

---

## Full-Featured 28-Pin Microcontrollers

---

### Description

PIC16(L)F15354/55 microcontrollers feature Analog, Core Independent Peripherals and Communication Peripherals, combined with eXtreme Low-Power (XLP) technology for a wide range of general purpose and low-power applications.

The devices feature multiple PWMs, multiple communication, temperature sensor, and memory features like Memory Access Partition (MAP) to support customers in data protection and bootloader applications, and Device Information Area (DIA) which stores factory calibration values to help improve temperature sensor accuracy.

### Core Features

- C Compiler Optimized RISC Architecture
- Operating Speed:
  - DC – 32 MHz clock input
  - 125 ns minimum instruction cycle
- Interrupt Capability
- 16-Level Deep Hardware Stack
- Timers:
  - 8-bit Timer2 with Hardware Limit Timer (HLT)
  - 16-bit Timer0/1
- Low-Current Power-on Reset (POR)
- Configurable Power-up Timer (PWRTE)
- Brown-out Reset (BOR)
- Low-Power BOR (LPBOR) Option
- Windowed Watchdog Timer (WWDT):
  - Variable prescaler selection
  - Variable window size selection
  - All sources configurable in hardware or software
- Programmable Code Protection

### Memory

- Up to 14 KB Flash Program Memory
- Up to 1024 Bytes Data SRAM
- Direct, Indirect and Relative Addressing modes
- Memory Access Partition (MAP):
  - Write protect
  - Customizable Partition
- Device Information Area (DIA)
- Device Configuration Information (DCI)
- High-Endurance Flash (HEF)
  - Last 128 words of Program Flash Memory

### Operating Characteristics

- Operating Voltage Range:
  - 1.8V to 3.6V (PIC16LF15354/55)
  - 2.3V to 5.5V (PIC16F15354/55)
- Temperature Range:
  - Industrial: -40°C to 85°C
  - Extended: -40°C to 125°C

### Power-Saving Functionality

- DOZE mode: Ability to Run the CPU Core Slower than the System Clock
- IDLE mode: Ability to halt CPU Core while Internal Peripherals Continue Operating
- SLEEP mode: Lowest Power Consumption
- Peripheral Module Disable (PMD):
  - Ability to disable hardware module to minimize active power consumption of unused peripherals

### 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  $\mu$ A @ 32 kHz, 1.8V, typical
  - 32  $\mu$ A/MHz @ 1.8V, typical

### Digital Peripherals

- Four Configurable Logic Cells (CLC):
  - Integrated combinational and sequential logic
- Complementary Waveform Generator (CWG):
  - Rising and falling edge dead-band control
  - Full-bridge, half-bridge, 1-channel drive
  - Multiple signal sources
- Two Capture/Compare/PWM (CCP) module:
  - 16-bit resolution for Capture/Compare modes
  - 10-bit resolution for PWM mode
- Four 10-Bit PWMs
- Numerically Controlled Oscillator (NCO):
  - Generates true linear frequency control and increased frequency resolution
  - Input Clock:  $0 \text{ Hz} < F_{\text{NCO}} < 32 \text{ MHz}$
  - Resolution:  $F_{\text{NCO}}/2^{20}$
- Two EUSART, RS-232, RS-485, LIN compatible
- Two SPI
- Two I<sup>2</sup>C, SMBus, PMBus™ compatible

## Digital Peripherals (Cont.)

- I/O Pins:
  - Individually programmable pull-ups
  - Slew rate control
  - Interrupt-on-change with edge-select
  - Input level selection control (ST or TTL)
  - Digital open-drain enable
- Peripheral Pin Select (PPS):
  - Enables pin mapping of digital I/O

## Analog Peripherals

- Analog-to-Digital Converter (ADC):
  - 10-bit with up to 43 external channels
  - Operates in Sleep
- Two Comparators:
  - FVR, DAC and external input pin available on inverting and noninverting input
  - Software selectable hysteresis
  - Outputs available internally to other modules, or externally through PPS
- 5-Bit Digital-to-Analog Converter (DAC):
  - 5-bit resolution, rail-to-rail
  - Positive Reference Selection
  - Unbuffered I/O pin output
  - Internal connections to ADCs and comparators
- Voltage Reference:
  - Fixed Voltage Reference with 1.024V, 2.048V and 4.096V output levels
- Zero-Cross Detect module:
  - AC high voltage zero-crossing detection for simplifying TRIAC control
  - Synchronized switching control and timing

## Flexible Oscillator Structure

- High-Precision Internal Oscillator:
  - Software selectable frequency range up to 32 MHz,  $\pm 1\%$  typical
- x2/x4 PLL with Internal and External Sources
- Low-Power Internal 32 kHz Oscillator (LFINTOSC)
- External 32 kHz Crystal Oscillator (SOSC)
- External Oscillator Block with:
  - Three crystal/resonator modes up to 20 MHz
  - Three external clock modes up to 32 MHz
- Fail-Safe Clock Monitor:
  - Allows for safe shutdown if primary clock stops
- Oscillator Start-up Timer (OST):
  - Ensures stability of crystal oscillator resources

# PIC16(L)F15354/55

**TABLE 1: PIC16(L)F153XX FAMILY TYPES**

Device	Data Sheet Index	Program Flash Memory (KW)	Program Flash Memory (KB)	Storage Area Flash (B)	Data SRAM (bytes)	I/O Pins	10-bit ADC	5-bit DAC	Comparator	8-bit/ (with HLT) Timer	16-bit Timer	Window Watchdog Timer	CCP/10-bit PWM	CWG	NCO	CLC	Zero-Cross Detect	Temperature Indicator	Memory Access Partition	Device Information Area	EUSART/ I <sup>2</sup> C-SPI	Peripheral Pin Select	Peripheral Module Disable	Debug (1)
PIC16(L)F15313	(C)	2	3.5	224	256	6	5	1	1	1	2	Y	2/4	1	1	4	Y	Y	Y	Y	1/1	Y	Y	I
PIC16(L)F15323	(C)	2	3.5	224	256	12	11	1	2	1	2	Y	2/4	1	1	4	Y	Y	Y	Y	1/1	Y	Y	I
PIC16(L)F15324	(D)	4	7	224	512	12	11	1	2	1	2	Y	2/4	1	1	4	Y	Y	Y	Y	2/1	Y	Y	I
PIC16(L)F15325	(B)	8	14	224	1024	12	11	1	2	1	2	Y	2/4	1	1	4	Y	Y	Y	Y	2/1	Y	Y	I
PIC16(L)F15344	(D)	4	7	224	512	18	17	1	2	1	2	Y	2/4	1	1	4	Y	Y	Y	Y	2/1	Y	Y	I
PIC16(L)F15345	(B)	8	14	224	1024	18	17	1	2	1	2	Y	2/4	1	1	4	Y	Y	Y	Y	2/1	Y	Y	I
PIC16(L)F15354	(A)	4	7	224	512	25	24	1	2	1	2	Y	2/4	1	1	4	Y	Y	Y	Y	2/2	Y	Y	I
PIC16(L)F15355	(A)	8	14	224	1024	25	24	1	2	1	2	Y	2/4	1	1	4	Y	Y	Y	Y	2/2	Y	Y	I
PIC16(L)F15356	(E)	16	28	224	2048	25	24	1	2	1	2	Y	2/4	1	1	4	Y	Y	Y	Y	2/2	Y	Y	I
PIC16(L)F15375	(E)	8	14	224	1024	36	35	1	2	1	2	Y	2/4	1	1	4	Y	Y	Y	Y	2/2	Y	Y	I
PIC16(L)F15376	(E)	16	28	224	2048	36	35	1	2	1	2	Y	2/4	1	1	4	Y	Y	Y	Y	2/2	Y	Y	I
PIC16(L)F15385	(E)	8	14	224	1024	44	43	1	2	1	2	Y	2/4	1	1	4	Y	Y	Y	Y	2/2	Y	Y	I
PIC16(L)F15386	(E)	16	28	224	2048	44	43	1	2	1	2	Y	2/4	1	1	4	Y	Y	Y	Y	2/2	Y	Y	I

**Note 1:** I - Debugging integrated on chip.

**Data Sheet Index:**

- A: DS40001853**      [PIC16\(L\)F15354/5 Data Sheet, 28-Pin](#)
- B: DS40001865**      [PIC16\(L\)F15325/45 Data Sheet, 14/20-Pin](#)
- C: DS40001897**      [PIC16\(L\)F15313/23 Data Sheet, 8/14-Pin](#)
- D: DS40001889**      [PIC16\(L\)F15324/44 Data Sheet, 14/20-Pin](#)
- E: DS40001866**      [PIC16\(L\)F15356/75/76/85/86 Data Sheet, 28/40/48-Pin](#)

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

# PIC16(L)F15354/55

---

---

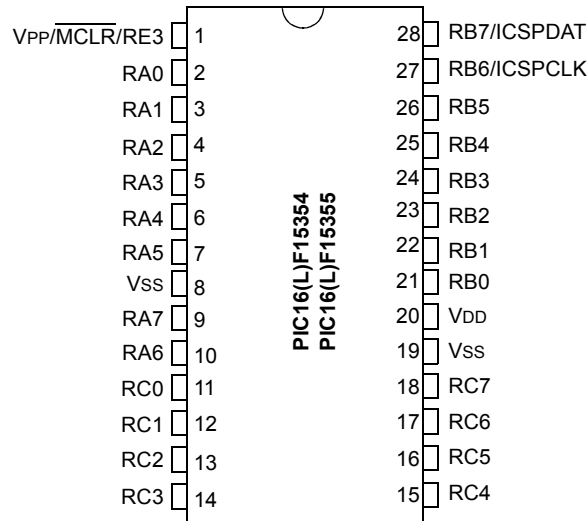
**TABLE 2: PACKAGES**

Device	SPDIP	SOIC	SSOP	UQFN (4x4)	UQFN (6x6)
PIC16(L)F15354	•	•	•	•	•
PIC16(L)F15355	•	•	•	•	•



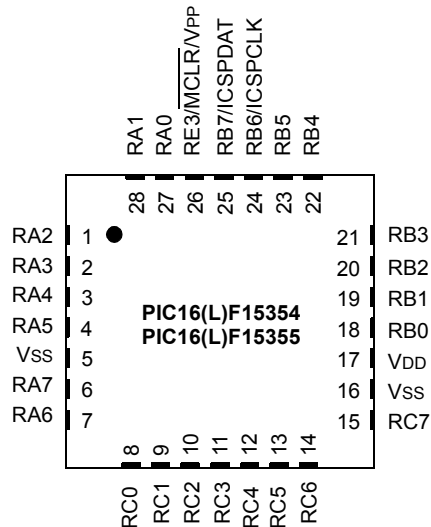
## PIN DIAGRAMS

28-PIN PDIP, SOIC, SSOP



- Note 1:** See [Table 3](#) for location of all peripheral functions.
- Note 2:** All VDD and all VSS pins must be connected at the circuit board level.

28-PIN UQFN (4x4), UQFN (6x6)



- Note 1:** See [Table 3](#) for location of all peripheral functions.
- Note 2:** All VDD and all VSS pins must be connected at the circuit board level. Allowing one or more VSS or VDD pins to float may result in degraded electrical performance or non-functionality.
- Note 3:** The bottom pad of the QFN/UQFN package should be connected to VSS at the circuit board level.

## PIN ALLOCATION TABLES

TABLE 3: 28-PIN ALLOCATION TABLE (PIC16(L)F15354, PIC16(L)F15355)

I/O <sup>(2)</sup>	28-Pin PDIP/SOIC/SSOP	28-Pin UQFN	ADC	Reference	Comparator	NCO	DAC	Timers	CCP	PWM	CWG	MSSP	ZCD	EUSART	CLC	CLKR	Interrupt	Pull-up	Basic
RA0	2	27	ANA0	—	C1IN0- C2IN0-	—	—	—	—	—	—	—	—	—	CLCIN0 <sup>(1)</sup>	—	IOCA0	Y	—
RA1	3	28	ANA1	—	C1IN1- C2IN1-	—	—	—	—	—	—	—	—	—	CLCIN1 <sup>(1)</sup>	—	IOCA1	Y	—
RA2	4	1	ANA2	—	C1IN0+ C2IN0+	—	DAC1OUT1	—	—	—	—	—	—	—	—	—	IOCA2	Y	—
RA3	5	2	ANA3	VREF+	C1IN1+	—	DAC1REF+	—	—	—	—	—	—	—	—	—	IOCA3	Y	—
RA4	6	3	ANA4	—	—	—	—	TOCKI	—	—	—	—	—	—	—	—	IOCA4	Y	—
RA5	7	4	ANA5	—	—	—	—	—	—	—	—	SS1 <sup>(1)</sup>	—	—	—	—	IOCA5	Y	—
RA6	10	7	ANA6	—	—	—	—	—	—	—	—	—	—	—	—	—	IOCA6	Y	CLKOUT OSC2
RA7	9	6	ANA7	—	—	—	—	—	—	—	—	—	—	—	—	—	IOCA7	Y	CLKIN OSC1
RB0	21	18	ANB0	—	C2IN1+	—	—	—	—	—	CWG1IN <sup>(1)</sup>	SS2 <sup>(1)</sup>	ZCD1	—	—	—	INT <sup>(1)</sup> IOCB0	Y	—
RB1	22	19	ANB1	—	C1IN3- C2IN3-	—	—	—	—	—	—	SCK2, SCL2 <sup>(1,4)</sup>	—	—	—	—	IOCB1	Y	—
RB2	23	20	ANB2	—	—	—	—	—	—	—	—	SDA2, SDI2 <sup>(1,4)</sup>	—	—	—	—	IOCB2	Y	—
RB3	24	21	ANB3	—	C1IN2- C2IN2-	—	—	—	—	—	—	—	—	—	—	—	IOCB3	Y	—
RB4	25	22	ANB4 ADACT <sup>(1)</sup>	—	—	—	—	—	—	—	—	—	—	—	—	—	IOCB4	Y	—
RB5	26	23	ANB5	—	—	—	—	T1G <sup>(1)</sup>	—	—	—	—	—	—	—	—	IOCB5	Y	—
RB6	27	24	ANB6	—	—	—	—	—	—	—	—	—	—	TX2 CK2 <sup>(1)</sup>	CLCIN2 <sup>(1)</sup>	—	IOCB6	Y	ICSPCLK
RB7	28	25	ANB7	—	—	—	DAC1OUT2	—	—	—	—	—	—	RX2 DT2 <sup>(1)</sup>	CLCIN3 <sup>(1)</sup>	—	IOCB7	Y	ICSPDAT

- Note**
- 1: This is a PPS remappable input signal. The input function may be moved from the default location shown to one of several other PORTx pins.
  - 2: All digital output signals shown in this row are PPS remappable. These signals may be mapped to output onto one or more PORTx pin options.
  - 3: This is a bidirectional signal. For normal module operation, the firmware should map this signal to the same pin in both the PPS input and PPS output registers.
  - 4: These pins are configured for I<sup>2</sup>C logic levels. PPS assignments to the other pins will operate, but input logic levels will be standard TTL/ST as selected by the INLVL register, instead of the I<sup>2</sup>C specific or SMBus input buffer thresholds.

**TABLE 3: 28-PIN ALLOCATION TABLE (PIC16(L)F15354, PIC16(L)F15355) (CONTINUED)**

I/O <sup>(2)</sup>	28-Pin PDIP/SSOP	28-Pin UQFN	ADC	Reference	Comparator	NCO	DAC	Timers	CCP	PWM	CWG	MSSP	ZCD	EUSART	CLC	CLKR	Interrupt	Pull-up	Basic
RC0	11	8	ANC0	—	—	—	—	SOSCO T1CK1	—	—	—	—	—	—	—	—	IOCC0	Y	—
RC1	12	9	ANC1	—	—	—	—	SOSCI	CCP2 <sup>(1)</sup>	—	—	—	—	—	—	—	IOCC1	Y	—
RC2	13	10	ANC2	—	—	—	—	—	CCP1 <sup>(1)</sup>	—	—	—	—	—	—	—	IOCC2	Y	—
RC3	14	11	ANC3	—	—	—	—	T2IN <sup>(1)</sup>	—	—	—	SCL1, SCK1 <sup>(1,4)</sup>	—	—	—	—	IOCC3	Y	—
RC4	15	12	ANC4	—	—	—	—	—	—	—	—	SDA1, SDI1 <sup>(1,4)</sup>	—	—	—	—	IOCC4	Y	—
RC5	16	13	ANC5	—	—	—	—	—	—	—	—	—	—	—	—	—	IOCC5	Y	—
RC6	17	14	ANC6	—	—	—	—	—	—	—	—	—	—	TX1 CK1 <sup>(1)</sup>	—	—	IOCC6	Y	—
RC7	18	15	ANC7	—	—	—	—	—	—	—	—	—	—	RX1 DT1 <sup>(1)</sup>	—	—	IOCC7	Y	—
RE3	1	26	—	—	—	—	—	—	—	—	—	—	—	—	—	—	IOCE3	Y	MCLR VPP
VDD	20	17	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	VDD
VSS	8	16	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	VSS
VSS	19	5	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	VSS
OUT <sup>(2)</sup>	—	—	—	—	C1OUT	NCO1OUT	—	TMR0	CCP1	PWM3OUT	CWG1A CWG2A	SDO1/2	—	DT <sup>(1,2)</sup>	CLC1OUT	CLKR	—	—	—
	—	—	—	—	C2OUT	—	—	—	CCP2	PWM4OUT	CWG1B CWG2B	SCK1/2	—	CK <sup>(1,2)</sup>	CLC2OUT	—	—	—	—
	—	—	—	—	—	—	—	—	—	PWM5OUT	CWG1C CWG2C	SCL1 <sup>(3,4)</sup> SCL2 <sup>(3,4)</sup>	—	TX <sup>(1,2)</sup>	CLC3OUT	—	—	—	—
	—	—	—	—	—	—	—	—	—	PWM6OUT	CWG1D CWG2D	SDA1 <sup>(3,4)</sup> SDA2 <sup>(3,4)</sup>	—	—	CLC4OUT	—	—	—	—

- Note**
- 1: This is a PPS remappable input signal. The input function may be moved from the default location shown to one of several other PORTx pins.
  - 2: All digital output signals shown in this row are PPS remappable. These signals may be mapped to output onto one or more PORTx pin options.
  - 3: This is a bidirectional signal. For normal module operation, the firmware should map this signal to the same pin in both the PPS input and PPS output registers.
  - 4: These pins are configured for I<sup>2</sup>C logic levels. PPS assignments to the other pins will operate, but input logic levels will be standard TTL/ST as selected by the INLVL register, instead of the I<sup>2</sup>C specific or SMBus input buffer thresholds.

## Table of Contents

1.0 Device Overview .....	10
2.0 Guidelines for Getting Started with PIC16(L)F15354/55 Microcontrollers .....	19
3.0 Enhanced Mid-Range CPU .....	22
4.0 Memory Organization .....	24
5.0 Device Configuration .....	75
6.0 Device Information Area .....	86
7.0 Device Configuration Information .....	88
8.0 Resets .....	89
9.0 Oscillator Module (with Fail-Safe Clock Monitor) .....	100
10.0 Interrupts .....	117
11.0 Power-Saving Operation Modes .....	139
12.0 Windowed Watchdog Timer (WWDT) .....	146
13.0 Nonvolatile Memory (NVM) Control .....	154
14.0 I/O Ports .....	172
15.0 Peripheral Pin Select (PPS) Module .....	194
16.0 Peripheral Module Disable .....	203
17.0 Interrupt-On-Change .....	211
18.0 Fixed Voltage Reference (FVR) .....	221
19.0 Temperature Indicator Module .....	224
20.0 Analog-to-Digital Converter (ADC) Module .....	226
21.0 5-Bit Digital-to-Analog Converter (DAC1) Module .....	240
22.0 Numerically Controlled Oscillator (NCO) Module .....	245
23.0 Comparator Module .....	255
24.0 Zero-Cross Detection (ZCD) Module .....	265
25.0 Timer0 Module .....	271
26.0 Timer1 Module with Gate Control .....	277
27.0 Timer2 Module With Hardware Limit Timer (HLT) .....	291
28.0 Capture/Compare/PWM Modules .....	312
29.0 Pulse-Width Modulation (PWM) .....	323
30.0 Complementary Waveform Generator (CWG) Module .....	330
31.0 Configurable Logic Cell (CLC) .....	355
32.0 Master Synchronous Serial Port (MSSPx) Modules .....	372
33.0 Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART) .....	423
34.0 Reference Clock Output Module .....	451
35.0 n-Circuit Serial Programming™ (ICSP™) .....	455
36.0 Instruction Set Summary .....	457
37.0 Electrical Specifications .....	470
38.0 DC and AC Characteristics Graphs and Charts .....	499
39.0 Development Support .....	519
40.0 Packaging Information .....	523

## TO OUR VALUED CUSTOMERS

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

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at [docerrors@microchip.com](mailto:docerrors@microchip.com). 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](http://www.microchip.com) to receive the most current information on all of our products.

# PIC16(L)F15354/55

## 1.0 DEVICE OVERVIEW

The PIC16(L)F15354/55 are described within this data sheet. The PIC16(L)F15354/55 devices are available in 28-pin SPDIP, SSOP, SOIC, and UQFN packages. [Figure 1-1](#) shows the block diagram of the PIC16(L)F15354/55 devices. [Table 1-2](#) shows the pinout descriptions.

Reference [Table 1-1](#) for peripherals available per device.

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

Peripheral	PIC16(L)F15354/55	
Analog-to-Digital Converter		•
Digital-to-Analog Converter (DAC1)		•
Fixed Voltage Reference (FVR)		•
Numerically Controlled Oscillator (NCO1)		•
Temperature Indicator Module (TIM)		•
Zero-Cross Detect (ZCD1)		•
Capture/Compare/PWM Modules (CCP)		
	CCP1	•
	CCP2	•
Comparator Module (Cx)		
	C1	•
	C2	•
Configurable Logic Cell (CLC)		
	CLC1	•
	CLC2	•
	CLC3	•
	CLC4	•
Complementary Waveform Generator (CWG)		
	CWG1	•
Enhanced Universal Synchronous/Asynchronous Receiver/Transmitter (EUSART)		
	EUSART1	•
	EUSART2	•
Master Synchronous Serial Ports (MSSP)		
	MSSP1	•
	MSSP2	•
Pulse-Width Modulator (PWM)		
	PWM3	•
	PWM4	•
	PWM5	•
	PWM6	•
Timers		
	Timer0	•
	Timer1	•
	Timer2	•

## 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 Push-Pull mode:

Example 1:

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

Example 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

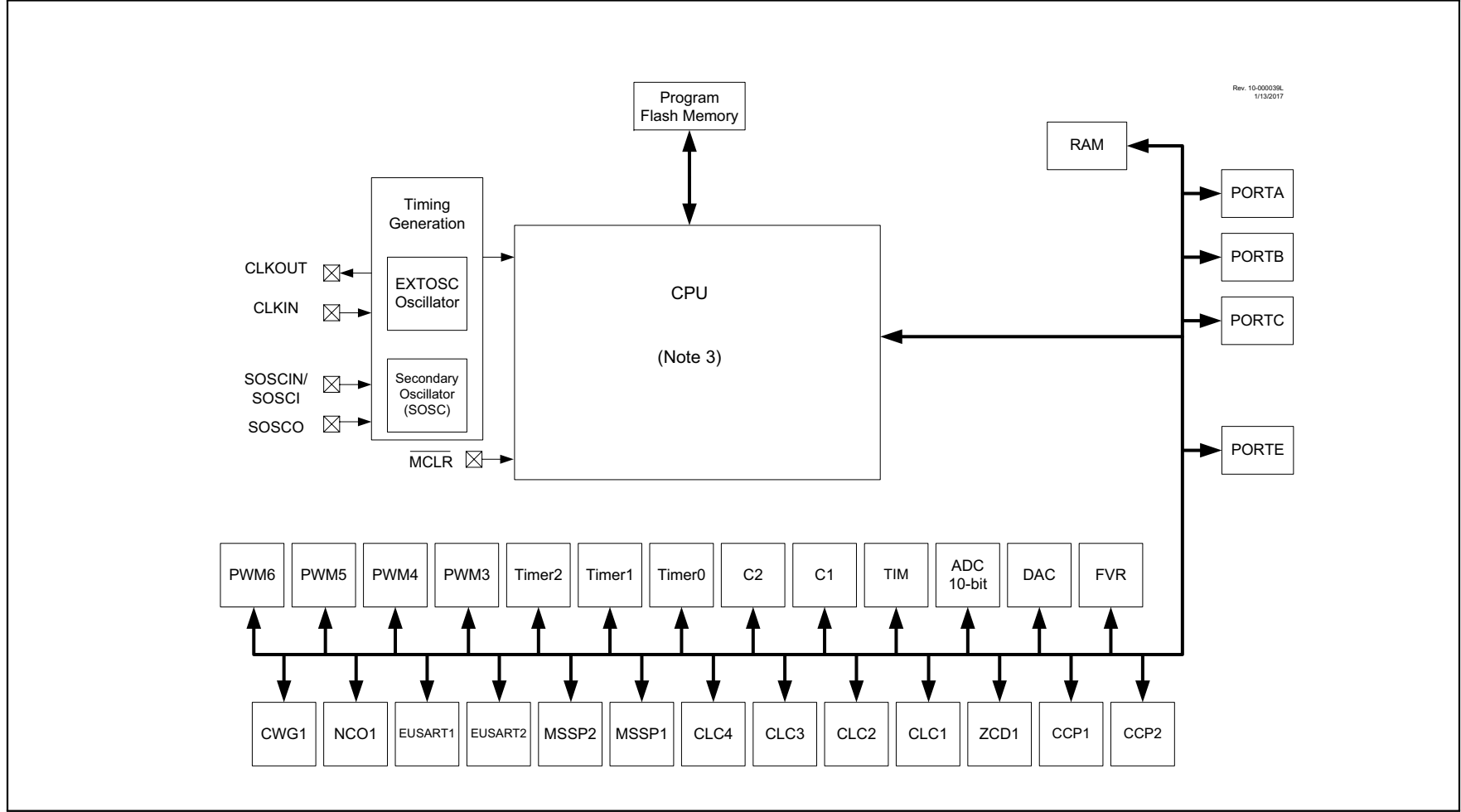
Status, interrupt enables, interrupt flags, and mirror bits are contained in registers that span more than one peripheral. In these cases, the bit name shown is unique so there is no prefix or short name variant.

### 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)F15354/55 BLOCK DIAGRAM**





**TABLE 1-2: PIC16(L)F15354/55 PINOUT DESCRIPTION**

Name	Function	Input Type	Output Type	Description
RA0/ANA0/C1IN0-/C2IN0-/CLCIN0 <sup>(1)</sup> /IOCA0	RA0	TTL/ST	CMOS/OD	General purpose I/O.
	ANA0	AN	—	ADC Channel A0 input.
	C1IN0-	AN	—	Comparator 1 negative input.
	C2IN0-	AN	—	Comparator 2 negative input.
	CLCIN0 <sup>(1)</sup>	TTL/ST	—	Configurable Logic Cell source input.
	IOCA0	TTL/ST	—	Interrupt-on-change input.
RA1/ANA1/C1IN1-/C2IN1-/CLCIN1 <sup>(1)</sup> /IOCA1	RA1	TTL/ST	CMOS/OD	General purpose I/O.
	ANA1	AN	—	ADC Channel A1 input.
	C1IN1-	AN	—	Comparator 1 negative input.
	C2IN1-	AN	—	Comparator 2 negative input.
	CLCIN1 <sup>(1)</sup>	TTL/ST	—	Configurable Logic Cell source input.
	IOCA1	TTL/ST	—	Interrupt-on-change input.
RA2/ANA2/C1IN0+/C2IN0+/DAC1OUT1/IOCA2	RA2	TTL/ST	CMOS/OD	General purpose I/O.
	ANA2	AN	—	ADC Channel A2 input.
	C1IN0+	AN	—	Comparator 2 positive input.
	C2IN0+	AN	—	Comparator 2 positive input.
	DAC1OUT1	—	AN	Digital-to-Analog Converter output.
	IOCA2	TTL/ST	—	Interrupt-on-change input.
RA3/ANA3/C1IN1+/VREF+/IOCA3/DAC1REF+	RA3	TTL/ST	CMOS/OD	General purpose I/O.
	ANA3	AN	—	ADC Channel A3 input.
	C1IN1+	AN	—	Comparator 1 positive input.
	VREF+	AN	—	External ADC and/or DAC positive reference input.
	IOCA3	TTL/ST	—	Interrupt-on-change input.
	DAC1REF+	TTL/ST	AN	DAC positive reference.
RA4/ANA4/T0CKI <sup>(1)</sup> /IOCA4	RA4	TTL/ST	CMOS/OD	General purpose I/O.
	ANA4	AN	—	ADC Channel A4 input.
	T0CKI <sup>(1)</sup>	TTL/ST	—	Timer0 clock input.
	IOCA4	TTL/ST	—	Interrupt-on-change input.
RA5/ANA5/ $\overline{SS1}$ <sup>(1)</sup> /IOCA5	RA5	TTL/ST	CMOS/OD	General purpose I/O.
	ANA5	AN	—	ADC Channel A5 input.
	$\overline{SS1}$ <sup>(1)</sup>	TTL/ST	—	MSSP1 SPI slave select input.
	IOCA5	TTL/ST	—	Interrupt-on-change input.

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

- Note**
- 1: This is a PPS remappable input signal. The input function may be moved from the default location shown to one of several other PORTx pins. Refer to [Table 15-1](#) for details on which PORT pins may be used for this signal.
  - 2: All output signals shown in this row are PPS remappable. These signals may be mapped to output onto one of several PORTx pin options as described in [Table 15-3](#).
  - 3: This is a bidirectional signal. For normal module operation, the firmware should map this signal to the same pin in both the PPS input and PPS output registers.
  - 4: These pins are configured for I<sup>2</sup>C logic levels. The SCLx/SDAx signals may be assigned to any of the RB1/RB2/RC3/RC4 pins. PPS assignments to the other pins (e.g., RA5) will operate, but input logic levels will be standard TTL/ST, as selected by the INLVL register, instead of the I<sup>2</sup>C specific or SMBus input buffer thresholds.

**TABLE 1-2: PIC16(L)F15354/55 PINOUT DESCRIPTION (CONTINUED)**

Name	Function	Input Type	Output Type	Description
RA6/ANA6/OSC2/CLKOUT/IOCA6	RA6	TTL/ST	CMOS/OD	General purpose I/O.
	ANA6	AN	—	ADC Channel A6 input.
	OSC2	—	XTAL	External Crystal/Resonator (LP, XT, HS modes) driver output.
	CLKOUT	—	CMOS/OD	Fosc/4 digital output (in non-crystal/resonator modes).
	IOCA6	TTL/ST	—	Interrupt-on-change input.
RA7/ANA7/OSC1/CLKIN/IOCA7	RA7	TTL/ST	CMOS/OD	General purpose I/O.
	ANA7	AN	—	ADC Channel A7 input.
	OSC1	XTAL	—	External Crystal/Resonator (LP, XT, HS modes) driver input.
	CLKIN	TTL/ST	—	External digital clock input.
	IOCA7	TTL/ST	—	Interrupt-on-change input.
RB0/ANB0/C2IN1+/ZCD1/ $\overline{SS}2^{(1)}$ / CWG1IN <sup>(1)</sup> /INT <sup>(1)</sup> /IOCB0	RB0	TTL/ST	CMOS/OD	General purpose I/O.
	ANB0	AN	—	ADC Channel B0 input.
	C2IN1+	AN	—	Comparator 2 positive input.
	ZCD1	AN	AN	Zero-cross detect input pin (with constant current sink/source).
	$\overline{SS}2^{(1)}$	TTL/ST	—	MSSP2 SPI slave select input.
	CWG1IN <sup>(1)</sup>	TTL/ST	—	Complementary Waveform Generator 1 input.
	INT <sup>(1)</sup>	TTL/ST	—	External interrupt request input.
	IOCB0	TTL/ST	—	Interrupt-on-change input.
RB1/ANB1/C1IN3-/C2IN3-/SCL2 <sup>(3,4)</sup> / SCK2 <sup>(1)</sup> /IOCB1	RB1	TTL/ST	CMOS/OD	General purpose I/O.
	ANB1	AN	—	ADC Channel B1 input.
	C1IN3-	AN	—	Comparator 1 negative input.
	C2IN3-	AN	—	Comparator 2 negative input.
	SCL2 <sup>(3,4)</sup>	I <sup>2</sup> C	OD	MSSP2 I <sup>2</sup> C clock input/output.
	SCK2 <sup>(1)</sup>	TTL/ST	CMOS/OD	MSSP2 SPI serial clock (default input location, SCK2 is a PPS remappable input and output).
	IOCB1	TTL/ST	—	Interrupt-on-change input.
RB2/ANB2/SDA2 <sup>(3,4)</sup> /SDI2 <sup>(1)</sup> /IOCB2	RB2	TTL/ST	CMOS/OD	General purpose I/O.
	ANB2	AN	—	ADC Channel B2 input.
	SDA2 <sup>(3,4)</sup>	I <sup>2</sup> C	OD	MSSP2 I <sup>2</sup> C serial data input/output.
	SDI2 <sup>(1)</sup>	TTL/ST	—	MSSP2 SPI serial data input.
	IOCB2	TTL/ST	—	Interrupt-on-change input.

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

- Note**
- 1: This is a PPS remappable input signal. The input function may be moved from the default location shown to one of several other PORTx pins. Refer to [Table 15-1](#) for details on which PORT pins may be used for this signal.
  - 2: All output signals shown in this row are PPS remappable. These signals may be mapped to output onto one of several PORTx pin options as described in [Table 15-3](#).
  - 3: This is a bidirectional signal. For normal module operation, the firmware should map this signal to the same pin in both the PPS input and PPS output registers.
  - 4: These pins are configured for I<sup>2</sup>C logic levels. The SCLx/SDAx signals may be assigned to any of the RB1/RB2/RC3/RC4 pins. PPS assignments to the other pins (e.g., RA5) will operate, but input logic levels will be standard TTL/ST, as selected by the INLVL register, instead of the I<sup>2</sup>C specific or SMBus input buffer thresholds.

# PIC16(L)F15354/55

**TABLE 1-2: PIC16(L)F15354/55 PINOUT DESCRIPTION (CONTINUED)**

Name	Function	Input Type	Output Type	Description
RB3/ANB3/C1IN2-/C2IN2-/IOCB3	RB3	TTL/ST	CMOS/OD	General purpose I/O.
	ANB3	AN	—	ADC Channel B3 input.
	C1IN2-	AN	—	Comparator 1 negative input.
	C2IN2-	AN	—	Comparator 2 negative input.
RB4/ANB4/ADACT <sup>(1)</sup> /IOCB4	RB4	TTL/ST	CMOS/OD	General purpose I/O.
	ANB4	AN	—	ADC Channel B4 input.
	ADACT <sup>(1)</sup>	TTL/ST	—	ADC Auto-Conversion Trigger input.
	IOCB4	TTL/ST	—	Interrupt-on-change input.
RB5/ANB5/T1G <sup>(1)</sup> /IOCB5	RB5	TTL/ST	CMOS/OD	General purpose I/O.
	ANB5	AN	—	ADC Channel B5 input.
	T1G <sup>(1)</sup>	ST	—	Timer1 Gate input.
	IOCB5	TTL/ST	—	Interrupt-on-change input.
RB6/ANB6/CLCIN2 <sup>(1)</sup> /IOCB6/TX2/CK2 <sup>(3)</sup> /ICSPCLK	RB6	TTL/ST	CMOS/OD	General purpose I/O.
	ANB6	AN	—	ADC Channel B6 input.
	CLCIN2 <sup>(1)</sup>	TTL/ST	—	Configurable Logic Cell source input.
	IOCB6	TTL/ST	—	Interrupt-on-change input.
	TX2	—	CMOS	EUSART2 asynchronous.
	CK2 <sup>(3)</sup>	TTL/ST	CMOS/OD	EUSART2 synchronous mode clock input/output.
RB7/ANB7/RX2/DT2/CLCIN3 <sup>(1)</sup> /IOCB7/DAC1OUT2/ICSPDAT	RB7	TTL/ST	CMOS/OD	General purpose I/O.
	ANB7	AN	—	ADC Channel B7 input.
	CLCIN3 <sup>(1)</sup>	TTL/ST	—	Configurable Logic Cell source input.
	IOCB7	TTL/ST	—	Interrupt-on-change input.
	RX2 <sup>(1)</sup>	TTL/ST	—	EUSART2 Asynchronous mode receiver data input.
	DT2 <sup>(3)</sup>	TTL/ST	CMOS/OD	EUSART2 Synchronous mode data input/output.
	DAC1OUT2	—	AN	Digital-to-Analog Converter output.
RC0/ANC0/T1CKI <sup>(1)</sup> /IOCC0/SOSCO	RC0	TTL/ST	CMOS/OD	General purpose I/O.
	ANC0	AN	—	ADC Channel C0 input.
	T1CKI <sup>(1)</sup>	TTL/ST	—	Timer1 external digital clock input.
	IOCC0	TTL/ST	—	Interrupt-on-change input.
	SOSCO	—	AN	32.768 kHz secondary oscillator crystal driver 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<sup>2</sup>C = Schmitt Trigger input with I<sup>2</sup>C  
HV = High Voltage      XTAL = Crystal levels

- Note**
- 1: This is a PPS remappable input signal. The input function may be moved from the default location shown to one of several other PORTx pins. Refer to [Table 15-1](#) for details on which PORT pins may be used for this signal.
  - 2: All output signals shown in this row are PPS remappable. These signals may be mapped to output onto one of several PORTx pin options as described in [Table 15-3](#).
  - 3: This is a bidirectional signal. For normal module operation, the firmware should map this signal to the same pin in both the PPS input and PPS output registers.
  - 4: These pins are configured for I<sup>2</sup>C logic levels. The SCLx/SDAx signals may be assigned to any of the RB1/RB2/RC3/RC4 pins. PPS assignments to the other pins (e.g., RA5) will operate, but input logic levels will be standard TTL/ST, as selected by the INLVL register, instead of the I<sup>2</sup>C specific or SMBus input buffer thresholds.

