 D030A D032	VIL	<b>Input Low Voltage</b>					
		I/O PORT: with TTL buffer	—	—	0.8	V	$4.5V \leq V_{DD} \leq 5.5V$
		MCLR	—	—	$0.15 V_{DD}$	V	$1.8V \leq V_{DD} \leq 4.5V$
D040 D040A D042	VIH	<b>Input High Voltage</b>					
		I/O PORT: with TTL buffer	2.0	—	—	V	$4.5V \leq V_{DD} \leq 5.5V$
		MCLR	$0.25 V_{DD} + 0.8$	—	—	V	$1.8V \leq V_{DD} \leq 4.5V$
D060  D061	IIL	<b>Input Leakage Current<sup>(1)</sup></b>					
		I/O Ports	—	$\pm 5$	$\pm 125$	nA	$V_{SS} \leq V_{PIN} \leq V_{DD}$ , Pin at high-impedance, 85°C
		MCLR <sup>(2)</sup>	—	$\pm 50$	$\pm 200$	nA	$V_{SS} \leq V_{PIN} \leq V_{DD}$ , Pin at high-impedance, 85°C
D070*	IPUR	<b>Weak Pull-up Current</b>					
			25	100	200	μA	$V_{DD} = 3.3V, V_{PIN} = V_{SS}$
			25	140	300	μA	$V_{DD} = 5.0V, V_{PIN} = V_{SS}$
D080	VOL	<b>Output Low Voltage</b>					
		I/O Ports	—	—	0.6	V	$I_{OL} = 8 \text{ mA}, V_{DD} = 5V$ $I_{OL} = 6 \text{ mA}, V_{DD} = 3.3V$ $I_{OL} = 1.8 \text{ mA}, V_{DD} = 1.8V$
D090	VOH	<b>Output High Voltage</b>					
		I/O Ports	$V_{DD} - 0.7$	—	—	V	$I_{OH} = 3.5 \text{ mA}, V_{DD} = 5V$ $I_{OH} = 3 \text{ mA}, V_{DD} = 3.3V$ $I_{OH} = 1 \text{ mA}, V_{DD} = 1.8V$
D101A*	CIO	<b>Capacitive Loading Specifications on Output Pins</b>					
		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.

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

**TABLE 25-5: MEMORY PROGRAMMING SPECIFICATIONS**

Standard Operating Conditions (unless otherwise stated)							
Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions
<b>Program Memory Programming Specifications</b>							
D110	VIHH	Voltage on $\overline{\text{MCLR}}/\text{VPP}$ pin	8.0	—	9.0	V	(Note 2)
D112	VPBE	VDD for Bulk Erase	2.7	—	VDDMAX	V	
D113	VPEW	VDD for Write or Row Erase	VDDMIN	—	VDDMAX	V	
D114	IPPPGM	Current on $\overline{\text{MCLR}}/\text{VPP}$ during Erase/Write	—	1.0	—	mA	
D115	IDDPGM	Current on VDD during Erase/Write	—	5.0	—	mA	
<b>Program Flash Memory</b>							
D121	EP	Cell Endurance	10K	—	—	E/W	-40°C ≤ TA ≤ +85°C (Note 1)
D122	VPRW	VDD for Read/Write	VDDMIN	—	VDDMAX	V	
D123	TIW	Self-timed Write Cycle Time	—	2	2.5	ms	Provided no other specifications are violated
D124	TRETD	Characteristic Retention	—	40	—	Year	
D125	EHEFC	High-Endurance Flash Cell	100K	—	—	E/W	0°C ≤ TA ≤ +60°C, lower byte last 128 addresses

† 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 25-6: THERMAL CHARACTERISTICS**

Standard Operating Conditions (unless otherwise stated)					
Param. No.	Sym.	Characteristic	Typ.	Units	Conditions
TH01	θJA	Thermal Resistance Junction to Ambient	62.2	°C/W	20-pin DIP package
			77.7	°C/W	20-pin SOIC package
			87.3	°C/W	20-pin SSOP package
			46.2	°C/W	20-pin QFN 4x4mm package
			32.8	°C/W	20-pin UQFN 4x4mm package
TH02	θJC	Thermal Resistance Junction to Case	27.5	°C/W	20-pin DIP package
			23.1	°C/W	20-pin SOIC package
			31.1	°C/W	20-pin SSOP package
			13.2	°C/W	20-pin QFN 4x4mm package
			27.4	°C/W	20-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 <sup>(1)</sup>
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 <sup>(2)</sup>

**Note 1:** IDD is current to run the chip alone without driving any load on the output pins.

**2:** TA = Ambient Temperature; TJ = Junction Temperature

# PIC16(L)F1507

## 25.4 AC Characteristics

Timing Parameter Symbolology has been created with one of the following formats:

1. TppS2ppS
2. TppS

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

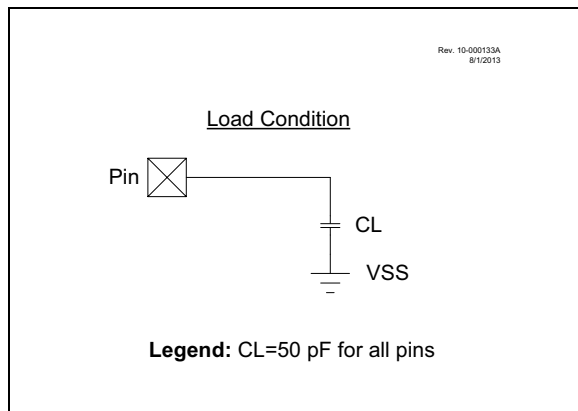
Lowercase letters (pp) and their meanings:

<b>pp</b>			
cc	CCP1	osc	CLKIN
ck	CLKOUT	rd	$\overline{RD}$
cs	$\overline{CS}$	rw	$\overline{RD}$ or $\overline{WR}$
di	SDIx	sc	SCKx
do	SDO	ss	$\overline{SS}$
dt	Data in	t0	T0CKI
io	I/O PORT	t1	T1CKI
mc	$\overline{MCLR}$	wr	$\overline{WR}$

Uppercase letters and their meanings:

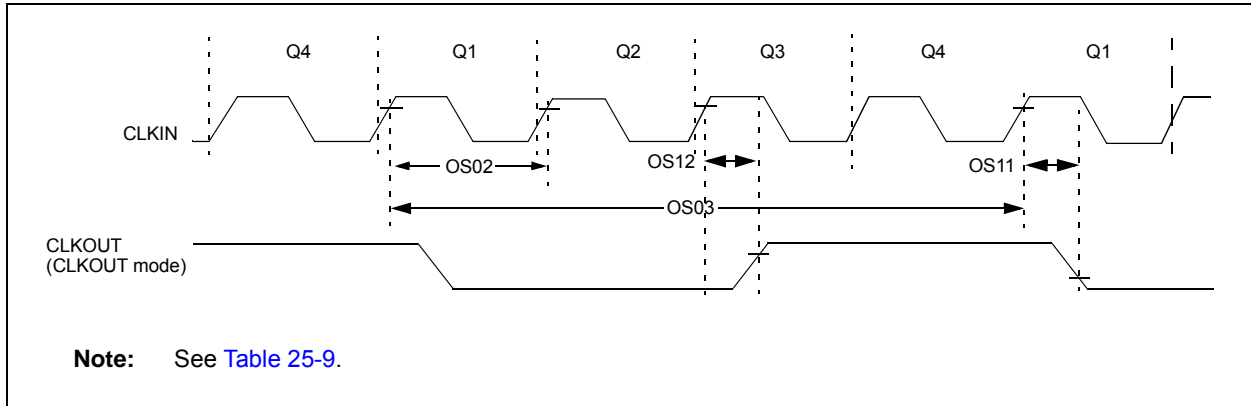
<b>S</b>			
F	Fall	P	Period
H	High	R	Rise
I	Invalid (High-impedance)	V	Valid
L	Low	Z	High-impedance

**FIGURE 25-4: LOAD CONDITIONS**





**FIGURE 25-5: CLOCK TIMING**



**TABLE 25-7: CLOCK OSCILLATOR TIMING REQUIREMENTS**

Standard Operating Conditions (unless otherwise stated)

Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions
OS01	Fosc	External CLKIN Frequency <sup>(1)</sup>	DC	—	0.5	MHz	External Clock (ECL)
			DC	—	4	MHz	External Clock (ECM)
			DC	—	20	MHz	External Clock (ECH)
OS02	Tosc	External CLKIN Period <sup>(1)</sup>	50	—	∞	ns	External Clock (EC)
OS03	Tcy	Instruction Cycle Time <sup>(1)</sup>	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.

# PIC16(L)F1507

**TABLE 25-8: 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 <sup>(1)</sup>	±2%	—	16.0	—	MHz	V <sub>DD</sub> = 3.0V, T <sub>A</sub> = 25°C, (Note 2)
OS09	LFosc	Internal LFINTOSC Frequency	—	—	31	—	kHz	(Note 3)
OS10*	TIOsc ST	HFINTOSC Wake-up from Sleep Start-up Time	—	—	5	15	μs	
OS10A*	TLFOsc ST	LFINTOSC Wake-up from Sleep Start-up Time	—	—	0.5	—	ms	-40°C ≤ T <sub>A</sub> ≤ +125°C

\* These parameters are characterized but not tested.

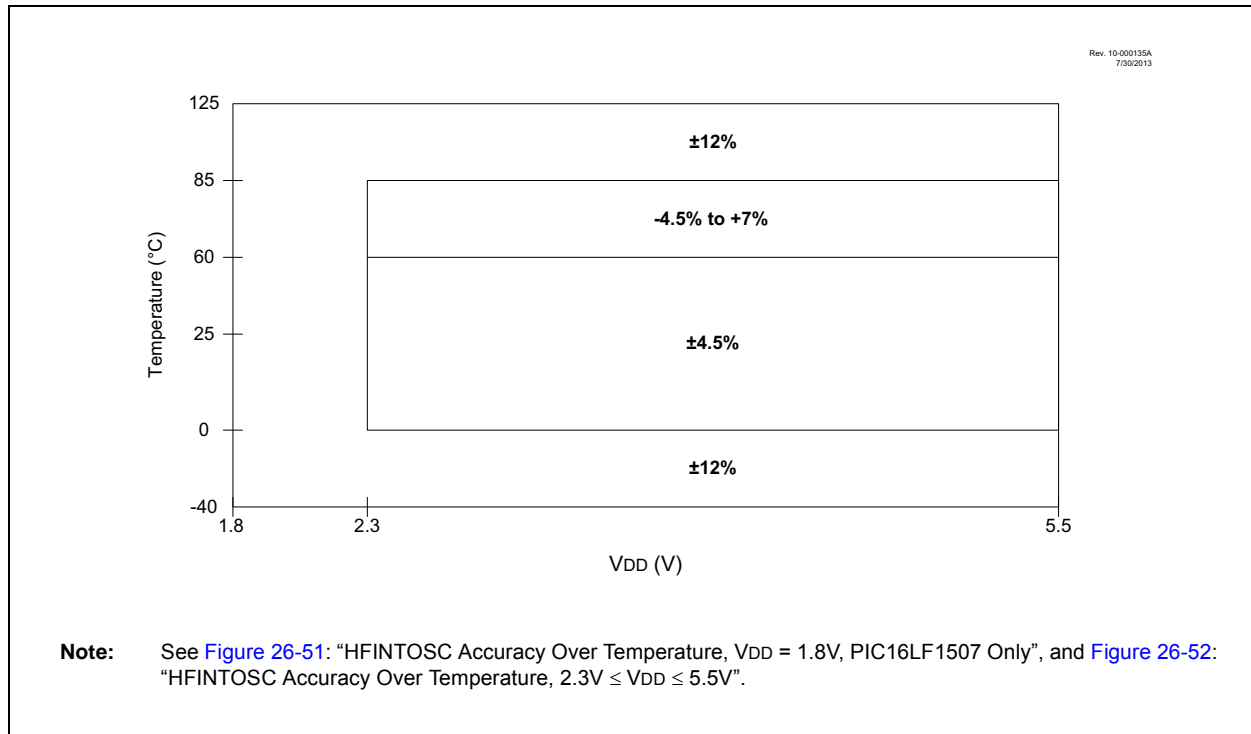
† Data in “Typ” column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note 1:** To ensure these oscillator frequency tolerances, V<sub>DD</sub> and V<sub>SS</sub> must be capacitively decoupled as close to the device as possible. 0.1 μF and 0.01 μF values in parallel are recommended.

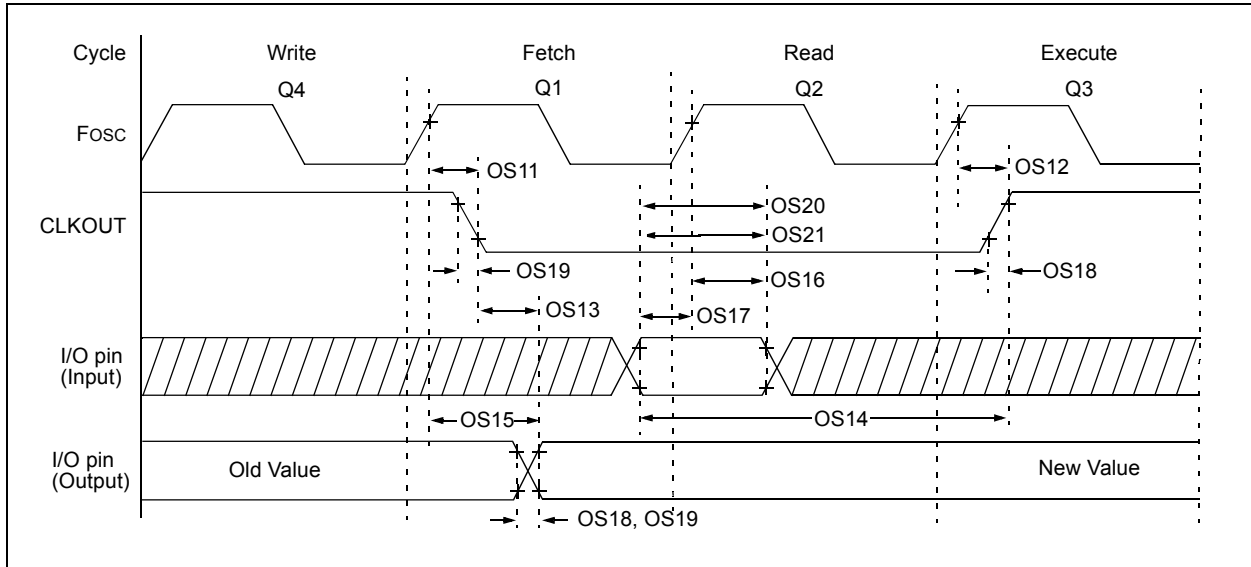
**2:** See Figure 25-6: “HFINTOSC Frequency Accuracy over Device V<sub>DD</sub> and Temperature”, Figure 26-51: “HFINTOSC Accuracy Over Temperature, V<sub>DD</sub> = 1.8V, PIC16LF1507 Only”, and Figure 26-52: “HFINTOSC Accuracy Over Temperature, 2.3V ≤ V<sub>DD</sub> ≤ 5.5V”.

**3:** See Figure 26-49: “LFINTOSC Frequency over V<sub>DD</sub> and Temperature, PIC16LF1507 Only”, and Figure 26-50: “LFINTOSC Frequency over V<sub>DD</sub> and Temperature, PIC16F1507”.

**FIGURE 25-6: HFINTOSC FREQUENCY ACCURACY OVER V<sub>DD</sub> AND TEMPERATURE**



**FIGURE 25-7: CLKOUT AND I/O TIMING**



**TABLE 25-9: 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↓ <sup>(1)</sup>	—	—	70	ns	3.3V ≤ VDD ≤ 5.0V
OS12	TosH2ckH	Fosc↑ to CLKOUT↑ <sup>(1)</sup>	—	—	72	ns	3.3V ≤ VDD ≤ 5.0V
OS13	TckL2ioV	CLKOUT↓ to Port out valid <sup>(1)</sup>	—	—	20	ns	
OS14	TioV2ckH	Port input valid before CLKOUT↑ <sup>(1)</sup>	Tosc + 200 ns	—	—	ns	
OS15	TosH2ioV	Fosc↑ (Q1 cycle) to Port out valid	—	50	70*	ns	3.3V ≤ VDD ≤ 5.0V
OS16	TosH2ioI	Fosc↑ (Q2 cycle) to Port input invalid (I/O in setup time)	50	—	—	ns	3.3V ≤ VDD ≤ 5.0V
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 = 1.8V 3.3V ≤ VDD ≤ 5.0V
OS19*	TioF	Port output fall time	—	28 15	55 30	ns	VDD = 1.8V 3.3V ≤ VDD ≤ 5.0V
OS20*	Tinp	INT pin input high or low time	25	—	—	ns	
OS21*	Tioc	Interrupt-on-change new input level time	25	—	—	ns	

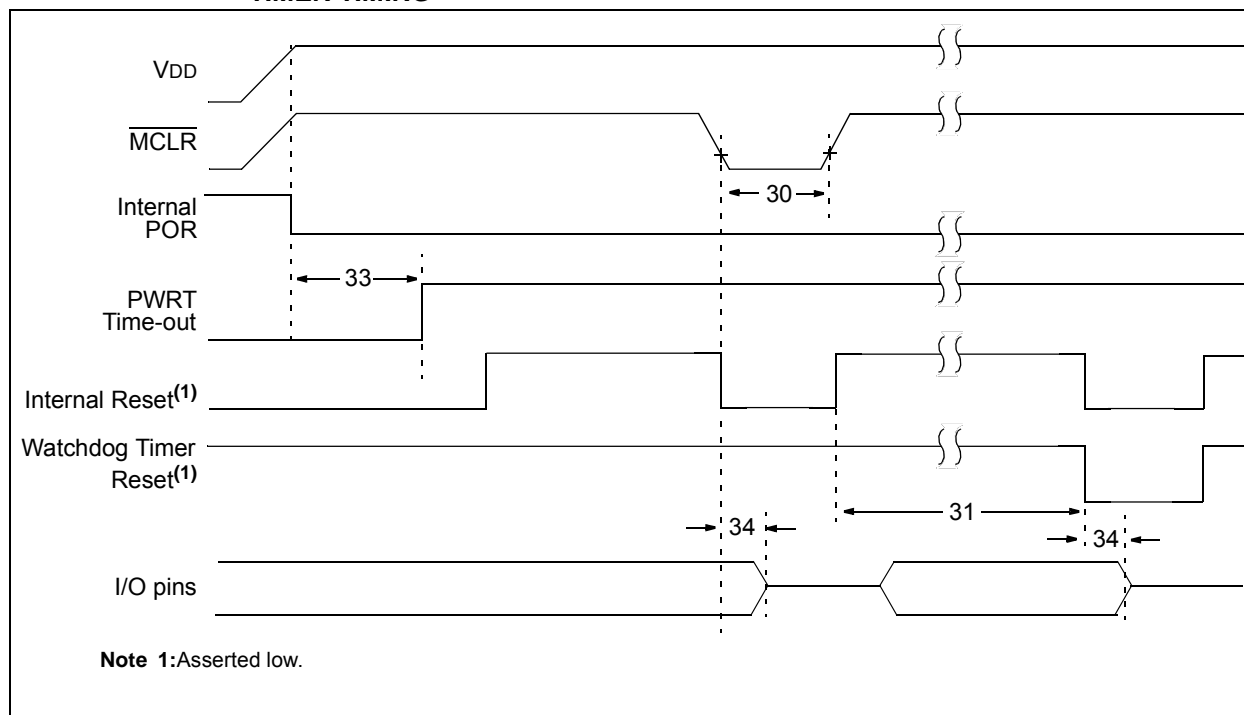
\* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated.

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

# PIC16(L)F1507

**FIGURE 25-8: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING**



**TABLE 25-10: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER AND BROWN-OUT RESET PARAMETERS**

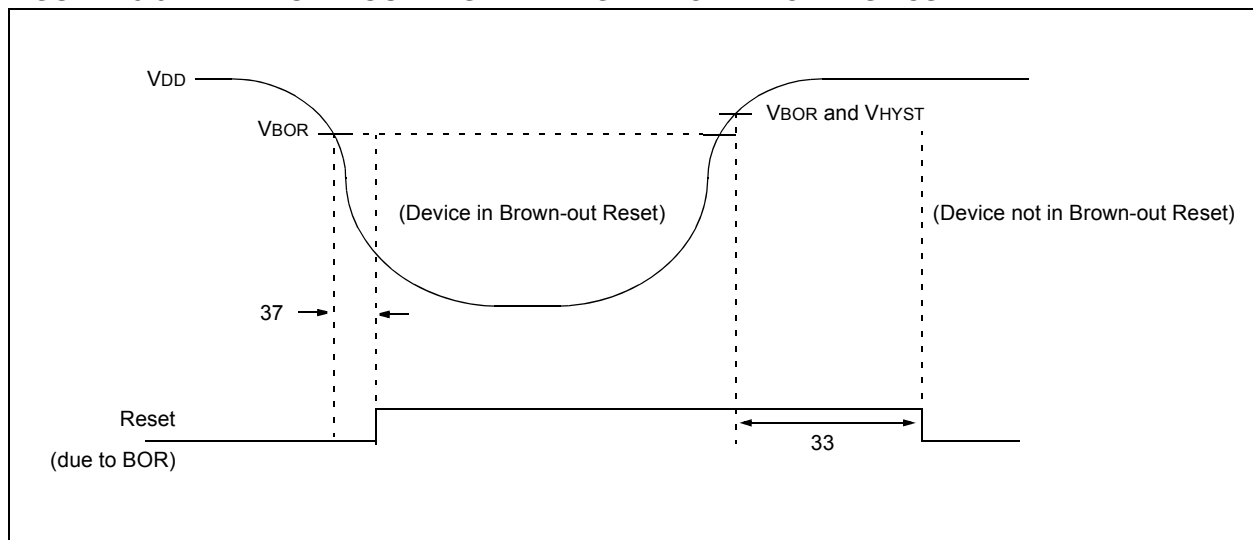
Standard Operating Conditions (unless otherwise stated)							
Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions
30	TMCL	MCLR Pulse Width (low)	2	—	—	μs	
31	TWDTLP	Low-Power Watchdog Timer Time-out Period	10	16	27	ms	VDD = 3.3V-5V, 1:512 Prescaler used
33*	TPWRT	Power-up Timer Period	40	65	140	ms	PWRTE = 0
34*	TIOZ	I/O high-impedance from MCLR Low or Watchdog Timer Reset	—	—	2.0	μs	
35	VBOR	Brown-out Reset Voltage <sup>(1)</sup>	2.55	2.70	2.85	V	BORV = 0
			2.35	2.45	2.58	V	BORV = 1 (PIC16F1507)
			1.80	1.90	2.05	V	BORV = 1 (PIC16LF1507)
36*	VHYST	Brown-out Reset Hysteresis	0	25	75	mV	-40°C ≤ TA ≤ +85°C
37*	TBORDC	Brown-out Reset DC Response Time	1	16	35	μs	VDD ≤ VBOR
38	VLPBOR	Low-Power Brown-Out Reset Voltage	1.8	2.1	2.5	V	LPBOR = 1

\* 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.

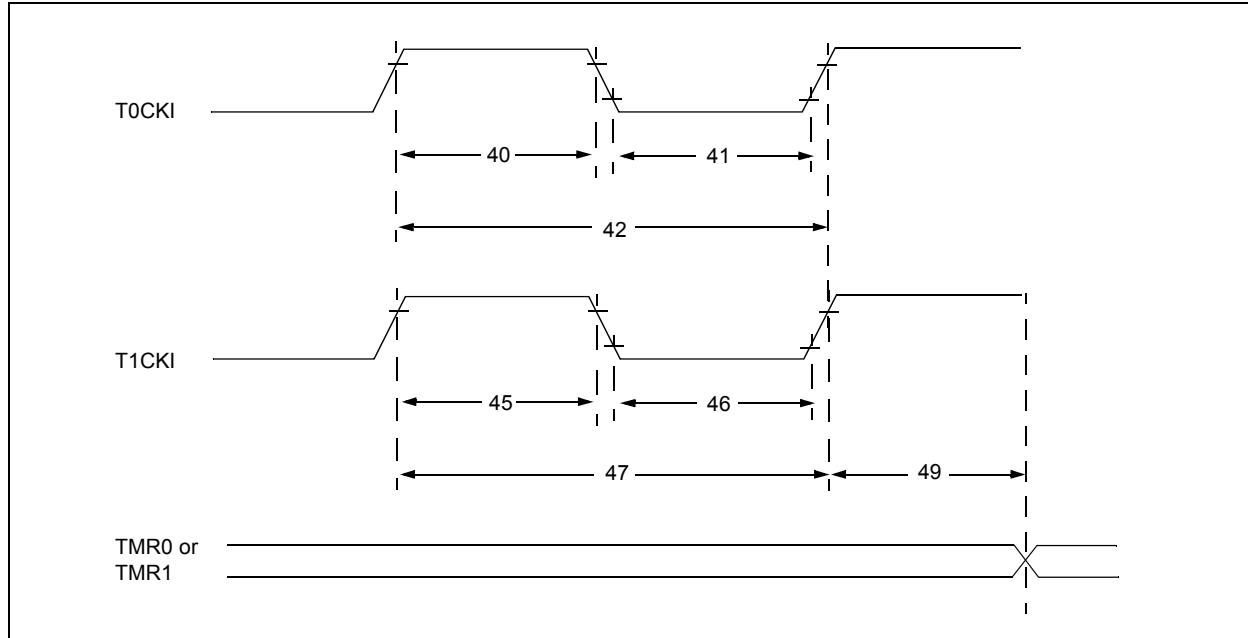
**Note 1:** To ensure these voltage tolerances, VDD and VSS must be capacitively decoupled as close to the device as possible. 0.1 μF and 0.01 μF values in parallel are recommended.

**FIGURE 25-9: BROWN-OUT RESET TIMING AND CHARACTERISTICS**



# PIC16(L)F1507

**FIGURE 25-10: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS**



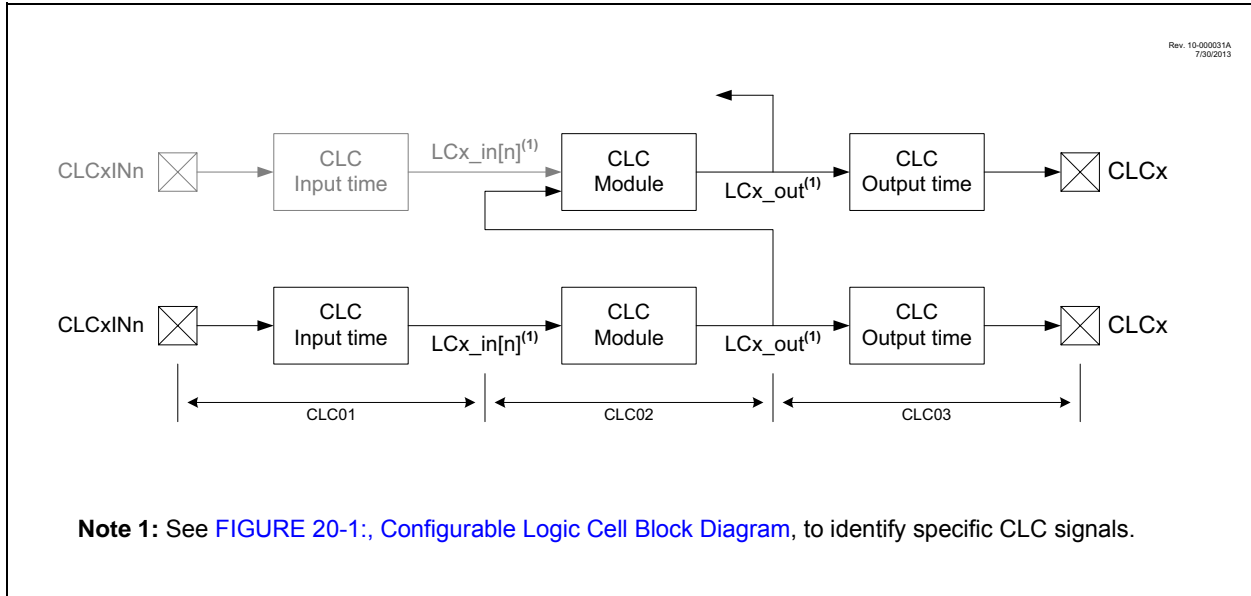
**TABLE 25-11: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS**

Standard Operating Conditions (unless otherwise stated)								
Param. No.	Sym.	Characteristic		Min.	Typ†	Max.	Units	Conditions
40*	Tt0H	T0CKI High Pulse Width	No Prescaler	$0.5 T_{CY} + 20$	—	—	ns	
			With Prescaler	10	—	—	ns	
41*	Tt0L	T0CKI Low Pulse Width	No Prescaler	$0.5 T_{CY} + 20$	—	—	ns	
			With Prescaler	10	—	—	ns	
42*	Tt0P	T0CKI Period		Greater of: $20$ or $\frac{T_{CY} + 40}{N}$	—	—	ns	N = prescale value
45*	Tt1H	T1CKI High Time	Synchronous, No Prescaler	$0.5 T_{CY} + 20$	—	—	ns	
			Synchronous, with Prescaler	15	—	—	ns	
			Asynchronous	30	—	—	ns	
46*	Tt1L	T1CKI Low Time	Synchronous, No Prescaler	$0.5 T_{CY} + 20$	—	—	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
			Asynchronous	60	—	—	ns	
49*	TCKEZTMR1	Delay from External Clock Edge to Timer Increment		$2 T_{OSC}$	—	$7 T_{OSC}$	—	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.

