# PIC16(L)F1946/47

## 64-Pin Flash-Based, 8-Bit CMOS Microcontrollers with LCD Driver and XLP Technology

### High-Performance RISC CPU

- Only 49 Instructions to Learn:
  - All single-cycle instructions except branches
- Operating Speed:
  - DC – 32 MHz oscillator/clock input
  - DC – 125 ns instruction cycle
- Up to 16K x 14 Words of Flash Program Memory
- Up to 1024 Bytes of Data Memory (RAM)
- Interrupt Capability with Automatic Context Saving
- 16-Level Deep Hardware Stack
- Direct, Indirect and Relative Addressing modes
- Processor Read Access to Program Memory

### Special Microcontroller Features

- Precision Internal Oscillator:
  - Factory-calibrated to ±1%, typical
  - Software-selectable frequency range from 32 MHz to 31 kHz
- Power-Saving Sleep mode
- Power-on Reset (POR)
- Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Brown-out Reset (BOR):
  - Selectable between two trip points
  - Disable in Sleep option
- Multiplexed Master Clear with Pull-up/Input Pin
- Programmable Code Protection
- High Endurance Flash/EEPROM cell:
  - 100,000 write Flash endurance
  - 1,000,000 write EEPROM endurance
  - Flash/Data EEPROM retention: > 40 years
- Wide Operating Voltage Range:
  - 1.8V-5.5V (PIC16F1946/47)
  - 1.8V-3.6V (PIC16LF1946/47)

### PIC16LF1946/47 Low-Power Features

- Standby Current:
  - 60 nA @ 1.8V, typical
- Operating Current:
  - 7.0 μA @ 32 kHz, 1.8V, typical
  - 35 μA/MHz, 1.8V, typical
- Timer1 Oscillator Current:
  - 600 nA @ 32 kHz, 1.8V, typical
- Low-Power Watchdog Timer Current:
  - 500 nA @ 1.8V, typical

### Peripheral Features

- 54 I/O Pins (One Input-only pin):
  - High-current source/sink for direct LED drive
  - Individually programmable Interrupt-on-pin change pins
  - Individually programmable weak pull-ups
- Integrated LCD Controller:
  - Up to 184 segments
  - Variable clock input
  - Contrast control
  - Internal voltage reference selections
- Capacitive Sensing (CSM) Module (mTouch®):
  - 17 selectable channels
- A/D Converter:
  - 10-bit resolution and 17 channels
  - Selectable 1.024/2.048/4.096V voltage reference
- Timer0: 8-Bit Timer/Counter with 8-Bit Programmable Prescaler
- Enhanced Timer1:
  - Dedicated low-power 32 kHz oscillator driver
  - 16-bit timer/counter with prescaler
  - External Gate Input mode with toggle and single shot modes
  - Interrupt-on-gate completion
- Timer2, 4, 6: 8-Bit Timer/Counter with 8-Bit Period Register, Prescaler and Postscaler
- Two Capture, Compare, PWM Modules (CCP):
  - 16-bit Capture, max. resolution 125 ns
  - 16-bit Compare, max. resolution 125 ns
  - 10-bit PWM, max. frequency 31.25 kHz
- Three Enhanced Capture, Compare, PWM Modules (ECCP):
  - Three PWM time-base options
  - Auto-shutdown and auto-restart
  - PWM steering
  - Programmable dead-band delay

© 2010-2016 Microchip Technology Inc.

## Peripheral Features (Continued)

- Two Master Synchronous Serial Ports (MSSPs) with SPI and $I^2C$ with:
  - 7-bit address masking
  - SMBus/PMBus™ compatibility
  - Auto-wake-up on start
- Two Enhanced Universal Synchronous: Asynchronous Receiver Transmitters (EUSARTs)
  - RS-232, RS-485 and LIN compatible
  - Auto-Baud Detect
- SR Latch (555 Timer):
  - Multiple Set/Reset input options
  - Emulates 555 Timer applications
- Three Comparators:
  - Rail-to-rail inputs/outputs
  - Power mode control
  - Software enable hysteresis
- Voltage Reference Module:
  - Fixed Voltage Reference (FVR) with 1.024V, 2.048V and 4.096V output levels
  - 5-bit rail-to-rail resistive DAC with positive and negative reference selection

## PIC16(L)F193X/194X Family Types

| Device | Data Sheet Index | Program Memory Flash (words) | Data EEPROM (bytes) | Data SRAM (bytes) | I/Os(2) | 10-bit ADC (ch) | CapSense (ch) | Comparators | Timers (8/16-bit) | EUSART | MSSP ($I^2C$/SPI) | ECCP | CCP | LCD (Com/Seg/Total) | Debug(1) | XLP |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PIC16(L)F1933 | (1) | 4096 | 256 | 256 | 25 | 11 | 8 | 2 | 4/1 | 1 | 1 | 3 | 2 | 4/16/60(3) | I/H | Y |
| PIC16(L)F1934 | (2) | 4096 | 256 | 256 | 36 | 14 | 16 | 2 | 4/1 | 1 | 1 | 3 | 2 | 4/24/96 | I/H | Y |
| PIC16(L)F1936 | (2) | 8192 | 256 | 512 | 25 | 11 | 8 | 2 | 4/1 | 1 | 1 | 3 | 2 | 4/16/60(3) | I/H | Y |
| PIC16(L)F1937 | (2) | 8192 | 256 | 512 | 36 | 14 | 16 | 2 | 4/1 | 1 | 1 | 3 | 2 | 4/24/96 | I/H | Y |
| PIC16(L)F1938 | (3) | 16384 | 256 | 1024 | 25 | 11 | 8 | 2 | 4/1 | 1 | 1 | 3 | 2 | 4/16/60(3) | I/H | Y |
| PIC16(L)F1939 | (3) | 16384 | 256 | 1024 | 36 | 14 | 16 | 2 | 4/1 | 1 | 1 | 3 | 2 | 4/24/96 | I/H | Y |
| PIC16(L)F1946 | (4) | 8192 | 256 | 512 | 54 | 17 | 17 | 3 | 4/1 | 2 | 2 | 3 | 2 | 4/46/184 | I | Y |
| PIC16(L)F1947 | (4) | 16384 | 256 | 1024 | 54 | 17 | 17 | 3 | 4/1 | 2 | 2 | 3 | 2 | 4/46/184 | I | Y |

**Note 1:** I – Debugging, Integrated on Chip; H – Debugging, Requires Debug Header.

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

**3:** COM3 and SEG15 share the same physical pin, therefore SEG15 is not available when using 1/4 multiplex displays.

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

**1:** DS41575 PIC16(L)F1933 Data Sheet, 28-Pin Flash, 8-bit Microcontrollers.

**2:** DS41364 PIC16(L)F1934/6/7 Data Sheet, 28/40/44-Pin Flash, 8-bit Microcontrollers.

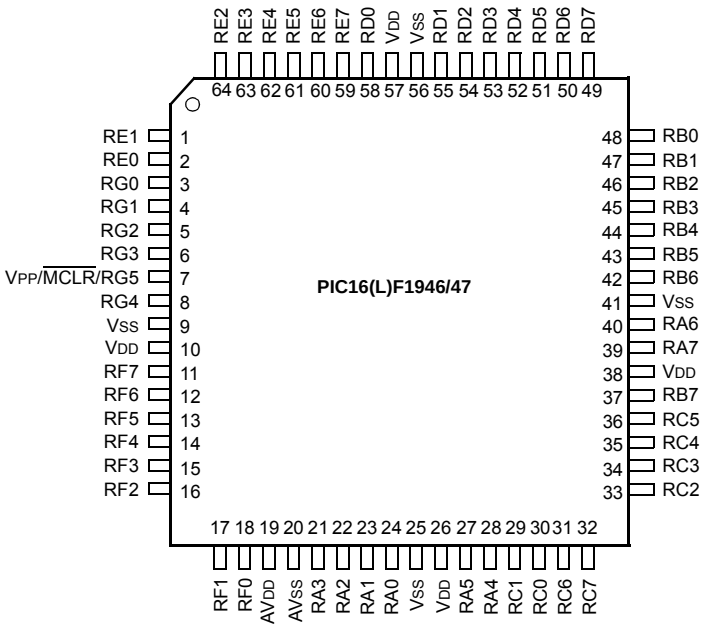
**3:** DS40001574 PIC16(L)F1938/9 Data Sheet, 28/40/44-Pin Flash, 8-bit Microcontrollers.

**4:** DS41414 PIC16(L)F1946/1947 Data Sheet, 64-Pin Flash, 8-bit Microcontrollers.


© 2010-2016 Microchip Technology Inc.

## Pin Diagram – 64-Pin TQFP/QFN (PIC16(L)F1946/47)

**64-pin TQFP, QFN**




**Note 1:** Pin location selected by APFCON register setting. Default location.

**2:** Pin function can be moved using the APFCON register. Alternate location.

**3:** QFN package orientation is the same. No leads are present on the QFN package.

**Note:** AVDD and AVSS are dedicated power connection pins for the on-board analog circuits of the PIC® microcontroller. The separate power pins help eliminate digital switching noise interference with the analog circuitry inside the device, especially on larger devices with more I/O pins and larger switching currents on the VDD/VSS pins. Customers typically connect these to the appropriate VDD or VSS connections on the PCB, unless there is a lot of noise on the external power rails. In those situations, they will add additional noise filtering components (like capacitors) on the AVDD/AVSS pins to help ensure good solid supply to the analog modules inside the device.

**TABLE 1: 64-PIN SUMMARY(PIC16(L)F1946/47)**

| I/O | 64-Pin TQFP, QFN | ANSEL | A/D | Reference | Cap Sense | Comparator | SR Latch | Timers | CCP | USART | MSSP | LCD | Interrupt | Pull-up | Basic |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RA0 | 24 | Y | AN0 | — | CPS0 | — | — | — | — | — | — | SEG33 | — | — | — |
| RA1 | 23 | Y | AN1 | — | CPS1 | — | — | — | — | — | — | SEG18 | — | — | — |
| RA2 | 22 | Y | AN2 | VREF- | CPS2 | — | — | — | — | — | — | SEG34 | — | — | — |
| RA3 | 21 | Y | AN3 | VREF+ | CPS3 | — | — | — | — | — | — | SEG35 | — | — | — |
| RA4 | 28 | — | — | — | — | — | — | T0CKI | — | — | — | SEG14 | — | — | — |
| RA5 | 27 | Y | AN4 | — | CPS4 | — | — | — | — | — | — | SEG15 | — | — | — |
| RA6 | 40 | — | — | — | — | — | — | — | — | — | — | SEG36 | — | — | OSC2/ CLKOUT |
| RA7 | 39 | — | — | — | — | — | — | — | — | — | — | SEG37 | — | — | OSC1/ CLKIN |
| RB0 | 48 | — | — | — | — | — | SRI | — | FLT0 | — | — | SEG30 | INT/ IOC | Y | — |
| RB1 | 47 | — | — | — | — | — | — | — | — | — | — | SEG8 | IOC | Y | — |
| RB2 | 46 | — | — | — | — | — | — | — | — | — | — | SEG9 | IOC | Y | — |
| RB3 | 45 | — | — | — | — | — | — | — | — | — | — | SEG10 | IOC | Y | — |
| RB4 | 44 | — | — | — | — | — | — | — | — | — | — | SEG11 | IOC | Y | — |
| RB5 | 43 | — | — | — | — | — | — | T1G | — | — | — | SEG29 | IOC | Y | — |
| RB6 | 42 | — | — | — | — | — | — | — | — | — | — | SEG38 | IOC | Y | ICSP-CLK/ ICDCLK |
| RB7 | 37 | — | — | — | — | — | — | — | — | — | — | SEG39 | IOC | Y | ICSP-DAT/ ICDDAT |
| RC0 | 30 | — | — | — | — | — | — | T1OSO/ T1CKI | — | — | — | SEG40 | — | — | — |
| RC1 | 29 | — | — | — | — | — | — | T1OSI | CCP2(1)/ P2A(1) | — | — | SEG32 | — | — | — |
| RC2 | 33 | — | — | — | — | — | — | — | CCP1/ P1A | — | — | SEG13 | — | — | — |
| RC3 | 34 | — | — | — | — | — | — | — | — | — | SCK1/ SCL1 | SEG17 | — | — | — |
| RC4 | 35 | — | — | — | — | — | — | — | — | — | SDI1/ SDA1 | SEG16 | — | — | — |
| RC5 | 36 | — | — | — | — | — | — | — | — | — | SDO1 | SEG12 | — | — | — |
| RC6 | 31 | — | — | — | — | — | — | — | — | TX1/ CK1 | — | SEG27 | — | — | — |
| RC7 | 32 | — | — | — | — | — | — | — | — | RX1/ DT1 | — | SEG28 | — | — | — |
| RD0 | 58 | — | — | — | — | — | — | — | P2D(2) | — | — | SEG0 | — | — | — |
| RD1 | 55 | — | — | — | — | — | — | — | P2C(2) | — | — | SEG1 | — | — | — |
| RD2 | 54 | — | — | — | — | — | — | — | P2B(2) | — | — | SEG2 | — | — | — |
| RD3 | 53 | — | — | — | — | — | — | — | P3C(2) | — | — | SEG3 | — | — | — |
| RD4 | 52 | — | — | — | — | — | — | — | P3B(2) | — | SDO2 | SEG4 | — | — | — |
| RD5 | 51 | — | — | — | — | — | — | — | P1C(2) | — | SDI2 SDA2 | SEG5 | — | — | — |
| RD6 | 50 | — | — | — | — | — | — | — | P1B(2) | — | SCK2/ SCL2 | SEG6 | — | — | — |
| RD7 | 49 | — | — | — | — | — | — | — | — | — | $\overline{SS2}$ | SEG7 | — | — | — |

**Note 1:** Pin functions can be moved using the APFCON register(s). Default location.

**2:** Pin function can be moved using the APFCON register. Alternate location.

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

**4:** See **Section 8.0 "Low Dropout (LDO) Voltage Regulator"**.


© 2010-2016 Microchip Technology Inc.

**TABLE 1: 64-PIN SUMMARY(PIC16(L)F1946/47) (Continued)**

| I/O | 64-Pin TQFP, QFN | ANSEL | A/D | Reference | Cap Sense | Comparator | SR Latch | Timers | CCP | USART | MSSP | LCD | Interrupt | Pull-up | Basic |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RE0 | 2 | Y | — | — | — | — | — | — | P2D(1) | — | | VLCD1 | — | — | — |
| RE1 | 1 | Y | — | — | — | — | — | — | P2C(1) | — | — | VLCD2 | — | — | — |
| RE2 | 64 | Y | — | — | — | — | — | — | P2B(1) | — | — | VLCD3 | — | — | — |
| RE3 | 63 | — | — | — | — | — | — | — | P3C(1) | — | — | COM0 | — | — | — |
| RE4 | 62 | — | — | — | — | — | — | — | P3B(1) | — | — | COM1 | — | — | — |
| RE5 | 61 | — | — | — | — | — | — | — | P1C(1) | — | — | COM2 | — | — | — |
| RE6 | 60 | — | — | — | — | — | — | — | P1B(1) | — | — | COM3 | — | — | — |
| RE7 | 59 | — | — | — | — | — | — | — | CCP2(2)/ P2A(2) | — | — | SEG31 | — | — | — |
| RF0 | 18 | Y | AN16 | — | CPS16 | C1IN0- C2IN0- | — | — | — | — | — | SEG41 | — | — | VCAP(4) |
| RF1 | 17 | Y | AN6 | — | CPS6 | C2OUT | SRNQ | — | — | — | — | SEG19 | — | — | — |
| RF2 | 16 | Y | AN7 | — | CPS7 | C1OUT | SRQ | — | — | — | — | SEG20 | — | — | — |
| RF3 | 15 | Y | AN8 | | CPS8 | C1IN2- C2IN2- C3IN2- | — | — | — | — | — | SEG21 | — | — | — |
| RF4 | 14 | Y | AN9 | — | CPS9 | C2IN+ | — | — | — | — | — | SEG22 | — | — | — |
| RF5 | 13 | Y | AN10 | DACOUT | CPS10 | C1IN1- C2IN1- | — | | — | — | — | SEG23 | — | — | — |
| RF6 | 12 | Y | AN11 | — | CPS11 | C1IN+ | — | — | — | — | — | SEG24 | — | — | — |
| RF7 | 11 | Y | AN5 | — | CPS5 | C1IN3- C2IN3- C3IN3- | — | — | — | — | $\overline{SS1}$ | SEG25 | — | — | — |
| RG0 | 3 | — | — | — | — | — | — | — | CCP3 P3A | — | — | SEG42 | — | — | — |
| RG1 | 4 | Y | AN15 | — | CPS15 | C3OUT | — | — | — | TX2/ CK2 | — | SEG43 | — | — | — |
| RG2 | 5 | Y | AN14 | — | CPS14 | C3IN+ | — | — | — | RX2/ DT2 | — | SEG44 | — | — | — |
| RG3 | 6 | Y | AN13 | — | CPS13 | C3IN0- | — | — | CCP4 P3D | — | — | SEG45 | — | — | — |
| RG4 | 8 | Y | AN12 | — | CPS12 | C3IN1- | — | — | CCP5 P1D | — | — | SEG26 | — | — | — |
| RG5 | 7 | — | — | — | — | — | — | — | — | — | — | — | — | Y(3) | $\overline{MCLR}$/VPP |
| VDD | 10 26 38 57 | — | — | — | — | — | — | — | — | — | — | — | — | — | VDD |
| VSS | 9 25 41 56 | — | — | — | — | — | — | — | — | — | — | — | — | — | VSS |
| AVDD | 19 | — | — | — | — | — | — | — | — | — | — | — | — | — | AVDD |
| AVSS | 20 | — | — | — | — | — | — | — | — | — | — | — | — | — | AVSS |

**Note 1:** Pin functions can be moved using the APFCON register(s). Default location.

**2:** Pin function can be moved using the APFCON register. Alternate location.

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

**4:** See **Section 8.0 "Low Dropout (LDO) Voltage Regulator"**.

## Table of Contents


1.0 Device Overview ..... 08
2.0 Enhanced Mid-Range CPU ..... 16
3.0 Memory Organization ..... 18
4.0 Device Configuration ..... 53
5.0 Oscillator Module (With Fail-Safe Clock Monitor) ..... 60
6.0 Resets ..... 77
7.0 Interrupts ..... 85
8.0 Low Dropout (LDO) Voltage Regulator ..... 99
9.0 Power-Down Mode (Sleep) ..... 100
10.0 Watchdog Timer (WDT) ..... 102
11.0 Data EEPROM and Flash Program Memory Control ..... 106
12.0 I/O Ports ..... 121
13.0 Interrupt-On-Change ..... 146
14.0 Fixed Voltage Reference (FVR) ..... 150
15.0 Temperature Indicator Module ..... 152
16.0 Analog-to-Digital Converter (ADC) Module ..... 161
17.0 Digital-to-Analog Converter (DAC) Module ..... 168
18.0 Comparator Module ..... 172
19.0 SR Latch ..... 181
20.0 Timer0 Module ..... 186
21.0 Timer1 Module with Gate Control ..... 189
22.0 Timer2/4/6 Modules ..... 200
23.0 Capture/Compare/PWM Modules ..... 204
24.0 Master Synchronous Serial Port (MSSP1 and MSSP2) Module ..... 232
25.0 Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART) ..... 287
26.0 Capacitive Sensing (CPS) Module ..... 316
27.0 Liquid Crystal Display (LCD) Driver Module ..... 324
28.0 In-Circuit Serial Programming™ (ICSP™) ..... 359
29.0 Instruction Set Summary ..... 362
30.0 Electrical Specifications ..... 376
31.0 DC and AC Characteristics Graphs and Charts ..... 409
32.0 Development Support ..... 443
33.0 Packaging Information ..... 447
Appendix A: Data Sheet Revision History ..... 455
Appendix B: Migrating From Other PIC® Devices ..... 455
The Microchip Website ..... 456
Customer Change Notification Service ..... 456
Customer Support ..... 456
Product Identification System ..... 457



© 2010-2016 Microchip Technology Inc.

## TO OUR VALUED CUSTOMERS

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

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

### Most Current Data Sheet

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

**http://www.microchip.com**

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

### Errata

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

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

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

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

### Customer Notification System

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

## 1.0 DEVICE OVERVIEW

The PIC16(L)F1946/47 are described within this data sheet. They are available in 64-pin packages. Figure 1-1 shows a block diagram of the PIC16(L)F1946/47 devices. Table 1-2 shows the pinout descriptions.

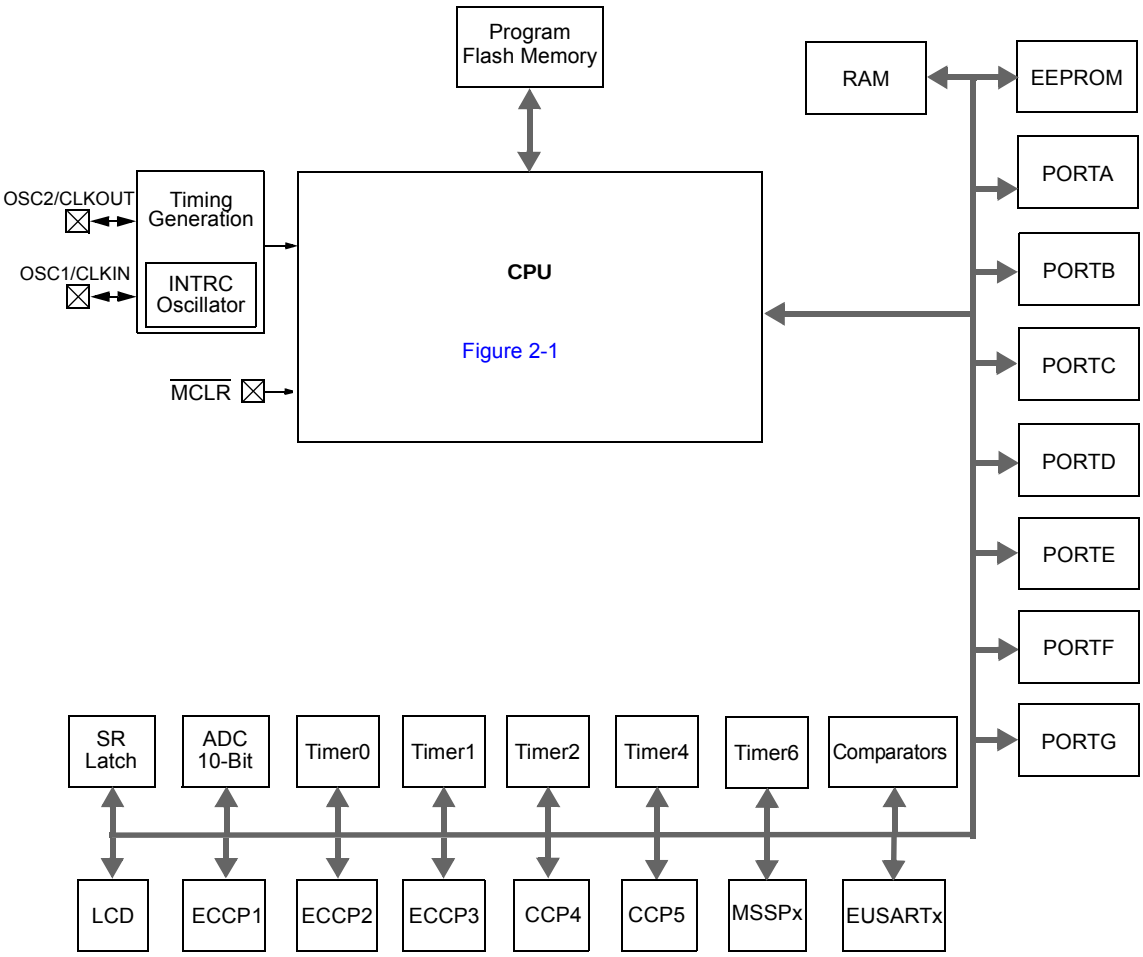
Reference Table 1-1 for peripherals available per device.

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

| Peripheral | | PIC16(L)F1946 | PIC16(L)F1947 |
|---|---|---|---|
| ADC | | ● | ● |
| Capacitive Sensing (CPS) Module | | ● | ● |
| Data EEPROM | | ● | ● |
| Digital-to-Analog Converter (DAC) | | ● | ● |
| Fixed Voltage Reference (FVR) | | ● | ● |
| LCD | | ● | ● |
| SR Latch | | ● | ● |
| Capture/Compare/PWM Modules | | | |
| | ECCP1 | ● | ● |
| | ECCP2 | ● | ● |
| | ECCP3 | ● | ● |
| | CCP4 | ● | ● |
| | CCP5 | ● | ● |
| Comparators | | | |
| | C1 | ● | ● |
| | C2 | ● | ● |
| | C3 | ● | ● |
| EUSARTS | | | |
| | EUSART1 | ● | ● |
| | EUSART2 | ● | ● |
| Master Synchronous Serial Ports | | | |
| | MSSP1 | ● | ● |
| | MSSP2 | ● | ● |
| Timers | | | |
| | Timer0 | ● | ● |
| | Timer1 | ● | ● |
| | Timer2 | ● | ● |
| | Timer4 | ● | ● |
| | Timer6 | ● | ● |


© 2010-2016 Microchip Technology Inc.

**FIGURE 1-1: PIC16(L)F1946/47 BLOCK DIAGRAM**




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

**TABLE 1-2: PIC16(L)F1946/47 PINOUT DESCRIPTION**

| Name | Function | Input Type | Output Type | Description |
|---|---|---|---|---|
| RA0/AN0/CPS0/SEG33 | RA0 | TTL | CMOS | General purpose I/O. |
| | AN0 | AN | — | A/D Channel input. |
| | CPS0 | AN | — | Capacitive sensing input 0. |
| | SEG33 | — | AN | LCD Analog output. |
| RA1/AN1/CPS1/SEG18 | RA1 | TTL | CMOS | General purpose I/O. |
| | AN1 | AN | — | A/D Channel input. |
| | CPS1 | AN | — | Capacitive sensing input. |
| | SEG18 | — | AN | LCD Analog output. |
| RA2/AN2/VREF-/CPS2/SEG34 | RA2 | TTL | CMOS | General purpose I/O. |
| | AN2 | AN | — | A/D Channel input. |
| | VREF- | AN | — | A/D Negative Voltage Reference input. |
| | CPS2 | AN | — | Capacitive sensing input. |
| | SEG34 | — | AN | LCD Analog output. |
| RA3/AN3/VREF+/CPS3/SEG35 | RA3 | TTL | CMOS | General purpose I/O. |
| | AN3 | AN | — | A/D Channel input. |
| | VREF+ | AN | — | A/D Voltage Reference input. |
| | CPS3 | AN | — | Capacitive sensing input. |
| | SEG35 | — | AN | LCD Analog output. |
| RA4/T0CKI/SEG14 | RA4 | TTL | CMOS | General purpose I/O. |
| | T0CKI | ST | — | Timer0 clock input. |
| | SEG14 | — | AN | LCD Analog output. |
| RA5/AN4/CPS4/SEG15 | RA5 | TTL | CMOS | General purpose I/O. |
| | AN4 | AN | — | A/D Channel input. |
| | CPS4 | AN | — | Capacitive sensing input. |
| | SEG15 | — | AN | LCD Analog output. |
| RA6/OSC2/CLKOUT/SEG36 | RA6 | TTL | CMOS | General purpose I/O. |
| | OSC2 | — | XTAL | Crystal/Resonator (LP, XT, HS modes). |
| | CLKOUT | — | CMOS | FOSC/4 output. |
| | SEG36 | — | AN | LCD Analog output. |
| RA7/OSC1/CLKIN/SEG37 | RA7 | TTL | CMOS | General purpose I/O. |
| | OSC1 | XTAL | — | Crystal/Resonator (LP, XT, HS modes). |
| | CLKIN | CMOS | — | External clock input (EC mode). |
| | SEG37 | — | AN | LCD Analog output. |
| RB0/INT/SRI/FLT0/SEG30 | RB0 | TTL | CMOS | General purpose I/O. Individually controlled interrupt-on-change. Individually enabled pull-up. |
| | INT | ST | — | External interrupt. |
| | SRI | — | ST | SR Latch input. |
| | FLT0 | ST | — | ECCP Auto-shutdown Fault input. |
| | SEG30 | — | AN | LCD analog output. |
| RB1/SEG8 | RB1 | TTL | CMOS | General purpose I/O. Individually controlled interrupt-on-change. Individually enabled pull-up. |
| | SEG8 | — | AN | LCD Analog output. |

**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; $I^2C$ = Schmitt Trigger input with $I^2C$ levels
HV = High Voltage; XTAL = Crystal

**Note 1:** Pin function is selectable via the APFCON register.


© 2010-2016 Microchip Technology Inc.

**TABLE 1-2: PIC16(L)F1946/47 PINOUT DESCRIPTION (CONTINUED)**

| Name | Function | Input Type | Output Type | Description |
|---|---|---|---|---|
| RB2/SEG9 | RB2 | TTL | CMOS | General purpose I/O. Individually controlled interrupt-on-change. Individually enabled pull-up. |
| | SEG9 | — | AN | LCD Analog output. |
| RB3/SEG10 | RB3 | TTL | CMOS | General purpose I/O. Individually controlled interrupt-on-change. Individually enabled pull-up. |
| | SEG10 | — | AN | LCD Analog output. |
| RB4/SEG11 | RB4 | TTL | CMOS | General purpose I/O. Individually controlled interrupt-on-change. Individually enabled pull-up. |
| | SEG11 | — | AN | LCD Analog output. |
| RB5/T1G/SEG29 | RB5 | TTL | CMOS | General purpose I/O. Individually controlled interrupt-on-change. Individually enabled pull-up. |
| | T1G | ST | — | Timer1 Gate input. |
| | SEG29 | — | AN | LCD Analog output. |
| RB6/ICSPCLK/ICDCLK/SEG38 | RB6 | TTL | CMOS | General purpose I/O. Individually controlled interrupt-on-change. Individually enabled pull-up. |
| | ICSPCLK | ST | — | Serial Programming Clock. |
| | ICDCLK | ST | — | In-Circuit Debug Clock. |
| | SEG38 | — | AN | LCD Analog output. |
| RB7/ICSPDAT/ICDDAT/SEG39 | RB7 | TTL | CMOS | General purpose I/O. Individually controlled interrupt-on-change. Individually enabled pull-up. |
| | ICSPDAT | ST | CMOS | ICSP™ Data I/O. |
| | ICDDAT | ST | CMOS | In-Circuit Data I/O. |
| | SEG39 | — | AN | LCD Analog output. |
| RC0/T1OSO/T1CKI/SEG40 | RC0 | ST | CMOS | General purpose I/O. |
| | T1OSO | XTAL | XTAL | Timer1 oscillator connection. |
| | T1CKI | ST | — | Timer1 clock input. |
| | SEG40 | — | AN | LCD Analog output. |
| RC1/T1OSI/P2A(1)/CCP2(1)/ SEG32 | RC1 | ST | CMOS | General purpose I/O. |
| | T1OSI | XTAL | XTAL | Timer1 oscillator connection. |
| | P2A | — | CMOS | PWM output. |
| | CCP2 | ST | CMOS | Capture/Compare/PWM. |
| | SEG32 | — | AN | LCD Analog output. |
| RC2/CCP1/P1A/SEG13 | RC2 | ST | CMOS | General purpose I/O. |
| | CCP1 | ST | CMOS | Capture/Compare/PWM. |
| | P1A | — | CMOS | PWM output. |
| | SEG13 | — | AN | LCD Analog output. |
| RC3/SCK1/SCL1/SEG17 | RC3 | ST | CMOS | General purpose I/O. |
| | SCK1 | ST | CMOS | SPI clock. |
| | SCL1 | $I^2C$ | OD | $I^2C$ clock. |
| | SEG17 | — | AN | LCD Analog output. |
| RC4/SDI1/SDA1/SEG16 | RC4 | ST | CMOS | General purpose I/O. |
| | SDI1 | ST | — | SPI data input. |
| | SDA1 | $I^2C$ | OD | $I^2C$ data input/output. |
| | SEG16 | — | AN | LCD Analog output. |

**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; $I^2C$ = Schmitt Trigger input with $I^2C$ levels
HV = High Voltage; XTAL = Crystal

**Note 1:** Pin function is selectable via the APFCON register.

**TABLE 1-2: PIC16(L)F1946/47 PINOUT DESCRIPTION (CONTINUED)**

| Name | Function | Input Type | Output Type | Description |
|---|---|---|---|---|
| RC5/SDO1/SEG12 | RC5 | ST | CMOS | General purpose I/O. |
| | SDO1 | — | CMOS | SPI data output. |
| | SEG12 | — | AN | LCD Analog output. |
| RC6/TX1/CK1/SEG27 | RC6 | ST | CMOS | General purpose I/O. |
| | TX1 | — | CMOS | USART1 asynchronous transmit. |
| | CK1 | ST | CMOS | USART1 synchronous clock. |
| | SEG27 | — | AN | LCD Analog output. |
| RC7/RX1/DT1/SEG28 | RC7 | ST | CMOS | General purpose I/O. |
| | RX1 | ST | — | USART1 asynchronous input. |
| | DT1 | ST | CMOS | USART1 synchronous data. |
| | SEG28 | — | AN | LCD Analog output. |
| RD0/P2D(1)/SEG0 | RD0 | ST | CMOS | General purpose I/O. |
| | P2D | — | CMOS | PWM output. |
| | SEG0 | — | AN | LCD Analog output. |
| RD1/P2C(1)/SEG1 | RD1 | ST | CMOS | General purpose I/O. |
| | P2C | — | CMOS | PWM output. |
| | SEG1 | — | AN | LCD Analog output. |
| RD2/P2B(1)/SEG2 | RD2 | ST | CMOS | General purpose I/O. |
| | P2B | — | CMOS | PWM output. |
| | SEG2 | — | AN | LCD Analog output. |
| RD3/P3C(1)/SEG3 | RD3 | ST | CMOS | General purpose I/O. |
| | P3C | — | CMOS | PWM output. |
| | SEG3 | — | AN | LCD analog output. |
| RD4/SDO2/P3B(1)/SEG4 | RD4 | ST | CMOS | General purpose I/O. |
| | SDO2 | — | CMOS | SPI data output. |
| | P3B | — | CMOS | PWM output. |
| | SEG4 | — | AN | LCD analog output. |
| RD5/SDI2/SDA2/P1C(1)/SEG5 | RD5 | ST | CMOS | General purpose I/O. |
| | SDI2 | ST | — | SPI data input. |
| | SDA2 | $I^2C$ | OD | $I^2C$ data input/output. |
| | P1C | — | CMOS | PWM output. |
| | SEG5 | — | AN | LCD analog output. |
| RD6/SCK2/SCL2/P1B(1)/SEG6 | RD6 | ST | CMOS | General purpose I/O. |
| | SCK2 | ST | CMOS | SPI clock. |
| | SCL2 | $I^2C$ | OD | $I^2C$ clock. |
| | P1B | — | CMOS | PWM output. |
| | SEG6 | — | AN | LCD analog output. |
| RD7/$\overline{SS2}$/SEG7 | RD7 | ST | CMOS | General purpose I/O. |
| | $\overline{SS2}$ | ST | — | Slave Select input. |
| | SEG7 | — | AN | LCD analog output. |
| RE0/P2D(1)/VLCD1 | RE0 | ST | CMOS | General purpose I/O. |
| | P2D | — | CMOS | PWM output. |
| | VLCD1 | AN | — | LCD analog 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; $I^2C$ = Schmitt Trigger input with $I^2C$ levels
HV = High Voltage; XTAL = Crystal

**Note 1:** Pin function is selectable via the APFCON register.


© 2010-2016 Microchip Technology Inc.

**TABLE 1-2: PIC16(L)F1946/47 PINOUT DESCRIPTION (CONTINUED)**

| Name | Function | Input Type | Output Type | Description |
|---|---|---|---|---|
| RE1/P2C(1)/VLCD2 | RE1 | ST | CMOS | General purpose I/O. |
| | P2C | — | CMOS | PWM output. |
| | VLCD2 | AN | — | LCD analog input. |
| RE2/P2B(1)/VLCD3 | RE2 | ST | CMOS | General purpose I/O. |
| | P2B | — | CMOS | PWM output. |
| | VLCD3 | AN | — | LCD analog input. |
| RE3/P3C(1)/COM0 | RE3 | ST | CMOS | General purpose I/O. |
| | P3C | — | CMOS | PWM output. |
| | COM0 | — | AN | LCD Analog output. |
| RE4/P3B(1)/COM1 | RE4 | ST | CMOS | General purpose I/O. |
| | P3B | — | CMOS | PWM output. |
| | COM1 | — | AN | LCD Analog output. |
| RE5/P1C(1)/COM2 | RE5 | ST | CMOS | General purpose I/O. |
| | P1C | — | CMOS | PWM output. |
| | COM2 | — | AN | LCD Analog output. |
| RE6/P1B(1)/COM3 | RE6 | ST | — | General purpose I/O. |
| | P1B | — | CMOS | PWM output. |
| | COM3 | — | AN | LCD Analog output. |
| RE7/CCP2(1)/P2A(1)/SEG31 | RE7 | ST | CMOS | General purpose I/O. |
| | CCP2 | ST | CMOS | Capture/Compare/PWM. |
| | P2A | — | CMOS | PWM output. |
| | SEG31 | — | AN | LCD analog output. |
| RF0/AN16/CPS16/C1IN0-/C2IN0 /SEG41/VCAP | RF0 | ST | CMOS | General purpose I/O. |
| | AN16 | AN | — | A/D Channel input. |
| | CPS16 | AN | — | Capacitive sensing input. |
| | C1IN0- | AN | — | Comparator negative input. |
| | C2IN0- | AN | — | Comparator negative input. |
| | SEG41 | — | AN | LCD Analog output. |
| | VCAP | Power | Power | Filter capacitor for Voltage Regulator. |
| RF1/AN6/CPS6/C2OUT/SRNQ/ SEG19 | RF1 | ST | CMOS | General purpose I/O. |
| | AN6 | AN | — | A/D Channel input. |
| | CPS6 | AN | — | Capacitive sensing input. |
| | C2OUT | — | CMOS | Comparator output. |
| | SRNQ | — | CMOS | SR Latch inverting output. |
| | SEG19 | — | AN | LCD Analog output. |
| RF2/AN7/CPS7/C1OUT/SRQ/ SEG20 | RF2 | ST | CMOS | General purpose I/O. |
| | AN7 | AN | — | A/D Channel input. |
| | CPS7 | AN | — | Capacitive sensing input. |
| | C1OUT | — | CMOS | Comparator output. |
| | SRQ | — | CMOS | SR Latch non-inverting output. |
| | SEG20 | — | AN | LCD Analog output. |

**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 $I^2C$ = Schmitt Trigger input with $I^2C$ levels
HV = High Voltage XTAL = Crystal

**Note 1:** Pin function is selectable via the APFCON register.

**TABLE 1-2: PIC16(L)F1946/47 PINOUT DESCRIPTION (CONTINUED)**

| Name | Function | Input Type | Output Type | Description |
|---|---|---|---|---|
| RF3/AN8/CPS8/C123IN2-/ SEG21 | RF3 | ST | CMOS | General purpose I/O. |
| | AN8 | AN | — | A/D Channel input. |
| | CPS8 | AN | — | Capacitive sensing input. |
| | C1IN2- | AN | — | Comparator negative input. |
| | C2IN2- | AN | — | Comparator negative input. |
| | C3IN2- | AN | — | Comparator negative input. |
| | SEG21 | — | AN | LCD Analog output. |
| RF4/AN9/CPS9/C2IN+/SEG22 | RF4 | ST | CMOS | General purpose I/O. |
| | AN9 | AN | — | A/D Channel input. |
| | CPS9 | AN | — | Capacitive sensing input. |
| | C2IN+ | AN | — | Comparator positive input. |
| | SEG22 | — | AN | LCD Analog output. |
| RF5/AN10/CPS10/C12IN1-/ DACOUT/SEG23 | RF5 | ST | CMOS | General purpose I/O. |
| | AN10 | AN | — | A/D Channel input. |
| | CPS10 | AN | — | Capacitive sensing input. |
| | C1IN1- | AN | — | Comparator negative input. |
| | C2IN1- | AN | — | Comparator negative input. |
| | DACOUT | — | AN | Voltage Reference output. |
| | SEG23 | — | AN | LCD Analog output. |
| RF6/AN11/CPS11/C1IN+/SEG24 | RF6 | ST | CMOS | General purpose I/O. |
| | AN11 | AN | — | A/D Channel input. |
| | CPS11 | AN | — | Capacitive sensing input. |
| | C1IN+ | AN | — | Comparator positive input. |
| | SEG24 | — | AN | LCD Analog output. |
| RF7/AN5/CPS5/C123IN3-/$\overline{SS1}$/ SEG25 | RF7 | ST | CMOS | General purpose I/O. |
| | AN5 | AN | — | A/D Channel input. |
| | CPS5 | AN | — | Capacitive sensing input. |
| | C1IN3- | AN | — | Comparator negative input. |
| | C2IN3- | AN | — | Comparator negative input. |
| | C3IN3- | AN | — | Comparator negative input. |
| | $\overline{SS1}$ | ST | — | Slave Select input. |
| | SEG25 | — | AN | LCD Analog output. |
| RG0/CCP3/P3A/SEG42 | RG0 | ST | CMOS | General purpose I/O. |
| | CCP3 | ST | CMOS | Capture/Compare/PWM. |
| | P3A | — | CMOS | PWM output. |
| | SEG42 | — | AN | LCD Analog output. |
| RG1/AN15/CPS15/TX2/CK2/ C3OUT/SEG43 | RG1 | ST | CMOS | General purpose I/O. |
| | AN15 | AN | — | A/D Channel input. |
| | CPS15 | AN | — | Capacitive sensing input. |
| | TX2 | — | CMOS | USART2 asynchronous transmit. |
| | CK2 | ST | CMOS | USART2 synchronous clock. |
| | C3OUT | — | CMOS | Comparator output. |
| | SEG43 | — | AN | LCD Analog output. |

**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 $I^2C$ = Schmitt Trigger input with $I^2C$ levels
HV = High Voltage XTAL = Crystal

**Note 1:** Pin function is selectable via the APFCON register.


© 2010-2016 Microchip Technology Inc.

**TABLE 1-2: PIC16(L)F1946/47 PINOUT DESCRIPTION (CONTINUED)**

| Name | Function | Input Type | Output Type | Description |
|---|---|---|---|---|
| RG2/AN14/CPS14/RX2/DT2/ C3IN+/SEG44 | RG2 | ST | CMOS | General purpose I/O. |
| | AN14 | AN | — | A/D Channel input. |
| | CPS14 | AN | — | Capacitive sensing input. |
| | RX2 | ST | — | USART2 asynchronous input. |
| | DT2 | ST | CMOS | USART2 synchronous data. |
| | C3IN+ | AN | — | Comparator positive input. |
| | SEG44 | — | AN | LCD Analog output. |
| RG3/AN13/CPS13/C3IN0-/ CCP4/P3D/SEG45 | RG3 | ST | CMOS | General purpose I/O. |
| | AN13 | AN | — | A/D Channel input. |
| | CPS13 | AN | — | Capacitive sensing input. |
| | C3IN0- | AN | — | Comparator negative input. |
| | CCP4 | ST | CMOS | Capture/Compare/PWM. |
| | P3D | — | CMOS | PWM output. |
| | SEG45 | — | AN | LCD Analog output. |
| RG4/AN12/CPS12/C3IN1-/ CCP5/P1D/SEG26 | RG4 | ST | CMOS | General purpose I/O. |
| | AN12 | AN | — | A/D Channel input. |
| | CPS12 | AN | — | Capacitive sensing input. |
| | C3IN1- | AN | — | Comparator negative input. |
| | CCP5 | ST | CMOS | Capture/Compare/PWM. |
| | P1D | — | CMOS | PWM output. |
| | SEG26 | — | AN | LCD Analog output. |
| RG5/$\overline{\text{MCLR}}$/$V_{PP}$ | RG5 | ST | — | General purpose input. |
| | $\overline{\text{MCLR}}$ | ST | — | Master Clear with internal pull-up. |
| | $V_{PP}$ | HV | — | Programming voltage. |
| $V_{DD}$ | $V_{DD}$ | Power | — | Positive supply. |
| $V_{SS}$ | $V_{SS}$ | 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 $I^2C$ = Schmitt Trigger input with $I^2C$ levels
HV = High Voltage XTAL = Crystal

**Note 1:** Pin function is selectable via the APFCON 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

### 2.1 Automatic Interrupt Context Saving

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

### 2.2 16-level Stack with Overflow and Underflow

These devices have an external stack memory 15-bit wide and 16-word 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 **Section 3.5 "Stack"** for more details.

### 2.3 File Select Registers

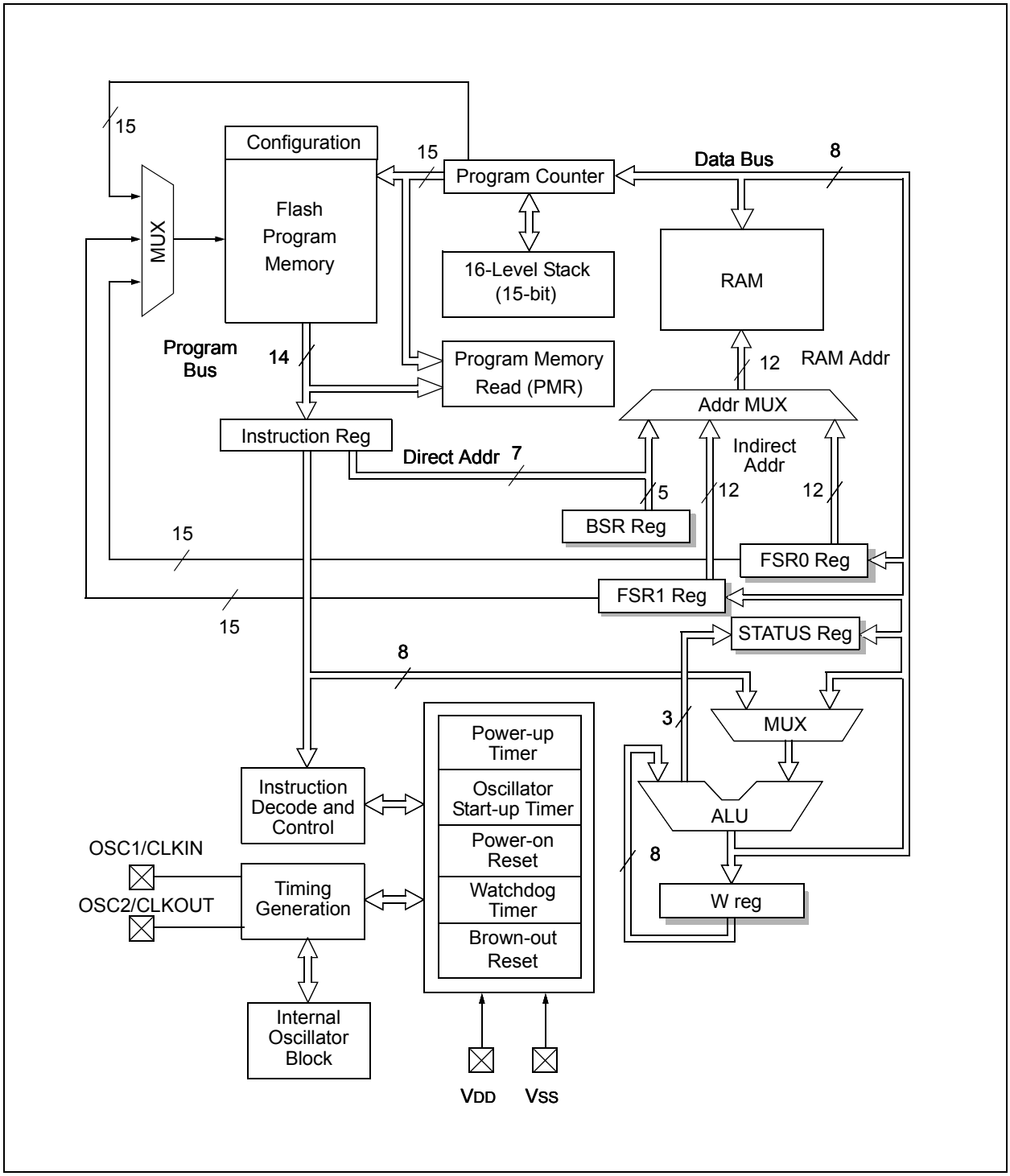
There are two 16-bit File Select Registers (FSR). FSRs can access all file registers and program memory, which allows one Data Pointer for all memory. When an FSR points to program memory, one additional instruction cycle is required to fetch the data at that address. General purpose memory can now also 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 29.0 "Instruction Set Summary"** for more details.


© 2010-2016 Microchip Technology Inc.

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



## 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
- Data EEPROM memory(1)

**Note 1:** The Data EEPROM Memory and the method to access Flash memory through the EECON registers is described in **Section 11.0 "Data EEPROM and Flash Program Memory Control"**.

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 for the PIC16(L)F1946/47 family. Accessing a location above these boundaries will cause a wrap-around within the implemented memory space. The Reset vector is at 0000h and the interrupt vector is at 0004h (see Figures 3-1 and 3-2).

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

| Device | Program Memory Space (Words) | Last Program Memory Address |
|---|---|---|
| PIC16(L)F1946 | 8,192 | 1FFFh |
| PIC16(L)F1947 | 16,384 | 3FFFh |


© 2010-2016 Microchip Technology Inc.

**FIGURE 3-1: PROGRAM MEMORY MAP AND STACK FOR PIC16(L)F1946**

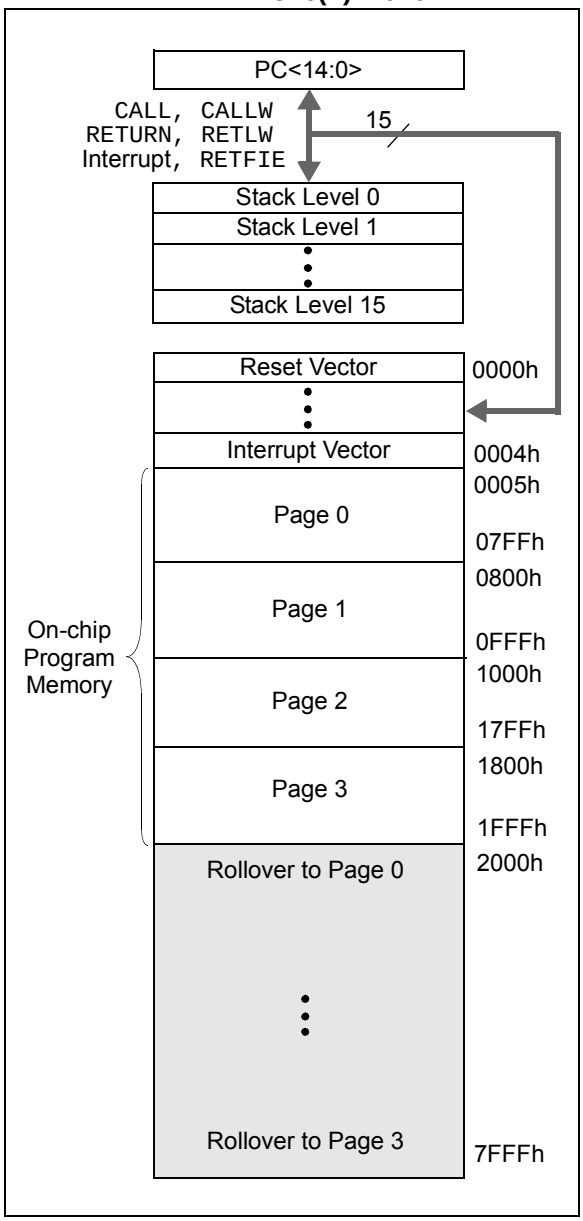
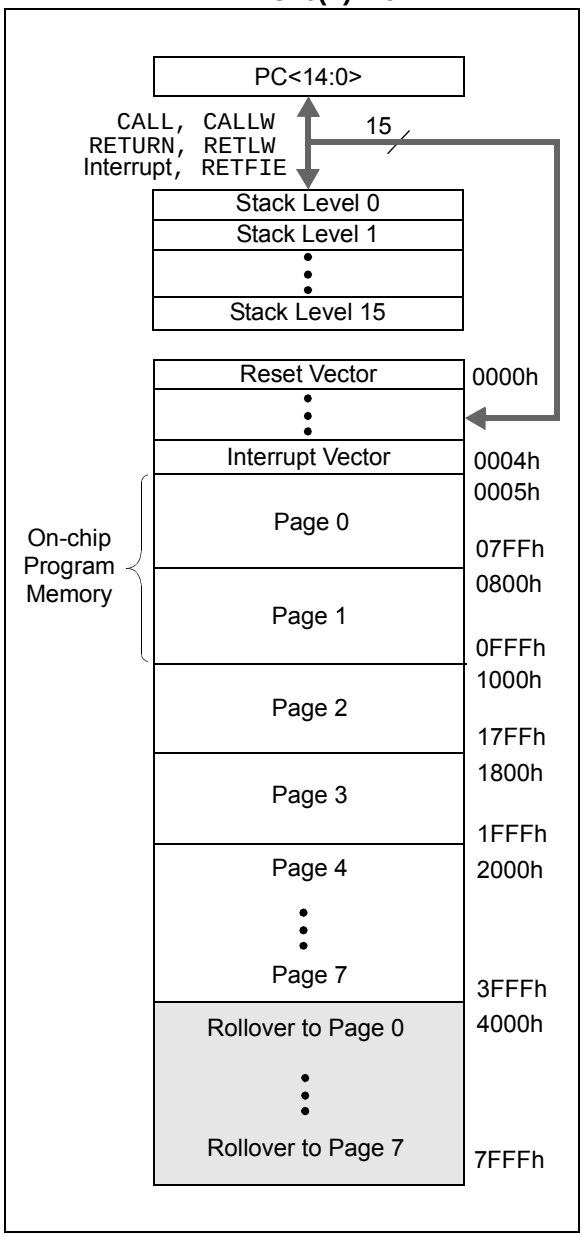



**FIGURE 3-2: PROGRAM MEMORY MAP AND STACK FOR PIC16(L)F1947**



### 3.1.1 READING PROGRAM MEMORY AS DATA

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

#### 3.1.1.1 RETLW Instruction

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

**EXAMPLE 3-1: RETLW INSTRUCTION**

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


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

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


© 2010-2016 Microchip Technology Inc.

#### 3.1.1.2 Indirect Read with FSR

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

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

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

```
constants
    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 constant ;MSb is set
automatically
    MOVWF   FSR1H
    BTFSC   STATUS,C    ;cary from ADDLW?
    INCF    FSR1H,f     ;yes
    MOVIW   0[FSR1]
;THE PROGRAM MEMORY IS IN W
```

## 3.2 Data Memory Organization

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

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

The active bank is selected by writing the bank number into the Bank Select Register (BSR). Unimplemented memory will read as '0'. All data memory can be accessed either directly (via instructions that use the file registers) or indirectly via the two File Select Registers (FSR). See **Section 3.6 "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.

### 3.2.1 CORE REGISTERS

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

**TABLE 3-2: CORE REGISTERS**

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

#### 3.2.1.1 STATUS Register

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

- the arithmetic status of the ALU
- the Reset status

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

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

It is recommended, therefore, that only `BCF`, `BSF`, `SWAPF` and `MOVWF` instructions are used to alter the STATUS register, because these instructions do not affect any Status bits. For other instructions not affecting any Status bits (Refer to **Section 29.0 "Instruction Set Summary"**).

**Note 1:** The C and DC bits operate as $\overline{Borrow}$ and $\overline{Digit\ Borrow}$ out bits, respectively, in subtraction.

## 3.3 Register Definitions: Status

**REGISTER 3-1: STATUS: STATUS REGISTER**

| U-0 | U-0 | U-0 | R-1/q | R-1/q | R/W-0/u | R/W-0/u | R/W-0/u |
|---|---|---|---|---|---|---|---|
| — | — | — | $\overline{TO}$ | $\overline{PD}$ | Z | DC(1) | C(1) |
| bit 7 | | | | | | | bit 0 |

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

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

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

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

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

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

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

**Note 1:** For $\overline{Borrow}$, 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.


© 2010-2016 Microchip Technology Inc.

### 3.3.1 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.

### 3.3.2 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).

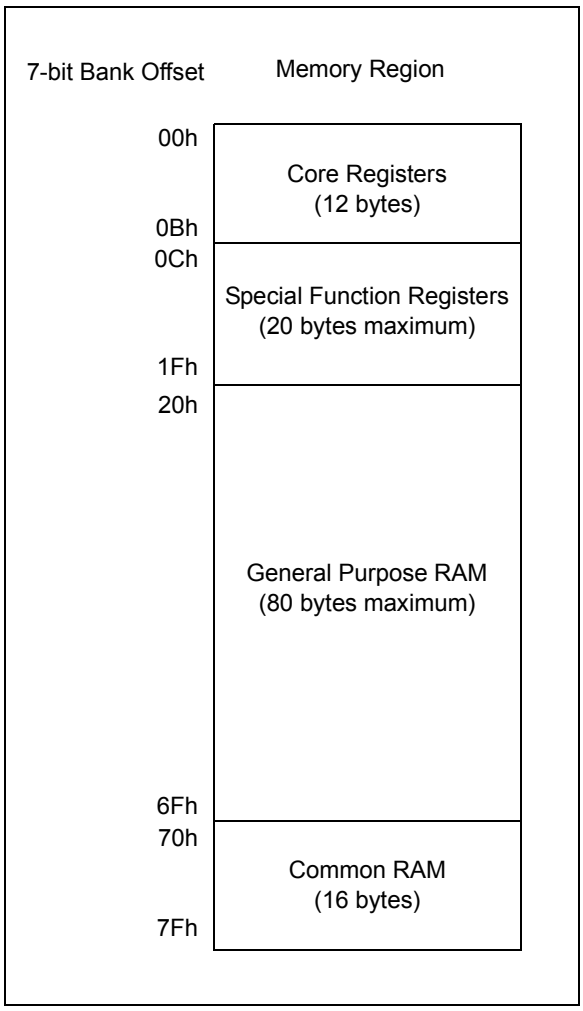
#### 3.3.2.1 Linear Access to GPR

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.

### 3.3.3 COMMON RAM

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

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




### 3.3.4 DEVICE MEMORY MAPS

The memory maps for the device family are as shown in Table 3-3.

**TABLE 3-3: MEMORY MAP TABLES**

| Device | Banks | Table No. |
|---|---|---|
| PIC16(L)F1946/47 | 0-7 | Table 3-4 |
| | 8-15 | Table 3-5, Table 3-8 |
| | 16-23 | Table 3-6 |
| | 23-31 | Table 3-7, Table 3-9 |

**TABLE 3-4: PIC16(L)F1946/47 MEMORY MAP, BANKS 0-7**

| | BANK 0 | | BANK 1 | | BANK 2 | | BANK 3 | | BANK 4 | | BANK 5 | | BANK 6 | | BANK 7 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 000h | INDF0 | 080h | INDF0 | 100h | INDF0 | 180h | INDF0 | 200h | INDF0 | 280h | INDF0 | 300h | INDF0 | 380h | INDF0 |
| 001h | INDF1 | 081h | INDF1 | 101h | INDF1 | 181h | INDF1 | 201h | INDF1 | 281h | INDF1 | 301h | INDF1 | 381h | INDF1 |
| 002h | PCL | 082h | PCL | 102h | PCL | 182h | PCL | 202h | PCL | 282h | PCL | 302h | PCL | 382h | PCL |
| 003h | STATUS | 083h | STATUS | 103h | STATUS | 183h | STATUS | 203h | STATUS | 283h | STATUS | 303h | STATUS | 383h | STATUS |
| 004h | FSR0L | 084h | FSR0L | 104h | FSR0L | 184h | FSR0L | 204h | FSR0L | 284h | FSR0L | 304h | FSR0L | 384h | FSR0L |
| 005h | FSR0H | 085h | FSR0H | 105h | FSR0H | 185h | FSR0H | 205h | FSR0H | 285h | FSR0H | 305h | FSR0H | 385h | FSR0H |
| 006h | FSR1L | 086h | FSR1L | 106h | FSR1L | 186h | FSR1L | 206h | FSR1L | 286h | FSR1L | 306h | FSR1L | 386h | FSR1L |
| 007h | FSR1H | 087h | FSR1H | 107h | FSR1H | 187h | FSR1H | 207h | FSR1H | 287h | FSR1H | 307h | FSR1H | 387h | FSR1H |
| 008h | BSR | 088h | BSR | 108h | BSR | 188h | BSR | 208h | BSR | 288h | BSR | 308h | BSR | 388h | BSR |
| 009h | WREG | 089h | WREG | 109h | WREG | 189h | WREG | 209h | WREG | 289h | WREG | 309h | WREG | 389h | WREG |
| 00Ah | PCLATH | 08Ah | PCLATH | 10Ah | PCLATH | 18Ah | PCLATH | 20Ah | PCLATH | 28Ah | PCLATH | 30Ah | PCLATH | 38Ah | PCLATH |
| 00Bh | INTCON | 08Bh | INTCON | 10Bh | INTCON | 18Bh | INTCON | 20Bh | INTCON | 28Bh | INTCON | 30Bh | INTCON | 38Bh | INTCON |
| 00Ch | PORTA | 08Ch | TRISA | 10Ch | LATA | 18Ch | ANSELA | 20Ch | — | 28Ch | PORTF | 30Ch | TRISF | 38Ch | LATF |
| 00Dh | PORTB | 08Dh | TRISB | 10Dh | LATB | 18Dh | — | 20Dh | WPUB | 28Dh | PORTG | 30Dh | TRISG | 38Dh | LATG |
| 00Eh | PORTC | 08Eh | TRISC | 10Eh | LATC | 18Eh | — | 20Eh | — | 28Eh | — | 30Eh | — | 38Eh | — |
| 00Fh | PORTD | 08Fh | TRISD | 10Fh | LATD | 18Fh | — | 20Fh | — | 28Fh | — | 30Fh | — | 38Fh | — |
| 010h | PORTE | 090h | TRISE | 110h | LATE | 190h | ANSELE | 210h | — | 290h | — | 310h | — | 390h | — |
| 011h | PIR1 | 091h | PIE1 | 111h | CM1CON0 | 191h | EEADRL | 211h | SSP1BUF | 291h | CCPR1L | 311h | CCPR3L | 391h | — |
| 012h | PIR2 | 092h | PIE2 | 112h | CM1CON1 | 192h | EEADRH | 212h | SSP1ADD | 292h | CCPR1H | 312h | CCPR3H | 392h | — |
| 013h | PIR3 | 093h | PIE3 | 113h | CM2CON0 | 193h | EEDATL | 213h | SSP1MSK | 293h | CCP1CON | 313h | CCP3CON | 393h | — |
| 014h | PIR4 | 094h | PIE4 | 114h | CM2CON1 | 194h | EEDATH | 214h | SSP1STAT | 294h | PWM1CON | 314h | PWM3CON | 394h | IOCBP |
| 015h | TMR0 | 095h | OPTION_REG | 115h | CMOUT | 195h | EECON1 | 215h | SSP1CON1 | 295h | CCP1AS | 315h | CCP3AS | 395h | IOCBN |
| 016h | TMR1L | 096h | PCON | 116h | BORCON | 196h | EECON2 | 216h | SSP1CON2 | 296h | PSTR1CON | 316h | PSTR3CON | 396h | IOCBF |
| 017h | TMR1H | 097h | WDTCON | 117h | FVRCON | 197h | — | 217h | SSP1CON3 | 297h | — | 317h | — | 397h | — |
| 018h | T1CON | 098h | OSCTUNE | 118h | DACCON0 | 198h | — | 218h | — | 298h | CCPR2L | 318h | CCPR4L | 398h | — |
| 019h | T1GCON | 099h | OSCCON | 119h | DACCON1 | 199h | RC1REG | 219h | SSP2BUF | 299h | CCPR2H | 319h | CCPR4H | 399h | — |
| 01Ah | TMR2 | 09Ah | OSCSTAT | 11Ah | SRCON0 | 19Ah | TX1REG | 21Ah | SSP2ADD | 29Ah | CCP2CON | 31Ah | CCP4CON | 39Ah | — |
| 01Bh | PR2 | 09Bh | ADRESL | 11Bh | SRCON1 | 19Bh | SP1BRGL | 21Bh | SSP2MSK | 29Bh | PWM2CON | 31Bh | — | 39Bh | — |
| 01Ch | T2CON | 09Ch | ADRESH | 11Ch | — | 19Ch | SP1BRGH | 21Ch | SSP2STAT | 29Ch | CCP2AS | 31Ch | CCPR5L | 39Ch | — |
| 01Dh | — | 09Dh | ADCON0 | 11Dh | APFCON | 19Dh | RC1STA | 21Dh | SSP2CON1 | 29Dh | PSTR2CON | 31Dh | CCPR5H | 39Dh | — |
| 01Eh | CPSCON0 | 09Eh | ADCON1 | 11Eh | CM3CON0 | 19Eh | TX1STA | 21Eh | SSP2CON2 | 29Eh | CCPTMRS0 | 31Eh | CCP5CON | 39Eh | — |
| 01Fh | CPSCON1 | 09Fh | — | 11Fh | CM3CON1 | 19Fh | BAUD1CON | 21Fh | SSP2CON3 | 29Fh | CCPTMRS1 | 31Fh | — | 39Fh | — |
| 020h<br>06Fh | General Purpose Register 96 Bytes | 0A0h<br>0EFh | General Purpose Register 80 Bytes | 120h<br>16Fh | General Purpose Register 80 Bytes | 1A0h<br>1EFh | General Purpose Register 80 Bytes | 220h<br>26Fh | General Purpose Register 80 Bytes | 2A0h<br>2EFh | General Purpose Register 80 Bytes | 320h<br>32Fh | General Purpose Register 16 Bytes | 3A0h<br>3EFh | General Purpose Register 80 Bytes(1) |
| | | | | | | | | | | | | 330h<br>36Fh | General Purpose Register 64 Bytes(1) | | |
| 070h<br>07Fh | Accesses 70h – 7Fh | 0F0h<br>0FFh | Accesses 70h – 7Fh | 170h<br>17Fh | Accesses 70h – 7Fh | 1F0h<br>1FFh | Accesses 70h – 7Fh | 270h<br>27Fh | Accesses 70h – 7Fh | 2F0h<br>2FFh | Accesses 70h – 7Fh | 370h<br>37Fh | Accesses 70h – 7Fh | 3F0h<br>3FFh | Accesses 70h – 7Fh |

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

**Note 1:** Not available on PIC16F1946.



© 2010-2016 Microchip Technology Inc.

**TABLE 3-5: PIC16(L)F1946/47 MEMORY MAP, BANKS 8-15**

| BANK 8 | | BANK 9 | | BANK 10 | | BANK 11 | | BANK 12 | | BANK 13 | | BANK 14 | | BANK 15 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 400h | INDF0 | 480h | INDF0 | 500h | INDF0 | 580h | INDF0 | 600h | INDF0 | 680h | INDF0 | 700h | INDF0 | 780h | INDF0 |
| 401h | INDF1 | 481h | INDF1 | 501h | INDF1 | 581h | INDF1 | 601h | INDF1 | 681h | INDF1 | 701h | INDF1 | 781h | INDF1 |
| 402h | PCL | 482h | PCL | 502h | PCL | 582h | PCL | 602h | PCL | 682h | PCL | 702h | PCL | 782h | PCL |
| 403h | STATUS | 483h | STATUS | 503h | STATUS | 583h | STATUS | 603h | STATUS | 683h | STATUS | 703h | STATUS | 783h | STATUS |
| 404h | FSR0L | 484h | FSR0L | 504h | FSR0L | 584h | FSR0L | 604h | FSR0L | 684h | FSR0L | 704h | FSR0L | 784h | FSR0L |
| 405h | FSR0H | 485h | FSR0H | 505h | FSR0H | 585h | FSR0H | 605h | FSR0H | 685h | FSR0H | 705h | FSR0H | 785h | FSR0H |
| 406h | FSR1L | 486h | FSR1L | 506h | FSR1L | 586h | FSR1L | 606h | FSR1L | 686h | FSR1L | 706h | FSR1L | 786h | FSR1L |
| 407h | FSR1H | 487h | FSR1H | 507h | FSR1H | 587h | FSR1H | 607h | FSR1H | 687h | FSR1H | 707h | FSR1H | 787h | FSR1H |
| 408h | BSR | 488h | BSR | 508h | BSR | 588h | BSR | 608h | BSR | 688h | BSR | 708h | BSR | 788h | BSR |
| 409h | WREG | 489h | WREG | 509h | WREG | 589h | WREG | 609h | WREG | 689h | WREG | 709h | WREG | 789h | WREG |
| 40Ah | PCLATH | 48Ah | PCLATH | 50Ah | PCLATH | 58Ah | PCLATH | 60Ah | PCLATH | 68Ah | PCLATH | 70Ah | PCLATH | 78Ah | PCLATH |
| 40Bh | INTCON | 48Bh | INTCON | 50Bh | INTCON | 58Bh | INTCON | 60Bh | INTCON | 68Bh | INTCON | 70Bh | INTCON | 78Bh | INTCON |
| 40Ch | ANSELF | 48Ch | — | 50Ch | — | 58Ch | — | 60Ch | — | 68Ch | — | 70Ch | — | 78Ch | — |
| 40Dh | ANSELG | 48Dh | WPUG | 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 | RC2REG | 511h | — | 591h | — | 611h | — | 691h | — | 711h | — | 791h | |
| 412h | — | 492h | TX2REG | 512h | — | 592h | — | 612h | — | 692h | — | 712h | — | 792h | |
| 413h | — | 493h | SP2BRGL | 513h | — | 593h | — | 613h | — | 693h | — | 713h | — | 793h | |
| 414h | — | 494h | SP2BRGH | 514h | — | 594h | — | 614h | — | 694h | — | 714h | — | 794h | |
| 415h | TMR4 | 495h | RC2STA | 515h | — | 595h | — | 615h | — | 695h | — | 715h | — | 795h | |
| 416h | PR4 | 496h | TX2STA | 516h | — | 596h | — | 616h | — | 696h | — | 716h | — | 796h | |
| 417h | T4CON | 497h | BAUD2CON | 517h | — | 597h | — | 617h | — | 697h | — | 717h | — | 797h | |
| 418h | — | 498h | — | 518h | — | 598h | — | 618h | — | 698h | — | 718h | — | 798h | |
| 419h | — | 499h | — | 519h | — | 599h | — | 619h | — | 699h | — | 719h | — | 799h | |
| 41Ah | — | 49Ah | — | 51Ah | — | 59Ah | — | 61Ah | — | 69Ah | — | 71Ah | — | 79Ah | |
| 41Bh | — | 49Bh | — | 51Bh | — | 59Bh | — | 61Bh | — | 69Bh | — | 71Bh | — | 79Bh | See Table 3-8 |
| 41Ch | TMR6 | 49Ch | — | 51Ch | — | 59Ch | — | 61Ch | — | 69Ch | — | 71Ch | — | 79Ch | |
| 41Dh | PR6 | 49Dh | — | 51Dh | — | 59Dh | — | 61Dh | — | 69Dh | — | 71Dh | — | 79Dh | |
| 41Eh | T6CON | 49Eh | — | 51Eh | — | 59Eh | — | 61Eh | — | 69Eh | — | 71Eh | — | 79Eh | |
| 41Fh | — | 49Fh | — | 51Fh | — | 59Fh | — | 61Fh | — | 69Fh | — | 71Fh | — | 79Fh | |
| 420h | General Purpose Register 80 Bytes(1) | 4A0h | General Purpose Register 80 Bytes(1) | 520h | General Purpose Register 80 Bytes(1) | 5A0h | General Purpose Register 80 Bytes(1) | 620h | General Purpose Register 48 Bytes(1) | 6A0h | Unimplemented Read as '0' | 720h | Unimplemented Read as '0' | 7A0h | |
| 46Fh | | 4EFh | | 56Fh | | 5EFh | | 66Fh | Unimplemented Read as '0' | 6EFh | | 76Fh | | 7EFh | |
| 470h | Accesses 70h – 7Fh | 4F0h | Accesses 70h – 7Fh | 570h | Accesses 70h – 7Fh | 5F0h | Accesses 70h – 7Fh | 670h | Accesses 70h – 7Fh | 6F0h | Accesses 70h – 7Fh | 770h | Accesses 70h – 7Fh | 7F0h | Accesses 70h – 7Fh |
| 47Fh | | 4FFh | | 57Fh | | 5FFh | | 67Fh | | 6FFh | | 77Fh | | 7FFh | |

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

**Note 1:** Not available on PIC16F1946.



© 2010-2016 Microchip Technology Inc.

**TABLE 3-6: PIC16(L)F1946/47 MEMORY MAP, BANKS 16-23**

| BANK 16 | | BANK 17 | | BANK 18 | | BANK 19 | | BANK 20 | | BANK 21 | | BANK 22 | | BANK 23 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 800h | INDF0 | 880h | INDF0 | 900h | INDF0 | 980h | INDF0 | A00h | INDF0 | A80h | INDF0 | B00h | INDF0 | B80h | INDF0 |
| 801h | INDF1 | 881h | INDF1 | 901h | INDF1 | 981h | INDF1 | A01h | INDF1 | A81h | INDF1 | B01h | INDF1 | B81h | INDF1 |
| 802h | PCL | 882h | PCL | 902h | PCL | 982h | PCL | A02h | PCL | A82h | PCL | B02h | PCL | B82h | PCL |
| 803h | STATUS | 883h | STATUS | 903h | STATUS | 983h | STATUS | A03h | STATUS | A83h | STATUS | B03h | STATUS | B83h | STATUS |
| 804h | FSR0L | 884h | FSR0L | 904h | FSR0L | 984h | FSR0L | A04h | FSR0L | A84h | FSR0L | B04h | FSR0L | B84h | FSR0L |
| 805h | FSR0H | 885h | FSR0H | 905h | FSR0H | 985h | FSR0H | A05h | FSR0H | A85h | FSR0H | B05h | FSR0H | B85h | FSR0H |
| 806h | FSR1L | 886h | FSR1L | 906h | FSR1L | 986h | FSR1L | A06h | FSR1L | A86h | FSR1L | B06h | FSR1L | B86h | FSR1L |
| 807h | FSR1H | 887h | FSR1H | 907h | FSR1H | 987h | FSR1H | A07h | FSR1H | A87h | FSR1H | B07h | FSR1H | B87h | FSR1H |
| 808h | BSR | 888h | BSR | 908h | BSR | 988h | BSR | A08h | BSR | A88h | BSR | B08h | BSR | B88h | BSR |
| 809h | WREG | 889h | WREG | 909h | WREG | 989h | WREG | A09h | WREG | A89h | WREG | B09h | WREG | B89h | WREG |
| 80Ah | PCLATH | 88Ah | PCLATH | 90Ah | PCLATH | 98Ah | PCLATH | A0Ah | PCLATH | A8Ah | PCLATH | B0Ah | PCLATH | B8Ah | PCLATH |
| 80Bh | INTCON | 88Bh | INTCON | 90Bh | INTCON | 98Bh | INTCON | A0Bh | INTCON | A8Bh | INTCON | B0Bh | INTCON | B8Bh | INTCON |
| 80Ch | — | 88Ch | — | 90Ch | — | 98Ch | — | A0Ch | — | A8Ch | — | B0Ch | — | B8Ch | — |
| 80Dh | — | 88Dh | — | 90Dh | — | 98Dh | — | A0Dh | — | A8Dh | — | B0Dh | — | B8Dh | — |
| 80Eh | — | 88Eh | — | 90Eh | — | 98Eh | — | A0Eh | — | A8Eh | — | B0Eh | — | B8Eh | — |
| 80Fh | — | 88Fh | — | 90Fh | — | 98Fh | — | A0Fh | — | A8Fh | — | B0Fh | — | B8Fh | — |
| 810h | — | 890h | — | 910h | — | 990h | — | A10h | — | A90h | — | B10h | — | B90h | — |
| 811h | — | 891h | — | 911h | — | 991h | — | A11h | — | A91h | — | B11h | — | B91h | — |
| 812h | — | 892h | — | 912h | — | 992h | — | A12h | — | A92h | — | B12h | — | B92h | — |
| 813h | — | 893h | — | 913h | — | 993h | — | A13h | — | A93h | — | B13h | — | B93h | — |
| 814h | — | 894h | — | 914h | — | 994h | — | A14h | — | A94h | — | B14h | — | B94h | — |
| 815h | — | 895h | — | 915h | — | 995h | — | A15h | — | A95h | — | B15h | — | B95h | — |
| 816h | — | 896h | — | 916h | — | 996h | — | A16h | — | A96h | — | B16h | — | B96h | — |
| 817h | — | 897h | — | 917h | — | 997h | — | A17h | — | A97h | — | B17h | — | B97h | — |
| 818h | — | 898h | — | 918h | — | 998h | — | A18h | — | A98h | — | B18h | — | B98h | — |
| 819h | — | 899h | — | 919h | — | 999h | — | A19h | — | A99h | — | B19h | — | B99h | — |
| 81Ah | — | 89Ah | — | 91Ah | — | 99Ah | — | A1Ah | — | A9Ah | — | B1Ah | — | B9Ah | — |
| 81Bh | — | 89Bh | — | 91Bh | — | 99Bh | — | A1Bh | — | A9Bh | — | B1Bh | — | B9Bh | — |
| 81Ch | — | 89Ch | — | 91Ch | — | 99Ch | — | A1Ch | — | A9Ch | — | B1Ch | — | B9Ch | — |
| 81Dh | — | 89Dh | — | 91Dh | — | 99Dh | — | A1Dh | — | A9Dh | — | B1Dh | — | B9Dh | — |
| 81Eh | — | 89Eh | — | 91Eh | — | 99Eh | — | A1Eh | — | A9Eh | — | B1Eh | — | B9Eh | — |
| 81Fh | — | 89Fh | — | 91Fh | — | 99Fh | — | A1Fh | — | A9Fh | — | B1Fh | — | B9Fh | — |
| 820h<br>Unimplemented Read as '0'<br>86Fh | | 8A0h<br>Unimplemented Read as '0'<br>8EFh | | 920h<br>Unimplemented Read as '0'<br>96Fh | | 9A0h<br>Unimplemented Read as '0'<br>9EFh | | A20h<br>Unimplemented Read as '0'<br>A6Fh | | AA0h<br>Unimplemented Read as '0'<br>AEFh | | B20h<br>Unimplemented Read as '0'<br>B6Fh | | BA0h<br>Unimplemented Read as '0'<br>BEFh | |
| 870h<br>Accesses 70h – 7Fh<br>87Fh | | 8F0h<br>Accesses 70h – 7Fh<br>8FFh | | 970h<br>Accesses 70h – 7Fh<br>97Fh | | 9F0h<br>Accesses 70h – 7Fh<br>9FFh | | A70h<br>Accesses 70h – 7Fh<br>A7Fh | | AF0h<br>Accesses 70h – 7Fh<br>AFFh | | B70h<br>Accesses 70h – 7Fh<br>B7Fh | | BF0h<br>Accesses 70h – 7Fh<br>BFFh | |

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



© 2010-2016 Microchip Technology Inc.

**TABLE 3-7: PIC16(L)F1946/47 MEMORY MAP, BANKS 24-31**

| | BANK 24 | | BANK 25 | | BANK 26 | | BANK 27 | | BANK 28 | | BANK 29 | | BANK 30 | | BANK 31 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| C00h | INDF0 | C80h | INDF0 | D00h | INDF0 | D80h | INDF0 | E00h | INDF0 | E80h | INDF0 | F00h | INDF0 | F80h | INDF0 |
| C01h | INDF1 | C81h | INDF1 | D01h | INDF1 | D81h | INDF1 | E01h | INDF1 | E81h | INDF1 | F01h | INDF1 | F81h | INDF1 |
| C02h | PCL | C82h | PCL | D02h | PCL | D82h | PCL | E02h | PCL | E82h | PCL | F02h | PCL | F82h | PCL |
| C03h | STATUS | C83h | STATUS | D03h | STATUS | D83h | STATUS | E03h | STATUS | E83h | STATUS | F03h | STATUS | F83h | STATUS |
| C04h | FSR0L | C84h | FSR0L | D04h | FSR0L | D84h | FSR0L | E04h | FSR0L | E84h | FSR0L | F04h | FSR0L | F84h | FSR0L |
| C05h | FSR0H | C85h | FSR0H | D05h | FSR0H | D85h | FSR0H | E05h | FSR0H | E85h | FSR0H | F05h | FSR0H | F85h | FSR0H |
| C06h | FSR1L | C86h | FSR1L | D06h | FSR1L | D86h | FSR1L | E06h | FSR1L | E86h | FSR1L | F06h | FSR1L | F86h | FSR1L |
| C07h | FSR1H | C87h | FSR1H | D07h | FSR1H | D87h | FSR1H | E07h | FSR1H | E87h | FSR1H | F07h | FSR1H | F87h | FSR1H |
| C08h | BSR | C88h | BSR | D08h | BSR | D88h | BSR | E08h | BSR | E88h | BSR | F08h | BSR | F88h | BSR |
| C09h | WREG | C89h | WREG | D09h | WREG | D89h | WREG | E09h | WREG | E89h | WREG | F09h | WREG | F89h | WREG |
| C0Ah | PCLATH | C8Ah | PCLATH | D0Ah | PCLATH | D8Ah | PCLATH | E0Ah | PCLATH | E8Ah | PCLATH | F0Ah | PCLATH | F8Ah | PCLATH |
| C0Bh | INTCON | C8Bh | INTCON | D0Bh | INTCON | D8Bh | INTCON | E0Bh | INTCON | E8Bh | INTCON | F0Bh | INTCON | F8Bh | INTCON |
| C0Ch | — | C8Ch | — | D0Ch | — | D8Ch | — | E0Ch | — | E8Ch | — | F0Ch | — | F8Ch | See Table 3-9 |
| 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 – C6Fh | Unimplemented Read as '0' | CA0h – CEFh | Unimplemented Read as '0' | D20h – D6Fh | Unimplemented Read as '0' | DA0h – DEFh | Unimplemented Read as '0' | E20h – E6Fh | Unimplemented Read as '0' | EA0h – EEFh | Unimplemented Read as '0' | F20h – F6Fh | Unimplemented Read as '0' | FA0h – FEFh | |
| C70h – CFFh | Accesses 70h – 7Fh | CF0h – CFFh | Accesses 70h – 7Fh | D70h – D7Fh | Accesses 70h – 7Fh | DF0h – DFFh | Accesses 70h – 7Fh | E70h – E7Fh | Accesses 70h – 7Fh | EF0h – EFFh | Accesses 70h – 7Fh | F70h – F7Fh | Accesses 70h – 7Fh | FF0h – FFFh | Accesses 70h – 7Fh |

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



© 2010-2016 Microchip Technology Inc.

**TABLE 3-8: PIC16(L)F1946/47 MEMORY MAP, BANK 15**

| Bank 15 | |
|---|---|
| 791h | LCDCON |
| 792h | LCDPS |
| 793h | LCDREF |
| 794h | LCDCST |
| 795h | LCDRL |
| 796h | — |
| 797h | — |
| 798h | LCDSE0 |
| 799h | LCDSE1 |
| 79Ah | LCDSE2 |
| 79Bh | LCDSE3 |
| 79Ch | LCDSE4 |
| 79Dh | LCDSE5 |
| 79Eh | — |
| 79Fh | — |
| 7A0h | LCDDATA0 |
| 7A1h | LCDDATA1 |
| 7A2h | LCDDATA2 |
| 7A3h | LCDDATA3 |
| 7A4h | LCDDATA4 |
| 7A5h | LCDDATA5 |
| 7A6h | LCDDATA6 |
| 7A7h | LCDDATA7 |
| 7A8h | LCDDATA8 |
| 7A9h | LCDDATA9 |
| 7AAh | LCDDATA10 |
| 7ABh | LCDDATA11 |
| 7ACh | LCDDATA12 |
| 7ADh | LCDDATA13 |
| 7AEh | LCDDATA14 |
| 7AFh | LCDDATA15 |
| 7B0h | LCDDATA16 |
| 7B1h | LCDDATA17 |
| 7B2h | LCDDATA18 |
| 7B3h | LCDDATA19 |
| 7B4h | LCDDATA20 |
| 7B5h | LCDDATA21 |
| 7B6h | LCDDATA22 |
| 7B7h | LCDDATA23 |
| 7B8h – 7EFh | Unimplemented Read as ‘0’ |

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

**TABLE 3-9: PIC16(L)F1946/47 MEMORY MAP, BANK 31**

| Bank 31 | |
|---|---|
| F8Ch | — |
| F8Dh | — |
| F8Eh | — |
| F8Fh | — |
| F90h | — |
| F91h | — |
| F92h | — |
| F93h | — |
| F94h | — |
| F95h | — |
| F96h | — |
| F97h | — |
| F98h | — |
| F99h | — |
| F9Ah | — |
| F9Bh | — |
| F9Ch | — |
| F9Dh | — |
| F9Eh | — |
| F9Fh | — |
| FA0h | — |
| FA1h | — |
| FA2h | — |
| FA3h | — |
| FA4h | — |
| FA5h | — |
| FA6h | — |
| FA7h | — |
| FA8h | — |
| FA9h | — |
| FAAh | — |
| FABh | — |
| FDFh | — |
| FC0h | — |
| FDFh | — |
| FE0h | — |
| FE1h | — |
| FE2h | — |
| FE3h | — |
| FE4h | STATUS_SHAD |
| FE5h | WREG_SHAD |
| FE6h | BSR_SHAD |
| FE7h | PCLATH_SHAD |
| FE8h | FSR0L_SHAD |
| FE9h | FSR0H_SHAD |
| FEAh | FSR1L_SHAD |
| FEBh | FSR1H_SHAD |
| FECh | — |
| FEDh | STKPTR |
| FEEh | TOSL |
| FEFh | TOSH |

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


© 2010-2016 Microchip Technology Inc.

### 3.3.5 SPECIAL FUNCTION REGISTERS SUMMARY

The Special Function Register Summary for the device family are as follows:

| Device | Bank(s) | Page No. |
|---|---|---|
| PIC16(L)F1946/1947 | 0 | 30 |
| | 1 | 31 |
| | 2 | 32 |
| | 3 | 33 |
| | 4 | 34 |
| | 5 | 35 |
| | 6 | 36 |
| | 7 | 37 |
| | 8 | 38 |
| | 9-14 | 40 |
| | 15 | 41 |
| | 16-30 | 44 |
| | 31 | 45 |

**TABLE 3-10: 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 0** | | | | | | | | | | | |
| 000h(2) | INDF0 | Addressing this location uses contents of FSR0H/FSR0L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| 001h(2) | INDF1 | Addressing this location uses contents of FSR1H/FSR1L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| 002h(2) | PCL | Program Counter (PC) Least Significant Byte | | | | | | | | 0000 0000 | 0000 0000 |
| 003h(2) | STATUS | — | — | — | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | ---1 1000 | ---q quuu |
| 004h(2) | FSR0L | Indirect Data Memory Address 0 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| 005h(2) | FSR0H | Indirect Data Memory Address 0 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| 006h(2) | FSR1L | Indirect Data Memory Address 1 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| 007h(2) | FSR1H | Indirect Data Memory Address 1 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| 008h(2) | BSR | — | — | — | BSR<4:0> | | | | | ---0 0000 | ---0 0000 |
| 009h(2) | WREG | Working Register | | | | | | | | 0000 0000 | uuuu uuuu |
| 00Ah(1, 2) | PCLATH | — | Write Buffer for the upper 7 bits of the Program Counter | | | | | | | -000 0000 | -000 0000 |
| 00Bh(2) | INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 0000 000x | 0000 000u |
| 00Ch | PORTA | PORTA Data Latch when written: PORTA pins when read | | | | | | | | xxxx xxxx | uuuu uuuu |
| 00Dh | PORTB | PORTB Data Latch when written: PORTB pins when read | | | | | | | | xxxx xxxx | uuuu uuuu |
| 00Eh | PORTC | PORTC Data Latch when written: PORTC pins when read | | | | | | | | xxxx xxxx | uuuu uuuu |
| 00Fh | PORTD | PORTD Data Latch when written: PORTD pins when read | | | | | | | | xxxx xxxx | uuuu uuuu |
| 010h | PORTE | PORTE Data Latch when written: PORTE pins when read | | | | | | | | xxxx xxxx | xxxx uuuu |
| 011h | PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 0000 0000 |
| 012h | PIR2 | OSFIF | C2IF | C1IF | EEIF | BCLIF | LCDIF | C3IF | CCP2IF | 0000 0000 | 0000 0000 |
| 013h | PIR3 | — | CCP5IF | CCP4IF | CCP3IF | TMR6IF | — | TMR4IF | — | -000 0-0- | -000 0-0- |
| 014h | PIR4 | — | — | RC2IF | TX2IF | — | — | BCL2IF | SSP2IF | --00 --00 | --00 --00 |
| 015h | TMR0 | Timer0 Module Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 016h | TMR1L | Holding Register for the Least Significant Byte of the 16-bit TMR1 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 017h | TMR1H | Holding Register for the Most Significant Byte of the 16-bit TMR1 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 018h | T1CON | TMR1CS<1:0> | | T1CKPS<1:0> | | T1OSCEN | $\overline{T1SYNC}$ | — | TMR1ON | 0000 00-0 | uuuu uu-u |
| 019h | T1GCON | TMR1GE | T1GPOL | T1GTM | T1GSPM | T1GGO/$\overline{DONE}$ | T1GVAL | T1GSS<1:0> | | 0000 0x00 | uuuu uxuu |
| 01Ah | TMR2 | Timer 2 Module Register | | | | | | | | 0000 0000 | 0000 0000 |
| 01Bh | PR2 | Timer 2 Period Register | | | | | | | | 1111 1111 | 1111 1111 |
| 01Ch | T2CON | — | T2OUTPS<3:0> | | | | TMR2ON | T2CKPS<1:0> | | -000 0000 | -000 0000 |
| 01Dh | — | Unimplemented | | | | | | | | — | — |
| 01Eh | CPSCON0 | CPSON | CPSRM | — | — | CPSRNG1 | CPSRNG0 | CPSOUT | T0XCS | 00-- 0000 | 00-- 0000 |
| 01Fh | CPSCON1 | — | — | — | CPSCH<4:0> | | | | | ---0 0000 | ---0 0000 |

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

**Note 1:** The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<14:8>, whose contents are transferred to the upper byte of the program counter.

**2:** These registers can be addressed from any bank.

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


© 2010-2016 Microchip Technology Inc.

**TABLE 3-10: 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 1** | | | | | | | | | | | |
| 080h(2) | INDF0 | Addressing this location uses contents of FSR0H/FSR0L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| 081h(2) | INDF1 | Addressing this location uses contents of FSR1H/FSR1L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| 082h(2) | PCL | Program Counter (PC) Least Significant Byte | | | | | | | | 0000 0000 | 0000 0000 |
| 083h(2) | STATUS | — | — | — | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | ---1 1000 | ---q quuu |
| 084h(2) | FSR0L | Indirect Data Memory Address 0 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| 085h(2) | FSR0H | Indirect Data Memory Address 0 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| 086h(2) | FSR1L | Indirect Data Memory Address 1 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| 087h(2) | FSR1H | Indirect Data Memory Address 1 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| 088h(2) | BSR | — | — | — | BSR<4:0> | | | | | ---0 0000 | ---0 0000 |
| 089h(2) | WREG | Working Register | | | | | | | | 0000 0000 | uuuu uuuu |
| 08Ah(1, 2) | PCLATH | — | Write Buffer for the upper 7 bits of the Program Counter | | | | | | | -000 0000 | -000 0000 |
| 08Bh(2) | INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 0000 000x | 0000 000u |
| 08Ch | TRISA | PORTA Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| 08Dh | TRISB | PORTB Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| 08Eh | TRISC | PORTC Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| 08Fh | TRISD | PORTD Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| 090h | TRISE | PORTE Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| 091h | PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 0000 0000 |
| 092h | PIE2 | OSFIE | C2IE | C1IE | EEIE | BCLIE | LCDIE | C3IE | CCP2IE | 0000 0000 | 0000 0000 |
| 093h | PIE3 | — | CCP5IE | CCP4IE | CCP3IE | TMR6IE | — | TMR4IE | — | -000 0-0- | -000 0-0- |
| 094h | PIE4 | — | — | RC2IE | TX2IE | — | — | BCL2IE | SSP2IE | --00 --00 | --00 --00 |
| 095h | OPTION_REG | $\overline{WPUEN}$ | INTEDG | T0CS | T0SE | PSA | PS<2:0> | | | 1111 1111 | 1111 1111 |
| 096h | PCON | STKOVF | STKUNF | — | — | $\overline{RMCLR}$ | $\overline{RI}$ | $\overline{POR}$ | $\overline{BOR}$ | 00-- 11qq | qq-- qquu |
| 097h | WDTCON | — | — | WDTPS<4:0> | | | | | SWDTEN | --01 0110 | --01 0110 |
| 098h | OSCTUNE | — | — | TUN<5:0> | | | | | | --00 0000 | --00 0000 |
| 099h | OSCCON | SPLLEN | IRCF<3:0> | | | | — | SCS<1:0> | | 0011 1-00 | 0011 1-00 |
| 09Ah | OSCSTAT | T1OSCR | PLLR | OSTS | HFIOFR | HFIOFL | MFIOFR | LFIOFR | HFIOFS | 00q0 0q0- | qqqq qq0- |
| 09Bh | ADRESL | A/D Result Register Low | | | | | | | | xxxx xxxx | uuuu uuuu |
| 09Ch | ADRESH | A/D Result Register High | | | | | | | | xxxx xxxx | uuuu uuuu |
| 09Dh | ADCON0 | — | CHS<4:0> | | | | | GO/$\overline{DONE}$ | ADON | -000 0000 | -000 0000 |
| 09Eh | ADCON1 | ADFM | ADCS<2:0> | | | — | ADNREF | ADPREF1 | ADPREF0 | 0000 -000 | 0000 -000 |
| 09Fh | — | Unimplemented | | | | | | | | — | — |

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

**Note 1:** The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<14:8>, whose contents are transferred to the upper byte of the program counter.

**2:** These registers can be addressed from any bank.

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

**TABLE 3-10: 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 2** | | | | | | | | | | | |
| 100h(2) | INDF0 | Addressing this location uses contents of FSR0H/FSR0L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| 101h(2) | INDF1 | Addressing this location uses contents of FSR1H/FSR1L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| 102h(2) | PCL | Program Counter (PC) Least Significant Byte | | | | | | | | 0000 0000 | 0000 0000 |
| 103h(2) | STATUS | — | — | — | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | ---1 1000 | ---q quuu |
| 104h(2) | FSR0L | Indirect Data Memory Address 0 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| 105h(2) | FSR0H | Indirect Data Memory Address 0 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| 106h(2) | FSR1L | Indirect Data Memory Address 1 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| 107h(2) | FSR1H | Indirect Data Memory Address 1 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| 108h(2) | BSR | — | — | — | BSR<4:0> | | | | | ---0 0000 | ---0 0000 |
| 109h(2) | WREG | Working Register | | | | | | | | 0000 0000 | uuuu uuuu |
| 10Ah(1, 2) | PCLATH | — | Write Buffer for the upper 7 bits of the Program Counter | | | | | | | -000 0000 | -000 0000 |
| 10Bh(2) | INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 0000 000x | 0000 000u |
| 10Ch | LATA | PORTA Data Latch | | | | | | | | xxxx xxxx | uuuu uuuu |
| 10Dh | LATB | PORTB Data Latch | | | | | | | | xxxx xxxx | uuuu uuuu |
| 10Eh | LATC | PORTC Data Latch | | | | | | | | xxxx xxxx | uuuu uuuu |
| 10Fh | LATD | PORTD Data Latch | | | | | | | | xxxx xxxx | uuuu uuuu |
| 110h | LATE | PORTE Data Latch | | | | | | | | xxxx xxxx | uuuu uuuu |
| 111h | CM1CON0 | C1ON | C1OUT | C1OE | C1POL | — | C1SP | C1HYS | C1SYNC | 0000 -100 | 0000 -100 |
| 112h | CM1CON1 | C1INTP | C1INTN | C1PCH1 | C1PCH0 | — | — | C1NCH<1:0> | | 0000 --00 | 0000 --00 |
| 113h | CM2CON0 | C2ON | C2OUT | C2OE | C2POL | — | C2SP | C2HYS | C2SYNC | 0000 -100 | 0000 -100 |
| 114h | CM2CON1 | C2INTP | C2INTN | C2PCH1 | C2PCH0 | — | — | C2NCH<1:0> | | 0000 --00 | 0000 --00 |
| 115h | CMOUT | — | — | — | — | — | MC3OUT | MC2OUT | MC1OUT | ---- -000 | ---- -000 |
| 116h | BORCON | SBOREN | — | — | — | — | — | — | BORRDY | 1--- ---q | u--- ---u |
| 117h | FVRCON | FVREN | FVRRDY | TSEN | TSRNG | CDAFVR1 | CDAFVR0 | ADFVR<1:0> | | 0q00 0000 | 0q00 0000 |
| 118h | DACCON0 | DACEN | DACLPS | DACOE | — | DACPSS<1:0> | | — | DACNSS | 000- 00-0 | 000- 00-0 |
| 119h | DACCON1 | — | — | — | DACR<4:0> | | | | | ---0 0000 | ---0 0000 |
| 11Ah | SRCON0 | SRLEN | SRCLK2 | SRCLK1 | SRCLK0 | SRQEN | SRNQEN | SRPS | SRPR | 0000 0000 | 0000 0000 |
| 11Bh | SRCON1 | SRSPE | SRSCKE | SRSC2E | SRSC1E | SRRPE | SRRCKE | SRRC2E | SRRC1E | 0000 0000 | 0000 0000 |
| 11Ch | — | Unimplemented | | | | | | | | — | — |
| 11Dh | APFCON | P3CSEL | P3BSEL | P2DSEL | P2CSEL | P2BSEL | CCP2SEL | P1CSEL | P1BSEL | 0000 0000 | 0000 0000 |
| 11Eh | CM3CON0 | C3ON | C3OUT | C3OE | C3POL | — | C3SP | C3HYS | C3SYNC | 0000 -100 | 0000 -100 |
| 11Fh | CM3CON1 | C3INTP | C3INTN | C3PCH1 | C3PCH0 | — | — | C3NCH<1:0> | | 0000 --00 | 0000 --00 |

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

**Note 1:** The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<14:8>, whose contents are transferred to the upper byte of the program counter.

**2:** These registers can be addressed from any bank.

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


© 2010-2016 Microchip Technology Inc.

**TABLE 3-10: 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 3** | | | | | | | | | | | |
| 180h(2) | INDF0 | Addressing this location uses contents of FSR0H/FSR0L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| 181h(2) | INDF1 | Addressing this location uses contents of FSR1H/FSR1L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| 182h(2) | PCL | Program Counter (PC) Least Significant Byte | | | | | | | | 0000 0000 | 0000 0000 |
| 183h(2) | STATUS | — | — | — | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | ---1 1000 | ---q quuu |
| 184h(2) | FSR0L | Indirect Data Memory Address 0 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| 185h(2) | FSR0H | Indirect Data Memory Address 0 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| 186h(2) | FSR1L | Indirect Data Memory Address 1 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| 187h(2) | FSR1H | Indirect Data Memory Address 1 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| 188h(2) | BSR | — | — | — | BSR<4:0> | | | | | ---0 0000 | ---0 0000 |
| 189h(2) | WREG | Working Register | | | | | | | | 0000 0000 | uuuu uuuu |
| 18Ah(1, 2) | PCLATH | — | Write Buffer for the upper 7 bits of the Program Counter | | | | | | | -000 0000 | -000 0000 |
| 18Bh(2) | INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 0000 000x | 0000 000u |
| 18Ch | ANSELA | — | — | ANSA5 | — | ANSA3 | ANSA2 | ANSA1 | ANSA0 | --1- 1111 | --1- 1111 |
| 18Dh | — | Unimplemented | | | | | | | | — | — |
| 18Eh | — | Unimplemented | | | | | | | | — | — |
| 18Fh | — | Unimplemented | | | | | | | | — | — |
| 190h | ANSELE | — | — | — | — | — | ANSE2 | ANSE1 | ANSE0 | ---- -111 | ---- -111 |
| 191h | EEADRL | EEPROM / Program Memory Address Register Low Byte | | | | | | | | 0000 0000 | 0000 0000 |
| 192h | EEADRH | —(3) | EEPROM / Program Memory Address Register High Byte | | | | | | | 1000 0000 | 1000 0000 |
| 193h | EEDATL | EEPROM / Program Memory Read Data Register Low Byte | | | | | | | | xxxx xxxx | uuuu uuuu |
| 194h | EEDATH | — | — | EEPROM / Program Memory Read Data Register High Byte | | | | | | --xx xxxx | --uu uuuu |
| 195h | EECON1 | EEPGD | CFGS | LWLO | FREE | WRERR | WREN | WR | RD | 0000 x000 | 0000 q000 |
| 196h | EECON2 | EEPROM control register 2 | | | | | | | | 0000 0000 | 0000 0000 |
| 197h | — | Unimplemented | | | | | | | | — | — |
| 198h | — | Unimplemented | | | | | | | | — | — |
| 199h | RC1REG | USART Receive Data Register | | | | | | | | 0000 0000 | 0000 0000 |
| 19Ah | TX1REG | USART Transmit Data Register | | | | | | | | 0000 0000 | 0000 0000 |
| 19Bh | SP1BRGL | EUSART1 Baud Rate Generator, Low Byte | | | | | | | | 0000 0000 | 0000 0000 |
| 19Ch | SP1BRGH | EUSART1 Baud Rate Generator, High Byte | | | | | | | | 0000 0000 | 0000 0000 |
| 19Dh | RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 0000 000x | 0000 000x |
| 19Eh | TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 0000 0010 | 0000 0010 |
| 19Fh | BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 01-0 0-00 | 01-0 0-00 |

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

**Note 1:** The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<14:8>, whose contents are transferred to the upper byte of the program counter.

**2:** These registers can be addressed from any bank.

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

**TABLE 3-10: 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 4** | | | | | | | | | | | |
| 200h(2) | INDF0 | Addressing this location uses contents of FSR0H/FSR0L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| 201h(2) | INDF1 | Addressing this location uses contents of FSR1H/FSR1L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| 202h(2) | PCL | Program Counter (PC) Least Significant Byte | | | | | | | | 0000 0000 | 0000 0000 |
| 203h(2) | STATUS | — | — | — | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | ---1 1000 | ---q quuu |
| 204h(2) | FSR0L | Indirect Data Memory Address 0 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| 205h(2) | FSR0H | Indirect Data Memory Address 0 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| 206h(2) | FSR1L | Indirect Data Memory Address 1 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| 207h(2) | FSR1H | Indirect Data Memory Address 1 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| 208h(2) | BSR | — | — | — | BSR<4:0> | | | | | ---0 0000 | ---0 0000 |
| 209h(2) | WREG | Working Register | | | | | | | | 0000 0000 | uuuu uuuu |
| 20Ah(1, 2) | PCLATH | — | Write Buffer for the upper 7 bits of the Program Counter | | | | | | | -000 0000 | -000 0000 |
| 20Bh(2) | INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 0000 000x | 0000 000u |
| 20Ch | — | Unimplemented | | | | | | | | — | — |
| 20Dh | WPUB | WPUB7 | WPUB6 | WPUB5 | WPUB4 | WPUB3 | WPUB2 | WPUB1 | WPUB0 | 1111 1111 | 1111 1111 |
| 20Eh | — | Unimplemented | | | | | | | | — | — |
| 20Fh | — | Unimplemented | | | | | | | | — | — |
| 210h | — | Unimplemented | | | | | | | | — | — |
| 211h | SSP1BUF | Synchronous Serial Port Receive Buffer/Transmit Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 212h | SSP1ADD | ADD<7:0> | | | | | | | | 0000 0000 | 0000 0000 |
| 213h | SSP1MSK | MSK<7:0> | | | | | | | | 1111 1111 | 1111 1111 |
| 214h | SSP1STAT | SMP | CKE | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF | 0000 0000 | 0000 0000 |
| 215h | SSP1CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM<3:0> | | | | 0000 0000 | 0000 0000 |
| 216h | SSP1CON2 | GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN | 0000 0000 | 0000 0000 |
| 217h | SSP1CON3 | ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN | 0000 0000 | 0000 0000 |
| 218h | — | Unimplemented | | | | | | | | — | — |
| 219h | SSP2BUF | Synchronous Serial Port Receive Buffer/Transmit Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 21Ah | SSP2ADD | ADD<7:0> | | | | | | | | 0000 0000 | 0000 0000 |
| 21Bh | SSP2MSK | MSK<7:0> | | | | | | | | 1111 1111 | 1111 1111 |
| 21Ch | SSP2STAT | SMP | CKE | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF | 0000 0000 | 0000 0000 |
| 21Dh | SSP2CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM<3:0> | | | | 0000 0000 | 0000 0000 |
| 21Eh | SSP2CON2 | GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN | 0000 0000 | 0000 0000 |
| 21Fh | SSP2CON3 | ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN | 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'.

**Note 1:** The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<14:8>, whose contents are transferred to the upper byte of the program counter.

**2:** These registers can be addressed from any bank.

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


© 2010-2016 Microchip Technology Inc.

**TABLE 3-10: 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 5** | | | | | | | | | | | |
| 280h(2) | INDF0 | Addressing this location uses contents of FSR0H/FSR0L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| 281h(2) | INDF1 | Addressing this location uses contents of FSR1H/FSR1L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| 282h(2) | PCL | Program Counter (PC) Least Significant Byte | | | | | | | | 0000 0000 | 0000 0000 |
| 283h(2) | STATUS | — | — | — | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | ---1 1000 | ---q quuu |
| 284h(2) | FSR0L | Indirect Data Memory Address 0 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| 285h(2) | FSR0H | Indirect Data Memory Address 0 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| 286h(2) | FSR1L | Indirect Data Memory Address 1 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| 287h(2) | FSR1H | Indirect Data Memory Address 1 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| 288h(2) | BSR | — | — | — | BSR<4:0> | | | | | ---0 0000 | ---0 0000 |
| 289h(2) | WREG | Working Register | | | | | | | | 0000 0000 | uuuu uuuu |
| 28Ah(1, 2) | PCLATH | — | Write Buffer for the upper 7 bits of the Program Counter | | | | | | | -000 0000 | -000 0000 |
| 28Bh(2) | INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 0000 000x | 0000 000u |
| 28Ch | PORTF | PORTF Data Latch when written: PORTF pins when read | | | | | | | | xxxx xxxx | uuuu uuuu |
| 28Dh | PORTG | — | — | RG5 | RG4 | RG3 | RG2 | RG1 | RG0 | --xx xxxx | --uu uuuu |
| 28Eh | — | Unimplemented | | | | | | | | — | — |
| 28Fh | — | Unimplemented | | | | | | | | — | — |
| 290h | — | Unimplemented | | | | | | | | — | — |
| 291h | CCPR1L | Capture/Compare/PWM Register 1 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 292h | CCPR1H | Capture/Compare/PWM Register 1 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 293h | CCP1CON | P1M<1:0> | | DC1B<1:0> | | CCP1M<3:0> | | | | 0000 0000 | 0000 0000 |
| 294h | PWM1CON | P1RSEN | P1DC<6:0> | | | | | | | 0000 0000 | 0000 0000 |
| 295h | CCP1AS | CCP1ASE | CCP1AS<2:0> | | | PSS1AC<1:0> | | PSS1BD<1:0> | | 0000 0000 | 0000 0000 |
| 296h | PSTR1CON | — | — | — | STR1SYNC | STR1D | STR1C | STR1B | STR1A | ---0 0001 | ---0 0001 |
| 297h | — | Unimplemented | | | | | | | | — | — |
| 298h | CCPR2L | Capture/Compare/PWM Register 2 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 299h | CCPR2H | Capture/Compare/PWM Register 2 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 29Ah | CCP2CON | P2M<1:0> | | DC2B<1:0> | | CCP2M<3:0> | | | | 0000 0000 | 0000 0000 |
| 29Bh | PWM2CON | P2RSEN | P2DC<6:0> | | | | | | | 0000 0000 | 0000 0000 |
| 29Ch | CCP2AS | CCP2ASE | CCP2AS<2:0> | | | PSS2AC<1:0> | | PSS2BD<1:0> | | 0000 0000 | 0000 0000 |
| 29Dh | PSTR2CON | — | — | — | STR2SYNC | STR2D | STR2C | STR2B | STR2A | ---0 0001 | ---0 0001 |
| 29Eh | CCPTMRS0 | C4TSEL1 | C4TSEL0 | C3TSEL1 | C3TSEL0 | C2TSEL1 | C2TSEL0 | C1TSEL1 | C1TSEL0 | 0000 0000 | 0000 0000 |
| 29Fh | CCPTMRS1 | — | — | — | — | — | — | C5TSEL<1:0> | | ---- --00 | ---- --00 |

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

**Note 1:** The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<14:8>, whose contents are transferred to the upper byte of the program counter.

**2:** These registers can be addressed from any bank.

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

**TABLE 3-10: 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 6** | | | | | | | | | | | |
| 300h(2) | INDF0 | Addressing this location uses contents of FSR0H/FSR0L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| 301h(2) | INDF1 | Addressing this location uses contents of FSR1H/FSR1L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| 302h(2) | PCL | Program Counter (PC) Least Significant Byte | | | | | | | | 0000 0000 | 0000 0000 |
| 303h(2) | STATUS | — | — | — | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | ---1 1000 | ---q quuu |
| 304h(2) | FSR0L | Indirect Data Memory Address 0 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| 305h(2) | FSR0H | Indirect Data Memory Address 0 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| 306h(2) | FSR1L | Indirect Data Memory Address 1 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| 307h(2) | FSR1H | Indirect Data Memory Address 1 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| 308h(2) | BSR | — | — | — | BSR<4:0> | | | | | ---0 0000 | ---0 0000 |
| 309h(2) | WREG | Working Register | | | | | | | | 0000 0000 | uuuu uuuu |
| 30Ah(1, 2) | PCLATH | — | Write Buffer for the upper 7 bits of the Program Counter | | | | | | | -000 0000 | -000 0000 |
| 30Bh(2) | INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 0000 000x | 0000 000u |
| 30Ch | TRISF | PORTF Data Direction Register | | | | | | | | 1111 1111 | 1111 1111 |
| 30Dh | TRISG | — | — | TRISG5 | TRISG4 | TRISG3 | TRISG2 | TRISG1 | TRISG0 | --11 1111 | --11 1111 |
| 30Eh | — | Unimplemented | | | | | | | | — | — |
| 30Fh | — | Unimplemented | | | | | | | | — | — |
| 310h | — | Unimplemented | | | | | | | | — | — |
| 311h | CCPR3L | Capture/Compare/PWM Register 3 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 312h | CCPR3H | Capture/Compare/PWM Register 3 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 313h | CCP3CON | P3M<1:0> | | DC3B<1:0> | | CCP3M<1:0> | | | | 0000 0000 | 0000 0000 |
| 314h | PWM3CON | P3RSEN | P3DC<6:0> | | | | | | | 0000 0000 | 0000 0000 |
| 315h | CCP3AS | CCP3ASE | CCP3AS<2:0> | | | PSS3AC<1:0> | | PSS3BD<1:0> | | 0000 0000 | 0000 0000 |
| 316h | PSTR3CON | — | — | — | STR3SYNC | STR3D | STR3C | STR3B | STR3A | ---0 0001 | ---0 0001 |
| 317h | — | Unimplemented | | | | | | | | — | — |
| 318h | CCPR4L | Capture/Compare/PWM Register 4 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 319h | CCPR4H | Capture/Compare/PWM Register 4 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 31Ah | CCP4CON | — | — | DC4B<1:0> | | CCP4M<3:0> | | | | --00 0000 | --00 0000 |
| 31Bh | — | Unimplemented | | | | | | | | — | — |
| 31Ch | CCPR5L | Capture/Compare/PWM Register 5 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 31Dh | CCPR5H | Capture/Compare/PWM Register 5 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 31Eh | CCP5CON | — | — | DC5B<1:0> | | CCP5M<3:0> | | | | --00 0000 | --00 0000 |
| 31Fh | — | Unimplemented | | | | | | | | — | — |

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

**Note 1:** The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<14:8>, whose contents are transferred to the upper byte of the program counter.

**2:** These registers can be addressed from any bank.

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


© 2010-2016 Microchip Technology Inc.

**TABLE 3-10: 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 7** | | | | | | | | | | | |
| 380h(2) | INDF0 | Addressing this location uses contents of FSR0H/FSR0L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| 381h(2) | INDF1 | Addressing this location uses contents of FSR1H/FSR1L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| 382h(2) | PCL | Program Counter (PC) Least Significant Byte | | | | | | | | 0000 0000 | 0000 0000 |
| 383h(2) | STATUS | — | — | — | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | ---1 1000 | ---q quuu |
| 384h(2) | FSR0L | Indirect Data Memory Address 0 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| 385h(2) | FSR0H | Indirect Data Memory Address 0 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| 386h(2) | FSR1L | Indirect Data Memory Address 1 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| 387h(2) | FSR1H | Indirect Data Memory Address 1 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| 388h(2) | BSR | — | — | — | BSR<4:0> | | | | | ---0 0000 | ---0 0000 |
| 389h(2) | WREG | Working Register | | | | | | | | 0000 0000 | uuuu uuuu |
| 38Ah(1, 2) | PCLATH | — | Write Buffer for the upper 7 bits of the Program Counter | | | | | | | -000 0000 | -000 0000 |
| 38Bh(2) | INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 0000 000x | 0000 000u |
| 38Ch | LATF | PORTF Data Latch | | | | | | | | xxxx xxxx | uuuu uuuu |
| 38Dh | LATG | — | — | LATG5 | LATG4 | LATG3 | LATG2 | LATG1 | LATG0 | --xx xxxx | --uu uuuu |
| 38Eh | — | Unimplemented | | | | | | | | — | — |
| 38Fh | — | Unimplemented | | | | | | | | — | — |
| 390h | — | Unimplemented | | | | | | | | — | — |
| 391h | — | Unimplemented | | | | | | | | — | — |
| 392h | — | Unimplemented | | | | | | | | — | — |
| 393h | — | Unimplemented | | | | | | | | — | — |
| 394h | IOCBP | IOCBP7 | IOCBP6 | IOCBP5 | IOCBP4 | IOCBP3 | IOCBP2 | IOCBP1 | IOCBP0 | 0000 0000 | 0000 0000 |
| 395h | IOCBN | IOCBN7 | IOCBN6 | IOCBN5 | IOCBN4 | IOCBN3 | IOCBN2 | IOCBN1 | IOCBN0 | 0000 0000 | 0000 0000 |
| 396h | IOCBF | IOCBF7 | IOCBF6 | IOCBF5 | IOCBF4 | IOCBF3 | IOCBF2 | IOCBF1 | IOCBF0 | 0000 0000 | 0000 0000 |
| 397h | — | Unimplemented | | | | | | | | — | — |
| 398h | — | Unimplemented | | | | | | | | — | — |
| 399h | — | Unimplemented | | | | | | | | — | — |
| 39Ah | — | Unimplemented | | | | | | | | — | — |
| 39Bh | — | Unimplemented | | | | | | | | — | — |
| 39Ch | — | Unimplemented | | | | | | | | — | — |
| 39Dh | — | Unimplemented | | | | | | | | — | — |
| 39Eh | — | Unimplemented | | | | | | | | — | — |
| 39Fh | — | Unimplemented | | | | | | | | — | — |

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

**Note 1:** The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<14:8>, whose contents are transferred to the upper byte of the program counter.

**2:** These registers can be addressed from any bank.

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

**TABLE 3-10: 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 8** | | | | | | | | | | | |
| 400h(2) | INDF0 | Addressing this location uses contents of FSR0H/FSR0L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| 401h(2) | INDF1 | Addressing this location uses contents of FSR1H/FSR1L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| 402h(2) | PCL | Program Counter (PC) Least Significant Byte | | | | | | | | 0000 0000 | 0000 0000 |
| 403h(2) | STATUS | — | — | — | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | ---1 1000 | ---q quuu |
| 404h(2) | FSR0L | Indirect Data Memory Address 0 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| 405h(2) | FSR0H | Indirect Data Memory Address 0 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| 406h(2) | FSR1L | Indirect Data Memory Address 1 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| 407h(2) | FSR1H | Indirect Data Memory Address 1 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| 408h(2) | BSR | — | — | — | BSR<4:0> | | | | | ---0 0000 | ---0 0000 |
| 409h(2) | WREG | Working Register | | | | | | | | 0000 0000 | uuuu uuuu |
| 40Ah(1, 2) | PCLATH | — | Write Buffer for the upper 7 bits of the Program Counter | | | | | | | -000 0000 | -000 0000 |
| 40Bh(2) | INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 0000 000x | 0000 000u |
| 40Ch | ANSELF | ANSELF7 | ANSELF6 | ANSELF5 | ANSELF4 | ANSELF3 | ANSELF2 | ANSELF1 | ANSELF0 | 1111 1111 | 1111 1111 |
| 40Dh | ANSELG | — | — | — | ANSELG4 | ANSELG3 | ANSELG2 | ANSELG1 | — | ---1 111- | ---1 111- |
| 40Eh | — | Unimplemented | | | | | | | | — | — |
| 40Fh | — | Unimplemented | | | | | | | | — | — |
| 410h | — | Unimplemented | | | | | | | | — | — |
| 411h | — | Unimplemented | | | | | | | | — | — |
| 412h | — | Unimplemented | | | | | | | | — | — |
| 413h | — | Unimplemented | | | | | | | | — | — |
| 414h | — | Unimplemented | | | | | | | | — | — |
| 415h | TMR4 | Timer 4 Module Register | | | | | | | | 0000 0000 | 0000 0000 |
| 416h | PR4 | Timer 4 Period Register | | | | | | | | 1111 1111 | 1111 1111 |
| 417h | T4CON | — | T4OUTPS<3:0> | | | | TMR4ON | T4CKPS<1:0> | | -000 0000 | -000 0000 |
| 418h | — | Unimplemented | | | | | | | | — | — |
| 419h | — | Unimplemented | | | | | | | | — | — |
| 41Ah | — | Unimplemented | | | | | | | | — | — |
| 41Bh | — | Unimplemented | | | | | | | | — | — |
| 41Ch | TMR6 | Timer 6 Module Register | | | | | | | | 0000 0000 | 0000 0000 |
| 41Dh | PR6 | Timer 6 Period Register | | | | | | | | 1111 1111 | 1111 1111 |
| 41Eh | T6CON | — | T6OUTPS<3:0> | | | | TMR6ON | T6CKPS<1:0> | | -000 0000 | -000 0000 |
| 41Fh | — | Unimplemented | | | | | | | | — | — |

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

**Note 1:** The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<14:8>, whose contents are transferred to the upper byte of the program counter.

**2:** These registers can be addressed from any bank.

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


© 2010-2016 Microchip Technology Inc.

**TABLE 3-10: 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 9** | | | | | | | | | | | |
| 480h(2) | INDF0 | Addressing this location uses contents of FSR0H/FSR0L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| 481h(2) | INDF1 | Addressing this location uses contents of FSR1H/FSR1L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| 482h(2) | PCL | Program Counter (PC) Least Significant Byte | | | | | | | | 0000 0000 | 0000 0000 |
| 483h(2) | STATUS | — | — | — | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | ---1 1000 | ---q quuu |
| 404h(2) | FSR0L | Indirect Data Memory Address 0 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| 485h(2) | FSR0H | Indirect Data Memory Address 0 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| 486h(2) | FSR1L | Indirect Data Memory Address 1 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| 487h(2) | FSR1H | Indirect Data Memory Address 1 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| 488h(2) | BSR | — | — | — | BSR<4:0> | | | | | ---0 0000 | ---0 0000 |
| 489h(2) | WREG | Working Register | | | | | | | | 0000 0000 | uuuu uuuu |
| 48Ah(1, 2) | PCLATH | — | Write Buffer for the upper 7 bits of the Program Counter | | | | | | | -000 0000 | -000 0000 |
| 48Bh(2) | INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 0000 000x | 0000 000u |
| 48Ch | — | Unimplemented | | | | | | | | — | — |
| 48Dh | WPUG | — | — | WPUG5 | — | — | — | — | — | --1- ---- | --1- ---- |
| 48Eh | — | Unimplemented | | | | | | | | — | — |
| 48Fh | — | Unimplemented | | | | | | | | — | — |
| 490h | — | Unimplemented | | | | | | | | — | — |
| 491h | RC2REG | USART Receive Data Register | | | | | | | | 0000 0000 | 0000 0000 |
| 492h | TX2REG | USART Transmit Data Register | | | | | | | | 0000 0000 | 0000 0000 |
| 493h | SP2BRGL | EUSART2 Baud Rate Generator, Low Byte | | | | | | | | 0000 0000 | 0000 0000 |
| 494h | SP2BRGH | EUSART2 Baud Rate Generator, High Byte | | | | | | | | 0000 0000 | 0000 0000 |
| 495h | RC2STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 0000 000x | 0000 000x |
| 496h | TX2STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 0000 0010 | 0000 0010 |
| 497h | BAUD2CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 01-0 0-00 | 01-0 0-00 |
| 498h | — | Unimplemented | | | | | | | | — | — |
| 499h | — | Unimplemented | | | | | | | | — | — |
| 49Ah | — | Unimplemented | | | | | | | | — | — |
| 49Bh | — | Unimplemented | | | | | | | | — | — |
| 49Ch | — | Unimplemented | | | | | | | | — | — |
| 49Dh | — | Unimplemented | | | | | | | | — | — |
| 49Eh | — | Unimplemented | | | | | | | | — | — |
| 49Fh | — | Unimplemented | | | | | | | | — | — |

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

**Note 1:** The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<14:8>, whose contents are transferred to the upper byte of the program counter.

**2:** These registers can be addressed from any bank.

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

**TABLE 3-10: 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Banks 10-14** | | | | | | | | | | | |
| x00h/ x80h(2) | INDF0 | Addressing this location uses contents of FSR0H/FSR0L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| x00h/ x81h(2) | INDF1 | Addressing this location uses contents of FSR1H/FSR1L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| x02h/ x82h(2) | PCL | Program Counter (PC) Least Significant Byte | | | | | | | | 0000 0000 | 0000 0000 |
| x03h/ x83h(2) | STATUS | — | — | — | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | ---1 1000 | ---q quuu |
| x04h/ x84h(2) | FSR0L | Indirect Data Memory Address 0 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| x05h/ x85h(2) | FSR0H | Indirect Data Memory Address 0 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| x06h/ x86h(2) | FSR1L | Indirect Data Memory Address 1 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| x07h/ x87h(2) | FSR1H | Indirect Data Memory Address 1 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| x08h/ x88h(2) | BSR | — | — | — | BSR<4:0> | | | | | ---0 0000 | ---0 0000 |
| x09h/ x89h(2) | WREG | Working Register | | | | | | | | 0000 0000 | uuuu uuuu |
| x0Ah/ x8Ah(1),(2) | PCLATH | — | Write Buffer for the upper 7 bits of the Program Counter | | | | | | | -000 0000 | -000 0000 |
| x0Bh/ x8Bh(2) | INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 0000 000x | 0000 000u |
| x0Ch/ x8Ch — x1Fh/ x9Fh | — | Unimplemented | | | | | | | | — | — |

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

**Note 1:** The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<14:8>, whose contents are transferred to the upper byte of the program counter.

**2:** These registers can be addressed from any bank.

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


© 2010-2016 Microchip Technology Inc.

**TABLE 3-10: 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 15** | | | | | | | | | | | |
| 780h(2) | INDF0 | Addressing this location uses contents of FSR0H/FSR0L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| 781h(2) | INDF1 | Addressing this location uses contents of FSR1H/FSR1L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| 782h(2) | PCL | Program Counter (PC) Least Significant Byte | | | | | | | | 0000 0000 | 0000 0000 |
| 783h(2) | STATUS | — | — | — | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | ---1 1000 | ---q quuu |
| 784h(2) | FSR0L | Indirect Data Memory Address 0 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| 785h(2) | FSR0H | Indirect Data Memory Address 0 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| 786h(2) | FSR1L | Indirect Data Memory Address 1 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| 787h(2) | FSR1H | Indirect Data Memory Address 1 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| 788h(2) | BSR | — | — | — | BSR<4:0> | | | | | ---0 0000 | ---0 0000 |
| 789h(2) | WREG | Working Register | | | | | | | | 0000 0000 | uuuu uuuu |
| 78Ah(1, 2) | PCLATH | — | Write Buffer for the upper 7 bits of the Program Counter | | | | | | | -000 0000 | -000 0000 |
| 78Bh(2) | INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 0000 000x | 0000 000u |
| 78Ch | — | Unimplemented | | | | | | | | — | — |
| 78Dh | — | Unimplemented | | | | | | | | — | — |
| 78Eh | — | Unimplemented | | | | | | | | — | — |
| 78Fh | — | Unimplemented | | | | | | | | — | — |
| 790h | — | Unimplemented | | | | | | | | — | — |
| 791h | LCDCON | LCDEN | SLPEN | WERR | — | CS<1:0> | | LMUX<1:0> | | 000- 0011 | 000- 0011 |
| 792h | LCDPS | WFT | BIASMD | LCDA | WA | LP<3:0> | | | | 0000 0000 | 0000 0000 |
| 793h | LCDREF | LCDIRE | LCDIRS | LCDIRI | — | VLCD3PE | VLCD2PE | VLCD1PE | — | 000- 000- | 000- 000- |
| 794h | LCDCST | — | — | — | — | — | LCDCST<2:0> | | | ---- -000 | ---- -000 |
| 795h | LCDRL | LRLAP<1:0> | | LRLBP<1:0> | | — | LRLAT<2:0> | | | 0000 -000 | 0000 -000 |
| 796h | — | Unimplemented | | | | | | | | — | — |
| 797h | — | Unimplemented | | | | | | | | — | — |
| 798h | LCDSE0 | SE<7:0> | | | | | | | | 0000 0000 | uuuu uuuu |
| 799h | LCDSE1 | SE<15:8> | | | | | | | | 0000 0000 | uuuu uuuu |
| 79Ah | LCDSE2 | SE<23:16> | | | | | | | | 0000 0000 | uuuu uuuu |
| 79Bh | LCDSE3 | SE<31:24> | | | | | | | | 0000 0000 | uuuu uuuu |
| 79Ch | LCDSE4 | SE<39:32> | | | | | | | | 0000 0000 | uuuu uuuu |
| 79Dh | LCDSE5 | — | — | SE<45:40> | | | | | | --00 0000 | --uu uuuu |
| 79Eh | — | Unimplemented | | | | | | | | — | — |

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

**Note 1:** The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<14:8>, whose contents are transferred to the upper byte of the program counter.

**2:** These registers can be addressed from any bank.

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

**TABLE 3-10: 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 79Fh | — | Unimplemented | | | | | | | | — | — |
| 7A0h | LCDDATA0 | SEG7 COM0 | SEG6 COM0 | SEG5 COM0 | SEG4 COM0 | SEG3 COM0 | SEG2 COM0 | SEG1 COM0 | SEG0 COM0 | xxxx xxxx | uuuu uuuu |
| 7A1h | LCDDATA1 | SEG15 COM0 | SEG14 COM0 | SEG13 COM0 | SEG12 COM0 | SEG11 COM0 | SEG10 COM0 | SEG9 COM0 | SEG8 COM0 | xxxx xxxx | uuuu uuuu |
| 7A2h | LCDDATA2 | SEG23 COM0 | SEG22 COM0 | SEG21 COM0 | SEG20 COM0 | SEG19 COM0 | SEG18 COM0 | SEG17 COM0 | SEG16 COM0 | xxxx xxxx | uuuu uuuu |
| 7A3h | LCDDATA3 | SEG7 COM1 | SEG6 COM1 | SEG5 COM1 | SEG4 COM1 | SEG3 COM1 | SEG2 COM1 | SEG1 COM1 | SEG0 COM1 | xxxx xxxx | uuuu uuuu |
| 7A4h | LCDDATA4 | SEG15 COM1 | SEG14 COM1 | SEG13 COM1 | SEG12 COM1 | SEG11 COM1 | SEG10 COM1 | SEG9 COM1 | SEG8 COM1 | xxxx xxxx | uuuu uuuu |
| 7A5h | LCDDATA5 | SEG23 COM1 | SEG22 COM1 | SEG21 COM1 | SEG20 COM1 | SEG19 COM1 | SEG18 COM1 | SEG17 COM1 | SEG16 COM1 | xxxx xxxx | uuuu uuuu |

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

**Note 1:** The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<14:8>, whose contents are transferred to the upper byte of the program counter.

**2:** These registers can be addressed from any bank.

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


© 2010-2016 Microchip Technology Inc.

**TABLE 3-10: 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 15 (Continued)** | | | | | | | | | | | |
| 7A6h | LCDDATA6 | SEG7 COM2 | SEG6 COM2 | SEG5 COM2 | SEG4 COM2 | SEG3 COM2 | SEG2 COM2 | SEG1 COM2 | SEG0 COM2 | xxxx xxxx | uuuu uuuu |
| 7A7h | LCDDATA7 | SEG15 COM2 | SEG14 COM2 | SEG13 COM2 | SEG12 COM2 | SEG11 COM2 | SEG10 COM2 | SEG9 COM2 | SEG8 COM2 | xxxx xxxx | uuuu uuuu |
| 7A8h | LCDDATA8 | SEG23 COM2 | SEG22 COM2 | SEG21 COM2 | SEG20 COM2 | SEG19 COM2 | SEG18 COM2 | SEG17 COM2 | SEG16 COM2 | xxxx xxxx | uuuu uuuu |
| 7A9h | LCDDATA9 | SEG7 COM3 | SEG6 COM3 | SEG5 COM3 | SEG4 COM3 | SEG3 COM3 | SEG2 COM3 | SEG1 COM3 | SEG0 COM3 | xxxx xxxx | uuuu uuuu |
| 7AAh | LCDDATA10 | SEG15 COM3 | SEG14 COM3 | SEG13 COM3 | SEG12 COM3 | SEG11 COM3 | SEG10 COM3 | SEG9 COM3 | SEG8 COM3 | xxxx xxxx | uuuu uuuu |
| 7ABh | LCDDATA11 | SEG23 COM3 | SEG22 COM3 | SEG21 COM3 | SEG20 COM3 | SEG19 COM3 | SEG18 COM3 | SEG17 COM3 | SEG16 COM3 | xxxx xxxx | uuuu uuuu |
| 7ACh | LCDDATA12 | SEG31 COM0 | SEG30 COM0 | SEG29 COM0 | SEG28 COM0 | SEG27 COM0 | SEG26 COM0 | SEG25 COM0 | SEG24 COM0 | xxxx xxxx | uuuu uuuu |
| 7ADh | LCDDATA13 | SEG39 COM0 | SEG38 COM0 | SEG37 COM0 | SEG36 COM0 | SEG35 COM0 | SEG34 COM0 | SEG33 COM0 | SEG32 COM0 | xxxx xxxx | uuuu uuuu |
| 7AEh | LCDDATA14 | — | — | SEG45 COM0 | SEG44 COM0 | SEG43 COM0 | SEG42 COM0 | SEG41 COM0 | SEG40 COM0 | --xx xxxx | --uu uuuu |
| 7AFh | LCDDATA15 | SEG31 COM1 | SEG30 COM1 | SEG29 COM1 | SEG28 COM1 | SEG27 COM1 | SEG26 COM1 | SEG25 COM1 | SEG24 COM1 | xxxx xxxx | uuuu uuuu |
| 7B0h | LCDDATA16 | SEG39 COM1 | SEG38 COM1 | SEG37 COM1 | SEG36 COM1 | SEG35 COM1 | SEG34 COM1 | SEG33 COM1 | SEG32 COM1 | xxxx xxxx | uuuu uuuu |
| 7B1h | LCDDATA17 | — | — | SEG45 COM1 | SEG44 COM1 | SEG43 COM1 | SEG42 COM1 | SEG41 COM1 | SEG40 COM1 | --xx xxxx | --uu uuuu |
| 7B2h | LCDDATA18 | SEG31 COM2 | SEG30 COM2 | SEG29 COM2 | SEG28 COM2 | SEG27 COM2 | SEG26 COM2 | SEG25 COM2 | SEG24 COM2 | xxxx xxxx | uuuu uuuu |
| 7B3h | LCDDATA19 | SEG39 COM2 | SEG38 COM2 | SEG37 COM2 | SEG36 COM2 | SEG35 COM2 | SEG34 COM2 | SEG33 COM2 | SEG32 COM2 | xxxx xxxx | uuuu uuuu |
| 7B4h | LCDDATA20 | — | — | SEG45 COM2 | SEG44 COM2 | SEG43 COM2 | SEG42 COM2 | SEG41 COM2 | SEG40 COM2 | --xx xxxx | --uu uuuu |
| 7B5h | LCDDATA21 | SEG31 COM3 | SEG30 COM3 | SEG29 COM3 | SEG28 COM3 | SEG27 COM3 | SEG26 COM3 | SEG25 COM3 | SEG24 COM3 | xxxx xxxx | uuuu uuuu |
| 7B6h | LCDDATA22 | SEG39 COM3 | SEG38 COM3 | SEG37 COM3 | SEG36 COM3 | SEG35 COM3 | SEG34 COM3 | SEG33 COM3 | SEG32 COM3 | xxxx xxxx | uuuu uuuu |
| 7B7h | LCDDATA23 | — | — | SEG45 COM3 | SEG44 COM3 | SEG43 COM3 | SEG42 COM3 | SEG41 COM3 | SEG40 COM3 | --xx xxxx | --uu uuuu |
| 7B8h — 7EFh | — | Unimplemented | | | | | | | | — | — |

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

**Note 1:** The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<14:8>, whose contents are transferred to the upper byte of the program counter.

**2:** These registers can be addressed from any bank.

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

**TABLE 3-10: 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Banks 16-30** | | | | | | | | | | | |
| x00h/ x80h(2) | INDF0 | Addressing this location uses contents of FSR0H/FSR0L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| x00h/ x81h(2) | INDF1 | Addressing this location uses contents of FSR1H/FSR1L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| x02h/ x82h(2) | PCL | Program Counter (PC) Least Significant Byte | | | | | | | | 0000 0000 | 0000 0000 |
| x03h/ x83h(2) | STATUS | — | — | — | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | ---1 1000 | ---q quuu |
| x04h/ x84h(2) | FSR0L | Indirect Data Memory Address 0 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| x05h/ x85h(2) | FSR0H | Indirect Data Memory Address 0 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| x06h/ x86h(2) | FSR1L | Indirect Data Memory Address 1 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| x07h/ x87h(2) | FSR1H | Indirect Data Memory Address 1 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| x08h/ x88h(2) | BSR | — | — | — | BSR<4:0> | | | | | ---0 0000 | ---0 0000 |
| x09h/ x89h(2) | WREG | Working Register | | | | | | | | 0000 0000 | uuuu uuuu |
| x0Ah/ x8Ah(1),(2) | PCLATH | — | Write Buffer for the upper 7 bits of the Program Counter | | | | | | | -000 0000 | -000 0000 |
| x0Bh/ x8Bh(2) | INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 0000 000x | 0000 000u |
| x0Ch/ x8Ch — x1Fh/ x9Fh | — | Unimplemented | | | | | | | | — | — |

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

**Note 1:** The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<14:8>, whose contents are transferred to the upper byte of the program counter.

**2:** These registers can be addressed from any bank.

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


© 2010-2016 Microchip Technology Inc.

**TABLE 3-10: 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 31** | | | | | | | | | | | |
| F80h(2) | INDF0 | Addressing this location uses contents of FSR0H/FSR0L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| F81h(2) | INDF1 | Addressing this location uses contents of FSR1H/FSR1L to address data memory (not a physical register) | | | | | | | | xxxx xxxx | xxxx xxxx |
| F82h(2) | PCL | Program Counter (PC) Least Significant Byte | | | | | | | | 0000 0000 | 0000 0000 |
| F83h(2) | STATUS | — | — | — | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | ---1 1000 | ---q quuu |
| F84h(2) | FSR0L | Indirect Data Memory Address 0 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| F85h(2) | FSR0H | Indirect Data Memory Address 0 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| F86h(2) | FSR1L | Indirect Data Memory Address 1 Low Pointer | | | | | | | | 0000 0000 | uuuu uuuu |
| F87h(2) | FSR1H | Indirect Data Memory Address 1 High Pointer | | | | | | | | 0000 0000 | 0000 0000 |
| F88h(2) | BSR | — | — | — | BSR<4:0> | | | | | ---0 0000 | ---0 0000 |
| F89h(2) | WREG | Working Register | | | | | | | | 0000 0000 | uuuu uuuu |
| F8Ah(1),(2) | PCLATH | — | Write Buffer for the upper 7 bits of the Program Counter | | | | | | | -000 0000 | -000 0000 |
| F8Bh(2) | INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 0000 000x | 0000 000u |
| F8Ch — FE3h | — | Unimplemented | | | | | | | | — | — |
| FE4h | STATUS_ SHAD | | | | | | Z | DC | C | ---- -xxx | ---- -uuu |
| FE5h | WREG_ SHAD | Working Register Normal (Non-ICD) Shadow | | | | | | | | xxxx xxxx | uuuu uuuu |
| FE6h | BSR_ SHAD | | | | Bank Select Register Normal (Non-ICD) Shadow | | | | | ---x xxxx | ---u uuuu |
| FE7h | PCLATH_ SHAD | | Program Counter Latch High Register Normal (Non-ICD) Shadow | | | | | | | -xxx xxxx | uuuu uuuu |
| FE8h | FSR0L_ SHAD | Indirect Data Memory Address 0 Low Pointer Normal (Non-ICD) Shadow | | | | | | | | xxxx xxxx | uuuu uuuu |
| FE9h | FSR0H_ SHAD | Indirect Data Memory Address 0 High Pointer Normal (Non-ICD) Shadow | | | | | | | | xxxx xxxx | uuuu uuuu |
| FEAh | FSR1L_ SHAD | Indirect Data Memory Address 1 Low Pointer Normal (Non-ICD) Shadow | | | | | | | | xxxx xxxx | uuuu uuuu |
| FEBh | FSR1H_ SHAD | Indirect Data Memory Address 1 High Pointer Normal (Non-ICD) Shadow | | | | | | | | xxxx xxxx | uuuu uuuu |
| FECh | — | Unimplemented | | | | | | | | — | — |
| FEDh | STKPTR | — | — | — | Current Stack Pointer | | | | | ---1 1111 | ---1 1111 |
| FEEh | TOSL | Top of Stack Low byte | | | | | | | | xxxx xxxx | uuuu uuuu |
| FEFh | TOSH | — | Top of Stack High byte | | | | | | | -xxx xxxx | -uuu uuuu |

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

**Note 1:** The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<14:8>, whose contents are transferred to the upper byte of the program counter.

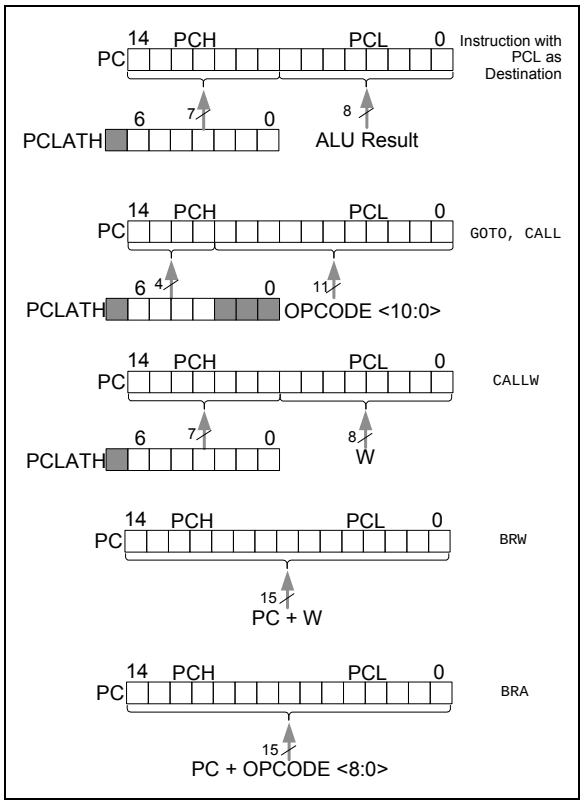
**2:** These registers can be addressed from any bank.

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

## 3.4 PCL and PCLATH

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

**FIGURE 3-4: LOADING OF PC IN DIFFERENT SITUATIONS**




### 3.4.1 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 the 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.


© 2010-2016 Microchip Technology Inc.

## 3.5 Stack

All devices have a 16-level x 15-bit wide hardware stack (refer to Figures 3-4 and 3-5). 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 has five bits to allow detection of overflow and underflow.

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

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

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

**FIGURE 3-5: ACCESSING THE STACK EXAMPLE 1**

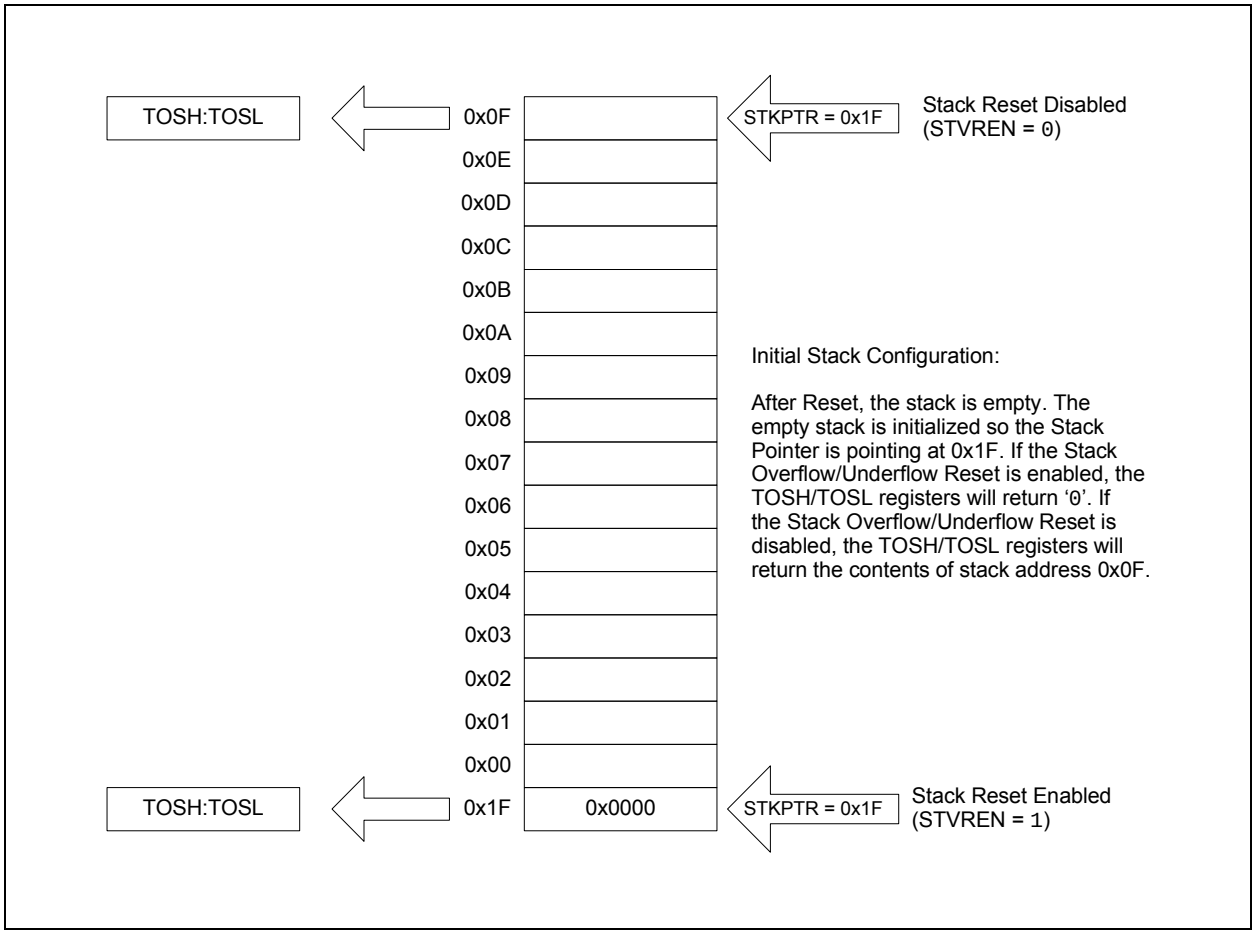


**FIGURE 3-6: ACCESSING THE STACK EXAMPLE 2**

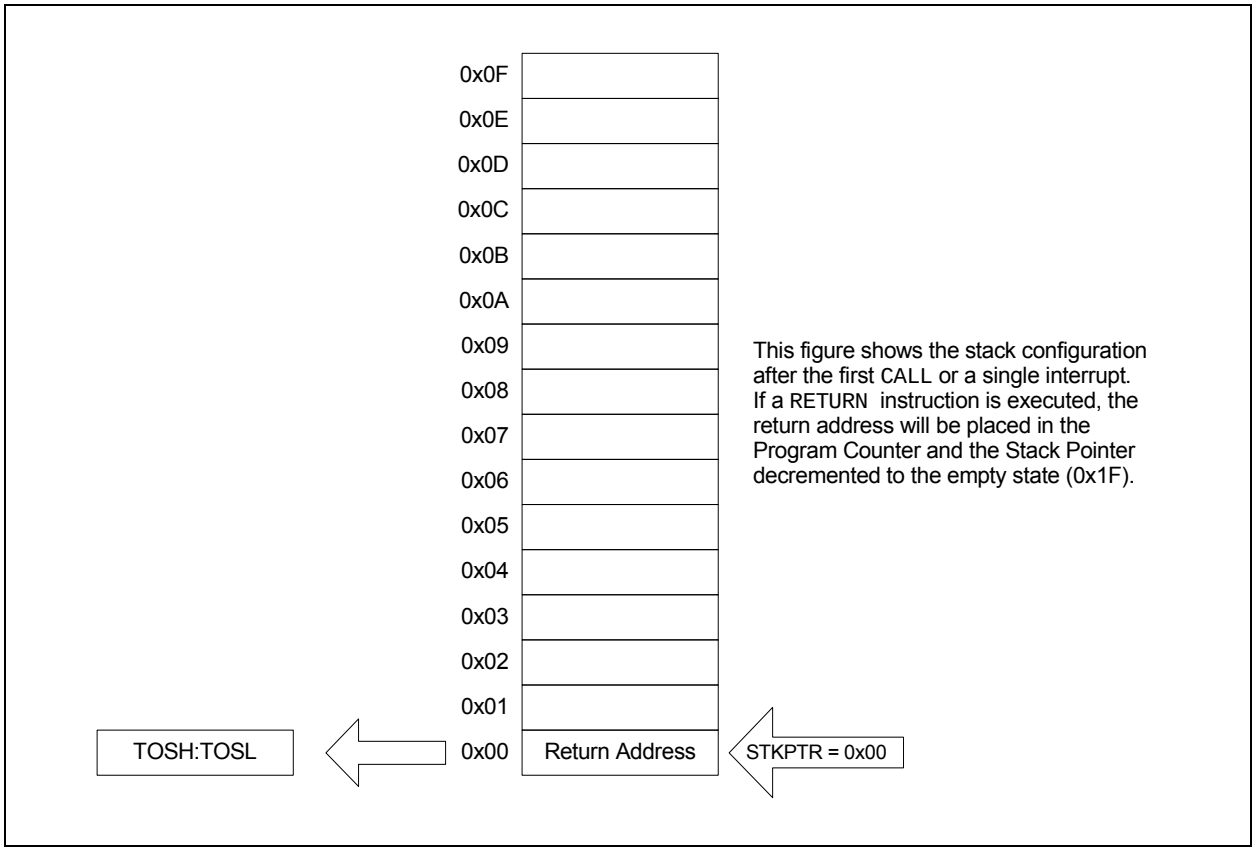
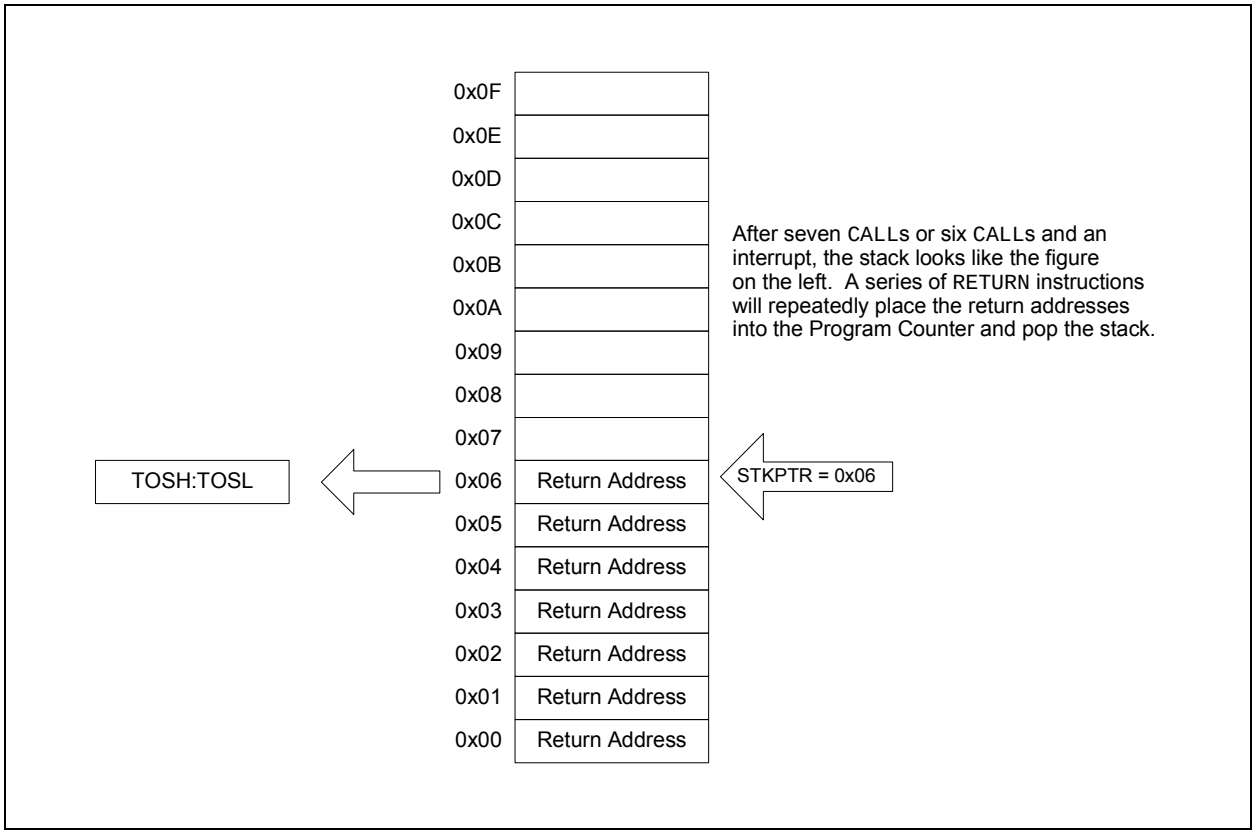


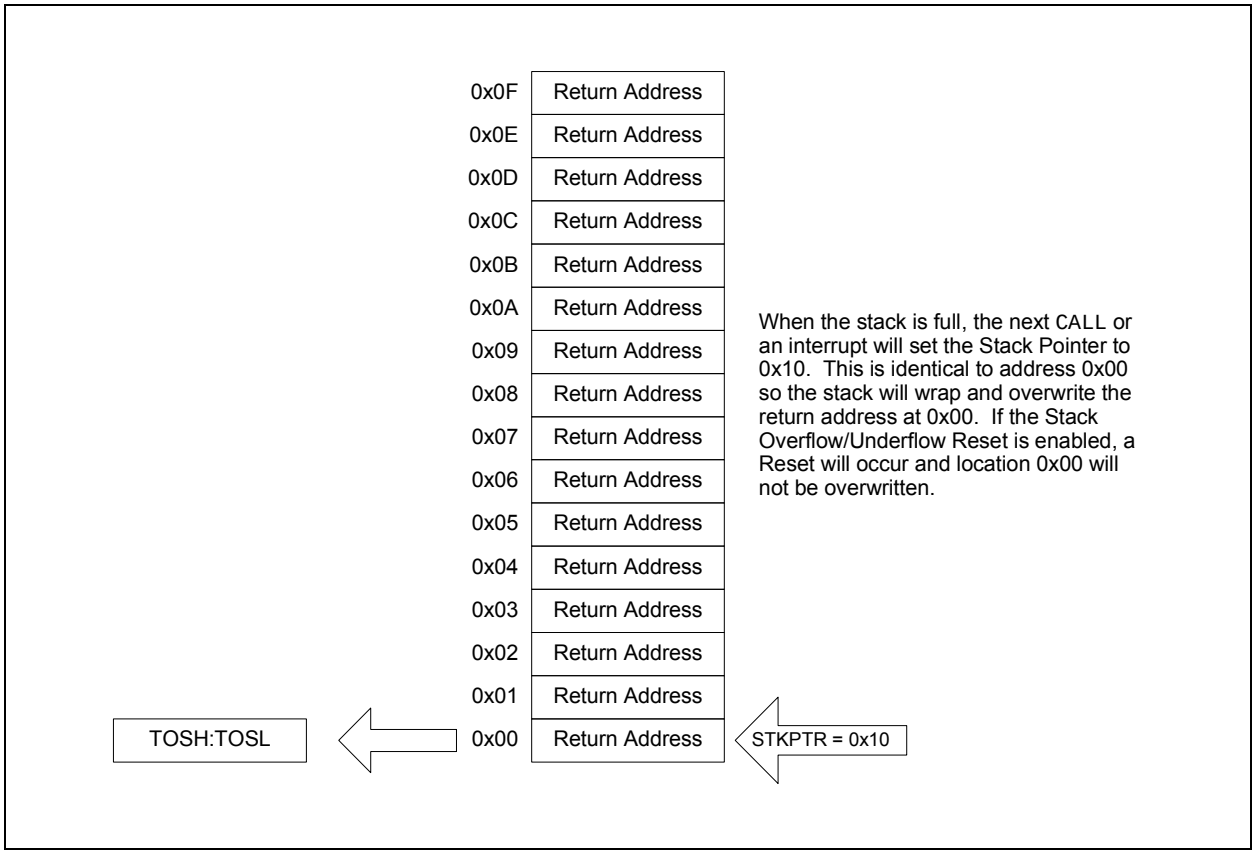

**FIGURE 3-7: ACCESSING THE STACK EXAMPLE 3**





© 2010-2016 Microchip Technology Inc.

**FIGURE 3-8: 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.

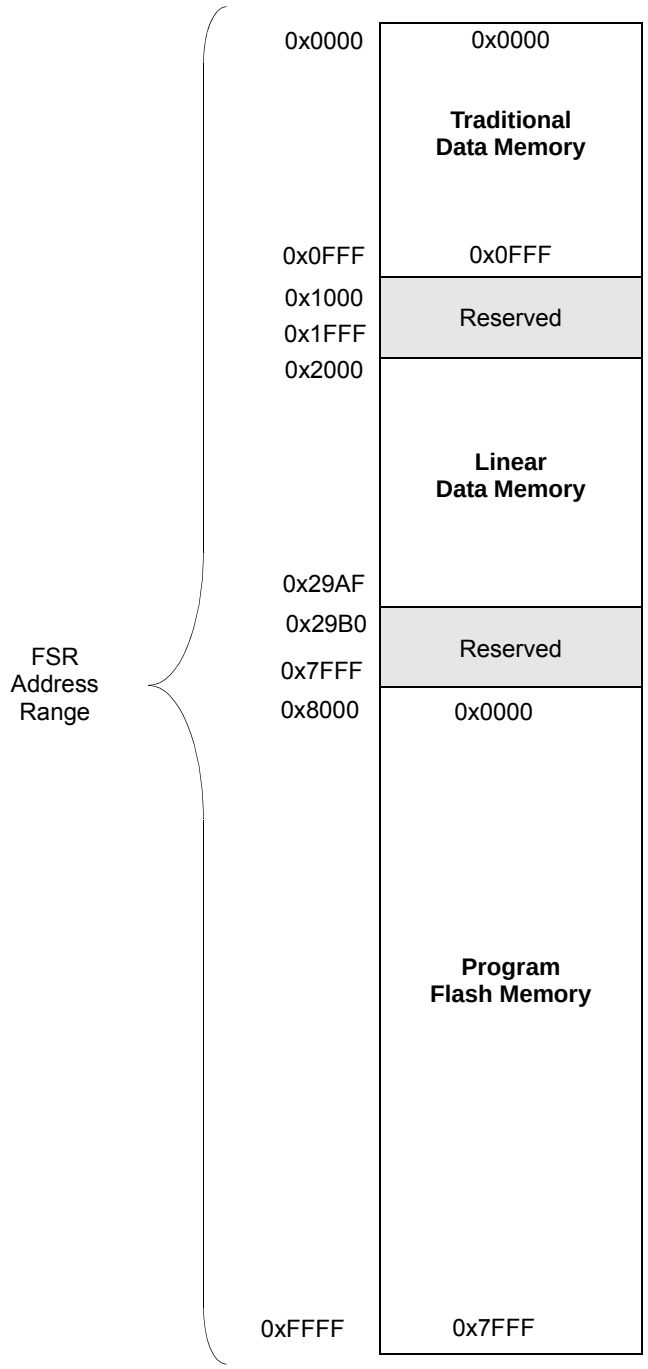
## 3.6 Indirect Addressing

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

**FIGURE 3-9: INDIRECT ADDRESSING**



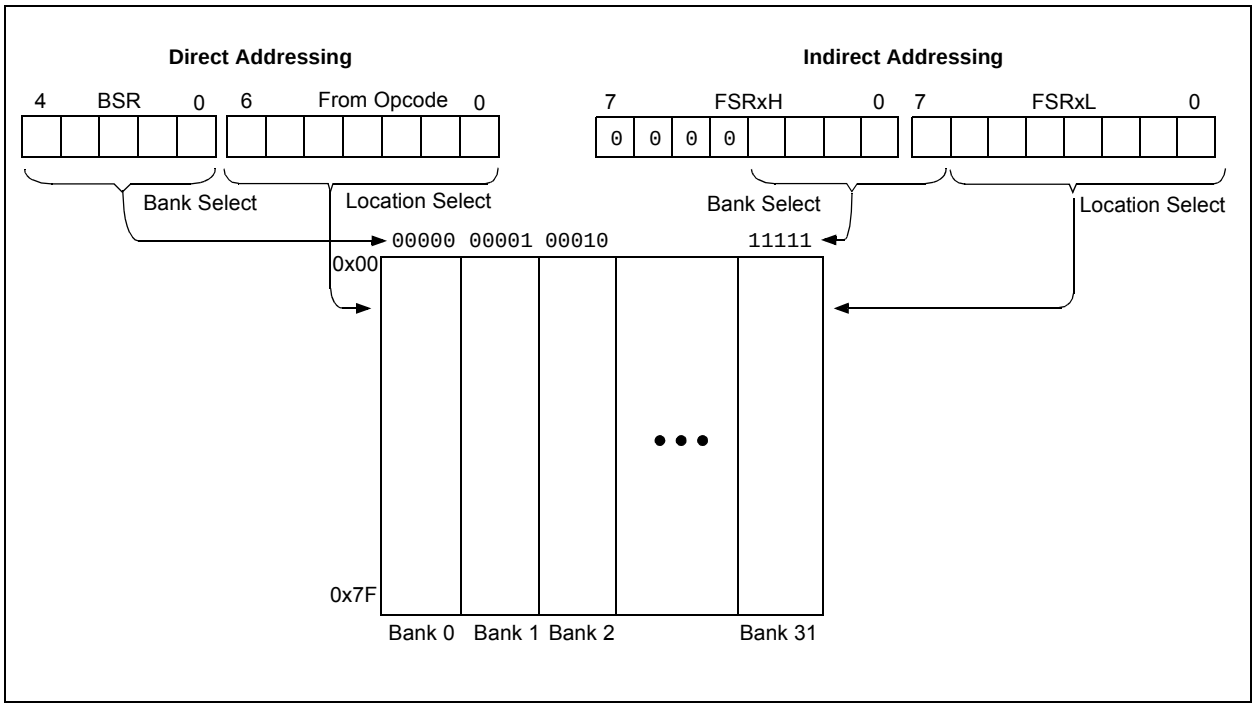

**Note:** Not all memory regions are completely implemented. Consult device memory tables for memory limits.


© 2010-2016 Microchip Technology Inc.

### 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 3-10: TRADITIONAL DATA MEMORY MAP**



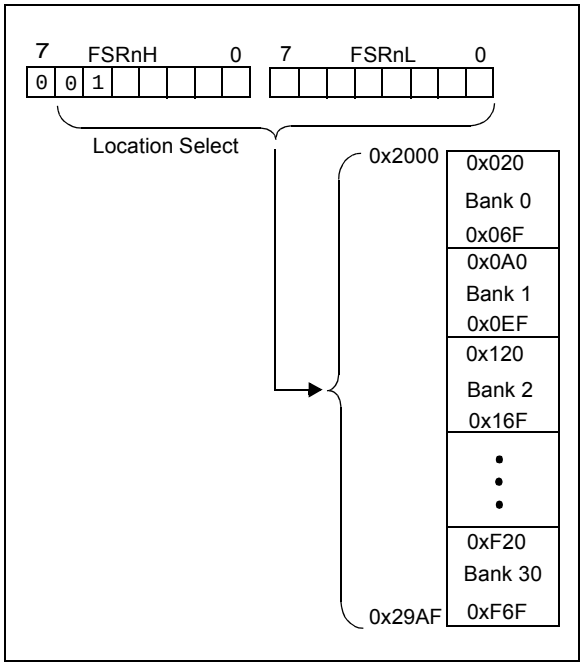
### 3.6.2 LINEAR DATA MEMORY

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.

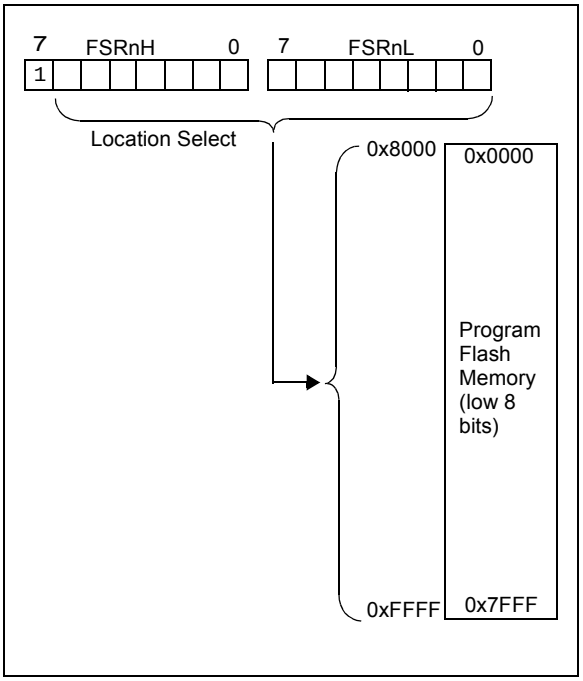
**FIGURE 3-11: LINEAR DATA MEMORY MAP**




### 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 3-12: PROGRAM FLASH MEMORY MAP**





© 2010-2016 Microchip Technology Inc.

## 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 Word 2 is managed automatically by device development tools including debuggers and programmers. For normal device operation, this bit should be maintained as a '1'. |

## 4.2 Register Definitions: Configuration Words

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

| R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 |
|---|---|---|---|---|---|
| FCMEN | IESO | $\overline{\text{CLKOUTEN}}$ | BOREN<1:0> | | $\overline{\text{CPD}}$ |
| bit 13 | | | | | bit 8 |

| R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 |
|---|---|---|---|---|---|---|---|
| $\overline{\text{CP}}$ | MCLRE | $\overline{\text{PWRTE}}$ | WDTE<1:0> | | FOSC<2:0> | | |
| 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 **FCMEN:** Fail-Safe Clock Monitor Enable bit
1 = Fail-Safe Clock Monitor is enabled
0 = Fail-Safe Clock Monitor is disabled

bit 12 **IESO:** Internal External Switchover bit
1 = Internal/External Switchover mode is enabled
0 = Internal/External Switchover mode is disabled

bit 11 **$\overline{\text{CLKOUTEN}}$:** Clock Out Enable bit
If FOSC configuration bits are set to LP, XT, HS modes:
This bit is ignored, CLKOUT function is disabled. Oscillator function on the CLKOUT pin.
All other FOSC modes:
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(1)
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 **$\overline{\text{CPD}}$:** Data Code Protection bit(2)
1 = Data memory code protection is disabled
0 = Data memory code protection is enabled

bit 7 **$\overline{\text{CP}}$:** Code Protection bit(3)
1 = Program memory code protection is disabled
0 = Program memory code protection is enabled

bit 6 **MCLRE:** $\overline{\text{MCLR}}$/$V_{PP}$ Pin Function Select bit
If LVP bit = 1:
This bit is ignored.
If LVP bit = 0:
1 = $\overline{\text{MCLR}}$/$V_{PP}$ pin function is $\overline{\text{MCLR}}$; Weak pull-up enabled.
0 = $\overline{\text{MCLR}}$/$V_{PP}$ pin function is digital input; $\overline{\text{MCLR}}$ internally disabled; Weak pull-up under control of WPUG5 bit.

**Note 1:** Enabling Brown-out Reset does not automatically enable Power-up Timer.
**2:** The entire data EEPROM will be erased when the code protection is turned off during an erase.
**3:** The entire program memory will be erased when the code protection is turned off.