# PIC16(L)F15354/55

**TABLE 1-2: PIC16(L)F15354/55 PINOUT DESCRIPTION (CONTINUED)**

Name	Function	Input Type	Output Type	Description
RC1/ANC1/CCP2 <sup>(1)</sup> /IOCC1/SOSCI	RC1	TTL/ST	CMOS/OD	General purpose I/O.
	ANC1	AN	—	ADC Channel C1 input.
	CCP2 <sup>(1)</sup>	TTL/ST	CMOS/OD	CCP2 Capture Input.
	IOCC1	TTL/ST	—	Interrupt-on-change input.
	SOSCI	AN	—	32.768 kHz secondary oscillator crystal driver input.
RC2/ANC2/CCP1 <sup>(1)</sup> /IOCC2	RC2	TTL/ST	CMOS/OD	General purpose I/O.
	ANC2	AN	—	ADC Channel C2 input.
	CCP1 <sup>(1)</sup>	TTL/ST	CMOS/OD	CCP1 Capture Input.
	IOCC2	TTL/ST	—	Interrupt-on-change input.
RC3/ANC3/SCL1 <sup>(3,4)</sup> /SCK1 <sup>(1)</sup> /T2IN <sup>(1)</sup> /IOCC3	RC3	TTL/ST	CMOS/OD	General purpose I/O.
	ANC3	AN	—	ADC Channel C3 input.
	SCL1 <sup>(3,4)</sup>	I <sup>2</sup> C	OD	MSSP1 I <sup>2</sup> C input/output.
	SCK1 <sup>(1)</sup>	TTL/ST	CMOS/OD	MSSP1 SPI clock input/output (default input location, SCK1 is a PPS remappable input and output).
	T2IN <sup>(1)</sup>	TTL/ST	—	Timer2 external input.
	IOCC3	TTL/ST	—	Interrupt-on-change input.
RC4/ANC4/SDA1 <sup>(3,4)</sup> /SDI1 <sup>(1)</sup> /IOCC4	RC4	TTL/ST	CMOS/OD	General purpose I/O.
	ANC4	AN	—	ADC Channel C4 input.
	SDA1 <sup>(3,4)</sup>	I <sup>2</sup> C	OD	MSSP1 I <sup>2</sup> C serial data input/output.
	SDI1 <sup>(1)</sup>	TTL/ST	—	MSSP1 SPI serial data input.
	IOCC4	TTL/ST	—	Interrupt-on-change input.
RC5/ANC5/IOCC5	RC5	TTL/ST	CMOS/OD	General purpose I/O.
	ANC5	AN	—	ADC Channel C5 input.
	IOCC5	TTL/ST	—	Interrupt-on-change input.
RC6/ANC6/TX1/CK1 <sup>(1)</sup> /IOCC6	RC6	TTL/ST	CMOS/OD	General purpose I/O.
	ANC6	AN	—	ADC Channel C6 input.
	TX1	—	CMOS	EUSART1 asynchronous transmit.
	CK1 <sup>(1)</sup>	TTL/ST	CMOS/OD	EUSART 1 synchronous mode clock input/output.
	IOCC6	TTL/ST	—	Interrupt-on-change input.
RC7/ANC7/RX1/DT1 <sup>(3)</sup> /IOCC7	RC7	TTL/ST	CMOS/OD	General purpose I/O.
	ANC7	AN	—	ADC Channel C7 input.
	RX1	TTL/ST	—	EUSART1 Asynchronous mode receiver data input.
	DT1 <sup>(3)</sup>	TTL/ST	CMOS/OD	EUSART1 Synchronous mode data input/output.
	IOCC7	TTL/ST	—	Interrupt-on-change input.

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

- Note**
- 1: This is a PPS remappable input signal. The input function may be moved from the default location shown to one of several other PORTx pins. Refer to [Table 15-1](#) for details on which PORT pins may be used for this signal.
  - 2: All output signals shown in this row are PPS remappable. These signals may be mapped to output onto one of several PORTx pin options as described in [Table 15-3](#).
  - 3: This is a bidirectional signal. For normal module operation, the firmware should map this signal to the same pin in both the PPS input and PPS output registers.
  - 4: These pins are configured for I<sup>2</sup>C logic levels. The SCLx/SDAx signals may be assigned to any of the RB1/RB2/RC3/RC4 pins. PPS assignments to the other pins (e.g., RA5) will operate, but input logic levels will be standard TTL/ST, as selected by the INLVL register, instead of the I<sup>2</sup>C specific or SMBus input buffer thresholds.

# PIC16(L)F15354/55

**TABLE 1-2: PIC16(L)F15354/55 PINOUT DESCRIPTION (CONTINUED)**

Name	Function	Input Type	Output Type	Description
RE3/IOCE3/MCLR/VPP	RE3	TTL/ST	—	General purpose input only (when MCLR is disabled by the Configuration bit).
	IOCE3	TTL/ST	—	Interrupt-on-change input.
	MCLR	ST	—	Master clear input with internal weak pull-up resistor.
	VPP	HV	—	ICSP™ High-Voltage Programming mode entry input.
VDD	VDD	Power	—	Positive supply voltage input.
VSS	VSS	Power	—	Ground reference.

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

- Note** 1: This is a PPS remappable input signal. The input function may be moved from the default location shown to one of several other PORTx pins. Refer to [Table 15-1](#) for details on which PORT pins may be used for this signal.
- 2: All output signals shown in this row are PPS remappable. These signals may be mapped to output onto one of several PORTx pin options as described in [Table 15-3](#).
- 3: This is a bidirectional signal. For normal module operation, the firmware should map this signal to the same pin in both the PPS input and PPS output registers.
- 4: These pins are configured for I<sup>2</sup>C logic levels. The SCLx/SDAx signals may be assigned to any of the RB1/RB2/RC3/RC4 pins. PPS assignments to the other pins (e.g., RA5) will operate, but input logic levels will be standard TTL/ST, as selected by the INLVL register, instead of the I<sup>2</sup>C specific or SMBus input buffer thresholds.

# PIC16(L)F15354/55

**TABLE 1-2: PIC16(L)F15354/55 PINOUT DESCRIPTION (CONTINUED)**

Name	Function	Input Type	Output Type	Description
OUT <sup>(2)</sup>	C1OUT	—	CMOS/OD	Comparator 1 output.
	C2OUT	—	CMOS/OD	Comparator 2 output.
	SDO1	—	CMOS/OD	MSSP1 SPI serial data output.
	SCK1	—	CMOS/OD	MSSP1 SPI serial clock output.
	SDO2	—	CMOS/OD	MSSP2 SPI serial data output.
	SCK2	—	CMOS/OD	MSSP2 SPI serial clock output.
	TX1	—	CMOS/OD	EUSART1 Asynchronous mode transmitter data output.
	CK1 <sup>(3)</sup>	—	CMOS/OD	EUSART1 Synchronous mode clock output.
	TX2	—	CMOS/OD	EUSART2 Asynchronous mode transmitter data output.
	CK2 <sup>(3)</sup>	—	CMOS/OD	EUSART2 Synchronous mode clock output.
	DT <sup>(3)</sup>	—	CMOS/OD	EUSART Synchronous mode data output.
	TMR0	—	CMOS/OD	Timer0 output.
	CCP1	—	CMOS/OD	CCP2 output (compare/PWM functions).
	CCP2	—	CMOS/OD	CCP2 output (compare/PWM functions).
	PWM3OUT	—	CMOS/OD	PWM3 output.
	PWM4OUT	—	CMOS/OD	PWM4 output.
	PWM5OUT	—	CMOS/OD	PWM5 output.
	PWM6OUT	—	CMOS/OD	PWM6 output.
	CWG1A	—	CMOS/OD	Complementary Waveform Generator 1 output A.
	CWG1B	—	CMOS/OD	Complementary Waveform Generator 1 output B.
	CWG1C	—	CMOS/OD	Complementary Waveform Generator 1 output C.
	CWG1D	—	CMOS/OD	Complementary Waveform Generator 1 output D.
	CLC1OUT	—	CMOS/OD	Configurable Logic Cell 1 output.
	CLC2OUT	—	CMOS/OD	Configurable Logic Cell 2 output.
	CLC3OUT	—	CMOS/OD	Configurable Logic Cell 3 output.
	CLC4OUT	—	CMOS/OD	Configurable Logic Cell 4 output.
	NCO1OUT	—	CMOS/OD	Numerically Controller Oscillator output.
CLKR	—	CMOS/OD	Clock Reference module 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<sup>2</sup>C = Schmitt Trigger input with I<sup>2</sup>C  
HV = High Voltage    XTAL = Crystal levels

- Note**
- 1: This is a PPS remappable input signal. The input function may be moved from the default location shown to one of several other PORTx pins. Refer to [Table 15-1](#) for details on which PORT pins may be used for this signal.
  - 2: All output signals shown in this row are PPS remappable. These signals may be mapped to output onto one of several PORTx pin options as described in [Table 15-3](#).
  - 3: This is a bidirectional signal. For normal module operation, the firmware should map this signal to the same pin in both the PPS input and PPS output registers.
  - 4: These pins are configured for I<sup>2</sup>C logic levels. The SCLx/SDAx signals may be assigned to any of the RB1/RB2/RC3/RC4 pins. PPS assignments to the other pins (e.g., RA5) will operate, but input logic levels will be standard TTL/ST, as selected by the INLVL register, instead of the I<sup>2</sup>C specific or SMBus input buffer thresholds.

## 2.0 GUIDELINES FOR GETTING STARTED WITH PIC16(L)F15354/55 MICROCONTROLLERS

### 2.1 Basic Connection Requirements

Getting started with the PIC16(L)F15354/55 family of 8-bit microcontrollers requires attention to a minimal set of device pin connections before proceeding with development.

The following pins must always be connected:

- All VDD and VSS pins (see [Section 2.2 “Power Supply Pins”](#))
- MCLR pin (see [Section 2.3 “Master Clear \(MCLR\) Pin”](#))

These pins must also be connected if they are being used in the end application:

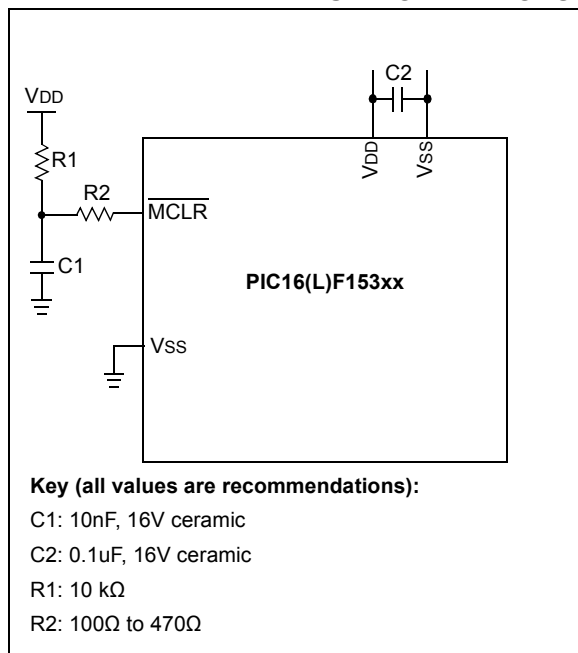
- ICSPCLK/ICSPDAT pins used for In-Circuit Serial Programming™ (ICSP™) and debugging purposes (see [Section 2.4 “ICSP™ Pins”](#))
- OSCI and OSCO pins when an external oscillator source is used (see [Section 2.5 “External Oscillator Pins”](#))

Additionally, the following pins may be required:

- VREF+/VREF- pins are used when external voltage reference for analog modules is implemented

The minimum mandatory connections are shown in [Figure 2-1](#).

**FIGURE 2-1: RECOMMENDED MINIMUM CONNECTIONS**



### 2.2 Power Supply Pins

#### 2.2.1 DECOUPLING CAPACITORS

The use of decoupling capacitors on every pair of power supply pins (VDD and VSS) is required.

Consider the following criteria when using decoupling capacitors:

- **Value and type of capacitor:** A 0.1  $\mu\text{F}$  (100 nF), 10-25V capacitor is recommended. The capacitor should be a low-ESR device, with a resonance frequency in the range of 200 MHz and higher. Ceramic capacitors are recommended.
- **Placement on the printed circuit board:** The decoupling capacitors should be placed as close to the pins as possible. It is recommended to place the capacitors on the same side of the board as the device. If space is constricted, the capacitor can be placed on another layer on the PCB using a via; however, ensure that the trace length from the pin to the capacitor is no greater than 0.25 inch (6 mm).
- **Handling high-frequency noise:** If the board is experiencing high-frequency noise (upward of tens of MHz), add a second ceramic type capacitor in parallel to the above described decoupling capacitor. The value of the second capacitor can be in the range of 0.01  $\mu\text{F}$  to 0.001  $\mu\text{F}$ . Place this second capacitor next to each primary decoupling capacitor. In high-speed circuit designs, consider implementing a decade pair of capacitances as close to the power and ground pins as possible (e.g., 0.1  $\mu\text{F}$  in parallel with 0.001  $\mu\text{F}$ ).
- **Maximizing performance:** On the board layout from the power supply circuit, run the power and return traces to the decoupling capacitors first, and then to the device pins. This ensures that the decoupling capacitors are first in the power chain. Equally important is to keep the trace length between the capacitor and the power pins to a minimum, thereby reducing PCB trace inductance.

#### 2.2.2 TANK CAPACITORS

On boards with power traces running longer than six inches in length, it is suggested to use a tank capacitor for integrated circuits, including microcontrollers, to supply a local power source. The value of the tank capacitor should be determined based on the trace resistance that connects the power supply source to the device, and the maximum current drawn by the device in the application. In other words, select the tank capacitor so that it meets the acceptable voltage sag at the device. Typical values range from 4.7  $\mu\text{F}$  to 47  $\mu\text{F}$ .

## 2.3 Master Clear ( $\overline{\text{MCLR}}$ ) Pin

The  $\overline{\text{MCLR}}$  pin provides two specific device functions: Device Reset, and Device Programming and Debugging. If programming and debugging are not required in the end application, a direct connection to VDD may be all that is required. The addition of other components, to help increase the application's resistance to spurious Resets from voltage sags, may be beneficial. A typical configuration is shown in Figure 2-1. Other circuit designs may be implemented, depending on the application's requirements.

During programming and debugging, the resistance and capacitance that can be added to the pin must be considered. Device programmers and debuggers drive the  $\overline{\text{MCLR}}$  pin. Consequently, specific voltage levels ( $V_{IH}$  and  $V_{IL}$ ) and fast signal transitions must not be adversely affected. Therefore, specific values of R1 and C1 will need to be adjusted based on the application and PCB requirements. For example, it is recommended that the capacitor, C1, be isolated from the  $\overline{\text{MCLR}}$  pin during programming and debugging operations by using a jumper (Figure 2-2). The jumper is replaced for normal run-time operations.

Any components associated with the  $\overline{\text{MCLR}}$  pin should be placed within 0.25 inch (6 mm) of the pin.

## 2.4 ICSP™ Pins

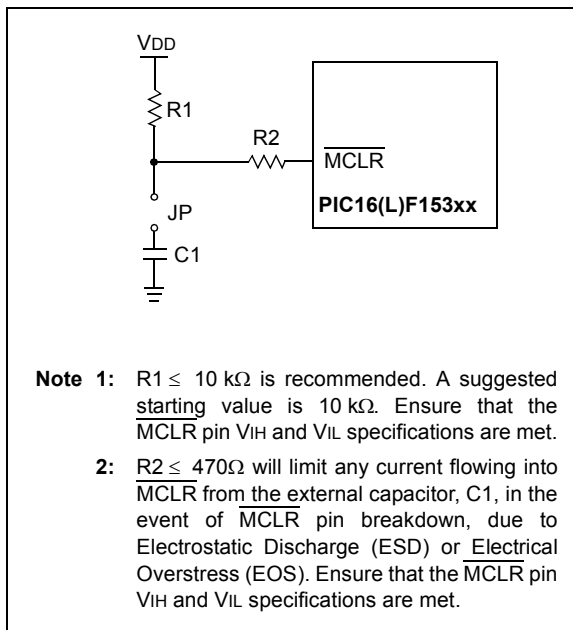
The ICSPCLK and ICSPDAT pins are used for In-Circuit Serial Programming™ (ICSP™) and debugging purposes. It is recommended to keep the trace length between the ICSP connector and the ICSP pins on the device as short as possible. If the ICSP connector is expected to experience an ESD event, a series resistor is recommended, with the value in the range of a few tens of ohms, not to exceed 100Ω.

Pull-up resistors, series diodes and capacitors on the ICSPCLK and ICSPDAT pins are not recommended as they will interfere with the programmer/debugger communications to the device. If such discrete components are an application requirement, they should be removed from the circuit during programming and debugging. Alternatively, refer to the AC/DC characteristics and timing requirements information in the respective device Flash programming specification for information on capacitive loading limits, and pin input voltage high ( $V_{IH}$ ) and input low ( $V_{IL}$ ) requirements.

For device emulation, ensure that the “Communication Channel Select” (i.e., ICSPCLK/ICSPDAT pins), programmed into the device, matches the physical connections for the ICSP to the Microchip debugger/emulator tool.

For more information on available Microchip development tools connection requirements, refer to [Section 39.0 “Development Support”](#).

**FIGURE 2-2: EXAMPLE OF  $\overline{\text{MCLR}}$  PIN CONNECTIONS**





## 2.5 External Oscillator Pins

Many microcontrollers have options for at least two oscillators: a high-frequency primary oscillator and a low-frequency secondary oscillator (refer to [Section 9.0 “Oscillator Module \(with Fail-Safe Clock Monitor\)”](#) for details).

The oscillator circuit should be placed on the same side of the board as the device. Place the oscillator circuit close to the respective oscillator pins with no more than 0.5 inch (12 mm) between the circuit components and the pins. The load capacitors should be placed next to the oscillator itself, on the same side of the board.

Use a grounded copper pour around the oscillator circuit to isolate it from surrounding circuits. The grounded copper pour should be routed directly to the MCU ground. Do not run any signal traces or power traces inside the ground pour. Also, if using a two-sided board, avoid any traces on the other side of the board where the crystal is placed.

Layout suggestions are shown in [Figure 2-3](#). In-line packages may be handled with a single-sided layout that completely encompasses the oscillator pins. With fine-pitch packages, it is not always possible to completely surround the pins and components. A suitable solution is to tie the broken guard sections to a mirrored ground layer. In all cases, the guard trace(s) must be returned to ground.

In planning the application's routing and I/O assignments, ensure that adjacent port pins, and other signals in close proximity to the oscillator, are benign (i.e., free of high frequencies, short rise and fall times, and other similar noise).

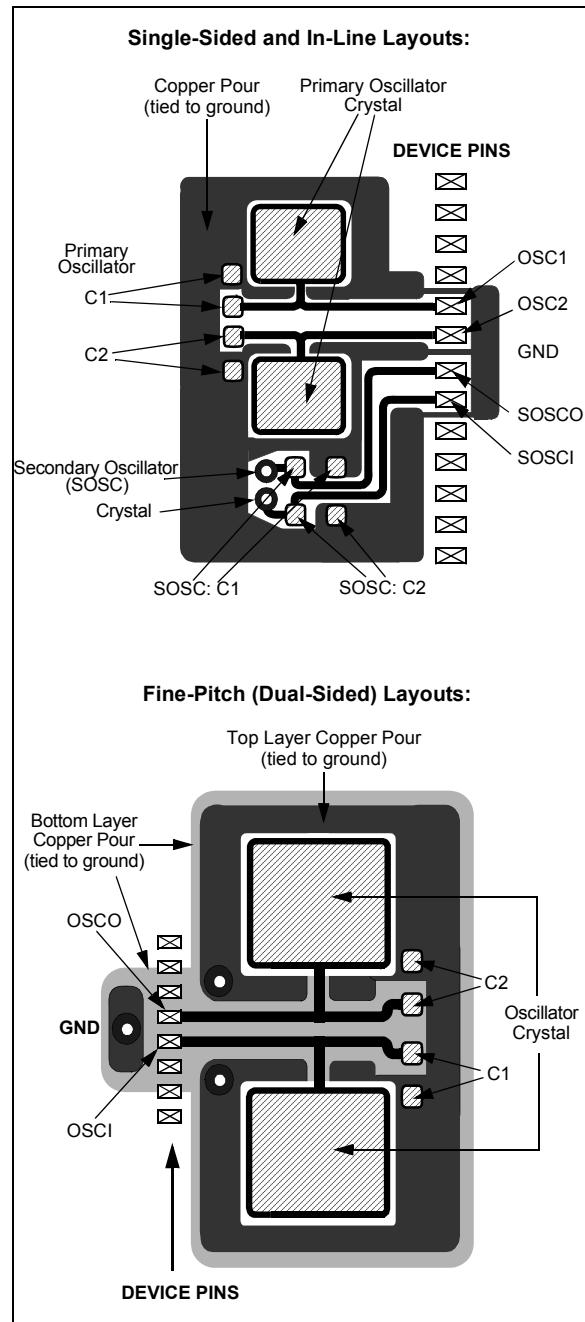
For additional information and design guidance on oscillator circuits, refer to these Microchip Application Notes, available at the corporate website ([www.microchip.com](http://www.microchip.com)):

- AN826, “Crystal Oscillator Basics and Crystal Selection for rPIC™ and PICmicro® Devices”
- AN849, “Basic PICmicro® Oscillator Design”
- AN943, “Practical PICmicro® Oscillator Analysis and Design”
- AN949, “Making Your Oscillator Work”

## 2.6 Unused I/Os

Unused I/O pins should be configured as outputs and driven to a logic low state. Alternatively, connect a 1 kΩ to 10 kΩ resistor to V<sub>SS</sub> on unused pins and drive the output to logic low.

**FIGURE 2-3: SUGGESTED PLACEMENT OF THE OSCILLATOR CIRCUIT**

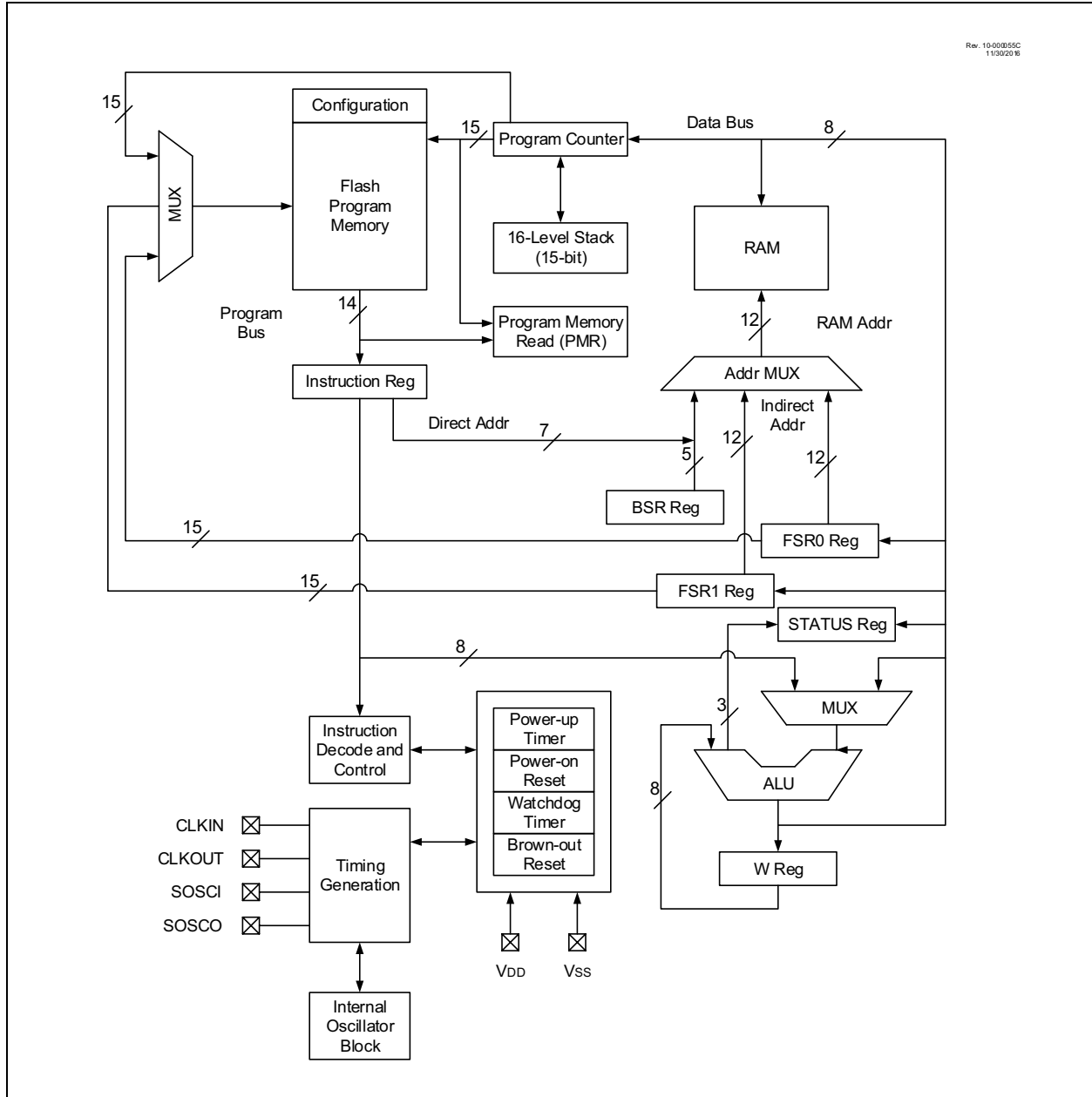


## 3.0 ENHANCED MID-RANGE CPU

This family of devices contains an enhanced mid-range 8-bit CPU core. The CPU has 48 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.

**FIGURE 3-1: CORE DATA PATH DIAGRAM**



## 3.1 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 10.5 “Automatic Context Saving”](#) for more information.

## 3.2 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 PCON0 register, and if enabled, will cause a software Reset. See [Section 4.5 “Stack”](#) for more details.

## 3.3 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 also be addressed linearly, providing the ability to access contiguous data larger than 80 bytes. See [Section 4.6 “Indirect Addressing”](#) for more details.

## 3.4 Instruction Set

There are 48 instructions for the enhanced mid-range CPU to support the features of the CPU. See [Section 36.0 “Instruction Set Summary”](#) for more details.

## 4.0 MEMORY ORGANIZATION

These devices contain the following types of memory:

- Program Memory
  - Configuration Words
  - Device ID
  - User ID
  - Program Flash Memory
  - Device Information Area (DIA)
  - Device Configuration Information (DCI)
  - Revision ID
- Data Memory
  - Core Registers
  - Special Function Registers
  - General Purpose RAM
  - Common RAM

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

- PCL and PCLATH
- Stack
- Indirect Addressing
- NVMREG access

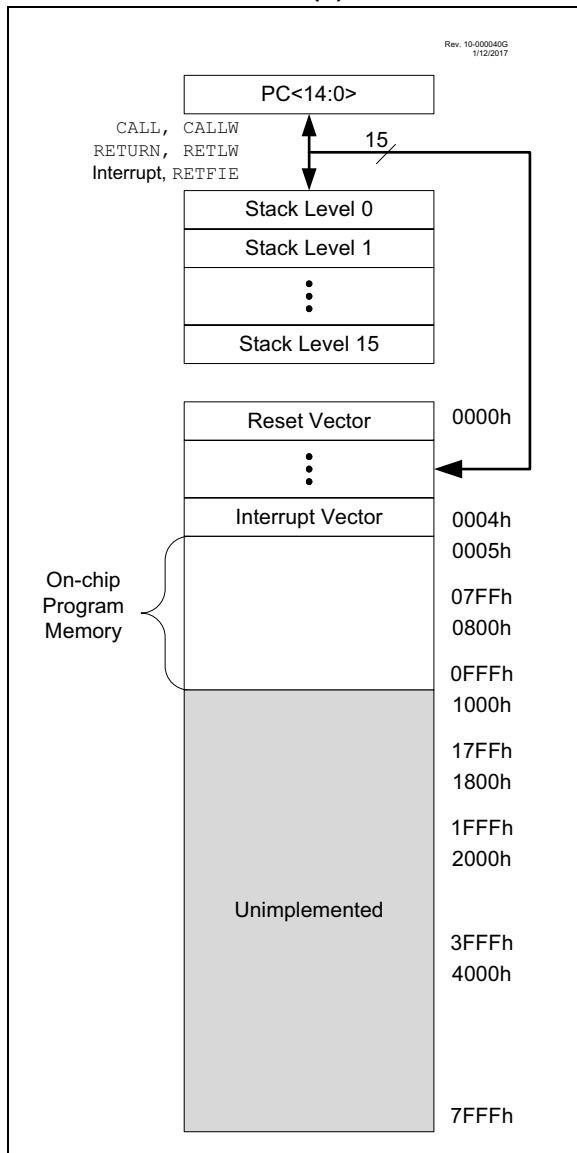
## 4.1 Program Memory Organization

The enhanced mid-range core has a 15-bit program counter capable of addressing 32K x 14 program memory space. [Table 4-1](#) shows the memory sizes implemented. The Reset vector is at 0000h and the interrupt vector is at 0004h (see [Figure 4-1](#)).

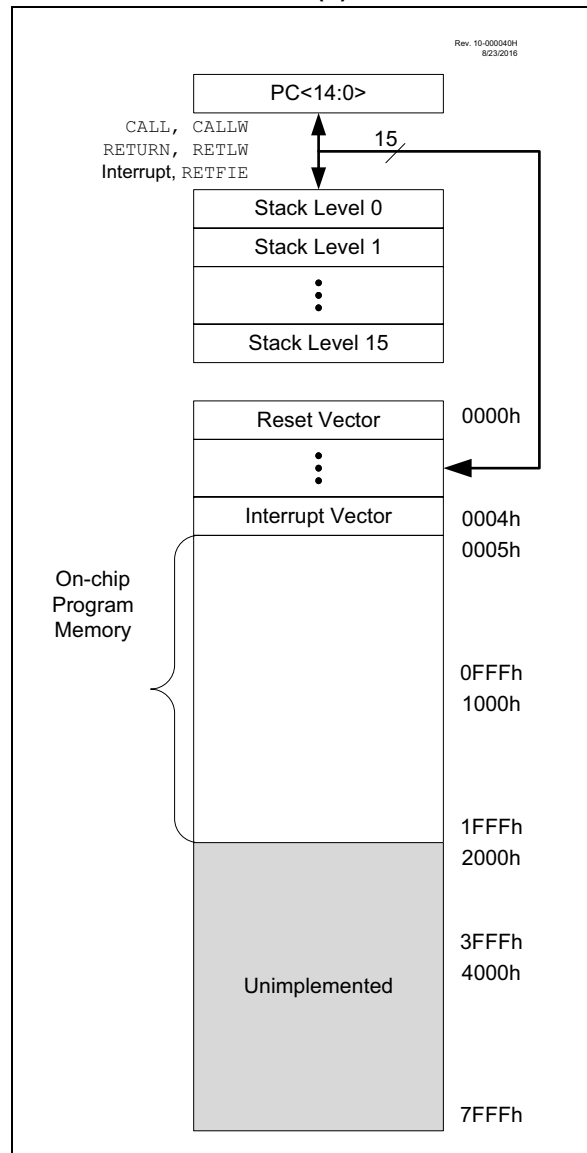
**TABLE 4-1: DEVICE SIZES AND ADDRESSES**

Device	Program Memory Size (Words)	Last Program Memory Address
PIC16(L)F15354	4096	0FFFh
PIC16(L)F15355	8192	1FFFh

**FIGURE 4-1: PROGRAM MEMORY MAP AND STACK FOR PIC16(L)F15354**



**FIGURE 4-2: PROGRAM MEMORY MAP AND STACK FOR PIC16(L)F15355**



## 4.1.1 READING PROGRAM MEMORY AS DATA

There are three 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. The third method is to use the NVMREG interface to access the program memory. For an example of NVMREG interface use, reference [Section 13.3, NVMREG Access](#).

### 4.1.1.1 RETLW Instruction

The RETLW instruction can be used to provide access to tables of constants. The recommended way to create such a table is shown in [Example 4-1](#).

#### EXAMPLE 4-1: 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.

### 4.1.1.2 Indirect Read with FSR

The program memory can be accessed as data by setting bit 7 of an FSRxH register and reading the matching INDFx register. The MOVLW 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 read the program memory via the FSR require one extra instruction cycle to complete. [Example 4-2](#) demonstrates reading the program memory via an FSR.

The HIGH directive will set bit 7 if a label points to a location in the program memory. This applies to the assembly code [Example 4-2](#) shown below.

## EXAMPLE 4-2: ACCESSING PROGRAM MEMORY VIA FSR

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

## 4.2 Memory Access Partition (MAP)

User Flash is partitioned into:

- Application Block
- Boot Block, and
- Storage Area Flash (SAF) Block

The user can allocate the memory usage by setting the  $\overline{\text{BBEN}}$  bit, selecting the size of the partition defined by  $\text{BBSIZE}[2:0]$  bits and enabling the Storage Area Flash by the  $\overline{\text{SAFEN}}$  bit of the Configuration Word (see [Register 5-4](#)). Refer to [Table 4-2](#) for the different user Flash memory partitions.

### 4.2.1 APPLICATION BLOCK

Default settings of the Configuration bits ( $\overline{\text{BBEN}} = 1$  and  $\overline{\text{SAFEN}} = 1$ ) assign all memory in the user Flash area to the Application Block.

### 4.2.2 BOOT BLOCK

If  $\overline{\text{BBEN}} = 1$ , the Boot Block is enabled and a specific address range is allotted as the Boot Block based on the value of the BBSIZE bits of Configuration Word ([Register 5-4](#)) and the sizes provided in [Table 5-1](#).

### 4.2.3 STORAGE AREA FLASH

Storage Area Flash (SAF) is enabled by clearing the  $\overline{\text{SAFEN}}$  bit of the Configuration Word in [Register 5-4](#). If enabled, the SAF block is placed at the end of memory and spans 128 words. If the Storage Area Flash (SAF) is enabled, the SAF area is not available for program execution. The 128 words of the SAF qualify as High Endurance Flash (HEF) memory.

### 4.2.4 MEMORY WRITE PROTECTION

All the memory blocks have corresponding write protection fuses  $\overline{\text{WRTAPP}}$ ,  $\overline{\text{WRTB}}$  and  $\overline{\text{WRTC}}$  bits in the Configuration Word 4 ([Register 5-4](#)). If write-protected locations are written from NVMCON registers, memory is not changed and the WRERR bit defined in Register 12-5 is set as explained in [Section 13.3.8 “WRERR Bit”](#).

### 4.2.5 MEMORY VIOLATION

A Memory Execution Violation Reset occurs while executing an instruction that has been fetched from outside a valid execution area, clearing the  $\overline{\text{MEMV}}$  bit. Refer to [Section 8.12 “Memory Execution Violation”](#) for the available valid program execution areas and the PCON1 register definition ([Register 8-3](#)) for  $\overline{\text{MEMV}}$  bit conditions.

**TABLE 4-2: MEMORY ACCESS PARTITION**

REG	Address	Partition			
		$\overline{\text{BBEN}} = 1$ $\overline{\text{SAFEN}} = 1$	$\overline{\text{BBEN}} = 1$ $\overline{\text{SAFEN}} = 0$	$\overline{\text{BBEN}} = 0$ $\overline{\text{SAFEN}} = 1$	$\overline{\text{BBEN}} = 0$ $\overline{\text{SAFEN}} = 0$
PFM	00 0000h ••• Last Boot Block Memory Address	APPLICATION BLOCK <sup>(4)</sup>	APPLICATION BLOCK <sup>(4)</sup>	BOOT BLOCK <sup>(4)</sup>	BOOT BLOCK <sup>(4)</sup>
	Last Boot Block Memory Address + 1 <sup>(1)</sup> ••• Last Program Memory Address - 80h			APPLICATION BLOCK <sup>(4)</sup>	APPLICATION BLOCK <sup>(4)</sup>
	Last Program Memory Address - 7Fh <sup>(2)</sup> ••• Last Program Memory Address		SAF <sup>(4)</sup>	APPLICATION BLOCK <sup>(4)</sup>	SAF <sup>(4)</sup>
	CONFIG		Config Memory Address <sup>(3)</sup>	CONFIG	

- Note 1:** Last Boot Block Memory Address is based on BBSIZE<2:0> given in [Table 5-1](#).
- Note 2:** Last Program Memory Address is the Flash size given in [Table 4-1](#).
- Note 3:** Config Memory Address are the address locations of the Configuration Words given in [Table 13-2](#).
- Note 4:** Each memory block has a corresponding write protection fuse defined by the  $\overline{\text{WRTAPP}}$ ,  $\overline{\text{WRTB}}$  and  $\overline{\text{WRTC}}$  bits in the Configuration Word ([Register 5-4](#)).

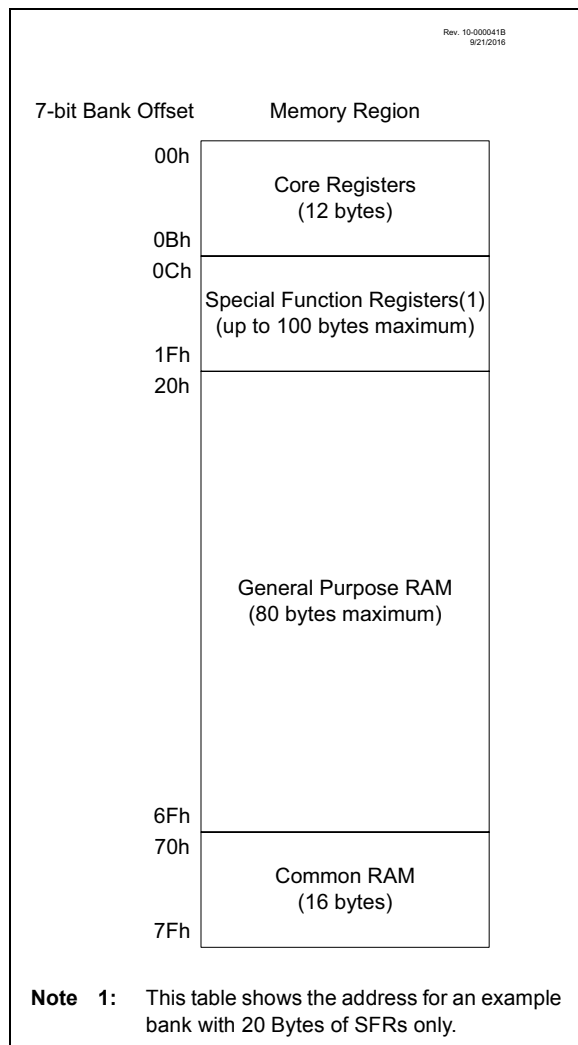


## 4.3 Data Memory Organization

The data memory is partitioned into 64 memory banks with 128 bytes in each bank. Each bank consists of:

- 12 core registers
- Up to 100 Special Function Registers (SFR)
- Up to 80 bytes of General Purpose RAM (GPR)
- 16 bytes of common RAM

**FIGURE 4-3: BANKED MEMORY PARTITIONING**



### 4.3.1 BANK SELECTION

The active bank is selected by writing the bank number into the Bank Select Register (BSR). 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 4.6 “Indirect Addressing”](#) for more information.

