# PIC16(L)F1773/6

## 28-Pin, 8-Bit Flash Microcontroller

### Description

PIC16(L)F1773/6 microcontrollers feature a high level of integration of intelligent analog and digital peripherals for a wide range of applications, such as lighting, power supplies, battery charging, motor control and other general purpose applications. These devices deliver multiple op amps, 5-/10-bit DACs, high-speed comparators, 10-bit ADC, 10-/16-bit PWMs, programmable ramp generator (PRG) and other peripherals that can be connected internally to create closed-loop systems without using pins or the printed circuit board (PCB) area. The 10-/16-bit PWMs, digital signal modulators and tri-state output op amp can be used together to create a LED dimming engine for lighting applications. The peripheral pin select (PPS) functionality provides flexibility, eases PCB layout and peripheral utilization by allowing digital peripheral pin mapping to an I/O.

### Core Features

- C Compiler Optimized RISC Architecture
- Only 49 Instructions
- Operating Speed:
  - DC – 32 MHz clock input
  - 125 ns minimum instruction cycle
- Interrupt Capability
- 16-Level Deep Hardware Stack
- Five 8-Bit Timers
- Three 16-Bit Timers
- Low-Current Power-on Reset (POR)
- Configurable Power-up Timer (PWRT)
- Brown-out Reset (BOR) with Selectable Trip Point
- Extended Watchdog Timer (EWDT):
  - Low-power 31 kHz WDT
  - Software selectable prescaler
  - Software selectable enable

### Memory

- Up to 28 Kbytes Program Flash Memory (PFM)
- Up to 2 Kbytes Data RAM
- Direct, Indirect and Relative Addressing modes
- High-Endurance Flash (HEF):
  - 128B of nonvolatile data storage
  - 100K Erase/Write cycles

### Operating Characteristics

- Operating Voltage Range:
  - 1.8V to 3.6V (PIC16LF1773/6)
  - 2.3V to 5.5V (PIC16F1773/6)
- Temperature Range:
  - Industrial: -40°C to 85°C
  - Extended: -40°C to 125°C

### eXtreme Low-Power (XLP) Features

- Sleep mode: 50 nA @ 1.8V, typical
- Watchdog Timer: 500 nA @ 1.8V, typical
- Secondary Oscillator: 500 nA @ 32 kHz
- Operating Current:
  - 8 uA @ 31 kHz, 1.8V, typical
  - 32 uA/MHz @ 1.8V, typical

### Intelligent Analog Peripherals

- 10-Bit Analog-to-Digital Converter (ADC):
  - Up to 28 external channels
  - Conversion available during Sleep
- Three Operational Amplifiers (OPA):
  - Selectable internal and external channels
  - Tri-state output
  - Part of LED dimming engine
  - Selectable internal and external channels
- Six High-Speed Comparators (HS Comp):
  - Up to nine external inverting inputs
  - Up to 12 external non-inverting inputs
  - Fixed Voltage Reference at inverting and non-inverting input(s)
  - Comparator outputs externally accessible
- Digital-to-Analog Converters (DAC):
  - Three 10-bit resolution DACs
  - 10-bit resolution, rail-to-rail
  - Conversion during Sleep
  - Internal connections to ADCs and HS Comparators
- Voltage Reference:
  - Fixed Voltage Reference (FVR)
  - 1.024V, 2.048V and 4.096V output levels
- Zero-Cross Detector (ZCD):
  - Detect high-voltage AC signal
- Three Programmable Ramp Generators (PRG):
  - Slope compensation
  - Ramp generation
- High-Current Drive I/Os:
  - Up to 100 mA sink or source @ 5V

## Digital Peripherals

- Four Configurable Logic Cells (CLC):
  - Integrated combinational and state logic
- Three Complementary Output Generators (COG):
  - Push-pull, Full-Bridge and Steering modes
- Three Capture/Compare/PWM (CCP) Modules
- Pulse-Width Modulator (PWM):
  - Three 16-bit PWMs
  - Independent timers
  - Multiple output modes (Edge-, Center-Aligned, set and toggle on register match)
  - User settings for phase, duty cycle, period, offset and polarity
  - 16-bit timer capability
  - Three 10-bit PWMs
- Digital Signal Modulator (DSM):
  - Modulates a carrier signal with a digital data to create custom carrier synchronized output waveforms
  - Part of LED dimming engine
- Peripheral Pin Select (PPS):
  - I/O remapping of digital peripherals
- Serial Communications:
  - Enhanced USART (EUSART)
  - SPI, $I^2C$, RS-232, RS-485, LIN compatible
  - Auto-Baud Detect, auto-wake-up on start
- Up to 25 I/O Pins:
  - Individually programmable pull-ups
  - Slew rate control
  - Interrupt-on-change with edge-select

## Clocking Structure

- Precision Internal Oscillator:
  - ±1% at calibration
  - Selectable frequency range 32 MHz to 31 kHz
- 31 kHz Low-Power Internal Oscillator
- 4x Phase-Locked Loop (PLL) for up to 32 MHz Internal Operation
- External Oscillator Block with Three External Clock modes up to 32 MHz

**TABLE 1: PIC16(L)F1773/6/7/8/9 FAMILY TYPES**

| Device | Data Sheet Index | Program Flash Memory (bytes) | Program Flash Memory (word) | High Endurance Flash (B) | Data SRAM (Bytes) | I/O Pins(1) | 8-Bit/16-Bit Timers | High-Speed Comparator | 10-bit ADC (ch) | 5/10-bit DAC | CCP | 10-bit/16-bit PWM | COG | CLC | Op Amp | Zero Cross Detect | Programmable Ramp Gen | High-Current I/Os | Peripheral Pin Select | EUSART | $I^2C$/SPI | Debug(2) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PIC16(L)F1773 | (A) | 7K | 4K | 128 | 512 | 25 | 5/3 | 6 | 17 | 3/3 | 3 | 3/3 | 3 | 4 | 3 | 1 | 3 | 2 | Y | 1 | 1 | I/H |
| PIC16(L)F1776 | (A) | 14K | 8K | 128 | 1K | 25 | 5/3 | 6 | 17 | 3/3 | 3 | 3/3 | 3 | 4 | 3 | 1 | 3 | 2 | Y | 1 | 1 | I/H |
| PIC16(L)F1777 | (B) | 14K | 8K | 128 | 1K | 36 | 5/3 | 8 | 28 | 4/4 | 4 | 4/4 | 4 | 4 | 4 | 1 | 4 | 2 | Y | 1 | 1 | I/H |
| PIC16(L)F1778 | (B) | 28K | 16K | 128 | 2K | 25 | 5/3 | 6 | 17 | 3/3 | 3 | 3/3 | 3 | 4 | 3 | 1 | 3 | 2 | Y | 1 | 1 | I/H |
| PIC16(L)F1779 | (B) | 28K | 16K | 128 | 2K | 36 | 5/3 | 8 | 28 | 4/4 | 4 | 4/4 | 4 | 4 | 4 | 1 | 4 | 2 | Y | 1 | 1 | I/H |

**Note 1:** One pin is input-only.
**2:** I – Debugging integrated on chip; H – Debugging via ICD header.

**Data Sheet Index:**

**A:** DS40001810 PIC16(L)F1773/6 Data Sheet, 28-Pin, 8-bit Flash Microcontrollers
**B:** Future Release PIC16(L)F1777/8/9 Data Sheet, 28/40/44-Pin, 8-bit Flash Microcontrollers

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

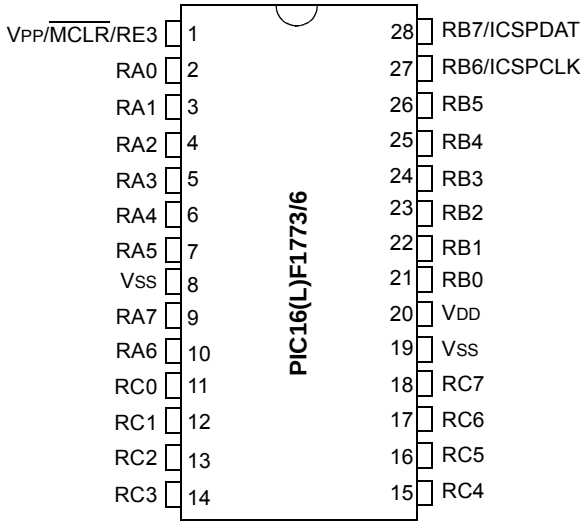
**TABLE 2: PACKAGES**

| Packages | SPDIP | PDIP | SOIC | SSOP | UQFN | TQFP |
|---|---|---|---|---|---|---|
| PIC16(L)F1773 | • | | • | • | • | |
| PIC16(L)F1776 | • | | • | • | • | |

**Note:** Pin details are subject to change.

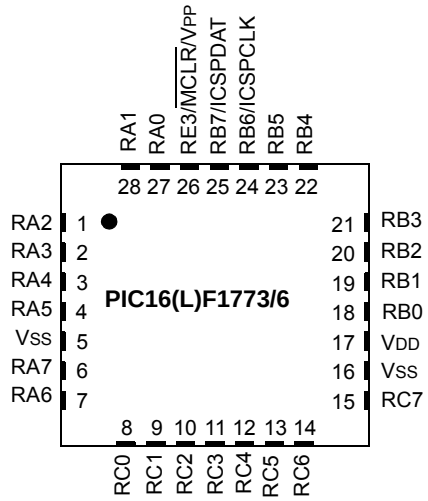
## PIN DIAGRAMS

**FIGURE 1: 28-PIN SPDIP, SOIC, SSOP**




**Note:** See Table 3 for location of all peripheral functions.

**FIGURE 2: 28-PIN UQFN (6x6x0.5 mm)**




**Note:** See Table 3 for location of all peripheral functions.

## PIN ALLOCATION TABLES

**TABLE 3: 28-PIN ALLOCATION TABLE (PIC16(L)F1773/6)**

| I/O | 28-Pin SPDIP/SOIC/SSOP | 28-Pin UQFN | PRG | ZCD | ADC | Voltage Reference | Comparator | Op Amp | DAC | Timer | Interrupt | Basic |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RA0 | 2 | 27 | — | — | AN0 | — | C1IN0-<br>C2IN0-<br>C3IN0-<br>C4IN0-<br>C5IN0-<br>C6IN0- | — | — | — | IOCA0 | — |
| RA1 | 3 | 28 | PRG1IN0<br>PRG2IN1 | — | AN1 | — | C1IN1-<br>C2IN1-<br>C3IN1-<br>C4IN1- | OPA1OUT<br>OPA2IN1+<br>OPA2IN1- | — | — | IOCA1 | — |
| RA2 | 4 | 1 | — | — | AN2<br>VREF- | DAC1REF0-<br>DAC2REF0-<br>DAC3REF0-<br>DAC4REF0-<br>DAC5REF0-<br>DAC7REF0- | C1IN0+<br>C2IN0+<br>C3IN0+<br>C4IN0+<br>C5IN0+<br>C6IN0+ | — | DAC1OUT1 | — | IOCA2 | — |
| RA3 | 5 | 2 | — | — | AN3<br>VREF+ | DAC1REF0+<br>DAC2REF0+<br>DAC3REF0+<br>DAC4REF0+<br>DAC5REF0+<br>DAC7REF0+ | C1IN1+ | — | — | — | IOCA3 | — |
| RA4 | 6 | 3 | — | — | — | — | — | OPA1IN0+ | DAC4OUT1 | T0CKI | IOCA4 | — |
| RA5 | 7 | 4 | — | — | AN4 | — | — | OPA1IN0- | DAC2OUT1 | — | IOCA5 | — |
| RA6 | 10 | 7 | — | — | — | — | CLKOUT<br>C6IN1+ | — | — | — | IOCA6 | OSC2 |
| RA7 | 9 | 6 | — | — | — | — | CLKIN | — | — | — | IOCA7 | OSC1 |
| RB0 | 21 | 18 | — | ZCD | AN12 | HIB0 | C2IN1+ | — | — | — | IOCB0 | — |
| RB1 | 22 | 19 | PRG1IN1<br>PRG2IN0 | — | AN10 | HIB1 | C1IN3-<br>C2IN3-<br>C3IN3-<br>C4IN3- | OPA2OUT<br>OPA1IN1+<br>OPA1IN1- | — | — | IOCB1 | — |
| RB2 | 23 | 20 | — | — | AN8 | — | — | OPA2IN0- | DAC3OUT1 | — | IOCB2 | — |
| RB3 | 24 | 21 | — | — | AN9 | — | C1IN2-<br>C2IN2-<br>C3IN2- | OPA2IN0+ | — | — | IOCB3 | — |
| RB4 | 25 | 22 | — | — | AN11 | — | C3IN1+ | — | — | T5G | IOCB4 | — |
| RB5 | 26 | 23 | — | — | AN13 | DAC5REF1-<br>DAC7REF1- | C4IN2- | — | — | T1G | IOCB5 | — |
| RB6 | 27 | 24 | — | — | — | DAC5REF1+<br>DAC7REF1+ | C4IN1+ | — | — | — | IOCB6 | ICSPCLK<br>ICDCLK |
| RB7 | 28 | 25 | — | — | — | — | C5IN1+ | — | DAC1OUT2<br>DAC2OUT2<br>DAC5OUT2<br>DAC3OUT2<br>DAC4OUT2<br>DAC7OUT2 | T6IN | IOCB7 | ICSPDAT<br>ICDDAT |
| RC0 | 11 | 8 | — | — | — | — | — | — | DAC5OUT1 | T1CKI<br>T3CKI<br>T3G | IOCC0 | — |
| RC1 | 12 | 9 | — | — | — | — | — | — | DAC7OUT1 | — | IOCC1 | — |

**TABLE 3: 28-PIN ALLOCATION TABLE (PIC16(L)F1773/6) (CONTINUED)**

| I/O | 28-Pin SPDIP/SOIC/SSOP | 28-Pin UQFN | PRG | ZCD | ADC | Voltage Reference | Comparator | Op Amp | DAC | Timer | Interrupt | Basic |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RC2 | 13 | 10 | — | — | AN14 | — | C5IN2-<br>C6IN2- | — | — | T5CKI | IOCC2 | — |
| RC3 | 14 | 11 | — | — | AN15 | — | C1IN4-<br>C2IN4-<br>C3IN4-<br>C4IN4-<br>C5IN4-<br>C6IN4- | — | — | T2IN | IOCC3 | — |
| RC4 | 15 | 12 | — | — | AN16 | — | C5IN3-<br>C6IN3- | — | — | T8IN | IOCC4 | — |
| RC5 | 16 | 13 | — | — | AN17 | — | — | OPA3IN0+ | — | T4IN | IOCC5 | — |
| RC6 | 17 | 14 | PRG3IN0 | — | AN18 | — | C5IN1-<br>C6IN1- | OPA3OUT | — | — | IOCC6 | — |
| RC7 | 18 | 15 | — | — | AN19 | — | — | OPA3IN0- | — | — | IOCC7 | — |
| RE3 | 1 | 26 | — | — | — | — | — | — | — | — | IOCE3 | $\overline{MCLR}$/VPP |
| VDD | 20 | 17 | — | — | VDD | — | — | — | — | — | — | — |
| VSS | 8 | 5 | — | — | VSS | — | — | — | — | — | — | — |
| VSS | 19 | 16 | — | — | — | — | — | — | — | — | — | — |

## Table of Contents

## TO OUR VALUED CUSTOMERS

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

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at **docerrors@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)F1773/6 are described within this data sheet. See Table 2 for available package configurations.

Figure 1-1 shows a block diagram of the PIC16(L)F1773/6 devices. Table 1-2 shows the pinout descriptions.

Refer to Table 1-1 for peripherals available per device.

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

| Peripheral | | PIC16(L)F1773 | PIC16(L)F1776 |
|---|---|---|---|
| Analog-to-Digital Converter (ADC) | | ● | ● |
| Fixed Voltage Reference (FVR) | | ● | ● |
| Zero-Cross Detection (ZCD) | | ● | ● |
| Temperature Indicator | | ● | ● |
| Complementary Output Generator (COG) | | | |
| | COG1 | ● | ● |
| | COG2 | ● | ● |
| | COG3 | ● | ● |
| Programmable Ramp Generator (PRG) | | | |
| | PRG1 | ● | ● |
| | PRG2 | ● | ● |
| | PRG3 | ● | ● |
| 10-bit Digital-to-Analog Converter (DAC) | | | |
| | DAC1 | ● | ● |
| | DAC2 | ● | ● |
| | DAC5 | ● | ● |
| 5-bit Digital-to-Analog Converter (DAC) | | | |
| | DAC3 | ● | ● |
| | DAC4 | ● | ● |
| | DAC7 | ● | ● |
| Capture/Compare/PWM (CCP/ECCP) Modules | | | |
| | CCP1 | ● | ● |
| | CCP2 | ● | ● |
| | CCP7 | ● | ● |
| Comparators | | | |
| | C1 | ● | ● |
| | C2 | ● | ● |
| | C3 | ● | ● |
| | C4 | ● | ● |
| | C5 | ● | ● |
| | C6 | ● | ● |

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

| Peripheral | | PIC16(L)F1773 | PIC16(L)F1776 |
|---|---|---|---|
| Configurable Logic Cell (CLC) | | | |
| | CLC1 | ● | ● |
| | CLC2 | ● | ● |
| | CLC3 | ● | ● |
| | CLC4 | ● | ● |
| Data Signal Modulator (DSM) | | | |
| | DSM1 | ● | ● |
| | DSM2 | ● | ● |
| | DSM3 | ● | ● |
| Enhanced Universal Synchronous/Asynchronous Receiver/Transmitter (EUSART) | | | |
| | EUSART | ● | ● |
| Master Synchronous Serial Ports | | | |
| | MSSP | ● | ● |
| Op Amps | | | |
| | Op Amp 1 | ● | ● |
| | Op Amp 2 | ● | ● |
| | Op Amp 3 | ● | ● |
| 10-bit Pulse-Width Modulator (PWM) | | | |
| | PWM3 | ● | ● |
| | PWM4 | ● | ● |
| | PWM9 | ● | ● |
| 16-bit Pulse-Width Modulator (PWM) | | | |
| | PWM5 | ● | ● |
| | PWM6 | ● | ● |
| | PWM11 | ● | ● |
| 8-bit Timers | | | |
| | Timer0 | ● | ● |
| | Timer2 | ● | ● |
| | Timer4 | ● | ● |
| | Timer6 | ● | ● |
| | Timer8 | ● | ● |
| 16-bit Timers | | | |
| | Timer1 | ● | ● |
| | Timer3 | ● | ● |
| | Timer5 | ● | ● |

## 1.1 Register and Bit naming conventions

### 1.1.1 REGISTER NAMES

When there are multiple instances of the same peripheral in a device, the peripheral control registers will be depicted as the concatenation of a peripheral identifier, peripheral instance, and control identifier. The control registers section will show just one instance of all the register names with an 'x' in the place of the peripheral instance number. This naming convention may also be applied to peripherals when there is only one instance of that peripheral in the device to maintain compatibility with other devices in the family that contain more than one.

### 1.1.2 BIT NAMES

There are two variants for bit names:

- Short name: Bit function abbreviation
- Long name: Peripheral abbreviation + short name

#### 1.1.2.1 Short Bit Names

Short bit names are an abbreviation for the bit function. For example, some peripherals are enabled with the EN bit. The bit names shown in the registers are the short name variant.

Short bit names are useful when accessing bits in C programs. The general format for accessing bits by the short name is *RegisterName*bits.*ShortName*. For example, the enable bit, EN, in the COG1CON0 register can be set in C programs with the instruction `COG1CON0bits.EN = 1`.

Short names are generally not useful in assembly programs because the same name may be used by different peripherals in different bit positions. When this occurs, during the include file generation, all instances of that short bit name are appended with an underscore plus the name of the register in which the bit resides to avoid naming contentions.

#### 1.1.2.2 Long Bit Names

Long bit names are constructed by adding a peripheral abbreviation prefix to the short name. The prefix is unique to the peripheral, thereby making every long bit name unique. The long bit name for the COG1 enable bit is the COG1 prefix, G1, appended with the enable bit short name, EN, resulting in the unique bit name G1EN.

Long bit names are useful in both C and assembly programs. For example, in C the COG1CON0 enable bit can be set with the `G1EN = 1` instruction. In assembly, this bit can be set with the `BSF COG1CON0,G1EN` instruction.

#### 1.1.2.3 Bit Fields

Bit fields are two or more adjacent bits in the same register. Bit fields adhere only to the short bit naming convention. For example, the three Least Significant bits of the COG1CON0 register contain the mode control bits. The short name for this field is MD. There is no long bit name variant. Bit field access is only possible in C programs. The following example demonstrates a C program instruction for setting the COG1 to the Push-Pull mode:

```
COG1CON0bits.MD = 0x5;
```

Individual bits in a bit field can also be accessed with long and short bit names. Each bit is the field name appended with the number of the bit position within the field. For example, the Most Significant mode bit has the short bit name MD2 and the long bit name is G1MD2. The following two examples demonstrate assembly program sequences for setting the COG1 to the Push-Pull mode:

**EXAMPLE 1-1:**

```
MOVLW  ~(1<<G1MD1)
ANDWF  COG1CON0,F
MOVLW  1<<G1MD2 | 1<<G1MD0
IORWF  COG1CON0,F
```

**EXAMPLE 1-2:**

```
BSF    COG1CON0,G1MD2
BCF    COG1CON0,G1MD1
BSF    COG1CON0,G1MD0
```

### 1.1.3 REGISTER AND BIT NAMING EXCEPTIONS

#### 1.1.3.1 Status, Interrupt, and Mirror Bits

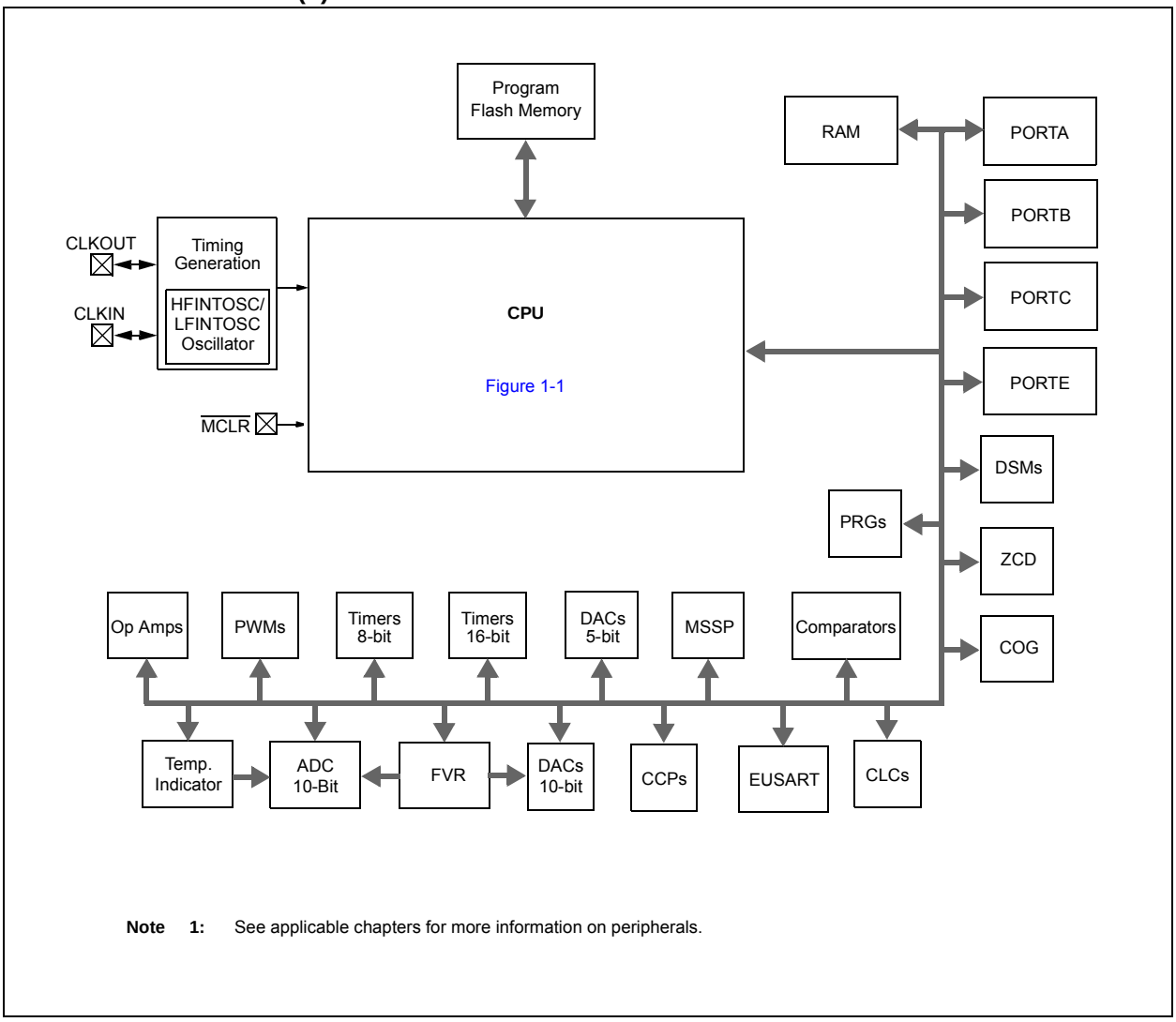
Status, interrupt enables, interrupt flags, and mirror bits are contained in registers that span more than one peripheral. In these cases, the bit name shown is unique so there is no prefix or short name variant.

#### 1.1.3.2 Legacy Peripherals

There are some peripherals that do not strictly adhere to these naming conventions. Peripherals that have existed for many years and are present in almost every device are the exceptions. These exceptions were necessary to limit the adverse impact of the new conventions on legacy code. Peripherals that do adhere to the new convention will include a table in the registers section indicating the long name prefix for each peripheral instance. Peripherals that fall into the exception category will not have this table. These peripherals include, but are not limited to, the following:

- EUSART
- MSSP

**FIGURE 1-1: PIC16(L)F1773/6 BLOCK DIAGRAM**




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

**TABLE 1-2: PIC16(L)F1773/6 PINOUT DESCRIPTION**

| Name | Function | Input Type | Output Type | Description |
|---|---|---|---|---|
| RA0/AN0/C1IN0-/C2IN0-/ C3IN0-/C4IN0-/C5IN0-/ C6IN0-/IOCA0 | RA0 | TTL/ST | CMOS | General purpose I/O. |
| | AN0 | AN | — | ADC Channel 0 input. |
| | C1IN0- | AN | — | Comparator C1 negative input. |
| | C2IN0- | AN | — | Comparator C2 negative input. |
| | C3IN0- | AN | — | Comparator C3 negative input. |
| | C4IN0- | AN | — | Comparator C4 negative input. |
| | C5IN0- | AN | — | Comparator C5 negative input. |
| | C6IN0- | AN | — | Comparator C6 negative input. |
| | IOCA0 | | | |
| RA1/AN1/C1IN1-/C2IN1-/ C3IN1-/C4IN1-/PRG1IN0/ PRG2IN1/OPA1OUT/OPA2IN1+/ OPA2IN1-/IOCA1 | RA1 | TTL/ST | CMOS | General purpose I/O. |
| | AN1 | AN | — | Channel 1 input. |
| | C1IN1- | AN | — | Comparator C1 negative input. |
| | C2IN1- | AN | — | Comparator C2 negative input. |
| | C3IN1- | AN | — | Comparator C3 negative input. |
| | C4IN1- | AN | — | Comparator C4 negative input. |
| | PRG1IN0 | AN | — | Ramp generator 1 reference voltage input. |
| | PRG2IN1 | AN | — | Ramp generator 2 reference voltage input. |
| | OPA1OUT | — | AN | Operational amplifier 1 output. |
| | OPA2IN1+ | AN | — | Operational amplifier 2 non-inverting input. |
| | OPA2IN1- | AN | — | Operational amplifier 2 inverting input. |
| | IOCA1 | | | |
| RA2/AN2/VREF-/DAC1REF0-/ DAC2REF0-/DAC3REF0-/ DAC4REF0-/DAC5REF0-/ DAC7REF0-/C1IN0+/C2IN0+/ C3IN0+/C4IN0+/C5IN0+/ C6IN0+/DAC1OUT1/IOCA2 | RA2 | TTL/ST | CMOS | General purpose I/O. |
| | AN2 | AN | — | ADC Channel 2 input. |
| | VREF- | AN | — | ADC negative reference. |
| | DAC1REF0- | AN | — | DAC1 negative reference. |
| | DAC2REF0- | AN | — | DAC2 negative reference. |
| | DAC3REF0- | AN | — | DAC3 negative reference. |
| | DAC4REF0- | AN | — | DAC4 negative reference. |
| | DAC5REF0- | AN | — | DAC5 negative reference. |
| | DAC7REF0- | AN | — | DAC7 negative reference. |
| | C1IN0+ | AN | — | Comparator C1 positive input. |
| | C2IN0+ | AN | — | Comparator C2 positive input. |
| | C3IN0+ | AN | — | Comparator C3 positive input. |
| | C4IN0+ | AN | — | Comparator C4 positive input. |
| | C5IN0+ | AN | — | Comparator C5 positive input. |
| | C6IN0+ | AN | — | Comparator C6 positive input. |
| | DAC1OUT1 | — | AN | DAC1 voltage output. |
| | IOCA2 | | | |

**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 $I^2C$
HV = High Voltage XTAL = Crystal levels

**Note 1:** Default peripheral input. Alternate pins can be selected as the peripheral input with the PPS input selection registers.
**2:** All pin digital outputs default to PORT latch data. Alternate outputs can be selected as the peripheral digital output with the PPS output selection registers.
**3:** These peripheral functions are bidirectional. The output pin selections must be the same as the input pin selections.

**TABLE 1-2: PIC16(L)F1773/6 PINOUT DESCRIPTION (CONTINUED)**

| Name | Function | Input Type | Output Type | Description |
|---|---|---|---|---|
| RA3/AN3/VREF+/DAC1REF0+/ DAC2REF0+/DAC3REF0+/ DAC4REF0+/DAC5REF0+/ DAC7REF0+/C1IN1+/IOCA3 | RA3 | TTL/ST | CMOS | General purpose I/O. |
| | AN3 | AN | — | ADC Channel 3 input. |
| | VREF+ | AN | — | ADC positive reference. |
| | DAC1REF0+ | AN | — | DAC1 positive reference. |
| | DAC2REF0+ | AN | — | DAC2 positive reference. |
| | DAC3REF0+ | AN | — | DAC3 positive reference. |
| | DAC4REF0+ | AN | — | DAC4 positive reference. |
| | DAC5REF0+ | AN | — | DAC5 positive reference. |
| | DAC7REF0+ | AN | — | DAC7 positive reference. |
| | C1IN1+ | AN | — | Comparator C1 positive input. |
| | IOCA3 | | | |
| RA4/OPA1IN0+/DAC4OUT1/ T0CKI/IOCA4 | RA4 | TTL/ST | CMOS | General purpose I/O. |
| | OPA1IN0+ | AN | — | Operational Amplifier 1 non-inverting input. |
| | DAC4OUT1 | — | AN | DAC4 voltage output. |
| | T0CKI(1) | TTL/ST | — | Timer0 clock input. |
| | IOCA4 | | | |
| RA5/AN4/OPA1IN0-/ DAC2OUT1/IOCA5 | RA5 | TTL/ST | CMOS | General purpose I/O. |
| | AN4 | AN | — | ADC Channel 4 input. |
| | OPA1IN0- | AN | — | Operational amplifier 1 inverting input. |
| | DAC2OUT1 | — | AN | DAC2 voltage output. |
| | IOCA5 | | | |
| RA6/CLKOUT/C6IN1+/OSC2/ IOCA6 | RA6 | TTL/ST | CMOS | General purpose I/O. |
| | CLKOUT | — | CMOS | FOSC/4 output. |
| | C6IN1+ | AN | — | Comparator C6 positive input. |
| | OSC2 | XTAL | — | Crystal/Resonator (LP, XT, HS modes). |
| | IOCA6 | | | |
| RA7/CLKIN/OSC1/IOCA7 | RA7 | TTL/ST | CMOS | General purpose I/O. |
| | CLKIN | TTL/ST | — | CLC input. |
| | OSC1 | XTAL | — | Crystal/Resonator (LP, XT, HS modes). |
| | IOCA7 | | | |
| RB0/AN12/ZCD/HIB0/C2IN1+/ IOCB0 | RB0 | TTL/ST | CMOS | General purpose I/O. |
| | AN12 | AN | — | ADC Channel 12 input. |
| | ZCD | AN | — | Zero cross detection input |
| | HIB0 | | | |
| | C2IN1+ | AN | — | Comparator C2 positive input. |
| | IOCB0 | | | |

**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 $I^2C$
HV = High Voltage XTAL = Crystal levels

**Note 1:** Default peripheral input. Alternate pins can be selected as the peripheral input with the PPS input selection registers.

**2:** All pin digital outputs default to PORT latch data. Alternate outputs can be selected as the peripheral digital output with the PPS output selection registers.

**3:** These peripheral functions are bidirectional. The output pin selections must be the same as the input pin selections.

**TABLE 1-2: PIC16(L)F1773/6 PINOUT DESCRIPTION (CONTINUED)**

| Name | Function | Input Type | Output Type | Description |
|---|---|---|---|---|
| RB1/AN10/PRG1IN1/PRG2IN0/ HIB1/C1IN3-/C2IN3-/C3IN3-/ C4IN3-/OPA2OUT/OPA1IN1+/ OPA1IN1-/IOCB1 | RB1 | TTL/ST | CMOS | General purpose I/O. |
| | AN10 | AN | — | ADC Channel 10 input. |
| | PRG1IN1 | AN | — | Ramp generator 1 reference voltage input. |
| | PRG2IN0 | AN | — | Ramp generator 2 reference voltage input. |
| | HIB1 | | | |
| | C1IN3- | AN | — | Comparator 1 negative input. |
| | C2IN3- | AN | — | Comparator 2 negative input. |
| | C3IN3- | AN | — | Comparator 3 negative input. |
| | C4IN3- | AN | — | Comparator 4 negative input. |
| | OPA2OUT | — | AN | Operational amplifier 2 output. |
| | OPA1IN1+ | AN | — | Operational amplifier 1 non-inverting input. |
| | OPA1IN1- | AN | — | Operational amplifier 1 inverting input. |
| | IOCB1 | | | |
| RB2/AN8/OPA2IN0-/ DAC3OUT1/IOCB2 | RB2 | TTL/ST | CMOS | General purpose I/O. |
| | AN8 | AN | — | ADC Channel 8 input. |
| | OPA2IN0- | AN | — | Operational amplifier 2 inverting input. |
| | DAC3OUT1 | — | AN | DAC3 voltage output. |
| | IOCB2 | | | |
| RB3/AN9/C1IN2-/C2IN2-/ C3IN2-/OPA2IN0+/IOCB3 | RB3 | TTL/ST | CMOS | General purpose I/O. |
| | AN9 | AN | — | ADC Channel 9 input. |
| | C1IN2- | AN | — | Comparator 1 negative input. |
| | C2IN2- | AN | — | Comparator 2 negative input. |
| | C3IN2- | AN | — | Comparator 3 negative input. |
| | OPA2IN0+ | AN | | Operational amplifier 2 non-inverting input. |
| | IOCB3 | | | |
| RB4/AN11/C3IN1+/T5G/IOCB4 | RB4 | TTL/ST | CMOS | General purpose I/O. |
| | AN11 | AN | — | ADC Channel 11 input. |
| | C3IN1+ | AN | — | Comparator C3 positive input. |
| | T5G(1) | TTL/ST | — | Timer5 gate input. |
| | IOCB4 | | | |
| RB5/AN13/DAC5REF1-/ DAC7REF1-/C4IN2-/T1G/IOCB5 | RB5 | TTL/ST | CMOS | General purpose I/O. |
| | AN13 | AN | — | ADC Channel 11 input. |
| | DAC5REF1- | AN | — | DAC5 negative reference. |
| | DAC7REF1- | AN | — | DAC7 negative reference. |
| | C4IN2- | AN | — | Comparator 4 negative input. |
| | T1G(1) | TTL/ST | — | Timer1 gate input. |
| | IOCB5 | | | |

**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 $I^2C$
HV = High Voltage XTAL = Crystal levels

**Note 1:** Default peripheral input. Alternate pins can be selected as the peripheral input with the PPS input selection registers.
**2:** All pin digital outputs default to PORT latch data. Alternate outputs can be selected as the peripheral digital output with the PPS output selection registers.
**3:** These peripheral functions are bidirectional. The output pin selections must be the same as the input pin selections.

**TABLE 1-2: PIC16(L)F1773/6 PINOUT DESCRIPTION (CONTINUED)**

| Name | Function | Input Type | Output Type | Description |
|---|---|---|---|---|
| RB6/DAC5REF1+/DAC7REF1+/ C4IN1+/ICSPCLK/IOCB6 | RB6 | TTL/ST | CMOS | General purpose I/O. |
| | DAC5REF1+ | AN | — | DAC5 positive reference. |
| | DAC7REF1+ | AN | — | DAC7 positive reference. |
| | C4IN1+ | AN | — | Comparator C2 positive input. |
| | ICSPCLK | ST | — | Serial Programming Clock. |
| | IOCB6 | | | |
| RB7/C5IN1+/DAC1OUT2/ DAC2OUT2/DAC3OUT2/ DAC4OUT2/DAC5OUT2/ DAC7OUT2/T6IN/ ICSPDAT/IOCB7 | RB7 | TTL/ST | CMOS | General purpose I/O. |
| | C5IN1+ | AN | — | Comparator C5 positive input. |
| | DAC1OUT2 | — | AN | DAC1 voltage output. |
| | DAC2OUT2 | — | AN | DAC2 voltage output. |
| | DAC3OUT2 | — | AN | DAC3 voltage output. |
| | DAC4OUT2 | — | AN | DAC4 voltage output. |
| | DAC5OUT2 | — | AN | DAC5 voltage output. |
| | DAC7OUT2 | — | AN | DAC7 voltage output. |
| | T6IN(1) | TTL/ST | — | Timer6 gate input. |
| | ICSPDAT | ST | CMOS | ICSP™ Data I/O. |
| | IOCB7 | | | |
| RC0/DAC5OUT1/T1CKI/T3CKI/ T3G/IOCC0 | RC0 | TTL/ST | CMOS | General purpose I/O. |
| | DAC5OUT1 | — | AN | DAC5 voltage output. |
| | T1CKI(1) | AN | — | Comparator 4 negative input. |
| | T3CKI(1) | TTL/ST | — | Timer3 clock input. |
| | T3G(1) | TTL/ST | — | Timer3 gate input. |
| | IOCC0 | | | |
| RC1/DAC7OUT1/IOCC | RC1 | TTL/ST | CMOS | General purpose I/O. |
| | DAC7OUT1 | — | AN | DAC7 voltage output. |
| | IOCC1 | | | |
| RC2/AN14/C5IN2-/C6IN2-/ T5CKI/IOCC2 | RC2 | TTL/ST | CMOS | General purpose I/O. |
| | AN14 | AN | — | ADC Channel 14 input. |
| | C5IN2- | AN | — | Comparator 5 negative input. |
| | C6IN2- | AN | — | Comparator 6 negative input. |
| | T5CKI(1) | TTL/ST | — | Timer5 clock input. |
| | IOCC2 | | | |

**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 $I^2C$
HV = High Voltage XTAL = Crystal levels

**Note 1:** Default peripheral input. Alternate pins can be selected as the peripheral input with the PPS input selection registers.

**2:** All pin digital outputs default to PORT latch data. Alternate outputs can be selected as the peripheral digital output with the PPS output selection registers.

**3:** These peripheral functions are bidirectional. The output pin selections must be the same as the input pin selections.

**TABLE 1-2: PIC16(L)F1773/6 PINOUT DESCRIPTION (CONTINUED)**

| Name | Function | Input Type | Output Type | Description |
|---|---|---|---|---|
| RC3/AN15/C1IN4-/C2IN4-/C3IN4-/C4IN4-/C5IN4-/C6IN4-/T2IN/IOCC3 | RC3 | TTL/ST | CMOS | General purpose I/O. |
| | AN15 | AN | — | ADC Channel 15 input. |
| | C1IN4- | AN | — | Comparator 1 negative input. |
| | C2IN4- | AN | — | Comparator 2 negative input. |
| | C3IN4- | AN | — | Comparator 3 negative input. |
| | C4IN4- | AN | — | Comparator 4 negative input. |
| | C5IN4- | AN | — | Comparator 5 negative input. |
| | C6IN4- | AN | — | Comparator 6 negative input. |
| | T2IN(1) | TTL/ST | — | Timer2 gate input. |
| | IOCC3 | | | |
| RC4/AN16/C5IN3-/C6IN3-/T8IN/IOCC4 | RC4 | TTL/ST | CMOS | General purpose I/O. |
| | AN16 | AN | — | ADC Channel 16 input. |
| | C5IN3- | AN | — | Comparator 5 negative input. |
| | C6IN3- | AN | — | Comparator 6 negative input. |
| | T8IN(1) | TTL/ST | — | Timer8 gate input. |
| | IOCC4 | | | |
| RC5/AN17/OPA3IN0+/T4IN/IOCC5 | RC5 | TTL/ST | CMOS | General purpose I/O. |
| | AN17 | AN | — | ADC Channel 17 input. |
| | OPA3IN0+ | AN | — | Operational amplifier 3 non-inverting input. |
| | T4IN(1) | TTL/ST | — | Timer4 gate input. |
| | IOCC5 | | | |
| RC6/AN18/PRG3IN0/C5IN1-/C6IN1-/OPA3OUT/IOCC6 | RC6 | TTL/ST | CMOS | General purpose I/O. |
| | AN18 | AN | — | ADC Channel 18 input. |
| | PRG3IN0 | AN | — | Ramp generator 3 reference voltage input. |
| | C5IN1- | AN | — | Comparator 5 negative input. |
| | C6IN1- | AN | — | Comparator 5 negative input. |
| | OPA3OUT | — | AN | Operational amplifier 3 output. |
| | IOCC6 | | | |
| RC7/AN19/OPA3IN0-/IOCC7 | RC7 | TTL/ST | CMOS | General purpose I/O. |
| | AN19 | AN | — | ADC Channel 19 input. |
| | OPA3IN0- | AN | — | Operational amplifier 3 inverting input. |
| | IOCC7 | | | |
| RE3/$\overline{\text{MCLR}}$/IOCE3 | RE3 | TTL/ST | CMOS | General purpose input. |
| | $\overline{\text{MCLR}}$ | ST | — | Master clear input. |
| | IOCE3 | | | |
| $V_{DD}$ | $V_{DD}$ | 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 $I^2C$
HV = High Voltage XTAL = Crystal levels

**Note 1:** Default peripheral input. Alternate pins can be selected as the peripheral input with the PPS input selection registers.

**2:** All pin digital outputs default to PORT latch data. Alternate outputs can be selected as the peripheral digital output with the PPS output selection registers.

**3:** These peripheral functions are bidirectional. The output pin selections must be the same as the input pin selections.

**TABLE 1-2: PIC16(L)F1773/6 PINOUT DESCRIPTION (CONTINUED)**

| Name | Function | Input Type | Output Type | Description |
|---|---|---|---|---|
| OUT(2) | C1OUT | | CMOS | Comparator 1 output. |
| | C2OUT | | CMOS | Comparator 2 output. |
| | C3OUT | | CMOS | Comparator 3 output. |
| | C4OUT | | CMOS | Comparator 4 output. |
| | C5OUT | | CMOS | Comparator 5 output. |
| | C6OUT | | CMOS | Comparator 6 output. |
| | CCP1 | | CMOS | Compare/PWM1 output. |
| | CCP2 | | CMOS | Compare/PWM2 output. |
| | CCP7 | | CMOS | Compare/PWM7 output. |
| | MD1OUT | | CMOS | Data signal modulator 1 output. |
| | MD2OUT | | CMOS | Data signal modulator 2 output. |
| | MD3OUT | | CMOS | Data signal modulator 3 output. |
| | PWM3OUT | | CMOS | PWM3 output. |
| | PWM4OUT | | CMOS | PWM4 output. |
| | PWM5OUT | | CMOS | PWM5 output. |
| | PWM6OUT | | CMOS | PWM6 output. |
| | PWM9OUT | | CMOS | PWM9 output. |
| | PWM11OUT | | CMOS | PWM11 output. |
| | COG1A | | CMOS | Complementary output generator 1 output A. |
| | COG1B | | CMOS | Complementary output generator 1 output B. |
| | COG1C | | CMOS | Complementary output generator 1 output C. |
| | COG1D | | CMOS | Complementary output generator 1 output D. |
| | COG2A | | CMOS | Complementary output generator 2 output A. |
| | COG2B | | CMOS | Complementary output generator 2 output B. |
| | COG2C | | CMOS | Complementary output generator 2 output C. |
| | COG2D | | CMOS | Complementary output generator 2 output D. |
| | COG3A | | CMOS | Complementary output generator 3 output A. |
| | COG3B | | CMOS | Complementary output generator 3 output B. |
| | COG3C | | CMOS | Complementary output generator 3 output C. |
| | COG3D | | CMOS | Complementary output generator 3 output D. |
| | SDA | | OD | $I^2C$ data output. |
| | SCK | | CMOS | SPI clock output. |
| | SCL(3) | | OD | $I^2C$ clock output. |
| | SDO | | CMOS | SPI data output. |
| | TX | | CMOS | EUSART asynchronous TX data out. |
| | CK | | CMOS | EUSART synchronous clock out. |
| | DT(3) | | CMOS | EUSART synchronous data output. |
| | CLC1OUT | | CMOS | Configurable logic cell 1 output. |
| | CLC2OUT | | CMOS | Configurable logic cell 2 output. |
| | CLC3OUT | | CMOS | Configurable logic cell 3 output. |
| | CLC4OUT | | CMOS | Configurable logic cell 4 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 $I^2C$
HV = High Voltage XTAL = Crystal levels

**Note 1:** Default peripheral input. Alternate pins can be selected as the peripheral input with the PPS input selection registers.

**2:** All pin digital outputs default to PORT latch data. Alternate outputs can be selected as the peripheral digital output with the PPS output selection registers.

**3:** These peripheral functions are bidirectional. The output pin selections must be the same as the input pin selections.

## 1.2 Peripheral Connection Matrix

Input selection multiplexers on many of the peripherals enable selecting the output of another peripheral such that the signal path is contained entirely within the device. Although the peripheral output can also be routed to a pin, with the PPS selection feature, it is not necessary to do so. Table 1-3 shows all the possible inter-peripheral signal connections. Please refer to corresponding peripheral section to obtain the multiplexer selection codes for the desired connection.

**TABLE 1-3: PERIPHERAL CONNECTION MATRIX**

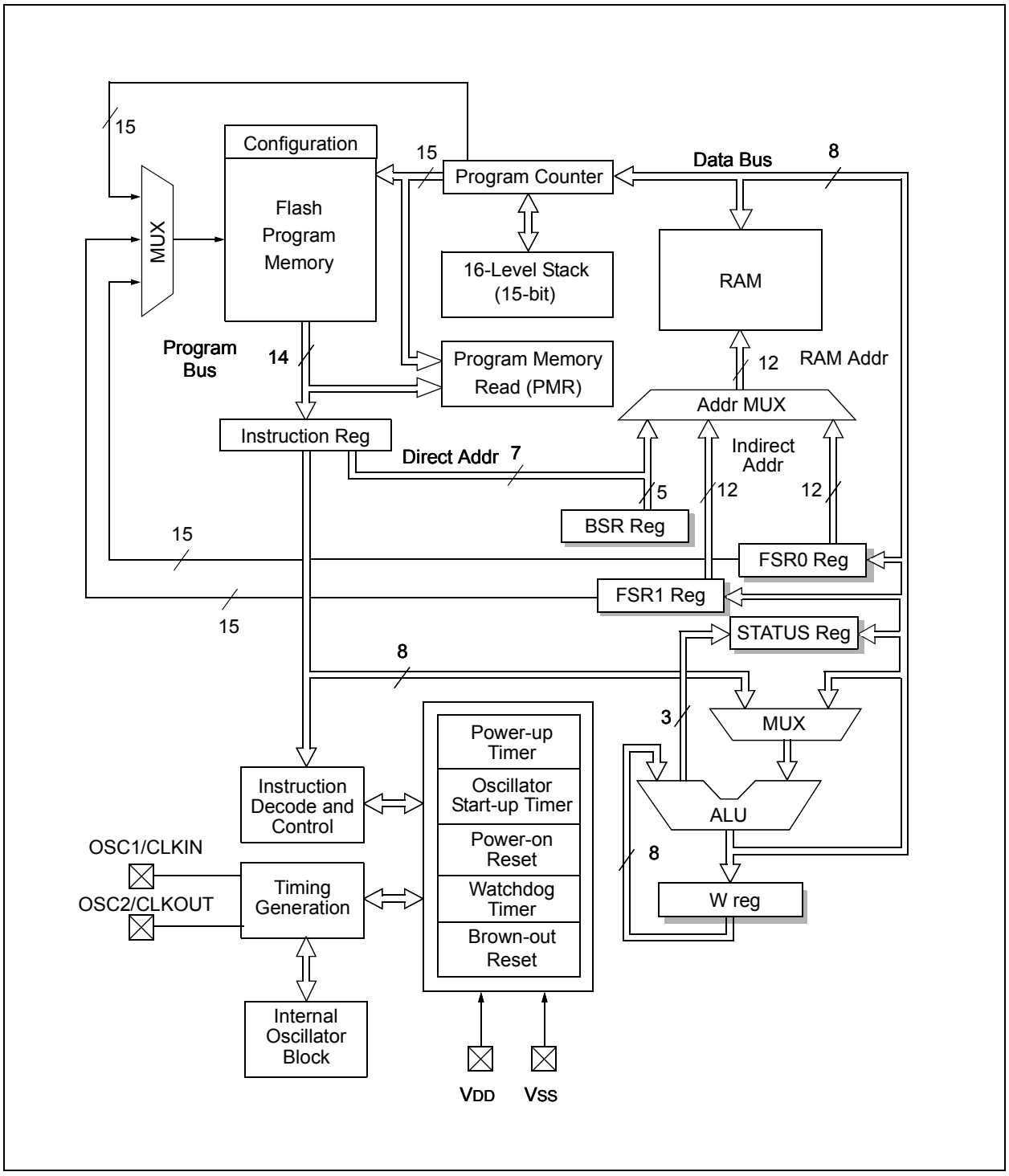
| | Peripheral Input | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Peripheral Output** | **ADC Trigger** | **COG Clock** | **COG Rising/Falling** | **COG Shutdown** | **10-bit DAC** | **5-bit DAC** | **PRG Analog Input** | **PRG Rising/Falling** | **Comparator +** | **Comparator -** | **CLC** | **DSM CH** | **DSM CL** | **DSM Mod** | **Op Amp +** | **Op Amp -** | **Op Amp Override** | **10-bit PWM** | **16-bit PWM** | **CCP Capture** | **CCP Clock** | **Timer2/4/6 Clock** | **Timer2/4/6 Reset** | **Timer1/3/5 Gate** | **Timer0 Clock** |
| FVR | | | | | ● | ● | ● | | ● | ● | | | | | ● | ● | | | | | | | | | |
| ZCD | | | | | | | | | | | ● | | | | | | ● | | | | | ● | | | |
| PRG | | | | | | | | | ● | | | | | | ● | ● | | | | | | | | | |
| 10-bit DAC | | | | | | | ● | | ● | | | | | | ● | ● | | | | | | | | | |
| 5-bit DAC | | | | | | | ● | | ● | | | | | | ● | ● | | | | | | | | | |
| CCP | ● | | ● | | | | | ● | | | ● | ● | ● | ● | | | ● | | | | | | ● | | |
| Comparator (sync) | ● | | | | | | | ● | | | ● | | | | | | ● | | | ● | | | ● | ● | |
| Comparator (async) | | | ● | ● | | | | | | | | | | ● | | | | | | | | | | | |
| CLC | ● | | ● | ● | | | | | | | ● | ● | ● | ● | | | ● | | | ● | | ● | ● | | |
| DSM | | | | | | | | | | | | | | | | | | | | | | | | | |
| COG | | | | | | | | | | | | | | | | | ● | | | | | | | | |
| EUSART TX/CK | | | | | | | | | | | ● | | | ● | | | | | | | | | | | |
| EUSART DT | | | | | | | | | | | ● | | | ● | | | | | | | | | | | |
| MSSP SCK/SCL | | | | | | | | | | | ● | | | ● | | | | | | | | | | | |
| MSSP SDO/SDA | | | | | | | | | | | ● | | | ● | | | | | | | | | | | |
| Op Amp | | | | | | | ● | | | | | | | | | | | | | | | | | | |
| 10-bit PWM | ● | | ● | | | | | ● | | | ● | ● | ● | ● | | | ● | | | | | | ● | | |
| 16-bit PWM | ● | | ● | | | | | ● | | | ● | ● | ● | ● | | | ● | | | | | | ● | | |
| Timer0 overflow | ● | | | | | | | | | | ● | | | | | | | | | | | | | ● | |
| Timer2 = T2PR | | | | ● | | | | | | | ● | | | | | | | ● | | | ● | | ● | | |
| Timer4 = T4PR | | | | ● | | | | | | | ● | | | | | | | ● | | | ● | | ● | | |
| Timer6 = T6PR | | | | ● | | | | | | | ● | | | | | | | ● | | | ● | | ● | | |
| Timer2 Postscale | ● | | | ● | | | | | | | ● | | | | | | | ● | | | ● | | ● | | |
| Timer4 Postscale | ● | | | ● | | | | | | | ● | | | | | | | ● | | | ● | | ● | | |
| Timer6 Postscale | ● | | | ● | | | | | | | ● | | | | | | | ● | | | ● | | ● | | |
| Timer1 overflow | ● | | | | | | | | | | ● | | | | | | | ● | | | ● | | | | |
| Timer3 overflow | ● | | | | | | | | | | ● | | | | | | | ● | | | ● | | | | |
| Timer5 overflow | ● | | | | | | | | | | ● | | | | | | | ● | | | ● | | | | |
| SOSC | | | | | | | | | | | | | | | | | | | ● | | | ● | | | |
| Fosc/4 | | ● | | | | | | | | | | | | | | | | | | | | ● | | | |
| Fosc | | ● | | | | | | | | | ● | ● | ● | | | | | | ● | | | ● | | | |
| HFINTOSC | | ● | | | | | | | | | ● | ● | ● | | | | | | ● | | | ● | | | |
| LFINTOSC | | | | | | | | | | | ● | | | | | | | | ● | | | ● | | | |
| MFINTOSC | | | | | | | | | | | | | | | | | | | | | | ● | | | |
| IOCIF | | | | | | | | | | | ● | | | | | | | | | ● | ● | | | | |
| PPS Input pin | | | ● | ● | | | | ● | | | | ● | ● | ● | | | | | | ● | ● | ● | ● | ● | ● |

## 2.0 ENHANCED MID-RANGE CPU

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

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

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



## 2.1 Automatic Interrupt Context Saving

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

## 2.2 16-Level Stack with Overflow and Underflow

These devices have a hardware stack memory 15 bits wide and 16 words deep. A Stack Overflow or Underflow will set the appropriate bit (STKOVF or STKUNF) in the PCON register, and if enabled, will cause a software Reset. See **Section 3.6 "Stack"** for more details.

## 2.3 File Select Registers

There are two 16-bit File Select Registers (FSR). FSRs can access all file registers and program memory, which allows one Data Pointer for all memory. When an FSR points to program memory, there is one additional instruction cycle in instructions using INDF to allow the data to be fetched. General purpose memory can now also be addressed linearly, providing the ability to access contiguous data larger than 80 bytes. There are also new instructions to support the FSRs. See **Section 3.7 "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 35.0 "Instruction Set Summary"** for more details.

## 3.0 MEMORY ORGANIZATION

These devices contain the following types of memory:

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

**Note 1:** The method to access Flash memory through the PMCON registers is described in **Section 10.0 "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)F1773/6 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).

### 3.2 High-Endurance Flash

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

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

| Device | Program Memory Space (Words) | Last Program Memory Address | High-Endurance Flash Memory Address Range(1) |
|---|---|---|---|
| PIC16(L)F1773 | 4,096 | 0FFFh | 0F80h-0FFFh |
| PIC16(L)F1776 | 8,192 | 1FFFh | 1F80h-1FFFh |

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

**FIGURE 3-1: PROGRAM MEMORY MAP AND STACK FOR PIC16(L)F1773**

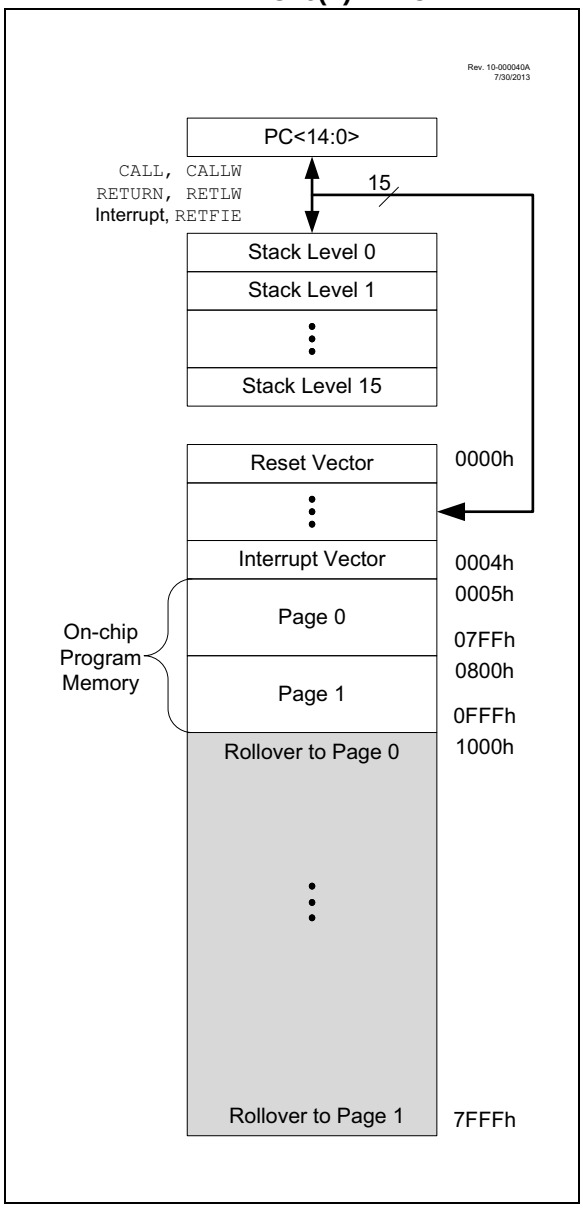
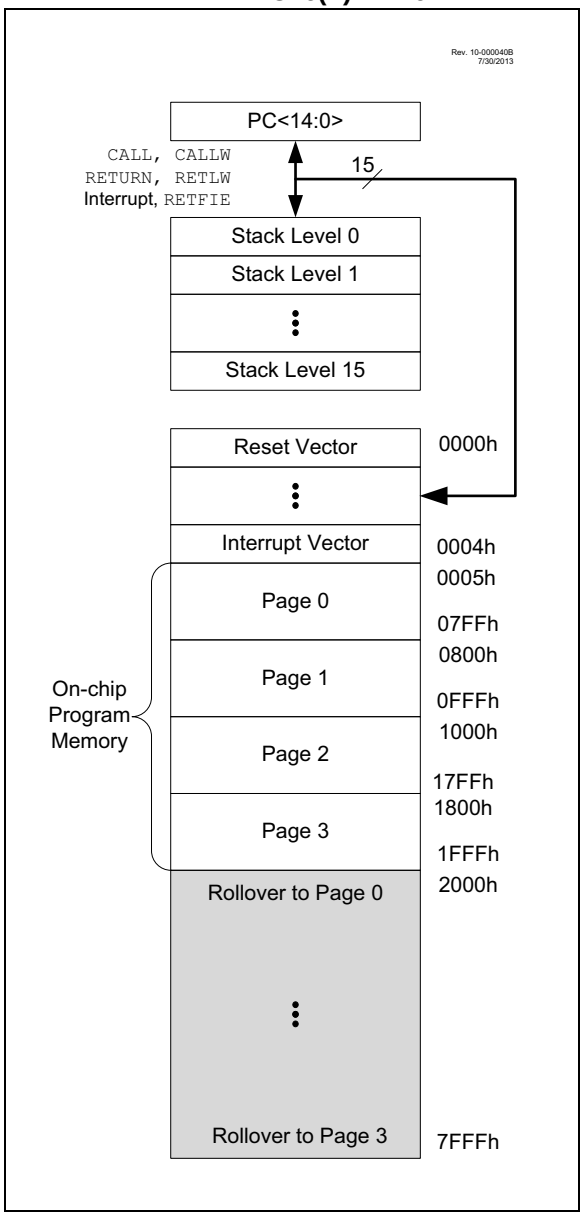