© 2010-2016 Microchip Technology Inc.

**REGISTER 4-1: CONFIG1: CONFIGURATION WORD 1 (CONTINUED)**

bit 5 **$\overline{\text{PWRTE}}$:** Power-up Timer Enable bit(1)
1 = PWRT disabled
0 = PWRT enabled

bit 4-3 **WDTE<1:0>:** Watchdog Timer Enable bit
11 = WDT enabled
10 = WDT enabled while running and disabled in Sleep
01 = WDT controlled by the SWDTEN bit in the WDTCON register
00 = WDT disabled

bit 2-0 **FOSC<2:0>:** Oscillator Selection bits
111 = ECH: External Clock, High-Power mode (4-20 MHz): device clock supplied to CLKIN pin
110 = ECM: External Clock, Medium-Power mode (0.5-4 MHz): device clock supplied to CLKIN pin
101 = ECL: External Clock, Low-Power mode (0-0.5 MHz): device clock supplied to CLKIN pin
100 = INTOSC oscillator: I/O function on CLKIN pin
011 = EXTRC oscillator: External RC circuit connected to CLKIN pin
010 = HS oscillator: High-speed crystal/resonator connected between OSC1 and OSC2 pins
001 = XT oscillator: Crystal/resonator connected between OSC1 and OSC2 pins
000 = LP oscillator: Low-power crystal connected between OSC1 and OSC2 pins

**Note 1:** Enabling Brown-out Reset does not automatically enable Power-up Timer.
**2:** The entire data EEPROM will be erased when the code protection is turned off during an erase.
**3:** The entire program memory will be erased when the code protection is turned off.

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

| R/P-1 | R/P-1 | U-1 | R/P-1 | R/P-1 | R/P-1 |
|---|---|---|---|---|---|
| LVP(1) | $\overline{\text{DEBUG}}$(2) | — | BORV(3) | STVREN | PLLEN |
| bit 13 | | | | | bit 8 |

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

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

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

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

bit 11 **Unimplemented:** Read as '1'

bit 10 **BORV:** Brown-out Reset Voltage Selection bit(3)
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 **PLLEN:** PLL Enable bit
1 = 4xPLL enabled
0 = 4xPLL disabled

bit 7-5 **Unimplemented:** Read as '1'

bit 4 **VCAPEN:** Voltage Regulator Capacitor Enable bits
0 = VCAP functionality is enabled on RF0
1 = No capacitor on VCAP pin

bit 3-2 **Unimplemented:** Read as '1'

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


© 2010-2016 Microchip Technology Inc.

**REGISTER 4-2: CONFIG2: CONFIGURATION WORD 2 (CONTINUED)**

bit 1-0 **WRT<1:0>:** Flash Memory Self-Write Protection bits

8 kW Flash memory (PIC16(L)F1946):

- 11 = Write protection off
- 10 = 000h to 1FFh write-protected, 200h to 1FFFh may be modified by EECON control
- 01 = 000h to FFFh write-protected, 1000h to 1FFFh may be modified by EECON control
- 00 = 000h to 1FFFh write-protected, no addresses may be modified by EECON control

16 kW Flash memory (PIC16(L)F1947):

- 11 = Write protection off
- 10 = 000h to 1FFh write-protected, 200h to 3FFFh may be modified by EECON control
- 01 = 000h to 1FFFh write-protected, 2000h to 3FFFh may be modified by EECON control
- 00 = 000h to 3FFFh write-protected, no addresses may be modified by EECON control

**Note 1:** The LVP bit cannot be programmed to '0' when Programming mode is entered via LVP.

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

**3:** See Vbor parameter for specific trip point voltages.

## 4.3 Code Protection

Code protection allows the device to be protected from unauthorized access. Program memory protection and data EEPROM protection are controlled independently. Internal access to the program memory and data EEPROM are 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.

### 4.3.2 DATA EEPROM PROTECTION

The entire data EEPROM is protected from external reads and writes by the $\overline{CPD}$ bit. When $\overline{CPD}$ = 0, external reads and writes of data EEPROM are inhibited. The CPU can continue to read and write data EEPROM regardless of the protection bit settings.

## 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 4.6 "Device ID and Revision ID"** for more information on accessing these memory locations. For more information on checksum calculation, see the "*PIC16F193X/LF193X/PIC16F194X/LF190X Memory Programming Specification*" (DS41397).


© 2010-2016 Microchip Technology Inc.

## 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 11.5 "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: DEVICEID: DEVICE ID REGISTER**

| R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DEV<8:3> | | | | | | | | DEV<2:0> | | | REV<4:0> | | | | |
| bit 13 | | | | | | | bit 8 | bit 7 | | | | | | | bit 0 |

| **Legend:** | | |
|---|---|---|
| R = Readable bit | | |
| '1' = Bit is set | '0' = Bit is cleared | -n = Value when blank or after Bulk Erase |

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

| Device | DEVICEID<13:0> Values | |
|---|---|---|
| | DEV<8:0> | REV<4:0> |
| PIC16F1946 | 10 0011 001 | x xxxx |
| PIC16F1947 | 10 0011 010 | x xxxx |
| PIC16LF1946 | 10 0011 011 | x xxxx |
| PIC16LF1947 | 10 0011 100 | x xxxx |

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

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

# 5.0 OSCILLATOR MODULE (WITH FAIL-SAFE CLOCK MONITOR)

## 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 external oscillators, quartz crystal resonators, ceramic resonators and Resistor-Capacitor (RC) circuits. In addition, the system clock source can be supplied from one of two internal oscillators and PLL circuits, with a choice of speeds selectable via software. Additional clock features include:

- Selectable system clock source between external or internal sources via software.
- Two-Speed Start-up mode, which minimizes latency between external oscillator start-up and code execution.
- Fail-Safe Clock Monitor (FSCM) designed to detect a failure of the external clock source (LP, XT, HS, EC or RC modes) and switch automatically to the internal oscillator.
- Oscillator Start-up Timer (OST) ensures stability of crystal oscillator sources

The oscillator module can be configured in one of eight clock modes.

1. ECL – External Clock Low Power mode (0 MHz to 0.5 MHz)
2. ECM – External Clock Medium Power mode (0.5 MHz to 4 MHz)
3. ECH – External Clock High Power mode (4 MHz to 32 MHz)
4. LP – 32 kHz Low-Power Crystal mode.
5. XT – Medium Gain Crystal or Ceramic Resonator Oscillator mode (up to 4 MHz)
6. HS – High Gain Crystal or Ceramic Resonator mode (4 MHz to 20 MHz)
7. RC – External Resistor-Capacitor (RC).
8. INTOSC – Internal oscillator (31 kHz to 32 MHz).

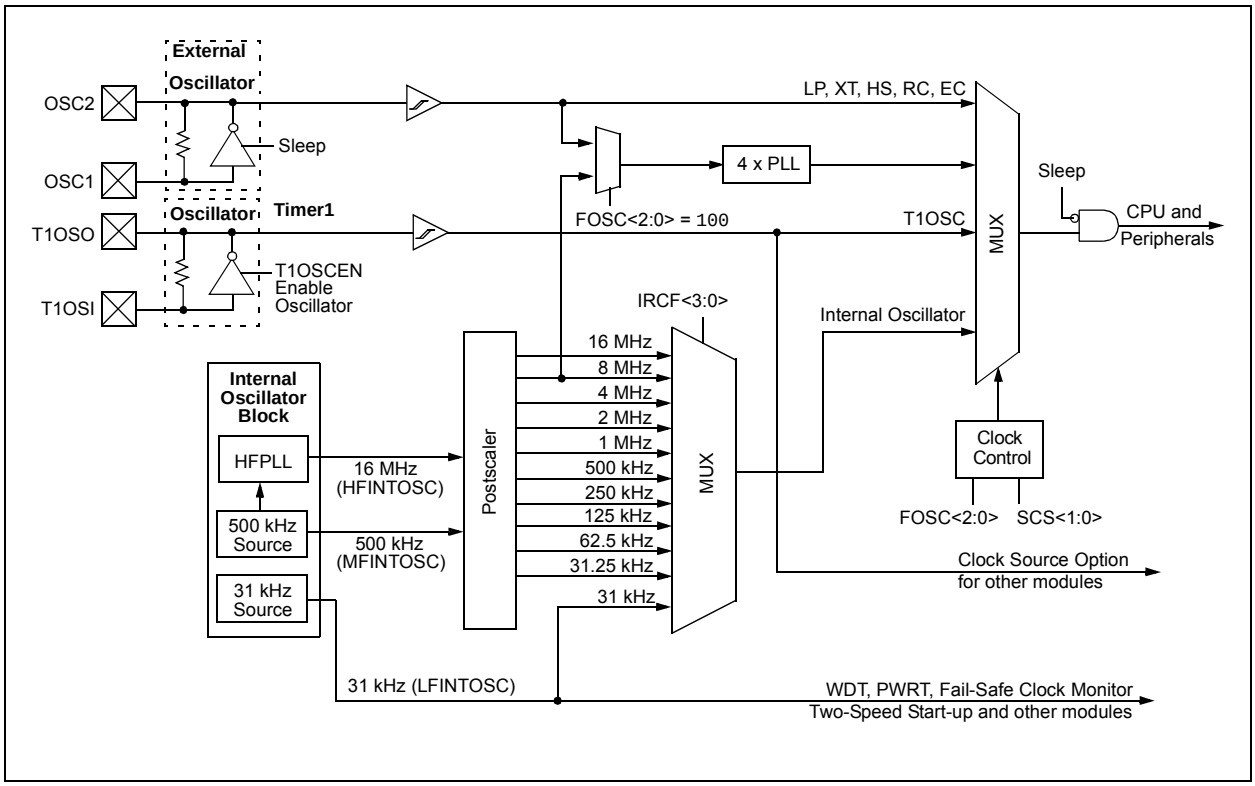
Clock Source modes are selected by the FOSC<2:0> bits in the Configuration Words. The FOSC bits determine the type of oscillator that will be used when the device is first powered.

The EC clock mode relies on an external logic level signal as the device clock source. The LP, XT, and HS clock modes require an external crystal or resonator to be connected to the device. Each mode is optimized for a different frequency range. The RC clock mode requires an external resistor and capacitor to set the oscillator frequency.

The INTOSC internal oscillator block produces low, medium, and high frequency clock sources, designated LFINTOSC, MFINTOSC, and HFINTOSC. (see Internal Oscillator Block, Figure 5-1). A wide selection of device clock frequencies may be derived from these three clock sources.


© 2010-2016 Microchip Technology Inc.

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



## 5.2 Clock Source Types

Clock sources can be classified as external or internal.

External clock sources rely on external circuitry for the clock source to function. Examples are: oscillator modules (EC mode), quartz crystal resonators or ceramic resonators (LP, XT and HS modes) and Resistor-Capacitor (RC) mode circuits.

Internal clock sources are contained within the oscillator module. The internal oscillator block has two internal oscillators and a dedicated Phase-Lock Loop (HFPLL) that are used to generate three internal system clock sources: the 16 MHz High-Frequency Internal Oscillator (HFINTOSC), 500 kHz (MFINTOSC) and the 31 kHz Low-Frequency Internal Oscillator (LFINTOSC).

The system clock can be selected between external or internal clock sources via the System Clock Select (SCS) bits in the OSCCON register. See **Section 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<2:0> 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:
  - Timer1 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

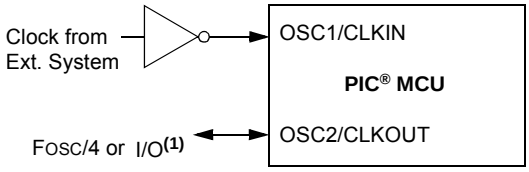
The External Clock (EC) mode allows an externally generated logic level signal to be the system clock source. When operating in this mode, an external clock source is connected to the OSC1 input. OSC2/CLKOUT is available for general purpose I/O or CLKOUT. Figure 5-2 shows the pin connections for EC mode.

EC mode has 3 power modes to select from through Configuration Words:

- High power, 4-32 MHz (FOSC = 111)
- Medium power, 0.5-4 MHz (FOSC = 110)
- Low power, 0-0.5 MHz (FOSC = 101)

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

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




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

#### 5.2.1.2 LP, XT, HS Modes

The LP, XT and HS modes support the use of quartz crystal resonators or ceramic resonators connected to OSC1 and OSC2 (Figure 5-3). The three modes select a low, medium or high gain setting of the internal inverter-amplifier to support various resonator types and speed.

**LP** Oscillator mode selects the lowest gain setting of the internal inverter-amplifier. LP mode current consumption is the least of the three modes. This mode is designed to drive only 32.768 kHz tuning-fork type crystals (watch crystals).

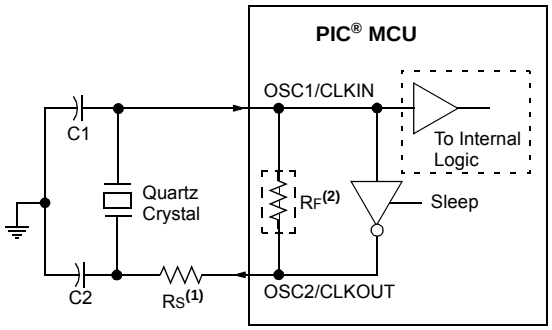
**XT** Oscillator mode selects the intermediate gain setting of the internal inverter-amplifier. XT mode current consumption is the medium of the three modes. This mode is best suited to drive resonators with a medium drive level specification.

**HS** Oscillator mode selects the highest gain setting of the internal inverter-amplifier. HS mode current consumption is the highest of the three modes. This mode is best suited for resonators that require a high drive setting.

Figure 5-3 and Figure 5-4 show typical circuits for quartz crystal and ceramic resonators, respectively.


© 2010-2016 Microchip Technology Inc.

**FIGURE 5-3: QUARTZ CRYSTAL OPERATION (LP, XT OR HS MODE)**




**Note 1:** A series resistor (RS) may be required for quartz crystals with low drive level.

**2:** The value of RF varies with the Oscillator mode selected (typically between 2 MΩ to 10 MΩ).

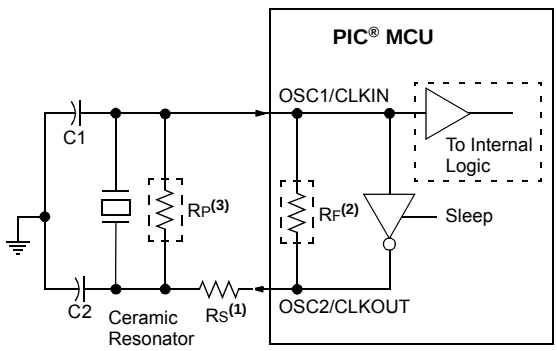
**Note 1:** Quartz crystal characteristics vary according to type, package and manufacturer. The user should consult the manufacturer data sheets for specifications and recommended application.

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

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

- *AN826*, *Crystal Oscillator Basics and Crystal Selection for rfPIC® and PIC® Devices* (DS00826)
- *AN849*, *Basic PICmicro® Oscillator Design* (DS00849)
- *AN943*, *Practical PICmicro® Oscillator Analysis and Design* (DS00943)
- *AN949*, *Making Your Oscillator Work* (DS00949)

**FIGURE 5-4: CERAMIC RESONATOR OPERATION (XT OR HS MODE)**




**Note 1:** A series resistor (RS) may be required for ceramic resonators with low drive level.

**2:** The value of RF varies with the Oscillator mode selected (typically between 2 MΩ to 10 MΩ).

**3:** An additional parallel feedback resistor (RP) may be required for proper ceramic resonator operation.

### 5.2.1.3 Oscillator Start-up Timer (OST)

If the oscillator module is configured for LP, XT or HS modes, the Oscillator Start-up Timer (OST) counts 1024 oscillations from OSC1. This occurs following a Power-on Reset (POR) and when the Power-up Timer (PWRT) has expired (if configured), or a wake-up from Sleep. During this time, the program counter does not increment and program execution is suspended. The OST ensures that the oscillator circuit, using a quartz crystal resonator or ceramic resonator, has started and is providing a stable system clock to the oscillator module.

In order to minimize latency between external oscillator start-up and code execution, the Two-Speed Clock Start-up mode can be selected (see **Section 5.4 "Two-Speed Clock Start-up Mode"**).

### 5.2.1.4 4x PLL

The oscillator module contains a 4x PLL that can be used with both external and internal clock sources to provide a system clock source. The input frequency for the 4x PLL must fall within specifications. See the PLL Clock Timing Specifications in **Section 30.0 "Electrical Specifications"**.

The 4x PLL may be enabled for use by one of two methods:

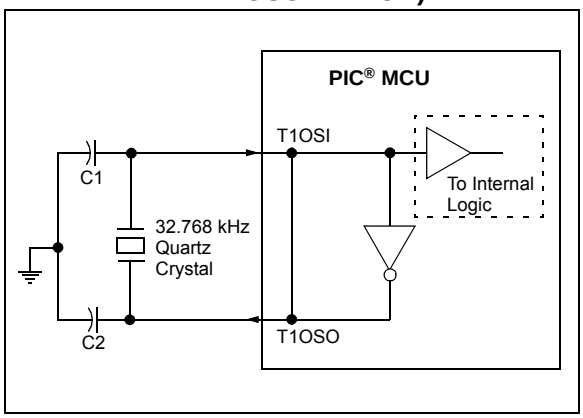
1. Program the PLLEN bit in Configuration Words to a '1'.
2. Write the SPLLEN bit in the OSCCON register to a '1'. If the PLLEN bit in Configuration Words is programmed to a '1', then the value of SPLLEN is ignored.

### 5.2.1.5 TIMER1 Oscillator

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

The Timer1 Oscillator can be used as an alternate system clock source and can be selected during run-time using clock switching. Refer to **Section 5.3 "Clock Switching"** for more information.

**FIGURE 5-5: QUARTZ CRYSTAL OPERATION (TIMER1 OSCILLATOR)**




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

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

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

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

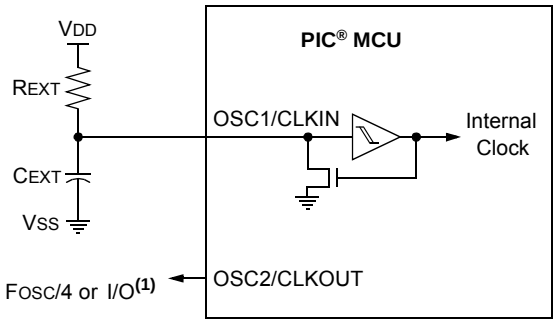
### 5.2.1.6 External RC Mode

The external Resistor-Capacitor (RC) modes support the use of an external RC circuit. This allows the designer maximum flexibility in frequency choice while keeping costs to a minimum when clock accuracy is not required.

The RC circuit connects to OSC1. OSC2/CLKOUT is available for general purpose I/O or CLKOUT. The function of the OSC2/CLKOUT pin is determined by the $\overline{\text{CLKOUTEN}}$ bit in Configuration Words.

Figure 5-6 shows the external RC mode connections.

**FIGURE 5-6: EXTERNAL RC MODES**




Recommended values: 10 kΩ ≤ REXT ≤ 100 kΩ, <3V
3 kΩ ≤ REXT ≤ 100 kΩ, 3-5V
CEXT > 20 pF, 2-5V

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

The RC oscillator frequency is a function of the supply voltage, the resistor (REXT) and capacitor (CEXT) values and the operating temperature. Other factors affecting the oscillator frequency are:

- threshold voltage variation
- component tolerances
- packaging variations in capacitance

The user also needs to take into account variation due to tolerance of external RC components used.


© 2010-2016 Microchip Technology Inc.

### 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<2:0> 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, OSC1/CLKIN is available for general purpose I/O. OSC2/CLKOUT is available for general purpose I/O or CLKOUT.

The function of the OSC2/CLKOUT pin is determined by the $\overline{\text{CLKOUTEN}}$ bit in Configuration Words.

The internal oscillator block has two independent oscillators and a dedicated Phase-Lock Loop, HFPLL that can produce one of three internal system clock sources.

1. The **HFINTOSC** (High-Frequency Internal Oscillator) is factory calibrated and operates at 16 MHz. The HFINTOSC source is generated from the 500 kHz MFINTOSC source and the dedicated Phase-Lock Loop, HFPLL. The frequency of the HFINTOSC can be user-adjusted via software using the OSCTUNE register (Register 5-3).
2. The **MFINTOSC** (Medium-Frequency Internal Oscillator) is factory calibrated and operates at 500 kHz. The frequency of the MFINTOSC can be user-adjusted via software using the OSCTUNE register (Register 5-3).
3. The **LFINTOSC** (Low-Frequency Internal Oscillator) is uncalibrated and 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 frequency of the HFINTOSC can be altered via software using the OSCTUNE register (Register 5-3).

The output of the HFINTOSC connects to a postscaler and multiplexer (see Figure 5-1). One of multiple frequencies derived from the HFINTOSC can be selected via software using the IRCF<3:0> bits of the OSCCON register. See **Section 5.2.2.7 "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<2:0> = `100`, 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 Status Locked bit (HFIOFL) of the OSCSTAT register indicates when the HFINTOSC is running within 2% of its final value.

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 MFINTOSC

The Medium-Frequency Internal Oscillator (MFINTOSC) is a factory calibrated 500 kHz internal clock source. The frequency of the MFINTOSC can be altered via software using the OSCTUNE register (Register 5-3).

The output of the MFINTOSC connects to a postscaler and multiplexer (see Figure 5-1). One of nine frequencies derived from the MFINTOSC can be selected via software using the IRCF<3:0> bits of the OSCCON register. See **Section 5.2.2.7 "Internal Oscillator Clock Switch Timing"** for more information.

The MFINTOSC is enabled by:

- Configure the IRCF<3:0> bits of the OSCCON register for the desired HF frequency, and
- FOSC<2:0> = `100`, or
- Set the System Clock Source (SCS) bits of the OSCCON register to '`1x`'

The Medium-Frequency Internal Oscillator Ready bit (MFIOFR) of the OSCSTAT register indicates when the MFINTOSC is running.

#### 5.2.2.3 Internal Oscillator Frequency Adjustment

The 500 kHz internal oscillator is factory calibrated. This internal oscillator can be adjusted in software by writing to the OSCTUNE register (Register 5-3). Since the HFINTOSC and MFINTOSC clock sources are derived from the 500 kHz internal oscillator a change in the OSCTUNE register value will apply to both.

The default value of the OSCTUNE register is '0'. The value is a 6-bit two's complement number. A value of 1Fh will provide an adjustment to the maximum frequency. A value of 20h will provide an adjustment to the minimum frequency.

When the OSCTUNE register is modified, the oscillator frequency will begin shifting to the new frequency. Code execution continues during this shift. There is no indication that the shift has occurred.

OSCTUNE does not affect the LFINTOSC frequency. Operation of features that depend on the LFINTOSC clock source frequency, such as the Power-up Timer (PWRT), Watchdog Timer (WDT), Fail-Safe Clock Monitor (FSCM) and peripherals, are *not* affected by the change in frequency.

#### 5.2.2.4 LFINTOSC

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

The output of the LFINTOSC connects to a multiplexer (see Figure 5-1). Select 31 kHz, via software, using the IRCF<3:0> bits of the OSCCON register. See **Section 5.2.2.7 "Internal Oscillator Clock Switch Timing"** for more information. The LFINTOSC is also the frequency for the Power-up Timer (PWRT), Watchdog Timer (WDT) and Fail-Safe Clock Monitor (FSCM).

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<2:0> = 100, 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)
- Fail-Safe Clock Monitor (FSCM)

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

#### 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 output of the 16 MHz HFINTOSC and 31 kHz LFINTOSC connects to a postscaler and multiplexer (see Figure 5-1). The Internal Oscillator Frequency Select bits IRCF<3:0> of the OSCCON register select the frequency output of the internal oscillators. One of the following frequencies can be selected via software:

- HFINTOSC
  - 32 MHz (requires 4x PLL)
  - 16 MHz
  - 8 MHz
  - 4 MHz
  - 2 MHz
  - 1 MHz
  - 500 kHz (default after Reset)
  - 250 kHz
  - 125 kHz
  - 62.5 kHz
  - 31.25 kHz
- LFINTOSC
  - 31 kHz

| **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.


© 2010-2016 Microchip Technology Inc.

#### 5.2.2.6 32 MHz Internal Oscillator Frequency Selection

The Internal Oscillator Block can be used with the 4x PLL associated with the External Oscillator Block to produce a 32 MHz internal system clock source. The following settings are required to use the 32 MHz internal clock source:

- The FOSC bits in Configuration Words must be set to use the INTOSC source as the device system clock (FOSC<2:0> = 100).
- The SCS bits in the OSCCON register must be cleared to use the clock determined by FOSC<2:0> in Configuration Words (SCS<1:0> = 00).
- The IRCF bits in the OSCCON register must be set to the 8 MHz HFINTOSC set to use (IRCF<3:0> = 1110).
- The SPLLEN bit in the OSCCON register must be set to enable the 4xPLL, or the PLLEN bit of the Configuration Words must be programmed to a '1'.

| | |
|---|---|
| **Note:** | When using the PLLEN bit of the Configuration Words, the 4xPLL cannot be disabled by software and the 8 MHz HFINTOSC option will no longer be available. |

The 4xPLL is not available for use with the internal oscillator when the SCS bits of the OSCCON register are set to '1x'. The SCS bits must be set to '00' to use the 4xPLL with the internal oscillator.

#### 5.2.2.7 Internal Oscillator Clock Switch Timing

When switching between the HFINTOSC, MFINTOSC and the LFINTOSC, the new oscillator may already be shut down to save power (see Figure 5-7). 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, MFINTOSC and LFINTOSC oscillators. The sequence of a frequency selection is as follows:

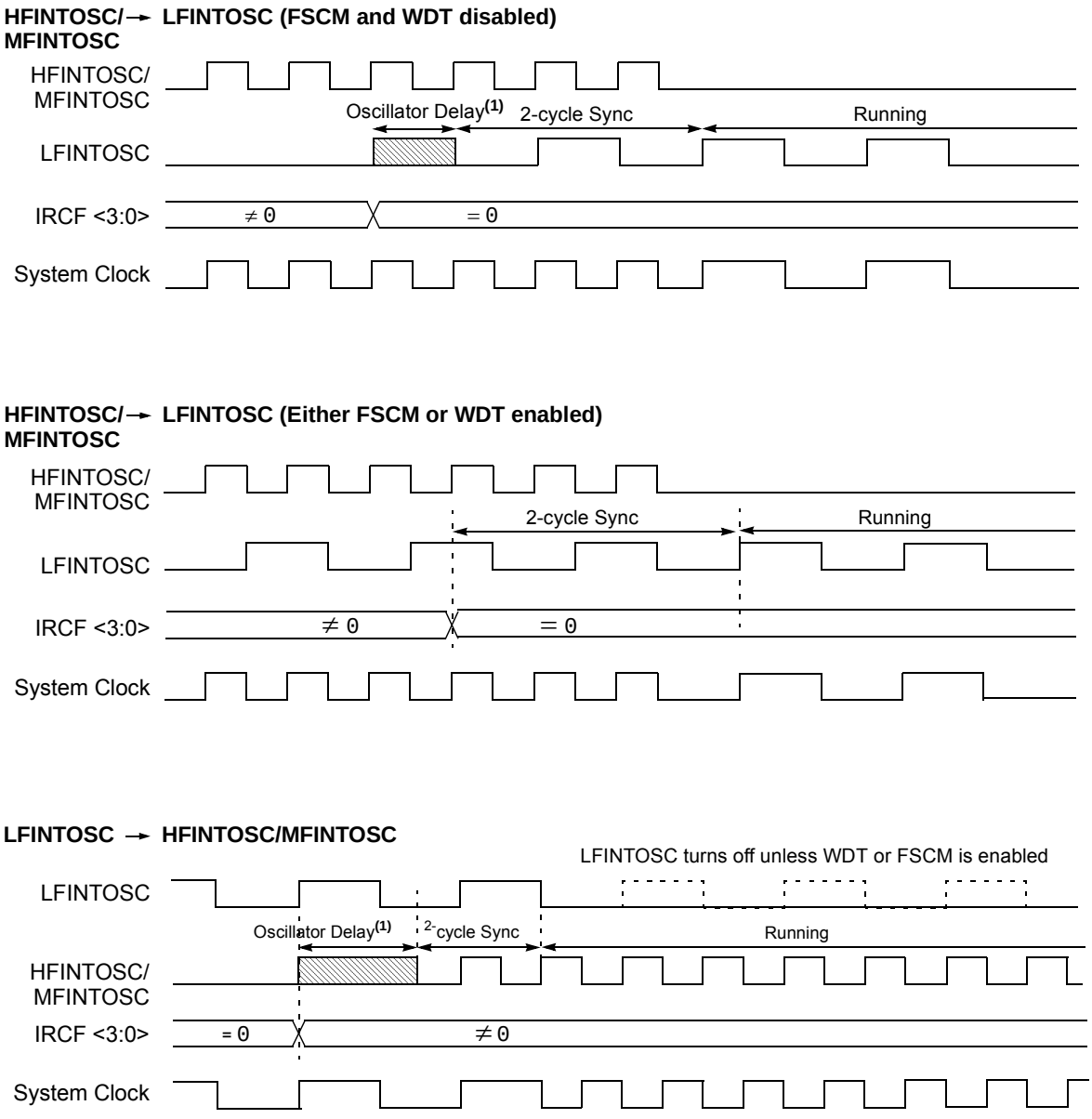
1. IRCF<3:0> bits of the OSCCON register are modified.
2. If the new clock is shut down, a clock start-up delay is started.
3. Clock switch circuitry waits for a falling edge of the current clock.
4. The current clock is held low and the clock switch circuitry waits for a rising edge in the new clock.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

See Figure 5-7 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 .

Start-up delay specifications are located in the oscillator tables of **Section 30.0 "Electrical Specifications"**

**FIGURE 5-7: INTERNAL OSCILLATOR SWITCH TIMING**




**Note:** See Table 5-1, Oscillator Switching Delays, for more information.


© 2010-2016 Microchip Technology Inc.

## 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
- Timer1 32 kHz crystal oscillator
- 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<2:0> bits in the Configuration Words.
- When the SCS bits of the OSCCON register = 01, the system clock source is the Timer1 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.

| | |
|---|---|
| **Note:** | Any automatic clock switch, which may occur from Two-Speed Start-up or Fail-Safe Clock Monitor, does not update the SCS bits of the OSCCON register. The user can monitor the OSTS bit of the OSCSTAT register to determine the current system clock source. |

When switching between clock sources, a delay is required to allow the new clock to stabilize. These oscillator delays are shown in Table 5-1.

### 5.3.2 OSCILLATOR START-UP TIME-OUT STATUS (OSTS) BIT

The Oscillator Start-up Time-out Status (OSTS) bit of the OSCSTAT register indicates whether the system clock is running from the external clock source, as defined by the FOSC<2:0> bits in the Configuration Words, or from the internal clock source. In particular, OSTS indicates that the Oscillator Start-up Timer (OST) has timed out for LP, XT or HS modes. The OST does not reflect the status of the Timer1 oscillator.

### 5.3.3 TIMER1 OSCILLATOR

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

The Timer1 oscillator is enabled using the T1OSCEN control bit in the T1CON register. See **Section 21.0 "Timer1 Module with Gate Control"** for more information about the Timer1 peripheral.

### 5.3.4 TIMER1 OSCILLATOR READY (T1OSCR) BIT

The user must ensure that the Timer1 oscillator is ready to be used before it is selected as a system clock source. The Timer1 Oscillator Ready (T1OSCR) bit of the OSCSTAT register indicates whether the Timer1 oscillator is ready to be used. After the T1OSCR bit is set, the SCS bits can be configured to select the Timer1 oscillator.

### 5.3.5 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 PLLR is cleared the switch from 32 MHz operation to the selected internal clock is complete.

## 5.4 Two-Speed Clock Start-up Mode

Two-Speed Start-up mode provides additional power savings by minimizing the latency between external oscillator start-up and code execution. In applications that make heavy use of the Sleep mode, Two-Speed Start-up will remove the external oscillator start-up time from the time spent awake and can reduce the overall power consumption of the device. This mode allows the application to wake-up from Sleep, perform a few instructions using the INTOSC internal oscillator block as the clock source and go back to Sleep without waiting for the external oscillator to become stable.

Two-Speed Start-up provides benefits when the oscillator module is configured for LP, XT, or HS modes. The Oscillator Start-up Timer (OST) is enabled for these modes and must count 1024 oscillations before the oscillator can be used as the system clock source.

If the oscillator module is configured for any mode other than LP, XT or HS mode, then Two-Speed Start-up is disabled. This is because the external clock oscillator does not require any stabilization time after POR or an exit from Sleep.

If the OST count reaches 1024 before the device enters Sleep mode, the OSTS bit of the OSCSTAT register is set and program execution switches to the external oscillator. However, the system may never operate from the external oscillator if the time spent awake is very short.

| **Note:** | Executing a SLEEP instruction will abort the oscillator start-up time and will cause the OSTS bit of the OSCSTAT register to remain clear. |
|---|---|

### 5.4.1 TWO-SPEED START-UP MODE CONFIGURATION

Two-Speed Start-up mode is configured by the following settings:

- IESO (of the Configuration Words) = 1; Internal/External Switchover bit (Two-Speed Start-up mode enabled).
- SCS (of the OSCCON register) = 00.
- FOSC<2:0> bits in the Configuration Words configured for LP, XT or HS mode.

Two-Speed Start-up mode is entered after:

- Power-on Reset (POR) and, if enabled, after Power-up Timer (PWRT) has expired, or
- Wake-up from Sleep.

**TABLE 5-1: OSCILLATOR SWITCHING DELAYS**

| Switch From | Switch To | Frequency | Oscillator Delay |
|---|---|---|---|
| Sleep | LFINTOSC(1)<br>MFINTOSC(1)<br>HFINTOSC(1) | 31 kHz<br>31.25 kHz-500 kHz<br>31.25 kHz-16 MHz | 2 cycles |
| Sleep/POR | EC, RC(1) | DC – 32 MHz | 2 cycles |
| LFINTOSC | EC, RC(1) | DC – 32 MHz | 1 cycle of each |
| Sleep/POR | Timer1 Oscillator<br>LP, XT, HS(1) | 32 kHz-20 MHz | 1024 Clock Cycles (OST) |
| Any clock source | MFINTOSC(1)<br>HFINTOSC(1) | 31.25 kHz-500 kHz<br>31.25 kHz-16 MHz | 2 μs (approx.) |
| Any clock source | LFINTOSC(1) | 31 kHz | 1 cycle of each |
| Any clock source | Timer1 Oscillator | 32 kHz | 1024 Clock Cycles (OST) |
| PLL inactive | PLL active | 16-32 MHz | 2 ms (approx.) |

**Note 1:** PLL inactive.


© 2010-2016 Microchip Technology Inc.

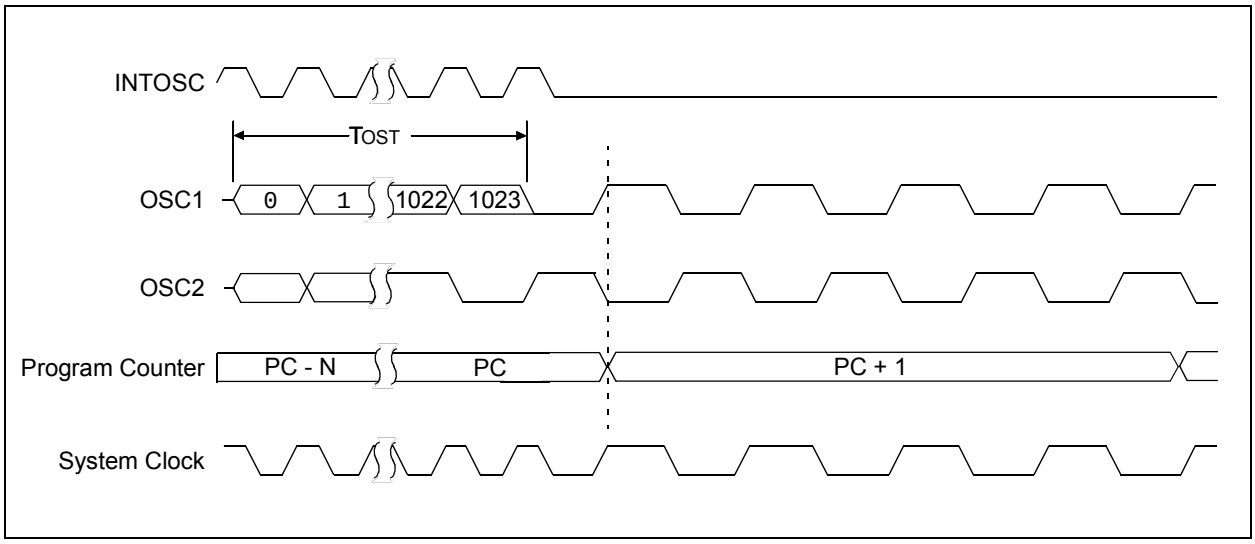
### 5.4.2 TWO-SPEED START-UP SEQUENCE

1. Wake-up from Power-on Reset or Sleep.
2. Instructions begin execution by the internal oscillator at the frequency set in the IRCF<3:0> bits of the OSCCON register.
3. OST enabled to count 1024 clock cycles.
4. OST timed out, wait for falling edge of the internal oscillator.
5. OSTS is set.
6. System clock held low until the next falling edge of new clock (LP, XT or HS mode).
7. System clock is switched to external clock source.

### 5.4.3 CHECKING TWO-SPEED CLOCK STATUS

Checking the state of the OSTS bit of the OSCSTAT register will confirm if the microcontroller is running from the external clock source, as defined by the FOSC<2:0> bits in the Configuration Words, or the internal oscillator.

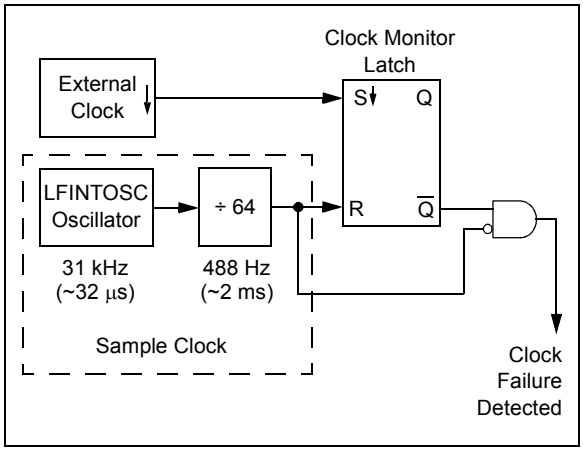
**FIGURE 5-8: TWO-SPEED START-UP**



## 5.5 Fail-Safe Clock Monitor

The Fail-Safe Clock Monitor (FSCM) allows the device to continue operating should the external oscillator fail. The FSCM can detect oscillator failure any time after the Oscillator Start-up Timer (OST) has expired. The FSCM is enabled by setting the FCMEN bit in the Configuration Words. The FSCM is applicable to all external Oscillator modes (LP, XT, HS, EC, Timer1 Oscillator and RC).

**FIGURE 5-9: FSCM BLOCK DIAGRAM**




### 5.5.1 FAIL-SAFE DETECTION

The FSCM module detects a failed oscillator by comparing the external oscillator to the FSCM sample clock. The sample clock is generated by dividing the LFINTOSC by 64. See Figure 5-9. Inside the fail detector block is a latch. The external clock sets the latch on each falling edge of the external clock. The sample clock clears the latch on each rising edge of the sample clock. A failure is detected when an entire half-cycle of the sample clock elapses before the external clock goes low.

### 5.5.2 FAIL-SAFE OPERATION

When the external clock fails, the FSCM switches the device clock to an internal clock source and sets the bit flag OSFIF of the PIR2 register. Setting this flag will generate an interrupt if the OSFIE bit of the PIE2 register is also set. The device firmware can then take steps to mitigate the problems that may arise from a failed clock. The system clock will continue to be sourced from the internal clock source until the device firmware successfully restarts the external oscillator and switches back to external operation.

The internal clock source chosen by the FSCM is determined by the IRCF<3:0> bits of the OSCCON register. This allows the internal oscillator to be configured before a failure occurs.

### 5.5.3 FAIL-SAFE CONDITION CLEARING

The Fail-Safe condition is cleared after a Reset, executing a SLEEP instruction or changing the SCS bits of the OSCCON register. When the SCS bits are changed, the OST is restarted. While the OST is running, the device continues to operate from the INTOSC selected in OSCCON. When the OST times out, the Fail-Safe condition is cleared after successfully switching to the external clock source. The OSFIF bit should be cleared prior to switching to the external clock source. If the Fail-Safe condition still exists, the OSFIF flag will again become set by hardware.

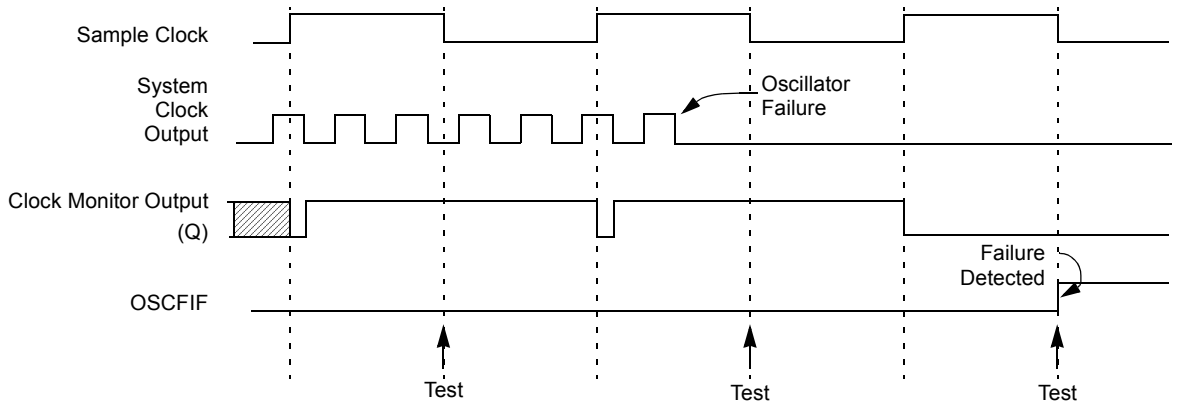
### 5.5.4 RESET OR WAKE-UP FROM SLEEP

The FSCM is designed to detect an oscillator failure after the Oscillator Start-up Timer (OST) has expired. The OST is used after waking up from Sleep and after any type of Reset. The OST is not used with the EC or RC Clock modes so that the FSCM will be active as soon as the Reset or wake-up has completed. When the FSCM is enabled, the Two-Speed Start-up is also enabled. Therefore, the device will always be executing code while the OST is operating.

| | |
|---|---|
| **Note:** | Due to the wide range of oscillator start-up times, the Fail-Safe circuit is not active during oscillator start-up (i.e., after exiting Reset or Sleep). After an appropriate amount of time, the user should check the Status bits in the OSCSTAT register to verify the oscillator start-up and that the system clock switchover has successfully completed. |


© 2010-2016 Microchip Technology Inc.

**FIGURE 5-10: FSCM TIMING DIAGRAM**




**Note:** The system clock is normally at a much higher frequency than the sample clock. The relative frequencies in this example have been chosen for clarity.

## 5.6 Register Definitions: Oscillator Control

**REGISTER 5-1: OSCCON: OSCILLATOR CONTROL REGISTER**

| R/W-0/0 | 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 |
|---|---|---|---|---|---|---|---|
| SPLLEN | 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 **SPLLEN:** Software PLL Enable bit
If PLLEN in Configuration Words = 1:
SPLLEN bit is ignored. 4x PLL is always enabled (subject to oscillator requirements)
If PLLEN in Configuration Words = 0:
1 = 4x PLL Is enabled
0 = 4x PLL is disabled

bit 6-3 **IRCF<3:0>:** Internal Oscillator Frequency Select bits
1111 = 16 MHz HF
1110 = 8 MHz or 32 MHz HF (see **Section 5.2.2.1 “HFINTOSC”**)
1101 = 4 MHz HF
1100 = 2 MHz HF
1011 = 1 MHz HF
1010 = 500 kHz HF(1)
1001 = 250 kHz HF(1)
1000 = 125 kHz HF(1)
0111 = 500 kHz MF (default upon Reset)
0110 = 250 kHz MF
0101 = 125 kHz MF
0100 = 62.5 kHz MF
0011 = 31.25 kHz HF(1)
0010 = 31.25 kHz MF
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 = Timer1 oscillator
00 = Clock determined by FOSC<2:0> in Configuration Words

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


© 2010-2016 Microchip Technology Inc.

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

| R-1/q | R-0/q | R-q/q | R-0/q | R-0/q | R-q/q | R-0/0 | R-0/q |
|---|---|---|---|---|---|---|---|
| T1OSCR | PLLR | OSTS | HFIOFR | HFIOFL | MFIOFR | LFIOFR | HFIOFS |
| bit 7 | | | | | | | bit 0 |

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

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

bit 6 **PLLR** 4x PLL Ready bit
1 = 4x PLL is ready
0 = 4x PLL is not ready

bit 5 **OSTS:** Oscillator Start-up Time-out Status bit
1 = Running from the clock defined by the FOSC<2:0> bits of the Configuration Words
0 = Running from an internal oscillator (FOSC<2:0> = 100)

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

bit 3 **HFIOFL:** High Frequency Internal Oscillator Locked bit
1 = HFINTOSC is at least 2% accurate
0 = HFINTOSC is not 2% accurate

bit 2 **MFIOFR:** Medium Frequency Internal Oscillator Ready bit
1 = MFINTOSC is ready
0 = MFINTOSC is not ready

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 is at least 0.5% accurate
0 = HFINTOSC is not 0.5% accurate

**REGISTER 5-3: OSCTUNE: OSCILLATOR TUNING 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 |
|---|---|---|---|---|---|---|---|
| — | — | TUN<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 | |

bit 7-6 **Unimplemented:** Read as '0'

bit 5-0 **TUN<5:0>:** Frequency Tuning bits

100000 = Minimum frequency
•
•
•
111111 =
000000 = Oscillator module is running at the factory-calibrated frequency.
000001 =
•
•
•
011110 =
011111 = Maximum frequency

**TABLE 5-2: SUMMARY OF REGISTERS ASSOCIATED WITH CLOCK SOURCES**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| OSCCON | SPLLEN | IRCF<3:0> | | | | — | SCS<1:0> | | 74 |
| OSCSTAT | T1OSCR | PLLR | OSTS | HFIOFR | HFIOFL | MFIOFR | LFIOFR | HFIOFS | 75 |
| OSCTUNE | — | — | TUN<5:0> | | | | | | 76 |
| PIE2 | OSFIE | C2IE | C1IE | EEIE | BCLIE | LCDIE | C3IE | CCP2IE(1) | 92 |
| PIR2 | OSFIF | C2IF | C1IF | EEIF | BCLIF | LCDIF | C3IF | CCP2IF(1) | 96 |
| T1CON | TMR1CS<1:0> | | T1CKPS<1:0> | | T1OSCEN | $\overline{\text{T1SYNC}}$ | — | TMR1ON | 197 |

**Legend:** — = unimplemented location, read as '0'. Shaded cells are not used by clock sources.
**Note 1:** PIC16F1947 only.

**TABLE 5-3: SUMMARY OF CONFIGURATION WORD WITH CLOCK SOURCES**

| Name | Bits | Bit -/7 | Bit -/6 | Bit 13/5 | Bit 12/4 | Bit 11/3 | Bit 10/2 | Bit 9/1 | Bit 8/0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|---|
| CONFIG1 | 13:8 | — | — | FCMEN | IESO | $\overline{\text{CLKOUTEN}}$ | BOREN<1:0> | | $\overline{\text{CPD}}$ | 54 |
| | 7:0 | $\overline{\text{CP}}$ | MCLRE | $\overline{\text{PWRTE}}$ | WDTE<1:0> | | FOSC<2:0> | | | |
| CONFIG2 | 13:8 | — | — | LVP | $\overline{\text{DEBUG}}$ | — | BORV | STVREN | PLLEN | 56 |
| | 7:0 | — | — | — | VCAPEN | — | — | WRT<1:0> | | |

**Legend:** — = unimplemented location, read as '0'. Shaded cells are not used by clock sources.
**Note 1:** PIC16F1946/47 only.


© 2010-2016 Microchip Technology Inc.

## 6.0 RESETS

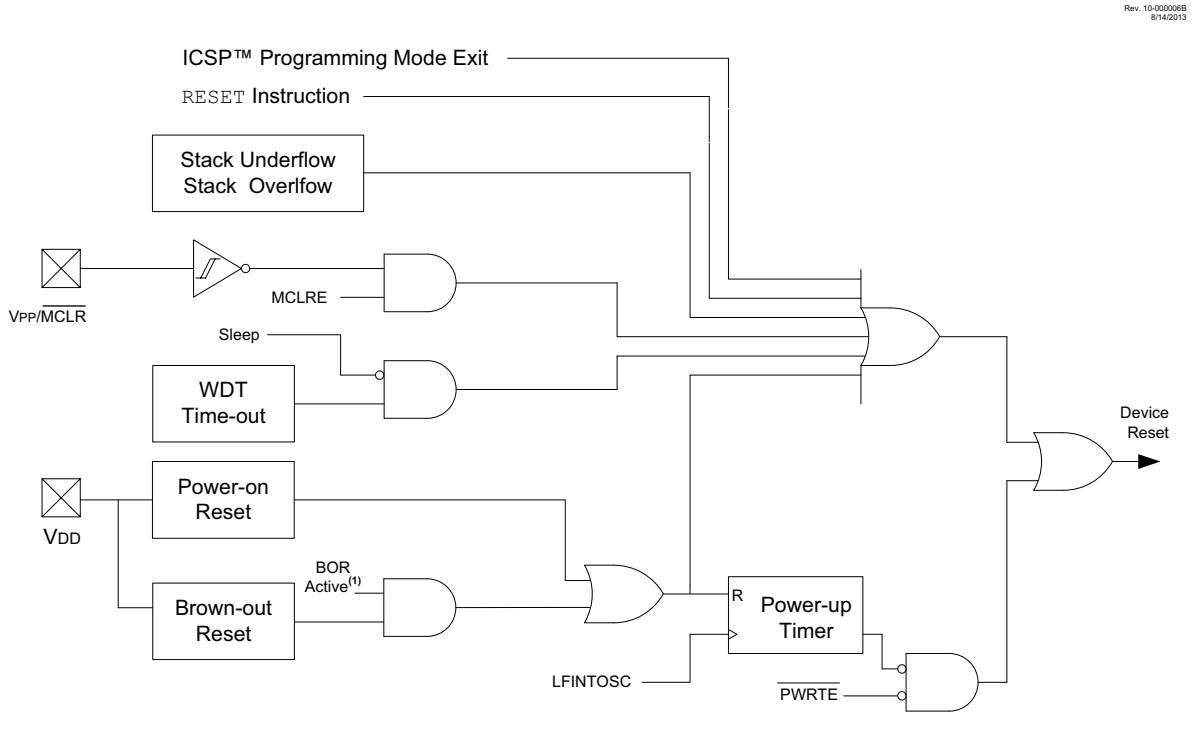
There are multiple ways to reset this device:

- Power-on Reset (POR)
- Brown-out Reset (BOR)
- $\overline{\text{MCLR}}$ 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**




**Note:** See Table 6-1 for BOR active conditions.

## 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 VBOR 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(1) (BORRDY = 1) |
| 10 | X | Awake | Active | Waits for BOR ready (BORRDY = 1) |
| | | Sleep | Disabled | |
| 01 | 1 | X | Active | Waits for BOR ready(1) (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

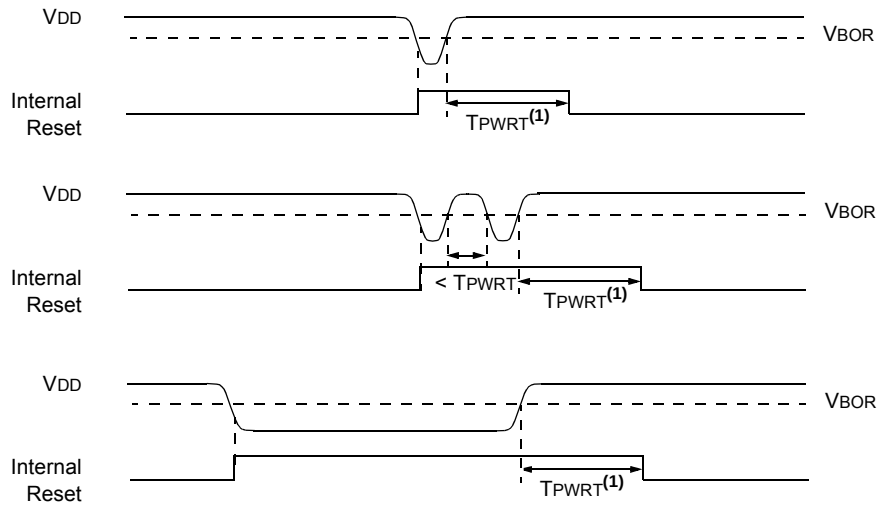
When the BOREN bits of Configuration Words are programmed to ‘01’, the BOR is controlled by the SBOREN bit of the BORCON register. The device start-up is not delayed by the BOR ready condition or the VDD level.

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.


© 2010-2016 Microchip Technology Inc.

**FIGURE 6-2: BROWN-OUT SITUATIONS**




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

## 6.3 Register Definitions: BOR Control

**REGISTER 6-1: BORCON: BROWN-OUT RESET CONTROL REGISTER**

| R/W-1/u | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R-q/u |
|---|---|---|---|---|---|---|---|
| SBOREN | — | — | — | — | — | — | 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:
SBOREN is read/write, but has no effect on the BOR.
If BOREN <1:0> in Configuration Words = 01:
1 = BOR Enabled
0 = BOR Disabled

bit 6-1 **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

## 6.4 MCLR

The MCLR is an optional external input that can reset the device. The MCLR function is controlled by the MCLRE bit of Configuration Words and the LVP bit of Configuration Words (Table 6-2).

**TABLE 6-2: MCLR CONFIGURATION**

| MCLRE | LVP | MCLR |
|---|---|---|
| 0 | 0 | Disabled |
| 1 | 0 | Enabled |
| x | 1 | Enabled |

### 6.4.1 MCLR ENABLED

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

The device has a noise filter in the MCLR Reset path. The filter will detect and ignore small pulses.

**Note:** A Reset does not drive the MCLR pin low.

### 6.4.2 MCLR DISABLED

When MCLR is disabled, the pin functions as a general purpose input and the internal weak pull-up is under software control. See **Section 12.15 "PORTG Registers"** for more information.

## 6.5 Watchdog Timer (WDT) Reset

The Watchdog Timer generates a Reset if the firmware does not issue a CLRWDT instruction within the time-out period. The TO and PD bits in the STATUS register are changed to indicate the WDT Reset. See **Section 10.0 "Watchdog Timer (WDT)"** for more information.

## 6.6 RESET Instruction

A RESET instruction will cause a device Reset. The RI bit in the PCON register will be set to '0'. See Table for default conditions after a RESET instruction has occurred.

## 6.7 Stack Overflow/Underflow Reset

The device can reset when the Stack Overflows or Underflows. The STKOVF or STKUNF bits of the PCON register indicate the Reset condition. These Resets are enabled by setting the STVREN bit in Configuration Words. See **Section 3.5.2 "Overflow/Underflow Reset"** for more information.

## 6.8 Programming Mode Exit

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

## 6.9 Power-Up Timer

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

The Power-up Timer is controlled by the PWRTE bit of Configuration Words.

## 6.10 Start-up Sequence

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

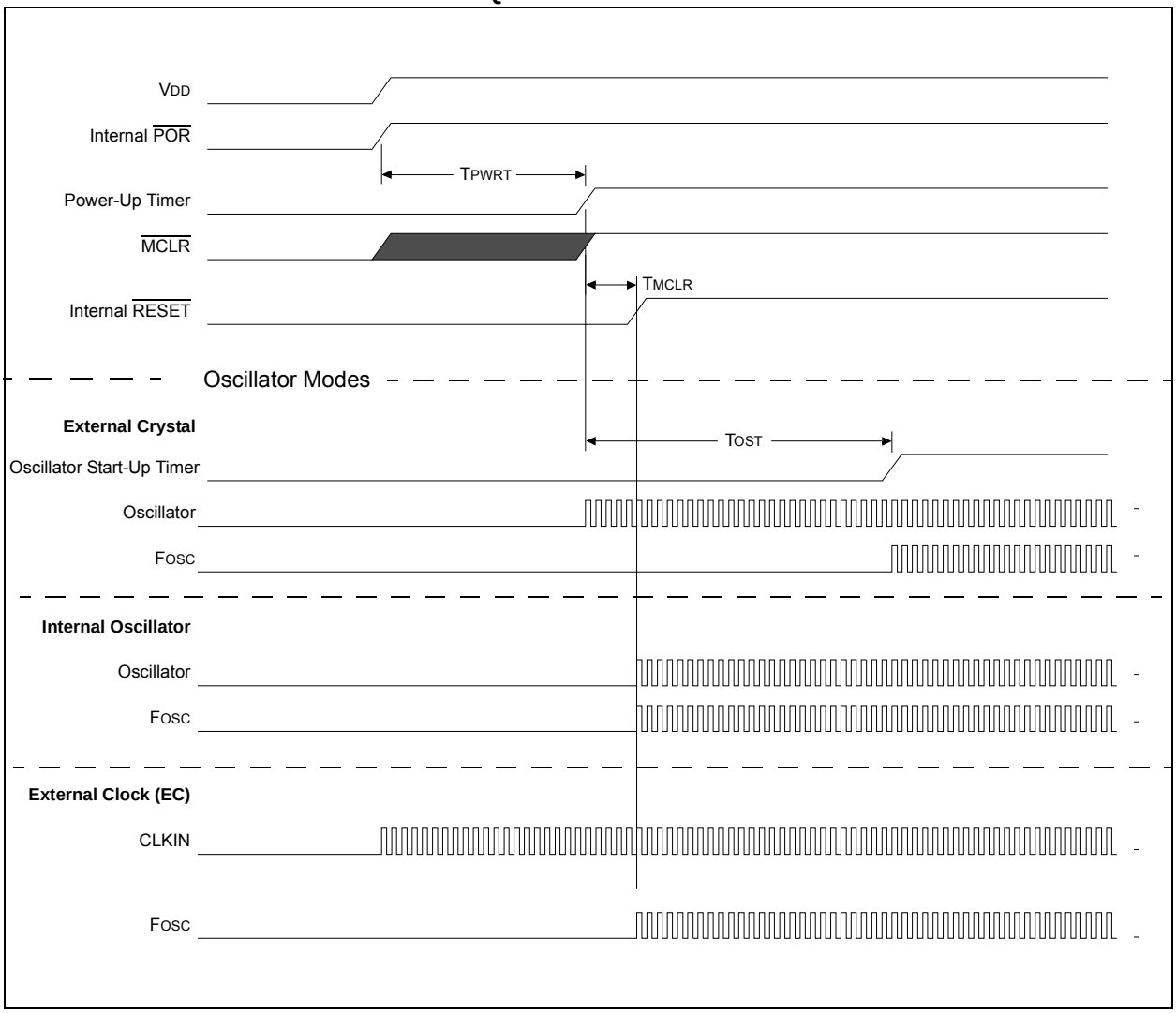
1. Power-up Timer runs to completion (if enabled).
2. Oscillator start-up timer runs to completion (if required for oscillator source).
3. MCLR must be released (if enabled).

The total time-out will vary based on oscillator configuration and Power-up Timer configuration. See **Section 5.0 "Oscillator Module (With Fail-Safe Clock Monitor)"** for more information.

The Power-up Timer and oscillator start-up timer run independently of MCLR Reset. If MCLR is kept low long enough, the Power-up Timer and oscillator start-up timer will expire. Upon bringing MCLR high, the device will begin execution immediately (see Figure 6-3). This is useful for testing purposes or to synchronize more than one device operating in parallel.


© 2010-2016 Microchip Technology Inc.

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



## 6.11 Determining the Cause of a Reset

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

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

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

**TABLE 6-4: RESET CONDITION FOR SPECIAL REGISTERS(2)**

| Condition | Program Counter | STATUS Register | PCON Register |
|---|---|---|---|
| Power-on Reset | 0000h | ---1 1000 | 00-- 110x |
| $\overline{\text{MCLR}}$ Reset during normal operation | 0000h | ---u uuuu | uu-- 0uuu |
| $\overline{\text{MCLR}}$ Reset during Sleep | 0000h | ---1 0uuu | uu-- 0uuu |
| WDT Reset | 0000h | ---0 uuuu | 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(1) | ---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 Global Enable bit (GIE) is set, the return address is pushed on the stack and PC is loaded with the interrupt vector (0004h) after execution of PC + 1.

**2:** If a Status bit is not implemented, that bit will be read as '0'.


© 2010-2016 Microchip Technology Inc.

## 6.12 Power Control (PCON) Register

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

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

The PCON register bits are shown in Register 6-2.

## 6.13 Register Definitions: Power Control

**REGISTER 6-2: PCON: POWER CONTROL REGISTER**

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

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

bit 7 **STKOVF:** Stack Overflow Flag bit
1 = A Stack Overflow occurred
0 = A Stack Overflow has not occurred or set to ‘0’ by firmware

bit 6 **STKUNF:** Stack Underflow Flag bit
1 = A Stack Underflow occurred
0 = A Stack Underflow has not occurred or set to ‘0’ by firmware

bit 5-4 **Unimplemented:** Read as ‘0’

bit 3 **$\overline{RMCLR}$:** $\overline{MCLR}$ Reset Flag bit
1 = A $\overline{MCLR}$ Reset has not occurred or set to ‘1’ by firmware
0 = A $\overline{MCLR}$ Reset has occurred (set to ‘0’ in hardware when a $\overline{MCLR}$ Reset occurs)

bit 2 **$\overline{RI}$:** RESET Instruction Flag bit
1 = A RESET instruction has not been executed or set to ‘1’ by firmware
0 = A RESET instruction has been executed (set to ‘0’ in hardware upon executing a RESET instruction)

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

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

**TABLE 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 | — | — | — | — | — | — | BORRDY | 79 |
| PCON | STKOVF | STKUNF | — | — | $\overline{\text{RMCLR}}$ | $\overline{\text{RI}}$ | $\overline{\text{POR}}$ | $\overline{\text{BOR}}$ | 83 |
| STATUS | — | — | — | $\overline{\text{TO}}$ | $\overline{\text{PD}}$ | Z | DC | C | 22 |
| WDTCON | — | — | WDTPS<4:0> | | | | | SWDTEN | 104 |

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

**Note 1:** Other (non Power-up) Resets include $\overline{\text{MCLR}}$ Reset and Watchdog Timer Reset during normal operation.


© 2010-2016 Microchip Technology Inc.

## 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 PIC microcontroller from Sleep mode.

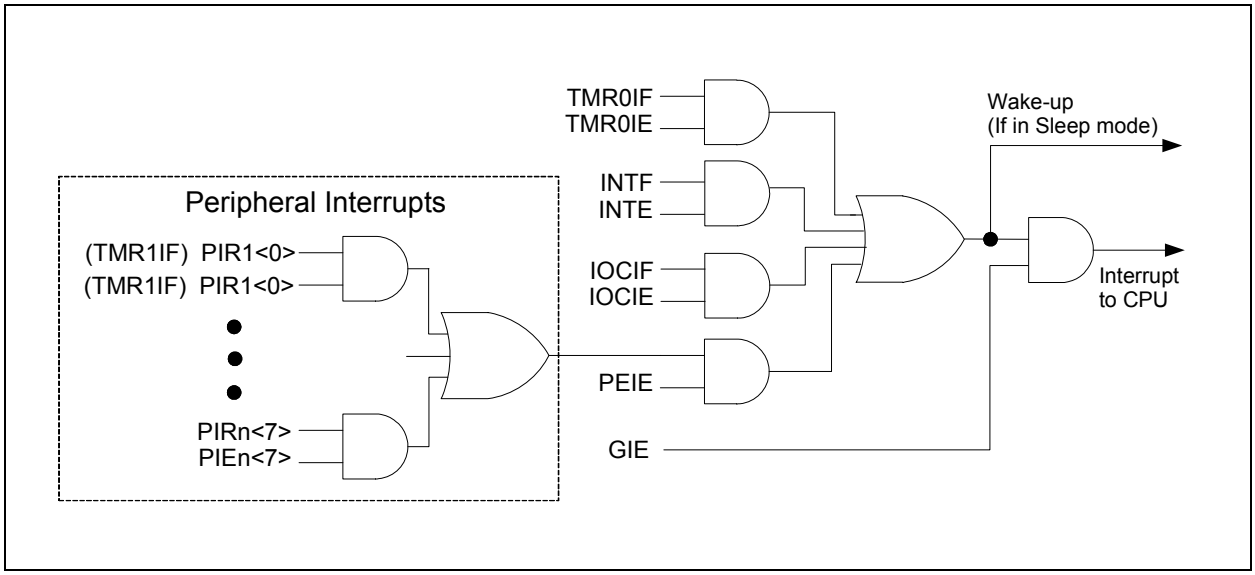
This chapter contains the following information for Interrupts:

- 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**



## 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, PIE3 and PIE4 registers)

The INTCON, PIR1, PIR2, PIR3 and PIR4 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 3 or 4 instruction cycles. For asynchronous interrupts, the latency is 3 to 5 instruction cycles, depending on when the interrupt occurs. See Figure 7-2 and Figure 7-3 for more details.


© 2010-2016 Microchip Technology Inc.

**FIGURE 7-2: INTERRUPT LATENCY**

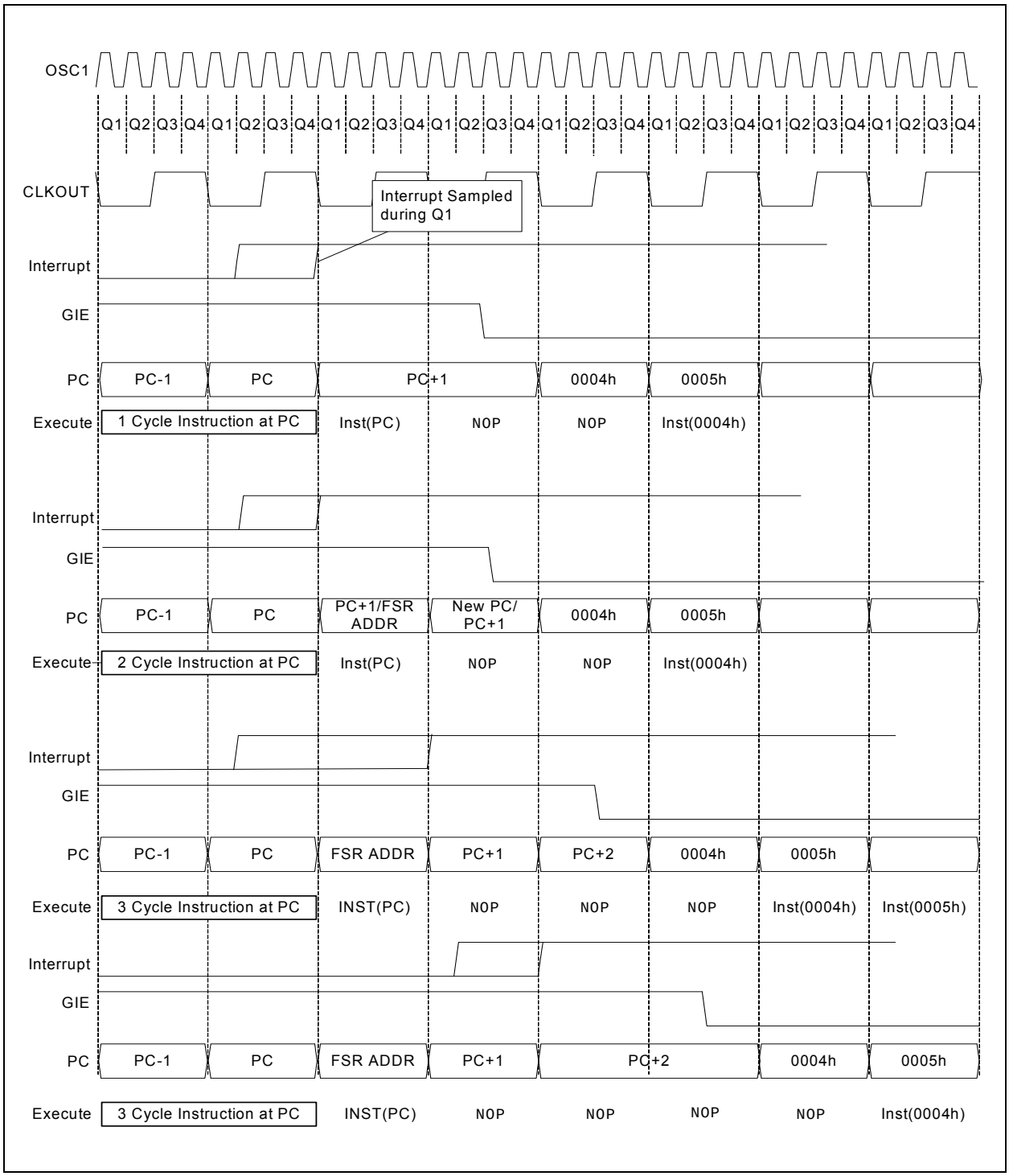
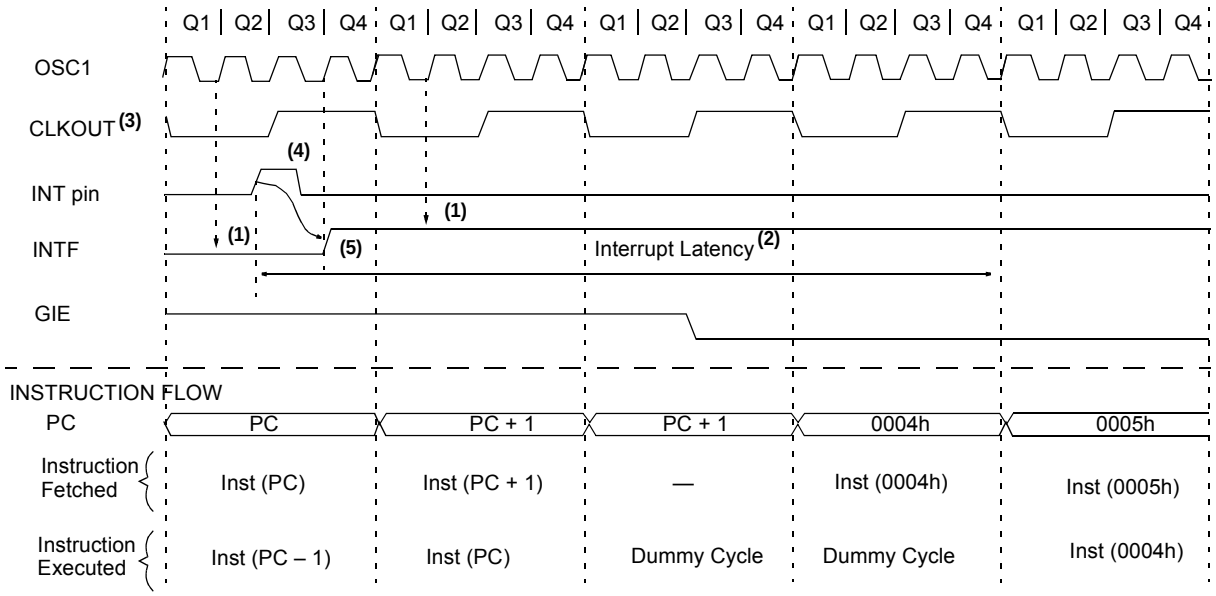


**FIGURE 7-3: INT PIN INTERRUPT TIMING**




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

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

**3:** CLKOUT not available in all Oscillator modes.

**4:** For minimum width of INT pulse, refer to AC specifications in **Section 30.0 "Electrical Specifications"**.

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


© 2010-2016 Microchip Technology Inc.

## 7.3 Interrupts During Sleep

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

On waking from Sleep, if the GIE bit is also set, the processor will branch to the interrupt vector. Otherwise, the processor will continue executing instructions after the SLEEP instruction. The instruction directly after the SLEEP instruction will always be executed before branching to the ISR. Refer to the **Section 9.0 "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.

## 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 | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF |
| bit 7 | | | | | | | bit 0 |

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

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

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

bit 5 **TMR0IE:** Timer0 Overflow Interrupt Enable bit
1 = Enables the Timer0 interrupt
0 = Disables the Timer0 interrupt

bit 4 **INTE:** INT External Interrupt Enable bit
1 = Enables the INT external interrupt
0 = Disables the INT external interrupt

bit 3 **IOCIE:** Interrupt-on-Change 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
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:** The IOCIF Flag bit is read-only and cleared when all the interrupt-on-change flags in the IOCBF register have been cleared by software.

**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.


© 2010-2016 Microchip Technology Inc.

**REGISTER 7-2: PIE1: PERIPHERAL INTERRUPT ENABLE REGISTER 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 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| TMR1GIE | ADIE | RCIE | TXIE | SSPIE | CCP1IE | 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:** A/D Converter (ADC) Interrupt Enable bit
1 = Enables the ADC interrupt
0 = Disables the ADC interrupt

bit 5 **RCIE:** USART1 Receive Interrupt Enable bit
1 = Enables the USART1 receive interrupt
0 = Disables the USART1 receive interrupt

bit 4 **TXIE:** USART1 Transmit Interrupt Enable bit
1 = Enables the USART1 transmit interrupt
0 = Disables the USART1 transmit interrupt

bit 3 **SSPIE:** Synchronous Serial Port (MSSP1) Interrupt Enable bit
1 = Enables the MSSP1 interrupt
0 = Disables the MSSP1 interrupt

bit 2 **CCP1IE:** CCP1 Interrupt Enable bit
1 = Enables the CCP1 interrupt
0 = Disables the CCP1 interrupt

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.

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

| 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 |
|---|---|---|---|---|---|---|---|
| OSFIE | C2IE | C1IE | EEIE | BCLIE | LCDIE | C3IE | CCP2IE |
| 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 **OSFIE:** Oscillator Fail Interrupt Enable bit
- 1 = Enables the Oscillator Fail interrupt
- 0 = Disables the Oscillator Fail interrupt

bit 6 **C2IE:** Comparator C2 Interrupt Enable bit
- 1 = Enables the Comparator C2 interrupt
- 0 = Disables the Comparator C2 interrupt

bit 5 **C1IE:** Comparator C1 Interrupt Enable bit
- 1 = Enables the Comparator C1 interrupt
- 0 = Disables the Comparator C1 interrupt

bit 4 **EEIE:** EEPROM Write Completion Interrupt Enable bit
- 1 = Enables the EEPROM Write Completion interrupt
- 0 = Disables the EEPROM Write Completion interrupt

bit 3 **BCLIE:** MSSP1 Bus Collision Interrupt Enable bit
- 1 = Enables the MSSP1 Bus Collision Interrupt
- 0 = Disables the MSSP1 Bus Collision Interrupt

bit 2 **LCDIE:** LCD Module Interrupt Enable bit
- 1 = Enables the LCD module interrupt
- 0 = Disables the LCD module interrupt

bit 1 **C3IE:** Comparator C3 Interrupt Enable bit
- 1 = Enables the Comparator C3 interrupt
- 0 = Disables the Comparator C3 interrupt

bit 0 **CCP2IE:** CCP2 Interrupt Enable bit
- 1 = Enables the CCP2 interrupt
- 0 = Disables the CCP2 interrupt

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


© 2010-2016 Microchip Technology Inc.

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

| U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | U-0 | R/W-0/0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | CCP5IE | CCP4IE | CCP3IE | TMR6IE | — | TMR4IE | — |
| 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 **CCP5IE:** CCP5 Interrupt Enable bit
- 1 = Enables the CCP5 interrupt
- 0 = Disables the CCP5 interrupt

bit 5 **CCP4IE:** CCP4 Interrupt Enable bit
- 1 = Enables the CCP4 interrupt
- 0 = Disables the CCP4 interrupt

bit 4 **CCP3IE:** CCP3 Interrupt Enable bit
- 1 = Enables the CCP3 interrupt
- 0 = Disables the CCP3 interrupt

bit 3 **TMR6IE:** TMR6 to PR6 Match Interrupt Enable bit
- 1 = Enables the TMR6 to PR6 Match interrupt
- 0 = Disables the TMR6 to PR6 Match interrupt

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

bit 1 **TMR4IE:** TMR4 to PR4 Match Interrupt Enable bit
- 1 = Enables the TMR4 to PR4 Match interrupt
- 0 = Disables the TMR4 to PR4 Match interrupt

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

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

**REGISTER 7-5: PIE4: PERIPHERAL INTERRUPT ENABLE REGISTER 4**

| U-0 | U-0 | R/W-0/0 | R/W-0/0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | RC2IE | TX2IE | — | — | BCL2IE | SSP2IE |
| 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 **RC2IE:** USART2 Receive Interrupt Enable bit
1 = Enables the USART2 receive interrupt
0 = Disables the USART2 receive interrupt

bit 4 **TX2IE:** USART2 Transmit Interrupt Enable bit
1 = Enables the USART2 transmit interrupt
0 = Disables the USART2 transmit interrupt

bit 3-2 **Unimplemented:** Read as '0'

bit 1 **BCL2IE:** MSSP2 Bus Collision Interrupt Enable bit
1 = Enables the MSSP2 Bus Collision Interrupt
0 = Disables the MSSP2 Bus Collision Interrupt

bit 0 **SSP2IE:** Synchronous Serial Port (MSSP2) Interrupt Enable bit
1 = Enables the MSSP2 interrupt
0 = Disables the MSSP2 interrupt

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


© 2010-2016 Microchip Technology Inc.

**REGISTER 7-6: PIR1: PERIPHERAL INTERRUPT REQUEST REGISTER 1**

| R/W-0/0 | R/W-0/0 | R-0/0 | R-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| TMR1GIF | ADIF | RCIF | TXIF | SSPIF | CCP1IF | 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:** A/D Converter Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 5 **RCIF:** USART1 Receive Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 4 **TXIF:** USART1 Transmit Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 3 **SSPIF:** Synchronous Serial Port (MSSP1) Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 2 **CCP1IF:** CCP1 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

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 Enable bit, GIE, of the INTCON register. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

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

| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | U-0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| OSFIF | C2IF | C1IF | EEIF | BCLIF | LCDIF | — | CCP2IF |
| 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 **OSFIF:** Oscillator Fail Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 6 **C2IF:** Comparator C2 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 5 **C1IF:** Comparator C1 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 4 **EEIF:** EEPROM Write Completion Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 3 **BCLIF:** MSSP1 Bus Collision Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 2 **LCDIF:** LCD Module Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

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

bit 0 **CCP2IF:** CCP2 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.


© 2010-2016 Microchip Technology Inc.

**REGISTER 7-8: PIR3: PERIPHERAL INTERRUPT REQUEST REGISTER 3**

| 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 |
|---|---|---|---|---|---|---|---|
| — | CCP5IF | CCP4IF | CCP3IF | TMR6IF | — | TMR4IF | — |
| 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 **CCP5IF:** CCP5 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 5 **CCP4IF:** CCP4 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 4 **CCP3IF:** CCP3 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 3 **TMR6IF:** TMR6 to PR6 Match Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

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

bit 1 **TMR4IF:** TMR4 to PR4 Match Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 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 Enable bit, GIE, of the INTCON register. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

**REGISTER 7-9: PIR4: PERIPHERAL INTERRUPT REQUEST REGISTER 4**

| U-0 | U-0 | R/W-0/0 | R/W-0/0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | RC2IF | TX2IF | — | — | BCL2IF | SSP2IF |
| 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 **RC2IF:** USART2 Receive Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 4 **TX2IF:** USART2 Transmit Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 3-2 **Unimplemented:** Read as '0'

bit 1 **BCL2IF:** MSSP2 Bus Collision Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 0 **SSP2IF:** Synchronous Serial Port (MSSP2) 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 | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| OPTION_REG | $\overline{\text{WPUEN}}$ | INTEDG | T0CS | T0SE | PSA | PS<2:0> | | | 188 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 91 |
| PIE2 | OSFIE | C2IE | C1IE | EEIE | BCLIE | LCDIE | C3IE | CCP2IE | 92 |
| PIE3 | — | CCP5IE | CCP4IE | CCP3IE | TMR6IE | — | TMR4IE | — | 93 |
| PIE4 | — | — | RC2IE | TX2IE | — | — | BCL2IE | SSP2IE | 94 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 95 |
| PIR2 | OSFIF | C2IF | C1IF | EEIF | BCLIF | LCDIF | C3IF | CCP2IF | 96 |
| PIR3 | — | CCP5IF | CCP4IF | CCP3IF | TMR6IF | — | TMR4IF | — | 97 |
| PIR4 | — | — | RC2IF | TX2IF | — | — | BCL2IF | SSP2IF | 98 |

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


© 2010-2016 Microchip Technology Inc.

## 8.0 LOW DROPOUT (LDO) VOLTAGE REGULATOR

The PIC16F1946/47 has an internal Low Dropout Regulator (LDO) which provides operation above 3.6V. The LDO regulates a voltage for the internal device logic while permitting the VDD and I/O pins to operate at a higher voltage. There is no user enable/disable control available for the LDO, it is always active. The PIC16LF1946/47 operates at a maximum VDD of 3.6V and does not incorporate an LDO.

A device I/O pin may be configured as the LDO voltage output, identified as the VCAP pin. Although not required, an external low-ESR capacitor may be connected to the VCAP pin for additional regulator stability.

The VCAPEN bit of Configuration Words enables or disables the VCAP pin. Refer to Table 8-1.

**TABLE 8-1: VCAPEN SELECT BIT**

| VCAPEN | Pin |
|---|---|
| 0 | RF0 |
| 1 | No VCAP |

On power-up, the external capacitor will load the LDO voltage regulator. To prevent erroneous operation, the device is held in Reset while a constant current source charges the external capacitor. After the cap is fully charged, the device is released from Reset. For more information on the constant current rate, refer to the LDO Regulator Characteristics Table in **Section 30.0 "Electrical Specifications"**.

**TABLE 8-2: SUMMARY OF CONFIGURATION WORD WITH LDO**

| 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 |
|---|---|---|---|---|---|---|---|---|---|---|
| CONFIG2 | 13:8 | — | — | LVP | $\overline{\text{DEBUG}}$ | — | BORV | STVREN | PLLEN | 56 |
| | 7:0 | — | — | — | VCAPEN | — | — | WRT1 | WRT0 | |

**Legend:** — = unimplemented locations read as '0'. Shaded cells are not used by LDO.

## 9.0 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 oscillator is unaffected and peripherals that operate from it may continue operation in Sleep.
7. ADC is unaffected, if the dedicated FRC clock is selected.
8. Capacitive Sensing oscillator is unaffected.
9. I/O ports maintain the status they had before SLEEP was executed (driving high, low or high-impedance).
10. Resets other than WDT are not affected by Sleep mode.

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

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

- I/O pins should not be floating
- External circuitry sinking current from I/O pins
- Internal circuitry sourcing current from I/O pins
- Current draw from pins with internal weak pull-ups
- Modules using 31 kHz LFINTOSC
- Modules using Timer1 oscillator

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

Examples of internal circuitry that might be sourcing current include modules such as the DAC and FVR modules. See **Section 17.0 "Digital-to-Analog Converter (DAC) Module"** and **Section 14.0 "Fixed Voltage Reference (FVR)"** for more information on these modules.

### 9.1 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 6.11 "Determining the Cause of a Reset"**.

When the SLEEP instruction is being executed, the next instruction (PC + 1) is prefetched. For the device to wake-up through an interrupt event, the corresponding interrupt enable bit must be enabled. Wake-up will occur regardless of the state of the GIE bit. If the GIE bit is disabled, the device continues execution at the instruction after the SLEEP instruction. If the GIE bit is enabled, the device executes the instruction after the SLEEP instruction; the device will then call the Interrupt Service Routine. In cases where the execution of the instruction following SLEEP is not desirable, the user should have a NOP after the SLEEP instruction.

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


© 2010-2016 Microchip Technology Inc.

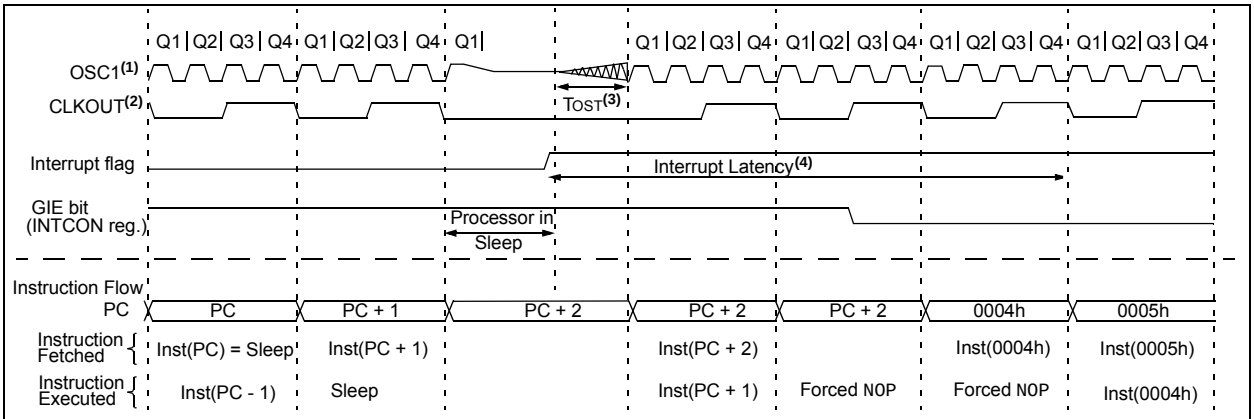
### 9.1.1 WAKE-UP USING INTERRUPTS

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 is executed, test the $\overline{PD}$ bit. If the $\overline{PD}$ bit is set, the SLEEP instruction was executed as a NOP.

**FIGURE 9-1: WAKE-UP FROM SLEEP THROUGH INTERRUPT**




**Note 1:** XT, HS or LP Oscillator mode assumed.
**2:** CLKOUT is not available in XT, HS, or LP Oscillator modes, but shown here for timing reference.
**3:** TOST = 1024 TOSC (drawing not to scale). This delay applies only to XT, HS or LP Oscillator modes.
**4:** GIE = 1 assumed. In this case after wake-up, the processor calls the ISR at 0004h. If GIE = 0, execution will continue in-line.

**TABLE 9-1: SUMMARY OF REGISTERS ASSOCIATED WITH POWER-DOWN MODE**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| IOCBF | IOCBF7 | IOCBF6 | IOCBF5 | IOCBF4 | IOCBF3 | IOCBF2 | IOCBF1 | IOCBF0 | 148 |
| IOCBN | IOCBN7 | IOCBN6 | IOCBN5 | IOCBN4 | IOCBN3 | IOCBN2 | IOCBN1 | IOCBN0 | 148 |
| IOCBP | IOCBP7 | IOCBP6 | IOCBP5 | IOCBP4 | IOCBP3 | IOCBP2 | IOCBP1 | IOCBP0 | 148 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 91 |
| PIE2 | OSFIE | C2IE | C1IE | EEIE | BCLIE | LCDIE | C3IE | CCP2IE | 92 |
| PIE3 | — | CCP5IE | CCP4IE | CCP3IE | TMR6IE | — | TMR4IE | — | 93 |
| PIE4 | — | — | RC2IE | TX2IE | — | — | BCL2IE | SSP2IE | 94 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 95 |
| PIR2 | OSFIF | C2IF | C1IF | EEIF | BCLIF | LCDIF | C3IF | CCP2IF | 96 |
| PIR3 | — | CCP5IF | CCP4IF | CCP3IF | TMR6IF | — | TMR4IF | — | 97 |
| PIR4 | — | — | RC2IF | TX2IF | — | — | BCL2IF | SSP2IF | 98 |
| STATUS | — | — | — | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | 22 |
| WDTCON | — | — | WDTPS<4:0> | | | | | SWDTEN | 104 |

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

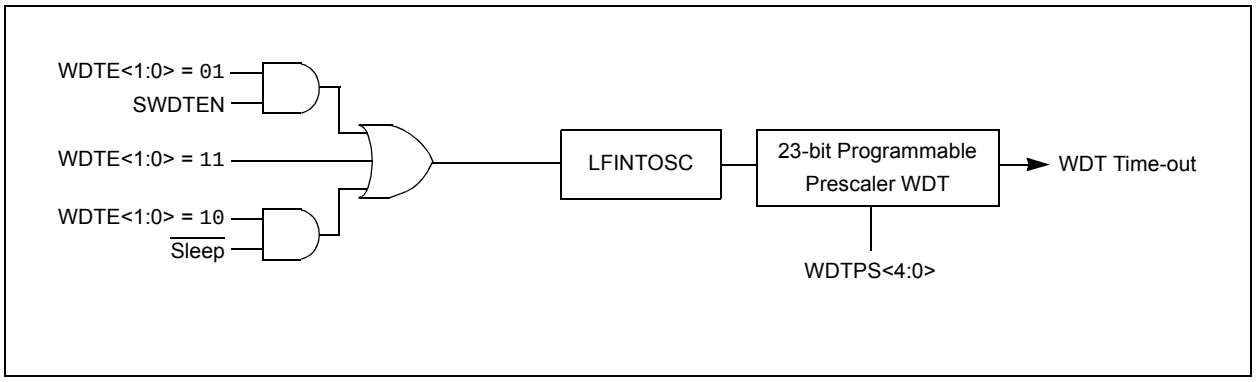
## 10.0 WATCHDOG TIMER (WDT)

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 10-1: WATCHDOG TIMER BLOCK DIAGRAM**





© 2010-2016 Microchip Technology Inc.

## 10.1 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 30.0 "Electrical Specifications"** for the LFINTOSC tolerances.

## 10.2 WDT Operating Modes

The Watchdog Timer module has four operating modes controlled by the WDTE<1:0> bits in Configuration Words. See Table 10-1.

### 10.2.1 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.

### 10.2.2 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.

### 10.2.3 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 10-1 for more details.

**TABLE 10-1: WDT OPERATING MODES**

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

## 10.3 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.

## 10.4 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
- Oscillator Start-up Timer (OST) is running

See Table 10-2 for more information.

## 10.5 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. The WDT remains clear until the OST, if enabled, completes. See **Section 5.0 "Oscillator Module (With Fail-Safe Clock Monitor)"** for more information on the OST.

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. See **Section 3.0 "Memory Organization"** and STATUS register (**Register 3-1**) for more information.

**TABLE 10-2: WDT CLEARING CONDITIONS**

| Conditions | WDT |
|---|---|
| WDTE<1:0> = 00 | Cleared |
| WDTE<1:0> = 01 and SWDTEN = 0 | |
| WDTE<1:0> = 10 and enter Sleep | |
| CLRWDT Command | |
| Oscillator Fail Detected | |
| Exit Sleep + System Clock = T1OSC, EXTRC, INTOSC, EXTCLK | |
| Exit Sleep + System Clock = XT, HS, LP | Cleared until the end of OST |
| Change INTOSC divider (IRCF bits) | Unaffected |

## 10.6 Register Definitions: Watchdog Control

**REGISTER 10-1: WDTCON: WATCHDOG TIMER CONTROL REGISTER**

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

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

bit 7-6 **Unimplemented:** Read as '0'

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

Bit Value = Prescale Rate

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

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

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

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


© 2010-2016 Microchip Technology Inc.

**TABLE 10-3: 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> | | 74 |
| STATUS | — | — | — | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | 22 |
| WDTCON | — | — | WDTPS<4:0> | | | | | SWDTEN | 104 |

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

**TABLE 10-4: 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 | — | — | FCMEN | IESO | $\overline{CLKOUTEN}$ | BOREN<1:0> | | $\overline{CPD}$ | 54 |
| | 7:0 | $\overline{CP}$ | MCLRE | $\overline{PWRTE}$ | WDTE<1:0> | | FOSC<2:0> | | | |

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

## 11.0 DATA EEPROM AND FLASH PROGRAM MEMORY CONTROL

The data EEPROM and Flash program memory are readable and writable during normal operation (full $V_{DD}$ range). These memories are not directly mapped in the register file space. Instead, they are indirectly addressed through the Special Function Registers (SFRs). There are six SFRs used to access these memories:

- EECON1
- EECON2
- EEDATL
- EEDATH
- EEADRL
- EEADRH

When interfacing the data memory block, EEDATL holds the 8-bit data for read/write, and EEADRL holds the address of the EEDATL location being accessed. These devices have 256 bytes of data EEPROM with an address range from 0h to 0FFh.

When accessing the program memory block, the EEDATH:EEDATL register pair forms a 2-byte word that holds the 14-bit data for read/write, and the EEADRL and EEADRH registers form a 2-byte word that holds the 15-bit address of the program memory location being read.

The EEPROM data memory allows byte read and write. An EEPROM byte write automatically erases the location and writes the new data (erase before write).

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 voltage range of the device for byte or word operations.

Depending on the setting of the Flash Program Memory Self Write Enable bits WRT<1:0> of the Configuration Words, the device may or may not be able to write certain blocks of the program memory. However, reads from the program memory are always allowed.

When the device is code-protected, the device programmer can no longer access data or program memory. When code-protected, the CPU may continue to read and write the data EEPROM memory and Flash program memory.

### 11.1 EEADRL and EEADRH Registers

The EEADRH:EEADRL register pair can address up to a maximum of 256 bytes of data EEPROM or 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 EEADRH register and the LSB is written to the EEADRL register. When selecting a EEPROM address value, only the LSB of the address is written to the EEADRL register.

#### 11.1.1 EECON1 AND EECON2 REGISTERS

EECON1 is the control register for EE memory accesses.

Control bit EEPGD determines if the access will be a program or data memory access. When clear, any subsequent operations will operate on the EEPROM memory. When set, any subsequent operations will operate on the program memory. On Reset, EEPROM is selected by default.

Control bits RD and WR initiate read and write, respectively. These bits cannot be cleared, only set, in software. They are cleared in 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.

Interrupt flag bit EEIF of the PIR2 register is set when write is complete. It must be cleared in the software.

Reading EECON2 will read all ‘0’s. The EECON2 register is used exclusively in the data EEPROM write sequence. To enable writes, a specific pattern must be written to EECON2.


© 2010-2016 Microchip Technology Inc.

## 11.2 Using the Data EEPROM

The data EEPROM is a high-endurance, byte addressable array that has been optimized for the storage of frequently changing information (e.g., program variables or other data that are updated often). When variables in one section change frequently, while variables in another section do not change, it is possible to exceed the total number of write cycles to the EEPROM without exceeding the total number of write cycles to a single byte. Refer to **Section 30.0 "Electrical Specifications"**. If this is the case, then a refresh of the array must be performed. For this reason, variables that change infrequently (such as constants, IDs, calibration, etc.) should be stored in Flash program memory.

### 11.2.1 READING THE DATA EEPROM MEMORY

To read a data memory location, the user must write the address to the EEADRL register, clear the EEPGD and CFGS control bits of the EECON1 register, and then set control bit RD. The data is available at the very next cycle, in the EEDATL register; therefore, it can be read in the next instruction. EEDATL will hold this value until another read or until it is written to by the user (during a write operation).

**EXAMPLE 11-1: DATA EEPROM READ**

```
BANKSEL EEADRL       ;
MOVLW   DATA_EE_ADDR ;
MOVWF   EEADRL       ;Data Memory
                     ;Address to read
BCF     EECON1, CFGS ;Deselect Config space
BCF     EECON1, EEPGD;Point to DATA memory
BSF     EECON1, RD   ;EE Read
MOVF    EEDATL, W    ;W = EEDATL
```

**Note:** Data EEPROM can be read regardless of the setting of the $\overline{\text{CPD}}$ bit.

### 11.2.2 WRITING TO THE DATA EEPROM MEMORY

To write an EEPROM data location, the user must first write the address to the EEADRL register and the data to the EEDATL register. Then the user must follow a specific sequence to initiate the write for each byte.

The write will not initiate if the above sequence is not followed exactly (write 55h to EECON2, write AAh to EECON2, then set the WR bit) for each byte. Interrupts should be disabled during this code segment.

Additionally, the WREN bit in EECON1 must be set to enable write. This mechanism prevents accidental writes to data EEPROM due to errant (unexpected) code execution (i.e., lost programs). The user should keep the WREN bit clear at all times, except when updating EEPROM. The WREN bit is not cleared by hardware.

After a write sequence has been initiated, clearing the WREN bit will not affect this write cycle. The WR bit will be inhibited from being set unless the WREN bit is set.

At the completion of the write cycle, the WR bit is cleared in hardware and the EE Write Complete Interrupt Flag bit (EEIF) is set. The user can either enable this interrupt or poll this bit. EEIF must be cleared by software.

### 11.2.3 PROTECTION AGAINST SPURIOUS WRITE

There are conditions when the user may not want to write to the data EEPROM memory. To protect against spurious EEPROM writes, various mechanisms have been built-in. On power-up, WREN is cleared. Also, the Power-up Timer (64 ms duration) prevents EEPROM write.

The write initiate sequence and the WREN bit together help prevent an accidental write during:

- Brown-out
- Power Glitch
- Software Malfunction

### 11.2.4 DATA EEPROM OPERATION DURING CODE-PROTECT

Data memory can be code-protected by programming the $\overline{\text{CPD}}$ bit in the Configuration Words to '0'.

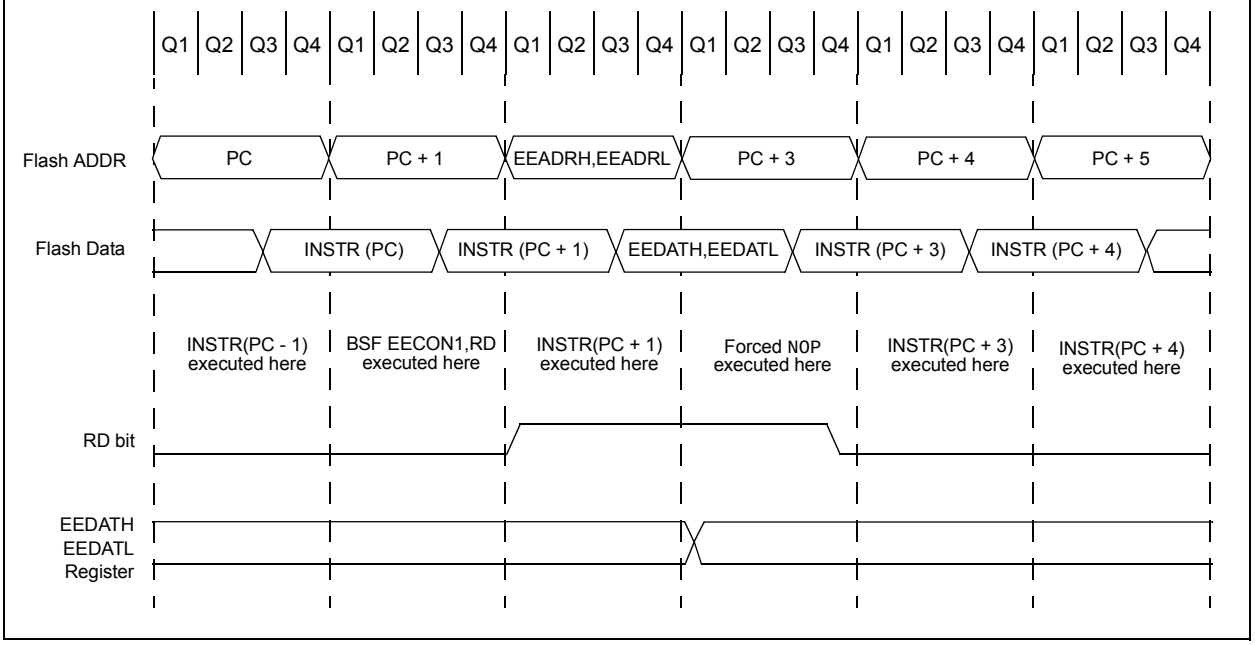
When the data memory is code-protected, only the CPU is able to read and write data to the data EEPROM. It is recommended to code-protect the program memory when code-protecting data memory. This prevents anyone from replacing your program with a program that will access the contents of the data EEPROM.

**EXAMPLE 11-2: DATA EEPROM WRITE**

```
        BANKSEL  EEADRL          ;
        MOVLW    DATA_EE_ADDR    ;
        MOVWF    EEADRL          ;Data Memory Address to write
        MOVLW    DATA_EE_DATA    ;
        MOVWF    EEDATL          ;Data Memory Value to write
        BCF      EECON1, CFGS    ;Deselect Configuration space
        BCF      EECON1, EEPGD   ;Point to DATA memory
        BSF      EECON1, WREN    ;Enable writes

        BCF      INTCON, GIE     ;Disable INTs.
Required
Sequence
        MOVLW    55h             ;
        MOVWF    EECON2          ;Write 55h
        MOVLW    0AAh            ;
        MOVWF    EECON2          ;Write AAh
        BSF      EECON1, WR      ;Set WR bit to begin write
        BSF      INTCON, GIE     ;Enable Interrupts
        BCF      EECON1, WREN    ;Disable writes
        BTFSC    EECON1, WR      ;Wait for write to complete
        GOTO     $-2             ;Done
```

**FIGURE 11-1: FLASH PROGRAM MEMORY READ CYCLE EXECUTION**





© 2010-2016 Microchip Technology Inc.

## 11.3 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 block size that can be erased by user software.

Flash program memory may only be written or erased if the destination address is in a segment of memory that is not write-protected, as defined in bits WRT<1:0> of Configuration Words.

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 EEDATH:EEDATL 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. |

The number of data write latches may not be equivalent to the number of row locations. During programming, user software may need to fill the set of write latches and initiate a programming operation multiple times in order to fully reprogram an erased row. For example, a device with a row size of 32 words and eight write latches will need to load the write latches with data and initiate a programming operation four times.

The size of a program memory row and the number of program memory write latches may vary by device. See Table 11-1 for details.

### 11.3.1 READING THE FLASH PROGRAM MEMORY

To read a program memory location, the user must:

1. Write the Least and Most Significant address bits to the EEADRH:EEADRL register pair.
2. Clear the CFGS bit of the EECON1 register.
3. Set the EEPGD control bit of the EECON1 register.
4. Then, set control bit RD of the EECON1 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 EECON1,RD`" instruction to be ignored. The data is available in the very next cycle, in the EEDATH:EEDATL register pair; therefore, it can be read as two bytes in the following instructions.

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

| | |
|---|---|
| **Note 1:** | The two instructions following a program memory read are required to be `NOP`s. This prevents the user from executing a two-cycle instruction on the next instruction after the RD bit is set. |
| **2:** | Flash program memory can be read regardless of the setting of the $\overline{CP}$ bit. |

**TABLE 11-1: FLASH MEMORY ORGANIZATION BY DEVICE**

| Device | Erase Block (Row) Size/Boundary | Number of Write Latches/Boundary |
|---|---|---|
| PIC16(L)F1946/47 | 32 words, EEADRL<4:0> = `00000` | 32 words, EEADRL<4:0> = `00000` |

**EXAMPLE 11-3: 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  EEADRL           ; Select Bank for EEPROM registers
    MOVLW    PROG_ADDR_LO     ;
    MOVWF    EEADRL           ; Store LSB of address
    MOVLW    PROG_ADDR_HI     ;
    MOVWL    EEADRH           ; Store MSB of address

    BCF      EECON1,CFGS      ; Do not select Configuration Space
    BSF      EECON1,EEPGD     ; Select Program Memory
    BCF      INTCON,GIE       ; Disable interrupts
    BSF      EECON1,RD        ; Initiate read
    NOP                       ; Executed (Figure 11-1)
    NOP                       ; Ignored (Figure 11-1)
    BSF      INTCON,GIE       ; Restore interrupts

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


© 2010-2016 Microchip Technology Inc.

### 11.3.2 ERASING FLASH PROGRAM MEMORY

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

1. Load the EEADRH:EEADRL register pair with the address of new row to be erased.
2. Clear the CFGS bit of the EECON1 register.
3. Set the EEPGD, FREE, and WREN bits of the EECON1 register.
4. Write 55h, then AAh, to EECON2 (Flash programming unlock sequence).
5. Set control bit WR of the EECON1 register to begin the erase operation.
6. Poll the FREE bit in the EECON1 register to determine when the row erase has completed.

See Example 11-4.

After the "BSF EECON1,WR" instruction, the processor requires two cycles to set up the erase operation. The user must place two NOP instructions after the WR bit is set. The processor will halt internal operations for the typical 2 ms erase time. This is not Sleep mode as the clocks and peripherals will continue to run. After the erase cycle, the processor will resume operation with the third instruction after the EECON1 write instruction.

### 11.3.3 WRITING TO FLASH PROGRAM MEMORY

Program memory is programmed using the following steps:

1. Load the starting address of the word(s) to be programmed.
2. Load the write latches 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 11-2 (block writes to program memory with 16 write latches) for more details. The write latches are aligned to the address boundary defined by EEADRL as shown in Table 11-1. Write operations do not cross these boundaries. At the completion of a program memory write operation, the write latches are reset to contain 0x3FFF.

The following steps should be completed to load the write latches and program a block of program memory. These steps are divided into two parts. First, all write latches are loaded with data except for the last program memory location. Then, the last write latch is loaded and the programming sequence is initiated. A special unlock sequence is required to load a write latch with data or initiate a Flash programming operation. This unlock sequence should not be interrupted.

1. Set the EEPGD and WREN bits of the EECON1 register.
2. Clear the CFGS bit of the EECON1 register.
3. Set the LWLO bit of the EECON1 register. When the LWLO bit of the EECON1 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 EEADRH:EEADRL register pair with the address of the location to be written.
5. Load the EEDATH:EEDATL register pair with the program memory data to be written.
6. Write 55h, then AAh, to EECON2, then set the WR bit of the EECON1 register (Flash programming unlock sequence). The write latch is now loaded.
7. Increment the EEADRH:EEADRL 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 EECON1 register. When the LWLO bit of the EECON1 register is '0', the write sequence will initiate the write to Flash program memory.
10. Load the EEDATH:EEDATL register pair with the program memory data to be written.
11. Write 55h, then AAh, to EECON2, then set the WR bit of the EECON1 register (Flash programming unlock sequence). The entire latch block is now written to Flash program memory.

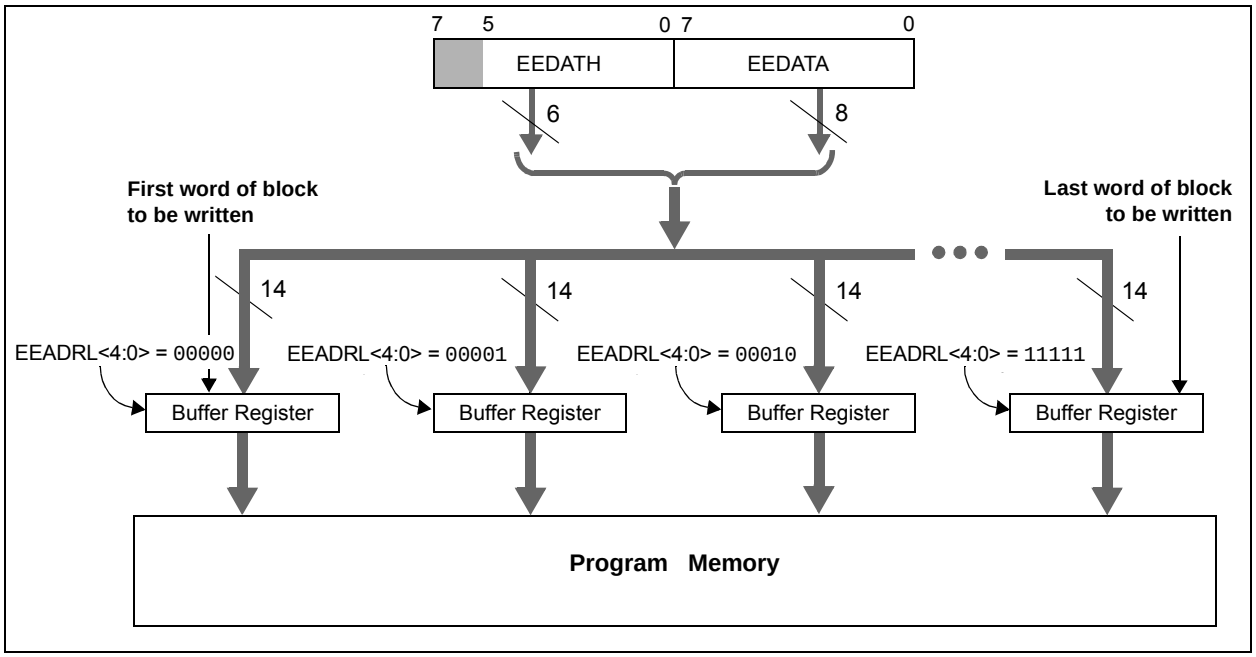
It is not necessary to load the entire write latch block with user program data. However, the entire write latch block will be written to program memory.

An example of the complete write sequence for eight words is shown in Example 11-5. The initial address is loaded into the EEADRH:EEADRL register pair; the eight words of data are loaded using indirect addressing.

| **Note:** | The code sequence provided in Example 11-5 must be repeated multiple times to fully program an erased program memory row. |
|---|---|

After the "BSF EECON1,WR" instruction, the processor requires two cycles to set up the write operation. The user must place two NOP instructions after the WR bit is set. The processor will halt internal operations for the typical 2 ms, only during the cycle in which the write takes place (i.e., the last word of the block write). This is not Sleep mode as the clocks and peripherals will continue to run. The processor does not stall when LWLO = 1, loading the write latches. After the write cycle, the processor will resume operation with the third instruction after the EECON1 write instruction.

**FIGURE 11-2: BLOCK WRITES TO FLASH PROGRAM MEMORY WITH 32 WRITE LATCHES**





© 2010-2016 Microchip Technology Inc.

**EXAMPLE 11-4: ERASING ONE ROW OF PROGRAM MEMORY -**

```
; This row erase routine assumes the following:
; 1. A valid address within the erase block 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     EEADRL
        MOVF        ADDRL,W         ; Load lower 8 bits of erase address boundary
        MOVWF       EEADRL
        MOVF        ADDRH,W         ; Load upper 6 bits of erase address boundary
        MOVWF       EEADRH
        BSF         EECON1,EEPGD    ; Point to program memory
        BCF         EECON1,CFGS     ; Not configuration space
        BSF         EECON1,FREE     ; Specify an erase operation
        BSF         EECON1,WREN     ; Enable writes

Required
Sequence
        MOVLW       55h             ; Start of required sequence to initiate erase
        MOVWF       EECON2          ; Write 55h
        MOVLW       0AAh            ;
        MOVWF       EECON2          ; Write AAh
        BSF         EECON1,WR       ; Set WR bit to begin erase
        NOP                         ; Any instructions here are ignored as processor
                                    ; halts to begin erase sequence
        NOP                         ; Processor will stop here and wait for erase complete.

                                    ; after erase processor continues with 3rd instruction

        BCF         EECON1,WREN     ; Disable writes
        BSF         INTCON,GIE      ; Enable interrupts
```

**EXAMPLE 11-5: WRITING TO FLASH PROGRAM MEMORY**

```
; This write routine assumes the following:
; 1. The 16 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 = 000) 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     EEADRH         ; Bank 3
        MOVF        ADDRH,W        ; Load initial address
        MOVWF       EEADRH         ;
        MOVF        ADDRL,W        ;
        MOVWF       EEADRL         ;
        MOVLW       LOW DATA_ADDR  ; Load initial data address
        MOVWF       FSR0L          ;
        MOVLW       HIGH DATA_ADDR ; Load initial data address
        MOVWF       FSR0H          ;
        BSF         EECON1,EEPGD   ; Point to program memory
        BCF         EECON1,CFGS    ; Not configuration space
        BSF         EECON1,WREN    ; Enable writes
        BSF         EECON1,LWLO    ; Only Load Write Latches
LOOP
        MOVIW       FSR0++         ; Load first data byte into lower
        MOVWF       EEDATL         ;
        MOVIW       FSR0++         ; Load second data byte into upper
        MOVWF       EEDATH         ;

        MOVF        EEADRL,W       ; Check if lower bits of address are '000'
        XORLW       0x07           ; Check if we're on the last of 8 addresses
        ANDLW       0x07           ;
        BTFSC       STATUS,Z       ; Exit if last of eight words,
        GOTO        START_WRITE    ;

  Required Sequence
        MOVLW       55h            ; Start of required write sequence:
        MOVWF       EECON2         ; Write 55h
        MOVLW       0AAh           ;
        MOVWF       EECON2         ; Write AAh
        BSF         EECON1,WR      ; Set WR bit to begin write
        NOP                        ; Any instructions here are ignored as processor
                                   ; halts to begin write sequence
        NOP                        ; Processor will stop here and wait for write to complete.

                                   ; After write processor continues with 3rd instruction.

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

START_WRITE
        BCF         EECON1,LWLO    ; No more loading latches - Actually start Flash program
                                   ; memory write

  Required Sequence
        MOVLW       55h            ; Start of required write sequence:
        MOVWF       EECON2         ; Write 55h
        MOVLW       0AAh           ;
        MOVWF       EECON2         ; Write AAh
        BSF         EECON1,WR      ; Set WR bit to begin write
        NOP                        ; Any instructions here are ignored as processor
                                   ; halts to begin write sequence
        NOP                        ; Processor will stop here and wait for write complete.

                                   ; after write processor continues with 3rd instruction
        BCF         EECON1,WREN    ; Disable writes
        BSF         INTCON,GIE     ; Enable interrupts
```


© 2010-2016 Microchip Technology Inc.

## 11.4 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.
8. Repeat steps 6 and 7 as many times as required to reprogram the erased row.

## 11.5 User ID, Device ID and Configuration Word Access

Instead of accessing program memory or EEPROM data memory, the User ID's, Device ID/Revision ID and Configuration Words can be accessed when CFGS = 1 in the EECON1 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 11-3.

When read access is initiated on an address outside the parameters listed in Table 11-3, the EEDATH:EEDATL register pair is cleared.

**TABLE 11-2: USER ID, DEVICE ID AND CONFIGURATION WORD ACCESS (CFGS = 1)**

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

**EXAMPLE 11-3: 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  EEADRL           ; Select correct Bank
    MOVLW    PROG_ADDR_LO     ;
    MOVWF    EEADRL           ; Store LSB of address
    CLRF     EEADRH           ; Clear MSB of address

    BSF      EECON1,CFGS      ; Select Configuration Space
    BCF      INTCON,GIE       ; Disable interrupts
    BSF      EECON1,RD        ; Initiate read
    NOP                       ; Executed (See Figure 11-1)
    NOP                       ; Ignored (See Figure 11-1)
    BSF      INTCON,GIE       ; Restore interrupts

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

### 11.6 Write Verify

Depending on the application, good programming practice may dictate that the value written to the data EEPROM or program memory should be verified (see Example 11-6) to the desired value to be written. Example 11-6 shows how to verify a write to EEPROM.

**EXAMPLE 11-6: EEPROM WRITE VERIFY**

```
BANKSEL EEDATL      ;
MOVF    EEDATL, W   ;EEDATL not changed
                    ;from previous write
BSF     EECON1, RD  ;YES, Read the
                    ;value written
XORWF   EEDATL, W   ;
BTFSS   STATUS, Z   ;Is data the same
GOTO    WRITE_ERR   ;No, handle error
:                   ;Yes, continue
```


© 2010-2016 Microchip Technology Inc.

## 11.7 Register Definitions: Data EEPROM Control

**REGISTER 11-1: EEDATL: EEPROM 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 |
|---|---|---|---|---|---|---|---|
| EEDAT<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 **EEDAT<7:0>**: Read/write value for EEPROM data byte or Least Significant bits of program memory

**REGISTER 11-2: EEDATH: EEPROM 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 |
|---|---|---|---|---|---|---|---|
| — | — | EEDAT<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 **EEDAT<13:8>**: Read/write value for Most Significant bits of program memory

**REGISTER 11-3: EEADRL: EEPROM 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 |
|---|---|---|---|---|---|---|---|
| EEADR<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 **EEADR<7:0>**: Specifies the Least Significant bits for program memory address or EEPROM address

**REGISTER 11-4: EEADRH: EEPROM 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) | EEADR<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 **EEADR<14:8>**: Specifies the Most Significant bits for program memory address or EEPROM address

**Note 1:** Unimplemented, read as ‘1’.


© 2010-2016 Microchip Technology Inc.

**REGISTER 11-5: EECON1: EEPROM CONTROL 1 REGISTER**

| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W/HC-0/0 | R/W-x/q | R/W-0/0 | R/S/HC-0/0 | R/S/HC-0/0 |
|---|---|---|---|---|---|---|---|
| EEPGD | 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 **EEPGD:** Flash Program/Data EEPROM Memory Select bit
- 1 = Accesses program space Flash memory
- 0 = Accesses data EEPROM memory

bit 6 **CFGS:** Flash Program/Data EEPROM or Configuration Select bit
- 1 = Accesses Configuration, User ID and Device ID Registers
- 0 = Accesses Flash Program or data EEPROM Memory

bit 5 **LWLO:** Load Write Latches Only bit

If CFGS = 1 (Configuration space) **OR** CFGS = 0 and EEPGD = 1 (program Flash):
- 1 = The next WR command does not initiate a write; only the program memory latches are updated.
- 0 = The next WR command writes a value from EEDATH:EEDATL into program memory latches and initiates a write of all the data stored in the program memory latches.

If CFGS = 0 and EEPGD = 0: (Accessing data EEPROM)
LWLO is ignored. The next WR command initiates a write to the data EEPROM.

bit 4 **FREE:** Program Flash Erase Enable bit

If CFGS = 1 (Configuration space) **OR** CFGS = 0 and EEPGD = 1 (program Flash):
- 1 = Performs an erase operation on the next WR command (cleared by hardware after completion of erase).
- 0 = Performs a write operation on the next WR command.

If EEPGD = 0 and CFGS = 0: (Accessing data EEPROM)
FREE is ignored. The next WR command will initiate both a erase cycle and a write cycle.

bit 3 **WRERR:** EEPROM 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 and data EEPROM

bit 1 **WR:** Write Control bit
- 1 = Initiates a program Flash or data EEPROM 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 or data EEPROM is complete and inactive.

bit 0 **RD:** Read Control bit
- 1 = Initiates an program Flash or data EEPROM 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 or data EEPROM data read.

**REGISTER 11-6: EECON2: EEPROM 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 |
|---|---|---|---|---|---|---|---|
| EEPROM 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 **Data EEPROM Unlock Pattern bits**

To unlock writes, a 55h must be written first, followed by an AAh, before setting the WR bit of the EECON1 register. The value written to this register is used to unlock the writes. There are specific timing requirements on these writes. Refer to **Section 11.2.2 "Writing to the Data EEPROM Memory"** for more information.

**TABLE 11-3: SUMMARY OF REGISTERS ASSOCIATED WITH DATA EEPROM**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| EECON1 | EEPGD | CFGS | LWLO | FREE | WRERR | WREN | WR | RD | 119 |
| EECON2 | EEPROM Control Register 2 (not a physical register) | | | | | | | | 106* |
| EEADRL | EEADRL<7:0> | | | | | | | | 118 |
| EEADRH | —(1) | EEADRH<6:0> | | | | | | | 118 |
| EEDATL | EEDATL<7:0> | | | | | | | | 117 |
| EEDATH | — | — | EEDATH<5:0> | | | | | | 117 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| PIE2 | OSFIE | C2IE | C1IE | EEIE | BCLIE | LCDIE | C3IE | CCP2IE | 92 |
| PIR2 | OSFIF | C2IF | C1IF | EEIF | BCLIF | LCDIF | C3IF | CCP2IF | 96 |

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

* Page provides register information.

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


© 2010-2016 Microchip Technology Inc.

## 12.0 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 12-1: PORT AVAILABILITY PER DEVICE**

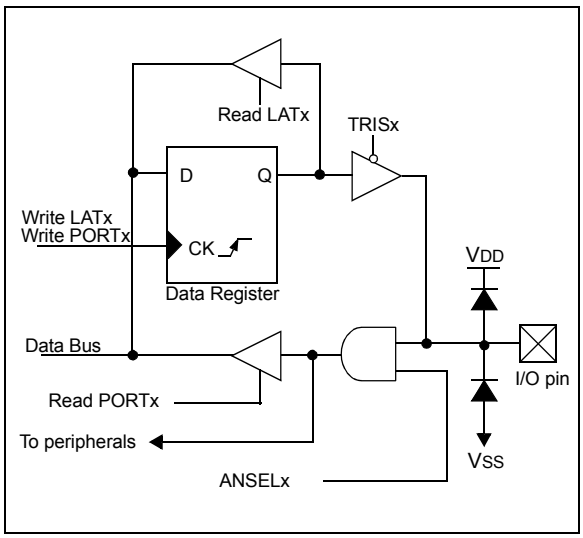
| Device | PORTA | PORTB | PORTC | PORTD | PORTE | PORTF | PORTG |
|---|---|---|---|---|---|---|---|
| PIC16F1946 | • | • | • | • | • | • | • |
| PIC16F1947 | • | • | • | • | • | • | • |

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 12-1.

**FIGURE 12-1: GENERIC I/O PORT OPERATION**




**EXAMPLE 12-1: INITIALIZING PORTA**

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

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

## 12.1 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 12-1. For this device family, the following functions can be moved between different pins.

- CCP3/P3C output
- CCP3/P3B output
- CCP2/P2D output
- CCP2/P2C output
- CCP2/P2B output
- CCP2/P2A output
- CCP1/P1C output
- CCP1/P1B output

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.


© 2010-2016 Microchip Technology Inc.

## 12.2 Register Definitions: Alternate Pin Function Control

**REGISTER 12-1: APFCON: ALTERNATE PIN FUNCTION 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/W-0/0 |
|---|---|---|---|---|---|---|---|
| P3CSEL | P3BSEL | P2DSEL | P2CSEL | P2BSEL | CCP2SEL | P1CSEL | P1BSEL |
| 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 **P3CSEL:** CCP3 PWM C Output Pin Selection bit
0 = P3C function is on RE3/P3C/COM0
1 = P3C function is on RD3/P3C/SEG3

bit 6 **P3BSEL:** CCP3 PWM B Output Pin Selection bit
0 = P3B function is on RE4/P3B/COM1
1 = P3B function is on RD4/P3B/SEG4

bit 5 **P2DSEL:** CCP2 PWM D Output Pin Selection bit
0 = P2D function is on RE0/P2D/VLCD1
1 = P2D function is on RD0/P2D/SEG0

bit 4 **P2CSEL:** CCP2 PWM C Output Pin Selection bit
0 = P2C function is on RE1/P2C/VLCD2
1 = P2C function is on RD1/P2C/SEG1

bit 3 **P2BSEL:** CCP2 PWM B Output Pin Selection bit
0 = P2B function is on RE2/P2B/VLCD3
1 = P2B function is on RD2/P2B/SEG2

bit 2 **CCP2SEL:** CCP2 Input/Output Pin Selection bit
0 = CCP2/P2A function is on RC1/CCP2/P2A/T1OSI/SEG32
1 = CCP2/P2A function is on RE7/CCP2/P2A/SEG31

bit 1 **P1CSEL:** CCP1 PWM C Output Pin Selection bit
0 = P1C function is on RE5/P1C/COM2
1 = P1C function is on RD5/P1C/SEG5

bit 0 **P1BSEL:** CCP1 PWM B Output Pin Selection bit
0 = P1B function is on RE6/P1B/COM3
1 = P1B function is on RD6/P1B/SEG6

## 12.3 PORTA Registers

PORTA is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISA (Register 12-3). 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). Example 12-1 shows how to initialize PORTA.

Reading the PORTA register (Register 12-2) 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).

The TRISA register (Register 12-3) 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'.

### 12.3.1 ANSELA REGISTER

The ANSELA register (Register 12-5) 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. |

### 12.3.2 PORTA FUNCTIONS AND OUTPUT PRIORITIES

Each PORTA pin is multiplexed with other functions. The pins, their combined functions and their output priorities are shown in Table 12-2.

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

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

**TABLE 12-2: PORTA OUTPUT PRIORITY**

| Pin Name | Function Priority(1) |
|---|---|
| RA0 | SEG33 (LCD)<br>RA0 |
| RA1 | SEG18<br>RA1 |
| RA2 | SEG34 (LCD)<br>RA2 |
| RA3 | SEG35 (LCD)<br>RA3 |
| RA4 | SEG14 (LCD)<br>RA4 |
| RA5 | SEG15 (LCD)<br>RA5 |
| RA6 | OSC2 (enabled by Configuration Word)<br>CLKOUT (enabled by Configuration Word)<br>SEG36 (LCD)<br>RA6 |
| RA7 | OSC1/CLKIN (enabled by Configuration Word)<br>SEG37 (LCD)<br>RA7 |

**Note 1:** Priority listed from highest to lowest.


© 2010-2016 Microchip Technology Inc.

## 12.4 Register Definitions: PORTA

**REGISTER 12-2: PORTA: PORTA 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 |
|---|---|---|---|---|---|---|---|
| RA7 | RA6 | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 |
| bit 7 | | | | | | | bit 0 |

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

bit 7-0 **RA<7:0>**: PORTA I/O Value bits(1)
1 = Port pin is ≥ VIH
0 = Port pin is ≤ VIL

**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 12-3: TRISA: PORTA 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 |
|---|---|---|---|---|---|---|---|
| TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 |
| bit 7 | | | | | | | bit 0 |

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

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

**REGISTER 12-4: LATA: PORTA DATA LATCH REGISTER**

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

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

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

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

**REGISTER 12-5: ANSELA: PORTA ANALOG SELECT REGISTER**

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

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

bit 7-6 **Unimplemented:** Read as '0'

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

bit 4 **Unimplemented:** Read as '0'

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

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

**TABLE 12-3: 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 |
|---|---|---|---|---|---|---|---|---|---|
| ADCON0 | — | CHS<4:0> | | | | | GO/$\overline{\text{DONE}}$ | ADON | 161 |
| ADCON1 | ADFM | ADCS<2:0> | | | — | — | ADPREF<1:0> | | 162 |
| ANSELA | — | — | ANSA5 | — | ANSA3 | ANSA2 | ANSA1 | ANSA0 | 126 |
| CPSCON0 | CPSON | CPSRM | — | — | CPSRNG1 | CPSRNG0 | CPSOUT | T0XCS | 322 |
| CPSCON1 | — | — | — | CPSCH<4:0> | | | | | 323 |
| DACCON0 | DACEN | DACLPS | DACOE | --- | DACPSS<1:0> | | --- | DACNSS | 171 |
| LATA | LATA7 | LATA6 | LATA5 | LATA4 | LATA3 | LATA2 | LATA1 | LATA0 | 125 |
| LCDSE1 | SE15 | SE14 | SE13 | SE12 | SE11 | SE10 | SE9 | SE8 | 330 |
| LCDSE2 | SE23 | SE22 | SE21 | SE20 | SE19 | SE18 | SE17 | SE16 | 330 |
| LCDSE4 | SE39 | SE38 | SE37 | SE36 | SE35 | SE34 | SE33 | SE32 | 330 |
| OPTION_REG | $\overline{\text{WPUEN}}$ | INTEDG | TMR0CS | TMR0SE | PSA | PS<2:0> | | | 188 |
| PORTA | RA7 | RA6 | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 | 125 |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 125 |

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

**TABLE 12-4: 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 | — | — | FCMEN | IESO | $\overline{\text{CLKOUTEN}}$ | BOREN<1:0> | | $\overline{\text{CPD}}$ | 54 |
| | 7:0 | $\overline{\text{CP}}$ | MCLRE | $\overline{\text{PWRTE}}$ | WDTE<1:0> | | FOSC<2:0> | | | |

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


© 2010-2016 Microchip Technology Inc.

## 12.5 PORTB Registers

PORTB is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISB (Register 12-7). Setting a TRISB bit (= 1) will make the corresponding PORTB pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISB bit (= 0) will make the corresponding PORTB pin an output (i.e., enable the output driver and put the contents of the output latch on the selected pin). Example 12-1 shows how to initialize an I/O port.

Reading the PORTB register (Register 12-6) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch (LATB).

The TRISB register (Register 12-7) 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 inputs always read '0'.

### 12.5.1 WEAK PULL-UPS

Each of the PORTB pins has an individually configurable internal weak pull-up. Control bits WPUB<7:0> enable or disable each pull-up (see Register 12-9). Each weak pull-up is automatically turned off when the port pin is configured as an output. All pull-ups are disabled on a Power-on Reset by the $\overline{\text{WPUEN}}$ bit of the OPTION_REG register.

### 12.5.2 INTERRUPT-ON-CHANGE

All of the PORTB pins are individually configurable as an interrupt-on-change pin. Control bits IOCB<7:0> enable or disable the interrupt function for each pin. The interrupt-on-change feature is disabled on a Power-on Reset. Reference **Section 13.0 "Interrupt-On-Change"** for more information.

### 12.5.3 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 12-5.

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

Analog input and some digital input functions are not included in the list below. These input functions can remain active when the pin is configured as an output. Certain digital input functions, such as the EUSART RX signal, override other port functions and are included in the priority list.

**TABLE 12-5: PORTB OUTPUT PRIORITY**

| Pin Name | Function Priority(1) |
|---|---|
| RB0 | SEG30 (LED)<br>SRI (SR Latch)<br>RB0 |
| RB1 | SEG8 (LCD)<br>RB1 |
| RB2 | SEG9 (LCD)<br>RB2 |
| RB3 | SEG10 (LCD)<br>RB3 |
| RB4 | SEG11 (LCD)<br>RB4 |
| RB5 | SEG29 (LCD)<br>RB5 |
| RB6 | ICSPCLK (Programming)<br>ICDCLK (enabled by Configuration Word)<br>SEG38 (LCD)<br>RB6 |
| RB7 | ICSPDAT (Programming)<br>ICDDAT (enabled by Configuration Word)<br>SEG39 (LCD)<br>RB7 |

**Note 1:** Priority listed from highest to lowest.

## 12.6 Register Definitions: PORTB

**REGISTER 12-6: PORTB: PORTB REGISTER**

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

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

bit 7-0 **RB<7:0>**: PORTB I/O Pin bit
1 = Port pin is $\geq V_{IH}$
0 = Port pin is $\leq V_{IL}$

**REGISTER 12-7: TRISB: PORTB TRI-STATE REGISTER**

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

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

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

**REGISTER 12-8: LATB: PORTB DATA LATCH REGISTER**

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

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

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

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


© 2010-2016 Microchip Technology Inc.

**REGISTER 12-9: WPUB: WEAK PULL-UP PORTB REGISTER**

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

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

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

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

**TABLE 12-6: 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 |
|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| IOCBP | IOCBP7 | IOCBP6 | IOCBP5 | IOCBP4 | IOCBP3 | IOCBP2 | IOCBP1 | IOCBP0 | 148 |
| IOCBN | IOCBN7 | IOCBN6 | IOCBN5 | IOCBN4 | IOCBN3 | IOCBN2 | IOCBN1 | IOCBN0 | 148 |
| IOCBF | IOCBF7 | IOCBF6 | IOCBF5 | IOCBF4 | IOCBF3 | IOCBF2 | IOCBF1 | IOCBF0 | 148 |
| LATB | LATB7 | LATB6 | LATB5 | LATB4 | LATB3 | LATB2 | LATB1 | LATB0 | 128 |
| LCDSE1 | SE15 | SE14 | SE13 | SE12 | SE11 | SE10 | SE9 | SE8 | 330 |
| LCDSE3 | SE31 | SE30 | SE29 | SE28 | SE27 | SE26 | SE25 | SE24 | 330 |
| LCDSE4 | SE39 | SE38 | SE37 | SE36 | SE35 | SE34 | SE33 | SE32 | 330 |
| OPTION_REG | $\overline{\text{WPUEN}}$ | INTEDG | TMR0CS | TMR0SE | PSA | PS<2:0> | | | 188 |
| PORTB | RB7 | RB6 | RB5 | RB4 | RB3 | RB2 | RB1 | RB0 | 128 |
| T1GCON | TMR1GE | T1GPOL | T1GTM | T1GSPM | T1GGO/$\overline{\text{DONE}}$ | T1GVAL | T1GSS<1:0> | | 198 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 128 |
| WPUB | WPUB7 | WPUB6 | WPUB5 | WPUB4 | WPUB3 | WPUB2 | WPUB1 | WPUB0 | 129 |

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

## 12.7 PORTC Registers

PORTC is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISC (Register 12-11). Setting a TRISC bit (= 1) will make the corresponding PORTC pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISC bit (= 0) will make the corresponding PORTC pin an output (i.e., enable the output driver and put the contents of the output latch on the selected pin). Example 12-1 shows how to initialize an I/O port.

Reading the PORTC register (Register 12-10) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch (LATC).

The TRISC register (Register 12-11) 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 inputs always read '0'.

### 12.7.1 PORTC FUNCTIONS AND OUTPUT PRIORITIES

Each PORTC pin is multiplexed with other functions. The pins, their combined functions and their output priorities are shown in Table 12-7.

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

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

**TABLE 12-7: PORTC OUTPUT PRIORITY**

| Pin Name | Function Priority(1) |
|---|---|
| RC0 | T1OSO (Timer1 Oscillator)<br>SEG40 (ICD)<br>RC0 |
| RC1 | T1OSI (Timer1 Oscillator)<br>CCP2(2)/P2A(2)<br>SEG32 (ICD)<br>RC1 |
| RC2 | SEG13 (LCD)<br>CCP1/P1A<br>RC2 |
| RC3 | SEG17 (LCD)<br>SCL1 (MSSP1)<br>SCK1 (MSSP1)<br>RC3 |
| RC4 | SEG16 (LCD)<br>SDA1 (MSSP1)<br>RC4 |
| RC5 | SEG12 (LCD)<br>SDO1 (MSSP1)<br>RC5 |
| RC6 | SEG27 (LCD)<br>TX1 (EUSART1)<br>CK1 (EUSART1)<br>RC6 |
| RC7 | SEG28 (LCD)<br>DT1 (EUSART1)<br>RC7 |

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


© 2010-2016 Microchip Technology Inc.

## 12.8 Register Definitions: PORTC

**REGISTER 12-10: PORTC: PORTC REGISTER**

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

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

bit 7-0 **RC<7:0>**: PORTC General Purpose I/O Pin bits
1 = Port pin is ≥ VIH
0 = Port pin is ≤ VIL

**REGISTER 12-11: TRISC: PORTC TRI-STATE REGISTER**

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

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

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

**REGISTER 12-12: LATC: PORTC DATA LATCH REGISTER**

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

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

bit 7-0 **LATC<7:0>**: PORTC Output Latch Value bits(1)

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

**TABLE 12-8: SUMMARY OF REGISTERS ASSOCIATED WITH PORTC**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| APFCON | P3CSEL | P3BSEL | P2DSEL | P2CSEL | P2BSEL | CCP2SEL | P1CSEL | P1BSEL | 123 |
| LATC | LATC7 | LATC6 | LATC5 | LATC4 | LATC3 | LATC2 | LATC1 | LATC0 | 131 |
| LCDSE1 | SE15 | SE14 | SE13 | SE12 | SE11 | SE10 | SE9 | SE8 | 330 |
| LCDSE2 | SE23 | SE22 | SE21 | SE20 | SE19 | SE18 | SE17 | SE16 | 330 |
| LCDSE3 | SE31 | SE30 | SE29 | SE28 | SE27 | SE26 | SE25 | SE24 | 330 |
| LCDSE4 | SE39 | SE38 | SE37 | SE36 | SE35 | SE34 | SE33 | SE32 | 330 |
| LCDSE5 | — | — | SE45 | SE44 | SE43 | SE42 | SE41 | SE40 | 330 |
| PORTC | RC7 | RC6 | RC5 | RC4 | RC3 | RC2 | RC1 | RC0 | 131 |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 298 |
| RC2STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 298 |
| SSP1CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM<3:0> | | | | 282 |
| SSP2STAT | SMP | CKE | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF | 281 |
| T1CON | TMR1CS<1:0> | | T1CKPS<1:0> | | T1OSCEN | $\overline{T1SYNC}$ | — | TMR1ON | 197 |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 297 |
| TX2STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 297 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 131 |

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


© 2010-2016 Microchip Technology Inc.

## 12.9 PORTD Registers

PORTD is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISB (Register 12-13). Setting a TRISD bit (= 1) will make the corresponding PORTD pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISD bit (= 0) will make the corresponding PORTD pin an output (i.e., enable the output driver and put the contents of the output latch on the selected pin). Example 12-1 shows how to initialize an I/O port.

Reading the PORTD register (Register 12-13) 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 (LATD).

The TRISD register (Register 12-14) controls the PORTD pin output drivers, even when they are being used as analog inputs. The user should ensure the bits in the TRISD register are maintained set when using them as analog inputs. I/O pins configured as analog inputs always read '0'.

### 12.9.1 PORTD FUNCTIONS AND OUTPUT PRIORITIES

Each PORTD pin is multiplexed with other functions. The pins, their combined functions and their output priorities are shown in Table 12-9.

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

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

**TABLE 12-9: PORTD OUTPUT PRIORITY**

| Pin Name | Function Priority(1) |
|---|---|
| RD0 | SEG0 (LCD)<br>P2D(2) (CCP)<br>RD0 |
| RD1 | SEG1 (LCD)<br>P2C(2) (CCP)<br>RD1 |
| RD2 | P2B(2) (CCP)<br>SEG2 (LCD)<br>RD2 |
| RD3 | SEG3 (LCD)<br>P3C(2) (CCP)<br>RD3 |
| RD4 | SEG4 (LCD)<br>P3B(2) (CCP)<br>SDO2 (SSP2)<br>RD4 |
| RD5 | SEG5 (LCD)<br>P1C(2) (CCP)<br>SDA2 (SSP2)<br>RD5 |
| RD6 | SEG6 (LED)<br>P1B(2) (CCP)<br>SCK2/SCL2 (SSP2)<br>RD6 |
| RD7 | SEG7 (LCD)<br>RD7 |

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

## 12.10 Register Definitions: PORTD

**REGISTER 12-13: PORTD: PORTD 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 |
|---|---|---|---|---|---|---|---|
| RD7 | RD6 | RD5 | RD4 | RD3 | RD2 | RD1 | RD0 |
| 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 **RD<7:0>**: PORTD General Purpose I/O Pin bits
1 = Port pin is $\geq V_{IH}$
0 = Port pin is $\leq V_{IL}$

**REGISTER 12-14: TRISD: PORTD 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 |
|---|---|---|---|---|---|---|---|
| TRISD7 | TRISD6 | TRISD5 | TRISD4 | TRISD3 | TRISD2 | TRISD1 | TRISD0 |
| 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 **TRISD<7:0>:** PORTD Tri-State Control bits
1 = PORTD pin configured as an input (tri-stated)
0 = PORTD pin configured as an output

**REGISTER 12-15: LATD: PORTD 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 |
|---|---|---|---|---|---|---|---|
| LATD7 | LATD6 | LATD5 | LATD4 | LATD3 | LATD2 | LATD1 | LATD0 |
| 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 **LATD<7:0>**: PORTD Output Latch Value bits**(1)**

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


© 2010-2016 Microchip Technology Inc.

**TABLE 12-10: SUMMARY OF REGISTERS ASSOCIATED WITH PORTD**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| APFCON | P3CSEL | P3BSEL | P2DSEL | P2CSEL | P2BSEL | CCP2SEL | P1CSEL | P1BSEL | 123 |
| CCPxCON | PxM<1:0>(1) | | DCxB<1:0> | | CCPxM<3:0> | | | | 227 |
| LATD | LATD7 | LATD6 | LATD5 | LATD4 | LATD3 | LATD2 | LATD1 | LATD0 | 134 |
| LCDCON | LCDEN | SLPEN | WERR | — | CS<1:0> | | LMUX<1:0> | | 326 |
| LCDSE0 | SE7 | SE6 | SE5 | SE4 | SE3 | SE2 | SE1 | SE0 | 330 |
| PORTD | RD7 | RD6 | RD5 | RD4 | RD3 | RD2 | RD1 | RD0 | 134 |
| TRISD | TRISD7 | TRISD6 | TRISD5 | TRISD4 | TRISD3 | TRISD2 | TRISD1 | TRISD0 | 134 |

**Legend:** x = unknown, u = unchanged, – = unimplemented locations read as '0'. Shaded cells are not used by PORTD.
**Note 1:** Applies to ECCP modules only.

## 12.11 PORTE Registers

PORTE is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISE. Setting a TRISE bit (= 1) will make the corresponding PORTE pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISE bit (= 0) will make the corresponding PORTE pin an output (i.e., enable the output driver and put the contents of the output latch on the selected pin). Example 12-1 shows how to initialize an I/O port.

Reading the PORTE register (Register 12-16) 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 (LATE).

### 12.11.1 ANSELE REGISTER

The ANSELE register (Register 12-19) is used to configure the Input mode of an I/O pin to analog. Setting the appropriate ANSELE 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 ANSELE 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.

The TRISE register (Register 12-17) controls the PORTE pin output drivers, even when they are being used as analog inputs. The user should ensure the bits in the TRISE register are maintained set when using them as analog inputs. I/O pins configured as analog inputs always read '0'.

| | |
|---|---|
| **Note:** | The ANSELE register must be initialized to configure an analog channel as a digital input. Pins configured as analog inputs will read '0'. |

### 12.11.2 PORTE FUNCTIONS AND OUTPUT PRIORITIES

Each PORTE pin is multiplexed with other functions. The pins, their combined functions and their output priorities are shown in Table 12-11.

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

Each PORTE pin is multiplexed with other functions. The pins, their combined functions and their output priorities are briefly described here. For additional information, refer to the appropriate section in this data sheet.

When multiple outputs are enabled, the actual pin control goes to the peripheral with the lowest number in the following lists.

Analog input and some digital input functions are not included in the list below. These input functions can remain active when the pin is configured as an output. Certain digital input functions, such as the EUSART RX signal, override other port functions and are included in the priority list.

**TABLE 12-11: PORTE OUTPUT PRIORITY**

| Pin Name | Function Priority(1) |
|---|---|
| RE0 | P2D(2) (CCP)<br>RE0 |
| RE1 | P2C(2) (CCP)<br>RE1 |
| RE2 | P2B(2) (CCP)<br>RE2 |
| RE3 | P3C(2) (CCP)<br>COM0 (LCD)<br>RE3 |
| RE4 | P3B(2) (CCP)<br>COM1 (LCD)<br>RE4 |
| RE5 | P1C(2) (CCP)<br>COM2 (LCD)<br>RE5 |
| RE6 | P1B(2) (CCP)<br>COM3 (LCD)<br>RE6 |
| RE7 | CCP2(3)/P2A(3) (CCP)<br>SEG31 (LCD)<br>RE7 |

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


© 2010-2016 Microchip Technology Inc.

## 12.12 Register Definitions: PORTE

**REGISTER 12-16: PORTE: PORTE 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 |
|---|---|---|---|---|---|---|---|
| RE7 | RE6 | RE5 | RE4 | RE3 | RE2 | RE1 | RE0 |
| 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 **RE<7:0>**: PORTE I/O Pin bits
1 = Port pin is ≥ VIH
0 = Port pin is ≤ VIL

**REGISTER 12-17: TRISE: PORTE TRI-STATE REGISTER**

| R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| TRISE7 | TRISE6 | TRISE5 | TRISE4 | TRISE3 | TRISE2 | TRISE1 | TRISE0 |
| 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 **TRISE<7:0>:** RE<7:0> Tri-State Control bits
1 = PORTE pin configured as an input (tri-stated)
0 = PORTE pin configured as an output

**REGISTER 12-18: LATE: PORTE 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 |
|---|---|---|---|---|---|---|---|
| LATE7 | LATE6 | LATE5 | LATE4 | LATE3 | LATE2 | LATE1 | LATE0 |
| 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 **LATE<7:0>**: PORTE Output Latch Value bits(1)

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

**REGISTER 12-19: ANSELE: PORTE ANALOG SELECT REGISTER**

| R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 | R/W-1 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | ANSE2 | ANSE1 | ANSE0 |
| 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 **ANSE<7:0>**: Analog Select between Analog or Digital Function on Pins RE<7:0>, respectively
0 = Digital I/O. Pin is assigned to port or digital special function.
1 = Analog input. Pin is assigned as analog input(1). Digital input buffer disabled.

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

**TABLE 12-12: SUMMARY OF REGISTERS ASSOCIATED WITH PORTE**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| APFCON | P3CSEL | P3BSEL | P2DSEL | P2CSEL | P2BSEL | CCP2SEL | P1CSEL | P1BSEL | 123 |
| ANSELE | — | — | — | — | — | ANSE2 | ANSE1 | ANSE0 | 138 |
| CCPxCON | PxM<1:0>(1) | | DCxB<1:0> | | CCPxM<3:0> | | | | 227 |
| LATE | LATE7 | LATE6 | LATE5 | LATE4 | LATE3 | LATE2 | LATE1 | LATE0 | 138 |
| LCDCON | LCDEN | SLPEN | WERR | — | CS<1:0> | | LMUX<1:0> | | 326 |
| LCDREF | LCDIRE | LCDIRS | LCDIRI | — | VLCD3PE | VLCD2PE | VLCD1PE | — | 328 |
| LCDSE2 | SE31 | SE30 | SE29 | SE28 | SE27 | SE26 | SE25 | SE24 | 330 |
| PORTE | RE7 | RE6 | RE5 | RE4 | RE3 | RE2 | RE1 | RE0 | 137 |
| TRISE | TRISE7 | TRISE6 | TRISE5 | TRISE4 | TRISE3 | TRISE2 | TRISE1 | TRISE0 | 137 |

**Legend:** x = unknown, u = unchanged, – = unimplemented locations read as '0'. Shaded cells are not used by PORTE.
**Note 1:** Applies to ECCP modules only.


© 2010-2016 Microchip Technology Inc.

## 12.13 PORTF Registers

PORTF is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISF (Register 12-21). Setting a TRISF bit (= 1) will make the corresponding PORTF pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISF bit (= 0) will make the corresponding PORTF pin an output (i.e., enable the output driver and put the contents of the output latch on the selected pin). Example 12-1 shows how to initialize an I/O port.

Reading the PORTF register (Register 12-13) 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 (LATF).

The TRISF register (Register 12-14) controls the PORTF pin output drivers, even when they are being used as analog inputs. The user should ensure the bits in the TRISF register are maintained set when using them as analog inputs. I/O pins configured as analog inputs always read '0'.

### 12.13.1 ANSELF REGISTER

The ANSELF register (Register 12-23) is used to configure the Input mode of an I/O pin to analog. Setting the appropriate ANSELF 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 ANSELF 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 ANSELF register must be initialized to configure an analog channel as a digital input. Pins configured as analog inputs will read '0'. |

### 12.13.2 PORTF FUNCTIONS AND OUTPUT PRIORITIES

Each PORTF pin is multiplexed with other functions. The pins, their combined functions and their output priorities are shown in Table 12-13.

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

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

**TABLE 12-13: PORTF OUTPUT PRIORITY**

| Pin Name | Function Priority(1) |
|---|---|
| RF0 | SEG41 (LCD)<br>RF0 |
| RF1 | C2OUT (Comparator)<br>SRNQ (SR Latch)<br>SEG19 (LCD)<br>RF1 |
| RF2 | C1OUT (Comparator)<br>SEG20 (LCD)<br>SRQ (SR Latch)<br>RF2 |
| RF3 | SEG21 (LCD)<br>RF3 |
| RF4 | SEG22 (LCD)<br>RF4 |
| RF5 | DACOUT (DAC)<br>SEG23 (LCD)<br>RF5 |
| RF6 | SEG24 (LCD)<br>RF6 |
| RF7 | SEG25 (LCD)<br>RF7 |

**Note 1:** Priority listed from highest to lowest.

## 12.14 Register Definitions: PORTF

**REGISTER 12-20: PORTF: PORTF 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 |
|---|---|---|---|---|---|---|---|
| RF7 | RF6 | RF5 | RF4 | RF3 | RF2 | RF1 | RF0 |
| 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 **RF<7:0>**: PORTF General Purpose I/O Pin bits
1 = Port pin is $\geq V_{IH}$
0 = Port pin is $\leq V_{IL}$

**REGISTER 12-21: TRISF: PORTF 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 |
|---|---|---|---|---|---|---|---|
| TRISF7 | TRISF6 | TRISF5 | TRISF4 | TRISF3 | TRISF2 | TRISF1 | TRISF0 |
| 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 **TRISF<7:0>:** PORTF Tri-State Control bits
1 = PORTF pin configured as an input (tri-stated)
0 = PORTF pin configured as an output

**REGISTER 12-22: LATF: PORTF 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 |
|---|---|---|---|---|---|---|---|
| LATF7 | LATF6 | LATF5 | LATF4 | LATF3 | LATF2 | LATF1 | LATF0 |
| 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 **LATF<7:0>**: PORTF Output Latch Value bits**(1)**

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


© 2010-2016 Microchip Technology Inc.

**REGISTER 12-23: ANSELF: PORTF ANALOG SELECT 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 |
|---|---|---|---|---|---|---|---|
| ANSF7 | ANSF6 | ANSF5 | ANSDF4 | ANSF3 | ANSF2 | ANSDF1 | ANSF0 |
| 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 **ANSF<7:0>**: Analog Select between Analog or Digital Function on Pins RF<7:0>, respectively
0 = Digital I/O. Pin is assigned to port or digital special function.
1 = Analog input. Pin is assigned as analog input(1). Digital input buffer disabled.

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

**TABLE 12-14: SUMMARY OF REGISTERS ASSOCIATED WITH PORTF**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| ADCON0 | — | CHS<4:0> | | | | | GO/$\overline{\text{DONE}}$ | ADON | 161 |
| ANSELF | ANSF7 | ANSF6 | ANSF5 | ANSF4 | ANSF3 | ANSF2 | ANSF1 | ANSF0 | 141 |
| CCPxCON | PxM<1:0>(1) | | DCxB<1:0> | | CCPxM<3:0> | | | | 227 |
| CMOUT | — | — | — | — | — | MC3OUT | MC2OUT | MC1OUT | 179 |
| CM1CON1 | C1INTP | C1INTN | C1PCH1 | C1PCH0 | — | — | C1NCH<1:0> | | 179 |
| CM2CON1 | C2INTP | C2INTN | C2PCH1 | C2PCH0 | — | — | C2NCH<1:0> | | 179 |
| CPSCON0 | CPSON | CPSRM | — | — | CPSRNG<1:0> | | CPSOUT | T0XCS | 322 |
| CPSCON1 | — | — | — | — | CPSCH<3:0> | | | | 323 |
| DACCON0 | DACEN | DACLPS | DACOE | — | DACPSS<1:0> | | — | DACNSS | 171 |
| LATF | LATF7 | LATF6 | LATF5 | LATF4 | LATF3 | LATF2 | LATF1 | LATF0 | 134 |
| LCDCON | LCDEN | SLPEN | WERR | — | CS<1:0> | | LMUX<1:0> | | 326 |
| LCDSE2 | SE23 | SE22 | SE21 | SE20 | SE19 | SE18 | SE17 | SE16 | 330 |
| LCDSE3 | SE31 | SE30 | SE29 | SE28 | SE27 | SE26 | SE25 | SE24 | 330 |
| LCDSE5 | — | — | SE45 | SE44 | SE43 | SE42 | SE41 | SE40 | 330 |
| PORTF | RF7 | RF6 | RF5 | RF4 | RF3 | RF2 | RF1 | RF0 | 140 |
| SRCON0 | SRLEN | SRCLK2 | SRCLK1 | SRCLK0 | SRQEN | SRNQEN | SRPS | SRPR | 184 |
| TRISF | TRISF7 | TRISF6 | TRISF5 | TRISF4 | TRISF3 | TRISF2 | TRISF1 | TRISF0 | 140 |

**Legend:** x = unknown, u = unchanged, – = unimplemented locations read as '0'. Shaded cells are not used by PORTF.
**Note 1:** Applies to ECCP modules only.

**TABLE 12-15: SUMMARY OF CONFIGURATION WORD ASSOCIATED WITH PORTF**

| 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 |
|---|---|---|---|---|---|---|---|---|---|---|
| CONFIG2 | 13:8 | — | — | LVP | $\overline{\text{DEBUG}}$ | — | BORV | STVREN | PLLEN | 56 |
| | 7:0 | — | — | — | VCAPEN | — | — | WRT<1:0> | | |

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

## 12.15 PORTG Registers

PORTG is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISG (Register 12-25). Setting a TRISG bit (= 1) will make the corresponding PORTG pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISG bit (= 0) will make the corresponding PORTG pin an output (i.e., enable the output driver and put the contents of the output latch on the selected pin). The exception is RG5, which is input only and its TRIS bit will always read as '1'. Example 12-1 shows how to initialize an I/O port.

Reading the PORTG register (Register 12-24) 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 (LATG). RG5 reads '0' when MCLRE = 1.

The TRISG register (Register 12-25) controls the PORTG pin output drivers, even when they are being used as analog inputs. The user should ensure the bits in the TRISG register are maintained set when using them as analog inputs. I/O pins configured as analog inputs always read '0'.

### 12.15.1 ANSELG REGISTER

The ANSELG register (Register 12-27) is used to configure the Input mode of an I/O pin to analog. Setting the appropriate ANSELG 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 ANSELG 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 ANSELG register must be initialized to configure an analog channel as a digital input. Pins configured as analog inputs will read '0'. |

### 12.15.2 PORTG FUNCTIONS AND OUTPUT PRIORITIES

Each PORTG pin is multiplexed with other functions. The pins, their combined functions and their output priorities are shown in Table 12-16.

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

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

**TABLE 12-16: PORTG OUTPUT PRIORITY**

| Pin Name | Function Priority(1) |
|---|---|
| RG0 | CCP3 (CCP)<br>P3A (CCP)<br>SEG42 (LCD)<br>RG0 |
| RG1 | TX2 (EUSART)<br>CK2 (EUSART)<br>C3OUT (Comparator)<br>SEG43 (LCD)<br>RG1 |
| RG2 | DT2<br>SEG44 (LCD)<br>RG2 |
| RG3 | CCP4 (CCP)<br>P3D (CCP)<br>SEG45 (LCD)<br>RG3 |
| RG4 | CCP5 (CCP)<br>P1D (CCP)<br>SEG26 (LCD)<br>RG4 |
| RG5 | Input-only pin |

**Note 1:** Priority listed from highest to lowest.


© 2010-2016 Microchip Technology Inc.

## 12.16 Register Definitions: PORTG

### REGISTER 12-24: PORTG: PORTG 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 |
|---|---|---|---|---|---|---|---|
| — | — | RG5 | RG4 | RG3 | RG2 | RG1 | RG0 |
| 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 **RG<5:0>**: PORTG General Purpose I/O Pin bits
1 = Port pin is ≥ $V_{IH}$
0 = Port pin is ≤ $V_{IL}$

### REGISTER 12-25: TRISG: PORTG TRI-STATE REGISTER

| U-0 | U-0 | R-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 |
|---|---|---|---|---|---|---|---|
| — | — | TRISG5 | TRISG4 | TRISG3 | TRISG2 | TRISG1 | TRISG0 |
| 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 **TRISG5:** PORTG Tri-State Control bit
This bit (RG5 pin) is an input only and always read as ‘1’.

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

### REGISTER 12-26: LATG: PORTG DATA LATCH 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 |
|---|---|---|---|---|---|---|---|
| — | — | LATG5 | LATG4 | LATG3 | LATG2 | LATG1 | LATG0 |
| 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 **LATG<5:0>**: PORTG Output Latch Value bits

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

**REGISTER 12-27: ANSELG: PORTG ANALOG SELECT REGISTER**

| U-0 | U-0 | U-0 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | ANSG4 | ANSG3 | ANSG2 | ANSG1 | — |
| 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 4-1 **ANSG<4:1>**: Analog Select between Analog or Digital Function on Pins RG<4:0>, respectively
0 = Digital I/O. Pin is assigned to port or digital special function.
1 = Analog input. Pin is assigned as analog input(1). Digital input buffer disabled.

bit 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.

**REGISTER 12-28: WPUG: WEAK PULL-UP PORTG REGISTER**

| U-0 | U-0 | R/W-1/1 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | WPUG5 | — | — | — | — | — |
| 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 **WPUG5**: Weak Pull-up Register bits
1 = Pull-up enabled
0 = Pull-up disabled

bit 4-0 Unimplemented: Read as '0'.

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


© 2010-2016 Microchip Technology Inc.

**TABLE 12-17: SUMMARY OF REGISTERS ASSOCIATED WITH PORTG**

| 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 | 161 |
| ANSELG | — | — | — | ANSG4 | ANSG3 | ANSG2 | ANSG1 | — | 144 |
| CCPxCON | PxM<1:0>(1) | | DCxB<1:0> | | CCPxM<3:0> | | | | 227 |
| CMOUT | — | — | — | — | — | MC3OUT | MC2OUT | MC1OUT | 179 |
| CM1CON1 | C1INTP | C1INTN | C1PCH1 | C1PCH0 | — | — | C1NCH<1:0> | | 179 |
| CM2CON1 | C2INTP | C2INTN | C2PCH1 | C2PCH0 | — | — | C2NCH<1:0> | | 179 |
| CPSCON0 | CPSON | CPSRM | — | — | CPSRNG<1:0> | | CPSOUT | T0XCS | 322 |
| CPSCON1 | — | — | — | — | CPSCH<3:0> | | | | 323 |
| LATG | — | — | — | LATG4 | LATG3 | LATG2 | LATG1 | LATG0 | 143 |
| LCDCON | LCDEN | SLPEN | WERR | — | CS<1:0> | | LMUX<1:0> | | 326 |
| LCDSE3 | SE31 | SE30 | SE29 | SE28 | SE27 | SE26 | SE25 | SE24 | 330 |
| LCDSE5 | — | — | SE45 | SE44 | SE43 | SE42 | SE41 | SE40 | 330 |
| PORTG | — | — | RG5 | RG4 | RG3 | RG2 | RG1 | RG0 | 143 |
| TRISG | — | — | TRISG5 | TRISG4 | TRISG3 | TRISG2 | TRISG1 | TRISG0 | 143 |
| WPUG | — | — | WPUG5 | — | — | — | — | — | 144 |

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

**Note 1:** Applies to ECCP modules only.

## 13.0 INTERRUPT-ON-CHANGE

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

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

Figure 13-1 is a block diagram of the IOC module.

### 13.1 Enabling the Module

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

### 13.2 Individual Pin Configuration

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

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

### 13.3 Interrupt Flags

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

### 13.4 Clearing Interrupt Flags

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

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

**EXAMPLE 13-1: CLEARING INTERRUPT FLAGS (PORTA EXAMPLE)**

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

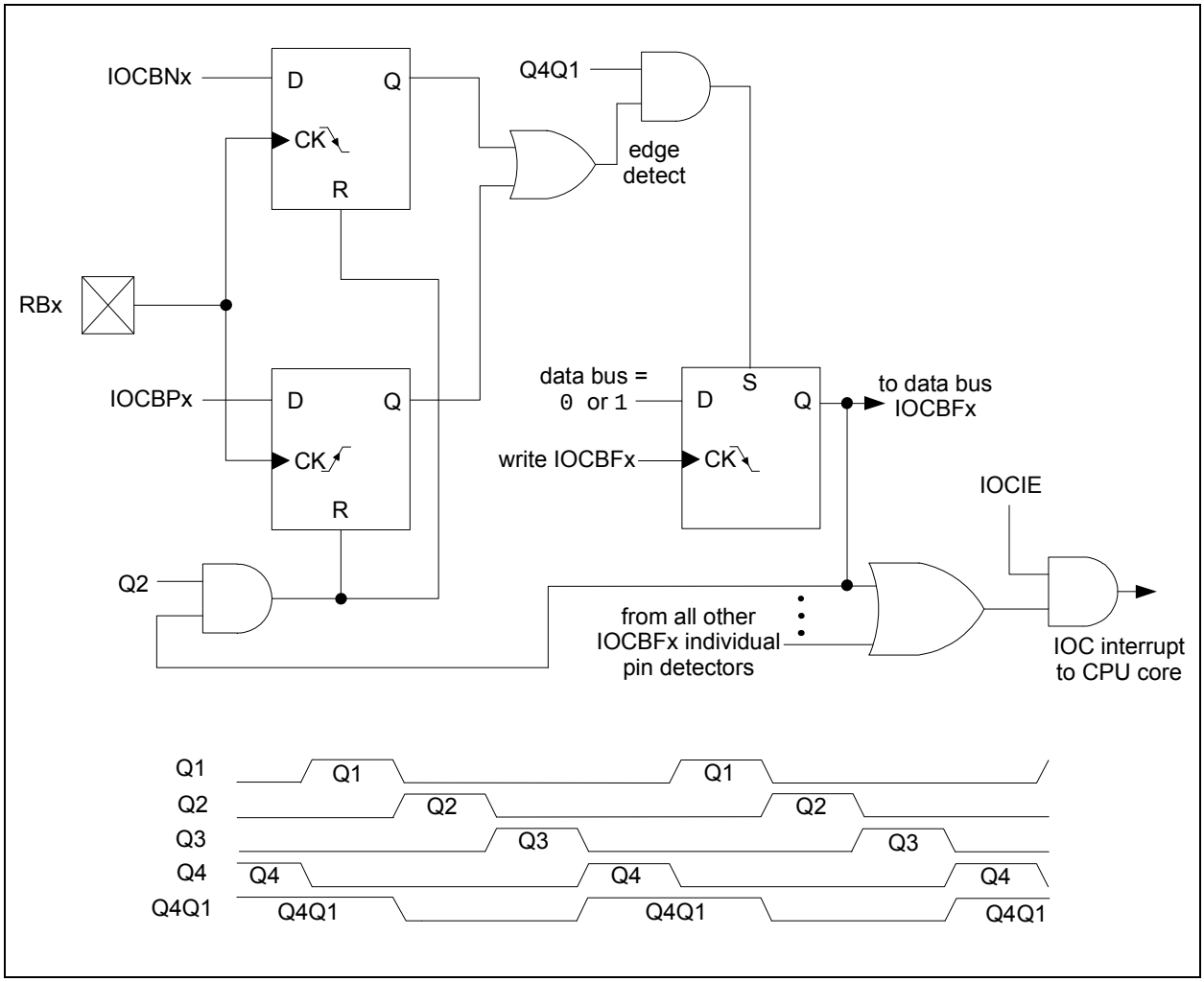
### 13.5 Operation in Sleep

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

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


© 2010-2016 Microchip Technology Inc.

**FIGURE 13-1: INTERRUPT-ON-CHANGE BLOCK DIAGRAM**



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

**REGISTER 13-1: IOCBP: INTERRUPT-ON-CHANGE PORTB POSITIVE EDGE REGISTER**

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

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

bit 7-0 **IOCBP<7:0>:** Interrupt-on-Change 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.

**REGISTER 13-2: IOCBN: INTERRUPT-ON-CHANGE PORTB NEGATIVE EDGE REGISTER**

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

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

bit 7-0 **IOCBN<7:0>:** Interrupt-on-Change 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.

**REGISTER 13-3: 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 | R/W/HS-0/0 | R/W/HS-0/0 | R/W/HS-0/0 | R/W/HS-0/0 |
|---|---|---|---|---|---|---|---|
| IOCBF7 | IOCBF6 | IOCBF5 | IOCBF4 | IOCBF3 | IOCBF2 | IOCBF1 | IOCBF0 |
| bit 7 | | | | | | | bit 0 |

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

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


© 2010-2016 Microchip Technology Inc.

**TABLE 13-1: 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 |
|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| IOCBF | IOCBF7 | IOCBF6 | IOCBF5 | IOCBF4 | IOCBF3 | IOCBF2 | IOCBF1 | IOCBF0 | 148 |
| IOCBN | IOCBN7 | IOCBN6 | IOCBN5 | IOCBN4 | IOCBN3 | IOCBN2 | IOCBN1 | IOCBN0 | 148 |
| IOCBP | IOCBP7 | IOCBP6 | IOCBP5 | IOCBP4 | IOCBP3 | IOCBP2 | IOCBP1 | IOCBP0 | 148 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 128 |

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

## 14.0 FIXED VOLTAGE REFERENCE (FVR)

The Fixed Voltage Reference, or FVR, is a stable voltage reference, independent of VDD, with 1.024V, 2.048V or 4.096V selectable output levels. The output of the FVR can be configured to supply a reference voltage to the following:

- ADC input channel
- ADC positive reference
- Comparator positive input
- Digital-to-Analog Converter (DAC)
- Capacitive Sensing (CPS) module
- LCD bias generator

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

### 14.1 Independent Gain Amplifiers

The output of the FVR supplied to the ADC, Comparators, DAC and CPS module is routed through two independent programmable gain amplifiers. Each amplifier can be configured to amplify the reference voltage by 1x, 2x or 4x, to produce the three possible voltage levels.