Data memory uses a 13-bit address. The upper six bits of the address define the Bank address and the lower seven bits select the registers/RAM in that bank.

### 4.3.2 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 4-3](#).

**TABLE 4-3: 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

## 4.3.2.1 STATUS Register

The STATUS register, shown in [Register 4-1](#), 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 bits <4:3> and <1:0>, 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 36.0 "Instruction Set Summary"](#).

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

**REGISTER 4-1: 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	$\overline{DC}^{(1)}$	$\overline{C}^{(1)}$	
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     **TO:** Time-Out bit  
1 = After power-up, `CLRWDT` instruction or `SLEEP` instruction  
0 = A WDT time-out occurred
- bit 3     **PD:** Power-Down bit  
1 = After power-up or by the `CLRWDT` instruction  
0 = By execution of the `SLEEP` instruction
- bit 2     **Z:** Zero bit  
1 = The result of an arithmetic or logic operation is zero  
0 = The result of an arithmetic or logic operation is not zero
- bit 1     **DC:** Digit Carry/Digit Borrow bit (`ADDWF`, `ADDLW`, `SUBLW`, `SUBWF` instructions)<sup>(1)</sup>  
1 = A carry-out from the 4th low-order bit of the result occurred  
0 = No carry-out from the 4th low-order bit of the result
- bit 0     **C:** Carry/Borrow bit<sup>(1)</sup> (`ADDWF`, `ADDLW`, `SUBLW`, `SUBWF` instructions)<sup>(1)</sup>  
1 = A carry-out from the Most Significant bit of the result occurred  
0 = No carry-out from the Most Significant bit of the result occurred

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

## 4.3.3 SPECIAL FUNCTION REGISTER

The Special Function Registers are registers used by the application to control the desired operation of peripheral functions in the device. The Special Function Registers occupy the 20 bytes of the data banks 0-59 and 100 bytes of the data banks 60-63, after the core registers.

The SFRs associated with the operation of the peripherals are described in the appropriate peripheral chapter of this data sheet.

## 4.3.4 GENERAL PURPOSE RAM

There are up to 80 bytes of GPR in each data memory bank.

### 4.3.4.1 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 4.6.2 “Linear Data Memory”](#) for more information.

## 4.3.5 COMMON RAM

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

## 4.3.6 DEVICE MEMORY MAPS

The memory maps are as shown in [Table 4-4](#) through [Table 4-9](#).

**TABLE 4-4: PIC16(L)F15354/55 MEMORY MAP, BANKS 0-7**

BANK 0		BANK 1		BANK 2		BANK 3		BANK 4		BANK 5		BANK 6		BANK 7							
000h	Core Register (Table 4-3)	080h	Core Register (Table 4-3)	100h	Core Register (Table 4-3)	180h	Core Register (Table 4-3)	200h	Core Register (Table 4-3)	280h	Core Register (Table 4-3)	300h	Core Register (Table 4-3)	380h	Core Register (Table 4-3)						
00Bh	—	08Bh	—	10Bh	—	18Bh	—	20Bh	—	28Bh	—	30Bh	—	38Bh	—						
00Ch	PORTA	08Ch	—	10Ch	—	18Ch	SSP1BUF	20Ch	TMR1L	28Ch	TMR2	30Ch	CCPR1L	38Ch	PWM6DCL						
00Dh	PORTB	08Dh	—	10Dh	—	18Dh	SSP1ADD	20Dh	TMR1H	28Dh	PR2	30Dh	CCPR1H	38Dh	PWM6DCH						
00Eh	PORTC	08Eh	—	10Eh	—	18Eh	SSP1MASK	20Eh	T1CON	28Eh	T2CON	30Eh	CCP1CON	38Eh	PWM6CON						
00Fh	—	08Fh	—	10Fh	—	18Fh	SSP1STAT	20Fh	T1GCON	28Fh	T2HLT	30Fh	CCP1CAP	38Fh	—						
010h	PORTE	090h	—	110h	—	190h	SSP1CON1	210h	T1GATE	290h	T2CLK	310h	CCPR2L	390h	—						
011h	—	091h	—	111h	—	191h	SSP1CON2	211h	T1CLK	291h	T2ERS	311h	CCPR2H	391h	—						
012h	TRISA	092h	—	112h	—	192h	SSP1CON3	212h	—	292h	—	312h	CCP2CON	392h	—						
013h	TRISB	093h	—	113h	—	193h	—	213h	—	293h	—	313h	CCP2CAP	393h	—						
014h	TRISC	094h	—	114h	—	194h	—	214h	—	294h	—	314h	PWM3DCL	394h	—						
015h	—	095h	—	115h	—	195h	—	215h	—	295h	—	315h	PWM3DCH	395h	—						
016h	TRISE	096h	—	116h	—	196h	SSP2BUF	216h	—	296h	—	316h	PWM3CON	396h	—						
017h	—	097h	—	117h	—	197h	SSP2ADD	217h	—	297h	—	317h	—	397h	—						
018h	LATA	098h	—	118h	—	198h	SSP2MASK	218h	—	298h	—	318h	PWM4DCL	398h	—						
019h	LATB	099h	—	119h	RC1REG1	199h	SSP2STAT	219h	—	299h	—	319h	PWM4DCH	399h	—						
01Ah	LATC	09Ah	—	11Ah	TX1REG1	19Ah	SSP2CON1	21Ah	—	29Ah	—	31Ah	PWM4CON	39Ah	—						
01Bh	—	09Bh	ADRESL	11Bh	SP1BRG1L	19Bh	SSP2CON2	21Bh	—	29Bh	—	31Bh	—	39Bh	—						
01Ch	LATE	09Ch	ADRESH	11Ch	SP1BRG1H	19Ch	SSP2CON3	21Ch	—	29Ch	—	31Ch	PWM5DCL	39Ch	—						
01Dh	—	09Dh	ADCON0	11Dh	RC1STA1	19Dh	—	21Dh	—	29Dh	—	31Dh	PWM5DCH	39Dh	—						
01Eh	—	09Eh	ADCON1	11Eh	TX1STA1	19Eh	—	21Eh	—	29Eh	—	31Eh	PWM5CON	39Eh	—						
01Fh	—	09Fh	ADACT	11Fh	BAUD1CON1	19Fh	—	21Fh	—	29Fh	—	31Fh	—	39Fh	—						
020h	General Purpose Register 96 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	General Purpose Register 80 Bytes <sup>(2)</sup>						
																		32Fh	General Purpose Register 64 Bytes <sup>(2)</sup>		
																			330h		
																			36Fh		
																			370h		
07Fh		0FFh	Common RAM Accesses 70h-7Fh	17Fh	Common RAM Accesses 70h-7Fh	1FFh	Common RAM Accesses 70h-7Fh	27Fh	Common RAM Accesses 70h-7Fh	2FFh	Common RAM Accesses 70h-7Fh	37Fh	Common RAM Accesses 70h-7Fh	3FFh	Common RAM Accesses 70h-7Fh						

**Note 1:** Unimplemented locations read as '0'.

**Note 2:** Present only in PIC16(L)F15355.

TABLE 4-5: PIC16(L)F15354/55 MEMORY MAP, BANKS 8-15

BANK 8		BANK 9		BANK 10		BANK 11		BANK 12		BANK 13		BANK 14		BANK 15	
400h	Core Register (Table 4-3)	480h	Core Register (Table 4-3)	500h	Core Register (Table 4-3)	580h	Core Register (Table 4-3)	600h	Core Register (Table 4-3)	680h	Core Register (Table 4-3)	700h	Core Register (Table 4-3)	780h	Core Register (Table 4-3)
40Bh	—	48Bh	—	50Bh	—	58Bh	NCO1ACCL	60Bh	CWG1CLK	68Bh	—	70Bh	PIR0	78Bh	—
40Ch	—	48Ch	—	50Ch	—	58Ch	NCO1ACCH	60Ch	CWG1DAT	68Ch	—	70Ch	PIR1	78Ch	—
40Dh	—	48Dh	—	50Dh	—	58Dh	NCO1ACCU	60Dh	CWG1DBR	68Dh	—	70Dh	PIR2	78Dh	—
40Eh	—	48Eh	—	50Eh	—	58Eh	NCO1INCL	60Eh	CWG1DBF	68Eh	—	70Eh	PIR3	78Eh	—
40Fh	—	48Fh	—	50Fh	—	58Fh	NCO1INCH	60Fh	CWG1CON0	68Fh	—	70Fh	PIR4	78Fh	—
410h	—	490h	—	510h	—	590h	NCO1INCU	610h	CWG1CON1	690h	—	710h	PIR5	790h	—
411h	—	491h	—	511h	—	591h	NCO1CON	611h	CWG1AS0	691h	—	711h	PIR6	791h	—
412h	—	492h	—	512h	—	592h	NCO1CLK	612h	CWG1AS1	692h	—	712h	PIR7	792h	—
413h	—	493h	—	513h	—	593h	—	613h	CWG1STR	693h	—	713h	—	793h	—
414h	—	494h	—	514h	—	594h	—	614h	—	694h	—	714h	—	794h	—
415h	—	495h	—	515h	—	595h	—	615h	—	695h	—	715h	—	795h	—
416h	—	496h	—	516h	—	596h	—	616h	—	696h	—	716h	PIE0	796h	PMD0
417h	—	497h	—	517h	—	597h	—	617h	—	697h	—	717h	PIE1	797h	PMD1
418h	—	498h	—	518h	—	598h	—	618h	—	698h	—	718h	PIE2	798h	PMD2
419h	—	499h	—	519h	—	599h	—	619h	—	699h	—	719h	PIE3	799h	PMD3
41Ah	—	49Ah	—	51Ah	—	59Ah	—	61Ah	—	69Ah	—	71Ah	PIE4	79Ah	PMD4
41Bh	—	49Bh	—	51Bh	—	59Bh	—	61Bh	—	69Bh	—	71Bh	PIE5	79Bh	PMD5
41Ch	—	49Ch	—	51Ch	—	59Ch	TMR0	61Ch	—	69Ch	—	71Ch	PIE6	79Ch	—
41Dh	—	49Dh	—	51Dh	—	59Dh	PR0	61Dh	—	69Dh	—	71Dh	PIE7	79Dh	—
41Eh	—	49Eh	—	51Eh	—	59Eh	TMR0CON0	61Eh	—	69Eh	—	71Eh	—	79Eh	—
41Fh	—	49Fh	—	51Fh	—	59Fh	TMR0CON1	61Fh	—	69Fh	—	71Fh	—	79Fh	—
420h	General	4A0h	General	520h	General	5A0h	General	620h	General Purpose Register 48 Bytes <sup>(2)</sup>	6A0h	Unimplemented Read as '0'	720h	Unimplemented Read as '0'	7A0h	Unimplemented Read as '0'
	Purpose Register 80 Bytes <sup>(2)</sup>		Purpose Register 80 Bytes <sup>(2)</sup>		Purpose Register 80 Bytes <sup>(2)</sup>		Purpose Register 80 Bytes <sup>(2)</sup>	64Fh	Unimplemented Read as '0'						
46Fh	—	4EFh	—	56Fh	—	5EFh	—	66Fh	—	6EFh	—	76Fh	—	7EFh	—
470h	Common RAM Accesses 70h-7Fh	4F0h	Common RAM Accesses 70h-7Fh	570h	Common RAM Accesses 70h-7Fh	5F0h	Common RAM Accesses 70h-7Fh	670h	Common RAM Accesses 70h-7Fh	6F0h	Common RAM Accesses 70h-7Fh	770h	Common RAM Accesses 70h-7Fh	7F0h	Common RAM Accesses 70h-7Fh
47Fh	—	4FFh	—	57Fh	—	5FFh	—	67Fh	—	6FFh	—	77Fh	—	7FFh	—

**Note 1:** Unimplemented locations read as '0'.

**Note 2:** Present only in PIC16(L)F15355.

**TABLE 4-6: PIC16(L)F15354/55 MEMORY MAP, BANKS 16-23**

BANK 16		BANK 17		BANK 18		BANK 19		BANK 20		BANK 21		BANK 22		BANK 23	
800h	Core Register (Table 4-3)	880h	Core Register (Table 4-3)	900h	Core Register (Table 4-3)	980h	Core Register (Table 4-3)	A00h	Core Register (Table 4-3)	A80h	Core Register (Table 4-3)	B00h	Core Register (Table 4-3)	B80h	Core Register (Table 4-3)
80Bh	—	88Bh	—	90Bh	—	98Bh	—	A0Bh	—	A8Bh	—	B0Bh	—	B8Bh	—
80Ch	WDTCON0	88Ch	CPUDOZE	90Ch	FVRCON	98Ch	—	A0Ch	—	A8Ch	—	B0Ch	—	B8Ch	—
80Dh	WDTCON1	88Dh	OSCCON1	90Dh	—	98Dh	—	A0Dh	—	A8Dh	—	B0Dh	—	B8Dh	—
80Eh	WDTL	88Eh	OSCCON2	90Eh	DAC1CON0	98Eh	—	A0Eh	—	A8Eh	—	B0Eh	—	B8Eh	—
80Fh	WDTH	88Fh	OSCCON3	90Fh	DAC1CON1	98Fh	CMOUT	A0Fh	—	A8Fh	—	B0Fh	—	B8Fh	—
810h	WDTU	890h	OSCSTAT1	910h	—	990h	CM1CON0	A10h	—	A90h	—	B10h	—	B90h	—
811h	BORCON	891h	OSCCEN	911h	—	991h	CM1CON1	A11h	—	A91h	—	B11h	—	B91h	—
812h	VREGCON <sup>(2)</sup>	892h	OSCTUNE	912h	—	992h	CM1NCH	A12h	—	A92h	—	B12h	—	B92h	—
813h	PCON0	893h	OSCFRQ	913h	—	993h	CM1PCH	A13h	—	A93h	—	B13h	—	B93h	—
814h	PCON1	894h	—	914h	—	994h	CM2CON0	A14h	—	A94h	—	B14h	—	B94h	—
815h	—	895h	CLKRCON	915h	—	995h	CM2CON1	A15h	—	A95h	—	B15h	—	B95h	—
816h	—	896h	CLKCLK	916h	—	996h	CM2NCH	A16h	—	A96h	—	B16h	—	B96h	—
817h	—	897h	—	917h	—	997h	CM2PCH	A17h	—	A97h	—	B17h	—	B97h	—
818h	—	898h	—	918h	—	998h	—	A18h	—	A98h	—	B18h	—	B98h	—
819h	—	899h	—	919h	—	999h	—	A19h	RC2REG	A99h	—	B19h	—	B99h	—
81Ah	NVMADRL	89Ah	—	91Ah	—	99Ah	—	A1Ah	TX2REG	A9Ah	—	B1Ah	—	B9Ah	—
81Bh	NVMADRH	89Bh	—	91Bh	—	99Bh	—	A1Bh	SP2BRGL	A9Bh	—	B1Bh	—	B9Bh	—
81Ch	NVMDATL	89Ch	—	91Ch	—	99Ch	—	A1Ch	SP2BRGH	A9Ch	—	B1Ch	—	B9Ch	—
81Dh	NVMDATH	89Dh	—	91Dh	—	99Dh	—	A1Dh	RC2STA	A9Dh	—	B1Dh	—	B9Dh	—
81Eh	NVMCON1	89Eh	—	91Eh	—	99Eh	—	A1Eh	TX2STA	A9Eh	—	B1Eh	—	B9Eh	—
81Fh	NVMCON2	89Fh	—	91Fh	ZCDCON	99Fh	—	A1Fh	BAUD2CON	A9Fh	—	B1Fh	—	B9Fh	—
820h	Unimplemented Read as '0'	8A0h	Unimplemented Read as '0'	920h	Unimplemented Read as '0'	9A0h	Unimplemented Read as '0'	A20h	Unimplemented Read as '0'	AA0h	Unimplemented Read as '0'	B20h	Unimplemented Read as '0'	BA0h	Unimplemented Read as '0'
86Fh	—	8EFh	—	96Fh	—	9EFh	—	A6Fh	—	A6Fh	—	B6Fh	—	BEFh	—
870h	Common RAM Accesses 70h-7Fh	8F0h	Common RAM Accesses 70h-7Fh	970h	Common RAM Accesses 70h-7Fh	9F0h	Common RAM Accesses 70h-7Fh	A70h	Common RAM Accesses 70h-7Fh	AF0h	Common RAM Accesses 70h-7Fh	B70h	Common RAM Accesses 70h-7Fh	BF0h	Common RAM Accesses 70h-7Fh
87Fh	—	8FFh	—	97Fh	—	9FFh	—	A7Fh	—	AFh	—	B7Fh	—	BFh	—

**Note** 1: Unimplemented locations read as '0'.  
2: Register not implemented on LF devices.

TABLE 4-7: PIC16(L)F15354/55 MEMORY MAP, BANKS 56-63

BANK 56		BANK 57		BANK 58		BANK 59		BANK 60		BANK 61		BANK 62		BANK 63	
1C00h	Core Register (Table 4-3)	1C80h	Core Register (Table 4-3)	1D00h	Core Register (Table 4-3)	1D80h	Core Register (Table 4-3)	1E00h	Core Register (Table 4-3)	1E80h	Core Register (Table 4-3)	1F00h	Core Register (Table 4-3)	1F80h	Core Register (Table 4-3)
1C0Bh	—	1C8Bh	—	1D0Bh	—	1D8Bh	—	1E0Bh	—	1E8Bh	—	1F0Bh	—	1F8Bh	—
1C0Ch	—	1C8Ch	—	1D0Ch	—	1D8Ch	—	1E0Ch	—	1E8Ch	—	1F0Ch	—	1F8Ch	—
1C0Dh	—	1C8Dh	—	1D0Dh	—	D8Dh1	—	1E0Dh	—	1E8Dh	—	1F0Dh	—	1F8Dh	—
1C0Eh	—	1C8Eh	—	1D0Eh	—	1D8Eh	—	1E0Eh	—	1E8Eh	—	1F0Eh	—	1F8Eh	—
1C0Fh	—	1C8Fh	—	1D0Fh	—	1D8Fh	—	1E0Fh	—	1E8Fh	—	1F0Fh	—	1F8Fh	—
1C10h	—	1C90h	—	1D10h	—	1D90h	—	1E10h	—	1E90h	—	1F10h	—	1F90h	—
1C11h	—	1C91h	—	1D11h	—	1D91h	—	1E11h	—	1E91h	—	1F11h	—	1F91h	—
1C12h	—	1C92h	—	1D12h	—	1D92h	—	1E12h	—	1E92h	—	1F12h	—	1F92h	—
1C13h	—	1C93h	—	1D13h	—	1D93h	—	1E13h	—	1E93h	—	1F13h	—	1F93h	—
1C14h	—	1C94h	—	1D14h	—	1D94h	—	1E14h	—	1E94h	—	1F14h	—	1F94h	—
1C15h	—	1C95h	—	1D15h	—	1D95h	—	1E15h	—	1E95h	—	1F15h	—	1F95h	—
1C16h	—	1C96h	—	1D16h	—	1D96h	—	1E16h	—	1E96h	—	1F16h	—	1F96h	—
1C17h	—	1C97h	—	1D17h	—	1D97h	—	1E17h	—	1E97h	—	1F17h	—	1F97h	—
1C18h	—	1C98h	—	1D18h	—	1D98h	—	1E18h	—	1E98h	—	1F18h	—	1F98h	—
1C19h	—	1C99h	—	1D19h	—	1D99h	—	1E19h	—	1E99h	—	1F19h	—	1F99h	—
1C1Ah	—	1C9Ah	—	1D1Ah	—	1D9Ah	—	1E1Ah	—	1E9Ah	—	1F1Ah	—	1F9Ah	—
1C1Bh	—	1C9Bh	—	1D1Bh	—	1D9Bh	—	1E1Bh	—	1E9Bh	—	1F1Bh	—	1F9Bh	—
1C1Ch	—	1C9Ch	—	1D1Ch	—	1D9Ch	—	1E1Ch	—	1E9Ch	—	1F1Ch	—	1F9Ch	—
1C1Dh	—	1C9Dh	—	1D1Dh	—	1D9Dh	—	1E1Dh	—	1E9Dh	—	1F1Dh	—	1F9Dh	—
1C1Eh	—	1C9Eh	—	1D1Eh	—	1D9Eh	—	1E1Eh	—	1E9Eh	—	1F1Eh	—	1F9Eh	—
1C1Fh	—	1C9Fh	—	1D1Fh	—	1D9Fh	—	1E1Fh	—	1E9Fh	—	1F1Fh	—	1F9Fh	—
1C20h	—	1CA0h	—	1D20h	—	1DA0h	—	1E20h	—	1EA0h	—	1F20h	—	1FA0h	—
	Unimplemented Read as '0'		Unimplemented Read as '0'		Unimplemented Read as '0'		Unimplemented Read as '0'								
1C6Fh	—	1CEFh	—	1D6Fh	—	1DEFh	—	1E6Fh	—	1EEFh	—	1F6Fh	—	1FEFh	—
1C70h	Common RAM Accesses 70h-7Fh	1CF0h	Common RAM Accesses 70h-7Fh	1D70h	Common RAM Accesses 70h-7Fh	1DF0h	Common RAM Accesses 70h-7Fh	1E70h	Common RAM Accesses 70h-7Fh	1EF0h	Common RAM Accesses 70h-7Fh	1F70h	Common RAM Accesses 70h-7Fh	1FF0h	Common RAM Accesses 70h-7Fh
1C7Fh	—	1CFFh	—	1D7Fh	—	1DFFh	—	1E7Fh	—	1EFFh	—	1F7Fh	—	1FFFh	—

**Note 1:** Unimplemented locations read as '0'.

**Note 2:** The banks 24-55 have been omitted from the tables in the data sheet since the banks have unimplemented registers.

# PIC16(L)F15354/55

**TABLE 4-8: PIC16(L)F15354/55 MEMORY MAP, BANKS 60, 61, 62, AND 63**

Bank 60		Bank 61		Bank 62		Bank 63	
1E0Ch	—	1E8Ch	—	1F0Ch	—	1F8Ch	—
1E0Dh	—	1E8Dh	—	1F0Dh	—	1F8Dh	—
1E0Eh	—	1E8Eh	—	1F0Eh	—	1F8Eh	—
1E0Fh	CLCDATA	1E8Fh	PPSLOCK	1F0Fh	—	1F8Fh	—
1E10h	CLC1CON	1E90h	INTPPS	1F10h	RA0PPS	1F90h	—
1E11h	CLC1POL	1E91h	T0CKIPPS	1F11h	RA1PPS	1F91h	—
1E12h	CLC1SEL0	1E92h	T1CKIPPS	1F12h	RA2PPS	1F92h	—
1E13h	CLC1SEL1	1E93h	T1GPPS	1F13h	RA3PPS	1F93h	—
1E14h	CLC1SEL2	1E94h	—	1F14h	RA4PPS	1F94h	—
1E15h	CLC1SEL3	1E95h	—	1F15h	RA5PPS	1F95h	—
1E16h	CLC1GLS0	1E96h	—	1F16h	RA6PPS	1F96h	—
1E17h	CLC1GLS1	1E97h	—	1F17h	RA7PPS	1F97h	—
1E18h	CLC1GLS2	1E98h	—	1F18h	RB0PPS	1F98h	—
1E19h	CLC1GLS3	1E99h	—	1F19h	RB1PPS	1F99h	—
1E1Ah	CLC2CON	1E9Ah	—	1F1Ah	RB2PPS	1F9Ah	—
1E1Bh	CLC2POL	1E9Bh	—	1F1Bh	RB3PPS	1F9Bh	—
1E1Ch	CLC2SEL0	1E9Ch	T2INPPS	1F1Ch	RB4PPS	1F9Ch	—
1E1Dh	CLC2SEL1	1E9Dh	—	1F1Dh	RB5PPS	1F9Dh	—
1E1Eh	CLC2SEL2	1E9Eh	—	1F1Eh	RB6PPS	1F9Eh	—
1E1Fh	CLC2SEL3	1E9Fh	—	1F1Fh	RB7PPS	1F9Fh	—
1E20h	CLC2GLS0	1EA0h	—	1F20h	RC0PPS	1FA0h	—
1E21h	CLC2GLS1	1EA1h	CCP1PPS	1F21h	RC1PPS	1FA1h	—
1E22h	CLC2GLS2	1EA2h	CCP2PPS	1F22h	RC2PPS	1FA2h	—
1E23h	CLC2GLS3	1EA3h	—	1F23h	RC3PPS	1FA3h	—
1E24h	CLC3CON	1EA4h	—	1F24h	RC4PPS	1FA4h	—
1E25h	CLC3POL	1EA5h	—	1F25h	RC5PPS	1FA5h	—
1E26h	CLC3SEL0	1EA6h	—	1F26h	RC6PPS	1FA6h	—
1E27h	CLC3SEL1	1EA7h	—	1F27h	RC7PPS	1FA7h	—
1E28h	CLC3SEL2	1EA8h	—	1F28h	—	1FA8h	—
1E29h	CLC3SEL3	1EA9h	—	1F29h	—	1FA9h	—
1E2Ah	CLC3GLS0	1EAAh	—	1F2Ah	—	1FAAh	—
1E2Bh	CLC3GLS1	1EABh	—	1F2Bh	—	1FABh	—
1E2Ch	CLC3GLS2	1EACH	—	1F2Ch	—	1FACh	—
1E2Dh	CLC3GLS3	1EADh	—	1F2Dh	—	1FADh	—
1E2Eh	CLC4CON	1EAEh	—	1F2Eh	—	1FAEh	—
1E2Fh	CLC4POL	1EAFh	—	1F2Fh	—	1FAFh	—
1E30h	CLC4SEL0	1EB0h	—	1F30h	—	1FB0h	—
1E31h	CLC4SEL1	1EB1h	CWG1PPS	1F31h	—	1FB1h	—
1E32h	CLC4SEL2	1EB2h	—	1F32h	—	1FB2h	—
1E33h	CLC4SEL3	1EB3h	—	1F33h	—	1FB3h	—
1E34h	CLC4GLS0	1EB4h	—	1F34h	—	1FB4h	—
1E35h	CLC4GLS1	1EB5h	—	1F35h	—	1FB5h	—
1E36h	CLC4GLS2	1EB6h	—	1F36h	—	1FB6h	—
1E37h	CLC4GLS3	1EB7h	—	1F37h	—	1FB7h	—
1E38h	—	1EB8h	—	1F38h	ANSELA	1FB8h	—
1E39h	—	1EB9h	—	1F39h	WPUA	1FB9h	—
1E3Ah	—	1EBAh	—	1F3Ah	ODCONA	1FBAh	—
1E3Bh	—	1EBBh	CLCIN0PPS	1F3Bh	SLRCONA	1FBBh	—
1E3Ch	—	1EBCh	CLCIN1PPS	1F3Ch	INLVLA	1FBCCh	—
1E3Dh	—	1EBDh	CLCIN2PPS	1F3Dh	IOCAP	1FBDh	—
1E3Eh	—	1EBEh	CLCIN3PPS	1F3Eh	IOCAN	1FBEh	—
1E3Fh	—	1EBFh	—	1F3Fh	IOCAF	1FBFh	—
1E40h	—	1EC0h	—	1F40h	—	1FC0h	—
1E41h	—	1EC1h	—	1F41h	—	1FC1h	—
1E42h	—	1EC2h	—	1F42h	—	1FC2h	—

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



# PIC16(L)F15354/55

**TABLE 4-8: PIC16(L)F15354/55 MEMORY MAP, BANKS 60, 61, 62, AND 63 (CONTINUED)**

Bank 60		Bank 61		Bank 62		Bank 63	
1E43h	—	1EC3h	ADACTPPS	1F43h	ANSELB	1FC3h	—
1E44h	—	1EC4h	—	1F44h	WPUB	1FC4h	—
1E45h	—	1EC5h	SSP1CLKPPS	1F45h	ODCONB	1FC5h	—
1E46h	—	1EC6h	SSP1DATPPS	1F46h	SLRCONB	1FC6h	—
1E47h	—	1EC7h	SSP1SSPPS	1F47h	INLVLB	1FC7h	—
1E48h	—	1EC8h	SSP2CLKPPS	1F48h	IOCBP	1FC8h	—
1E49h	—	1EC9h	SSP2DATPPS	1F49h	IOCBN	1FC9h	—
1E4Ah	—	1ECAh	SSP2SSPPS	1F4Ah	IOCBF	1FCAh	—
1E4Bh	—	1ECBh	RX1DTPPS	1F4Bh	—	1FCBh	—
1E4Ch	—	1ECh	TX1CKPPS	1F4Ch	—	1FCh	—
1E4Dh	—	1ECDh	RX2DTPPS	1F4Dh	—	1FCDh	—
1E4Eh	—	1ECEh	TX2CKPPS	1F4Eh	ANSELC	1FCEh	—
1E4Fh	—	1ECFh	—	1F4Fh	WPUC	1FCFh	—
1E50h	—	1ED0h	—	1F50h	ODCONC	1FD0h	—
1E51h	—	1ED1h	—	1F51h	SLRCONC	1FD1h	—
1E52h	—	1ED2h	—	1F52h	INLVLC	1FD2h	—
1E53h	—	1ED3h	—	1F53h	IOCCP	1FD3h	—
1E54h	—	1ED4h	—	1F54h	IOCCN	1FD4h	—
1E55h	—	1ED5h	—	1F55h	IOCCF	1FD5h	—
1E56h	—	1ED6h	—	1F56h	—	1FD6h	—
1E57h	—	1ED7h	—	1F57h	—	1FD7h	—
1E58h	—	1ED8h	—	1F58h	—	1FD8h	—
1E59h	—	1ED9h	—	1F59h	—	1FD9h	—
1E5Ah	—	1EDAh	—	1F5Ah	—	1FDAh	—
1E5Bh	—	1EDBh	—	1F5Bh	—	1FDBh	—
1E5Ch	—	1EDCh	—	1F5Ch	—	1FDC	—
1E5Dh	—	1EDDh	—	1F5Dh	—	1FDDh	—
1E5Eh	—	1EDEh	—	1F5Eh	—	1FDEh	—
1E5Fh	—	1EDFh	—	1F5Fh	—	1FDFh	—
1E60h	—	1EE0h	—	1F60h	—	1FE0h	—
1E61h	—	1EE1h	—	1F61h	—	1FE1h	—
1E62h	—	1EE2h	—	1F62h	—	1FE2h	—
1E63h	—	1EE3h	—	1F63h	—	1FE3h	BSR_ICD_SHAD
1E64h	—	1EE4h	—	1F64h	—	1FE4h	STATUS_SHAD
1E65h	—	1EE5h	—	1F65h	WPUE	1FE5h	WREG_SHAD
1E66h	—	1EE6h	—	1F66h	—	1FE6h	BSR_SHAD
1E67h	—	1EE7h	—	1F67h	—	1FE7h	PCLATH_SHAD
1E68h	—	1EE8h	—	1F68h	INLVLE	1FE8h	FSR0L_SHAD
1E69h	—	1EE9h	—	1F69h	IOCEP	1FE9h	FSR0H_SHAD
1E6Ah	—	1EEAh	—	1F6Ah	IOCEN	1FEAh	FSR1L_SHAD
1E6Bh	—	1EEBh	—	1F6Bh	IOCEF	1FEBh	FSR1H_SHAD
1E6Ch	—	1EECh	—	1F6Ch	—	1FEC	—
1E6Dh	—	1EEDh	—	1F6Dh	—	1FEDh	STKPTR
1E6Eh	—	1EEEh	—	1F6Eh	—	1FEEh	TOSL
1E6Fh	—	1EEFh	—	1F6Fh	—	1FEFh	TOSH

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

# PIC16(L)F15354/55

**TABLE 4-9: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (ALL BANKS)**

Bank Offset Bank 0-Bank 63	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR	
<b>All Banks</b>												
x00h or x80h	INDF0	Addressing this location uses contents of FSR0H/FSR0L to address data memory (not a physical register)								xxxx xxxx	xxxx xxxx	
x01h or x81h	INDF1	Addressing this location uses contents of FSR1H/FSR1L to address data memory (not a physical register)								xxxx xxxx	xxxx xxxx	
x02h or x82h	PCL	PCL								0000 0000	0000 0000	
x03h or x83h	STATUS	—	—	—	$\overline{TO}$	$\overline{PD}$	Z	DC	C	---1 1000	---q quuu	
x04h or x84h	FSR0L	FSR0L	Indirect Data Memory Address 0 Low Pointer								0000 0000	uuuu uuuu
x05h or x85h	FSR0H	FSR0H	Indirect Data Memory Address 0 High Pointer								0000 0000	0000 0000
x06h or x86h	FSR1L	FSR1L	Indirect Data Memory Address 1 Low Pointer								0000 0000	uuuu uuuu
x07h or x87h	FSR1H	FSR1H	Indirect Data Memory Address 1 High Pointer								0000 0000	0000 0000
x08h or x88h	BSR	—	—	BSR<5:0>						--00 0000	--00 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	—	—	—	—	—	INTEDG	00-- ---1	00-- ---1	

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

**Note 1:** These Registers can be accessed from any bank.

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR
<b>Bank 0</b>											
CPU CORE REGISTERS; see <a href="#">Table 4-9</a> for specifics											
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	—	Unimplemented								---- ----	---- ----
012h	TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	1111 1111	1111 1111
013h	TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111	1111 1111
014h	TRISC	TRISC7 <sup>(1)</sup>	TRISC6 <sup>(1)</sup>	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	1111 1111	1111 1111
015h	—	Unimplemented								—	—
016h	TRISE	—	—	—	—	— <sup>(1)</sup>	—	—	—	---- 1---	---- 1---
017h	—	Unimplemented								—	—
018h	LATA	LATA7	LATA6	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	xxxx xxxx	uuuu uuuu
019h	LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	xxxx xxxx	uuuu uuuu
01Ah	LATC	LATC7 <sup>(1)</sup>	LATC6 <sup>(1)</sup>	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	xxxx xxxx	uuuu uuuu
01Bh	—	Unimplemented								—	—
01Ch	—	Unimplemented								—	—
01Dh	—	Unimplemented								—	—
01Eh	—	Unimplemented								—	—
01Fh	—	Unimplemented								—	—

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

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

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR	
<b>Bank 1</b>												
CPU CORE REGISTERS; see <a href="#">Table 4-3</a> for specifics												
08Ch — 09Ah	—	Unimplemented								—	—	
09Bh	ADRESL	ADC Result Register Low								xxxx xxxx	uuuu uuuu	
09Ch	ADRESH	ADC Result Register High								xxxx xxxx	uuuu uuuu	
09Dh	ADCON0	CHS<5:0>						GO/DONE	ADON	0000 0000	0000 0000	
09Eh	ADCON1	ADFM	ADCS<2:0>			—	—	ADPREF<1:0>		0000 --00	0000 --00	
09Fh	ADACT	—	—	—	ADACT<4:0>						---0 0000	---0 0000

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

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR
<b>Bank 2</b>											
CPU CORE REGISTERS; see Table 4-3 for specifics											
10Ch — 118h	—	Unimplemented								—	—
119h	RC1REG	EUSART Receive Data Register								0000 0000	0000 0000
11Ah	TX1REG	EUSART Transmit Data Register								0000 0000	0000 0000
11Bh	SP1BRGL	SP1BRG<7:0>								0000 0000	0000 0000
11Ch	SP1BRGH	SP1BRG<15:8>								0000 0000	0000 0000
11Dh	RC1STA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 0000	0000 0000
11Eh	TX1STA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	0000 0010
11Fh	BAUD1CON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	01-0 0-00	01-0 0-00

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

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR
<b>Bank 3</b>											
CPU CORE REGISTERS; see Table 4-3 for specifics											
18Ch	SSP1BUF	Synchronous Serial Port Receive Buffer/Transmit Register								xxxx xxxx	xxxx xxxx
18Dh	SSP1ADD	ADD<7:0>								0000 0000	0000 0000
18Eh	SSP1MSK	MSK<7:0>								1111 1111	1111 1111
18Fh	SSP1STAT	SMP	CKE	D/ $\bar{A}$	P	S	R/ $\bar{W}$	UA	BF	0000 0000	0000 0000
190h	SSP1CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
191h	SSP1CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000	0000 0000
192h	SSP1CON3	ACKTIM	PCIE	SCIE	BOEN	SDAHT	SBCDE	AHEN	DHEN	0000 0000	0000 0000
193h	—	Unimplemented								—	—
194h	—	Unimplemented								—	—
195h	—	Unimplemented								—	—
196h	SSP2BUF	Synchronous Serial Port Receive Buffer/Transmit Register								xxxx xxxx	xxxx xxxx
197h	SSP2ADD	ADD<7:0>								0000 0000	0000 0000
198h	SSP2MSK	MSK<7:0>								1111 1111	1111 1111
199h	SSP2STAT	SMP	CKE	D/ $\bar{A}$	P	S	R/ $\bar{W}$	UA	BF	0000 0000	0000 0000
19Ah	SSP2CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
19Bh	SSP2CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000	0000 0000
19Ch	SSP2CON3	ACKTIM	PCIE	SCIE	BOEN	SDAHT	SBCDE	AHEN	DHEN	0000 0000	0000 0000
19Dh	—	Unimplemented								—	—
19Eh	—	Unimplemented								—	—
19Fh	—	Unimplemented								—	—

**Legend:** x = unknown, u = unchanged,  $\bar{q}$  = depends on condition, - = unimplemented, read as '0', r = reserved. Shaded locations unimplemented, read as '0'.