**FIGURE 25-11: CLC PROPAGATION TIMING**



**TABLE 25-12: CONFIGURATION LOGIC CELL (CLC) CHARACTERISTICS**

Standard Operating Conditions (unless otherwise stated)								
Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions	
CLC01*	TCLCIN	CLC input time	—	7	—	ns		
CLC02*	TCLC	CLC module input to output propagation time	—	24	—	ns	$V_{DD} = 1.8V$	
			—	12	—	ns	$V_{DD} > 3.6V$	
CLC03*	TCLCOUT	CLC output time	Rise Time	—	OS18	—	—	<b>(Note 1)</b>
			Fall Time	—	OS19	—	—	<b>(Note 1)</b>
CLC04*	FCLCMAX	CLC maximum switching frequency	—	45	—	MHz		

\* 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:** See [Table 25-9](#) for OS18 and OS19 rise and fall times.

# PIC16(L)F1507

**TABLE 25-13: ANALOG-TO-DIGITAL CONVERTER (ADC) CHARACTERISTICS<sup>(1,2,3)</sup>**

Operating Conditions (unless otherwise stated) V <sub>DD</sub> = 3.0V, T <sub>A</sub> = 25°C							
Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions
AD01	NR	Resolution	—	—	10	bit	
AD02	EIL	Integral Error	—	±1	±1.7	LSb	V <sub>REF</sub> = 3.0V
AD03	EDL	Differential Error	—	±1	±1	LSb	No missing codes V <sub>REF</sub> = 3.0V
AD04	E <sub>OFF</sub>	Offset Error	—	±1	±2.5	LSb	V <sub>REF</sub> = 3.0V
AD05	E <sub>GN</sub>	Gain Error	—	±1	±2.0	LSb	V <sub>REF</sub> = 3.0V
AD06	V <sub>REF</sub>	Reference Voltage	1.8	—	V <sub>DD</sub>	V	V <sub>REF</sub> = (V <sub>RPOS</sub> - V <sub>RNEG</sub> ) ( <b>Note 4</b> )
AD07	V <sub>AIN</sub>	Full-Scale Range	V <sub>SS</sub>	—	V <sub>REF</sub>	V	
AD08	Z <sub>AIN</sub>	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.

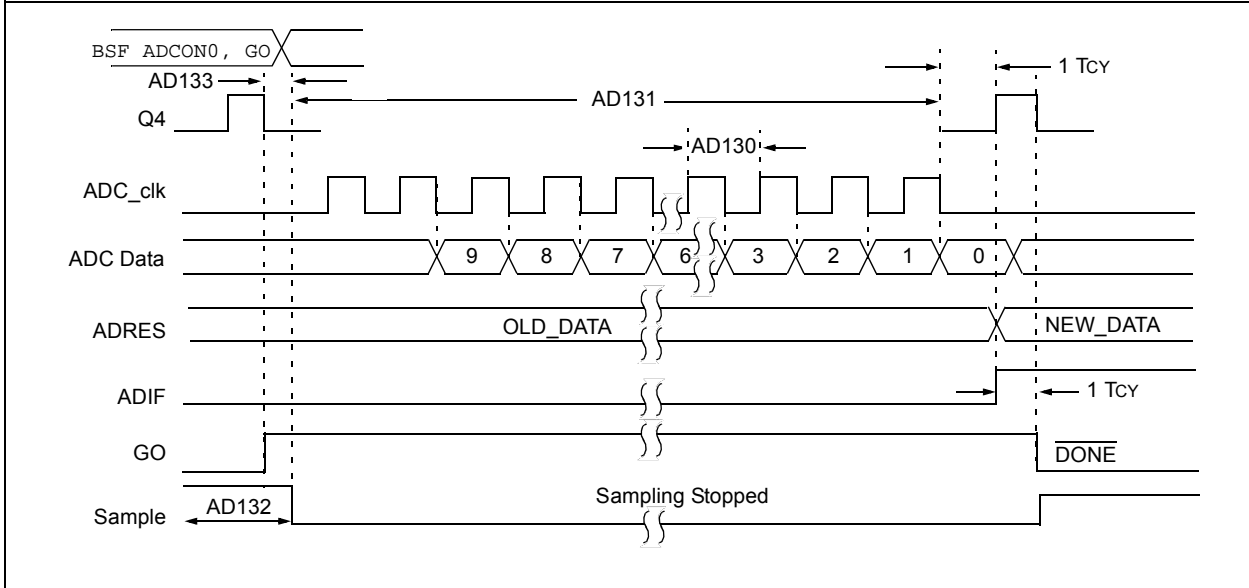
**2:** The ADC conversion result never decreases with an increase in the input voltage and has no missing codes.

**3:** See [Section 26.0 “DC and AC Characteristics Graphs and Charts”](#) for operating characterization.

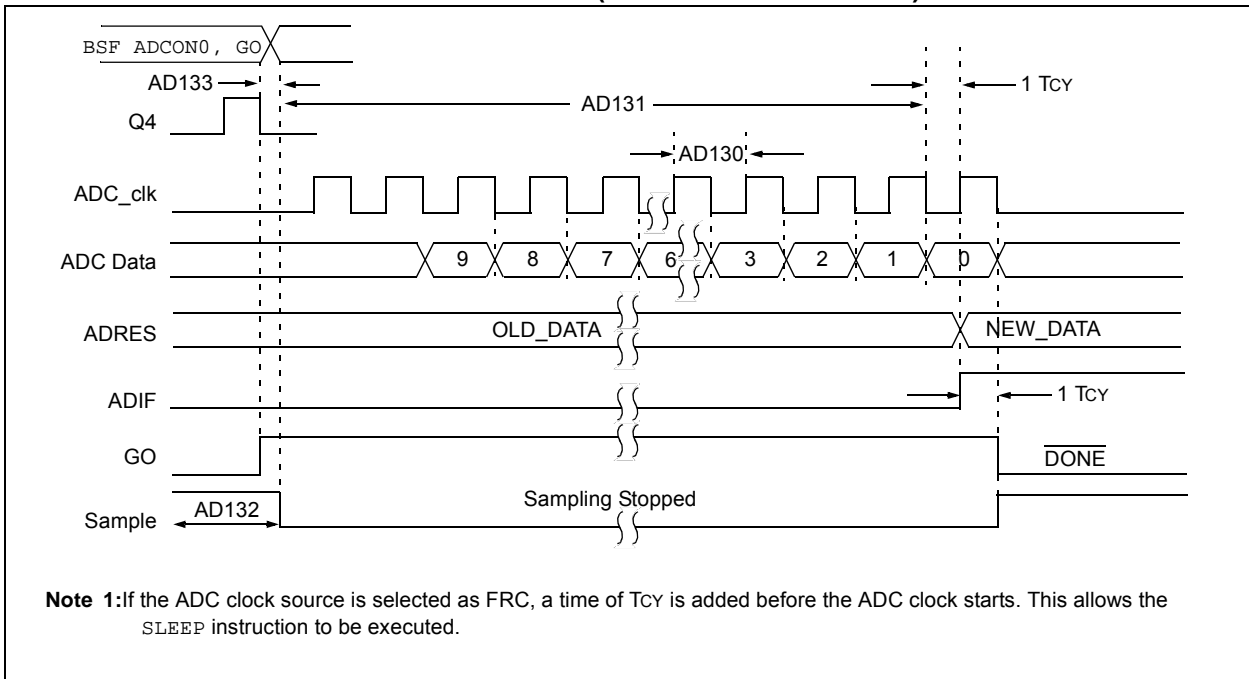
**4:** ADC V<sub>REF</sub> is selected by ADPREF<0> bit.



**FIGURE 25-12: ADC CONVERSION TIMING (ADC CLOCK Fosc-BASED)**



**FIGURE 25-13: ADC CONVERSION TIMING (ADC CLOCK FROM FRC)**



# PIC16(L)F1507

**TABLE 25-14: ADC 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) <sup>(1)</sup>	—	11	—	TAD	Set GO/DONE bit to conversion complete
AD132*	TACQ	Acquisition Time	—	5.0	—	μs	
AD133*	THCD	Holding Capacitor Disconnect Time	—	1/2 TAD	—		FOSC-based
			—	1/2 TAD + 1TCY	—		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.

## 26.0 DC AND AC CHARACTERISTICS GRAPHS AND CHARTS

The graphs and tables provided in this section are for **design guidance** and are **not tested**.

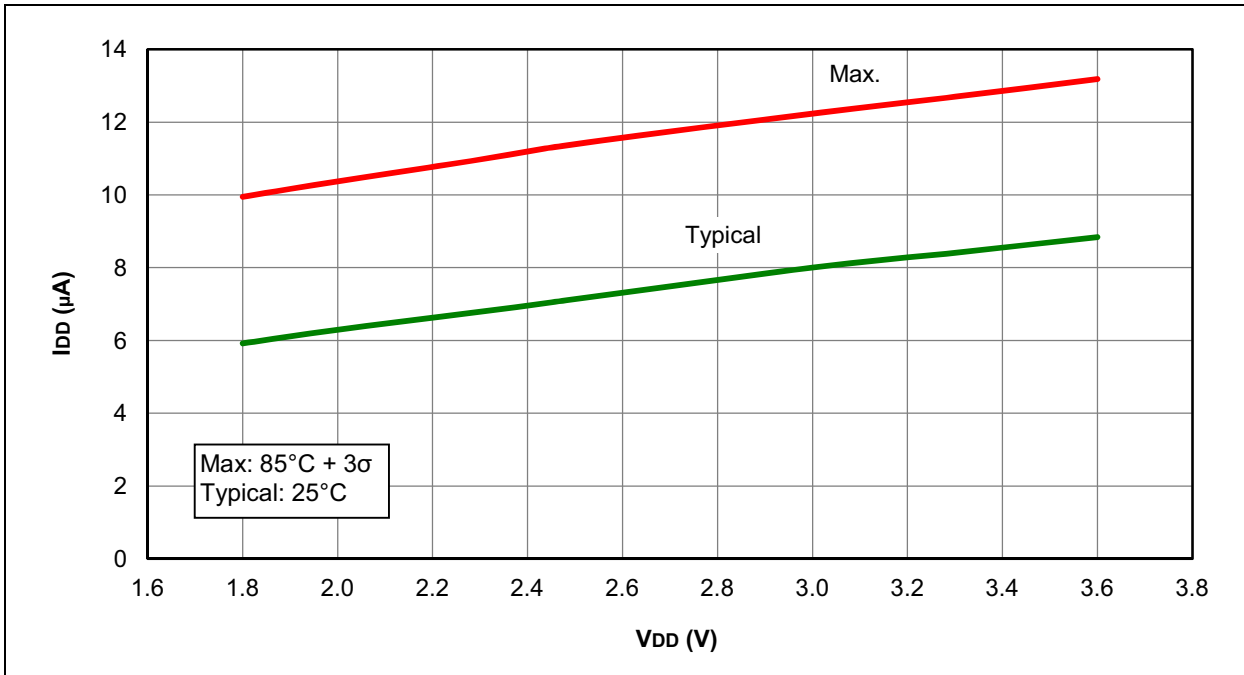
In some graphs or tables, the data presented are **outside specified operating range** (i.e., outside specified VDD range). This is for **information only** and devices are ensured to operate properly only within the specified range.

<p><b>Note:</b> The graphs and tables provided following this note are a statistical summary based on a limited number of samples and are provided for informational purposes only. The performance characteristics listed herein are not tested or guaranteed. In some graphs or tables, the data presented may be outside the specified operating range (e.g., outside specified power supply range) and therefore, outside the warranted range.</p>
--

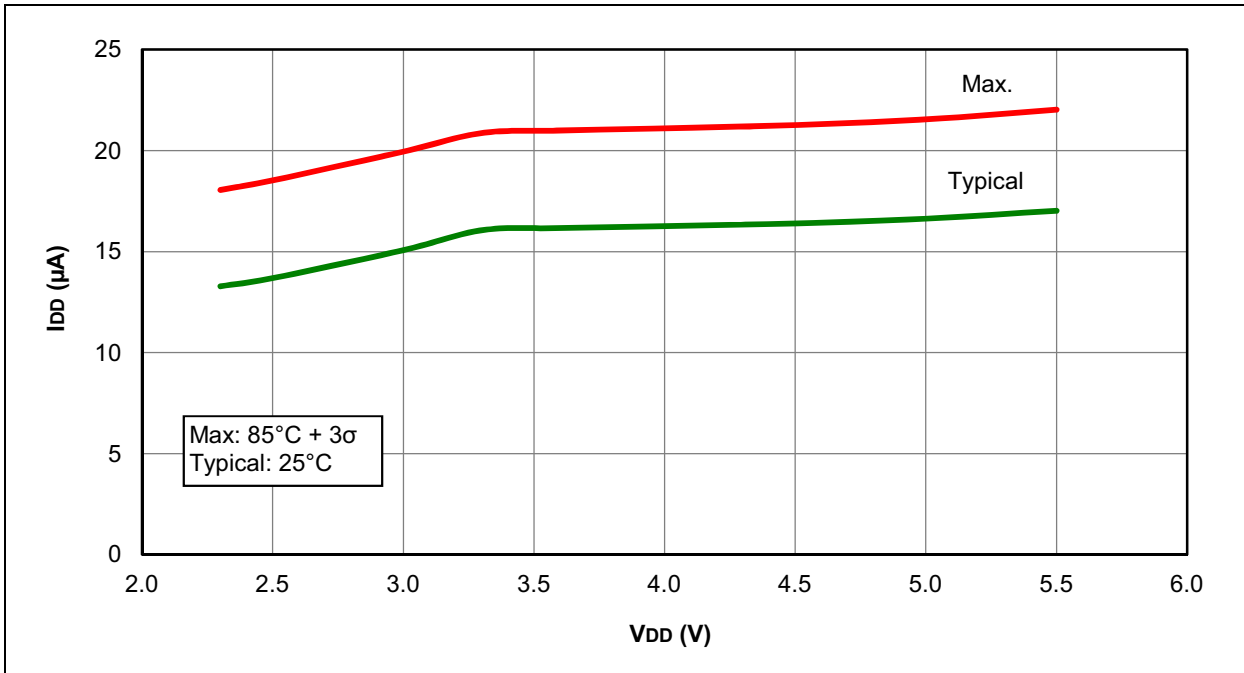
“Typical” represents the mean of the distribution at 25°C. “MAXIMUM”, “Max.”, “MINIMUM” or “Min.” represents (mean + 3 $\sigma$ ) or (mean - 3 $\sigma$ ) respectively, where  $\sigma$  is a standard deviation, over each temperature range.

# PIC16(L)F1507

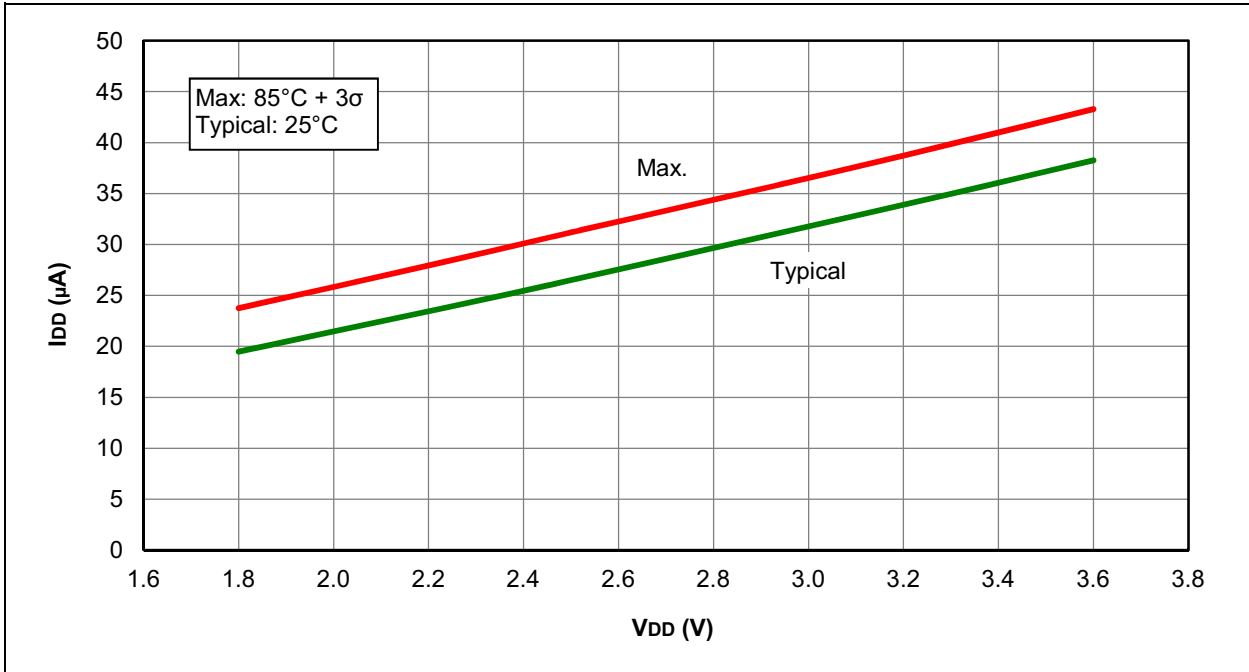
**FIGURE 26-1:  $I_{DD}$ , EXTERNAL CLOCK (ECL), LOW-POWER MODE,  $F_{osc} = 32$  kHz, PIC16LF1507 ONLY**



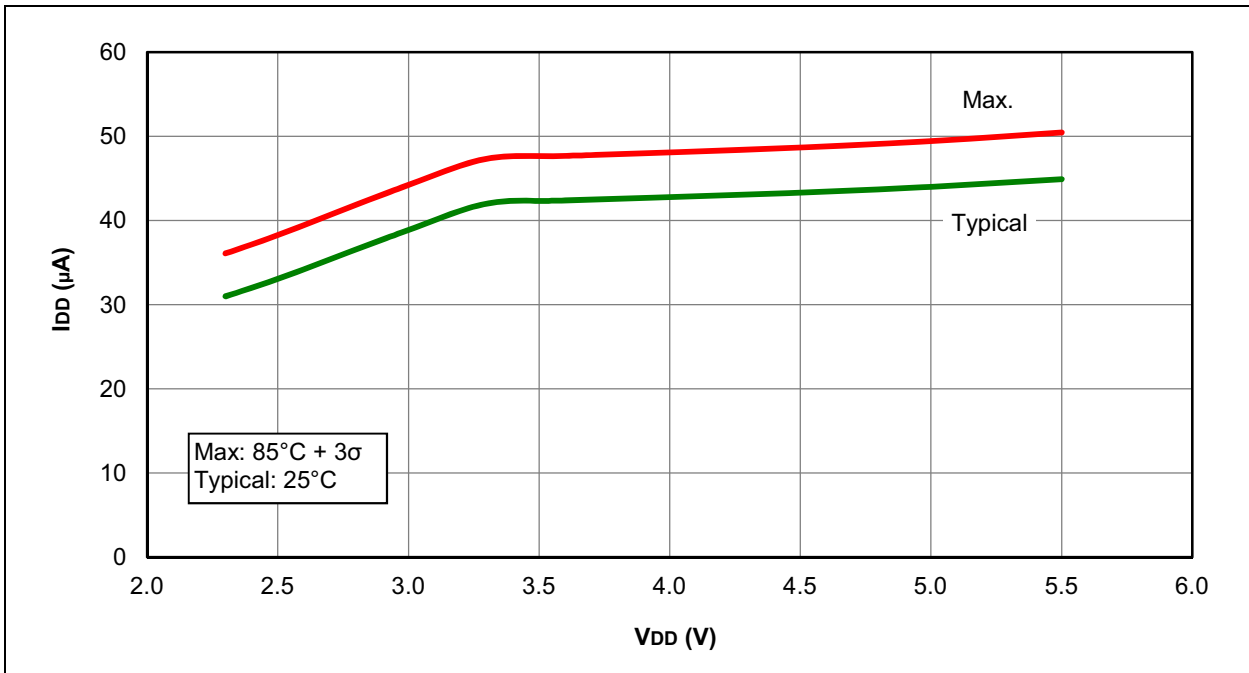
**FIGURE 26-2:  $I_{DD}$ , EXTERNAL CLOCK (ECL), LOW-POWER MODE,  $F_{osc} = 32$  kHz, PIC16F1507 ONLY**



**FIGURE 26-3:  $I_{DD}$ , EXTERNAL CLOCK (ECL), LOW-POWER MODE,  $F_{osc} = 500$  kHz, PIC16LF1507 ONLY**

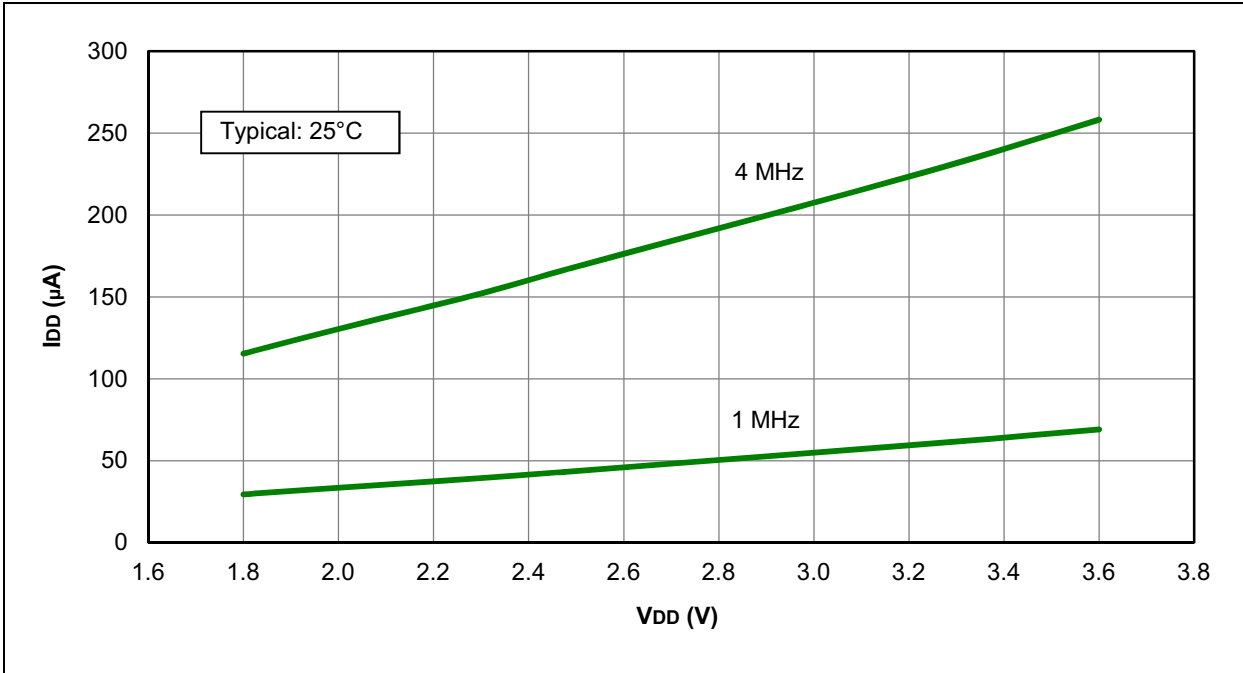


**FIGURE 26-4:  $I_{DD}$ , EXTERNAL CLOCK (ECL), LOW-POWER MODE,  $F_{osc} = 500$  kHz, PIC16F1507 ONLY**

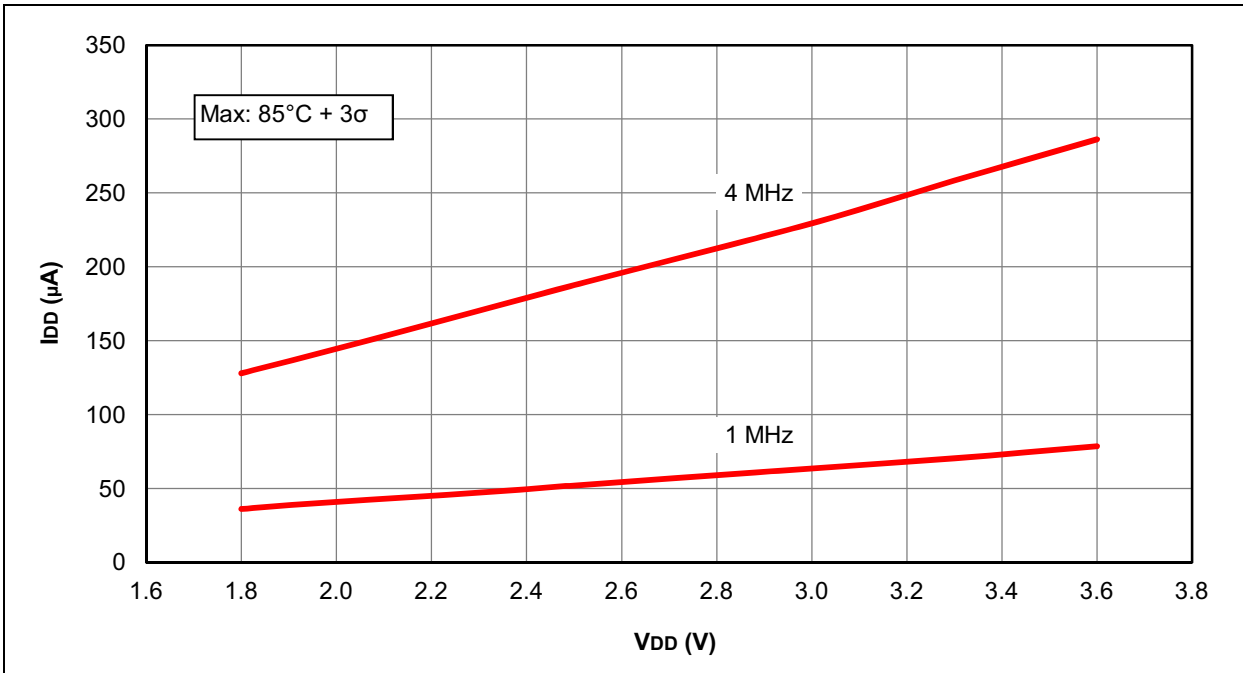


# PIC16(L)F1507

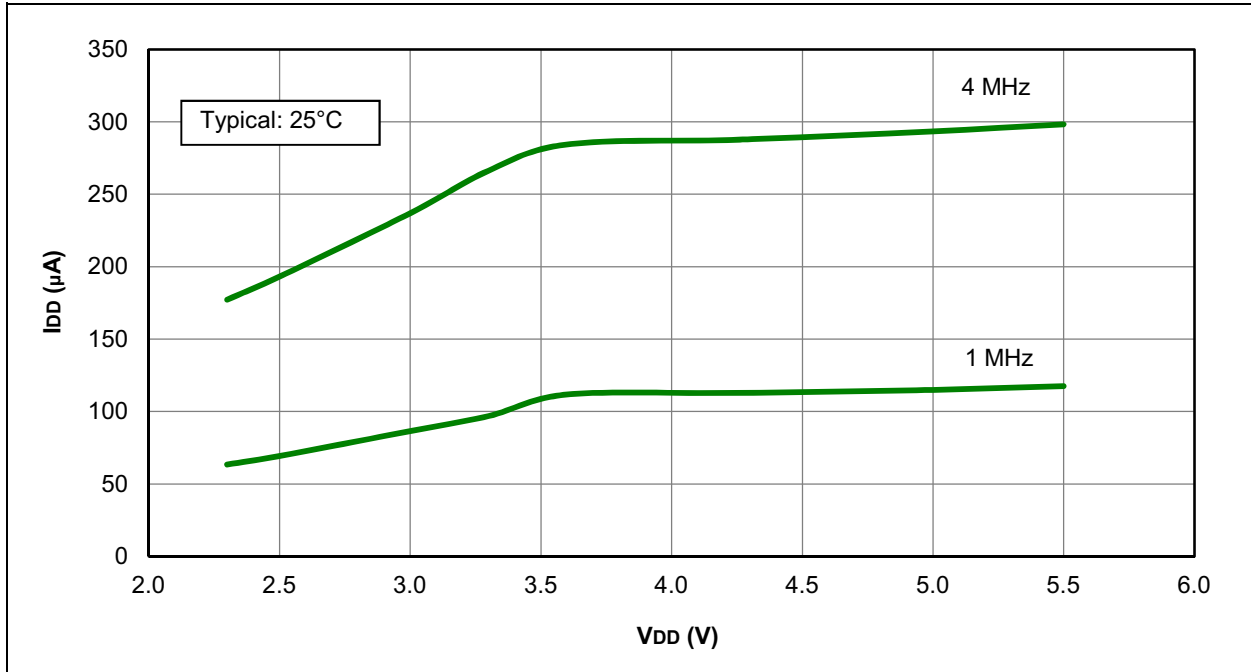
**FIGURE 26-5: I<sub>DD</sub> TYPICAL, EXTERNAL CLOCK (ECM), MEDIUM POWER MODE, PIC16LF1507 ONLY**