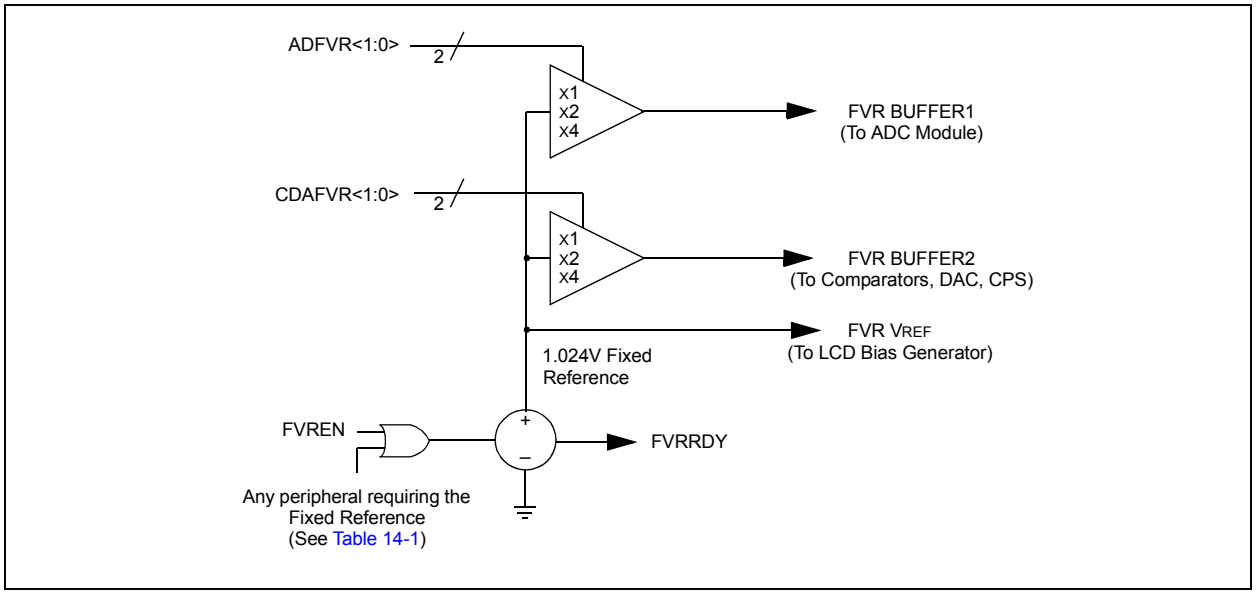
The ADFVR<1:0> bits of the FVRCON register are used to enable and configure the gain amplifier settings for the reference supplied to the ADC module. Reference **Section 16.0 "Analog-to-Digital Converter (ADC) Module"** for additional information.

The CDAFVR<1:0> bits of the FVRCON register are used to enable and configure the gain amplifier settings for the reference supplied to the Comparators, DAC and CPS module. Reference **Section 17.0 "Digital-to-Analog Converter (DAC) Module"**, **Section 18.0 "Comparator Module"** and **Section 26.0 "Capacitive Sensing (CPS) Module"** for additional information.

### 14.2 FVR Stabilization Period

When the Fixed Voltage Reference module is enabled, it requires time for the reference and amplifier circuits to stabilize. Once the circuits stabilize and are ready for use, the FVRRDY bit of the FVRCON register will be set. See **Section 30.0 "Electrical Specifications"** for the minimum delay requirement.

**FIGURE 14-1: VOLTAGE REFERENCE BLOCK DIAGRAM**





© 2010-2016 Microchip Technology Inc.

## 14.3 Register Definitions: FVR Control

**REGISTER 14-1: FVRCON: FIXED VOLTAGE REFERENCE CONTROL REGISTER**

| R/W-0/0 | R-q/q | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| FVREN | FVRRDY(1) | TSEN | TSRNG | CDAFVR<1:0> | | ADFVR<1:0> | |
| bit 7 | | | | | | | bit 0 |

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

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

bit 6 **FVRRDY:** Fixed Voltage Reference Ready Flag bit(1)
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(3)
1 = Temperature Indicator is enabled
0 = Temperature Indicator is disabled

bit 4 **TSRNG:** Temperature Indicator Range Selection bit(3)
1 = VOUT = VDD - 4VT (High Range)
0 = VOUT = VDD - 2VT (Low Range)

bit 3-2 **CDAFVR<1:0>:** Comparator and DAC Fixed Voltage Reference Selection bit
11 = Comparator and DAC and CPS Fixed Voltage Reference Peripheral output is 4x (4.096V)(2)
10 = Comparator and DAC and CPS Fixed Voltage Reference Peripheral output is 2x (2.048V)(2)
01 = Comparator and DAC and CPS Fixed Voltage Reference Peripheral output is 1x (1.024V)
00 = Comparator and DAC and CPS Fixed Voltage Reference Peripheral output is off

bit 1-0 **ADFVR<1:0>:** ADC Fixed Voltage Reference Selection bit
11 = ADC Fixed Voltage Reference Peripheral output is 4x (4.096V)(2)
10 = ADC Fixed Voltage Reference Peripheral output is 2x (2.048V)(2)
01 = ADC Fixed Voltage Reference Peripheral output is 1x (1.024V)
00 = ADC Fixed Voltage Reference Peripheral output is off

**Note 1:** FVRRDY is always '1' on PIC16F1946/47 only.
**2:** Fixed Voltage Reference output cannot exceed VDD.
**3:** See **Section 15.0 "Temperature Indicator Module"** for additional information.

**TABLE 14-1: 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 | CDAFVR<1:0> | | ADFVR<1:0> | | 151 |

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

## 15.0 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°C and +85°C. The output is a voltage that is proportional to the device temperature. The output of the temperature indicator is internally connected to the device ADC.

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

### 15.1 Circuit Operation

Figure 15-1 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 15-1 describes the output characteristics of the temperature indicator.

**EQUATION 15-1: VOUT RANGES**

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

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

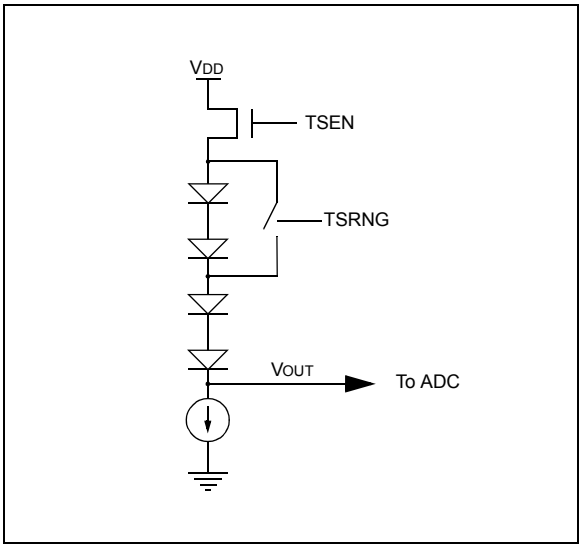
The temperature sense circuit is integrated with the Fixed Voltage Reference (FVR) module. See **Section 14.0 "Fixed Voltage Reference (FVR)"** for more information.

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

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

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

**FIGURE 15-1: TEMPERATURE CIRCUIT DIAGRAM**




### 15.2 Minimum Operating VDD

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

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

Table 15-1 shows the recommended minimum VDD vs. range setting.

**TABLE 15-1: RECOMMENDED VDD VS. RANGE**

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

### 15.3 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 16.0 "Analog-to-Digital Converter (ADC) Module"** for detailed information.

| | |
|---|---|
| **Note:** | Every time the ADC MUX is changed to the temperature indicator output selection (CHS bit in the ADCCON0 register), wait 500 μsec for the sampling capacitor to fully charge before sampling the temperature indicator output. |


© 2010-2016 Microchip Technology Inc.

## 15.4 ADC Acquisition Time

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

**TABLE 15-1: 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> | | 151 |

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

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

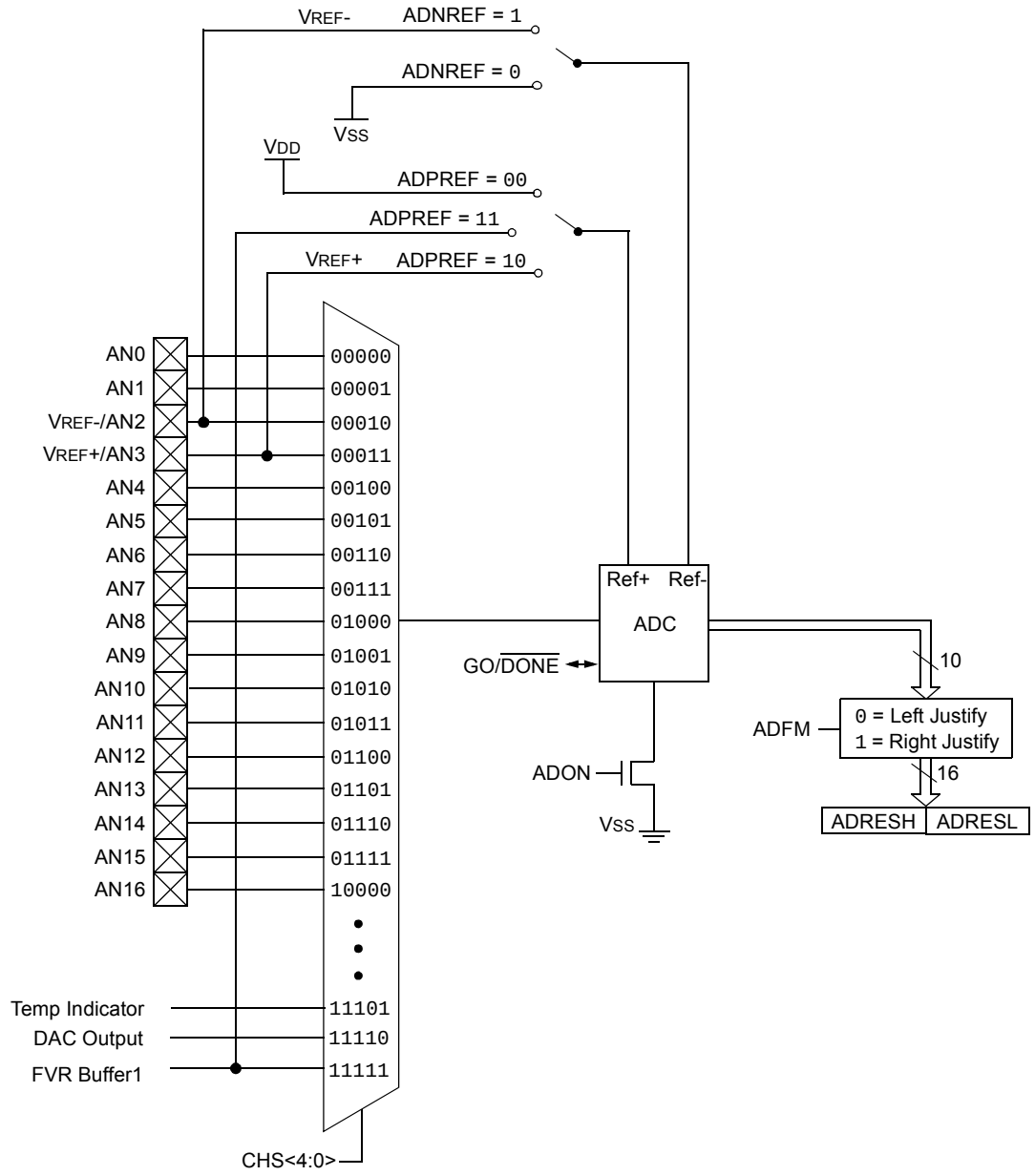
The Analog-to-Digital Converter (ADC) allows conversion of an analog input signal to a 10-bit binary representation of that signal. This device uses analog inputs, which are multiplexed into a single sample and hold circuit. The output of the sample and hold is connected to the input of the converter. 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 16-1 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.


© 2010-2016 Microchip Technology Inc.

**FIGURE 16-1: ADC BLOCK DIAGRAM**




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

## 16.1 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

### 16.1.1 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 12.0 "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. |

### 16.1.2 CHANNEL SELECTION

There are 20 selections available:

- AN<16:0> pins
- Temperature Indicator
- DAC Output
- FVR (Fixed Voltage Reference) Output

Refer to **Section 15.0 "Temperature Indicator Module"**, **Section 17.0 "Digital-to-Analog Converter (DAC) Module"** and **Section 14.0 "Fixed Voltage Reference (FVR)"** for more information on these channel selections.

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

When changing channels, a delay is required before starting the next conversion. Refer to **Section 16.2 "ADC Operation"** for more information.

### 16.1.3 ADC VOLTAGE REFERENCE

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

- VREF+ pin
- VDD

The ADNREF bit of the ADCON1 register provides control of the negative voltage reference. The negative voltage reference can be:

- VREF- pin
- VSS

See **Section 14.0 "Fixed Voltage Reference (FVR)"** for more details on the fixed voltage reference.

### 16.1.4 CONVERSION CLOCK

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

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

The time to complete one bit conversion is defined as TAD. One full 10-bit conversion requires 11.5 TAD periods as shown in Figure 16-2.

For correct conversion, the appropriate TAD specification must be met. Refer to the A/D conversion requirements in **Section 30.0 "Electrical Specifications"** for more information. Table 16-1 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. |


© 2010-2016 Microchip Technology Inc.

**TABLE 16-1: ADC CLOCK PERIOD ($T_{AD}$) VS. DEVICE OPERATING FREQUENCIES**

| ADC Clock Period ($T_{AD}$) | | Device Frequency ($F_{OSC}$) | | | | | |
|---|---|---|---|---|---|---|---|
| ADC Clock Source | ADCS<2:0> | 32 MHz | 20 MHz | 16 MHz | 8 MHz | 4 MHz | 1 MHz |
| Fosc/2 | 000 | 62.5ns(2) | 100 ns(2) | 125 ns(2) | 250 ns(2) | 500 ns(2) | 2.0 μs |
| Fosc/4 | 100 | 125 ns(2) | 200 ns(2) | 250 ns(2) | 500 ns(2) | 1.0 μs | 4.0 μs |
| Fosc/8 | 001 | 0.5 μs(2) | 400 ns(2) | 0.5 μs(2) | 1.0 μs | 2.0 μs | 8.0 μs(3) |
| Fosc/16 | 101 | 800 ns | 800 ns | 1.0 μs | 2.0 μs | 4.0 μs | 16.0 μs(3) |
| Fosc/32 | 010 | 1.0 μs | 1.6 μs | 2.0 μs | 4.0 μs | 8.0 μs(3) | 32.0 μs(3) |
| Fosc/64 | 110 | 2.0 μs | 3.2 μs | 4.0 μs | 8.0 μs(3) | 16.0 μs(3) | 64.0 μs(3) |
| FRC | x11 | 1.0-6.0 μs(1,4) | 1.0-6.0 μs(1,4) | 1.0-6.0 μs(1,4) | 1.0-6.0 μs(1,4) | 1.0-6.0 μs(1,4) | 1.0-6.0 μs(1,4) |

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

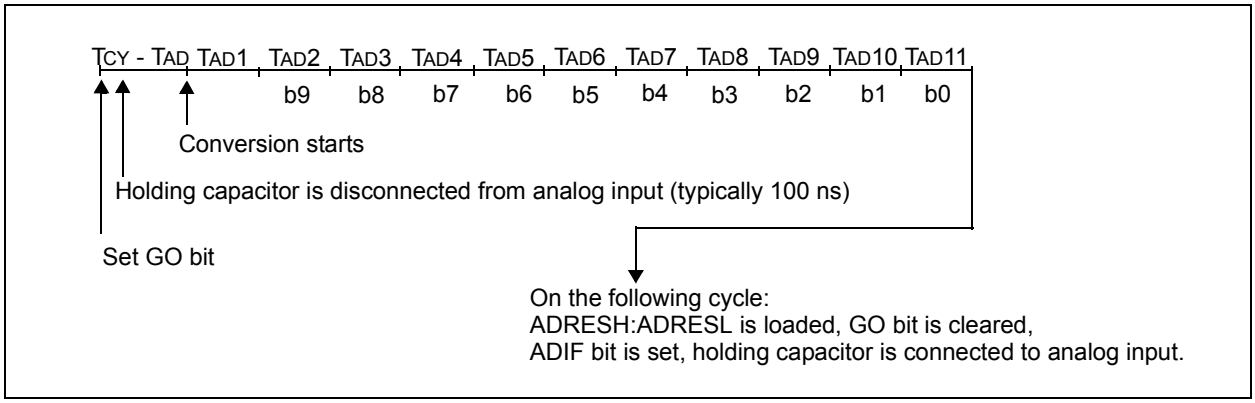
**Note 1:** The FRC source has a typical TAD time of 1.6 μs for VDD.

**2:** These values violate the minimum required TAD time.

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

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

**FIGURE 16-2: ANALOG-TO-DIGITAL CONVERSION $T_{AD}$ CYCLES**



### 16.1.5 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. |

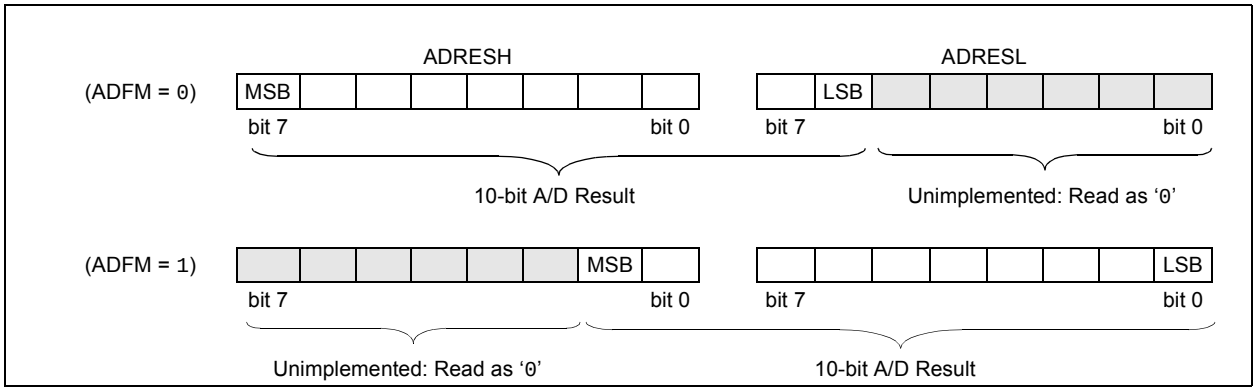
This interrupt can be generated while the device is operating or while in Sleep. If the device is in Sleep, the interrupt will wake-up the device. Upon waking from Sleep, the next instruction following the SLEEP instruction is always executed. If the user is attempting to wake-up from Sleep and resume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

### 16.1.6 RESULT FORMATTING

The 10-bit A/D conversion result can be supplied in two formats, left justified or right justified. The ADFM bit of the ADCON1 register controls the output format.

Figure 16-4 shows the two output formats.

**FIGURE 16-3: 10-BIT A/D CONVERSION RESULT FORMAT**





© 2010-2016 Microchip Technology Inc.

## 16.2 ADC Operation

### 16.2.1 STARTING A CONVERSION

To enable the ADC module, the ADON bit of the ADCON0 register must be set to a '1'. Setting the GO/$\overline{\text{DONE}}$ bit of the ADCON0 register to a '1' will start the Analog-to-Digital conversion.

| | |
|---|---|
| **Note:** | The GO/$\overline{\text{DONE}}$ bit should not be set in the same instruction that turns on the ADC. Refer to **Section 16.3.2 "A/D Conversion Procedure"**. |

### 16.2.2 COMPLETION OF A CONVERSION

When the conversion is complete, the ADC module will:

- Clear the GO/$\overline{\text{DONE}}$ bit
- Set the ADIF Interrupt Flag bit
- Update the ADRESH and ADRESL registers with the new conversion result

### 16.2.3 TERMINATING A CONVERSION

If a conversion must be terminated before completion, the GO/$\overline{\text{DONE}}$ bit can be cleared in software. The ADRESH and ADRESL registers will be updated with the partially complete Analog-to-Digital conversion sample. Incomplete bits will match the last bit converted.

| | |
|---|---|
| **Note:** | A device Reset forces all registers to their Reset state. Thus, the ADC module is turned off and any pending conversion is terminated. |

## 16.3 ADC Operation During Sleep

The ADC module can operate during Sleep. This requires the ADC clock source to be set to the FRC option. When the FRC clock source is selected, the ADC waits one additional instruction before starting the conversion. This allows the SLEEP instruction to be executed, which can reduce system noise during the conversion. If the ADC interrupt is enabled, the device will wake-up from Sleep when the conversion completes. If the ADC interrupt is disabled, the ADC module is turned off after the conversion completes, although the ADON bit remains set.

When the ADC clock source is something other than FRC, a SLEEP instruction causes the present conversion to be aborted and the ADC module is turned off, although the ADON bit remains set.

### 16.3.1 SPECIAL EVENT TRIGGER

The Special Event Trigger of the CCPx/ECCPX module allows periodic ADC measurements without software intervention. When this trigger occurs, the GO/$\overline{\text{DONE}}$ bit is set by hardware and the Timer1 counter resets to zero.

**TABLE 16-2: SPECIAL EVENT TRIGGER**

| Device | CCPx/ECCPx |
|---|---|
| PIC16(L)F1946/47 | CCP5 |

Using the Special Event Trigger does not assure proper ADC timing. It is the user's responsibility to ensure that the ADC timing requirements are met.

Refer to **Section 23.0 "Capture/Compare/PWM Modules"** for more information.

### 16.3.2 A/D CONVERSION PROCEDURE

This is an example procedure for using the ADC to perform an Analog-to-Digital conversion:

1. Configure Port:
   - Disable pin output driver (Refer to the TRIS register)
   - Configure pin as analog (Refer to the ANSEL register)
   - 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(1)
4. Wait the required acquisition time(2).
5. Start conversion by setting the GO/$\overline{\text{DONE}}$ bit.
6. Wait for ADC conversion to complete by one of the following:
   - Polling the GO/$\overline{\text{DONE}}$ bit
   - Waiting for the ADC interrupt (interrupts enabled)
7. Read ADC Result.
8. Clear the ADC interrupt flag (required if interrupt is enabled).

**Note 1:** The global interrupt can be disabled if the user is attempting to wake-up from Sleep and resume in-line code execution.

**2:** Refer to **Section 16.5 "A/D Acquisition Requirements"**.

**EXAMPLE 16-1: A/D CONVERSION**

```
;This code block configures the ADC
;for polling, Vdd and Vss references, Frc
;clock and AN0 input.
;
;Conversion start & polling for completion
; are included.
;
BANKSEL   ADCON1      ;
MOVLW     B'11110000' ;Right justify, Frc
                      ;clock
MOVWF     ADCON1      ;Vdd and Vss Vref
BANKSEL   TRISA       ;
BSF       TRISA,0     ;Set RA0 to input
BANKSEL   ANSEL       ;
BSF       ANSEL,0     ;Set RA0 to analog
BANKSEL   ADCON0      ;
MOVLW     B'00000001' ;Select channel AN0
BANKSEL   WPUA
BCF       WPUA, 0     ;Disable weak
                      ;pull-up on RA0
MOVWF     ADCON0      ;Turn ADC On
CALL      SampleTime  ;Acquisiton delay
BSF       ADCON0,ADGO ;Start conversion
BTFSC     ADCON0,ADGO ;Is conversion done?
GOTO      $-1         ;No, test again
BANKSEL   ADRESH      ;
MOVF      ADRESH,W    ;Read upper 2 bits
MOVWF     RESULTHI    ;store in GPR space
BANKSEL   ADRESL      ;
MOVF      ADRESL,W    ;Read lower 8 bits
MOVWF     RESULTLO    ;Store in GPR space
```


© 2010-2016 Microchip Technology Inc.

## 16.4 Register Definitions: ADC Control

**REGISTER 16-1: ADCON0: A/D 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/$\overline{\text{DONE}}$ | 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

11111 = FVR (Fixed Voltage Reference) Buffer 1 Output(2)
11110 = DAC output(1)
11101 = Temperature Indicator(3)
11100 = Reserved. No channel connected.
•
•
•
10001 = Reserved. No channel connected.
10000 = AN16
01111 = AN15
01110 = AN14
01101 = AN13
01100 = AN12
01011 = AN11
01010 = AN10
01001 = AN9
01000 = AN8
00111 = AN7
00110 = AN6
00101 = AN5
00100 = AN4
00011 = AN3
00010 = AN2
00001 = AN1
00000 =AN0

bit 1 **GO/$\overline{\text{DONE}}$:** A/D Conversion Status bit

1 = A/D conversion cycle in progress. Setting this bit starts an A/D conversion cycle.
This bit is automatically cleared by hardware when the A/D conversion has completed.
0 = A/D conversion completed/not in progress

bit 0 **ADON:** ADC Enable bit

1 = ADC is enabled
0 = ADC is disabled and consumes no operating current

**Note 1:** See **Section 17.0 "Digital-to-Analog Converter (DAC) Module"** for more information.
**2:** See **Section 14.0 "Fixed Voltage Reference (FVR)"** for more information.
**3:** See **Section 15.0 "Temperature Indicator Module"** for more information.

**REGISTER 16-2: ADCON1: A/D CONTROL REGISTER 1**

| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| ADFM | ADCS<2:0> | | | — | ADNREF | ADPREF<1:0> | |
| bit 7 | | | | | | | bit 0 |

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

bit 7 **ADFM:** A/D Result Format Select bit
1 = Right justified. Six Most Significant bits of ADRESH are set to '0' when the conversion result is loaded.
0 = Left justified. Six Least Significant bits of ADRESL are set to '0' when the conversion result is loaded.

bit 6-4 **ADCS<2:0>:** A/D Conversion Clock Select bits
111 = FRC (clock supplied from a dedicated RC oscillator)
110 = FOSC/64
101 = FOSC/16
100 = FOSC/4
011 = FRC (clock supplied from a dedicated RC oscillator)
010 = FOSC/32
001 = FOSC/8
000 = FOSC/2

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

bit 2 **ADNREF:** A/D Negative Voltage Reference Configuration bit
1 = VREF- is connected to external VREF- pin(1)
0 = VREF- is connected to VSS

bit 1-0 **ADPREF<1:0>:** A/D Positive Voltage Reference Configuration bits
11 = VREF+ is connected to internal Fixed Voltage Reference (FVR) module(1)
10 = VREF+ is connected to external VREF+ pin(1)
01 = Reserved
00 = VREF+ is connected to VDD

**Note 1:** When selecting the FVR or the VREF+ pin as the source of the positive reference, be aware that a minimum voltage specification exists. See **Section 30.0 "Electrical Specifications"** for details.


© 2010-2016 Microchip Technology Inc.

**REGISTER 16-3: ADRESH: ADC RESULT REGISTER HIGH (ADRESH) ADFM = 0**

| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
|---|---|---|---|---|---|---|---|
| ADRES<9:2> | | | | | | | |
| bit 7 | | | | | | | bit 0 |

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

bit 7-0 **ADRES<9:2>**: ADC Result Register bits
Upper eight bits of 10-bit conversion result

**REGISTER 16-4: ADRESL: ADC RESULT REGISTER LOW (ADRESL) ADFM = 0**

| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
|---|---|---|---|---|---|---|---|
| ADRES<1:0> | | — | — | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

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

bit 7-6 **ADRES<1:0>**: ADC Result Register bits
Lower two bits of 10-bit conversion result

bit 5-0 **Reserved**: Do not use.

**REGISTER 16-5: ADRESH: ADC RESULT REGISTER HIGH (ADRESH) ADFM = 1**

| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | ADRES<9:8> | |
| bit 7 | | | | | | | bit 0 |

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

bit 7-2 **Reserved**: Do not use.

bit 1-0 **ADRES<9:8>**: ADC Result Register bits
Upper two bits of 10-bit conversion result

**REGISTER 16-6: ADRESL: ADC RESULT REGISTER LOW (ADRESL) ADFM = 1**

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

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

bit 7-0 **ADRES<7:0>**: ADC Result Register bits
Lower eight bits of 10-bit conversion result


© 2010-2016 Microchip Technology Inc.

## 16.5 A/D Acquisition Requirements

For the ADC to meet its specified accuracy, the charge holding capacitor (CHOLD) must be allowed to fully charge to the input channel voltage level. The Analog Input model is shown in Figure 16-4. The source impedance (RS) and the internal sampling switch (RSS) impedance directly affect the time required to charge the capacitor CHOLD. The sampling switch (RSS) impedance varies over the device voltage (VDD), refer to Figure 16-4. **The maximum recommended impedance for analog sources is 10 kΩ.** As the source impedance is decreased, the acquisition time may be decreased. After the analog input channel is selected (or changed), an A/D acquisition must be done before the conversion can be started. To calculate the minimum acquisition time, Equation 16-1 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 16-1: ACQUISITION TIME EXAMPLE**

*Assumptions:* *Temperature = 50°C and external impedance of 10kΩ 5.0V VDD*

$$\begin{aligned} TACQ &= \text{Amplifier Settling Time} + \text{Hold Capacitor Charging Time} + \text{Temperature Coefficient} \\ &= TAMP + TC + TCOFF \\ &= 2\mu s + TC + [(\text{Temperature} - 25°C)(0.05\mu s/°C)] \end{aligned}$$

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

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

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

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

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

*Solving for TC:*

$$\begin{aligned} TC &= -CHOLD(RIC + RSS + RS)\ ln(1/2047) \\ &= -10pF(1k\Omega + 7k\Omega + 10k\Omega)\ln(4.88 \times 10^{-4}) \\ &= 1.37\mu s \end{aligned}$$

*Therefore:*

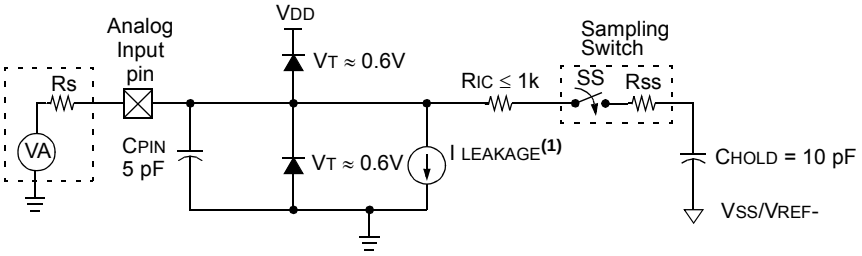
$$\begin{aligned} TACQ &= 2\mu s + 1.37\mu s + [(50°C - 25°C)(0.05\mu s/°C)] \\ &= 4.62\mu s \end{aligned}$$

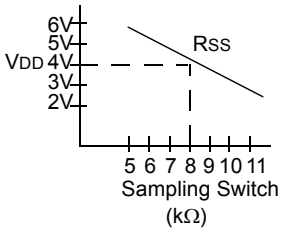
**Note 1:** The reference voltage (VREF) has no effect on the equation, since it cancels itself out.

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

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

**FIGURE 16-4: ANALOG INPUT MODEL**



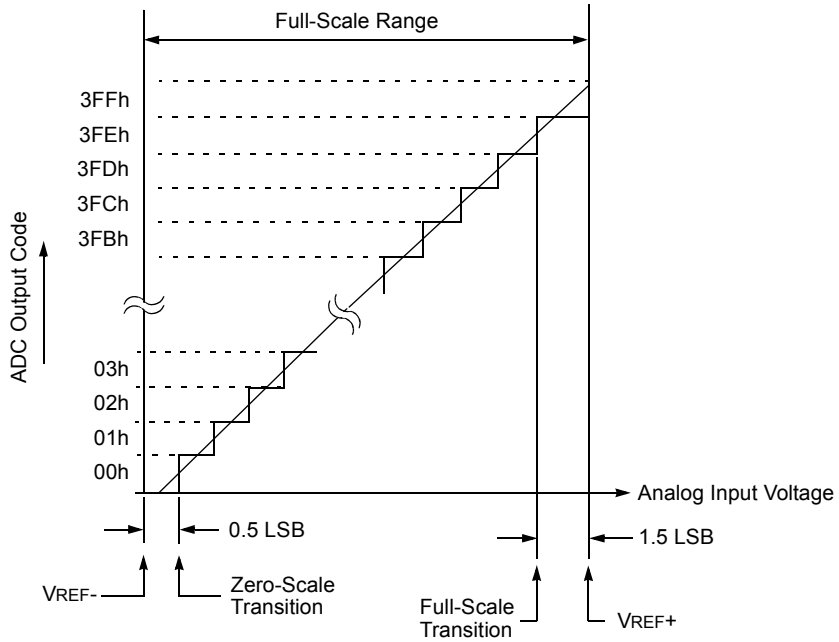




**Legend:**

| | | |
|---|---|---|
| CHOLD | = | Sample/Hold Capacitance |
| CPIN | = | Input Capacitance |
| I LEAKAGE | = | Leakage current at the pin due to various junctions |
| RIC | = | Interconnect Resistance |
| RSS | = | Resistance of Sampling Switch |
| SS | = | Sampling Switch |
| VT | = | Threshold Voltage |

**Note 1:** Refer to **Section 30.0 "Electrical Specifications"**.

**FIGURE 16-5: ADC TRANSFER FUNCTION**





© 2010-2016 Microchip Technology Inc.

**TABLE 16-3: 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 | 161 |
| ADCON1 | ADFM | ADCS<2:0> | | | — | ADNREF | ADPREF<1:0> | | 162 |
| ADRESH | A/D Result Register High | | | | | | | | 163 |
| ADRESL | A/D Result Register Low | | | | | | | | 163 |
| ANSELA | — | — | ANSA5 | — | ANSA3 | ANSA2 | ANSA1 | ANSA0 | 126 |
| ANSELF | ANSELF7 | ANSELF6 | ANSELF5 | ANSELF4 | ANSELF3 | ANSELF2 | ANSELF1 | ANSELF0 | 141 |
| ANSELG | — | — | — | ANSELG4 | ANSELG3 | ANSELG2 | ANSELG1 | — | 144 |
| CCP1CON | P1M<1:0> | | DC1B<1:0> | | CCP1M<3:0> | | | | 227 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 91 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 95 |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 125 |
| TRISF | TRISF7 | TRISF6 | TRISF5 | TRISF4 | TRISF3 | TRISF2 | TRISF1 | TRISF0 | 140 |
| TRISG | TRISG7 | TRISG6 | TRISG5 | TRISG4 | TRISG3 | TRISG2 | TRISG1 | TRISG0 | 143 |
| FVRCON | FVREN | FVRRDY | TSEN | TSRNG | CDAFVR<1:0> | | ADFVR<1:0> | | 151 |
| DACCON0 | DACEN | DACLPS | DACOE | — | DACPSS<1:0> | | — | DACNSS | 171 |
| DACCON1 | — | — | — | DACR<4:0> | | | | | 171 |

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

## 17.0 DIGITAL-TO-ANALOG CONVERTER (DAC) MODULE

The Digital-to-Analog Converter supplies a variable voltage reference, ratiometric with the input source, with 32 selectable output levels.

The input of the DAC can be connected to:

- External VREF pins
- VDD supply voltage
- FVR (Fixed Voltage Reference)

The output of the DAC can be configured to supply a reference voltage to the following:

- Comparator positive input
- ADC input channel
- DACOUT pin
- Capacitive Sensing module (CPS)

The Digital-to-Analog Converter (DAC) can be enabled by setting the DACEN bit of the DACCON0 register.

### 17.1 Output Voltage Selection

The DAC has 32 voltage level ranges. The 32 levels are set with the DACR<4:0> bits of the DACCON1 register.

The DAC output voltage is determined by the following equations:

**EQUATION 17-1: DAC OUTPUT VOLTAGE**

***IF DACEN = 1***

$$V_{OUT} = \left( (V_{SOURCE+} - V_{SOURCE-}) \times \frac{DACR[4:0]}{2^5} \right) + V_{SOURCE-}$$

***IF DACEN = 0 & DACLPS = 1 & DACR[4:0] = 11111***

$$V_{OUT} = V_{SOURCE+}$$

***IF DACEN = 0 & DACLPS = 0 & DACR[4:0] = 00000***

$$V_{OUT} = V_{SOURCE-}$$

*VSOURCE+ = VDD, VREF, or FVR BUFFER 2*

*VSOURCE- = VSS*

### 17.2 Ratiometric Output Level

The DAC output value is derived using a resistor ladder with each end of the ladder tied to a positive and negative voltage reference input source. If the voltage of either input source fluctuates, a similar fluctuation will result in the DAC output value.

The value of the individual resistors within the ladder can be found in **Section 30.0 "Electrical Specifications"**.

### 17.3 DAC Voltage Reference Output

The DAC can be output to the DACOUT pin by setting the DACOE bit of the DACCON0 register to '1'. Selecting the DAC reference voltage for output on the DACOUT pin automatically overrides the digital output buffer and digital input threshold detector functions of that pin. Reading the DACOUT pin when it has been configured for DAC reference voltage output will always return a '0'.

Due to the limited current drive capability, a buffer must be used on the DAC voltage reference output for external connections to DACOUT. Figure 17-2 shows an example buffering technique.


© 2010-2016 Microchip Technology Inc.

**FIGURE 17-1: DIGITAL-TO-ANALOG CONVERTER BLOCK DIAGRAM**

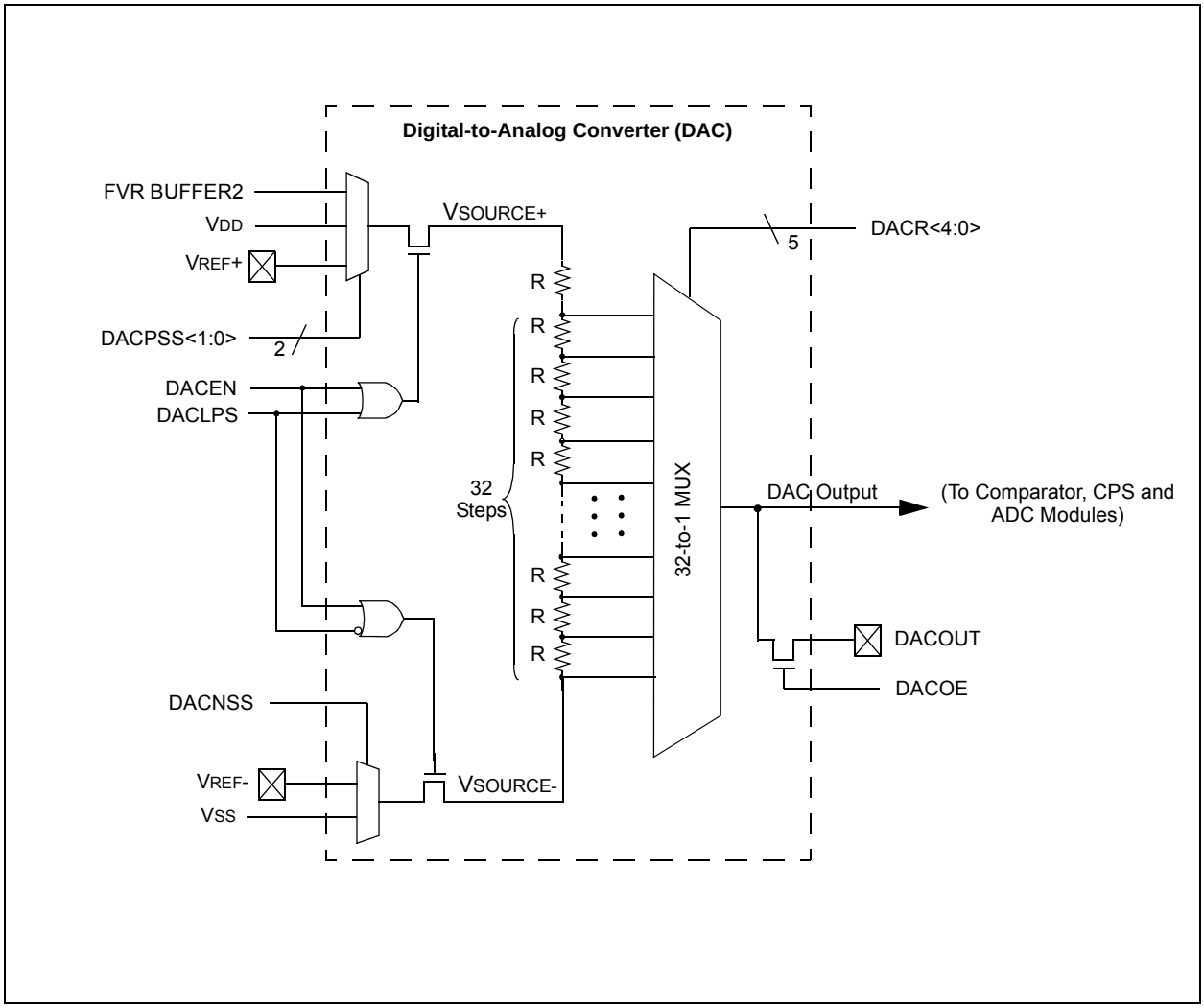
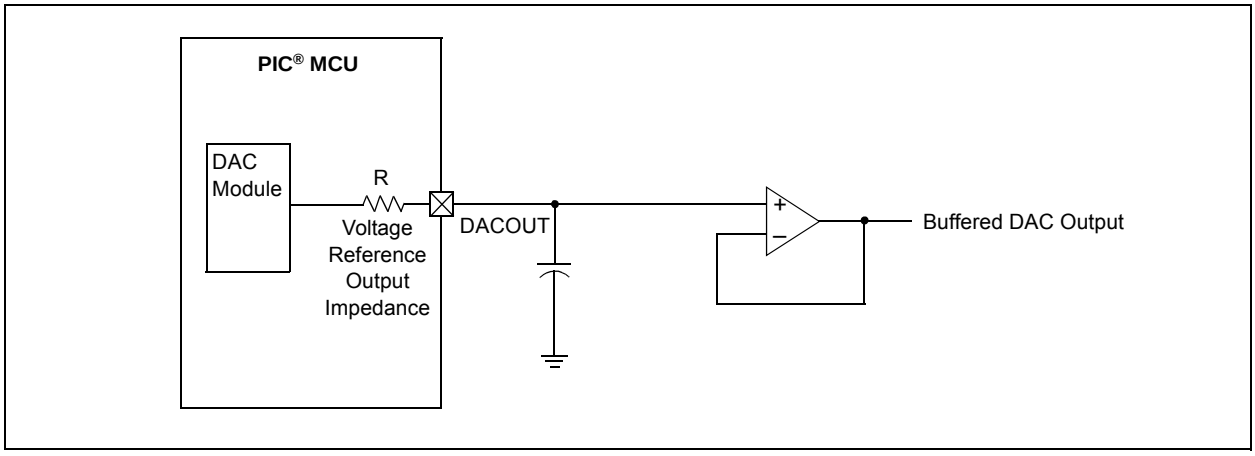



**FIGURE 17-2: VOLTAGE REFERENCE OUTPUT BUFFER EXAMPLE**



## 17.4 Low-Power Voltage State

In order for the DAC module to consume the least amount of power, one of the two voltage reference input sources to the resistor ladder must be disconnected. Either the positive voltage source, ($V_{SOURCE+}$), or the negative voltage source, ($V_{SOURCE-}$) can be disabled.

The negative voltage source is disabled by setting the DACLPS bit in the DACCON0 register. Clearing the DACLPS bit in the DACCON0 register disables the positive voltage source.

### 17.4.1 OUTPUT CLAMPED TO POSITIVE VOLTAGE SOURCE

The DAC output voltage can be set to $V_{SOURCE+}$ with the least amount of power consumption by performing the following:

- Clearing the DACEN bit in the DACCON0 register.
- Setting the DACLPS bit in the DACCON0 register.
- Configuring the DACPSS bits to the proper positive source.
- Configuring the DACR<4:0> bits to '11111' in the DACCON1 register.

This is also the method used to output the voltage level from the FVR to an output pin. See **Section 17.5 "Operation During Sleep"** for more information.

Refer to Figure 17-3 for output clamping examples.

### 17.4.2 OUTPUT CLAMPED TO NEGATIVE VOLTAGE SOURCE

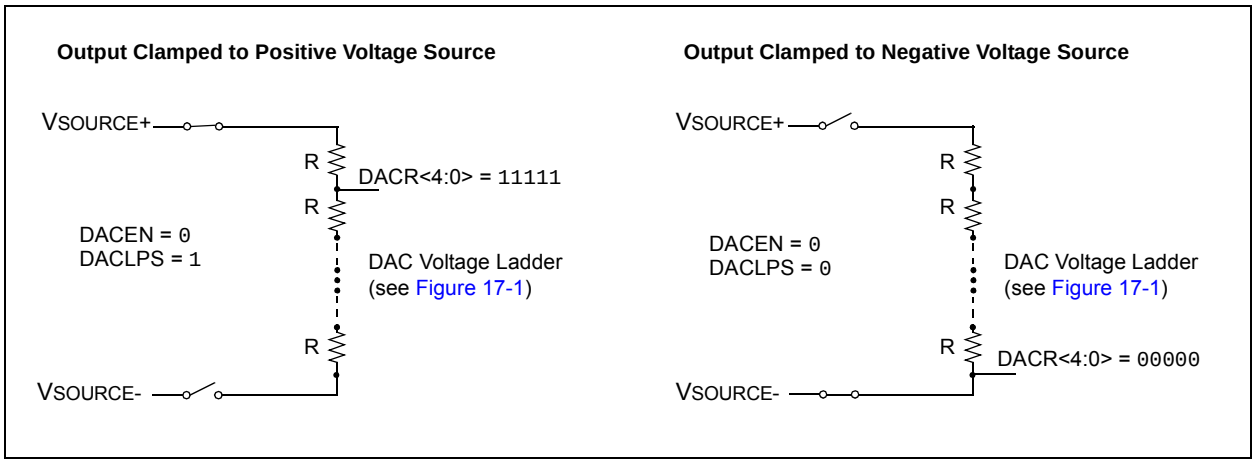
The DAC output voltage can be set to $V_{SOURCE-}$ with the least amount of power consumption by performing the following:

- Clearing the DACEN bit in the DACCON0 register.
- Clearing the DACLPS bit in the DACCON0 register.
- Configuring the DACNSS bits to the proper negative source.
- Configuring the DACR<4:0> bits to '00000' in the DACCON1 register.

This allows the comparator to detect a zero-crossing while not consuming additional current through the DAC module.

Refer to Figure 17-3 for output clamping examples.

**FIGURE 17-3: OUTPUT VOLTAGE CLAMPING EXAMPLES**




## 17.5 Operation During Sleep

When the device wakes up from Sleep through an interrupt or a Watchdog Timer time-out, the contents of the DACCON0 register are not affected. To minimize current consumption in Sleep mode, the voltage reference should be disabled.

## 17.6 Effects of a Reset

A device Reset affects the following:

- DAC is disabled.
- DAC output voltage is removed from the DACOUT pin.
- The DACR<4:0> range select bits are cleared.


© 2010-2016 Microchip Technology Inc.

## 17.7 Register Definitions: DAC Control

### REGISTER 17-1: DACCON0: VOLTAGE REFERENCE CONTROL REGISTER 0

| R/W-0/0 | R/W-0/0 | R/W-0/0 | U-0 | R/W-0/0 | R/W-0/0 | U-0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| DACEN | DACLPS | DACOE | — | DACPSS<1:0> | | — | DACNSS |
| 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 **DACEN:** DAC Enable bit
1 = DAC is enabled
0 = DAC is disabled

bit 6 **DACLPS:** DAC Low-Power Voltage State Select bit
1 = DAC Positive reference source selected
0 = DAC Negative reference source selected

bit 5 **DACOE:** DAC Voltage Output Enable bit
1 = DAC voltage level is also an output on the DACOUT pin
0 = DAC voltage level is disconnected from the DACOUT pin

bit 4 **Unimplemented:** Read as '0'

bit 3-2 **DACPSS<1:0>:** DAC Positive Source Select bits
00 = $V_{DD}$
01 = $V_{REF+}$ pin
10 = FVR Buffer2 output
11 = Reserved, do not use

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

bit 0 **DACNSS:** DAC Negative Source Select bits
1 = $V_{REF-}$
0 = $V_{SS}$

### REGISTER 17-2: DACCON1: VOLTAGE REFERENCE CONTROL REGISTER 1

| U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | DACR<4:0> | | | | |
| 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 **Unimplemented:** Read as '0'

bit 4-0 **DACR<4:0>:** DAC Voltage Output Select bits

### TABLE 17-1: SUMMARY OF REGISTERS ASSOCIATED WITH THE DAC MODULE

| 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 | CDAFVR<1:0> | | ADFVR1 | ADFVR0 | 151 |
| DACCON0 | DACEN | DACLPS | DACOE | — | DACPSS<1:0> | | — | DACNSS | 171 |
| DACCON1 | — | — | — | DACR<4:0> | | | | | 171 |

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

## 18.0 COMPARATOR MODULE

Comparators are used to interface analog circuits to a digital circuit by comparing two analog voltages and providing a digital indication of their relative magnitudes. Comparators are very useful mixed signal building blocks because they provide analog functionality independent of program execution. The analog comparator module includes the following features:

- Independent comparator control
- Programmable input selection
- Comparator output is available internally/externally
- Programmable output polarity
- Interrupt-on-change
- Wake-up from Sleep
- Programmable Speed/Power optimization
- PWM shutdown
- Programmable and fixed voltage reference

### 18.1 Comparator Overview

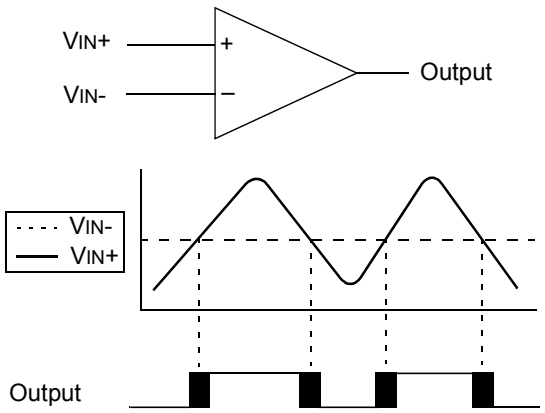
A single comparator is shown in Figure 18-1 along with the relationship between the analog input levels and the digital output. When the analog voltage at VIN+ is less than the analog voltage at VIN-, the output of the comparator is a digital low level. When the analog voltage at VIN+ is greater than the analog voltage at VIN-, the output of the comparator is a digital high level.

The comparators available for this device are located in Table 18-1.

**TABLE 18-1: COMPARATOR AVAILABILITY PER DEVICE**

| Device | C1 | C2 | C3 |
|---|---|---|---|
| PIC16(L)F1946 | • | • | • |
| PIC16(L)F1947 | • | • | • |

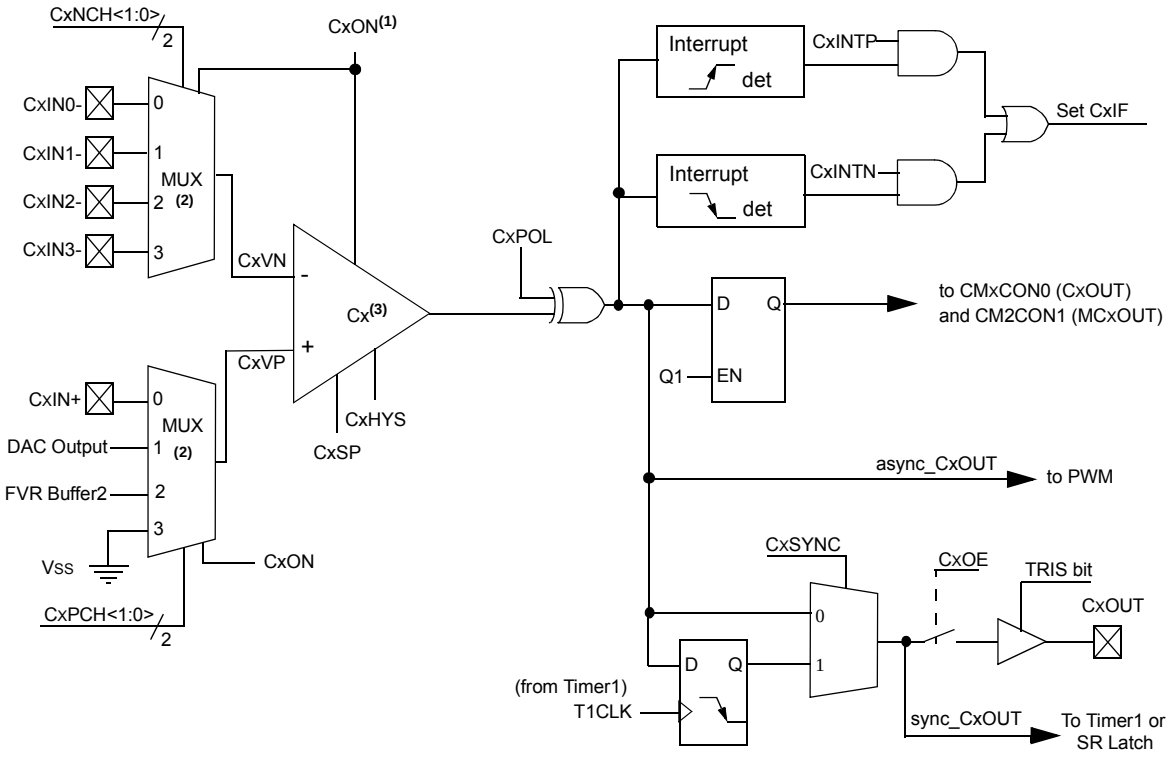
**FIGURE 18-1: SINGLE COMPARATOR**




**Note:** The black areas of the output of the comparator represents the uncertainty due to input offsets and response time.


© 2010-2016 Microchip Technology Inc.

**FIGURE 18-2: COMPARATOR MODULE SIMPLIFIED BLOCK DIAGRAM**




**Note 1:** When CxON = 0, the Comparator will produce a '0' at the output.
**2:** When CxON = 0, all multiplexer inputs are disconnected.
**3:** Output of comparator can be frozen during debugging.

## 18.2 Comparator Control

Each comparator has two control registers: CMxCON0 and CMxCON1.

The CMxCON0 registers (see Register 18-1) contain Control and Status bits for the following:

- Enable
- Output selection
- Output polarity
- Speed/Power selection
- Hysteresis enable
- Output synchronization

The CMxCON1 registers (see Register 18-2) contain Control bits for the following:

- Interrupt enable
- Interrupt edge polarity
- Positive input channel selection
- Negative input channel selection

### 18.2.1 COMPARATOR ENABLE

Setting the CxON bit of the CMxCON0 register enables the comparator for operation. Clearing the CxON bit disables the comparator resulting in minimum current consumption.

### 18.2.2 COMPARATOR OUTPUT SELECTION

The output of the comparator can be monitored by reading either the CxOUT bit of the CMxCON0 register or the MCxOUT bit of the CMOUT register. In order to make the output available for an external connection, the following conditions must be true:

- CxOE bit of the CMxCON0 register must be set
- Corresponding TRIS bit must be cleared
- CxON bit of the CMxCON0 register must be set

| | |
|---|---|
| **Note 1:** | The CxOE bit of the CMxCON0 register overrides the PORT data latch. Setting the CxON bit of the CMxCON0 register has no impact on the port override. |
| **2:** | The internal output of the comparator is latched with each instruction cycle. Unless otherwise specified, external outputs are not latched. |

### 18.2.3 COMPARATOR OUTPUT POLARITY

Inverting the output of the comparator is functionally equivalent to swapping the comparator inputs. The polarity of the comparator output can be inverted by setting the CxPOL bit of the CMxCON0 register. Clearing the CxPOL bit results in a noninverted output.

Table 18-2 shows the output state versus input conditions, including polarity control.

**TABLE 18-2: COMPARATOR OUTPUT STATE VS. INPUT CONDITIONS**

| Input Condition | CxPOL | CxOUT |
|---|---|---|
| CxVN > CxVP | 0 | 0 |
| CxVN < CxVP | 0 | 1 |
| CxVN > CxVP | 1 | 1 |
| CxVN < CxVP | 1 | 0 |

### 18.2.4 COMPARATOR SPEED/POWER SELECTION

The trade-off between speed or power can be optimized during program execution with the CxSP control bit. The default state for this bit is '1' which selects the normal speed mode. Device power consumption can be optimized at the cost of slower comparator propagation delay by clearing the CxSP bit to '0'.


© 2010-2016 Microchip Technology Inc.

## 18.3 Comparator Hysteresis

A selectable amount of separation voltage can be added to the input pins of each comparator to provide a hysteresis function to the overall operation. Hysteresis is enabled by setting the CxHYS bit of the CMxCON0 register.

See **Section 30.0 "Electrical Specifications"** for more information.

## 18.4 Timer1 Gate Operation

The output resulting from a comparator operation can be used as a source for gate control of Timer1. See **Section 21.6 "Timer1 Gate"** for more information. This feature is useful for timing the duration or interval of an analog event.

It is recommended that the comparator output be synchronized to Timer1. This ensures that Timer1 does not increment while a change in the comparator is occurring.

### 18.4.1 COMPARATOR OUTPUT SYNCHRONIZATION

The output from a comparator can be synchronized with Timer1 by setting the CxSYNC bit of the CMxCON0 register.

Once enabled, the comparator output is latched on the falling edge of the Timer1 source clock. If a prescaler is used with Timer1, the comparator output is latched after the prescaling function. To prevent a race condition, the comparator output is latched on the falling edge of the Timer1 clock source and Timer1 increments on the rising edge of its clock source. See the Comparator Block Diagram (Figure 18-2) and the Timer1 Block Diagram (Figure 21-1) for more information.

## 18.5 Comparator Interrupt

An interrupt can be generated upon a change in the output value of the comparator for each comparator, a rising edge detector and a falling edge detector are present.

When either edge detector is triggered and its associated enable bit is set (CxINTP and/or CxINTN bits of the CMxCON1 register), the Corresponding Interrupt Flag bit (CxIF bit of the PIR2 register) will be set.

To enable the interrupt, you must set the following bits:

- CxON, CxPOL and CxSP bits of the CMxCON0 register
- CxIE bit of the PIE2 register
- CxINTP bit of the CMxCON1 register (for a rising edge detection)
- CxINTN bit of the CMxCON1 register (for a falling edge detection)
- PEIE and GIE bits of the INTCON register

The associated interrupt flag bit, CxIF bit of the PIR2 register, must be cleared in software. If another edge is detected while this flag is being cleared, the flag will still be set at the end of the sequence.

| | |
|---|---|
| **Note:** | Although a comparator is disabled, an interrupt can be generated by changing the output polarity with the CxPOL bit of the CMxCON0 register, or by switching the comparator on or off with the CxON bit of the CMxCON0 register. |

## 18.6 Comparator Positive Input Selection

Configuring the CxPCH<1:0> bits of the CMxCON1 register directs an internal voltage reference or an analog pin to the non-inverting input of the comparator:

- CxIN+ analog pin
- DAC output
- FVR (Fixed Voltage Reference)
- VSS (Ground)

See **Section 14.0 "Fixed Voltage Reference (FVR)"** for more information on the Fixed Voltage Reference module.

See **Section 17.0 "Digital-to-Analog Converter (DAC) Module"** for more information on the DAC input signal.

Any time the comparator is disabled (CxON = 0), all comparator inputs are disabled.

## 18.7 Comparator Negative Input Selection

The CxNCH<1:0> bits of the CMxCON0 register direct one of four analog pins to the comparator inverting input.

| | |
|---|---|
| **Note:** | To use CxIN+ and CxINx- pins as analog input, the appropriate bits must be set in the ANSEL register and the corresponding TRIS bits must also be set to disable the output drivers. |

## 18.8 Comparator Response Time

The comparator output is indeterminate for a period of time after the change of an input source or the selection of a new reference voltage. This period is referred to as the response time. The response time of the comparator differs from the settling time of the voltage reference. Therefore, both of these times must be considered when determining the total response time to a comparator input change. See the Comparator and Voltage Reference Specifications in **Section 30.0 "Electrical Specifications"** for more details.

## 18.9 Interaction with ECCP Logic

The comparators can be used as general purpose comparators. Their outputs can be brought out to the CxOUT pins. When the ECCP Auto-Shutdown is active it can use one or both comparator signals. If auto-restart is also enabled, the comparators can be configured as a closed loop analog feedback to the ECCP, thereby, creating an analog controlled PWM.

| | |
|---|---|
| **Note:** | When the Comparator module is first initialized the output state is unknown. Upon initialization, the user should verify the output state of the comparator prior to relying on the result, primarily when using the result in connection with other peripheral features, such as the ECCP Auto-Shutdown mode. |

## 18.10 Analog Input Connection Considerations

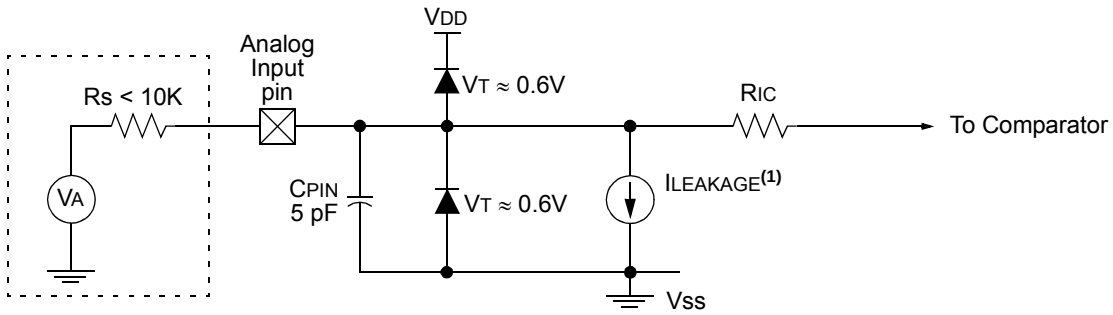
A simplified circuit for an analog input is shown in Figure 18-3. Since the analog input pins share their connection with a digital input, they have reverse biased ESD protection diodes to $V_{DD}$ and $V_{SS}$. The analog input, therefore, must be between $V_{SS}$ and $V_{DD}$. If the input voltage deviates from this range by more than 0.6V in either direction, one of the diodes is forward biased and a latch-up may occur.

A maximum source impedance of 10 kΩ is recommended for the analog sources. Also, any external component connected to an analog input pin, such as a capacitor or a Zener diode, should have very little leakage current to minimize inaccuracies introduced.

| | |
|---|---|
| **Note 1:** | When reading a PORT register, all pins configured as analog inputs will read as a '0'. Pins configured as digital inputs will convert as an analog input, according to the input specification. |
| **2:** | Analog levels on any pin defined as a digital input, may cause the input buffer to consume more current than is specified. |


© 2010-2016 Microchip Technology Inc.

**FIGURE 18-3: ANALOG INPUT MODEL**




**Legend:** CPIN = Input Capacitance
ILEAKAGE = Leakage Current at the pin due to various junctions
RIC = Interconnect Resistance
RS = Source Impedance
VA = Analog Voltage
VT = Threshold Voltage

**Note 1:** See **Section 30.0 “Electrical Specifications”**

## 18.11 Register Definitions: Comparator Control

**REGISTER 18-1: CMxCON0: COMPARATOR Cx CONTROL REGISTER 0**

| R/W-0/0 | R-0/0 | R/W-0/0 | R/W-0/0 | U-0 | R/W-1/1 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| CxON | CxOUT | CxOE | CxPOL | — | CxSP | CxHYS | CxSYNC |
| 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 **CxON:** Comparator Enable bit
1 = Comparator is enabled
0 = Comparator is disabled and consumes no active power

bit 6 **CxOUT:** Comparator Output bit
If CxPOL = 1 (inverted polarity):
1 = CxVP < CxVN
0 = CxVP > CxVN
If CxPOL = 0 (non-inverted polarity):
1 = CxVP > CxVN
0 = CxVP < CxVN

bit 5 **CxOE:** Comparator Output Enable bit
1 = CxOUT is present on the CxOUT pin. Requires that the associated TRIS bit be cleared to actually drive the pin. Not affected by CxON.
0 = CxOUT is internal only

bit 4 **CxPOL:** Comparator Output Polarity Select bit
1 = Comparator output is inverted
0 = Comparator output is not inverted

bit 3 **Unimplemented:** Read as ‘0’

bit 2 **CxSP:** Comparator Speed/Power Select bit
1 = Comparator operates in normal power, higher speed mode
0 = Comparator operates in low-power, low-speed mode

bit 1 **CxHYS:** Comparator Hysteresis Enable bit
1 = Comparator hysteresis enabled
0 = Comparator hysteresis disabled

bit 0 **CxSYNC:** Comparator Output Synchronous Mode bit
1 = Comparator output to Timer1 and I/O pin is synchronous to changes on Timer1 clock source. Output updated on the falling edge of Timer1 clock source.
0 = Comparator output to Timer1 and I/O pin is asynchronous.


© 2010-2016 Microchip Technology Inc.

**REGISTER 18-2: CMxCON1: COMPARATOR Cx 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 |
|---|---|---|---|---|---|---|---|
| CxINTP | CxINTN | CxPCH<1:0> | | — | — | CxNCH<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 **CxINTP:** Comparator Interrupt on Positive Going Edge Enable bits
1 = The CxIF interrupt flag will be set upon a positive going edge of the CxOUT bit
0 = No interrupt flag will be set on a positive going edge of the CxOUT bit

bit 6 **CxINTN:** Comparator Interrupt on Negative Going Edge Enable bits
1 = The CxIF interrupt flag will be set upon a negative going edge of the CxOUT bit
0 = No interrupt flag will be set on a negative going edge of the CxOUT bit

bit 5-4 **CxPCH<1:0>:** Comparator Positive Input Channel Select bits
11 = CxVP connects to Vss
10 = CxVP connects to FVR Voltage Reference
01 = CxVP connects to DAC Voltage Reference
00 = CxVP connects to CxIN+ pin

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

bit 1-0 **CxNCH<1:0>:** Comparator Negative Input Channel Select bits
11 = CxVN connects to CxIN3- pin
10 = CxVN connects to CxIN2- pin
01 = CxVN connects to CxIN1- pin
00 = CxVN connects to CxIN0- pin

**REGISTER 18-3: CMOUT: COMPARATOR OUTPUT REGISTER**

| U-0 | U-0 | U-0 | U-0 | U-0 | R-0/0 | R-0/0 | R-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | MC3OUT | MC2OUT | MC1OUT |
| 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 **MC3OUT:** Mirror Copy of C3OUT bit

bit 1 **MC2OUT:** Mirror Copy of C2OUT bit

bit 0 **MC1OUT:** Mirror Copy of C1OUT bit

**TABLE 18-3: SUMMARY OF REGISTERS ASSOCIATED WITH COMPARATOR MODULE**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| ANSELF | ANSF7 | ANSF6 | ANSF5 | ANSF4 | ANSF3 | ANSF2 | ANSF1 | ANSF0 | 141 |
| ANSELG | — | — | — | ANSG4 | ANSG3 | ANSG2 | ANSG1 | — | 144 |
| CM1CON0 | C1ON | C1OUT | C1OE | C1POL | — | C1SP | C1HYS | C1SYNC | 178 |
| CM2CON0 | C2ON | C2OUT | C2OE | C2POL | — | C2SP | C2HYS | C2SYNC | 178 |
| CM1CON1 | C1NTP | C1INTN | C1PCH<1:0> | | — | — | C1NCH<1:0> | | 179 |
| CM2CON1 | C2NTP | C2INTN | C2PCH<1:0> | | — | — | C2NCH<1:0> | | 179 |
| CM3CON0 | C3ON | C3OUT | C3OE | C3POL | — | C3SP | C3HYS | C3SYNC | 178 |
| CM3CON1 | C3INTP | C3INTN | C3PCH1 | C3PCH0 | — | — | C3NCH<1:0> | | 179 |
| CMOUT | — | — | — | — | — | MC3OUT | MC2OUT | MC1OUT | 179 |
| FVRCON | FVREN | FVRRDY | TSEN | TSRNG | CDAFVR<1:0> | | ADFVR<1:0> | | 151 |
| DACCON0 | DACEN | DACLPS | DACOE | — | DACPSS<1:0> | | — | DACNSS | 171 |
| DACCON1 | — | — | — | DACR<4:0> | | | | | 171 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| PIE2 | OSFIE | C2IE | C1IE | EEIE | BCLIE | LCDIE | C3IE | CCP2IE | 92 |
| PIR2 | OSFIF | C2IF | C1IF | EEIF | BCLIF | LCDIF | C3IF | CCP2IF | 96 |
| TRISF | TRISF7 | TRISF6 | TRISF5 | TRISF4 | TRISF3 | TRISF2 | TRISF1 | TRISF0 | 140 |
| TRISG | — | — | TRISG5 | TRISG4 | TRISG3 | TRISG2 | TRISG1 | TRISG0 | 143 |

**Legend:** — = unimplemented location, read as '0'. Shaded cells are unused by the comparator module.


© 2010-2016 Microchip Technology Inc.

## 19.0 SR LATCH

The module consists of a single SR Latch with multiple Set and Reset inputs as well as separate latch outputs. The SR Latch module includes the following features:

- Programmable input selection
- SR Latch output is available externally
- Separate Q and $\overline{Q}$ outputs
- Firmware Set and Reset

The SR Latch can be used in a variety of analog applications, including oscillator circuits, one-shot circuit, hysteretic controllers, and analog timing applications.

### 19.1 Latch Operation

The latch is a Set-Reset Latch that does not depend on a clock source. Each of the Set and Reset inputs are active-high. The latch can be set or reset by:

- Software control (SRPS and SRPR bits)
- Comparator C1 output (sync_C1OUT)
- Comparator C2 output (sync_C2OUT)
- SRI pin
- Programmable clock (SRCLK)

The SRPS and the SRPR bits of the SRCON0 register may be used to set or reset the SR Latch, respectively. The latch is Reset-dominant. Therefore, if both Set and Reset inputs are high, the latch will go to the Reset state. Both the SRPS and SRPR bits are self resetting which means that a single write to either of the bits is all that is necessary to complete a latch Set or Reset operation.

The output from Comparator C1 or C2 can be used as the Set or Reset inputs of the SR Latch. The output of either comparator can be synchronized to the Timer1 clock source. See **Section 18.0 "Comparator Module"** and **Section 21.0 "Timer1 Module with Gate Control"** for more information.

An external source on the SRI pin can be used as the Set or Reset inputs of the SR Latch.

An internal clock source is available that can periodically set or reset the SR Latch. The SRCLK<2:0> bits in the SRCON0 register are used to select the clock source period. The SRSCKE and SRRCKE bits of the SRCON1 register enable the clock source to set or reset the SR Latch, respectively.

### 19.2 Latch Output

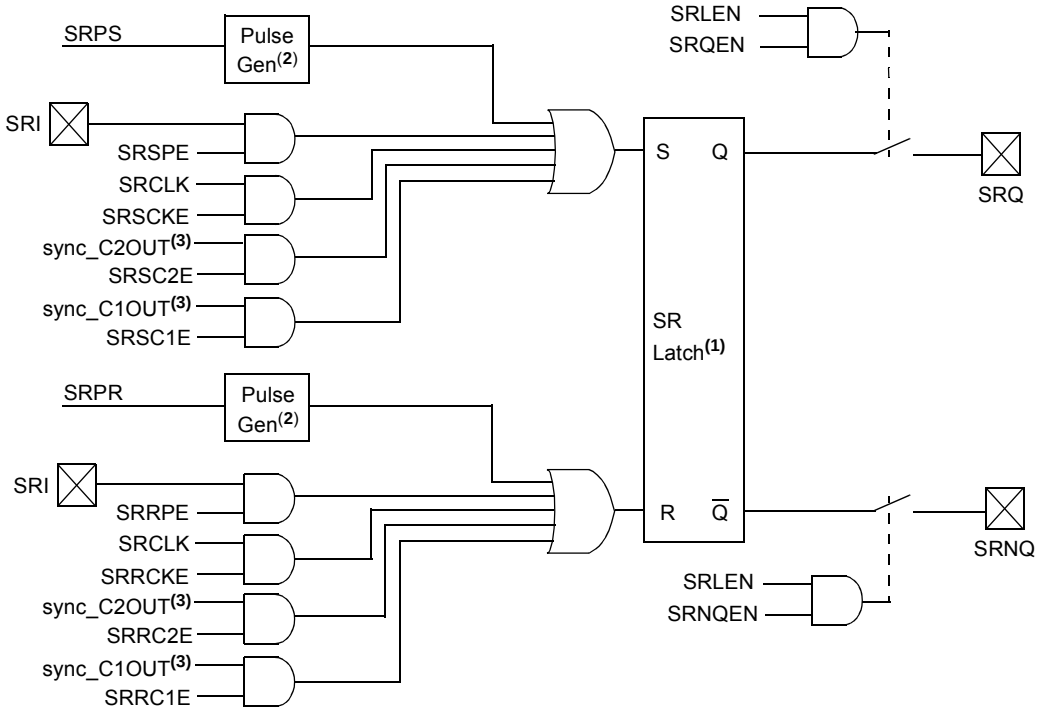
The SRQEN and SRNQEN bits of the SRCON0 register control the Q and $\overline{Q}$ latch outputs. Both of the SR Latch outputs may be directly output to an I/O pin at the same time.

The applicable TRIS bit of the corresponding port must be cleared to enable the port pin output driver.

### 19.3 Effects of a Reset

Upon any device Reset, the SR Latch output is not initialized to a known state. The user's firmware is responsible for initializing the latch output before enabling the output pins.

**FIGURE 19-1: SR LATCH SIMPLIFIED BLOCK DIAGRAM**




**Note 1:** If R = 1 and S = 1 simultaneously, Q = 0, $\overline{Q}$ = 1.
**2:** Pulse generator causes a 1 Q-state pulse width.
**3:** Name denotes the connection point at the comparator output.


© 2010-2016 Microchip Technology Inc.

**TABLE 19-1: SRCLK FREQUENCY TABLE**

| SRCLK | Divider | Fosc = 32 MHz | Fosc = 20 MHz | Fosc = 16 MHz | Fosc = 4 MHz | Fosc = 1 MHz |
|---|---|---|---|---|---|---|
| 111 | 512 | 62.5 kHz | 39.0 kHz | 31.3 kHz | 7.81 kHz | 1.95 kHz |
| 110 | 256 | 125 kHz | 78.1 kHz | 62.5 kHz | 15.6 kHz | 3.90 kHz |
| 101 | 128 | 250 kHz | 156 kHz | 125 kHz | 31.25 kHz | 7.81 kHz |
| 100 | 64 | 500 kHz | 313 kHz | 250 kHz | 62.5 kHz | 15.6 kHz |
| 011 | 32 | 1 MHz | 625 kHz | 500 kHz | 125 kHz | 31.3 kHz |
| 010 | 16 | 2 MHz | 1.25 MHz | 1 MHz | 250 kHz | 62.5 kHz |
| 001 | 8 | 4 MHz | 2.5 MHz | 2 MHz | 500 kHz | 125 kHz |
| 000 | 4 | 8 MHz | 5 MHz | 4 MHz | 1 MHz | 250 kHz |

© 2010-2016 Microchip Technology Inc.

## 19.4 Register Definitions: SR Latch Control

**REGISTER 19-2: SRCON0: SR LATCH CONTROL 0 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/S-0/0 | R/S-0/0 |
|---|---|---|---|---|---|---|---|
| SRLEN | SRCLK<2:0> | | | SRQEN | SRNQEN | SRPS | SRPR |
| 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 | S = Bit is set only |

bit 7 **SRLEN:** SR Latch Enable bit
1 = SR Latch is enabled
0 = SR Latch is disabled

bit 6-4 **SRCLK<2:0>:** SR Latch Clock Divider bits
111 = Generates a 1 FOSC wide pulse every 512th FOSC cycle clock
110 = Generates a 1 FOSC wide pulse every 256th FOSC cycle clock
101 = Generates a 1 FOSC wide pulse every 128th FOSC cycle clock
100 = Generates a 1 FOSC wide pulse every 64th FOSC cycle clock
011 = Generates a 1 FOSC wide pulse every 32nd FOSC cycle clock
010 = Generates a 1 FOSC wide pulse every 16th FOSC cycle clock
001 = Generates a 1 FOSC wide pulse every 8th FOSC cycle clock
000 = Generates a 1 FOSC wide pulse every 4th FOSC cycle clock

bit 3 **SRQEN:** SR Latch Q Output Enable bit
If SRLEN = 1:
1 = Q is present on the SRQ pin
0 = External Q output is disabled
If SRLEN = 0:
SR Latch is disabled

bit 2 **SRNQEN:** SR Latch $\overline{Q}$ Output Enable bit
If SRLEN = 1:
1 = $\overline{Q}$ is present on the SRnQ pin
0 = External $\overline{Q}$ output is disabled
If SRLEN = 0:
SR Latch is disabled

bit 1 **SRPS:** Pulse Set Input of the SR Latch bit(1)
1 = Pulse set input for 1 Q-clock period
0 = No effect on set input

bit 0 **SRPR:** Pulse Reset Input of the SR Latch bit(1)
1 = Pulse Reset input for 1 Q-clock period
0 = No effect on Reset input

**Note 1:** Set only, always reads back '0'.


© 2010-2016 Microchip Technology Inc.

**REGISTER 19-3: SRCON1: SR LATCH CONTROL 1 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 |
|---|---|---|---|---|---|---|---|
| SRSPE | SRSCKE | SRSC2E | SRSC1E | SRRPE | SRRCKE | SRRC2E | SRRC1E |
| 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 **SRSPE:** SR Latch Peripheral Set Enable bit
1 = SR Latch is set when the SRI pin is high
0 = SRI pin has no effect on the set input of the SR Latch

bit 6 **SRSCKE:** SR Latch Set Clock Enable bit
1 = Set input of SR Latch is pulsed with SRCLK
0 = SRCLK has no effect on the set input of the SR Latch

bit 5 **SRSC2E:** SR Latch C2 Set Enable bit
1 = SR Latch is set when the C2 Comparator output is high
0 = C2 Comparator output has no effect on the set input of the SR Latch

bit 4 **SRSC1E:** SR Latch C1 Set Enable bit
1 = SR Latch is set when the C1 Comparator output is high
0 = C1 Comparator output has no effect on the set input of the SR Latch

bit 3 **SRRPE:** SR Latch Peripheral Reset Enable bit
1 = SR Latch is reset when the SRI pin is high
0 = SRI pin has no effect on the Reset input of the SR Latch

bit 2 **SRRCKE:** SR Latch Reset Clock Enable bit
1 = Reset input of SR Latch is pulsed with SRCLK
0 = SRCLK has no effect on the Reset input of the SR Latch

bit 1 **SRRC2E:** SR Latch C2 Reset Enable bit
1 = SR Latch is reset when the C2 Comparator output is high
0 = C2 Comparator output has no effect on the Reset input of the SR Latch

bit 0 **SRRC1E:** SR Latch C1 Reset Enable bit
1 = SR Latch is reset when the C1 Comparator output is high
0 = C1 Comparator output has no effect on the Reset input of the SR Latch

**TABLE 19-2: SUMMARY OF REGISTERS ASSOCIATED WITH SR LATCH MODULE**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| ANSELF | ANSELF7 | ANSELF6 | ANSELF5 | ANSELF4 | ANSELF3 | ANSELF2 | ANSELF1 | ANSELF0 | 126 |
| SRCON0 | SRLEN | SRCLK<2:0> | | | SRQEN | SRNQEN | SRPS | SRPR | 184 |
| SRCON1 | SRSPE | SRSCKE | SRSC2E | SRSC1E | SRRPE | SRRCKE | SRRC2E | SRRC1E | 185 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 125 |
| TRISF | TRISF7 | TRISF6 | TRISF5 | TRISF4 | TRISF3 | TRISF2 | TRISF1 | TRISF0 | 125 |

**Legend:** — = unimplemented location, read as '0'. Shaded cells are unused by the SR Latch module.

## 20.0 TIMER0 MODULE

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

- 8-bit timer/counter register (TMR0)
- 8-bit prescaler (independent of Watchdog Timer)
- Programmable internal or external clock source
- Programmable external clock edge selection
- Interrupt on overflow
- TMR0 can be used to gate Timer1

Figure 20-1 is a block diagram of the Timer0 module.

## 20.1 Timer0 Operation

The Timer0 module can be used as either an 8-bit timer or an 8-bit counter.

### 20.1.1 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. |
|---|---|

### 20.1.2 8-BIT COUNTER MODE

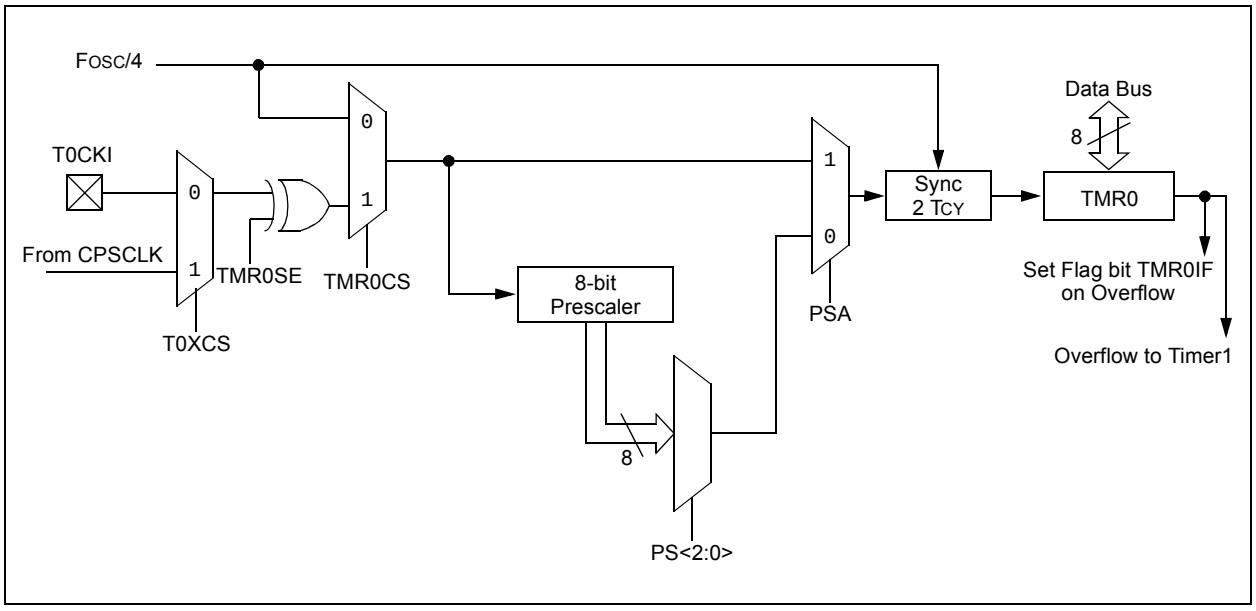
In 8-Bit Counter mode, the Timer0 module will increment on every rising or falling edge of the T0CKI pin or the Capacitive Sensing Oscillator (CPSCLK) signal.

8-Bit Counter mode using the T0CKI pin is selected by setting the TMR0CS bit in the OPTION_REG register to '1' and resetting the T0XCS bit in the CPSCON0 register to '0'.

8-Bit Counter mode using the Capacitive Sensing Oscillator (CPSCLK) signal is selected by setting the TMR0CS bit in the OPTION_REG register to '1' and setting the T0XCS bit in the CPSCON0 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 20-1: BLOCK DIAGRAM OF THE TIMER0**





© 2010-2016 Microchip Technology Inc.

### 20.1.3 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.

### 20.1.4 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. |

### 20.1.5 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 30.0 "Electrical Specifications"**.

### 20.1.6 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.

## 20.2 Register Definitions: Option Register

**REGISTER 20-1: 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 20-1: 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 |
|---|---|---|---|---|---|---|---|---|---|
| CPSCON0 | CPSON | CPSRM | — | — | CPSRNG<1:0> | | CPSOUT | T0XCS | 322 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| OPTION_REG | $\overline{\text{WPUEN}}$ | INTEDG | TMR0CS | TMR0SE | PSA | PS<2:0> | | | 188 |
| TMR0 | Timer0 Module Register | | | | | | | | 186* |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 125 |

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


© 2010-2016 Microchip Technology Inc.

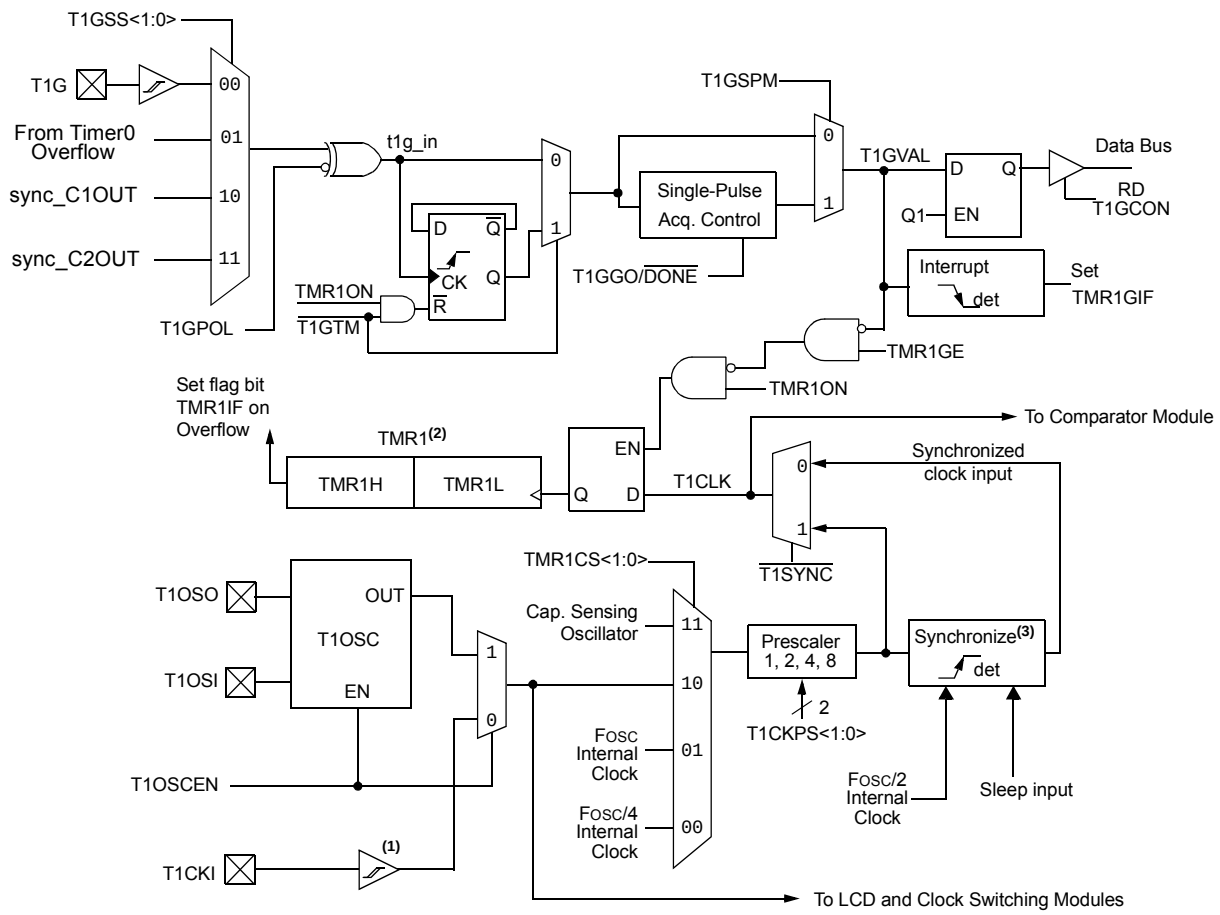
## 21.0 TIMER1 MODULE WITH GATE CONTROL

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

- 16-bit timer/counter register pair (TMR1H:TMR1L)
- Programmable internal or external clock source
- 2-bit prescaler
- Dedicated 32 kHz oscillator circuit
- Optionally synchronized comparator out
- Multiple Timer1 gate (count enable) sources
- Interrupt on overflow
- Wake-up on overflow (external clock, Asynchronous mode only)
- Time base for the Capture/Compare function
- Special Event Trigger (with CCP/ECCP)
- Selectable Gate Source Polarity
- Gate Toggle mode
- Gate Single-Pulse mode
- Gate Value Status
- Gate Event Interrupt

Figure 21-1 is a block diagram of the Timer1 module.

**FIGURE 21-1: TIMER1 BLOCK DIAGRAM**




**Note 1:** ST Buffer is high speed type when using T1CKI.
**2:** Timer1 register increments on rising edge.
**3:** Synchronize does not operate while in Sleep.

## 21.1 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 21-1 displays the Timer1 enable selections.

**TABLE 21-1: TIMER1 ENABLE SELECTIONS**

| TMR1ON | TMR1GE | Timer1 Operation |
|---|---|---|
| 0 | 0 | Off |
| 0 | 1 | Off |
| 1 | 0 | Always On |
| 1 | 1 | Count Enabled |

## 21.2 Clock Source Selection

The TMR1CS<1:0> and T1OSCEN bits of the T1CON register are used to select the clock source for Timer1. Table 21-2 displays the clock source selections.

### 21.2.1 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

### 21.2.2 EXTERNAL CLOCK SOURCE

When the external clock source is selected, the Timer1 module may work as a timer or a counter.

When enabled to count, Timer1 is incremented on the rising edge of the external clock input T1CKI or the capacitive sensing oscillator signal. Either of these external clock sources can be synchronized to the microcontroller system clock or they can run asynchronously.

When used as a timer with a clock oscillator, an external 32.768 kHz crystal can be used in conjunction with the dedicated internal oscillator circuit.

**Note:** In Counter mode, a falling edge must be registered by the counter prior to the first incrementing rising edge after any one or more of the following conditions:

- Timer1 enabled after POR
- Write to TMR1H or TMR1L
- Timer1 is disabled
- Timer1 is disabled (TMR1ON = 0) when T1CKI is high then Timer1 is enabled (TMR1ON=1) when T1CKI is low.

**TABLE 21-2: CLOCK SOURCE SELECTIONS**

| TMR1CS1 | TMR1CS0 | T1OSCEN | Clock Source |
|---|---|---|---|
| 0 | 1 | x | System Clock (FOSC) |
| 0 | 0 | x | Instruction Clock (FOSC/4) |
| 1 | 1 | x | Capacitive Sensing Oscillator |
| 1 | 0 | 0 | External Clocking on T1CKI Pin |
| 1 | 0 | 1 | Osc.Circuit On T1OSI/T1OSO Pins |


© 2010-2016 Microchip Technology Inc.

## 21.3 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.

## 21.4 Timer1 Oscillator

A dedicated low-power 32.768 kHz oscillator circuit is built-in between pins T1OSI (input) and T1OSO (amplifier output). This internal circuit is to be used in conjunction with an external 32.768 kHz crystal.

The oscillator circuit is enabled by setting the T1OSCEN bit of the T1CON register. The oscillator will continue to run during Sleep.

| | |
|---|---|
| **Note:** | The oscillator requires a start-up and stabilization time before use. Thus, T1OSCEN should be set and a suitable delay observed prior to using Timer1. A suitable delay, similar to the OST delay can be implemented in software by clearing the TMR1IF bit, then presetting the TMR1H:TMR1L register pair to FC00h. The TMR1IF flag will be set when 1024 clock cycles have elapsed, thereby indicating that the oscillator is running and is reasonably stable. |

## 21.5 Timer1 Operation in Asynchronous Counter Mode

If control bit $\overline{\text{T1SYNC}}$ of the T1CON register is set, the external clock input is not synchronized. The timer increments asynchronously to the internal phase clocks. If the external clock source is selected then the timer will continue to run during Sleep and can generate an interrupt on overflow, which will wake-up the processor. However, special precautions in software are needed to read/write the timer (see **Section 21.5.1 "Reading and Writing Timer1 in Asynchronous Counter Mode"**).

| | |
|---|---|
| **Note:** | When switching from synchronous to asynchronous operation, it is possible to skip an increment. When switching from asynchronous to synchronous operation, it is possible to produce an additional increment. |

### 21.5.1 READING AND WRITING TIMER1 IN ASYNCHRONOUS COUNTER MODE

Reading TMR1H or TMR1L while the timer is running from an external asynchronous clock will ensure a valid read (taken care of in hardware). However, the user should keep in mind that reading the 16-bit timer in two 8-bit values itself, poses certain problems, since the timer may overflow between the reads.

For writes, it is recommended that the user simply stop the timer and write the desired values. A write contention may occur by writing to the timer registers, while the register is incrementing. This may produce an unpredictable value in the TMR1H:TMR1L register pair.

## 21.6 Timer1 Gate

Timer1 can be configured to count freely or the count can be enabled and disabled using Timer1 gate circuitry. This is also referred to as Timer1 Gate Enable.

Timer1 gate can also be driven by multiple selectable sources.

### 21.6.1 TIMER1 GATE ENABLE

The Timer1 Gate Enable mode is enabled by setting the TMR1GE bit of the T1GCON register. The polarity of the Timer1 Gate Enable mode is configured using the T1GPOL bit of the T1GCON register.

When Timer1 Gate Enable mode is enabled, Timer1 will increment on the rising edge of the Timer1 clock source. When Timer1 Gate Enable mode is disabled, no incrementing will occur and Timer1 will hold the current count. See Figure 21-3 for timing details.

**TABLE 21-3: TIMER1 GATE ENABLE SELECTIONS**

| T1CLK | T1GPOL | T1G | Timer1 Operation |
|---|---|---|---|
| ↑ | 0 | 0 | Counts |
| ↑ | 0 | 1 | Holds Count |
| ↑ | 1 | 0 | Holds Count |
| ↑ | 1 | 1 | Counts |

### 21.6.2 TIMER1 GATE SOURCE SELECTION

Timer1 gate source selections are shown in Table 21-4. Source selection is controlled by the T1GSS bits of the T1GCON register. The polarity for each available source is also selectable. Polarity selection is controlled by the T1GPOL bit of the T1GCON register.

**TABLE 21-4: TIMER1 GATE SOURCES**

| T1GSS | Timer1 Gate Source |
|---|---|
| 00 | Timer1 Gate Pin |
| 01 | Overflow of Timer0 (TMR0 increments from FFh to 00h) |
| 10 | Comparator 1 Output sync_C1OUT (optionally Timer1 synchronized output) |
| 11 | Comparator 2 Output sync_C2OUT (optionally Timer1 synchronized output) |

#### 21.6.2.1 T1G Pin Gate Operation

The T1G pin is one source for Timer1 gate control. It can be used to supply an external source to the Timer1 gate circuitry.

#### 21.6.2.2 Timer0 Overflow Gate Operation

When Timer0 increments from FFh to 00h, a low-to-high pulse will automatically be generated and internally supplied to the Timer1 gate circuitry.

#### 21.6.2.3 Comparator C1 Gate Operation

The output resulting from a Comparator 1 operation can be selected as a source for Timer1 gate control. The Comparator 1 output (sync_C1OUT) can be synchronized to the Timer1 clock or left asynchronous. For more information see **Section 18.4.1 "Comparator Output Synchronization"**.

#### 21.6.2.4 Comparator C2 Gate Operation

The output resulting from a Comparator 2 operation can be selected as a source for Timer1 gate control. The Comparator 2 output (sync_C2OUT) can be synchronized to the Timer1 clock or left asynchronous. For more information see **Section 18.4.1 "Comparator Output Synchronization"**.

### 21.6.3 TIMER1 GATE TOGGLE MODE

When Timer1 Gate Toggle mode is enabled, it is possible to measure the full-cycle length of a Timer1 gate signal, as opposed to the duration of a single level pulse.

The Timer1 gate source is routed through a flip-flop that changes state on every incrementing edge of the signal. See Figure 21-4 for timing details.

Timer1 Gate Toggle mode is enabled by setting the T1GTM bit of the T1GCON register. When the T1GTM bit is cleared, the flip-flop is cleared and held clear. This is necessary in order to control which edge is measured.

| **Note:** | Enabling Toggle mode at the same time as changing the gate polarity may result in indeterminate operation. |
|---|---|

### 21.6.4 TIMER1 GATE SINGLE-PULSE MODE

When Timer1 Gate Single-Pulse mode is enabled, it is possible to capture a single-pulse gate event. Timer1 Gate Single-Pulse mode is first enabled by setting the T1GSPM bit in the T1GCON register. Next, the T1GGO/$\overline{\text{DONE}}$ bit in the T1GCON register must be set. The Timer1 will be fully enabled on the next incrementing edge. On the next trailing edge of the pulse, the T1GGO/$\overline{\text{DONE}}$ bit will automatically be cleared. No other gate events will be allowed to increment Timer1 until the T1GGO/$\overline{\text{DONE}}$ bit is once again set in software. See Figure 21-5 for timing details.

If the Single-Pulse Gate mode is disabled by clearing the T1GSPM bit in the T1GCON register, the T1GGO/$\overline{\text{DONE}}$ bit should also be cleared.

Enabling the Toggle mode and the Single-Pulse mode simultaneously will permit both sections to work together. This allows the cycle times on the Timer1 gate source to be measured. See Figure 21-6 for timing details.

### 21.6.5 TIMER1 GATE VALUE STATUS

When Timer1 Gate Value Status is utilized, it is possible to read the most current level of the gate control value. The value is stored in the T1GVAL bit in the T1GCON register. The T1GVAL bit is valid even when the Timer1 gate is not enabled (TMR1GE bit is cleared).

### 21.6.6 TIMER1 GATE EVENT INTERRUPT

When Timer1 Gate Event Interrupt is enabled, it is possible to generate an interrupt upon the completion of a gate event. When the falling edge of T1GVAL occurs, the TMR1GIF flag bit in the PIR1 register will be set. If the TMR1GIE bit in the PIE1 register is set, then an interrupt will be recognized.

The TMR1GIF flag bit operates even when the Timer1 gate is not enabled (TMR1GE bit is cleared).


© 2010-2016 Microchip Technology Inc.

## 21.7 Timer1 Interrupt

The Timer1 register pair (TMR1H:TMR1L) increments to FFFFh and rolls over to 0000h. When Timer1 rolls over, the Timer1 interrupt flag bit of the PIR1 register is set. To enable the interrupt on rollover, you must set these bits:

- TMR1ON bit of the T1CON register
- TMR1IE bit of the PIE1 register
- PEIE bit of the INTCON register
- GIE bit of the INTCON register

The interrupt is cleared by clearing the TMR1IF bit in the Interrupt Service Routine.

| **Note:** | The TMR1H:TMR1L register pair and the TMR1IF bit should be cleared before enabling interrupts. |
|---|---|

## 21.8 Timer1 Operation During Sleep

Timer1 can only operate during Sleep when setup in Asynchronous Counter mode. In this mode, an external crystal or clock source can be used to increment the counter. To set up the timer to wake the device:

- TMR1ON bit of the T1CON register must be set
- TMR1IE bit of the PIE1 register must be set
- PEIE bit of the INTCON register must be set
- $\overline{\text{T1SYNC}}$ bit of the T1CON register must be set
- TMR1CS bits of the T1CON register must be configured
- T1OSCEN bit of the T1CON register must be configured

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

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

## 21.9 ECCP/CCP Capture/Compare Time Base

The CCP modules use the TMR1H:TMR1L register pair as the time base when operating in Capture or Compare mode.

In Capture mode, the value in the TMR1H:TMR1L register pair is copied into the CCPR1H:CCPR1L register pair on a configured event.

In Compare mode, an event is triggered when the value CCPR1H:CCPR1L register pair matches the value in the TMR1H:TMR1L register pair. This event can be a Special Event Trigger.

For more information, see **Section 23.0 "Capture/Compare/PWM Modules"**.

## 21.10 ECCP/CCP Special Event Trigger

When any of the CCP's are configured to trigger a special event, the trigger will clear the TMR1H:TMR1L register pair. This special event does not cause a Timer1 interrupt. The CCP module may still be configured to generate a CCP interrupt.

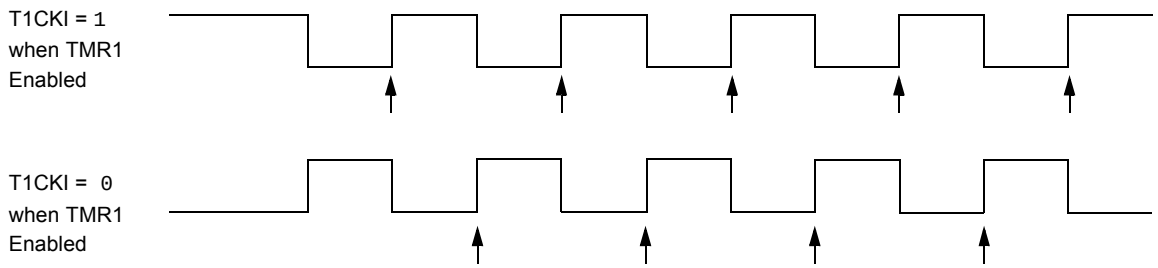
In this mode of operation, the CCPR1H:CCPR1L register pair becomes the period register for Timer1.

Timer1 should be synchronized and FOSC/4 should be selected as the clock source in order to utilize the Special Event Trigger. Asynchronous operation of Timer1 can cause a Special Event Trigger to be missed.

In the event that a write to TMR1H or TMR1L coincides with a Special Event Trigger from the CCP, the write will take precedence.

For more information, see **Section 16.3.1 "Special Event Trigger"**.

**FIGURE 21-2: TIMER1 INCREMENTING EDGE**




**Note 1:** Arrows indicate counter increments.

**2:** In Counter mode, a falling edge must be registered by the counter prior to the first incrementing rising edge of the clock.

**FIGURE 21-3: TIMER1 GATE ENABLE MODE**

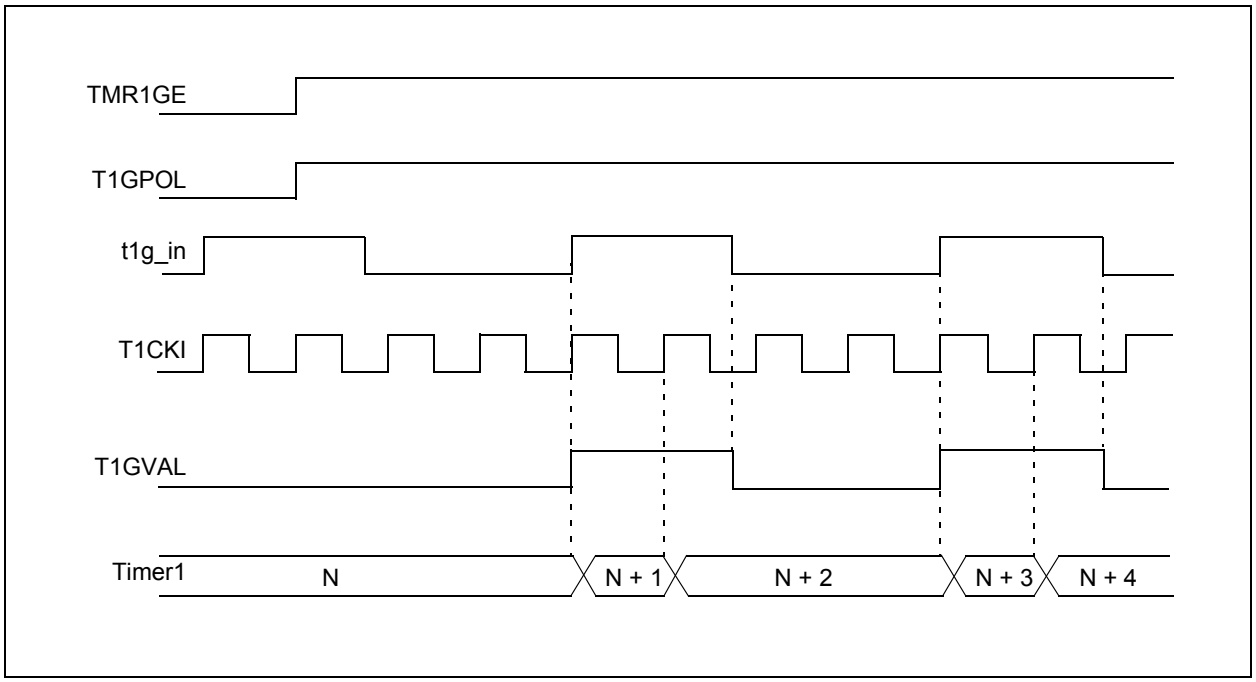
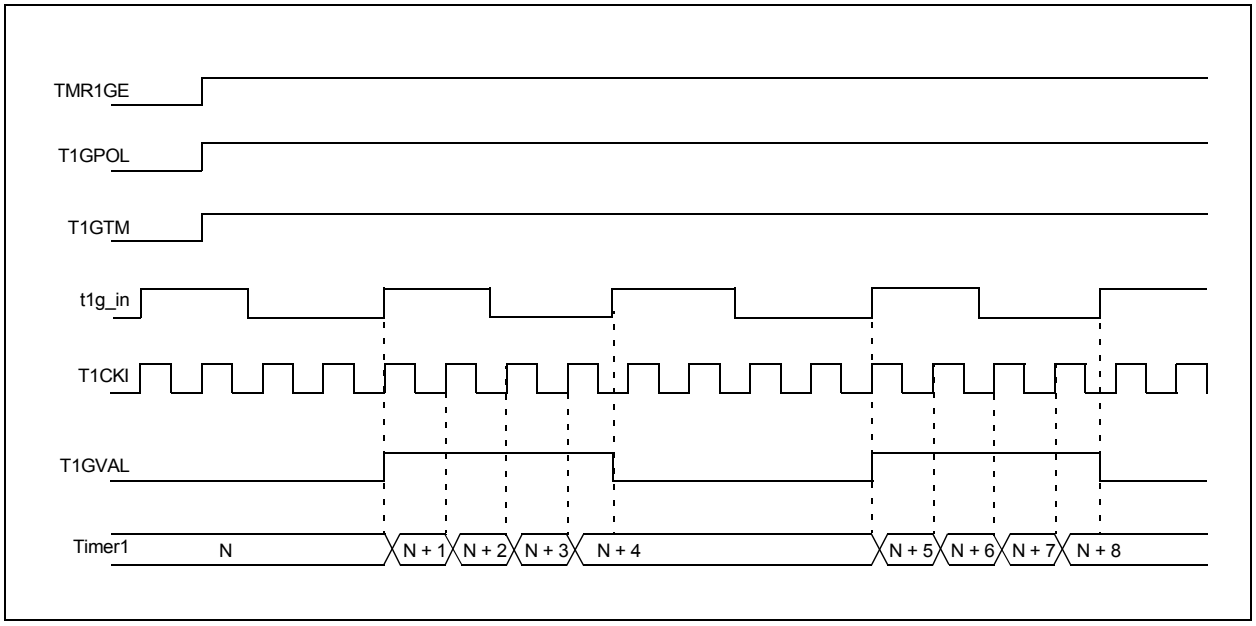



**FIGURE 21-4: TIMER1 GATE TOGGLE MODE**





© 2010-2016 Microchip Technology Inc.

**FIGURE 21-5: TIMER1 GATE SINGLE-PULSE MODE**

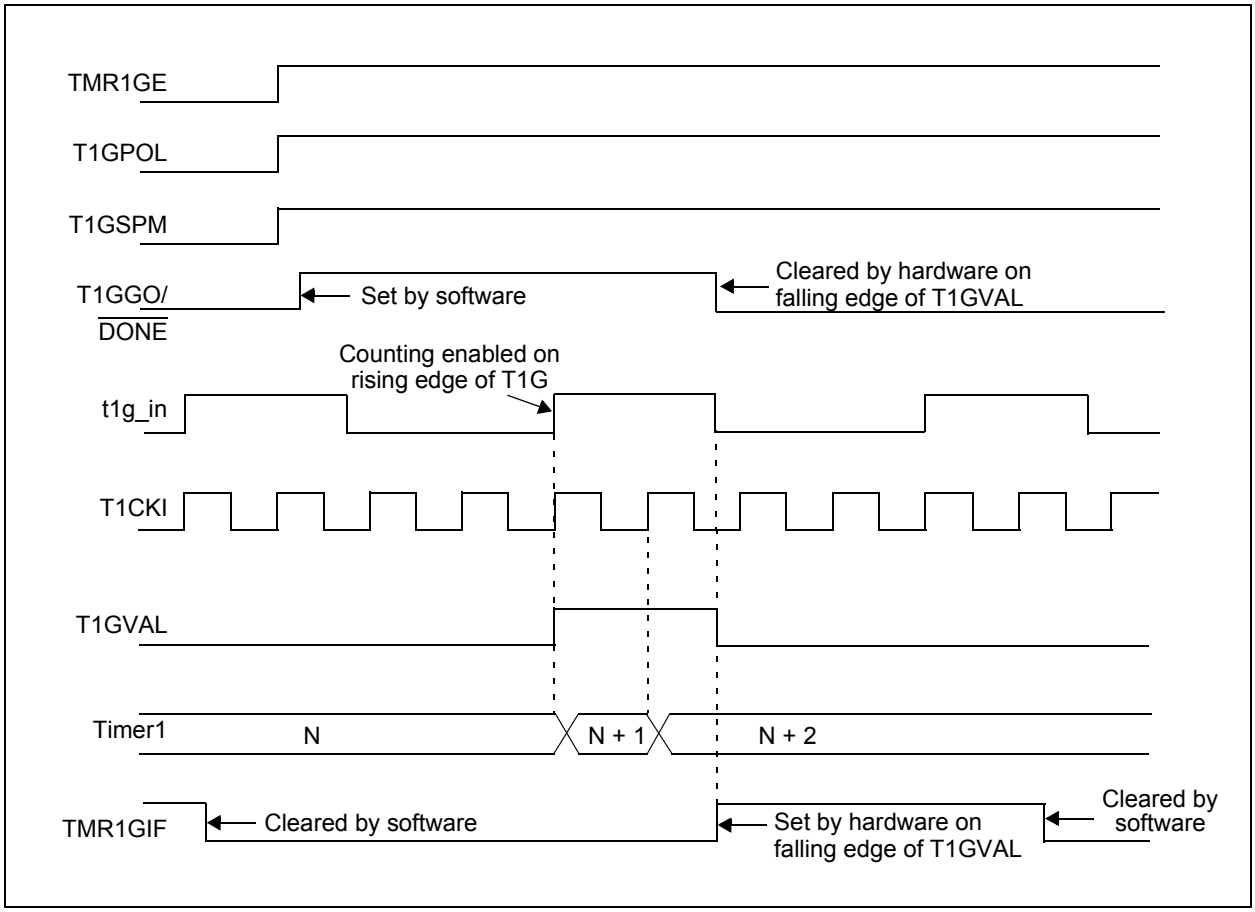
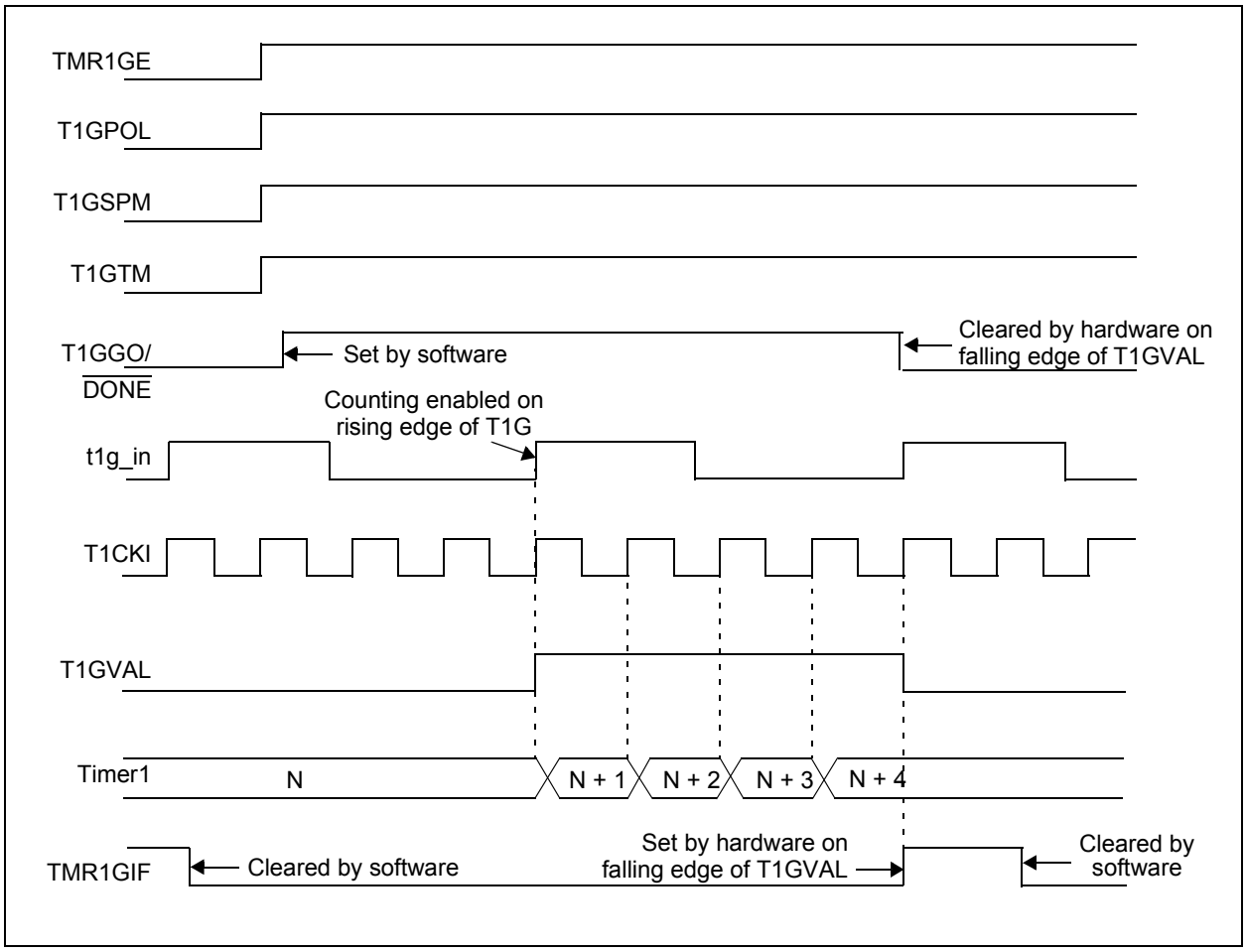


**FIGURE 21-6: TIMER1 GATE SINGLE-PULSE AND TOGGLE COMBINED MODE**





© 2010-2016 Microchip Technology Inc.

## 21.11 Register Definitions: Timer1 Control

**REGISTER 21-1: T1CON: TIMER1 CONTROL REGISTER**

| R/W-0/u | R/W-0/u | R/W-0/u | R/W-0/u | R/W-0/u | R/W-0/u | U-0 | R/W-0/u |
|---|---|---|---|---|---|---|---|
| TMR1CS<1:0> | | T1CKPS<1:0> | | T1OSCEN | $\overline{\text{T1SYNC}}$ | — | TMR1ON |
| bit 7 | | | | | | | bit 0 |

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

bit 7-6 **TMR1CS<1:0>:** Timer1 Clock Source Select bits
- 11 = Timer1 clock source is Capacitive Sensing Oscillator (CAPOSC)
- 10 = Timer1 clock source is pin or oscillator:
  - If T1OSCEN = 0:
  - External clock from T1CKI pin (on the rising edge)
  - If T1OSCEN = 1:
  - Crystal oscillator on T1OSI/T1OSO pins
- 01 = Timer1 clock source is system clock (FOSC)
- 00 = Timer1 clock source is instruction clock (FOSC/4)

bit 5-4 **T1CKPS<1:0>:** Timer1 Input Clock Prescale Select bits
- 11 = 1:8 Prescale value
- 10 = 1:4 Prescale value
- 01 = 1:2 Prescale value
- 00 = 1:1 Prescale value

bit 3 **T1OSCEN:** LP Oscillator Enable Control bit
- 1 = Dedicated Timer1 oscillator circuit enabled
- 0 = Dedicated Timer1 oscillator circuit disabled

bit 2 **$\overline{\text{T1SYNC}}$:** Timer1 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 21-2: T1GCON: TIMER1 GATE CONTROL REGISTER**

| R/W-0/u | R/W-0/u | R/W-0/u | R/W-0/u | R/W/HC-0/u | R-x/x | R/W-0/u | R/W-0/u |
|---|---|---|---|---|---|---|---|
| TMR1GE | T1GPOL | T1GTM | T1GSPM | T1GGO/$\overline{\text{DONE}}$ | T1GVAL | T1GSS<1:0> | |
| bit 7 | | | | | | | bit 0 |

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

bit 7 **TMR1GE:** Timer1 Gate Enable bit

If TMR1ON = 0:
This bit is ignored
If TMR1ON = 1:
1 = Timer1 counting is controlled by the Timer1 gate function
0 = Timer1 counts regardless of Timer1 gate function

bit 6 **T1GPOL:** Timer1 Gate Polarity bit

1 = Timer1 gate is active-high (Timer1 counts when gate is high)
0 = Timer1 gate is active-low (Timer1 counts when gate is low)

bit 5 **T1GTM:** Timer1 Gate Toggle Mode bit

1 = Timer1 Gate Toggle mode is enabled
0 = Timer1 Gate Toggle mode is disabled and toggle flip-flop is cleared
Timer1 gate flip-flop toggles on every rising edge.

bit 4 **T1GSPM:** Timer1 Gate Single-Pulse Mode bit

1 = Timer1 Gate Single-Pulse mode is enabled and is controlling Timer1 gate
0 = Timer1 Gate Single-Pulse mode is disabled

bit 3 **T1GGO/$\overline{\text{DONE}}$:** Timer1 Gate Single-Pulse Acquisition Status bit

1 = Timer1 gate single-pulse acquisition is ready, waiting for an edge
0 = Timer1 gate single-pulse acquisition has completed or has not been started

bit 2 **T1GVAL:** Timer1 Gate Current State bit

Indicates the current state of the Timer1 gate that could be provided to TMR1H:TMR1L.
Unaffected by Timer1 Gate Enable (TMR1GE).

bit 1-0 **T1GSS<1:0>:** Timer1 Gate Source Select bits

11 = Comparator 2 optionally synchronized output (sync_C2OUT)
10 = Comparator 1 optionally synchronized output (sync_C1OUT)
01 = Timer0 overflow output
00 = Timer1 gate pin


© 2010-2016 Microchip Technology Inc.

**TABLE 21-5: 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 |
|---|---|---|---|---|---|---|---|---|---|
| CCP1CON | P1M<1:0> | | DC1B<1:0> | | CCP1M<3:0> | | | | 227 |
| CCP2CON | P2M<1:0> | | DC2B<1:0> | | CCP2M<3:0> | | | | 227 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 91 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 95 |
| TMR1H | Holding Register for the Most Significant Byte of the 16-bit TMR1 Register | | | | | | | | 193* |
| TMR1L | Holding Register for the Least Significant Byte of the 16-bit TMR1 Register | | | | | | | | 193* |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 128 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 131 |
| T1CON | TMR1CS<1:0> | | T1CKPS<1:0> | | T1OSCEN | $\overline{\text{T1SYNC}}$ | — | TMR1ON | 197 |
| T1GCON | TMR1GE | T1GPOL | T1GTM | T1GSPM | T1GGO/ $\overline{\text{DONE}}$ | T1GVAL | T1GSS<1:0> | | 198 |

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

* Page provides register information.

## 22.0 TIMER2/4/6 MODULES

There are up to three identical Timer2-type modules available. To maintain pre-existing naming conventions, the Timers are called Timer2, Timer4 and Timer6 (also Timer2/4/6).

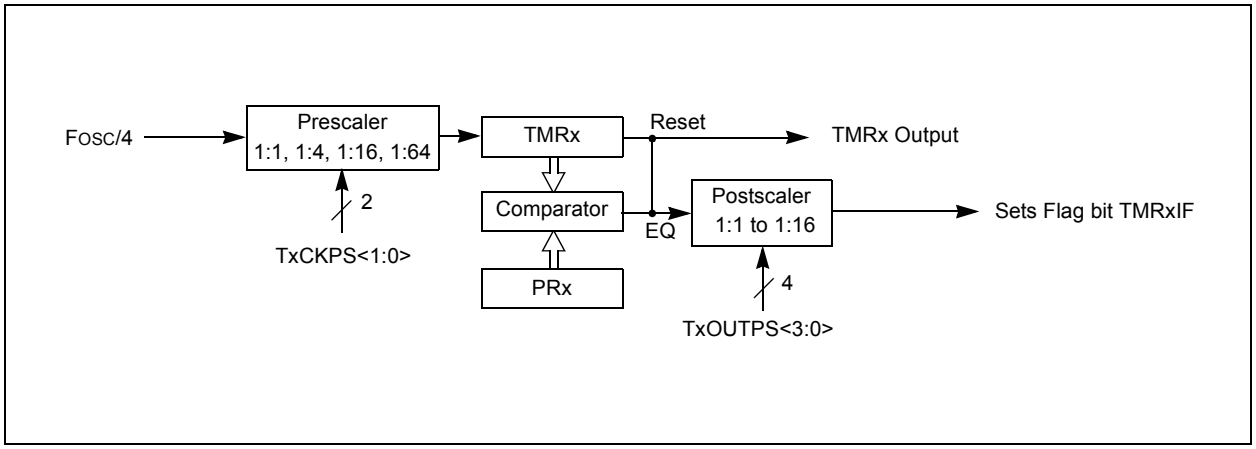
| **Note:** | The 'x' variable used in this section is used to designate Timer2, Timer4, or Timer6. For example, TxCON references T2CON, T4CON or T6CON. PRx references PR2, PR4 or PR6. |
|---|---|

The Timer2/4/6 modules incorporate the following features:

- 8-bit Timer and Period registers (TMRx and PRx, 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 TMRx match with PRx, respectively
- Optional use as the shift clock for the MSSPx modules (Timer2 only)

See Figure 22-1 for a block diagram of Timer2/4/6.

**FIGURE 22-1: TIMER2/4/6 BLOCK DIAGRAM**





© 2010-2016 Microchip Technology Inc.

## 22.1 Timer2/4/6 Operation

The clock input to the Timer2/4/6 modules is the system instruction clock (FOSC/4).

TMRx 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, TxCKPS<1:0> of the TxCON register. The value of TMRx is compared to that of the Period register, PRx, 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 TMRx to 00h on the next cycle and drives the output counter/postscaler (see **Section 22.2 "Timer2/4/6 Interrupt"**).

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

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

| | |
|---|---|
| **Note:** | TMRx is not cleared when TxCON is written. |

## 22.2 Timer2/4/6 Interrupt

Timer2/4/6 can also generate an optional device interrupt. The Timer2/4/6 output signal (TMRx-to-PRx match) provides the input for the 4-bit counter/postscaler. This counter generates the TMRx match interrupt flag which is latched in TMRxIF of the PIRx register. The interrupt is enabled by setting the TMRx Match Interrupt Enable bit, TMRxIE of the PIEx register.

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

## 22.3 Timer2/4/6 Output

The unscaled output of TMRx is available primarily to the CCP modules, where it is used as a time base for operations in PWM mode.

Timer2 can be optionally used as the shift clock source for the MSSPx modules operating in SPI mode. Additional information is provided in **Section 24.0 "Master Synchronous Serial Port (MSSP1 and MSSP2) Module"**.

## 22.4 Timer2/4/6 Operation During Sleep

The Timer2/4/6 timers cannot be operated while the processor is in Sleep mode. The contents of the TMRx and PRx registers will remain unchanged while the processor is in Sleep mode.

## 22.5 Register Definitions: Timer2 Control

**REGISTER 22-1: TXCON: TIMER2/TIMER4/TIMER6 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 |
|---|---|---|---|---|---|---|---|
| — | TxOUTPS<3:0> | | | | TMRxON | TxCKPS<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 **TxOUTPS<3:0>:** Timerx Output Postscaler Select bits

1111 = 1:16 Postscaler
1110 = 1:15 Postscaler
1101 = 1:14 Postscaler
1100 = 1:13 Postscaler
1011 = 1:12 Postscaler
1010 = 1:11 Postscaler
1001 = 1:10 Postscaler
1000 = 1:9 Postscaler
0111 = 1:8 Postscaler
0110 = 1:7 Postscaler
0101 = 1:6 Postscaler
0100 = 1:5 Postscaler
0011 = 1:4 Postscaler
0010 = 1:3 Postscaler
0001 = 1:2 Postscaler
0000 = 1:1 Postscaler

bit 2 **TMRxON:** Timerx On bit

1 = Timerx is on
0 = Timerx is off

bit 1-0 **TxCKPS<1:0>:** Timer2-type Clock Prescale Select bits

11 = Prescaler is 64
10 = Prescaler is 16
01 = Prescaler is 4
00 = Prescaler is 1


© 2010-2016 Microchip Technology Inc.

**TABLE 22-1: SUMMARY OF REGISTERS ASSOCIATED WITH TIMER2/4/6**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| CCP2CON | P2M<1:0> | | DC2B<1:0> | | CCP2M<3:0> | | | | 227 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 91 |
| PIE3 | — | CCP5IE | CCP4IE | CCP3IE | TMR6IE | — | TMR4IE | — | 93 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 95 |
| PIR3 | — | CCP5IF | CCP4IF | CCP3IF | TMR6IF | — | TMR4IF | — | 97 |
| PR2 | Timer2 Module Period Register | | | | | | | | 200* |
| PR4 | Timer4 Module Period Register | | | | | | | | 200* |
| PR6 | Timer6 Module Period Register | | | | | | | | 200* |
| T2CON | — | T2OUTPS<3:0> | | | | TMR2ON | T2CKPS<1:0> | | 202 |
| T4CON | — | T4OUTPS<3:0> | | | | TMR4ON | T4CKPS<1:0> | | 202 |
| T6CON | — | T6OUTPS<3:0> | | | | TMR6ON | T6CKPS<1:0> | | 202 |
| TMR2 | Holding Register for the 8-bit TMR2 Register | | | | | | | | 200* |
| TMR4 | Holding Register for the 8-bit TMR4 Register | | | | | | | | 200* |
| TMR6 | Holding Register for the 8-bit TMR6 Register | | | | | | | | 200* |

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

* Page provides register information.

## 23.0 CAPTURE/COMPARE/PWM MODULES

The Capture/Compare/PWM module is a peripheral which allows the user to time and control different events, and to generate Pulse-Width Modulation (PWM) signals. In Capture mode, the peripheral allows the timing of the duration of an event. The Compare mode allows the user to trigger an external event when a predetermined amount of time has expired. The PWM mode can generate pulse-width modulated signals of varying frequency and duty cycle.

This family of devices contains three Enhanced Capture/Compare/PWM modules (ECCP1, ECCP2, and ECCP3) and two standard Capture/Compare/PWM modules (CCP4 and CCP5).

The Capture and Compare functions are identical for all five CCP modules (ECCP1, ECCP2, ECCP3, CCP4, and CCP5). The only differences between CCP modules are in the Pulse-Width Modulation (PWM) function. The standard PWM function is identical in modules CCP4 and CCP5. In CCP modules ECCP1, ECCP2, and ECCP3, the Enhanced PWM function has slight variations from one another. Full-Bridge ECCP modules have four available I/O pins while half-bridge ECCP modules only have two available I/O pins. See Table 23-1 for more information.

**Note 1:** In devices with more than one CCP module, it is very important to pay close attention to the register names used. A number placed after the module acronym is used to distinguish between separate modules. For example, the CCP1CON and CCP2CON control the same operational aspects of two completely different CCP modules.

**2:** Throughout this section, generic references to a CCP module in any of its operating modes may be interpreted as being equally applicable to ECCP1, ECCP2, ECCP3, CCP4 and CCP5. Register names, module signals, I/O pins, and bit names may use the generic designator 'x' to indicate the use of a numeral to distinguish a particular module, when required.

**TABLE 23-1: PWM RESOURCES**

| Device Name | ECCP1 | ECCP2 | ECCP3 | CCP4 | CCP5 |
|---|---|---|---|---|---|
| PIC16(L)F1946/47 | Enhanced PWM Full-Bridge | Enhanced PWM Full-Bridge | Enhanced PWM Full-Bridge | Standard PWM | Standard PWM |


© 2010-2016 Microchip Technology Inc.

## 23.1 Capture Mode

The Capture mode function described in this section is available and identical for CCP modules ECCP1, ECCP2, ECCP3, CCP4 and CCP5.

Capture mode makes use of the 16-bit Timer1 resource. When an event occurs on the CCPx pin, the 16-bit CCPRxH:CCPRxL register pair captures and stores the 16-bit value of the TMR1H:TMR1L register pair, respectively. An event is defined as one of the following and is configured by the CCPxM<3:0> bits of the CCPxCON register:

- Every falling edge
- Every rising edge
- Every 4th rising edge
- Every 16th rising edge

When a capture is made, the Interrupt Request Flag bit CCPxIF of the PIRx register is set. The interrupt flag must be cleared in software. If another capture occurs before the value in the CCPRxH, CCPRxL register pair is read, and the old captured value is overwritten by the new captured value.

Figure 23-1 shows a simplified diagram of the Capture operation.

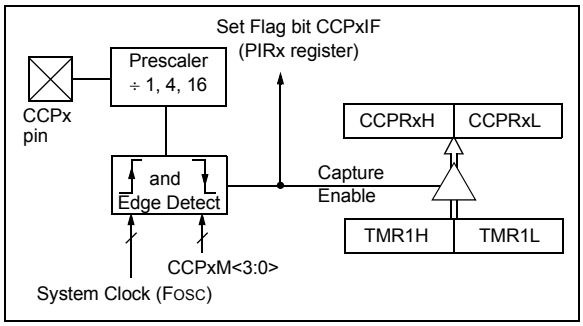
### 23.1.1 CCP PIN CONFIGURATION

In Capture mode, the CCPx pin should be configured as an input by setting the associated TRIS control bit.

Also, the CCPx pin function can be moved to alternative pins using the APFCON register. Refer to **Section 12.1 "Alternate Pin Function"** for more details.

| **Note:** | If the CCPx pin is configured as an output, a write to the port can cause a capture condition. |
|---|---|

**FIGURE 23-1: CAPTURE MODE OPERATION BLOCK DIAGRAM**




### 23.1.2 TIMER1 MODE RESOURCE

Timer1 must be running in Timer mode or Synchronized Counter mode for the CCP module to use the capture feature. In Asynchronous Counter mode, the capture operation may not work.

See **Section 21.0 "Timer1 Module with Gate Control"** for more information on configuring Timer1.

### 23.1.3 SOFTWARE INTERRUPT MODE

When the Capture mode is changed, a false capture interrupt may be generated. The user should keep the CCPxIE interrupt enable bit of the PIEx register clear to avoid false interrupts. Additionally, the user should clear the CCPxIF interrupt flag bit of the PIRx register following any change in Operating mode.

| **Note:** | Clocking Timer1 from the system clock (FOSC) should not be used in Capture mode. In order for the Capture mode to recognize the trigger event on the CCPx pin, Timer1 must be clocked from the instruction clock (FOSC/4) or from an external clock source. |
|---|---|

### 23.1.4 CCP PRESCALER

There are four prescaler settings specified by the CCPxM<3:0> bits of the CCPxCON register. Whenever the CCP module is turned off, or the CCP module is not in Capture mode, the prescaler counter is cleared. Any Reset will clear the prescaler counter.

Switching from one capture prescaler to another does not clear the prescaler and may generate a false interrupt. To avoid this unexpected operation, turn the module off by clearing the CCPxCON register before changing the prescaler. Example 23-1 demonstrates the code to perform this function.

**EXAMPLE 23-1: CHANGING BETWEEN CAPTURE PRESCALERS**

```
BANKSEL CCPxCON     ;Set Bank bits to point
                    ;to CCPxCON
CLRF    CCPxCON     ;Turn CCP module off
MOVLW   NEW_CAPT_PS ;Load the W reg with
                    ;the new prescaler
                    ;move value and CCP ON
MOVWF   CCPxCON     ;Load CCPxCON with this
                    ;value
```

### 23.1.5 CAPTURE DURING SLEEP

Capture mode depends upon the Timer1 module for proper operation. There are two options for driving the Timer1 module in Capture mode. It can be driven by the instruction clock (FOSC/4), or by an external clock source.

When Timer1 is clocked by FOSC/4, Timer1 will not increment during Sleep. When the device wakes from Sleep, Timer1 will continue from its previous state.

Capture mode will operate during Sleep when Timer1 is clocked by an external clock source.

### 23.1.6 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 12.1 "Alternate Pin Function"** for more information.

**TABLE 23-2: SUMMARY OF REGISTERS ASSOCIATED WITH CAPTURE**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| APFCON | P3CSEL | P3BSEL | P2DSEL | P2CSEL | P2BSEL | CCP2SEL | P1CSEL | P1BSEL | 123 |
| CCPxCON | PxM<1:0>(1) | | DCxB<1:0> | | CCPxM<3:0> | | | | 227 |
| CCPRxL | Capture/Compare/PWM Register x Low Byte (LSB) | | | | | | | | 205* |
| CCPRxH | Capture/Compare/PWM Register x High Byte (MSB) | | | | | | | | 205* |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 91 |
| PIE2 | OSFIE | C2IE | C1IE | EEIE | BCLIE | LCDIE | C3IE | CCP2IE | 92 |
| PIE3 | — | CCP5IE | CCP4IE | CCP3IE | TMR6IE | — | TMR4IE | — | 93 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 95 |
| PIR2 | OSFIF | C2IF | C1IF | EEIF | BCLIF | LCDIF | C3IF | CCP2IF | 96 |
| PIR3 | — | CCP5IF | CCP4IF | CCP3IF | TMR6IF | — | TMR4IF | — | 97 |
| T1CON | TMR1CS<1:0> | | T1CKPS<1:0> | | T1OSCEN | $\overline{\text{T1SYNC}}$ | — | TMR1ON | 197 |
| T1GCON | TMR1GE | T1GPOL | T1GTM | T1GSPM | T1GGO/DONE | T1GVAL | T1GSS<1:0> | | 198 |
| TMR1L | Holding Register for the Least Significant Byte of the 16-bit TMR1 Register | | | | | | | | 193* |
| TMR1H | Holding Register for the Most Significant Byte of the 16-bit TMR1 Register | | | | | | | | 193* |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 128 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 131 |
| TRISE | TRISE7 | TRISE6 | TRISE5 | TRISE4 | TRISE3 | TRISE2 | TRISE1 | TRISE0 | 137 |
| TRISG | TRISG7 | TRISG6 | TRISG5 | TRISG4 | TRISG3 | TRISG2 | TRISG1 | TRISG0 | 143 |

**Legend:** — = Unimplemented location, read as '0'. Shaded cells are not used by Capture mode.

**Note 1:** Applies to ECCP modules only.

* Page provides register information.


© 2010-2016 Microchip Technology Inc.

## 23.2 Compare Mode

The Compare mode function described in this section is available and identical for CCP modules ECCP1, ECCP2, ECCP3, CCP4 and CCP5.

Compare mode makes use of the 16-bit Timer1 resource. The 16-bit value of the CCPRxH:CCPRxL register pair is constantly compared against the 16-bit value of the TMR1H:TMR1L register pair. When a match occurs, one of the following events can occur:

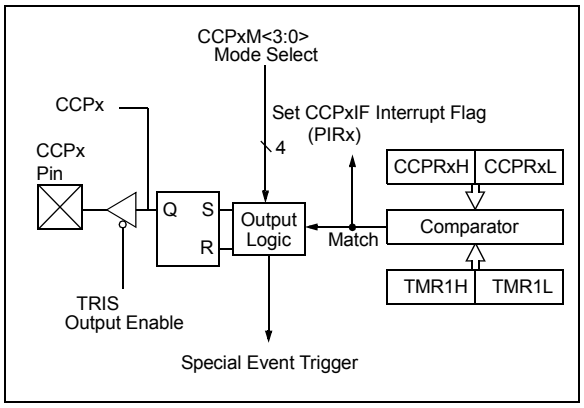
- Toggle the CCPx output
- Set the CCPx output
- Clear the CCPx output
- Generate a Special Event Trigger
- Generate a Software Interrupt

The action on the pin is based on the value of the CCPxM<3:0> control bits of the CCPxCON register. At the same time, the interrupt flag CCPxIF bit is set.

All Compare modes can generate an interrupt.

Figure 23-2 shows a simplified diagram of the Compare operation.

**FIGURE 23-2: COMPARE MODE OPERATION BLOCK DIAGRAM**




### 23.2.1 CCP PIN CONFIGURATION

The user must configure the CCPx pin as an output by clearing the associated TRIS bit.

Also, the CCPx pin function can be moved to alternative pins using the APFCON register. Refer to **Section 12.1 "Alternate Pin Function"** for more details.

| | |
|---|---|
| **Note:** | Clearing the CCPxCON register will force the CCPx compare output latch to the default low level. This is not the PORT I/O data latch. |

### 23.2.2 TIMER1 MODE RESOURCE

In Compare mode, Timer1 must be running in either Timer mode or Synchronized Counter mode. The compare operation may not work in Asynchronous Counter mode.

See **Section 21.0 "Timer1 Module with Gate Control"** for more information on configuring Timer1.

| | |
|---|---|
| **Note:** | Clocking Timer1 from the system clock (FOSC) should not be used in Capture mode. In order for Capture mode to recognize the trigger event on the CCPx pin, TImer1 must be clocked from the instruction clock (FOSC/4) or from an external clock source. |

### 23.2.3 SOFTWARE INTERRUPT MODE

When Generate Software Interrupt mode is chosen (CCPxM<3:0> = 1010), the CCPx module does not assert control of the CCPx pin (see the CCPxCON register).

### 23.2.4 SPECIAL EVENT TRIGGER

When Special Event Trigger mode is chosen (CCPxM<3:0> = 1011), the CCPx module does the following:

- Resets Timer1
- Starts an ADC conversion if ADC is enabled (CCP5 only)

The CCPx module does not assert control of the CCPx pin in this mode.

The Special Event Trigger output of the CCP occurs immediately upon a match between the TMR1H, TMR1L register pair and the CCPRxH, CCPRxL register pair. The TMR1H, TMR1L register pair is not reset until the next rising edge of the Timer1 clock. The Special Event Trigger output starts an A/D conversion (if the A/D module is enabled). This allows the CCPRxH, CCPRxL register pair to effectively provide a 16-bit programmable period register for Timer1.

**TABLE 23-3: SPECIAL EVENT TRIGGER**

| Device | CCPx/ECCPx |
|---|---|
| PIC16(L)F1946/47 | ECCP1, ECCP2, CCP3, CCP4, CCP5 |

Refer to **Section 16.0 "Analog-to-Digital Converter (ADC) Module"** for more information.

| | |
|---|---|
| **Note 1:** | The Special Event Trigger from the CCP module does not set interrupt flag bit TMR1IF of the PIR1 register. |
| **2:** | Removing the match condition by changing the contents of the CCPRxH and CCPRxL register pair, between the clock edge that generates the Special Event Trigger and the clock edge that generates the Timer1 Reset, will preclude the Reset from occurring. |

### 23.2.5 COMPARE DURING SLEEP

The Compare mode is dependent upon the system clock (FOSC) for proper operation. Since FOSC is shut down during Sleep mode, the Compare mode will not function properly during Sleep.

### 23.2.6 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 12.1 "Alternate Pin Function"** for more information.

**TABLE 23-4: SUMMARY OF REGISTERS ASSOCIATED WITH COMPARE**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| APFCON | P3CSEL | P3BSEL | P2DSEL | P2CSEL | P2BSEL | CCP2SEL | P1CSEL | P1BSEL | 123 |
| CCPxCON | PxM<1:0>(1) | | DCxB<1:0> | | CCPxM<3:0> | | | | 227 |
| CCPRxL | Capture/Compare/PWM Register x Low Byte (LSB) | | | | | | | | 205* |
| CCPRxH | Capture/Compare/PWM Register x High Byte (MSB) | | | | | | | | 205* |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 91 |
| PIE2 | OSFIE | C2IE | C1IE | EEIE | BCLIE | LCDIE | C3IE | CCP2IE | 92 |
| PIE3 | — | CCP5IE | CCP4IE | CCP3IE | TMR6IE | — | TMR4IE | — | 93 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 95 |
| PIR2 | OSFIF | C2IF | C1IF | EEIF | BCLIF | LCDIF | C31F | CCP2IF | 96 |
| PIR3 | — | CCP5IF | CCP4IF | CCP3IF | TMR6IF | — | TMR4IF | — | 97 |
| T1CON | TMR1CS<1:0> | | T1CKPS<1:0> | | T1OSCEN | $\overline{\text{T1SYNC}}$ | — | TMR1ON | 197 |
| T1GCON | TMR1GE | T1GPOL | T1GTM | T1GSPM | T1GGO/DONE | T1GVAL | T1GSS<1:0> | | 198 |
| TMR1L | Holding Register for the Least Significant Byte of the 16-bit TMR1 Register | | | | | | | | 193* |
| TMR1H | Holding Register for the Most Significant Byte of the 16-bit TMR1 Register | | | | | | | | 193* |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 128 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 131 |
| TRISE | TRISE7 | TRISE6 | TRISE5 | TRISE4 | TRISE3 | TRISE2 | TRISE1 | TRISE0 | 137 |
| TRISG | TRISG7 | TRISG6 | TRISG5 | TRISG4 | TRISG3 | TRISG2 | TRISG1 | TRISG0 | 143 |

**Legend:** — = Unimplemented location, read as '0'. Shaded cells are not used by Compare mode.

**Note 1:** Applies to ECCP modules only.

* Page provides register information.


© 2010-2016 Microchip Technology Inc.

## 23.3 PWM Overview

Pulse-Width Modulation (PWM) is a scheme that provides power to a load by switching quickly between fully On and fully Off states. The PWM signal resembles a square wave where the high portion of the signal is considered the On state and the low portion of the signal is considered the Off state. The high portion, also known as the pulse width, can vary in time and is defined in steps. A larger number of steps applied, which lengthens the pulse width, also supplies more power to the load. Lowering the number of steps applied, which shortens the pulse width, supplies less power. The PWM period is defined as the duration of one complete cycle or the total amount of On and Off time combined.

PWM resolution defines the maximum number of steps that can be present in a single PWM period. A higher resolution allows for more precise control of the pulse width time and in turn the power that is applied to the load.

The term duty cycle describes the proportion of the On time to the Off time and is expressed in percentages, where 0% is fully Off and 100% is fully On. A lower duty cycle corresponds to less power applied and a higher duty cycle corresponds to more power applied.

Figure 23-3 shows a typical waveform of the PWM signal.

### 23.3.1 STANDARD PWM OPERATION

The standard PWM function described in this section is available and identical for CCP modules ECCP1, ECCP2, ECCP3, CCP4 and CCP5.

The standard PWM mode generates a Pulse-Width modulation (PWM) signal on the CCPx pin with up to ten bits of resolution. The period, duty cycle, and resolution are controlled by the following registers:

- PRx registers
- TxCON registers
- CCPRxL registers
- CCPxCON registers

Figure 23-4 shows a simplified block diagram of the PWM operation.

| | |
|---|---|
| **Note 1:** | The corresponding TRIS bit must be cleared to enable the PWM output on the CCPx pin. |
| **2:** | Clearing the CCPxCON register will relinquish control of the CCPx pin. |

**FIGURE 23-3: CCP PWM OUTPUT SIGNAL**

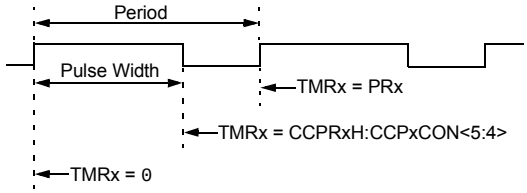
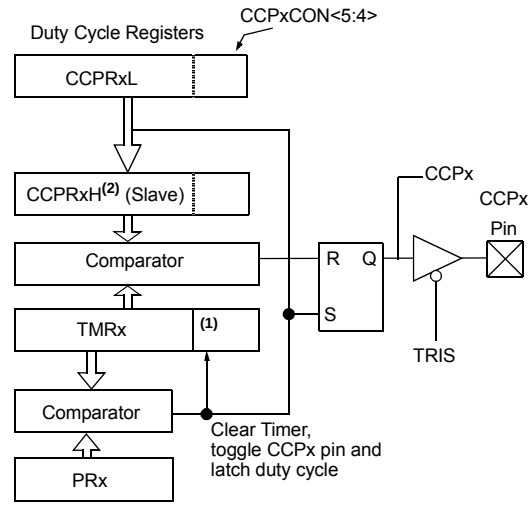



**FIGURE 23-4: SIMPLIFIED PWM BLOCK DIAGRAM**




**Note 1:** The 8-bit timer TMRx register is concatenated with the 2-bit internal system clock (FOSC), or 2 bits of the prescaler, to create the 10-bit time base.

**2:** In PWM mode, CCPRxH is a read-only register.

### 23.3.2 SETUP FOR PWM OPERATION

The following steps should be taken when configuring the CCP module for standard PWM operation:

1. Disable the CCPx pin output driver by setting the associated TRIS bit.
2. Timer2/4/6 resource selection:
   - Select the Timer2/4/6 resource to be used for PWM generation by setting the CxTSEL<1:0> bits in the CCPTMRSx register.
3. Load the PRx register with the PWM period value.
4. Configure the CCP module for the PWM mode by loading the CCPxCON register with the appropriate values.
5. Load the CCPRxL register and the DCxBx bits of the CCPxCON register, with the PWM duty cycle value.
6. Configure and start Timer2/4/6:
   - Clear the TMRxIF interrupt flag bit of the PIRx register. See Note below.
   - Configure the TxCKPS bits of the TxCON register with the Timer prescale value.
   - Enable the Timer by setting the TMRxON bit of the TxCON register.
7. Enable PWM output pin:
   - Wait until the Timer overflows and the TMRxIF bit of the PIRx register is set. See Note below.
   - Enable the CCPx pin output driver by clearing the associated TRIS bit.

| **Note:** | In order to send a complete duty cycle and period on the first PWM output, the above steps must be included in the setup sequence. If it is not critical to start with a complete PWM signal on the first output, then step 6 may be ignored. |
|---|---|

### 23.3.3 TIMER2/4/6 TIMER RESOURCE

The PWM standard mode makes use of one of the 8-bit Timer2/4/6 timer resources to specify the PWM period.

Configuring the CxTSEL<1:0> bits in the CCPTMRSx register selects which Timer2/4/6 timer is used.

### 23.3.4 PWM PERIOD

The PWM period is specified by the PRx register of Timer2/4/6. The PWM period can be calculated using the formula of Equation 23-1.

**EQUATION 23-1: PWM PERIOD**

$$PWM\ Period = [(PRx) + 1] \bullet 4 \bullet T_{OSC} \bullet (TMRx\ Prescale\ Value)$$

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

When TMRx is equal to PRx, the following three events occur on the next increment cycle:

- TMRx is cleared
- The CCPx pin is set. (Exception: If the PWM duty cycle = 0%, the pin will not be set.)
- The PWM duty cycle is latched from CCPRxL into CCPRxH.

| **Note:** | The Timer postscaler (see **Section 22.1 "Timer2/4/6 Operation"**) is not used in the determination of the PWM frequency. |
|---|---|

### 23.3.5 PWM DUTY CYCLE

The PWM duty cycle is specified by writing a 10-bit value to multiple registers: CCPRxL register and DCxB<1:0> bits of the CCPxCON register. The CCPRxL contains the eight MSbs and the DCxB<1:0> bits of the CCPxCON register contain the two LSbs. CCPRxL and DCxB<1:0> bits of the CCPxCON register can be written to at any time. The duty cycle value is not latched into CCPRxH until after the period completes (i.e., a match between PRx and TMRx registers occurs). While using the PWM, the CCPRxH register is read-only.

Equation 23-2 is used to calculate the PWM pulse width.

Equation 23-3 is used to calculate the PWM duty cycle ratio.

**EQUATION 23-2: PULSE WIDTH**

$$Pulse\ Width = (CCPRxL{:}CCPxCON\langle 5{:}4\rangle) \bullet T_{OSC} \bullet (TMRx\ Prescale\ Value)$$

**EQUATION 23-3: DUTY CYCLE RATIO**

$$Duty\ Cycle\ Ratio = \frac{(CCPRxL{:}CCPxCON\langle 5{:}4\rangle)}{4(PRx + 1)}$$

The CCPRxH register and a 2-bit internal latch are used to double buffer the PWM duty cycle. This double buffering is essential for glitchless PWM operation.

The 8-bit timer TMRx register is concatenated with either the 2-bit internal system clock (FOSC), or two bits of the prescaler, to create the 10-bit time base. The system clock is used if the Timer2/4/6 prescaler is set to 1:1.

When the 10-bit time base matches the CCPRxH and 2-bit latch, then the CCPx pin is cleared (see Figure 23-4).


© 2010-2016 Microchip Technology Inc.

### 23.3.6 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 PRx is 255. The resolution is a function of the PRx register value as shown by Equation 23-4.

**EQUATION 23-4: PWM RESOLUTION**

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

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

**TABLE 23-5: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS (FOSC = 32 MHz)**

| PWM Frequency | 1.95 kHz | 7.81 kHz | 31.25 kHz | 125 kHz | 250 kHz | 333.3 kHz |
|---|---|---|---|---|---|---|
| Timer Prescale | 16 | 4 | 1 | 1 | 1 | 1 |
| PRx Value | 0xFF | 0xFF | 0xFF | 0x3F | 0x1F | 0x17 |
| Maximum Resolution (bits) | 10 | 10 | 10 | 8 | 7 | 6.6 |

**TABLE 23-6: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS (FOSC = 20 MHz)**

| PWM Frequency | 1.22 kHz | 4.88 kHz | 19.53 kHz | 78.12 kHz | 156.3 kHz | 208.3 kHz |
|---|---|---|---|---|---|---|
| Timer Prescale | 16 | 4 | 1 | 1 | 1 | 1 |
| PRx Value | 0xFF | 0xFF | 0xFF | 0x3F | 0x1F | 0x17 |
| Maximum Resolution (bits) | 10 | 10 | 10 | 8 | 7 | 6.6 |

**TABLE 23-7: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS (FOSC = 8 MHz)**

| PWM Frequency | 1.22 kHz | 4.90 kHz | 19.61 kHz | 76.92 kHz | 153.85 kHz | 200.0 kHz |
|---|---|---|---|---|---|---|
| Timer Prescale | 16 | 4 | 1 | 1 | 1 | 1 |
| PRx Value | 0x65 | 0x65 | 0x65 | 0x19 | 0x0C | 0x09 |
| Maximum Resolution (bits) | 8 | 8 | 8 | 6 | 5 | 5 |

© 2010-2016 Microchip Technology Inc.

### 23.3.7 OPERATION IN SLEEP MODE

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

### 23.3.8 CHANGES IN SYSTEM CLOCK FREQUENCY

The PWM frequency is derived from the system clock frequency. Any changes in the system clock frequency will result in changes to the PWM frequency. See **Section 5.0 "Oscillator Module (With Fail-Safe Clock Monitor)"** for additional details.

### 23.3.9 EFFECTS OF RESET

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

### 23.3.10 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 12.1 "Alternate Pin Function"** for more information.

**TABLE 23-8: SUMMARY OF REGISTERS ASSOCIATED WITH STANDARD PWM**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| APFCON | P3CSEL | P3BSEL | P2DSEL | P2CSEL | P2BSEL | CCP2SEL | P1CSEL | P1BSEL | 123 |
| CCPxCON | PxM<1:0>(1) | | DCxB<1:0> | | CCPxM<3:0> | | | | 227 |
| CCPTMRS0 | C4TSEL<1:0> | | C3TSEL<1:0> | | C2TSEL<1:0> | | C1TSEL<1:0> | | 228 |
| CCPTMRS1 | — | — | — | — | — | — | C5TSEL<1:0> | | 228 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 91 |
| PIE2 | OSFIE | C2IE | C1IE | EEIE | BCLIE | LCDIE | C3IE | CCP2IE | 92 |
| PIE3 | — | CCP5IE | CCP4IE | CCP3IE | TMR6IE | — | TMR4IE | — | 93 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 95 |
| PIR2 | OSFIF | C2IF | C1IF | EEIF | BCLIF | LCDIF | C3IF | CCP2IF | 96 |
| PIR3 | — | CCP5IF | CCP4IF | CCP3IF | TMR6IF | — | TMR4IF | — | 97 |
| PR2 | Timer2 Period Register | | | | | | | | 200* |
| PR4 | Timer4Period Register | | | | | | | | 200* |
| PR6 | Timer6 Period Register | | | | | | | | 200* |
| T2CON | — | T2OUTPS<3:0> | | | | TMR2ON | T2CKPS<:0>1 | | 202 |
| T4CON | — | T4OUTPS<3:0> | | | | TMR4ON | T4CKPS<:0>1 | | 202 |
| T6CON | — | T6OUTPS<3:0> | | | | TMR6ON | T6CKPS<:0>1 | | 202 |
| TMR2 | Timer2 Module Register | | | | | | | | 200* |
| TMR4 | Timer4 Module Register | | | | | | | | 200* |
| TMR6 | Timer6 Module Register | | | | | | | | 200* |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 128 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 131 |
| TRISE | TRISE7 | TRISE6 | TRISE5 | TRISE4 | TRISE3 | TRISE2 | TRISE1 | TRISE0 | 137 |
| TRISG | TRISG7 | TRISG6 | TRISG5 | TRISG4 | TRISG3 | TRISG2 | TRISG1 | TRISG0 | 143 |

**Legend:** — = Unimplemented location, read as '0'. Shaded cells are not used by the PWM.
**Note 1:** Applies to ECCP modules only.
* Page provides register information.


© 2010-2016 Microchip Technology Inc.

## 23.4 PWM (Enhanced Mode)

The enhanced PWM function described in this section is available for CCP modules ECCP1, ECCP2 and ECCP3, with any differences between modules noted.

The enhanced PWM mode generates a Pulse-Width Modulation (PWM) signal on up to four different output pins with up to ten bits of resolution. The period, duty cycle, and resolution are controlled by the following registers:

- PRx registers
- TxCON registers
- CCPRxL registers
- CCPxCON registers

The ECCP modules have the following additional PWM registers which control Auto-shutdown, Auto-restart, Dead-band Delay and PWM Steering modes:

- CCPxAS registers
- PSTRxCON registers
- PWMxCON registers

The enhanced PWM module can generate the following five PWM Output modes:

- Single PWM
- Half-Bridge PWM
- Full-Bridge PWM, Forward Mode
- Full-Bridge PWM, Reverse Mode
- Single PWM with PWM Steering Mode

To select an Enhanced PWM Output mode, the PxM bits of the CCPxCON register must be configured appropriately.

The PWM outputs are multiplexed with I/O pins and are designated PxA, PxB, PxC and PxD. The polarity of the PWM pins is configurable and is selected by setting the CCPxM bits in the CCPxCON register appropriately.

Figure 23-5 shows an example of a simplified block diagram of the Enhanced PWM module.

Table shows the pin assignments for various Enhanced PWM modes.

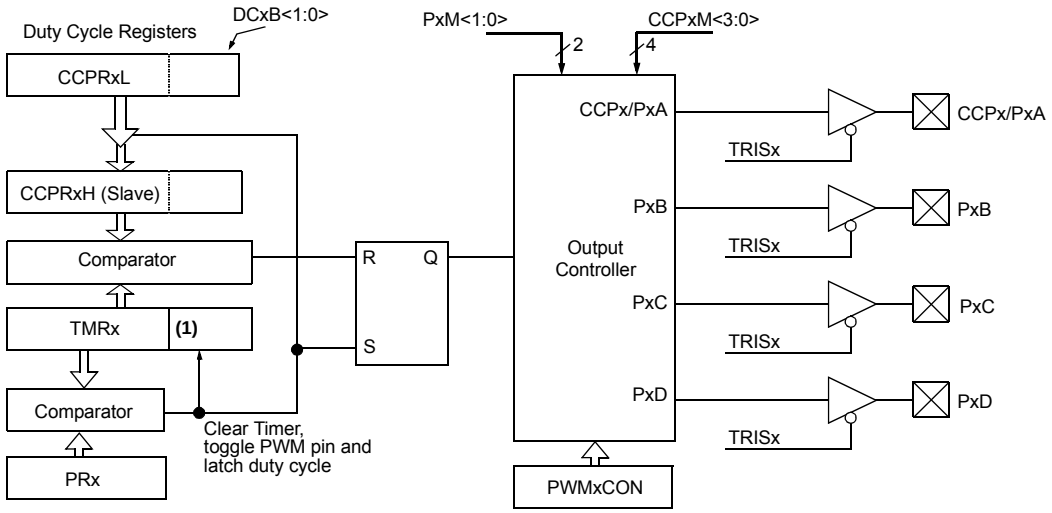
**Note 1:** The corresponding TRIS bit must be cleared to enable the PWM output on the CCPx pin.

**2:** Clearing the CCPxCON register will relinquish control of the CCPx pin.

**3:** Any pin not used in the enhanced PWM mode is available for alternate pin functions, if applicable.

**4:** To prevent the generation of an incomplete waveform when the PWM is first enabled, the ECCP module waits until the start of a new PWM period before generating a PWM signal.

**FIGURE 23-5: EXAMPLE SIMPLIFIED BLOCK DIAGRAM OF THE ENHANCED PWM MODE**



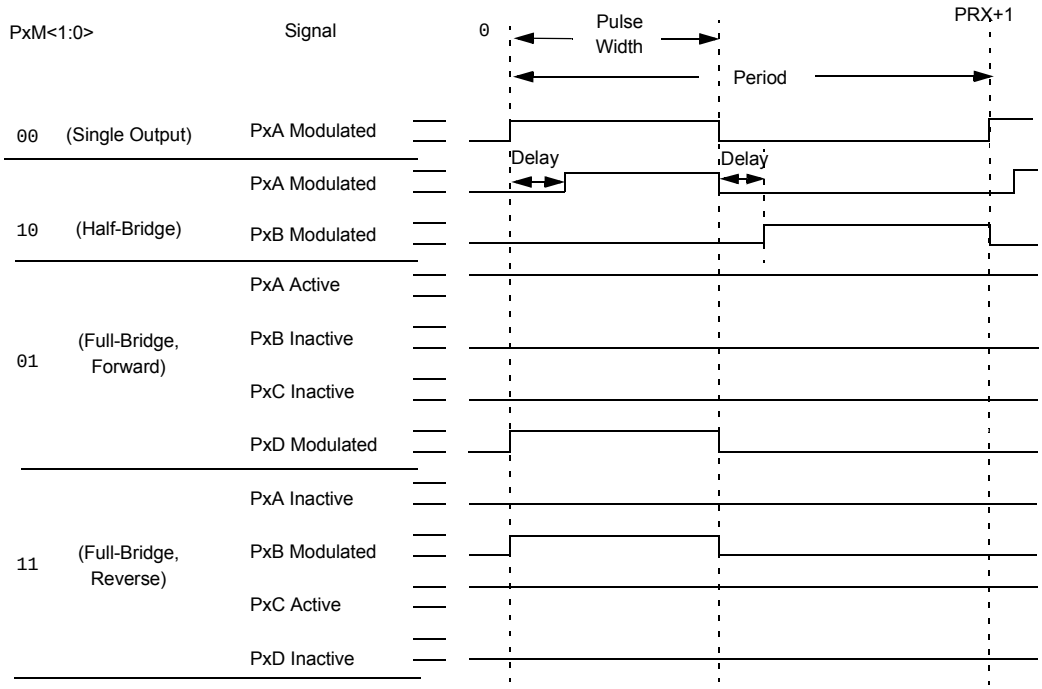

**Note 1:** The 8-bit timer TMRx register is concatenated with the 2-bit internal Q clock, or 2 bits of the prescaler to create the 10-bit time base.

**TABLE 23-9: EXAMPLE PIN ASSIGNMENTS FOR VARIOUS PWM ENHANCED MODES**

| ECCP Mode | PxM<1:0> | CCPx/PxA | PxB | PxC | PxD |
|---|---|---|---|---|---|
| Single | 00 | Yes(1) | Yes(1) | Yes(1) | Yes(1) |
| Half-Bridge | 10 | Yes | Yes | No | No |
| Full-Bridge, Forward | 01 | Yes | Yes | Yes | Yes |
| Full-Bridge, Reverse | 11 | Yes | Yes | Yes | Yes |

**Note 1:** PWM Steering enables outputs in Single mode.

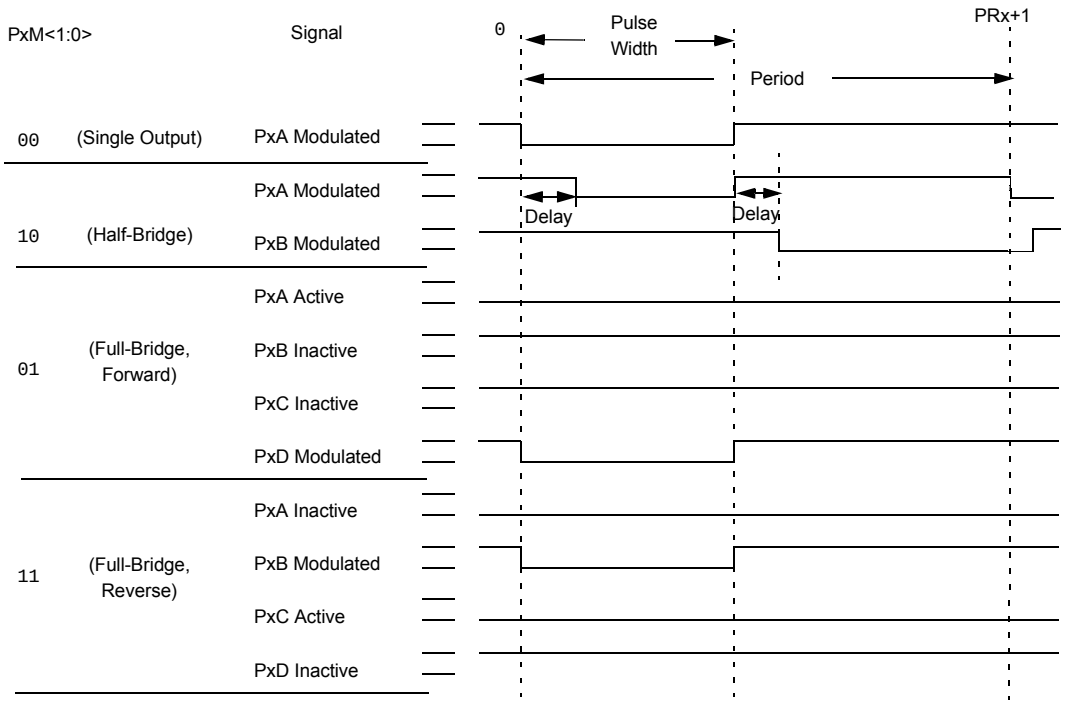
**FIGURE 23-6: EXAMPLE PWM (ENHANCED MODE) OUTPUT RELATIONSHIPS (ACTIVE-HIGH STATE)**




Relationships:
- Period = 4 * Tosc * (PRx + 1) * (TMRx Prescale Value)
- Pulse Width = Tosc * (CCPRxL<7:0>:CCPxCON<5:4>) * (TMRx Prescale Value)
- Delay = 4 * Tosc * (PWMxCON<6:0>)


© 2010-2016 Microchip Technology Inc.

**FIGURE 23-7: EXAMPLE ENHANCED PWM OUTPUT RELATIONSHIPS (ACTIVE-LOW STATE)**




Relationships:

- Period = 4 * Tosc * (PRx + 1) * (TMRx Prescale Value)
- Pulse Width = Tosc * (CCPRxL<7:0>:CCPxCON<5:4>) * (TMRx Prescale Value)
- Delay = 4 * Tosc * (PWMxCON<6:0>)

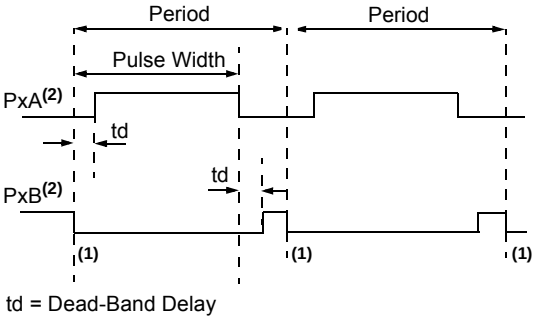
### 23.4.1 HALF-BRIDGE MODE

In Half-Bridge mode, two pins are used as outputs to drive push-pull loads. The PWM output signal is output on the CCPx/PxA pin, while the complementary PWM output signal is output on the PxB pin (see Figure 23-9). This mode can be used for Half-Bridge applications, as shown in Figure 23-9, or for Full-Bridge applications, where four power switches are being modulated with two PWM signals.

In Half-Bridge mode, the programmable dead-band delay can be used to prevent shoot-through current in Half-Bridge power devices. The value of the PDC<6:0> bits of the PWMxCON register sets the number of instruction cycles before the output is driven active. If the value is greater than the duty cycle, the corresponding output remains inactive during the entire cycle. See **Section 23.4.5 "Programmable Dead-Band Delay Mode"** for more details of the dead-band delay operations.

Since the PxA and PxB outputs are multiplexed with the PORT data latches, the associated TRIS bits must be cleared to configure PxA and PxB as outputs.

**FIGURE 23-8: EXAMPLE OF HALF-BRIDGE PWM OUTPUT**



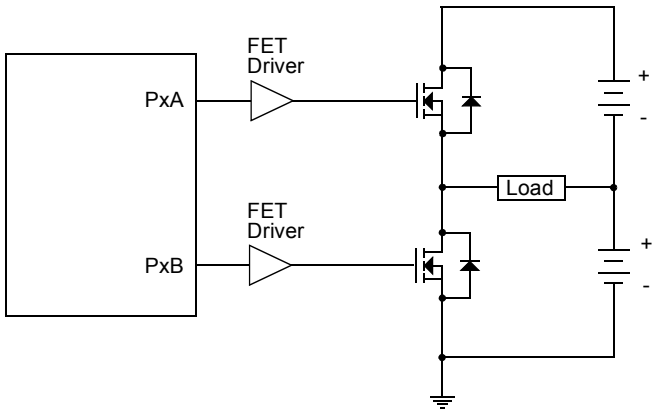

td = Dead-Band Delay

**Note 1:** At this time, the TMRx register is equal to the PRx register.

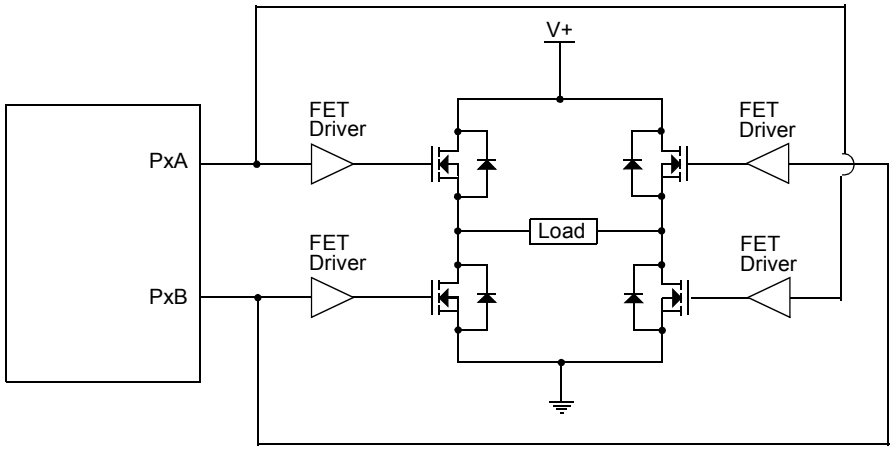
**2:** Output signals are shown as active-high.