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR
<b>Bank 4</b>											
CPU CORE REGISTERS; see <a href="#">Table 4-3</a> for specifics											
20Ch	TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								0000 0000	uuuu uuuu
20Dh	TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								0000 0000	uuuu uuuu
20Eh	T1CON	—	—	CKPS<1:0>		—	SYNC	RD16	ON	--00 -000	--uu -u0u
20Fh	T1GCON	GE	GPOL	GTM	GSPM	GGO/DONE	GVAL	—	—	0000 0x--	uuuu ux--
210h	T1GATE	—	—	—	GSS<4:0>					---0 0000	---u uuuu
211h	T1CLK	—	—	—	—	CS<3:0>				---- 0000	---- uuuu
212h	—	Unimplemented								—	—
21Fh	—	Unimplemented								—	—

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

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR
<b>Bank 5</b>											
CPU CORE REGISTERS; see <a href="#">Table 4-3</a> for specifics											
28Ch	T2TMR	Holding Register for the 8-bit TMR2 Register								0000 0000	0000 0000
28Dh	T2PR	TMR2 Period Register								1111 1111	1111 1111
28Eh	T2CON	ON	CKPS<2:0>			OUTPS<3:0>			0000 0000	0000 0000	
28Fh	T2HLT	PSYNC	CKPOL	CKSYNC	MODE<4:0>			0000 0000	0000 0000		
290h	T2CLKCON	—	—	—	—	CS<3:0>			---- 0000	---- 0000	
291h	T2RST	—	—	—	—	RSEL<3:0>			---- 0000	---- 0000	
292h — 29Fh	—	Unimplemented								—	—

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



**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR	
<b>Bank 6</b>												
CPU CORE REGISTERS; see Table 4-3 for specifics												
30Ch	CCPR1L	Capture/Compare/PWM Register 1 (LSB)									xxxx xxxx	uuuu uuuu
30Dh	CCPR1H	Capture/Compare/PWM Register 1 (MSB)									xxxx xxxx	uuuu uuuu
30Eh	CCP1CON	EN	—	OUT	FMT	MODE<3:0>				0-00 0000	0-00 0000	
30Fh	CCP1CAP	—	—	—	—	CTS<2:0>				---- -000	---- -000	
310h	CCPR2L	Capture/Compare/PWM Register 2 (LSB)									xxxx xxxx	uuuu uuuu
311h	CCPR2H	Capture/Compare/PWM Register 2 (MSB)									xxxx xxxx	uuuu uuuu
312h	CCP2CON	EN	—	OUT	FMT	MODE<3:0>				0-00 0000	0-00 0000	
313h	CCP2CAP	—	—	—	—	CTS<2:0>				---- -000	---- -000	
314h	PWM3DCL	DC<1:0>		—	—	—	—	—	—	xx-- ----	uu-- ----	
315h	PWM3DCH	DC<9:0>									xxxx xxxx	uuuu uuuu
316h	PWM3CON	EN	—	OUT	POL	—	—	—	—	0-00 ----	0-00 ----	
317h	—	Unimplemented									—	—
318h	PWM4DCL	DC<1:0>		—	—	—	—	—	—	xx-- ----	uu-- ----	
319h	PWM4DCH	DC<9:0>									xxxx xxxx	uuuu uuuu
31Ah	PWM4CON	EN	—	OUT	POL	—	—	—	—	0-00 ----	0-00 ----	
31Bh	—	Unimplemented									—	—
31Ch	PWM5DCL	DC<1:0>		—	—	—	—	—	—	xx-- ----	uu-- ----	
31Dh	PWM5DCH	DC<9:0>									xxxx xxxx	uuuu uuuu
31Eh	PWM5CON	EN	—	OUT	POL	—	—	—	—	0-00 ----	0-00 ----	
31Fh	—	Unimplemented									—	—

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

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR	
<b>Bank 7</b>												
CPU CORE REGISTERS; see <a href="#">Table 4-3</a> for specifics												
38Ch	PWM6DCL	DC<1:0>		—	—	—	—	—	—	xx-- ----	uu-- ----	
38Dh	PWM6DCH	DC<9:0>									xxxx xxxx	uuuu uuuu
38Eh	PWM6CON	EN	—	OUT	POL	—	—	—	—	0-00 ----	0-00 ----	
38Fh — 39Fh	—	Unimplemented								—	—	

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

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR
<b>Bank 8-10</b>											
CPU CORE REGISTERS; see <a href="#">Table 4-3</a> for specifics											
x0Ch/ x8Ch — x1Fh/ x9Fh	—	Unimplemented									

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

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR
<b>Bank 11</b>											
CPU CORE REGISTERS; see Table 4-3 for specifics											
58Ch	NCO1ACCL	NCO1ACC<7:0>								0000 0000	0000 0000
58Dh	NCO1ACCH	NCO1ACC<15:8>								0000 0000	0000 0000
58Eh	NCO1ACCU	—	—	—	—	NCO1ACC<19:16>				---- 0000	---- 0000
58Fh	NCO1INCL	NCO1INC<7:0>								0000 0001	0000 0001
590h	NCO1INCH	NCO1INC<15:8>								0000 0000	0000 0000
591h	NCO1INCUI	—	—	—	—	NCO1INC<19:16>				---- 0000	---- 0000
592h	NCO1CON	N1EN	—	N1OUT	N1POL	—	—	—	N1PFM	0-00 ---0	0-00 ---0
593h	NCO1CLK	N1PWS<2:0>			—	—	N1CKS<2:0>			000- -000	000- -000
594h	—	Unimplemented								—	—
595h	—	Unimplemented								—	—
596h	—	Unimplemented								—	—
597h	—	Unimplemented								—	—
598h	—	Unimplemented								—	—
599h	—	Unimplemented								—	—
59Ah	—	Unimplemented								—	—
59Bh	—	Unimplemented								—	—
59Ch	TMR0L	Holding Register for the Least Significant Byte of the 16-bit TMR0 Register								0000 0000	0000 0000
59Dh	TMR0H	Holding Register for the Most Significant Byte of the 16-bit TMR0 Register								1111 1111	1111 1111
59Eh	T0CON0	T0EN	—	T0OUT	T016BIT	T0OUTPS<3:0>			0-00 0000	0-00 0000	
59Fh	T0CON1	T0CS<2:0>			T0ASYNC	T0CKPS<3:0>			0000 0000	0000 0000	

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

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR		
<b>Bank 12</b>													
CPU CORE REGISTERS; see <a href="#">Table 4-3</a> for specifics													
60Ch	CWG1CLKCON	—	—	—	—	—	—	—	CS	---- --0	---- --0		
60Dh	CWG1DAT	—	—	—	—	DAT<3:0>				---- 0000	---- 0000		
60Eh	CWG1DBR	—	—	DBR<5:0>						--00 0000	--00 0000		
60Fh	CWG1DBF	—	—	DBF<5:0>						--00 0000	--00 0000		
610h	CWG1CON0	EN	LD	—	—	—	MODE<2:0>				00-- -000	00-- -000	
611h	CWG1CON1	—	—	IN	—	POLD	POLC	POLB	POLA	--x- 0000	--u- 0000		
612h	CWG1AS0	SHUTDOWN	REN	LSBD<2:0>			LSAC<2:0>			—	—	0001 01--	0001 01--
613h	CWG1AS1	—	—	—	AS4E	AS3E	AS2E	AS1E	AS0E	---0 0000	---u 0000		
614h	CWG1STR	OVRD	OVRC	OVRB	OVRA	STRD	STRC	STRB	STRA	0000 0000	0000 0000		
615h 61Fh	—	Unimplemented								—	—		

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

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR
<b>Bank 13</b>											
CPU CORE REGISTERS; see <a href="#">Table 4-3</a> for specifics											
68Ch — 69Fh	—	Unimplemented								—	—

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

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR
<b>Bank 14</b>											
CPU CORE REGISTERS; see <a href="#">Table 4-3</a> for specifics											
70Ch	PIR0	—	—	TMR0IF	IOCIF	—	—	—	INTF	--00 ---0	--00 ---0
70Dh	PIR1	OSFIF	CSWIF	—	—	—	—	—	ADIF	00-- --00	00-- --00
70Eh	PIR2	—	ZCDIF	—	—	—	—	C2IF	C1IF	-0-- --00	-0-- --00
70Fh	PIR3	RC2IF	TX2IF	RC1IF	TX1IF	BCL2IF	SSP2IF	BCL1IF	SSP1IF	0000 0000	0000 0000
710h	PIR4	—	—	—	—	—	—	TMR2IF	TMR1IF	---- --00	---- --00
711h	PIR5	CLC4IF	CLC3IF	CLC2IF	CLC1IF	—	—	—	TMR1GIF	0000 ---0	0000 ---0
712h	PIR6	—	—	—	—	—	—	CCP2IF	CCP1IF	---- --00	---- --00
713h	PIR7	—	—	NVMIF	NCO1IF	—	—	—	CWG1IF	--00 ---0	--00 ---0
714h	—	Unimplemented								—	—
715h	—	Unimplemented								—	—
716h	PIE0	—	—	TMR0IE	IOCIE	—	—	—	INTE	--00 ---0	--00 ---0
717h	PIE1	OSFIE	CSWIE	—	—	—	—	—	ADIE	00-- --00	00-- --00
718h	PIE2	—	ZCDIE	—	—	—	—	C2IE	C1IE	-0-- --00	-0-- --00
719h	PIE3	RC2IE	TX2IE	RC1IE	TX1IE	BCL2IE	SSP2IE	BCL1IE	SSP1IE	0000 0000	0000 0000
71Ah	PIE4	—	—	—	—	—	—	TMR2IE	TMR1IE	---- --00	---- --00
71Bh	PIE5	CLC4IE	CLC3IE	CLC2IE	CLC1IE	—	—	—	TMR1GIE	0000 ---0	0000 ---0
71Ch	PIE6	—	—	—	—	—	—	CCP2IE	CCP1IE	---- --00	---- --00
71Dh	PIE7	—	—	NVMIE	NCO1IE	—	—	—	CWG1IE	--00 ---0	--00 ---0
71Eh	—	Unimplemented								—	—
71Fh	—	Unimplemented								—	—

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

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR
<b>Bank 15</b>											
CPU CORE REGISTERS; see <a href="#">Table 4-3</a> for specifics											
78Ch	—	Unimplemented								—	—
795h	—	Unimplemented								—	—
796h	PMD0	SYSCMD	FVRMD	—	—	—	NVMMD	CLKRMD	IOCMD	00-- -000	00-- -000
797h	PMD1	NCO1MD	—	—	—	—	TMR2MD	TMR1MD	TMR0MD	0--- -000	0--- -000
798h	PMD2	—	DAC1MD	ADCMD	—	—	CMP2MD	CMP1MD	ZCDMD	-00- -000	-00- -000
799h	PMD3	—	—	PWM6MD	PWM5MD	PWM4MD	PWM3MD	CCP2MD	CCP1MD	--00 0000	--00 0000
79Ah	PMD4	UART2MD	UART1MD	MSSP2MD	MSSP1MD	—	—	—	CWG1MD	0000 ---0	0000 ---0
79Bh	PMD5	—	—	—	CLC4MD	CLC3MD	CLC2MD	CLC1MD	—	---0 000-	---0 000-
79Ch	—	Unimplemented								—	—
79Dh	—	Unimplemented								—	—
79Eh	—	Unimplemented								—	—
79Fh	—	Unimplemented								—	—

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



**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR	
<b>Bank 16</b>												
CPU CORE REGISTERS; see Table 4-3 for specifics												
80Ch	WDTCON0	—	—	WDTPS<4:0>				SWDTEN		--qq qq0	--qq qq0	
80Dh	WDTCON1	—	WDTC<2:0>			—	WINDOW<2:0>			-qq qq	-qq qq	
80Eh	WDTPSL	PSCNT<7:0>								0000 0000	0000 0000	
80Fh	WDTPSH	PSCNT<15:8>								0000 0000	0000 0000	
810h	WDTTMR	—	WDTTMR<3:0>				STATE	PSCNT17	PSCNT16	xxxx x000	xxxx x000	
811h	BORCON	SBOREN	—	—	—	—	—	—	BORRDY	1--- ---q	u--- ---u	
812h	VREGCON	—	—	—	—	—	—	VREGPM <sup>(1)</sup>	—	---- --0-	---- --0-	
813h	PCON0	STKOVF	STKUNF	WDTWV	RWDT	RMCLR	RI	POR	BOR	0011 110q	qqqq qquu	
814h	PCON1	—	—	—	—	—	—	MEMV	—	---- --1-	---- --u-	
815h	—	Unimplemented								—	—	
816h	—	Unimplemented								—	—	
817h	—	Unimplemented								—	—	
818h	—	Unimplemented								—	—	
819h	—	Unimplemented								—	—	
81Ah	NVMADRL	NVMADR<7:0>								xxxx xxxx	uuuu uuuu	
81Bh	NVMADRH	—	NVMADR<14:8>								-xxx xxxx	-uuu uuuu
81Ch	NVMDATL	NVMDAT<7:0>								0000 0000	0000 0000	
81Dh	NVMDATH	—	—	NVMDAT<13:8>						--00 0000	--00 0000	
81Eh	NVMCON1	—	NVMREGS	LWLO	FREE	WRERR	WREN	WR	RD	-000 x000	-000 q000	
81Fh	NVMCON2	NVMCON2<7:0>								xxxx xxxx	uuuu uuuu	

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

**Note 1:** Present only on PIC16F15354/55.

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR	
<b>Bank 17</b>												
CPU CORE REGISTERS; see Table 4-3 for specifics												
88Ch	CPUDOZE	IDLEN	DOZEN	ROI	DOE	—	DOZE<2:0>			0000 -000	u000 -000	
88Dh	OSCCON1	—	NOSC<2:0>			NDIV<3:0>			-qbb 0000	-qbb 0000		
88Eh	OSCCON2	—	COSC<2:0>			CDIV<3:0>			-qbb qbb	-qbb qbb		
88Fh	OSCCON3	CSWHOLD	SOSCPWR	—	ORDY	NOSCR	—	—	—	00-0 0---	00-0 0---	
890h	OSCSTAT	EXTOR	HFOR	MFOR	LFOR	SOR	ADOR	—	PLL	q00 qd-0	qbb qbb-q	
891h	OSCEEN	EXTOEN	HFOEN	MFOEN	LFOEN	SOSCEEN	ADOEN	—	—	0000 00--	0000 00--	
892h	OSCTUNE	—	—	HFTUN<5:0>			HFFRQ<2:0>			--10 0000	--10 0000	
893h	OSCFRQ	—	—	—	—	—	HFFRQ<2:0>			---- -bbb	---- -bbb	
894h	—	Unimplemented									—	—
895h	CLKRCON	CLKREN	—	—	CLKRDC<1:0>		CLKRDIV<2:0>			0--x xxxxx	0--u uuuu	
896h	CLKRCLK	—	—	—	—	CLKRCLK<3:0>			---- 0000	---- 0000		
897h	—	Unimplemented									—	—
89Fh	—	Unimplemented									—	—

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

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR	
<b>Bank 18</b>												
CPU CORE REGISTERS; see <a href="#">Table 4-3</a> for specifics												
90Ch	FVRCON	FVREN	FVRRDY	TSEN	TSRNG	CDAFVR<1:0>		ADFVR<1:0>		0x00 xxxxx	0q00 uuuu	
90Dh	—	Unimplemented									—	—
90Eh	DAC1CON0	EN	—	OE1	OE2	PSS<1:0>		—	NSS	0-00 00-0	0-00 00-0	
90Fh	DAC1CON1	—	—	—	DAC1R<4:0>						---0 0000	---0 0000
910h 91Eh	—	Unimplemented									—	—
91Fh	ZDCON	ZCDSEN	—	ZCDOUT	ZCDPOL	—	—	ZCDINTP	ZCDINTN	0-x0 --00	0-x0 --00	

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

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR
<b>Bank 19</b>											
CPU CORE REGISTERS; see <a href="#">Table 4-3</a> for specifics											
98Ch	—	Unimplemented								—	—
98Dh	—	Unimplemented								—	—
98Eh	—	Unimplemented								—	—
98Fh	CMOUT	—	—	—	—	—	—	MC2OUT	MC1OUT	---- --00	---- --00
990h	CM1CON0	EN	OUT	—	POL	—	—	HYS	SYNC	00-0 --00	00-0 --00
991h	CM1CON1	—	—	—	—	—	—	INTP	INTN	---- --00	---- --00
992h	CM1NCH	—	—	—	—	—	NCH<2:0>		----	-000	---- -000
993h	CM1PCH	—	—	—	—	—	PCH<2:0>		----	-000	---- -000
994h	CM2CON0	EN	OUT	—	POL	—	—	HYS	SYNC	00-0 --00	00-0 --00
995h	CM2CON1	—	—	—	—	—	—	INTP	INTN	---- --00	---- --00
996h	CM2NCH	—	—	—	—	—	NCH<2:0>		----	-000	---- -000
997h	CM2PCH	—	—	—	—	—	PCH<2:0>		----	-000	---- -000
998h 99Fh	—	Unimplemented								—	—

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

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR
<b>Bank 20</b>											
CPU CORE REGISTERS; see <a href="#">Table 4-3</a> for specifics											
A0Ch — A18h	—	Unimplemented								—	—
A19h	RC2REG	RC2REG<7:0>								0000 0000	0000 0000
A1Ah	TX2REG	TX2REG<7:0>								0000 0000	0000 0000
A1Bh	SP2BRGL	SP2BRGL<7:0>								0000 0000	0000 0000
A1Ch	SP2BRGH	SP2BRGH<7:0>								0000 0000	0000 0000
A1Dh	RC2STA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 0000	0000 0000
A1Eh	TX2STA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	0000 0010
A1Fh	BAUD2CON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	01-0 0-00	01-0 0-00

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

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR
<b>Bank 21-59</b>											
CPU CORE REGISTERS; see <a href="#">Table 4-3</a> for specifics											
x0Ch/ x8Ch — x1Fh/ x9Fh	—	Unimplemented								—	—

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

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR
<b>Bank 60</b>											
CPU CORE REGISTERS; see Table 4-3 for specifics											
1E0Ch	—	Unimplemented								—	—
1E0Dh	—	Unimplemented								—	—
1E0Eh	—	Unimplemented								—	—
1E0Fh	CLCDATA	—	—	—	—	MLC4OUT	MLC3OUT	MLC2OUT	MLC1OUT	---- xxxx	---- uuuu
1E10h	CLCCON	LC1EN	—	LC1OUT	LC1INTP	LC1INTN	LC1MODE<2:0>			0-00 0000	0-00 0000
1E11h	CLC1POL	LC1POL	—	—	—	LC1G4POL	LC1G3POL	LC1G2POL	LC1G1POL	0--- xxxx	0--- uuuu
1E12h	CLC1SEL0	—	—	LC1D1S<5:0>						--xx xxxx	--uu uuuu
1E13h	CLC1SEL1	—	—	LC1D2S<5:0>						--xx xxxx	--uu uuuu
1E14h	CLC1SEL2	—	—	LC1D3S<5:0>						--xx xxxx	--uu uuuu
1E15h	CLC1SEL3	—	—	LC1D4S<5:0>						--xx xxxx	--uu uuuu
1E16h	CLC1GLS0	LC1G1D4T	LC1G4D3N	LC1G1D3T	LC1G1D3N	LC1G1D2T	LC1G1D2N	LC1G1D1T	LC1G1D1N	xxxx xxxx	uuuu uuuu
1E17h	CLC1GLS1	LC1G2D4T	LC1G4D3N	LC1G2D3T	LC1G2D3N	LC1G2D2T	LC1G2D2N	LC1G2D1T	LC1G2D1N	xxxx xxxx	uuuu uuuu
1E18h	CLC1GLS2	LC1G3D4T	LC1G4D3N	LC1G3D3T	LC1G3D3N	LC1G3D2T	LC1G3D2N	LC1G3D1T	LC1G3D1N	xxxx xxxx	uuuu uuuu
1E19h	CLC1GLS3	LC1G4D4T	LC1G4D3N	LC1G4D3T	LC1G4D3N	LC1G4D2T	LC1G4D2N	LC1G4D1T	LC1G4D1N	xxxx xxxx	uuuu uuuu
1E1Ah	CLC2CON	LC2EN	—	LC2OUT	LC2INTP	LC2INTN	LC2MODE<2:0>			0-00 0000	0-00 0000
1E1Bh	CLC2POL	LC2POL	—	—	—	LC2G4POL	LC2G3POL	LC2G2POL	LC2G1POL	0--- xxxx	0--- uuuu
1E1Ch	CLC2SEL0	—	—	LC2D1S<5:0>						--xx xxxx	--uu uuuu
1E1Dh	CLC2SEL1	—	—	LC2D2S<5:0>						--xx xxxx	--uu uuuu
1E1Eh	CLC2SEL2	—	—	LC2D3S<5:0>						--xx xxxx	--uu uuuu
1E1Fh	CLC2SEL3	—	—	LC2D4S<5:0>						--xx xxxx	--uu uuuu
1E20h	CLC2GLS0	LC2G1D4T	LC2G4D3N	LC2G1D3T	LC2G1D3N	LC2G1D2T	LC2G1D2N	LC2G1D1T	LC2G1D1N	xxxx xxxx	uuuu uuuu
1E21h	CLC2GLS1	LC2G2D4T	LC2G4D3N	LC2G2D3T	LC2G2D3N	LC2G2D2T	LC2G2D2N	LC2G2D1T	LC2G2D1N	xxxx xxxx	uuuu uuuu
1E22h	CLC2GLS2	LC2G3D4T	LC2G4D3N	LC2G3D3T	LC2G3D3N	LC2G3D2T	LC2G3D2N	LC2G3D1T	LC2G3D1N	xxxx xxxx	uuuu uuuu
1E23h	CLC2GLS3	LC2G4D4T	LC2G4D3N	LC2G4D3T	LC2G4D3N	LC2G4D2T	LC2G4D2N	LC2G4D1T	LC2G4D1N	xxxx xxxx	uuuu uuuu
1E24h	CLC3CON	LC3EN	—	LC3OUT	LC3INTP	LC3INTN	LC3MODE			0-00 0000	0-00 0000
1E25h	CLC3POL	LC3POL	—	—	—	LC3G4POL	LC3G3POL	LC3G2POL	LC3G1POL	0--- xxxx	0--- uuuu
1E26h	CLC3SEL0	—	—	LC3D1S<5:0>						--xx xxxx	--uu uuuu
1E27h	CLC3SEL1	—	—	LC3D2S<5:0>						--xx xxxx	--uu uuuu
1E28h	CLC3SEL2	—	—	LC3D3S<5:0>						--xx xxxx	--uu uuuu
1E29h	CLC3SEL3	—	—	LC3D4S<5:0>						--xx xxxx	--uu uuuu
1E2Ah	CLC3GLS0	LC3G1D4T	LC3G4D3N	LC3G1D3T	LC3G1D3N	LC3G1D2T	LC3G1D2N	LC3G1D1T	LC3G1D1N	xxxx xxxx	uuuu uuuu

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

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR
<b>Bank 60 (Continued)</b>											
1E2Bh	CLC3GLS1	LC3G2D4T	LC3G4D3N	LC3G2D3T	LC3G2D3N	LC3G2D2T	LC3G2D2N	LC3G2D1T	LC3G2D1N	xxxx xxxx	uuuu uuuu
1E2Ch	CLC3GLS2	LC3G3D4T	LC3G4D3N	LC3G3D3T	LC3G3D3N	LC3G3D2T	LC3G3D2N	LC3G3D1T	LC3G3D1N	xxxx xxxx	uuuu uuuu
1E2Dh	CLC3GLS3	LC3G4D4T	LC3G4D3N	LC3G4D3T	LC3G4D3N	LC3G4D2T	LC3G4D2N	LC3G4D1T	LC3G4D1N	xxxx xxxx	uuuu uuuu
1E2Eh	CLC4CON	LC4EN	—	LC4OUT	LC4INTP	LC4INTN	LC4MODE<2:0>			0-00 0000	0-00 0000
1E2Fh	CLC4POL	LC4POL	—	—	—	LC4G4POL	LC4G3POL	LC4G2POL	LC4G1POL	0--- xxxx	0--- uuuu
1E30h	CLC4SEL0	—	—	LC4D1S<5:0>						--xx xxxx	--uu uuuu
1E31h	CLC4SEL1	—	—	LC4D2S<5:0>						--xx xxxx	--uu uuuu
1E32h	CLC4SEL2	—	—	LC4D3S<5:0>						--xx xxxx	--uu uuuu
1E33h	CLC4SEL3	—	—	LC4D4S<5:0>						--xx xxxx	--uu uuuu
1E34h	CLC4GLS0	LC4G1D4T	LC4G4D3N	LC4G1D3T	LC4G1D3N	LC4G1D2T	LC4G1D2N	LC4G1D1T	LC4G1D1N	xxxx xxxx	uuuu uuuu
1E35h	CLC4GLS1	LC4G2D4T	LC4G4D3N	LC4G2D3T	LC4G2D3N	LC4G2D2T	LC4G2D2N	LC4G2D1T	LC4G2D1N	xxxx xxxx	uuuu uuuu
1E36h	CLC4GLS2	LC4G3D4T	LC4G4D3N	LC4G3D3T	LC4G3D3N	LC4G3D2T	LC4G3D2N	LC4G3D1T	LC4G3D1N	xxxx xxxx	uuuu uuuu
1E37h	CLC4GLS3	LC4G4D4T	LC4G4D3N	LC4G4D3T	LC4G4D3N	LC4G4D2T	LC4G4D2N	LC4G4D1T	LC4G4D1N	xxxx xxxx	uuuu uuuu
1E38h — 1E6Fh	—	Unimplemented								—	—

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



**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR
<b>Bank 61</b>											
CPU CORE REGISTERS; see Table 4-3 for specifics											
1E8Ch	—	Unimplemented								—	—
1E8Dh	—	Unimplemented								—	—
1E8Eh	—	Unimplemented								—	—
1E8Fh	PPSLOCK	—	—	—	—	—	—	—	PPSLOCKED	---- --0	---- --0
1E90h	INTPPS	—	—	INTPPS<5:0>						--00 1000	--uu uuuu
1E91h	T0CKIPPS	—	—	T0CKIPPS<5:0>						--00 0100	--uu uuuu
1E92h	T1CKIPPS	—	—	T1CKIPPS<5:0>						--01 0000	--uu uuuu
1E93h	T1GPPS	—	—	T1GPPS<5:0>						--00 1101	--uu uuuu
1E94h	—	Unimplemented								—	—
1E9Bh	—	Unimplemented								—	—
1E9Ch	T2INPPS	—	—	T2INPPS<5:0>						--01 0011	--uu uuuu
1E9Dh	—	Unimplemented								—	—
1EA0h	—	Unimplemented								—	—
1EA1h	CCP1PPS	—	—	CCP1PPS<5:0>						--01 0010	--uu uuuu
1EA2h	CCP2PPS	—	—	CCP2PPS<5:0>						--01 0001	--uu uuuu
1EA3h	—	Unimplemented								—	—
1EB0h	—	Unimplemented								—	—
1EB1h	CWG1PPS	—	—	CWG1PPS<5:0>						--00 1000	--uu uuuu
1EB2h	—	Unimplemented								—	—
1EBAh	—	Unimplemented								—	—
1EBBh	CLCIN0PPS	—	—	CLCIN0PPS<5:0>						--00 0000	--uu uuuu
1EBCh	CLCIN1PPS	—	—	CLCIN1PPS<5:0>						--00 0001	--uu uuuu
1EBDh	CLCIN2PPS	—	—	CLCIN2PPS<5:0>						--00 1110	--uu uuuu
1EBEh	CLCIN3PPS	—	—	CLCIN3PPS<5:0>						--00 1111	--uu uuuu
1EBFh	—	Unimplemented								—	—
1EC2h	—	Unimplemented								—	—
1EC3h	ADACTPPS	—	—	ADACTPPS<5:0>						--001100	--uuuuuu
1EC4h	—	Unimplemented								—	—

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

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR
<b>Bank 61 (Continued)</b>											
1EC5h	SSP1CLKPPS	—	—				SSP1CLKPPS<5:0>			--01 0011	--uu uuuu
1EC6h	SSP1DATPPS	—	—				SSP1DATPPS<5:0>			--01 0100	--uu uuuu
1EC7h	SSP1SSPPS	—	—				SSP1SSPPS<5:0>			--00 0101	--uu uuuu
1EC8h	SSP2CLKPPS	—	—				SSP2CLKPPS<5:0>			--00 1001	--uu uuuu
1EC9h	SSP2DATPPS	—	—				SSP2DATPPS<5:0>			--00 1000	--uu uuuu
1ECAh	SSP2SSPPS	—	—				SSP2SSPPS<5:0>			--00 1000	--uu uuuu
1ECBh	RX1DTPPS	—	—				RX1DTPPS<5:0>			--01 0111	--uu uuuu
1ECCh	TX1CKPPS	—	—				TX1CKPPS<5:0>			--01 0110	--uu uuuu
1ECDh	RX2DTPPS	—	—				RX2DTPPS<5:0>			--00 1111	--uu uuuu
1ECEh	TX2CKPPS	—	—				TX2CKPPS<5:0>			--00 1110	--uu uuuu
1ECFh — 1EEFh	—	Unimplemented								—	—

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

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR	
<b>Bank 62</b>												
CPU CORE REGISTERS; see Table 4-3 for specifics												
1F0Ch	—	Unimplemented								—	—	
1F0Dh	—	Unimplemented								—	—	
1F0Eh	—	Unimplemented								—	—	
1F0Fh	—	Unimplemented								—	—	
1F10h	RA0PPS	—	—	—	RA0PPS<4:0>				---	0000	---u	uuuu
1F11h	RA1PPS	—	—	—	RA1PPS<4:0>				---	0000	---u	uuuu
1F12h	RA2PPS	—	—	—	RA2PPS<4:0>				---	0000	---u	uuuu
1F13h	RA3PPS	—	—	—	RA3PPS<4:0>				---	0000	---u	uuuu
1F14h	RA4PPS	—	—	—	RA4PPS<4:0>				---	0000	---u	uuuu
1F15h	RA5PPS	—	—	—	RA5PPS<4:0>				---	0000	---u	uuuu
1F16h	RA6PPS	—	—	—	RA6PPS<4:0>				---	0000	---u	uuuu
1F17h	RA7PPS	—	—	—	RA7PPS<4:0>				---	0000	---u	uuuu
1F18h	RB0PPS	—	—	—	RB0PPS<4:0>				---	0000	---u	uuuu
1F19h	RB1PPS	—	—	—	RB1PPS<4:0>				---	0000	---u	uuuu
1F1Ah	RB2PPS	—	—	—	RB2PPS<4:0>				---	0000	---u	uuuu
1F1Bh	RB3PPS	—	—	—	RB3PPS<4:0>				---	0000	---u	uuuu
1F1Ch	RB4PPS	—	—	—	RB4PPS<4:0>				---	0000	---u	uuuu
1F1Dh	RB5PPS	—	—	—	RB5PPS<4:0>				---	0000	---u	uuuu
1F1Eh	RB6PPS	—	—	—	RB6PPS<4:0>				---	0000	---u	uuuu
1F1Fh	RB7PPS	—	—	—	RB7PPS<4:0>				---	0000	---u	uuuu
1F20h	RC0PPS	—	—	—	RC0PPS<4:0>				---	0000	---u	uuuu
1F21h	RC1PPS	—	—	—	RC1PPS<4:0>				---	0000	---u	uuuu
1F22h	RC2PPS	—	—	—	RC2PPS<4:0>				---	0000	---u	uuuu
1F23h	RC3PPS	—	—	—	RC3PPS<4:0>				---	0000	---u	uuuu
1F24h	RC4PPS	—	—	—	RC4PPS<4:0>				---	0000	---u	uuuu
1F25h	RC5PPS	—	—	—	RC5PPS<4:0>				---	0000	---u	uuuu
1F26h	RC6PPS	—	—	—	RC6PPS<4:0>				---	0000	---u	uuuu
1F27h	RC7PPS	—	—	—	RC7PPS<4:0>				---	0000	---u	uuuu
1F28h — 1F37h	—	Unimplemented								—	—	

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

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR
<b>Bank 62 (Continued)</b>											
1F38h	ANSELA	ANSA7	ANSA6	ANSA5	ANSA4	ANSA3	ANSA2	ANSA1	ANSA0	1111 1111	1111 1111
1F39h	WPUA	WPUA7	WPUA6	WPUA5	WPUA4	WPUA3	WPUA2	WPUA1	WPUA0	0000 0000	0000 0000
1F3Ah	ODCONA	ODCA7	ODCA6	ODCA5	ODCA4	ODCA3	ODCA2	ODCA1	ODCA0	0000 0000	0000 0000
1F3Bh	SLRCONA	SLRA7	SLRA6	SLRA5	SLRA4	SLRA3	SLRA2	SLRA1	SLRA0	1111 1111	1111 1111
1F3Ch	INLVLA	INLVLA7	INLVLA6	INLVLA5	INLVLA4	INLVLA3	INLVLA2	INLVLA1	INLVLA0	1111 1111	1111 1111
1F3Dh	IOCAP	IOCAP7	IOCAP6	IOCAP5	IOCAP4	IOCAP3	IOCAP2	IOCAP1	IOCAP0	0000 0000	0000 0000
1F3Eh	IOCAN	IOCAN7	IOCAN6	IOCAN5	IOCAN4	IOCAN3	IOCAN2	IOCAN1	IOCAN0	0000 0000	0000 0000
1F3Fh	IOCAF	IOCAF7	IOCAF6	IOCAF5	IOCAF4	IOCAF3	IOCAF2	IOCAF1	IOCAF0	0000 0000	0000 0000
1F40h — 1F42h	—	Unimplemented								—	—
1F43h	ANSELB	ANSB7	ANSB6	ANSB5	ANSB4	ANSB3	ANSB2	ANSB1	ANSB0	1111 1111	1111 1111
1F44h	WPUB	WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0	0000 0000	0000 0000
1F45h	ODCONB	ODCB7	ODCB6	ODCB5	ODCB4	ODCB3	ODCB2	ODCB1	ODCB0	0000 0000	0000 0000
1F46h	SLRCONB	SLRB7	SLRB6	SLRB5	SLRB4	SLRB3	SLRB2	SLRB1	SLRB0	1111 1111	1111 1111
1F47h	INVLVB	INVLVB7	INVLVB6	INVLVB5	INVLVB4	INVLVB3	INVLVB2	INVLVB1	INVLVB0	1111 1111	1111 1111
1F48h	IOCBP	IOCBP7	IOCBP6	IOCBP5	IOCBP4	IOCBP3	IOCBP2	IOCBP1	IOCBP0	0000 0000	0000 0000
1F49h	IOCBN	IOCBN7	IOCBN6	IOCBN5	IOCBN4	IOCBN3	IOCBN2	IOCBN1	IOCBN0	0000 0000	0000 0000
1F4Ah	IOCBF	IOCBF7	IOCBF6	IOCBF5	IOCBF4	IOCBF3	IOCBF2	IOCBF1	IOCBF0	0000 0000	0000 0000
1F4Bh — 1F4Dh	—	Unimplemented								—	—
1F4Eh	ANSELC	ANSC7	ANSC6	ANSC5	ANSC4	ANSC3	ANSC2	ANSC1	ANSC0	1111 1111	1111 1111
1F4Fh	WPUC	WPUC7	WPUC6	WPUC5	WPUC4	WPUC3	WPUC2	WPUC1	WPUC0	0000 0000	0000 0000
1F50h	ODCONC	ODCC7	ODCC6	ODCC5	ODCC4	ODCC3	ODCC2	ODCC1	ODCC0	0000 0000	0000 0000
1F51h	SLRCONC	SLRC7	SLRC6	SLRC5	SLRC4	SLRC3	SLRC2	SLRC1	SLRC0	1111 1111	1111 1111
1F52h	INLVLC	INLVLC7	INLVLC6	INLVLC5	INLVLC4	INLVLC3	INLVLC2	INLVLC1	INLVLC0	1111 1111	1111 1111
1F53h	IOCCP	IOCCP7	IOCCP6	IOCCP5	IOCCP4	IOCCP3	IOCCP2	IOCCP1	IOCCP0	0000 0000	0000 0000
1F54h	IOCCN	IOCCN7	IOCCN6	IOCCN5	IOCCN4	IOCCN3	IOCCN2	IOCCN1	IOCCN0	0000 0000	0000 0000
1F55h	IOCCF	IOCCF7	IOCCF6	IOCCF5	IOCCF4	IOCCF3	IOCCF2	IOCCF1	IOCCF0	0000 0000	0000 0000
1F56h — 1F64h	—	Unimplemented								—	—
1F65h	WPUE	—	—	—	—	WPUE3	—	—	—	---- 0---	---- u---
1F66h	—	Unimplemented								—	—
1F67h	—	Unimplemented								—	—

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

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR
<b>Bank 62 (Continued)</b>											
1F68h	INLVLE	—	—	—	—	INLVLE3	—	—	—	---- 1---	----u ---
1F69h	IOCEP	—	—	—	—	IOCEP3	—	—	—	---- 0---	---- 0---
1F6Ah	IOCEN	—	—	—	—	IOCEN3	—	—	—	---- 0---	---- 0---
1F6Bh	IOCEF	—	—	—	—	IOCEF3	—	—	—	---- 0---	---- 0---
1F6Ch — 1F6Fh	—	Unimplemented								—	—

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

**TABLE 4-10: SPECIAL FUNCTION REGISTER SUMMARY BANKS 0-63 (CONTINUED)**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on: MCLR	
<b>Bank 63</b>												
CPU CORE REGISTERS; see <a href="#">Table 4-3</a> for specifics												
1F8Ch — 1FE3h	—	Unimplemented								—	—	
1FE4h	STATUS_SHAD	—	—	—	—	—	Z	DC	C	---- -xxx	---- -uuu	
1FE5h	WREG_SHAD	Working Register Shadow								xxxx xxxx	uuuu uuuu	
1FE6h	BSR_SHAD	—	—	—	Bank Select Register Shadow					--x xxxx	--u uuuu	
1FE7h	PCLATH_SHAD	—	Program Counter Latch High Register Shadow								-xxx xxxx	uuuu uuuu
1FE8h	FSR0L_SHAD	Indirect Data Memory Address 0 Low Pointer Shadow								xxxx xxxx	uuuu uuuu	
1FE9h	FSR0H_SHAD	Indirect Data Memory Address 0 High Pointer Shadow								xxxx xxxx	uuuu uuuu	
1FEAh	FSR1L_SHAD	Indirect Data Memory Address 1 Low Pointer Shadow								xxxx xxxx	uuuu uuuu	
1FEBh	FSR1H_SHAD	Indirect Data Memory Address 1 High Pointer Shadow								xxxx xxxx	uuuu uuuu	
1FEC	—	Unimplemented								—	—	
1FEDh	STKPTR	—	—	—	Current Stack Pointer					---1 1111	---1 1111	
1FEEh	TOSL	Top of Stack Low byte								xxxx xxxx	uuuu uuuu	
1FEFh	TOSH	—	Top of Stack High byte								-xxx xxxx	-uuu uuuu

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

## 4.4 PCL and PCLATH

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

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



### 4.4.1 MODIFYING PCL

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

### 4.4.2 COMPUTED GOTO

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

### 4.4.3 COMPUTED FUNCTION CALLS

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

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

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

### 4.4.4 BRANCHING

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

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

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

## 4.5 Stack

All devices have a 16-level x 15-bit wide hardware stack (refer to [Figure 4-5](#) through [Figure 4-8](#)). The stack space is not part of either program or data space. The PC is PUSHed onto the stack when `CALL` or `CALLW` instructions are executed or an interrupt causes a branch. The stack is POPed in the event of a `RETURN`, `RETLW` or a `RETFIE` instruction execution. `PCLATH` is not affected by a `PUSH` or `POP` operation.