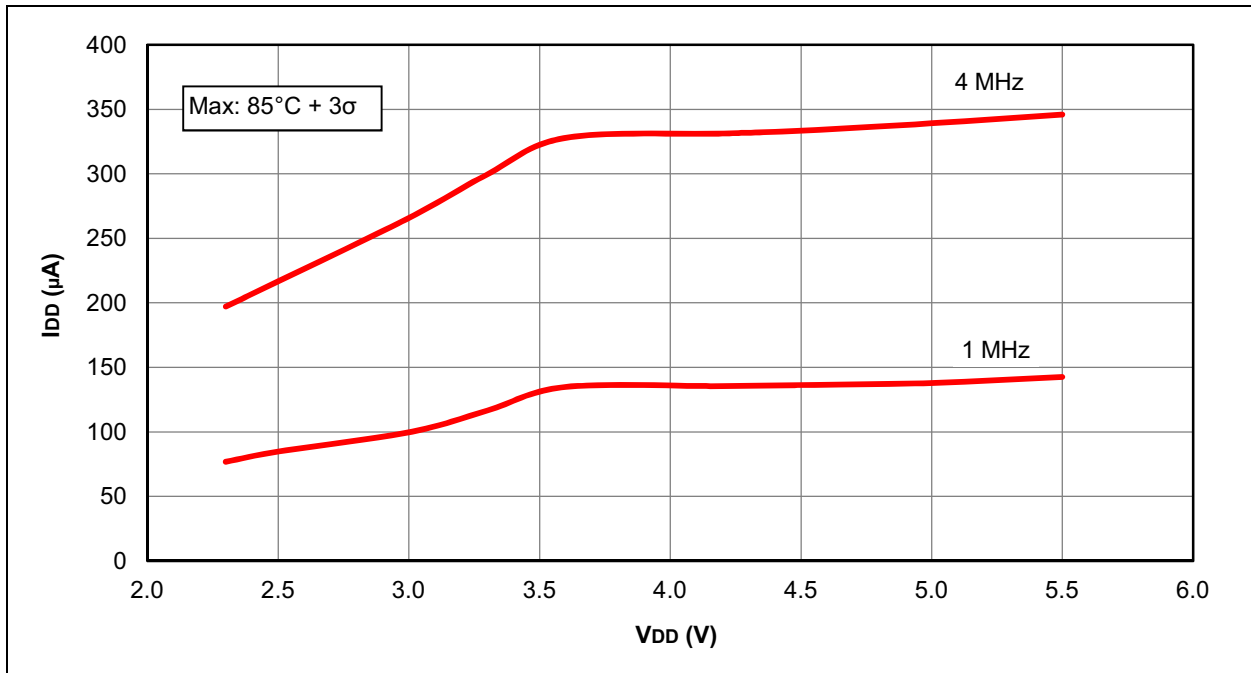
**FIGURE 26-6: I<sub>DD</sub> MAXIMUM, EXTERNAL CLOCK (ECM), MEDIUM POWER MODE, PIC16LF1507 ONLY**



**FIGURE 26-7: I<sub>DD</sub> TYPICAL, EXTERNAL CLOCK (ECM), MEDIUM POWER MODE, PIC16F1507 ONLY**

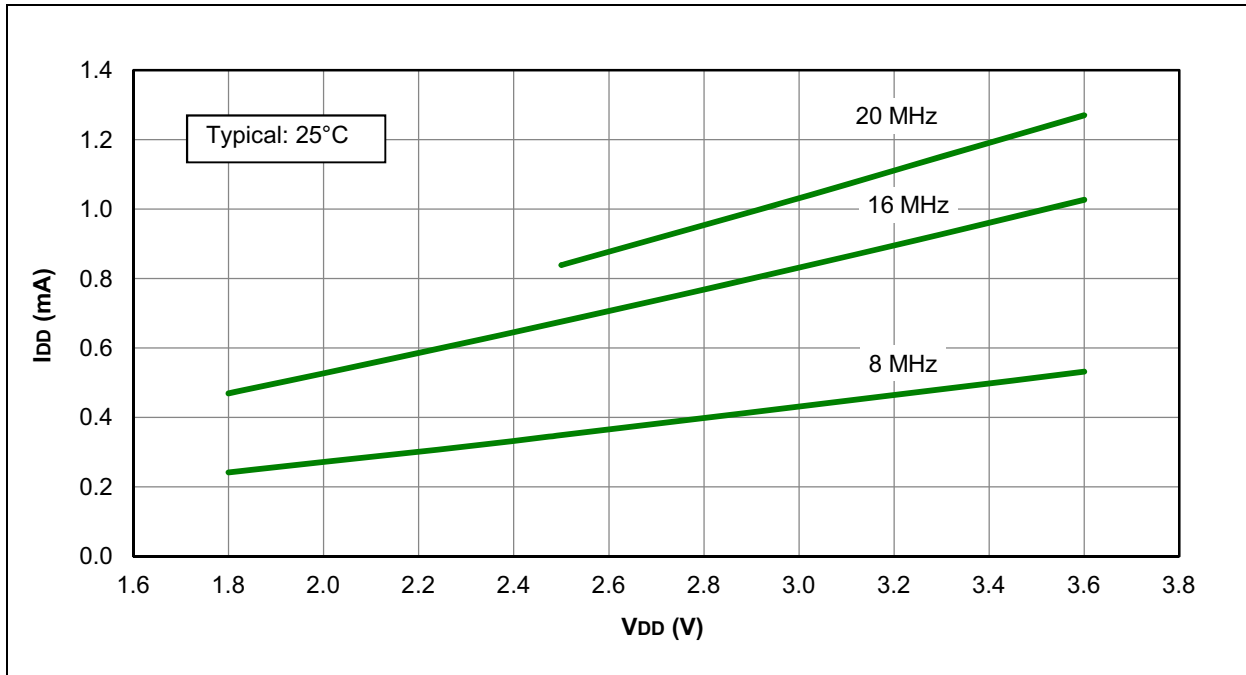


**FIGURE 26-8: I<sub>DD</sub> MAXIMUM, EXTERNAL CLOCK (ECM), MEDIUM POWER MODE, PIC16F1507 ONLY**

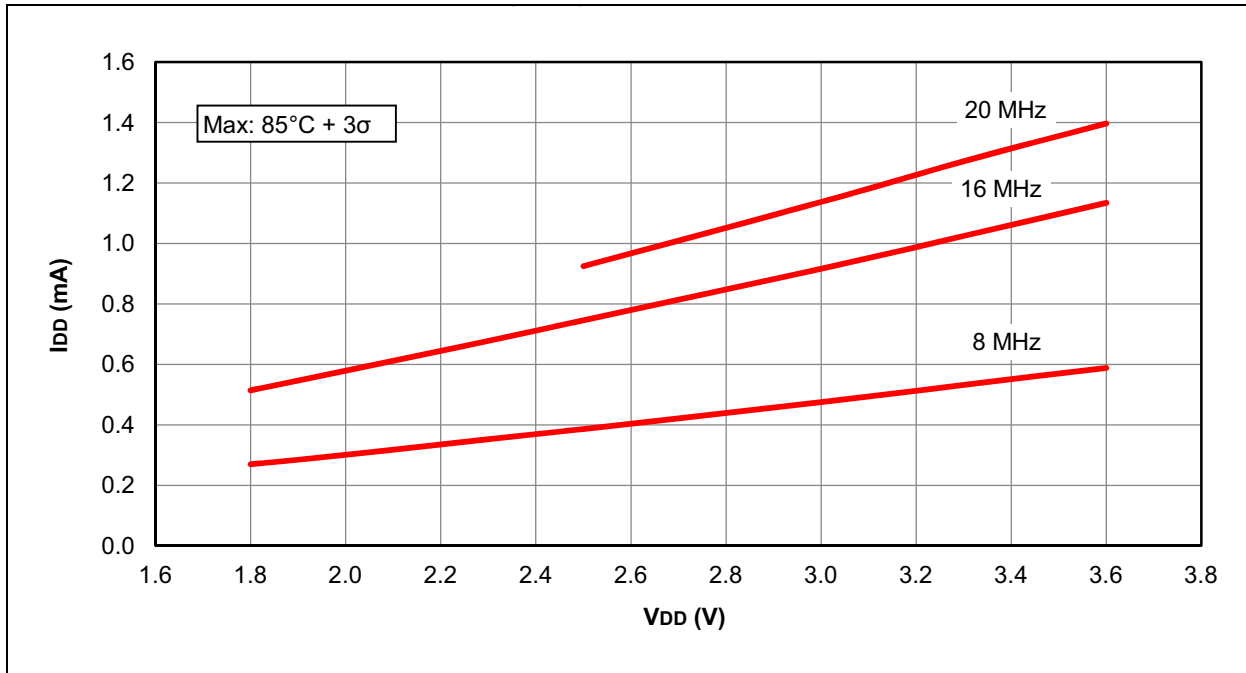


# PIC16(L)F1507

**FIGURE 26-9: I<sub>DD</sub> TYPICAL, EXTERNAL CLOCK (ECH), HIGH-POWER MODE, PIC16LF1507 ONLY**

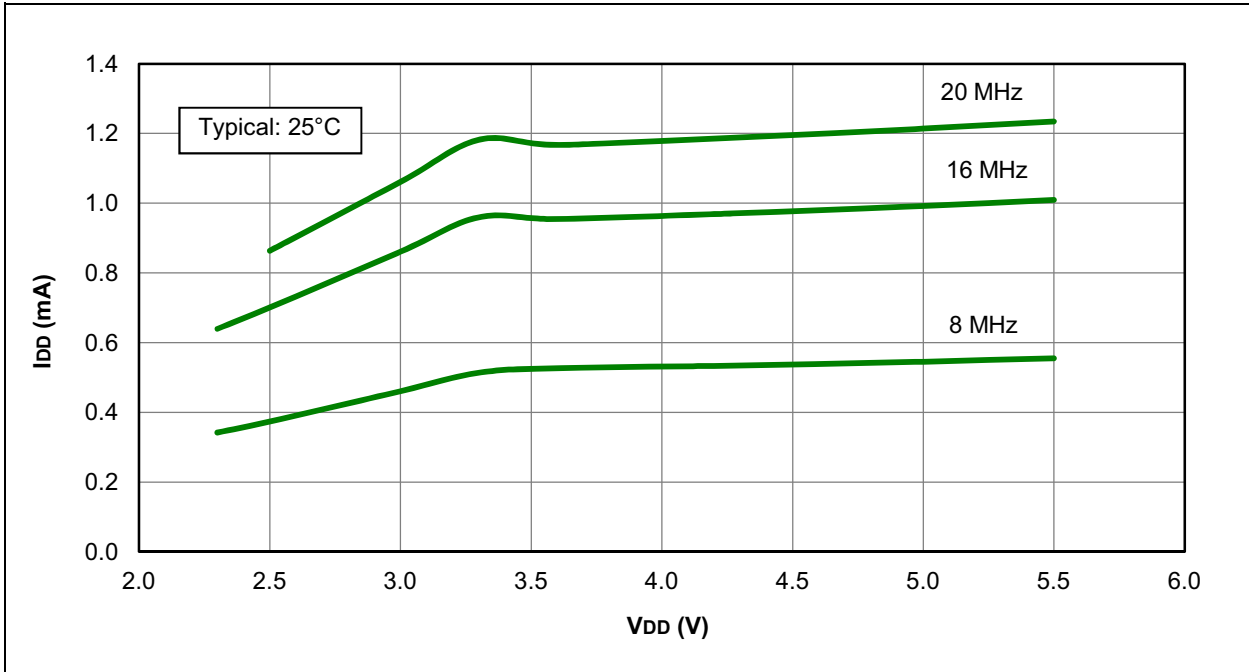


**FIGURE 26-10: I<sub>DD</sub> MAXIMUM, EXTERNAL CLOCK (ECH), HIGH-POWER MODE, PIC16LF1507 ONLY**

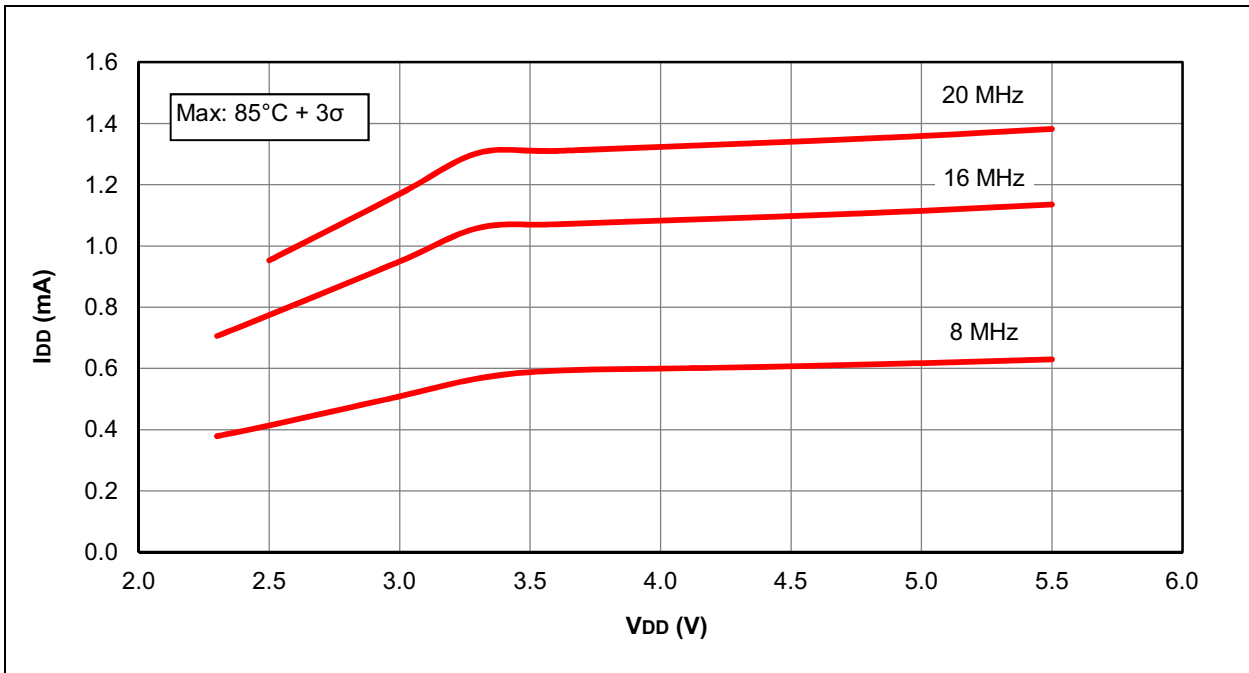




**FIGURE 26-11: I<sub>DD</sub> TYPICAL, EXTERNAL CLOCK (ECH), HIGH-POWER MODE, PIC16F1507 ONLY**



**FIGURE 26-12: I<sub>DD</sub> MAXIMUM, EXTERNAL CLOCK (ECH), HIGH-POWER MODE, PIC16F1507 ONLY**



# PIC16(L)F1507

FIGURE 26-13:  $I_{DD}$ , LFINTOSC,  $F_{OSC} = 31$  kHz, PIC16LF1507 ONLY

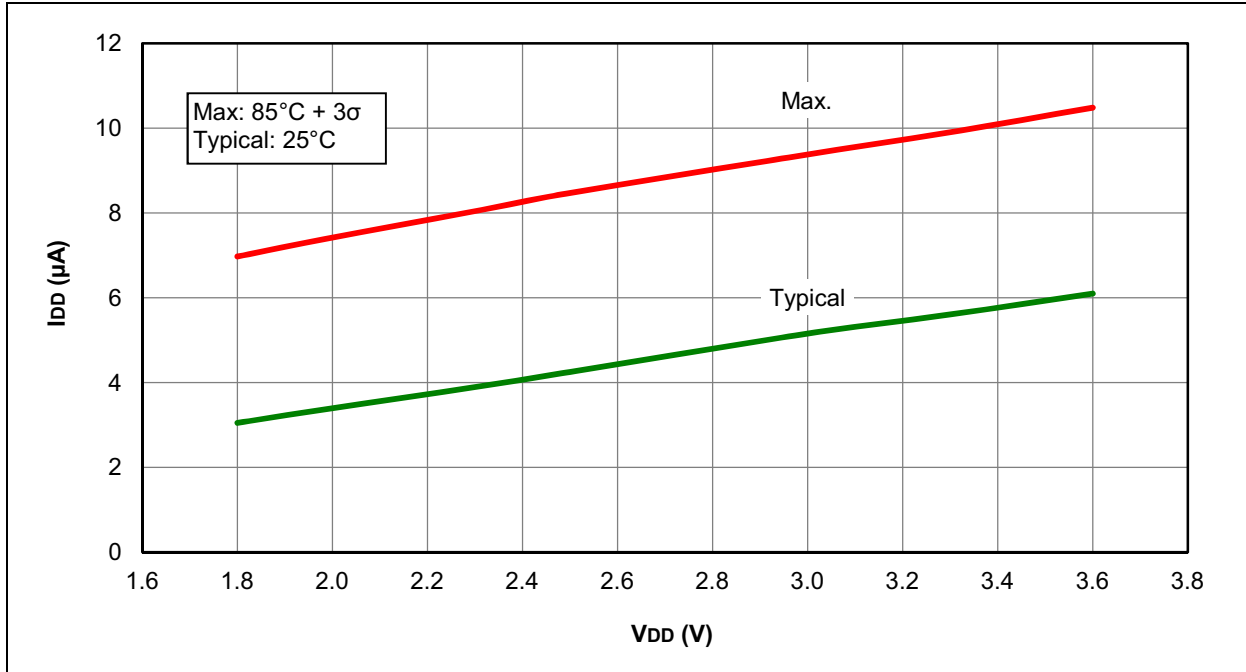
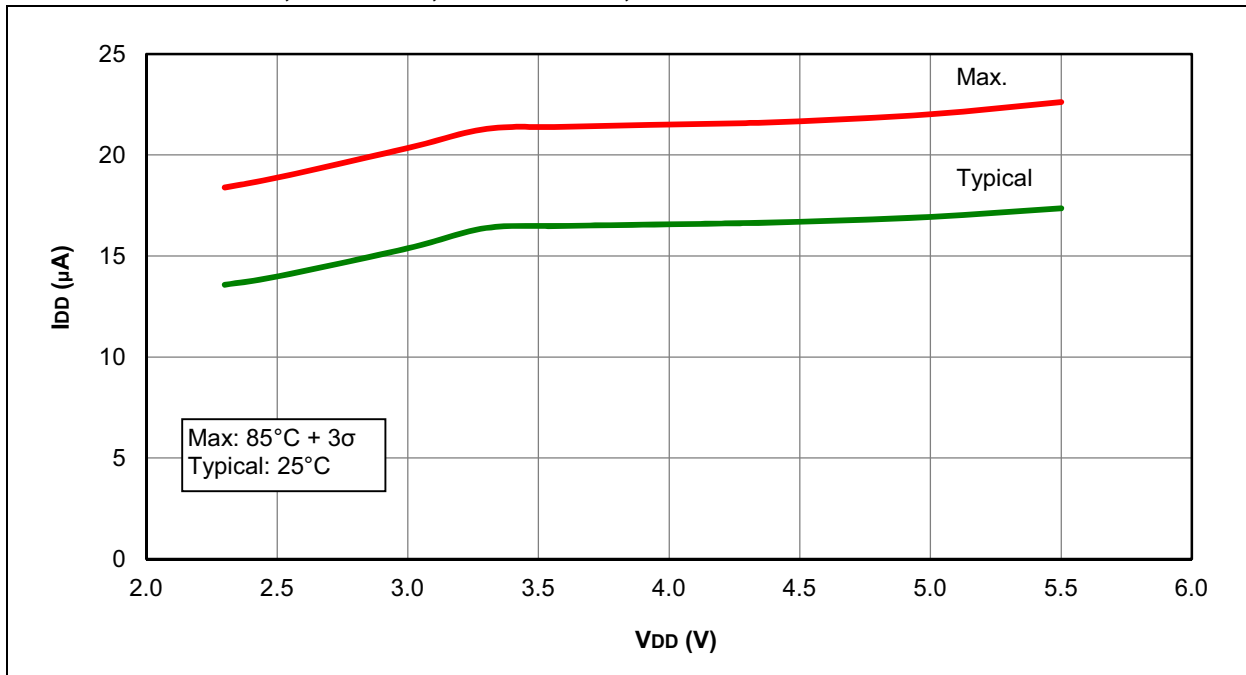
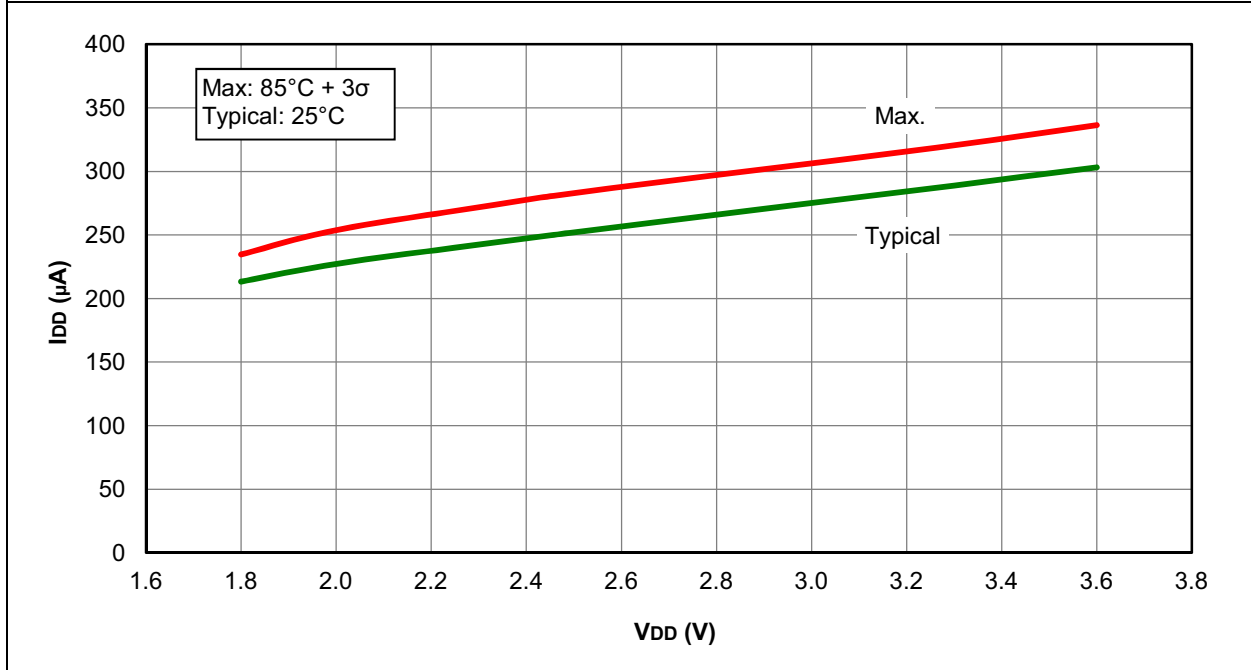


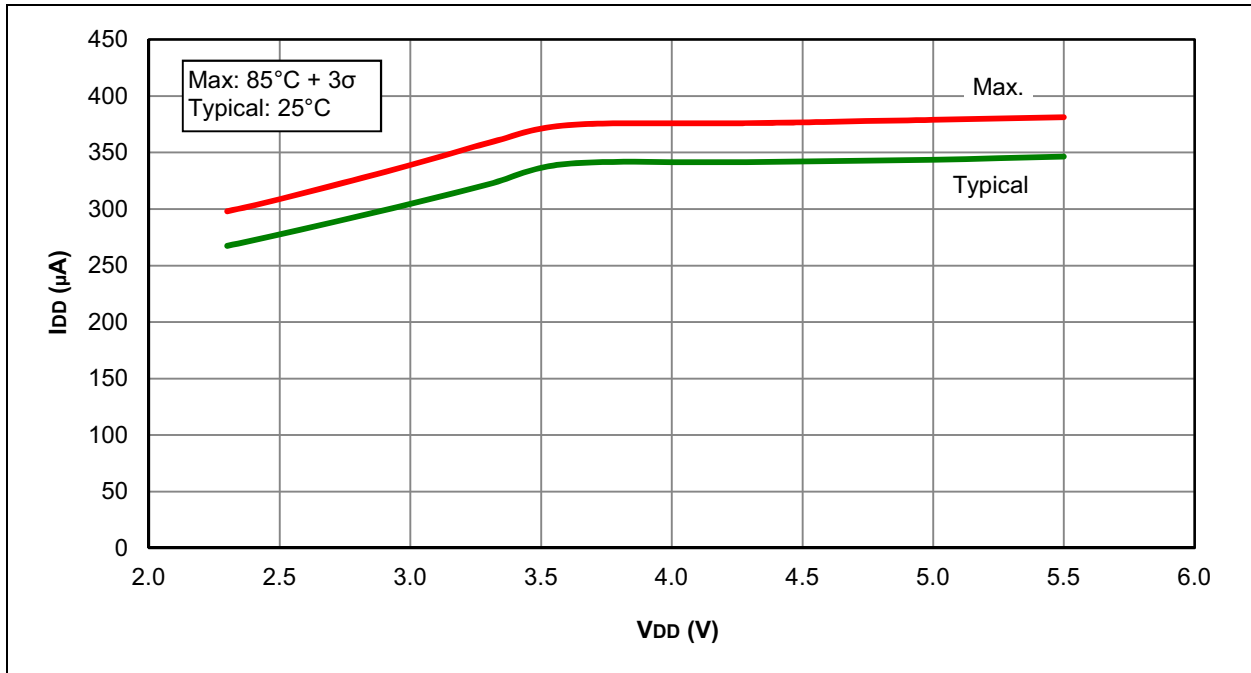
FIGURE 26-14:  $I_{DD}$ , LFINTOSC,  $F_{OSC} = 31$  kHz, PIC16F1507 ONLY