**FIGURE 23-9: EXAMPLE OF HALF-BRIDGE APPLICATIONS**

**Standard Half-Bridge Circuit ("Push-Pull")**




**Half-Bridge Output Driving a Full-Bridge Circuit**





© 2010-2016 Microchip Technology Inc.

### 23.4.2 FULL-BRIDGE MODE

In Full-Bridge mode, all four pins are used as outputs. An example of Full-Bridge application is shown in Figure 23-10.

In the Forward mode, pin CCPx/PxA is driven to its active state, pin PxD is modulated, while PxB and PxC will be driven to their inactive state as shown in Figure 23-11.

In the Reverse mode, PxC is driven to its active state, pin PxB is modulated, while PxA and PxD will be driven to their inactive state as shown Figure 23-11.

PxA, PxB, PxC and PxD outputs are multiplexed with the PORT data latches. The associated TRIS bits must be cleared to configure the PxA, PxB, PxC and PxD pins as outputs.

**FIGURE 23-10: EXAMPLE OF FULL-BRIDGE APPLICATION**

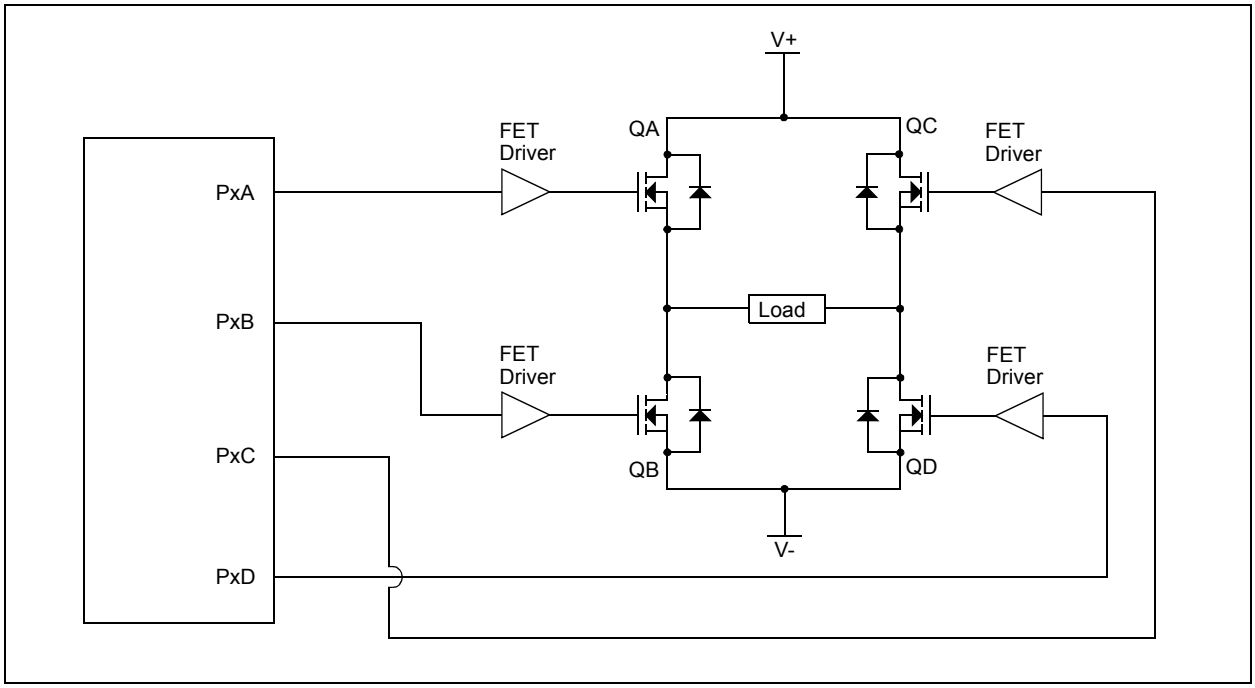
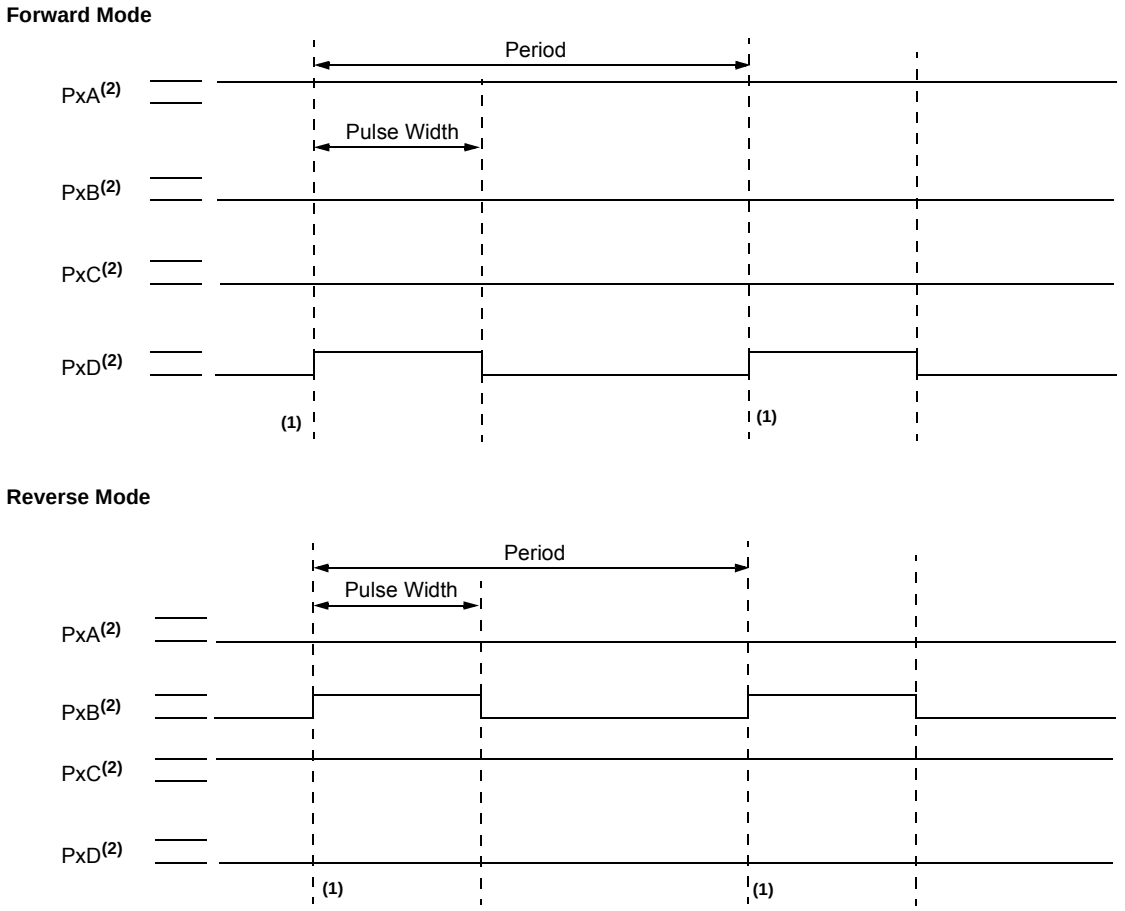


**FIGURE 23-11: EXAMPLE OF FULL-BRIDGE PWM OUTPUT**




**Note 1:** At this time, the TMRx register is equal to the PRx register.

**2:** Output signal is shown as active-high.


© 2010-2016 Microchip Technology Inc.

#### 23.4.2.1 Direction Change in Full-Bridge Mode

In the Full-Bridge mode, the PxM1 bit in the CCPxCON register allows users to control the forward/reverse direction. When the application firmware changes this direction control bit, the module will change to the new direction on the next PWM cycle.

A direction change is initiated in software by changing the PxM1 bit of the CCPxCON register. The following sequence occurs four Timer cycles prior to the end of the current PWM period:

- The modulated outputs (PxB and PxD) are placed in their inactive state.
- The associated unmodulated outputs (PxA and PxC) are switched to drive in the opposite direction.
- PWM modulation resumes at the beginning of the next period.

See Figure 23-12 for an illustration of this sequence.

The Full-Bridge mode does not provide dead-band delay. As one output is modulated at a time, dead-band delay is generally not required. There is a situation where dead-band delay is required. This situation occurs when both of the following conditions are true:

1. The direction of the PWM output changes when the duty cycle of the output is at or near 100%.
2. The turn-off time of the power switch, including the power device and driver circuit, is greater than the turn on time.

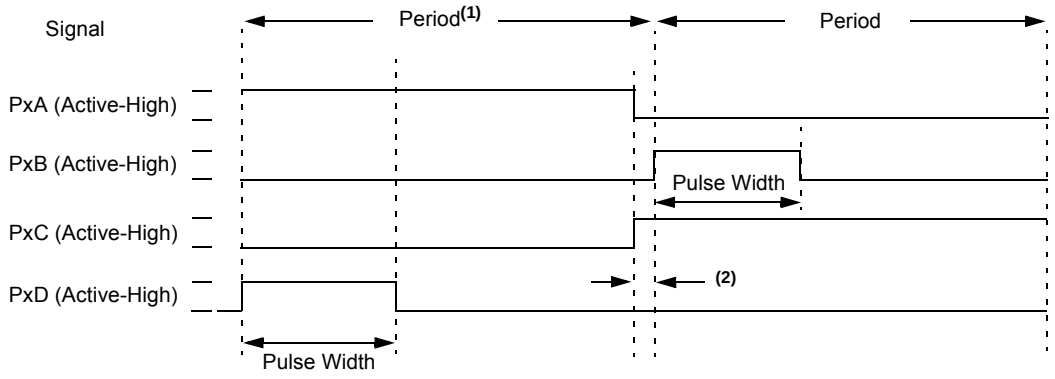
Figure 23-13 shows an example of the PWM direction changing from forward to reverse, at a near 100% duty cycle. In this example, at time t1, the output PxA and PxD become inactive, while output PxC becomes active. Since the turn-off time of the power devices is longer than the turn-on time, a shoot-through current will flow through power devices QC and QD (see Figure 23-10) for the duration of 't'. The same phenomenon will occur to power devices QA and QB for PWM direction change from reverse to forward.

If changing PWM direction at high duty cycle is required for an application, two possible solutions for eliminating the shoot-through current are:

1. Reduce PWM duty cycle for one PWM period before changing directions.
2. Use switch drivers that can drive the switches off faster than they can drive them on.

Other options to prevent shoot-through current may exist.

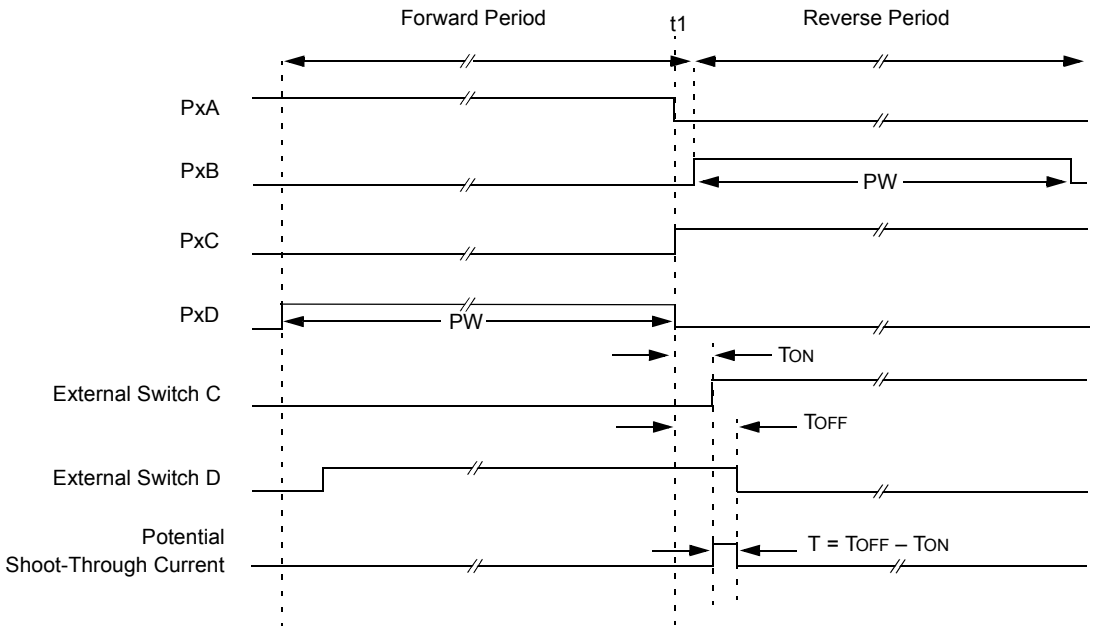
**FIGURE 23-12: EXAMPLE OF PWM DIRECTION CHANGE**




**Note 1:** The direction bit PxM1 of the CCPxCON register is written any time during the PWM cycle.

**2:** When changing directions, the PxA and PxC signals switch before the end of the current PWM cycle. The modulated PxB and PxD signals are inactive at this time. The length of this time is four Timer counts.

**FIGURE 23-13: EXAMPLE OF PWM DIRECTION CHANGE AT NEAR 100% DUTY CYCLE**




**Note 1:** All signals are shown as active-high.

**2:** TON is the turn-on delay of power switch QC and its driver.

**3:** TOFF is the turn-off delay of power switch QD and its driver.


© 2010-2016 Microchip Technology Inc.

### 23.4.3 ENHANCED PWM AUTO-SHUTDOWN MODE

The PWM mode supports an Auto-Shutdown mode that will disable the PWM outputs when an external shutdown event occurs. Auto-Shutdown mode places the PWM output pins into a predetermined state. This mode is used to help prevent the PWM from damaging the application.

The auto-shutdown sources are selected using the CCPxAS<2:0> bits of the CCPxAS register. A shutdown event may be generated by:

- A logic '0' on the FLT0 pin
- A logic '1' on a Comparator (async_CxOUT) output

A shutdown condition is indicated by the CCPxASE (Auto-Shutdown Event Status) bit of the CCPxAS register. If the bit is a '0', the PWM pins are operating normally. If the bit is a '1', the PWM outputs are in the shutdown state.

When a shutdown event occurs, two things happen:

The CCPxASE bit is set to '1'. The CCPxASE will remain set until cleared in firmware or an auto-restart occurs (see **Section 23.4.4 "Auto-Restart Mode"**).

The enabled PWM pins are asynchronously placed in their shutdown states. The PWM output pins are grouped into pairs [PxA/PxC] and [PxB/PxD]. The state of each pin pair is determined by the PSSxAC and PSSxBD bits of the CCPxAS register. Each pin pair may be placed into one of three states:

- Drive logic '1'
- Drive logic '0'
- Tri-state (high-impedance)

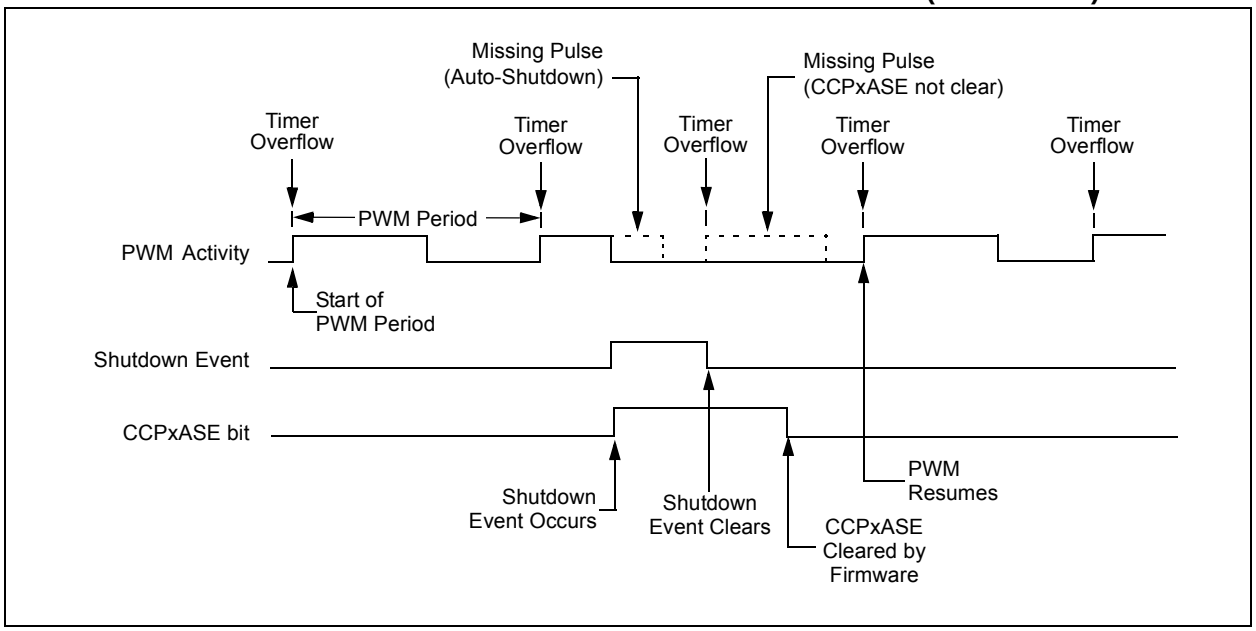
**Note 1:** The auto-shutdown condition is a level-based signal, not an edge-based signal. As long as the level is present, the auto-shutdown will persist.

**2:** Writing to the CCPxASE bit of the CCPxAS register is disabled while an auto-shutdown condition persists.

**3:** Once the auto-shutdown condition has been removed and the PWM restarted (either through firmware or auto-restart) the PWM signal will always restart at the beginning of the next PWM period.

**4:** Prior to an auto-shutdown event caused by a comparator output or FLT0 pin event, a software shutdown can be triggered in firmware by setting the CCPxASE bit of the CCPxAS register to '1'. The Auto-Restart feature tracks the active status of a shutdown caused by a comparator output or FLT0 pin event only. If it is enabled at this time, it will immediately clear this bit and restart the ECCP module at the beginning of the next PWM period.

**FIGURE 23-14: PWM AUTO-SHUTDOWN WITH FIRMWARE RESTART (PXRSEN = 0)**

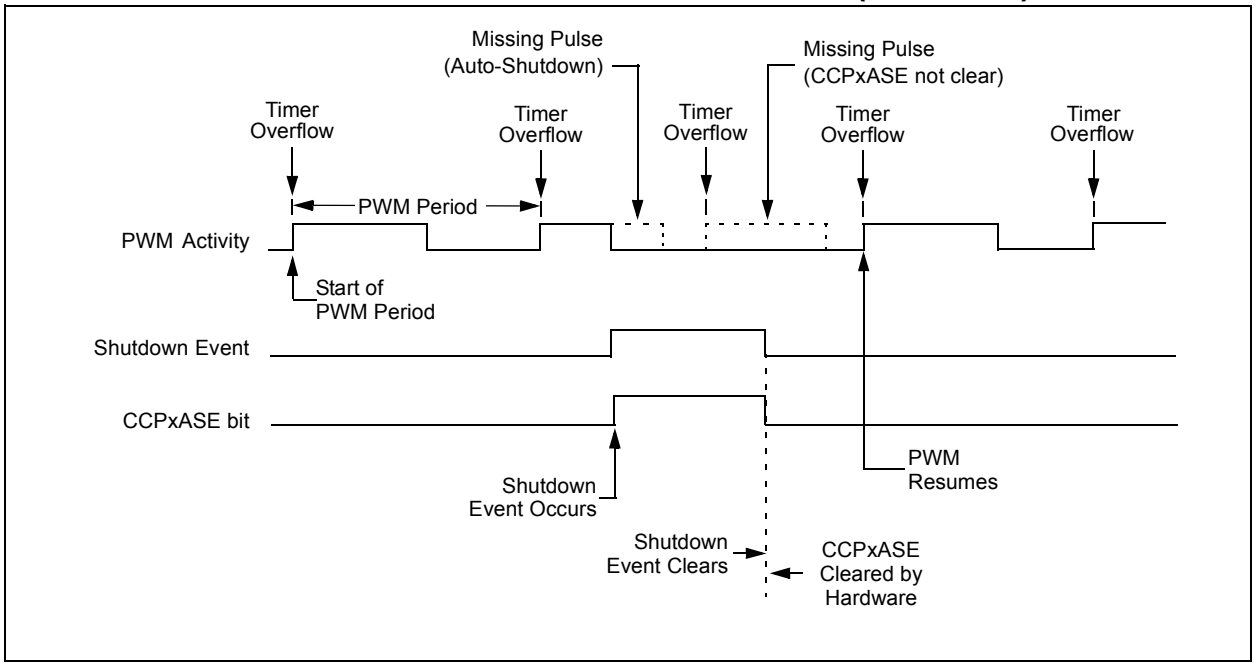



### 23.4.4 AUTO-RESTART MODE

The Enhanced PWM can be configured to automatically restart the PWM signal once the auto-shutdown condition has been removed. Auto-restart is enabled by setting the PxRSEN bit in the PWMxCON register.

If auto-restart is enabled, the CCPxASE bit will remain set as long as the auto-shutdown condition is active. When the auto-shutdown condition is removed, the CCPxASE bit will be cleared via hardware and normal operation will resume.

**FIGURE 23-15: PWM AUTO-SHUTDOWN WITH AUTO-RESTART (PXRSEN = 1)**





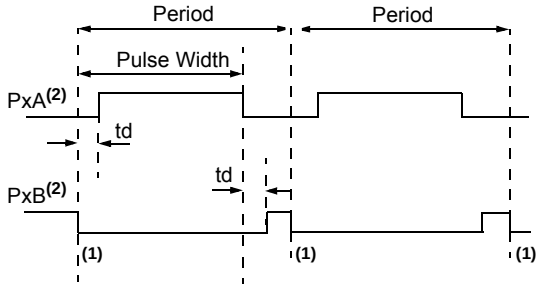
© 2010-2016 Microchip Technology Inc.

### 23.4.5 PROGRAMMABLE DEAD-BAND DELAY MODE

In half-bridge applications, where all power switches are modulated at the PWM frequency, the power switches normally require more time to turn off than to turn on. If both the upper and lower power switches are switched at the same time (one turned on, and the other turned off), both switches may be on for a short period of time until one switch completely turns off. During this brief interval, a very high current (*shoot-through current*) will flow through both power switches, shorting the bridge supply. To avoid this potentially destructive shoot-through current from flowing during switching, turning on either of the power switches is normally delayed to allow the other switch to completely turn off.

In Half-Bridge mode, a digitally programmable dead-band delay is available to avoid shoot-through current from destroying the bridge power switches. The delay occurs at the signal transition from the non-active state to the active state. See Figure 23-16 for illustration. The lower seven bits of the associated PWMxCON register (Register 23-5) sets the delay period in terms of microcontroller instruction cycles (TCY or 4 TOSC).

**FIGURE 23-16: EXAMPLE OF HALF-BRIDGE PWM OUTPUT**

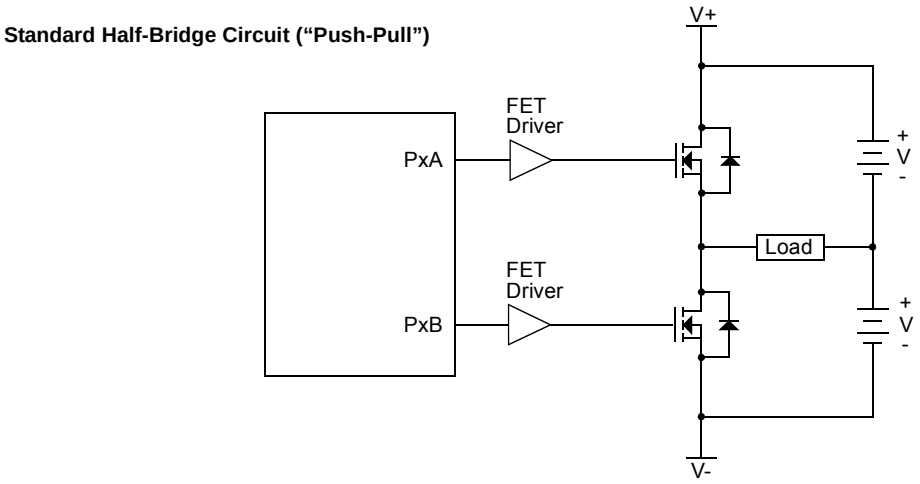



td = Dead-Band Delay

**Note 1:** At this time, the TMRx register is equal to the PRx register.

**2:** Output signals are shown as active-high.

**FIGURE 23-17: EXAMPLE OF HALF-BRIDGE APPLICATIONS**



### 23.4.6 PWM STEERING MODE

In Single Output mode, PWM steering allows any of the PWM pins to be the modulated signal. Additionally, the same PWM signal can be simultaneously available on multiple pins.

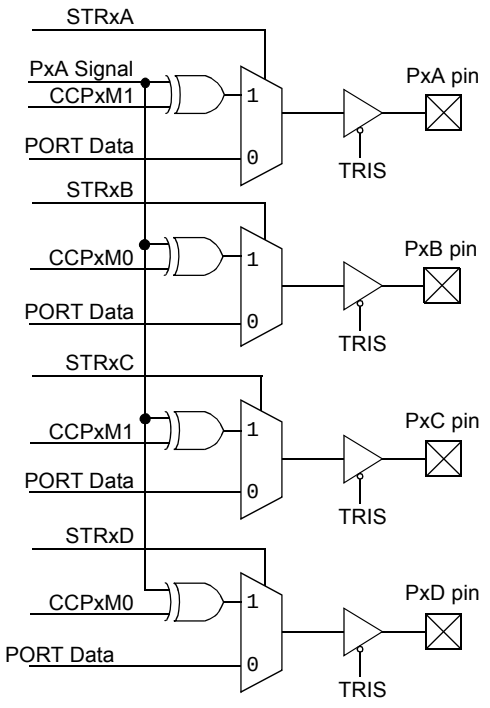
Once the Single Output mode is selected (CCPxM<3:2> = 11 and PxM<1:0> = 00 of the CCPxCON register), the user firmware can bring out the same PWM signal to one, two, three or four output pins by setting the appropriate STRx<D:A> bits of the PSTRxCON register, as shown in Table 23-9.

| **Note:** | The associated TRIS bits must be set to output ('0') to enable the pin output driver in order to see the PWM signal on the pin. |
|---|---|

While the PWM Steering mode is active, CCPxM<1:0> bits of the CCPxCON register select the PWM output polarity for the Px<D:A> pins.

The PWM auto-shutdown operation also applies to PWM Steering mode as described in **Section 23.4.3 "Enhanced PWM Auto-shutdown mode"**. An auto-shutdown event will only affect pins that have PWM outputs enabled.

**FIGURE 23-18: SIMPLIFIED STEERING BLOCK DIAGRAM**




**Note 1:** Port outputs are configured as shown when the CCPxCON register bits PxM<1:0> = 00 and CCPxM<3:2> = 11.

**2:** Single PWM output requires setting at least one of the STRx bits.


© 2010-2016 Microchip Technology Inc.

#### 23.4.6.1 Steering Synchronization

The STRxSYNC bit of the PSTRxCON register gives the user two selections of when the steering event will happen. When the STRxSYNC bit is '0', the steering event will happen at the end of the instruction that writes to the PSTRxCON register. In this case, the output signal at the Px<D:A> pins may be an incomplete PWM waveform. This operation is useful when the user firmware needs to immediately remove a PWM signal from the pin.

When the STRxSYNC bit is '1', the effective steering update will happen at the beginning of the next PWM period. In this case, steering on/off the PWM output will always produce a complete PWM waveform.

Figures 23-19 and 23-20 illustrate the timing diagrams of the PWM steering depending on the STRxSYNC setting.

### 23.4.7 START-UP CONSIDERATIONS

When any PWM mode is used, the application hardware must use the proper external pull-up and/or pull-down resistors on the PWM output pins.

The CCPxM<1:0> bits of the CCPxCON register allow the user to choose whether the PWM output signals are active-high or active-low for each pair of PWM output pins (PxA/PxC and PxB/PxD). The PWM output polarities must be selected before the PWM pin output drivers are enabled. Changing the polarity configuration while the PWM pin output drivers are enable is not recommended since it may result in damage to the application circuits.

The PxA, PxB, PxC and PxD output latches may not be in the proper states when the PWM module is initialized. Enabling the PWM pin output drivers at the same time as the Enhanced PWM modes may cause damage to the application circuit. The Enhanced PWM modes must be enabled in the proper Output mode and complete a full PWM cycle before enabling the PWM pin output drivers. The completion of a full PWM cycle is indicated by the TMRxIF bit of the PIRx register being set as the second PWM period begins.

| | |
|---|---|
| **Note:** | When the microcontroller is released from Reset, all of the I/O pins are in the high-impedance state. The external circuits must keep the power switch devices in the Off state until the microcontroller drives the I/O pins with the proper signal levels or activates the PWM output(s). |

**FIGURE 23-19: EXAMPLE OF STEERING EVENT AT END OF INSTRUCTION (STRxSYNC = 0)**

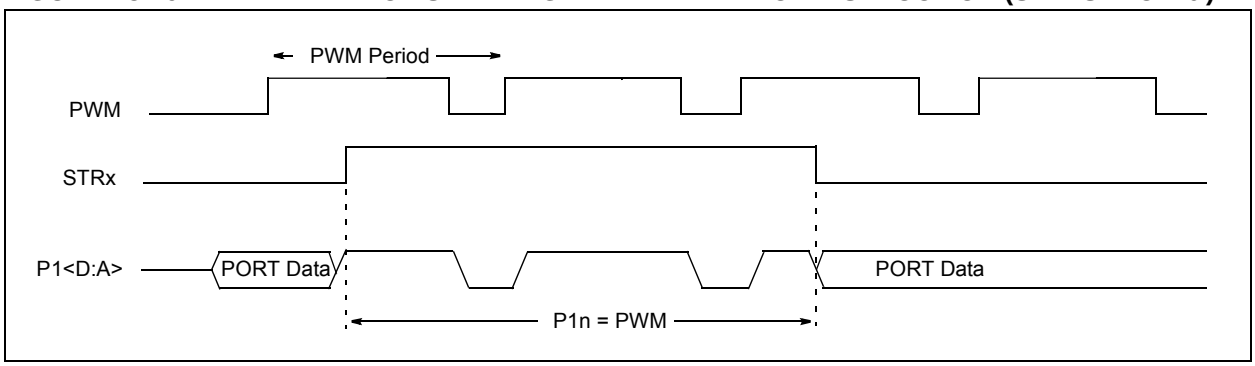
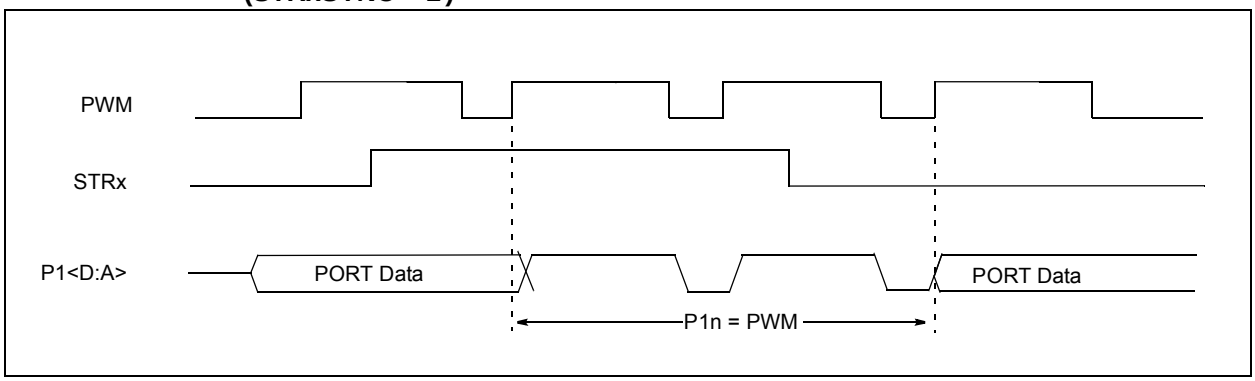



**FIGURE 23-20: EXAMPLE OF STEERING EVENT AT BEGINNING OF INSTRUCTION (STRxSYNC = 1)**



### 23.4.8 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 12.1 "Alternate Pin Function"** for more information.

**TABLE 23-10: SUMMARY OF REGISTERS ASSOCIATED WITH ENHANCED PWM**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| APFCON | P3CSEL | P3BSEL | P2DSEL | P2CSEL | P2BSEL | CCP2SEL | P1CSEL | P1BSEL | 123 |
| CCPxCON | PxM<1:0>(1) | | DCxB<1:0> | | CCPxM<3:0> | | | | 227 |
| CCPxAS | CCPxASE | CCPxAS<2:0> | | | PSSxAC<1:0> | | PSSxBD<1:0> | | 229 |
| CCPTMRS0 | C4TSEL<1:0> | | C3TSEL<1:0> | | C2TSEL<1:0> | | C1TSEL<1:0> | | 228 |
| CCPTMRS1 | — | — | — | — | — | — | C5TSEL<1:0> | | 228 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 91 |
| PIE2 | OSFIE | C2IE | C1IE | EEIE | BCLIE | LCDIE | C3IE | CCP2IE | 92 |
| PIE3 | — | CCP5IE | CCP4IE | CCP3IE | TMR6IE | — | TMR4IE | — | 93 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 95 |
| PIR2 | OSFIF | C2IF | C1IF | EEIF | BCLIF | LCDIF | C3IF | CCP2IF | 96 |
| PIR3 | — | CCP5IF | CCP4IF | CCP3IF | TMR6IF | — | TMR4IF | — | 97 |
| PR2 | Timer2 Period Register | | | | | | | | 200* |
| PR4 | Timer4 Period Register | | | | | | | | 200* |
| PR6 | Timer6 Period Register | | | | | | | | 200* |
| PSTRxCON | — | — | — | STRx-SYNC | STRxD | STRxC | STRxB | STRxA | 231 |
| PWMxCON | PxRSEN | PxDC<6:0> | | | | | | | 230 |
| T2CON | — | T2OUTPS<3:0> | | | | TMR2ON | T2CKPS<:0>1 | | 202 |
| T4CON | — | T4OUTPS<3:0> | | | | TMR4ON | T4CKPS<:0>1 | | 202 |
| T6CON | — | T6OUTPS<3:0> | | | | TMR6ON | T6CKPS<:0>1 | | 202 |
| TMR2 | Timer2 Module Register | | | | | | | | 200* |
| TMR4 | Timer4 Module Register | | | | | | | | 200* |
| TMR6 | Timer6 Module Register | | | | | | | | 200* |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 128 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 131 |
| TRISD | TRISD7 | TRISD6 | TRISD5 | TRISD4 | TRISD3 | TRISD2 | TRISD1 | TRISD0 | 134 |
| TRISE | TRISE7 | TRISE6 | TRISE5 | TRISE4 | TRISE3 | TRISE2 | TRISE1 | TRISE0 | 137 |
| TRISG | TRISG7 | TRISG6 | TRISG5 | TRISG4 | TRISG3 | TRISG2 | TRISG1 | TRISG0 | 143 |

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

**Note 1:** Applies to ECCP modules only.

* Page provides register information.


© 2010-2016 Microchip Technology Inc.

## 23.5 Register Definitions: ECCP Control

### REGISTER 23-1: CCPxCON: CCPx CONTROL REGISTER

| R/W-00 | 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 |
|---|---|---|---|---|---|---|---|
| PxM<1:0>(1) | | DCxB<1:0> | | CCPxM<3:0> | | | |
| bit 7 | | | | | | | bit 0 |

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

bit 7-6 **PxM<1:0>:** Enhanced PWM Output Configuration bits(1)

Capture mode:
Unused

Compare mode:
Unused

If CCPxM<3:2> = 00, 01, 10:
xx = PxA assigned as Capture/Compare input; PxB, PxC, PxD assigned as port pins

If CCPxM<3:2> = 11:
11 = Full-Bridge output reverse; PxB modulated; PxC active; PxA, PxD inactive
10 = Half-Bridge output; PxA, PxB modulated with dead-band control; PxC, PxD assigned as port pins
01 = Full-Bridge output forward; PxD modulated; PxA active; PxB, PxC inactive
00 = Single output; PxA modulated; PxB, PxC, PxD assigned as port pins

bit 5-4 **DCxB<1:0>:** PWM Duty Cycle Least Significant bits

Capture mode:
Unused

Compare mode:
Unused

PWM mode:
These bits are the two LSbs of the PWM duty cycle. The eight MSbs are found in CCPRxL.

bit 3-0 **CCPxM<3:0>:** ECCPx Mode Select bits

1011 = Compare mode; Special Event Trigger (ECCPx resets Timer, sets CCPxIF bit, starts A/D conversion if A/D module is enabled)(2)
1010 = Compare mode: generate software interrupt only; ECCPx pin reverts to I/O state
1001 = Compare mode: initialize ECCPx pin high; clear output on compare match (set CCPxIF)
1000 = Compare mode: initialize ECCPx pin low; set output on compare match (set CCPxIF)

0111 = Capture mode: every 16th rising edge
0110 = Capture mode: every 4th rising edge
0101 = Capture mode: every rising edge
0100 = Capture mode: every falling edge

0011 = Reserved
0010 = Compare mode: toggle output on match
0001 = Reserved
0000 = Capture/Compare/PWM off (resets ECCPx module)

CCP4/CCP5 only:
11xx = PWM mode

ECCP1/ECCP2/ECCP3 only:
1111 = PWM mode: PxA, PxC active-low; PxB, PxD active-low
1110 = PWM mode: PxA, PxC active-low; PxB, PxD active-high
1101 = PWM mode: PxA, PxC active-high; PxB, PxD active-low
1100 = PWM mode: PxA, PxC active-high; PxB, PxD active-high

**Note 1:** These bits are not implemented on CCP<5:4>.
**2:** A/D conversion start applies to CCP5 only.

**REGISTER 23-2: CCPTMRS0: PWM TIMER SELECTION CONTROL REGISTER 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 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| C4TSEL<1:0> | | C3TSEL<1:0> | | C2TSEL<1:0> | | C1TSEL<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 **C4TSEL<1:0>:** CCP4 Timer Selection
- 11 = Reserved
- 10 = CCP4 is based off Timer6 in PWM Mode
- 01 = CCP4 is based off Timer4 in PWM Mode
- 00 = CCP4 is based off Timer2 in PWM Mode

bit 5-4 **C3TSEL<1:0>:** CCP3 Timer Selection
- 11 = Reserved
- 10 = CCP3 is based off Timer6 in PWM Mode
- 01 = CCP3 is based off Timer4 in PWM Mode
- 00 = CCP3 is based off Timer2 in PWM Mode

bit 3-2 **C2TSEL<1:0>:** CCP2 Timer Selection
- 11 = Reserved
- 10 = CCP2 is based off Timer6 in PWM Mode
- 01 = CCP2 is based off Timer4 in PWM Mode
- 00 = CCP2 is based off Timer2 in PWM Mode

bit 1-0 **C1TSEL<1:0>:** CCP1 Timer Selection
- 11 = Reserved
- 10 = CCP1 is based off Timer6 in PWM Mode
- 01 = CCP1 is based off Timer4 in PWM Mode
- 00 = CCP1 is based off Timer2 in PWM Mode

**REGISTER 23-3: CCPTMRS1: PWM TIMER SELECTION CONTROL REGISTER 1**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | C5TSEL<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 7-2 **Unimplemented:** Read as '0'

bit 1-0 **C5TSEL<1:0>:** CCP5 Timer Selection
- 11 = Reserved
- 10 = CCP5 is based off Timer6 in PWM Mode
- 01 = CCP5 is based off Timer4 in PWM Mode
- 00 = CCP5 is based off Timer2 in PWM Mode


© 2010-2016 Microchip Technology Inc.

**REGISTER 23-4: CCPxAS: CCPX AUTO-SHUTDOWN 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/W-0/0 |
|---|---|---|---|---|---|---|---|
| CCPxASE | CCPxAS<2:0> | | | PSSxAC<1:0> | | PSSxBD<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 **CCPxASE:** CCPx Auto-Shutdown Event Status bit
1 = A shutdown event has occurred; CCPx outputs are in shutdown state
0 = CCPx outputs are operating

bit 6-4 **CCPxAS<2:0>:** CCPx Auto-Shutdown Source Select bits
111 = VIL on FLT0 pin or Comparator C1 or Comparator C2 high(1, 2)
110 = VIL on FLT0 pin or Comparator C2 high(1, 2)
101 = VIL on FLT0 pin or Comparator C1 high(1)
100 = VIL on FLT0 pin
011 = Either Comparator C1 or C2 high(1, 2)
010 = Comparator C2 output high(1, 2)
001 = Comparator C1 output high(1)
000 = Auto-shutdown is disabled

bit 3-2 **PSSxAC<1:0>:** Pins PxA and PxC Shutdown State Control bits
1x = Pins PxA and PxC tri-state
01 = Drive pins PxA and PxC to '1'
00 = Drive pins PxA and PxC to '0'

bit 1-0 **PSSxBD<1:0>:** Pins PxB and PxD Shutdown State Control bits
1x = Pins PxB and PxD tri-state
01 = Drive pins PxB and PxD to '1'
00 = Drive pins PxB and PxD to '0'

**Note 1:** If CxSYNC is enabled, the shutdown will be delayed by Timer1.
**2:** For PIC16F1946/47 devices in ECCP3 mode, CCPxAS uses C3 instead of C2.

**REGISTER 23-5: PWMxCON: ENHANCED PWM CONTROL REGISTER(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 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| PxRSEN | PxDC<6:0> | | | | | | |
| 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 **PxRSEN:** PWM Restart Enable bit

1 = Upon auto-shutdown, the CCPxASE bit clears automatically once the shutdown event goes away; the PWM restarts automatically

0 = Upon auto-shutdown, CCPxASE must be cleared in software to restart the PWM

bit 6-0 **PxDC<6:0>:** PWM Delay Count bits

PxDCx = Number of FOSC/4 (4 * TOSC) cycles between the scheduled time when a PWM signal **should** transition active and the **actual** time it transitions active

**Note 1:** Bit resets to '0' with Two-Speed Start-up and LP, XT or HS selected as the Oscillator mode or Fail-Safe mode is enabled.


© 2010-2016 Microchip Technology Inc.

**REGISTER 23-6: PSTRxCON: PWM STEERING CONTROL REGISTER(1)**

| U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-1/1 |
|---|---|---|---|---|---|---|---|
| — | — | — | STRxSYNC | STRxD | STRxC | STRxB | STRxA |
| 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 **STRxSYNC:** Steering Sync bit
1 = Output steering update occurs on next PWM period
0 = Output steering update occurs at the beginning of the instruction cycle boundary

bit 3 **STRxD:** Steering Enable bit D
1 = PxD pin has the PWM waveform with polarity control from CCPxM<1:0>
0 = PxD pin is assigned to port pin

bit 2 **STRxC:** Steering Enable bit C
1 = PxC pin has the PWM waveform with polarity control from CCPxM<1:0>
0 = PxC pin is assigned to port pin

bit 1 **STRxB:** Steering Enable bit B
1 = PxB pin has the PWM waveform with polarity control from CCPxM<1:0>
0 = PxB pin is assigned to port pin

bit 0 **STRxA:** Steering Enable bit A
1 = PxA pin has the PWM waveform with polarity control from CCPxM<1:0>
0 = PxA pin is assigned to port pin

**Note 1:** The PWM Steering mode is available only when the CCPxCON register bits CCPxM<3:2> = 11 and PxM<1:0> = 00.

## 24.0 MASTER SYNCHRONOUS SERIAL PORT (MSSP1 AND MSSP2) MODULE

### 24.1 Master SSPx (MSSPx) Module Overview

The Master Synchronous Serial Port (MSSPx) module is a serial interface useful for communicating with other peripheral or microcontroller devices. These peripheral devices may be serial EEPROMs, shift registers, display drivers, A/D converters, etc. The MSSPx module can operate in one of two modes:

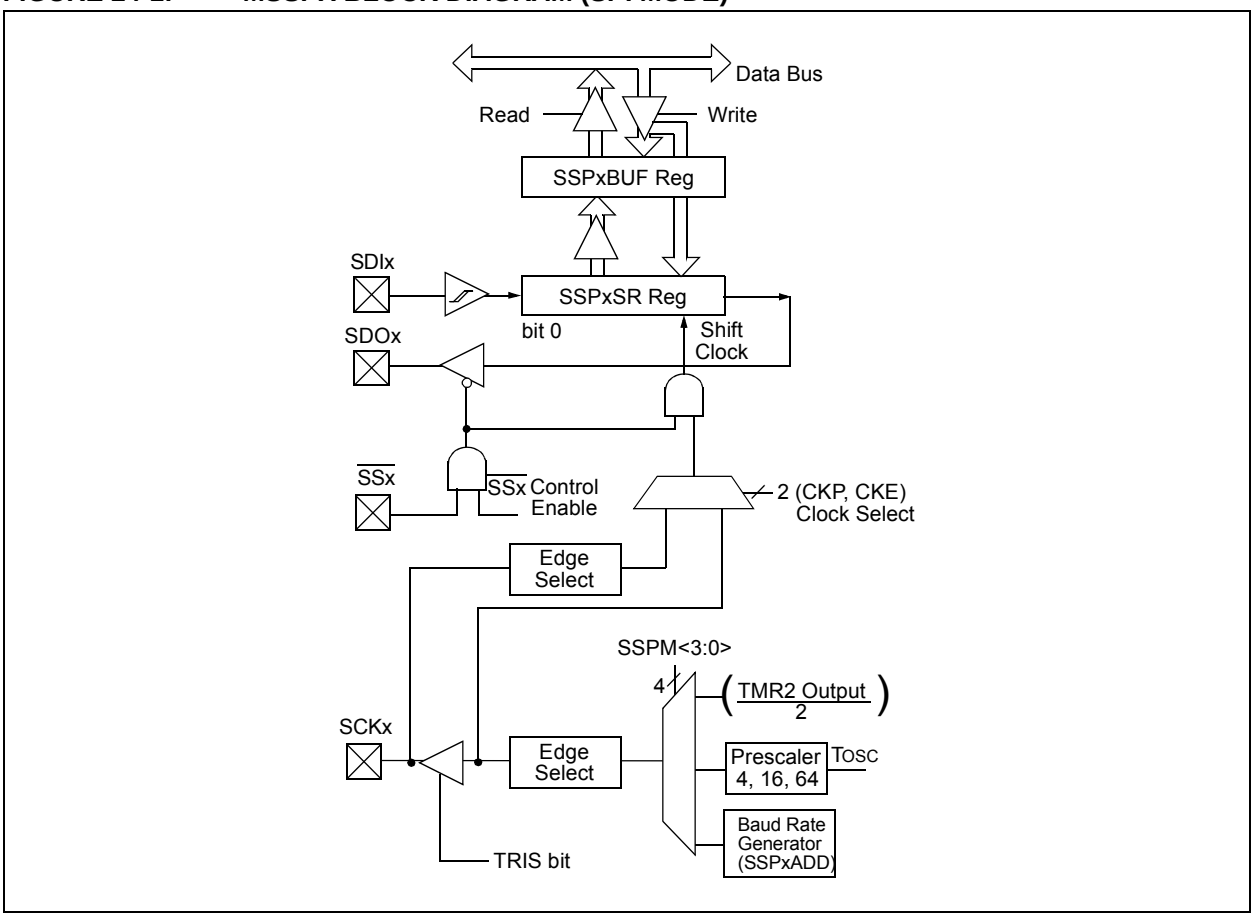
- Serial Peripheral Interface (SPI)
- Inter-Integrated Circuit ($I^2C$)

The SPI interface supports the following modes and features:

- Master mode
- Slave mode
- Clock Parity
- Slave Select Synchronization (Slave mode only)
- Daisy-chain connection of slave devices

Figure 24-1 is a block diagram of the SPI interface module.

**FIGURE 24-1: MSSPX BLOCK DIAGRAM (SPI MODE)**





© 2010-2016 Microchip Technology Inc.

The $I^2C$ interface supports the following modes and features:

- Master mode
- Slave mode
- Byte NACKing (Slave mode)
- Limited Multi-master support
- 7-bit and 10-bit addressing
- Start and Stop interrupts
- Interrupt masking
- Clock stretching
- Bus collision detection
- General call address matching
- Address masking
- Address Hold and Data Hold modes
- Selectable SDAx hold times

Figure 24-2 is a block diagram of the $I^2C$ interface module in Master mode. Figure 24-3 is a diagram of the $I^2C$ interface module in Slave mode.

The PIC16F1947 has two MSSP modules, MSSP1 and MSSP2, each module operating independently from the other.

**Note 1:** In devices with more than one MSSP module, it is very important to pay close attention to SSPxCONx register names. SSP1CON1 and SSP1CON2 registers control different operational aspects of the same module, while SSP1CON1 and SSP2CON1 control the same features for two different modules.

**2:** Throughout this section, generic references to an MSSP module in any of its operating modes may be interpreted as being equally applicable to MSSP1 or MSSP2. Register names, module I/O signals, and bit names may use the generic designator 'x' to indicate the use of a numeral to distinguish a particular module when required.

**FIGURE 24-2: MSSPx BLOCK DIAGRAM ($I^2C$ MASTER MODE)**

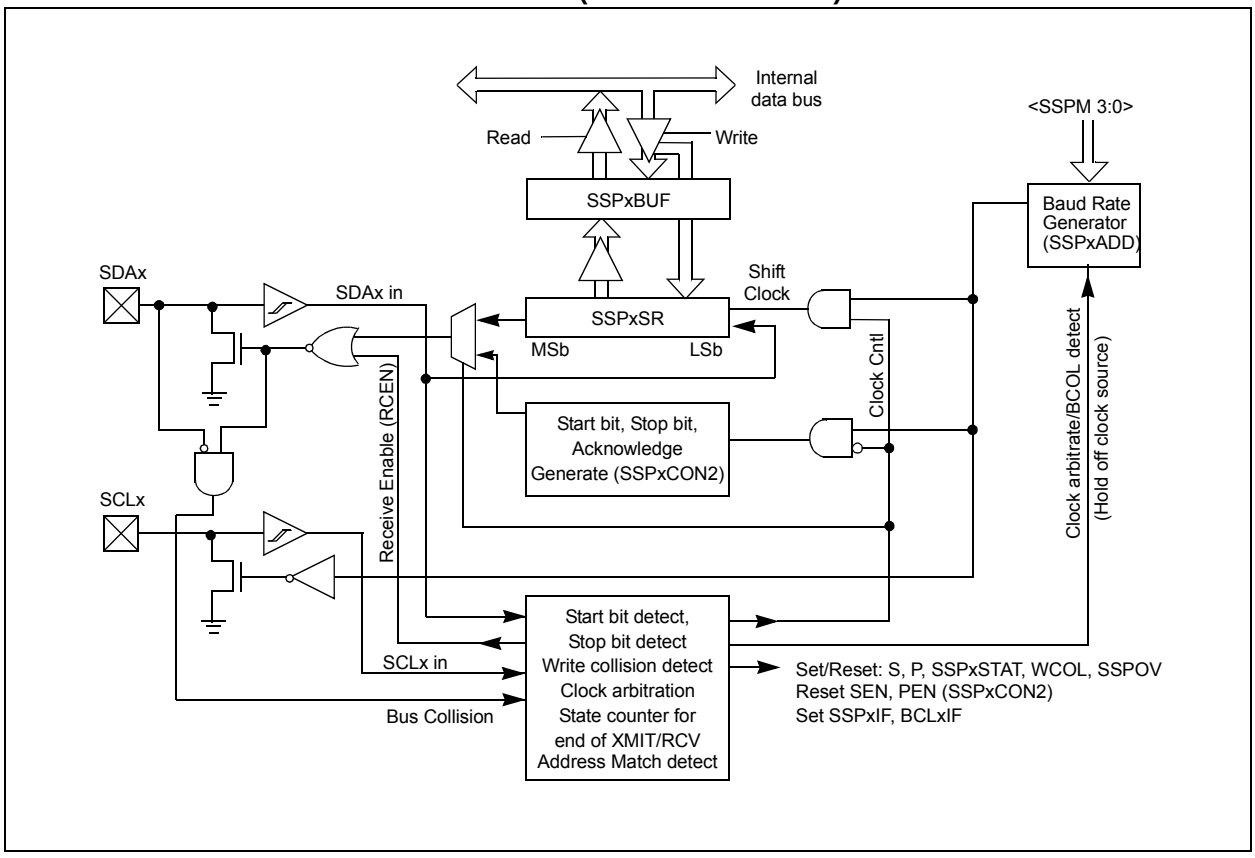
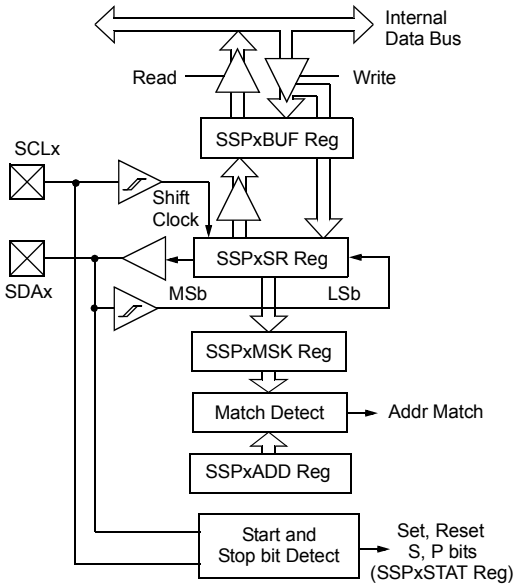


**FIGURE 24-3: MSSPx BLOCK DIAGRAM ($I^2C$ SLAVE MODE)**





© 2010-2016 Microchip Technology Inc.

## 24.2 SPI Mode Overview

The Serial Peripheral Interface (SPI) bus is a synchronous serial data communication bus that operates in Full Duplex mode. Devices communicate in a master/slave environment where the master device initiates the communication. A slave device is controlled through a chip select known as Slave Select.

The SPI bus specifies four signal connections:

- Serial Clock (SCKx)
- Serial Data Out (SDOx)
- Serial Data In (SDIx)
- Slave Select ($\overline{SS}$x)

Figure 24-1 shows the block diagram of the MSSPx module when operating in SPI Mode.

The SPI bus operates with a single master device and one or more slave devices. When multiple slave devices are used, an independent Slave Select connection is required from the master device to each slave device.

Figure 24-4 shows a typical connection between a master device and multiple slave devices.

The master selects only one slave at a time. Most slave devices have tri-state outputs so their output signal appears disconnected from the bus when they are not selected.

Transmissions involve two shift registers, eight bits in size, one in the master and one in the slave. With either the master or the slave device, data is always shifted out one bit at a time, with the Most Significant bit (MSb) shifted out first. At the same time, a new Least Significant bit (LSb) is shifted into the same register.

Figure 24-5 shows a typical connection between two processors configured as master and slave devices.

Data is shifted out of both shift registers on the programmed clock edge and latched on the opposite edge of the clock.

The master device transmits information out on its SDOx output pin which is connected to, and received by, the slave's SDIx input pin. The slave device transmits information out on its SDOx output pin, which is connected to, and received by, the master's SDIx input pin.

To begin communication, the master device first sends out the clock signal. Both the master and the slave devices should be configured for the same clock polarity.

The master device starts a transmission by sending out the MSb from its shift register. The slave device reads this bit from that same line and saves it into the LSb position of its shift register.

During each SPI clock cycle, a full duplex data transmission occurs. This means that while the master device is sending out the MSb from its shift register (on its SDOx pin) and the slave device is reading this bit and saving it as the LSb of its shift register, that the slave device is also sending out the MSb from its shift register (on its SDOx pin) and the master device is reading this bit and saving it as the LSb of its shift register.

After eight bits have been shifted out, the master and slave have exchanged register values.

If there is more data to exchange, the shift registers are loaded with new data and the process repeats itself.

Whether the data is meaningful or not (dummy data), depends on the application software. This leads to three scenarios for data transmission:

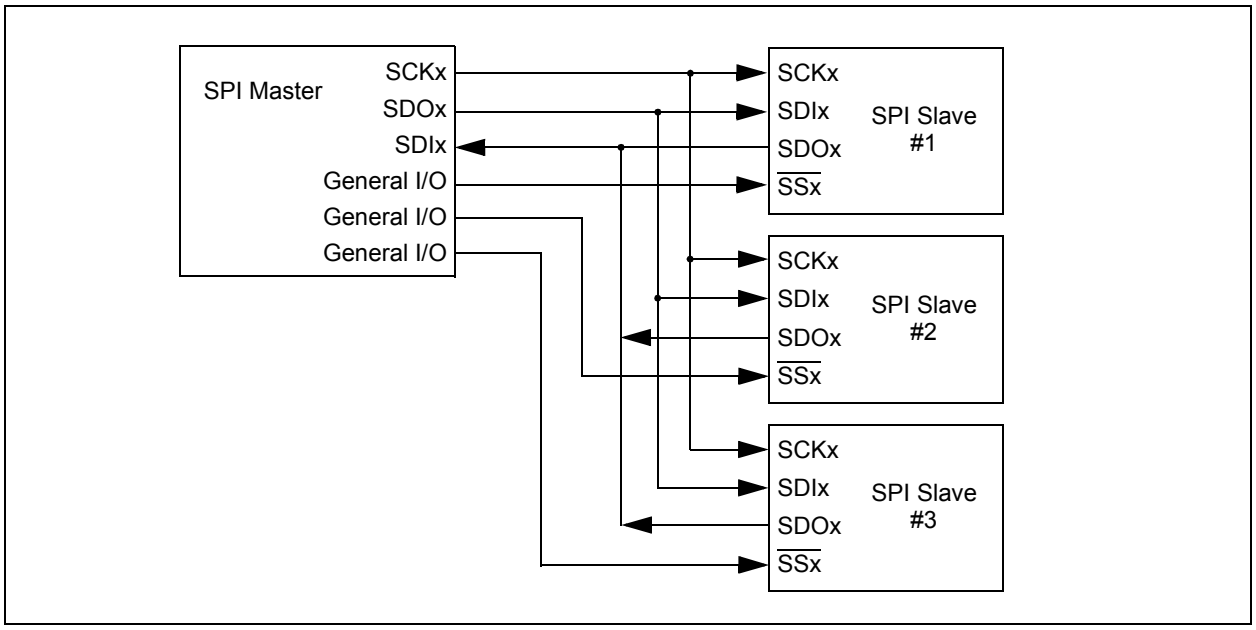
- Master sends useful data and slave sends dummy data.
- Master sends useful data and slave sends useful data.
- Master sends dummy data and slave sends useful data.

Transmissions may involve any number of clock cycles. When there is no more data to be transmitted, the master stops sending the clock signal and it deselects the slave.

Every slave device connected to the bus that has not been selected through its slave select line must disregard the clock and transmission signals and must not transmit out any data of its own.

© 2010-2016 Microchip Technology Inc.

**FIGURE 24-4: SPI MASTER AND MULTIPLE SLAVE CONNECTION**




### 24.2.1 SPI MODE REGISTERS

The MSSPx module has five registers for SPI mode operation. These are:

- MSSPx STATUS register (SSPxSTAT)
- MSSPx Control Register 1 (SSPxCON1)
- MSSPx Control Register 3 (SSPxCON3)
- MSSPx Data Buffer register (SSPxBUF)
- MSSPx Address register (SSPxADD)
- MSSPx Shift register (SSPxSR) (Not directly accessible)

SSPxCON1 and SSPxSTAT are the control and STATUS registers in SPI mode operation. The SSPxCON1 register is readable and writable. The lower six bits of the SSPxSTAT are read-only. The upper two bits of the SSPxSTAT are read/write.

In SPI master mode, SSPxADD can be loaded with a value used in the Baud Rate Generator. More information on the Baud Rate Generator is available in **Section 24.7 "Baud Rate Generator"**.

SSPxSR is the shift register used for shifting data in and out. SSPxBUF provides indirect access to the SSPxSR register. SSPxBUF is the buffer register to which data bytes are written, and from which data bytes are read.

In receive operations, SSPxSR and SSPxBUF together create a buffered receiver. When SSPxSR receives a complete byte, it is transferred to SSPxBUF and the SSPxIF interrupt is set.

During transmission, the SSPxBUF is not buffered. A write to SSPxBUF will write to both SSPxBUF and SSPxSR.


© 2010-2016 Microchip Technology Inc.

### 24.2.2 SPI MODE OPERATION

When initializing the SPI, several options need to be specified. This is done by programming the appropriate control bits (SSPxCON1<5:0> and SSPxSTAT<7:6>). These control bits allow the following to be specified:

- Master mode (SCKx is the clock output)
- Slave mode (SCKx is the clock input)
- Clock Polarity (Idle state of SCKx)
- Data Input Sample Phase (middle or end of data output time)
- Clock Edge (output data on rising/falling edge of SCKx)
- Clock Rate (Master mode only)
- Slave Select mode (Slave mode only)

To enable the serial port, SSPx Enable bit, SSPEN of the SSPxCON1 register, must be set. To reset or reconfigure SPI mode, clear the SSPEN bit, re-initialize the SSPxCONx registers and then set the SSPEN bit. This configures the SDIx, SDOx, SCKx and $\overline{SSx}$ pins as serial port pins. For the pins to behave as the serial port function, some must have their data direction bits (in the TRIS register) appropriately programmed as follows:

- SDIx must have corresponding TRIS bit set
- SDOx must have corresponding TRIS bit cleared
- SCKx (Master mode) must have corresponding TRIS bit cleared
- SCKx (Slave mode) must have corresponding TRIS bit set
- $\overline{SSx}$ must have corresponding TRIS bit set

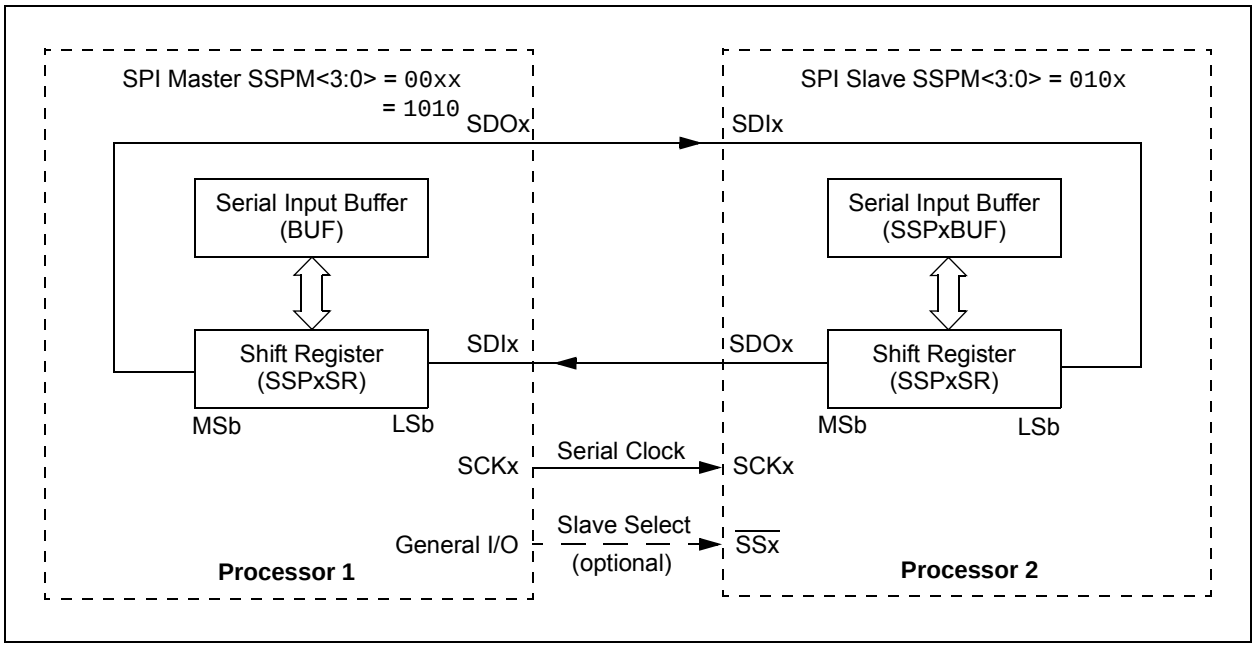
Any serial port function that is not desired may be overridden by programming the corresponding data direction (TRIS) register to the opposite value.

The MSSPx consists of a transmit/receive shift register (SSPxSR) and a buffer register (SSPxBUF). The SSPxSR shifts the data in and out of the device, MSb first. The SSPxBUF holds the data that was written to the SSPxSR until the received data is ready. Once the eight bits of data have been received, that byte is moved to the SSPxBUF register. Then, the Buffer Full Detect bit, BF of the SSPxSTAT register, and the interrupt flag bit, SSPxIF, are set. This double-buffering of the received data (SSPxBUF) allows the next byte to start reception before reading the data that was just received. Any write to the SSPxBUF register during transmission/reception of data will be ignored and the write collision detect bit WCOL of the SSPxCON1 register, will be set. User software must clear the WCOL bit to allow the following write(s) to the SSPxBUF register to complete successfully.

When the application software is expecting to receive valid data, the SSPxBUF should be read before the next byte of data to transfer is written to the SSPxBUF. The Buffer Full bit, BF of the SSPxSTAT register, indicates when SSPxBUF has been loaded with the received data (transmission is complete). When the SSPxBUF is read, the BF bit is cleared. This data may be irrelevant if the SPI is only a transmitter. Generally, the MSSPx interrupt is used to determine when the transmission/reception has completed. If the interrupt method is not going to be used, then software polling can be done to ensure that a write collision does not occur.

The SSPxSR is not directly readable or writable and can only be accessed by addressing the SSPxBUF register. Additionally, the SSPxSTAT register indicates the various Status conditions.

**FIGURE 24-5: SPI MASTER/SLAVE CONNECTION**





© 2010-2016 Microchip Technology Inc.

### 24.2.3 SPI MASTER MODE

The master can initiate the data transfer at any time because it controls the SCKx line. The master determines when the slave (Processor 2, Figure 24-5) is to broadcast data by the software protocol.

In Master mode, the data is transmitted/received as soon as the SSPxBUF register is written to. If the SPI is only going to receive, the SDOx output could be disabled (programmed as an input). The SSPxSR register will continue to shift in the signal present on the SDIx pin at the programmed clock rate. As each byte is received, it will be loaded into the SSPxBUF register as if a normal received byte (interrupts and Status bits appropriately set).

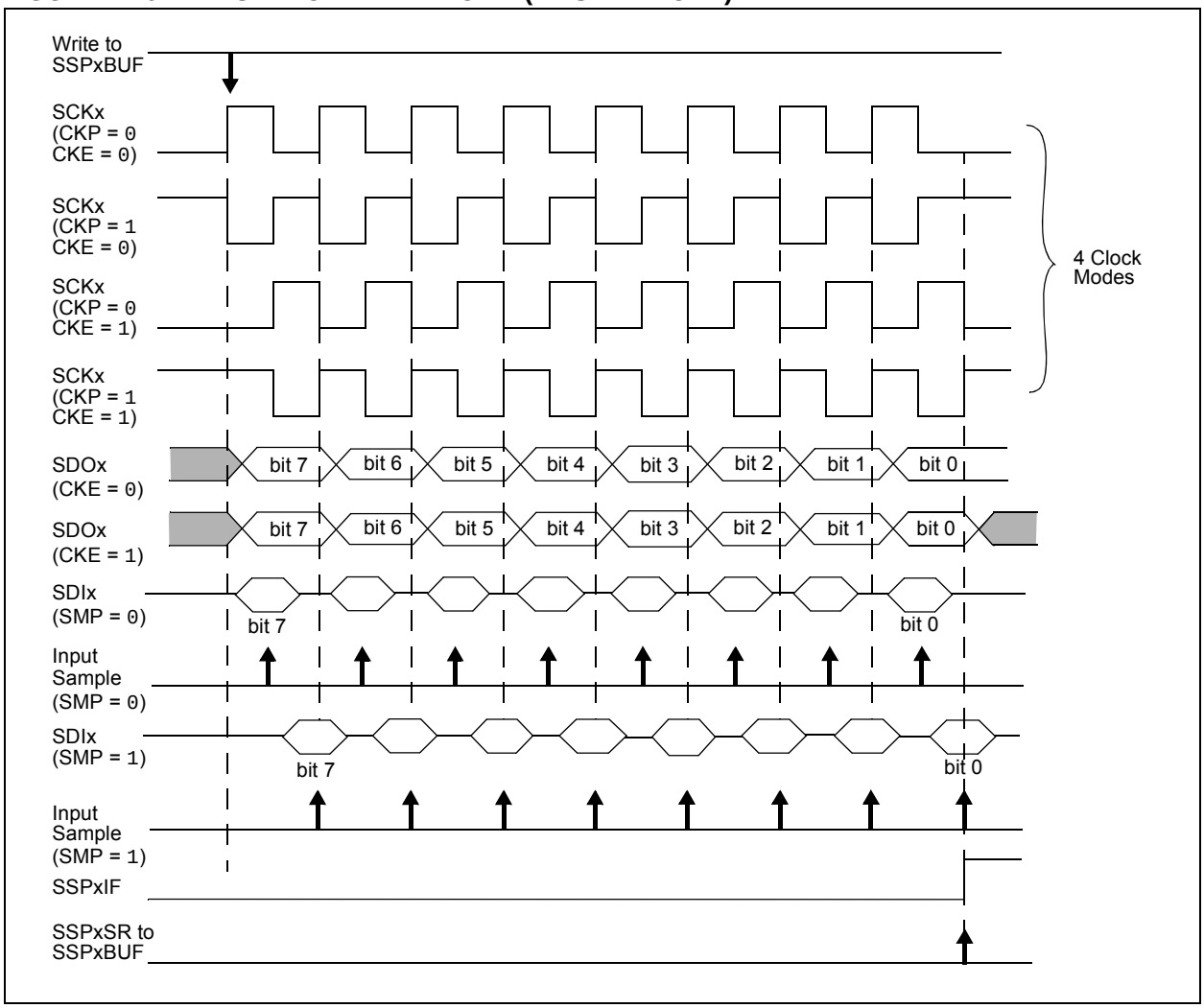
The clock polarity is selected by appropriately programming the CKP bit of the SSPxCON1 register and the CKE bit of the SSPxSTAT register. This then, would give waveforms for SPI communication as shown in Figure 24-6, Figure 24-8 and Figure 24-9, where the MSb is transmitted first. In Master mode, the SPI clock rate (bit rate) is user programmable to be one of the following:

- FOSC/4 (or TCY)
- FOSC/16 (or 4 * TCY)
- FOSC/64 (or 16 * TCY)
- Timer2 output/2
- Fosc/(4 * (SSPxADD + 1))

Figure 24-6 shows the waveforms for Master mode.

When the CKE bit is set, the SDOx data is valid before there is a clock edge on SCKx. The change of the input sample is shown based on the state of the SMP bit. The time when the SSPxBUF is loaded with the received data is shown.

**FIGURE 24-6: SPI MODE WAVEFORM (MASTER MODE)**



### 24.2.4 SPI SLAVE MODE

In Slave mode, the data is transmitted and received as external clock pulses appear on SCKx. When the last bit is latched, the SSPxIF interrupt flag bit is set.

Before enabling the module in SPI Slave mode, the clock line must match the proper Idle state. The clock line can be observed by reading the SCKx pin. The Idle state is determined by the CKP bit of the SSPxCON1 register.

While in Slave mode, the external clock is supplied by the external clock source on the SCKx pin. This external clock must meet the minimum high and low times as specified in the electrical specifications.

While in Sleep mode, the slave can transmit/receive data. The shift register is clocked from the SCKx pin input and when a byte is received, the device will generate an interrupt. If enabled, the device will wake-up from Sleep.

#### 24.2.4.1 Daisy-Chain Configuration

The SPI bus can sometimes be connected in a daisy-chain configuration. The first slave output is connected to the second slave input, the second slave output is connected to the third slave input, and so on. The final slave output is connected to the master input. Each slave sends out, during a second group of clock pulses, an exact copy of what was received during the first group of clock pulses. The whole chain acts as one large communication shift register. The daisy-chain feature only requires a single Slave Select line from the master device.

Figure 24-7 shows the block diagram of a typical daisy-chain connection when operating in SPI Mode.

In a daisy-chain configuration, only the most recent byte on the bus is required by the slave. Setting the BOEN bit of the SSPxCON3 register will enable writes to the SSPxBUF register, even if the previous byte has not been read. This allows the software to ignore data that may not apply to it.

### 24.2.5 SLAVE SELECT SYNCHRONIZATION

The Slave Select can also be used to synchronize communication. The Slave Select line is held high until the master device is ready to communicate. When the Slave Select line is pulled low, the slave knows that a new transmission is starting.

If the slave fails to receive the communication properly, it will be reset at the end of the transmission, when the Slave Select line returns to a high state. The slave is then ready to receive a new transmission when the Slave Select line is pulled low again. If the Slave Select line is not used, there is a risk that the slave will eventually become out of sync with the master. If the slave misses a bit, it will always be one bit off in future transmissions. Use of the Slave Select line allows the slave and master to align themselves at the beginning of each transmission.

The $\overline{SSx}$ pin allows a Synchronous Slave mode. The SPI must be in Slave mode with $\overline{SSx}$ pin control enabled (SSPxCON1<3:0> = 0100).

When the $\overline{SSx}$ pin is low, transmission and reception are enabled and the SDOx pin is driven.

When the $\overline{SSx}$ pin goes high, the SDOx pin is no longer driven, even if in the middle of a transmitted byte and becomes a floating output. External pull-up/pull-down resistors may be desirable depending on the application.

**Note 1:** When the SPI is in Slave mode with $\overline{SSx}$ pin control enabled (SSPxCON1<3:0> = 0100), the SPI module will reset if the $\overline{SSx}$ pin is set to VDD.

**2:** When the SPI is used in Slave mode with CKE set; the user must enable $\overline{SSx}$ pin control.

**3:** While operated in SPI Slave mode, the SMP bit of the SSPxSTAT register must remain clear.

When the SPI module resets, the bit counter is forced to '0'. This can be done by either forcing the $\overline{SSx}$ pin to a high level or clearing the SSPEN bit.


© 2010-2016 Microchip Technology Inc.

**FIGURE 24-7: SPI DAISY-CHAIN CONNECTION**

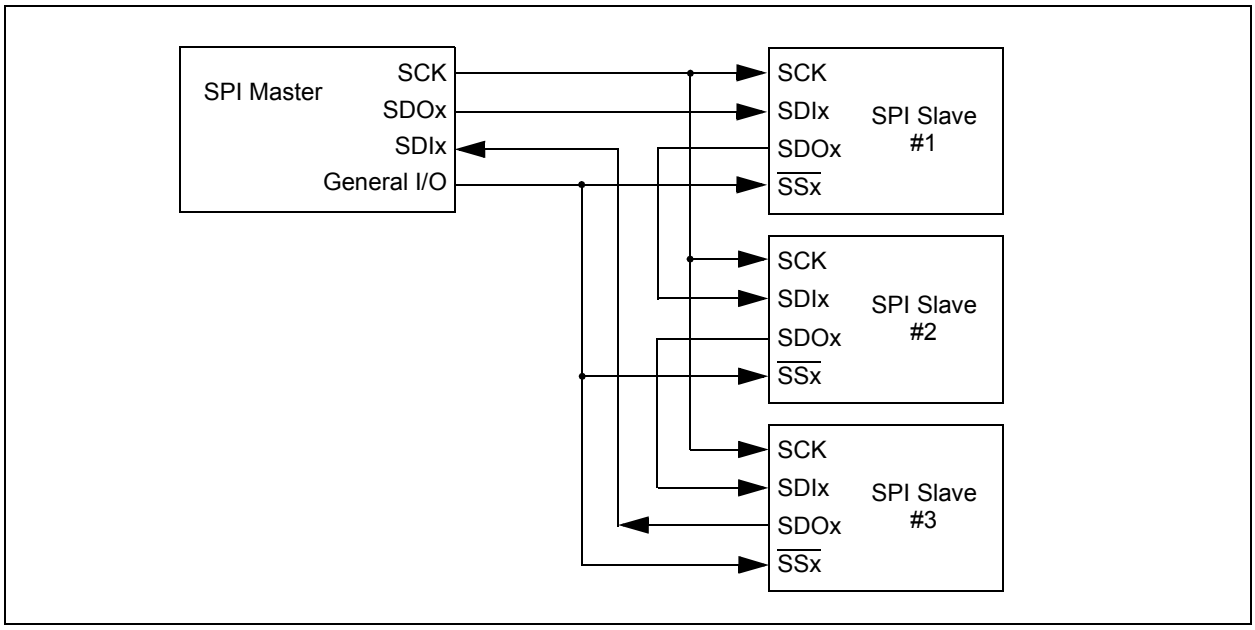



**FIGURE 24-8: SLAVE SELECT SYNCHRONOUS WAVEFORM**

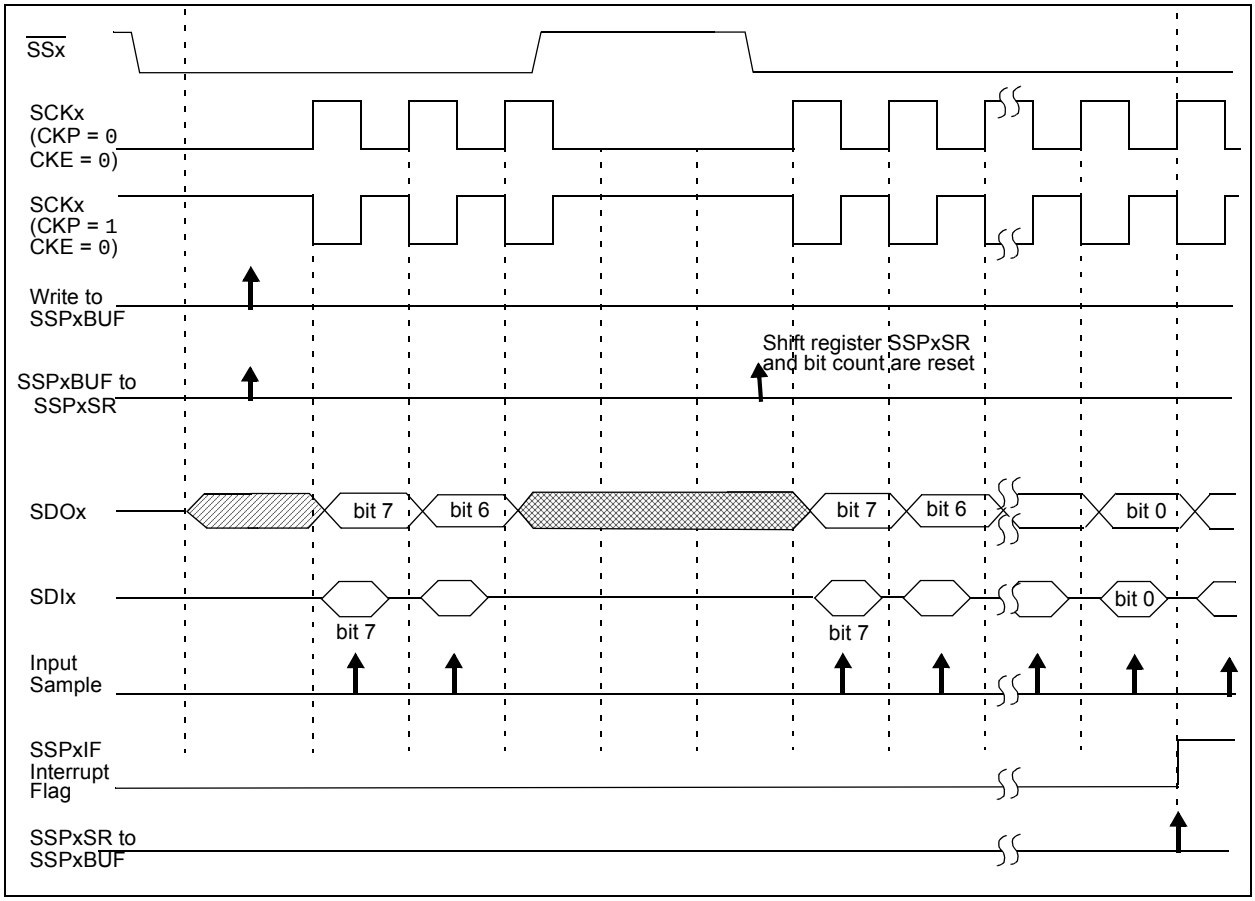


**FIGURE 24-9: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 0)**

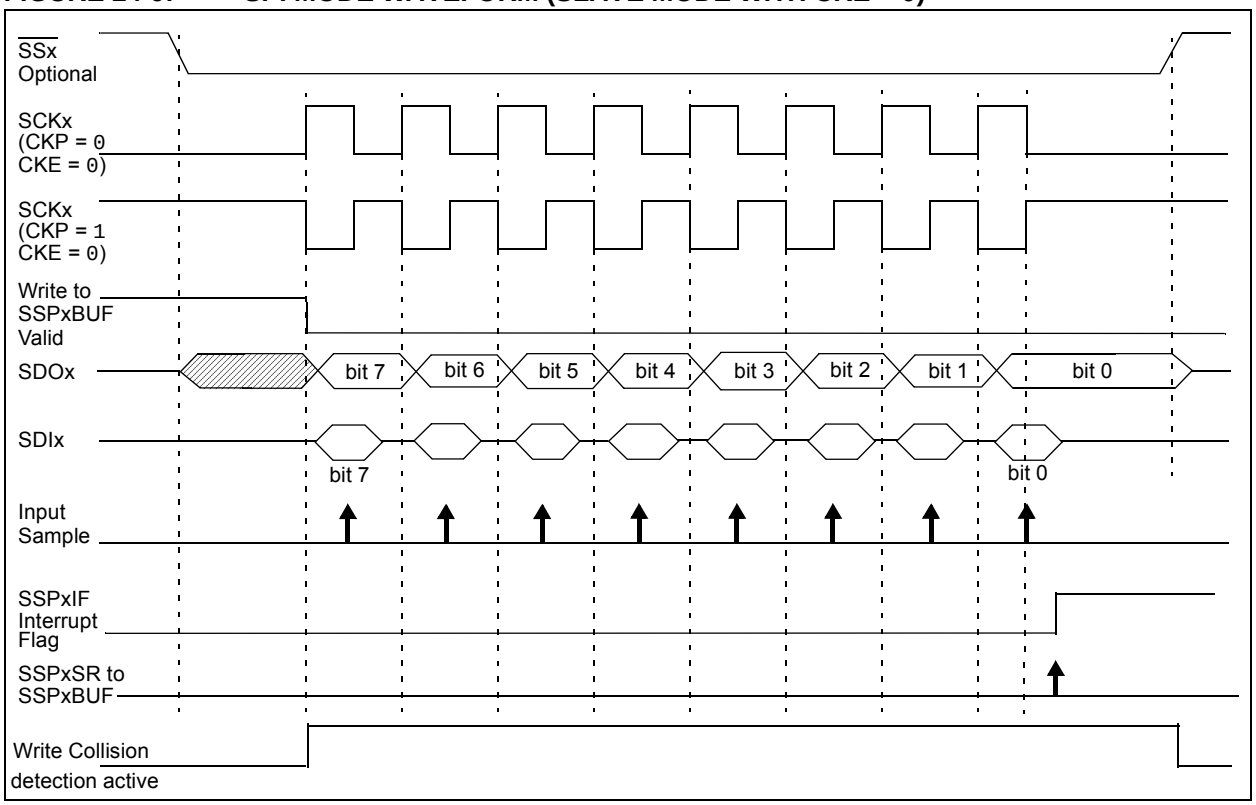
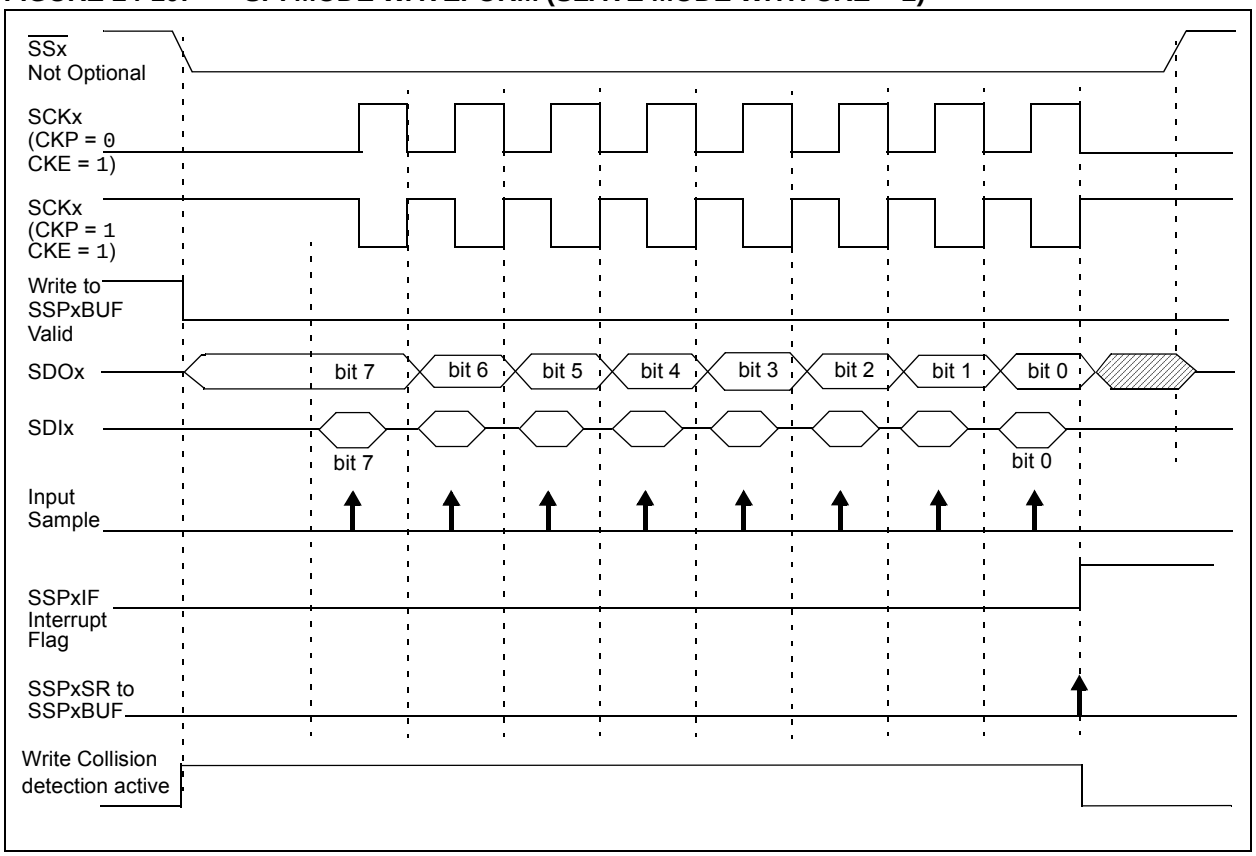



**FIGURE 24-10: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 1)**





© 2010-2016 Microchip Technology Inc.

### 24.2.6 SPI OPERATION IN SLEEP MODE

In SPI Master mode, module clocks may be operating at a different speed than when in Full Power mode; in the case of the Sleep mode, all clocks are halted.

Special care must be taken by the user when the MSSPx clock is much faster than the system clock.

In Slave mode, when MSSPx interrupts are enabled, after the master completes sending data, an MSSPx interrupt will wake the controller from Sleep.

If an exit from Sleep mode is not desired, MSSPx interrupts should be disabled.

In SPI Master mode, when the Sleep mode is selected, all module clocks are halted and the transmission/reception will remain in that state until the device wakes. After the device returns to Run mode, the module will resume transmitting and receiving data.

In SPI Slave mode, the SPI Transmit/Receive Shift register operates asynchronously to the device. This allows the device to be placed in Sleep mode and data to be shifted into the SPI Transmit/Receive Shift register. When all eight bits have been received, the MSSPx interrupt flag bit will be set and if enabled, will wake the device.

**TABLE 24-1: SUMMARY OF REGISTERS ASSOCIATED WITH SPI OPERATION**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 91 |
| PIE4 | — | — | RC2IE | TX2IE | — | — | BCL2IE | SSP2IE | 94 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 95 |
| PIR4 | — | — | RC2IF | TX2IF | — | — | BCL2IF | SSP2IF | 98 |
| SSP1BUF | Synchronous Serial Port Receive Buffer/Transmit Register | | | | | | | | 236* |
| SSP2BUF | Synchronous Serial Port Receive Buffer/Transmit Register | | | | | | | | 236* |
| SSP1CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM<3:0> | | | | 282 |
| SSP1CON3 | ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN | 285 |
| SSP1STAT | SMP | CKE | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF | 281 |
| SSP2CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM<3:0> | | | | 282 |
| SSP2CON3 | ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN | 285 |
| SSP2STAT | SMP | CKE | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF | 281 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 131 |
| TRISD | TRISD7 | TRISD6 | TRISD5 | TRISD4 | TRISD3 | TRISD2 | TRISD1 | TRISD0 | 134 |
| TRISF | TRISF7 | TRISF6 | TRISF5 | TRISF4 | TRISF3 | TRISF2 | TRISF1 | TRISF0 | 140 |

**Legend:** — = Unimplemented location, read as '0'. Shaded cells are not used by the MSSPx in SPI mode.
* Page provides register information.

## 24.3 I²C MODE OVERVIEW

The Inter-Integrated Circuit Bus (I²C) is a multi-master serial data communication bus. Devices communicate in a master/slave environment where the master devices initiate the communication. A Slave device is controlled through addressing.

The I²C bus specifies two signal connections:

- Serial Clock (SCLx)
- Serial Data (SDAx)

Figure 24-2 and Figure 24-3 show the block diagrams of the MSSPx module when operating in I²C mode.

Both the SCLx and SDAx connections are bidirectional open-drain lines, each requiring pull-up resistors for the supply voltage. Pulling the line to ground is considered a logical zero and letting the line float is considered a logical one.

Figure 24-11 shows a typical connection between two processors configured as master and slave devices.

The I²C bus can operate with one or more master devices and one or more slave devices.

There are four potential modes of operation for a given device:

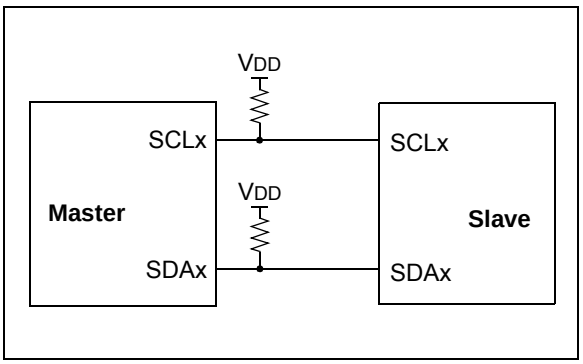
- Master Transmit mode
  (master is transmitting data to a slave)
- Master Receive mode
  (master is receiving data from a slave)
- Slave Transmit mode
  (slave is transmitting data to a master)
- Slave Receive mode
  (slave is receiving data from the master)

To begin communication, a master device starts out in Master Transmit mode. The master device sends out a Start bit followed by the address byte of the slave it intends to communicate with. This is followed by a single Read/Write bit, which determines whether the master intends to transmit to or receive data from the slave device.

If the requested slave exists on the bus, it will respond with an Acknowledge bit, otherwise known as an $\overline{\text{ACK}}$. The master then continues in either Transmit mode or Receive mode and the slave continues in the complement, either in Receive mode or Transmit mode, respectively.

A Start bit is indicated by a high-to-low transition of the SDAx line while the SCLx line is held high. Address and data bytes are sent out, Most Significant bit (MSb) first. The Read/Write bit is sent out as a logical one when the master intends to read data from the slave, and is sent out as a logical zero when it intends to write data to the slave.

**FIGURE 24-11: I²C MASTER/ SLAVE CONNECTION**




The Acknowledge bit ($\overline{\text{ACK}}$) is an active-low signal, which holds the SDAx line low to indicate to the transmitter that the slave device has received the transmitted data and is ready to receive more.

The transition of a data bit is always performed while the SCLx line is held low. Transitions that occur while the SCLx line is held high are used to indicate Start and Stop bits.

If the master intends to write to the slave, then it repeatedly sends out a byte of data, with the slave responding after each byte with an $\overline{\text{ACK}}$ bit. In this example, the master device is in Master Transmit mode and the slave is in Slave Receive mode.

If the master intends to read from the slave, then it repeatedly receives a byte of data from the slave, and responds after each byte with an $\overline{\text{ACK}}$ bit. In this example, the master device is in Master Receive mode and the slave is Slave Transmit mode.

On the last byte of data communicated, the master device may end the transmission by sending a Stop bit. If the master device is in Receive mode, it sends the Stop bit in place of the last $\overline{\text{ACK}}$ bit. A Stop bit is indicated by a low-to-high transition of the SDAx line while the SCLx line is held high.

In some cases, the master may want to maintain control of the bus and re-initiate another transmission. If so, the master device may send another Start bit in place of the Stop bit or last $\overline{\text{ACK}}$ bit when it is in receive mode.

The I²C bus specifies three message protocols:

- Single message where a master writes data to a slave.
- Single message where a master reads data from a slave.
- Combined message where a master initiates a minimum of two writes, or two reads, or a combination of writes and reads, to one or more slaves.


© 2010-2016 Microchip Technology Inc.

When one device is transmitting a logical one, or letting the line float, and a second device is transmitting a logical zero, or holding the line low, the first device can detect that the line is not a logical one. This detection, when used on the SCLx line, is called clock stretching. Clock stretching gives slave devices a mechanism to control the flow of data. When this detection is used on the SDAx line, it is called arbitration. Arbitration ensures that there is only one master device communicating at any single time.

### 24.3.1 CLOCK STRETCHING

When a slave device has not completed processing data, it can delay the transfer of more data through the process of clock stretching. An addressed slave device may hold the SCLx clock line low after receiving or sending a bit, indicating that it is not yet ready to continue. The master that is communicating with the slave will attempt to raise the SCLx line in order to transfer the next bit, but will detect that the clock line has not yet been released. Because the SCLx connection is open-drain, the slave has the ability to hold that line low until it is ready to continue communicating.

Clock stretching allows receivers that cannot keep up with a transmitter to control the flow of incoming data.

### 24.3.2 ARBITRATION

Each master device must monitor the bus for Start and Stop bits. If the device detects that the bus is busy, it cannot begin a new message until the bus returns to an Idle state.

However, two master devices may try to initiate a transmission on or about the same time. When this occurs, the process of arbitration begins. Each transmitter checks the level of the SDAx data line and compares it to the level that it expects to find. The first transmitter to observe that the two levels do not match, loses arbitration, and must stop transmitting on the SDAx line.

For example, if one transmitter holds the SDAx line to a logical one (lets it float) and a second transmitter holds it to a logical zero (pulls it low), the result is that the SDAx line will be low. The first transmitter then observes that the level of the line is different than expected and concludes that another transmitter is communicating.

The first transmitter to notice this difference is the one that loses arbitration and must stop driving the SDAx line. If this transmitter is also a master device, it also must stop driving the SCLx line. It then can monitor the lines for a Stop condition before trying to reissue its transmission. In the meantime, the other device that has not noticed any difference between the expected and actual levels on the SDAx line continues with its original transmission. It can do so without any complications, because so far, the transmission appears exactly as expected with no other transmitter disturbing the message.

Slave Transmit mode can also be arbitrated, when a master addresses multiple slaves, but this is less common.

If two master devices are sending a message to two different slave devices at the address stage, the master sending the lower slave address always wins arbitration. When two master devices send messages to the same slave address, and addresses can sometimes refer to multiple slaves, the arbitration process must continue into the data stage.

Arbitration usually occurs very rarely, but it is a necessary process for proper multi-master support.

## 24.4 I²C Mode Operation

All MSSPx I²C communication is byte-oriented and shifts out MSb first. Six SFR registers and two interrupt flags interface the module with the PIC® microcontroller and user software. Two pins, SDAx and SCLx, are exercised by the module to communicate with other external I²C devices.

### 24.4.1 BYTE FORMAT

All communication in I²C is done in 9-bit segments. A byte is sent from a Master to a Slave or vice-versa, followed by an Acknowledge bit sent back. After the eighth falling edge of the SCLx line, the device outputting data on the SDAx changes that pin to an input and reads in an acknowledge value on the next clock pulse.

The clock signal, SCLx, is provided by the master. Data is valid to change while the SCLx signal is low, and sampled on the rising edge of the clock. Changes on the SDAx line while the SCLx line is high define special conditions on the bus, explained below.

### 24.4.2 DEFINITION OF I²C TERMINOLOGY

There is language and terminology in the description of I²C communication that have definitions specific to I²C. That word usage is defined below and may be used in the rest of this document without explanation. This table was adapted from the Philips I²C specification.

### 24.4.3 SDAx AND SCLx PINS

Selection of any I²C mode with the SSPEN bit set, forces the SCLx and SDAx pins to be open-drain. These pins should be set by the user to inputs by setting the appropriate TRIS bits.

**Note:** Data is tied to output zero when an I²C mode is enabled.

### 24.4.4 SDAx HOLD TIME

The hold time of the SDAx pin is selected by the SDAHT bit of the SSPxCON3 register. Hold time is the time SDAx is held valid after the falling edge of SCLx. Setting the SDAHT bit selects a longer 300 ns minimum hold time and may help on buses with large capacitance.

**TABLE 24-2: I²C BUS TERMS**

| TERM | Description |
|---|---|
| Transmitter | The device which shifts data out onto the bus. |
| Receiver | The device which shifts data in from the bus. |
| Master | The device that initiates a transfer, generates clock signals and terminates a transfer. |
| Slave | The device addressed by the master. |
| Multi-master | A bus with more than one device that can initiate data transfers. |
| Arbitration | Procedure to ensure that only one master at a time controls the bus. Winning arbitration ensures that the message is not corrupted. |
| Synchronization | Procedure to synchronize the clocks of two or more devices on the bus. |
| Idle | No master is controlling the bus, and both SDAx and SCLx lines are high. |
| Active | Any time one or more master devices are controlling the bus. |
| Addressed Slave | Slave device that has received a matching address and is actively being clocked by a master. |
| Matching Address | Address byte that is clocked into a slave that matches the value stored in SSPxADD. |
| Write Request | Slave receives a matching address with R/$\overline{W}$ bit clear, and is ready to clock in data. |
| Read Request | Master sends an address byte with the R/$\overline{W}$ bit set, indicating that it wishes to clock data out of the Slave. This data is the next and all following bytes until a Restart or Stop. |
| Clock Stretching | When a device on the bus hold SCLx low to stall communication. |
| Bus Collision | Any time the SDAx line is sampled low by the module while it is outputting and expected high state. |


© 2010-2016 Microchip Technology Inc.

### 24.4.5 START CONDITION

The $I^2C$ specification defines a Start condition as a transition of SDAx from a high to a low state while SCLx line is high. A Start condition is always generated by the master and signifies the transition of the bus from an Idle to an Active state. Figure 24-10 shows waveforms for Start and Stop conditions.

A bus collision can occur on a Start condition if the module samples the SDAx line low before asserting it low. This does not conform to the $I^2C$ Specification that states no bus collision can occur on a Start.

### 24.4.6 STOP CONDITION

A Stop condition is a transition of the SDAx line from low-to-high state while the SCLx line is high.

**Note:** At least one SCLx low time must appear before a Stop is valid, therefore, if the SDAx line goes low then high again while the SCLx line stays high, only the Start condition is detected.

### 24.4.7 RESTART CONDITION

A Restart is valid any time that a Stop would be valid. A master can issue a Restart if it wishes to hold the bus after terminating the current transfer. A Restart has the same effect on the slave that a Start would, resetting all slave logic and preparing it to clock in an address. The master may want to address the same or another slave.

In 10-bit Addressing Slave mode, a Restart is required for the master to clock data out of the addressed slave. Once a slave has been fully addressed, matching both high and low address bytes, the master can issue a Restart and the high address byte with the R/$\overline{W}$ bit set. The slave logic will then hold the clock and prepare to clock out data.

After a full match with R/$\overline{W}$ clear in 10-bit mode, a prior match flag is set and maintained. Until a Stop condition, a high address with R/$\overline{W}$ clear, or high address match fails.

### 24.4.8 START/STOP CONDITION INTERRUPT MASKING

The SCIE and PCIE bits of the SSPxCON3 register can enable the generation of an interrupt in Slave modes that do not typically support this function. Slave modes where interrupt on Start and Stop detect are already enabled, these bits will have no effect.

**FIGURE 24-12: $I^2C$ START AND STOP CONDITIONS**

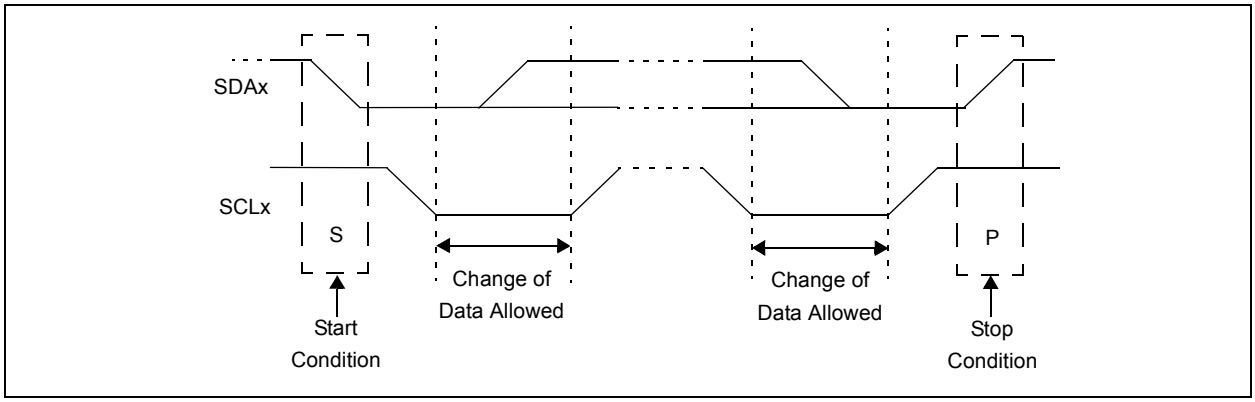
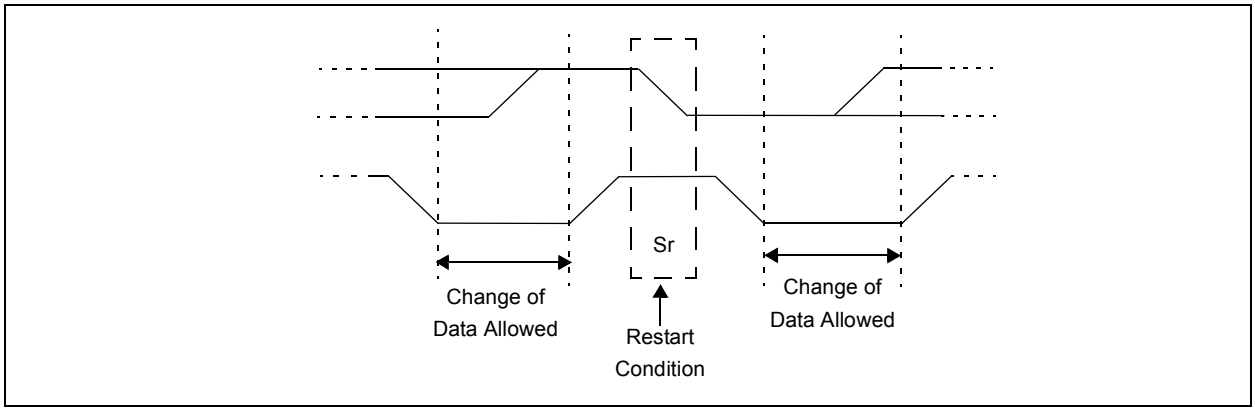



**FIGURE 24-13: $I^2C$ RESTART CONDITION**



#### 24.4.9 ACKNOWLEDGE SEQUENCE

The 9th SCLx pulse for any transferred byte in $I^2C$ is dedicated as an Acknowledge. It allows receiving devices to respond back to the transmitter by pulling the SDAx line low. The transmitter must release control of the line during this time to shift in the response. The Acknowledge ($\overline{ACK}$) is an active-low signal, pulling the SDAx line low indicated to the transmitter that the device has received the transmitted data and is ready to receive more.

The result of an $\overline{ACK}$ is placed in the ACKSTAT bit of the SSPxCON2 register.

Slave software, when the AHEN and DHEN bits are set, allows the user to set the $\overline{ACK}$ value sent back to the transmitter. The ACKDT bit of the SSPxCON2 register is set/cleared to determine the response.

Slave hardware will generate an $\overline{ACK}$ response if the AHEN and DHEN bits of the SSPxCON3 register are clear.

An $\overline{ACK}$ will not be sent by the slave when an overflow condition is detected. An overflow condition is defined by either the SSPxSTAT register bit BF being set, or by the SSPxCON1 register bit SSPOV being set.

When the module is addressed, after the eighth falling edge of SCLx on the bus, the ACKTIM bit of the SSPxCON3 register is set. The ACKTIM bit indicates the acknowledge time of the active bus. The ACKTIM Status bit is only active when the AHEN bit or DHEN bit is enabled.

### 24.5 $I^2C$ SLAVE MODE OPERATION

The MSSPx Slave mode operates in one of four modes selected in the SSPM bits of SSPxCON1 register. The modes can be divided into 7-bit and 10-bit Addressing mode. 10-bit Addressing modes operate the same as 7-bit with some additional overhead for handling the larger addresses.

Modes with Start and Stop bit interrupts operated the same as the other modes with SSPxIF additionally getting set upon detection of a Start, Restart, or Stop condition.

#### 24.5.1 SLAVE MODE ADDRESSES

The SSPxADD register (Register 24-6) contains the Slave mode address. The first byte received after a Start or Restart condition is compared against the value stored in this register. If the byte matches, the value is loaded into the SSPxBUF register and an interrupt is generated. If the value does not match, the module goes idle and no indication is given to the software that anything happened.

The SSPx Mask register (Register 24-5) affects the address matching process. See **Section 24.5.9 "SSPx Mask Register"** for more information.

##### 24.5.1.1 $I^2C$ Slave 7-bit Addressing Mode

In 7-bit Addressing mode, the LSb of the received data byte is ignored when determining if there is an address match.

##### 24.5.1.2 $I^2C$ Slave 10-bit Addressing Mode

In 10-bit Addressing mode, the first received byte is compared to the binary value of '`1 1 1 1 0 A9 A8 0`'. A9 and A8 are the two MSb of the 10-bit address and stored in bits 2 and 1 of the SSPxADD register.

After the acknowledge of the high byte, the UA bit is set and SCLx is held low until the user updates SSPxADD with the low address. The low address byte is clocked in and all eight bits are compared to the low address value in SSPxADD. Even if there is not an address match, SSPxIF and UA are set, and SCLx is held low until SSPxADD is updated to receive a high byte again. When SSPxADD is updated, the UA bit is cleared. This ensures the module is ready to receive the high address byte on the next communication.

A high and low address match as a write request is required at the start of all 10-bit addressing communication. A transmission can be initiated by issuing a Restart once the slave is addressed, and clocking in the high address with the R/$\overline{W}$ bit set. The slave hardware will then acknowledge the read request and prepare to clock out data. This is only valid for a slave after it has received a complete high and low address byte match.


© 2010-2016 Microchip Technology Inc.

### 24.5.2 SLAVE RECEPTION

When the R/$\overline{W}$ bit of a matching received address byte is clear, the R/$\overline{W}$ bit of the SSPxSTAT register is cleared. The received address is loaded into the SSPxBUF register and acknowledged.

When the overflow condition exists for a received address, then NACK is given. An overflow condition is defined by either the SSPxSTAT register bit BF being set, or by the SSPxCON1 register bit SSPOV being set. The BOEN bit of the SSPxCON3 register modifies this operation. For more information see Register 24-4.

An MSSPx interrupt is generated for each transferred data byte. Flag bit, SSPxIF, must be cleared by software.

When the SEN bit of the SSPxCON2 register is set, SCLx will be held low (clock stretch) following each received byte. The clock must be released by setting the CKP bit of the SSPxCON1 register, except in 10-bit mode, when the SSPxSTAT register bit UA is set. See **Section 24.5.6.2 "10-bit Addressing Mode"** for more details.

#### 24.5.2.1 7-bit Addressing Reception

This section describes a standard sequence of events for the MSSPx module configured as an $I^2C$ Slave in 7-bit Addressing mode. Figure 24-13 and Figure 24-14 are used as visual references for this description.

This is a step by step process of what typically must be done to accomplish $I^2C$ communication.

1. Start bit detected.
2. S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
3. Matching address with R/$\overline{W}$ bit clear is received.
4. The slave pulls SDAx low sending an $\overline{ACK}$ to the master, and sets the SSPxIF bit.
5. Software clears the SSPxIF bit.
6. Software reads received address from SSPxBUF, clearing the BF flag.
7. If SEN = 1; Slave software sets CKP bit to release the SCLx line.
8. The master clocks out a data byte.
9. Slave drives SDAx low, sending an $\overline{ACK}$ to the master, and sets SSPxIF bit.
10. Software clears SSPxIF.
11. Software reads the received byte from SSPxBUF clearing BF.
12. Steps 8-12 are repeated for all received bytes from the Master.
13. Master sends Stop condition, setting P bit of SSPxSTAT, and the bus goes Idle.

#### 24.5.2.2 7-bit Reception with AHEN and DHEN

Slave device reception with AHEN and DHEN set operates the same as without these options with extra interrupts and clock stretching added after the eighth falling edge of SCLx. These additional interrupts allow the slave software to decide whether it wants to $\overline{ACK}$ the receive address or data byte, rather than the hardware. This functionality adds support for the PMBus™ that was not present on previous versions of this module.

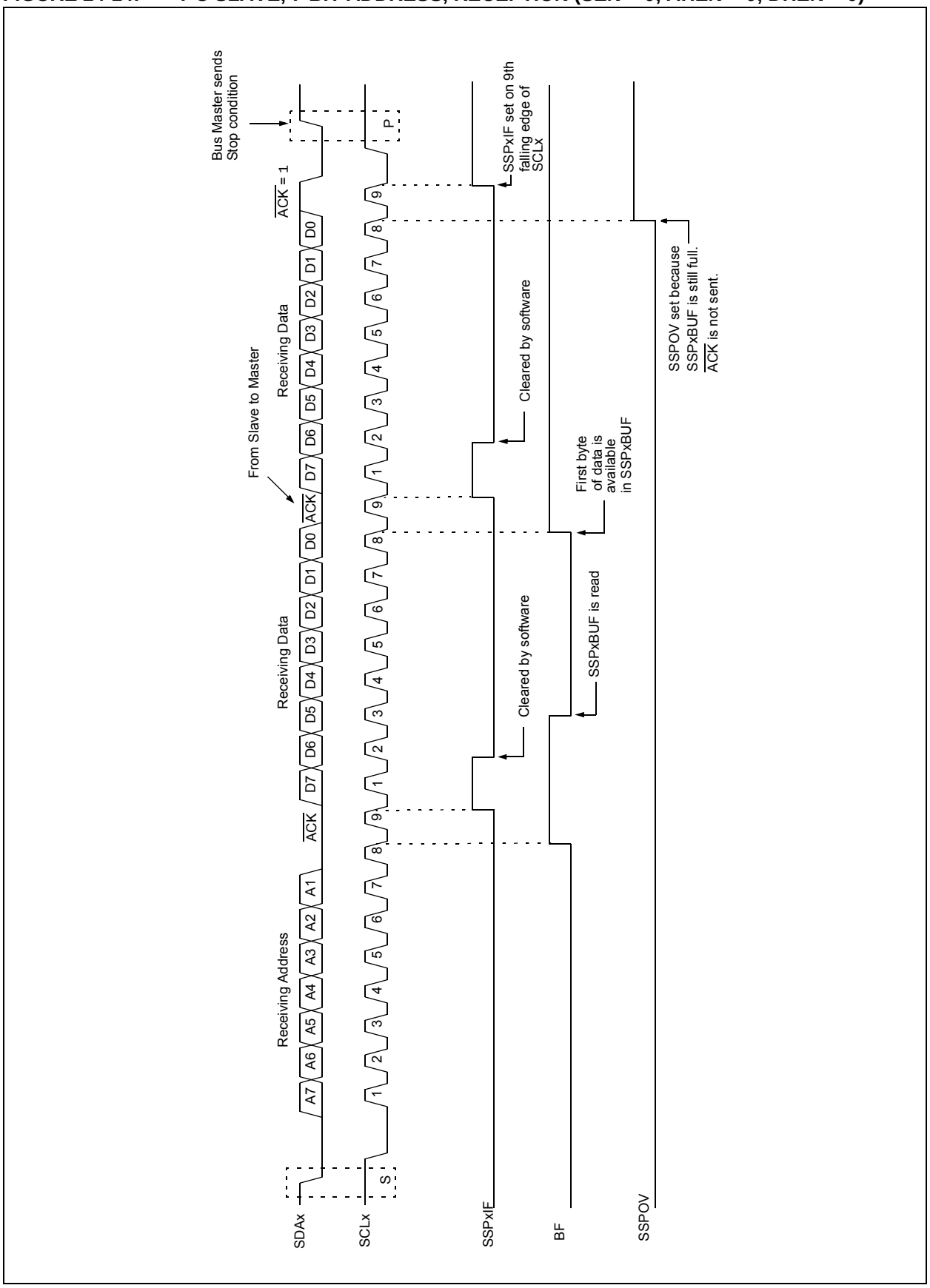
This list describes the steps that need to be taken by the slave software to use these options for $I^2C$ communication. Figure 24-15 displays a module using both address and data holding. Figure 24-16 includes the operation with the SEN bit of the SSPxCON2 register set.

1. The S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
2. Matching address with R/$\overline{W}$ bit clear is clocked in. SSPxIF is set and CKP cleared after the eighth falling edge of SCLx.
3. Slave clears the SSPxIF.
4. Slave can look at the ACKTIM bit of the SSPxCON3 register to determine if the SSPxIF was after or before the $\overline{ACK}$.
5. Slave reads the address value from SSPxBUF, clearing the BF flag.
6. Slave sets $\overline{ACK}$ value clocked out to the master by setting ACKDT.
7. Slave releases the clock by setting CKP.
8. SSPxIF is set after an $\overline{ACK}$, not after a NACK.
9. If SEN = 1, the slave hardware will stretch the clock after the $\overline{ACK}$.
10. Slave clears SSPxIF.

**Note:** SSPxIF is still set after the ninth falling edge of SCLx even if there is no clock stretching and BF has been cleared. SSPxIF is not set only when a NACK was sent to the MASTER.

11. SSPxIF is set and CKP cleared after eighth falling edge of SCLx for a received data byte.
12. Slave looks at the ACKTIM bit of SSPxCON3 to determine the source of the interrupt.
13. Slave reads the received data from SSPxBUF, clearing BF.
14. Steps 7-14 are the same for each received data byte.
15. Communication is ended by either the slave sending an $\overline{ACK}$ = 1, or the master sending a Stop condition. If a Stop is sent and Interrupt on Stop Detect is disabled, the slave must poll the P bit of the SSTSTAT register to know that the Stop condition was received.

**FIGURE 24-14: I²C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 0, AHEN = 0, DHEN = 0)**





© 2010-2016 Microchip Technology Inc.

**FIGURE 24-15: I²C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 1, AHEN = 0, DHEN = 0)**

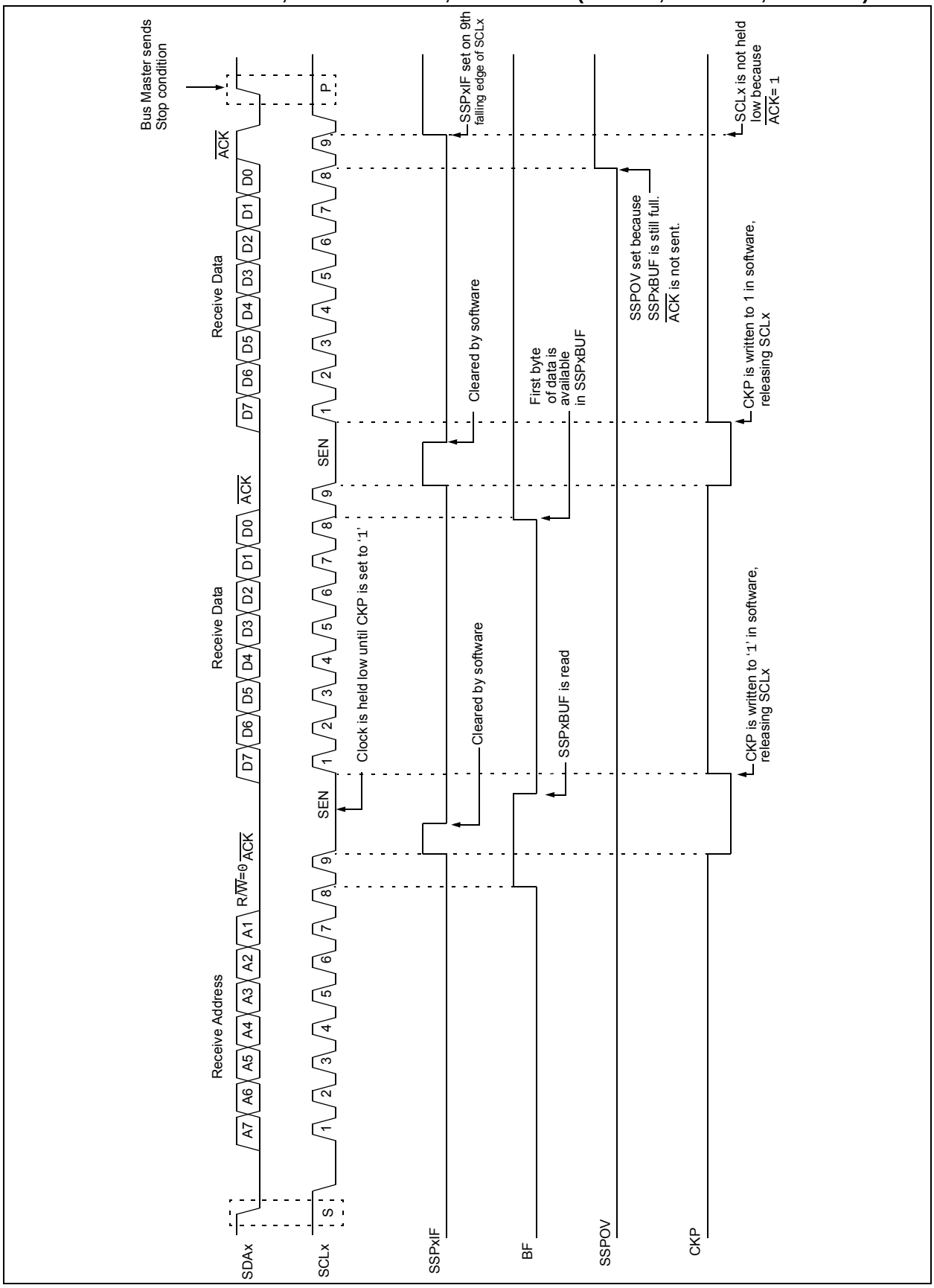
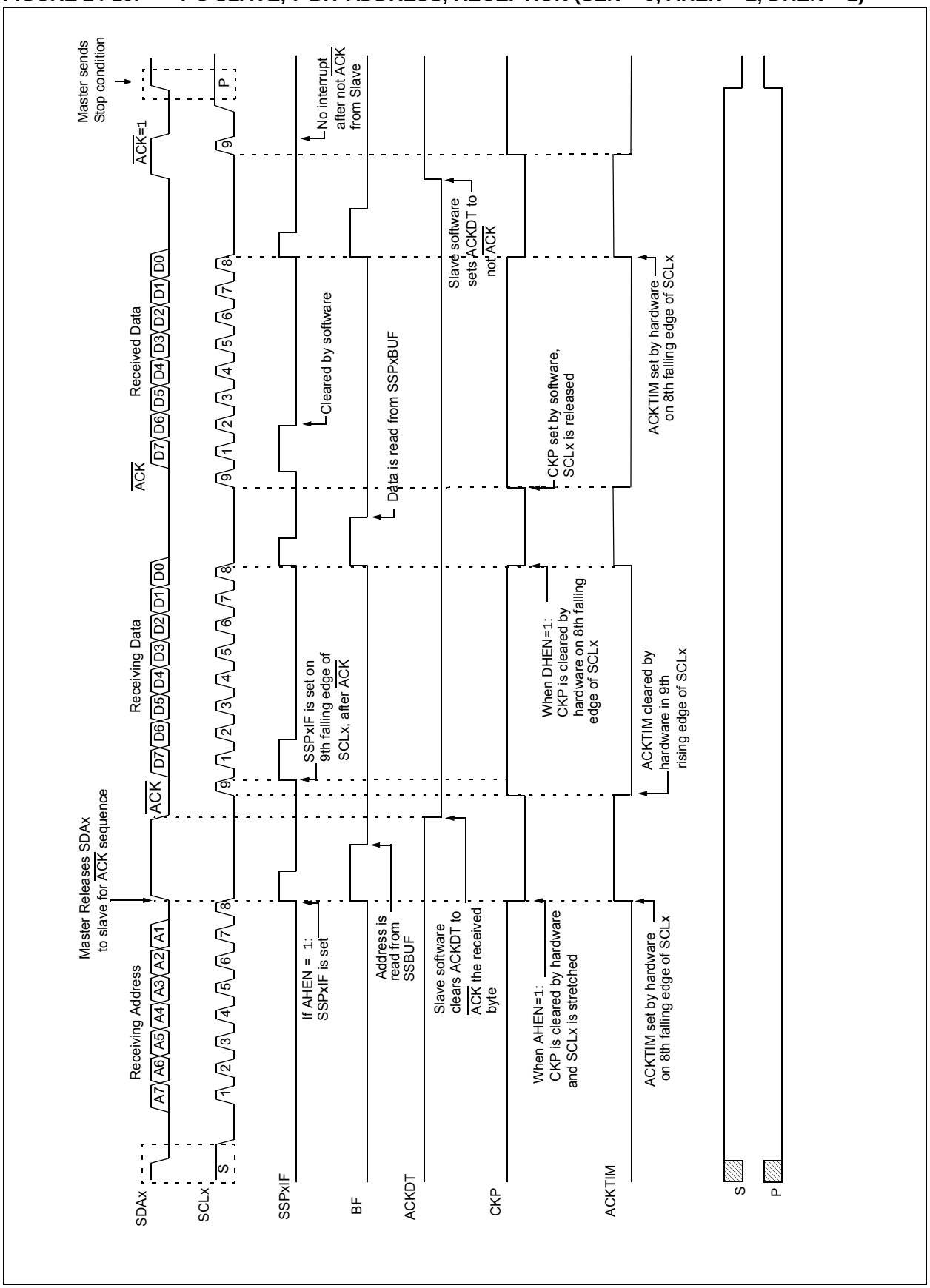


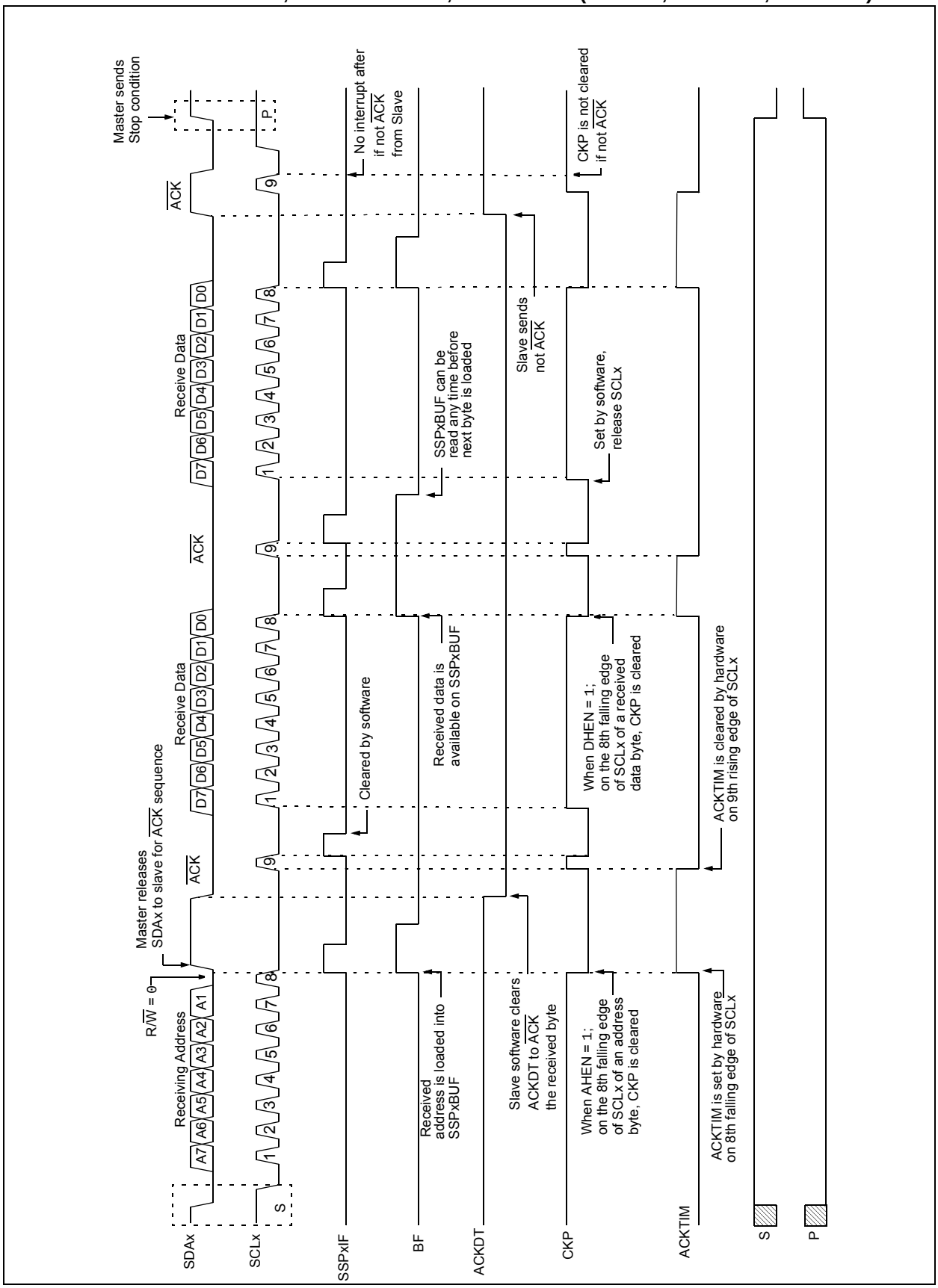
**FIGURE 24-16: I²C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 0, AHEN = 1, DHEN = 1)**





© 2010-2016 Microchip Technology Inc.

**FIGURE 24-17: I²C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 1, AHEN = 1, DHEN = 1)**



### 24.5.3 SLAVE TRANSMISSION

When the R/$\overline{W}$ bit of the incoming address byte is set and an address match occurs, the R/$\overline{W}$ bit of the SSPxSTAT register is set. The received address is loaded into the SSPxBUF register, and an $\overline{ACK}$ pulse is sent by the slave on the ninth bit.

Following the $\overline{ACK}$, slave hardware clears the CKP bit and the SCLx pin is held low (see **Section 24.5.6 "Clock Stretching"** for more details). By stretching the clock, the master will be unable to assert another clock pulse until the slave is done, preparing the transmit data.

The transmit data must be loaded into the SSPxBUF register which also loads the SSPxSR register. Then, the SCLx pin should be released by setting the CKP bit of the SSPxCON1 register. The eight data bits are shifted out on the falling edge of the SCLx input. This ensures that the SDAx signal is valid during the SCLx high time.

The $\overline{ACK}$ pulse from the master-receiver is latched on the rising edge of the ninth SCLx input pulse. This $\overline{ACK}$ value is copied to the ACKSTAT bit of the SSPxCON2 register. If ACKSTAT is set (not $\overline{ACK}$), then the data transfer is complete. In this case, when the not $\overline{ACK}$ is latched by the slave, the slave goes Idle and waits for another occurrence of the Start bit. If the SDAx line was low ($\overline{ACK}$), the next transmit data must be loaded into the SSPxBUF register. Again, the SCLx pin must be released by setting bit CKP.

An MSSPx interrupt is generated for each data transfer byte. The SSPxIF bit must be cleared by software and the SSPxSTAT register is used to determine the status of the byte. The SSPxIF bit is set on the falling edge of the ninth clock pulse.

#### 24.5.3.1 Slave Mode Bus Collision

A slave receives a Read request and begins shifting data out on the SDAx line. If a bus collision is detected and the SBCDE bit of the SSPxCON3 register is set, the BCLxIF bit of the PIRx register is set. Once a bus collision is detected, the slave goes Idle and waits to be addressed again. User software can use the BCLxIF bit to handle a slave bus collision.

#### 24.5.3.2 7-bit Transmission

A master device can transmit a read request to a slave, and then clock data out of the slave. The list below outlines what software for a slave will need to do to accomplish a standard transmission. Figure 24-17 can be used as a reference to this list.

1. Master sends a Start condition on SDAx and SCLx.
2. S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
3. Matching address with R/$\overline{W}$ bit set is received by the Slave setting SSPxIF bit.
4. Slave hardware generates an $\overline{ACK}$ and sets SSPxIF.
5. SSPxIF bit is cleared by the user.
6. Software reads the received address from SSPxBUF, clearing BF.
7. R/$\overline{W}$ is set so CKP is automatically cleared after the $\overline{ACK}$.
8. The slave software loads the transmit data into SSPxBUF.
9. CKP bit is set releasing SCLx, allowing the master to clock the data out of the slave.
10. SSPxIF is set after the $\overline{ACK}$ response from the master is loaded into the ACKSTAT register.
11. SSPxIF bit is cleared.
12. The slave software checks the ACKSTAT bit to see if the master wants to clock out more data.

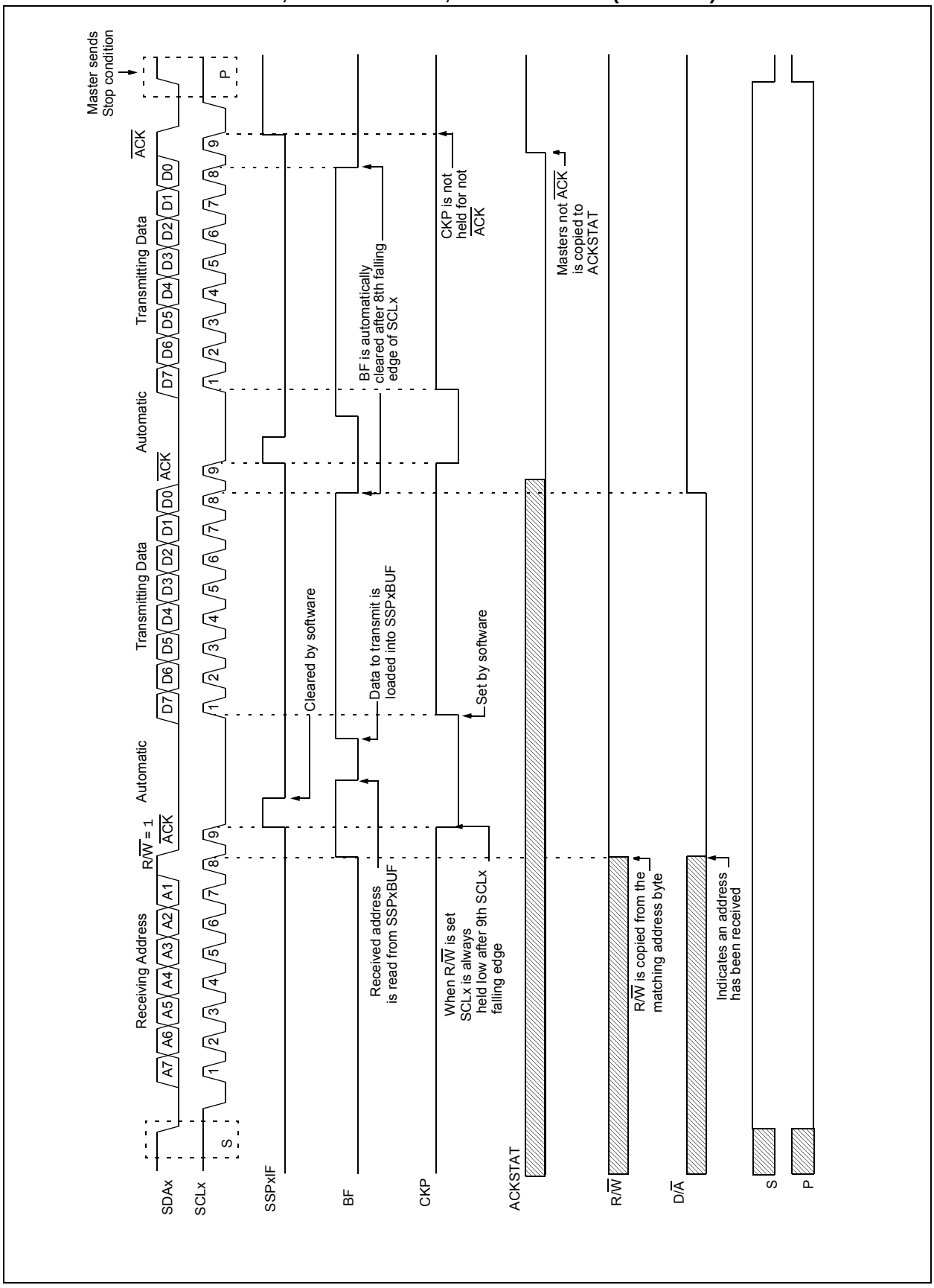
**Note 1:** If the master $\overline{ACK}$s, the clock will be stretched.

**2:** ACKSTAT is the only bit updated on the rising edge of SCLx (9th) rather than on the falling.

13. Steps 9-13 are repeated for each transmitted byte.
14. If the master sends a not $\overline{ACK}$; the clock is not held, but SSPxIF is still set.
15. The master sends a Restart condition or a Stop.
16. The slave is no longer addressed.


© 2010-2016 Microchip Technology Inc.

**FIGURE 24-18: I²C SLAVE, 7-BIT ADDRESS, TRANSMISSION (AHEN = 0)**



#### 24.5.3.3 7-bit Transmission with Address Hold Enabled

Setting the AHEN bit of the SSPxCON3 register enables additional clock stretching and interrupt generation after the eighth falling edge of a received matching address. Once a matching address has been clocked in, CKP is cleared and the SSPxIF interrupt is set.

Figure 24-18 displays a standard waveform of a 7-bit Address Slave Transmission with AHEN enabled.

1. Bus starts Idle.
2. Master sends Start condition; the S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
3. Master sends matching address with R/$\overline{W}$ bit set. After the eighth falling edge of the SCLx line the CKP bit is cleared and SSPxIF interrupt is generated.
4. Slave software clears SSPxIF.
5. Slave software reads the ACKTIM bit of the SSPxCON3 register, and R/$\overline{W}$ and D/$\overline{A}$ of the SSPxSTAT register to determine the source of the interrupt.
6. Slave reads the address value from the SSPxBUF register clearing the BF bit.
7. Slave software decides from this information if it wishes to $\overline{ACK}$ or not $\overline{ACK}$ and sets the ACKDT bit of the SSPxCON2 register accordingly.
8. Slave sets the CKP bit releasing SCLx.
9. Master clocks in the $\overline{ACK}$ value from the slave.
10. Slave hardware automatically clears the CKP bit and sets SSPxIF after the $\overline{ACK}$ if the R/$\overline{W}$ bit is set.
11. Slave software clears SSPxIF.
12. Slave loads value to transmit to the master into SSPxBUF setting the BF bit.

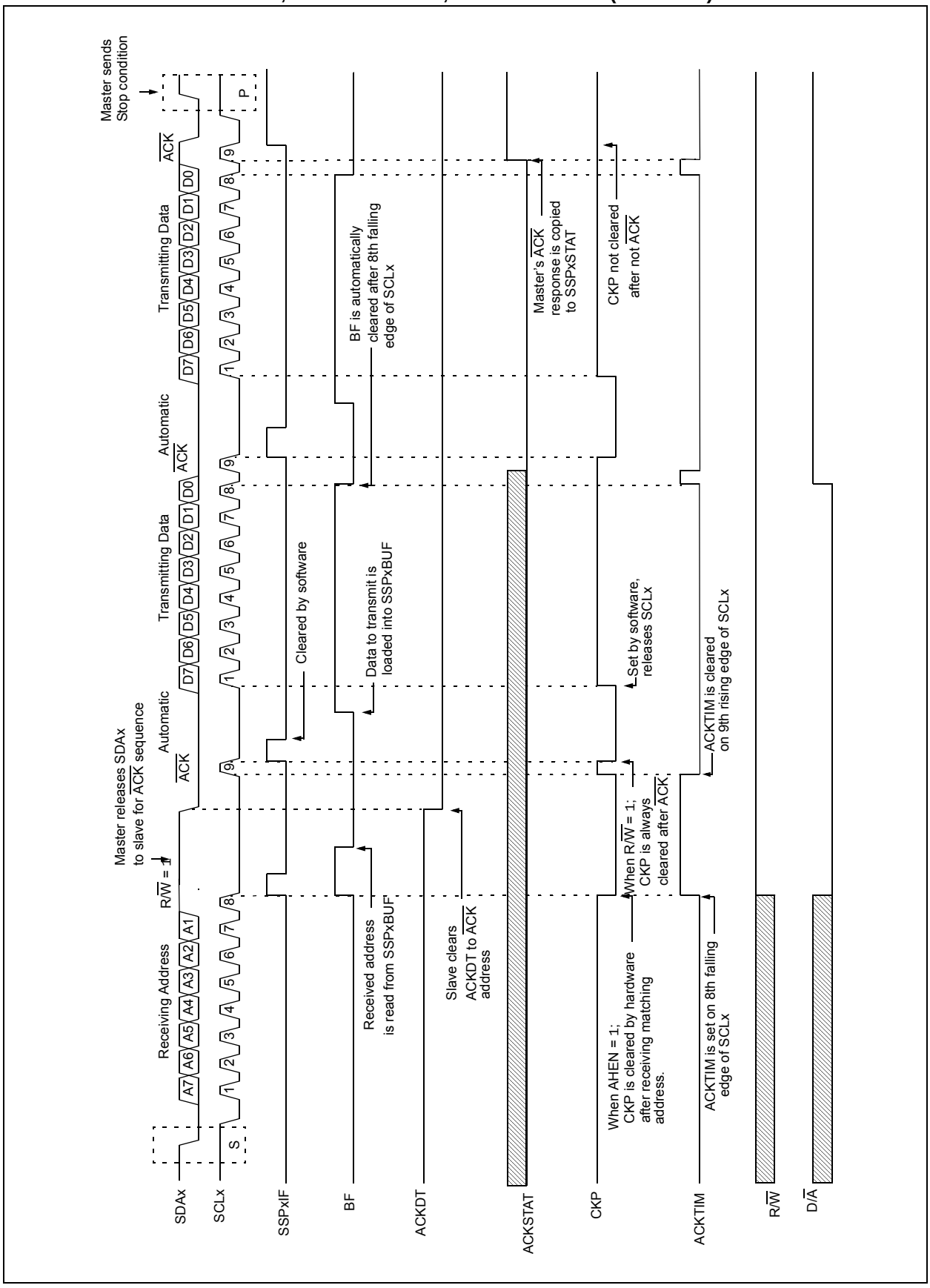
**Note:** SSPxBUF cannot be loaded until after the $\overline{ACK}$.

13. Slave sets the CKP bit, releasing the clock.
14. Master clocks out the data from the slave and sends an $\overline{ACK}$ value on the ninth SCLx pulse.
15. Slave hardware copies the $\overline{ACK}$ value into the ACKSTAT bit of the SSPxCON2 register.
16. Steps 10-15 are repeated for each byte transmitted to the master from the slave.
17. If the master sends a not $\overline{ACK}$, the slave releases the bus allowing the master to send a Stop and end the communication.

**Note:** Master must send a not $\overline{ACK}$ on the last byte to ensure that the slave releases the SCLx line to receive a Stop.


© 2010-2016 Microchip Technology Inc.

**FIGURE 24-19: I²C SLAVE, 7-BIT ADDRESS, TRANSMISSION (AHEN = 1)**



### 24.5.4 SLAVE MODE 10-BIT ADDRESS RECEPTION

This section describes a standard sequence of events for the MSSPx module configured as an $I^2C$ Slave in 10-bit Addressing mode.

Figure 24-19 is used as a visual reference for this description.

This is a step by step process of what must be done by slave software to accomplish $I^2C$ communication.

1. Bus starts Idle.
2. Master sends Start condition; S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
3. Master sends matching high address with R/$\overline{W}$ bit clear; UA bit of the SSPxSTAT register is set.
4. Slave sends $\overline{ACK}$ and SSPxIF is set.
5. Software clears the SSPxIF bit.
6. Software reads received address from SSPxBUF clearing the BF flag.
7. Slave loads low address into SSPxADD, releasing SCLx.
8. Master sends matching low address byte to the Slave; UA bit is set.

**Note:** Updates to the SSPxADD register are not allowed until after the $\overline{ACK}$ sequence.

9. Slave sends $\overline{ACK}$ and SSPxIF is set.

**Note:** If the low address does not match, SSPxIF and UA are still set so that the slave software can set SSPxADD back to the high address. BF is not set because there is no match. CKP is unaffected.

10. Slave clears SSPxIF.
11. Slave reads the received matching address from SSPxBUF clearing BF.
12. Slave loads high address into SSPxADD.
13. Master clocks a data byte to the slave and clocks out the slaves $\overline{ACK}$ on the ninth SCLx pulse; SSPxIF is set.
14. If SEN bit of SSPxCON2 is set, CKP is cleared by hardware and the clock is stretched.
15. Slave clears SSPxIF.
16. Slave reads the received byte from SSPxBUF clearing BF.
17. If SEN is set, the slave sets CKP to release the SCLx.
18. Steps 13-17 repeat for each received byte.
19. Master sends Stop to end the transmission.

### 24.5.5 10-BIT ADDRESSING WITH ADDRESS OR DATA HOLD

Reception using 10-bit addressing with AHEN or DHEN set is the same as with 7-bit modes. The only difference is the need to update the SSPxADD register using the UA bit. All functionality, specifically when the CKP bit is cleared and SCLx line is held low are the same. Figure 24-20 can be used as a reference of a slave in 10-bit addressing with AHEN set.

Figure 24-21 shows a standard waveform for a slave transmitter in 10-bit Addressing mode.


© 2010-2016 Microchip Technology Inc.

**FIGURE 24-20: I²C SLAVE, 10-BIT ADDRESS, RECEPTION (SEN = 1, AHEN = 0, DHEN = 0)**

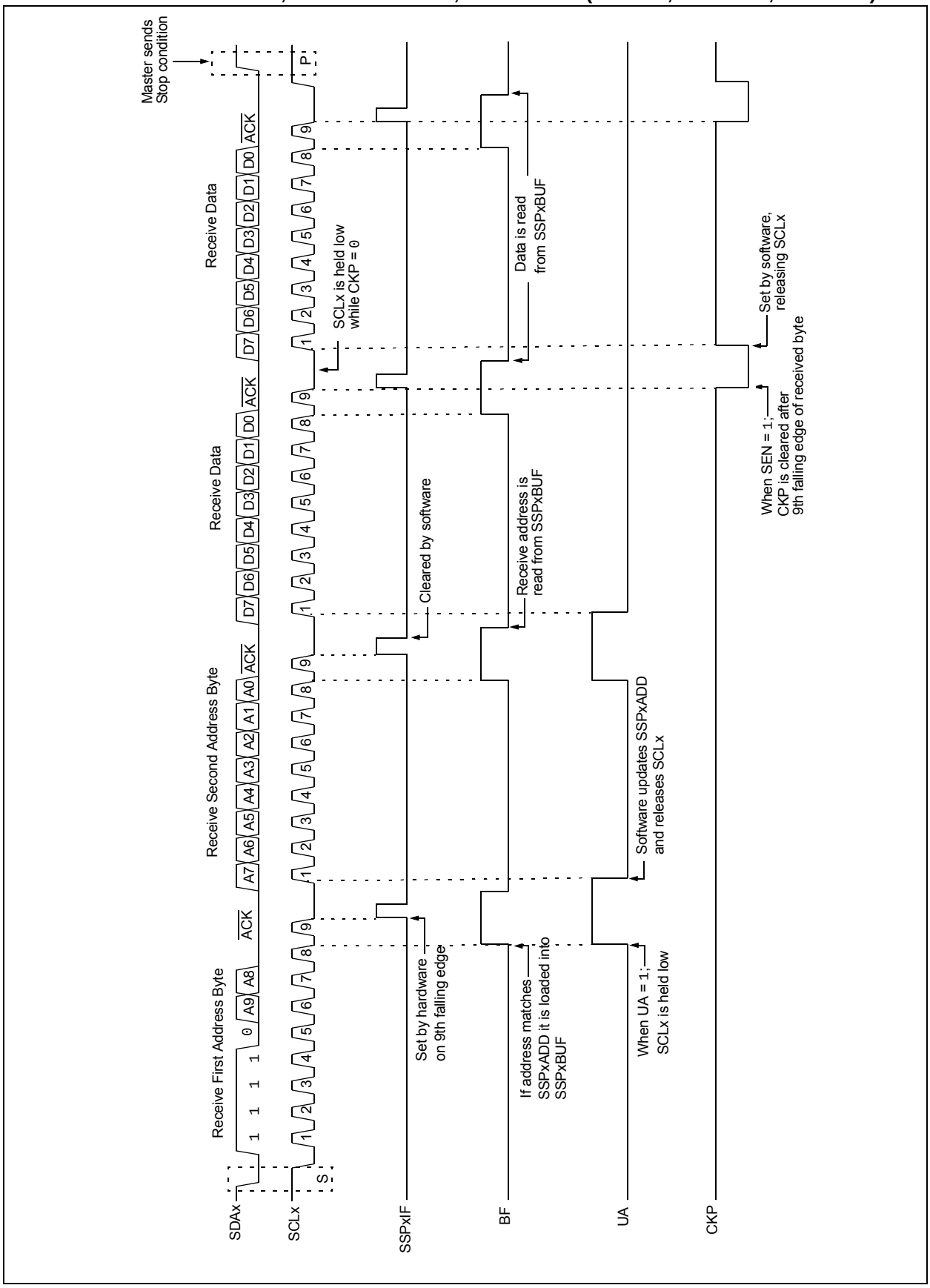
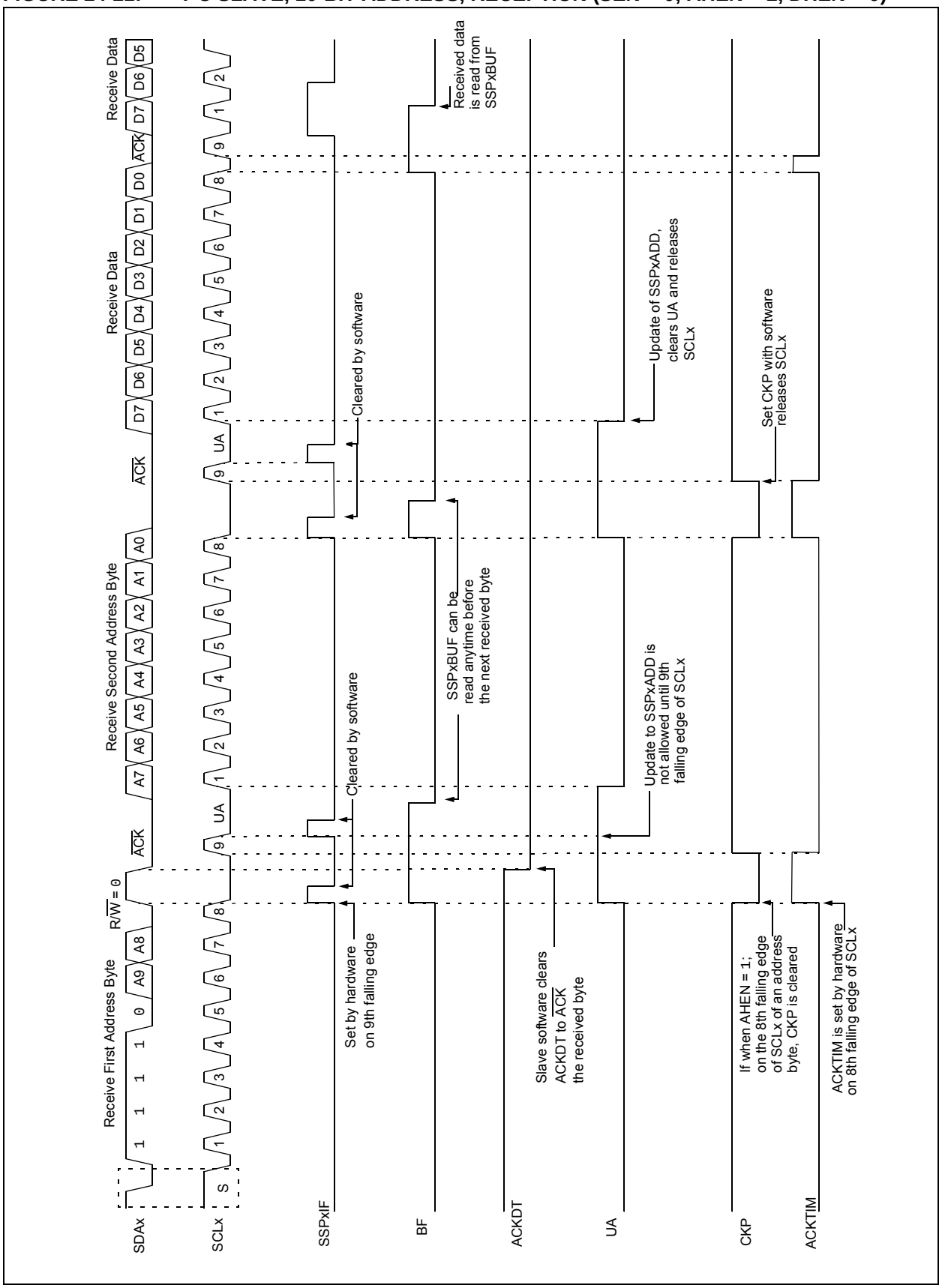


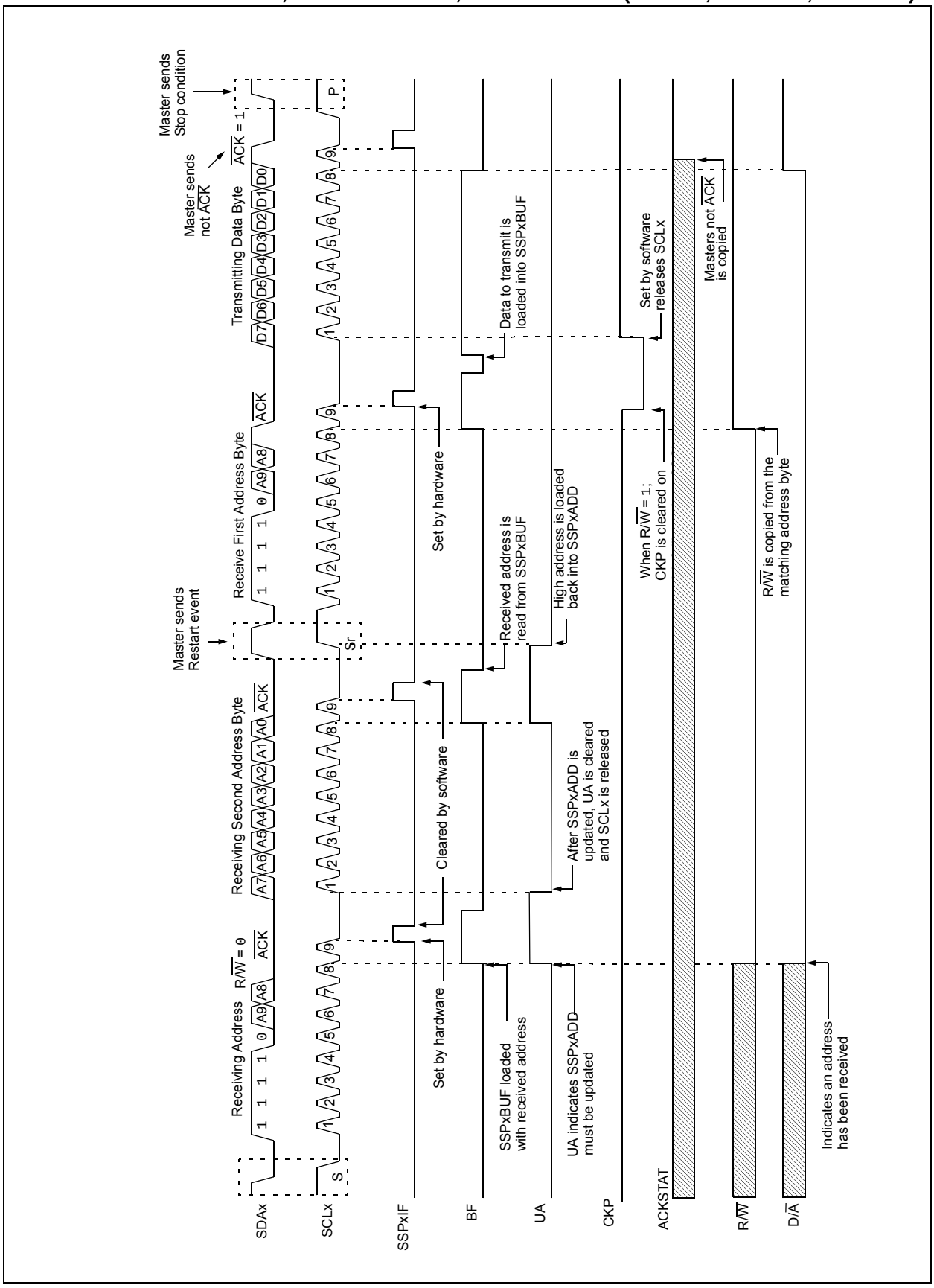
**FIGURE 24-21: I²C SLAVE, 10-BIT ADDRESS, RECEPTION (SEN = 0, AHEN = 1, DHEN = 0)**





© 2010-2016 Microchip Technology Inc.

**FIGURE 24-22: I²C SLAVE, 10-BIT ADDRESS, TRANSMISSION (SEN = 0, AHEN = 0, DHEN = 0)**



### 24.5.6 CLOCK STRETCHING

Clock stretching occurs when a device on the bus holds the SCLx line low effectively, pausing communication. The slave may stretch the clock to allow more time to handle data or prepare a response for the master device. A master device is not concerned with stretching as anytime it is active on the bus and not transferring data it is stretching. Any stretching done by a slave is invisible to the master software and by the hardware that generates SCLx.

The CKP bit of the SSPxCON1 register is used to control stretching in software. Any time the CKP bit is cleared, the module will wait for the SCLx line to go low and then hold it. Setting CKP will release SCLx and allow more communication.

#### 24.5.6.1 Normal Clock Stretching

Following an $\overline{ACK}$, if the R/$\overline{W}$ bit of SSPxSTAT is set, the slave hardware will clear CKP. This allows the slave time to update SSPxBUF with data to transfer to the master. If the SEN bit of SSPxCON2 is set, the slave hardware will always stretch the clock after the $\overline{ACK}$ sequence. Once the slave is ready, CKP is set by software and communication resumes.

**Note 1:** The BF bit has no effect if the clock will be stretched or not. The previous version of the module did not stretch the clock if SSPxBUF was read before the 9th falling edge of SCLx.

**2:** The previous versions of the module did not stretch the clock for a transmission if SSPxBUF was loaded before the 9th falling edge of SCLx. It is now always cleared for read requests.

#### 24.5.6.2 10-bit Addressing Mode

In 10-bit Addressing mode, when the UA bit is set, the clock is always stretched. This is the only time the SCLx is stretched without CKP being cleared. SCLx is released immediately after a write to SSPxADD.

**Note:** Previous versions of the module did not stretch the clock if the second address byte did not match.

#### 24.5.6.3 Byte NACKing

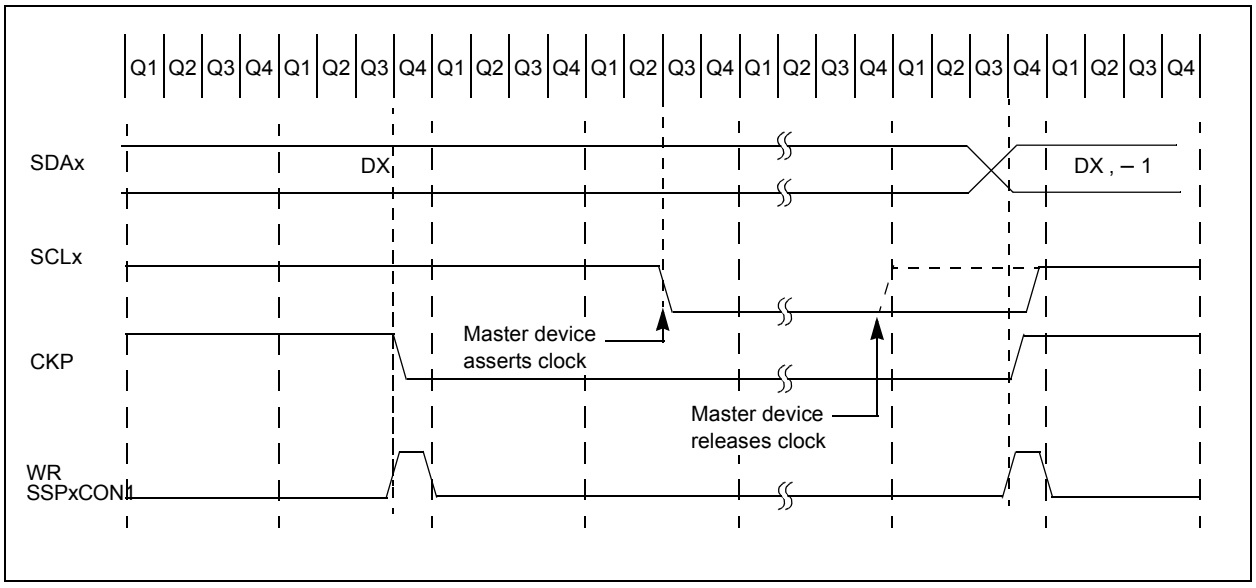
When AHEN bit of SSPxCON3 is set, CKP is cleared by hardware after the eighth falling edge of SCLx for a received matching address byte. When the DHEN bit of SSPxCON3 is set, CKP is cleared after the eighth falling edge of SCLx for received data.

Stretching after the eighth falling edge of SCLx allows the slave to look at the received address or data and decide if it wants to ACK the received data.

### 24.5.7 CLOCK SYNCHRONIZATION AND THE CKP BIT

Any time the CKP bit is cleared, the module will wait for the SCLx line to go low and then hold it. However, clearing the CKP bit will not assert the SCLx output low until the SCLx output is already sampled low. Therefore, the CKP bit will not assert the SCLx line until an external $I^2C$ master device has already asserted the SCLx line. The SCLx output will remain low until the CKP bit is set and all other devices on the $I^2C$ bus have released SCLx. This ensures that a write to the CKP bit will not violate the minimum high time requirement for SCLx (see Figure 24-23).

**FIGURE 24-23: CLOCK SYNCHRONIZATION TIMING**





© 2010-2016 Microchip Technology Inc.

### 24.5.8 GENERAL CALL ADDRESS SUPPORT

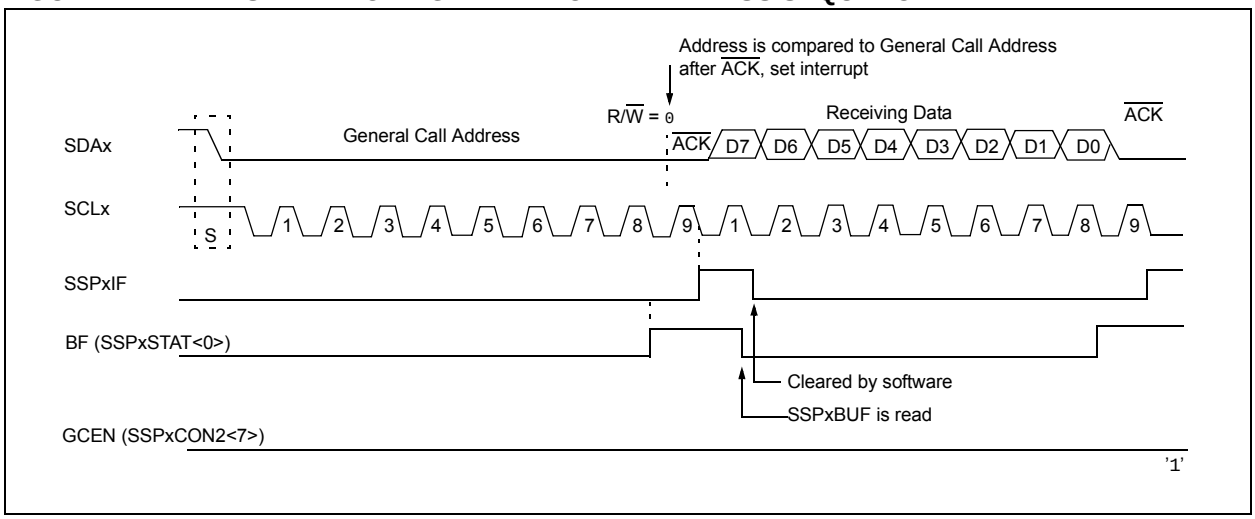
The addressing procedure for the I²C bus is such that the first byte after the Start condition usually determines which device will be the slave addressed by the master device. The exception is the general call address which can address all devices. When this address is used, all devices should, in theory, respond with an acknowledge.

The general call address is a reserved address in the I²C protocol, defined as address 0x00. When the GCEN bit of the SSPxCON2 register is set, the slave module will automatically ACK the reception of this address, regardless of the value stored in SSPxADD. After the slave clocks in an address of all zeros with the R/W bit clear, an interrupt is generated and slave software can read SSPxBUF and respond. Figure 24-23 shows a general call reception sequence.

In 10-bit Address mode, the UA bit will not be set on the reception of the general call address. The slave will prepare to receive the second byte as data, just as it would in 7-bit mode.

If the AHEN bit of the SSPxCON3 register is set, just as with any other address reception, the slave hardware will stretch the clock after the eighth falling edge of SCLx. The slave must then set its ACKDT value and release the clock with communication, progressing as it would normally.

**FIGURE 24-24: SLAVE MODE GENERAL CALL ADDRESS SEQUENCE**




### 24.5.9 SSPx MASK REGISTER

An SSPx Mask (SSPxMSK) register (Register 24-5) is available in I²C Slave mode as a mask for the value held in the SSPxSR register during an address comparison operation. A zero ('0') bit in the SSPxMSK register has the effect of making the corresponding bit of the received address a "don't care".

This register is reset to all '1's upon any Reset condition and, therefore, has no effect on standard SSPx operation until written with a mask value.

The SSPx Mask register is active during:

- 7-bit Address mode: address compare of A<7:1>.
- 10-bit Address mode: address compare of A<7:0>, only. The SSPx mask has no effect during the reception of the first (high) byte of the address.

## 24.6 I²C MASTER MODE

Master mode is enabled by setting and clearing the appropriate SSPM bits in the SSPxCON1 register and by setting the SSPEN bit. In Master mode, the SDAx and SCKx pins must be configured as inputs. The MSSP peripheral hardware will override the output driver TRIS controls when necessary to drive the pins low.

Master mode of operation is supported by interrupt generation on the detection of the Start and Stop conditions. The Stop (P) and Start (S) bits are cleared from a Reset or when the MSSPx module is disabled. Control of the I²C bus may be taken when the P bit is set, or the bus is Idle.

In Firmware Controlled Master mode, user code conducts all I²C bus operations based on Start and Stop bit condition detection. Start and Stop condition detection is the only active circuitry in this mode. All other communication is done by the user software directly manipulating the SDAx and SCLx lines.

The following events will cause the SSPx Interrupt Flag bit, SSPxIF, to be set (SSPx interrupt, if enabled):

- Start condition detected
- Stop condition detected
- Data transfer byte transmitted/received
- Acknowledge transmitted/received
- Repeated Start generated

**Note 1:** The MSSPx module, when configured in I²C Master mode, does not allow queuing of events. For instance, the user is not allowed to initiate a Start condition and immediately write the SSPxBUF register to initiate transmission before the Start condition is complete. In this case, the SSPxBUF will not be written to and the WCOL bit will be set, indicating that a write to the SSPxBUF did not occur.

**2:** Master mode suspends Start/Stop detection when sending the Start/Stop condition by means of the SEN/PEN control bits. The SSPxIF bit is set at the end of the Start/Stop generation when hardware clears the control bit.

### 24.6.1 I²C MASTER MODE OPERATION

The master device generates all of the serial clock pulses and the Start and Stop conditions. A transfer is ended with a Stop condition or with a Repeated Start condition. Since the Repeated Start condition is also the beginning of the next serial transfer, the I²C bus will not be released.