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

**Note 1:** There are no instructions/mnemonics called `PUSH` or `POP`. These are actions that occur from the execution of the `CALL`, `CALLW`, `RETURN`, `RETLW` and `RETFIE` instructions or the vectoring to an interrupt address.

### 4.5.1 ACCESSING THE STACK

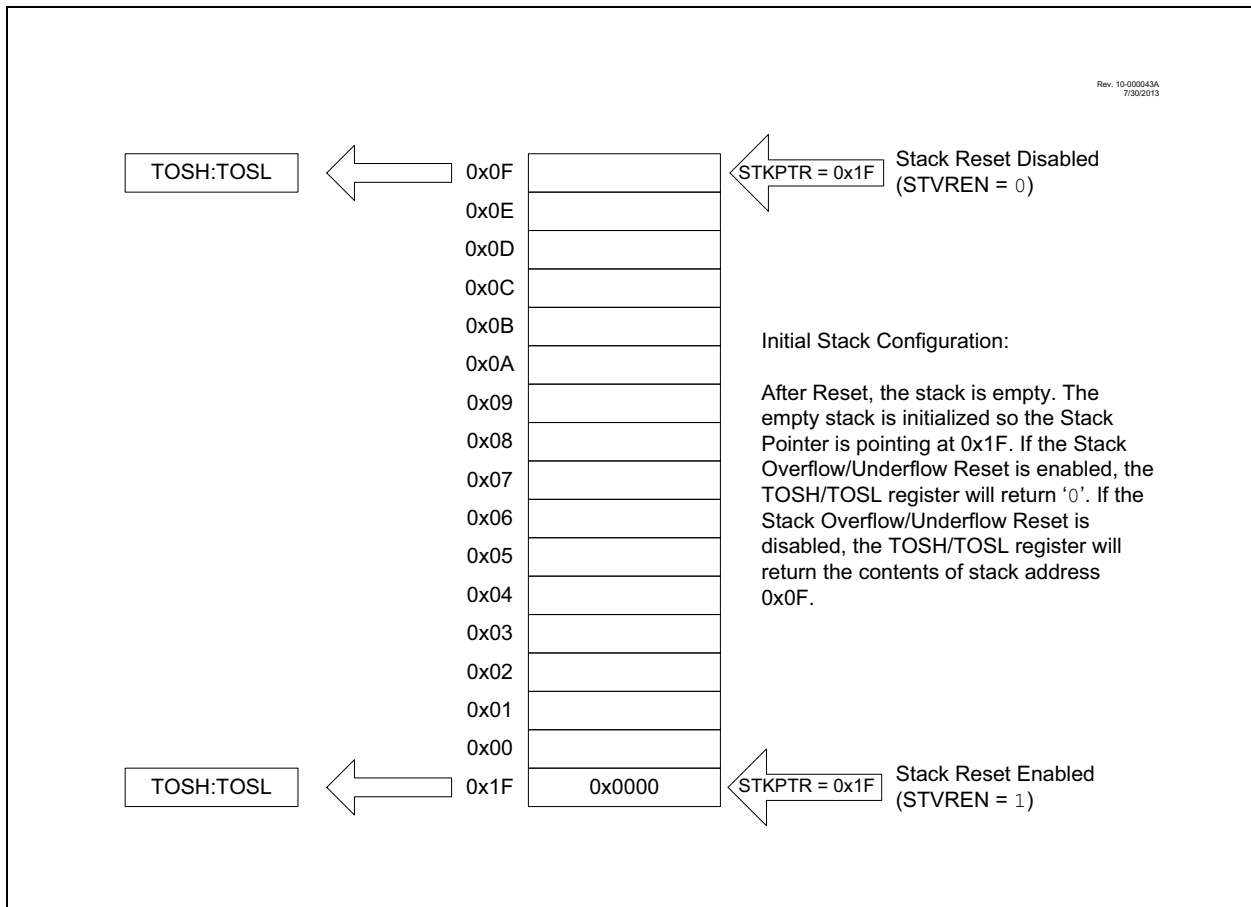
The stack is accessible through the `TOSH`, `TOSL` and `STKPTR` registers. `STKPTR` is the current value of the Stack Pointer. `TOSH:TOSL` register pair points to the TOP of the stack. Both registers are read/writable. `TOS` is split into `TOSH` and `TOSL` due to the 15-bit size of the PC. To access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

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

During normal program operation, `CALL`, `CALLW` and interrupts will increment `STKPTR` while `RETLW`, `RETURN`, and `RETFIE` will decrement `STKPTR`. `STKPTR` can be monitored to obtain to value of stack memory left at any given time. 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 value from the stack and then decrement the `STKPTR`.

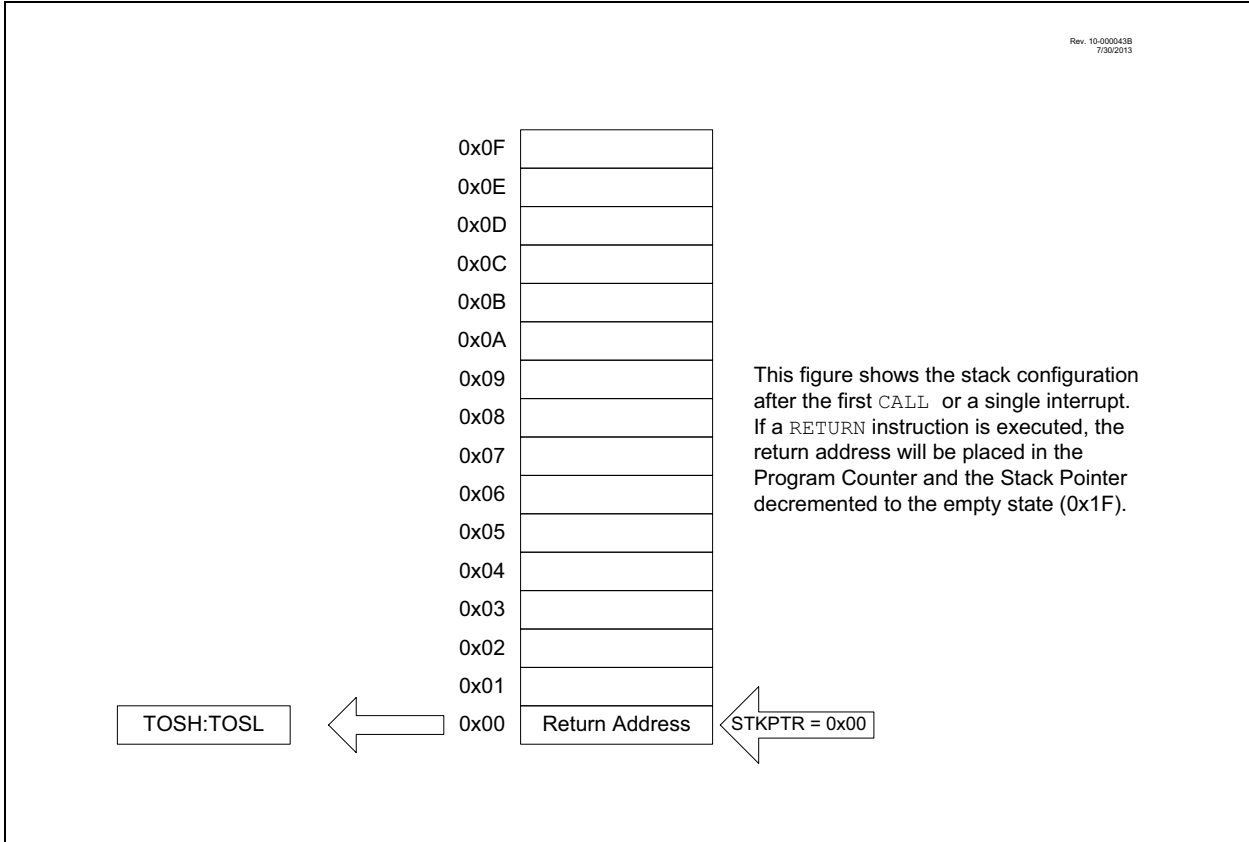
Reference [Figure 4-5](#) through [Figure 4-8](#) for examples of accessing the stack.

**FIGURE 4-5: ACCESSING THE STACK EXAMPLE 1**

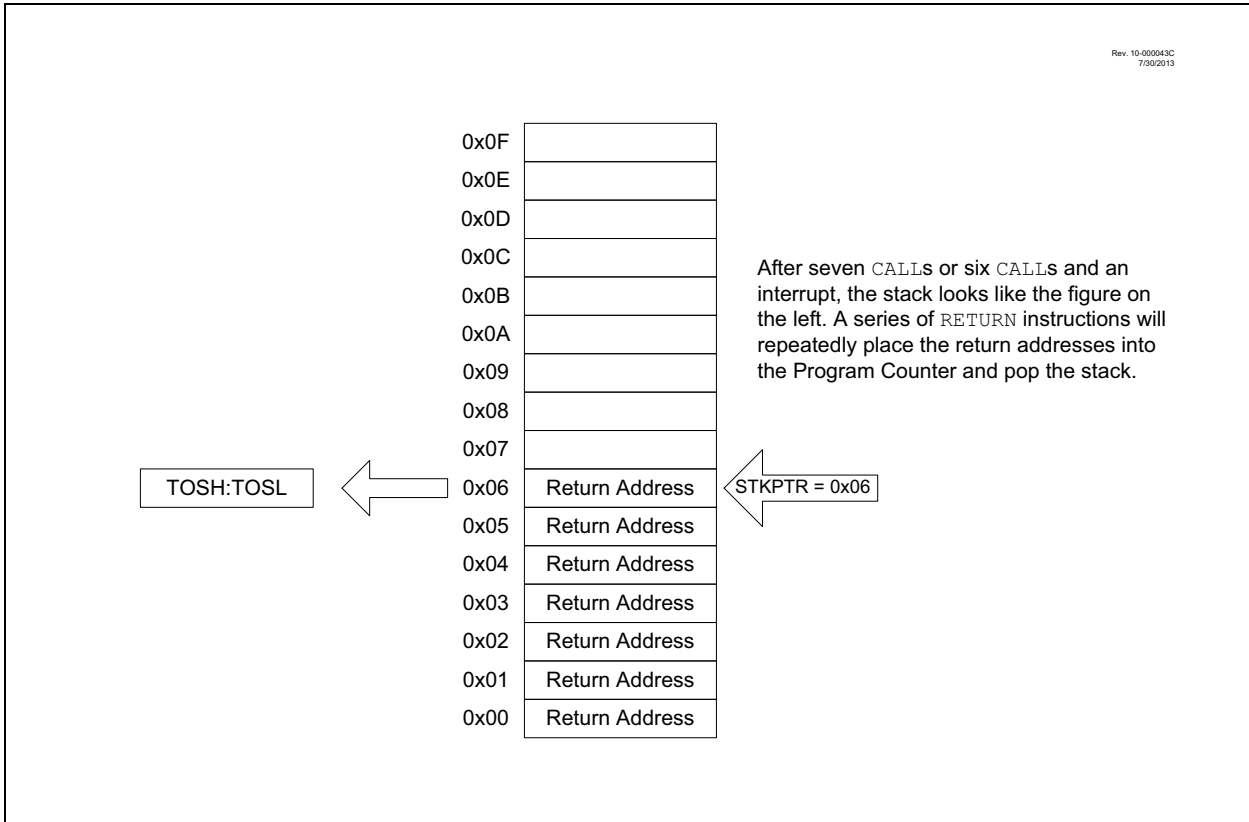




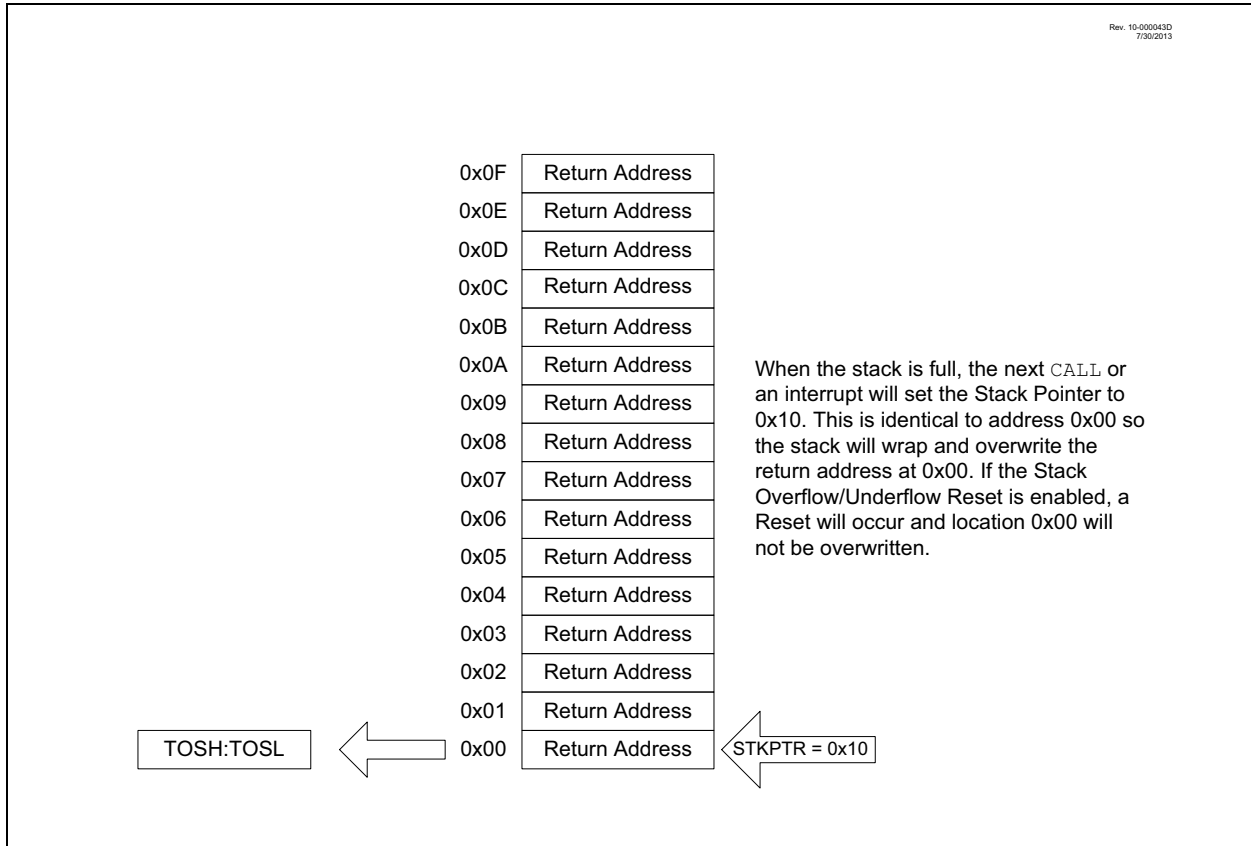
**FIGURE 4-6: ACCESSING THE STACK EXAMPLE 2**



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



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



## 4.5.2 OVERFLOW/UNDERFLOW RESET

If the `STVREN` bit in Configuration Words ([Register 5-2](#)) is programmed to '1', the device will be Reset if the stack is PUSHed beyond the sixteenth level or POPed beyond the first level, setting the appropriate bits (`STKOVF` or `STKUNF`, respectively) in the `PCON` register.

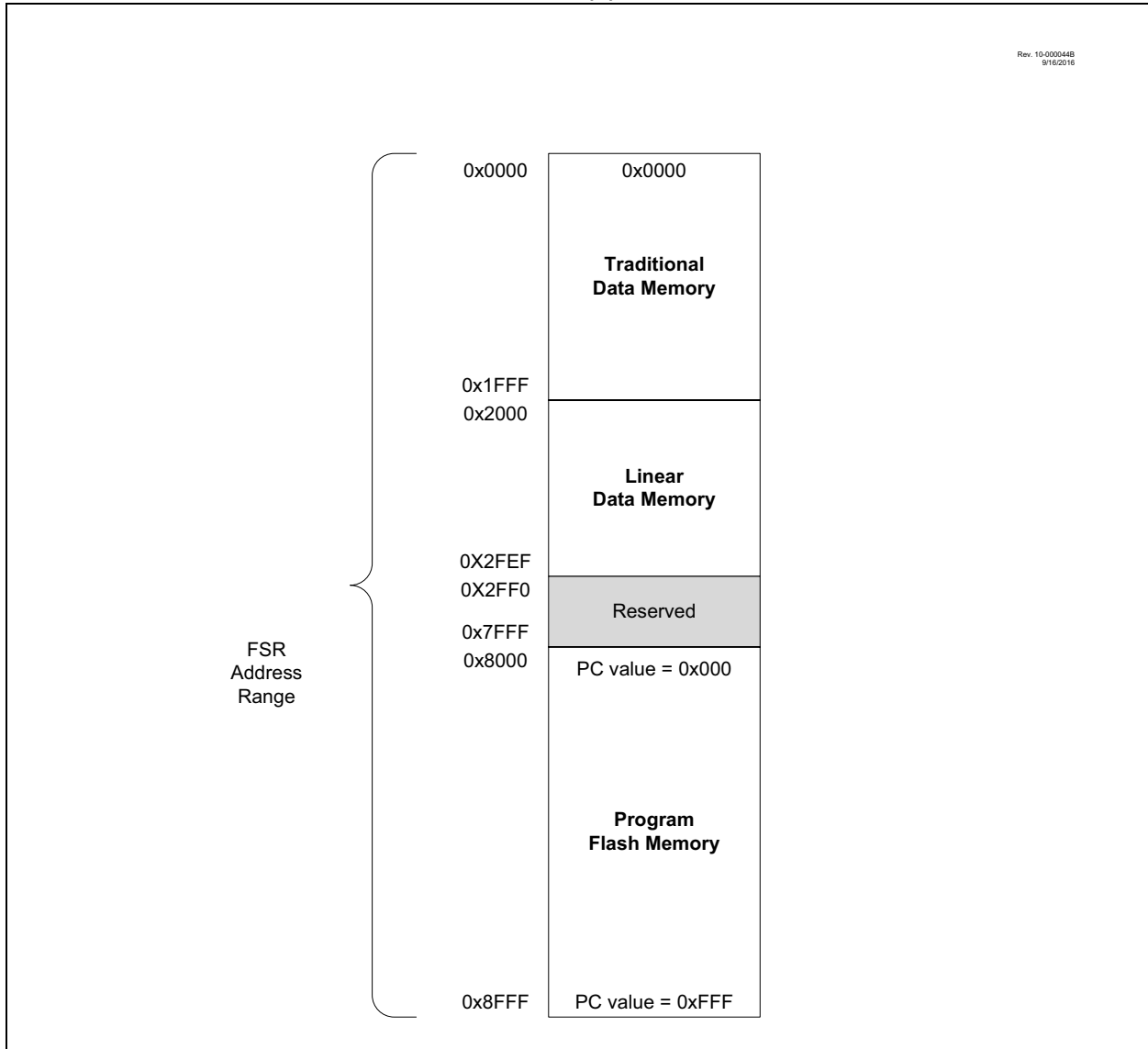
## 4.6 Indirect Addressing

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

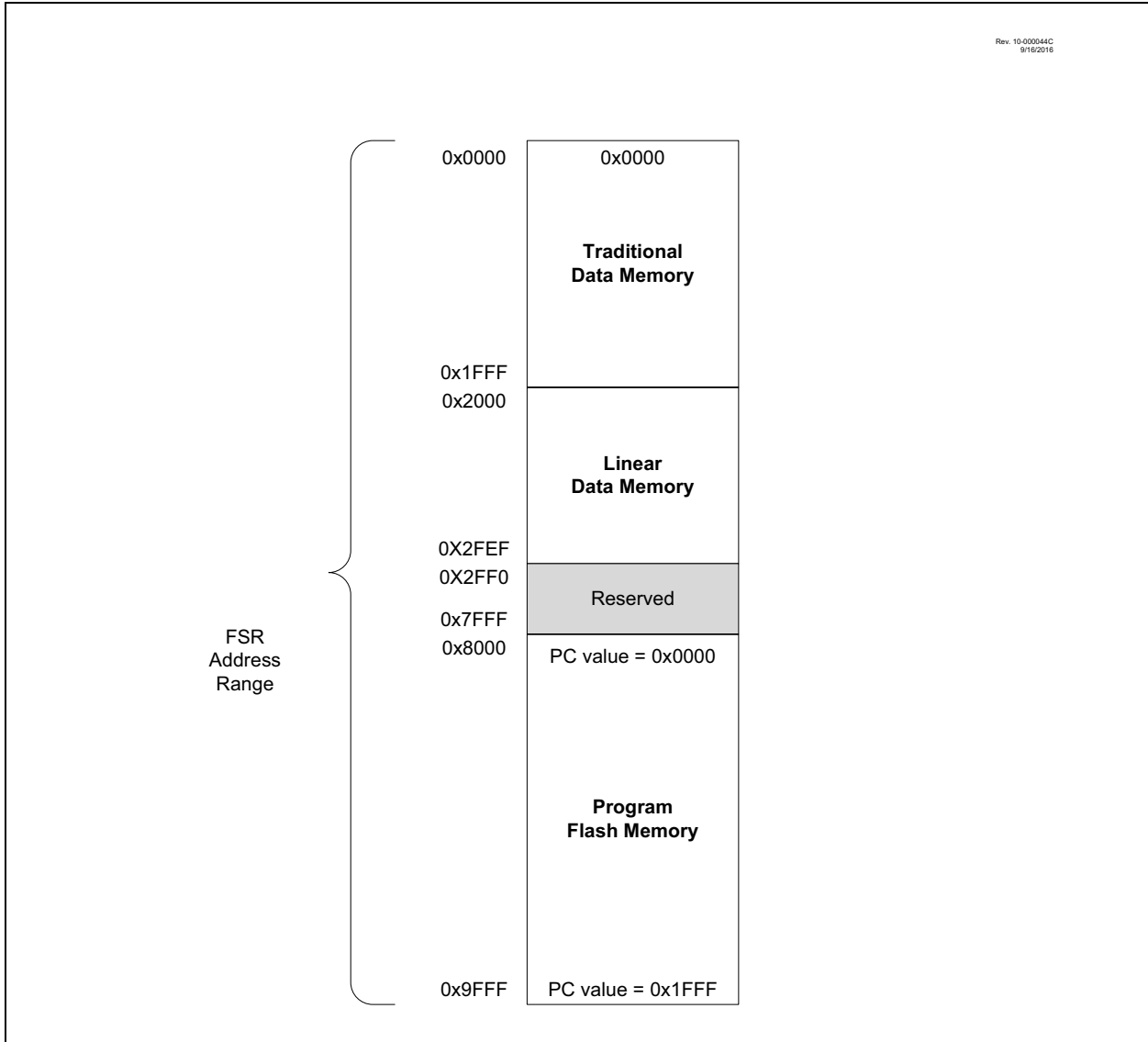
The `FSR` registers form a 16-bit address that allows an addressing space with 65536 locations. These locations are divided into three memory regions:

- Traditional/Banked Data Memory
- Linear Data Memory
- Program Flash Memory

**FIGURE 4-9: INDIRECT ADDRESSING PIC16(L)F15354**



**FIGURE 4-10: INDIRECT ADDRESSING PIC16(L)F15355**



## 4.6.1 TRADITIONAL/BANKED DATA MEMORY

The traditional or banked data memory is a region from FSR address 0x000 to FSR address 0x1FFF. The addresses correspond to the absolute addresses of all SFR, GPR and common registers.

**FIGURE 4-11: TRADITIONAL/BANKED DATA MEMORY MAP**



## 4.6.2 LINEAR DATA MEMORY

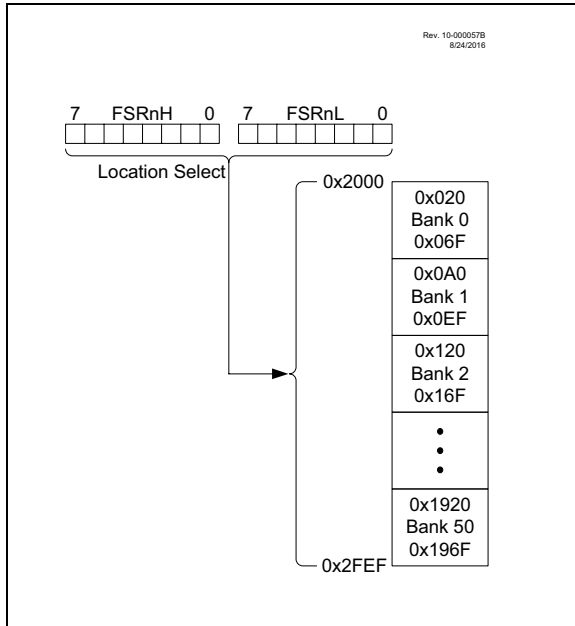
The linear data memory is the region from FSR address 0x2000 to FSR address 0x2FEF. This region is a virtual region that points back to the 80-byte blocks of GPR memory in all the banks. Refer to [Figure 4-12](#) for the Linear Data Memory Map.

**Note:** The address range 0x2000 to 0x2FF0 represents the complete addressable Linear Data Memory up to Bank 50. The actual implemented Linear Data Memory will differ from one device to the other in a family. Confirm the memory limits on every device.

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.

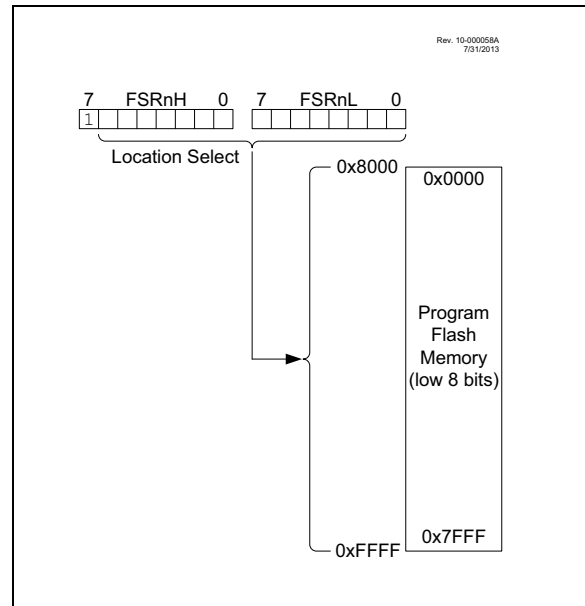
**FIGURE 4-12: LINEAR DATA MEMORY MAP**



## 4.6.3 PROGRAM FLASH MEMORY

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

**FIGURE 4-13: PROGRAM FLASH MEMORY MAP**



## 5.0 DEVICE CONFIGURATION

Device configuration consists of the Configuration Words, User ID, Device ID, Device Information Area (DIA), (see [Section 6.0 “Device Information Area”](#)), and the Device Configuration Information (DCI) regions, (see [Section 7.0 “Device Configuration Information”](#)).

### 5.1 Configuration Words

The devices have several Configuration Words starting at address 8007h. The Configuration bits establish configuration values prior to the execution of any software; Configuration bits enable or disable device-specific features.

In terms of programming, these important Configuration bits should be considered:

#### 1. LVP: Low-Voltage Programming Enable bit

- 1 = ON – Low-Voltage Programming is enabled. MCLR/VPP pin function is MCLR. MCLRE Configuration bit is ignored.
- 0 = OFF – HV on MCLR/VPP must be used for programming.

#### 2. CP: User Nonvolatile Memory (NVM) Program Memory Code Protection bit

- 1 = OFF – User NVM code protection disabled
- 0 = ON – User NVM code protection enabled

## 5.2 Register Definitions: Configuration Words

### REGISTER 5-1: CONFIGURATION WORD 1: OSCILLATORS

R/P-1	U-1	R/P-1	U-1	U-1	R/P-1
FCMEN	—	CSWEN	—	—	CLKOUTEN
bit 13					bit 8

U-1	R/P-1	R/P-1	R/P-1	U-1	R/P-1	R/P-1	R/P-1
—	RSTOSC2	RSTOSC1	RSTOSC0	—	FEXTOSC2	FEXTOSC1	FEXTOSC0
bit 7							bit 0

#### Legend:

R = Readable bit	P = Programmable bit	x = Bit is unknown	U = Unimplemented bit, read as '1'
'0' = Bit is cleared	'1' = Bit is set	W = Writable bit	n = Value when blank or after Bulk Erase

- bit 13 **FCMEN:** Fail-Safe Clock Monitor Enable bit  
 1 = FSCM timer enabled  
 0 = FSCM timer disabled
- bit 12 **Unimplemented:** Read as '1'
- bit 11 **CSWEN:** Clock Switch Enable bit  
 1 = Writing to NOSC and NDIV is allowed  
 0 = The NOSC and NDIV bits cannot be changed by user software
- bit 10-9 **Unimplemented:** Read as '1'
- bit 8 **CLKOUTEN:** Clock Out Enable bit  
**If FEXTOSC = EC (high, mid or low) or Not Enabled:**  
 1 = CLKOUT function is disabled; I/O or oscillator function on OSC2  
 0 = CLKOUT function is enabled; FOSC/4 clock appears at OSC2  
**Otherwise:**  
 This bit is ignored.
- bit 7 **Unimplemented:** Read as '1'
- bit 6-4 **RSTOSC<2:0>:** Power-up Default Value for COSC bits  
 This value is the Reset-default value for COSC and selects the oscillator first used by user software.  
 111 = EXTOSC operating per FEXTOSC bits  
 110 = HFINTOSC (1 MHz) with OSCFRQ = '010' (4 MHz) and CDIV = '0010' (4:1)  
 101 = LFINTOSC  
 100 = SOSC  
 011 = Reserved  
 010 = EXTOSC with 4x PLL, with EXTOSC operating per FEXTOSC bits  
 001 = HFINTOSC with 2x PLL (32 MHz), with OSCFRQ = '101' (16 MHz) and CDIV = '0000' (1:1)  
 000 = HFINTOSC (32 MHz), with OSCFRQ = '110' (32 MHz) and CDIV = '0000' (1:1)
- bit 3 **Unimplemented:** Read as '1'
- bit 2-0 **FEXTOSC<2:0>:**FEXTOSC External Oscillator Mode Selection bits  
 111 = EC (External Clock) above 8 MHz  
 110 = EC (External Clock) for 100 kHz to 8 MHz  
 101 = EC (External Clock) below 100 kHz  
 100 = Oscillator not enabled  
 011 = Reserved (do not use)  
 010 = HS (Crystal oscillator) above 4 MHz  
 001 = XT (Crystal oscillator) above 100 kHz, below 4 MHz  
 000 = LP (Crystal oscillator) optimized for 32.768 kHz



**REGISTER 5-2:**
**CONFIGURATION WORD 2: SUPERVISORS**

R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	U-1
$\overline{\text{DEBUG}}$	STVREN	PPS1WAY	ZCDDIS	BORV	—
bit 13					bit 8

R/P-1	R/P-1	R/P-1	U-1	U-1	U-1	R/P-1	R/P-1
BOREN1	BOREN0	$\overline{\text{LPBOREN}}$	—	—	—	$\overline{\text{PWRTE}}$	MCLRE
bit 7							bit 0

**Legend:**

R = Readable bit	P = Programmable bit	x = Bit is unknown	U = Unimplemented bit, read as '1'
'0' = Bit is cleared	'1' = Bit is set	W = Writable bit	n = Value when blank or after Bulk Erase

- bit 13 **DEBUG**: Debugger Enable bit  
 1 = Background debugger disabled  
 0 = Background debugger enabled
- bit 12 **STVREN**: Stack Overflow/Underflow Reset Enable bit  
 1 = Stack Overflow or Underflow will cause a Reset  
 0 = Stack Overflow or Underflow will not cause a Reset
- bit 11 **PPS1WAY**: PPSLOCK One-Way Set Enable bit  
 1 = The PPSLOCK bit can be cleared and set only once; PPS registers remain locked after one clear/set cycle  
 0 = The PPSLOCK bit can be set and cleared repeatedly (subject to the unlock sequence)
- bit 10 **ZCDDIS**: Zero-Cross Detect Disable bit  
 1 = ZCD disabled. ZCD can be enabled by setting the ZCDSEN bit of the ZCDCON register  
 0 = ZCD always enabled (ZCDSEN bit is ignored)
- bit 9 **BORV**: Brown-out Reset Voltage Selection bit<sup>(1)</sup>  
 1 = Brown-out Reset voltage (VBOR) set to lower trip point level  
 0 = Brown-out Reset voltage (VBOR) set to higher trip point level
- bit 8 **Unimplemented**: Read as '1'
- bit 7-6 **BOREN<1:0>**: Brown-out Reset Enable bits  
 When enabled, Brown-out Reset Voltage (VBOR) is set by the BORV bit  
 11 = Brown-out Reset is enabled; SBOREN bit is ignored  
 10 = Brown-out Reset is enabled while running, disabled in Sleep; SBOREN bit is ignored  
 01 = Brown-out Reset is enabled according to SBOREN  
 00 = Brown-out Reset is disabled
- bit 5 **LPBOREN**: Low-Power BOR Enable bit  
 1 = ULPBOR is disabled  
 0 = ULPBOR is enabled
- bit 4-2 **Unimplemented**: Read as '1'
- bit 1 **PWRTE**: Power-up Timer Enable bit  
 1 = PWRT is disabled  
 0 = PWRT is enabled
- bit 0 **MCLRE**: Master Clear ( $\overline{\text{MCLR}}$ ) Enable bit  
 If LVP = 1:  
 RE3 pin function is  $\overline{\text{MCLR}}$  (it will reset the device when driven low)  
 If LVP = 0:  
 1 =  $\overline{\text{MCLR}}$  pin is  $\overline{\text{MCLR}}$  (it will reset the device when driven low)  
 0 =  $\overline{\text{MCLR}}$  pin may be used as general purpose RE3 input

- Note 1:** See Vbor parameter for specific trip point voltages.  
**Note 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'.

# PIC16(L)F15354/55

## REGISTER 5-3: CONFIGURATION WORD 3: WINDOWED WATCHDOG

R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1
WDTCCS2	WDTCCS1	WDTCCS0	WDTCWS2	WDTCWS1	WDTCWS0
bit 13					bit 8

U-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1
—	WDTE1	WDTE0	WDTCP4	WDTCP3	WDTCP2	WDTCP1	WDTCP0
bit 7							bit 0

### Legend:

R = Readable bit      P = Programmable bit      x = Bit is unknown      U = Unimplemented bit, read as '1'  
 '0' = Bit is cleared      '1' = Bit is set      W = Writable bit      n = Value when blank or after Bulk Erase

bit 13-11    **WDTCCS<2:0>**: WDT Input Clock Selector bits  
 111 = Software Control  
 110 = Reserved  
 .  
 .  
 .  
 010 = SOSC 32 kHz  
 001 = WDT reference clock is the 31.25 kHz HFINTOSC (MFINTOSC) output  
 000 = WDT reference clock is the 31.0 kHz LFINTOSC

bit 10-8    **WDTCWS<2:0>**: WDT Window Select bits

WDTCWS	WDTWS at POR			Software control of WDTWS?	Keyed access required?
	Value	Window delay Percent of time	Window opening Percent of time		
111	111	n/a	100	Yes	No
110	111	n/a	100	No	Yes
101	101	25	75		
100	100	37.5	62.5		
011	011	50	50		
010	010	62.5	37.5		
001	001	75	25		
000	000	87.5	12.5		

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

bit 6-5    **WDTE<1:0>**: WDT Operating mode:  
 11 =WDT enabled regardless of Sleep; SWDTEN is ignored  
 10 =WDT enabled while Sleep = 0, suspended when Sleep = 1; SWDTEN ignored  
 01 =WDT enabled/disabled by SWDTEN bit in WDTCON0  
 00 =WDT disabled, SWDTEN is ignored

# PIC16(L)F15354/55

## REGISTER 5-3: CONFIGURATION WORD 3: WINDOWED WATCHDOG (CONTINUED)

bit 4-0 WDTCP5<4:0>: WDT Period Select bits

WDTCP5	WDTPS at POR				Software Control of WDTPS?
	Value	Divider Ratio		Typical Time Out (F <sub>IN</sub> = 31 kHz)	
11111 <sup>(1)</sup>	01011	1:65536	2 <sup>16</sup>	2 s	Yes
11110 ... 10011	11110 ... 10011	1:32	2 <sup>5</sup>	1 ms	No
10010	10010	1:8388608	2 <sup>23</sup>	256 s	No
10001	10001	1:4194304	2 <sup>22</sup>	128 s	
10000	10000	1:2097152	2 <sup>21</sup>	64 s	
01111	01111	1:1048576	2 <sup>20</sup>	32 s	
01110	01110	1:524299	2 <sup>19</sup>	16 s	
01101	01101	1:262144	2 <sup>18</sup>	8 s	
01100	01100	1:131072	2 <sup>17</sup>	4 s	
01011	01011	1:65536	2 <sup>16</sup>	2 s	
01010	01010	1:32768	2 <sup>15</sup>	1 s	
01001	01001	1:16384	2 <sup>14</sup>	512 ms	
01000	01000	1:8192	2 <sup>13</sup>	256 ms	
00111	00111	1:4096	2 <sup>12</sup>	128 ms	
00110	00110	1:2048	2 <sup>11</sup>	64 ms	
00101	00101	1:1024	2 <sup>10</sup>	32 ms	
00100	00100	1:512	2 <sup>9</sup>	16 ms	
00011	00011	1:256	2 <sup>8</sup>	8 ms	
00010	00010	1:128	2 <sup>7</sup>	4 ms	
00001	00001	1:64	2 <sup>6</sup>	2 ms	
00000	00000	1:32	2 <sup>5</sup>	1 ms	

**Note 1:** 0b11111 is the default value of the WDTCP5 bits.

## REGISTER 5-4: CONFIGURATION WORD 4: MEMORY

R/W-1	U-1	R/W-1	U-1	R/W-1	R/W-1
LVP	—	$\overline{\text{WRTSAF}}^{(1)}$	—	$\overline{\text{WRTC}}^{(1)}$	$\overline{\text{WRTB}}^{(1)}$
bit 13	12	11	10	9	bit 8

R/W-1	U-1	U-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
$\overline{\text{WRTAPP}}^{(1)}$	—	—	$\overline{\text{SAFEN}}^{(1)}$	$\overline{\text{BBEN}}^{(1)}$	BBSIZE2	BBSIZE1	BBSIZE0
bit 7	6	5	4	3	2	1	bit 0

### Legend:

R = Readable bit	P = Programmable bit	x = Bit is unknown	U = Unimplemented bit, read as '1'
'0' = Bit is cleared	'1' = Bit is set	W = Writable bit	n = Value when blank or after Bulk Erase

- bit 13 **LVP:** Low Voltage Programming Enable bit  
 1 = Low voltage programming enabled. MCLR/VPP pin function is  $\overline{\text{MCLR}}$ . MCLR Configuration bit is ignored.  
 0 = HV on MCLR/VPP must be used for programming.  
 The LVP bit cannot be written (to zero) while operating from the LVP programming interface. The purpose of this rule is to prevent the user from dropping out of LVP mode while programming from LVP mode, or accidentally eliminating LVP mode from the configuration state.  
 The preconditioned (erased) state for this bit is critical.
- bit 12 **Unimplemented:** Read as '1'
- bit 11 **WRTSAF:** Storage Area Flash Write Protection bit  
 1 = SAF NOT write-protected  
 0 = SAF write-protected  
 Unimplemented, if SAF is not supported in the device family and only applicable if  $\overline{\text{SAFEN}} = 0$ .
- bit 10 **Unimplemented:** Read as '1'
- bit 9 **WRTC:** Configuration Register Write Protection bit  
 1 = Configuration Register NOT write-protected  
 0 = Configuration Register write-protected
- bit 8 **WRTB:** Boot Block Write Protection bit  
 1 = Boot Block NOT write-protected  
 0 = Boot Block write-protected  
 Only applicable if  $\overline{\text{BBEN}} = 0$ .
- bit 7 **WRTAPP:** Application Block Write Protection bit  
 1 = Application Block NOT write-protected  
 0 = Application Block write-protected
- bit 6-5 **Unimplemented:** Read as '1'
- bit 4 **SAFEN:** SAF Enable bit  
 1 = SAF disabled  
 0 = SAF enabled
- bit 3 **BBEN:** Boot Block Enable bit  
 1 = Boot Block disabled  
 0 = Boot Block enabled
- bit 2-0 **BBSIZE<2:0>:** Boot Block Size Selection bits (See [Table 5-1](#))  
 BBSIZE is used only when  $\overline{\text{BBEN}} = 0$   
 BBSIZ bits can only be written while  $\overline{\text{BBEN}} = 1$ ; after  $\overline{\text{BBEN}} = 0$ , BBSIZ is write-protected.