**FIGURE 3-2: PROGRAM MEMORY MAP AND STACK FOR PIC16(L)F1776**



### 3.2.1 READING PROGRAM MEMORY AS DATA

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

#### 3.2.1.1 RETLW Instruction

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

**EXAMPLE 3-1: RETLW INSTRUCTION**

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


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

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

#### 3.2.1.2 Indirect Read with FSR

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

The high 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 constants;MSb sets
                        automatically
  MOVWF  FSR1H
  BTFSC  STATUS, C      ;carry from ADDLW?
  INCF   FSR1H, f       ;yes
  MOVIW  0[FSR1]
;THE PROGRAM MEMORY IS IN W
```

## 3.3 Data Memory Organization

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

- 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 3.7 "Indirect Addressing"** for more information.

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

### 3.3.1 CORE REGISTERS

The core registers contain the registers that directly affect the basic operation. The core registers occupy the first 12 addresses of every data memory bank (addresses x00h/x08h through x0Bh/x8Bh). These registers are listed below in Table 3-2. For detailed information, see Table 3-11.

**TABLE 3-2: CORE REGISTERS**

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

#### 3.3.1.1 STATUS Register

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

- the arithmetic status of the ALU
- the Reset status

The STATUS register can be the destination for any instruction, like any other register. If the STATUS register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. These bits are set or cleared according to the device logic. Furthermore, the $\overline{TO}$ 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 35.0 "Instruction Set Summary"**).

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

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

bit 0 **C:** Carry/$\overline{\text{Borrow}}$ 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 $\overline{\text{Borrow}}$, the polarity is reversed. A subtraction is executed by adding the two's complement of the second operand.

### 3.4.1 SPECIAL FUNCTION REGISTER

The Special Function Registers (SFR) are registers used by the application to control the desired operation of peripheral functions in the device. The SFR occupies 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 each peripheral are described in the corresponding peripheral chapters of this data sheet.

### 3.4.2 GENERAL PURPOSE RAM

There are up to 80 bytes of General Purpose Registers (GPR) in each data memory bank. The GPR occupies the space immediately after the SFR of selected data memory banks. The number of banks selected depends on the total amount of GPR space available in the device.

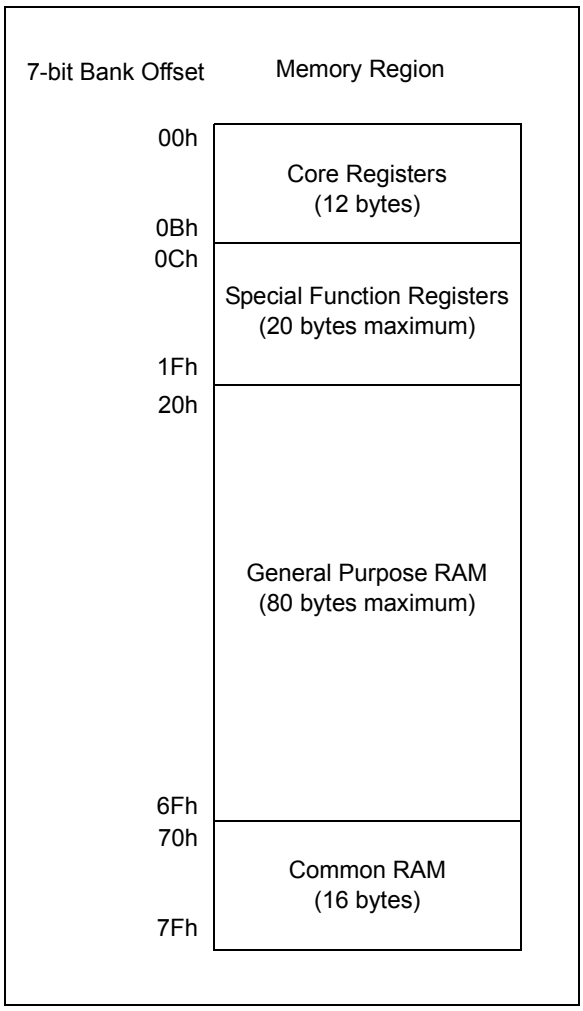
#### 3.4.2.1 Linear Access to GPR

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 3.7.2 "Linear Data Memory"** for more information.

### 3.4.3 COMMON RAM

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

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




### 3.4.4 DEVICE MEMORY MAPS

The memory maps for the device family are as shown in Tables 3-3 through 3-10.

## TABLE 3-3: PIC16(L)F1773 MEMORY MAP (BANKS 0-7)

| BANK 0 | | BANK 1 | | BANK 2 | | BANK 3 | | BANK 4 | | BANK 5 | | BANK 6 | | BANK 7 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 000h | Core Registers (Table 3-2) | 080h | Core Registers (Table 3-2) | 100h | Core Registers (Table 3-2) | 180h | Core Registers (Table 3-2) | 200h | Core Registers (Table 3-2) | 280h | Core Registers (Table 3-2) | 300h | Core Registers (Table 3-2) | 380h | Core Registers (Table 3-2) |
| 00Bh | | 08Bh | | 10Bh | | 18Bh | | 20Bh | | 28Bh | | 30Bh | | 38Bh | |
| 00Ch | PORTA | 08Ch | TRISA | 10Ch | LATA | 18Ch | ANSELA | 20Ch | WPUA | 28Ch | ODCONA | 30Ch | SLRCONA | 38Ch | INLVLA |
| 00Dh | PORTB | 08Dh | TRISB | 10Dh | LATB | 18Dh | ANSELB | 20Dh | WPUB | 28Dh | ODCONB | 30Dh | SLRCONB | 38Dh | INLVLB |
| 00Eh | PORTC | 08Eh | TRISC | 10Eh | LATC | 18Eh | ANSELC | 20Eh | WPUC | 28Eh | ODCONC | 30Eh | SLRCONC | 38Eh | INLVLC |
| 00Fh | — | 08Fh | — | 10Fh | — | 18Fh | — | 20Fh | — | 28Fh | — | 30Fh | — | 38Fh | — |
| 010h | PORTE | 090h | TRISE | 110h | — | 190h | — | 210h | WPUE | 290h | — | 310h | — | 390h | INLVLE |
| 011h | PIR1 | 091h | PIE1 | 111h | CMOUT | 191h | PMADRL | 211h | SSP1BUF | 291h | CCPR1L | 311h | — | 391h | IOCAP |
| 012h | PIR2 | 092h | PIE2 | 112h | CM1CON0 | 192h | PMADRH | 212h | SSP1ADD | 292h | CCPR1H | 312h | — | 392h | IOCAN |
| 013h | PIR3 | 093h | PIE3 | 113h | CM1CON1 | 193h | PMDATL | 213h | SSP1MSK | 293h | CCP1CON | 313h | — | 393h | IOCAF |
| 014h | PIR4 | 094h | PIE4 | 114h | CM1NSEL | 194h | PMDATH | 214h | SSP1STAT | 294h | CCP1CAP | 314h | — | 394h | IOCBP |
| 015h | PIR5 | 095h | PIE5 | 115h | CM1PSEL | 195h | PMCON1 | 215h | SSP1CON1 | 295h | CCPR2L | 315h | MD1CON0 | 395h | IOCBN |
| 016h | PIR6 | 096h | PIE6 | 116h | CM2CON0 | 196h | PMCON2 | 216h | SSP1CON2 | 296h | CCPR2H | 316h | MD1CON1 | 396h | IOCBF |
| 017h | TMR0 | 097h | OPTION_REG | 117h | CM2CON1 | 197h | VREGCON(1) | 217h | SSP1CON3 | 297h | CCP2CON | 317h | MD1SRC | 397h | IOCCP |
| 018h | TMR1L | 098h | PCON | 118h | CM2NSEL | 198h | — | 218h | — | 298h | CCP2CAP | 318h | MD1CARL | 398h | IOCCN |
| 019h | TMR1H | 099h | WDTCON | 119h | CM2PSEL | 199h | RC1REG | 219h | — | 299h | CCPR7L | 319h | MD1CARH | 399h | IOCCF |
| 01Ah | T1CON | 09Ah | OSCTUNE | 11Ah | CM3CON0 | 19Ah | TX1REG | 21Ah | — | 29Ah | CCPR7H | 31Ah | — | 39Ah | — |
| 01Bh | T1GCON | 09Bh | OSCCON | 11Bh | CM3CON1 | 19Bh | SP1BRGL | 21Bh | MD3CON0 | 29Bh | CCP7CON | 31Bh | MD2CON0 | 39Bh | — |
| 01Ch | TMR3L | 09Ch | OSCSTAT | 11Ch | CM3NSEL | 19Ch | SP1BRGH | 21Ch | MD3CON1 | 29Ch | CCP7CAP | 31Ch | MD2CON1 | 39Ch | — |
| 01Dh | TMR3H | 09Dh | BORCON | 11Dh | CM3PSEL | 19Dh | RC1STA | 21Dh | MD3SRC | 29Dh | — | 31Dh | MD2SRC | 39Dh | IOCEP |
| 01Eh | T3CON | 09Eh | FVRCON | 11Eh | — | 19Eh | TX1STA | 21Eh | MD3CARL | 29Eh | CCPTMRS1 | 31Eh | MD2CARL | 39Eh | IOCEN |
| 01Fh | T3GCON | 09Fh | ZCD1CON | 11Fh | — | 19Fh | BAUD1CON | 21Fh | MD3CARH | 29Fh | CCPTMRS2 | 31Fh | MD2CARH | 39Fh | IOCEF |
| 020h | General Purpose Register 80 Bytes | 0A0h | General Purpose Register 80 Bytes | 120h | General Purpose Register 80 Bytes | 1A0h | General Purpose Register 80 Bytes | 220h | General Purpose Register 80 Bytes | 2A0h | General Purpose Register 80 Bytes | 320h | General Purpose Register 16 Bytes | 3A0h | Unimplemented Read as '0' |
| 06Fh | | 0EFh | | 16Fh | | 1EFh | | 26Fh | | 2EFh | | 32Fh | | | |
| 070h | Common RAM 70h – 7Fh | 0F0h | Accesses 70h – 7Fh | 170h | Accesses 70h – 7Fh | 1F0h | Accesses 70h – 7Fh | 270h | Accesses 70h – 7Fh | 2F0h | Accesses 70h – 7Fh | 330h | Unimplemented Read as '0' | | |
| 07Fh | | 0FFh | | 17Fh | | 1FFh | | 27Fh | | 2FFh | | 36Fh | | 3EFh | |
| | | | | | | | | | | | | 370h | Accesses 70h – 7Fh | 3F0h | Accesses 70h – 7Fh |
| | | | | | | | | | | | | 37Fh | | 3FFh | |

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

**Note 1:** Unimplemented on PIC16LF1773.

## TABLE 3-4: PIC16(L)F1776 MEMORY MAP (BANKS 0-7)

| BANK 0 | | BANK 1 | | BANK 2 | | BANK 3 | | BANK 4 | | BANK 5 | | BANK 6 | | BANK 7 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 000h | Core Registers (Table 3-2) | 080h | Core Registers (Table 3-2) | 100h | Core Registers (Table 3-2) | 180h | Core Registers (Table 3-2) | 200h | Core Registers (Table 3-2) | 280h | Core Registers (Table 3-2) | 300h | Core Registers (Table 3-2) | 380h | Core Registers (Table 3-2) |
| 00Bh | | 08Bh | | 10Bh | | 18Bh | | 20Bh | | 28Bh | | 30Bh | | 38Bh | |
| 00Ch | PORTA | 08Ch | TRISA | 10Ch | LATA | 18Ch | ANSELA | 20Ch | WPUA | 28Ch | ODCONA | 30Ch | SLRCONA | 38Ch | INLVLA |
| 00Dh | PORTB | 08Dh | TRISB | 10Dh | LATB | 18Dh | ANSELB | 20Dh | WPUB | 28Dh | ODCONB | 30Dh | SLRCONB | 38Dh | INLVLB |
| 00Eh | PORTC | 08Eh | TRISC | 10Eh | LATC | 18Eh | ANSELC | 20Eh | WPUC | 28Eh | ODCONC | 30Eh | SLRCONC | 38Eh | INLVLC |
| 00Fh | — | 08Fh | — | 10Fh | — | 18Fh | — | 20Fh | — | 28Fh | — | 30Fh | — | 38Fh | — |
| 010h | PORTE | 090h | TRISE | 110h | — | 190h | — | 210h | WPUE | 290h | — | 310h | — | 390h | INLVLE |
| 011h | PIR1 | 091h | PIE1 | 111h | CMOUT | 191h | PMADRL | 211h | SSP1BUF | 291h | CCPR1L | 311h | — | 391h | IOCAP |
| 012h | PIR2 | 092h | PIE2 | 112h | CM1CON0 | 192h | PMADRH | 212h | SSP1ADD | 292h | CCPR1H | 312h | — | 392h | IOCAN |
| 013h | PIR3 | 093h | PIE3 | 113h | CM1CON1 | 193h | PMDATL | 213h | SSP1MSK | 293h | CCP1CON | 313h | — | 393h | IOCAF |
| 014h | PIR4 | 094h | PIE4 | 114h | CM1NSEL | 194h | PMDATH | 214h | SSP1STAT | 294h | CCP1CAP | 314h | — | 394h | IOCBP |
| 015h | PIR5 | 095h | PIE5 | 115h | CM1PSEL | 195h | PMCON1 | 215h | SSP1CON1 | 295h | CCPR2L | 315h | MD1CON0 | 395h | IOCBN |
| 016h | PIR6 | 096h | PIE6 | 116h | CM2CON0 | 196h | PMCON2 | 216h | SSP1CON2 | 296h | CCPR2H | 316h | MD1CON1 | 396h | IOCBF |
| 017h | TMR0 | 097h | OPTION_REG | 117h | CM2CON1 | 197h | VREGCON(1) | 217h | SSP1CON3 | 297h | CCP2CON | 317h | MD1SRC | 397h | IOCCP |
| 018h | TMR1L | 098h | PCON | 118h | CM2NSEL | 198h | — | 218h | — | 298h | CCP2CAP | 318h | MD1CARL | 398h | IOCCN |
| 019h | TMR1H | 099h | WDTCON | 119h | CM2PSEL | 199h | RC1REG | 219h | — | 299h | CCPR7L | 319h | MD1CARH | 399h | IOCCF |
| 01Ah | T1CON | 09Ah | OSCTUNE | 11Ah | CM3CON0 | 19Ah | TX1REG | 21Ah | — | 29Ah | CCPR7H | 31Ah | — | 39Ah | — |
| 01Bh | T1GCON | 09Bh | OSCCON | 11Bh | CM3CON1 | 19Bh | SP1BRGL | 21Bh | MD3CON0 | 29Bh | CCP7CON | 31Bh | MD2CON0 | 39Bh | — |
| 01Ch | TMR3L | 09Ch | OSCSTAT | 11Ch | CM3NSEL | 19Ch | SP1BRGH | 21Ch | MD3CON1 | 29Ch | CCP7CAP | 31Ch | MD2CON1 | 39Ch | — |
| 01Dh | TMR3H | 09Dh | BORCON | 11Dh | CM3PSEL | 19Dh | RC1STA | 21Dh | MD3SRC | 29Dh | — | 31Dh | MD2SRC | 39Dh | IOCEP |
| 01Eh | T3CON | 09Eh | FVRCON | 11Eh | — | 19Eh | TX1STA | 21Eh | MD3CARL | 29Eh | CCPTMRS1 | 31Eh | MD2CARL | 39Eh | IOCEN |
| 01Fh | T3GCON | 09Fh | ZCD1CON | 11Fh | — | 19Fh | BAUD1CON | 21Fh | MD3CARH | 29Fh | CCPTMRS2 | 31Fh | MD2CARH | 39Fh | IOCEF |
| 020h | General Purpose Register 80 Bytes | 0A0h | General Purpose Register 80 Bytes | 120h | General Purpose Register 80 Bytes | 1A0h | General Purpose Register 80 Bytes | 220h | General Purpose Register 80 Bytes | 2A0h | General Purpose Register 80 Bytes | 320h | General Purpose Register 80 Bytes | 3A0h | General Purpose Register 80 Bytes |
| 06Fh | | 0EFh | | 16Fh | | 1EFh | | 26Fh | | 2EFh | | 36Fh | | 3EFh | |
| 070h | Common RAM 70h – 7Fh | 0F0h | Accesses 70h – 7Fh | 170h | Accesses 70h – 7Fh | 1F0h | Accesses 70h – 7Fh | 270h | Accesses 70h – 7Fh | 2F0h | Accesses 70h – 7Fh | 370h | Accesses 70h – 7Fh | 3F0h | Accesses 70h – 7Fh |
| 07Fh | | 0FFh | | 17Fh | | 1FFh | | 27Fh | | 2FFh | | 37Fh | | 3FFh | |

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

**Note 1:** Unimplemented on PIC16LF1776.

## TABLE 3-5: PIC16(L)F1773 MEMORY MAP, BANK 8-15

| BANK 8 | | BANK 9 | | BANK 10 | | BANK 11 | | BANK 12 | | BANK 13 | | BANK 14 | | BANK 15 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 400h | Core Registers (Table 3-2) | 480h | Core Registers (Table 3-2) | 500h | Core Registers (Table 3-2) | 580h | Core Registers (Table 3-2) | 600h | Core Registers (Table 3-2) | 680h | Core Registers (Table 3-2) | 700h | Core Registers (Table 3-2) | 780h | Core Registers (Table 3-2) |
| 40Bh | | 48Bh | | 50Bh | | 58Bh | | 60Bh | | 68Bh | | 70Bh | | 78Bh | |
| 40Ch | — | 48Ch | — | 50Ch | — | 58Ch | — | 60Ch | — | 68Ch | — | 70Ch | — | 78Ch | — |
| 40Dh | HIDRVB | 48Dh | — | 50Dh | — | 58Dh | DACLD | 60Dh | — | 68Dh | COG1PHR | 70Dh | COG2PHR | 78Dh | — |
| 40Eh | — | 48Eh | ADRESL | 50Eh | — | 58Eh | DAC1CON0 | 60Eh | — | 68Eh | COG1PHF | 70Eh | COG2PHF | 78Eh | PRG1RTSS |
| 40Fh | TMR5L | 48Fh | ADRESH | 50Fh | OPA1NCHS | 58Fh | DAC1REFL | 60Fh | — | 68Fh | COG1BLKR | 70Fh | COG2BLKR | 78Fh | PRG1FTSS |
| 410h | TMR5H | 490h | ADCON0 | 510h | OPA1PCHS | 590h | DAC1REFH | 610h | — | 690h | COG1BLKF | 710h | COG2BLKF | 790h | PRG1INS |
| 411h | T5CON | 491h | ADCON1 | 511h | OPA1CON | 591h | DAC2CON0 | 611h | — | 691h | COG1DBR | 711h | COG2DBR | 791h | PRG1CON0 |
| 412h | T5GCON | 492h | ADCON2 | 512h | OPA1ORS | 592h | DAC2REFL | 612h | — | 692h | COG1DBF | 712h | COG2DBF | 792h | PRG1CON1 |
| 413h | T4TMR | 493h | T2TMR | 513h | OPA2NCHS | 593h | DAC2REFH | 613h | — | 693h | COG1CON0 | 713h | COG2CON0 | 793h | PRG1CON2 |
| 414h | T4PR | 494h | T2PR | 514h | OPA2PCHS | 594h | DAC3CON0 | 614h | PWM3DCL | 694h | COG1CON1 | 714h | COG2CON1 | 794h | PRG2RTSS |
| 415h | T4CON | 495h | T2CON | 515h | OPA2CON | 595h | DAC3REF | 615h | PWM3DCH | 695h | COG1RIS0 | 715h | COG2RIS0 | 795h | PRG2FTSS |
| 416h | T4HLT | 496h | T2HLT | 516h | OPA2ORS | 596h | DAC4CON0 | 616h | PWM3CON | 696h | COG1RIS1 | 716h | COG2RIS1 | 796h | PRG2INS |
| 417h | T4CLKCON | 497h | T2CLKCON | 517h | OPA3NCHS | 597h | DAC4REF | 617h | PWM4DCL | 697h | COG1RSIM0 | 717h | COG2RSIM0 | 797h | PRG2CON0 |
| 418h | T4RST | 498h | T2RST | 518h | OPA3PCHS | 598h | DAC5CON0 | 618h | PWM4DCH | 698h | COG1RSIM1 | 718h | COG2RSIM1 | 798h | PRG2CON1 |
| 419h | — | 499h | — | 519h | OPA3CON | 599h | DAC5REFL | 619h | PWM4CON | 699h | COG1FIS0 | 719h | COG2FIS0 | 799h | PRG2CON2 |
| 41Ah | T6TMR | 49Ah | T8TMR | 51Ah | OPA3ORS | 59Ah | DAC5REFH | 61Ah | PWM9DCL | 69Ah | COG1FIS1 | 71Ah | COG2FIS1 | 79Ah | PRG3RTSS |
| 41Bh | T6PR | 49Bh | T8PR | 51Bh | — | 59Bh | — | 61Bh | PWM9DCH | 69Bh | COG1FSIM0 | 71Bh | COG2FSIM0 | 79Bh | PRG3FTSS |
| 41Ch | T6CON | 49Ch | T8CON | 51Ch | — | 59Ch | — | 61Ch | PWM9CON | 69Ch | COG1FSIM1 | 71Ch | COG2FSIM1 | 79Ch | PRG3INS |
| 41Dh | T6HLT | 49Dh | T8HLT | 51Dh | — | 59Dh | — | 61Dh | — | 69Dh | COG1ASD0 | 71Dh | COG2ASD0 | 79Dh | PRG3CON0 |
| 41Eh | T6CLKCON | 49Eh | T8CLKCON | 51Eh | — | 59Eh | DAC7CON0 | 61Eh | — | 69Eh | COG1ASD1 | 71Eh | COG2ASD1 | 79Eh | PRG3CON1 |
| 41Fh | T6RST | 49Fh | T8RST | 51Fh | — | 59Fh | DAC7REF | 61Fh | — | 69Fh | COG1STR | 71Fh | COG2STR | 79Fh | PRG3CON2 |
| 420h | Unimplemented Read as '0' | 4A0h | Unimplemented Read as '0' | 520h | Unimplemented Read as '0' | 5A0h | Unimplemented Read as '0' | 620h | Unimplemented Read as '0' | 6A0h | Unimplemented Read as '0' | 720h | Unimplemented Read as '0' | 7A0h | Unimplemented Read as '0' |
| 46Fh | | 4EFh | | 56Fh | | 5EFh | | 66Fh | | 6EFh | | 76Fh | | 7EFh | |
| 470h | 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'.

## TABLE 3-6: PIC16(L)F1776 MEMORY MAP, BANK 8-15

| BANK 8 | | BANK 9 | | BANK 10 | | BANK 11 | | BANK 12 | | BANK 13 | | BANK 14 | | BANK 15 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 400h | Core Registers (Table 3-2) | 480h | Core Registers (Table 3-2) | 500h | Core Registers (Table 3-2) | 580h | Core Registers (Table 3-2) | 600h | Core Registers (Table 3-2) | 680h | Core Registers (Table 3-2) | 700h | Core Registers (Table 3-2) | 780h | Core Registers (Table 3-2) |
| 40Bh | | 48Bh | | 50Bh | | 58Bh | | 60Bh | | 68Bh | | 70Bh | | 78Bh | |
| 40Ch | — | 48Ch | — | 50Ch | — | 58Ch | — | 60Ch | — | 68Ch | — | 70Ch | — | 78Ch | — |
| 40Dh | HIDRVB | 48Dh | — | 50Dh | — | 58Dh | DACLD | 60Dh | — | 68Dh | COG1PHR | 70Dh | COG2PHR | 78Dh | — |
| 40Eh | — | 48Eh | ADRESL | 50Eh | — | 58Eh | DAC1CON0 | 60Eh | — | 68Eh | COG1PHF | 70Eh | COG2PHF | 78Eh | PRG1RTSS |
| 40Fh | TMR5L | 48Fh | ADRESH | 50Fh | OPA1NCHS | 58Fh | DAC1REFL | 60Fh | — | 68Fh | COG1BLKR | 70Fh | COG2BLKR | 78Fh | PRG1FTSS |
| 410h | TMR5H | 490h | ADCON0 | 510h | OPA1PCHS | 590h | DAC1REFH | 610h | — | 690h | COG1BLKF | 710h | COG2BLKF | 790h | PRG1INS |
| 411h | T5CON | 491h | ADCON1 | 511h | OPA1CON | 591h | DAC2CON0 | 611h | — | 691h | COG1DBR | 711h | COG2DBR | 791h | PRG1CON0 |
| 412h | T5GCON | 492h | ADCON2 | 512h | OPA1ORS | 592h | DAC2REFL | 612h | — | 692h | COG1DBF | 712h | COG2DBF | 792h | PRG1CON1 |
| 413h | T4TMR | 493h | T2TMR | 513h | OPA2NCHS | 593h | DAC2REFH | 613h | — | 693h | COG1CON0 | 713h | COG2CON0 | 793h | PRG1CON2 |
| 414h | T4PR | 494h | T2PR | 514h | OPA2PCHS | 594h | DAC3CON0 | 614h | PWM3DCL | 694h | COG1CON1 | 714h | COG2CON1 | 794h | PRG2RTSS |
| 415h | T4CON | 495h | T2CON | 515h | OPA2CON | 595h | DAC3REF | 615h | PWM3DCH | 695h | COG1RIS0 | 715h | COG2RIS0 | 795h | PRG2FTSS |
| 416h | T4HLT | 496h | T2HLT | 516h | OPA2ORS | 596h | DAC4CON0 | 616h | PWM3CON | 696h | COG1RIS1 | 716h | COG2RIS1 | 796h | PRG2INS |
| 417h | T4CLKCON | 497h | T2CLKCON | 517h | OPA3NCHS | 597h | DAC4REF | 617h | PWM4DCL | 697h | COG1RSIM0 | 717h | COG2RSIM0 | 797h | PRG2CON0 |
| 418h | T4RST | 498h | T2RST | 518h | OPA3PCHS | 598h | DAC5CON0 | 618h | PWM4DCH | 698h | COG1RSIM1 | 718h | COG2RSIM1 | 798h | PRG2CON1 |
| 419h | — | 499h | — | 519h | OPA3CON | 599h | DAC5REFL | 619h | PWM4CON | 699h | COG1FIS0 | 719h | COG2FIS0 | 799h | PRG2CON2 |
| 41Ah | T6TMR | 49Ah | T8TMR | 51Ah | OPA3ORS | 59Ah | DAC5REFH | 61Ah | PWM9DCL | 69Ah | COG1FIS1 | 71Ah | COG2FIS1 | 79Ah | PRG3RTSS |
| 41Bh | T6PR | 49Bh | T8PR | 51Bh | — | 59Bh | — | 61Bh | PWM9DCH | 69Bh | COG1FSIM0 | 71Bh | COG2FSIM0 | 79Bh | PRG3FTSS |
| 41Ch | T6CON | 49Ch | T8CON | 51Ch | — | 59Ch | — | 61Ch | PWM9CON | 69Ch | COG1FSIM1 | 71Ch | COG2FSIM1 | 79Ch | PRG3INS |
| 41Dh | T6HLT | 49Dh | T8HLT | 51Dh | — | 59Dh | — | 61Dh | — | 69Dh | COG1ASD0 | 71Dh | COG2ASD0 | 79Dh | PRG3CON0 |
| 41Eh | T6CLKCON | 49Eh | T8CLKCON | 51Eh | — | 59Eh | DAC7CON0 | 61Eh | — | 69Eh | COG1ASD1 | 71Eh | COG2ASD1 | 79Eh | PRG3CON1 |
| 41Fh | T6RST | 49Fh | T8RST | 51Fh | — | 59Fh | DAC7REF | 61Fh | — | 69Fh | COG1STR | 71Fh | COG2STR | 79Fh | PRG3CON2 |
| 420h | General Purpose Register 80 Bytes | 4A0h | General Purpose Register 80 Bytes | 520h | General Purpose Register 80 Bytes | 5A0h | General Purpose Register 80 Bytes | 620h<br>64Fh | General Purpose Register 48 Bytes | 6A0h | Unimplemented Read as '0' | 720h | Unimplemented Read as '0' | 7A0h | Unimplemented Read as '0' |
| | | | | | | | | 650h<br>66Fh | Unimplemented Read as '0' | | | | | | |
| 46Fh | | 4EFh | | 56Fh | | 5EFh | | | | 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'.

## TABLE 3-7: PIC16(L)F1773/6 MEMORY MAP, BANK 16-23

| | BANK 16 | BANK 17 | BANK 18 | BANK 19 | BANK 20 | BANK 21 | BANK 22 | BANK 23 |
|---|---|---|---|---|---|---|---|---|
| 800h | Core Registers (Table 3-2) | 880h Core Registers (Table 3-2) | 900h Core Registers (Table 3-2) | 980h Core Registers (Table 3-2) | A00h Core Registers (Table 3-2) | A80h Core Registers (Table 3-2) | B00h Core Registers (Table 3-2) | B80h Core Registers (Table 3-2) |
| 80Bh | | 88Bh | 90Bh | 98Bh | A0Bh | A8Bh | B0Bh | B8Bh |
| 80Ch | — | 88Ch Unimplemented Read as '0' | 90Ch CM4CON0 | 98Ch Unimplemented Read as '0' | A0Ch Unimplemented Read as '0' | A8Ch Unimplemented Read as '0' | B0Ch Unimplemented Read as '0' | B8Ch Unimplemented Read as '0' |
| 80Dh | COG3PHR | | 90Dh CM4CON1 | | | | | |
| 80Eh | COG3PHF | | 90Eh CM4NSEL | | | | | |
| 80Fh | COG3BLKR | | 90Fh CM4PSEL | | | | | |
| 810h | COG3BLKF | | 910h CM5CON0 | | | | | |
| 811h | COG3DBR | | 911h CM5CON1 | | | | | |
| 812h | COG3DBF | | 912h CM5NSEL | | | | | |
| 813h | COG3CON0 | | 913h CM5PSEL | | | | | |
| 814h | COG3CON1 | | 914h CM6CON0 | | | | | |
| 815h | COG3RIS0 | | 915h CM6CON1 | | | | | |
| 816h | COG3RIS1 | | 916h CM6NSEL | | | | | |
| 817h | COG3RSIM0 | | 917h CM6PSEL | | | | | |
| 818h | COG3RSIM1 | | 918h Unimplemented Read as '0' | | | | | |
| 819h | COG3FIS0 | | | | | | | |
| 81Ah | COG3FIS1 | | | | | | | |
| 81Bh | COG3FSIM0 | | | | | | | |
| 81Ch | COG3FSIM1 | | | | | | | |
| 81Dh | COG3ASD0 | | | | | | | |
| 81Eh | COG3ASD1 | | | | | | | |
| 81Fh | COG3STR | | | | | | | |
| | Unimplemented Read as '0' | | | | | | | |
| 86Fh | | 8EFh | 96Fh | 9EFh | A6Fh | AEFh | B6Fh | BEFh |
| 870h | Accesses 70h – 7Fh | 8F0h Accesses 70h – 7Fh | 970h Accesses 70h – 7Fh | 9F0h Accesses 70h – 7Fh | A70h Accesses 70h – 7Fh | AF0h Accesses 70h – 7Fh | B70h Accesses 70h – 7Fh | BF0h Accesses 70h – 7Fh |
| 87Fh | | 8FFh | 97Fh | 9FFh | A7Fh | AFFh | B7Fh | BFFh |

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

## TABLE 3-8: PIC16(L)F1773/6 MEMORY MAP, BANK 24-31

| BANK 24 | BANK 25 | BANK 26 | BANK 27 | BANK 28 | BANK 29 | BANK 30 | BANK 31 |
|---|---|---|---|---|---|---|---|
| C00h Core Registers (Table 3-2) C0Bh | C80h Core Registers (Table 3-2) C8Bh | D00h Core Registers (Table 3-2) D0Bh | D80h Core Registers (Table 3-2) D8Bh | E00h Core Registers (Table 3-2) E0Bh | E80h Core Registers (Table 3-2) E8Bh | F00h Core Registers (Table 3-2) F0Bh | F80h Core Registers (Table 3-2) F8Bh |
| C0Ch — | C8Ch — | D0Ch — | See Table 3-9 for register mapping details | See Table 3-9 for register mapping details | See Table 3-9 for register mapping details | See Table 3-9 for register mapping details | See Table 3-10 for register mapping details |
| C0Dh — | C8Dh — | D0Dh — | | | | | |
| C0Eh — | C8Eh — | D0Eh — | | | | | |
| C0Fh — | C8Fh — | D0Fh — | | | | | |
| C10h — | C90h — | D10h — | | | | | |
| C11h — | C91h — | D11h — | | | | | |
| C12h — | C92h — | D12h — | | | | | |
| C13h — | C93h — | D13h — | | | | | |
| C14h — | C94h — | D14h — | | | | | |
| C15h — | C95h — | D15h — | | | | | |
| C16h — | C96h — | D16h — | | | | | |
| C17h — | C97h — | D17h — | | | | | |
| C18h — | C98h — | D18h — | | | | | |
| C19h — | C99h — | D19h — | | | | | |
| C1Ah — | C9Ah — | D1Ah — | | | | | |
| C1Bh — | C9Bh — | D1Bh — | | | | | |
| C1Ch — | C9Ch — | D1Ch — | | | | | |
| C1Dh — | C9Dh — | D1Dh — | | | | | |
| C1Eh — | C9Eh — | D1Eh — | | | | | |
| C1Fh — | C9Fh — | D1Fh — | | | | | |
| C20h Unimplemented Read as '0' C6Fh | CA0h Unimplemented Read as '0' CEFh | D20h Unimplemented Read as '0' D6Fh | DEFh | E6Fh | EEFh | F6Fh | FEFh |
| C70h Accesses 70h – 7Fh CFFh | CF0h Accesses 70h – 7Fh CFFh | D70h Accesses 70h – 7Fh D7Fh | DF0h Accesses 70h – 7Fh DFFh | E70h Accesses 70h – 7Fh E7Fh | EF0h Accesses 70h – 7Fh EFFh | F70h Accesses 70h – 7Fh F7Fh | FF0h Accesses 70h – 7Fh FFFh |

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

**TABLE 3-9: PIC16(L)F1773/6 MEMORY MAP, BANK 27-30**

| | Bank 27 | | Bank 28 | | Bank 29 | | Bank 30 |
|---|---|---|---|---|---|---|---|
| D8Ch | — | E0Ch | PPSLOCK | E8Ch | — | F0Ch | — |
| D8Dh | — | E0Dh | INTPPS | E8Dh | — | F0Dh | — |
| D8Eh | PWMEN | E0Eh | T0CKIPPS | E8Eh | — | F0Eh | — |
| D8Fh | PWMLD | E0Fh | T1CKIPPS | E8Fh | — | F0Fh | CLCDATA |
| D90h | PWMOUT | E10h | T1GPPS | E90h | RA0PPS | F10h | CLC1CON |
| D91h | PWM5PHL | E11h | T3CKIPPS | E91h | RA1PPS | F11h | CLC1POL |
| D92h | PWM5PHH | E12h | T3GPPS | E92h | RA2PPS | F12h | CLC1SEL0 |
| D93h | PWM5DCL | E13h | T5CKIPPS | E93h | RA3PPS | F13h | CLC1SEL1 |
| D94h | PWM5DCH | E14h | T5GPPS | E94h | RA4PPS | F14h | CLC1SEL2 |
| D95h | PWM5PRL | E15h | T2INPPS | E95h | RA5PPS | F15h | CLC1SEL3 |
| D96h | PWM5PRH | E16h | T4INPPS | E96h | RA6PPS | F16h | CLC1GLS0 |
| D97h | PWM5OFL | E17h | T6INPPS | E97h | RA7PPS | F17h | CLC1GLS1 |
| D98h | PWM5OFH | E18h | T8INPPS | E98h | RB0PPS | F18h | CLC1GLS2 |
| D99h | PWM5TMRL | E19h | CCP1PPS | E99h | RB1PPS | F19h | CLC1GLS3 |
| D9Ah | PWM5TMRH | E1Ah | CCP2PPS | E9Ah | RB2PPS | F1Ah | CLC2CON |
| D9Bh | PWM5CON | E1Bh | CCP7PPS | E9Bh | RB3PPS | F1Bh | CLC2POL |
| D9Ch | PWM5INTE | E1Ch | — | E9Ch | RB4PPS | F1Ch | CLC2SEL0 |
| D9Dh | PWM5INTF | E1Dh | COG1INPPS | E9Dh | RB5PPS | F1Dh | CLC2SEL1 |
| D9Eh | PWM5CLKCON | E1Eh | COG2INPPS | E9Eh | RB6PPS | F1Eh | CLC2SEL2 |
| D9Fh | PWM5LDCON | E1Fh | COG3INPPS | E9Fh | RB7PPS | F1Fh | CLC2SEL3 |
| DA0h | PWM5OFCON | E20h | — | EA0h | RC0PPS | F20h | CLC2GLS0 |
| DA1h | PWM6PHL | E21h | MD1CLPPS | EA1h | RC1PPS | F21h | CLC2GLS1 |
| DA2h | PWM6PHH | E22h | MD1CHPPS | EA2h | RC2PPS | F22h | CLC2GLS2 |
| DA3h | PWM6DCL | E23h | MD1MODPPS | EA3h | RC3PPS | F23h | CLC2GLS3 |
| DA4h | PWM6DCH | E24h | MD2CLPPS | EA4h | RC4PPS | F24h | CLC3CON |
| DA5h | PWM6PRL | E25h | MD2CHPPS | EA5h | RC5PPS | F25h | CLC3POL |
| DA6h | PWM6PRH | E26h | MD2MODPPS | EA6h | RC6PPS | F26h | CLC3SEL0 |
| DA7h | PWM6OFL | E27h | MD3CLPPS | EA7h | RC7PPS | F27h | CLC3SEL1 |
| DA8h | PWM6OFH | E28h | MD3CHPPS | EA8h | — | F28h | CLC3SEL2 |
| DA9h | PWM6TMRL | E29h | MD3MODPPS | EA9h | — | F29h | CLC3SEL3 |
| DAAh | PWM6TMRH | E2Ah | — | EAAh | — | F2Ah | CLC3GLS0 |
| DABh | PWM6CON | E2Bh | — | EABh | — | F2Bh | CLC3GLS1 |
| DACh | PWM6INTE | E2Ch | — | EACh | — | F2Ch | CLC3GLS2 |
| DADh | PWM6INTF | E2Dh | PRG1RPPS | EADh | — | F2Dh | CLC3GLS3 |
| DAEh | PWM6CLKCON | E2Eh | PRG1FPPS | EAEh | — | F2Eh | CLC4CON |
| DAFh | PWM6LDCON | E2Fh | PRG2RPPS | EAFh | — | F2Fh | CLC4POL |
| DB0h | PWM6OFCON | E30h | PRG2FPPS | EB0h | — | F30h | CLC4SEL0 |
| DB1h | PWM11PHL | E31h | PRG3FPPS | EB1h | — | F31h | CLC4SEL1 |
| DB2h | PWM11PHH | E32h | PRG3RPPS | EB2h | — | F32h | CLC4SEL2 |
| DB3h | PWM11DCL | E33h | | EB3h | — | F33h | CLC4SEL3 |
| DB4h | PWM11DCH | E34h | | EB4h | — | F34h | CLC4GLS0 |
| DB5h | PWM11PRL | E35h | CLCIN0PPS | EB5h | — | F35h | CLC4GLS1 |
| DB6h | PWM11PRH | E36h | CLCIN1PPS | EB6h | — | F36h | CLC4GLS2 |
| DB7h | PWM11OFL | E37h | CLCIN2PPS | EB7h | — | F37h | CLC4GLS3 |
| DB8h | PWM11OFH | E38h | CLCIN3PPS | EB8h | — | F38h | — |
| DB9h | PWM11TMRL | E39h | ADCACTPPS | EB9h | — | F39h | — |
| DBAh | PWM11TMRH | E3Ah | SSPCLKPPS | EBAh | — | F3Ah | — |
| DBBh | PWM11CON | E3Bh | SSPDATPPS | EBBh | — | F3Bh | — |
| DBCh | PWM11INTE | E3Ch | SSPSSPPS | EBCh | — | F3Ch | — |
| DBDh | PWM11INTF | E3Dh | RXPPS | EBDh | — | F3Dh | — |
| DBEh | PWM11CLKCON | E3Eh | CKPPS | EBEh | — | F3Eh | — |
| DBFh | PWM11LDCON | E3Fh | — | EBFh | — | F3Fh | — |
| DC0h | PWM11OFCON | E40h | | EC0h | | F40h | |
| | | | — | | — | | — |
| DEFh | | E6Fh | | EEFh | | F6Fh | |

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

**TABLE 3-10: PIC16(L)F1773/6 MEMORY MAP, BANK 31**

| Bank 31 | |
|---|---|
| F8Ch<br>FE3h | Unimplemented<br>Read as '0' |
| FE4h | STATUS_SHAD |
| FE5h | WREG_SHAD |
| FE6h | BSR_SHAD |
| FE7h | PCLATH_SHAD |
| FE8h | FSR0L_SHAD |
| FE9h | FSR0H_SHAD |
| FEAh | FSR1L_SHAD |
| FEBh | FSR1H_SHAD |
| FECh | — |
| FEDh | STKPTR |
| FEEh | TOSL |
| FEFh | TOSH |

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

### 3.4.5 CORE FUNCTION REGISTERS SUMMARY

The Core Function registers listed in Table 3-11 can be addressed from any Bank.

**TABLE 3-11: CORE FUNCTION REGISTERS SUMMARY(1)**

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

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

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

## TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 0** | | | | | | | | | | | |
| 00Ch | PORTA | RA7 | RA6 | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 | xxxx xxxx | uuuu uuuu |
| 00Dh | PORTB | RB7 | RB6 | RB5 | RB4 | RB3 | RB2 | RB1 | RB0 | xxxx xxxx | uuuu uuuu |
| 00Eh | PORTC | RC7 | RC6 | RC5 | RC4 | RC3 | RC2 | RC1 | RC0 | xxxx xxxx | uuuu uuuu |
| 00Fh | — | Unimplemented | | | | | | | | — | — |
| 010h | PORTE | — | — | — | — | RE3 | — | — | — | ---- x--- | ---- u--- |
| 011h | PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 0000 0000 | 0000 0000 |
| 012h | PIR2 | OSFIF | C2IF | C1IF | COG1IF | BCL1IF | C4IF | C3IF | CCP2IF | 0000 0000 | 0000 0000 |
| 013h | PIR3 | — | — | COG2IF | ZCDIF | CLC4IF | CLC3IF | CLC2IF | CLC1IF | --00 0000 | --00 0000 |
| 014h | PIR4 | — | TMR8IF | TMR5GIF | TMR5IF | TMR3GIF | TMR3IF | TMR6IF | TRM4IF | -000 0000 | -000 0000 |
| 015h | PIR5 | — | CCP7IF | — | COG3IF | — | — | C6IF | C5IF | -0-0 --00 | -0-0 --00 |
| 016h | PIR6 | — | — | — | — | — | PWM11IF | PWM6IF | PWM5IF | ---- -000 | ---- -000 |
| 017h | TMR0 | Timer0 Module Register | | | | | | | | 0000 0000 | 0000 0000 |
| 018h | TMR1L | Holding Register for the Least Significant Byte of the 16-bit TMR1 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 019h | TMR1H | Holding Register for the Most Significant Byte of the 16-bit TMR1 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 01Ah | T1CON | CS<1:0> | | CKPS<1:0> | | OSCEN | $\overline{\text{SYNC}}$ | — | ON | 0000 00-0 | uuuu uu-u |
| 01Bh | T1GCON | GE | GPOL | GTM | GSPM | GGO/DONE | GVAL | GSS<1:0> | | 0000 0x00 | uuuu uxuu |
| 01Ch | TMR3L | Holding Register for the Least Significant Byte of the 16-bit TMR3 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 01Dh | TMR3H | Holding Register for the Most Significant Byte of the 16-bit TMR3 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 01Eh | T3CON | CS<1:0> | | CKPS<1:0> | | OSCEN | $\overline{\text{SYNC}}$ | — | ON | 0000 00-0 | uuuu uu-u |
| 01Fh | T3GCON | GE | GPOL | GTM | GSPM | GGO/DONE | GVAL | GSS<1:0> | | 0000 0x00 | uuuu uxuu |

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

**2:** Unimplemented on PIC16LF1776.

**TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 1** | | | | | | | | | | | |
| 08Ch | TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 1111 1111 | 1111 1111 |
| 08Dh | TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 1111 1111 | 1111 1111 |
| 08Eh | TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 1111 1111 | 1111 1111 |
| 08Fh | — | Unimplemented | | | | | | | | — | — |
| 090h | TRISE | — | — | — | — | TRISE3 | — | — | — | ---- 1--- | ---- 1--- |
| 091h | PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 0000 0000 | 0000 0000 |
| 092h | PIE2 | OSFIE | C2IE | C1IE | COG1IE | BCL1IE | C4IE | C3IE | CCP2IE | 0000 0000 | 0000 0000 |
| 093h | PIE3 | — | — | COG2IE | ZCDIE | CLC4IE | CLC3IE | CLC2IE | CLC1IE | --00 0000 | --00 0000 |
| 094h | PIE4 | — | TMR8IE | TMR5GIE | TMR5IE | TMR3GIE | TMR3IE | TMR6IE | TRM4IE | -000 0000 | -000 0000 |
| 095h | PIE5 | — | CCP7IE | — | COG3IE | — | — | C6IE | C5IE | -0-0 --00 | -0-0 --00 |
| 096h | PIE6 | — | — | — | — | — | PWM11IE | PWM6IE | PWM5IE | ---- -000 | ---- -000 |
| 097h | OPTION_REG | $\overline{\text{WPUEN}}$ | INTEDG | TMR0CS | TMR0SE | PSA | PS<2:0> | | | 1111 1111 | 1111 1111 |
| 098h | PCON | STKOVF | STKUNF | — | $\overline{\text{RWDT}}$ | $\overline{\text{RMCLR}}$ | $\overline{\text{RI}}$ | $\overline{\text{POR}}$ | $\overline{\text{BOR}}$ | 00-1 11qq | qq-q qquu |
| 099h | WDTCON | — | — | WDTPS<4:0> | | | | | SWDTEN | --01 0110 | --01 0110 |
| 09Ah | OSCTUNE | — | — | TUN<5:0> | | | | | | --00 0000 | --00 0000 |
| 09Bh | OSCCON | SPLLEN | IRCF<3:0> | | | | — | SCS<1:0> | | 0011 1-00 | 0011 1-00 |
| 09Ch | OSCSTAT | SOSCR | PLLR | OSTS | HFIOFR | HFIOFL | MFIOFR | LFIOFR | HFIOFS | 0qq0 0q0q | qqqq qq0q |
| 09Dh | BORCON | SBOREN | BORFS | — | — | — | — | — | BORRDY | 10-- ---q | uu-- ---u |
| 09Eh | FVRCON | FVREN | FVRRDY | TSEN | TSRNG | CDAFVR<1:0> | | ADFVR<1:0> | | 0q00 0000 | 0q00 0000 |
| 09Fh | ZCD1CON | EN | — | OUT | POL | — | — | INTP | INTN | 0-x0 --00 | 0-x0 --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:** Unimplemented, read as '1'.

**2:** Unimplemented on PIC16LF1776.

**TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 2** | | | | | | | | | | | |
| 10Ch | LATA | LATA7 | LATA6 | LATA5 | LATA4 | LATA3 | LATA2 | LATA1 | LATA0 | xxxx xxxx | uuuu uuuu |
| 10Dh | LATB | LATB7 | LATB6 | LATB5 | LATB4 | LATB3 | LATB2 | LATB1 | LATB0 | xxxx xxxx | uuuu uuuu |
| 10Eh | LATC | LATC7 | LATC6 | LATC5 | LATC4 | LATC3 | LATC2 | LATC1 | LATC0 | xxxx xxxx | uuuu uuuu |
| 10Fh | — | Unimplemented | | | | | | | | — | — |
| 110h | — | Unimplemented | | | | | | | | — | — |
| 111h | CMOUT | — | — | MC6OUT | MC5OUT | MC4OUT | MC3OUT | MC2OUT | MC1OUT | --00 0000 | --00 0000 |
| 112h | CM1CON0 | ON | OUT | — | POL | ZLF | Reserved | HYS | SYNC | 00-0 0100 | 00-0 0100 |
| 113h | CM1CON1 | — | — | — | — | — | — | INTP | INTN | ---- --00 | ---- --00 |
| 114h | CM1NSEL | — | — | — | — | NCH<3:0> | | | | ---- 0000 | ---- 0000 |
| 115h | CM1PSEL | — | — | — | — | PCH<3:0> | | | | ---- 0000 | ---- 0000 |
| 116h | CM2CON0 | ON | OUT | — | POL | ZLF | Reserved | HYS | SYNC | 00-0 0100 | 00-0 0100 |
| 117h | CM2CON1 | — | — | — | — | — | — | INTP | INTN | ---- --00 | ---- --00 |
| 118h | CM2NSEL | — | — | — | — | NCH<3:0> | | | | ---- 0000 | ---- 0000 |
| 119h | CM2PSEL | — | — | — | — | PCH<3:0> | | | | ---- 0000 | ---- 0000 |
| 11Ah | CM3CON0 | ON | OUT | — | POL | ZLF | Reserved | HYS | SYNC | 00-0 0100 | 00-0 0100 |
| 11Bh | CM3CON1 | — | — | — | — | — | — | INTP | INTN | ---- --00 | ---- --00 |
| 11Ch | CM3NSEL | — | — | — | — | NCH<3:0> | | | | ---- 0000 | ---- 0000 |
| 11Dh | CM3PSEL | — | — | — | — | PCH<3:0> | | | | ---- 0000 | ---- 0000 |
| 11Eh | — | Unimplemented | | | | | | | | — | — |
| 11Fh | — | 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:** Unimplemented, read as '1'.

**2:** Unimplemented on PIC16LF1776.

**TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 3** | | | | | | | | | | | |
| 18Ch | ANSELA | — | — | ANSA5 | ANSA4 | ANSA3 | ANSA2 | ANSA1 | ANSA0 | --11 1111 | --11 1111 |
| 18Dh | ANSELB | — | — | ANSB5 | ANSB4 | ANSB3 | ANSB2 | ANSB1 | ANSB0 | --11 1111 | --11 1111 |
| 18Eh | ANSELC | ANSC7 | ANSC6 | ANSC5 | ANSC4 | ANSC3 | ANSC2 | — | — | 1111 11-- | 1111 11-- |
| 18Fh | — | Unimplemented | | | | | | | | — | — |
| 190h | — | Unimplemented | | | | | | | | — | — |
| 191h | PMADRL | Program Memory Address Register Low Byte | | | | | | | | 0000 0000 | 0000 0000 |
| 192h | PMADRH | —(1) | Program Memory Address Register High Byte | | | | | | | 1000 0000 | 1000 0000 |
| 193h | PMDATL | Program Memory Read Data Register Low Byte | | | | | | | | xxxx xxxx | uuuu uuuu |
| 194h | PMDATH | — | — | Program Memory Read Data Register High Byte | | | | | | --xx xxxx | --uu uuuu |
| 195h | PMCON1 | —(1) | CFGS | LWLO | FREE | WRERR | WREN | WR | RD | 1000 x000 | 1000 q000 |
| 196h | PMCON2 | Program Memory Control Register 2 | | | | | | | | 0000 0000 | 0000 0000 |
| 197h | VREGCON | — | — | — | — | — | — | VREGPM(2) | Reserved | ---- --01 | ---- --01 |
| 198h | — | Unimplemented | | | | | | | | — | — |
| 198h | — | Unimplemented | | | | | | | | — | — |
| 199h | RC1REG | EUSART Receive Data Register | | | | | | | | 0000 0000 | 0000 0000 |
| 19Ah | TX1REG | EUSART Transmit Data Register | | | | | | | | 0000 0000 | 0000 0000 |
| 19Bh | SP1BRGL | SP1BRG<7:0> | | | | | | | | 0000 0000 | 0000 0000 |
| 19Ch | SP1BRGH | SP1BRG<15:8> | | | | | | | | 0000 0000 | 0000 0000 |
| 19Dh | RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 0000 0000 | 0000 0000 |
| 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 |
| 199h — 19Fh | — | 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:** Unimplemented, read as '1'.

**2:** Unimplemented on PIC16LF1776.

**TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 4** | | | | | | | | | | | |
| 20Ch | WPUA | WPUA7 | WPUA6 | WPUA5 | WPUA4 | WPUA3 | WPUA2 | WPUA1 | WPUA0 | 1111 1111 | 1111 1111 |
| 20Dh | WPUB | WPUB7 | WPUB6 | WPUB5 | WPUB4 | WPUB3 | WPUB2 | WPUB1 | WPUB0 | 1111 1111 | 1111 1111 |
| 20Eh | WPUC | WPUC7 | WPUC6 | WPUC5 | WPUC4 | WPUC3 | WPUC2 | WPUC1 | WPUC0 | 1111 1111 | 1111 1111 |
| 20Fh | — | Unimplemented | | | | | | | | — | — |
| 210h | WPUE | — | — | — | — | WPUE3 | — | — | — | ---- 1--- | ---- 1--- |
| 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 — 21Ah | — | Unimplemented | | | | | | | | — | — |
| 21Bh | MD3CON0 | EN | — | OUT | OPOL | — | — | — | BIT | 0-00 ---0 | 0-00 ---0 |
| 21Ch | MD3CON1 | — | — | CHPOL | CHSYNC | — | — | CLPOL | CLSYNC | --00 --00 | --00 --00 |
| 21Dh | MD3SRC | — | — | — | MS<4:0> | | | | | ---0 0000 | ---0 0000 |
| 21Eh | MD3CARL | — | — | — | CL<4:0> | | | | | ---0 0000 | ---0 0000 |
| 21Fh | MD3CARH | — | — | — | CH<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:** Unimplemented, read as '1'.

**2:** Unimplemented on PIC16LF1776.

**TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 5** | | | | | | | | | | | |
| 28Ch | ODCONA | ODA7 | ODA6 | ODA5 | ODA4 | ODA3 | ODA2 | ODA1 | ODA0 | 0000 0000 | 0000 0000 |
| 28Dh | ODCONB | ODB7 | ODB6 | ODB5 | ODB4 | ODB3 | ODB2 | ODB1 | ODB0 | 0000 0000 | 0000 0000 |
| 28Eh | ODCONC | ODC7 | ODC6 | ODC5 | ODC4 | ODC3 | ODC2 | ODC1 | ODC0 | 0000 0000 | 0000 0000 |
| 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 | EN | — | OUT | FMT | MODE<3:0> | | | | 0-00 0000 | 0-00 0000 |
| 294h | CCP1CAP | — | — | — | — | CTS<3:0> | | | | ---- 0000 | ---- 0000 |
| 295h | CCPR2L | Capture/Compare/PWM Register 2 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 296h | CCPR2H | Capture/Compare/PWM Register 2 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 297h | CCP2CON | EN | — | OUT | FMT | MODE<3:0> | | | | 0-00 0000 | 0-00 0000 |
| 298h | CCP2CAP | — | — | — | — | CTS<3:0> | | | | ---- 0000 | ---- 0000 |
| 299h | CCPR7L | Capture/Compare/PWM Register 7 (LSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 29Ah | CCPR7H | Capture/Compare/PWM Register 7 (MSB) | | | | | | | | xxxx xxxx | uuuu uuuu |
| 29Bh | CCP7CON | EN | — | OUT | FMT | MODE<3:0> | | | | 0-00 0000 | 0-00 0000 |
| 29Ch | CCP7CAP | — | — | — | — | CTS<3:0> | | | | ---- 0000 | ---- 0000 |
| 29Dh | — | Unimplemented | | | | | | | | — | — |
| 29Eh | CCPTMRS1 | — | — | C7TSEL<1:0> | | C2TSEL<1:0> | | C1TSEL<1:0> | | --00 0000 | --00 0000 |
| 29Fh | CCPTMRS2 | — | — | P9TSEL<1:0> | | P4TSEL<1:0> | | P3TSEL<1:0> | | --00 0000 | --00 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:** Unimplemented, read as '1'.

**2:** Unimplemented on PIC16LF1776.

**TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 6** | | | | | | | | | | | |
| 30Ch | SLRCONA | SLRA7 | SLRA6 | SLRA5 | SLRA4 | SLRA3 | SLRA2 | SLRA1 | SLRA0 | 1111 1111 | 1111 1111 |
| 30Dh | SLRCONB | SLRB7 | SLRB6 | SLRB5 | SLRB4 | SLRB3 | SLRB2 | SLRB1 | SLRB0 | 1111 1111 | 1111 1111 |
| 30Eh | SLRCONC | SLRC7 | SLRC6 | SLRC5 | SLRC4 | SLRC3 | SLRC2 | SLRC1 | SLRC0 | 1111 1111 | 1111 1111 |
| 30Fh — 314h | — | Unimplemented | | | | | | | | — | — |
| 315h | MD1CON0 | EN | — | OUT | OPOL | — | — | — | BIT | 0-00 ---0 | 0-00 ---0 |
| 316h | MD1CON1 | — | — | CHPOL | CHSYNC | — | — | CLPOL | CLSYNC | --00 --00 | --00 --00 |
| 317h | MD1SRC | — | — | — | MS<4:0> | | | | | ---0 0000 | ---0 0000 |
| 318h | MD1CARL | — | — | — | CL<4:0> | | | | | ---0 0000 | ---0 0000 |
| 319h | MD1CARH | — | — | — | CH<4:0> | | | | | ---0 0000 | ---0 0000 |
| 31Ah | — | Unimplemented | | | | | | | | — | — |
| 31Bh | MD2CON0 | EN | — | OUT | OPOL | — | — | — | BIT | 0-00 ---0 | 0-00 ---0 |
| 31Ch | MD2CON1 | — | — | CHPOL | CHSYNC | — | — | CLPOL | CLSYNC | --00 --00 | --00 --00 |
| 31Dh | MD2SRC | — | — | — | MS<4:0> | | | | | ---0 0000 | ---0 0000 |
| 31Eh | MD2CARL | — | — | — | CL<4:0> | | | | | ---0 0000 | ---0 0000 |
| 31Fh | MD2CARH | — | — | — | CH<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:** Unimplemented, read as '1'.
**2:** Unimplemented on PIC16LF1776.

**TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 7** | | | | | | | | | | | |
| 38Ch | INLVLA | INLVLA7 | INLVLA6 | INLVLA5 | INLVLA4 | INLVLA3 | INLVLA2 | INLVLA1 | INLVLA0 | 1111 1111 | 1111 1111 |
| 38Dh | INLVLB | INLVLB7 | INLVLB6 | INLVLB5 | INLVLB4 | INLVLB3 | INLVLB2 | INLVLB1 | INLVLB0 | 1111 1111 | 1111 1111 |
| 38Eh | INLVLC | INLVLC7 | INLVLC6 | INLVLC5 | INLVLC4 | INLVLC3 | INLVLC2 | INLVLC1 | INLVLC0 | 1111 1111 | 1111 1111 |
| 38Fh | — | Unimplemented | | | | | | | | — | — |
| 390h | INLVLE | — | — | — | — | INLVLE3 | — | — | — | ---- 1--- | ---- 1--- |
| 391h | IOCAP | IOCAP7 | IOCAP6 | IOCAP5 | IOCAP4 | IOCAP3 | IOCAP2 | IOCAP1 | IOCAP0 | 0000 0000 | 0000 0000 |
| 392h | IOCAN | IOCAN7 | IOCAN6 | IOCAN5 | IOCAN4 | IOCAN3 | IOCAN2 | IOCAN1 | IOCAN0 | 0000 0000 | 0000 0000 |
| 393h | IOCAF | IOCAF7 | IOCAF6 | IOCAF5 | IOCAF4 | IOCAF3 | IOCAF2 | IOCAF1 | IOCAF0 | 0000 0000 | 0000 0000 |
| 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 | IOCCP | IOCCP7 | IOCCP6 | IOCCP5 | IOCCP4 | IOCCP3 | IOCCP2 | IOCCP1 | IOCCP0 | 0000 0000 | 0000 0000 |
| 398h | IOCCN | IOCCN7 | IOCCN6 | IOCCN5 | IOCCN4 | IOCCN3 | IOCCN2 | IOCCN1 | IOCCN0 | 0000 0000 | 0000 0000 |
| 399h | IOCCF | IOCCF7 | IOCCF6 | IOCCF5 | IOCCF4 | IOCCF3 | IOCCF2 | IOCCF1 | IOCCF0 | 0000 0000 | 0000 0000 |
| 39Ah — 39Ch | — | Unimplemented | | | | | | | | — | — |
| 39Dh | IOCEP | — | — | — | — | IOCEP3 | — | — | — | ---- 0--- | ---- 0--- |
| 39Eh | IOCEN | — | — | — | — | IOCEN3 | — | — | — | ---- 0--- | ---- 0--- |
| 39Fh | IOCEF | — | — | — | — | IOCEF3 | — | — | — | ---- 0--- | ---- 0--- |

**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:** Unimplemented, read as '1'.
**2:** Unimplemented on PIC16LF1776.

**TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 8** | | | | | | | | | | | |
| 40Ch — 40Ch | — | Unimplemented | | | | | | | | — | — |
| 40Dh | HIDRVB | — | — | — | — | — | — | HIDB1 | HIDB0 | ---- --00 | ---- --00 |
| 40Eh | — | Unimplemented | | | | | | | | — | — |
| 40Fh | TMR5L | Holding Register for the Least Significant Byte of the 16-bit TMR5 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 410h | TMR5H | Holding Register for the Most Significant Byte of the 16-bit TMR5 Register | | | | | | | | xxxx xxxx | uuuu uuuu |
| 411h | T5CON | CS<1:0> | | CKPS<1:0> | | OSCEN | $\overline{\text{SYNC}}$ | — | ON | 0000 00-0 | uuuu uu-u |
| 412h | T5GCON | GE | GPOL | GTM | GSPM | GGO/ $\overline{\text{DONE}}$ | GVAL | GSS<1:0> | | 0000 0x00 | uuuu uxuu |
| 413h | T4TMR | Holding Register for the 8-bit TMR4 Register | | | | | | | | 0000 0000 | 0000 0000 |
| 414h | T4PR | TMR4 Period Register | | | | | | | | 1111 1111 | 1111 1111 |
| 415h | T4CON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | | 0000 0000 | 0000 0000 |
| 416h | T4HLT | PSYNC | CKPOL | CKSYNC | MODE<4:0> | | | | | ---0 0000 | ---0 0000 |
| 417h | T4CLKCON | — | — | — | — | CS<3:0> | | | | ---- 0000 | ---- 0000 |
| 418h | T4RST | — | — | — | RSEL<4:0> | | | | | ---0 0000 | ---0 0000 |
| 419h | — | Unimplemented | | | | | | | | — | — |
| 41Ah | T6TMR | Holding Register for the 8-bit TMR6 Register | | | | | | | | 0000 0000 | 0000 0000 |
| 41Bh | T6PR | TMR6 Period Register | | | | | | | | 1111 1111 | 1111 1111 |
| 41Ch | T6CON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | | 0000 0000 | 0000 0000 |
| 41Dh | T6HLT | PSYNC | CKPOL | CKSYNC | MODE<4:0> | | | | | ---0 0000 | ---0 0000 |
| 41Eh | T6CLKCON | — | — | — | — | CS<3:0> | | | | ---- 0000 | ---- 0000 |
| 41Fh | T6RST | — | — | — | RSEL<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:** Unimplemented, read as '1'.

**2:** Unimplemented on PIC16LF1776.

**TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 9** | | | | | | | | | | | |
| 48Ch to 48Dh | — | Unimplemented | | | | | | | | — | — |
| 48Eh | ADRESL | ADC Result Register Low | | | | | | | | xxxx xxxx | uuuu uuuu |
| 48Fh | ADRESH | ADC Result Register High | | | | | | | | xxxx xxxx | uuuu uuuu |
| 490h | ADCON0 | CHS<5:0> | | | | | | GO/$\overline{\text{DONE}}$ | ADON | 0000 0000 | 0000 0000 |
| 491h | ADCON1 | ADFM | ADCS<2:0> | | | — | ADNREF | ADPREF<1:0> | | 0000 -000 | 0000 -000 |
| 492h | ADCON2 | — | — | TRIGSEL<5:0> | | | | | | --00 0000 | --00 0000 |
| 493h | T2TMR | Holding Register for the 8-bit TMR2 Register | | | | | | | | 0000 0000 | 0000 0000 |
| 494h | T2PR | TMR2 Period Register | | | | | | | | 1111 1111 | 1111 1111 |
| 495h | T2CON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | | 0000 0000 | 0000 0000 |
| 496h | T2HLT | PSYNC | CKPOL | CKSYNC | MODE<4:0> | | | | | 0000 0000 | 0000 0000 |
| 497h | T2CLKCON | — | — | — | — | CS<3:0> | | | | ---- 0000 | ---- 0000 |
| 498h | T2RST | — | — | — | RSEL<4:0> | | | | | ---0 0000 | ---0 0000 |
| 499h | — | Unimplemented | | | | | | | | — | — |
| 49Ah | T8TMR | Holding Register for the 8-bit TMR8 Register | | | | | | | | 0000 0000 | 0000 0000 |
| 49Bh | T8PR | TMR8 Period Register | | | | | | | | 1111 1111 | 1111 1111 |
| 49Ch | T8CON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | | 0000 0000 | 0000 0000 |
| 49Dh | T8HLT | PSYNC | CKPOL | CKSYNC | MODE<4:0> | | | | | 0000 0000 | 0000 0000 |
| 49Eh | T8CLKCON | — | — | — | — | CS<3:0> | | | | ---- 0000 | ---- 0000 |
| 49Fh | T8RST | — | — | — | RSEL<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:** Unimplemented, read as '1'.

**2:** Unimplemented on PIC16LF1776.

**TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 10** | | | | | | | | | | | |
| 50Ch — 50Eh | — | Unimplemented | | | | | | | | — | — |
| 50Fh | OPA1NCHS | — | — | — | — | NCH<3:0> | | | | ---- 0000 | ---- 0000 |
| 510h | OPA1PCHS | — | — | — | — | PCH<3:0> | | | | ---- 0000 | ---- 0000 |
| 511h | OPA1CON | EN | — | — | UG | — | ORPOL | ORM<1:0> | | 0--0 -000 | 0--0 -000 |
| 512h | OPA1ORS | — | — | — | ORS<4:0> | | | | | ---0 0000 | ---0 0000 |
| 513h | OPA2NCHS | — | — | — | — | NCH<3:0> | | | | ---- 0000 | ---- 0000 |
| 514h | OPA2PCHS | — | — | — | — | PCH<3:0> | | | | ---- 0000 | ---- 0000 |
| 515h | OPA2CON | EN | — | — | UG | — | ORPOL | ORM<1:0> | | 0--0 -000 | 0--0 -000 |
| 516h | OPA2ORS | — | — | — | ORS<4:0> | | | | | ---0 0000 | ---0 0000 |
| 517h | OPA3NCHS | — | — | — | — | NCH<3:0> | | | | ---- 0000 | ---- 0000 |
| 518h | OPA3PCHS | — | — | — | — | PCH<3:0> | | | | ---- 0000 | ---- 0000 |
| 519h | OPA3CON | EN | SP | — | UG | — | ORPOL | ORM<1:0> | | 00-0 -000 | 00-0 -000 |
| 51Ah | OPA3ORS | — | — | — | ORS<4:0> | | | | | ---0 0000 | ---0 0000 |
| 51Bh — 51Fh | — | 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:** Unimplemented, read as '1'.

**2:** Unimplemented on PIC16LF1776.

**TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 11** | | | | | | | | | | | |
| 58Ch | — | Unimplemented | | | | | | | | — | — |
| 58Dh | DACLD | — | — | — | DAC5LD | — | — | DAC2LD | DAC1LD | ---0 --00 | ---0 --00 |
| 58Eh | DAC1CON0 | EN | FM | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 0000 0000 | 0000 0000 |
| 58Fh | DAC1REFL | REF<7:0> | | | | | | | | 00000 0000 | 0000 0000 |
| 590h | DAC1REFH | REF<15:8> | | | | | | | | 00000 0000 | 0000 0000 |
| 591h | DAC2CON0 | EN | FM | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 0000 0000 | 0000 0000 |
| 592h | DAC2REFL | REF<7:0> | | | | | | | | 00000 0000 | 0000 0000 |
| 593h | DAC2REFH | REF<15:8> | | | | | | | | 00000 0000 | 0000 0000 |
| 594h | DAC3CON0 | EN | — | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 0-00 0000 | 0-00 0000 |
| 595h | DAC3REF | — | — | — | REF<4:0> | | | | | ---0 0000 | ---0 0000 |
| 596h | DAC4CON0 | EN | — | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 0-00 0000 | 0-00 0000 |
| 597h | DAC4REF | — | — | — | REF<4:0> | | | | | ---0 0000 | 0000 0000 |
| 598h | DAC5CON0 | EN | FM | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 0000 0000 | 0000 0000 |
| 599h | DAC5REFL | REF<7:0> | | | | | | | | 00000 0000 | 0000 0000 |
| 59Ah | DAC5REFH | REF<15:8> | | | | | | | | 00000 0000 | 0000 0000 |
| 59Bh to 59Dh | — | Unimplemented | | | | | | | | — | — |
| 59Eh | DAC7CON0 | EN | — | OE1 | OE2 | PSS<1:0> | | NSS1 | NSS0 | 0-00 0000 | 0-00 0000 |
| 59Fh | DAC7REF | — | — | — | REF<4:0> | | | | | ---0 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:** Unimplemented, read as '1'.
**2:** Unimplemented on PIC16LF1776.

**TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 12** | | | | | | | | | | | |
| 60Ch to 613h | — | Unimplemented | | | | | | | | — | — |
| 614h | PWM3DCL | DC<1:0> | | — | — | — | — | — | — | xx-- ---- | uu-- ---- |
| 615h | PWM3DCH | DC<9:2> | | | | | | | | xxxx xxxx | uuuu uuuu |
| 616h | PWM3CON | EN | — | OUT | POL | — | — | — | — | 0-00 ---- | 0-00 ---- |
| 617h | PWM4DCL | DC<1:0> | | — | — | — | — | — | — | xx-- ---- | uu-- ---- |
| 618h | PWM4DCH | DC<9:2> | | | | | | | | xxxx xxxx | uuuu uuuu |
| 619h | PWM4CON | EN | — | OUT | POL | — | — | — | — | 0-00 ---- | 0-00 ---- |
| 61Ah | PWM9DCL | DC<1:0> | | — | — | — | — | — | — | xx-- ---- | uu-- ---- |
| 61Bh | PWM9DCH | DC<9:2> | | | | | | | | xxxx xxxx | uuuu uuuu |
| 61Ch | PWM9CON | EN | — | OUT | POL | — | — | — | — | 0-00 ---- | 0-00 ---- |
| 61Dh — 61Fh | — | 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:** Unimplemented, read as '1'.
**2:** Unimplemented on PIC16LF1776.

**TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 13** | | | | | | | | | | | |
| 68Ch | — | Unimplemented | | | | | | | | — | — |
| 68Dh | COG1PHR | — | — | COG Rising Edge Phase Delay Count Register | | | | | | --00 0000 | --00 0000 |
| 68Eh | COG1PHF | — | — | COG Falling Edge Phase Delay Count Register | | | | | | --00 0000 | --00 0000 |
| 68Fh | COG1BLKR | — | — | COG Rising Edge Blanking Count Register | | | | | | --00 0000 | --00 0000 |
| 690h | COG1BLKF | — | — | COG Falling Edge Blanking Count Register | | | | | | --00 0000 | --00 0000 |
| 691h | COG1DBR | — | — | COG Rising Edge Dead-band Count Register | | | | | | --00 0000 | --00 0000 |
| 692h | COG1DBF | — | — | COG Falling Edge Dead-band Count Register | | | | | | --00 0000 | --00 0000 |
| 693h | COG1CON0 | EN | LD | — | CS<1:0> | | MD<2:0> | | | 00-0 0000 | 00-0 0000 |
| 694h | COG1CON1 | RDBS | FDBS | — | — | POLD | POLC | POLB | POLA | 00-- 0000 | 00-- 0000 |
| 695h | COG1RIS0 | RIS7 | RIS6 | RIS5 | RIS4 | RIS3 | RIS2 | RIS1 | RIS0 | 0000 0000 | 0000 0000 |
| 696h | COG1RIS1 | — | — | RIS13 | RIS12 | RIS11 | RIS10 | RIS9 | RIS8 | --00 0000 | --00 0000 |
| 697h | COG1RSIM0 | RSIM7 | RSIM6 | RSIM5 | RSIM4 | RSIM3 | RSIM2 | RSIM1 | RSIM0 | 0000 0000 | 0000 0000 |
| 698h | COG1RSIM1 | — | — | RSIM13 | RSIM12 | RSIM11 | RSIM10 | RSIM9 | RSIM8 | --00 0000 | --00 0000 |
| 699h | COG1FIS0 | FIS7 | FIS6 | FIS5 | FIS4 | FIS3 | FIS2 | FIS1 | FIS0 | 0000 0000 | 0000 0000 |
| 69Ah | COG1FIS1 | — | — | FIS13 | FIS12 | FIS11 | FIS10 | FIS9 | FIS8 | --00 0000 | --00 0000 |
| 69Bh | COG1FSIM0 | FSIM7 | FSIM6 | FSIM5 | FSIM4 | FSIM3 | FSIM2 | FSIM1 | FSIM0 | 0000 0000 | 0000 0000 |
| 69Ch | COG1FSIM1 | — | — | FSIM13 | FSIM12 | FSIM11 | FSIM10 | FSIM9 | FSIM8 | --00 0000 | --00 0000 |
| 69Dh | COG1ASD0 | ASE | ARSEN | ASDBD<1:0> | | ASDAC<1:0> | | — | — | 0001 01-- | 0001 01-- |
| 69Eh | COG1ASD1 | AS7E | AS6E | AS5E | AS4E | AS3E | AS2E | AS1E | AS0E | 0000 0000 | 0000 0000 |
| 69Fh | COG1STR | SDATD | SDATC | SDATB | SDATA | STRD | STRC | STRB | STRA | 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:** Unimplemented, read as '1'.

**2:** Unimplemented on PIC16LF1776.

**TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 14** | | | | | | | | | | | |
| 70Ch | — | Unimplemented | | | | | | | | — | — |
| 70Dh | COG2PHR | — | — | COG Rising Edge Phase Delay Count Register | | | | | | --00 0000 | --00 0000 |
| 70Eh | COG2PHF | — | — | COG Falling Edge Phase Delay Count Register | | | | | | --00 0000 | --00 0000 |
| 70Fh | COG2BLKR | — | — | COG Rising Edge Blanking Count Register | | | | | | --00 0000 | --00 0000 |
| 710h | COG2BLKF | — | — | COG Falling Edge Blanking Count Register | | | | | | --00 0000 | --00 0000 |
| 711h | COG2DBR | — | — | COG Rising Edge Dead-band Count Register | | | | | | --00 0000 | --00 0000 |
| 712h | COG2DBF | — | — | COG Falling Edge Dead-band Count Register | | | | | | --00 0000 | --00 0000 |
| 713h | COG2CON0 | EN | LD | — | CS<1:0> | | MD<2:0> | | | 00-0 0000 | 00-0 0000 |
| 714h | COG2CON1 | RDBS | FDBS | — | — | POLD | POLC | POLB | POLA | 00-- 0000 | 00-- 0000 |
| 715h | COG2RIS0 | RIS7 | RIS6 | RIS5 | RIS4 | RIS3 | RIS2 | RIS1 | RIS0 | 0000 0000 | 0000 0000 |
| 716h | COG2RIS1 | — | — | RIS13 | RIS12 | RIS11 | RIS10 | RIS9 | RIS8 | --00 0000 | --00 0000 |
| 717h | COG2RSIM0 | RSIM7 | RSIM6 | RSIM5 | RSIM4 | RSIM3 | RSIM2 | RSIM1 | RSIM0 | 0000 0000 | 0000 0000 |
| 718h | COG2RSIM1 | — | — | RSIM13 | RSIM12 | RSIM11 | RSIM10 | RSIM9 | RSIM8 | --00 0000 | --00 0000 |
| 719h | COG2FIS0 | FIS7 | FIS6 | FIS5 | FIS4 | FIS3 | FIS2 | FIS1 | FIS0 | 0000 0000 | 0000 0000 |
| 71Ah | COG2FIS1 | — | — | FIS13 | FIS12 | FIS11 | FIS10 | FIS9 | FIS8 | --00 0000 | --00 0000 |
| 71Bh | COG2FSIM0 | FSIM7 | FSIM6 | FSIM5 | FSIM4 | FSIM3 | FSIM2 | FSIM1 | FSIM0 | 0000 0000 | 0000 0000 |
| 71Ch | COG2FSIM1 | — | — | FSIM13 | FSIM12 | FSIM11 | FSIM10 | FSIM9 | FSIM8 | --00 0000 | --00 0000 |
| 71Dh | COG2ASD0 | ASE | ARSEN | ASDBD<1:0> | | ASDAC<1:0> | | — | — | 0001 01-- | 0001 01-- |
| 71Eh | COG2ASD1 | AS7E | AS6E | AS5E | AS4E | AS3E | AS2E | AS1E | AS0E | 0000 0000 | 0000 0000 |
| 71Fh | COG2STR | SDATD | SDATC | SDATB | SDATA | STRD | STRC | STRB | STRA | 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:** Unimplemented, read as '1'.

**2:** Unimplemented on PIC16LF1776.

**TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 15** | | | | | | | | | | | |
| 78Ch — 78Dh | — | Unimplemented | | | | | | | | — | — |
| 78Eh | PRG1RTSS | — | — | — | — | RTSS<3:0> | | | | ---- 0000 | ---- 0000 |
| 78Fh | PRG1FTSS | — | — | — | — | FTSS<3:0> | | | | ---- 0000 | ---- 0000 |
| 790h | PRG1INS | — | — | — | — | INS<3:0> | | | | ---- 0000 | ---- 0000 |
| 791h | PRG1CON0 | EN | — | FEDG | REDG | MODE<1:0> | | OS | GO | 0-000 0000 | 0-00 0000 |
| 792h | PRG1CON1 | — | — | — | — | — | RDY | FPOL | RPOL | ---- -000 | ---- -000 |
| 793h | PRG1CON2 | — | — | — | ISET<4:0> | | | | | ---0 0000 | ---0 0000 |
| 794h | PRG2RTSS | — | — | — | — | RTSS<3:0> | | | | ---- 0000 | ---- 0000 |
| 795h | PRG2FTSS | — | — | — | — | FTSS<3:0> | | | | ---- 0000 | ---- 0000 |
| 796h | PRG2INS | — | — | — | — | INS<3:0> | | | | ---- 0000 | ---- 0000 |
| 797h | PRG2CON0 | EN | — | FEDG | REDG | MODE<1:0> | | OS | GO | 0-000 0000 | 0-00 0000 |
| 798h | PRG2CON1 | — | — | — | — | — | RDY | FPOL | RPOL | ---- -000 | ---- -000 |
| 799h | PRG2CON2 | — | — | — | ISET<4:0> | | | | | ---0 0000 | ---0 0000 |
| 79Ah | PRG3RTSS | — | — | — | — | RTSS<3:0> | | | | ---- 0000 | ---- 0000 |
| 79Bh | PRG3FTSS | — | — | — | — | FTSS<3:0> | | | | ---- 0000 | ---- 0000 |
| 79Ch | PRG3INS | — | — | — | — | INS<3:0> | | | | ---- 0000 | ---- 0000 |
| 79Dh | PRG3CON0 | EN | — | FEDG | REDG | MODE<1:0> | | OS | GO | 0-000 0000 | 0-00 0000 |
| 79Eh | PRG3CON1 | — | — | — | — | — | RDY | FPOL | RPOL | ---- -000 | ---- -000 |
| 79Fh | PRG3CON2 | — | — | — | ISET<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:** Unimplemented, read as '1'.

**2:** Unimplemented on PIC16LF1776.

**TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 16** | | | | | | | | | | | |
| 80Ch | — | Unimplemented | | | | | | | | — | — |
| 80Dh | COG3PHR | — | — | COG Rising Edge Phase Delay Count Register | | | | | | --00 0000 | --00 0000 |
| 80Eh | COG3PHF | — | — | COG Falling Edge Phase Delay Count Register | | | | | | --00 0000 | --00 0000 |
| 80Fh | COG3BLKR | — | — | COG Rising Edge Blanking Count Register | | | | | | --00 0000 | --00 0000 |
| 810h | COG3BLKF | — | — | COG Falling Edge Blanking Count Register | | | | | | --00 0000 | --00 0000 |
| 811h | COG3DBR | — | — | COG Rising Edge Dead-band Count Register | | | | | | --00 0000 | --00 0000 |
| 812h | COG3DBF | — | — | COG Falling Edge Dead-band Count Register | | | | | | --00 0000 | --00 0000 |
| 813h | COG3CON0 | EN | LD | — | CS<1:0> | | MD<2:0> | | | 00-0 0000 | 00-0 0000 |
| 814h | COG3CON1 | RDBS | FDBS | — | — | POLD | POLC | POLB | POLA | 00-- 0000 | 00-- 0000 |
| 815h | COG3RIS0 | RIS7 | RIS6 | RIS5 | RIS4 | RIS3 | RIS2 | RIS1 | RIS0 | 0000 0000 | 0000 0000 |
| 816h | COG3RIS1 | RIS15 | RIS14 | RIS13 | RIS12 | RIS11 | RIS10 | RIS9 | RIS8 | 0000 0000 | 0000 0000 |
| 817h | COG3RSIM0 | RSIM7 | RSIM6 | RSIM5 | RSIM4 | RSIM3 | RSIM2 | RSIM1 | RSIM0 | 0000 0000 | 0000 0000 |
| 818h | COG3RSIM1 | RSIM15 | RSIM14 | RSIM13 | RSIM12 | RSIM11 | RSIM10 | RSIM9 | RSIM8 | 0000 0000 | 0000 0000 |
| 819h | COG3FIS0 | FIS7 | FIS6 | FIS5 | FIS4 | FIS3 | FIS2 | FIS1 | FIS0 | 0000 0000 | 0000 0000 |
| 81Ah | COG3FIS1 | FIS15 | FIS14 | FIS13 | FIS12 | FIS11 | FIS10 | FIS9 | FIS8 | 0000 0000 | 0000 0000 |
| 81Bh | COG3FSIM0 | FSIM7 | FSIM6 | FSIM5 | FSIM4 | FSIM3 | FSIM2 | FSIM1 | FSIM0 | 0000 0000 | 0000 0000 |
| 81Ch | COG3FSIM1 | FSIM15 | FSIM14 | FSIM13 | FSIM12 | FSIM11 | FSIM10 | FSIM9 | FSIM8 | 0000 0000 | 0000 0000 |
| 81Dh | COG3ASD0 | ASE | ARSEN | ASDBD<1:0> | | ASDAC<1:0> | | — | — | 0001 01-- | 0001 01-- |
| 81Eh | COG3ASD1 | AS7E | AS6E | AS5E | AS4E | AS3E | AS2E | AS1E | AS0E | 0000 0000 | 0000 0000 |
| 81Fh | COG3STR | SDATD | SDATC | SDATB | SDATA | STRD | STRC | STRB | STRA | 0000 0000 | 0000 0000 |
| **Bank 17** | | | | | | | | | | | |
| 880h — 89Fh | — | 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:** Unimplemented, read as '1'.
**2:** Unimplemented on PIC16LF1776.

**TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 18** | | | | | | | | | | | |
| 90Ch | CM4CON0 | ON | OUT | — | POL | ZLF | Reserved | HYS | SYNC | 00-0 0100 | 00-0 0100 |
| 90Dh | CM4CON1 | — | — | — | — | — | — | INTP | INTN | ---- --00 | ---- --00 |
| 90Eh | CM4NSEL | — | — | — | — | NCH<3:0> | | | | ---- 0000 | ---- 0000 |
| 90Fh | CM4PSEL | — | — | — | — | PCH<3:0> | | | | ---- 0000 | ---- 0000 |
| 910h | CM5CON0 | ON | OUT | — | POL | ZLF | Reserved | HYS | SYNC | 00-0 0100 | 00-0 0100 |
| 911h | CM5CON1 | — | — | — | — | — | — | INTP | INTN | ---- --00 | ---- --00 |
| 912h | CM5NSEL | — | — | — | — | NCH<3:0> | | | | ---- 0000 | ---- 0000 |
| 913h | CM5PSEL | — | — | — | — | PCH<3:0> | | | | ---- 0000 | ---- 0000 |
| 914h | CM6CON0 | ON | OUT | — | POL | ZLF | Reserved | HYS | SYNC | 00-0 0100 | 00-0 0100 |
| 915h | CM6CON1 | — | — | — | — | — | — | INTP | INTN | ---- --00 | ---- --00 |
| 916h | CM6NSEL | — | — | — | — | NCH<3:0> | | | | ---- 0000 | ---- 0000 |
| 917h | CM6PSEL | — | — | — | — | PCH<3:0> | | | | ---- 0000 | ---- 0000 |
| 918h — 91Fh | — | Unimplemented | | | | | | | | — | — |
| **Bank 19-26** | | | | | | | | | | | |
| 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:** Unimplemented, read as '1'.

**2:** Unimplemented on PIC16LF1776.

**TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 27** | | | | | | | | | | | |
| D8Ch | — | Unimplemented | | | | | | | | — | — |
| D8Dh | — | Unimplemented | | | | | | | | — | — |
| D8Eh | PWMEN | — | — | — | — | — | MPWM11EN | MPWM6EN | MPWM5EN | ---- -000 | ---- -000 |
| D8Fh | PWMLD | — | — | — | — | — | MPWM11LD | MPWM6LD | MPWM5LD | ---- -000 | ---- -000 |
| D90h | PWMOUT | — | — | — | — | — | MPWM11OUT | MPWM6OUT | MPWM5OUT | ---- -000 | ---- -000 |
| D91h | PWM5PHL | PH<7:0> | | | | | | | | xxxx xxxx | uuuu uuuu |
| D92h | PWM5PHH | PH<15:8> | | | | | | | | xxxx xxxx | uuuu uuuu |
| D93h | PWM5DCL | DC<7:0> | | | | | | | | xxxx xxxx | uuuu uuuu |
| D94h | PWM5DCH | DC<15:8> | | | | | | | | xxxx xxxx | uuuu uuuu |
| D95h | PWM5PRL | PR<7:0> | | | | | | | | xxxx xxxx | uuuu uuuu |
| D96h | PWM5PRH | PR<15:8> | | | | | | | | xxxx xxxx | uuuu uuuu |
| D97h | PWM5OFL | OF<7:0> | | | | | | | | xxxx xxxx | uuuu uuuu |
| D98h | PWM5OFH | OF<15:8> | | | | | | | | xxxx xxxx | uuuu uuuu |
| D99h | PWM5TMRL | TMR<7:0> | | | | | | | | 0000 0000 | 0000 0000 |
| D9Ah | PWM5TMRH | TMR<15:8> | | | | | | | | 0000 0000 | 0000 0000 |
| D9Bh | PWM5CON | EN | — | OUT | POL | MODE<1:0> | | — | — | 0-00 00-- | 0-00 00-- |
| D9Ch | PWM5INTE | — | — | — | — | OFIE | PHIE | DCIE | PRIE | ---- 0000 | ---- 0000 |
| D9Dh | PWM5INTF | — | — | — | — | OFIF | PHIF | DCIF | PRIF | ---- 0000 | ---- 0000 |
| D9Eh | PWM5CLKCON | — | PS<2:0> | | | — | — | CS<1:0> | | -000 --00 | -000 --00 |
| D9Fh | PWM5LDCON | LDA | LDT | — | — | — | — | LDS<1:0> | | 00-- --00 | 00-- --00 |
| DA0h | PWM5OFCON | — | OFM<1:0> | | OFO | — | — | OFS<1:0> | | -000 --00 | -000 --00 |
| DA1h | PWM6PHL | PH<7:0> | | | | | | | | xxxx xxxx | uuuu uuuu |
| DA2h | PWM6PHH | PH<15:8> | | | | | | | | xxxx xxxx | uuuu uuuu |
| DA3h | PWM6DCL | DC<7:0> | | | | | | | | xxxx xxxx | uuuu uuuu |
| DA4h | PWM6DCH | DC<15:8> | | | | | | | | xxxx xxxx | uuuu uuuu |
| DA5h | PWM6PRL | PR<7:0> | | | | | | | | xxxx xxxx | uuuu uuuu |
| DA6h | PWM6PRH | PR<15:8> | | | | | | | | xxxx xxxx | uuuu uuuu |
| DA7h | PWM6OFL | OF<7:0> | | | | | | | | xxxx xxxx | uuuu uuuu |
| DA8h | PWM6OFH | OF<15:8> | | | | | | | | xxxx xxxx | uuuu uuuu |
| DA9h | PWM6TMRL | TMR<7:0> | | | | | | | | 0000 0000 | 0000 0000 |
| DAAh | PWM6TMRH | TMR<15:8> | | | | | | | | 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:** Unimplemented, read as '1'.

**2:** Unimplemented on PIC16LF1776.

## TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 27 (Continued)** | | | | | | | | | | | |
| DABh | PWM6CON | EN | — | OUT | POL | MODE<1:0> | | — | — | 0-00 00-- | 0-00 00-- |
| DACh | PWM6INTE | — | — | — | — | OFIE | PHIE | DCIE | PRIE | ---- 0000 | ---- 0000 |
| DADh | PWM6INTF | — | — | — | — | OFIF | PHIF | DCIF | PRIF | ---- 0000 | ---- 0000 |
| DAEh | PWM6CLKCON | — | PS<2:0> | | | — | — | CS<1:0> | | -000 --00 | -000 --00 |
| DAFh | PWM6LDCON | LDA | LDT | — | — | — | — | LDS<1:0> | | 00-- --00 | 00-- --00 |
| DB0h | PWM6OFCON | — | OFM<1:0> | | OFO | — | — | OFS<1:0> | | -000 --00 | -000 --00 |
| DB1h | PWM11PHL | PH<7:0> | | | | | | | | xxxx xxxx | uuuu uuuu |
| DB2h | PWM11PHH | PH<15:8> | | | | | | | | xxxx xxxx | uuuu uuuu |
| DB3h | PWM11DCL | DC<7:0> | | | | | | | | xxxx xxxx | uuuu uuuu |
| DB4h | PWM11DCH | DC<15:8> | | | | | | | | xxxx xxxx | uuuu uuuu |
| DB5h | PWM11PRL | PR<7:0> | | | | | | | | xxxx xxxx | uuuu uuuu |
| DB6h | PWM11PRH | PR<15:8> | | | | | | | | xxxx xxxx | uuuu uuuu |
| DB7h | PWM11OFL | OF<7:0> | | | | | | | | xxxx xxxx | uuuu uuuu |
| DB8h | PWM11OFH | OF<15:8> | | | | | | | | xxxx xxxx | uuuu uuuu |
| DB9h | PWM11TMRL | TMR<7:0> | | | | | | | | 0000 0000 | 0000 0000 |
| DBAh | PWM11TMRH | TMR<15:8> | | | | | | | | 0000 0000 | 0000 0000 |
| DBBh | PWM11CON | EN | — | OUT | POL | MODE<1:0> | | — | — | 0-00 00-- | 0-00 00-- |
| DBCh | PWM11INTE | — | — | — | — | OFIE | PHIE | DCIE | PRIE | ---- 0000 | ---- 0000 |
| DBDh | PWM11INTF | — | — | — | — | OFIF | PHIF | DCIF | PRIF | ---- 0000 | ---- 0000 |
| DBEh | PWM11CLKCON | — | PS<2:0> | | | — | — | CS<1:0> | | -000 --00 | -000 --00 |
| DBFh | PWM11LDCON | LDA | LDT | — | — | — | — | LDS<1:0> | | 00-- --00 | 00-- --00 |
| DC0h | PWM11OFCON | — | OFM<1:0> | | OFO | — | — | OFS<1:0> | | -000 --00 | -000 --00 |
| DC1h to DEFh | — | 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:** Unimplemented, read as '1'.
**2:** Unimplemented on PIC16LF1776.

**TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 28** | | | | | | | | | | | |
| E0Ch — E0Bh | — | Unimplemented | | | | | | | | — | — |
| E0Ch | PPSLOCK | — | — | — | — | — | — | — | PPSLOCKED | ---- ---0 | ---- ---0 |
| E0Dh | INTPPS | — | — | INTPPS<5:0> | | | | | | --00 1000 | --uu uuuu |
| E0Eh | T0CKIPPS | — | — | T0CKIPPS<5:0> | | | | | | --00 0100 | --uu uuuu |
| E0Fh | T1CKIPPS | — | — | T1CKIPPS<5:0> | | | | | | --01 0000 | --uu uuuu |
| E10h | T1GPPS | — | — | T1GPPS<5:0> | | | | | | --00 1101 | --uu uuuu |
| E11h | T3CKIPPS | — | — | T3CKIPPS<5:0> | | | | | | --01 0000 | --uu uuuu |
| E12h | T3GPPS | — | — | T3GPPS<5:0> | | | | | | --01 0000 | --uu uuuu |
| E13h | T5CKIPPS | — | — | T5CKIPPS<5:0> | | | | | | --01 0000 | --uu uuuu |
| E14h | T5GPPS | — | — | T5GPPS<5:0> | | | | | | --00 1100 | --uu uuuu |
| E15h | T2INPPS | — | — | T2INPPS<5:0> | | | | | | --01 0011 | --uu uuuu |
| E16h | T4INPPS | — | — | T4INPPS<5:0> | | | | | | --01 0101 | --uu uuuu |
| E17h | T6INPPS | — | — | T6INPPS<5:0> | | | | | | --01 0100 | --uu uuuu |
| E18h | T8INPPS | — | — | T8INPPS<5:0> | | | | | | --01 0010 | --uu uuuu |
| E19h | CCP1PPS | — | — | CCP1PPS<5:0> | | | | | | --01 0010 | --uu uuuu |
| E1Ah | CCP2PPS | — | — | CCP2PPS<5:0> | | | | | | --01 0001 | --uu uuuu |
| E1Bh | CCP7PPS | — | — | CCP7PPS<5:0> | | | | | | --00 1101 | --uu uuuu |
| E1Ch | — | Unimplemented | | | | | | | | — | — |
| E1Dh | COGIN1PPS | — | — | COG1PPS<5:0> | | | | | | --00 1000 | --uu uuuu |
| E1Eh | COG2INPPS | — | — | COG2PPS<5:0> | | | | | | --00 1001 | --uu uuuu |
| E1Fh | COG3INPPS | — | — | COG3PPS<5:0> | | | | | | --00 1010 | --uu uuuu |
| E20h | — | Unimplemented | | | | | | | | — | — |
| E21h | MD1CLPPS | — | — | MD1CLPPS<5:0> | | | | | | --00 0100 | --uu uuuu |
| E22h | MD1CHPPS | — | — | MD1CHPPS<5:0> | | | | | | --00 0011 | --uu uuuu |
| E23h | MD1MODPPS | — | — | MD1MODPPS<5:0> | | | | | | --00 0101 | --uu uuuu |
| E24h | MD2CLPPS | — | — | MD2CLPPS<5:0> | | | | | | --00 0100 | --uu uuuu |
| E25h | MD2CHPPS | — | — | MD2CHPPS<5:0> | | | | | | --00 0011 | --uu uuuu |
| E26h | MD2MODPPS | — | — | MD2MODPPS<5:0> | | | | | | --00 0101 | --uu 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:** Unimplemented, read as '1'.

**2:** Unimplemented on PIC16LF1776.

## TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 28 (Continued)** | | | | | | | | | | | |
| E27h | MD3CLPPS | — | — | MD3CLPPS<5:0> | | | | | | --00 1100 | --uu uuuu |
| E28h | MD3CHPPS | — | — | MD3CHPPS<5:0> | | | | | | --00 1011 | --uu uuuu |
| E29h | MD3MODPPS | — | — | MD3MODPPS<5:0> | | | | | | --00 0101 | --uu uuuu |
| E2Ah — E2Ch | — | Unimplemented | | | | | | | | — | — |
| E2Dh | PRG1RPPS | — | — | PRG1RPPS<5:0> | | | | | | --00 0100 | --uu uuuu |
| E2Eh | PRG1FPPS | — | — | PRG1FPPS<5:0> | | | | | | --00 0101 | --uu uuuu |
| E2Fh | PRG2RPPS | — | — | PRG2RPPS<5:0> | | | | | | --01 0001 | --uu uuuu |
| E30h | PRG2FPPS | — | — | PRG2FPPS<5:0> | | | | | | --01 0010 | --uu uuuu |
| E31h | PRG3RPPS | — | — | PRG3RPPS<5:0> | | | | | | --01 0100 | --uu uuuu |
| E32h | PRG3FPPS | — | — | PRG3FPPS<5:0> | | | | | | --01 0101 | --uu uuuu |
| E33h | — | Unimplemented | | | | | | | | — | — |
| E34h | — | Unimplemented | | | | | | | | — | — |
| E35h | CLC1IN0PPS | — | — | CLCIN0PPS<5:0> | | | | | | --00 0000 | --uu uuuu |
| E36h | CLC1IN1PPS | — | — | CLCIN1PPS<5:0> | | | | | | --00 0001 | --uu uuuu |
| E37h | CLC1IN2PPS | — | — | CLCIN2PPS<5:0> | | | | | | --00 1110 | --uu uuuu |
| E38h | CLC1IN3PPS | — | — | CLCIN3PPS<5:0> | | | | | | --00 1111 | --uu uuuu |
| E39h | ADCACTPPS | — | — | ADCACTPPS<5:0> | | | | | | --00 1100 | --uu uuuu |
| E3Ah | SSPCLKPPS | — | — | SSPCLKPPS<5:0> | | | | | | --01 0011 | --uu uuuu |
| E3Bh | SSPDATPPS | — | — | SSPDATPPS<5:0> | | | | | | --01 0100 | --uu uuuu |
| E3Ch | SSPSSPPS | — | — | SSPSSPPS<5:0> | | | | | | --00 0101 | --uu uuuu |
| E3Dh | RXPPS | — | — | RXPPS<5:0> | | | | | | --01 0111 | --uu uuuu |
| E3Eh | CKPPS | — | — | CKPPS<5:0> | | | | | | --01 0110 | --uu uuuu |
| E3Fh — E6Fh | — | 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:** Unimplemented, read as '1'.

**2:** Unimplemented on PIC16LF1776.

**TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 29** | | | | | | | | | | | |
| E8Ch — E8Fh | — | Unimplemented | | | | | | | | — | — |
| E90h | RA0PPS | — | — | RA0PPS<4:0> | | | | | | --00 0000 | --uu uuuu |
| E91h | RA1PPS | — | — | RA1PPS<4:0> | | | | | | --00 0000 | --uu uuuu |
| E92h | RA2PPS | — | — | RA2PPS<4:0> | | | | | | --00 0000 | --uu uuuu |
| E93h | RA3PPS | — | — | RA3PPS<4:0> | | | | | | --00 0000 | --uu uuuu |
| E94h | RA4PPS | — | — | RA4PPS<4:0> | | | | | | --00 0000 | --uu uuuu |
| E95h | RA5PPS | — | — | RA5PPS<4:0> | | | | | | --00 0000 | --uu uuuu |
| E96h | RA6PPS | — | — | RA6PPS<4:0> | | | | | | --00 0000 | --uu uuuu |
| E97h | RA7PPS | — | — | RA7PPS<4:0> | | | | | | --00 0000 | --uu uuuu |
| E98h | RB0PPS | — | — | RB0PPS<5:0> | | | | | | --00 0000 | --uu uuuu |
| E99h | RB1PPS | — | — | RB1PPS<5:0> | | | | | | --00 0000 | --uu uuuu |
| E9Ah | RB2PPS | — | — | RB2PPS<5:0> | | | | | | --00 0000 | --uu uuuu |
| E9Bh | RB3PPS | — | — | RB3PPS<5:0> | | | | | | --00 0000 | --uu uuuu |
| E9Ch | RB4PPS | — | — | RB4PPS<5:0> | | | | | | --00 0000 | --uu uuuu |
| E9Dh | RB5PPS | — | — | RB5PPS<5:0> | | | | | | --00 0000 | --uu uuuu |
| E9Eh | RB6PPS | — | — | RB6PPS<5:0> | | | | | | --00 0000 | --uu uuuu |
| E9Fh | RB7PPS | — | — | RB7PPS<5:0> | | | | | | --00 0000 | --uu uuuu |
| EA0h | RC0PPS | — | — | RC0PPS<5:0> | | | | | | --00 0000 | --uu uuuu |
| EA1h | RC1PPS | — | — | RC1PPS<5:0> | | | | | | --00 0000 | --uu uuuu |
| EA2h | RC2PPS | — | — | RC2PPS<5:0> | | | | | | --00 0000 | --uu uuuu |
| EA3h | RC3PPS | — | — | RC3PPS<5:0> | | | | | | --00 0000 | --uu uuuu |
| EA4h | RC4PPS | — | — | RC4PPS<5:0> | | | | | | --00 0000 | --uu uuuu |
| EA5h | RC5PPS | — | — | RC5PPS<5:0> | | | | | | --00 0000 | --uu uuuu |
| EA6h | RC6PPS | — | — | RC6PPS<5:0> | | | | | | --00 0000 | --uu uuuu |
| EA7h | RC7PPS | — | — | RC7PPS<5:0> | | | | | | --00 0000 | --uu uuuu |
| EA8h — EEFh | — | 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:** Unimplemented, read as '1'.
**2:** Unimplemented on PIC16LF1776.

**TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 30** | | | | | | | | | | | |
| F0Ch — F0Eh | — | Unimplemented | | | | | | | | — | — |
| F0Fh | CLCDATA | — | — | — | — | MLC4OUT | MLC3OUT | MLC2OUT | MLC1OUT | ---- x000 | ---- u000 |
| F10h | CLC1CON | EN | — | OUT | INTP | INTN | MODE<2:0> | | | 0-00 0000 | 0-00 0000 |
| F11h | CLC1POL | POL | — | — | — | G4POL | G3POL | G2POL | G1POL | 0--- xxxx | 0--- uuuu |
| F12h | CLC1SEL0 | — | — | D1S<5:0> | | | | | | --xx xxxx | --uu uuuu |
| F13h | CLC1SEL1 | — | — | D2S<5:0> | | | | | | --xx xxxx | --uu uuuu |
| F14h | CLC1SEL2 | — | — | D3S<5:0> | | | | | | --xx xxxx | --uu uuuu |
| F15h | CLC1SEL3 | — | — | D4S<5:0> | | | | | | --xx xxxx | --uu uuuu |
| F16h | CLC1GLS0 | G1D4T | G1D4N | G1D3T | G1D3N | G1D2T | G1D2N | G1D1T | G1D1N | xxxx xxxx | uuuu uuuu |
| F17h | CLC1GLS1 | G2D4T | G2D4N | G2D3T | G2D3N | G2D2T | G2D2N | G2D1T | G2D1N | xxxx xxxx | uuuu uuuu |
| F18h | CLC1GLS2 | G3D4T | G3D4N | G3D3T | G3D3N | G3D2T | G3D2N | G3D1T | G3D1N | xxxx xxxx | uuuu uuuu |
| F19h | CLC1GLS3 | G4D4T | G4D4N | G4D3T | G4D3N | G4D2T | G4D2N | G4D1T | G4D1N | xxxx xxxx | uuuu uuuu |
| F1Ah | CLC2CON | EN | — | OUT | INTP | INTN | MODE<2:0> | | | 0-00 0000 | 0-00 0000 |
| F1Bh | CLC2POL | POL | — | — | — | G4POL | G3POL | G2POL | G1POL | 0--- xxxx | 0--- uuuu |
| F1Ch | CLC2SEL0 | — | — | D1S<5:0> | | | | | | --xx xxxx | --uu uuuu |
| F1Dh | CLC2SEL1 | — | — | D2S<5:0> | | | | | | --xx xxxx | --uu uuuu |
| F1Eh | CLC2SEL2 | — | — | D3S<5:0> | | | | | | --xx xxxx | --uu uuuu |
| F1Fh | CLC2SEL3 | — | — | D4S<5:0> | | | | | | --xx xxxx | --uu uuuu |
| F20h | CLC2GLS0 | G1D4T | G1D4N | G1D3T | G1D3N | G1D2T | G1D2N | G1D1T | G1D1N | xxxx xxxx | uuuu uuuu |
| F21h | CLC2GLS1 | G2D4T | G2D4N | G2D3T | G2D3N | G2D2T | G2D2N | G2D1T | G2D1N | xxxx xxxx | uuuu uuuu |
| F22h | CLC2GLS2 | G3D4T | G3D4N | G3D3T | G3D3N | G3D2T | G3D2N | G3D1T | G3D1N | xxxx xxxx | uuuu uuuu |
| F23h | CLC2GLS3 | G4D4T | G4D4N | G4D3T | G4D3N | G4D2T | G4D2N | G4D1T | G4D1N | xxxx xxxx | uuuu uuuu |
| F24h | CLC3CON | EN | — | OUT | INTP | INTN | MODE<2:0> | | | 0-00 0000 | 0-00 0000 |
| F25h | CLC3POL | POL | — | — | — | G4POL | G3POL | G2POL | G1POL | 0--- xxxx | 0--- uuuu |
| F26h | CLC3SEL0 | — | — | D1S<5:0> | | | | | | --xx xxxx | --uu uuuu |
| F27h | CLC3SEL1 | — | — | D2S<5:0> | | | | | | --xx xxxx | --uu uuuu |
| F28h | CLC3SEL2 | — | — | D3S<5:0> | | | | | | --xx xxxx | --uu uuuu |
| F29h | CLC3SEL3 | — | — | D4S<5:0> | | | | | | --xx xxxx | --uu uuuu |
| F2Ah | CLC3GLS0 | G1D4T | G1D4N | G1D3T | G1D3N | G1D2T | G1D2N | G1D1T | G1D1N | 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:** Unimplemented, read as '1'.

**2:** Unimplemented on PIC16LF1776.

**TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)**

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 30 (Continued)** | | | | | | | | | | | |
| F2Bh | CLC3GLS1 | G2D4T | G2D4N | G2D3T | G2D3N | G2D2T | G2D2N | G2D1T | G2D1N | xxxx xxxx | uuuu uuuu |
| F2Ch | CLC3GLS2 | G3D4T | G3D4N | G3D3T | G3D3N | G3D2T | G3D2N | G3D1T | G3D1N | xxxx xxxx | uuuu uuuu |
| F2Dh | CLC3GLS3 | G4D4T | G4D4N | G4D3T | G4D3N | G4D2T | G4D2N | G4D1T | G4D1N | xxxx xxxx | uuuu uuuu |
| F2Eh | CLC4CON | EN | OE | OUT | INTP | INTN | MODE<2:0> | | | 0000 0000 | 0000 0000 |
| F2Fh | CLC4POL | POL | — | — | — | G4POL | G3POL | G2POL | G1POL | 0--- xxxx | 0--- uuuu |
| F30h | CLC4SEL0 | D1S<7:0> | | | | | | | | xxxx xxxx | uuuu uuuu |
| F31h | CLC4SEL1 | D2S<7:0> | | | | | | | | xxxx xxxx | uuuu uuuu |
| F32h | CLC4SEL2 | D3S<7:0> | | | | | | | | xxxx xxxx | uuuu uuuu |
| F33h | CLC4SEL3 | D4S<7:0> | | | | | | | | xxxx xxxx | uuuu uuuu |
| F34h | CLC4GLS0 | G1D4T | G1D4N | G1D3T | G1D3N | G1D2T | G1D2N | G1D1T | G1D1N | xxxx xxxx | uuuu uuuu |
| F35h | CLC4GLS1 | G2D4T | G2D4N | G2D3T | G2D3N | G2D2T | G2D2N | G2D1T | G2D1N | xxxx xxxx | uuuu uuuu |
| F36h | CLC4GLS2 | G3D4T | G3D4N | G3D3T | G3D3N | G3D2T | G3D2N | G3D1T | G3D1N | xxxx xxxx | uuuu uuuu |
| F37h | CLC4GLS3 | G4D4T | G4D4N | G4D3T | G4D3N | G4D2T | G4D2N | G4D1T | G4D1N | xxxx xxxx | uuuu uuuu |
| F2Eh — F6Fh | — | 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:** Unimplemented, read as '1'.

**2:** Unimplemented on PIC16LF1776.

## TABLE 3-12: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

| Addr | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Value on POR, BOR | Value on all other Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Bank 31** | | | | | | | | | | | |
| F8Ch to FE3h | — | Unimplemented | | | | | | | | — | — |
| FE4h | STATUS_ SHAD | — | — | — | — | — | Z | DC | C | ---- -xxx | ---- -uuu |
| FE5h | WREG_ SHAD | Working Register Shadow | | | | | | | | xxxx xxxx | uuuu uuuu |
| FE6h | BSR_SHAD | — | — | — | Bank Select Register Shadow | | | | | ---x xxxx | ---u uuuu |
| FE7h | PCLATH_ SHAD | — | Program Counter Latch High Register Shadow | | | | | | | -xxx xxxx | uuuu uuuu |
| FE8h | FSR0L_ SHAD | Indirect Data Memory Address 0 Low Pointer Shadow | | | | | | | | xxxx xxxx | uuuu uuuu |
| FE9h | FSR0H_ SHAD | Indirect Data Memory Address 0 High Pointer Shadow | | | | | | | | xxxx xxxx | uuuu uuuu |
| FEAh | FSR1L_ SHAD | Indirect Data Memory Address 1 Low Pointer Shadow | | | | | | | | xxxx xxxx | uuuu uuuu |
| FEBh | FSR1H_ SHAD | Indirect Data Memory Address 1 High Pointer Shadow | | | | | | | | xxxx xxxx | uuuu uuuu |
| FECh | — | Unimplemented | | | | | | | | — | — |
| FEDh | STKPTR | — | — | — | Current Stack Pointer | | | | | ---1 1111 | ---1 1111 |
| FEEh | TOSL | Top of Stack Low byte | | | | | | | | xxxx xxxx | uuuu uuuu |
| FEFh | TOSH | — | Top of Stack High byte | | | | | | | -xxx xxxx | -uuu uuuu |

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

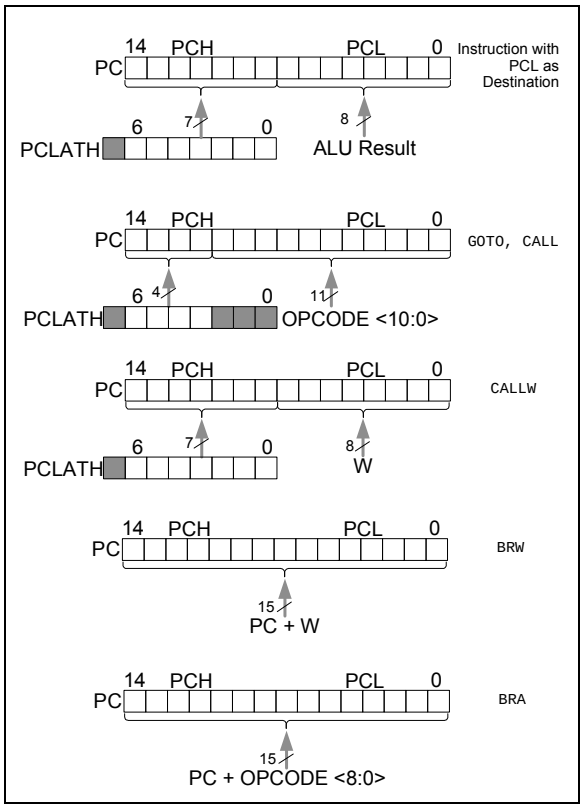
**Note 1:** Unimplemented, read as '1'.

**2:** Unimplemented on PIC16LF1776.

## 3.5 PCL and PCLATH

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

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




### 3.5.1 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.

### 3.5.2 COMPUTED GOTO

A computed GOTO is accomplished by adding an offset to the program counter (ADDWF PCL). When performing a table read using a computed GOTO method, care should be exercised if the table location crosses a PCL memory boundary (each 256-byte block). Refer to Application Note AN556, *"Implementing a Table Read"* (DS00556).

### 3.5.3 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.

### 3.5.4 BRANCHING

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

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

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

## 3.6 Stack

All devices have a 16-level x 15-bit wide hardware stack (refer to Figures 3-1 and 3-2). 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:** | 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. |
|---|---|

### 3.6.1 ACCESSING THE STACK

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

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

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

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

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

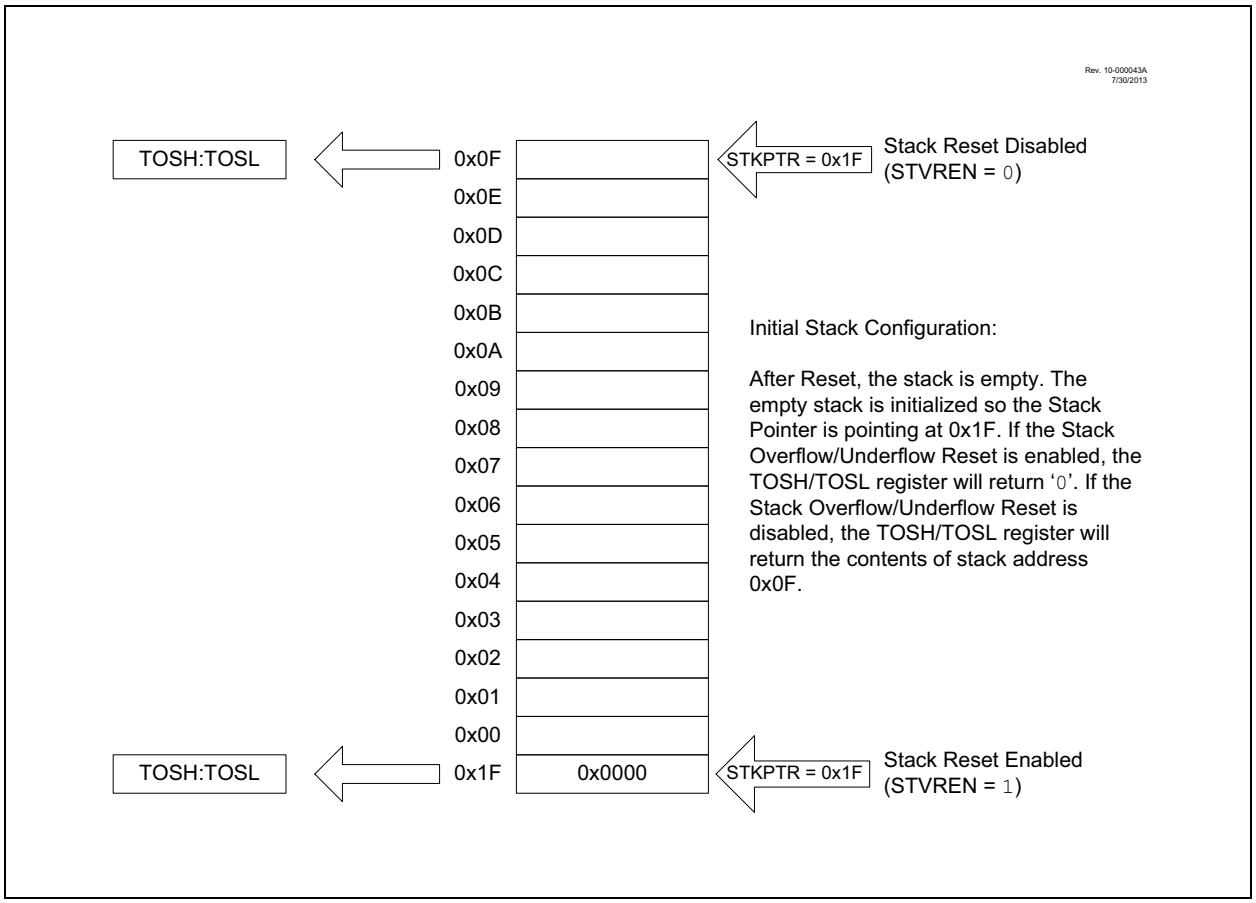


**FIGURE 3-6: ACCESSING THE STACK EXAMPLE 2**

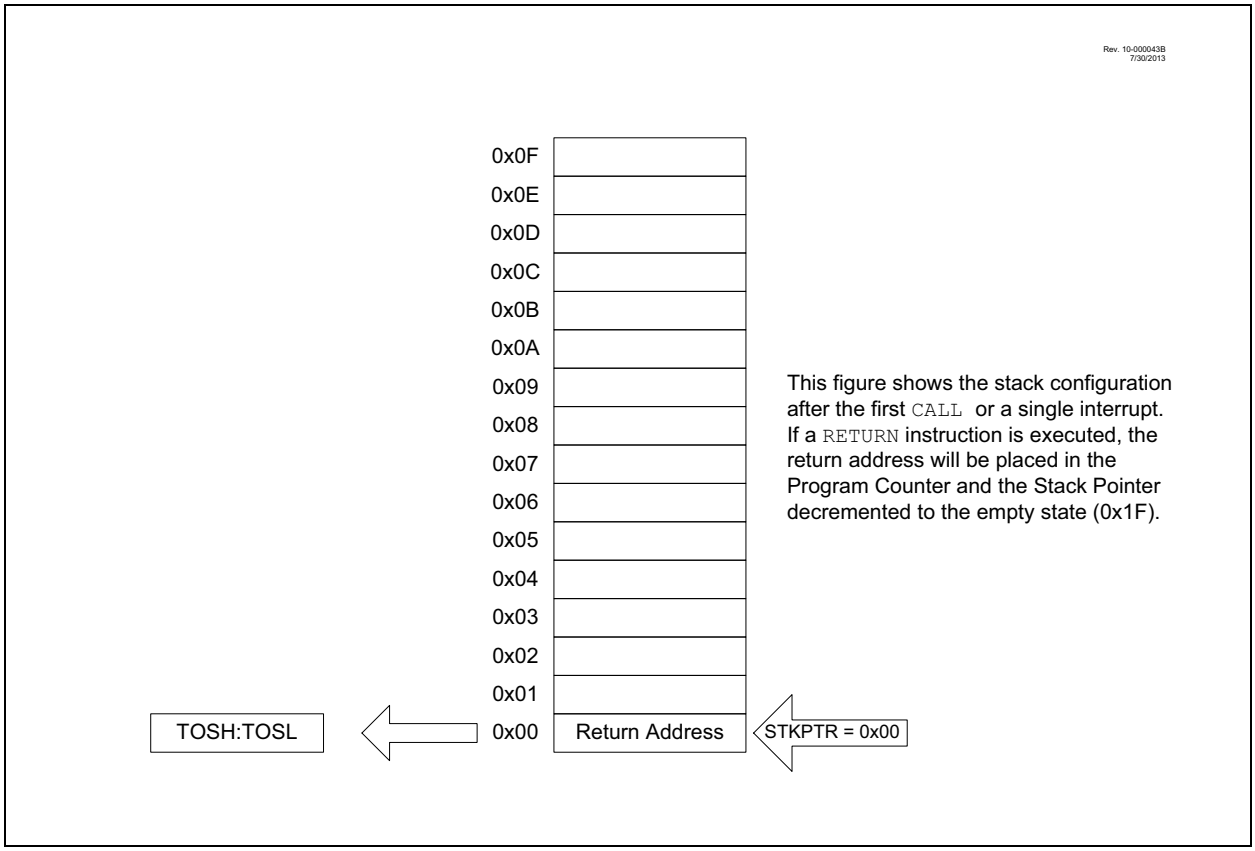



**FIGURE 3-7: ACCESSING THE STACK EXAMPLE 3**

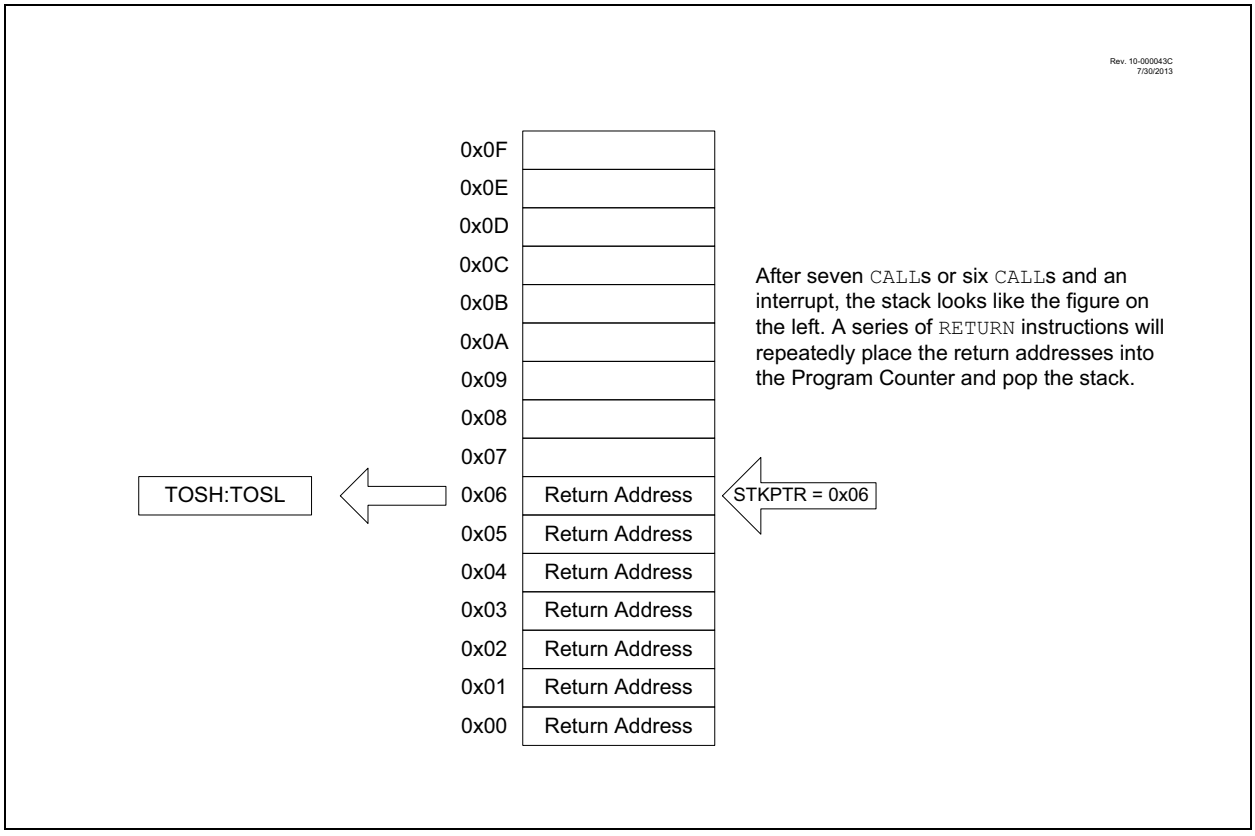
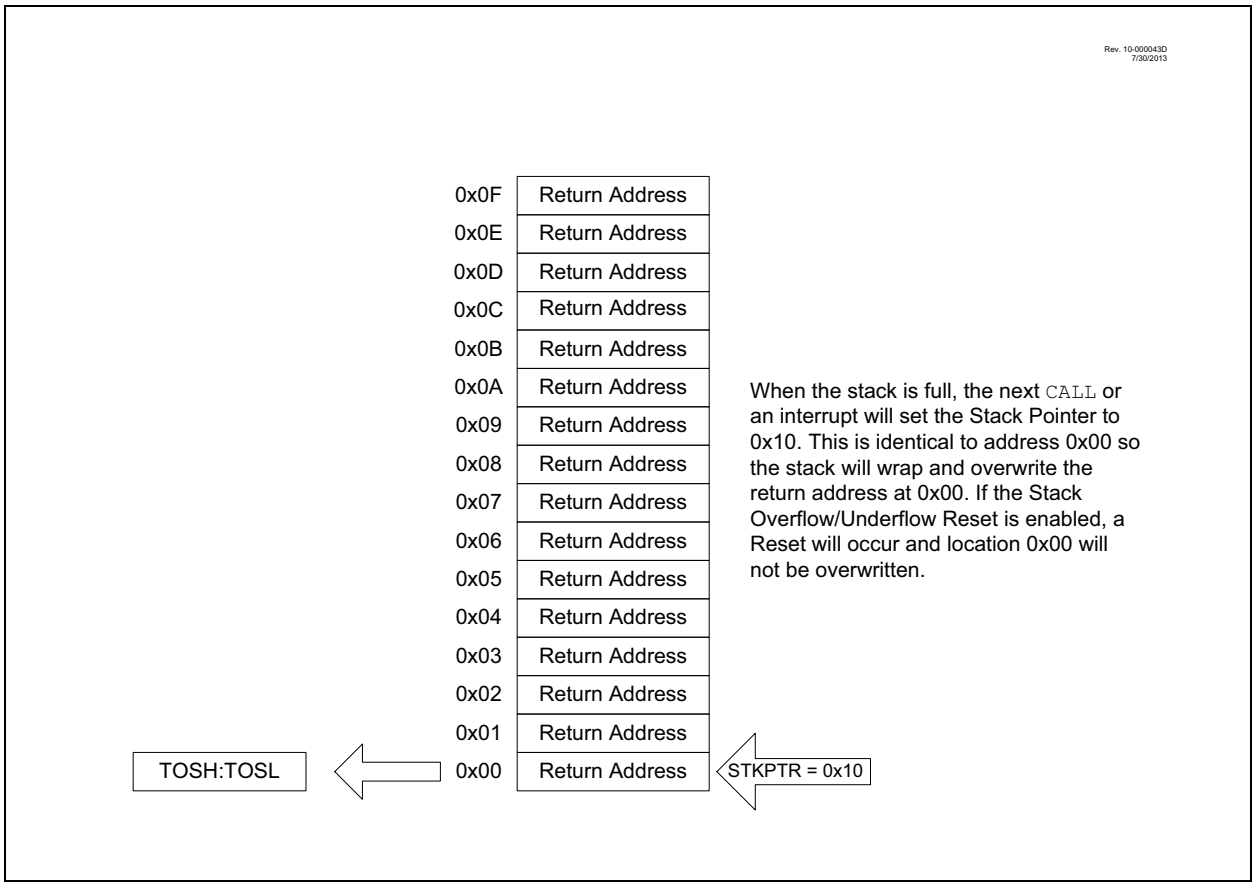


**FIGURE 3-8: ACCESSING THE STACK EXAMPLE 4**




### 3.6.2 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.

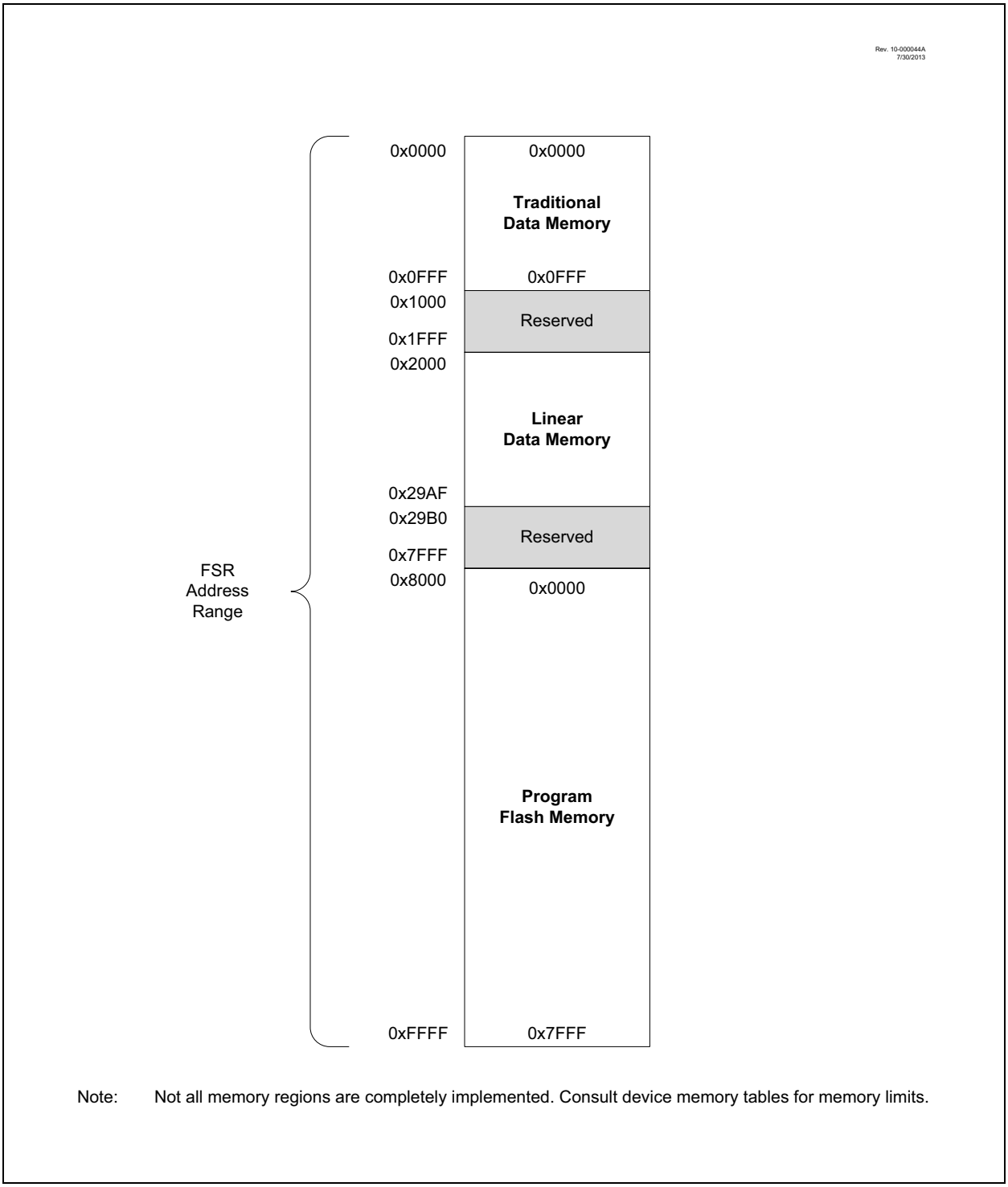
## 3.7 Indirect Addressing

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**

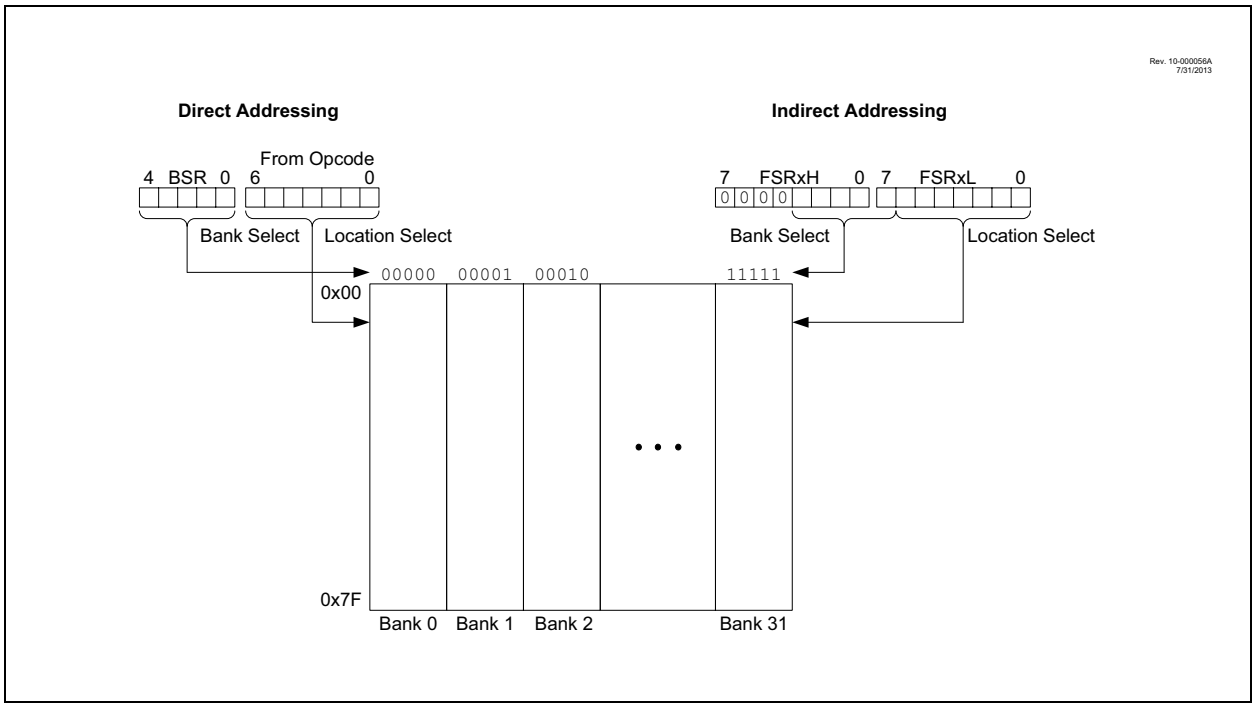



Note: Not all memory regions are completely implemented. Consult device memory tables for memory limits.

### 3.7.1 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**



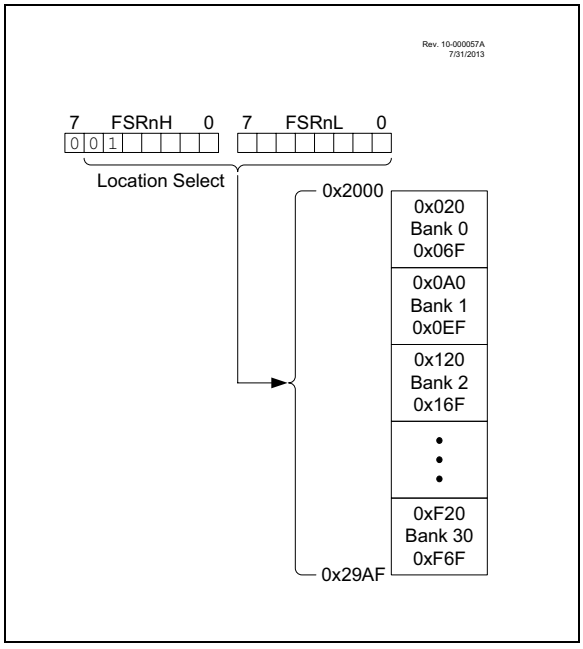
### 3.7.2 LINEAR DATA MEMORY

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.

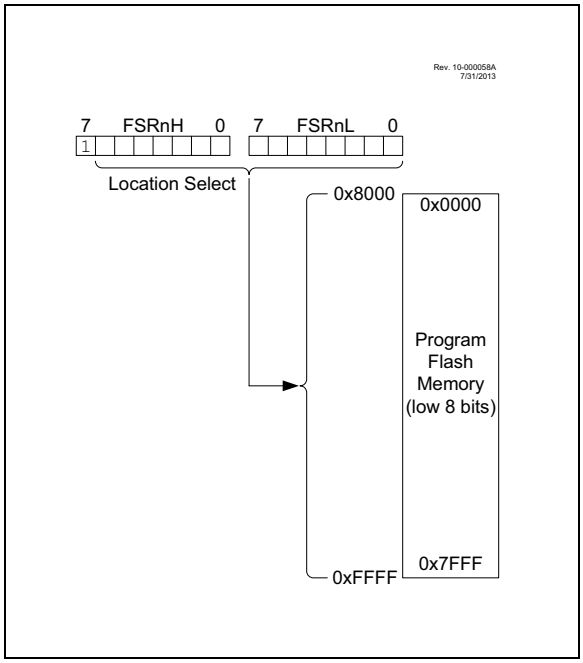
**FIGURE 3-11: LINEAR DATA MEMORY MAP**




### 3.7.3 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**



## 4.0 DEVICE CONFIGURATION

Device configuration consists of Configuration Words, Code Protection and Device ID.

### 4.1 Configuration Words

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

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

## 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 | U-1 |
|---|---|---|---|---|---|
| FCMEN | IESO | $\overline{\text{CLKOUTEN}}$ | BOREN<1:0> | | — |
| bit 13 | | | | | bit 8 |

| R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 | R/P-1 |
|---|---|---|---|---|---|---|---|
| $\overline{\text{CP}}$(1) | 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 = ON Fail-Safe Clock Monitor and internal/external switchover are both enabled
0 = OFF Fail-Safe Clock Monitor is disabled

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

bit 11 **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 = OFF CLKOUT function is disabled. I/O function on the CLKOUT pin.
0 = ON CLKOUT function is enabled on the CLKOUT pin.

bit 10-9 **BOREN<1:0>:** Brown-out Reset Enable bits
11 = ON BOR enabled
10 = NSLEEP BOR enabled during operation and disabled in Sleep
01 = SBODEN BOR controlled by SBOREN bit of the BORCON register
00 = OFF BOR disabled

bit 8 **Unimplemented:** Read as '1'

bit 7 **$\overline{\text{CP}}$:** Code Protection bit(1)
1 = OFF Program memory code protection is disabled
0 = ON 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 = ON $\overline{\text{MCLR}}$/$V_{PP}$ pin function is $\overline{\text{MCLR}}$; Weak pull-up enabled.
0 = OFF $\overline{\text{MCLR}}$/$V_{PP}$ pin function is digital input; $\overline{\text{MCLR}}$ internally disabled; Weak pull-up under control of WPUA3 bit.

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

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

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

bit 2-0 **FOSC<2:0>:** Oscillator Selection bits

| | | |
|---|---|---|
| 111 = | ECH | External Clock, High-Power mode: CLKIN supplied to OSC1/CLKIN pin |
| 110 = | ECM | External Clock, Medium Power mode: CLKIN supplied to OSC1/CLKIN pin |
| 101 = | ECL | External Clock, Low-Power mode: CLKIN supplied to OSC1/CLKIN pin |
| 100 = | INTOSC | Internal HFINTOSC. I/O function on CLKIN pin |
| 011 = | EXTRC | External RC circuit connected to CLKIN pin |
| 010 = | HS | High-speed crystal/resonator connected between OSC1 and OSC2 pins |
| 001 = | XT | Crystal/resonator connected between OSC1 and OSC2 pins |
| 000 = | LP | Low-power crystal connected between OSC1 and OSC2 pins |

**Note 1:** The entire Flash program memory will be erased when the code protection is turned off during an erase. When a Bulk Erase Program Memory command is executed, the entire program Flash memory and configuration memory will be erased.

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

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

| R/P-1 | U-1 | U-1 | U-1 | U-1 | R/P-1 | R/P-1 | R/P-1 |
|---|---|---|---|---|---|---|---|
| $\overline{\text{ZCD}}$ | — | — | — | — | PPS1WAY | 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 = ON Low-voltage programming enabled
- 0 = OFF High-voltage on $\overline{\text{MCLR}}$ must be used for programming

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

bit 11 **$\overline{\text{LPBOR}}$:** Low-Power BOR Enable bit
- 1 = OFF Low-Power Brown-out Reset is disabled
- 0 = ON Low-Power Brown-out Reset is enabled

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

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

bit 8 **PLLEN:** PLL Enable bit
- 1 = ON 4xPLL enabled
- 0 = OFF 4xPLL disabled

bit 7 **$\overline{\text{ZCD}}$:** ZCD Enable bit
- 1 = OFF ZCD disabled. ZCD can be enabled by setting the ZCDSEN bit of ZCDCON
- 0 = ON ZCD always enabled

bit 6-3 **Unimplemented:** Read as '1'

bit 2 **PPS1WAY**: PPSLOCK Bit One-Way Set Enable bit
- 1 = ON The PPSLOCK bit can only be set once after an unlocking sequence is executed; once PPSLOCK is set, all future changes to PPS registers are prevented
- 0 = OFF The PPSLOCK bit can be set and cleared as needed (provided an unlocking sequence is executed)

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

4 kW Flash memory (PIC16(L)F1764/8)
- 11 = OFF Write protection off
- 10 = BOOT 0000h to 01FFh write protected, 0200h to 0FFFh may be modified by PMCON control
- 01 = HALF 0000h to 07FFh write protected, 0800h to 0FFFh may be modified by PMCON control
- 00 = ALL 0000h to 0FFFh write protected, no addresses may be modified by PMCON control

8 kW Flash memory (PIC16(L)F1765/9)
- 11 = OFF Write protection off
- 10 = BOOT 0000h to 01FFh write protected, 0200h to 1FFFh may be modified by PMCON control
- 01 = HALF 0000h to 0FFFh write protected, 1000h to 1FFFh may be modified by PMCON control
- 00 = ALL 0000h to 1FFFh write protected, no addresses may be modified by PMCON control