**FIGURE 26-15:  $I_{DD}$ , MFINTOSC,  $F_{osc} = 500$  kHz, PIC16LF1507 ONLY**



**FIGURE 26-16:  $I_{DD}$ , MFINTOSC,  $F_{osc} = 500$  kHz, PIC16F1507 ONLY**



# PIC16(L)F1507

FIGURE 26-17: I<sub>DD</sub> TYPICAL, HFINTOSC, PIC16LF1507 ONLY

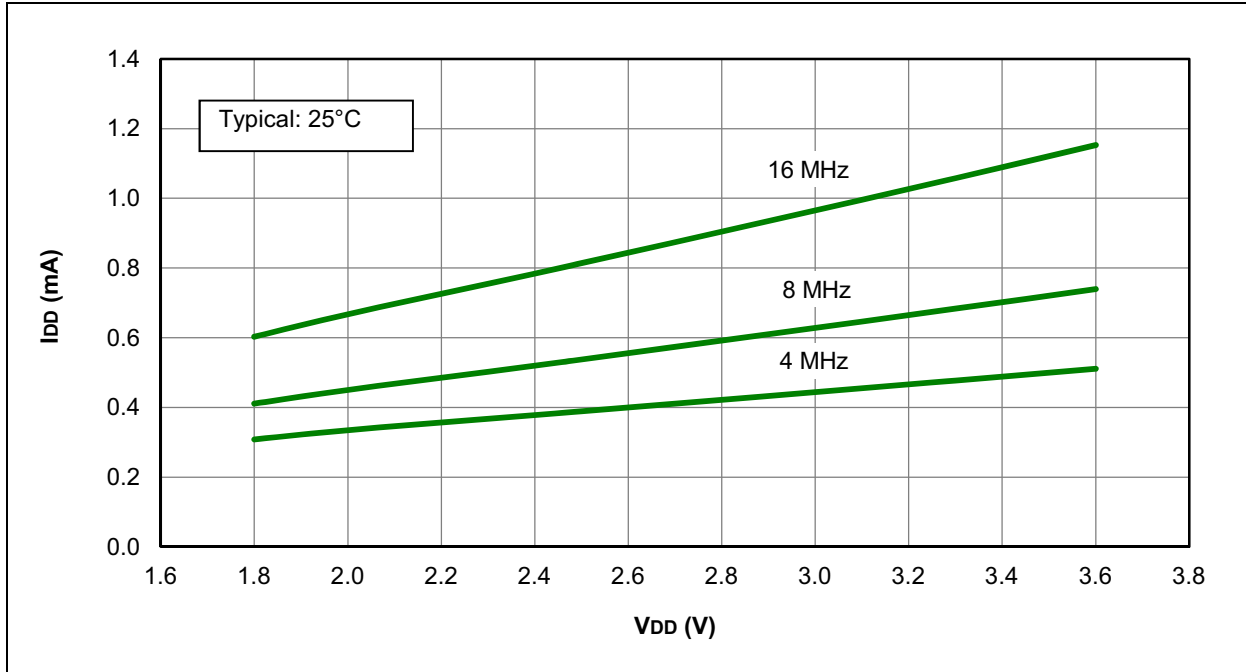
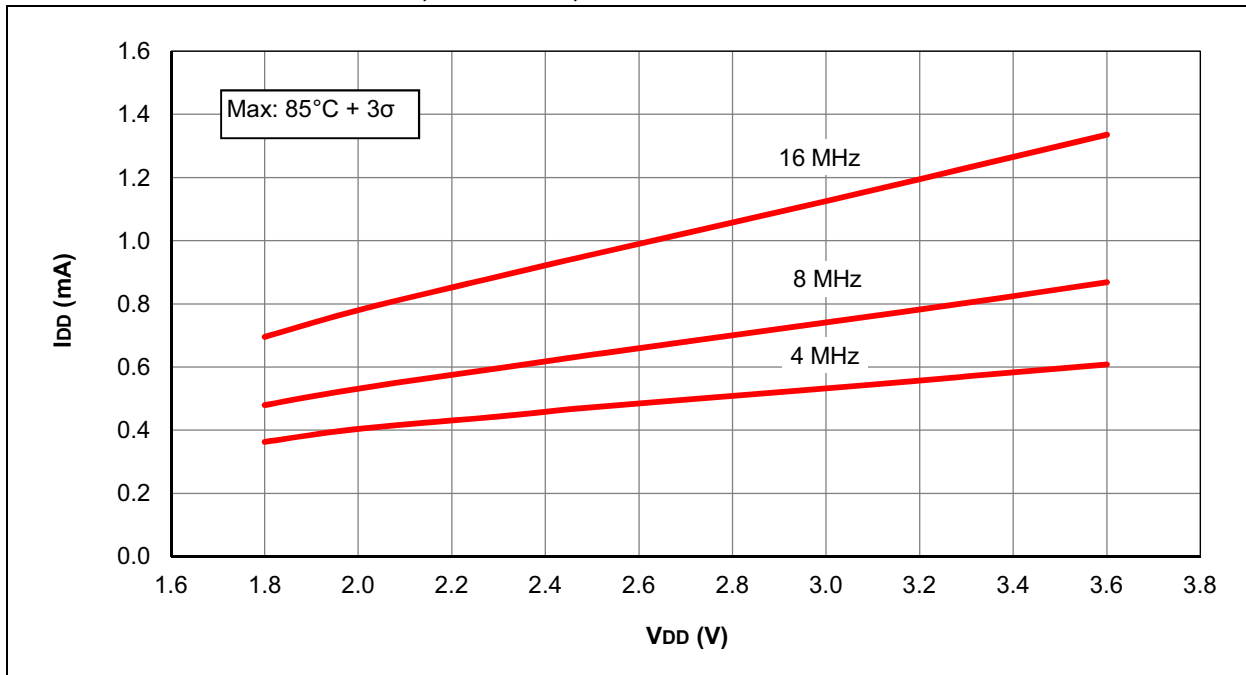
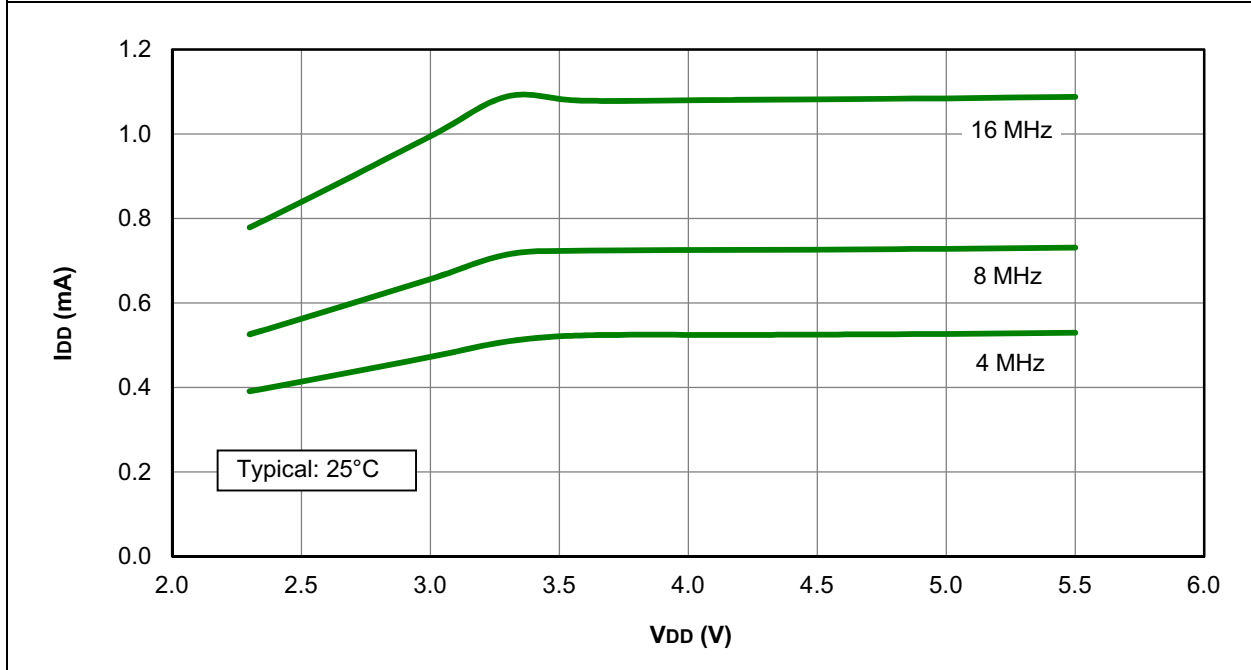


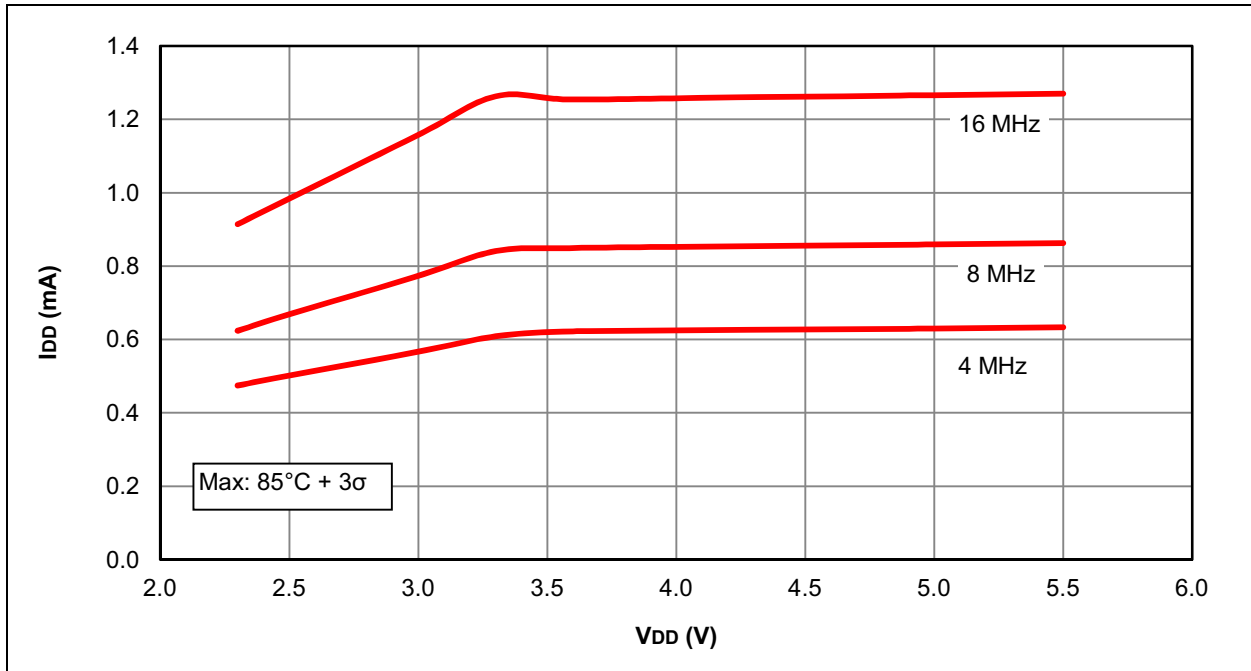
FIGURE 26-18: I<sub>DD</sub> MAXIMUM, HFINTOSC, PIC16LF1507 ONLY



**FIGURE 26-19: I<sub>DD</sub> TYPICAL, HFINTOSC, PIC16F1507 ONLY**

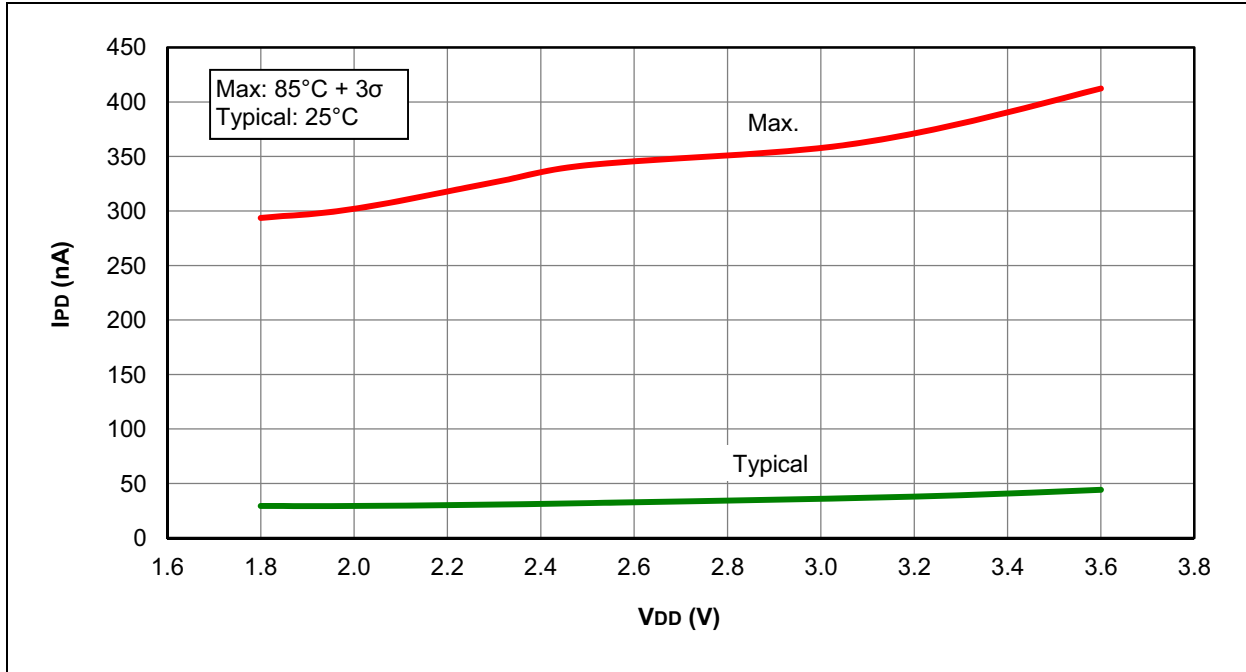


**FIGURE 26-20: I<sub>DD</sub> MAXIMUM, HFINTOSC, PIC16F1507 ONLY**

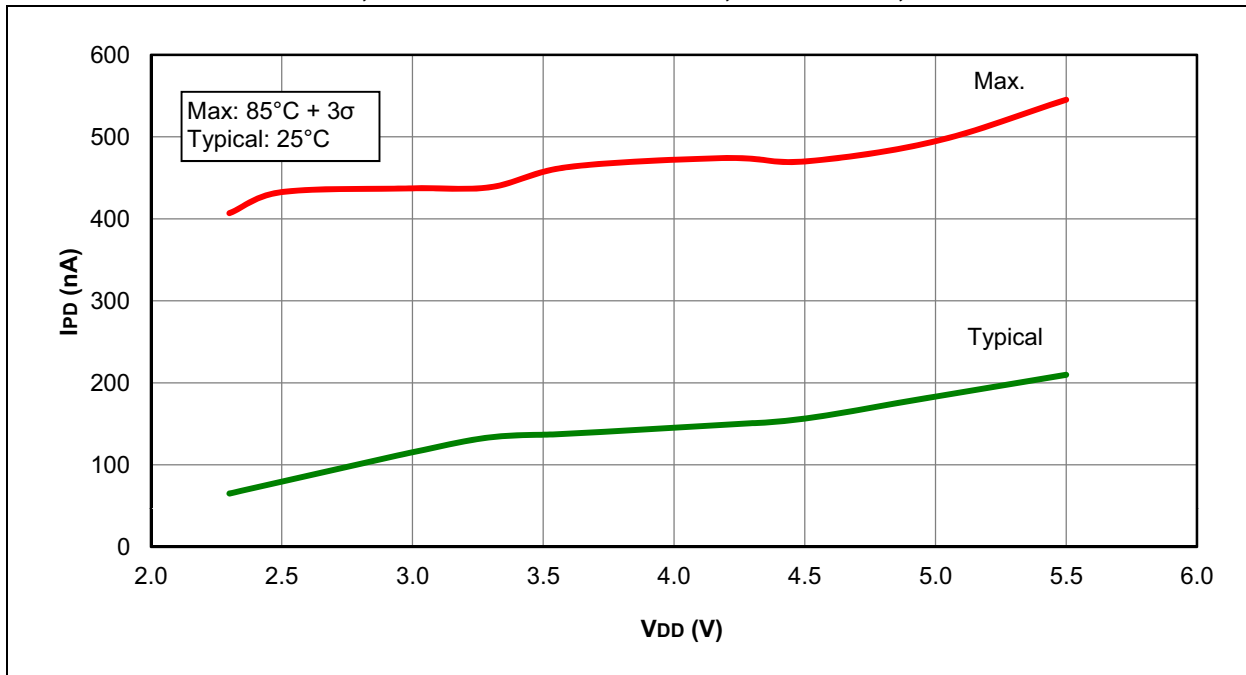


# PIC16(L)F1507

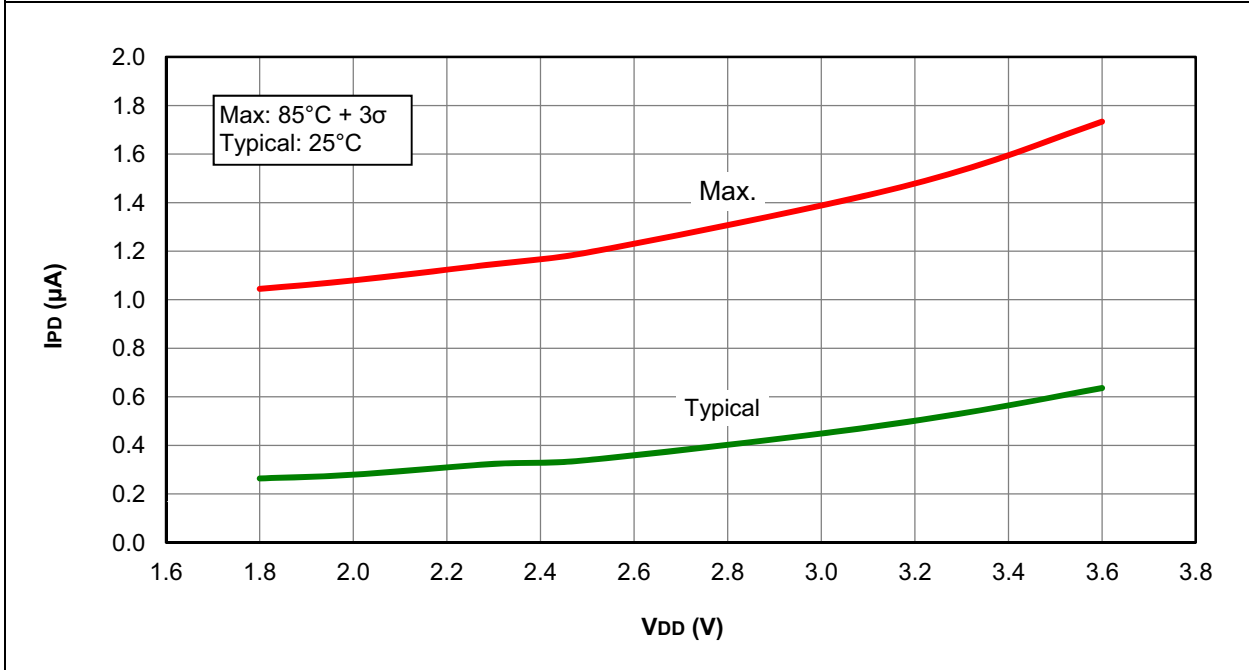
**FIGURE 26-21: IPD BASE, LOW-POWER SLEEP MODE, PIC16LF1507 ONLY**



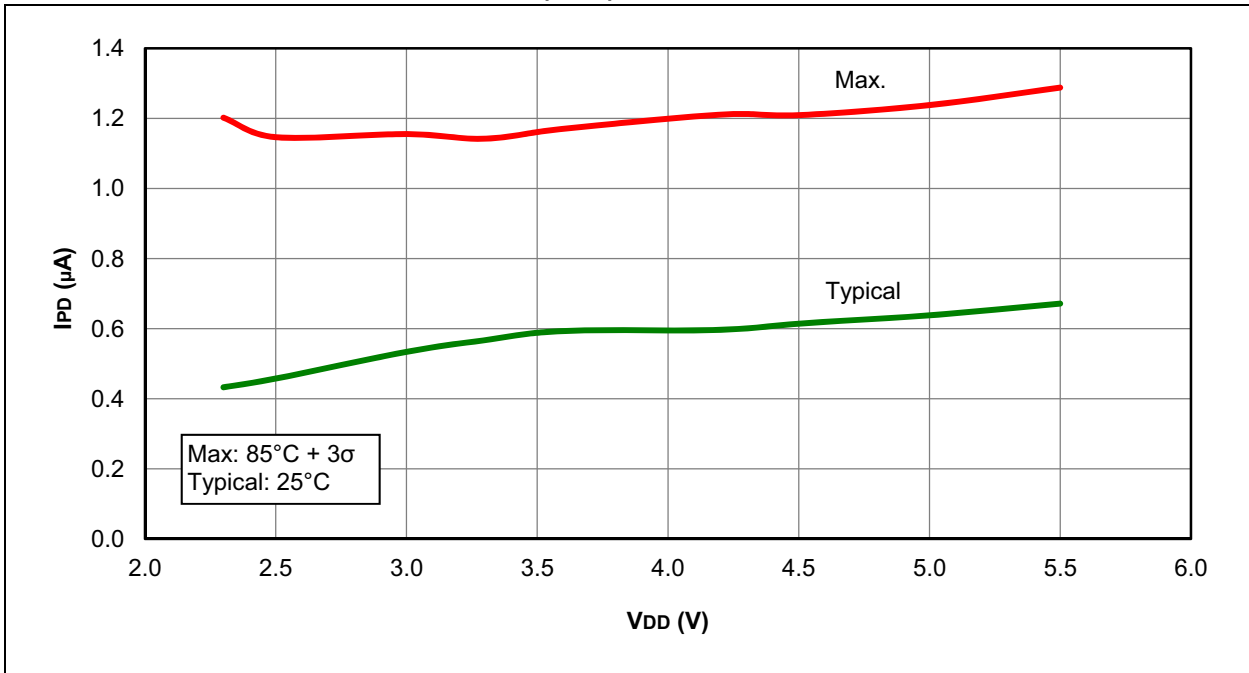
**FIGURE 26-22: IPD BASE, LOW-POWER SLEEP MODE, VREGPM = 1, PIC16F1507 ONLY**



**FIGURE 26-23: IPD, WATCHDOG TIMER (WDT), PIC16LF1507 ONLY**



**FIGURE 26-24: IPD, WATCHDOG TIMER (WDT), PIC16F1507 ONLY**



# PIC16(L)F1507

FIGURE 26-25: IPD, FIXED VOLTAGE REFERENCE (FVR), PIC16LF1507 ONLY

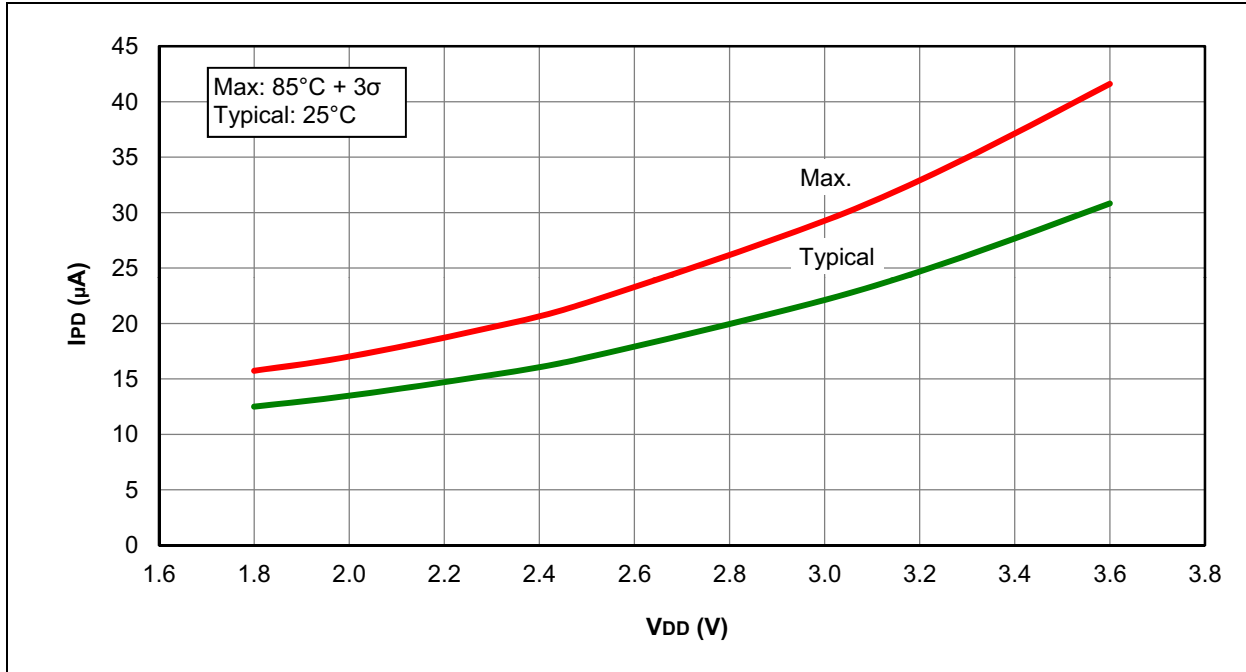
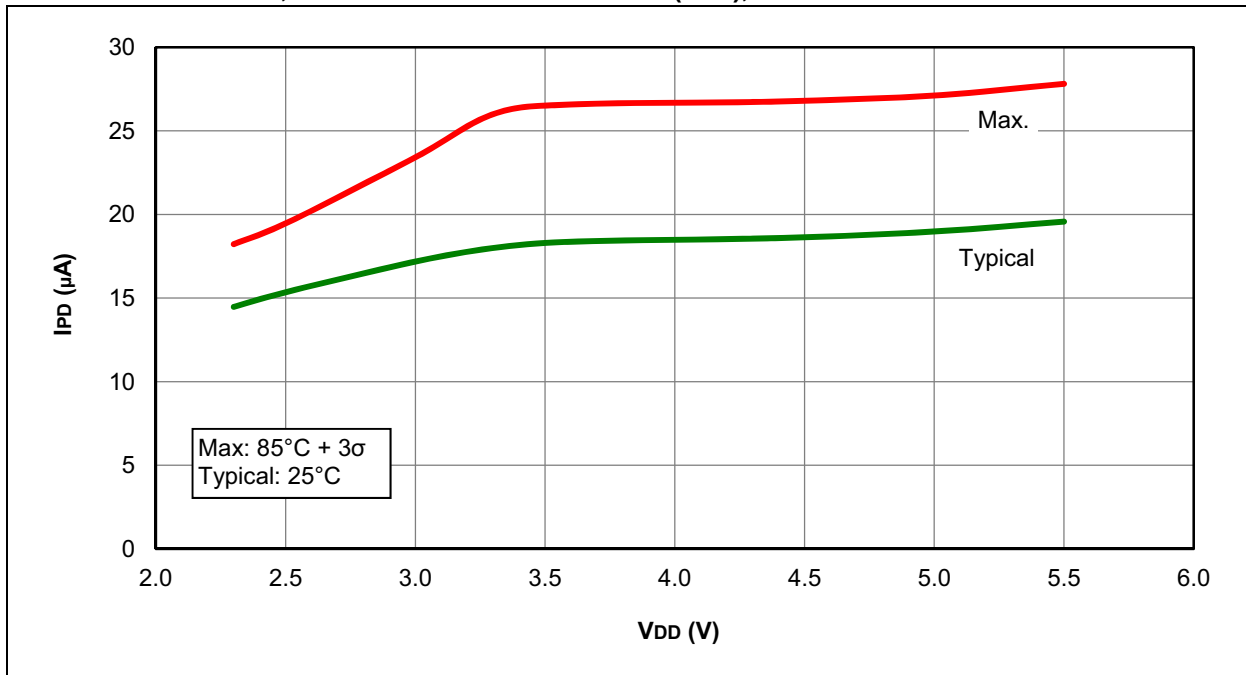
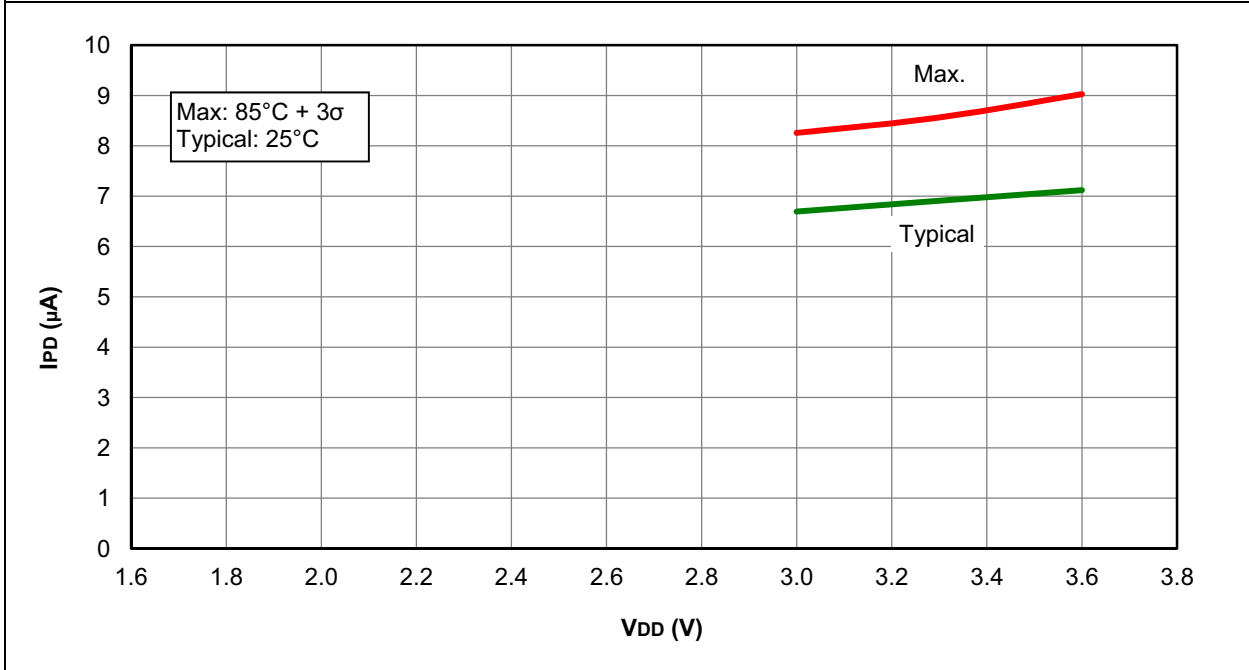