**Note 1:** Bits are implemented as sticky bits. Once protection is enabled, it can only be reset through a Bulk Erase.

**TABLE 5-1: BOOT BLOCK SIZE BITS**

$\overline{\text{BBEN}}$	BBSIZE<2:0>	Actual Boot Block Size User Program Memory Size (words)		Last Boot Block Memory Access
		PIC16(L)F15354	PIC16(L)F15355	
1	xxx	0	0	–
0	111	512	512	01FFh
0	110	1024	1024	03FFh
0	101	2048	2048	07FFh
0	100-000	2048	4096	0FFFh

**Note:** The maximum boot block size is half the user program memory size. All selections higher than the maximum are set to half size. For example, all BBSIZE = 000 - 100 produce a boot block size of 4kW on a 8kW device.

## REGISTER 5-5: CONFIGURATION WORD 5: CODE PROTECTION

U-1	U-1	U-1	U-1	U-1	U-1
—	—	—	—	—	—
bit 13					bit 8

U-1	U-1	U-1	U-1	U-1	U-1	U-1	R/P-1
—	—	—	—	—	—	—	$\overline{\text{CP}}$
bit 7							bit 0

### Legend:

R = Readable bit  
'0' = Bit is cleared

P = Programmable bit  
'1' = Bit is set

x = Bit is unknown  
W = Writable bit

U = Unimplemented bit, read as '1'  
n = Value when blank or after Bulk Erase

bit 13-1

**Unimplemented:** Read as '1'

bit 0

**$\overline{\text{CP}}$ :** Program Flash Memory Code Protection bit  
1 = Program Flash Memory code protection disabled  
0 = Program Flash Memory code protection enabled

## 5.3 Code Protection

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

### 5.3.1 PROGRAM MEMORY PROTECTION

The entire program memory space is protected from external reads and writes by the 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. Self-writing the program memory is dependent upon the write protection setting. See [Section 5.4 "Write Protection"](#) for more information.

## 5.4 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  $\overline{WRTAPP}$ ,  $\overline{WRTSAF}$ ,  $\overline{WRTB}$ ,  $\overline{WRTC}$  bits in Configuration Words ([Register 5-4](#)) define whether the corresponding region of the program memory block is protected or not.

## 5.5 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 13.3.6 "NVMREG Access to Device Information Area, Device Configuration Area, User ID, Device ID and Configuration Words"](#) for more information on accessing these memory locations. For more information on checksum calculation, see the "PIC16(L)F153xx Memory Programming Specification" (DS40001838).

## 5.6 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.

Development tools, such as device programmers and debuggers, may be used to read the Device ID, Revision ID and Configuration Words. These locations can also be read from the NVMCON register.

## 5.7 Register Definitions: Device and Revision

### REGISTER 5-6: 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
PIC16F15354	11 0000 1010 1100 (30ACh)
PIC16LF15354	11 0000 1010 1101 (30ADh)
PIC16F15355	11 0000 1010 1110 (30AEh)
PIC16LF15355	11 0000 1010 1111 (30AFh)



# PIC16(L)F15354/55

## REGISTER 5-7: REVISIONID: REVISION ID REGISTER

R	R	R	R	R	R	R	R	R	R	R	R	R	R
1	0	MJRREV<5:0>						MNRREV<5:0>					
bit 13												bit 0	

### Legend:

R = Readable bit

'0' = Bit is cleared

'1' = Bit is set

x = Bit is unknown

bit 13-12 **Fixed Value:** Read-only bits

These bits are fixed with value '10' for all devices included in this data sheet.

bit 11-6 **MJRREV<5:0>:** Major Revision ID bits

These bits are used to identify a major revision.

bit 5-0 **MNRREV<5:0>:** Minor Revision ID bits

These bits are used to identify a minor revision.

## 6.0 DEVICE INFORMATION AREA

The Device Information Area (DIA) is a dedicated region in the program memory space; it is a new feature in the PIC16(L)F15354/55 family of devices. The DIA contains the calibration data for the internal temperature indicator module, stores the Microchip Unique Identifier words and the Fixed Voltage Reference voltage readings measured in mV.

The complete DIA table is shown in [Table 6-1: Device Information Area](#), followed by a description of each region and its functionality. The data is mapped from 8100h to 811Fh in the PIC16(L)F15354/55 family. These locations are read-only and cannot be erased or modified. The data is programmed into the device during manufacturing.

**TABLE 6-1: DEVICE INFORMATION AREA**

Address Range	Name of Region	Standard Device Information
8100h-8108h	MUI0	Microchip Unique Identifier (9 Words)
	MUI1	
	MUI2	
	MUI3	
	MUI4	
	MUI5	
	MUI6	
	MUI7	
	MUI8	
8109h	MUI9	1 Word Reserved
810Ah-8111h	EUI0	Unassigned (8 Words)
	EUI1	
	EUI2	
	EUI3	
	EUI4	
	EUI5	
	EUI6	
	EUI7	
8112h	Reserved	Unassigned (1 word)
8113h	TSLR2	Temperature indicator ADC reading at 90°C (low range setting)
8114h	Reserved	Unassigned (1 word)
8115h	Reserved	Unassigned (1 word)
8116h	TSHR2	Temperature indicator ADC reading at 90°C (high range setting)
8117h	Reserved	Unassigned (1 Word)
8118h	FVRA1X	ADC FVR1 Output voltage for 1x setting (in mV)
8119h	FVRA2X	ADC FVR1 Output Voltage for 2x setting (in mV)
811Ah	FVRA4X <sup>(1)</sup>	ADC FVR1 Output Voltage for 4x setting (in mV)
811Bh	FVRC1X	Comparator FVR2 output voltage for 1x setting (in mV)
811Ch	FVRC2X	Comparator FVR2 output voltage for 2x setting (in mV)
811Dh	FVRC4X <sup>(1)</sup>	Comparator FVR2 output voltage for 4x setting (in mV)
811Eh-811Fh		Unassigned (2 Words)

**Note 1:** Value not present on LF devices.

## 6.1 Microchip Unique identifier (MUI)

The PIC16(L)F15354/55 devices are individually encoded during final manufacturing with a Microchip Unique Identifier, or MUI. The MUI cannot be erased by a Bulk Erase command or any other user-accessible means. This feature allows for manufacturing traceability of Microchip Technology devices in applications where this is a required. It may also be used by the application manufacturer for a number of functions that require unverified unique identification, such as:

- Tracking the device
- Unique serial number

The MUI consists of nine program words. When taken together, these fields form a unique identifier. The MUI is stored in nine read-only locations, located between 8100h to 8109h in the DIA space. [Table 6-1](#) lists the addresses of the identifier words.

**Note:** For applications that require verified unique identification, contact your Microchip Technology sales office to create a Serialized Quick Turn Programming option.

## 6.2 External Unique Identifier (EUI)

The EUI data is stored at locations 810Ah to 8111h in the program memory region. This region is an optional space for placing application specific information. The data is coded per customer requirements during manufacturing. The EUI cannot be erased by a Bulk Erase command.

**Note:** Data is stored in this address range on receiving a request from the customer. The customer may contact the local sales representative or Field Applications Engineer, and provide them the unique identifier information that is required to be stored in this region.

## 6.3 Analog-to-Digital Conversion Data of the Temperature Sensor

The purpose of the temperature indicator module is to provide a temperature-dependent voltage that can be measured by an analog module. [Section 19.0 “Temperature Indicator Module”](#) explains the operation of the Temperature Indicator module and defines terms such as the low range and high range settings of the sensor.

The DIA table contains the internal ADC measurement values of the temperature sensor for low and high range at fixed points of reference. The values are measured during test and are unique to each device. The right-justified ADC readings are stored in the DIA memory region. The calibration data can be used to plot the approximate sensor output voltage,  $V_{TSENSE}$  vs. Temperature curve.

- **TSLR2:** Address 8113h stores the measurements for the low range setting of the temperature sensor at  $V_{DD} = 3V$ .
- **TSHR2:** Address 8116h stores the measurements for the high range setting of the temperature sensor at  $V_{DD} = 3V$ .

The stored measurements are made by the device ADC using the internal  $V_{REF} = 2.048V$ .

## 6.4 Fixed Voltage Reference Data

The Fixed Voltage Reference, or FVR, is a stable voltage reference, independent of  $V_{DD}$ , 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

For more information on the FVR, refer to [Section 18.0 “Fixed Voltage Reference \(FVR\)”](#).

The DIA stores measured FVR voltages for this device in mV for the different buffer settings of 1x, 2x or 4x at program memory locations 8118h to 811Dh.

- FVRA1X stores the value of ADC FVR1 Output voltage for 1x setting (in mV)
- FVRA2X stores the value of ADC FVR1 Output Voltage for 2x setting (in mV)
- FVRA4X stores the value of ADC FVR1 Output Voltage for 4x setting (in mV)
- FVRC1X stores the value of Comparator FVR2 output voltage for 1x setting (in mV)
- FVRC2X stores the value of Comparator FVR2 output voltage for 2x setting (in mV)
- FVRC4X stores the value of Comparator FVR2 output voltage for 4x setting (in mV)

## 7.0 DEVICE CONFIGURATION INFORMATION

The Device Configuration Information (DCI) is a dedicated region in the Program Flash Memory mapped from 8200h to 821Fh. The data stored in the DCI memory is hard-coded into the device during manufacturing.

Refer to [Table 7-1](#) for the complete DCI table address and description. The DCI holds information about the device which is useful for programming and bootloader applications. These locations are read-only and cannot be erased or modified.

**TABLE 7-1: DEVICE CONFIGURATION INFORMATION FOR PIC16(L)F15354/55 DEVICES**

ADDRESS	Name	DESCRIPTION	VALUE		UNITS
			PIC16(L)F15354	PIC16(L)F15355	
8200h	ERSIZ	Erase Row Size	32	32	Words
8201h	WLSIZ	Number of write latches	32	32	Latches
8202h	URSIZ	Number of User Rows	128	256	Rows
8203h	EESIZ	EE Data memory size	0	0	Bytes
8204h	PCNT	Pin Count	28	28	Pins

### 7.1 DIA and DCI Access

The DIA and DCI data are read-only and cannot be erased or modified. See [13.3.6 “NVMREG Access to Device Information Area, Device Configuration Area, User ID, Device ID and Configuration Words”](#) for more information on accessing these memory locations.

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

## 8.0 RESETS

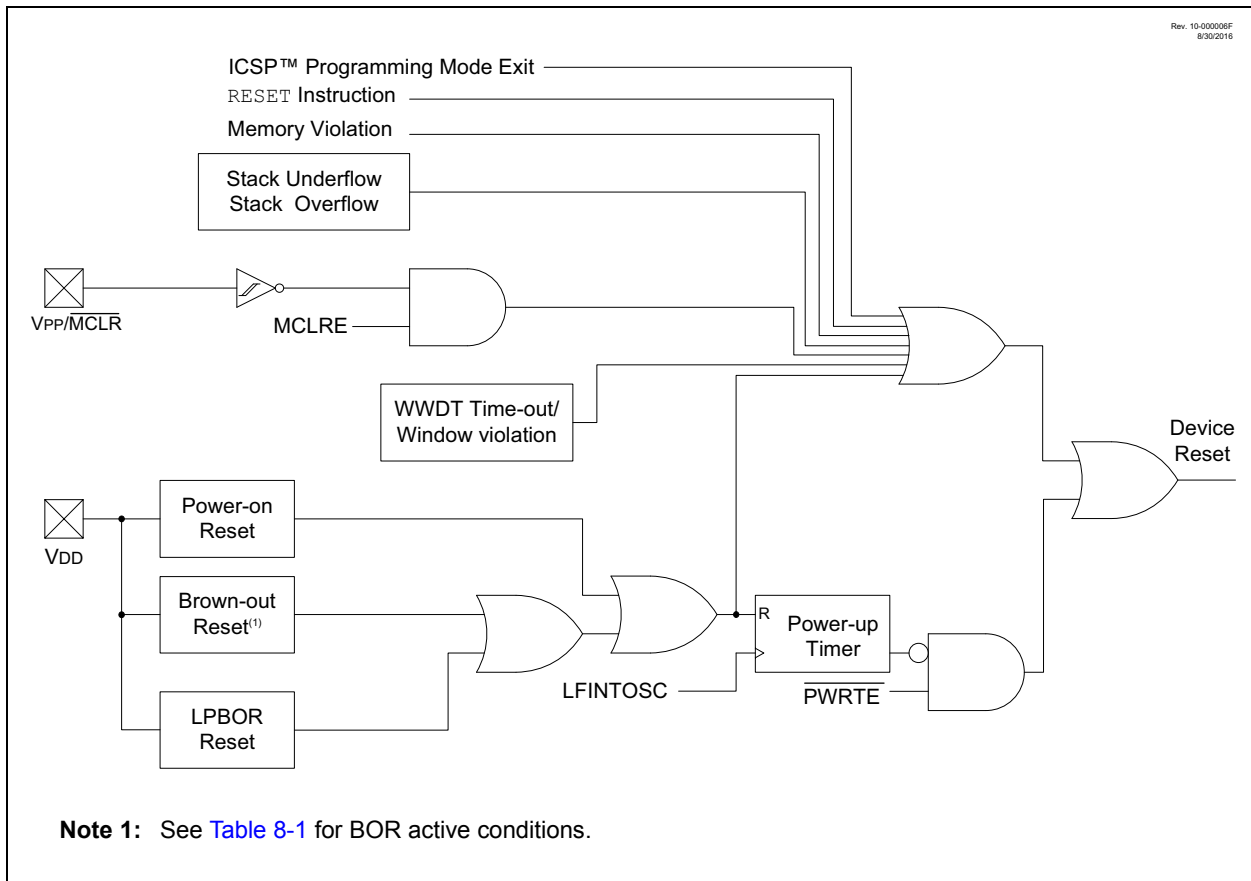
A simplified block diagram of the On-Chip Reset Circuit is shown in [Figure 8-1](#).

There are multiple ways to reset this device:

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

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

**FIGURE 8-1: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT**



## 8.1 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.

## 8.2 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 8-1](#) 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 8-2](#) for more information.

**TABLE 8-1: 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	Wait for release of BOR <sup>(1)</sup> (BORRDY = 1)
10	x	Awake	Active	Waits for release of BOR (BORRDY = 1) Waits for BOR Reset release
		Sleep	Disabled	
01	1	X	Active	Waits for BOR Reset release (BORRDY = 1) Begins immediately (BORRDY = x)
	0	X	Disabled	
00	x	X	Disabled	

**Note 1:** In this specific case, “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.

### 8.2.1 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.

### 8.2.2 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.

## 8.2.3 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.

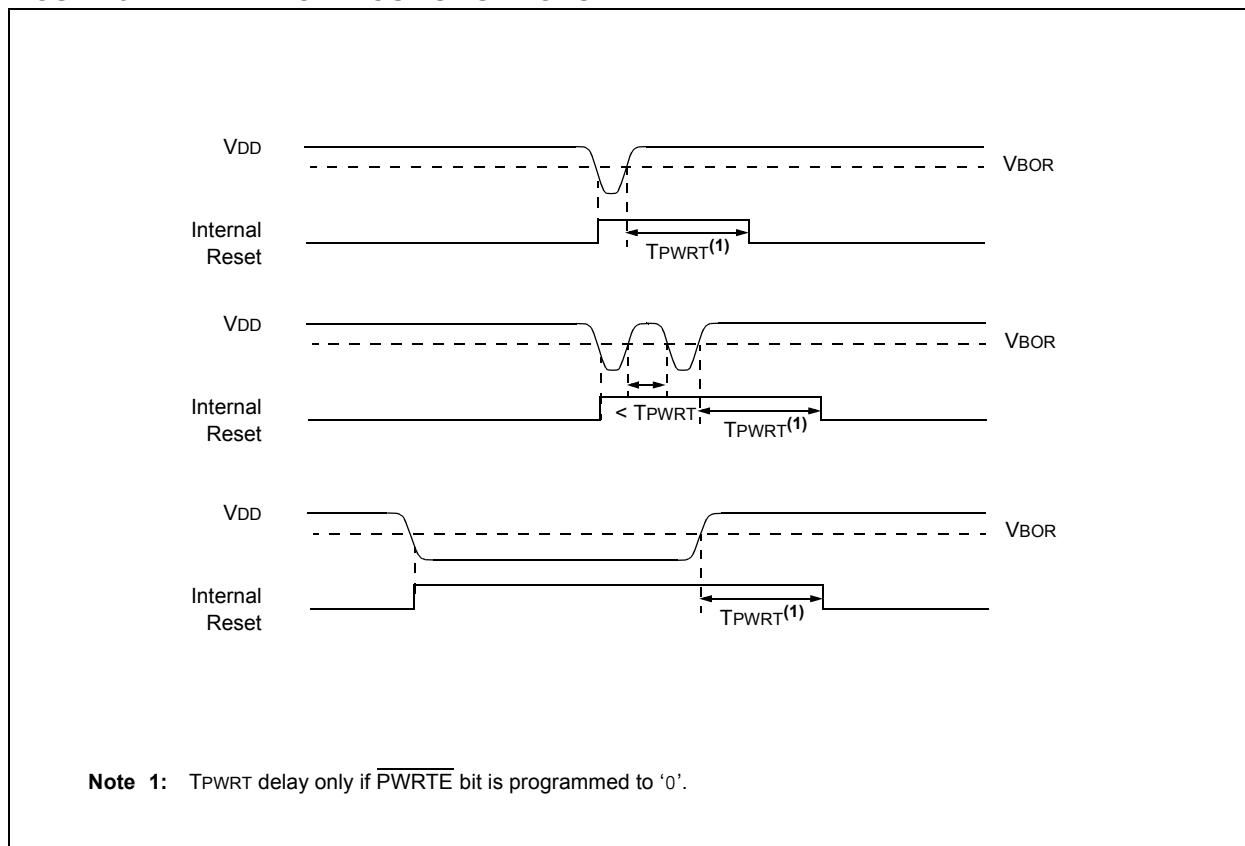
BOR protection begins as soon as the BOR circuit is ready. The status of the BOR circuit is reflected in the BORRDY bit of the BORCON register.

BOR protection is unchanged by Sleep.

## 8.2.4 BOR IS ALWAYS OFF

When the BOREN bits of the Configuration Words are programmed to '00', the BOR is off at all times. The device start-up is not delayed by the BOR ready condition or the VDD level.

**FIGURE 8-2: BROWN-OUT SITUATIONS**



## 8.3 Register Definitions: Brown-out Reset Control

### REGISTER 8-1: BORCON: BROWN-OUT RESET CONTROL REGISTER

R/W-1/u	U-0	U-0	U-0	U-0	U-0	U-0	R-q/u
SBOREN <sup>(1)</sup>	—	—	—	—	—	—	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<sup>(1)</sup>  
 If BOREN <1:0> in Configuration Words ≠ 01:  
 SBOREN is read/write, but has no effect on the BOR.  
 If BOREN <1:0> in Configuration Words = 01:  
 1 = BOR Enabled  
 0 = BOR Disabled
- bit 6-1    **Unimplemented:** Read as '0'
- bit 0      **BORRDY:** Brown-out Reset Circuit Ready Status bit  
 1 = The Brown-out Reset circuit is active  
 0 = The Brown-out Reset circuit is inactive

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



## 8.4 Low-Power Brown-out Reset (LPBOR)

The Low-Power Brown-out Reset (LPBOR) is an important part of the Reset subsystem. Refer to [Figure 8-1](#) to see how the BOR and LPBOR interact 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.

### 8.4.1 ENABLING LPBOR

The LPBOR is controlled by the  $\overline{\text{LPBOR}}$  bit of the Configuration Word ([Register 5-1](#)). When the device is erased, the LPBOR module defaults to disabled.

### 8.4.2 LPBOR MODULE OUTPUT

The output of the LPBOR module is a signal indicating whether or not a Reset is to be asserted. When this occurs, a register bit (BOR) is changed to indicate that a BOR Reset has occurred. The same bit is set for either the BOR or the LPBOR (refer to [Register 8-3](#)). This signal is OR'd with the output of the BOR module to provide the generic  $\overline{\text{BOR}}$  signal, which goes to the PCON register and to the power control block. Refer to [Figure 8-1](#) for the OR gate connections of the BOR and LPBOR Reset signals, which eventually generates one common BOR Reset.

## 8.5 $\overline{\text{MCLR}}$

The  $\overline{\text{MCLR}}$  is an optional external input that can reset the device. The  $\overline{\text{MCLR}}$  function is controlled by the MCLRE bit of Configuration Words and the LVP bit of Configuration Words ([Table 8-2](#)).

**TABLE 8-2:  $\overline{\text{MCLR}}$  CONFIGURATION**

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

### 8.5.1 $\overline{\text{MCLR}}$ ENABLED

When  $\overline{\text{MCLR}}$  is enabled and the pin is held low, the device is held in Reset. The  $\overline{\text{MCLR}}$  pin is connected to VDD through an internal weak pull-up. Refer to [Section 2.3 “Master Clear \(MCLR\) Pin”](#) for recommended  $\overline{\text{MCLR}}$  connections.

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

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

### 8.5.2 $\overline{\text{MCLR}}$ DISABLED

When  $\overline{\text{MCLR}}$  is disabled, the pin functions as a general purpose input and the internal weak pull-up is under software control. See [Section 14.1 “I/O Priorities”](#) for more information.

## 8.6 Windowed Watchdog Timer (WWDT) Reset

The Watchdog Timer generates a Reset if the firmware does not issue a  $\overline{\text{CLRWDT}}$  instruction within the time-out period and the window is open. The  $\overline{\text{TO}}$  and  $\overline{\text{PD}}$  bits in the STATUS register and the  $\overline{\text{WDT}}$  bit in PCON are changed to indicate a WDT Reset caused by the timer overflowing, and WDTWV bit in the PCON register is changed to indicate a WDT Reset caused by a window violation. See [Section 12.0 “Windowed Watchdog Timer \(WWDT\)”](#) for more information.

## 8.7 RESET Instruction

A RESET instruction will cause a device Reset. The  $\overline{\text{RI}}$  bit in the PCON register will be set to '0'. See [Table 8-4](#) for default conditions after a RESET instruction has occurred.

## 8.8 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 4.5.2 “Overflow/Underflow Reset”](#) for more information.

## 8.9 Programming Mode Exit

Upon exit of In-Circuit Serial Programming™ (ICSP™) mode, the device will behave as if a POR had just occurred (the device does not reset upon run time self-programming/erase operations).

## 8.10 Power-up Timer

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

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

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  $\overline{\text{PWRT}}$  is active. The  $\overline{\text{PWRT}}$  delay allows additional time for the VDD to rise to an acceptable level. The Power-up Timer is enabled by clearing the  $\overline{\text{PWRT}}$  bit in the 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).

## 8.11 Start-up Sequence

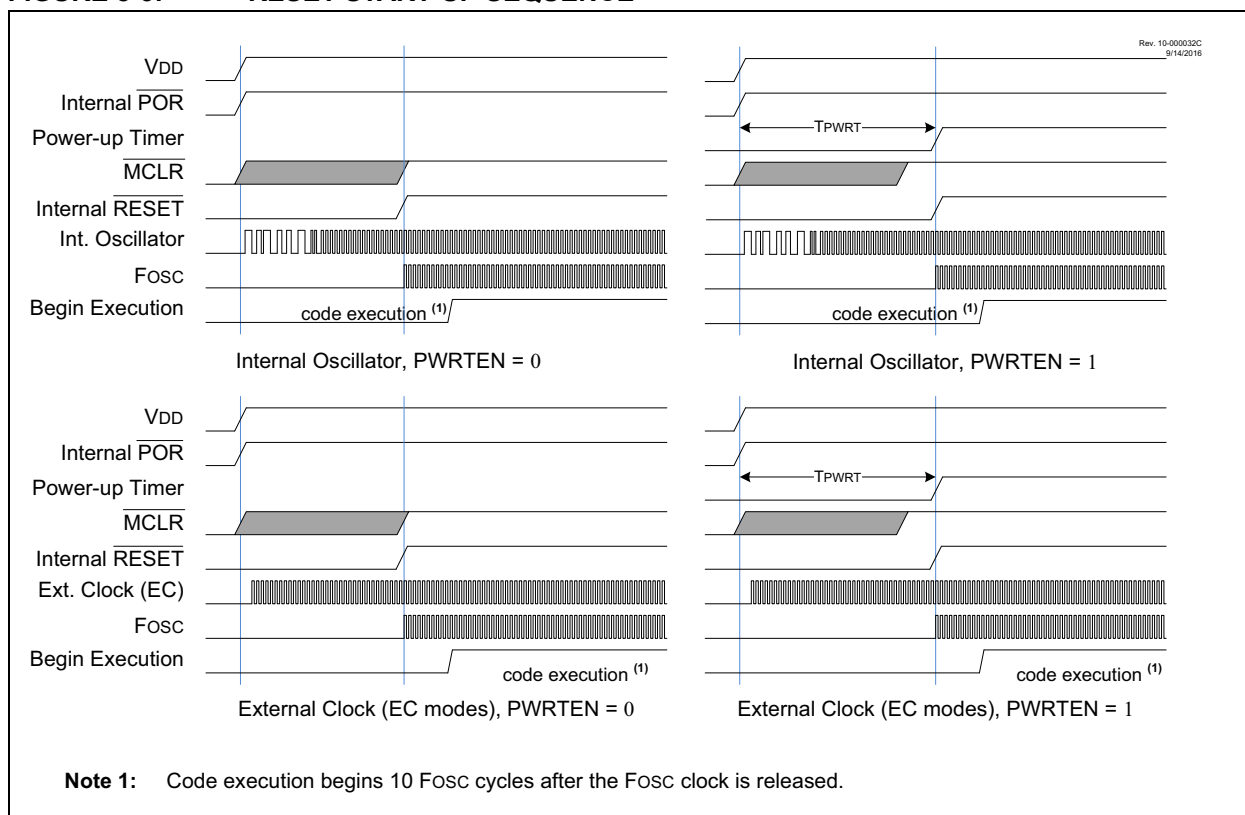
Upon the release of a POR or BOR, the following must occur before the device will begin executing:

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

The total time-out will vary based on oscillator configuration and Power-up Timer Configuration. See [Section 9.0 “Oscillator Module \(with Fail-Safe Clock Monitor\)”](#) for more information.

The Power-up Timer runs independently of  $\overline{\text{MCLR}}$  Reset. If  $\overline{\text{MCLR}}$  is kept low long enough, the Power-up Timer and oscillator start-up timer will expire. This is useful for testing purposes or to synchronize more than one device operating in parallel. See [Figure 8-3](#).

**FIGURE 8-3: RESET START-UP SEQUENCE**



## 8.12 Memory Execution Violation

A Memory Execution Violation Reset occurs if executing an instruction being fetched from outside the valid execution area. The different valid execution areas are defined as follows:

- Flash Memory: [Table 4-1](#) shows the addresses available on the PIC16(L)F15354/55 devices based on user Flash size. Execution outside this region generates a memory execution violation.
- Storage Area Flash (SAF): If Storage Area Flash (SAF) is enabled ([Section 4.2.3 “Storage Area Flash”](#)), the SAF area ([Table 4-2](#)) is not a valid execution area.

Prefetched instructions that are not executed do not cause memory execution violations. For example, a GOTO instruction in the last memory location will prefetch from an invalid location; this is not an error. If an instruction from an invalid location tries to execute, the memory violation is generated immediately, and any concurrent interrupt requests are ignored. When a memory execution violation is generated, the device is reset and flag MEMV is cleared in PCON1 ([Register 8-3](#)) to signal the cause. The flag needs to be set in code after a memory execution violation.

## 8.13 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 8-3 and Table 8-4 show the Reset conditions of these registers.

**TABLE 8-3: RESET STATUS BITS AND THEIR SIGNIFICANCE**

STOVF	STKUNF	RWD $\overline{T}$	MCLR	RI	POR	BOR	TO	PD	MEMV	Condition
0	0	1	1	1	0	x	1	1	1	Power-on Reset
0	0	1	1	1	0	x	0	x	u	Illegal, $\overline{TO}$ is set on $\overline{POR}$
0	0	1	1	1	0	x	x	0	u	Illegal, $\overline{PD}$ is set on $\overline{POR}$
0	0	u	1	1	u	0	1	1	u	Brown-out Reset
u	u	0	u	u	u	u	0	u	u	WWD $\overline{T}$ Reset
u	u	u	u	u	u	u	0	0	u	WWD $\overline{T}$ Wake-up from Sleep
u	u	u	u	u	u	u	1	0	u	Interrupt Wake-up from Sleep
u	u	u	0	u	u	u	u	u	1	$\overline{MCLR}$ Reset during normal operation
u	u	u	0	u	u	u	1	0	u	$\overline{MCLR}$ Reset during Sleep
u	u	u	u	0	u	u	u	u	u	RESET Instruction Executed
1	u	u	u	u	u	u	u	u	u	Stack Overflow Reset (STVREN = 1)
u	1	u	u	u	u	u	u	u	u	Stack Underflow Reset (STVREN = 1)
u	u	u	u	u	u	u	u	u	0	Memory violation Reset

**TABLE 8-4: RESET CONDITION FOR SPECIAL REGISTERS**

Condition	Program Counter	STATUS Register	PCON0 Register	PCON1 Register
Power-on Reset	0000h	---1 1000	0011 110x	---- --1-
$\overline{MCLR}$ Reset during normal operation	0000h	---u uuuu	uuuu 0uuu	---- --1-
$\overline{MCLR}$ Reset during Sleep	0000h	---1 0uuu	uuuu 0uuu	---- --u-
WWD $\overline{T}$ Timeout Reset	0000h	---0 uuuu	uuu0 uuuu	---- --u-
WWD $\overline{T}$ Wake-up from Sleep	PC + 1	---0 0uuu	uuuu uuuu	---- --u-
WWD $\overline{T}$ Window Violation	0000h	---u uuuu	uu0u uuuu	---- --u-
Brown-out Reset	0000h	---1 1000	0011 11u0	---- --u-
Interrupt Wake-up from Sleep	PC + 1 <sup>(1)</sup>	---1 0uuu	uuuu uuuu	---- --u-
RESET Instruction Executed	0000h	---u uuuu	uuuu u0uu	---- --u-
Stack Overflow Reset (STVREN = 1)	0000h	---u uuuu	1uuu uuuu	---- --u-
Stack Underflow Reset (STVREN = 1)	0000h	---u uuuu	u1uu uuuu	---- --u-
Memory Violation Reset ( $\overline{MEMV}$ = 0)	0	-uuu uuuu	uuuu uuuu	---- --0-

**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.

## 8.14 Power Control (PCONx) Registers

The Power Control (PCONx) registers contain flag bits to differentiate between a:

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

The PCON0 register bits are shown in [Register 8-2](#).  
The PCON1 register bits are shown in [Register 8-3](#).

Hardware will change the corresponding register bit during the Reset process; if the Reset was not caused by the condition, the bit remains unchanged ([Table 8-4](#)).

Software should reset the bit to the inactive state after the restart (hardware will not reset the bit).

Software may also set any PCON bit to the active state, so that user code may be tested, but no reset action will be generated.

## 8.15 Register Definitions: Power Control

### REGISTER 8-2: PCON0: POWER CONTROL REGISTER 0

R/W/HS-0/q	R/W/HS-0/q	R/W/HC-1/q	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	WDTWV	RWDT	RMCLR	RI	POR	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 = Bit is unchanged	x = Bit is unknown
'1' = Bit is set	'0' = Bit is cleared
	U = Unimplemented bit, read as '0'
	-m/n = Value at POR/Value at all other Resets
	q = Value depends on condition

bit 7	<p><b>STKOVF:</b> Stack Overflow Flag bit</p> <p>1 = A Stack Overflow occurred</p> <p>0 = A Stack Overflow has not occurred or cleared by firmware</p>
bit 6	<p><b>STKUNF:</b> Stack Underflow Flag bit</p> <p>1 = A Stack Underflow occurred</p> <p>0 = A Stack Underflow has not occurred or cleared by firmware</p>
bit 5	<p><b>WDTWV:</b> WDT Window Violation Flag bit</p> <p>1 = A WDT Window Violation Reset has not occurred or set to '1' by firmware</p> <p>0 = A WDT Window Violation Reset has occurred (a CLRWDT instruction was executed either without arming the window or outside the window (cleared by hardware))</p>
bit 4	<p><b>RWDT:</b> Watchdog Timer Reset Flag bit</p> <p>1 = A Watchdog Timer Reset has not occurred or set to '1' by firmware</p> <p>0 = A Watchdog Timer Reset has occurred (cleared by hardware)</p>
bit 3	<p><b>RMCLR:</b> MCLR Reset Flag bit</p> <p>1 = A MCLR Reset has not occurred or set to '1' by firmware</p> <p>0 = A MCLR Reset has occurred (cleared by hardware)</p>
bit 2	<p><b>RI:</b> RESET Instruction Flag bit</p> <p>1 = A RESET instruction has not been executed or set to '1' by firmware</p> <p>0 = A RESET instruction has been executed (cleared by hardware)</p>
bit 1	<p><b>POR:</b> Power-on Reset Status bit</p> <p>1 = No Power-on Reset occurred</p> <p>0 = A Power-on Reset occurred (must be set in software after a Power-on Reset occurs)</p>
bit 0	<p><b>BOR:</b> Brown-out Reset Status bit</p> <p>1 = No Brown-out Reset occurred</p> <p>0 = A Brown-out Reset occurred (must be set in software after a Power-on Reset or Brown-out Reset occurs)</p>

# PIC16(L)F15354/55

## REGISTER 8-3: PCON1: POWER CONTROL REGISTER 0

U-0	U-0	U-0	U-0	U-0	U-0	R/W/HC-1/u	U-0
—	—	—	—	—	—	MEMV	—
bit 7						bit 0	

### Legend:

HC = Bit is cleared 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 7-2 **Unimplemented:** Read as '0'

bit 1 **MEMV:** Memory Violation Flag bit

1 = No Memory Violation Reset occurred or set to '1' by firmware.

0 = A Memory Violation Reset occurred (set to '0' in hardware when a Memory Violation occurs)

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

**TABLE 8-5: SUMMARY OF REGISTERS ASSOCIATED WITH RESETS**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
BORCON	SBOREN	—	—	—	—	—	—	BORRDY	92
PCON0	STKOVF	STKUNF	WDTWV	RWD $\bar{T}$	RMCLR	RI	POR	BOR	99
PCON1	—	—	—	—	—	—	MEMV	—	99
STATUS	—	—	—	TO	PD	Z	DC	C	30
WDTCON0	—	—	WDTPS<4:0>					SWDTEN	150

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

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

### 9.1 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 9-1](#) illustrates a block diagram of the oscillator module.

Clock sources can be supplied from external oscillators, quartz-crystal resonators. 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.
- Fail-Safe Clock Monitor (FSCM) designed to detect a failure of the external clock source (LP, XT, HS, ECH, ECM, ECL) and switch automatically to the internal oscillator.
- Oscillator Start-up Timer (OST) ensures stability of crystal oscillator sources.

The RSTOSC bits of Configuration Word 1 determine the type of oscillator that will be used when the device reset, including when it is first powered up.

The internal clock modes, LFINTOSC, HFINTOSC (set at 1 MHz), or HFINTOSC (set at 32 MHz) can be set through the RSTOSC bits.

If an external clock source is selected, the FEXTOSC bits of Configuration Word 1 must be used to select the external clock mode.