In Master Transmitter mode, serial data is output through SDAx, while SCLx outputs the serial clock. The first byte transmitted contains the slave address of the receiving device (seven bits) and the Read/Write (R/W) bit. In this case, the R/W bit will be logic '0'. Serial data is transmitted eight bits at a time. After each byte is transmitted, an Acknowledge bit is received. Start and Stop conditions are output to indicate the beginning and the end of a serial transfer.

In Master Receive mode, the first byte transmitted contains the slave address of the transmitting device (seven bits) and the R/W bit. In this case, the R/W bit will be logic '1'. Thus, the first byte transmitted is a 7-bit slave address followed by a '1' to indicate the receive bit. Serial data is received via SDAx, while SCLx outputs the serial clock. Serial data is received eight bits at a time. After each byte is received, an Acknowledge bit is transmitted. Start and Stop conditions indicate the beginning and end of transmission.

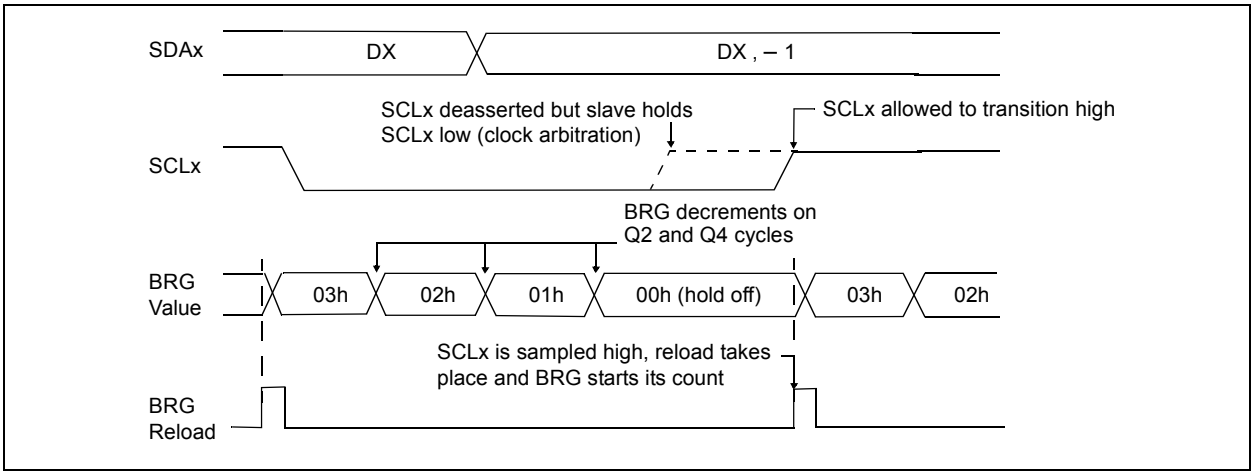
A Baud Rate Generator is used to set the clock frequency output on SCLx. See **Section 24.7 "Baud Rate Generator"** for more details.


© 2010-2016 Microchip Technology Inc.

### 24.6.2 CLOCK ARBITRATION

Clock arbitration occurs when the master, during any receive, transmit or Repeated Start/Stop condition, releases the SCLx pin (SCLx allowed to float high). When the SCLx pin is allowed to float high, the Baud Rate Generator (BRG) is suspended from counting until the SCLx pin is actually sampled high. When the SCLx pin is sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<7:0> and begins counting. This ensures that the SCLx high time will always be at least one BRG rollover count in the event that the clock is held low by an external device (Figure 24-25).

**FIGURE 24-25: BAUD RATE GENERATOR TIMING WITH CLOCK ARBITRATION**




### 24.6.3 WCOL STATUS FLAG

If the user writes the SSPxBUF when a Start, Restart, Stop, Receive or Transmit sequence is in progress, the WCOL bit is set and the contents of the buffer are unchanged (the write does not occur). Any time the WCOL bit is set, it indicates that an action on SSPxBUF was attempted while the module was not Idle.

| | |
|---|---|
| **Note:** | Because queuing of events is not allowed, writing to the lower five bits of SSPxCON2 is disabled until the Start condition is complete. |

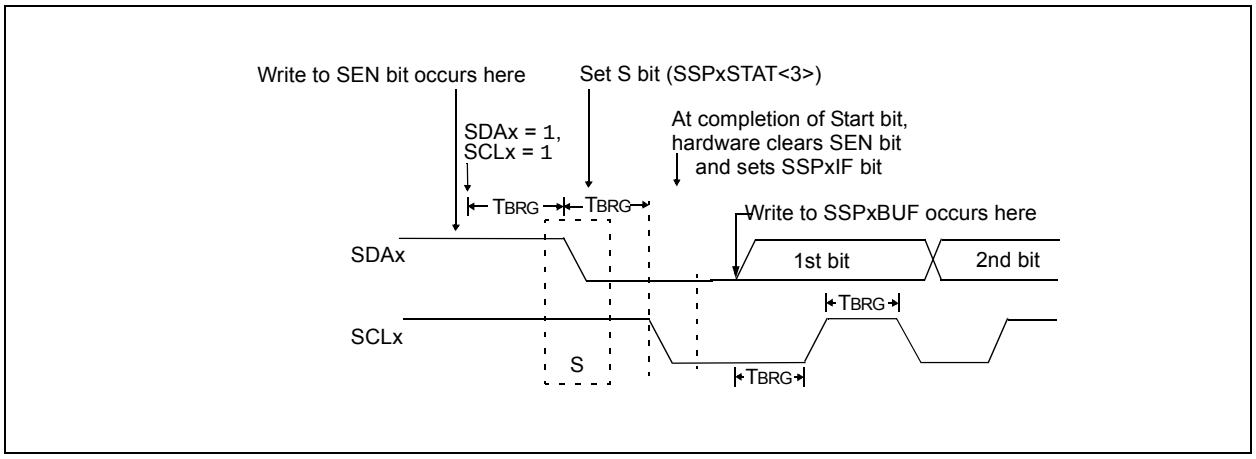
### 24.6.4 I²C MASTER MODE START CONDITION TIMING

To initiate a Start condition, the user sets the Start Enable bit, SEN bit of the SSPxCON2 register. If the SDAx and SCLx pins are sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<7:0> and starts its count. If SCLx and SDAx are both sampled high when the Baud Rate Generator times out (TBRG), the SDAx pin is driven low. The action of the SDAx being driven low while SCLx is high is the Start condition and causes the S bit of the SSPxSTAT1 register to be set. Following this, the Baud Rate Generator is reloaded with the contents of SSPxADD<7:0> and resumes its count. When the Baud Rate Generator times out (TBRG), the SEN bit of the SSPxCON2 register will be automatically cleared by hardware; the Baud Rate Generator is suspended, leaving the SDAx line held low and the Start condition is complete.

**Note 1:** If, at the beginning of the Start condition, the SDAx and SCLx pins are already sampled low, or if during the Start condition, the SCLx line is sampled low before the SDAx line is driven low, a bus collision occurs. The Bus Collision Interrupt Flag, BCLxIF, is set, the Start condition is aborted and the I²C module is reset into its Idle state.

**2:** The Philips I²C Specification states that a bus collision cannot occur on a Start.

**FIGURE 24-26: FIRST START BIT TIMING**





© 2010-2016 Microchip Technology Inc.

### 24.6.5 I²C MASTER MODE REPEATED START CONDITION TIMING

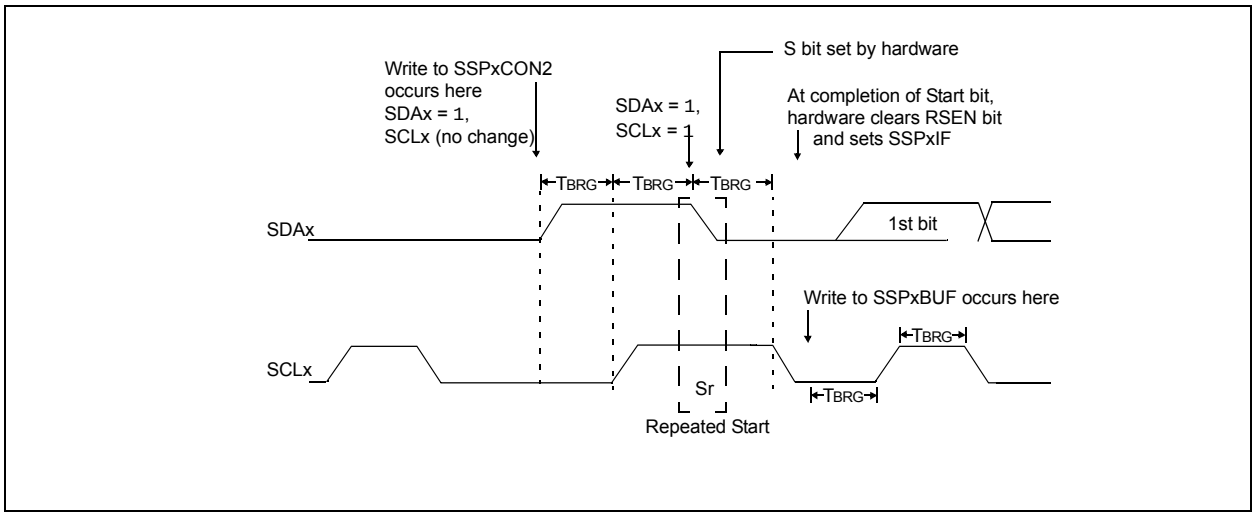
A Repeated Start condition occurs when the RSEN bit of the SSPxCON2 register is programmed high and the Master state machine is no longer active. When the RSEN bit is set, the SCLx pin is asserted low. When the SCLx pin is sampled low, the Baud Rate Generator is loaded and begins counting. The SDAx pin is released (brought high) for one Baud Rate Generator count (TBRG). When the Baud Rate Generator times out, if SDAx is sampled high, the SCLx pin will be deasserted (brought high). When SCLx is sampled high, the Baud Rate Generator is reloaded and begins counting. SDAx and SCLx must be sampled high for one TBRG. This action is then followed by assertion of the SDAx pin (SDAx = 0) for one TBRG while SCLx is high. SCLx is asserted low. Following this, the RSEN bit of the SSPxCON2 register will be automatically cleared and the Baud Rate Generator will not be reloaded, leaving the SDAx pin held low. As soon as a Start condition is detected on the SDAx and SCLx pins, the S bit of the SSPxSTAT register will be set. The SSPxIF bit will not be set until the Baud Rate Generator has timed out.

**Note 1:** If RSEN is programmed while any other event is in progress, it will not take effect.

**2:** A bus collision during the Repeated Start condition occurs if:

- SDAx is sampled low when SCLx goes from low-to-high.
- SCLx goes low before SDAx is asserted low. This may indicate that another master is attempting to transmit a data '1'.

**FIGURE 24-27: REPEAT START CONDITION WAVEFORM**



### 24.6.6 I²C MASTER MODE TRANSMISSION

Transmission of a data byte, a 7-bit address or the other half of a 10-bit address is accomplished by simply writing a value to the SSPxBUF register. This action will set the Buffer Full flag bit, BF, and allow the Baud Rate Generator to begin counting and start the next transmission. Each bit of address/data will be shifted out onto the SDAx pin after the falling edge of SCLx is asserted. SCLx is held low for one Baud Rate Generator rollover count (TBRG). Data should be valid before SCLx is released high. When the SCLx pin is released high, it is held that way for TBRG. The data on the SDAx pin must remain stable for that duration and some hold time after the next falling edge of SCLx. After the eighth bit is shifted out (the falling edge of the eighth clock), the BF flag is cleared and the master releases SDAx. This allows the slave device being addressed to respond with an $\overline{ACK}$ bit during the ninth bit time if an address match occurred, or if data was received properly. The status of $\overline{ACK}$ is written into the ACKSTAT bit on the rising edge of the ninth clock. If the master receives an Acknowledge, the Acknowledge Status bit, ACKSTAT, is cleared. If not, the bit is set. After the ninth clock, the SSPxIF bit is set and the master clock (Baud Rate Generator) is suspended until the next data byte is loaded into the SSPxBUF, leaving SCLx low and SDAx unchanged (Figure 24-27).

After the write to the SSPxBUF, each bit of the address will be shifted out on the falling edge of SCLx until all seven address bits and the R/$\overline{W}$ bit are completed. On the falling edge of the eighth clock, the master will release the SDAx pin, allowing the slave to respond with an Acknowledge. On the falling edge of the ninth clock, the master will sample the SDAx pin to see if the address was recognized by a slave. The status of the $\overline{ACK}$ bit is loaded into the ACKSTAT Status bit of the SSPxCON2 register. Following the falling edge of the ninth clock transmission of the address, the SSPxIF is set, the BF flag is cleared and the Baud Rate Generator is turned off until another write to the SSPxBUF takes place, holding SCLx low and allowing SDAx to float.

#### 24.6.6.1 BF Status Flag

In Transmit mode, the BF bit of the SSPxSTAT register is set when the CPU writes to SSPxBUF and is cleared when all eight bits are shifted out.

#### 24.6.6.2 WCOL Status Flag

If the user writes the SSPxBUF when a transmit is already in progress (i.e., SSPxSR is still shifting out a data byte), the WCOL bit is set and the contents of the buffer are unchanged (the write does not occur).

WCOL must be cleared by software before the next transmission.

#### 24.6.6.3 ACKSTAT Status Flag

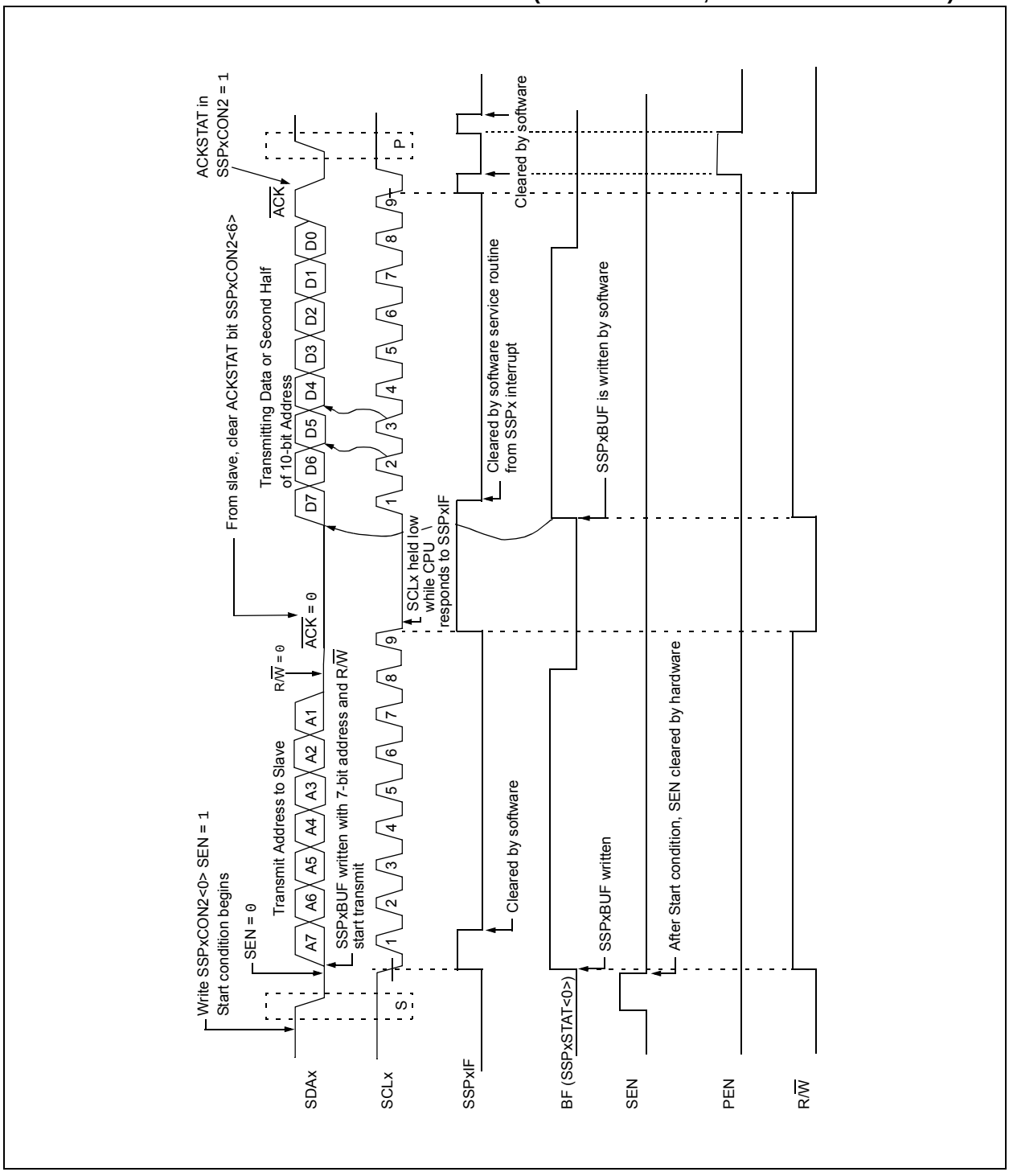
In Transmit mode, the ACKSTAT bit of the SSPxCON2 register is cleared when the slave has sent an Acknowledge ($\overline{ACK}$ = 0) and is set when the slave does not Acknowledge ($\overline{ACK}$ = 1). A slave sends an Acknowledge when it has recognized its address (including a general call), or when the slave has properly received its data.

#### 24.6.6.4 Typical transmit sequence:

1. The user generates a Start condition by setting the SEN bit of the SSPxCON2 register.
2. SSPxIF is set by hardware on completion of the Start.
3. SSPxIF is cleared by software.
4. The MSSPx module will wait the required start time before any other operation takes place.
5. The user loads the SSPxBUF with the slave address to transmit.
6. Address is shifted out the SDAx pin until all eight bits are transmitted. Transmission begins as soon as SSPxBUF is written to.
7. The MSSPx module shifts in the $\overline{ACK}$ bit from the slave device and writes its value into the ACKSTAT bit of the SSPxCON2 register.
8. The MSSPx module generates an interrupt at the end of the ninth clock cycle by setting the SSPxIF bit.
9. The user loads the SSPxBUF with eight bits of data.
10. Data is shifted out the SDAx pin until all eight bits are transmitted.
11. The MSSPx module shifts in the $\overline{ACK}$ bit from the slave device and writes its value into the ACKSTAT bit of the SSPxCON2 register.
12. Steps 8-11 are repeated for all transmitted data bytes.
13. The user generates a Stop or Restart condition by setting the PEN or RSEN bits of the SSPxCON2 register. Interrupt is generated once the Stop/Restart condition is complete.


© 2010-2016 Microchip Technology Inc.

**FIGURE 24-28: I²C MASTER MODE WAVEFORM (TRANSMISSION, 7 OR 10-BIT ADDRESS)**



### 24.6.7 I²C MASTER MODE RECEPTION

Master mode reception is enabled by programming the Receive Enable bit, RCEN bit of the SSPxCON2 register.

| **Note:** | The MSSPx module must be in an Idle state before the RCEN bit is set or the RCEN bit will be disregarded. |
|---|---|

The Baud Rate Generator begins counting and on each rollover, the state of the SCLx pin changes (high-to-low/low-to-high) and data is shifted into the SSPxSR. After the falling edge of the eighth clock, the receive enable flag is automatically cleared, the contents of the SSPxSR are loaded into the SSPxBUF, the BF flag bit is set, the SSPxIF flag bit is set and the Baud Rate Generator is suspended from counting, holding SCLx low. The MSSPx is now in Idle state awaiting the next command. When the buffer is read by the CPU, the BF flag bit is automatically cleared. The user can then send an Acknowledge bit at the end of reception by setting the Acknowledge Sequence Enable, ACKEN bit of the SSPxCON2 register.

#### 24.6.7.1 BF Status Flag

In receive operation, the BF bit is set when an address or data byte is loaded into SSPxBUF from SSPxSR. It is cleared when the SSPxBUF register is read.

#### 24.6.7.2 SSPOV Status Flag

In receive operation, the SSPOV bit is set when eight bits are received into the SSPxSR and the BF flag bit is already set from a previous reception.

#### 24.6.7.3 WCOL Status Flag

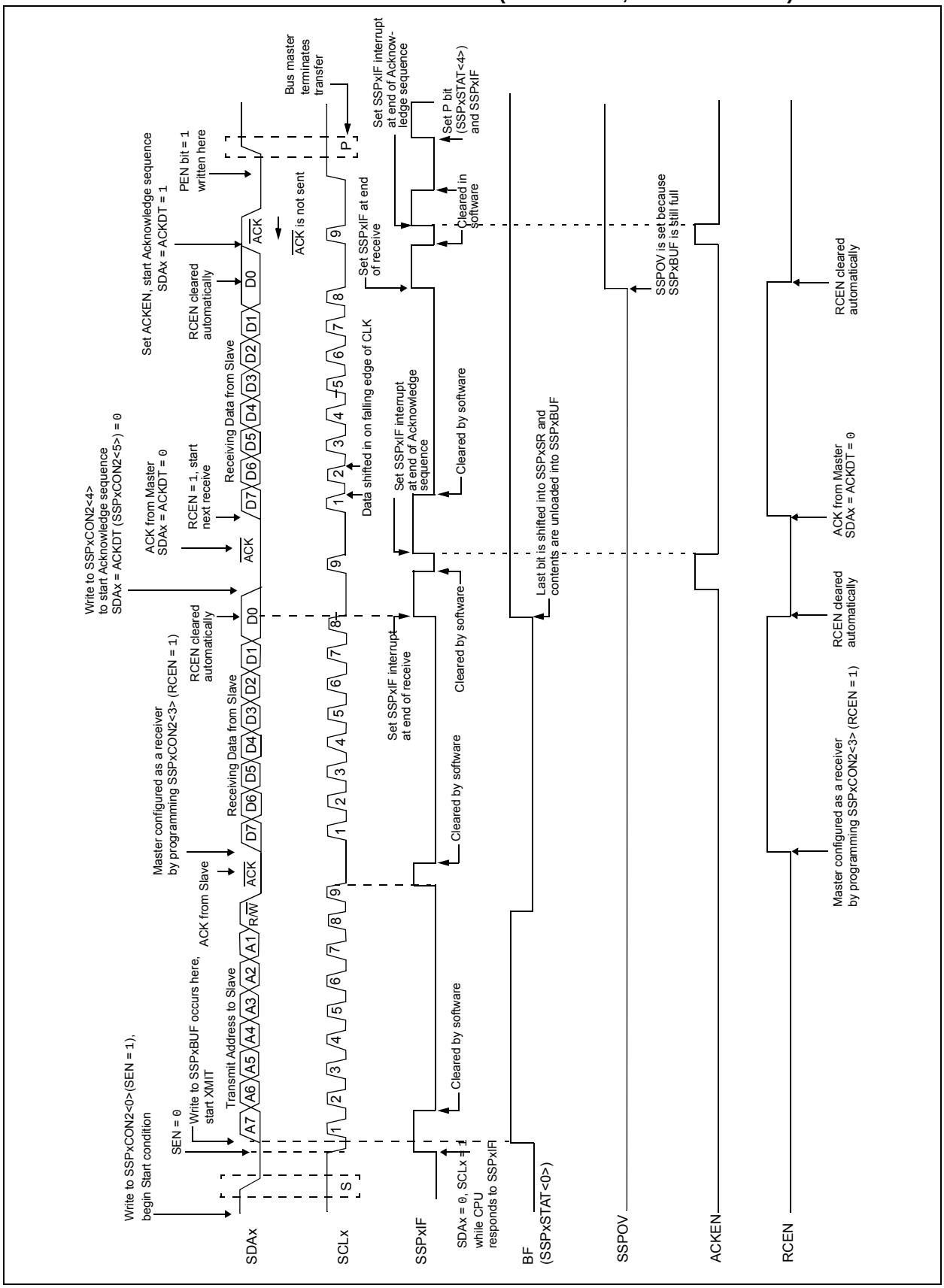
If the user writes the SSPxBUF when a receive is already in progress (i.e., SSPxSR is still shifting in a data byte), the WCOL bit is set and the contents of the buffer are unchanged (the write does not occur).

#### 24.6.7.4 Typical Receive Sequence

1. The user generates a Start condition by setting the SEN bit of the SSPxCON2 register.
2. SSPxIF is set by hardware on completion of the Start.
3. SSPxIF is cleared by software.
4. User writes SSPxBUF with the slave address to transmit and the R/$\overline{W}$ bit set.
5. Address is shifted out the SDAx pin until all eight bits are transmitted. Transmission begins as soon as SSPxBUF is written to.
6. The MSSPx module shifts in the $\overline{ACK}$ bit from the slave device and writes its value into the ACKSTAT bit of the SSPxCON2 register.
7. The MSSPx module generates an interrupt at the end of the ninth clock cycle by setting the SSPxIF bit.
8. User sets the RCEN bit of the SSPxCON2 register and the Master clocks in a byte from the slave.
9. After the eighth falling edge of SCLx, SSPxIF and BF are set.
10. Master clears SSPxIF and reads the received byte from SSPxUF, clears BF.
11. Master sets $\overline{ACK}$ value sent to slave in ACKDT bit of the SSPxCON2 register and initiates the $\overline{ACK}$ by setting the ACKEN bit.
12. Masters $\overline{ACK}$ is clocked out to the Slave and SSPxIF is set.
13. User clears SSPxIF.
14. Steps 8-13 are repeated for each received byte from the slave.
15. Master sends a not $\overline{ACK}$ or Stop to end communication.


© 2010-2016 Microchip Technology Inc.

**FIGURE 24-29: I²C MASTER MODE WAVEFORM (RECEPTION, 7-BIT ADDRESS)**



### 24.6.8 ACKNOWLEDGE SEQUENCE TIMING

An Acknowledge sequence is enabled by setting the Acknowledge Sequence Enable bit, ACKEN bit of the SSPxCON2 register. When this bit is set, the SCLx pin is pulled low and the contents of the Acknowledge data bit are presented on the SDAx pin. If the user wishes to generate an Acknowledge, then the ACKDT bit should be cleared. If not, the user should set the ACKDT bit before starting an Acknowledge sequence. The Baud Rate Generator then counts for one rollover period (TBRG) and the SCLx pin is deasserted (pulled high). When the SCLx pin is sampled high (clock arbitration), the Baud Rate Generator counts for TBRG. The SCLx pin is then pulled low. Following this, the ACKEN bit is automatically cleared, the Baud Rate Generator is turned off and the MSSPx module then goes into Idle mode (Figure 24-29).

#### 24.6.8.1 WCOL Status Flag

If the user writes the SSPxBUF when an Acknowledge sequence is in progress, then the WCOL bit is set and the contents of the buffer are unchanged (the write does not occur).

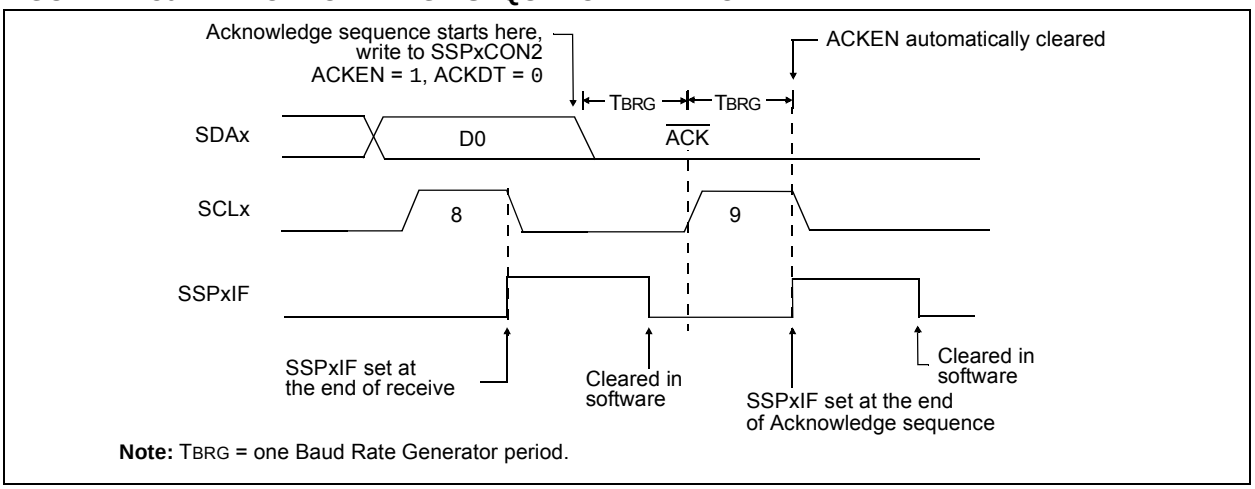
### 24.6.9 STOP CONDITION TIMING

A Stop bit is asserted on the SDAx pin at the end of a receive/transmit by setting the Stop Sequence Enable bit, PEN bit of the SSPxCON2 register. At the end of a receive/transmit, the SCLx line is held low after the falling edge of the ninth clock. When the PEN bit is set, the master will assert the SDAx line low. When the SDAx line is sampled low, the Baud Rate Generator is reloaded and counts down to '0'. When the Baud Rate Generator times out, the SCLx pin will be brought high and one TBRG (Baud Rate Generator rollover count) later, the SDAx pin will be deasserted. When the SDAx pin is sampled high while SCLx is high, the P bit of the SSPxSTAT register is set. A TBRG later, the PEN bit is cleared and the SSPxIF bit is set (Figure 24-30).

#### 24.6.9.1 WCOL Status Flag

If the user writes the SSPxBUF when a Stop sequence is in progress, then the WCOL bit is set and the contents of the buffer are unchanged (the write does not occur).

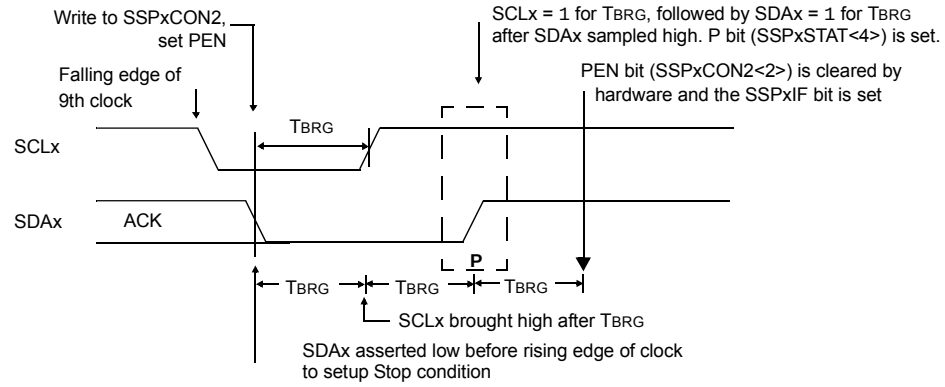
**FIGURE 24-30: ACKNOWLEDGE SEQUENCE WAVEFORM**




**Note:** TBRG = one Baud Rate Generator period.


© 2010-2016 Microchip Technology Inc.

**FIGURE 24-31: STOP CONDITION RECEIVE OR TRANSMIT MODE**




**Note:** TBRG = One Baud Rate Generator period.

### 24.6.10 SLEEP OPERATION

While in Sleep mode, the I$^2$C Slave module can receive addresses or data and when an address match or complete byte transfer occurs, or wake the processor from Sleep (if the MSSPx interrupt is enabled).

### 24.6.11 EFFECTS OF A RESET

A Reset disables the MSSPx module and terminates the current transfer.

### 24.6.12 MULTI-MASTER MODE

In Multi-Master mode, the interrupt generation on the detection of the Start and Stop conditions allows the determination of when the bus is free. The Stop (P) and Start (S) bits are cleared from a Reset or when the MSSPx module is disabled. Control of the I$^2$C bus may be taken when the P bit of the SSPxSTAT register is set, or the bus is Idle, with both the S and P bits clear. When the bus is busy, enabling the SSPx interrupt will generate the interrupt when the Stop condition occurs.

In multi-master operation, the SDAx line must be monitored for arbitration to see if the signal level is the expected output level. This check is performed by hardware with the result placed in the BCLxIF bit.

The states where arbitration can be lost are:

- Address Transfer
- Data Transfer
- A Start Condition
- A Repeated Start Condition
- An Acknowledge Condition

### 24.6.13 MULTI -MASTER COMMUNICATION, BUS COLLISION AND BUS ARBITRATION

Multi-Master mode support is achieved by bus arbitration. When the master outputs address/data bits onto the SDAx pin, arbitration takes place when the master outputs a '1' on SDAx, by letting SDAx float high and another master asserts a '0'. When the SCLx pin floats high, data should be stable. If the expected data on SDAx is a '1' and the data sampled on the SDAx pin is '0', then a bus collision has taken place. The master will set the Bus Collision Interrupt Flag, BCLxIF, and reset the I$^2$C port to its Idle state (Figure 24-31).

If a transmit was in progress when the bus collision occurred, the transmission is halted, the BF flag is cleared, the SDAx and SCLx lines are deasserted and the SSPxBUF can be written to. When the user services the bus collision Interrupt Service Routine and if the I$^2$C bus is free, the user can resume communication by asserting a Start condition.

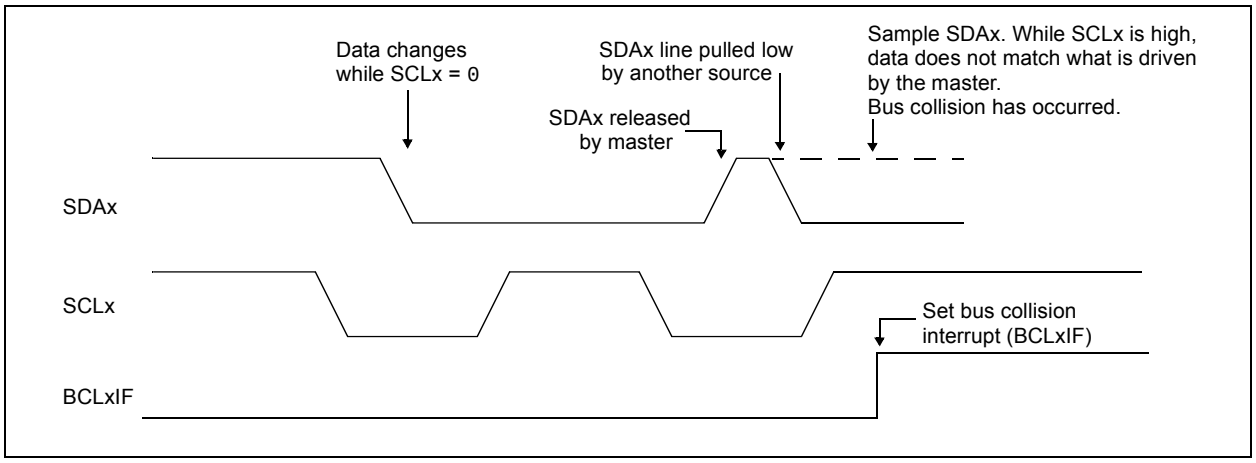
If a Start, Repeated Start, Stop or Acknowledge condition was in progress when the bus collision occurred, the condition is aborted, the SDAx and SCLx lines are deasserted and the respective control bits in the SSPxCON2 register are cleared. When the user services the bus collision Interrupt Service Routine and if the I$^2$C bus is free, the user can resume communication by asserting a Start condition.

The master will continue to monitor the SDAx and SCLx pins. If a Stop condition occurs, the SSPxIF bit will be set.

A write to the SSPxBUF will start the transmission of data at the first data bit, regardless of where the transmitter left off when the bus collision occurred.

In Multi-Master mode, the interrupt generation on the detection of Start and Stop conditions allows the determination of when the bus is free. Control of the I$^2$C bus can be taken when the P bit is set in the SSPxSTAT register, or the bus is Idle and the S and P bits are cleared.

**FIGURE 24-32: BUS COLLISION TIMING FOR TRANSMIT AND ACKNOWLEDGE**





© 2010-2016 Microchip Technology Inc.

#### 24.6.13.1 Bus Collision During a Start Condition

During a Start condition, a bus collision occurs if:

a) SDAx or SCLx are sampled low at the beginning of the Start condition (Figure 24-32).

b) SCLx is sampled low before SDAx is asserted low (Figure 24-33).

During a Start condition, both the SDAx and the SCLx pins are monitored.

If the SDAx pin is already low, or the SCLx pin is already low, then all of the following occur:

- the Start condition is aborted
- the BCLxIF flag is set and
- the MSSPx module is reset to its Idle state (Figure 24-32).

The Start condition begins with the SDAx and SCLx pins deasserted. When the SDAx pin is sampled high, the Baud Rate Generator is loaded and counts down. If the SCLx pin is sampled low while SDAx is high, a bus collision occurs because it is assumed that another master is attempting to drive a data '1' during the Start condition.

If the SDAx pin is sampled low during this count, the BRG is reset and the SDAx line is asserted early (Figure 24-34). If, however, a '1' is sampled on the SDAx pin, the SDAx pin is asserted low at the end of the BRG count. The Baud Rate Generator is then reloaded and counts down to zero; if the SCLx pin is sampled as '0' during this time, a bus collision does not occur. At the end of the BRG count, the SCLx pin is asserted low.

**Note:** The reason that bus collision is not a factor during a Start condition is that no two bus masters can assert a Start condition at the exact same time. Therefore, one master will always assert SDAx before the other. This condition does not cause a bus collision because the two masters must be allowed to arbitrate the first address following the Start condition. If the address is the same, arbitration must be allowed to continue into the data portion, Repeated Start or Stop conditions.

**FIGURE 24-33: BUS COLLISION DURING START CONDITION (SDAx ONLY)**

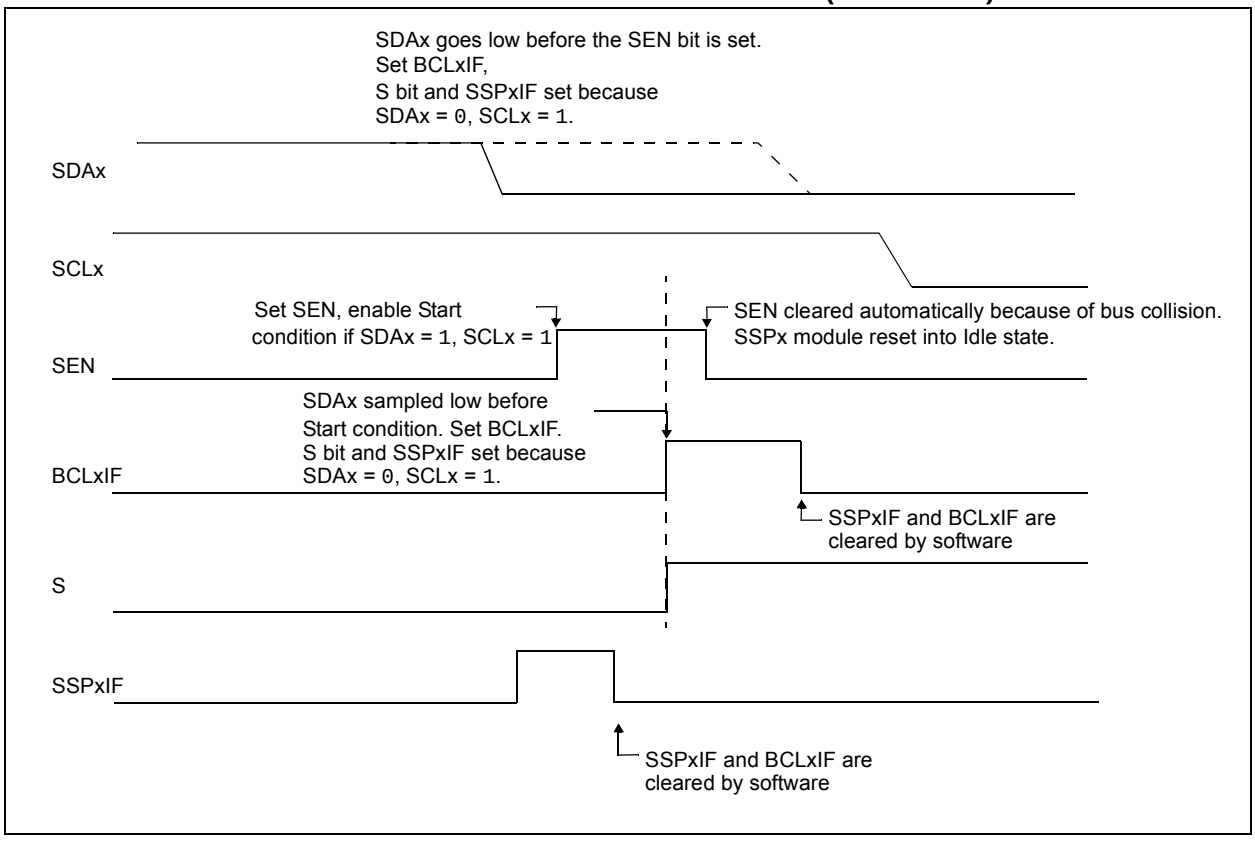


**FIGURE 24-34: BUS COLLISION DURING START CONDITION (SCLx = 0)**

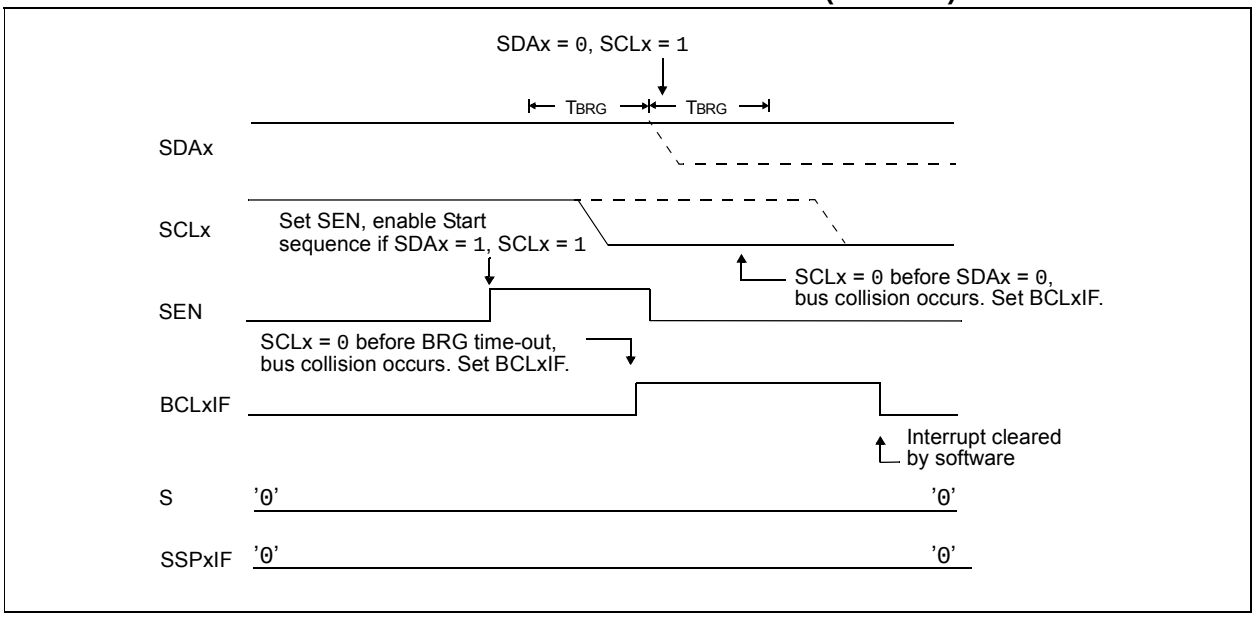
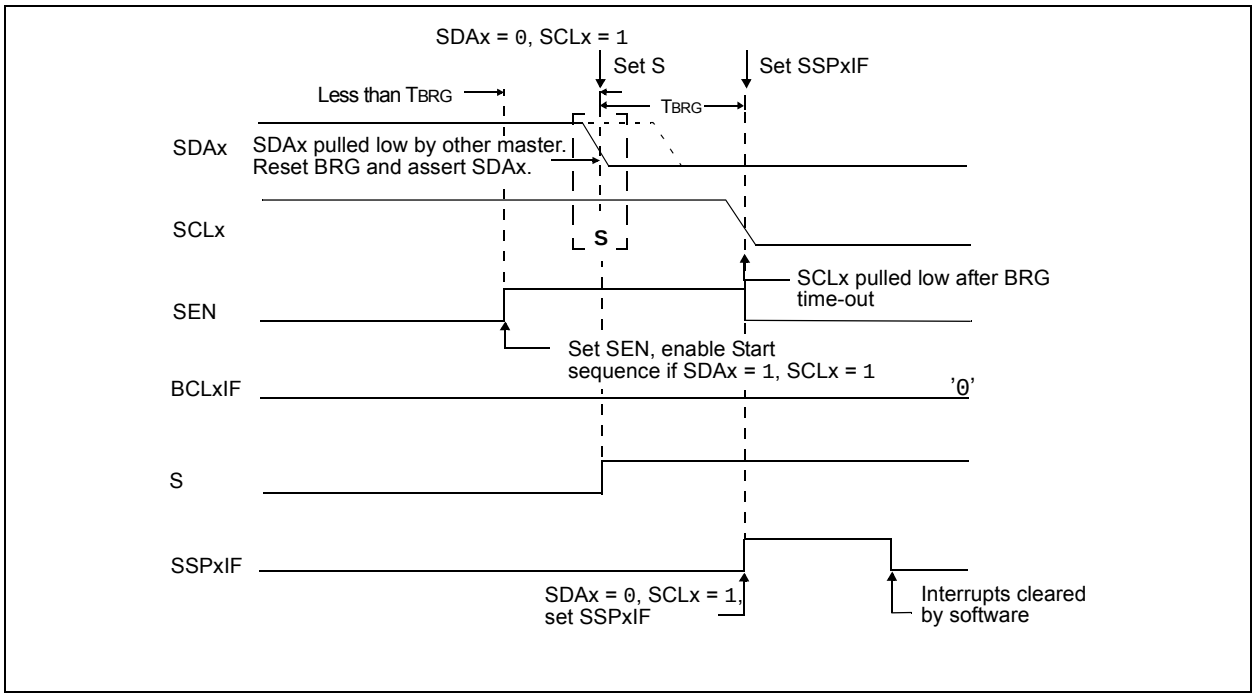



**FIGURE 24-35: BRG RESET DUE TO SDA ARBITRATION DURING START CONDITION**





© 2010-2016 Microchip Technology Inc.

### 24.6.13.2 Bus Collision During a Repeated Start Condition

During a Repeated Start condition, a bus collision occurs if:

a) A low level is sampled on SDAx when SCLx goes from low level to high level.

b) SCLx goes low before SDAx is asserted low, indicating that another master is attempting to transmit a data '1'.

When the user releases SDAx and the pin is allowed to float high, the BRG is loaded with SSPxADD and counts down to zero. The SCLx pin is then deasserted and when sampled high, the SDAx pin is sampled.

If SDAx is low, a bus collision has occurred (i.e., another master is attempting to transmit a data '0', Figure 24-35). If SDAx is sampled high, the BRG is reloaded and begins counting. If SDAx goes from high-to-low before the BRG times out, no bus collision occurs because no two masters can assert SDAx at exactly the same time.

If SCLx goes from high-to-low before the BRG times out and SDAx has not already been asserted, a bus collision occurs. In this case, another master is attempting to transmit a data '1' during the Repeated Start condition, see Figure 24-36.

If, at the end of the BRG timeout, both SCLx and SDAx are still high, the SDAx pin is driven low and the BRG is reloaded and begins counting. At the end of the count, regardless of the status of the SCLx pin, the SCLx pin is driven low and the Repeated Start condition is complete.

**FIGURE 24-36: BUS COLLISION DURING A REPEATED START CONDITION (CASE 1)**

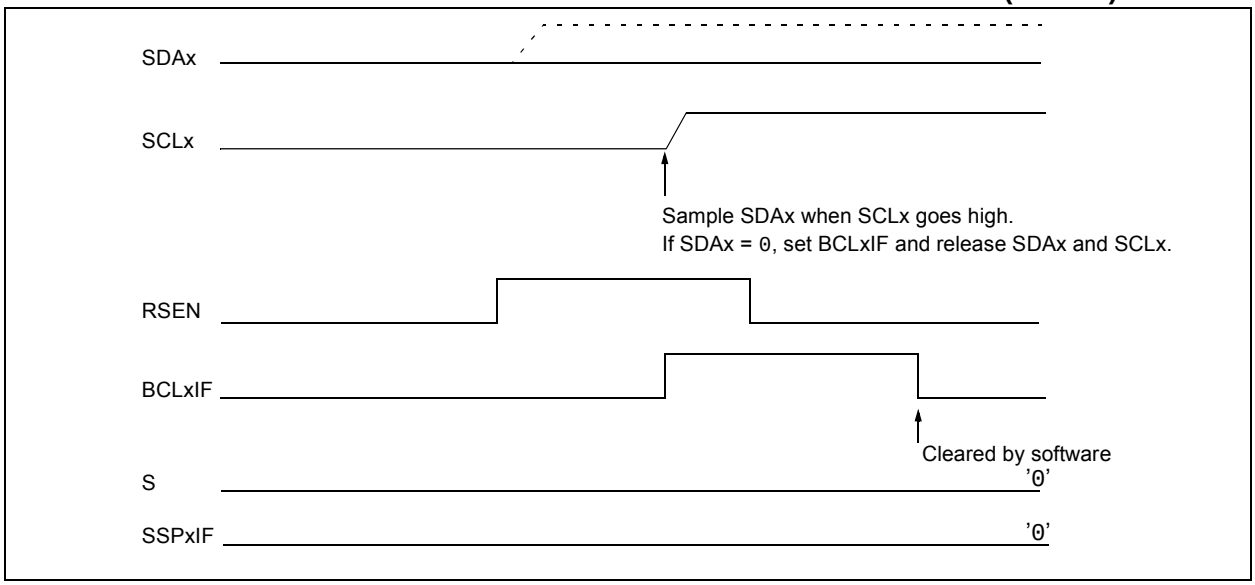
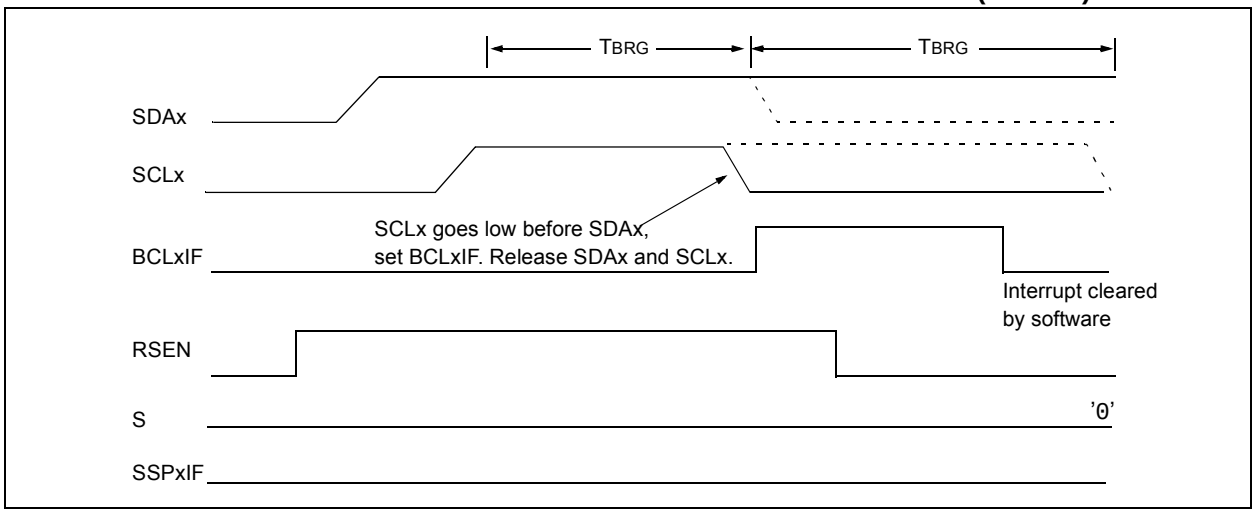



**FIGURE 24-37: BUS COLLISION DURING REPEATED START CONDITION (CASE 2)**



#### 24.6.13.3 Bus Collision During a Stop Condition

Bus collision occurs during a Stop condition if:

a) After the SDAx pin has been deasserted and allowed to float high, SDAx is sampled low after the BRG has timed out.

b) After the SCLx pin is deasserted, SCLx is sampled low before SDAx goes high.

The Stop condition begins with SDAx asserted low. When SDAx is sampled low, the SCLx pin is allowed to float. When the pin is sampled high (clock arbitration), the Baud Rate Generator is loaded with SSPxADD and counts down to zero. After the BRG times out, SDAx is sampled. If SDAx is sampled low, a bus collision has occurred. This is due to another master attempting to drive a data '0' (Figure 24-37). If the SCLx pin is sampled low before SDAx is allowed to float high, a bus collision occurs. This is another case of another master attempting to drive a data '0' (Figure 24-38).

**FIGURE 24-38: BUS COLLISION DURING A STOP CONDITION (CASE 1)**

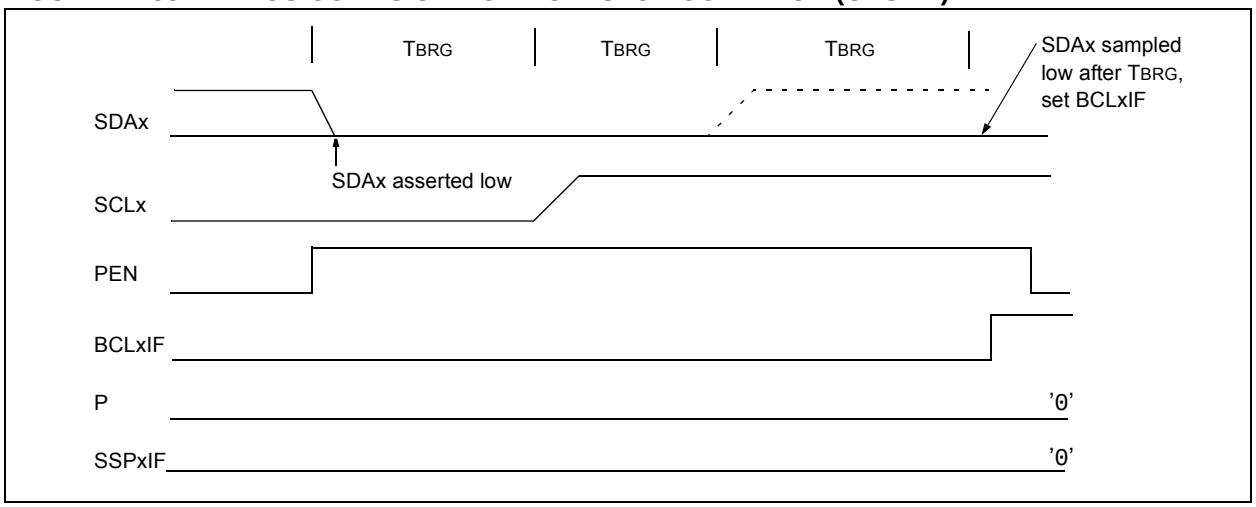
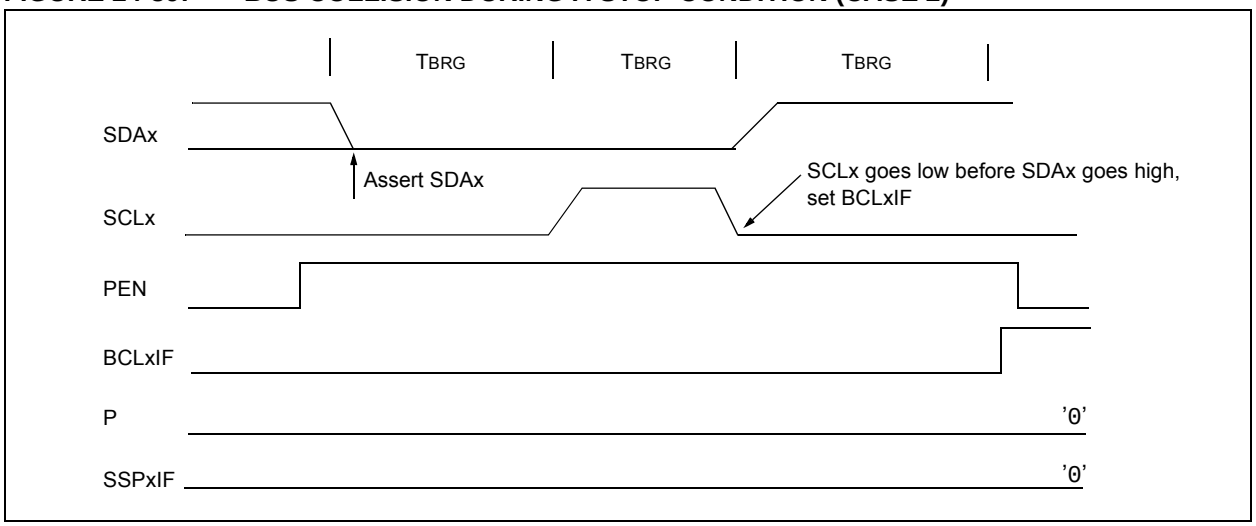



**FIGURE 24-39: BUS COLLISION DURING A STOP CONDITION (CASE 2)**





© 2010-2016 Microchip Technology Inc.

**TABLE 24-3: SUMMARY OF REGISTERS ASSOCIATED WITH I²C OPERATION**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on Page |
|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSPIE | CCP1IE | TMR2IE | TMR1IE | 91 |
| PIE2 | OSFIE | C2IE | C1IE | EEIE | BCLIE | LCDIE | C3IE | CCP2IE(1) | 92 |
| PIE4(1) | — | — | RC2IE | TX2IE | — | — | BCL2IE | SSP2IE | 94 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSPIF | CCP1IF | TMR2IF | TMR1IF | 95 |
| PIR2 | OSFIF | C2IF | C1IF | EEIF | BCLIF | LCDIF | C3IF | CCP2IF(1) | 96 |
| PIR4(1) | — | — | RC2IF | TX2IF | — | — | BCL2IF | SSP2IF | 98 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 131 |
| TRISD | TRISD7 | TRISD6 | TRISD5 | TRISD4 | TRISD3 | TRISD2 | TRISD1 | TRISD0 | 134 |
| SSP1ADD | ADD<7:0> | | | | | | | | 286 |
| SSP1BUF | MSSPx Receive Buffer/Transmit Register | | | | | | | | 236* |
| SSP1CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM<3:0> | | | | 282 |
| SSP1CON2 | GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN | 284 |
| SSP1CON3 | ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN | 285 |
| SSP1MSK | MSK<7:0> | | | | | | | | 286 |
| SSP1STAT | SMP | CKE | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF | 281 |
| SSP2ADD | ADD<7:0> | | | | | | | | 286 |
| SSP2BUF | MSSP2 Receive Buffer/Transmit Register | | | | | | | | 236* |
| SSP2CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM<3:0> | | | | 282 |
| SSP2CON2 | GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN | 284 |
| SSP2CON3 | ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN | 285 |
| SSP2MSK | MSK<7:0> | | | | | | | | 286 |
| SSP2STAT | SMP | CKE | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF | 281 |

**Legend:** — = unimplemented location, read as '0'. Shaded cells are not used by the MSSP module in I²C mode.

* Page provides register information.

**Note 1:** PIC16F1947 only.

## 24.7 BAUD RATE GENERATOR

The MSSPx module has a Baud Rate Generator available for clock generation in both $I^2C$ and SPI Master modes. The Baud Rate Generator (BRG) reload value is placed in the SSPxADD register (Register 24-6). When a write occurs to SSPxBUF, the Baud Rate Generator will automatically begin counting down.

Once the given operation is complete, the internal clock will automatically stop counting and the clock pin will remain in its last state.

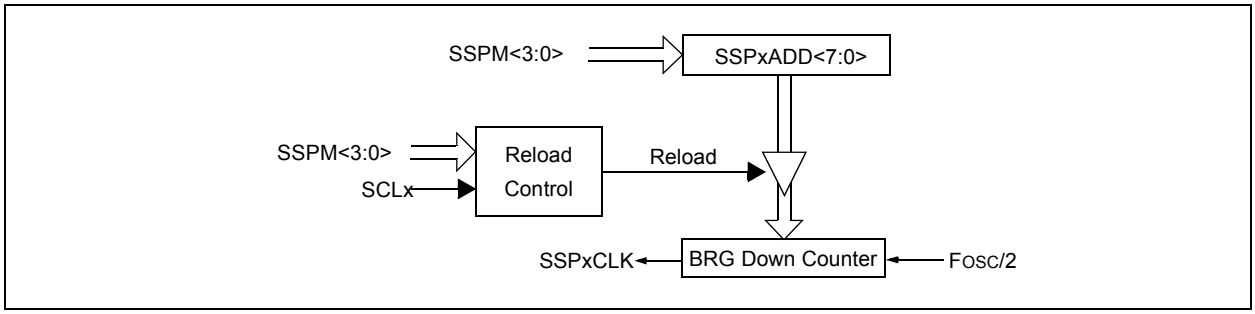
An internal signal “Reload” in Figure 24-39 triggers the value from SSPxADD to be loaded into the BRG counter. This occurs twice for each oscillation of the module clock line. The logic dictating when the reload signal is asserted depends on the mode the MSSPx is being operated in.

Table 24-4 demonstrates clock rates based on instruction cycles and the BRG value loaded into SSPxADD.

**EQUATION 24-1: MSSP CLOCK FREQUENCY CALCULATION**

$$F_{CLOCK} = \frac{F_{OSC}}{(SSPxADD + 1)(4)}$$

**FIGURE 24-40: BAUD RATE GENERATOR BLOCK DIAGRAM**




**Note:** Values of 0x00, 0x01 and 0x02 are not valid for SSPxADD when used as a Baud Rate Generator for $I^2C$. This is an implementation limitation.

**TABLE 24-4: MSSPx CLOCK RATE W/BRG**

| FOSC | FCY | BRG Value | FCLOCK (2 Rollovers of BRG) |
|---|---|---|---|
| 32 MHz | 8 MHz | 13h | 400 kHz(1) |
| 32 MHz | 8 MHz | 19h | 308 kHz |
| 32 MHz | 8 MHz | 4Fh | 100 kHz |
| 16 MHz | 4 MHz | 09h | 400 kHz(1) |
| 16 MHz | 4 MHz | 0Ch | 308 kHz |
| 16 MHz | 4 MHz | 27h | 100 kHz |
| 4 MHz | 1 MHz | 09h | 100 kHz |

**Note 1:** The $I^2C$ interface does not conform to the 400 kHz $I^2C$ specification (which applies to rates greater than 100 kHz) in all details, but may be used with care where higher rates are required by the application.


© 2010-2016 Microchip Technology Inc.

## 24.8 Register Definitions: MSSP Control

### REGISTER 24-1: SSPxSTAT: SSPx STATUS REGISTER

| R/W-0/0 | R/W-0/0 | R-0/0 | R-0/0 | R-0/0 | R-0/0 | R-0/0 | R-0/0 |
|---|---|---|---|---|---|---|---|
| SMP | CKE | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF |
| 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 **SMP:** SPI Data Input Sample bit
SPI Master mode:
1 = Input data sampled at end of data output time
0 = Input data sampled at middle of data output time
SPI Slave mode:
SMP must be cleared when SPI is used in Slave mode
In $I^2C$ Master or Slave mode:
1 = Slew rate control disabled for standard speed mode (100 kHz and 1 MHz)
0 = Slew rate control enabled for high speed mode (400 kHz)

bit 6 **CKE:** SPI Clock Edge Select bit (SPI mode only)
In SPI Master or Slave mode:
1 = Transmit occurs on transition from active to Idle clock state
0 = Transmit occurs on transition from Idle to active clock state
In $I^2C$ mode only:
1 = Enable input logic so that thresholds are compliant with SMBus specification
0 = Disable SMBus specific inputs

bit 5 **D/$\overline{A}$:** Data/$\overline{Address}$ bit ($I^2C$ mode only)
1 = Indicates that the last byte received or transmitted was data
0 = Indicates that the last byte received or transmitted was address

bit 4 **P:** Stop bit
($I^2C$ mode only. This bit is cleared when the MSSPx module is disabled, SSPEN is cleared.)
1 = Indicates that a Stop bit has been detected last (this bit is '0' on Reset)
0 = Stop bit was not detected last

bit 3 **S:** Start bit
($I^2C$ mode only. This bit is cleared when the MSSPx module is disabled, SSPEN is cleared.)
1 = Indicates that a Start bit has been detected last (this bit is '0' on Reset)
0 = Start bit was not detected last

bit 2 **R/$\overline{W}$:** Read/$\overline{Write}$ bit information ($I^2C$ mode only)
This bit holds the R/$\overline{W}$ bit information following the last address match. This bit is only valid from the address match to the next Start bit, Stop bit, or not ACK bit.
In $I^2C$ Slave mode:
1 = Read
0 = Write
In $I^2C$ Master mode:
1 = Transmit is in progress
0 = Transmit is not in progress
OR-ing this bit with SEN, RSEN, PEN, RCEN or ACKEN will indicate if the MSSPx is in Idle mode.

bit 1 **UA:** Update Address bit (10-bit $I^2C$ mode only)
1 = Indicates that the user needs to update the address in the SSPxADD register
0 = Address does not need to be updated

bit 0 **BF:** Buffer Full Status bit
Receive (SPI and $I^2C$ modes):
1 = Receive complete, SSPxBUF is full
0 = Receive not complete, SSPxBUF is empty
Transmit ($I^2C$ mode only):
1 = Data transmit in progress (does not include the $\overline{ACK}$ and Stop bits), SSPxBUF is full
0 = Data transmit complete (does not include the $\overline{ACK}$ and Stop bits), SSPxBUF is empty

**REGISTER 24-2: SSPxCON1: SSPx CONTROL REGISTER 1**

| R/C/HS-0/0 | R/C/HS-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 |
|---|---|---|---|---|---|---|---|
| WCOL | SSPOV | SSPEN | CKP | SSPM<3:0> | | | |
| bit 7 | | | | | | | bit 0 |

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

bit 7 **WCOL:** Write Collision Detect bit

Master mode:

1 = A write to the SSPxBUF register was attempted while the $I^2C$ conditions were not valid for a transmission to be started

0 = No collision

Slave mode:

1 = The SSPxBUF register is written while it is still transmitting the previous word (must be cleared in software)

0 = No collision

bit 6 **SSPOV:** Receive Overflow Indicator bit(1)

In SPI mode:

1 = A new byte is received while the SSPxBUF register is still holding the previous data. In case of overflow, the data in SSPxSR is lost. Overflow can only occur in Slave mode. In Slave mode, the user must read the SSPxBUF, even if only transmitting data, to avoid setting overflow. In Master mode, the overflow bit is not set since each new reception (and transmission) is initiated by writing to the SSPxBUF register (must be cleared in software).

0 = No overflow

In $I^2C$ mode:

1 = A byte is received while the SSPxBUF register is still holding the previous byte. SSPOV is a "don't care" in Transmit mode (must be cleared in software).

0 = No overflow

bit 5 **SSPEN:** Synchronous Serial Port Enable bit

In both modes, when enabled, these pins must be properly configured as input or output

In SPI mode:

1 = Enables serial port and configures SCKx, SDOx, SDIx and $\overline{SSx}$ as the source of the serial port pins(2)

0 = Disables serial port and configures these pins as I/O port pins

In $I^2C$ mode:

1 = Enables the serial port and configures the SDAx and SCLx pins as the source of the serial port pins(3)

0 = Disables serial port and configures these pins as I/O port pins

bit 4 **CKP:** Clock Polarity Select bit

In SPI mode:

1 = Idle state for clock is a high level

0 = Idle state for clock is a low level

In $I^2C$ Slave mode:

SCLx release control

1 = Enable clock

0 = Holds clock low (clock stretch). (Used to ensure data setup time.)

In $I^2C$ Master mode:

Unused in this mode


© 2010-2016 Microchip Technology Inc.

**REGISTER 24-2: SSPxCON1: SSPx CONTROL REGISTER 1 (CONTINUED)**

bit 3-0 **SSPM<3:0>:** Synchronous Serial Port Mode Select bits
0000 = SPI Master mode, clock = FOSC/4
0001 = SPI Master mode, clock = FOSC/16
0010 = SPI Master mode, clock = FOSC/64
0011 = SPI Master mode, clock = TMR2 output/2
0100 = SPI Slave mode, clock = SCKx pin, $\overline{SSx}$ pin control enabled
0101 = SPI Slave mode, clock = SCKx pin, $\overline{SSx}$ pin control disabled, $\overline{SSx}$ can be used as I/O pin
0110 = $I^2C$ Slave mode, 7-bit address
0111 = $I^2C$ Slave mode, 10-bit address
1000 = $I^2C$ Master mode, clock = FOSC / (4 * (SSPxADD+1))**(4)**
1001 = Reserved
1010 = SPI Master mode, clock = FOSC/(4 * (SSPxADD+1))**(5)**
1011 = $I^2C$ firmware controlled Master mode (Slave idle)
1100 = Reserved
1101 = Reserved
1110 = $I^2C$ Slave mode, 7-bit address with Start and Stop bit interrupts enabled
1111 = $I^2C$ Slave mode, 10-bit address with Start and Stop bit interrupts enabled

**Note 1:** In Master mode, the overflow bit is not set since each new reception (and transmission) is initiated by writing to the SSPxBUF register.
**2:** When enabled, these pins must be properly configured as input or output.
**3:** When enabled, the SDAx and SCLx pins must be configured as inputs.
**4:** SSPxADD values of 0, 1 or 2 are not supported for $I^2C$ Mode. Use SSPM = 0000 instead of SSPxADD = 0 to set the SPI Master mode clock to FOSC/4.

**REGISTER 24-3: SSPxCON2: SSPx CONTROL REGISTER 2**

| R/W-0/0 | R-0/0 | R/W-0/0 | R/S/HS-0/0 | R/S/HS-0/0 | R/S/HS-0/0 | R/S/HS-0/0 | R/W/HS-0/0 |
|---|---|---|---|---|---|---|---|
| GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN |
| bit 7 | | | | | | | bit 0 |

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

bit 7 **GCEN:** General Call Enable bit (in $I^2C$ Slave mode only)
1 = Enable interrupt when a general call address (0x00 or 00h) is received in the SSPxSR
0 = General call address disabled

bit 6 **ACKSTAT:** Acknowledge Status bit (in $I^2C$ mode only)
1 = Acknowledge was not received
0 = Acknowledge was received

bit 5 **ACKDT:** Acknowledge Data bit (in $I^2C$ mode only)
In Receive mode:
Value transmitted when the user initiates an Acknowledge sequence at the end of a receive
1 = Not Acknowledge
0 = Acknowledge

bit 4 **ACKEN:** Acknowledge Sequence Enable bit (in $I^2C$ Master mode only)**(1)**
In Master Receive mode:
1 = Initiate Acknowledge sequence on SDAx and SCLx pins, and transmit ACKDT data bit. Automatically cleared by hardware.
0 = Acknowledge sequence idle

bit 3 **RCEN:** Receive Enable bit (in $I^2C$ Master mode only)**(1)**
1 = Enables Receive mode for $I^2C$
0 = Receive idle

bit 2 **PEN:** Stop Condition Enable bit (in $I^2C$ Master mode only)**(1)**
SCKx Release Control:
1 = Initiate Stop condition on SDAx and SCLx pins. Automatically cleared by hardware.
0 = Stop condition Idle

bit 1 **RSEN:** Repeated Start Condition Enable bit (in $I^2C$ Master mode only)**(1)**
1 = Initiate Repeated Start condition on SDAx and SCLx pins. Automatically cleared by hardware.
0 = Repeated Start condition Idle

bit 0 **SEN:** Start Condition Enable bit**(1)**
In Master mode:
1 = Initiate Start condition on SDAx and SCLx pins. Automatically cleared by hardware.
0 = Start condition Idle
In Slave mode:
1 = Clock stretching is enabled for both slave transmit and slave receive (stretch enabled)
0 = Clock stretching is disabled

**Note 1:** For bits ACKEN, RCEN, PEN, RSEN, SEN: If the $I^2C$ module is not in Idle mode, this bit may not be set (no spooling) and the SSPxBUF may not be written (or writes to the SSPxBUF are disabled).


© 2010-2016 Microchip Technology Inc.

**REGISTER 24-4: SSPxCON3: SSPx CONTROL REGISTER 3**

| R-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 |
|---|---|---|---|---|---|---|---|
| ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN |
| 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 **ACKTIM:** Acknowledge Time Status bit ($I^2C$ mode only)**(3)**

1 = Indicates the $I^2C$ bus is in an Acknowledge sequence, set on $8^{TH}$ falling edge of SCLx clock
0 = Not an Acknowledge sequence, cleared on $9^{TH}$ rising edge of SCLx clock

bit 6 **PCIE**: Stop Condition Interrupt Enable bit ($I^2C$ Slave mode only)

1 = Enable interrupt on detection of Stop condition
0 = Stop detection interrupts are disabled**(2)**

bit 5 **SCIE**: Start Condition Interrupt Enable bit ($I^2C$ Slave mode only)

1 = Enable interrupt on detection of Start or Restart conditions
0 = Start detection interrupts are disabled**(2)**

bit 4 **BOEN:** Buffer Overwrite Enable bit

In SPI Slave mode:**(1)**

1 = SSPxBUF updates every time that a new data byte is shifted in ignoring the BF bit
0 = If new byte is received with BF bit of the SSPxSTAT register already set, SSPOV bit of the SSPxCON1 register is set, and the buffer is not updated

In $I^2C$ Master mode and SPI Master mode:

This bit is ignored.

In $I^2C$ Slave mode:

1 = SSPxBUF is updated and $\overline{ACK}$ is generated for a received address/data byte, ignoring the state of the SSPOV bit only if the BF bit = 0.
0 = SSPxBUF is only updated when SSPOV is clear

bit 3 **SDAHT:** SDAx Hold Time Selection bit ($I^2C$ mode only)

1 = Minimum of 300 ns hold time on SDAx after the falling edge of SCLx
0 = Minimum of 100 ns hold time on SDAx after the falling edge of SCLx

bit 2 **SBCDE:** Slave Mode Bus Collision Detect Enable bit ($I^2C$ Slave mode only)

If on the rising edge of SCLx, SDAx is sampled low when the module is outputting a high state, the BCLxIF bit of the PIR2 register is set, and bus goes Idle

1 = Enable slave bus collision interrupts
0 = Slave bus collision interrupts are disabled

bit 1 **AHEN:** Address Hold Enable bit ($I^2C$ Slave mode only)

1 = Following the 8th falling edge of SCLx for a matching received address byte; CKP bit of the SSPxCON1 register will be cleared and the SCLx will be held low.
0 = Address holding is disabled

bit 0 **DHEN:** Data Hold Enable bit ($I^2C$ Slave mode only)

1 = Following the 8th falling edge of SCLx for a received data byte; slave hardware clears the CKP bit of the SSPxCON1 register and SCLx is held low.
0 = Data holding is disabled

**Note 1:** For daisy-chained SPI operation; allows the user to ignore all but the last received byte. SSPOV is still set when a new byte is received and BF = 1, but hardware continues to write the most recent byte to SSPxBUF.

**2:** This bit has no effect in Slave modes that Start and Stop condition detection is explicitly listed as enabled.

**3:** The ACKTIM Status bit is only active when the AHEN bit or DHEN bit is set.

© 2010-2016 Microchip Technology Inc.

**REGISTER 24-5: SSPxMSK: SSPx MASK 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 |
|---|---|---|---|---|---|---|---|
| MSK<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 7-1 **MSK<7:1>:** Mask bits
1 = The received address bit n is compared to SSPxADD<n> to detect $I^2C$ address match
0 = The received address bit n is not used to detect $I^2C$ address match

bit 0 **MSK<0>:** Mask bit for $I^2C$ Slave mode, 10-bit Address
$I^2C$ Slave mode, 10-bit address (SSPM<3:0> = 0111 or 1111):
1 = The received address bit 0 is compared to SSPxADD<0> to detect $I^2C$ address match
0 = The received address bit 0 is not used to detect $I^2C$ address match
$I^2C$ Slave mode, 7-bit address, the bit is ignored

**REGISTER 24-6: SSPxADD: MSSPx ADDRESS AND BAUD RATE REGISTER ($I^2C$ MODE)**

| 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 |
|---|---|---|---|---|---|---|---|
| ADD<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 | |

**Master mode:**

bit 7-0 **ADD<7:0>:** Baud Rate Clock Divider bits
SCLx pin clock period = ((ADD<7:0> + 1) *4)/FOSC

**10-Bit Slave mode – Most Significant Address byte:**

bit 7-3 **Not used:** Unused for Most Significant Address byte. Bit state of this register is a "don't care". Bit pattern sent by master is fixed by $I^2C$ specification and must be equal to '11110'. However, those bits are compared by hardware and are not affected by the value in this register.

bit 2-1 **ADD<2:1>:** Two Most Significant bits of 10-bit address

bit 0 **Not used:** Unused in this mode. Bit state is a "don't care".

**10-Bit Slave mode – Least Significant Address byte:**

bit 7-0 **ADD<7:0>:** Eight Least Significant bits of 10-bit address

**7-Bit Slave mode:**

bit 7-1 **ADD<7:1>:** 7-bit address

bit 0 **Not used:** Unused in this mode. Bit state is a "don't care".


© 2010-2016 Microchip Technology Inc.

## 25.0 ENHANCED UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (EUSART)

**Note:** The PIC16(L)F1946/47 devices have two EUSARTs. Therefore, all information in this section refers to both EUSART 1 and EUSART 2.

The Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART) module is a serial I/O communications peripheral. It contains all the clock generators, shift registers and data buffers necessary to perform an input or output serial data transfer independent of device program execution. The EUSART, also known as a Serial Communications Interface (SCI), can be configured as a full-duplex asynchronous system or half-duplex synchronous system. Full-Duplex mode is useful for communications with peripheral systems, such as CRT terminals and personal computers. Half-Duplex Synchronous mode is intended for communications with peripheral devices, such as A/D or D/A integrated circuits, serial EEPROMs or other microcontrollers. These devices typically do not have internal clocks for baud rate generation and require the external clock signal provided by a master synchronous device.

The EUSART module includes the following capabilities:

- Full-duplex asynchronous transmit and receive
- Two-character input buffer
- One-character output buffer
- Programmable 8-bit or 9-bit character length
- Address detection in 9-bit mode
- Input buffer overrun error detection
- Received character framing error detection
- Half-duplex synchronous master
- Half-duplex synchronous slave
- Programmable clock polarity in synchronous modes
- Sleep operation

The EUSART module implements the following additional features, making it ideally suited for use in Local Interconnect Network (LIN) bus systems:

- Automatic detection and calibration of the baud rate
- Wake-up on Break reception
- 13-bit Break character transmit

The block diagrams of the EUSART transmitter and receiver are shown in Figure 25-1 and Figure 25-2.

**FIGURE 25-1: EUSART TRANSMIT BLOCK DIAGRAM**

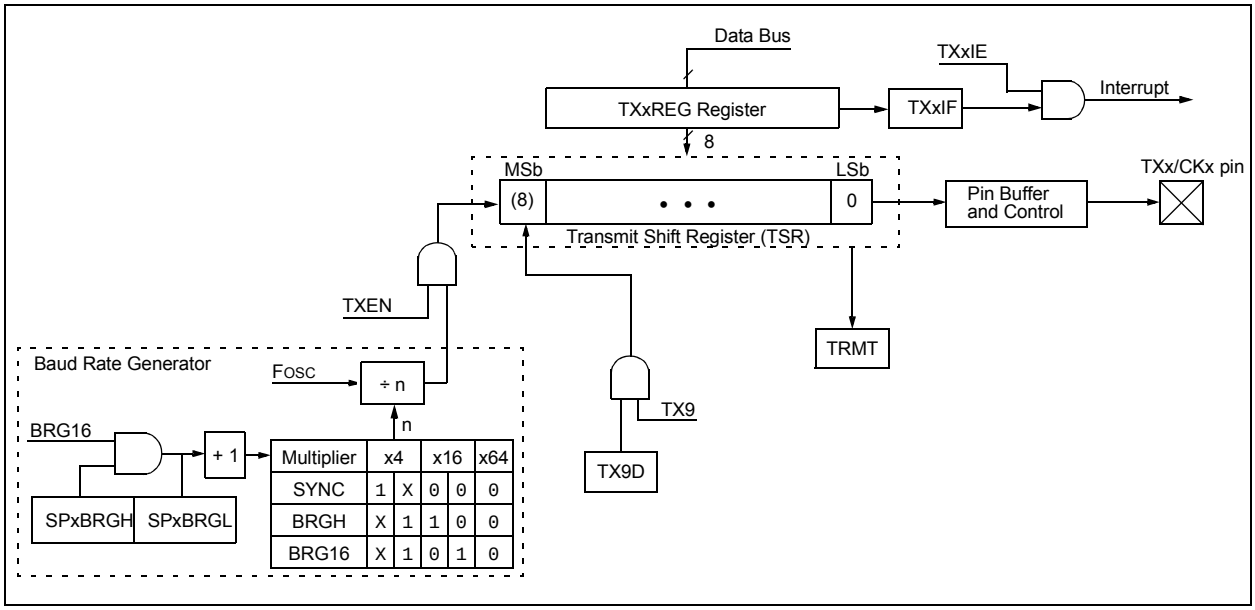
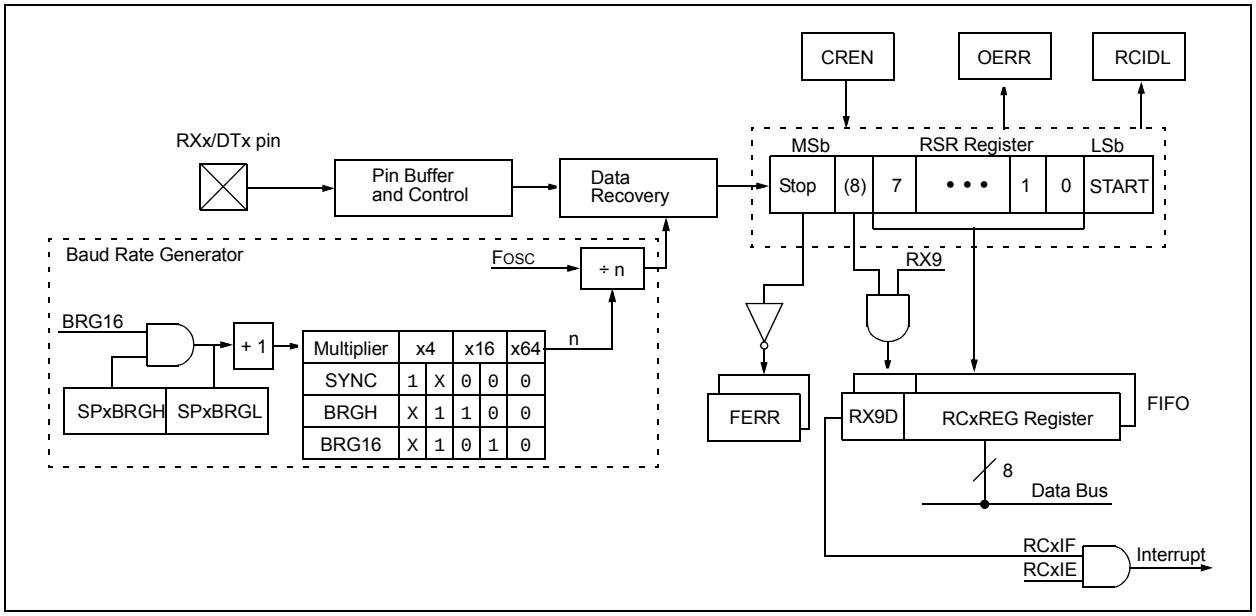


**FIGURE 25-2: EUSART RECEIVE BLOCK DIAGRAM**




The operation of the EUSART module is controlled through three registers:

- Transmit Status and Control (TXxSTA)
- Receive Status and Control (RCxSTA)
- Baud Rate Control (BAUDxCON)

These registers are detailed in Register 25-1, Register 25-2 and Register 25-3, respectively.

For all modes of the EUSART operation, the TRIS control bits corresponding to the RXx/DTx and TXx/CKx pins should be set to '1'. The EUSART control will automatically reconfigure the pin from input to output, as needed.

When the receiver or transmitter section is not enabled, then the corresponding RXx/DTx or TXx/CKx pin may be used for general purpose input and output.


© 2010-2016 Microchip Technology Inc.

## 25.1 EUSART Asynchronous Mode

The EUSART transmits and receives data using the standard non-return-to-zero (NRZ) format. NRZ is implemented with two levels: a VOH Mark state which represents a '1' data bit, and a VOL Space state which represents a '0' data bit. NRZ refers to the fact that consecutively transmitted data bits of the same value stay at the output level of that bit without returning to a neutral level between each bit transmission. An NRZ transmission port idles in the Mark state. Each character transmission consists of one Start bit followed by eight or nine data bits and is always terminated by one or more Stop bits. The Start bit is always a space and the Stop bits are always marks. The most common data format is eight bits. Each transmitted bit persists for a period of 1/(Baud Rate). An on-chip dedicated 8-bit/16-bit Baud Rate Generator is used to derive standard baud rate frequencies from the system oscillator. See Table 25-5 for examples of baud rate configurations.

The EUSART transmits and receives the LSb first. The EUSART's transmitter and receiver are functionally independent, but share the same data format and baud rate. Parity is not supported by the hardware, but can be implemented in software and stored as the ninth data bit.

### 25.1.1 EUSART ASYNCHRONOUS TRANSMITTER

The EUSART transmitter block diagram is shown in Figure 25-1. The heart of the transmitter is the serial Transmit Shift Register (TSR), which is not directly accessible by software. The TSR obtains its data from the transmit buffer, which is the TXxREG register.

#### 25.1.1.1 Enabling the Transmitter

The EUSART transmitter is enabled for asynchronous operations by configuring the following three control bits:

- TXEN = 1
- SYNC = 0
- SPEN = 1

All other EUSART control bits are assumed to be in their default state.

Setting the TXEN bit of the TXxSTA register enables the transmitter circuitry of the EUSART. Clearing the SYNC bit of the TXxSTA register configures the EUSART for asynchronous operation. Setting the SPEN bit of the RCxSTA register enables the EUSART. The programmer must set the corresponding TRIS bit to configure the TXx/CKx I/O pin as an output. If the TXx/CKx pin is shared with an analog peripheral, the analog I/O function must be disabled by clearing the corresponding ANSEL bit.

| **Note:** | The TXxIF transmitter interrupt flag is set when the TXEN enable bit is set. |
|---|---|

#### 25.1.1.2 Transmitting Data

A transmission is initiated by writing a character to the TXxREG register. If this is the first character, or the previous character has been completely flushed from the TSR, the data in the TXxREG is immediately transferred to the TSR register. If the TSR still contains all or part of a previous character, the new character data is held in the TXxREG until the Stop bit of the previous character has been transmitted. The pending character in the TXxREG is then transferred to the TSR in one TCY immediately following the Stop bit transmission. The transmission of the Start bit, data bits and Stop bit sequence commences immediately following the transfer of the data to the TSR from the TXxREG.

#### 25.1.1.3 Transmit Data Polarity

The polarity of the transmit data can be controlled with the SCKP bit of the BAUDxCON register. The default state of this bit is '0' which selects high true transmit idle and data bits. Setting the SCKP bit to '1' will invert the transmit data resulting in low true idle and data bits. The SCKP bit controls transmit data polarity only in Asynchronous mode. In Synchronous mode the SCKP bit has a different function. See **Section 25.5.1.2 "Clock Polarity"**.

#### 25.1.1.4 Transmit Interrupt Flag

The TXxIF interrupt flag bit of the PIR1/PIR3 register is set whenever the EUSART transmitter is enabled and no character is being held for transmission in the TXxREG. In other words, the TXxIF bit is only clear when the TSR is busy with a character and a new character has been queued for transmission in the TXxREG. The TXxIF flag bit is not cleared immediately upon writing TXxREG. TXxIF becomes valid in the second instruction cycle following the write execution. Polling TXxIF immediately following the TXxREG write will return invalid results. The TXxIF bit is read-only, it cannot be set or cleared by software.

The TXxIF interrupt can be enabled by setting the TXxIE interrupt enable bit of the PIE1/PIE4 register. However, the TXxIF flag bit will be set whenever the TXxREG is empty, regardless of the state of the TXxIE enable bit.

To use interrupts when transmitting data, set the TXxIE bit only when there is more data to send. Clear the TXxIE interrupt enable bit upon writing the last character of the transmission to the TXxREG.

#### 25.1.1.5 TSR Status

The TRMT bit of the TXxSTA register indicates the status of the TSR register. This is a read-only bit. The TRMT bit is set when the TSR register is empty and is cleared when a character is transferred to the TSR register from the TXxREG. The TRMT bit remains clear until all bits have been shifted out of the TSR register. No interrupt logic is tied to this bit, so the user needs to poll this bit to determine the TSR status.

| **Note:** | The TSR register is not mapped in data memory, so it is not available to the user. |
|---|---|

#### 25.1.1.6 Transmitting 9-Bit Characters

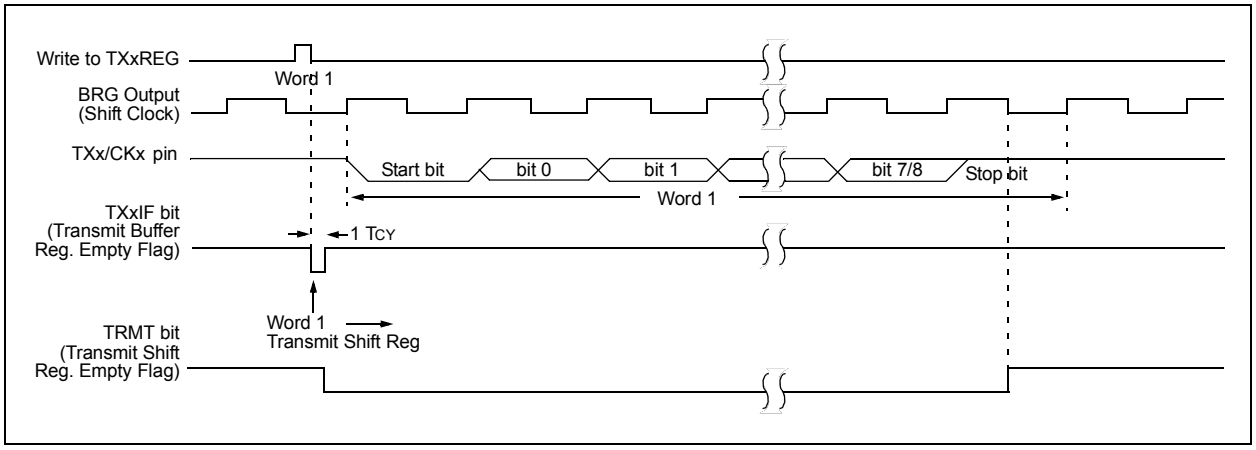
The EUSART supports 9-bit character transmissions. When the TX9 bit of the TXxSTA register is set the EUSART will shift nine bits out for each character transmitted. The TX9D bit of the TXxSTA register is the ninth, and Most Significant, data bit. When transmitting 9-bit data, the TX9D data bit must be written before writing the eight Least Significant bits into the TXxREG. All nine bits of data will be transferred to the TSR shift register immediately after the TXxREG is written.

A special 9-bit Address mode is available for use with multiple receivers. See **Section 25.1.2.7 "Address Detection"** for more information on the Address mode.

#### 25.1.1.7 Asynchronous Transmission Setup:

1. Initialize the SPxBRGH:SPxBRGL register pair and the BRGH and BRG16 bits to achieve the desired baud rate (see **Section 25.4 "EUSART Baud Rate Generator (BRG)"**).
2. Set the RXx/DTx and TXx/CKx TRIS controls to '1'.
3. Enable the asynchronous serial port by clearing the SYNC bit and setting the SPEN bit.
4. If 9-bit transmission is desired, set the TX9 control bit. A set ninth data bit will indicate that the eight Least Significant data bits are an address when the receiver is set for address detection.
5. Set the SCKP control bit if inverted transmit data polarity is desired.
6. Enable the transmission by setting the TXEN control bit. This will cause the TXxIF interrupt bit to be set.
7. If interrupts are desired, set the TXxIE interrupt enable bit. An interrupt will occur immediately provided that the GIE and PEIE bits of the INTCON register are also set.
8. If 9-bit transmission is selected, the ninth bit should be loaded into the TX9D data bit.
9. Load 8-bit data into the TXxREG register. This will start the transmission.

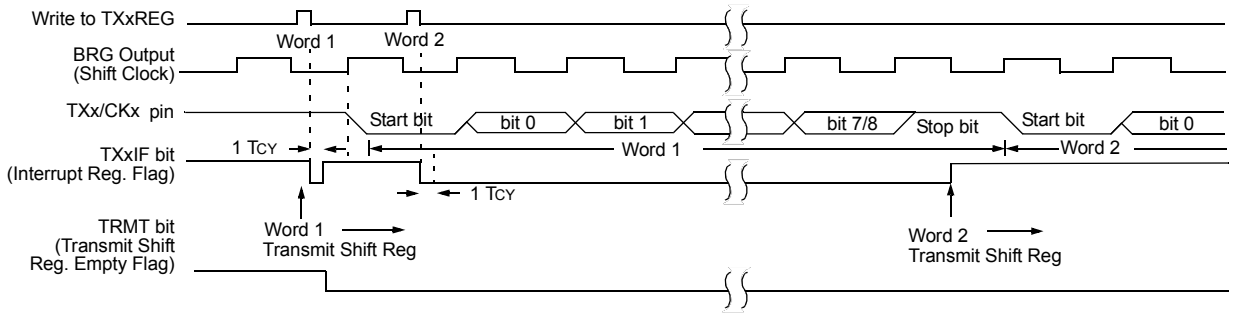
**FIGURE 25-3: ASYNCHRONOUS TRANSMISSION**





© 2010-2016 Microchip Technology Inc.

**FIGURE 25-4: ASYNCHRONOUS TRANSMISSION (BACK-TO-BACK)**




**Note:** This timing diagram shows two consecutive transmissions.

**TABLE 25-1: REGISTERS ASSOCIATED WITH ASYNCHRONOUS TRANSMISSION**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on page |
|---|---|---|---|---|---|---|---|---|---|
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 299 |
| BAUD2CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 299 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| PIE1 | — | ADIE | RC1IE | TX1IE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 91 |
| PIE4 | — | — | RC2IE | TX2IE | — | — | BCL2IE | SSP2IE | 94 |
| PIR1 | — | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 95 |
| PIR4 | — | — | RC2IF | TX2IF | — | — | BCL2IF | SSP2IF | 98 |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 298 |
| RC2STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 298 |
| SP1BRGL | EUSART1 Baud Rate Generator, Low Byte | | | | | | | | 300* |
| SP1BRGH | EUSART1 Baud Rate Generator, High Byte | | | | | | | | 300* |
| SP2BRGL | EUSART2 Baud Rate Generator, Low Byte | | | | | | | | 300* |
| SP2BRGH | EUSART2 Baud Rate Generator, High Byte | | | | | | | | 300* |
| TX1REG | EUSART1 Transmit Register | | | | | | | | 292* |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 297 |
| TX2REG | EUSART2 Transmit Register | | | | | | | | 292* |
| TX2STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 297 |

**Legend:** — = unimplemented locations, read as '0'. Shaded bits are not used for asynchronous transmission.
* Page provides register information.

### 25.1.2 EUSART ASYNCHRONOUS RECEIVER

The Asynchronous mode would typically be used in RS-232 systems. The receiver block diagram is shown in Figure 25-2. The data is received on the RXx/DTx pin and drives the data recovery block. The data recovery block is actually a high-speed shifter operating at 16 times the baud rate, whereas the serial Receive Shift Register (RSR) operates at the bit rate. When all eight or nine bits of the character have been shifted in, they are immediately transferred to a two character First-In-First-Out (FIFO) memory. The FIFO buffering allows reception of two complete characters and the start of a third character before software must start servicing the EUSART receiver. The FIFO and RSR registers are not directly accessible by software. Access to the received data is made via the RCxREG register.

#### 25.1.2.1 Enabling the Receiver

The EUSART receiver is enabled for asynchronous operation by configuring the following three control bits:

- CREN = 1
- SYNC = 0
- SPEN = 1

All other EUSART control bits are assumed to be in their default state.

Setting the CREN bit of the RCxSTA register enables the receiver circuitry of the EUSART. Clearing the SYNC bit of the TXxSTA register configures the EUSART for asynchronous operation. Setting the SPEN bit of the RCxSTA register enables the EUSART. The programmer must set the corresponding TRIS bit to configure the RXx/DTx I/O pin as an input.

| | |
|---|---|
| **Note 1:** | If the RX/DT function is on an analog pin, the corresponding ANSEL bit must be cleared for the receiver to function. |

If the RXx/DTx pin is shared with an analog peripheral, the analog I/O function must be disabled by clearing the corresponding ANSEL bit.

#### 25.1.2.2 Receiving Data

The receiver data recovery circuit initiates character reception on the falling edge of the first bit. The first bit, also known as the Start bit, is always a zero. The data recovery circuit counts one-half bit time to the center of the Start bit and verifies that the bit is still a zero. If it is not a zero then the data recovery circuit aborts character reception, without generating an error, and resumes looking for the falling edge of the Start bit. If the Start bit zero verification succeeds then the data recovery circuit counts a full bit time to the center of the next bit. The bit is then sampled by a majority detect circuit and the resulting '0' or '1' is shifted into the RSR. This repeats until all data bits have been sampled and shifted into the RSR. One final bit time is measured and the level sampled. This is the Stop bit, which is always a '1'. If the data recovery circuit samples a '0' in the Stop bit position then a framing error is set for this character, otherwise the framing error is cleared for this character. See **Section 25.1.2.4 "Receive Framing Error"** for more information on framing errors.

Immediately after all data bits and the Stop bit have been received, the character in the RSR is transferred to the EUSART receive FIFO and the RCxIF interrupt flag bit of the PIR1/PIR3 register is set. The top character in the FIFO is transferred out of the FIFO by reading the RCxREG register.

| | |
|---|---|
| **Note:** | If the receive FIFO is overrun, no additional characters will be received until the overrun condition is cleared. See **Section 25.1.2.5 "Receive Overrun Error"** for more information on overrun errors. |


© 2010-2016 Microchip Technology Inc.

#### 25.1.2.3 Receive Interrupts

The RCxIF interrupt flag bit of the PIR1/PIR3 register is set whenever the EUSART receiver is enabled and there is an unread character in the receive FIFO. The RCxIF interrupt flag bit is read-only, it cannot be set or cleared by software.

RCxIF interrupts are enabled by setting the following bits:

- RCxIE interrupt enable bit of the PIE1/PIE4 register
- PEIE peripheral interrupt enable bit of the INTCON register
- GIE global interrupt enable bit of the INTCON register

The RCxIF interrupt flag bit will be set when there is an unread character in the FIFO, regardless of the state of interrupt enable bits.

#### 25.1.2.4 Receive Framing Error

Each character in the receive FIFO buffer has a corresponding framing error Status bit. A framing error indicates that a Stop bit was not seen at the expected time. The framing error status is accessed via the FERR bit of the RCxSTA register. The FERR bit represents the status of the top unread character in the receive FIFO. Therefore, the FERR bit must be read before reading the RCxREG.

The FERR bit is read-only and only applies to the top unread character in the receive FIFO. A framing error (FERR = 1) does not preclude reception of additional characters. It is not necessary to clear the FERR bit. Reading the next character from the FIFO buffer will advance the FIFO to the next character and the next corresponding framing error.

The FERR bit can be forced clear by clearing the SPEN bit of the RCxSTA register which resets the EUSART. Clearing the CREN bit of the RCxSTA register does not affect the FERR bit. A framing error by itself does not generate an interrupt.

| | |
|---|---|
| **Note:** | If all receive characters in the receive FIFO have framing errors, repeated reads of the RCxREG will not clear the FERR bit. |

#### 25.1.2.5 Receive Overrun Error

The receive FIFO buffer can hold two characters. An overrun error will be generated If a third character, in its entirety, is received before the FIFO is accessed. When this happens the OERR bit of the RCxSTA register is set. The characters already in the FIFO buffer can be read but no additional characters will be received until the error is cleared. The error must be cleared by either clearing the CREN bit of the RCxSTA register or by resetting the EUSART by clearing the SPEN bit of the RCxSTA register.

#### 25.1.2.6 Receiving 9-bit Characters

The EUSART supports 9-bit character reception. When the RX9 bit of the RCxSTA register is set, the EUSART will shift nine bits into the RSR for each character received. The RX9D bit of the RCxSTA register is the ninth and Most Significant data bit of the top unread character in the receive FIFO. When reading 9-bit data from the receive FIFO buffer, the RX9D data bit must be read before reading the eight Least Significant bits from the RCxREG.

#### 25.1.2.7 Address Detection

A special Address Detection mode is available for use when multiple receivers share the same transmission line, such as in RS-485 systems. Address detection is enabled by setting the ADDEN bit of the RCxSTA register.

Address detection requires 9-bit character reception. When address detection is enabled, only characters with the ninth data bit set will be transferred to the receive FIFO buffer, thereby setting the RCxIF interrupt bit. All other characters will be ignored.

Upon receiving an address character, user software determines if the address matches its own. Upon address match, user software must disable address detection by clearing the ADDEN bit before the next Stop bit occurs. When user software detects the end of the message, determined by the message protocol used, software places the receiver back into the Address Detection mode by setting the ADDEN bit.

#### 25.1.2.8 Asynchronous Reception Set-up:

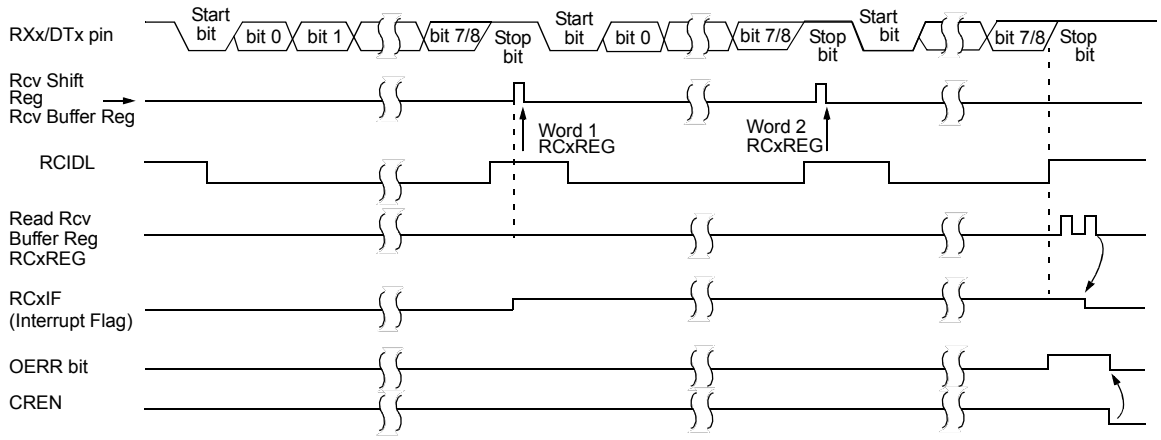
1. Initialize the SPxBRGH:SPxBRGL register pair and the BRGH and BRG16 bits to achieve the desired baud rate (see **Section 25.4 "EUSART Baud Rate Generator (BRG)"**).
2. Set the RXx/DTx and TXx/CKx TRIS controls to '1'.
3. Enable the serial port by setting the SPEN bit and the RXx/DTx pin TRIS bit. The SYNC bit must be clear for asynchronous operation.
4. If interrupts are desired, set the RCxIE interrupt enable bit and set the GIE and PEIE bits of the INTCON register.
5. If 9-bit reception is desired, set the RX9 bit.
6. Enable reception by setting the CREN bit.
7. The RCxIF interrupt flag bit will be set when a character is transferred from the RSR to the receive buffer. An interrupt will be generated if the RCxIE interrupt enable bit was also set.
8. Read the RCxSTA register to get the error flags and, if 9-bit data reception is enabled, the ninth data bit.
9. Get the received eight Least Significant data bits from the receive buffer by reading the RCxREG register.
10. If an overrun occurred, clear the OERR flag by clearing the CREN receiver enable bit.

#### 25.1.2.9 9-bit Address Detection Mode Set-up

This mode would typically be used in RS-485 systems. To set up an Asynchronous Reception with Address Detect Enable:

1. Initialize the SPxBRGH, SPxBRGL register pair and the BRGH and BRG16 bits to achieve the desired baud rate (see **Section 25.4 "EUSART Baud Rate Generator (BRG)"**).
2. Set the RXx/DTx and TXx/CKx TRIS controls to '1'.
3. Enable the serial port by setting the SPEN bit. The SYNC bit must be clear for asynchronous operation.
4. If interrupts are desired, set the RCxIE interrupt enable bit and set the GIE and PEIE bits of the INTCON register.
5. Enable 9-bit reception by setting the RX9 bit.
6. Enable address detection by setting the ADDEN bit.
7. Enable reception by setting the CREN bit.
8. The RCxIF interrupt flag bit will be set when a character with the ninth bit set is transferred from the RSR to the receive buffer. An interrupt will be generated if the RCxIE interrupt enable bit was also set.
9. Read the RCxSTA register to get the error flags. The ninth data bit will always be set.
10. Get the received eight Least Significant data bits from the receive buffer by reading the RCxREG register. Software determines if this is the device's address.
11. If an overrun occurred, clear the OERR flag by clearing the CREN receiver enable bit.
12. If the device has been addressed, clear the ADDEN bit to allow all received data into the receive buffer and generate interrupts.

**FIGURE 25-5: ASYNCHRONOUS RECEPTION**




**Note:** This timing diagram shows three words appearing on the RXx/DTx input. The RCxREG (receive buffer) is read after the third word, causing the OERR (overrun) bit to be set.


© 2010-2016 Microchip Technology Inc.

**TABLE 25-2: REGISTERS ASSOCIATED WITH ASYNCHRONOUS RECEPTION**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 299 |
| BAUD2CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 299 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| PIE1 | — | ADIE | RC1IE | TX1IE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 91 |
| PIE4 | — | — | RC2IE | TX2IE | — | — | BCL2IE | SSP2IE | 94 |
| PIR1 | — | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 95 |
| PIR4 | — | — | RC2IF | TX2IF | — | — | BCL2IF | SSP2IF | 98 |
| RC1REG | EUSART1 Receive Register | | | | | | | | 292* |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 298 |
| RC2REG | EUSART2 Receive Register | | | | | | | | 292* |
| RC2STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 298 |
| SP1BRGL | EUSART1 Baud Rate Generator, Low Byte | | | | | | | | 300* |
| SP1BRGH | EUSART1 Baud Rate Generator, High Byte | | | | | | | | 300* |
| SP2BRGL | EUSART2 Baud Rate Generator, Low Byte | | | | | | | | 300* |
| SP2BRGH | EUSART2 Baud Rate Generator, High Byte | | | | | | | | 300* |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 131 |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 292 |
| TX2STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 297 |

**Legend:** — = unimplemented locations, read as '0'. Shaded bits are not used for asynchronous reception.

* Page provides register information.

## 25.2 Clock Accuracy with Asynchronous Operation

The factory calibrates the internal oscillator block output (HFINTOSC). However, the HFINTOSC frequency may drift as $V_{DD}$ or temperature changes, and this directly affects the asynchronous baud rate. Two methods may be used to adjust the baud rate clock, but both require a reference clock source of some kind.

The first (preferred) method uses the OSCTUNE register to adjust the HFINTOSC output. Adjusting the value in the OSCTUNE register allows for fine resolution changes to the system clock source. See **Section 5.2 "Clock Source Types"** for more information.

The other method adjusts the value in the Baud Rate Generator. This can be done automatically with the Auto-Baud Detect feature (see **Section 25.4.1 "Auto-Baud Detect"**). There may not be fine enough resolution when adjusting the Baud Rate Generator to compensate for a gradual change in the peripheral clock frequency.


© 2010-2016 Microchip Technology Inc.

## 25.3 Register Definitions: EUSART Control

**REGISTER 25-1: TXxSTA: TRANSMIT STATUS AND CONTROL REGISTER**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R-1 | R/W-0 |
|---|---|---|---|---|---|---|---|
| CSRC | TX9 | TXEN(1) | SYNC | SENDB | BRGH | TRMT | TX9D |
| bit 7 | | | | | | | bit 0 |

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

bit 7 **CSRC:** Clock Source Select bit
Asynchronous mode:
Don't care
Synchronous mode:
1 = Master mode (clock generated internally from BRG)
0 = Slave mode (clock from external source)

bit 6 **TX9:** 9-bit Transmit Enable bit
1 = Selects 9-bit transmission
0 = Selects 8-bit transmission

bit 5 **TXEN:** Transmit Enable bit(1)
1 = Transmit enabled
0 = Transmit disabled

bit 4 **SYNC:** EUSART Mode Select bit
1 = Synchronous mode
0 = Asynchronous mode

bit 3 **SENDB:** Send Break Character bit
Asynchronous mode:
1 = Send Sync Break on next transmission (cleared by hardware upon completion)
0 = Sync Break transmission completed
Synchronous mode:
Don't care

bit 2 **BRGH:** High Baud Rate Select bit
Asynchronous mode:
1 = High speed
0 = Low speed
Synchronous mode:
Unused in this mode

bit 1 **TRMT:** Transmit Shift Register Status bit
1 = TSR empty
0 = TSR full

bit 0 **TX9D:** Ninth bit of Transmit Data
Can be address/data bit or a parity bit.

**Note 1:** SREN/CREN overrides TXEN in Sync mode.

**REGISTER 25-2: RCxSTA: RECEIVE STATUS AND CONTROL REGISTER**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R-0 | R-0 | R-x |
|---|---|---|---|---|---|---|---|
| SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D |
| bit 7 | | | | | | | bit 0 |

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

bit 7 **SPEN:** Serial Port Enable bit

1 = Serial port enabled (configures RXx/DTx and TXx/CKx pins as serial port pins)
0 = Serial port disabled (held in Reset)

bit 6 **RX9:** 9-bit Receive Enable bit

1 = Selects 9-bit reception
0 = Selects 8-bit reception

bit 5 **SREN:** Single Receive Enable bit

Asynchronous mode:

Don't care

Synchronous mode – Master:

1 = Enables single receive
0 = Disables single receive
This bit is cleared after reception is complete.

Synchronous mode – Slave

Don't care

bit 4 **CREN:** Continuous Receive Enable bit

Asynchronous mode:

1 = Enables receiver
0 = Disables receiver

Synchronous mode:

1 = Enables continuous receive until enable bit CREN is cleared (CREN overrides SREN)
0 = Disables continuous receive

bit 3 **ADDEN:** Address Detect Enable bit

Asynchronous mode 9-bit (RX9 = 1):

1 = Enables address detection, enable interrupt and load the receive buffer when RSR<8> is set
0 = Disables address detection, all bytes are received and ninth bit can be used as parity bit

Asynchronous mode 8-bit (RX9 = 0):

Don't care

bit 2 **FERR:** Framing Error bit

1 = Framing error (can be updated by reading RCxREG register and receive next valid byte)
0 = No framing error

bit 1 **OERR:** Overrun Error bit

1 = Overrun error (can be cleared by clearing bit CREN)
0 = No overrun error

bit 0 **RX9D:** Ninth bit of Received Data

This can be address/data bit or a parity bit and must be calculated by user firmware.


© 2010-2016 Microchip Technology Inc.

**REGISTER 25-3: BAUDxCON: BAUD RATE CONTROL REGISTER**

| R-0/0 | R-1/1 | U-0 | R/W-0/0 | R/W-0/0 | U-0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN |
| 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 **ABDOVF:** Auto-Baud Detect Overflow bit
Asynchronous mode:
1 = Auto-baud timer overflowed
0 = Auto-baud timer did not overflow
Synchronous mode:
Don't care

bit 6 **RCIDL**: Receive Idle Flag bit
Asynchronous mode:
1 = Receiver is Idle
0 = Start bit has been received and the receiver is receiving
Synchronous mode:
Don't care

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

bit 4 **SCKP**: Synchronous Clock Polarity Select bit
Asynchronous mode:
1 = Transmit inverted data to the TXx/CKx pin
0 = Transmit non-inverted data to the TXx/CKx pin
Synchronous mode:
1 = Data is clocked on rising edge of the clock
0 = Data is clocked on falling edge of the clock

bit 3 **BRG16:** 16-bit Baud Rate Generator bit
1 = 16-bit Baud Rate Generator is used
0 = 8-bit Baud Rate Generator is used

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

bit 1 **WUE:** Wake-up Enable bit
Asynchronous mode:
1 = Receiver is waiting for a falling edge. No character will be received, byte RCIF will be set. WUE will automatically clear after RCIF is set.
0 = Receiver is operating normally
Synchronous mode:
Don't care

bit 0 **ABDEN**: Auto-Baud Detect Enable bit
Asynchronous mode:
1 = Auto-Baud Detect mode is enabled (clears when auto-baud is complete)
0 = Auto-Baud Detect mode is disabled
Synchronous mode:
Don't care

## 25.4 EUSART Baud Rate Generator (BRG)

The Baud Rate Generator (BRG) is an 8-bit or 16-bit timer that is dedicated to the support of both the asynchronous and synchronous EUSART operation. By default, the BRG operates in 8-bit mode. Setting the BRG16 bit of the BAUDxCON register selects 16-bit mode.

The SPxBRGH:SPxBRGL register pair determines the period of the free running baud rate timer. In Asynchronous mode, the multiplier of the baud rate period is determined by both the BRGH bit of the TXxSTA register and the BRG16 bit of the BAUDxCON register. In Synchronous mode, the BRGH bit is ignored.

Example 25-1 provides a sample calculation for determining the desired baud rate, actual baud rate, and baud rate % error.

Typical baud rates and error values for various asynchronous modes have been computed for your convenience and are shown in Table 25-5. It may be advantageous to use the high baud rate (BRGH = 1), or the 16-bit BRG (BRG16 = 1) to reduce the baud rate error. The 16-bit BRG mode is used to achieve slow baud rates for fast oscillator frequencies.

Writing a new value to the SPxBRGH, SPxBRGL register pair causes the BRG timer to be reset (or cleared). This ensures that the BRG does not wait for a timer overflow before outputting the new baud rate.

If the system clock is changed during an active receive operation, a receive error or data loss may result. To avoid this problem, check the status of the RCIDL bit to make sure that the receive operation is Idle before changing the system clock.

**EXAMPLE 25-1: CALCULATING BAUD RATE ERROR**

For a device with FOSC of 16 MHz, desired baud rate of 9600, Asynchronous mode, 8-bit BRG:

$$Desired\ Baud\ Rate = \frac{FOSC}{64([SPxBRGH:SPxBRG] + 1)}$$

Solving for SPxBRGH:SPxBRGL:

$$SPxBRGH:SPxBRGL = \frac{\frac{FOSC}{Desired\ Baud\ Rate}}{64} - 1$$

$$= \frac{\frac{16000000}{9600}}{64} - 1$$

$$= [25.042] = 25$$

$$ActualBaudRate = \frac{16000000}{64(25 + 1)}$$

$$= 9615$$

$$Baud\ Rate\ \%\ Error = \frac{Calc.\ Baud\ Rate - Desired\ Baud\ Rate}{Desired\ Baud\ Rate}$$

$$= \frac{(9615 - 9600)}{9600} = 0.16\%$$

**TABLE 25-3: BAUD RATE FORMULAS**

| Configuration Bits | | | BRG/EUSART Mode | Baud Rate Formula |
|---|---|---|---|---|
| SYNC | BRG16 | BRGH | | |
| 0 | 0 | 0 | 8-bit/Asynchronous | FOSC/[64 (n+1)] |
| 0 | 0 | 1 | 8-bit/Asynchronous | FOSC/[16 (n+1)] |
| 0 | 1 | 0 | 16-bit/Asynchronous | |
| 0 | 1 | 1 | 16-bit/Asynchronous | FOSC/[4 (n+1)] |
| 1 | 0 | x | 8-bit/Synchronous | |
| 1 | 1 | x | 16-bit/Synchronous | |

**Legend:** x = Don't care, n = value of SPxBRGH, SPxBRGL register pair


© 2010-2016 Microchip Technology Inc.

**TABLE 25-4: REGISTERS ASSOCIATED WITH BAUD RATE GENERATOR**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on page |
|---|---|---|---|---|---|---|---|---|---|
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 299 |
| BAUD2CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 299 |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 298 |
| RC2STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 298 |
| SP1BRGL | EUSART1 Baud Rate Generator, Low Byte | | | | | | | | 300* |
| SP1BRGH | EUSART1 Baud Rate Generator, High Byte | | | | | | | | 300* |
| SP2BRGL | EUSART2 Baud Rate Generator, Low Byte | | | | | | | | 300* |
| SP2BRGH | EUSART2 Baud Rate Generator, High Byte | | | | | | | | 300* |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 292 |
| TX2STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 297 |

**Legend:** — = unimplemented, read as '0'. Shaded bits are not used by the BRG.

* Page provides register information.

**TABLE 25-5: BAUD RATES FOR ASYNCHRONOUS MODES**

| BAUD RATE | SYNC = 0, BRGH = 0, BRG16 = 0 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Fosc = 32.000 MHz | | | Fosc = 18.432 MHz | | | Fosc = 16.000 MHz | | | Fosc = 11.0592 MHz | | |
| | Actual Rate | % Error | SPxBRGL value (decimal) | Actual Rate | % Error | SPxBRGL value (decimal) | Actual Rate | % Error | SPxBRGL value (decimal) | Actual Rate | % Error | SPxBRGL value (decimal) |
| 300 | — | — | — | — | — | — | — | — | — | — | — | — |
| 1200 | — | — | — | 1200 | 0.00 | 239 | 1202 | 0.16 | 207 | 1200 | 0.00 | 143 |
| 2400 | 2404 | 0.16 | 207 | 2400 | 0.00 | 119 | 2404 | 0.16 | 103 | 2400 | 0.00 | 71 |
| 9600 | 9615 | 0.16 | 51 | 9600 | 0.00 | 29 | 9615 | 0.16 | 25 | 9600 | 0.00 | 17 |
| 10417 | 10417 | 0.00 | 47 | 10286 | -1.26 | 27 | 10417 | 0.00 | 23 | 10165 | -2.42 | 16 |
| 19.2k | 19.23k | 0.16 | 25 | 19.20k | 0.00 | 14 | 19.23k | 0.16 | 12 | 19.20k | 0.00 | 8 |
| 57.6k | 55.55k | -3.55 | 3 | 57.60k | 0.00 | 7 | — | — | — | 57.60k | 0.00 | 2 |
| 115.2k | — | — | — | — | — | — | — | — | — | — | — | — |

| BAUD RATE | SYNC = 0, BRGH = 0, BRG16 = 0 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Fosc = 8.000 MHz | | | Fosc = 4.000 MHz | | | Fosc = 3.6864 MHz | | | Fosc = 1.000 MHz | | |
| | Actual Rate | % Error | SPxBRGL value (decimal) | Actual Rate | % Error | SPxBRGL value (decimal) | Actual Rate | % Error | SPxBRGL value (decimal) | Actual Rate | % Error | SPxBRGL value (decimal) |
| 300 | — | — | — | 300 | 0.16 | 207 | 300 | 0.00 | 191 | 300 | 0.16 | 51 |
| 1200 | 1202 | 0.16 | 103 | 1202 | 0.16 | 51 | 1200 | 0.00 | 47 | 1202 | 0.16 | 12 |
| 2400 | 2404 | 0.16 | 51 | 2404 | 0.16 | 25 | 2400 | 0.00 | 23 | — | — | — |
| 9600 | 9615 | 0.16 | 12 | — | — | — | 9600 | 0.00 | 5 | — | — | — |
| 10417 | 10417 | 0.00 | 11 | 10417 | 0.00 | 5 | — | — | — | — | — | — |
| 19.2k | — | — | — | — | — | — | 19.20k | 0.00 | 2 | — | — | — |
| 57.6k | — | — | — | — | — | — | 57.60k | 0.00 | 0 | — | — | — |
| 115.2k | — | — | — | — | — | — | — | — | — | — | — | — |

**TABLE 25-5: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)**

| BAUD RATE | SYNC = 0, BRGH = 1, BRG16 = 0 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | FOSC = 32.000 MHz | | | FOSC = 18.432 MHz | | | FOSC = 16.000 MHz | | | FOSC = 11.0592 MHz | | |
| | Actual Rate | % Error | SPxBRGL value (decimal) | Actual Rate | % Error | SPxBRGL value (decimal) | Actual Rate | % Error | SPxBRGL value (decimal) | Actual Rate | % Error | SPxBRGL value (decimal) |
| 300 | — | — | — | — | — | — | — | — | — | — | — | — |
| 1200 | — | — | — | — | — | — | — | — | — | — | — | — |
| 2400 | — | — | — | — | — | — | — | — | — | — | — | — |
| 9600 | 9615 | 0.16 | 207 | 9600 | 0.00 | 119 | 9615 | 0.16 | 103 | 9600 | 0.00 | 71 |
| 10417 | 10417 | 0.00 | 191 | 10378 | -0.37 | 110 | 10417 | 0.00 | 95 | 10473 | 0.53 | 65 |
| 19.2k | 19.23k | 0.16 | 103 | 19.20k | 0.00 | 59 | 19.23k | 0.16 | 51 | 19.20k | 0.00 | 35 |
| 57.6k | 57.14k | -0.79 | 34 | 57.60k | 0.00 | 19 | 58.82k | 2.12 | 16 | 57.60k | 0.00 | 11 |
| 115.2k | 117.64k | 2.12 | 16 | 115.2k | 0.00 | 9 | 111.1k | -3.55 | 8 | 115.2k | 0.00 | 5 |

| BAUD RATE | SYNC = 0, BRGH = 1, BRG16 = 0 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | FOSC = 8.000 MHz | | | FOSC = 4.000 MHz | | | FOSC = 3.6864 MHz | | | FOSC = 1.000 MHz | | |
| | Actual Rate | % Error | SPxBRGL value (decimal) | Actual Rate | % Error | SPxBRGL value (decimal) | Actual Rate | % Error | SPxBRGL value (decimal) | Actual Rate | % Error | SPxBRGL value (decimal) |
| 300 | — | — | — | — | — | — | — | — | — | 300 | 0.16 | 207 |
| 1200 | — | — | — | 1202 | 0.16 | 207 | 1200 | 0.00 | 191 | 1202 | 0.16 | 51 |
| 2400 | 2404 | 0.16 | 207 | 2404 | 0.16 | 103 | 2400 | 0.00 | 95 | 2404 | 0.16 | 25 |
| 9600 | 9615 | 0.16 | 51 | 9615 | 0.16 | 25 | 9600 | 0.00 | 23 | — | — | — |
| 10417 | 10417 | 0.00 | 47 | 10417 | 0.00 | 23 | 10473 | 0.53 | 21 | 10417 | 0.00 | 5 |
| 19.2k | 19231 | 0.16 | 25 | 19.23k | 0.16 | 12 | 19.2k | 0.00 | 11 | — | — | — |
| 57.6k | 55556 | -3.55 | 8 | — | — | — | 57.60k | 0.00 | 3 | — | — | — |
| 115.2k | — | — | — | — | — | — | 115.2k | 0.00 | 1 | — | — | — |

| BAUD RATE | SYNC = 0, BRGH = 0, BRG16 = 1 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | FOSC = 32.000 MHz | | | FOSC = 18.432 MHz | | | FOSC = 16.000 MHz | | | FOSC = 11.0592 MHz | | |
| | Actual Rate | % Error | SPxBRGH: SPxBRGL (decimal) | Actual Rate | % Error | SPxBRGH: SPxBRGL (decimal) | Actual Rate | % Error | SPxBRGH: SPxBRGL (decimal) | Actual Rate | % Error | SPxBRGH: SPxBRGL (decimal) |
| 300 | 300.0 | 0.00 | 6666 | 300.0 | 0.00 | 3839 | 300.03 | 0.01 | 3332 | 300.0 | 0.00 | 2303 |
| 1200 | 1200.1 | 0.02 | 3332 | 1200 | 0.00 | 959 | 1200.5 | 0.04 | 832 | 1200 | 0.00 | 575 |
| 2400 | 2401 | -0.04 | 832 | 2400 | 0.00 | 479 | 2398 | -0.08 | 416 | 2400 | 0.00 | 287 |
| 9600 | 9615 | 0.16 | 207 | 9600 | 0.00 | 119 | 9615 | 0.16 | 103 | 9600 | 0.00 | 71 |
| 10417 | 10417 | 0.00 | 191 | 10378 | -0.37 | 110 | 10417 | 0.00 | 95 | 10473 | 0.53 | 65 |
| 19.2k | 19.23k | 0.16 | 103 | 19.20k | 0.00 | 59 | 19.23k | 0.16 | 51 | 19.20k | 0.00 | 35 |
| 57.6k | 57.14k | -0.79 | 34 | 57.60k | 0.00 | 19 | 58.82k | 2.12 | 16 | 57.60k | 0.00 | 11 |
| 115.2k | 117.6k | 2.12 | 16 | 115.2k | 0.00 | 9 | 111.11k | -3.55 | 8 | 115.2k | 0.00 | 5 |


© 2010-2016 Microchip Technology Inc.

**TABLE 25-5: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)**

| BAUD RATE | SYNC = 0, BRGH = 0, BRG16 = 1 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | FOSC = 8.000 MHz | | | FOSC = 4.000 MHz | | | FOSC = 3.6864 MHz | | | FOSC = 1.000 MHz | | |
| | Actual Rate | % Error | SPxBRGH: SPxBRGL (decimal) | Actual Rate | % Error | SPxBRGH: SPxBRGL (decimal) | Actual Rate | % Error | SPxBRGH: SPxBRGL (decimal) | Actual Rate | % Error | SPxBRGH: SPxBRGL (decimal) |
| 300 | 299.9 | -0.02 | 1666 | 300.1 | 0.04 | 832 | 300.0 | 0.00 | 767 | 300.5 | 0.16 | 207 |
| 1200 | 1199 | -0.08 | 416 | 1202 | 0.16 | 207 | 1200 | 0.00 | 191 | 1202 | 0.16 | 51 |
| 2400 | 2404 | 0.16 | 207 | 2404 | 0.16 | 103 | 2400 | 0.00 | 95 | 2404 | 0.16 | 25 |
| 9600 | 9615 | 0.16 | 51 | 9615 | 0.16 | 25 | 9600 | 0.00 | 23 | — | — | — |
| 10417 | 10417 | 0.00 | 47 | 10417 | 0.00 | 23 | 10473 | 0.53 | 21 | 10417 | 0.00 | 5 |
| 19.2k | 19.23k | 0.16 | 25 | 19.23k | 0.16 | 12 | 19.20k | 0.00 | 11 | — | — | — |
| 57.6k | 55556 | -3.55 | 8 | — | — | — | 57.60k | 0.00 | 3 | — | — | — |
| 115.2k | — | — | — | — | — | — | 115.2k | 0.00 | 1 | — | — | — |

| BAUD RATE | SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | FOSC = 32.000 MHz | | | FOSC = 18.432 MHz | | | FOSC = 16.000 MHz | | | FOSC = 11.0592 MHz | | |
| | Actual Rate | % Error | SPxBRGH: SPxBRGL (decimal) | Actual Rate | % Error | SPxBRGH: SPxBRGL (decimal) | Actual Rate | % Error | SPxBRGH: SPxBRGL (decimal) | Actual Rate | % Error | SPxBRGH: SPxBRGL (decimal) |
| 300 | 300 | 0.00 | 26666 | 300.0 | 0.00 | 15359 | 300.0 | 0.00 | 13332 | 300.0 | 0.00 | 9215 |
| 1200 | 1200 | 0.00 | 6666 | 1200 | 0.00 | 3839 | 1200.1 | 0.01 | 3332 | 1200 | 0.00 | 2303 |
| 2400 | 2400 | 0.01 | 3332 | 2400 | 0.00 | 1919 | 2399.5 | -0.02 | 1666 | 2400 | 0.00 | 1151 |
| 9600 | 9604 | 0.04 | 832 | 9600 | 0.00 | 479 | 9592 | -0.08 | 416 | 9600 | 0.00 | 287 |
| 10417 | 10417 | 0.00 | 767 | 10425 | 0.08 | 441 | 10417 | 0.00 | 383 | 10433 | 0.16 | 264 |
| 19.2k | 19.18k | -0.08 | 416 | 19.20k | 0.00 | 239 | 19.23k | 0.16 | 207 | 19.20k | 0.00 | 143 |
| 57.6k | 57.55k | -0.08 | 138 | 57.60k | 0.00 | 79 | 57.97k | 0.64 | 68 | 57.60k | 0.00 | 47 |
| 115.2k | 115.9 | 0.64 | 68 | 115.2k | 0.00 | 39 | 114.29k | -0.79 | 34 | 115.2k | 0.00 | 23 |

| BAUD RATE | SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | FOSC = 8.000 MHz | | | FOSC = 4.000 MHz | | | FOSC = 3.6864 MHz | | | FOSC = 1.000 MHz | | |
| | Actual Rate | % Error | SPxBRGH: SPxBRGL (decimal) | Actual Rate | % Error | SPxBRGH: SPxBRGL (decimal) | Actual Rate | % Error | SPxBRGH: SPxBRGL (decimal) | Actual Rate | % Error | SPxBRGH: SPxBRGL (decimal) |
| 300 | 300.0 | 0.00 | 6666 | 300.0 | 0.01 | 3332 | 300.0 | 0.00 | 3071 | 300.1 | 0.04 | 832 |
| 1200 | 1200 | -0.02 | 1666 | 1200 | 0.04 | 832 | 1200 | 0.00 | 767 | 1202 | 0.16 | 207 |
| 2400 | 2401 | 0.04 | 832 | 2398 | 0.08 | 416 | 2400 | 0.00 | 383 | 2404 | 0.16 | 103 |
| 9600 | 9615 | 0.16 | 207 | 9615 | 0.16 | 103 | 9600 | 0.00 | 95 | 9615 | 0.16 | 25 |
| 10417 | 10417 | 0.00 | 191 | 10417 | 0.00 | 95 | 10473 | 0.53 | 87 | 10417 | 0.00 | 23 |
| 19.2k | 19.23k | 0.16 | 103 | 19.23k | 0.16 | 51 | 19.20k | 0.00 | 47 | 19.23k | 0.16 | 12 |
| 57.6k | 57.14k | -0.79 | 34 | 58.82k | 2.12 | 16 | 57.60k | 0.00 | 15 | — | — | — |
| 115.2k | 117.6k | 2.12 | 16 | 111.1k | -3.55 | 8 | 115.2k | 0.00 | 7 | — | — | — |

### 25.4.1 AUTO-BAUD DETECT

The EUSART module supports automatic detection and calibration of the baud rate.

In the Auto-Baud Detect (ABD) mode, the clock to the BRG is reversed. Rather than the BRG clocking the incoming RXx signal, the RXx signal is timing the BRG. The Baud Rate Generator is used to time the period of a received 55h (ASCII “U”) which is the Sync character for the LIN bus. The unique feature of this character is that it has five rising edges including the Stop bit edge.

Setting the ABDEN bit of the BAUDxCON register starts the auto-baud calibration sequence (Figure 25.4.2). While the ABD sequence takes place, the EUSART state machine is held in Idle. On the first rising edge of the receive line, after the Start bit, the SPxBRGL begins counting up using the BRG counter clock as shown in Table 25-6. The fifth rising edge will occur on the RXx/DTx pin at the end of the eighth bit period. At that time, an accumulated value totaling the proper BRG period is left in the SPxBRGH:SPxBRGL register pair, the ABDEN bit is automatically cleared, and the RCxIF interrupt flag is set. A read operation on the RCxREG needs to be performed to clear the RCxIF interrupt. RCxREG content should be discarded. When calibrating for modes that do not use the SPxBRGH register the user can verify that the SPxBRGL register did not overflow by checking for 00h in the SPxBRGH register.

The BRG auto-baud clock is determined by the BRG16 and BRGH bits as shown in Table 25-6. During ABD, both the SPxBRGH and SPxBRGL registers are used as a 16-bit counter, independent of the BRG16 bit setting. While calibrating the baud rate period, the SPxBRGH and SPxBRGL registers are clocked at 1/8$^{th}$ the BRG base clock rate. The resulting byte measurement is the average bit time when clocked at full speed.

**Note 1:** If the WUE bit is set with the ABDEN bit, the auto-baud detection will occur on the byte following the Break character (see **Section 25.4.3 “Auto-Wake-up on Break”**).

**2:** It is up to the user to determine that the incoming character baud rate is within the range of the selected BRG clock source. Some combinations of oscillator frequency and EUSART baud rates are not possible.

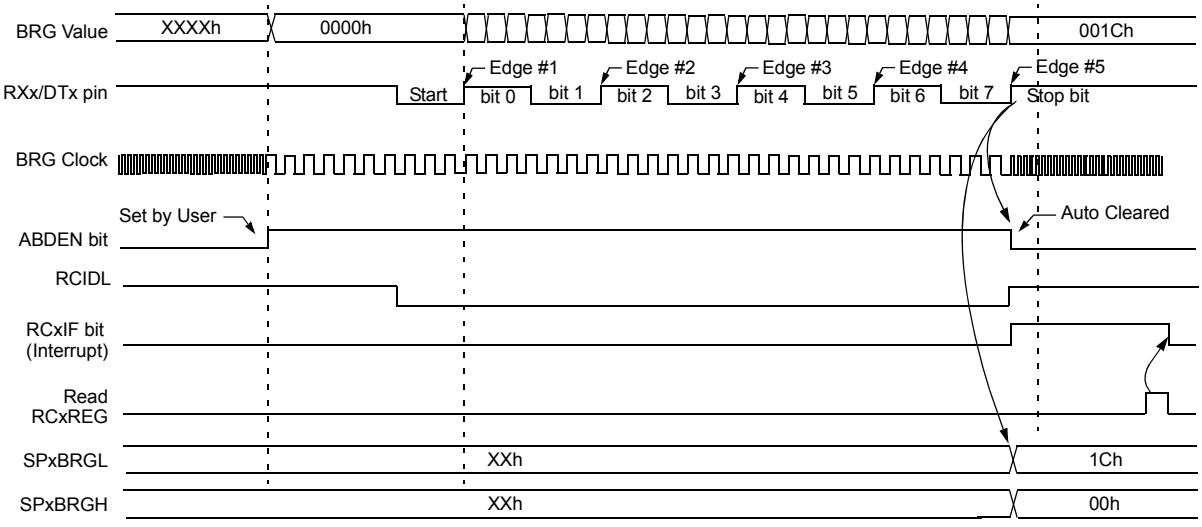
**3:** During the auto-baud process, the auto-baud counter starts counting at 1. Upon completion of the auto-baud sequence, to achieve maximum accuracy, subtract 1 from the SPxBRGH:SPxBRGL register pair.

**TABLE 25-6: BRG COUNTER CLOCK RATES**

| BRG16 | BRGH | BRG Base Clock | BRG ABD Clock |
|---|---|---|---|
| 0 | 0 | FOSC/64 | FOSC/512 |
| 0 | 1 | FOSC/16 | FOSC/128 |
| 1 | 0 | FOSC/16 | FOSC/128 |
| 1 | 1 | FOSC/4 | FOSC/32 |

**Note:** During the ABD sequence, SPxBRGL and SPxBRGH registers are both used as a 16-bit counter, independent of BRG16 setting.

**FIGURE 25-6: AUTOMATIC BAUD RATE CALIBRATION**




**Note 1:** The ABD sequence requires the **EUSART** module to be configured in Asynchronous mode.


© 2010-2016 Microchip Technology Inc.

### 25.4.2 AUTO-BAUD OVERFLOW

During the course of automatic baud detection, the ABDOVF bit of the BAUDxCON register will be set if the baud rate counter overflows before the fifth rising edge is detected on the RX pin. The ABDOVF bit indicates that the counter has exceeded the maximum count that can fit in the 16 bits of the SPxBRGH:SPxBRGL register pair. After the ABDOVF bit has been set, the counter continues to count until the fifth rising edge is detected on the RXx/DTx pin. Upon detecting the fifth RXx/DTx edge, the hardware will set the RCxIF interrupt flag and clear the ABDEN bit of the BAUDxCON register. The RCxIF flag can be subsequently cleared by reading the RCxREG. The ABDOVF flag can be cleared by software directly.

To terminate the auto-baud process before the RCxIF flag is set, clear the ABDEN bit then clear the ABDOVF bit. The ABDOVF bit will remain set if the ABDEN bit is not cleared first.

### 25.4.3 AUTO-WAKE-UP ON BREAK

During Sleep mode, all clocks to the EUSART are suspended. Because of this, the Baud Rate Generator is inactive and a proper character reception cannot be performed. The Auto-Wake-up feature allows the controller to wake-up due to activity on the RXx/DTx line. This feature is available only in Asynchronous mode.

The Auto-Wake-up feature is enabled by setting the WUE bit of the BAUDxCON register. Once set, the normal receive sequence on RXx/DTx is disabled, and the EUSART remains in an Idle state, monitoring for a wake-up event independent of the CPU mode. A wake-up event consists of a high-to-low transition on the RXx/DTx line. (This coincides with the start of a Sync Break or a wake-up signal character for the LIN protocol.)

The EUSART module generates an RCxIF interrupt coincident with the wake-up event. The interrupt is generated synchronously to the Q clocks in normal CPU operating modes (Figure 25-7), and asynchronously if the device is in Sleep mode (Figure 25-8). The interrupt condition is cleared by reading the RCxREG register.

The WUE bit is automatically cleared by the low-to-high transition on the RXx line at the end of the Break. This signals to the user that the Break event is over. At this point, the EUSART module is in Idle mode waiting to receive the next character.

#### 25.4.3.1 Special Considerations

Break Character

To avoid character errors or character fragments during a wake-up event, the wake-up character must be all zeros.

When the wake-up is enabled the function works independent of the low time on the data stream. If the WUE bit is set and a valid non-zero character is received, the low time from the Start bit to the first rising edge will be interpreted as the wake-up event. The remaining bits in the character will be received as a fragmented character and subsequent characters can result in framing or overrun errors.

Therefore, the initial character in the transmission must be all ‘0’s. This must be ten or more bit times, 13-bit times recommended for LIN bus, or any number of bit times for standard RS-232 devices.

Oscillator Startup Time

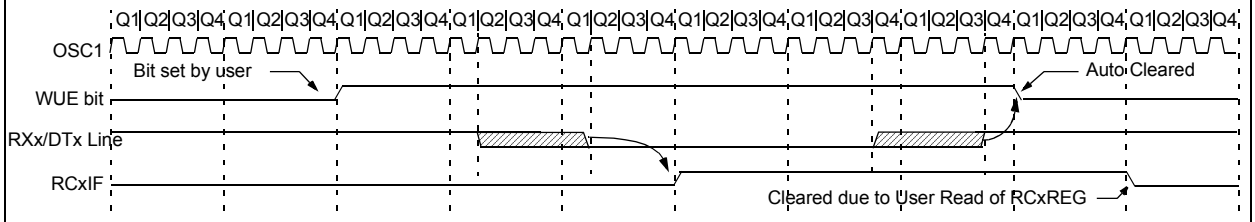
Oscillator start-up time must be considered, especially in applications using oscillators with longer start-up intervals (i.e., LP, XT or HS/PLL mode). The Sync Break (or wake-up signal) character must be of sufficient length, and be followed by a sufficient interval, to allow enough time for the selected oscillator to start and provide proper initialization of the EUSART.

WUE Bit

The wake-up event causes a receive interrupt by setting the RCxIF bit. The WUE bit is cleared by hardware by a rising edge on RXx/DTx. The interrupt condition is then cleared by software by reading the RCxREG register and discarding its contents.

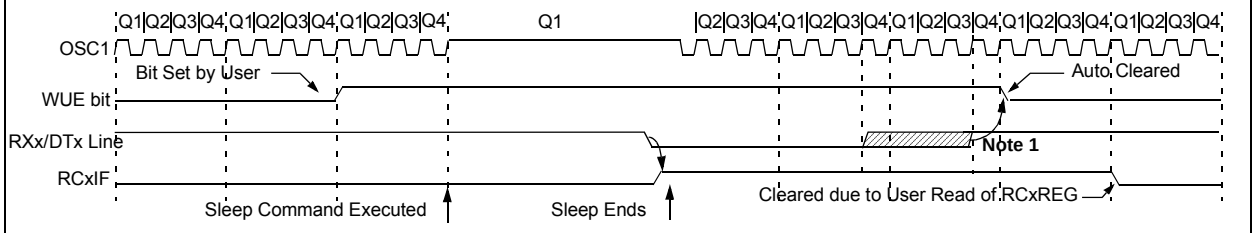
To ensure that no actual data is lost, check the RCIDL bit to verify that a receive operation is not in process before setting the WUE bit. If a receive operation is not occurring, the WUE bit may then be set just prior to entering the Sleep mode.

**FIGURE 25-7: AUTO-WAKE-UP BIT (WUE) TIMING DURING NORMAL OPERATION**




**Note 1:** The **EUSART** remains in Idle while the WUE bit is set.

**FIGURE 25-8: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING SLEEP**




**Note 1:** If the wake-up event requires long oscillator warm-up time, the automatic clearing of the WUE bit can occur while the `stposc` signal is still active. This sequence should not depend on the presence of Q clocks.

**2:** The **EUSART** remains in Idle while the WUE bit is set.


© 2010-2016 Microchip Technology Inc.

### 25.4.4 BREAK CHARACTER SEQUENCE

The EUSART module has the capability of sending the special Break character sequences that are required by the LIN bus standard. A Break character consists of a Start bit, followed by 12 '0' bits and a Stop bit.

To send a Break character, set the SENDB and TXEN bits of the TXxSTA register. The Break character transmission is then initiated by a write to the TXxREG. The value of data written to TXxREG will be ignored and all '0's will be transmitted.

The SENDB bit is automatically reset by hardware after the corresponding Stop bit is sent. This allows the user to preload the transmit FIFO with the next transmit byte following the Break character (typically, the Sync character in the LIN specification).

The TRMT bit of the TXxSTA register indicates when the transmit operation is active or Idle, just as it does during normal transmission. See Figure 25-9 for the timing of the Break character sequence.

#### 25.4.4.1 Break and Sync Transmit Sequence

The following sequence will start a message frame header made up of a Break, followed by an auto-baud Sync byte. This sequence is typical of a LIN bus master.

1. Configure the EUSART for the desired mode.
2. Set the TXEN and SENDB bits to enable the Break sequence.
3. Load the TXxREG with a dummy character to initiate transmission (the value is ignored).
4. Write '55h' to TXxREG to load the Sync character into the transmit FIFO buffer.
5. After the Break has been sent, the SENDB bit is reset by hardware and the Sync character is then transmitted.

When the TXxREG becomes empty, as indicated by the TXxIF, the next data byte can be written to TXxREG.

### 25.4.5 RECEIVING A BREAK CHARACTER

The Enhanced EUSART module can receive a Break character in two ways.

The first method to detect a Break character uses the FERR bit of the RCxSTA register and the Received data as indicated by RCxREG. The Baud Rate Generator is assumed to have been initialized to the expected baud rate.

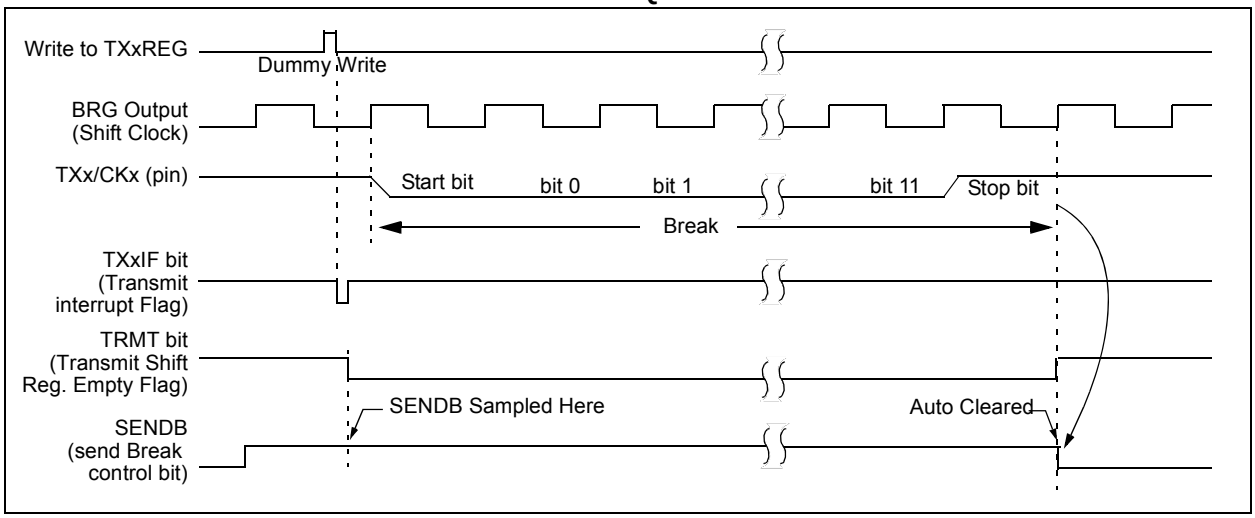
A Break character has been received when;

- RCxIF bit is set
- FERR bit is set
- RCxREG = 00h

The second method uses the Auto-Wake-up feature described in **Section 25.4.3 "Auto-Wake-up on Break"**. By enabling this feature, the EUSART will sample the next two transitions on RXx/DTx, cause an RCxIF interrupt, and receive the next data byte followed by another interrupt.

Note that following a Break character, the user will typically want to enable the Auto-Baud Detect feature. For both methods, the user can set the ABDEN bit of the BAUDxCON register before placing the EUSART in Sleep mode.

**FIGURE 25-9: SEND BREAK CHARACTER SEQUENCE**



## 25.5 EUSART Synchronous Mode

Synchronous serial communications are typically used in systems with a single master and one or more slaves. The master device contains the necessary circuitry for baud rate generation and supplies the clock for all devices in the system. Slave devices can take advantage of the master clock by eliminating the internal clock generation circuitry.

There are two signal lines in Synchronous mode: a bidirectional data line and a clock line. Slaves use the external clock supplied by the master to shift the serial data into and out of their respective receive and transmit shift registers. Since the data line is bidirectional, synchronous operation is half-duplex only. Half-duplex refers to the fact that master and slave devices can receive and transmit data but not both simultaneously. The EUSART can operate as either a master or slave device.

Start and Stop bits are not used in synchronous transmissions.

### 25.5.1 SYNCHRONOUS MASTER MODE

The following bits are used to configure the EUSART for Synchronous Master operation:

- SYNC = 1
- CSRC = 1
- SREN = 0 (for transmit); SREN = 1 (for receive)
- CREN = 0 (for transmit); CREN = 1 (for receive)
- SPEN = 1

Setting the SYNC bit of the TXxSTA register configures the device for synchronous operation. Setting the CSRC bit of the TXxSTA register configures the device as a master. Clearing the SREN and CREN bits of the RCxSTA register ensures that the device is in the Transmit mode, otherwise the device will be configured to receive. Setting the SPEN bit of the RCxSTA register enables the EUSART. If the RXx/DTx or TXx/CKx pins are shared with an analog peripheral the analog I/O functions must be disabled by clearing the corresponding ANSEL bits.

The TRIS bits corresponding to the RXx/DTx and TXx/CKx pins should be set.

#### 25.5.1.1 Master Clock

Synchronous data transfers use a separate clock line, which is synchronous with the data. A device configured as a master transmits the clock on the TXx/CKx line. The TXx/CKx pin output driver is automatically enabled when the EUSART is configured for synchronous transmit or receive operation. Serial data bits change on the leading edge to ensure they are valid at the trailing edge of each clock. One clock cycle is generated for each data bit. Only as many clock cycles are generated as there are data bits.

#### 25.5.1.2 Clock Polarity

A clock polarity option is provided for Microwire compatibility. Clock polarity is selected with the SCKP bit of the BAUDxCON register. Setting the SCKP bit sets the clock Idle state as high. When the SCKP bit is set, the data changes on the falling edge of each clock and is sampled on the rising edge of each clock. Clearing the SCKP bit sets the Idle state as low. When the SCKP bit is cleared, the data changes on the rising edge of each clock and is sampled on the falling edge of each clock.

#### 25.5.1.3 Synchronous Master Transmission

Data is transferred out of the device on the RXx/DTx pin. The RXx/DTx and TXx/CKx pin output drivers are automatically enabled when the EUSART is configured for synchronous master transmit operation.

A transmission is initiated by writing a character to the TXxREG register. If the TSR still contains all or part of a previous character the new character data is held in the TXxREG until the last bit of the previous character has been transmitted. If this is the first character, or the previous character has been completely flushed from the TSR, the data in the TXxREG is immediately transferred to the TSR. The transmission of the character commences immediately following the transfer of the data to the TSR from the TXxREG.

Each data bit changes on the leading edge of the master clock and remains valid until the subsequent leading clock edge.

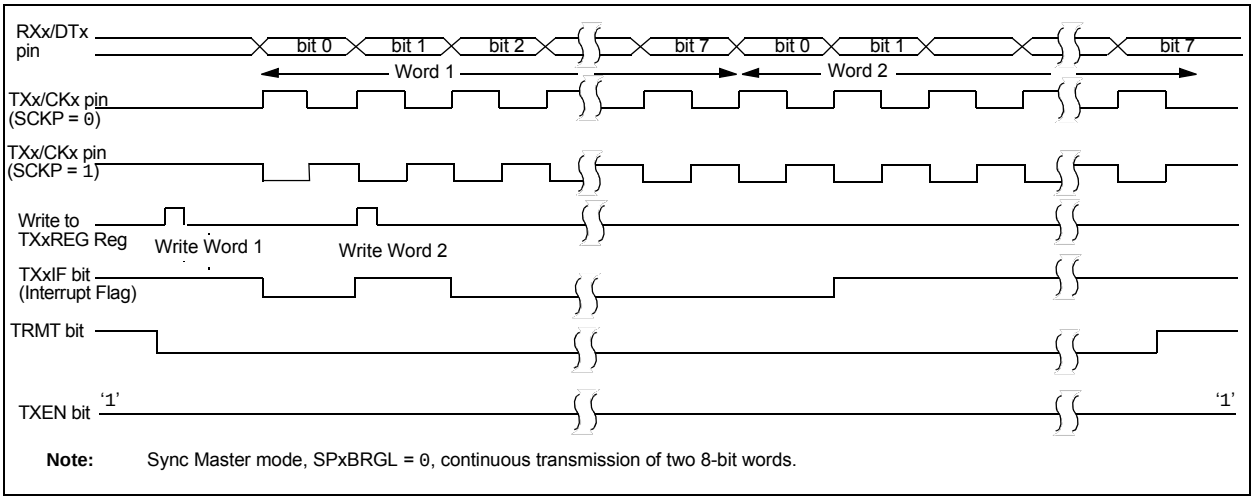
| **Note:** | The TSR register is not mapped in data memory, so it is not available to the user. |
|---|---|


© 2010-2016 Microchip Technology Inc.

25.5.1.4 Synchronous Master Transmission Set-up:

1. Initialize the SPxBRGH, SPxBRGL register pair and the BRGH and BRG16 bits to achieve the desired baud rate (see **Section 25.4 "EUSART Baud Rate Generator (BRG)"**).
2. Set the RXx/DTx and TXx/CKx TRIS controls to '1'.
3. Enable the synchronous master serial port by setting bits SYNC, SPEN and CSRC. Set the TRIS bits corresponding to the RXx/DTx and TXx/CKx I/O pins.
4. Disable Receive mode by clearing bits SREN and CREN.
5. Enable Transmit mode by setting the TXEN bit.
6. If 9-bit transmission is desired, set the TX9 bit.
7. If interrupts are desired, set the TXxIE, GIE and PEIE interrupt enable bits.
8. If 9-bit transmission is selected, the ninth bit should be loaded in the TX9D bit.
9. Start transmission by loading data to the TXxREG register.

**FIGURE 25-10: SYNCHRONOUS TRANSMISSION**




**Note:** Sync Master mode, SPxBRGL = 0, continuous transmission of two 8-bit words.

**FIGURE 25-11: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)**

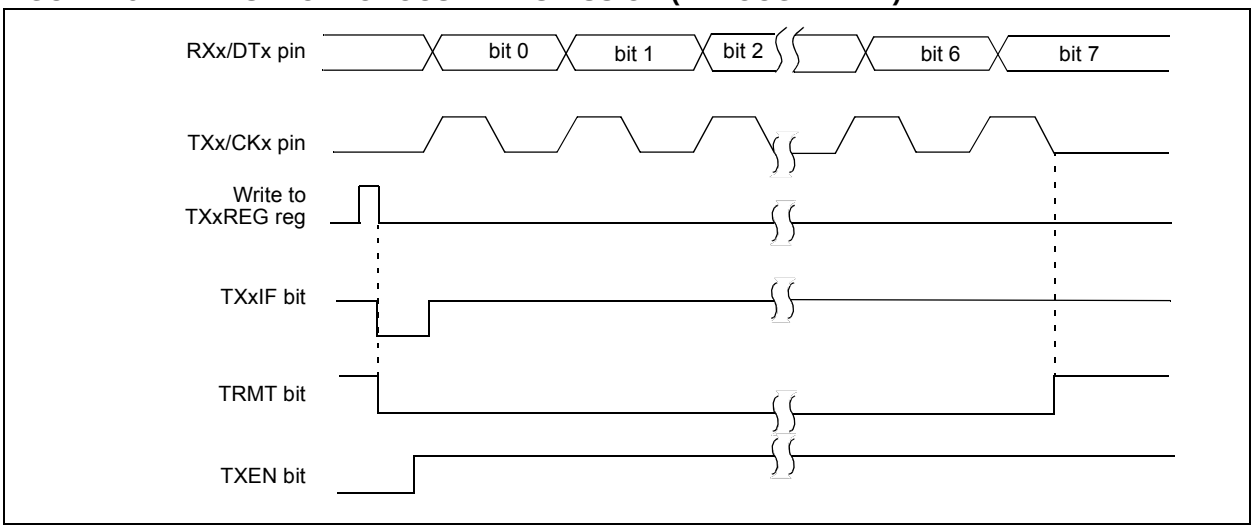


**TABLE 25-7: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER TRANSMISSION**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 299 |
| BAUD2CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 299 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| PIE1 | — | ADIE | RC1IE | TX1IE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 91 |
| PIE4 | — | — | RC2IE | TX2IE | — | — | BCL2IE | SSP2IE | 94 |
| PIR1 | — | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 95 |
| PIR4 | — | — | RC2IF | TX2IF | — | — | BCL2IF | SSP2IF | 98 |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 298 |
| RC2STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 298 |
| SP1BRGL | EUSART1 Baud Rate Generator, Low Byte | | | | | | | | 300* |
| SP1BRGH | EUSART1 Baud Rate Generator, High Byte | | | | | | | | 300* |
| SP2BRGL | EUSART2 Baud Rate Generator, Low Byte | | | | | | | | 300* |
| SP2BRGH | EUSART2 Baud Rate Generator, High Byte | | | | | | | | 300* |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 131 |
| TX1REG | EUSART1 Transmit Register | | | | | | | | 292* |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 297 |
| TX2REG | EUSART2 Transmit Register | | | | | | | | 292* |
| TX2STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 297 |

**Legend:** — = unimplemented locations, read as '0'. Shaded bits are not used for synchronous master transmission.

* Page provides register information.


© 2010-2016 Microchip Technology Inc.

#### 25.5.1.5 Synchronous Master Reception

Data is received at the RXx/DTx pin. The RXx/DTx pin output driver must be disabled by setting the corresponding TRIS bits when the EUSART is configured for synchronous master receive operation.

In Synchronous mode, reception is enabled by setting either the Single Receive Enable bit (SREN of the RCxSTA register) or the Continuous Receive Enable bit (CREN of the RCxSTA register).

When SREN is set and CREN is clear, only as many clock cycles are generated as there are data bits in a single character. The SREN bit is automatically cleared at the completion of one character. When CREN is set, clocks are continuously generated until CREN is cleared. If CREN is cleared in the middle of a character the CK clock stops immediately and the partial character is discarded. If SREN and CREN are both set, then SREN is cleared at the completion of the first character and CREN takes precedence.

To initiate reception, set either SREN or CREN. Data is sampled at the RXx/DTx pin on the trailing edge of the TXx/CKx clock pin and is shifted into the Receive Shift Register (RSR). When a complete character is received into the RSR, the RCxIF bit is set and the character is automatically transferred to the two character receive FIFO. The Least Significant eight bits of the top character in the receive FIFO are available in RCxREG. The RCxIF bit remains set as long as there are un-read characters in the receive FIFO.

#### 25.5.1.6 Slave Clock

Synchronous data transfers use a separate clock line, which is synchronous with the data. A device configured as a slave receives the clock on the TXx/CKx line. The TXx/CKx pin output driver must be disabled by setting the associated TRIS bit when the device is configured for synchronous slave transmit or receive operation. Serial data bits change on the leading edge to ensure they are valid at the trailing edge of each clock. One data bit is transferred for each clock cycle. Only as many clock cycles should be received as there are data bits.

#### 25.5.1.7 Receive Overrun Error

The receive FIFO buffer can hold two characters. An overrun error will be generated if a third character, in its entirety, is received before RCxREG is read to access the FIFO. When this happens the OERR bit of the RCxSTA register is set. Previous data in the FIFO will not be overwritten. The two characters in the FIFO buffer can be read, however, no additional characters will be received until the error is cleared. The OERR bit can only be cleared by clearing the overrun condition. If the overrun error occurred when the SREN bit is set and CREN is clear then the error is cleared by reading RCxREG.

If the overrun occurred when the CREN bit is set then the error condition is cleared by either clearing the CREN bit of the RCxSTA register or by clearing the SPEN bit which resets the EUSART.

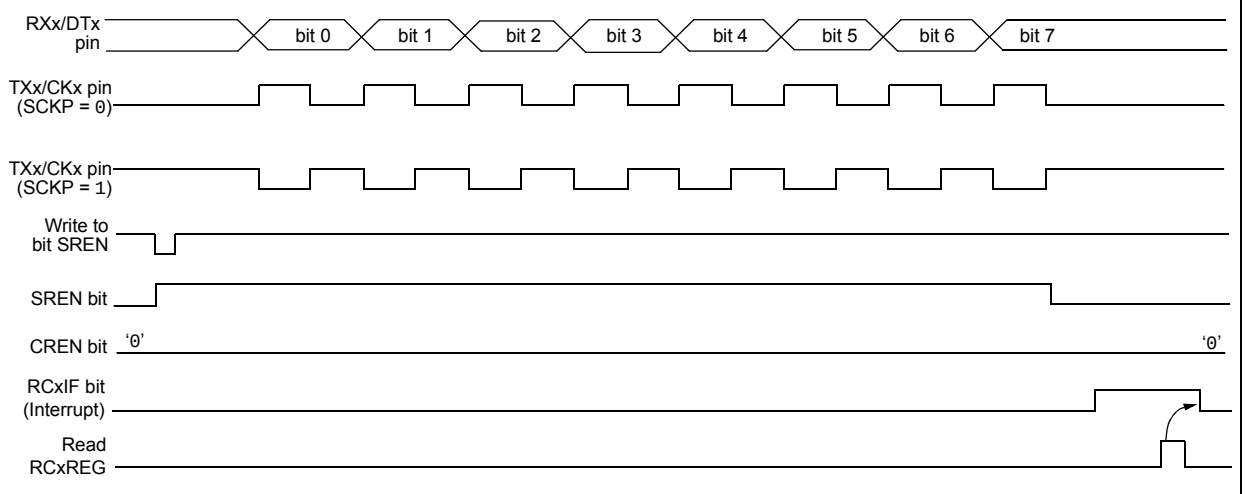
#### 25.5.1.8 Receiving 9-bit Characters

The EUSART supports 9-bit character reception. When the RX9 bit of the RCxSTA register is set the EUSART will shift 9-bits into the RSR for each character received. The RX9D bit of the RCxSTA register is the ninth, and Most Significant, data bit of the top unread character in the receive FIFO. When reading 9-bit data from the receive FIFO buffer, the RX9D data bit must be read before reading the eight Least Significant bits from the RCxREG.

#### 25.5.1.9 Synchronous Master Reception Setup:

1. Initialize the SPxBRGH, SPxBRGL register pair for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Set the RXx/DTx and TXx/CKx TRIS controls to '1'.
3. Enable the synchronous master serial port by setting bits SYNC, SPEN and CSRC. Disable RXx/DTx and TXx/CKx output drivers by setting the corresponding TRIS bits.
4. Ensure bits CREN and SREN are clear.
5. If using interrupts, set the GIE and PEIE bits of the INTCON register and set RCxIE.
6. If 9-bit reception is desired, set bit RX9.
7. Start reception by setting the SREN bit or for continuous reception, set the CREN bit.
8. Interrupt flag bit RCxIF will be set when reception of a character is complete. An interrupt will be generated if the enable bit RCxIE was set.
9. Read the RCxSTA register to get the ninth bit (if enabled) and determine if any error occurred during reception.
10. Read the 8-bit received data by reading the RCxREG register.
11. If an overrun error occurs, clear the error by either clearing the CREN bit of the RCxSTA register or by clearing the SPEN bit which resets the EUSART.

**FIGURE 25-12: SYNCHRONOUS RECEPTION (MASTER MODE, SREN)**




**Note:** Timing diagram demonstrates Sync Master mode with bit SREN = 1 and bit BRGH = 0.

**TABLE 25-8: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER RECEPTION**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 299 |
| BAUD2CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 299 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| PIE1 | — | ADIE | RC1IE | TX1IE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 91 |
| PIE4 | — | — | RC2IE | TX2IE | — | — | BCL2IE | SSP2IE | 94 |
| PIR1 | — | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 95 |
| PIR4 | — | — | RC2IF | TX2IF | — | — | BCL2IF | SSP2IF | 98 |
| RC1REG | EUSART1 Receive Register | | | | | | | | 292* |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 298 |
| RC2REG | EUSART2 Receive Register | | | | | | | | 292* |
| RC2STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 298 |
| SP1BRGL | EUSART1 Baud Rate Generator, Low Byte | | | | | | | | 300* |
| SP1BRGH | EUSART1 Baud Rate Generator, High Byte | | | | | | | | 300* |
| SP2BRGL | EUSART2 Baud Rate Generator, Low Byte | | | | | | | | 300* |
| SP2BRGH | EUSART2 Baud Rate Generator, High Byte | | | | | | | | 300* |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 297 |
| TX2STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 297 |

**Legend:** — = unimplemented locations, read as '0'. Shaded bits are not used for synchronous master reception.

* Page provides register information.


© 2010-2016 Microchip Technology Inc.

### 25.5.2 SYNCHRONOUS SLAVE MODE

The following bits are used to configure the EUSART for Synchronous slave operation:

- SYNC = 1
- CSRC = 0
- SREN = 0 (for transmit); SREN = 1 (for receive)
- CREN = 0 (for transmit); CREN = 1 (for receive)
- SPEN = 1

Setting the SYNC bit of the TXxSTA register configures the device for synchronous operation. Clearing the CSRC bit of the TXxSTA register configures the device as a slave. Clearing the SREN and CREN bits of the RCxSTA register ensures that the device is in the Transmit mode, otherwise the device will be configured to receive. Setting the SPEN bit of the RCxSTA register enables the EUSART. If the RXx/DTx or TXx/CKx pins are shared with an analog peripheral the analog I/O functions must be disabled by clearing the corresponding ANSEL bits.

RXx/DTx and TXx/CKx pin output drivers must be disabled by setting the corresponding TRIS bits.

#### 25.5.2.1 EUSART Synchronous Slave Transmit

The operation of the Synchronous Master and Slave modes are identical (see **Section 25.5.1.3 "Synchronous Master Transmission"**), except in the case of the Sleep mode.

If two words are written to the TXxREG and then the SLEEP instruction is executed, the following will occur:

1. The first character will immediately transfer to the TSR register and transmit.
2. The second word will remain in TXxREG register.
3. The TXxIF bit will not be set.
4. After the first character has been shifted out of TSR, the TXxREG register will transfer the second character to the TSR and the TXxIF bit will now be set.
5. If the PEIE and TXxIE bits are set, the interrupt will wake the device from Sleep and execute the next instruction. If the GIE bit is also set, the program will call the Interrupt Service Routine.

#### 25.5.2.2 Synchronous Slave Transmission Set-up:

1. Set the SYNC and SPEN bits and clear the CSRC bit.
2. Set the RXx/DTx and TXx/CKx TRIS controls to '1'.
3. Clear the CREN and SREN bits.
4. If using interrupts, ensure that the GIE and PEIE bits of the INTCON register are set and set the TXxIE bit.
5. If 9-bit transmission is desired, set the TX9 bit.
6. Enable transmission by setting the TXEN bit.
7. If 9-bit transmission is selected, insert the Most Significant bit into the TX9D bit.
8. Start transmission by writing the Least Significant eight bits to the TXxREG register.

**TABLE 25-9: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE TRANSMISSION**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 299 |
| BAUD2CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 299 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| PIE1 | — | ADIE | RC1IE | TX1IE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 91 |
| PIE4 | — | — | RC2IE | TX2IE | — | — | BCL2IE | SSP2IE | 94 |
| PIR1 | — | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 95 |
| PIR4 | — | — | RC2IF | TX2IF | — | — | BCL2IF | SSP2IF | 98 |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 298 |
| RC2STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 298 |
| SP1BRGL | EUSART1 Baud Rate Generator, Low Byte | | | | | | | | 300* |
| SP1BRGH | EUSART1 Baud Rate Generator, High Byte | | | | | | | | 300* |
| SP2BRGL | EUSART2 Baud Rate Generator, Low Byte | | | | | | | | 300* |
| SP2BRGH | EUSART2 Baud Rate Generator, High Byte | | | | | | | | 300* |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 131 |
| TX1REG | EUSART1 Transmit Register | | | | | | | | 292* |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 297 |
| TX2REG | EUSART2 Transmit Register | | | | | | | | 292* |
| TX2STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 297 |

**Legend:** — = unimplemented locations, read as '0'. Shaded bits are not used for synchronous slave transmission.

* Page provides register information.