FIGURE 26-26: IPD, FIXED VOLTAGE REFERENCE (FVR), PIC16F1507 ONLY

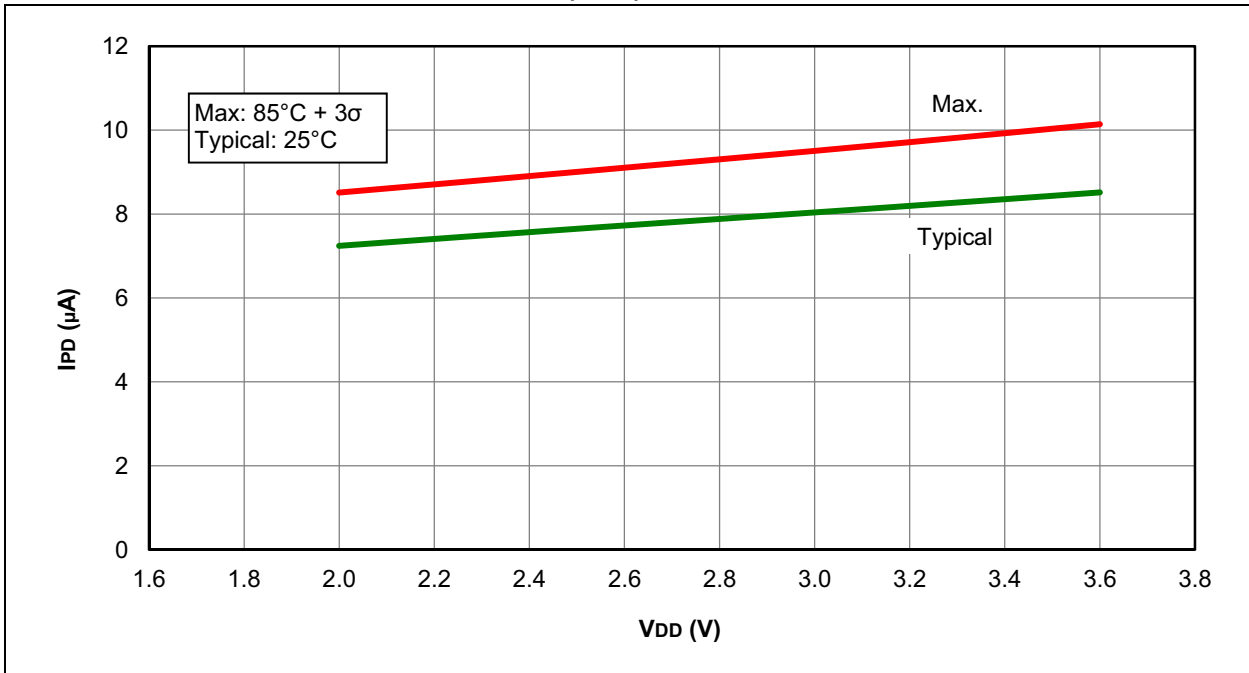




**FIGURE 26-27: IPD, BROWN-OUT RESET (BOR), BORV = 0, PIC16LF1507 ONLY**



**FIGURE 26-28: IPD, BROWN-OUT RESET (BOR), BORV = 1, PIC16LF1507 ONLY**



# PIC16(L)F1507

FIGURE 26-29: IPD, BROWN-OUT RESET (BOR), BORV = 0, PIC16F1507 ONLY

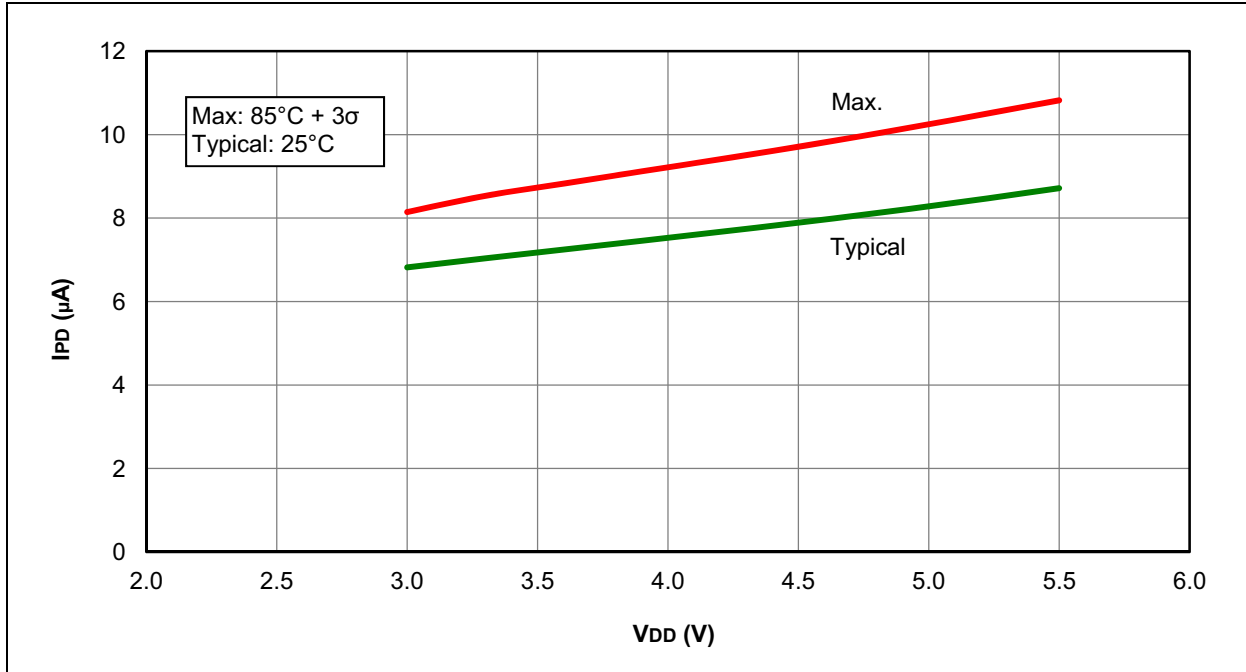
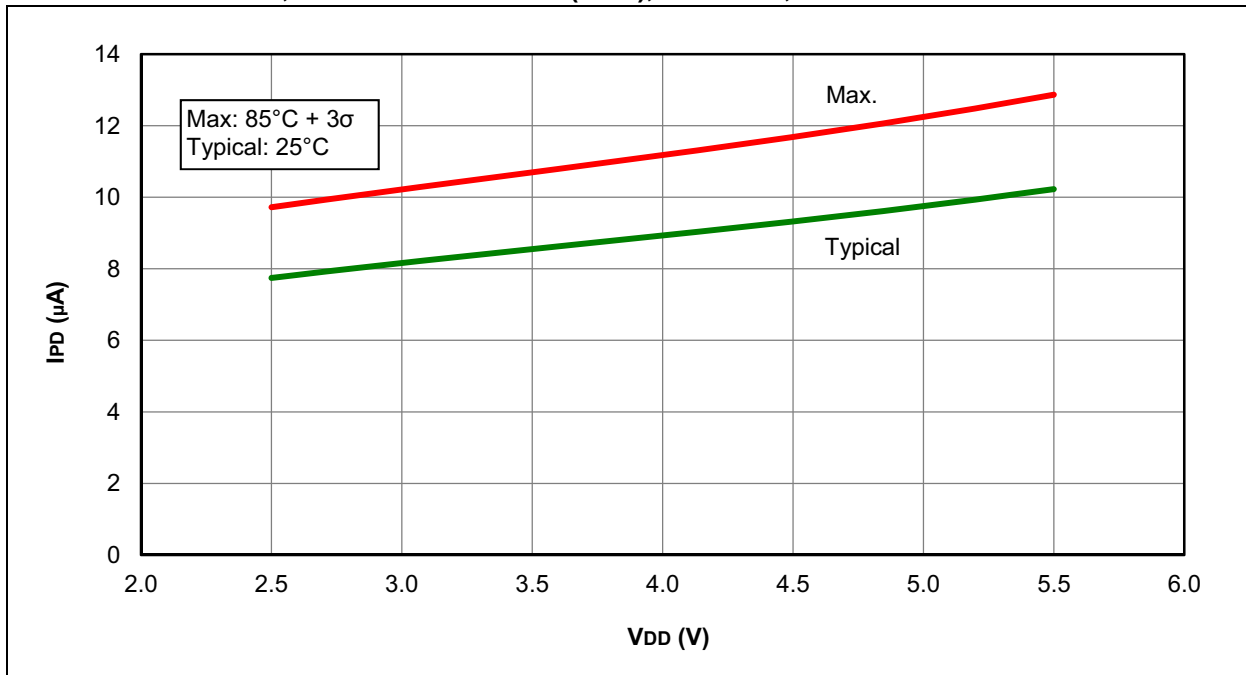
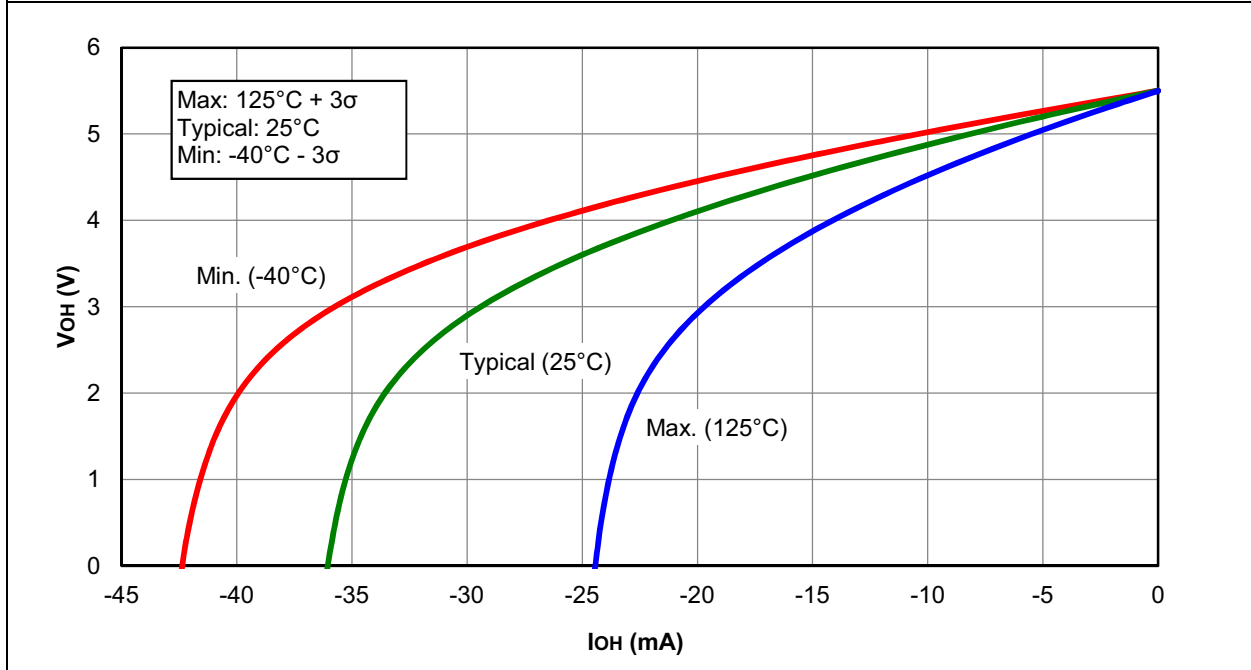


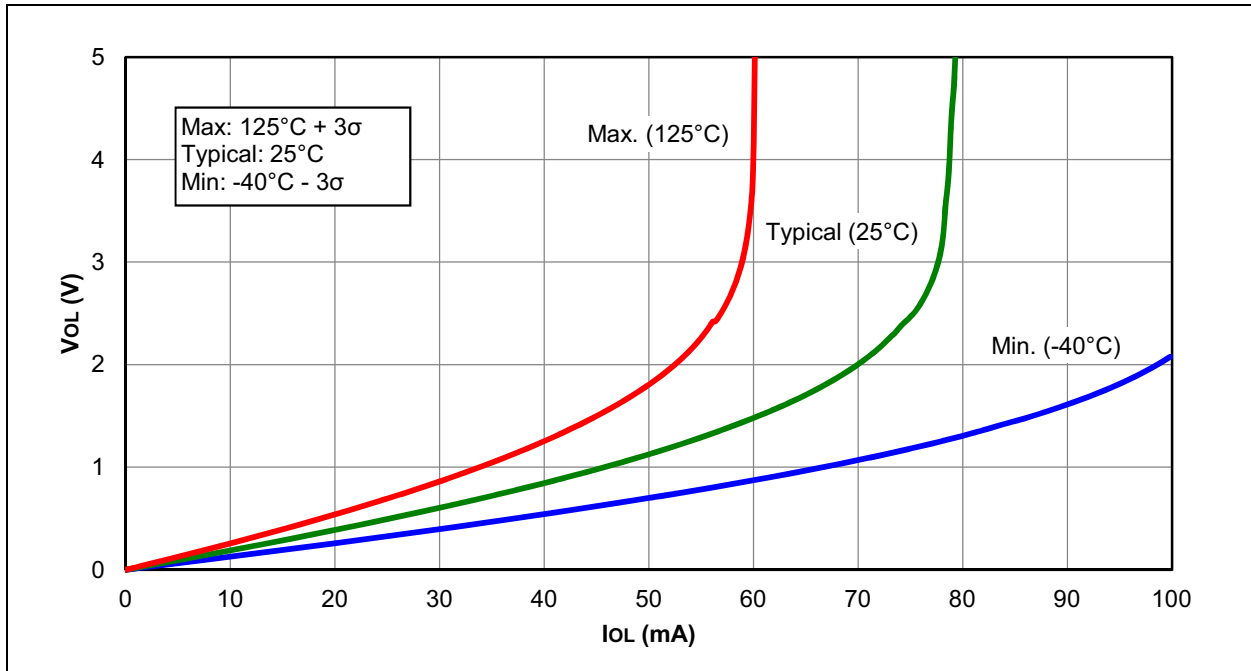
FIGURE 26-30: IPD, BROWN-OUT RESET (BOR), BORV = 1, PIC16F1507 ONLY



**FIGURE 26-31:  $V_{OH}$  vs.  $I_{OH}$  OVER TEMPERATURE,  $V_{DD} = 5.5V$ , PIC16F1507 ONLY**



**FIGURE 26-32:  $V_{OL}$  vs.  $I_{OL}$  OVER TEMPERATURE,  $V_{DD} = 5.5V$ , PIC16F1507 ONLY**



# PIC16(L)F1507

FIGURE 26-33:  $V_{OH}$  vs.  $I_{OH}$  OVER TEMPERATURE,  $V_{DD} = 3.0V$

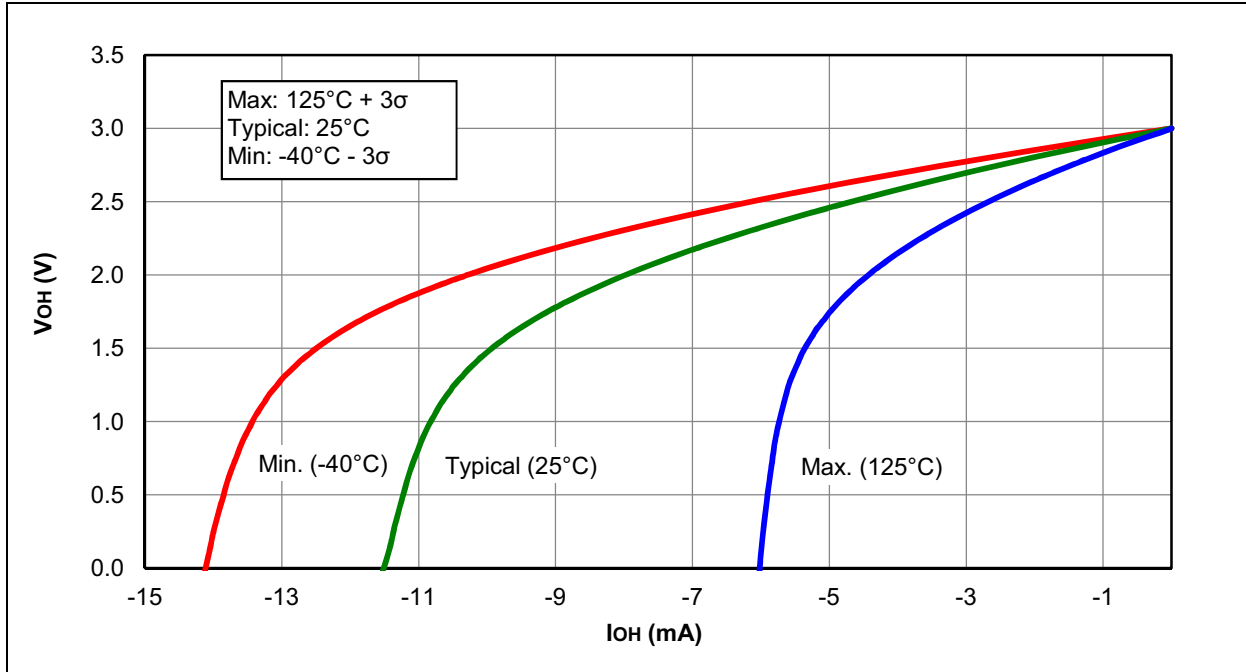
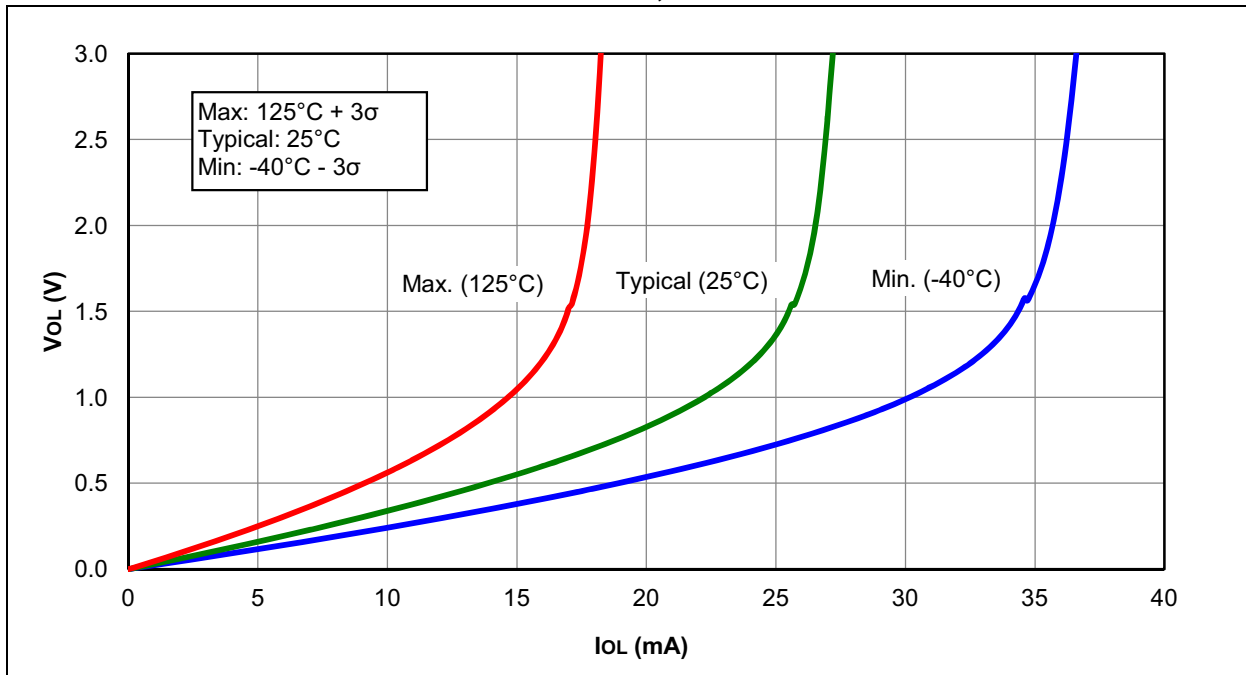
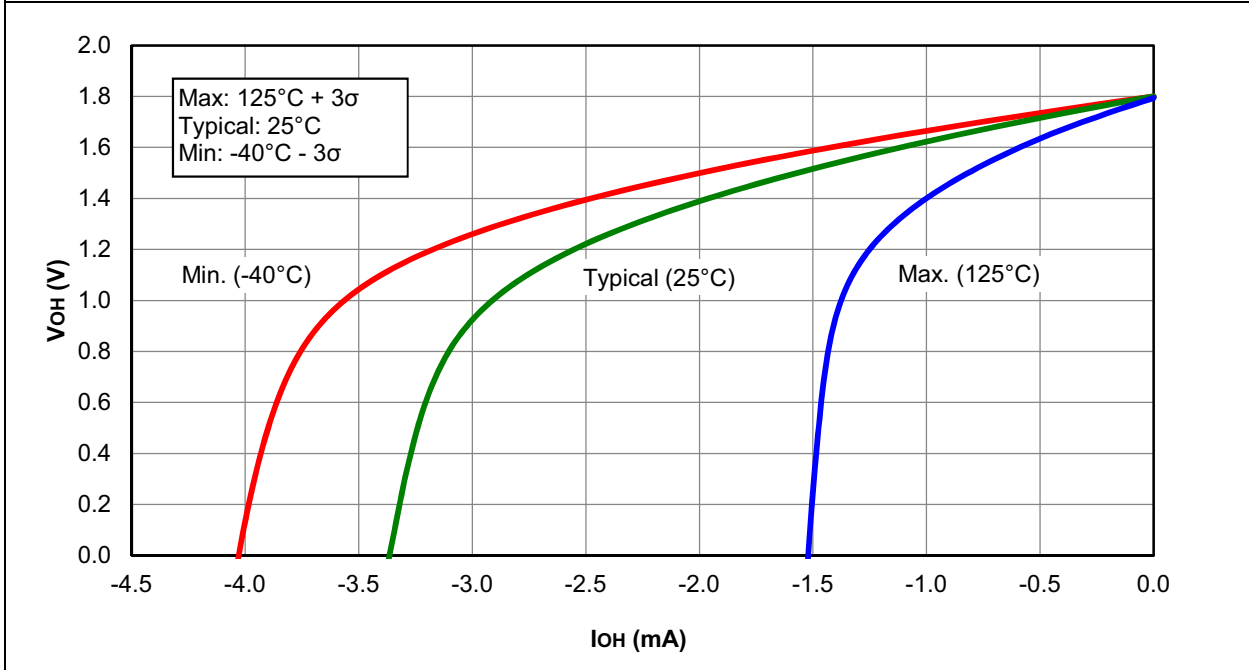


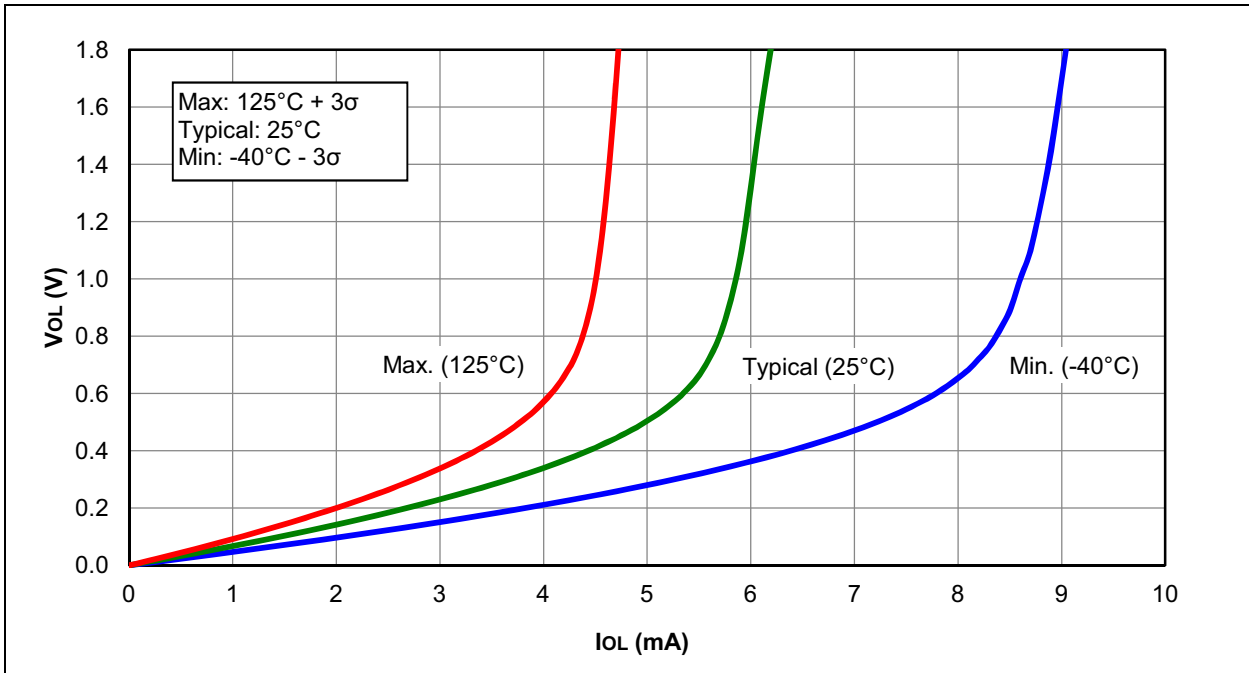
FIGURE 26-34:  $V_{OL}$  vs.  $I_{OL}$  OVER TEMPERATURE,  $V_{DD} = 3.0V$



**FIGURE 26-35:  $V_{OH}$  vs.  $I_{OH}$  OVER TEMPERATURE,  $V_{DD} = 1.8V$ , PIC16LF1507 ONLY**



**FIGURE 26-36:  $V_{OL}$  vs.  $I_{OL}$  OVER TEMPERATURE,  $V_{DD} = 1.8V$ , PIC16LF1507 ONLY**



# PIC16(L)F1507

FIGURE 26-37: POR RELEASE VOLTAGE

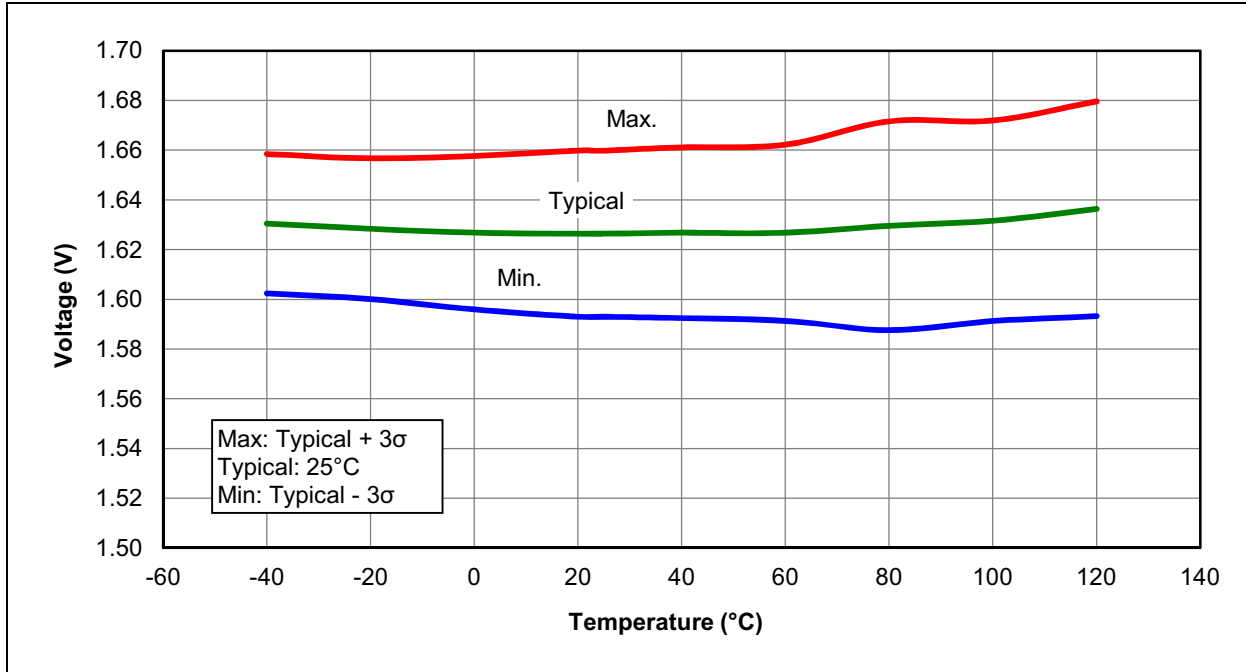
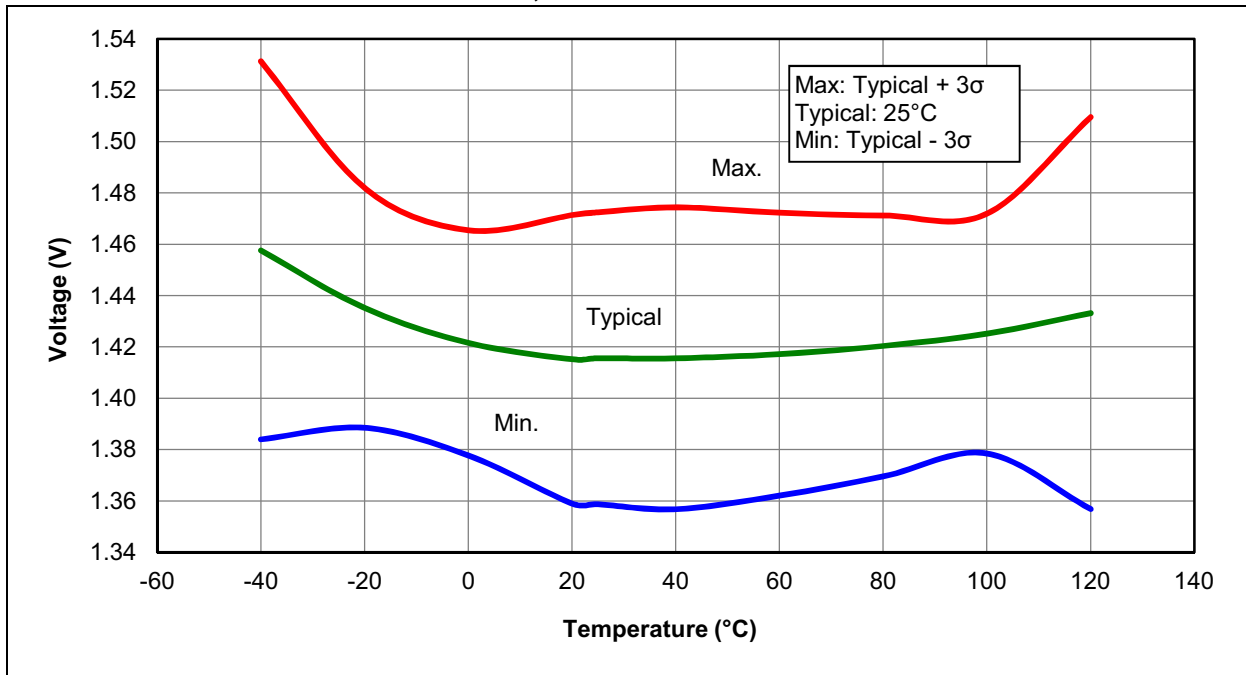
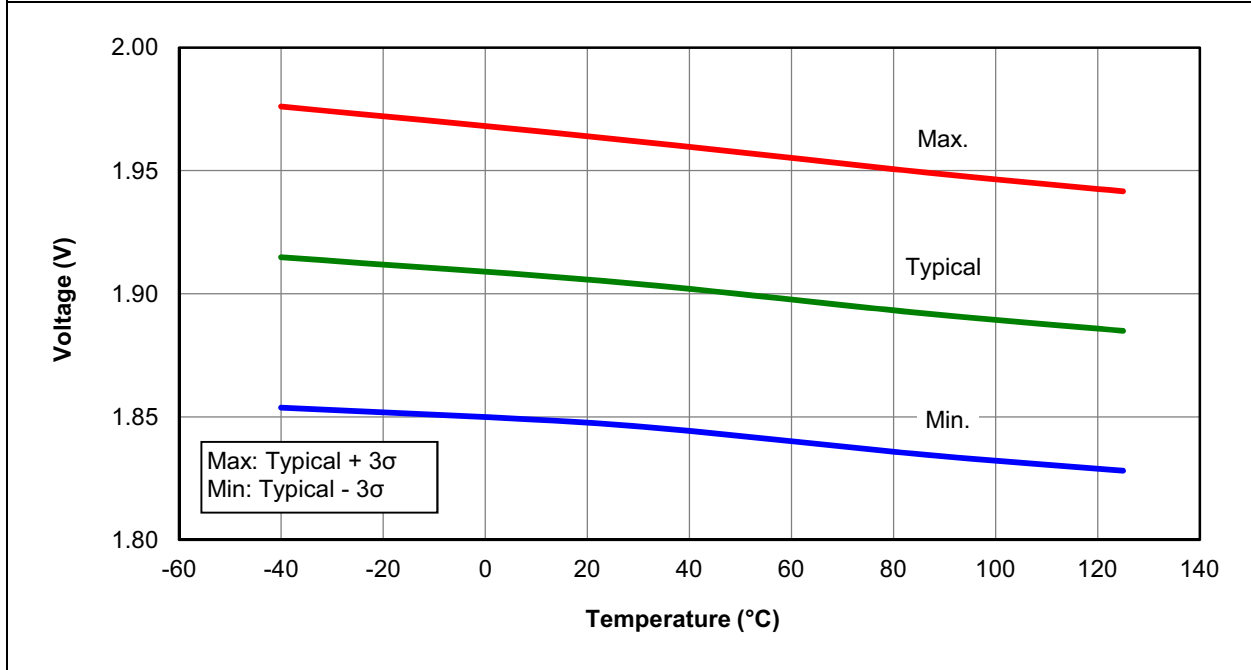