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

## 4.3 Code Protection

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

### 4.3.1 PROGRAM MEMORY PROTECTION

The entire program memory space is protected from external reads and writes by the $\overline{CP}$ bit in Configuration Words. When $\overline{CP}$ = 0, external reads and writes of program memory are inhibited and a read will return all '0's. The CPU can continue to read program memory, regardless of the protection bit settings. Writing the program memory is dependent upon the write protection setting. See **Section 4.4 "Write Protection"** for more information.

## 4.4 Write Protection

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

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

## 4.5 User ID

Four memory locations (8000h-8003h) are designated as ID locations where the user can store checksum or other code identification numbers. These locations are readable and writable during normal execution. See **Section 10.4 "User ID, Device ID and Configuration Word Access"** for more information on accessing these memory locations. For more information on checksum calculation, see the *"PIC16(L)F170X Memory Programming Specification"* (DS41683).

## 4.6 Device ID and Revision ID

The 14-bit device ID word is located at 8006h and the 14-bit revision ID is located at 8005h. These locations are read-only and cannot be erased or modified. See **Section 10.4 "User ID, Device ID and Configuration Word Access"** for more information on accessing these memory locations.

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

## 4.7 Register Definitions: Device and Revision

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

| R | R | R | R | R | R |
|---|---|---|---|---|---|
| DEV<13:8> | | | | | |
| bit 13 | | | | | bit 8 |

| R | R | R | R | R | R | R | R |
|---|---|---|---|---|---|---|---|
| DEV<7:0> | | | | | | | |
| bit 7 | | | | | | | bit 0 |

| **Legend:** | |
|---|---|
| R = Readable bit | |
| '1' = Bit is set | '0' = Bit is cleared |

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

| Device | DEVID<13:0> Values |
|---|---|
| PIC16F1773 | 11 0000 1000 0000 (308A) |
| PIC16F1776 | 11 0000 1000 0001 (308B) |
| PIC16LF1773 | 11 0000 1000 0100 (308C) |
| PIC16LF1776 | 11 0000 1000 0101 (308D) |

**REGISTER 4-4: REVID: REVISION ID REGISTER**

| R | R | R | R | R | R |
|---|---|---|---|---|---|
| REV<13:8> | | | | | |
| bit 13 | | | | | bit 8 |

| R | R | R | R | R | R | R | R |
|---|---|---|---|---|---|---|---|
| REV<7:0> | | | | | | | |
| bit 7 | | | | | | | bit 0 |

| **Legend:** | |
|---|---|
| R = Readable bit | |
| '1' = Bit is set | '0' = Bit is cleared |

bit 13-0 **REV<13:0>:** Revision ID bits

# 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, ECH, ECM, ECL or EXTRC 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 the following clock modes.

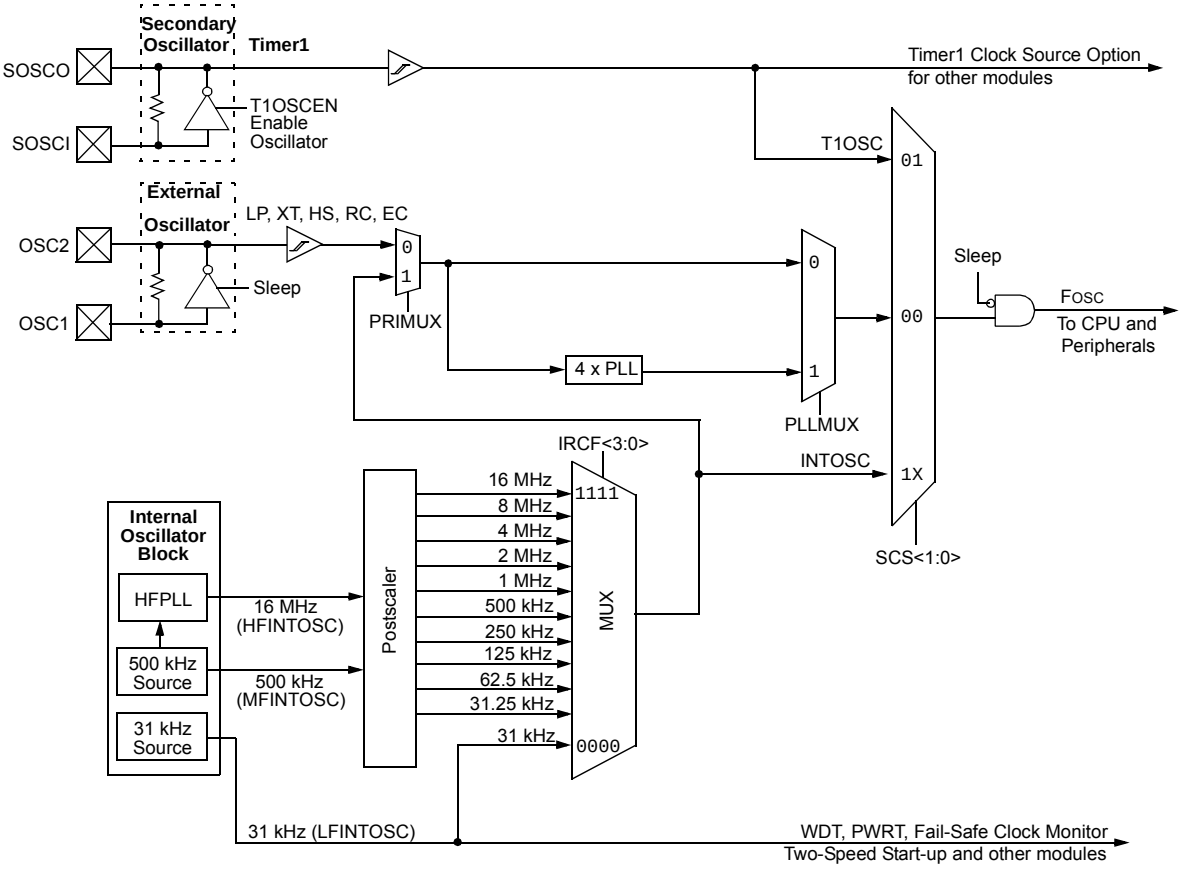
1. ECL – External Clock Low-Power mode (0 MHz to 0.5 MHz)
2. ECM – External Clock Medium Power mode (0.5 MHz to 4 MHz)
3. ECH – External Clock High-Power mode (4 MHz to 32 MHz)
4. LP – 32 kHz Low-Power Crystal mode.
5. XT – Medium Gain Crystal or Ceramic Resonator Oscillator mode (up to 4 MHz)
6. HS – High Gain Crystal or Ceramic Resonator mode (4 MHz to 20 MHz)
7. EXTRC – External Resistor-Capacitor
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 ECH, ECM, and ECL clock modes rely 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 EXTRC 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.

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




| Inputs | | | | Outputs | |
|---|---|---|---|---|---|
| **SCS** | **FOSC<2:0>** | **PLLEN or SPLLEN** | **IRCF** | **PRIMUX** | **PLLMUX** |
| =00 | =100 | 0 | x | 1 | 0 |
| | | 1 | =1110 | 1 | 1 |
| | | | ≠1110 | 1 | 0 |
| | ≠100 | 0 | x | 0 | 0 |
| | | 1 | x | 0 | 1 |
| ≠00 | X | X | X | X | X |

## 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 (ECH, ECM, ECL mode), quartz crystal resonators or ceramic resonators (LP, XT and HS modes) and Resistor-Capacitor (EXTRC) 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:
  - Secondary oscillator during run-time, or
  - An external clock source determined by the value of the FOSC bits.

See **Section 5.3 "Clock Switching"** for more information.

#### 5.2.1.1 EC Mode

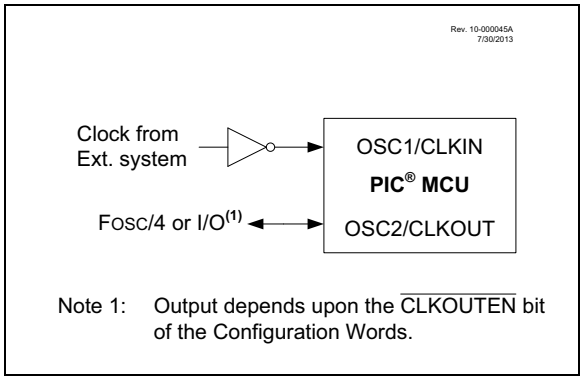
The External Clock (EC) mode allows an externally generated logic level signal to be the system clock source. When operating in this mode, an external clock source is connected to the OSC1 input. OSC2/CLKOUT is available for general purpose I/O or CLKOUT. Figure 5-2 shows the pin connections for EC mode.

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

- ECH – High power, 4-32 MHz
- ECM – Medium power, 0.5-4 MHz
- ECL – Low power, 0-0.5 MHz

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**




#### 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.

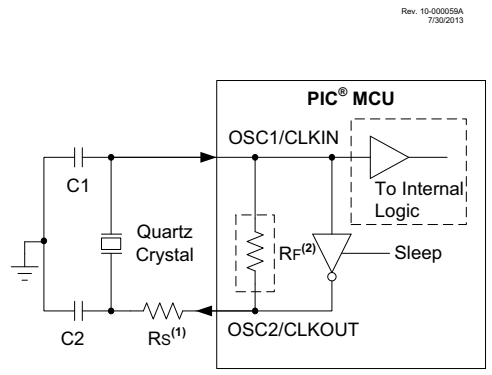
**LP** Oscillator mode selects the lowest gain setting of the internal inverter-amplifier. LP mode current consumption is the least of the three modes. This mode is designed to drive only 32.768 kHz tuning-fork type crystals (watch crystals).

**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.

**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 and 10 MΩ).

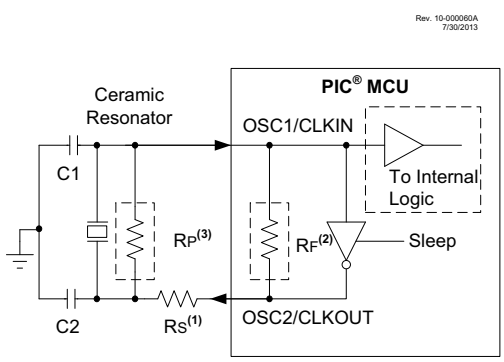
**Note 1:** Quartz crystal characteristics vary according to type, package and manufacturer. The user should consult the manufacturer data sheets for specifications and recommended application.

**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 Application Notes:

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

**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 and 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, unless either FSCM or Two-Speed Start-Up are enabled. In this case, code will continue to execute at the selected INTOSC frequency while the OST is counting. 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 Table 36-9.

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

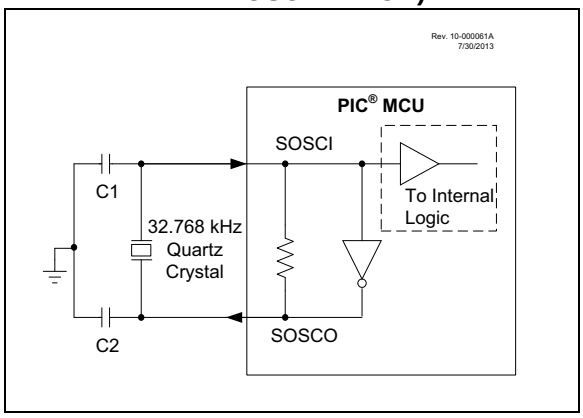
1. Program the PLLEN bit in Configuration Words to a '1'.
2. Write the SPLLEN bit in the OSCCON register to a '1'. If the PLLEN bit in Configuration Words is programmed to a '1', then the value of SPLLEN is ignored.

#### 5.2.1.5 Secondary Oscillator

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

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

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




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

**2:** Always verify oscillator performance over the $V_{DD}$ and temperature range that is expected for the application.

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

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

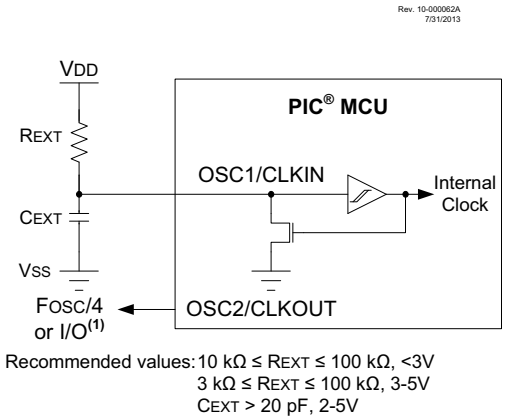
#### 5.2.1.6 External RC Mode

The external Resistor-Capacitor (EXTRC) mode supports 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 CLKOUTEN bit in Configuration Words.

Figure 5-6 shows the external RC mode connections.

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




**Note 1:** Output depends upon the 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.

### 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 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 postscaled output of the 16 MHz HFINTOSC, 500 kHz MFINTOSC, and 31 kHz LFINTOSC connect to a multiplexer (see Figure 5-1). The Internal Oscillator Frequency Select bits IRCF<3:0> of the OSCCON register select the frequency output of the internal oscillators. One of the following frequencies can be selected via software:

- 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
- 31 kHz (LFINTOSC)

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

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

#### 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 4x PLL, 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 4x PLL cannot be disabled by software and the SPLLEN option will not be available. |
|---|---|

The 4x PLL 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 4x PLL 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:

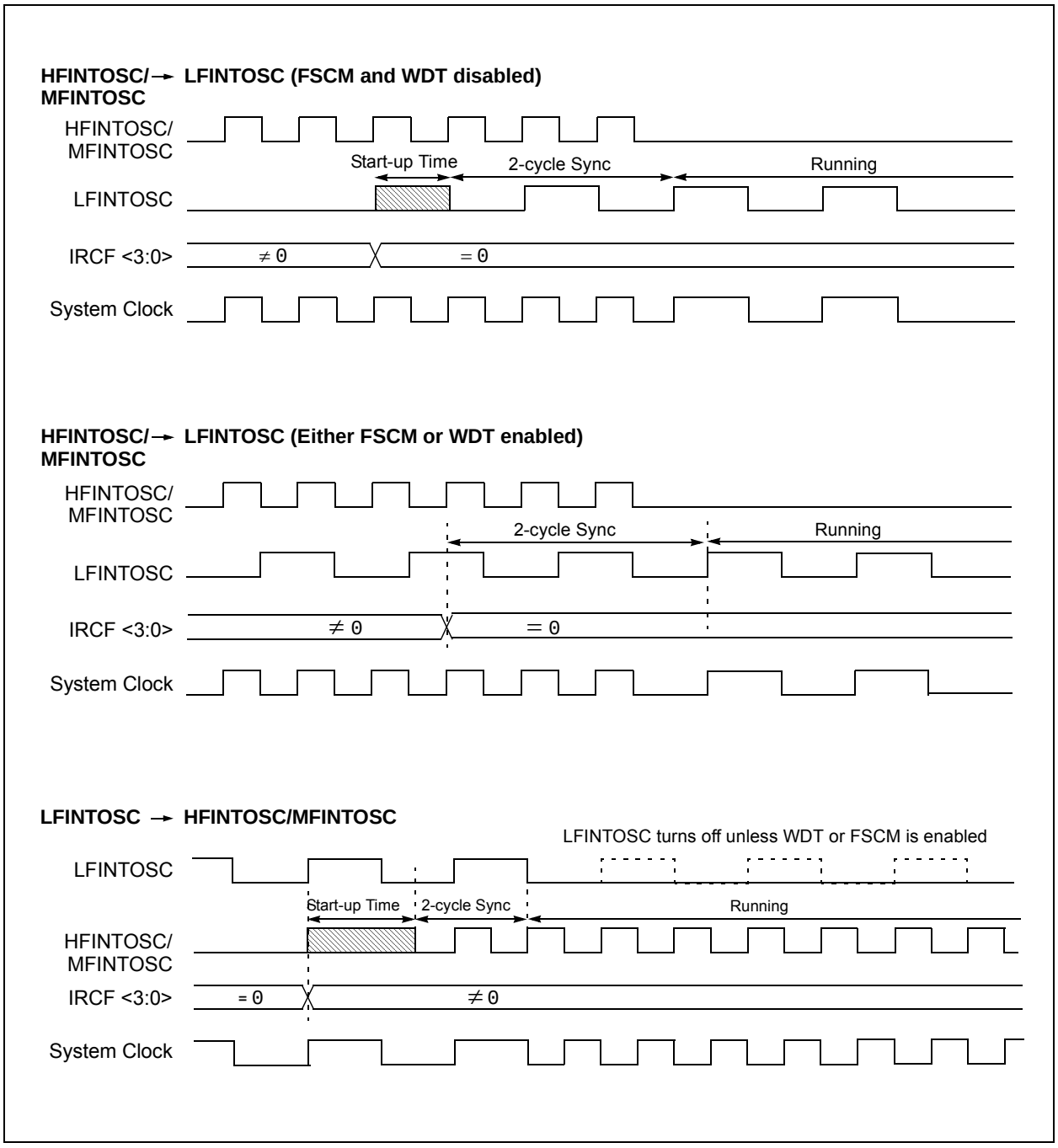
1. IRCF<3:0> bits of the OSCCON register are modified.
2. If the new clock is shut down, a clock start-up delay is started.
3. Clock switch circuitry waits for a falling edge of the current clock.
4. The current clock is held low and the clock switch circuitry waits for a rising edge in the new clock.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

Start-up delay specifications are located in the oscillator tables of **Section 36.0 "Electrical Specifications"**.

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



## 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 select 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 the 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 secondary oscillator.
- When the SCS bits of the OSCCON register = 1x, the system clock source is chosen by the internal oscillator frequency selected by the IRCF<3:0> bits of the OSCCON register. After a Reset, the SCS bits of the OSCCON register are always cleared.

| | |
|---|---|
| **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 TIMER STATUS (OSTS) BIT

The Oscillator Start-up Timer 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 secondary oscillator.

### 5.3.3 SECONDARY OSCILLATOR

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

The secondary oscillator is enabled using the OSCEN control bit in the T1CON register. See **Section 22.0 "Timer1/3/5 Module with Gate Control"** for more information about the Timer1 peripheral.

### 5.3.4 SECONDARY OSCILLATOR READY (SOSCR) BIT

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

### 5.3.5 CLOCK SWITCH BEFORE SLEEP

When a clock switch 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 | Oscillator Warm-up Delay (TWARM)(2) |
| Sleep/POR | EC, RC(1) | DC – 32 MHz | 2 cycles |
| LFINTOSC | EC, RC(1) | DC – 32 MHz | 1 cycle of each |
| Sleep/POR | Secondary Oscillator 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 | Secondary Oscillator | 32 kHz | 1024 Clock Cycles (OST) |
| PLL inactive | PLL active | 16-32 MHz | 2 ms (approx.) |

**Note 1:** PLL inactive.

**2:** See **Section 36.0 "Electrical Specifications"**.

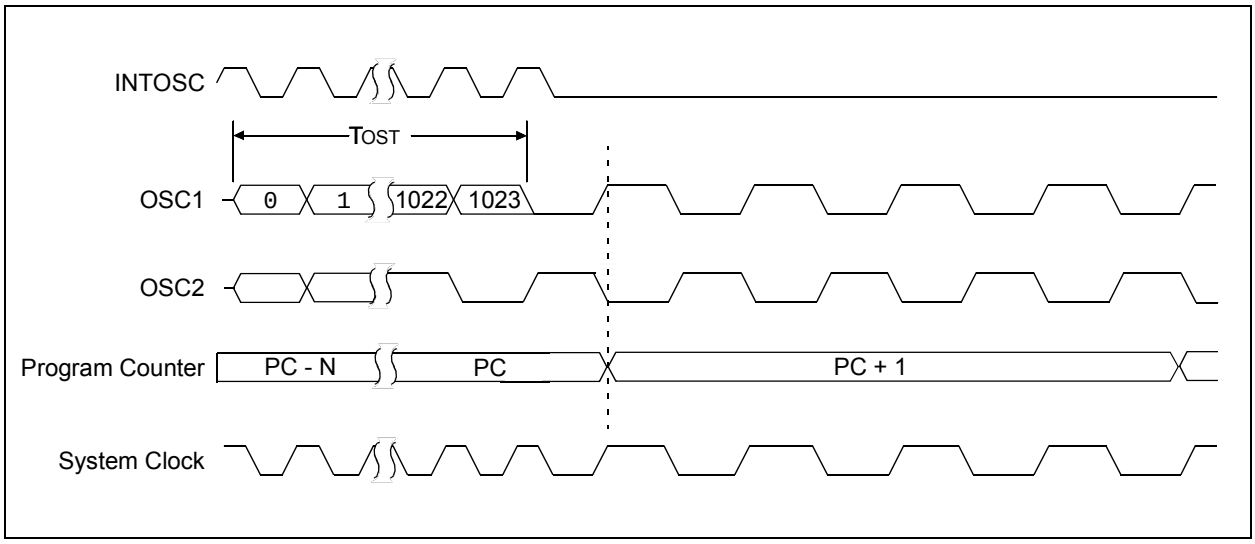
### 5.4.2 TWO-SPEED START-UP SEQUENCE

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.

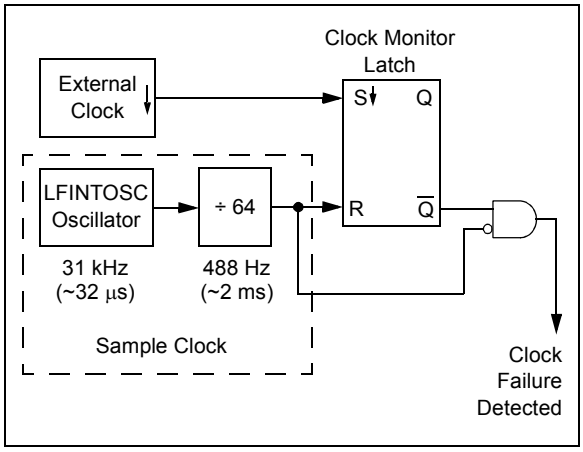
**FIGURE 5-8: TWO-SPEED START-UP**



## 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, Secondary 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

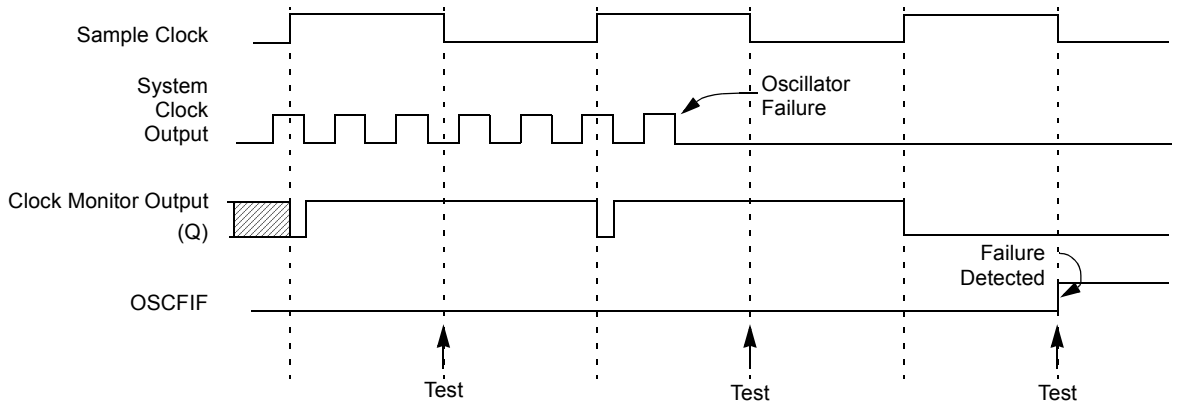
The Fail-Safe condition is cleared after a Reset, executing a SLEEP instruction or changing the SCS bits of the OSCCON register. When the SCS bits are changed, the OST is restarted. While the OST is running, the device continues to operate from the INTOSC selected in OSCCON. When the OST times out, the 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. |

**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(2)
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 = Secondary oscillator
00 = Clock determined by FOSC<2:0> in Configuration Words

**Note 1:** Duplicate frequency derived from HFINTOSC.
**2:** 32 MHz when SPLLEN bit is set. Refer to **Section 5.2.2.6 “32 MHz Internal Oscillator Frequency Selection”**.

**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 |
|---|---|---|---|---|---|---|---|
| SOSCR | 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 **SOSCR:** Secondary Oscillator Ready bit
If T1OSCEN = 1:
1 = Secondary oscillator is ready
0 = Secondary oscillator is not ready
If T1OSCEN = 0:
1 = Secondary 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 Timer 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> | | 92 |
| OSCSTAT | SOSCR | PLLR | OSTS | HFIOFR | HFIOFL | MFIOFR | LFIOFR | HFIOFS | 93 |
| OSCTUNE | — | — | TUN<5:0> | | | | | | 94 |
| PIR2 | OSFIF | C2IF | C1IF | COG1IF | BCL1IF | C4IF | C3IF | CCP2IF | 116 |
| PIE2 | OSFIE | C2IE | C1IE | COG1IE | BCL1IE | C4IE | C3IE | CCP2IE | 110 |
| T1CON | CS<1:0> | | CKPS<1:0> | | OSCEN | $\overline{\text{SYNC}}$ | — | ON | 245 |

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

**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> | | — | 71 |
| | 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 clock sources.

## 6.0 RESETS

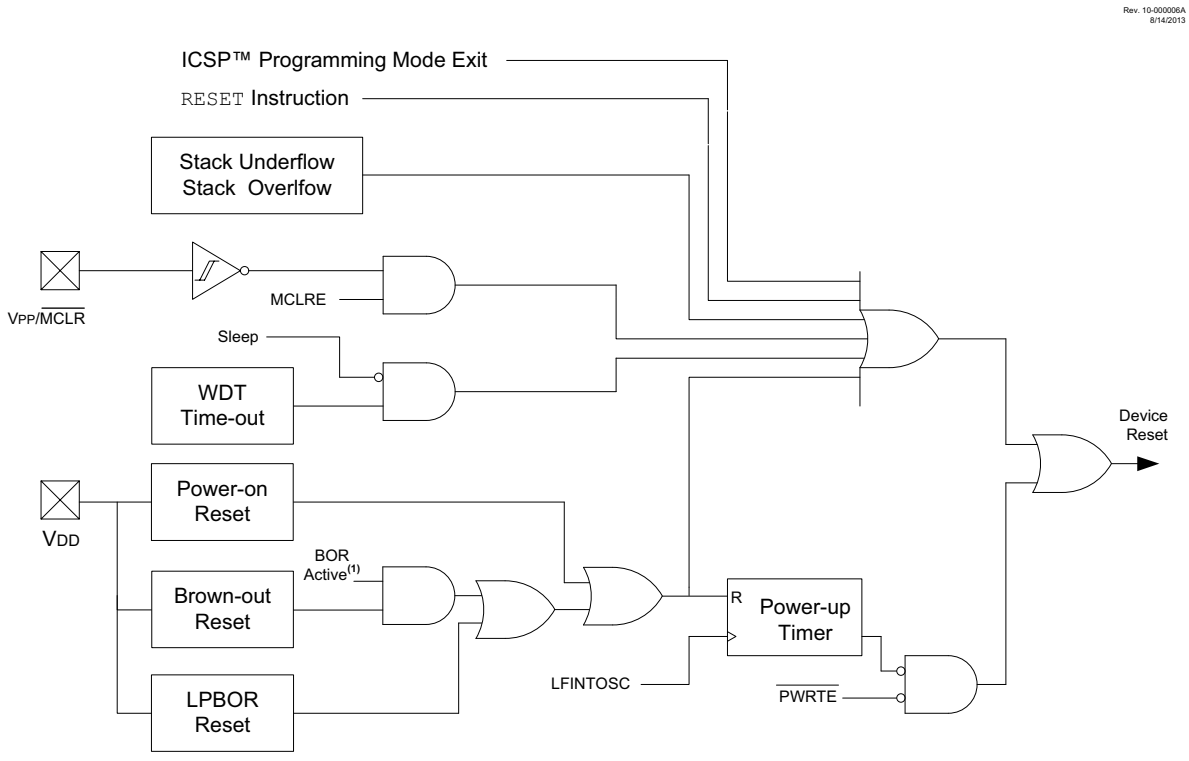
There are multiple ways to reset this device:

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

To allow VDD to stabilize, an optional Power-up Timer can be enabled to extend the Reset time after a BOR or POR event.

A simplified block diagram of the On-Chip Reset Circuit is shown in Figure 6-1.

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




**Note 1:** See Table 6.2.1 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 6.2.1 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.

### 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.

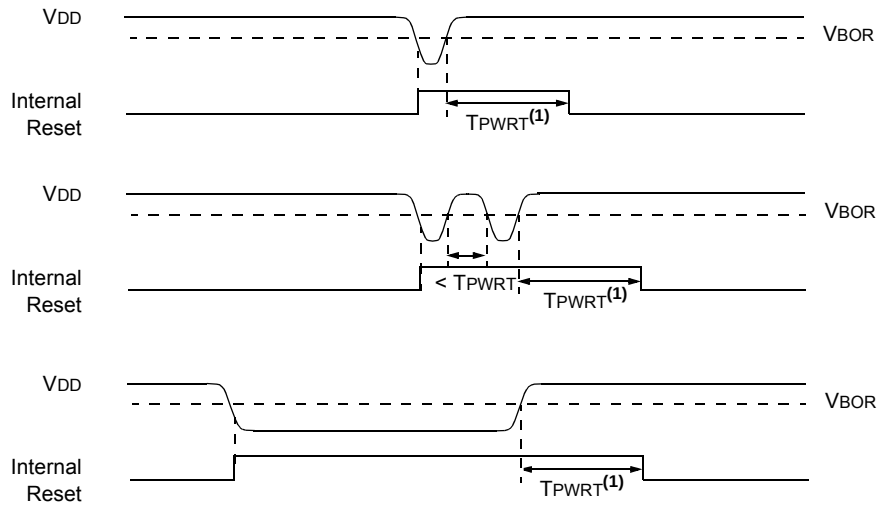
BOR protection is unchanged by Sleep.

**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.

**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 | R/W-0/u | U-0 | U-0 | U-0 | U-0 | U-0 | R-q/u |
|---|---|---|---|---|---|---|---|
| SBOREN | BORFS(1) | — | — | — | — | — | 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 **BORFS:** Brown-out Reset Fast Start bit(1)
If BOREN<1:0> = 11 (Always on) or BOREN<1:0> = 00 (Always off)
BORFS is Read/Write, but has no effect.
If BOREN <1:0> = 10 (Disabled in Sleep) or BOREN<1:0> = 01 (Under software control):
1 = Band gap is forced on always (covers sleep/wake-up/operating cases)
0 = Band gap operates normally, and may turn off

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

bit 0 **BORRDY:** Brown-out Reset Circuit Ready Status bit
1 = The Brown-out Reset circuit is active
0 = The Brown-out Reset circuit is inactive

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

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

The Low-Power Brown-Out Reset (LPBOR) is an essential part of the Reset subsystem. Refer to Figure 6-1 to see how the BOR interacts with other modules.

The LPBOR is used to monitor the external VDD pin. When too low of a voltage is detected, the device is held in Reset. When this occurs, a register bit ($\overline{BOR}$) is changed to indicate that a BOR Reset has occurred. The same bit is set for both the BOR and the LPBOR. Refer to Register 6-2.

### 6.4.1 ENABLING LPBOR

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

#### 6.4.1.1 LPBOR Module Output

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

## 6.5 $\overline{MCLR}$

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

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

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

### 6.5.1 $\overline{MCLR}$ ENABLED

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

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

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

### 6.5.2 $\overline{MCLR}$ DISABLED

When $\overline{MCLR}$ is disabled, the pin functions as a general purpose input and the internal weak pull-up is under software control. See **Section 11.1 "PORTA Registers"** for more information.

## 6.6 Watchdog Timer (WDT) Reset

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

## 6.7 RESET Instruction

A RESET instruction will cause a device Reset. The $\overline{RI}$ bit in the PCON register will be set to '0'. See Table 6-4 for default conditions after a RESET instruction has occurred.

## 6.8 Stack Overflow/Underflow Reset

The device can reset when the Stack Overflows or Underflows. The STKOVF or STKUNF bits of the PCON register indicate the Reset condition. These Resets are enabled by setting the STVREN bit in Configuration Words. See **Section 3.6.2 "Overflow/Underflow Reset"** for more information.

## 6.9 Programming Mode Exit

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

## 6.10 Power-Up Timer

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

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

## 6.11 Start-up Sequence

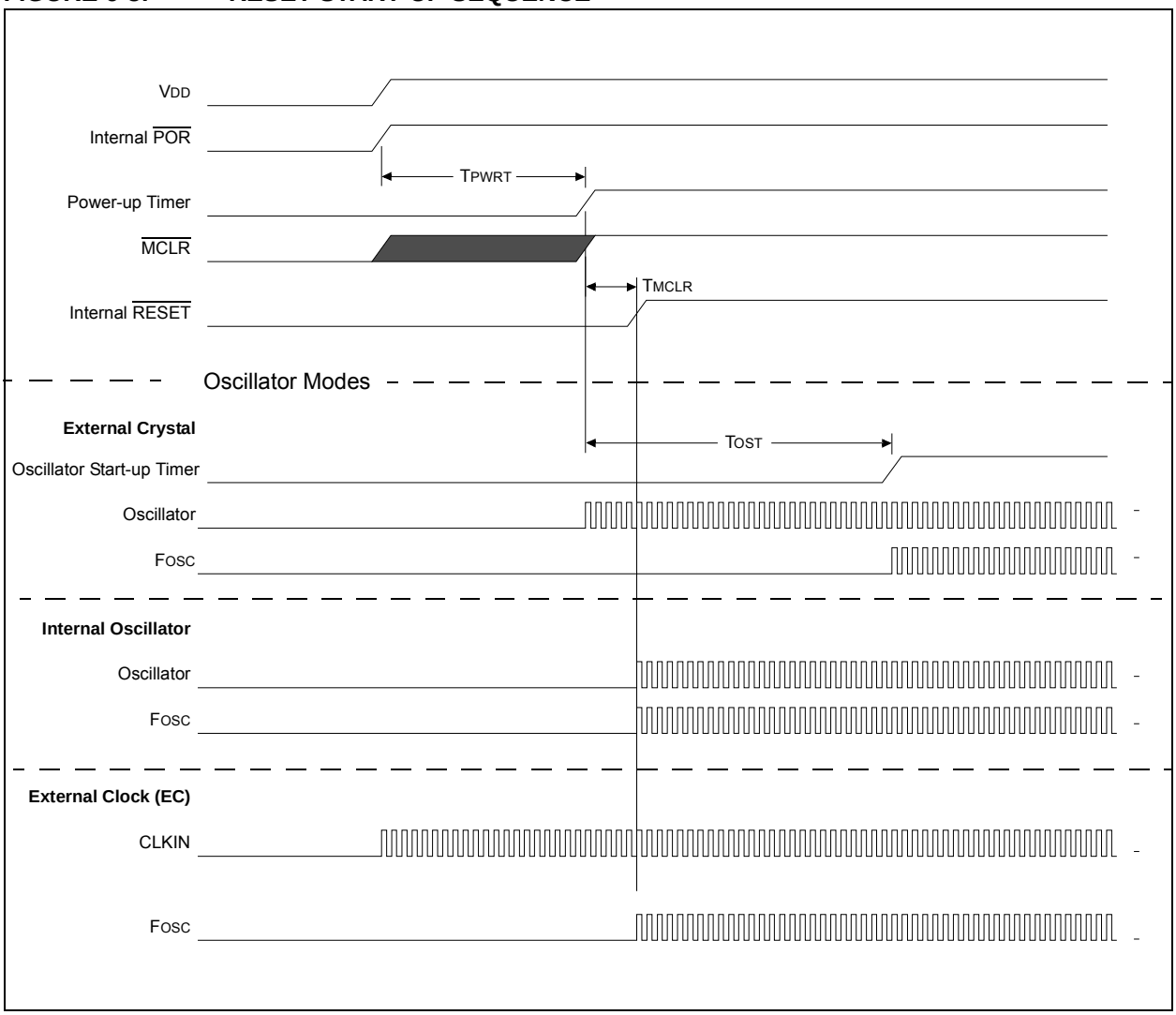
Upon the release of a POR or BOR, the following must occur before the device will begin executing:

1. Power-up Timer runs to completion (if enabled).
2. Oscillator start-up timer runs to completion (if required for oscillator source).
3. $\overline{MCLR}$ 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 $\overline{MCLR}$ Reset. If $\overline{MCLR}$ is kept low long enough, the Power-up Timer and oscillator start-up timer will expire. Upon bringing $\overline{MCLR}$ high, the device will begin execution after 10 FOSC cycles (see Figure 6-3). This is useful for testing purposes or to synchronize more than one device operating in parallel.

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



## 6.12 Determining the Cause of a Reset

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

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

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

**TABLE 6-4: RESET CONDITION FOR SPECIAL REGISTERS**

| 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.

## 6.13 Power Control (PCON) Register

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

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 ($\overline{RI}$)
- $\overline{MCLR}$ Reset ($\overline{RMCLR}$)
- Watchdog Timer Reset ($\overline{RWDT}$)
- Stack Underflow Reset (STKUNF)
- Stack Overflow Reset (STKOVF)

## 6.14 Register Definitions: Power Control

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

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

bit 6 **STKUNF:** Stack Underflow Flag bit
1 = A Stack Underflow occurred
0 = A Stack Underflow has not occurred or cleared by firmware

bit 5 **Unimplemented:** Read as ‘0’

bit 4 **$\overline{RWDT}$:** Watchdog Timer Reset Flag bit
1 = A Watchdog Timer Reset has not occurred or set to ‘1’ by firmware
0 = A Watchdog Timer Reset has occurred (cleared by hardware)

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 (cleared by hardware)

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 (cleared by hardware)

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 | BORFS | — | — | — | — | — | BORRDY | 97 |
| PCON | STKOVF | STKUNF | — | $\overline{\text{RWDT}}$ | $\overline{\text{RMCLR}}$ | $\overline{\text{RI}}$ | $\overline{\text{POR}}$ | $\overline{\text{BOR}}$ | 101 |
| STATUS | — | — | — | $\overline{\text{TO}}$ | $\overline{\text{PD}}$ | Z | DC | C | 26 |
| WDTCON | — | — | WDTPS<4:0> | | | | | SWDTEN | 130 |

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

## 7.0 INTERRUPTS

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

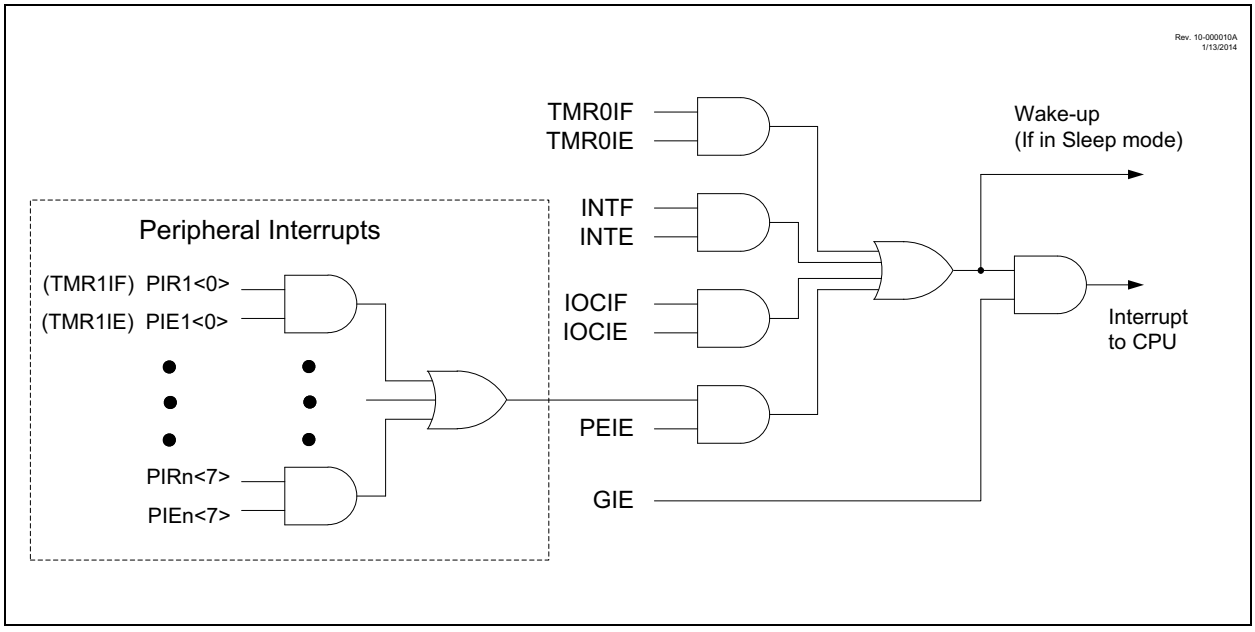
This chapter contains the following information for Interrupts:

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

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

A block diagram of the interrupt logic is shown in Figure 7-1.

**FIGURE 7-1: INTERRUPT LOGIC**



## 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 or PIE2 registers)

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

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

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

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

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

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

| | |
|---|---|
| **Note 1:** | Individual interrupt flag bits are set, regardless of the state of any other enable bits. |
| **2:** | All interrupts will be ignored while the GIE bit is cleared. Any interrupt occurring while the GIE bit is clear will be serviced when the GIE bit is set again. |

## 7.2 Interrupt Latency

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

**FIGURE 7-2: INTERRUPT LATENCY**

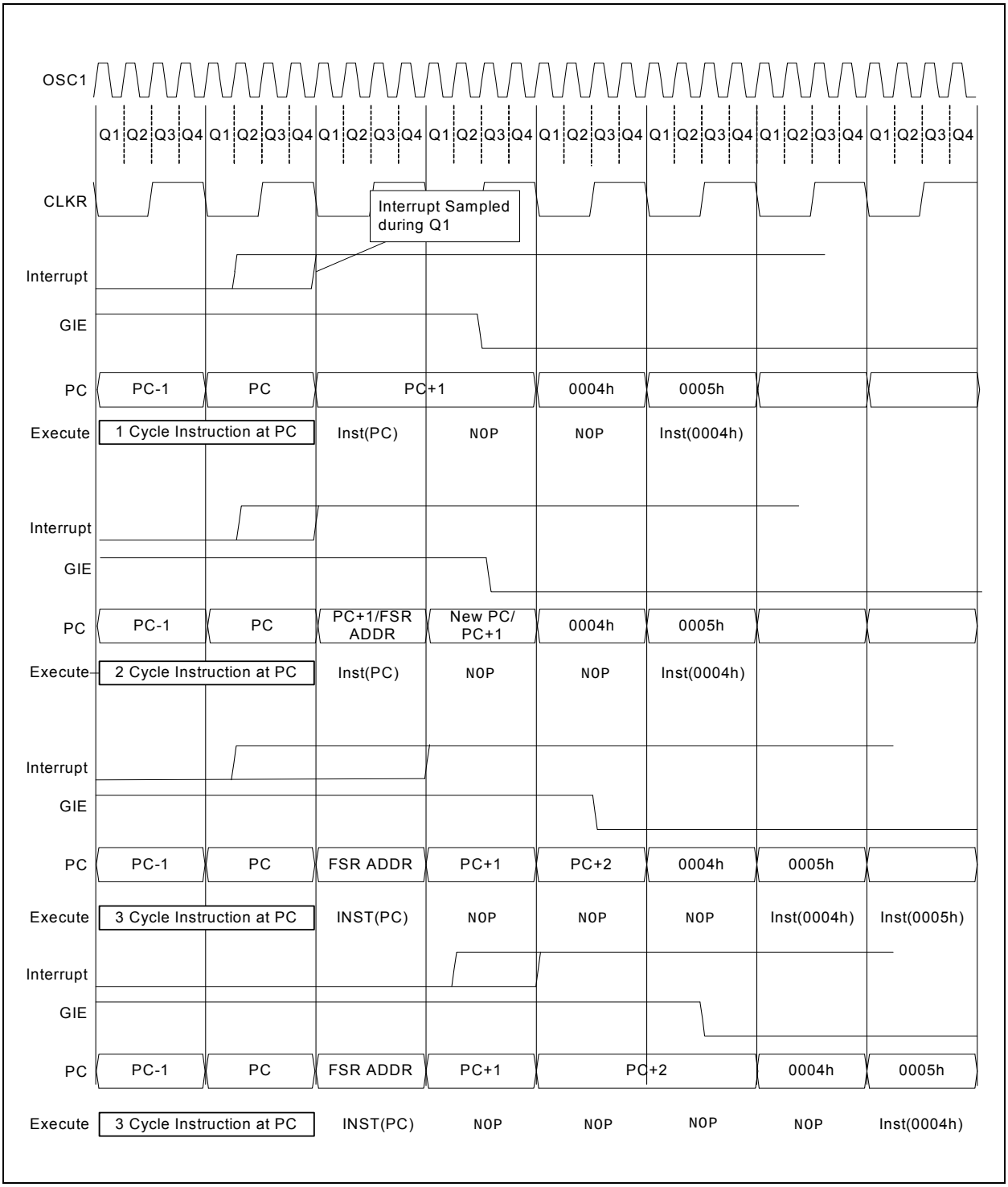
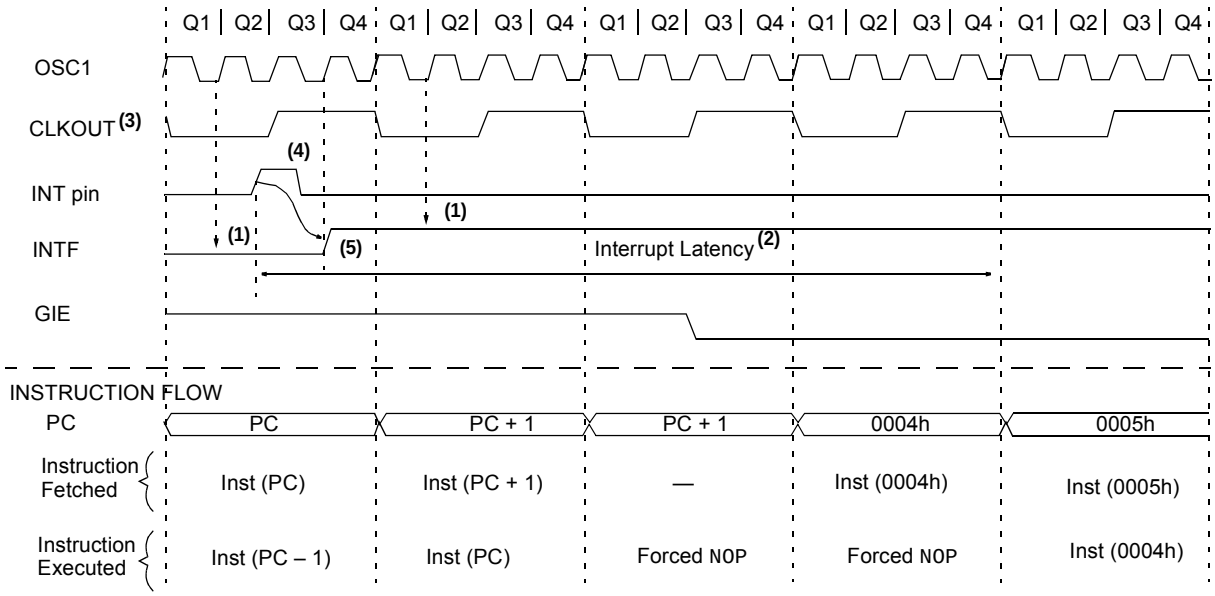


**FIGURE 7-3: INT PIN INTERRUPT TIMING**




**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 36.0 "Electrical Specifications"**".

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

## 7.3 Interrupts During Sleep

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

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

## 7.4 INT Pin

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

## 7.5 Automatic Context Saving

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

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

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

## 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(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 | |

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)
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 IOCxF registers 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. |
|---|---|

**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 | SSP1IE | 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:** Analog-to-Digital Converter (ADC) Interrupt Enable bit
1 = Enables the ADC interrupt
0 = Disables the ADC interrupt

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

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

bit 3 **SSP1IE:** Synchronous Serial Port (MSSP) Interrupt Enable bit
1 = Enables the MSSP interrupt
0 = Disables the MSSP interrupt

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

bit 1 **TMR2IE:** TMR2 to T2PR Match Interrupt Enable bit
1 = Enables the Timer2 to T2PR match interrupt
0 = Disables the Timer2 to T2PR 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 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| OSFIE | C2IE | C1IE | COG1IE | BCL1IE | C4IE | 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 **COG1IE:** COG1 Auto-Shutdown Interrupt Enable bit
1 = COG1 interrupt enabled
0 = COG1 interrupt disabled

bit 3 **BCL1IE:** MSSP Bus Collision Interrupt Enable bit
1 = Enables the MSSP Bus Collision interrupt
0 = Disables the MSSP Bus Collision interrupt

bit 2 **C4IE:** TMR6 to T6PR Match Interrupt Enable bit
1 = Enables the Comparator C4 interrupt
0 = Disables the Comparator C4 interrupt

bit 1 **C3IE:** TMR4to T4PR Match 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.

**REGISTER 7-4: PIE3: PERIPHERAL INTERRUPT ENABLE 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 |
|---|---|---|---|---|---|---|---|
| — | — | COG2IE | ZCDIE | CLC4IE | CLC3IE | CLC2IE | CLC1IE |
| bit 7 | | | | | | | bit 0 |

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

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

bit 5 **COG2IE:** COG2 Auto-Shutdown Interrupt Enable bit
- 1 = COG2 interrupt enabled
- 0 = COG2 interrupt disabled

bit 4 **ZCDIE:** Zero-Cross Detection Interrupt Enable bit
- 1 = ZCD interrupt enabled
- 0 = ZCD interrupt disabled

bit 3 **CLC4IE:** CLC4 Interrupt Enable bit
- 1 = CLC4 interrupt enabled
- 0 = CLC4 interrupt disabled

bit 2 **CLC3IE:** CLC3 Interrupt Enable bit
- 1 = CLC3 interrupt enabled
- 0 = CLC3 interrupt disabled

bit 1 **CLC2IE:** CLC2 Interrupt Enable bit
- 1 = CLC2 interrupt enabled
- 0 = CLC2 interrupt disabled

bit 0 **CLC1IE:** CLC1 Interrupt Enable bit
- 1 = CLC1 interrupt enabled
- 0 = CLC1 interrupt disabled

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

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

| 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 |
|---|---|---|---|---|---|---|---|
| — | TMR8IE | TMR5GIE | TMR5IE | TMR3GIE | TMR3IE | TMR6IE | TRM4IE |
| 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 **TMR8IE:** TMR8 to T8PR Match Interrupt Enable bit
1 = Enables the Timer8 to T8PR match interrupt
0 = Disables the Timer8 to T8PR match interrupt

bit 5 **TMR5GIE:** Timer5 Gate Interrupt Enable bit
1 = Enables the Timer5 gate acquisition interrupt
0 = Disables the Timer5 gate acquisition interrupt

bit 4 **TMR5IE:** TMR5 to Overflow Interrupt Enable bit
1 = Enables the Timer5 to T5PR match interrupt
0 = Disables the Timer5 to T5PR match interrupt

bit 3 **TMR3GIE:** Timer3 Gate Interrupt Enable bit
1 = Enables the Timer3 gate acquisition interrupt
0 = Disables the Timer3 gate acquisition interrupt

bit 2 **TMR3IE:** TMR3 to Overflow Interrupt Enable bit
1 = Enables the Timer3 to T3PR match interrupt
0 = Disables the Timer3 to T3PR match interrupt

bit 1 **TMR6IE:** TMR6 to T6PR Match Interrupt Enable bit
1 = Enables the Timer6 to T6PR match interrupt
0 = Disables the Timer6 to T6PR match interrupt

bit 0 **TMR4IE:** TMR4 to T4PR Match Interrupt Enable bit
1 = Enables the Timer4 to T4PR match interrupt
0 = Disables the Timer4 to T4PR match interrupt

**REGISTER 7-6: PIE5: PERIPHERAL INTERRUPT ENABLE REGISTER 5**

| U-0 | R/W-0/0 | U-0 | R/W-0/0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | CCP7IE | — | COG3IE | — | — | C6IE | C5IE |
| 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 **CCP7IE:** CCP7 Interrupt Enable bit
- 1 = Enables the CCP7 interrupt
- 0 = Disables the CCP7 interrupt

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

bit 4 **COG3IE:** COG3 Auto-Shutdown Interrupt Enable bit
- 1 = COG3 interrupt enabled
- 0 = COG3 interrupt disabled

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

bit 1 **C6IE:** Comparator C6 Interrupt Enable bit
- 1 = Enables the Comparator C6 interrupt
- 0 = Disables the Comparator C6 interrupt

bit 0 **C5IE:** Comparator C5 Interrupt Enable bit
- 1 = Enables the Comparator C5 interrupt
- 0 = Disables the Comparator C5 interrupt

**REGISTER 7-7: PIE6: PERIPHERAL INTERRUPT ENABLE REGISTER 6**

| 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 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | PWM11IE | PWM6IE | PWM5IE |
| 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 **PWM11IE:** PWM11 Interrupt Enable bit
1 = PWM11 interrupt enabled
0 = PWM11 interrupt disabled

bit 1 **PWM6IE:** PWM6 Interrupt Enable bit
1 = PWM6 interrupt enabled
0 = PWM6 interrupt disabled

bit 0 **PWM5IE:** PWM5 Interrupt Enable bit
1 = PWM5 interrupt enabled
0 = PWM5 interrupt disabled

**REGISTER 7-8: 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 | SSP1IF | 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:** Analog-to-Digital Converter (ADC) Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

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

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

bit 3 **SSP1IF:** Synchronous Serial Port (MSSP) 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 T2PR 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 7-9: PIR2: PERIPHERAL INTERRUPT REQUEST REGISTER 2**

| 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 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| OSFIF | C2IF | C1IF | COG1IF | BCL1IF | C4IF | C3IF | 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 **COG1IF:** COG1 Auto-Shutdown Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 3 **BCL1IF:** MSSP Bus Collision Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 2 **C4IF:** Comparator C4 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 1 **C3IF:** Comparator C3 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

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.

**REGISTER 7-10: 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 |
|---|---|---|---|---|---|---|---|
| — | — | COG2IF | ZCDIF | CLC4IF | CLC3IF | CLC2IF | CLC1IF |
| bit 7 | | | | | | | bit 0 |

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

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

bit 5 **COG2IF:** COG2 Auto-Shutdown Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 4 **ZCDIF:** Zero-Cross Detection Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 3 **CLC4IF:** CLC4 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 2 **CLC3IF:** CLC3 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 1 **CLC2IF:** CLC2 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 0 **CLC1IF:** CLC1 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 7-11: PIR4: PERIPHERAL INTERRUPT REQUEST REGISTER 4**

| 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 |
|---|---|---|---|---|---|---|---|
| — | TMR8IF | TMR5GIF | TMR5IF | TMR3GIF | TMR3IF | TMR6IF | TRM4IF |
| 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 **TMR8IF:** TMR8 to T8PR Match Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 5 **TMR5GIF:** Timer5 Gate Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 4 **TMR5IF:** TMR5 to Overflow Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 3 **TMR3GIF:** Timer3 Gate Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 2 **TMR3IF:** TMR3 to Overflow Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 1 **TMR6IF:** TMR6 to T6PR Match Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 0 **TMR4IF:** TMR4 to T4PR Match Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

**REGISTER 7-12: PIR5: PERIPHERAL INTERRUPT REQUEST REGISTER 5**

| U-0 | R/W-0/0 | U-0 | R/W-0/0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | CCP7IF | — | COG3IF | — | — | C6IF | C5IF |
| 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 **CCP7IF:** CCP7 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 5 **Unimplemented:** Read as ‘0’

bit 4 **COG3IF:** COG3 Auto-Shutdown Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

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

bit 1 **C6IF:** Comparator C6 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 0 **C5IF:** Comparator C5 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

**REGISTER 7-13: PIR6: PERIPHERAL INTERRUPT REQUEST REGISTER 6**

| 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 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | PWM11IF | PWM6IF | PWM5IF |
| 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 **PWM11IF:** PWM11 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 1 **PWM6IF:** PWM6 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

bit 0 **PWM5IF:** PWM5 Interrupt Flag bit
1 = Interrupt is pending
0 = Interrupt is not pending

**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 | 108 |
| OPTION_REG | $\overline{\text{WPUEN}}$ | INTEDG | TMR0CS | TMR0SE | PSA | PS<2:0> | | | 236 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 109 |
| PIE2 | OSFIE | C2IE | C1IE | COG1IE | BCL1IE | C4IE | C3IE | CCP2IE | 110 |
| PIE3 | — | — | COG2IE | ZCDIE | CLC4IE | CLC3IE | CLC2IE | CLC1IE | 111 |
| PIE4 | — | TMR8IE | TMR5GIE | TMR5IE | TMR3GIE | TMR3IE | TMR6IE | TRM4IE | 112 |
| PIE5 | — | CCP7IE | — | COG3IE | — | — | C6IE | C5IE | 113 |
| PIE6 | — | — | — | — | — | PWM11IE | PWM6IE | PWM5IE | 114 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 115 |
| PIR2 | OSFIF | C2IF | C1IF | COG1IF | BCL1IF | C4IF | C3IF | CCP2IF | 116 |
| PIR3 | — | — | COG2IF | ZCDIF | CLC4IF | CLC3IF | CLC2IF | CLC1IF | 117 |
| PIR4 | — | TMR8IF | TMR5GIF | TMR5IF | TMR3GIF | TMR3IF | TMR6IF | TRM4IF | 118 |
| PIR5 | — | CCP7IF | — | COG3IF | — | — | C6IF | C5IF | 119 |
| PIR6 | — | — | — | — | — | PWM11IF | PWM6IF | PWM5IF | 120 |

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

## 8.0 POWER-DOWN MODE (SLEEP)

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

Upon entering Sleep mode, the following conditions exist:

1. WDT will be cleared but keeps running, if enabled for operation during Sleep.
2. $\overline{PD}$ bit of the STATUS register is cleared.
3. $\overline{TO}$ bit of the STATUS register is set.
4. CPU clock is disabled.
5. 31 kHz LFINTOSC is unaffected and peripherals that operate from it may continue operation in Sleep.
6. Timer1 and peripherals that operate from Timer1 continue operation in Sleep when the Timer1 clock source selected is:
   - LFINTOSC
   - T1CKI
   - Secondary oscillator
7. ADC is unaffected, if the dedicated FRC oscillator is selected.
8. I/O ports maintain the status they had before SLEEP was executed (driving high, low or high-impedance).
9. Resets other than WDT are not affected by Sleep mode.

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

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

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

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

Examples of internal circuitry that might be sourcing current include modules such as the DAC and FVR modules. See **Section 17.0 "5-Bit Digital-to-Analog Converter (DAC) Module"** and **Section 14.0 "Fixed Voltage Reference (FVR)"** for more information on these modules.

### 8.1 Wake-up from Sleep

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

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

The first three events will cause a device Reset. The last three events are considered a continuation of program execution. To determine whether a device Reset or wake-up event occurred, refer to **Section 6.12 "Determining the Cause of a Reset"**.

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

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

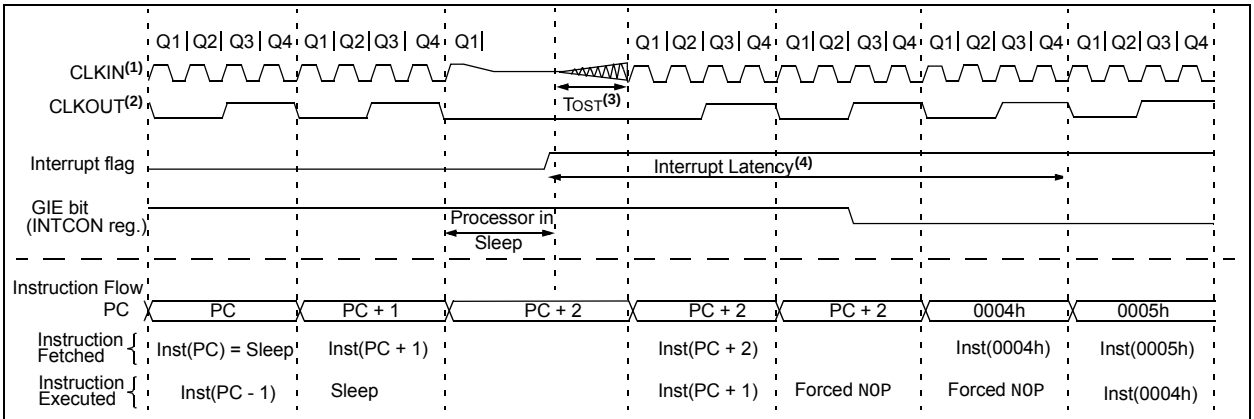
### 8.1.1 WAKE-UP USING INTERRUPTS

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

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

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

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




**Note 1:** External clock. High, Medium, Low mode assumed.
**2:** CLKOUT is shown here for timing reference.
**3:** TOST = 1024 TOSC. This delay does not apply to EC, RC and INTOSC Oscillator modes or Two-Speed Start-up (see **Section 5.4 "Two-Speed Clock Start-up Mode"**.
**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.

## 8.2 Low-Power Sleep Mode

The PIC16F1773/6 devices contain an internal Low Dropout (LDO) voltage regulator, which allows the device I/O pins to operate at voltages up to 5.5V while the internal device logic operates at a lower voltage. The LDO and its associated reference circuitry must remain active when the device is in Sleep mode. The PIC16F1773/6 allow the user to optimize the operating current in Sleep, depending on the application requirements.

A Low-Power Sleep mode can be selected by setting the VREGPM bit of the VREGCON register. With this bit set, the LDO and reference circuitry are placed in a low-power state when the device is in Sleep.

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

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

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

### 8.2.2 PERIPHERAL USAGE IN SLEEP

Some peripherals that can operate in Sleep mode will not operate properly with the Low-Power Sleep mode selected. The Low-Power Sleep mode is intended for use with the following peripherals only:

- Brown-Out Reset (BOR)
- Watchdog Timer (WDT)
- External interrupt pin/interrupt-on-change pins
- Timer1 (with external clock source < 100 kHz)

| | |
|---|---|
| **Note:** | The PIC16LF1773/6 do not have a configurable Low-Power Sleep mode. PIC16LF1773/6 are unregulated devices and are always in the lowest power state when in Sleep, with no wake-up time penalty. These devices have a lower maximum $V_{DD}$ and I/O voltage than the PIC16F1773/6. See **Section 36.0 "Electrical Specifications"** for more information. |

## 8.3 Register Definitions: Voltage Regulator Control

**REGISTER 8-1: VREGCON: VOLTAGE REGULATOR CONTROL REGISTER(1)**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-1/1 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | VREGPM | Reserved |
| bit 7 | | | | | | | bit 0 |

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

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

bit 1 **VREGPM:** Voltage Regulator Power Mode Selection bit
- 1 = Low-Power Sleep mode enabled in Sleep(2)
  Draws lowest current in Sleep, slower wake-up
- 0 = Normal Power mode enabled in Sleep(2)
  Draws higher current in Sleep, faster wake-up

bit 0 **Reserved:** Read as '1'. Maintain this bit set.

**Note 1:** PIC16F1773/6 only.
**2:** See **Section 36.0 "Electrical Specifications"**.

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

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 108 |
| IOCAP | IOCAP7 | IOCAP6 | IOCAP5 | IOCAP4 | IOCAP3 | IOCAP2 | IOCAP1 | IOCAP0 | 179 |
| IOCAN | IOCAN7 | IOCAN6 | IOCAN5 | IOCAN4 | IOCAN3 | IOCAN2 | IOCAN1 | IOCAN0 | 179 |
| IOCAF | IOCAF7 | IOCAF6 | IOCAF5 | IOCAF4 | IOCAF3 | IOCAF2 | IOCAF1 | IOCAF0 | 179 |
| IOCBP | IOCBP7 | IOCBP6 | IOCBP5 | IOCBP4 | IOCBP3 | IOCBP2 | IOCBP1 | IOCBP0 | 179 |
| IOCBN | IOCBN7 | IOCBN6 | IOCBN5 | IOCBN4 | IOCBN3 | IOCBN2 | IOCBN1 | IOCBN0 | 179 |
| IOCBF | IOCBF7 | IOCBF6 | IOCBF5 | IOCBF4 | IOCBF3 | IOCBF2 | IOCBF1 | IOCBF0 | 179 |
| IOCCP | IOCCP7 | IOCCP6 | IOCCP5 | IOCCP4 | IOCCP3 | IOCCP2 | IOCCP1 | IOCCP0 | 180 |
| IOCCN | IOCCN7 | IOCCN6 | IOCCN5 | IOCCN4 | IOCCN3 | IOCCN2 | IOCCN1 | IOCCN0 | 180 |
| IOCCF | IOCCF7 | IOCCF6 | IOCCF5 | IOCCF4 | IOCCF3 | IOCCF2 | IOCCF1 | IOCCF0 | 180 |
| IOCEP | — | — | — | — | IOCEP3 | — | — | — | 182 |
| IOCEN | — | — | — | — | IOCEN3 | — | — | — | 182 |
| IOCEF | — | — | — | — | IOCEF3 | — | — | — | 183 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 109 |
| PIE2 | OSFIE | C2IE | C1IE | COG1IE | BCL1IE | C4IE | C3IE | CCP2IE | 110 |
| PIE3 | — | — | COG2IE | ZCDIE | CLC4IE | CLC3IE | CLC2IE | CLC1IE | 111 |
| PIE4 | — | TMR8IE | TMR5GIE | TMR5IE | TMR3GIE | TMR3IE | TMR6IE | TRM4IE | 112 |
| PIE5 | — | CCP7IE | — | COG3IE | — | — | C6IE | C5IE | 113 |
| PIE6 | — | — | — | — | — | PWM11IE | PWM6IE | PWM5IE | 114 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 115 |
| PIR2 | OSFIF | C2IF | C1IF | COG1IF | BCL1IF | C4IF | C3IF | CCP2IF | 116 |
| PIR3 | — | — | COG2IF | ZCDIF | CLC4IF | CLC3IF | CLC2IF | CLC1IF | 117 |
| PIR4 | — | TMR8IF | TMR5GIF | TMR5IF | TMR3GIF | TMR3IF | TMR6IF | TRM4IF | 118 |
| PIR5 | — | CCP7IF | — | COG3IF | — | — | C6IF | C5IF | 119 |
| PIR6 | — | — | — | — | — | PWM11IF | PWM6IF | PWM5IF | 120 |
| STATUS | — | — | — | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | 26 |
| VREGCON(1) | — | — | — | — | — | — | VREGPM(2) | Reserved | 125 |
| WDTCON | — | — | WDTPS<4:0> | | | | | SWDTEN | 130 |

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

**Note 1:** PIC16(L)F1776 only.

**2:** Unimplemented on PIC16LF1776.

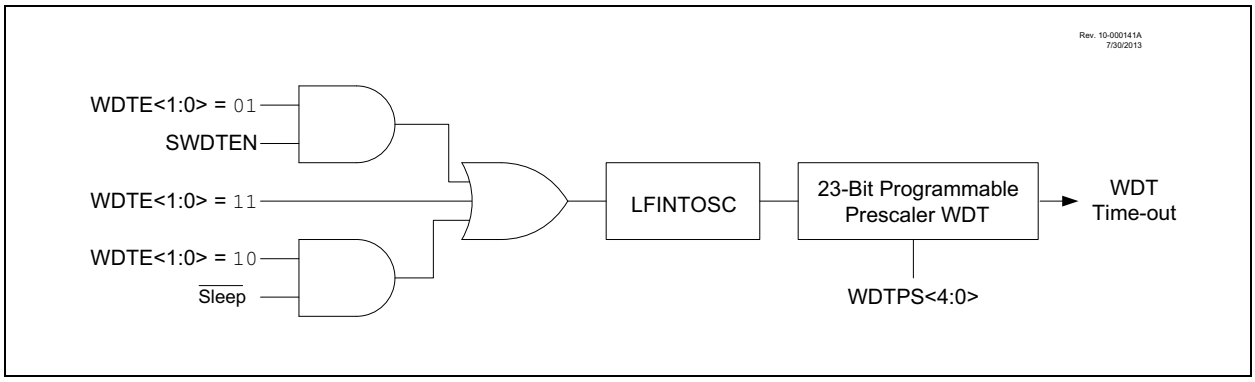
## 9.0 WATCHDOG TIMER (WDT)

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

The WDT has the following features:

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

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



## 9.1 Independent Clock Source

The WDT derives its time base from the 31 kHz LFINTOSC internal oscillator. Time intervals in this chapter are based on a nominal interval of 1 ms. See Table 36-8: Oscillator Parameters for the LFINTOSC specification.

## 9.2 WDT Operating Modes

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

### 9.2.1 WDT IS ALWAYS ON

When the WDTE bits of Configuration Words are set to '11', the WDT is always on.

WDT protection is active during Sleep.

### 9.2.2 WDT IS OFF IN SLEEP

When the WDTE bits of Configuration Words are set to '10', the WDT is on, except in Sleep.

WDT protection is not active during Sleep.

### 9.2.3 WDT CONTROLLED BY SOFTWARE

When the WDTE bits of Configuration Words are set to '01', the WDT is controlled by the SWDTEN bit of the WDTCON register.

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

**TABLE 9-1: WDT OPERATING MODES**

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

## 9.3 Time-Out Period

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

## 9.4 Clearing the WDT

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

- Any Reset
- CLRWDT instruction is executed
- Device enters Sleep
- Device wakes up from Sleep
- Oscillator fail
- WDT is disabled
- Oscillator Start-up Timer (OST) is running

See Table 9-2 for more information.

## 9.5 Operation During Sleep

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

When the device exits Sleep, the WDT is cleared again. 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 STATUS Register (Register 3-1) for more information.

**TABLE 9-2: WDT CLEARING CONDITIONS**

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

## 9.6 Register Definitions: Watchdog Control

**REGISTER 9-1: WDTCON: WATCHDOG TIMER CONTROL REGISTER**

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

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

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

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

Bit Value = Prescale Rate

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

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

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

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

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

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

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| OSCCON | SPLLEN | IRCF<3:0> | | | | — | SCS<1:0> | | 92 |
| STATUS | — | — | — | $\overline{TO}$ | $\overline{PD}$ | Z | DC | C | 26 |
| WDTCON | — | — | WDTPS<4:0> | | | | | SWDTEN | 130 |

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

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

| Name | Bits | Bit -/7 | Bit -/6 | Bit 13/5 | Bit 12/4 | Bit 11/3 | Bit 10/2 | Bit 9/1 | Bit 8/0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|---|
| CONFIG1 | 13:8 | — | — | FCMEN | IESO | $\overline{CLKOUTEN}$ | BOREN<1:0> | | — | 71 |
| | 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.

## 10.0 FLASH PROGRAM MEMORY CONTROL

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

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

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

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

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

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

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

| | |
|---|---|
| **Note 1:** | Code protection of the entire Flash program memory array is enabled by clearing the $\overline{CP}$ bit of Configuration Words. |

## 10.1 PMADRL and PMADRH Registers

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

### 10.1.1 PMCON1 AND PMCON2 REGISTERS

PMCON1 is the control register for Flash program memory accesses.

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

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

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

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

## 10.2 Flash Program Memory Overview

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

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

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

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

**TABLE 10-1: FLASH MEMORY ORGANIZATION BY DEVICE**

| Device | Row Erase (words) | Write Latches (words) |
|---|---|---|
| PIC16(L)F1773 | 32 | 32 |
| PIC16(L)F1776 | | |

### 10.2.1 READING THE FLASH PROGRAM MEMORY

To read a program memory location, the user must:

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

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

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

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

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

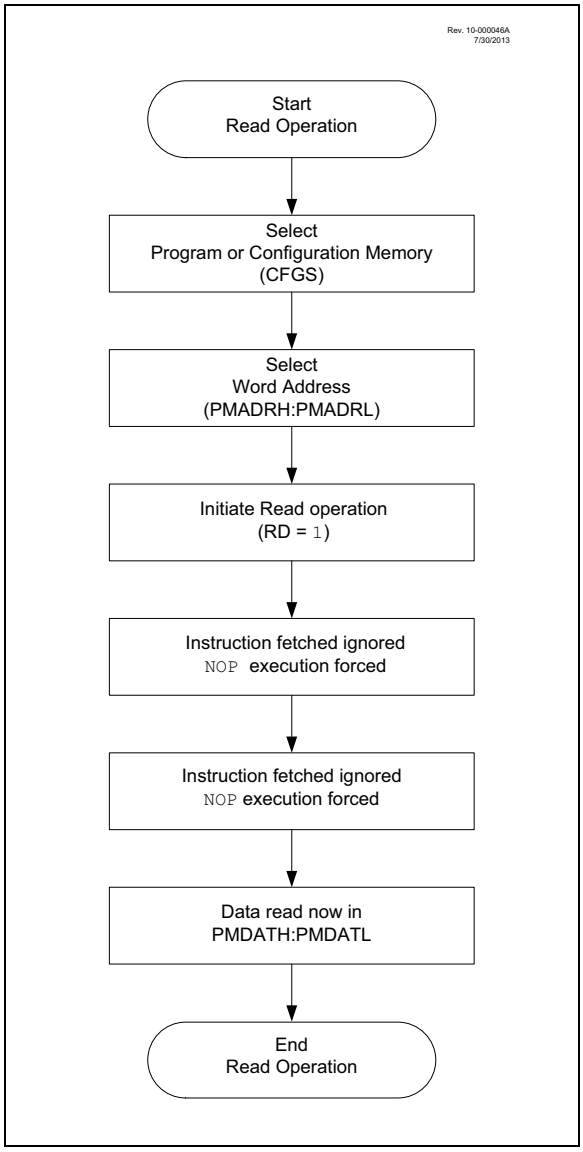
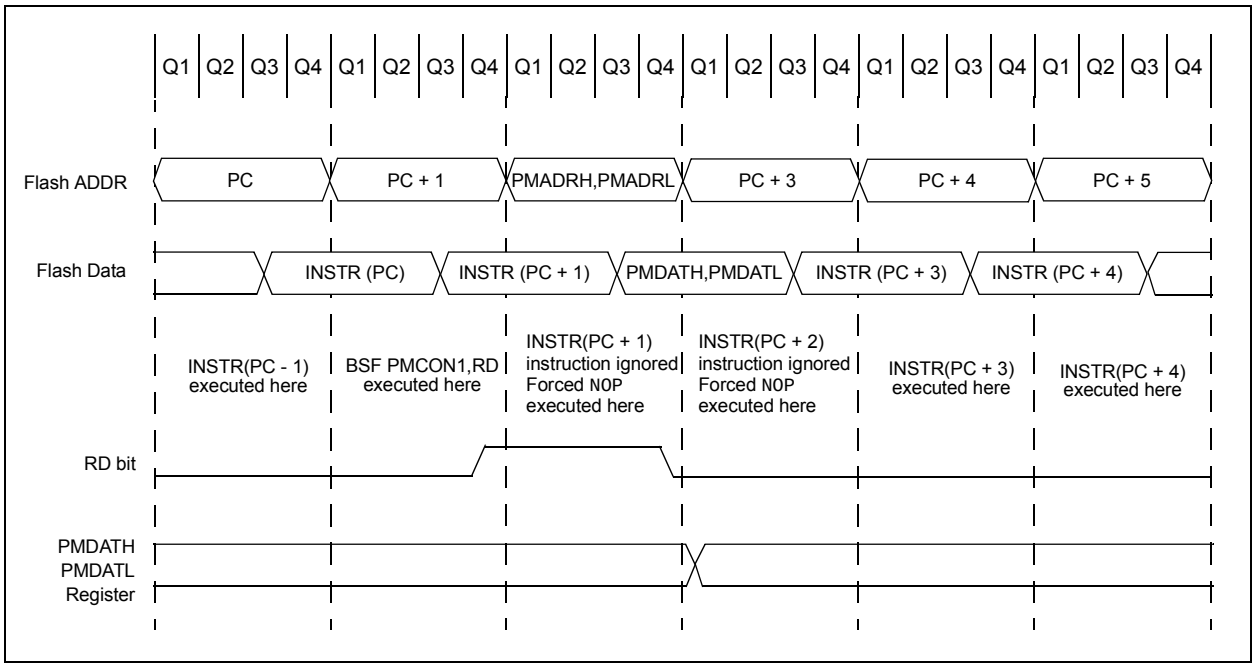


**FIGURE 10-2: FLASH PROGRAM MEMORY READ CYCLE EXECUTION**




**EXAMPLE 10-1: FLASH PROGRAM MEMORY READ**

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

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

    BCF      PMCON1,CFGS      ; Do not select Configuration Space
    BSF      PMCON1,RD        ; Initiate read
    NOP                       ; Ignored (Figure 10-1)
    NOP                       ; Ignored (Figure 10-1)

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

### 10.2.2 FLASH MEMORY UNLOCK SEQUENCE

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

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

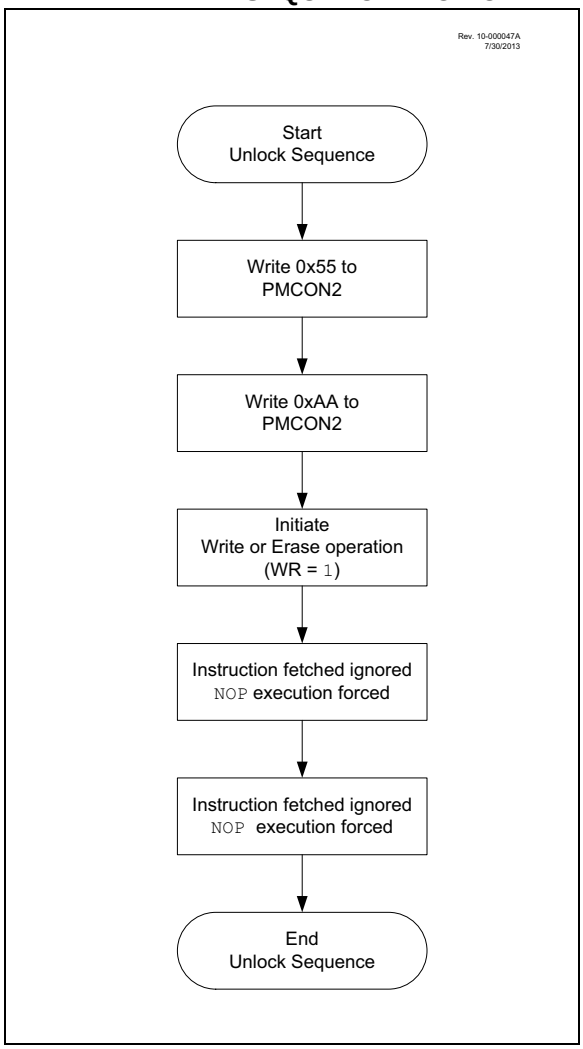
The unlock sequence consists of the following steps:

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

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

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

**FIGURE 10-3: FLASH PROGRAM MEMORY UNLOCK SEQUENCE FLOWCHART**



### 10.2.3 ERASING FLASH PROGRAM MEMORY

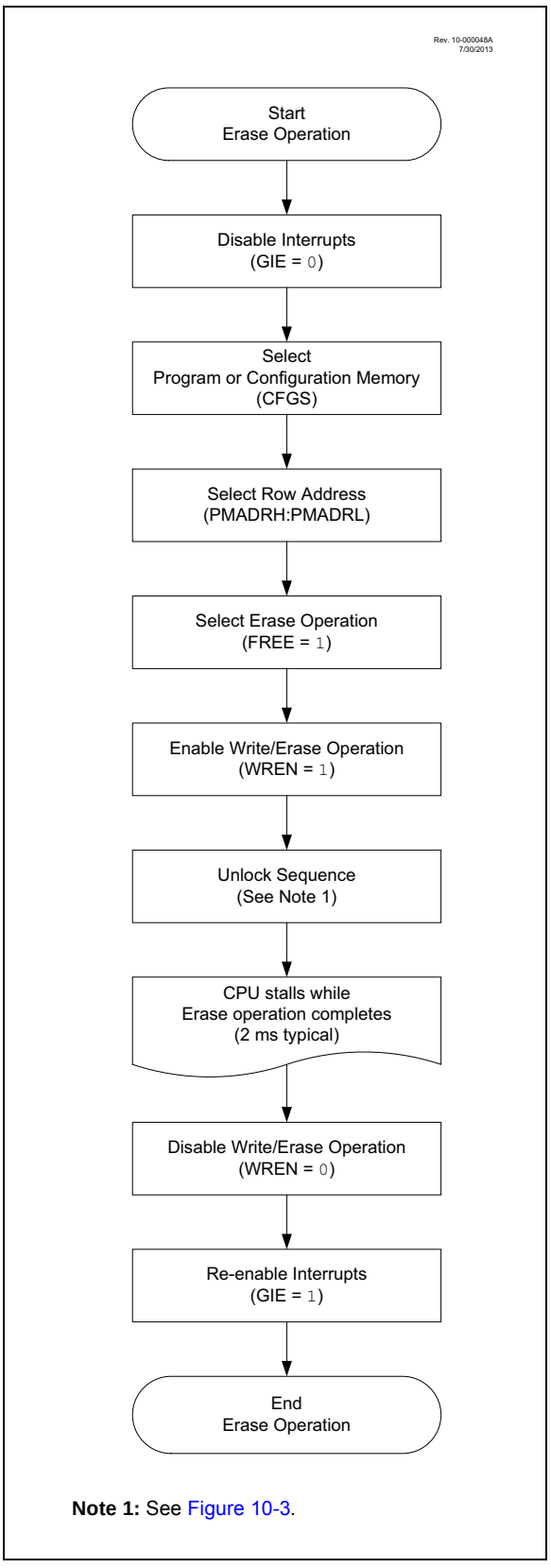
While executing code, program memory can only be erased by rows. To erase a row:

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

See Example 10-2.

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

**FIGURE 10-4: FLASH PROGRAM MEMORY ERASE FLOWCHART**




**Note 1:** See Figure 10-3.

**EXAMPLE 10-2: ERASING ONE ROW OF PROGRAM MEMORY**

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


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

Required
Sequence
        MOVLW       55h             ; Start of required sequence to initiate erase
        MOVWF       PMCON2          ; Write 55h
        MOVLW       0AAh            ;
        MOVWF       PMCON2          ; Write AAh
        BSF         PMCON1,WR       ; Set WR bit to begin erase
        NOP                         ; NOP instructions are forced as processor starts
        NOP                         ; row erase of program memory.
                                    ;
                                    ; The processor stalls until the erase process is complete
                                    ; after erase processor continues with 3rd instruction

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

### 10.2.4 WRITING TO FLASH PROGRAM MEMORY

Program memory is programmed using the following steps:

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

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

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

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

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

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

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

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

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

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

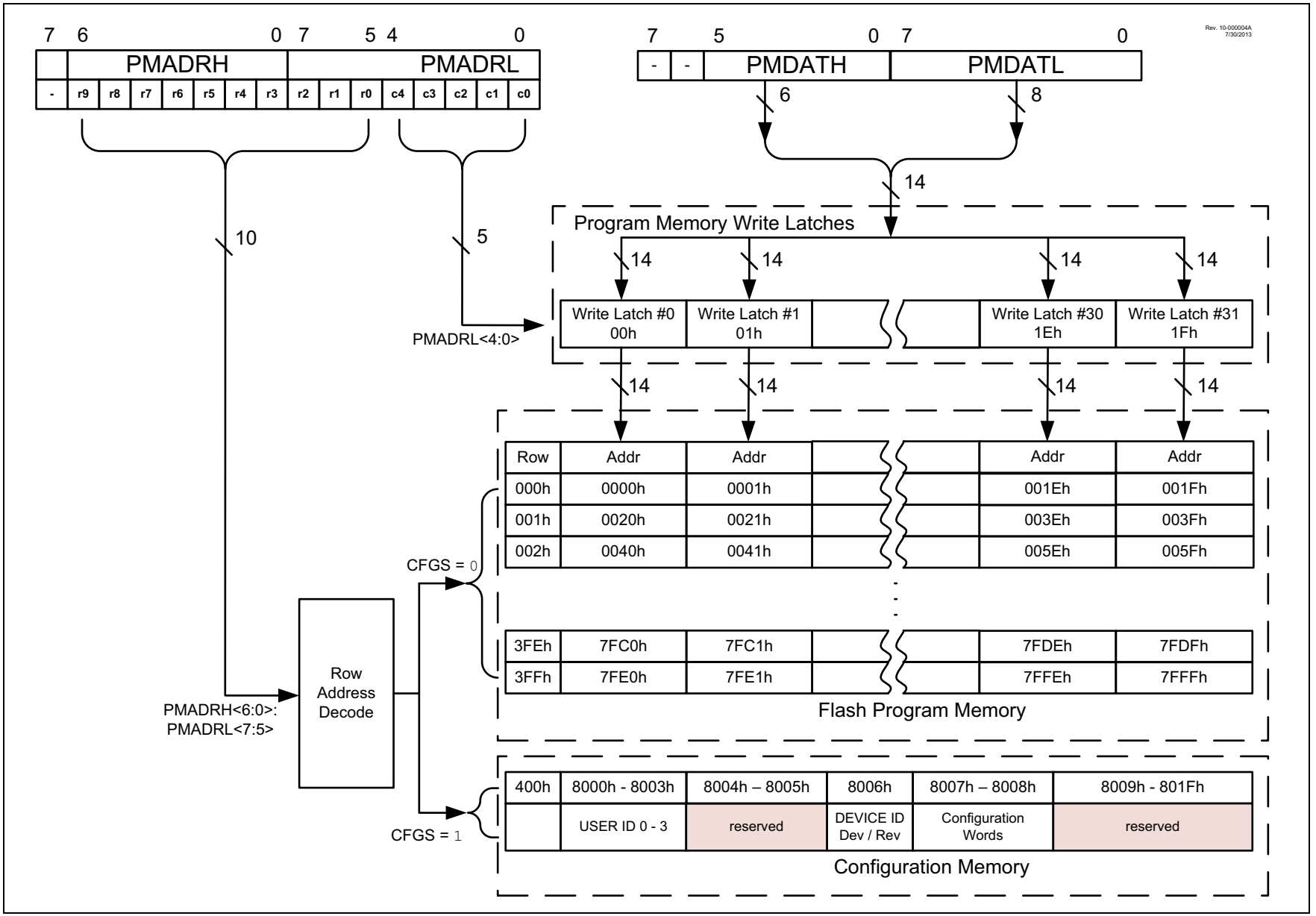
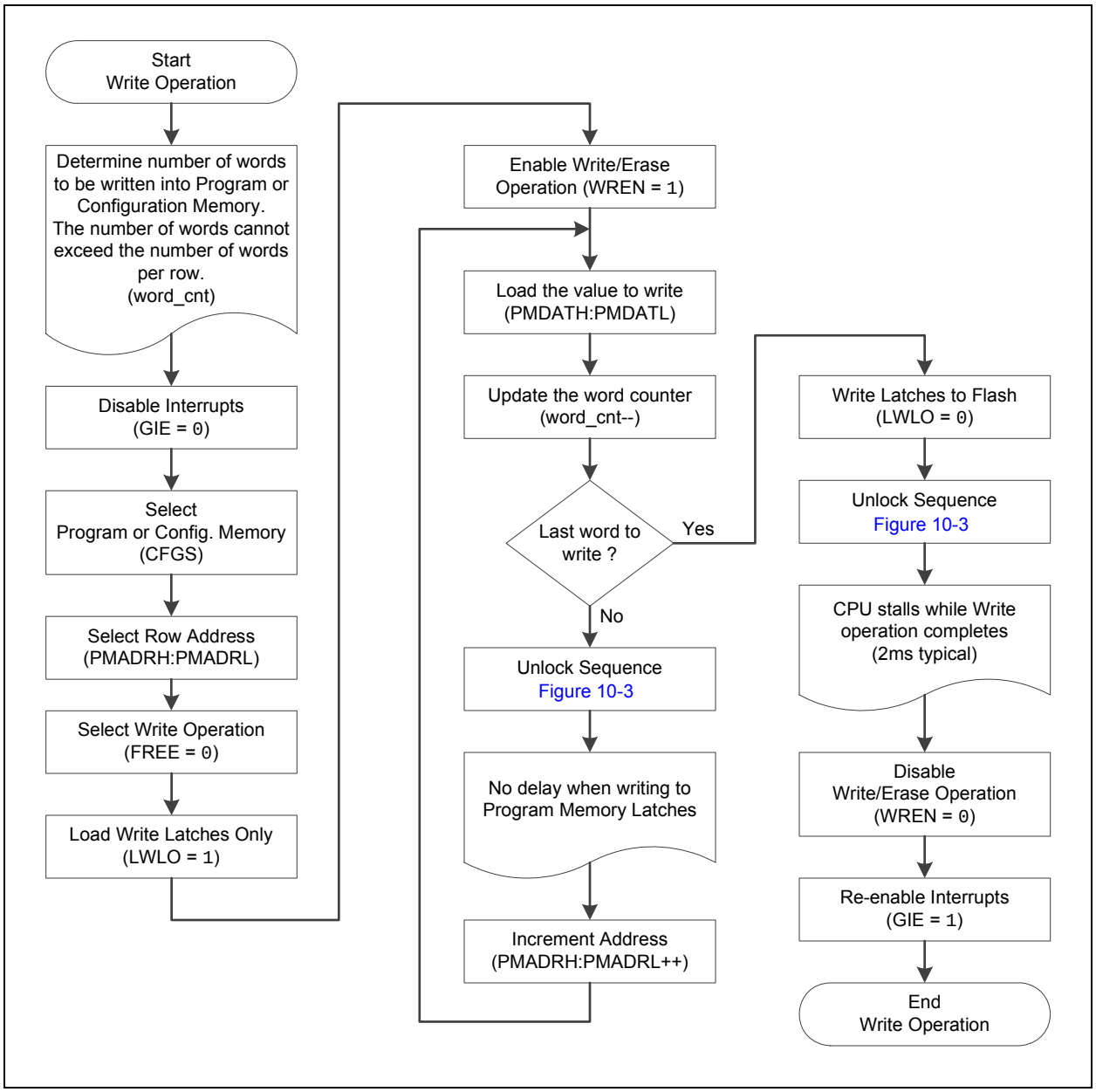


**FIGURE 10-6: FLASH PROGRAM MEMORY WRITE FLOWCHART**



**EXAMPLE 10-3: WRITING TO FLASH PROGRAM MEMORY**

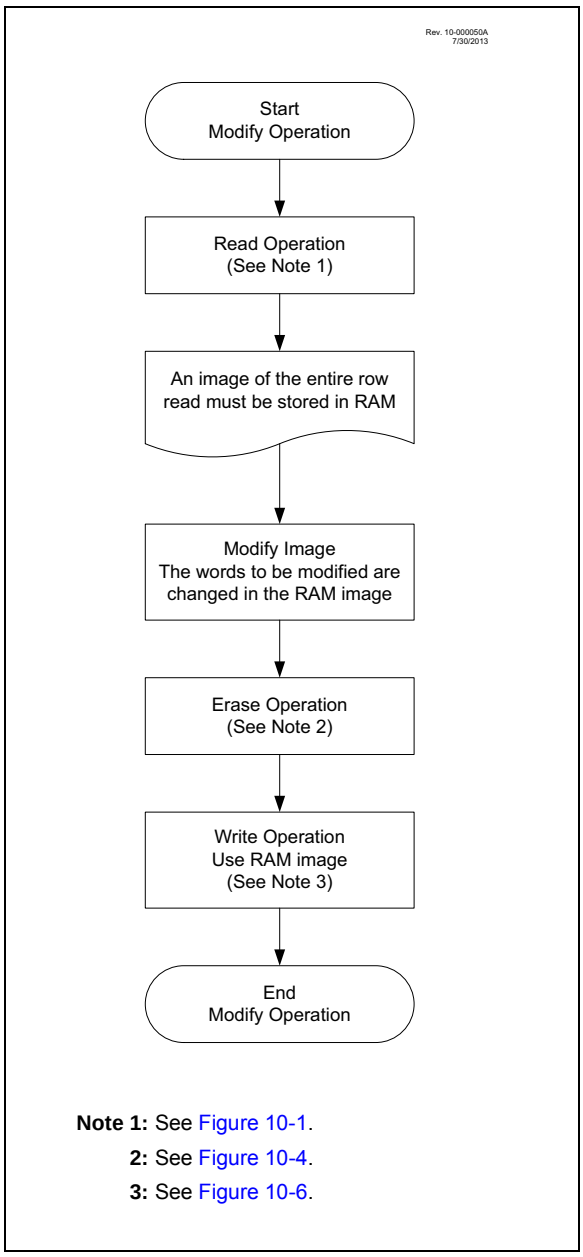
```
; This write routine assumes the following:
; 1. 64 bytes of data are loaded, starting at the address in DATA_ADDR
; 2. Each word of data to be written is made up of two adjacent bytes in DATA_ADDR,
; stored in little endian format
; 3. A valid starting address (the least significant bits = 00000)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

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

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

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

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

  Required Sequence
        MOVLW       55h             ; Start of required write sequence:
        MOVWF       PMCON2          ; Write 55h
        MOVLW       0AAh            ;
        MOVWF       PMCON2          ; Write AAh
        BSF         PMCON1,WR       ; Set WR bit to begin write
        NOP                         ; NOP instructions are forced as processor writes
                                    ; all the program memory write latches simultaneously
        NOP                         ; to program memory.
                                    ; After NOPs, the processor
                                    ; stalls until the self-write process in complete
                                    ; after write processor continues with 3rd instruction
        BCF         PMCON1,WREN     ; Disable writes
        BSF         INTCON,GIE      ; Enable interrupts
```

## 10.3 Modifying Flash Program Memory

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

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

**FIGURE 10-7: FLASH PROGRAM MEMORY MODIFY FLOWCHART**




**Note 1:** See Figure 10-1.
**2:** See Figure 10-4.
**3:** See Figure 10-6.

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

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

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

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

| Address | Function | Read Access | Write Access |
|---|---|---|---|
| 8000h-8003h | User IDs | Yes | Yes |
| 8005h-8006h | Device ID/Revision ID | Yes | No |
| 8007h-8008h | Configuration Words 1 and 2 | Yes | No |

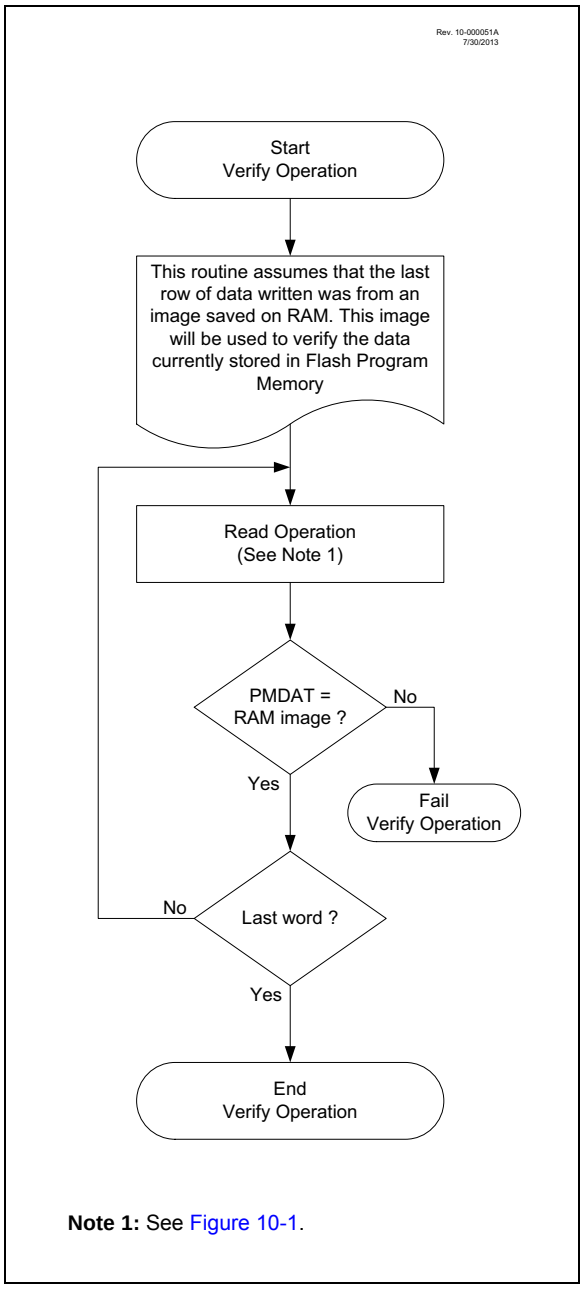
**EXAMPLE 10-4: CONFIGURATION WORD AND DEVICE ID ACCESS**

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

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

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

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

## 10.5 Write/Verify

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

**FIGURE 10-8: FLASH PROGRAM MEMORY VERIFY FLOWCHART**




**Note 1:** See Figure 10-1.

## 10.6 Register Definitions: Flash Program Memory Control

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

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

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

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

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

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

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

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

bit 5-0 **PMDAT<13:8>**: Read/write value for Most Significant bits of program memory

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

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

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

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

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

| U-1 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| __(1) | PMADR<14:8> | | | | | | |
| bit 7 | | | | | | | bit 0 |

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

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

bit 6-0 **PMADR<14:8>**: Specifies the Most Significant bits for program memory address

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

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

| U-1 | R/W-0/0 | R/W-0/0 | R/W/HC-0/0 | R/W/HC-x/q(2) | R/W-0/0 | R/S/HC-0/0 | R/S/HC-0/0 |
|---|---|---|---|---|---|---|---|
| —(1) | CFGS | LWLO(3) | FREE | WRERR | WREN | WR | RD |
| bit 7 | | | | | | | bit 0 |

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

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

bit 6 **CFGS:** Configuration Select bit
1 = Access Configuration, User ID and Device ID Registers
0 = Access Flash program memory

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

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

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

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

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

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

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

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

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

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

bit 7-0 **Flash Memory Unlock Pattern bits**

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

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

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 108 |
| PMCON1 | —(1) | CFGS | LWLO | FREE | WRERR | WREN | WR | RD | 147 |
| PMCON2 | Program Memory Control Register 2 | | | | | | | | 148 |
| PMADRL | PMADRL<7:0> | | | | | | | | 145 |
| PMADRH | —(1) | PMADRH<6:0> | | | | | | | 146 |
| PMDATL | PMDATL<7:0> | | | | | | | | 145 |
| PMDATH | — | — | PMDATH<5:0> | | | | | | 145 |

**Legend:** — = unimplemented location, read as '0'. Shaded cells are not used by Flash program memory.
**Note 1:** Unimplemented, read as '1'.

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

| Name | Bits | Bit -/7 | Bit -/6 | Bit 13/5 | Bit 12/4 | Bit 11/3 | Bit 10/2 | Bit 9/1 | Bit 8/0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|---|
| CONFIG1 | 13:8 | — | — | FCMEN | IESO | $\overline{\text{CLKOUTEN}}$ | BOREN<1:0> | | — | 71 |
| | 7:0 | $\overline{\text{CP}}$ | MCLRE | $\overline{\text{PWRTE}}$ | WDTE<1:0> | | FOSC<2:0> | | | |
| CONFIG2 | 13:8 | — | — | LVP | $\overline{\text{DEBUG}}$ | $\overline{\text{LPBOR}}$ | BORV | STVREN | PLLEN | 73 |
| | 7:0 | $\overline{\text{ZCD}}$ | — | — | — | — | PPS1WAY | WRT<1:0> | | |

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

## 11.0 I/O PORTS

Each port has six 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)
- INLVLx (input level control)
- ODCONx registers (open-drain)
- SLRCONx registers (slew rate

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

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

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

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

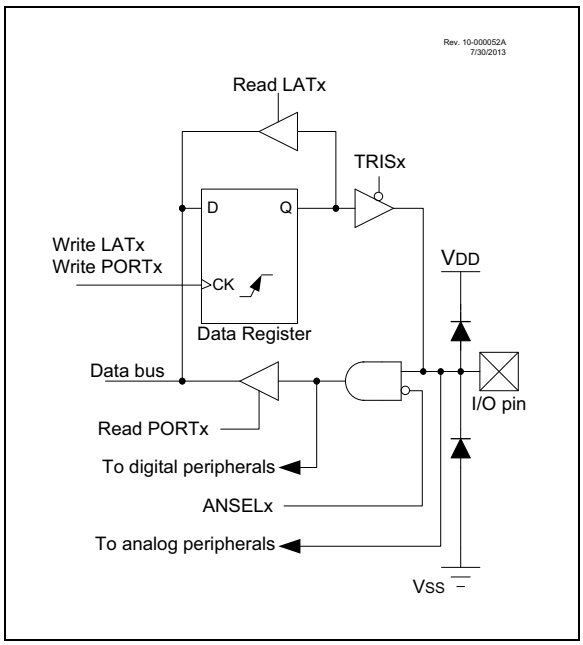
| Device | PORTA | PORTB | PORTC | PORTE |
|---|---|---|---|---|
| PIC16(L)F1773 | • | • | • | • |
| PIC16(L)F1776 | • | • | • | • |

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

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

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

**FIGURE 11-1: GENERIC I/O PORT OPERATION**



## 11.1 PORTA Registers

### 11.1.1 DATA REGISTER

PORTA is a 6-bit wide, bidirectional port. The corresponding data direction register is TRISA (Register 11-2). Setting a TRISA bit (= 1) will make the corresponding PORTA pin an input (i.e., disable the output driver). Clearing a TRISA bit (= 0) will make the corresponding PORTA pin an output (i.e., enables output driver and puts the contents of the output latch on the selected pin). The exception is RA3, which is input-only and its TRIS bit will always read as '1'. Example 11-1 shows how to initialize PORTA.

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

### 11.1.2 DIRECTION CONTROL

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

### 11.1.3 OPEN-DRAIN CONTROL

The ODCONA register (Register 11-6) controls the open-drain feature of the port. Open-drain operation is independently selected for each pin. When an ODCONA bit is set, the corresponding port output becomes an open-drain driver capable of sinking current only. When an ODCONA bit is cleared, the corresponding port output pin is the standard push-pull drive capable of sourcing and sinking current.

### 11.1.4 SLEW RATE CONTROL

The SLRCONA register (Register 11-7) controls the slew rate option for each port pin. Slew rate control is independently selectable for each port pin. When an SLRCONA bit is set, the corresponding port pin drive is slew rate limited. When an SLRCONA bit is cleared, The corresponding port pin drive slews at the maximum rate possible.

### 11.1.5 INPUT THRESHOLD CONTROL

The INLVLA register (Register 11-8) controls the input voltage threshold for each of the available PORTA input pins. A selection between the Schmitt Trigger CMOS or the TTL Compatible thresholds is available. The input threshold is important in determining the value of a read of the PORTA register and also the level at which an interrupt-on-change occurs, if that feature is enabled. See Table 36-4: I/O Ports for more information on threshold levels.

| | |
|---|---|
| **Note:** | Changing the input threshold selection should be performed while all peripheral modules are disabled. Changing the threshold level during the time a module is active may inadvertently generate a transition associated with an input pin, regardless of the actual voltage level on that pin. |

### 11.1.6 ANALOG CONTROL

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

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

| | |
|---|---|
| **Note:** | The ANSELA bits default to the Analog mode after reset. To use any pins as digital general purpose or peripheral inputs, the corresponding ANSEL bits must be initialized to '0' by user software. |

**EXAMPLE 11-1: INITIALIZING PORTA**

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

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

### 11.1.7 PORTA FUNCTIONS AND OUTPUT PRIORITIES

Each PORTA pin is multiplexed with other functions.

Each pin defaults to the PORT latch data after reset. Other functions are selected with the peripheral pin select logic. See **Section 12.0 "Peripheral Pin Select (PPS) Module"** for more information.

Analog input functions, such as ADC and comparator inputs are not shown in the peripheral pin select lists. These inputs are active when the I/O pin is set for Analog mode using the ANSELA register. Digital output functions may continue to control the pin when it is in Analog mode.

## 11.2 Register Definitions: PORTA

**REGISTER 11-1: PORTA: PORTA REGISTER**

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

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

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

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

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

| R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/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 11-3: LATA: PORTA DATA LATCH REGISTER**

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

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

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

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

**REGISTER 11-4: ANSELA: PORTA ANALOG SELECT REGISTER**

| U-0 | U-0 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 | R/W-1/1 |
|---|---|---|---|---|---|---|---|
| — | — | ANSA5 | ANSA4 | 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 5-0 **ANSA<5:0>:** Analog Select between Analog or Digital Function on pins RA<5:0>

1 = Analog input. Pin is assigned as analog input(1). Digital input buffer disabled.

0 = Digital I/O. Pin is assigned to port or digital special function.

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

**REGISTER 11-5: WPUA: WEAK PULL-UP PORTA 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 |
|---|---|---|---|---|---|---|---|
| WPUA7 | WPUA6 | WPUA5 | WPUA4 | WPUA3 | WPUA2 | WPUA1 | WPUA0 |
| bit 7 | | | | | | | bit 0 |

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

bit 7-0 **WPUA<7:0>**: Weak Pull-up Register bits(1),(2)

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 configured as an output.

**REGISTER 11-6: ODCONA: PORTA OPEN-DRAIN 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 |
|---|---|---|---|---|---|---|---|
| ODA7 | ODA6 | ODA5 | ODA4 | ODA3 | ODA2 | ODA1 | ODA0 |
| 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 **ODA<7:0>:** PORTA Open-Drain Enable bits
For RA<7:0> pins
1 = Port pin operates as open-drain drive (sink current only)
0 = Port pin operates as standard push-pull drive (source and sink current)

**REGISTER 11-7: SLRCONA: PORTA SLEW RATE CONTROL 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 |
|---|---|---|---|---|---|---|---|
| SLRA7 | SLRA6 | SLRA5 | SLRA4 | SLRA3 | SLRA2 | SLRA1 | SLRA0 |
| 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 **SLRA<7:0>:** PORTA Slew Rate Enable bits
For RA<7:0> pins
1 = Port pin slew rate is limited
0 = Port pin slews at maximum rate

**REGISTER 11-8: INLVLA: PORTA INPUT LEVEL 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 |
|---|---|---|---|---|---|---|---|
| INLVLA7 | INLVLA6 | INLVLA5 | INLVLA4 | INLVLA3 | INLVLA2 | INLVLA1 | INLVLA0 |
| 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 **INLVLA<7:0>:** PORTA Input Level Select bits
For RA<7:0> pins
1 = Port pin digital input operates with ST thresholds
0 = Port pin digital input operates with TTL thresholds

**TABLE 11-2: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| ANSELA | — | — | ANSA5 | ANSA4 | ANSA3 | ANSA2 | ANSA1 | ANSA0 | 153 |
| INLVLA | INLVLA7 | INLVLA6 | INLVLA5 | INLVLA4 | INLVLA3 | INLVLA2 | INLVLA1 | INLVLA0 | 154 |
| LATA | LATA7 | LATA6 | LATA5 | LATA4 | LATA3 | LATA2 | LATA1 | LATA0 | 152 |
| ODCONA | ODA7 | ODA6 | ODA5 | ODA4 | ODA3 | ODA2 | ODA1 | ODA0 | 154 |
| OPTION_REG | $\overline{\text{WPUEN}}$ | INTEDG | TMR0CS | TMR0SE | PSA | PS<2:0> | | | 236 |
| PORTA | RA7 | RA6 | RA5 | RA4 | RA3 | RA2 | RA1 | RA0 | 152 |
| SLRCONA | SLRA7 | SLRA6 | SLRA5 | SLRA4 | SLRA3 | SLRA2 | SLRA1 | SLRA0 | 154 |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 152 |
| WPUA | WPUA7 | WPUA6 | WPUA5 | WPUA4 | WPUA3 | WPUA2 | WPUA1 | WPUA0 | 153 |

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

**TABLE 11-3: 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> | | — | 71 |
| | 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.

## 11.3 PORTB Registers

### 11.3.1 DATA REGISTER

PORTB is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISB (Register 11-10). Setting a TRISB bit (= 1) will make the corresponding PORTB pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISB bit (= 0) will make the corresponding PORTB pin an output (i.e., enable the output driver and put the contents of the output latch on the selected pin). Example 11-1 shows how to initialize an I/O port.

Reading the PORTB register (Register 11-9) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch (LATB).

### 11.3.2 DIRECTION CONTROL

The TRISB register (Register 11-10) 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'.

### 11.3.3 OPEN-DRAIN CONTROL

The ODCONB register (Register 11-14) controls the open-drain feature of the port. Open-drain operation is independently selected for each pin. When an ODCONB bit is set, the corresponding port output becomes an open-drain driver capable of sinking current only. When an ODCONB bit is cleared, the corresponding port output pin is the standard push-pull drive capable of sourcing and sinking current.

### 11.3.4 SLEW RATE CONTROL

The SLRCONB register (Register 11-15) controls the slew rate option for each port pin. Slew rate control is independently selectable for each port pin. When an SLRCONB bit is set, the corresponding port pin drive is slew rate limited. When an SLRCONB bit is cleared, The corresponding port pin drive slews at the maximum rate possible.

### 11.3.5 INPUT THRESHOLD CONTROL

The INLVLB register (Register 11-16) controls the input voltage threshold for each of the available PORTB input pins. A selection between the Schmitt Trigger CMOS or the TTL Compatible thresholds is available. The input threshold is important in determining the value of a read of the PORTB register and also the level at which an interrupt-on-change occurs, if that feature is enabled. See Table 36-4: I/O Ports for more information on threshold levels.

| | |
|---|---|
| **Note:** | Changing the input threshold selection should be performed while all peripheral modules are disabled. Changing the threshold level during the time a module is active may inadvertently generate a transition associated with an input pin, regardless of the actual voltage level on that pin. |

### 11.3.6 ANALOG CONTROL

The ANSELB register (Register 11-12) is used to configure the Input mode of an I/O pin to analog. Setting the appropriate ANSELB bit high will cause all digital reads on the pin to be read as '0' and allow analog functions on the pin to operate correctly.

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

| | |
|---|---|
| **Note:** | The ANSELB bits default to the Analog mode after reset. To use any pins as digital general purpose or peripheral inputs, the corresponding ANSEL bits must be initialized to '0' by user software. |

### 11.3.7 PORTB FUNCTIONS AND OUTPUT PRIORITIES

Each pin defaults to the PORT latch data after reset. Other functions are selected with the peripheral pin select logic. See **Section 12.0 "Peripheral Pin Select (PPS) Module"** for more information. Analog input functions, such as ADC and op amp inputs, are not shown in the peripheral pin select lists. These inputs are active when the I/O pin is set for Analog mode using the ANSELB register. Digital output functions may continue to control the pin when it is in Analog mode.

### 11.3.8 HIGH CURRENT DRIVE CONTROL

The output drivers on RB1 and RB0 are capable of sourcing and sinking up to 100 mA. This extra drive capacity can be enabled and disabled with the control bits in the HIDRVB register (Register 11-17).

## 11.4 Register Definitions: PORTB

**REGISTER 11-9: PORTB: PORTB REGISTER**

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

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

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

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

**REGISTER 11-10: TRISB: PORTB TRI-STATE REGISTER**

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

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

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

**REGISTER 11-11: LATB: PORTB DATA LATCH REGISTER**

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

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

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

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

**REGISTER 11-12: ANSELB: PORTB ANALOG SELECT REGISTER**

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

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

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

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

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

**REGISTER 11-13: 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 7-4 **WPUB<7:0>**: Weak Pull-up Register bits(1,2)
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 configured as an output.

**REGISTER 11-14: ODCONB: PORTB OPEN-DRAIN 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 |
|---|---|---|---|---|---|---|---|
| ODB7 | ODB6 | ODB5 | ODB4 | ODB3 | ODB2 | ODB1 | ODB0 |
| 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 **ODB<7:0>:** PORTB Open-Drain Enable bits
For RB<7:0> pins
1 = Port pin operates as open-drain drive (sink current only)
0 = Port pin operates as standard push-pull drive (source and sink current)

**REGISTER 11-15: SLRCONB: PORTB SLEW RATE CONTROL 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 |
|---|---|---|---|---|---|---|---|
| SLRB7 | SLRB6 | SLRB5 | SLRB4 | SLRB3 | SLRB2 | SLRB1 | SLRB0 |
| 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 **SLRB<7:0>:** PORTB Slew Rate Enable bits
For RB<7:0> pins
1 = Port pin slew rate is limited
0 = Port pin slews at maximum rate

**REGISTER 11-16: INLVLB: PORTB INPUT LEVEL 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 |
|---|---|---|---|---|---|---|---|
| INLVLB7 | INLVLB6 | INLVLB5 | INLVLB4 | INLVLB3 | INLVLB2 | INLVLB1 | INLVLB0 |
| 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 **INLVLB<7:0>:** PORTB Input Level Select bits
For RB<7:0> pins
1 = Port pin digital input operates with ST thresholds
0 = Port pin digital input operates with TTL thresholds

**REGISTER 11-17: HIDRVB: PORTB HIGH DRIVE CONTROL REGISTER**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | HIDB1 | HIDB0 |
| 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 **HIDB<1:0>:** PORTB High Drive Enable bits
For RB<1:0> pins
1 = High current source and sink enabled
0 = Standard current source and sink

**TABLE 11-4: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| ANSELB | — | — | ANSB5 | ANSB4 | ANSB3 | ANSB2 | ANSB1 | ANSB0 | 158 |
| HIDRVB | — | — | — | — | — | — | HIDB1 | HIDB0 | 160 |
| INLVLB | INLVLB7 | INLVLB6 | INLVLB5 | INLVLB4 | INLVLB3 | INLVLB2 | INLVLB1 | INLVLB0 | 159 |
| LATB | LATB7 | LATB6 | LATB5 | LATB4 | LATB3 | LATB2 | LATB1 | LATB0 | 157 |
| ODCONB | ODB7 | ODB6 | ODB5 | ODB4 | ODB3 | ODB2 | ODB1 | ODB0 | 159 |
| PORTB | RB7 | RB6 | RB5 | RB4 | RB3 | RB2 | RB1 | RB0 | 157 |
| SLRCONB | SLRB7 | SLRB6 | SLRB5 | SLRB4 | SLRB3 | SLRB2 | SLRB1 | SLRB0 | 159 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 159 |
| WPUB | WPUB7 | WPUB6 | WPUB5 | WPUB4 | WPUB3 | WPUB2 | WPUB1 | WPUB0 | 158 |

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

## 11.5 PORTC Registers

### 11.5.1 DATA REGISTER

PORTC is an 8-bit wide bidirectional port in the PIC16(L)F1773/6 devices. The corresponding data direction register is TRISC (Register 11-19). Setting a TRISC bit (= 1) will make the corresponding PORTC pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISC bit (= 0) will make the corresponding PORTC pin an output (i.e., enable the output driver and put the contents of the output latch on the selected pin). Example 11-1 shows how to initialize an I/O port.

Reading the PORTC register (Register 11-18) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch (LATC).

### 11.5.2 DIRECTION CONTROL

The TRISC register (Register 11-19) 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'.

### 11.5.3 INPUT THRESHOLD CONTROL

The INLVLC register (Register 11-25) controls the input voltage threshold for each of the available PORTC input pins. A selection between the Schmitt Trigger CMOS or the TTL Compatible thresholds is available. The input threshold is important in determining the value of a read of the PORTC register and also the level at which an interrupt-on-change occurs, if that feature is enabled. See Table 36-4: I/O Ports for more information on threshold levels.

| | |
|---|---|
| **Note:** | Changing the input threshold selection should be performed while all peripheral modules are disabled. Changing the threshold level during the time a module is active may inadvertently generate a transition associated with an input pin, regardless of the actual voltage level on that pin. |

### 11.5.4 OPEN-DRAIN CONTROL

The ODCONC register (Register 11-23) controls the open-drain feature of the port. Open-drain operation is independently selected for each pin. When an ODCONC bit is set, the corresponding port output becomes an open-drain driver capable of sinking current only. When an ODCONC bit is cleared, the corresponding port output pin is the standard push-pull drive capable of sourcing and sinking current.

### 11.5.5 SLEW RATE CONTROL

The SLRCONC register (Register 11-24) controls the slew rate option for each port pin. Slew rate control is independently selectable for each port pin. When an SLRCONC bit is set, the corresponding port pin drive is slew rate limited. When an SLRCONC bit is cleared, The corresponding port pin drive slews at the maximum rate possible.

### 11.5.6 ANALOG CONTROL

The ANSELC register (Register 11-21) is used to configure the Input mode of an I/O pin to analog. Setting the appropriate ANSELC 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 ANSELC bits has no effect on digital output functions. A pin with TRIS clear and ANSELC 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 ANSELC bits default to the Analog mode after reset. To use any pins as digital general purpose or peripheral inputs, the corresponding ANSEL bits must be initialized to '0' by user software. |

### 11.5.7 PORTC FUNCTIONS AND OUTPUT PRIORITIES

Each pin defaults to the PORT latch data after reset. Other functions are selected with the peripheral pin select logic. See **Section 12.0 "Peripheral Pin Select (PPS) Module"** for more information.

Analog input functions, such as ADC and comparator inputs, are not shown in the peripheral pin select lists. These inputs are active when the I/O pin is set for Analog mode using the ANSELC register. Digital output functions may continue to control the pin when it is in Analog mode.

## 11.6 Register Definitions: PORTC

**REGISTER 11-18: PORTC: PORTC REGISTER**

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

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

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

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

**REGISTER 11-19: TRISC: PORTC TRI-STATE REGISTER**

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

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

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

**REGISTER 11-20: 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

**REGISTER 11-21: ANSELC: PORTC 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 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| ANSC7 | ANSC6 | ANSC5 | ANSC4 | ANSC3 | ANSC2 | — | — |
| 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 **ANSC<7:2>**: Analog Select between Analog or Digital Function on pins RC<7:2>**(1)**
1 = Analog input. Pin is assigned as analog input**(1)**. Digital input buffer disabled.
0 = Digital I/O. Pin is assigned to port or digital special function.

bit 1-0 **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 11-22: WPUC: WEAK PULL-UP PORTC 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 |
|---|---|---|---|---|---|---|---|
| WPUC7 | WPUC6 | WPUC5 | WPUC4 | WPUC3 | WPUC2 | WPUC1 | WPUC0 |
| 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 **WPUC<7:0>**: Weak Pull-up Register bits(1, 2)
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 configured as an output.

**REGISTER 11-23: ODCONC: PORTC OPEN-DRAIN 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 |
|---|---|---|---|---|---|---|---|
| ODC7 | ODC6 | ODC5 | ODC4 | ODC3 | ODC2 | ODC1 | ODC0 |
| 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 **ODC<7:0>:** PORTC Open-Drain Enable bits
For RC<7:0> pins
1 = Port pin operates as open-drain drive (sink current only)
0 = Port pin operates as standard push-pull drive (source and sink current)

**REGISTER 11-24: SLRCONC: PORTC SLEW RATE CONTROL 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 |
|---|---|---|---|---|---|---|---|
| SLRC7 | SLRC6 | SLRC5 | SLRC4 | SLRC3 | SLRC2 | SLRC1 | SLRC0 |
| 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 **SLRC<7:0>:** PORTC Slew Rate Enable bits
For RC<7:0> pins
1 = Port pin slew rate is limited
0 = Port pin slews at maximum rate

**REGISTER 11-25: INLVLC: PORTC INPUT LEVEL CONTROL 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 |
|---|---|---|---|---|---|---|---|
| INLVLC7 | INLVLC6 | INLVLC5 | INLVLC4 | INLVLC3 | INLVLC2 | INLVLC1 | INLVLC0 |
| 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 **INLVLC<7:0>:** PORTC Input Level Select bits
For RC<7:0> pins
1 = Port pin digital input operates with ST thresholds
0 = Port pin digital input operates with TTL thresholds

**TABLE 11-5: SUMMARY OF REGISTERS ASSOCIATED WITH PORTC**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| ANSELC | ANSC7 | ANSC6 | ANSC5 | ANSC4 | ANSC3 | ANSC2 | — | — | 163 |
| INLVLC | INLVLC7 | INLVLC6 | INLVLC5 | INLVLC4 | INLVLC3 | INLVLC2 | INLVLC1 | INLVLC0 | 165 |
| LATC | LATC7 | LATC6 | LATC5 | LATC4 | LATC3 | LATC2 | LATC1 | LATC0 | 163 |
| ODCONC | ODC7 | ODC6 | ODC5 | ODC4 | ODC3 | ODC2 | ODC1 | ODC0 | 164 |
| PORTC | RC7 | RC6 | RC5 | RC4 | RC3 | RC2 | RC1 | RC0 | 162 |
| SLRCONC | SLRC7 | SLRC6 | SLRC5 | SLRC4 | SLRC3 | SLRC2 | SLRC1 | SLRC0 | 165 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 162 |
| WPUC | WPUC7 | WPUC6 | WPUC5 | WPUC4 | WPUC3 | WPUC2 | WPUC1 | WPUC0 | 164 |

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

## 11.7 PORTE Registers

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

### 11.7.1 INPUT THRESHOLD CONTROL

The INLVLE register (Register 11-29) controls the input voltage threshold for each of the available PORTE input pins. A selection between the Schmitt Trigger CMOS or the TTL Compatible thresholds is available. The input threshold is important in determining the value of a read of the PORTE register and also the level at which an interrupt-on-change occurs, if that feature is enabled. See Table 36-1 for more information on threshold levels.

| | |
|---|---|
| **Note:** | Changing the input threshold selection should be performed while all peripheral modules are disabled. Changing the threshold level during the time a module is active may inadvertently generate a transition associated with an input pin, regardless of the actual voltage level on that pin. |

### 11.7.2 PORTE FUNCTIONS AND OUTPUT PRIORITIES

No output priorities, RE3 is an input-only pin.

**REGISTER 11-26: PORTE: PORTE REGISTER**

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

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

bit 7-4 **Unimplemented**: Read as '0'

bit 3 **RE3**: PORTE Input Pin bit
1 = Port pin is > $V_{IH}$
0 = Port pin is < $V_{IL}$

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

## 11.8 Register Definitions: PORTE

**REGISTER 11-27: TRISE: PORTE TRI-STATE REGISTER**

| U-0 | U-0 | U-0 | U-0 | U-1[(1)] | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

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

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

bit 3 **Unimplemented**: Read as ‘1’

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

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

**REGISTER 11-28: WPUE: WEAK PULL-UP PORTE REGISTER**

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

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

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

bit 3 **WPUE3:** Weak Pull-up Register bit
1 = Pull-up enabled
0 = Pull-up disabled

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

**REGISTER 11-29: INLVLE: PORTE INPUT LEVEL CONTROL REGISTER**

| U-0 | U-0 | U-0 | U-0 | R/W-1/1 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | INLVLE3 | — | — | — |
| bit 7 | | | | | | | bit 0 |

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

bit 7-4 **Unimplemented:** Read as '0'

bit 3 **INLVLE3:** PORTE Input Level Select bit
1 = ST input used for PORT reads and interrupt-on-change
0 = TTL input used for PORT reads and interrupt-on-change

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

**TABLE 11-6: 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 |
|---|---|---|---|---|---|---|---|---|---|
| INLVLE | — | — | — | — | INLVLE3 | — | — | — | 169 |
| PORTE | — | — | — | — | RE3 | — | — | — | 167 |
| TRISE | — | — | — | — | —(1) | — | — | — | 168 |
| WPUE | — | — | — | — | WPUE3 | — | — | — | 168 |

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

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

## 12.0 PERIPHERAL PIN SELECT (PPS) MODULE

The Peripheral Pin Select (PPS) module connects peripheral inputs and outputs to the device I/O pins. Only digital signals are included in the selections. All analog inputs and outputs remain fixed to their assigned pins. Input and output selections are independent as shown in the simplified block diagram Figure 12-1.

### 12.1 PPS Inputs

Each peripheral has a PPS register with which the inputs to the peripheral are selected. Inputs include the device pins.

Multiple peripherals can operate from the same source simultaneously. Port reads always return the pin level regardless of peripheral PPS selection. If a pin also has associated analog functions, the ANSEL bit for that pin must be cleared to enable the digital input buffer.

Although every peripheral has its own PPS input selection register, the selections are identical for every peripheral as shown in Register 12-1.

| **Note:** | The notation “xxx” in the register name is a place holder for the peripheral identifier. For example, CLC1PPS. |
|---|---|

### 12.2 PPS Outputs

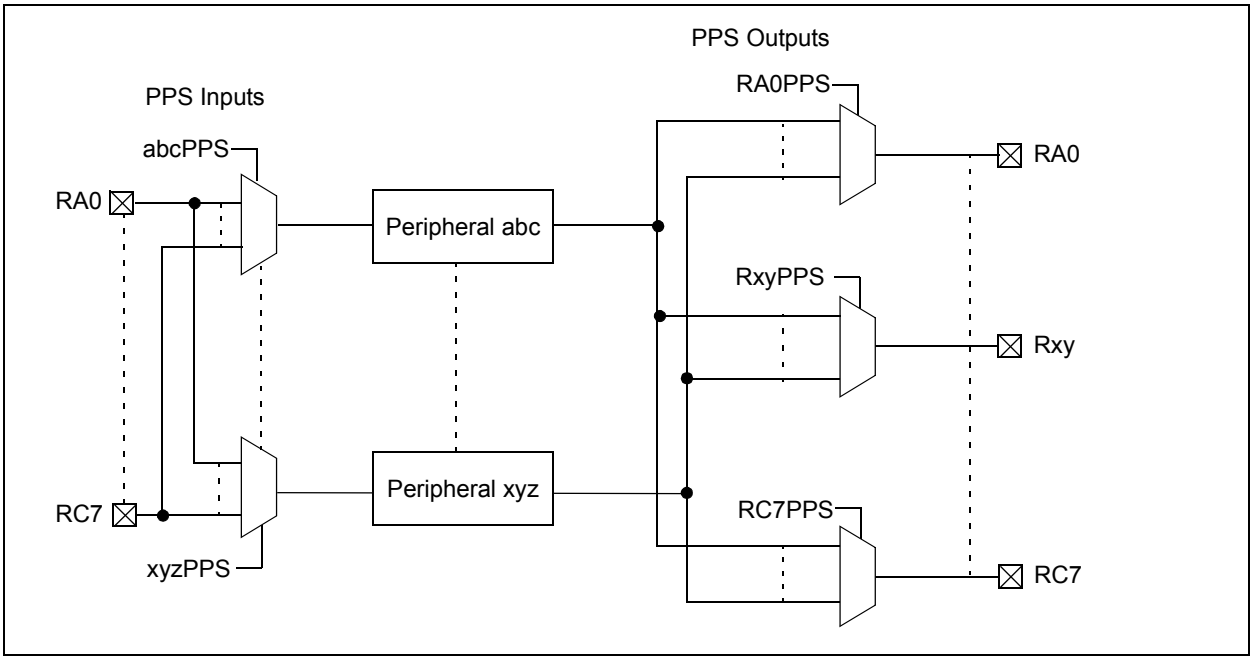
Each I/O pin has a PPS register with which the pin output source is selected. With few exceptions, the port TRIS control associated with that pin retains control over the pin output driver. Peripherals that control the pin output driver as part of the peripheral operation will override the TRIS control as needed. These peripherals include:

- EUSART (synchronous operation)
- MSSP ($I^2C$)
- COG (auto-shutdown)

Although every pin has its own PPS peripheral selection register, the selections are identical for every pin as shown in Register 12-2.

| **Note:** | The notation “Rxy” is a place holder for the pin identifier. For example, RA0PPS. |
|---|---|

**FIGURE 12-1: SIMPLIFIED PPS BLOCK DIAGRAM**



## 12.3 Bidirectional Pins

PPS selections for peripherals with bidirectional signals on a single pin must be made so that the PPS input and PPS output select the same pin. Peripherals that have bidirectional signals include:

- EUSART (synchronous operation)
- MSSP ($I^2C$)

| **Note:** | The $I^2C$ default input pins are $I^2C$ and SMBus compatible and are the only pins on the device with this compatibility. |
|---|---|

## 12.4 PPS Lock

The PPS includes a mode in which all input and output selections can be locked to prevent inadvertent changes. PPS selections are locked by setting the PPSLOCKED bit of the PPSLOCK register. Setting and clearing this bit requires a special sequence as an extra precaution against inadvertent changes. Examples of setting and clearing the PPSLOCKED bit are shown in Example 12-1.

**EXAMPLE 12-1: PPS LOCK/UNLOCK SEQUENCE**