© 2010-2016 Microchip Technology Inc.

#### 25.5.2.3 EUSART Synchronous Slave Reception

The operation of the Synchronous Master and Slave modes is identical (**Section 25.5.1.5 "Synchronous Master Reception"**), with the following exceptions:

- Sleep
- CREN bit is always set, therefore the receiver is never Idle
- SREN bit, which is a "don't care" in Slave mode

A character may be received while in Sleep mode by setting the CREN bit prior to entering Sleep. Once the word is received, the RSR register will transfer the data to the RCxREG register. If the RCxIE enable bit is set, the interrupt generated will wake the device from Sleep and execute the next instruction. If the GIE bit is also set, the program will branch to the interrupt vector.

#### 25.5.2.4 Synchronous Slave Reception Setup:

1. Set the SYNC and SPEN bits and clear the CSRC bit.
2. Set the RXx/DTx and TXx/CKx TRIS controls to '1'.
3. If using interrupts, ensure that the GIE and PEIE bits of the INTCON register are set and set the RCxIE bit.
4. If 9-bit reception is desired, set the RX9 bit.
5. Set the CREN bit to enable reception.
6. The RCxIF bit will be set when reception is complete. An interrupt will be generated if the RCxIE bit was set.
7. If 9-bit mode is enabled, retrieve the Most Significant bit from the RX9D bit of the RCxSTA register.
8. Retrieve the eight Least Significant bits from the receive FIFO by reading the RCxREG register.
9. If an overrun error occurs, clear the error by either clearing the CREN bit of the RCxSTA register or by clearing the SPEN bit which resets the EUSART.

**TABLE 25-10: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE RECEPTION**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 299 |
| BAUD2CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 299 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| PIE1 | — | ADIE | RC1IE | TX1IE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 91 |
| PIE4 | — | — | RC2IE | TX2IE | — | — | BCL2IE | SSP2IE | 94 |
| PIR1 | — | ADIF | RC1IF | TX1IF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 95 |
| PIR4 | — | — | RC2IF | TX2IF | — | — | BCL2IF | SSP2IF | 98 |
| RC1REG | EUSART1 Receive Register | | | | | | | | 292* |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 298 |
| RC2REG | EUSART2 Receive Register | | | | | | | | 292* |
| RC2STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 298 |
| SP1BRGL | EUSART1 Baud Rate Generator, Low Byte | | | | | | | | 300* |
| SP1BRGH | EUSART1 Baud Rate Generator, High Byte | | | | | | | | 300* |
| SP2BRGL | EUSART2 Baud Rate Generator, Low Byte | | | | | | | | 300* |
| SP2BRGH | EUSART2 Baud Rate Generator, High Byte | | | | | | | | 300* |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 297 |
| TX2STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 297 |

**Legend:** — = unimplemented locations, read as '0'. Shaded bits are not used for synchronous slave reception.

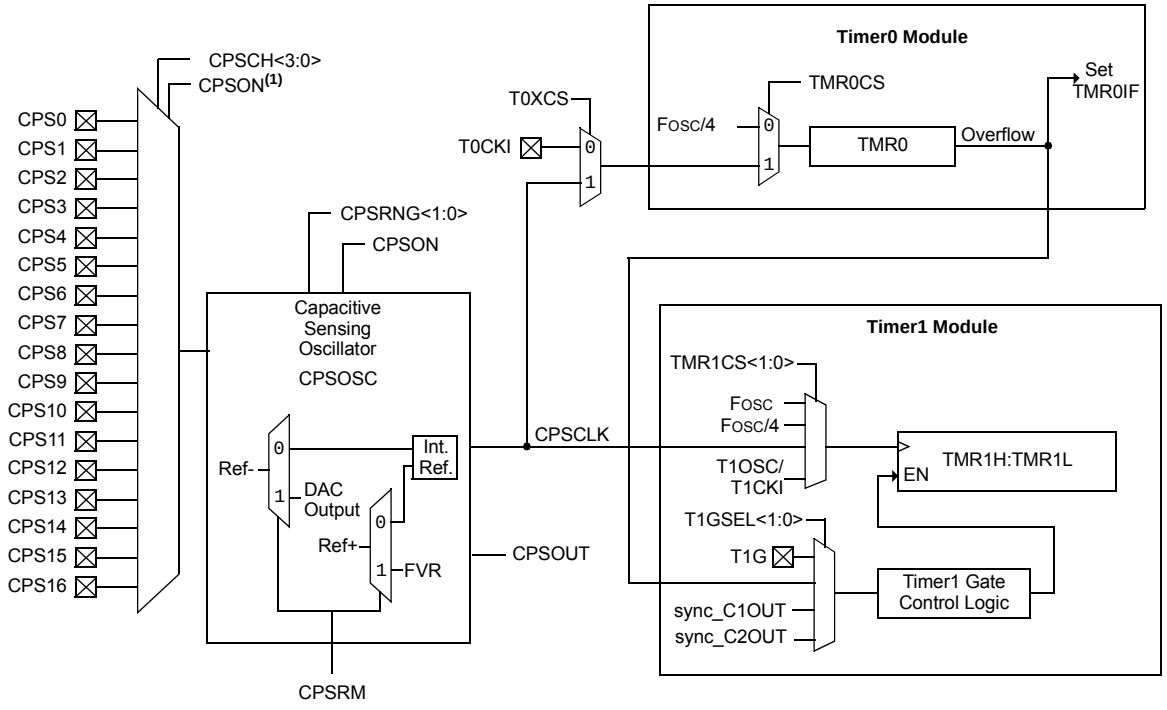
* Page provides register information.

## 26.0 CAPACITIVE SENSING (CPS) MODULE

The Capacitive Sensing (CPS) module allows for an interaction with an end user without a mechanical interface. In a typical application, the CPS module is attached to a pad on a Printed Circuit Board (PCB), which is electrically isolated from the end user. When the end user places their finger over the PCB pad, a capacitive load is added, causing a frequency shift in the CPS module. The CPS module requires software and at least one timer resource to determine the change in frequency. Key features of this module include:

- Analog MUX for monitoring multiple inputs
- Capacitive sensing oscillator
- Multiple power modes
- Multiple current ranges
- Multiple voltage reference modes
- Software control
- Operation during Sleep

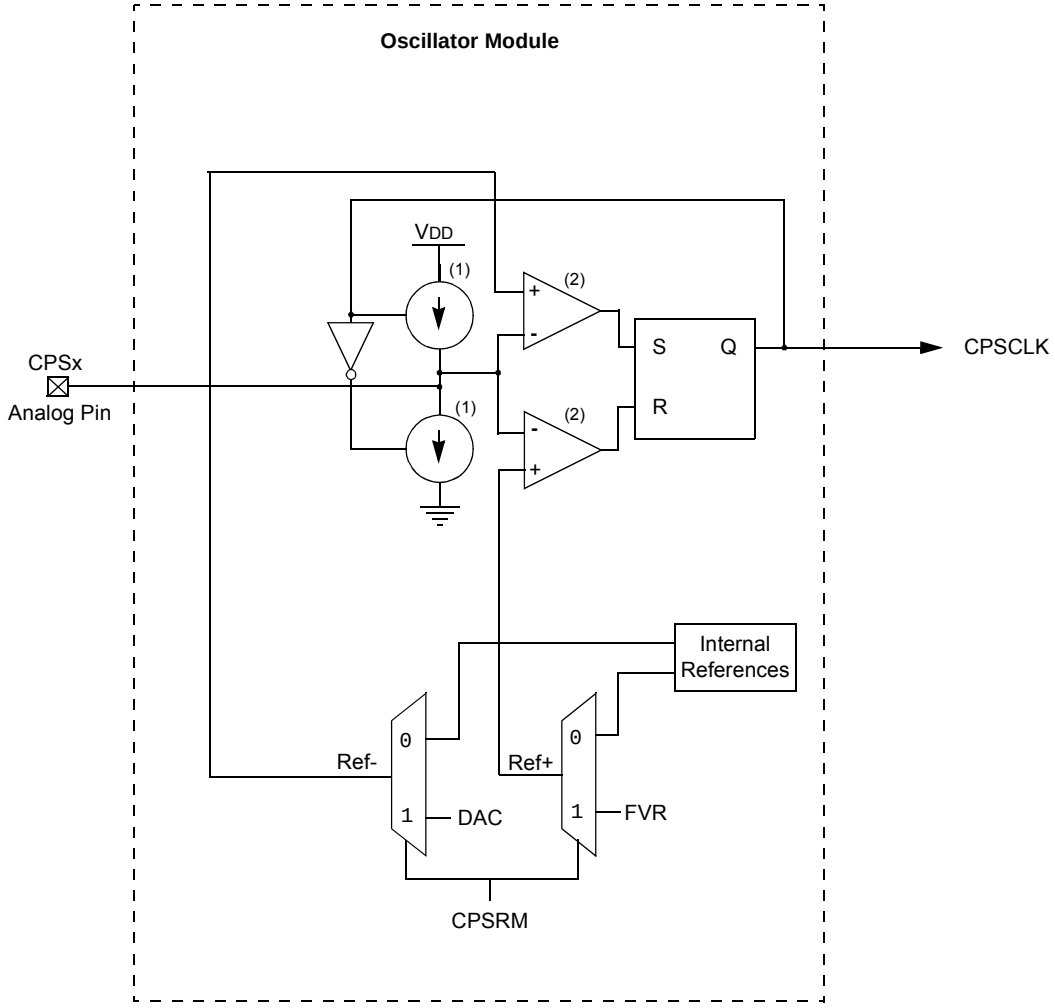
**FIGURE 26-1: CAPACITIVE SENSING BLOCK DIAGRAM**




**Note 1:** If CPSON = 0, disabling capacitive sensing, no channel is selected.


© 2010-2016 Microchip Technology Inc.

**FIGURE 26-2: CAPACITIVE SENSING OSCILLATOR BLOCK DIAGRAM**




**Note 1:** Module Enable and Power mode selections are not shown.
**2:** Comparators remain active in Noise Detection mode.

## 26.1 Analog MUX

The CPS module can monitor up to 16 inputs. The capacitive sensing inputs are defined as CPS<15:0>. To determine if a frequency change has occurred, the user must:

- Select the appropriate CPS pin by setting the appropriate CPSCH bits of the CPSCON1 register.
- Set the corresponding ANSEL bit.
- Set the corresponding TRIS bit.
- Run the software algorithm.

Selection of the CPSx pin while the module is enabled will cause the capacitive sensing oscillator to be on the CPSx pin. Failure to set the corresponding ANSEL and TRIS bits can cause the capacitive sensing oscillator to stop, leading to false frequency readings.

## 26.2 Capacitive Sensing Oscillator

The capacitive sensing oscillator consists of a constant current source and a constant current sink, to produce a triangle waveform. The CPSOUT bit of the CPSCON0 register shows the status of the capacitive sensing oscillator, whether it is a sinking or sourcing current. The oscillator is designed to drive a capacitive load (single PCB pad) and at the same time, be a clock source to either Timer0 or Timer1. The oscillator has three different current settings as defined by CPSRNG<1:0> of the CPSCON0 register. The different current settings for the oscillator serve two purposes:

- Maximize the number of counts in a timer for a fixed time base.
- Maximize the count differential in the timer during a change in frequency.

### 26.2.1 VOLTAGE REFERENCE MODES

The capacitive sensing oscillator uses voltage references to provide two voltage thresholds for oscillation. The upper voltage threshold is referred to as Ref+ and the lower voltage threshold is referred to as Ref-.

The user can elect to use fixed voltage references, which are internal to the capacitive sensing oscillator, or variable voltage references, which are supplied by the Fixed Voltage Reference (FVR) module and the Digital-to-Analog Converter (DAC) module.

When the fixed voltage references are used, the $V_{SS}$ voltage determines the lower threshold level (Ref-) and the $V_{DD}$ voltage determines the upper threshold level (Ref+).

When the variable voltage references are used, the DAC voltage determines the lower threshold level (Ref-) and the FVR voltage determines the upper threshold level (Ref+). An advantage of using these reference sources is that oscillation frequency remains constant with changes in $V_{DD}$.

Different oscillation frequencies can be obtained through the use of these variable voltage references. The more the upper voltage reference level is lowered and the more the lower voltage reference level is raised, the higher the capacitive sensing oscillator frequency becomes.

Selection between the voltage references is controlled by the CPSRM bit of the CPSCON0 register. Setting this bit selects the variable voltage references and clearing this bit selects the fixed voltage references.

See **Section 14.0 “Fixed Voltage Reference (FVR)”** and **Section 17.0 “Digital-to-Analog Converter (DAC) Module”** for more information on configuring the variable voltage levels.


© 2010-2016 Microchip Technology Inc.

### 26.2.2 CURRENT RANGES

The capacitive sensing oscillator can operate in one of seven different power modes. The power modes are separated into two ranges: the low range and the high range.

When the oscillator's low range is selected, the fixed internal voltage references of the capacitive sensing oscillator are being used. When the oscillator's high range is selected, the variable voltage references supplied by the FVR and DAC modules are being used. Selection between the voltage references is controlled by the CPSRM bit of the CPSCON0 register. See **Section 26.2.1 "Voltage Reference Modes"** for more information.

Within each range there are three distinct power modes: low, medium and high. Current consumption is dependent upon the range and mode selected. Selecting Power modes within each range is accomplished by configuring the CPSRNG <1:0> bits in the CPSCON0 register. See Table for proper power mode selection.

The remaining mode is a Noise Detection mode that resides within the high range. The Noise Detection mode is unique in that it disables the sinking and sourcing of current on the analog pin but leaves the rest of the oscillator circuitry active. This reduces the oscillation frequency on the analog pin to zero and also greatly reduces the current consumed by the oscillator module.

When noise is introduced onto the pin, the oscillator is driven at the frequency determined by the noise. This produces a detectable signal at the comparator output, indicating the presence of activity on the pin.

Figure 26-2 shows a more detailed drawing of the current sources and comparators associated with the oscillator.

**TABLE 26-1: POWER MODE SELECTION**

| CPSRM | Range | CPSRNG<1:0> | Current Range(1) |
|---|---|---|---|
| 1 | High | 00 | Noise Detection |
| | | 01 | Low |
| | | 10 | Medium |
| | | 11 | High |
| 0 | Low | 00 | Off |
| | | 01 | Low |
| | | 10 | Medium |
| | | 11 | High |

**Note 1:** See Power-Down Currents (IPD) in **Section 30.0 "Electrical Specifications"** for more information.

### 26.2.3 TIMER RESOURCES

To measure the change in frequency of the capacitive sensing oscillator, a fixed time base is required. For the period of the fixed time base, the capacitive sensing oscillator is used to clock either Timer0 or Timer1. The frequency of the capacitive sensing oscillator is equal to the number of counts in the timer divided by the period of the fixed time base.

### 26.2.4 FIXED TIME BASE

To measure the frequency of the capacitive sensing oscillator, a fixed time base is required. Any timer resource or software loop can be used to establish the fixed time base. It is up to the end user to determine the method in which the fixed time base is generated.

| | |
|---|---|
| **Note:** | The fixed time base can not be generated by the timer resource that the capacitive sensing oscillator is clocking. |

#### 26.2.4.1 Timer0

To select Timer0 as the timer resource for the CPS module:

- Set the T0XCS bit of the CPSCON0 register.
- Clear the TMR0CS bit of the OPTION_REG register.

When Timer0 is chosen as the timer resource, the capacitive sensing oscillator will be the clock source for Timer0. Refer to **Section 20.0 "Timer0 Module"** for additional information.

#### 26.2.4.2 Timer1

To select Timer1 as the timer resource for the CPS module, set the TMR1CS<1:0> of the T1CON register to '11'. When Timer1 is chosen as the timer resource, the capacitive sensing oscillator will be the clock source for Timer1. Because the Timer1 module has a gate control, developing a time base for the frequency measurement can be simplified by using the Timer0 overflow flag.

It is recommended that the Timer0 overflow flag, in conjunction with the Toggle mode of the Timer1 Gate, be used to develop the fixed time base required by the software portion of the CPS module. Refer to **Section 21.11 "Register Definitions: Timer1 Control"** for additional information.

**TABLE 26-3: TIMER1 ENABLE FUNCTION**

| TMR1ON | TMR1GE | Timer1 Operation |
|---|---|---|
| 0 | 0 | Off |
| 0 | 1 | Off |
| 1 | 0 | On |
| 1 | 1 | Count Enabled by input |

### 26.2.5 SOFTWARE CONTROL

The software portion of the CPS module is required to determine the change in frequency of the capacitive sensing oscillator. This is accomplished by the following:

- Setting a fixed time base to acquire counts on Timer0 or Timer1.
- Establishing the nominal frequency for the capacitive sensing oscillator.
- Establishing the reduced frequency for the capacitive sensing oscillator due to an additional capacitive load.
- Set the frequency threshold.

#### 26.2.5.1 Nominal Frequency (No Capacitive Load)

To determine the nominal frequency of the capacitive sensing oscillator:

- Remove any extra capacitive load on the selected CPSx pin.
- At the start of the fixed time base, clear the timer resource.
- At the end of the fixed time base save the value in the timer resource.

The value of the timer resource is the number of oscillations of the capacitive sensing oscillator for the given time base. The frequency of the capacitive sensing oscillator is equal to the number of counts on in the timer divided by the period of the fixed time base.

#### 26.2.5.2 Reduced Frequency (additional capacitive load)

The extra capacitive load will cause the frequency of the capacitive sensing oscillator to decrease. To determine the reduced frequency of the capacitive sensing oscillator:

- Add a typical capacitive load on the selected CPSx pin.
- Use the same fixed time base as the nominal frequency measurement.
- At the start of the fixed time base, clear the timer resource.
- At the end of the fixed time base save the value in the timer resource.

The value of the timer resource is the number of oscillations of the capacitive sensing oscillator, with an additional capacitive load. The frequency of the capacitive sensing oscillator is equal to the number of counts in the timer divided by the period of the fixed time base. This frequency should be lower than the value obtained during the nominal frequency measurement.

#### 26.2.5.3 Frequency Threshold

The frequency threshold should be placed midway between the value of the nominal frequency and the reduced frequency of the capacitive sensing oscillator. Refer to Application Note *AN1103*, *Software Handling for Capacitive Sensing* (DS01103) for more detailed information on the software required for CPS module.

**Note:** For more information on general capacitive sensing refer to Application Notes:

- *AN1101*, *Introduction to Capacitive Sensing* (DS01101)
- *AN1102*, *Layout and Physical Design Guidelines for Capacitive Sensing* (DS01102)


© 2010-2016 Microchip Technology Inc.

## 26.3 Operation during Sleep

The capacitive sensing oscillator will continue to run as long as the module is enabled, independent of the part being in Sleep. In order for the software to determine if a frequency change has occurred, the part must be awake. However, the part does not have to be awake when the timer resource is acquiring counts.

| | |
|---|---|
| **Note:** | Timer0 does not operate in Sleep, and therefore cannot be used for capacitive sense measurements in Sleep. |

## 26.4 Register Definitions: Capacitive Sensing Control

**REGISTER 26-2: CPSCON0: CAPACITIVE SENSING CONTROL REGISTER 0**

| R/W-0/0 | R/W-0/0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| CPSON | CPSRM | — | — | CPSRNG<1:0> | | CPSOUT | T0XCS |
| 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 **CPSON:** CPS Module Enable bit
1 = CPS module is enabled
0 = CPS module is disabled

bit 6 **CPSRM:** Capacitive Sensing Reference Mode bit
1 = CPS module is in high range. DAC and FVR provide oscillator voltage references.
0 = CPS module is in the low range. Internal oscillator voltage references are used.

bit 5-4 **Unimplemented:** Read as ‘0’

bit 3-2 **CPSRNG<1:0>:** Capacitive Sensing Current Range bits
If CPSRM = 0 (low range):
11 = Oscillator is in High Range. Charge/Discharge Current is nominally 18 µA
10 = Oscillator is in Medium Range. Charge/Discharge Current is nominally 1.2 µA
01 = Oscillator is in Low Range. Charge/Discharge Current is nominally 0.1 µA
00 = Oscillator is off

If CPSRM = 1 (high range):
11 = Oscillator is in High Range. Charge/Discharge Current is nominally 100 µA
10 = Oscillator is in Medium Range. Charge/Discharge Current is nominally 30 µA
01 = Oscillator is in Low Range. Charge/Discharge Current is nominally 9 µA
00 = Oscillator is on. Noise Detection mode. No Charge/Discharge current is supplied.

bit 1 **CPSOUT:** Capacitive Sensing Oscillator Status bit
1 = Oscillator is sourcing current (Current flowing out of the pin)
0 = Oscillator is sinking current (Current flowing into the pin)

bit 0 **T0XCS:** Timer0 External Clock Source Select bit
If TMR0CS = 1:
The T0XCS bit controls which clock external to the core/Timer0 module supplies Timer0:
1 = Timer0 clock source is the capacitive sensing oscillator
0 = Timer0 clock source is the T0CKI pin
If TMR0CS = 0:
Timer0 clock source is controlled by the core/Timer0 module and is FOSC/4


© 2010-2016 Microchip Technology Inc.

**REGISTER 26-3: CPSCON1: CAPACITIVE SENSING CONTROL REGISTER 1**

| U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | CPSCH<4:0> | | | | |
| 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 **Unimplemented:** Read as '0'

bit 4-0 **CPSCH<4:0>:** Capacitive Sensing Channel Select bits

If CPSON = 0:

These bits are ignored. No channel is selected.

If CPSON = 1:

00000 = channel 0, (CPS0)
00001 = channel 1, (CPS1)
00010 = channel 2, (CPS2)
00011 = channel 3, (CPS3)
00100 = channel 4, (CPS4)
00101 = channel 5, (CPS5)
00110 = channel 6, (CPS6)
00111 = channel 7, (CPS7)
01000 = channel 8, (CPS8)
01001 = channel 9, (CPS9)
01010 = channel 10, (CPS10)
01011 = channel 11, (CPS11)
01100 = channel 12, (CPS12)
01101 = channel 13, (CPS13)
01110 = channel 14, (CPS14)
01111 = channel 15, (CPS15)
10000 = channel 16, (CPS16)
10001 = Reserved. Do not use.
.
.
.
11111 = Reserved. Do not use.

**TABLE 26-1: SUMMARY OF REGISTERS ASSOCIATED WITH CAPACITIVE SENSING**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| ANSELA | — | — | ANSA5 | ANSA4 | ANSA3 | ANSA2 | ANSA1 | ANSA0 | 126 |
| CPSCON0 | CPSON | CPSRM | — | — | CPSRNG<1:0> | | CPSOUT | T0XCS | 322 |
| CPSCON1 | — | — | — | CPSCH<4:0> | | | | | 323 |
| OPTION_REG | $\overline{\text{WPUEN}}$ | INTEDG | TMR0CS | TMR0SE | PSA | PS2 | PS1 | PS0 | 188 |
| T1CON | TMR1CS<1:0> | | T1CKPS<1:0> | | T1OSCEN | $\overline{\text{T1SYNC}}$ | — | TMR1ON | 197 |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 125 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 128 |
| TRISD | TRISD<7:0> | | | | | | | | 134 |

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

## 27.0 LIQUID CRYSTAL DISPLAY (LCD) DRIVER MODULE

The Liquid Crystal Display (LCD) driver module generates the timing control to drive a static or multiplexed LCD panel. In the PIC16(L)F1946/47 device, the module drives the panels of up to four commons and up to 46 segments. The LCD module also provides control of the LCD pixel data.

The LCD driver module supports:

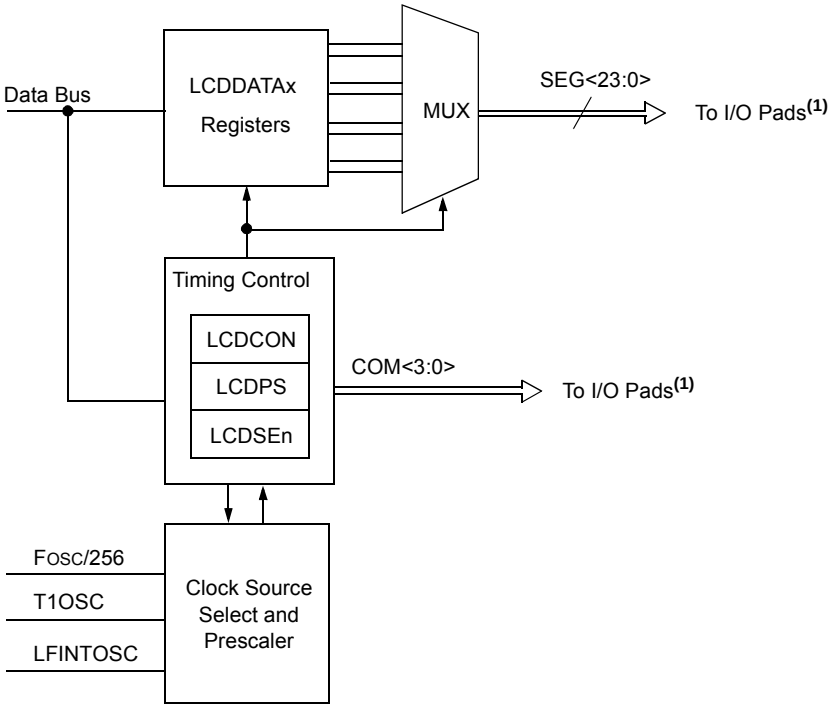
- Direct driving of LCD panel
- Three LCD clock sources with selectable prescaler
- Up to four common pins:
  - Static (one common)
  - 1/2 multiplex (two commons)
  - 1/3 multiplex (three commons)
  - 1/4 multiplex (four commons)
- Segment pins up to:
  - 64 (PIC16(L)F1946/47)
- Static, 1/2 or 1/3 LCD Bias

### 27.1 LCD Registers

The module contains the following registers:

- LCD Control register (LCDCON)
- LCD Phase register (LCDPS)
- LCD Reference Ladder register (LCDRL)
- LCD Contrast Control register (LCDCST)
- LCD Reference Voltage Control register (LCDREF)
- Up to six LCD Segment Enable registers (LCDSEn)
- Up to 24 LCD data registers (LCDDATAn)

**FIGURE 27-1: LCD DRIVER MODULE BLOCK DIAGRAM**




**Note 1:** These are not directly connected to the I/O pads, but may be tri-stated, depending on the configuration of the LCD module.


© 2010-2016 Microchip Technology Inc.

**TABLE 27-1: LCD SEGMENT AND DATA REGISTERS**

| Device | # of LCD Registers | |
|---|---|---|
| | Segment Enable | Data |
| PIC16(L)F1946/47 | 6 | 24 |

The LCDCON register (Register 27-1) controls the operation of the LCD driver module. The LCDPS register (Register 27-2) configures the LCD clock source prescaler and the type of waveform; Type-A or Type-B. The LCDSEn registers (Register 27-5) configure the functions of the port pins.

The following LCDSEn registers are available:

- LCDSE0 SE<7:0>
- LCDSE1 SE<15:8>
- LCDSE2 SE<23:16>(1)
- LCDSE3 SE<31:24>
- LCDSE4 SE<39:32>
- LCDSE5 SE<45:40>

Once the module is initialized for the LCD panel, the individual bits of the LCDDATAn registers are cleared/set to represent a clear/dark pixel, respectively:

- LCDDATA0 SEG<7:0>COM0
- LCDDATA1 SEG<15:8>COM0
- LCDDATA2 SEG<23:16>COM0
- LCDDATA3 SEG<7:0>COM1
- LCDDATA4 SEG<15:8>COM1
- LCDDATA5 SEG<23:16>COM1
- LCDDATA6 SEG<7:0>COM2
- LCDDATA7 SEG<15:8>COM2
- LCDDATA8 SEG<23:16>COM2
- LCDDATA9 SEG<7:0>COM3
- LCDDATA10 SEG<15:8>COM3
- LCDDATA11 SEG<23:16>COM3
- LCDDATA12 SEG<31:24>COM0
- LCDDATA13 SEG<39:32>COM0
- LCDDATA14 SEG<45:40>COM0
- LCDDATA15 SEG<31:24>COM1
- LCDDATA16 SEG<39:32>COM1
- LCDDATA17 SEG<45:40>COM1
- LCDDATA18 SEG<31:24>COM2
- LCDDATA19 SEG<39:32>COM2
- LCDDATA20 SEG<45:40>COM2
- LCDDATA21 SEG<31:24>COM3
- LCDDATA22 SEG<39:32>COM3
- LCDDATA23 SEG<45:40>COM3

As an example, LCDDATAn is detailed in Register 27-6.

Once the module is configured, the LCDEN bit of the LCDCON register is used to enable or disable the LCD module. The LCD panel can also operate during Sleep by clearing the SLPEN bit of the LCDCON register.

## 27.2 Register Definitions: Liquid Crystal Display (LCD) Control

**REGISTER 27-1: LCDCON: LIQUID CRYSTAL DISPLAY (LCD) CONTROL REGISTER**

| R/W-0/0 | R/W-0/0 | R/C-0/0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-1/1 | R/W-1/1 |
|---|---|---|---|---|---|---|---|
| LCDEN | SLPEN | WERR | — | CS<1:0> | | LMUX<1:0> | |
| bit 7 | | | | | | | bit 0 |

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

bit 7 **LCDEN:** LCD Driver Enable bit
1 = LCD driver module is enabled
0 = LCD driver module is disabled

bit 6 **SLPEN:** LCD Driver Enable in Sleep Mode bit
1 = LCD driver module is disabled in Sleep mode
0 = LCD driver module is enabled in Sleep mode

bit 5 **WERR:** LCD Write Failed Error bit
1 = LCDDATAn register written while the WA bit of the LCDPS register = 0 (must be cleared in software)
0 = No LCD write error

bit 4 **Unimplemented:** Read as ‘0’

bit 3-2 **CS<1:0>:** Clock Source Select bits
00 = FOSC/256
01 = T1OSC (Timer1)
1x = LFINTOSC (31 kHz)

bit 1-0 **LMUX<1:0>:** Commons Select bits

| LMUX<1:0> | Multiplex | Maximum Number of Pixels | Bias |
|---|---|---|---|
| | | PIC16F1946/47/ PIC16LF1946/47 | |
| 00 | Static (COM0) | 46 | Static |
| 01 | 1/2 (COM<1:0>) | 92 | 1/2 or 1/3 |
| 10 | 1/3 (COM<2:0>) | 138 | 1/2 or 1/3 |
| 11 | 1/4 (COM<3:0>) | 184 | 1/3 |


© 2010-2016 Microchip Technology Inc.

**REGISTER 27-2: LCDPS: LCD PHASE REGISTER**

| R/W-0/0 | R/W-0/0 | R-0/0 | R-0/0 | R/W-0/0 | R/W-0/0 | R/W-1/1 | R/W-1/1 |
|---|---|---|---|---|---|---|---|
| WFT | BIASMD | LCDA | WA | LP<3:0> | | | |
| bit 7 | | | | | | | bit 0 |

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

bit 7 **WFT:** Waveform Type bit

1 = Type-B phase changes on each frame boundary
0 = Type-A phase changes within each common type

bit 6 **BIASMD:** Bias Mode Select bit

When LMUX<1:0> = 00:

0 = Static Bias mode (do not set this bit to '1')

When LMUX<1:0> = 01:

1 = 1/2 Bias mode
0 = 1/3 Bias mode

When LMUX<1:0> = 10:

1 = 1/2 Bias mode
0 = 1/3 Bias mode

When LMUX<1:0> = 11:

0 = 1/3 Bias mode (do not set this bit to '1')

bit 5 **LCDA:** LCD Active Status bit

1 = LCD driver module is active
0 = LCD driver module is inactive

bit 4 **WA:** LCD Write Allow Status bit

1 = Writing to the LCDDATAn registers is allowed
0 = Writing to the LCDDATAn registers is not allowed

bit 3-0 **LP<3:0>:** LCD Prescaler Selection bits

1111 = 1:16
1110 = 1:15
1101 = 1:14
1100 = 1:13
1011 = 1:12
1010 = 1:11
1001 = 1:10
1000 = 1:9
0111 = 1:8
0110 = 1:7
0101 = 1:6
0100 = 1:5
0011 = 1:4
0010 = 1:3
0001 = 1:2
0000 = 1:1

**REGISTER 27-3: LCDREF: LCD REFERENCE VOLTAGE CONTROL REGISTER**

| R/W-0/0 | R/W-0/0 | R/W-0/0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | U-0 |
|---|---|---|---|---|---|---|---|
| LCDIRE | LCDIRS | LCDIRI | — | VLCD3PE | VLCD2PE | VLCD1PE | — |
| bit 7 | | | | | | | bit 0 |

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

bit 7 **LCDIRE:** LCD Internal Reference Enable bit
1 = Internal LCD Reference is enabled and connected to the Internal Contrast Control circuit
0 = Internal LCD Reference is disabled

bit 6 **LCDIRS:** LCD Internal Reference Source bit
If LCDIRE = 1:
0 = Internal LCD Contrast Control is powered by $V_{DD}$
1 = Internal LCD Contrast Control is powered by a 3.072V output of the FVR.
If LCDIRE = 0:
Internal LCD Contrast Control is unconnected. LCD bandgap buffer is disabled.

bit 5 **LCDIRI:** LCD Internal Reference Ladder Idle Enable bit
Allows the Internal FVR buffer to shut down when the LCD Reference Ladder is in power mode 'B'
1 = When the LCD Reference Ladder is in power mode 'B', the LCD Internal FVR buffer is disabled.
0 = The LCD Internal FVR Buffer ignores the LCD Reference Ladder Power mode.

bit 4 **Unimplemented:** Read as '0'

bit 3 **VLCD3PE:** VLCD3 Pin Enable bit
1 = The VLCD3 pin is connected to the internal bias voltage LCDBIAS3**(1)**
0 = The VLCD3 pin is not connected

bit 2 **VLCD2PE:** VLCD2 Pin Enable bit
1 = The VLCD2 pin is connected to the internal bias voltage LCDBIAS2**(1)**
0 = The VLCD2 pin is not connected

bit 1 **VLCD1PE:** VLCD1 Pin Enable bit
1 = The VLCD1 pin is connected to the internal bias voltage LCDBIAS1**(1)**
0 = The VLCD1 pin is not connected

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

**Note 1:** Normal pin controls of TRISx and ANSELx are unaffected.


© 2010-2016 Microchip Technology Inc.

**REGISTER 27-4: LCDCST: LCD CONTRAST CONTROL REGISTER**

| U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | LCDCST<2:0> | | |
| bit 7 | | | | | | | bit 0 |

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

bit 7-3 **Unimplemented:** Read as ‘0’

bit 2-0 **LCDCST<2:0>:** LCD Contrast Control bits
Selects the resistance of the LCD contrast control resistor ladder
Bit Value = Resistor ladder
000 = Minimum resistance (maximum contrast). Resistor ladder is shorted.
001 = Resistor ladder is at 1/7th of maximum resistance
010 = Resistor ladder is at 2/7th of maximum resistance
011 = Resistor ladder is at 3/7th of maximum resistance
100 = Resistor ladder is at 4/7th of maximum resistance
101 = Resistor ladder is at 5/7th of maximum resistance
110 = Resistor ladder is at 6/7th of maximum resistance
111 = Resistor ladder is at maximum resistance (minimum contrast).

**REGISTER 27-5: LCDSEn: LCD SEGMENT ENABLE REGISTERS**

| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| SEn | SEn | SEn | SEn | SEn | SEn | SEn | SEn |
| bit 7 | | | | | | | bit 0 |

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

bit 7-0 **SEn:** Segment Enable bits
1 = Segment function of the pin is enabled
0 = I/O function of the pin is enabled

**REGISTER 27-6: LCDDATAn: LCD DATA REGISTERS**

| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
|---|---|---|---|---|---|---|---|
| SEGx-COMy | SEGx-COMy | SEGx-COMy | SEGx-COMy | SEGx-COMy | SEGx-COMy | SEGx-COMy | SEGx-COMy |
| bit 7 | | | | | | | bit 0 |

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

bit 7-0 **SEGx-COMy:** Pixel On bits
1 = Pixel ON (dark)
0 = Pixel OFF (clear)


© 2010-2016 Microchip Technology Inc.

## 27.3 LCD Clock Source Selection

The LCD module has three possible clock sources:

- FOSC/256
- T1OSC
- LFINTOSC

The first clock source is the system clock divided by 256 (FOSC/256). This divider ratio is chosen to provide about 1 kHz output when the system clock is 8 MHz. The divider is not programmable. Instead, the LCD prescaler bits LP<3:0> of the LCDPS register are used to set the LCD frame clock rate.

The second clock source is the T1OSC. This also gives about 1 kHz when a 32.768 kHz crystal is used with the Timer1 oscillator. To use the Timer1 oscillator as a clock source, the T1OSCEN bit of the T1CON register should be set.

The third clock source is the 31 kHz LFINTOSC, which provides approximately 1 kHz output.

The second and third clock sources may be used to continue running the LCD while the processor is in Sleep.

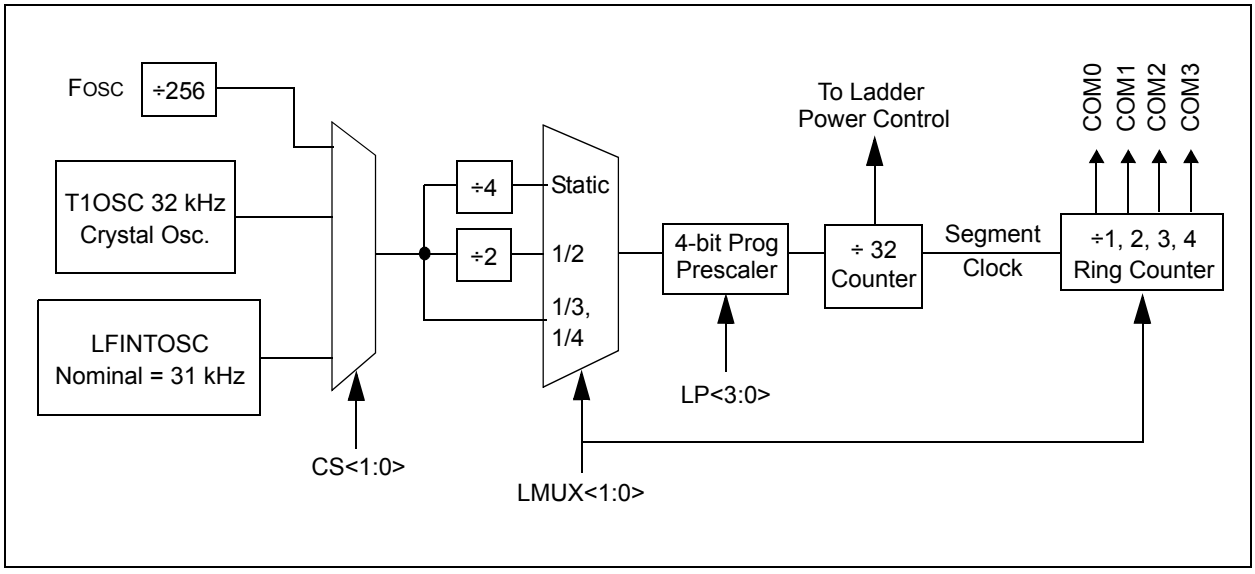
Using bits CS<1:0> of the LCDCON register can select any of these clock sources.

### 27.3.1 LCD PRESCALER

A 4-bit counter is available as a prescaler for the LCD clock. The prescaler is not directly readable or writable; its value is set by the LP<3:0> bits of the LCDPS register, which determine the prescaler assignment and prescale ratio.

The prescale values are selectable from 1:1 through 1:16.

**FIGURE 27-2: LCD CLOCK GENERATION**



## 27.4 LCD Bias Voltage Generation

The LCD module can be configured for one of three bias types:

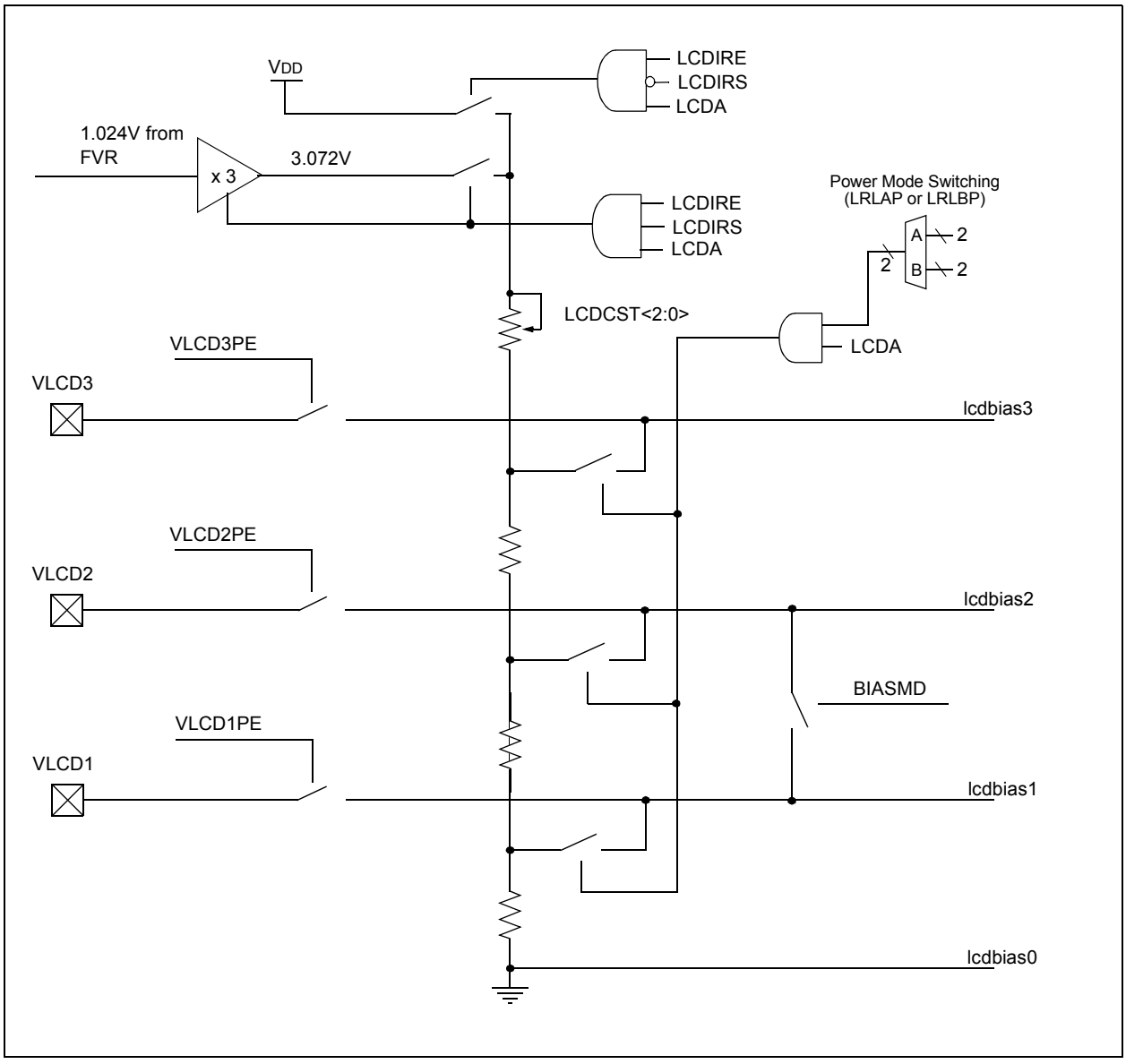
- Static Bias (two-voltage levels: VSS and VLCD)
- 1/2 Bias (three-voltage levels: VSS, 1/2 VLCD and VLCD)
- 1/3 Bias (four voltage levels: VSS, 1/3 VLCD, 2/3 VLCD and VLCD)

**TABLE 27-2: LCD BIAS VOLTAGES**

| | Static Bias | 1/2 Bias | 1/3 Bias |
|---|---|---|---|
| LCD Bias 0 | VSS | VSS | VSS |
| LCD Bias 1 | — | 1/2 VDD | 1/3 VDD |
| LCD Bias 2 | — | 1/2 VDD | 2/3 VDD |
| LCD Bias 3 | VLCD3 | VLCD3 | VLCD3 |

The internal contrast control and an internal reference ladder are provided internally to the PIC16(L)F1946/47 so that the user is not forced to place external components and use up to three pins for bias voltage generation. Both of these features may be used in conjunction with the external VLCD<3:1> pins, to provide maximum flexibility. Refer to Figure 27-3.

**FIGURE 27-3: LCD BIAS VOLTAGE GENERATION BLOCK DIAGRAM**





© 2010-2016 Microchip Technology Inc.

## 27.5 LCD Bias Internal Reference Ladder

The internal reference ladder can be used to divide the LCD bias voltage two or three equally spaced voltages that will be supplied to the LCD segment pins. To create this, the reference ladder consists of three matched resistors. Refer to Figure 27-3.

### 27.5.1 BIAS MODE INTERACTION

When in 1/2 Bias mode (BIASMD = 1), the middle resistor of the ladder is shorted out so that only two voltages are generated. The current consumption of the ladder is higher in this mode, with the one resistor removed.

**TABLE 27-3: LCD INTERNAL LADDER POWER MODES (1/3 BIAS)**

| Power Mode | Nominal Resistance of Entire Ladder | Nominal IDD |
|---|---|---|
| Low | 3 Mohm | 1 µA |
| Medium | 300 kohm | 10 µA |
| High | 30 kohm | 100 µA |

### 27.5.2 POWER MODES

The internal reference ladder may be operated in one of three power modes. This allows the user to trade off LCD contrast for power in the specific application. The larger the LCD glass, the more capacitance is present on a physical LCD segment, requiring more current to maintain the same contrast level.

Three different power modes are available, LP, MP and HP. The internal reference ladder can also be turned off for applications that wish to provide an external ladder or to minimize power consumption. Disabling the internal reference ladder results in all of the ladders being disconnected, allowing external voltages to be supplied.

Whenever the LCD module is inactive (LCDA = 0), the internal reference ladder will be turned off.

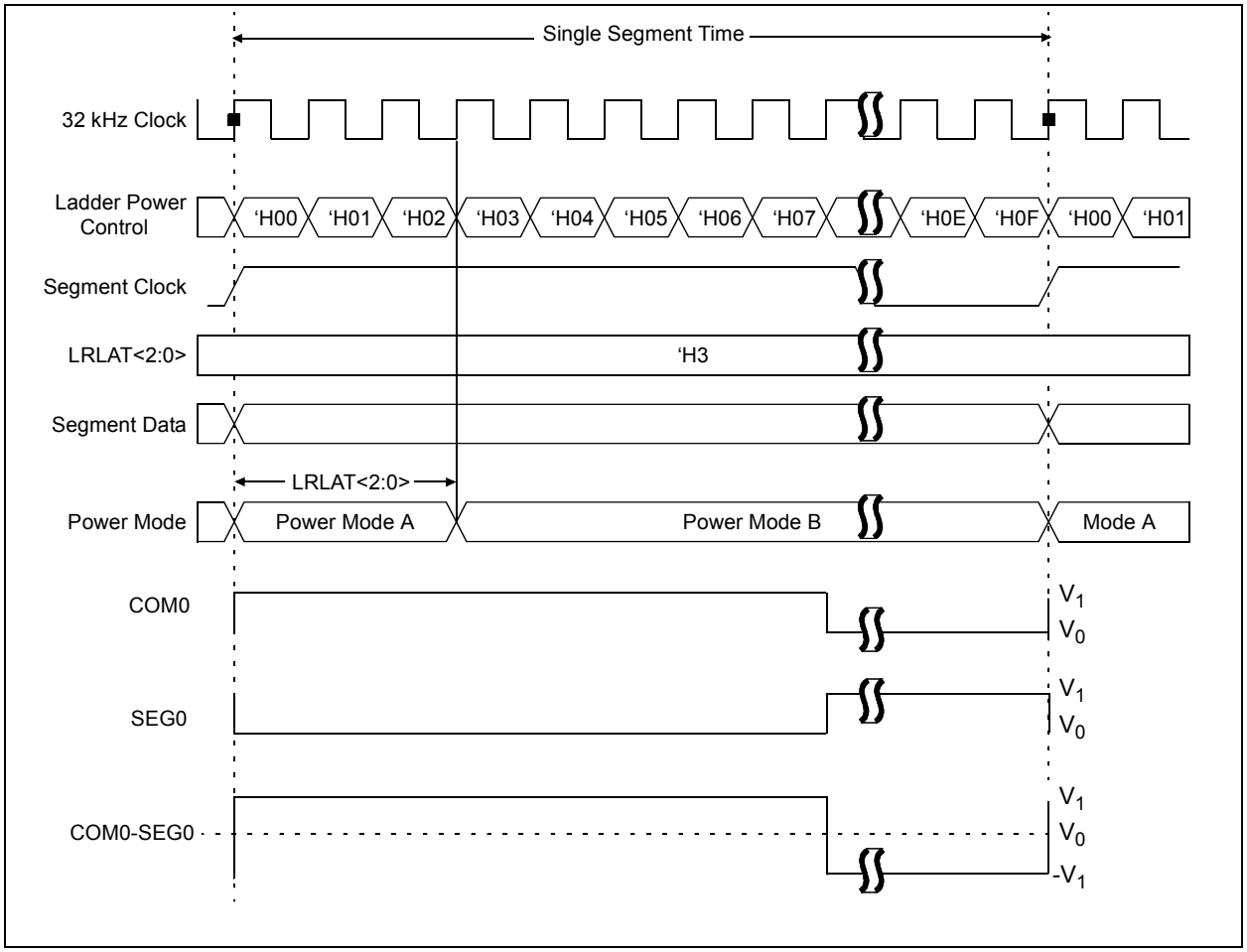
### 27.5.3 AUTOMATIC POWER MODE SWITCHING

As an LCD segment is electrically only a capacitor, current is drawn only during the interval where the voltage is switching. To minimize total device current, the LCD internal reference ladder can be operated in a different power mode for the transition portion of the duration. This is controlled by the LCDRL Register (Register 27-7).

The LCDRL register allows switching between two power modes, designated 'A' and 'B'. 'A' Power mode is active for a programmable time, beginning at the time when the LCD segments transition. 'B' Power mode is the remaining time before the segments or commons change again. The LRLAT<2:0> bits select how long, if any, that the 'A' Power mode is active. Refer to Figure 27-4.

To implement this, the 5-bit prescaler used to divide the 32 kHz clock down to the LCD controller's 1 kHz base rate is used to select the power mode.

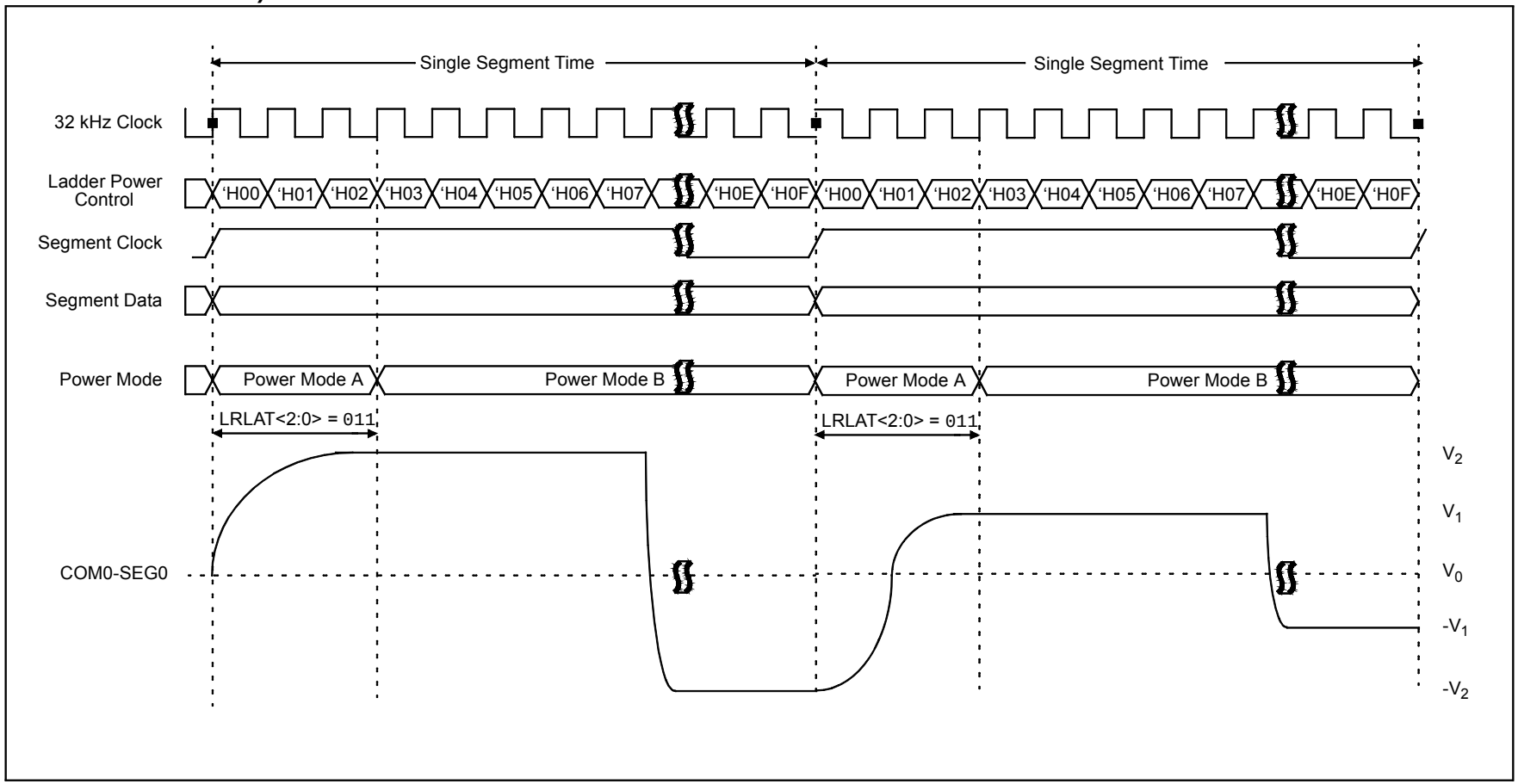
**FIGURE 27-4: LCD INTERNAL REFERENCE LADDER POWER MODE SWITCHING DIAGRAM – TYPE A**





© 2010-2016 Microchip Technology Inc.

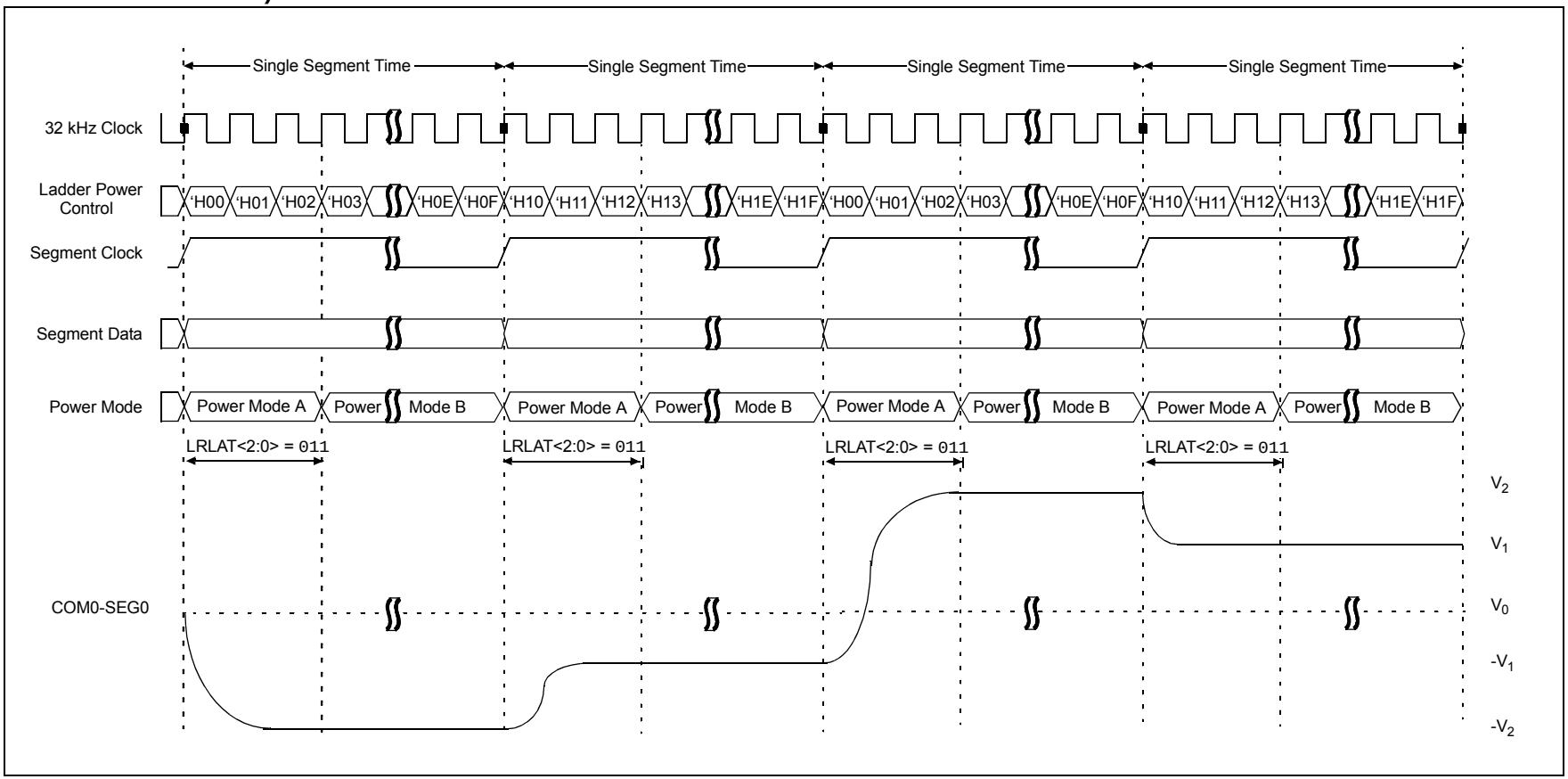
**FIGURE 27-5: LCD INTERNAL REFERENCE LADDER POWER MODE SWITCHING DIAGRAM – TYPE A WAVEFORM (1/2 MUX, 1/2 BIAS DRIVE)**






© 2010-2016 Microchip Technology Inc.

**FIGURE 27-6: LCD INTERNAL REFERENCE LADDER POWER MODE SWITCHING DIAGRAM – TYPE B WAVEFORM (1/2 MUX, 1/2 BIAS DRIVE)**




© 2010-2016 Microchip Technology Inc.

## 27.6 Register Definitions: LCD Ladder Control

### REGISTER 27-7: LCDRL: LCD REFERENCE LADDER CONTROL REGISTERS

| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| LRLAP<1:0> | | LRLBP<1:0> | | — | LRLAT<2:0> | | |
| bit 7 | | | | | | | bit 0 |

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

bit 7-6 **LRLAP<1:0>:** LCD Reference Ladder A Time Power Control bits
During Time interval A (Refer to Figure 27-4):
00 = Internal LCD Reference Ladder is powered down and unconnected
01 = Internal LCD Reference Ladder is powered in Low-Power mode
10 = Internal LCD Reference Ladder is powered in Medium-Power mode
11 = Internal LCD Reference Ladder is powered in High-Power mode

bit 5-4 **LRLBP<1:0>:** LCD Reference Ladder B Time Power Control bits
During Time interval B (Refer to Figure 27-4):
00 = Internal LCD Reference Ladder is powered down and unconnected
01 = Internal LCD Reference Ladder is powered in Low-Power mode
10 = Internal LCD Reference Ladder is powered in Medium-Power mode
11 = Internal LCD Reference Ladder is powered in High-Power mode

bit 3 **Unimplemented:** Read as ‘0’

bit 2-0 **LRLAT<2:0>:** LCD Reference Ladder A Time interval control bits
Sets the number of 32 kHz clocks that the A Time interval power mode is active

For type A waveforms (WFT = 0):

000 = Internal LCD Reference Ladder is always in ‘B’ Power mode
001 = Internal LCD Reference Ladder is in ‘A’ Power mode for one clock and ‘B’ Power mode for 15 clocks
010 = Internal LCD Reference Ladder is in ‘A’ Power mode for two clocks and ‘B’ Power mode for 14 clocks
011 = Internal LCD Reference Ladder is in ‘A’ Power mode for three clocks and ‘B’ Power mode for 13 clocks
100 = Internal LCD Reference Ladder is in ‘A’ Power mode for four clocks and ‘B’ Power mode for 12 clocks
101 = Internal LCD Reference Ladder is in ‘A’ Power mode for five clocks and ‘B’ Power mode for 11 clocks
110 = Internal LCD Reference Ladder is in ‘A’ Power mode for six clocks and ‘B’ Power mode for 10 clocks
111 = Internal LCD Reference Ladder is in ‘A’ Power mode for seven clocks and ‘B’ Power mode for nine clocks

For type B waveforms (WFT = 1):

000 = Internal LCD Reference Ladder is always in ‘B’ Power mode.
001 = Internal LCD Reference Ladder is in ‘A’ Power mode for one clock and ‘B’ Power mode for 31 clocks
010 = Internal LCD Reference Ladder is in ‘A’ Power mode for two clocks and ‘B’ Power mode for 30 clocks
011 = Internal LCD Reference Ladder is in ‘A’ Power mode for three clocks and ‘B’ Power mode for 29 clocks
100 = Internal LCD Reference Ladder is in ‘A’ Power mode for four clocks and ‘B’ Power mode for 28 clocks
101 = Internal LCD Reference Ladder is in ‘A’ Power mode for five clocks and ‘B’ Power mode for 27 clocks
110 = Internal LCD Reference Ladder is in ‘A’ Power mode for six clocks and ‘B’ Power mode for 26 clocks
111 = Internal LCD Reference Ladder is in ‘A’ Power mode for seven clocks and ‘B’ Power mode for 25 clocks

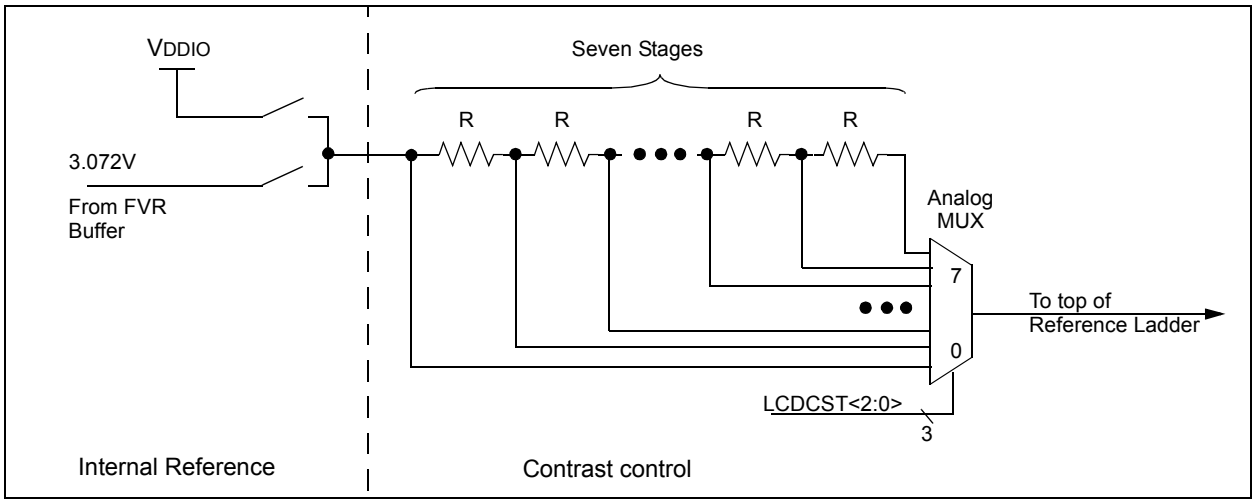
### 27.6.1 CONTRAST CONTROL

The LCD contrast control circuit consists of a seven-tap resistor ladder, controlled by the LCDCST bits. Refer to Figure 27-7.

The contrast control circuit is used to decrease the output voltage of the signal source by a total of approximately 10%, when LCDCST = 111.

Whenever the LCD module is inactive (LCDA = 0), the contrast control ladder will be turned off (open).

**FIGURE 27-7: INTERNAL REFERENCE AND CONTRAST CONTROL BLOCK DIAGRAM**




### 27.6.2 INTERNAL REFERENCE

Under firmware control, an internal reference for the LCD bias voltages can be enabled. When enabled, the source of this voltage can be either VDDIO or a voltage three times the main fixed voltage reference (3.072V). When no internal reference is selected, the LCD contrast control circuit is disabled and LCD bias must be provided externally.

Whenever the LCD module is inactive (LCDA = 0), the internal reference will be turned off.

When the internal reference is enabled and the Fixed Voltage Reference is selected, the LCDIRI bit can be used to minimize power consumption by tying into the LCD reference ladder automatic power mode switching. When LCDIRI = 1 and the LCD reference ladder is in Power mode 'B', the LCD internal FVR buffer is disabled.

| **Note:** | The LCD module automatically turns on the Fixed Voltage Reference when needed. |
|---|---|

### 27.6.3 VLCD<3:1> PINS

The VLCD<3:1> pins provide the ability for an external LCD bias network to be used instead of the internal ladder. Use of the VLCD<3:1> pins does not prevent use of the internal ladder. Each VLCD pin has an independent control in the LCDREF register (Register 27-3), allowing access to any or all of the LCD Bias signals. This architecture allows for maximum flexibility in different applications.

For example, the VLCD<3:1> pins may be used to add capacitors to the internal reference ladder, increasing the drive capacity.

For applications where the internal contrast control is insufficient, the firmware can choose to only enable the VLCD3 pin, allowing an external contrast control circuit to use the internal reference divider.


© 2010-2016 Microchip Technology Inc.

## 27.7 LCD Multiplex Types

The LCD driver module can be configured into one of four multiplex types:

- Static (only COM0 is used)
- 1/2 multiplex (COM<1:0> are used)
- 1/3 multiplex (COM<2:0> are used)
- 1/4 multiplex (COM<3:0> are used)

The LMUX<1:0> bit setting of the LCDCON register decides which of the LCD common pins are used (see Table 27-4 for details).

If the pin is a digital I/O, the corresponding TRIS bit controls the data direction. If the pin is a COM drive, then the TRIS setting of that pin is overridden.

**TABLE 27-4: COMMON PIN USAGE**

| Multiplex | LMUX <1:0> | COM3 | COM2 | COM1 | COM0 |
|---|---|---|---|---|---|
| Static | 00 | Unused | Unused | Unused | Active |
| 1/2 | 01 | Unused | Unused | Active | Active |
| 1/3 | 10 | Unused | Active | Active | Active |
| 1/4 | 11 | Active | Active | Active | Active |

## 27.8 Segment Enables

The LCDSEn registers are used to select the pin function for each segment pin. The selection allows each pin to operate as either an LCD segment driver or as one of the pin's alternate functions. To configure the pin as a segment pin, the corresponding bits in the LCDSEn registers must be set to '1'.

If the pin is a digital I/O, the corresponding TRIS bit controls the data direction. Any bit set in the LCDSEn registers overrides any bit settings in the corresponding TRIS register.

| **Note:** | On a Power-on Reset, these pins are configured as normal I/O, not LCD pins. |
|---|---|

## 27.9 Pixel Control

The LCDDATAx registers contain bits which define the state of each pixel. Each bit defines one unique pixel.

Register 27-6 shows the correlation of each bit in the LCDDATAx registers to the respective common and segment signals.

Any LCD pixel location not being used for display can be used as general purpose RAM.

## 27.10 LCD Frame Frequency

The rate at which the COM and SEG outputs change is called the LCD frame frequency.

**TABLE 27-5: FRAME FREQUENCY FORMULAS**

| Multiplex | Frame Frequency(2) = |
|---|---|
| Static | Clock source/(4 x (LCD Prescaler) x 32 x 1)) |
| 1/2 | Clock source/(2 x (LCD Prescaler) x 32 x 2)) |
| 1/3 | Clock source/(1 x (LCD Prescaler) x 32 x 3)) |
| 1/4 | Clock source/(1 x (LCD Prescaler) x 32 x 4)) |

**Note 1:** Clock source is FOSC/256, T1OSC or LFINTOSC.

**2:** See Figure 27-2.

**TABLE 27-6: APPROXIMATE FRAME FREQUENCY (IN Hz) USING FOSC @ 8 MHz, TIMER1 @ 32.768 kHz OR LFINTOSC**

| LP<3:0> | Static | 1/2 | 1/3 | 1/4 |
|---|---|---|---|---|
| 2 | 122 | 122 | 162 | 122 |
| 3 | 81 | 81 | 108 | 81 |
| 4 | 61 | 61 | 81 | 61 |
| 5 | 49 | 49 | 65 | 49 |
| 6 | 41 | 41 | 54 | 41 |
| 7 | 35 | 35 | 47 | 35 |

© 2010-2016 Microchip Technology Inc.

**TABLE 27-7: LCD SEGMENT MAPPING WORKSHEET**

| LCD Function | COM0 | | COM1 | | COM2 | | COM3 | |
|---|---|---|---|---|---|---|---|---|
| | LCDDATAx Address | LCD Segment | LCDDATAx Address | LCD Segment | LCDDATAx Address | LCD Segment | LCDDATAx Address | LCD Segment |
| SEG0 | LCDDATA0, 0 | | LCDDATA3, 0 | | LCDDATA6, 0 | | LCDDATA9, 0 | |
| SEG1 | LCDDATA0, 1 | | LCDDATA3, 1 | | LCDDATA6, 1 | | LCDDATA9, 1 | |
| SEG2 | LCDDATA0, 2 | | LCDDATA3, 2 | | LCDDATA6, 2 | | LCDDATA9, 2 | |
| SEG3 | LCDDATA0, 3 | | LCDDATA3, 3 | | LCDDATA6, 3 | | LCDDATA9, 3 | |
| SEG4 | LCDDATA0, 4 | | LCDDATA3, 4 | | LCDDATA6, 4 | | LCDDATA9, 4 | |
| SEG5 | LCDDATA0, 5 | | LCDDATA3, 5 | | LCDDATA6, 5 | | LCDDATA9, 5 | |
| SEG6 | LCDDATA0, 6 | | LCDDATA3, 6 | | LCDDATA6, 6 | | LCDDATA9, 6 | |
| SEG7 | LCDDATA0, 7 | | LCDDATA3, 7 | | LCDDATA6, 7 | | LCDDATA9, 7 | |
| SEG8 | LCDDATA1, 0 | | LCDDATA4, 0 | | LCDDATA7, 0 | | LCDDATA10, 0 | |
| SEG9 | LCDDATA1, 1 | | LCDDATA4, 1 | | LCDDATA7, 1 | | LCDDATA10, 1 | |
| SEG10 | LCDDATA1, 2 | | LCDDATA4, 2 | | LCDDATA7, 2 | | LCDDATA10, 2 | |
| SEG11 | LCDDATA1, 3 | | LCDDATA4, 3 | | LCDDATA7, 3 | | LCDDATA10, 3 | |
| SEG12 | LCDDATA1, 4 | | LCDDATA4, 4 | | LCDDATA7, 4 | | LCDDATA10, 4 | |
| SEG13 | LCDDATA1, 5 | | LCDDATA4, 5 | | LCDDATA7, 5 | | LCDDATA10, 5 | |
| SEG14 | LCDDATA1, 6 | | LCDDATA4, 6 | | LCDDATA7, 6 | | LCDDATA10, 6 | |
| SEG15 | LCDDATA1, 7 | | LCDDATA4, 7 | | LCDDATA7, 7 | | LCDDATA10, 7 | |
| SEG16 | LCDDATA2, 0 | | LCDDATA5, 0 | | LCDDATA8, 0 | | LCDDATA11, 0 | |
| SEG17 | LCDDATA2, 1 | | LCDDATA5, 1 | | LCDDATA8, 1 | | LCDDATA11, 1 | |
| SEG18 | LCDDATA2, 2 | | LCDDATA5, 2 | | LCDDATA8, 2 | | LCDDATA11, 2 | |
| SEG19 | LCDDATA2, 3 | | LCDDATA5, 3 | | LCDDATA8, 3 | | LCDDATA11, 3 | |
| SEG20 | LCDDATA2, 4 | | LCDDATA5, 4 | | LCDDATA8, 4 | | LCDDATA11, 4 | |
| SEG21 | LCDDATA2, 5 | | LCDDATA5, 5 | | LCDDATA8, 5 | | LCDDATA11, 5 | |
| SEG22 | LCDDATA2, 6 | | LCDDATA5, 6 | | LCDDATA8, 6 | | LCDDATA11, 6 | |
| SEG23 | LCDDATA2, 7 | | LCDDATA5, 7 | | LCDDATA8, 7 | | LCDDATA11, 7 | |
| SEG24 | LCDDATA12, 0 | | LCDDATA15, 0 | | LCDDATA18, 0 | | LCDDATA21, 0 | |
| SEG25 | LCDDATA12, 1 | | LCDDATA15, 1 | | LCDDATA18, 1 | | LCDDATA21, 1 | |
| SEG26 | LCDDATA12, 2 | | LCDDATA15, 2 | | LCDDATA18, 2 | | LCDDATA21, 2 | |
| SEG27 | LCDDATA12, 3 | | LCDDATA15, 3 | | LCDDATA18, 3 | | LCDDATA21, 3 | |
| SEG28 | LCDDATA12, 4 | | LCDDATA15, 4 | | LCDDATA18, 4 | | LCDDATA21, 4 | |
| SEG29 | LCDDATA12, 5 | | LCDDATA15, 5 | | LCDDATA18, 5 | | LCDDATA21, 5 | |
| SEG30 | LCDDATA12, 6 | | LCDDATA15, 6 | | LCDDATA18, 6 | | LCDDATA21, 6 | |
| SEG31 | LCDDATA12, 7 | | LCDDATA15, 7 | | LCDDATA18, 7 | | LCDDATA21, 7 | |
| SEG32 | LCDDATA13, 0 | | LCDDATA16, 0 | | LCDDATA19, 0 | | LCDDATA22, 0 | |
| SEG33 | LCDDATA13, 1 | | LCDDATA16, 1 | | LCDDATA19, 1 | | LCDDATA22, 1 | |
| SEG34 | LCDDATA13, 2 | | LCDDATA16, 2 | | LCDDATA19, 2 | | LCDDATA22, 2 | |
| SEG35 | LCDDATA13, 3 | | LCDDATA16, 3 | | LCDDATA19, 3 | | LCDDATA22, 3 | |
| SEG36 | LCDDATA13, 4 | | LCDDATA16, 4 | | LCDDATA19, 4 | | LCDDATA22, 4 | |
| SEG37 | LCDDATA13, 5 | | LCDDATA16, 5 | | LCDDATA19, 5 | | LCDDATA22, 5 | |
| SEG38 | LCDDATA13, 6 | | LCDDATA16, 6 | | LCDDATA19, 6 | | LCDDATA22, 6 | |
| SEG39 | LCDDATA13, 7 | | LCDDATA16, 7 | | LCDDATA19, 7 | | LCDDATA22, 7 | |
| SEG40 | LCDDATA14, 0 | | LCDDATA17, 0 | | LCDDATA20, 0 | | LCDDATA23, 0 | |
| SEG41 | LCDDATA14, 1 | | LCDDATA17, 1 | | LCDDATA20, 1 | | LCDDATA23, 1 | |
| SEG42 | LCDDATA14, 2 | | LCDDATA17, 2 | | LCDDATA20, 2 | | LCDDATA23, 2 | |
| SEG43 | LCDDATA14, 3 | | LCDDATA17, 3 | | LCDDATA20, 3 | | LCDDATA23, 3 | |
| SEG44 | LCDDATA14, 4 | | LCDDATA17, 4 | | LCDDATA20, 4 | | LCDDATA23, 4 | |
| SEG45 | LCDDATA14, 5 | | LCDDATA17, 5 | | LCDDATA20, 5 | | LCDDATA23, 5 | |


© 2010-2016 Microchip Technology Inc.

## 27.11 LCD Waveform Generation

LCD waveforms are generated so that the net AC voltage across the dark pixel should be maximized and the net AC voltage across the clear pixel should be minimized. The net DC voltage across any pixel should be zero.

The COM signal represents the time slice for each common, while the SEG contains the pixel data.

The pixel signal (COM-SEG) will have no DC component and it can take only one of the two RMS values. The higher RMS value will create a dark pixel and a lower RMS value will create a clear pixel.

As the number of commons increases, the delta between the two RMS values decreases. The delta represents the maximum contrast that the display can have.

The LCDs can be driven by two types of waveforms: Type-A and Type-B. In Type-A waveform, the phase changes within each common type, whereas in Type-B waveform, the phase changes on each frame boundary. Thus, Type-A waveform maintains 0 VDC over a single frame, whereas Type-B waveform takes two frames.

| | |
|---|---|
| **Note 1:** | If Sleep has to be executed with LCD Sleep disabled (LCDCON<SLPEN> is '1'), then care must be taken to execute Sleep only when VDC on all the pixels is '0'. |
| **2:** | When the LCD clock source is FOSC/256, if Sleep is executed, irrespective of the LCDCON<SLPEN> setting, the LCD immediately goes into Sleep. Thus, take care to see that VDC on all pixels is '0' when Sleep is executed. |

Figure 27-8 through Figure 27-18 provide waveforms for static, half-multiplex, 1/3-multiplex and 1/4-multiplex drives for Type-A and Type-B waveforms.

**FIGURE 27-8: TYPE-A/TYPE-B WAVEFORMS IN STATIC DRIVE**

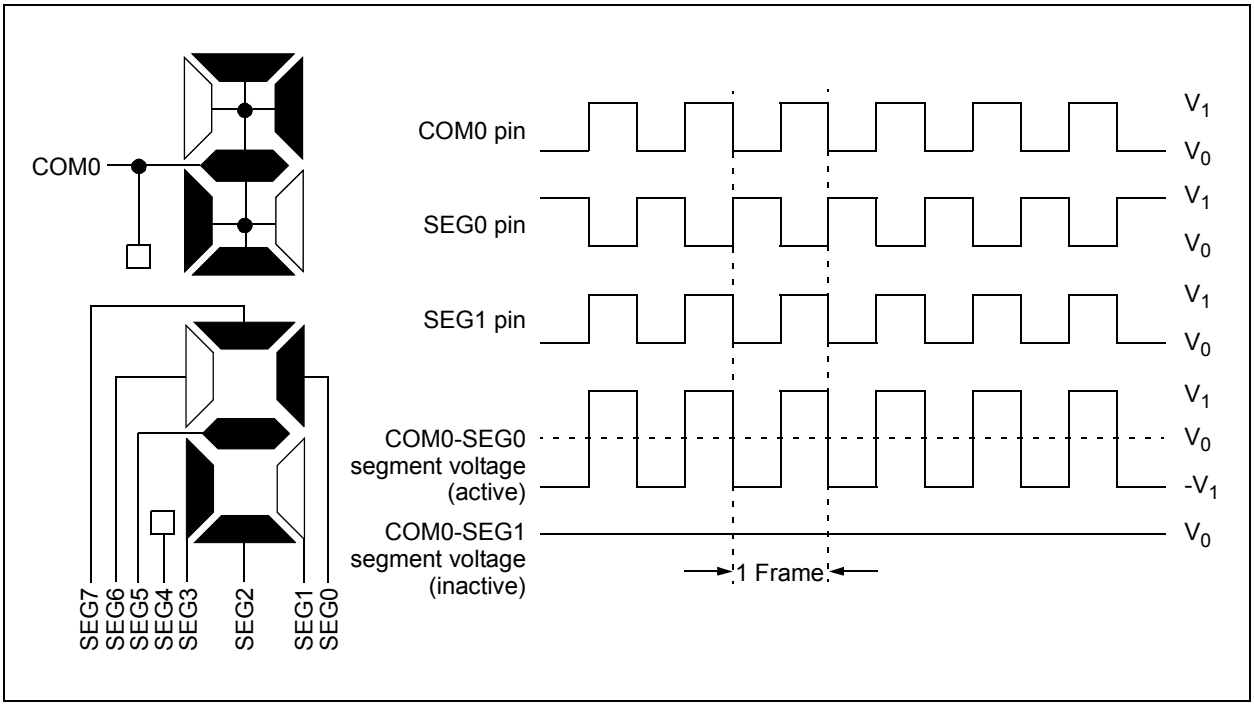
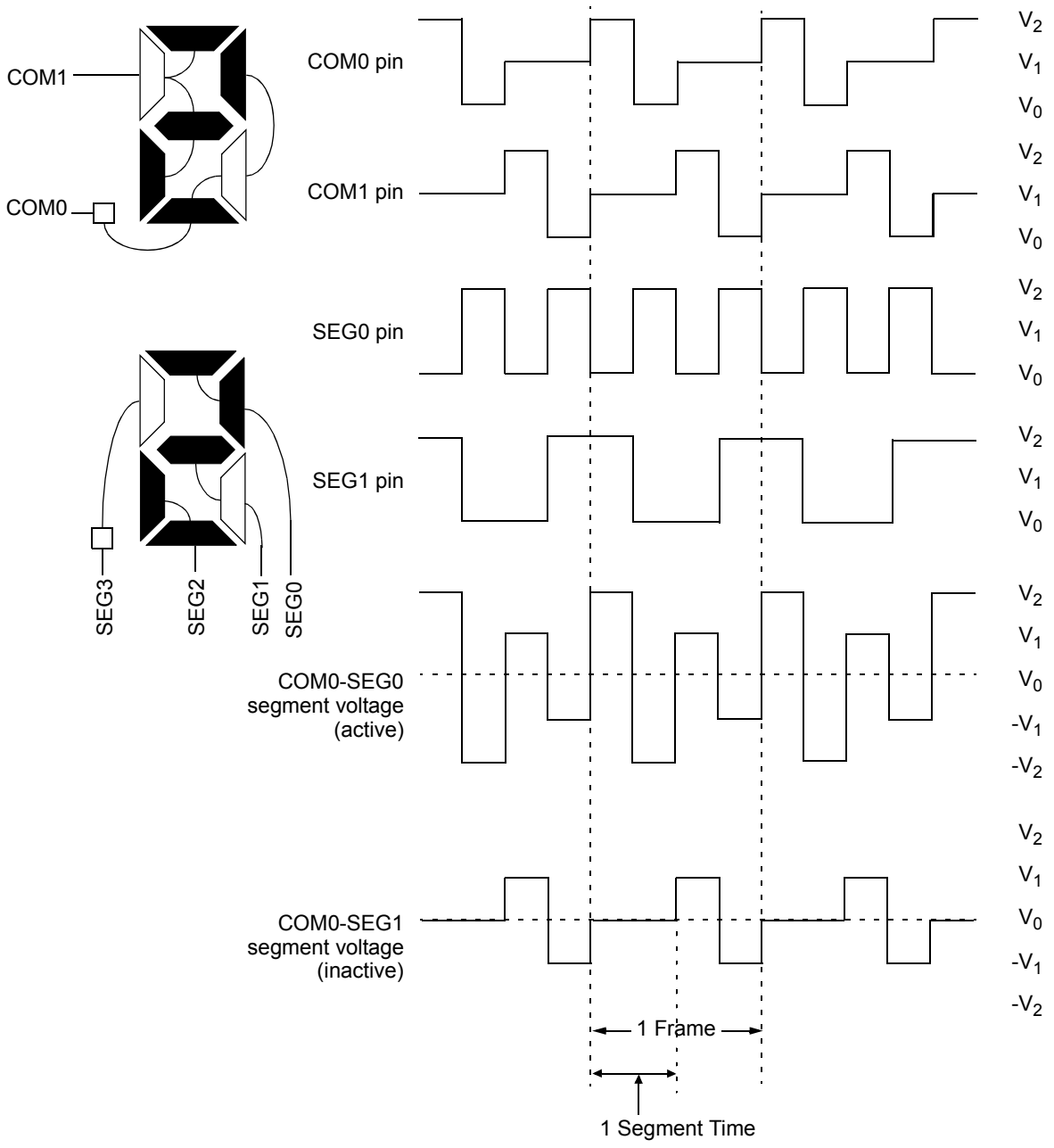


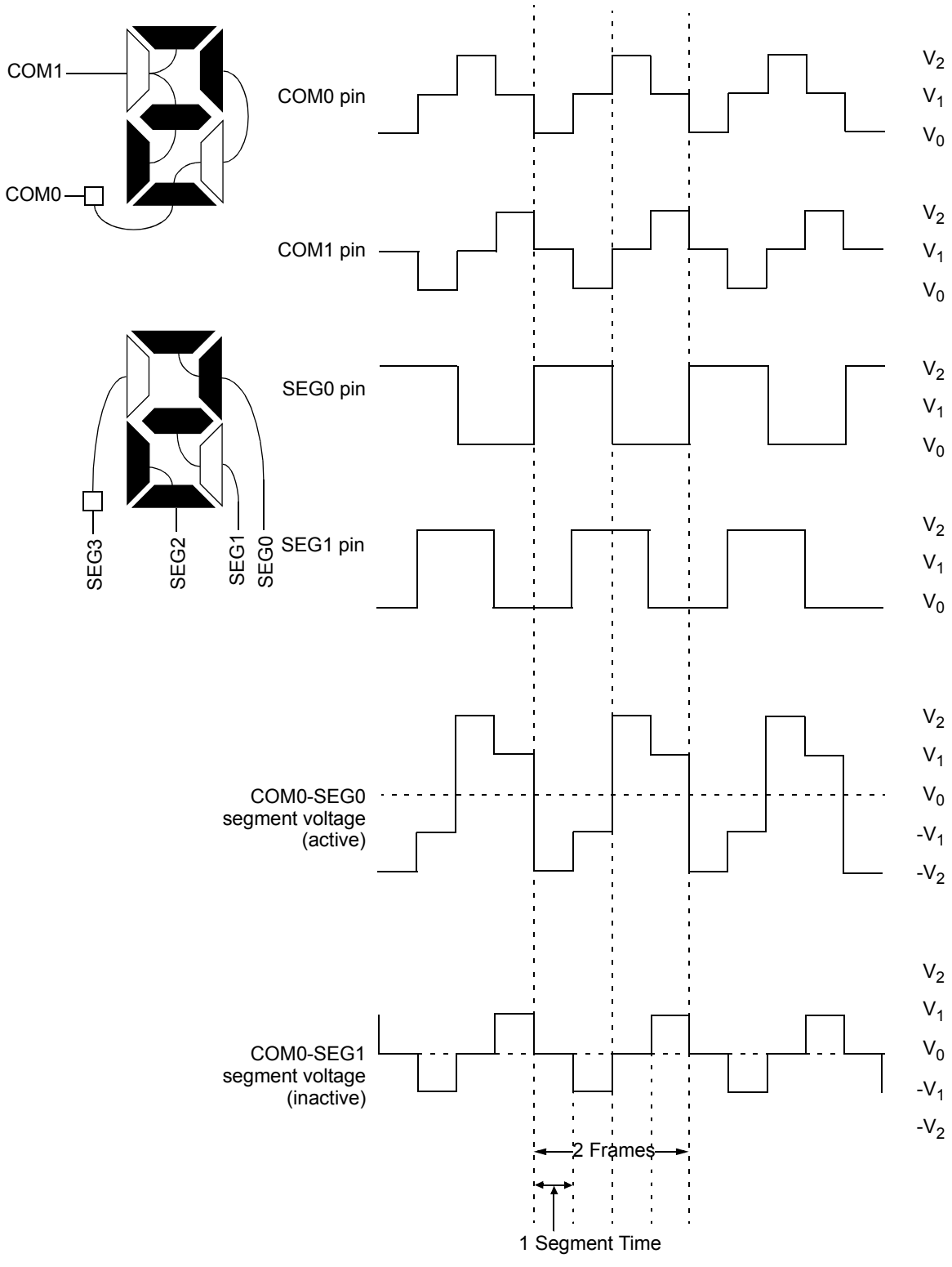
**FIGURE 27-9: TYPE-A WAVEFORMS IN 1/2 MUX, 1/2 BIAS DRIVE**




**Note:** 1 Frame = 2 single-segment times.

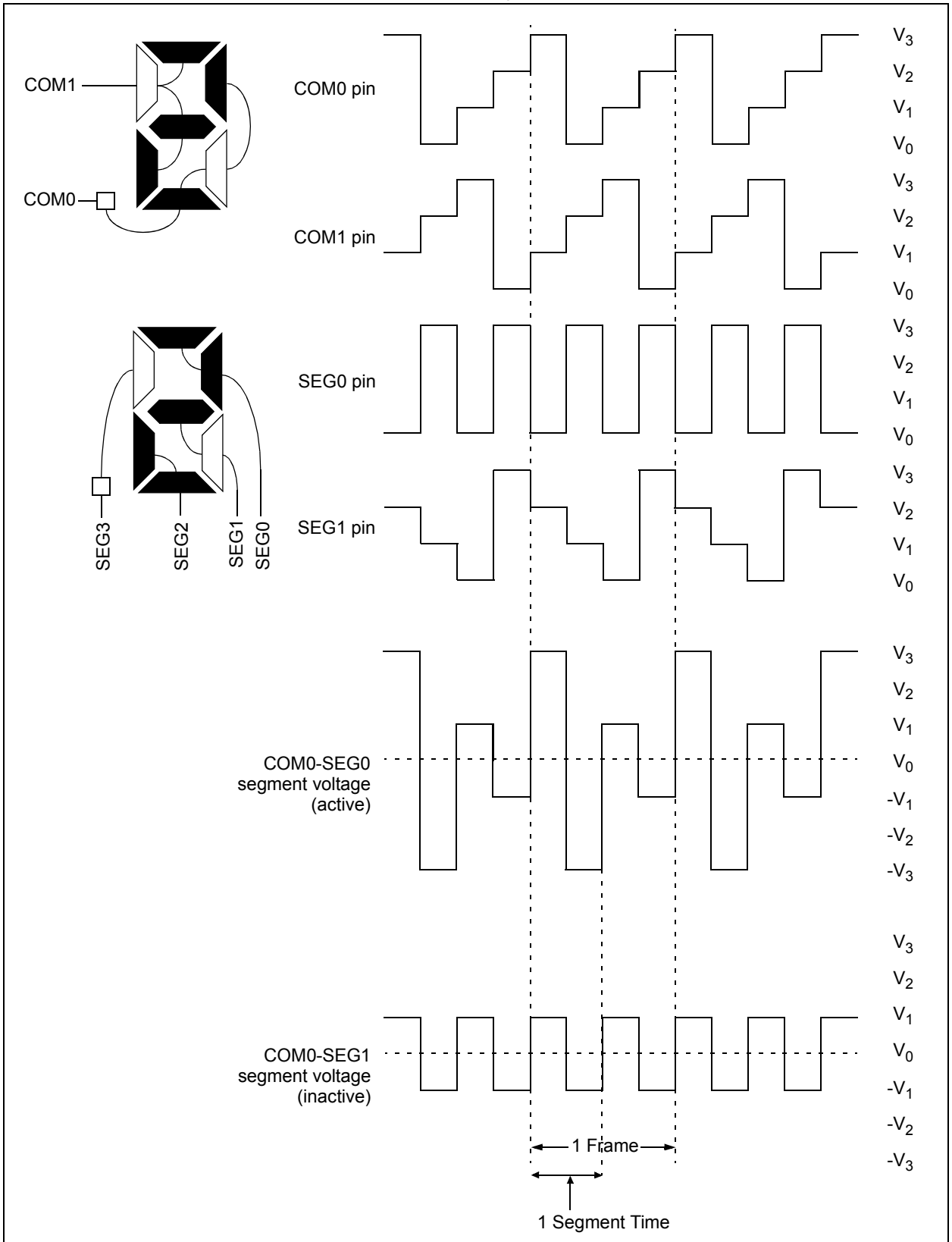

© 2010-2016 Microchip Technology Inc.

**FIGURE 27-10: TYPE-B WAVEFORMS IN 1/2 MUX, 1/2 BIAS DRIVE**




**Note:** 1 Frame = 2 single-segment times.

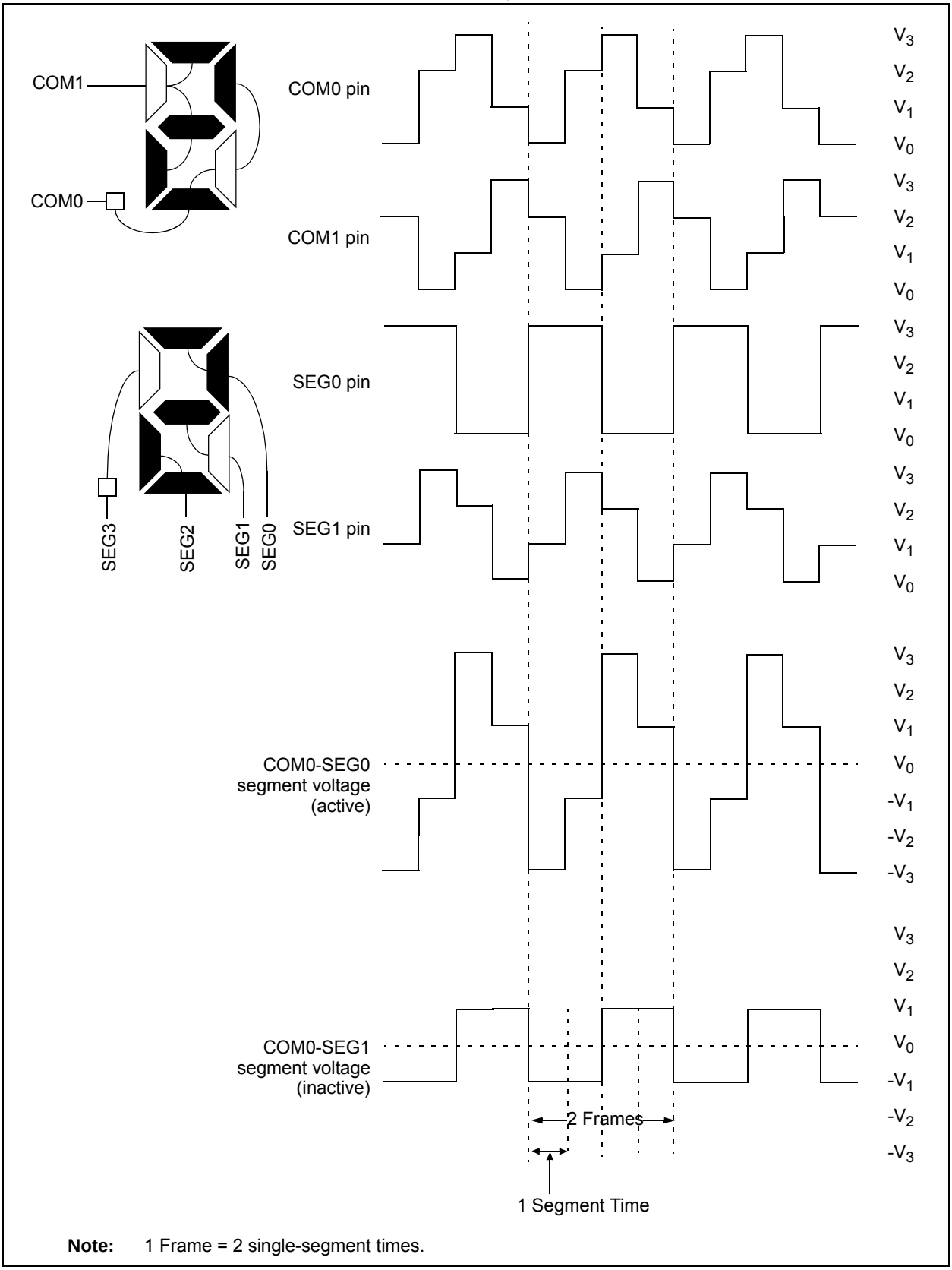
**FIGURE 27-11: TYPE-A WAVEFORMS IN 1/2 MUX, 1/3 BIAS DRIVE**




**Note:** 1 Frame = 2 single-segment times.

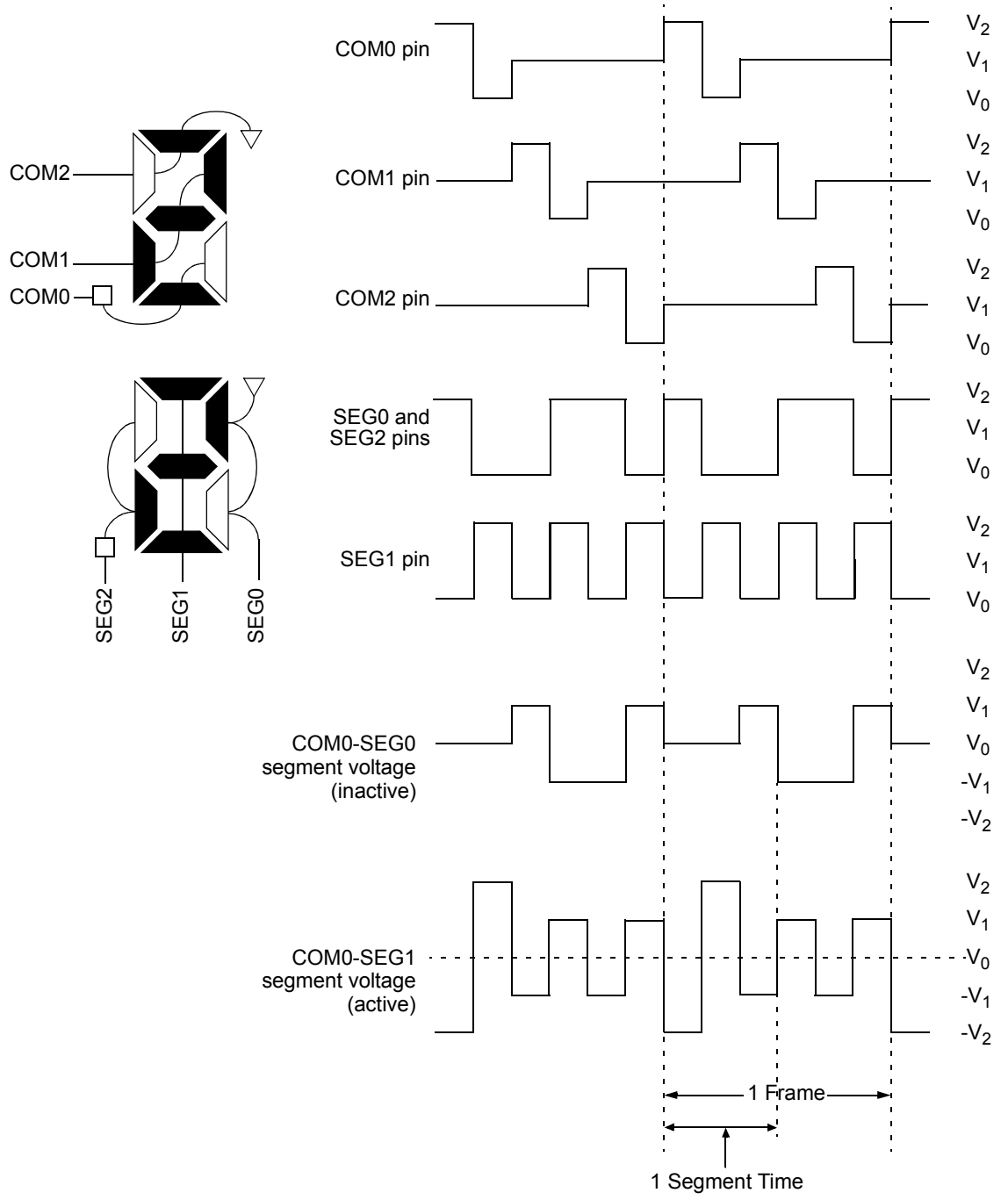

© 2010-2016 Microchip Technology Inc.

**FIGURE 27-12: TYPE-B WAVEFORMS IN 1/2 MUX, 1/3 BIAS DRIVE**




**Note:** 1 Frame = 2 single-segment times.

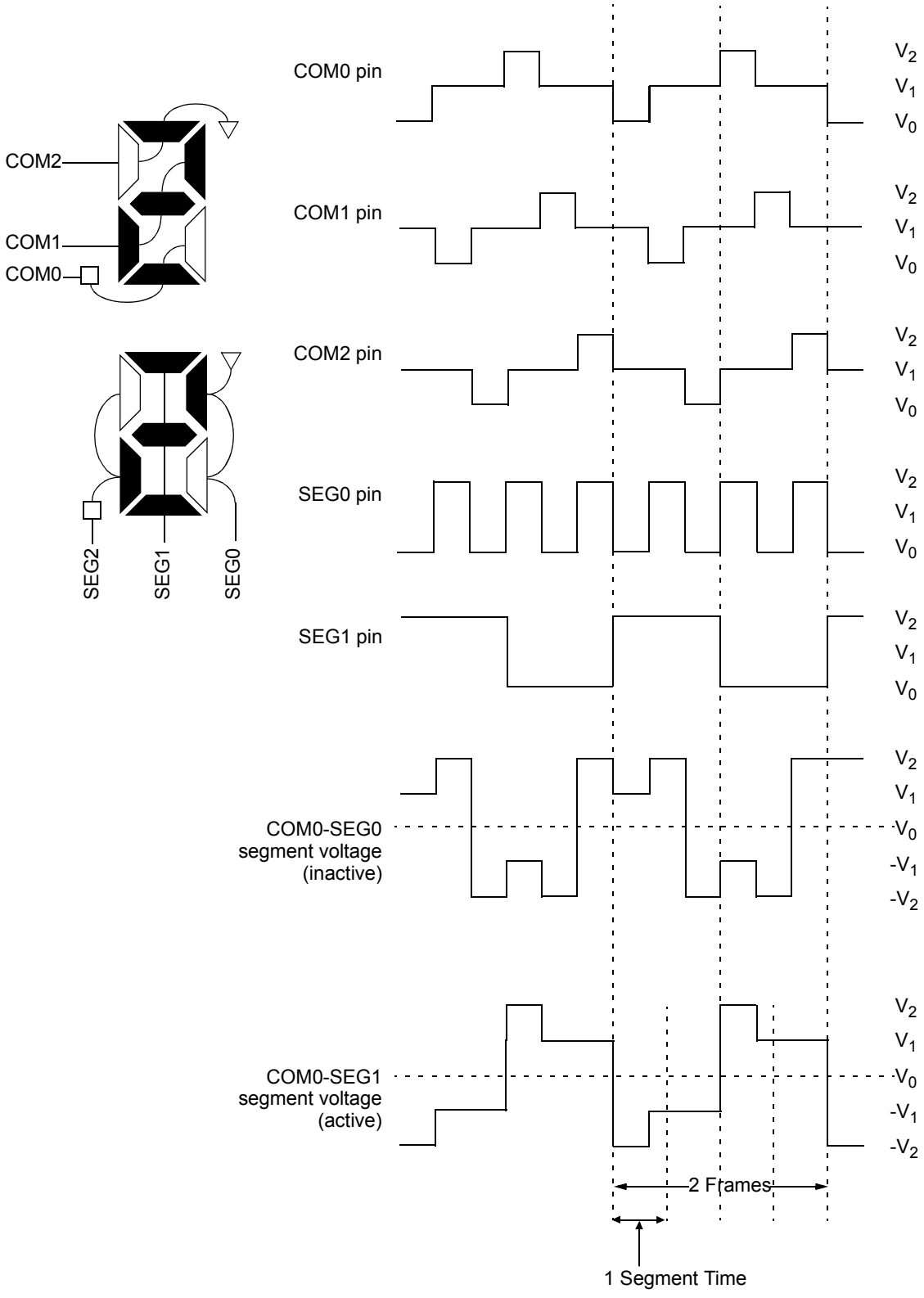
**FIGURE 27-13: TYPE-A WAVEFORMS IN 1/3 MUX, 1/2 BIAS DRIVE**




**Note:** 1 Frame = 2 single-segment times.

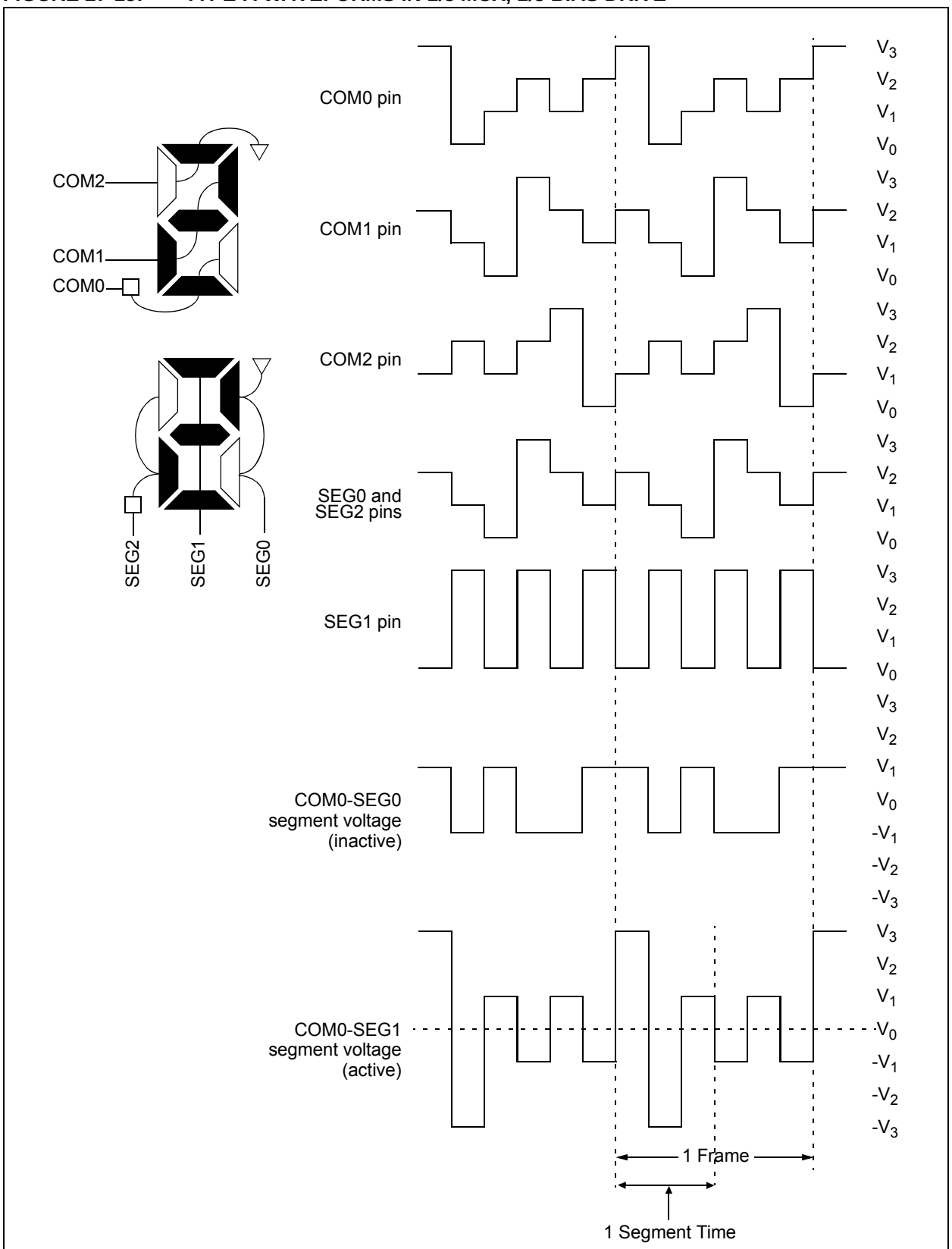

© 2010-2016 Microchip Technology Inc.

**FIGURE 27-14: TYPE-B WAVEFORMS IN 1/3 MUX, 1/2 BIAS DRIVE**




**Note:** 1 Frame = 2 single-segment times.

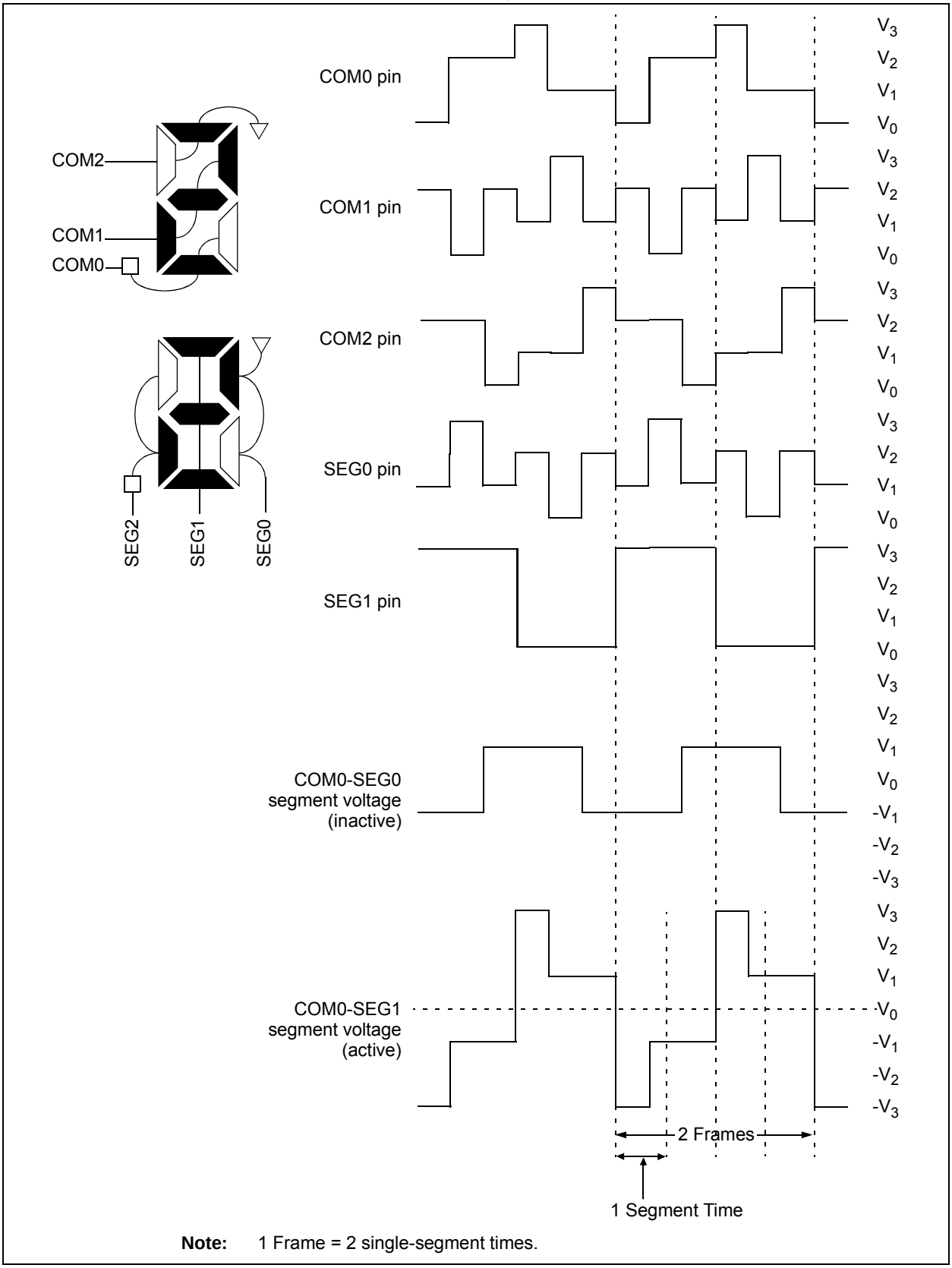
**FIGURE 27-15: TYPE-A WAVEFORMS IN 1/3 MUX, 1/3 BIAS DRIVE**




**Note:** 1 Frame = 2 single-segment times.

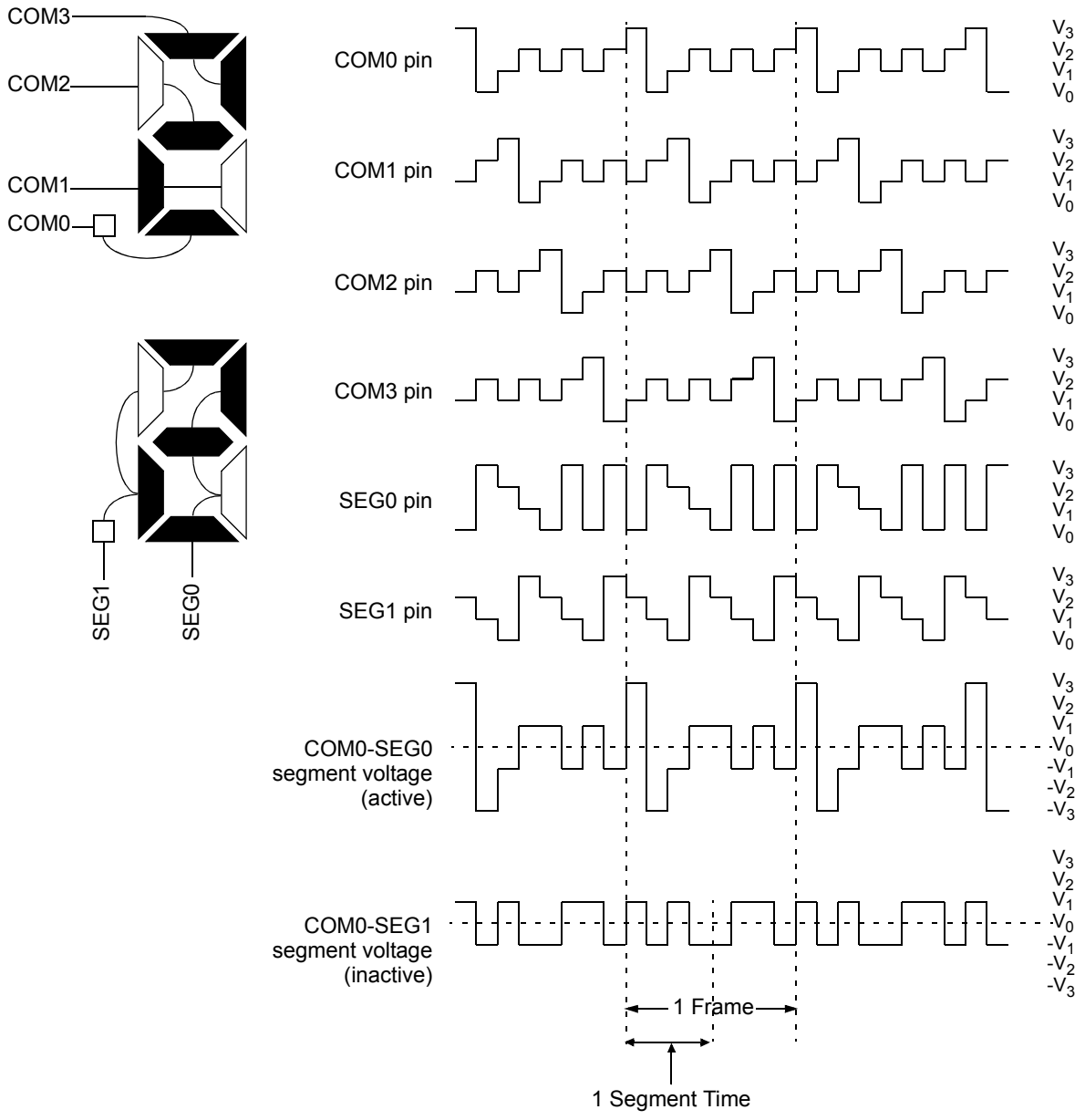

© 2010-2016 Microchip Technology Inc.

**FIGURE 27-16: TYPE-B WAVEFORMS IN 1/3 MUX, 1/3 BIAS DRIVE**




**Note:** 1 Frame = 2 single-segment times.

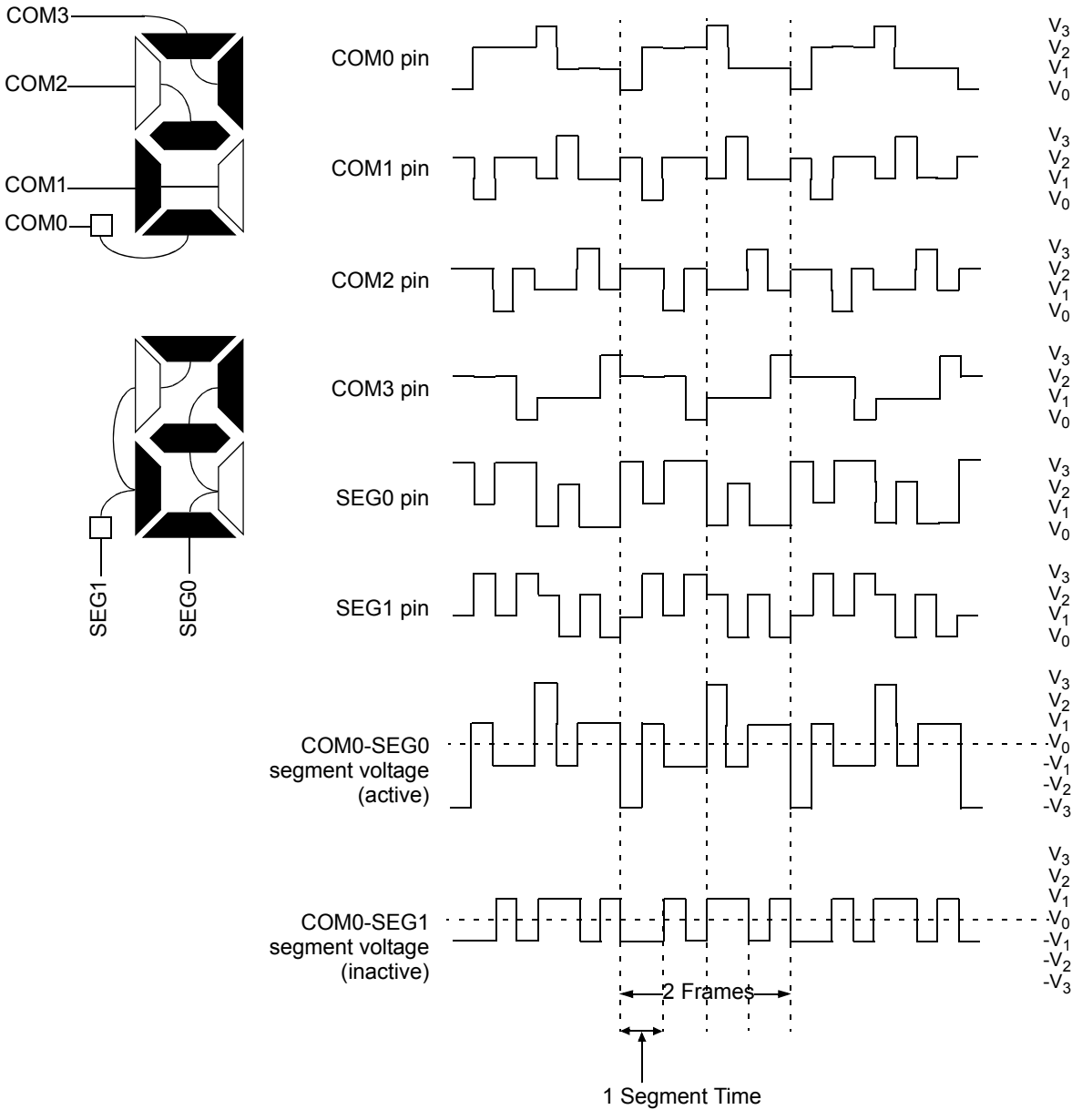
**FIGURE 27-17: TYPE-A WAVEFORMS IN 1/4 MUX, 1/3 BIAS DRIVE**




**Note:** 1 Frame = 2 single-segment times.


© 2010-2016 Microchip Technology Inc.

**FIGURE 27-18: TYPE-B WAVEFORMS IN 1/4 MUX, 1/3 BIAS DRIVE**




**Note:** 1 Frame = 2 single-segment times.

## 27.12 LCD Interrupts

The LCD module provides an interrupt in two cases. An interrupt when the LCD controller goes from active to inactive controller. An interrupt also provides unframed boundaries for Type B waveform. The LCD timing generation provides an interrupt that defines the LCD frame timing.

### 27.12.1 LCD INTERRUPT ON MODULE SHUTDOWN

An LCD interrupt is generated when the module completes shutting down (LCDA goes from '1' to '0').

### 27.12.2 LCD FRAME INTERRUPTS

A new frame is defined to begin at the leading edge of the COM0 common signal. The interrupt will be set immediately after the LCD controller completes accessing all pixel data required for a frame. This will occur at a fixed interval before the frame boundary (TFINT), as shown in Figure 27-19. The LCD controller will begin to access data for the next frame within the interval from the interrupt to when the controller begins to access data after the interrupt (TFWR). New data must be written within TFWR, as this is when the LCD controller will begin to access the data for the next frame.

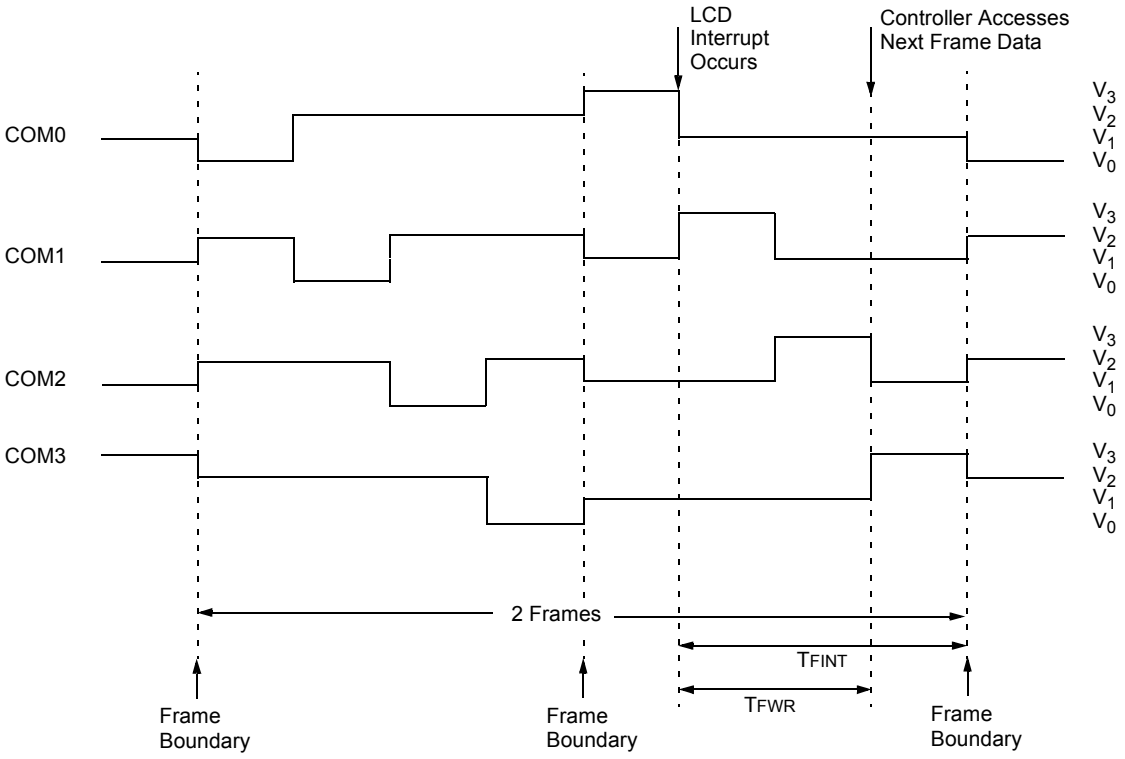
When the LCD driver is running with Type-B waveforms and the LMUX<1:0> bits are not equal to '00' (static drive), there are some additional issues that must be addressed. Since the DC voltage on the pixel takes two frames to maintain zero volts, the pixel data must not change between subsequent frames. If the pixel data were allowed to change, the waveform for the odd frames would not necessarily be the complement of the waveform generated in the even frames and a DC component would be introduced into the panel. Therefore, when using Type-B waveforms, the user must synchronize the LCD pixel updates to occur within a subframe after the frame interrupt.

To correctly sequence writing while in Type-B, the interrupt will only occur on complete phase intervals. If the user attempts to write when the write is disabled, the WERR bit of the LCDCON register is set and the write does not occur.

| | |
|---|---|
| **Note:** | The LCD frame interrupt is not generated when the Type-A waveform is selected and when the Type-B with no multiplex (static) is selected. |


© 2010-2016 Microchip Technology Inc.

**FIGURE 27-19: WAVEFORMS AND INTERRUPT TIMING IN QUARTER-DUTY CYCLE DRIVE (EXAMPLE – TYPE-B, NON-STATIC)**




$T_{FWR} = T_{FRAME}/2*(LMUX<1:0> + 1) + T_{CY}/2$

$T_{FINT} = (T_{FWR}/2 – (2\ T_{CY} + 40\ ns)) \rightarrow \text{minimum} = 1.5(T_{FRAME}/4) – (2\ T_{CY} + 40\ ns)$
$(T_{FWR}/2 – (1\ T_{CY} + 40\ ns)) \rightarrow \text{maximum} = 1.5(T_{FRAME}/4) – (1\ T_{CY} + 40\ ns)$

## 27.13 Operation During Sleep

The LCD module can operate during Sleep. The selection is controlled by bit SLPEN of the LCDCON register. Setting the SLPEN bit allows the LCD module to go to Sleep. Clearing the SLPEN bit allows the module to continue to operate during Sleep.

If a SLEEP instruction is executed and SLPEN = 1, the LCD module will cease all functions and go into a very low-current Consumption mode. The module will stop operation immediately and drive the minimum LCD voltage on both segment and common lines. Figure 27-20 shows this operation.

The LCD module can be configured to operate during Sleep. The selection is controlled by bit SLPEN of the LCDCON register. Clearing SLPEN and correctly configuring the LCD module clock will allow the LCD module to operate during Sleep. Setting SLPEN and correctly executing the LCD module shutdown will disable the LCD module during Sleep and save power.

If a SLEEP instruction is executed and SLPEN = 1, the LCD module will immediately cease all functions, drive the outputs to VSS and go into a very low-current mode. The SLEEP instruction should only be executed after the LCD module has been disabled and the current cycle completed, thus ensuring that there are no DC voltages on the glass. To disable the LCD module, clear the LCDEN bit. The LCD module will complete the disabling process after the current frame, clear the LCDA bit and optionally cause an interrupt.

The steps required to properly enter Sleep with the LCD disabled are:

- Clear LCDEN
- Wait for LCDA = 0 either by polling or by interrupt
- Execute SLEEP

If SLPEN = 0 and SLEEP is executed while the LCD module clock source is FOSC/4, then the LCD module will halt with the pin driving the last LCD voltage pattern. Prolonged exposure to a fixed LCD voltage pattern will cause damage to the LCD glass. To prevent LCD glass damage, either perform the proper LCD module shutdown prior to Sleep, or change the LCD module clock to allow the LCD module to continue operation during Sleep.

If a SLEEP instruction is executed and SLPEN = 0 and the LCD module clock is either T1OSC or LFINTOSC, the module will continue to display the current contents of the LCDDATA registers. While in Sleep, the LCD data cannot be changed. If the LCDIE bit is set, the device will wake from Sleep on the next LCD frame boundary. The LCD module current consumption will not decrease in this mode; however, the overall device power consumption will be lower due to the shutdown of the CPU and other peripherals.

Table 27-8 shows the status of the LCD module during a Sleep while using each of the three available clock sources.

| | |
|---|---|
| **Note:** | When the LCDEN bit is cleared, the LCD module will be disabled at the completion of frame. At this time, the port pins will revert to digital functionality. To minimize power consumption due to floating digital inputs, the LCD pins should be driven low using the PORT and TRIS registers. |

If a SLEEP instruction is executed and SLPEN = 0, the module will continue to display the current contents of the LCDDATA registers. To allow the module to continue operation while in Sleep, the clock source must be either the LFINTOSC or T1OSC external oscillator. While in Sleep, the LCD data cannot be changed. The LCD module current consumption will not decrease in this mode; however, the overall consumption of the device will be lower due to shut down of the core and other peripheral functions.

Table 27-8 below shows the status of the LCD module during Sleep, using each of the three available clock sources.

**TABLE 27-8: LCD MODULE STATUS DURING SLEEP**

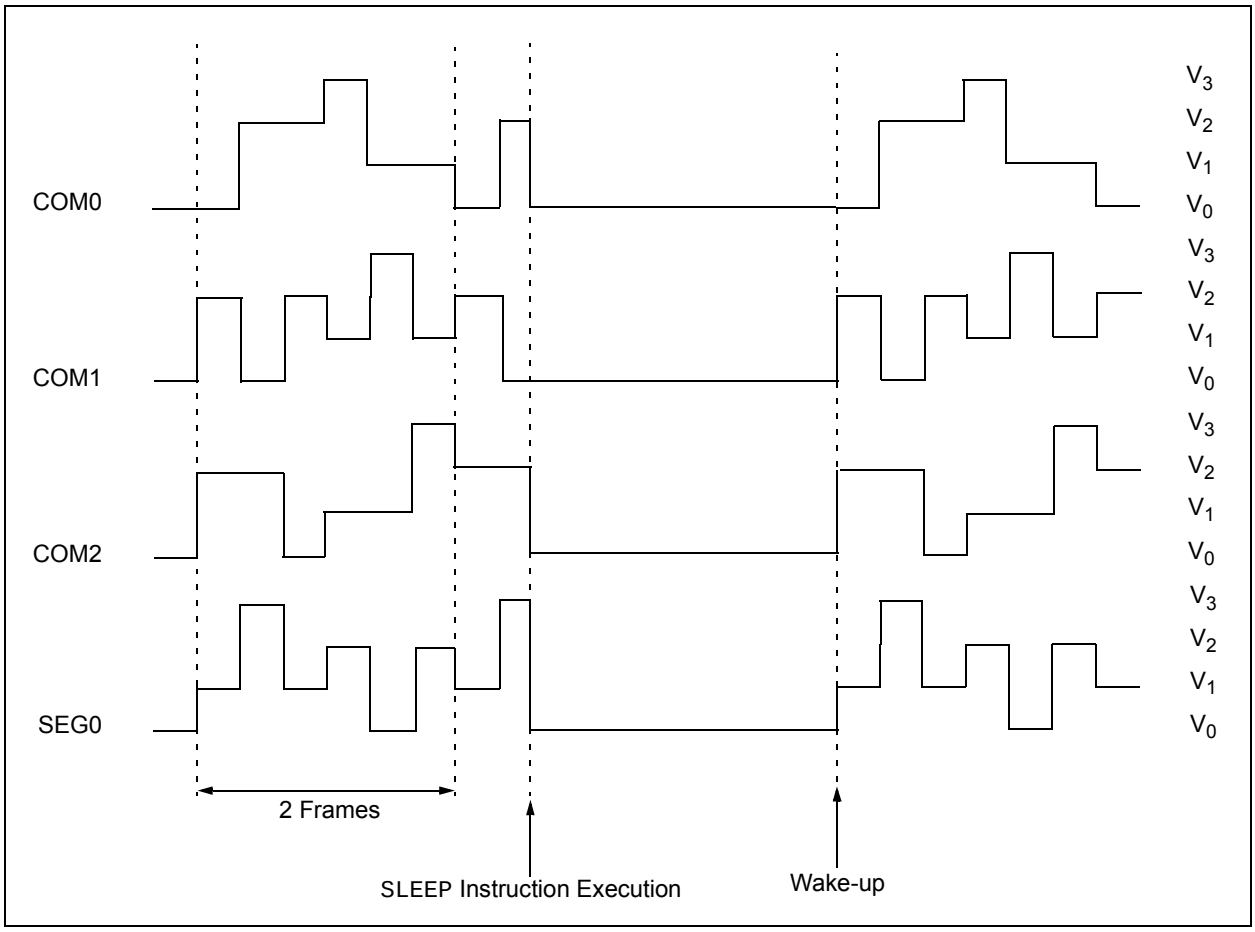
| Clock Source | SLPEN | Operational During Sleep |
|---|---|---|
| T1OSC | 0 | Yes |
| | 1 | No |
| LFINTOSC | 0 | Yes |
| | 1 | No |
| FOSC/4 | 0 | No |
| | 1 | No |

| | |
|---|---|
| **Note:** | The LFINTOSC or external T1OSC oscillator must be used to operate the LCD module during Sleep. |

If LCD interrupts are being generated (Type-B waveform with a multiplex mode not static) and LCDIE = 1, the device will awaken from Sleep on the next frame boundary.


© 2010-2016 Microchip Technology Inc.

**FIGURE 27-20: SLEEP ENTRY/EXIT WHEN SLPEN = 1**



## 27.14 Configuring the LCD Module

The following is the sequence of steps to configure the LCD module.

1. Select the frame clock prescale using bits LP<3:0> of the LCDPS register.
2. Configure the appropriate pins to function as segment drivers using the LCDSEn registers.
3. Configure the LCD module for the following using the LCDCON register:
   - Multiplex and Bias mode, bits LMUX<1:0>
   - Timing source, bits CS<1:0>
   - Sleep mode, bit SLPEN
4. Write initial values to pixel data registers, LCDDATA0 through LCDDATA23.
5. Clear LCD Interrupt Flag, LCDIF bit of the PIR2 register and if desired, enable the interrupt by setting bit LCDIE of the PIE2 register.
6. Configure bias voltages by setting the LCDRL, LCDREF and the associated ANSELx registers as needed.
7. Enable the LCD module by setting bit LCDEN of the LCDCON register.

## 27.15 Disabling the LCD Module

To disable the LCD module, write all '0's to the LCDCON register.

## 27.16 LCD Current Consumption

When using the LCD module, the current consumption consists of the following three factors:

- Oscillator Selection
- LCD Bias Source
- Capacitance of the LCD segments

The current consumption of the LCD module only can be considered negligible compared to these other factors.

### 27.16.1 OSCILLATOR SELECTION

The current consumed by the clock source selected must be considered when using the LCD module. See **Section 30.0 "Electrical Specifications"** for oscillator current consumption information.

### 27.16.2 LCD BIAS SOURCE

The LCD bias source, internal or external, can contribute significantly to the current consumption. Use the highest possible resistor values while maintaining contrast to minimize current.

### 27.16.3 CAPACITANCE OF THE LCD SEGMENTS

The LCD segments which can be modeled as capacitors must be both charged and discharged every frame. The size of the LCD segment and its technology determine the segment's capacitance.


© 2010-2016 Microchip Technology Inc.

**TABLE 27-9: SUMMARY OF REGISTERS ASSOCIATED WITH LCD OPERATION**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 90 |
| LCDCON | LCDEN | SLPEN | WERR | — | CS<1:0> | | LMUX<1:0> | | 326 |
| LCDCST | — | — | — | — | — | LCDCST<2:0> | | | 329 |
| LCDDATA0 | SEG7 COM0 | SEG6 COM0 | SEG5 COM0 | SEG4 COM0 | SEG3 COM0 | SEG2 COM0 | SEG1 COM0 | SEG0 COM0 | 330 |
| LCDDATA1 | SEG15 COM0 | SEG14 COM0 | SEG13 COM0 | SEG12 COM0 | SEG11 COM0 | SEG10 COM0 | SEG9 COM0 | SEG8 COM0 | 330 |
| LCDDATA2 | SEG23 COM0 | SEG22 COM0 | SEG21 COM0 | SEG20 COM0 | SEG19 COM0 | SEG18 COM0 | SEG17 COM0 | SEG16 COM0 | 330 |
| LCDDATA3 | SEG7 COM1 | SEG6 COM1 | SEG5 COM1 | SEG4 COM1 | SEG3 COM1 | SEG2 COM1 | SEG1 COM1 | SEG0 COM1 | 330 |
| LCDDATA4 | SEG15 COM1 | SEG14 COM1 | SEG13 COM1 | SEG12 COM1 | SEG11 COM1 | SEG10 COM1 | SEG9 COM1 | SEG8 COM1 | 330 |
| LCDDATA5 | SEG23 COM1 | SEG22 COM1 | SEG21 COM1 | SEG20 COM1 | SEG19 COM1 | SEG18 COM1 | SEG17 COM1 | SEG16 COM1 | 330 |
| LCDDATA6 | SEG7 COM2 | SEG6 COM2 | SEG5 COM2 | SEG4 COM2 | SEG3 COM2 | SEG2 COM2 | SEG1 COM2 | SEG0 COM2 | 330 |
| LCDDATA7 | SEG15 COM2 | SEG14 COM2 | SEG13 COM2 | SEG12 COM2 | SEG11 COM2 | SEG10 COM2 | SEG9 COM2 | SEG8 COM2 | 330 |
| LCDDATA8 | SEG23 COM2 | SEG22 COM2 | SEG21 COM2 | SEG20 COM2 | SEG19 COM2 | SEG18 COM2 | SEG17 COM2 | SEG16 COM2 | 330 |
| LCDDATA9 | SEG7 COM3 | SEG6 COM3 | SEG5 COM3 | SEG4 COM3 | SEG3 COM3 | SEG2 COM3 | SEG1 COM3 | SEG0 COM3 | 330 |
| LCDDATA10 | SEG15 COM3 | SEG14 COM3 | SEG13 COM3 | SEG12 COM3 | SEG11 COM3 | SEG10 COM3 | SEG9 COM3 | SEG8 COM3 | 330 |
| LCDDATA11 | SEG23 COM3 | SEG22 COM3 | SEG21 COM3 | SEG20 COM3 | SEG19 COM3 | SEG18 COM3 | SEG17 COM3 | SEG16 COM3 | 330 |
| LCDDATA12 | SEG31 COM0 | SEG30 COM0 | SEG29 COM0 | SEG28 COM0 | SEG27 COM0 | SEG26 COM0 | SEG25 COM0 | SEG24 COM0 | 330 |
| LCDDATA13 | SEG39 COM0 | SEG38 COM0 | SEG37 COM0 | SEG36 COM0 | SEG35 COM0 | SEG34 COM0 | SEG33 COM0 | SEG32 COM0 | 330 |
| LCDDATA14 | — | — | SEG45 COM0 | SEG44 COM0 | SEG43 COM0 | SEG42 COM0 | SEG41 COM0 | SEG40 COM0 | 330 |
| LCDDATA15 | SEG31 COM1 | SEG30 COM1 | SEG29 COM1 | SEG28 COM1 | SEG27 COM1 | SEG26 COM1 | SEG25 COM1 | SEG24 COM1 | 330 |
| LCDDATA16 | SEG39 COM1 | SEG38 COM1 | SEG37 COM1 | SEG36 COM1 | SEG35 COM1 | SEG34 COM1 | SEG33 COM1 | SEG32 COM1 | 330 |
| LCDDATA17 | — | — | SEG45 COM1 | SEG44 COM1 | SEG43 COM1 | SEG42 COM1 | SEG41 COM1 | SEG40 COM1 | 330 |
| LCDDATA18 | SEG31 COM2 | SEG30 COM2 | SEG29 COM2 | SEG28 COM2 | SEG27 COM2 | SEG26 COM2 | SEG25 COM2 | SEG24 COM2 | 330 |
| LCDDATA19 | SEG39 COM2 | SEG38 COM2 | SEG37 COM2 | SEG36 COM2 | SEG35 COM2 | SEG34 COM2 | SEG33 COM2 | SEG32 COM2 | 330 |
| LCDDATA20 | — | — | SEG45 COM2 | SEG44 COM2 | SEG43 COM2 | SEG42 COM2 | SEG41 COM2 | SEG40 COM2 | 330 |
| LCDDATA21 | SEG31 COM3 | SEG30 COM3 | SEG29 COM3 | SEG28 COM3 | SEG27 COM3 | SEG26 COM3 | SEG25 COM3 | SEG24 COM3 | 330 |

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

**TABLE 27-9: SUMMARY OF REGISTERS ASSOCIATED WITH LCD OPERATION (CONTINUED)**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| LCDDATA22 | SEG39 COM3 | SEG38 COM3 | SEG37 COM3 | SEG36 COM3 | SEG35 COM3 | SEG34 COM3 | SEG33 COM3 | SEG32 COM3 | 330 |
| LCDDATA23 | — | — | SEG45 COM3 | SEG44 COM3 | SEG43 COM3 | SEG42 COM3 | SEG41 COM3 | SEG40 COM3 | 330 |
| LCDPS | WFT | BIASMD | LCDA | WA | LP<3:0> | | | | 327 |
| LCDREF | LCDIRE | LCDIRS | LCDIRI | — | VLCD3PE | VLCD2PE | VLCD1PE | — | 328 |
| LCDRL | LRLAP<1:0> | | LRLBP<1:0> | | — | LRLAT<2:0> | | | 337 |
| LCDSE0 | SE<7:0> | | | | | | | | 330 |
| LCDSE1 | SE<15:8> | | | | | | | | 330 |
| LCDSE2 | SE<23:16> | | | | | | | | 330 |
| LCDSE3 | SE<31:24> | | | | | | | | 330 |
| LCDSE4 | SE<39:32> | | | | | | | | 330 |
| LCDSE5 | — | — | SE<45:40> | | | | | | 330 |
| PIE2 | OSFIE | C2IE | C1IE | EEIE | BCLIE | LCDIE | C3IE | CCP2IE | 92 |
| PIR2 | OSFIF | C2IF | C1IF | EEIF | BCLIF | LCDIF | C3IF | CCP2IF | 96 |
| T1CON | TMR1CS<1:0> | | T1CKPS<1:0> | | T1OSCEN | $\overline{\text{T1SYNC}}$ | — | TMR1ON | 197 |

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


© 2010-2016 Microchip Technology Inc.

## 28.0 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
- $\overline{\text{MCLR}}/V_{PP}$
- $V_{DD}$
- $V_{SS}$

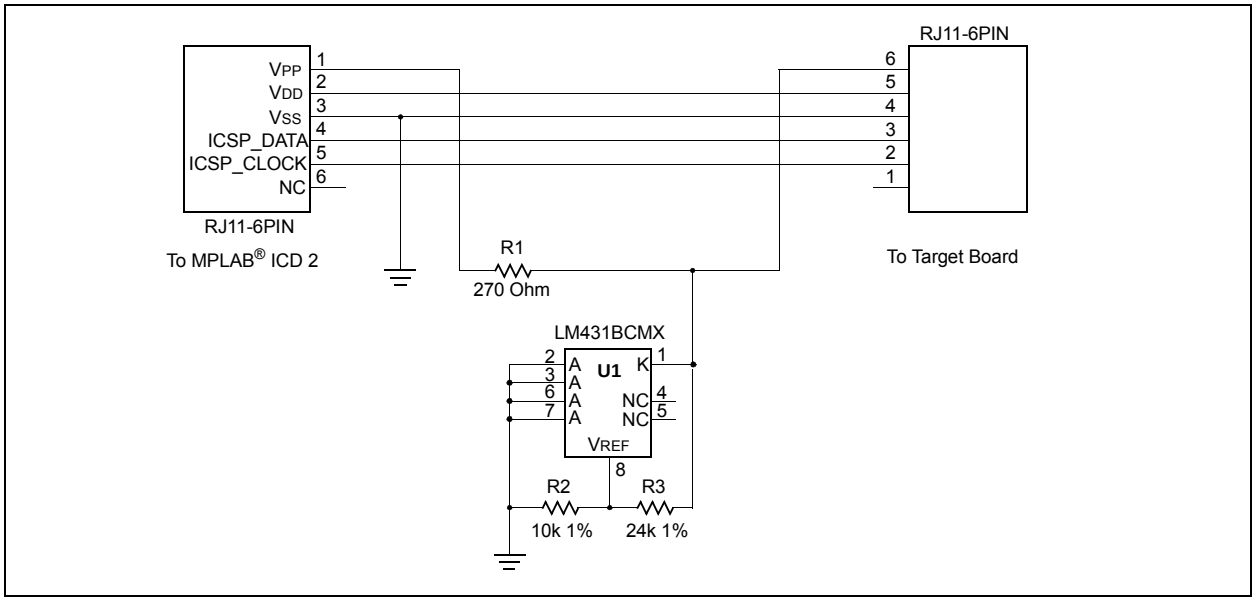
In Program/Verify mode the program memory, User IDs and the Configuration Words are programmed through serial communications. The ICSPDAT pin is a bidirectional I/O used for transferring the serial data and the ICSPCLK pin is the clock input. For more information on ICSP™ refer to the *PIC16F193X/LF193X/PIC16F194X/LF194X/PIC16LF190X Memory Programming Specification* (DS41397).

### 28.1 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 $\overline{\text{MCLR}}/V_{PP}$ to $V_{IHH}$.

Some programmers produce $V_{PP}$ greater than $V_{IHH}$ (9.0V), an external circuit is required to limit the $V_{PP}$ voltage. See Figure 28-1 for example circuit.

**FIGURE 28-1: $V_{PP}$ LIMITER EXAMPLE CIRCUIT**




| **Note:** | The MPLAB® ICD 2 produces a $V_{PP}$ voltage greater than the maximum $V_{PP}$ specification of the PIC16(L)F1946/47. |
|---|---|

## 28.2 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 low-voltage ICSP programming entry is enabled. To disable the Low-Voltage ICSP mode, the LVP bit must be programmed to '0'.

Entry into the Low-Voltage Programming Entry mode requires the following steps:

1. $\overline{\text{MCLR}}$ is brought to VIL.
2. A 32-bit key sequence is presented on ICSPDAT, while clocking ICSPCLK.

Once the key sequence is complete, $\overline{\text{MCLR}}$ must be held at VIL for as long as Program/Verify mode is to be maintained.

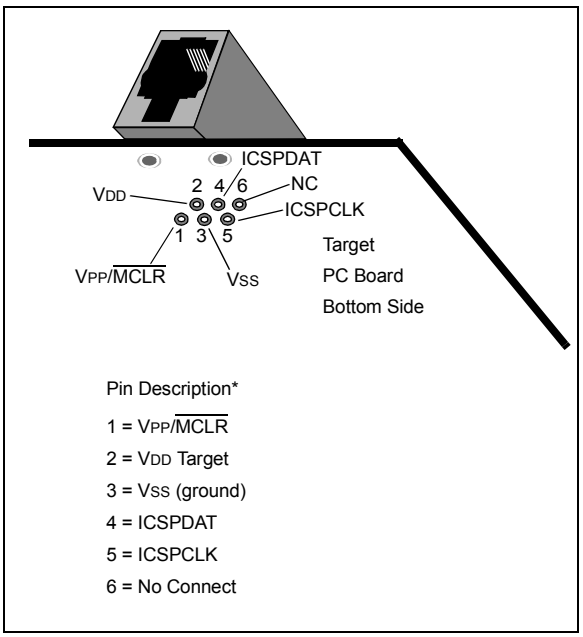
If low-voltage programming is enabled (LVP = 1), the $\overline{\text{MCLR}}$ Reset function is automatically enabled and cannot be disabled. See **Section 6.4 "MCLR"** for more information.

The LVP bit can only be reprogrammed to '0' by using the High-Voltage Programming mode.

## 28.3 Common Programming Interfaces

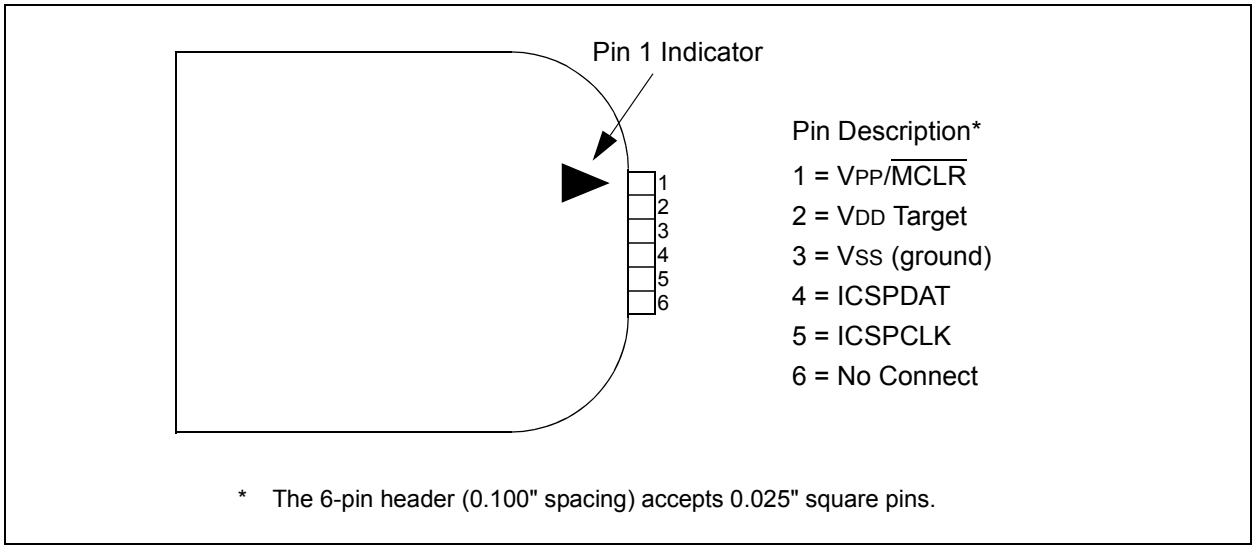
Connection to a target device is typically done through an ICSP™ header. A commonly found connector on development tools is the RJ-11 in the 6P6C (6-pin, 6 connector) configuration. See Figure 28-2.

**FIGURE 28-2: 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 28-3.

**FIGURE 28-3: PICKit™ PROGRAMMER STYLE CONNECTOR INTERFACE**



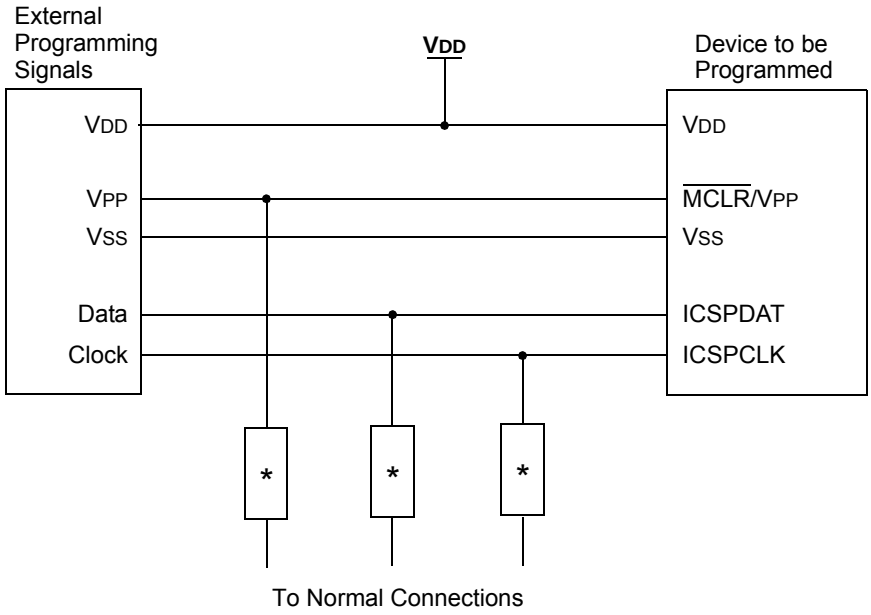


© 2010-2016 Microchip Technology Inc.

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 28-4 for more information.

**FIGURE 28-4: TYPICAL CONNECTION FOR ICSP™ PROGRAMMING**




* Isolation devices (as required).

## 29.0 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 29-3 lists the instructions recognized by the MPASM™ assembler.

All instructions are executed within a single instruction cycle, with the following exceptions, which may take two or three cycles:

- Subroutine takes two cycles (CALL, CALLW)
- Returns from interrupts or subroutines take two cycles (RETURN, RETLW, RETFIE)
- Program branching takes two cycles (GOTO, BRA, BRW, BTFSS, BTFSC, DECFSZ, INCSFZ)
- One additional instruction cycle will be used when any instruction references an indirect file register and the file select register is pointing to program memory.

One instruction cycle consists of four oscillator cycles; for an oscillator frequency of 4 MHz, this gives a nominal instruction execution rate of 1 MHz.

All instruction examples use the format '0xhh' to represent a hexadecimal number, where 'h' signifies a hexadecimal digit.

### 29.1 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 29-1: OPCODE FIELD DESCRIPTIONS**

| Field | Description |
|---|---|
| f | Register file address (0x00 to 0x7F) |
| W | Working register (accumulator) |
| b | Bit address within an 8-bit file register |
| k | Literal field, constant data or label |
| x | Don't care location (= 0 or 1).<br>The assembler will generate code with x = 0.<br>It is the recommended form of use for compatibility with all Microchip software tools. |
| d | Destination select; d = 0: store result in W,<br>d = 1: store result in file register f.<br>Default is d = 1. |
| n | FSR or INDF number. (0-1) |
| mm | Pre-post increment-decrement mode selection |

**TABLE 29-2: 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 |


© 2010-2016 Microchip Technology Inc.

**FIGURE 29-1: GENERAL FORMAT FOR INSTRUCTIONS**

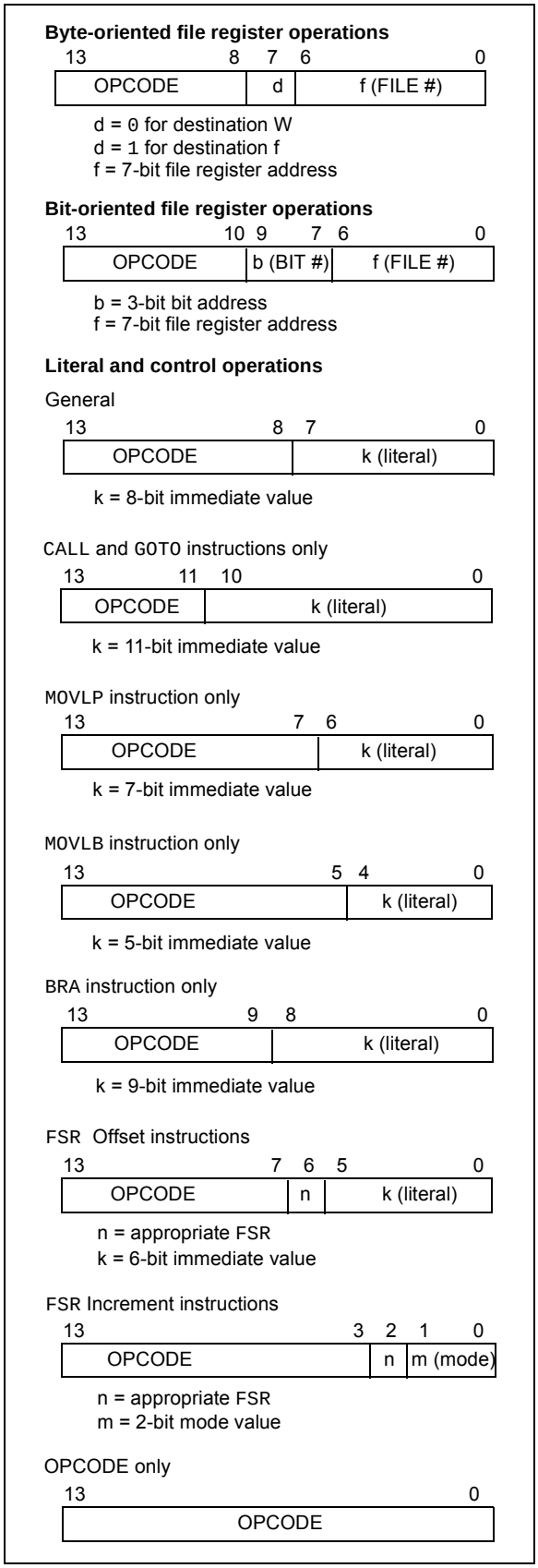


**TABLE 29-3: PIC16(L)F1946/47 INSTRUCTION SET**

| Mnemonic, Operands | | Description | Cycles | 14-Bit Opcode MSb | | | LSb | Status Affected | Notes |
|---|---|---|---|---|---|---|---|---|---|
| **BYTE-ORIENTED FILE REGISTER OPERATIONS** | | | | | | | | | |
| ADDWF | f, d | Add W and f | 1 | 00 | 0111 | dfff | ffff | C, DC, Z | 2 |
| ADDWFC | f, d | Add with Carry W and f | 1 | 11 | 1101 | dfff | ffff | C, DC, Z | 2 |
| ANDWF | f, d | AND W with f | 1 | 00 | 0101 | dfff | ffff | Z | 2 |
| ASRF | f, d | Arithmetic Right Shift | 1 | 11 | 0111 | dfff | ffff | C, Z | 2 |
| LSLF | f, d | Logical Left Shift | 1 | 11 | 0101 | dfff | ffff | C, Z | 2 |
| LSRF | f, d | Logical Right Shift | 1 | 11 | 0110 | dfff | ffff | C, Z | 2 |
| CLRF | f | Clear f | 1 | 00 | 0001 | lfff | ffff | Z | 2 |
| CLRW | – | Clear W | 1 | 00 | 0001 | 0000 | 00xx | Z | |
| COMF | f, d | Complement f | 1 | 00 | 1001 | dfff | ffff | Z | 2 |
| DECF | f, d | Decrement f | 1 | 00 | 0011 | dfff | ffff | Z | 2 |
| INCF | f, d | Increment f | 1 | 00 | 1010 | dfff | ffff | Z | 2 |
| IORWF | f, d | Inclusive OR W with f | 1 | 00 | 0100 | dfff | ffff | Z | 2 |
| MOVF | f, d | Move f | 1 | 00 | 1000 | dfff | ffff | Z | 2 |
| MOVWF | f | Move W to f | 1 | 00 | 0000 | 1fff | ffff | | 2 |
| RLF | f, d | Rotate Left f through Carry | 1 | 00 | 1101 | dfff | ffff | C | 2 |
| RRF | f, d | Rotate Right f through Carry | 1 | 00 | 1100 | dfff | ffff | C | 2 |
| SUBWF | f, d | Subtract W from f | 1 | 00 | 0010 | dfff | ffff | C, DC, Z | 2 |
| SUBWFB | f, d | Subtract with Borrow W from f | 1 | 11 | 1011 | dfff | ffff | C, DC, Z | 2 |
| SWAPF | f, d | Swap nibbles in f | 1 | 00 | 1110 | dfff | ffff | | 2 |
| XORWF | f, d | Exclusive OR W with f | 1 | 00 | 0110 | dfff | ffff | Z | 2 |
| **BYTE ORIENTED SKIP OPERATIONS** | | | | | | | | | |
| DECFSZ | f, d | Decrement f, Skip if 0 | 1(2) | 00 | 1011 | dfff | ffff | | 1, 2 |
| INCFSZ | f, d | Increment f, Skip if 0 | 1(2) | 00 | 1111 | dfff | ffff | | 1, 2 |
| **BIT-ORIENTED FILE REGISTER OPERATIONS** | | | | | | | | | |
| BCF | f, b | Bit Clear f | 1 | 01 | 00bb | bfff | ffff | | 2 |
| BSF | f, b | Bit Set f | 1 | 01 | 01bb | bfff | ffff | | 2 |
| **BIT-ORIENTED SKIP OPERATIONS** | | | | | | | | | |
| BTFSC | f, b | Bit Test f, Skip if Clear | 1 (2) | 01 | 10bb | bfff | ffff | | 1, 2 |
| BTFSS | f, b | Bit Test f, Skip if Set | 1 (2) | 01 | 11bb | bfff | ffff | | 1, 2 |
| **LITERAL OPERATIONS** | | | | | | | | | |
| ADDLW | k | Add literal and W | 1 | 11 | 1110 | kkkk | kkkk | C, DC, Z | |
| ANDLW | k | AND literal with W | 1 | 11 | 1001 | kkkk | kkkk | Z | |
| IORLW | k | Inclusive OR literal with W | 1 | 11 | 1000 | kkkk | kkkk | Z | |
| MOVLB | k | Move literal to BSR | 1 | 00 | 0000 | 001k | kkkk | | |
| MOVLP | k | Move literal to PCLATH | 1 | 11 | 0001 | 1kkk | kkkk | | |
| MOVLW | k | Move literal to W | 1 | 11 | 0000 | kkkk | kkkk | | |
| SUBLW | k | Subtract W from literal | 1 | 11 | 1100 | kkkk | kkkk | C, DC, Z | |
| XORLW | k | Exclusive OR literal with W | 1 | 11 | 1010 | kkkk | kkkk | Z | |

**Note 1:** If the Program Counter (PC) is modified, or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a NOP.

**2:** If this instruction addresses an INDF register and the MSb of the corresponding FSR is set, this instruction will require one additional instruction cycle.


© 2010-2016 Microchip Technology Inc.

**TABLE 29-3: PIC16(L)F1946/47 ENHANCED INSTRUCTION SET (CONTINUED)**

| Mnemonic, Operands | | Description | Cycles | 14-Bit Opcode MSb LSb | Status Affected | Notes |
|---|---|---|---|---|---|---|
| **CONTROL OPERATIONS** | | | | | | |
| BRA | k | Relative Branch | 2 | 11 001k kkkk kkkk | | |
| BRW | – | Relative Branch with W | 2 | 00 0000 0000 1011 | | |
| CALL | k | Call Subroutine | 2 | 10 0kkk kkkk kkkk | | |
| CALLW | – | Call Subroutine with W | 2 | 00 0000 0000 1010 | | |
| GOTO | k | Go to address | 2 | 10 1kkk kkkk kkkk | | |
| RETFIE | k | Return from interrupt | 2 | 00 0000 0000 1001 | | |
| RETLW | k | Return with literal in W | 2 | 11 0100 kkkk kkkk | | |
| RETURN | – | Return from Subroutine | 2 | 00 0000 0000 1000 | | |
| **INHERENT OPERATIONS** | | | | | | |
| CLRWDT | – | Clear Watchdog Timer | 1 | 00 0000 0110 0100 | $\overline{TO}$, $\overline{PD}$ | |
| 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 | | |
| **C-COMPILER OPTIMIZED** | | | | | | |
| ADDFSR | n, k | Add Literal k to FSRn | 1 | 11 0001 0nkk kkkk | | |
| MOVIW | n mm | Move Indirect FSRn to W with pre/post inc/dec modifier, mm | 1 | 00 0000 0001 0nmm | Z | **2, 3** |
| | k[n] | Move INDFn to W, Indexed Indirect. | 1 | 11 1111 0nkk kkkk | Z | **2** |
| MOVWI | n mm | Move W to Indirect FSRn with pre/post inc/dec modifier, mm | 1 | 00 0000 0001 1nmm | | **2, 3** |
| | k[n] | Move W to INDFn, Indexed Indirect. | 1 | 11 1111 1nkk kkkk | | **2** |

**Note 1:** If the Program Counter (PC) is modified, or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a NOP.

**2:** If this instruction addresses an INDF register and the MSb of the corresponding FSR is set, this instruction will require one additional instruction cycle.

**3:** See Table in the MOVIW and MOVWI instruction descriptions.

## 29.2 Instruction Descriptions

**ADDFSR — Add Literal to FSRn**

| | |
|---|---|
| Syntax: | [ *label* ] ADDFSR FSRn, k |
| Operands: | -32 ≤ k ≤ 31<br>n ∈ [ 0, 1] |
| Operation: | FSR(n) + k → FSR(n) |
| Status Affected: | None |
| Description: | The signed 6-bit literal 'k' is added to the contents of the FSRnH:FSRnL register pair.<br><br>FSRn is limited to the range 0000h - FFFFh. Moving beyond these bounds will cause the FSR to wrap around. |

**ADDLW — Add literal and W**

| | |
|---|---|
| Syntax: | [ *label* ] ADDLW k |
| Operands: | 0 ≤ k ≤ 255 |
| Operation: | (W) + k → (W) |
| Status Affected: | C, DC, Z |
| Description: | The contents of the W register are added to the 8-bit literal 'k' and the result is placed in the W register. |

**ADDWF — Add W and f**

| | |
|---|---|
| Syntax: | [ *label* ] ADDWF f,d |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (W) + (f) → (destination) |
| Status Affected: | C, DC, Z |
| Description: | Add the contents of the W register with register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'. |

**ADDWFC — ADD W and CARRY bit to f**

| | |
|---|---|
| Syntax: | [ *label* ] ADDWFC f {,d} |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (W) + (f) + (C) → dest |
| Status Affected: | C, DC, Z |
| Description: | Add W, the Carry flag and data memory location 'f'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed in data memory location 'f'. |

**ANDLW — AND literal with W**

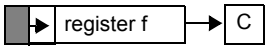
| | |
|---|---|
| Syntax: | [ *label* ] ANDLW k |
| Operands: | 0 ≤ k ≤ 255 |
| Operation: | (W) .AND. (k) → (W) |
| Status Affected: | Z |
| Description: | The contents of W register are AND'ed with the 8-bit literal 'k'. The result is placed in the W register. |

**ANDWF — AND W with f**

| | |
|---|---|
| Syntax: | [ *label* ] ANDWF f,d |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (W) .AND. (f) → (destination) |
| Status Affected: | Z |
| Description: | AND the W register with register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'. |

**ASRF — Arithmetic Right Shift**

| | |
|---|---|
| Syntax: | [ *label* ] ASRF f {,d} |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (f<7>)→ dest<7><br>(f<7:1>) → dest<6:0>,<br>(f<0>) → C, |
| Status Affected: | C, Z |
| Description: | The contents of register 'f' are shifted one bit to the right through the Carry flag. The MSb remains unchanged. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is stored back in register 'f'. |





© 2010-2016 Microchip Technology Inc.