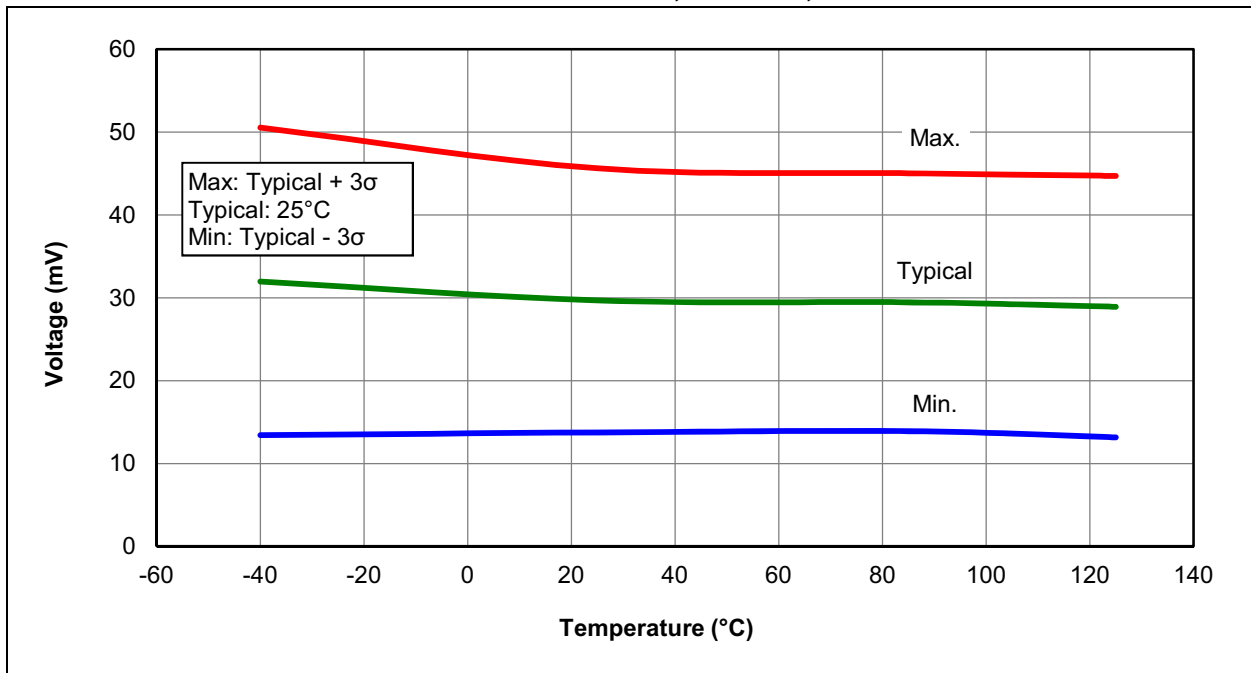
FIGURE 26-38: POR REARM VOLTAGE, PIC16F1507 ONLY



**FIGURE 26-39: BROWN-OUT RESET VOLTAGE, BORV = 1, PIC16LF1507 ONLY**



**FIGURE 26-40: BROWN-OUT RESET HYSTERESIS, BORV = 1, PIC16LF1507 ONLY**



# PIC16(L)F1507

FIGURE 26-41: BROWN-OUT RESET VOLTAGE, BORV = 1, PIC16F1507 ONLY

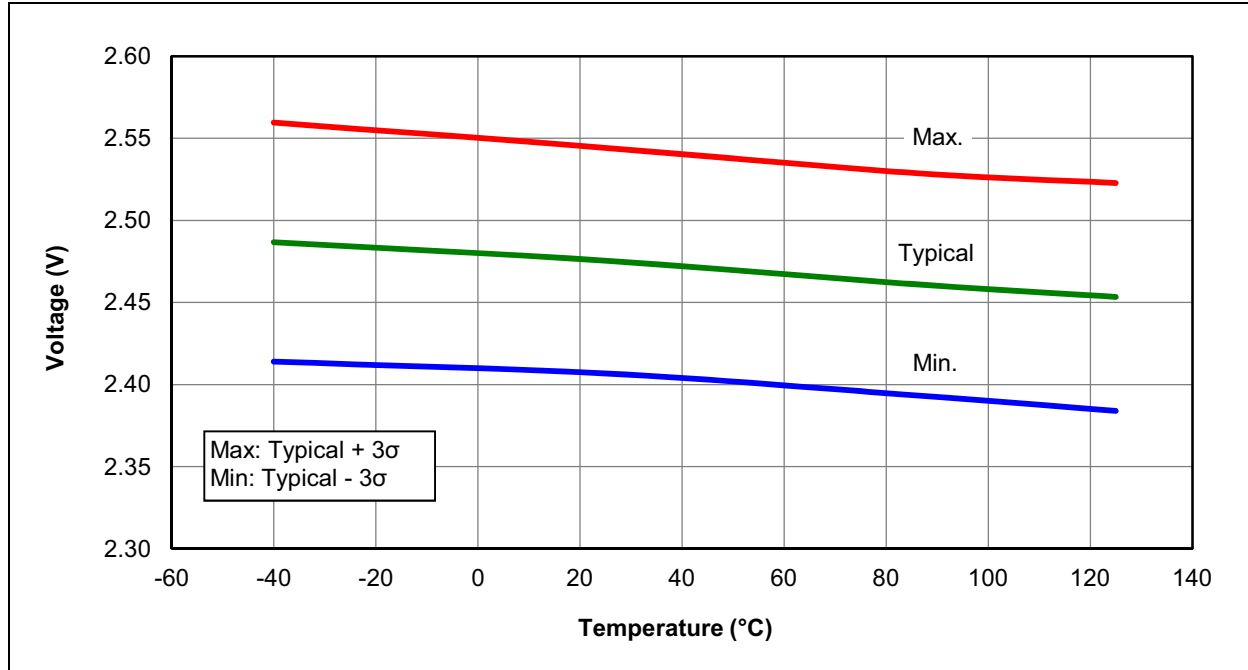


FIGURE 26-42: BROWN-OUT RESET HYSTERESIS, BORV = 1, PIC16F1507 ONLY

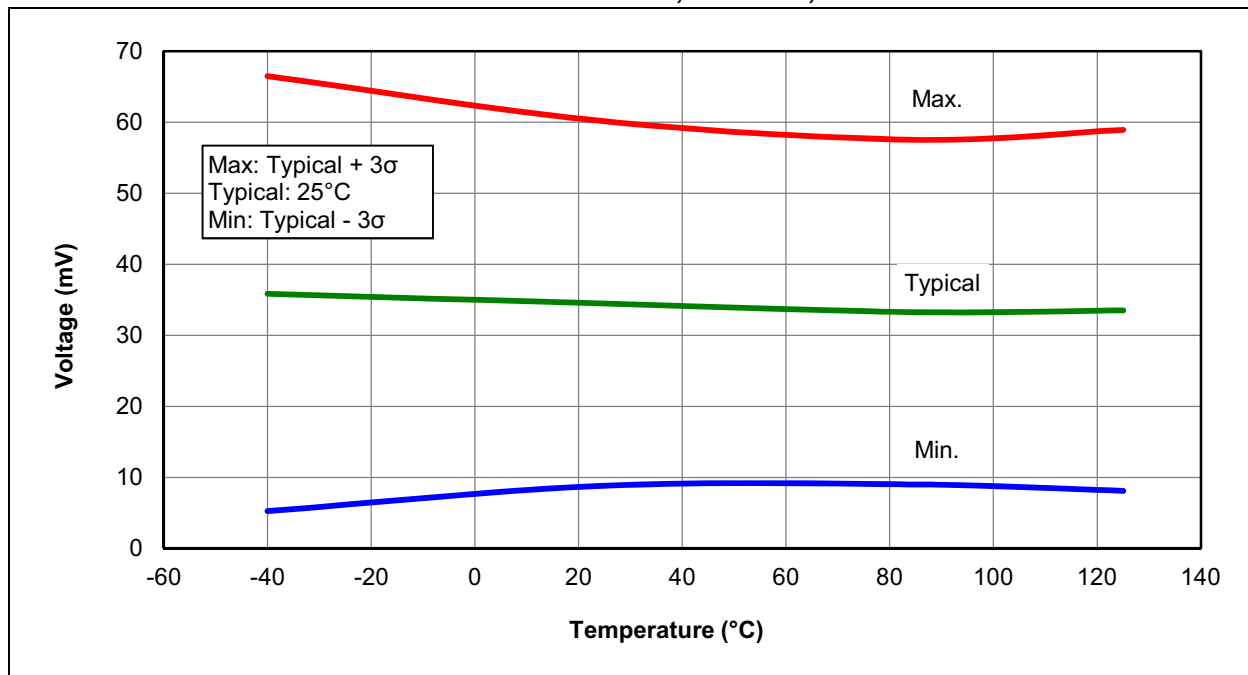
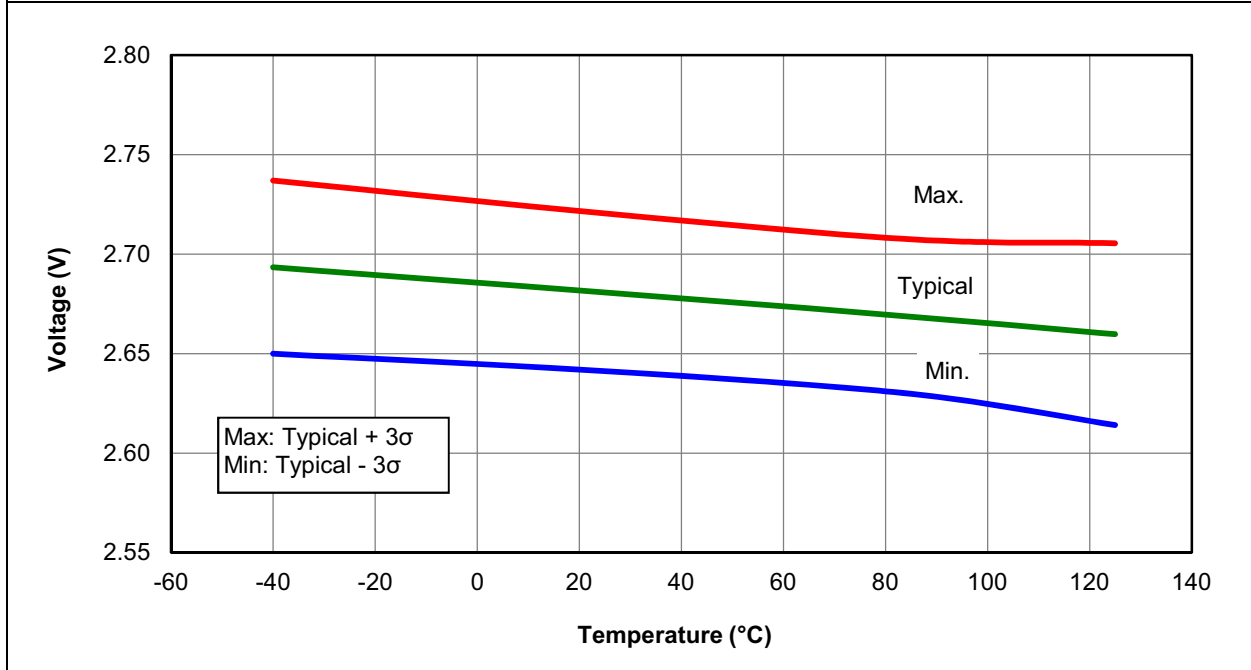




FIGURE 26-43: BROWN-OUT RESET VOLTAGE, BORV = 0



# PIC16(L)F1507

FIGURE 26-44: LOW-POWER BROWN-OUT RESET VOLTAGE, LPBOR = 0

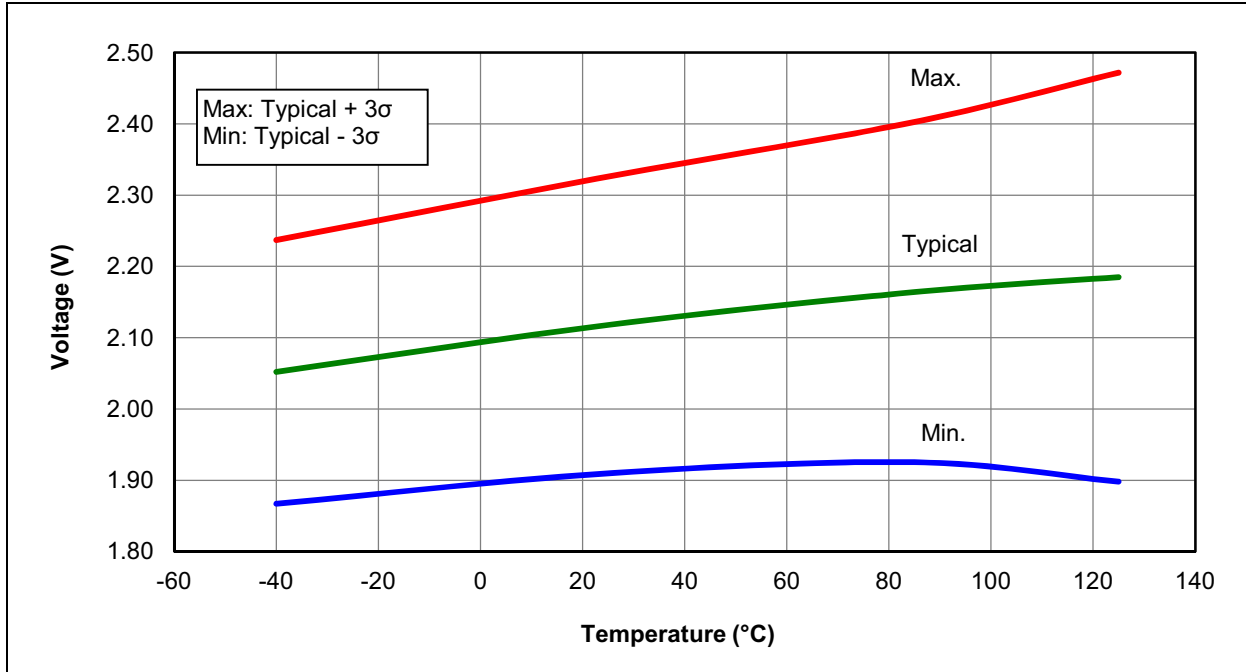
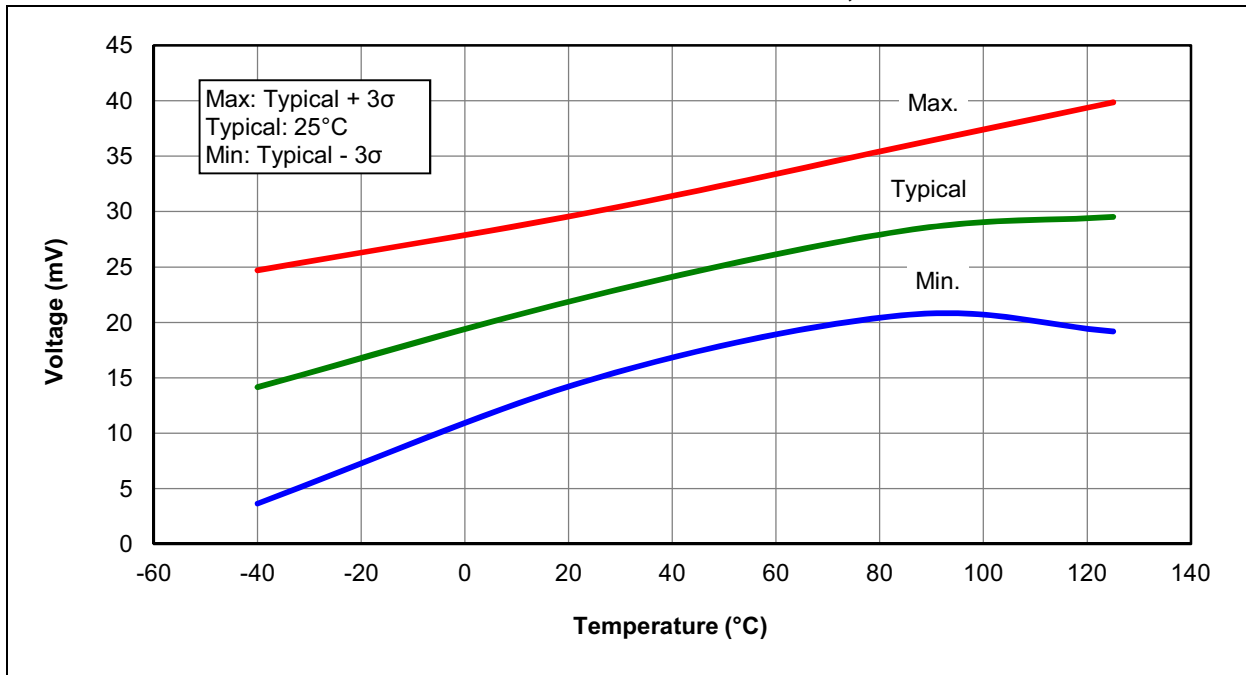
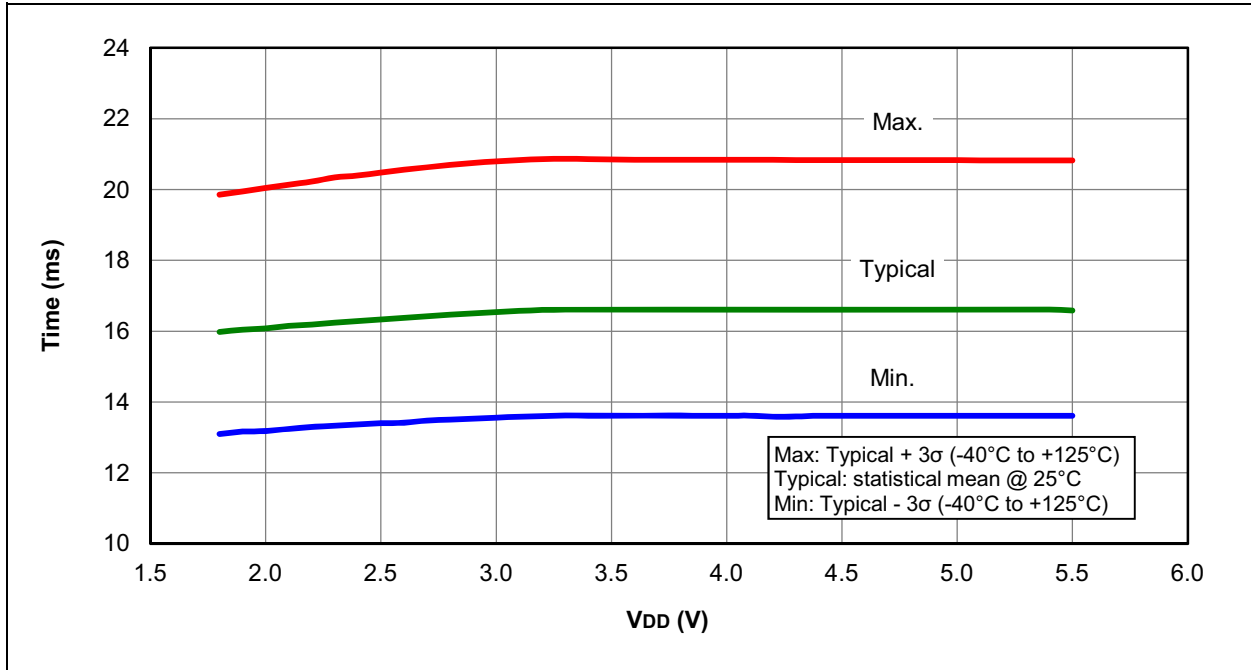