```
; suspend interrupts
    bcf     INTCON,GIE
;   BANKSEL PPSLOCK     ; set bank
; required sequence, next 5 instructions
    movlw   0x55
    movwf   PPSLOCK
    movlw   0xAA
    movwf   PPSLOCK
; Set PPSLOCKED bit to disable writes or
; Clear PPSLOCKED bit to enable writes
    bsf     PPSLOCK,PPSLOCKED
; restore interrupts
    bsf     INTCON,GIE
```

## 12.5 PPS Permanent Lock

The PPS can be permanently locked by setting the PPS1WAY Configuration bit. When this bit is set, the PPSLOCKED bit can only be cleared and set one time after a device Reset. This allows for clearing the PPSLOCKED bit so that the input and output selections can be made during initialization. When the PPSLOCKED bit is set after all selections have been made, it will remain set and cannot be cleared until after the next device Reset event.

## 12.6 Operation During Sleep

PPS input and output selections are unaffected by Sleep.

## 12.7 Effects of a Reset

A device Power-On Reset (POR) clears all PPS input and output selections to their default values. All other Resets leave the selections unchanged. Default input selections are shown in Table 12-1.

## 12.8 Register Definitions: PPS Input Selection

### REGISTER 12-1: xxxPPS: PERIPHERAL xxx INPUT SELECTION

| U-0 | U-0 | R/W-q/u | R/W-q/u | R/W-q/u | R/W-q/u | R/W-q/u | R/W-q/u |
|---|---|---|---|---|---|---|---|
| — | — | xxxPPS<5:0> | | | | | |
| bit 7 | | | | | | | bit 0 |

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

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

bit 5-3 **xxxPPS<5:3>:** Peripheral xxx Input PORT Selection bits
100 = Peripheral input is PORTE
011 = Reserved
010 = Peripheral input is PORTC
001 = Peripheral input is PORTB
000 = Peripheral input is PORTA

bit 2-0 **xxxPPS<2:0>:** Peripheral xxx Input Bit Selection bits (1)
111 = Peripheral input is from PORTx Bit 7 (Rx7)
110 = Peripheral input is from PORTx Bit 6 (Rx6)
101 = Peripheral input is from PORTx Bit 5 (Rx5)
100 = Peripheral input is from PORTx Bit 4 (Rx4)
011 = Peripheral input is from PORTx Bit 3 (Rx3)
010 = Peripheral input is from PORTx Bit 2 (Rx2)
001 = Peripheral input is from PORTx Bit 1 (Rx1)
000 = Peripheral input is from PORTx Bit 0 (Rx0)

**Note 1:** See Table 12-1 for xxxPPS register list and Reset values.

### REGISTER 12-2: RxyPPS: PIN Rxy OUTPUT SOURCE SELECTION REGISTER

| U-0 | U-0 | R/W-0/u | R/W-0/u | R/W-0/u | R/W-0/u | R/W-0/u | R/W-0/u |
|---|---|---|---|---|---|---|---|
| — | — | RxyPPS<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 **RxyPPS<5:0>:** Pin Rxy Output Source Selection bits
Selection code determines the output signal on the port pin.
See Table 12-2 for the selection codes

**REGISTER 12-3: PPSLOCK: PPS LOCK REGISTER**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | — | PPSLOCKED |
| 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 **Unimplemented:** Read as ‘0’

bit 0 **PPSLOCKED:** PPS Locked bit

1 = PPS is locked. PPS selections can not be changed.

0 = PPS is not locked. PPS selections can be changed.

**TABLE 12-1: PPS INPUT REGISTER RESET VALUES**

| Peripheral | xxxPPS Register | Default Pin Selection | Reset Value (xxxPPS<5:0>) | Available Ports PIC16(L)F1773/6 | | |
|---|---|---|---|---|---|---|
| | | PIC16(L)F1773/6 | PIC16(L)F1773/6 | A | B | C |
| Interrupt-on-change | INTPPS | RB0 | 001000 | ● | ● | — |
| Timer0clock | T0CKIPPS | RA4 | 000100 | ● | ● | — |
| Timer1clock | T1CKIPPS | RC0 | 010000 | ● | — | ● |
| Timer1 gate | T1GPPS | RB5 | 001101 | — | ● | ● |
| Timer3 clock | T3CKIPPS | RC0 | 010000 | — | ● | ● |
| Timer3 gate | T3GPPS | RC0 | 010000 | ● | — | ● |
| Timer5 clock | T5CKIPPS | RC2 | 010010 | ● | — | ● |
| Timer5 gate | T5GPPS | RB4 | 001100 | — | ● | ● |
| Timer2 input | T2INPPS | RC3 | 010011 | ● | — | ● |
| Timer4 input | T4INPPS | RC5 | 010101 | — | ● | ● |
| Timer6 input | T6INPPS | RB7 | 001111 | — | ● | ● |
| Timer8 input | T8INPPS | RC4 | 010100 | — | ● | ● |
| CCP1 | CCP1PPS | RC2 | 010010 | — | ● | ● |
| CCP2 | CCP2PPS | RC1 | 010001 | — | ● | ● |
| CCP7 | CCP7PPS | RB5 | 001101 | — | ● | ● |
| COG1 | COG1INPPS | RB0 | 001000 | — | ● | ● |
| COG2 | COG2INPPS | RB1 | 001001 | — | ● | ● |
| COG3 | COG3INPPS | RB2 | 001010 | — | ● | ● |
| DSM1 low carrier | MD1CLPPS | RA3 | 000011 | ● | — | ● |
| DSM1 high carrier | MD1CHPPS | RA4 | 000100 | ● | — | ● |
| DSM1 modulation | MD1MODPPS | RA5 | 000101 | ● | — | ● |
| DSM2 low carrier | MD2CLPPS | RC3 | 010011 | ● | — | ● |
| DSM2 high carrier | MD2CHPPS | RC4 | 010100 | ● | — | ● |
| DSM2 modulation | MD2MODPPS | RC5 | 010101 | ● | — | ● |
| DSM3 low carrier | MD3CLPPS | RB3 | 001011 | — | ● | ● |
| DSM3 high carrier | MD3CHPPS | RB4 | 001100 | — | ● | ● |
| DSM3 modulation | MD3MODPPS | RB5 | 001101 | — | ● | ● |
| PRG1 set rising | PRG1RPPS | RA4 | 000100 | ● | — | ● |
| PRG1 set falling | PRG1FPPS | RA5 | 000101 | ● | — | ● |
| PRG2 set rising | PRG2RPPS | RC1 | 010001 | ● | — | ● |
| PRG2 set falling | PRG2FPPS | RC2 | 010010 | ● | — | ● |
| PRG3 set rising | PRG3RPPS | RC4 | 010100 | — | ● | ● |
| PRG3 set falling | PRG3FPPS | RC5 | 010101 | — | ● | ● |
| ADC trigger | ADCACTPPS | RB4 | 001100 | — | ● | ● |
| SPI and $I^2C$ clock | SSPCLKPPS | RC3 | 010011 | — | ● | ● |
| SPI and $I^2C$ data | SSPDATPPS | RC4 | 010100 | — | ● | ● |
| SPI slave select | SSPSSPPS | RA5 | 000101 | — | ● | ● |
| EUSART RX | RXPPS | RC7 | 010111 | — | ● | ● |
| EUSART CK | CKPPS | RC6 | 010110 | — | ● | ● |
| All CLCs | CLCIN0PPS | RA0 | 000000 | ● | — | ● |
| All CLCs | CLCIN1PPS | RA1 | 000001 | ● | — | ● |
| All CLCs | CLCIN2PPS | RB6 | 001110 | — | ● | ● |
| All CLCs | CLCIN3PPS | RB7 | 001111 | — | ● | ● |

Example: CCP1PPS = 0x13 selects RC3 as the CCP1 input.

**TABLE 12-2: AVAILABLE PORTS FOR OUTPUT BY PERIPHERAL(2)**

| RxyPPS<5:0> | Output Signal | PIC16(L)F1773/6 | | |
|---|---|---|---|---|
| | | PORTA | PORTB | PORTC |
| 110000 | MD3_out | — | • | • |
| 101111 | MD2_out | • | — | • |
| 101110 | MD1_out | • | — | • |
| 101011 | sync_C6OUT | • | — | • |
| 101010 | sync_C5OUT | • | — | • |
| 101001 | sync_C4OUT | — | • | • |
| 101000 | sync_C3OUT | — | • | • |
| 100111 | sync_C2OUT | • | — | • |
| 100110 | sync_C1OUT | • | — | • |
| 100101 | DT | — | • | • |
| 100100 | TX/CK | — | • | • |
| 100011 | SDO | — | • | • |
| 100010 | SDA | — | • | • |
| 100001 | SCK/SCL(1) | — | • | • |
| 011111 | PWM11_out | — | • | • |
| 011110 | PWM6_out | • | — | • |
| 011101 | PWM5_out | • | — | • |
| 011011 | PWM9_out | • | — | • |
| 011010 | PWM4_out | • | — | • |
| 011001 | PWM3_out | • | — | • |
| 010111 | CCP7_out | — | • | • |
| 010110 | CCP2_out | — | • | • |
| 010101 | CCP1_out | — | • | • |
| 010000 | COG3D(1) | • | — | • |
| 001111 | COG3C(1) | • | — | • |
| 001110 | COG3B(1) | • | — | • |
| 001101 | COG3A(1) | • | — | • |
| 001100 | COG2D(1) | — | • | • |
| 001011 | COG2C(1) | — | • | • |
| 001010 | COG2B(1) | — | • | • |
| 001001 | COG2A(1) | — | • | • |
| 001000 | COG1D(1) | — | • | • |
| 000111 | COG1C(1) | — | • | • |
| 000110 | COG1B(1) | — | • | • |
| 000101 | COG1A(1) | — | • | • |
| 000100 | LC4_out | — | • | • |
| 000011 | LC3_out | — | • | • |
| 000010 | LC2_out | • | — | • |
| 000001 | LC1_out | • | — | • |
| 000000 | LATxy | • | — | • |

**Note 1:** TRIS control is overridden by the peripheral as required.

**2:** Unsupported peripherals will output a '0'.

**TABLE 12-3: SUMMARY OF REGISTERS ASSOCIATED WITH THE PPS MODULE**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on page |
|---|---|---|---|---|---|---|---|---|---|
| PPSLOCK | — | — | — | — | — | — | — | PPSLOCKED | 173 |
| INTPPS | — | — | INTPPS<5:0> | | | | | | 172 |
| T0CKIPPS | — | — | T0CKIPPS<5:0> | | | | | | 172 |
| T1CKIPPS | — | — | T1CKIPPS<5:0> | | | | | | 172 |
| T1GPPS | — | — | T1GPPS<5:0> | | | | | | 172 |
| T3CKIPPS | — | — | T3CKIPPS<5:0> | | | | | | 172 |
| T3GPPS | — | — | T3GPPS<5:0> | | | | | | 172 |
| T5CKIPPS | — | — | T5CKIPPS<5:0> | | | | | | 172 |
| T5GPPS | — | — | T5GPPS<5:0> | | | | | | 172 |
| T2INPPS | — | — | T2INPPS<5:0> | | | | | | 172 |
| T4INPPS | — | — | T4INPPS<5:0> | | | | | | 172 |
| T6INPPS | — | — | T6INPPS<5:0> | | | | | | 172 |
| T8INPPS | — | — | T8INPPS<5:0> | | | | | | 172 |
| CCP1PPS | — | — | CCP1PPS<5:0> | | | | | | 172 |
| CCP2PPS | — | — | CCP2PPS<5:0> | | | | | | 172 |
| CCP7PPS | — | — | CCP7PPS<5:0> | | | | | | 172 |
| COGIN1PPS | — | — | COG1PPS<5:0> | | | | | | 172 |
| COG2INPPS | — | — | COG2PPS<5:0> | | | | | | 172 |
| COG3INPPS | — | — | COG3PPS<5:0> | | | | | | 172 |
| MD1CLPPS | — | — | MD1CLPPS<5:0> | | | | | | 172 |
| MD1CHPPS | — | — | MD1CHPPS<5:0> | | | | | | 172 |
| MD1MODPPS | — | — | MD1MODPPS<5:0> | | | | | | 172 |
| MD2CLPPS | — | — | MD2CLPPS<5:0> | | | | | | 172 |
| MD2CHPPS | — | — | MD2CHPPS<5:0> | | | | | | 172 |
| MD2MODPPS | — | — | MD2MODPPS<5:0> | | | | | | 172 |
| MD3CLPPS | — | — | MD3CLPPS<5:0> | | | | | | 172 |
| MD3CHPPS | — | — | MD3CHPPS<5:0> | | | | | | 172 |
| MD3MODPPS | — | — | MD3MODPPS<5:0> | | | | | | 172 |
| PRG1RPPS | — | — | PRG1RPPS<5:0> | | | | | | 172 |
| PRG1FPPS | — | — | PRG1FPPS<5:0> | | | | | | 172 |
| PRG2RPPS | — | — | PRG2RPPS<5:0> | | | | | | 172 |
| PRG2FPPS | — | — | PRG2FPPS<5:0> | | | | | | 172 |
| PRG3RPPS | — | — | PRG3RPPS<5:0> | | | | | | 172 |
| PRG3FPPS | — | — | PRG3FPPS<5:0> | | | | | | 172 |
| CLC1IN0PPS | — | — | CLCIN0PPS<5:0> | | | | | | 172 |
| CLC1IN1PPS | — | — | CLCIN1PPS<5:0> | | | | | | 172 |
| CLC1IN2PPS | — | — | CLCIN2PPS<5:0> | | | | | | 172 |
| CLC1IN3PPS | — | — | CLCIN3PPS<5:0> | | | | | | 172 |
| ADCACTPPS | — | — | ADCACTPPS<5:0> | | | | | | 172 |
| SSPCLKPPS | — | — | SSPCLKPPS<5:0> | | | | | | 172 |
| SSPDATPPS | — | — | SSPDATPPS<5:0> | | | | | | 172 |
| SSPSSPPS | — | — | SSPSSPPS<5:0> | | | | | | 172 |
| RXPPS | — | — | RXPPS<5:0> | | | | | | 172 |
| CKPPS | — | — | CKPPS<5:0> | | | | | | 172 |

**Legend:** — = unimplemented, read as '0'. Shaded cells are unused by the PPS module.

## 13.0 INTERRUPT-ON-CHANGE

All pins on all ports 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 pin, or combination of 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 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 pin, a rising edge detector and a falling edge detector are present. To enable a pin to detect a rising edge, the associated bit of the IOCxP register is set. To enable a pin to detect a falling edge, the associated bit of the IOCxN register is set.

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

### 13.3 Interrupt Flags

The bits located in the IOCxF registers are status flags that correspond to the Interrupt-On-Change pins of each port. If an expected edge is detected on an appropriately enabled pin, then the status flag for that pin will be set, and an interrupt will be generated if the IOCIE bit is set. The IOCIF bit of the INTCON register reflects the status of all IOCxF bits.

### 13.4 Clearing Interrupt Flags

The individual status flags, (IOCxF register 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)**

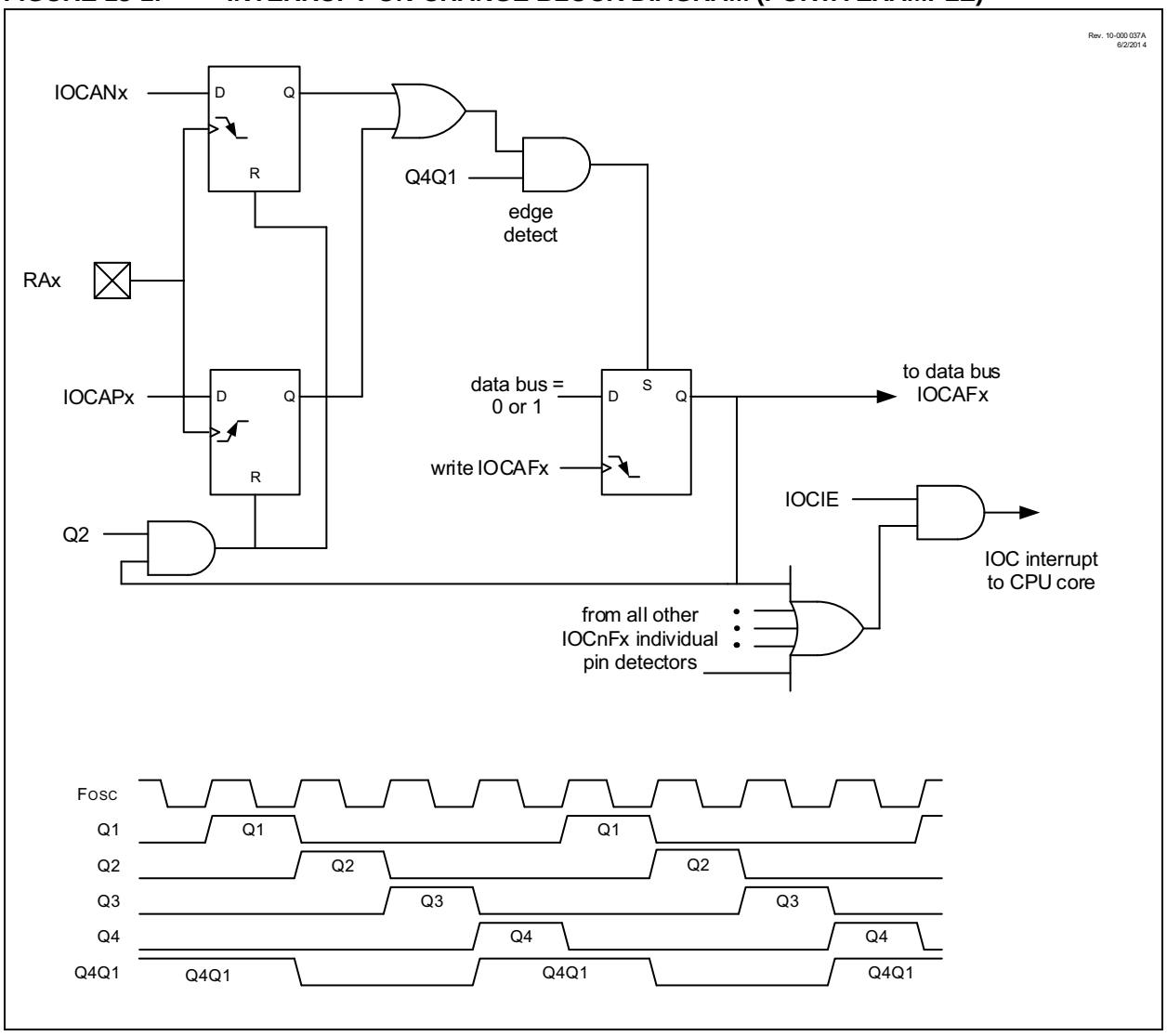
```
MOVLW  0xff
XORWF  IOCAF, W
ANDWF  IOCAF, F
```

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

**FIGURE 13-1: INTERRUPT-ON-CHANGE BLOCK DIAGRAM (PORTA EXAMPLE)**



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

**REGISTER 13-1: IOCAP: INTERRUPT-ON-CHANGE PORTA 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 |
|---|---|---|---|---|---|---|---|
| IOCAP7 | IOCAP6 | IOCAP5 | IOCAP4 | IOCAP3 | IOCAP2 | IOCAP1 | IOCAP0 |
| bit 7 | | | | | | | bit 0 |

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

bit 7-0 **IOCAP<7:0>:** Interrupt-On-Change PORTA Positive Edge Enable bits
1 = Interrupt-on-Change enabled on the pin for a positive going edge. IOCAFx bit and IOCIF flag will be set upon detecting an edge.
0 = Interrupt-on-Change disabled for the associated pin.

**REGISTER 13-2: IOCAN: INTERRUPT-ON-CHANGE PORTA 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 |
|---|---|---|---|---|---|---|---|
| IOCAN7 | IOCAN6 | IOCAN5 | IOCAN4 | IOCAN3 | IOCAN2 | IOCAN1 | IOCAN0 |
| bit 7 | | | | | | | bit 0 |

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

bit 7-0 **IOCAN<7:0>:** Interrupt-On-Change PORTA Negative Edge Enable bits
1 = Interrupt-on-Change enabled on the pin for a negative going edge. IOCAFx bit and IOCIF flag will be set upon detecting an edge.
0 = Interrupt-on-Change disabled for the associated pin.

**REGISTER 13-3: IOCAF: INTERRUPT-ON-CHANGE PORTA 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 |
|---|---|---|---|---|---|---|---|
| IOCAF7 | IOCAF6 | IOCAF5 | IOCAF4 | IOCAF3 | IOCAF2 | IOCAF1 | IOCAF0 |
| bit 7 | | | | | | | bit 0 |

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

bit 7-0 **IOCAF<7:0>:** Interrupt-On-Change PORTA Flag bits
1 = An enabled change was detected on the associated pin.
Set when IOCAPx = 1 and a rising edge was detected on RAx, or when IOCANx = 1 and a falling edge was detected on RAx.
0 = No change was detected, or the user cleared the detected change.

**REGISTER 13-4: IOCBP: INTERRUPT-ON-CHANGE PORTB POSITIVE EDGE REGISTER**

| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | 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 13-5: IOCBN: INTERRUPT-ON-CHANGE PORTB NEGATIVE EDGE REGISTER**

| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | 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 13-6: IOCBF: INTERRUPT-ON-CHANGE PORTB FLAG REGISTER**

| R/W/HS-0/0 | R/W/HS-0/0 | R/W/HS-0/0 | R/W/HS-0/0 | 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.

**REGISTER 13-7: IOCCP: INTERRUPT-ON-CHANGE PORTC 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 |
|---|---|---|---|---|---|---|---|
| IOCCP7 | IOCCP6 | IOCCP5 | IOCCP4 | IOCCP3 | IOCCP2 | IOCCP1 | IOCCP0 |
| 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 **IOCCP<7:0>:** Interrupt-On-Change PORTC Positive Edge Enable bits
1 = Interrupt-on-Change enabled on the pin for a positive going edge. IOCCFx bit and IOCIF flag will be set upon detecting an edge.
0 = Interrupt-on-Change disabled for the associated pin.

**REGISTER 13-8: IOCCN: INTERRUPT-ON-CHANGE PORTC 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 |
|---|---|---|---|---|---|---|---|
| IOCCN7 | IOCCN6 | IOCCN5 | IOCCN4 | IOCCN3 | IOCCN2 | IOCCN1 | IOCCN0 |
| 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 **IOCCN<7:0>:** Interrupt-On-Change PORTC Negative Edge Enable bits
1 = Interrupt-on-Change enabled on the pin for a negative going edge. IOCCFx bit and IOCIF flag will be set upon detecting an edge.
0 = Interrupt-on-Change disabled for the associated pin.

**REGISTER 13-9: IOCCF: INTERRUPT-ON-CHANGE PORTC 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 |
|---|---|---|---|---|---|---|---|
| IOCCF7 | IOCCF6 | IOCCF5 | IOCCF4 | IOCCF3 | IOCCF2 | IOCCF1 | IOCCF0 |
| 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 **IOCCF<7:0>:** Interrupt-On-Change PORTC Flag bits
1 = An enabled change was detected on the associated pin.
Set when IOCCPx = 1 and a rising edge was detected on RCx, or when IOCCNx = 1 and a falling edge was detected on RCx.
0 = No change was detected, or the user cleared the detected change.

**REGISTER 13-10: IOCEP: INTERRUPT-ON-CHANGE PORTE POSITIVE EDGE REGISTER**

| U-0 | U-0 | U-0 | U-0 | R/W-0/0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | IOCEP3 | — | — | — |
| bit 7 | | | | | | | bit 0 |

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

bit 7-4 **Unimplemented:** Read as '0'

bit 3 **IOCEP3:** Interrupt-On-Change PORTE Positive Edge Enable bits
1 = Interrupt-on-Change enabled on the pin for a positive going edge. IOCEFx bit and IOCIF flag will be set upon detecting an edge.
0 = Interrupt-on-Change disabled for the associated pin.

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

**REGISTER 13-11: IOCEN: INTERRUPT-ON-CHANGE PORTE NEGATIVE EDGE REGISTER**

| U-0 | U-0 | U-0 | U-0 | R/W-0/0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | IOCEN3 | — | — | — |
| bit 7 | | | | | | | bit 0 |

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

bit 7-4 **Unimplemented:** Read as '0'

bit 3 **IOCEN3:** Interrupt-On-Change PORTE Negative Edge Enable bits
1 = Interrupt-on-Change enabled on the pin for a negative going edge. IOCEFx bit and IOCIF flag will be set upon detecting an edge.
0 = Interrupt-on-Change disabled for the associated pin.

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

**REGISTER 13-12: IOCEF: INTERRUPT-ON-CHANGE PORTE FLAG REGISTER**

| U-0 | U-0 | U-0 | U-0 | R/W/HS-0/0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | IOCEF3 | — | — | — |
| bit 7 | | | | | | | bit 0 |

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

bit 7-4 **Unimplemented:** Read as '0'

bit 3 **IOCEF3:** Interrupt-On-Change PORTE Flag bits
1 = An enabled change was detected on the associated pin.
Set when IOCEPx = 1 and a rising edge was detected on REx, or when IOCENx = 1 and a falling edge was detected on REx.
0 = No change was detected, or the user cleared the detected change.

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

**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 |
|---|---|---|---|---|---|---|---|---|---|
| ANSELA | — | — | ANSA5 | ANSA4 | ANSA3 | ANSA2 | ANSA1 | ANSA0 | 153 |
| ANSELB | — | — | ANSB5 | ANSB4 | ANSB3 | ANSB2 | ANSB1 | ANSB0 | 158 |
| ANSELC | ANSC7 | ANSC6 | ANSC5 | ANSC4 | ANSC3 | ANSC2 | — | — | 163 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 108 |
| IOCAF | IOCAF7 | IOCAF6 | IOCAF5 | IOCAF4 | IOCAF3 | IOCAF2 | IOCAF1 | IOCAF0 | 179 |
| IOCAN | IOCAN7 | IOCAN6 | IOCAN5 | IOCAN4 | IOCAN3 | IOCAN2 | IOCAN1 | IOCAN0 | 179 |
| IOCAP | IOCAP7 | IOCAP6 | IOCAP5 | IOCAP4 | IOCAP3 | IOCAP2 | IOCAP1 | IOCAP0 | 179 |
| IOCBF | IOCBF7 | IOCBF6 | IOCBF5 | IOCBF4 | IOCBF3 | IOCBF2 | IOCBF1 | IOCBF0 | 180 |
| IOCBN | IOCBN7 | IOCBN6 | IOCBN5 | IOCBN4 | IOCBN3 | IOCBN2 | IOCBN1 | IOCBN0 | 180 |
| IOCBP | IOCBP7 | IOCBP6 | IOCBP5 | IOCBP4 | IOCBP3 | IOCBP2 | IOCBP1 | IOCBP0 | 180 |
| IOCCF | IOCCF7 | IOCCF6 | IOCCF5 | IOCCF4 | IOCCF3 | IOCCF2 | IOCCF1 | IOCCF0 | 181 |
| IOCCN | IOCCN7 | IOCCN6 | IOCCN5 | IOCCN4 | IOCCN3 | IOCCN2 | IOCCN1 | IOCCN0 | 181 |
| IOCCP | IOCCP7 | IOCCP6 | IOCCP5 | IOCCP4 | IOCCP3 | IOCCP2 | IOCCP1 | IOCCP0 | 181 |
| IOCEP | — | — | — | — | IOCEP3 | — | — | — | 182 |
| IOCEN | — | — | — | — | IOCEN3 | — | — | — | 182 |
| IOCEF | — | — | — | — | IOCEF3 | — | — | — | 183 |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 152 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 157 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 162 |
| TRISE | — | — | — | — | TRISE3 | — | — | — | 168 |

**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)

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, and DAC is routed through two independent programmable gain amplifiers. Each amplifier can be programmed for a gain of 1x, 2x or 4x, to produce the three possible voltage levels.

The ADFVR<1:0> bits of the FVRCON register are used to enable and configure the gain amplifier settings for the reference supplied to the ADC module. Reference **Section 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 DAC and comparator module. Reference **Section 17.0 "5-Bit Digital-to-Analog Converter (DAC) Module"** and **Section 19.0 "Comparator 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 Figure 37-77: FVR Stabilization Period, PIC16LF1773/6 Only.

### 14.3 FVR Buffer Stabilization Period

When either FVR Buffer1 or Buffer2 is enabled then the buffer amplifier circuits require 30 μs to stabilize.This stabilization time is required even when the FVR is already operating and stable.

**FIGURE 14-1: VOLTAGE REFERENCE BLOCK DIAGRAM**

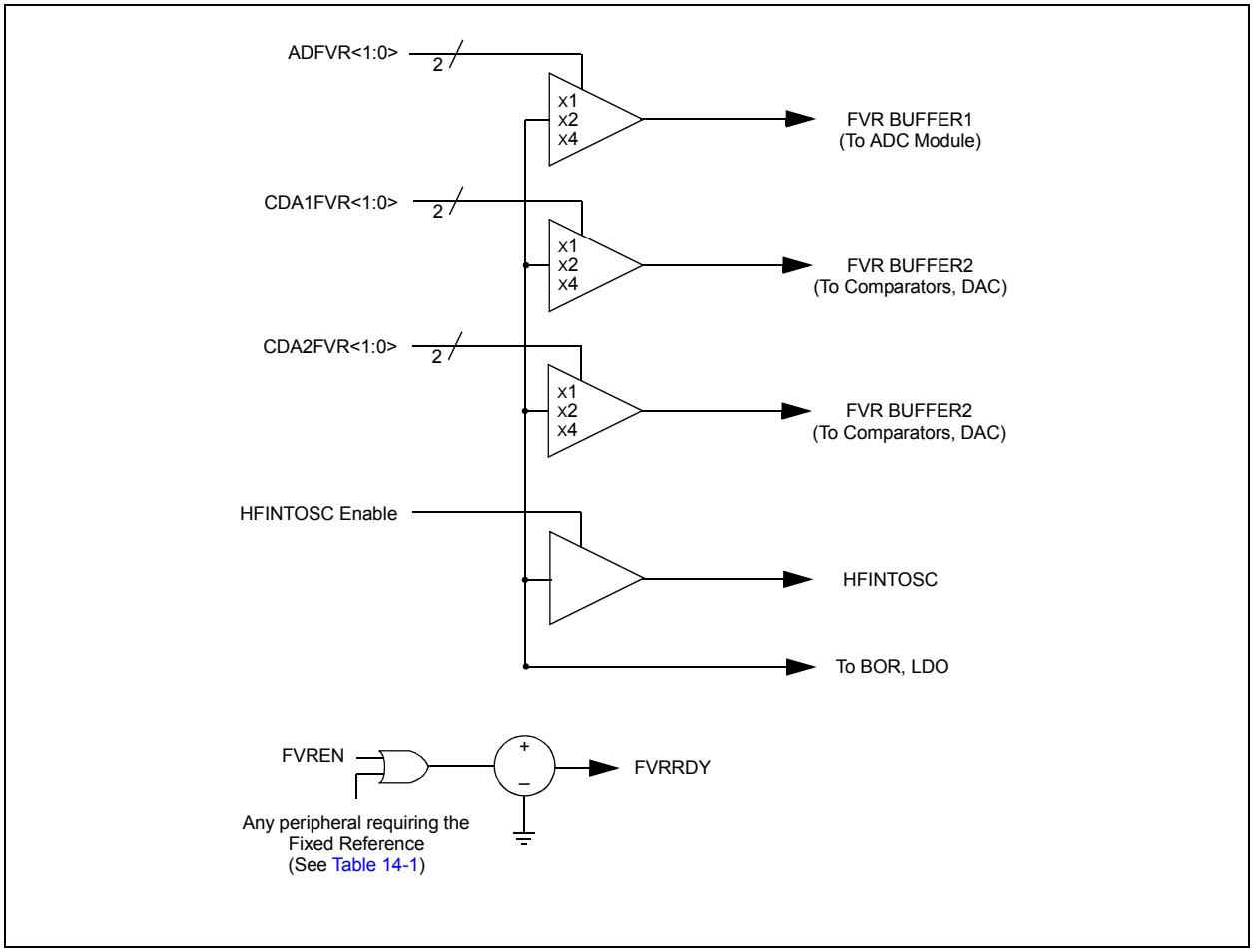



**TABLE 14-1: PERIPHERALS REQUIRING THE FIXED VOLTAGE REFERENCE (FVR)**

| Peripheral | Conditions | Description |
|---|---|---|
| HFINTOSC | FOSC<2:0> = 100 and IRCF<3:0> ≠ 000x | INTOSC is active and device is not in Sleep |
| BOR | BOREN<1:0> = 11 | BOR always enabled |
| | BOREN<1:0> = 10 and BORFS = 1 | BOR disabled in Sleep mode, BOR Fast Start enabled |
| | BOREN<1:0> = 01 and BORFS = 1 | BOR under software control, BOR Fast Start enabled |
| LDO | All PIC16F1773/6 devices, when VREGPM = 1 and not in Sleep | The device runs off of the ULP regulator when in Sleep mode |

## 14.4 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(3) | TSRNG(3) | 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 = $V_{OUT} = V_{DD} - 4V_T$ (High Range)
0 = $V_{OUT} = V_{DD} - 2V_T$ (Low Range)

bit 3-2 **CDAFVR<1:0>:** Comparator/DAC FVR Buffer Gain Selection bits
11 = Comparator/DAC FVR Buffer Gain is 4x, with output $V_{CDAFVR} = 4x\ V_{FVR}$(2)
10 = Comparator/DAC FVR Buffer Gain is 2x, with output $V_{CDAFVR} = 2x\ V_{FVR}$(2)
01 = Comparator/DAC FVR Buffer Gain is 1x, with output $V_{CDAFVR} = 1x\ V_{FVR}$
00 = Comparator/DAC FVR Buffer is off

bit 1-0 **ADFVR<1:0>:** ADC FVR Buffer Gain Selection bits
11 = ADC FVR Buffer Gain is 4x, with output $V_{ADFVR} = 4x\ V_{FVR}$(2)
10 = ADC FVR Buffer Gain is 2x, with output $V_{ADFVR} = 2x\ V_{FVR}$(2)
01 = ADC FVR Buffer Gain is 1x, with output $V_{ADFVR} = 1x\ V_{FVR}$
00 = ADC FVR Buffer is off

**Note 1:** FVRRDY is always '1' on PIC16F1773/6 only.
**2:** Fixed Voltage Reference output cannot exceed $V_{DD}$.
**3:** See **Section 15.0 "Temperature Indicator Module"** for additional information.

**TABLE 14-2: SUMMARY OF REGISTERS ASSOCIATED WITH FIXED VOLTAGE REFERENCE**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on page |
|---|---|---|---|---|---|---|---|---|---|
| FVRCON | FVREN | FVRRDY | TSEN | TSRNG | CDAFVR<1:0> | | ADFVR<1:0> | | 186 |

**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. Reference 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: $V_{OUT} = V_{DD} - 2V_T$

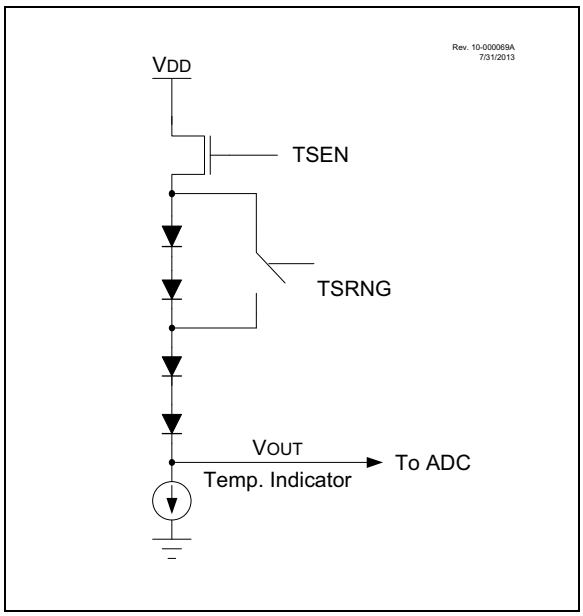
The temperature sense circuit is integrated with the Fixed Voltage Reference (FVR) module. See **Section 14.0 "Fixed Voltage Reference (FVR)"** for more information.

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.

## 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 15-2: SUMMARY OF REGISTERS ASSOCIATED WITH THE TEMPERATURE INDICATOR**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on page |
|---|---|---|---|---|---|---|---|---|---|
| FVRCON | FVREN | FVRRDY | TSEN | TSRNG | CDFVR<1:0> | | ADFVR<1:0> | | 186 |

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

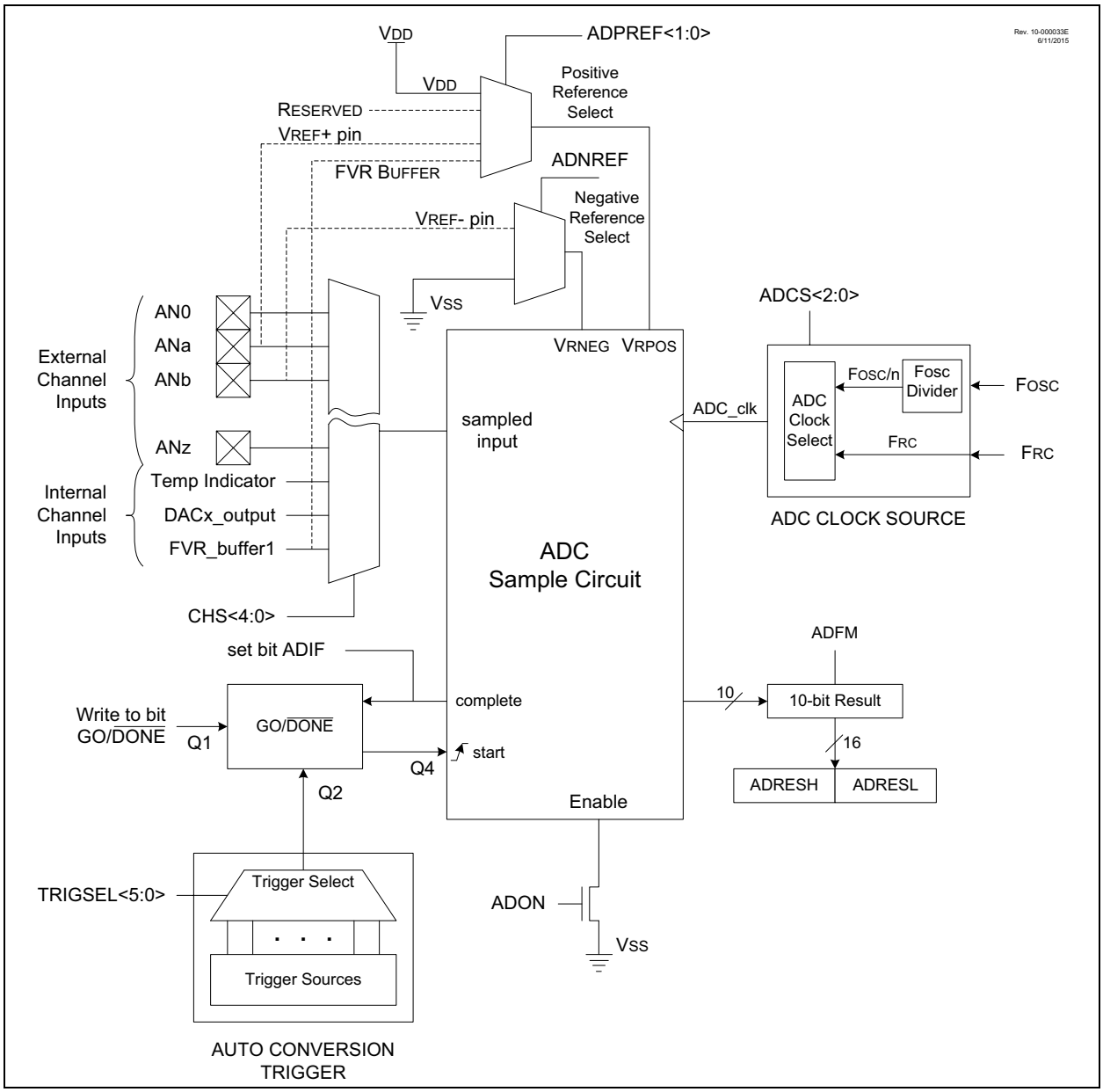
## 16.0 ANALOG-TO-DIGITAL CONVERTER (ADC) MODULE

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.

**FIGURE 16-1: ADC BLOCK DIAGRAM**



## 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 11.0 "I/O Ports"** for more information.

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

### 16.1.2 CHANNEL SELECTION

There are up to 18 channel selections available:

- AN<7:0> pins
- AN<11:8> pins (PIC16(L)F1776 only)
- Temperature Indicator
- DAC1_output and DAC3_output
- DAC2_output and DAC4_output (PIC16(L)F1776 only)
- FVR_buffer1

The CHS bits of the ADCON0 register (Register 16-1) 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 POSITIVE VOLTAGE REFERENCE

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

- VREF+ pin
- VDD
- FVR 2.048V
- FVR 4.096V (Not available on LF devices)
- VSS

See **Section 16.0 "Analog-to-Digital Converter (ADC) Module"** for more details on the Fixed Voltage Reference.

### 16.1.4 ADC NEGATIVE VOLTAGE REFERENCE

The ADNREF bit of the ADCON1 register provides control of the negative voltage reference. The negative voltage reference can be:

- VREF- pin
- VSS

### 16.1.5 CONVERSION CLOCK

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

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

The time to complete one bit conversion is defined as TAD. One full 10-bit conversion requires 11.5 TAD periods as shown in Figure 16-2.

For correct conversion, the appropriate TAD specification must be met. Refer to Table 36-16: ADC Conversion Requirements 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. |

**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 |
| $F_{OSC}$/2 | 000 | 62.5ns(2) | 100 ns(2) | 125 ns(2) | 250 ns(2) | 500 ns(2) | 2.0 µs |
| $F_{OSC}$/4 | 100 | 125 ns(2) | 200 ns(2) | 250 ns(2) | 500 ns(2) | 1.0 µs | 4.0 µs |
| $F_{OSC}$/8 | 001 | 0.5 µs(2) | 400 ns(2) | 0.5 µs(2) | 1.0 µs | 2.0 µs | 8.0 µs(3) |
| $F_{OSC}$/16 | 101 | 800 ns | 800 ns | 1.0 µs | 2.0 µs | 4.0 µs | 16.0 µs(3) |
| $F_{OSC}$/32 | 010 | 1.0 µs | 1.6 µs | 2.0 µs | 4.0 µs | 8.0 µs(3) | 32.0 µs(2) |
| $F_{OSC}$/64 | 110 | 2.0 µs | 3.2 µs | 4.0 µs | 8.0 µs(3) | 16.0 µs(2) | 64.0 µs(2) |
| 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.

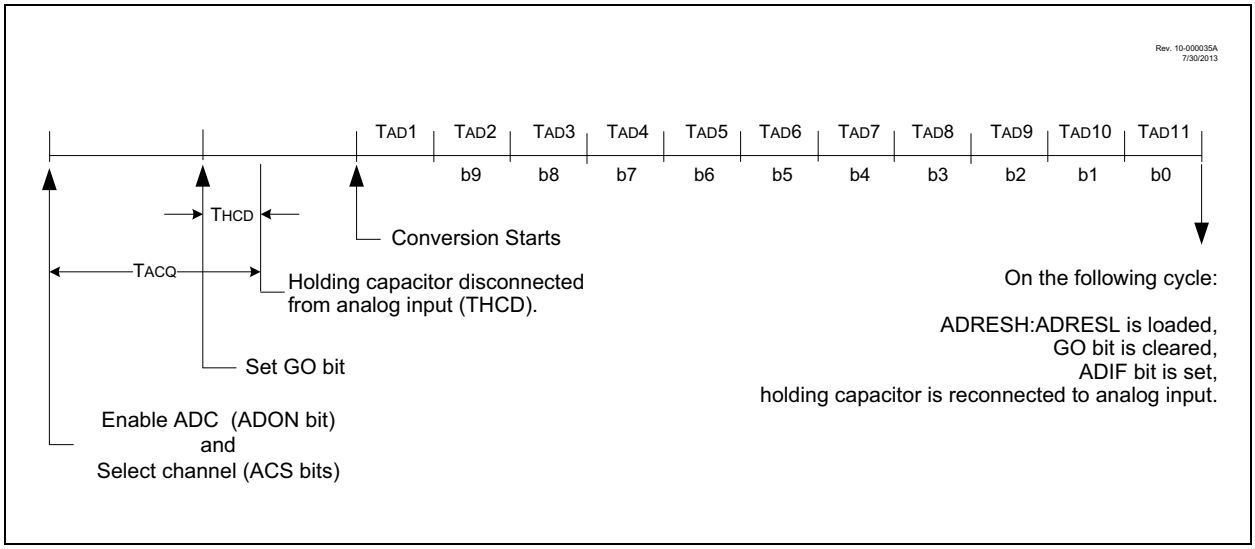
**Note 1:** See $T_{AD}$ parameter for FRC source typical $T_{AD}$ value.

**2:** These values violate the required $T_{AD}$ time.

**3:** Outside the recommended $T_{AD}$ time.

**4:** The ADC clock period ($T_{AD}$) and total ADC conversion time can be minimized when the ADC clock is derived from the system clock $F_{OSC}$. However, the FRC oscillator 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**



### 16.1.6 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. |

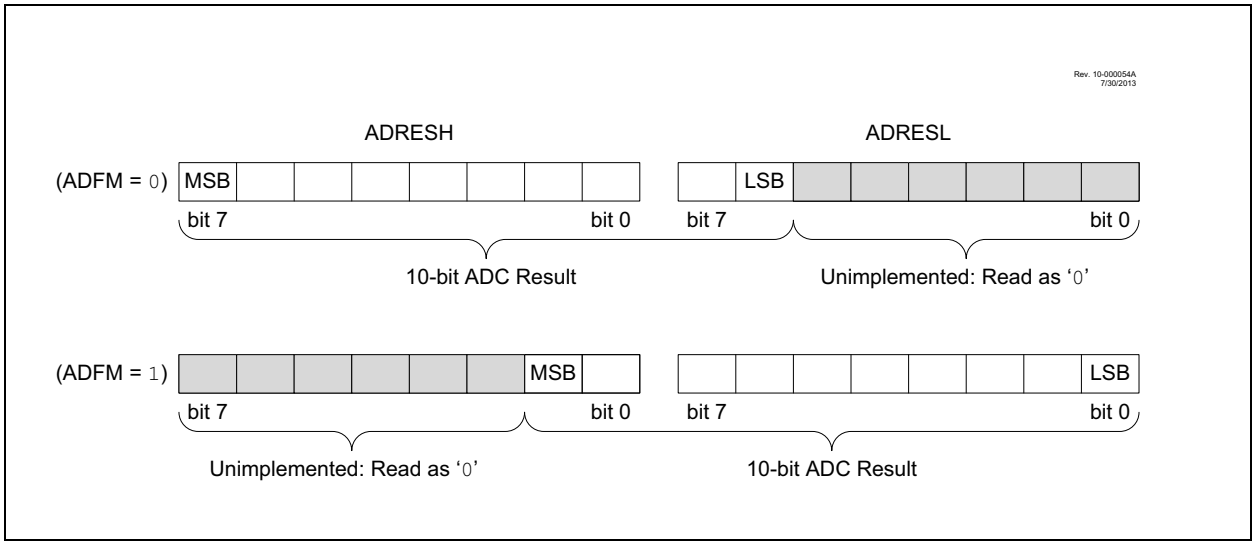
This interrupt can be generated while the device is operating or while in Sleep. If the device is in Sleep, the interrupt will wake-up the device. Upon waking from Sleep, the next instruction following the SLEEP instruction is always executed. If the user is attempting to wake-up from Sleep and resume in-line code execution, the ADIE bit of the PIE1 register and the PEIE bit of the INTCON register must both be set and the GIE bit of the INTCON register must be cleared. If all three of these bits are set, the execution will switch to the Interrupt Service Routine.

### 16.1.7 RESULT FORMATTING

The 10-bit ADC conversion result can be supplied in two formats, left justified or right justified. The ADFM bit of the ADCON1 register controls the output format.

Figure 16-3 shows the two output formats.

**FIGURE 16-3: 10-BIT ADC CONVERSION RESULT FORMAT**



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

| | |
|---|---|
| **Note:** | The GO/DONE bit should not be set in the same instruction that turns on the ADC. Refer to **Section 16.2.6 "ADC Conversion Procedure"**. |

### 16.2.2 COMPLETION OF A CONVERSION

When the conversion is complete, the ADC module will:

- Clear the GO/DONE bit
- Set the ADIF Interrupt Flag bit
- Update the ADRESH and ADRESL registers with new conversion result

### 16.2.3 TERMINATING A CONVERSION

If a conversion must be terminated before completion, the GO/DONE bit can be cleared in software. The ADRESH and ADRESL registers will be updated with the partially complete Analog-to-Digital conversion sample. Incomplete bits will match the last bit converted.

| | |
|---|---|
| **Note:** | A device Reset forces all registers to their Reset state. Thus, the ADC module is turned off and any pending conversion is terminated. |

### 16.2.4 ADC OPERATION DURING SLEEP

The ADC module can operate during Sleep. This requires the ADC clock source to be set to the FRC option. When the FRC oscillator 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.

### 16.2.5 AUTO-CONVERSION TRIGGER

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

The Auto-conversion Trigger source is selected with the TRIGSEL<5:0> bits of the ADCON2 register.

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

See Table 16-2 for auto-conversion sources.

**TABLE 16-2: AUTO-CONVERSION SOURCES**

| Source Peripheral | Signal Name |
|---|---|
| CCP1 | CCP1_trigger |
| CCP2 | CCP2_trigger |
| CCP7 | CCP7_trigger |
| Timer0 | T0_overflow |
| Timer1 | T1_overflow |
| Timer3 | T3_overflow |
| Timer5 | T5_overflow |
| Timer2 | T2_postscaled |
| Timer4 | T4_postscaled |
| Timer6 | T6_postscaled |
| Timer8 | T8_postscaled |
| Comparator C1 | sync_C1OUT |
| Comparator C2 | sync_C2OUT |
| Comparator C3 | sync_C3OUT |
| Comparator C4 | sync_C4OUT |
| Comparator C5 | sync_C5OUT |
| Comparator C6 | sync_C6OUT |
| CLC1 | LC1_out |
| CLC2 | LC2_out |
| CLC3 | LC3_out |
| CLC4 | LC4_out |
| PWM3 | PWM3OUT |
| PWM4 | PWM4OUT |
| PWM9 | PWM9OUT |
| PWM9 | PR/PH/OF/DC9_match |
| PWM5 | PR/PH/OF/DC5_match |
| PWM6 | PR/PH/OF/DC6_match |
| PWM11 | PR/PH/OF/DC11_match |
| ADCACT | ADCACTPPS Pin |

### 16.2.6 ADC CONVERSION PROCEDURE

This is an example procedure for using the ADC to perform an Analog-to-Digital conversion:

1. Configure Port:
   - Disable pin output driver (refer to the TRIS register)
   - Configure pin as analog (refer to the ANSEL register)
   - Disable weak pull-ups either globally (refer to the OPTION_REG register) or individually (refer to the appropriate WPUx register)
2. Configure the ADC module:
   - Select ADC conversion clock
   - Configure voltage reference
   - Select ADC input channel
   - Turn on ADC module
3. Configure ADC interrupt (optional):
   - Clear ADC interrupt flag
   - Enable ADC interrupt
   - Enable peripheral interrupt
   - Enable global interrupt(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 16.4 "ADC Acquisition Requirements"**.

**EXAMPLE 16-1: ADC CONVERSION**

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

## 16.3 Register Definitions: ADC Control

### REGISTER 16-1: ADCON0: ADC 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 |
|---|---|---|---|---|---|---|---|
| CHS<5:0> | | | | | | 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 7-2 **CHS<5:0>:** Analog Channel Select bits

111111 = FVR (Fixed Voltage Reference) Buffer 1 Output(2)
111110 = DAC1_output(1)
111101 = Temperature Indicator(3)
111100 = DAC2_output(1)
111011 = DAC3_output(4)
111010 = DAC4_output(4)
111001 = DAC5_output(4)
111000 = Reserved. No channel connected.
110111 = DAC7_output(4)
110110 = Reserved. No channel connected.
110101 = Reserved. No channel connected.
110010 = Switched AN18(5)
110001 = Reserved. No channel connected.
•
•
•
101011 = Reserved. No channel connected.
101010 = Switched AN10(5)
101001 = Reserved. No channel connected.
•
•
•
100010 = Reserved. No channel connected.
100001 = Switched AN1(5)
011011 = Reserved. No channel connected.
•
•
•
010100 = Reserved. No channel connected.
010011 = AN19
010010 = AN18
010001 = AN17
010000 = AN16
001111 = AN15
001110 = AN14
001101 = AN13
001100 = AN12
001011 = AN11
001010 = AN10
001001 = AN9
001000 = AN8
000111 = Reserved. No channel connected.
000110 = Reserved. No channel connected.
000101 = Reserved. No channel connected.
000100 = AN4
000011 = AN3
000010 = AN2
000001 = AN1
000000 = AN0

## REGISTER 16-1: ADCON0: ADC CONTROL REGISTER 0 (CONTINUED)

bit 1 **GO/$\overline{\text{DONE}}$:** ADC Conversion Status bit
1 = ADC conversion cycle in progress. Setting this bit starts an ADC conversion cycle.
This bit is automatically cleared by hardware when the ADC conversion has completed.
0 = ADC conversion completed/not in progress

bit 0 **ADON:** ADC Enable bit
1 = ADC is enabled
0 = ADC is disabled and consumes no operating current

**Note 1:** See **Section 17.0 "5-Bit 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.
**4:** See **Section 18.0 "10-bit Digital-to-Analog Converter (DAC) Module"** for more information.
**5:** Input source is switched off when op amp override is forcing tri-state. See **Section 29.3 "Override Control"**

**REGISTER 16-2: ADCON1: ADC 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:** ADC Result Format Select bit
1 = Right justified. Six Most Significant bits of ADRESH are set to '0' when the conversion result is loaded.
0 = Left justified. Six Least Significant bits of ADRESL are set to '0' when the conversion result is loaded.

bit 6-4 **ADCS<2:0>:** ADC Conversion Clock Select bits
111 = FRC (clock supplied from an internal RC oscillator)
110 = FOSC/64
101 = FOSC/16
100 = FOSC/4
011 = FRC (clock supplied from an internal RC oscillator)
010 = FOSC/32
001 = FOSC/8
000 = FOSC/2

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

bit 2 **ADNREF:** ADC Negative Voltage Reference Configuration bit
1 = VREF- is connected to external VREF- pin
0 = VREF- is connected to VSS

bit 1-0 **ADPREF<1:0>:** ADC 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 VREF+ pin as the source of the positive reference, be aware that a minimum voltage specification exists. See Table 36-16: ADC Conversion Requirements for details.

**REGISTER 16-3: ADCON2: ADC CONTROL REGISTER 2**

| 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 |
|---|---|---|---|---|---|---|---|
| — | — | TRIGSEL<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 **TRIGSEL<5:0>:** Auto-Conversion Trigger Selection bits(1)

101001 = PWM11 – OF11_match
101000 = PWM11 – PH11_match
100111 = PWM11 – PR11_match
100110 = PWM11 – DC11_match
100101 = PWM6 – OF6_match
100100 = PWM6 – PH6_match
100011 = PWM6 – PR6_match
100010 = PWM6 – DC6_match
100001 = PWM5 – PH5_match
100000 = PWM5 – PH5_match
011111 = PWM5 – PH5_match
011110 = PWM5 – PH5_match
011100 = PWM9 – PWM9OUT
011011 = PWM4 – PWM4OUT
011010 = PWM3 – PWM3OUT
011000 = CCP7 – CCP7_trigger
010111 = CCP2 – CCP2_trigger
010110 = CCP1 – CCP1_trigger
010101 = CLC4 – LC4_out
010100 = CLC3 – LC3_out
010011 = CLC2 – LC2_out
010010 = CLC1 – LC1_out
001111 = Comparator C6 – sync_C6OUT
001110 = Comparator C5 – sync_C5OUT
001101 = Comparator C4 – sync_C4OUT
001100 = Comparator C3 – sync_C3OUT
001011 = Comparator C2 – sync_C2OUT
001010 = Comparator C1 – sync_C1OUT
001001 = Timer8 – T8_postscaled
001000 = Timer6 – T6_postscaled
000111 = Timer5 – T5_overflow
000110 = Timer4 – T4_postscaled
000101 = Timer3 – T3_overflow
000100 = Timer2 – T2_postscaled
000011 = Timer1 – T1_overflow
000010 = Timer0 – T0_overflow
000001 = ADCACT – ADCACTPPS Pin
000000 = No Auto-conversion Trigger selected

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

**REGISTER 16-4: ADRESH: ADC RESULT REGISTER HIGH (ADRESH) ADFM = 0**

| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
|---|---|---|---|---|---|---|---|
| ADRES<9:2> | | | | | | | |
| bit 7 | | | | | | | bit 0 |

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

bit 7-0 **ADRES<9:2>**: ADC Result Register bits
Upper eight bits of 10-bit conversion result

**REGISTER 16-5: ADRESL: ADC RESULT REGISTER LOW (ADRESL) ADFM = 0**

| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
|---|---|---|---|---|---|---|---|
| ADRES<1:0> | | — | — | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

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

bit 7-6 **ADRES<1:0>**: ADC Result Register bits
Lower two bits of 10-bit conversion result

bit 5-0 **Reserved**: Do not use.

**REGISTER 16-6: ADRESH: ADC RESULT REGISTER HIGH (ADRESH) ADFM = 1**

| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | ADRES<9:8> | |
| bit 7 | | | | | | | bit 0 |

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

bit 7-2 **Reserved**: Do not use.

bit 1-0 **ADRES<9:8>**: ADC Result Register bits
Upper two bits of 10-bit conversion result

**REGISTER 16-7: ADRESL: ADC RESULT REGISTER LOW (ADRESL) ADFM = 1**

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

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

bit 7-0 **ADRES<7:0>**: ADC Result Register bits
Lower eight bits of 10-bit conversion result

## 16.4 ADC Acquisition Requirements

For the ADC to meet its specified accuracy, the charge holding capacitor (CHOLD) must be allowed to fully charge to the input channel voltage level. The Analog Input model is shown in Figure 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 ADC 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(1,2,3)**

*Assumptions:* $Temperature = 50°C$ *and external impedance of* $10k\Omega$ *5.0V* $V_{DD}$

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

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

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

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

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

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

*Solving for* $T_C$*:*

$$T_C = -C_{HOLD}(R_{IC} + R_{SS} + R_S)\ ln(1/2047)$$
$$= -10pF(1k\Omega + 7k\Omega + 10k\Omega)\ ln(0.0004885)$$
$$= 1.37\mu s$$

*Therefore:*

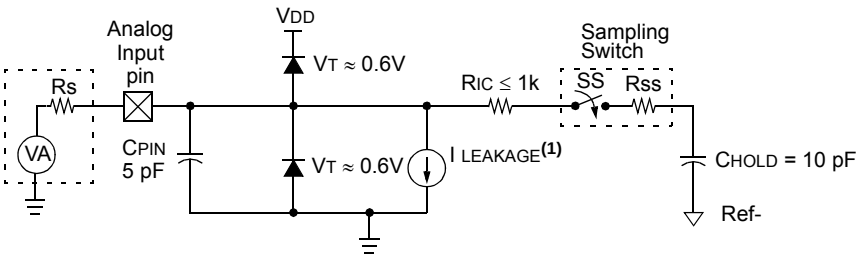
$$T_{ACQ} = 2\mu s + 892ns + [(50°C - 25°C)(0.05\mu s/°C)]$$
$$= 4.62\mu s$$

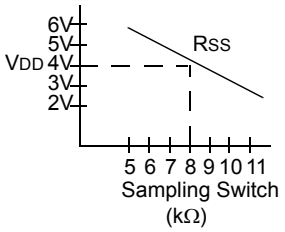
**Note 1:** The reference voltage (VREF) has no effect on the equation, since it cancels itself out.

**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 Table 36-4: I/O Ports (parameter D060).

**FIGURE 16-5: ADC TRANSFER FUNCTION**

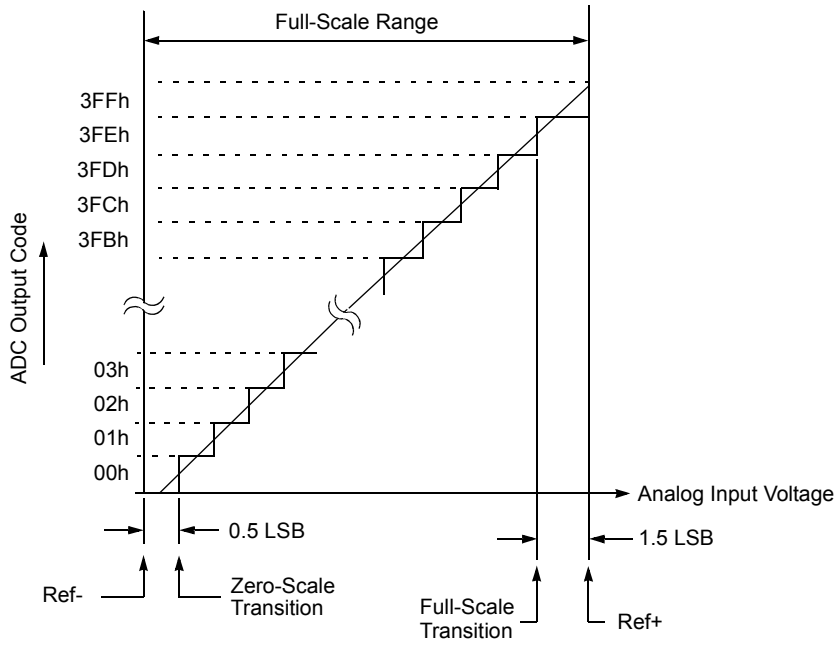


**TABLE 16-3: SUMMARY OF REGISTERS ASSOCIATED WITH ADC**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| ADCON0 | CHS<5:0> | | | | | | GO/$\overline{\text{DONE}}$ | ADON | 195 |
| ADCON1 | ADFM | ADCS<2:0> | | | — | ADNREF | ADPREF<1:0> | | 197 |
| ADCON2 | — | — | TRIGSEL<5:0> | | | | | | 198 |
| ADRESH | ADC Result Register High | | | | | | | | 199, 200 |
| ADRESL | ADC Result Register Low | | | | | | | | 199, 200 |
| ANSELA | — | — | ANSA5 | ANSA4 | ANSA3 | ANSA2 | ANSA1 | ANSA0 | 153 |
| ANSELB | — | — | ANSB5 | ANSB4 | ANSB3 | ANSB2 | ANSB1 | ANSB0 | 158 |
| ANSELC | ANSC7 | ANSC6 | ANSC5 | ANSC4 | ANSC3 | ANSC2 | — | — | 163 |
| FVRCON | FVREN | FVRRDY | TSEN | TSRNG | CDAFVR<1:0> | | ADFVR<1:0> | | 186 |
| DAC1CON0 | EN | FM | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 207 |
| DAC2CON0 | EN | FM | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 207 |
| DAC5CON0 | EN | FM | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 207 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 108 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 109 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 115 |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 152 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB6 | TRISB6 | TRISB1 | TRISB0 | 157 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 162 |

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

## 17.0 5-BIT 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
- Operational amplifier inverting and non-inverting inputs
- ADC input channel
- DACxOUTx pin

**TABLE 17-1: AVAILABLE 5-BIT DACS**

| Device | D3 | D4 | D7 |
|---|---|---|---|
| PIC16(L)F1773 | • | • | • |
| PIC16(L)F1776 | • | • | • |

The Digital-to-Analog Converter (DAC) is enabled by setting the EN bit of the DACxCON0 register.

### 17.1 Output Voltage Selection

The DAC has 32 voltage level ranges. The 32 levels are set with the REF<4:0> bits of the DACxREF register.

The DAC output voltage is determined by Equation 17-1:

**EQUATION 17-1: DAC OUTPUT VOLTAGE**

*IF DACxEN = 1*

$$V_{OUT} = \left( (V_{SOURCE+} - V_{SOURCE-}) \times \frac{DACxR[4:0]}{2^5} \right) + V_{SOURCE-}$$

*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 Table 36-20: 10-bit Digital-to-Analog Converter (DAC) Specifications.

### 17.3 DAC Voltage Reference Output

The DAC voltage can be output to the DACxOUTx pin by setting the OEx bit of the DACxCON0 register. Selecting the DAC voltage for output on the DACxOUTx pin automatically overrides the digital output buffer and digital input threshold detector functions of that pin. Reading the DACxOUTx pin when it has been configured for DAC voltage output will always return a '0'.

Due to the limited current drive capability, a buffer must be used on the DAC voltage output for external connections to the DACxOUTx pin. Figure 17-2 shows an example buffering technique.

**FIGURE 17-1: DIGITAL-TO-ANALOG CONVERTER BLOCK DIAGRAM**

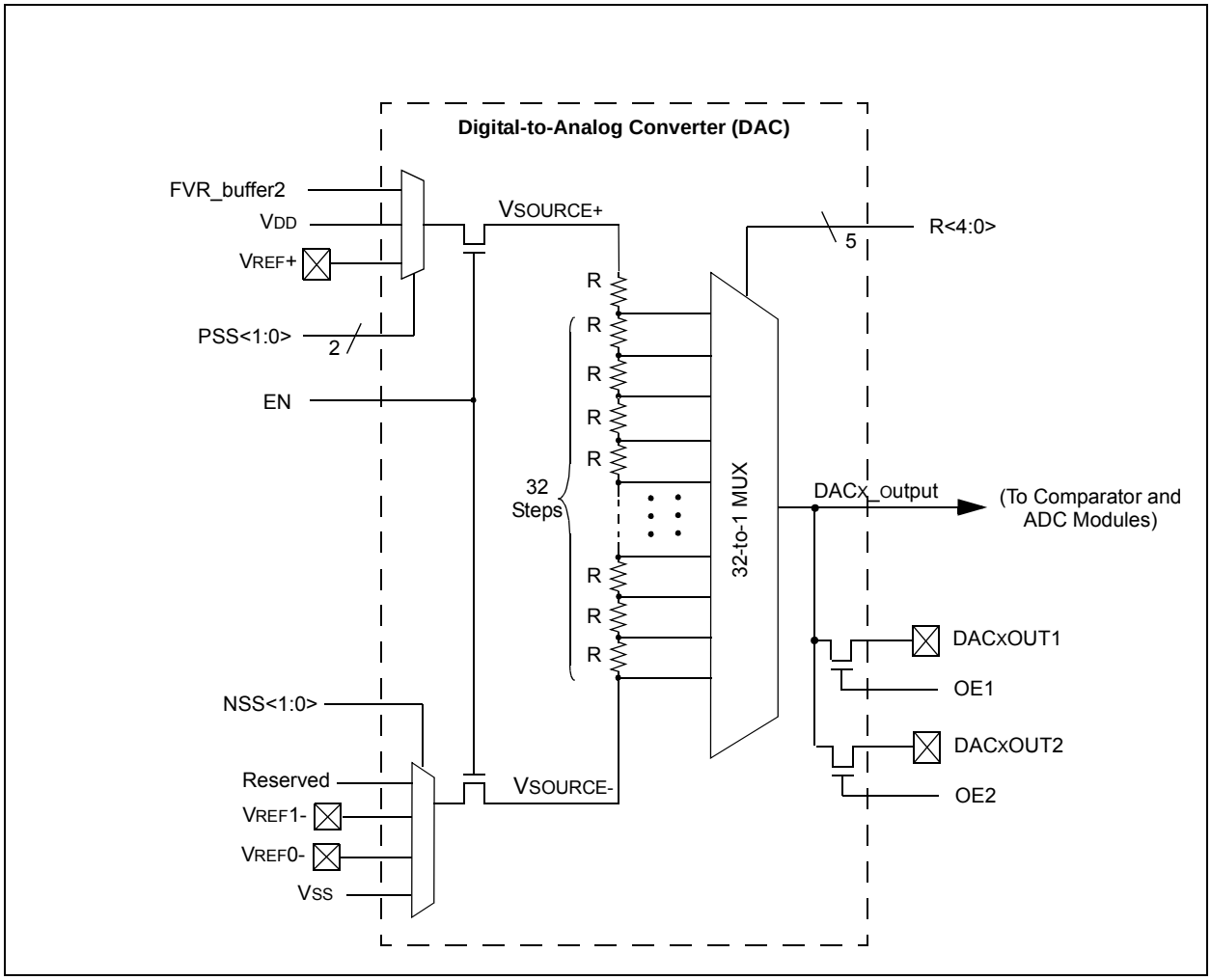
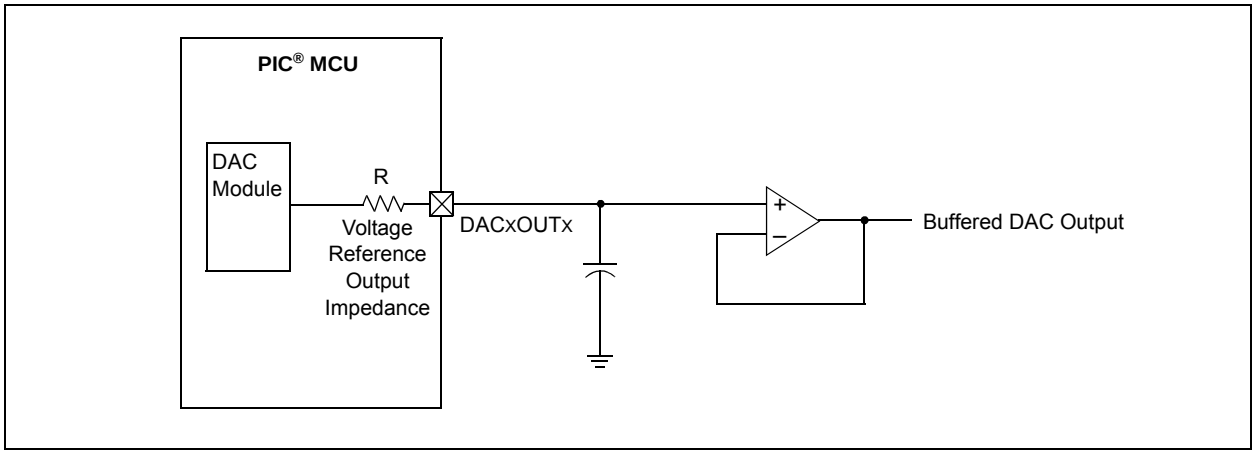



**FIGURE 17-2: VOLTAGE REFERENCE OUTPUT BUFFER EXAMPLE**



## 17.4 Operation During Sleep

The DAC continues to function during Sleep. When the device wakes up from Sleep through an interrupt or a Watchdog Timer time-out, the contents of the DACxCON0 register are not affected. To minimize current consumption in Sleep mode, the voltage reference should be disabled.

## 17.5 Effects of a Reset

A device Reset affects the following:

- DAC is disabled.
- DAC output voltage is removed from the DACxOUTx pin.
- The REF<4:0> voltage reference control bits are cleared.

## 17.6 Register Definitions: DAC Control

Long bit name prefixes for the 5-bit DAC peripherals are shown in Table 17-2. Refer to **Section 1.1 "Register and Bit naming conventions"** for more information

TABLE 17-2:

| Peripheral | Bit Name Prefix |
|---|---|
| DAC3 | DAC3 |
| DAC4 | DAC4 |
| DAC7 | DAC7 |

**REGISTER 17-1: DACxCON0: DACx CONTROL REGISTER 0**

| R/W-0/0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | U-0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| EN | — | OE1 | OE2 | PSS<1:0> | | NSS<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 **EN:** DAC Enable bit
1 = DAC is enabled
0 = DAC is disabled

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

bit 5 **OE1:** DAC Voltage Output Enable bit
1 = DAC voltage level is also an output on the DACxOUT1 pin
0 = DAC voltage level is disconnected from the DACxOUT1 pin

bit 4 **OE2:** DAC Voltage Output Enable bit
1 = DAC voltage level is also an output on the DACxOUT2 pin
0 = DAC voltage level is disconnected from the DACxOUT2 pin

bit 3-2 **PSS<1:0>:** DAC Positive Source Select bits
11 = Reserved, do not use
10 = FVR Buffer2 output
01 = $V_{REF+}$ pin
00 = $V_{DD}$

bit 1-0 **NSS<1:0>:** DAC Negative Source Select bits
11 = Reserved, do not use
10 = DACxREF1- (DAC7) or Reserved (DAC3/4)
01 = DACxREF0-
00 = $A_{GND}$ ($AV_{SS}$)

**REGISTER 17-2: DACxREF: DACx REFERENCE VOLTAGE OUTPUT SELECT REGISTER**

| 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 |
|---|---|---|---|---|---|---|---|
| — | — | — | REF<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 **REF<4:0>:** DACx Reference Voltage Output Select bits (See Equation 17-1)

**TABLE 17-3: SUMMARY OF REGISTERS ASSOCIATED WITH THE DACx MODULE**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on page |
|---|---|---|---|---|---|---|---|---|---|
| DAC3CON0 | EN | — | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 207 |
| DAC4CON0 | EN | --- | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 207 |
| DAC7CON0 | EN | --- | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 207 |
| DAC3REF | --- | --- | --- | REF<4:0> | | | | | 208 |
| DAC4REF | --- | --- | --- | REF<4:0> | | | | | 208 |
| DAC7REF | --- | --- | --- | REF<4:0> | | | | | 208 |

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

## 18.0 10-BIT DIGITAL-TO-ANALOG CONVERTER (DAC) MODULE

The 10-bit Digital-to-Analog Converter (DAC) supplies a variable voltage reference, ratiometric with the input source, with 1024 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
- DACxOUT1 pin
- Op Amp

The Digital-to-Analog Converter is enabled by setting the EN bit of the DACxCON0 register.

**TABLE 18-1: AVAILABLE 10-BIT DACS**

| Device | D1 | D2 | D5 |
|---|---|---|---|
| PIC16(L)F1773 | • | • | • |
| PIC16(L)F1776 | • | • | • |

### 18.1 Output Voltage Level Selection

The DAC has 1024 voltage levels that are set by the 10-bit reference selection word contained in the DACxREFH and DACxREFL registers. This 10-bit word can be left or right justified. See **Section 18.4 "DAC Reference Selection Justification"** for more detail.

The DAC output voltage can be determined with Equation 18-1.

### 18.2 Ratiometric Output Voltage

The DAC output voltage is derived using a resistor ladder with each end of the ladder tied to a positive and negative voltage 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 Table 36-20.

### 18.3 DAC Output

The DAC voltage is always available to the internal peripherals that use it. The DAC voltage can be output to the DACxOUTx pin by setting the OEx bit of the DACxCON0 register. Selecting the DAC voltage for output on the DACxOUTx pin automatically overrides the digital output buffer and digital input threshold detector functions of that pin. Reading the DACxOUTx pin when it has been configured for DAC voltage output will always return a '0'.

Due to the limited current drive capability, a buffer must be used on the DAC voltage output for external connections to either DACxOUTx pin. Figure 18-3 shows a buffering technique example.

### 18.4 DAC Reference Selection Justification

The DAC reference selection can be configured to be left or right justified. When the FM bit of the DACxCON0 register is set, the 10-bit word is left justified such that the eight Most Significant bits fill the DACxREFH register and the two Least Significant bits are left justified in the DACxREFL register. When the FM bit is cleared, the 10-bit word is right justified such that the eight Least Significant bits fill the DACxREFL register and the two Most Significant bits are right justified in the DACxREFH register. Refer to Figure 18-1.

The DACxREFL and DACxREFH registers are double buffered. Writing to either register does not take effect immediately. Writing a '1' to the DACxLD bit of the DACLD register transfers the contents of the DACxREFH and DACxREFL registers to the buffers, thereby changing all 10-bits of the DAC reference selection simultaneously.

**EQUATION 18-1: DAC OUTPUT VOLTAGE**

$$\underline{IF\ EN = 1}$$

$$DACx\_output = \left( (V_{SOURCE}+ - V_{SOURCE}-) \times \frac{DACxR[9:0]}{2^{10}} \right) + V_{SOURCE}-$$

$$V_{SOURCE}+ = V_{DD}, V_{REF}+, \text{ or } FVR\_buffer2$$

$$V_{SOURCE}- = V_{SS} \text{ OR } V_{REF}-$$

**FIGURE 18-1: DAC JUSTIFICATION**

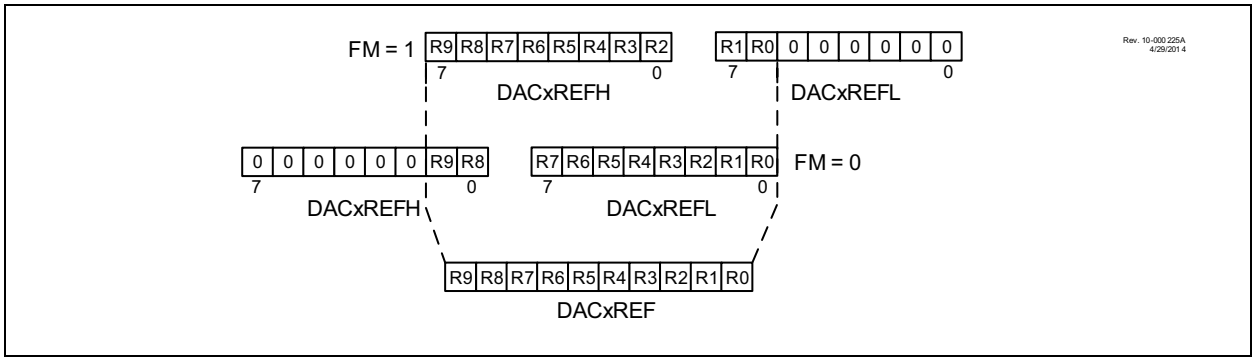
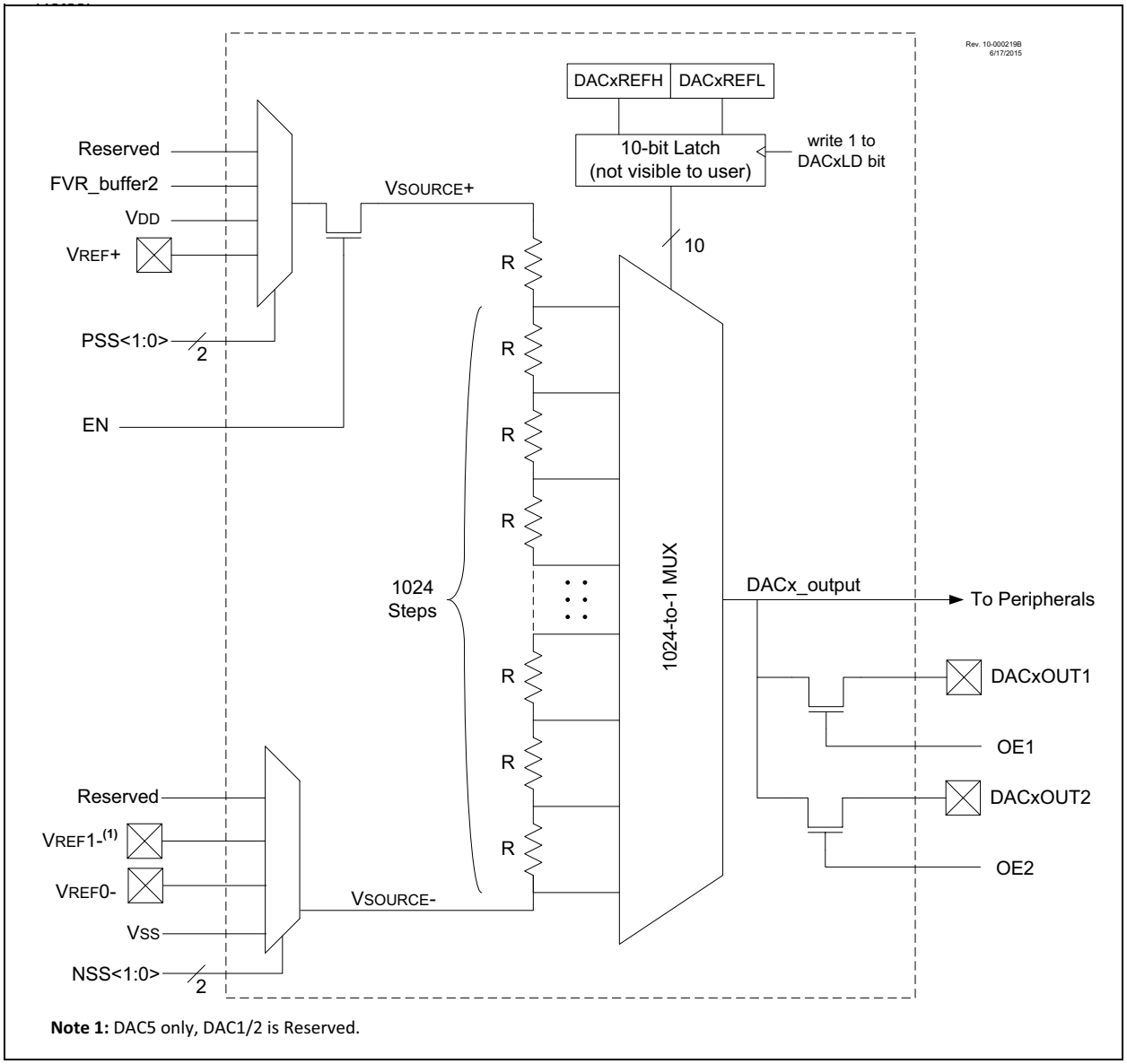
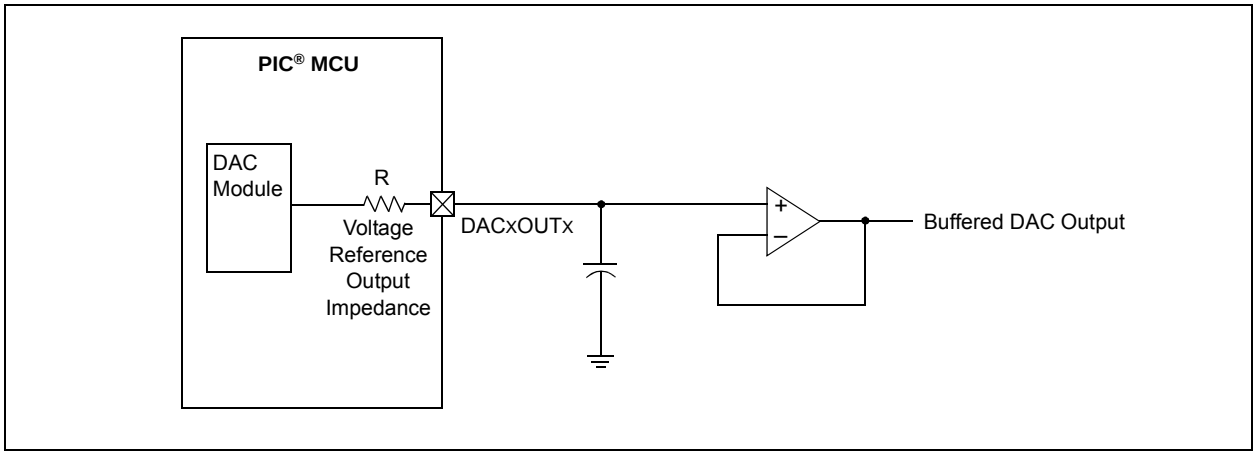



**FIGURE 18-2: DIGITAL-TO-ANALOG CONVERTER BLOCK DIAGRAM**




**Note 1:** DAC5 only, DAC1/2 is Reserved.

**FIGURE 18-3: VOLTAGE REFERENCE OUTPUT BUFFER EXAMPLE**




## 18.5 Operation During Sleep

When the device wakes up from Sleep as the result of an interrupt or a Watchdog Timer time-out, the contents of the DACxCON0 register are not affected. To minimize current consumption in Sleep mode, the voltage reference should be disabled.

## 18.6 Effects of a Reset

A device Reset affects the following:

- DAC is disabled
- DAC output voltage is removed from the DACxOUTx pin
- The REF<9:0> reference selection bits are cleared

## 18.7 Register Definitions: DAC Control

Long bit name prefixes for the 10-bit DAC peripherals are shown in Table 18-2. Refer to **Section 1.1 "Register and Bit naming conventions"** for more information

**TABLE 18-2:**

| Peripheral | Bit Name Prefix |
|---|---|
| DAC1 | DAC1 |
| DAC2 | DAC2 |
| DAC5 | DAC5 |

**REGISTER 18-1: DACxCON0: DAC 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 |
|---|---|---|---|---|---|---|---|
| EN | FM | OE1 | OE2 | PSS<1:0> | | NSS<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 **EN:** DAC Enable bit
- 1 = DACx is enabled
- 0 = DACx is disabled

bit 6 **FM:** DAC Reference Format bit
- 1 = DACx reference selection is left justified
- 0 = DACx reference selection is right justified

bit 5 **OE1:** DAC Voltage Output Enable bit
- 1 = DACx voltage level is also an output on the DACxOUT1 pin
- 0 = DACx voltage level is disconnected from the DACxOUT1 pin

bit 4 **OE2:** DAC Voltage Output Enable bit
- 1 = DACx voltage level is also an output on the DACxOUT2 pin
- 0 = DACx voltage level is disconnected from the DACxOUT2 pin

bit 3-2 **PSS<1:0>:** DAC Positive Source Select bits
- 11 = Reserved. Do not use.
- 10 = FVR_buffer2
- 01 = $V_{REF+}$ pin
- 00 = $V_{DD}$

bit 1-0 **NSS<1:0>:** DAC Negative Source Select bit
- 11 = Reserved. Do not use.
- 10 = DACxREF1- (DAC5) or Reserved (DAC1/2)
- 01 = DACxREF0-
- 00 = $A_{GND}$ ($AV_{SS}$)

**REGISTER 18-2: DACxREFH: DAC REFERENCE VOLTAGE SELECT HIGH 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 |
|---|---|---|---|---|---|---|---|
| REF<9:x> (x Depends on FM bit) | | | | | | | |
| 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 | |

When FM = 1 (left justified)

bit 7-0 **REF<9:2>**: DAC Reference Voltage Output Select bits
DACxOUT1 = f(REF<9:0>) (See Equation 18-1)

When FM = 0 (right justified)

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

bit 1-0 **REF<9:8>**: DAC Reference Voltage Output Select bits
DACxOUT1 = f(REF<9:0>) (See Equation 18-1)

**REGISTER 18-3: DACxREFL: DAC REFERENCE VOLTAGE SELECT LOW 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 |
|---|---|---|---|---|---|---|---|
| REF<x-1:0> (x Depends on FM bit) | | | | | | | |
| 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 | |

When FM = 1 (left justified)

bit 7-6 **REF<1:0>**: DAC Reference Voltage Output Select bits
DACxOUT1 = f(REF<9:0>) (See Equation 18-1)

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

When FM = 0 (right justified)

bit 7-0 **REF<7:0>**: DAC Reference Voltage Output Select bits
DACxOUT1 = f(REF<9:0>) (See Equation 18-1)

**REGISTER 18-4: DACLD: DAC BUFFER LOAD REGISTER**

| U-0 | U-0 | U-0 | R/W-0/0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | DAC5LD | — | — | DAC2LD | DAC1LD |
| 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 configuration bits |

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

bit 4 **DAC5LD:** DAC5 Double Buffer Load bit
- 1 = DAC5REFHL:DAC5REFL values are transfered to the double buffer. Bit is cleared automatically by hardware.
- 0 = DAC5REFHL:DAC5REFL double buffers remain unchanged.

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

bit 1 **DAC2LD:** DAC2 Double Buffer Load bit
- 1 = DAC2REFHL:DAC2REFL values are transfered to the double buffer. Bit is cleared automatically by hardware.
- 0 = DAC2REFHL:DAC2REFL double buffers remain unchanged.

bit 0 **DAC1LD:** DAC1 Double Buffer Load bit
- 1 = DAC1REFHL:DAC1REFL values are transfered to the double buffer. Bit is cleared automatically by hardware.
- 0 = DAC1REFHL:DAC1REFL double buffers remain unchanged.

**TABLE 18-3: SUMMARY OF REGISTERS ASSOCIATED WITH THE DACx MODULE**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| DAC1CON0 | EN | FM | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 212 |
| DAC2CON0 | EN | FM | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 212 |
| DAC5CON0 | EN | FM | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 212 |
| DAC1REFH | REF<9:x> (x Depends on FM bit) | | | | | | | | 213 |
| DAC2REFH | REF<9:x> (x Depends on FM bit) | | | | | | | | 213 |
| DAC5REFH | REF<9:x> (x Depends on FM bit) | | | | | | | | 213 |
| DAC1REFL | REF<x-1:0> (x Depends on FM bit) | | | | | | | | 213 |
| DAC2REFL | REF<x-1:0> (x Depends on FM bit) | | | | | | | | 213 |
| DAC5REFL | REF<x-1:0> (x Depends on FM bit) | | | | | | | | 213 |
| DACLD | — | — | — | DAC5LD | — | — | DAC2LD | DAC1LD | 214 |

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

## 19.0 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

### 19.1 Comparator Overview

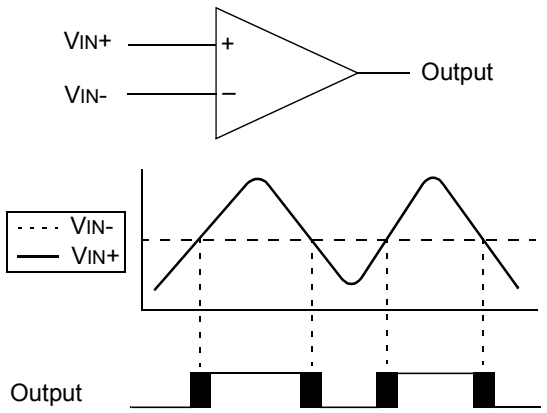
A single comparator is shown in Figure 19-1 along with the relationship between the analog input levels and the digital output. When the analog voltage at VIN+ is less than the analog voltage at VIN-, the output of the comparator is a digital low level. When the analog voltage at VIN+ is greater than 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 19-1.

**TABLE 19-1: AVAILABLE COMPARATORS**

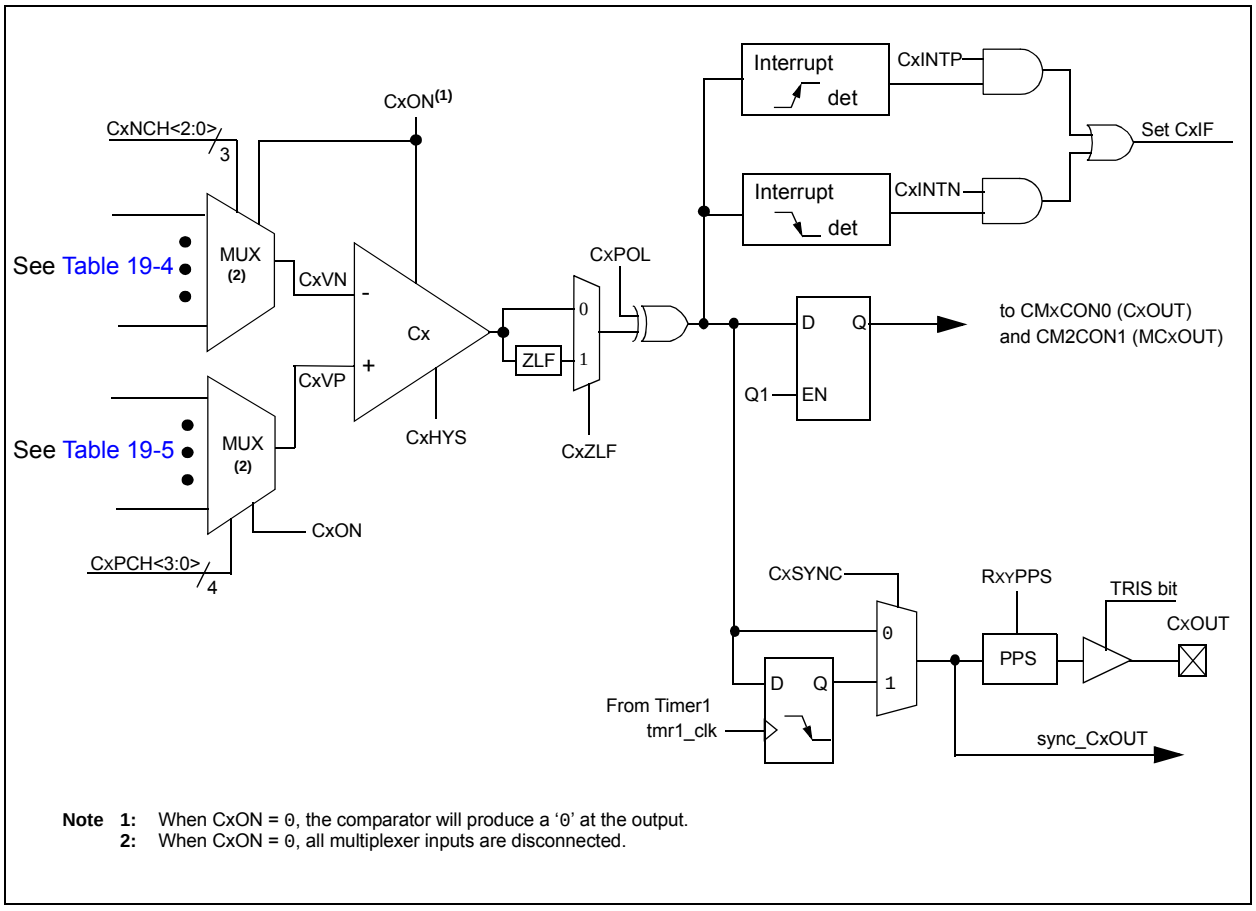
| Device | C1 | C2 | C3 | C4 | C5 | C6 |
|---|---|---|---|---|---|---|
| PIC16(L)F1773 | • | • | • | • | • | • |
| PIC16(L)F1776 | • | • | • | • | • | • |

**FIGURE 19-1: SINGLE COMPARATOR**




**Note:** The black areas of the output of the comparator represents the uncertainty due to input offsets and response time.

**FIGURE 19-2: 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.

## 19.2 Comparator Control

Each comparator has two control registers: CMxCON0 and CMxCON1.

The CMxCON0 register (see Register 19-1) contains Control and Status bits for the following:

- Enable
- Output
- Output polarity
- Zero latency filter
- Speed/Power selection
- Hysteresis enable
- Output synchronization

The CMxCON1 register (see Register 19-2) contains Control bits for the following:

- Interrupt enable
- Interrupt edge polarity
- Positive input channel selection
- Negative input channel selection

### 19.2.1 COMPARATOR ENABLE

Setting the ON bit of the CMxCON0 register enables the comparator for operation. Clearing the ON bit disables the comparator resulting in minimum current consumption.

### 19.2.2 COMPARATOR OUTPUT SELECTION

The output of the comparator can be monitored by reading either the OUT 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:

- Desired pin PPS control
- Corresponding TRIS bit must be cleared
- ON bit of the CMxCON0 register must be set

| | |
|---|---|
| **Note 1:** | The internal output of the comparator is latched with each instruction cycle. Unless otherwise specified, external outputs are not latched. |

### 19.2.3 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 POL bit of the CMxCON0 register. Clearing the POL bit results in a non-inverted output.

Table 19-2 shows the output state versus input conditions, including polarity control.

**TABLE 19-2: 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 |

## 19.3 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 HYS bit of the CMxCON0 register.

See Comparator Specifications in Table 36-19: Comparator Specifications for more information.

## 19.4 Timer1 Gate Operation

The output resulting from a comparator operation can be used as a source for gate control of Timer1. See **Section 22.6 "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.

### 19.4.1 COMPARATOR OUTPUT SYNCHRONIZATION

The output from a comparator can be synchronized with Timer1 by setting the SYNC 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 19-2) and the Timer1 Block Diagram (Figure 22-1) for more information.

## 19.5 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 (INTP and/or INTN 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:

- ON and POL bits of the CMxCON0 register
- CxIE bit of the PIE2 register
- INTP bit of the CMxCON1 register (for a rising edge detection)
- INTN 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 POL bit of the CMxCON0 register, or by switching the comparator on or off with the ON bit of the CMxCON0 register. |
|---|---|

## 19.6 Comparator Positive Input Selection

Configuring the PCH<2:0> 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
- Programmable ramp generator output
- 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 "5-Bit 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.

## 19.7 Comparator Negative Input Selection

The NCH<2:0> bits of the CMxCON0 register direct an analog input pin and internal reference voltage or analog ground to the inverting input of the comparator:

- CxIN- pin
- FVR (Fixed Voltage Reference)
- Analog Ground

Some inverting input selections share a pin with the operational amplifier output function. Enabling both functions at the same time will direct the operational amplifier output to the comparator inverting input.

| **Note:** | To use CxINy+ and CxINy- 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. |
|---|---|

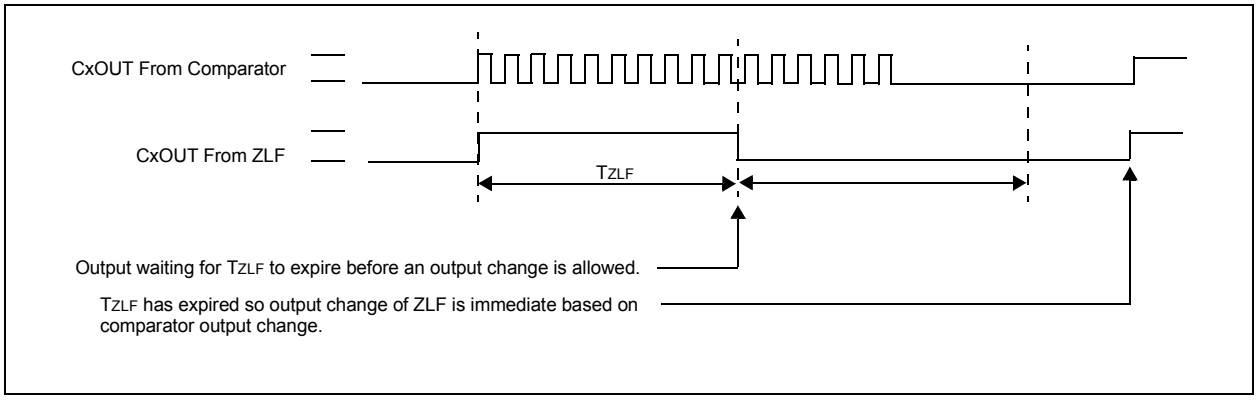
## 19.8 Comparator Response Time

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 Table 36-19: Comparator Specifications for more details.

## 19.9 Zero Latency Filter

In high-speed operation, and under proper circuit conditions, it is possible for the comparator output to oscillate. This oscillation can have adverse effects on the hardware and software relying on this signal. Therefore, a digital filter has been added to the comparator output to suppress the comparator output oscillation. Once the comparator output changes, the output is prevented from reversing the change for a nominal time of 20 ns. This allows the comparator output to stabilize without affecting other dependent devices. Refer to Figure 19-3.

**FIGURE 19-3: COMPARATOR ZERO LATENCY FILTER OPERATION**



## 19.10 Analog Input Connection Considerations

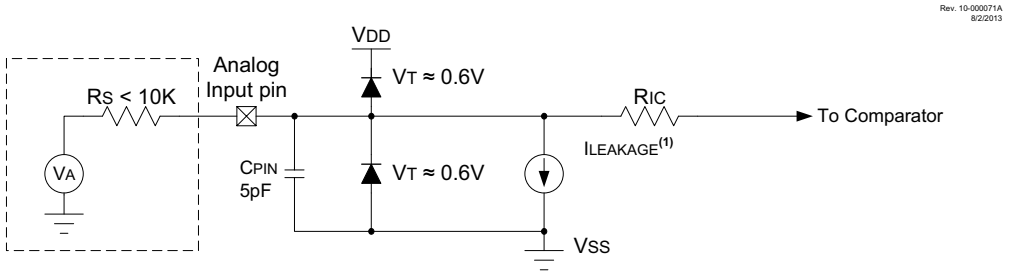
A simplified circuit for an analog input is shown in Figure 19-4. Since the analog input pins share their connection with a digital input, they have reverse biased ESD protection diodes to VDD and VSS. The analog input, therefore, must be between VSS and VDD. If the input voltage deviates from this range 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.

**FIGURE 19-4: 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 I/O Ports in Table 36-4: I/O Ports.

## 19.11 Register Definitions: Comparator Control

Long bit name prefixes for the DSM peripherals are shown in Table 19-3. Refer to **Section 1.1.2.2 "Long Bit Names"** for more information

**TABLE 19-3:**

| Peripheral | Bit Name Prefix |
|---|---|
| Comparator 1 | C1 |
| Comparator 2 | C2 |
| Comparator 3 | C3 |
| Comparator 4 | C4 |
| Comparator 5 | C5 |
| Comparator 6 | C6 |

**REGISTER 19-1: CMxCON0: COMPARATOR Cx CONTROL REGISTER 0**

| R/W-0/0 | R-0/0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-1/1 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| ON | OUT | — | POL | ZLF | CxSP | HYS | SYNC |
| 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 **ON:** Comparator Enable bit
1 = Comparator is enabled
0 = Comparator is disabled and consumes no active power

bit 6 **OUT:** Comparator Output bit
If POL = 1 (inverted polarity):
1 = CxVP < CxVN
0 = CxVP > CxVN
If POL = 0 (non-inverted polarity):
1 = CxVP > CxVN
0 = CxVP < CxVN

bit 5 **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 4 **POL:** Comparator Output Polarity Select bit
1 = Comparator output is inverted
0 = Comparator output is not inverted

bit 3 **ZLF:** Comparator Zero Latency Filter Enable bit
1 = Comparator output is filtered
0 = Comparator output is unfiltered

bit 2 **Reserved:** Read as '1'. Maintain this bit set.

bit 1 **HYS:** Comparator Hysteresis Enable bit
1 = Comparator hysteresis enabled
0 = Comparator hysteresis disabled

bit 0 **SYNC:** 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

**REGISTER 19-2: CMxCON1: COMPARATOR Cx CONTROL REGISTER 1**

| U-0 | U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | — | INTP | INTN |
| 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 **INTP:** 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 0 **INTN:** 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

**REGISTER 19-3: CMxNSEL: COMPARATOR Cx NEGATIVE CHANNEL SELECT REGISTER**

| U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | NCH<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 | |

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

bit 3-0 **NCH<3:0>:** Comparator Negative Input Channel Select bits
CxVN connects to input source indicated by Table 19-4: Negative Input Sources

**TABLE 19-4: NEGATIVE INPUT SOURCES**

| NCH<3:0> | C1, C2, C3, C4 | C5, C6 |
|---|---|---|
| 1111 | Reserved. Do not use | Reserved. Do not use |
| 1110 | Reserved. Do not use | Reserved. Do not use |
| 1101 | Reserved. Do not use | Reserved. Do not use |
| 1100 | Reserved. Do not use | Reserved. Do not use |
| 1011 | Reserved. Do not use | Reserved. Do not use |
| 1010 | OPA2IN- pin | Reserved. Do not use |
| 1001 | OPA1IN- pin | OPA3IN- pin |
| 1000 | AGND | AGND |
| 0111 | PRG2_out | Reserved. Do not use |
| 0110 | PRG1_out | PRG3_out |
| 0101 | FVR_Buffer2 | FVR_Buffer2 |
| 0100 | CxIN4- pin | CxIN4- pin |
| 0011 | CxIN3- (OPA2OUT) pin | CxIN3- pin |
| 0010 | CxIN2- pin | CxIN2- pin |
| 0001 | CxIN1- (OPA1OUT) pin | CxIN1- (OPA3OUT) pin |
| 0000 | CxIN0- pin | CxIN0- pin |

**REGISTER 19-4: CMxPSEL: COMPARATOR Cx POSITIVE CHANNEL SELECT REGISTER 1**

| U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | PCH<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 | |

bit 7-4 **Unimplemented:** Read as '0'

bit 3-0 **PCH<3:0>:** Comparator Positive Input Channel Select bits
CxVP connects to input source indicated by Table 19-5: Positive Input Sources

**TABLE 19-5: POSITIVE INPUT SOURCES**

| PCH<3:0> | C1, C2, C3, C4 | C5, C6 |
|---|---|---|
| 1111 | Reserved. Do not use | Reserved. Do not use |
| 1110 | Reserved. Do not use | Reserved. Do not use |
| 1101 | Reserved. Do not use | Reserved. Do not use |
| 1100 | Reserved. Do not use | Reserved. Do not use |
| 1011 | Reserved. Do not use | Reserved. Do not use |
| 1010 | Reserved. Do not use | Reserved. Do not use |
| 1001 | AGND | AGND |
| 1000 | DAC4_out | Reserved. Do not use |
| 0111 | DAC3_out | DAC7_out |
| 0110 | DAC2_out | Reserved. Do not use |
| 0101 | DAC1_out | DAC5_out |
| 0100 | PRG2_out | Reserved. Do not use |
| 0011 | PRG1_out | PRG3_out |
| 0010 | FVR_Buffer2 | FVR_Buffer2 |
| 0001 | CxIN1+ pin | CxIN1+ pin |
| 0000 | CxIN0+ pin | CxIN0+ pin |

**Note:** There are no long and short bit name variants for the following mirror register

**REGISTER 19-5: CMOUT: COMPARATOR OUTPUT REGISTER**

| U-0 | U-0 | R-0/0 | R-0/0 | R-0/0 | R-0/0 | R-0/0 | R-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | MC6OUT | MC5OUT | MC4OUT | 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 7-6 **Unimplemented:** Read as ‘0’

bit 5 **MC6OUT:** Mirror Copy of C6OUT bit

bit 4 **MC5OUT:** Mirror Copy of C5OUT bit

bit 3 **MC4OUT:** Mirror Copy of C4OUT bit

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 19-6: 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 |
|---|---|---|---|---|---|---|---|---|---|
| ANSELA | — | — | ANSA5 | ANSA4 | ANSA3 | ANSA2 | ANSA1 | ANSA0 | 153 |
| ANSELB | — | — | ANSB5 | ANSB4 | ANSB3 | ANSB2 | ANSB1 | ANSB0 | 158 |
| ANSELC | ANSC7 | ANSC6 | ANSC5 | ANSC4 | ANSC3 | ANSC2 | — | — | 163 |
| CM1CON0 | ON | OUT | — | POL | ZLF | Reserved | HYS | SYNC | 221 |
| CM2CON0 | ON | OUT | — | POL | ZLF | Reserved | HYS | SYNC | 221 |
| CM3CON0 | ON | OUT | — | POL | ZLF | Reserved | HYS | SYNC | 221 |
| CM4CON0 | ON | OUT | — | POL | ZLF | Reserved | HYS | SYNC | 221 |
| CM5CON0 | ON | OUT | — | POL | ZLF | Reserved | HYS | SYNC | 221 |
| CM6CON0 | ON | OUT | — | POL | ZLF | Reserved | HYS | SYNC | 221 |
| CM1CON1 | — | — | — | — | — | — | INTP | INTN | 222 |
| CM2CON1 | — | — | — | — | — | — | INTP | INTN | 222 |
| CM3CON1 | — | — | — | — | — | — | INTP | INTN | 222 |
| CM4CON1 | — | — | — | — | — | — | INTP | INTN | 222 |
| CM5CON1 | — | — | — | — | — | — | INTP | INTN | 222 |
| CM6CON1 | — | — | — | — | — | — | INTP | INTN | 222 |
| CM1NSEL | — | — | — | — | NCH<3:0> | | | | 223 |
| CM2NSEL | — | — | — | — | NCH<3:0> | | | | 223 |
| CM3NSEL | — | — | — | — | NCH<3:0> | | | | 223 |
| CM4NSEL | — | — | — | — | NCH<3:0> | | | | 223 |
| CM5NSEL | — | — | — | — | NCH<3:0> | | | | 223 |
| CM6NSEL | — | — | — | — | NCH<3:0> | | | | 223 |
| CM1PSEL | — | — | — | — | PCH<3:0> | | | | 224 |
| CM2PSEL | — | — | — | — | PCH<3:0> | | | | 224 |
| CM3PSEL | — | — | — | — | PCH<3:0> | | | | 224 |
| CM4PSEL | — | — | — | — | PCH<3:0> | | | | 224 |
| CM5PSEL | — | — | — | — | PCH<3:0> | | | | 224 |
| CM6PSEL | — | — | — | — | PCH<3:0> | | | | 224 |
| CMOUT | — | — | MC6OUT | MC5OUT | MC4OUT | MC3OUT | MC2OUT | MC1OUT | 225 |
| FVRCON | FVREN | FVRRDY | TSEN | TSRNG | CDAFVR<1:0> | | ADFVR<1:0> | | 186 |
| DAC1CON0 | EN | FM | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 212 |
| DAC2CON0 | EN | FM | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 212 |
| DAC5CON0 | EN | FM | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 212 |
| DAC3CON0 | EN | — | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 207 |
| DAC4CON0 | EN | — | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 207 |
| DAC7CON0 | EN | — | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 207 |
| DAC3REF | --- | --- | --- | REF<4:0> | | | | | 208 |
| DAC4REF | --- | --- | --- | REF<4:0> | | | | | 208 |
| DAC7REF | --- | --- | --- | REF<4:0> | | | | | 208 |
| DAC1REFH | REF<9:x> (x Depends on FM bit) | | | | | | | | 213 |
| DAC2REFH | REF<9:x> (x Depends on FM bit) | | | | | | | | 213 |
| DAC5REFH | REF<9:x> (x Depends on FM bit) | | | | | | | | 213 |

**Legend:** — = unimplemented location, read as '0'. Shaded cells are unused by the comparator module.

**TABLE 19-4: SUMMARY OF REGISTERS ASSOCIATED WITH COMPARATOR MODULE (CONT.)**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| DAC1REFL | REF<x-1:0> (x Depends on FM bit) | | | | | | | | 213 |
| DAC2REFL | REF<x-1:0> (x Depends on FM bit) | | | | | | | | 213 |
| DAC5REFL | REF<x-1:0> (x Depends on FM bit) | | | | | | | | 213 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 108 |
| PIE2 | OSFIE | C2IE | C1IE | COG1IE | BCL1IE | C4IE | C3IE | CCP2IE | 110 |
| PIE5 | — | CCP7IE | — | COG3IE | — | — | C6IE | C5IE | 113 |
| PIR2 | OSFIF | C2IF | C1IF | COG1IF | BCL1IF | C4IF | C3IF | CCP2IF | 116 |
| PIR5 | — | CCP7IF | — | COG3IF | — | — | C6IF | C5IF | 119 |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 152 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 157 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 162 |

**Legend:** — = unimplemented location, read as '0'. Shaded cells are unused by the comparator module.

## 20.0 ZERO-CROSS DETECTION (ZCD) MODULE

The ZCD module detects when an A/C signal crosses through the ground potential. The actual zero-crossing threshold is the zero-crossing reference voltage, ZCPINV, which is typically 0.75V above ground.

The connection to the signal to be detected is through a series current limiting resistor. The module applies a current source or sink to the ZCD pin to maintain a constant voltage on the pin, thereby preventing the pin voltage from forward biasing the ESD protection diodes. When the applied voltage is greater than the reference voltage, the module sinks current. When the applied voltage is less than the reference voltage, the module sources current. The current source and sink action keeps the pin voltage constant over the full range of the applied voltage. The ZCD module is shown in the simplified block diagram Figure 20-2.

The ZCD module is useful when monitoring an AC waveform for, but not limited to, the following purposes:

- A/C period measurement
- Accurate long term time measurement
- Dimmer phase delayed drive
- Low EMI cycle switching

## 20.1 External Resistor Selection

The ZCD module requires a current limiting resistor in series with the external voltage source. The impedance and rating of this resistor depends on the external source peak voltage. Select a resistor value that will drop all of the peak voltage when the current through the resistor is nominally 300 µA. Refer to Equation 20-1 and Figure 20-1. Make sure that the ZCD I/O pin internal weak pull-up is disabled so it does not interfere with the current source and sink.

**EQUATION 20-1: EXTERNAL RESISTOR**

$$R_{series} = \frac{V_{peak}}{3\times10^{-4}}$$

**FIGURE 20-1: EXTERNAL VOLTAGE**

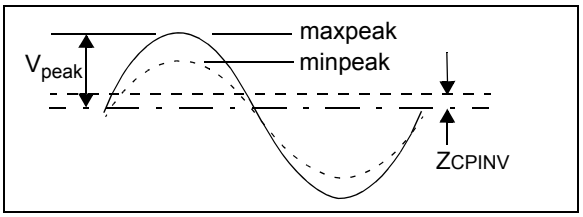
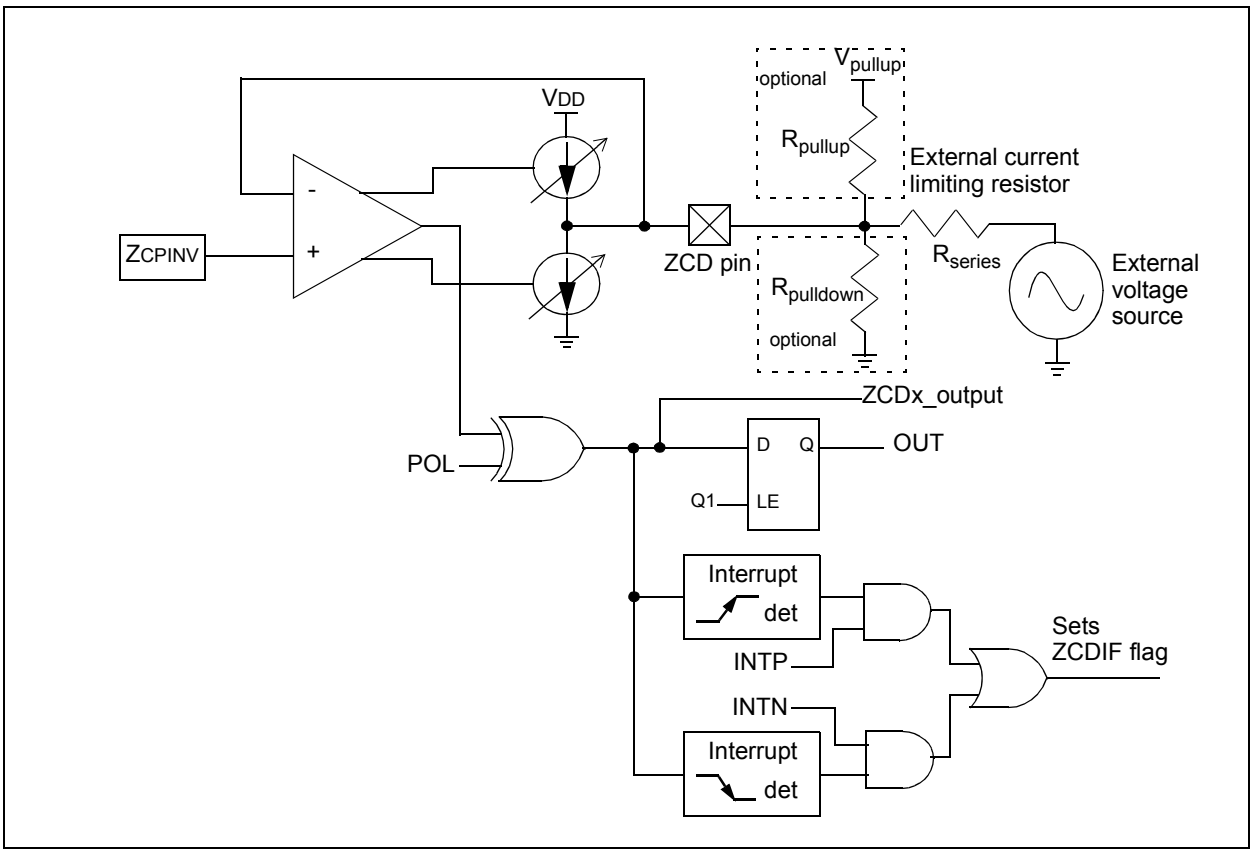



**FIGURE 20-2: SIMPLIFIED ZCD BLOCK DIAGRAM**



## 20.2 ZCD Logic Output

The ZCD module includes a Status bit, which can be read to determine whether the current source or sink is active. The OUT bit of the ZCDCON register is set when the current sink is active, and cleared when the current source is active. The OUT bit is affected by the polarity bit.

## 20.3 ZCD Logic Polarity

The POL bit of the ZCDxCON register inverts the OUT bit relative to the current source and sink output. When the POL bit is set, a OUT high indicates that the current source is active, and a low output indicates that the current sink is active.

The POL bit affects the ZCD interrupts. See **Section 20.4 "ZCD Interrupts"**.

## 20.4 ZCD Interrupts

An interrupt will be generated upon a change in the ZCD logic output when the appropriate interrupt enables are set. A rising edge detector and a falling edge detector are present in the ZCD for this purpose.

The ZCDIF bit of the PIR3 register will be set when either edge detector is triggered and its associated enable bit is set. The INTP enables rising edge interrupts and the INTN bit enables falling edge interrupts. Both are located in the ZCDxCON register.

To fully enable the interrupt, the following bits must be set:

- ZCDIE bit of the PIE3 register
- INTP bit of the ZCDxCON register (for a rising edge detection)
- INTN bit of the ZCDxCON register (for a falling edge detection)
- PEIE and GIE bits of the INTCON register

Changing the POL bit will cause an interrupt, regardless of the level of the EN bit.

The ZCDIF bit of the PIR3 register must be cleared in software as part of the interrupt service. If another edge is detected while this flag is being cleared, the flag will still be set at the end of the sequence.

## 20.5 Correcting for ZCPINV Offset

The actual voltage at which the ZCD switches is the reference voltage at the non-inverting input of the ZCD op amp. For external voltage source waveforms other than square waves this voltage offset from zero causes the zero-cross event to occur either too early or too late.

### 20.5.1 CORRECTION BY AC COUPLING

When the external voltage source is sinusoidal, the effects of the ZCPINV offset can be eliminated by isolating the external voltage source from the ZCD pin with a capacitor in addition to the voltage reducing resistor. The capacitor will cause a phase shift resulting in the ZCD output switch in advance of the actual zero crossing event. The phase shift will be the same for both rising and falling zero crossings, which can be compensated for by either delaying the CPU response to the ZCD switch by a timer or other means, or selecting a capacitor value large enough that the phase shift is negligible.

To determine the series resistor and capacitor values for this configuration, start by computing the impedance, Z, to obtain a peak current of 300 μA. Next, arbitrarily select a suitably large non-polar capacitor and compute its reactance, $X_C$, at the external voltage source frequency. Finally, compute the series resistor, capacitor peak voltage, and phase shift by the formulas shown in Equation 20-2.

When this technique is used and the input signal is not present then the ZCD will tend to oscillate. To avoid this oscillation connect the ZCD pin to VDD or GND with a high-impedance resistor such as 200K.

**EQUATION 20-2: R-C CALCULATIONS**

$V_{peak}$ = external voltage source peak voltage
f = external voltage source frequency
C = series capacitor
R = series resistor
$V_C$ = Peak capacitor voltage
$\phi$ = Capacitor induced zero crossing phase advance in radians
$T_\phi$ = Time ZC event occurs before actual zero crossing

$$Z = \frac{V_{PEAK}}{3x10^{-4}}$$

$$X_C = \frac{1}{(2\pi fC)}$$

$$R = \sqrt{Z^2 - X_C}$$

$$V_C = X_C(3x10^{-4})$$

$$\phi = Tan^{-1}\left(\frac{X_C}{R}\right)$$

$$T_\phi = \frac{\phi}{(2\pi f)}$$

$V_{rms}$ = 120

**EQUATION 20-3: R-C CALCULATIONS EXAMPLE**

$$V_{peak} = V_{rms} \cdot \sqrt{2} = 169.7$$

f = 60 Hz
C = 0.1 µf

$$Z = \frac{V_{peak}}{3 \times 10^{-4}} = \frac{169.7}{3 \times 10^{-4}} = 565.7\ kOhms$$

$$X_C = \frac{1}{(2\pi fC)} = \frac{1}{(2\pi \cdot 60 \cdot 1 \cdot 10^{-7})} = 26.53\ \text{kOhms}$$

R = 560 kOhms

$$Z_R = \sqrt{(R^2 + X_c^{\ 2})} = 560.6\ kOhm\ \text{(using actual resistor)}$$

$$I_{peak} = \frac{V_{peak}}{Z_R} = 302.7 \cdot 10^{-6}$$

$$V_C = X_C \cdot I_{peak} = 8.0\ \text{V}$$

$$\phi = Tan^{-1}\left(\frac{X_C}{R}\right) = 0.047\ radians$$

$$T_\phi = \frac{\phi}{(2\pi f)} = 125.6\ \mu s$$

### 20.5.2 CORRECTION BY OFFSET CURRENT

When the waveform is varying relative to Vss then the zero cross is detected too early as the waveform falls and too late as the waveform rises. When the waveform is varying relative to VDD then the zero cross is detected too late as the waveform rises and too early as the waveform falls. The actual offset time can be determined for sinusoidal waveforms with the corresponding equations shown in Equation 20-4.

**EQUATION 20-4: ZCD EVENT OFFSET**

When External Voltage Source is relative to Vss:

$$T_{offset} = \frac{\text{asin}\left(\frac{Z_{cpinv}}{V_{peak}}\right)}{2\pi \bullet Freq}$$

When External Voltage Source is relative to VDD:

$$T_{offset} = \frac{\text{asin}\left(\frac{VDD - Z_{cpinv}}{V_{peak}}\right)}{2\pi \bullet Freq}$$

This offset time can be compensated for by adding a pull-up or pull-down biasing resistor to the ZCD pin. A pull-up resistor is used when the external voltage source is varying relative to Vss. A pull-down resistor is used when the voltage is varying relative to VDD.The resistor adds a bias to the ZCD pin so that the target external voltage source must go to zero to pull the pin voltage to the ZCPINV switching voltage. The pull-up or pull-down value can be determined with the equations shown in Equation 20-5.

**EQUATION 20-5: ZCD PULL-UP/DOWN**

When External Signal is relative to Vss:

$$R_{pullup} = \frac{R_{series}(V_{pullup} - Z_{cpinv})}{Z_{cpinv}}$$

When External Signal is relative to VDD:

$$R_{pulldown} = \frac{R_{series}(Z_{cpinv})}{(VDD - Z_{cpinv})}$$

## 20.6 Handling $V_{peak}$ variations

If the peak amplitude of the external voltage is expected to vary then the series resistor must be selected to keep the ZCD current source and sink below the design maximum range of ± 600 µA for the maximum expected voltage and high enough to be detected accurately at the minimum peak voltage. A general rule of thumb is that the maximum peak voltage can be no more than six times the minimum peak voltage. To ensure that the maximum current does not exceed ± 600 µA and the minimum is at least ± 100 µA compute the series resistance as shown in Equation 20-6. The compensating pull-up for this series resistance can be determined with Equation 20-5 because the pull-up value is independent from the peak voltage.

**EQUATION 20-6: SERIES R FOR V RANGE**

$$R_{series} = \frac{V_{maxpeak} + V_{minpeak}}{7 \times 10^{-4}}$$

## 20.7 Operation During Sleep

The ZCD current sources and interrupts are unaffected by Sleep.

## 20.8 Effects of a Reset

The ZCD circuit can be configured to default to the active or inactive state on Power-On-Reset (POR). When the $\overline{ZCD}$ Configuration bit is cleared, the ZCD circuit will be active at POR. When the $\overline{ZCD}$ Configuration bit is set, the ZCDEN bit of the ZCDCON register must be set to enable the ZCD module.

## 20.9 Register Definitions: ZCD Control

Long bit name prefixes for the zero-cross detect peripheral is shown in Table 20-1. Refer to **1.1.2.2 "Long Bit Names"** for more information

**TABLE 20-1:**

| Peripheral | Bit Name Prefix |
|---|---|
| ZCD1 | ZCD1 |

**REGISTER 20-1: ZCDxCON: ZERO-CROSS DETECTION CONTROL REGISTER**

| R/W-0/0 | U-0 | R-x/x | R/W-0/0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| EN | — | OUT | POL | — | — | INTP | INTN |
| 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 configuration bits |

bit 7 **EN:** Zero-Cross Detection Enable bit(1)
- 1 = Zero-cross detect is enabled. ZCD pin is forced to output to source and sink current.
- 0 = Zero-cross detect is disabled. ZCD pin operates according to PPS and TRIS controls.

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

bit 5 **OUT:** Zero-Cross Detection Logic Level bit

POL bit = 0:
- 1 = ZCD pin is sinking current
- 0 = ZCD pin is sourcing current

POL bit = 1:
- 1 = ZCD pin is sourcing current
- 0 = ZCD pin is sinking current

bit 4 **POL:** Zero-Cross Detection Logic Output Polarity bit
- 1 = ZCD logic output is inverted
- 0 = ZCD logic output is not inverted

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

bit 1 **INTP:** Zero-Cross Positive Edge Interrupt Enable bit
- 1 = ZCDIF bit is set on low-to-high OUT transition
- 0 = ZCDIF bit is unaffected by low-to-high OUT transition

bit 0 **INTN:** Zero-Cross Negative Edge Interrupt Enable bit
- 1 = ZCDIF bit is set on high-to-low OUT transition
- 0 = ZCDIF bit is unaffected by high-to-low OUT transition

**Note 1:** The EN bit has no effect when the $\overline{ZCD}$ Configuration bit is cleared.

**TABLE 20-2: SUMMARY OF REGISTERS ASSOCIATED WITH THE ZCD MODULE**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on page |
|---|---|---|---|---|---|---|---|---|---|
| PIE3 | — | — | COG2IE | ZCDIE | CLC4IE | CLC3IE | CLC2IE | CLC1IE | 111 |
| PIR3 | — | — | COG2IF | ZCDIF | CLC4IF | CLC3IF | CLC2IF | CLC1IF | 117 |
| ZCD1CON | EN | — | OUT | POL | — | — | INTP | INTN | 232 |

**Legend:** — = unimplemented, read as '0'. Shaded cells are unused by the ZCD module.

**TABLE 20-3: SUMMARY OF CONFIGURATION WORD WITH THE ZCD MODULE**

| 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 | $\overline{\text{DEBUG}}$ | $\overline{\text{LPBOR}}$ | BORV | STVREN | PLLEN | 73 |
| | 7:0 | $\overline{\text{ZCD}}$ | — | — | — | — | PPS1WAY | WRT<1:0> | | |

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

## 21.0 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 21-1 is a block diagram of the Timer0 module.

## 21.1 Timer0 Operation

The Timer0 module can be used as either an 8-bit timer or an 8-bit counter.

### 21.1.1 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. |

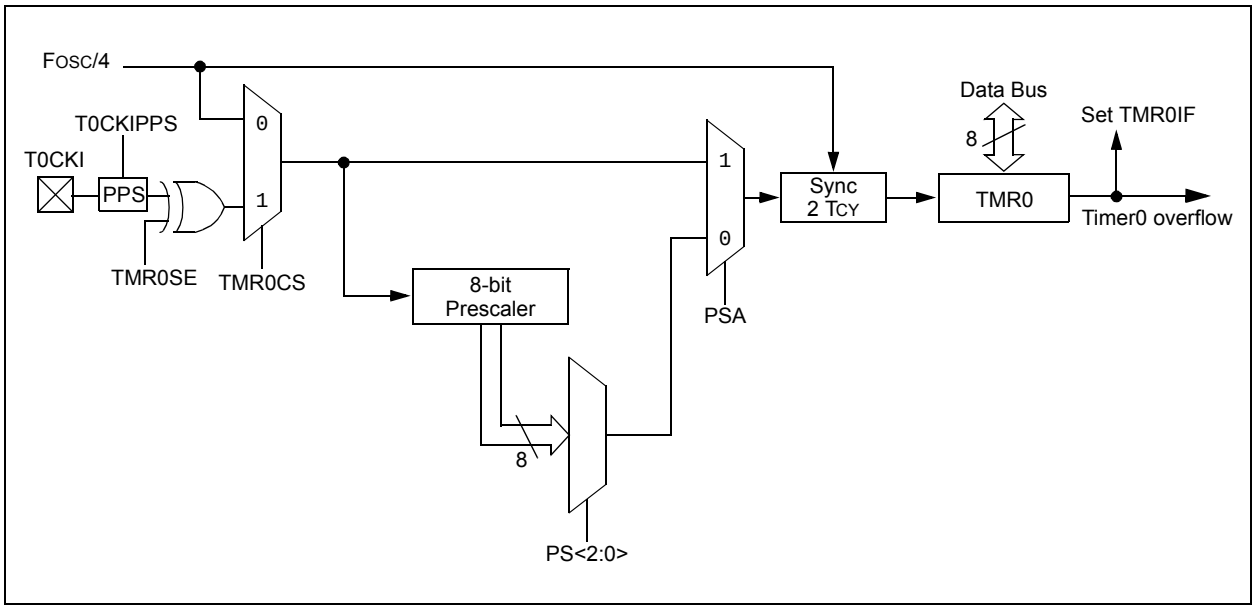
### 21.1.2 8-BIT COUNTER MODE

In 8-Bit Counter mode, the Timer0 module will increment on every rising or falling edge of the T0CKI pin.

8-Bit Counter mode using the T0CKI pin is selected by setting the TMR0CS bit in the OPTION_REG register to '1'.

The rising or falling transition of the incrementing edge for either input source is determined by the TMR0SE bit in the OPTION_REG register.

**FIGURE 21-1: BLOCK DIAGRAM OF THE TIMER0**



### 21.1.3 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.

### 21.1.4 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. |

### 21.1.5 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 Table 36-12: Timer0 and Timer1 External Clock Requirements.

### 21.1.6 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.

## 21.2 Register Definitions: Option Register

**REGISTER 21-1: 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 21-1: SUMMARY OF REGISTERS ASSOCIATED WITH TIMER0**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 108 |
| OPTION_REG | $\overline{\text{WPUEN}}$ | INTEDG | TMR0CS | TMR0SE | PSA | PS<2:0> | | | 236 |
| TMR0 | Timer0 Module Register | | | | | | | | 234* |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 152 |

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

## 22.0 TIMER1/3/5 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
- Auto-conversion Trigger (with CCP)
- Selectable Gate Source Polarity
- Gate Toggle mode
- Gate Single-pulse mode
- Gate Value Status
- Gate Event Interrupt

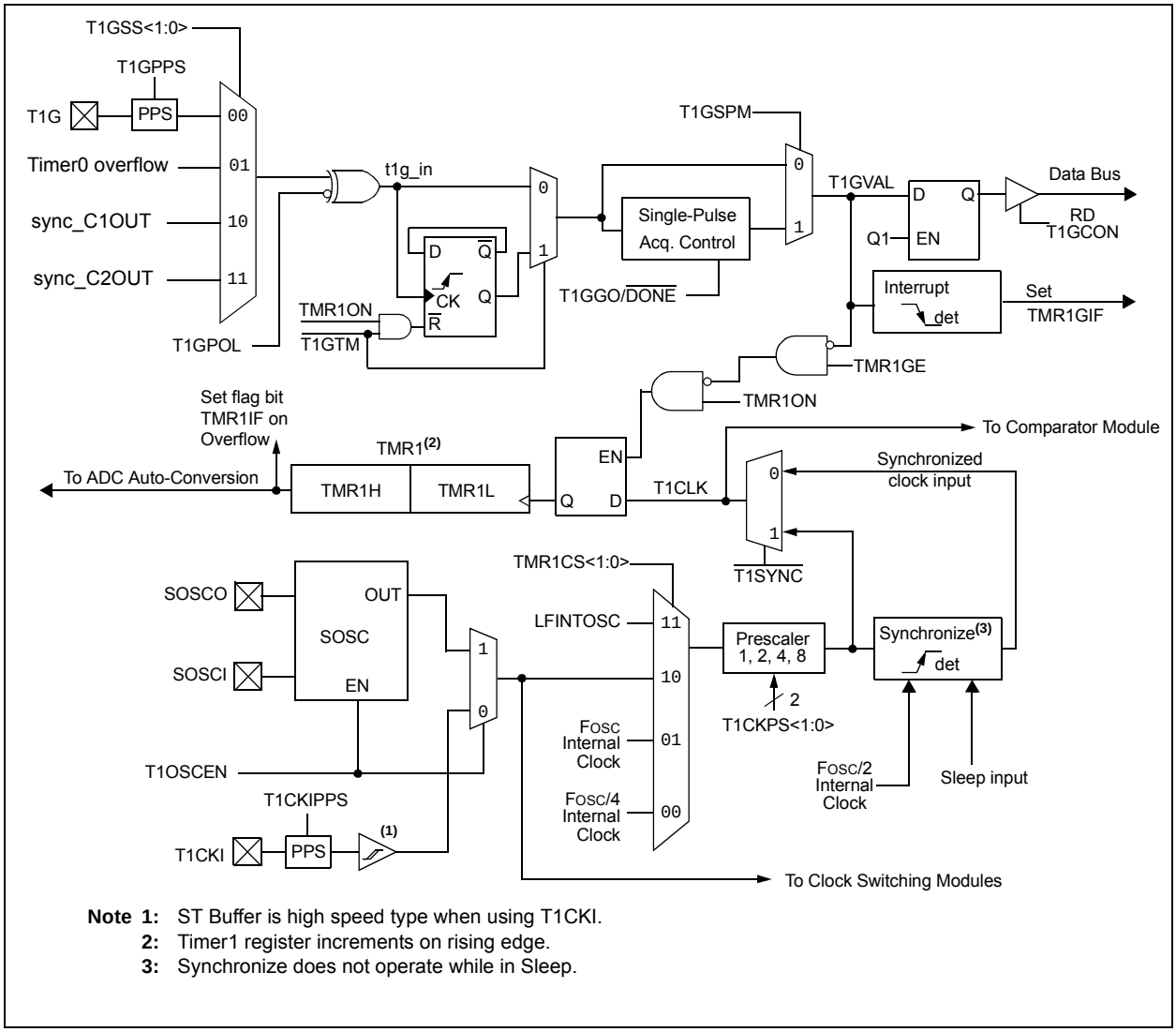
Figure 22-1 is a block diagram of the Timer1 module.

This device has three instances of Timer1 type modules. They include:

- Timer1
- Timer3
- Timer5

| **Note:** | All references to Timer1 and Timer1 Gate apply to Timer3 and Timer5. |
|---|---|

**FIGURE 22-1: 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.

## 22.1 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 ON and GE bits in the T1CON and T1GCON registers, respectively. Table 22-1 displays the Timer1 enable selections.

**TABLE 22-1: TIMER1 ENABLE SELECTIONS**

| TMR1ON | TMR1GE | Timer1 Operation |
|---|---|---|
| 0 | 0 | Off |
| 0 | 1 | Off |
| 1 | 0 | Always On |
| 1 | 1 | Count Enabled |

## 22.2 Clock Source Selection

The CS<1:0> and OSCEN bits of the T1CON register are used to select the clock source for Timer1. Table 22-2 displays the clock source selections.

### 22.2.1 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

### 22.2.2 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, which can be synchronized to the microcontroller system clock or 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 22-2: CLOCK SOURCE SELECTIONS**

| TMR1CS<1:0> | T1OSCEN | Clock Source |
|---|---|---|
| 11 | x | LFINTOSC |
| 10 | 0 | External Clocking on T1CKI Pin |
| 01 | x | System Clock (FOSC) |
| 00 | x | Instruction Clock (FOSC/4) |

## 22.3 Timer1 Prescaler

Timer1 has four prescaler options allowing 1, 2, 4 or 8 divisions of the clock input. The CKPS 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.

## 22.4 Timer1 (Secondary) Oscillator

A dedicated low-power 32.768 kHz oscillator circuit is built-in between pins SOSCI (input) and SOSCO (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 OSCEN 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, OSCEN 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 reasonably stable. |

## 22.5 Timer1 Operation in Asynchronous Counter Mode

If the control bit $\overline{\text{SYNC}}$ 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 22.5.1 "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. |

### 22.5.1 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.

## 22.6 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.

### 22.6.1 TIMER1 GATE ENABLE

The Timer1 Gate Enable mode is enabled by setting the GE bit of the T1GCON register. The polarity of the Timer1 Gate Enable mode is configured using the GPOL 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 22-3 for timing details.

**TABLE 22-3: TIMER1 GATE ENABLE SELECTIONS**

| T1CLK | T1GPOL | T1G | Timer1 Operation |
|---|---|---|---|
| ↑ | 0 | 0 | Counts |
| ↑ | 0 | 1 | Holds Count |
| ↑ | 1 | 0 | Holds Count |
| ↑ | 1 | 1 | Counts |

### 22.6.2 TIMER1 GATE SOURCE SELECTION

Timer1 gate source selections are shown in Table 22-4. 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 22-4: 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) |

#### 22.6.2.1 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.

#### 22.6.2.2 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.

#### 22.6.2.3 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 19.4.1 "Comparator Output Synchronization"**.

#### 22.6.2.4 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 19.4.1 "Comparator Output Synchronization"**.

### 22.6.3 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 22-4 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. |

### 22.6.4 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 22-5 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 22-6 for timing details.

### 22.6.5 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).

### 22.6.6 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).

## 22.7 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:

- ON 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. |
|---|---|

## 22.8 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:

- ON 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
- $\overline{\text{SYNC}}$ bit of the T1CON register must be set
- CS bits of the T1CON register must be configured
- OSCEN 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.

Secondary oscillator will continue to operate in Sleep regardless of the $\overline{\text{SYNC}}$ bit setting.

## 22.9 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 in the CCPR1H:CCPR1L register pair matches the value in the TMR1H:TMR1L register pair. This event can be an Auto-conversion Trigger.

For more information, see **Section 24.0 "Capture/Compare/PWM Modules"**.

## 22.10 CCP Auto-Conversion Trigger

When any of the CCP's are configured to trigger an auto-conversion, the trigger will clear the TMR1H:TMR1L register pair. This auto-conversion does not cause a Timer1 interrupt. The CCP module may still be configured to generate a CCP interrupt.

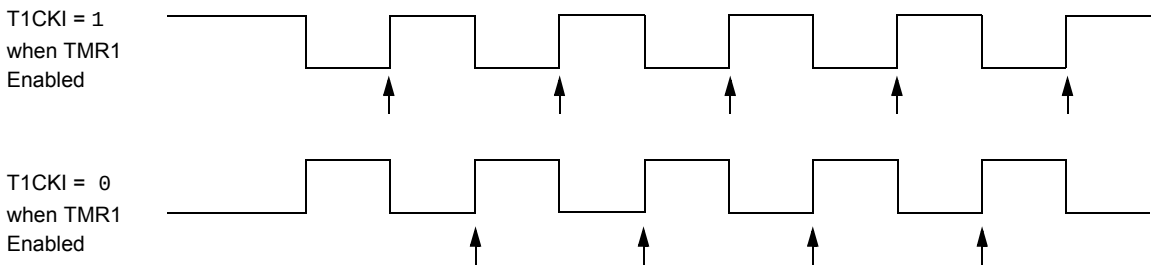
In this mode of operation, the CCPR1H:CCPR1L register pair becomes the period register for Timer1.

Timer1 should be synchronized and FOSC/4 should be selected as the clock source in order to utilize the Auto-conversion Trigger. Asynchronous operation of Timer1 can cause an Auto-conversion Trigger to be missed.

In the event that a write to TMR1H or TMR1L coincides with an Auto-conversion Trigger from the CCP, the write will take precedence.

For more information, see **Section 24.2.1 "Auto-Conversion Trigger"**.

**FIGURE 22-2: 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 22-3: TIMER1 GATE ENABLE MODE**

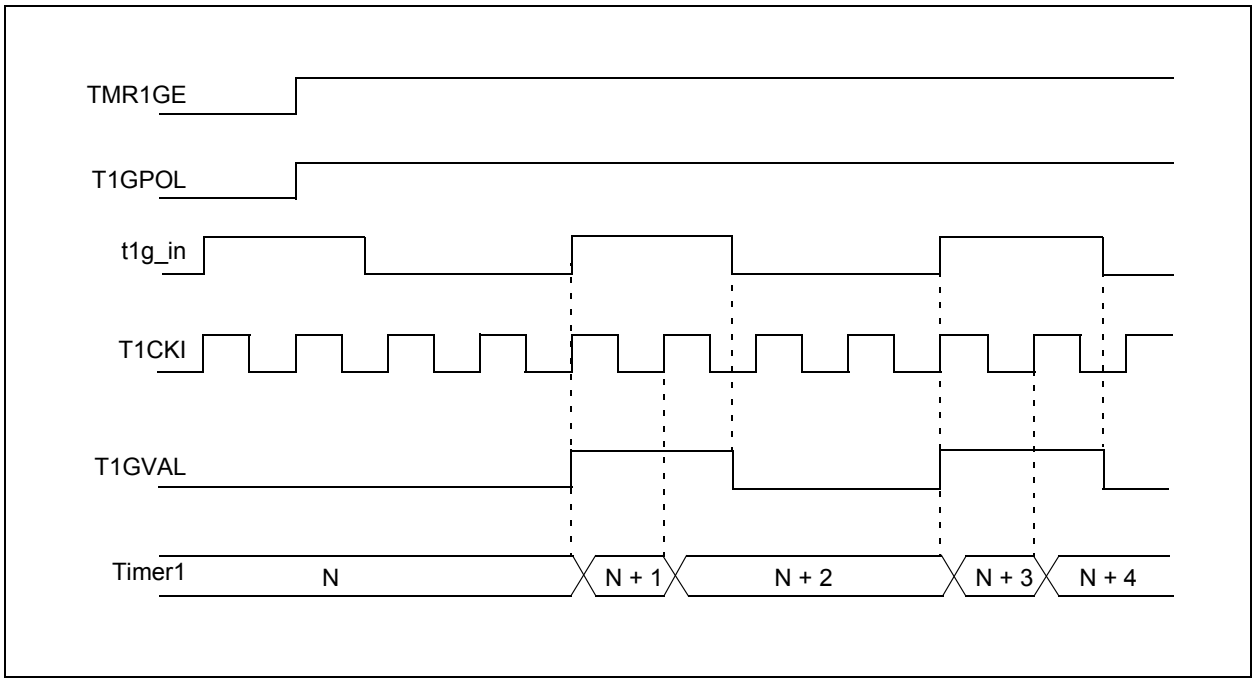



**FIGURE 22-4: TIMER1 GATE TOGGLE MODE**

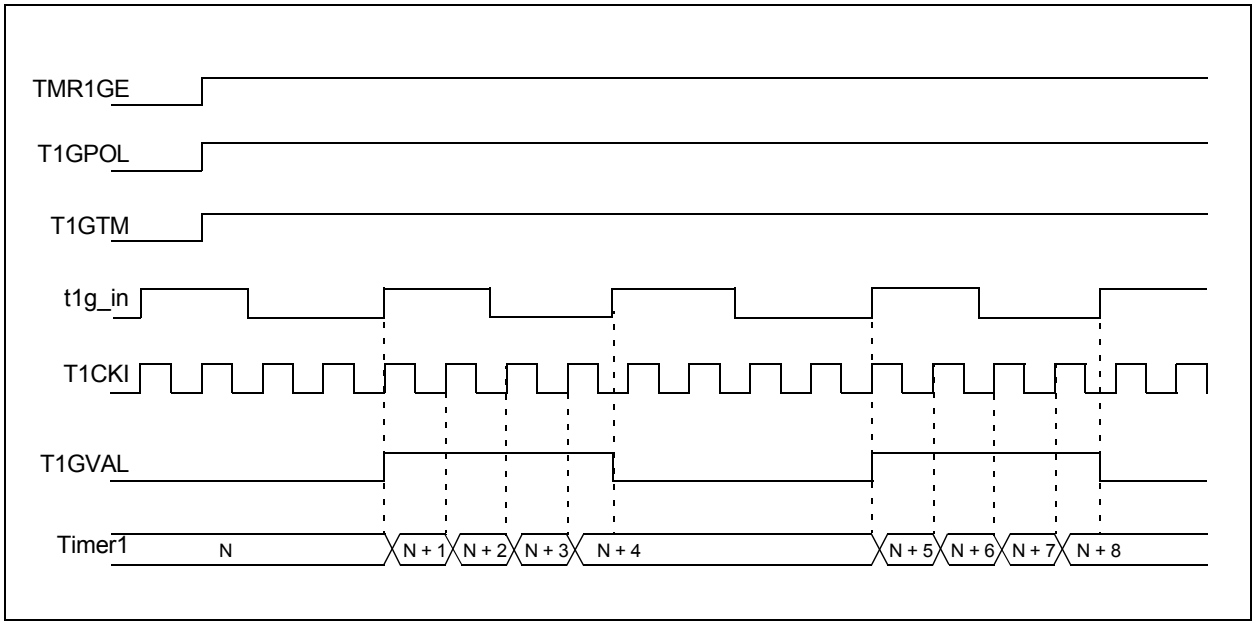


**FIGURE 22-5: TIMER1 GATE SINGLE-PULSE MODE**

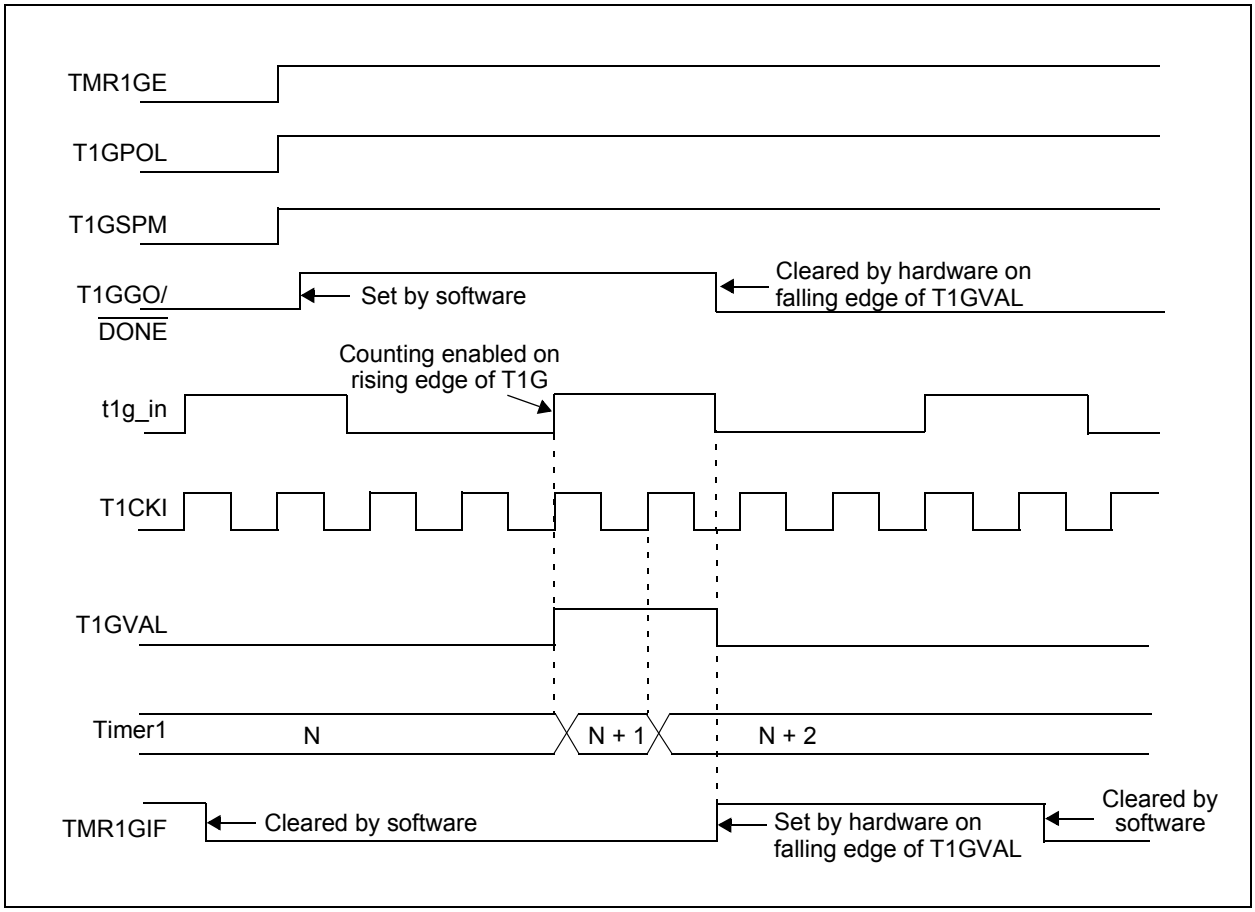
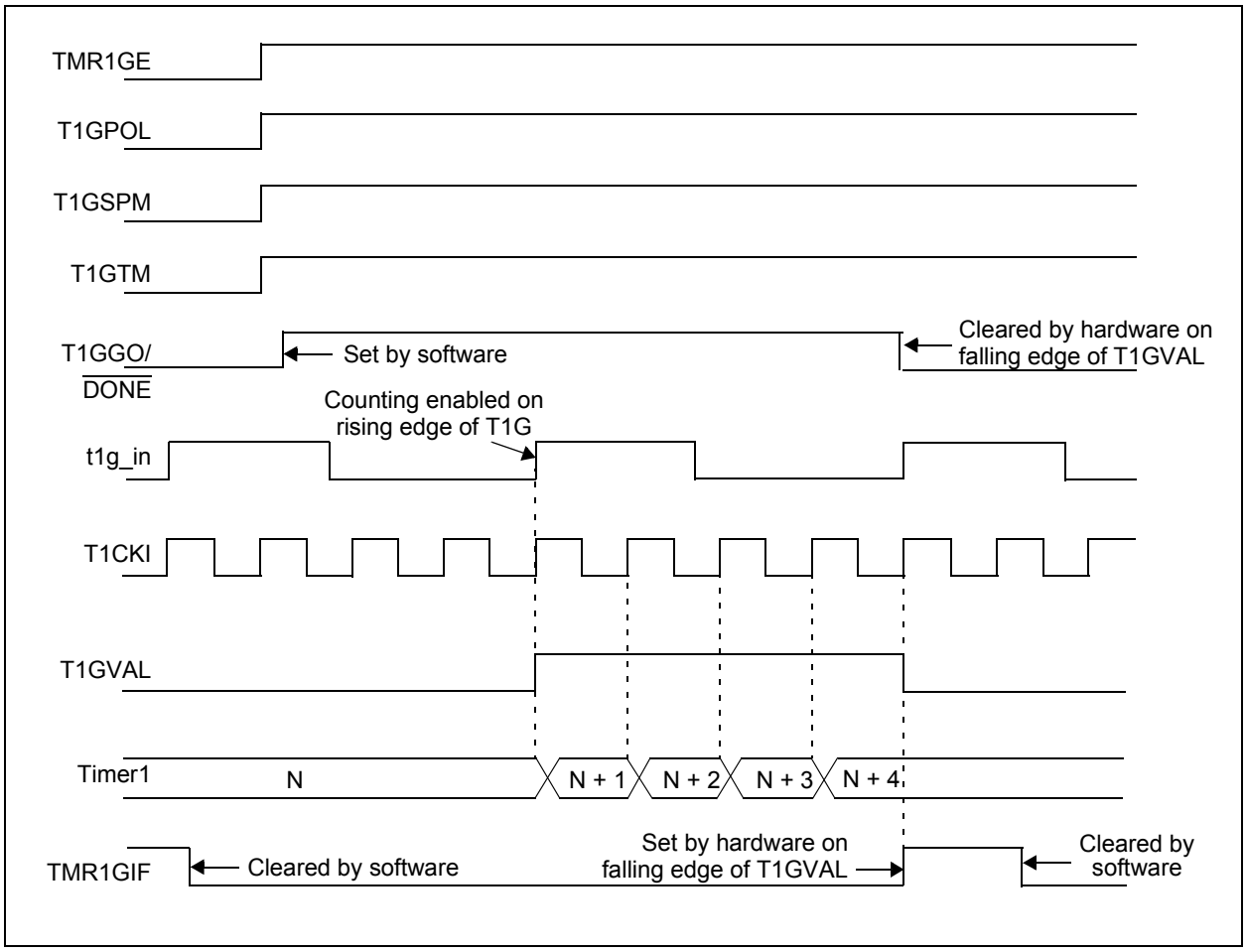


**FIGURE 22-6: TIMER1 GATE SINGLE-PULSE AND TOGGLE COMBINED MODE**



## 22.11 Register Definitions: Timer1 Control

Long bit name prefixes for the Timer1 peripherals are shown in Table 22-5. Refer to **Section 1.1.2.2 "Long Bit Names"** for more information

**TABLE 22-5:**

| Peripheral | Bit Name Prefix |
|---|---|
| Timer1 | T1 |
| Timer3 | T3 |
| Timer5 | T5 |

**REGISTER 22-1: 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 |
|---|---|---|---|---|---|---|---|
| CS<1:0> | | CKPS<1:0> | | OSCEN | $\overline{\text{SYNC}}$ | — | ON |
| 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 **CS<1:0>:** Timer1 Clock Source Select bits
- 11 = LFINTOSC
- 10 = Timer1 clock source is pin or oscillator:(1)
  - If T1OSCEN = 0:
  - External clock from T1CKI pin (on the rising edge)
  - If T1OSCEN = 1:
  - Crystal oscillator on SOSCI/SOSCO pins
- 01 = Timer1 clock source is system clock (FOSC)
- 00 = Timer1 clock source is instruction clock (FOSC/4)

bit 5-4 **CKPS<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 **OSCEN:** LP Oscillator Enable Control bit(1)
- 1 = Dedicated secondary oscillator circuit enabled
- 0 = Dedicated secondary oscillator circuit disabled

bit 2 **$\overline{\text{SYNC}}$:** 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 **ON:** Timer1 On bit
- 1 = Enables Timer1
- 0 = Stops Timer1 and clears Timer1 gate flip-flop

**Note 1:** Timer1 only. Reserved, do not use for Timer3 and Timer5.

**REGISTER 22-2: 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 |
|---|---|---|---|---|---|---|---|
| GE | GPOL | GTM | GSPM | GGO/$\overline{\text{DONE}}$ | GVAL | GSS<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 **GE:** 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 **GPOL:** 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 **GTM:** 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 **GSPM:** 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 **GGO/$\overline{\text{DONE}}$:** 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 **GVAL:** Timer1 Gate Value Status bit

Indicates the current state of the Timer1 gate that could be provided to TMR1H:TMR1L
Unaffected by Timer1 Gate Enable (TMR1GE)

bit 1-0 **GSS<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

**TABLE 22-6: SUMMARY OF REGISTERS ASSOCIATED WITH TIMER1**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| ANSELA | — | — | ANSA5 | ANSA4 | ANSA3 | ANSA2 | ANSA1 | ANSA0 | 153 |
| CCPxCON | EN | — | OUT | FMT | MODE<3:0> | | | | 281 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 108 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 109 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 115 |
| TMRxH | Holding Register for the Most Significant Byte of the 16-bit TMR1/3/5 Register | | | | | | | | 237* |
| TMRxL | Holding Register for the Least Significant Byte of the 16-bit TMR1/3/5 Register | | | | | | | | 237* |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 152 |
| TxCON | CS<1:0> | | CKPS<1:0> | | OSCEN | $\overline{\text{SYNC}}$ | — | ON | 245 |
| TxGCON | GE | GPOL | GTM | GSPM | $\overline{\text{GGO}}$/$\overline{\text{DONE}}$ | GVAL | GSS<1:0> | | 246 |

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

* Page provides register information.

## 23.0 TIMER2/4/6/8 MODULE

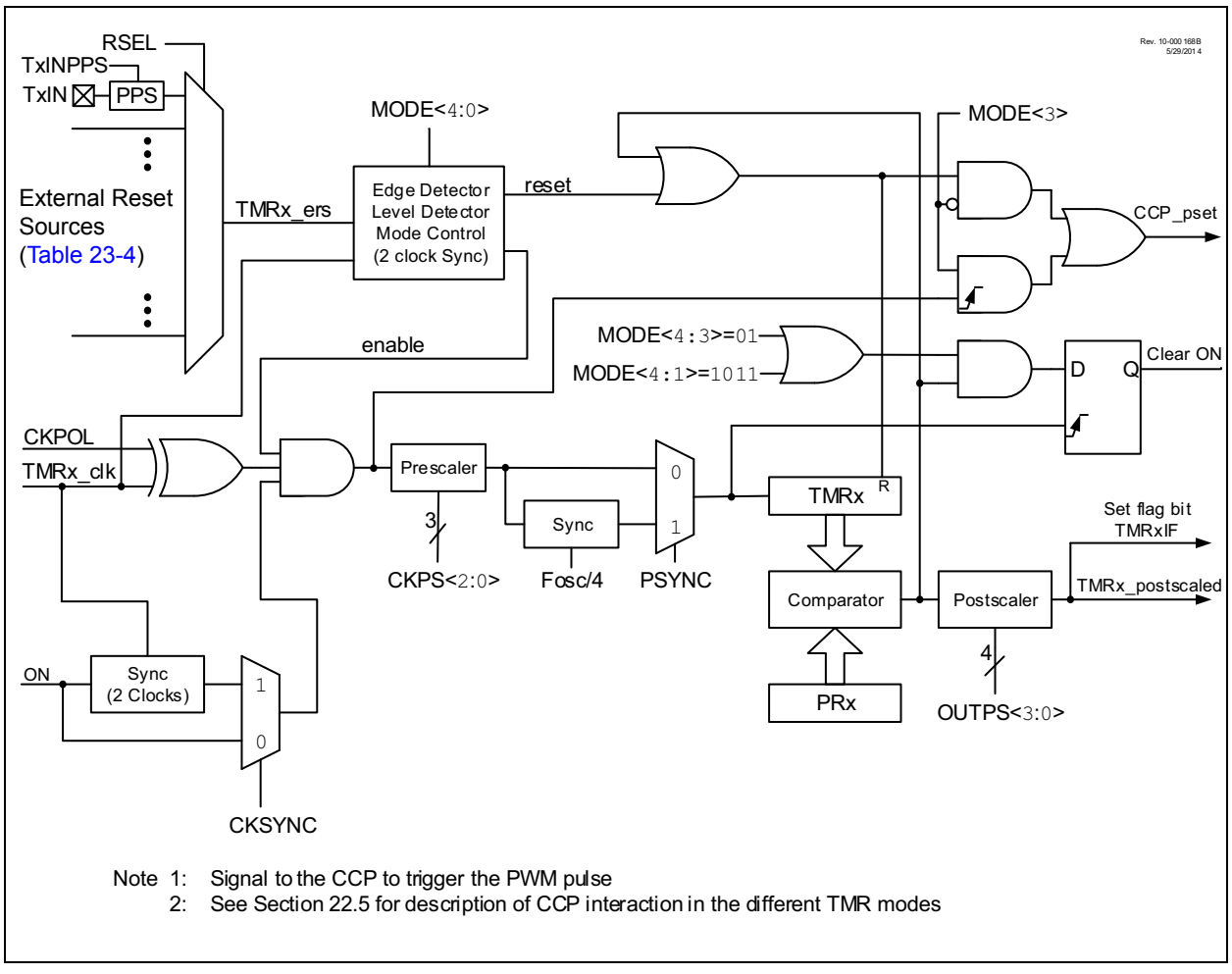
The Timer2/4/6/8 modules are 8-bit timers that can operate as free-running period counters or in conjunction with external signals that control start, run, freeze, and reset operation in One-Shot and Monostable modes of operation. Sophisticated waveform control such as pulse density modulation are possible by combining the operation of these timers with other internal peripherals such as the comparators and CCP modules. Features of the timer include:

- 8-bit timer register
- 8-bit period register
- Selectable external hardware timer Resets
- Programmable prescaler (1:1 to 1:128)
- Programmable postscaler (1:1 to 1:16)
- Selectable synchronous/asynchronous operation
- Alternate clock sources
- Interrupt-on-period
- Three modes of operation:
  - Free Running Period
  - One-shot
  - Monostable

See Figure 23-1 for a block diagram of Timer2. See Figure 23-2 for the clock source block diagram.

| **Note:** | Three identical Timer2 modules are implemented on this device. The timers are named Timer2, Timer4, Timer6, and Timer8. All references to Timer2 apply as well to Timer4, Timer6 and Timer8. All references to T2PR apply as well to T4PR, T6PR and T8PR. |
|---|---|

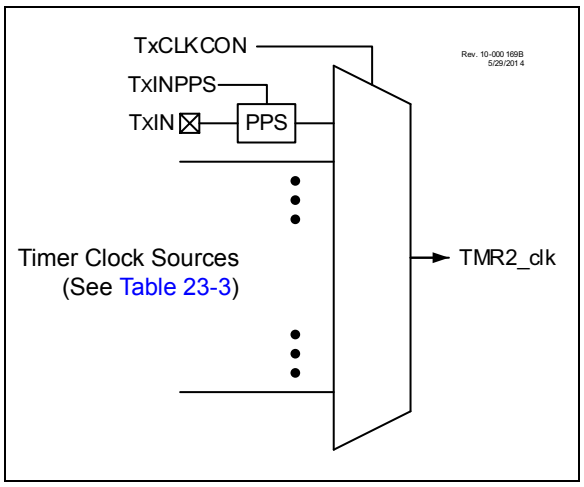
**FIGURE 23-1: TIMER2 BLOCK DIAGRAM**




Note 1: Signal to the CCP to trigger the PWM pulse
2: See Section 22.5 for description of CCP interaction in the different TMR modes

**FIGURE 23-2: TIMER2 CLOCK SOURCE BLOCK DIAGRAM**




## 23.1 Timer2 Operation

Timer2 operates in three major modes:

- Free Running Period
- One-shot
- Monostable

Within each mode there are several options for starting, stopping, and reset. Table 23-1 lists the options.

In all modes, the TMR2 count register is incremented on the rising edge of the clock signal from the programmable prescaler. When TMR2 equals T2PR, a high level is output to the postscaler counter. TMR2 is cleared on the next clock input.

An external signal from hardware can also be configured to gate the timer operation or force a TMR2 count Reset. In Gate modes the counter stops when the gate is disabled and resumes when the gate is enabled. In Reset modes the TMR2 count is reset on either the level or edge from the external source.

The TMR2 and T2PR registers are both directly readable and writable. The TMR2 register is cleared and the T2PR register initializes to FFh on any device Reset. Both the prescaler and postscaler counters are cleared on the following events:

- a write to the TMR2 register
- a write to the T2CON register
- any device Reset
- External Reset Source event that resets the timer.

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

### 23.1.1 FREE RUNNING PERIOD MODE

The value of TMR2 is compared to that of the Period register, T2PR, on each clock cycle. When the two values match, the comparator resets the value of TMR2 to 00h on the next cycle and increments the output postscaler counter. When the postscaler count equals the value in the OUTPS<4:0> bits of the TMRxCON1 register then a one clock period wide pulse occurs on the TMR2_postscaled output, and the postscaler count is cleared.

### 23.1.2 ONE-SHOT MODE

The One-Shot mode is identical to the Free Running Period mode except that the ON bit is cleared and the timer is stopped when TMR2 matches T2PR and will not restart until the T2ON bit is cycled off and on. Postscaler OUTPS<4:0> values other than 0 are meaningless in this mode because the timer is stopped at the first period event and the postscaler is reset when the timer is restarted.

### 23.1.3 MONOSTABLE MODE

Monostable modes are similar to One-Shot modes except that the ON bit is not cleared and the timer can be restarted by an external Reset event.

## 23.2 PRx Period Register

The PRx period register is double buffered, software reads and writes the PRx register. However, the timer uses a buffered PRx register for operation. Software does not have direct access to the buffered PRx register. The contents of the PRx register is transferred to the buffer by any of the following events:

- A write to the TMRx register
- A write to the TMRxCON register
- When TMRx = PRx buffer and the prescaler rolls over
- An external Reset event

## 23.3 Timer2 Output

The Timer2 module's primary output is TMR2_postscaled, which pulses for a single TMR2_clk period when the postscaler counter matches the value in the OUTPS bits of the TMR2xCON register. The T2PR postscaler is incremented each time the TMR2 value matches the T2PR value. This signal can be selected as an input to several other input modules:

- The ADC module, as an Auto-conversion Trigger
- COG, as an auto-shutdown source

In addition, the Timer2 is also used by the CCP module for pulse generation in PWM mode. Both the actual TMR2 value as well as other internal signals are sent to the CCP module to properly clock both the period and pulse width of the PWM signal. See **Section 24.6 "CCP/PWM Clock Selection"** for more details on setting up Timer2 for use with the CCP, as well as the timing diagrams in **Section 23.6 "Operation Examples"** for examples of how the varying Timer2 modes affect CCP PWM output.

## 23.4 External Reset Sources

In addition to the clock source, the Timer2 also takes in an external Reset source. This external Reset source is selected for Timer2, Timer4, Timer6 and Timer8 with the T2RST, T4RST, T6RST and T8RST registers, respectively. This source can control starting and stopping of the timer, as well as resetting the timer, depending on which mode the timer is in. The mode of the timer is controlled by the MODE<4:0> bits of the TMRxHLT register. Edge-Triggered modes require six Timer clock periods between external triggers. Level-Triggered modes require the triggering level to be at least three Timer clock periods long. External triggers are ignored while in Debug Freeze mode.

**TABLE 23-1: TIMER2 OPERATING MODES**

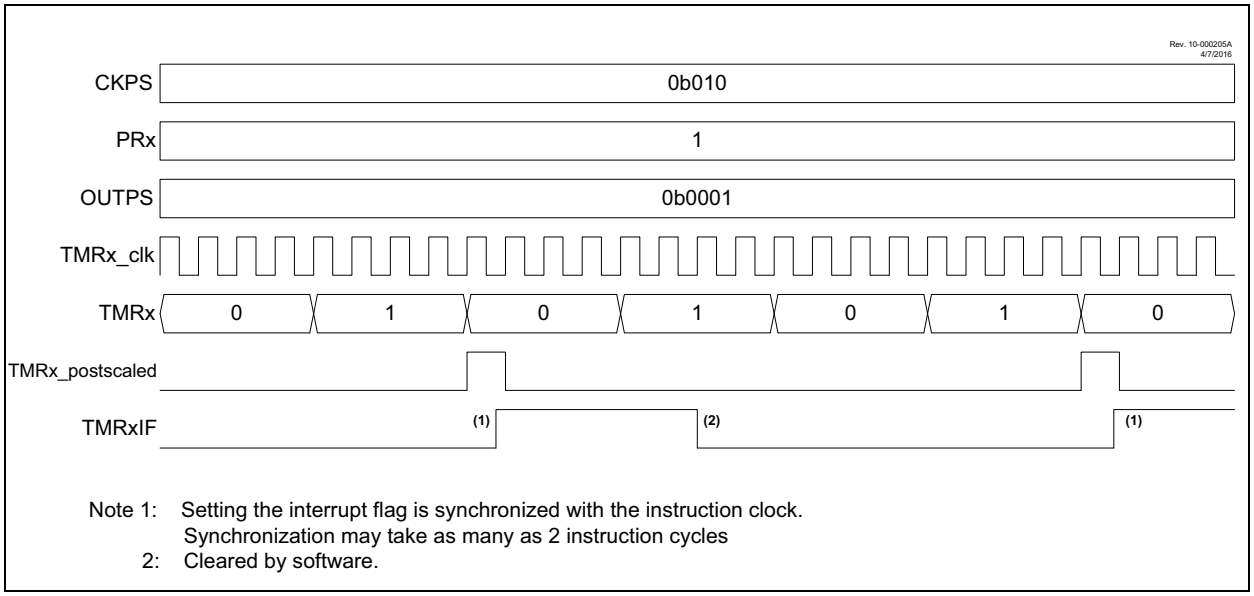
| Mode | MODE<4:0> | | Output Operation | Operation | Timer Control | | |
|---|---|---|---|---|---|---|---|
| | <4:3> | <2:0> | | | Start | Reset | Stop |
| Free Running Period | 00 | 000 | Period Pulse | Software gate (Figure 23-4) | ON = 1 | — | ON = 0 |
| | | 001 | | Hardware gate, active-high (Figure 23-5) | ON = 1 and TMRx_ers = 1 | — | ON = 0 or TMRx_ers = 0 |
| | | 010 | | Hardware gate, active-low | ON = 1 and TMRx_ers = 0 | — | ON = 0 or TMRx_ers = 1 |
| | | 011 | Period Pulse with Hardware Reset | Rising or falling edge Reset | ON = 1 | TMRx_ers ↕ | ON = 0 |
| | | 100 | | Rising edge Reset (Figure 23-6) | | TMRx_ers ↑ | |
| | | 101 | | Falling edge Reset | | TMRx_ers ↓ | |
| | | 110 | | Low level Reset | | TMRx_ers = 0 | ON = 0 or TMRx_ers = 0 |
| | | 111 | | High level Reset (Figure 23-7) | | TMRx_ers = 1 | ON = 0 or TMRx_ers = 1 |
| One-shot | 01 | 000 | One-shot | Software start (Figure 23-8) | ON = 1 | — | ON = 0 or Next clock after TMRx = PRx (**Note 2**) |
| | | 001 | Edge triggered start (**Note 1**) | Rising edge start (Figure 23-9) | ON = 1 and TMRx_ers ↑ | — | |
| | | 010 | | Falling edge start | ON = 1 and TMRx_ers ↓ | — | |
| | | 011 | | Any edge start | ON = 1 and TMRx_ers ↕ | — | |
| | | 100 | Edge triggered start and hardware Reset (**Note 1**) | Rising edge start and Rising edge Reset (Figure 23-10) | ON = 1 and TMRx_ers ↑ | TMRx_ers ↑ | |
| | | 101 | | Falling edge start and Falling edge Reset | ON = 1 and TMRx_ers ↓ | TMRx_ers ↓ | |
| | | 110 | | Rising edge start and Low level Reset (Figure 23-11) | ON = 1 and TMRx_ers ↑ | TMRx_ers = 0 | |
| | | 111 | | Falling edge start and High level Reset | ON = 1 and TMRx_ers ↓ | TMRx_ers = 1 | |
| Mono-stable | 10 | 000 | Reserved | | | | |
| | | 001 | Edge triggered start (**Note 1**) | Rising edge start (Figure 23-12) | ON = 1 and TMRx_ers ↑ | — | ON = 0 or Next clock after TMRx = PRx (**Note 3**) |
| | | 010 | | Falling edge start | ON = 1 and TMRx_ers ↓ | — | |
| | | 011 | | Any edge start | ON = 1 and TMRx_ers ↕ | — | |
| Reserved | | 100 | Reserved | | | | |
| Reserved | | 101 | Reserved | | | | |
| One-shot | | 110 | Level triggered start and hardware Reset | High level start and Low level Reset (Figure 23-13) | ON = 1 and TMRx_ers = 1 | TMRx_ers = 0 | ON = 0 or Held in Reset (**Note 2**) |
| | | 111 | | Low level start & High level Reset | ON = 1 and TMRx_ers = 0 | TMRx_ers = 1 | |