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

| | |
|---|---|
| Syntax: | [ *label* ] BCF f,b |
| Operands: | 0 ≤ f ≤ 127<br>0 ≤ b ≤ 7 |
| Operation: | 0 → (f<b>) |
| Status Affected: | None |
| Description: | Bit 'b' in register 'f' is cleared. |

**BRA** **Relative Branch**

| | |
|---|---|
| Syntax: | [ *label* ] BRA label<br>[ *label* ] BRA $+k |
| Operands: | -256 ≤ label - PC + 1 ≤ 255<br>-256 ≤ k ≤ 255 |
| Operation: | (PC) + 1 + k → PC |
| Status Affected: | None |
| Description: | Add the signed 9-bit literal 'k' to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 1 + k. This instruction is a 2-cycle instruction. This branch has a limited range. |

**BRW** **Relative Branch with W**

| | |
|---|---|
| Syntax: | [ *label* ] BRW |
| Operands: | None |
| Operation: | (PC) + (W) → PC |
| Status Affected: | None |
| Description: | Add the contents of W (unsigned) to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 1 + (W). This instruction is a 2-cycle instruction. |

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

| | |
|---|---|
| Syntax: | [ *label* ] BSF f,b |
| Operands: | 0 ≤ f ≤ 127<br>0 ≤ b ≤ 7 |
| Operation: | 1 → (f<b>) |
| Status Affected: | None |
| Description: | Bit 'b' in register 'f' is set. |

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

| | |
|---|---|
| Syntax: | [ *label* ] BTFSC f,b |
| Operands: | 0 ≤ f ≤ 127<br>0 ≤ b ≤ 7 |
| Operation: | skip if (f<b>) = 0 |
| Status Affected: | None |
| Description: | If bit 'b' in register 'f' is '1', the next instruction is executed.<br>If bit 'b', in register 'f', is '0', the next instruction is discarded, and a NOP is executed instead, making this a 2-cycle instruction. |

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

| | |
|---|---|
| Syntax: | [ *label* ] BTFSS f,b |
| Operands: | 0 ≤ f ≤ 127<br>0 ≤ 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.<br>If bit 'b' is '1', then the next instruction is discarded and a NOP is executed instead, making this a 2-cycle instruction. |

**CALL** **Call Subroutine**

| | |
|---|---|
| Syntax: | [ *label* ] CALL k |
| Operands: | 0 ≤ k ≤ 2047 |
| Operation: | (PC)+ 1→ TOS,<br>k → PC<10:0>,<br>(PCLATH<6:3>) → PC<14:11> |
| Status Affected: | None |
| Description: | Call Subroutine. First, return address (PC + 1) is pushed onto the stack. The 11-bit immediate address is loaded into PC bits <10:0>. The upper bits of the PC are loaded from PCLATH. CALL is a 2-cycle instruction. |

**CALLW** **Subroutine Call With W**

| | |
|---|---|
| Syntax: | [ *label* ] CALLW |
| Operands: | None |
| Operation: | (PC) +1 → TOS,<br>(W) → PC<7:0>,<br>(PCLATH<6:0>) → PC<14:8> |
| Status Affected: | None |
| Description: | Subroutine call with W. First, the return address (PC + 1) is pushed onto the return stack. Then, the contents of W is loaded into PC<7:0>, and the contents of PCLATH into PC<14:8>. CALLW is a 2-cycle instruction. |

**CLRF** **Clear f**

| | |
|---|---|
| Syntax: | [ *label* ] CLRF f |
| Operands: | 0 ≤ f ≤ 127 |
| Operation: | 00h → (f)<br>1 → Z |
| Status Affected: | Z |
| Description: | The contents of register ‘f’ are cleared and the Z bit is set. |

**CLRW** **Clear W**

| | |
|---|---|
| Syntax: | [ *label* ] CLRW |
| Operands: | None |
| Operation: | 00h → (W)<br>1 → Z |
| Status Affected: | Z |
| Description: | W register is cleared. Zero bit (Z) is set. |

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

| | |
|---|---|
| Syntax: | [ *label* ] CLRWDT |
| Operands: | None |
| Operation: | 00h → WDT<br>0 → WDT prescaler,<br>1 → $\overline{TO}$<br>1 → $\overline{PD}$ |
| Status Affected: | $\overline{TO}$, $\overline{PD}$ |
| Description: | CLRWDT instruction resets the Watchdog Timer. It also resets the prescaler of the WDT.<br>Status bits $\overline{TO}$ and $\overline{PD}$ are set. |

**COMF** **Complement f**

| | |
|---|---|
| Syntax: | [ *label* ] COMF f,d |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | ($\bar{f}$) → (destination) |
| Status Affected: | Z |
| Description: | The contents of register ‘f’ are complemented. If ‘d’ is ‘0’, the result is stored in W. If ‘d’ is ‘1’, the result is stored back in register ‘f’. |

**DECF** **Decrement f**

| | |
|---|---|
| Syntax: | [ *label* ] DECF f,d |
| Operands: | 0 ≤ f ≤ 127<br>d ∈ [0,1] |
| Operation: | (f) - 1 → (destination) |
| Status Affected: | Z |
| Description: | Decrement register ‘f’. If ‘d’ is ‘0’, the result is stored in the W register. If ‘d’ is ‘1’, the result is stored back in register ‘f’. |


© 2010-2016 Microchip Technology Inc.

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

| | |
|---|---|
| Syntax: | [ *label* ] DECFSZ f,d |
| Operands: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| Operation: | (f) - 1 → (destination);<br>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'.<br>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: | $0 \le k \le 2047$ |
| Operation: | k → PC<10:0><br>PCLATH<6:3> → PC<14:11> |
| Status Affected: | None |
| Description: | GOTO is an unconditional branch. The 11-bit immediate value is loaded into PC bits <10:0>. The upper bits of PC are loaded from PCLATH<4:3>. GOTO is a 2-cycle instruction. |

**INCF** **Increment f**

| | |
|---|---|
| Syntax: | [ *label* ] INCF f,d |
| Operands: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| 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: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| Operation: | (f) + 1 → (destination),<br>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'.<br>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: | $0 \le k \le 255$ |
| 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: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| 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'. |

© 2010-2016 Microchip Technology Inc.

**LSLF** **Logical Left Shift**

Syntax: [ *label* ] LSLF f {,d}

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

Operation: (f<7>) → C
(f<6:0>) → dest<7:1>
0 → dest<0>

Status Affected: C, Z

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

**LSRF** **Logical Right Shift**

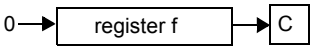
Syntax: [ *label* ] LSRF f {,d}

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

Operation: 0 → dest<7>
(f<7:1>) → dest<6:0>,
(f<0>) → C,

Status Affected: C, Z

Description: The contents of register 'f' are shifted one bit to the right through the Carry flag. A '0' is shifted into the MSb. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is stored back in register 'f'.




**MOVF** **Move f**

Syntax: [ *label* ] MOVF f,d

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

Operation: (f) → (dest)

Status Affected: Z

Description: The contents of register f is moved to a destination dependent upon the status of d. If d = 0, destination is W register. If d = 1, the destination is file register f itself. d = 1 is useful to test a file register since status flag Z is affected.

Words: 1

Cycles: 1

Example:

```
MOVF    FSR, 0
```

After Instruction
W = value in FSR register
Z = 1


© 2010-2016 Microchip Technology Inc.

**MOVIW** **Move INDFn to W**

Syntax: [ *label* ] MOVIW ++FSRn
[ *label* ] MOVIW --FSRn
[ *label* ] MOVIW FSRn++
[ *label* ] MOVIW FSRn--
[ *label* ] MOVIW k[FSRn]

Operands: n ∈ [0,1]
mm ∈ [00,01, 10, 11]
-32 ≤ k ≤ 31

Operation: INDFn → W
Effective address is determined by
- FSR + 1 (preincrement)
- FSR - 1 (predecrement)
- FSR + k (relative offset)

After the Move, the FSR value will be either:
- FSR + 1 (all increments)
- FSR - 1 (all decrements)
- Unchanged

Status Affected: Z

| Mode | Syntax | mm |
|---|---|---|
| Preincrement | ++FSRn | 00 |
| Predecrement | --FSRn | 01 |
| Postincrement | FSRn++ | 10 |
| Postdecrement | FSRn-- | 11 |

Description: This instruction is used to move data between W and one of the indirect registers (INDFn). Before/after this move, the pointer (FSRn) is updated by pre/post incrementing/decrementing it.

Note: The INDFn registers are not physical registers. Any instruction that accesses an INDFn register actually accesses the register at the address specified by the FSRn.

FSRn is limited to the range 0000h - FFFFh. Incrementing/decrementing it beyond these bounds will cause it to wrap around.

**MOVLB** **Move literal to BSR**

Syntax: [ *label* ] MOVLB k

Operands: 0 ≤ k ≤ 15

Operation: k → BSR

Status Affected: None

Description: The 5-bit literal 'k' is loaded into the Bank Select Register (BSR).

**MOVLP** **Move literal to PCLATH**

Syntax: [ *label* ] MOVLP k

Operands: 0 ≤ k ≤ 127

Operation: k → PCLATH

Status Affected: None

Description: The 7-bit literal 'k' is loaded into the PCLATH register.

**MOVLW** **Move literal to W**

Syntax: [ *label* ] MOVLW k

Operands: 0 ≤ k ≤ 255

Operation: k → (W)

Status Affected: None

Description: The 8-bit literal 'k' is loaded into W register. The "don't cares" will assemble as '0's.

Words: 1

Cycles: 1

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

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

Syntax: [ *label* ] MOVWF f

Operands: 0 ≤ f ≤ 127

Operation: (W) → (f)

Status Affected: None

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

Words: 1

Cycles: 1

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.

Words: 1

Cycles: 1

Example:
```
OPTION
```
Before Instruction
OPTION_REG = 0xFF
W = 0x4F
After Instruction
OPTION_REG = 0x4F
W = 0x4F

**RESET** **Software Reset**

Syntax: [ *label* ] RESET

Operands: None

Operation: Execute a device Reset. Resets the nRI flag of the PCON register.

Status Affected: None

Description: This instruction provides a way to execute a hardware Reset by software.


© 2010-2016 Microchip Technology Inc.

**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

**RETLW** **Return with literal in W**

Syntax: [ *label* ] RETLW k

Operands: 0 ≤ k ≤ 255

Operation: k → (W);
TOS → PC

Status Affected: None

Description: The W register is loaded with the 8-bit literal 'k'. The program counter is loaded from the top of the stack (the return address). This is a 2-cycle instruction.

Words: 1

Cycles: 2

Example:

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

TABLE

Before Instruction
W = 0x07
After Instruction
W = value of k8

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

Syntax: [ *label* ] RETURN

Operands: None

Operation: TOS → PC

Status Affected: None

Description: Return from subroutine. The stack is POPed and the top of the stack (TOS) is loaded into the program counter. This is a 2-cycle instruction.

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

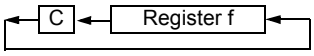
Syntax: [ *label* ] RLF f,d

Operands: 0 ≤ f ≤ 127
d ∈ [0,1]

Operation: See description below

Status Affected: C

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




Words: 1

Cycles: 1

Example:

```
RLF     REG1,0
```

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

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

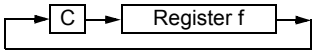
Syntax: [ *label* ] RRF f,d

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

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: $0 \le k \le 255$

Operation: k - (W) → (W)

Status Affected: C, DC, Z

Description: The W register is subtracted (2's complement method) from the 8-bit literal 'k'. The result is placed in the W register.

| | |
|---|---|
| C = 0 | W > k |
| C = 1 | W ≤ k |
| DC = 0 | W<3:0> > k<3:0> |
| DC = 1 | W<3:0> ≤ k<3:0> |

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

Syntax: [ *label* ] SUBWF f,d

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

Operation: (f) - (W) → (destination)

Status Affected: C, DC, Z

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

| | |
|---|---|
| C = 0 | W > f |
| C = 1 | W ≤ f |
| DC = 0 | W<3:0> > f<3:0> |
| DC = 1 | W<3:0> ≤ f<3:0> |

**SUBWFB** **Subtract W from f with Borrow**

Syntax: SUBWFB f {,d}

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

Operation: (f) – (W) – ($\overline{B}$) → dest

Status Affected: C, DC, Z

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


© 2010-2016 Microchip Technology Inc.

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

| | |
|---|---|
| Syntax: | [ *label* ] SWAPF f,d |
| Operands: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| Operation: | (f<3:0>) → (destination<7:4>),<br>(f<7:4>) → (destination<3:0>) |
| Status Affected: | None |
| Description: | The upper and lower nibbles of register 'f' are exchanged. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed in register 'f'. |

**TRIS** **Load TRIS Register with W**

| | |
|---|---|
| Syntax: | [ *label* ] TRIS f |
| Operands: | $5 \le f \le 7$ |
| Operation: | (W) → TRIS register 'f' |
| Status Affected: | None |
| Description: | Move data from W register to TRIS register.<br>When 'f' = 5, TRISA is loaded.<br>When 'f' = 6, TRISB is loaded.<br>When 'f' = 7, TRISC is loaded. |

**XORLW** **Exclusive OR literal with W**

| | |
|---|---|
| Syntax: | [ *label* ] XORLW k |
| Operands: | $0 \le k \le 255$ |
| Operation: | (W) .XOR. k → (W) |
| Status Affected: | Z |
| Description: | The contents of the W register are XOR'ed with the 8-bit literal 'k'. The result is placed in the W register. |

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

| | |
|---|---|
| Syntax: | [ *label* ] XORWF f,d |
| Operands: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| Operation: | (W) .XOR. (f) → (destination) |
| Status Affected: | Z |
| Description: | Exclusive OR the contents of the W register with register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'. |

## 30.0 ELECTRICAL SPECIFICATIONS

### Absolute Maximum Ratings(†)

Ambient temperature under bias ..... -40°C to +125°C
Storage temperature ..... -65°C to +150°C
Voltage on $V_{DD}$ with respect to $V_{SS}$, PIC16F1946/47 ..... -0.3V to +6.5V
Voltage on $V_{CAP}$ pin with respect to $V_{SS}$ ..... -0.3V to +4.0V
Voltage on $V_{DD}$ with respect to $V_{SS}$, PIC16LF1946/47 ..... -0.3V to +4.0V
Voltage on $\overline{MCLR}$ with respect to Vss ..... -0.3V to +9.0V
Voltage on all other pins with respect to $V_{SS}$ ..... -0.3V to ($V_{DD}$ + 0.3V)
Total power dissipation(1) ..... 800 mW
Maximum current out of $V_{SS}$ pin, -40°C ≤ $T_A$ ≤ +85°C for industrial ..... 350 mA
Maximum current out of $V_{SS}$ pin, -40°C ≤ $T_A$ ≤ +125°C for extended ..... 120 mA
Maximum current into $V_{DD}$ pin, -40°C ≤ $T_A$ ≤ +85°C for industrial ..... 350 mA
Maximum current into $V_{DD}$ pin, -40°C ≤ $T_A$ ≤ +125°C for extended ..... 120 mA
Clamp current, $I_K$ ($V_{PIN}$ < 0 or $V_{PIN}$ > $V_{DD}$) ..... ± 20 mA
Maximum output current sunk by any I/O pin ..... 50 mA
Maximum output current sourced by any I/O pin ..... 50 mA

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

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


© 2010-2016 Microchip Technology Inc.

## 30.1 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: $T_{A_MIN} \leq T_A \leq T_{A_MAX}$

**$V_{DD}$ — Operating Supply Voltage(1)**

PIC16LF1946/47

- $V_{DDMIN}$ ($F_{OSC} \leq$ 16 MHz) ..... +1.8V
- $V_{DDMIN}$ ($F_{OSC} \leq$ 32 MHz) ..... +2.5V
- $V_{DDMAX}$ ..... +3.6V

PIC16F1946/47

- $V_{DDMIN}$ ($F_{OSC} \leq$ 16 MHz) ..... +2.3V
- $V_{DDMIN}$ ($F_{OSC} \leq$ 32 MHz) ..... +2.5V
- $V_{DDMAX}$ ..... +5.5V

**$T_A$ — Operating Ambient Temperature Range**

Industrial Temperature

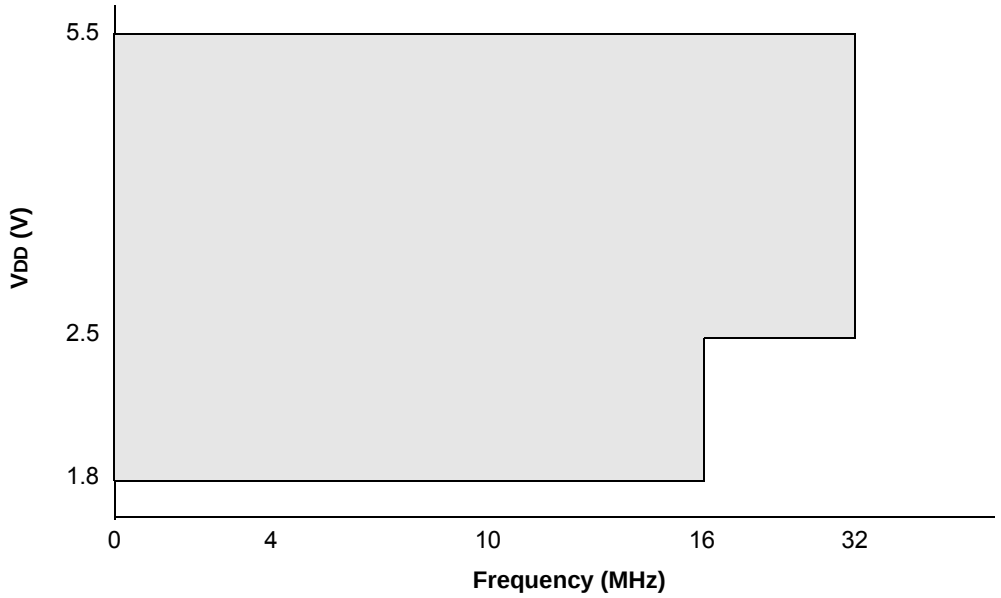
- $T_{A_MIN}$ ..... -40°C
- $T_{A_MAX}$ ..... +85°C

Extended Temperature

- $T_{A_MIN}$ ..... -40°C
- $T_{A_MAX}$ ..... +125°C

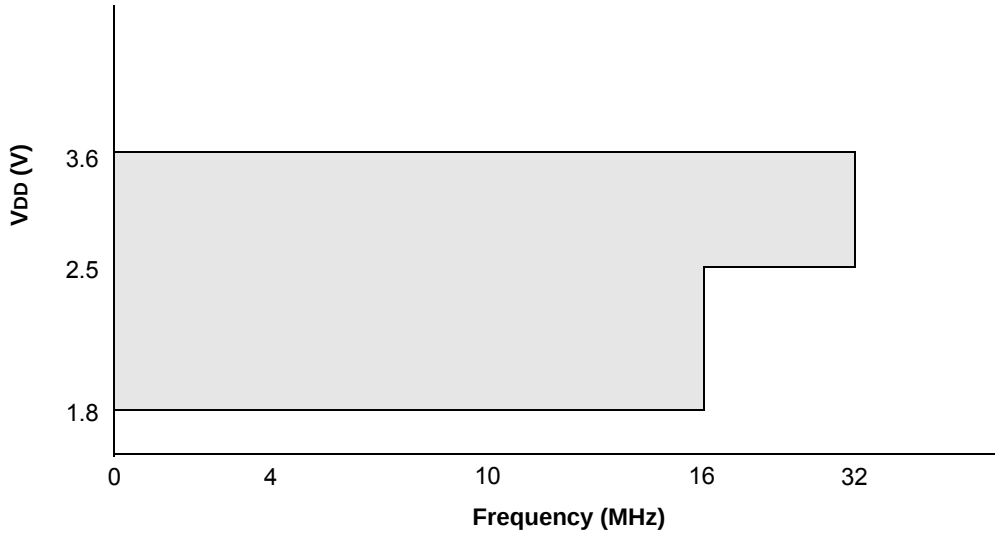
**Note 1:** See Parameter D001, DS Characteristics: Supply Voltage.

**FIGURE 30-1: PIC16F1946/47 VOLTAGE FREQUENCY GRAPH, -40°C ≤ TA ≤ +125°C**




**Note 1:** The shaded region indicates the permissible combinations of voltage and frequency.
**2:** Refer to Table 30-1 for each Oscillator mode's supported frequencies.

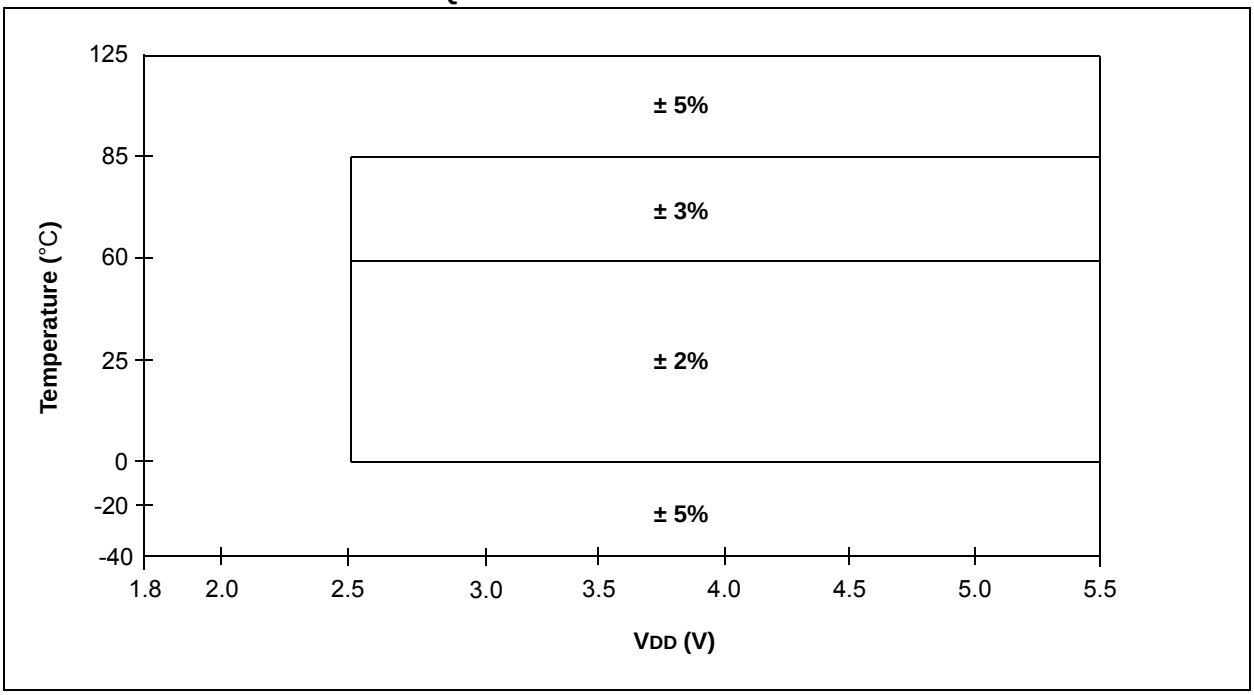
**FIGURE 30-2: PIC16LF1946/47 VOLTAGE FREQUENCY GRAPH, -40°C ≤ TA ≤ +125°C**




**Note 1:** The shaded region indicates the permissible combinations of voltage and frequency.
**2:** Refer to Table 30-1 for each Oscillator mode's supported frequencies.


© 2010-2016 Microchip Technology Inc.

**FIGURE 30-3: HFINTOSC FREQUENCY ACCURACY OVER DEVICE VDD AND TEMPERATURE**



## 30.2 DC Characteristics: PIC16(L)F1946/47-I/E (Industrial, Extended)

| PIC16LF1946/47 | | | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | |
|---|---|---|---|---|---|---|---|
| **PIC16F1946/47** | | | **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | |
| **Param. No.** | **Sym.** | **Characteristic** | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** |
| D001 | VDD | **Supply Voltage (VDDMIN, VDDMAX)** | | | | | |
| | | PIC16LF1946/47 | 1.8<br>2.5 | —<br>— | 3.6<br>3.6 | V<br>V | FOSC ≤ 16 MHz<br>FOSC ≤ 32 MHz (**NOTE 2**) |
| D001 | | PIC16F1946/47 | 1.8<br>2.5 | —<br>— | 5.5<br>5.5 | V<br>V | FOSC ≤ 16 MHz<br>FOSC ≤ 32 MHz (**NOTE 2**) |
| D002* | VDR | **RAM Data Retention Voltage(1)** | | | | | |
| | | PIC16LF1946/47 | 1.5 | — | — | V | Device in Sleep mode |
| | | PIC16F1946/47 | 1.7 | — | — | V | Device in Sleep mode |
| D002A* | VPOR* | **Power-on Reset Release Voltage** | | | | | |
| | | PIC16LF1946/47 | — | 1.6 | — | V | |
| | | PIC16F1946/47 | — | 1.6 | — | V | |
| D002B* | VPORR* | **Power-on Reset Rearm Voltage** | | | | | |
| | | PIC16LF1946/47 | — | 0.8 | — | V | Device in Sleep mode |
| | | PIC16F1946/47 | — | 1.5 | — | V | Device in Sleep mode |
| D003 | VADFVR | **Fixed Voltage Reference Voltage for ADC** | -8 | — | 6 | % | 1.024V, VDD ≥ 2.5V<br>2.048V, VDD ≥ 2.5V<br>4.096V, VDD ≥ 4.75V |
| D003A | VCDAFVR | **Fixed Voltage Reference Voltage for Comparator and DAC** | -11 | — | 7 | % | 1.024V, VDD ≥ 2.5V<br>2.048V, VDD ≥ 2.5V<br>4.096V, VDD ≥ 4.75V |
| D003B | VLCDFVR | **Fixed Voltage Reference Voltage for LCD Bias, Initial Accuracy** | -11 | — | 10 | % | 3.072V, VDD ≥ 3.6V |
| D004* | SVDD | **VDD Rise Rate** to ensure internal Power-on Reset signal | 0.05 | — | — | V/ms | See **Section 6.1 "Power-on Reset (POR)"** for details. |

* These parameters are characterized but not tested.

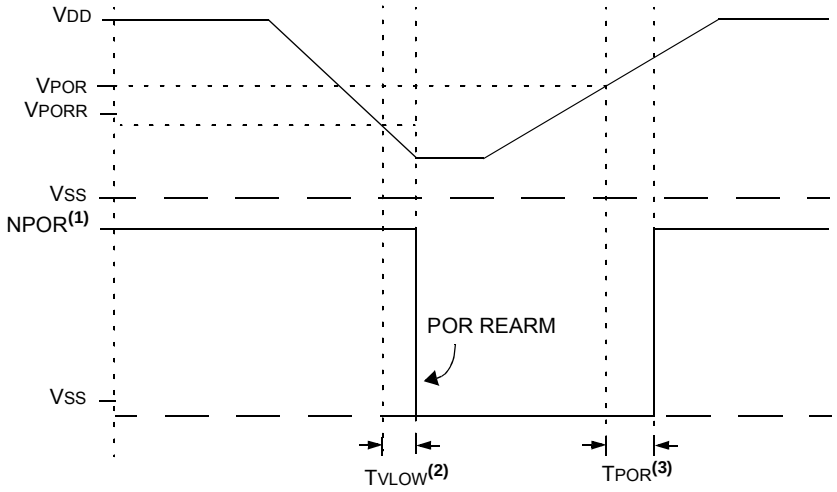
† Data in "Typ" column is at 3.3V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

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

**2:** PLL required for 32 MHz operation.


© 2010-2016 Microchip Technology Inc.

**FIGURE 30-4: POR AND POR REARM WITH SLOW RISING VDD**




**Note 1:** When NPOR is low, the device is held in Reset.
**2:** TPOR 1 μs typical.
**3:** TVLOW 2.7 μs typical.

## 30.3 DC Characteristics: PIC16(L)F1946/47-I/E (Industrial, Extended)

| PIC16LF1946/47 | | | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | |
|---|---|---|---|---|---|---|---|
| **PIC16F1946/47** | | | **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | |
| **Param No.** | **Device Characteristics** | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** | |
| | | | | | | **VDD** | **Note** |
| | **Supply Current (IDD)(1, 2)** | | | | | | |
| D009 | LDO Regulator | — | 350 | — | µA | — | HS, EC OR HFINTOSC Clock modes with VCAP pin disabled |
| | | — | 30 | — | µA | — | |
| | | — | 5 | — | µA | — | LP/LFINTOSC Clock mode or Sleep (requires FVR and BOR to be disabled) |
| D010 | | — | 5.0 | 11 | µA | 1.8 | FOSC = 32 kHz<br>LP Oscillator mode **(Note 4)**,<br>-40°C ≤ TA ≤ +85°C |
| | | — | 6.0 | 13 | µA | 3.0 | |
| D010 | | — | 24 | 53 | µA | 1.8 | FOSC = 32 kHz<br>LP Oscillator mode **(Note 4, 5)**,<br>-40°C ≤ TA ≤ +85°C |
| | | — | 30 | 58 | µA | 3.0 | |
| | | — | 32 | 63 | µA | 5.0 | |
| D010A | | — | 7.0 | 23 | µA | 1.8 | FOSC = 32 kHz<br>LP Oscillator mode **(Note 4)**<br>-40°C ≤ TA ≤ +125°C |
| | | — | 9.0 | 27 | µA | 3.0 | |
| D010A | | — | 24 | 68 | µA | 1.8 | FOSC = 32 kHz<br>LP Oscillator mode **(Note 4, 5)**<br>-40°C ≤ TA ≤ +125°C |
| | | — | 30 | 88 | µA | 3.0 | |
| | | — | 32 | 95 | µA | 5.0 | |
| D011 | | — | 60 | 105 | µA | 1.8 | FOSC = 1 MHz<br>XT Oscillator mode |
| | | — | 120 | 190 | µA | 3.0 | |
| D011 | | — | 95 | 130 | µA | 1.8 | FOSC = 1 MHz<br>XT Oscillator mode **(Note 5)** |
| | | — | 170 | 220 | µA | 3.0 | |
| | | — | 190 | 270 | µA | 5.0 | |
| D012 | | — | 160 | 300 | µA | 1.8 | FOSC = 4 MHz<br>XT Oscillator mode |
| | | — | 300 | 500 | µA | 3.0 | |
| D012 | | — | 200 | 330 | µA | 1.8 | FOSC = 4 MHz<br>XT Oscillator mode **(Note 5)** |
| | | — | 300 | 500 | µA | 3.0 | |
| | | — | 400 | 650 | µA | 5.0 | |

**Note 1:** The test conditions for all IDD measurements in active operation mode are: OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD; $\overline{MCLR}$ = VDD; WDT disabled.

**2:** The supply current is mainly a function of the operating voltage and frequency. Other factors, such as I/O pin loading and switching rate, oscillator type, internal code execution pattern and temperature, also have an impact on the current consumption.

**3:** For RC oscillator configurations, current through REXT is not included. The current through the resistor can be extended by the formula IR = VDD/2 REXT (mA) with REXT in kΩ.

**4:** FVR and BOR are disabled.

**5:** 0.1 µF capacitor on VCAP (RF0).

**6:** 8 MHz crystal oscillator with 4x PLL enabled.


© 2010-2016 Microchip Technology Inc.

## 30.3 DC Characteristics: PIC16(L)F1946/47-I/E (Industrial, Extended) (Continued)

| PIC16LF1946/47 | | | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | |
|---|---|---|---|---|---|---|---|
| **PIC16F1946/47** | | | **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | |
| **Param No.** | **Device Characteristics** | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** | |
| | | | | | | **VDD** | **Note** |
| | **Supply Current (IDD)(1, 2)** | | | | | | |
| D013 | | — | 15 | 40 | µA | 1.8 | FOSC = 500 kHz<br>EC Oscillator Low-Power mode |
| | | — | 30 | 75 | µA | 3.0 | |
| D013 | | — | 30 | 60 | µA | 1.8 | FOSC = 500 kHz<br>EC Oscillator Low-Power mode **(Note 5)** |
| | | — | 45 | 85 | µA | 3.0 | |
| | | — | 50 | 90 | µA | 5.0 | |
| D014 | | — | 140 | 250 | µA | 1.8 | FOSC = 4 MHz<br>EC Oscillator mode<br>Medium-Power mode |
| | | — | 270 | 400 | µA | 3.0 | |
| D014 | | — | 160 | 270 | µA | 1.8 | FOSC = 4 MHz<br>EC Oscillator mode **(Note 5)**<br>Medium-Power mode |
| | | — | 270 | 430 | µA | 3.0 | |
| | | — | 320 | 500 | µA | 5.0 | |
| D015 | | — | 2.0 | 3.2 | mA | 3.0 | FOSC = 32 MHz<br>EC Oscillator High-Power mode |
| | | — | 2.3 | 3.9 | mA | 3.6 | |
| D015 | | — | 2.0 | 3.2 | mA | 3.0 | FOSC = 32 MHz<br>EC Oscillator High-Power mode **(Note 5)** |
| | | — | 2.2 | 3.9 | mA | 5.0 | |
| D016 | | — | 3.0 | 11 | µA | 1.8 | FOSC = 32 kHz, LFINTOSC mode **(Note 4)**<br>-40°C ≤ TA ≤ +85°C |
| | | — | 5.0 | 13 | µA | 3.0 | |
| D016 | | — | 24 | 40 | µA | 1.8 | FOSC = 32 kHz, LFINTOSC mode **(Note 4, 5)**<br>-40°C ≤ TA ≤ +85°C |
| | | — | 30 | 48 | µA | 3.0 | |
| | | — | 32 | 58 | µA | 5.0 | |

**Note 1:** The test conditions for all IDD measurements in active operation mode are: OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD; $\overline{MCLR}$ = VDD; WDT disabled.

**2:** The supply current is mainly a function of the operating voltage and frequency. Other factors, such as I/O pin loading and switching rate, oscillator type, internal code execution pattern and temperature, also have an impact on the current consumption.

**3:** For RC oscillator configurations, current through REXT is not included. The current through the resistor can be extended by the formula IR = VDD/2 REXT (mA) with REXT in kΩ.

**4:** FVR and BOR are disabled.

**5:** 0.1 µF capacitor on VCAP (RF0).

**6:** 8 MHz crystal oscillator with 4x PLL enabled.

## 30.3 DC Characteristics: PIC16(L)F1946/47-I/E (Industrial, Extended) (Continued)

| PIC16LF1946/47 | | | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | |
|---|---|---|---|---|---|---|---|
| PIC16F1946/47 | | | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | |
| Param No. | Device Characteristics | Min. | Typ† | Max. | Units | Conditions | |
| | | | | | | VDD | Note |
| D017 | **Supply Current (IDD)(1, 2)** | | | | | | |
| | | — | 100 | 200 | µA | 1.8 | FOSC = 500 kHz<br>MFINTOSC mode |
| | | — | 120 | 230 | µA | 3.0 | |
| D017 | | — | 110 | 210 | µA | 1.8 | FOSC = 500 kHz<br>MFINTOSC mode (Note 5) |
| | | — | 120 | 240 | µA | 3.0 | |
| | | — | 160 | 290 | µA | 5.0 | |
| D018 | | — | 0.5 | 1.1 | mA | 1.8 | FOSC = 8 MHz<br>HFINTOSC mode |
| | | — | 0.8 | 1.6 | mA | 3.0 | |
| D018 | | — | 0.5 | 1.2 | mA | 1.8 | FOSC = 8 MHz<br>HFINTOSC mode (Note 5) |
| | | — | 0.8 | 1.7 | mA | 3.0 | |
| | | — | 0.9 | 1.8 | mA | 5.0 | |
| D019 | | — | 0.8 | 1.5 | mA | 1.8 | FOSC = 16 MHz<br>HFINTOSC mode |
| | | — | 1.2 | 2.3 | mA | 3.0 | |
| D019 | | — | 0.8 | 1.6 | mA | 1.8 | FOSC = 16 MHz<br>HFINTOSC mode (Note 5) |
| | | — | 1.2 | 2.4 | mA | 3.0 | |
| | | — | 1.4 | 2.5 | mA | 5.0 | |
| | | — | 2.1 | 3.6 | mA | 3.0 | FOSC = 32 MHz<br>HFINTOSC mode |
| | | — | 2.3 | 4.3 | mA | 3.6 | |
| | | — | 2.1 | 3.7 | mA | 3.0 | FOSC = 32 MHz<br>HFINTOSC mode |
| | | — | 2.2 | 4.1 | mA | 5.0 | |
| D020 | | — | 150 | 300 | µA | 1.8 | FOSC = 4 MHz<br>EXTRC mode (Note 3) |
| | | — | 270 | 500 | µA | 3.0 | |
| D020 | | — | 170 | 330 | µA | 1.8 | FOSC = 4 MHz<br>EXTRC mode (Note 3, Note 5) |
| | | — | 290 | 500 | µA | 3.0 | |
| | | — | 320 | 650 | µA | 5.0 | |
| D021 | | — | 2.1 | 3.6 | mA | 3.0 | FOSC = 32 MHz<br>HS Oscillator mode (Note 6) |
| | | — | 2.3 | 4.3 | mA | 3.6 | |
| D021 | | — | 2.1 | 3.7 | mA | 3.0 | FOSC = 32 MHz<br>HS Oscillator mode (Note 5, Note 6) |
| | | — | 2.2 | 4.1 | mA | 5.0 | |

**Note 1:** The test conditions for all IDD measurements in active operation mode are: OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD; $\overline{MCLR}$ = VDD; WDT disabled.

**2:** The supply current is mainly a function of the operating voltage and frequency. Other factors, such as I/O pin loading and switching rate, oscillator type, internal code execution pattern and temperature, also have an impact on the current consumption.

**3:** For RC oscillator configurations, current through REXT is not included. The current through the resistor can be extended by the formula IR = VDD/2 REXT (mA) with REXT in kΩ.

**4:** FVR and BOR are disabled.

**5:** 0.1 µF capacitor on VCAP (RF0).

**6:** 8 MHz crystal oscillator with 4x PLL enabled.


© 2010-2016 Microchip Technology Inc.

## 30.4 DC Characteristics: PIC16(L)F1946/47-I/E (Power-Down)

| PIC16LF1946/47 | | | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | | |
|---|---|---|---|---|---|---|---|---|
| **PIC16F1946/47** | | | **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | | |
| **Param No.** | **Device Characteristics** | **Min.** | **Typ†** | **Max. +85°C** | **Max. +125°C** | **Units** | **Conditions** | |
| | | | | | | | **VDD** | **Note** |
| | **Power-down Base Current (IPD)(2)** | | | | | | | |
| D023 | | — | 0.06 | 1.0 | 8.0 | µA | 1.8 | WDT, BOR, FVR, and T1OSC disabled, all Peripherals Inactive |
| | | — | 0.08 | 2.0 | 9.0 | µA | 3.0 | |
| D023 | | — | 21 | 55 | 63 | µA | 1.8 | WDT, BOR, FVR, and T1OSC disabled, all Peripherals Inactive |
| | | — | 25 | 58 | 78 | µA | 3.0 | |
| | | — | 27 | 60 | 88 | µA | 5.0 | |
| D024 | | — | 0.5 | 6.0 | 9.0 | µA | 1.8 | LPWDT Current **(Note 1)** |
| | | — | 0.8 | 7.0 | 10 | µA | 3.0 | |
| D024 | | — | 23 | 57 | 65 | µA | 1.8 | LPWDT Current **(Note 1)** |
| | | — | 26 | 59 | 80 | µA | 3.0 | |
| | | — | 28 | 61 | 90 | µA | 5.0 | |
| D025 | | — | 15 | 28 | 30 | µA | 1.8 | FVR current |
| | | — | 15 | 30 | 33 | µA | 3.0 | |
| D025 | | — | 38 | 96 | 100 | µA | 1.8 | FVR current **(Note 4)** |
| | | — | 45 | 110 | 120 | µA | 3.0 | |
| | | — | 90 | 140 | 155 | µA | 5.0 | |
| D026 | | — | 13 | 16 | 20 | µA | 3.0 | BOR Current **(Note 1)** |
| D026 | | — | 40 | 110 | 120 | µA | 3.0 | BOR Current **(Note 1, Note 4)** |
| | | — | 87 | 140 | 155 | µA | 5.0 | |
| D027 | | — | 0.6 | 6.0 | 9.0 | µA | 1.8 | T1OSC Current **(Note 1)** |
| | | — | 1.8 | 10 | 12 | µA | 3.0 | |
| D027 | | — | 22 | 57 | 60 | µA | 1.8 | T1OSC Current **(Note 1)** |
| | | — | 29 | 62 | 70 | µA | 3.0 | |
| | | — | 35 | 66 | 85 | 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 current is the sum of the base IDD or IPD and the additional current consumed when this peripheral is enabled. The peripheral Δ current can be determined by subtracting the base IDD or IPD current from this limit. Max values should be used when calculating total current consumption.

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

**3:** A/D oscillator source is FRC.

**4:** 0.1 µF capacitor on VCAP (RF0).

## 30.4 DC Characteristics: PIC16(L)F1946/47-I/E (Power-Down) (Continued)

| PIC16LF1946/47 | | | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | | |
|---|---|---|---|---|---|---|---|---|
| PIC16F1946/47 | | | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | | |
| Param No. | Device Characteristics | Min. | Typ† | Max. +85°C | Max. +125°C | Units | Conditions | |
| | | | | | | | VDD | Note |
| | **Power-down Base Current (IPD)(2)** | | | | | | | |
| D028 | | — | 0.1 | 5.0 | 8.0 | µA | 1.8 | A/D Current (Note 1, Note 3), no conversion in progress |
| | | — | 0.1 | 6.0 | 9.0 | µA | 3.0 | |
| D028 | | — | 22 | 56 | 63 | µA | 1.8 | A/D Current (Note 1, Note 3), no conversion in progress |
| | | — | 26 | 58 | 78 | µA | 3.0 | |
| | | — | 27 | 61 | 88 | µA | 5.0 | |
| D029 | | — | 250 | — | — | µA | 1.8 | A/D Current (Note 1, Note 3), conversion in progress |
| | | — | 250 | — | — | µA | 3.0 | |
| D029 | | — | 280 | — | — | µA | 1.8 | A/D Current (Note 1, Note 3, Note 4), conversion in progress |
| | | — | 280 | — | — | µA | 3.0 | |
| | | — | 280 | — | — | µA | 5.0 | |
| D030 | | — | 1 | — | — | µA | 3.0 | LCD Bias Ladder, Low-power |
| | | — | 10 | — | — | µA | 3.0 | LCD Bias Ladder, Medium-power |
| | | — | 75 | — | — | µA | 3.0 | LCD Bias Ladder, High-power |
| D030 | | — | 1 | — | — | µA | 5.0 | LCD Bias Ladder, Low-power |
| | | — | 10 | — | — | µA | 5.0 | LCD Bias Ladder, Medium-power |
| | | — | 75 | — | — | µA | 5.0 | LCD Bias Ladder, High-power |
| D031 | | — | 7.6 | 22 | 25 | µA | 1.8 | Comparator, Low-Power mode |
| | | — | 8.0 | 23 | 27 | µA | 3.0 | |
| D031 | | — | 24 | 55 | 65 | µA | 1.8 | Comparator, Low-Power mode |
| | | — | 26 | 58 | 80 | µA | 3.0 | |
| | | — | 28 | 60 | 90 | µA | 5.0 | |
| D032A* | | — | 2.0 | — | — | µA | 1.8 | Cap Sense, Low-Power mode, CPSRM=0 |
| | | — | 3.0 | — | — | µA | 3.0 | |
| D032A* | | — | 23 | — | — | µA | 1.8 | Cap Sense, Low-Power mode, CPSRM=0 |
| | | — | 28 | — | — | µA | 3.0 | |
| | | — | 30 | — | — | 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 current is the sum of the base IDD or IPD and the additional current consumed when this peripheral is enabled. The peripheral Δ current can be determined by subtracting the base IDD or IPD current from this limit. Max values should be used when calculating total current consumption.

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

**3:** A/D oscillator source is FRC.

**4:** 0.1 µF capacitor on VCAP (RF0).


© 2010-2016 Microchip Technology Inc.

## 30.4 DC Characteristics: PIC16(L)F1946/47-I/E (Power-Down) (Continued)

| PIC16LF1946/47 | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended |
|---|---|
| PIC16F1946/47 | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended |

| Param No. | Device Characteristics | Min. | Typ† | Max. +85°C | Max. +125°C | Units | Conditions | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | VDD | Note |
| D032B* | **Power-down Base Current (IPD)(2)** | | | | | | | |
| | | — | 80 | — | — | µA | 1.8 | Cap Sense, Low Power mode, CPSRM = 1, includes FVR and DAC current |
| | | — | 90 | — | — | µA | 3.0 | |
| D032B* | | — | 110 | — | — | µA | 1.8 | Cap Sense, Low-Power mode, CPSRM = 1, includes FVR and DAC current |
| | | — | 120 | — | — | µA | 3.0 | |
| | | — | 130 | — | — | µA | 5.0 | |
| D032C* | | — | 4.0 | — | — | µA | 1.8 | Cap Sense, Medium-Power mode, CPSRM = 0 |
| | | — | 6.0 | — | — | µA | 3.0 | |
| D032C* | | — | 25 | — | — | µA | 1.8 | Cap Sense, Medium-Power mode, CPSRM = 0 |
| | | — | 30 | — | — | µA | 3.0 | |
| | | — | 32 | — | — | µA | 5.0 | |
| D032D* | | — | 90 | — | — | µA | 1.8 | Cap Sense, Medium-Power mode, CPSRM = 1, includes FVR and DAC current |
| | | — | 120 | — | — | µA | 3.0 | |
| D032D* | | — | 120 | — | — | µA | 1.8 | Cap Sense, Medium-Power mode, CPSRM = 1, includes FVR and DAC current |
| | | — | 140 | — | — | µA | 3.0 | |
| | | — | 150 | — | — | µA | 5.0 | |
| D032E* | | — | 12 | — | — | µA | 1.8 | Cap Sense, High-Power mode, CPSRM = 0 |
| | | — | 31 | — | — | µA | 3.0 | |
| D032E* | | — | 33 | — | — | µA | 1.8 | Cap Sense, High-Power mode, CPSRM = 0 |
| | | — | 52 | — | — | µA | 3.0 | |
| | | — | 62 | — | — | µA | 5.0 | |
| D032F* | | — | 120 | — | — | µA | 1.8 | Cap Sense, High-Power mode, CPSRM = 1, includes FVR and DAC current |
| | | — | 160 | — | — | µA | 3.0 | |
| D032F* | | — | 150 | — | — | µA | 1.8 | Cap Sense, High-Power mode, CPSRM = 1, includes FVR and DAC current |
| | | — | 180 | — | — | µA | 3.0 | |
| | | — | 190 | — | — | 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 current is the sum of the base IDD or IPD and the additional current consumed when this peripheral is enabled. The peripheral Δ current can be determined by subtracting the base IDD or IPD current from this limit. Max values should be used when calculating total current consumption.

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

**3:** A/D oscillator source is FRC.

**4:** 0.1 µF capacitor on VCAP (RF0).

## 30.5 DC Characteristics: PIC16(L)F1946/47-I/E

| DC CHARACTERISTICS | | | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +85°C for industrial<br>-40°C ≤ TA ≤ +125°C for extended | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** |
| | VIL | **Input Low Voltage** | | | | | |
| | | I/O PORT: | | | | | |
| D032 | | with TTL buffer | — | — | 0.8 | V | 4.5V ≤ VDD ≤ 5.5V |
| D032A | | | — | — | 0.15 VDD | V | 1.8V ≤ VDD ≤ 4.5V |
| D033 | | with Schmitt Trigger buffer | — | — | 0.2 VDD | V | 2.0V ≤ VDD ≤ 5.5V |
| | | with I2C levels | — | — | 0.3 VDD | V | |
| | | with SMBus levels | — | — | 0.8 | V | 2.7V ≤ VDD ≤ 5.5V |
| D034 | | $\overline{MCLR}$, OSC1 (RC mode)(1) | — | — | 0.2 VDD | V | |
| D034A | | OSC1 (HS mode) | — | — | 0.3 VDD | V | |
| | VIH | **Input High Voltage** | | | | | |
| | | I/O ports: | | | | | |
| D040 | | with TTL buffer | 2.0 | — | — | V | 4.5V ≤ VDD ≤ 5.5V |
| D040A | | | 0.25 VDD + 0.8 | — | — | V | 1.8V ≤ VDD ≤ 4.5V |
| D041 | | with Schmitt Trigger buffer | 0.8 VDD | — | — | V | 2.0V ≤ VDD ≤ 5.5V |
| | | with I2C levels | 0.7 VDD | — | — | V | |
| | | with SMBus levels | 2.1 | — | — | V | 2.7V ≤ VDD ≤ 5.5V |
| D042 | | $\overline{MCLR}$ | 0.8 VDD | — | — | V | |
| D043A | | OSC1 (HS mode) | 0.7 VDD | — | — | V | |
| D043B | | OSC1 (RC mode) | 0.9 VDD | — | — | V | (Note 1) VDD > 2.0V |
| | IIL | **Input Leakage Current(2)** | | | | | |
| D060 | | I/O ports | — | ± 5 | ± 125 | nA | VSS ≤ VPIN ≤ VDD, Pin at high-impedance @ 85°C |
| | | | | ± 5 | ± 1000 | nA | 125°C |
| D061 | | $\overline{MCLR}$(3) | — | ± 50 | ± 200 | nA | VSS ≤ VPIN ≤ VDD @ 85°C |
| | IPUR | **Weak Pull-up Current** | | | | | |
| D070* | | | 25<br>25 | 100<br>140 | 200<br>300 | µA | VDD = 3.3V, VPIN = VSS<br>VDD = 5.0V, VPIN = VSS |
| | VOL | **Output Low Voltage(4)** | | | | | |
| D080 | | I/O ports | — | — | 0.6 | V | IOL = 8mA, VDD = 5V<br>IOL = 6mA, VDD = 3.3V<br>IOL = 1.8mA, VDD = 1.8V |
| | VOH | **Output High Voltage(4)** | | | | | |
| D090 | | I/O ports | VDD - 0.7 | — | — | V | IOH = 3.5mA, VDD = 5V<br>IOH = 3mA, VDD = 3.3V<br>IOH = 1mA, VDD = 1.8V |
| | | **Capacitive Loading Specs on Output Pins** | | | | | |
| D101* | COSC2 | OSC2 pin | — | — | 15 | pF | In XT, HS and LP modes when external clock is used to drive OSC1 |
| D101A* | CIO | All I/O pins | — | — | 50 | pF | |
| | | **VCAP Capacitor Charging** | | | | | |
| D102 | | Charging current | — | 200 | — | µA | |
| D102A | | Source/sink capability when charging complete | — | 0.0 | — | mA | |

* 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:** In RC oscillator configuration, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended to use an external clock in RC mode.

**2:** Negative current is defined as current sourced by the pin.

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

**4:** Including OSC2 in CLKOUT mode.


© 2010-2016 Microchip Technology Inc.

## 30.6 Memory Programming Requirements

| DC CHARACTERISTICS | | | Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +125°C | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** |
| | | **Program Memory Programming Specifications** | | | | | |
| D110 | VIHH | Voltage on MCLR/VPP pin | 8.0 | — | 9.0 | V | **(Note 3, Note 4)** |
| D111 | IDDP | Supply Current during Programming | — | — | 10 | mA | |
| 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 MCLR/VPP during Erase/Write | — | — | 1.0 | mA | |
| D115 | IDDPGM | Current on VDD during Erase/Write | — | | 5.0 | mA | |
| | | **Data EEPROM Memory** | | | | | |
| D116 | ED | Byte Endurance | 100K | — | — | E/W | -40°C to +85°C |
| D117 | VDRW | VDD for Read/Write | VDDMIN | — | VDDMAX | V | |
| D118 | TDEW | Erase/Write Cycle Time | — | 4.0 | 5.0 | ms | |
| D119 | TRETD | Characteristic Retention | — | 40 | — | Year | -40°C to +55°C<br>Provided no other specifications are violated |
| D120 | TREF | Number of Total Erase/Write Cycles before Refresh(2) | 1M | 10M | — | E/W | -40°C to +85°C |
| | | **Program Flash Memory** | | | | | |
| D121 | EP | Cell Endurance | 10K | — | — | E/W | -40°C to +85°C (**Note 1**) |
| D122 | VPRW | VDD for Read/Write | VDDMIN | — | VDDMAX | V | |
| D123 | TIW | Self-timed Write Cycle Time | — | 2 | 2.5 | ms | |
| D124 | TRETD | Characteristic Retention | — | 40 | — | Year | Provided no other specifications are violated |

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note 1:** Self-write and Block Erase.

**2:** Refer to **Section 11.2 "Using the Data EEPROM"** for a more detailed discussion on data EEPROM endurance.

**3:** Required only if single-supply programming is disabled.

**4:** The MPLAB® ICD 2 does not support variable VPP output. Circuitry to limit the MPLAB ICD 2 VPP voltage must be placed between the MPLAB ICD 2 and target system when programming or debugging with the MPLAB ICD 2.

## 30.7 Thermal Considerations

| Standard Operating Conditions (unless otherwise stated)<br>Operating temperature -40°C ≤ TA ≤ +125°C | | | | | |
|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Typ.** | **Units** | **Conditions** |
| TH01 | θJA | Thermal Resistance Junction to Ambient | 48.3 | °C/W | 64-pin TQFP package |
| | | | 28 | °C/W | 64-pin QFN package |
| TH02 | θJC | Thermal Resistance Junction to Case | 26.1 | °C/W | 64-pin TQFP package |
| | | | 0.24 | °C/W | 64-pin QFN package |
| TH03 | TJMAX | Maximum Junction Temperature | 150 | °C | |
| TH04 | PD | Power Dissipation | — | W | PD = PINTERNAL + PI/O |
| TH05 | PINTERNAL | Internal Power Dissipation | — | W | PINTERNAL = IDD x VDD(1) |
| TH06 | PI/O | I/O Power Dissipation | — | W | PI/O = Σ (IOL * VOL) + Σ (IOH * (VDD - VOH)) |
| TH07 | PDER | Derated Power | — | W | PDER = PDMAX (TJ - TA)/θJA(2),(3) |

**Note 1:** IDD is current to run the chip alone without driving any load on the output pins.
**2:** TA = Ambient Temperature
**3:** TJ = Junction Temperature


© 2010-2016 Microchip Technology Inc.

## 30.8 Timing Parameter Symbology

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

1. TppS2ppS
2. TppS

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

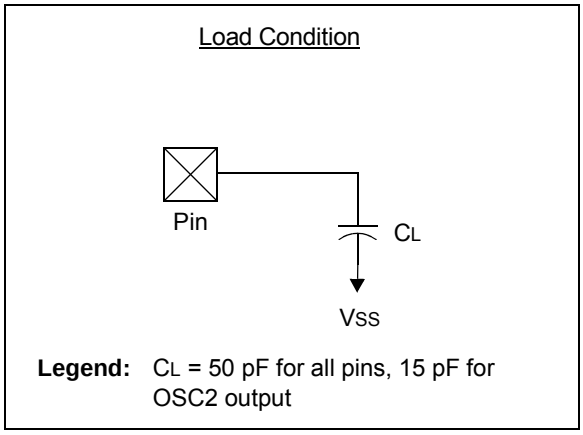
Lowercase letters (pp) and their meanings:

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

Uppercase letters and their meanings:

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

**FIGURE 30-5: LOAD CONDITIONS**




**Legend:** CL = 50 pF for all pins, 15 pF for OSC2 output

## 30.9 AC Characteristics: PIC16(L)F1946/47-I/E

**FIGURE 30-6: CLOCK TIMING**

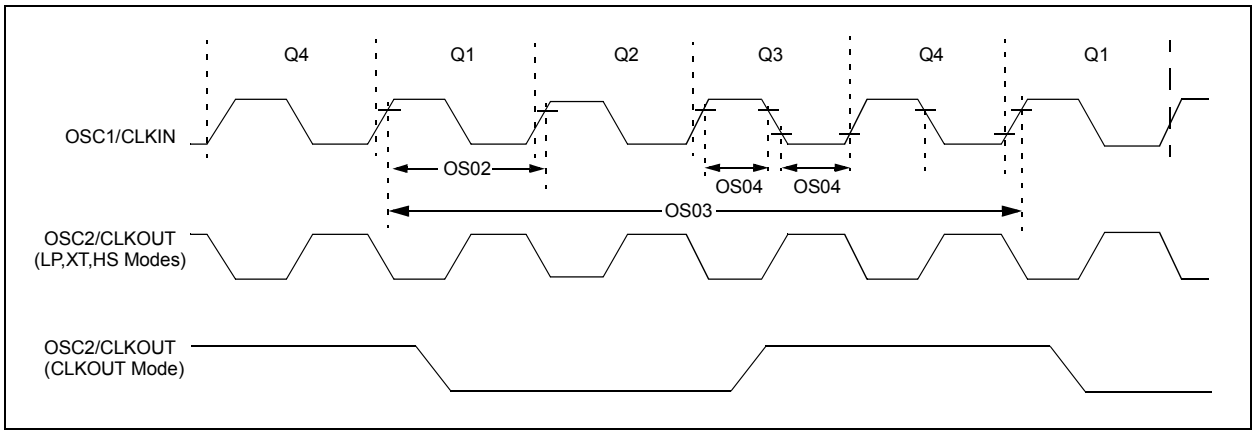



**TABLE 30-1: CLOCK OSCILLATOR TIMING REQUIREMENTS**

**Standard Operating Conditions (unless otherwise stated)**
Operating temperature -40°C ≤ TA ≤ +125°C

| Param No. | Sym. | Characteristic | Min. | Typ† | Max. | Units | Conditions |
|---|---|---|---|---|---|---|---|
| OS01 | FOSC | External CLKIN Frequency(1) | DC | — | 0.5 | MHz | EC Oscillator mode (low) |
| | | | DC | — | 4 | MHz | EC Oscillator mode (medium) |
| | | | DC | — | 20 | MHz | EC Oscillator mode (high) |
| | | Oscillator Frequency(1) | — | 32.768 | — | kHz | LP Oscillator mode |
| | | | 0.1 | — | 4 | MHz | XT Oscillator mode |
| | | | 1 | — | 4 | MHz | HS Oscillator mode |
| | | | 1 | — | 20 | MHz | HS Oscillator mode, VDD > 2.7V |
| | | | DC | — | 4 | MHz | RC Oscillator mode, VDD > 2.0V |
| OS02 | TOSC | External CLKIN Period(1) | 27 | — | ∞ | µs | LP Oscillator mode |
| | | | 250 | — | ∞ | ns | XT Oscillator mode |
| | | | 50 | — | ∞ | ns | HS Oscillator mode |
| | | | 50 | — | ∞ | ns | EC Oscillator mode |
| | | Oscillator Period(1) | — | 30.5 | — | µs | LP Oscillator mode |
| | | | 250 | — | 10,000 | ns | XT Oscillator mode |
| | | | 50 | — | 1,000 | ns | HS Oscillator mode |
| | | | 250 | — | — | ns | RC Oscillator mode |
| OS03 | TCY | Instruction Cycle Time(1) | 200 | TCY | DC | ns | TCY = 4/FOSC |
| OS04* | TosH, TosL | External CLKIN High, External CLKIN Low | 2 | — | — | µs | LP oscillator |
| | | | 100 | — | — | ns | XT oscillator |
| | | | 20 | — | — | ns | HS oscillator |

* 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 OSC1 pin. When an external clock input is used, the "max" cycle time limit is "DC" (no clock) for all devices.


© 2010-2016 Microchip Technology Inc.

**TABLE 30-1: CLOCK OSCILLATOR TIMING REQUIREMENTS (CONTINUED)**

| **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +125°C | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** |
| OS05* | TosR, TosF | External CLKIN Rise, External CLKIN Fall | 0 | — | ∞ | ns | LP oscillator |
| | | | 0 | — | ∞ | ns | XT oscillator |
| | | | 0 | — | ∞ | ns | HS oscillator |

* 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 OSC1 pin. When an external clock input is used, the "max" cycle time limit is "DC" (no clock) for all devices.

**TABLE 30-2: OSCILLATOR PARAMETERS**

| **Standard Operating Conditions (unless otherwise stated)**<br>Operating Temperature -40°C ≤ TA ≤ +125°C | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Freq. Tolerance** | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** |
| OS08 | HFOSC | Internal-Calibrated HFINTOSC Frequency(1) | ±2% | — | 16.0 | — | MHz | 0°C ≤ TA ≤ +60°C, VDD ≥ 2.5V |
| | | | ±3% | — | 16.0 | — | MHz | 60°C ≤ TA ≤ +85°C, VDD ≥ 2.5V |
| | | | ±5% | — | 16.0 | — | MHz | -40°C ≤ TA ≤ +125°C |
| OS08A | MFOSC | Internal-Calibrated MFINTOSC Frequency(1) | ±2% | — | 500 | — | kHz | 0°C ≤ TA ≤ +60°C, VDD ≥ 2.5V |
| | | | ±3% | — | 500 | — | kHz | 60°C ≤ TA ≤ +85°C, VDD ≥ 2.5V |
| | | | ±5% | — | 500 | — | kHz | -40°C ≤ TA ≤ +125°C |
| OS09 | LFOSC | Internal LFINTOSC Frequency | — | — | 31 | — | kHz | -40°C ≤ TA ≤ +125°C |
| OS10* | TIOSC ST | HFINTOSC Wake-up from Sleep Start-up Time | — | — | 3.2 | 8 | µs | |
| | | MFINTOSC Wake-up from Sleep Start-up Time | — | — | 24 | 35 | µs | |

* 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, 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.

**TABLE 30-3: PLL CLOCK TIMING SPECIFICATIONS (VDD = 2.7V TO 5.5V)**

| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** |
|---|---|---|---|---|---|---|---|
| F10 | FOSC | Oscillator Frequency Range | 4 | — | 8 | MHz | |
| F11 | FSYS | On-Chip VCO System Frequency | 16 | — | 32 | MHz | |
| F12 | TRC | PLL Start-up Time (Lock Time) | — | — | 2 | ms | |
| F13* | ΔCLK | CLKOUT Stability (Jitter) | -0.25% | — | +0.25% | % | |

* These parameters are characterized but not tested.

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

**FIGURE 30-7: CLKOUT AND I/O TIMING**

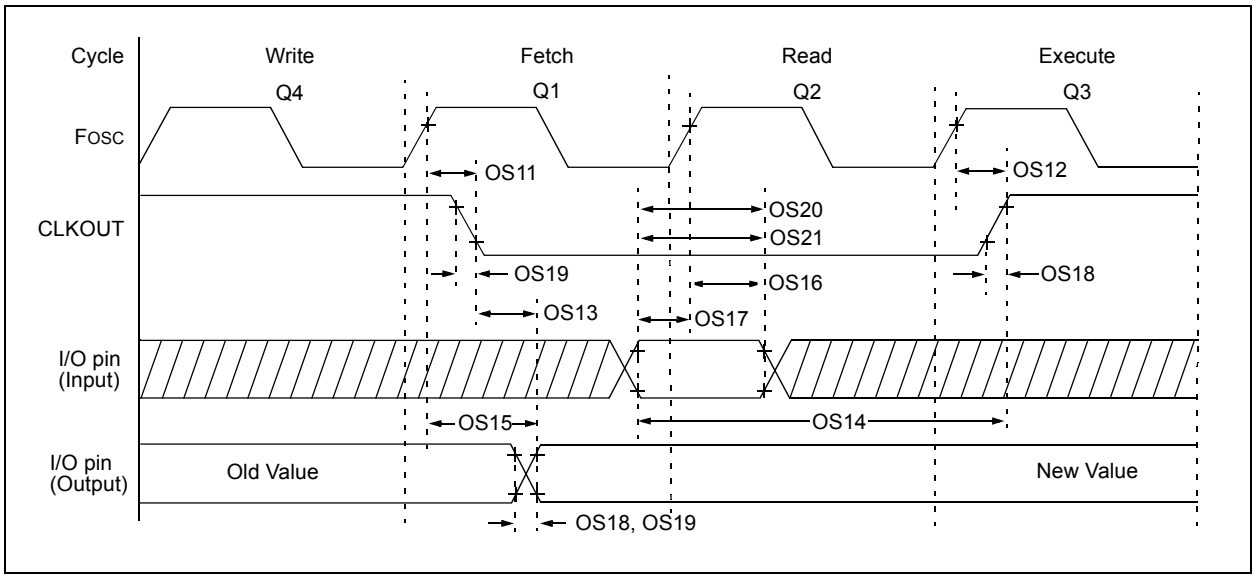



**TABLE 30-4: CLKOUT AND I/O TIMING PARAMETERS**

| **Standard Operating Conditions (unless otherwise stated)**<br>Operating Temperature -40°C ≤ TA ≤ +125°C | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** |
| OS11 | TosH2ckL | FOSC↑ to CLKOUT↓ (1) | — | — | 70 | ns | VDD = 3.3-5.0V |
| OS12 | TosH2ckH | FOSC↑ to CLKOUT↑ (1) | — | — | 72 | ns | VDD = 3.3-5.0V |
| OS13 | TckL2ioV | CLKOUT↓ to Port out valid(1) | — | — | 20 | ns | |
| OS14 | TioV2ckH | Port input valid before CLKOUT↑(1) | TOSC + 200 ns | — | — | ns | |
| OS15 | TosH2ioV | FOSC↑ (Q1 cycle) to Port out valid | — | 50 | 70* | ns | VDD = 3.3-5.0V |
| OS16 | TosH2ioI | FOSC↑ (Q2 cycle) to Port input invalid (I/O in hold time) | 50 | — | — | ns | VDD = 3.3-5.0V |
| OS17 | TioV2osH | Port input valid to Fosc↑ (Q2 cycle) (I/O in setup time) | 20 | — | — | ns | |
| OS18 | TioR | Port output rise time | —<br>— | 40<br>15 | 72<br>32 | ns | VDD = 1.8V<br>VDD = 3.3-5.0V |
| OS19 | TioF | Port output fall time | —<br>— | 28<br>15 | 55<br>30 | ns | VDD = 1.8V<br>VDD = 3.3-5.0V |
| OS20* | Tinp | INT pin input high or low time | 25 | — | — | ns | |
| OS21* | Tioc | Interrupt-on-change new input level time | 25 | — | — | ns | |

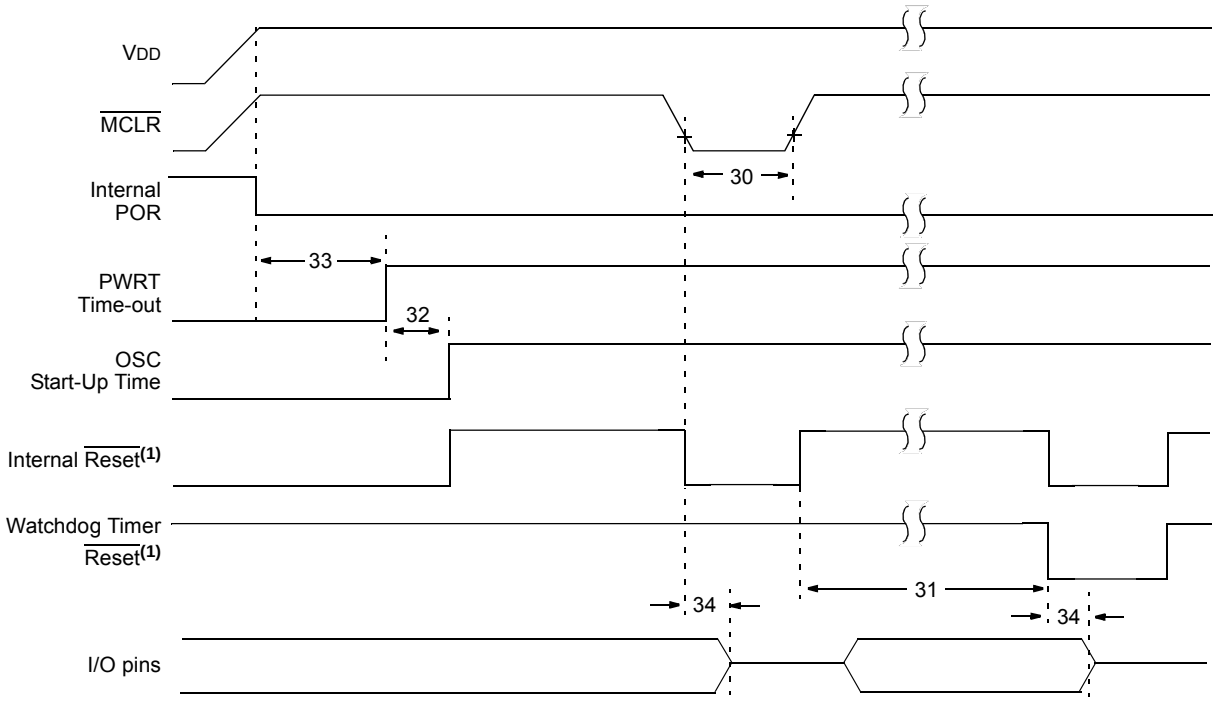
* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated.

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

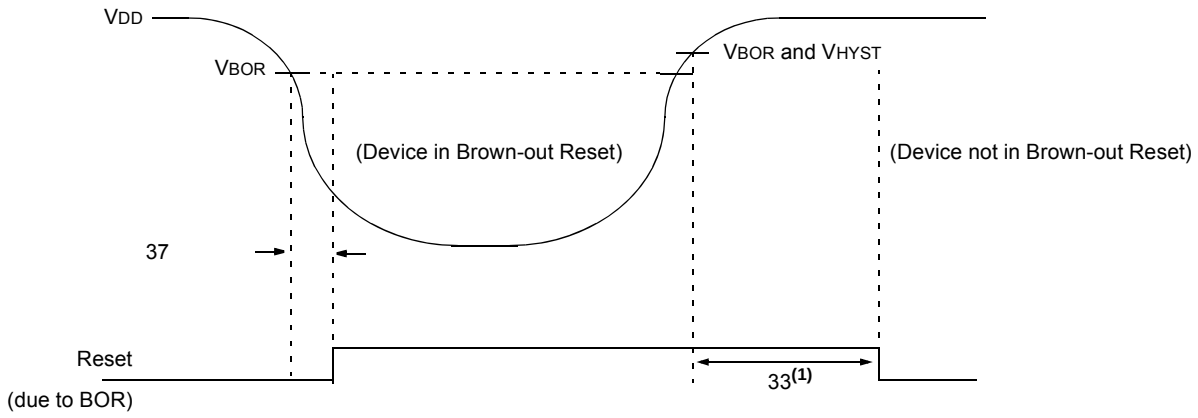

© 2010-2016 Microchip Technology Inc.

**FIGURE 30-8: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING**




**Note 1:** Asserted low.

**FIGURE 30-9: BROWN-OUT RESET TIMING AND CHARACTERISTICS**




**Note 1:** 64 ms delay only if PWRTE bit in the Configuration Word register is programmed to '0'.
2 ms delay if PWRTE = 0 and VREGEN = 1.

**TABLE 30-5: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER AND BROWN-OUT RESET PARAMETERS**

| **Standard Operating Conditions (unless otherwise stated)**<br>Operating Temperature -40°C ≤ TA ≤ +125°C | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** |
| 30 | TMCL | $\overline{MCLR}$ Pulse Width (low) | 2 | — | — | μs | |
| 31 | TWDTLP | Watchdog Timer Time-out Period | 10 | 16 | 27 | ms | VDD = 3.3V-5V,<br>1:512 Prescaler used |
| 32 | TOST | Oscillator Start-up Timer Period(1) | — | 1024 | — | TOSC | |
| 33* | TPWRT | Power-up Timer Period,<br>$\overline{PWRTE}$ = 0 | 40 | 65 | 140 | ms | |
| 34* | TIOZ | I/O high-impedance from $\overline{MCLR}$ Low or Watchdog Timer Reset | — | — | 2.0 | μs | |
| 35 | VBOR | Brown-out Reset Voltage(2) | 2.55<br>1.80 | 2.70<br>1.90 | 2.85<br>2.11 | V<br>V | BORV = 0<br>BORV = 1 |
| 36* | VHYST | Brown-out Reset Hysteresis | 0 | 25 | 50 | mV | -40°C to +85°C |
| 37* | TBORDC | Brown-out Reset DC Response Time | 1 | 3 | 35 | μs | VDD ≤ VBOR |

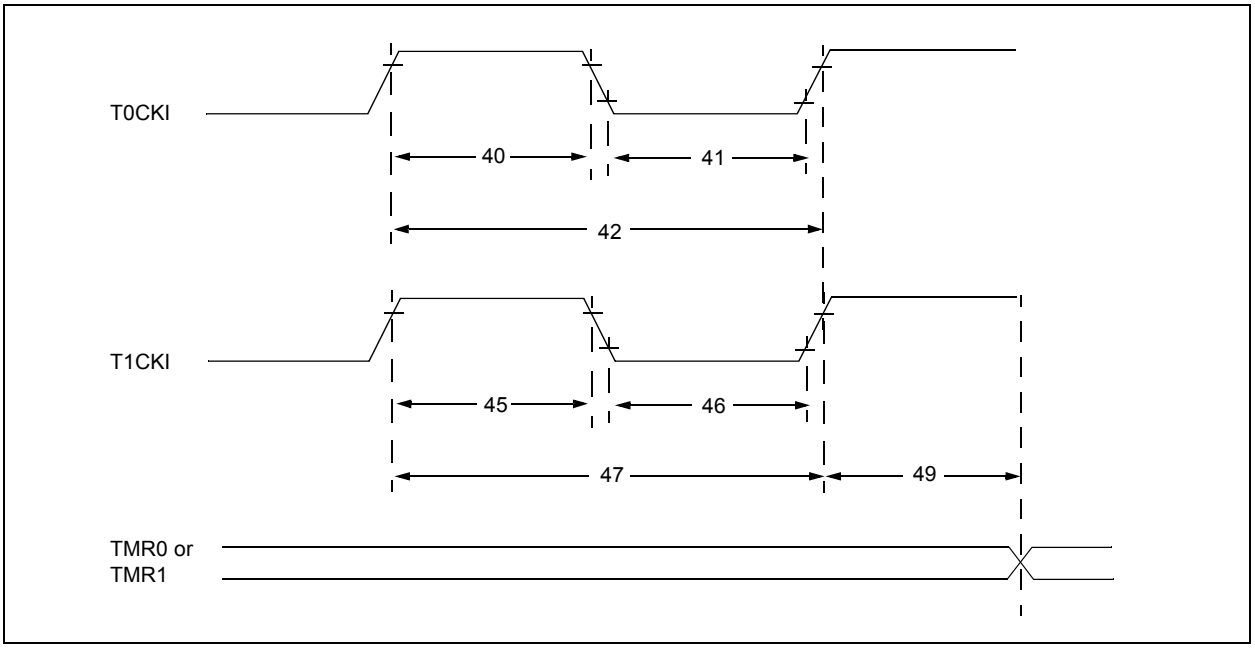
* These parameters are characterized but not tested.

† 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:** By design, the Oscillator Start-up Timer (OST) counts the first 1024 cycles, independent of frequency.

**2:** 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 30-10: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS**





© 2010-2016 Microchip Technology Inc.

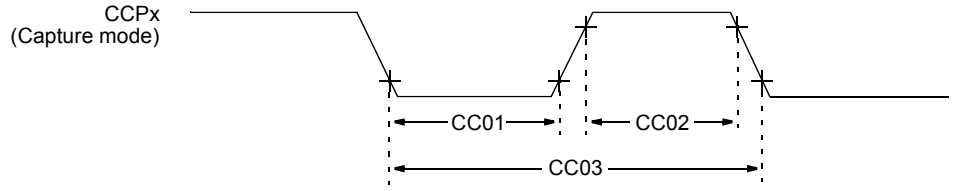
**TABLE 30-6: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS**

| **Standard Operating Conditions (unless otherwise stated)**<br>Operating Temperature -40°C ≤ TA ≤ +125°C | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** |
| 40* | TT0H | T0CKI High-Pulse Width | No Prescaler | 0.5 TCY + 20 | — | — | ns | |
| | | | With Prescaler | 10 | — | — | ns | |
| 41* | TT0L | T0CKI Low-Pulse Width | No Prescaler | 0.5 TCY + 20 | — | — | ns | |
| | | | With Prescaler | 10 | — | — | ns | |
| 42* | TT0P | T0CKI Period | | Greater of: 20 or TCY + 40 / N | — | — | ns | N = prescale value (2, 4, ..., 256) |
| 45* | TT1H | T1CKI High Time | Synchronous, No Prescaler | 0.5 TCY + 20 | — | — | ns | |
| | | | Synchronous, with Prescaler | 15 | — | — | ns | |
| | | | Asynchronous | 30 | — | — | ns | |
| 46* | TT1L | T1CKI Low Time | Synchronous, No Prescaler | 0.5 TCY + 20 | — | — | ns | |
| | | | Synchronous, with Prescaler | 15 | — | — | ns | |
| | | | Asynchronous | 30 | — | — | ns | |
| 47* | TT1P | T1CKI Input Period | Synchronous | Greater of: 30 or TCY + 40 / N | — | — | ns | N = prescale value (1, 2, 4, 8) |
| | | | Asynchronous | 60 | — | — | ns | |
| 48 | FT1 | Timer1 Oscillator Input Frequency Range (oscillator enabled by setting bit T1OSCEN) | | 32.4 | 32.768 | 33.1 | kHz | |
| 49* | TCKEZTMR1 | Delay from External Clock Edge to Timer Increment | | 2 TOSC | — | 7 TOSC | — | Timers in Sync mode |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**FIGURE 30-11: CAPTURE/COMPARE/PWM TIMINGS (CCP)**




**Note:** Refer to Figure 30-5 for load conditions.

### TABLE 30-7: CAPTURE/COMPARE/PWM REQUIREMENTS (CCP)

| Standard Operating Conditions (unless otherwise stated)<br>Operating Temperature -40°C ≤ TA ≤ +125°C | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** |
| CC01* | TccL | CCPx Input Low Time | No Prescaler | 0.5TCY + 20 | — | — | ns | |
| | | | With Prescaler | 20 | — | — | ns | |
| CC02* | TccH | CCPx Input High Time | No Prescaler | 0.5TCY + 20 | — | — | ns | |
| | | | With Prescaler | 20 | — | — | ns | |
| CC03* | TccP | CCPx Input Period | | $\frac{3T_{CY} + 40}{N}$ | — | — | ns | N = prescale value (1, 4 or 16) |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

### TABLE 30-8: PIC16(L)F1946/47 A/D CONVERTER (ADC) CHARACTERISTICS(1,2,3)

| Standard Operating Conditions (unless otherwise stated)<br>Operating temperature TA = 25°C | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** |
| AD01 | NR | Resolution | — | — | 10 | bit | |
| AD02 | EIL | Integral Error | — | — | ±1.7 | LSb | VREF = 3.0V |
| AD03 | EDL | Differential Error | — | — | ±1 | LSb | No missing codes<br>VREF = 3.0V |
| AD04 | EOFF | Offset Error | — | — | ±2.5 | LSb | VREF = 3.0V |
| AD05 | EGN | Gain Error | — | — | ±2.0 | LSb | VREF = 3.0V |
| AD06 | VREF | Reference Voltage(4) | 1.8 | — | VDD | V | VREF = (VREF+ minus VREF-) |
| AD07 | VAIN | Full-Scale Range | VSS | — | VREF | V | |
| AD08 | ZAIN | Recommended Impedance of Analog Voltage Source | — | — | 10 | kΩ | Can go higher if external 0.01 µF capacitor is present on input pin. |

† 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 A/D conversion result never decreases with an increase in the input voltage and has no missing codes.

**3:** When ADC is OFF, it will not consume any current other than leakage current. The power-down current specification includes any such leakage from the ADC module.

**4:** ADC Reference Voltage (Ref+) is the selected reference input, VREF+ pin, VDD pin or the FVR Buffer1. When the FVR is selected as the reference input, the FVR Buffer1 output selection must be 2.048V or 4.096V, (ADFVR<1:0> = 1x).


© 2010-2016 Microchip Technology Inc.

**TABLE 30-9: PIC16(L)F1946/47 A/D CONVERSION REQUIREMENTS**

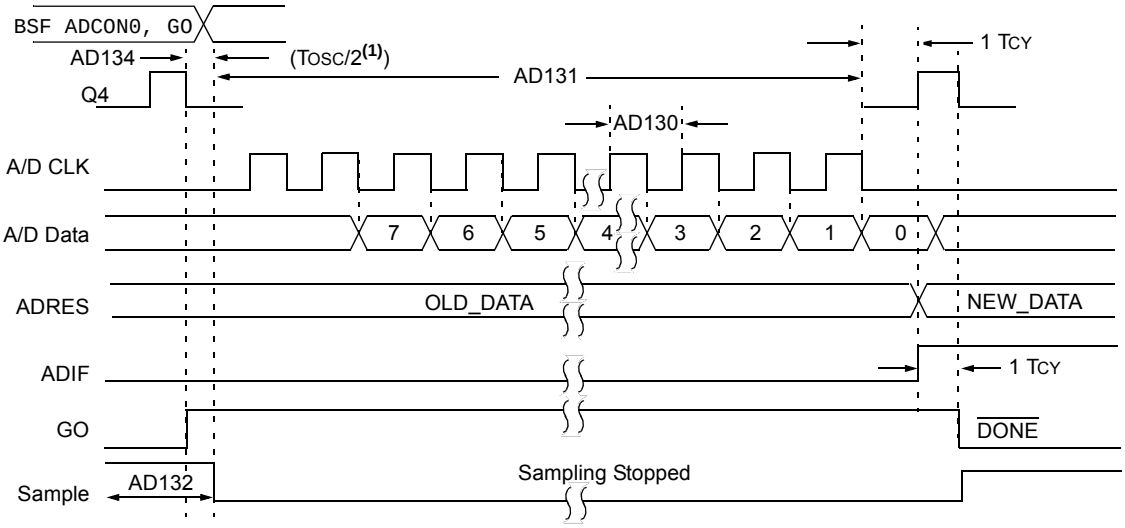
| **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +125°C | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** |
| AD130* | TAD | A/D Clock Period | 1.0 | — | 9.0 | µs | TOSC-based |
| | | A/D Internal RC Oscillator Period | 1.0 | 2.5 | 6.0 | µs | ADCS<1:0> = 11 (ADRC mode) |
| AD131 | TCNV | Conversion Time (not including Acquisition Time)(1) | — | 11 | — | TAD | Set GO/$\overline{\text{DONE}}$ bit to conversion complete |
| AD132* | TACQ | Acquisition Time | — | 5.0 | — | µs | |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

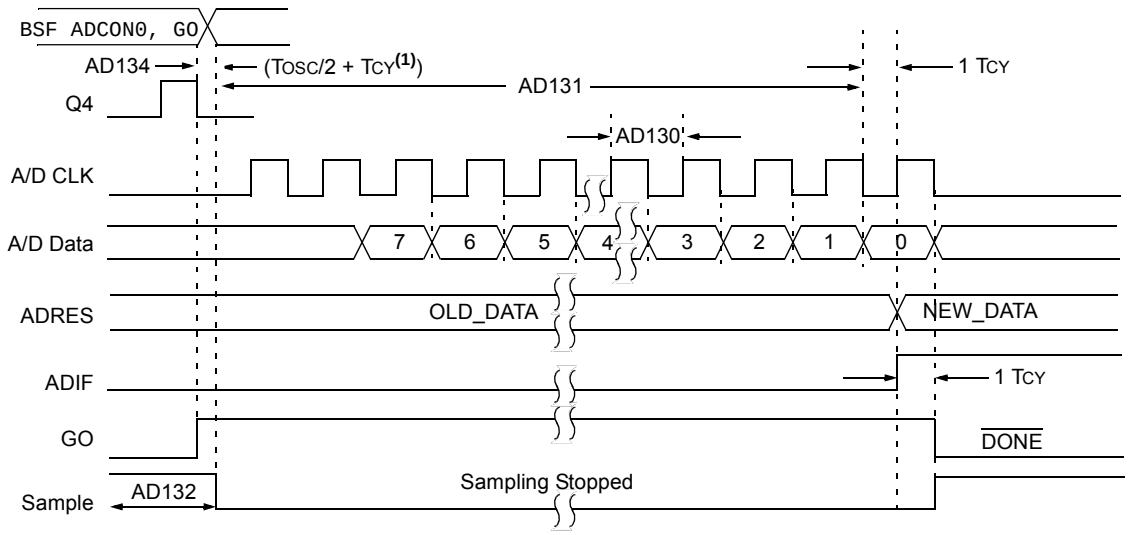
**Note 1:** The ADRES register may be read on the following TCY cycle.

**FIGURE 30-12: PIC16(L)F1946/47 A/D CONVERSION TIMING (NORMAL MODE)**




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

**FIGURE 30-13: PIC16(L)F1946/47 A/D CONVERSION TIMING (SLEEP MODE)**




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


© 2010-2016 Microchip Technology Inc.

**TABLE 30-10: COMPARATOR SPECIFICATIONS**

| **Standard Operating Conditions:** 1.8V < VDD < 5.5V, -40°C < TA < +125°C (unless otherwise stated). | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristics** | **Min.** | **Typ.** | **Max.** | **Units** | **Comments** |
| CM01 | VIOFF | Input Offset Voltage(1) | — | ±7.5 | ±60 | mV | High-Power mode VICM = VDD/2 |
| CM02 | VICM | Input Common Mode Voltage | 0 | — | VDD | V | |
| CM03 | CMRR | Common Mode Rejection Ratio | — | 50 | — | dB | |
| CM04A | TRESP | Response Time Rising Edge | — | 400 | 800 | ns | High-Power mode |
| CM04B | | Response Time Falling Edge | — | 200 | 400 | ns | High-Power mode |
| CM04C | | Response Time Rising Edge | — | 1200 | — | ns | Low-Power mode |
| CM04D | | Response Time Falling Edge | — | 550 | — | ns | Low-Power mode |
| CM05 | TMC2OV | Comparator Mode Change to Output Valid* | — | — | 10 | µs | |
| CM06 | CHYSTER | Comparator Hysteresis(2) | — | 45 | — | mV | CxHYS = 1 |

* These parameters are characterized but not tested.

**Note 1:** High power only.

**2:** Comparator Hysteresis is available when the CxHYS bit of the CMxCON0 register is enabled.

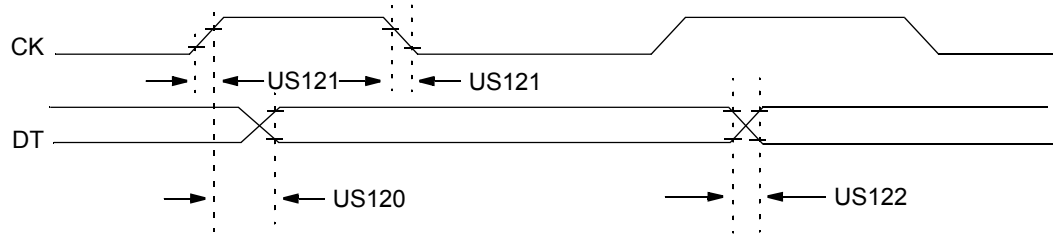
**TABLE 30-11: DIGITAL-TO-ANALOG CONVERTER (DAC) SPECIFICATIONS**

| **Standard Operating Conditions:** 2.5V < VDD < 5.5V, -40°C < TA < +85°C (unless otherwise stated). | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristics** | **Min.** | **Typ.** | **Max.** | **Units** | **Comments** |
| DAC01* | CLSB | Step Size | — | VDD/32 | — | V | |
| DAC02* | CACC | Absolute Accuracy | — | — | ± 1/2 | LSb | |
| DAC03* | CR | Unit Resistor Value (R) | — | 5K | — | Ω | |
| DAC04* | CST | Settling Time(1) | — | — | 10 | µs | |

* These parameters are characterized but not tested.

**Note 1:** Settling time measured while DACR<4:0> transitions from '0000' to '1111'.

**FIGURE 30-14: USART SYNCHRONOUS TRANSMISSION (MASTER/SLAVE) TIMING**

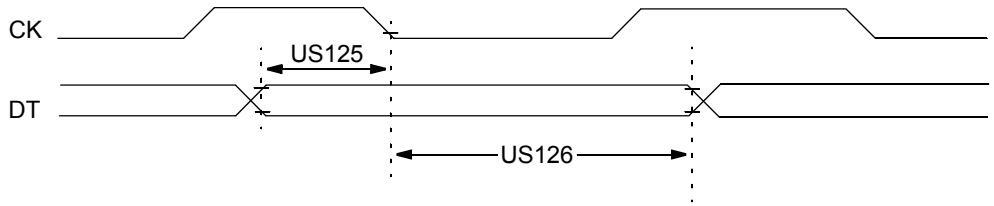



**Note:** Refer to Figure 30-5 for load conditions.

**TABLE 30-12: USART SYNCHRONOUS TRANSMISSION REQUIREMENTS**

| **Standard Operating Conditions (unless otherwise stated)**<br>Operating Temperature -40°C ≤ TA ≤ +125°C | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param. No.** | **Symbol** | **Characteristic** | | **Min.** | **Max.** | **Units** | **Conditions** |
| US120 | TCKH2DTV | SYNC XMIT (Master and Slave)<br>Clock high to data-out valid | 3.0-5.5V | — | 80 | ns | |
| | | | 1.8-5.5V | — | 100 | ns | |
| US121 | TCKRF | Clock out rise time and fall time (Master mode) | 3.0-5.5V | — | 45 | ns | |
| | | | 1.8-5.5V | — | 50 | ns | |
| US122 | TDTRF | Data-out rise time and fall time | 3.0-5.5V | — | 45 | ns | |
| | | | 1.8-5.5V | — | 50 | ns | |

**FIGURE 30-15: USART SYNCHRONOUS RECEIVE (MASTER/SLAVE) TIMING**



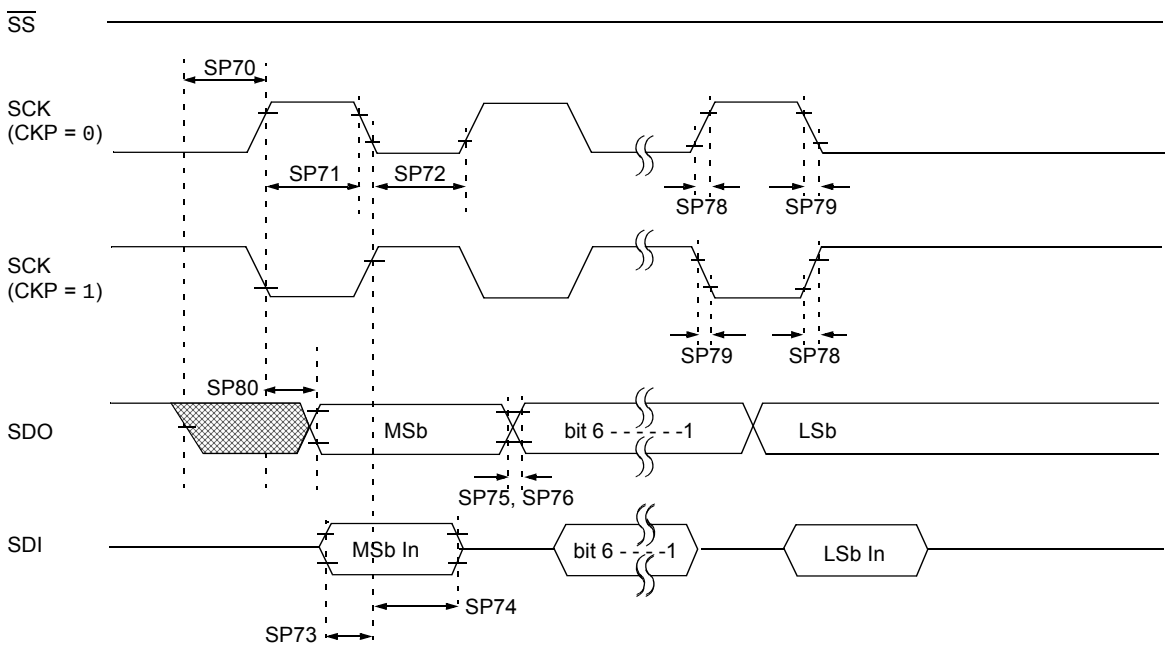

**Note:** Refer to Figure 30-5 for load conditions.

**TABLE 30-13: USART SYNCHRONOUS RECEIVE REQUIREMENTS**

| **Standard Operating Conditions (unless otherwise stated)**<br>Operating Temperature -40°C ≤ TA ≤ +125°C | | | | | | |
|---|---|---|---|---|---|---|
| **Param. No.** | **Symbol** | **Characteristic** | **Min.** | **Max.** | **Units** | **Conditions** |
| US125 | TDTV2CKL | SYNC RCV (Master and Slave)<br>Data-hold before CK ↓ (DT hold time) | 10 | — | ns | |
| US126 | TCKL2DTL | Data-hold after CK ↓ (DT hold time) | 15 | — | ns | |

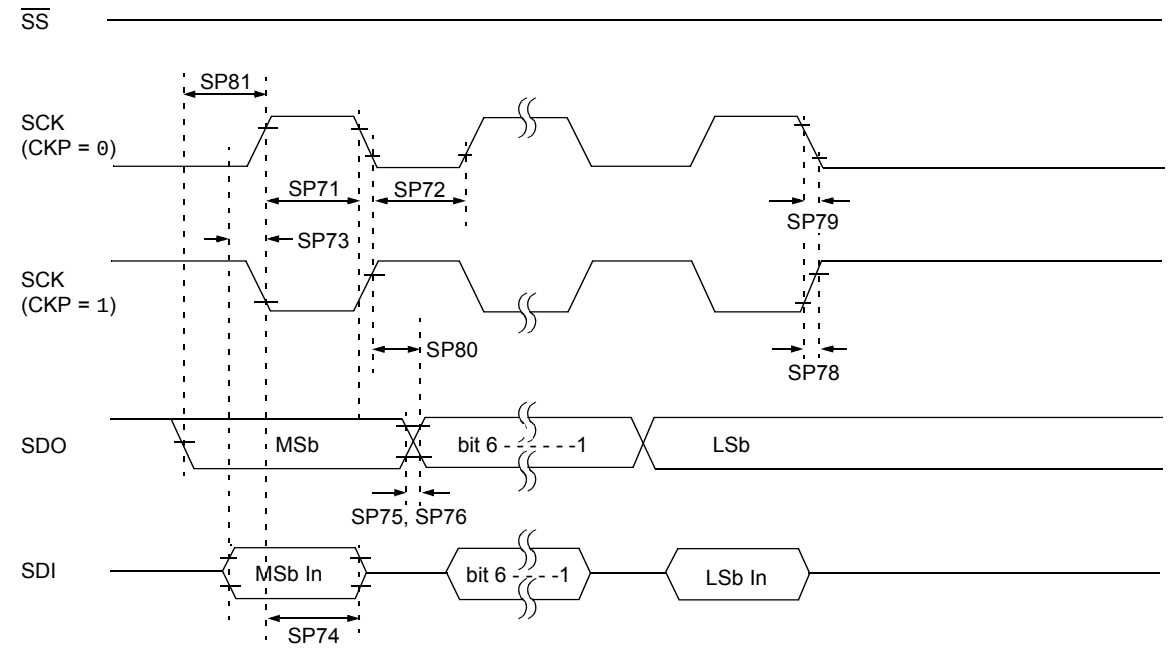

© 2010-2016 Microchip Technology Inc.

**FIGURE 30-16: SPI MASTER MODE TIMING (CKE = 0, SMP = 0)**



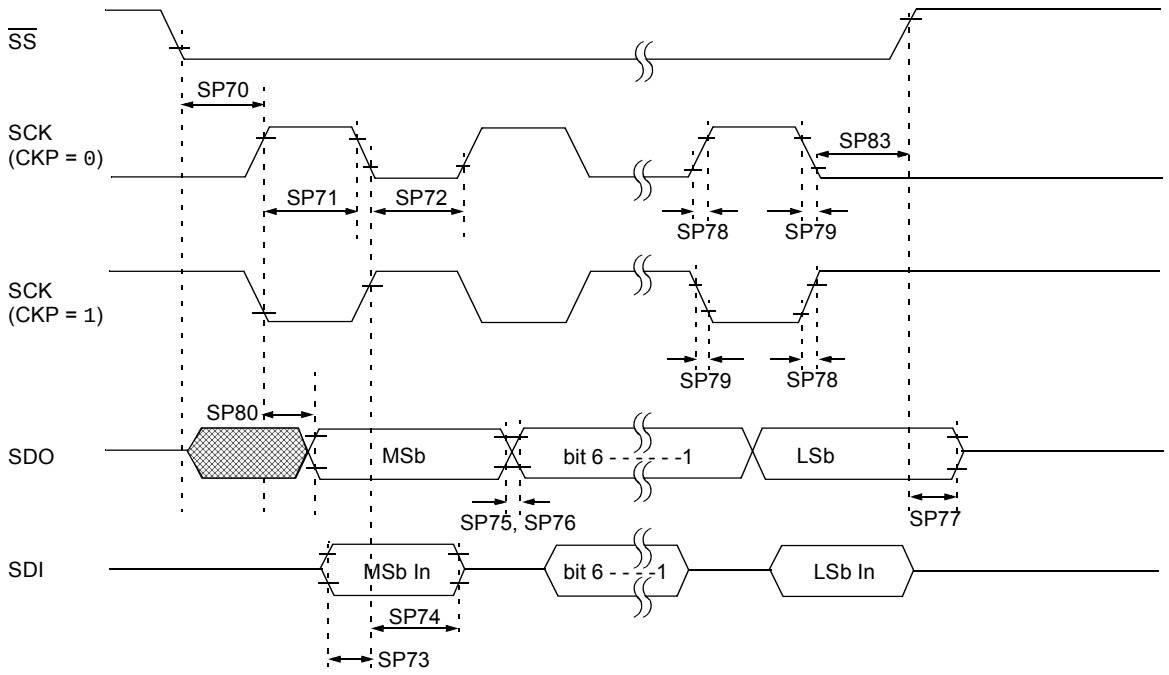

**Note:** Refer to Figure 30-5 for load conditions.

**FIGURE 30-17: SPI MASTER MODE TIMING (CKE = 1, SMP = 1)**



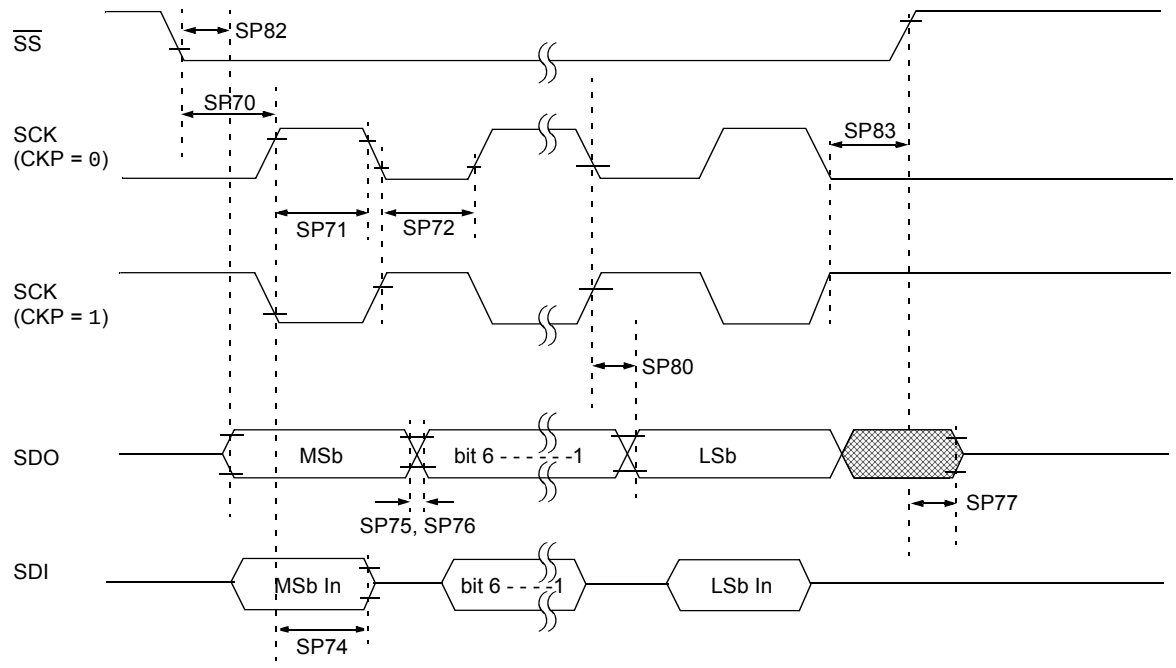

**Note:** Refer to Figure 30-5 for load conditions.

**FIGURE 30-18: SPI SLAVE MODE TIMING (CKE = 0)**




**Note:** Refer to Figure 30-5 for load conditions.

**FIGURE 30-19: SPI SLAVE MODE TIMING (CKE = 1)**




**Note:** Refer to Figure 30-5 for load conditions.


© 2010-2016 Microchip Technology Inc.

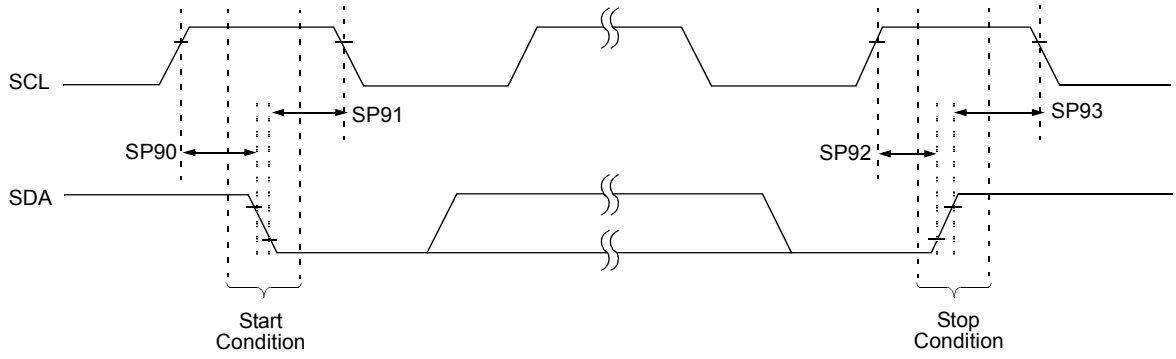
**TABLE 30-14: SPI MODE REQUIREMENTS**

| Param No. | Symbol | Characteristic | | Min. | Typ† | Max. | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| SP70* | TssL2scH, TssL2scL | $\overline{SS}$↓ to SCK↓ or SCK↑ input | | 2.25 Tcy | — | — | ns | |
| SP71* | TscH | SCK input high time (Slave mode) | | Tcy + 20 | — | — | ns | |
| SP72* | TscL | SCK input low time (Slave mode) | | Tcy + 20 | — | — | ns | |
| SP73* | TdiV2scH, TdiV2scL | Setup time of SDI data input to SCK edge | | 100 | — | — | ns | |
| SP74* | TscH2diL, TscL2diL | Hold time of SDI data input to SCK edge | | 100 | — | — | ns | |
| SP75* | TdoR | SDO data output rise time | 3.0-5.5V | — | 10 | 25 | ns | |
| | | | 1.8-5.5V | — | 25 | 50 | ns | |
| SP76* | TdoF | SDO data output fall time | | — | 10 | 25 | ns | |
| SP77* | TssH2doZ | $\overline{SS}$↑ to SDO output high-impedance | | 10 | — | 50 | ns | |
| SP78* | TscR | SCK output rise time (Master mode) | 3.0-5.5V | — | 10 | 25 | ns | |
| | | | 1.8-5.5V | — | 25 | 50 | ns | |
| SP79* | TscF | SCK output fall time (Master mode) | | — | 10 | 25 | ns | |
| SP80* | TscH2doV, TscL2doV | SDO data output valid after SCK edge | 3.0-5.5V | — | — | 50 | ns | |
| | | | 1.8-5.5V | — | — | 145 | ns | |
| SP81* | TdoV2scH, TdoV2scL | SDO data output setup to SCK edge | | Tcy | — | — | ns | |
| SP82* | TssL2doV | SDO data output valid after $\overline{SS}$↓ edge | | — | — | 50 | ns | |
| SP83* | TscH2ssH, TscL2ssH | $\overline{SS}$ ↑ after SCK edge | | 1.5 Tcy + 40 | — | — | ns | |

* 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 30-20: $I^2C$ BUS START/STOP BITS TIMING**



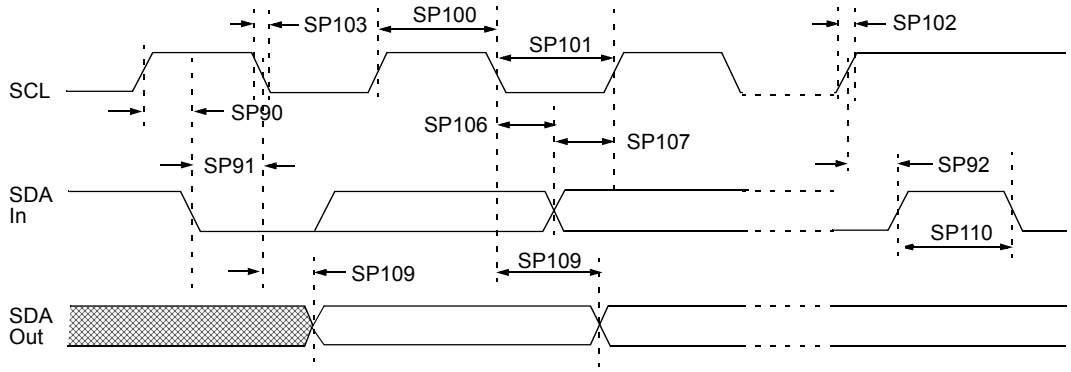

**Note**: Refer to Figure 30-5 for load conditions.

**TABLE 30-15: I²C BUS START/STOP BITS REQUIREMENTS**

| Param No. | Symbol | Characteristic | | Min. | Typ | Max. | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| SP90* | TSU:STA | Start condition Setup time | 100 kHz mode | 4700 | — | — | ns | Only relevant for Repeated Start condition |
| | | | 400 kHz mode | 600 | — | — | | |
| SP91* | THD:STA | Start condition Hold time | 100 kHz mode | 4000 | — | — | ns | After this period, the first clock pulse is generated |
| | | | 400 kHz mode | 600 | — | — | | |
| SP92* | TSU:STO | Stop condition Setup time | 100 kHz mode | 4700 | — | — | ns | |
| | | | 400 kHz mode | 600 | — | — | | |
| SP93 | THD:STO | Stop condition Hold time | 100 kHz mode | 4000 | — | — | ns | |
| | | | 400 kHz mode | 600 | — | — | | |

* These parameters are characterized but not tested.

**FIGURE 30-21: I²C BUS DATA TIMING**




**Note:** Refer to Figure 30-5 for load conditions.


© 2010-2016 Microchip Technology Inc.

**TABLE 30-16: $I^2C$ BUS DATA REQUIREMENTS**

| Param. No. | Symbol | Characteristic | | Min. | Max. | Units | Conditions |
|---|---|---|---|---|---|---|---|
| SP100* | THIGH | Clock high time | 100 kHz mode | 4.0 | — | µs | Device must operate at a minimum of 1.5 MHz |
| | | | 400 kHz mode | 0.6 | — | µs | Device must operate at a minimum of 10 MHz |
| | | | SSP module | 1.5 TCY | — | | |
| SP101* | TLOW | Clock low time | 100 kHz mode | 4.7 | — | µs | Device must operate at a minimum of 1.5 MHz |
| | | | 400 kHz mode | 1.3 | — | µs | Device must operate at a minimum of 10 MHz |
| | | | SSP module | 1.5 TCY | — | | |
| SP102* | TR | SDA and SCL rise time | 100 kHz mode | — | 1000 | ns | |
| | | | 400 kHz mode | 20 + 0.1 CB | 300 | ns | CB is specified to be from 10-400 pF |
| SP103* | TF | SDA and SCL fall time | 100 kHz mode | — | 250 | ns | |
| | | | 400 kHz mode | 20 + 0.1 CB | 250 | ns | CB is specified to be from 10-400 pF |
| SP106* | THD:DAT | Data input hold time | 100 kHz mode | 0 | — | ns | |
| | | | 400 kHz mode | 0 | 0.9 | µs | |
| SP107* | TSU:DAT | Data input setup time | 100 kHz mode | 250 | — | ns | **(Note 2)** |
| | | | 400 kHz mode | 100 | — | ns | |
| SP109* | TAA | Output valid from clock | 100 kHz mode | — | 3500 | ns | **(Note 1)** |
| | | | 400 kHz mode | — | — | ns | |
| SP110* | TBUF | Bus free time | 100 kHz mode | 4.7 | — | µs | Time the bus must be free before a new transmission can start |
| | | | 400 kHz mode | 1.3 | — | µs | |
| SP111 | CB | Bus capacitive loading | | — | 400 | pF | |

* These parameters are characterized but not tested.

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

**2:** A Fast mode (400 kHz) $I^2C$ bus device can be used in a Standard mode (100 kHz) $I^2C$ bus system, but the requirement $T_{SU:DAT} \geq 250$ ns must then be met. This will automatically be the case if the device does not stretch the low period of the SCL signal. If such a device does stretch the low period of the SCL signal, it must output the next data bit to the SDA line $T_R$ max. + $T_{SU:DAT}$ = 1000 + 250 = 1250 ns (according to the Standard mode $I^2C$ bus specification), before the SCL line is released.

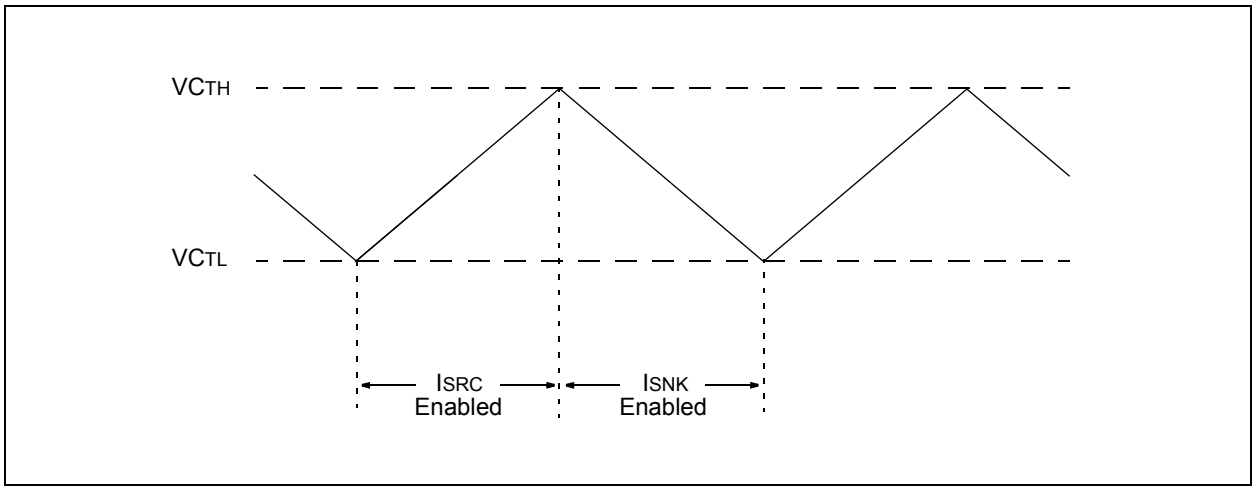
**TABLE 30-17: CAP SENSE OSCILLATOR SPECIFICATIONS**

| Param. No. | Symbol | Characteristic | | Min. | Typ† | Max. | Units | Conditions |
|---|---|---|---|---|---|---|---|---|
| CS01* | $I_{SRC}$ | Current Source | High | — | -8 | — | µA | |
| | | | Medium | — | -1.5 | — | µA | |
| | | | Low | — | -0.3 | — | µA | |
| CS02* | $I_{SNK}$ | Current Sink | High | — | 7.5 | — | µA | |
| | | | Medium | — | 1.5 | — | µA | |
| | | | Low | — | 0.25 | — | µA | |
| CS03* | $V_{CTH}$ | Cap Threshold | | — | 0.8 | — | V | |
| CS04* | $V_{CTL}$ | Cap Threshold | | — | 0.4 | — | V | |
| CS05* | $V_{CHYST}$ | Cap Hysteresis ($V_{CTH}$-$V_{CTL}$) | High<br>Medium<br>Low | —<br>—<br>— | 525<br>375<br>300 | —<br>—<br>— | mV<br>mV<br>mV | |

* 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 30-22: CAP SENSE OSCILLATOR**





© 2010-2016 Microchip Technology Inc.

## 31.0 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 $V_{DD}$ range). This is for **information only** and devices are ensured to operate properly only within the specified range.

| | |
|---|---|
| **Note:** | 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. |

**"Typical" represents the mean of the distribution at 25°C. "MAXIMUM", "Max.", "MINIMUM" or "Min." represents (mean + 3σ) or (mean - 3σ) respectively, where σ is a standard deviation, over each temperature range.**

**FIGURE 31-1: IDD, LP OSCILLATOR MODE, FOSC = 32 kHz, PIC16LF1946/47 ONLY**

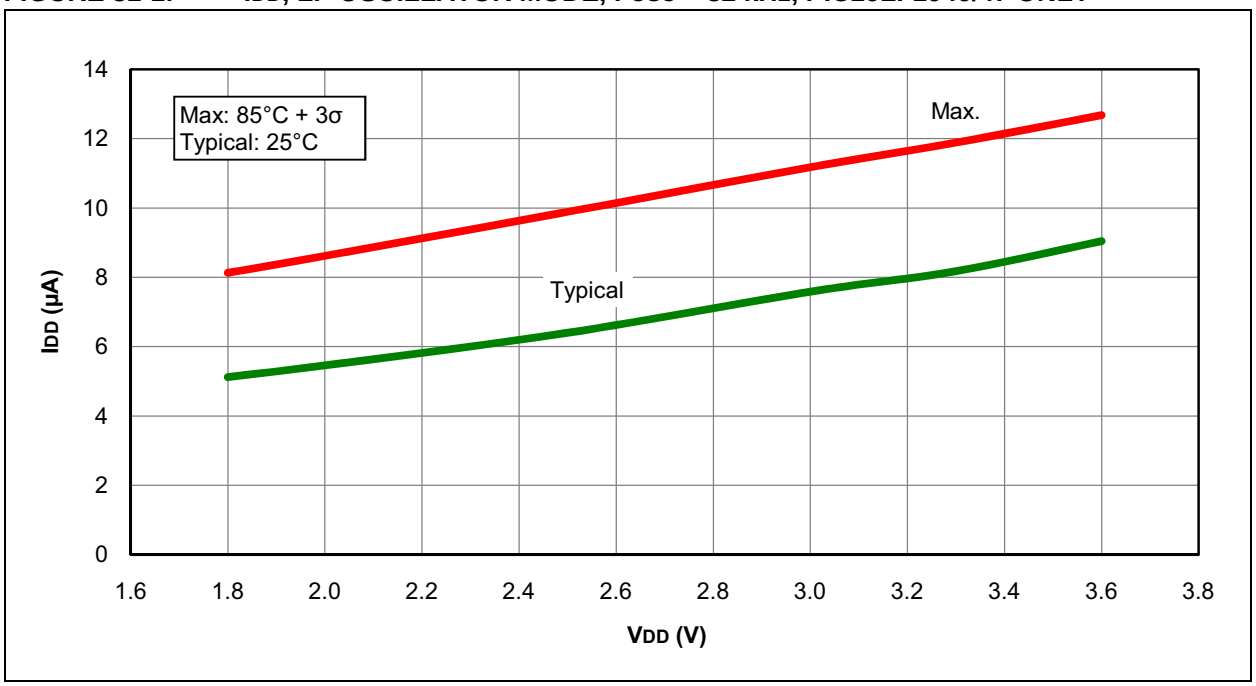
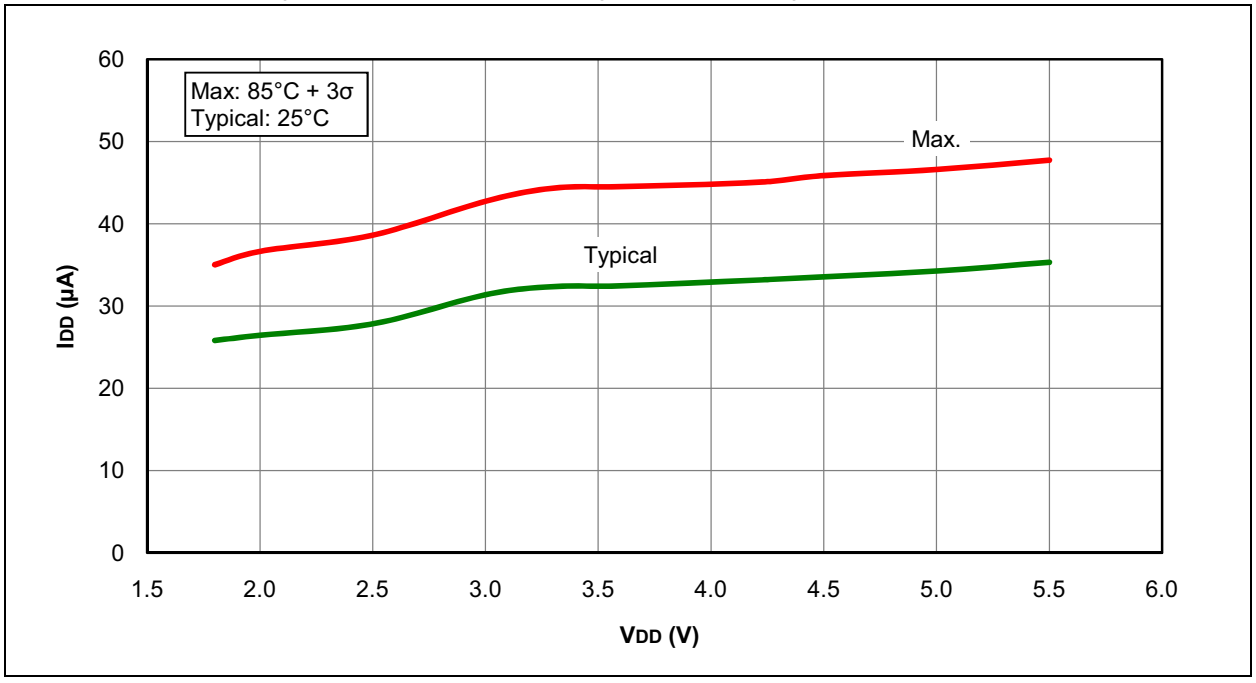



**FIGURE 31-2: IDD, LP OSCILLATOR MODE, FOSC = 32 kHz, PIC16F1946/47 ONLY**





© 2010-2016 Microchip Technology Inc.

**FIGURE 31-3: IDD TYPICAL, XT AND EXTRC OSCILLATOR, PIC16LF1946/47 ONLY**

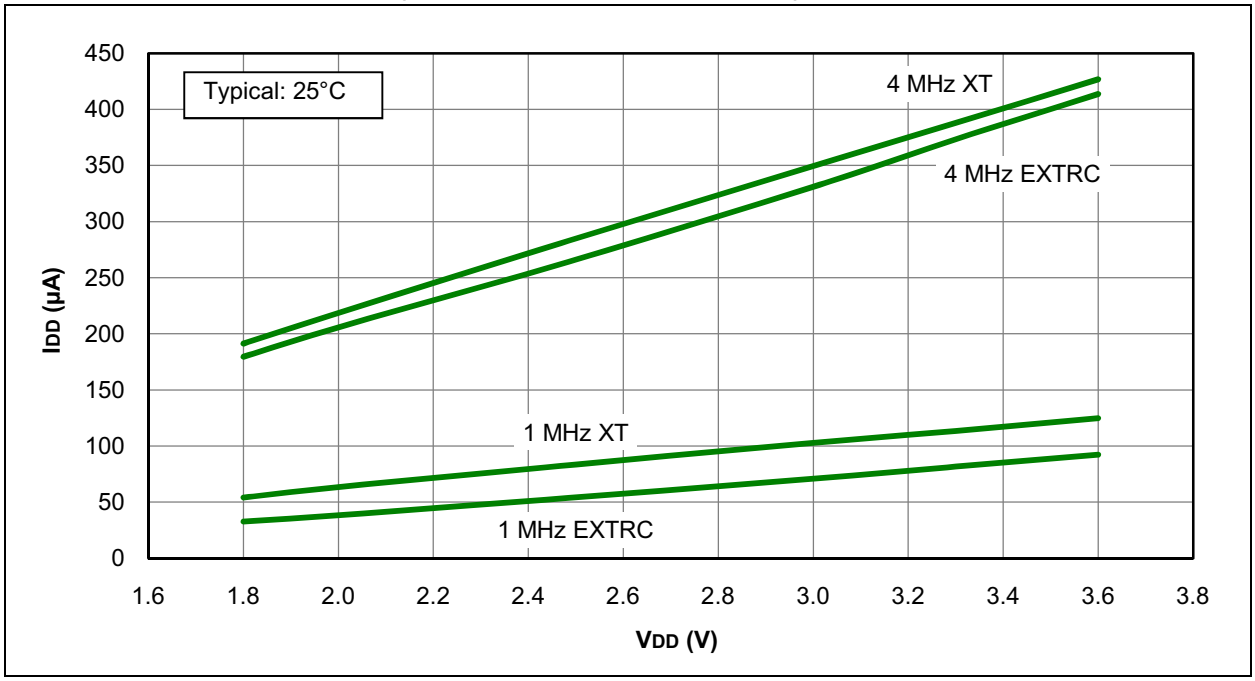



**FIGURE 31-4: IDD MAXIMUM, XT AND EXTRC OSCILLATOR, PIC16LF1946/47 ONLY**

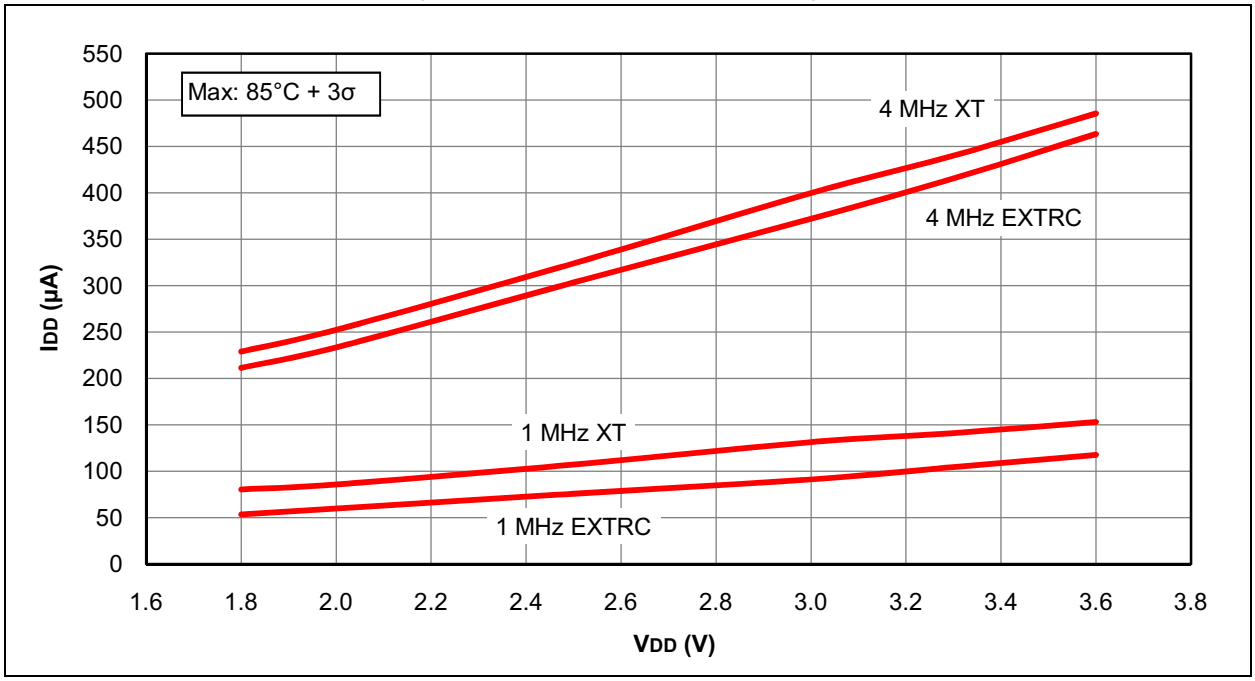


**FIGURE 31-5: IDD TYPICAL, XT AND EXTRC OSCILLATOR, PIC16F1946/47 ONLY**

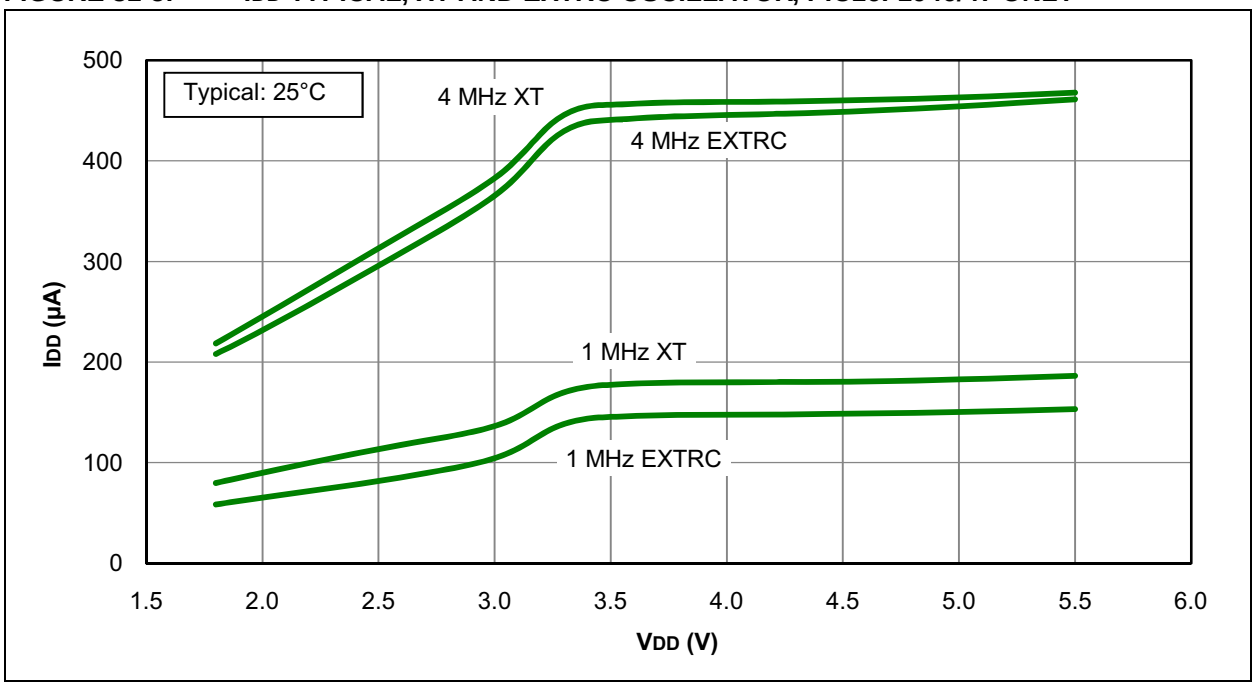
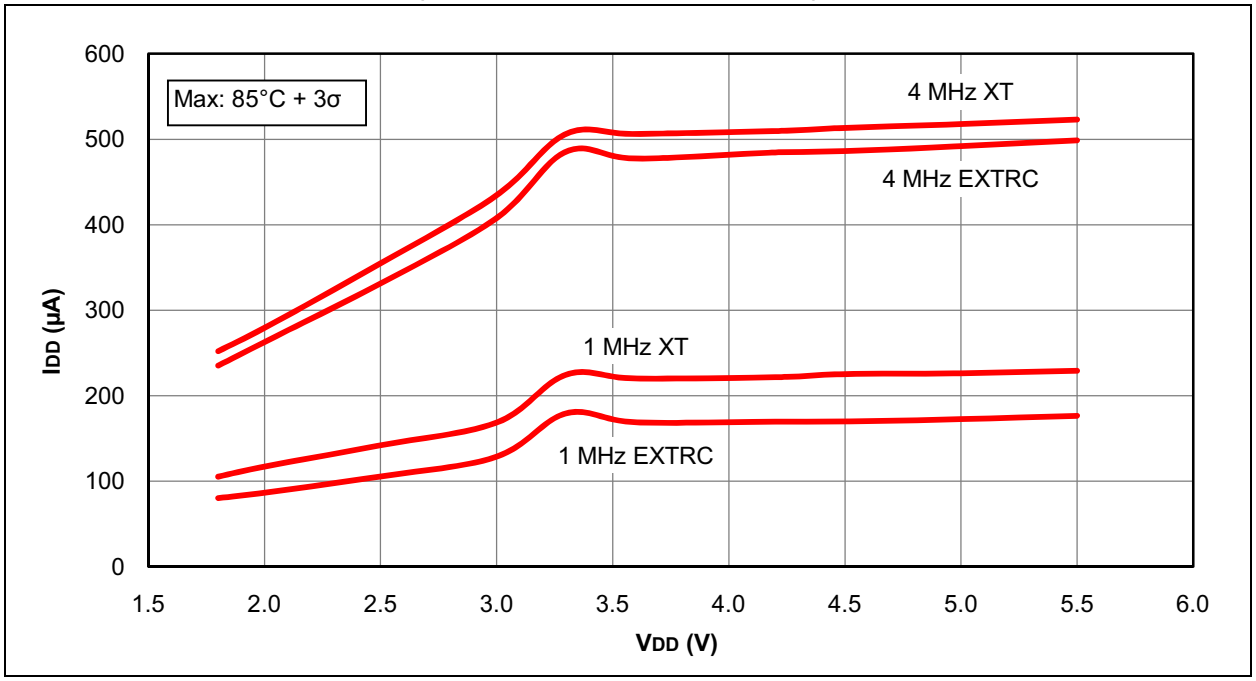



**FIGURE 31-6: IDD MAXIMUM, XT AND EXTRC OSCILLATOR, PIC16F1946/47 ONLY**





© 2010-2016 Microchip Technology Inc.

**FIGURE 31-7: IDD, EC OSCILLATOR, LOW-POWER MODE, FOSC = 32 kHz, PIC16LF1946/47 ONLY**

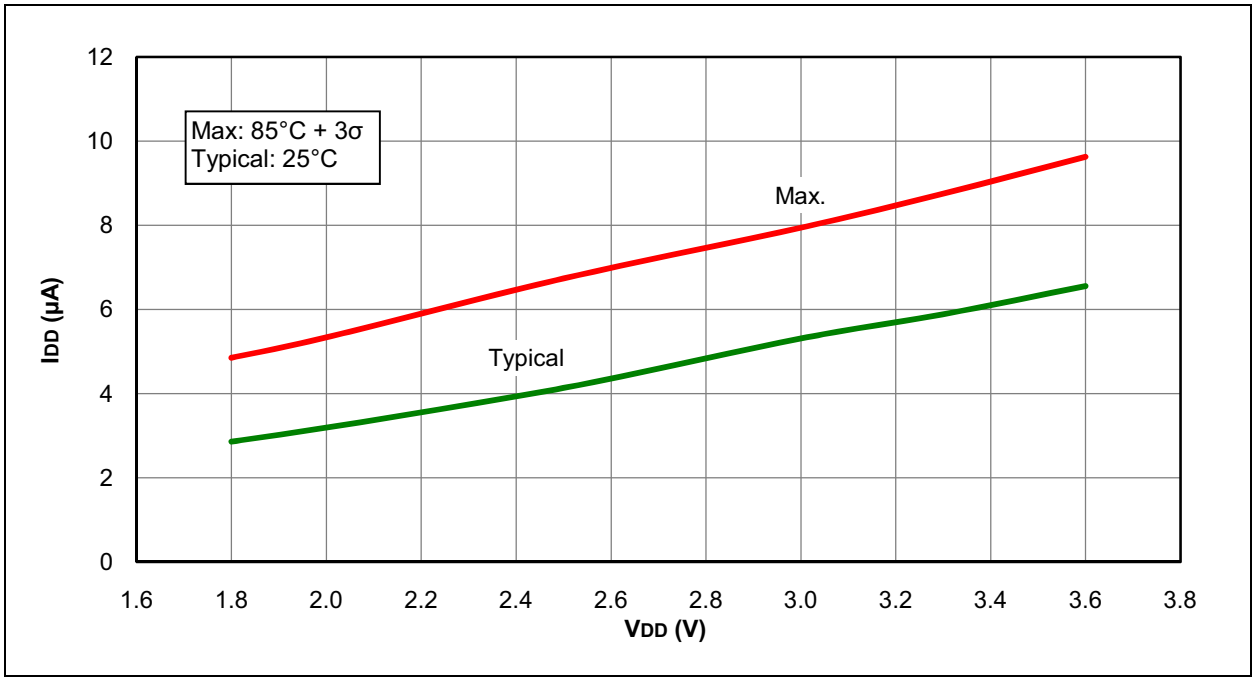



**FIGURE 31-8: IDD, EC OSCILLATOR, LOW-POWER MODE, FOSC = 32 kHz, PIC16F1946/47 ONLY**

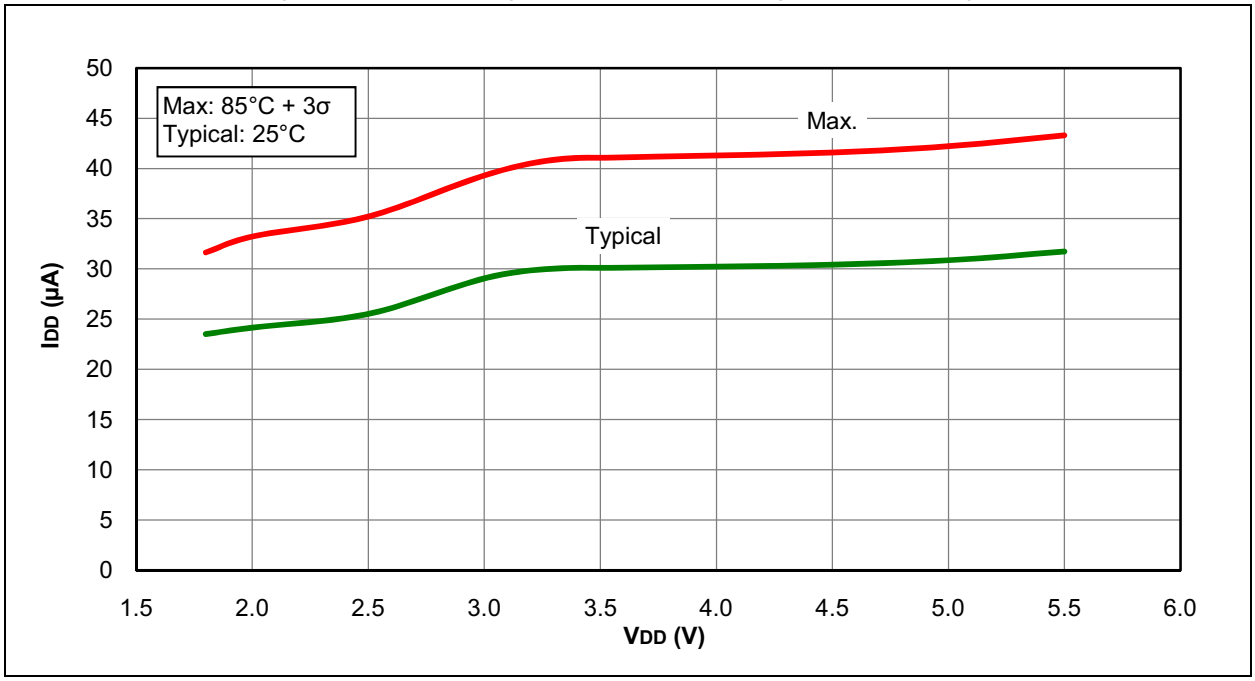


**FIGURE 31-9: IDD, EC OSCILLATOR, LOW-POWER MODE, FOSC = 500 kHz, PIC16LF1946/47 ONLY**

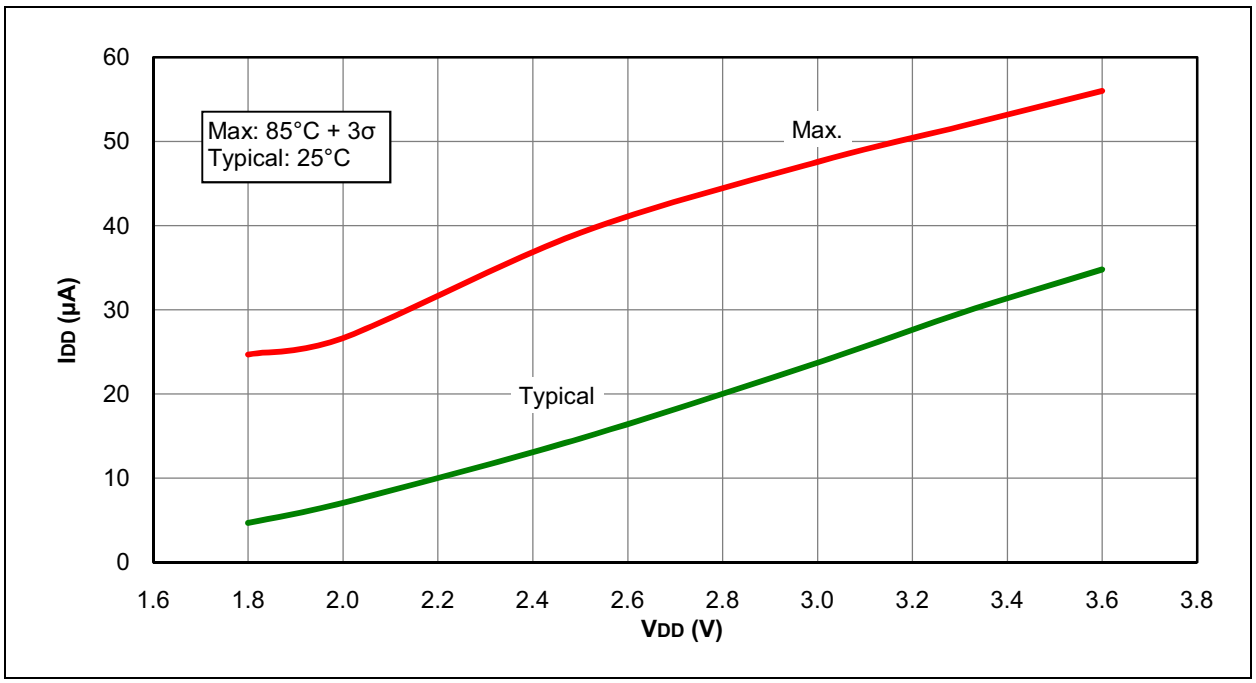
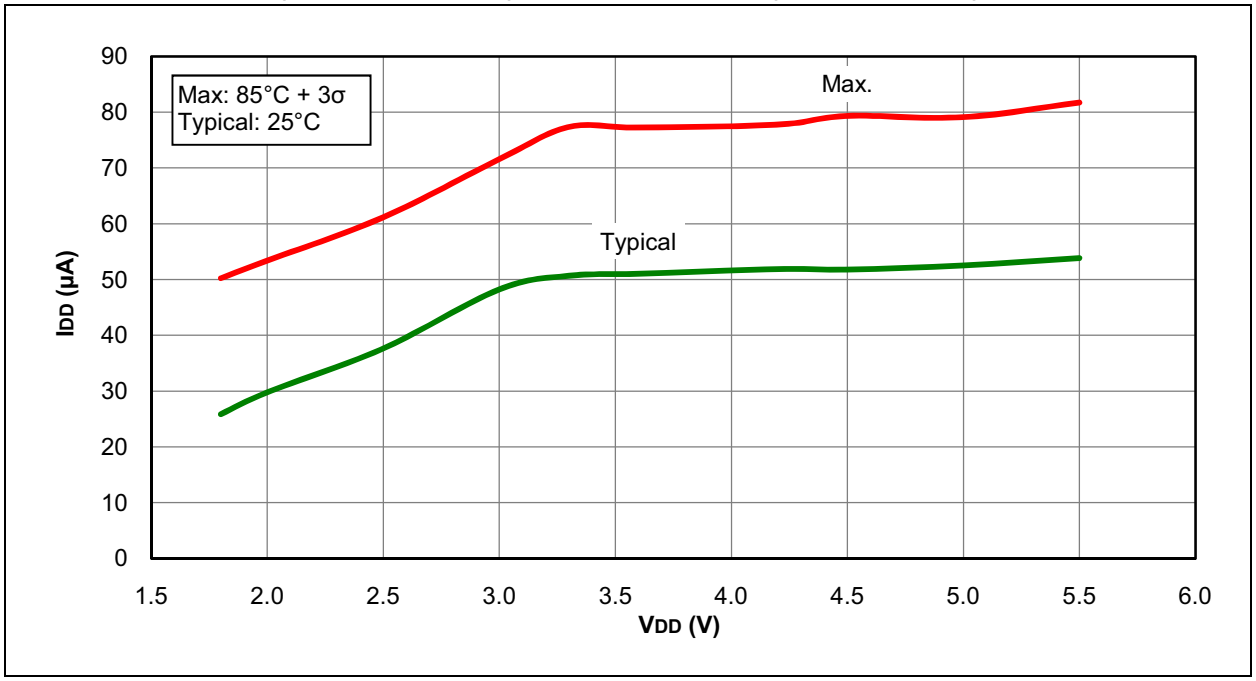



**FIGURE 31-10: IDD, EC OSCILLATOR, LOW-POWER MODE, FOSC = 500 kHz, PIC16F1946/47 ONLY**





© 2010-2016 Microchip Technology Inc.

**FIGURE 31-11: IDD TYPICAL, EC OSCILLATOR, MEDIUM-POWER MODE, PIC16LF1946/47 ONLY**

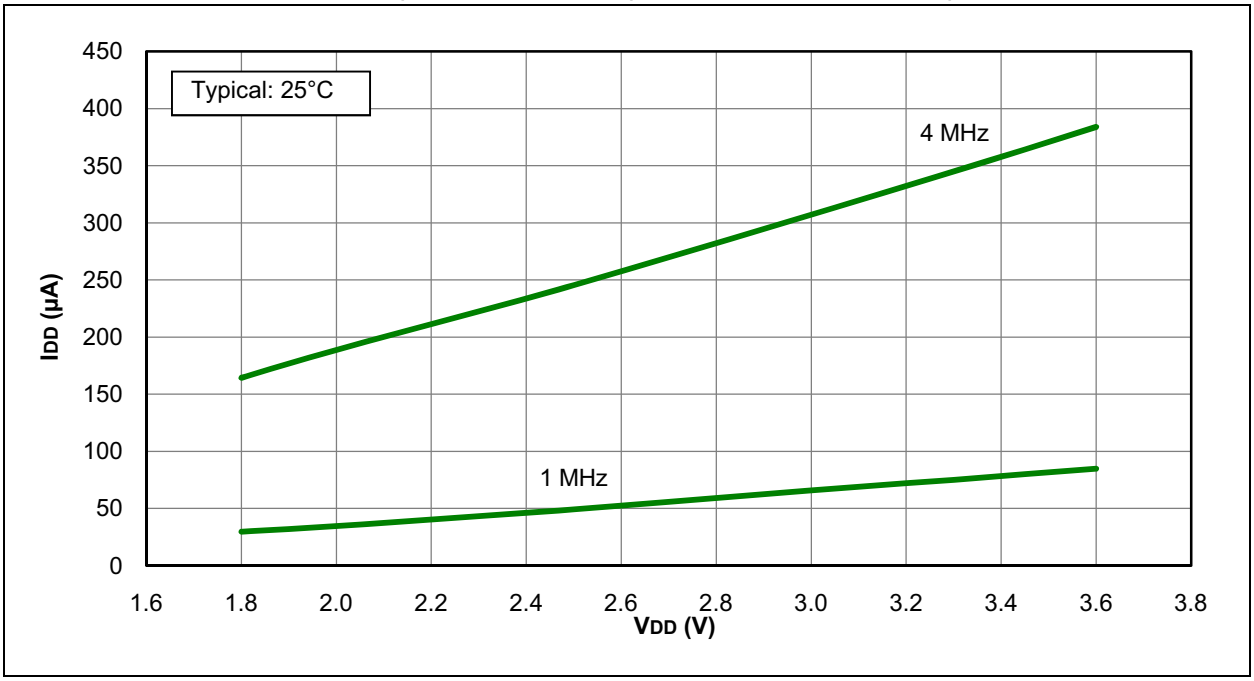



**FIGURE 31-12: IDD MAXIMUM, EC OSCILLATOR, MEDIUM-POWER MODE, PIC16LF1946/47 ONLY**

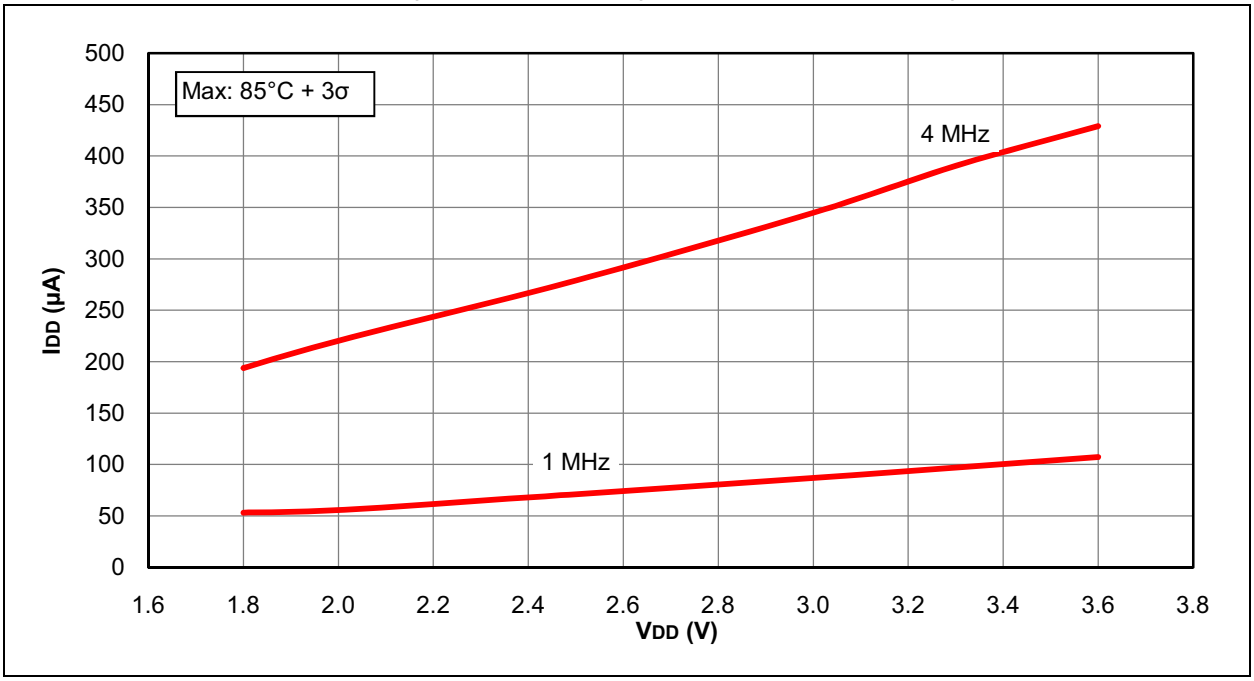


**FIGURE 31-13: IDD TYPICAL, EC OSCILLATOR, MEDIUM-POWER MODE, PIC16F1946/47 ONLY**

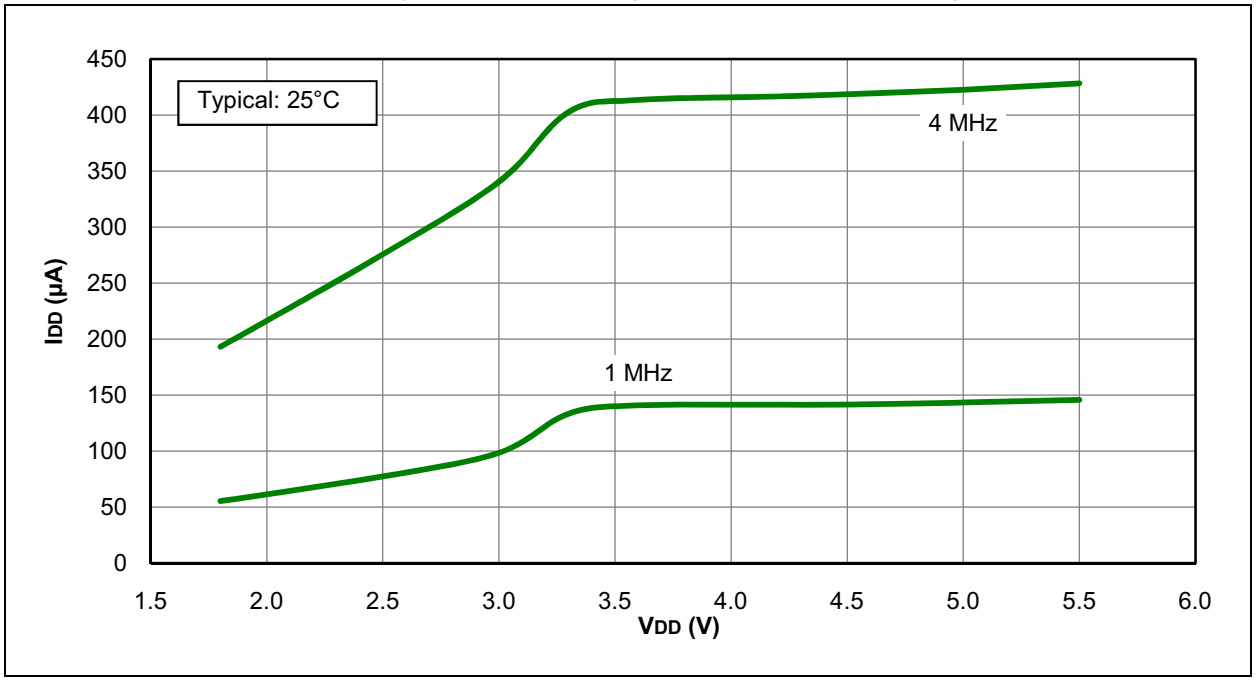
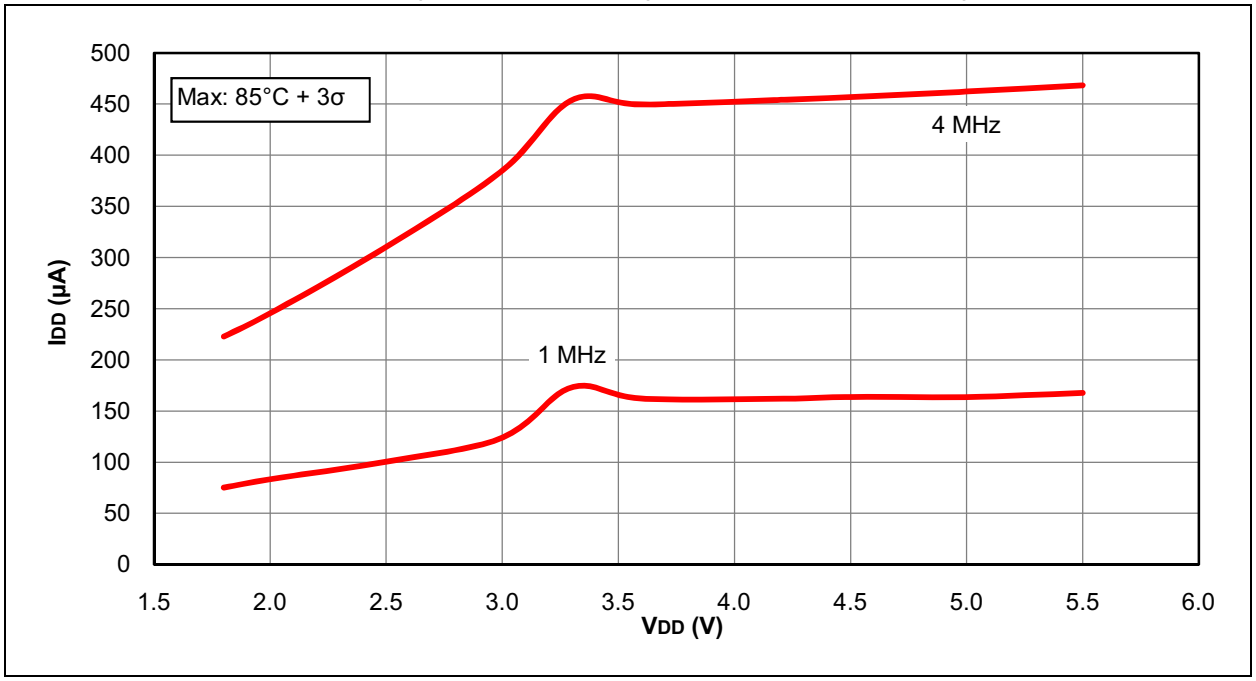



**FIGURE 31-14: IDD MAXIMUM, EC OSCILLATOR, MEDIUM-POWER MODE, PIC16F1946/47 ONLY**





© 2010-2016 Microchip Technology Inc.

**FIGURE 31-15: IDD TYPICAL, EC OSCILLATOR, HIGH-POWER MODE, PIC16LF1946/47 ONLY**

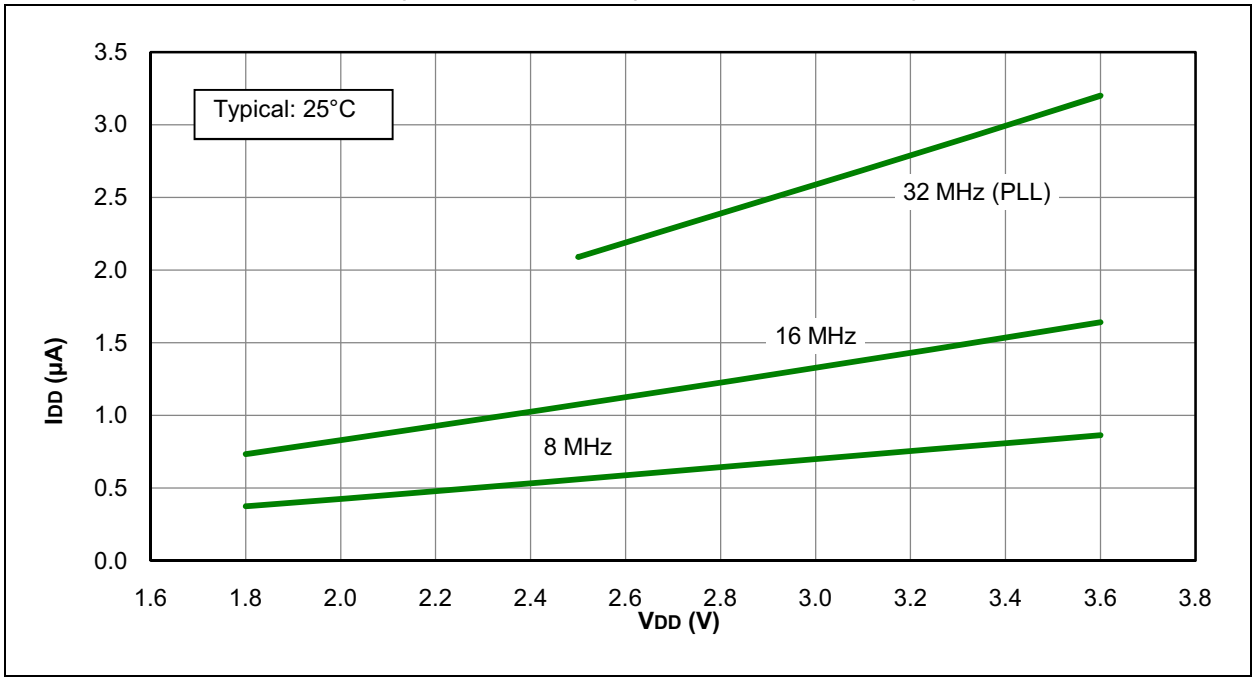



**FIGURE 31-16: IDD MAXIMUM, EC OSCILLATOR, HIGH-POWER MODE, PIC16LF1946/47 ONLY**

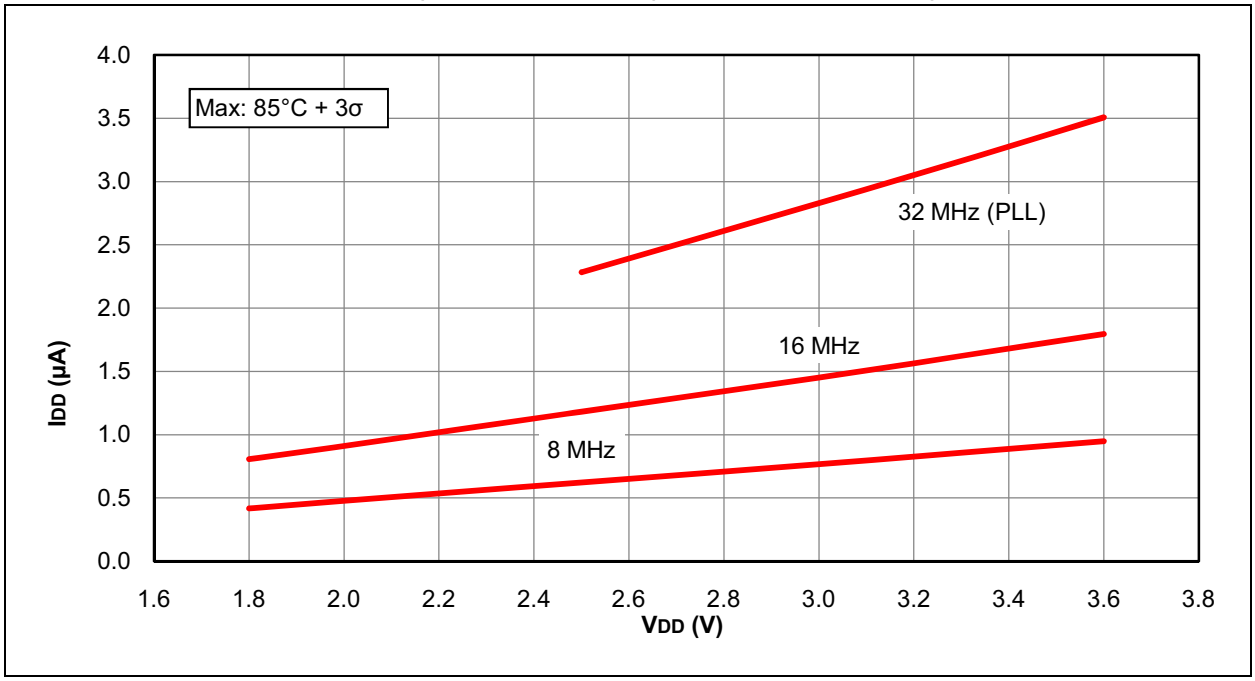


**FIGURE 31-17: IDD TYPICAL, EC OSCILLATOR, HIGH-POWER MODE, PIC16F1946/47 ONLY**

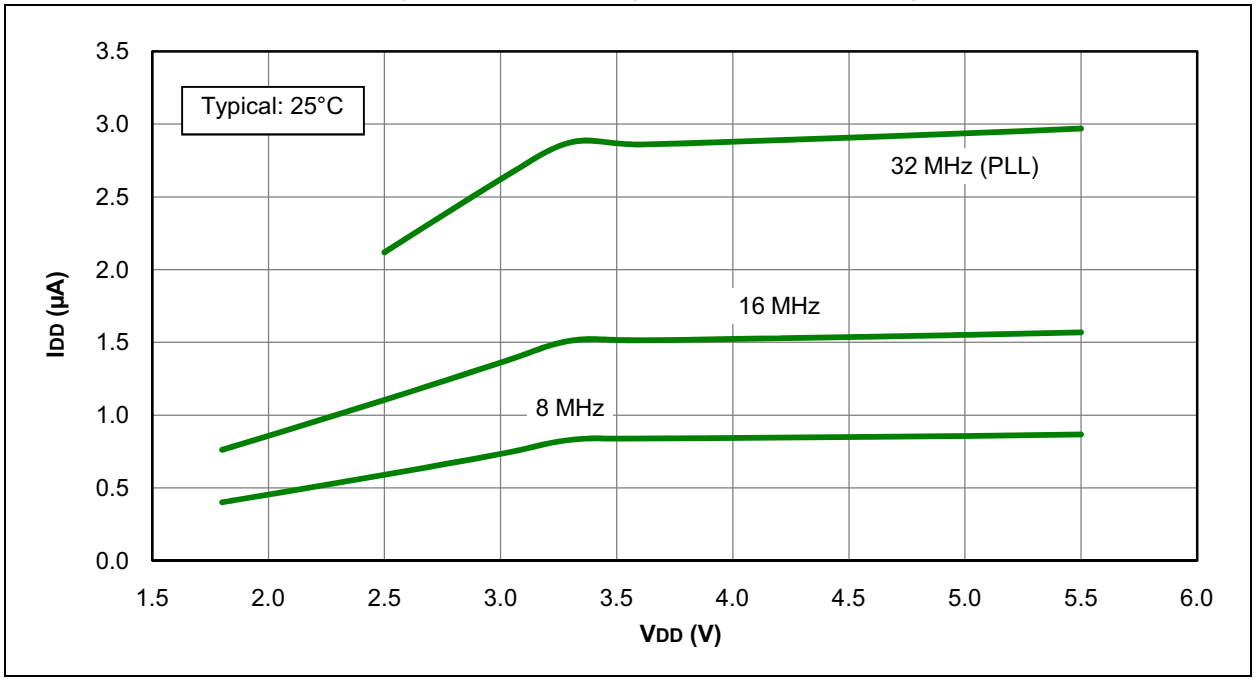
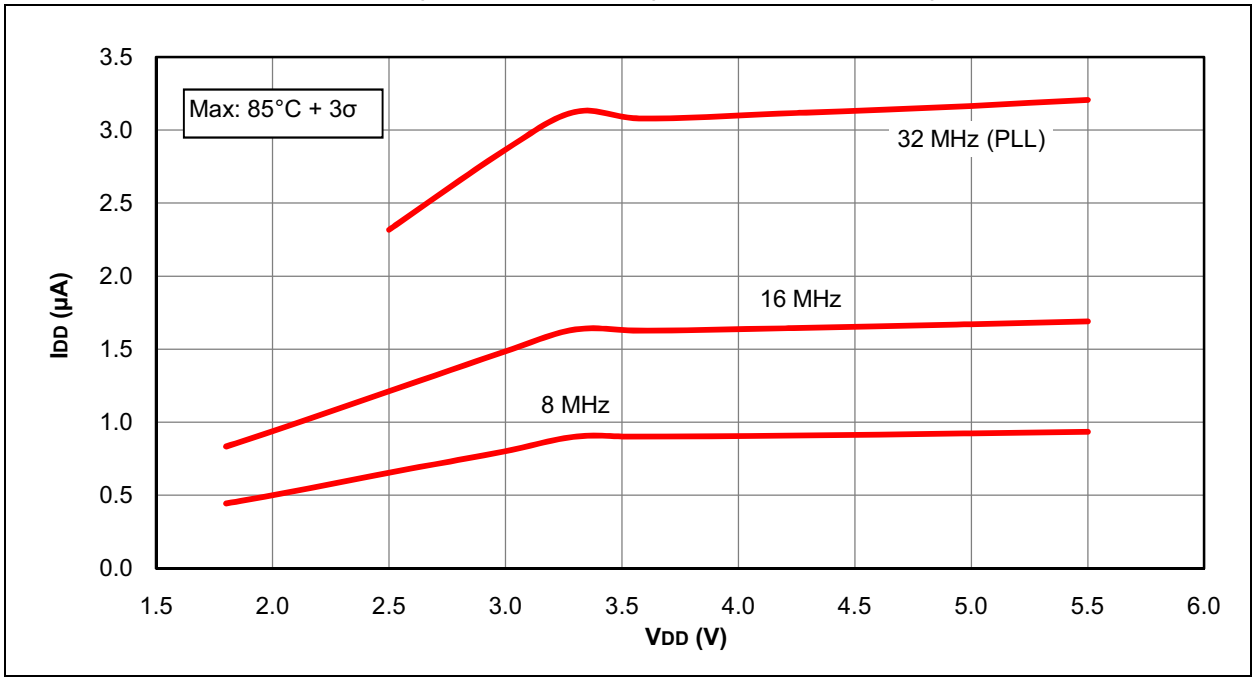



**FIGURE 31-18: IDD MAXIMUM, EC OSCILLATOR, HIGH-POWER MODE, PIC16F1946/47 ONLY**





© 2010-2016 Microchip Technology Inc.