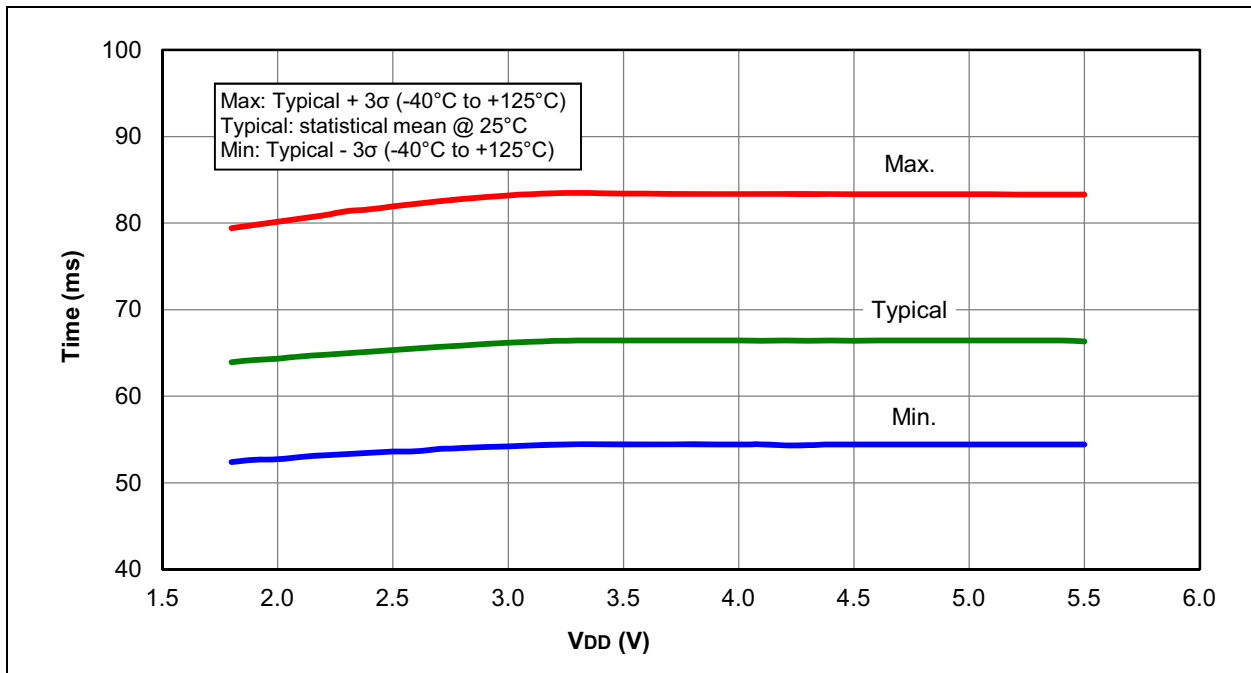
FIGURE 26-45: LOW-POWER BROWN-OUT RESET HYSTERESIS, LPBOR = 0



**FIGURE 26-46: WDT TIME-OUT PERIOD**

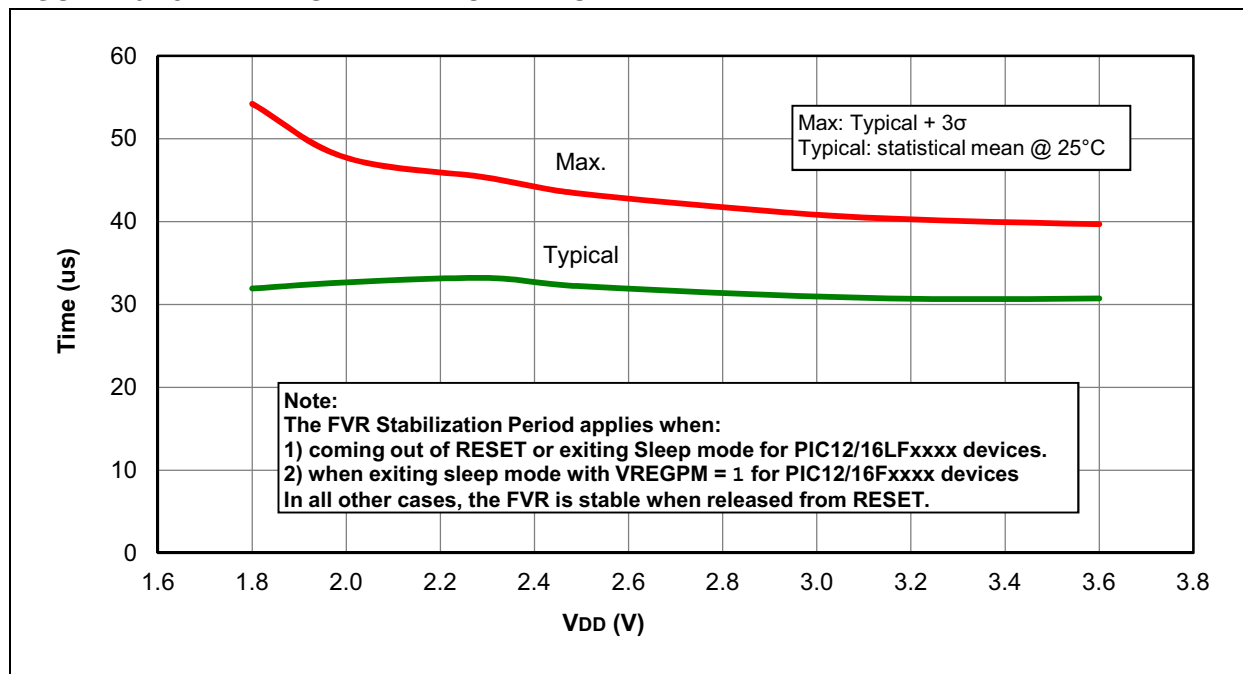


**FIGURE 26-47: PWRT PERIOD**

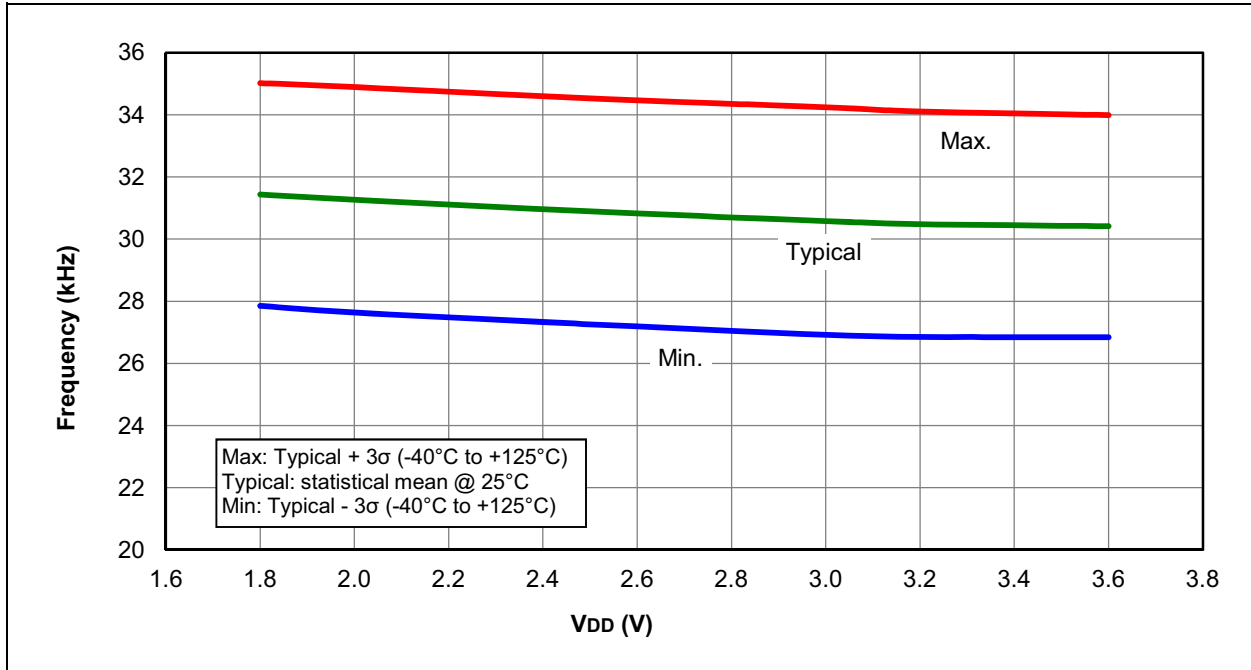


# PIC16(L)F1507

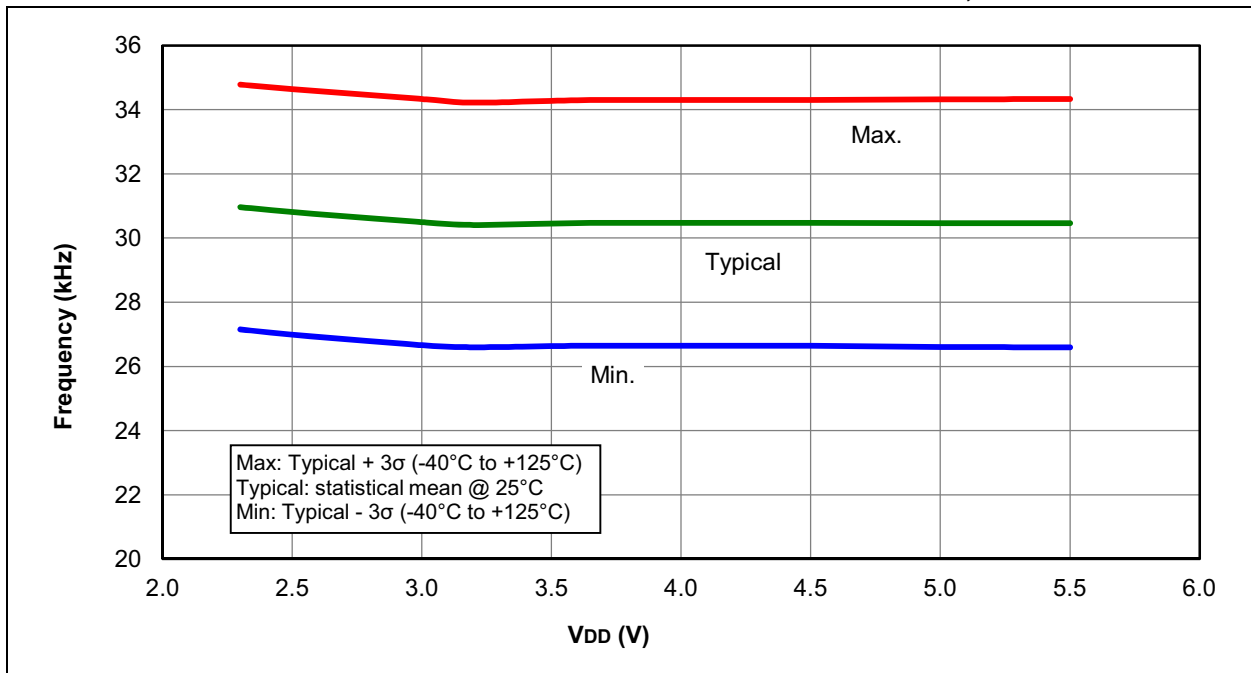
FIGURE 26-48: FVR STABILIZATION PERIOD



**FIGURE 26-49: LFINTOSC FREQUENCY OVER V<sub>DD</sub> AND TEMPERATURE, PIC16LF1507 ONLY**

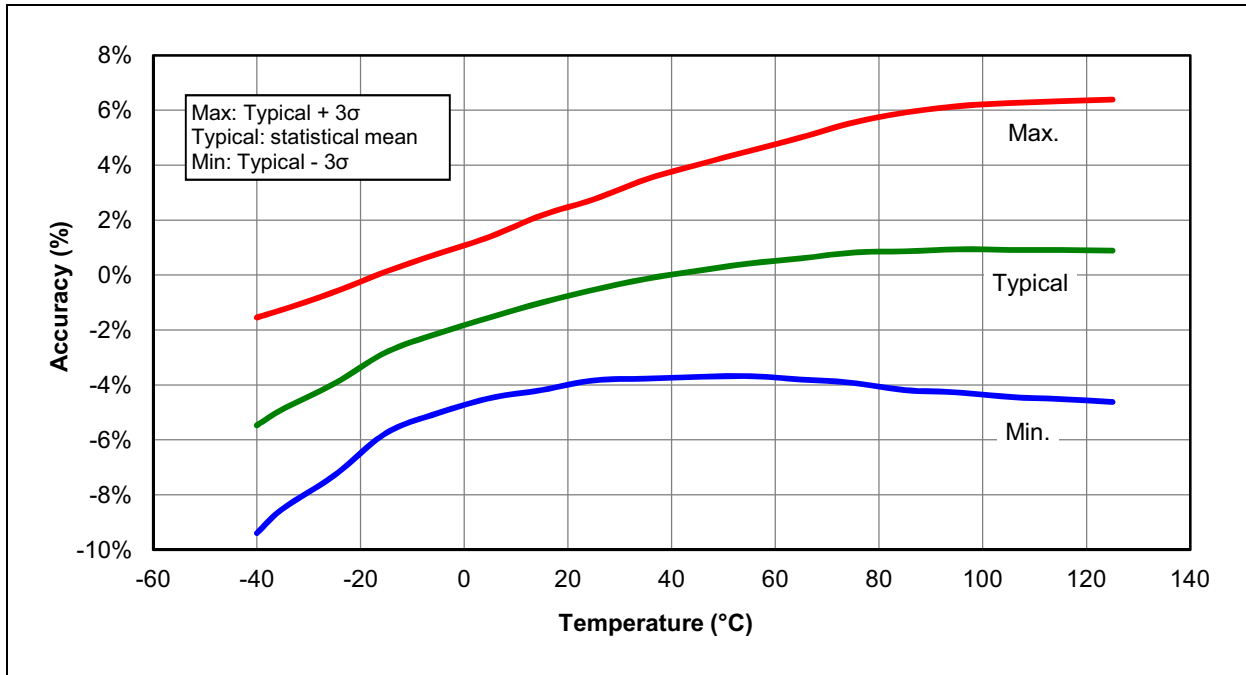


**FIGURE 26-50: LFINTOSC FREQUENCY OVER V<sub>DD</sub> AND TEMPERATURE, PIC16F1507 ONLY**

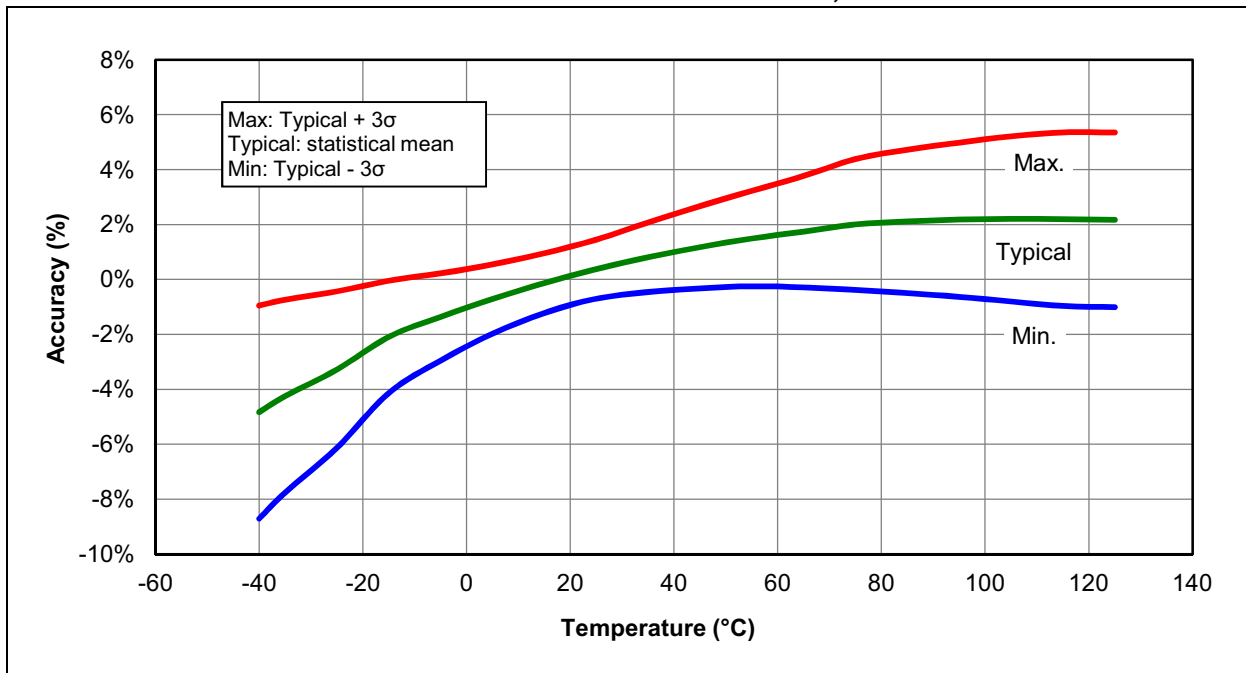


# PIC16(L)F1507

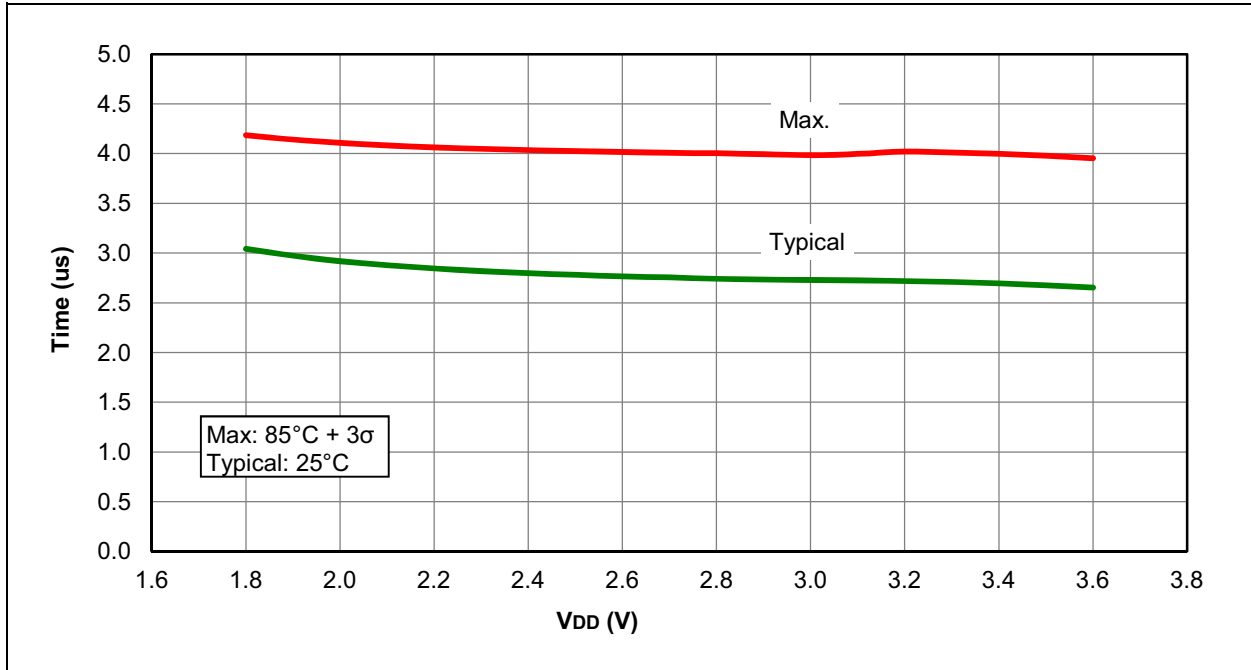
**FIGURE 26-51: HFINTOSC ACCURACY OVER TEMPERATURE,  $V_{DD} = 1.8V$ , PIC16LF1507 ONLY**



**FIGURE 26-52: HFINTOSC ACCURACY OVER TEMPERATURE,  $2.3V \leq V_{DD} \leq 5.5V$**

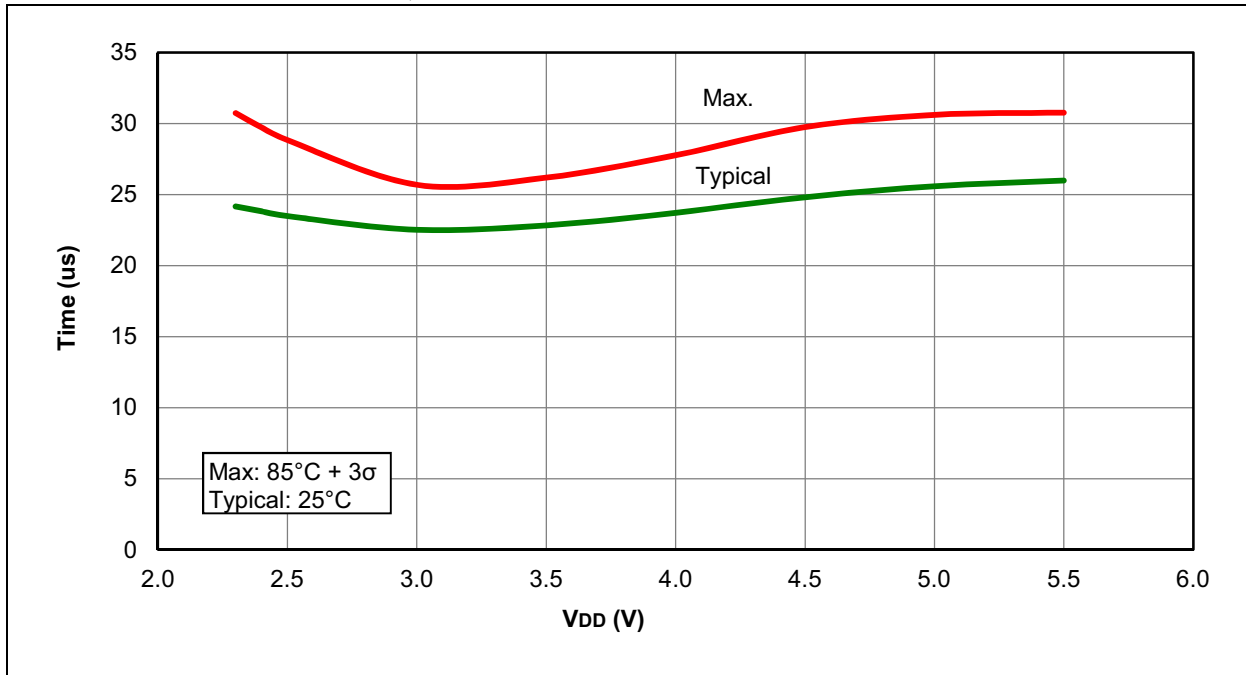


**FIGURE 26-53: SLEEP MODE, WAKE PERIOD WITH HFINTOSC SOURCE, PIC16LF1507 ONLY**

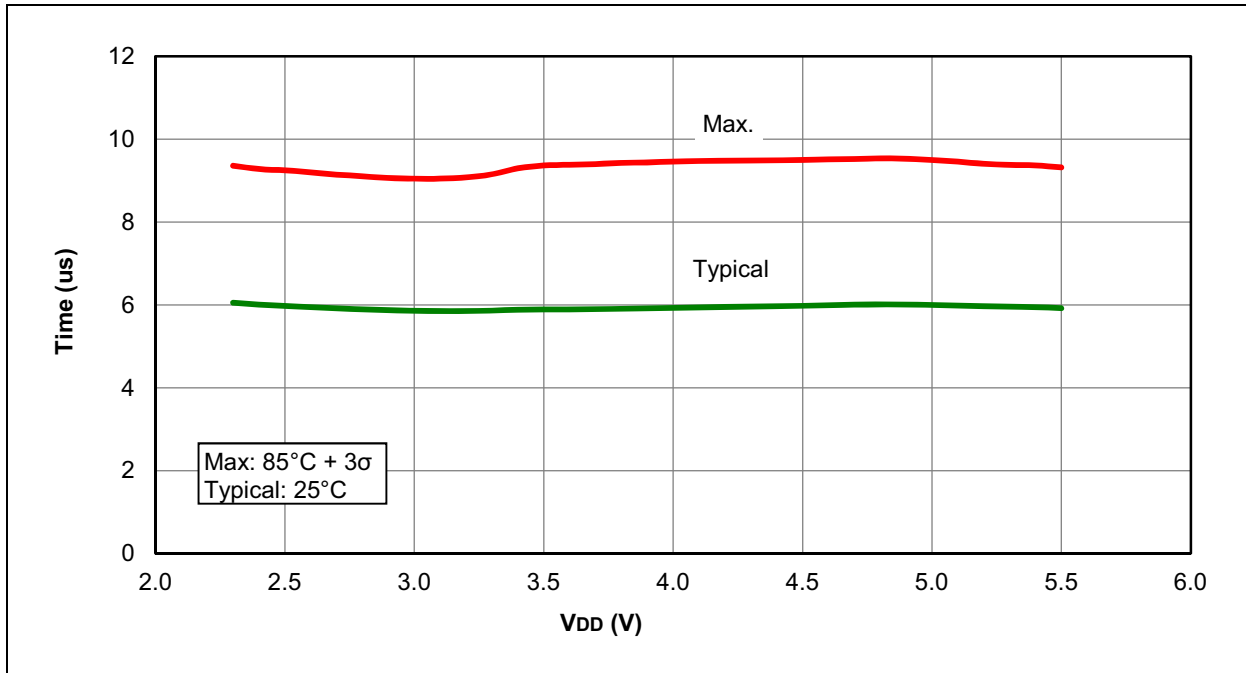


# PIC16(L)F1507

**FIGURE 26-54: LOW-POWER SLEEP MODE, WAKE PERIOD WITH HFINTOSC SOURCE, VREGPM = 1, PIC16F1507 ONLY**



**FIGURE 26-55: SLEEP MODE, WAKE PERIOD WITH HFINTOSC SOURCE, VREGPM = 0, PIC16F1507 ONLY**





## 27.0 DEVELOPMENT SUPPORT

The PIC<sup>®</sup> microcontrollers (MCU) and dsPIC<sup>®</sup> digital signal controllers (DSC) are supported with a full range of software and hardware development tools:

- Integrated Development Environment
  - MPLAB<sup>®</sup> X IDE Software
- Compilers/Assemblers/Linkers
  - MPLAB XC Compiler
  - MPASM<sup>™</sup> Assembler
  - MPLINK<sup>™</sup> Object Linker/  
MPLIB<sup>™</sup> Object Librarian
  - MPLAB Assembler/Linker/Librarian for  
Various Device Families
- Simulators
  - MPLAB X SIM Software Simulator
- Emulators
  - MPLAB REAL ICE<sup>™</sup> In-Circuit Emulator
- In-Circuit Debuggers/Programmers
  - MPLAB ICD 3
  - PICKit<sup>™</sup> 3
- Device Programmers
  - MPLAB PM3 Device Programmer
- Low-Cost Demonstration/Development Boards,  
Evaluation Kits and Starter Kits
- Third-party development tools

## 27.1 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<sup>®</sup>, Linux and Mac OS<sup>®</sup> 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

## 27.2 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

## 27.3 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

## 27.4 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

## 27.5 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

## 27.6 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.

## 27.7 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.

## 27.8 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.

## 27.9 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™).

## 27.10 MPLAB PM3 Device Programmer

The MPLAB PM3 Device Programmer is a universal, CE compliant device programmer with programmable voltage verification at VDDMIN and VDDMAX 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.

## 27.11 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](http://www.microchip.com)) for the complete list of demonstration, development and evaluation kits.

## 27.12 Third-Party Development Tools

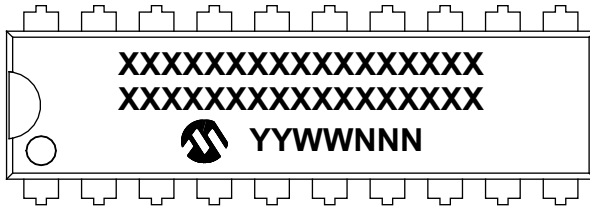
Microchip also offers a great collection of tools from third-party vendors. These tools are carefully selected to offer good value and unique functionality.

- 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®

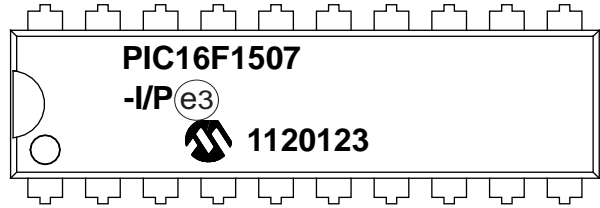
## 28.0 PACKAGING INFORMATION

### 28.1 Package Marking Information

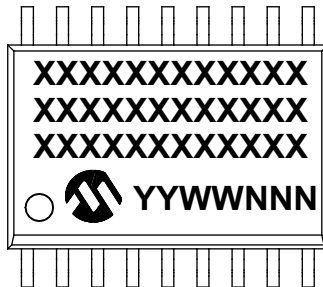
20-Lead PDIP (300 mil)



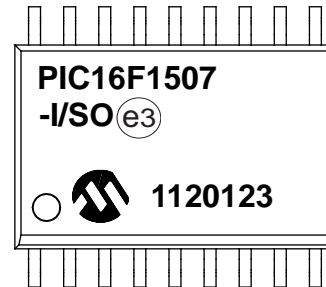
Example



20-Lead SOIC (7.50 mm)



Example



<b>Legend:</b>	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.

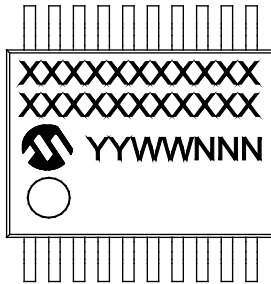
**Note:** In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.

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

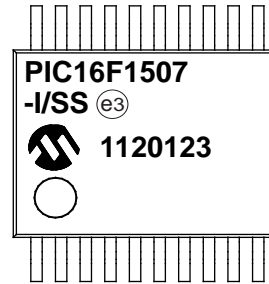
# PIC16(L)F1507

## Package Marking Information (Continued)

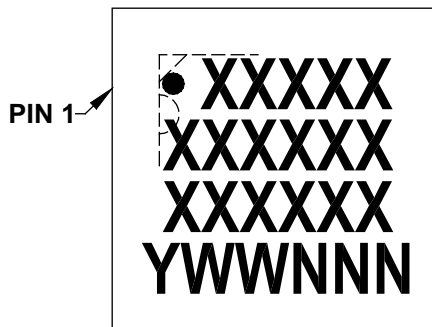
20-Lead SSOP (5.30 mm)



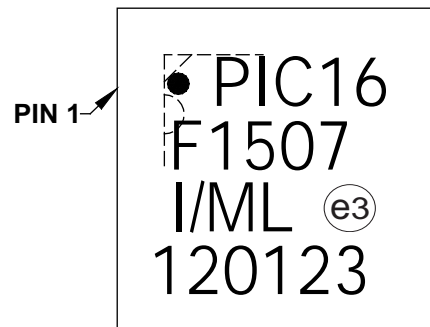
Example



20-Lead QFN (4x4x0.9 mm)  
20-Lead UQFN (4x4x0.5 mm)



Example



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

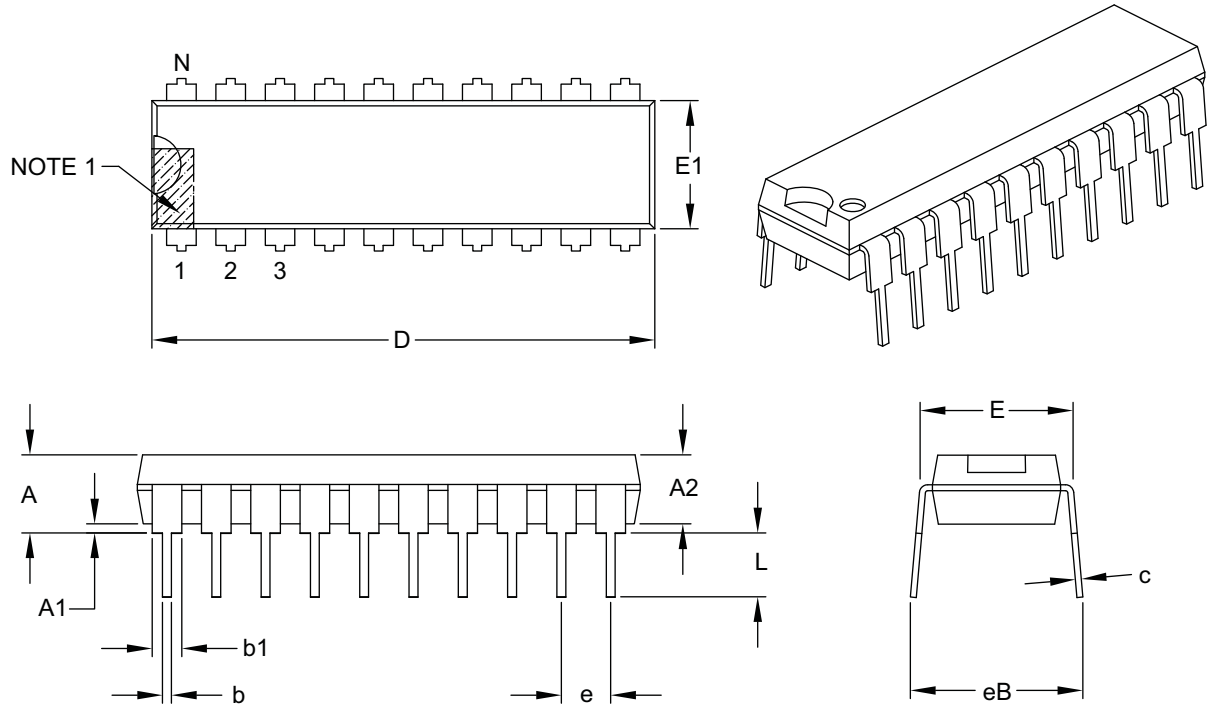
\* Standard PICmicro® device marking consists of Microchip part number, year code, week code and traceability code. For PICmicro device marking beyond this, certain price adders apply. Please check with your Microchip Sales Office. For QTP devices, any special marking adders are included in QTP price.

## 28.2 Package Details

The following sections give the technical details of the packages.