| Reserved | 11 | xxx | Reserved | | | | |

**Note 1:** If ON = 0 then an edge is required to restart the timer after ON = 1.
**2:** When TMRx = PRx then the next clock clears ON and stops TMRx at 00h.
**3:** When TMRx = PRx then the next clock stops TMRx at 00h but does not clear ON.

## 23.5 Timer2 Interrupt

Timer2 can also generate a device interrupt. The interrupt is generated when the postscaler counter matches one of 16 postscale options (from 1:1 through 1:16), which are selected with the postscaler control bits, OUTPS<3:0> of the T2CON register. The interrupt is enabled by setting the TMR2IE interrupt enable bit of the PIE1 register. Interrupt timing is illustrated in Figure 23-3.

**FIGURE 23-3: TIMER2 PRESCALER, POSTSCALER, AND INTERRUPT TIMING DIAGRAM**




Note 1: Setting the interrupt flag is synchronized with the instruction clock. Synchronization may take as many as 2 instruction cycles
2: Cleared by software.

## 23.6 Operation Examples

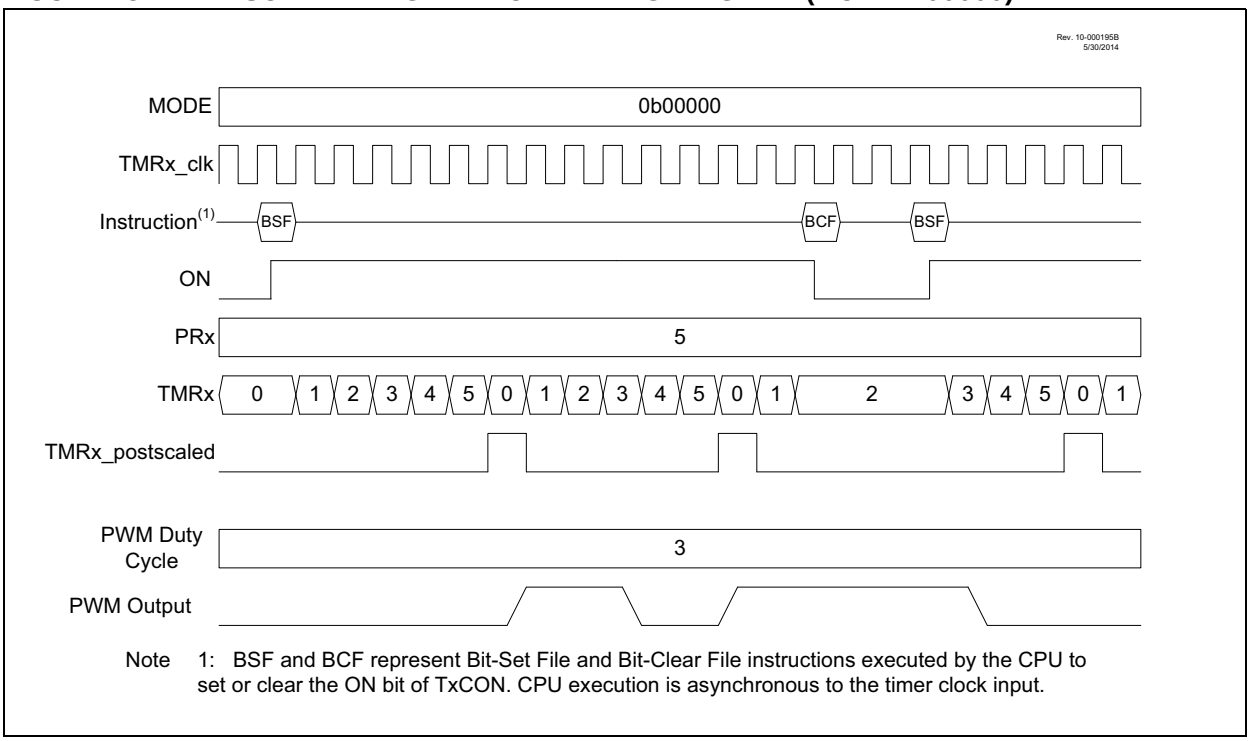
Unless otherwise specified, the following notes apply to the following timing diagrams:

- Both the prescaler and postscaler are set to 1:1 (both the CKPS and OUTPS bits in the TxCON register are cleared).
- The diagrams illustrate any clock except Fosc/4 and show clock-sync delays of at least two full cycles for both ON and Timer2_ers. When using Fosc/4, the clock-sync delay is at least one instruction period for Timer2_ers; ON applies in the next instruction period.
- The PWM Duty Cycle and PWM output are illustrated assuming that the timer is used for the PWM function of the CCP module as described in **Section 24.6 "CCP/PWM Clock Selection"**. The signals are not a part of the Timer2 module.

### 23.6.1 SOFTWARE GATE MODE

This mode corresponds to legacy Timer2 operation. The timer increments with each clock input when ON = 1 and does not increment when ON = 0. When the TMRx count equals the PRx period count the timer resets on the next clock and continues counting from 0. Operation with the ON bit software controlled is illustrated in Figure 23-4. With PRx = 5, the counter advances until TMRx = 5, and goes to zero with the next clock.

**FIGURE 23-4: SOFTWARE GATE MODE TIMING DIAGRAM (MODE = 00000)**




Note 1: BSF and BCF represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

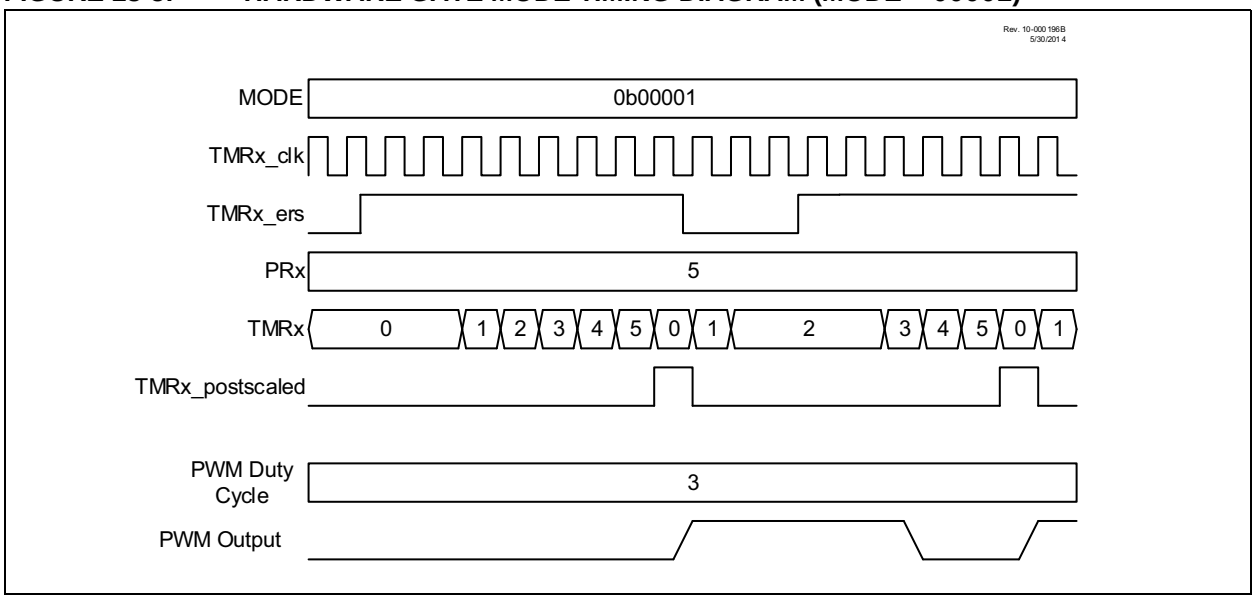
### 23.6.2 HARDWARE GATE MODE

The Hardware Gate modes operate the same as the Software Gate mode except the TMRx_ers external signal can also gate the timer. When used with the CCP the gating extends the PWM period. If the timer is stopped when the PWM output is high then the duty cycle is also extended.

When MODE<4:0> = 00001 then the timer is stopped when the external signal is high. When MODE<4:0> = 00010 then the timer is stopped when the external signal is low.

Figure 23-5 illustrates the Hardware Gating mode for MODE<4:0> = 00001 in which a high input level starts the counter.

**FIGURE 23-5: HARDWARE GATE MODE TIMING DIAGRAM (MODE = 00001)**



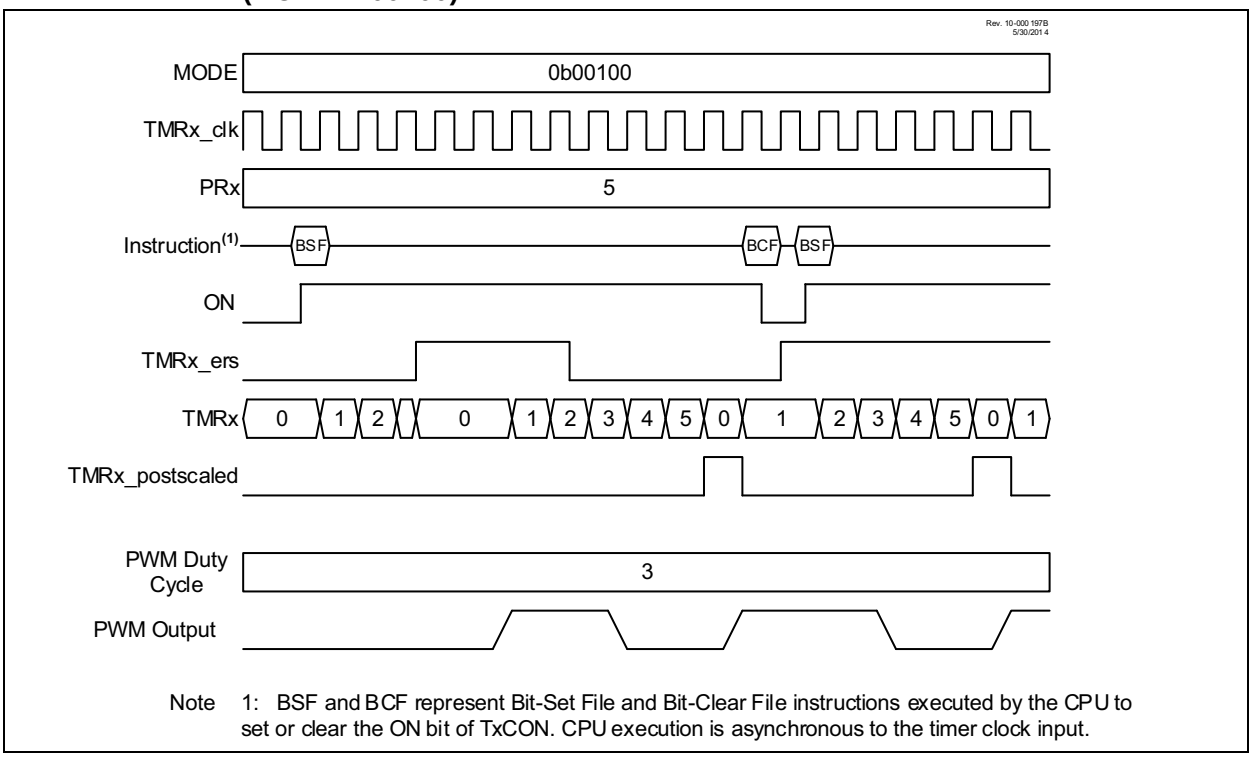
### 23.6.3 EDGE-TRIGGERED HARDWARE LIMIT MODE

In Hardware Limit mode the timer can be reset by the TMRx_ers external signal before the timer reaches the period count. Three types of Resets are possible:

- Reset on rising or falling edge (MODE<4:0>= 00011)
- Reset on rising edge (MODE<4:0> = 00100)
- Reset on falling edge (MODE<4:0> = 00101)

When the timer is used in conjunction with the CCP in PWM mode then an early Reset shortens the period and restarts the PWM pulse after a two clock delay. Refer to Figure 23-6.

**FIGURE 23-6: EDGE-TRIGGERED HARDWARE LIMIT MODE TIMING DIAGRAM (MODE = 00100)**




Note 1: BSF and BCF represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

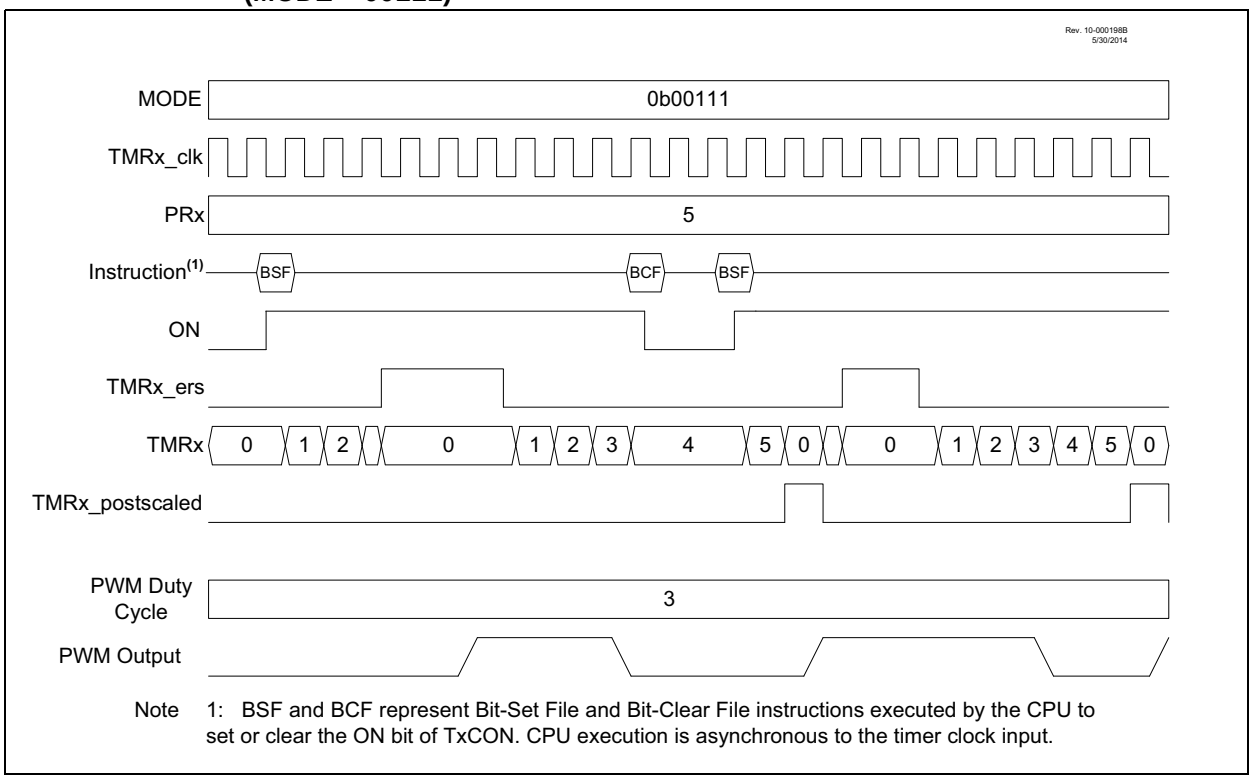
### 23.6.4 LEVEL-TRIGGERED HARDWARE LIMIT MODE

In the Level-Triggered Hardware Limit Timer modes the counter is reset by high or low levels of the external signal TMRx_ers, as shown in Figure 23-7. Selecting MODE<4:0> = 00110 will cause the timer to reset on a low level external signal. Selecting MODE<4:0> = 00111 will cause the timer to reset on a high level external signal. In the example, the counter is reset while TMRx_ers = 1. ON is controlled by BSF and BCF instructions. When ON = 0 the external signal is ignored.

When the CCP uses the timer as the PWM time base then the PWM output will be set high when the timer starts counting and then set low only when the timer count matches the CCPRx value. The timer is reset when either the timer count matches the PRx value or two clock periods after the external Reset signal goes true and stays true.

The timer starts counting, and the PWM output is set high, on either the clock following the PRx match or two clocks after the external Reset signal relinquishes the Reset. The PWM output will remain high until the timer counts up to match the CCPRx pulse width value. If the external Reset signal goes true while the PWM output is high then the PWM output will remain high until the Reset signal is released allowing the timer to count up to match the CCPRx value.

**FIGURE 23-7: LEVEL-TRIGGERED HARDWARE LIMIT MODE TIMING DIAGRAM (MODE = 00111)**




Note 1: BSF and BCF represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

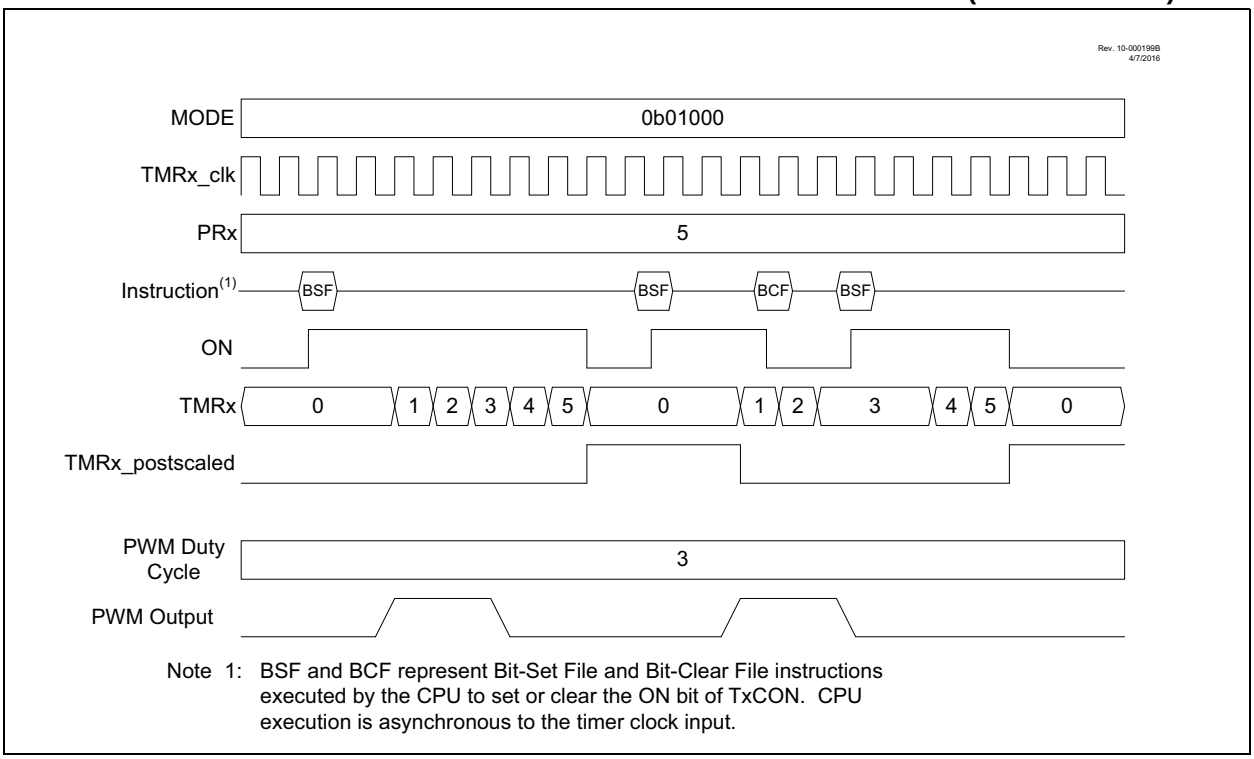
 

### 23.6.5 SOFTWARE START ONE-SHOT MODE

In One-Shot mode the timer resets and the ON bit is cleared when the timer value matches the PRx period value. The ON bit must be set by software to start another timer cycle. Setting MODE<4:0> = 01000 selects One-Shot mode which is illustrated in Figure 23-8. In the example, ON is controlled by BSF and BCF instructions. In the first case, a BSF instruction sets ON and the counter runs to completion and clears ON. In the second case, a BSF instruction starts the cycle, BCF/BSF instructions turn the counter off and on during the cycle, and then it runs to completion.

When One-Shot mode is used in conjunction with the CCP PWM operation the PWM pulse drive starts concurrent with setting the ON bit. Clearing the ON bit while the PWM drive is active will extend the PWM drive. The PWM drive will terminate when the timer value matches the CCPRx pulse width value. The PWM drive will remain off until software sets the ON bit to start another cycle. If software clears the ON bit after the CCPRx match but before the PRx match then the PWM drive will be extended by the length of time the ON bit remains cleared. Another timing cycle can only be initiated by setting the ON bit after it has been cleared by a PRx period count match.

**FIGURE 23-8: SOFTWARE START ONE-SHOT MODE TIMING DIAGRAM (MODE = 01000)**




Note 1: BSF and BCF represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

### 23.6.6 EDGE-TRIGGERED ONE-SHOT MODE

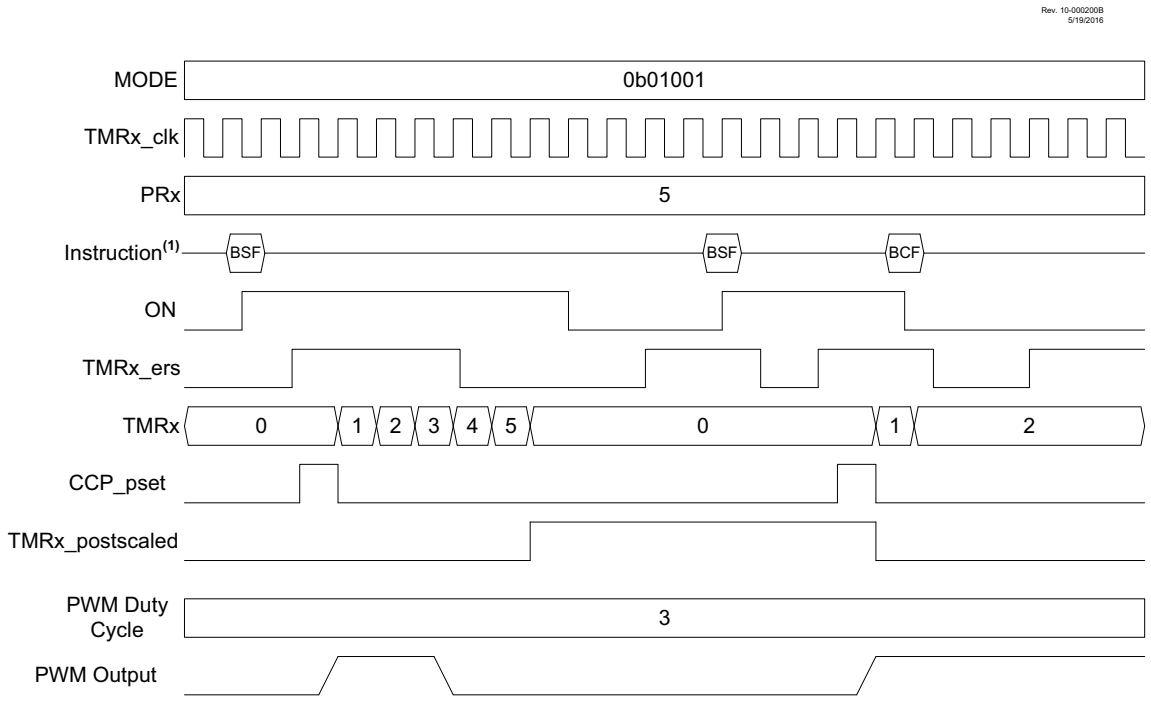
The Edge-Triggered One-Shot modes start the timer on an edge from the external signal input, after the ON bit is set, and clear the ON bit when the timer matches the PRx period value. The following edges will start the timer:

- Rising edge (MODE<4:0> = 01001)
- Falling edge (MODE<4:0> = 01010)
- Rising or Falling edge (MODE<4:0> = 01011)

If the timer is halted by clearing the ON bit then another TMRx_ers edge is required after the ON bit is set to resume counting. Figure 23-9 illustrates operation in the rising edge One-Shot mode.

When Edge-Triggered One-Shot mode is used in conjunction with the CCP then the edge-trigger will activate the PWM drive and the PWM drive will deactivate when the timer matches the CCPRx pulse width value and stay deactivated when the timer halts at the PRx period count match.

**FIGURE 23-9: EDGE-TRIGGERED ONE-SHOT MODE TIMING DIAGRAM (MODE = 01001)**




Note 1: BSF and BCF represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

### 23.6.7 EDGE-TRIGGERED HARDWARE LIMIT ONE-SHOT MODE

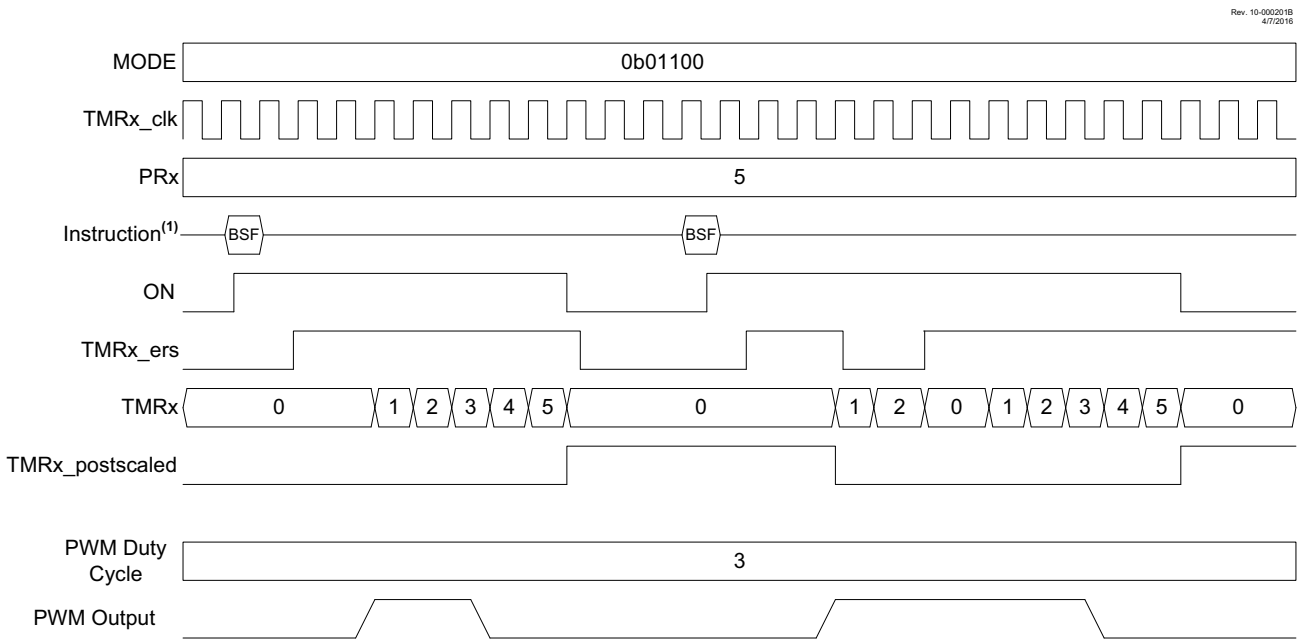
In Edge-Triggered Hardware Limit One-Shot modes the timer starts on the first external signal edge after the ON bit is set and resets on all subsequent edges. Only the first edge after the ON bit is set is needed to start the timer. The counter will resume counting automatically two clocks after all subsequent external Reset edges. Edge triggers are as follows:

- Rising edge start and Reset (MODE<4:0> = 01100)
- Falling edge start and Reset (MODE<4:0> = 01101)

The timer resets and clears the ON bit when the timer value matches the PRx period value. External signal edges will have no effect until after software sets the ON bit. Figure 23-10 illustrates the rising edge hardware limit one-shot operation.

When this mode is used in conjunction with the CCP then the first starting edge trigger, and all subsequent Reset edges, will activate the PWM drive. The PWM drive will deactivate when the timer matches the CCPRx pulse-width value and stay deactivated until the timer halts at the PRx period match unless an external signal edge resets the timer before the match occurs.

**FIGURE 23-10: EDGE-TRIGGERED HARDWARE LIMIT ONE-SHOT MODE TIMING DIAGRAM (MODE = 01100)**




Note 1: BSF and BCF represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

### 23.6.8 LEVEL RESET, EDGE-TRIGGERED HARDWARE LIMIT ONE-SHOT MODES

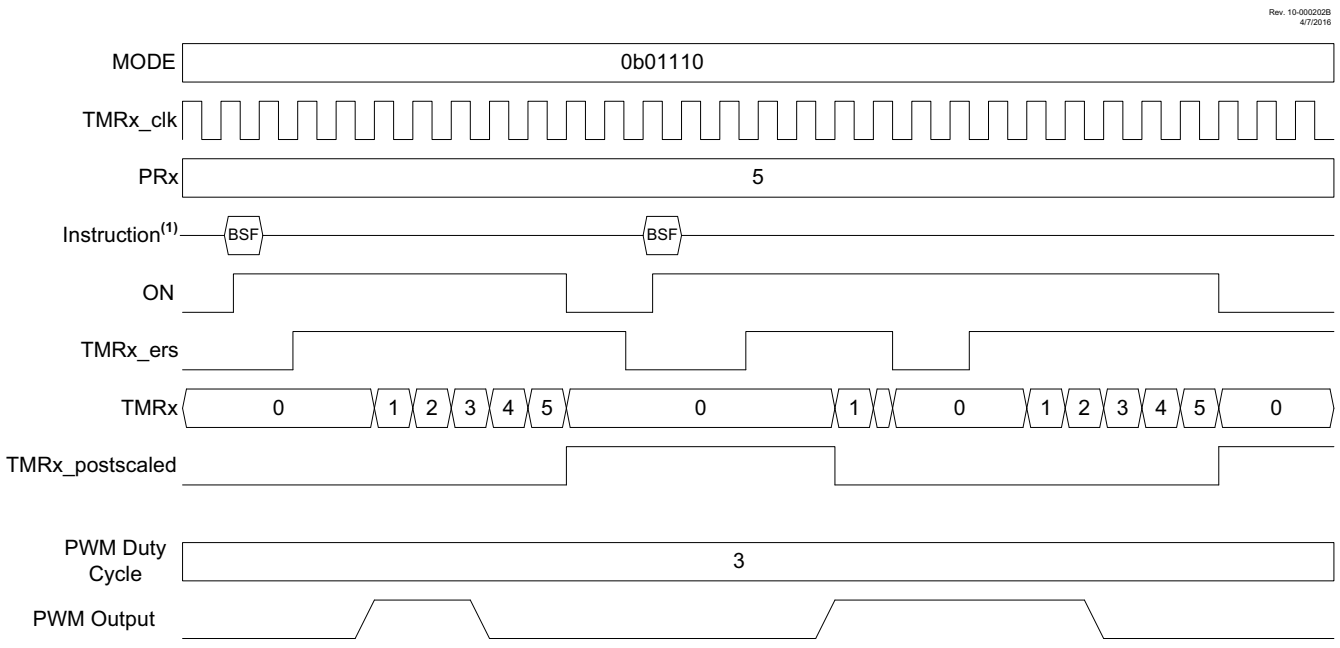
In Level -Triggered One-Shot mode the timer count is reset on the external signal level and starts counting on the rising/falling edge of the transition from Reset level to the active level while the ON bit is set. Reset levels are selected as follows:

- Low Reset level (MODE<4:0> = 01110)
- High Reset level (MODE<4:0> = 01111)

When the timer count matches the PRx period count, the timer is reset and the ON bit is cleared. When the ON bit is cleared by either a PRx match or by software control a new external signal edge is required after the ON bit is set to start the counter.

When Level-Triggered Reset One-Shot mode is used in conjunction with the CCP PWM operation the PWM drive goes active with the external signal edge that starts the timer. The PWM drive goes inactive when the timer count equals the CCPRx pulse width count. The PWM drive does not go active when the timer count clears at the PRx period count match.

**FIGURE 23-11: LOW LEVEL RESET, EDGE-TRIGGERED HARDWARE LIMIT ONE-SHOT MODE TIMING DIAGRAM (MODE = 01110)**




Note 1: BSF and BCF represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

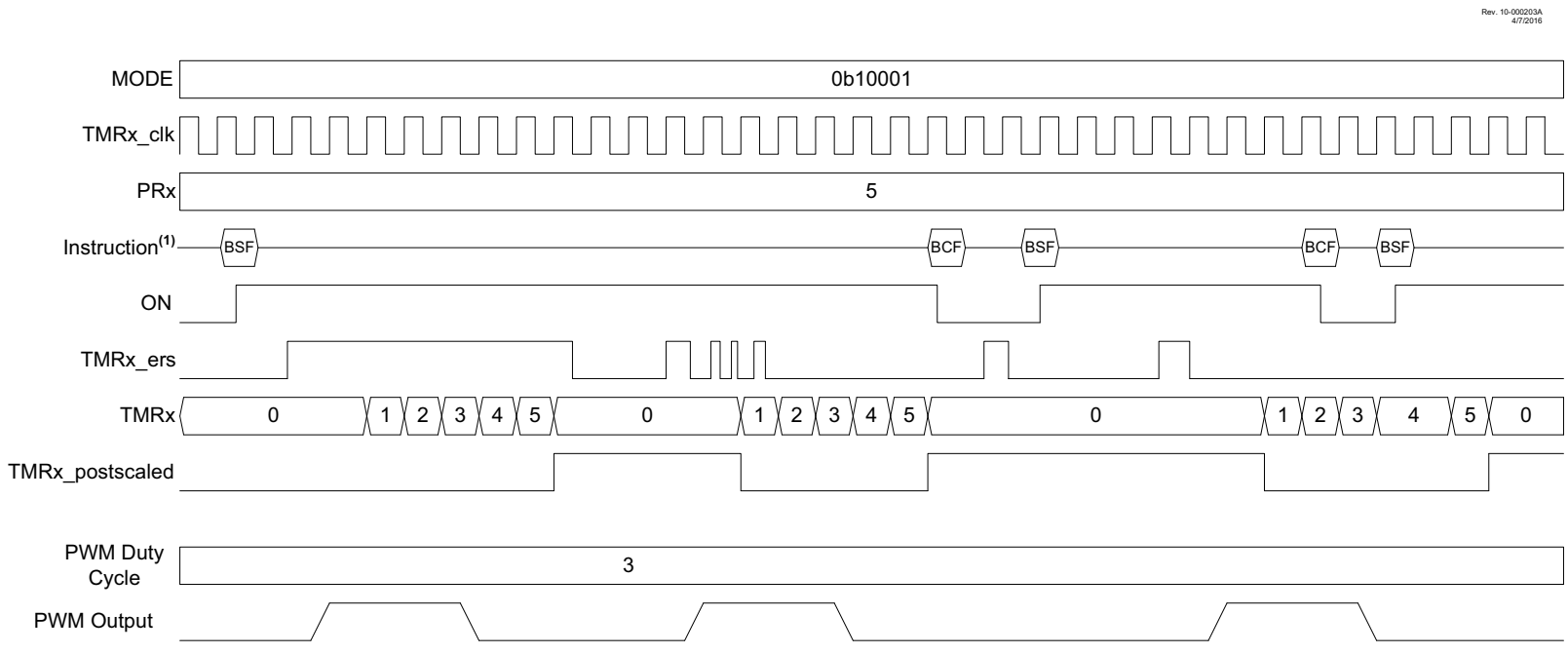
### 23.6.9 EDGE-TRIGGERED MONOSTABLE MODES

The Edge-Triggered Monostable modes start the timer on an edge from the external Reset signal input, after the ON bit is set, and stop incrementing the timer when the timer matches the PRx period value. The following edges will start the timer:

- Rising edge (MODE<4:0> = 10001)
- Falling edge (MODE<4:0> = 10010)
- Rising or Falling edge (MODE<4:0> = 10011)

When an Edge-Triggered Monostable mode is used in conjunction with the CCP PWM operation the PWM drive goes active with the external Reset signal edge that starts the timer, but will not go active when the timer matches the PRx value. While the timer is incrementing, additional edges on the external Reset signal will not affect the CCP PWM.

**FIGURE 23-12: RISING EDGE-TRIGGERED MONOSTABLE MODE TIMING DIAGRAM (MODE = 10001)**




Note 1: BSF and BCF represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

### 23.6.10 LEVEL-TRIGGERED HARDWARE LIMIT ONE-SHOT MODES

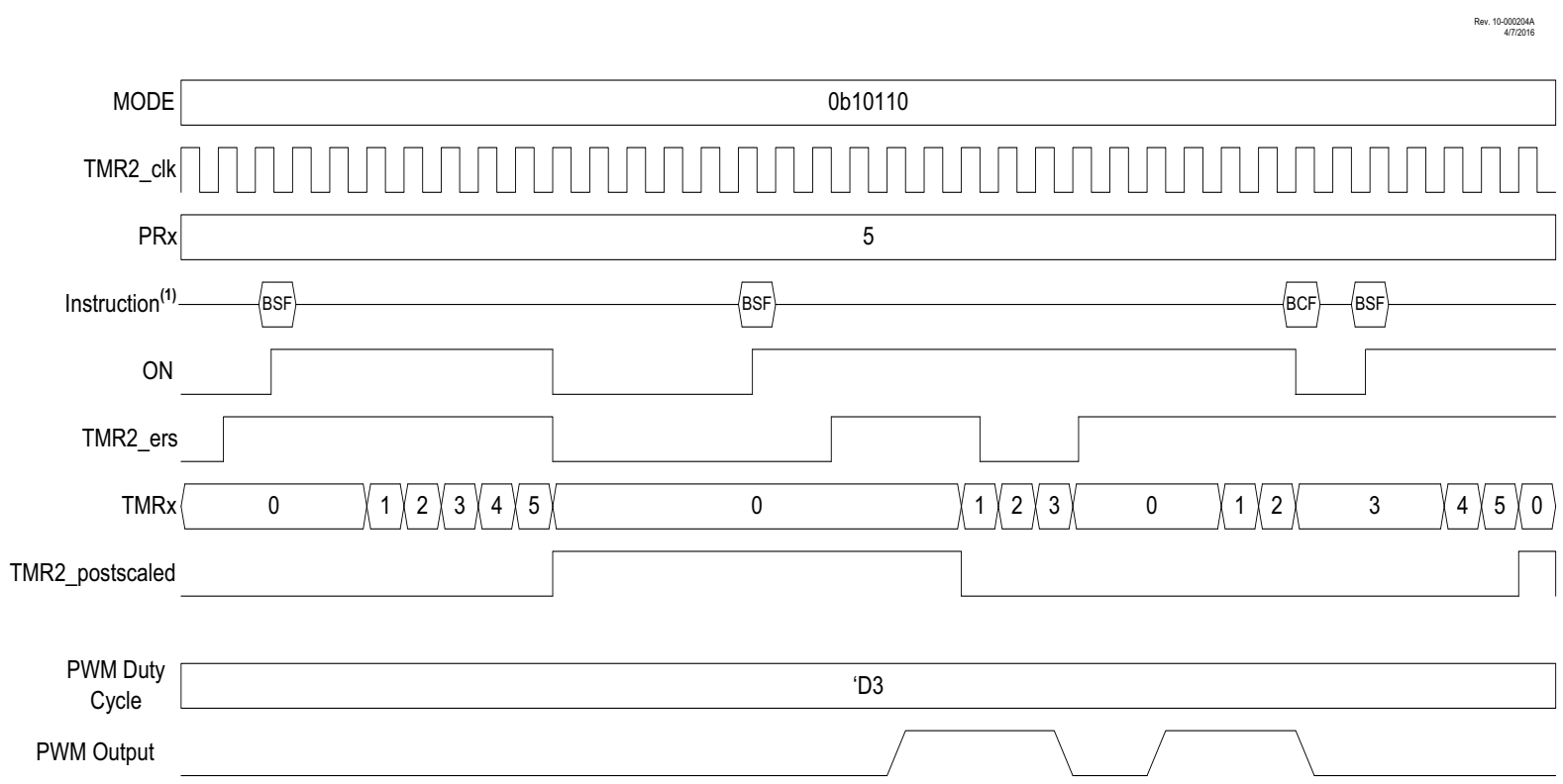
The Level-Triggered Hardware Limit One-Shot modes hold the timer in Reset on an external Reset level and start counting when both the ON bit is set and the external signal is not at the Reset level. If one of either the external signal is not in Reset or the ON bit is set then the other signal being set/made active will start the timer. Reset levels are selected as follows:

- Low Reset level (MODE<4:0> = `10110`)
- High Reset level (MODE<4:0> = `10111`)

When the timer count matches the PRx period count, the timer is reset and the ON bit is cleared. When the ON bit is cleared by either a PRx match or by software control the timer will stay in Reset until both the ON bit is set and the external signal is not at the Reset level.

When Level-Triggered Hardware Limit One-Shot modes are used in conjunction with the CCP PWM operation the PWM drive goes active with either the external signal edge or the setting of the ON bit, whichever of the two starts the timer.

**FIGURE 23-13: LEVEL-TRIGGERED HARDWARE LIMIT ONE-SHOT MODE TIMING DIAGRAM (MODE = 10110)**




Note 1: BSF and BCF represent Bit-Set File and Bit-Clear File instructions executed by the CPU to set or clear the ON bit of TxCON. CPU execution is asynchronous to the timer clock input.

## 23.7 PR2 Period Register

The PR2 period register (T2PR) is double-buffered. Software reads and writes the PR2 register. However, the timer uses a buffered PR2 register for operation. Software does not have direct access to the buffered PR2 register. The contents of the PR2 register are transferred to the buffer by any of the following events:

- A write to the TMR2 register
- A write to the TMR2CON register
- When TMR2 = PR2 buffer and the prescaler rolls over
- An external Reset event

## 23.8 Timer2 Operation During Sleep

When PSYNC = 1, Timer2 cannot be operated while the processor is in Sleep mode. The contents of the TMR2 and T2PR registers will remain unchanged while processor is in Sleep mode.

When PSYNC = 0, Timer2 will operate in Sleep as long as the clock source selected is also still running. Selecting the LFINTOSC, MFINTOSC, or HFINTOSC oscillator as the timer clock source will keep the selected oscillator running during Sleep.

## 23.9 Register Definitions: Timer2/4/6/8 Control

Long bit name prefixes for the Timer2/4/6/8 peripherals are shown in Table 23-2. Refer to **Section 1.1.2.2 "Long Bit Names"** for more information

**TABLE 23-2:**

| Peripheral | Bit Name Prefix |
|---|---|
| Timer2 | T2 |
| Timer4 | T4 |
| Timer6 | T6 |
| Timer8 | T8 |

**REGISTER 23-1: TxCLKCON: TIMERx CLOCK SELECTION REGISTER**

| U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | CS<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 | |

bit 7-4 **Unimplemented:** Read as '0'

bit 3-0 **CS<3:0>:** Timerx Clock Selection bits
See Table 23-3.

**TABLE 23-3: TIMERx CLOCK SOURCES**

| CS<3:0> | Timer2 | Timer4 | Timer6 | Timer8 |
|---|---|---|---|---|
| 1100-1111 | Reserved | Reserved | Reserved | Reserved |
| 1011 | LC4_out | LC4_out | LC4_out | LC4_out |
| 1010 | LC3_out | LC3_out | LC3_out | LC3_out |
| 1001 | LC2_out | LC2_out | LC2_out | LC2_out |
| 1000 | LC1_out | LC1_out | LC1_out | LC1_out |
| 0111 | ZCD_out | ZCD_out | ZCD_out | ZCD_out |
| 0110 | SOSC | SOSC | SOSC | SOSC |
| 0101 | MFINTOSC | MFINTOSC | MFINTOSC | MFINTOSC |
| 0100 | LFINTOSC | LFINTOSC | LFINTOSC | LFINTOSC |
| 0011 | HFINTOSC | HFINTOSC | HFINTOSC | HFINTOSC |
| 0010 | Fosc | Fosc | Fosc | Fosc |
| 0001 | Fosc/4 | Fosc/4 | Fosc/4 | Fosc/4 |
| 0000 | Pin selected by T2INPPS | Pin selected by T4INPPS | Pin selected by T6INPPS | Pin selected by T8INPPS |

**REGISTER 23-2: TxCON: TIMERx CONTROL REGISTER**

| R/W/HC-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 |
|---|---|---|---|---|---|---|---|
| ON(1) | CKPS<2:0> | | | OUTPS<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 | HC = Bit is cleared by hardware |

bit 7 **ON:** Timerx On bit
- 1 = Timerx is on
- 0 = Timerx is off: all counters and state machines are reset

bit 6-4 **CKPS<2:0>:** Timer2-type Clock Prescale Select bits
- 111 = 1:128 Prescaler
- 110 = 1:64 Prescaler
- 101 = 1:32 Prescaler
- 100 = 1:16 Prescaler
- 011 = 1:8 Prescaler
- 010 = 1:4 Prescaler
- 001 = 1:2 Prescaler
- 000 = 1:1 Prescaler

bit 3-0 **OUTPS<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

**Note 1:** In certain modes, the ON bit will be auto-cleared by hardware. See **Section 23.6 "Operation Examples"**.

**REGISTER 23-3: TxHLT: TIMERx HARDWARE LIMIT 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 |
|---|---|---|---|---|---|---|---|
| PSYNC(1, 2) | CKPOL(3) | CKSYNC(4, 5) | MODE<4:0>(6, 7) | | | | |
| 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 **PSYNC:** Timerx Prescaler Synchronization Enable bit(1, 2)
- 1 = TMRx Prescaler Output is synchronized to Fosc/4
- 0 = TMRx Prescaler Output is not synchronized to Fosc/4

bit 6 **CKPOL:** Timerx Clock Polarity Selection bit(3)
- 1 = Falling edge of input clock clocks timer/prescaler
- 0 = Rising edge of input clock clocks timer/prescaler

bit 5 **CKSYNC:** Timerx Clock Synchronization Enable bit(4, 5)
- 1 = ON register bit is synchronized to TMR2_clk input
- 0 = ON register bit is not synchronized to TMR2_clk input

bit 4-0 **MODE<4:0>:** Timerx Control Mode Selection bits(6, 7)
See Table 23-1.

**Note 1:** Setting this bit ensures that reading TMRx will return a valid value.
**2:** When this bit is '1', Timer2 cannot operate in Sleep mode.
**3:** CKPOL should not be changed while ON = 1.
**4:** Setting this bit ensures glitch-free operation when the ON is enabled or disabled.
**5:** When this bit is set then the timer operation will be delayed by two TMRx input clocks after the ON bit is set.
**6:** Unless otherwise indicated, all modes start upon ON = 1 and stop upon ON = 0 (stops occur without affecting the value of TMRx).
**7:** When TMRx = PRx, the next clock clears TMRx, regardless of the operating mode.

**REGISTER 23-4: TxRST: TIMERx EXTERNAL RESET SIGNAL SELECTION REGISTER**

| 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 |
|---|---|---|---|---|---|---|---|
| — | — | — | RSEL<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 **RSEL<4:0>:** TimerX External Reset Signal Source Selection bits
See Table 23-4.

**TABLE 23-4: EXTERNAL RESET SOURCES**

| RSEL<4:0> | Timer2 | Timer4 | Timer6 | Timer8 |
|---|---|---|---|---|
| 11111 | Reserved | Reserved | Reserved | Reserved |
| 11110 | Reserved | Reserved | Reserved | Reserved |
| 11101 | LC4_out | LC4_out | LC4_out | LC4_out |
| 11100 | LC3_out | LC3_out | LC3_out | LC3_out |
| 11011 | LC2_out | LC2_out | LC2_out | LC2_out |
| 11010 | LC1_out | LC1_out | LC1_out | LC1_out |
| 11001 | ZCD_out | ZCD_out | ZCD_out | ZCD_out |
| 11000 | Reserved | Reserved | Reserved | Reserved |
| 10111 | Reserved | Reserved | Reserved | Reserved |
| 10110 | sync_C6OUT | sync_C6OUT | sync_C6OUT | sync_C6OUT |
| 10101 | sync_C5OUT | sync_C5OUT | sync_C5OUT | sync_C5OUT |
| 10100 | sync_C4OUT | sync_C4OUT | sync_C4OUT | sync_C4OUT |
| 10011 | sync_C3OUT | sync_C3OUT | sync_C3OUT | sync_C3OUT |
| 10010 | sync_C2OUT | sync_C2OUT | sync_C2OUT | sync_C2OUT |
| 10001 | sync_C1OUT | sync_C1OUT | sync_C1OUT | sync_C1OUT |
| 10000 | Reserved | Reserved | Reserved | Reserved |
| 01111 | PWM11_out | PWM11_out | PWM11_out | PWM11_out |
| 01110 | PWM6_out | PWM6_out | PWM6_out | PWM6_out |
| 01101 | PWM5_out | PWM5_out | PWM5_out | PWM5_out |
| 01100 | Reserved | Reserved | Reserved | Reserved |
| 01011 | PWM9_out | PWM9_out | PWM9_out | PWM9_out |
| 01010 | PWM4_out | PWM4_out | PWM4_out | PWM4_out |
| 01001 | PWM3_out | PWM3_out | PWM3_out | PWM3_out |
| 01000 | Reserved | Reserved | Reserved | Reserved |
| 00111 | CCP7_out | CCP7_out | CCP7_out | CCP7_out |
| 00110 | CCP2_out | CCP2_out | CCP2_out | CCP2_out |
| 00101 | CCP1_out | CCP1_out | CCP1_out | CCP1_out |
| 00100 | TMR8_postscaled | TMR8_postscaled | TMR8_postscaled | Reserved |
| 00011 | TMR6_postscaled | TMR6_postscaled | Reserved | TMR6_postscaled |
| 00010 | TMR4_postscaled | Reserved | TMR4_postscaled | TMR4_postscaled |
| 00001 | Reserved | TMR2_postscaled | TMR2_postscaled | TMR2_postscaled |
| 00000 | Pin selected byT2INPPS | Pin selected by T4INPPS | Pin selected by T6INPPS | Pin selected by T6INPPS |

**TABLE 23-5: SUMMARY OF REGISTERS ASSOCIATED WITH TIMER2**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| CCP1CON | EN | — | OUT | FMT | MODE<3:0> | | | | 281 |
| CCP2CON | EN | — | OUT | FMT | MODE<3:0> | | | | 281 |
| CCP7CON | EN | — | OUT | FMT | MODE<3:0> | | | | 281 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 108 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 109 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 115 |
| T2PR | Timer2 Module Period Register | | | | | | | | 249* |
| TMR2 | Holding Register for the 8-bit TMR2 Register | | | | | | | | 249* |
| T2CON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | | 269 |
| T2CLKCON | — | — | — | — | CS<3:0> | | | | 268 |
| T2RST | — | — | — | RSEL<4:0> | | | | | 271 |
| T2HLT | PSYNC | CKPOL | CKSYNC | MODE<4:0> | | | | | 270 |
| T4PR | Timer4 Module Period Register | | | | | | | | 249* |
| TMR4 | Holding Register for the 8-bit TMR4 Register | | | | | | | | 249* |
| T4CON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | | 269 |
| T4CLKCON | — | — | — | — | CS<3:0> | | | | 268 |
| T4RST | — | — | — | RSEL<4:0> | | | | | 271 |
| T4HLT | PSYNC | CKPOL | CKSYNC | MODE<4:0> | | | | | 270 |
| T6PR | Timer6 Module Period Register | | | | | | | | 249* |
| TMR6 | Holding Register for the 8-bit TMR6 Register | | | | | | | | 249* |
| T6CON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | | 269 |
| T6CLKCON | — | — | — | — | CS<3:0> | | | | 268 |
| T6RST | — | — | — | RSEL<4:0> | | | | | 271 |
| T6HLT | PSYNC | CKPOL | CKSYNC | MODE<4:0> | | | | | 270 |
| T8PR | Timer6 Module Period Register | | | | | | | | 249* |
| TMR8 | Holding Register for the 8-bit TMR6 Register | | | | | | | | 249* |
| T8CON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | | 269 |
| T8CLKCON | — | — | — | — | CS<3:0> | | | | 268 |
| T8RST | — | — | — | RSEL<4:0> | | | | | 271 |
| T8HLT | PSYNC | CKPOL | CKSYNC | MODE<4:0> | | | | | 270 |

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

* Page provides register information.

## 24.0 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.

**TABLE 24-1: AVAILABLE CCP MODULES**

| Device | CCP1 | CCP2 | CCP7 |
|---|---|---|---|
| PIC16(L)F1773 | • | • | • |
| PIC16(L)F1776 | • | • | • |

**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 a CCPx module. 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.

### 24.1 Capture Mode

The Capture mode function described in this section is available and identical for all CCP modules.

Capture mode makes use of the 16-bit Timer1 resource. When an event occurs on the CCPx input, 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 MODE<3:0> bits of the CCPxCON register:

- Every edge (rising or falling)
- Every falling edge
- Every rising edge
- Every 4th rising edge
- Every 16th rising edge

The CCPx capture input signal is configured by the CTS bits of the CCPxCAP register with the following options:

- CCPx pin
- Comparator 1 output (C1_OUT_sync)
- Comparator 2 output (C2_OUT_sync)
- Comparator 7 output (C7_OUT_sync)
- LC2_output
- LC3_output
- Interrupt-on-change interrupt trigger (IOC_interrupt)

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, the old captured value is overwritten by the new captured value.

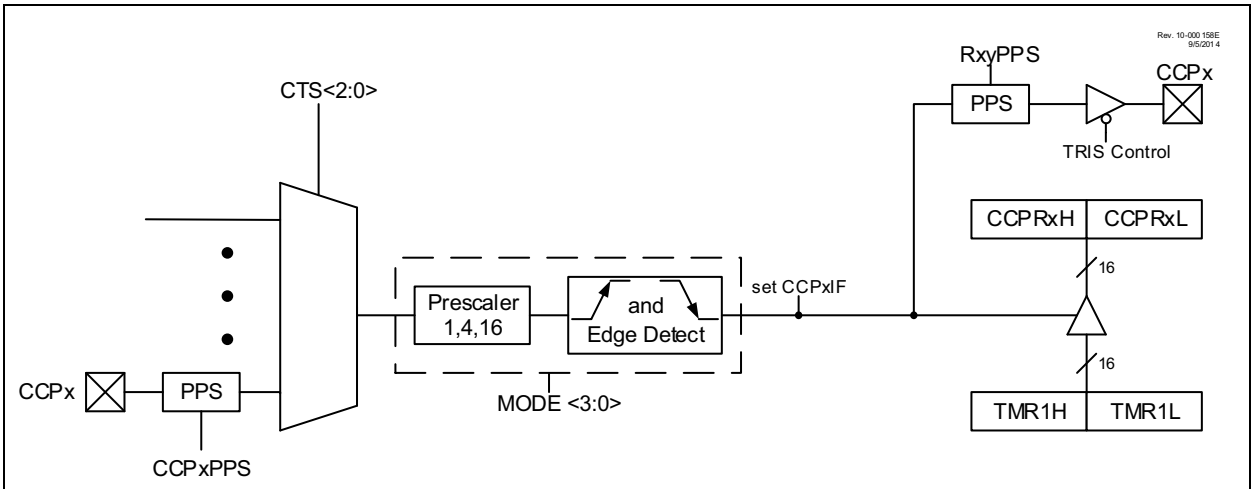
Figure 24-1 shows a simplified diagram of the capture operation.

#### 24.1.1 CCP PIN CONFIGURATION

In Capture mode, select the interrupt source using the CTS bits of the CCPxCAP register. If the CCPx pin is chosen, it should be configured as an input by setting the associated TRIS control bit.

**Note:** If the CCPx pin is configured as an output, a write to the port can cause a capture condition.

**FIGURE 24-1: CAPTURE MODE OPERATION BLOCK DIAGRAM**




**Note:** Capture sources. See Register 24-6.

### 24.1.2 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 22.0 "Timer1/3/5 Module with Gate Control"** for more information on configuring Timer1.

### 24.1.3 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 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. |

### 24.1.4 CCP PRESCALER

There are four prescaler settings specified by the MODE<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 EN bit of the CCPxCON register before changing the prescaler.

### 24.1.5 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.

### 24.1.6 ALTERNATE PIN LOCATIONS

This module incorporates I/O pins that can be moved to other locations with the use of the PPS controls. See **Section 12.0 "Peripheral Pin Select (PPS) Module"** for more details.

### 24.1.7 CAPTURE OUTPUT

Whenever a capture occurs, the output of the CCP will go high for a period equal to one system clock period (1/FOSC). This output is available as an input signal to the following peripherals:

- ADC Trigger
- COG
- PRG
- DSM
- CLC
- Op Amp override
- Timer2/4/6/8 Reset
- Any device pins

In addition, the CCP output can be output to any pin with that pin's PPS control.

## 24.2 Compare Mode

The Compare mode function described in this section is available and identical for all CCP modules.

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
- Pulse the CCPx output
- Generate a Software Interrupt
- Auto-conversion Trigger

The action on the pin is based on the value of the MODE<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 24-2 shows a simplified diagram of the compare operation.

### 24.2.1 AUTO-CONVERSION TRIGGER

When Auto-Conversion Trigger mode is chosen (CCPxM<3:0> = 1011), the CCPx module does the following:

- Resets Timer1
- Starts an ADC conversion if ADC is enabled

The CCPx module does not assert control of the CCPx pin in this mode.

The Auto-conversion 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 Auto-conversion Trigger output starts an ADC conversion (if the ADC module is enabled). This allows the CCPRxH:CCPRxL register pair to effectively provide a 16-bit programmable period register for Timer1.

Refer to **Section 16.2.5 "Auto-Conversion Trigger"** for more information.

| | |
|---|---|
| **Note 1:** | The Auto-conversion 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 Auto-conversion Trigger and the clock edge that generates the Timer1 Reset, will preclude the Reset from occurring. |

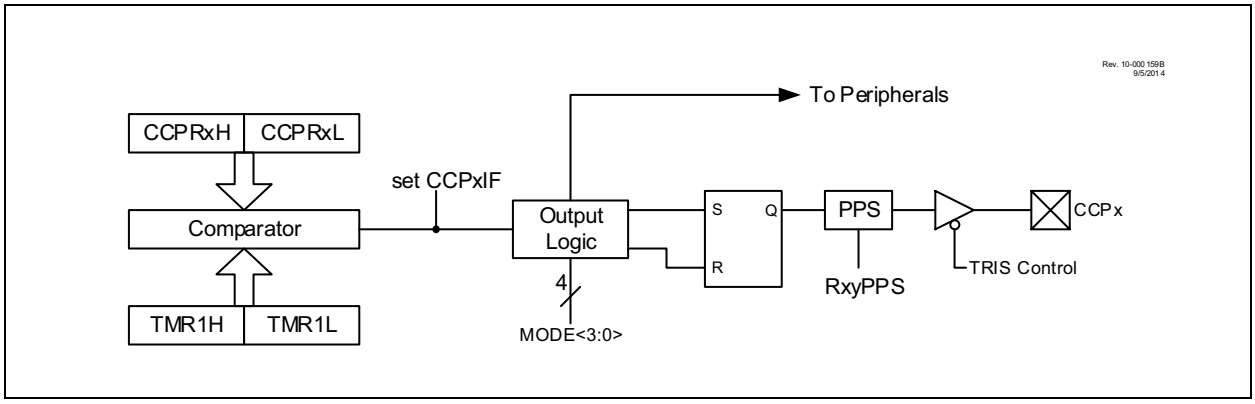
### 24.2.2 CCPx PIN CONFIGURATION

The user must configure the CCPx pin as an output by clearing the associated TRIS bit.

The CCPx pin function can be moved to alternate pins using the PPS controls. See **Section 12.0 "Peripheral Pin Select (PPS) Module"** for more detail.

| | |
|---|---|
| **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. |

**FIGURE 24-2: COMPARE MODE OPERATION BLOCK DIAGRAM**



### 24.2.3 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 22.0 "Timer1/3/5 Module with Gate Control"** for more information on configuring Timer1.

| | |
|---|---|
| **Note:** | Clocking Timer1 from the system clock (FOSC) should not be used in Compare mode. In order for Compare 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. |

### 24.2.4 SOFTWARE INTERRUPT MODE

When Generate Software Interrupt mode is chosen (MODE<3:0> = 1010), the CCPx module does not assert control of the CCPx pin (see the CCPxCON register).

### 24.2.5 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.

### 24.2.6 ALTERNATE PIN LOCATIONS

This module incorporates I/O pins that can be moved to other locations with the use of the PPS controls. See **Section 12.0 "Peripheral Pin Select (PPS) Module"** for more detail.

### 24.2.7 CAPTURE OUTPUT

When in Compare mode, the CCP will provide an output upon the 16-bit value of the CCPRxH:CCPRxL register pair matching the TMR1H:TMR1L register pair. The compare output depends on which Compare mode the CCP is configured as. If the MODE bits of CCPxCON register are equal to '1011' or '1010', the CCP module will output high, while TMR1 is equal to the CCPRxH:CCPRxL register pair. This means that the pulse width is determined by the TMR1 prescaler. If the MODE bits of CCPxCON are equal to '0001' or '0010', the output will toggle upon a match, going from '0' to '1' or vice-versa. If the MODE bits of CCPxCON are equal to '1001', the output is cleared on a match, and if the MODE bits are equal to '1000', the output is set on a match. This output is available to the following peripherals:

- ADC Trigger
- COG
- PRG
- DSM
- CLC
- Op Amp override
- Timer2/4/6/8 Reset
- Any device pins

## 24.3 PWM Overview

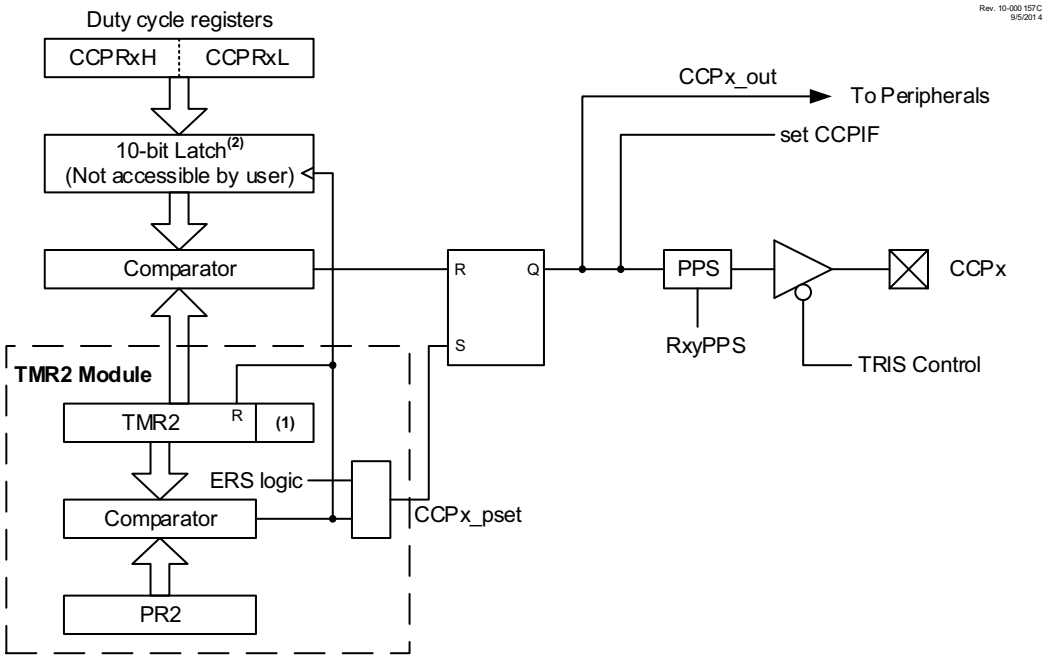
Pulse-Width Modulation (PWM) is a scheme that provides power to a load by switching quickly between fully on and fully off states. The PWM signal resembles a square wave where the high portion of the signal is considered the on state and the low portion of the signal is considered the off state. The 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 24-3 shows a typical waveform of the PWM signal.

**FIGURE 24-3: SIMPLIFIED PWM BLOCK DIAGRAM**




Notes: 1. 8-bit timer is concatenated with two bits generated by Fosc or two bits of the internal prescaler to create 10-bit time-base.
2. The alignment of the 10 bits from the CCPR register is determined by the FMT bit.

### 24.3.1 STANDARD PWM OPERATION

The standard PWM function described in this section is available and identical for all CCP modules.

The standard PWM mode generates a Pulse-Width Modulation (PWM) signal on the CCPx pin with up to 10 bits of resolution. The period, duty cycle, and resolution are controlled by the following registers:

- T2PR/T4PR/T6PR/T8PR registers
- T2CON/T4CON/T6CON/T8CON registers
- CCPRxH:CCPRxL register pair

Figure 24-3 shows a simplified block diagram of 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. |

### 24.3.2 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. Select the timer associated with the PWM by setting the CCPTMRS register.
3. Load the associated T2PR/T4PR/T6PR/T8PR 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 CCPRxH:CCPRxL register pair with the PWM duty cycle value.
6. Configure and start the timer selected in step 2:
   - Clear the timer interrupt flag bit of the PIRx register. See Note below.
   - Configure the CKPS bits of the TxCON register with the Timer prescale value.
   - Enable the Timer by setting the ON bit of the TxCON register.
7. Enable PWM output pin:
   - Wait until the Timer overflows and the timer interrupt 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. |

## 24.4 CCP/PWM Clock Selection

The PIC16(L)F1773/6 allows each individual CCP and PWM module to select the timer source that controls the module. Each module has an independent selection.

As there are up to four 8-bit timers with auto-reload (Timer2/4/6/8). The PWM mode on the CCP and PWM modules can use any of these timers.

The CCPTMRS register is used to select which timer is used.

### 24.4.1 USING THE TMR2/4/6/8 WITH THE CCP MODULE

This device has a new version of the TMR2 module that has many new modes, which allow for greater customization and control of the PWM signals than older parts. Refer to **Section 23.6 "Operation Examples"** for examples of PWM signal generation using the different modes of Timer2. The CCP operation requires that the timer used as the PWM time base has the FOSC/4 clock source selected.

### 24.4.2 PWM PERIOD

The PWM period is specified by the T2PR/T4PR/T6PR/T8PR register of Timer2/4/6/8. The PWM period can be calculated using the formula of Equation 24-1.

**EQUATION 24-1: PWM PERIOD**

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

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

When TMR2/4/6/8 is equal to its respective T2PR/T4PR/T6PR/T8PR register, the following three events occur on the next increment cycle:

- TMR2/4/6/8 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 the CCPRxH:CCPRxL pair into the internal 10-bit latch.

| **Note:** | The Timer postscaler (see Figure 24-1) is not used in the determination of the PWM frequency. |
|---|---|

### 24.4.3 PWM DUTY CYCLE

The PWM duty cycle is specified by writing a 10-bit value to two registers: the CCPRxH:CCPRxL register pair. Where the particular bits go is determined by the FMT bit of the CCPxCON register. If FMT = 0, the two Most Significant bits of the duty cycle value should be written to bits <1:0> of the CCPRxH register and the remaining eight bits to the CCPRxL register. If FMT = 1, the Least Significant two bits of the duty cycle should be written to bits <7:6> of the CCPRxL register and the Most Significant eight bits to the CCPRxH register. This is illustrated in Figure 24-4. These bits can be written at any time. The duty cycle value is not latched into the internal latch until after the period completes (i.e., a match between T2PR/T4PR/T6PR/T8PR and TMR2/4/6/8 registers occurs).

Equation 24-2 is used to calculate the PWM pulse width. Equation 24-3 is used to calculate the PWM duty cycle ratio.

**EQUATION 24-2: PULSE WIDTH**

$$Pulse\ Width = CCPRxH:CCPRxL \bullet TOSC \bullet (TMR2\ Prescale\ Value)$$

**EQUATION 24-3: DUTY CYCLE RATIO**

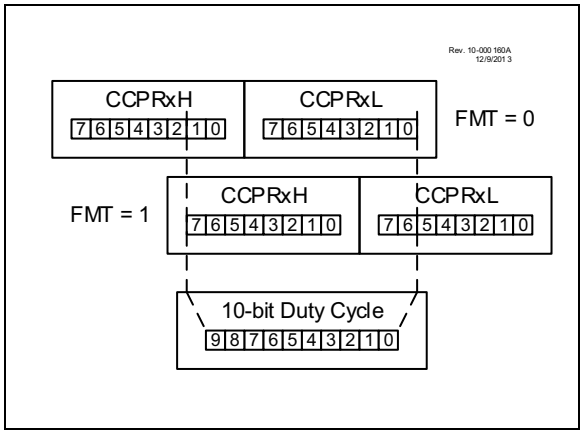
$$Duty\ Cycle\ Ratio = \frac{(CCPRxH:CCPRxL)}{4(PRx + 1)}$$

The PWM duty cycle registers are double buffered for glitchless PWM operation.

The 8-bit timer TMR2/4/6/8 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 Timer2/4/6/8 prescaler is set to 1:1.

When the 10-bit time base matches the internal buffer register, then the CCPx pin is cleared (see Figure 24-3).

**FIGURE 24-4: CCPx DUTY CYCLE ALIGNMENT**



### 24.4.4 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 T2PR/T4PR/T6PR/T8PR is 255. The resolution is a function of the T2PR/T4PR/T6PR/T8PR register value as shown by Equation 24-4.

**EQUATION 24-4: PWM RESOLUTION**

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

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

**TABLE 24-2: 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 |
| T2PR Value | 0xFF | 0xFF | 0xFF | 0x3F | 0x1F | 0x17 |
| Maximum Resolution (bits) | 10 | 10 | 10 | 8 | 7 | 6 |

**TABLE 24-3: 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 |
| T2PR Value | 0x65 | 0x65 | 0x65 | 0x19 | 0x0C | 0x09 |
| Maximum Resolution (bits) | 8 | 8 | 8 | 6 | 5 | 5 |

### 24.4.5 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.

### 24.4.6 EFFECTS OF RESET

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

### 24.4.7 PWM OUTPUT

The output of the CCP in PWM mode is the PWM signal generated by the module and described above. This output is available to the following peripherals:

- ADC Trigger
- COG
- PRG
- DSM
- CLC
- Op Amp override
- Timer2/4/6/8 Reset
- Any device pins

## 24.5 Register Definitions: CCP Control

**REGISTER 24-1: CCPxCON: CCPx CONTROL REGISTER**

| R/W-0/0 | U-0 | R-x | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| EN | — | OUT | FMT | MODE<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 **EN:** CCPx Module Enable bit
1 = CCPx is enabled
0 = CCPx is disabled

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

bit 5 **OUT:** CCPx Output Data bit (read-only)

bit 4 **FMT:** CCPW (Pulse-Width) Alignment bit
If MODE = PWM Mode
1 = Left-aligned format, CCPRxH <7> is the MSB of the PWM duty cycle
0 = Right-aligned format, CCPRxL<0> is the LSB of the PWM duty cycle

bit 3-0 **MODE<3:0>:** CCPx Mode Selection bits
11xx = PWM mode

1011 = Compare mode: Pulse output, clear TMR1
1010 = Compare mode: Pulse output (0 - 1 - 0)
1001 = Compare mode: clear output on compare match. Output is set upon selection of this mode.
1000 = Compare mode: set output on compare match. Output is set upon selection of this mode.

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 = Capture mode: every rising or falling edge
0010 = Compare mode: toggle output on match
0001 = Compare mode: Toggle output and clear TMR1 on match
0000 = Capture/Compare/PWM off (resets CCPx module) (reserved for backwards compatibility)

**REGISTER 24-2: CCPTMRS1: PWM TIMER SELECTION 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 |
|---|---|---|---|---|---|---|---|
| — | — | C7TSEL<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 **Unimplemented:** Read as ‘0’

bit 5-4 **C7TSEL<1:0>:** CCP7 (PWM7) Timer Selection bits
- 11 = CCP7 is based off Timer8 in PWM mode
- 10 = CCP7 is based off Timer6 in PWM mode
- 01 = CCP7 is based off Timer4 in PWM mode
- 00 = CCP7 is based off Timer2 in PWM mode

bit 3-2 **C2TSEL<1:0>:** CCP2 (PWM2) Timer Selection bits
- 11 = CCP2 is based off Timer8 in PWM mode
- 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 (PWM1) Timer Selection bits
- 11 = CCP1 is based off Timer8 in PWM mode
- 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 24-3: CCPTMRS2: PWM TIMER SELECTION CONTROL REGISTER 2**

| 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 |
|---|---|---|---|---|---|---|---|
| — | — | P9TSEL<1:0> | | P4TSEL<1:0> | | P3TSEL<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 **Unimplemented:** Read as ‘0’

bit 5-4 **P9TSEL<1:0>:** PWM9 Timer Selection bits
- 11 = PWM9 is based off Timer8 in PWM mode
- 10 = PWM9 is based off Timer6 in PWM mode
- 01 = PWM9 is based off Timer4 in PWM mode
- 00 = PWM9 is based off Timer2 in PWM mode

bit 3-2 **P4TSEL<1:0>:** PWM4 Timer Selection bits
- 11 = PWM4 is based off Timer8 in PWM mode
- 10 = PWM4 is based off Timer6 in PWM mode
- 01 = PWM4 is based off Timer4 in PWM mode
- 00 = PWM4 is based off Timer2 in PWM mode

bit 1-0 **P3TSEL<1:0>:** PWM3 Timer Selection bits
- 11 = PWM3 is based off Timer8 in PWM mode
- 10 = PWM3 is based off Timer6 in PWM mode
- 01 = PWM3 is based off Timer4 in PWM mode
- 00 = PWM3 is based off Timer2 in PWM mode

**REGISTER 24-4: CCPRxL: CCPx 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 |
|---|---|---|---|---|---|---|---|
| CCPR<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 Reset |
| '1' = Bit is set | '0' = Bit is cleared | |

bit 7-0

MODE = Capture Mode
**CCPRxL<7:0>:** LSB of captured TMR1 value
MODE = Compare Mode
**CCPRxL<7:0>:** LSB compared to TMR1 value
MODE = PWM Mode && FMT = 0
**CCPRxL<7:0>:** CCPW<7:0> – Pulse width Least Significant eight bits
MODE = PWM Mode && FMT = 1
**CCPRxL<7:6>:** CCPW<1:0> – Pulse width Least Significant two bits
**CCPRxL<5:0>:** Not used

**REGISTER 24-5: CCPRxH: CCPx HIGH 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 |
|---|---|---|---|---|---|---|---|
| CCPR<15:8> | | | | | | | |
| bit 7 | | | | | | | bit 0 |

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

bit 7-0

MODE = Capture Mode
**CCPRxH<7:0>:** MSB of captured TMR1 value
MODE = Compare Mode
**CCPRxH<7:0>:** MSB compared to TMR1 value
MODE = PWM Mode && FMT = 0
**CCPRxH<7:2>:** Not used
**CCPRxH<1:0>:** CCPW<9:8> – Pulse width Most Significant two bits
MODE = PWM Mode && FMT = 1
**CCPRxH<7:0>:** CCPW<9:2> – Pulse width Most Significant eight bits

**REGISTER 24-6: CCPxCAP: CCPx CAPTURE INPUT SELECTION REGISTER**

| U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | CTS<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 7-4 **Unimplemented:** Read as ‘0’

bit 3-0 **CTS<3:0>:** Capture Trigger Input Selection bits

1101 = IOC_event
1100 = LC4_output
1011 = LC3_output
1010 = LC2_output
1001 = LC1_output
1000 = Reserved
0111 = Reserved
0110 = C6_sync_out
0101 = C5_sync_out
0100 = C4_sync_out
0011 = C3_sync_out
0010 = C2_sync_out
0001 = C1_sync_out
0000 = Pin selected with the CCPxPPS register

## 24.6 CCP/PWM Clock Selection

This device allows each individual CCP and PWM module to select the timer source that controls the module. Each module has an independent selection.

As there are four 8-bit timers with auto-reload (Timer2, Timer4, Timer6 and Timer8). The PWM mode on the CCP and 10-bit PWM modules can use any of these timers.

The CCPTMRS register is used to select which timer is used.

## 24.7 Register Definitions: CCP/PWM Timers Control

**REGISTER 24-7: CCPTMRS: 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 |
|---|---|---|---|---|---|---|---|
| P4TSEL<1:0> | | P3TSEL<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 **P4TSEL<1:0>:** PWM4 Timer Selection bits
11 = Reserved
10 = PWM4 is based off Timer6
01 = PWM4 is based off Timer4
00 = PWM4 is based off Timer2

bit 5-4 **P3TSEL<1:0>:** PWM3 Timer Selection bits
11 = Reserved
10 = PWM3 is based off Timer6
01 = PWM3 is based off Timer4
00 = PWM3 is based off Timer2

bit 3-2 **C2TSEL<1:0>:** CCP2 (PWM2) Timer Selection bits
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 (PWM1) Timer Selection bits
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

**TABLE 24-4: 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 |
|---|---|---|---|---|---|---|---|---|---|
| CCPxCAP | — | — | — | — | CTS<3:0> | | | | 285 |
| CCPxCON | EN | — | OUT | FMT | MODE<3:0> | | | | 281 |
| CCPRxL | Capture/Compare/PWM Register x (LSB) | | | | | | | | 284 |
| CCPRxH | Capture/Compare/PWM Register x (MSB) | | | | | | | | 284 |
| CCPTMRS1 | — | — | C7TSEL<1:0> | | C2TSEL<1:0> | | C1TSEL<1:0> | | 283 |
| CCPTMRS2 | — | — | P9TSEL<1:0> | | P4TSEL<1:0> | | P3TSEL<1:0> | | 283 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 108 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 109 |
| PIE2 | OSFIE | C2IE | C1IE | COG1IE | BCL1IE | C4IE | C3IE | CCP2IE | 110 |
| PIE5 | — | CCP7IE | — | COG3IE | — | — | C6IE | C5IE | 113 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 115 |
| PIR2 | OSFIF | C2IF | C1IF | COG1IF | BCL1IF | C4IF | C3IF | CCP2IF | 116 |
| PIR5 | — | CCP7IF | — | COG3IF | — | — | C6IF | C5IF | 119 |
| T2PR | Timer2 Period Register | | | | | | | | 249* |
| T2CON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | | 269 |
| TMR2 | Timer2 Module Register | | | | | | | | 249 |
| T4PR | Timer4 Period Register | | | | | | | | 249* |
| T4CON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | | 269 |
| TMR4 | Timer4 Module Register | | | | | | | | 249 |
| T6PR | Timer6 Period Register | | | | | | | | 249* |
| T6CON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | | 269 |
| TMR6 | Timer6 Module Register | | | | | | | | 249 |
| T8PR | Timer8 Period Register | | | | | | | | 249* |
| T8CON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | | 269 |
| TMR8 | Timer8 Module Register | | | | | | | | 249 |

**Legend:** — = Unimplemented location, read as '0'. Shaded cells are not used by the PWM.
* Page provides register information.

## 25.0 10-BIT PULSE-WIDTH MODULATION (PWM) MODULE

The 10-bit PWM module generates a Pulse-Width Modulated signal determined by the duty cycle, period, and resolution that are configured by the following registers:

- T2PR
- T2CON
- PWMxDCH
- PWMxDCL
- PWMxCON

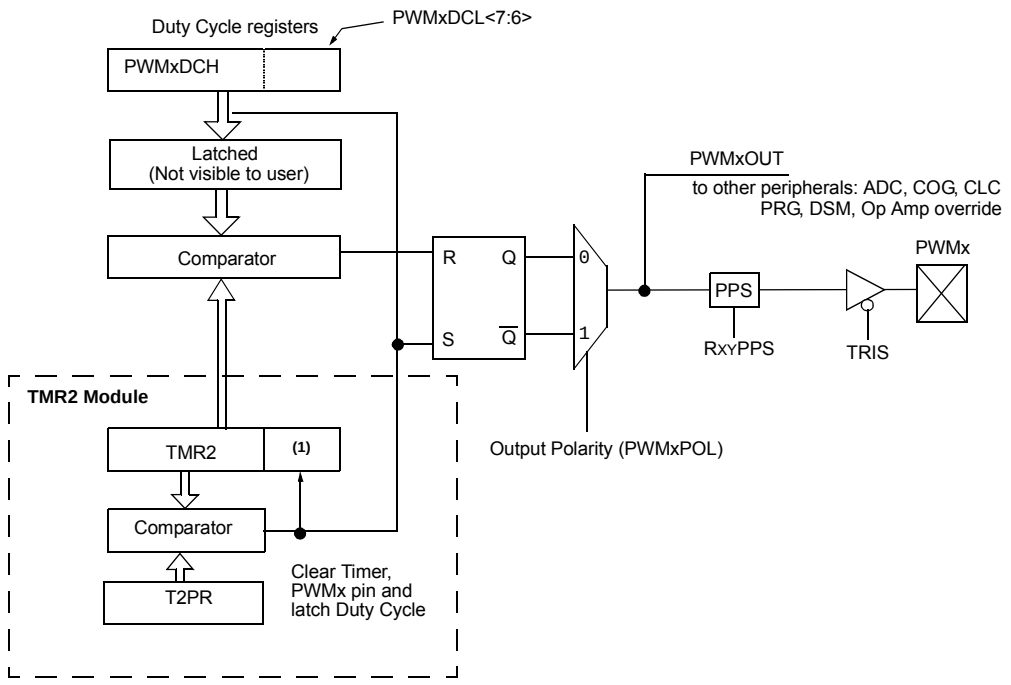
Figure 25-1 shows a simplified block diagram of PWM operation.

Figure 25-2 shows a typical waveform of the PWM signal.

**TABLE 25-1: AVAILABLE 10-BIT PWM MODULES**

| Device | PWM3 | PWM4 | PWM9 |
|---|---|---|---|
| PIC16(L)F1773 | • | • | • |
| PIC16(L)F1776 | • | • | • |

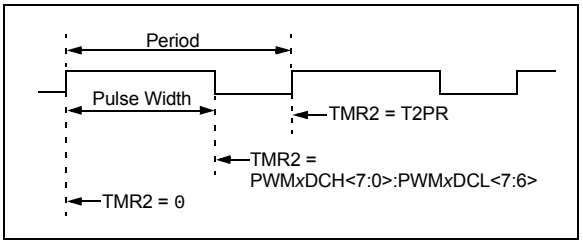
**FIGURE 25-1: SIMPLIFIED PWM BLOCK DIAGRAM**




**Note 1:** 8-bit timer is concatenated with the two Least Significant bits of 1/FOSC adjusted by the Timer2 prescaler to create a 10-bit time base.

For a step-by-step procedure on how to set up this module for PWM operation, refer to **Section 25.1.9 "Setup for PWM Operation using PWMx Output Pins"**.

**FIGURE 25-2: PWM OUTPUT**



## 25.1 PWMx Pin Configuration

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

### 25.1.1 FUNDAMENTAL OPERATION

The PWM module produces a 10-bit resolution output. Timer2 and T2PR set the period of the PWM. The PWMxDCL and PWMxDCH registers configure the duty cycle. The period is common to all PWM modules, whereas the duty cycle is independently controlled.

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

All PWM outputs associated with Timer2 are set when TMR2 is cleared. Each PWMx is cleared when TMR2 is equal to the value specified in the corresponding PWMxDCH (8 MSb) and PWMxDCL<7:6> (2 LSb) registers. When the value is greater than or equal to T2PR, the PWM output is never cleared (100% duty cycle).

| | |
|---|---|
| **Note:** | The PWMxDCH and PWMxDCL registers are double buffered. The buffers are updated when Timer2 matches T2PR. Care should be taken to update both registers before the timer match occurs. |

### 25.1.2 PWM OUTPUT POLARITY

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

### 25.1.3 PWM PERIOD

The PWM period is specified by the T2PR register of Timer2. The PWM period can be calculated using the formula of Equation 25-1.

**EQUATION 25-1: PWM PERIOD**

$$PWM\ Period = [T2PR + 1] \bullet 4 \bullet T_{OSC} \bullet (TMR2\ Prescale\ Value)$$

**Note:** TOSC = 1/FOSC

When TMR2 is equal to T2PR, the following three events occur on the next increment cycle:

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

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

### 25.1.4 PWM DUTY CYCLE

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

Equation 25-2 is used to calculate the PWM pulse width.

Equation 25-3 is used to calculate the PWM duty cycle ratio.

**EQUATION 25-2: PULSE WIDTH**

$$Pulse\ Width = (PWMxDCH:PWMxDCL<7:6>) \bullet T_{OSC} \bullet (TMR2\ Prescale\ Value)$$

**Note:** TOSC = 1/FOSC

**EQUATION 25-3: DUTY CYCLE RATIO**

$$Duty\ Cycle\ Ratio = \frac{(PWMxDCH:PWMxDCL<7:6>)}{4(T2PR + 1)}$$

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

### 25.1.5 PWM RESOLUTION

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

The maximum PWM resolution is ten bits when T2PR is 255. The resolution is a function of the T2PR register value as shown by Equation 25-4.

**EQUATION 25-4: PWM RESOLUTION**

$$Resolution = \frac{log[4(T2PR + 1)]}{log(2)} \ bits$$

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

**TABLE 25-2: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS (FOSC = 20 MHz)**

| PWM Frequency | 0.31 kHz | 4.88 kHz | 19.53 kHz | 78.12 kHz | 156.3 kHz | 208.3 kHz |
|---|---|---|---|---|---|---|
| Timer Prescale | 64 | 4 | 1 | 1 | 1 | 1 |
| T2PR Value | 0xFF | 0xFF | 0xFF | 0x3F | 0x1F | 0x17 |
| Maximum Resolution (bits) | 10 | 10 | 10 | 8 | 7 | 6.6 |

**TABLE 25-3: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS (FOSC = 8 MHz)**

| PWM Frequency | 0.31 kHz | 4.90 kHz | 19.61 kHz | 76.92 kHz | 153.85 kHz | 200.0 kHz |
|---|---|---|---|---|---|---|
| Timer Prescale | 64 | 4 | 1 | 1 | 1 | 1 |
| T2PR Value | 0x65 | 0x65 | 0x65 | 0x19 | 0x0C | 0x09 |
| Maximum Resolution (bits) | 8 | 8 | 8 | 6 | 5 | 5 |

### 25.1.6 OPERATION IN SLEEP MODE

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

### 25.1.7 CHANGES IN SYSTEM CLOCK FREQUENCY

The PWM frequency is derived from the system clock frequency (FOSC). Any changes in the system clock frequency will result in changes to the PWM frequency. Refer to **Section 5.0 "Oscillator Module (with Fail-Safe Clock Monitor)"** for additional details.

### 25.1.8 EFFECTS OF RESET

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

### 25.1.9 SETUP FOR PWM OPERATION USING PWMx OUTPUT PINS

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

1. Disable the PWMx pin output driver(s) by setting the associated TRIS bit(s).
2. Clear the PWMxCON register.
3. Load the T2PR register with the PWM period value.
4. Load the PWMxDCH register and bits <7:6> of the PWMxDCL register with the PWM duty cycle value.
5. Configure and start Timer2:
   - Clear the TMR2IF interrupt flag bit of the PIR1 register. See Note below.
   - Configure the CKPS bits of the T2CON register with the Timer2 prescale value.
   - Enable Timer2 by setting the ON bit of the T2CON register.
6. Enable PWM output pin and wait until Timer2 overflows, TMR2IF bit of the PIR1 register is set. See Note below.
7. Enable the PWMx pin output driver(s) by clearing the associated TRIS bit(s) and setting the desired pin PPS control bits.
8. Configure the PWM module by loading the PWMxCON register with the appropriate values.

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

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

### 25.1.10 SETUP FOR PWM OPERATION TO OTHER DEVICE PERIPHERALS

The following steps should be taken when configuring the module for PWM operation to be used by other device peripherals:

1. Disable the PWMx pin output driver(s) by setting the associated TRIS bit(s).
2. Clear the PWMxCON register.
3. Load the T2PR register with the PWM period value.
4. Load the PWMxDCH register and bits <7:6> of the PWMxDCL register with the PWM duty cycle value.
5. Configure and start Timer2:
   - Clear the TMR2IF interrupt flag bit of the PIR1 register. See Note below.
   - Configure the CKPS bits of the T2CON register with the Timer2 prescale value.
   - Enable Timer2 by setting the ON bit of the T2CON register.
6. Enable PWM output pin:
   - Wait until Timer2 overflows, TMR2IF bit of the PIR1 register is set. See Note below.
7. Configure the PWM module by loading the PWMxCON register with the appropriate values.

**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.

## 25.2 Register Definitions: 10-Bit PWM Control

Long bit name prefixes for the DSM peripherals are shown in Table 25-4. Refer to **Section 1.1.2.2 "Long Bit Names"** for more information

**TABLE 25-4:**

| Peripheral | Bit Name Prefix |
|---|---|
| PWM3 | PWM3 |
| PWM4 | PWM4 |
| PWM9 | PWM9 |

**REGISTER 25-1: PWMxCON: PWM CONTROL REGISTER**

| R/W-0/0 | U-0 | R-0/0 | R/W-0/0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| EN | — | OUT | POL | — | — | — | — |
| 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 **EN:** PWM Module Enable bit
1 = PWM module is enabled
0 = PWM module is disabled

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

bit 5 **OUT:** PWM module output level when bit is read.

bit 4 **POL:** PWMx Output Polarity Select bit
1 = PWM output is active-low
0 = PWM output is active-high

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

**REGISTER 25-2: PWMxDCH: PWM DUTY CYCLE HIGH BITS**

| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
|---|---|---|---|---|---|---|---|
| DC<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 **DC<9:2>:** PWM Duty Cycle Most Significant bits
These bits are the MSbs of the PWM duty cycle. The two LSbs are found in the PWMxDCL Register.

**REGISTER 25-3: PWMxDCL: PWM DUTY CYCLE LOW BITS**

| R/W-x/u | R/W-x/u | U-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| DC<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 **DC<1:0>:** PWM Duty Cycle Least Significant bits
These bits are the LSbs of the PWM duty cycle. The MSbs are found in the PWMxDCH Register.

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

**TABLE 25-5: SUMMARY OF REGISTERS ASSOCIATED WITH 10-BIT PWM**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| CCPTMRS2 | — | — | P9TSEL<1:0> | | P4TSEL<1:0> | | P3TSEL<1:0> | | 281 |
| PWMxCON | EN | — | OUT | POL | — | — | — | — | 292 |
| PWMxDCH | DC<9:2> | | | | | | | | 293 |
| PWMxDCL | DC<1:0> | | — | — | — | — | — | — | 293 |
| RxyPPS | — | — | RxyPPS<5:0> | | | | | | 172 |
| TxCON | ON | CKPS<2:0> | | | OUTPS<3:0> | | | | 269 |
| TxCLKCON | — | — | — | — | CS<3:0> | | | | 268 |
| TxPR | TMRx Period Register | | | | | | | | 249 |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 152 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 157 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 162 |

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

## 26.0 16-BIT PULSE-WIDTH MODULATION (PWM) MODULE

The Pulse-Width Modulation (PWM) module generates a pulse-width modulated signal determined by the phase, duty cycle, period, and offset event counts that are contained in the following registers:

- PWMxPH register
- PWMxDC register
- PWMxPR register
- PWMxOF register

Figure 26-1 shows a simplified block diagram of the PWM operation. Each PWM module has four modes of operation:

- Standard
- Set On Match
- Toggle On Match
- Center Aligned

For a more detailed description of each PWM mode, refer to **Section 26.2 "PWM Modes"**.

Each PWM module has four offset modes:

- Independent Run
- Slave Run with Synchronous Start
- One-Shot Slave with Synchronous Start
- Continuous Run Slave with Synchronous Start and Timer Reset

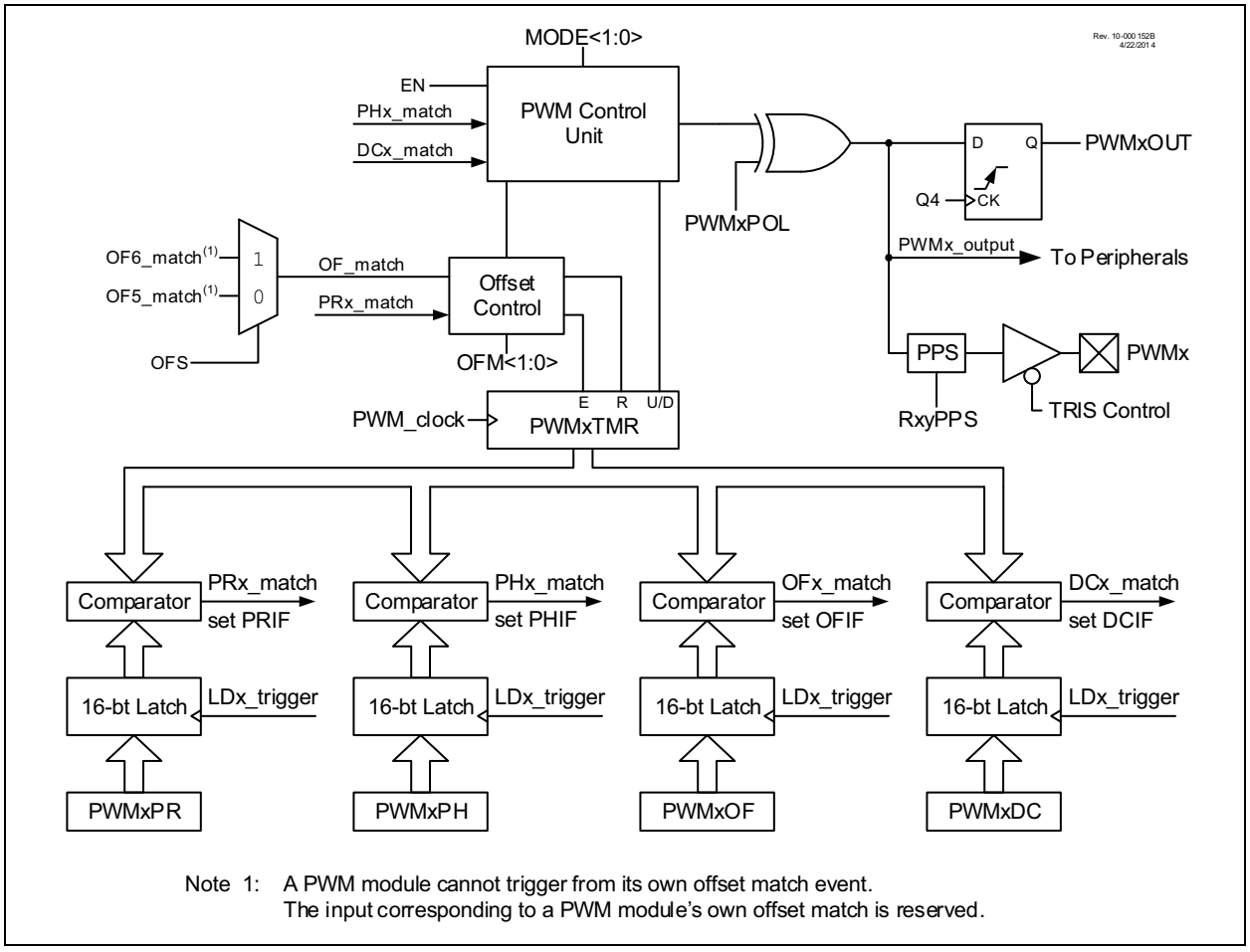
Using the offset modes, each PWM module can offset its waveform relative to any other PWM module in the same device. For a more detailed description of the offset modes refer to **Section 26.3 "Offset Modes"**.

Every PWM module has a configurable reload operation to ensure all event count buffers change at the end of a period, thereby avoiding signal glitches. Figure 26-2 shows a simplified block diagram of the reload operation. For a more detailed description of the reload operation, refer to Section **Section 26.4 "Reload Operation"**.

**TABLE 26-1: AVAILABLE 16-BIT PWM MODULES**

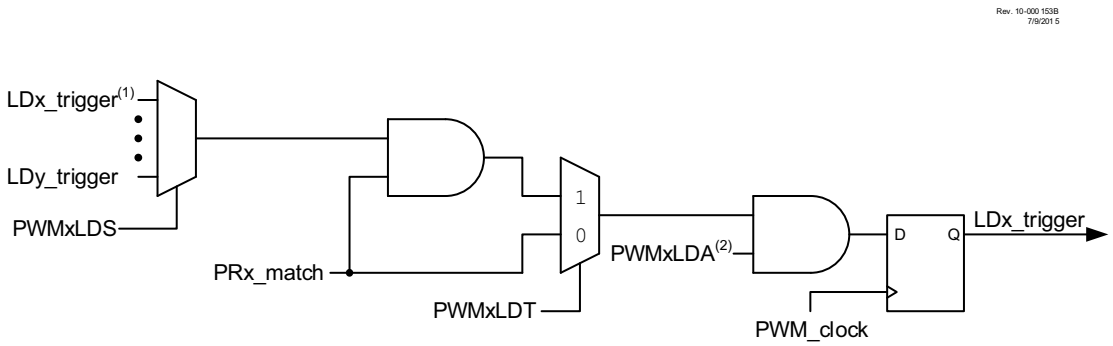
| Device | PWM5 | PWM6 | PWM11 |
|---|---|---|---|
| PIC16(L)F1773 | • | • | • |
| PIC16(L)F1776 | • | • | • |

**FIGURE 26-1: 16-BIT PWM BLOCK DIAGRAM**




Note 1: A PWM module cannot trigger from its own offset match event. The input corresponding to a PWM module's own offset match is reserved.

**FIGURE 26-2: LOAD TRIGGER BLOCK DIAGRAM**




Note 1. The input corresponding to a PWM module's own load trigger is reserved.
2. PWMxLDA is cleared by hardware upon LDx_trigger.

## 26.1 Fundamental Operation

The PWM module produces a 16-bit resolution pulse-width modulated output.

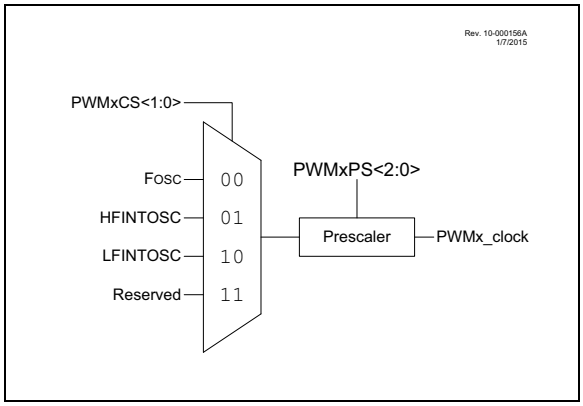
Each PWM module has an independent timer driven by a selection of clock sources determined by the PWMxCLKCON register (Register 26-4). The timer value is compared to event count registers to generate the various events of a the PWM waveform, such as the period and duty cycle. For a block diagram describing the clock sources refer to Figure 26-3.

Each PWM module can be enabled individually using the EN bit of the PWMxCON register, or several PWM modules can be enabled simultaneously using the MPWMxEN bits of the PWMEN register.

The current state of the PWM output can be read using the OUT bit of the PWMxCON register. In some modes this bit can be set and cleared by software giving additional software control over the PWM waveform. This bit is synchronized to FOSC/4 and therefore does not change in real time with respect to the PWM_clock.

| **Note:** | If PWM_clock > FOSC/4, the OUT bit may not accurately represent the output state of the PWM. |
|---|---|

**FIGURE 26-3: PWM CLOCK SOURCE BLOCK DIAGRAM**




### 26.1.1 PWMx PIN CONFIGURATION

This device uses the PPS control circuitry to route peripherals to any device I/O pin. Select the desired pin, or pins, for PWM output with the device pin RxyPPS control registers (Register 12-2).

All PWM outputs are multiplexed with the PORT data latch, so the pins must also be configured as outputs by clearing the associated PORT TRIS bits.

The slew rate feature may be configured to optimize the rate to be used in conjunction with the PWM outputs. High-speed output switching is attained by clearing the associated PORT SLRCON bits.

The PWM outputs can be configured to be open-drain outputs by setting the associated PORT ODCON bits.

### 26.1.2 PWMx Output Polarity

The output polarity is inverted by setting the POL bit of the PWMxCON register. The polarity control affects the PWM output even when the module is not enabled.

## 26.2 PWM Modes

PWM modes are selected with MODE<1:0> bits of the PWMxCON register (Register 26-1).

In all PWM modes an offset match event can also be used to synchronize the PWMxTMR in three offset modes. See **Section26.3 "Offset Modes"** for more information.

### 26.2.1 STANDARD MODE

The Standard mode (MODE = 00) selects a single phase PWM output. The PWM output in this mode is determined by when the period, duty cycle, and phase counts match the PWMxTMR value. The start of the duty cycle occurs on the phase match and the end of the duty cycle occurs on the duty cycle match. The period match resets the timer. The offset match can also be used to synchronize the PWMxTMR in the offset modes. See **Section26.3 "Offset Modes"** for more information.

Equation 26-1 is used to calculate the PWM period in Standard mode.

Equation 26-2 is used to calculate the PWM duty cycle ratio in Standard mode.

**EQUATION 26-1: PWM PERIOD IN STANDARD MODE**

$$Period = \frac{(PWMxPR + 1) \cdot Prescale}{PWM\_clock}$$

**EQUATION 26-2: PWM DUTY CYCLE IN STANDARD MODE**

$$Duty\ Cycle = \frac{(PWMxDC - PWMxPH)}{PWMxPR + 1}$$

A detailed timing diagram for Standard mode is shown in Figure 26-4.

### 26.2.2 SET ON MATCH MODE

The Set On Match mode (MODE = 01) generates an active output when the phase count matches the PWMxTMR value. The output stays active until the OUT bit of the PWMxCON register is cleared or the PWM module is disabled. The duty cycle count has no effect in this mode. The period count only determines the maximum PWMxTMR value above which no phase matches can occur.

The PWMxOUT bit can be used to set or clear the output of the PWM in this mode. Writes to this bit will take place on the next rising edge of the PWM_clock after the bit is written.

A detailed timing diagram for Set On Match is shown in Figure 26-5.

### 26.2.3 TOGGLE ON MATCH MODE

The Toggle On Match mode (MODE = 10) generates a 50% duty cycle PWM with a period twice as long as that computed for the standard PWM mode. Duty cycle count has no effect in this mode. The phase count determines how many PWMxTMR periods after a period event the output will toggle.

Writes to the OUT bit of the PWMxCON register will have no effect in this mode.

A detailed timing diagram for Toggle On Match is shown in Figure 26-6.

### 26.2.4 CENTER ALIGNED MODE

The Center Aligned mode (MODE = 11) generates a PWM waveform that is centered in the period. In this mode the period is two times the PWMxPR count. The PWMxTMR counts up to the period value then counts back down to 0. The duty cycle count determines both the start and end of the active PWM output. The start of the duty cycle occurs at the match event when PWMxTMR is incrementing and the duty cycle ends at the match event when PWMxTMR is decrementing. The incrementing match value is the period count minus the duty cycle count. The decrementing match value is the incrementing match value plus 1.

Equation 26-3 is used to calculate the PWM period in Center Aligned mode.

**EQUATION 26-3: PWM PERIOD IN CENTER ALIGNED MODE**

$$Period = \frac{(PWMxPR + 1) \cdot 2 \cdot Prescale}{PWM\_clock}$$

Equation 26-4 is used to calculate the PWM duty cycle ratio in Center Aligned mode

**EQUATION 26-4: PWM DUTY CYCLE IN CENTER ALIGNED MODE**

$$Duty\ Cycle = \frac{PWMxDC \cdot 2}{(PWMxPR + 1) \cdot 2}$$

Writes to PWMxOUT will have no effect in this mode.

A detailed timing diagram for Center Aligned mode is shown in Figure 26-7.

**FIGURE 26-4: STANDARD PWM MODE TIMING DIAGRAM**

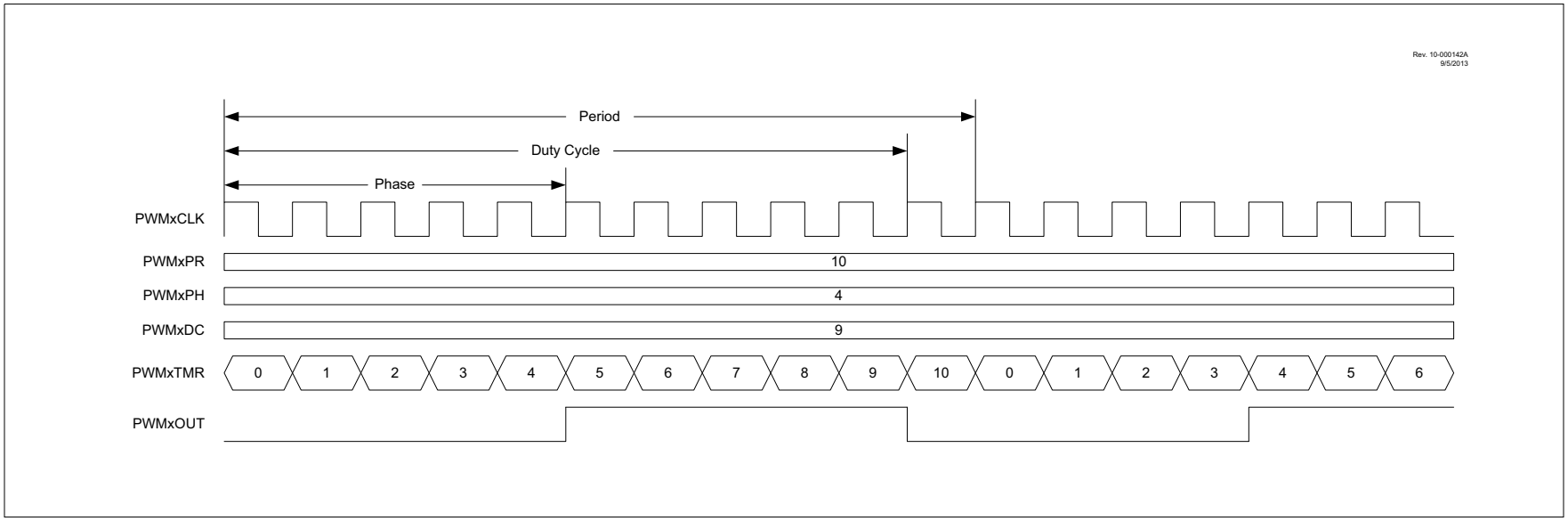



**FIGURE 26-5: SET ON MATCH PWM MODE TIMING DIAGRAM**

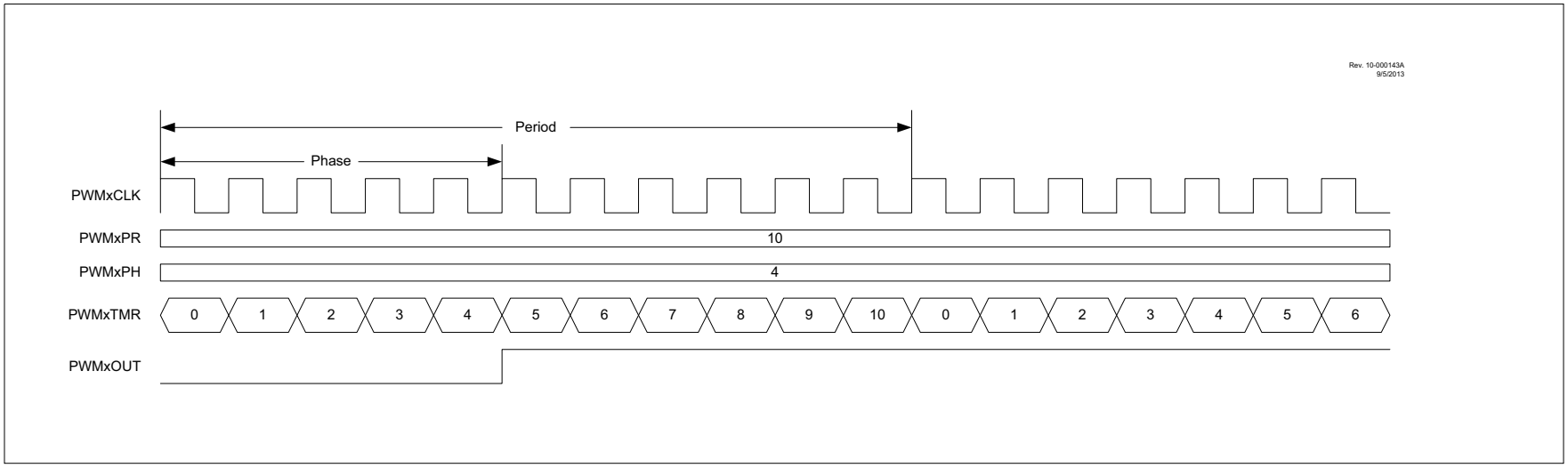


**FIGURE 26-6: TOGGLE ON MATCH PWM MODE TIMING DIAGRAM**

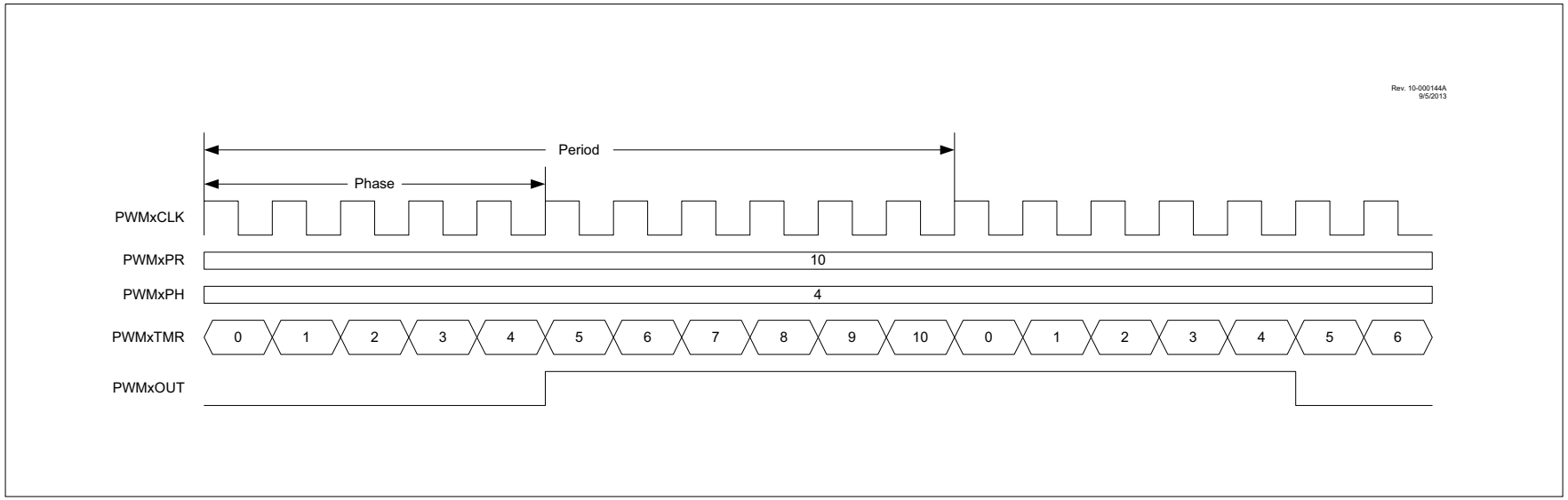
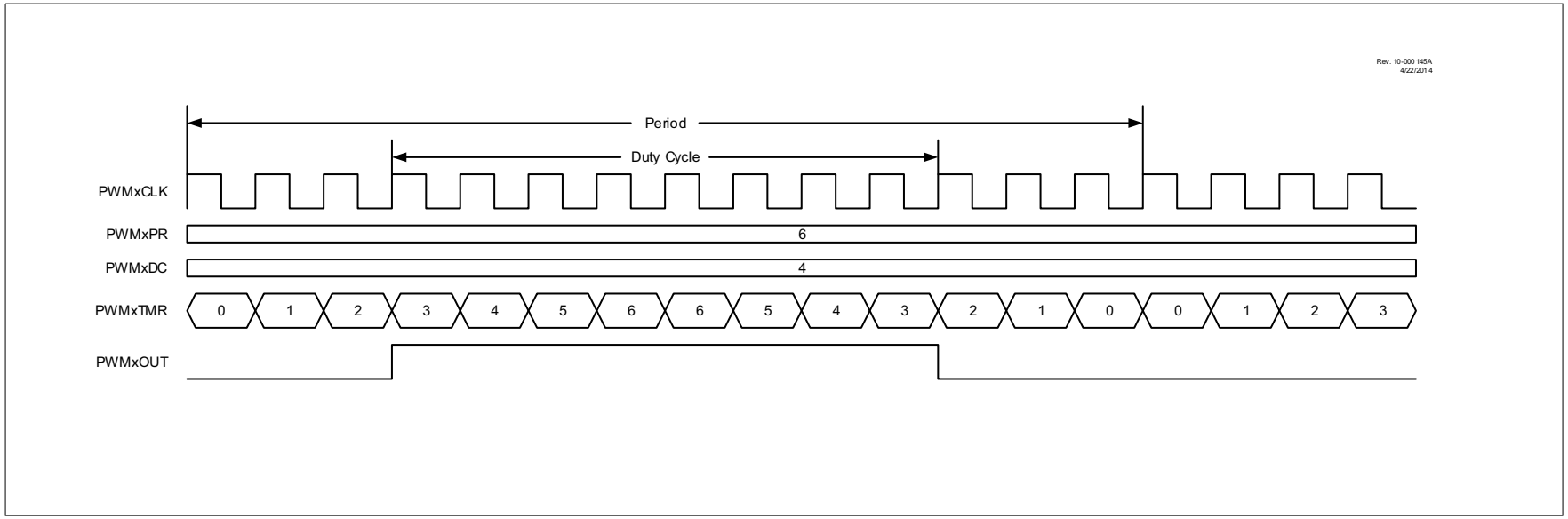



**FIGURE 26-7: CENTER ALIGNED PWM MODE TIMING DIAGRAM**



## 26.3 Offset Modes

The offset modes provide the means to adjust the waveform of a slave PWM module relative to the waveform of a master PWM module in the same device.

### 26.3.1 INDEPENDENT RUN MODE

In Independent Run mode (OFM = 00), the PWM module is unaffected by the other PWM modules in the device. The PWMxTMR associated with the PWM module in this mode starts counting as soon as the EN bit associated with this PWM module is set and continues counting until the EN bit is cleared. Period events reset the PWMxTMR to zero after which the timer continues to count.

A detailed timing diagram of this mode used with Standard PWM mode is shown in Figure 26-8.

### 26.3.2 SLAVE RUN MODE WITH SYNC START

In Slave Run mode with Sync Start (OFM = 01), the slave PWMxTMR waits for the master's OF_match event. When this event occurs, if the EN bit is set, the PWMxTMR begins counting and continues to count until software clears the EN bit. Slave period events reset the PWMxTMR to zero after which the timer continues to count.

A detailed timing diagram of this mode used with Standard PWM mode is shown in Figure 26-9.

### 26.3.3 ONE-SHOT SLAVE MODE WITH SYNC START

In One-Shot Slave mode with Synchronous Start (OFM = 10), the slave PWMxTMR waits until the master's OF_match event. The timer then begins counting, starting from the value that is already in the timer, and continues to count until the period match event. When the period event occurs the timer resets to zero and stops counting. The timer then waits until the next master OF_match event after which it begins counting again to repeat the cycle.

A detailed timing diagram of this mode used with Standard PWM mode is shown in Figure 26-10.

### 26.3.4 CONTINUOUS RUN SLAVE MODE WITH SYNC START AND TIMER RESET

In Continuous Run Slave mode with Synchronous Start and Timer Reset (OFM = 11) the slave PWMxTMR is inhibited from counting after the slave PWM enable is set. The first master OF_match event starts the slave PWMxTMR. Subsequent master OF_match events reset the slave PWMxTMR timer value back to 1 after which the slave PWMxTMR continues to count. The next master OF_match event resets the slave PWMxTMR back to 1 to repeat the cycle. Slave period events that occur before the master's OF_match event will reset the slave PWMxTMR to zero after which the timer will continue to count. Slaves operating in this mode must have a PWMxPH register pair value equal to or greater than 1, otherwise the phase match event will not occur precluding the start of the PWM output duty cycle.

The offset timing will persist if both the master and slave PWMxPR values are the same and the Slave Offset mode is changed to Independent Run mode while the PWM module is operating.

A detailed timing diagram of this mode used in Standard PWM mode is shown in Figure 26-11.

| **Note:** | Unexpected results will occur if the slave PWM_clock is a higher frequency than the master PWM_clock. |
|---|---|

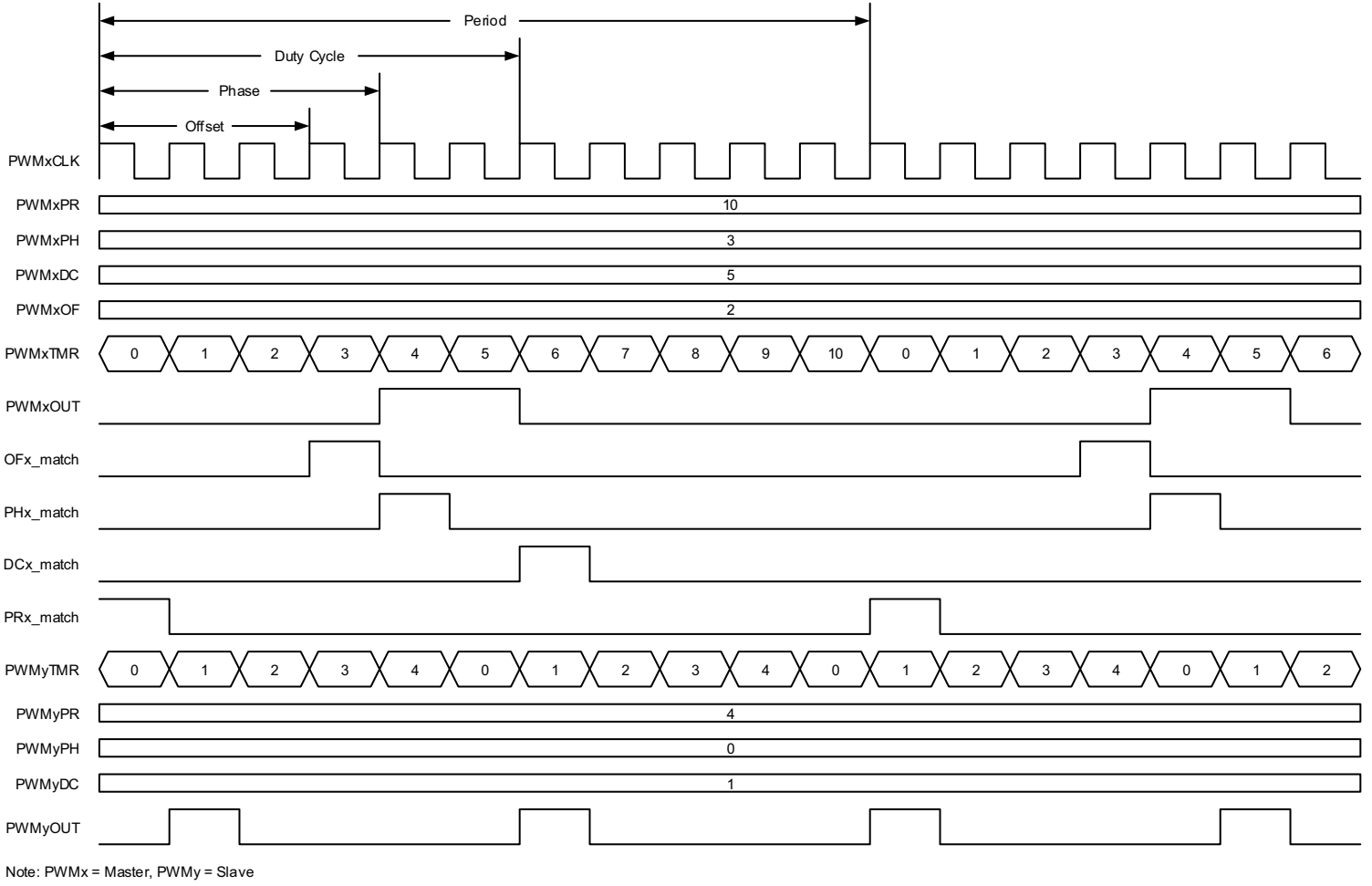
### 26.3.5 OFFSET MATCH IN CENTER ALIGNED MODE

When a master is operating in Center-Aligned mode the offset match event depends on which direction the PWMxTMR is counting. Clearing the OFO bit of the PWMxOFCON register will cause the OF_match event to occur when the timer is counting up. Setting the OFO bit of the PWMxOFCON register will cause the OF_match event to occur when the timer is counting down. The OFO bit is ignored in Non-Center-Aligned modes.

The OFO bit is double buffered and requires setting the LDA bit to take effect when the PWM module is operating.

Detailed timing diagrams of Center-Aligned mode using offset match control in Independent Slave with Sync Start mode can be seen in Figure 26-12 and Figure 26-13.

**FIGURE 26-8: INDEPENDENT RUN MODE TIMING DIAGRAM**




Note: PWMx = Master, PWMy = Slave

**FIGURE 26-9: SLAVE RUN MODE WITH SYNC START TIMING DIAGRAM**

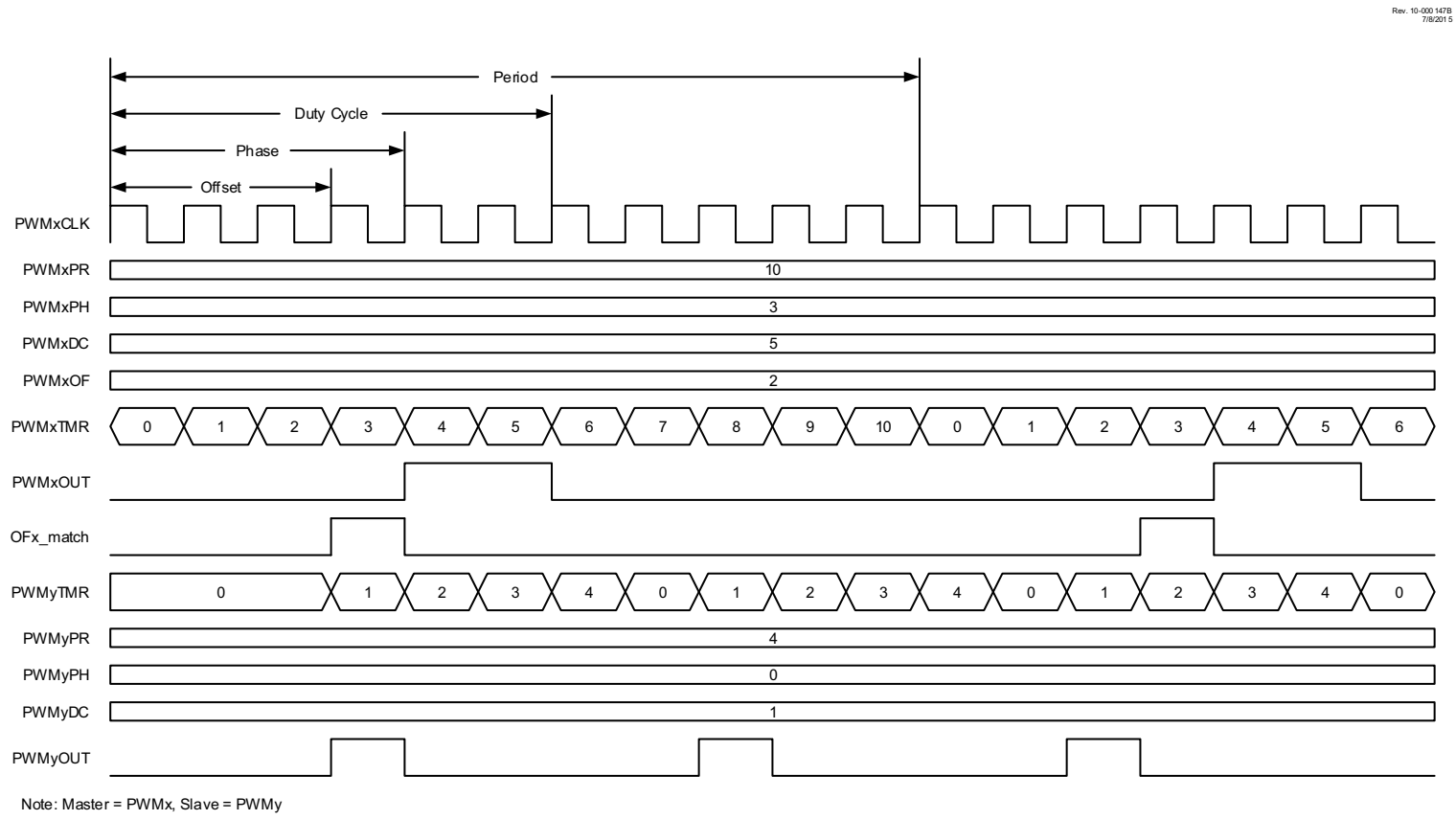


**FIGURE 26-10: ONE-SHOT SLAVE RUN MODE WITH SYNC START TIMING DIAGRAM**

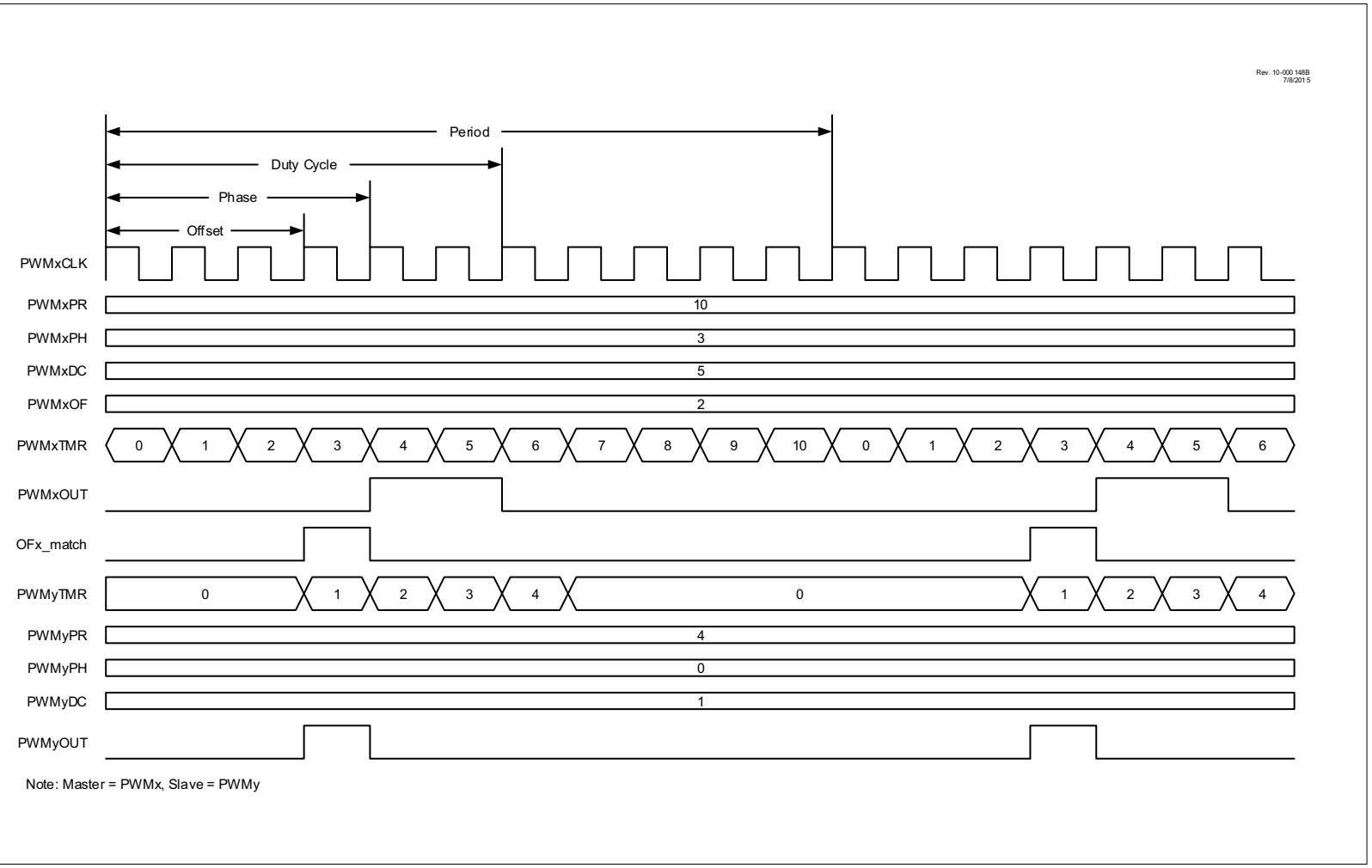


**FIGURE 26-11: CONTINUOUS SLAVE RUN MODE WITH IMMEDIATE RESET AND SYNC START TIMING DIAGRAM**

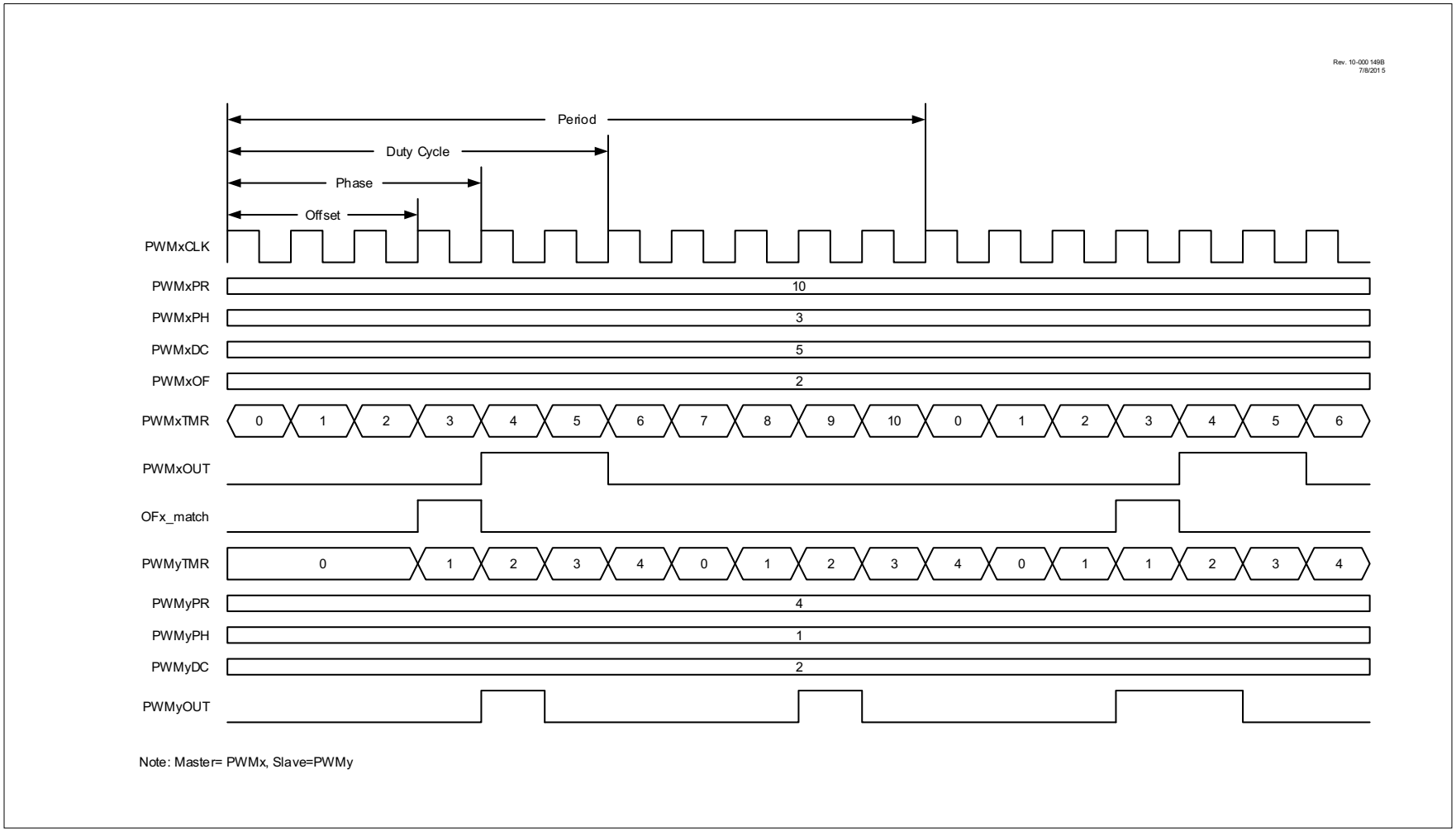


**FIGURE 26-12: OFFSET MATCH ON INCREMENTING TIMER TIMING DIAGRAM**

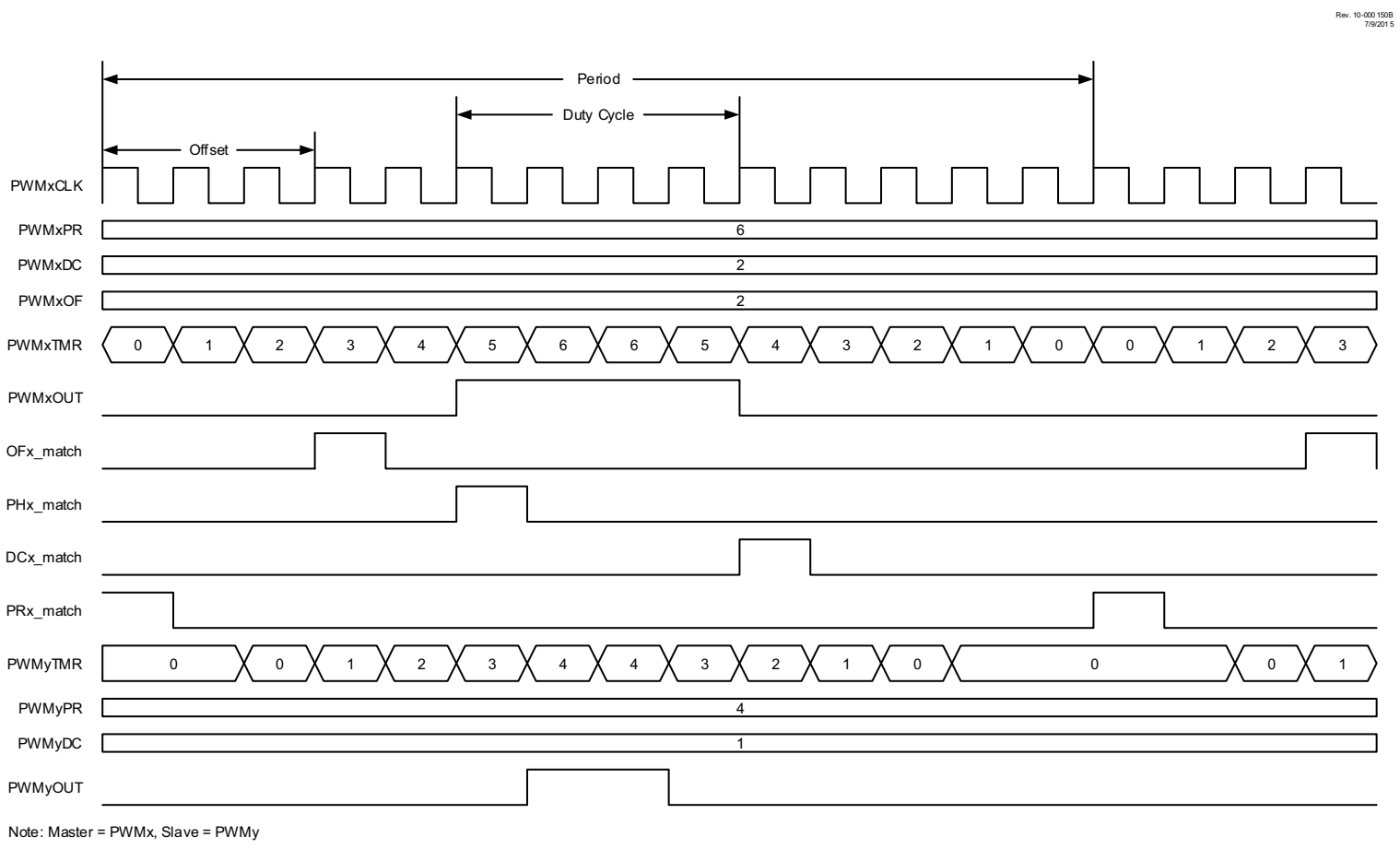
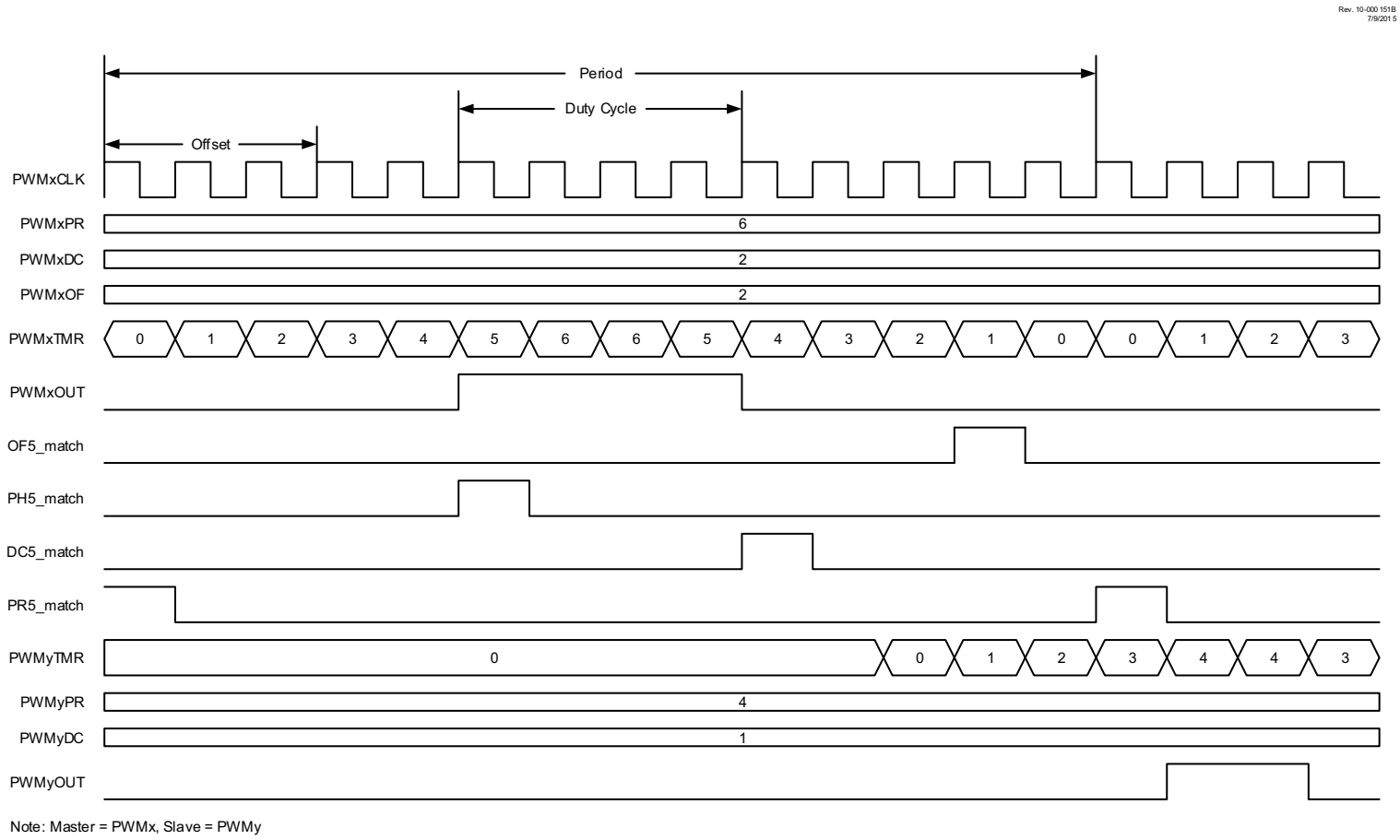


**FIGURE 26-13: OFFSET MATCH ON DECREMENTING TIMER TIMING DIAGRAM**



## 26.4 Reload Operation

Four of the PWM module control register pairs and one control bit are double buffered so that all can be updated simultaneously. These include:

- PWMxPHH:PWMxPHL register pair
- PWMxDCH:PWMxDCL register pair
- PWMxPRH:PWMxPRL register pair
- PWMxOFH:PWMxOFL register pair
- ODO control bit

When written to, these registers do not immediately affect the operation of the PWM. By default, writes to these registers will not be loaded into the PWM operating buffer registers until after the arming conditions are met. The arming control has two methods of operation:

- Immediate
- Triggered

The LDT bit of the PWMxLDCON register controls the arming method. Both methods require the LDA bit to be set. All four buffer pairs will load simultaneously at the loading event.

### 26.4.1 IMMEDIATE RELOAD

When the LDT bit is clear then the Immediate mode is selected and the buffers will be loaded at the first period event after the LDA bit is set. Immediate reloading is used when a PWM module is operating stand-alone or when the PWM module is operating as a master to other slave PWM modules.

### 26.4.2 TRIGGERED RELOAD

When the LDT bit is set then the Triggered mode is selected and a trigger event is required for the LDA bit to take effect. The trigger source is the buffer load event of one of the other PWM modules in the device. The triggering source is selected by the LDS<1:0> bits of the PWMxLDCON register. The buffers will be loaded at the first period event following the trigger event. Triggered reloading is used when a PWM module is operating as a slave to another PWM and it is necessary to synchronize the buffer reloads in both modules.

| | |
|---|---|
| **Note 1:** | The buffer load operation clears the LDA bit. |
| **2:** | If the LDA bit is set at the same time as PWMxTMR = PWMxPR, the LDA bit is ignored until the next period event. Such is the case when triggered reload is selected and the triggering event occurs simultaneously with the target's period event |

## 26.5 Operation in Sleep Mode

Each PWM module will continue to operate in Sleep mode when either the HFINTOSC or LFINTOSC is selected as the clock source by PWMxCLKCON<1:0>.

## 26.6 Interrupts

Each PWM module has four independent interrupts based on the phase, duty cycle, period, and offset match events. The interrupt flag is set on the rising edge of each of these signals. Refer to Figures 26-8 and 26-12 for detailed timing diagrams of the match signals.

## 26.7 Register Definitions: PWM Control

Long bit name prefixes for the 16-bit PWM peripherals are shown in Table 26-2. Refer to **Section1.1 "Register and Bit naming conventions"** for more information

**TABLE 26-2:**

| Peripheral | Bit Name Prefix |
|---|---|
| PWM5 | PWM5 |
| PWM6 | PWM6 |
| PWM11 | PWM11 |

**REGISTER 26-1: PWMxCON: PWM CONTROL REGISTER**

| R/W-0/0 | U-0 | R/HS/HC-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| EN | — | OUT | POL | MODE<1:0> | | — | — |
| bit 7 | | | | | | | bit 0 |

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

bit 7 **EN:** PWM Module Enable bit
1 = Module is enabled
0 = Module is disabled

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

bit 5 **OUT:** Output State of the PWM module

bit 4 **POL:** PWM Output Polarity Control bit
1 = PWM output active state is low
0 = PWM output active state is high

bit 3-2 **MODE<1:0>:** PWM Mode Control bits
11 = Center Aligned mode
10 = Toggle On Match mode
01 = Set On Match mode
00 = Standard PWM mode

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

**REGISTER 26-2: PWMxINTE: PWM INTERRUPT ENABLE REGISTER**

| U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | OFIE | PHIE | DCIE | PRIE |
| bit 7 | | | | | | | bit 0 |

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

bit 7-4 **Unimplemented:** Read as '0'

bit 3 **OFIE**: Offset Interrupt Enable bit
1 = Interrupt CPU on Offset Match
0 = Do not interrupt CPU on Offset Match

bit 2 **PHIE:** Phase Interrupt Enable bit
1 = Interrupt CPU on Phase Match
0 = Do not Interrupt CPU on Phase Match

bit 1 **DCIE:** Duty Cycle Interrupt Enable bit
1 = Interrupt CPU on Duty Cycle Match
0 = Do not interrupt CPU on Duty Cycle Match

bit 0 **PRIE:** Period Interrupt Enable bit
1 = Interrupt CPU on Period Match
0 = Do not interrupt CPU on Period Match

**REGISTER 26-3: PWMxINTF: PWM INTERRUPT REQUEST REGISTER**

| U-0 | U-0 | U-0 | U-0 | R/W/HS-0/0 | R/W/HS-0/0 | R/W/HS-0/0 | R/W/HS-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | OFIF | PHIF | DCIF | PRIF |
| bit 7 | | | | | | | bit 0 |

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

bit 7-4 **Unimplemented:** Read as '0'

bit 3 **OFIF**: Offset Interrupt Flag bit(1)
1 = Offset Match Event occurred
0 = Offset Match Event did not occur

bit 2 **PHIF:** Phase Interrupt Flag bit(1)
1 = Phase Match Event occurred
0 = Phase Match Event did not occur

bit 1 **DCIF:** Duty Cycle Interrupt Flag bit(1)
1 = Duty Cycle Match Event occurred
0 = Duty Cycle Match Event did not occur

bit 0 **PRIF:** Period Interrupt Flag bit(1)
1 = Period Match Event occurred
0 = Period Match Event did not occur

**Note 1:** Bit is forced clear by hardware while module is disabled (EN = 0).

**REGISTER 26-4: PWMxCLKCON: PWM CLOCK CONTROL REGISTER**

| U-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 |
|---|---|---|---|---|---|---|---|
| — | PS<2:0> | | | — | — | CS<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 6-4 **PS<2:0>:** Clock Source Prescaler Select bits
111 = Divide clock source by 128
110 = Divide clock source by 64
101 = Divide clock source by 32
100 = Divide clock source by 16
011 = Divide clock source by 8
010 = Divide clock source by 4
001 = Divide clock source by 2
000 = No Prescaler

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

bit 1-0 **CS<1:0>:** Clock Source Select bits
11 = Reserved
10 = LFINTOSC (continues to operate during Sleep)
01 = HFINTOSC (continues to operate during Sleep)
00 = FOSC

**REGISTER 26-5: PWMxLDCON: PWM RELOAD TRIGGER SOURCE SELECT REGISTER**

| R/W/HC-0/0 | R/W-0/0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| LDA(1) | LDT | — | — | — | — | LDS<1:0>(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 | HC = Cleared by hardware |

bit 7 **LDA:** Load Buffer Armed bit(1)
If LDT = 1:
1 = Load the ODO bit and OFx, PHx, DCx and PRx buffers at the end of the period in which the selected trigger occurs
0 = Do not load buffers, load has completed
If LDT = 0:
1 = Load the ODO bit and OFx, PHx, DCx and PRx buffers at the end of the current period
0 = Do not load buffers, load has completed

bit 6 **LDT:** Load Buffer on Trigger bit
1 = Wait for trigger selected by the LDS<1:0> bits to occur before enabling the LDA bit
0 = Load triggering is disabled. Buffer loads are controlled by the LDA bit alone.

bit 5-2 **Unimplemented:** Read as '0'

bit 1-0 **LDS<1:0>:** Load Trigger Source Select bits(2)
10 = LD11_trigger
01 = LD6_trigger
00 = LD5_trigger

**Note 1:** This bit is cleared by the module after a reload operation. It can be cleared in software to clear an existing arming event.
**2:** The source corresponding to a PWM module's own LDx_trigger is reserved.

**REGISTER 26-6: PWMxOFCON: PWM OFFSET TRIGGER SOURCE SELECT REGISTER**

| U-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 |
|---|---|---|---|---|---|---|---|
| — | OFM<1:0> | | OFO(1) | — | — | OFS<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 6-5 **OFM<1:0>:** Offset Mode Select bits
11 = Continuous Run Slave mode with offset triggered timer Reset and synchronized start
10 = One-shot Slave mode with offset triggered synchronized start
01 = Slave Run mode with offset triggered synchronized start
00 = Independent Run mode

bit 4 **OFO:** Offset Match Output Control bit(1)
If MODE<1:0> = 11 (PWM center aligned mode):
1 = OFx_match occurs when the PWMxTMR is counting up
0 = OFx_match occurs when the PWMxTMR is counting down
If MODE<1:0> = 00, 01, or 10 (all other modes):
this bit is ignored

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

bit 1-0 **OFS<1:0>:** Offset Trigger Source Select bit
10 = OF11_match
01 = OF6_match
00 = OF5_match

**Note 1:** The source corresponding to the PWM module's own OFx_match is reserved.

**REGISTER 26-7: PWMxPHH: PWMx PHASE COUNT HIGH 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 |
|---|---|---|---|---|---|---|---|
| PH<15:8> | | | | | | | |
| bit 7 | | | | | | | bit 0 |

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

bit 7-0 **PH<15:8>**: PWM Phase High bits
Upper eight bits of PWM phase count

**REGISTER 26-8: PWMxPHL: PWMx PHASE COUNT LOW 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 |
|---|---|---|---|---|---|---|---|
| PH<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 **PH<7:0>**: PWM Phase Low bits
Lower eight bits of PWM phase count

**REGISTER 26-9: PWMxDCH: PWMx DUTY CYCLE COUNT HIGH 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 |
|---|---|---|---|---|---|---|---|
| DC<15:8> | | | | | | | |
| bit 7 | | | | | | | bit 0 |

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

bit 7-0 **DC<15:8>**: PWM Duty Cycle High bits
Upper eight bits of PWM duty cycle count

**REGISTER 26-10: PWMxDCL: PWMx DUTY CYCLE COUNT LOW 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 |
|---|---|---|---|---|---|---|---|
| DC<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 **DC<7:0>**: PWM Duty Cycle Low bits
Lower eight bits of PWM duty cycle count

**REGISTER 26-11: PWMxPRH: PWMx PERIOD COUNT HIGH 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 |
|---|---|---|---|---|---|---|---|
| PR<15:8> | | | | | | | |
| bit 7 | | | | | | | bit 0 |

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

bit 7-0 **PR<15:8>**: PWM Period High bits
Upper eight bits of PWM period count

**REGISTER 26-12: PWMxPRL: PWMx PERIOD COUNT LOW 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 |
|---|---|---|---|---|---|---|---|
| PR<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 **PR<7:0>**: PWM Period Low bits
Lower eight bits of PWM period count

**REGISTER 26-13: PWMxOFH: PWMx OFFSET COUNT HIGH 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 |
|---|---|---|---|---|---|---|---|
| OF<15:8> | | | | | | | |
| bit 7 | | | | | | | bit 0 |

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

bit 7-0 **OF<15:8>**: PWM Offset High bits
Upper eight bits of PWM offset count

**REGISTER 26-14: PWMxOFL: PWMx OFFSET COUNT LOW 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 |
|---|---|---|---|---|---|---|---|
| OF<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 **OF<7:0>**: PWM Offset Low bits
Lower eight bits of PWM offset count

**REGISTER 26-15: PWMxTMRH: PWMx TIMER HIGH 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 |
|---|---|---|---|---|---|---|---|
| TMR<15:8> | | | | | | | |
| bit 7 | | | | | | | bit 0 |

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

bit 7-0 **TMR<15:8>**: PWM Timer High bits
Upper eight bits of PWM timer counter

**REGISTER 26-16: PWMxTMRL: PWMx TIMER LOW 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 |
|---|---|---|---|---|---|---|---|
| TMR<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 **TMR<7:0>**: PWM Timer Low bits
Lower eight bits of PWM timer counter

**Note:** There are no long and short bit name variants for the following three mirror registers

### REGISTER 26-17: PWMEN: PWMEN BIT MIRROR REGISTER

| U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | MPWM11EN | MPWM6EN | MPWM5EN |
| bit 7 | | | | | | | bit 0 |

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

bit 7-3 **Unimplemented:** Read as '0'

bit 2-0 **PWMxEN:** PWM11/PWM6/PWM5 Enable bits
Mirror copy of each PWM module's PWMxCON<7> bit

### REGISTER 26-18: PWMLD: LDA BIT MIRROR REGISTER

| U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | MPWM11LD | MPWM6LD | MPWM5LD |
| bit 7 | | | | | | | bit 0 |

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

bit 7-3 **Unimplemented:** Read as '0'

bit 2 **MPWM11LD:** PWM11LD bits
Mirror copy of each PWM module's PWMxLDCON<7> bit

bit 1 **MPWM6LD:** PWM6LD bits
Mirror copy of each PWM module's PWMxLDCON<7> bit

bit 0 **MPWM5LD:** PWM5LD bits
Mirror copy of each PWM module's PWMxLDCON<7> bit

### REGISTER 26-19: PWMOUT: PWMOUT BIT MIRROR REGISTER

| U-0 | U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | MPWM11OUT | MPWM6OUT | MPWM5OUT |
| bit 7 | | | | | | | bit 0 |

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

bit 7-3 **Unimplemented:** Read as '0'

bit 2 **MPWM11OUT:** PWM11 OUT bits
Mirror copy of each PWM module's PWMxCON<5> bit

bit 1 **MPWM6OUT:** PWM6 OUT bits
Mirror copy of each PWM module's PWMxCON<5> bit

bit 0 **MPWM5OUT:** PWM5 OUT bits
Mirror copy of each PWM module's PWMxCON<5> bit

**TABLE 26-3: SUMMARY OF REGISTERS ASSOCIATED WITH PWM**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| OSCCON | SPLLEN | IRCF<3:0> | | | | — | SCS<1:0> | | 92 |
| PWMEN | — | — | — | — | — | MPWM11EN | MPWM6EN | MPWM5EN | 317 |
| PWMLD | — | — | — | — | — | MPWM11LD | MPWM6LD | MPWM5LD | 317 |
| PWMOUT | — | — | — | — | — | MPWM11OUT | MPWM6OUT | MPWM5OUT | 317 |
| PWM5PHL | PH<7:0> | | | | | | | | 312 |
| PWM5PHH | PH<15:8> | | | | | | | | 312 |
| PWM5DCL | DC<7:0> | | | | | | | | 313 |
| PWM5DCH | DC<15:8> | | | | | | | | 313 |
| PWM5PRL | PR<7:0> | | | | | | | | 314 |
| PWM5PRH | PR<15:8> | | | | | | | | 314 |
| PWM5OFL | OF<7:0> | | | | | | | | 315 |
| PWM5OFH | OF<15:8> | | | | | | | | 315 |
| PWM5TMRL | TMR<7:0> | | | | | | | | 316 |
| PWM5TMRH | TMR<15:8> | | | | | | | | 316 |
| PWM5CON | EN | — | OUT | POL | MODE<1:0> | | — | — | 307 |
| PWM5INTE | — | — | — | — | OFIE | PHIE | DCIE | PRIE | 308 |
| PWM5INTF | — | — | — | — | OFIF | PHIF | DCIF | PRIF | 308 |
| PWM5CLKCON | — | PS<2:0> | | | — | — | CS<1:0> | | 309 |
| PWM5LDCON | LDA | LDT | — | — | — | — | LDS<1:0> | | 310 |
| PWM5OFCON | — | OFM<1:0> | | OFO | — | — | OFS<1:0> | | 311 |
| PWM6PHL | PH<7:0> | | | | | | | | 312 |
| PWM6PHH | PH<15:8> | | | | | | | | 312 |
| PWM6DCL | DC<7:0> | | | | | | | | 313 |
| PWM6DCH | DC<15:8> | | | | | | | | 313 |
| PWM6PRL | PR<7:0> | | | | | | | | 314 |
| PWM6PRH | PR<15:8> | | | | | | | | 314 |
| PWM6OFL | OF<7:0> | | | | | | | | 315 |
| PWM6OFH | OF<15:8> | | | | | | | | 315 |
| PWM6TMRL | TMR<7:0> | | | | | | | | 316 |
| PWM6TMRH | TMR<15:8> | | | | | | | | 316 |
| PWM6CON | EN | — | OUT | POL | MODE<1:0> | | — | — | 307 |
| PWM6INTE | — | — | — | — | OFIE | PHIE | DCIE | PRIE | 308 |
| PWM6INTF | — | — | — | — | OFIF | PHIF | DCIF | PRIF | 308 |
| PWM6CLKCON | — | PS<2:0> | | | — | — | CS<1:0> | | 309 |
| PWM6LDCON | LDA | LDT | — | — | — | — | LDS<1:0> | | 310 |
| PWM6OFCON | — | OFM<1:0> | | OFO | — | — | OFS<1:0> | | 311 |
| PWM11PHL | PH<7:0> | | | | | | | | 312 |
| PWM11PHH | PH<15:8> | | | | | | | | 312 |
| PWM11DCL | DC<7:0> | | | | | | | | 313 |
| PWM11DCH | DC<15:8> | | | | | | | | 313 |
| PWM11PRL | PR<7:0> | | | | | | | | 314 |
| PWM11PRH | PR<15:8> | | | | | | | | 314 |
| PWM11OFL | OF<7:0> | | | | | | | | 315 |
| PWM11OFH | OF<15:8> | | | | | | | | 315 |
| PWM11TMRL | TMR<7:0> | | | | | | | | 316 |
| PWM11TMRH | TMR<15:8> | | | | | | | | 316 |
| PWM11CON | EN | — | OUT | POL | MODE<1:0> | | — | — | 307 |
| PWM11INTE | — | — | — | — | OFIE | PHIE | DCIE | PRIE | 308 |
| PWM11INTF | — | — | — | — | OFIF | PHIF | DCIF | PRIF | 308 |
| PWM11CLKCON | — | PS<2:0> | | | — | — | CS<1:0> | | 309 |
| PWM11LDCON | LDA | LDT | — | — | — | — | LDS<1:0> | | 310 |
| PWM11OFCON | — | OFM<1:0> | | OFO | — | — | OFS<1:0> | | 311 |

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|

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

**TABLE 26-4: SUMMARY OF CONFIGURATION WORDS 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> | | — | 71 |
| | 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 clock sources.

## 27.0 COMPLEMENTARY OUTPUT GENERATOR (COG) MODULES

The primary purpose of the Complementary Output Generator (COG) is to convert a single-output PWM signal into a two-output complementary PWM signal. The COG can also convert two separate input events into a single or complementary PWM output.

**TABLE 27-1: AVAILABLE COG MODULES**

| Device | COG1 | COG2 | COG3 |
|---|---|---|---|
| PIC16(L)F1773 | • | • | • |
| PIC16(L)F1776 | • | • | • |

The COG PWM frequency and duty cycle are determined by a rising event input and a falling event input. The rising event and falling event may be the same source. Sources may be synchronous or asynchronous to the COG_clock.

The rate at which the rising event occurs determines the PWM frequency. The time from the rising event to the falling event determines the duty cycle.

A selectable clock input is used to generate the phase delay, blanking, and dead-band times. Dead-band time can also be generated with a programmable delay chain, which is independent from all clock sources.

Simplified block diagrams of the various COG modes are shown in Figure 27-2 through Figure 27-6.

The COG module has the following features:

- Six modes of operation:
  - Steered PWM mode
  - Synchronous Steered PWM mode
  - Forward Full-Bridge mode
  - Reverse Full-Bridge mode
  - Half-Bridge mode
  - Push-Pull mode
- Selectable COG_clock clock source
- Independently selectable rising event sources
- Independently selectable falling event sources
- Independently selectable edge or level event sensitivity
- Independent output polarity selection
- Phase delay with independent rising and falling delay times
- Dead-band control with:
  - independent rising and falling event dead-band times
  - Synchronous and asynchronous timing
- Blanking control with independent rising and falling event blanking times
- Auto-shutdown control with:
  - Independently selectable shutdown sources
  - Auto-restart enable
  - Auto-shutdown pin override control (high, low, off, and High-Z)

### 27.1 Output to Pins (all modes)

The COG peripheral has four outputs: COGA, COGB, COGC, and COGD.

The operating mode, selected with the MD<2:0> bits of the COGxCON0 register, determine the waveform available at each output. An individual peripheral source control for each device pin selects the pin or pins at which the outputs will appear. Please refer to the RxyPPS register (Register 12-2) for more information.

### 27.2 Event-Driven PWM (All Modes)

Besides generating PWM and complementary outputs from a single PWM input, the COG can also generate PWM waveforms from a periodic rising event and a separate falling event. In this case, the falling event is usually derived from analog feedback within the external PWM driver circuit. In this configuration, high-power switching transients may trigger a false falling event that needs to be blanked out. The COG can be configured to blank falling (and rising) event inputs for a period of time immediately following the rising (and falling) event drive output. This is referred to as input blanking and is covered in **Section 27.8 "Blanking Control"**.

It may be necessary to guard against the possibility of external circuit faults. In this case, the active drive must be terminated before the Fault condition causes damage. This is referred to as auto-shutdown and is covered in **Section 27.10 "Auto-Shutdown Control"**.

The COG can be configured to operate in phase delayed conjunction with another PWM. The active drive cycle is delayed from the rising event by a phase delay timer. Phase delay is covered in more detail in **Section 27.9 "Phase Delay"**.

A typical operating waveform, with phase delay and dead band, generated from a single CCP1 input is shown in Figure 27-10.

## 27.3 Modes of Operation

### 27.3.1 STEERED PWM MODES

In Steered PWM mode, the PWM signal derived from the input event sources is output as a single phase PWM which can be steered to any combination of the four COG outputs. Output steering takes effect on the instruction cycle following the write to the COGxSTR register.

Synchronous Steered PWM mode is identical to the Steered PWM mode except that changes to the output steering take effect on the first rising event after the COGxSTR register write. Static output data is not synchronized.

Steering mode configurations are shown in Figure 27-2 and Figure 27-3.

Steered PWM and Synchronous Steered PWM modes are selected by setting the MD<2:0> bits of the COGxCON0 register (Register 27-1) to '000' and '001', respectively.

### 27.3.2 FULL-BRIDGE MODES

In both Forward and Reverse Full-Bridge modes, two of the four COG outputs are active and the other two are inactive. Of the two active outputs, one is modulated by the PWM input signal and the other is on at 100% duty cycle. When the direction is changed, the dead-band time is inserted to delay the modulated output. This gives the unmodulated driver time to shut down, thereby, preventing shoot-through current in the series connected power devices.

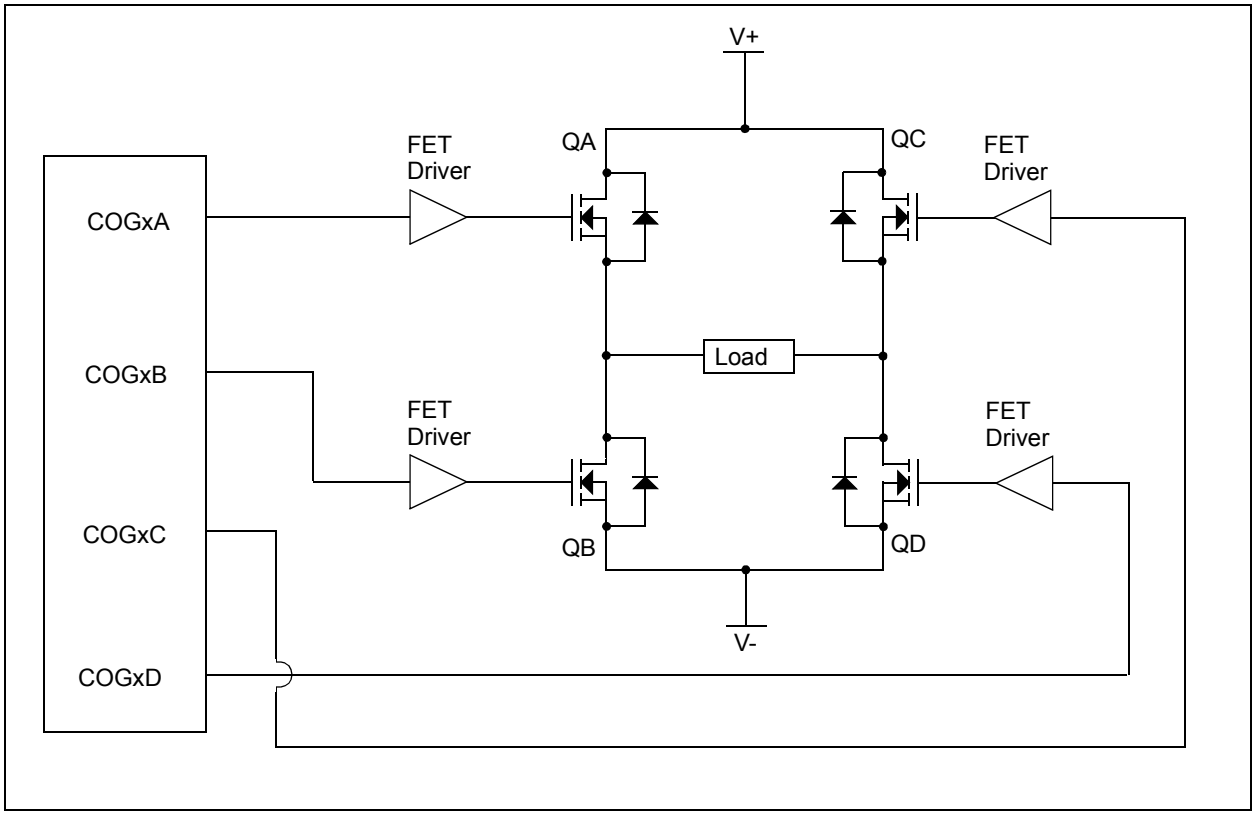
In Forward Full-Bridge mode, the PWM input modulates the COGxD output and drives the COGA output at 100%.

In Reverse Full-Bridge mode, the PWM input modulates the COGxB output and drives the COGxC output at 100%.

The full-bridge configuration is shown in Figure 27-4. Typical full-bridge waveforms are shown in Figure 27-12 and Figure 27-13.

Full-Bridge Forward and Full-Bridge Reverse modes are selected by setting the MD<2:0> bits of the COGxCON0 register to '010' and '011', respectively.

**FIGURE 27-1: EXAMPLE OF FULL-BRIDGE APPLICATION**



### 27.3.3 HALF-BRIDGE MODE

In Half-Bridge mode, the COG generates a two output complementary PWM waveform from rising and falling event sources. In the simplest configuration, the rising and falling event sources are the same signal, which is a PWM signal with the desired period and duty cycle. The COG converts this single PWM input into a dual complementary PWM output. The frequency and duty cycle of the dual PWM output match those of the single input PWM signal. The off-to-on transition of each output can be delayed from the on-to-off transition of the other output, thereby, creating a time immediately after the PWM transition where neither output is driven. This is referred to as dead-band time and is covered in **Section 27.7 "Dead-Band Control"**.

The half-bridge configuration is shown in Figure 27-5. A typical operating waveform, with dead band, generated from a single CCP1 input is shown in Figure 27-9.

The primary output is available on either, or both, COGxA and COGxC. The complementary output is available on either, or both, COGxB and COGxD.

Half-Bridge mode is selected by setting the MD<2:0> bits of the COGxCON0 register to '100'.

### 27.3.4 PUSH-PULL MODE

In Push-Pull mode, the COG generates a single PWM output that alternates between the two pairs of the COG outputs at every PWM period. COGxA has the same signal as COGxC. COGxB has the same signal as COGxD. The output drive activates with the rising input event and terminates with the falling event input. Each rising event starts a new period and causes the output to switch to the COG pair not used in the previous period.

The Push-Pull configuration is shown in Figure 27-6. A typical Push-Pull waveform generated from a single CCP1 input is shown in Figure 27-11.

Push-Pull mode is selected by setting the MD<2:0> bits of the COGxCON0 register to '101'.

**FIGURE 27-2: SIMPLIFIED COG BLOCK DIAGRAM (STEERED PWM MODE, MD = 0)**

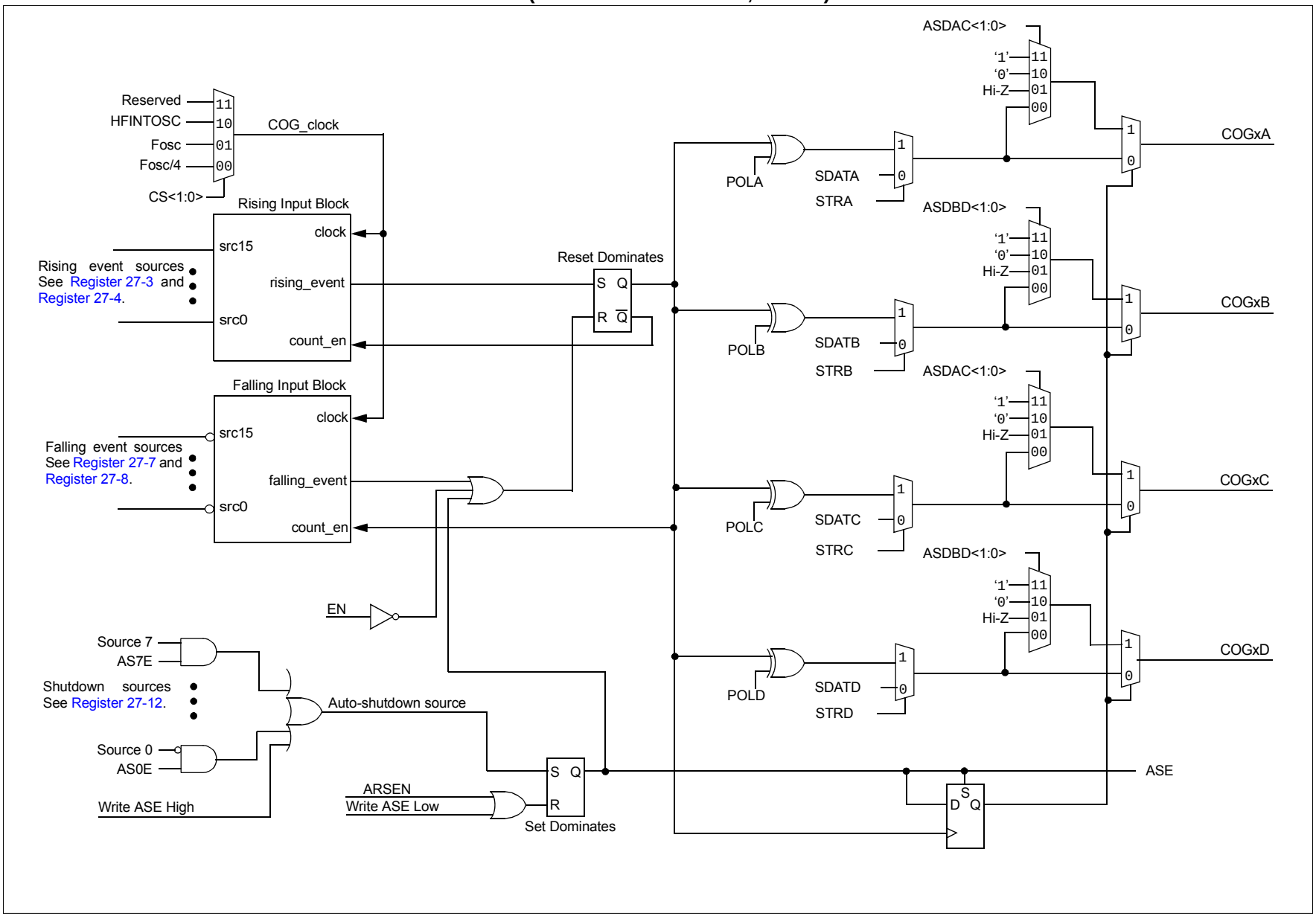


**FIGURE 27-3: SIMPLIFIED COG BLOCK DIAGRAM (SYNCHRONOUS STEERED PWM MODE, MD = 1)**

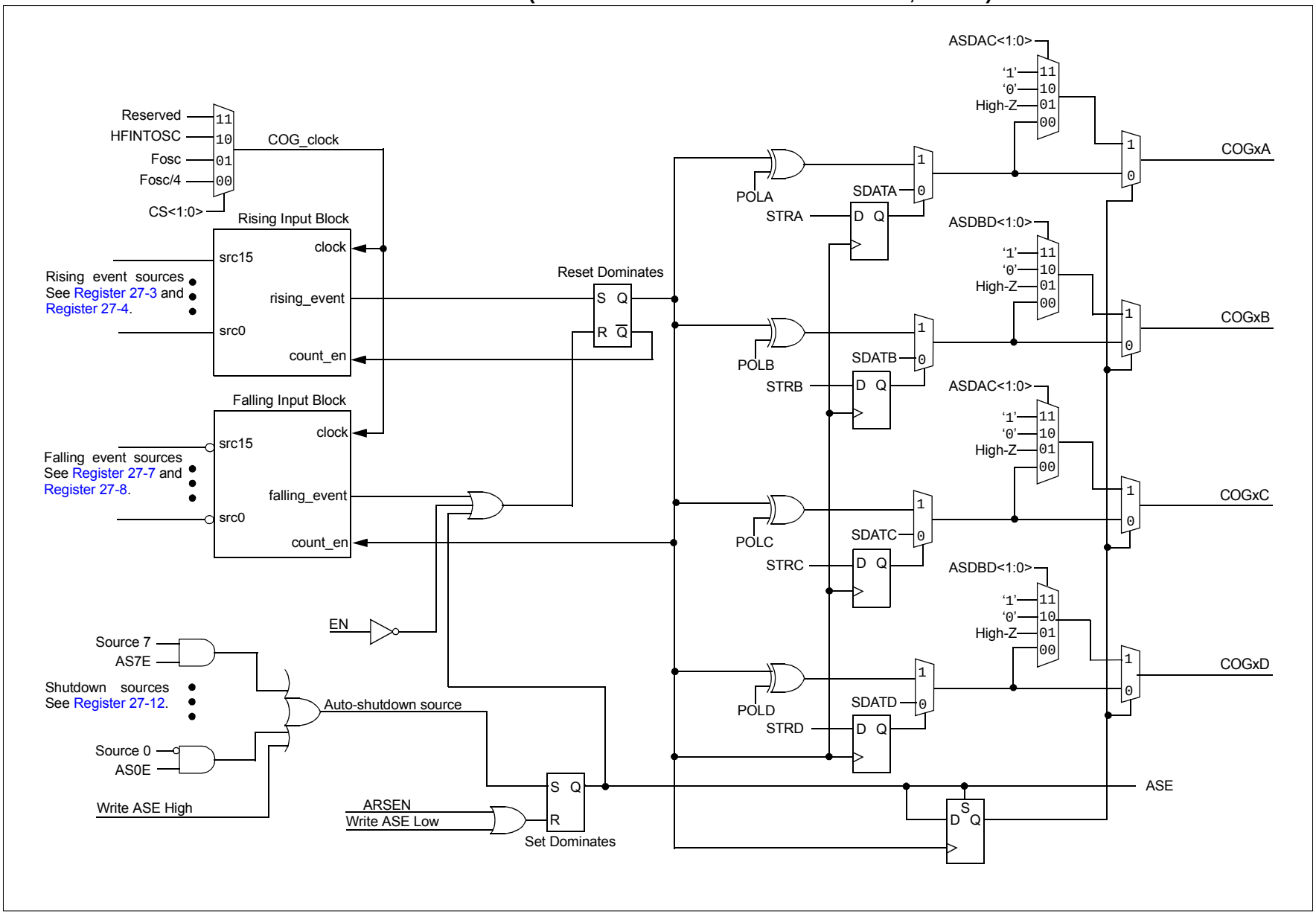


**FIGURE 27-4: SIMPLIFIED COG BLOCK DIAGRAM (FULL-BRIDGE MODES, FORWARD: MD = 2, REVERSE: MD = 3)**

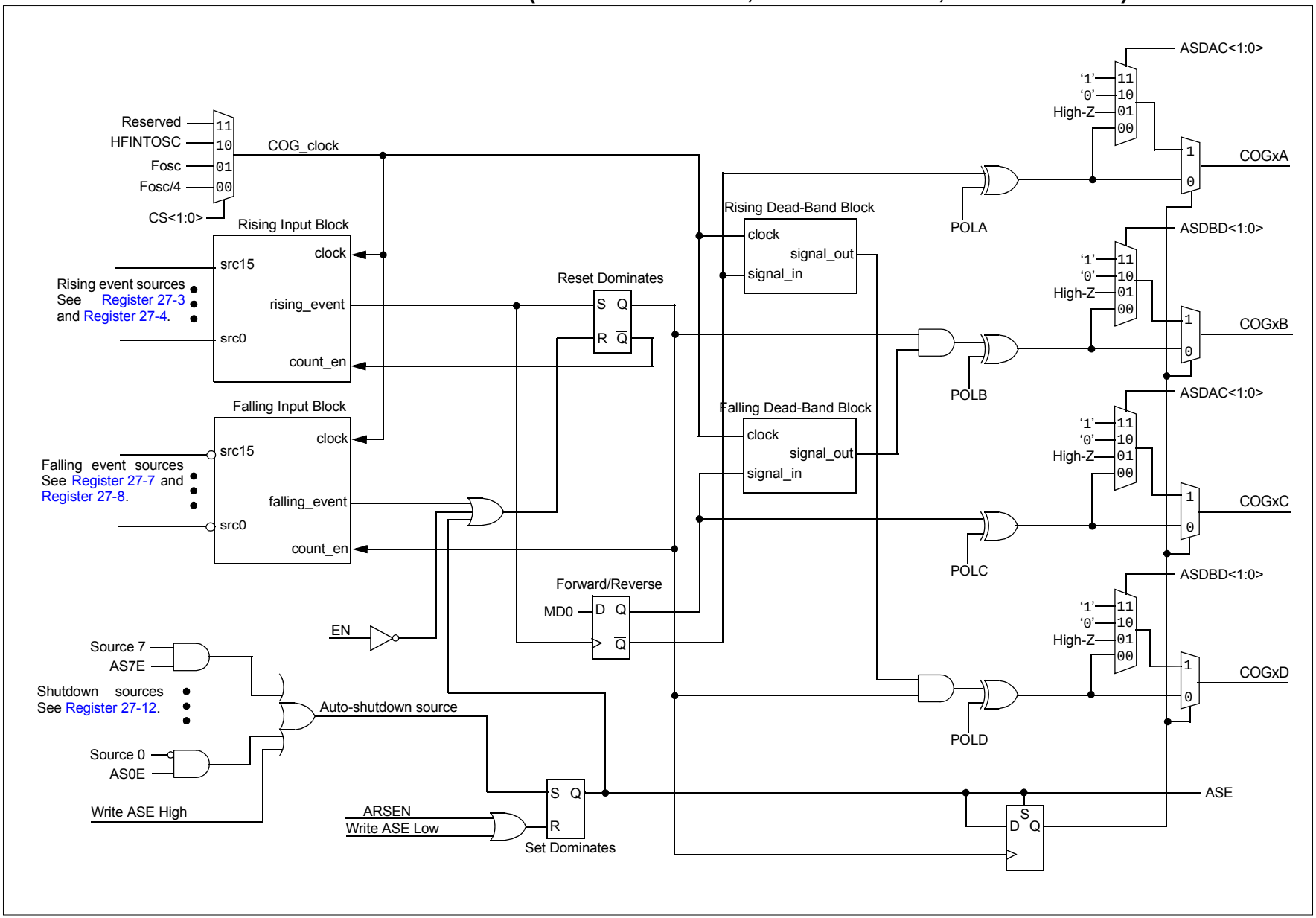


**FIGURE 27-5: SIMPLIFIED COG BLOCK DIAGRAM (HALF-BRIDGE MODE, MD = 4)**

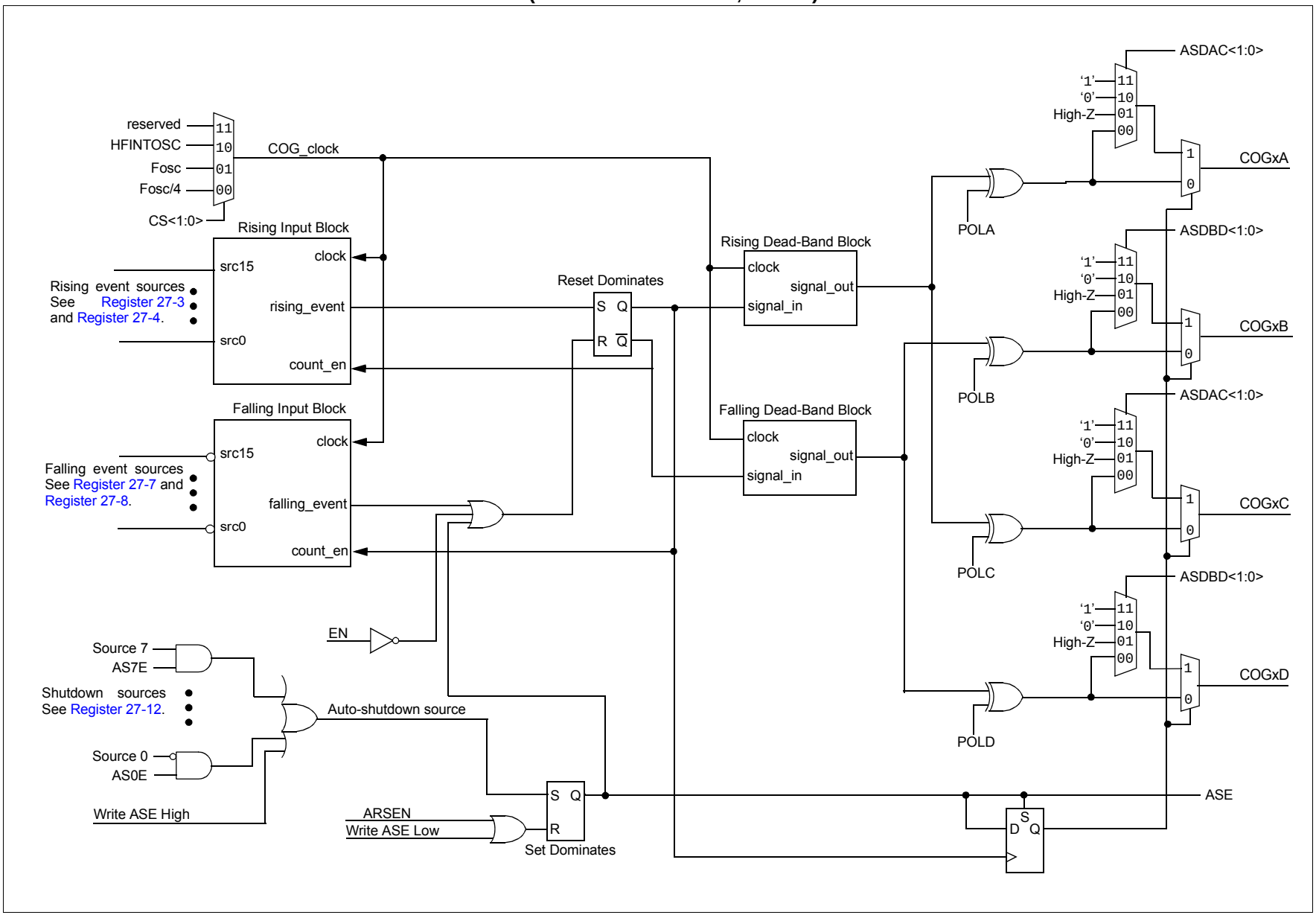


**FIGURE 27-6: SIMPLIFIED COG BLOCK DIAGRAM (PUSH-PULL MODE, MD = 5)**

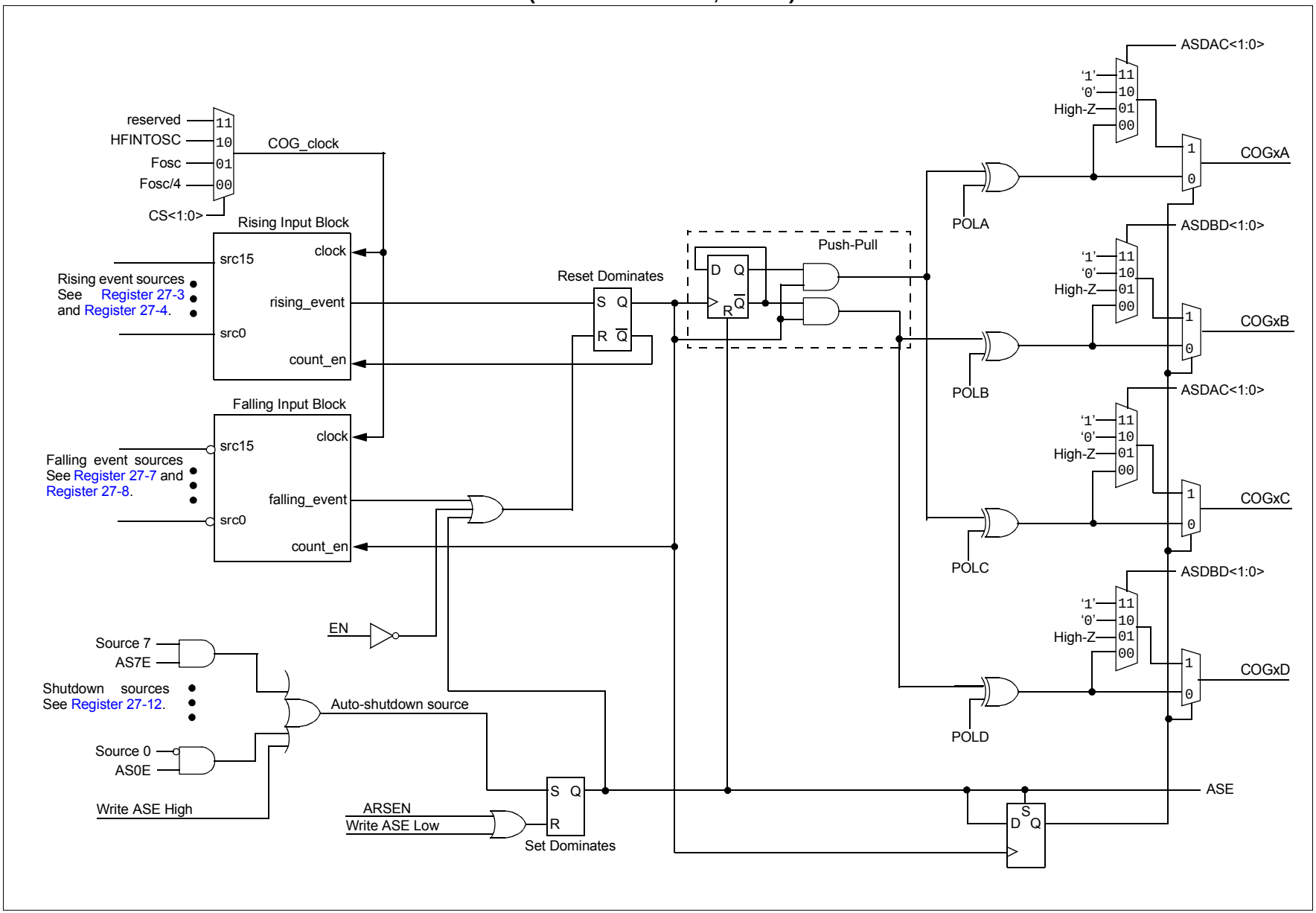


**FIGURE 27-7: COG (RISING/FALLING) INPUT BLOCK**

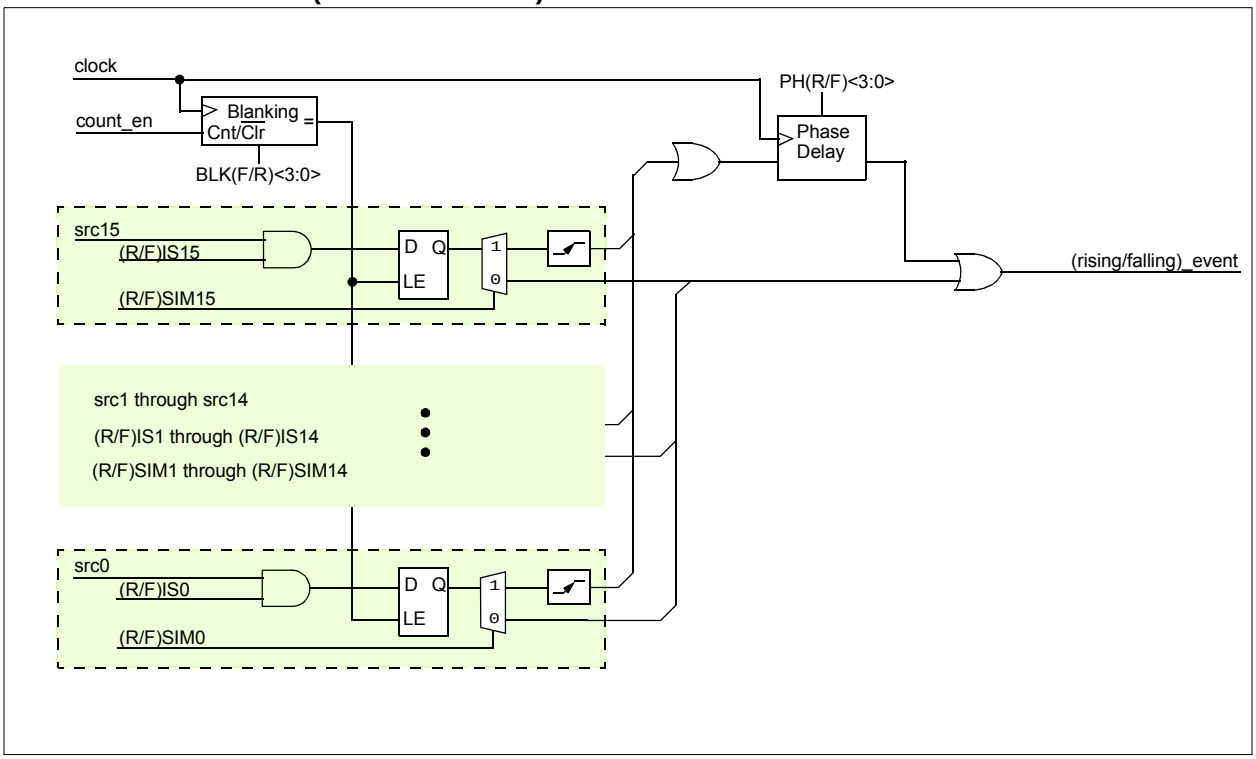



**FIGURE 27-8: COG (RISING/FALLING) DEAD-BAND BLOCK**

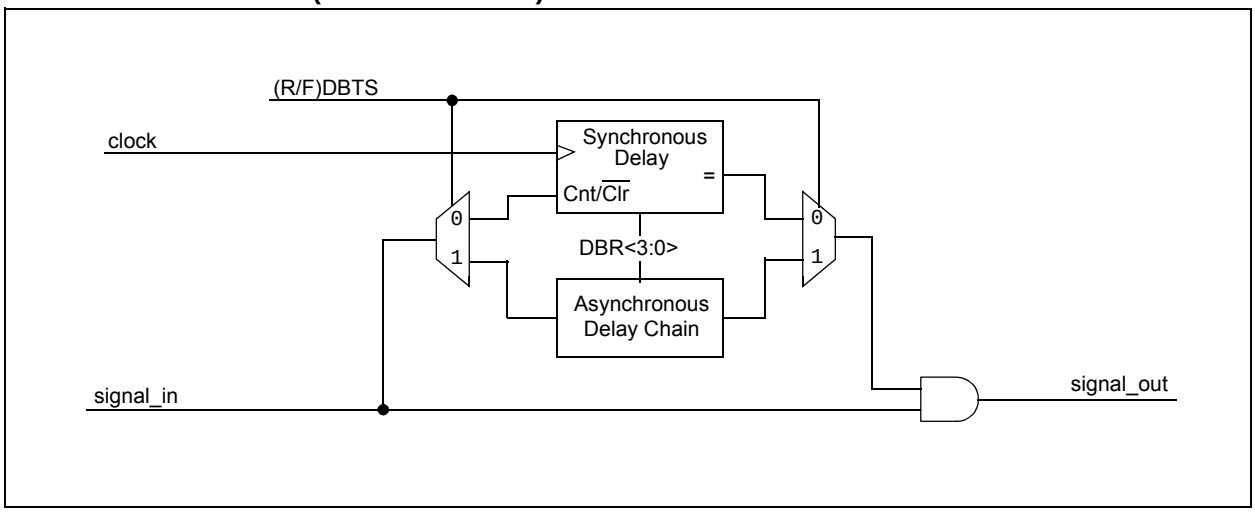


**FIGURE 27-9: TYPICAL HALF-BRIDGE MODE COG OPERATION WITH CCP1**

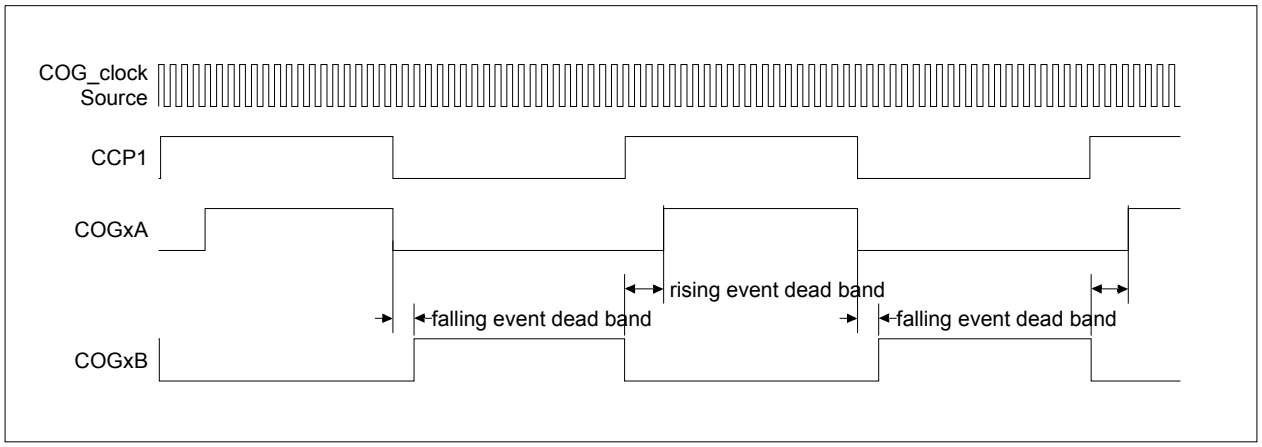



**FIGURE 27-10: HALF-BRIDGE MODE COG OPERATION WITH CCP1 AND PHASE DELAY**

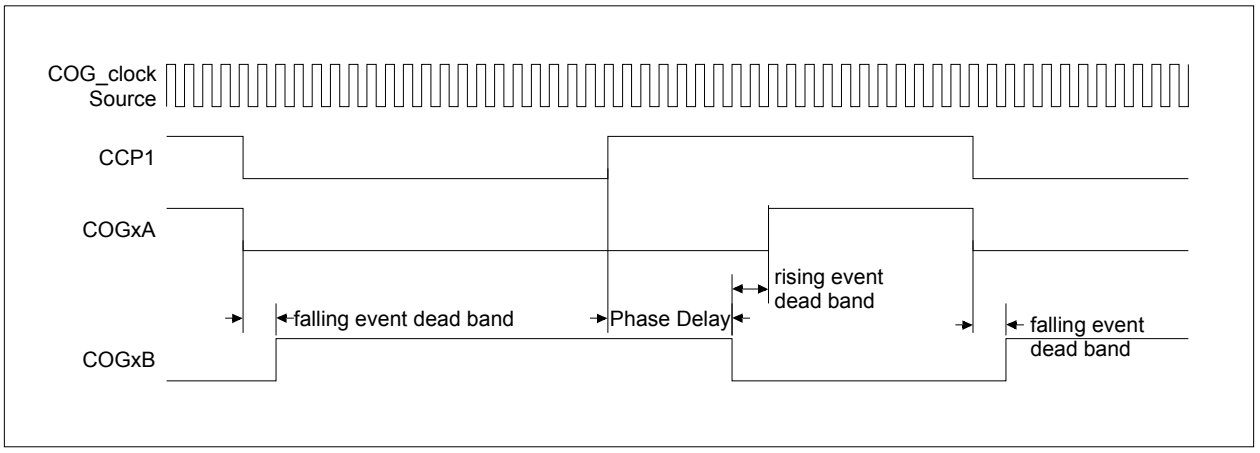



**FIGURE 27-11: PUSH-PULL MODE COG OPERATION WITH CCP1**

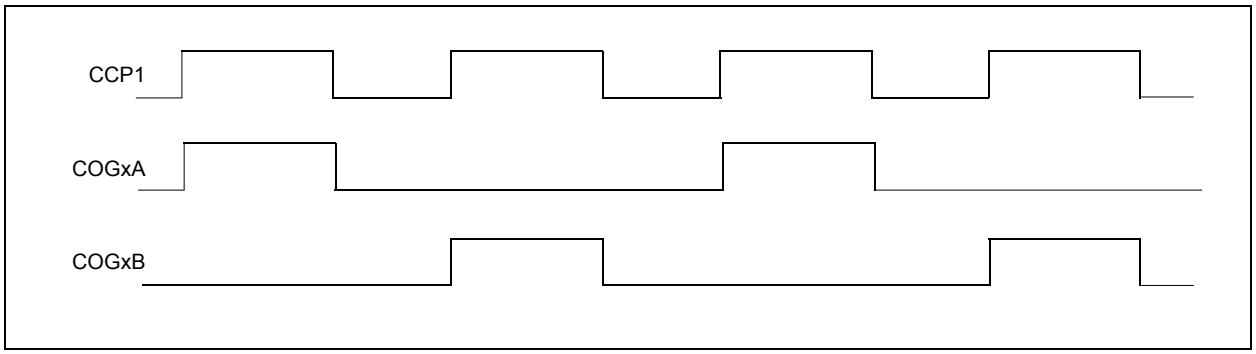


**FIGURE 27-12: FULL-BRIDGE FORWARD MODE COG OPERATION WITH CCP1**

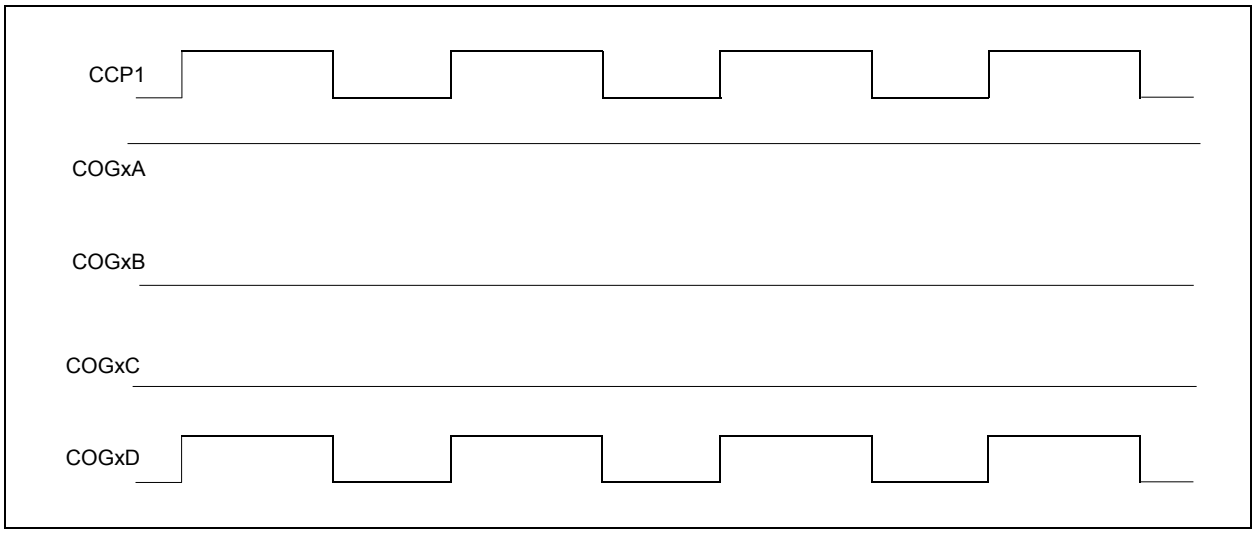
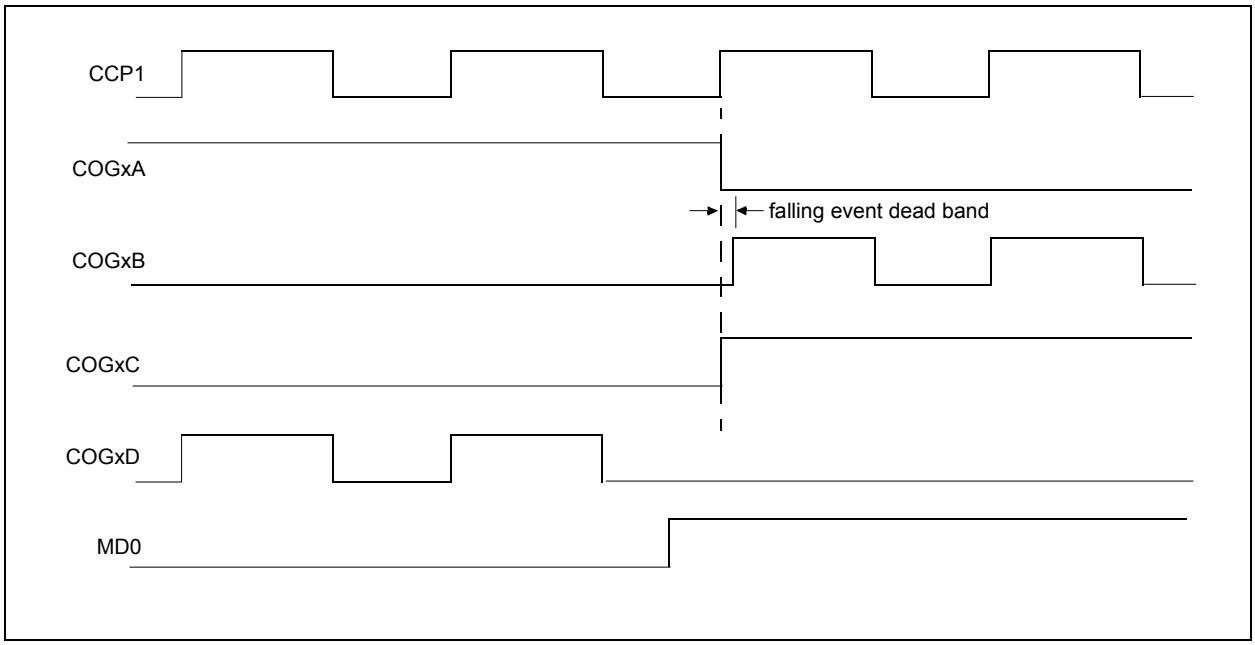



**FIGURE 27-13: FULL-BRIDGE MODE COG OPERATION WITH CCP1 AND DIRECTION CHANGE**



## 27.4 Clock Sources

The COG_clock is used as the reference clock to the various timers in the peripheral. Timers that use the COG_clock include:

- Rising and falling dead-band time
- Rising and falling blanking time
- Rising and falling event phase delay

Clock sources available for selection include:

- 16 MHz HFINTOSC (active during Sleep)
- Instruction clock (FOSC/4)
- System clock (FOSC)

The clock source is selected with the CS<1:0> bits of the COGxCON0 register (Register 27-1).

## 27.5 Selectable Event Sources

The COG uses any combination of independently selectable event sources to generate the complementary waveform. Sources fall into two categories:

- Rising event sources
- Falling event sources

The rising event sources are selected by setting bits in the COGxRIS0 and COGxRIS1 registers (Register 27-3 and Register 27-4). The falling event sources are selected by setting bits in the COGxFIS0 and COGxF1 registers (Register 27-7 and Register 27-8). All selected sources are OR'd together to generate the corresponding event signal. Refer to Figure 27-7.

### 27.5.1 EDGE VS. LEVEL SENSING

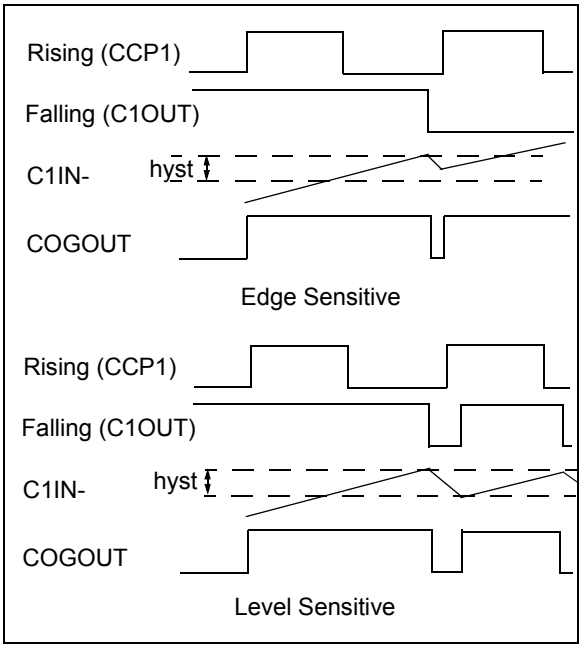
Event input detection may be selected as level or edge sensitive. The detection mode is individually selectable for every source. Rising source detection modes are selected with the COGxRSIM0 and COGxRSIM1 registers (Register 27-5 and Register 27-6). Falling source detection modes are selected with the COGxFSIM0 and COGxFSIM1 registers (Register 27-9 and Register 27-10). A set bit selects edge detection for the corresponding event source. A cleared bit selects level detection.

In general, events that are driven from a periodic source should be edge detected and events that are derived from voltage thresholds at the target circuit should be level sensitive. Consider the following two examples:

1. The first example is an application in which the period is determined by a 50% duty cycle clock on the rising event input and the COG output duty cycle is determined by a voltage level fed back through a comparator on the falling event input. If the clock input is level sensitive, duty cycles less than 50% will exhibit erratic operation because the level sensitive clock will suppress the comparator feedback.

2. The second example is similar to the first except that the duty cycle is close to 100%. The feedback comparator high-to-low transition trips the COG drive off, but almost immediately the period source turns the drive back on. If the off cycle is short enough, the comparator input may not reach the low side of the hysteresis band precluding an output change. The comparator output stays low and without a high-to-low transition to trigger the edge sense, the drive of the COG output will be stuck in a constant drive-on condition. See Figure 27-14.

**FIGURE 27-14: EDGE VS. LEVEL SENSE**



### 27.5.2 RISING EVENT

The rising event starts the PWM output active duty cycle period. The rising event is the low-to-high transition of the rising_event output. When the rising event phase delay and dead-band time values are zero, the primary output starts immediately. Otherwise, the primary output is delayed. The rising event source causes all the following actions:

- Start rising event phase delay counter (if enabled).
- Clear complementary output after phase delay.
- Start falling event input blanking (if enabled).
- Start dead-band delay (if enabled).
- Set primary output after dead-band delay expires.

### 27.5.3 FALLING EVENT

The falling event terminates the PWM output active duty cycle period. The falling event is the high-to-low transition of the falling_event output. When the falling event phase delay and dead-band time values are zero, the complementary output starts immediately. Otherwise, the complementary output is delayed. The falling event source causes all the following actions:

- Start falling event phase delay counter (if enabled).
- Clear primary output.
- Start rising event input blanking (if enabled).
- Start falling event dead-band delay (if enabled).
- Set complementary output after dead-band delay expires.

## 27.6 Output Control

Upon disabling, or immediately after enabling the COG module, the primary COG outputs are inactive and complementary COG outputs are active.

### 27.6.1 OUTPUT ENABLES

There are no output enable controls in the COG module. Instead, each device pin has an individual output selection control called the PPS register. All four COG outputs are available for selection in the PPS register of every pin.

When a COG output is enabled by PPS selection, the output on the pin has several possibilities which depend on the mode, steering control, EN bit, and shutdown state as shown in Table 27-2 and Table 27-3.

.

**TABLE 27-2: PIN OUTPUT STATES MD<2:0> = 00x**

| EN | STR bit | Shutdown | Output |
|---|---|---|---|
| x | 0 | Inactive | Static steering data |
| x | 1 | Active | Shutdown override |
| 0 | 1 | Inactive | Inactive state |
| 1 | 1 | Inactive | Active PWM signal |

**TABLE 27-3: PIN OUTPUT STATES MD<2:0> > 001**

| EN | STR bit | Shutdown | Output |
|---|---|---|---|
| x | x | Inactive | Inactive state |
| x | x | Active | Shutdown override |
| 1 | x | Inactive | Active PWM signal |

### 27.6.2 POLARITY CONTROL

The polarity of each COG output can be selected independently. When the output polarity bit is set, the corresponding output is active-low. Clearing the output polarity bit configures the corresponding output as active-high. However, polarity affects the outputs in only one of the four shutdown override modes. See **Section 27.10 "Auto-Shutdown Control"** for more details.

Output polarity is selected with the POLA through POLD bits of the COGxCON1 register (Register 27-2).

## 27.7 Dead-Band Control

The dead-band control provides for non-overlapping PWM output signals to prevent shoot-through current in the external power switches. Dead-band time affects the output only in the Half-Bridge mode and when changing direction in the Full-Bridge mode.

The COG contains two dead-band timers. One dead-band timer is used for rising event dead-band control. The other is used for falling event dead-band control. Timer modes are selectable as either:

- Asynchronous delay chain
- Synchronous counter

The Dead-Band Timer mode is selected for the rising event and falling event dead-band times with the respective RDBS and FDBS bits of the COGxCON1 register (Register 27-2).

In Half-Bridge mode, the rising event dead-band time delays all selected primary outputs from going active for the selected dead-band time after the rising event. COGxA and COGxC are the primary outputs in Half-Bridge mode.

In Half-Bridge mode, the falling event dead-band time delays all selected complementary outputs from going active for the selected dead-band time after the falling event. COGxB and COGxD are the complementary outputs in Half-Bridge mode.

In Full-Bridge mode, the dead-band delay occurs only during direction changes. The modulated output is delayed for the falling event dead-band time after a direction change from forward to reverse. The modulated output is delayed for the rising event dead-band time after a direction change from reverse to forward.

### 27.7.1 ASYNCHRONOUS DELAY CHAIN DEAD-BAND DELAY

Asynchronous dead-band delay is determined by the time it takes the input to propagate through a series of delay elements. Each delay element is a nominal five nanoseconds.

For rising event asynchronous dead-band delay set the RDBS bit of the COGxCON0 register and set the COGxDBR register (Register 27-14) value to the desired number of delay elements in the rising event dead-band time.

For falling event asynchronous dead-band delay set the FDBS bit of the COGxCON0 register and set the COGxDBF register (Register 27-15) value to the desired number of delay elements in the falling event dead-band time.