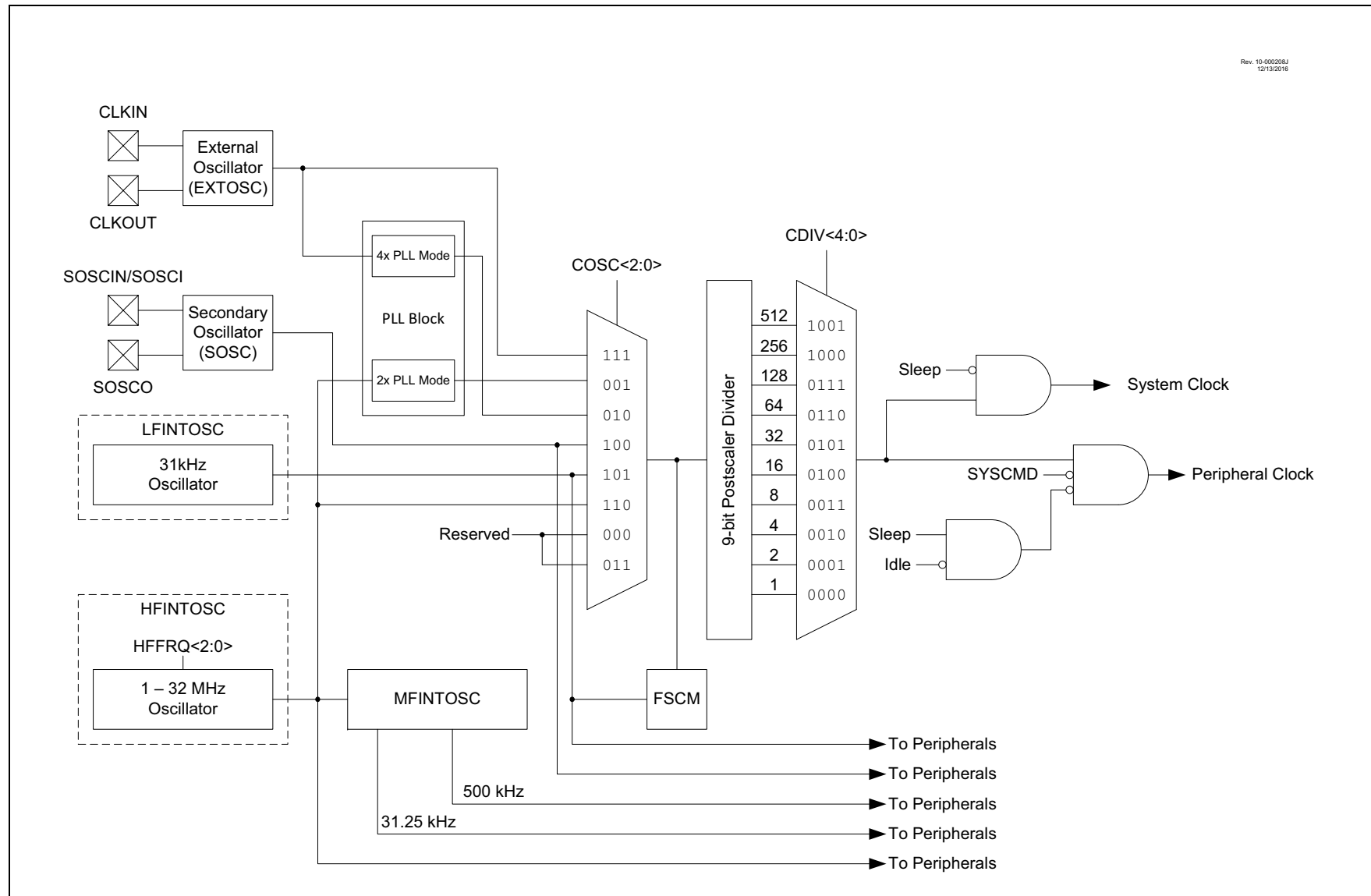
The external oscillator module can be configured in one of the following clock modes, by setting the FEXTOSC<2:0> bits of Configuration Word 1:

1. ECL – External Clock Low-Power mode  
ECL ≤ 500 kHz
2. ECM – External Clock Medium Power mode  
ECM ≤ 8 MHz
3. ECH – External Clock High-Power mode  
ECH ≤ 32 MHz
4. LP – 32 kHz Low-Power Crystal mode.
5. XT – Medium Gain Crystal or Ceramic Resonator Oscillator mode (between 100 kHz and 4 MHz)
6. HS – High Gain Crystal or Ceramic Resonator mode (above 4 MHz)

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 INTOSC internal oscillator block produces low and high-frequency clock sources, designated LFINTOSC and HFINTOSC. (see Internal Oscillator Block, [Figure 9-1](#)). A wide selection of device clock frequencies may be derived from these clock sources.



**FIGURE 9-1: SIMPLIFIED PIC® MCU CLOCK SOURCE BLOCK DIAGRAM**



## 9.2 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).

There is also a secondary oscillator block which is optimized for a 32.768 kHz external clock source, which can be used as an alternate clock source.

There are two internal oscillator blocks:

- HFINTOSC
- LFINTOSC

The HFINTOSC can produce clock frequencies from 1-32 MHz, and is responsible for generating the two MFINTOSC frequencies (500 kHz and 32 kHz) that can be used by some peripherals. The LFINTOSC generates a 31 kHz clock frequency.

There is a 4x PLL that can be used by the external oscillator. See [Section 9.2.1.4 “4x PLL”](#) for more details. Additionally, there is a PLL that can be used by the HFINTOSC at certain frequencies. See [Section 9.2.2.2 “Internal Oscillator Frequency Adjustment”](#) for more details.

### 9.2.1 EXTERNAL CLOCK SOURCES

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

- Program the RSTOSC<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 NOSC<2:0> and NDIV<3:0> bits in the OSCCON1 register to switch the system clock source

See [Section 9.3 “Clock Switching”](#) for more information.

#### 9.2.1.1 EC Mode

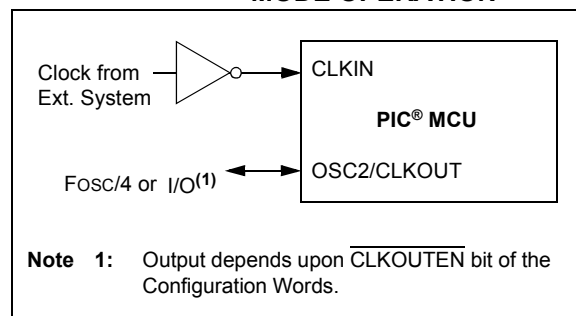
The External Clock (EC) mode allows an externally generated logic level signal to be the system clock source. When operating in this mode, an external clock source is connected to the OSC1/CLKIN input. OSC2/CLKOUT is available for general purpose I/O or CLKOUT. [Figure 9-2](#) shows the pin connections for EC mode.

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

- ECH – High power, ≤ 32 MHz
- ECM – Medium power, ≤ 8 MHz
- ECL – Low power, ≤ 0.1 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 9-2: EXTERNAL CLOCK (EC) MODE OPERATION**



#### 9.2.1.2 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 9-3](#)). The three modes select a low, medium or high gain setting of the internal inverter-amplifier to support various resonator types and speed.

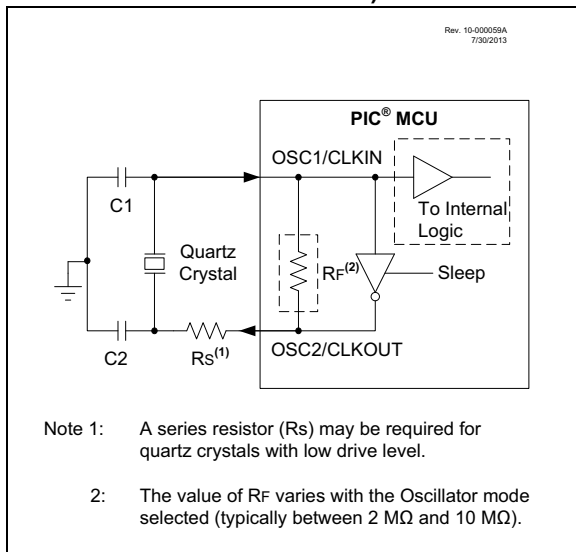
**LP** Oscillator mode selects the lowest gain setting of the internal inverter-amplifier. LP mode current consumption is the least of the three modes. This mode is designed to drive only 32.768 kHz tuning-fork type crystals (watch crystals), but can operate up to 100 kHz.

**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 crystals and resonators with a frequency range up to 4 MHz.

**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 operating frequencies up to 20 MHz.

[Figure 9-3](#) and [Figure 9-4](#) show typical circuits for quartz crystal and ceramic resonators, respectively.

**FIGURE 9-3: QUARTZ CRYSTAL OPERATION (LP, XT OR HS MODE)**



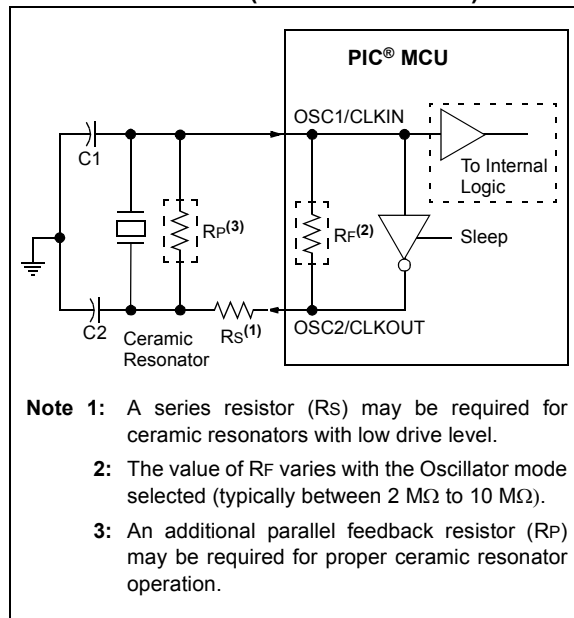
**Note 1:** Quartz crystal characteristics vary according to type, package and manufacturer. The user should consult the manufacturer data sheets for specifications and recommended application.

**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 *rPIC*<sup>®</sup> and *PIC*<sup>®</sup> Devices" (DS00826)
- AN849, "Basic *PIC*<sup>®</sup> Oscillator Design" (DS00849)
- AN943, "Practical *PIC*<sup>®</sup> Oscillator Analysis and Design" (DS00943)
- AN949, "Making Your Oscillator Work" (DS00949)

**FIGURE 9-4: CERAMIC RESONATOR OPERATION (XT OR HS MODE)**



### 9.2.1.3 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), Brown-out Reset (BOR) or a wake-up from Sleep. 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.

## 9.2.1.4 4x PLL

The oscillator module contains a PLL that can be used with external clock sources to provide a system clock source. The input frequency for the PLL must fall within specifications. See the PLL Clock Timing Specifications in [Table 37-9](#).

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

1. Program the RSTOSC bits in the Configuration Word 1 to enable the EXTOSC with 4x PLL (RSTOSC<2:0> = '010').
2. Write the NOSC bits in the OSCCON1 register to enable the EXTOSC with 4x PLL (NOSC<2:0> = '010').

## 9.2.1.5 Secondary Oscillator

The secondary oscillator is a separate oscillator block that can be used as an alternate system clock source. The secondary oscillator is optimized for 32.768 kHz, and can be used with an external crystal oscillator connected to the SOSCI and SOSCO device pins, or an external clock source connected to the SOSCIN pin. Refer to [Section 9.3 “Clock Switching”](#) for more information.

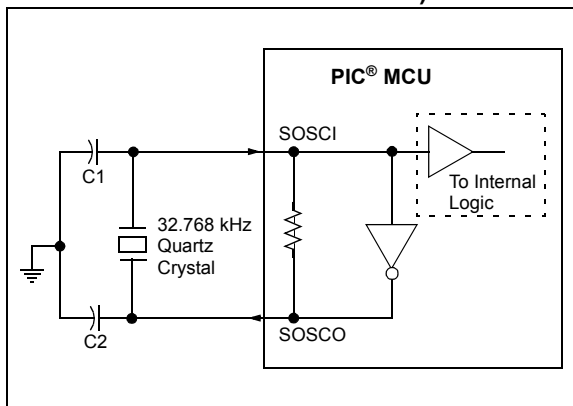
**Note 1:** Quartz crystal characteristics vary according to type, package and manufacturer. The user should consult the manufacturer data sheets for specifications and recommended application.

**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)
- 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)

**FIGURE 9-5: QUARTZ CRYSTAL OPERATION (SECONDARY OSCILLATOR)**



## 9.2.2 INTERNAL CLOCK SOURCES

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

- Program the RSTOSC<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 NOSC<2:0> bits in the OSCCON1 register to switch the system clock source to the internal oscillator during run-time. See [Section 9.3 “Clock Switching”](#) for more information.

In **INTOSC** mode, the OSC1/CLKIN pin is available for general purpose I/O. The OSC2/CLKOUT pin 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 that can produce two internal system clock sources.

1. The **HFINTOSC** (High-Frequency Internal Oscillator) is factory calibrated and operates up to 32 MHz.
2. The **LFINTOSC** (Low-Frequency Internal Oscillator) is factory-calibrated and operates at 31 kHz.

### 9.2.2.1 HFINTOSC

The High-Frequency Internal Oscillator (HFINTOSC) is a precision digitally-controlled internal clock source that produces a stable clock up to 32 MHz. The HFINTOSC can be enabled through one of the following methods:

- Programming the RSTOSC<2:0> bits in Configuration Word 1 to ‘110’ (1 MHz) or ‘001’ (32 MHz) to set the oscillator upon device Power-up or Reset.
- Write to the NOSC<2:0> bits of the OSCCON1 register during run-time.

The HFINTOSC frequency can be selected by setting the HFFRQ<2:0> bits of the OSCFRQ register. The MFINTOSC is an internal clock source within the HFINTOSC that provides two (500 kHz, 32 kHz) constant clock outputs. These constant clock outputs are available for selection to various peripherals, internally.

The NDIV<3:0> bits of the OSCCON1 register allow for division of the HFINTOSC output from a range between 1:1 and 1:512.

## 9.2.2.2 Internal Oscillator Frequency Adjustment

The HFINTOSC and LFINTOSC internal oscillators are both factory-calibrated. The HFINTOSC oscillator can be adjusted in software by writing to the OSCTUNE register ([Register 9-7](#)). OSCTUNE does not affect the LFINTOSC frequency.

The default value of the OSCTUNE register is 00h. 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 current HFINTOSC oscillator frequency will begin shifting to the new frequency. Code execution continues during this shift. There is no indication that the shift has occurred.

## 9.2.2.3 LFINTOSC

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

The LFINTOSC is the clock source for the Power-up Timer (PWRT), Watchdog Timer (WDT), and Fail-Safe Clock Monitor (FSCM). The LFINTOSC can also be used as the system clock, or as a clock or input source to certain peripherals.

The LFINTOSC is selected as the clock source through one of the following methods:

- Programming the RSTOSC<2:0> bits of Configuration Word 1 to enable LFINTOSC (RSTOSC<2:0> = '101')
- Write to the NOSC<2:0> bits of the OSCCON1 register (NOSC<2:0> = '101')

Peripherals that use the LFINTOSC are:

- Power-up Timer (PWRT)
- Windowed Watchdog Timer (WWDT)
- Timer1
- Timer0
- Timer2
- Fail-Safe Clock Monitor (FSCM)
- CLKR
- CLC

## 9.2.2.4 Oscillator Status and Manual Enable

The 'ready' status of each oscillator is displayed in the OSCSTAT register ([Register 9-4](#)). The oscillators can also be manually enabled through the OSCEN register ([Register 9-7](#)). Manual enabling makes it possible to verify the operation of the EXTOSC or SOSC crystal oscillators. This can be achieved by enabling the selected oscillator, then watching the corresponding 'ready' state of the oscillator in the OSCSTAT register.

## 9.3 Clock Switching

The system clock source can be switched between external and internal clock sources via software using the New Oscillator Source (NOSC) and New Divider selection request (NDIV) bits of the OSCCON1 register. The following clock sources can be selected:

- External Oscillator (EXTOSC)
- High-Frequency Internal Oscillator (HFINTOSC)
- Low-Frequency Internal Oscillator (LFINTOSC)
- Secondary Oscillator (SOSC)
- EXTOSC with 4x PLL
- HFINTOSC with 2x PLL

### 9.3.1 NEW OSCILLATOR SOURCE (NOSC) AND NEW DIVIDER SELECTION REQUEST (NDIV) BITS

The New Oscillator Source (NOSC) and New Divider selection request (NDIV) bits of the OSCCON1 register select the system clock source and the frequency that are used for the CPU and peripherals.

When new values of NOSC and NDIV are written to OSCCON1, the current oscillator selection will continue to operate while waiting for the new clock source to indicate that it is stable and ready. In some cases, the newly requested source may already be in use, and is ready immediately. In the case of a divider-only change, the new and old sources are the same, and will be immediately ready. The device may enter Sleep while waiting for the switch as described in [Section 9.3.3 "Clock Switch and Sleep"](#).

When the new oscillator is ready, the New Oscillator is Ready (NOSCR) bit of OSCCON3 and the Clock Switch Interrupt Flag (CSWIF) bit of PIR1 become set (CSWIF = 1). If Clock Switch Interrupts are enabled (CSWIE = 1), an interrupt will be generated at that time. The Oscillator Ready (ORDY) bit of OSCCON3 can also be polled to determine when the oscillator is ready in lieu of an interrupt.

If the Clock Switch Hold (CSWHOLD) bit of OSCCON3 is clear, the oscillator switch will occur when the new Oscillator's READY bit (NOSCR) is set, and the interrupt (if enabled) will be serviced at the new oscillator setting.

If CSWHOLD is set, the oscillator switch is suspended, while execution continues using the current (old) clock source. When the NOSCR bit is set, software should:

- set CSWHOLD = 0 so the switch can complete, or
- copy COSC into NOSC to abandon the switch.

If DOZE is in effect, the switch occurs on the next clock cycle, whether or not the CPU is operating during that cycle.

Changing the clock post-divider without changing the clock source (e.g., changing Fosc from 1 MHz to 2 MHz) is handled in the same manner as a clock source change, as described previously. The clock source will already be active, so the switch is relatively quick. CSWHOLD must be clear (CSWHOLD = 0) for the switch to complete.

The current COSC and CDIV are indicated in the OSCCON2 register up to the moment when the switch actually occurs, at which time OSCCON2 is updated and ORDY is set. NOSCR is cleared by hardware to indicate that the switch is complete.

### 9.3.2 PLL INPUT SWITCH

Switching between the PLL and any non-PLL source is managed as described above. The input to the PLL is established when NOSCR selects the PLL, and maintained by the COSC setting.

When NOSCR and COSC select the PLL with different input sources, the system continues to run using the COSC setting, and the new source is enabled per NOSCR. When the new oscillator is ready (and CSWHOLD = 0), system operation is suspended while the PLL input is switched and the PLL acquires lock.

### 9.3.3 CLOCK SWITCH AND SLEEP

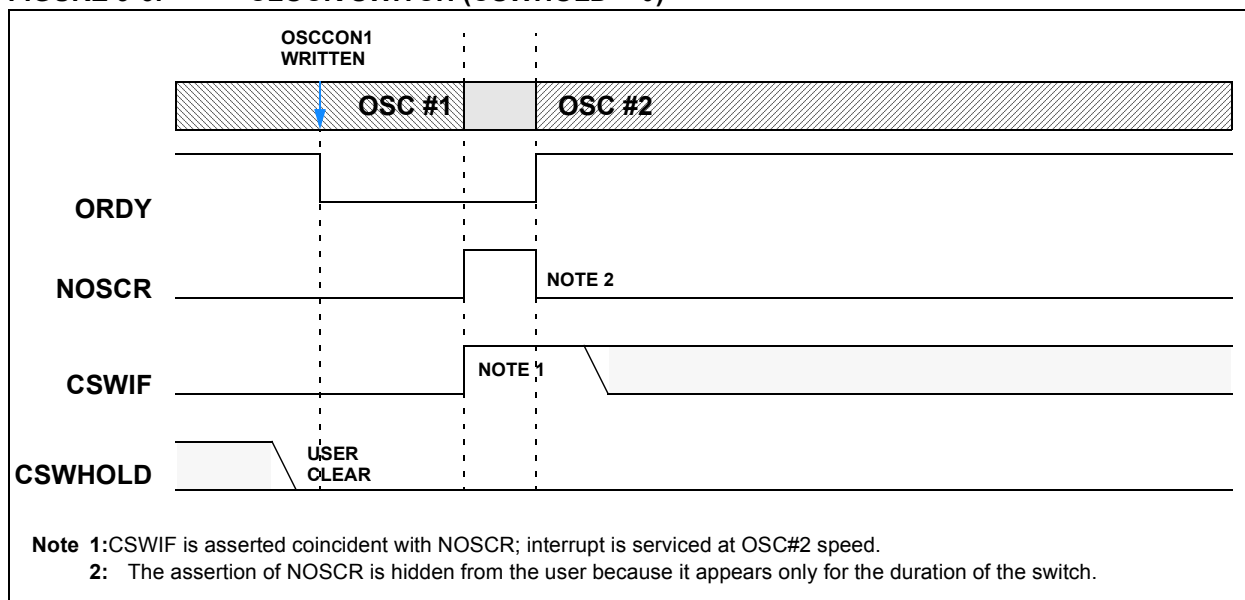
If OSCCON1 is written with a new value and the device is put to Sleep before the switch completes, the switch will not take place and the device will enter Sleep mode.

When the device wakes from Sleep and the CSWHOLD bit is clear, the device will wake with the 'new' clock active, and the clock switch interrupt flag bit (CSWIF) will be set.

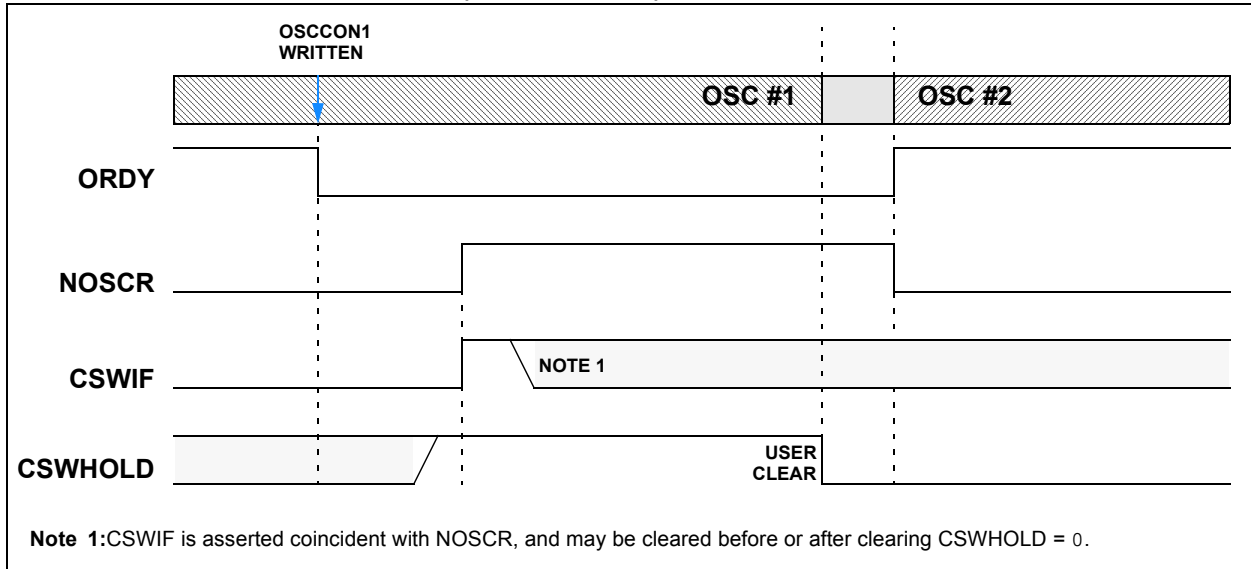
When the device wakes from Sleep and the CSWHOLD bit is set, the device will wake with the 'old' clock active and the new clock will be requested again.

**Note:** If the PLL fails to lock, the FSCM will trigger.

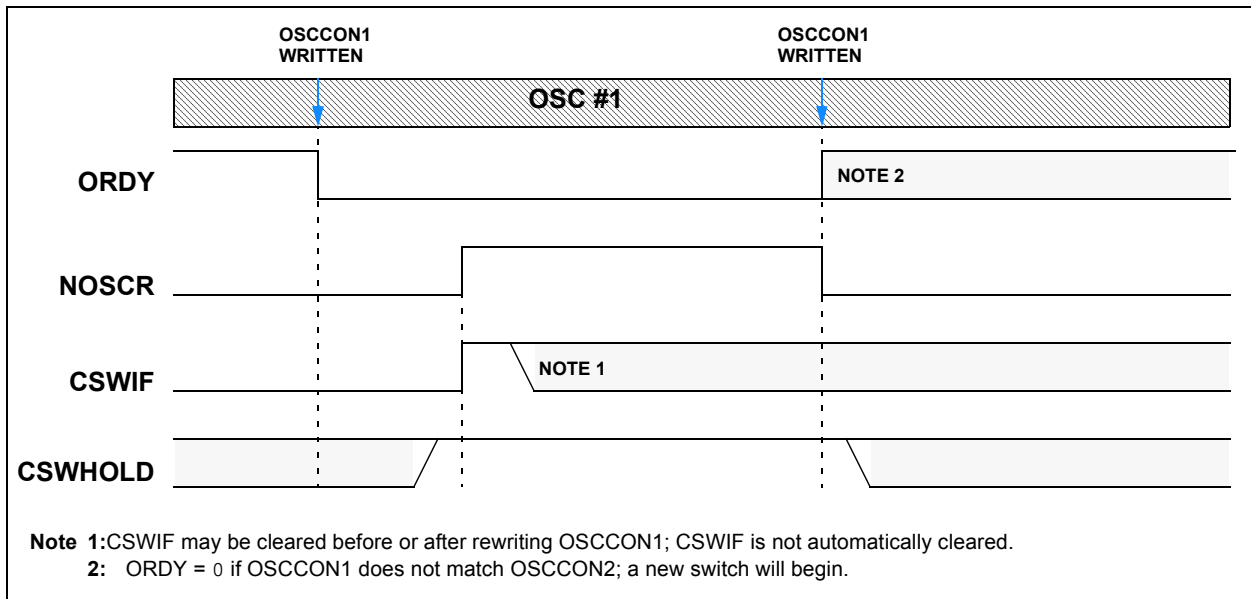
**FIGURE 9-6: CLOCK SWITCH (CSWHOLD = 0)**



**FIGURE 9-7: CLOCK SWITCH (CSWHOLD = 1)**



**FIGURE 9-8: CLOCK SWITCH ABANDONED**

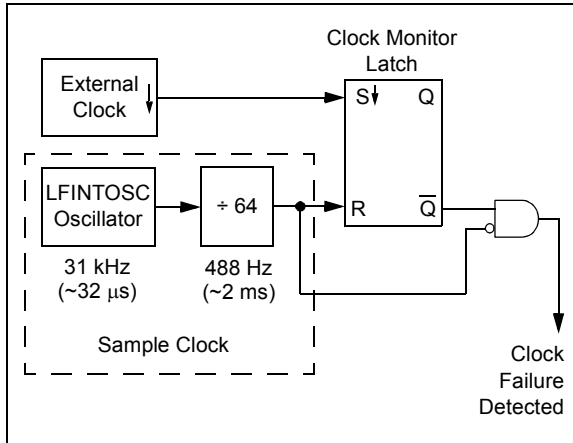




## 9.4 Fail-Safe Clock Monitor

The Fail-Safe Clock Monitor (FSCM) allows the device to continue operating should the external oscillator fail. The FSCM is enabled by setting the FCMEN bit in Configuration Word 1. The FSCM is applicable to all external Oscillator modes (LP, XT, HS, ECL, ECM, ECH and Secondary Oscillator).

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



### 9.4.1 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 9-9. 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.

### 9.4.2 FAIL-SAFE OPERATION

When the external clock fails, the FSCM switches the device clock to the HFINTOSC at 1 MHz clock frequency and sets the bit flag OSFIF of the PIR1 register. Setting this flag will generate an interrupt if the OSFIE bit of the PIE1 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, by writing to the NOSC and NDIV bits of the OSCCON1 register.

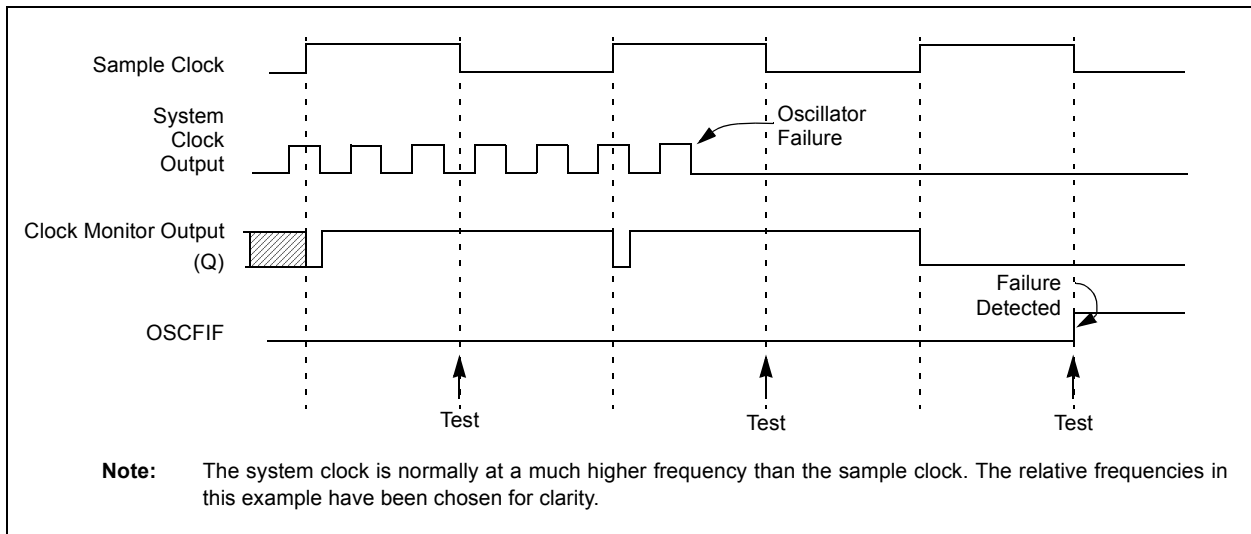
### 9.4.3 FAIL-SAFE CONDITION CLEARING

The Fail-Safe condition is cleared after a Reset, executing a SLEEP instruction or changing the NOSC and NDIV bits of the OSCCON1 register. When switching to the external oscillator, or external oscillator and PLL, the OST is restarted. While the OST is running, the device continues to operate from the INTOSC selected in OSCCON1. 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.

### 9.4.4 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 Clock modes so that the FSCM will be active as soon as the Reset or wake-up has completed. Therefore, the device will always be executing code while the OST is operating.

**FIGURE 9-10: FSCM TIMING DIAGRAM**



## 9.5 Register Definitions: Oscillator Control

**REGISTER 9-1: OSCCON1: OSCILLATOR CONTROL REGISTER1**

U-0	R/W-f/f <sup>(1)</sup>	R/W-f/f <sup>(1)</sup>	R/W-f/f <sup>(1)</sup>	R/W-q/q	R/W-q/q	R/W-q/q	R/W-q/q
—	NOSC<2:0> <sup>(2,3)</sup>			NDIV<3:0> <sup>(2,3,4)</sup>			
bit 7							bit 0

**Legend:**

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

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

bit 6-4 **NOSC<2:0>:** New Oscillator Source Request bits  
The setting requests a source oscillator and PLL combination per [Table 9-1](#).  
POR value = RSTOSC ([Register 5-1](#)).

bit 3-0 **NDIV<3:0>:** New Divider Selection Request bits  
The setting determines the new postscaler division ratio per [Table 9-1](#).

- Note 1:** The default value (f/f) is set equal to the RSTOSC Configuration bits.  
**2:** If NOSC is written with a reserved value ([Table 9-1](#)), the operation is ignored and neither NOSC nor NDIV is written.  
**3:** When CSWEN = 0, this register is read-only and cannot be changed from the POR value.  
**4:** When NOSC = 110 (HFINTOSC 1 MHz), the NDIV bits will default to '0010' upon Reset; for all other NOSC settings the NDIV bits will default to '0000' upon Reset.

**REGISTER 9-2: OSCCON2: OSCILLATOR CONTROL REGISTER 2**

U-0	R-n/n <sup>(2)</sup>	R-n/n <sup>(2)</sup>	R-n/n <sup>(2)</sup>	R-n/n <sup>(2)</sup>	R-n/n <sup>(2)</sup>	R-n/n <sup>(2)</sup>	R-n/n <sup>(2)</sup>
—	COSC<2:0>			CDIV<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 **Unimplemented:** Read as '0'

bit 6-4 **COSC<2:0>:** Current Oscillator Source Select bits (read-only)  
Indicates the current source oscillator and PLL combination per [Table 9-1](#).

bit 3-0 **CDIV<3:0>:** Current Divider Select bits (read-only)  
Indicates the current postscaler division ratio per [Table 9-1](#).

- Note 1:** The POR value is the value present when user code execution begins.  
**2:** The Reset value (n/n) is the same as the NOSC/NDIV bits.

**TABLE 9-1: NOSC/COSC BIT SETTINGS**

NOSC<2:0>/ COSC<2:0>	Clock Source
111	EXTOSC <sup>(1)</sup>
110	HFINTOSC (1 MHz) <sup>(2)</sup>
101	LFINTOSC
100	SOSC
011	Reserved
010	EXTOSC with 4x PLL <sup>(1)</sup>
001	HFINTOSC with 2x PLL (32 MHz) <sup>(1)</sup>
000	HFINTOSC (32 MHz)

- Note** 1: EXTOSC configured by the FEXTOSC bits of Configuration Word 1 (Register 5-1).  
 2: HFINTOSC settings are configured with the HFFRQ bits of the OSCFRQ register (Register 9-6).

**TABLE 9-2: NDIV/CDIV BIT SETTINGS**

NDIV<3:0>/ CDIV<3:0>	Clock divider
1111-1010	Reserved
1001	512
1000	256
0111	128
0110	64
0101	32
0100	16
0011	8
0010	4
0001	2
0000	1

**REGISTER 9-3: OSCCON3: OSCILLATOR CONTROL REGISTER 3**

R/W/HC-0/0	R/W-0/0	U-0	R-0/0	R-0/0	U-0	U-0	U-0
CSWHOLD	SOSCPWR	—	ORDY	NOSCR	—	—	—
bit 7							bit 0

**Legend:**

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

- bit 7            **CSWHOLD:** Clock Switch Hold bit  
 1 = Clock switch will hold (with interrupt) when the oscillator selected by NOSC is ready  
 0 = Clock switch may proceed when the oscillator selected by NOSC is ready; if this bit is clear at the time that NOSCR becomes '1', the switch will occur
- bit 6            **SOSCPWR:** Secondary Oscillator Power Mode Select bit  
 1 = Secondary oscillator operating in High-power mode  
 0 = Secondary oscillator operating in Low-power mode
- bit 5            **Unimplemented:** Read as '0'.
- bit 4            **ORDY:** Oscillator Ready bit (read-only)  
 1 = OSCCON1 = OSCCON2; the current system clock is the clock specified by NOSC  
 0 = A clock switch is in progress
- bit 3            **NOSCR:** New Oscillator is Ready bit (read-only)  
 1 = A clock switch is in progress and the oscillator selected by NOSC indicates a "ready" condition  
 0 = A clock switch is not in progress, or the NOSC-selected oscillator is not yet ready
- bit 2-0        **Unimplemented:** Read as '0'

## REGISTER 9-4: OSCSTAT: OSCILLATOR STATUS REGISTER 1

R-q/q	R-q/q	R-q/q	R-q/q	R-q/q	R-q/q	U-0	R-q/q
EXTOR	HFOR	MFOR	LFOR	SOR	ADOR	—	PLLr
bit 7						bit 0	

### Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	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 = Reset value is determined by hardware

bit 7	<b>EXTOR:</b> EXTOSC (external) Oscillator Ready bit 1 = The oscillator is ready to be used 0 = The oscillator is not enabled, or is not yet ready to be used.
bit 6	<b>HFOR:</b> HFINTOSC Oscillator Ready bit 1 = The oscillator is ready to be used 0 = The oscillator is not enabled, or is not yet ready to be used.
bit 5	<b>MFOR:</b> MFINTOSC Oscillator Ready bit 1 = The oscillator is ready to be used 0 = The oscillator is not enabled, or is not yet ready to be used.
bit 4	<b>LFOR:</b> LFINTOSC Oscillator Ready bit 1 = The oscillator is ready to be used 0 = The oscillator is not enabled, or is not yet ready to be used.
bit 3	<b>SOR:</b> Secondary (Timer1) Oscillator Ready bit 1 = The oscillator is ready to be used 0 = The oscillator is not enabled, or is not yet ready to be used.
bit 2	<b>ADOR:</b> CRC Oscillator Ready bit 1 = The oscillator is ready to be used 0 = The oscillator is not enabled, or is not yet ready to be used.
bit 1	<b>Unimplemented:</b> Read as '0'
bit 0	<b>PLLr:</b> PLL is Ready bit 1 = The PLL is ready to be used 0 = The PLL is not enabled, the required input source is not ready, or the PLL is not locked.

## REGISTER 9-5: OSCEN: OSCILLATOR MANUAL ENABLE 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	U-0	U-0
EXTOEN	HFOEN	MFOEN	LFOEN	SOSCEN	ADOEN	—	—
bit 7						bit 0	

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

u = Bit is unchanged

x = Bit is unknown

-n/n = Value at POR and BOR/Value at all other Resets

'1' = Bit is set

'0' = Bit is cleared

- bit 7      **EXTOEN:** External Oscillator Manual Request Enable bit<sup>(1)</sup>  
 1 = EXTOSC is explicitly enabled, operating as specified by FEXTOSC  
 0 = EXTOSC could be enabled by some modules
- bit 6      **HFOEN:** HFINTOSC Oscillator Manual Request Enable bit  
 1 = HFINTOSC is explicitly enabled, operating as specified by OSCFRQ  
 0 = HFINTOSC could be enabled by another module
- bit 5      **MFOEN:** MFINTOSC Oscillator Manual Request Enable bit  
 1 = MFINTOSC is explicitly enabled  
 0 = MFINTOSC could be enabled by another module
- bit 4      **LFOEN:** LFINTOSC (31 kHz) Oscillator Manual Request Enable bit  
 1 = LFINTOSC is explicitly enabled  
 0 = LFINTOSC could be enabled by another module
- bit 3      **SOSCEN:** Secondary (Timer1) Oscillator Manual Request bit  
 1 = Secondary oscillator is explicitly enabled, operating as specified by SOSCPWR  
 0 = Secondary oscillator could be enabled by another module
- bit 2      **ADOEN:** FRC Oscillator Manual Request Enable bit  
 1 = FRC is explicitly enabled  
 0 = FRC could be enabled by another module
- bit 1-0    **Unimplemented:** Read as '0'

## REGISTER 9-6: OSCFRQ: HFINTOSC FREQUENCY SELECTION REGISTER

U-0	U-0	U-0	U-0	U-0	R/W-q/q	R/W-q/q	R/W-q/q
—	—	—	—	—	HFFRQ<2:0> <sup>(1)</sup>		
bit 7					bit 0		

### Legend:

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

bit 7-3      **Unimplemented:** Read as '0'

bit 2-0      **HFFRQ<2:0>:** HFINTOSC Frequency Selection bits

#### Nominal Freq (MHz):

111 = Reserved

110 = 32

101 = 16

100 = 12

011 = 8

010 = 4

001 = 2

000 = 1

**Note 1:** When RSTOSC=110 (HFINTOSC 1 MHz), the HFFRQ bits will default to '010' upon Reset; when RSTOSC = 001 (HFINTOSC 32 MHz), the HFFRQ bits will default to '101' upon Reset.