**FIGURE 31-19: IDD, LFINTOSC MODE, FOSC = 32 kHz, PIC16LF1946/47 ONLY**

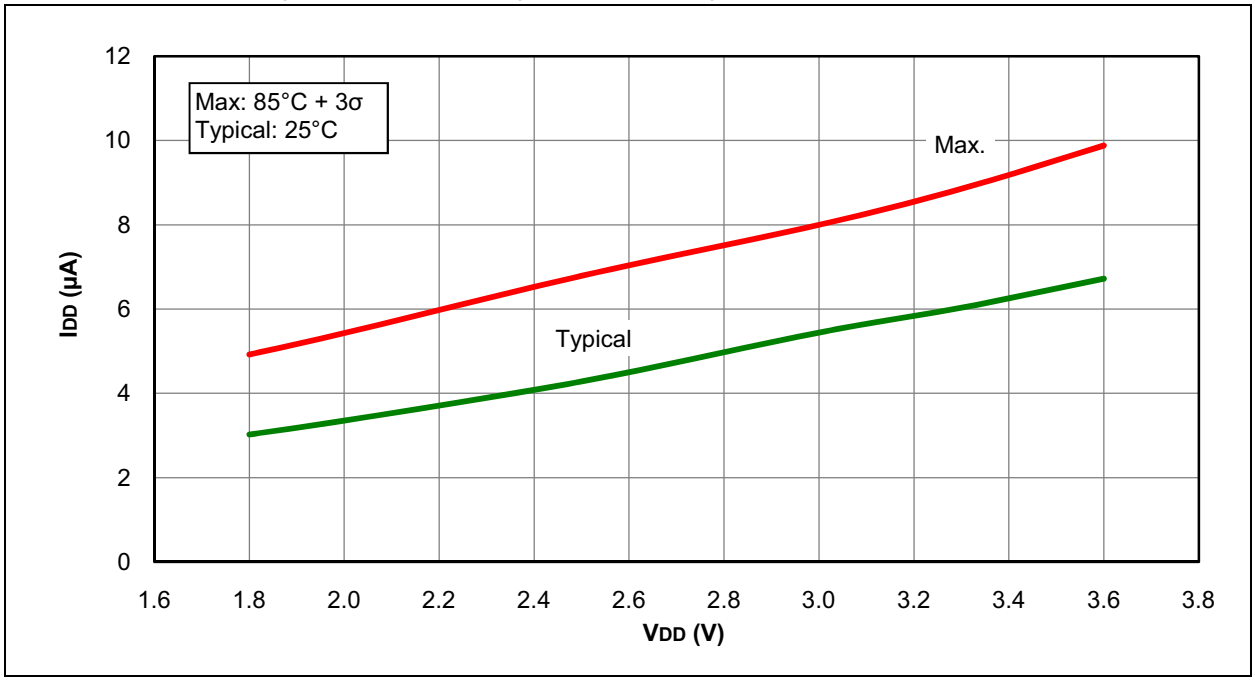



**FIGURE 31-20: IDD, LFINTOSC MODE, FOSC = 32 kHz, PIC16F1946/47 ONLY**

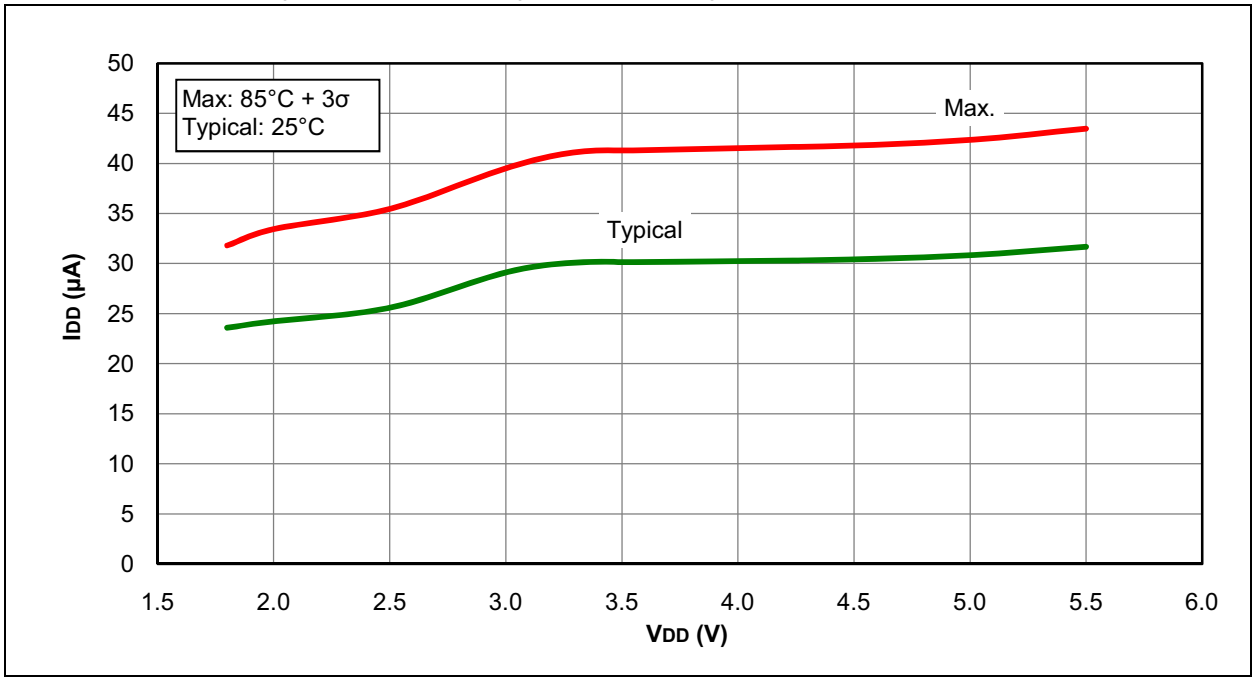


**FIGURE 31-21: IDD, MFINTOSC MODE, FOSC = 500 kHz, PIC16LF1946/47 ONLY**

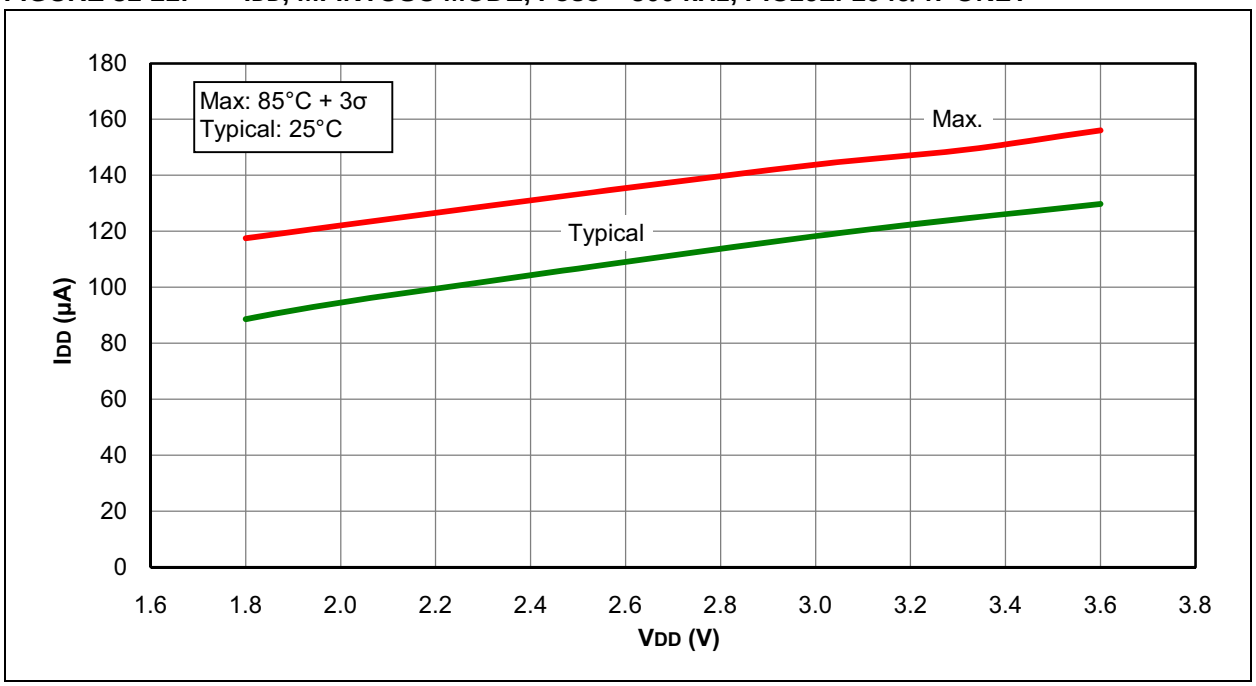
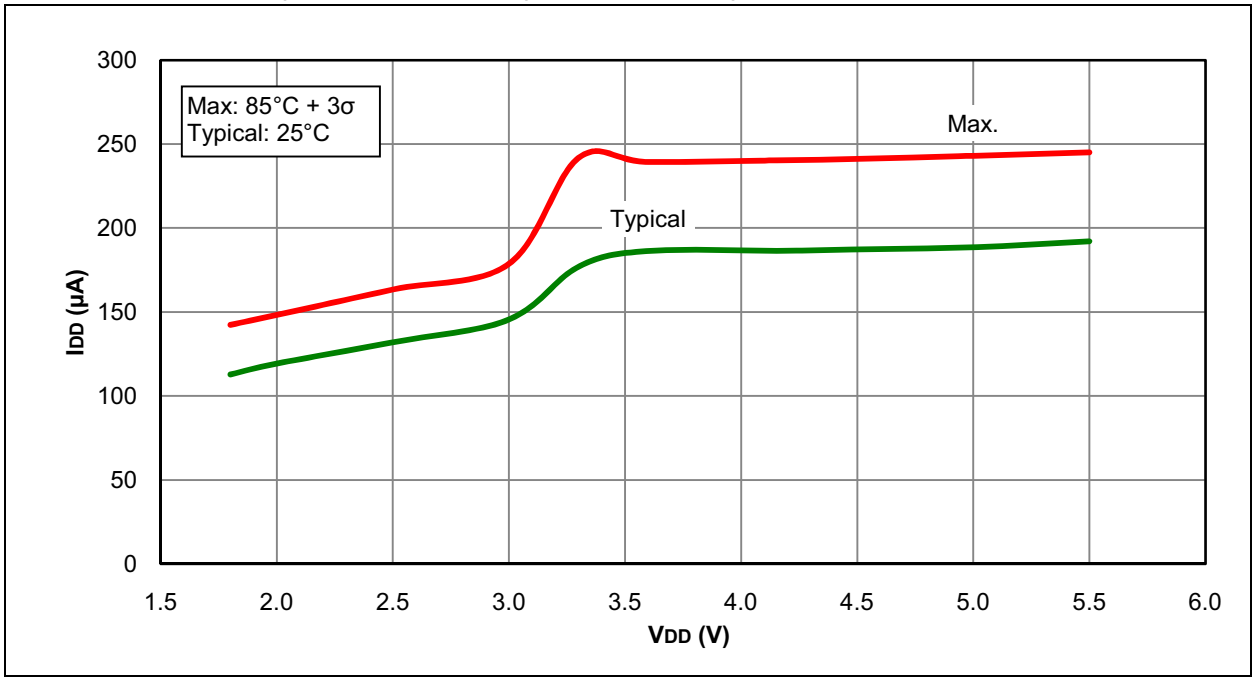



**FIGURE 31-22: IDD, MFINTOSC MODE, FOSC = 500 kHz, PIC16F1946/47 ONLY**





© 2010-2016 Microchip Technology Inc.

**FIGURE 31-23: IDD TYPICAL, HFINTOSC MODE, PIC16LF1946/47 ONLY**

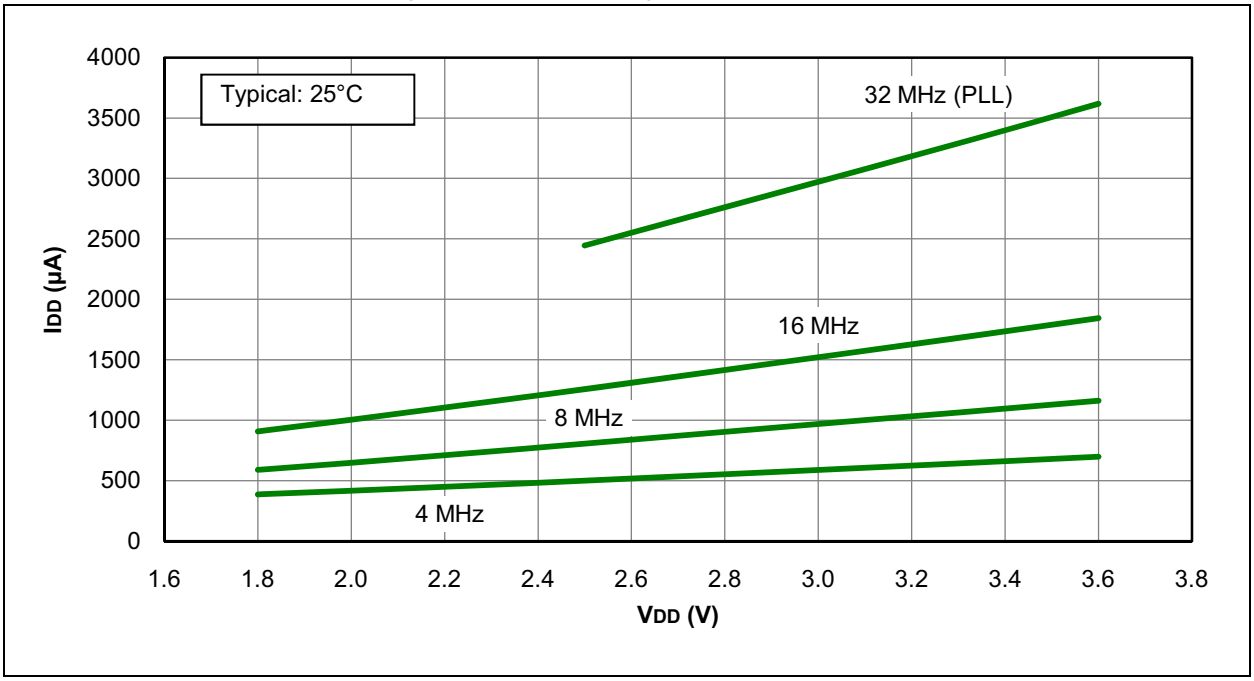



**FIGURE 31-24: IDD MAXIMUM, HFINTOSC MODE, PIC16LF1946/47 ONLY**

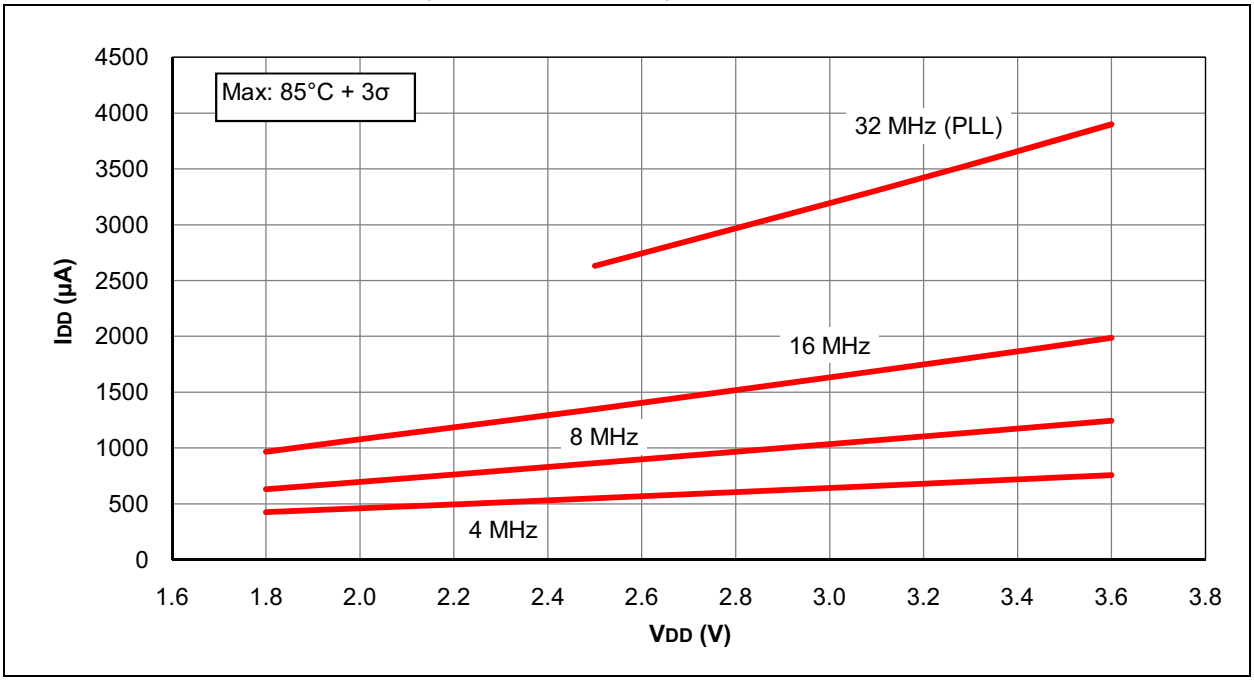


**FIGURE 31-25: IDD TYPICAL, HFINTOSC MODE, PIC16F1946/47 ONLY**

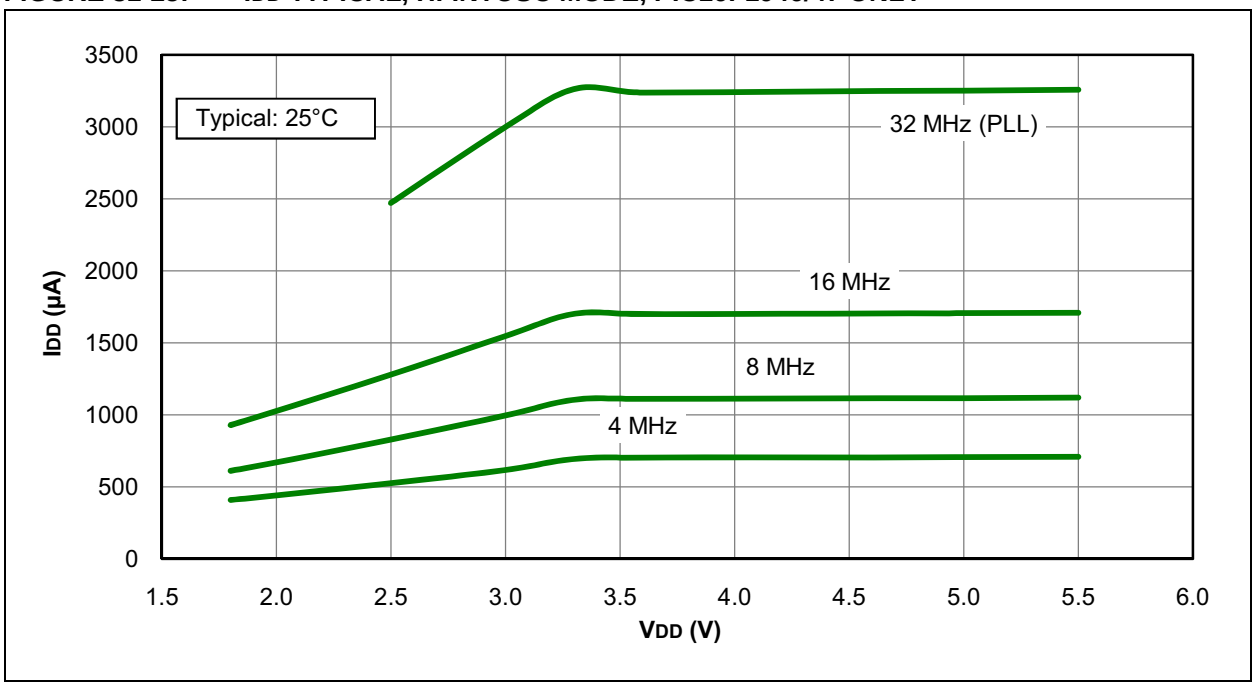
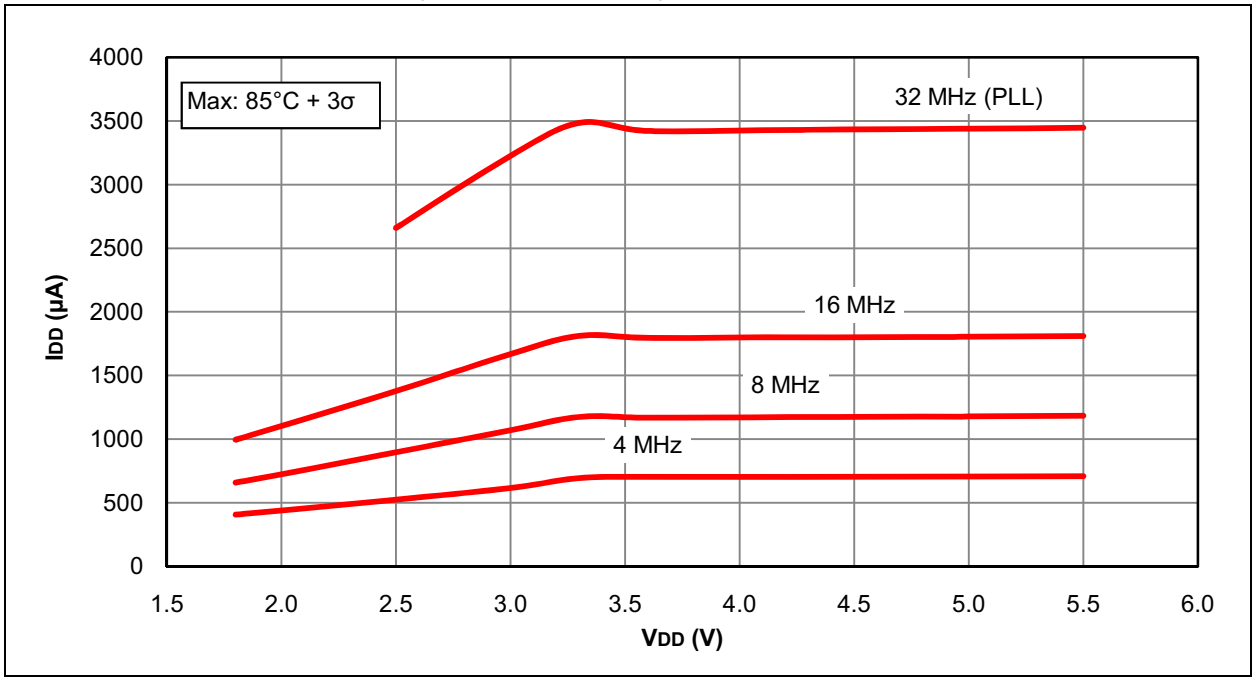



**FIGURE 31-26: IDD MAXIMUM, HFINTOSC MODE, PIC16F1946/47 ONLY**





© 2010-2016 Microchip Technology Inc.

**FIGURE 31-27: IDD TYPICAL, HS OSCILLATOR, PIC16LF1946/47 ONLY**

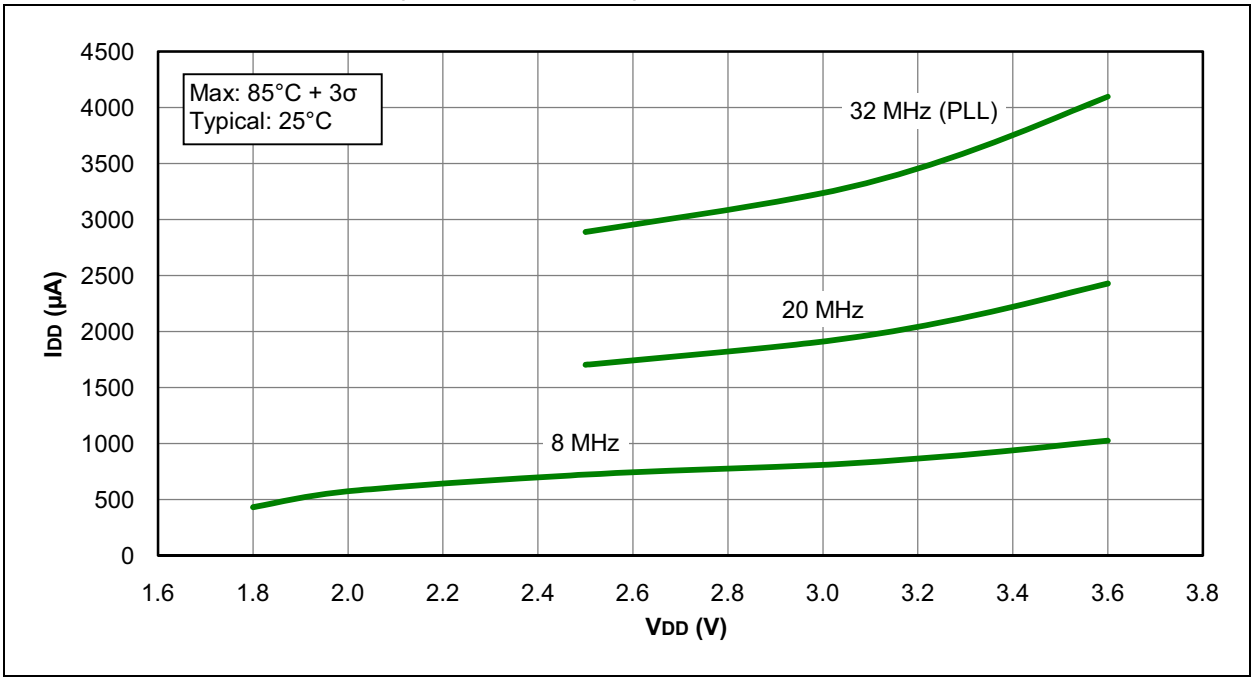



**FIGURE 31-28: IDD MAXIMUM, HS OSCILLATOR, PIC16LF1946/47 ONLY**

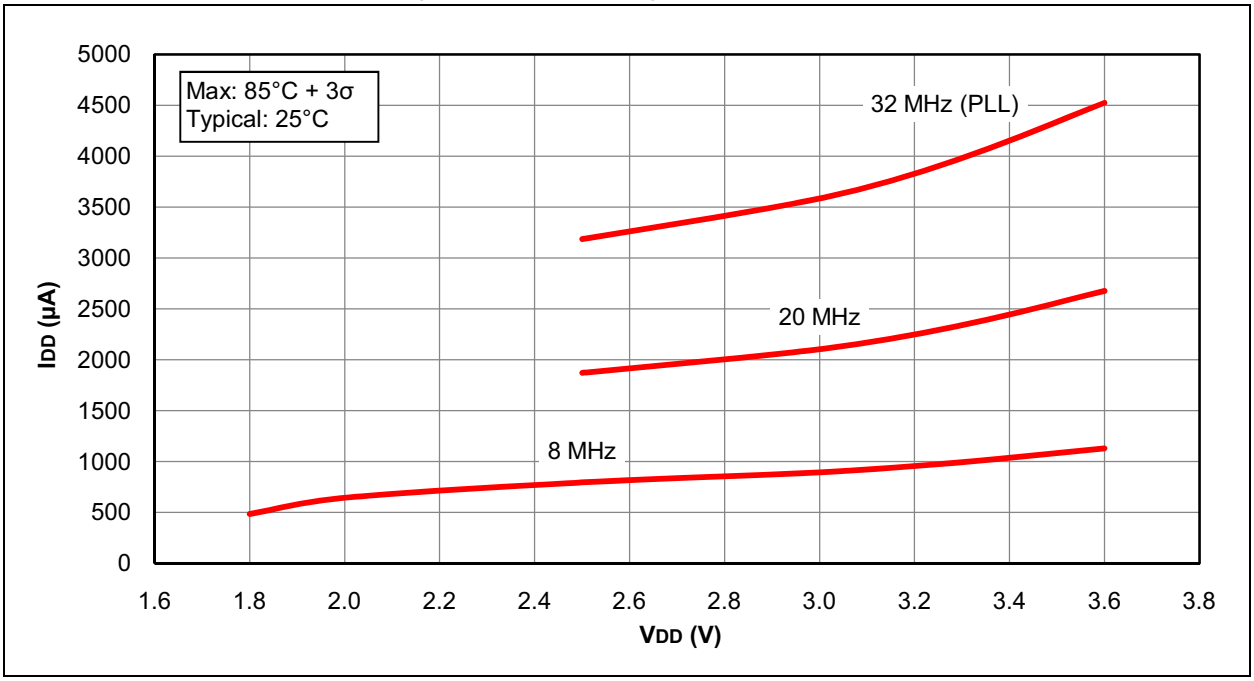


**FIGURE 31-29: IDD TYPICAL, HS OSCILLATOR, PIC16F1946/47 ONLY**

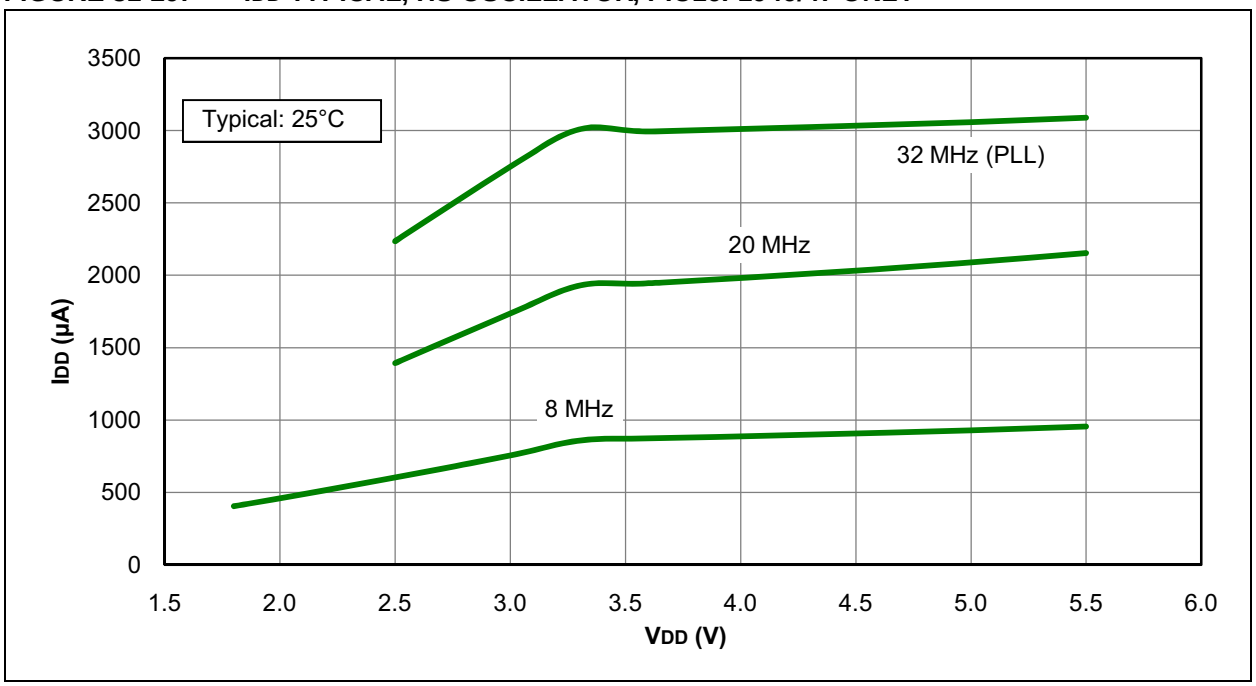
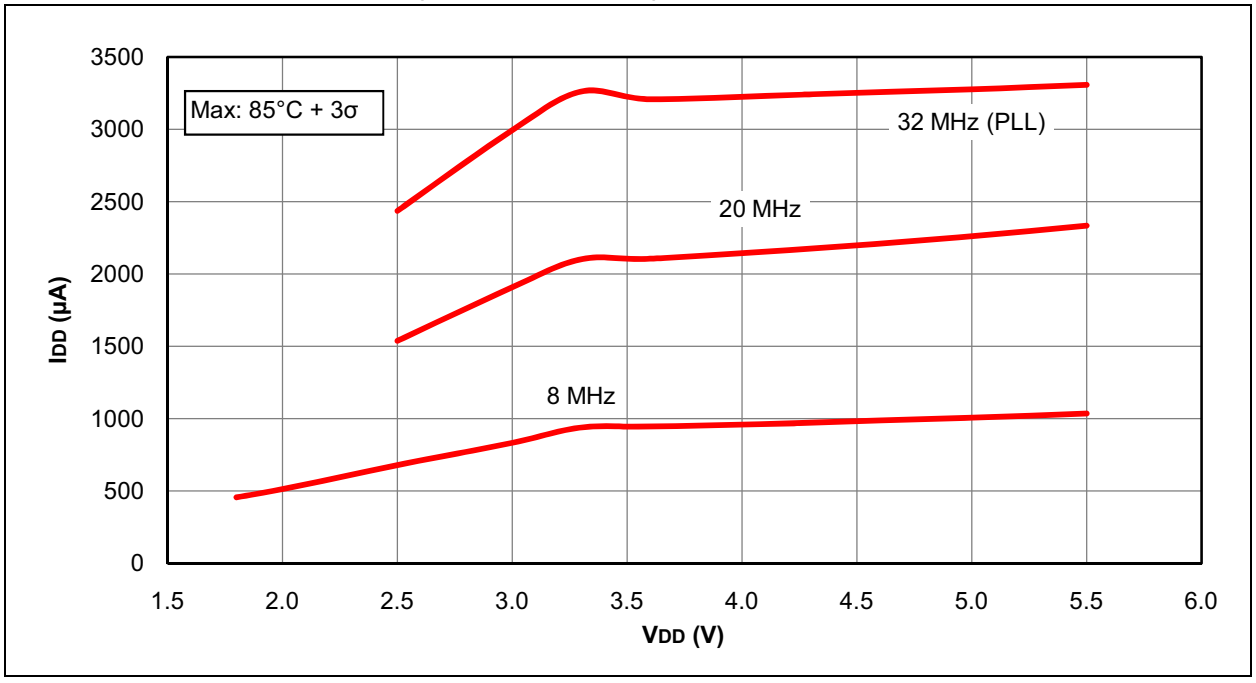



**FIGURE 31-30: IDD MAXIMUM, HS OSCILLATOR, PIC16F1946/47 ONLY**





© 2010-2016 Microchip Technology Inc.

**FIGURE 31-31: IPD BASE, PIC16LF1946/47 ONLY**

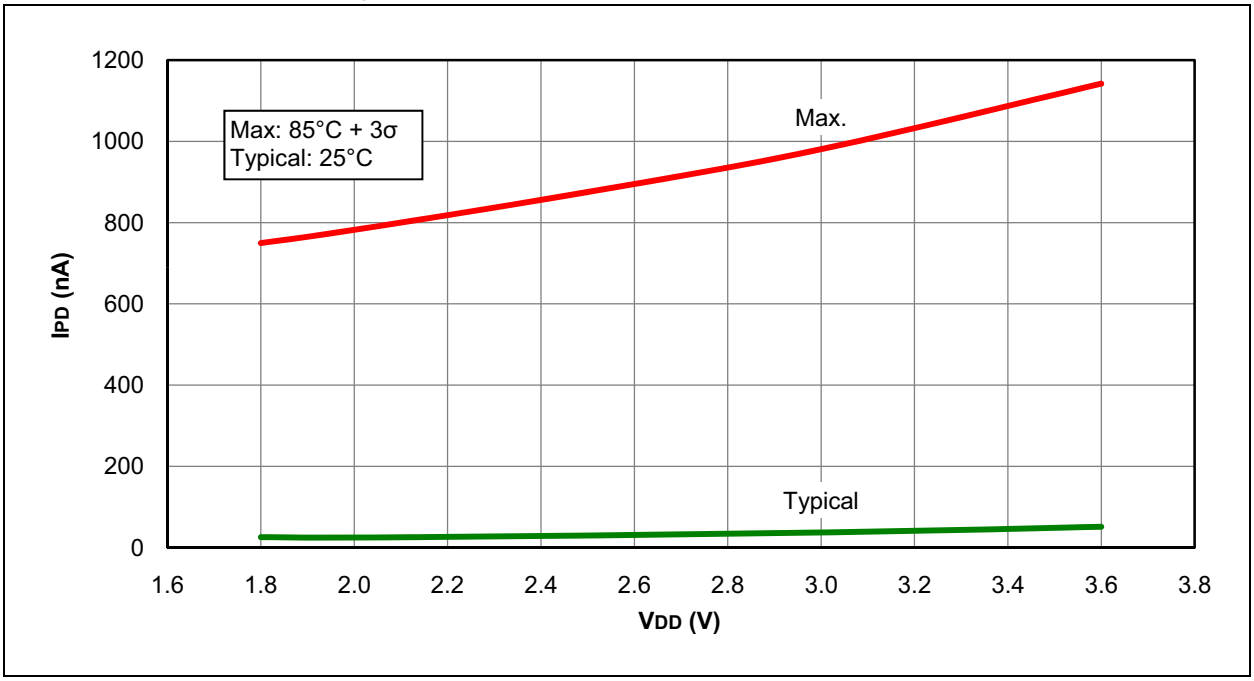



**FIGURE 31-32: IPD BASE, PIC16F1946/47 ONLY**

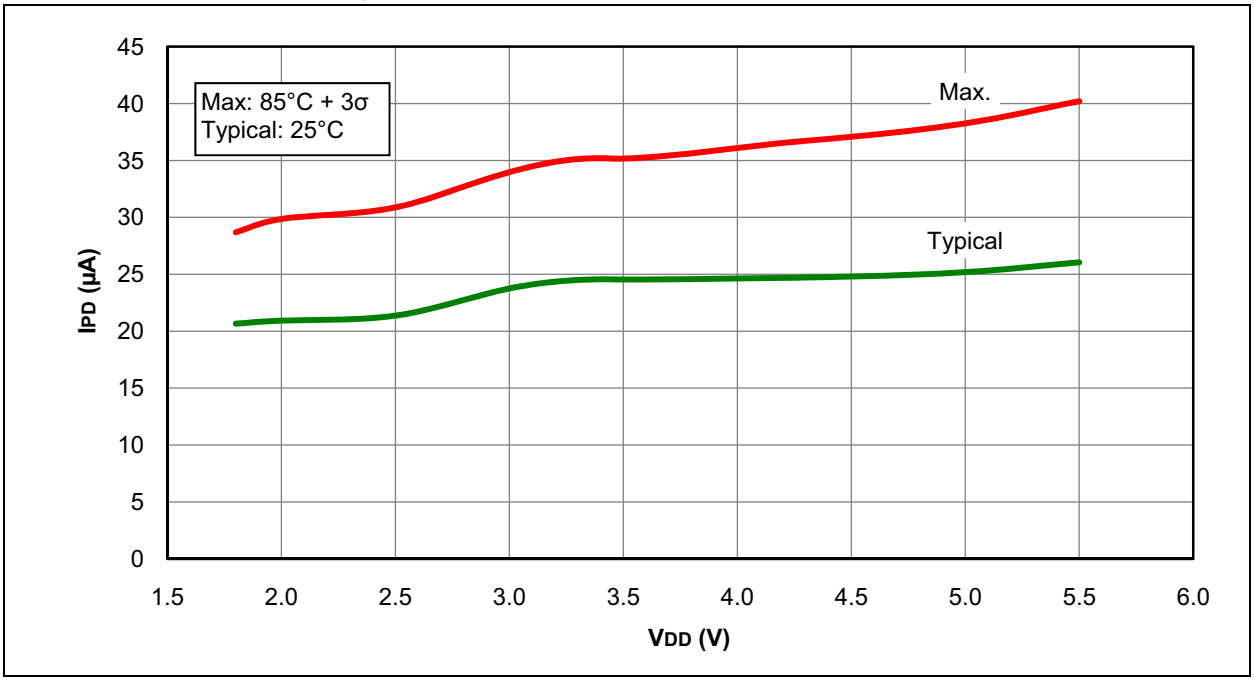


**FIGURE 31-33: IPD, WATCHDOG TIMER (WDT), PIC16LF1946/47 ONLY**

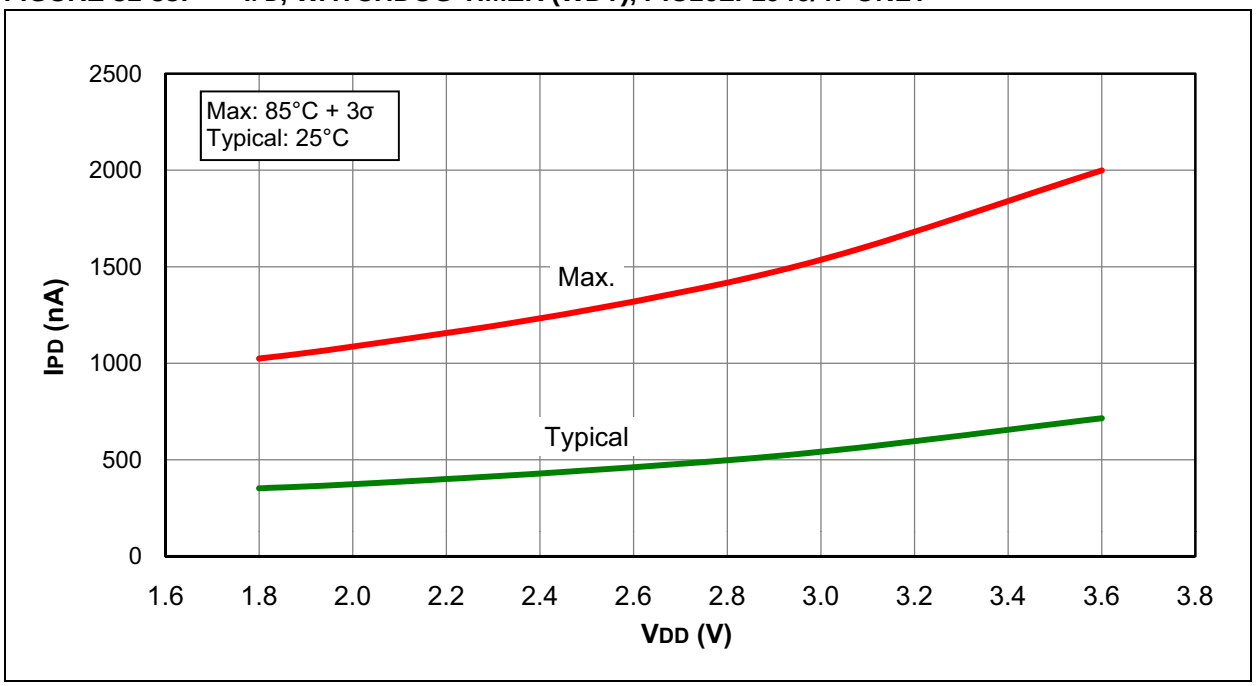
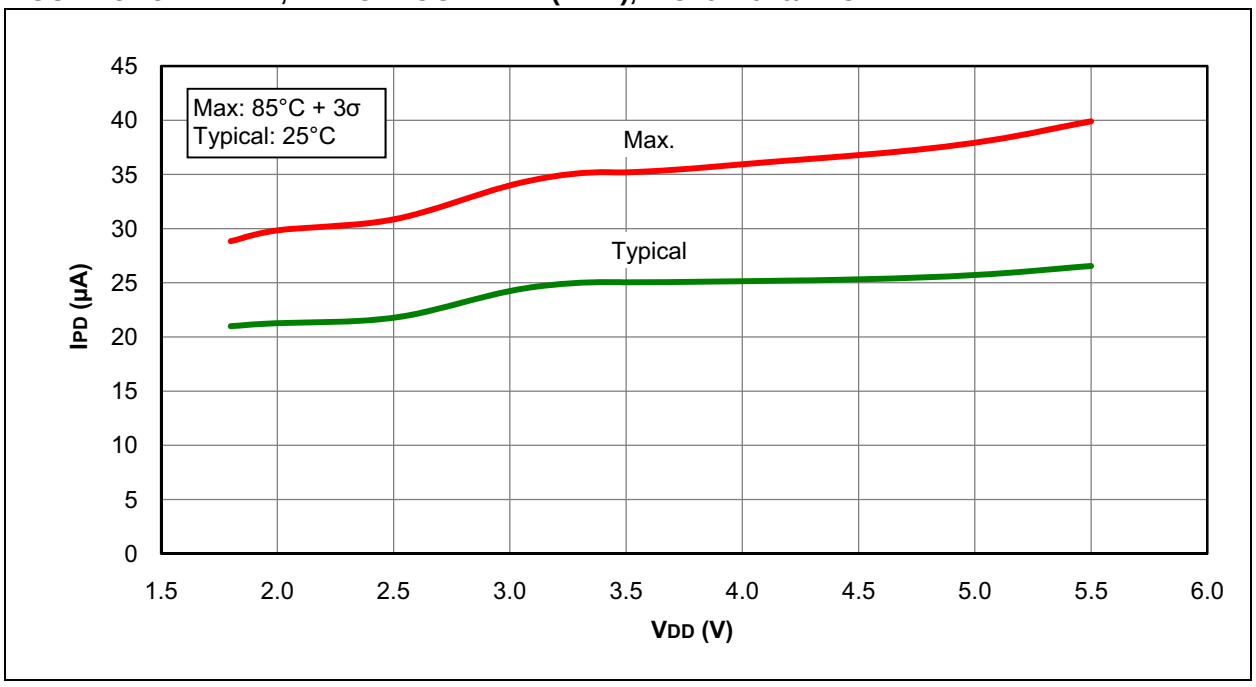



**FIGURE 31-34: IPD, WATCHDOG TIMER (WDT), PIC16F1946/47 ONLY**





© 2010-2016 Microchip Technology Inc.

**FIGURE 31-35: IPD, FIXED VOLTAGE REFERENCE (FVR), PIC16LF1946/47 ONLY**

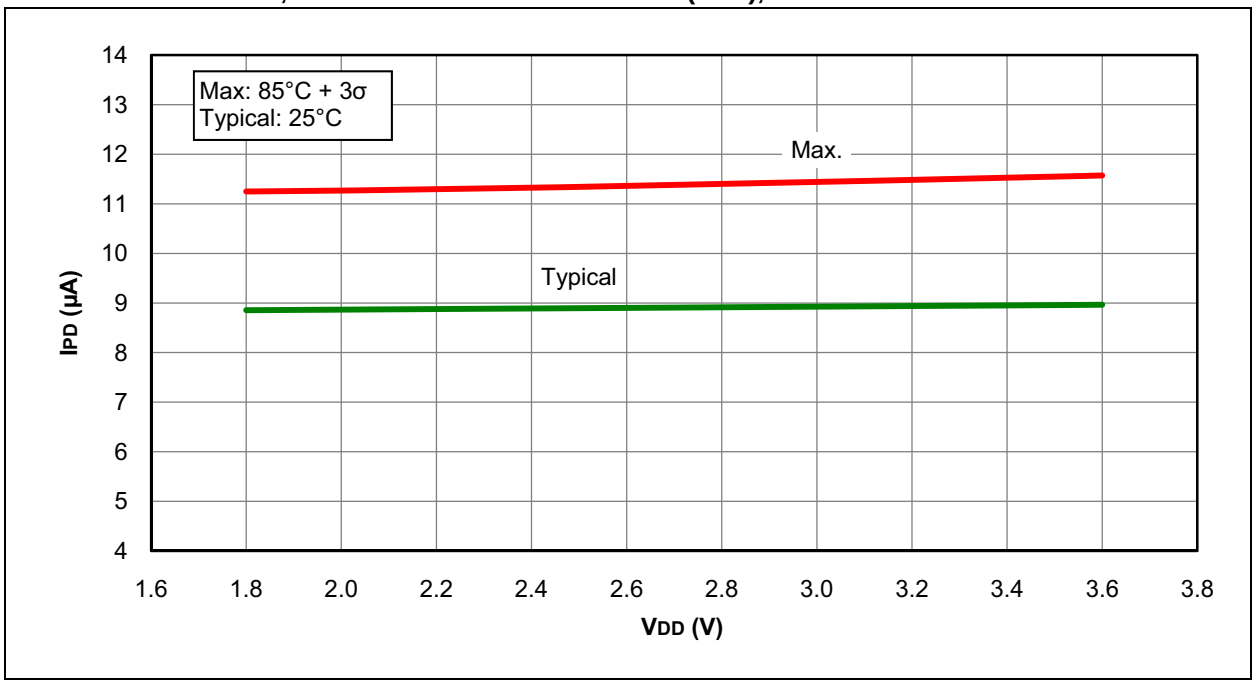



**FIGURE 31-36: IPD, FIXED VOLTAGE REFERENCE (FVR), PIC16F1946/47 ONLY**

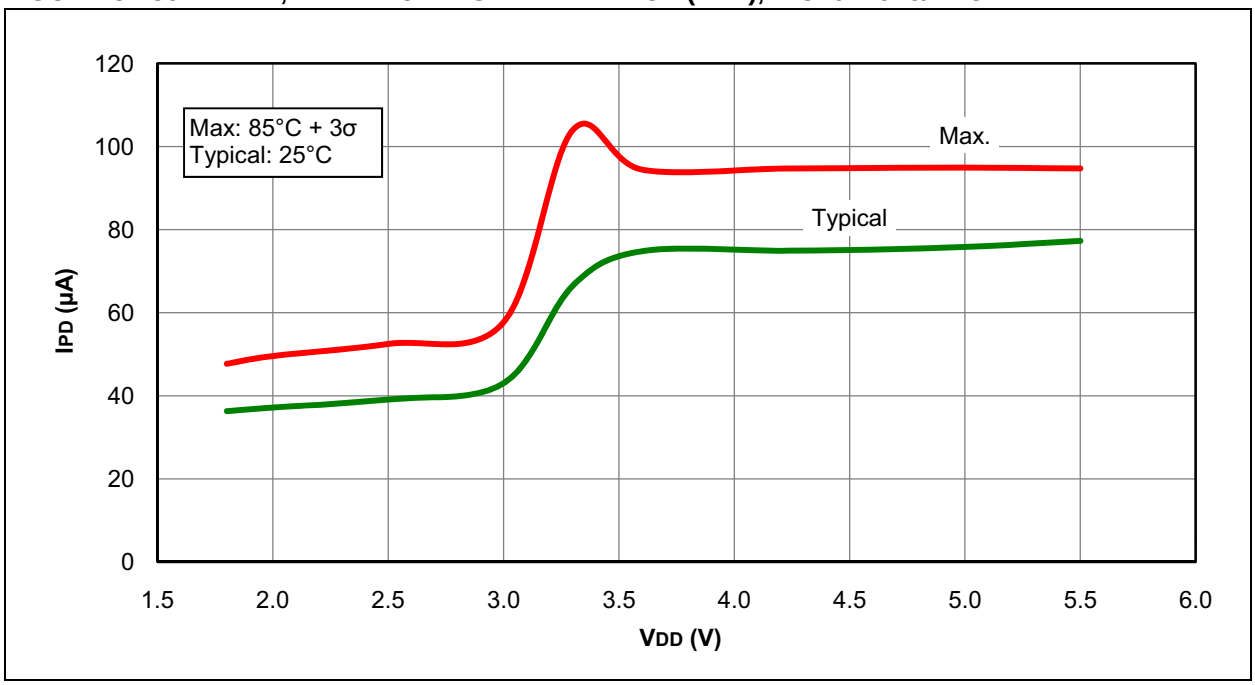


**FIGURE 31-37: IPD, BROWN-OUT RESET (BOR), PIC16LF1946/47 ONLY**

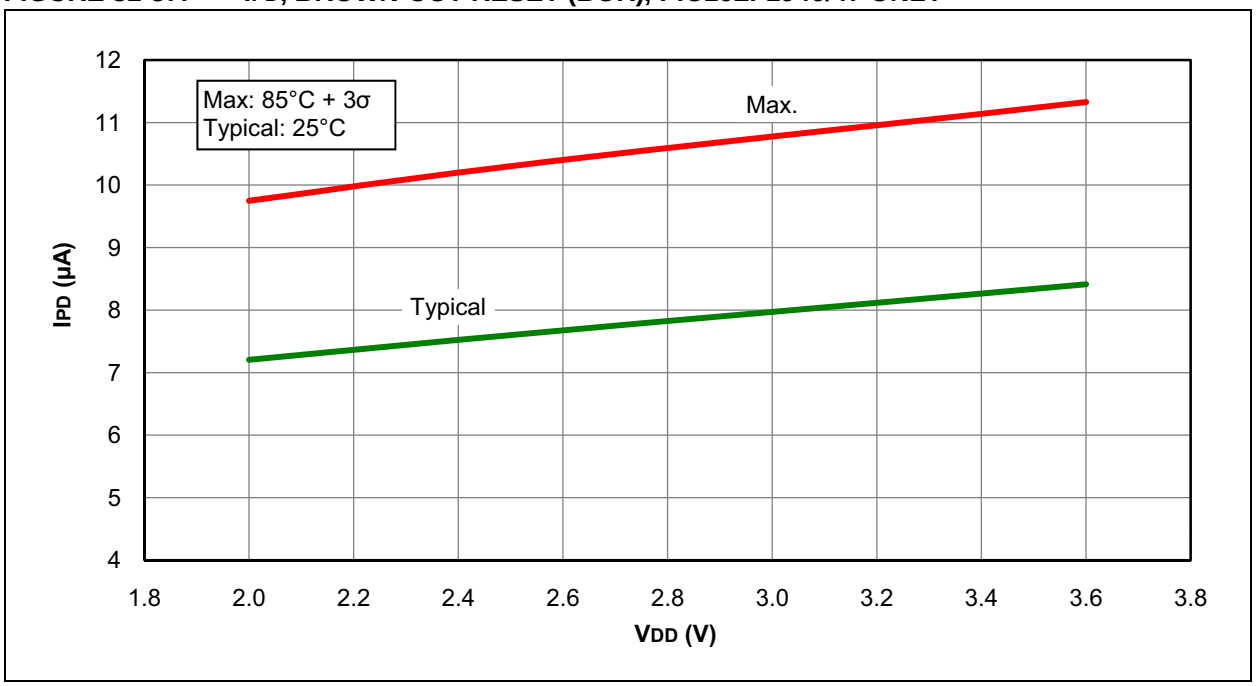
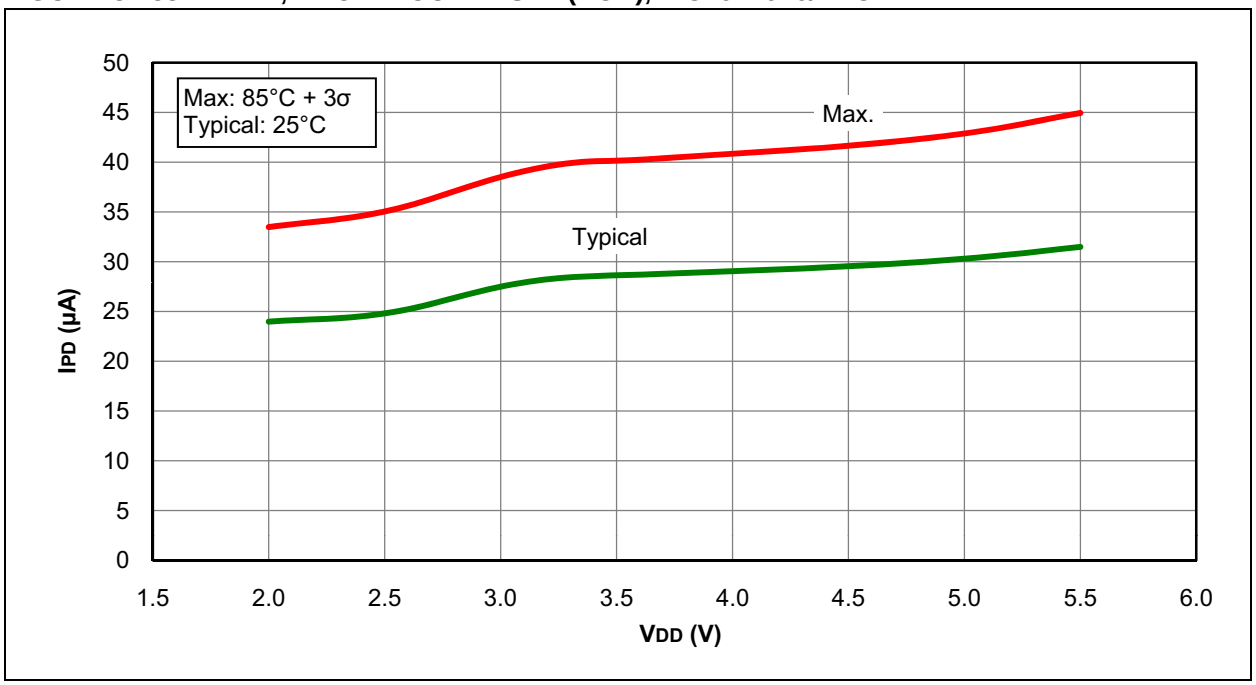



**FIGURE 31-38: IPD, BROWN-OUT RESET (BOR), PIC16F1946/47 ONLY**





© 2010-2016 Microchip Technology Inc.

**FIGURE 31-39: IPD, TIMER1 OSCILLATOR, FOSC = 32 kHz, PIC16LF1946/47 ONLY**

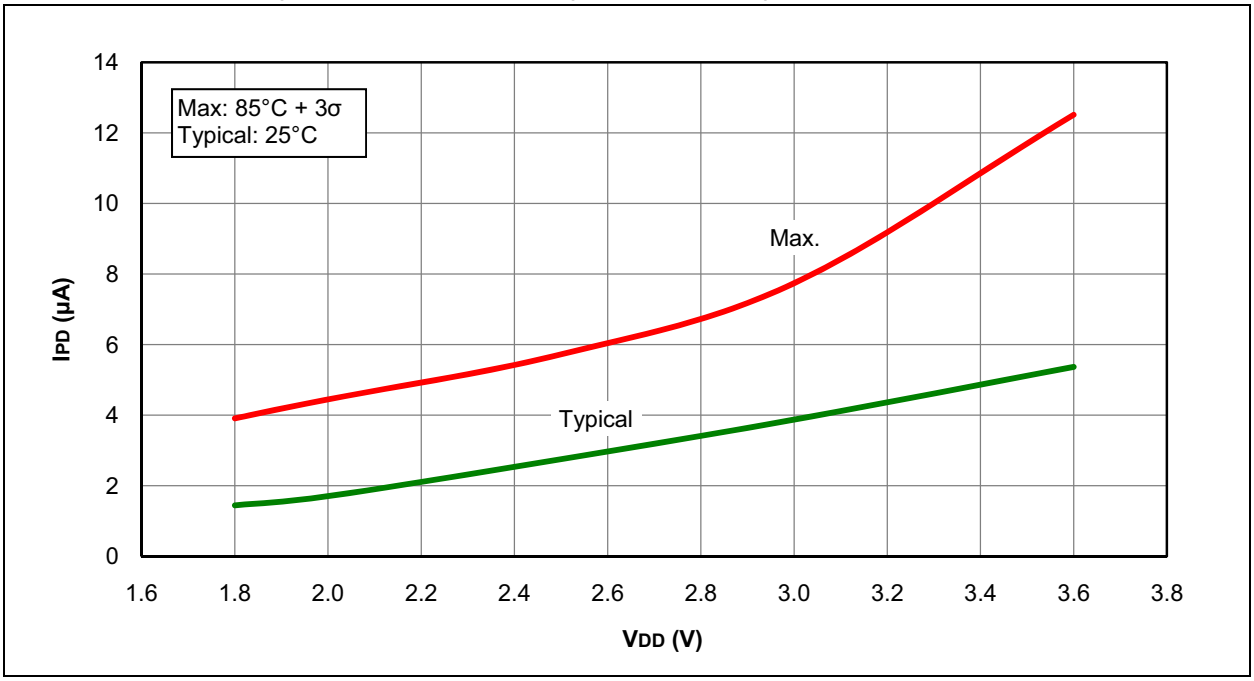



**FIGURE 31-40: IPD, TIMER1 OSCILLATOR, FOSC = 32 kHz, PIC16F1946/47 ONLY**

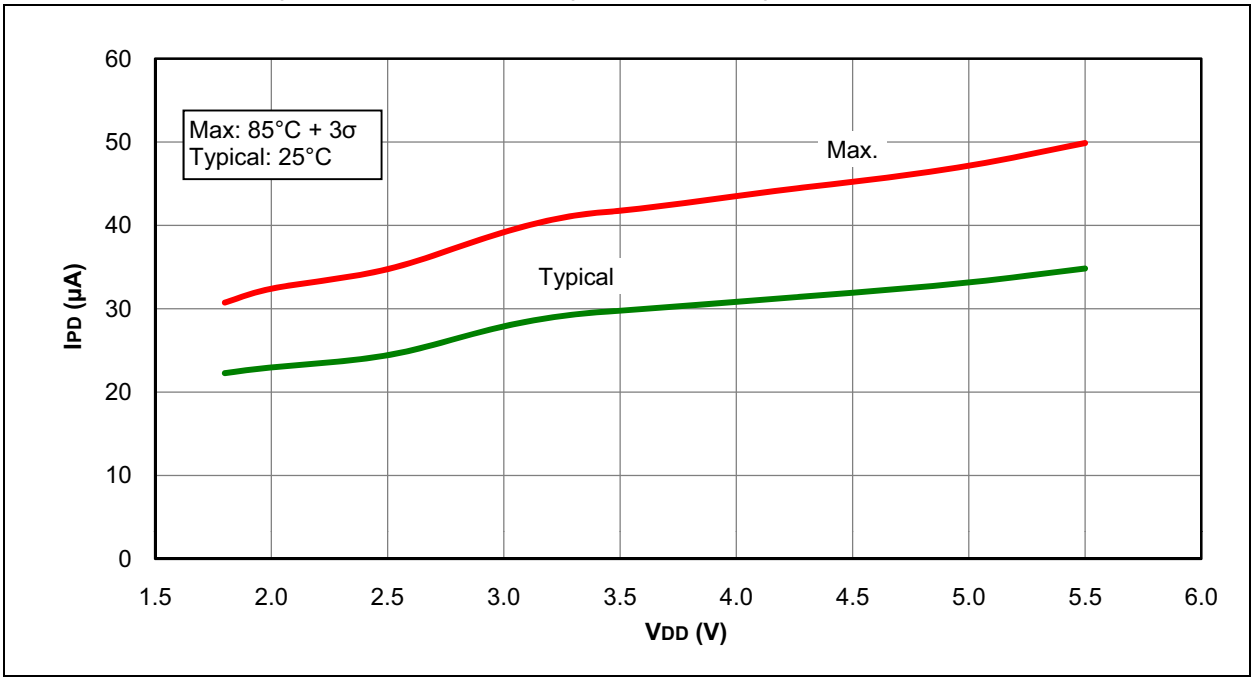


**FIGURE 31-41: IPD, CAPACITIVE SENSING (CPS) MODULE, LOW-CURRENT RANGE, CPSRM = 0, PIC16LF1946/47 ONLY**

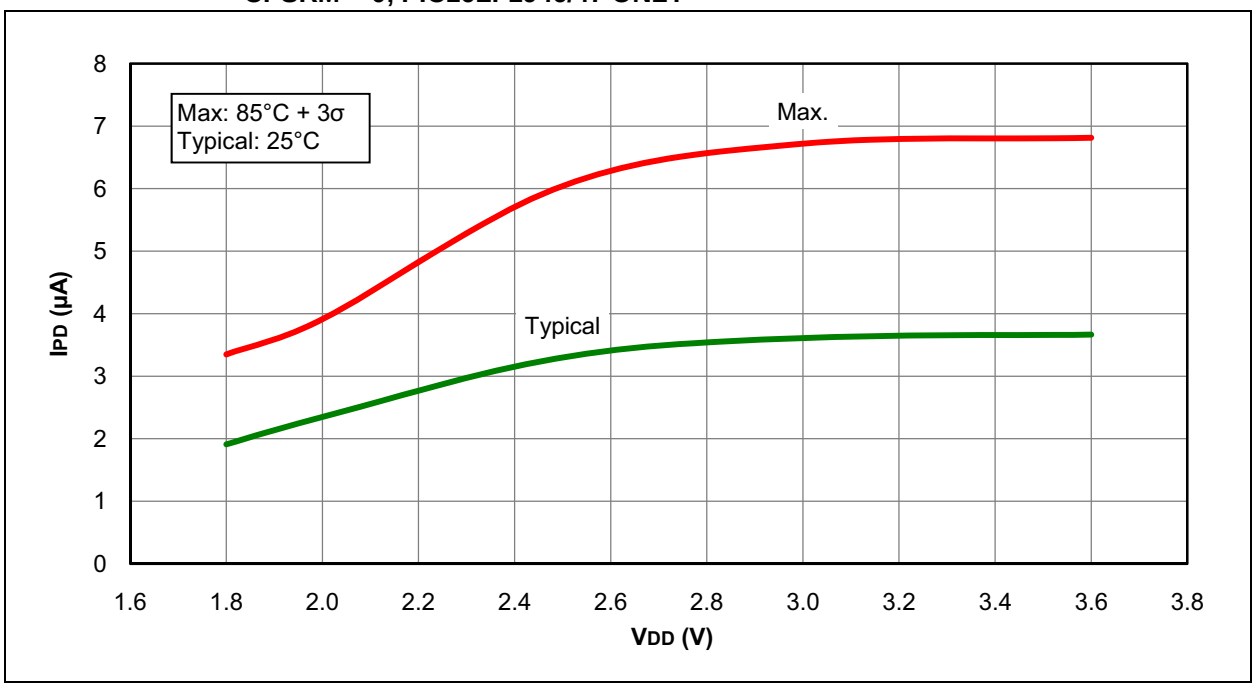
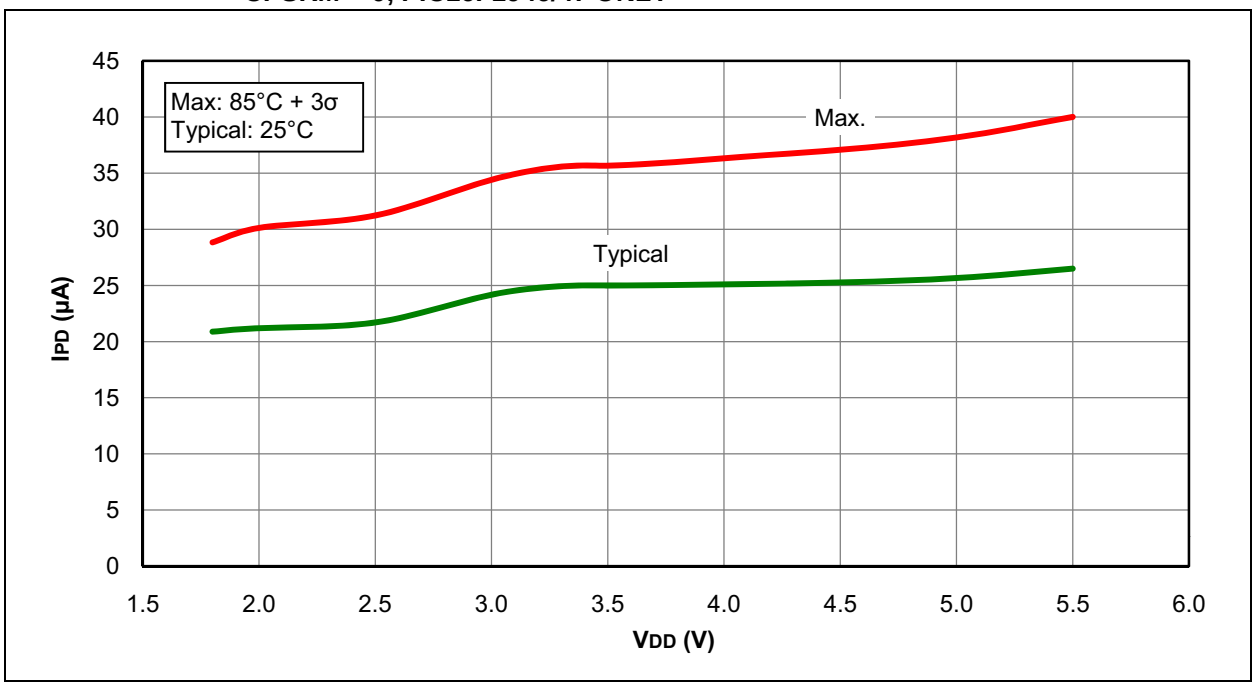



**FIGURE 31-42: IPD, CAPACITIVE SENSING (CPS) MODULE, LOW-CURRENT RANGE, CPSRM = 0, PIC16F1946/47 ONLY**





© 2010-2016 Microchip Technology Inc.

**FIGURE 31-43: IPD, CAPACITIVE SENSING (CPS) MODULE, MEDIUM-CURRENT RANGE, CPSRM = 0, PIC16LF1946/47 ONLY**

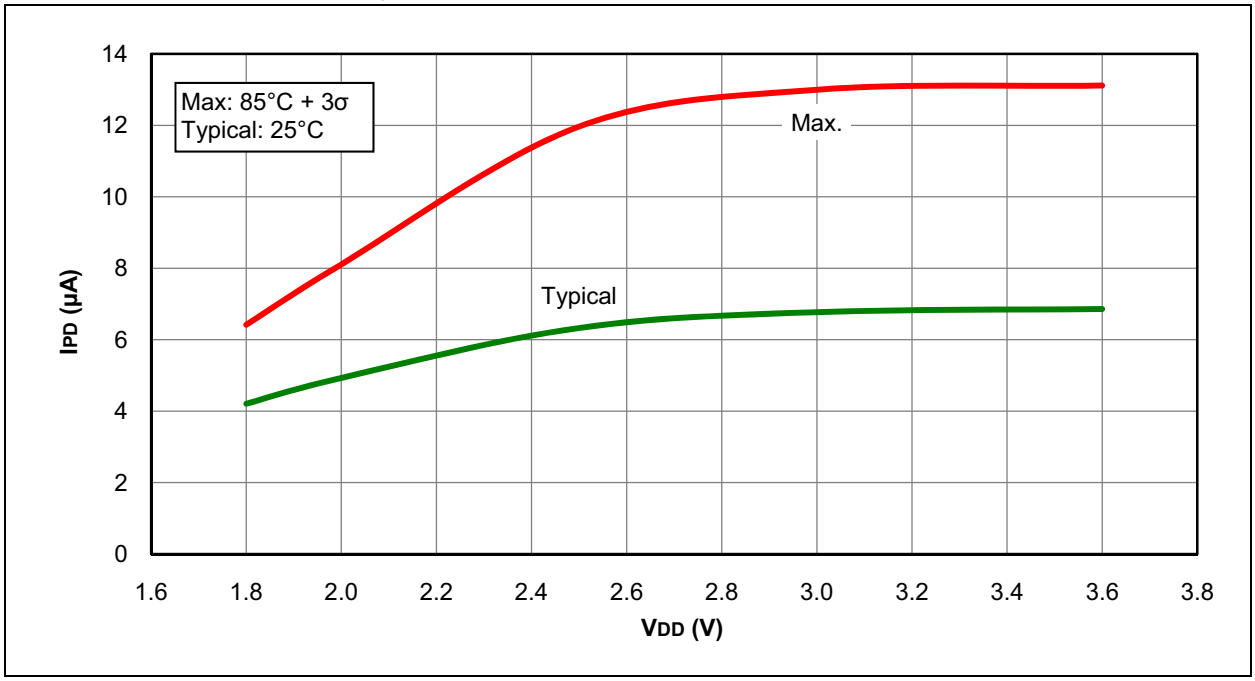



**FIGURE 31-44: IPD, CAPACITIVE SENSING (CPS) MODULE, MEDIUM-CURRENT RANGE, CPSRM = 0, PIC16F1946/47 ONLY**

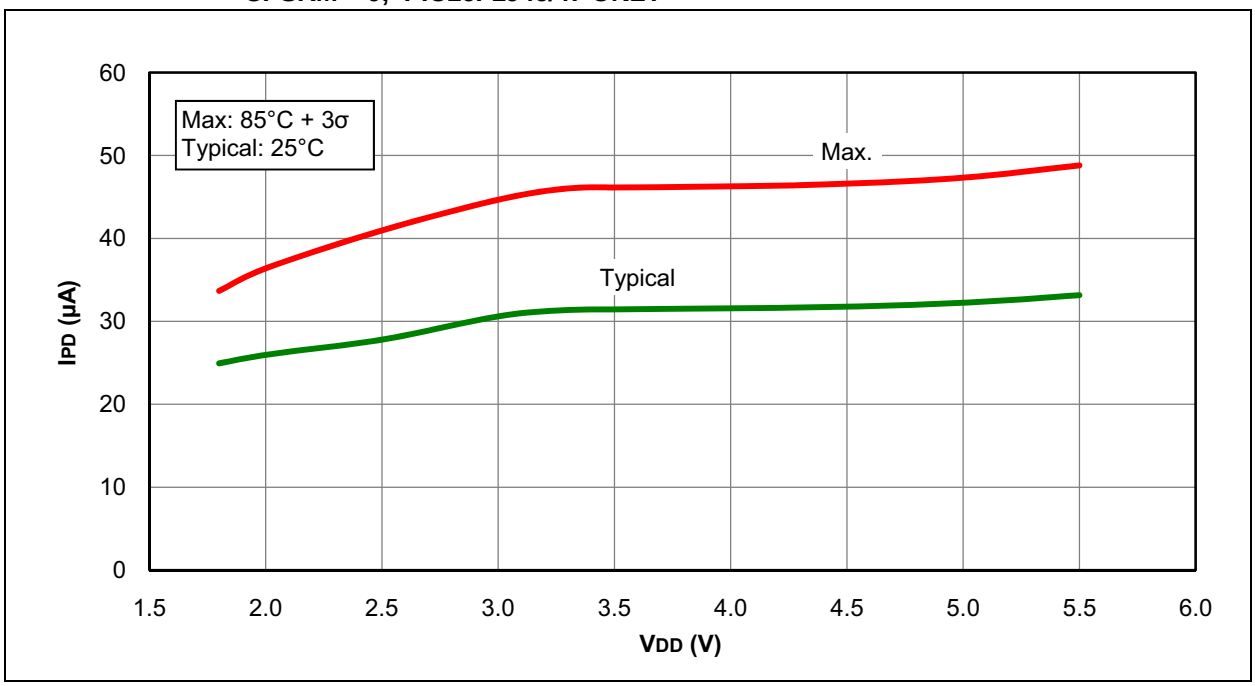


**FIGURE 31-45: IPD, CAPACITIVE SENSING (CPS) MODULE, HIGH-CURRENT RANGE, CPSRM = 0, PIC16LF1946/47 ONLY**

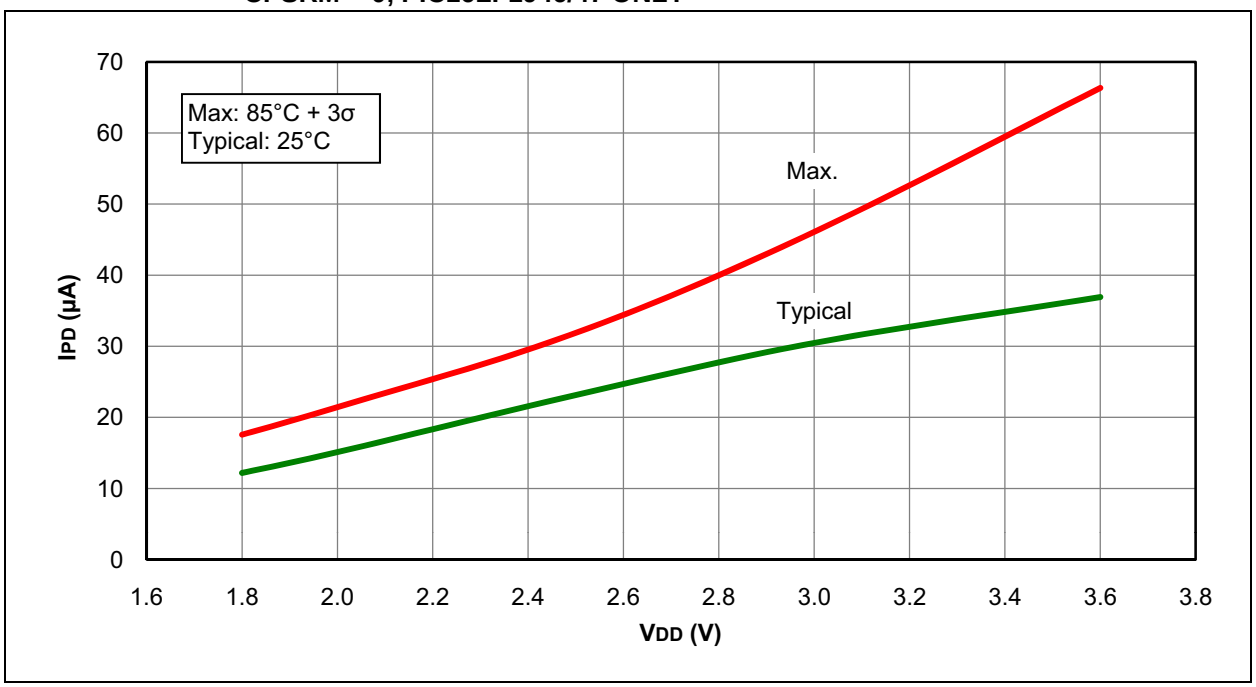
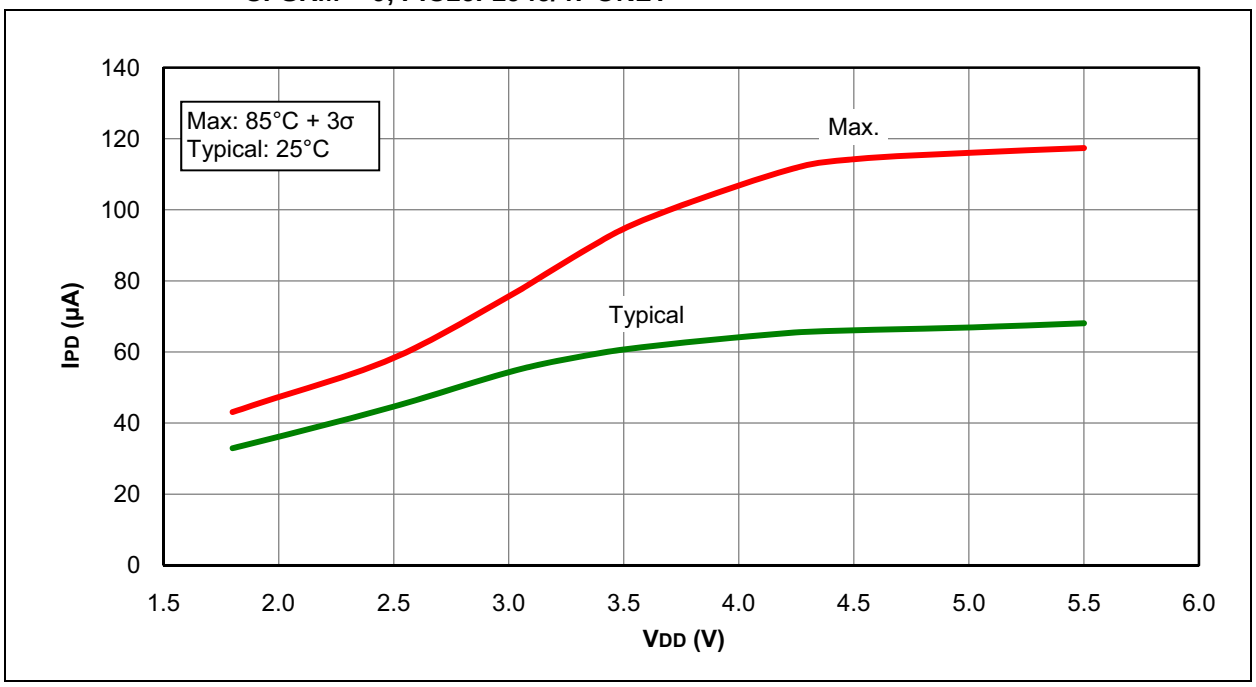



**FIGURE 31-46: IPD, CAPACITIVE SENSING (CPS) MODULE, HIGH-CURRENT RANGE, CPSRM = 0, PIC16F1946/47 ONLY**





© 2010-2016 Microchip Technology Inc.

**FIGURE 31-47: IPD, COMPARATOR, LOW-POWER MODE, PIC16LF1946/47 ONLY**

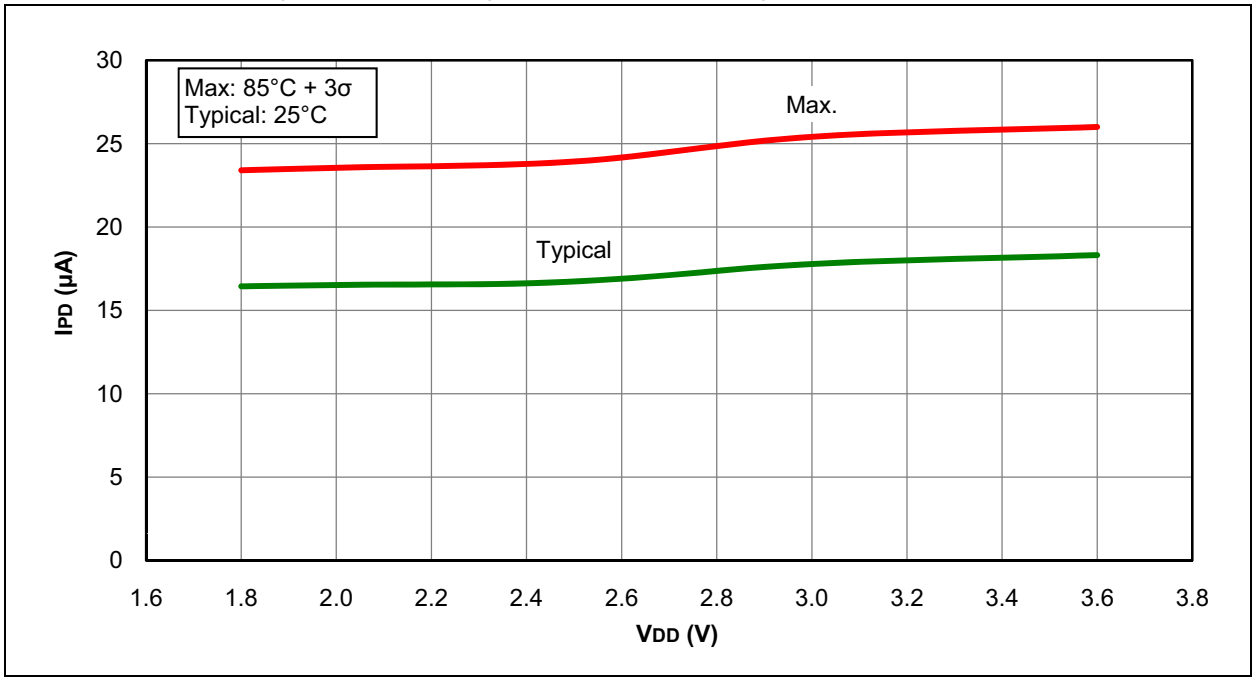



**FIGURE 31-48: IPD, COMPARATOR, LOW-POWER MODE, PIC16F1946/47 ONLY**

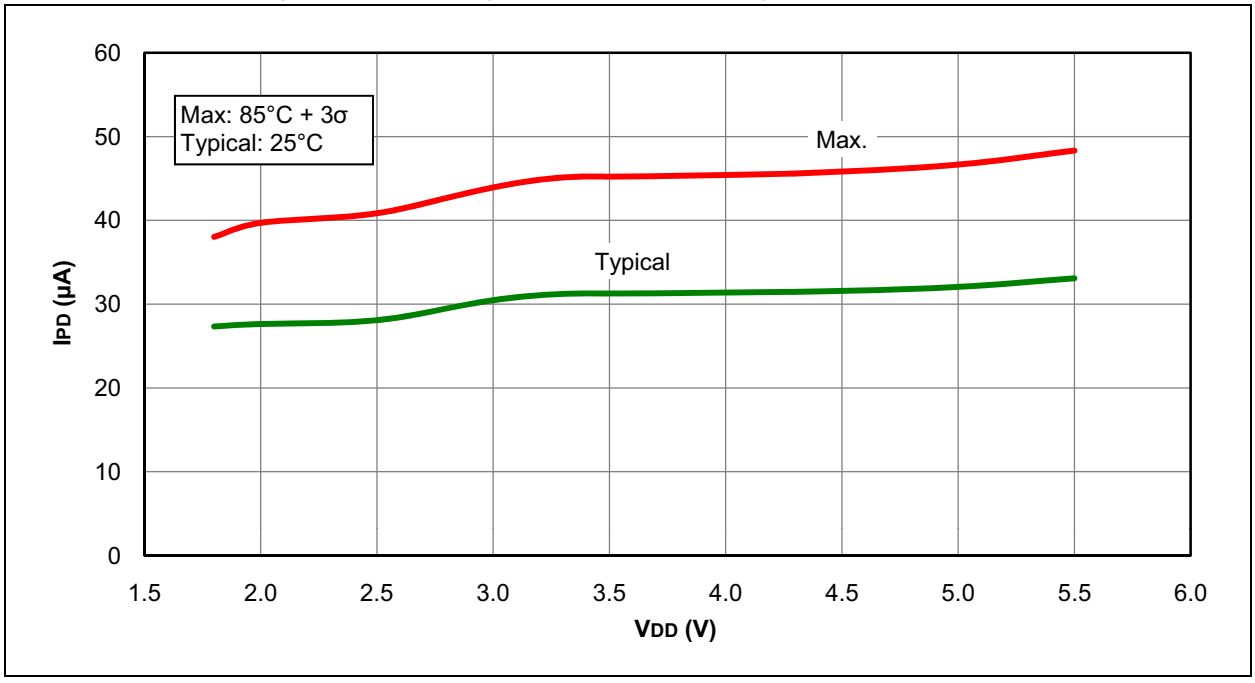


**FIGURE 31-49: IPD, COMPARATOR, HIGH-POWER MODE, PIC16LF1946/47 ONLY**

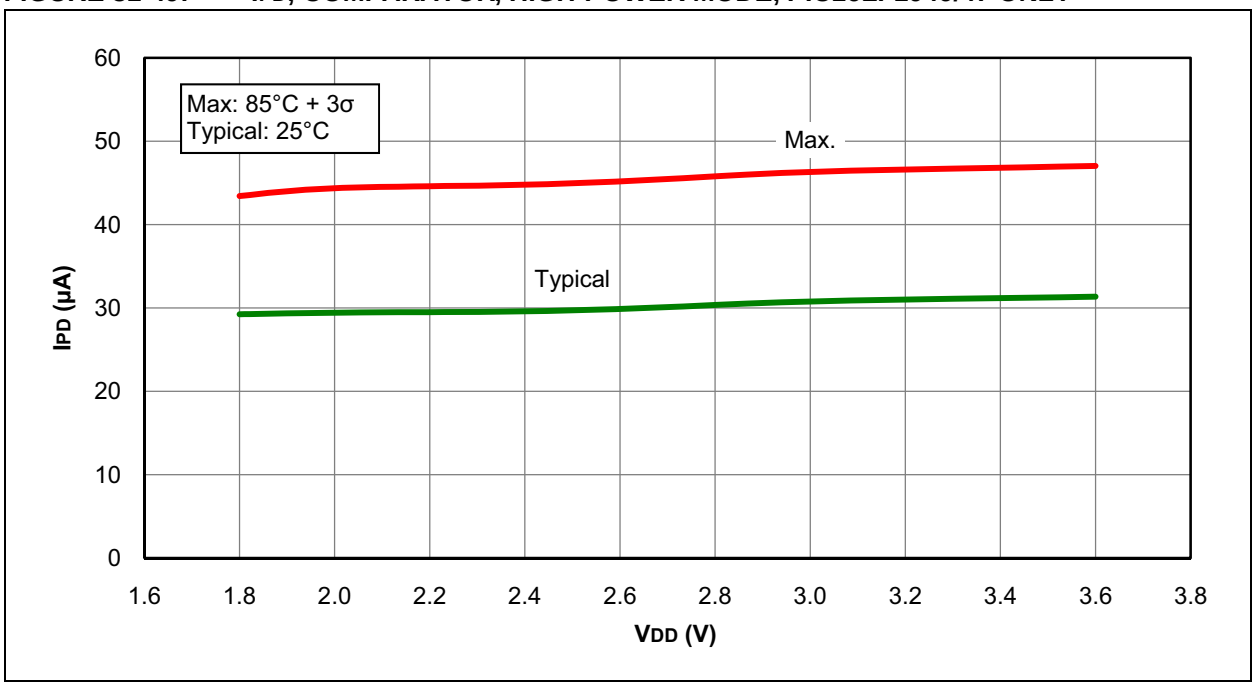
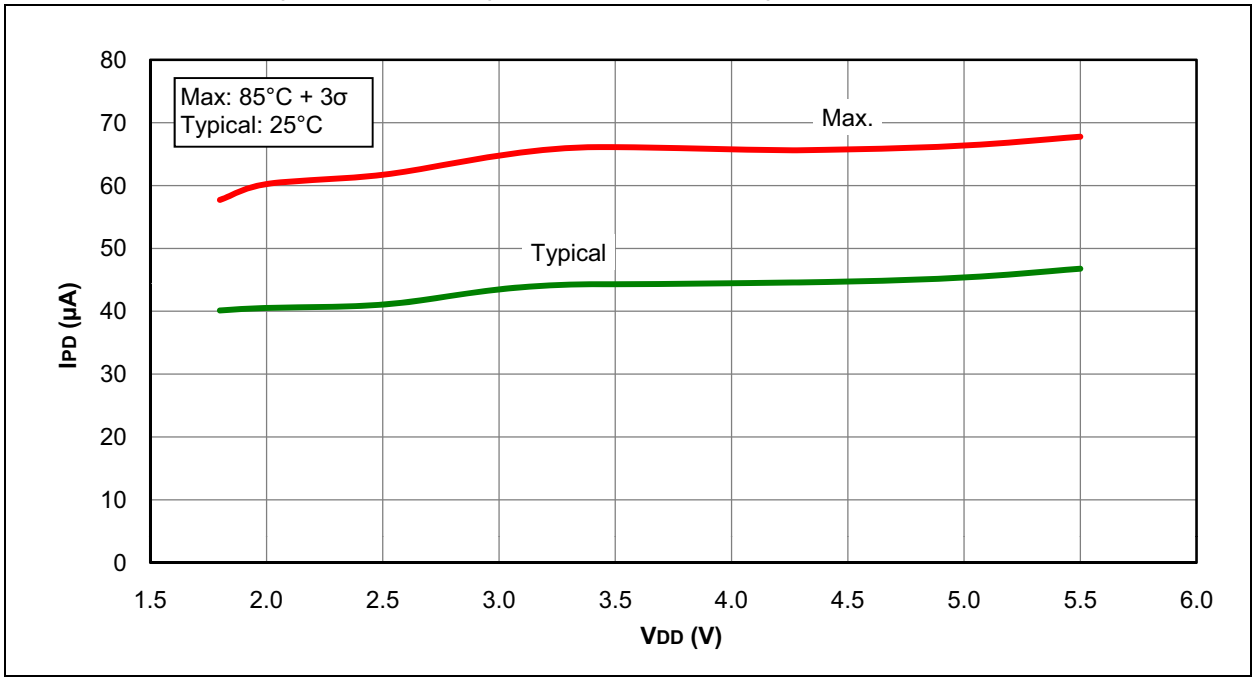



**FIGURE 31-50: IPD, COMPARATOR, HIGH-POWER MODE, PIC16F1946/47 ONLY**





© 2010-2016 Microchip Technology Inc.

**FIGURE 31-51: VOH VS. IOH OVER TEMPERATURE, VDD = 5.0V, PIC16F1946/47 ONLY**

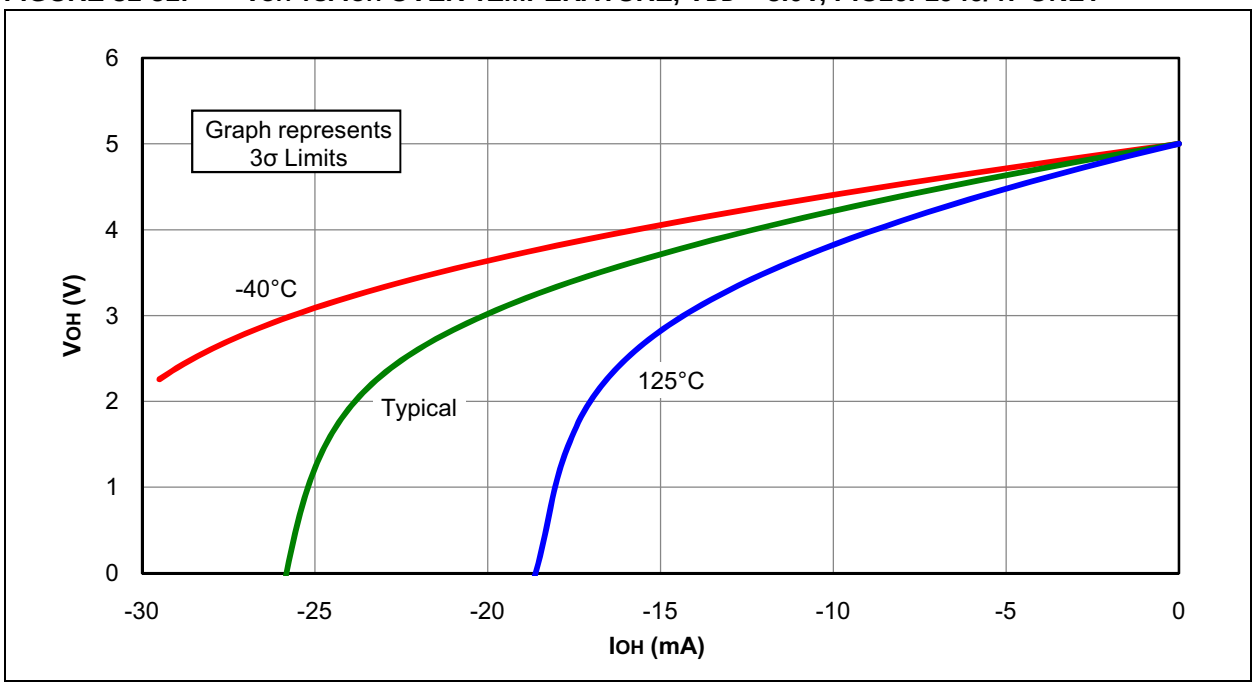



**FIGURE 31-52: VOL VS. IOL OVER TEMPERATURE, VDD = 5.0V, PIC16F1946/47 ONLY**

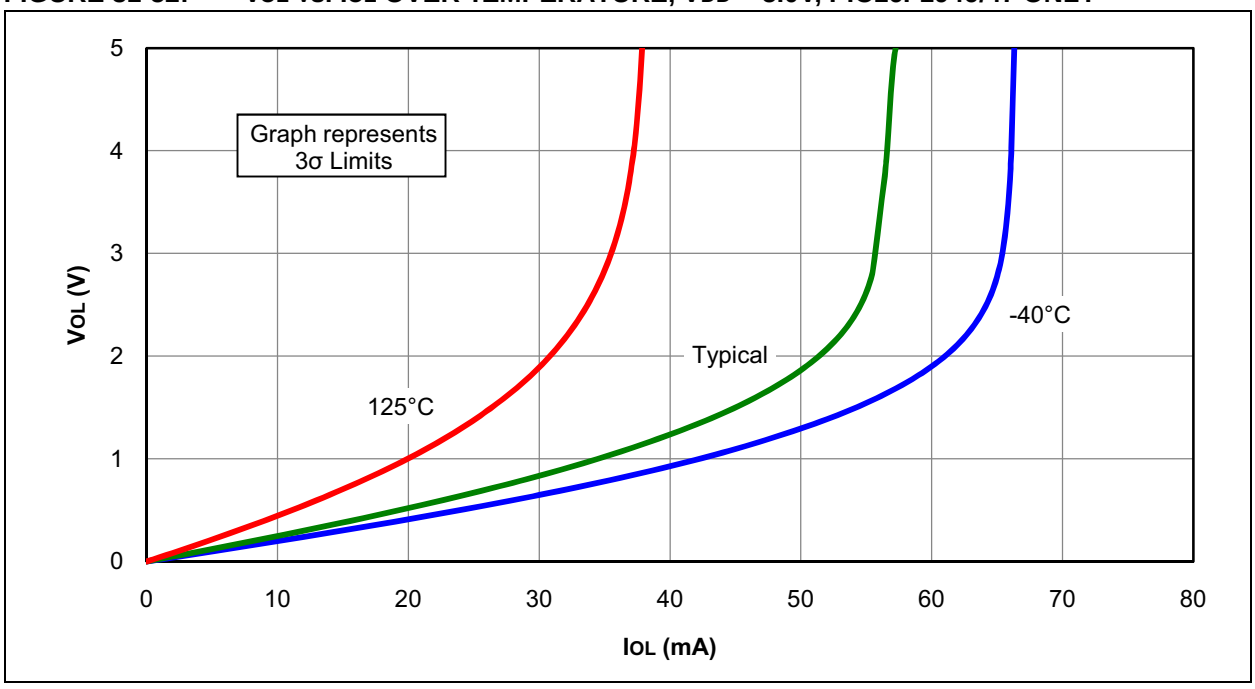


**FIGURE 31-53: VOH VS. IOH OVER TEMPERATURE, VDD = 3.0V**

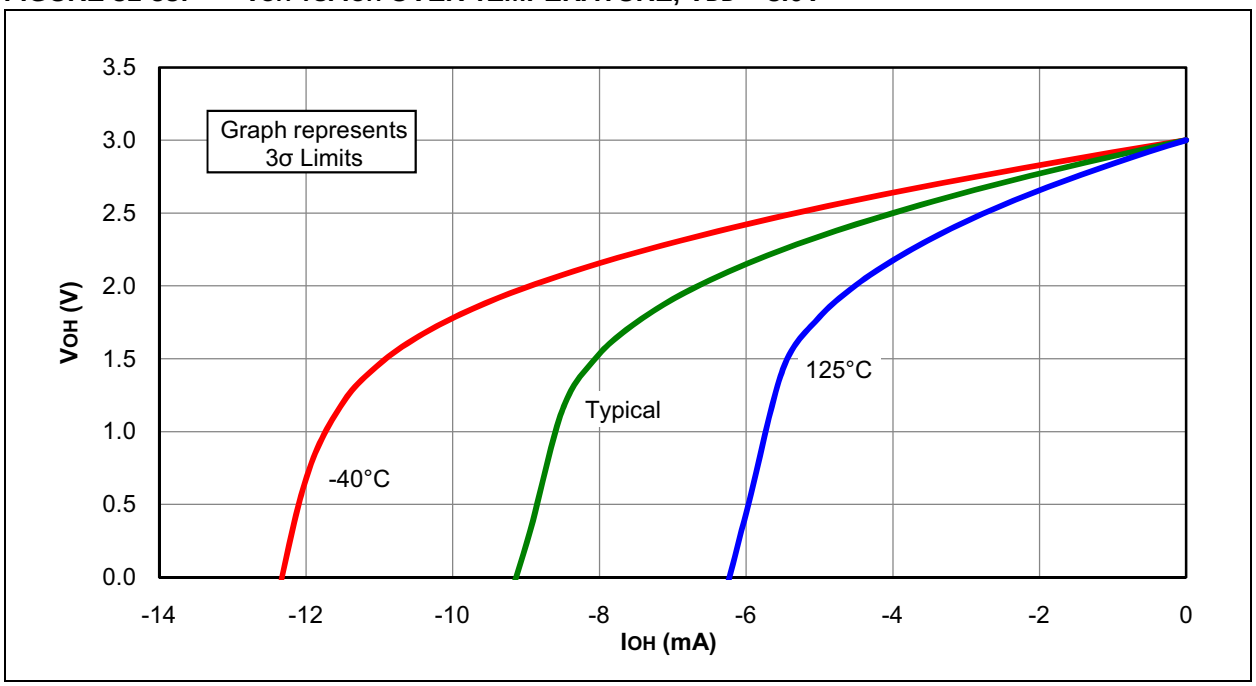
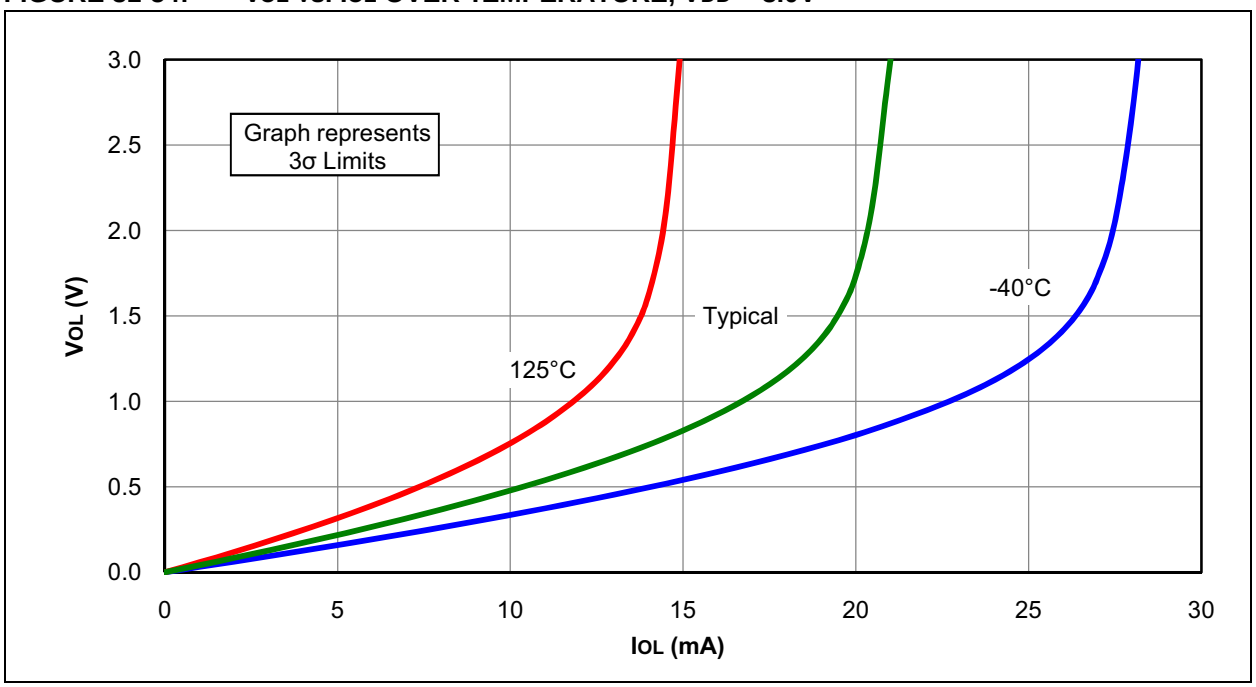



**FIGURE 31-54: VOL VS. IOL OVER TEMPERATURE, VDD = 3.0V**





© 2010-2016 Microchip Technology Inc.

**FIGURE 31-55: VOH VS. IOH OVER TEMPERATURE, VDD = 1.8V**

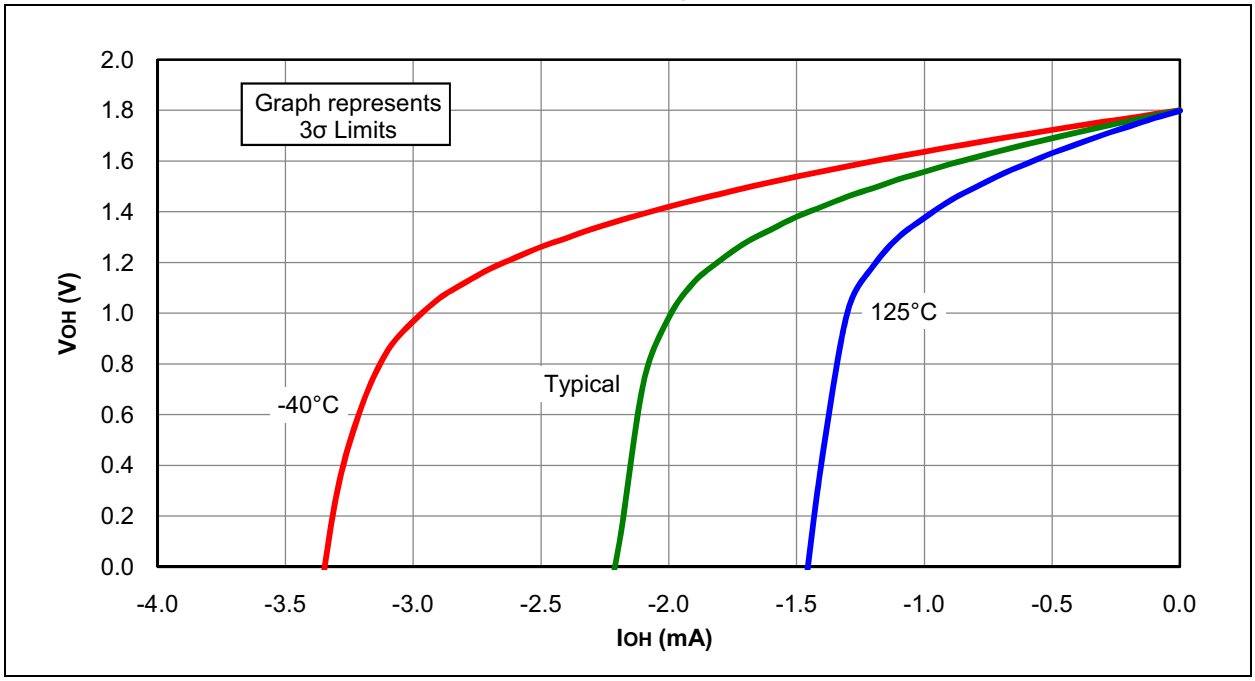



**FIGURE 31-56: VOL VS. IOL OVER TEMPERATURE, VDD = 1.8V**

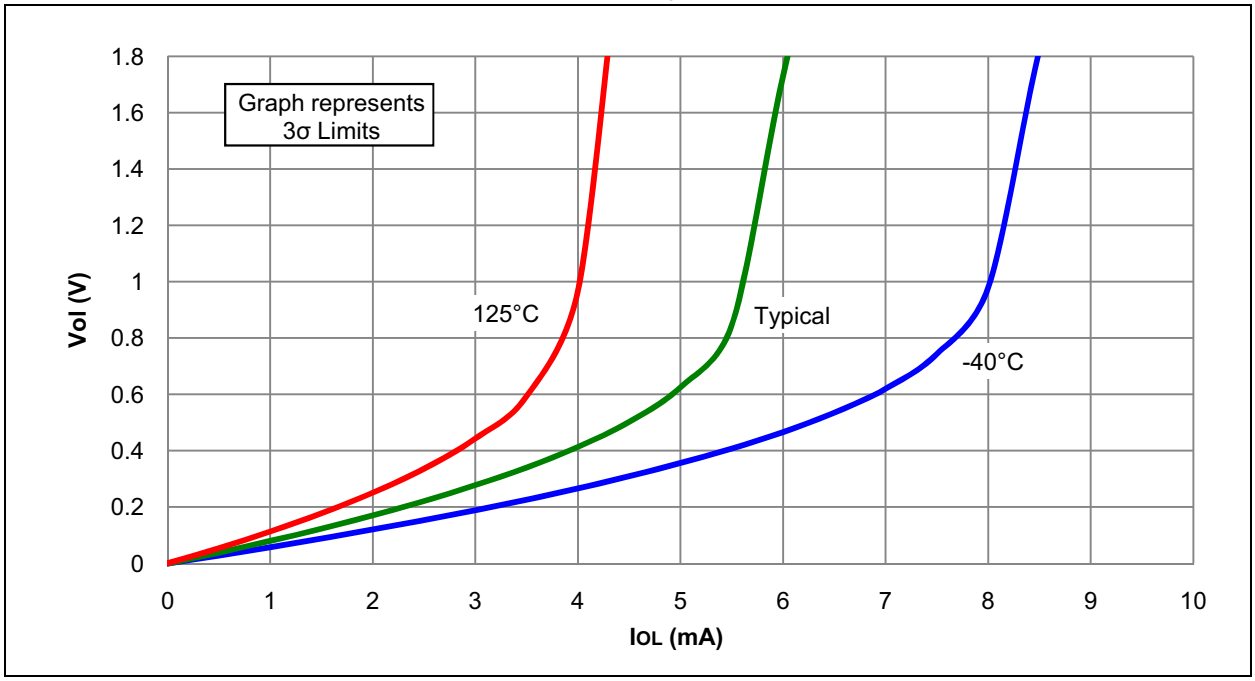


**FIGURE 31-57: BROWN-OUT RESET VOLTAGE, BORV = 1**

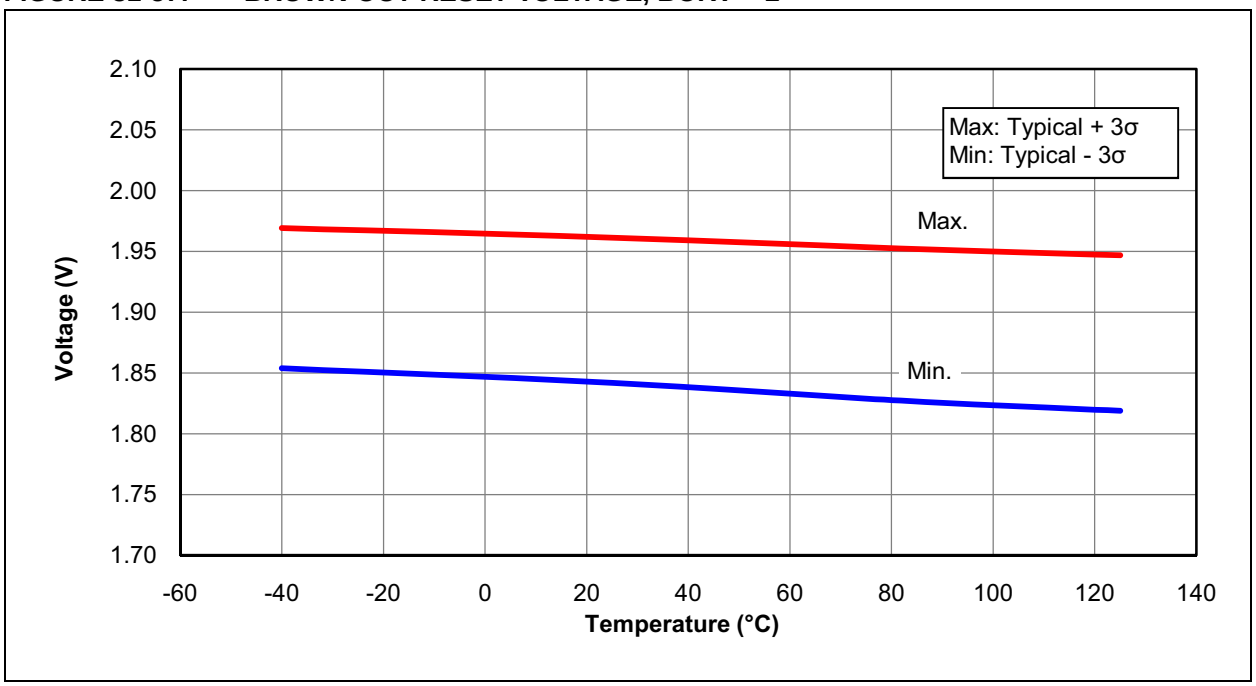
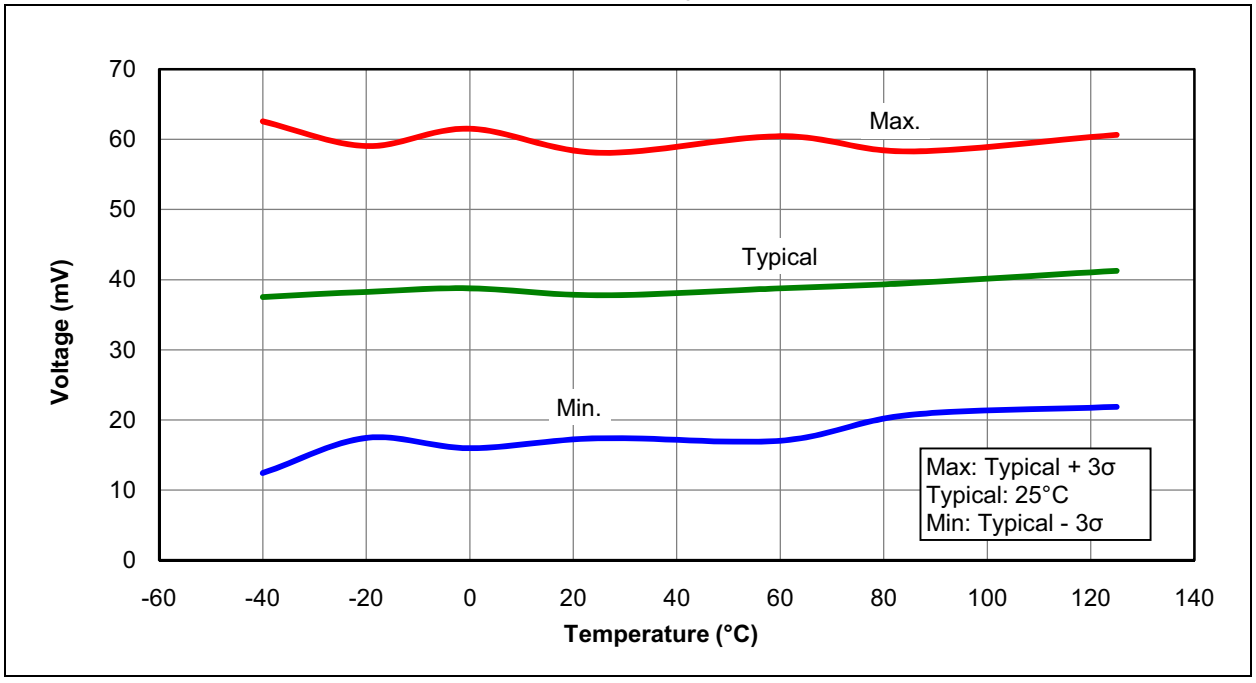



**FIGURE 31-58: BROWN-OUT RESET HYSTERESIS, BORV = 1**





© 2010-2016 Microchip Technology Inc.

**FIGURE 31-59: BROWN-OUT RESET VOLTAGE, BORV = 0**

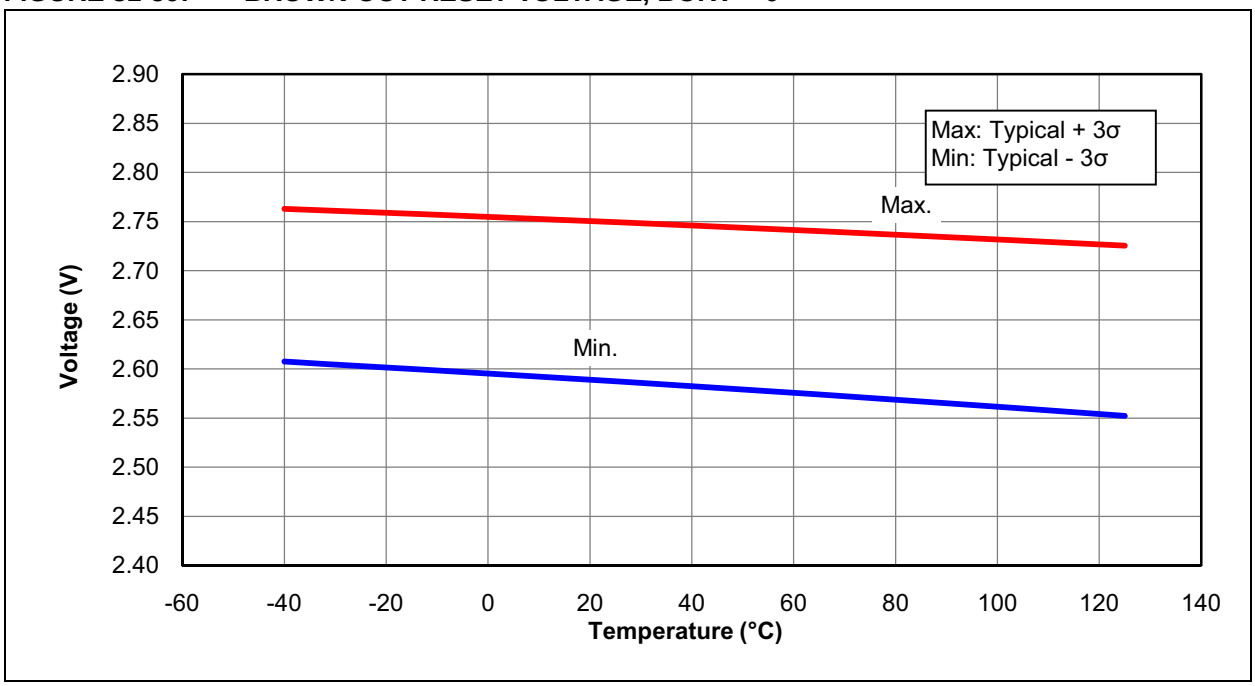



**FIGURE 31-60: BROWN-OUT RESET HYSTERESIS, BORV = 0**

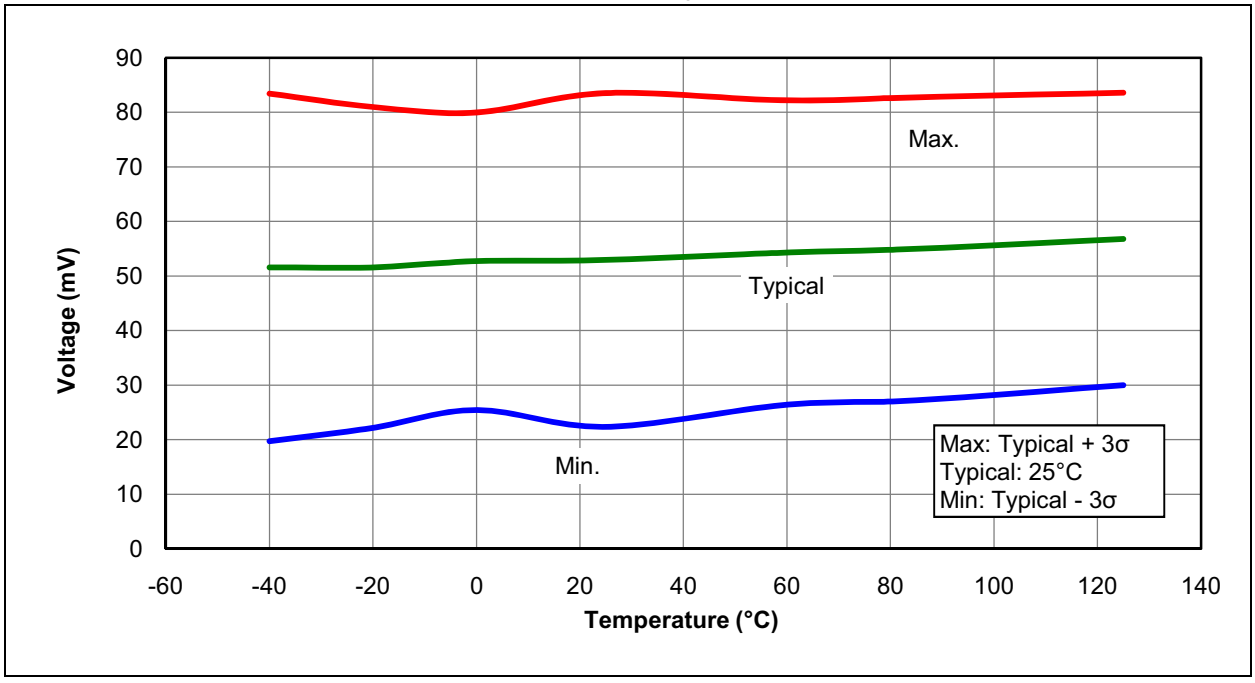


**FIGURE 31-61: COMPARATOR HYSTERESIS, HIGH-POWER MODE**

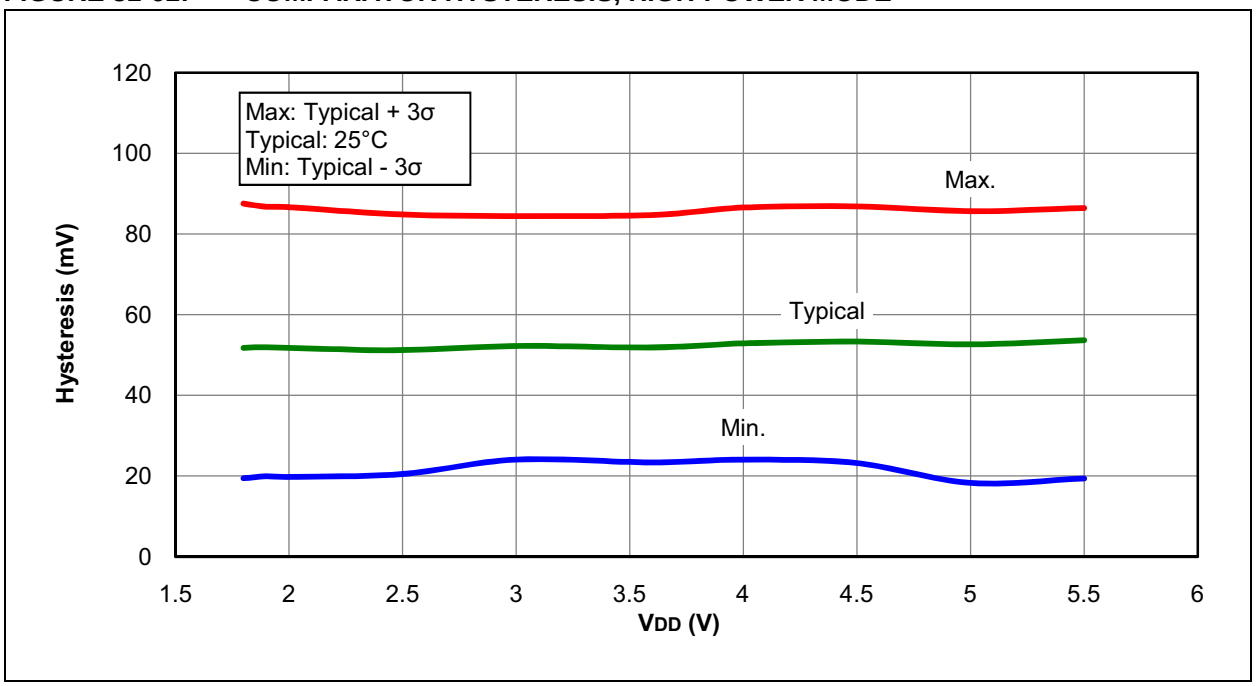
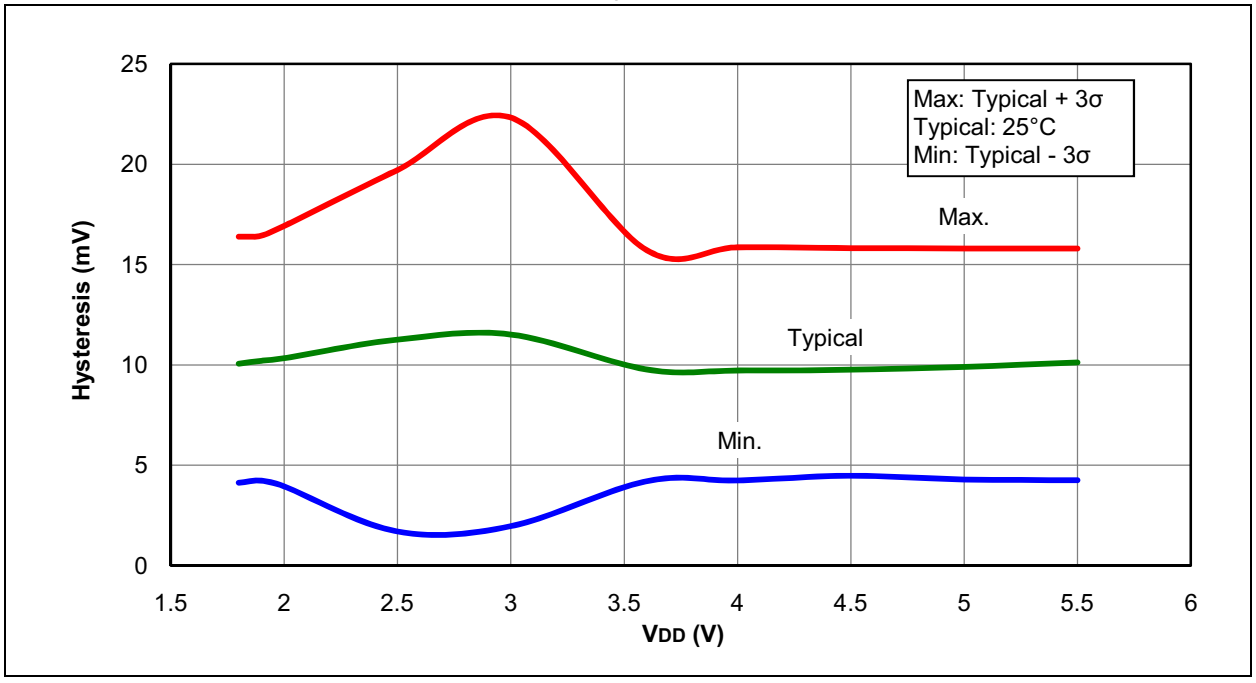



**FIGURE 31-62: COMPARATOR HYSTERESIS, LOW-POWER MODE**





© 2010-2016 Microchip Technology Inc.

**FIGURE 31-63: COMPARATOR RESPONSE TIME, HIGH-POWER MODE**

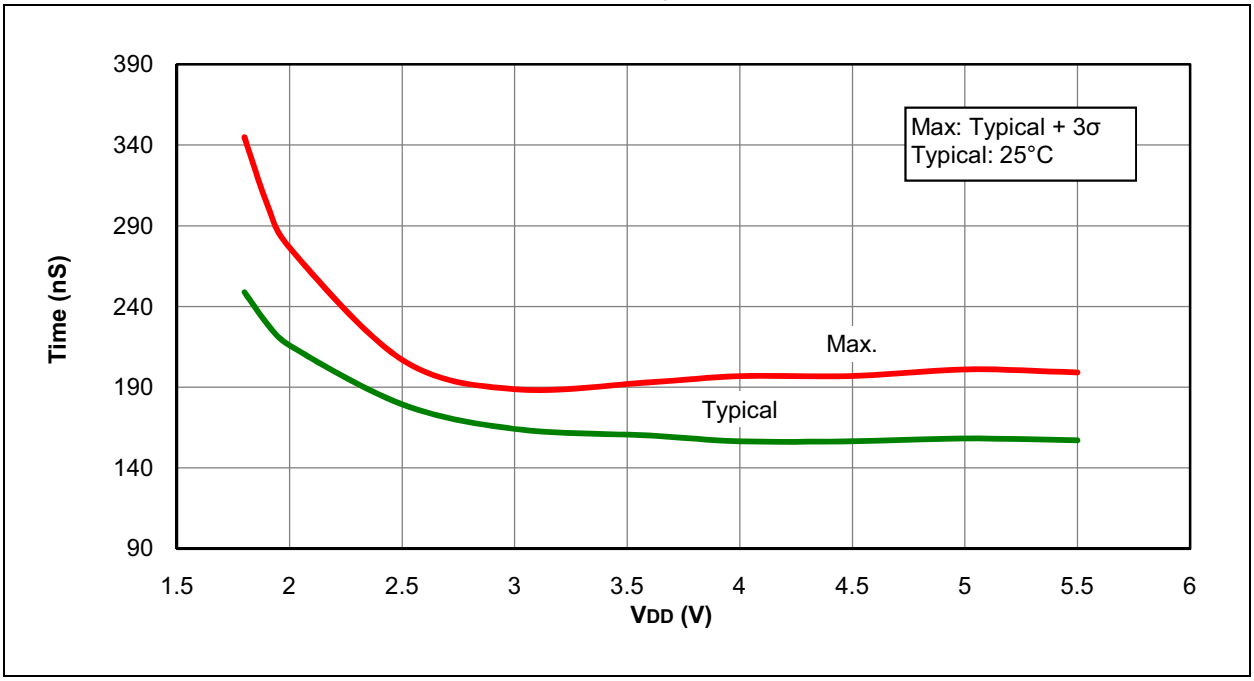



**FIGURE 31-64: COMPARATOR RESPONSE TIME OVER TEMPERATURE, HIGH-POWER MODE**

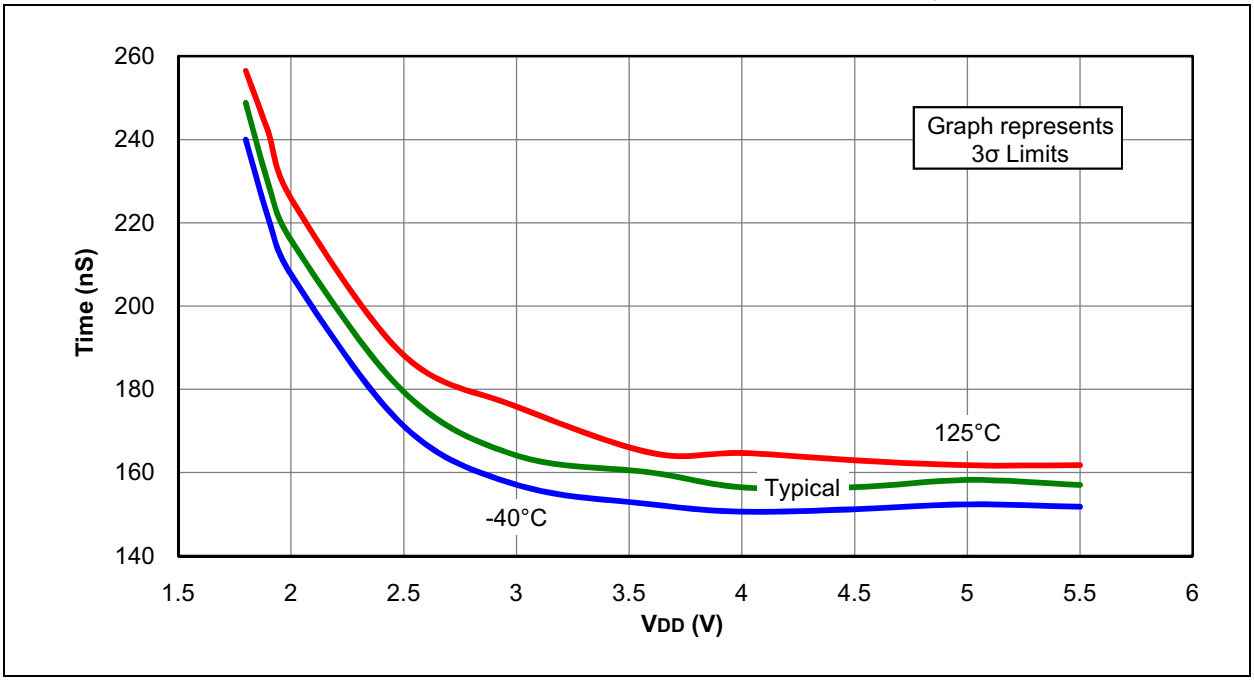
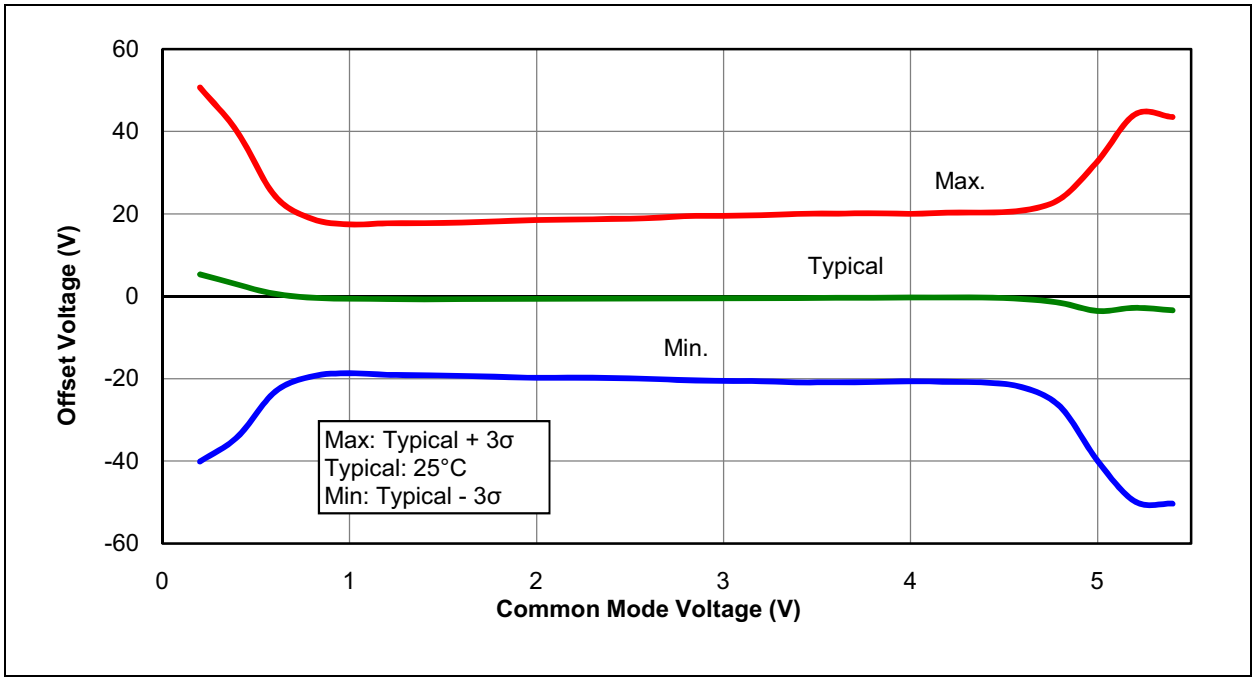


**FIGURE 31-65: COMPARATOR INPUT OFFSET AT 25°C, HIGH-POWER MODE, PIC16F1946/47 ONLY**





© 2010-2016 Microchip Technology Inc.

## 32.0 DEVELOPMENT SUPPORT

The PIC® microcontrollers (MCU) and dsPIC® digital signal controllers (DSC) are supported with a full range of software and hardware development tools:

- Integrated Development Environment
  - MPLAB® X IDE Software
- Compilers/Assemblers/Linkers
  - MPLAB XC Compiler
  - MPASM™ Assembler
  - MPLINK™ Object Linker/ MPLIB™ Object Librarian
  - MPLAB Assembler/Linker/Librarian for Various Device Families
- Simulators
  - MPLAB X SIM Software Simulator
- Emulators
  - MPLAB REAL ICE™ In-Circuit Emulator
- In-Circuit Debuggers/Programmers
  - MPLAB ICD 3
  - PICkit™ 3
- Device Programmers
  - MPLAB PM3 Device Programmer
- Low-Cost Demonstration/Development Boards, Evaluation Kits and Starter Kits
- Third-party development tools

### 32.1 MPLAB X Integrated Development Environment Software

The MPLAB X IDE is a single, unified graphical user interface for Microchip and third-party software, and hardware development tool that runs on Windows®, Linux and Mac OS® X. Based on the NetBeans IDE, MPLAB X IDE is an entirely new IDE with a host of free software components and plug-ins for high-performance application development and debugging. Moving between tools and upgrading from software simulators to hardware debugging and programming tools is simple with the seamless user interface.

With complete project management, visual call graphs, a configurable watch window and a feature-rich editor that includes code completion and context menus, MPLAB X IDE is flexible and friendly enough for new users. With the ability to support multiple tools on multiple projects with simultaneous debugging, MPLAB X IDE is also suitable for the needs of experienced users.

Feature-Rich Editor:

- Color syntax highlighting
- Smart code completion makes suggestions and provides hints as you type
- Automatic code formatting based on user-defined rules
- Live parsing

User-Friendly, Customizable Interface:

- Fully customizable interface: toolbars, toolbar buttons, windows, window placement, etc.
- Call graph window

Project-Based Workspaces:

- Multiple projects
- Multiple tools
- Multiple configurations
- Simultaneous debugging sessions

File History and Bug Tracking:

- Local file history feature
- Built-in support for Bugzilla issue tracker

## 32.2 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

## 32.3 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

## 32.4 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

## 32.5 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


© 2010-2016 Microchip Technology Inc.

## 32.6 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.

## 32.7 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.

## 32.8 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.

## 32.9 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™).

## 32.10 MPLAB PM3 Device Programmer

The MPLAB PM3 Device Programmer is a universal, CE compliant device programmer with programmable voltage verification at $V_{DDMIN}$ and $V_{DDMAX}$ for maximum reliability. It features a large LCD display (128 x 64) for menus and error messages, and a modular, detachable socket assembly to support various package types. The ICSP cable assembly is included as a standard item. In Stand-Alone mode, the MPLAB PM3 Device Programmer can read, verify and program PIC devices without a PC connection. It can also set code protection in this mode. The MPLAB PM3 connects to the host PC via an RS-232 or USB cable. The MPLAB PM3 has high-speed communications and optimized algorithms for quick programming of large memory devices, and incorporates an MMC card for file storage and data applications.

## 32.11 Demonstration/Development Boards, Evaluation Kits, and Starter Kits

A wide variety of demonstration, development and evaluation boards for various PIC MCUs and dsPIC DSCs allows quick application development on fully functional systems. Most boards include prototyping areas for adding custom circuitry and provide application firmware and source code for examination and modification.

The boards support a variety of features, including LEDs, temperature sensors, switches, speakers, RS-232 interfaces, LCD displays, potentiometers and additional EEPROM memory.

The demonstration and development boards can be used in teaching environments, for prototyping custom circuits and for learning about various microcontroller applications.

In addition to the PICDEM™ and dsPICDEM™ demonstration/development board series of circuits, Microchip has a line of evaluation kits and demonstration software for analog filter design, KEELOQ® security ICs, CAN, IrDA®, PowerSmart battery management, SEEVAL® evaluation system, Sigma-Delta ADC, flow rate sensing, plus many more.

Also available are starter kits that contain everything needed to experience the specified device. This usually includes a single application and debug capability, all on one board.

Check the Microchip web page (www.microchip.com) for the complete list of demonstration, development and evaluation kits.

## 32.12 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.

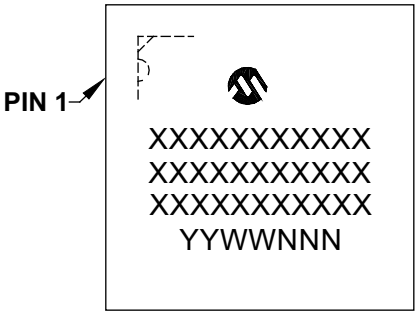
- Device Programmers and Gang Programmers from companies, such as SoftLog and CCS
- Software Tools from companies, such as Gimpel and Trace Systems
- Protocol Analyzers from companies, such as Saleae and Total Phase
- Demonstration Boards from companies, such as MikroElektronika, Digilent® and Olimex
- Embedded Ethernet Solutions from companies, such as EZ Web Lynx, WIZnet and IPLogika®


© 2010-2016 Microchip Technology Inc.

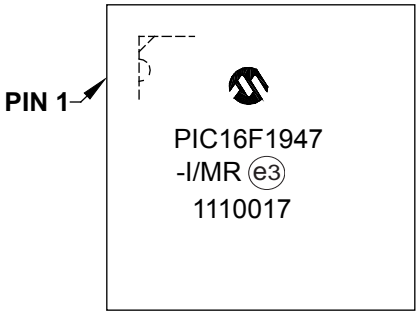
## 33.0 PACKAGING INFORMATION

### 33.1 Package Marking Information

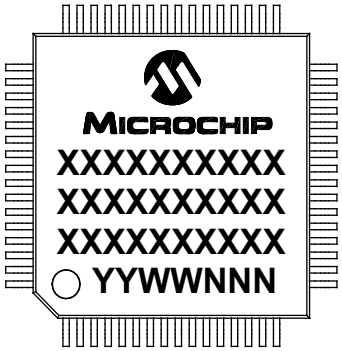
64-Lead QFN (9x9x0.9 mm)



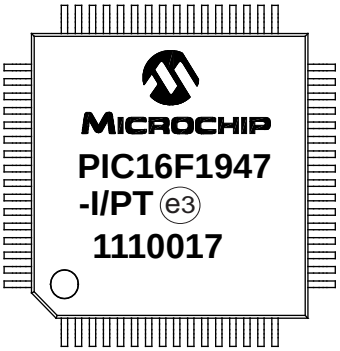

Example




64-Lead TQFP (10x10x1 mm)




Example




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

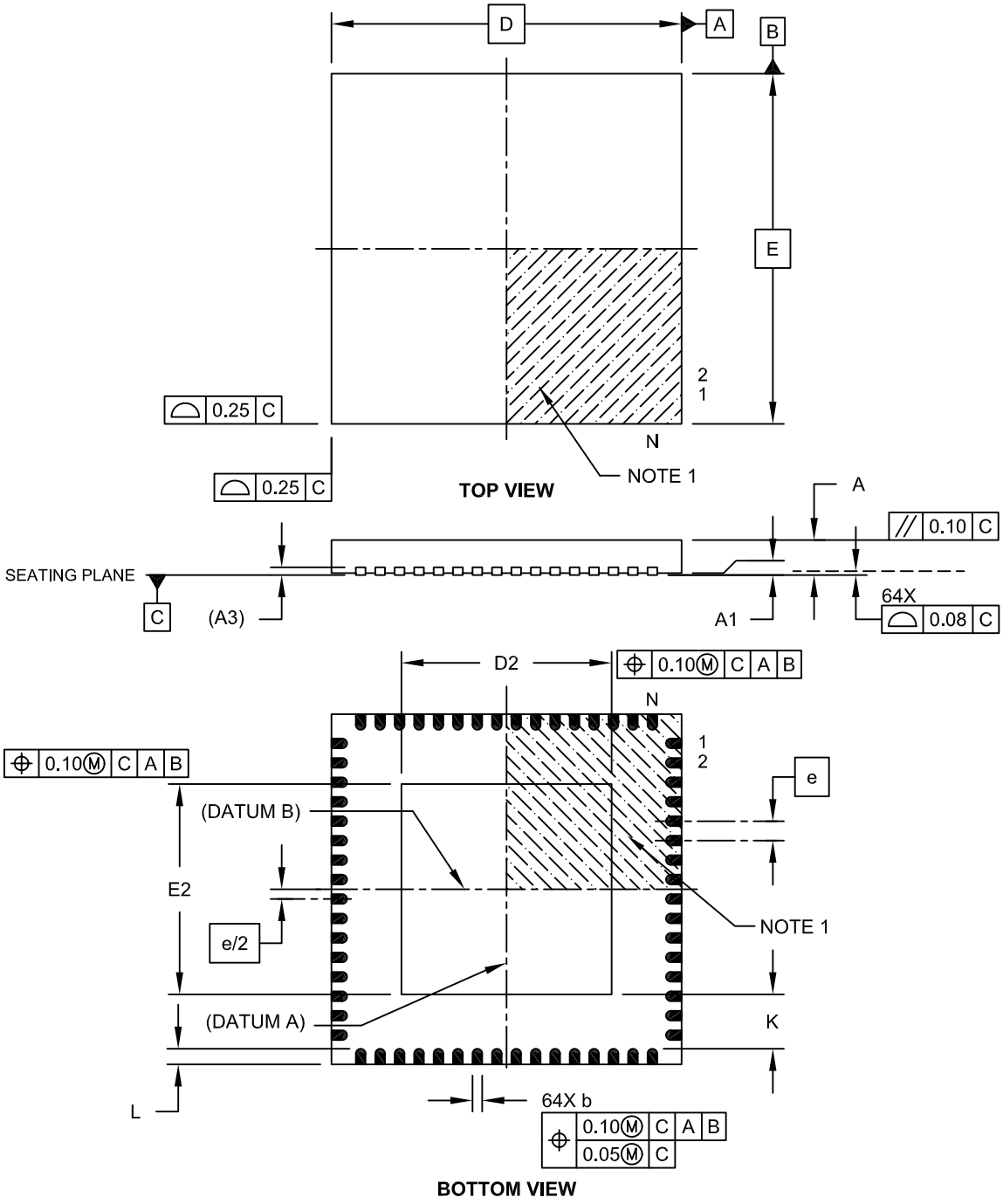
**Note**: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.

## 33.2 Package Details

The following sections give the technical details of the packages.

### 64-Lead Plastic Quad Flat, No Lead Package (MR) – 9x9x0.9 mm Body with 5.40 x 5.40 Exposed Pad [QFN]

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

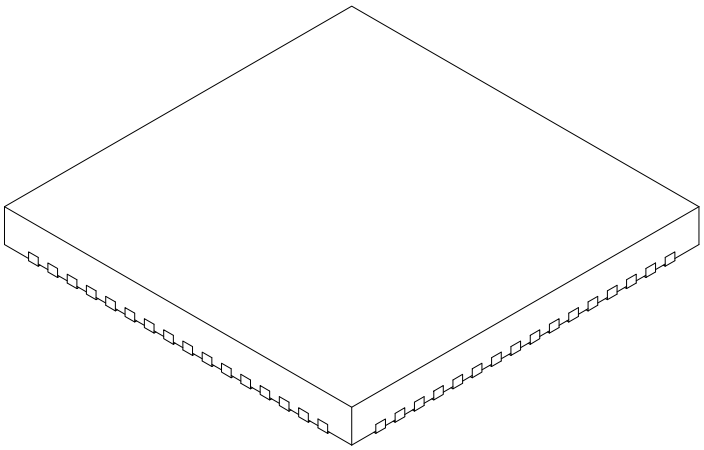



Microchip Technology Drawing C04-154A Sheet 1 of 2


© 2010-2016 Microchip Technology Inc.

### 64-Lead Plastic Quad Flat, No Lead Package (MR) – 9x9x0.9 mm Body with 5.40 x 5.40 Exposed Pad [QFN]

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



| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Pins | N | | 64 | |
| 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 | | 9.00 BSC | |
| Exposed Pad Width | E2 | 5.30 | 5.40 | 5.50 |
| Overall Length | D | | 9.00 BSC | |
| Exposed Pad Length | D2 | 5.30 | 5.40 | 5.50 |
| Contact Width | b | 0.20 | 0.25 | 0.30 |
| Contact Length | L | 0.30 | 0.40 | 0.50 |
| Contact-to-Exposed Pad | K | 0.20 | - | - |

**Notes:**

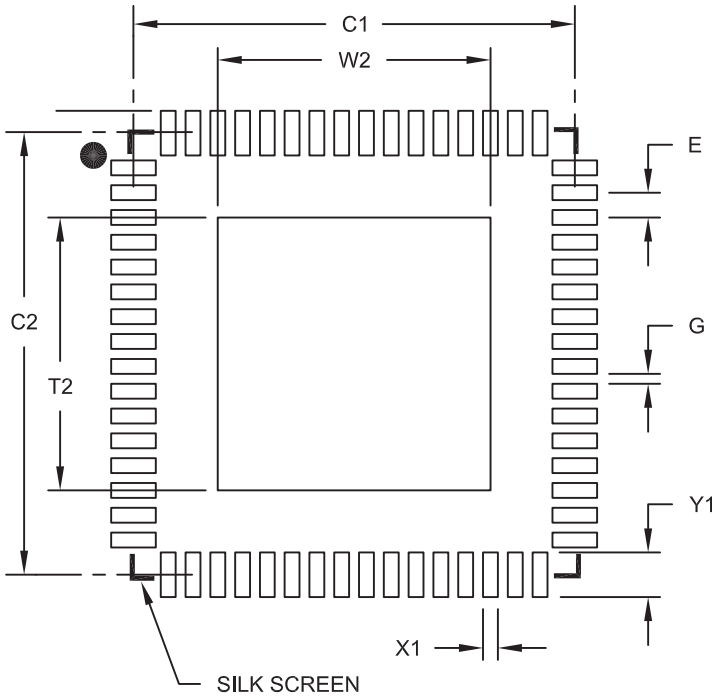
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated.
3. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-154A Sheet 2 of 2

64-Lead Plastic Quad Flat, No Lead Package (MR) – 9x9x0.9 mm Body [QFN]
With 0.40 mm Contact Length and 5.40x5.40mm Exposed Pad

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




RECOMMENDED LAND PATTERN

| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Contact Pitch | E | 0.50 BSC | | |
| Optional Center Pad Width | W2 | | | 5.50 |
| Optional Center Pad Length | T2 | | | 5.50 |
| Contact Pad Spacing | C1 | | 8.90 | |
| Contact Pad Spacing | C2 | | 8.90 | |
| Contact Pad Width (X64) | X1 | | | 0.30 |
| Contact Pad Length (X64) | Y1 | | | 0.85 |
| Distance Between Pads | G | 0.20 | | |

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

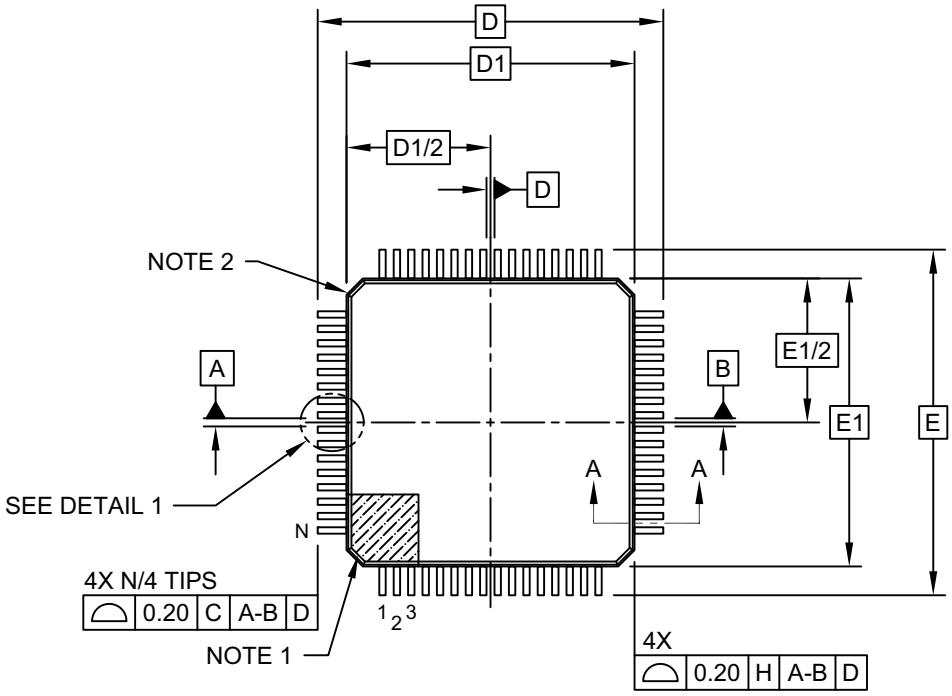
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2154A

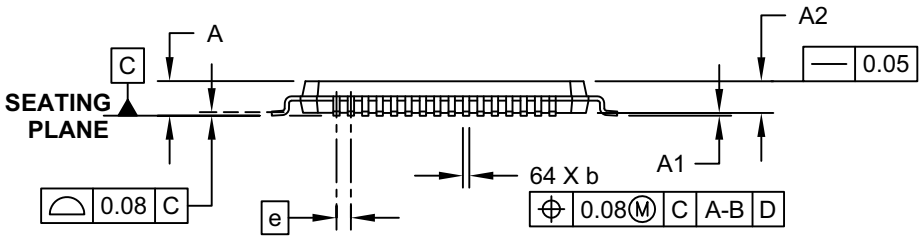

© 2010-2016 Microchip Technology Inc.

**64-Lead Plastic Thin Quad Flatpack (PT)-10x10x1 mm Body, 2.00 mm Footprint [TQFP]**

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




TOP VIEW

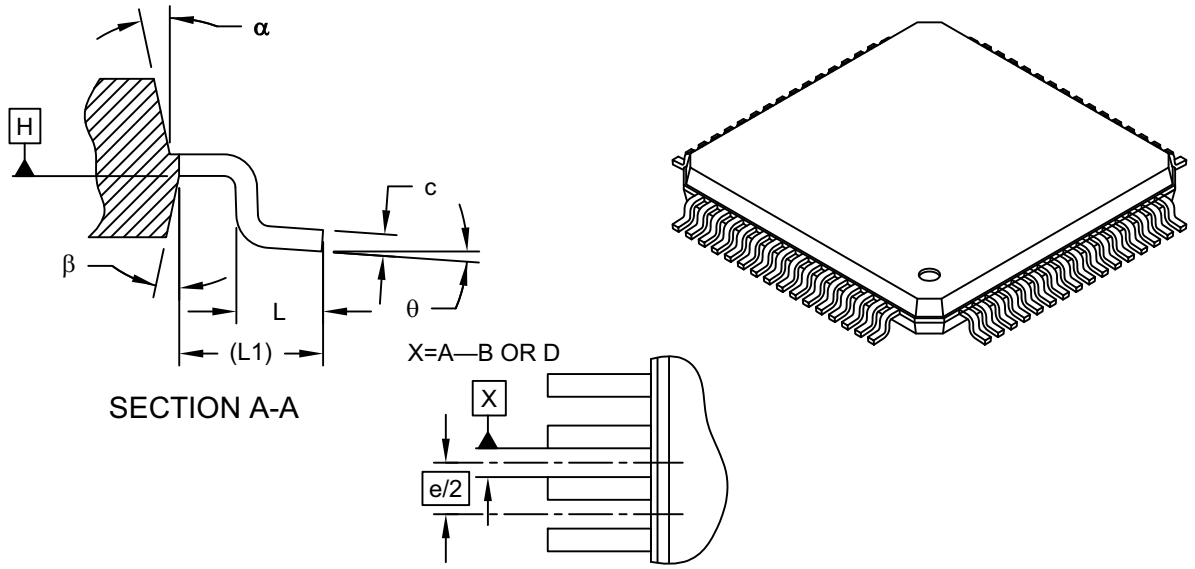



SIDE VIEW

Microchip Technology Drawing C04-085C Sheet 1 of 2

### 64-Lead Plastic Thin Quad Flatpack (PT)-10x10x1 mm Body, 2.00 mm Footprint [TQFP]

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




| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Leads | N | | 64 | |
| Lead Pitch | e | | 0.50 BSC | |
| Overall Height | A | - | - | 1.20 |
| Molded Package Thickness | A2 | 0.95 | 1.00 | 1.05 |
| Standoff | A1 | 0.05 | - | 0.15 |
| Foot Length | L | 0.45 | 0.60 | 0.75 |
| Footprint | L1 | | 1.00 REF | |
| Foot Angle | ϕ | 0° | 3.5° | 7° |
| Overall Width | E | | 12.00 BSC | |
| Overall Length | D | | 12.00 BSC | |
| Molded Package Width | E1 | | 10.00 BSC | |
| Molded Package Length | D1 | | 10.00 BSC | |
| Lead Thickness | c | 0.09 | - | 0.20 |
| Lead Width | b | 0.17 | 0.22 | 0.27 |
| Mold Draft Angle Top | α | 11° | 12° | 13° |
| Mold Draft Angle Bottom | β | 11° | 12° | 13° |

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Chamfers at corners are optional; size may vary.
3. Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M

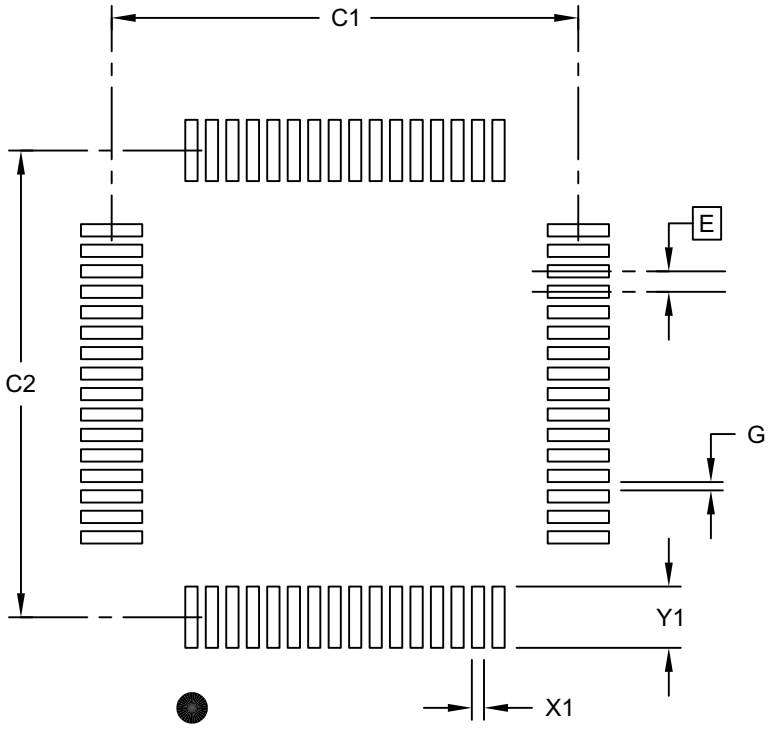
BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-085C Sheet 2 of 2


© 2010-2016 Microchip Technology Inc.

### 64-Lead Plastic Thin Quad Flatpack (PT)-10x10x1 mm Body, 2.00 mm Footprint [TQFP]

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




RECOMMENDED LAND PATTERN

| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Contact Pitch | E | | 0.50 BSC | |
| Contact Pad Spacing | C1 | | 11.40 | |
| Contact Pad Spacing | C2 | | 11.40 | |
| Contact Pad Width (X28) | X1 | | | 0.30 |
| Contact Pad Length (X28) | Y1 | | | 1.50 |
| 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 C04-2085B Sheet 1 of 1

**NOTES:**


© 2010-2016 Microchip Technology Inc.

# APPENDIX A: DATA SHEET REVISION HISTORY

## Revision A (3/2010)

Original release.

## Revision B (9/2010)

Updated with current electrical specifications; Added Temperature Indicator Module section; Other minor corrections.

## Revision C (5/2011)

Updated the EUSART section; Updated the Electrical Specifications section; Updated Table 3-8, Figure 13-1 and Equation 16-1.

## Revision D (02/2012)

Updated Electrical Specifications and added Characterization Graphs.

## Revision E (11/2016)

Updated Electrical Specification; Updated Table 1-2, Example 3-2, Figure 5-7, Added Section 5.3.5, Updated Table 5-1, Figure 6-1, Table 12-1, Table 12-3, Table 12-5, Table 12-7, Table 12-9, Table 12-11, Table 12-14, Table 12-17, Added Section 15-4, Updated Figure 16-1, Example 16-1, Table 16-3, Table 19-2; Other minor corrections.

# APPENDIX B: MIGRATING FROM OTHER PIC® DEVICES

This shows a comparison of features in the migration from the PIC16F917 device to the PIC16F1946 family of devices.

## B.1 PIC16F917 to PIC16F1946

**TABLE B-1: FEATURE COMPARISON**

| Feature | PIC16F917 | PIC16F1946 |
|---|---|---|
| Max. Operating Speed | 20 MHz | 32 MHz |
| Max. Program Memory (Words) | 8K | 8K |
| Max. SRAM (Bytes) | 368 | 512 |
| A/D Resolution | 10-bit | 10-bit |
| Timers (8/16-bit) | 2/1 | 4/1 |
| Oscillator Modes | 4 | 8 |
| Brown-out Reset | Y | Y |
| Internal Pull-ups | RB<7:0> | RB<7:0> |
| Interrupt-on-change | RB<7:4> | RB<7:0> |
| Comparator | 2 | 2 |
| AUSART/EUSART | 1/0 | 0/2 |
| Extended WDT | Y | Y |
| Software Control Option of WDT/BOR | N | Y |
| INTOSC Frequencies | 30 kHz - 8 MHz | 31 kHz - 16 MHz |
| Clock Switching | Y | Y |
| Capacitive Sensing | N | Y |
| CCP/ECCP | 2/0 | 2/3 |
| Enhanced PIC16 CPU | N | Y |
| MSSP/SSP | 0/1 | 2/0 |
| LCD | Y | Y |

## THE MICROCHIP WEBSITE

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

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

## CUSTOMER CHANGE NOTIFICATION SERVICE

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

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

## CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

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

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

**Technical support is available through the website at: http://microchip.com/support**


© 2010-2016 Microchip Technology Inc.

## PRODUCT IDENTIFICATION SYSTEM

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

| PART NO. | [X](1) | X | /XX | XXX |
|---|---|---|---|---|
| Device | Tape and Reel Option | Temperature Range | Package | Pattern |

| | |
|---|---|
| **Device:** | PIC16F1946, PIC16LF1946, PIC16F1947, PIC16LF1947 |
| **Tape and Reel Option:** | Blank = Standard packaging (tube or tray)<br>T = Tape and Reel(1) |
| **Temperature Range:** | I = -40°C to +85°C (Industrial)<br>E = -40°C to +125°C (Extended) |
| **Package:** | PT = TQFP (Thin Quad Flatpack)<br>MR = QFN |
| **Pattern:** | QTP, SQTP, Code or Special Requirements (blank otherwise) |

**Examples:**

a) PIC16LF1946 - I/MR 301 = Industrial temp., PLCC package, Extended $V_{DD}$ limits, QTP pattern #301.

b) PIC16LF1947 - I/PT = Industrial temp., TQFP package, Extended $V_{DD}$ limits.

**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.

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

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

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

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

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

**QUALITY MANAGEMENT SYSTEM**
**CERTIFIED BY DNV**
**═ ISO/TS 16949 ═**

**Trademarks**

The Microchip name and logo, the Microchip logo, AnyRate, dsPIC, FlashFlex, flexPWR, Heldo, JukeBlox, KeeLoq, KeeLoq logo, Kleer, LANCheck, LINK MD, MediaLB, MOST, MOST logo, MPLAB, OptoLyzer, PIC, PICSTART, PIC32 logo, RightTouch, SpyNIC, SST, SST Logo, SuperFlash and UNI/O are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

ClockWorks, The Embedded Control Solutions Company, ETHERSYNCH, Hyper Speed Control, HyperLight Load, IntelliMOS, mTouch, Precision Edge, and QUIET-WIRE are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, BodyCom, chipKIT, chipKIT logo, CodeGuard, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, EtherGREEN, In-Circuit Serial Programming, ICSP, Inter-Chip Connectivity, JitterBlocker, KleerNet, KleerNet logo, MiWi, motorBench, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, PureSilicon, RightTouch logo, REAL ICE, Ripple Blocker, Serial Quad I/O, SQI, SuperSwitcher, SuperSwitcher II, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

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

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

GestIC is a registered 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.

© 2010-2016, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

ISBN: 978-1-5224-1072-0



© 2010-2016 Microchip Technology Inc.

# Worldwide Sales and Service

## AMERICAS

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

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

**Austin, TX**
Tel: 512-257-3370

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

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

**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
Tel: 317-536-2380

**Los Angeles**
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608
Tel: 951-273-7800

**Raleigh, NC**
Tel: 919-844-7510

**New York, NY**
Tel: 631-435-6000

**San Jose, CA**
Tel: 408-735-9110
Tel: 408-436-4270

**Canada - Toronto**
Tel: 905-695-1980
Fax: 905-695-2078

## 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 - Guangzhou**
Tel: 86-20-8755-8029

**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: 86-21-3326-8000
Fax: 86-21-3326-8021

**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: 886-7-213-7830

**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

**Finland - Espoo**
Tel: 358-9-4520-820

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

**France - Saint Cloud**
Tel: 33-1-30-60-70-00

**Germany - Garching**
Tel: 49-8931-9700

**Germany - Haan**
Tel: 49-2129-3766400

**Germany - Heilbronn**
Tel: 49-7131-67-3636

**Germany - Karlsruhe**
Tel: 49-721-625370

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

**Germany - Rosenheim**
Tel: 49-8031-354-560

**Israel - Ra'anana**
Tel: 972-9-744-7705

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

**Italy - Padova**
Tel: 39-049-7625286

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

**Norway - Trondheim**
Tel: 47-7289-7561

**Poland - Warsaw**
Tel: 48-22-3325737

**Romania - Bucharest**
Tel: 40-21-407-87-50

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

**Sweden - Gothenberg**
Tel: 46-31-704-60-40

**Sweden - Stockholm**
Tel: 46-8-5090-4654

**UK - Wokingham**
Tel: 44-118-921-5800
Fax: 44-118-921-5820

© 2010-2016 Microchip Technology Inc.

# Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

Microchip:

PIC16F1946-E/PT PIC16F1946-I/PT PIC16F1946T-I/PT PIC16F1947-E/PT PIC16F1947-I/PT PIC16F1947T-I/PT PIC16LF1946-E/PT PIC16LF1946-I/PT PIC16LF1946T-I/PT PIC16LF1947-E/PT PIC16LF1947-I/PT PIC16LF1947T-I/PT PIC16F1946-E/MR PIC16F1946-I/MR PIC16F1946T-I/MR PIC16F1947-E/MR PIC16F1947-I/MR PIC16F1947T-I/MR PIC16LF1946-E/MR PIC16LF1946-I/MR PIC16LF1946T-I/MR PIC16LF1947-E/MR PIC16LF1947-I/MR PIC16LF1947T-I/MR

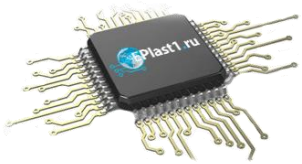
Компания «ЭлектроПласт» предлагает заключение долгосрочных отношений при поставках импортных электронных компонентов на взаимовыгодных условиях!

Наши преимущества:

- Оперативные поставки широкого спектра электронных компонентов отечественного и импортного производства напрямую от производителей и с крупнейших мировых складов;
- Поставка более 17-ти миллионов наименований электронных компонентов;
- Поставка сложных, дефицитных, либо снятых с производства позиций;
- Оперативные сроки поставки под заказ (от 5 рабочих дней);
- Экспресс доставка в любую точку России;
- Техническая поддержка проекта, помощь в подборе аналогов, поставка прототипов;
- Система менеджмента качества сертифицирована по Международному стандарту ISO 9001;
- Лицензия ФСБ на осуществление работ с использованием сведений, составляющих государственную тайну;
- Поставка специализированных компонентов (Xilinx, Altera, Analog Devices, Intersil, Interpoint, Microsemi, Aeroflex, Peregrine, Syfer, Eurofarad, Texas Instrument, Miteq, Cobham, E2V, MA-COM, Hittite, Mini-Circuits,General Dynamics и др.);

Помимо этого, одним из направлений компании «ЭлектроПласт» является направление «Источники питания». Мы предлагаем Вам помощь Конструкторского отдела:

- Подбор оптимального решения, техническое обоснование при выборе компонента;
- Подбор аналогов;
- Консультации по применению компонента;
- Поставка образцов и прототипов;
- Техническая поддержка проекта;
- Защита от снятия компонента с производства.




**Как с нами связаться**

**Телефон:** 8 (812) 309 58 32 (многоканальный)
**Факс:** 8 (812) 320-02-42
**Электронная почта:** org@eplast1.ru
**Адрес:** 198099, г. Санкт-Петербург, ул. Калинина, дом 2, корпус 4, литера А.