Setting the value to zero disables dead-band delay.

### 27.7.2 SYNCHRONOUS COUNTER DEAD-BAND DELAY

Synchronous counter dead band is timed by counting COG_clock periods from zero up to the value in the dead-band count register. Use Equation 27-1 to calculate dead-band times.

For rising event synchronous dead-band delay clear the RDBS bit of the COGxCON0 register and set the COGxDBR count register value to the number of COG_clock periods in the rising event dead-band time.

For falling event synchronous dead-band delay clear the FDBS bit of the COGxCON0 register and set the COGxDBF count register value to the number of COG_clock periods in the falling event dead-band time.

When the value is zero, dead-band delay is disabled.

### 27.7.3 SYNCHRONOUS COUNTER DEAD-BAND TIME UNCERTAINTY

When the rising and falling events that trigger the dead-band counters come from asynchronous inputs, it creates uncertainty in the synchronous counter dead-band time. The maximum uncertainty is equal to one COG_clock period. Refer to Example 27-1 for more detail.

When event input sources are asynchronous with no phase delay, use the Asynchronous Delay Chain Dead-Band mode to avoid the dead-band time uncertainty.

### 27.7.4 RISING EVENT DEAD BAND

Rising event dead band delays the turn-on of the primary outputs from when complementary outputs are turned off. The rising event dead-band time starts when the rising_ event output goes true.

See **Section 27.7.1 "Asynchronous Delay Chain Dead-Band Delay"** and **Section 27.7.2 "Synchronous Counter Dead-Band Delay"** for more information on setting the rising edge dead-band time.

### 27.7.5 FALLING EVENT DEAD BAND

Falling event dead band delays the turn-on of complementary outputs from when the primary outputs are turned off. The falling event dead-band time starts when the falling_event output goes true.

See **Section 27.7.1 "Asynchronous Delay Chain Dead-Band Delay"** and **Section 27.7.2 "Synchronous Counter Dead-Band Delay"** for more information on setting the rising edge dead-band time.

### 27.7.6 DEAD-BAND OVERLAP

There are two cases of potential dead-band overlap:

- Rising-to-falling
- Falling-to-rising

#### 27.7.6.1 Rising-to-Falling Overlap

In this case, the falling event occurs while the rising event dead-band counter is still counting. When this happens, the primary drives are suppressed and the dead band extends by the falling event dead-band time. At the termination of the extended dead-band time, the complementary drive goes true.

#### 27.7.6.2 Falling-to-Rising Overlap

In this case, the rising event occurs while the falling event dead-band counter is still counting. When this happens, the complementary drive is suppressed and the dead band extends by the rising event dead-band time. At the termination of the extended dead-band time, the primary drive goes true.

## 27.8 Blanking Control

Input blanking is a function whereby the event inputs can be masked or blanked for a short period of time. This is to prevent electrical transients caused by the turn-on/off of power components from generating a false input event.

The COG contains two blanking counters: one triggered by the rising event and the other triggered by the falling_event. The counters are cross coupled with the events they are blanking. The falling event blanking counter is used to blank rising input events and the rising event blanking counter is used to blank falling input events. Once started, blanking extends for the time specified by the corresponding blanking counter.

Blanking is timed by counting COG_clock periods from zero up to the value in the blanking count register. Use Equation 27-1 to calculate blanking times.

### 27.8.1 FALLING EVENT BLANKING OF RISING EVENT INPUTS

The falling event blanking counter inhibits rising event inputs from triggering a rising event. The falling event blanking time starts when the rising_event output drive goes false.

The falling event blanking time is set by the value contained in the COGxBLKF register (Register 27-17). Blanking times are calculated using the formula shown in Equation 27-1.

When the COGxBLKF value is zero, falling event blanking is disabled and the blanking counter output is true, thereby, allowing the event signal to pass straight through to the event trigger circuit.

### 27.8.2 RISING EVENT BLANKING OF FALLING EVENT INPUTS

The rising event blanking counter inhibits falling event inputs from triggering a falling event. The rising event blanking time starts when the falling_event output drive goes false.

The rising event blanking time is set by the value contained in the COGxBLKR register (Register 27-16).

When the COGxBLKR value is zero, rising event blanking is disabled and the blanking counter output is true, thereby, allowing the event signal to pass straight through to the event trigger circuit.

### 27.8.3 BLANKING TIME UNCERTAINTY

When the rising and falling sources that trigger the blanking counters are asynchronous to the COG_clock, it creates uncertainty in the blanking time. The maximum uncertainty is equal to one COG_clock period. Refer to Equation 27-1 and Example 27-1 for more detail.

## 27.9 Phase Delay

It is possible to delay the assertion of either or both the rising event and falling events. This is accomplished by placing a non-zero value in COGxPHR or COGxPHF phase-delay count registers, respectively (Register 27-18 and Register 27-19). Refer to Figure 27-10 for COG operation with CCP1 and phase delay. The delay from the input rising event signal switching to the actual assertion of the events is calculated the same as the dead-band and blanking delays. Refer to Equation 27-1.

When the phase-delay count value is zero, phase delay is disabled and the phase-delay counter output is true, thereby, allowing the event signal to pass straight through to the complementary output driver flop.

### 27.9.1 CUMULATIVE UNCERTAINTY

It is not possible to create more than one COG_clock of uncertainty by successive stages. Consider that the phase-delay stage comes after the blanking stage, the dead-band stage comes after either the blanking or phase-delay stages, and the blanking stage comes after the dead-band stage. When the preceding stage is enabled, the output of that stage is necessarily synchronous with the COG_clock, which removes any possibility of uncertainty in the succeeding stage.

**EQUATION 27-1: PHASE, DEAD-BAND, AND BLANKING TIME CALCULATION**

$$T_{min} = \frac{Count}{F_{COG\_clock}}$$

$$T_{max} = \frac{Count + 1}{F_{COG\_clock}}$$

$$T_{uncertainty} = T_{max} - T_{min}$$

*Also:*

$$T_{uncertainty} = \frac{1}{F_{COG\_clock}}$$

*Where:*

| T | Count |
|---|---|
| Rising Phase Delay | COGxPHR |
| Falling Phase Delay | COGxPHF |
| Rising dead band | COGxDBR |
| Falling dead band | COGxDBF |
| Rising Event Blanking | COGxBLKR |
| Falling Event Blanking | COGxBLKF |

**EXAMPLE 27-1: TIMER UNCERTAINTY**

*Given:*

$$Count = Ah = 10d$$

$$F_{COG\_Clock} = 8\text{MHz}$$

*Therefore:*

$$T_{uncertainty} = \frac{1}{F_{COG\_clock}}$$

$$= \frac{1}{8MHz} = 125ns$$

---

*Proof:*

$$T_{min} = \frac{Count}{F_{COG\_clock}}$$

$$= 125ns \bullet 10d = 1.25\mu s$$

$$T_{max} = \frac{Count + 1}{F_{COG\_clock}}$$

$$= 125ns \bullet (10d + 1)$$

$$= 1.375\mu s$$

*Therefore:*

$$T_{uncertainty} = T_{max} - T_{min}$$

$$= 1.375\mu s - 1.25\mu s$$

$$= 125ns$$

## 27.10 Auto-Shutdown Control

Auto-shutdown is a method to immediately override the COG output levels with specific overrides that allow for safe shutdown of the circuit.

The shutdown state can be either cleared automatically or held until cleared by software. In either case, the shutdown overrides remain in effect until the first rising event after the shutdown is cleared.

### 27.10.1 SHUTDOWN

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

- Software generated
- External Input

#### 27.10.1.1 Software Generated Shutdown

Setting the ASE bit of the COGxASD0 register (Register 27-11) will force the COG into the shutdown state.

When auto-restart is disabled, the shutdown state will persist until the first rising event after the ASE bit is cleared by software.

When auto-restart is enabled, the ASE bit will clear automatically and resume operation on the first rising event after the shutdown input clears. See Figure 27-15 and **Section 27.10.3.2 "Auto-Restart"**.

#### 27.10.1.2 External Shutdown Source

External shutdown inputs provide the fastest way to safely suspend COG operation in the event of a Fault condition. When any of the selected shutdown inputs go true, the output drive latches are reset and the COG outputs immediately go to the selected override levels without software delay.

Any combination of the input sources can be selected to cause a shutdown condition. Shutdown occurs when the selected source is low. Shutdown input sources include:

- Any input pin selected with the COGxINPPS control
- Comparator 1
- Comparator 2
- Comparator 3
- Comparator 4
- CLC2 output/CLC4 output
- Timer2 output/Timer6 output
- Timer4 output/Timer8 output

Shutdown inputs are selected independently with bits of the COGxASD1 register (Register 27-12).

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

### 27.10.2 PIN OVERRIDE LEVELS

The levels driven to the output pins, while the shutdown is active, are controlled by the ASDAC<1:0> and ASDBC<1:0> bits of the COGxASD0 register (Register 27-11). ASDAC<1:0> controls the COGxA and COGxC override levels and ASDBC<1:0> controls the COGxB and COGxD override levels. There are four override options for each output pair:

- Forced low
- Forced high
- Tri-state
- PWM inactive state (same state as that caused by a falling event)

| | |
|---|---|
| **Note:** | The polarity control does not apply to the forced low and high override levels but does apply to the PWM inactive state. |

### 27.10.3 AUTO-SHUTDOWN RESTART

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

- Software controlled
- Auto-restart

The restart method is selected with the ARSEN bit of the COGxASD0 register. Waveforms of a software controlled automatic restart are shown in Figure 27-15.

#### 27.10.3.1 Software Controlled Restart

When the ARSEN bit of the COGxASD0 register is cleared, software must clear the ASE bit to restart COG operation after an auto-shutdown event.

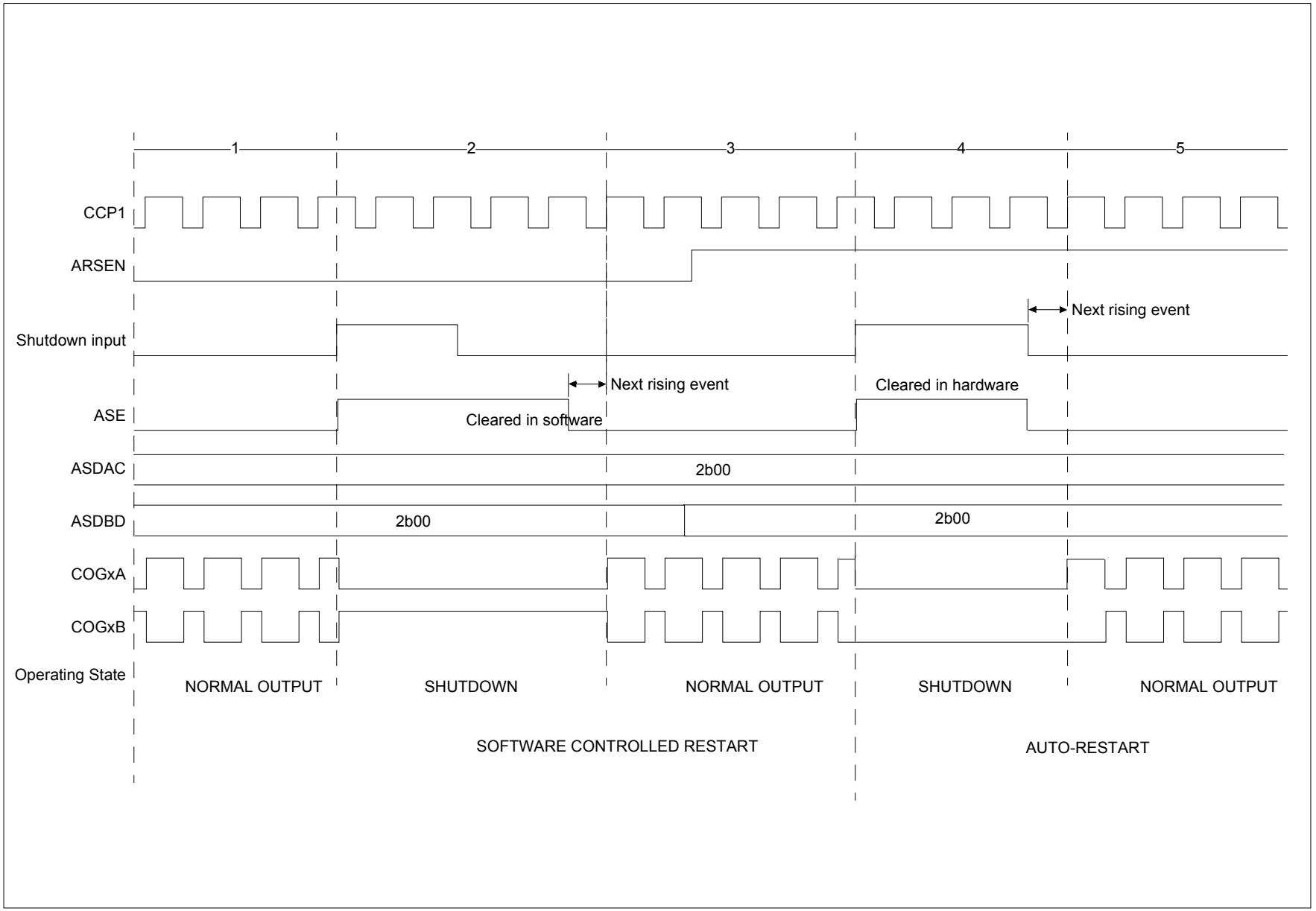
The COG will resume operation on the first rising event after the ASE bit is cleared. Clearing the shutdown state requires all selected shutdown inputs to be false, otherwise, the ASE bit will remain set.

#### 27.10.3.2 Auto-Restart

When the ARSEN bit of the COGxASD0 register is set, the COG will restart from the auto-shutdown state automatically.

The ASE bit will clear automatically and the COG will resume operation on the first rising event after all selected shutdown inputs go false.

**FIGURE 27-15: AUTO-SHUTDOWN WAVEFORM – CCP1 AS RISING AND FALLING EVENT INPUT SOURCE**



## 27.11 Buffer Updates

Changes to the phase, dead-band, and blanking count registers need to occur simultaneously during COG operation to avoid unintended operation that may occur as a result of delays between each register write. This is accomplished with the LD bit of the COGxCON0 register and double buffering of the phase, blanking and dead-band count registers.

Before the COG module is enabled, writing the count registers loads the count buffers without need of the LD bit. However, when the COG is enabled, the count buffer updates are suspended after writing the count registers until after the LD bit is set. When the LD bit is set, the phase, dead-band and blanking register values are transferred to the corresponding buffers synchronous with COG operation. The LD bit is cleared by hardware when the transfer is complete.

## 27.12 Input and Output Pin Selection

The COG has one selection for an input from a device pin. That one input can be used as rising and falling event source or a fault source. The COGxINPPS register is used to select the pin. Refer to registers xxxPPS (Register 12-1) and RxyPPS (Register 12-2).

The pin PPS control registers are used to enable the COG outputs. Any combination of outputs to pins is possible including multiple pins for the same output. See the RxyPPS control register and **Section 12.2 "PPS Outputs"** for more details.

## 27.13 Operation During Sleep

The COG continues to operate in Sleep provided that the COG_clock, rising event, and falling event sources remain active.

The HFINTSOC remains active during Sleep when the COG is enabled and the HFINTOSC is selected as the COG_clock source.

## 27.14 Configuring the COG

The following steps illustrate how to properly configure the COG to ensure a synchronous start with the rising event input:

1. If a pin is to be used for the COG fault or event input, use the COGxINPPS register to configure the desired pin.
2. Clear all ANSEL register bits associated with pins that are used for COG functions.
3. Ensure that the TRIS control bits corresponding to the COG outputs to be used are set so that all are configured as inputs. The COG module will enable the output drivers as needed later.
4. Clear the EN bit, if not already cleared.
5. Set desired dead-band times with the COGxDBR and COGxDBF registers and select the source with the RDBS and FDBS bits of the COGxCON1 register.
6. Set desired blanking times with the COGxBLKR and COGxBLKF registers.
7. Set desired phase delay with the COGxPHR and COGxPHF registers.
8. Select the desired shutdown sources with the COGxASD1 register.
9. Setup the following controls in COGxASD0 auto-shutdown register:
   - Select both output override controls to the desired levels (this is necessary, even if not using auto-shutdown because start-up will be from a shutdown state).
   - Set the ASE bit and clear the ARSEN bit.
10. Select the desired rising and falling event sources with the COGxRIS0, COGxRIS1, COGxFIS0, and COGxFIS1 registers.
11. Select the desired rising and falling event modes with the COGxRSIM0, COGxRSIMI1, COGxFSIM0, and COGxFSIM1 registers.
12. Configure the following controls in the COGxCON1 register:
    - Set the polarity for each output
    - Select the desired dead-band timing sources
13. Configure the following controls in the COGxCON0 register:
    - Set the desired operating mode
    - Select the desired clock source
14. If one of the steering modes is selected then configure the following controls in the COGxSTR register:
    - Set the steering bits of the outputs to be used.
    - Set the desired static levels.
15. Set the EN bit.
16. Set the pin PPS controls to direct the COG outputs to the desired pins.
17. If auto-restart is to be used, set the ARSEN bit and the ASE will be cleared automatically. Otherwise, clear the ASE bit to start the COG.

## 27.15 Register Definitions: COG Control

Long bit name prefixes for the COG peripherals are shown in Table 27-4. Refer to **Section 1.1 "Register and Bit naming conventions"** for more information

**TABLE 27-4:**

| Peripheral | Bit Name Prefix |
|---|---|
| COG1 | G1 |
| COG2 | G2 |
| COG3 | G3 |

**REGISTER 27-1: COGxCON0: COG CONTROL REGISTER 0**

| R/W-0/0 | R/W-0/0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| EN | LD | — | CS<1:0> | | MD<2:0> | | |
| bit 7 | | | | | | | bit 0 |

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

bit 7 **EN:** COGx Enable bit
- 1 = Module is enabled
- 0 = Module is disabled

bit 6 **LD:** COGx Load Buffers bit
- 1 = Phase, blanking, and dead-band buffers to be loaded with register values on next input events
- 0 = Register to buffer transfer is complete

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

bit 4-3 **CS<1:0>:** COGx Clock Selection bits
- 11 = Reserved. Do not use.
- 10 = COG_clock is HFINTOSC (stays active during Sleep)
- 01 = COG_clock is FOSC
- 00 = COG_clock is FOSC/4

bit 2-0 **MD<2:0>:** COGx Mode Selection bits
- 11x = Reserved. Do not use.
- 101 = COG outputs operate in Push-Pull mode
- 100 = COG outputs operate in Half-Bridge mode
- 011 = COG outputs operate in Reverse Full-Bridge mode
- 010 = COG outputs operate in Forward Full-Bridge mode
- 001 = COG outputs operate in synchronous steered PWM mode
- 000 = COG outputs operate in steered PWM mode

**REGISTER 27-2: COGxCON1: COG CONTROL REGISTER 1**

| R/W-0/0 | R/W-0/0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| RDBS | FDBS | — | — | POLD | POLC | POLB | POLA |
| 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 **RDBS:** COGx Rising Event Dead-band Timing Source Select bit
1 = Delay chain and COGxDBR are used for dead-band timing generation
0 = COGx_clock and COGxDBR are used for dead-band timing generation

bit 6 **FDBS:** COGx Falling Event Dead-band Timing Source select bit
1 = Delay chain and COGxDBF are used for dead-band timing generation
0 = COGx_clock and COGxDBF are used for dead-band timing generation

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

bit 3 **POLD:** COGxD Output Polarity Control bit
1 = Active level of COGxD output is low
0 = Active level of COGxD output is high

bit 2 **POLC:** COGxC Output Polarity Control bit
1 = Active level of COGxC output is low
0 = Active level of COGxC output is high

bit 1 **POLB:** COGxB Output Polarity Control bit
1 = Active level of COGxB output is low
0 = Active level of COGxB output is high

bit 0 **POLA:** COGxA Output Polarity Control bit
1 = Active level of COGxA output is low
0 = Active level of COGxA output is high

### REGISTER 27-3: COGxRIS0: COG RISING EVENT INPUT SELECTION 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 |
|---|---|---|---|---|---|---|---|
| RIS7 | RIS6 | RIS5 | RIS4 | RIS3 | RIS2 | RIS1 | RIS0 |
| 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 7-0 **RIS<7:0>:** Source Rising Event Input <n> Source Enable bits(1). See Table 27-5.
1 = Source <n> output is enabled as a rising event input
0 = Source <n> output has no effect on the rising event

**Note 1:** Any combination of <n> bits can be selected.

### REGISTER 27-4: COGxRIS1: COG RISING EVENT INPUT SELECTION 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 |
|---|---|---|---|---|---|---|---|
| RIS15 | RIS14 | RIS13 | RIS12 | RIS11 | RIS10 | RIS9 | RIS8 |
| 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 15-8 **RIS<15:8>:** COGx Rising Event Input <n> Source Enable bits(1). See Table 27-5.
1 = Source <n> output is enabled as a rising event input
0 = Source <n> output has no effect on the rising event

**Note 1:** Any combination of <n> bits can be selected.

### TABLE 27-5: RISING/FALLING EVENT INPUT SOURCES

| Bit <n> | COG1 | COG2 | COG3 |
|---|---|---|---|
| 15 | LC4_out | LC4_out | LC4_out |
| 14 | LC3_out | LC3_out | LC3_out |
| 13 | LC2_out | LC2_out | LC2_out |
| 12 | LC1_out | LC1_out | LC1_out |
| 11 | MD1_out | MD2_out | MD3_out |
| 10 | PWM6_output | PWM6_output | Reserved |
| 9 | PWM5_output | PWM5_output | PWM11_output |
| 8 | PWM4_output | PWM4_output | Reserved |
| 7 | PWM3_output | PWM3_output | PWM9_output |
| 6 | CCP2_out | CCP2_out | CCP3_out |
| 5 | CCP1_out | CCP1_out | CCP1_out |
| 4 | sync_CM4_out | sync_CM4_out | sync_CM6_out |
| 3 | sync_CM3_out | sync_CM3_out | sync_CM5_out |
| 2 | sync_CM2_out | sync_CM2_out | sync_CM2_out |
| 1 | sync_CM1_out | sync_CM1_out | sync_CM1_out |
| 0 | Pin selected with COG1PPS | Pin selected with COG2PPS | Pin selected with COG3PPS |

**REGISTER 27-5: COGxRSIM0: COG RISING EVENT SOURCE INPUT MODE 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 |
|---|---|---|---|---|---|---|---|
| RSIM7 | RSIM6 | RSIM5 | RSIM4 | RSIM3 | RSIM2 | RSIM1 | RSIM0 |
| 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 7-0 **RSIM<7:0>:** Rising Event Input Source <n> Mode bits(1). See Table 27-5.
RIS<n> = 1:
1 = Source <n> output low-to-high transition will cause a rising event after rising event phase delay
0 = Source <n> output high level will cause an immediate rising event
RIS<n> = 0:
Source <n> output has no effect on rising event

**Note 1:** Any combination of <n> bits can be selected.

**REGISTER 27-6: COGxRSIM1: COG RISING EVENT SOURCE INPUT MODE 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 |
|---|---|---|---|---|---|---|---|
| RSIM15 | RSIM14 | RSIM13 | RSIM12 | RSIM11 | RSIM10 | RSIM9 | RSIM8 |
| 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 15-8 **RSIM<15:8>:** Rising Event Input Source <n> Mode bits(1). See Table 27-5.
RIS<n> = 1:
1 = Source <n> output low-to-high transition will cause a rising event after rising event phase delay
0 = Source <n> output high level will cause an immediate rising event
RIS<n> = 0:
Source <n> output has no effect on rising event

**Note 1:** Any combination of <n> bits can be selected.

**REGISTER 27-7: COGxFIS0: COG FALLING EVENT INPUT SELECTION 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 |
|---|---|---|---|---|---|---|---|
| FIS7 | FIS6 | FIS5 | FIS4 | FIS3 | FIS2 | FIS1 | FIS0 |
| 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 7-0 **FIS<7:0>:** Falling Event Input Source <n> Enable bits(1). See Table 27-5.

1 = Source <n> output is enabled as a falling event input
0 = Source <n> output has no effect on the falling event

**Note 1:** Any combination of <n> bits can be selected.

**REGISTER 27-8: COGxFIS1: COG FALLING EVENT INPUT SELECTION 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 |
|---|---|---|---|---|---|---|---|
| FIS15 | FIS14 | FIS13 | FIS12 | FIS11 | FIS10 | FIS9 | FIS8 |
| 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 15-8 **FIS<15:8>:** Falling Event Input Source <n> Enable bits(1). See Table 27-5.

1 = Source <n> output is enabled as a falling event input
0 = Source <n> output has no effect on the falling event

**Note 1:** Any combination of <n> bits can be selected.

**REGISTER 27-9: COGxFSIM0: COG FALLING EVENT SOURCE INPUT MODE 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 |
|---|---|---|---|---|---|---|---|
| FSIM7 | FSIM6 | FSIM5 | FSIM4 | FSIM3 | FSIM2 | FSIM1 | FSIM0 |
| 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 7-0 **FSIM<7:0>:** Falling Event Input Source <n> Mode bits(1). See Table 27-5.
FIS<n> = 1:
1 = Source <n> output high-to-low transition will cause a falling event after falling event phase delay
0 = Source <n> output low level will cause an immediate falling event
FIS<n> = 0:
Source <n> output has no effect on falling event

**Note 1:** Any combination of <n> bits can be selected.

**REGISTER 27-10: COGxFSIM1: COG FALLING EVENT SOURCE INPUT MODE 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 |
|---|---|---|---|---|---|---|---|
| FSIM15 | FSIM14 | FSIM13 | FSIM12 | FSIM11 | FSIM10 | FSIM9 | FSIM8 |
| 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 15-8 **FSIM<15:8>:** Falling Event Input Source <n> Mode bits(1). See Table 27-5.
FIS<n> = 1:
1 = Source <n> output high-to-low transition will cause a falling event after falling event phase delay
0 = Source <n> output low level will cause an immediate falling event
FIS<n> = 0:
Source <n> output has no effect on falling event

**Note 1:** Any combination of <n> bits can be selected.

**REGISTER 27-11: COGxASD0: COG AUTO-SHUTDOWN CONTROL REGISTER 0**

| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-1/1 | R/W-0/0 | R/W-1/1 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| ASE | ARSEN | ASDBD<1:0> | | ASDAC<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 **ASE:** Auto-Shutdown Event Status bit
- 1 = COG is in the shutdown state
- 0 = COG is either not in the shutdown state or will exit the shutdown state on the next rising event

bit 6 **ARSEN:** Auto-Restart Enable bit
- 1 = Auto-restart is enabled
- 0 = Auto-restart is disabled

bit 5-4 **ASDBD<1:0>:** COGxB and COGxD Auto-shutdown Override Level Select bits
- 11 = A logic '1' is placed on COGxB and COGxD when shutdown is active
- 10 = A logic '0' is placed on COGxB and COGxD when shutdown is active
- 01 = COGxB and COGxD are tri-stated when shutdown is active
- 00 = The inactive state of the pin, including polarity, is placed on COGxB and COGxD when shutdown is active

bit 3-2 **ASDAC<1:0>:** COGxA and COGxC Auto-shutdown Override Level Select bits
- 11 = A logic '1' is placed on COGxA and COGxC when shutdown is active
- 10 = A logic '0' is placed on COGxA and COGxC when shutdown is active
- 01 = COGxA and COGxC are tri-stated when shutdown is active
- 00 = The inactive state of the pin, including polarity, is placed on COGxA and COGxC when shutdown is active

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

**REGISTER 27-12: COGxASD1: COG AUTO-SHUTDOWN 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 |
|---|---|---|---|---|---|---|---|
| AS7E | AS6E | AS5E | AS4E | AS3E | AS2E | AS1E | AS0E |
| 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 7-0 **AS<7:0>E:** Auto-shutdown Source <n> Enable bits(1). See Table 27-6.
1 = COGx is shutdown when source <n> output is low
0 = Source <n> has no effect on shutdown

**Note 1:** Any combination of <n> bits can be selected.

**TABLE 27-6: AUTO-SHUTDOWN SOURCES**

| Bit <n> | COG1 | COG2 | COG3 |
|---|---|---|---|
| 7 | TMR4_postscaled(1) | TMR4_postscaled(1) | TMR8_postscaled(1) |
| 6 | TMR2_postscaled(1) | TMR2_postscaled(1) | TMR6_postscaled(1) |
| 5 | LC2_out | LC2_out | LC4_out |
| 4 | sync_CM4_out | sync_CM4_out | sync_CM6_out |
| 3 | sync_CM3_out | sync_CM3_out | sync_CM5_out |
| 2 | sync_CM2_out | sync_CM2_out | sync_CM2_out |
| 1 | sync_CM1_out | sync_CM1_out | sync_CM1_out |
| 0 | Pin selected by COG1PPS | Pin selected by COG2PPS | Pin selected by COG3PPS |

**Note 1:** Shutdown when source is high.

**REGISTER 27-13: COGxSTR: COG STEERING CONTROL REGISTER 1(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 |
|---|---|---|---|---|---|---|---|
| SDATD | SDATC | SDATB | SDATA | STRD | STRC | STRB | STRA |
| 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 **SDATD:** COGxD Static Output Data bit
- 1 = COGxD static data is high
- 0 = COGxD static data is low

bit 6 **SDATC:** COGxC Static Output Data bit
- 1 = COGxC static data is high
- 0 = COGxC static data is low

bit 5 **SDATB:** COGxB Static Output Data bit
- 1 = COGxB static data is high
- 0 = COGxB static data is low

bit 4 **SDATA:** COGxA Static Output Data bit
- 1 = COGxA static data is high
- 0 = COGxA static data is low

bit 3 **STRD:** COGxD Steering Control bit
- 1 = COGxD output has the COGxD waveform with polarity control from POLD bit
- 0 = COGxD output is the static data level determined by the SDATD bit

bit 2 **STRC:** COGxC Steering Control bit
- 1 = COGxC output has the COGxC waveform with polarity control from POLC bit
- 0 = COGxC output is the static data level determined by the SDATC bit

bit 1 **STRB:** COGxB Steering Control bit
- 1 = COGxB output has the COGxB waveform with polarity control from POLB bit
- 0 = COGxB output is the static data level determined by the SDATB bit

bit 0 **STRA:** COGxA Steering Control bit
- 1 = COGxA output has the COGxA waveform with polarity control from POLA bit
- 0 = COGxA output is the static data level determined by the SDATA bit

**Note 1:** Steering is active only when the MD<1:0> bits of the COGxCON0 register = 00x. (See Register 27-1).

**REGISTER 27-14: COGxDBR: COG RISING EVENT DEAD-BAND COUNT 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 |
|---|---|---|---|---|---|---|---|
| | | DBR<5:0> | | | | | |
| bit 7 | | | | | | | bit 0 |

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

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

bit 5-0 **DBR<5:0>:** Rising Event Dead-Band Count Value bits

RDBS = 0:

= Number of COGx clock periods to delay primary output after rising event

RDBS = 1:

= Number of delay chain element periods to delay primary output after rising event

**REGISTER 27-15: COGxDBF: COG FALLING EVENT DEAD-BAND COUNT 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 |
|---|---|---|---|---|---|---|---|
| | | DBF<5:0> | | | | | |
| bit 7 | | | | | | | bit 0 |

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

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

bit 5-0 **DBF<5:0>:** Falling Event Dead-Band Count Value bits

FDBS = 0:

= Number of COGx clock periods to delay complementary output after falling event input

FDBS = 1:

= Number of delay chain element periods to delay complementary output after falling event input

**REGISTER 27-16: COGxBLKR: COG RISING EVENT BLANKING COUNT REGISTER**

| U-0 | U-0 | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
|---|---|---|---|---|---|---|---|
| | | BLKR<5:0> | | | | | |
| bit 7 | | | | | | | bit 0 |

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

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

bit 5-0 **BLKR<5:0>:** Rising Event Blanking Count Value bits
= Number of COGx clock periods to inhibit falling event inputs

**REGISTER 27-17: COGxBLKF: COG FALLING EVENT BLANKING COUNT 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 |
|---|---|---|---|---|---|---|---|
| | | BLKF<5:0> | | | | | |
| bit 7 | | | | | | | bit 0 |

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

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

bit 5-0 **BLKF<5:0>:** Falling Event Blanking Count Value bits
= Number of COGx clock periods to inhibit rising event inputs

**REGISTER 27-18: COGxPHR: COG RISING EVENT PHASE DELAY COUNT 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 |
|---|---|---|---|---|---|---|---|
| — | — | PHR<5:0> | | | | | |
| bit 7 | | | | | | | bit 0 |

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

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

bit 5-0 **PHR<5:0>:** Rising Event Phase Delay Count Value bits
= Number of COGx clock periods to delay rising event

**REGISTER 27-19: COGxPHF: COG FALLING EVENT PHASE DELAY COUNT 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 |
|---|---|---|---|---|---|---|---|
| — | — | PHF<5:0> | | | | | |
| bit 7 | | | | | | | bit 0 |

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

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

bit 5-0 **PHF<5:0>:** Falling Event Phase Delay Count Value bits
= Number of COGx clock periods to delay falling event

**TABLE 27-7: SUMMARY OF REGISTERS ASSOCIATED WITH COGx**

| 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 | 153 |
| ANSELB | — | — | ANSB5 | ANSB4 | ANSB3 | ANSB2 | ANSB1 | ANSB0 | 158 |
| ANSELC | ANSC7 | ANSC6 | ANSC5 | ANSC4 | ANSC3 | ANSC2 | — | — | 163 |
| COGxASD0 | ASE | ARSEN | ASDBD<1:0> | | ASDAC<1:0> | | — | — | 346 |
| COGxASD1 | AS7E | AS6E | AS5E | AS4E | AS3E | AS2E | AS1E | AS0E | 347 |
| COGxBLKR | — | — | BLKR<5:0> | | | | | | 350 |
| COGxBLKF | — | — | BLKF<5:0> | | | | | | 350 |
| COGxCON0 | EN | LD | — | CS<1:0> | | MD<2:0> | | | 340 |
| COGxCON1 | RDBS | FDBS | — | — | POLD | POLC | POLB | POLA | 341 |
| COGxDBR | — | — | DBR<5:0> | | | | | | 349 |
| COGxDBF | — | — | DBF<5:0> | | | | | | 349 |
| COGxFIS0 | FIS7 | FIS6 | FIS5 | FIS4 | FIS3 | FIS2 | FIS1 | FIS0 | 344 |
| COGxFIS1 | FIS15 | FIS14 | FIS13 | FIS12 | FIS11 | FIS10 | FIS9 | FIS8 | 344 |
| COGxFSIM0 | FSIM7 | FSIM6 | FSIM5 | FSIM4 | FSIM3 | FSIM2 | FSIM1 | FSIM0 | 345 |
| COGxFSIM1 | FSIM15 | FSIM14 | FSIM13 | FSIM12 | FSIM11 | FSIM10 | FSIM9 | FSIM8 | 345 |
| COGxPHR | — | — | PHR<5:0> | | | | | | 351 |
| COGxPHF | — | — | PHF<5:0> | | | | | | 351 |
| COGxPPS | — | — | COG1PPS<5:0> | | | | | | 172, 174 |
| COGxRIS0 | RIS7 | RIS6 | RIS5 | RIS4 | RIS3 | RIS2 | RIS1 | RIS0 | 342 |
| COGxRIS1 | RIS15 | RIS14 | RIS13 | RIS12 | RIS11 | RIS10 | RIS9 | RIS8 | 342 |
| COGxRSIM0 | RSIM7 | RSIM6 | RSIM5 | RSIM4 | RSIM3 | RSIM2 | RSIM1 | RSIM0 | 343 |
| COGxRSIM1 | RSIM15 | RSIM14 | RSIM13 | RSIM12 | RSIM11 | RSIM10 | RSIM9 | RSIM8 | 343 |
| COGxSTR | SDATD | SDATC | SDATB | SDATA | STRD | STRC | STRB | STRA | 348 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 108 |
| RxyPPS | — | — | RxyPPS<5:0> | | | | | | 172 |

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

## 28.0 CONFIGURABLE LOGIC CELL (CLC)

The Configurable Logic Cell (CLCx) provides programmable logic that operates outside the speed limitations of software execution. The logic cell takes up to 32 input signals and, through the use of configurable gates, reduces the 32 inputs to four logic lines that drive one of eight selectable single-output logic functions.

Input sources are a combination of the following:

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

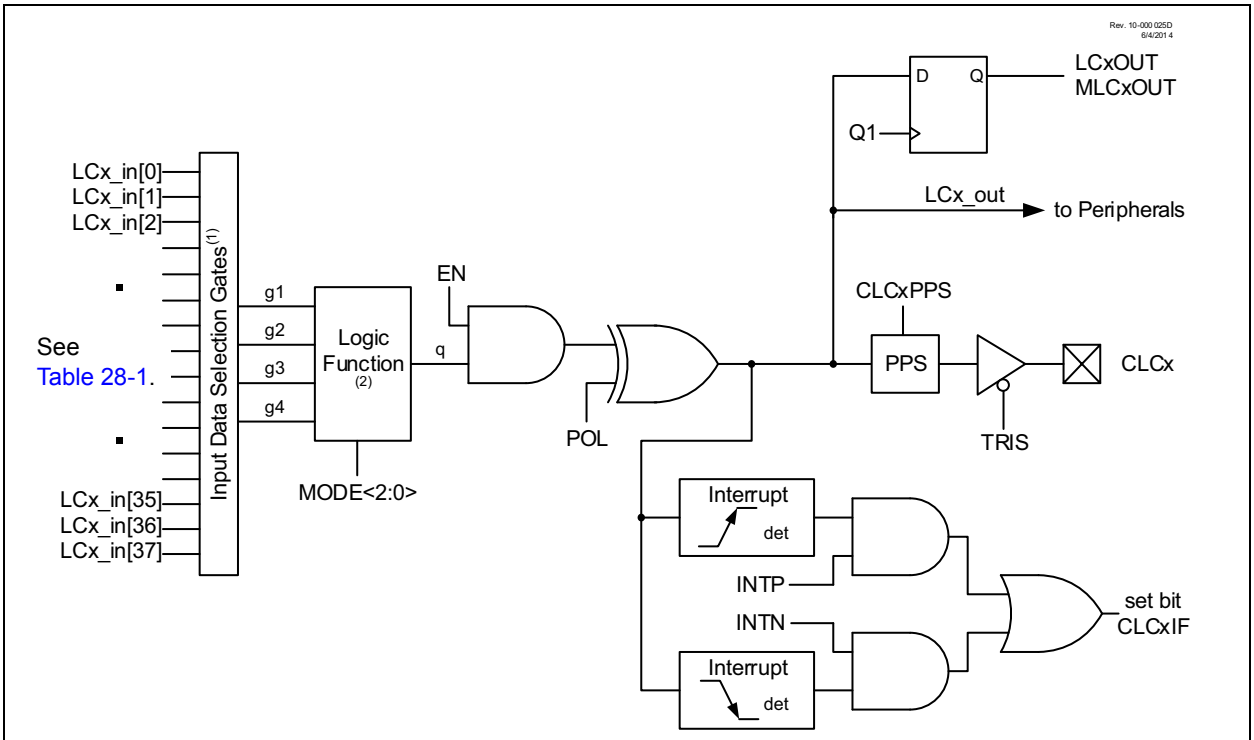
The output can be directed internally to peripherals and to an output pin.

Refer to Figure 28-1 for a simplified diagram showing signal flow through the CLCx.

Possible configurations include:

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

**FIGURE 28-1: CLCx SIMPLIFIED BLOCK DIAGRAM**




**Note 1:** See Figure 28-2: Input Data Selection and Gating.
**2:** See Figure 28-3: Programmable Logic Functions.

## 28.1 CLCx Setup

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

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

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

### 28.1.1 DATA SELECTION

There are 32 signals available as inputs to the configurable logic. Four 32-input multiplexers are used to select the inputs to pass on to the next stage.

Data selection is through four multiplexers as indicated on the left side of Figure 28-2. Data inputs in the figure are identified by a generic numbered input name.

Table 28-1 correlates the generic input name to the actual signal for each CLC module. The column labeled dy indicates the MUX selection code for the selected data input. DxS is an abbreviation for the MUX select input codes: D1S<4:0> through D4S<4:0>.

Data inputs are selected with CLCxSEL0 through CLCxSEL3 registers (Register 28-3 through Register 28-6).

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

**TABLE 28-1: CLCx DATA INPUT SELECTION**

| Data Input | dy DxS | CLCx |
|---|---|---|
| LCx_in[54] | 110110 | MD1_OUT OR MD2_OUT OR MD3_OUT |
| LCx_in[53] | 110101 | FOSC |
| LCx_in[52] | 110100 | HFINTOSC |
| LCx_in[51] | 110011 | LFINTOSC |
| LCx_in[50] | 110010 | FRC (ADC RC clock) |
| LCx_in[49] | 110001 | IOCIF set |
| LCx_in[48] | 110000 | Timer8_postscaled |
| LCx_in[47] | 101111 | Timer6_postscaled |
| LCx_in[46] | 101110 | Timer4_postscaled |
| LCx_in[45] | 101101 | Timer2_postscaled |
| LCx_in[44] | 101100 | Timer5 overflow |
| LCx_in[43] | 101011 | Timer3 overflow |
| LCx_in[42] | 101010 | Timer1 overflow |
| LCx_in[41] | 101001 | Timer0 overflow |
| LCx_in[40] | 101000 | EUSART RX |
| LCx_in[39] | 100111 | EUSART TX |
| LCx_in[38] | 100110 | ZCD1_output |
| LCx_in[37] | 100101 | MSSP1 SDO/SDA |
| LCx_in[36] | 100100 | MSSP1 SCL/SCK |

**TABLE 28-1: CLCx DATA INPUT SELECTION**

| Data Input | dy DxS | CLCx |
|---|---|---|
| LCx_in[35] | 100011 | Reserved |
| LCx_in[34] | 100010 | PWM11_out |
| LCx_in[33] | 100001 | PWM6_out |
| LCx_in[32] | 100000 | PWM5_out |
| LCx_in[31] | 011111 | Reserved |
| LCx_in[30] | 011110 | PWM9_out |
| LCx_in[29] | 011101 | PWM4_out |
| LCx_in[28] | 011100 | PWM3_out |
| LCx_in[27] | 011011 | Reserved |
| LCx_in[26] | 011010 | CCP7_out |
| LCx_in[25] | 011001 | CCP2_out |
| LCx_in[24] | 011000 | CCP1_out |
| LCx_in[23] | 010111 | Reserved |
| LCx_in[22] | 010110 | Reserved |
| LCx_in[21] | 010101 | COG3B |
| LCx_in[20] | 010100 | COG3A |
| LCx_in[19] | 010011 | COG2B |
| LCx_in[18] | 010010 | COG2A |
| LCx_in[17] | 010001 | COG1B |
| LCx_in[16] | 010000 | COG1A |
| LCx_in[15] | 001111 | Reserved |
| LCx_in[14] | 001110 | Reserved |
| LCx_in[13] | 001101 | sync_C6OUT |
| LCx_in[12] | 001100 | sync_C5OUT |
| LCx_in[11] | 001011 | sync_C4OUT |
| LCx_in[10] | 001010 | sync_C3OUT |
| LCx_in[9] | 001001 | sync_C2OUT |
| LCx_in[8] | 001000 | sync_C1OUT |
| LCx_in[7] | 000111 | LC4_out from the CLC4 |
| LCx_in[6] | 000110 | LC3_out from the CLC3 |
| LCx_in[5] | 000101 | LC2_out from the CLC2 |
| LCx_in[4] | 000100 | LC1_out from the CLC1 |
| LCx_in[3] | 000011 | CLCIN3 pin input selected in CLCIN3PPS register |
| LCx_in[2] | 000010 | CLCIN2 pin input selected in CLCIN2PPS register |
| LCx_in[1] | 000001 | CLCIN1 pin input selected in CLCIN1PPS register |
| LCx_in[0] | 000000 | CLCIN0 pin input selected in CLCIN0PPS register |

### 28.1.2 DATA GATING

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

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

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

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

Table 28-2 summarizes the basic logic that can be obtained in gate 1 by using the gate logic select bits. The table shows the logic of four input variables, but each gate can be configured to use less than four. If no inputs are selected, the output will be zero or one, depending on the gate output polarity bit.

**TABLE 28-2: DATA GATING LOGIC**

| CLCxGLS0 | G1POL | Gate Logic |
|---|---|---|
| 0x55 | 1 | AND |
| 0x55 | 0 | NAND |
| 0xAA | 1 | NOR |
| 0xAA | 0 | OR |
| 0x00 | 0 | Logic 0 |
| 0x00 | 1 | Logic 1 |

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

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

- Gate 1: CLCxGLS0 (Register 28-7)
- Gate 2: CLCxGLS1 (Register 28-8)
- Gate 3: CLCxGLS2 (Register 28-9)
- Gate 4: CLCxGLS3 (Register 28-10)

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

Data gating is indicated in the right side of Figure 28-2. Only one gate is shown in detail. The remaining three gates are configured identically with the exception that the data enables correspond to the enables for that gate.

### 28.1.3 LOGIC FUNCTION

There are eight available logic functions including:

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

Logic functions are shown in Figure 28-3. Each logic function has four inputs and one output. The four inputs are the four data gate outputs of the previous stage. The output is fed to the inversion stage and from there to other peripherals, an output pin, and back to the CLCx itself.

### 28.1.4 OUTPUT POLARITY

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

### 28.1.5 CLCx SETUP STEPS

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

- Disable CLCx by clearing the EN bit.
- Select desired inputs using CLCxSEL0 through CLCxSEL3 registers (See Table 28-1).
- Clear any associated ANSEL bits.
- Set all TRIS bits associated with inputs.
- Clear all TRIS bits associated with outputs.
- Enable the chosen inputs through the four gates using CLCxGLS0, CLCxGLS1, CLCxGLS2, and CLCxGLS3 registers.
- Select the gate output polarities with the POLy bits of the CLCxPOL register.
- Select the desired logic function with the MODE<2:0> bits of the CLCxCON register.
- Select the desired polarity of the logic output with the POL bit of the CLCxPOL register. (This step may be combined with the previous gate output polarity step).
- If driving a device pin, set the desired pin PPS control register and also clear the TRIS bit corresponding to that output.
- If interrupts are desired, configure the following bits:
  - Set the INTP bit in the CLCxCON register for rising event.
  - Set the INTN bit in the CLCxCON register for falling event.
  - Set the CLCxIE bit of the associated PIE registers.
  - Set the GIE and PEIE bits of the INTCON register.
- Enable the CLCx by setting the EN bit of the CLCxCON register.

## 28.2 CLCx Interrupts

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

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

To fully enable the interrupt, set the following bits:

- ON bit of the CLCxCON register
- CLCxIE bit of the associated PIE registers
- INTP bit of the CLCxCON register (for a rising edge detection)
- INTN bit of the CLCxCON register (for a falling edge detection)
- PEIE and GIE bits of the INTCON register

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

## 28.3 Output Mirror Copies

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

## 28.4 Effects of a Reset

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

## 28.5 Operation During Sleep

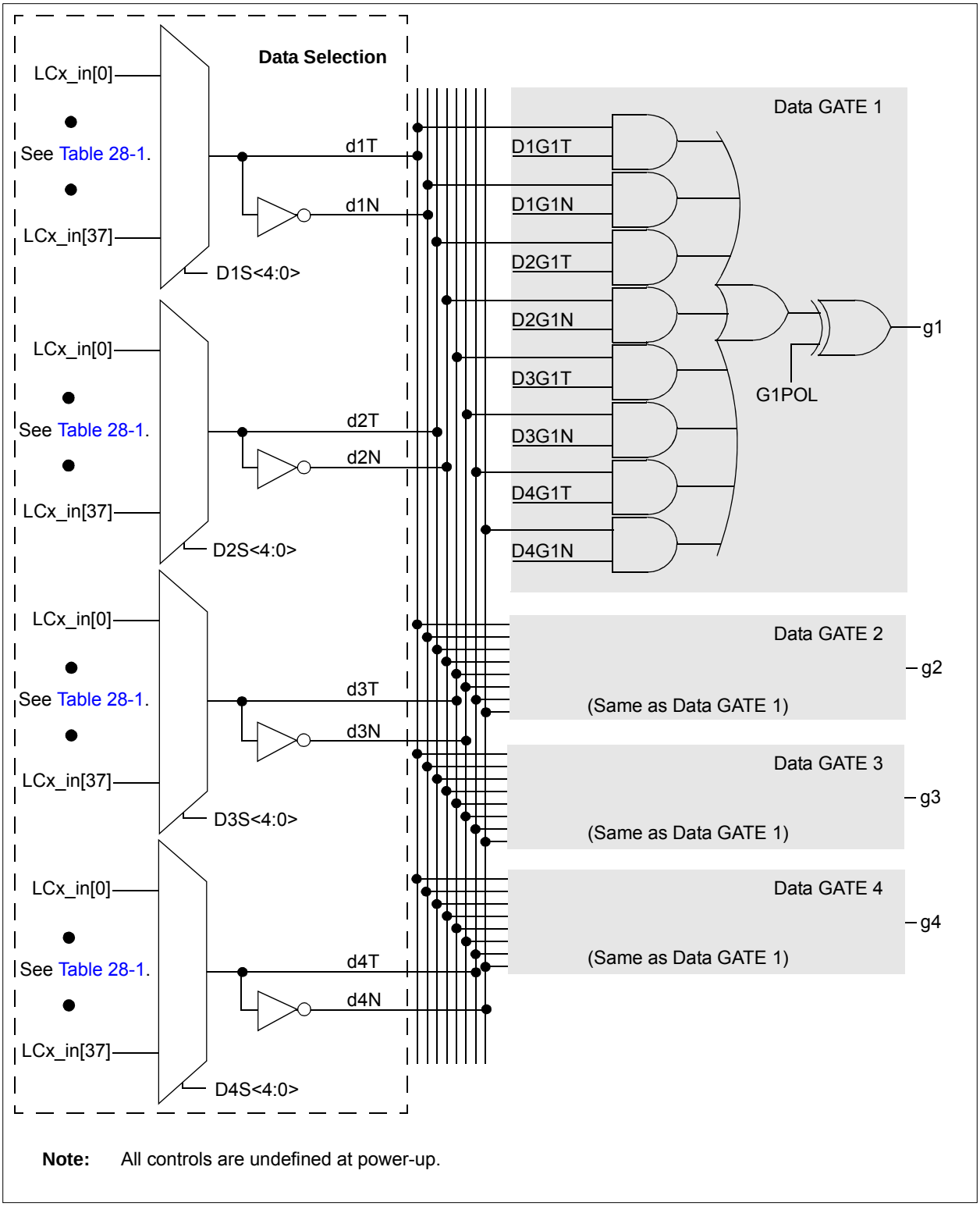
The CLC module operates independently from the system clock and will continue to run during Sleep, provided that the input sources selected remain active.

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

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

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

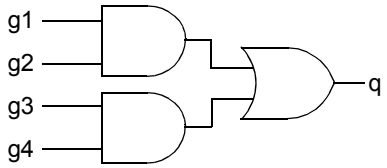
**FIGURE 28-2: INPUT DATA SELECTION AND GATING**




**Note:** All controls are undefined at power-up.

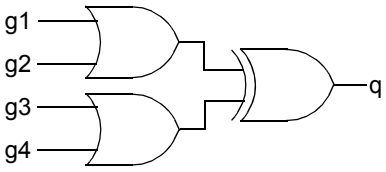
**FIGURE 28-3: PROGRAMMABLE LOGIC FUNCTIONS**

**AND – OR**



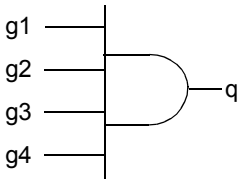

MODE<2:0>= 000

**OR – XOR**



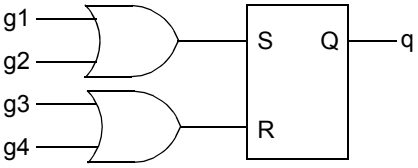

MODE<2:0>= 001

**4-Input AND**



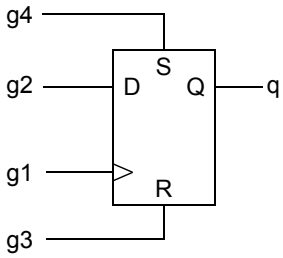

MODE<2:0>= 010

**S-R Latch**



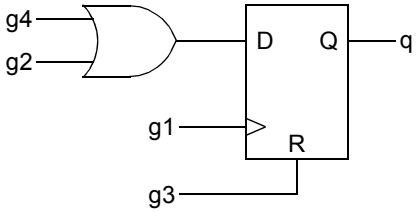

MODE<2:0>= 011

**1-Input D Flip-Flop with S and R**



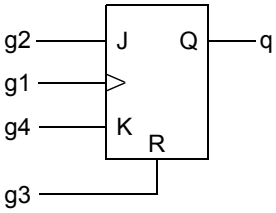

MODE<2:0>= 100

**2-Input D Flip-Flop with R**



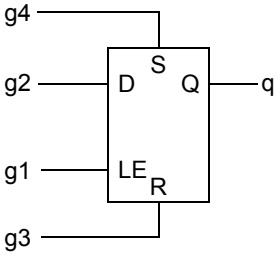

MODE<2:0>= 101

**J-K Flip-Flop with R**




MODE<2:0>= 110

**1-Input Transparent Latch with S and R**




MODE<2:0>= 111

## 28.6 Register Definitions: CLC Control

Long bit name prefixes for the CLC peripherals are shown in Table 28-3. Refer to **Section 1.1 "Register and Bit naming conventions"** for more information

**TABLE 28-3:**

| Peripheral | Bit Name Prefix |
|---|---|
| CLC1 | LC1 |
| CLC2 | LC2 |
| CLC3 | LC3 |
| CLC4 | LC4 |

**REGISTER 28-1: CLCxCON: CONFIGURABLE LOGIC CELL CONTROL REGISTER**

| R/W-0/0 | U-0 | R-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| EN | — | OUT | INTP | INTN | MODE<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 **EN:** Configurable Logic Cell Enable bit
1 = Configurable logic cell is enabled and mixing input signals
0 = Configurable logic cell is disabled and has logic zero output

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

bit 5 **OUT:** Configurable Logic Cell Data Output bit
Read-only: logic cell output data, after POL; sampled from lcx_out wire.

bit 4 **INTP:** Configurable Logic Cell Positive Edge Going Interrupt Enable bit
1 = CLCxIF will be set when a rising edge occurs on lcx_out
0 = CLCxIF will not be set

bit 3 **INTN:** Configurable Logic Cell Negative Edge Going Interrupt Enable bit
1 = CLCxIF will be set when a falling edge occurs on lcx_out
0 = CLCxIF will not be set

bit 2-0 **MODE<2:0>:** Configurable Logic Cell Functional Mode bits
111 = Cell is 1-input transparent latch with S and R
110 = Cell is J-K flip-flop with R
101 = Cell is 2-input D flip-flop with R
100 = Cell is 1-input D flip-flop with S and R
011 = Cell is S-R latch
010 = Cell is 4-input AND
001 = Cell is OR-XOR
000 = Cell is AND-OR

**REGISTER 28-2: CLCxPOL: SIGNAL POLARITY CONTROL REGISTER**

| R/W-0/0 | U-0 | U-0 | U-0 | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
|---|---|---|---|---|---|---|---|
| POL | — | — | — | G4POL | G3POL | G2POL | G1POL |
| 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 **POL:** LCOUT Polarity Control bit
- 1 = The output of the logic cell is inverted
- 0 = The output of the logic cell is not inverted

bit 6-4 **Unimplemented:** Read as '0'

bit 3 **G4POL:** Gate 4 Output Polarity Control bit
- 1 = The output of gate 4 is inverted when applied to the logic cell
- 0 = The output of gate 4 is not inverted

bit 2 **G3POL:** Gate 3 Output Polarity Control bit
- 1 = The output of gate 3 is inverted when applied to the logic cell
- 0 = The output of gate 3 is not inverted

bit 1 **G2POL:** Gate 2 Output Polarity Control bit
- 1 = The output of gate 2 is inverted when applied to the logic cell
- 0 = The output of gate 2 is not inverted

bit 0 **G1POL:** Gate 1 Output Polarity Control bit
- 1 = The output of gate 1 is inverted when applied to the logic cell
- 0 = The output of gate 1 is not inverted

**REGISTER 28-3: CLCxSEL0: GENERIC CLCx DATA 1 SELECT 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 |
|---|---|---|---|---|---|---|---|
| — | — | D1S<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 **D1S<5:0>:** CLCx Data1 Input Selection bits
See Table 28-1.

**REGISTER 28-4: CLCxSEL1: GENERIC CLCx DATA 2 SELECT 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 |
|---|---|---|---|---|---|---|---|
| — | — | D2S<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 **D2S<5:0>:** CLCx Data 2 Input Selection bits
See Table 28-1.

**REGISTER 28-5: CLCxSEL2: GENERIC CLCx DATA 3 SELECT 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 |
|---|---|---|---|---|---|---|---|
| — | — | D3S<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 **D3S<5:0>:** CLCx Data 3 Input Selection bits
See Table 28-1.

**REGISTER 28-6: CLCxSEL3: GENERIC CLCx DATA 4 SELECT 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 |
|---|---|---|---|---|---|---|---|
| — | — | D4S<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 **D4S<5:0>:** CLCx Data 4 Input Selection bits
See Table 28-1.

**REGISTER 28-7: CLCxGLS0: GATE 1 LOGIC SELECT REGISTER**

| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
|---|---|---|---|---|---|---|---|
| G1D4T | G1D4N | G1D3T | G1D3N | G1D2T | G1D2N | G1D1T | G1D1N |
| 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 **G1D4T:** Gate 1 Data 4 True (non-inverted) bit
1 = d4T is gated into g1
0 = d4T is not gated into g1

bit 6 **G1D4N:** Gate 1 Data 4 Negated (inverted) bit
1 = d4N is gated into g1
0 = d4N is not gated into g1

bit 5 **G1D3T:** Gate 1 Data 3 True (non-inverted) bit
1 = d3T is gated into g1
0 = d3T is not gated into g1

bit 4 **G1D3N:** Gate 1 Data 3 Negated (inverted) bit
1 = d3N is gated into g1
0 = d3N is not gated into g1

bit 3 **G1D2T:** Gate 1 Data 2 True (non-inverted) bit
1 = d2T is gated into g1
0 = d2T is not gated into g1

bit 2 **G1D2N:** Gate 1 Data 2 Negated (inverted) bit
1 = d2N is gated into g1
0 = d2N is not gated into g1

bit 1 **G1D1T:** Gate 1 Data 1 True (non-inverted) bit
1 = d1T is gated into g1
0 = d1T is not gated into g1

bit 0 **G1D1N:** Gate 1 Data 1 Negated (inverted) bit
1 = d1N is gated into g1
0 = d1N is not gated into g1

**REGISTER 28-8: CLCxGLS1: GATE 2 LOGIC SELECT REGISTER**

| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
|---|---|---|---|---|---|---|---|
| G2D4T | G2D4N | G2D3T | G2D3N | G2D2T | G2D2N | G2D1T | G2D1N |
| 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 **G2D4T:** Gate 2 Data 4 True (non-inverted) bit
1 = d4T is gated into g2
0 = d4T is not gated into g2

bit 6 **G2D4N:** Gate 2 Data 4 Negated (inverted) bit
1 = d4N is gated into g2
0 = d4N is not gated into g2

bit 5 **G2D3T:** Gate 2 Data 3 True (non-inverted) bit
1 = d3T is gated into g2
0 = d3T is not gated into g2

bit 4 **G2D3N:** Gate 2 Data 3 Negated (inverted) bit
1 = d3N is gated into g2
0 = d3N is not gated into g2

bit 3 **G2D2T:** Gate 2 Data 2 True (non-inverted) bit
1 = d2T is gated into g2
0 = d2T is not gated into g2

bit 2 **G2D2N:** Gate 2 Data 2 Negated (inverted) bit
1 = d2N is gated into g2
0 = d2N is not gated into g2

bit 1 **G2D1T:** Gate 2 Data 1 True (non-inverted) bit
1 = d1T is gated into g2
0 = d1T is not gated into g2

bit 0 **G2D1N:** Gate 2 Data 1 Negated (inverted) bit
1 = d1N is gated into g2
0 = d1N is not gated into g2

**REGISTER 28-9: CLCxGLS2: GATE 3 LOGIC SELECT REGISTER**

| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
|---|---|---|---|---|---|---|---|
| G3D4T | G3D4N | G3D3T | G3D3N | G3D2T | G3D2N | G3D1T | G3D1N |
| 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 **G3D4T:** Gate 3 Data 4 True (non-inverted) bit
1 = d4T is gated into g3
0 = d4T is not gated into g3

bit 6 **G3D4N:** Gate 3 Data 4 Negated (inverted) bit
1 = d4N is gated into g3
0 = d4N is not gated into g3

bit 5 **G3D3T:** Gate 3 Data 3 True (non-inverted) bit
1 = d3T is gated into g3
0 = d3T is not gated into g3

bit 4 **G3D3N:** Gate 3 Data 3 Negated (inverted) bit
1 = d3N is gated into g3
0 = d3N is not gated into g3

bit 3 **G3D2T:** Gate 3 Data 2 True (non-inverted) bit
1 = d2T is gated into g3
0 = d2T is not gated into g3

bit 2 **G3D2N:** Gate 3 Data 2 Negated (inverted) bit
1 = d2N is gated into g3
0 = d2N is not gated into g3

bit 1 **G3D1T:** Gate 3 Data 1 True (non-inverted) bit
1 = d1T is gated into g3
0 = d1T is not gated into g3

bit 0 **G3D1N:** Gate 3 Data 1 Negated (inverted) bit
1 = d1N is gated into g3
0 = d1N is not gated into g3

**REGISTER 28-10: CLCxGLS3: GATE 4 LOGIC SELECT REGISTER**

| R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
|---|---|---|---|---|---|---|---|
| G4D4T | G4D4N | G4D3T | G4D3N | G4D2T | G4D2N | G4D1T | G4D1N |
| 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 **G4D4T:** Gate 4 Data 4 True (non-inverted) bit
1 = d4T is gated into g4
0 = d4T is not gated into g4

bit 6 **G4D4N:** Gate 4 Data 4 Negated (inverted) bit
1 = d4N is gated into g4
0 = d4N is not gated into g4

bit 5 **G4D3T:** Gate 4 Data 3 True (non-inverted) bit
1 = d3T is gated into g4
0 = d3T is not gated into g4

bit 4 **G4D3N:** Gate 4 Data 3 Negated (inverted) bit
1 = d3N is gated into g4
0 = d3N is not gated into g4

bit 3 **G4D2T:** Gate 4 Data 2 True (non-inverted) bit
1 = d2T is gated into g4
0 = d2T is not gated into g4

bit 2 **G4D2N:** Gate 4 Data 2 Negated (inverted) bit
1 = d2N is gated into g4
0 = d2N is not gated into g4

bit 1 **G4D1T:** Gate 4 Data 1 True (non-inverted) bit
1 = d1T is gated into g4
0 = d1T is not gated into g4

bit 0 **G4D1N:** Gate 4 Data 1 Negated (inverted) bit
1 = d1N is gated into g4
0 = d1N is not gated into g4

**REGISTER 28-11: CLCDATA: CLC DATA OUTPUT**

| U-0 | U-0 | U-0 | U-0 | R-0 | R-0 | R-0 | R-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | MLC4OUT | MLC3OUT | MLC2OUT | MLC1OUT |
| bit 7 | | | | | | | bit 0 |

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

| | |
|---|---|
| bit 7-4 | Unimplemented: Read as ‘0’ |
| bit 3 | MLC4OUT: Mirror copy of LC4OUT bit |
| bit 2 | MLC3OUT: Mirror copy of LC3OUT bit |
| bit 1 | MLC2OUT: Mirror copy of LC2OUT bit |
| bit 0 | MLC1OUT: Mirror copy of LC1OUT bit |

**TABLE 28-4: SUMMARY OF REGISTERS ASSOCIATED WITH CLCx**

| Name | Bit7 | Bit6 | Bit5 | Bit4 | BIt3 | Bit2 | Bit1 | Bit0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| ANSELA | — | — | ANSA5 | ANSA4 | ANSA3 | ANSA2 | ANSA1 | ANSA0 | 153 |
| ANSELB | — | — | ANSB5 | ANSB4 | ANSB3 | ANSB2 | ANSB1 | ANSB0 | 158 |
| ANSELC | ANSC7 | ANSC6 | ANSC5 | ANSC4 | ANSC3 | ANSC2 | — | — | 163 |
| CLCxCON | EN | — | OUT | INTP | INTN | MODE<2:0> | | | 359 |
| CLCDATA | — | — | — | — | MLC4OUT | MLC3OUT | MLC2OUT | MLC1OUT | 366 |
| CLCxGLS0 | G1D4T | G1D4N | G1D3T | G1D3N | G1D2T | G1D2N | G1D1T | G1D1N | 362 |
| CLCxGLS1 | G2D4T | G2D4N | G2D3T | G2D3N | G2D2T | G2D2N | G2D1T | G2D1N | 363 |
| CLCxGLS2 | G3D4T | G3D4N | G3D3T | G3D3N | G3D2T | G3D2N | G3D1T | G3D1N | 364 |
| CLCxGLS3 | G4D4T | G4D4N | G4D3T | G4D3N | G4D2T | G4D2N | G4D1T | G4D1N | 365 |
| CLCxPOL | POL | — | — | — | G4POL | G3POL | G2POL | G1POL | 360 |
| CLCxSEL0 | — | — | D1S<5:0> | | | | | | 361 |
| CLCxSEL1 | — | — | D2S<5:0> | | | | | | 361 |
| CLCxSEL2 | — | — | D3S<5:0> | | | | | | 361 |
| CLCxSEL3 | — | — | D4S<5:0> | | | | | | 362 |
| CLCINxPPS | — | — | CLCINxPPS<5:0> | | | | | | 172, 174 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 108 |
| PIE3 | — | — | COG2IE | ZCDIE | COG2IE | CLC3IE | CLC2IE | CLC1IE | 111 |
| PIR3 | — | — | COG2IF | ZCDIF | COG2IF | CLC3IF | CLC2IF | CLC1IF | 117 |
| RxyPPS | — | — | RxyPPS<5:0> | | | | | | 172 |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 152 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 157 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 162 |

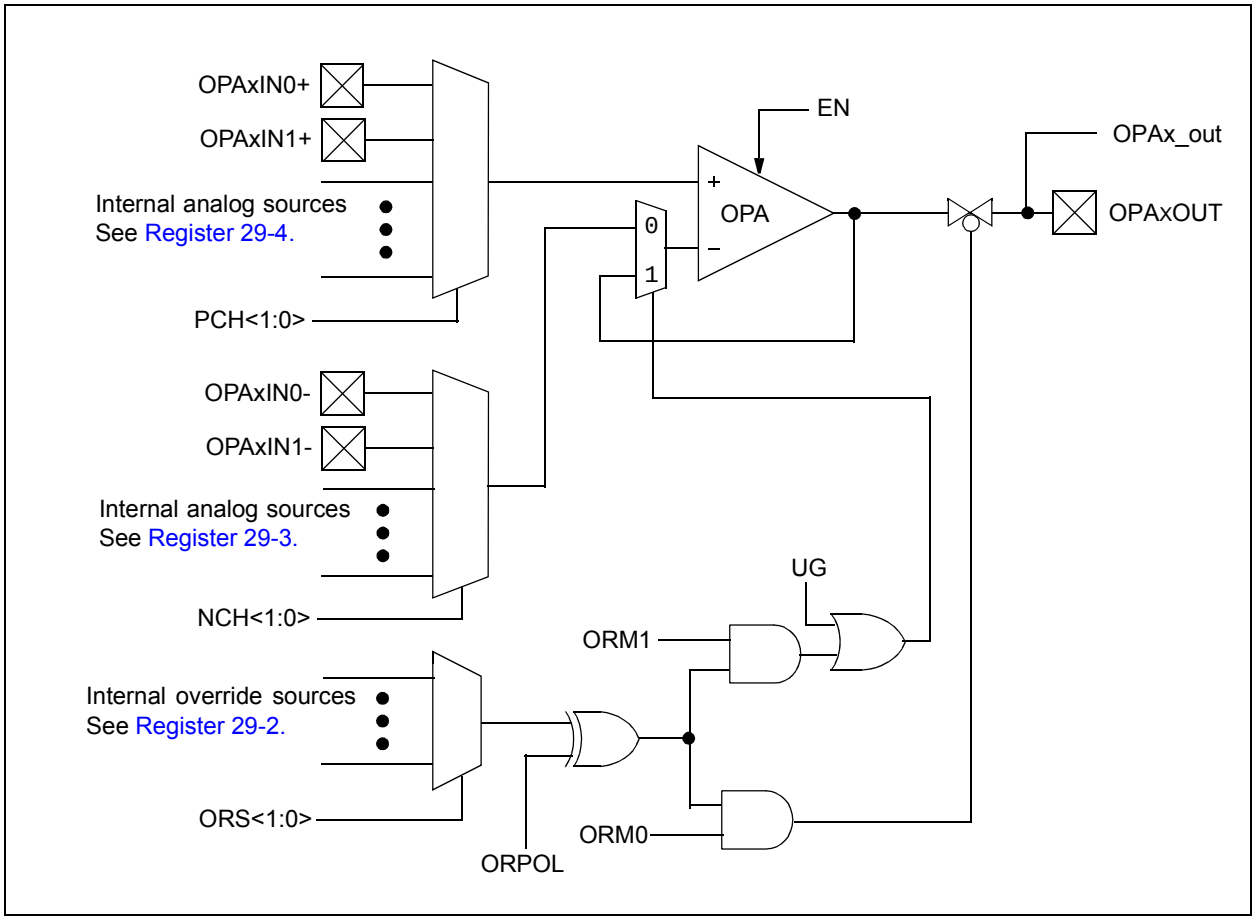
**Legend:** — = unimplemented read as '0'. Shaded cells are not used for CLC module.

## 29.0 OPERATIONAL AMPLIFIER (OPA) MODULES

The Operational Amplifier (OPA) is a standard three-terminal device requiring external feedback to operate. The OPA module has the following features:

- External connections to I/O ports
- Low leakage inputs
- Factory Calibrated Input Offset Voltage
- Unity gain control
- Programmable positive and negative source selections
- Override controls
  - Forced tri-state output
  - Forced unity gain

**FIGURE 29-1: OPAx MODULE BLOCK DIAGRAM**



## 29.1 OPA Module Performance

Common AC and DC performance specifications for the OPA module:

- Common-Mode Voltage Range
- Leakage Current
- Input Offset Voltage
- Open-Loop Gain
- Gain Bandwidth Product

**Common-mode voltage range** is the specified voltage range for the OPA+ and OPA- inputs, for which the OPA module will perform to within its specifications. The OPA module is designed to operate with input voltages between VSS and VDD. Behavior for common-mode voltages greater than VDD, or below VSS, are not guaranteed.

**Leakage current** is a measure of the small source or sink currents on the OPA+ and OPA- inputs. To minimize the effect of leakage currents, the effective impedances connected to the OPA+ and OPA- inputs should be kept as small as possible and equal.

**Input offset voltage** is a measure of the voltage difference between the OPA+ and OPA- inputs in a closed loop circuit with the OPA in its linear region. The offset voltage will appear as a DC offset in the output equal to the input offset voltage, multiplied by the gain of the circuit. The input offset voltage is also affected by the common-mode voltage. The OPA is factory calibrated to minimize the input offset voltage of the module.

**Open-loop gain** is the ratio of the output voltage to the differential input voltage, (OPA+) - (OPA-). The gain is greatest at DC and falls off with frequency.

**Gain Bandwidth Product** or GBWP is the frequency at which the open-loop gain falls off to 0 dB.

## 29.2 OPA Module Control

The OPA module is enabled by setting the OPAxEN bit of the OPAxCON register (Register 29-1). When enabled, the OPA forces the output driver of the OPAxOUT pin into tri-state to prevent contention between the driver and the OPA output.

| | |
|---|---|
| **Note:** | When the OPA module is enabled, the OPAxOUT pin is driven by the op amp output, not by the PORT digital driver. Refer to Table 36-17: Operational Amplifier (OPA) for the op amp output drive capability. |

### 29.2.1 UNITY GAIN MODE

The OPAxUG bit of the OPAxCON register (Register 29-1) selects the Unity Gain mode. When unity gain is selected, the OPA output is connected to the inverting input and the OPAxIN pin is relinquished, releasing the pin for general purpose input and output.

### 29.2.2 PROGRAMMABLE SOURCE SELECTIONS

The inverting and non-inverting sources are selected with the OPAxNCHS (Register 29-3) and OPAxPCHS (Register 29-4) registers, respectively. Sources include:

- Internal DACs
- Device pins
- Internal slope compensation ramp generator
- Other op amps in the device

## 29.3 Override Control

### 29.3.1 OVERRIDE MODE

The op amp operation can be overridden in two ways:

- Forced tri-state output
- Force unity gain

The Override mode is selected with the ORM<1:0> bits of the OPxCON register (Register 29-1). The override is in effect when the mode is selected and the override source is true.

### 29.3.2 OVERRIDE SOURCES

The override source is selected with the OPAxORS register (Register 29-2). Sources are from internal peripherals including:

- CCP outputs
- PWM outputs
- Comparator outputs
- Zero-cross detect output
- Configurable Logic Cell outputs
- COG outputs

### 29.3.3 OVERRIDE SOURCE POLARITY

The override source polarity can be inverted so that the override will occur on either the high or low level of the selected source. Override polarity is controlled by the ORPOL bit of the OPAxCON register (Register 29-1).

## 29.4 Effects of Reset

A device Reset forces all registers to their Reset state. This disables the OPA module.

## 29.5 Effects of Sleep

The operational amplifier continues to operate when the device is put in Sleep mode.

## 29.6 Register Definitions: Op Amp Control

Long bit name prefixes for the op amp peripherals are shown in Table 29-1. Refer to **Section 1.1 "Register and Bit naming conventions"** for more information

**TABLE 29-1:**

| Peripheral | Bit Name Prefix |
|---|---|
| OPA1 | OPA1 |
| OPA2 | OPA2 |
| OPA3 | OPA3 |

**REGISTER 29-1: OPAxCON: OPERATIONAL AMPLIFIER (OPAx) CONTROL REGISTER**

| R/W-0/0 | U-0 | U-0 | R/W-0/0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| EN | — | — | UG | — | ORPOL | ORM<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 **EN:** Op Amp Enable bit
1 = Op amp is enabled
0 = Op amp is disabled and consumes no active power

bit 6-5 **Unimplemented:** Read as '0'

bit 4 **UG:** Op Amp Unity Gain Select bit
1 = OPA output is connected to inverting input. OPAxIN- pin is available for general purpose I/O.
0 = Inverting input is connected to the OPAxIN- pin

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

bit 2 **ORPOL:** Op Amp Override Source Polarity bit
1 = Override source polarity is inverted. Override occurs when source is high.
0 = Override source polarity is not inverted. Override occurs when source is low.

bit 1-0 **ORM<1:0>:** Op Amp Override Mode Selection bits
11 = Reserved. Do not use.
10 = Op amp is forced to unity gain when override source is true.
01 = Op amp output is tri-stated when override source is true.
00 = Output override function is disabled.

**REGISTER 29-2: OPAxORS: OP AMP OVERRIDE SOURCE SELECTION REGISTER**

| U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/x | R/W-0/x | R/W-0/0 | R/W-0/x |
|---|---|---|---|---|---|---|---|
| — | — | — | ORS<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 | q = Value depends on condition |

bit 7-5 **Unimplemented:** Read as ‘0’

bit 4-0 **ORS<4:0>:** Op Amp Output Override Source Selection bits
See Table 29-2: Override Sources

**TABLE 29-2: OVERRIDE SOURCES**

| ORS<4:0> | OPA1 | OPA2 | OPA3 |
|---|---|---|---|
| 11111 | COG2D | COG2D | Reserved |
| 11110 | COG2C | COG2C | Reserved |
| 11101 | COG2B | COG2B | Reserved |
| 11100 | COG2A | COG2A | Reserved |
| 11011 | COG1D | COG1D | COG3D |
| 11010 | COG1C | COG1C | COG3C |
| 11001 | COG1B | COG1B | COG3B |
| 11000 | COG1A | COG1A | COG3A |
| 10111 | LC4_out | LC4_out | LC4_out |
| 10110 | LC3_out | LC3_out | LC3_out |
| 10101 | LC2_out | LC2_out | LC2_out |
| 10100 | LC1_out | LC1_out | LC1_out |
| 10011 | Reserved | Reserved | Reserved |
| 10010 | Reserved | Reserved | Reserved |
| 10001 | sync_C6OUT | sync_C6OUT | sync_C6OUT |
| 10000 | sync_C5OUT | sync_C5OUT | sync_C5OUT |
| 01111 | sync_C4OUT | sync_C4OUT | sync_C4OUT |
| 01110 | sync_C3OUT | sync_C3OUT | sync_C3OUT |
| 01101 | sync_C2OUT | sync_C2OUT | sync_C2OUT |
| 01100 | sync_C1OUT | sync_C1OUT | sync_C1OUT |
| 01011 | Reserved | Reserved | Reserved |
| 01010 | PWM11_out | PWM11_out | PWM11_out |
| 01001 | PWM6_out | PWM6_out | PWM6_out |
| 01000 | PWM5_out | PWM5_out | PWM5_out |
| 00111 | Reserved | Reserved | Reserved |
| 00110 | PWM9_out | PWM9_out | PWM9_out |
| 00101 | PWM4_out | PWM4_out | PWM4_out |
| 00100 | PWM3_out | PWM3_out | PWM3_out |
| 00011 | Reserved | Reserved | Reserved |
| 00010 | CCP3_out | CCP3_out | CCP3_out |

**TABLE 29-2: OVERRIDE SOURCES**

| ORS<4:0> | OPA1 | OPA2 | OPA3 |
|---|---|---|---|
| 00001 | CCP2_out | CCP2_out | CCP2_out |
| 00000 | CCP1_out | CCP1_out | CCP1_out |

**REGISTER 29-3: OPAxNCHS: OP AMP NEGATIVE CHANNEL SOURCE SELECT REGISTER**

| U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | NCH<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 | q = Value depends on condition |

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

bit 3-0 **NCH<3:0>:** Op Amp Inverting Input Channel Selection bits
See Table 29-3: Inverting Input Sources

**TABLE 29-3: INVERTING INPUT SOURCES**

| NCH<3:0> | OPA1 | OPA2 | OPA3 |
|---|---|---|---|
| 1111 | Reserved. Do not use | Reserved. Do not use | Reserved. Do not use |
| 1110 | Reserved. Do not use | Reserved. Do not use | Reserved. Do not use |
| 1101 | Reserved. Do not use | Reserved. Do not use | Reserved. Do not use |
| 1100 | Reserved. Do not use | Reserved. Do not use | Reserved. Do not use |
| 1011 | Reserved. Do not use | Reserved. Do not use | Reserved. Do not use |
| 1010 | Reserved. Do not use | Reserved. Do not use | Reserved. Do not use |
| 1001 | Reserved. Do not use | Reserved. Do not use | Reserved. Do not use |
| 1000 | PRG2_out | PRG2_out | Reserved. Do not use |
| 0111 | PRG1_out | PRG1_out | PRG3_out |
| 0110 | FVR_Buffer2 | FVR_Buffer2 | FVR_Buffer2 |
| 0101 | DAC4_out | DAC4_out | Reserved. Do not use |
| 0100 | DAC3_out | DAC3_out | DAC7_out |
| 0011 | DAC2_out | DAC2_out | Reserved. Do not use |
| 0010 | DAC1_out | DAC1_out | DAC5_out |
| 0001 | OPA1IN1- | OPA2IN1- | Reserved. Do not use |
| 0000 | OPA1IN0- | OPA2IN0- | OPA3IN0- |

**REGISTER 29-4: OPAxPCHS: OP AMP POSITIVE CHANNEL SOURCE SELECT REGISTER**

| U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | PCH<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 | q = Value depends on condition |

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

bit 3-0 **PCH<3:0>:** Op Amp Non-Inverting Input Channel Selection bits
See Table 29-4: Non-Inverting Input Sources

**TABLE 29-4: NON-INVERTING INPUT SOURCES**

| NCH<3:0> | OPA1 | OPA2 | OPA3 |
|---|---|---|---|
| 1111 | Reserved. Do not use | Reserved. Do not use | Reserved. Do not use |
| 1110 | Reserved. Do not use | Reserved. Do not use | Reserved. Do not use |
| 1101 | Reserved. Do not use | Reserved. Do not use | Reserved. Do not use |
| 1100 | Reserved. Do not use | Reserved. Do not use | Reserved. Do not use |
| 1011 | Reserved. Do not use | Reserved. Do not use | Reserved. Do not use |
| 1010 | Reserved. Do not use | Reserved. Do not use | Reserved. Do not use |
| 1001 | Reserved. Do not use | Reserved. Do not use | Reserved. Do not use |
| 1000 | PRG2_out | PRG2_out | Reserved. Do not use |
| 0111 | PRG1_out | PRG1_out | PRG3_out |
| 0110 | FVR_Buffer2 | FVR_Buffer2 | FVR_Buffer2 |
| 0101 | DAC4_out | DAC4_out | Reserved. Do not use |
| 0100 | DAC3_out | DAC3_out | DAC7_out |
| 0011 | DAC2_out | DAC2_out | Reserved. Do not use |
| 0010 | DAC1_out | DAC1_out | DAC5_out |
| 0001 | OPA1IN1+ | OPA2IN1+ | Reserved. Do not use |
| 0000 | OPA1IN0+ | OPA2IN0+ | OPA3IN0+ |

**TABLE 29-5: SUMMARY OF REGISTERS ASSOCIATED WITH OP AMPS**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| ANSELB | --- | --- | ANSB5 | ANSB4 | ANSB3 | ANSB2 | ANSB1 | ANSB0 | 158 |
| ANSELC | ANSC7 | ANSC6 | ANSC5 | ANSC4 | ANSC3 | ANSC2 | — | — | 163 |
| DAC1CON0 | EN | FM | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 212 |
| DAC2CON0 | EN | FM | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 212 |
| DAC5CON0 | EN | FM | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 212 |
| DAC3CON0 | EN | — | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 207 |
| DAC4CON0 | EN | — | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 207 |
| DAC7CON0 | EN | — | OE1 | OE2 | PSS<1:0> | | NSS<1:0> | | 207 |
| DAC3REF | --- | --- | --- | REF<4:0> | | | | | 208 |
| DAC4REF | --- | --- | --- | REF<4:0> | | | | | 208 |
| DAC7REF | --- | --- | --- | REF<4:0> | | | | | 208 |
| DAC1REFH | REF<9:x> (x Depends on FM bit) | | | | | | | | 213 |
| DAC2REFH | REF<9:x> (x Depends on FM bit) | | | | | | | | 213 |
| DAC5REFH | REF<9:x> (x Depends on FM bit) | | | | | | | | 213 |
| FVRCON | FVREN | FVRRDY | TSEN | TSRNG | CDAFVR<1:0> | | ADFVR<1:0> | | 186 |
| OPAxCON | EN | — | — | UG | — | ORPOL | ORM<1:0> | | 370 |
| OPAxNCHS | — | — | — | — | NCH<3:0> | | | | 372 |
| OPAxPCHS | — | — | — | — | PCH<3:0> | | | | 373 |
| OPAxORS | — | — | — | ORS<4:0> | | | | | 371 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 157 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 162 |

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

## 30.0 PROGRAMMABLE RAMP GENERATOR (PRG) MODULE

The Programmable Ramp Generator (PRG) module is designed to provide rising and falling linear ramps. Typical applications include slope compensation for fixed frequency, continuous current, and Current mode switched power supplies. Slope compensation is a necessary feature of these power supplies because it prevents frequency instabilities at duty cycles greater than 50%.

The PRG has the following features:

- Linear positive and negative voltage ramp outputs
- Programmable current source/sink
- Internal and external reference voltage selection
- Internal and external timing source selection

A simplified block diagram of the PRG is shown in Figure 30-1.

## 30.1 Fundamental Operation

The PRG can be operated in three voltage ramp generator modes:

- Falling Voltage (slope compensation)
- Rising Voltage
- Alternating Rising and Falling Voltage

In the Rising or Falling mode an internal capacitor is discharged when the set_falling timing input is true and charged by an internally generated constant current when the set_rising timing input is true. The resulting linear ramp starts at the selected voltage input level and resets back to that level when the ramp is terminated by the set_falling timing input. The set_falling input dominates when both timing inputs are true.

To control the operation with a single-ended source, select the same source for both the set_rising and set_falling inputs and invert the polarity of one of them with the corresponding polarity control bit.

In the Alternating mode the capacitor is not discharged but alternates between being charged in one direction then the other.

Input selections are identical for all modes. The input voltage is supplied by any of the following:

- The PRGxIN0 or PRGxIN1 pins
- The buffered output of the internal Fixed Voltage Reference (FVR),
- Any of the internal DACs.

The timing sources are selected from the following:

- The synchronized output of any comparator
- Any PWM output
- Any I/O pin

The ramp output is available as an input to any of the comparators or op amps.

### 30.1.1 SLOPE COMPENSATION

Slope compensation works by quickly discharging an internal capacitor at the beginning of each PWM period. One side of the internal capacitor is connected to the voltage input source and the other side is connected to the internal current sink. The internal current sink charges this capacitor at a programmable rate. As the capacitor charges, the capacitor voltage is subtracted from the voltage source, producing a linear voltage decay at the required rate (see Figure 30-2). The ramp terminates and the capacitor is discharged when the set_falling timing input goes true. The next ramp starts when the set_rising timing input goes true.

Enabling the optional one-shot by setting the OS bit of the PRGxCON0 register ensures that the capacitor is fully discharged by overriding the set_rising timing input and holding the shorting switch closed for at least the one-shot period, typically 50 ns. Edge sensitive timing inputs that occur during the one-shot period will be ignored. Level sensitive timing inputs that occur during, and extend beyond, the one-shot period will be suspended until the end of the one-shot time.

### 30.1.2 RAMP GENERATION

Ramp generation is similar to slope compensation except that the slope is either both rising and falling or just rising.

#### 30.1.2.1 Alternating Rising/Falling Ramps

The alternating rising/falling ramp generation function works by employing the built-in current source and sink and relying on the synchronous control of the internal analog switches and timing sources to ramp the module's output voltage up and then subsequently down.

Once initialized, the output voltage is ramped up linearly by the current source at a programmable rate until the set_falling timing source goes true, at which point the current source is disengaged. At the same time, the current sink is engaged to linearly ramp down the output voltage, also at a programmable rate, until the set_rising timing input goes true, thereby reversing the ramp slope. The process then repeats to create a saw tooth like waveform as shown in Figure 30-3 and Figure 30-4.

The set_rising and set_falling timing inputs can be either edge or level sensitive which is selected with the respective REDG and FEDG bits of the PRGxCON0 register. Edge sensitive operation is recommended for periodic signals such as clocks, and level sensitive operation is recommended for analog limit triggers such as comparator outputs.

When the one-shot is enabled (OS bit is set) then both the falling and rising ramps will persist for a minimum of the one-shot period. Edge sensitive timing inputs that occur during the one-shot period will be ignored. Level sensitive timing inputs that occur during, and extend beyond, the one-shot period will be suspended until the end of the one-shot time.

#### 30.1.2.2 Rising Ramp

The Rising Ramp mode is identical to the Slope Compensation mode except that the ramps have a rising slope instead of a falling slope. One side of the internal capacitor is connected to the voltage input source and the other side is connected to the internal current source. The internal current source charges this capacitor at a programmable rate. As the capacitor charges, the capacitor voltage is added to the voltage source, producing a linear voltage rise at the required rate (see Figure 30-5). The ramp terminates and the capacitor is discharged when the set_falling timing input goes true. The next ramp starts when the set_rising timing input goes true.

Enabling the optional one-shot by setting the OS bit of the PRGxCON0 register ensures that the capacitor is fully discharged by overriding the set_rising timing input and holding the shorting switch closed for at least the one-shot period, typically 50 ns. Edge sensitive timing inputs that occur during the one-shot period will be ignored. Level sensitive timing inputs that occur during, and extend beyond, the one-shot period will be suspended until the end of the one-shot time.

## 30.2 Enable, Ready, Go

The EN bit of the PRGxCON0 register enables the analog circuitry including the current sources. This permits preparing the PRG module for use and allowing it to become stable before putting it into operation. When the EN bit is set then the timing inputs are enabled so that initial ramp action can be determined before the GO bit is set. The capacitor shorting switch is closed when the EN bit is set and remains closed while the GO bit is zero.

The RDY bit of the PRGxCON1 register indicates that the analog circuits and current sources are stable.

The GO bit of the PRGxCON0 register enables the switch control circuits, thereby putting the PRG into operation. The GO transition from cleared to set triggers the one-shot, thereby extending the capacitor shorting switch closure for the one-shot period.

To ensure predictable operation, set the EN bit first then wait for the RDY bit to go high before setting the GO bit.

## 30.3 Independent Set_rising and Set_falling Timing Inputs

The timing inputs determine when the ramp starts and stops. In the Alternating Rising/Falling mode the ramp rises when the set_rising input goes true and falls when the set_falling input goes true. In the Slope Compensation and Rising Ramp modes the capacitor is discharged when the set_falling timing input goes true and the ramp starts when the set_rising timing input goes true. The set_falling input dominates the set_rising input.

## 30.4 Level and Edge Timing Sensitivity

The set_rising and set_falling timing inputs can be independently configured as either level or edge sensitive.

Level sensitive operation is useful when it is necessary to detect a timing input true state after an overriding condition ceases. For example, level sensitivity is useful for capacitor generated timing inputs that may be suppressed by the overriding action of the one-shot. With level sensitivity a capacitor output that changes during the one-shot period will be detected at the end of the one-shot time. With edge sensitivity the change would be ignored.

Edge sensitive operation is useful for periodic timing inputs such as those generated by PWMs and clocks. The duty cycle of a level sensitive periodic signal may interfere with the other timing input. Consider an Alternating Ramp mode with a level sensitive 50% PWM as the set_rising timing source and a level sensitive comparator as the set_falling timing source. If the comparator output reverses the ramp while the PWM signal is still high then the ramp will improperly reverse again when the comparator signal goes low. That same scenario with the set_rising timing input set for edge sensitivity would properly change the ramp output to rising only on the rising edge of the PWM signal.

Set_rising and set_falling timing input edge sensitivity is selected with the respective REDG and FEDG bits of the PRGxCON1 register.

## 30.5 One-Shot Minimum Timing

The one-shot timer ensures a minimum capacitor discharge time in the Slope Compensation and Rising Ramp modes, and a minimum rising or falling ramp duration in the Alternating Ramp mode. Setting the OS bit of the PRGxCON0 register enables the one-shot timer.

## 30.6 DAC Voltage Sources

When using any of the DACs as the voltage source expect a voltage offset equal to the current setting times the DAC equivalent resistance. This will be a constant offset in the Slope Compensation and Ramp modes and a positive/negative step offset in the Alternating mode. To avoid this limitation, feed the DAC output to the PRG input through one of the op amps set for unity gain.

## 30.7 Operation During Sleep

The RG module is unaffected by Sleep.

## 30.8 Effects of a Reset

The RG module resets to a disabled condition.

**FIGURE 30-1: SIMPLIFIED PRG MODULE BLOCK DIAGRAM**

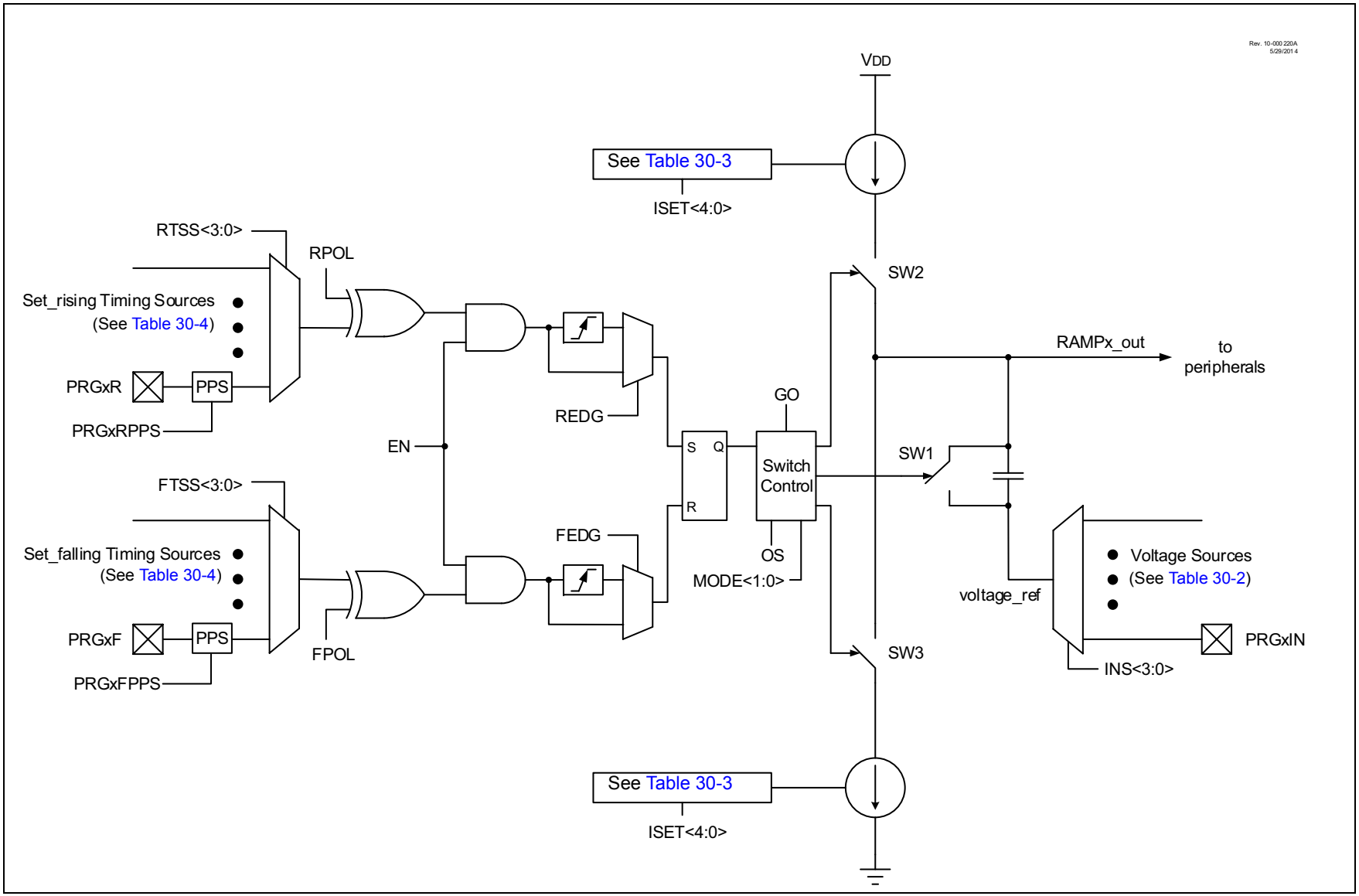


**FIGURE 30-2: SLOPE COMPENSATION (FALLING RAMP) TIMING DIAGRAM (MODE = 00)**

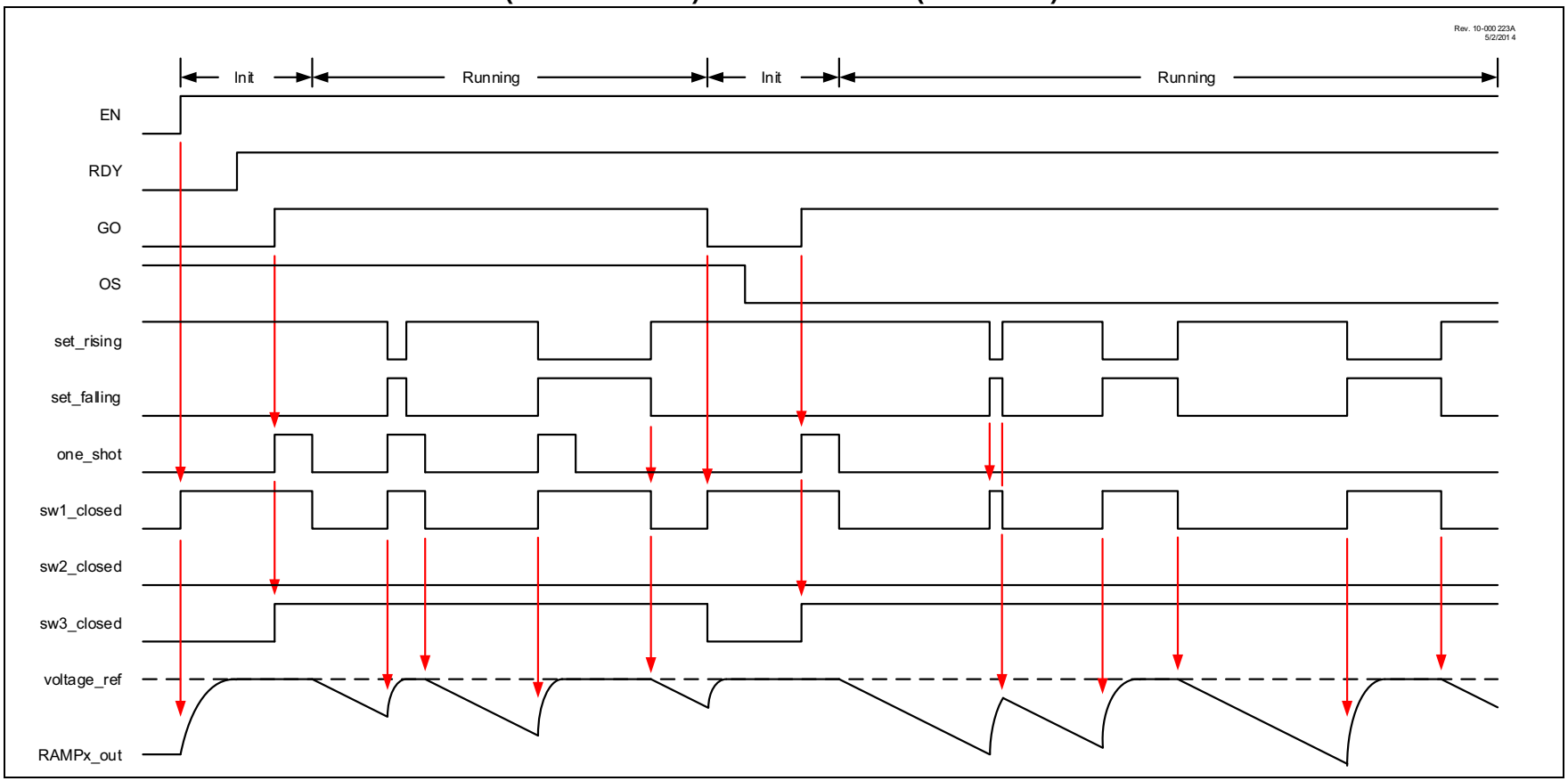


**FIGURE 30-3: ALTERNATING RISING/FALLING RAMP GENERATION TIMING DIAGRAM (OS = 0, MODE = 01)**

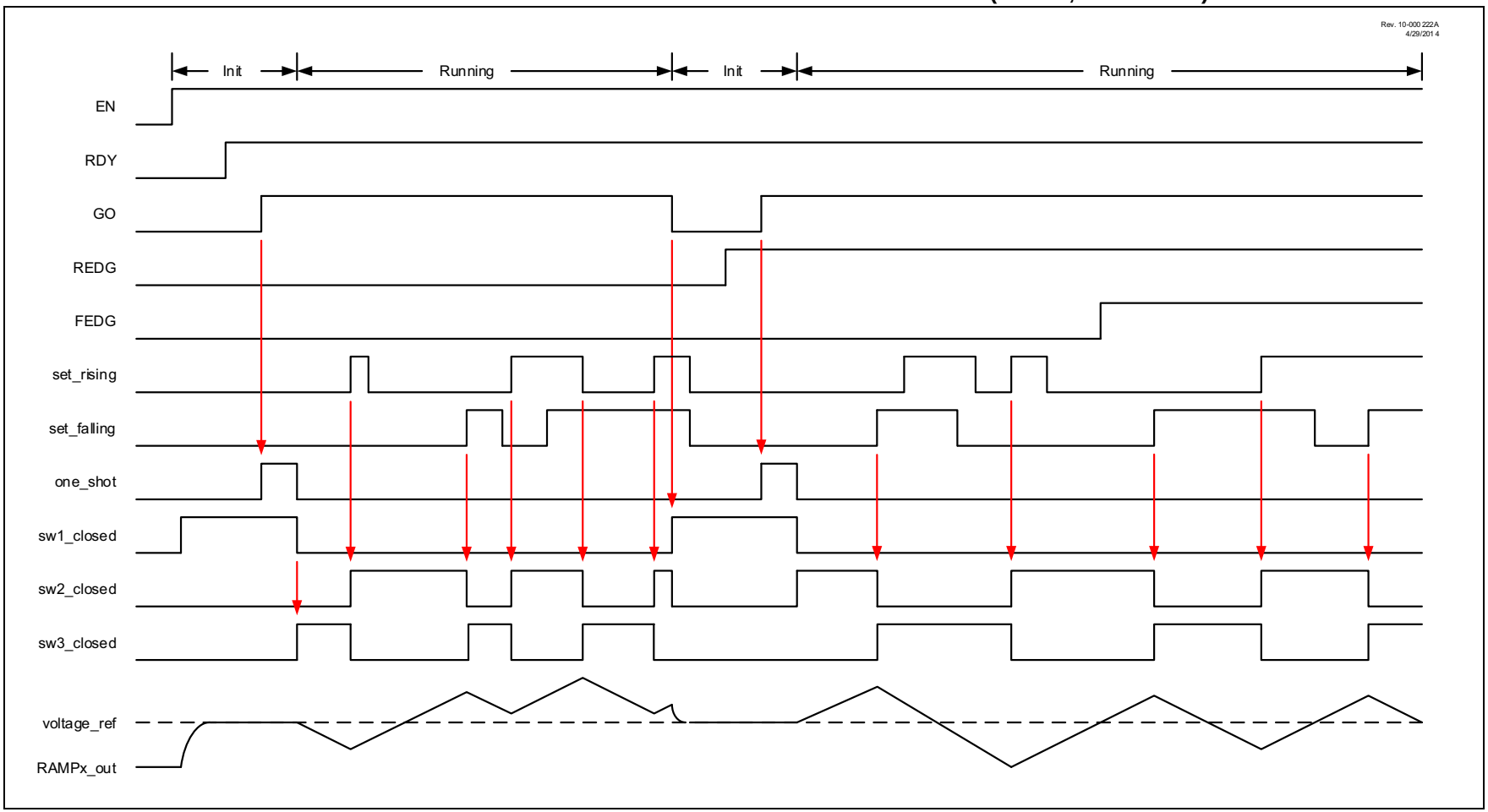


**FIGURE 30-4: ALTERNATING RISING/FALLING RAMP GENERATION TIMING DIAGRAM (OS = 1, MODE = 01)**

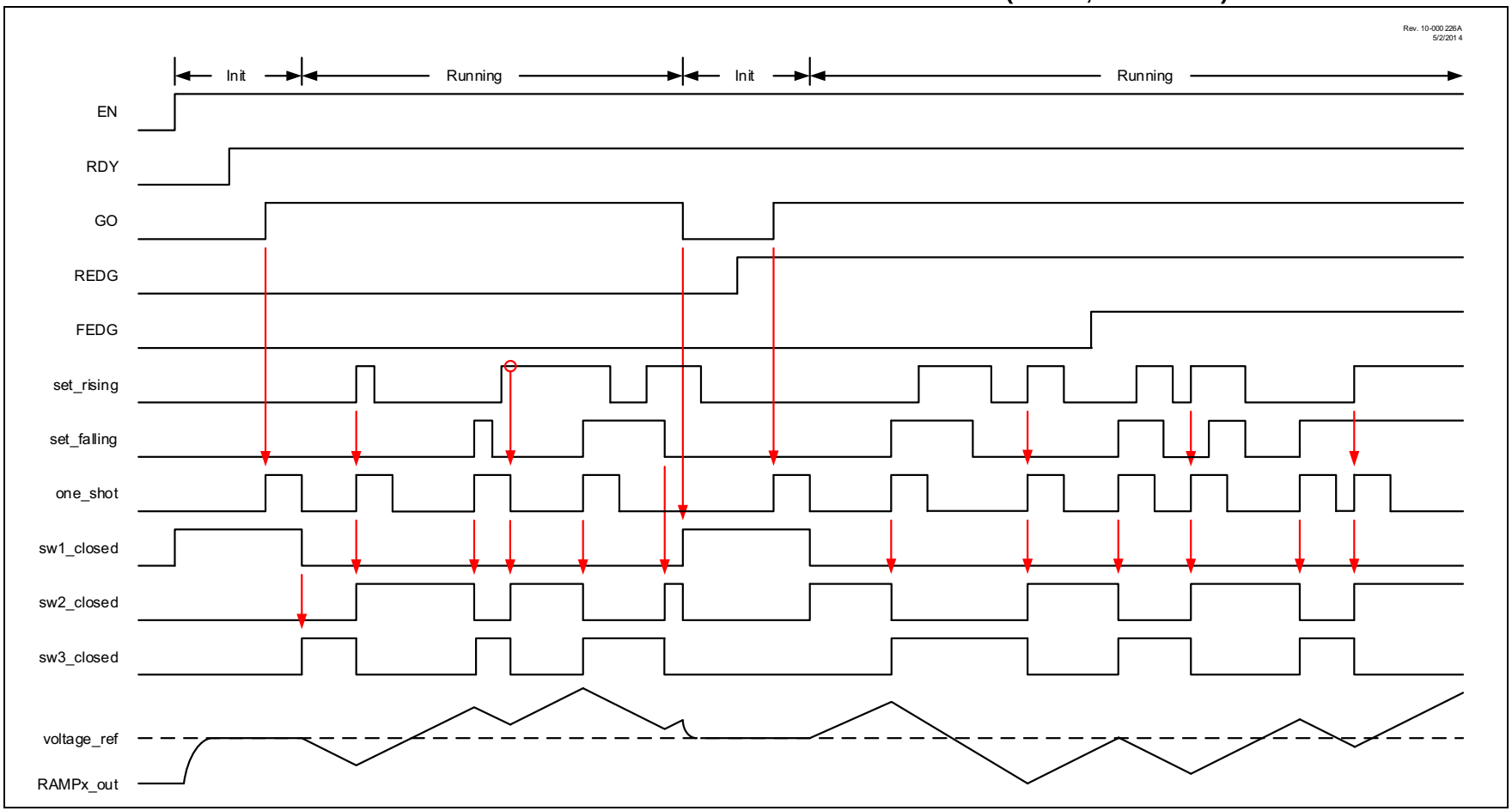
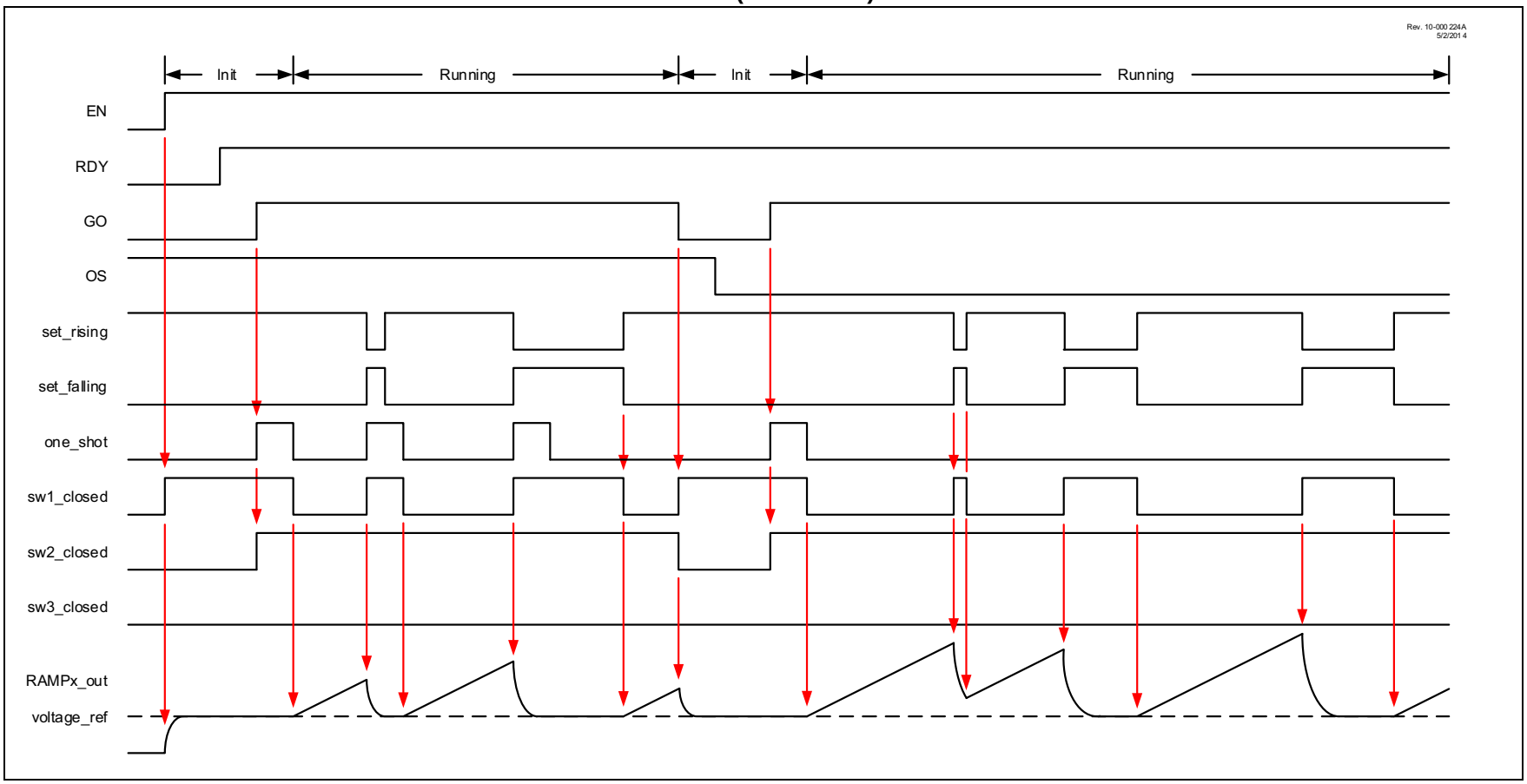


**FIGURE 30-5: RISING RAMP GENERATION TIMING DIAGRAM (MODE = 10)**



## 30.9 Slope Compensation Application

An example slope compensation circuit is shown in Figure 30-6. The PRG input voltage is PRGxIN which shares an I/O pin with the op amp output. The op amp output is designed to operate at the expected peak current sense voltage (i.e., VREF). The PRG output voltage starts at VREF and should fall at a rate less than half the target circuit current sense voltage rate of rise. Therefore, the compensator slope expressed as volts per µs can be computed by Equation 30-1.

**EQUATION 30-1:**

$$\frac{V}{\mu s} \geq \frac{\frac{VREF}{2}}{PWM\ Period\ (\mu s)}$$

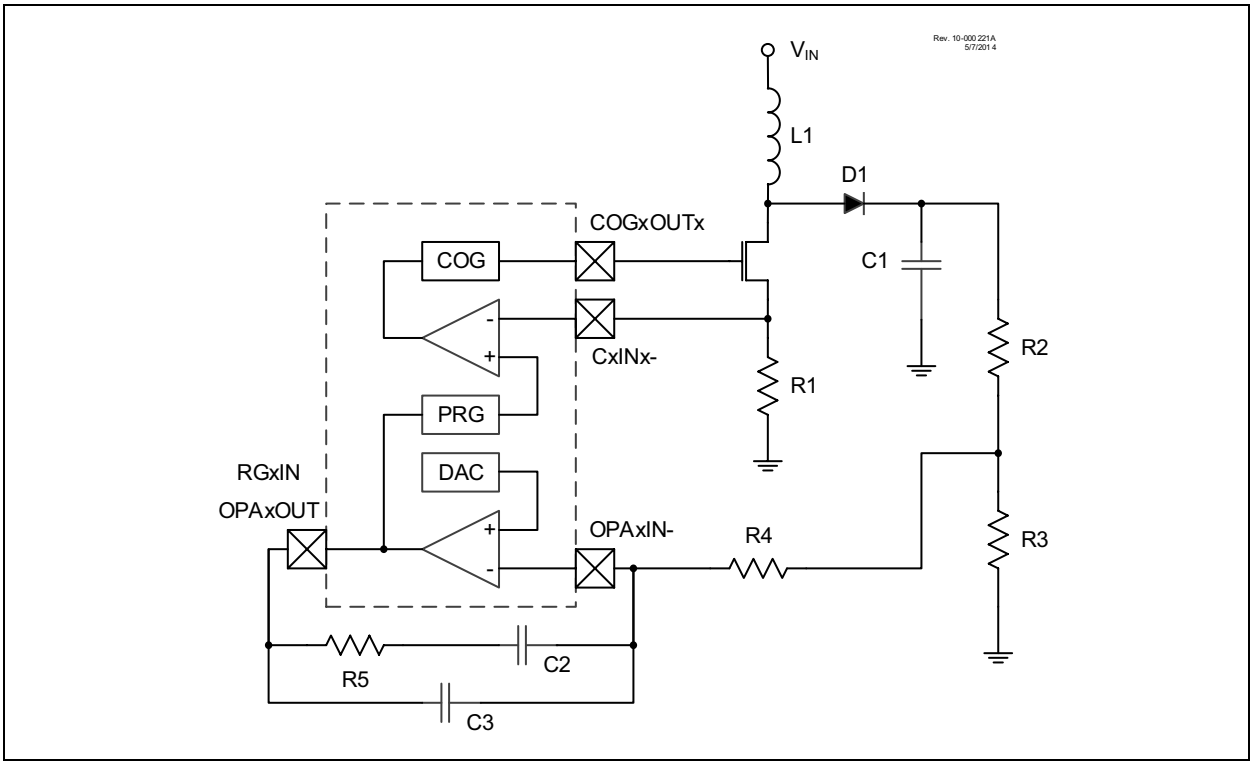
For example, when the circuit is using a 1Ω current sense resistor and the peak current is 1A, then the peak current expressed as a voltage is 1V. Therefore, for this example, the op amp output should be designed to operate at 1V. If the power supply PWM frequency is 1 MHz, then the period is 1 µs. Therefore, the desired slope is 0.5 V/µs, which is computed as shown in Equation 30-2.

**EQUATION 30-2:**

$$\frac{\frac{VREF}{2}}{PWM\ Period\ (\mu s)} = \frac{\frac{1}{2}}{1\mu s} = 0.5V/\mu s$$

**Note**: The setting for 0.5V/µs is ISET<4:0> = 6

**FIGURE 30-6: EXAMPLE SLOPE COMPENSATION CIRCUIT**



## 30.10 Register Definitions: Slope Compensation Control

Long bit name prefixes for the PRG peripherals are shown in Table 30-1. Refer to **Section 1.1 "Register and Bit naming conventions"** for more information

**TABLE 30-1:**

| Peripheral | Bit Name Prefix |
|---|---|
| PRG1 | RG1 |
| PRG2 | RG2 |
| PRG3 | RG3 |

**REGISTER 30-1: PRGxCON0: PROGRAMMABLE RAMP GENERATOR CONTROL 0 REGISTER**

| R/W-0/0 | 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 |
|---|---|---|---|---|---|---|---|
| EN | — | FEDG | REDG | MODE<1:0> | | OS | GO |
| 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 configuration bits |

bit 7 **EN:** Programmable Ramp Generator Enable bit
1 = PRG module is enabled
0 = PRG module is disabled

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

bit 5 **FEDG:** Set_falling Input Mode Select bit
1 = Set_falling timing input is edge sensitive
0 = Set_falling timing input is level sensitive

bit 4 **REDG:** Set_rising Input Mode Select bit
1 = Set_rising timing input is edge sensitive
0 = Set_rising timing input is level sensitive

bit 3-2 **MODE<1:0>:** Programmable Ramp Generator Mode Selection bits
11 = Reserved
10 = Rising Ramp Generator
01 = Alternating Rising/Falling Ramp Generator
00 = Slope Compensation

bit 1 **OS:** One-Shot Enable bit
1 = One-shot is enabled. Minimum capacitor discharge is internally timed by one-shot.
0 = One-shot is disabled. Capacitor is discharged when timing input is true.

bit 0 **GO:** Ramp Generation Control Start bit
If EN = 0
This bit is forced to 0
if EN = 1
1 = Slope or Ramp function is operating
0 = Slope or Ramp function is not operating. All current source current source switches are open and capacitor discharge switch is closed.

### REGISTER 30-2: PRGxCON1: PROGRAMMABLE RAMP GENERATOR CONTROL 1 REGISTER

| U-0 | U-0 | U-0 | U-0 | U-0 | R-0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | — | RDY | FPOL | RPOL |
| 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 configuration bits |

bit 7-3 **Unimplemented:** Read as ‘0’

bit 2 **RDY:** Slope Generator Ready Status bit
1 = PRG is ready
0 = PRG is not ready

bit 1 **FPOL:** Fall Event Polarity Select bit
1 = Set_falling timing input is active-low
0 = Set_falling timing input is active-high

bit 0 **RPOL:** Rise Event Polarity Select bit
1 = Set_rising timing input is active-low
0 = Set_rising timing input is active-high

### REGISTER 30-3: PRGxINS: VOLTAGE INPUT SELECT REGISTER

| U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | INS<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 | q = value depends on configuration bits |

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

bit 3-0 **INS<3:0>:** Voltage Input Select bits
Selects source of voltage level at which the ramp starts. See Table 30-2.

### TABLE 30-2: VOLTAGE INPUT SOURCES

| INS<2:0> | PRG1 Voltage Source | PRG2 Voltage Source | PRG3 Voltage Source |
|---|---|---|---|
| 1010-1111 | Reserved | Reserved | Reserved |
| 1001(1) | Switched PRG1IN1/OPA2OUT | Switched PRG1IN1/OPA2OUT | Reserved |
| 1000(1) | Switched PRG1IN0/OPA1OUT | Switched PRG1IN0/OPA1OUT | Switched PRG3IN0/OPA3OUT |
| 0111 | Reserved | Reserved | Reserved |
| 0110 | DAC4_output | DAC4_output | Reserved |
| 0101 | DAC3_output | DAC3_output | DAC7_output |
| 0100 | DAC2_output | DAC2_output | Reserved |
| 0011 | DAC1_output | DAC1_output | DAC5_output |
| 0010 | FVR_buffer2 | FVR_buffer2 | FVR_buffer2 |
| 0001 | PRG1IN1/OPA2OUT | PRG2IN1/OPA1OUT | Reserved |
| 0000 | PRG1IN0/OPA1OUT | PRG2IN0/OPA2OUT | PRG3IN0/OPA3OUT |

**Note 1:** Input source is switched off when op amp override is forcing tri-state. See **Section 29.3 “Override Control”**.

**REGISTER 30-4: PRGxCON2: PROGRAMMABLE RAMP GENERATOR CONTROL 2 REGISTER**

| 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 |
|---|---|---|---|---|---|---|---|
| — | — | — | ISET<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 | q = value depends on configuration bits |

bit 7-5 **Unimplemented:** Read as ‘0’

bit 4-0 **ISET<4:0>:** PRG Current Source/Sink Set bits
Current source/sink setting and slope rate. See Table 30-3.

**TABLE 30-3: PROGRAMMABLE RAMP GENERATOR CURRENT SETTINGS**

| ISET<4:0> | Current Setting (uA) | Slope Rate (V/us) | ISET<4:0> | Current Setting (uA) | Slope Rate (V/us) |
|---|---|---|---|---|---|
| 0h | 2 | 0.2 | 10h | 10 | 1.0 |
| 1h | 2.5 | 0.25 | 11h | 11 | 1.1 |
| 2h | 3 | 0.3 | 12h | 12 | 1.2 |
| 3h | 3.5 | 0.35 | 13h | 13 | 1.3 |
| 4h | 4 | 0.4 | 14h | 14 | 1.4 |
| 5h | 4.5 | 0.45 | 15h | 15 | 1.5 |
| 6h | 5 | 0.5 | 16h | 16 | 1.6 |
| 7h | 5.5 | 0.55 | 17h | 17 | 1.7 |
| 8h | 6 | 0.6 | 18h | 18 | 1.8 |
| 9h | 6.5 | 0.65 | 19h | 19 | 1.9 |
| Ah | 7 | 0.7 | 1Ah | 20 | 2.0 |
| Bh | 7.5 | 0.75 | 1Bh | 21 | 2.1 |
| Ch | 8 | 0.8 | 1Ch | 22 | 2.2 |
| Dh | 8.5 | 0.85 | 1Dh | 23 | 2.3 |
| Eh | 9 | 0.9 | 1Eh | 24 | 2.4 |
| Fh | 9.5 | 0.95 | 1Fh | 25 | 2.5 |

**REGISTER 30-5: PRGxRTSS: SET_RISING TIMING SOURCE SELECT REGISTER**

| U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | RTSS<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 | q = value depends on configuration bits |

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

bit 3-0 **RTSS<3:0>:** Set_rising Timing Source Select bits
See Table 30-4.

**REGISTER 30-6: PRGxFTSS: SET_FALLING TIMING SOURCE SELECT REGISTER**

| U-0 | U-0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | — | — | FTSS<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 | q = value depends on configuration bits |

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

bit 3-0 **FTSS<3:0>:** Set_falling Timing Source Select bits
See Table 30-4.

**TABLE 30-4: PROGRAMMABLE RAMP GENERATOR TIMING SOURCES**

| RTSS<3:0>/FTSS<3:0> | PRG1 Timing Source | PRG2 Timing Source | PRG3 Timing Source |
|---|---|---|---|
| 1111 | Reserved | Reserved | Reserved |
| 1110 | Reserved | Reserved | PWM11_output |
| 1101 | Reserved | Reserved | PWM6_output |
| 1100 | Reserved | Reserved | PWM5_output |
| 1011 | Reserved | Reserved | Reserved |
| 1010 | PWM9_output | PWM9_output | Reserved |
| 1001 | PWM4_output | PWM4_output | Reserved |
| 1000 | PWM3_output | PWM3_output | Reserved |
| 0111 | PRGxR/PRGxF Pin(1) | PRGxR/PRGxF Pin(1) | PRGxR/PRGxF Pin(1) |
| 0110 | Reserved | Reserved | Reserved |
| 0101 | Reserved | Reserved | sync_C6OUT |
| 0100 | Reserved | Reserved | sync_C5OUT |
| 0011 | sync_C4OUT | sync_C4OUT | Reserved |
| 0010 | sync_C3OUT | sync_C3OUT | Reserved |
| 0001 | sync_C2OUT | sync_C2OUT | Reserved |
| 0000 | sync_C1OUT | sync_C1OUT | Reserved |

**Note 1:** Input pin is selected with the PRGxRPPS or PRGxFPPS register.

**TABLE 30-5: SUMMARY OF REGISTERS ASSOCIATED WITH THE PRG MODULE**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| PRG1CON0 | EN | — | FEDG | REDG | MODE<1:0> | | OS | GO | 383 |
| PRG1CON1 | — | — | — | — | — | RDY | FPOL | RPOL | 384 |
| PRG1CON2 | — | — | — | ISET<4:0> | | | | | 385 |
| PRG1INS | — | — | — | — | INS<3:0> | | | | 384 |
| PRG1RPPS | — | — | PRG1RPPS<5:0> | | | | | | 386 |
| PRG1FPPS | — | — | PRG1FPPS<5:0> | | | | | | 386 |
| PRG1RTSS | — | — | — | — | RTSS<3:0> | | | | 172, 174 |
| PRG1FTSS | — | — | — | — | FTSS<3:0> | | | | 172, 174 |
| PRG2CON0 | EN | — | FEDG | REDG | MODE<1:0> | | OS | GO | 383 |
| PRG2CON1 | — | — | — | — | — | RDY | FPOL | RPOL | 384 |
| PRG2CON2 | — | — | — | ISET<4:0> | | | | | 385 |
| PRG2INS | — | — | — | — | INS<3:0> | | | | 384 |
| PRG2RPPS | — | — | PRG2RPPS<5:0> | | | | | | 386 |
| PRG2FPPS | — | — | PRG2FPPS<5:0> | | | | | | 386 |
| PRG2RTSS | — | — | — | — | RTSS<3:0> | | | | 172, 174 |
| PRG2FTSS | — | — | — | — | FTSS<3:0> | | | | 172, 174 |
| PORTC | RC7 | RC6 | RC5 | RC4 | RC3 | RC2 | RC1 | RC0 | 162 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 162 |
| ANSELC | ANSC7 | ANSC6 | ANSC5 | ANSC4 | ANSC3 | ANSC2 | — | — | 163 |
| WPUC | WPUC7 | WPUC6 | WPUC5 | WPUC4 | WPUC3 | WPUC2 | WPUC1 | WPUC0 | 164 |

**Legend:** — = unimplemented, read as '0'. Shaded cells are unused by the PRG module.

## 31.0 DATA SIGNAL MODULATOR (DSM)

The Data Signal Modulator (DSM) is a peripheral that allows the user to mix a data stream, also known as a modulator signal, with a carrier signal to produce a modulated output.

Both the carrier and the modulator signals are supplied to the DSM module either internally, from the output of a peripheral, or externally through an input pin.

The modulated output signal is generated by performing a logical “AND” operation of both the carrier and modulator signals and then provided to the MDxOUT pin.

The carrier signal is comprised of two distinct and separate signals. A carrier high (CARH) signal and a carrier low (CARL) signal. During the time in which the modulator (MOD) signal is in a logic high state, the DSM mixes the carrier high signal with the modulator signal. When the modulator signal is in a logic low state, the DSM mixes the carrier low signal with the modulator signal.

Using this method, the DSM can generate the following types of Key Modulation schemes:

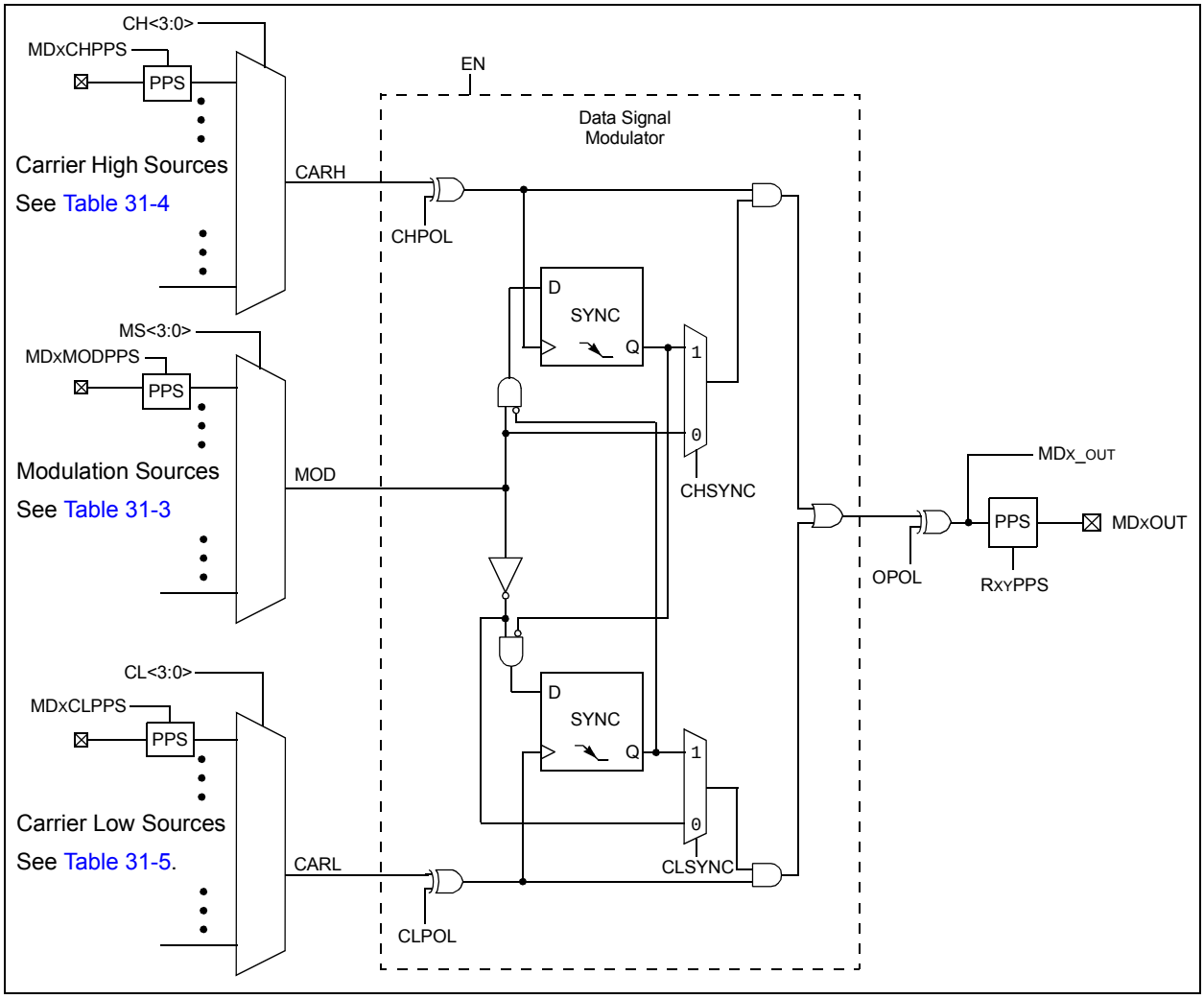
- Frequency-Shift Keying (FSK)
- Phase-Shift Keying (PSK)
- On-Off Keying (OOK)

Additionally, the following features are provided within the DSM module:

- Carrier Synchronization
- Carrier Source Polarity Select
- Carrier Source Pin Disable
- Programmable Modulator Data
- Modulator Source Pin Disable
- Modulated Output Polarity Select
- Slew Rate Control

Figure 31-1 shows a Simplified Block Diagram of the Data Signal Modulator peripheral.

**FIGURE 31-1: SIMPLIFIED BLOCK DIAGRAM OF THE DATA SIGNAL MODULATOR**



## 31.1 DSM Operation

The DSM module is enabled by setting the EN bit in the MDxCON register. Clearing the EN bit in the MDxCON register disables the DSM module by automatically switching the carrier high and carrier low signals to the VSS signal source. The modulator signal source is also switched to the BIT bit in the MDxCON0 register. This not only assures that the DSM module is inactive, but that it is also consuming the least amount of current.

The values used to select the carrier high, carrier low, and modulator sources held by the Modulation Source, Modulation High Carrier, and Modulation Low Carrier control registers are not affected when the EN bit is cleared and the DSM module is disabled. The values inside these registers remain unchanged while the DSM is inactive. The sources for the carrier high, carrier low and modulator signals will once again be selected when the EN bit is set and the DSM module is enabled and active.

The modulated output signal can be output on any device I/O pin by selecting the desired DSM module in the pin's PPS control register (see Register 12-2). If the output is not directed to any I/O pin then the DSM module will remain active and continue to mix signals, but the output value will not be sent to any pin.

## 31.2 Modulator Signal Sources

The modulator signal is selected by configuring the MS<4:0> bits of the MDxSRC register. Selections are shown in Table 31-3.

## 31.3 Carrier Signal Sources

The carrier high signal is selected by configuring the CH<3:0> bits of the MDxCARH register. Selections are shown in Table 31-4.

The carrier low signal is selected by configuring the CL<3:0> bits of the MDxCARL register. Selections are shown in Table 31-5.

## 31.4 Carrier Synchronization

During the time when the DSM switches between carrier high and carrier low signal sources, the carrier data in the modulated output signal can become truncated. To prevent this, the carrier signal can be synchronized to the modulator signal. When synchronization is enabled, the carrier pulse that is being mixed at the time of the transition is allowed to transition low before the DSM switches over to the next carrier source.

Synchronization is enabled separately for the carrier high and carrier low signal sources. Synchronization for the carrier high signal is enabled by setting the CHSYNC bit of the MDxCON1 register. Synchronization for the carrier low signal is enabled by setting the CLSYNC bit of the MDxCON1 register.

Figure 31-1 through Figure 31-6 show timing diagrams of using various synchronization methods.

**FIGURE 31-2: ON OFF KEYING (OOK) SYNCHRONIZATION**

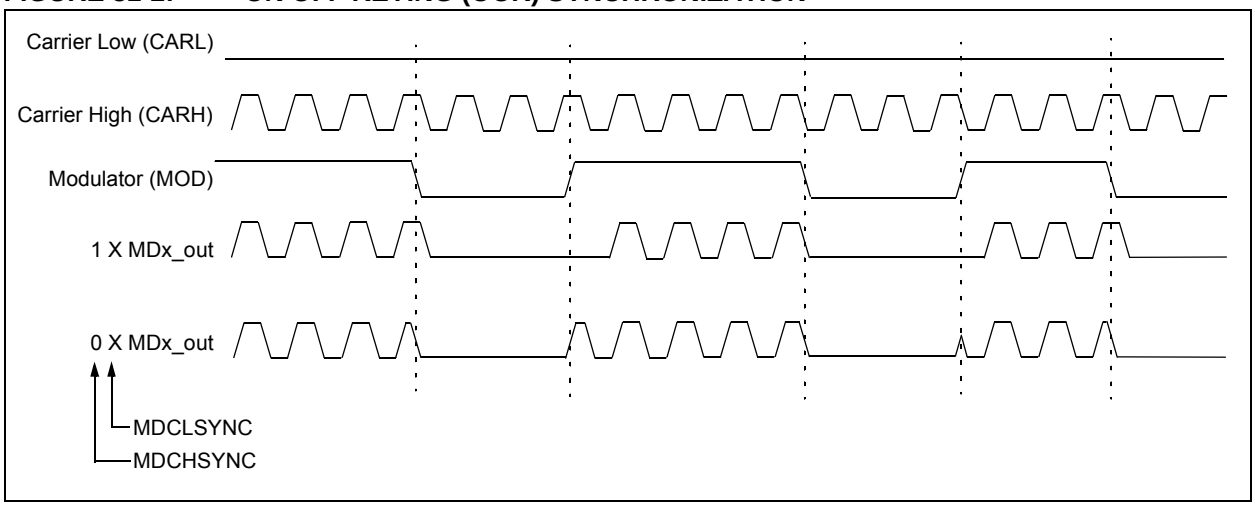



**FIGURE 31-3: NO SYNCHRONIZATION (MDCHSYNC = 0, MDCLSYNC = 0)**

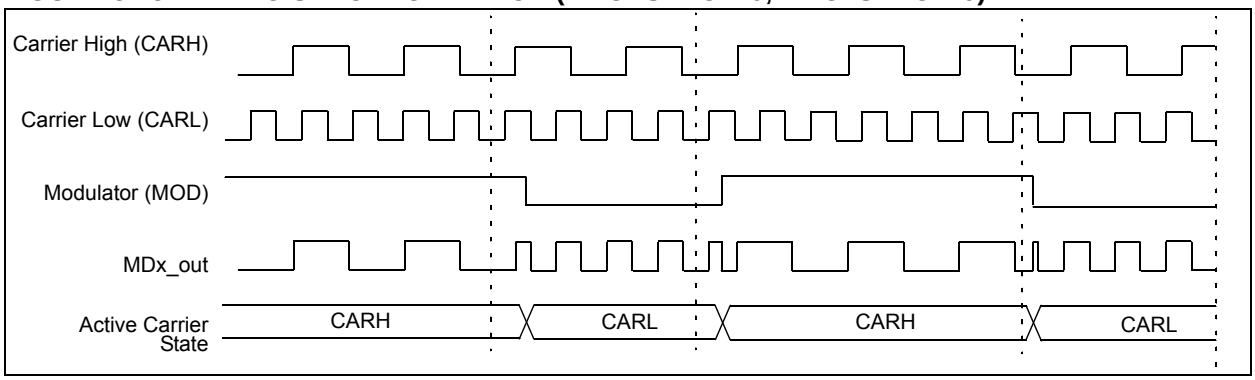



**FIGURE 31-4: CARRIER HIGH SYNCHRONIZATION (MDCHSYNC = 1, MDCLSYNC = 0)**

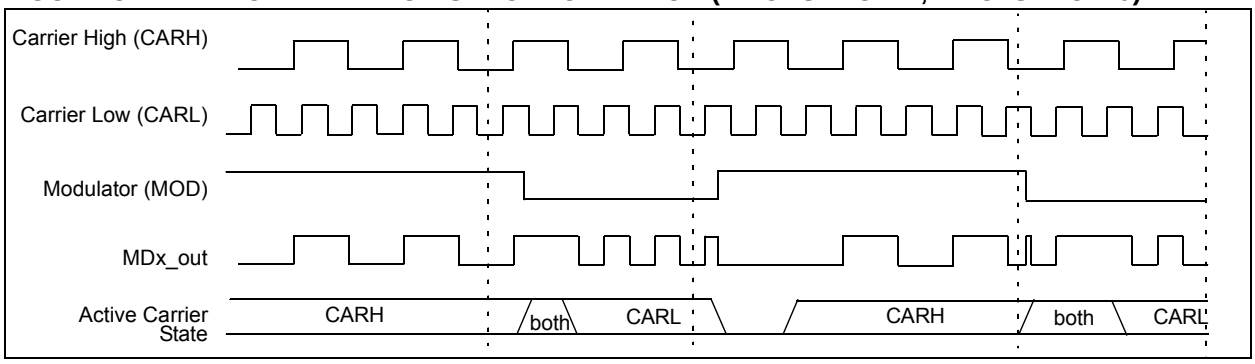


**FIGURE 31-5: CARRIER LOW SYNCHRONIZATION (MDCHSYNC = 0, MDCLSYNC = 1)**

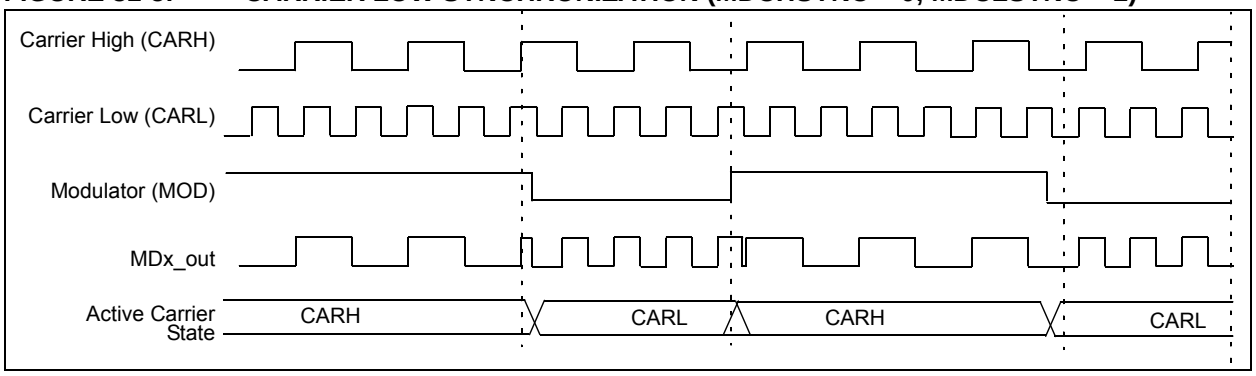
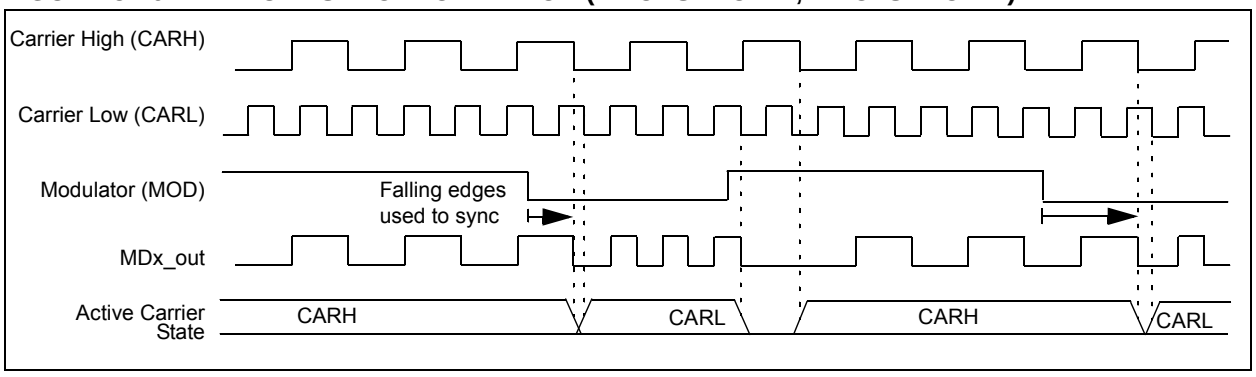



**FIGURE 31-6: FULL SYNCHRONIZATION (MDCHSYNC = 1, MDCLSYNC = 1)**



## 31.5 Input and Output Through Pins

The modulation and carrier sources may be selected to come from any device pin with the PPS control logic. Selecting a pin requires two settings: The source selection determines that the PPS will be used and the PPS control selects the desired pin. Source and PPS registers are identified in Table 31-1. PPS register pin selections are shown in Register 12-1 and Register 12-2.

**TABLE 31-1:**

| Source | Source Register | PPS Register |
|---|---|---|
| Modulation | MDxSRC | MDxMODPPS |
| Carrier High | MDxCARH | MDxCHPPS |
| Carrier Low | MDxCARL | MDxCLPPS |

Any device pin can be selected as the modulation output with the individual pin PPS controls. See Register 12-2 for the pin output selections.

## 31.6 Carrier Source Polarity Select

The signal provided from any selected input source for the carrier high and carrier low signals can be inverted. Inverting the signal for the carrier high source is enabled by setting the CHPOL bit of the MDxCON1 register. Inverting the signal for the carrier low source is enabled by setting the CLPOL bit of the MDxCON1 register.

## 31.7 Programmable Modulator Data

The BIT bit of the MDxCON0 register can be selected as the source for the modulator signal. When the BIT source is selected then software generates the modulation signal by setting and clearing the BIT bit at the respective desired modulation high and low times.

## 31.8 Modulated Output Polarity

The modulated output signal provided on the MDxOUT pin can also be inverted. Inverting the modulated output signal is enabled by setting the OPOL bit of the MDxCON0 register.

## 31.9 Operation in Sleep Mode

The DSM module is not affected by Sleep mode. The DSM will operate during Sleep provided that the Carrier and Modulator input sources are also active during Sleep.

## 31.10 Effects of a Reset

Upon any device Reset, the data signal modulator module is disabled. The user's firmware is responsible for initializing the module before enabling the output. The registers are reset to their default values.

## 31.11 Register Definitions: Data Signal Modulator

Long bit name prefixes for the DSM peripherals are shown in Table 31-2. Refer to **Section 1.1.2.2 "Long Bit Names"** for more information

**TABLE 31-2:**

| Peripheral | Bit Name Prefix |
|---|---|
| DSM1 | MD1 |
| DSM2 | MD2 |
| DSM3 | MD3 |

**REGISTER 31-1: MDxCON0: MODULATION CONTROL REGISTER 0**

| R/W-0/0 | U-0 | R-0/0 | R/W-0/0 | U-0 | U-0 | U-0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| EN | — | OUT | OPOL | — | — | — | BIT |
| 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 **EN:** Modulator Module Enable bit
- 1 = Modulator module is enabled and mixing input signals
- 0 = Modulator module is disabled and has no output

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

bit 5 **OUT:** Modulator Output bit
Displays the current output value of the modulator module.(1)

bit 4 **OPOL:** Modulator Output Polarity Select bit
- 1 = Modulator output signal is inverted. Idle high output.
- 0 = Modulator output signal is not inverted. Idle low output.

bit 3-1 **Unimplemented:** Read as '0'

bit 0 **BIT:** Allows direct software control of the modulation source input to module(2)
- 1 = Modulator uses High Carrier source
- 0 = Modulator uses Low Carrier source

**Note 1:** The modulated output frequency can be greater and asynchronous from the clock that updates this register bit, the bit value may not be valid for higher speed modulator or carrier signals.

**2:** BIT must be selected as the modulation source in the MDSRC register for this operation.

**REGISTER 31-2: MDxCON1: MODULATION CONTROL REGISTER 1**

| U-0 | U-0 | R/W-0/0 | R/W-0/0 | U-0 | U-0 | R/W-0/0 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| — | — | CHPOL | CHSYNC | — | — | CLPOL | CLSYNC |
| 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 **CHPOL:** Modulation High Carrier Polarity Select bit
1 = Selected high carrier source is inverted
0 = Selected high carrier source is not inverted

bit 4 **CHSYNC:** Modulation High Carrier Synchronization Enable bit
1 = Modulator waits for a low edge on the high carrier before allowing a switch to the low carrier
0 = Modulator output is not synchronized to the high carrier(1)

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

bit 1 **CLPOL:** Modulation Low Carrier Polarity Select bit
1 = Selected low carrier source is inverted
0 = Selected low carrier source is not inverted

bit 0 **CLSYNC:** Modulation Low Carrier Synchronization Enable bit
1 = Modulator waits for a low edge on the low carrier before allowing a switch to the high carrier
0 = Modulator output is not synchronized to the low carrier(1)

**Note 1:** Narrowed carrier pulse widths or spurs may occur in the signal stream if the carrier is not synchronized.

**REGISTER 31-3: MDxSRC: MODULATION SOURCE CONTROL REGISTER**

| U-0 | U-0 | U-0 | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
|---|---|---|---|---|---|---|---|
| — | — | — | MS<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 **MS<4:0>** Modulation Source Selection bits
See Table 31-3.

**TABLE 31-3: MODULATION SOURCE**

| MS<4:0> | Modulation Source PIC16(L)F1773/6 |
|---|---|
| 11111 | Reserved |
| 11110 | Reserved |
| 11101 | Reserved |
| 11100 | Reserved |
| 11011 | Reserved |
| 11010 | sync_C6OUT |
| 11001 | sync_C5OUT |
| 11000 | sync_C4OUT |
| 10111 | sync_C3OUT |
| 10110 | sync_C2OUT |
| 10101 | sync_C1OUT |
| 10100 | LC4_out |
| 10011 | LC3_out |
| 10010 | LC2_out |
| 10001 | LC1_out |
| 10000 | Reserved |
| 01111 | PWM11_out |
| 01110 | PWM6_out |
| 01101 | PWM5_out |
| 01100 | Reserved |
| 01011 | PWM9_out |
| 01010 | PWM4_out |
| 01001 | PWM3_out |
| 01000 | Reserved |
| 00111 | CCP7_out |
| 00110 | CCP2_out |
| 00101 | CCP1_out |
| 00100 | SDO_OUT |
| 00011 | DT |
| 00010 | TX_out |
| 00001 | MDxBIT |
| 00000 | MDxMODPPS pin selection |

**REGISTER 31-4: MDxCARH: MODULATION HIGH CARRIER CONTROL REGISTER**

| U-0 | U-0 | U-0 | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
|---|---|---|---|---|---|---|---|
| — | — | — | CH<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 **CH<4:0>** Modulator Data High Carrier Selection bits(1)
See Table 31-4.

**Note 1:** Narrowed carrier pulse widths or spurs may occur in the signal stream if the carrier is not synchronized.

**TABLE 31-4: HIGH CARRIER SOURCE**

| CH<4:0> | Carrier Source PIC16(L)F1773/6 |
|---|---|
| 11111 | Reserved |
| 11110 | Reserved |
| 11101 | Reserved |
| 11100 | Reserved |
| 11011 | Reserved |
| 11010 | Reserved |
| 11001 | Reserved |
| 11000 | Reserved |
| 10111 | Reserved |
| 10110 | Reserved |
| 10101 | Reserved |
| 10100 | Reserved |
| 10011 | Reserved |
| 10010 | LC4_out |
| 10001 | LC3_out |
| 10000 | LC2_out |
| 01111 | LC1_out |
| 01110 | Reserved |
| 01101 | PWM11_out |
| 01100 | PWM6_out |
| 01011 | PWM5_out |
| 01010 | Reserved |
| 01001 | PWM9_out |
| 01000 | PWM4_out |
| 00111 | PWM3_out |
| 00110 | Reserved |
| 00101 | CCP3_out |
| 00100 | CCP2_out |
| 00011 | CCP1_out |
| 00010 | HFINTOSC |
| 00001 | FOSC |
| 00000 | MDxCHPPS pin selection |

**REGISTER 31-5: MDxCARL: MODULATION LOW CARRIER CONTROL REGISTER**

| U-0 | U-0 | U-0 | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u | R/W-x/u |
|---|---|---|---|---|---|---|---|
| — | — | — | CL<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 **CL<4:0>** Modulator Data Low Carrier Selection bits(1)
See Table 31-5.

**Note 1:** Narrowed carrier pulse widths or spurs may occur in the signal stream if the carrier is not synchronized.

**TABLE 31-5: LOW CARRIER SOURCE**

| CH<4:0> | Carrier Source PIC16(L)F1773/6 |