### 20-Lead Plastic Dual In-Line (P) – 300 mil Body [PDIP]

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



Dimension Limits	Units	INCHES		
		MIN	NOM	MAX
Number of Pins	N	20		
Pitch	e	.100 BSC		
Top to Seating Plane	A	–	–	.210
Molded Package Thickness	A2	.115	.130	.195
Base to Seating Plane	A1	.015	–	–
Shoulder to Shoulder Width	E	.300	.310	.325
Molded Package Width	E1	.240	.250	.280
Overall Length	D	.980	1.030	1.060
Tip to Seating Plane	L	.115	.130	.150
Lead Thickness	c	.008	.010	.015
Upper Lead Width	b1	.045	.060	.070
Lower Lead Width	b	.014	.018	.022
Overall Row Spacing §	eB	–	–	.430

**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.

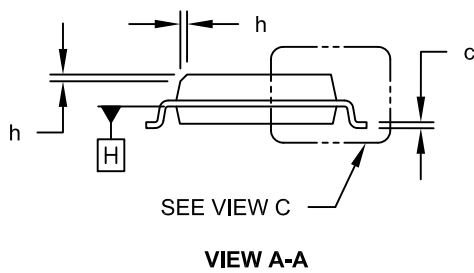
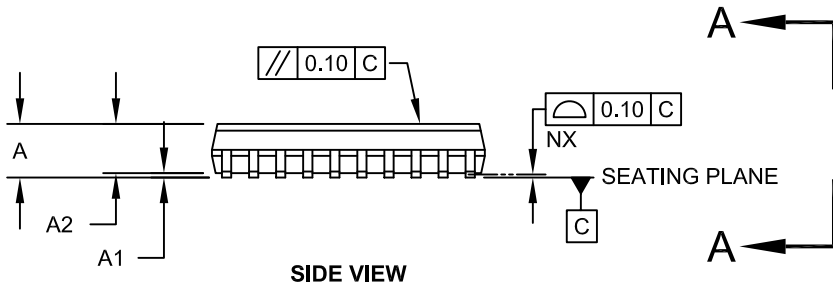
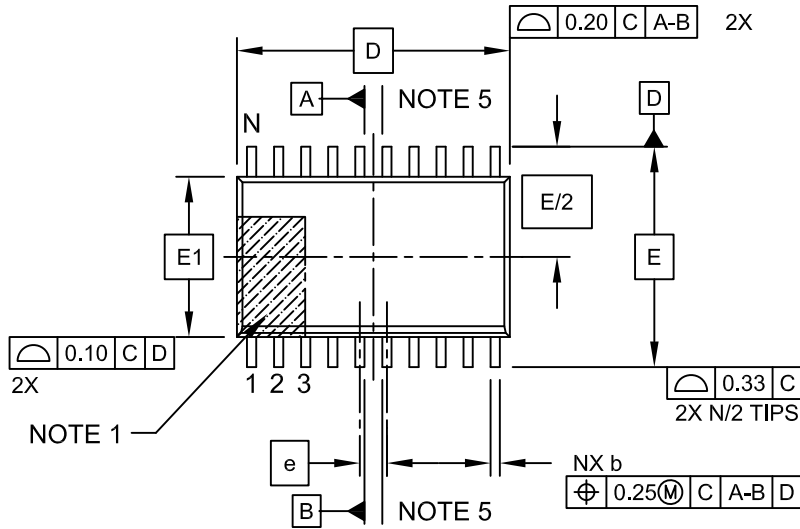
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-019B

# PIC16(L)F1507

## 20-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>

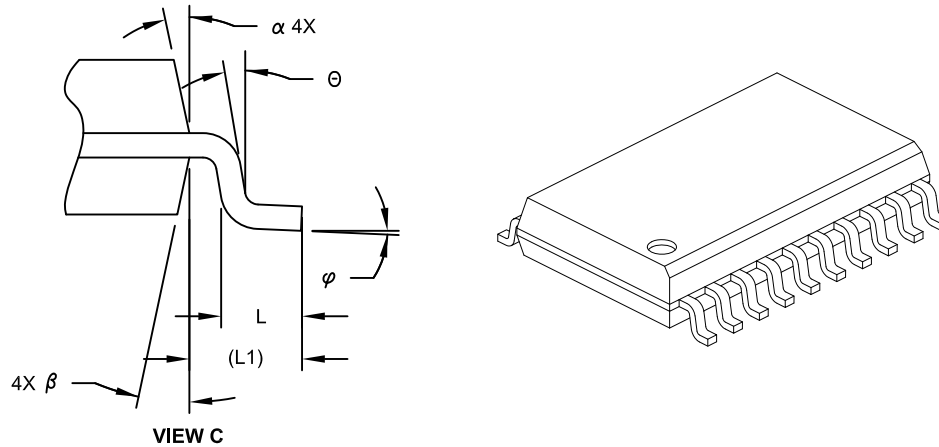


Microchip Technology Drawing C04-094C Sheet 1 of 2



## 20-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>



Dimension Limits	Units	MILLIMETERS		
		MIN	NOM	MAX
Number of Pins	N	20		
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	12.80 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	0.20	-	0.33
Lead Width	b	0.31	-	0.51
Mold Draft Angle Top	α	5°	-	15°
Mold Draft Angle Bottom	β	5°	-	15°

### Notes:

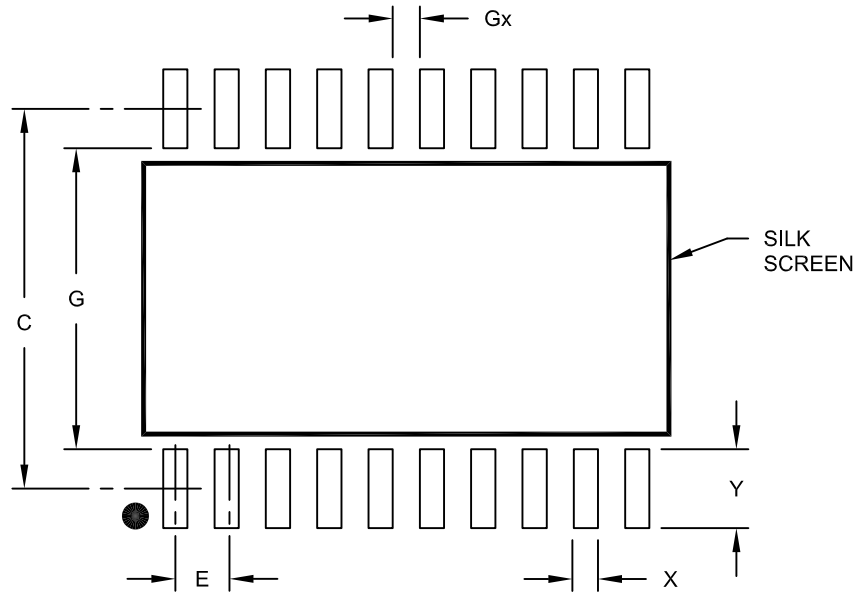
- Pin 1 visual index feature may vary, but must be located within the hatched area.
- § Significant Characteristic
- Dimension D does not include mold flash, protrusions or gate burrs, which shall not exceed 0.15 mm per end. Dimension E1 does not include interlead flash or protrusion, which shall not exceed 0.25 mm per side.
- 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.
- Datums A & B to be determined at Datum H.

Microchip Technology Drawing No. C04-094C Sheet 2 of 2

# PIC16(L)F1507

20-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

Dimension Limits	Units	MILLIMETERS		
		MIN	NOM	MAX
Contact Pitch	E	1.27 BSC		
Contact Pad Spacing	C		9.40	
Contact Pad Width (X20)	X			0.60
Contact Pad Length (X20)	Y			1.95
Distance Between Pads	Gx	0.67		
Distance Between Pads	G	7.45		

Notes:

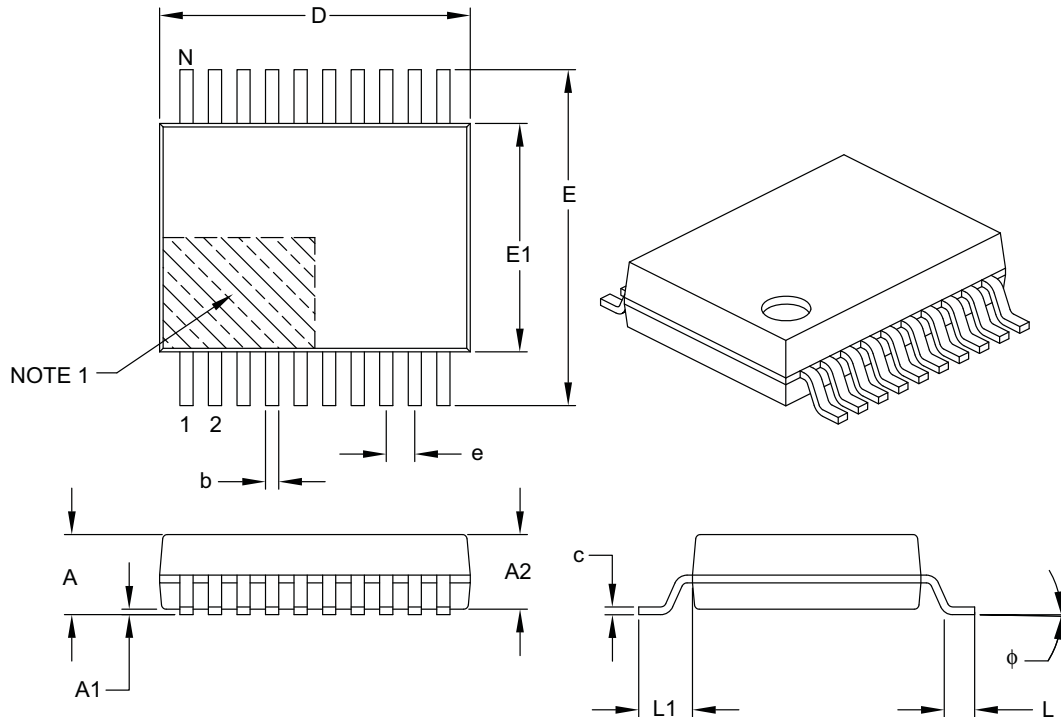
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2094A

## 20-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>



Dimension Limits	Units	MILLIMETERS		
		MIN	NOM	MAX
Number of Pins	N	20		
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	6.90	7.20	7.50
Foot Length	L	0.55	0.75	0.95
Footprint	L1	1.25 REF		
Lead Thickness	c	0.09	–	0.25
Foot Angle	$\phi$	0°	4°	8°
Lead Width	b	0.22	–	0.38

**Notes:**

- Pin 1 visual index feature may vary, but must be located within the hatched area.
- Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.20 mm per side.
- Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

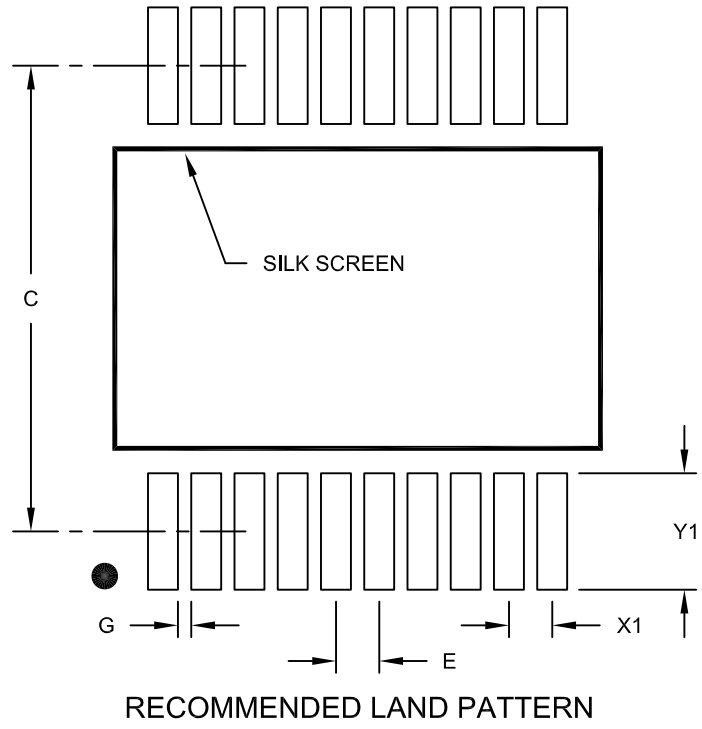
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-072B

# PIC16(L)F1507

20-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>



Dimension Limits	Units	MILLIMETERS		
		MIN	NOM	MAX
Contact Pitch	E	0.65 BSC		
Contact Pad Spacing	C		7.20	
Contact Pad Width (X20)	X1			0.45
Contact Pad Length (X20)	Y1			1.75
Distance Between Pads	G	0.20		

Notes:

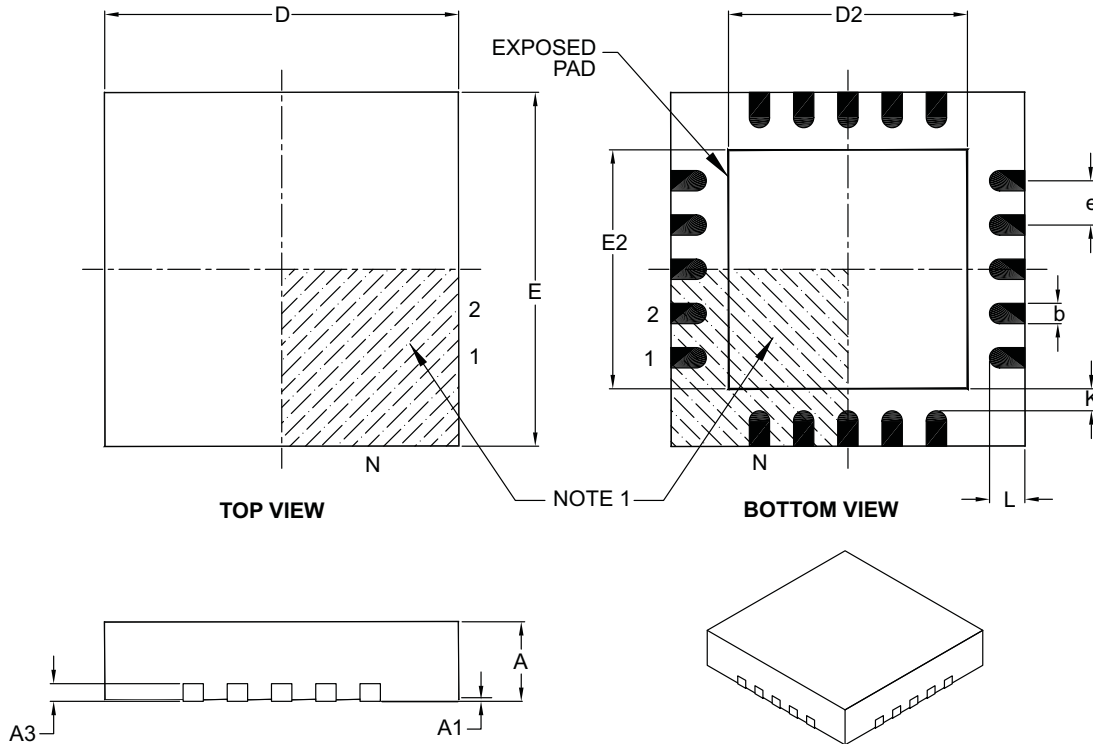
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2072A

## 20-Lead Plastic Quad Flat, No Lead Package (ML) – 4x4x0.9 mm Body [QFN]

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



Dimension Limits	Units	MILLIMETERS		
		MIN	NOM	MAX
Number of Pins	N	20		
Pitch	e	0.50 BSC		
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Contact Thickness	A3	0.20 REF		
Overall Width	E	4.00 BSC		
Exposed Pad Width	E2	2.60	2.70	2.80
Overall Length	D	4.00 BSC		
Exposed Pad Length	D2	2.60	2.70	2.80
Contact Width	b	0.18	0.25	0.30
Contact Length	L	0.30	0.40	0.50
Contact-to-Exposed Pad	K	0.20	–	–

**Notes:**

- Pin 1 visual index feature may vary, but must be located within the hatched area.
- Package is saw singulated.
- Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

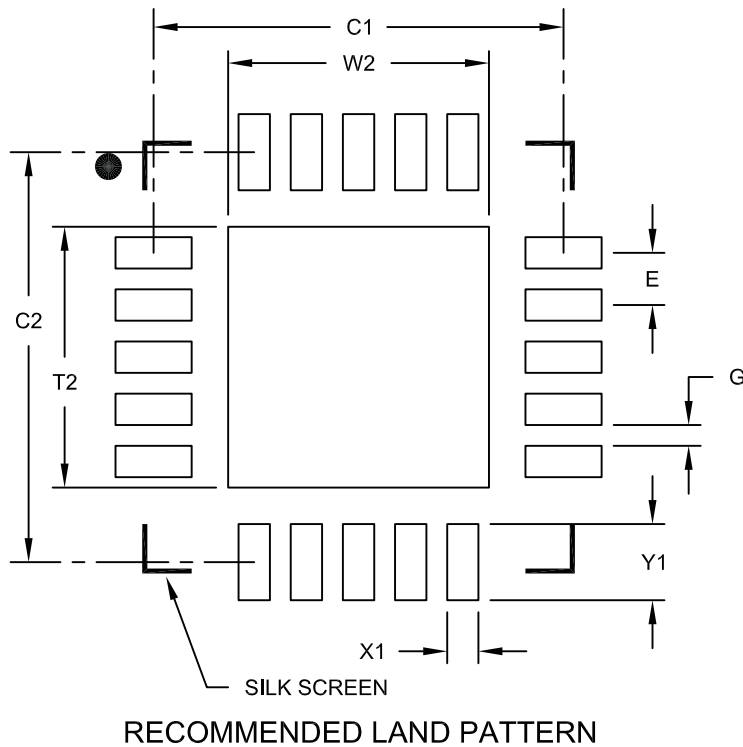
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-126B

# PIC16(L)F1507

20-Lead Plastic Quad Flat, No Lead Package (ML) - 4x4 mm Body [QFN]  
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>



Dimension Limits	Units	MILLIMETERS		
		MIN	NOM	MAX
Contact Pitch	E	0.50 BSC		
Optional Center Pad Width	W2			2.50
Optional Center Pad Length	T2			2.50
Contact Pad Spacing	C1		3.93	
Contact Pad Spacing	C2		3.93	
Contact Pad Width	X1			0.30
Contact Pad Length	Y1			0.73
Distance Between Pads	G	0.20		

**Notes:**

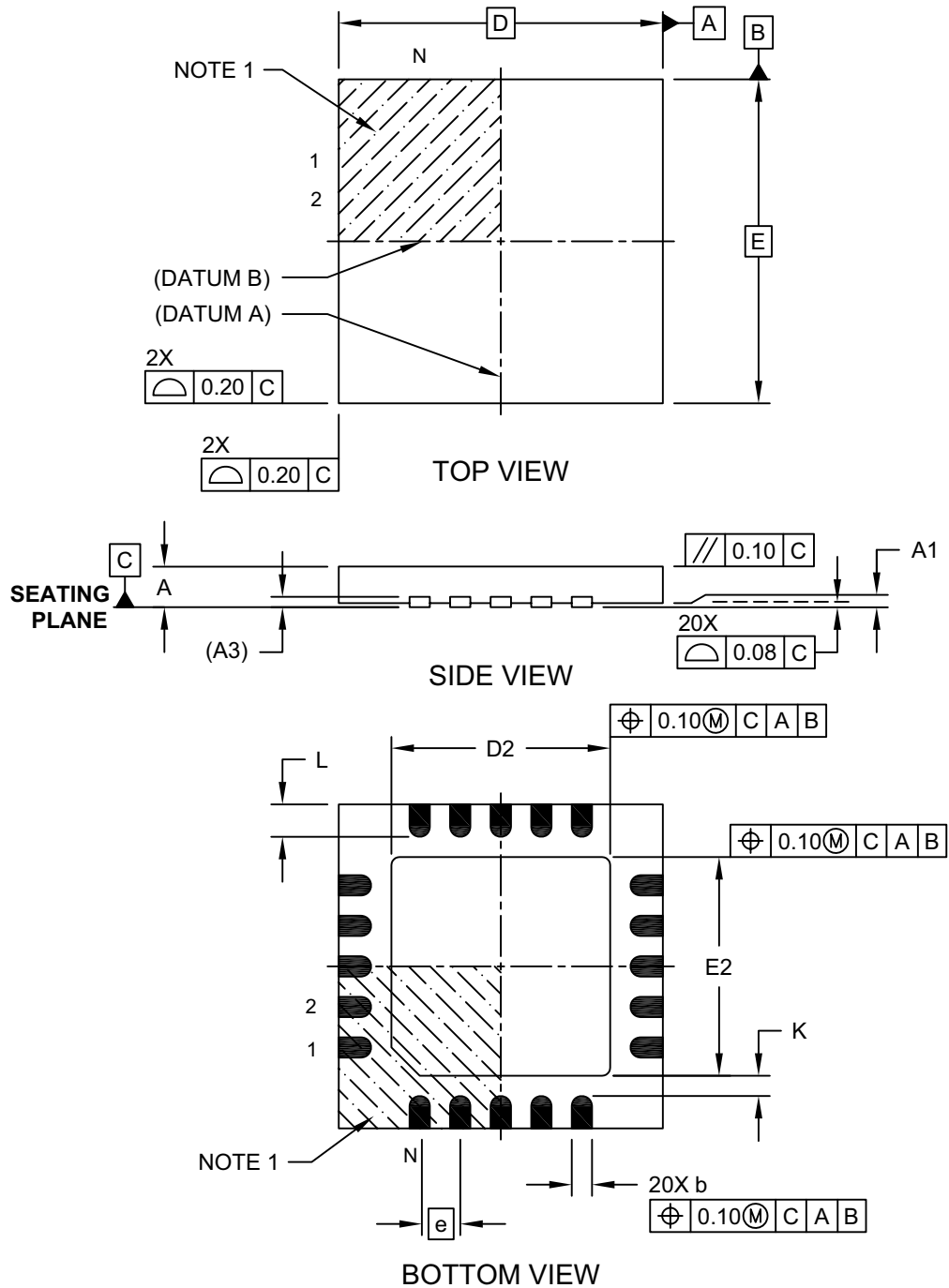
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2126A

## 20-Lead Ultra Thin Plastic Quad Flat, No Lead Package (GZ) - 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>

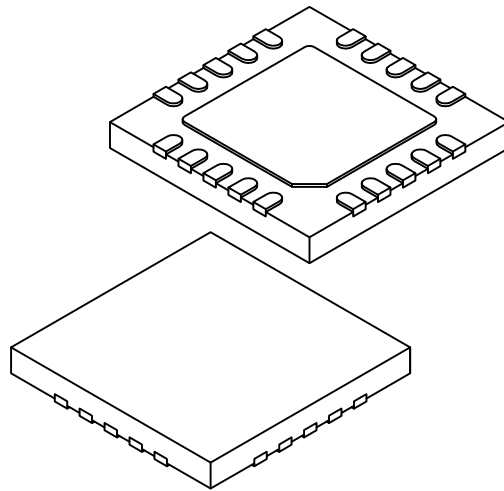


Microchip Technology Drawing C04-255A Sheet 1 of 2

# PIC16(L)F1507

## 20-Lead Ultra Thin Plastic Quad Flat, No Lead Package (GZ) - 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>



Dimension Limits	Units	MILLIMETERS		
		MIN	NOM	MAX
Number of Terminals	N	20		
Pitch	e	0.50 BSC		
Overall Height	A	0.45	0.50	0.55
Standoff	A1	0.00	0.02	0.05
Terminal Thickness	A3	0.127 REF		
Overall Width	E	4.00 BSC		
Exposed Pad Width	E2	2.60	2.70	2.80
Overall Length	D	4.00 BSC		
Exposed Pad Length	D2	2.60	2.70	2.80
Terminal Width	b	0.20	0.25	0.30
Terminal Length	L	0.30	0.40	0.50
Terminal-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.

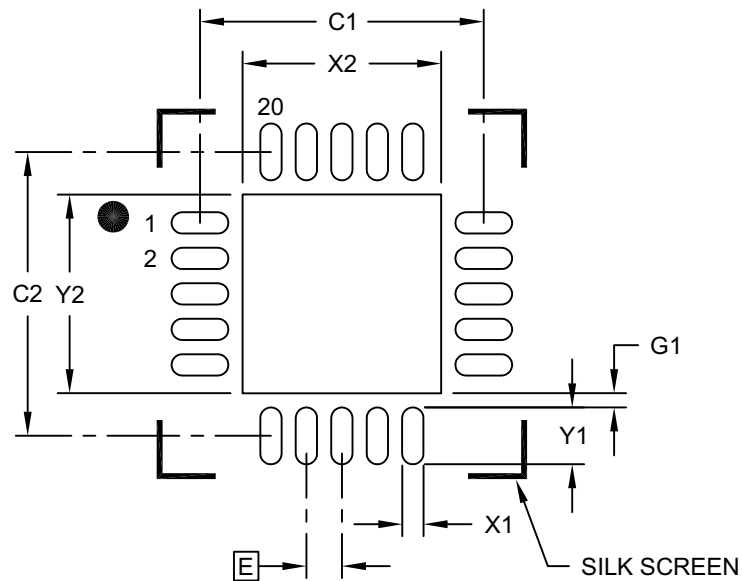
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-255A Sheet 2 of 2



## 20-Lead Ultra Thin Plastic Quad Flat, No Lead Package (GZ) - 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>



### RECOMMENDED LAND PATTERN

Dimension Limits	Units	MILLIMETERS		
		MIN	NOM	MAX
Contact Pitch	E	0.50 BSC		
Optional Center Pad Width	X2			2.80
Optional Center Pad Length	Y2			2.80
Contact Pad Spacing	C1		4.00	
Contact Pad Spacing	C2		4.00	
Contact Pad Width (X20)	X1			0.30
Contact Pad Length (X20)	Y1			0.80
Contact Pad to Center Pad (X20)	G1	0.20		

**Notes:**

1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-2255A

# PIC16(L)F1507

---

## APPENDIX A: DATA SHEET REVISION HISTORY

### Revision A (6/2011)

Original release.

### Revision B (04/2014)

Updated Electrical Specifications and added Characterization Data.

### Revision C (06/2014)

Updated with new 20-lead UQFN 4x4x0.5mm package.

Updated Product Identification System page and added new specifications for new packages.

Updated Table 25-6.

### Revision D (10/2015)

Added Section 3.2 High Endurance Flash. Updated Figure 22-1; Register 22-3; and Sections 20.1.5, 22.9.1.2, 22.11.1, and 25.1.

## THE MICROCHIP WEBSITE

Microchip provides online support via our website at [www.microchip.com](http://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](http://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>**

# PIC16(L)F1507

## PRODUCT IDENTIFICATION SYSTEM

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

<u>PART NO.</u>	<u>[X]<sup>(1)</sup></u>	-	<u>X</u>	<u>/XX</u>	<u>XXX</u>
Device	Tape and Reel Option		Temperature Range	Package	Pattern
<b>Device:</b>	PIC16LF1507, PIC16F1507				
<b>Tape and Reel Option:</b>	Blank = Standard packaging (tube or tray) T = Tape and Reel <sup>(1)</sup>				
<b>Temperature Range:</b>	I = -40°C to +85°C (Industrial) E = -40°C to +125°C (Extended)				
<b>Package:</b>	GZ = Micro Lead Frame (4x4x0.5mm) (UQFN) ML = Micro Lead Frame (4x4x0.9mm) (QFN) P = Plastic DIP SO = SOIC SS = SSOP				
<b>Pattern:</b>	QTP, SQTP, Code or Special Requirements (blank otherwise)				

**Examples:**

- a) PIC16LF1507T - I/SO  
Tape and Reel,  
Industrial temperature,  
SOIC package
- b) PIC16F1507 - I/P  
Industrial temperature  
PDIP package
- c) PIC16F1507 - E/ML 298  
Extended temperature,  
QFN package  
QTP pattern #298

**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.

**2:** For other small form-factor package availability and marking information, please visit [www.microchip.com/packaging](http://www.microchip.com/packaging) or contact your local sales office.

**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.

## Trademarks

The Microchip name and logo, the Microchip logo, dsPIC, FlashFlex, flexPWR, JukeBlox, KEELOQ, KEELOQ logo, Klear, LANCheck, MediaLB, MOST, MOST logo, MPLAB, OptoLyzer, PIC, PICSTART, PIC<sup>32</sup> 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.

The Embedded Control Solutions Company and mTouch are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Analog-for-the-Digital Age, BodyCom, chipKIT, chipKIT logo, CodeGuard, dsPICDEM, dsPICDEM.net, ECAN, In-Circuit Serial Programming, ICSP, Inter-Chip Connectivity, KlearNet, KlearNet logo, MiWi, motorBench, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, RightTouch logo, REAL ICE, SQI, Serial Quad I/O, 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 trademark 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.

© 2011-2015, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

ISBN: 978-1-63277-917-5

*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 =**



## 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](http://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 - Druenen**  
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