|---|---|
| 11111 | Reserved |
| 11110 | Reserved |
| 11101 | Reserved |
| 11100 | Reserved |
| 11011 | Reserved |
| 11010 | Reserved |
| 11001 | Reserved |
| 11000 | Reserved |
| 10111 | Reserved |
| 10110 | Reserved |
| 10101 | Reserved |
| 10100 | Reserved |
| 10011 | Reserved |
| 10010 | LC4_out |
| 10001 | LC3_out |
| 10000 | LC2_out |
| 01111 | LC1_out |
| 01110 | Reserved |
| 01101 | PWM11_out |
| 01100 | PWM6_out |
| 01011 | PWM5_out |
| 01010 | Reserved |
| 01001 | PWM9_out |
| 01000 | PWM4_out |
| 00111 | PWM3_out |
| 00110 | Reserved |
| 00101 | CCP3_out |
| 00100 | CCP2_out |
| 00011 | CCP1_out |
| 00010 | HFINTOSC |
| 00001 | FOSC |
| 00000 | MDxCLPPS pin selection |

**TABLE 31-6: SUMMARY OF REGISTERS ASSOCIATED WITH DATA SIGNAL MODULATOR MODE**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| MDxCARH | — | — | — | CH<4:0> | | | | | 397 |
| MDxCARL | — | — | — | CL<4:0> | | | | | 399 |
| MDxSRC | — | — | — | MS<4:0> | | | | | 395 |
| MDxCON0 | EN | — | OUT | OPOL | — | — | — | BIT | 394 |
| MDxCON1 | — | — | CHPOL | CHSYNC | — | — | CLPOL | CLSYNC | 394 |

**Legend:** — = unimplemented, read as '0'. Shaded cells are not used in the Data Signal Modulator mode.

# 32.0 MASTER SYNCHRONOUS SERIAL PORT (MSSP) MODULE

## 32.1 MSSP Module Overview

The Master Synchronous Serial Port (MSSP) 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 MSSP module can operate in one of two modes:

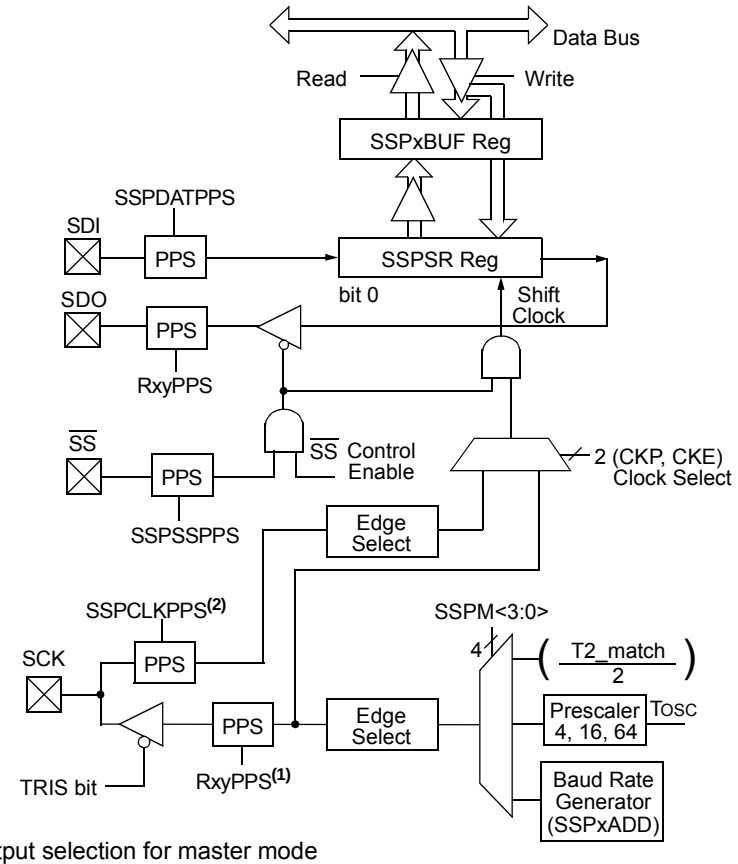
- Serial Peripheral Interface (SPI)
- Inter-Integrated Circuit ($I^2C$)

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 32-1 is a block diagram of the SPI interface module.

**FIGURE 32-1: MSSP BLOCK DIAGRAM (SPI MODE)**




**Note 1:** Output selection for master mode

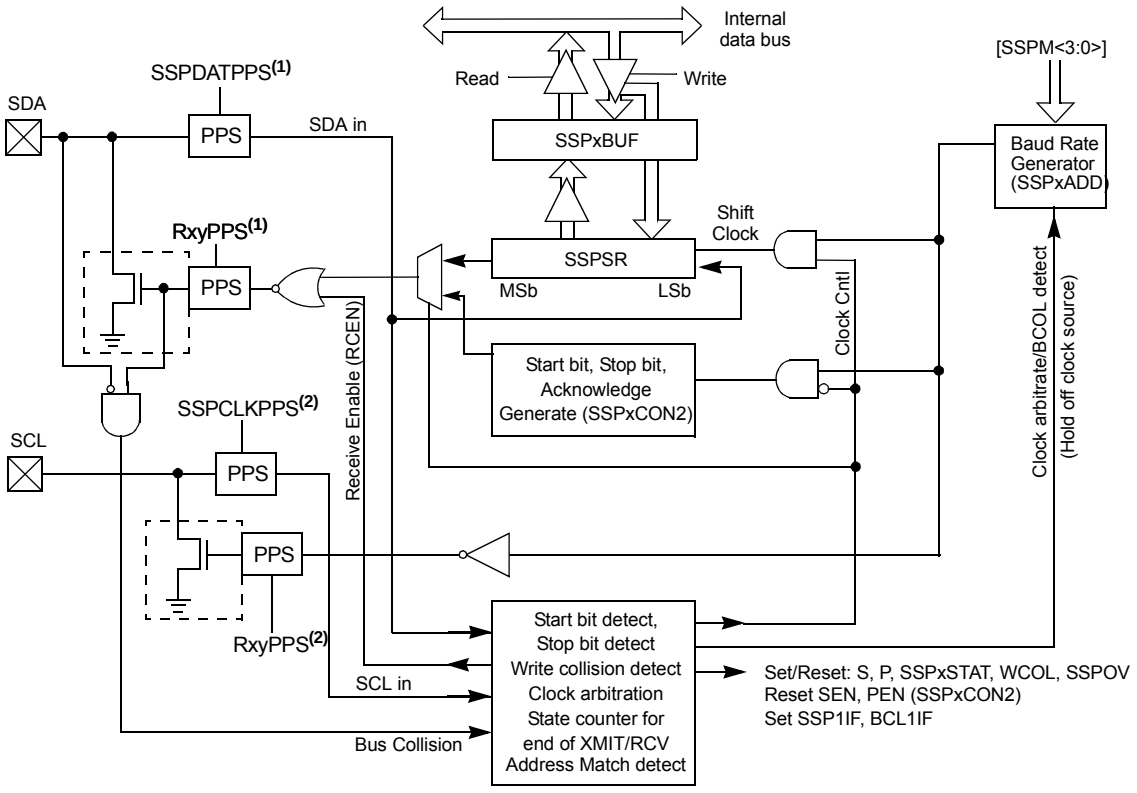
**2:** Input selection for slave mode

The I²C 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 SDA hold times

Figure 32-2 is a block diagram of the I²C interface module in Master mode. Figure 32-3 is a diagram of the I²C interface module in Slave mode.

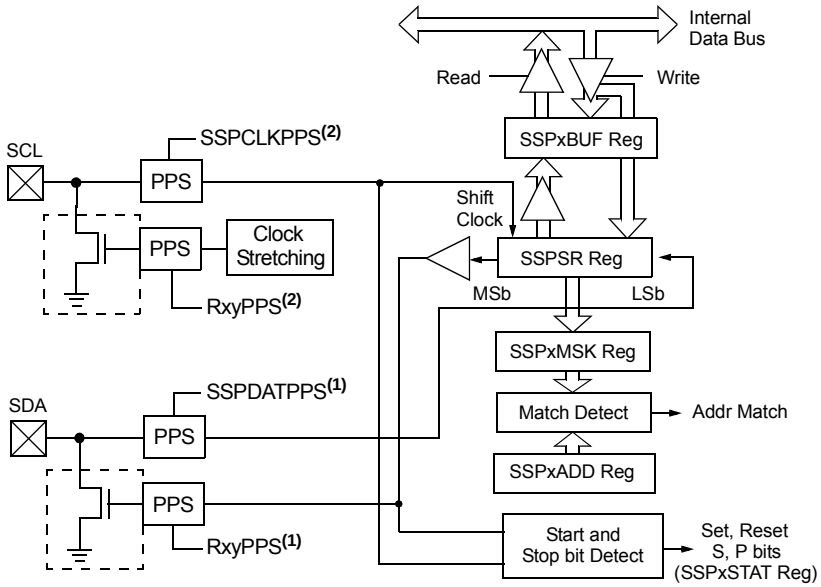
**FIGURE 32-2: MSSP BLOCK DIAGRAM (I²C MASTER MODE)**




**Note 1:** SDA pin selections must be the same for input and output

**2:** SCL pin selections must be the same for input and output

**FIGURE 32-3: MSSP BLOCK DIAGRAM ($I^2C$ SLAVE MODE)**




**Note 1:** SDA pin selections must be the same for input and output

**2:** SCL pin selections must be the same for input and output

## 32.2 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 (SCK)
- Serial Data Out (SDO)
- Serial Data In (SDI)
- Slave Select ($\overline{SS}$)

Figure 32-1 shows the block diagram of the MSSP 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 32-4 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 32-5 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 SDO output pin which is connected to, and received by, the slave's SDI input pin. The slave device transmits information out on its SDO output pin, which is connected to, and received by, the master's SDI 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 SDO 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 SDO 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.

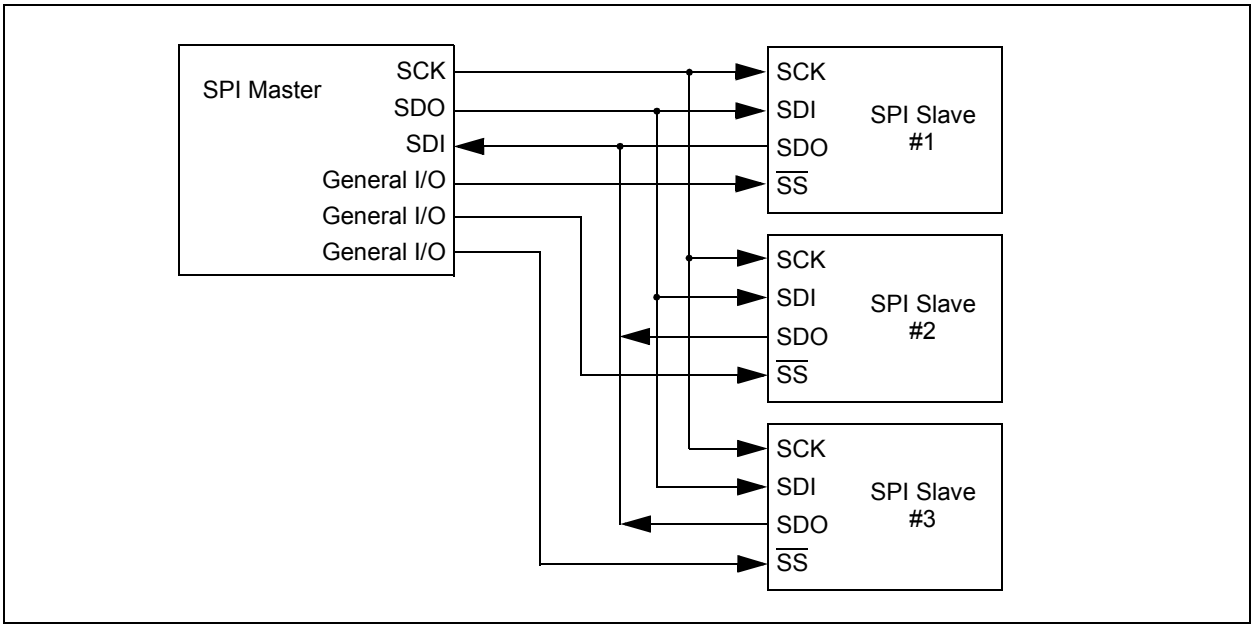
Whether the data is meaningful or not (dummy data), depends on the application software. This leads to three scenarios for data transmission:

- 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.

**FIGURE 32-4: SPI MASTER AND MULTIPLE SLAVE CONNECTION**




### 32.2.1 SPI MODE REGISTERS

The MSSP module has five registers for SPI mode operation. These are:

- MSSP STATUS register (SSPxSTAT)
- MSSP Control register 1 (SSPxCON1)
- MSSP Control register 3 (SSPxCON3)
- MSSP Data Buffer register (SSPxBUF)
- MSSP Address register (SSPxADD)
- MSSP Shift register (SSPSR) (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 one 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 32.7 "Baud Rate Generator"**.

SSPSR is the shift register used for shifting data in and out. SSPxBUF provides indirect access to the SSPSR register. SSPxBUF is the buffer register to which data bytes are written, and from which data bytes are read.

In receive operations, SSPSR and SSPxBUF together create a buffered receiver. When SSPSR 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 SSPSR.

### 32.2.2 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 (SCK is the clock output)
- Slave mode (SCK is the clock input)
- Clock Polarity (Idle state of SCK)
- Data Input Sample Phase (middle or end of data output time)
- Clock Edge (output data on rising/falling edge of SCK)
- Clock Rate (Master mode only)
- Slave Select mode (Slave mode only)

To enable the serial port, SSP 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 SDI, SDO, SCK and $\overline{SS}$ 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:

- SDI must have corresponding TRIS bit set
- SDO must have corresponding TRIS bit cleared
- SCK (Master mode) must have corresponding TRIS bit cleared
- SCK (Slave mode) must have corresponding TRIS bit set
- $\overline{SS}$ must have corresponding TRIS bit set

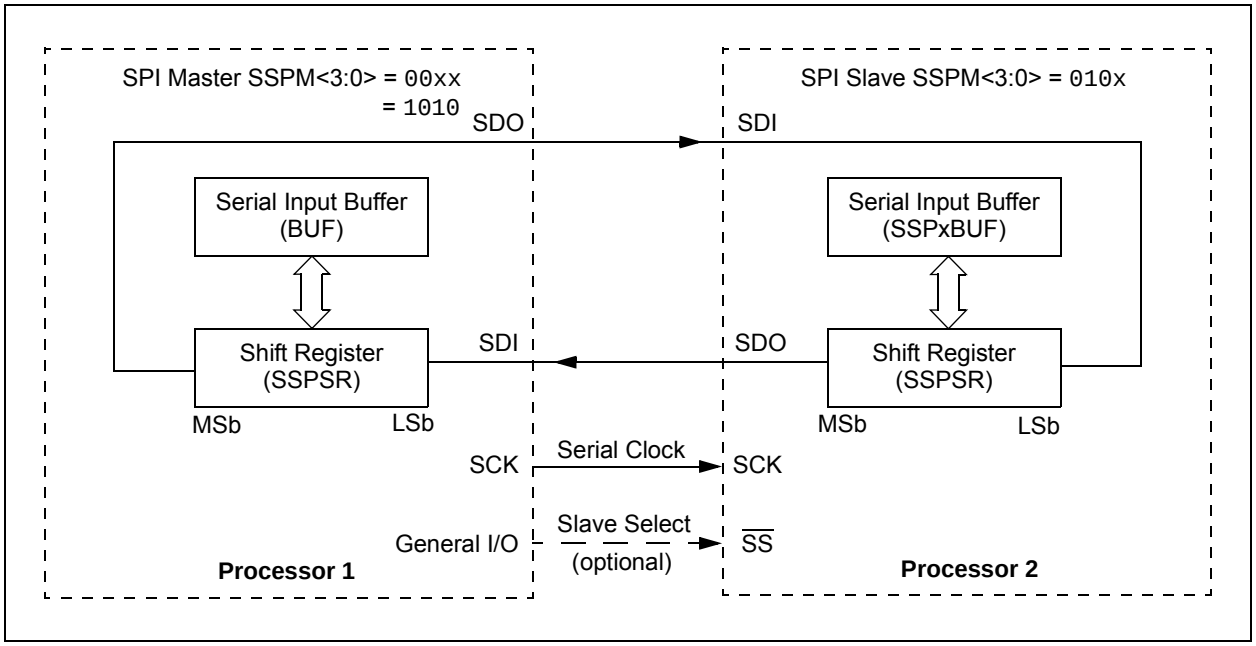
Any serial port function that is not desired may be overridden by programming the corresponding data direction (TRIS) register to the opposite value.

The MSSP consists of a transmit/receive shift register (SSPSR) and a buffer register (SSPxBUF). The SSPSR shifts the data in and out of the device, MSb first. The SSPxBUF holds the data that was written to the SSPSR 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 MSSP 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 SSPSR 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 32-5: SPI MASTER/SLAVE CONNECTION**



### 32.2.3 SPI MASTER MODE

The master can initiate the data transfer at any time because it controls the SCK line. The master determines when the slave (Processor 2, Figure 32-5) 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 SDO output could be disabled (programmed as an input). The SSPSR register will continue to shift in the signal present on the SDI 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 32-6, Figure 32-8, Figure 32-9 and Figure 32-10, where the MSB is transmitted first. In Master mode, the SPI clock rate (bit rate) is user programmable to be one of the following:

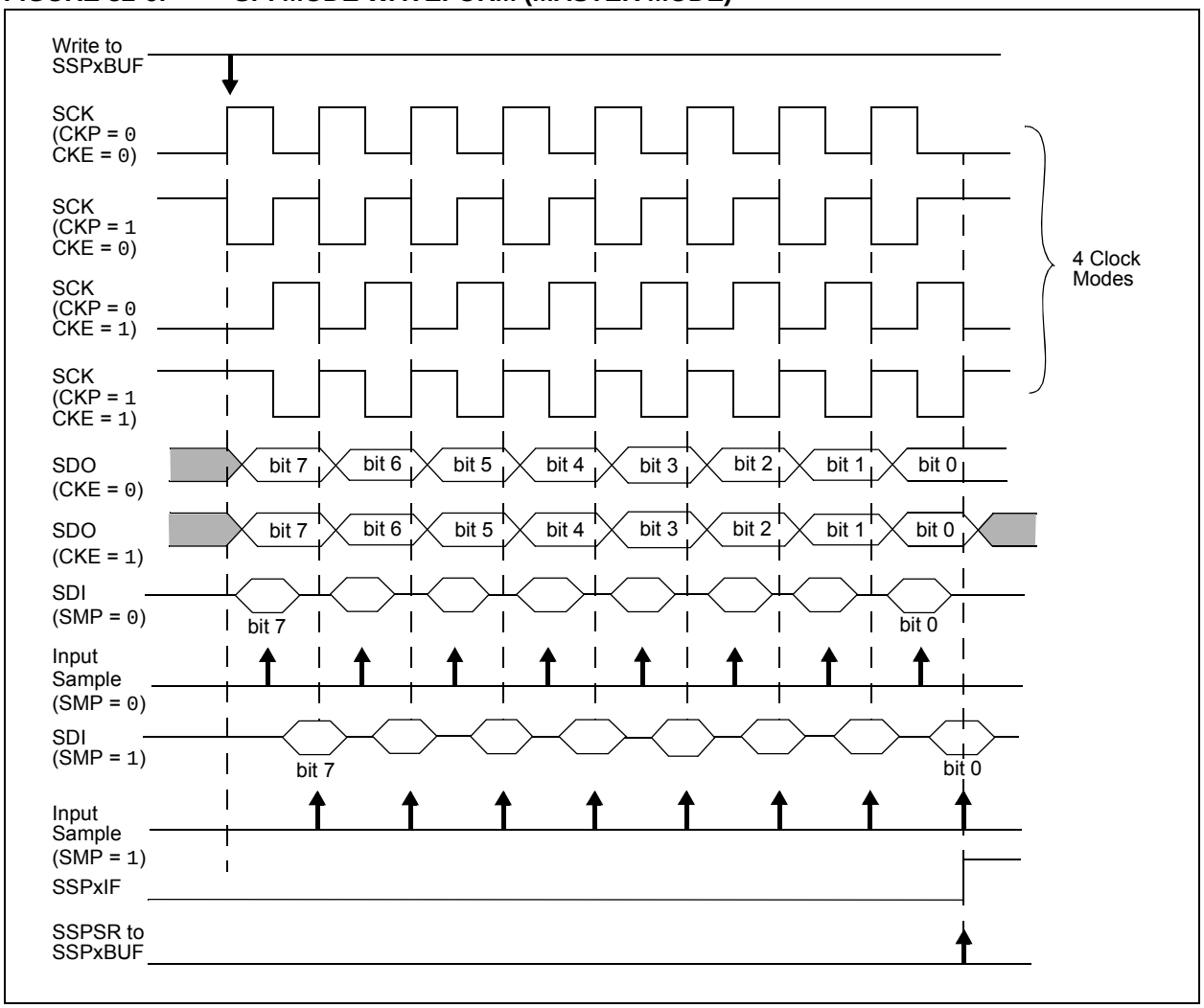
- FOSC/4 (or TCY)
- FOSC/16 (or 4 * TCY)
- FOSC/64 (or 16 * TCY)
- Timer2 output/2
- FOSC/(4 * (SSPxADD + 1))

Figure 32-6 shows the waveforms for Master mode.

When the CKE bit is set, the SDO data is valid before there is a clock edge on SCK. 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.

| **Note:** | In Master mode the clock signal output to the SCK pin is also the clock signal input to the peripheral. The pin selected for output with the RxyPPS register must also be selected as the peripheral input with the SSPCLKPPS register. |
|---|---|

**FIGURE 32-6: SPI MODE WAVEFORM (MASTER MODE)**




### 32.2.4 SPI SLAVE MODE

In Slave mode, the data is transmitted and received as external clock pulses appear on SCK. 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 SCK 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 SCK 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 SCK pin input and when a byte is received, the device will generate an interrupt. If enabled, the device will wake-up from Sleep.

#### 32.2.4.1 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 32-7 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.

### 32.2.5 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 $\overline{SS}$ pin allows a Synchronous Slave mode. The SPI must be in Slave mode with $\overline{SS}$ pin control enabled (SSPxCON1<3:0> = 0100).

When the $\overline{SS}$ pin is low, transmission and reception are enabled and the SDO pin is driven.

When the $\overline{SS}$ pin goes high, the SDO 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 $\overline{SS}$ pin control enabled (SSPxCON1<3:0> = 0100), the SPI module will reset if the $\overline{SS}$ pin is set to VDD. |
| **2:** | When the SPI is used in Slave mode with CKE set; the user must enable $\overline{SS}$ 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 $\overline{SS}$ pin to a high level or clearing the SSPEN bit.

**FIGURE 32-7: SPI DAISY-CHAIN CONNECTION**

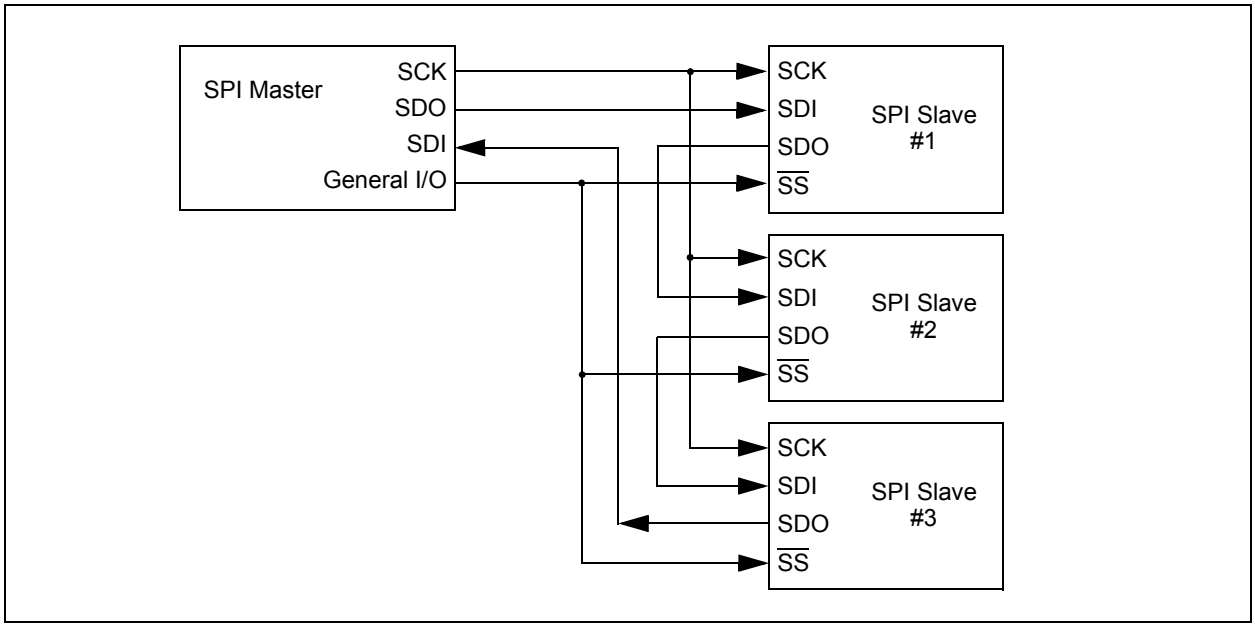


**FIGURE 32-8: SLAVE SELECT SYNCHRONOUS WAVEFORM**

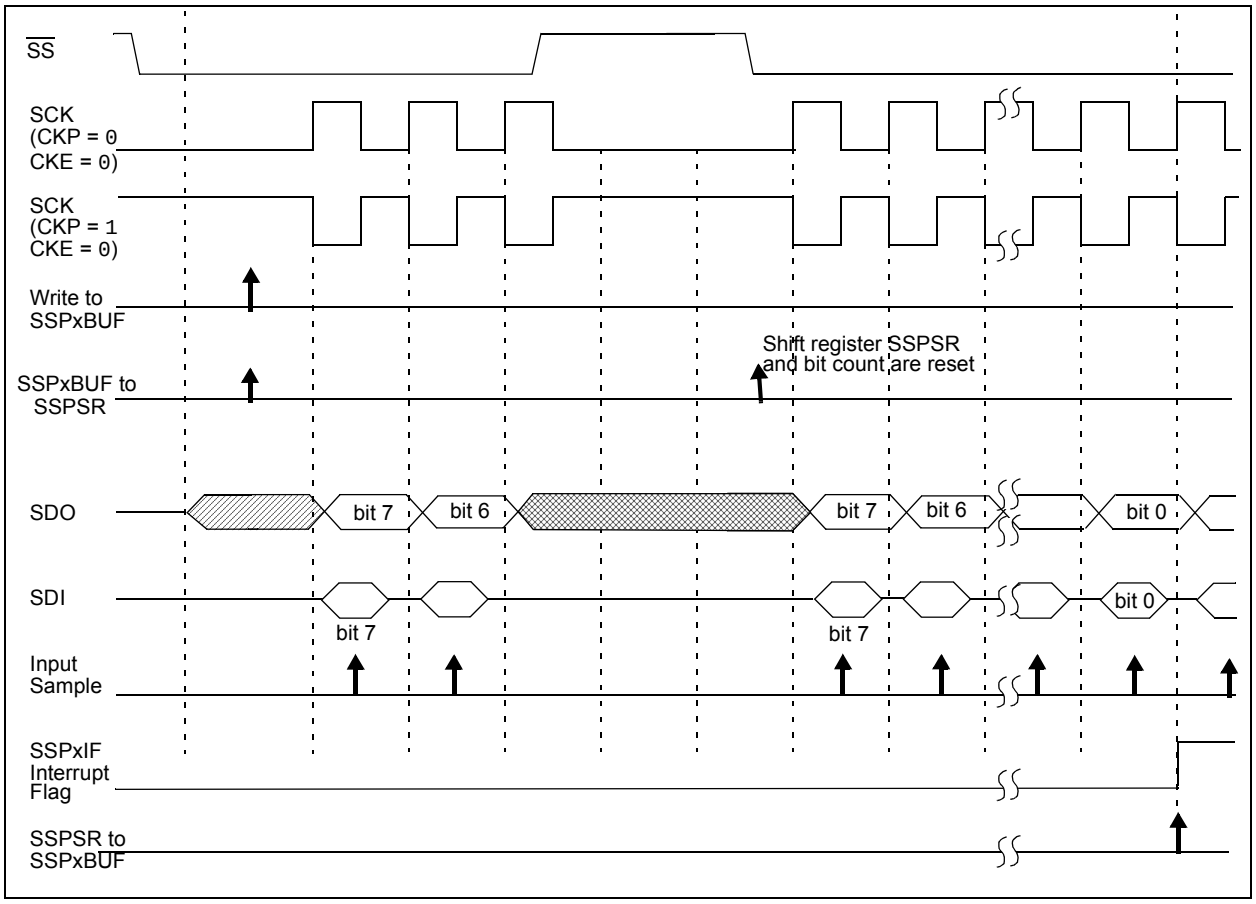


**FIGURE 32-9: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 0)**

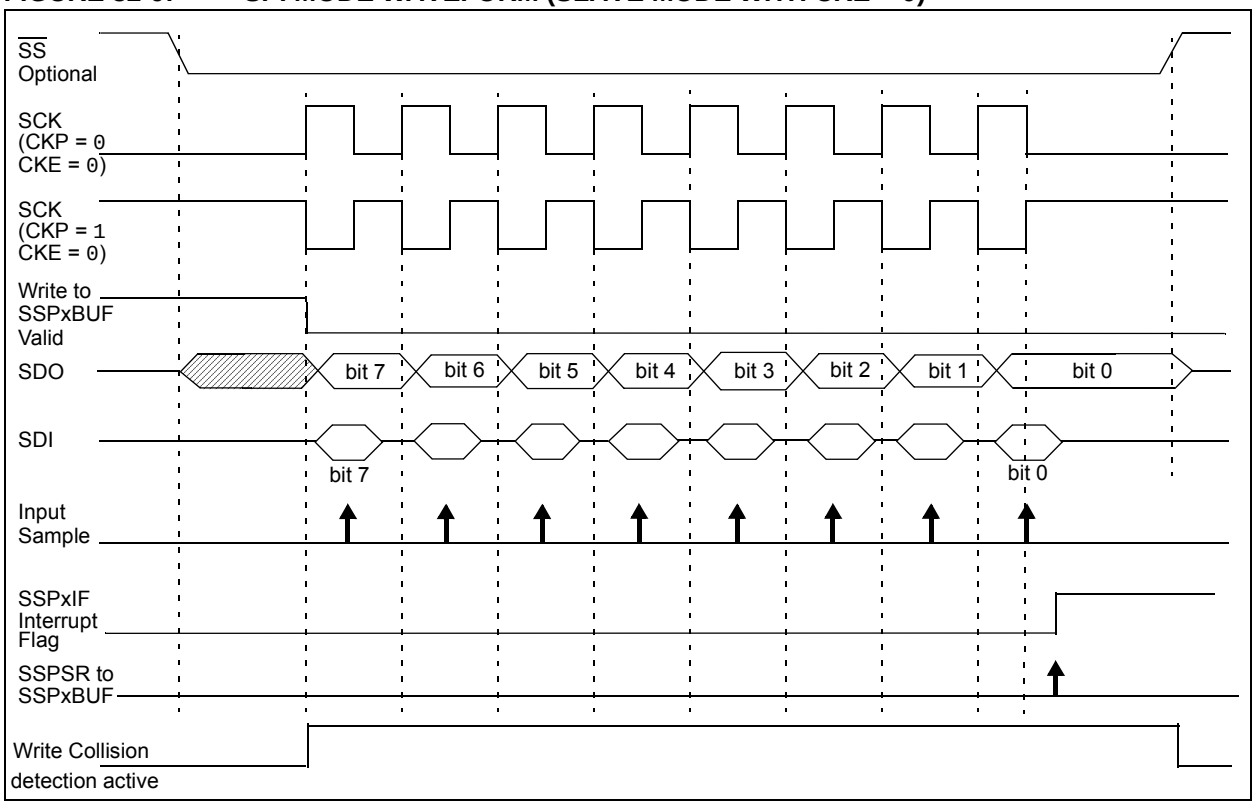
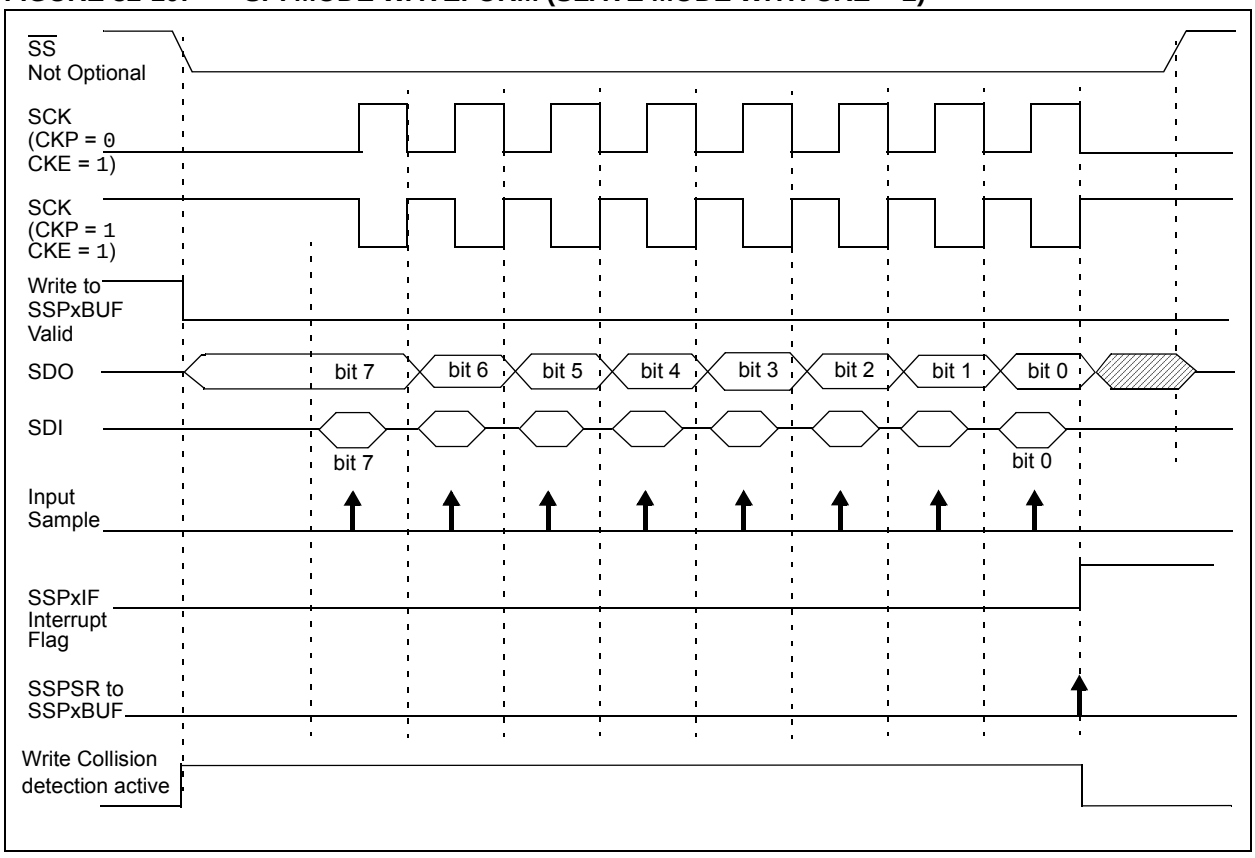



**FIGURE 32-10: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 1)**



### 32.2.6 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 MSSP clock is much faster than the system clock.

In Slave mode, when MSSP interrupts are enabled, after the master completes sending data, an MSSP interrupt will wake the controller from Sleep.

If an exit from Sleep mode is not desired, MSSP 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 MSSP interrupt flag bit will be set and if enabled, will wake the device.

**TABLE 32-1: 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 |
|---|---|---|---|---|---|---|---|---|---|
| ANSELA | — | — | ANSA5 | ANSA4 | ANSA3 | ANSA2 | ANSA1 | ANSA0 | 153 |
| ANSELC | ANSC7 | ANSC6 | ANSC5 | ANSC4 | ANSC3 | ANSC2 | — | — | 163 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 108 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 109 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 115 |
| RxyPPS | — | — | RxyPPS<5:0> | | | | | | 172 |
| SSPCLKPPS | — | — | SSPCLKPPS<5:0> | | | | | | 172, 174 |
| SSPDATPPS | — | — | SSPDATPPS<5:0> | | | | | | 172, 174 |
| SSPSSPPS | — | — | SSPSSPPS<5:0> | | | | | | 172, 174 |
| SSP1BUF | Synchronous Serial Port Receive Buffer/Transmit Register | | | | | | | | 405* |
| SSP1CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM<3:0> | | | | 450 |
| SSP1CON3 | ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN | 449 |
| SSP1STAT | SMP | CKE | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF | 449 |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 152 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 157 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 162 |

**Legend:** — = Unimplemented location, read as '0'. Shaded cells are not used by the MSSP in SPI mode.

* Page provides register information.

## 32.3 I²C MODE OVERVIEW

The Inter-Integrated Circuit (I²C) bus 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 (SCL)
- Serial Data (SDA)

Figure 32-11 shows the block diagram of the MSSP module when operating in I²C mode.

Both the SCL and SDA 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 32-11 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:

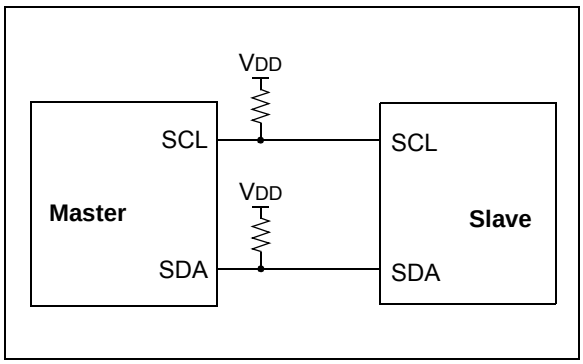
- Master Transmit mode
  (master is transmitting data to a slave)
- Master Receive mode
  (master is receiving data from a slave)
- Slave Transmit mode
  (slave is transmitting data to a master)
- Slave Receive mode
  (slave is receiving data from the master)

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 SDA line while the SCL 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 32-11: I²C MASTER/ SLAVE CONNECTION**




The Acknowledge bit ($\overline{\text{ACK}}$) is an active-low signal, which holds the SDA 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 SCL line is held low. Transitions that occur while the SCL 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 SDA line while the SCL 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.

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 SCL 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 SDA line, it is called arbitration. Arbitration ensures that there is only one master device communicating at any single time.

### 32.3.1 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 SCL 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 SCL line in order to transfer the next bit, but will detect that the clock line has not yet been released. Because the SCL 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.

### 32.3.2 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 SDA 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 SDA line.

For example, if one transmitter holds the SDA 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 SDA 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 SDA line. If this transmitter is also a master device, it also must stop driving the SCL 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 SDA 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.

## 32.4 I²C MODE OPERATION

All MSSP I²C communication is byte oriented and shifted out MSb first. Six SFR registers and two interrupt flags interface the module with the PIC® microcontroller and user software. Two pins, SDA and SCL, are exercised by the module to communicate with other external I²C devices.

### 32.4.1 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 SCL line, the device outputting data on the SDA changes that pin to an input and reads in an acknowledge value on the next clock pulse.

The clock signal, SCL, is provided by the master. Data is valid to change while the SCL signal is low, and sampled on the rising edge of the clock. Changes on the SDA line while the SCL line is high define special conditions on the bus, explained below.

### 32.4.2 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.

### 32.4.3 SDA AND SCL PINS

Selection of any I²C mode with the SSPEN bit set, forces the SCL and SDA pins to be open-drain. These pins should be set by the user to inputs by setting the appropriate TRIS bits.

**Note 1:** Data is tied to output zero when an I²C mode is enabled.

**2:** Any device pin can be selected for SDA and SCL functions with the PPS peripheral. These functions are bidirectional. The SDA input is selected with the SSPDATPPS registers. The SCL input is selected with the SSPCLKPPS registers. Outputs are selected with the RxyPPS registers. It is the user's responsibility to make the selections so that both the input and the output for each function is on the same pin.

### 32.4.4 SDA HOLD TIME

The hold time of the SDA pin is selected by the SDAHT bit of the SSPxCON3 register. Hold time is the time SDA is held valid after the falling edge of SCL. Setting the SDAHT bit selects a longer 300 ns minimum hold time and may help on buses with large capacitance.

**TABLE 32-2: 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 SDA and SCL 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 SCL low to stall communication. |
| Bus Collision | Any time the SDA line is sampled low by the module while it is outputting and expected high state. |

### 32.4.5 START CONDITION

The $I^2C$ specification defines a Start condition as a transition of SDA from a high to a low state while SCL 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 32-12 shows wave forms for Start and Stop conditions.

A bus collision can occur on a Start condition if the module samples the SDA 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.

### 32.4.6 STOP CONDITION

A Stop condition is a transition of the SDA line from low-to-high state while the SCL line is high.

**Note:** At least one SCL low time must appear before a Stop is valid, therefore, if the SDA line goes low then high again while the SCL line stays high, only the Start condition is detected.

### 32.4.7 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. Figure 32-13 shows the wave form for a Restart condition.

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.

### 32.4.8 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 32-12: $I^2C$ START AND STOP CONDITIONS**

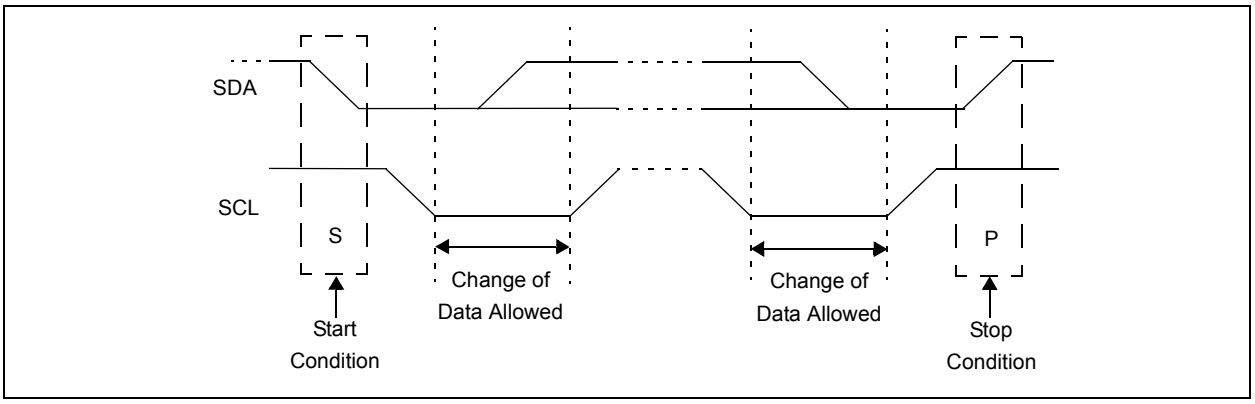
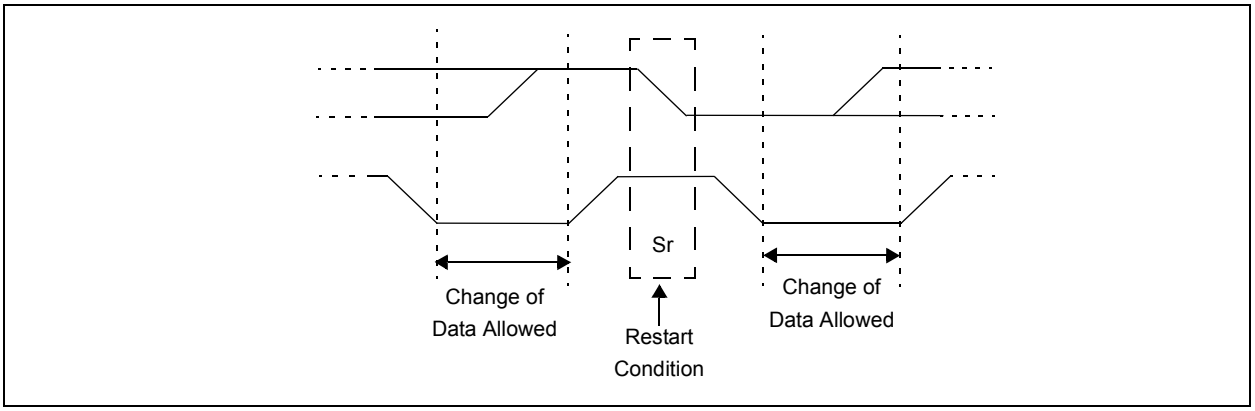



**FIGURE 32-13: $I^2C$ RESTART CONDITION**



### 32.4.9 ACKNOWLEDGE SEQUENCE

The ninth SCL pulse for any transferred byte in I$^2$C is dedicated as an Acknowledge. It allows receiving devices to respond back to the transmitter by pulling the SDA 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 SDA line low indicates 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, allow 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.

There are certain conditions where an $\overline{ACK}$ will not be sent by the slave. If the BF bit of the SSPxSTAT register or the SSPOV bit of the SSPxCON1 register are set when a byte is received.

When the module is addressed, after the eighth falling edge of SCL 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.

## 32.5 I$^2$C SLAVE MODE OPERATION

The MSSP Slave mode operates in one of four modes selected by 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 operate the same as the other modes with SSPxIF additionally getting set upon detection of a Start, Restart, or Stop condition.

### 32.5.1 SLAVE MODE ADDRESSES

The SSPxADD register (Register 32-6) 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 SSP Mask register (Register 32-5) affects the address matching process. See **Section 32.5.8 "SSP Mask Register"** for more information.

#### 32.5.1.1 I$^2$C 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.

#### 32.5.1.2 I$^2$C 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's 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 SCL 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 SCL 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.

### 32.5.2 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 not Acknowledge is given. An overflow condition is defined as either bit BF of the SSPxSTAT register is set, or bit SSPOV of the SSPxCON1 register is set. The BOEN bit of the SSPxCON3 register modifies this operation. For more information see Register 32-4.

An MSSP 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, SCL 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 sometimes in 10-bit mode. See **Section 32.5.6.2 "10-bit Addressing Mode"** for more detail.

#### 32.5.2.1 7-bit Addressing Reception

This section describes a standard sequence of events for the MSSP module configured as an $I^2C$ slave in 7-bit Addressing mode. Figure 32-14 and Figure 32-15 is used as a visual reference 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 SDA low sending an $\overline{ACK}$ to the master, and sets 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 SCL line.
8. The master clocks out a data byte.
9. Slave drives SDA 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.

#### 32.5.2.2 7-bit Reception with AHEN and DHEN

Slave device reception with AHEN and DHEN set operate the same as without these options with extra interrupts and clock stretching added after the eighth falling edge of SCL. 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 PMBus™ that was not present on previous versions of this module.

This list describes the steps that need to be taken by slave software to use these options for $I^2C$ communication. Figure 32-16 displays a module using both address and data holding. Figure 32-17 includes the operation with the SEN bit of the SSPxCON2 register set.

1. 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 SCL.
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 SCL even if there is no clock stretching and BF has been cleared. Only if NACK is sent to master is SSPxIF not set

11. SSPxIF set and CKP cleared after eighth falling edge of SCL for a received data byte.
12. Slave looks at 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 will only know by polling the P bit of the SSTSTAT register.

**FIGURE 32-14: I²C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 0, AHEN = 0, DHEN = 0)**

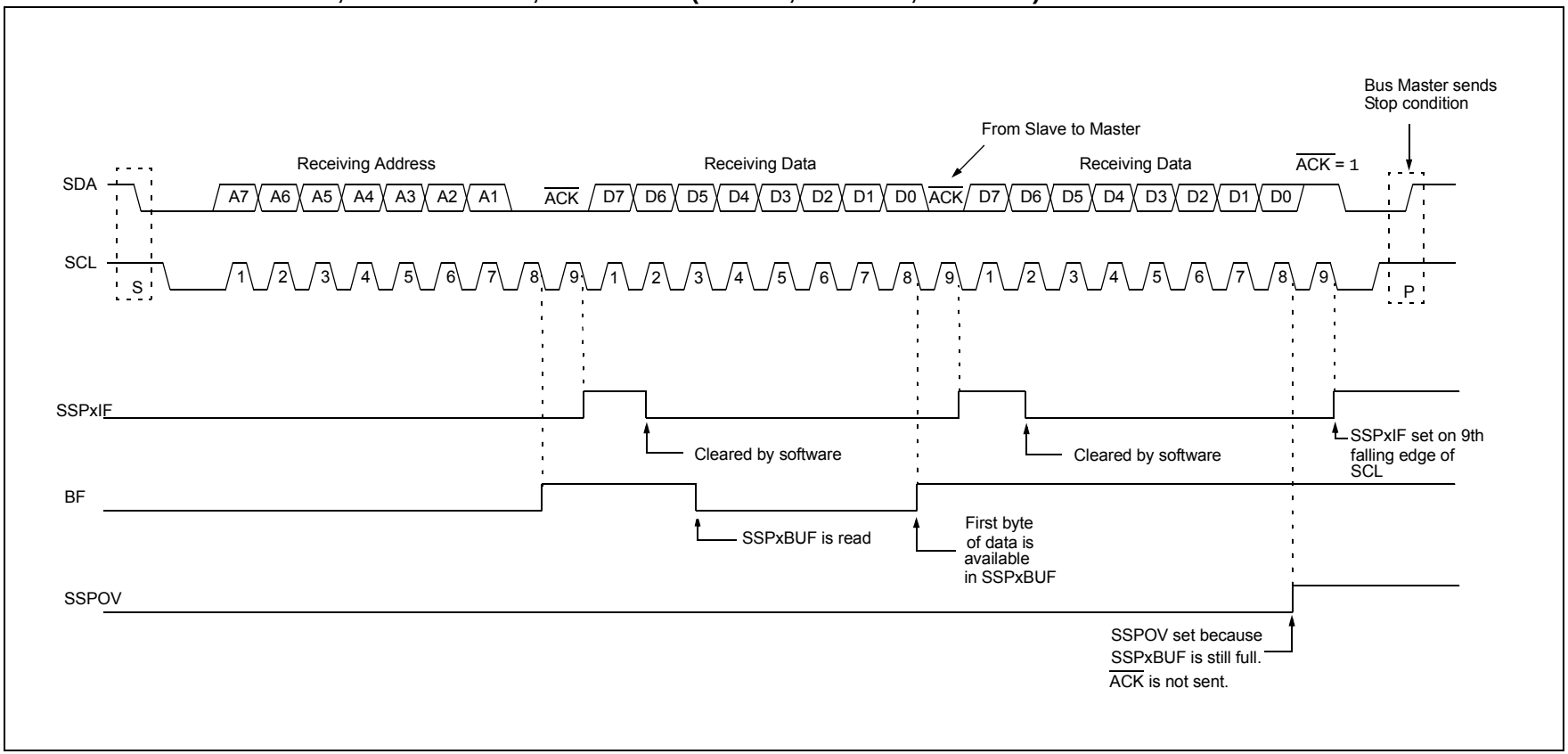


**FIGURE 32-15: I²C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 1, AHEN = 0, DHEN = 0)**

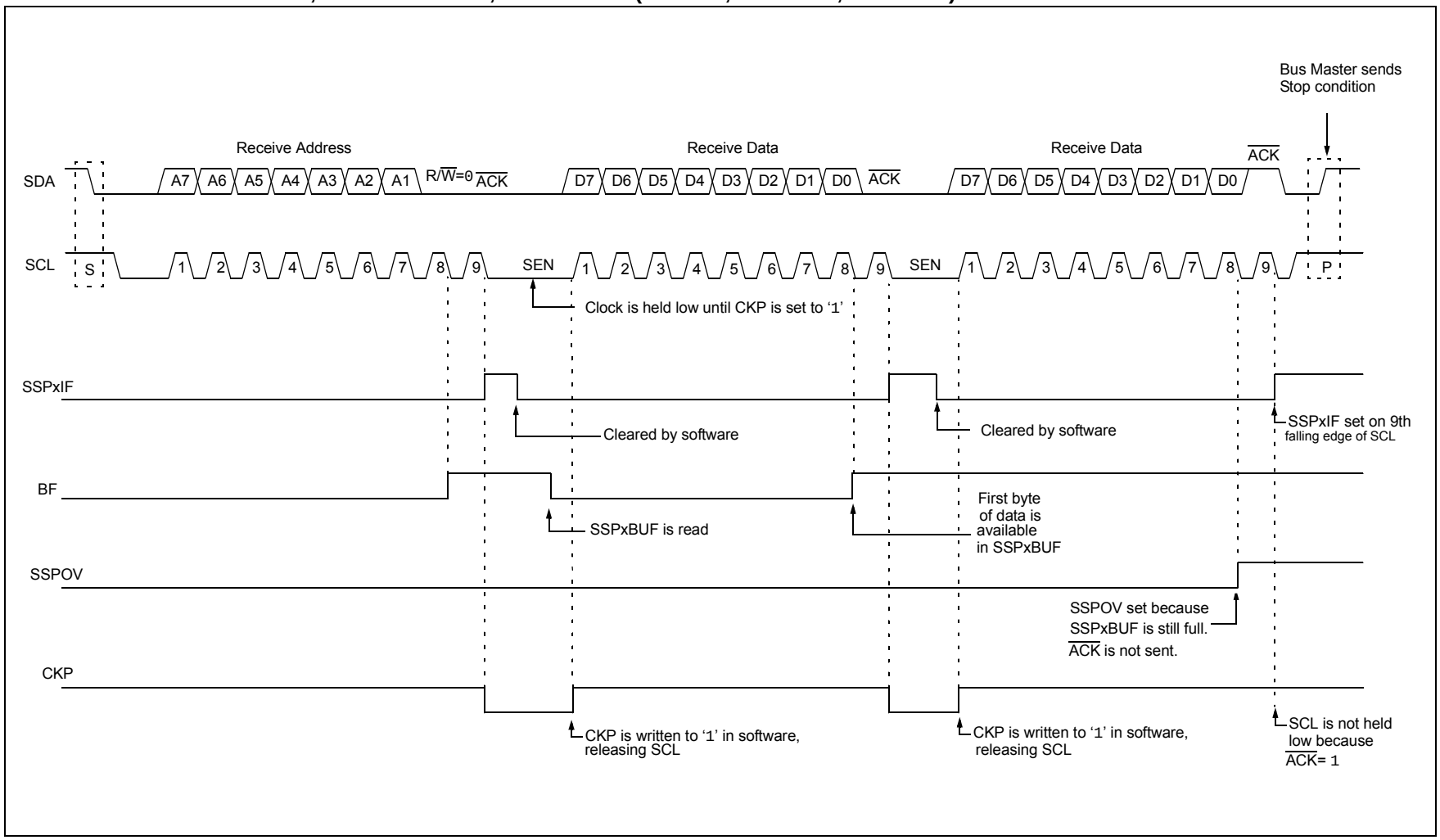


**FIGURE 32-16: I²C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 0, AHEN = 1, DHEN = 1)**

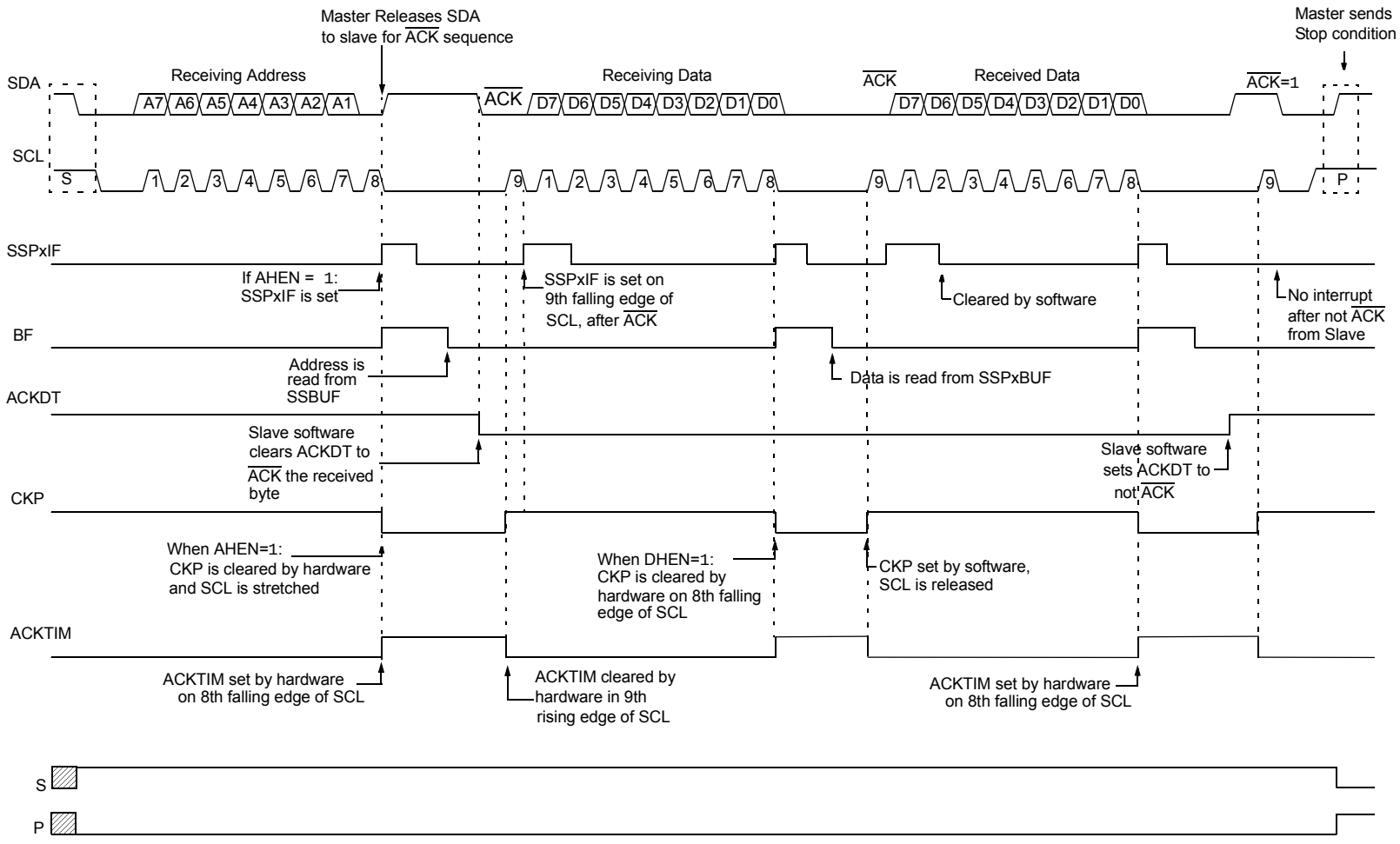
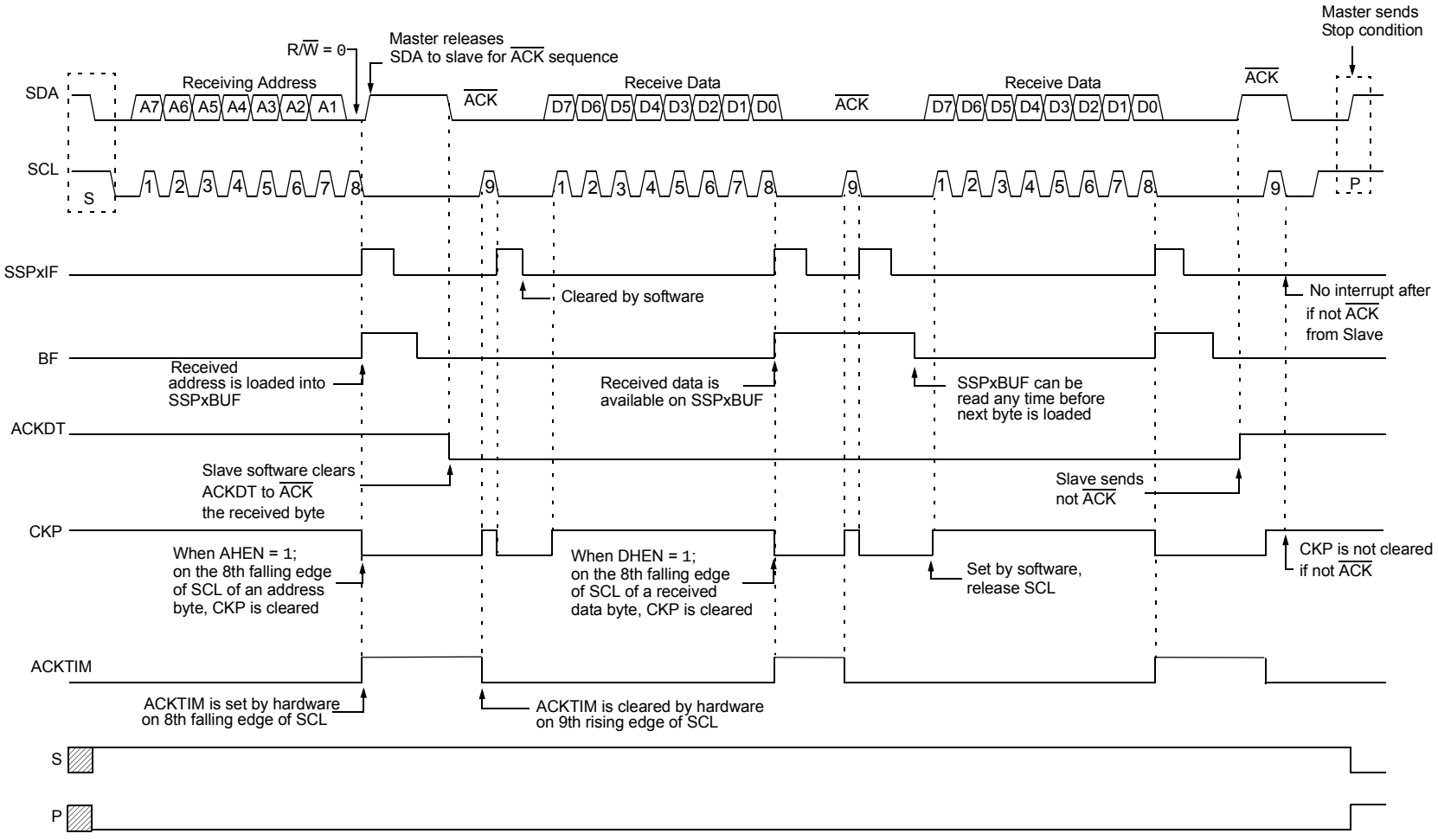


**FIGURE 32-17: I²C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 1, AHEN = 1, DHEN = 1)**



### 32.5.3 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 SCL pin is held low (see **Section 32.5.6 "Clock Stretching"** for more detail). 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 SSPSR register. Then the SCL 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 SCL input. This ensures that the SDA signal is valid during the SCL high time.

The $\overline{ACK}$ pulse from the master-receiver is latched on the rising edge of the ninth SCL 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 SDA line was low ($\overline{ACK}$), the next transmit data must be loaded into the SSPxBUF register. Again, the SCL pin must be released by setting bit CKP.

An MSSP 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.

#### 32.5.3.1 Slave Mode Bus Collision

A slave receives a Read request and begins shifting data out on the SDA line. If a bus collision is detected and the SBCDE bit of the SSPxCON3 register is set, the BCLIF bit of the PIR register is set. Once a bus collision is detected, the slave goes idle and waits to be addressed again. User software can use the BCLIF bit to handle a slave bus collision.

#### 32.5.3.2 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 32-18 can be used as a reference to this list.

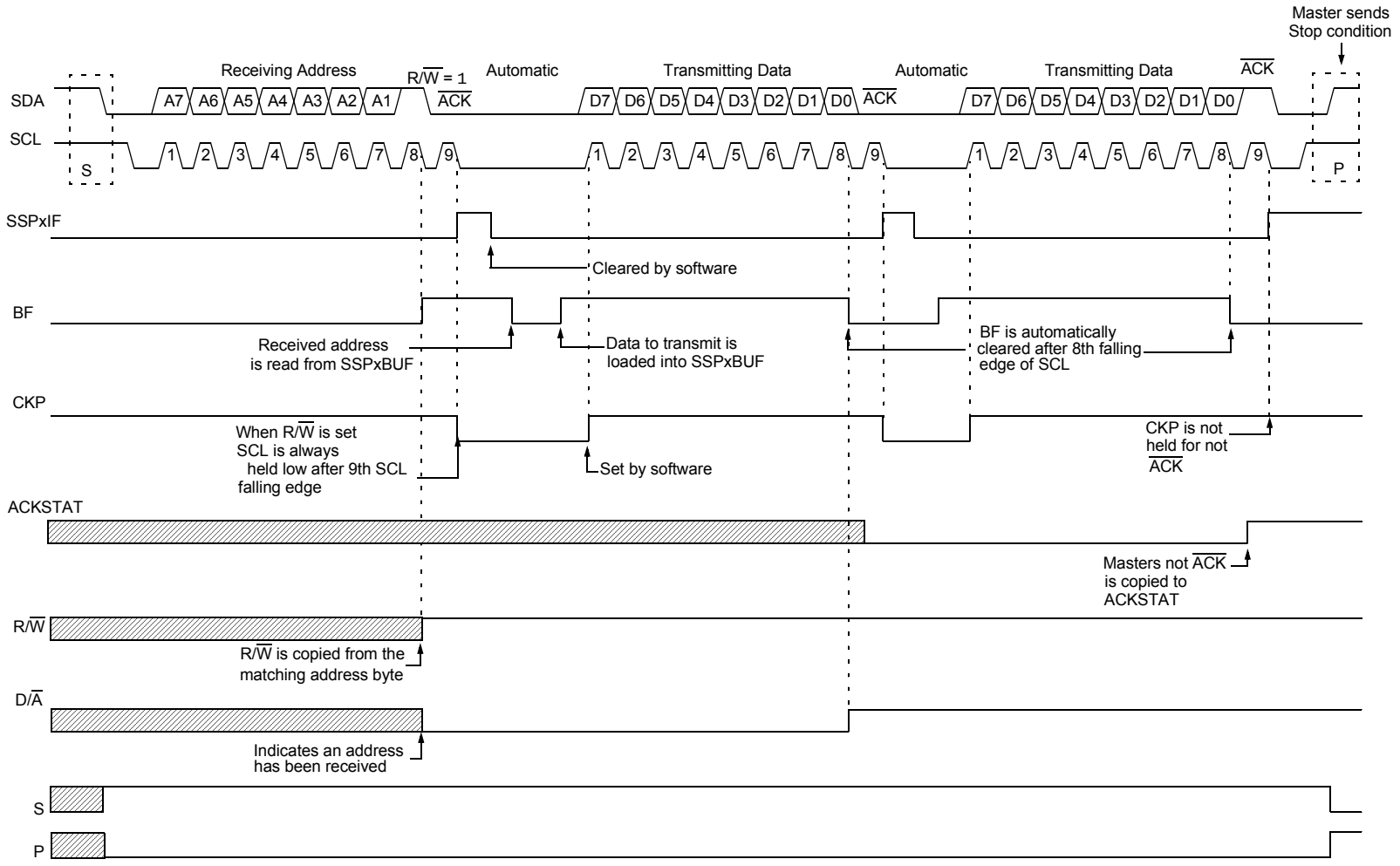
1. Master sends a Start condition on SDA and SCL.
2. S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
3. Matching address with R/$\overline{W}$ bit set is received by the Slave setting SSPxIF bit.
4. Slave hardware generates an $\overline{ACK}$ and sets SSPxIF.
5. SSPxIF bit is cleared by user.
6. Software reads the received address from SSPxBUF, clearing BF.
7. R/$\overline{W}$ is set so CKP was automatically cleared after the $\overline{ACK}$.
8. The slave software loads the transmit data into SSPxBUF.
9. CKP bit is set releasing SCL, 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 SCL (9th) rather than 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.

**FIGURE 32-18: I²C SLAVE, 7-BIT ADDRESS, TRANSMISSION (AHEN = 0)**



#### 32.5.3.3 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 32-19 displays a standard waveform of a 7-bit address slave transmission with AHEN enabled.

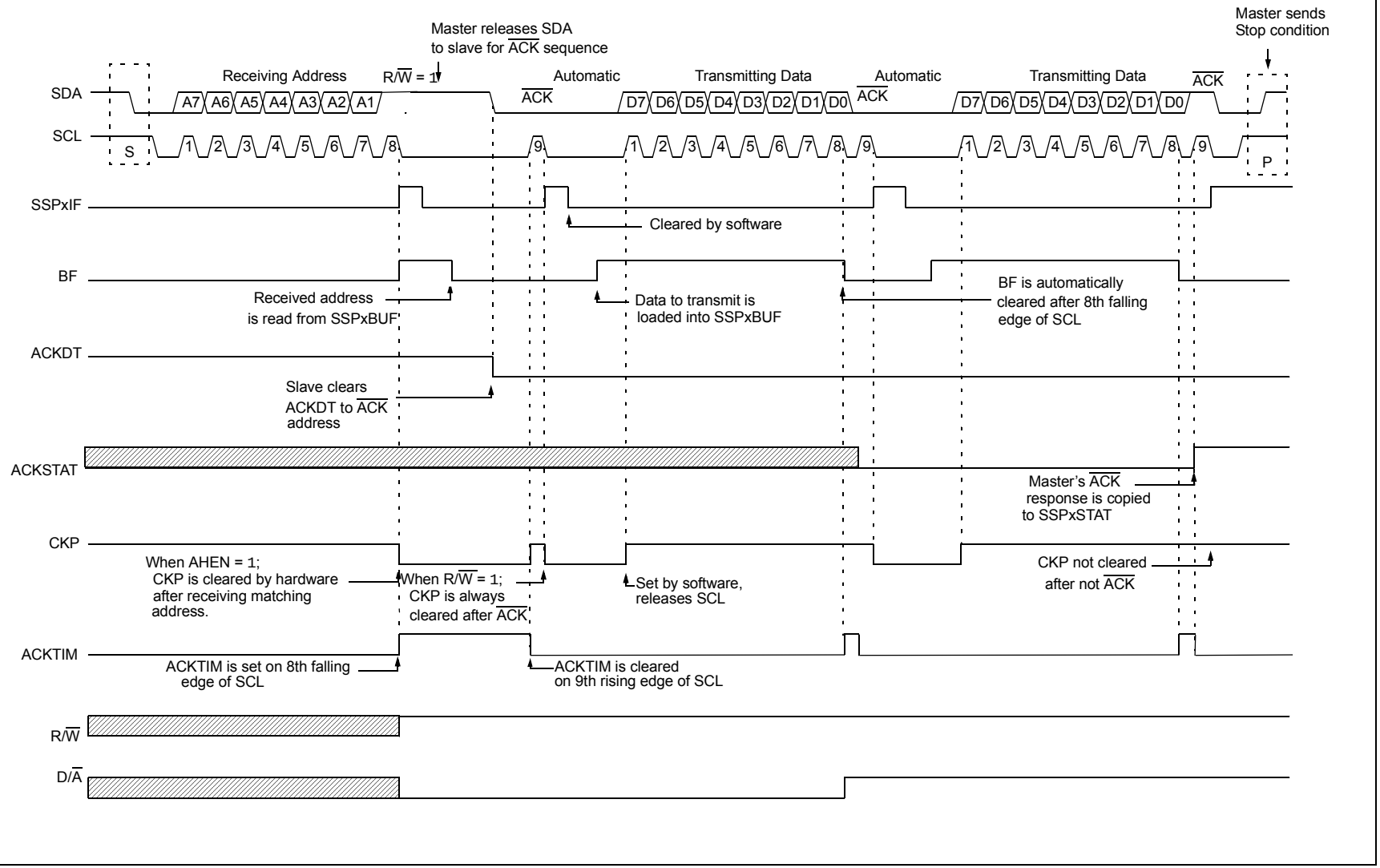
1. Bus starts Idle.
2. Master sends Start condition; the S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
3. Master sends matching address with R/$\overline{W}$ bit set. After the eighth falling edge of the SCL line the CKP bit is cleared and SSPxIF interrupt is generated.
4. Slave software clears SSPxIF.
5. Slave software reads 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 SCL.
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 SCL 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 SCL line to receive a Stop. |
|---|---|

**FIGURE 32-19: I²C SLAVE, 7-BIT ADDRESS, TRANSMISSION (AHEN = 1)**



### 32.5.4 SLAVE MODE 10-BIT ADDRESS RECEPTION

This section describes a standard sequence of events for the MSSP module configured as an $I^2C$ slave in 10-bit Addressing mode.

Figure 32-20 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 SCL.
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 SCL 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 SCL.
18. Steps 13-17 repeat for each received byte.
19. Master sends Stop to end the transmission.

### 32.5.5 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 SCL line is held low are the same. Figure 32-21 can be used as a reference of a slave in 10-bit addressing with AHEN set.

Figure 32-22 shows a standard waveform for a slave transmitter in 10-bit Addressing mode.

**FIGURE 32-20: I²C SLAVE, 10-BIT ADDRESS, RECEPTION (SEN = 1, AHEN = 0, DHEN = 0)**

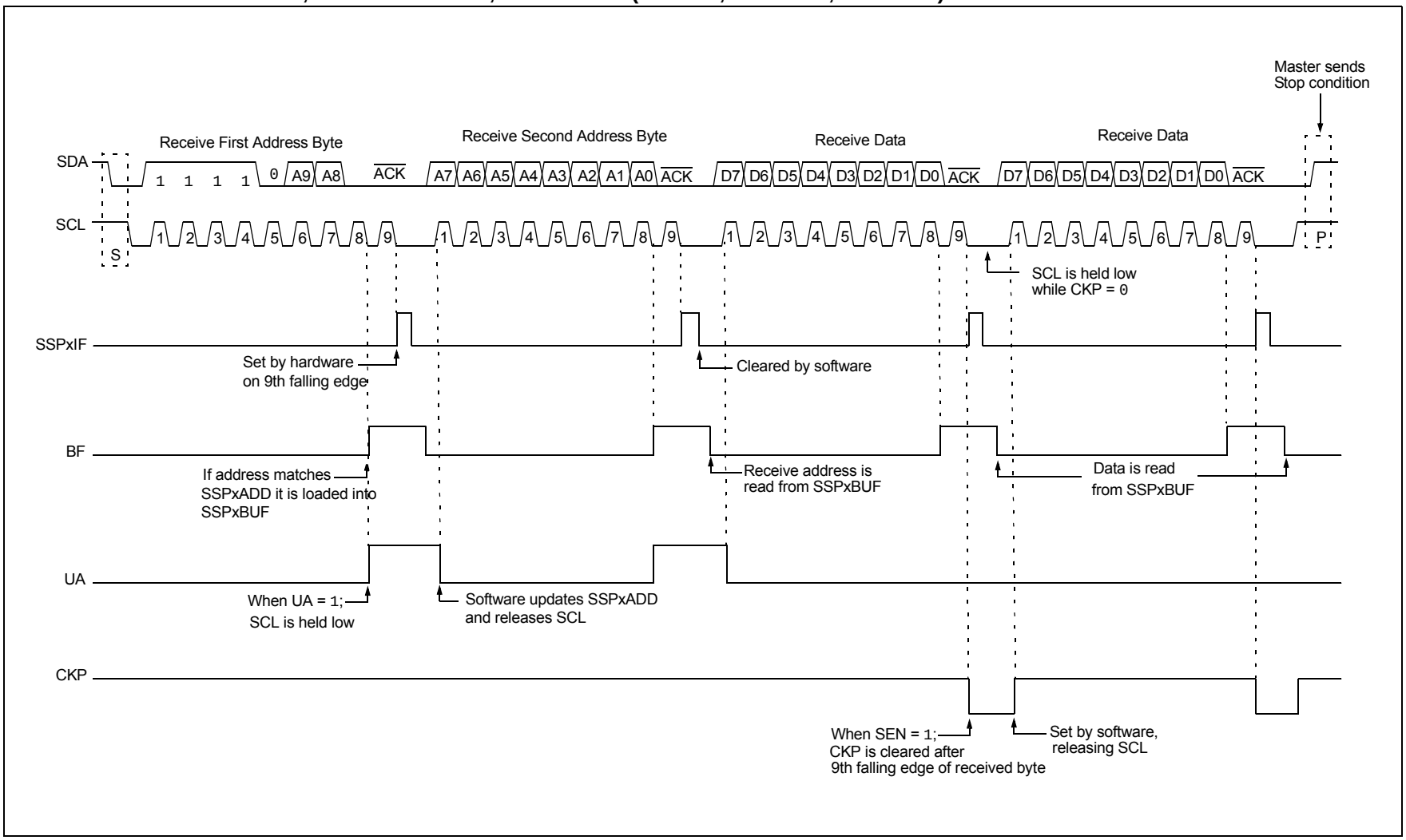


**FIGURE 32-21: I²C SLAVE, 10-BIT ADDRESS, RECEPTION (SEN = 0, AHEN = 1, DHEN = 0)**

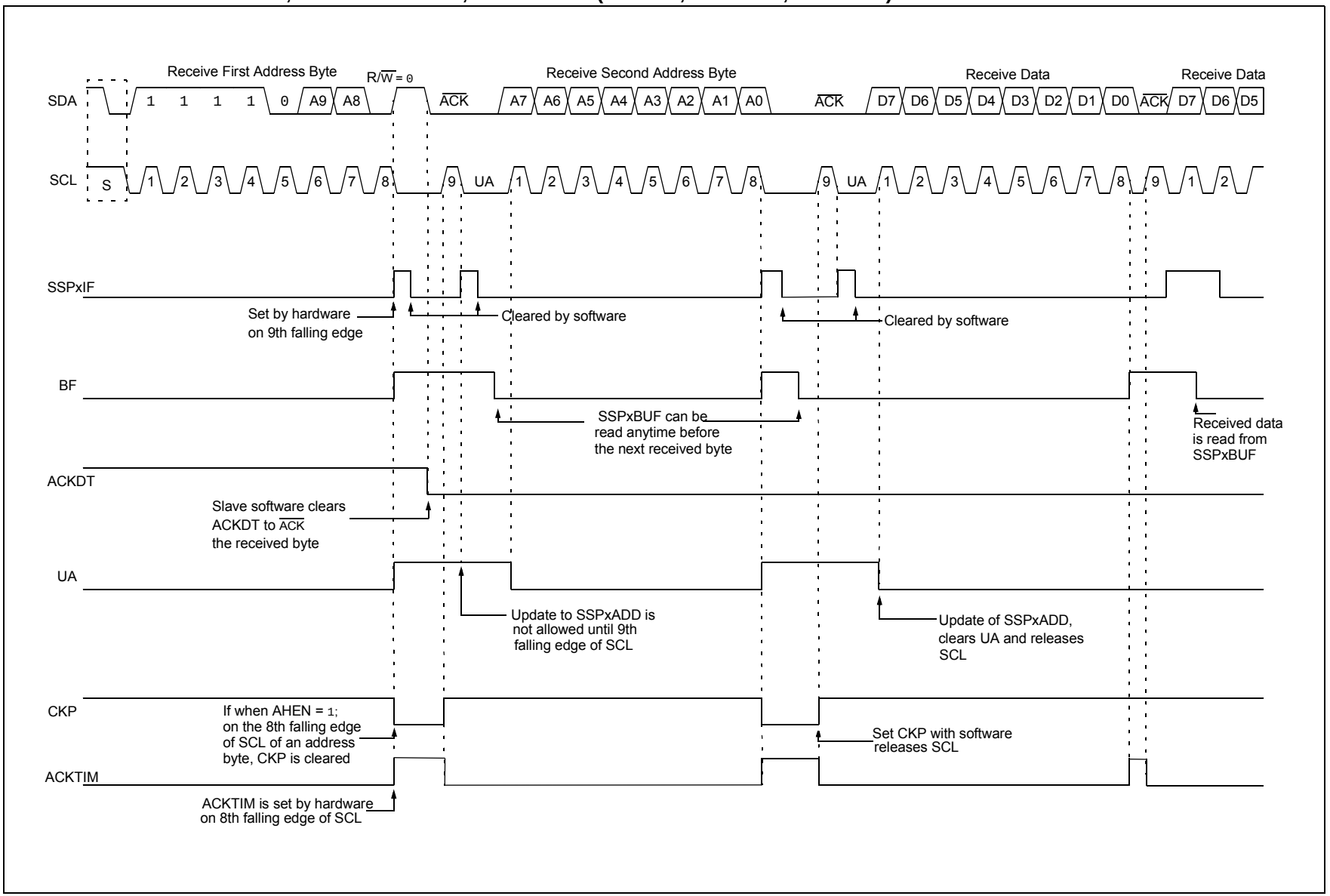
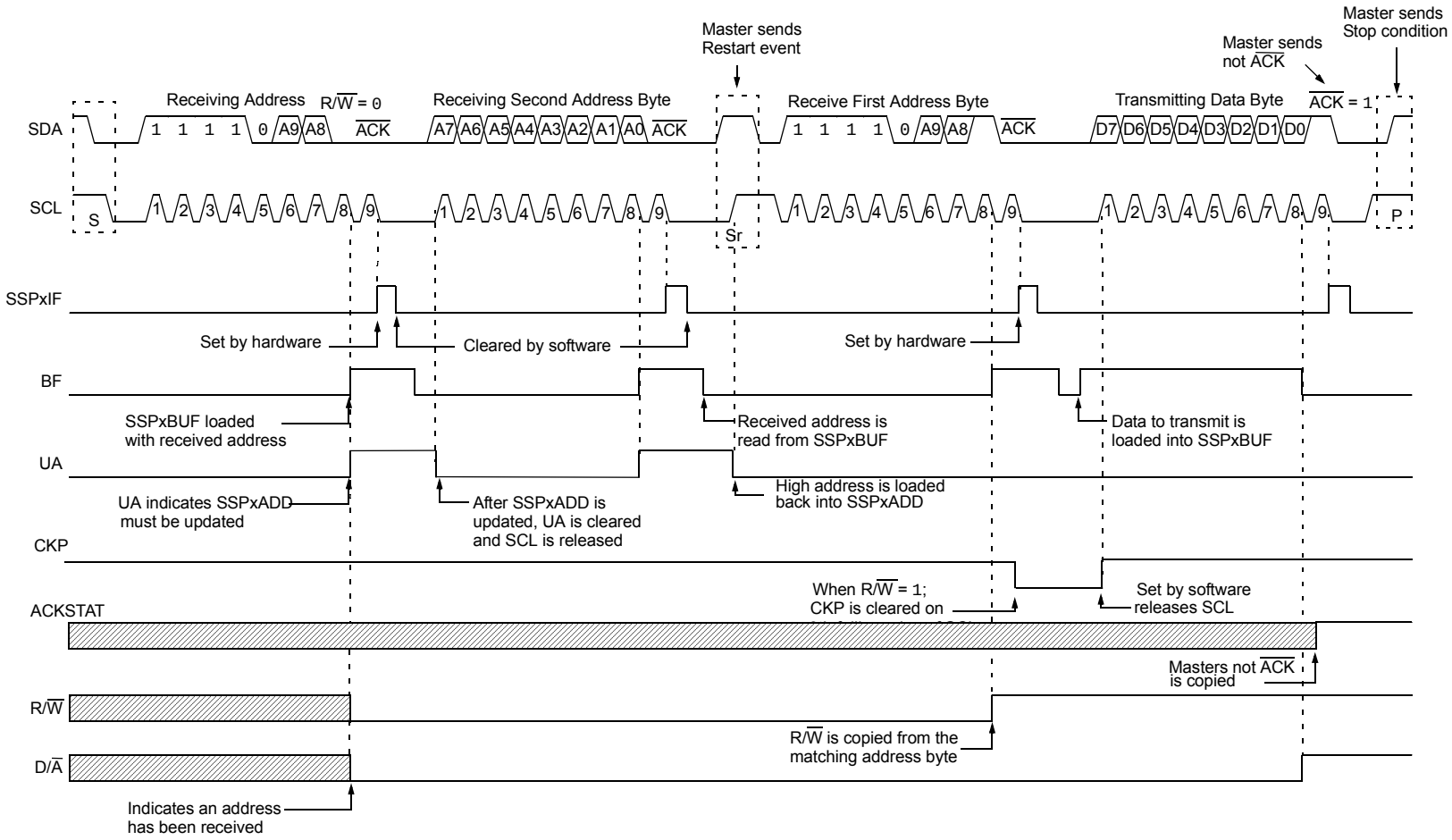


**FIGURE 32-22: I²C SLAVE, 10-BIT ADDRESS, TRANSMISSION (SEN = 0, AHEN = 0, DHEN = 0)**



### 32.5.6 CLOCK STRETCHING

Clock stretching occurs when a device on the bus holds the SCL 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 handled by the hardware that generates SCL.

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 SCL line to go low and then hold it. Setting CKP will release SCL and allow more communication.

#### 32.5.6.1 Normal Clock Stretching

Following an $\overline{ACK}$ if the R/$\overline{W}$ bit of SSPxSTAT is set, a read request, 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 on if the clock will be stretched or not. This is different than previous versions of the module that would not stretch the clock, clear CKP, if SSPxBUF was read before the ninth falling edge of SCL.

**2:** Previous versions of the module did not stretch the clock for a transmission if SSPxBUF was loaded before the ninth falling edge of SCL. It is now always cleared for read requests.

#### 32.5.6.2 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 SCL is stretched without CKP being cleared. SCL 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.

#### 32.5.6.3 Byte NACKing

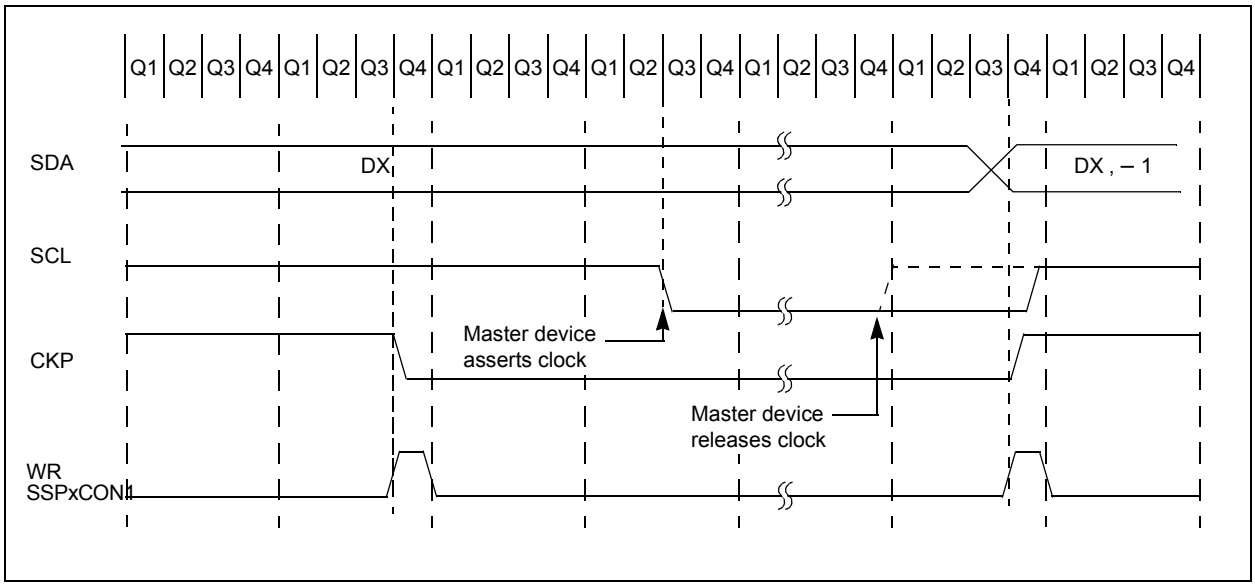
When AHEN bit of SSPxCON3 is set; CKP is cleared by hardware after the eighth falling edge of SCL for a received matching address byte. When DHEN bit of SSPxCON3 is set; CKP is cleared after the eighth falling edge of SCL for received data.

Stretching after the eighth falling edge of SCL allows the slave to look at the received address or data and decide if it wants to ACK the received data.

#### 32.5.6.4 Clock Synchronization and the CKP Bit

Any time the CKP bit is cleared, the module will wait for the SCL line to go low and then hold it. However, clearing the CKP bit will not assert the SCL output low until the SCL output is already sampled low. Therefore, the CKP bit will not assert the SCL line until an external $I^2C$ master device has already asserted the SCL line. The SCL output will remain low until the CKP bit is set and all other devices on the $I^2C$ bus have released SCL. This ensures that a write to the CKP bit will not violate the minimum high time requirement for SCL (see Figure 32-23).

**FIGURE 32-23: CLOCK SYNCHRONIZATION TIMING**



### 32.5.7 GENERAL CALL ADDRESS SUPPORT

The addressing procedure for the $I^2C$ 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 $I^2C$ protocol, defined as address 0x00. When the GCEN bit of the SSPxCON2 register is set, the slave module will automatically $\overline{ACK}$ 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/$\overline{W}$ bit clear, an interrupt is generated and slave software can read SSPxBUF and respond. Figure 32-24 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 SCL. The slave must then set its ACKDT value and release the clock with communication progressing as it would normally.

### 32.5.8 SSP MASK REGISTER

An SSP Mask (SSPxMSK) register (Register 32-5) is available in $I^2C$ Slave mode as a mask for the value held in the SSPSR 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”.

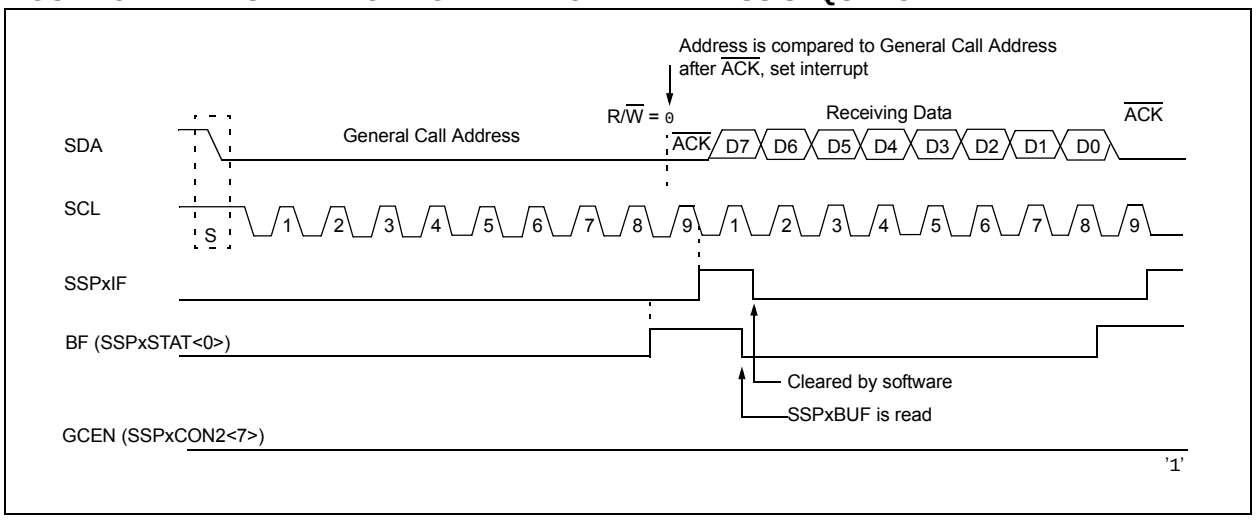
This register is reset to all ‘1’s upon any Reset condition and, therefore, has no effect on standard SSP operation until written with a mask value.

The SSP 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 SSP mask has no effect during the reception of the first (high) byte of the address.

**FIGURE 32-24: SLAVE MODE GENERAL CALL ADDRESS SEQUENCE**



## 32.6 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 SDA and SCK 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 MSSP 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 SDA and SCL lines.

The following events will cause the SSP Interrupt Flag bit, SSPxIF, to be set (SSP interrupt, if enabled):

- Start condition detected
- Stop condition detected
- Data transfer byte transmitted/received
- Acknowledge transmitted/received
- Repeated Start generated

**Note 1:** The MSSP 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:** When in Master mode, Start/Stop detection is masked and an interrupt is generated when the SEN/PEN bit is cleared and the generation is complete.

### 32.6.1 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 SDA, while SCL outputs the serial clock. The first byte transmitted contains the slave address of the receiving device (7 bits) and the Read/Write (R/$\overline{W}$) bit. In this case, the R/$\overline{W}$ 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.

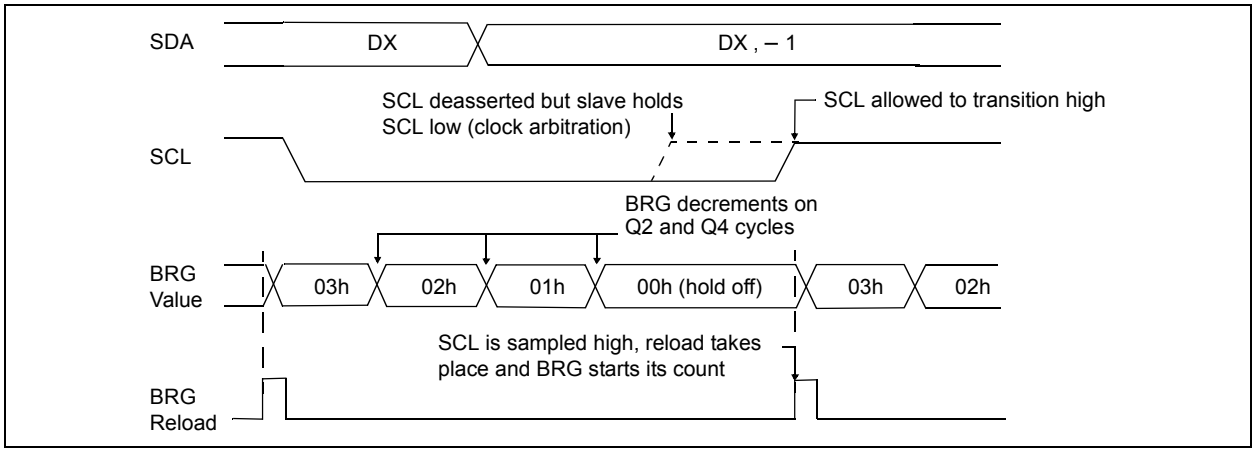
In Master Receive mode, the first byte transmitted contains the slave address of the transmitting device (7 bits) and the R/$\overline{W}$ bit. In this case, the R/$\overline{W}$ bit will be logic '1'. Thus, the first byte transmitted is a 7-bit slave address followed by a '1' to indicate the receive bit. Serial data is received via SDA, while SCL 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 SCL. See **Section 32.7 "Baud Rate Generator"** for more detail.

### 32.6.2 CLOCK ARBITRATION

Clock arbitration occurs when the master, during any receive, transmit or Repeated Start/Stop condition, releases the SCL pin (SCL allowed to float high). When the SCL pin is allowed to float high, the Baud Rate Generator (BRG) is suspended from counting until the SCL pin is actually sampled high. When the SCL pin is sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<7:0> and begins counting. This ensures that the SCL 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 32-25).

**FIGURE 32-25: BAUD RATE GENERATOR TIMING WITH CLOCK ARBITRATION**




### 32.6.3 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. |

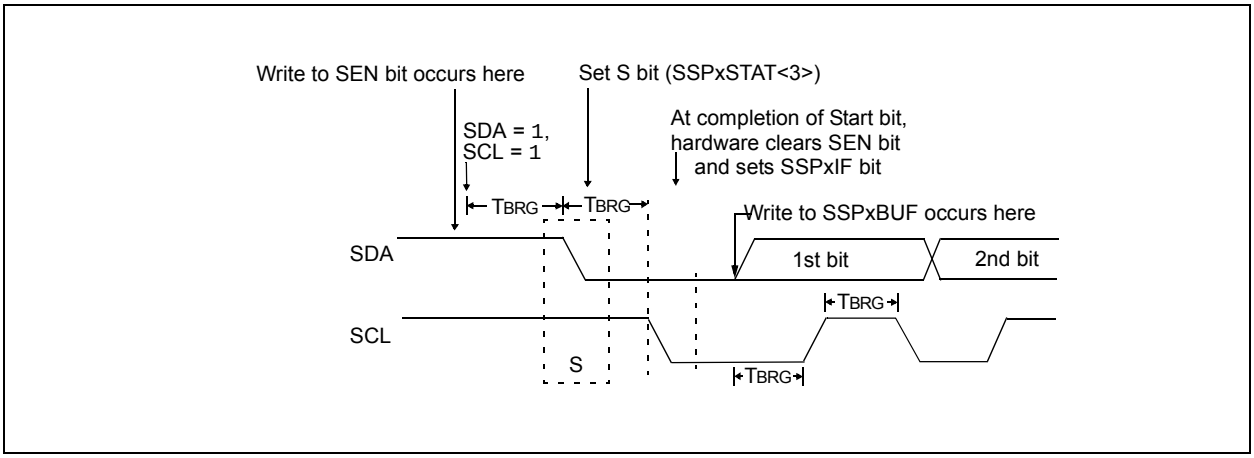
### 32.6.4 I²C MASTER MODE START CONDITION TIMING

To initiate a Start condition (Figure 32-26), the user sets the Start Enable bit, SEN bit of the SSPxCON2 register. If the SDA and SCL pins are sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<7:0> and starts its count. If SCL and SDA are both sampled high when the Baud Rate Generator times out (TBRG), the SDA pin is driven low. The action of the SDA being driven low while SCL 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 SDA line held low and the Start condition is complete.

**Note 1:** If at the beginning of the Start condition, the SDA and SCL pins are already sampled low, or if during the Start condition, the SCL line is sampled low before the SDA line is driven low, a bus collision occurs, the Bus Collision Interrupt Flag, BCLIF, 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 32-26: FIRST START BIT TIMING**



### 32.6.5 I²C MASTER MODE REPEATED START CONDITION TIMING

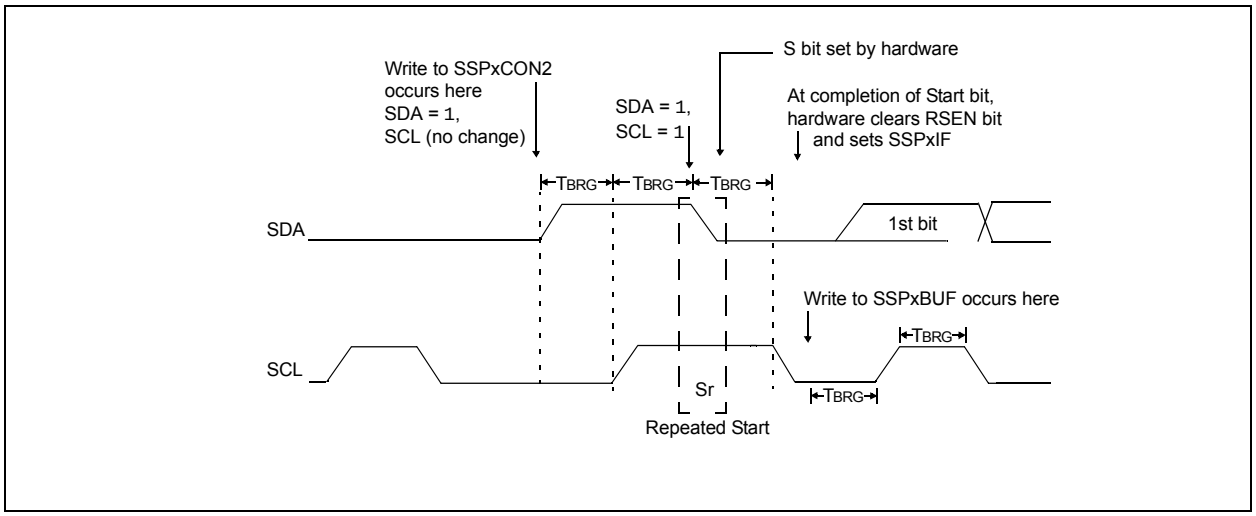
A Repeated Start condition (Figure 32-27) occurs when the RSEN bit of the SSPxCON2 register is programmed high and the master state machine is no longer active. When the RSEN bit is set, the SCL pin is asserted low. When the SCL pin is sampled low, the Baud Rate Generator is loaded and begins counting. The SDA pin is released (brought high) for one Baud Rate Generator count (TBRG). When the Baud Rate Generator times out, if SDA is sampled high, the SCL pin will be deasserted (brought high). When SCL is sampled high, the Baud Rate Generator is reloaded and begins counting. SDA and SCL must be sampled high for one TBRG. This action is then followed by assertion of the SDA pin (SDA = 0) for one TBRG while SCL is high. SCL 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 SDA pin held low. As soon as a Start condition is detected on the SDA and SCL 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:

- SDA is sampled low when SCL goes from low-to-high.
- SCL goes low before SDA is asserted low. This may indicate that another master is attempting to transmit a data '1'.

**FIGURE 32-27: REPEATED START CONDITION WAVEFORM**



### 32.6.6 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 SDA pin after the falling edge of SCL is asserted. SCL is held low for one Baud Rate Generator rollover count (TBRG). Data should be valid before SCL is released high. When the SCL pin is released high, it is held that way for TBRG. The data on the SDA pin must remain stable for that duration and some hold time after the next falling edge of SCL. After the eighth bit is shifted out (the falling edge of the eighth clock), the BF flag is cleared and the master releases SDA. 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 SCL low and SDA unchanged (Figure 32-28).

After the write to the SSPxBUF, each bit of the address will be shifted out on the falling edge of SCL 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 SDA pin, allowing the slave to respond with an Acknowledge. On the falling edge of the ninth clock, the master will sample the SDA 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 SCL low and allowing SDA to float.

#### 32.6.6.1 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.

#### 32.6.6.2 WCOL Status Flag

If the user writes the SSPxBUF when a transmit is already in progress (i.e., SSPSR 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.

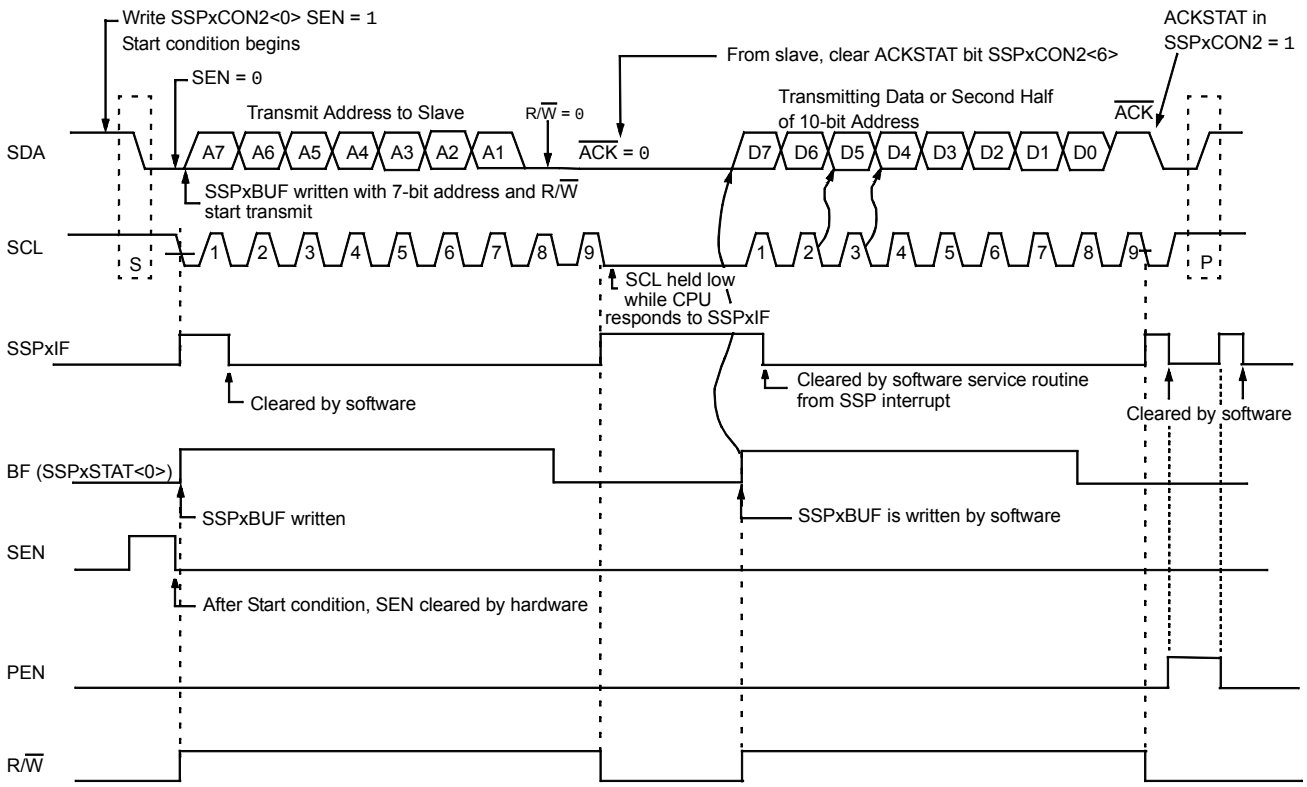
#### 32.6.6.3 ACKSTAT Status Flag

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.

#### 32.6.6.4 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 MSSP 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 SDA pin until all eight bits are transmitted. Transmission begins as soon as SSPxBUF is written to.
7. The MSSP 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 MSSP 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 SDA pin until all eight bits are transmitted.
11. The MSSP 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.

**FIGURE 32-28: $I^2C$ MASTER MODE WAVEFORM (TRANSMISSION, 7 OR 10-BIT ADDRESS)**



### 32.6.7 I²C MASTER MODE RECEPTION

Master mode reception (Figure 32-29) is enabled by programming the Receive Enable bit, RCEN bit of the SSPxCON2 register.

| **Note:** | The MSSP 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 SCL pin changes (high-to-low/low-to-high) and data is shifted into the SSPSR. After the falling edge of the eighth clock, the receive enable flag is automatically cleared, the contents of the SSPSR 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 SCL low. The MSSP 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.

#### 32.6.7.1 BF Status Flag

In receive operation, the BF bit is set when an address or data byte is loaded into SSPxBUF from SSPSR. It is cleared when the SSPxBUF register is read.

#### 32.6.7.2 SSPOV Status Flag

In receive operation, the SSPOV bit is set when eight bits are received into the SSPSR and the BF flag bit is already set from a previous reception.

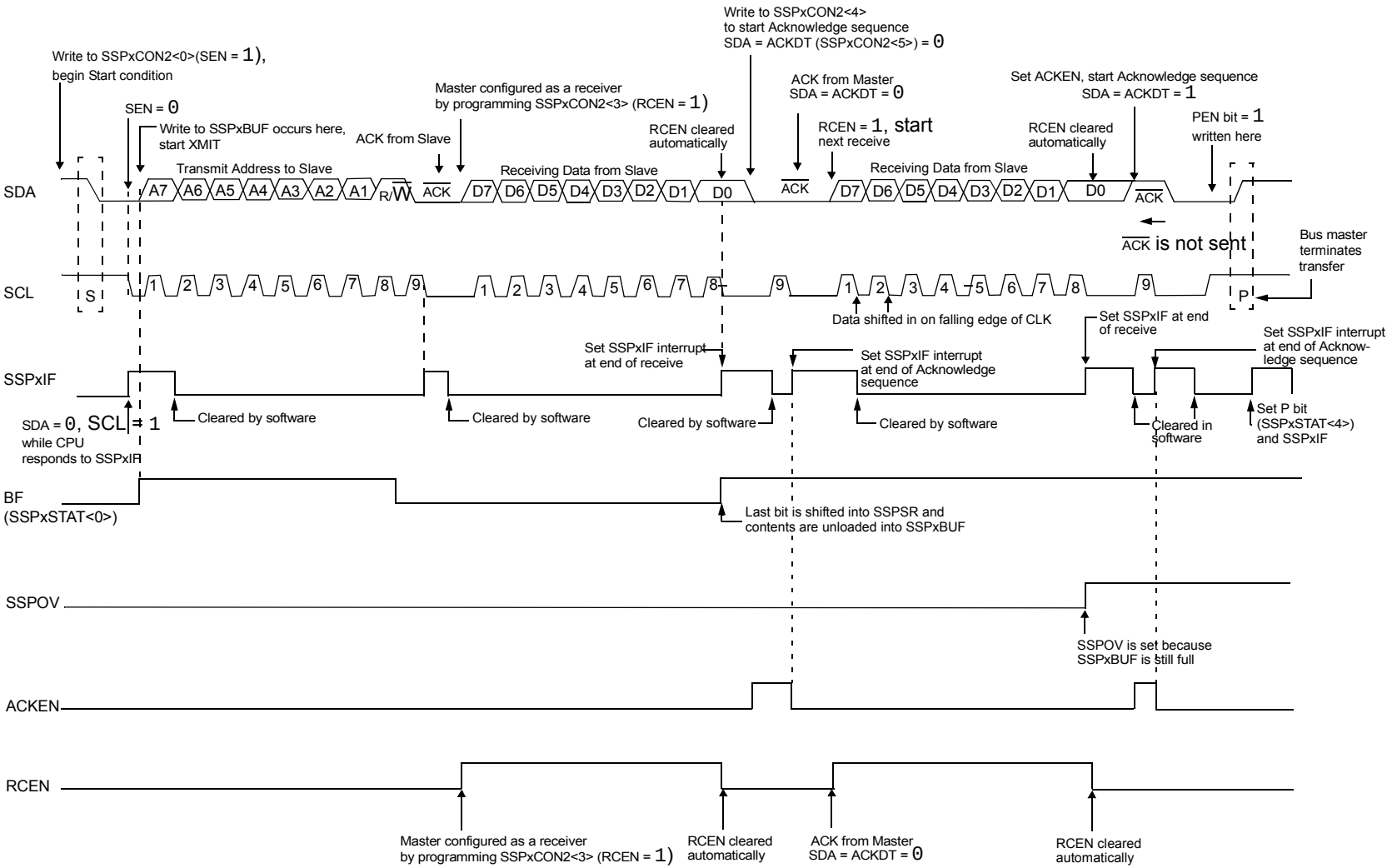
#### 32.6.7.3 WCOL Status Flag

If the user writes the SSPxBUF when a receive is already in progress (i.e., SSPSR 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).

#### 32.6.7.4 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 SDA pin until all eight bits are transmitted. Transmission begins as soon as SSPxBUF is written to.
6. The MSSP 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 MSSP 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 SCL, SSPxIF and BF are set.
10. Master clears SSPxIF and reads the received byte from SSPxBUF, 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. Master's $\overline{ACK}$ 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.

**FIGURE 32-29: $I^2C$ MASTER MODE WAVEFORM (RECEPTION, 7-BIT ADDRESS)**



### 32.6.8 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 SCL pin is pulled low and the contents of the Acknowledge data bit are presented on the SDA 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 SCL pin is deasserted (pulled high). When the SCL pin is sampled high (clock arbitration), the Baud Rate Generator counts for TBRG. The SCL pin is then pulled low. Following this, the ACKEN bit is automatically cleared, the Baud Rate Generator is turned off and the MSSP module then goes into Idle mode (Figure 32-30).

#### 32.6.8.1 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).

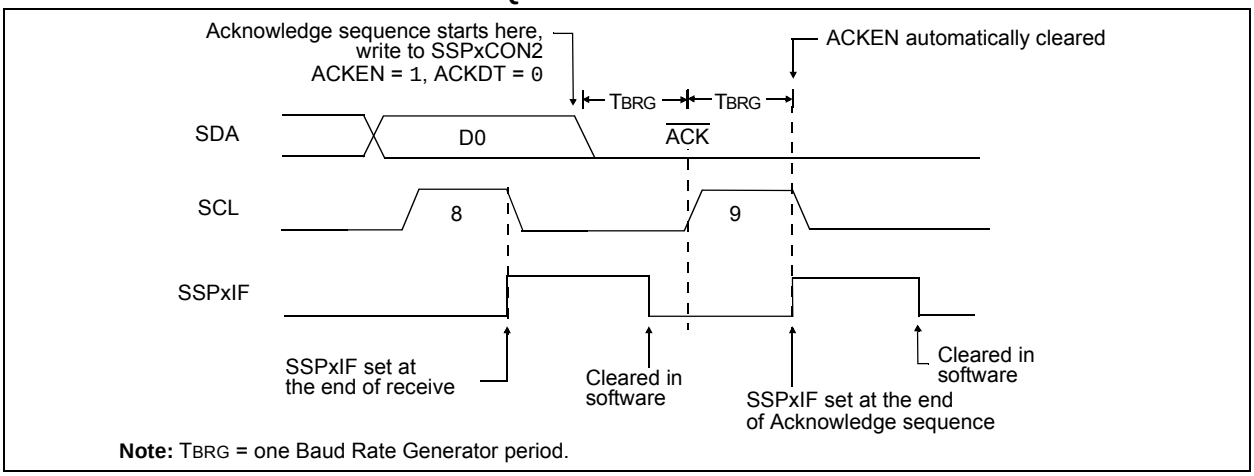
### 32.6.9 STOP CONDITION TIMING

A Stop bit is asserted on the SDA 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 SCL line is held low after the falling edge of the ninth clock. When the PEN bit is set, the master will assert the SDA line low. When the SDA line is sampled low, the Baud Rate Generator is reloaded and counts down to '0'. When the Baud Rate Generator times out, the SCL pin will be brought high and one TBRG (Baud Rate Generator rollover count) later, the SDA pin will be deasserted. When the SDA pin is sampled high while SCL 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 32-31).

#### 32.6.9.1 WCOL Status Flag

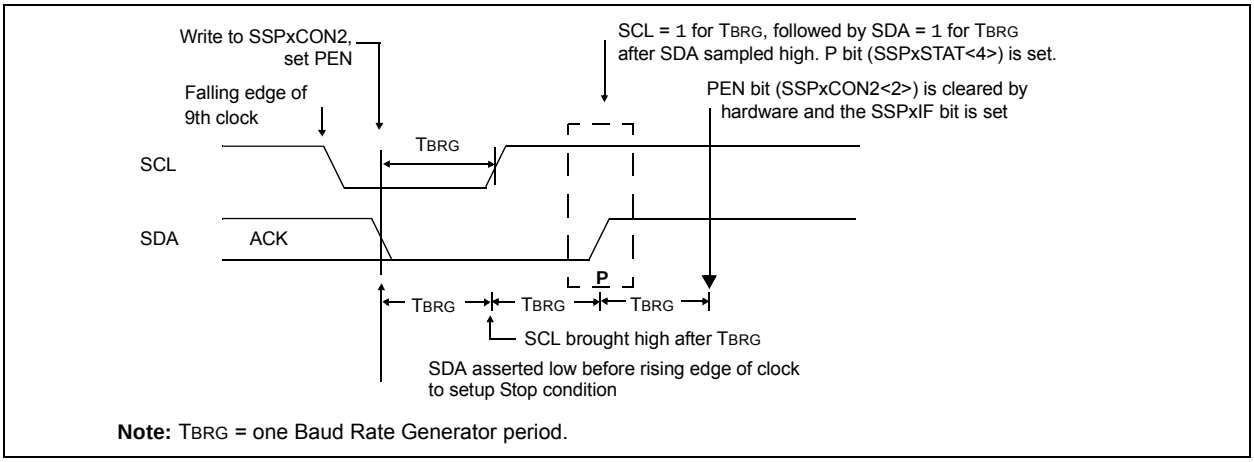
If the user writes the SSPxBUF when a Stop sequence is in progress, then the WCOL bit is set and the contents of the buffer are unchanged (the write does not occur).

**FIGURE 32-30: ACKNOWLEDGE SEQUENCE WAVEFORM**




**Note:** TBRG = one Baud Rate Generator period.

**FIGURE 32-31: STOP CONDITION RECEIVE OR TRANSMIT MODE**




**Note:** TBRG = one Baud Rate Generator period.

### 32.6.10 SLEEP OPERATION

While in Sleep mode, the $I^2C$ slave module can receive addresses or data and when an address match or complete byte transfer occurs, wake the processor from Sleep (if the MSSP interrupt is enabled).

### 32.6.11 EFFECTS OF A RESET

A Reset disables the MSSP module and terminates the current transfer.

### 32.6.12 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 MSSP module is disabled. Control of the $I^2C$ 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 SSP interrupt will generate the interrupt when the Stop condition occurs.

In multi-master operation, the SDA 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 BCLIF bit.

The states where arbitration can be lost are:

- Address Transfer
- Data Transfer
- A Start Condition
- A Repeated Start Condition
- An Acknowledge Condition

### 32.6.13 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 SDA pin, arbitration takes place when the master outputs a '1' on SDA, by letting SDA float high and another master asserts a '0'. When the SCL pin floats high, data should be stable. If the expected data on SDA is a '1' and the data sampled on the SDA pin is '0', then a bus collision has taken place. The master will set the Bus Collision Interrupt Flag, BCLIF and reset the $I^2C$ port to its Idle state (Figure 32-32).

If a transmit was in progress when the bus collision occurred, the transmission is halted, the BF flag is cleared, the SDA and SCL lines are deasserted and the SSPxBUF can be written to. When the user services the bus collision Interrupt Service Routine and if the $I^2C$ bus is free, the user can resume communication by asserting a Start condition.

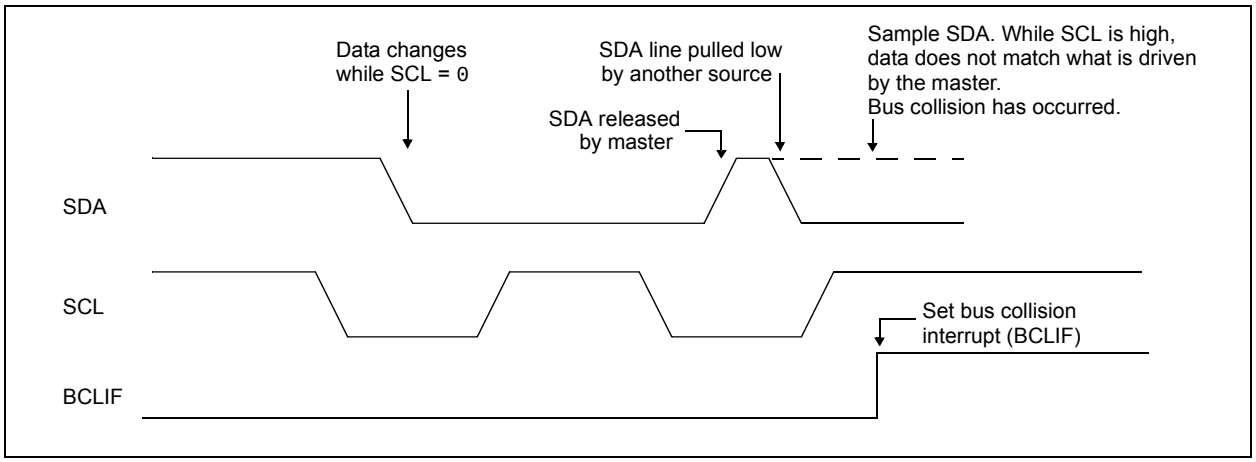
If a Start, Repeated Start, Stop or Acknowledge condition was in progress when the bus collision occurred, the condition is aborted, the SDA and SCL lines are deasserted and the respective control bits in the SSPxCON2 register are cleared. When the user services the bus collision Interrupt Service Routine and if the $I^2C$ bus is free, the user can resume communication by asserting a Start condition.

The master will continue to monitor the SDA and SCL 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 $I^2C$ 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 32-32: BUS COLLISION TIMING FOR TRANSMIT AND ACKNOWLEDGE**



#### 32.6.13.1 Bus Collision During a Start Condition

During a Start condition, a bus collision occurs if:

a) SDA or SCL are sampled low at the beginning of the Start condition (Figure 32-33).

b) SCL is sampled low before SDA is asserted low (Figure 32-34).

During a Start condition, both the SDA and the SCL pins are monitored.

If the SDA pin is already low, or the SCL pin is already low, then all of the following occur:

- the Start condition is aborted,
- the BCLIF flag is set and
- the MSSP module is reset to its Idle state (Figure 32-33).

The Start condition begins with the SDA and SCL pins deasserted. When the SDA pin is sampled high, the Baud Rate Generator is loaded and counts down. If the SCL pin is sampled low while SDA 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 SDA pin is sampled low during this count, the BRG is reset and the SDA line is asserted early (Figure 32-35). If, however, a '1' is sampled on the SDA pin, the SDA 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 SCL pin is sampled as '0' during this time, a bus collision does not occur. At the end of the BRG count, the SCL 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 SDA 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 32-33: BUS COLLISION DURING START CONDITION (SDA ONLY)**

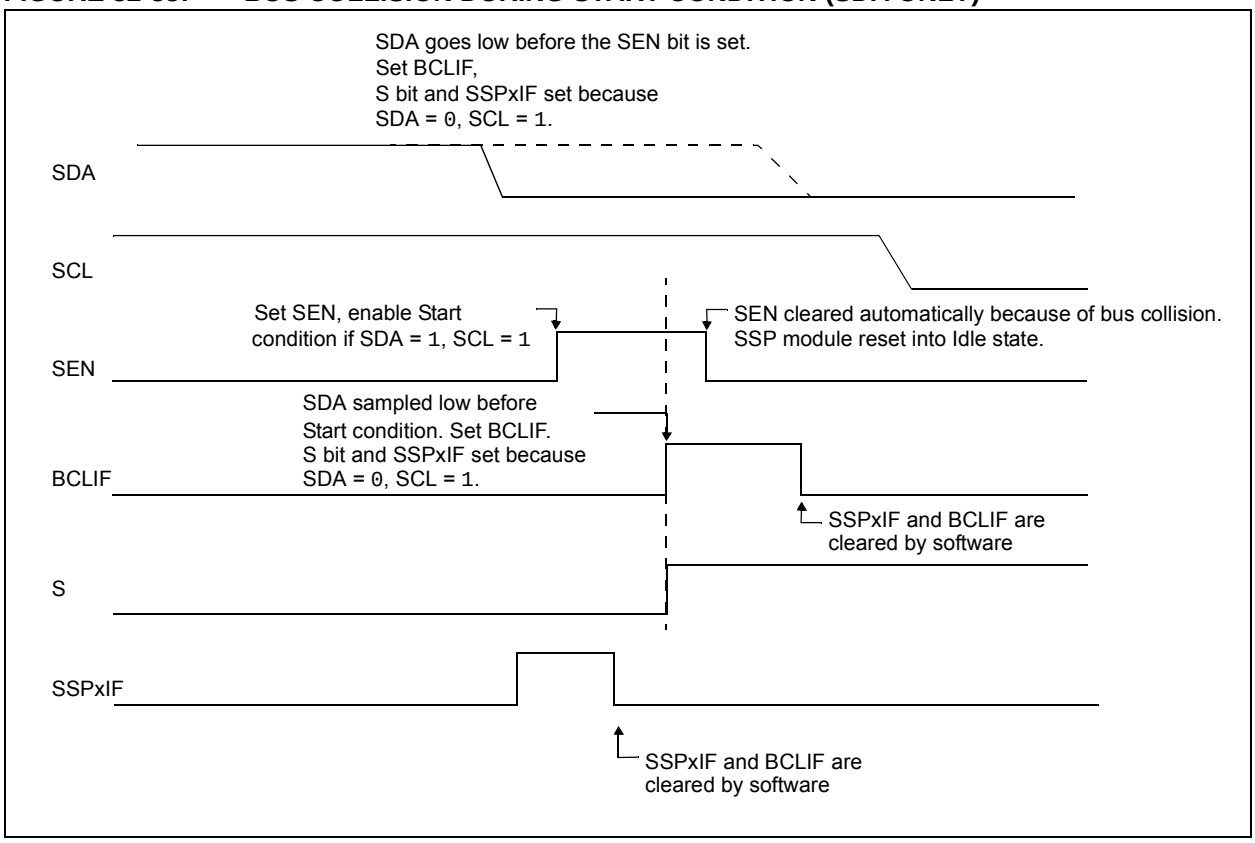


**FIGURE 32-34: BUS COLLISION DURING START CONDITION (SCL = 0)**

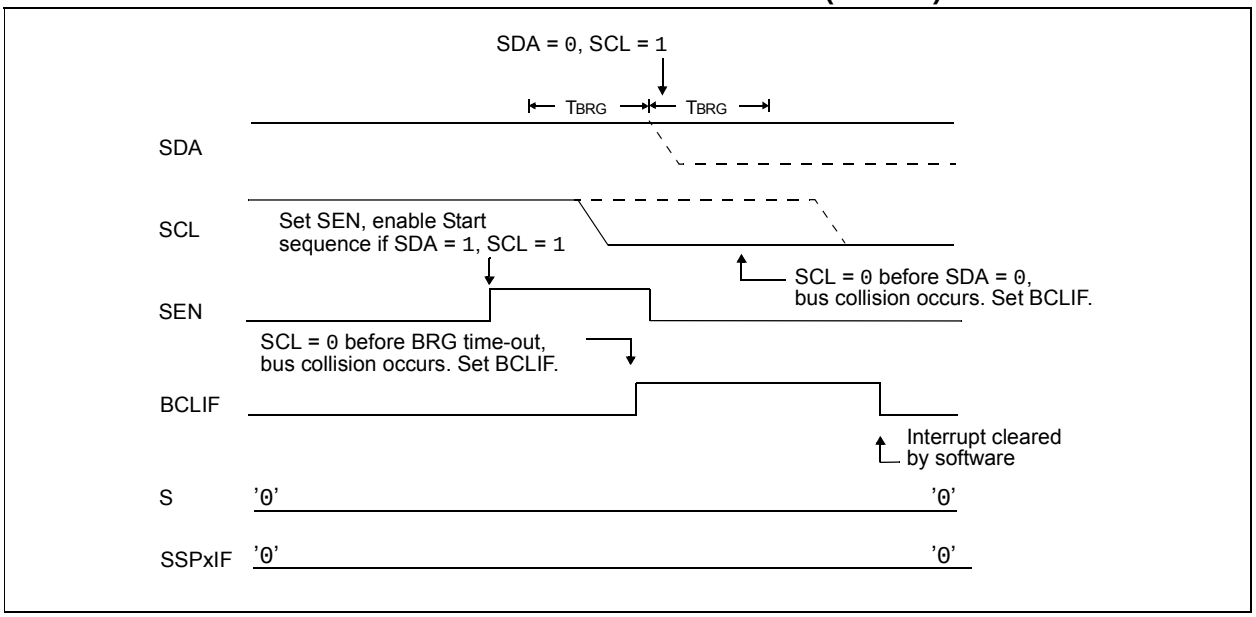
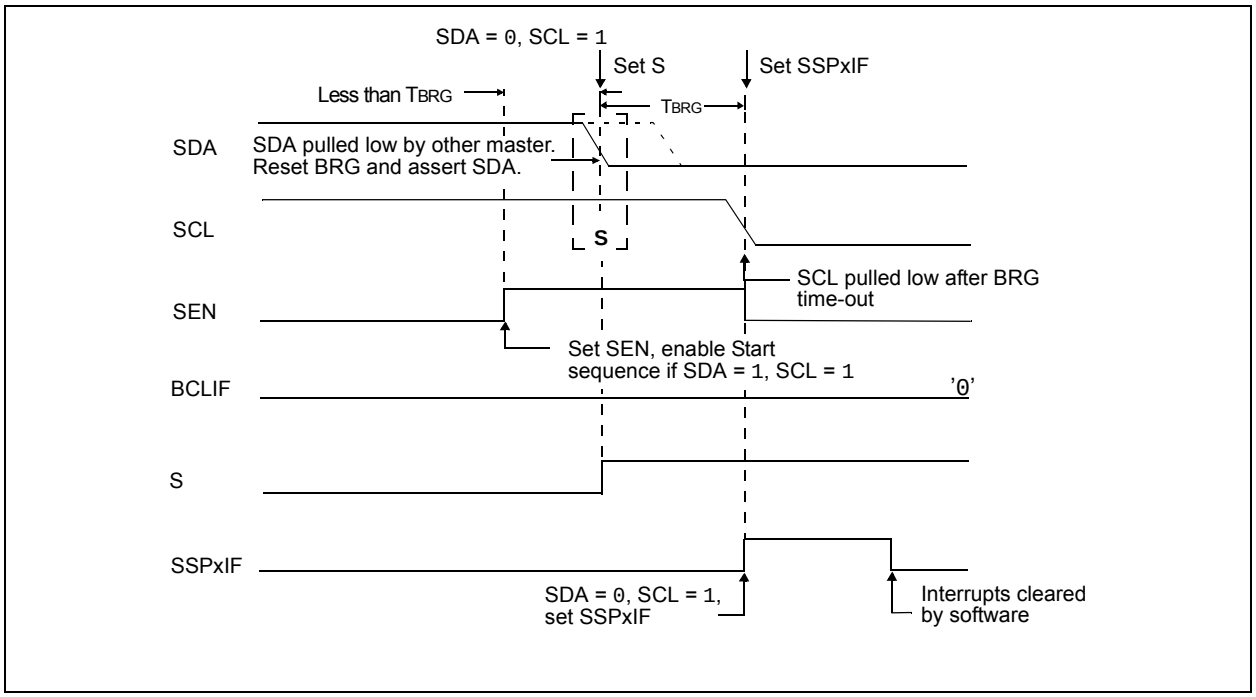



**FIGURE 32-35: BRG RESET DUE TO SDA ARBITRATION DURING START CONDITION**



### 32.6.13.2 Bus Collision During a Repeated Start Condition

During a Repeated Start condition, a bus collision occurs if:

a) A low level is sampled on SDA when SCL goes from low level to high level (Case 1).

b) SCL goes low before SDA is asserted low, indicating that another master is attempting to transmit a data '1' (Case 2).

When the user releases SDA and the pin is allowed to float high, the BRG is loaded with SSPxADD and counts down to zero. The SCL pin is then deasserted and when sampled high, the SDA pin is sampled.

If SDA is low, a bus collision has occurred (i.e., another master is attempting to transmit a data '0', Figure 32-36). If SDA is sampled high, the BRG is reloaded and begins counting. If SDA goes from high-to-low before the BRG times out, no bus collision occurs because no two masters can assert SDA at exactly the same time.

If SCL goes from high-to-low before the BRG times out and SDA 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 32-37.

If, at the end of the BRG time-out, both SCL and SDA are still high, the SDA pin is driven low and the BRG is reloaded and begins counting. At the end of the count, regardless of the status of the SCL pin, the SCL pin is driven low and the Repeated Start condition is complete.

**FIGURE 32-36: BUS COLLISION DURING A REPEATED START CONDITION (CASE 1)**

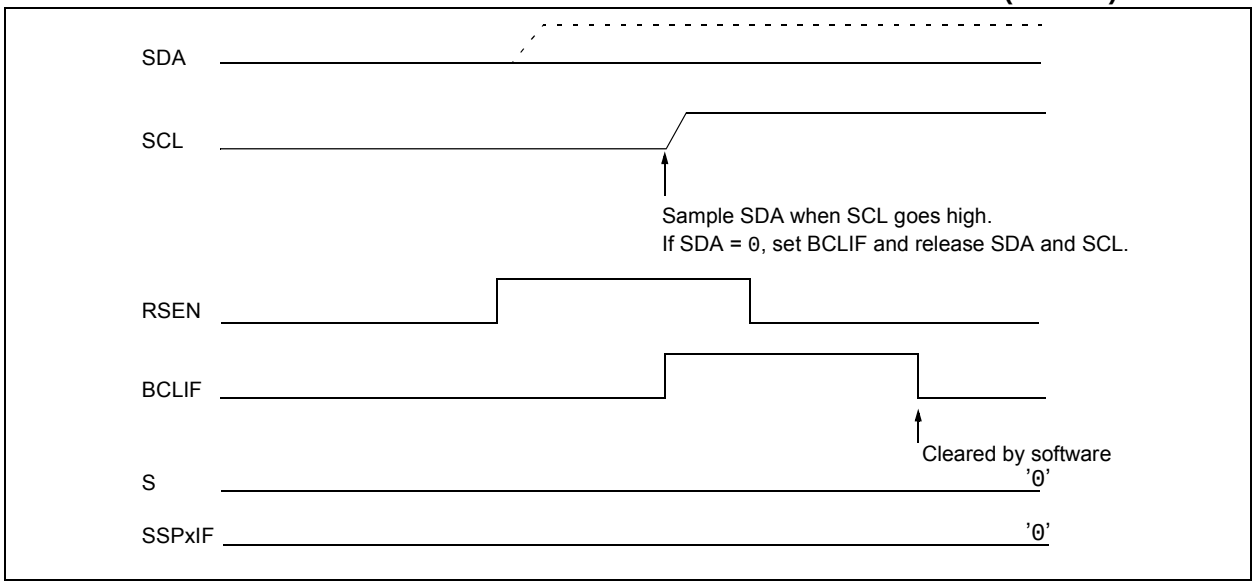
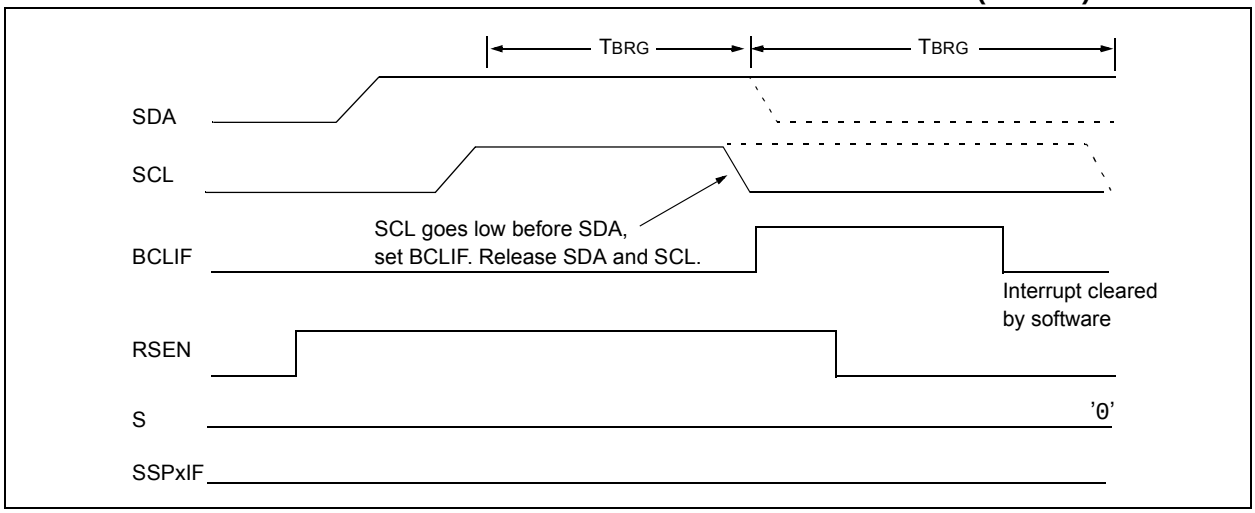



**FIGURE 32-37: BUS COLLISION DURING REPEATED START CONDITION (CASE 2)**



#### 32.6.13.3 Bus Collision During a Stop Condition

Bus collision occurs during a Stop condition if:

a) After the SDA pin has been deasserted and allowed to float high, SDA is sampled low after the BRG has timed out (Case 1).

b) After the SCL pin is deasserted, SCL is sampled low before SDA goes high (Case 2).

The Stop condition begins with SDA asserted low. When SDA is sampled low, the SCL 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, SDA is sampled. If SDA is sampled low, a bus collision has occurred. This is due to another master attempting to drive a data '0' (Figure 32-38). If the SCL pin is sampled low before SDA is allowed to float high, a bus collision occurs. This is another case of another master attempting to drive a data '0' (Figure 32-39).

**FIGURE 32-38: BUS COLLISION DURING A STOP CONDITION (CASE 1)**

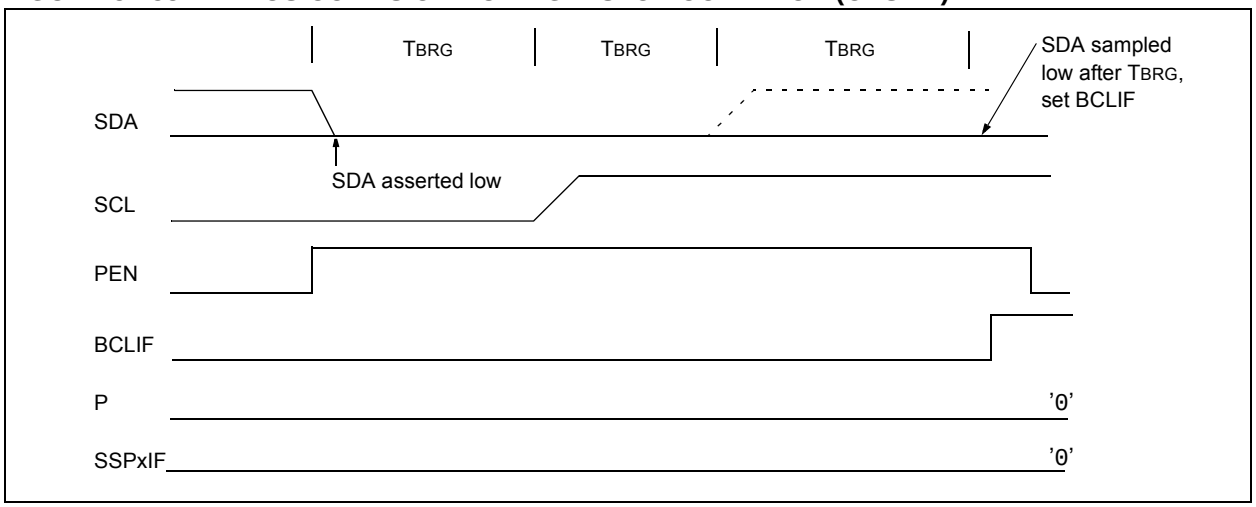



**FIGURE 32-39: BUS COLLISION DURING A STOP CONDITION (CASE 2)**

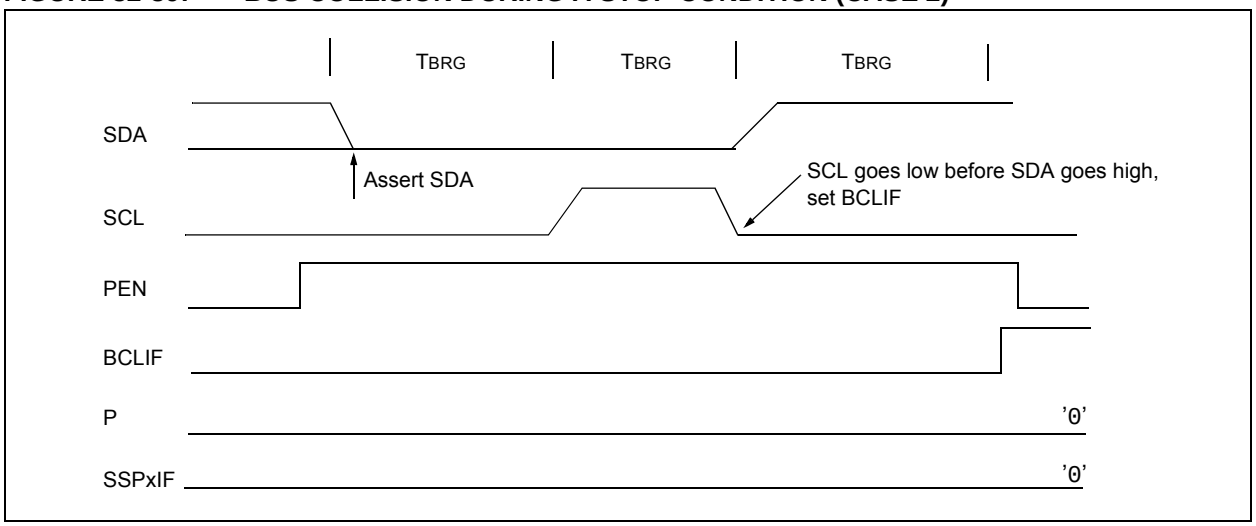


**TABLE 32-3: SUMMARY OF REGISTERS ASSOCIATED WITH I²C OPERATION**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Reset Values on Page: |
|---|---|---|---|---|---|---|---|---|---|
| ANSELA | — | — | ANSA5 | ANSA4 | ANSA3 | ANSA2 | ANSA1 | ANSA0 | 153 |
| ANSELB | — | — | ANSB5 | ANSB4 | ANSB3 | ANSB2 | ANSB1 | ANSB0 | 158 |
| ANSELC | ANSC7 | ANSC6 | ANSC5 | ANSC4 | ANSC3 | ANSC2 | — | — | 163 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 108 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 109 |
| PIE2 | OSFIE | C2IE | C1IE | COG1IE | BCL1IE | C4IE | C3IE | CCP2IE | 110 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 115 |
| PIR2 | OSFIF | C2IF | C1IF | COG1IF | BCL1IF | C4IF | C3IF | CCP2IF | 116 |
| RxyPPS | — | — | RxyPPS<5:0> | | | | | | 172 |
| SSPCLKPPS | — | — | SSPCLKPPS<5:0> | | | | | | 172, 174 |
| SSPDATPPS | — | — | SSPDATPPS<5:0> | | | | | | 172, 174 |
| SSPSSPPS | — | — | SSPSSPPS<5:0> | | | | | | 172, 174 |
| SSP1ADD | ADD<7:0> | | | | | | | | 453 |
| SSP1BUF | Synchronous Serial Port Receive Buffer/Transmit Register | | | | | | | | 405* |
| SSP1CON1 | WCOL | SSPOV | SSPEN | CKP | SSPM<3:0> | | | | 450 |
| SSP1CON2 | GCEN | ACKSTAT | ACKDT | ACKEN | RCEN | PEN | RSEN | SEN | 451 |
| SSP1CON3 | ACKTIM | PCIE | SCIE | BOEN | SDAHT | SBCDE | AHEN | DHEN | 452 |
| SSP1MSK | MSK<7:0> | | | | | | | | 453 |
| SSP1STAT | SMP | CKE | D/$\overline{A}$ | P | S | R/$\overline{W}$ | UA | BF | 449 |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 152 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 157 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 162 |

**Legend:** — = unimplemented location, read as '0'. Shaded cells are not used by the MSSP module in I²C mode.

* Page provides register information.

## 32.7 BAUD RATE GENERATOR

The MSSP 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 32-6). 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.

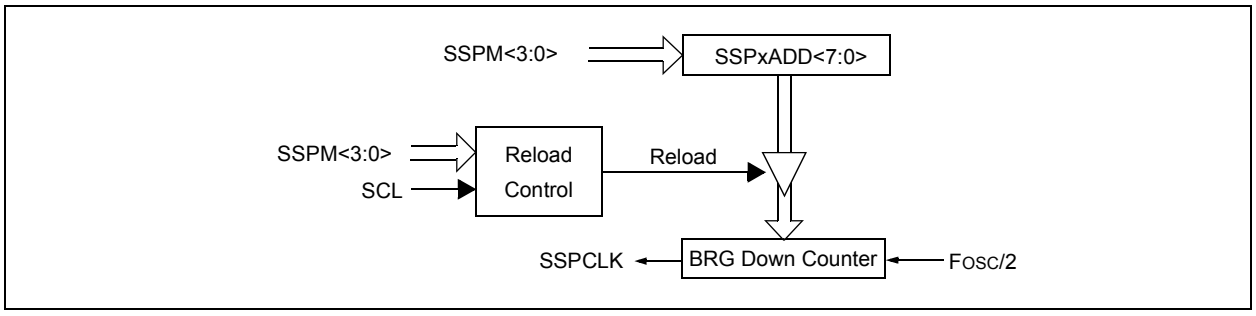
An internal signal “Reload” in Figure 32-40 triggers the value from SSPxADD to be loaded into the BRG counter. This occurs twice for each oscillation of the module clock line. The logic dictating when the reload signal is asserted depends on the mode the MSSP is being operated in.

Table 32-4 demonstrates clock rates based on instruction cycles and the BRG value loaded into SSPxADD.

**EQUATION 32-1:**

$$F_{CLOCK} = \frac{F_{OSC}}{(SSPxADD + 1)(4)}$$

**FIGURE 32-40: 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 32-4: MSSP CLOCK RATE W/BRG**

| FOSC | FCY | BRG Value | FCLOCK (2 Rollovers of BRG) |
|---|---|---|---|
| 32 MHz | 8 MHz | 13h | 400 kHz |
| 32 MHz | 8 MHz | 19h | 308 kHz |
| 32 MHz | 8 MHz | 4Fh | 100 kHz |
| 16 MHz | 4 MHz | 09h | 400 kHz |
| 16 MHz | 4 MHz | 0Ch | 308 kHz |
| 16 MHz | 4 MHz | 27h | 100 kHz |
| 4 MHz | 1 MHz | 09h | 100 kHz |

**Note:** Refer to the I/O port electrical specifications in Table 36-4 to ensure the system is designed to support IOL requirements.

## 32.8 Register Definitions: MSSP Control

### REGISTER 32-1: SSP1STAT: SSP 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 MSSP 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 MSSP 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 $\overline{ACK}$ 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 MSSP 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 SSP1ADD 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, SSP1BUF is full
0 = Receive not complete, SSP1BUF is empty
Transmit ($I^2C$ mode only):
1 = Data transmit in progress (does not include the $\overline{ACK}$ and Stop bits), SSP1BUF is full
0 = Data transmit complete (does not include the $\overline{ACK}$ and Stop bits), SSP1BUF is empty

## REGISTER 32-2: SSP1CON1: SSP 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(1) | 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 SSP1BUF register was attempted while the I²C conditions were not valid for a transmission to be started
0 = No collision
Slave mode:
1 = The SSP1BUF 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 SSP1BUF register is still holding the previous data. In case of overflow, the data in SSPSR is lost. Overflow can only occur in Slave mode. In Slave mode, the user must read the SSP1BUF, 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 SSP1BUF register (must be cleared in software).
0 = No overflow
In I²C mode:
1 = A byte is received while the SSP1BUF 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 SCK, SDO, SDI and $\overline{SS}$ as the source of the serial port pins(2)
0 = Disables serial port and configures these pins as I/O port pins
In I²C mode:
1 = Enables the serial port and configures the SDA and SCL 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 I²C Slave mode:
SCL release control
1 = Enable clock
0 = Holds clock low (clock stretch). (Used to ensure data setup time.)
In I²C Master mode:
Unused in this mode

bit 3-0 **SSPM<3:0>:** Synchronous Serial Port Mode Select bits
1111 = I²C Slave mode, 10-bit address with Start and Stop bit interrupts enabled
1110 = I²C Slave mode, 7-bit address with Start and Stop bit interrupts enabled
1101 = Reserved
1100 = Reserved
1011 = I²C firmware controlled Master mode (slave idle)
1010 = SPI Master mode, clock = FOSC/(4 * (SSP1ADD+1))(5)
1001 = Reserved
1000 = I²C Master mode, clock = FOSC / (4 * (SSP1ADD+1))(4)
0111 = I²C Slave mode, 10-bit address
0110 = I²C Slave mode, 7-bit address
0101 = SPI Slave mode, clock = SCK pin, $\overline{SS}$ pin control disabled, $\overline{SS}$ can be used as I/O pin
0100 = SPI Slave mode, clock = SCK pin, $\overline{SS}$ pin control enabled
0011 = SPI Master mode, clock = T2_match/2
0010 = SPI Master mode, clock = FOSC/64
0001 = SPI Master mode, clock = FOSC/16
0000 = SPI Master mode, clock = FOSC/4

**Note 1:** In Master mode, the overflow bit is not set since each new reception (and transmission) is initiated by writing to the SSP1BUF register.
**2:** When enabled, these pins must be properly configured as input or output. Use SSPSSPPS, SSPCLKPPS, SSPDATPPS, and RxyPPS to select the pins.
**3:** When enabled, the SDA and SCL pins must be configured as inputs. Use SSPCLKPPS, SSPDATPPS, and RxyPPS to select the pins.
**4:** SSP1ADD values of 0, 1 or 2 are not supported for I²C mode.
**5:** SSP1ADD value of '0' is not supported. Use SSPM = 0000 instead.

**REGISTER 32-3: SSP1CON2: SSP CONTROL REGISTER 2(1)**

| 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 SSPSR
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)
In Master Receive mode:
1 = Initiate Acknowledge sequence on SDA and SCL 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 = Enables Receive mode for $I^2C$
0 = Receive idle

bit 2 **PEN:** Stop Condition Enable bit (in $I^2C$ Master mode only)
SCKMSSP Release Control:
1 = Initiate Stop condition on SDA and SCL pins. Automatically cleared by hardware.
0 = Stop condition Idle

bit 1 **RSEN:** Repeated Start Condition Enable bit (in $I^2C$ Master mode only)
1 = Initiate Repeated Start condition on SDA and SCL pins. Automatically cleared by hardware.
0 = Repeated Start condition Idle

bit 0 **SEN:** Start Condition Enable/Stretch Enable bit
In Master mode:
1 = Initiate Start condition on SDA and SCL 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 the Idle mode, this bit may not be set (no spooling) and the SSP1BUF may not be written (or writes to the SSP1BUF are disabled).

**REGISTER 32-4: SSP1CON3: SSP 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(3) | 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 ($I^2C$ slave mode only)(3)
1 = Indicates the $I^2C$ bus is in an Acknowledge sequence, set on eighth falling edge of SCL clock
0 = Not an Acknowledge sequence, cleared on ninth rising edge of SCL 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 = SSP1BUF 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 SSP1STAT register already set, SSPOV bit of the SSP1CON1 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 = SSP1BUF 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 = SSP1BUF is only updated when SSPOV is clear

bit 3 **SDAHT:** SDA Hold Time Selection bit ($I^2C$ mode only)
1 = Minimum of 300 ns hold time on SDA after the falling edge of SCL
0 = Minimum of 100 ns hold time on SDA after the falling edge of SCL

bit 2 **SBCDE:** Slave Mode Bus Collision Detect Enable bit ($I^2C$ Slave mode only)
If, on the rising edge of SCL, SDA is sampled low when the module is outputting a high state, the BCL1IF 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 eighth falling edge of SCL for a matching received address byte; CKP bit of the SSP1CON1 register will be cleared and the SCL will be held low.
0 = Address holding is disabled

bit 0 **DHEN:** Data Hold Enable bit ($I^2C$ Slave mode only)
1 = Following the eighth falling edge of SCL for a received data byte; slave hardware clears the CKP bit of the SSP1CON1 register and SCL 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 SSP1BUF.
**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.

**REGISTER 32-5: SSP1MSK: SSP 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 SSP1ADD<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 SSP1ADD<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 32-6: SSP1ADD: MSSP 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
SCL 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".

## 33.0 ENHANCED UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (EUSART)

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:

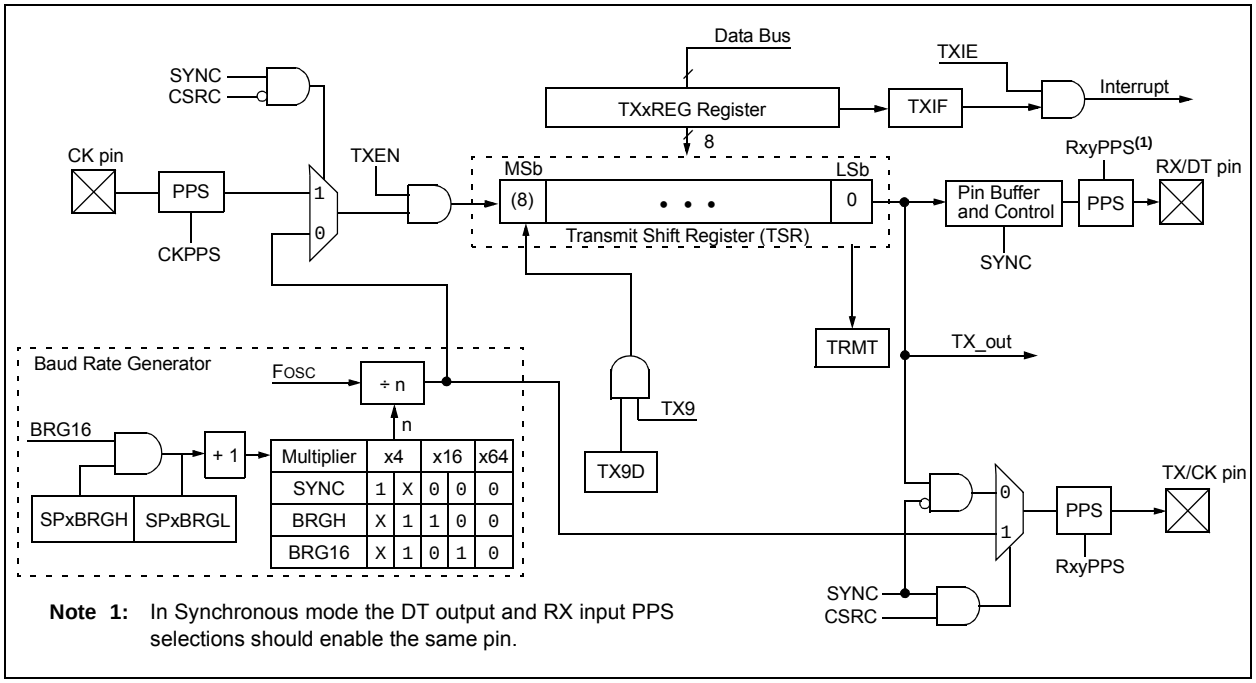
- Automatic detection and calibration of the baud rate
- Wake-up on Break reception
- 13-bit Break character transmit

Block diagrams of the EUSART transmitter and receiver are shown in Figure 33-1 and Figure 33-2.

The EUSART transmit output (TX_out) is available to the TX/CK pin and internally to the following peripherals:

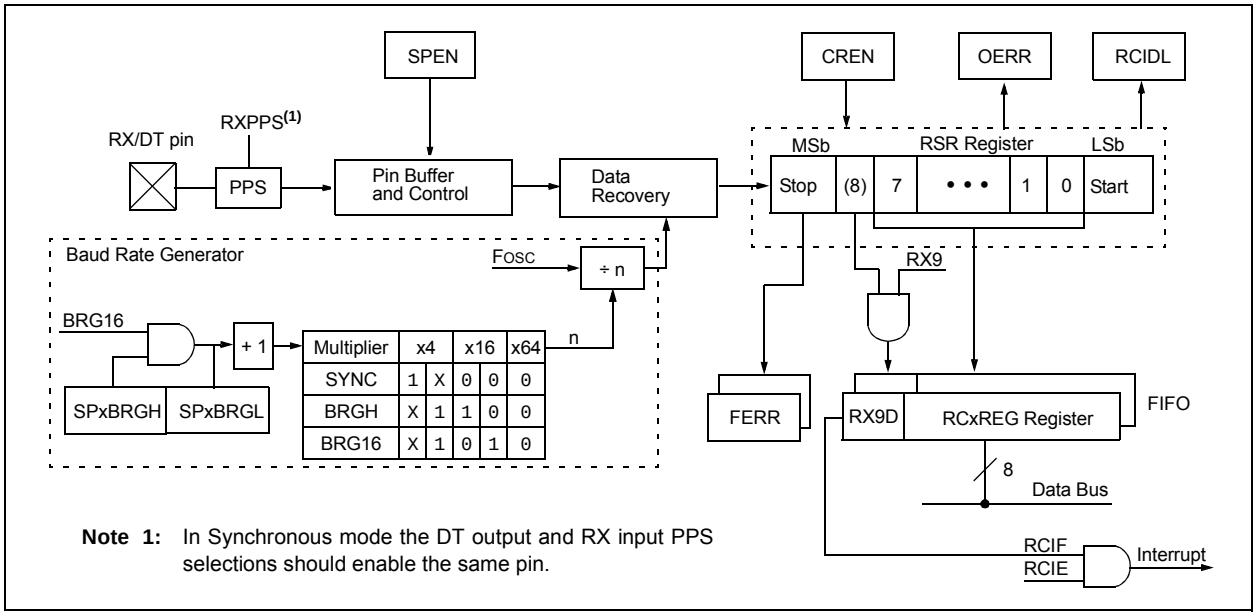
- Configurable Logic Cell (CLC)
- Data signal modulator (DSM)

**FIGURE 33-1: EUSART TRANSMIT BLOCK DIAGRAM**




**Note 1:** In Synchronous mode the DT output and RX input PPS selections should enable the same pin.

**FIGURE 33-2: EUSART RECEIVE BLOCK DIAGRAM**




**Note 1:** In Synchronous mode the DT output and RX input PPS selections should enable the same pin.

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 33-1, Register 33-2 and Register 33-3, respectively.

The RX and CK input pins are selected with the RXPPS and CKPPS registers, respectively. TX, CK, and DT output pins are selected with each pin's RxyPPS register. Since the RX input is coupled with the DT output in Synchronous mode, it is the user's responsibility to select the same pin for both of these functions when operating in Synchronous mode. The EUSART control logic will control the data direction drivers automatically.

## 33.1 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 33-5 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.

### 33.1.1 EUSART ASYNCHRONOUS TRANSMITTER

The EUSART transmitter block diagram is shown in Figure 33-1. 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.

#### 33.1.1.1 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 and automatically configures the TX/CK I/O pin as an output. If the TX/CK pin is shared with an analog peripheral, the analog I/O function must be disabled by clearing the corresponding ANSEL bit.

| **Note:** | The TXIF Transmitter Interrupt flag is set when the TXEN enable bit is set. |
|---|---|

#### 33.1.1.2 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.

#### 33.1.1.3 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 in Asynchronous mode only. In Synchronous mode, the SCKP bit has a different function. See **Section 33.5.1.2 "Clock Polarity"**.

#### 33.1.1.4 Transmit Interrupt Flag

The TXIF interrupt flag bit of the PIR1 register is set whenever the EUSART transmitter is enabled and no character is being held for transmission in the TXxREG. In other words, the TXIF 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 TXIF flag bit is not cleared immediately upon writing TXxREG. TXIF becomes valid in the second instruction cycle following the write execution. Polling TXIF immediately following the TXxREG write will return invalid results. The TXIF bit is read-only, it cannot be set or cleared by software.

The TXIF interrupt can be enabled by setting the TXIE interrupt enable bit of the PIE1 register. However, the TXIF flag bit will be set whenever the TXxREG is empty, regardless of the state of the TXIE enable bit.

To use interrupts when transmitting data, set the TXIE bit only when there is more data to send. Clear the TXIE interrupt enable bit upon writing the last character of the transmission to the TXxREG.

#### 33.1.1.5 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 has 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. |
|---|---|

#### 33.1.1.6 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 33.1.2.7 "Address Detection"** for more information on the Address mode.

#### 33.1.1.7 Asynchronous Transmission Set-up:

1. Initialize the SPxBRGH:SPxBRGL register pair and the BRGH and BRG16 bits to achieve the desired baud rate (see **Section 33.4 "EUSART Baud Rate Generator (BRG)"**).
2. Enable the asynchronous serial port by clearing the SYNC bit and setting the SPEN bit.
3. 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.
4. Set the SCKP bit if inverted transmit is desired.
5. Enable the transmission by setting the TXEN control bit. This will cause the TXIF interrupt bit to be set.
6. If interrupts are desired, set the TXIE interrupt enable bit of the PIE1 register. An interrupt will occur immediately provided that the GIE and PEIE bits of the INTCON register are also set.
7. If 9-bit transmission is selected, the ninth bit should be loaded into the TX9D data bit.
8. Load 8-bit data into the TXxREG register. This will start the transmission.

**FIGURE 33-3: ASYNCHRONOUS TRANSMISSION**

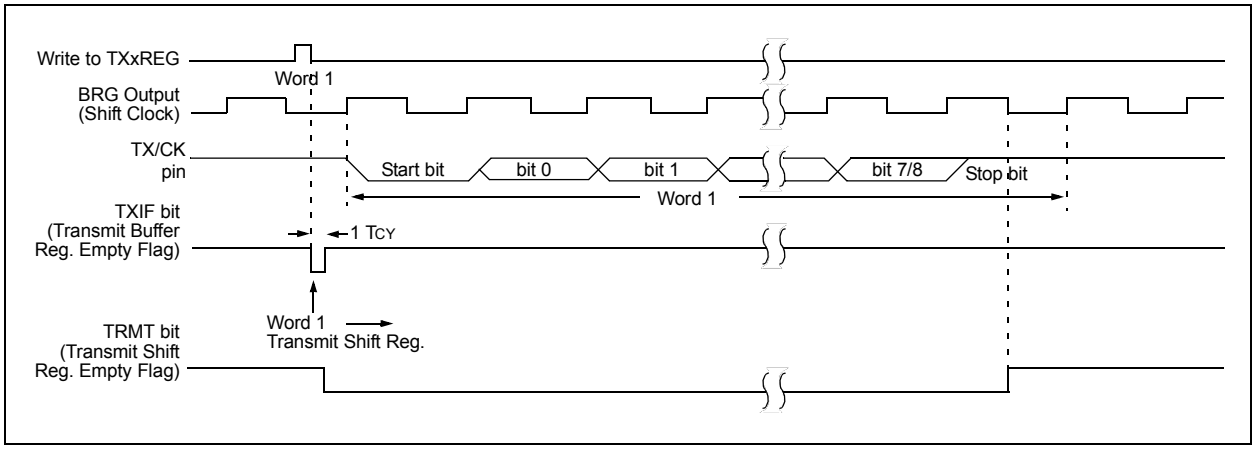
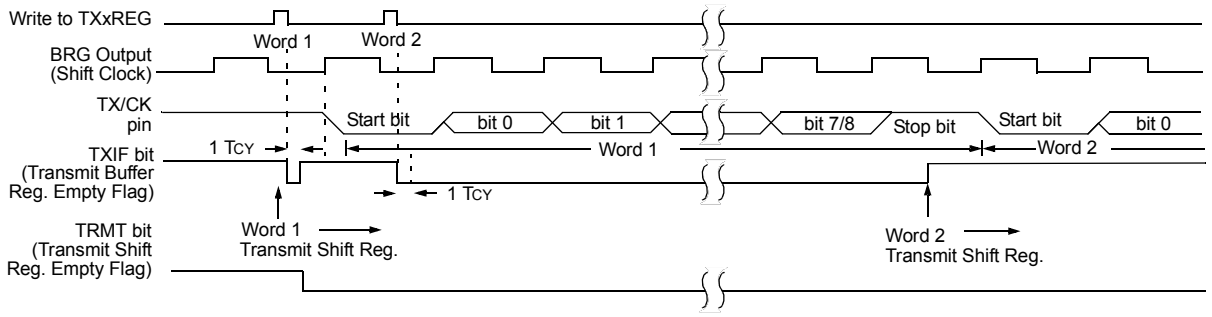


**FIGURE 33-4: ASYNCHRONOUS TRANSMISSION (BACK-TO-BACK)**




**Note:** This timing diagram shows two consecutive transmissions.

**TABLE 33-1: SUMMARY OF 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 |
|---|---|---|---|---|---|---|---|---|---|
| ANSELA | — | — | ANSA5 | ANSA4 | ANSA3 | ANSA2 | ANSA1 | ANSA0 | 153 |
| ANSELB | — | — | ANSB5 | ANSB4 | ANSB3 | ANSB2 | ANSB1 | ANSB0 | 158 |
| ANSELC | ANSC7 | ANSC6 | ANSC5 | ANSC4 | ANSC3 | ANSC2 | — | — | 163 |
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 466 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 108 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 109 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 115 |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 465 |
| RxyPPS | — | — | RxyPPS<5:0> | | | | | | 172, 174 |
| SP1BRGL | SP1BRG<7:0> | | | | | | | | 467* |
| SP1BRGH | SP1BRG<15:8> | | | | | | | | 467* |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 152 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 157 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 162 |
| TX1REG | EUSART Transmit Data Register | | | | | | | | 456* |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 464 |

**Legend:** — = unimplemented location, read as '0'. Shaded cells are not used for asynchronous transmission.

* Page provides register information.

### 33.1.2 EUSART ASYNCHRONOUS RECEIVER

The Asynchronous mode is typically used in RS-232 systems. The receiver block diagram is shown in Figure 33-2. The data is received on the RX/DT 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 via the RCxREG register.

#### 33.1.2.1 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 RX/DT I/O pin as an input.

| **Note:** | If the RX/DT function is on an analog pin, the corresponding ANSEL bit must be cleared for the receiver to function. |
|---|---|

#### 33.1.2.2 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 33.1.2.4 "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 RCIF interrupt flag bit of the PIR1 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 33.1.2.5 "Receive Overrun Error"** for more information on overrun errors. |
|---|---|

#### 33.1.2.3 Receive Interrupts

The RCIF interrupt flag bit of the PIR1 register is set whenever the EUSART receiver is enabled and there is an unread character in the receive FIFO. The RCIF interrupt flag bit is read-only, it cannot be set or cleared by software.

RCIF interrupts are enabled by setting all of the following bits:

- RCIE, Interrupt Enable bit of the PIE1 register
- PEIE, Peripheral Interrupt Enable bit of the INTCON register
- GIE, Global Interrupt Enable bit of the INTCON register

The RCIF interrupt flag bit will be set when there is an unread character in the FIFO, regardless of the state of interrupt enable bits.

#### 33.1.2.4 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. |

#### 33.1.2.5 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.

#### 33.1.2.6 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.

#### 33.1.2.7 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 RCIF 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.

#### 33.1.2.8 Asynchronous Reception Set-up

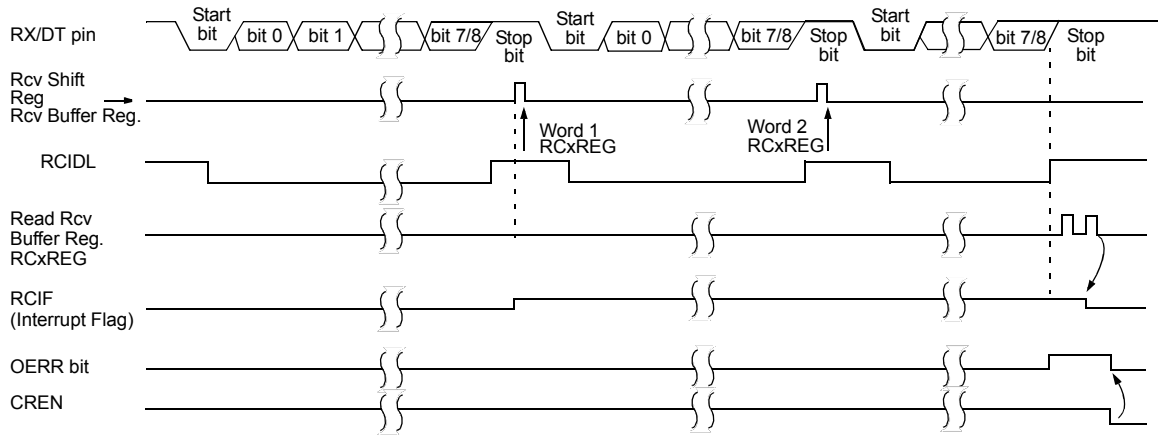
1. Initialize the SPxBRGH:SPxBRGL register pair and the BRGH and BRG16 bits to achieve the desired baud rate (see **Section 33.4 "EUSART Baud Rate Generator (BRG)"**).
2. Clear the ANSEL bit for the RX pin (if applicable).
3. Enable the serial port by setting the SPEN bit. The SYNC bit must be clear for asynchronous operation.
4. If interrupts are desired, set the RCIE bit of the PIE1 register and 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 RCIF 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 RCIE 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.

#### 33.1.2.9 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 33.4 "EUSART Baud Rate Generator (BRG)"**).
2. Clear the ANSEL bit for the RX pin (if applicable).
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 RCIE bit of the PIE1 register and 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 RCIF 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 RCIE 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 33-5: ASYNCHRONOUS RECEPTION**




**Note:** This timing diagram shows three words appearing on the RX input. The RCxREG (receive buffer) is read after the third word, causing the OERR (overrun) bit to be set.

**TABLE 33-2: SUMMARY OF 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 |
|---|---|---|---|---|---|---|---|---|---|
| ANSELA | — | — | ANSA5 | ANSA4 | ANSA3 | ANSA2 | ANSA1 | ANSA0 | 153 |
| ANSELB | — | — | ANSB5 | ANSB4 | ANSB3 | ANSB2 | ANSB1 | ANSB0 | 158 |
| ANSELC | ANSC7 | ANSC6 | ANSC5 | ANSC4 | ANSC3 | ANSC2 | — | — | 163 |
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 466 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 108 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 109 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 115 |
| RC1REG | EUSART Receive Data Register | | | | | | | | 459* |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 465 |
| RxyPPS | — | — | RxyPPS<5:0> | | | | | | 172 |
| SP1BRGL | SP1BRG<7:0> | | | | | | | | 467 |
| SP1BRGH | SP1BRG<15:8> | | | | | | | | 467 |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 152 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 157 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 162 |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 464 |

**Legend:** — = unimplemented location, read as '0'. Shaded cells are not used for asynchronous reception.

* Page provides register information.

## 33.2 Clock Accuracy with Asynchronous Operation

The factory calibrates the internal oscillator block output (INTOSC). However, the INTOSC 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 INTOSC output. Adjusting the value in the OSCTUNE register allows for fine resolution changes to the system clock source. See **Section 5.2.2.3 "Internal Oscillator Frequency Adjustment"** 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 33.4.1 "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.

## 33.3 Register Definitions: EUSART Control

**REGISTER 33-1: TX1STA: TRANSMIT STATUS AND CONTROL REGISTER**

| R/W-/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R-1/1 | R/W-0/0 |
|---|---|---|---|---|---|---|---|
| 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' |
| 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 **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 33-2: RC1STA: RECEIVE STATUS AND CONTROL REGISTER**

| R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R/W-0/0 | R-0/0 | R-0/0 | R-0/0 |
|---|---|---|---|---|---|---|---|
| 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' |
| 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 **SPEN:** Serial Port Enable bit
1 = Serial port enabled
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.

**REGISTER 33-3: BAUD1CON: 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 TX/CK pin
0 = Transmit non-inverted data to the TX/CK 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

## 33.4 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.

Table 33-3 contains the formulas for determining the baud rate. Example 33-1 provides a sample calculation for determining the 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 33-5. 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 33-1: CALCULATING BAUD RATE ERROR

For a device with FOSC of 16 MHz, desired baud rate of 9600, Asynchronous mode, 8-bit BRG:

$$Desired\ Baud\ Rate = \frac{FOSC}{64([SPBRGH:SPBRGL] + 1)}$$

Solving for SPxBRGH:SPxBRGL:

$$X = \frac{\frac{FOSC}{Desired\ Baud\ Rate}}{64} - 1$$

$$= \frac{\frac{16000000}{9600}}{64} - 1$$

$$= [25.042] = 25$$

$$Calculated\ Baud\ Rate = \frac{16000000}{64(25 + 1)}$$

$$= 9615$$

$$Error = \frac{Calc.\ Baud\ Rate - Desired\ Baud\ Rate}{Desired\ Baud\ Rate}$$

$$= \frac{(9615 - 9600)}{9600} = 0.16\%$$

**TABLE 33-3: 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.

**TABLE 33-4: SUMMARY OF REGISTERS ASSOCIATED WITH THE BAUD RATE GENERATOR**

| 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 | 466 |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 465 |
| SP1BRGL | SP1BRG<7:0> | | | | | | | | 467 |
| SP1BRGH | SP1BRG<15:8> | | | | | | | | 467 |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 464 |

**Legend:** — = unimplemented location, read as '0'. Shaded cells are not used for the Baud Rate Generator.

* Page provides register information.

**TABLE 33-5: BAUD RATES FOR ASYNCHRONOUS MODES**

| BAUD RATE | SYNC = 0, BRGH = 0, BRG16 = 0 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Fosc = 32.000 MHz | | | Fosc = 20.000 MHz | | | Fosc = 18.432 MHz | | | Fosc = 11.0592 MHz | | |
| | Actual Rate | % Error | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG value (decimal) |
| 300 | — | — | — | — | — | — | — | — | — | — | — | — |
| 1200 | — | — | — | 1221 | 1.73 | 255 | 1200 | 0.00 | 239 | 1200 | 0.00 | 143 |
| 2400 | 2404 | 0.16 | 207 | 2404 | 0.16 | 129 | 2400 | 0.00 | 119 | 2400 | 0.00 | 71 |
| 9600 | 9615 | 0.16 | 51 | 9470 | -1.36 | 32 | 9600 | 0.00 | 29 | 9600 | 0.00 | 17 |
| 10417 | 10417 | 0.00 | 47 | 10417 | 0.00 | 29 | 10286 | -1.26 | 27 | 10165 | -2.42 | 16 |
| 19.2k | 19.23k | 0.16 | 25 | 19.53k | 1.73 | 15 | 19.20k | 0.00 | 14 | 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 | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG 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 | — | — | — | — | — | — | — | — | — | — | — | — |

| BAUD RATE | SYNC = 0, BRGH = 1, BRG16 = 0 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Fosc = 32.000 MHz | | | Fosc = 20.000 MHz | | | Fosc = 18.432 MHz | | | Fosc = 11.0592 MHz | | |
| | Actual Rate | % Error | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG value (decimal) |
| 300 | — | — | — | — | — | — | — | — | — | — | — | — |
| 1200 | — | — | — | — | — | — | — | — | — | — | — | — |
| 2400 | — | — | — | — | — | — | — | — | — | — | — | — |
| 9600 | 9615 | 0.16 | 207 | 9615 | 0.16 | 129 | 9600 | 0.00 | 119 | 9600 | 0.00 | 71 |
| 10417 | 10417 | 0.00 | 191 | 10417 | 0.00 | 119 | 10378 | -0.37 | 110 | 10473 | 0.53 | 65 |
| 19.2k | 19.23k | 0.16 | 103 | 19.23k | 0.16 | 64 | 19.20k | 0.00 | 59 | 19.20k | 0.00 | 35 |
| 57.6k | 57.14k | -0.79 | 34 | 56.82k | -1.36 | 21 | 57.60k | 0.00 | 19 | 57.60k | 0.00 | 11 |
| 115.2k | 117.64k | 2.12 | 16 | 113.64k | -1.36 | 10 | 115.2k | 0.00 | 9 | 115.2k | 0.00 | 5 |

**TABLE 33-5: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)**

| 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 | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG 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 = 20.000 MHz | | | Fosc = 18.432 MHz | | | Fosc = 11.0592 MHz | | |
| | Actual Rate | % Error | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG value (decimal) |
| 300 | 300.0 | 0.00 | 6666 | 300.0 | -0.01 | 4166 | 300.0 | 0.00 | 3839 | 300.0 | 0.00 | 2303 |
| 1200 | 1200 | -0.02 | 3332 | 1200 | -0.03 | 1041 | 1200 | 0.00 | 959 | 1200 | 0.00 | 575 |
| 2400 | 2401 | -0.04 | 832 | 2399 | -0.03 | 520 | 2400 | 0.00 | 479 | 2400 | 0.00 | 287 |
| 9600 | 9615 | 0.16 | 207 | 9615 | 0.16 | 129 | 9600 | 0.00 | 119 | 9600 | 0.00 | 71 |
| 10417 | 10417 | 0.00 | 191 | 10417 | 0.00 | 119 | 10378 | -0.37 | 110 | 10473 | 0.53 | 65 |
| 19.2k | 19.23k | 0.16 | 103 | 19.23k | 0.16 | 64 | 19.20k | 0.00 | 59 | 19.20k | 0.00 | 35 |
| 57.6k | 57.14k | -0.79 | 34 | 56.818 | -1.36 | 21 | 57.60k | 0.00 | 19 | 57.60k | 0.00 | 11 |
| 115.2k | 117.6k | 2.12 | 16 | 113.636 | -1.36 | 10 | 115.2k | 0.00 | 9 | 115.2k | 0.00 | 5 |

| 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 | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG value (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 | — | — | — |

**TABLE 33-5: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)**

| BAUD RATE | SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1 | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | FOSC = 32.000 MHz | | | FOSC = 20.000 MHz | | | FOSC = 18.432 MHz | | | FOSC = 11.0592 MHz | | |
| | Actual Rate | % Error | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG value (decimal) |
| 300 | 300.0 | 0.00 | 26666 | 300.0 | 0.00 | 16665 | 300.0 | 0.00 | 15359 | 300.0 | 0.00 | 9215 |
| 1200 | 1200 | 0.00 | 6666 | 1200 | -0.01 | 4166 | 1200 | 0.00 | 3839 | 1200 | 0.00 | 2303 |
| 2400 | 2400 | 0.01 | 3332 | 2400 | 0.02 | 2082 | 2400 | 0.00 | 1919 | 2400 | 0.00 | 1151 |
| 9600 | 9604 | 0.04 | 832 | 9597 | -0.03 | 520 | 9600 | 0.00 | 479 | 9600 | 0.00 | 287 |
| 10417 | 10417 | 0.00 | 767 | 10417 | 0.00 | 479 | 10425 | 0.08 | 441 | 10433 | 0.16 | 264 |
| 19.2k | 19.18k | -0.08 | 416 | 19.23k | 0.16 | 259 | 19.20k | 0.00 | 239 | 19.20k | 0.00 | 143 |
| 57.6k | 57.55k | -0.08 | 138 | 57.47k | -0.22 | 86 | 57.60k | 0.00 | 79 | 57.60k | 0.00 | 47 |
| 115.2k | 115.9k | 0.64 | 68 | 116.3k | 0.94 | 42 | 115.2k | 0.00 | 39 | 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 | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG value (decimal) | Actual Rate | % Error | SPxBRG value (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 | 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 | — | — | — |

### 33.4.1 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 RX signal, the RX 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. 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 SPxBRG begins counting up using the BRG counter clock as shown in Figure 33-6. The fifth rising edge will occur on the RX 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 RCIF interrupt flag is set. The value in the RCxREG needs to be read to clear the RCIF 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 33-6. 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 1/8th 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, auto-baud detection will occur on the byte following the Break character (see **Section 33.4.3 “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.

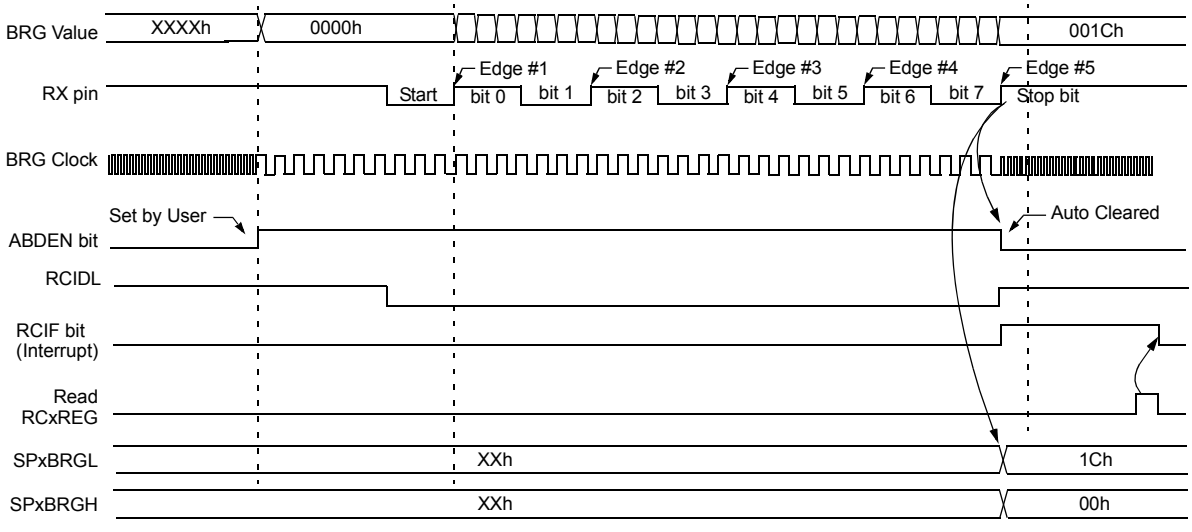
**3:** During the auto-baud process, the auto-baud counter starts counting at one. Upon completion of the auto-baud sequence, to achieve maximum accuracy, subtract 1 from the SPxBRGH:SPxBRGL register pair.

**TABLE 33-6: 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 the BRG16 setting.

**FIGURE 33-6: AUTOMATIC BAUD RATE CALIBRATION(1)**




**Note 1:** The ABD sequence requires the **EUSART** module to be configured in Asynchronous mode.

### 33.4.2 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. The overflow condition will set the RCIF flag. The counter continues to count until the fifth rising edge is detected on the RX pin. The RCIDL bit will remain false ('0') until the fifth rising edge at which time the RCIDL bit will be set. If the RCREG is read after the overflow occurs but before the fifth rising edge then the fifth rising edge will set the RCIF again.

Terminating the auto-baud process early to clear an overflow condition will prevent proper detection of the sync character fifth rising edge. If any falling edges of the sync character have not yet occurred when the ABDEN bit is cleared then those will be falsely detected as Start bits. The following steps are recommended to clear the overflow condition:

1. Read RCREG to clear RCIF.
2. If RCIDL is zero then wait for RCIF and repeat step 1.
3. Clear the ABDOVF bit.

### 33.4.3 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 RX/DT 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 RX/DT 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 RX/DT 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 RCIF interrupt coincident with the wake-up event. The interrupt is generated synchronously to the Q clocks in normal CPU operating modes (Figure 33-7), and asynchronously if the device is in Sleep mode (Figure 33-8). The interrupt condition is cleared by reading the RCxREG register.

The WUE bit is automatically cleared by the low-to-high transition on the RX 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.

#### 33.4.3.1 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 Start-up Time

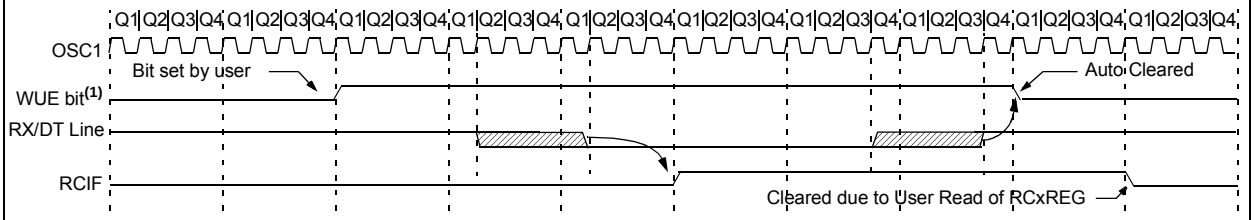
Oscillator start-up time must be considered, especially in applications using oscillators with longer start-up intervals (i.e., LP, XT or HS/PLL mode). The Sync Break (or wake-up signal) character must be of sufficient length, and be followed by a sufficient interval, to allow enough time for the selected oscillator to start and provide proper initialization of the EUSART.

WUE Bit

The wake-up event causes a receive interrupt by setting the RCIF bit. The WUE bit is cleared in hardware by a rising edge on RX/DT. The interrupt condition is then cleared in software by reading the RCxREG register and discarding its contents.

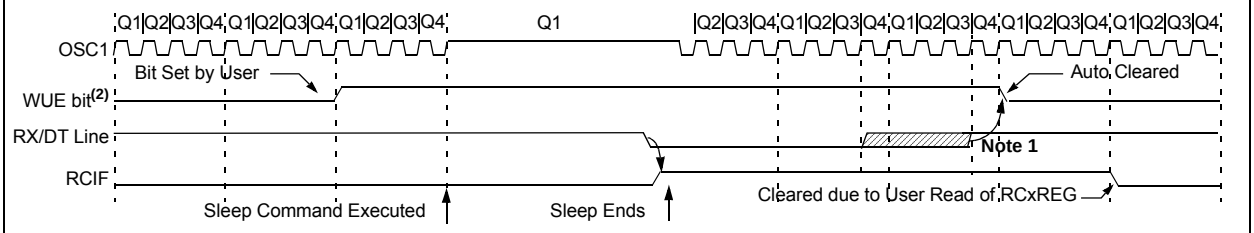
To ensure that no actual data is lost, check the RCIDL bit to verify that a receive operation is not in process before setting the WUE bit. If a receive operation is not occurring, the WUE bit may then be set just prior to entering the Sleep mode.

**FIGURE 33-7: AUTO-WAKE-UP BIT (WUE) TIMING DURING NORMAL OPERATION**




**Note 1:** The **EUSART** remains in Idle while the WUE bit is set.

**FIGURE 33-8: 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.

### 33.4.4 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 33-9 for the timing of the Break character sequence.

#### 33.4.4.1 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 TXIF, the next data byte can be written to TXxREG.

### 33.4.5 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.

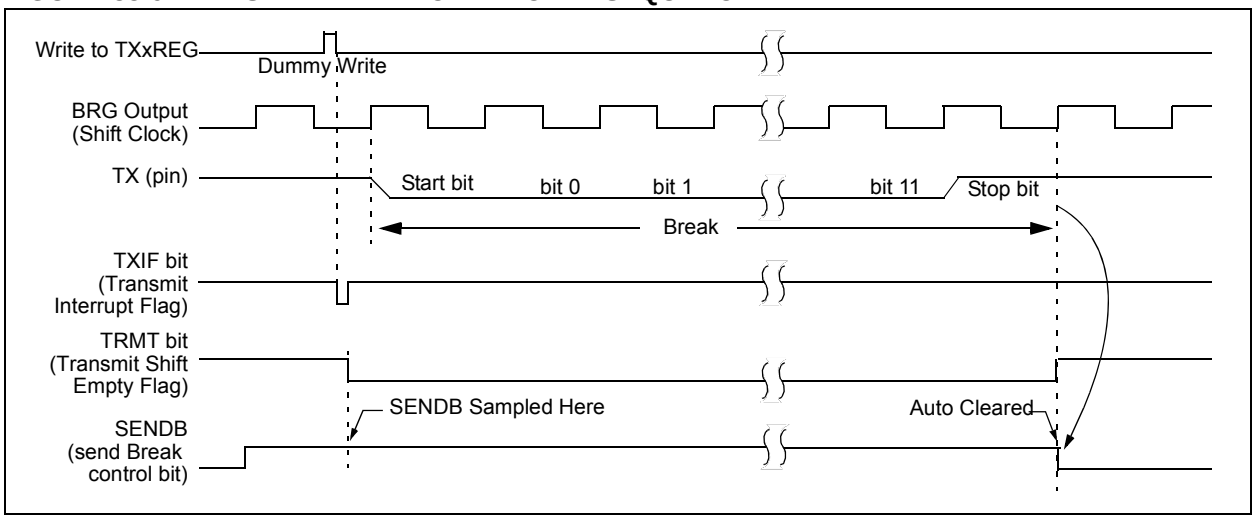
A Break character has been received when:

- RCIF bit is set
- FERR bit is set
- RCxREG = 00h

The second method uses the Auto-Wake-up feature described in **Section 33.4.3 "Auto-Wake-up on Break"**. By enabling this feature, the EUSART will sample the next two transitions on RX/DT, cause an RCIF 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 33-9: SEND BREAK CHARACTER SEQUENCE**



## 33.5 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.

### 33.5.1 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.

#### 33.5.1.1 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 TX/CK line. The TX/CK 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.

#### 33.5.1.2 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. 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.

#### 33.5.1.3 Synchronous Master Transmission

Data is transferred out of the device on the RX/DT pin. The RX/DT and TX/CK 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.

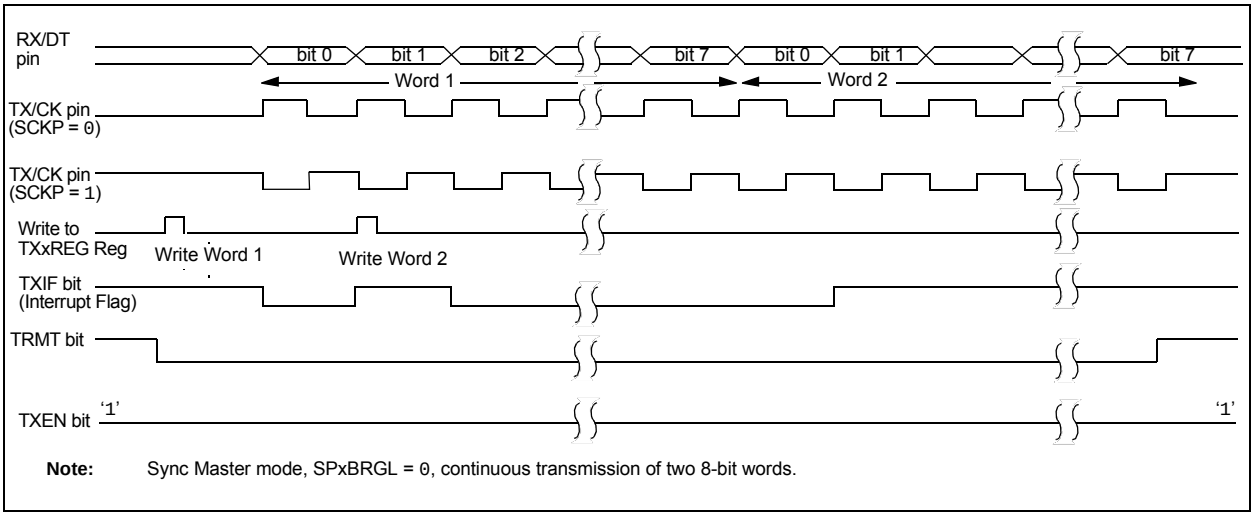
Each data bit changes on the leading edge of the master clock and remains valid until the subsequent leading clock edge.

| **Note:** | The TSR register is not mapped in data memory, so it is not available to the user. |
|---|---|

#### 33.5.1.4 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 33.4 "EUSART Baud Rate Generator (BRG)"**).
2. Enable the synchronous master serial port by setting bits SYNC, SPEN and CSRC.
3. Disable Receive mode by clearing bits SREN and CREN.
4. Enable Transmit mode by setting the TXEN bit.
5. If 9-bit transmission is desired, set the TX9 bit.
6. If interrupts are desired, set the TXIE bit of the PIE1 register and the GIE and PEIE bits of the INTCON register.
7. If 9-bit transmission is selected, the ninth bit should be loaded in the TX9D bit.
8. Start transmission by loading data to the TXxREG register.

**FIGURE 33-10: SYNCHRONOUS TRANSMISSION**




**Note:** Sync Master mode, SPxBRGL = 0, continuous transmission of two 8-bit words.

**FIGURE 33-11: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)**

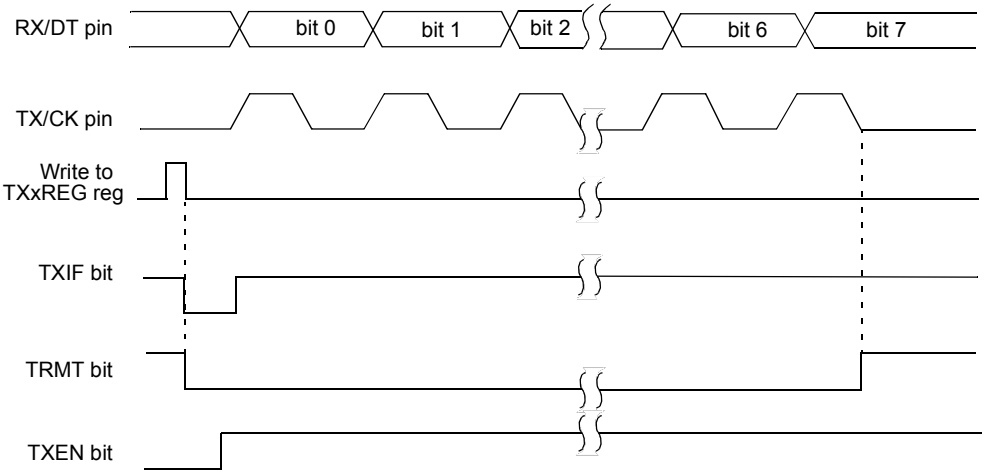


**TABLE 33-7: SUMMARY OF REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER TRANSMISSION**

| Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Register on Page |
|---|---|---|---|---|---|---|---|---|---|
| ANSELA | — | — | ANSA5 | ANSA4 | ANSA3 | ANSA2 | ANSA1 | ANSA0 | 153 |
| ANSELB | — | — | ANSB5 | ANSB4 | ANSB3 | ANSB2 | ANSB1 | ANSB0 | 158 |
| ANSELC | ANSC7 | ANSC6 | ANSC5 | ANSC4 | ANSC3 | ANSC2 | — | — | 163 |
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 466 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 108 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 109 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 115 |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 465 |
| RxyPPS | — | — | RxyPPS<5:0> | | | | | | 172 |
| SP1BRGL | SP1BRG<7:0> | | | | | | | | 467 |
| SP1BRGH | SP1BRG<15:8> | | | | | | | | 467 |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 152 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 157 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 162 |
| TX1REG | EUSART Transmit Data Register | | | | | | | | 456* |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 464 |

**Legend:** — = unimplemented location, read as '0'. Shaded cells are not used for synchronous master transmission.
* Page provides register information.

#### 33.5.1.5 Synchronous Master Reception

Data is received at the RX/DT pin. The RX/DT pin output driver is automatically disabled 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 RX/DT pin on the trailing edge of the TX/CK clock pin and is shifted into the Receive Shift Register (RSR). When a complete character is received into the RSR, the RCIF 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 RCIF bit remains set as long as there are unread characters in the receive FIFO.

| | |
|---|---|
| **Note:** | If the RX/DT function is on an analog pin, the corresponding ANSEL bit must be cleared for the receiver to function. |

#### 33.5.1.6 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 TX/CK line. The TX/CK pin output driver is automatically disabled 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.

| | |
|---|---|
| **Note:** | If the device is configured as a slave and the TX/CK function is on an analog pin, the corresponding ANSEL bit must be cleared. |

#### 33.5.1.7 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.

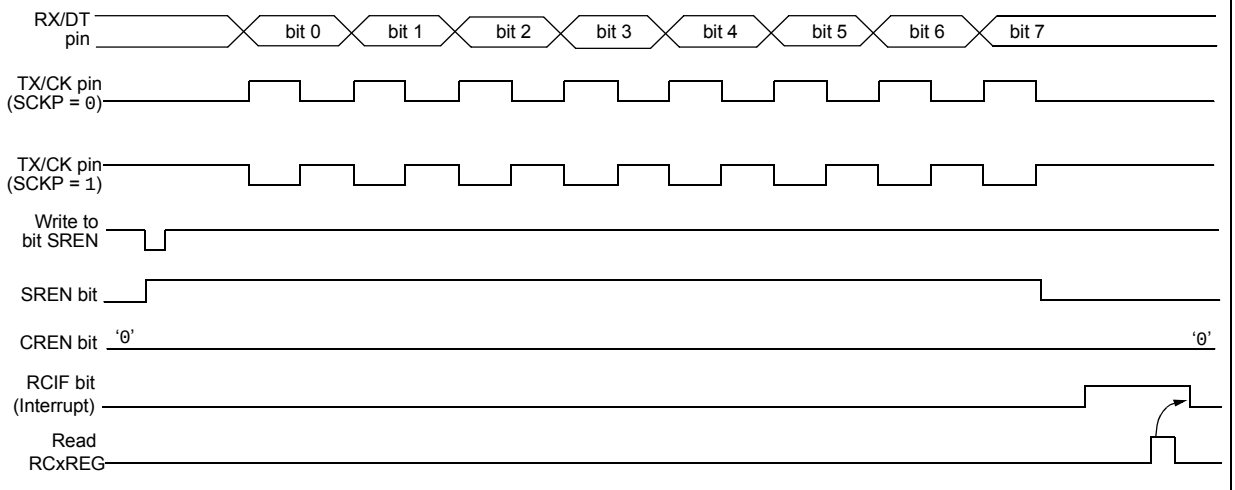
#### 33.5.1.8 Receiving 9-bit Characters

The EUSART supports 9-bit character reception. When the RX9 bit of the RCxSTA register is set, the EUSART will shift 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.

#### 33.5.1.9 Synchronous Master Reception Set-up:

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. Clear the ANSEL bit for the RX pin (if applicable).
3. Enable the synchronous master serial port by setting bits SYNC, SPEN and CSRC.
4. Ensure bits CREN and SREN are clear.
5. If interrupts are desired, set the RCIE bit of the PIE1 register and the GIE and PEIE bits of the INTCON register.
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 RCIF will be set when reception of a character is complete. An interrupt will be generated if the enable bit RCIE 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 33-12: SYNCHRONOUS RECEPTION (MASTER MODE, SREN)**




**Note:** Timing diagram demonstrates Sync Master mode with bit SREN = 1 and bit BRGH = 0.

**TABLE 33-8: SUMMARY OF 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 |
|---|---|---|---|---|---|---|---|---|---|
| ANSELA | — | — | ANSA5 | ANSA4 | ANSA3 | ANSA2 | ANSA1 | ANSA0 | 153 |
| ANSELB | — | — | ANSB5 | ANSB4 | ANSB3 | ANSB2 | ANSB1 | ANSB0 | 158 |
| ANSELC | ANSC7 | ANSC6 | ANSC5 | ANSC4 | ANSC3 | ANSC2 | — | — | 163 |
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 466 |
| CKPPS | — | — | CKPPS<5:0> | | | | | | 172, 174 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 108 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 109 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 115 |
| RC1REG | EUSART Receive Data Register | | | | | | | | 459* |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 465 |
| RXPPS | — | — | RXPPS<5:0> | | | | | | 172, 174 |
| RxyPPS | — | — | RxyPPS<5:0> | | | | | | 172 |
| SP1BRGL | SP1BRG<7:0> | | | | | | | | 467* |
| SP1BRGH | SP1BRG<15:8> | | | | | | | | 467* |
| TRISA | TRISA5 | TRISA4 | TRISA5 | TRISA4 | TRISA5 | TRISA2 | TRISA1 | TRISA0 | 152 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 157 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 162 |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 464 |

**Legend:** — = unimplemented location, read as '0'. Shaded cells are not used for synchronous master reception.

* Page provides register information.

### 33.5.2 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.

#### 33.5.2.1 EUSART Synchronous Slave Transmit

The operation of the Synchronous Master and Slave modes are identical (see **Section 33.5.1.3 "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 the TXxREG register.
3. The TXIF 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 TXIF bit will now be set.
5. If the PEIE and TXIE 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.

#### 33.5.2.2 Synchronous Slave Transmission Set-up:

1. Set the SYNC and SPEN bits and clear the CSRC bit.
2. Clear the ANSEL bit for the CK pin (if applicable).
3. Clear the CREN and SREN bits.
4. If interrupts are desired, set the TXIE bit of the PIE1 register and the GIE and PEIE bits of the INTCON register.
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 33-9: SUMMARY OF 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 |
|---|---|---|---|---|---|---|---|---|---|
| ANSELA | — | — | ANSA5 | ANSA4 | ANSA3 | ANSA2 | ANSA1 | ANSA0 | 153 |
| ANSELB | — | — | ANSB5 | ANSB4 | ANSB3 | ANSB2 | ANSB1 | ANSB0 | 158 |
| ANSELC | ANSC7 | ANSC6 | ANSC5 | ANSC4 | ANSC3 | ANSC2 | — | — | 163 |
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 466 |
| CKPPS | — | — | CKPPS<5:0> | | | | | | 172, 174 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 108 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 109 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 115 |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 465 |
| RXPPS | — | — | RXPPS<5:0> | | | | | | 172, 174 |
| RxyPPS | — | — | RxyPPS<5:0> | | | | | | 172 |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 152 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 157 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 162 |
| TX1REG | EUSART Transmit Data Register | | | | | | | | 456* |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 464 |

**Legend:** — = unimplemented location, read as '0'. Shaded cells are not used for synchronous slave transmission.

* Page provides register information.

#### 33.5.2.3 EUSART Synchronous Slave Reception

The operation of the Synchronous Master and Slave modes is identical (**Section 33.5.1.5 "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 RCIE 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.

#### 33.5.2.4 Synchronous Slave Reception Set-up:

1. Set the SYNC and SPEN bits and clear the CSRC bit.
2. Clear the ANSEL bit for both the CK and DT pins (if applicable).
3. If interrupts are desired, set the RCIE bit of the PIE1 register and the GIE and PEIE bits of the INTCON register.
4. If 9-bit reception is desired, set the RX9 bit.
5. Set the CREN bit to enable reception.
6. The RCIF bit will be set when reception is complete. An interrupt will be generated if the RCIE 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 33-10: SUMMARY OF 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 |
|---|---|---|---|---|---|---|---|---|---|
| ANSELA | — | — | ANSA5 | ANSA4 | ANSA3 | ANSA2 | ANSA1 | ANSA0 | 153 |
| ANSELB | — | — | ANSB5 | ANSB4 | ANSB3 | ANSB2 | ANSB1 | ANSB0 | 158 |
| ANSELC | ANSC7 | ANSC6 | ANSC5 | ANSC4 | ANSC3 | ANSC2 | — | — | 163 |
| BAUD1CON | ABDOVF | RCIDL | — | SCKP | BRG16 | — | WUE | ABDEN | 466 |
| CKPPS | — | — | CKPPS<5:0> | | | | | | 172, 174 |
| INTCON | GIE | PEIE | TMR0IE | INTE | IOCIE | TMR0IF | INTF | IOCIF | 108 |
| PIE1 | TMR1GIE | ADIE | RCIE | TXIE | SSP1IE | CCP1IE | TMR2IE | TMR1IE | 109 |
| PIR1 | TMR1GIF | ADIF | RCIF | TXIF | SSP1IF | CCP1IF | TMR2IF | TMR1IF | 115 |
| RC1REG | EUSART Receive Data Register | | | | | | | | 459* |
| RC1STA | SPEN | RX9 | SREN | CREN | ADDEN | FERR | OERR | RX9D | 465 |
| RXPPS | — | — | RXPPS<5:0> | | | | | | 172, 174 |
| TRISA | TRISA7 | TRISA6 | TRISA5 | TRISA4 | TRISA3 | TRISA2 | TRISA1 | TRISA0 | 152 |
| TRISB | TRISB7 | TRISB6 | TRISB5 | TRISB4 | TRISB3 | TRISB2 | TRISB1 | TRISB0 | 157 |
| TRISC | TRISC7 | TRISC6 | TRISC5 | TRISC4 | TRISC3 | TRISC2 | TRISC1 | TRISC0 | 162 |
| TX1STA | CSRC | TX9 | TXEN | SYNC | SENDB | BRGH | TRMT | TX9D | 464 |

**Legend:** — = unimplemented location, read as '0'. Shaded cells are not used for synchronous slave reception.

* Page provides register information.

## 33.6 EUSART Operation During Sleep

The EUSART will remain active during Sleep only in the Synchronous Slave mode. All other modes require the system clock and therefore cannot generate the necessary signals to run the Transmit or Receive Shift registers during Sleep.

Synchronous Slave mode uses an externally generated clock to run the Transmit and Receive Shift registers.

### 33.6.1 SYNCHRONOUS RECEIVE DURING SLEEP

To receive during Sleep, all the following conditions must be met before entering Sleep mode:

- RCxSTA and TXxSTA Control registers must be configured for Synchronous Slave Reception (see **Section 33.5.2.4 "Synchronous Slave Reception Set-up:"**).
- If interrupts are desired, set the RCIE bit of the PIE1 register and the GIE and PEIE bits of the INTCON register.
- The RCIF interrupt flag must be cleared by reading RCxREG to unload any pending characters in the receive buffer.

Upon entering Sleep mode, the device will be ready to accept data and clocks on the RX/DT and TX/CK pins, respectively. When the data word has been completely clocked in by the external device, the RCIF interrupt flag bit of the PIR1 register will be set. Thereby, waking the processor from Sleep.

Upon waking from Sleep, the instruction following the SLEEP instruction will be executed. If the Global Interrupt Enable (GIE) bit of the INTCON register is also set, then the Interrupt Service Routine at address 004h will be called.

### 33.6.2 SYNCHRONOUS TRANSMIT DURING SLEEP

To transmit during Sleep, all the following conditions must be met before entering Sleep mode:

- The RCxSTA and TXxSTA Control registers must be configured for synchronous slave transmission (see **Section 33.5.2.2 "Synchronous Slave Transmission Set-up:"**).
- The TXIF interrupt flag must be cleared by writing the output data to the TXxREG, thereby filling the TSR and transmit buffer.
- If interrupts are desired, set the TXIE bit of the PIE1 register and the PEIE bit of the INTCON register.
- Interrupt enable bits TXIE of the PIE1 register and PEIE of the INTCON register must set.

Upon entering Sleep mode, the device will be ready to accept clocks on TX/CK pin and transmit data on the RX/DT pin. When the data word in the TSR has been completely clocked out by the external device, the pending byte in the TXxREG will transfer to the TSR and the TXIF flag will be set. Thereby, waking the processor from Sleep. At this point, the TXxREG is available to accept another character for transmission, which will clear the TXIF flag.

Upon waking from Sleep, the instruction following the SLEEP instruction will be executed. If the Global Interrupt Enable (GIE) bit is also set then the Interrupt Service Routine at address 0004h will be called.

## 34.0 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
- $\overline{MCLR}/V_{PP}$
- $V_{DD}$
- $V_{SS}$

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 **"***PIC16(L)F177X Memory Programming Specification***"** (DS40001792).

### 34.1 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 $\overline{MCLR}/V_{PP}$ to $V_{IHH}$.

### 34.2 Low-Voltage Programming Entry Mode

The Low-Voltage Programming Entry mode allows the PIC® Flash MCUs to be programmed using $V_{DD}$ 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. $\overline{MCLR}$ is brought to $V_{IL}$.
2. A 32-bit key sequence is presented on ICSPDAT, while clocking ICSPCLK.

Once the key sequence is complete, $\overline{MCLR}$ must be held at $V_{IL}$ for as long as Program/Verify mode is to be maintained.

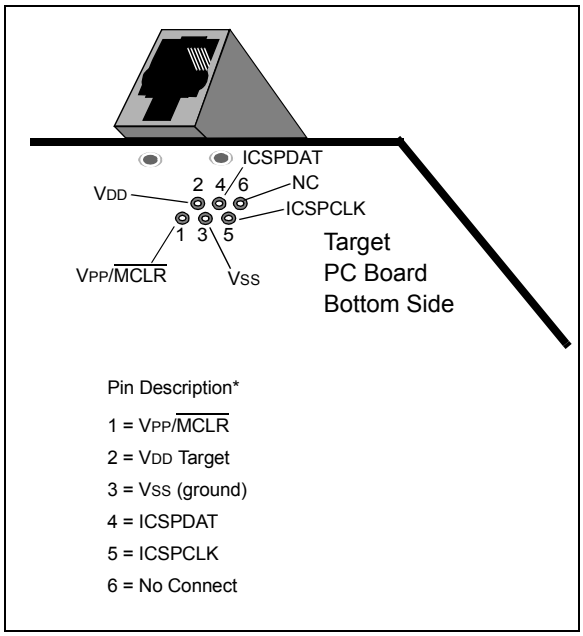
If low-voltage programming is enabled (LVP = 1), the $\overline{MCLR}$ Reset function is automatically enabled and cannot be disabled. See **Section 6.5 "MCLR"** for more information.

The LVP bit can only be reprogrammed to '0' by using the High-Voltage Programming mode.

### 34.3 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 34-1.

**FIGURE 34-1: ICD RJ-11 STYLE CONNECTOR INTERFACE**

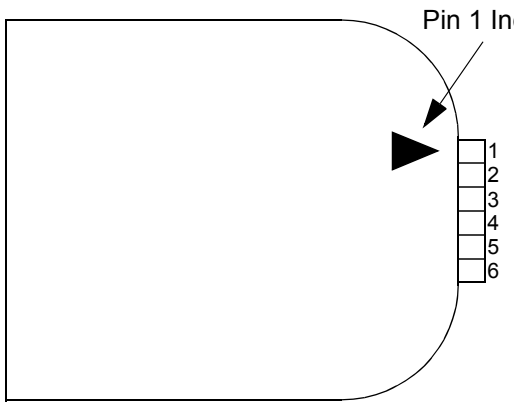



Another connector often found in use with the PICkit™ programmers is a standard 6-pin header with 0.1 inch spacing. Refer to Figure 34-2.

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 34-3 for more information.

**FIGURE 34-2: PICkit™ PROGRAMMER STYLE CONNECTOR INTERFACE**




Pin Description*

1 = VPP/$\overline{\text{MCLR}}$

2 = VDD Target

3 = VSS (ground)

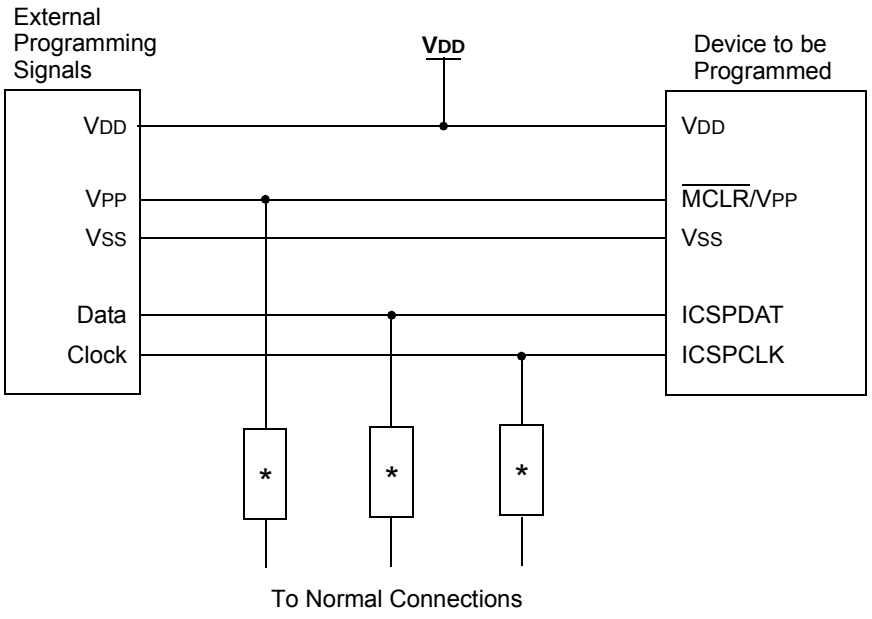
4 = ICSPDAT

5 = ICSPCLK

6 = No Connect

* The 6-pin header (0.100" spacing) accepts 0.025" square pins.

**FIGURE 34-3: TYPICAL CONNECTION FOR ICSP™ PROGRAMMING**




* Isolation devices (as required).

## 35.0 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 35-3 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.

### 35.1 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 35-1: OPCODE FIELD DESCRIPTIONS**

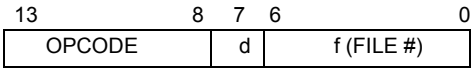
| Field | Description |
|---|---|
| f | Register file address (0x00 to 0x7F) |
| W | Working register (accumulator) |
| b | Bit address within an 8-bit file register |
| k | Literal field, constant data or label |
| x | Don't care location (= 0 or 1).<br>The assembler will generate code with x = 0.<br>It 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 35-2: 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 |

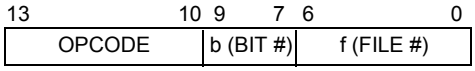
**FIGURE 35-1: GENERAL FORMAT FOR INSTRUCTIONS**

**Byte-oriented file register operations**




d = 0 for destination W
d = 1 for destination f
f = 7-bit file register address

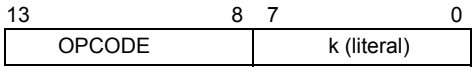
**Bit-oriented file register operations**




b = 3-bit bit address
f = 7-bit file register address

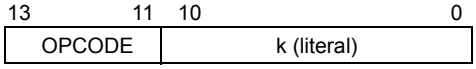
**Literal and control operations**

General



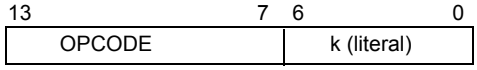

k = 8-bit immediate value

CALL and GOTO instructions only



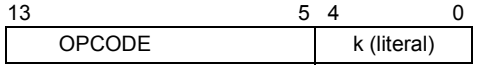

k = 11-bit immediate value

MOVLP instruction only



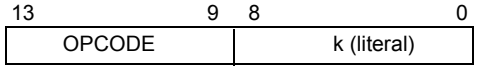

k = 7-bit immediate value

MOVLB instruction only



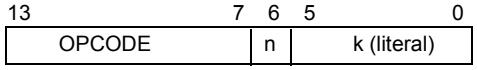

k = 5-bit immediate value

BRA instruction only



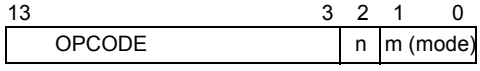

k = 9-bit immediate value

FSR Offset instructions




n = appropriate FSR
k = 6-bit immediate value

FSR Increment instructions




n = appropriate FSR
m = 2-bit mode value

OPCODE only

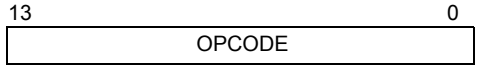


**TABLE 35-3: PIC16(L)F1773/6 INSTRUCTION SET**

| 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.

**TABLE 35-3: PIC16(L)F1773/6 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.

## 35.2 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**

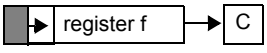
| | |
|---|---|
| Syntax: | [ *label* ] ANDLW k |
| Operands: | 0 ≤ k ≤ 255 |
| Operation: | (W) .AND. (k) → (W) |
| Status Affected: | Z |
| Description: | The contents of W register are AND'ed with the 8-bit literal 'k'. The result is placed in the W register. |

**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'. |



**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: | $0 \le k \le 2047$ |
| 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: | $0 \le f \le 127$ |
| 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. Status bits $\overline{TO}$ and $\overline{PD}$ are set. |

**COMF** **Complement f**

| | |
|---|---|
| Syntax: | [ *label* ] COMF f,d |
| Operands: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| Operation: | ($\bar{f}$) → (destination) |
| Status Affected: | Z |
| Description: | The contents of register ‘f’ are complemented. If ‘d’ is ‘0’, the result is stored in W. If ‘d’ is ‘1’, the result is stored back in register ‘f’. |

**DECF** **Decrement f**

| | |
|---|---|
| Syntax: | [ *label* ] DECF f,d |
| Operands: | $0 \le f \le 127$<br>$d \in [0,1]$ |
| Operation: | (f) - 1 → (destination) |
| Status Affected: | Z |
| Description: | Decrement register ‘f’. If ‘d’ is ‘0’, the result is stored in the W register. If ‘d’ is ‘1’, the result is stored back in register ‘f’. |

**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'. |

**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**

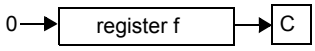
Syntax: [ *label* ] LSRF f {,d}

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

**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 ≤ 31

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: $n \in [0,1]$
$mm \in [00,01, 10, 11]$
$-32 \leq k \leq 31$

Operation: $W \rightarrow INDFn$
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 $\overline{RI}$ flag of the PCON register.

Status Affected: None

Description: This instruction provides a way to execute a hardware Reset by software.

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

Syntax: [ *label* ] RETFIE k

Operands: None

Operation: TOS → PC,
1 → GIE

Status Affected: None

Description: Return from Interrupt. Stack is POPed and Top-of-Stack (TOS) is loaded in the PC. Interrupts are enabled by setting Global Interrupt Enable bit, GIE (INTCON<7>). This is a 2-cycle instruction.

Words: 1

Cycles: 2

Example:

```
RETFIE
```

After Interrupt
PC = TOS
GIE = 1

**RETLW** **Return with literal in W**

Syntax: [ *label* ] RETLW k

Operands: 0 ≤ k ≤ 255

Operation: k → (W);
TOS → PC

Status Affected: None

Description: The W register is loaded with the 8-bit literal 'k'. The program counter is loaded from the top of the stack (the return address). This is a 2-cycle instruction.

Words: 1

Cycles: 2

Example:

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

Before Instruction
W = 0x07
After Instruction
W = value of k8

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

Syntax: [ *label* ] RETURN

Operands: None

Operation: TOS → PC

Status Affected: None

Description: Return from subroutine. The stack is POPed and the top of the stack (TOS) is loaded into the program counter. This is a 2-cycle instruction.

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

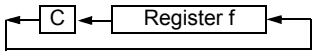
Syntax: [ *label* ] RLF f,d

Operands: 0 ≤ f ≤ 127
d ∈ [0,1]

Operation: See description below

Status Affected: C

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




Words: 1

Cycles: 1

Example:

```
RLF     REG1,0
```

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

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

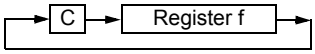
Syntax: [ *label* ] RRF f,d

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

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'.

**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'. |

# 36.0 ELECTRICAL SPECIFICATIONS

## 36.1 Absolute Maximum Ratings(†)

Ambient temperature under bias ..... -40°C to +125°C
Storage temperature ..... -65°C to +150°C
Voltage on pins with respect to $V_{SS}$
- on $V_{DD}$ pin
  - PIC16F1773/6 ..... -0.3V to +6.5V
  - PIC16LF1773/6 ..... -0.3V to +4.0V
- on $\overline{MCLR}$ pin ..... -0.3V to +9.0V
- on all other pins ..... -0.3V to ($V_{DD}$ + 0.3V)

Maximum current
- on $V_{SS}$ pin(1)
  - -40°C ≤ $T_A$ ≤ +85°C ..... 250 mA
  - -40°C ≤ $T_A$ ≤ +125°C ..... 85 mA
- on $V_{DD}$ pin(1)
  - -40°C ≤ $T_A$ ≤ +85°C ..... 250 mA
  - -40°C ≤ $T_A$ ≤ +125°C ..... 85 mA
- Current by any standard I/O pin ..... 50 mA
- Sunk by any High Current I/O pin ..... 100 mA
- Sourced by any Op Amp ..... 100 mA

Clamp current, $I_K$ ($V_{PIN}$ < 0 or $V_{PIN}$ > $V_{DD}$) ..... ±20 mA
Total power dissipation(2) ..... 800 mW

**Note 1:** Maximum current rating requires even load distribution across I/O pins. Maximum current rating may be limited by the device package power dissipation characterizations, see Table 36-6: Thermal Characteristics to calculate device specifications.

**2:** Power dissipation is calculated as follows:

$Pdis = V_{DD} * \{Idd - \Sigma Ioh\} + \Sigma\{V_{DD} - Voh) * Ioh\} + \Sigma(Vol * Iol)$.

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

## 36.2 Standard Operating Conditions

The standard operating conditions for any device are defined as:

Operating Voltage: $V_{DDMIN} \leq V_{DD} \leq V_{DDMAX}$
Operating Temperature: $T_{A\_MIN} \leq T_A \leq T_{A\_MAX}$

**$V_{DD}$ — Operating Supply Voltage(1)**

PIC16LF1773/6

$V_{DDMIN}$ (Fosc ≤ 16 MHz) ..... +1.8V
$V_{DDMIN}$ (Fosc > 16 MHz) ..... +2.5V
$V_{DDMAX}$ ..... +3.6V

PIC16F1773/6

$V_{DDMIN}$ (Fosc ≤ 16 MHz) ..... +2.3V
$V_{DDMIN}$ (Fosc > 16 MHz) ..... +2.5V
$V_{DDMAX}$ ..... +5.5V

**$T_A$ — Operating Ambient Temperature Range**

Industrial Temperature

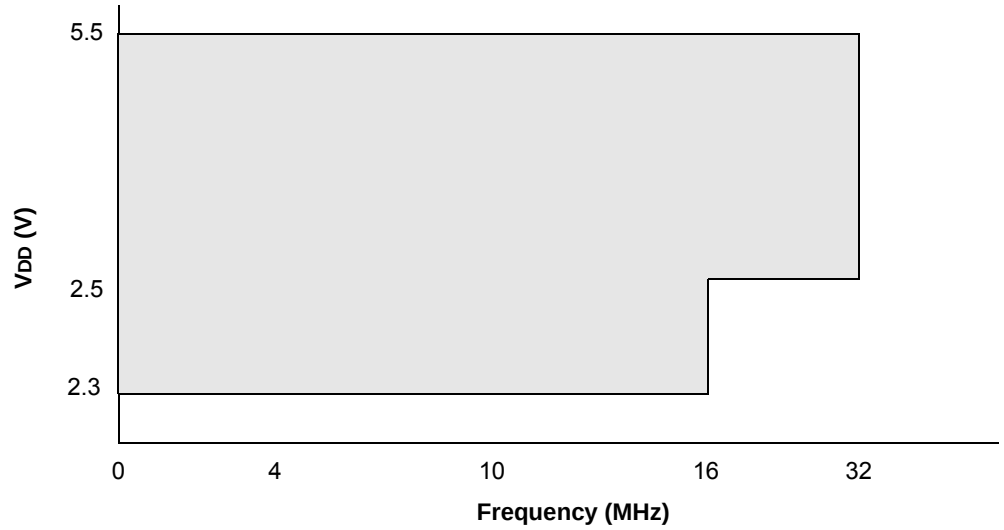
$T_{A\_MIN}$ ..... -40°C
$T_{A\_MAX}$ ..... +85°C

Extended Temperature

$T_{A\_MIN}$ ..... -40°C
$T_{A\_MAX}$ ..... +125°C

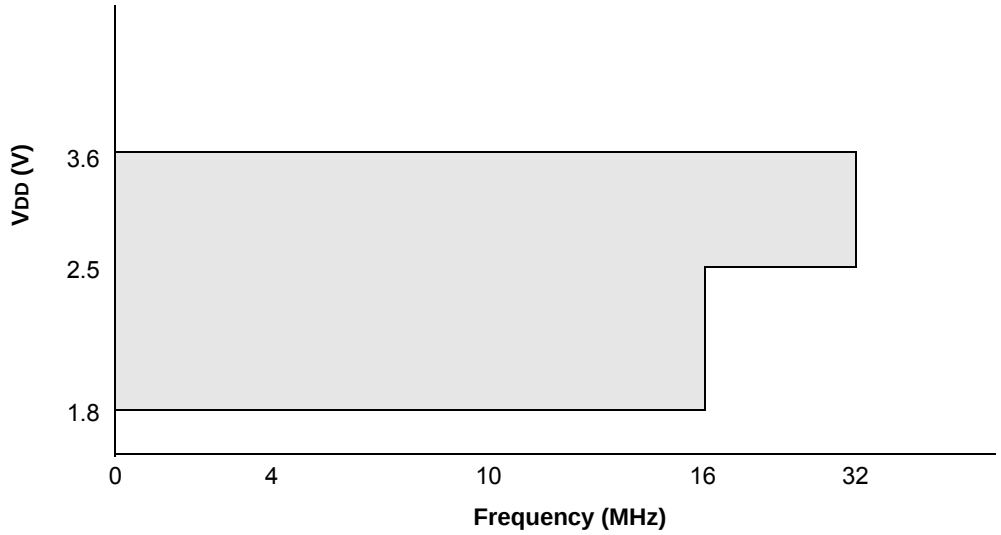
**Note 1:** See Parameter D001, DS Characteristics: Supply Voltage.

**FIGURE 36-1: VOLTAGE FREQUENCY GRAPH, -40°C ≤ TA ≤ +125°C, PIC16F1773/6 ONLY**




**Note 1:** The shaded region indicates the permissible combinations of voltage and frequency.
**2:** Refer to Table 36-7 for each Oscillator mode's supported frequencies.

**FIGURE 36-2: VOLTAGE FREQUENCY GRAPH, -40°C ≤ TA ≤ +125°C, PIC16LF1773/6 ONLY**




**Note 1:** The shaded region indicates the permissible combinations of voltage and frequency.
**2:** Refer to Table 36-7 for each Oscillator mode's supported frequencies.

## 36.3 DC Characteristics

### TABLE 36-1: SUPPLY VOLTAGE

| PIC16LF1773/6 | | | Standard Operating Conditions (unless otherwise stated) | | | | |
|---|---|---|---|---|---|---|---|
| PIC16F1773/6 | | | | | | | |
| Param. No. | Sym. | Characteristic | Min. | Typ† | Max. | Units | Conditions |
| D001 | VDD | **Supply Voltage** | | | | | |
| | | | VDDMIN<br>1.8<br>2.5 | <br>—<br>— | VDDMAX<br>3.6<br>3.6 | <br>V<br>V | <br>FOSC ≤ 16 MHz<br>FOSC ≤ 32 MHz |
| D001 | | PIC16F1773/6 | 2.3<br>2.5 | —<br>— | 5.5<br>5.5 | V<br>V | FOSC ≤ 16 MHz:<br>FOSC ≤ 32 MHz |
| D002* | VDR | **RAM Data Retention Voltage(1)** | | | | | |
| | | | 1.5 | — | — | V | Device in Sleep mode |
| D002* | | | 1.7 | — | — | V | Device in Sleep mode |
| D002A* | VPOR | **Power-on Reset Release Voltage(2)** | | | | | |
| | | | — | 1.6 | — | V | |
| D002A* | | | — | 1.6 | — | V | |
| D002B* | VPORR* | **Power-on Reset Rearm Voltage(2)** | | | | | |
| | | | — | 0.8 | — | V | |
| D002B* | | | — | 1.5 | — | V | |
| D003 | VFVR | **Fixed Voltage Reference Voltage(3)** | -4<br>-4<br>-7 | —<br>—<br>— | +4<br>+4<br>+7 | %<br>%<br>% | 1x gain, 1.024, VDD ≥ 2.5V, -40°C to +85°C<br>2x gain, 2.048, VDD ≥ 2.5V, -40°C to +85°C<br>4x gain, 4.096, VDD ≥ 4.5V, -40°C to +85°C |
| D004* | SVDD | **VDD Rise Rate** | 0.05 | — | — | V/ms | Ensures that the Power-on Reset signal is released properly. |

* These parameters are characterized but not tested.

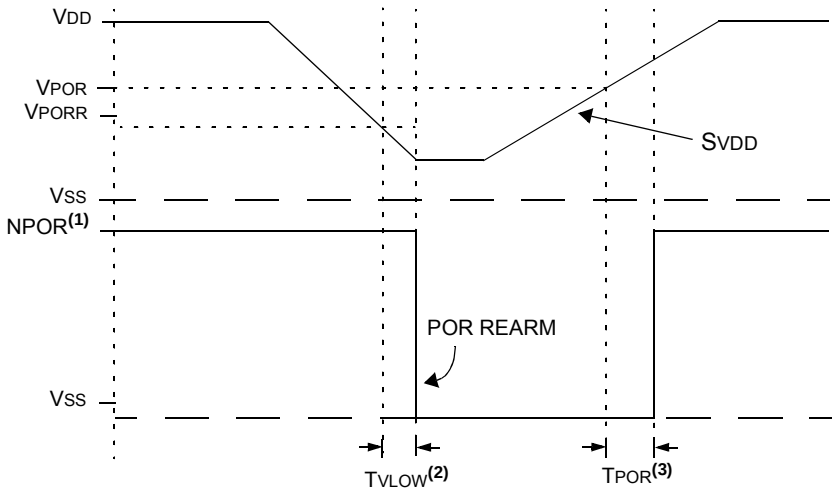
† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

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

**2:** See Figure 36-3: POR and POR Rearm with Slow Rising VDD.

**3:** Industrial temperature range only.

**FIGURE 36-3: POR AND POR REARM WITH SLOW RISING VDD**




**Note 1:** When NPOR is low, the device is held in Reset.
**2:** TVLOW 1 μs typical.
**3:** TPOR 2.7 μs typical.

**TABLE 36-2: SUPPLY CURRENT ($I_{DD}$)(1,2)**

| PIC16LF1773/6 | | Standard Operating Conditions (unless otherwise stated) | | | | | |
|---|---|---|---|---|---|---|---|
| **PIC16F1773/6** | | | | | | | |
| **Param No.** | **Device Characteristics** | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** | |
| | | | | | | **$V_{DD}$** | **Note** |
| D009 | LDO Regulator | — | 75 | — | µA | — | High-Power mode, normal operation |
| | | — | 15 | — | µA | — | Sleep, VREGCON<1> = 0 |
| | | — | 0.3 | — | µA | — | Sleep, VREGCON<1> = 1 |
| D010 | | — | 8 | 25 | µA | 1.8 | $F_{OSC}$ = 32 kHz, LP Oscillator mode, -40°C ≤ $T_A$ ≤ +85°C |
| | | — | 12 | 30 | µA | 3.0 | |
| D010 | | — | 15 | 30 | µA | 2.3 | $F_{OSC}$ = 32 kHz, LP Oscillator mode **(Note 4)** -40°C ≤ $T_A$ ≤ +85°C |
| | | — | 17 | 34 | µA | 3.0 | |
| | | — | 21 | 46 | µA | 5.0 | |
| D012 | | — | 140 | 440 | µA | 1.8 | $F_{OSC}$ = 4 MHz, XT Oscillator mode |
| | | — | 250 | 620 | µA | 3.0 | |
| D012 | | — | 210 | 530 | µA | 2.3 | $F_{OSC}$ = 4 MHz, XT Oscillator mode |
| | | — | 280 | 680 | µA | 3.0 | |
| | | — | 340 | 790 | µA | 5.0 | |
| D014 | | — | 115 | 380 | µA | 1.8 | $F_{OSC}$ = 4 MHz, External Clock (ECM), Medium Power mode |
| | | — | 210 | 520 | µA | 3.0 | |
| D014 | | — | 180 | 480 | µA | 2.3 | $F_{OSC}$ = 4 MHz, External Clock (ECM), Medium Power mode |
| | | — | 240 | 600 | µA | 3.0 | |
| | | — | 300 | 690 | µA | 5.0 | |
| D015 | | — | 2.1 | 4 | mA | 3.0 | $F_{OSC}$ = 32 MHz, External Clock (ECH), High-Power mode |
| | | — | 2.5 | 4.5 | mA | 3.6 | |
| D015 | | — | 2.1 | 3.8 | mA | 3.0 | $F_{OSC}$ = 32 MHz, External Clock (ECH), High-Power mode |
| | | — | 2.2 | 4.2 | mA | 5.0 | |

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note 1:** The test conditions for all $I_{DD}$ measurements in active operation mode are: OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to $V_{DD}$; $\overline{MCLR}$ = $V_{DD}$; 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 EXTRC oscillator configurations, current through $R_{EXT}$ is not included. The current through the resistor can be extended by the formula $I_R = V_{DD}/2R_{EXT}$ (mA) with $R_{EXT}$ in kΩ.

**4:** FVR and BOR are disabled.

**5:** 8 MHz crystal/oscillator with 4x PLL enabled.

**TABLE 36-2: SUPPLY CURRENT ($I_{DD}$)(1,2) (CONTINUED)**

| PIC16LF1773/6 | | Standard Operating Conditions (unless otherwise stated) | | | | | |
|---|---|---|---|---|---|---|---|
| PIC16F1773/6 | | | | | | | |
| Param No. | Device Characteristics | Min. | Typ† | Max. | Units | Conditions | |
| | | | | | | $V_{DD}$ | Note |
| D017 | | — | 130 | 190 | µA | 1.8 | $F_{OSC}$ = 500 kHz, MFINTOSC mode |
| | | — | 150 | 320 | µA | 3.0 | |
| D017 | | — | 150 | 210 | µA | 2.3 | $F_{OSC}$ = 500 kHz, MFINTOSC mode |
| | | — | 170 | 340 | µA | 3.0 | |
| | | — | 220 | 420 | µA | 5.0 | |
| D019 | | — | 0.8 | 1.5 | mA | 1.8 | $F_{OSC}$ = 16 MHz, HFINTOSC mode |
| | | — | 1.2 | 2.3 | mA | 3.0 | |
| D019 | | — | 1.0 | 1.8 | mA | 2.3 | $F_{OSC}$ = 16 MHz, HFINTOSC mode |
| | | — | 1.3 | 2.3 | mA | 3.0 | |
| | | — | 1.4 | 2.5 | mA | 5.0 | |
| D020 | | — | 2.1 | 4.2 | mA | 3.0 | $F_{OSC}$ = 32 MHz, HFINTOSC mode |
| | | — | 2.5 | 5.0 | mA | 3.6 | |
| D020 | | — | 2.1 | 4.2 | mA | 3.0 | $F_{OSC}$ = 32 MHz, HFINTOSC mode |
| | | — | 2.2 | 4.6 | mA | 5.0 | |
| D022 | | — | 2.1 | 4 | mA | 3.0 | $F_{OSC}$ = 32 MHz, HS Oscillator mode **(Note 5)** |
| | | — | 2.5 | 4.2 | mA | 3.6 | |
| D022 | | — | 2.1 | 4 | mA | 3.0 | $F_{OSC}$ = 32 MHz HS Oscillator mode **(Note 5)** |
| | | — | 2.2 | 4.5 | mA | 5.0 | |

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note 1:** The test conditions for all $I_{DD}$ measurements in active operation mode are: OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to $V_{DD}$; $\overline{MCLR}$ = $V_{DD}$; 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 EXTRC oscillator configurations, current through $R_{EXT}$ is not included. The current through the resistor can be extended by the formula $I_R = V_{DD}/2R_{EXT}$ (mA) with $R_{EXT}$ in kΩ.

**4:** FVR and BOR are disabled.

**5:** 8 MHz crystal/oscillator with 4x PLL enabled.

**TABLE 36-3: POWER-DOWN CURRENTS ($I_{PD}$)(1,2)**

| PIC16LF1773/6 | | Operating Conditions: (unless otherwise stated)<br>Low-Power Sleep Mode | | | | | | |
|---|---|---|---|---|---|---|---|---|
| PIC16F1773/6 | | Low-Power Sleep Mode, VREGPM = 1 | | | | | | |
| Param No. | Device Characteristics | Min. | Typ† | Max. +85°C | Max. +125°C | Units | Conditions | |
| | | | | | | | $V_{DD}$ | Note |
| D023 | Base $I_{PD}$ | — | 0.05 | 1.0 | 8.0 | µA | 1.8 | WDT, BOR, FVR, and SOSC disabled, all Peripherals Inactive |
| | | — | 0.08 | 2.0 | 9.0 | µA | 3.0 | |
| D023 | Base $I_{PD}$ | — | 0.3 | 3 | 11 | µA | 2.3 | WDT, BOR, FVR, and SOSC disabled, all Peripherals Inactive, Low-Power Sleep mode |
| | | — | 0.4 | 4 | 12 | µA | 3.0 | |
| | | — | 0.5 | 6 | 15 | µA | 5.0 | |
| D023A | Base $I_{PD}$ | — | 9.8 | 16 | 18 | µA | 2.3 | WDT, BOR, FVR and SOSC disabled, all Peripherals inactive, Normal Power Sleep mode VREGPM = 0 |
| | | — | 10.3 | 18 | 20 | µA | 3.0 | |
| | | — | 11.5 | 21 | 26 | µA | 5.0 | |
| D024 | | — | 0.5 | 6 | 14 | µA | 1.8 | WDT Current |
| | | — | 0.8 | 7 | 17 | µA | 3.0 | |
| D024 | | — | 0.8 | 6 | 15 | µA | 2.3 | WDT Current |
| | | — | 0.9 | 7 | 20 | µA | 3.0 | |
| | | — | 1.0 | 8 | 22 | µA | 5.0 | |
| D025 | | — | 15 | 28 | 30 | µA | 1.8 | FVR Current (ADC) |
| | | — | 18 | 30 | 33 | µA | 3.0 | |
| D025 | | — | 18 | 33 | 35 | µA | 2.3 | FVR Current (ADC) |
| | | — | 19 | 36 | 38 | µA | 3.0 | |
| | | — | 20 | 39 | 41 | µA | 5.0 | |
| D025A | | — | 25 | 50 | 55 | µA | 1.8 | FVR Current (DAC) |
| | | — | 30 | 65 | 70 | µA | 3.0 | |
| D025A | | — | 30 | 58 | 60 | µA | 2.3 | FVR Current (DAC) |
| | | — | 32 | 70 | 72 | µA | 3.0 | |
| | | — | 35 | 77 | 80 | µA | 5.0 | |
| D026 | | — | 7.5 | 25 | 28 | µA | 3.0 | BOR Current |
| D026 | | — | 10 | 25 | 28 | µA | 3.0 | BOR Current |
| | | — | 12 | 28 | 31 | µA | 5.0 | |
| D027 | | — | 0.5 | 4 | 10 | µA | 3.0 | LPBOR Current |
| D027 | | — | 0.8 | 6 | 14 | µA | 3.0 | LPBOR Current |
| | | — | 1 | 8 | 17 | µA | 5.0 | |
| D028 | | — | 0.5 | 5 | 9 | µA | 1.8 | SOSC Current |
| | | — | 0.8 | 8.5 | 12 | µA | 3.0 | |
| D028 | | — | 1.1 | 6 | 10 | µA | 2.3 | SOSC Current |
| | | — | 1.3 | 8.5 | 20 | µA | 3.0 | |
| | | — | 1.4 | 10 | 25 | 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 $I_{PD}$ and the additional current consumed when this peripheral is enabled. The peripheral Δ current can be determined by subtracting the base $I_{DD}$ or $I_{PD}$ current from this limit. Max values should be used when calculating total current consumption.

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

**3:** ADC clock source is FRC.

**TABLE 36-3: POWER-DOWN CURRENTS ($I_{PD}$)(1,2) (CONTINUED)**

| PIC16LF1773/6 | | Operating Conditions: (unless otherwise stated)<br>Low-Power Sleep Mode | | | | | | |
|---|---|---|---|---|---|---|---|---|
| PIC16F1773/6 | | Low-Power Sleep Mode, VREGPM = 1 | | | | | | |
| Param No. | Device Characteristics | Min. | Typ† | Max. +85°C | Max. +125°C | Units | Conditions | |
| | | | | | | | $V_{DD}$ | Note |
| D029 | | — | 0.05 | 2 | 9 | µA | 1.8 | ADC Current **(Note 3)**, no conversion in progress |
| | | — | 0.08 | 3 | 10 | µA | 3.0 | |
| D029 | | — | 0.3 | 4 | 12 | µA | 2.3 | ADC Current **(Note 3)**, no conversion in progress |
| | | — | 0.4 | 5 | 13 | µA | 3.0 | |
| | | — | 0.5 | 7 | 16 | µA | 5.0 | |
| D030 | | — | 250 | — | — | µA | 1.8 | ADC Current **(Note 3)**, conversion in progress |
| | | — | 280 | — | — | µA | 3.0 | |
| D030 | | — | 230 | — | — | µA | 2.3 | ADC Current **(Note 3)**, conversion in progress |
| | | — | 250 | — | — | µA | 3.0 | |
| | | — | 350 | — | — | µA | 5.0 | |
| D031 | | — | 250 | 650 | — | µA | 3.0 | Op Amp (High power) |
| D031 | | — | 250 | 650 | — | µA | 3.0 | Op Amp (High power) |
| | | — | 350 | 850 | — | µA | 5.0 | |
| D032 | | — | 250 | 600 | — | µA | 1.8 | Comparator |
| | | — | 300 | 650 | — | µA | 3.0 | |
| D032 | | — | 280 | 600 | — | µA | 2.3 | Comparator, VREGPM = 0 |
| | | — | 300 | 650 | — | µA | 3.0 | |
| | | — | 310 | 650 | — | 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 $I_{PD}$ and the additional current consumed when this peripheral is enabled. The peripheral Δ current can be determined by subtracting the base $I_{DD}$ or $I_{PD}$ current from this limit. Max values should be used when calculating total current consumption.

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

**3:** ADC clock source is FRC.

**TABLE 36-4: I/O PORTS**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** |
| | VIL | **Input Low Voltage** | | | | | |
| | | I/O PORT: | | | | | |
| D034 | | with TTL buffer | — | — | 0.8 | V | 4.5V ≤ VDD ≤ 5.5V |
| D034A | | | — | — | 0.15 VDD | V | 1.8V ≤ VDD ≤ 4.5V |
| D035 | | with Schmitt Trigger buffer | — | — | 0.2 VDD | V | 2.0V ≤ VDD ≤ 5.5V |
| | | with $I^2C$ levels | — | — | 0.3 VDD | V | |
| | | with SMBus levels | — | — | 0.8 | V | 2.7V ≤ VDD ≤ 5.5V |
| D036 | | $\overline{MCLR}$, OSC1 (EXTRC mode) | — | — | 0.2 VDD | V | **(Note 1)** |
| D036A | | 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 $I^2C$ 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 (EXTRC oscillator) | 0.9 VDD | — | — | V | VDD > 2.0V**(Note 1)** |
| | IIL | **Input Leakage Current(2)** | | | | | |
| D060 | | I/O Ports | — | ± 5 | ± 125 | nA | VSS ≤ VPIN ≤ VDD, Pin at high-impedance, 85°C |
| | | | — | ± 5 | ± 1000 | nA | VSS ≤ VPIN ≤ VDD, Pin at high-impedance, 125°C |
| D061 | | $\overline{MCLR}$(3) | — | ± 50 | ± 200 | nA | VSS ≤ VPIN ≤ VDD, Pin at high-impedance, 85°C |
| | IPUR | **Weak Pull-up Current** | | | | | |
| D070* | | | 25 | 100 | 200 | µA | VDD = 3.3V, VPIN = VSS |
| | VOL | **Output Low Voltage(4)** | | | | | |
| D080 | | Standard I/O ports | — | — | 0.6 | V | IOL = 8mA, VDD = 5V<br>IOL = 6mA, VDD = 3.3V<br>IOL = 1.8mA, VDD = 1.8V |
| D080A | | High Drive I/O ports | — | — | 0.6 | V | IOH = 10mA, VDD = 2.3V, HIDCx = 1 |
| | | | — | 0.6 | — | V | IOH = 32mA, VDD = 3.0V, HIDCx = 1 |
| | | | — | 0.6 | — | V | IOH = 51mA, VDD = 5.0V, HIDCx = 1 |
| | VOH | **Output High Voltage(4)** | | | | | |
| D090 | | Standard I/O ports | VDD - 0.7 | — | — | V | IOH = 3.5mA, VDD = 5V<br>IOH = 3mA, VDD = 3.3V<br>IOH = 1mA, VDD = 1.8V |
| D090A | | High Drive I/O ports | VDD - 0.7 | — | — | V | IOH = 10mA, VDD = 2.3V, HIDCx = 1 |
| | | | — | VDD - 0.7 | — | V | IOH = 37mA, VDD = 3.0V, HIDCx = 1 |
| | | | — | VDD - 0.7 | — | V | IOH = 54mA, VDD = 5.0V, HIDCx = 1 |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note 1:** In EXTRC oscillator configuration, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended to use an external clock in EXTRC 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.

**TABLE 36-4: I/O PORTS (CONTINUED) (CONTINUED)**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** |
| | | **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 | |

* 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 EXTRC oscillator configuration, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended to use an external clock in EXTRC 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.

**TABLE 36-5: MEMORY PROGRAMMING SPECIFICATIONS**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** |
| | | **Program Memory Programming Specifications** | | | | | |
| D110 | $V_{IHH}$ | Voltage on $\overline{MCLR}/V_{PP}$ pin | 8.0 | — | 9.0 | V | **(Note 2)** |
| D111 | $I_{DDP}$ | Supply Current during Programming | — | — | 10 | mA | |
| D112 | $V_{BE}$ | $V_{DD}$ for Bulk Erase | 2.7 | — | $V_{DDMAX}$ | V | |
| D113 | $V_{PEW}$ | $V_{DD}$ for Write or Row Erase | $V_{DDMIN}$ | — | $V_{DDMAX}$ | V | |
| D114 | $I_{PPPGM}$ | Current on $\overline{MCLR}/V_{PP}$ during Erase/Write | — | 1.0 | — | mA | |
| D115 | $I_{DDPGM}$ | Current on $V_{DD}$ during Erase/Write | — | 5.0 | — | mA | |
| | | **Program Flash Memory** | | | | | |
| D121 | $E_P$ | Cell Endurance | 10K | — | — | E/W | -40°C ≤ $T_A$ ≤ +85°C **(Note 1)** |
| D122 | $V_{PRW}$ | $V_{DD}$ for Read/Write | $V_{DDMIN}$ | — | $V_{DDMAX}$ | V | |
| D123 | $T_{IW}$ | Self-timed Write Cycle Time | — | 2 | 2.5 | ms | |
| D124 | $T_{RETD}$ | Characteristic Retention | — | 40 | — | Year | Provided no other specifications are violated |
| D125 | $E_{HEFC}$ | High-Endurance Flash Cell | 100K | — | — | E/W | -0°C ≤ $T_A$ ≤ +60°C, Lower byte last 128 addresses |

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note 1:** Self-write and Block Erase.

**2:** Required only if single-supply programming is disabled.

**TABLE 36-6: THERMAL CHARACTERISTICS**

| Standard Operating Conditions (unless otherwise stated) | | | | | |
|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Typ.** | **Units** | **Conditions** |
| TH01 | $\theta_{JA}$ | Thermal Resistance Junction to Ambient | 60 | °C/W | 28-pin SPDIP package |
| | | | 80 | °C/W | 28-pin SOIC package |
| | | | 90 | °C/W | 28-pin SSOP package |
| | | | 48 | °C/W | 28-pin UQFN 4x4 mm package |
| TH02 | $\theta_{JC}$ | Thermal Resistance Junction to Case | 31.4 | °C/W | 28-pin SPDIP package |
| | | | 24 | °C/W | 28-pin SOIC package |
| | | | 24 | °C/W | 28-pin SSOP package |
| | | | 12 | °C/W | 28-pin UQFN 4x4 mm package |
| TH03 | $T_{JMAX}$ | Maximum Junction Temperature | 150 | °C | |
| TH04 | PD | Power Dissipation | — | W | $PD = P_{INTERNAL} + P_{I/O}$ |
| TH05 | $P_{INTERNAL}$ | Internal Power Dissipation | — | W | $P_{INTERNAL} = I_{DD} \times V_{DD}$ (1) |
| TH06 | $P_{I/O}$ | I/O Power Dissipation | — | W | $P_{I/O} = \Sigma (I_{OL} * V_{OL}) + \Sigma (I_{OH} * (V_{DD} - V_{OH}))$ |
| TH07 | $P_{DER}$ | Derated Power | — | W | $P_{DER} = PD_{MAX} (T_J - T_A)/\theta_{JA}$ (2) |

**Note 1:** $I_{DD}$ is current to run the chip alone without driving any load on the output pins.

**2:** $T_A$ = Ambient Temperature, $T_J$ = Junction Temperature

## 36.4 AC Characteristics

Timing Parameter Symbology has been created with one of the following formats:

1. TppS2ppS
2. TppS

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

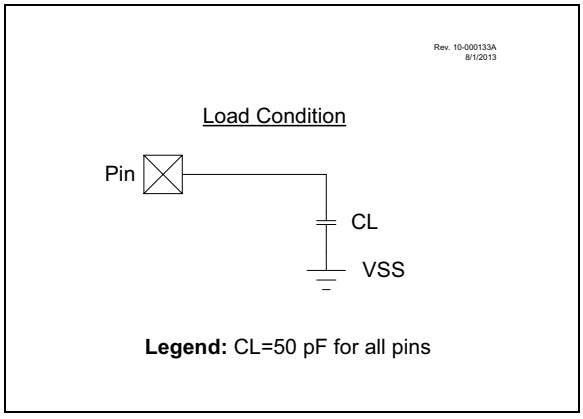
Lowercase letters (pp) and their meanings:

| | | | |
|---|---|---|---|
| **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:

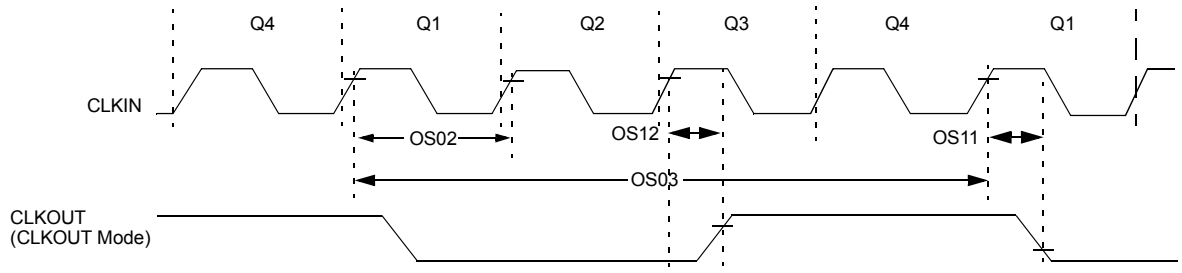
| | | | |
|---|---|---|---|
| **S** | | | |
| F | Fall | P | Period |
| H | High | R | Rise |
| I | Invalid (High-impedance) | V | Valid |
| L | Low | Z | High-impedance |

**FIGURE 36-4: LOAD CONDITIONS**




**Legend:** CL=50 pF for all pins

**FIGURE 36-5: CLOCK TIMING**




**Note 1:** See Table 36-10.

**TABLE 36-7: CLOCK OSCILLATOR TIMING REQUIREMENTS**

**Standard Operating Conditions (unless otherwise stated)**

| Param No. | Sym. | Characteristic | Min. | Typ† | Max. | Units | Conditions |
|---|---|---|---|---|---|---|---|
| OS01 | FOSC | External CLKIN Frequency(1) | DC | — | 0.5 | MHz | External Clock (ECL) |
| | | | DC | — | 4 | MHz | External Clock (ECM) |
| | | | DC | — | 32 | MHz | External Clock (ECH) |
| | | Oscillator Frequency(1) | — | 32.768 | — | kHz | LP Oscillator |
| | | | 0.1 | — | 4 | MHz | XT Oscillator |
| | | | 1 | — | 4 | MHz | HS Oscillator |
| | | | 1 | — | 20 | MHz | HS Oscillator, VDD > 2.7V |
| | | | DC | — | 4 | MHz | EXTRC, VDD > 2.0V |
| OS02 | TOSC | External CLKIN Period(1) | 27 | — | ∞ | µs | LP Oscillator |
| | | | 250 | — | ∞ | ns | XT Oscillator |
| | | | 50 | — | ∞ | ns | HS Oscillator |
| | | | 31.25 | — | ∞ | ns | External Clock (EC) |
| | | Oscillator Period(1) | — | 30.5 | — | µs | LP Oscillator |
| | | | 250 | — | 10,000 | ns | XT Oscillator |
| | | | 50 | — | 1,000 | ns | HS Oscillator |
| | | | 250 | — | — | ns | EXTRC |
| OS03 | TCY | Instruction Cycle Time(1) | 125 | 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 |
| 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 36-8: OSCILLATOR PARAMETERS**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Param No. | Sym. | Characteristic | Freq. Tolerance | Min. | Typ† | Max. | Units | Conditions |
| OS08 | HFOSC | Internal Calibrated HFINTOSC Frequency(1) | ±2% | — | 16.0 | — | MHz | VDD = 3.0V, TA = 25°C, **(Note 2)** |
| OS08A | MFOSC | Internal Calibrated MFINTOSC Frequency(1) | ±2% | — | 500 | — | kHz | VDD = 3.0V, TA = 25°C, **(Note 2)** |
| OS09 | LFOSC | Internal LFINTOSC Frequency | — | — | 31 | — | kHz | -40°C ≤ TA ≤ +125°C |
| OS10* | TWARM | HFINTOSC Wake-up from Sleep Start-up Time | — | — | 3.2 | 8 | µs | |
| | | MFINTOSC Wake-up from Sleep Start-up Time | — | — | 24 | 35 | µs | |
| | TLFOSC ST | LFINTOSC Wake-up from Sleep Start-up Time | — | — | 0.5 | — | ms | |

* 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.

**2:** See Figure 36-6: HFINTOSC Frequency Accuracy Over Device VDD and Temperature, Figure 37-75: Wake From Sleep, VREGPM = 0., and Figure 36-6: HFINTOSC Frequency Accuracy Over Device VDD and Temperature.

**3:** See Figure 37-58: LFINTOSC Frequency, PIC16LF1773/6 Only., and Figure 37-59: LFINTOSC Frequency, PIC16F1773/6 Only..

**FIGURE 36-6: HFINTOSC FREQUENCY ACCURACY OVER DEVICE VDD AND TEMPERATURE**

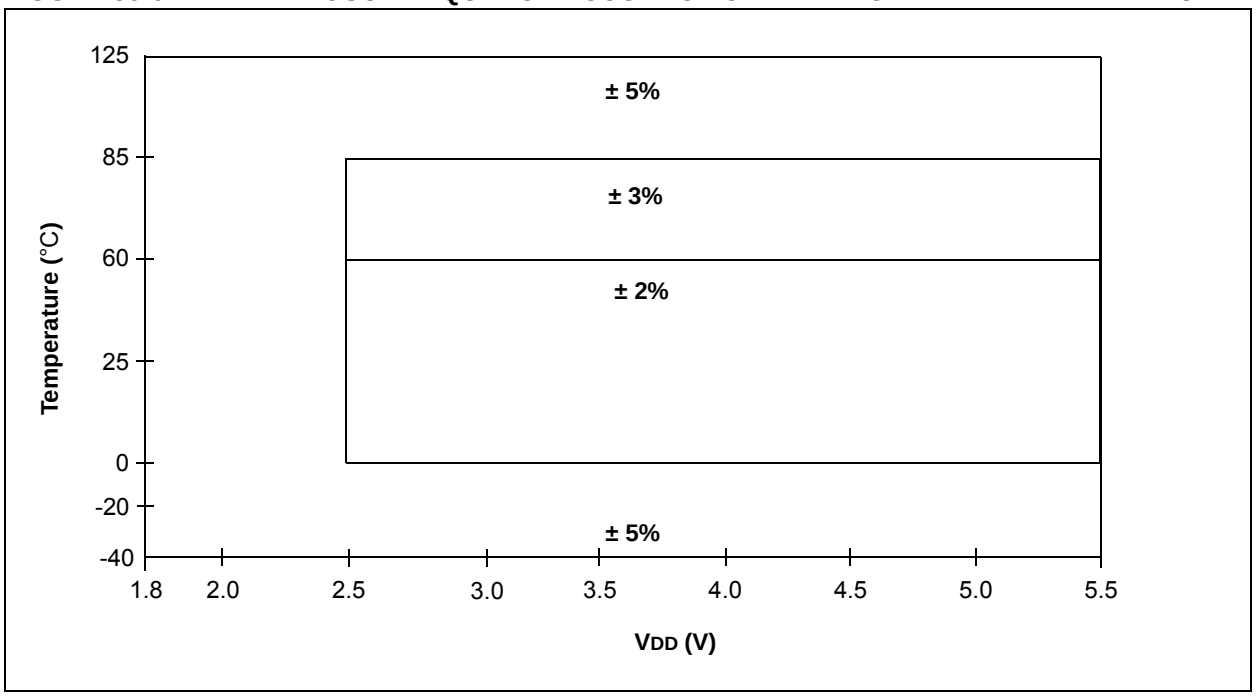


**TABLE 36-9: PLL CLOCK TIMING SPECIFICATIONS**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **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 36-7: CLKOUT AND I/O TIMING**

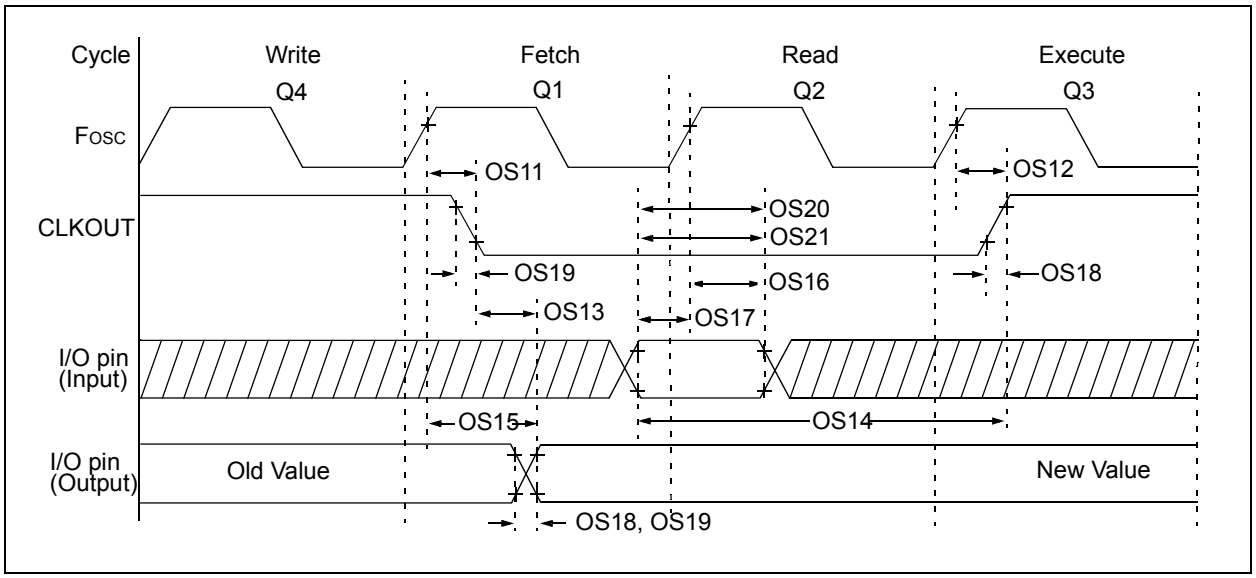



**TABLE 36-10: CLKOUT AND I/O TIMING PARAMETERS**

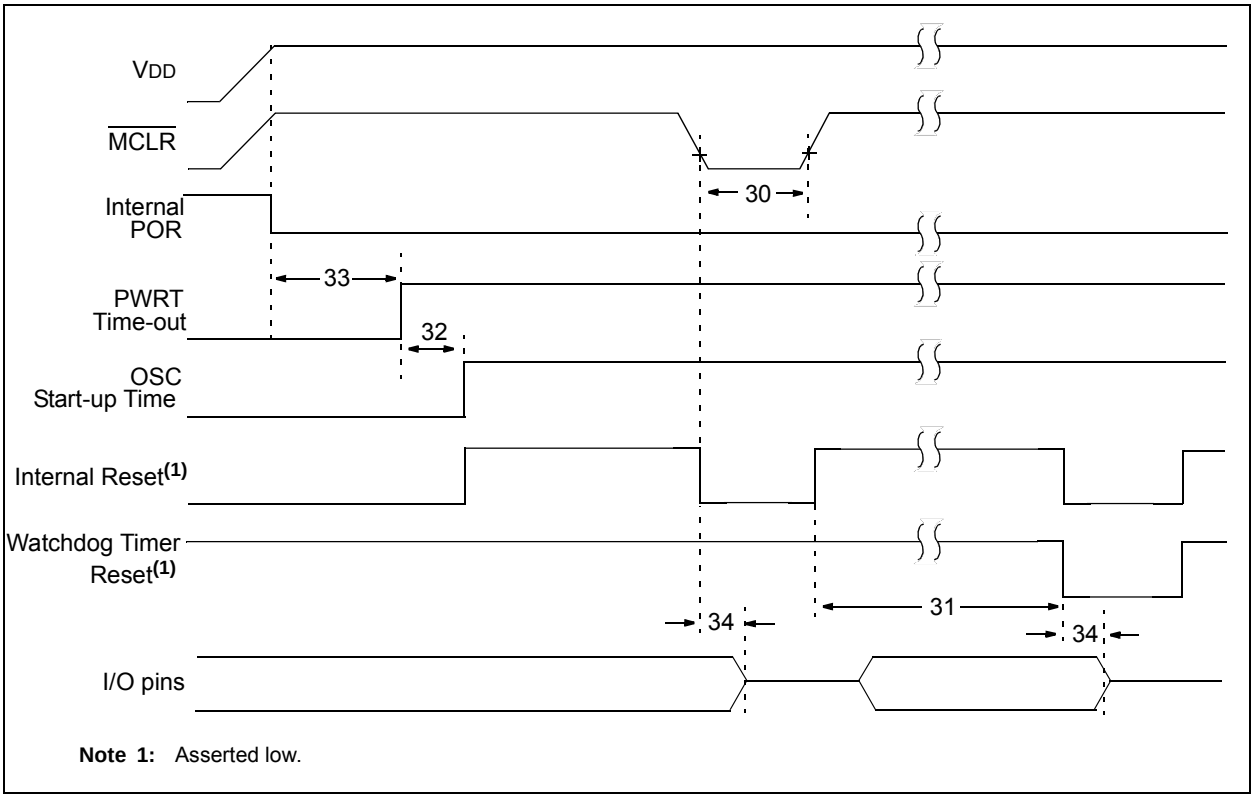
| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** |
| OS11 | TosH2ckL | FOSC↑ to CLKOUT↓ (1) | — | — | 70 | ns | 3.3V ≤ VDD ≤ 5.0V |
| OS12 | TosH2ckH | FOSC↑ to CLKOUT↑ (1) | — | — | 72 | ns | 3.3V ≤ VDD ≤ 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 | 3.3V ≤ VDD ≤ 5.0V |
| OS16 | TosH2ioI | Fosc↑ (Q2 cycle) to Port input invalid (I/O in hold time) | 50 | — | — | ns | 3.3V ≤ VDD ≤ 5.0V |
| OS17 | TioV2osH | Port input valid to Fosc↑ (Q2 cycle) (I/O in setup time) | 20 | — | — | ns | |
| OS18* | TioR | Port output rise time | —<br>— | 40<br>15 | 72<br>32 | ns | VDD = 1.8V<br>3.3V ≤ VDD ≤ 5.0V |
| OS19* | TioF | Port output fall time | —<br>— | 28<br>15 | 55<br>30 | ns | VDD = 1.8V<br>3.3V ≤ VDD ≤ 5.0V |
| OS20* | Tinp | INT pin input high or low time | 25 | — | — | ns | |
| OS21* | Tioc | Interrupt-on-change new input level time | 25 | — | — | ns | |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated.

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

**FIGURE 36-8: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING**




**Note 1:** Asserted low.

**TABLE 36-11: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER AND BROWN-OUT RESET PARAMETERS**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** |
| 30 | TMCL | $\overline{MCLR}$ Pulse Width (low) | 2 | — | — | µs | |
| 31 | TWDTLP | Low-Power 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, $\overline{PWRTE}$ = 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><br>2.30<br>1.80 | 2.70<br><br>2.45<br>1.90 | 2.85<br><br>2.60<br>2.10 | V<br><br>V<br>V | BORV = 0<br><br>BORV = 1 (PIC16F1773/6)<br>BORV = 1 (PIC16LF1773/6) |
| 35A | VLPBOR | Low-Power Brown-out | 1.8 | 2.1 | 2.5 | V | LPBOR = 1 |
| 36* | VHYST | Brown-out Reset Hysteresis | 0 | 25 | 75 | mV | -40°C ≤ TA ≤ +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 36-9: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS**

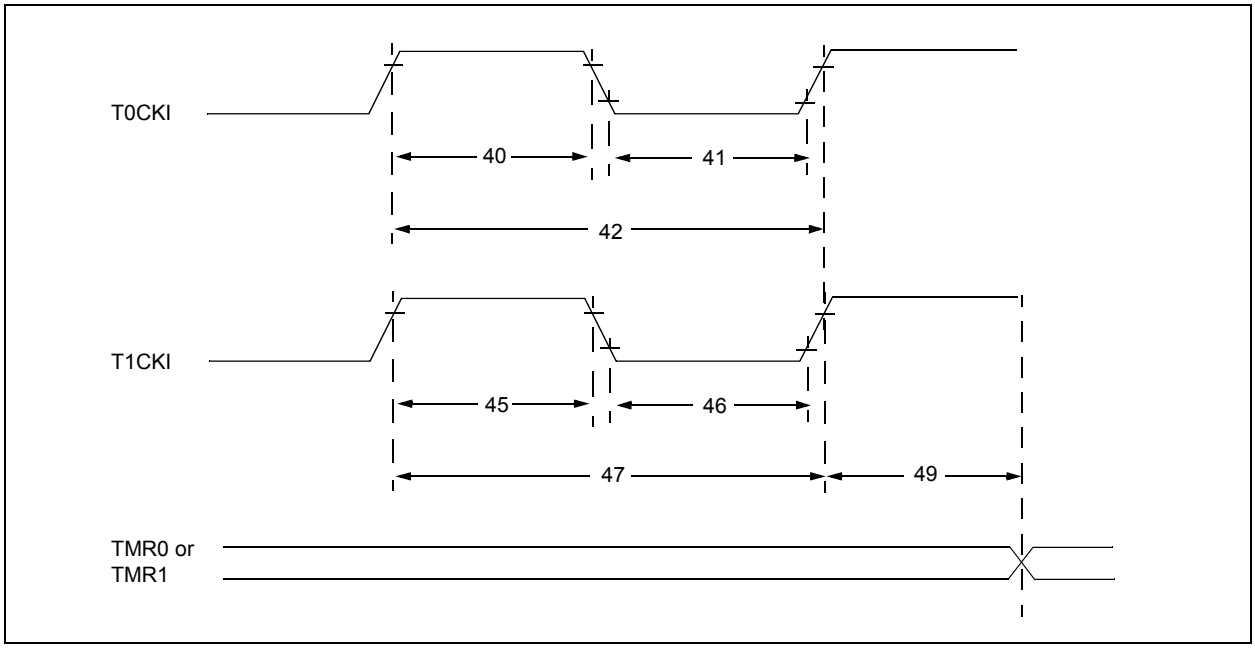
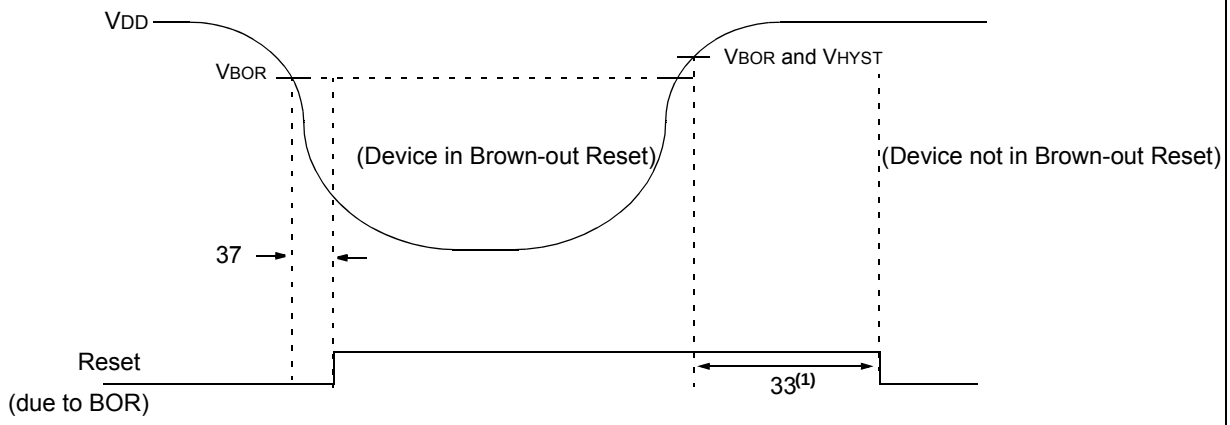



**FIGURE 36-10: BROWN-OUT RESET TIMING AND CHARACTERISTICS**




**Note 1:** The delay, ($T_{PWRT}$) releasing Reset, only occurs when the Power-up Timer is enabled, ($\overline{PWRTE}$ = 0).

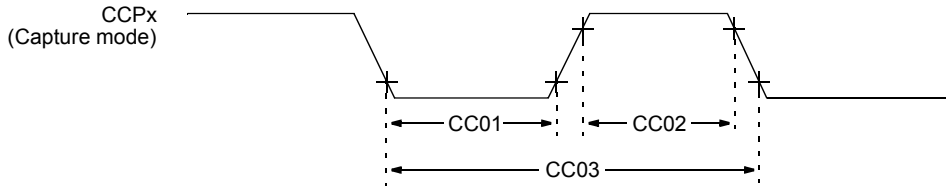
**TABLE 36-12: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS**

| **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 |
| 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 |
| | | | Asynchronous | 60 | — | — | ns | |
| 48 | FT1 | Secondary 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 36-11: CAPTURE/COMPARE/PWM TIMINGS (CCP)**




**Note:** Refer to Figure 36-4 for load conditions.

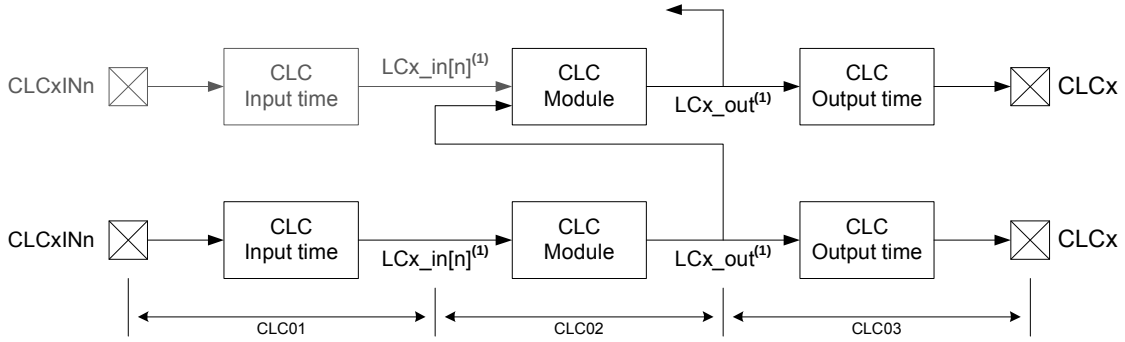
**TABLE 36-13: CAPTURE/COMPARE/PWM REQUIREMENTS (CCP)**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **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 |

* 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 36-12: CLC PROPAGATION TIMING**




**Note 1:** See Figure 28-1 to identify specific CLC signals.

**TABLE 36-14: CONFIGURATION LOGIC CELL (CLC) CHARACTERISTICS**

| **Standard Operating Conditions (unless otherwise stated)**<br>Operating temperature -40°C ≤ TA ≤ +125°C | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param. No.** | **Sym.** | **Characteristic** | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** |
| CLC01* | TCLCIN | CLC input time | — | 7 | OS17 | ns | **(Note 1)** |
| CLC02* | TCLC | CLC module input to output progagation time | —<br>— | 24<br>12 | —<br>— | ns<br>ns | VDD = 1.8V<br>VDD > 3.6V |
| CLC03* | TCLCOUT | CLC output time Rise Time | — | OS18 | — | — | **(Note 1)** |
| | | Fall Time | — | OS19 | — | — | **(Note 1)** |
| CLC04* | FCLCMAX | CLC maximum switching frequency | — | 45 | — | MHz | |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note 1:** See Table 36-10 for OS17, OS18 and OS19 rise and fall times.

**TABLE 36-15: ANALOG-TO-DIGITAL CONVERTER (ADC) CHARACTERISTICS(1,2,3,4):**

| Operating Conditions (unless otherwise stated)<br>VDD = 3.0V, TA = 25°C, Single-ended, 2 μs TAD, VREF+ = 3V, VREF- = VSS | | | | | | | |
|---|---|---|---|---|---|---|---|
| **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, 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 | 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 0.01μF capacitor is present on input pin. |

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note 1:** Total Absolute Error includes integral, differential, offset and gain errors.

**2:** The ADC conversion result never decreases with an increase in the input voltage and has no missing codes.

**3:** ADC VREF is from external VREF+ pin, VDD pin or FVR, whichever is selected as reference input.

**4:** See **Section 37.0 "DC and AC Characteristics Graphs and Charts"** for operating characterization.

**TABLE 36-16: ADC CONVERSION REQUIREMENTS**

| **Standard Operating Conditions** (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristic** | **Min.** | **Typ†** | **Max.** | **Units** | **Conditions** |
| AD130* | TAD | ADC Clock Period (TADC) | 1.0 | — | 9.0 | μs | FOSC-based |
| | | ADC Internal FRC Oscillator Period (TFRC) | 1.0 | 2.5 | 6.0 | μs | ADCS<1:0> = 11 (ADC FRC mode) |
| AD131 | TCNV | Conversion Time (not including Acquisition Time)(1) | — | 13 | — | TAD | Set GO/$\overline{\text{DONE}}$ bit to conversion complete |
| AD132* | TACQ | Acquisition Time | — | 5.0 | — | μs | |
| AD133* | THCD | Holding Capacitor Disconnect Time | — | 1/2 TAD | — | | ADCS<2:0> ≠ x11 (FOSC-based) |
| | | | — | 1/2 TAD + 1TCY | — | | ADCS<2:0> = x11 (FRC-based) |

* 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 36-13: ADC CONVERSION TIMING (ADC CLOCK FOSC-BASED)**

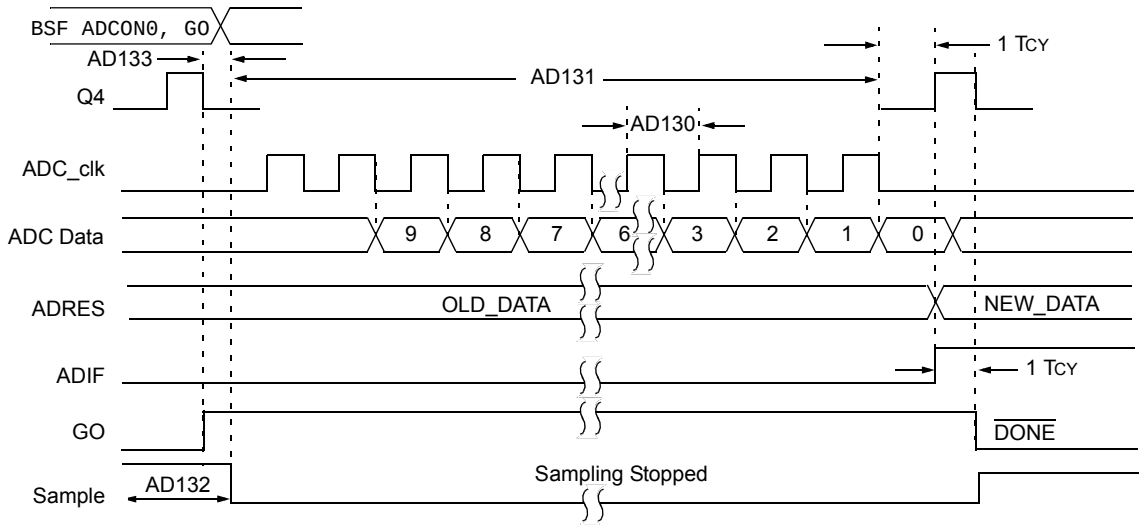
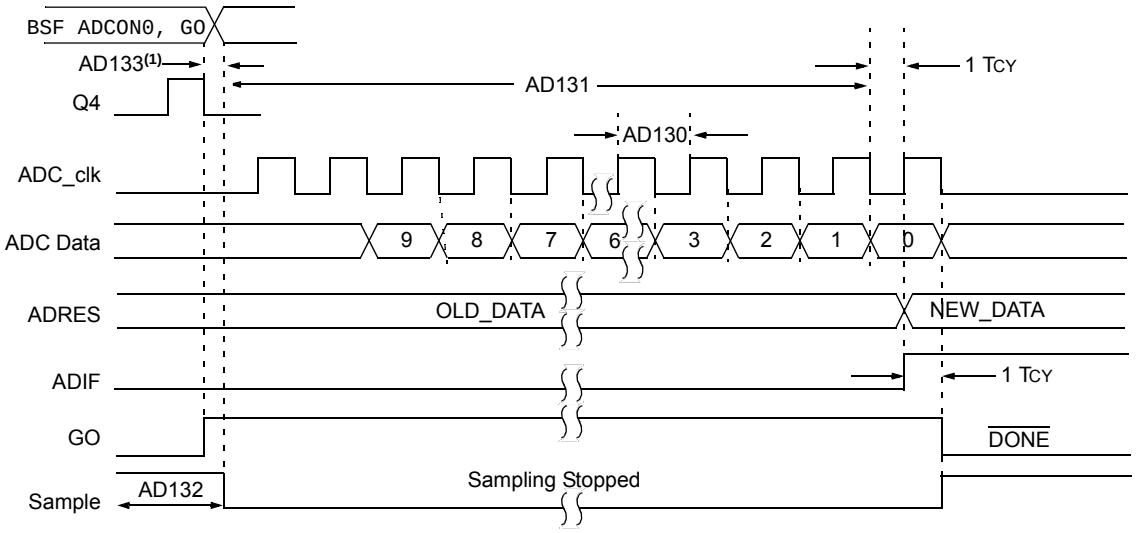



**FIGURE 36-14: ADC CONVERSION TIMING (ADC CLOCK FROM FRC)**




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

**TABLE 36-17: OPERATIONAL AMPLIFIER (OPA)**

| **Operating Conditions (unless otherwise stated)**<br>VDD = 3.0V, TA = 25°C, OPAxSP = 1 (High GBWP mode) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Symbol** | **Parameters** | **Min.** | **Typ.** | **Max.** | **Units** | **Conditions** |
| OPA01* | GBWP | Gain Bandwidth Product | — | 3.5 | — | MHz | |
| OPA02* | TON | Turn-on Time | — | 10 | — | µs | |
| OPA03* | PM | Phase Margin | — | 40 | — | degrees | |
| OPA04* | SR | Slew Rate | — | 3 | — | V/µs | |
| OPA05 | OFF | Offset | — | ±3 | ±9 | mV | |
| OPA06 | CMRR | Common-Mode Rejection Ratio | 52 | 70 | — | dB | |
| OPA07* | AOL | Open Loop Gain | — | 90 | — | dB | |
| OPA08 | VICM | Input Common-Mode Voltage | 0 | — | VDD | V | VDD > 2.5V |
| OPA09* | PSRR | Power Supply Rejection Ratio | — | 80 | — | dB | |
| OPA10* | HZ | High-Impedance On/Off Time | — | 50 | — | ns | |

* These parameters are characterized but not tested.

**TABLE 36-18: PROGRAMMABLE RAMP GENERATOR (PRG) SPECIFICATIONS**

| **Operating Conditions (unless otherwise stated)**<br>VDD = 3.0V, TA = 25°C (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristics** | **Min.** | **Typ.** | **Max.** | **Units** | **Comments** |
| PRG01 | RRR | Rising Ramp Rate(1) | — | 1 | — | V/µs | PRGxCON2 = 10h |
| PRG02 | FRR | Falling Ramp Rate(1) | — | 1 | — | V/µs | PRGxCON2 = 10h |

* These parameters are characterized but not tested.

**Note 1:** Response time measured with one comparator input at VDD/2, while the other input transitions from VSS to VDD.

**TABLE 36-19: COMPARATOR SPECIFICATIONS**

| **Operating Conditions (unless otherwise stated)**<br>VDD = 3.0V, TA = 25°C<br>See **Section 37.0 "DC and AC Characteristics Graphs and Charts"** for operating characterization. | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Param No.** | **Sym.** | **Characteristics** | **Min.** | **Typ.** | **Max.** | **Units** | **Comments** |
| CM01 | VIOFF | Input Offset Voltage | — | ±2.5 | ±5 | mV | VICM = VDD/2 |
| CM02 | VICM | Input Common-Mode Voltage | 0 | — | VDD | V | |
| CM03 | CMRR | Common-Mode Rejection Ratio | 40 | 50 | — | dB | |
| CM04A | TRESP(1) | Response Time Rising Edge | — | 60 | 85 | ns | |
| CM04B | | Response Time Falling Edge | — | 60 | 90 | ns | |
| CM05* | TMC2OV | Comparator Mode Change to Output Valid* | — | — | 10 | µs | |
| CM06 | CHYSTER | Comparator Hysteresis | 20 | 45 | 75 | mV | CxHYS = 1 |

* These parameters are characterized but not tested.

**Note 1:** Response time measured with one comparator input at VDD/2, while the other input transitions from VSS to VDD.

### TABLE 36-20: 10-BIT DIGITAL-TO-ANALOG CONVERTER (DAC) SPECIFICATIONS

**Operating Conditions (unless otherwise stated)**
$V_{DD}$ = 3.0V, $T_A$ = 25°C
See **Section 37.0 "DC and AC Characteristics Graphs and Charts"** for operating characterization.

| Param No. | Sym. | Characteristics | Min. | Typ. | Max. | Units | Comments |
|---|---|---|---|---|---|---|---|
| DAC01* | $C_{LSB}$ | Step Size | — | $V_{DD}$/1024 | — | V | |
| DAC02 | $C_{INL}$ | Integral Error(2) | — | — | ± 2 | LSb | For codes 0x004 to 0x3FB |
| DAC03 | $C_{DNL}$ | Differential Error(2) | — | — | ± 1 | LSb | |
| DAC04 | $C_{OFF}$ | Offset Error(2) | — | — | ± 3 | LSb | |
| DAC05 | $C_{GN}$ | Gain Error(2) | — | — | ± 3 | LSb | |
| DAC06* | $C_R$ | Unit Resistor Value (R) | — | 300 | — | Ω | |
| DAC07* | $C_{ST}$ | Settling Time(1) | — | — | 10 | µs | |

* These parameters are characterized but not tested.

**Note 1:** Settling time measured while DACR<9:0> transitions from '0x000' to '0x3FF'.

**2:** Buffered by op amp in unity gain.

### TABLE 36-21: 5-BIT DIGITAL-TO-ANALOG CONVERTER (DAC) SPECIFICATIONS

**Operating Conditions (unless otherwise stated)**
$V_{DD}$ = 3.0V, $T_A$ = 25°C
See **Section 37.0 "DC and AC Characteristics Graphs and Charts"** for operating characterization.

| Param No. | Sym. | Characteristics | Min. | Typ. | Max. | Units | Comments |
|---|---|---|---|---|---|---|---|
| DAC10* | $C_{LSB}$ | Step Size | — | $V_{DD}$/32 | — | V | |
| DAC11 | $C_{ACC}$ | Absolute Accuracy(2) | — | — | ± 0.5 | LSb | |
| DAC12* | $C_R$ | Unit Resistor Value (R) | — | 6000 | — | Ω | |
| DAC13* | $C_{ST}$ | Settling Time(1) | — | — | 10 | µs | |

* These parameters are characterized but not tested.

**Note 1:** Settling time measured while DACR<4:0> transitions from '0x00' to '0x1F'.

**2:** Buffered by op amp in unity gain.

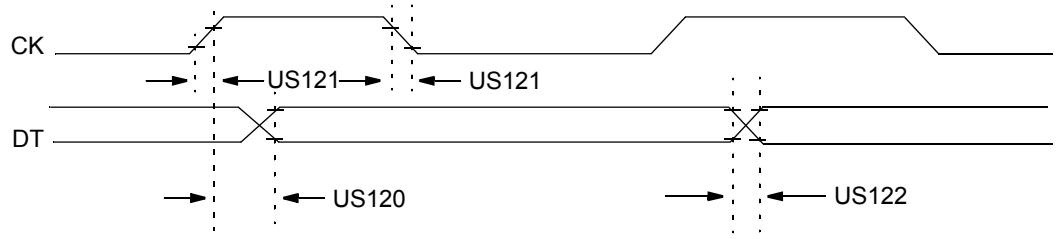
### TABLE 36-22: ZERO CROSS PIN SPECIFICATIONS

**Operating Conditions (unless otherwise stated)**
$V_{DD}$ = 3.0V, $T_A$ = 25°C

| Param. No. | Sym. | Characteristics | Min. | Typ. | Max. | Units | Comments |
|---|---|---|---|---|---|---|---|
| ZC01 | $Z_{CPINV}$ | Voltage on Zero Cross Pin | — | 0.75 | — | V | |
| ZC02 | $Z_{CDRV}$ | Maximum source or sink current | — | — | 600 | µA | |
| ZC03 | $Z_{CSNK}$ | Sink current | 600 | — | — | µA | |
| ZC04 | $Z_{CISW}$ | Response Time Rising Edge | — | 1 | — | µs | |
| | | Response Time Falling Edge | — | 1 | — | µs | |
| ZC05 | $Z_{COUT}$ | Response Time Rising Edge | — | 1 | — | µs | |
| | | Response Time Falling Edge | — | 1 | — | µs | |

* These parameters are characterized but not tested.

**FIGURE 36-15: EUSART SYNCHRONOUS TRANSMISSION (MASTER/SLAVE) TIMING**

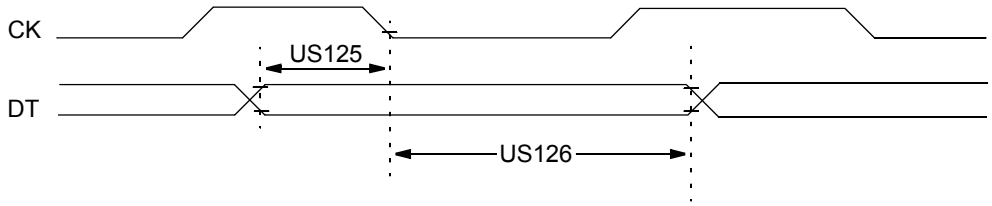



**Note:** Refer to Figure 36-4 for load conditions.

**TABLE 36-23: EUSART SYNCHRONOUS TRANSMISSION REQUIREMENTS**

| Standard Operating Conditions (unless otherwise stated) | | | | | | |
|---|---|---|---|---|---|---|
| **Param. No.** | **Symbol** | **Characteristic** | **Min.** | **Max.** | **Units** | **Conditions** |
| US120 | TCKH2DTV | SYNC XMIT (Master and Slave) Clock high to data-out valid | — | 80 | ns | 3.0V ≤ VDD ≤ 5.5V |
| | | | — | 100 | ns | 1.8V ≤ VDD ≤ 5.5V |
| US121 | TCKRF | Clock out rise time and fall time (Master mode) | — | 45 | ns | 3.0V ≤ VDD ≤ 5.5V |
| | | | — | 50 | ns | 1.8V ≤ VDD ≤ 5.5V |
| US122 | TDTRF | Data-out rise time and fall time | — | 45 | ns | 3.0V ≤ VDD ≤ 5.5V |
| | | | — | 50 | ns | 1.8V ≤ VDD ≤ 5.5V |

**FIGURE 36-16: EUSART SYNCHRONOUS RECEIVE (MASTER/SLAVE) TIMING**

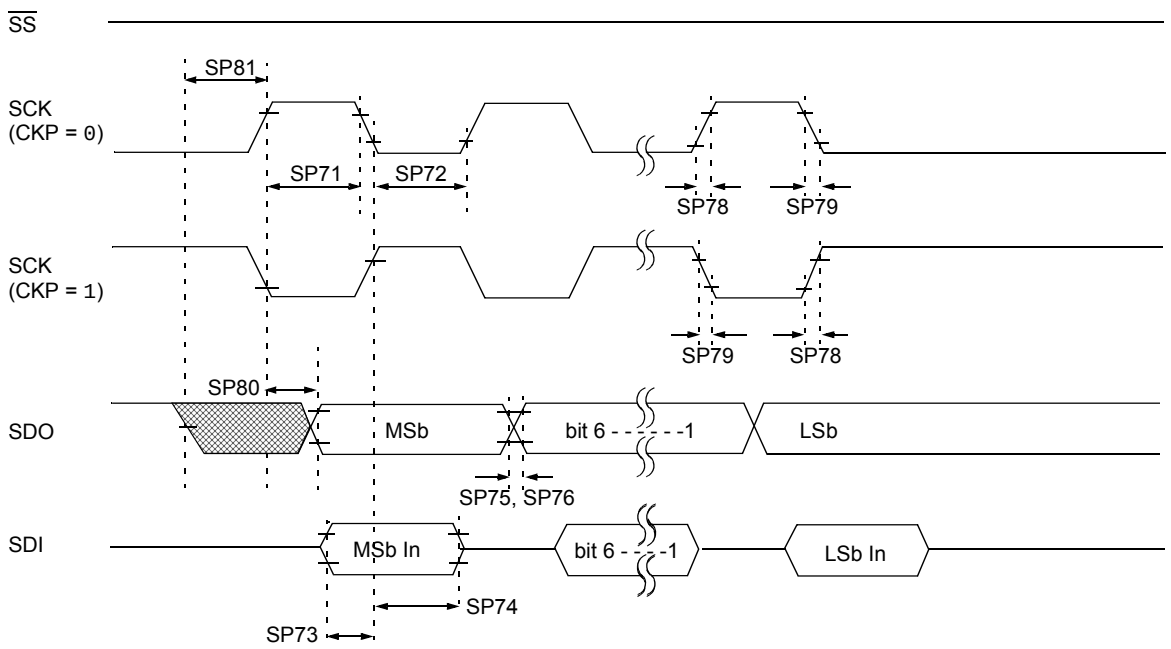



**Note:** Refer to Figure 36-4 for load conditions.

**TABLE 36-24: EUSART SYNCHRONOUS RECEIVE REQUIREMENTS**

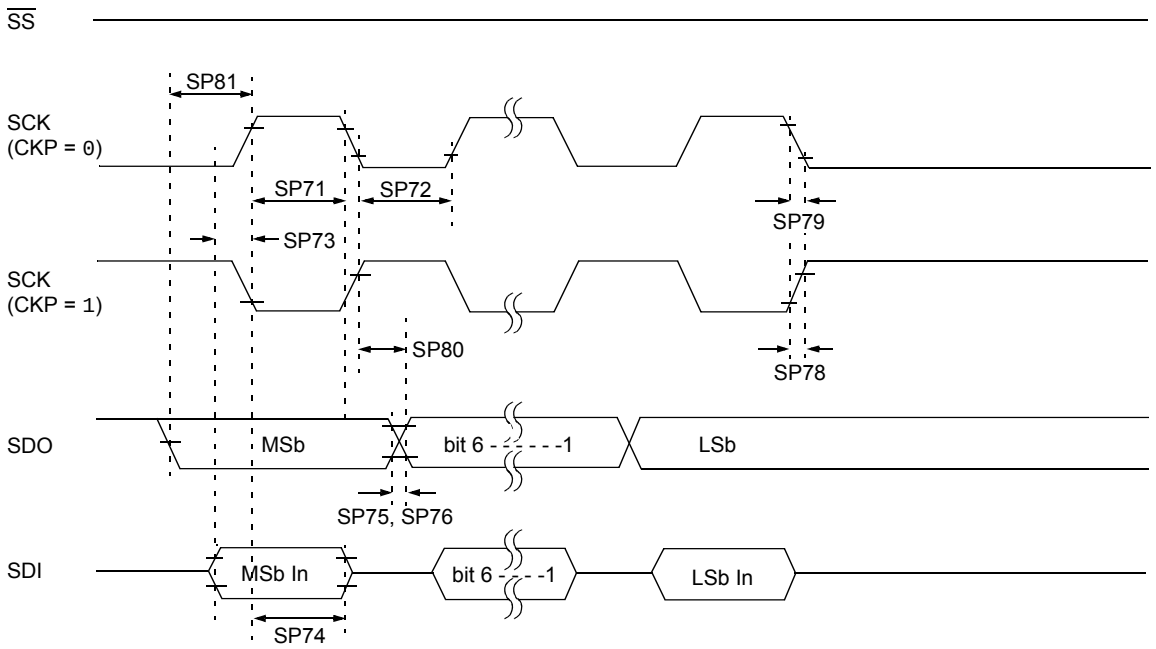
| Standard Operating Conditions (unless otherwise stated) | | | | | | |
|---|---|---|---|---|---|---|
| **Param. No.** | **Symbol** | **Characteristic** | **Min.** | **Max.** | **Units** | **Conditions** |
| US125 | TDTV2CKL | SYNC RCV (Master and Slave) Data-setup before CK ↓ (DT hold time) | 10 | — | ns | |
| US126 | TCKL2DTL | Data-hold after CK ↓ (DT hold time) | 15 | — | ns | |

**FIGURE 36-17: SPI MASTER MODE TIMING (CKE = 0, SMP = 0)**



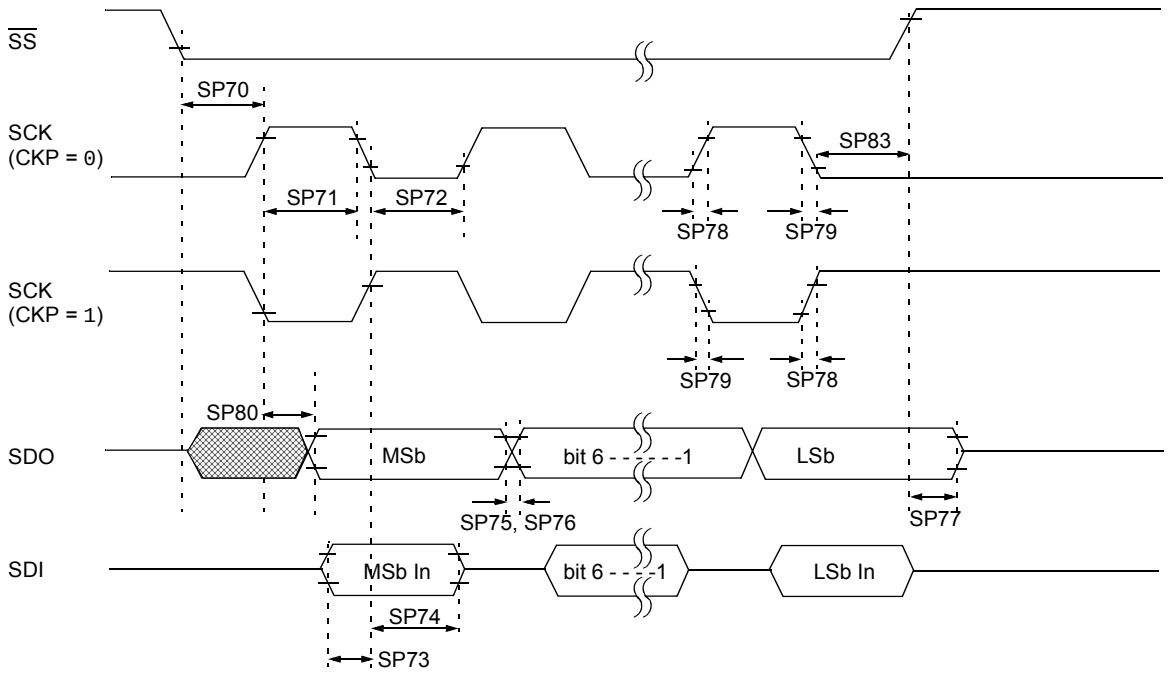

**Note:** Refer to Figure 36-4 for load conditions.

**FIGURE 36-18: SPI MASTER MODE TIMING (CKE = 1, SMP = 1)**



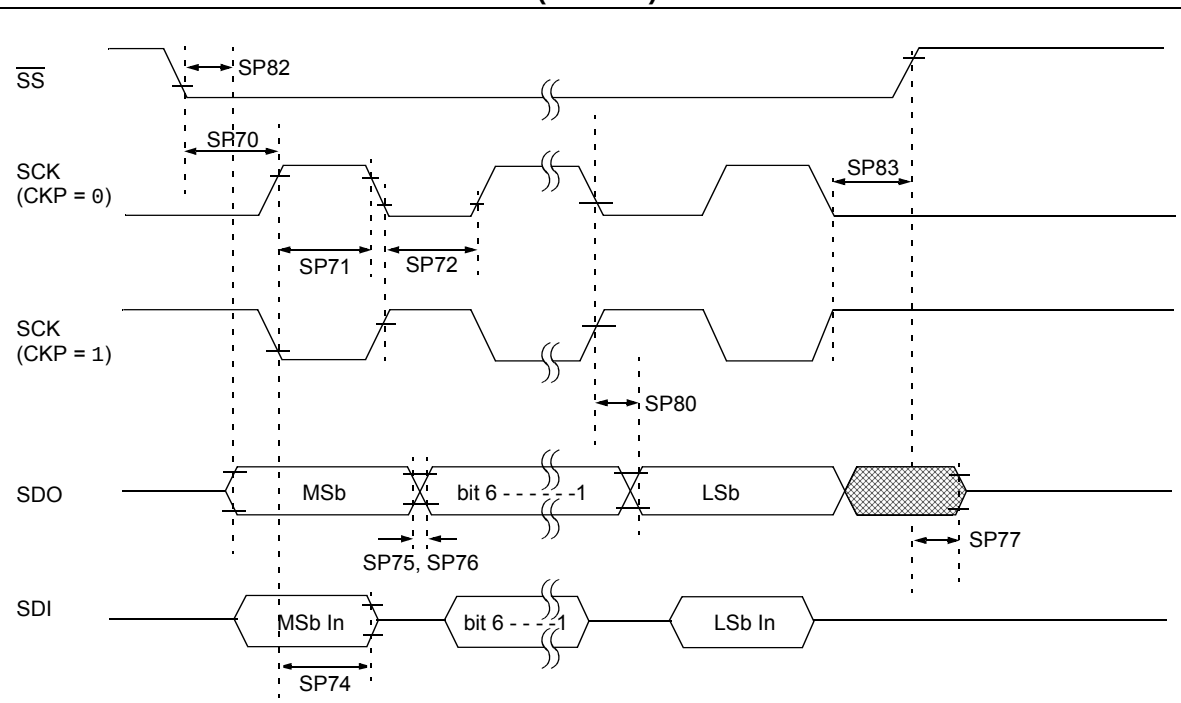

**Note:** Refer to Figure 36-4 for load conditions.

**FIGURE 36-19: SPI SLAVE MODE TIMING (CKE = 0)**




**Note:** Refer to Figure 36-4 for load conditions.

**FIGURE 36-20: SPI SLAVE MODE TIMING (CKE = 1)**




**Note:** Refer to Figure 36-4 for load conditions.

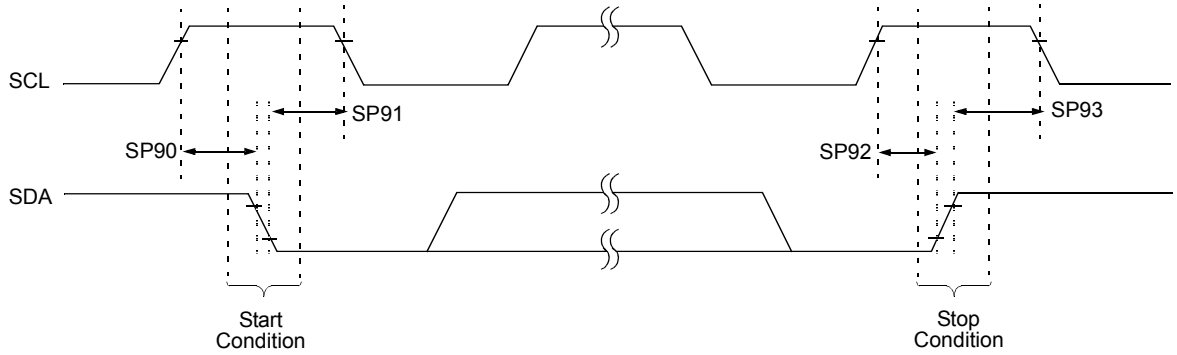
**TABLE 36-25: SPI MODE REQUIREMENTS**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **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 | — | 10 | 25 | ns | 3.0V ≤ VDD ≤ 5.5V |
| | | | — | 25 | 50 | ns | 1.8V ≤ VDD ≤ 5.5V |
| 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) | — | 10 | 25 | ns | 3.0V ≤ VDD ≤ 5.5V |
| | | | — | 25 | 50 | ns | 1.8V ≤ VDD ≤ 5.5V |
| SP79* | TscF | SCK output fall time (Master mode) | — | 10 | 25 | ns | |
| SP80* | TscH2doV, TscL2doV | SDO data output valid after SCK edge | — | — | 50 | ns | 3.0V ≤ VDD ≤ 5.5V |
| | | | — | — | 145 | ns | 1.8V ≤ VDD ≤ 5.5V |
| SP81* | TdoV2scH, TdoV2scL | SDO data output setup to SCK edge | 1 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 36-21: I²C BUS START/STOP BITS TIMING**



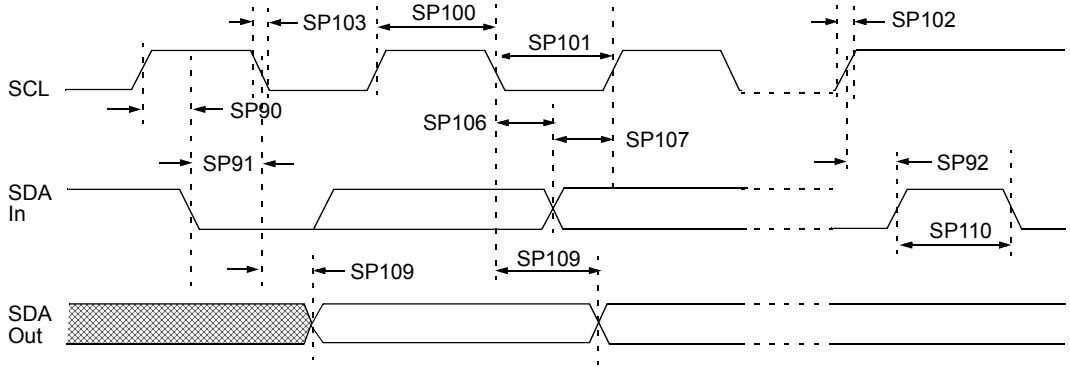

**Note**: Refer to Figure 36-4 for load conditions.

**TABLE 36-26: I²C BUS START/STOP BITS REQUIREMENTS**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **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 36-22: I²C BUS DATA TIMING**




**Note:** Refer to Figure 36-4 for load conditions.

**TABLE 36-27: I²C BUS DATA REQUIREMENTS**

| Standard Operating Conditions (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **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 | 1.5TCY | — | | |
| 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 | 1.5TCY | — | | |
| SP102* | TR | SDA and SCL rise time | 100 kHz mode | — | 1000 | ns | |
| | | | 400 kHz mode | 20 + 0.1CB | 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 + 0.1CB | 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) I²C bus device can be used in a Standard mode (100 kHz) I²C bus system, but the requirement TSU:DAT ≥ 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 TR max. + TSU:DAT = 1000 + 250 = 1250 ns (according to the Standard mode I²C bus specification), before the SCL line is released.

## 37.0 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.

Unless otherwise noted, all graphs apply to both the L and LF devices.

| | |
|---|---|
| **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.**

**Note:** Unless otherwise noted, VIN = 5V, FOSC = 300 kHz, CIN = 0.1 µF, TA = 25°C.

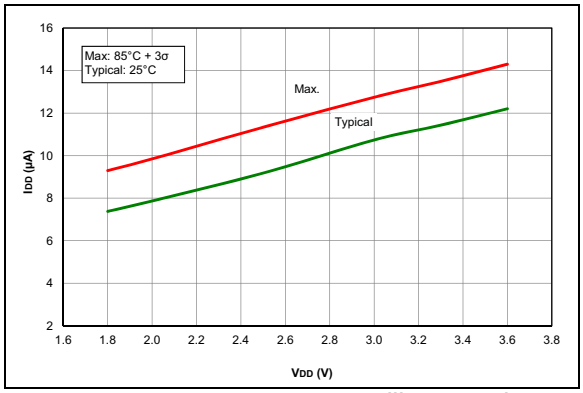



***FIGURE 37-1:*** *IDD, LP Oscillator Mode, Fosc = 32 kHz, PIC16LF1773/6 Only.*

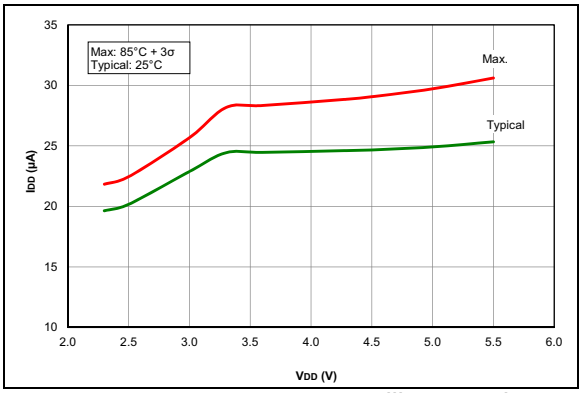



***FIGURE 37-2:*** *IDD, LP Oscillator Mode, Fosc = 32 kHz, PIC16F1773/6 Only.*

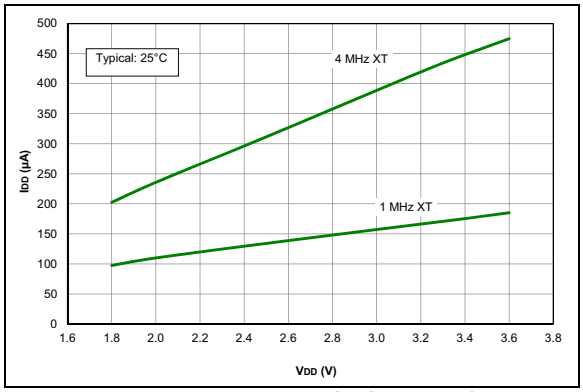



***FIGURE 37-3:*** *IDD Typical, XT and EXTRC Oscillator, PIC16LF1773/6 Only.*

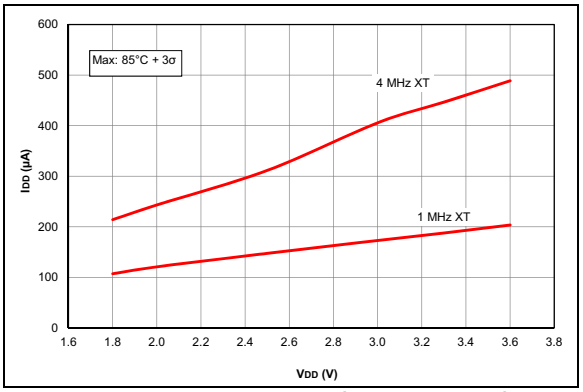



***FIGURE 37-4:*** *IDD Maximum, XT and EXTRC Oscillator, PIC16LF1773/6 Only.*

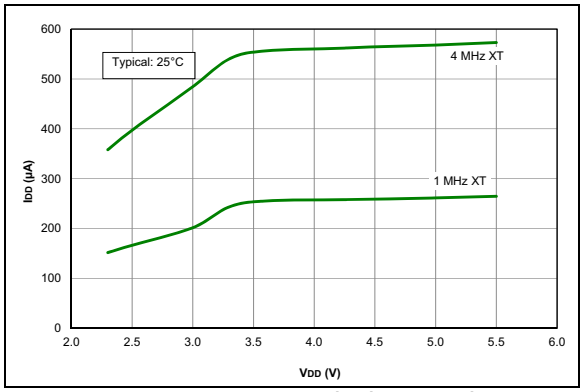



***FIGURE 37-5:*** *IDD Typical, XT and EXTRC Oscillator, PIC16F1773/6 Only.*

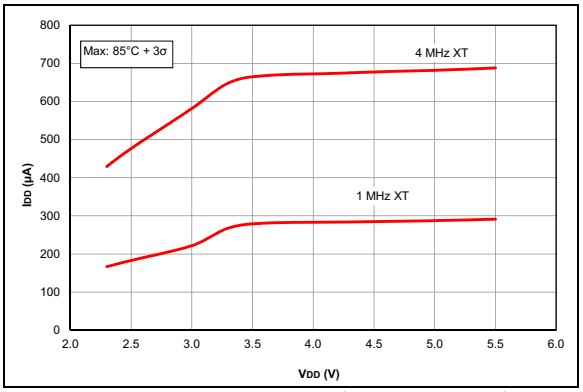



***FIGURE 37-6:*** *IDD Maximum, XT and EXTRC Oscillator, PIC16F1773/6 Only.*

**Note:** Unless otherwise noted, VIN = 5V, FOSC = 300 kHz, CIN = 0.1 µF, TA = 25°C.

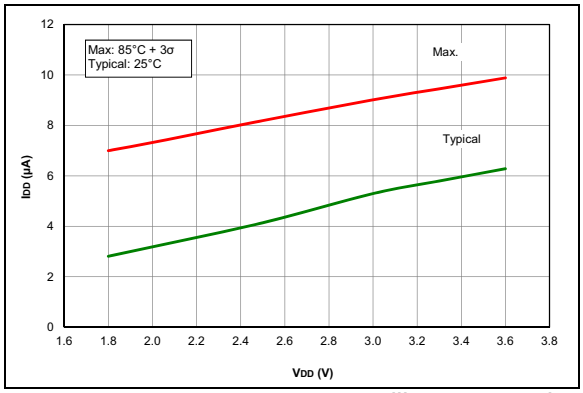



***FIGURE 37-7:*** *IDD, EC Oscillator LP Mode, Fosc = 32 kHz, PIC16LF1773/6 Only.*

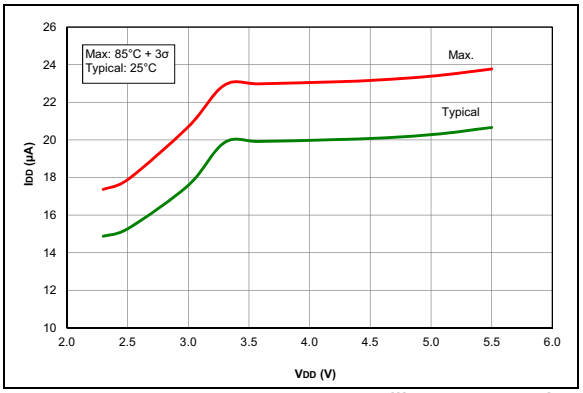



***FIGURE 37-8:*** *IDD, EC Oscillator LP Mode, Fosc = 32 kHz, PIC16F1773/6 Only.*

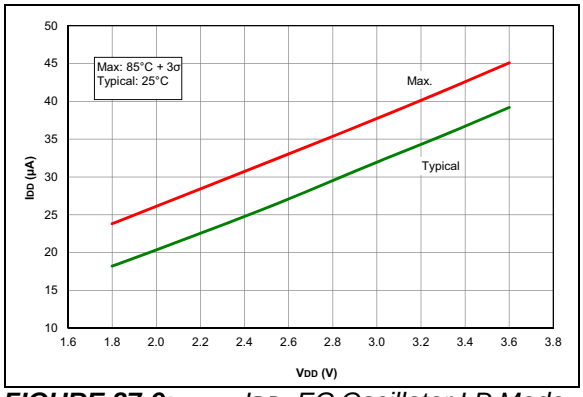



***FIGURE 37-9:*** *IDD, EC Oscillator LP Mode, Fosc = 500 kHz, PIC16LF1773/6 Only.*

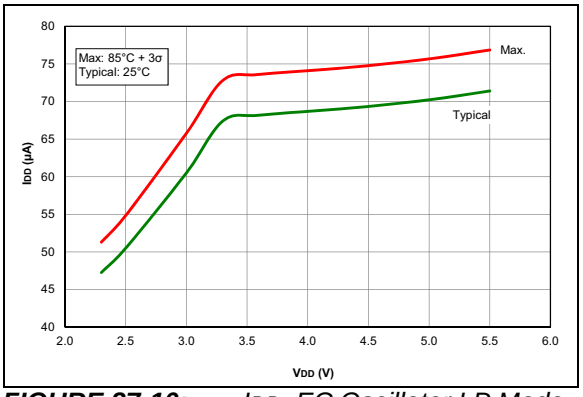



***FIGURE 37-10:*** *IDD, EC Oscillator LP Mode, Fosc = 500 kHz, PIC16F1773/6 Only.*

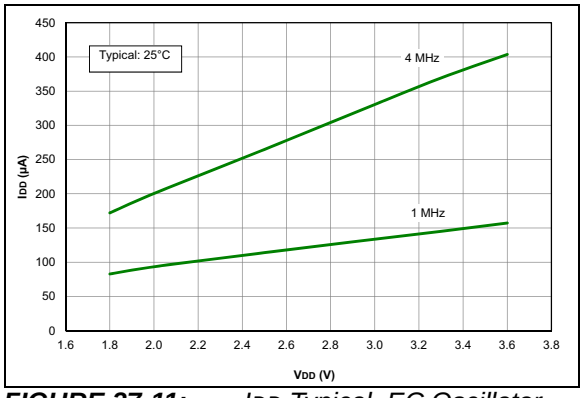



***FIGURE 37-11:*** *IDD Typical, EC Oscillator MP Mode, PIC16LF1773/6 Only.*

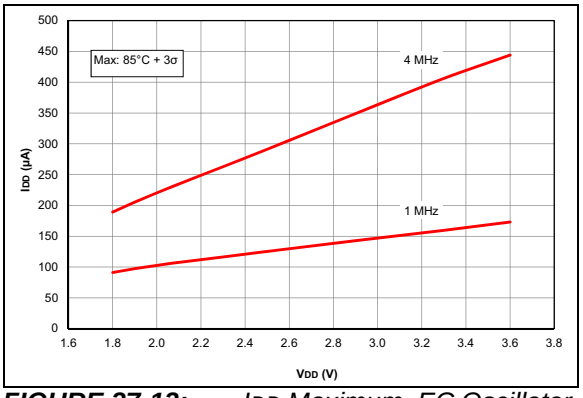



***FIGURE 37-12:*** *IDD Maximum, EC Oscillator MP Mode, PIC16LF1773/6 Only.*

**Note:** Unless otherwise noted, VIN = 5V, FOSC = 300 kHz, CIN = 0.1 µF, TA = 25°C.

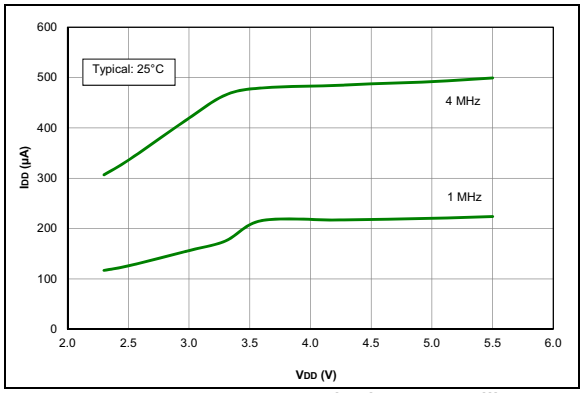



***FIGURE 37-13:*** *IDD Typical, EC Oscillator MP Mode, PIC16F1773/6 Only.*

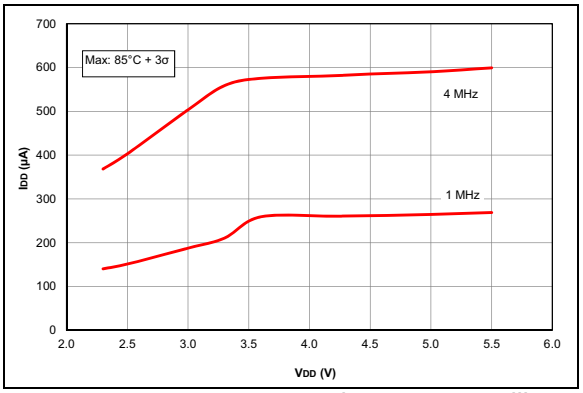



***FIGURE 37-14:*** *IDD Maximum, EC Oscillator MP Mode, PIC16F1773/6 Only.*

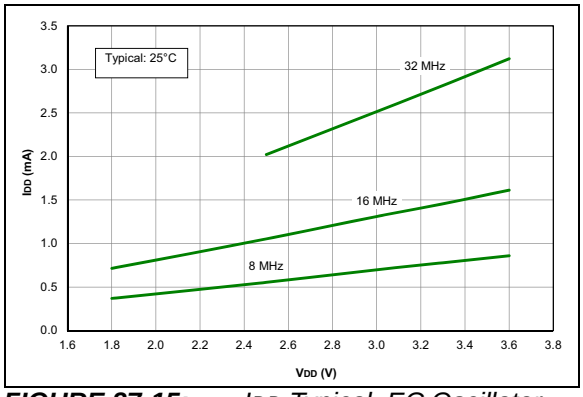



***FIGURE 37-15:*** *IDD Typical, EC Oscillator HP Mode, PIC16LF1773/6 Only.*

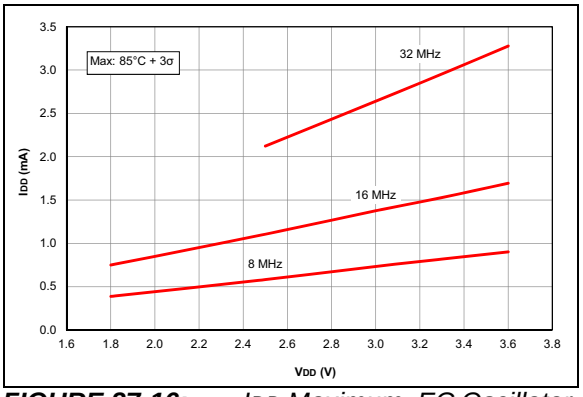



***FIGURE 37-16:*** *IDD Maximum, EC Oscillator HP Mode, PIC16LF1773/6 Only.*

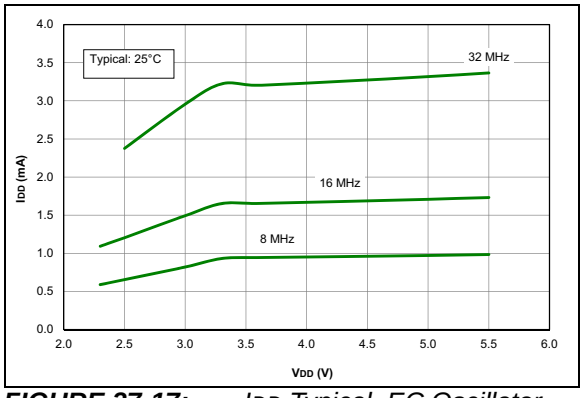



***FIGURE 37-17:*** *IDD Typical, EC Oscillator HP Mode, PIC16F1773/6 Only.*

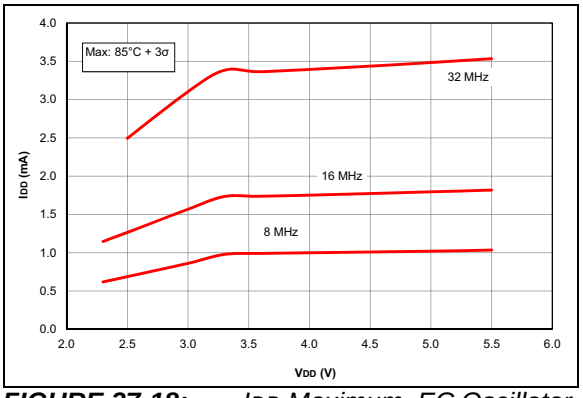



***FIGURE 37-18:*** *IDD Maximum, EC Oscillator HP Mode, PIC16F1773/6 Only.*

**Note:** Unless otherwise noted, VIN = 5V, FOSC = 300 kHz, CIN = 0.1 µF, TA = 25°C.

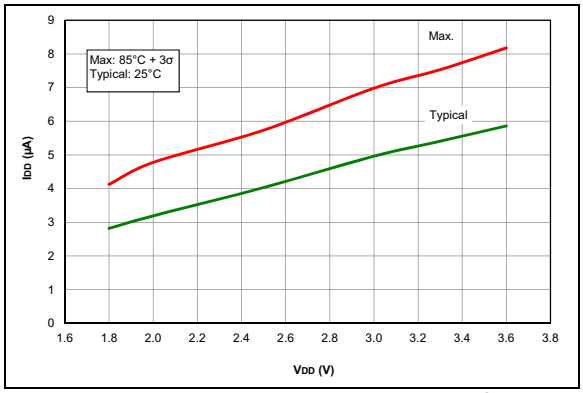



***FIGURE 37-19:*** *IDD, LFINTOSC Mode, Fosc = 31 kHz, PIC16LF1773/6 Only.*

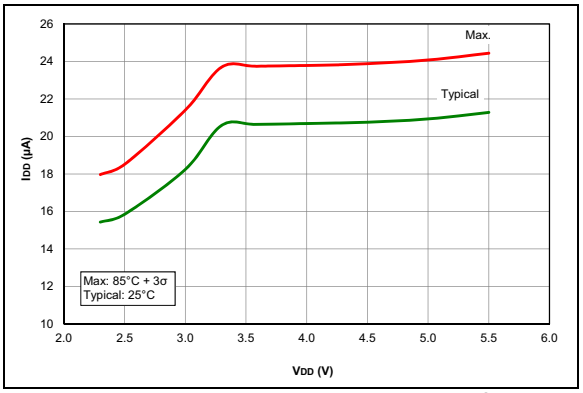



***FIGURE 37-20:*** *IDD, LFINTOSC Mode, Fosc = 31 kHz, PIC16F1773/6 Only.*

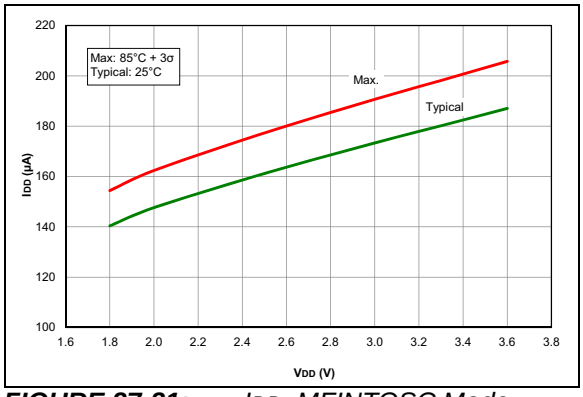



***FIGURE 37-21:*** *IDD, MFINTOSC Mode, Fosc = 500 kHz, PIC16LF1773/6 Only.*

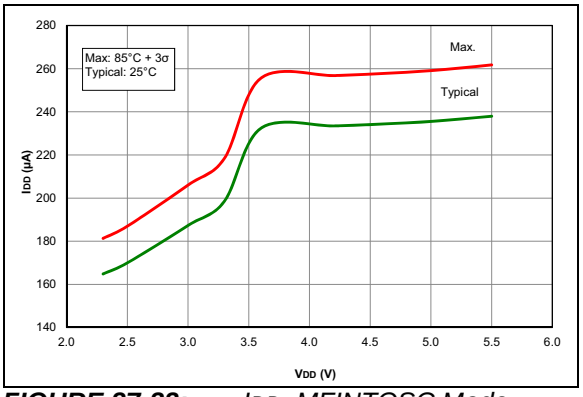



***FIGURE 37-22:*** *IDD, MFINTOSC Mode, Fosc = 500 kHz, PIC16F1773/6 Only.*

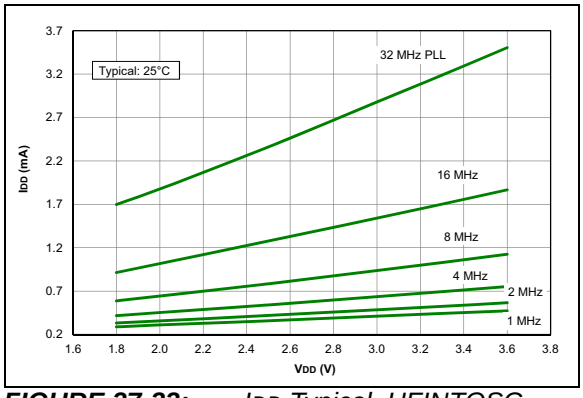



***FIGURE 37-23:*** *IDD Typical, HFINTOSC Mode, PIC16LF1773/6 Only.*

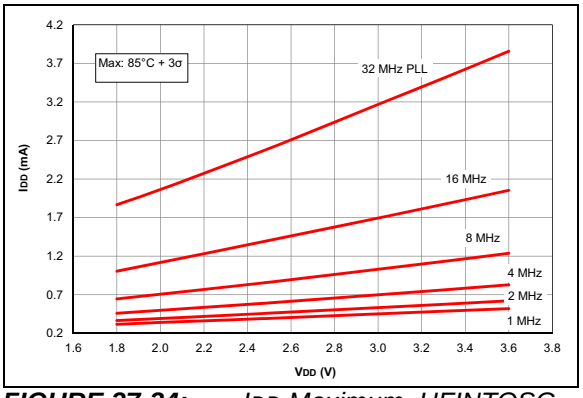



***FIGURE 37-24:*** *IDD Maximum, HFINTOSC Mode, PIC16LF1773/6 Only.*

**Note:** Unless otherwise noted, VIN = 5V, FOSC = 300 kHz, CIN = 0.1 µF, TA = 25°C.

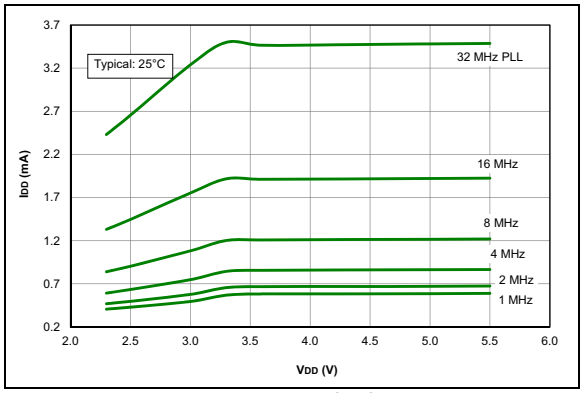



***FIGURE 37-25:*** *IDD Typical, HFINTOSC Mode, PIC16F1773/6 Only.*

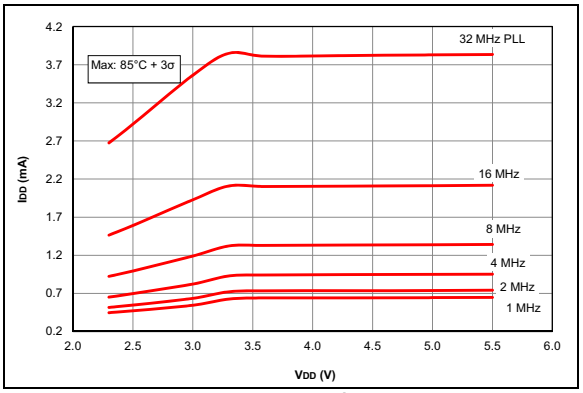



***FIGURE 37-26:*** *IDD Maximum, HFINTOSC Mode, PIC16F1773/6 Only.*

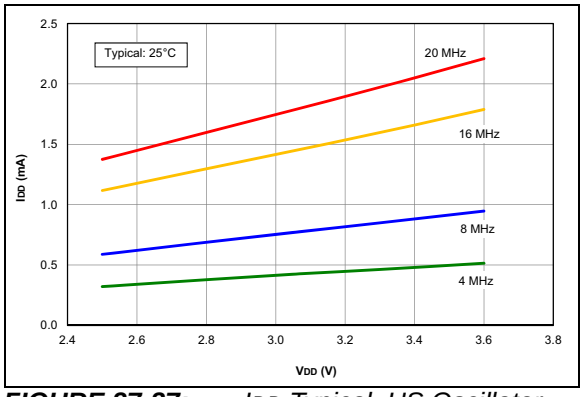



***FIGURE 37-27:*** *IDD Typical, HS Oscillator, 25°C, PIC16LF1773/6 Only.*

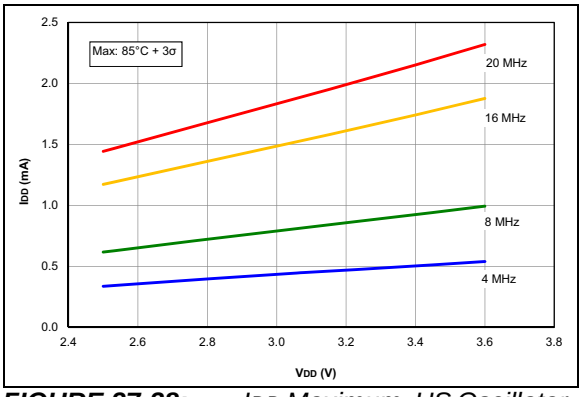



***FIGURE 37-28:*** *IDD Maximum, HS Oscillator, PIC16LF1773/6 Only.*

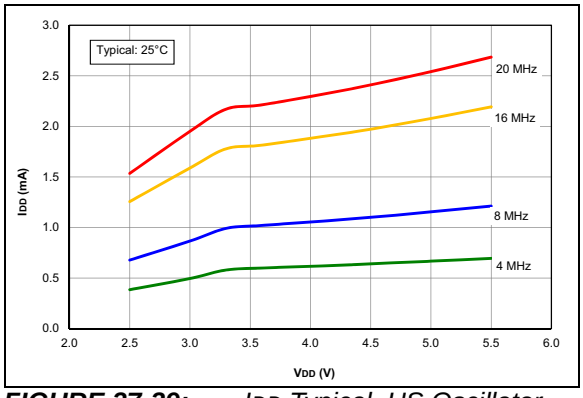



***FIGURE 37-29:*** *IDD Typical, HS Oscillator, 25°C, PIC16F1773/6 Only.*

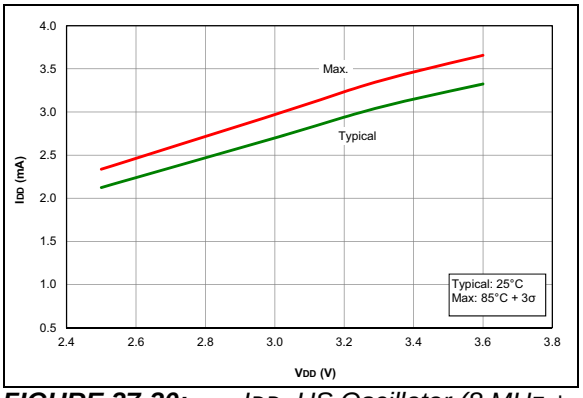



***FIGURE 37-30:*** *IDD, HS Oscillator (8 MHz + 4x PLL), PIC16LF1773/6 Only.*

**Note:** Unless otherwise noted, VIN = 5V, FOSC = 300 kHz, CIN = 0.1 µF, TA = 25°C.

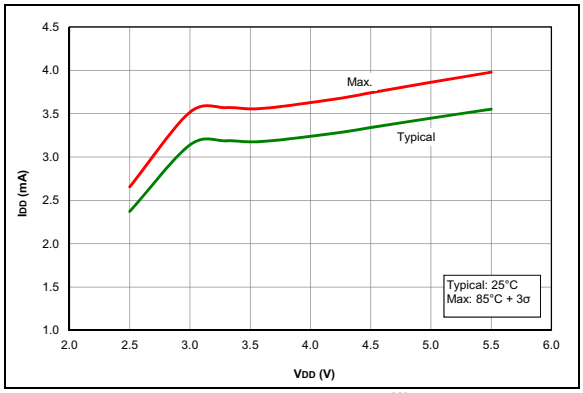



***FIGURE 37-31:*** *IDD, HS Oscillator, 32 MHz (8 MHz + 4x PLL), PIC16F1773/6 Only.*

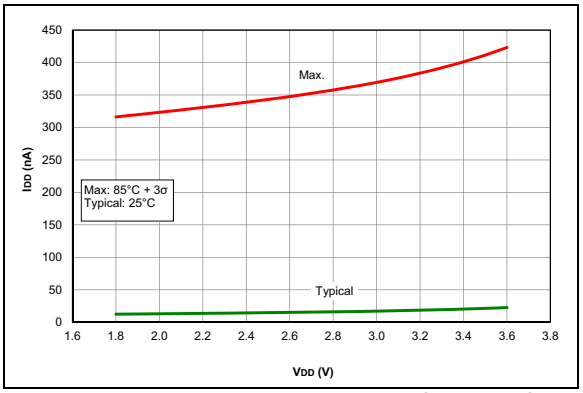



***FIGURE 37-32:*** *IPD Base, LP Sleep Mode, PIC16LF1773/6 Only.*

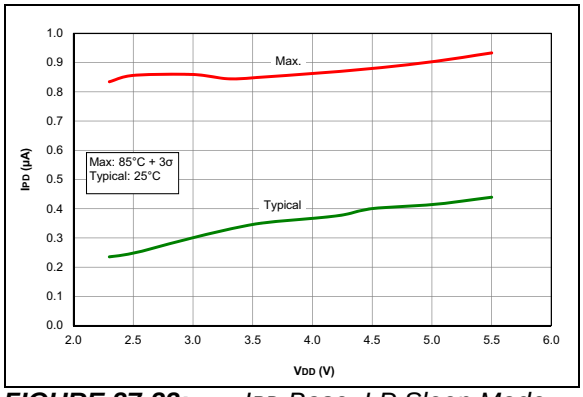



***FIGURE 37-33:*** *IPD Base, LP Sleep Mode (VREGPM = 1), PIC16F1773/6 Only.*

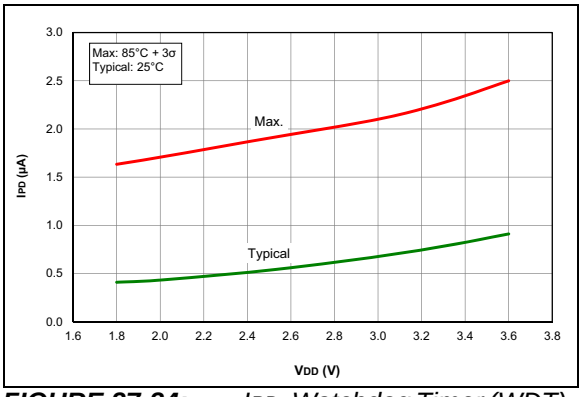



***FIGURE 37-34:*** *IPD, Watchdog Timer (WDT), PIC16LF1773/6 Only.*

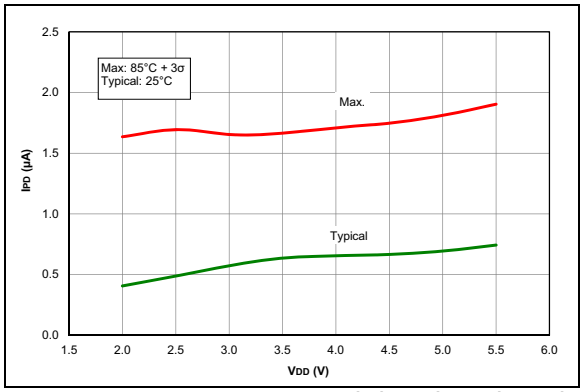



***FIGURE 37-35:*** *IPD, Watchdog Timer (WDT), PIC16F1773/6 Only.*

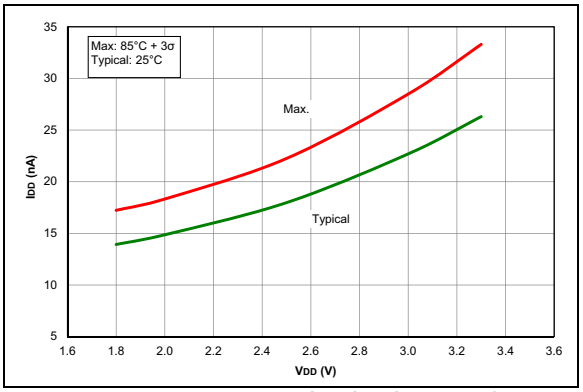



***FIGURE 37-36:*** *IPD, Fixed Voltage Reference (FVR), ADC, PIC16LF1773/6 Only.*

**Note:** Unless otherwise noted, VIN = 5V, FOSC = 300 kHz, CIN = 0.1 µF, TA = 25°C.

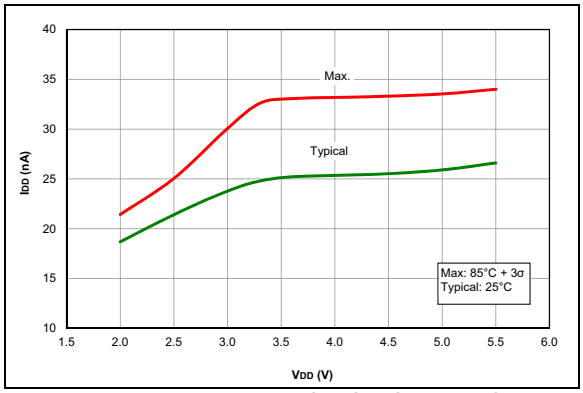



***FIGURE 37-37:*** *IPD, Fixed Voltage Reference (FVR), ADC, PIC16F1773/6 Only.*

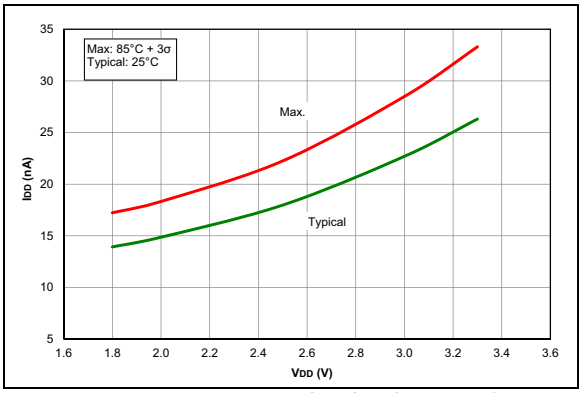



***FIGURE 37-38:*** *IPD, Fixed Voltage Reference (FVR), DAC/Comparator, PIC16LF1773/6 Only.*

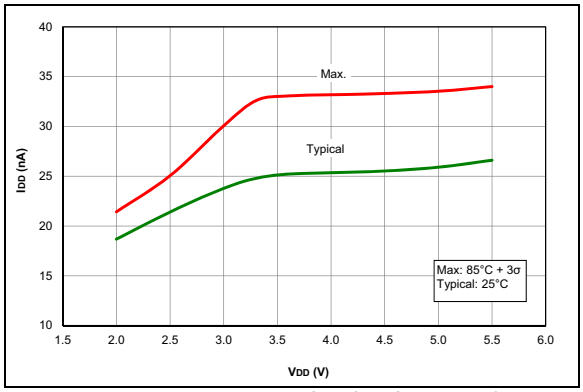



***FIGURE 37-39:*** *IPD, Fixed Voltage Reference (FVR), DAC/Comparator, PIC16F1773/6 Only.*

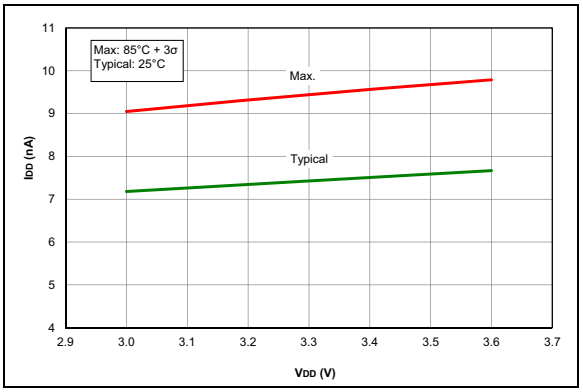



***FIGURE 37-40:*** *IPD, Brown-Out Reset (BOR), BORV = 1, PIC16LF1773/6 Only.*

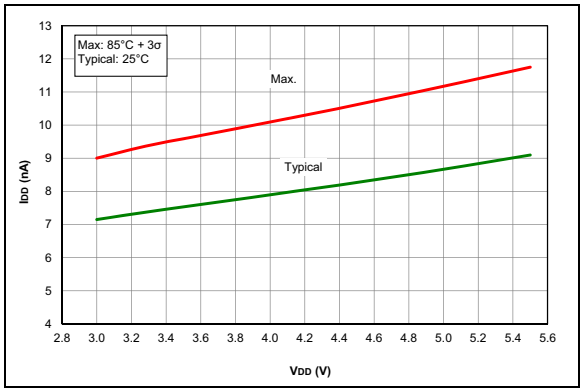



***FIGURE 37-41:*** *IPD, Brown-Out Reset (BOR), BORV = 1, PIC16F1773/6 Only.*

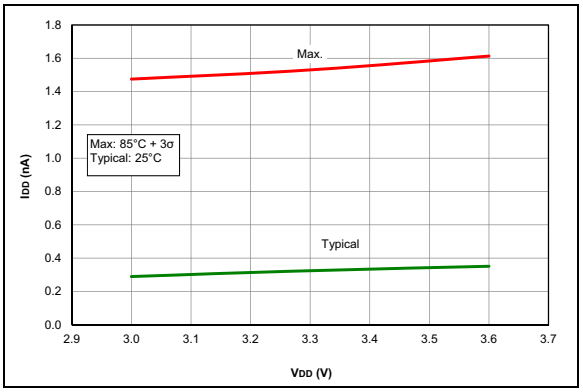



***FIGURE 37-42:*** *IPD, LP Brown-Out Reset (LPBOR = 0), PIC16LF1773/6 Only.*

**Note:** Unless otherwise noted, $V_{IN}$ = 5V, $F_{OSC}$ = 300 kHz, $C_{IN}$ = 0.1 µF, $T_A$ = 25°C.

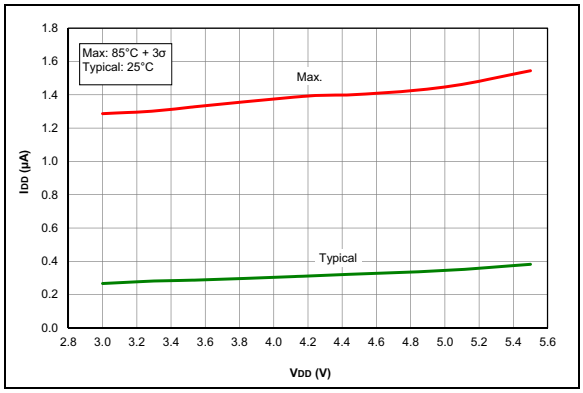



***FIGURE 37-43:*** *IPD, LP Brown-Out Reset (LPBOR = 0), PIC16F1773/6 Only.*

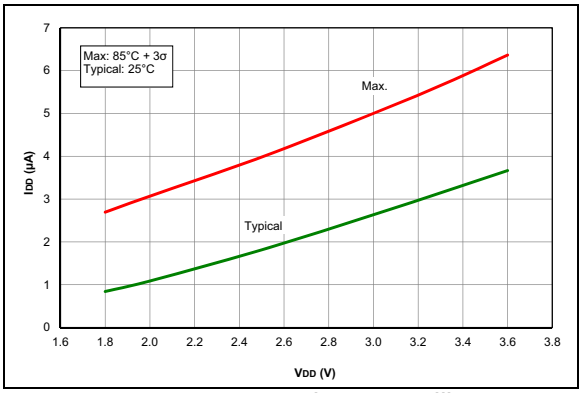



***FIGURE 37-44:*** *IPD, Timer1 Oscillator, FOSC = 32 kHz, PIC16LF1773/6 Only.*

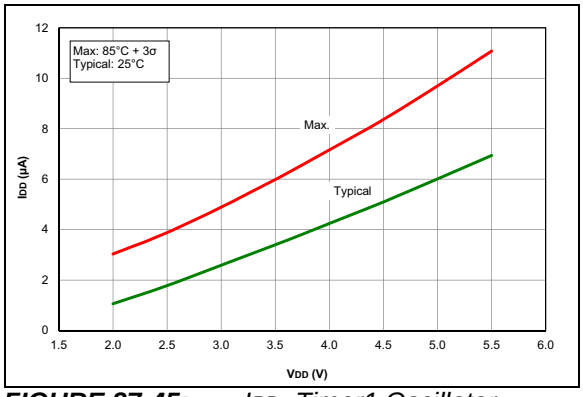



***FIGURE 37-45:*** *IPD, Timer1 Oscillator, FOSC = 32 kHz, PIC16F1773/6 Only.*

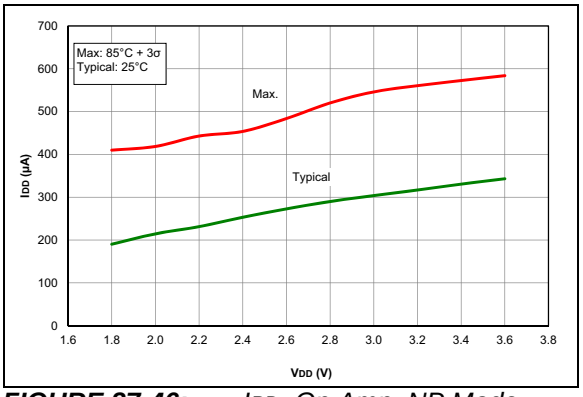



***FIGURE 37-46:*** *IPD, Op Amp, NP Mode (VREFPM = 0), PIC16LF1773/6 Only.*

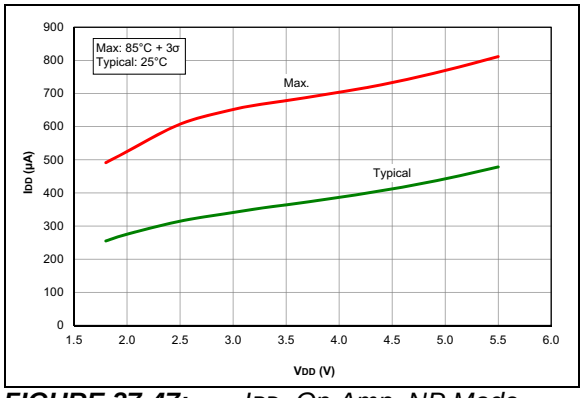



***FIGURE 37-47:*** *IPD, Op Amp, NP Mode (VREFPM = 0), PIC16F1773/6 Only.*

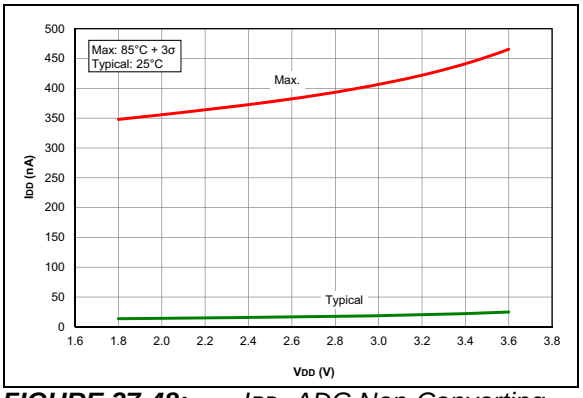



***FIGURE 37-48:*** *IPD, ADC Non-Converting, PIC16LF1773/6 Only.*

**Note:** Unless otherwise noted, $V_{IN}$ = 5V, $F_{OSC}$ = 300 kHz, $C_{IN}$ = 0.1 µF, $T_A$ = 25°C.

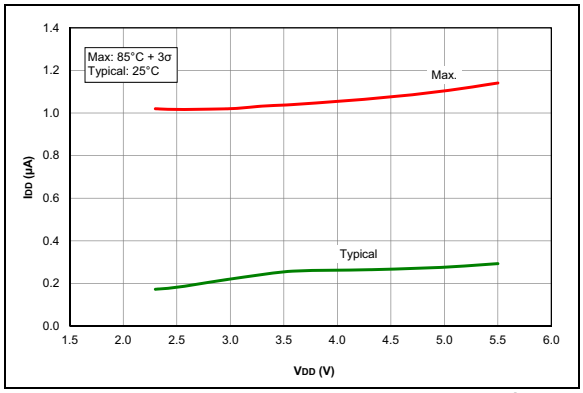



***FIGURE 37-49:*** *IPD, ADC Non-Converting, PIC16F1773/6 Only.*

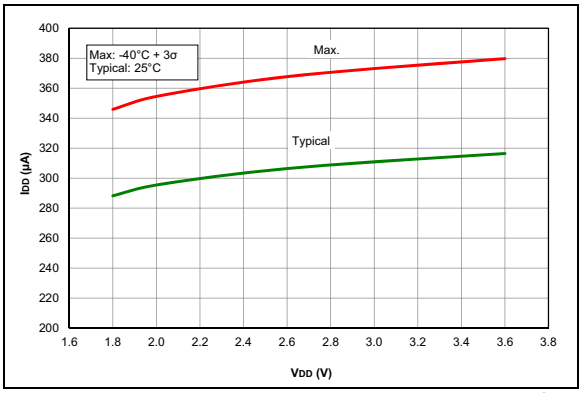



***FIGURE 37-50:*** *IPD, Comparator, NP Mode (VREGPM = 0), PIC16LF1773/6 Only.*

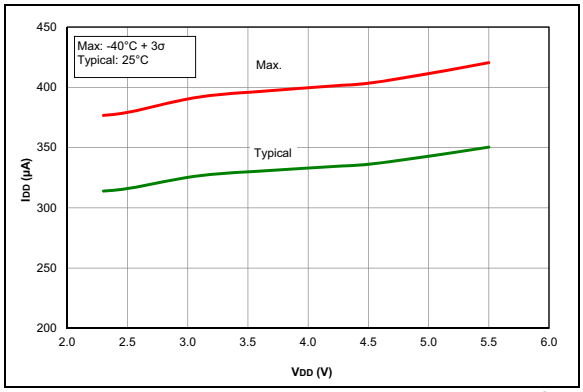



***FIGURE 37-51:*** *IPD, Comparator, NP Mode (VREGPM = 0), PIC16F1773/6 Only.*

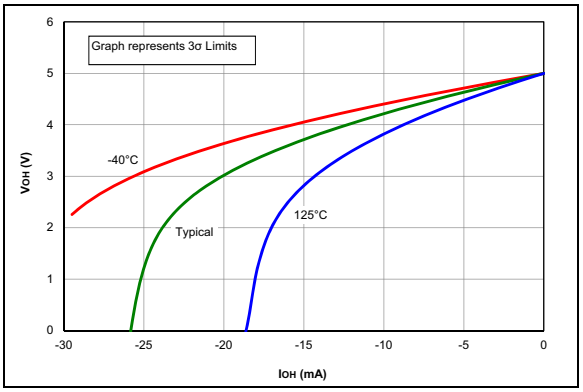



***FIGURE 37-52:*** *VOL vs. IOL Over Temperature, VDD = 5.0V, PIC16F1773/6 Only.*

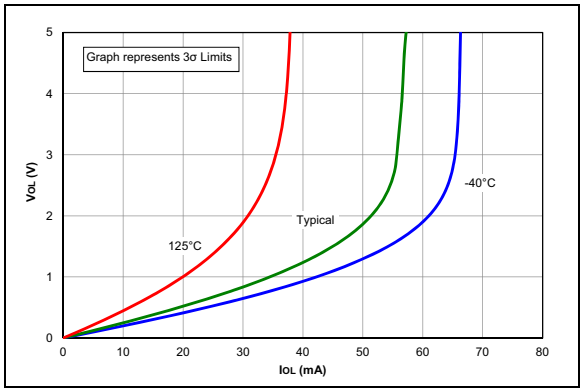



***FIGURE 37-53:*** *VOH vs. IOH Over Temperature, VDD = 5.0V, PIC16F1773/6 Only.*

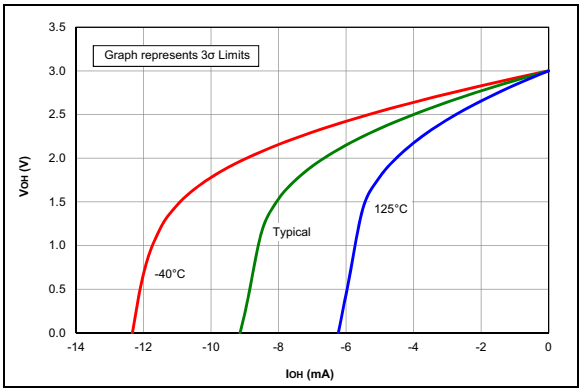



***FIGURE 37-54:*** *VOL vs. IOL Over Temperature, VDD = 3.0V.*

**Note:** Unless otherwise noted, $V_{IN}$ = 5V, $F_{OSC}$ = 300 kHz, $C_{IN}$ = 0.1 µF, $T_A$ = 25°C.

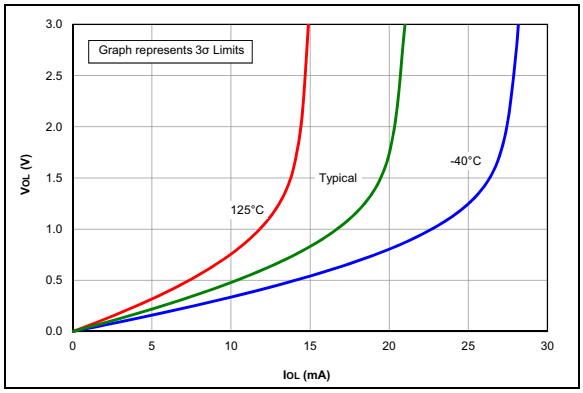



***FIGURE 37-55:*** *$V_{OH}$ vs. $I_{OH}$ Over Temperature, $V_{DD}$ = 3.0V.*

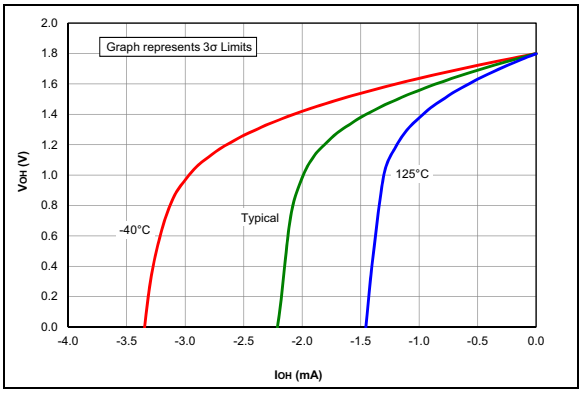



***FIGURE 37-56:*** *$V_{OL}$ vs. $I_{OL}$ Over Temperature, $V_{DD}$ = 1.8V, PIC16LF1773/6 Only.*

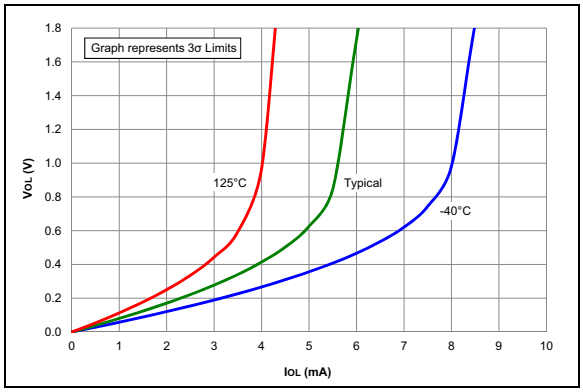



***FIGURE 37-57:*** *$V_{OL}$ vs. $I_{OL}$ Over Temperature, $V_{DD}$ = 1.8V, PIC16LF1773/6 Only.*

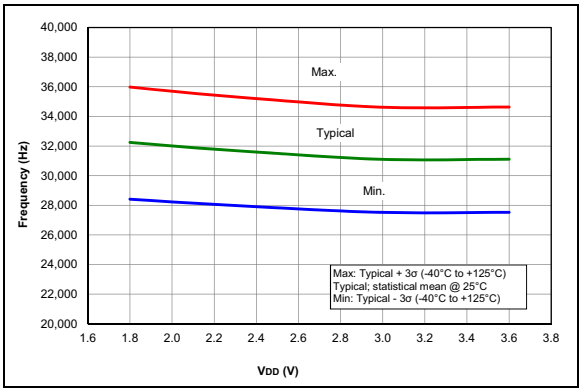



***FIGURE 37-58:*** *LFINTOSC Frequency, PIC16LF1773/6 Only.*

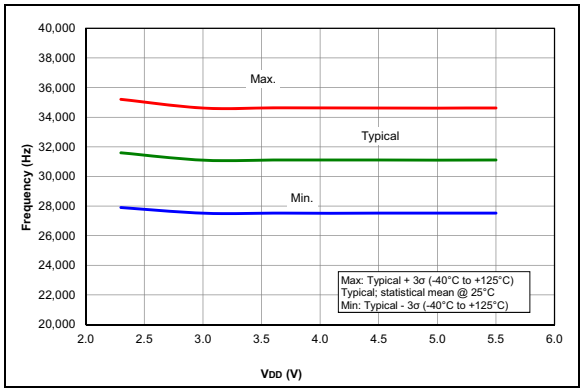



***FIGURE 37-59:*** *LFINTOSC Frequency, PIC16F1773/6 Only.*

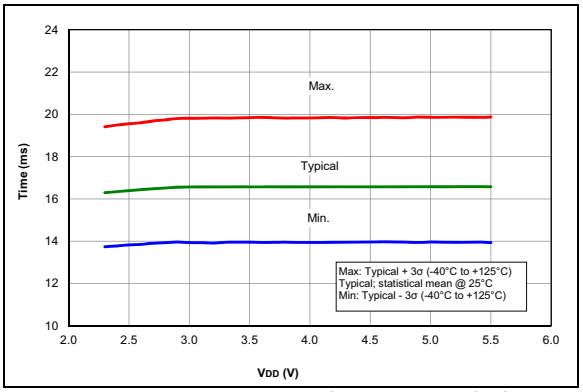



***FIGURE 37-60:*** *WDT Time-Out Period, PIC16F1773/6 Only.*

**Note:** Unless otherwise noted, VIN = 5V, FOSC = 300 kHz, CIN = 0.1 µF, TA = 25°C.

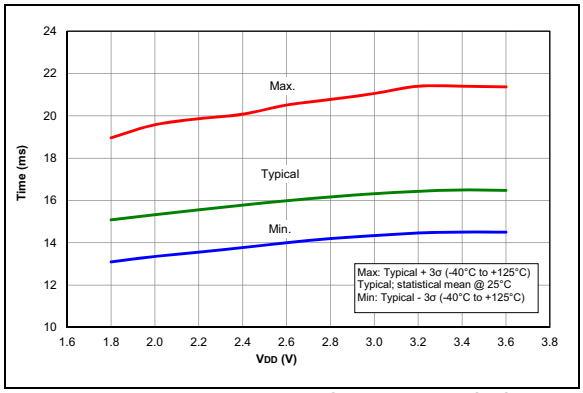



***FIGURE 37-61:*** *WDT Time-Out Period, PIC16LF1773/6 Only.*

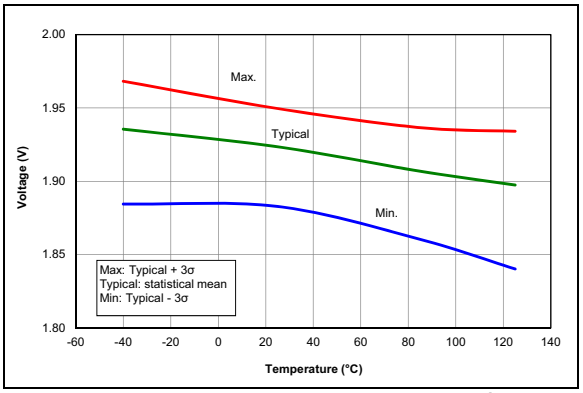



***FIGURE 37-62:*** *Brown-Out Reset Voltage, Low Trip Point (BORV = 1), PIC16LF1773/6 Only.*

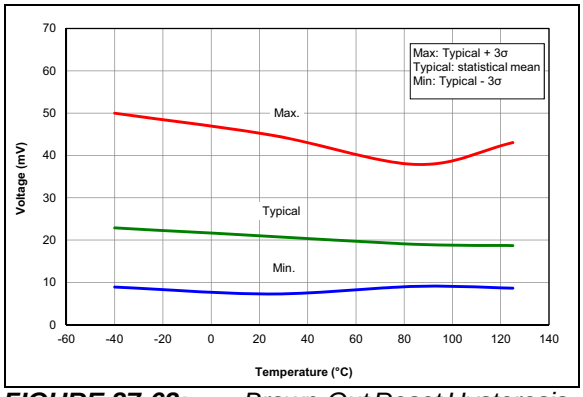



***FIGURE 37-63:*** *Brown-Out Reset Hysteresis, Low Trip Point (BORV = 1), PIC16LF1773/6 Only.*

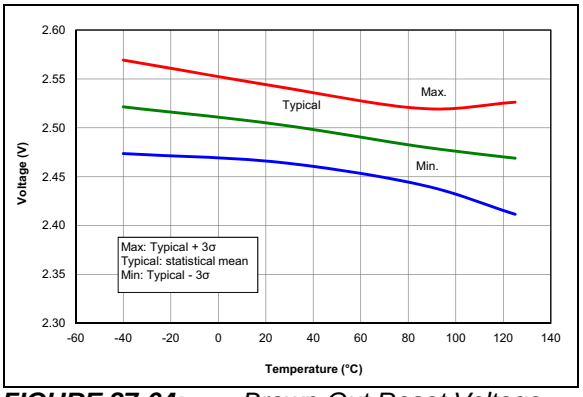



***FIGURE 37-64:*** *Brown-Out Reset Voltage, Low Trip Point (BORV = 1), PIC16F1773/6 Only.*

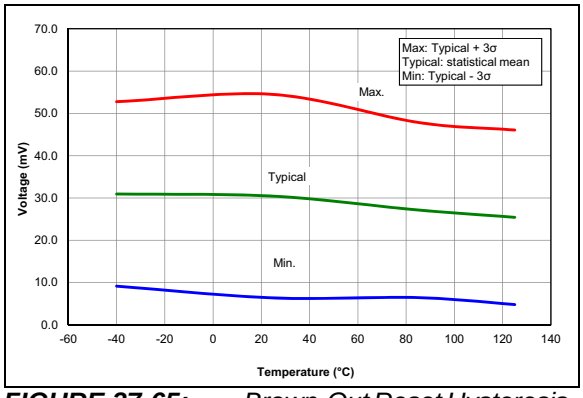



***FIGURE 37-65:*** *Brown-Out Reset Hysteresis, Low Trip Point (BORV = 1), PIC16F1773/6 Only.*

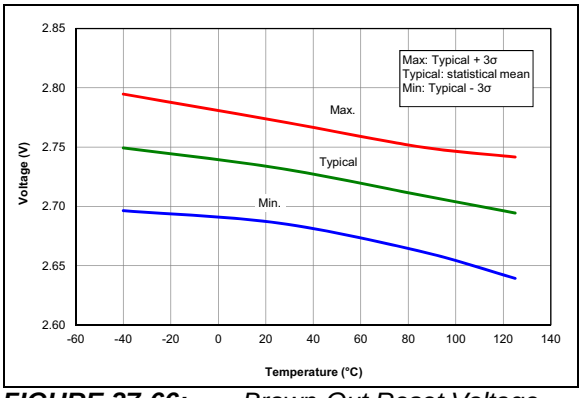



***FIGURE 37-66:*** *Brown-Out Reset Voltage, High Trip Point (BORV = 0).*

**Note:** Unless otherwise noted, $V_{IN}$ = 5V, $F_{OSC}$ = 300 kHz, $C_{IN}$ = 0.1 µF, $T_A$ = 25°C.

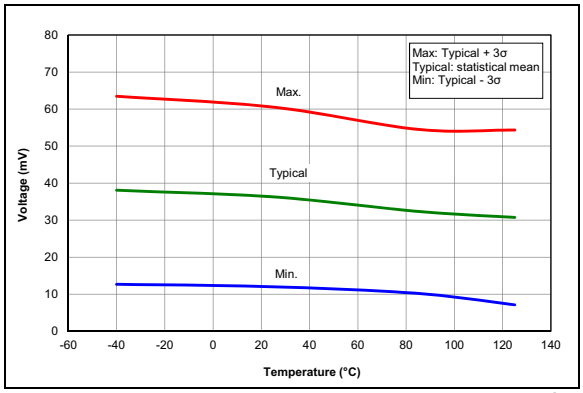



***FIGURE 37-67:*** *Brown-Out Reset Hysteresis, High Trip Point (BORV = 0).*

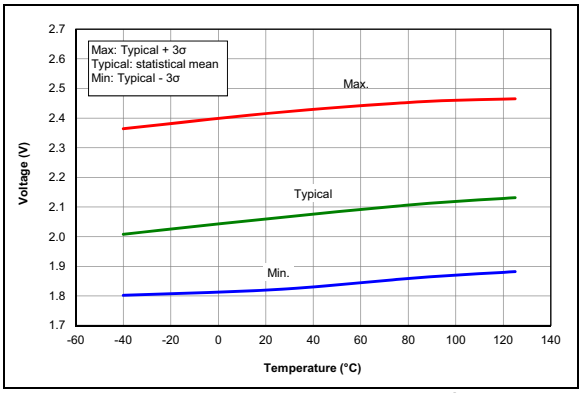



***FIGURE 37-68:*** *LPBOR Reset Voltage.*

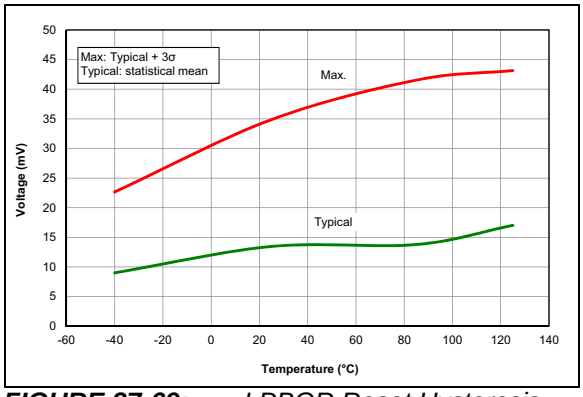



***FIGURE 37-69:*** *LPBOR Reset Hysteresis.*

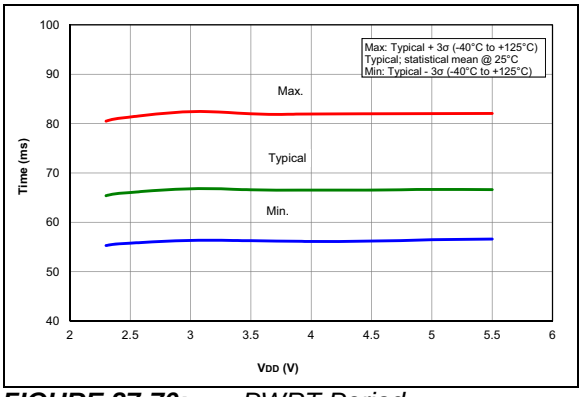



***FIGURE 37-70:*** *PWRT Period, PIC16F1773/6 Only.*

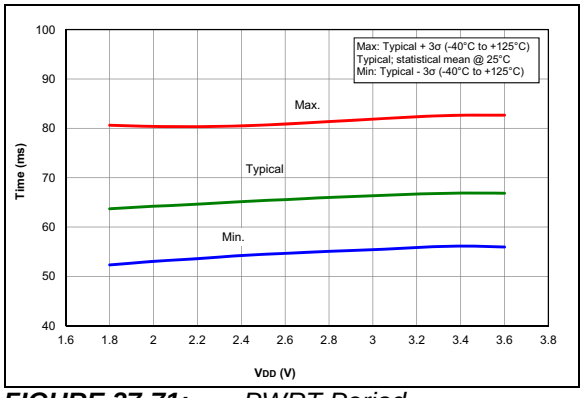



***FIGURE 37-71:*** *PWRT Period, PIC16LF1773/6 Only.*

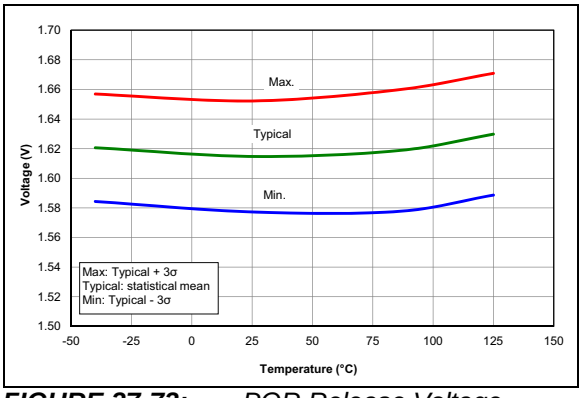



***FIGURE 37-72:*** *POR Release Voltage.*

**Note:** Unless otherwise noted, VIN = 5V, FOSC = 300 kHz, CIN = 0.1 µF, TA = 25°C.

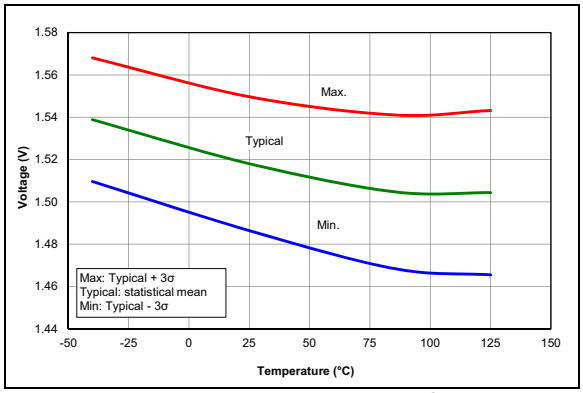



***FIGURE 37-73:*** *POR Rearm Voltage, NP Mode (VREGPM1 = 0), PIC16F1773/6 Only.*

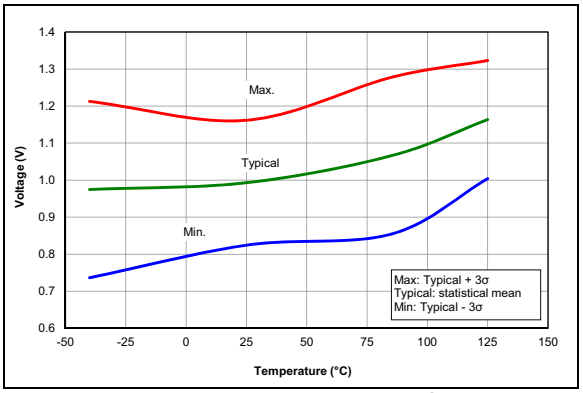



***FIGURE 37-74:*** *POR Rearm Voltage, NP Mode, PIC16LF1773/6 Only.*

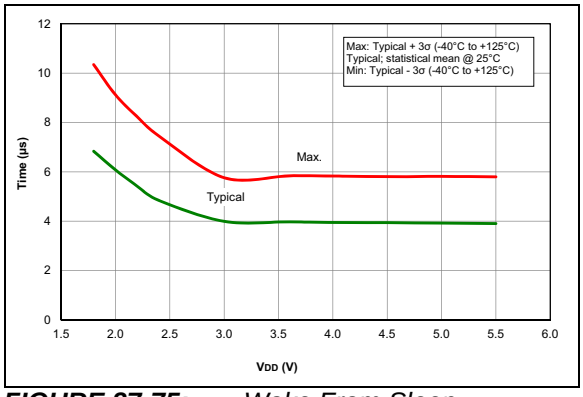



***FIGURE 37-75:*** *Wake From Sleep, VREGPM = 0.*

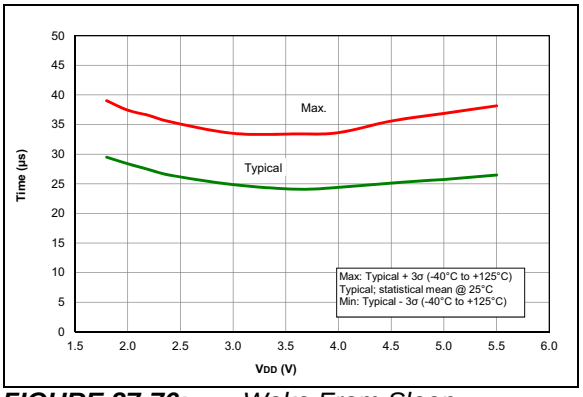



***FIGURE 37-76:*** *Wake From Sleep, VREGPM = 1.*

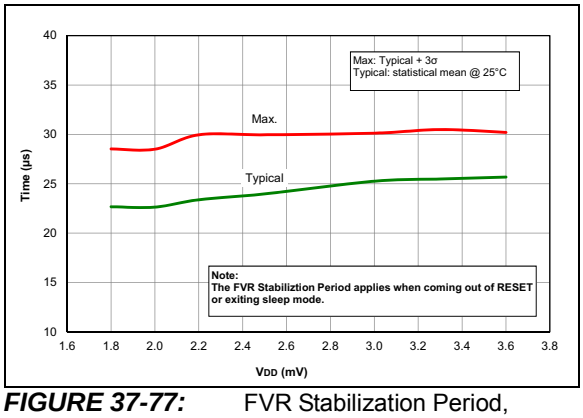



***FIGURE 37-77:*** FVR Stabilization Period, *PIC16LF1773/6 Only.*

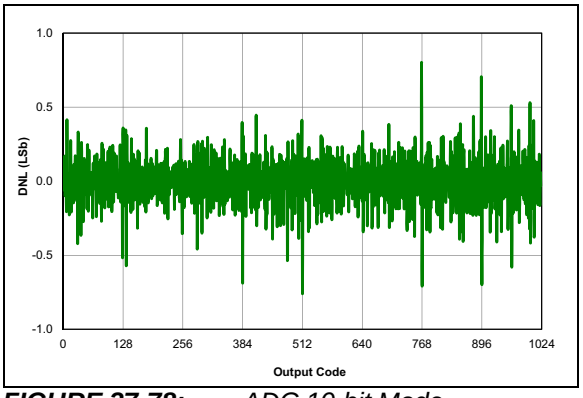



***FIGURE 37-78:*** *ADC 10-bit Mode, Single-Ended DNL, VDD = 3.0V, TAD = 1 µS, 25°C.*

**Note:** Unless otherwise noted, VIN = 5V, FOSC = 300 kHz, CIN = 0.1 µF, TA = 25°C.

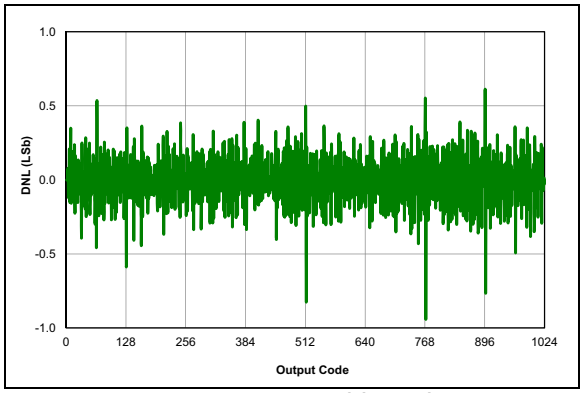



***FIGURE 37-79:*** *ADC 10-bit Mode, Single-Ended DNL, VDD = 3.0V, TAD = 4 µS, 25°C.*

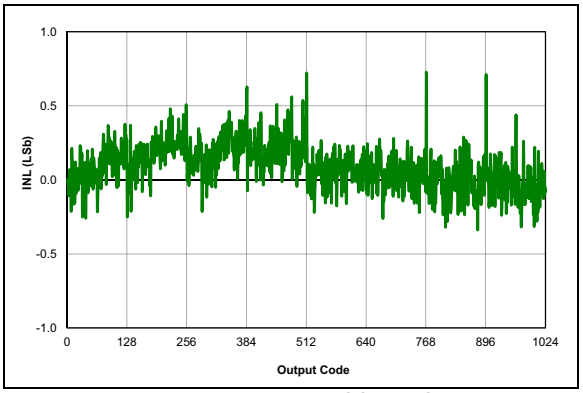



***FIGURE 37-80:*** *ADC 10-bit Mode, Single-Ended INL, VDD = 3.0V, TAD = 1 µS, 25°C.*

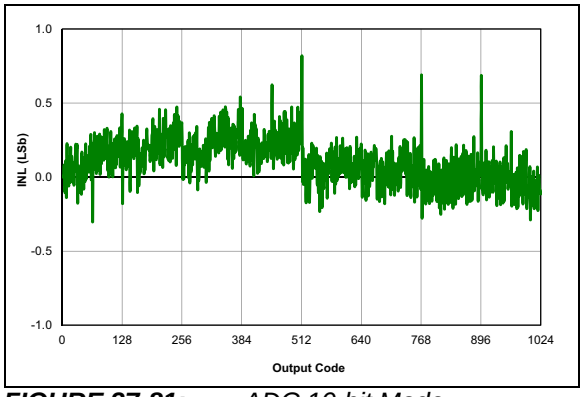



***FIGURE 37-81:*** *ADC 10-bit Mode, Single-Ended INL, VDD = 3.0V, TAD = 4 µS, 25°C.*

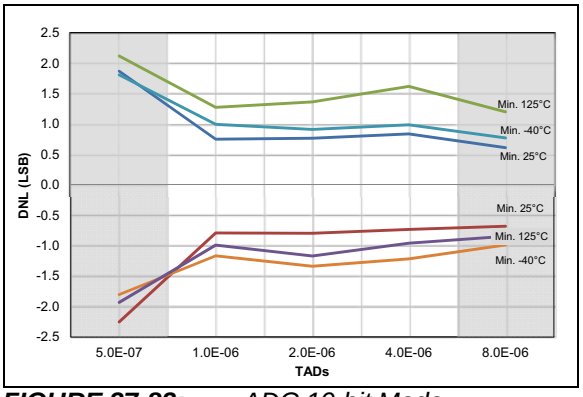



***FIGURE 37-82:*** *ADC 10-bit Mode, Single-Ended DNL, VDD = 3.0V, VREF = 3.0V.*

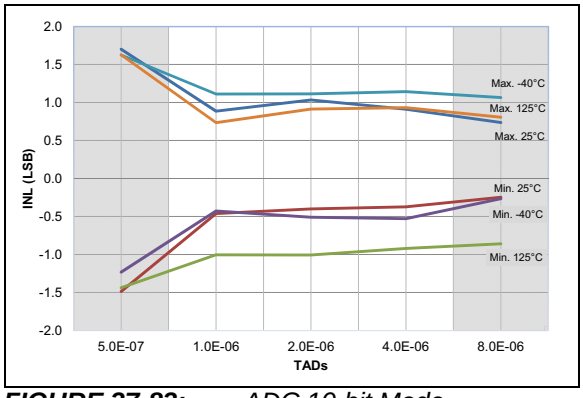



***FIGURE 37-83:*** *ADC 10-bit Mode, Single-Ended INL, VDD = 3.0V, VREF = 3.0V.*

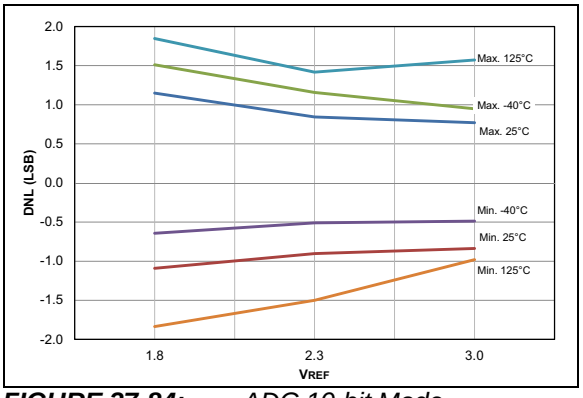



***FIGURE 37-84:*** *ADC 10-bit Mode, Single-Ended DNL, VDD = 3.0V, TAD = 1 µS.*

**Note:** Unless otherwise noted, VIN = 5V, FOSC = 300 kHz, CIN = 0.1 µF, TA = 25°C.

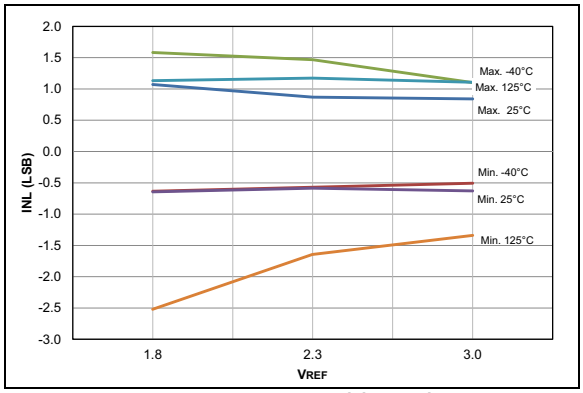



***FIGURE 37-85:*** *ADC 10-bit Mode, Single-Ended INL, VDD = 3.0V, TAD = 1 µS.*

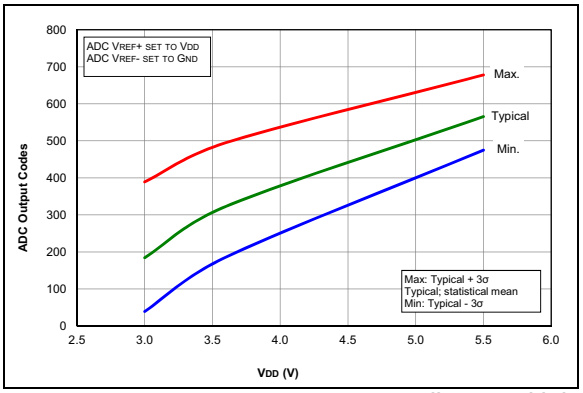



***FIGURE 37-86:*** *Temperature Indicator Initial Offset, High Range, Temp. = 20°C, PIC16F1773/6 only.*

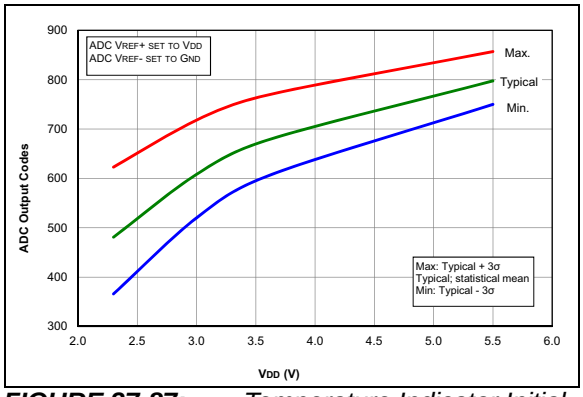



***FIGURE 37-87:*** *Temperature Indicator Initial Offset, Low Range, Temp. = 20°C, PIC16F1773/6 only.*

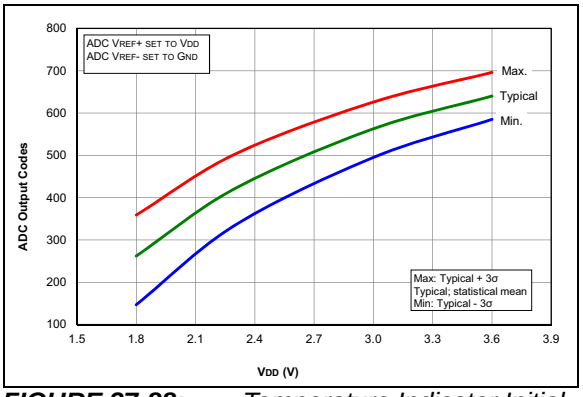



***FIGURE 37-88:*** *Temperature Indicator Initial Offset, Low Range, Temp. = 20°C, PIC16LF1773/6 only.*

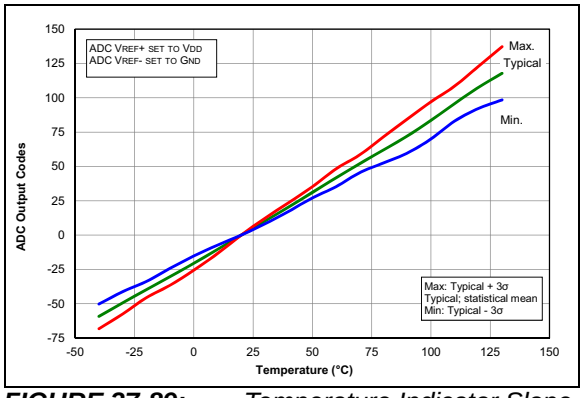



***FIGURE 37-89:*** *Temperature Indicator Slope Normalized to 20°C, PIC16F1773/6 only.*

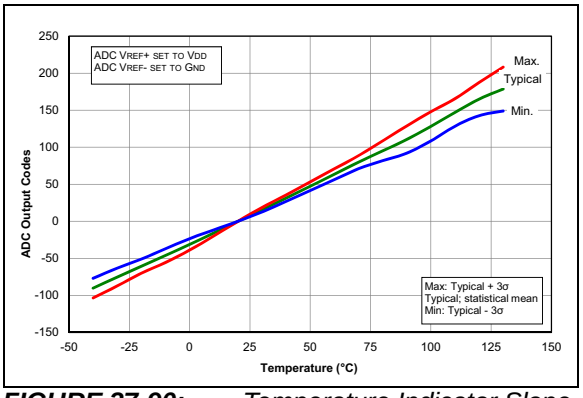



***FIGURE 37-90:*** *Temperature Indicator Slope Normalized to 20°C, High Range, VDD = 3.6V, PIC16F1773/6 only.*

**Note:** Unless otherwise noted, VIN = 5V, FOSC = 300 kHz, CIN = 0.1 µF, TA = 25°C.

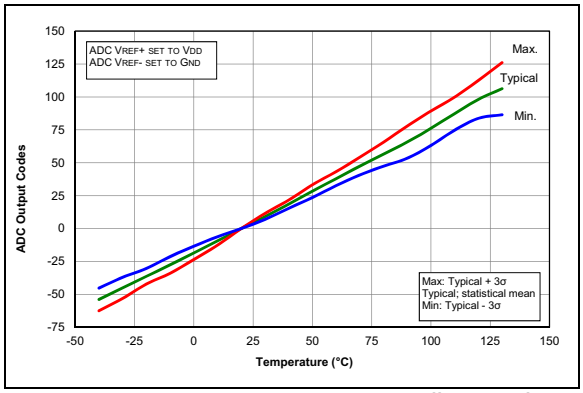



***FIGURE 37-91:*** *Temperature Indicator Slope Normalized to 20°C, Low Range, VDD = 3.0V, PIC16F1773/6 only.*

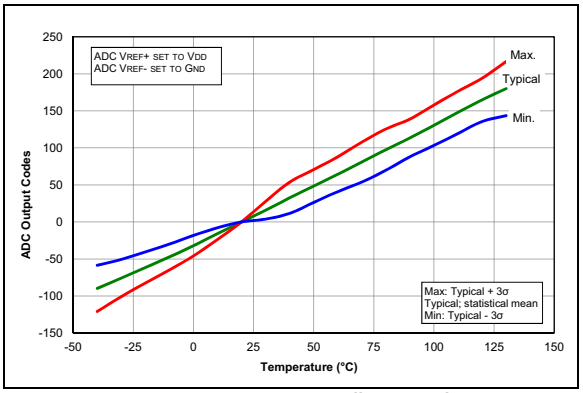



***FIGURE 37-92:*** *Temp. Indicator Slope Normalized to 20°C, Low Range, VDD = 1.8V, PIC16LF1773/6 Only.*

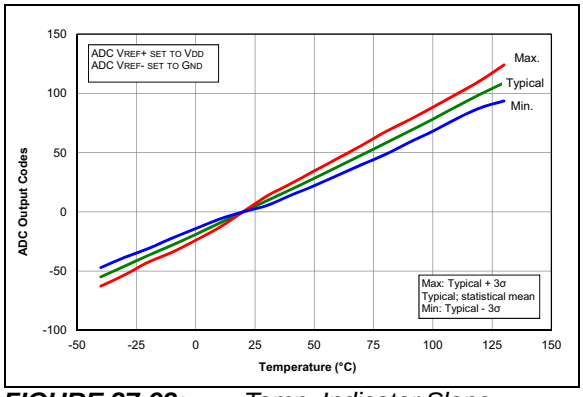



***FIGURE 37-93:*** *Temp. Indicator Slope Normalized to 20°C, Low Range, VDD = 3.0V, PIC16LF1773/6 Only.*

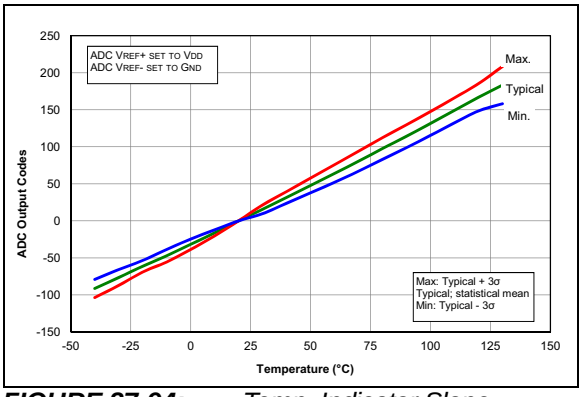



***FIGURE 37-94:*** *Temp. Indicator Slope Normalized to 20°C, High Range, VDD = 3.6V, PIC16LF1773/6 Only.*

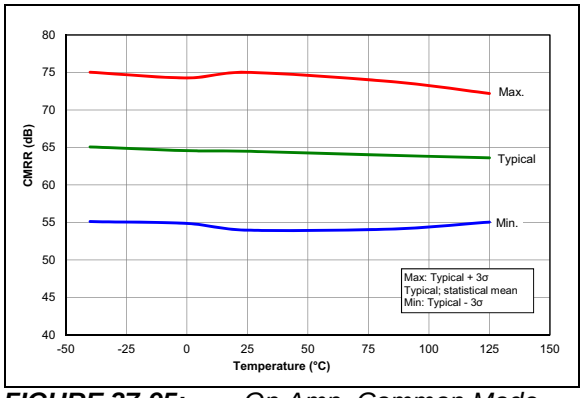



***FIGURE 37-95:*** *Op Amp, Common Mode Rejection Ratio (CMRR), VDD = 3.0V.*

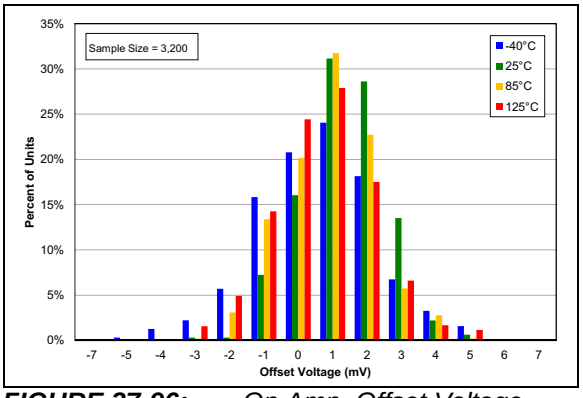



***FIGURE 37-96:*** *Op Amp, Offset Voltage Histogram, VDD = 3.0V, VCM = VDD/2.*

**Note:** Unless otherwise noted, VIN = 5V, FOSC = 300 kHz, CIN = 0.1 µF, TA = 25°C.

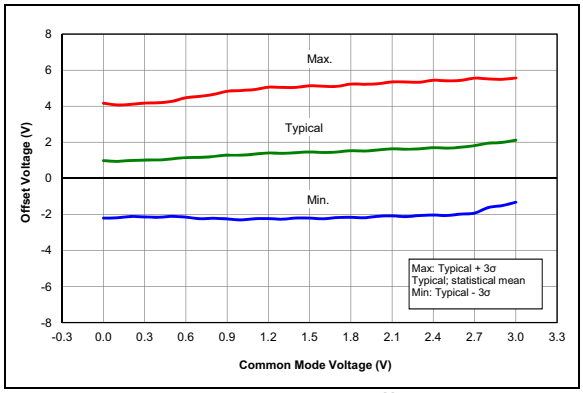



***FIGURE 37-97:*** *Op Amp, Offset over Common Mode Voltage, VDD = 3.0V, Temp. = 25°C*

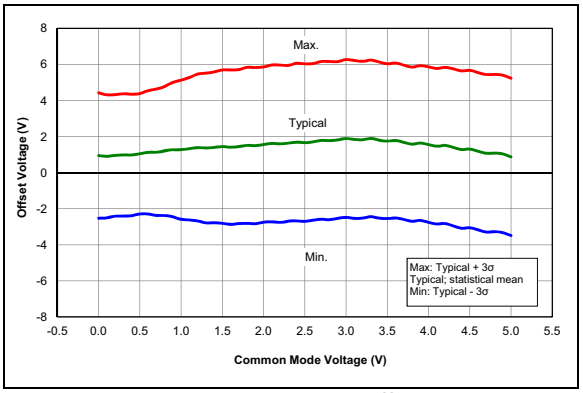



***FIGURE 37-98:*** *Op Amp, Offset over Common Mode Voltage, VDD = 5.0V, Temp. = 25°C, PIC16F1773/6 Only.*

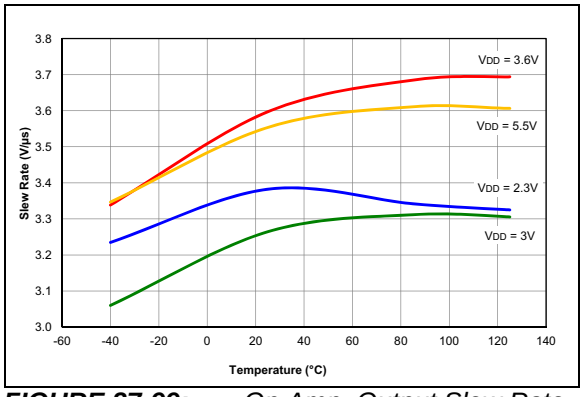



***FIGURE 37-99:*** *Op Amp, Output Slew Rate, Rising Edge, PIC16F1773/6 Only.*

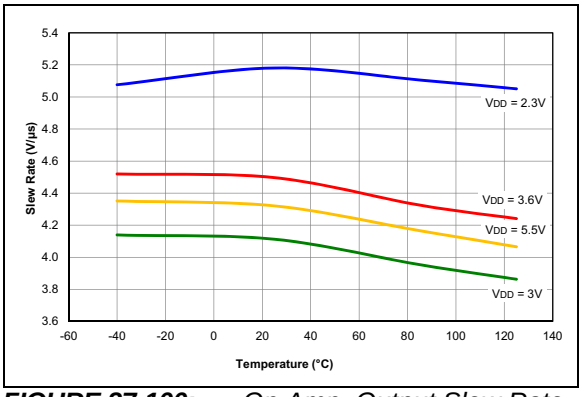



***FIGURE 37-100:*** *Op Amp, Output Slew Rate, Falling Edge, PIC16F1773/6 Only.*

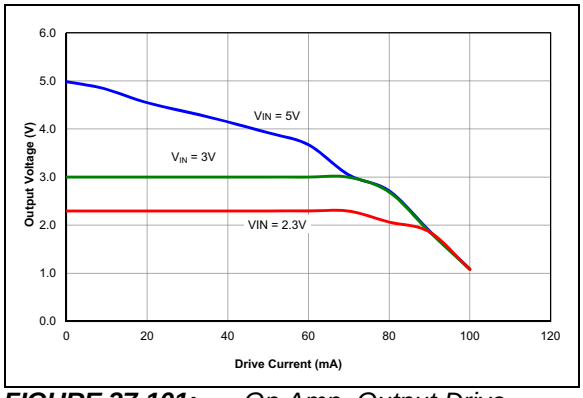



***FIGURE 37-101:*** *Op Amp, Output Drive Strength, VDD = 5.0V, Temp. = 25°C, PIC16F1773/6 Only.*

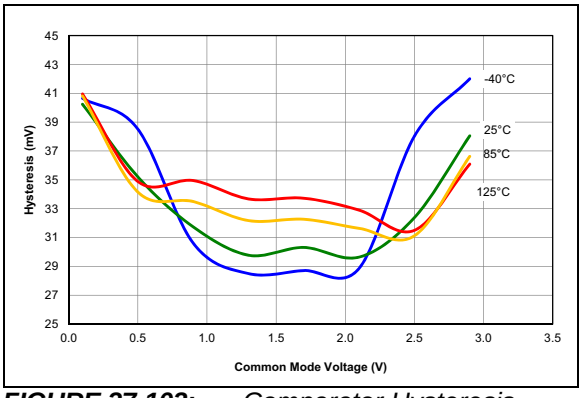



***FIGURE 37-102:*** *Comparator Hysteresis, NP Mode (CxSP = 1), VDD = 3.0V, Typical Measured Values.*

**Note:** Unless otherwise noted, VIN = 5V, FOSC = 300 kHz, CIN = 0.1 µF, TA = 25°C.

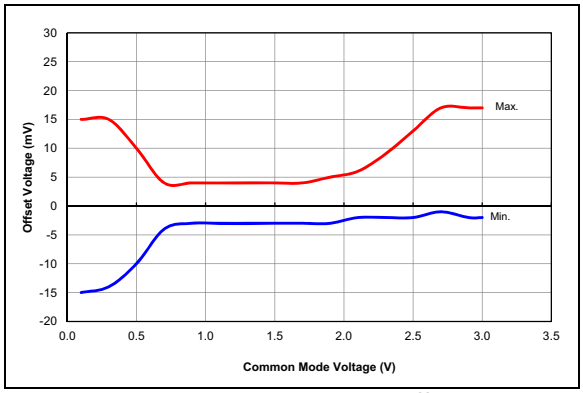



***FIGURE 37-103:*** *Comparator Offset, NP Mode (CxSP = 1), VDD = 3.0V, Typical Measured Values at 25°C.*

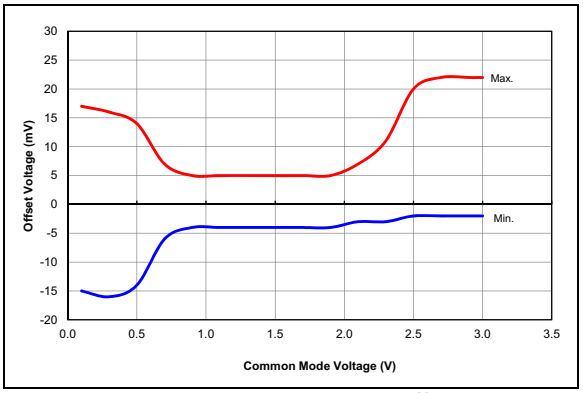



***FIGURE 37-104:*** *Comparator Offset, NP Mode (CxSP = 1), VDD = 3.0V, Typical Measured Values from -40°C to 125°C.*

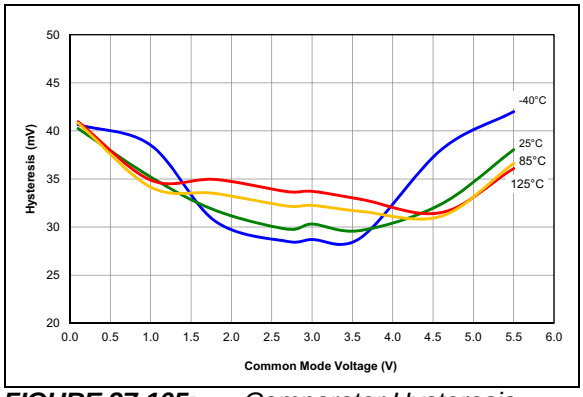



***FIGURE 37-105:*** *Comparator Hysteresis, NP Mode (CxSP = 1), VDD = 5.5V, Typical Measured Values, PIC16F1773/6 only.*

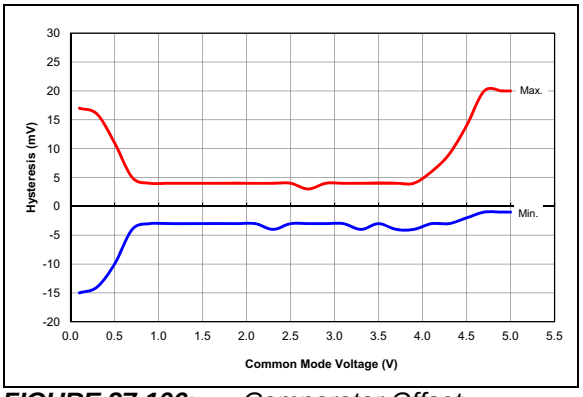



***FIGURE 37-106:*** *Comparator Offset, NP Mode (CxSP = 1), VDD = 5.0V, Typical Measured Values at 25°C, PIC16F1773/6 only.*

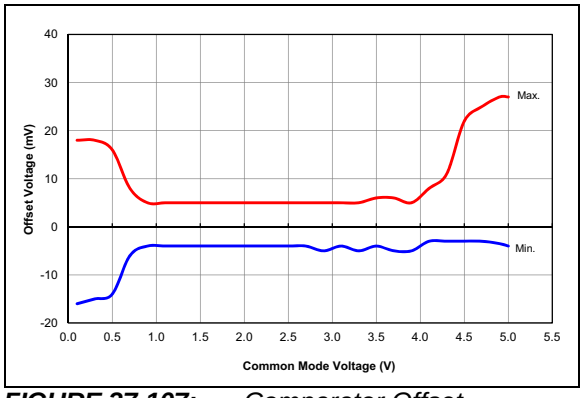



***FIGURE 37-107:*** *Comparator Offset, NP Mode (CxSP = 1), VDD = 5.5V, Typical Measured Values from -40°C to 125°C, PIC16F1773/6 only.*

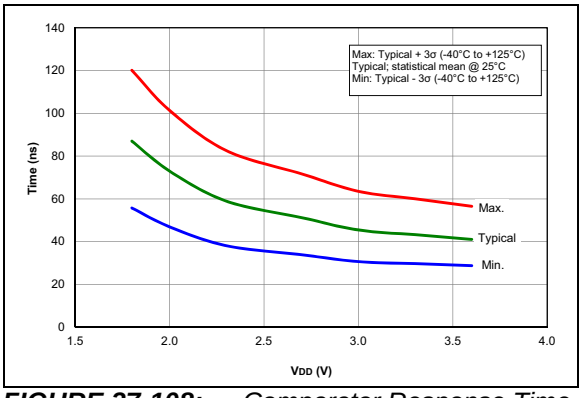



***FIGURE 37-108:*** *Comparator Response Time Over Voltage, NP Mode (CxSP = 1), Typical Measured Values.*

**Note:** Unless otherwise noted, $V_{IN}$ = 5V, $F_{OSC}$ = 300 kHz, $C_{IN}$ = 0.1 µF, $T_A$ = 25°C.

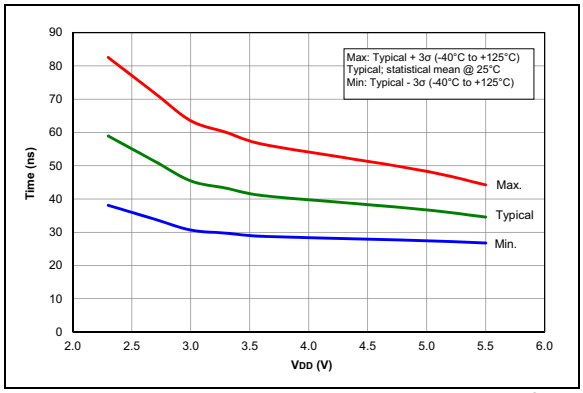



***FIGURE 37-109:*** *Comparator Response Time Over Voltage, NP Mode (CxSP = 1), Typical Measured Values, PIC16F1773/6 Only.*

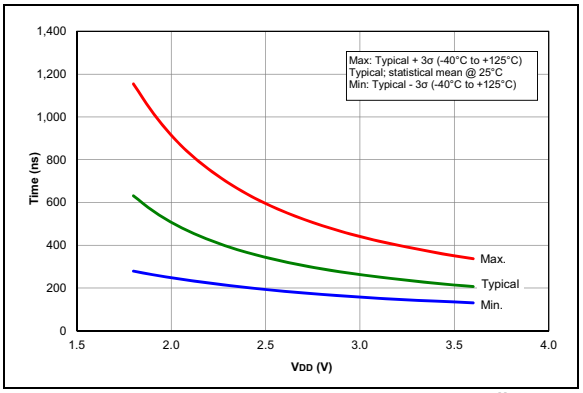



***FIGURE 37-110:*** *Comparator Output Filter Delay Time Over Temp., NP Mode (CxSP = 1), Typical Measured Values, PIC16LF1773/6 Only.*

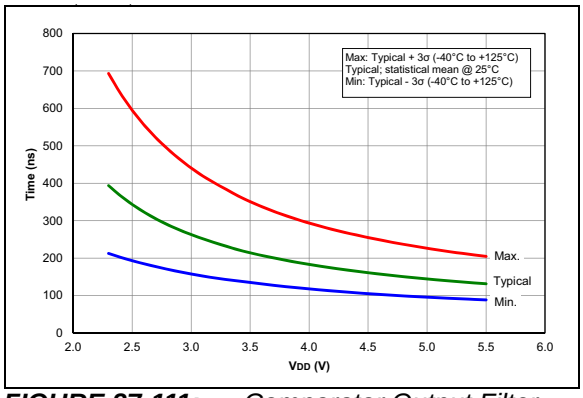



***FIGURE 37-111:*** *Comparator Output Filter Delay Time Over Temp., NP Mode (CxSP = 1), Typical Measured Values, PIC16F1773/6 Only.*

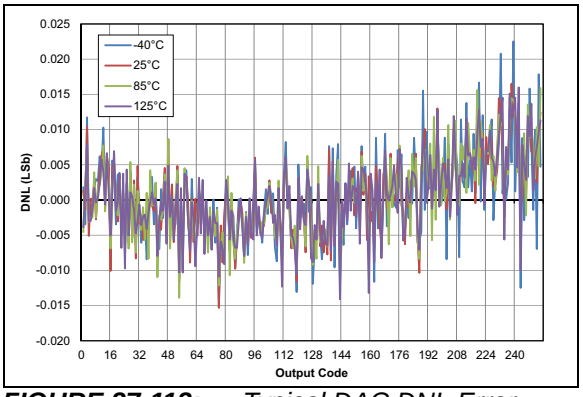



***FIGURE 37-112:*** *Typical DAC DNL Error, $V_{DD}$ = 3.0V, $V_{REF}$ = External 3V.*

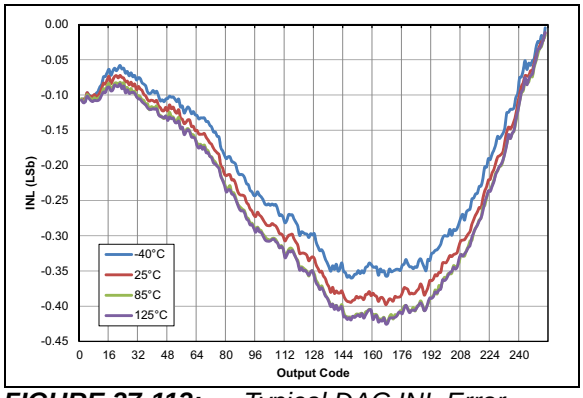



***FIGURE 37-113:*** *Typical DAC INL Error, $V_{DD}$ = 3.0V, $V_{REF}$ = External 3V.*

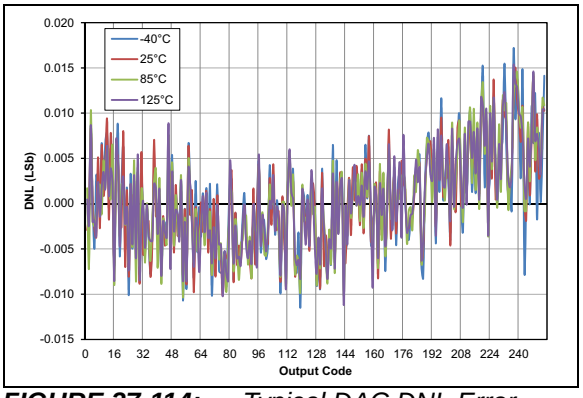



***FIGURE 37-114:*** *Typical DAC DNL Error, $V_{DD}$ = 5.0V, $V_{REF}$ = External 5V, PIC16F1773/6 Only.*

**Note:** Unless otherwise noted, VIN = 5V, FOSC = 300 kHz, CIN = 0.1 µF, TA = 25°C.

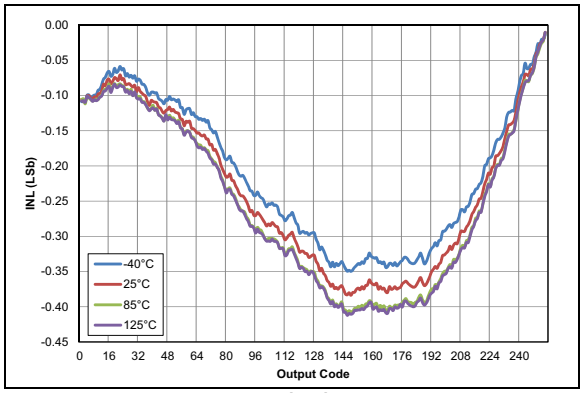



***FIGURE 37-115:*** *Typical DAC INL Error, VDD = 5.0V, VREF = External 5V, PIC16F1773/6 Only.*

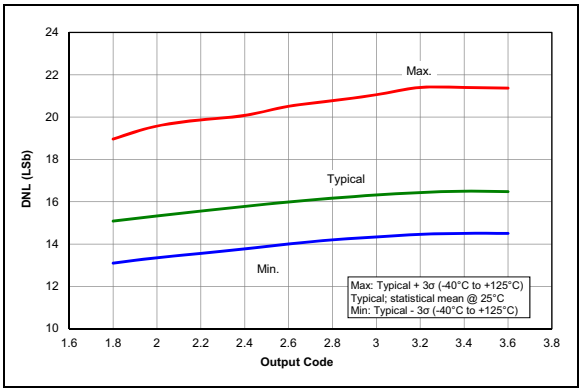



***FIGURE 37-116:*** *DAC INL Error, VDD = 3.0V, PIC16LF1773/6 Only.*

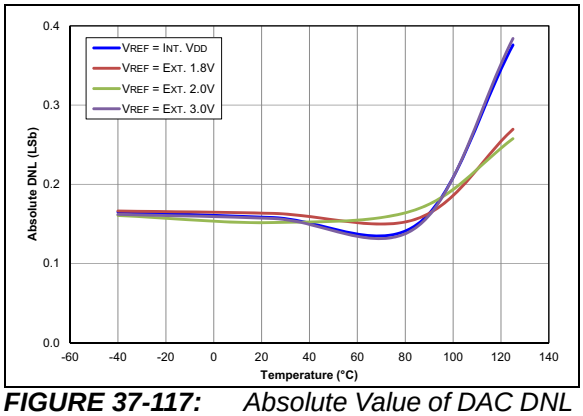



***FIGURE 37-117:*** *Absolute Value of DAC DNL Error, VDD = 3.0V, VREF = VDD.*

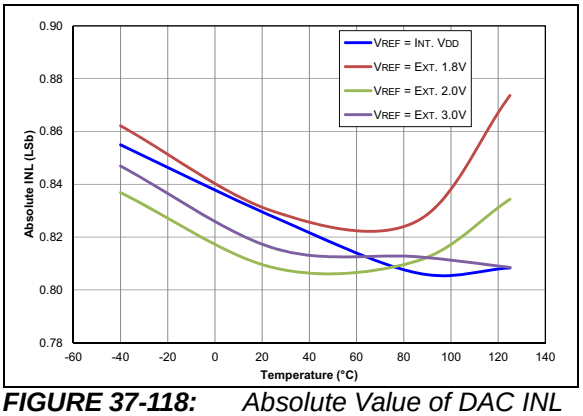



***FIGURE 37-118:*** *Absolute Value of DAC INL Error, VDD = 3.0V, VREF = VDD.*

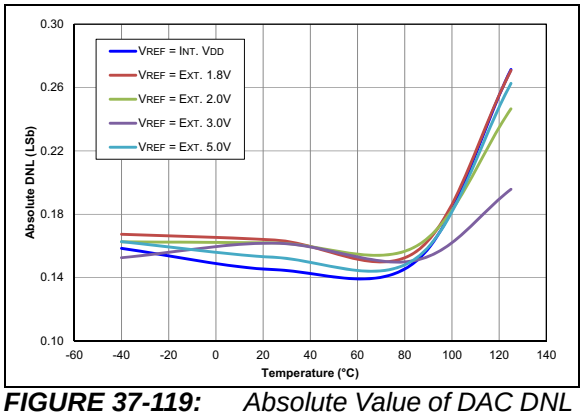



***FIGURE 37-119:*** *Absolute Value of DAC DNL Error, VDD = 5.0V, VREF = VDD, PIC16F1773/6 Only.*

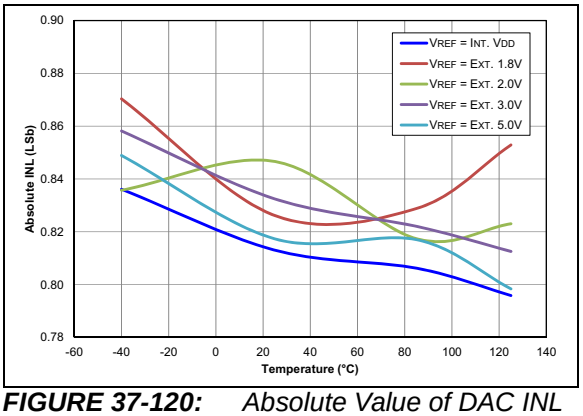



***FIGURE 37-120:*** *Absolute Value of DAC INL Error, VDD = 5.0V, VREF = VDD, PIC16F1773/6 Only.*

**Note:** Unless otherwise noted, VIN = 5V, FOSC = 300 kHz, CIN = 0.1 µF, TA = 25°C.

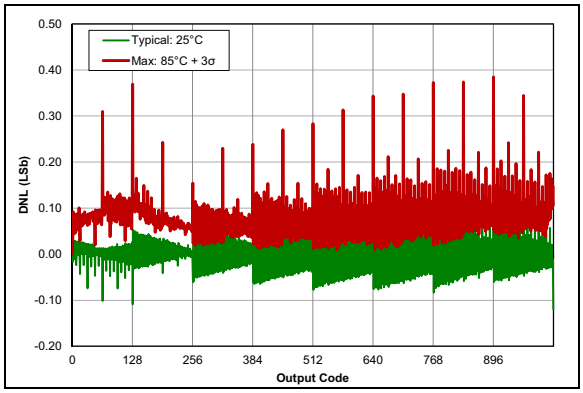



***FIGURE 37-121:*** *DAC DNL Error, VDD = 3.0V, VREF = External 3V.*

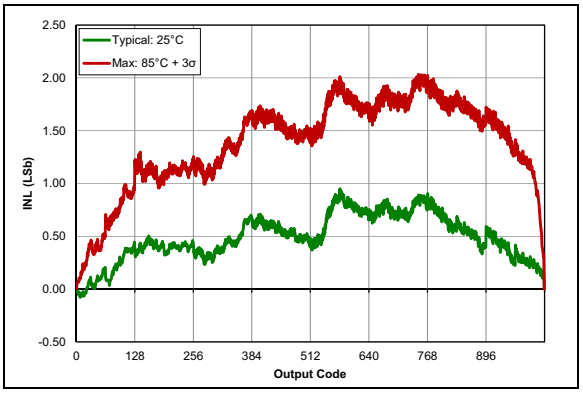



***FIGURE 37-122:*** *DAC INL Error, VDD = 3.0V, VREF = External 3V.*

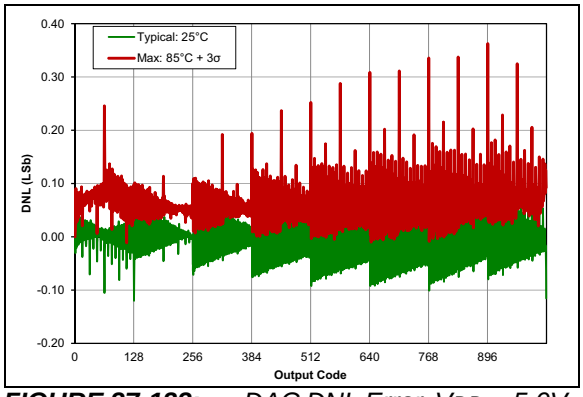



***FIGURE 37-123:*** *DAC DNL Error, VDD = 5.0V, VREF = External 5V, PIC16F1773/6 Only.*

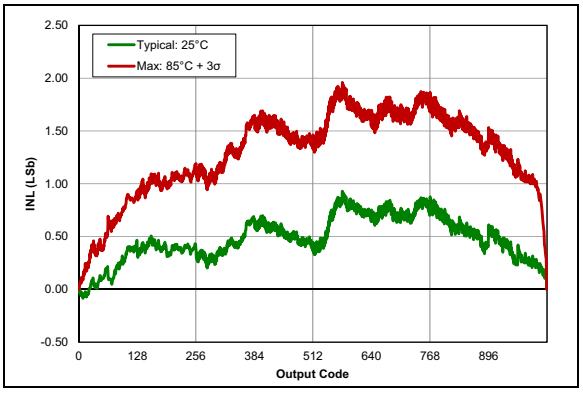



***FIGURE 37-124:*** *Typical DAC INL Error, VDD = 5.0V, VREF = External 5V, PIC16F1773/6 Only.*

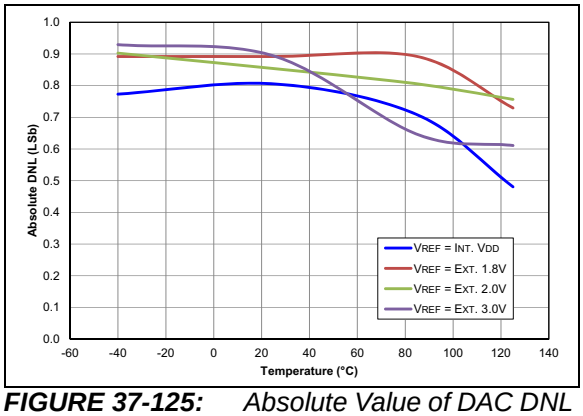



***FIGURE 37-125:*** *Absolute Value of DAC DNL Error, VDD = 3.0V, VREF = VDD.*

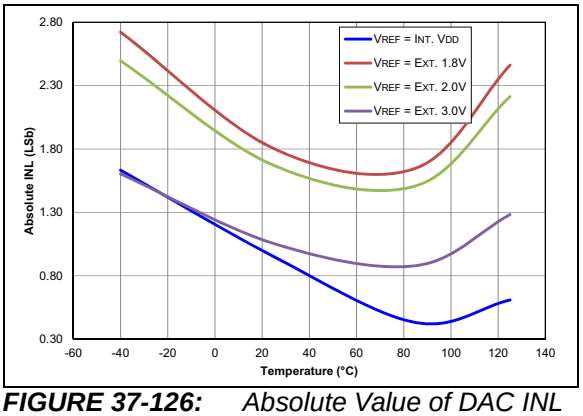



***FIGURE 37-126:*** *Absolute Value of DAC INL Error, VDD = 3.0V, VREF = VDD.*

**Note:** Unless otherwise noted, VIN = 5V, FOSC = 300 kHz, CIN = 0.1 µF, TA = 25°C.

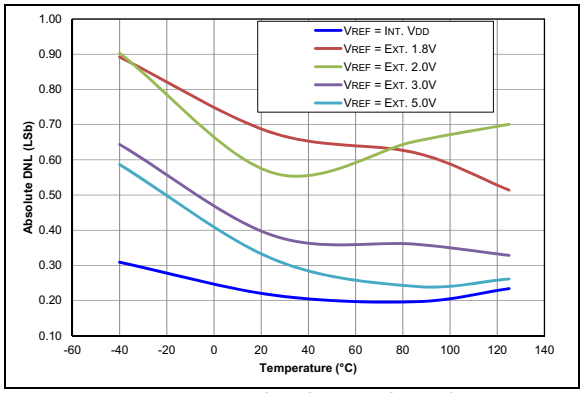



***FIGURE 37-127:*** *Absolute Value of DAC DNL Error, VDD = 5.0V, VREF = VDD, PIC16F1773/6 Only.*

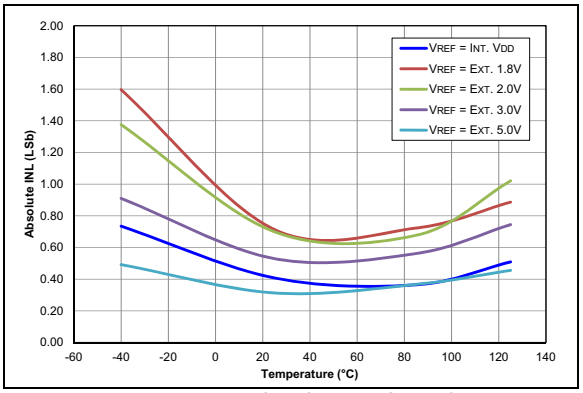



***FIGURE 37-128:*** *Absolute Value of DAC INL Error, VDD = 5.0V, VREF = VDD, PIC16F1773/6 Only.*

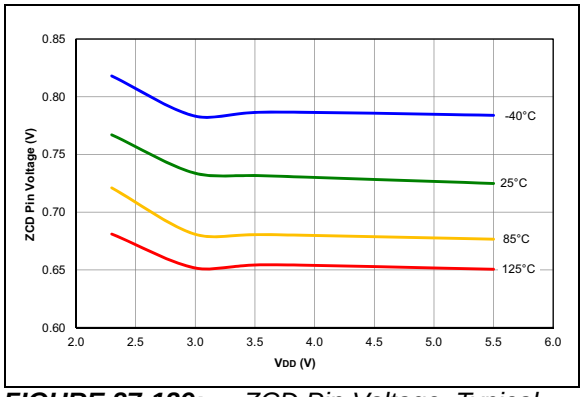



***FIGURE 37-129:*** *ZCD Pin Voltage, Typical Measured Values.*

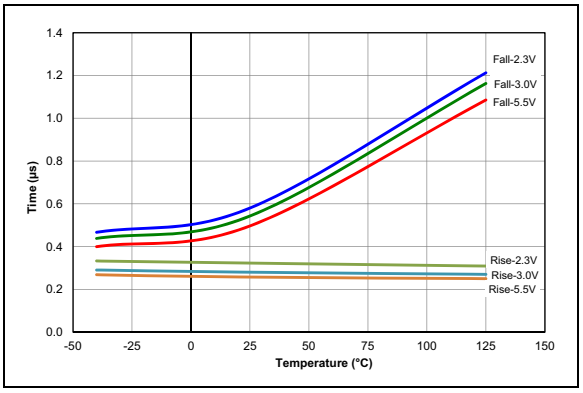



***FIGURE 37-130:*** *ZCD Response Time over Voltage, Typical Measured Values.*

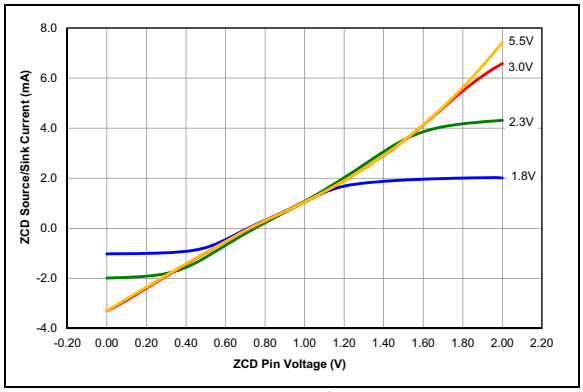



***FIGURE 37-131:*** *ZCD Pin Current over ZCD Pin Voltage, Typical Measured Values from -40°C to 125°C.*

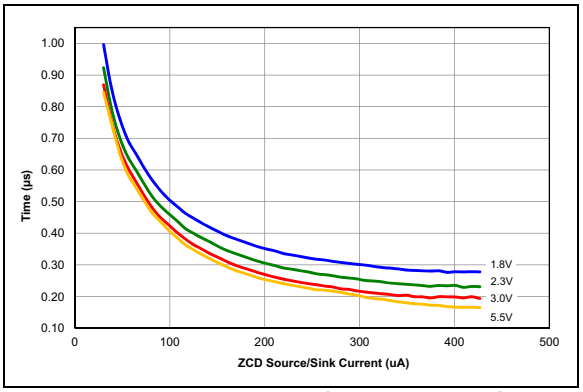



***FIGURE 37-132:*** *ZCD Pin Response Time over Current, Typical Measured Values from -40°C to 125°C.*

**Note:** Unless otherwise noted, VIN = 5V, FOSC = 300 kHz, CIN = 0.1 µF, TA = 25°C.

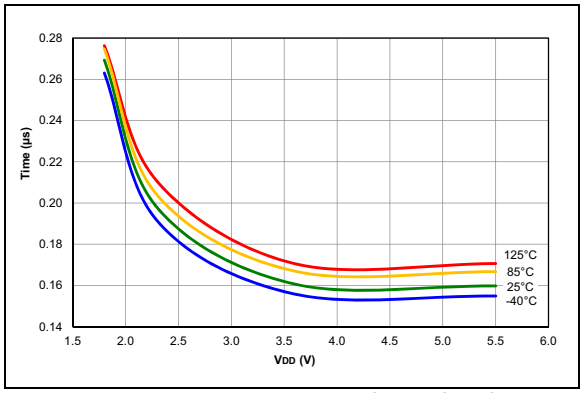



***FIGURE 37-133:*** *COG Dead Band Delay, DBR/DBF = 32, Typical Measured Values.*

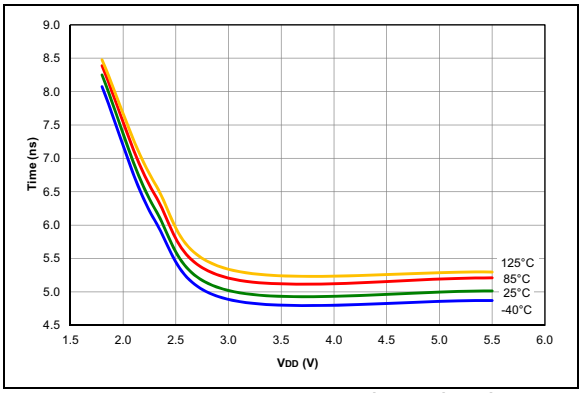



***FIGURE 37-134:*** *COG Dead Band Delay, DBR/DBF Delay per Step, Typical Measured Values.*

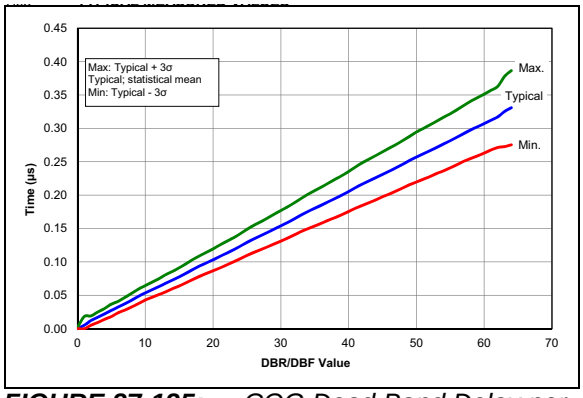



***FIGURE 37-135:*** *COG Dead Band Delay per Step, Typical Measured Values.*

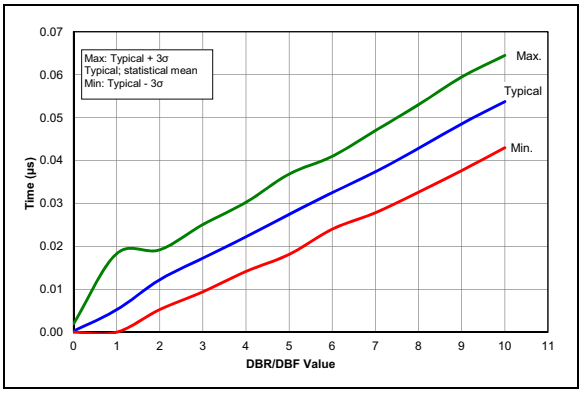



***FIGURE 37-136:*** *COG Dead Band Delay per Step, Zoomed to First 10 Codes, Typical Measured Values.*

## 38.0 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

### 38.1 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

## 38.2 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

## 38.3 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

## 38.4 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

## 38.5 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

## 38.6 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.

## 38.7 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 upgradeable 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.

## 38.8 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.

## 38.9 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™).

## 38.10 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.

## 38.11 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.

## 38.12 Third-Party Development Tools

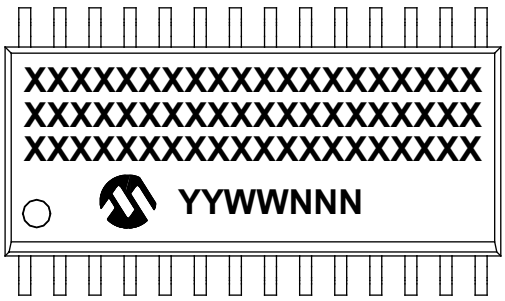
Microchip also offers a great collection of tools from third-party vendors. These tools are carefully selected to offer good value and unique functionality.

- 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®

## 39.0 PACKAGING INFORMATION

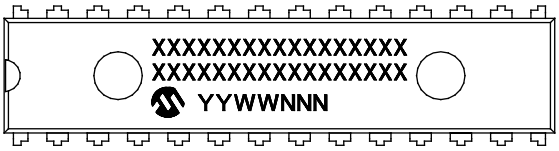
### 39.1 Package Marking Information

28-Lead SOIC (7.50 mm)

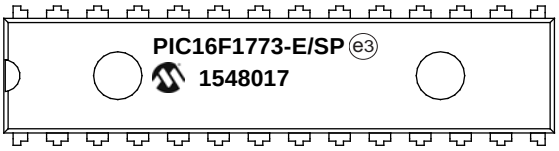



Example

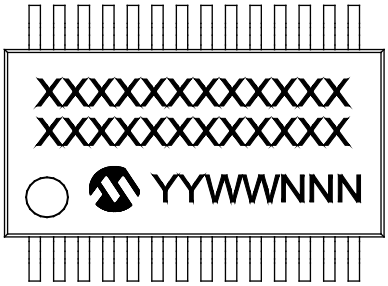

28-Lead SPDIP (.300")



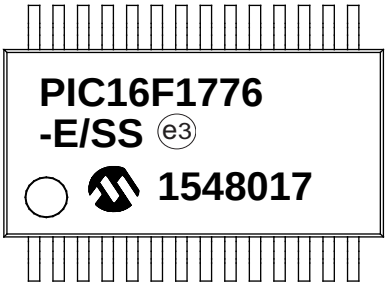

Example




28-Lead SSOP (5.30 mm)




Example

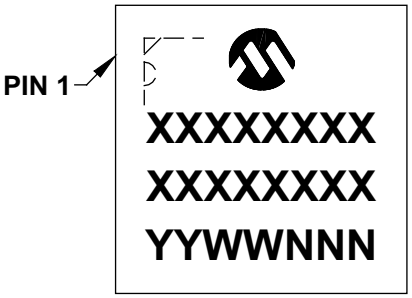



| | | |
|---|---|---|
| **Legend:** | XX...X | Customer-specific information |
| | Y | Year code (last digit of calendar year) |
| | YY | Year code (last 2 digits of calendar year) |
| | WW | Week code (week of January 1 is week '01') |
| | NNN | Alphanumeric traceability code |
| | (e3) | Pb-free JEDEC® 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. |

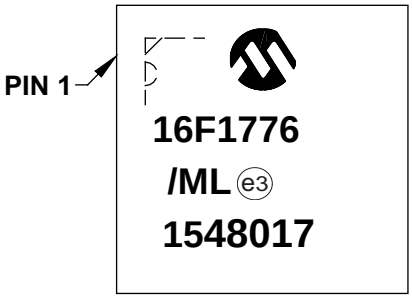
**Note**: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.

**Package Marking Information (Continued)**

28-Lead UQFN (6x6 mm)




Example

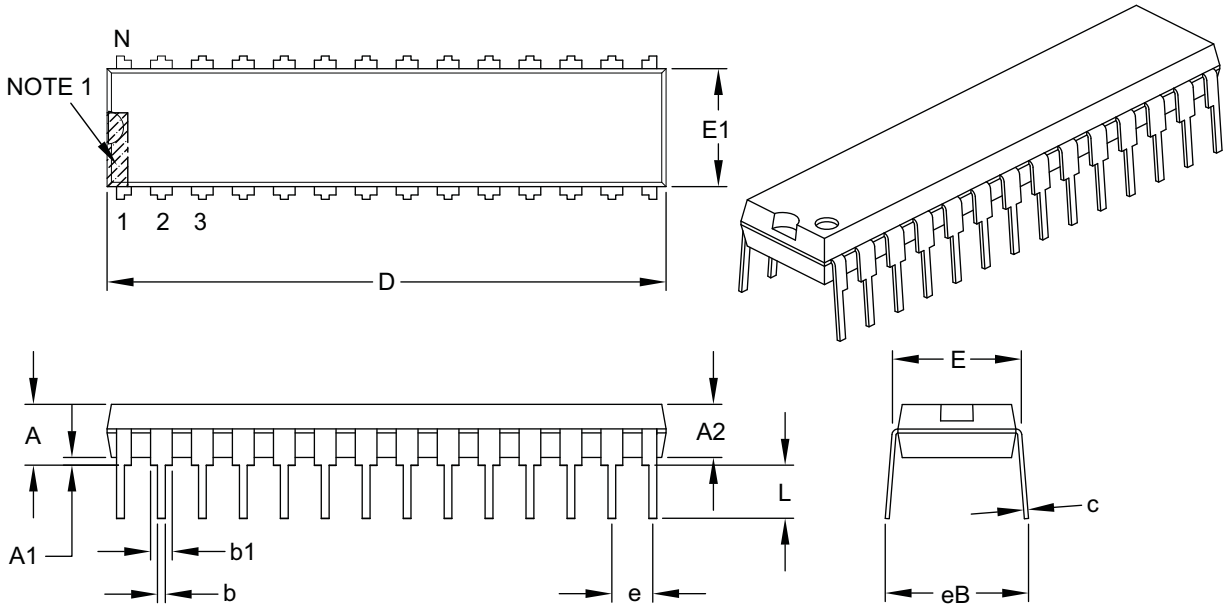


## 39.2 Package Details

The following sections give the technical details of the packages.

### 28-Lead Skinny Plastic Dual In-Line (SP) – 300 mil Body [SPDIP]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




| Units | | INCHES | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Pins | N | | 28 | |
| Pitch | e | | .100 BSC | |
| Top to Seating Plane | A | – | – | .200 |
| Molded Package Thickness | A2 | .120 | .135 | .150 |
| Base to Seating Plane | A1 | .015 | – | – |
| Shoulder to Shoulder Width | E | .290 | .310 | .335 |
| Molded Package Width | E1 | .240 | .285 | .295 |
| Overall Length | D | 1.345 | 1.365 | 1.400 |
| Tip to Seating Plane | L | .110 | .130 | .150 |
| Lead Thickness | c | .008 | .010 | .015 |
| Upper Lead Width | b1 | .040 | .050 | .070 |
| Lower Lead Width | b | .014 | .018 | .022 |
| Overall Row Spacing § | eB | – | – | .430 |

**Notes:**

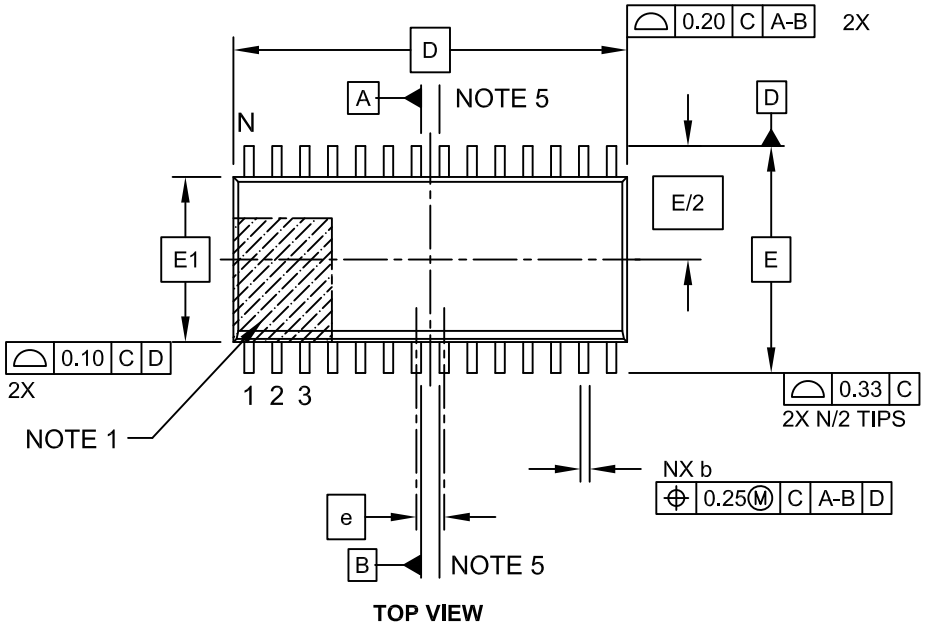
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

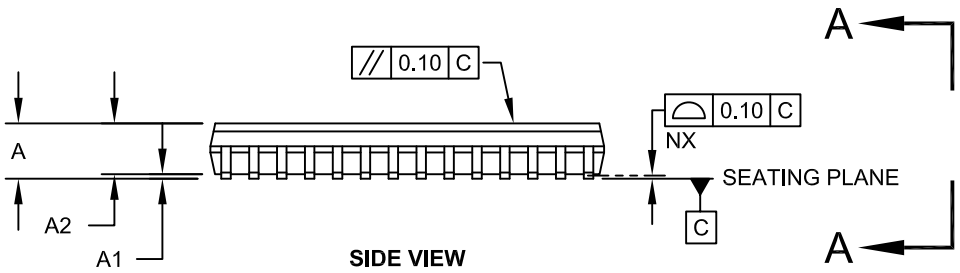
Microchip Technology Drawing C04-070B

### 28-Lead Plastic Small Outline (SO) - Wide, 7.50 mm Body [SOIC]

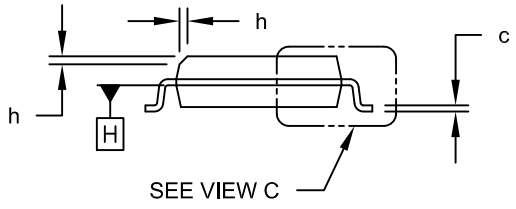
**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




**TOP VIEW**




**SIDE VIEW**

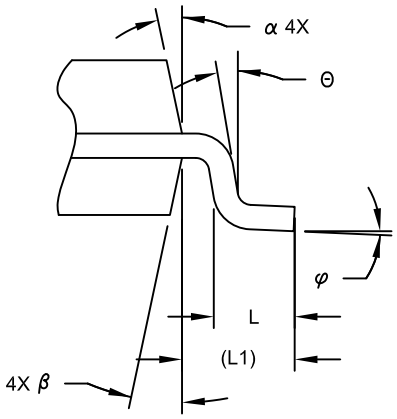



**VIEW A-A**

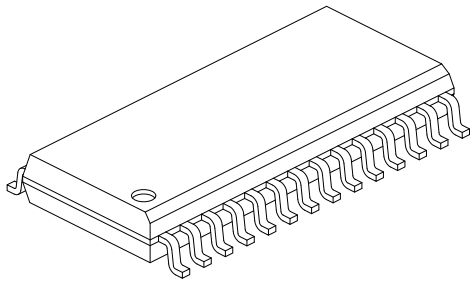
Microchip Technology Drawing C04-052C Sheet 1 of 2

## 28-Lead Plastic Small Outline (SO) - Wide, 7.50 mm Body [SOIC]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




VIEW C



| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Pins | N | | 28 | |
| Pitch | e | | 1.27 BSC | |
| Overall Height | A | - | - | 2.65 |
| Molded Package Thickness | A2 | 2.05 | - | - |
| Standoff § | A1 | 0.10 | - | 0.30 |
| Overall Width | E | | 10.30 BSC | |
| Molded Package Width | E1 | | 7.50 BSC | |
| Overall Length | D | | 17.90 BSC | |
| Chamfer (Optional) | h | 0.25 | - | 0.75 |
| Foot Length | L | 0.40 | - | 1.27 |
| Footprint | L1 | | 1.40 REF | |
| Lead Angle | Θ | 0° | - | - |
| Foot Angle | φ | 0° | - | 8° |
| Lead Thickness | c | 0.18 | - | 0.33 |
| Lead Width | b | 0.31 | - | 0.51 |
| Mold Draft Angle Top | α | 5° | - | 15° |
| Mold Draft Angle Bottom | β | 5° | - | 15° |

**Notes:**

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic
3. Dimension D does not include mold flash, protrusions or gate burrs, which shall not exceed 0.15 mm per end. Dimension E1 does not include interlead flash or protrusion, which shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M

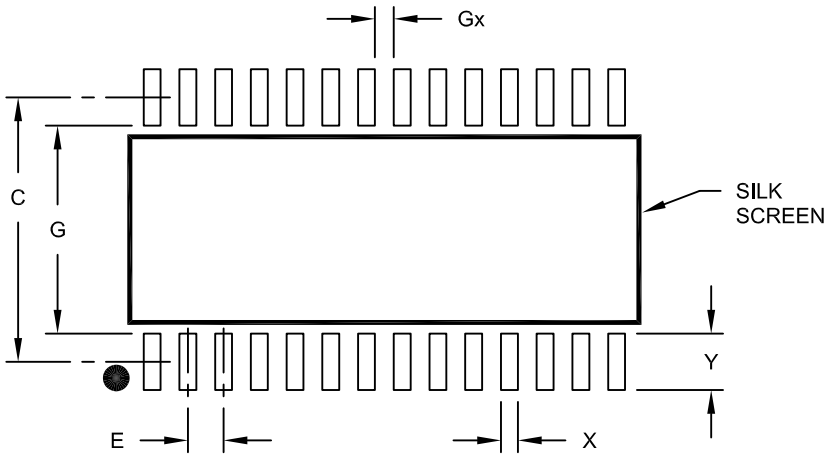
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

   REF: Reference Dimension, usually without tolerance, for information purposes only.
5. Datums A & B to be determined at Datum H.

Microchip Technology Drawing C04-052C Sheet 2 of 2

28-Lead Plastic Small Outline (SO) - Wide, 7.50 mm Body [SOIC]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




RECOMMENDED LAND PATTERN

| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Contact Pitch | E | | 1.27 BSC | |
| Contact Pad Spacing | C | | 9.40 | |
| Contact Pad Width (X28) | X | | | 0.60 |
| Contact Pad Length (X28) | Y | | | 2.00 |
| Distance Between Pads | Gx | 0.67 | | |
| Distance Between Pads | G | 7.40 | | |

Notes:

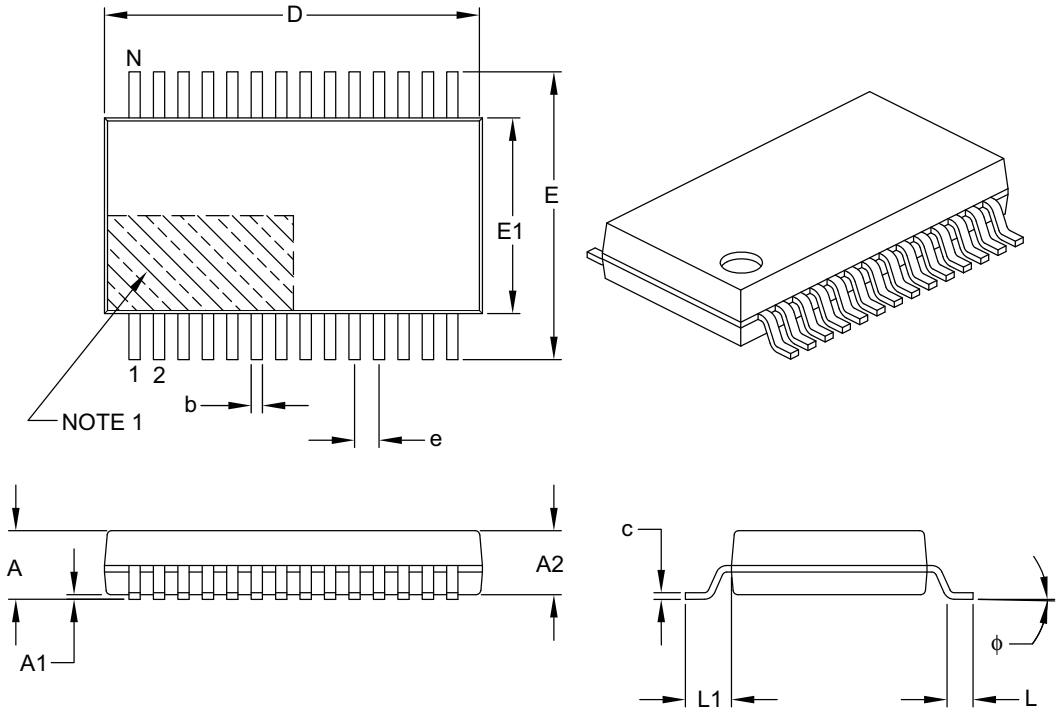
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2052A

### 28-Lead Plastic Shrink Small Outline (SS) – 5.30 mm Body [SSOP]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Pins | N | | 28 | |
| Pitch | e | | 0.65 BSC | |
| Overall Height | A | – | – | 2.00 |
| Molded Package Thickness | A2 | 1.65 | 1.75 | 1.85 |
| Standoff | A1 | 0.05 | – | – |
| Overall Width | E | 7.40 | 7.80 | 8.20 |
| Molded Package Width | E1 | 5.00 | 5.30 | 5.60 |
| Overall Length | D | 9.90 | 10.20 | 10.50 |
| Foot Length | L | 0.55 | 0.75 | 0.95 |
| Footprint | L1 | | 1.25 REF | |
| Lead Thickness | c | 0.09 | – | 0.25 |
| Foot Angle | φ | 0° | 4° | 8° |
| Lead Width | b | 0.22 | – | 0.38 |

**Notes:**

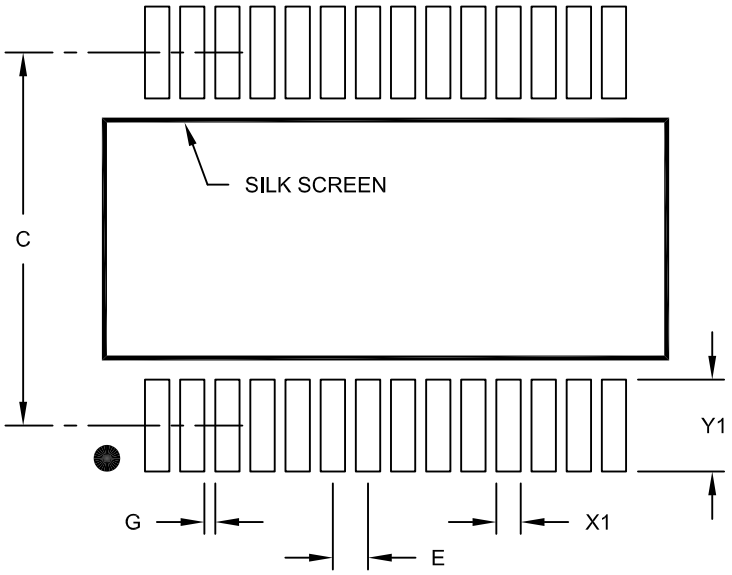
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.20 mm per side.
3. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-073B

28-Lead Plastic Shrink Small Outline (SS) - 5.30 mm Body [SSOP]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




RECOMMENDED LAND PATTERN

| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Contact Pitch | E | | 0.65 BSC | |
| Contact Pad Spacing | C | | 7.20 | |
| Contact Pad Width (X28) | X1 | | | 0.45 |
| Contact Pad Length (X28) | Y1 | | | 1.75 |
| Distance Between Pads | G | 0.20 | | |

Notes:

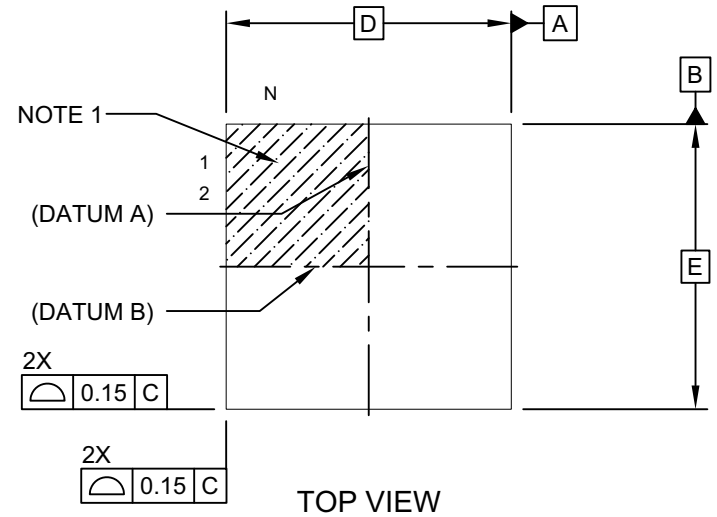
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

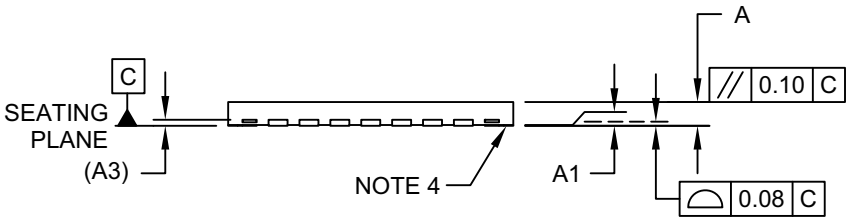
Microchip Technology Drawing No. C04-2073A

**28-Lead Plastic Quad Flat, No Lead Package (MX) - 6x6x0.5mm Body [UQFN]**
**Ultra-Thin with 0.40 x 0.60 mm Terminal Width/Length and Corner Anchors**

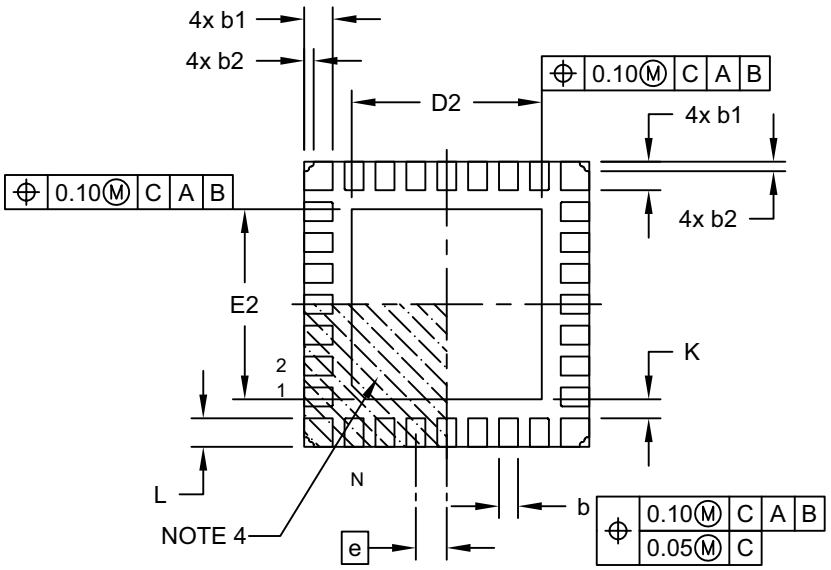
| **Note:** | For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging |
|---|---|




TOP VIEW




SIDE VIEW

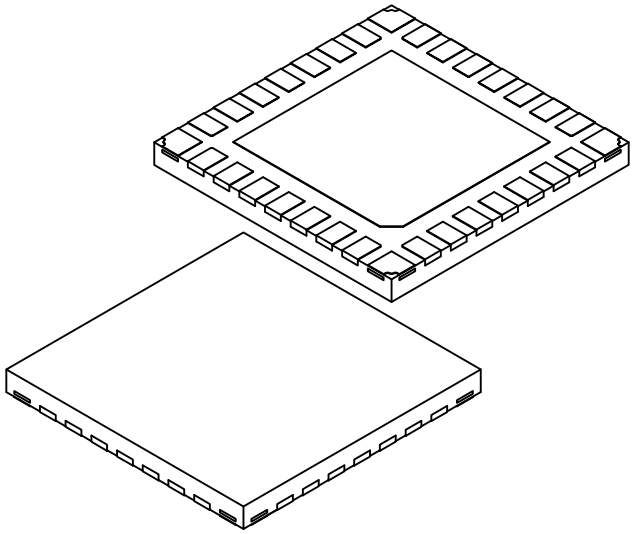



BOTTOM VIEW

Microchip Technology Drawing C04-0209 Rev C Sheet 1 of 2

### 28-Lead Plastic Quad Flat, No Lead Package (MX) - 6x6x0.5mm Body [UQFN] Ultra-Thin with 0.40 x 0.60 mm Terminal Width/Length and Corner Anchors

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Pins | N | | 28 | |
| Pitch | e | | 0.65 BSC | |
| Overall Height | A | 0.40 | 0.50 | 0.60 |
| Standoff | A1 | 0.00 | 0.02 | 0.05 |
| Terminal Thickness | (A3) | | 0.127 REF | |
| Overall Width | E | | 6.00 BSC | |
| Exposed Pad Width | E2 | | 4.00 | |
| Overall Length | D | | 6.00 BSC | |
| Exposed Pad Length | D2 | | 4.00 | |
| Terminal Width | b | 0.35 | 0.40 | 0.45 |
| Corner Pad | b1 | 0.55 | 0.60 | 0.65 |
| Corner Pad, Metal Free Zone | b2 | 0.15 | 0.20 | 0.25 |
| Terminal Length | L | 0.55 | 0.60 | 0.65 |
| Terminal-to-Exposed Pad | K | 0.20 | - | - |

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated
3. Dimensioning and tolerancing per ASME Y14.5M

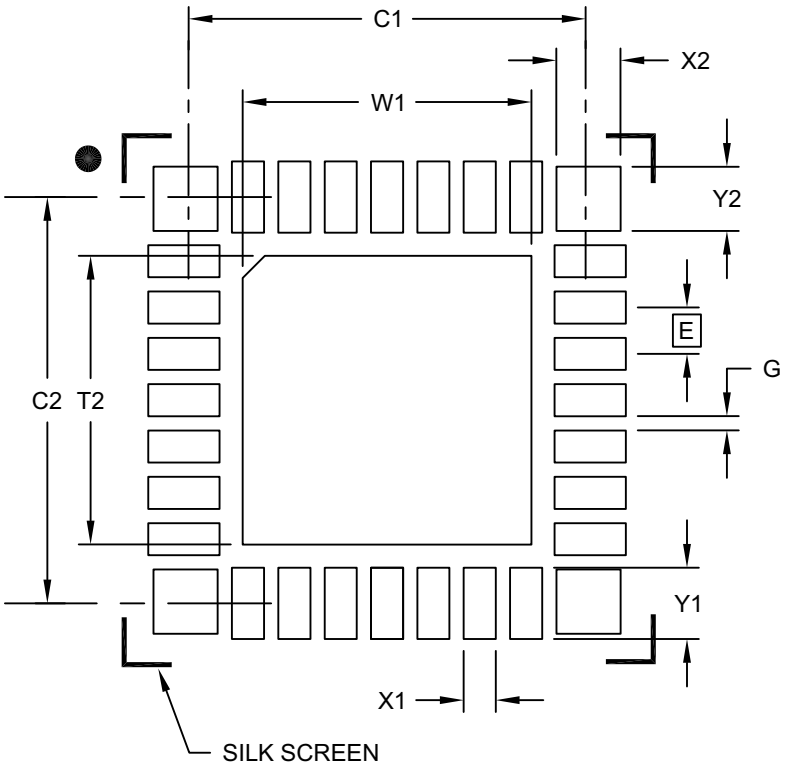
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

   REF: Reference Dimension, usually without tolerance, for information purposes only.
4. Outermost portions of corner structures may vary slightly.

Microchip Technology Drawing C04-0209 Rev C Sheet 2 of 2

### 28-Lead Plastic Quad Flat, No Lead Package (MX) - 6x6 mm Body [UQFN] With 0.60mm Contact Length And Corner Anchors

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




RECOMMENDED LAND PATTERN

| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Contact Pitch | E | | 0.65 BSC | |
| Optional Center Pad Width | W1 | | | 4.05 |
| Optional Center Pad Length | T2 | | | 4.05 |
| Contact Pad Spacing | C1 | | 5.70 | |
| Contact Pad Spacing | C2 | | 5.70 | |
| Contact Pad Width (X28) | X1 | | | 0.45 |
| Contact Pad Length (X28) | Y1 | | | 1.00 |
| Corner Pad Width (X4) | X2 | | | 0.90 |
| Corner Pad Length (X4) | Y2 | | | 0.90 |
| 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. C04-2209B

## APPENDIX A: DATA SHEET REVISION HISTORY

### Revision A (09/2015)

Initial release of this document.

### Revision B (10/2016)

Added characterization graphs.

Updated Figures 23-3, 23-8, 23-9, 23-10, 23-11, 23-12, and 23-13; Registers 7-5, 7-11, 11-4, 24-3, 27-11, and 32-4; Tables 3, 1-2, 3-5, 3-6, 3-9, 3-12, 12-1, 18-1, 24-4, 25-5, 27-5, 28-1, 30-4, 36-1, 36-2, 36-6, 36-20 and 36-25.

Updated the Cover page.

Section 20.5 rewritten. Added Characterization Data.

## THE MICROCHIP WEBSITE

Microchip provides online support via our website at www.microchip.com. This website is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the website contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

## CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip website at www.microchip.com. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

## CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or Field Application Engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

**Technical support is available through the website at: http://www.microchip.com/support**

## PRODUCT IDENTIFICATION SYSTEM

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

| PART NO. | [X](1) | - | X | /XX | XXX |
|---|---|---|---|---|---|
| Device | Tape and Reel Option | | Temperature Range | Package | Pattern |

| | | |
|---|---|---|
| **Device:** | PIC16F1773, PIC16LF1773, PIC16F1776, PIC16LF1776, | |
| **Tape and Reel Option:** | Blank<br>T | = Standard packaging (tube or tray)<br>= Tape and Reel(1) |
| **Temperature Range:** | I<br>E | = -40°C to +85°C (Industrial)<br>= -40°C to +125°C (Extended) |
| **Package:(2)** | ML<br>SP<br>SO<br>SS | = UQFN<br>= SPDIP<br>= SOIC<br>= SSOP |
| **Pattern:** | QTP, SQTP, Code or Special Requirements (blank otherwise) | |

**Examples:**

a) PIC16LF1773- I/P
Industrial temperature
PDIP package

b) PIC16F1776- E/SS
Extended temperature,
SSOP package

**Note 1:** Tape and Reel identifier only appears in the catalog part number description. This identifier is used for ordering purposes and is not printed on the device package. Check with your Microchip Sales Office for package availability with the Tape and Reel option.

**2:** Small form-factor packaging options may be available. Please check www.microchip.com/packaging for small-form factor package availability, or contact your local Sales Office.

**Note the following details of the code protection feature on Microchip devices:**

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.



*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**



ISBN: 978-1-5224-1017-1

# Worldwide Sales and Service

## AMERICAS

**Corporate Office**
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
http://www.microchip.com/support
Web Address:
www.microchip.com

**Atlanta**
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

**Austin, TX**
Tel: 512-257-3370

**Boston**
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

**Chicago**
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

**Cleveland**
Independence, OH
Tel: 216-447-0464
Fax: 216-447-0643

**Dallas**
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

**Detroit**
Novi, MI
Tel: 248-848-4000

**Houston, TX**
Tel: 281-894-5983

**Indianapolis**
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453

**Los Angeles**
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

**New York, NY**
Tel: 631-435-6000

**San Jose, CA**
Tel: 408-735-9110

**Canada - Toronto**
Tel: 905-695-1980
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: 86-21-5407-5533
Fax: 86-21-5407-5066

**China - Shenyang**
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

**China - Shenzhen**
Tel: 86-755-8864-2200
Fax: 86-755-8203-1760

**China - Wuhan**
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

**China - Xian**
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

## ASIA/PACIFIC

**China - Xiamen**
Tel: 86-592-2388138
Fax: 86-592-2388130

**China - Zhuhai**
Tel: 86-756-3210040
Fax: 86-756-3210049

**India - Bangalore**
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

**India - New Delhi**
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

**India - Pune**
Tel: 91-20-3019-1500

**Japan - Osaka**
Tel: 81-6-6152-7160
Fax: 81-6-6152-9310

**Japan - Tokyo**
Tel: 81-3-6880- 3770
Fax: 81-3-6880-3771

**Korea - Daegu**
Tel: 82-53-744-4301
Fax: 82-53-744-4302

**Korea - Seoul**
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

**Malaysia - Kuala Lumpur**
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

**Malaysia - Penang**
Tel: 60-4-227-8870
Fax: 60-4-227-4068

**Philippines - Manila**
Tel: 63-2-634-9065
Fax: 63-2-634-9069

**Singapore**
Tel: 65-6334-8870
Fax: 65-6334-8850

**Taiwan - Hsin Chu**
Tel: 886-3-5778-366
Fax: 886-3-5770-955

**Taiwan - Kaohsiung**
Tel: 886-7-213-7828

**Taiwan - Taipei**
Tel: 886-2-2508-8600
Fax: 886-2-2508-0102

**Thailand - Bangkok**
Tel: 66-2-694-1351
Fax: 66-2-694-1350

## EUROPE

**Austria - Wels**
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

**Denmark - Copenhagen**
Tel: 45-4450-2828
Fax: 45-4485-2829

**France - Paris**
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

**Germany - Dusseldorf**
Tel: 49-2129-3766400

**Germany - Karlsruhe**
Tel: 49-721-625370

**Germany - Munich**
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

**Italy - Milan**
Tel: 39-0331-742611
Fax: 39-0331-466781

**Italy - Venice**
Tel: 39-049-7625286

**Netherlands - Drunen**
Tel: 31-416-690399
Fax: 31-416-690340

**Poland - Warsaw**
Tel: 48-22-3325737

**Spain - Madrid**
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

**Sweden - Stockholm**
Tel: 46-8-5090-4654

**UK - Wokingham**
Tel: 44-118-921-5800
Fax: 44-118-921-5820

06/23/16

# Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

Microchip:

PIC16LF1776-I/SP PIC16LF1776T-I/MX PIC16LF1773-E/SO PIC16LF1776T-I/SO PIC16LF1773-E/SS PIC16LF1773T-I/SS PIC16LF1776-E/SS PIC16LF1776-I/MX PIC16LF1773-E/SP PIC16LF1776T-I/SS PIC16LF1773-I/SO PIC16LF1776-E/MX PIC16LF1773-E/MX PIC16LF1776-E/SO PIC16LF1773T-I/SO PIC16LF1773T-I/MX PIC16LF1773-I/MX PIC16LF1773-I/SP PIC16LF1776-I/SS PIC16LF1776-I/SO PIC16LF1776-E/SP PIC16LF1773-I/SS PIC16LF1777T-I/ML PIC16F1777T-I/ML PIC16F1777-E/ML PIC16LF1777-E/ML PIC16F1777-I/ML PIC16LF1777-I/ML PIC16F1773-E/MX PIC16F1773-E/SO PIC16F1773-E/SP PIC16F1773-E/SS PIC16F1773-I/MX PIC16F1773-I/SO PIC16LF1777T-I/MV PIC16LF1777T-I/PT PIC16LF1777-E/MV PIC16LF1777-E/P PIC16LF1777-E/PT PIC16LF1777-I/MV PIC16LF1777-I/P PIC16LF1777-I/PT PIC16F1777-I/MV PIC16F1777-I/P PIC16F1777-I/PT PIC16F1777T-I/MV PIC16F1777T-I/PT PIC16F1776-I/SS PIC16F1776T-I/MX PIC16F1776T-I/SO PIC16F1776T-I/SS PIC16F1777-E/MV PIC16F1777-E/P PIC16F1776-E/SO PIC16F1776-E/SP PIC16F1776-E/SS PIC16F1776-I/MX PIC16F1776-I/SO PIC16F1776-I/SP PIC16F1773-I/SP PIC16F1773-I/SS PIC16F1773T-I/MX PIC16F1773T-I/SO PIC16F1773T-I/SS PIC16F1776-E/MX