## REGISTER 9-7: OSCTUNE: HFINTOSC TUNING REGISTER

U-0	U-0	R/W-1/1	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	
—	—	HFTUN<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

**HFTUN<5:0>:** HFINTOSC Frequency Tuning bits

01 1111 = Maximum frequency

01 1110 =

...

00 0001 =

00 0000 = Center frequency. Oscillator module is running at the calibrated frequency (default value).

11 1111 =

...

10 0001 =

10 0000 = Minimum frequency.

# PIC16(L)F15354/55

**TABLE 9-3: SUMMARY OF REGISTERS ASSOCIATED WITH CLOCK SOURCES**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
OSCCON1	—	NOSC<2:0>			NDIV<3:0>				110
OSCCON2	—	COSC<2:0>			CDIV<3:0>				110
OSCCON3	CWSHOLD	SOSCPWR	—	ORDY	NOSCR	—	—	—	111
OSCFRQ	—	—	—	—	—	HFFRQ<2:0>			114
OSCSTAT	EXTOR	HFOR	MFOR	LFOR	SOR	ADOR	—	PLLOR	112
OSCTUNE	—	—	HFTUN<5:0>						115
OSCCON	EXTOEN	HFOEN	MFOEN	LFOEN	SOSCEN	ADOEN	—	—	113

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

**TABLE 9-4: SUMMARY OF CONFIGURATION WORD WITH CLOCK SOURCES**

Name	Bits	Bit -7	Bit -6	Bit 13/5	Bit 12/4	Bit 11/3	Bit 10/2	Bit 9/1	Bit 8/0	Register on Page
CONFIG1	13:8	—	—	FCMEN	—	CSWEN	—	—	CLKOUTEN	76
	7:0	—	RSTOSC<2:0>			—	FEXTOSC<2:0>			

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



## 10.0 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.

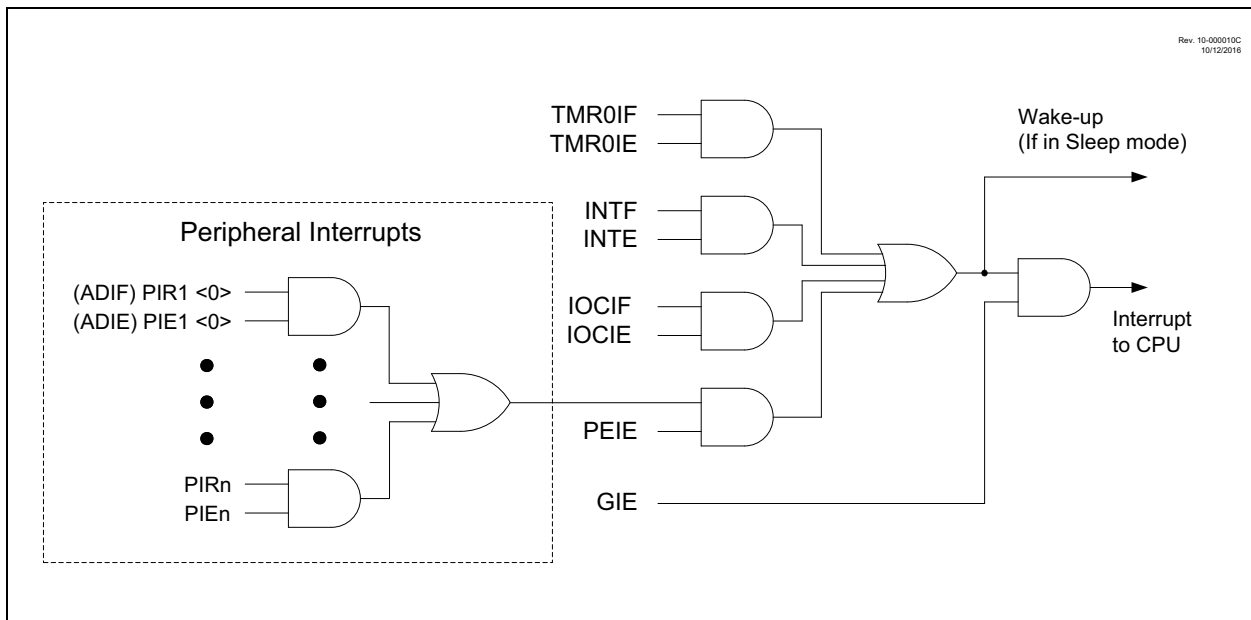
This chapter contains the following information for Interrupts:

- 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 10-1](#).

**FIGURE 10-1: INTERRUPT LOGIC**



## 10.1 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) of the PIRx[y] registers 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 PIRx registers)

The PIR1, PIR2, PIR3, PIR4, PIR5, PIR6, and PIR7 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 10.5 “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.

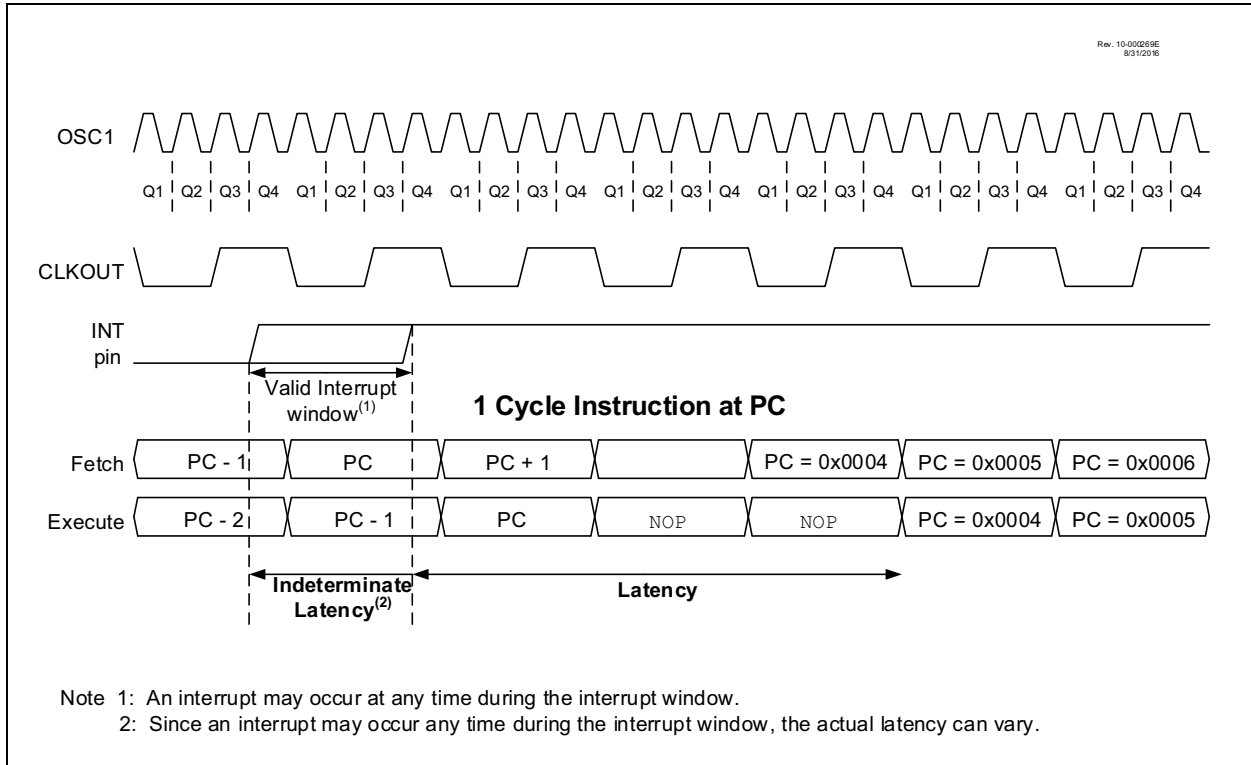
For additional information on a specific interrupts operation, refer to its peripheral chapter.

- |  |
|--|
| <p><b>Note 1:</b> Individual interrupt flag bits are set, regardless of the state of any other enable bits.</p> <p><b>2:</b> 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.</p> |
|--|

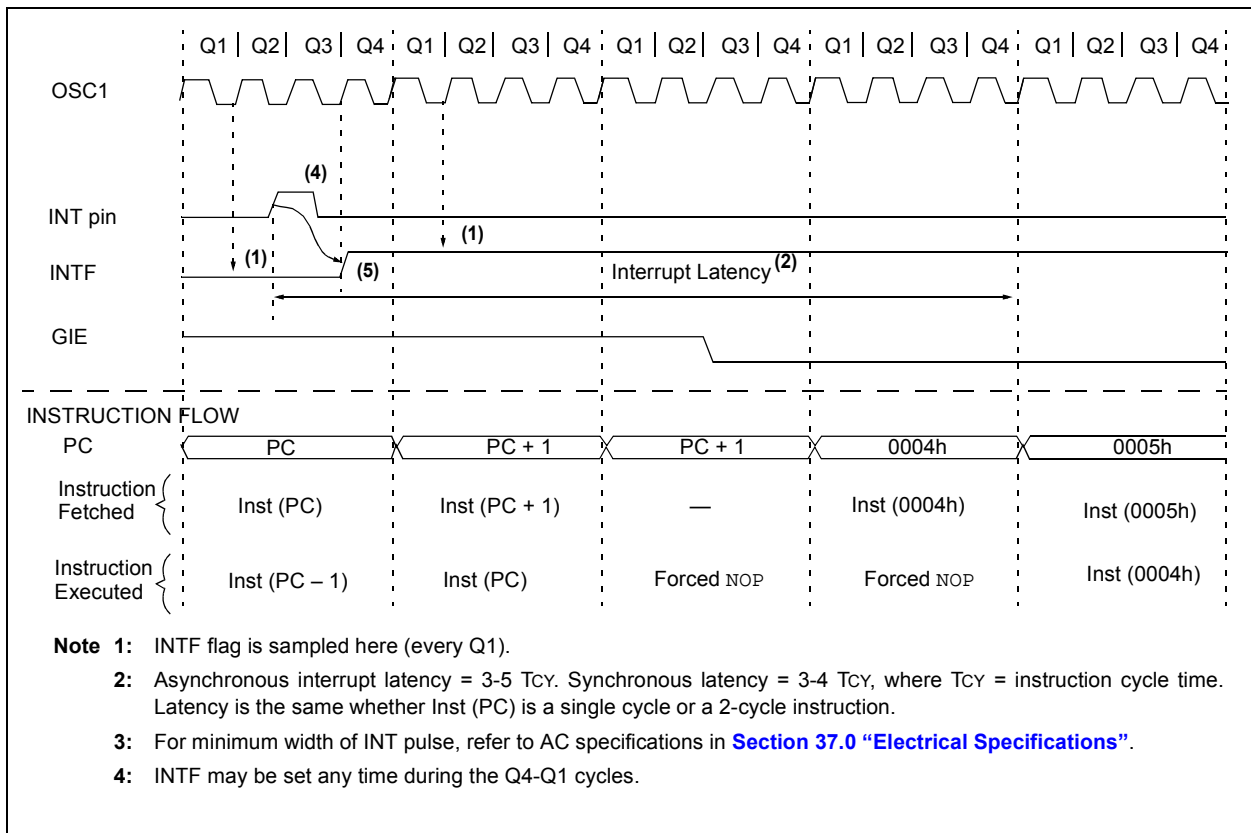
## 10.2 Interrupt Latency

Interrupt latency is defined as the time from when the interrupt event occurs to the time code execution at the interrupt vector begins. The interrupt is sampled during Q1 of the instruction cycle. The actual interrupt latency then depends on the instruction that is executing at the time the interrupt is detected. See [Figure 10-2](#) and [Figure 10-3](#) for more details.

**FIGURE 10-2: INTERRUPT LATENCY**



**FIGURE 10-3: INT PIN INTERRUPT TIMING**



## 10.3 Interrupts During Sleep

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 11.0 “Power-Saving Operation Modes”](#) for more details.

## 10.4 INT Pin

The INT pin can be used to generate an asynchronous edge-triggered interrupt. Refer to [Figure 10-3](#). This interrupt is enabled by setting the INTE bit of the PIE0 register. The INTEDG bit of the INTCON 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 PIR0 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.

## 10.5 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.

## 10.6 Register Definitions: Interrupt Control

**REGISTER 10-1: INTCON: INTERRUPT CONTROL REGISTER**

R/W-0/0	R/W-0/0	U-0	U-0	U-0	U-0	U-0	R/W-1/1
GIE	PEIE	—	—	—	—	—	INTEDG
bit 7							bit 0

**Legend:**

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit 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 5-1    **Unimplemented:** Read as '0'
- bit 0      **INTEDG:** Interrupt Edge Select bit  
1 = Interrupt on rising edge of INT pin  
0 = Interrupt on falling edge of INT pin

**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 10-2: PIE0: PERIPHERAL INTERRUPT ENABLE REGISTER 0

U-0	U-0	R/W-0/0	R/W-0/0	U-0	U-0	U-0	R/W-0/0
—	—	TMR0IE	IOCIE	—	—	—	INTE
bit 7							bit 0

### Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	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 = Hardware set

- bit 7-6      **Unimplemented:** Read as '0'
- bit 5      **TMR0IE:** Timer0 Overflow Interrupt Enable bit  
 1 = Enables the Timer0 interrupt  
 0 = Disables the Timer0 interrupt
- bit 4      **IOCIE:** Interrupt-on-Change Interrupt Enable bit  
 1 = Enables the IOC change interrupt  
 0 = Disables the IOC change interrupt
- bit 3-1      **Unimplemented:** Read as '0'
- bit 0      **INTE:** INT External Interrupt Flag bit<sup>(1)</sup>  
 1 = Enables the INT external interrupt  
 0 = Disables the INT external interrupt

**Note 1:** The External Interrupt INT pin is selected by INTPPS ([Register 15-1](#)).

**Note:** Bit PEIE of the INTCON register must be set to enable any peripheral interrupt controlled by PIE1-PIE7. Interrupt sources controlled by the PIE0 register do not require PEIE to be set in order to allow interrupt vectoring (when GIE is set).

## REGISTER 10-3: PIE1: PERIPHERAL INTERRUPT ENABLE REGISTER 1

R/W-0/0	R/W-0/0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0
OSFIE	CSWIE	—	—	—	—	—	ADIE
bit 7							bit 0

### Legend:

R = Readable bit                      W = Writable bit                      U = Unimplemented bit, read as '0'  
 u = Bit is unchanged                  x = Bit 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            **CSWIE:** Clock Switch Complete Interrupt Enable bit  
                   1 = The clock switch module interrupt is enabled  
                   0 = The clock switch module interrupt is disabled
- bit 5-1        **Unimplemented:** Read as '0'
- bit 0            **ADIE:** Analog-to-Digital Converter (ADC) Interrupt Enable bit  
                   1 = Enables the ADC interrupt  
                   0 = Disables the ADC interrupt

**Note:** Bit PEIE of the INTCON register must be set to enable any peripheral interrupt controlled by registers PIE1-PIE7

## REGISTER 10-4: PIE2: PERIPHERAL INTERRUPT ENABLE REGISTER 2

U-0	R/W-0/0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0
—	ZCDIE	—	—	—	—	C2IE	C1IE
bit 7						bit 0	

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

u = Bit is unchanged

x = Bit 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 **ZCDIE:** Zero-Cross Detection (ZCD) Interrupt Enable bit

1 = Enables the ZCD interrupt

0 = Disables the ZCD interrupt

bit 5-2 **Unimplemented:** Read as '0'

bit 1 **C2IE:** Comparator C2 Interrupt Enable bit

1 = Enables the Comparator C2 interrupt

0 = Disables the Comparator C2 interrupt

bit 0 **C1IE:** Comparator C1 Interrupt Enable bit

1 = Enables the Comparator C1 interrupt

0 = Disables the Comparator C1 interrupt

**Note:** Bit PEIE of the INTCON register must be set to enable any peripheral interrupt controlled by registers PIE1-PIE7.



## REGISTER 10-5: 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
RC2IE	TX2IE	RC1IE	TX1IE	BCL2IE	SSP2IE	BCL1IE	SSP1IE
bit 7						bit 0	

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

u = Bit is unchanged

x = Bit is unknown

-n/n = Value at POR and BOR/Value at all other Resets

'1' = Bit is set

'0' = Bit is cleared

- bit 7      **RC2IE:** USART2 Receive Interrupt Enable bit  
           1 = Enables the USART2 receive interrupt  
           0 = Enables the USART2 receive interrupt
- bit 6      **TX2IE:** USART2 Transmit Interrupt Enable bit  
           1 = Enables the USART2 transmit interrupt  
           0 = Disables the USART2 transmit interrupt
- bit 5      **RC1IE:** USART1 Receive Interrupt Enable bit  
           1 = Enables the USART1 receive interrupt  
           0 = Enables the USART1 receive interrupt
- bit 4      **TX1IE:** USART1 Transmit Interrupt Enable bit  
           1 = Enables the USART1 transmit interrupt  
           0 = Disables the USART1 transmit interrupt
- bit 3      **BCL2IE:** MSSP2 Bus Collision Interrupt Enable bit  
           1 = MSSP bus Collision interrupt enabled  
           0 = MSSP bus Collision interrupt disabled
- bit 2      **SSP2IE:** MSSP2 Interrupt Enable bit  
           1 = Enables the MSSP2 Interrupt  
           0 = Disables the MSSP Interrupt
- bit 1      **BCL1IE:** MSSP1 Bus Collision Interrupt Enable bit  
           1 = MSSP1 bus collision interrupt enabled  
           0 = MSSP1 bus collision interrupt disabled
- bit 0      **SSP1IE:** MSSP1 Interrupt Enable bit  
           1 = Enables the MSSP1 interrupt  
           0 = Disables the MSSP1 interrupt

**Note:** Bit PEIE of the INTCON register must be set to enable any peripheral interrupt controlled by PIE1-PIE7.

## REGISTER 10-6: PIE4: PERIPHERAL INTERRUPT ENABLE REGISTER 4

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0
—	—	—	—	—	—	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	HS = Hardware set

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

bit 1      **TMR2IE:** TMR2 to PR2 Match Interrupt Enable bit  
 1 = Enables the Timer2 to PR2 match interrupt  
 0 = Disables the Timer2 to PR2 match interrupt

bit 0      **TMR1IE:** Timer1 Overflow Interrupt Enable bit  
 1 = Enables the Timer1 overflow interrupt  
 0 = Enables the Timer1 overflow interrupt

**Note:** Bit PEIE of the INTCON register must be set to enable any peripheral interrupt controlled by registers PIE1-PIE7.

# PIC16(L)F15354/55

## REGISTER 10-7: PIE5: PERIPHERAL INTERRUPT ENABLE REGISTER 5

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	U-0	U-0	R/W-0/0
CLC4IE	CLC3IE	CLC2IE	CLC1IE	—	—	—	TMR1GIE
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

u = Bit is unchanged

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 = Hardware set

- bit 7      **CLC4IE:** CLC4 Interrupt Enable bit  
 1 = CLC4 interrupt enabled  
 0 = CLC4 interrupt disabled
- bit 6      **CLC3IE:** CLC3 Interrupt Enable bit  
 1 = CLC3 interrupt enabled  
 0 = CLC3 interrupt disabled
- bit 5      **CLC2IE:** CLC2 Interrupt Enable bit  
 1 = CLC2 interrupt enabled  
 0 = CLC2 interrupt disabled
- bit 4      **CLC1IE:** CLC1 Interrupt Enable bit  
 1 = CLC1 interrupt enabled  
 0 = CLC1 interrupt disabled
- bit 3-1    **Unimplemented:** Read as '0'
- bit 0      **TMR1GIE:** Timer1 Gate Interrupt Enable bit  
 1 = Enables the Timer1 gate acquisition interrupt  
 0 = Disables the Timer1 gate acquisition interrupt

**Note:** Bit PEIE of the INTCON register must be set to enable any peripheral interrupt controlled by registers PIE1-PIE7.

## REGISTER 10-8: PIE6: PERIPHERAL INTERRUPT ENABLE REGISTER 6

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0
—	—	—	—	—	—	CCP2IE	CCP1IE
bit 7						bit 0	

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

u = Bit is unchanged

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 = Hardware set

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

bit 1      **CCP2IE:** CCP2 Interrupt Enable bit

1 = CCP2 interrupt is enabled

0 = CCP2 interrupt is disabled

bit 0      **CCP1IE:** CCP1 Interrupt Enable bit

1 = CCP1 interrupt is enabled

0 = CCP1 interrupt is disabled

**Note:** Bit PEIE of the INTCON register must be set to enable any peripheral interrupt controlled by registers PIE1-PIE7.

## REGISTER 10-9: PIE7: PERIPHERAL INTERRUPT ENABLE REGISTER 7

U-0	U-0	R/W-0/0	R/W-0/0	U-0	U-0	U-0	R/W-0/0
—	—	NVMIE	NCO1IE	—	—	—	CWG1IE
bit 7						bit 0	

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

u = Bit is unchanged

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 = Hardware set

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

bit 5 **NVMIE:** NVM Interrupt Enable bit  
 1 = NVM task complete interrupt enabled  
 0 = NVM interrupt not enabled

bit 4 **NCO1IE:** NCO Interrupt Enable bit  
 1 = NCO rollover interrupt enabled  
 0 = NCO rollover interrupt disabled

bit 3-1 **Unimplemented:** Read as '0'.

bit 0 **CWG1IE:** CWG1 Interrupt Enable bit  
 1 = CWG1 interrupt is enabled  
 0 = CWG1 interrupt disabled

**Note:** Bit PEIE of the INTCON register must be set to enable any peripheral interrupt controlled by registers PIE1-PIE7.

## REGISTER 10-10: PIR0: PERIPHERAL INTERRUPT STATUS REGISTER 0

U-0	U-0	R/W/HS-0/0	R-0	U-0	U-0	U-0	R/W/HS-0/0
—	—	TMR0IF	IOCIF	—	—	—	INTF <sup>(1)</sup>
bit 7							bit 0

### Legend:

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

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

bit 5      **TMR0IF:** Timer0 Overflow Interrupt Flag bit  
 1 = Timer0 register has overflowed (must be cleared in software)  
 0 = Timer0 register did not overflow

bit 4      **IOCIF:** Interrupt-on-Change Interrupt Flag bit (read-only)<sup>(2)</sup>  
 1 = One or more of the IOCAF-IOCEF register bits are currently set, indicating an enabled edge was detected by the IOC module.  
 0 = None of the IOCAF-IOCEF register bits are currently set

bit 3-1      **Unimplemented:** Read as '0'

bit 0      **INTF:** INT External Interrupt Flag bit<sup>(1)</sup>  
 1 = The INT external interrupt occurred (must be cleared in software)  
 0 = The INT external interrupt did not occur

- Note 1:** The External Interrupt INT pin is selected by INTPPS ([Register 15-1](#)).
- 2:** The IOCIF bit is the logical OR of all the IOCAF-IOCEF flags. Therefore, to clear the IOCIF flag, application firmware must clear all of the lower level IOCAF-IOCEF register bits.

**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 10-11: PIR1: PERIPHERAL INTERRUPT REQUEST REGISTER 1

R/W/HS-0/0	R/W/HS-0/0	U-0	U-0	U-0	U-0	U-0	R/W/HS-0/0
OSFIF	CSWIF	—	—	—	—	—	ADIF
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

u = Bit is unchanged

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 = Hardware set

- bit 7      **OSFIF:** Oscillator Fail-Safe Interrupt Flag bit  
 1 = Oscillator fail-safe interrupt has occurred (must be cleared in software)  
 0 = No oscillator fail-safe interrupt
- bit 6      **CSWIF:** Clock Switch Complete Interrupt Flag bit  
 1 = The clock switch module indicates an interrupt condition and is ready to complete the clock switch operation (must be cleared in software)  
 0 = The clock switch does not indicate an interrupt condition
- bit 5-1    **Unimplemented:** Read as '0'
- bit 0      **ADIF:** Analog-to-Digital Converter (ADC) Interrupt Flag bit  
 1 = An A/D conversion or complex operation has completed (must be cleared in software)  
 0 = An A/D conversion or complex operation is not complete

**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.

# PIC16(L)F15354/55

## REGISTER 10-12: PIR2: PERIPHERAL INTERRUPT REQUEST REGISTER 2

U-0	R/W/HS-0/0	U-0	U-0	U-0	U-0	R/W/HS-0/0	R/W/HS-0/0
—	ZCDIF	—	—	—	—	C2IF	C1IF
bit 7						bit 0	

### Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	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 = Hardware set

- bit 7      **Unimplemented:** Read as '0'
- bit 6      **ZCDIF:** Zero-Cross Detect (ZCD1) Interrupt Flag bit
  - 1 = An enabled rising and/or falling ZCD1 event has been detected (must be cleared in software)
  - 0 = No ZCD1 event has occurred
- bit 5-2    **Unimplemented:** Read as '0'
- bit 1      **C2IF:** Comparator C2 Interrupt Flag bit
  - 1 = Comparator 2 interrupt asserted (must be cleared in software)
  - 0 = Comparator 2 interrupt not asserted
- bit 0      **C1IF:** Comparator C1 Interrupt Flag bit
  - 1 = Comparator 1 interrupt asserted (must be cleared in software)
  - 0 = Comparator 1 interrupt not asserted

**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 10-13: PIR3: PERIPHERAL INTERRUPT REQUEST REGISTER 3

R/HS-0/0	R/HS-0/0	R/HS-0/0	R/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0
RC2IF	TX2IF	RC1IF	TX1IF	BCL2IF	SSP2IF	BCL1IF	SSP1IF
bit 7							bit 0

### Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	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 = Hardware clearable

- bit 7      **RC2IF:** EUSART2 Receive Interrupt Flag bit<sup>(1)</sup>  
 1 = The EUSART2 receive buffer is not empty (contains at least one byte)  
 0 = The EUSART2 receive buffer is empty
- bit 6      **TX2IF:** EUSART2 Transmit Interrupt Flag bit<sup>(2)</sup>  
 1 = The EUSART2 transmit buffer contains at least one unoccupied space  
 0 = The EUSART2 transmit buffer is currently full. The application firmware should not write to TXxREG
- bit 5      **RC1IF:** EUSART1 Receive Interrupt Flag bit <sup>(1)</sup>  
 1 = The EUSART1 receive buffer is not empty (contains at least one byte)  
 0 = The EUSART1 receive buffer is empty
- bit 4      **TX1IF:** EUSART1 Transmit Interrupt Flag bit<sup>(2)</sup>  
 1 = The EUSART1 transmit buffer contains at least one unoccupied space  
 0 = The EUSART1 transmit buffer is currently full. The application firmware should not write to TXxREG.
- bit 3      **BCL2IF:** MSSP2 Bus Collision Interrupt Flag bit  
 1 = A bus collision was detected (must be cleared in software)  
 0 = No bus collision was detected
- bit 2      **SSP2IF:** MSSP2 Interrupt Flag bit  
 1 = The Transmission/Reception/Bus Condition is complete (must be cleared in software)  
 0 = Waiting for the Transmission/Reception/Bus Condition in progress
- bit 1      **BCL1IF:** MSSP1 Bus Collision Interrupt Flag bit  
 1 = A bus collision was detected (must be cleared in software)  
 0 = No bus collision was detected
- bit 0      **SSP1IF:** MSSP1 Interrupt Flag bit  
 1 = The Transmission/Reception/Bus Condition is complete (must be cleared in software)  
 0 = Waiting for the Transmission/Reception/Bus Condition in progress

- Note 1:** The RCxIF flag is a read-only bit. To clear the RCxIF flag, the firmware must read from RCxREG enough times to remove all bytes from the receive buffer.
- 2:** The TXxIF flag is a read-only bit, indicating if there is room in the transmit buffer. To clear the TX1IF flag, the firmware must write enough data to TXxREG to completely fill all available bytes in the buffer. The TXxIF flag does not indicate transmit completion (use TRMT for this purpose instead).

**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.

# PIC16(L)F15354/55

## REGISTER 10-14: PIR4: PERIPHERAL INTERRUPT REQUEST REGISTER 4

U-0	U-0	U-0	U-0	U-0	U-0	R/W/HS-0/0	R/W/HS-0/0
—	—	—	—	—	—	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

HS = Hardware set

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

bit 1      **TRM2IF:** Timer2 Interrupt Flag bit

1 = The TMR2 postscaler overflowed, or in 1:1 mode, a TMR2 to PR2 match occurred (must be cleared in software)

0 = No TMR2 event has occurred

bit 0      **TRM1IF:** Timer1 Overflow Interrupt Flag bit

1 = Timer1 overflow occurred (must be cleared in software)

0 = No Timer1 overflow occurred

**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 10-15: PIR5: PERIPHERAL INTERRUPT REQUEST REGISTER 5

R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	U-0	U-0	U-0	R/W/HS-0/0
CLC4IF	CLC3IF	CLC2IF	CLC1IF	—	—	—	TMR1GIF
bit 7							bit 0

### Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	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 = Hardware set

- bit 7      **CLC4IF:** CLC4 Interrupt Flag bit  
 1 = A CLC4OUT interrupt condition has occurred (must be cleared in software)  
 0 = No CLC4 interrupt event has occurred
- bit 6      **CLC3IF:** CLC3 Interrupt Flag bit  
 1 = A CLC3OUT interrupt condition has occurred (must be cleared in software)  
 0 = No CLC3 interrupt event has occurred
- bit 5      **CLC2IF:** CLC2 Interrupt Flag bit  
 1 = A CLC2OUT interrupt condition has occurred (must be cleared in software)  
 0 = No CLC2 interrupt event has occurred
- bit 4      **CLC1IF:** CLC1 Interrupt Flag bit  
 1 = A CLC1OUT interrupt condition has occurred (must be cleared in software)  
 0 = No CLC1 interrupt event has occurred
- bit 3-1    **Unimplemented:** Read as '0'
- bit 0      **TMR1GIF:** Timer1 Gate Interrupt Flag bit  
 1 = The Timer1 Gate has gone inactive (the acquisition is complete)  
 0 = The Timer1 Gate has not gone inactive

**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.

# PIC16(L)F15354/55

## REGISTER 10-16: PIR6: PERIPHERAL INTERRUPT REQUEST REGISTER 6

U-0	U-0	U-0	U-0	U-0	U-0	R/W/HS-0/0	R/W/HS-0/0
—	—	—	—	—	—	CCP2IF	CCP1IF
bit 7						bit 0	

### Legend:

R = Readable bit                      W = Writable bit                      U = Unimplemented bit, read as '0'  
u = Bit is unchanged                      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 = Hardware set

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

bit 1                      **CCP2IF:** CCP2 Interrupt Flag bit

Value	CCPM Mode		
	Capture	Compare	PWM
1	Capture occurred (must be cleared in software)	Compare match occurred (must be cleared in software)	Output trailing edge occurred (must be cleared in software)
0	Capture did not occur	Compare match did not occur	Output trailing edge did not occur

bit 0                      **CCP1IF:** CCP1 Interrupt Flag bit

Value	CCPM Mode		
	Capture	Compare	PWM
1	Capture occurred (must be cleared in software)	Compare match occurred (must be cleared in software)	Output trailing edge occurred (must be cleared in software)
0	Capture did not occur	Compare match did not occur	Output trailing edge did not occur

**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 10-17: PIR7: PERIPHERAL INTERRUPT REQUEST REGISTER 7

U-0	U-0	R/W/HS-0/0	R/W/HS-0/0	U-0	U-0	U-0	R/W/HS-0/0
—	—	NVMIF	NCO1IF	—	—	—	CWG1IF
bit 7							bit 0

### Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	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 = Hardware set

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

bit 5 **NVMIF:** Nonvolatile Memory (NVM) Interrupt Flag bit  
 1 = The requested NVM operation has completed  
 0 = NVM interrupt not asserted

bit 4 **NCO1IF:** Numerically Controlled Oscillator (NCO) Interrupt Flag bit  
 1 = The NCO has rolled over  
 0 = No NCO interrupt event has occurred

bit 3-1 **Unimplemented:** Read as '0'

bit 0 **CWG1IF:** CWG1 Interrupt Flag bit  
 1 = CWG1 has gone into shutdown  
 0 = CWG1 is operating normally, or interrupt cleared

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

**TABLE 10-1: 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	—	—	—	—	—	INTEDG	121
PIE0	—	—	TMR0IE	IOCIE	—	—	—	INTE	122
PIE1	OSFIE	CSWIE	—	—	—	—	—	ADIE	123
PIE2	—	ZCDIE	—	—	—	—	C2IE	C1IE	124
PIE3	RC2IE	TX2IE	RC1IE	TX1IE	BCL2IE	SSP2IE	BCL1IE	SSP1IE	125
PIE4	—	—	—	—	—	—	TMR2IE	TMR1IE	126
PIE5	CLC4IE	CLC3IE	CLC2IE	CLC1IE	—	—	—	TMR1GIE	127
PIE6	—	—	—	—	—	—	CCP2IE	CCP1IE	128
PIE7	—	—	NVMIE	NCO1IE	—	—	—	CWG1IE	129
PIR0	—	—	TMR0IF	IOCIF	—	—	—	INTF	130
PIR1	OSFIF	CSWIF	—	—	—	—	—	ADIF	131
PIR2	—	ZCDIF	—	—	—	—	C2IF	C1IF	132
PIR3	RC2IF	TX2IF	RC1IF	TX1IF	BCL2IF	SSP2IF	BCL1IF	SSP1IF	133
PIR4	—	—	—	—	—	—	TMR2IF	TMR1IF	134
PIR5	CLC4IF	CLC3IF	CLC2IF	CLC1IF	—	—	—	TMR1GIF	135
PIR6	—	—	—	—	—	—	CCP2IF	CCP1IF	136
PIR7	—	—	NVMIF	NCO1IF	—	—	—	CWG1IF	137

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

## 11.0 POWER-SAVING OPERATION MODES

The purpose of the Power-Down modes is to reduce power consumption. There are three Power-Down modes: DOZE mode, IDLE mode, and SLEEP mode.

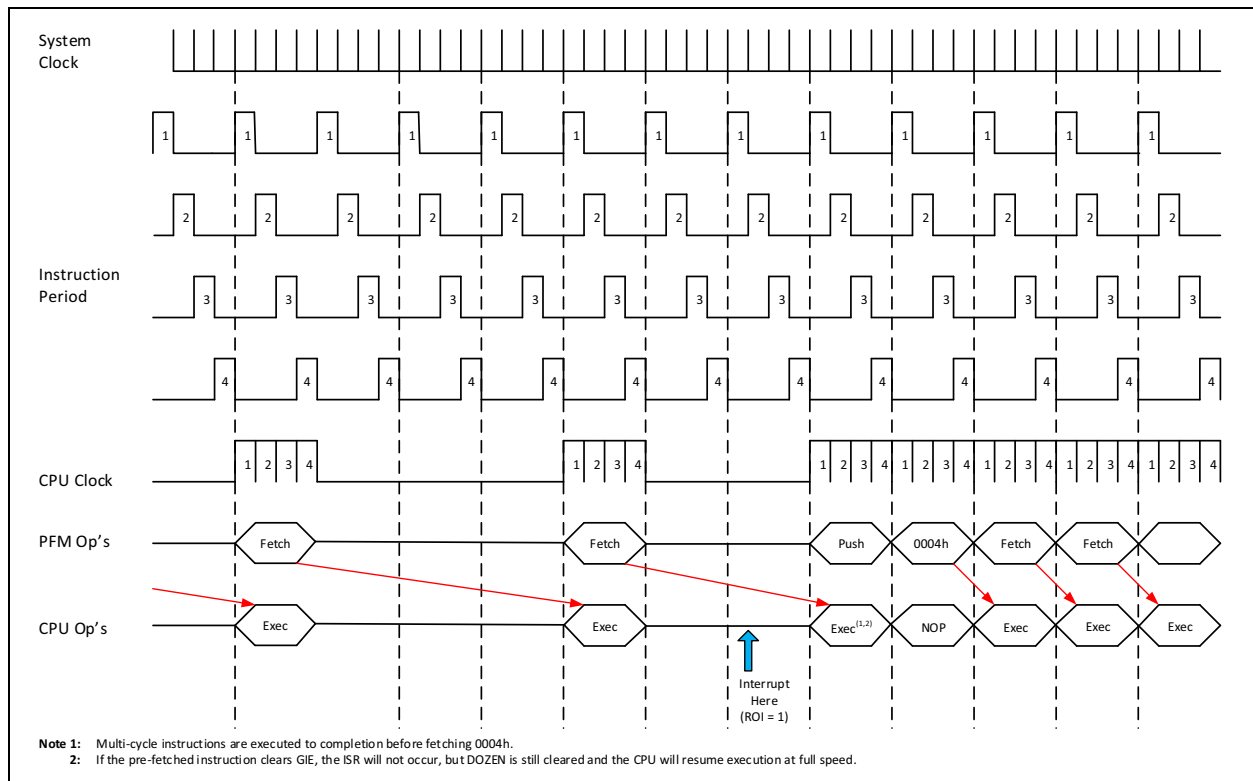
### 11.1 DOZE Mode

DOZE mode saves power by reducing CPU execution and program memory access, without affecting peripheral operation. DOZE mode differs from Sleep mode because the system oscillators continue to operate, while only the CPU and program memory are

affected. The reduced execution saves power by eliminating unnecessary operations within the CPU and memory.

When the Doze Enable (DOZEN) bit is set (DOZEN = 1), the CPU executes only one instruction cycle out of every N cycles as defined by the DOZE<2:0> bits of the CPUDOZE register. For example, if DOZE<2:0> = 100, the instruction cycle ratio is 1:32. The CPU and memory execute for one instruction cycle and then lay idle for 31 instruction cycles. During the unused cycles, the peripherals continue to operate at the system clock speed.

**FIGURE 11-1: DOZE MODE OPERATION EXAMPLE**



#### 11.1.1 DOZE OPERATION

The Doze operation is illustrated in Figure 11-1. For example, if ROI = 1 and DOZE<2:0> = '001', the instruction cycle ratio is 1:4. The CPU and memory operate for one instruction cycle and stay idle for the next three instruction cycles.

As with normal operation, the program memory fetches for the next instruction cycle. The system clock to the peripherals continue throughout.

## 11.1.2 INTERRUPTS DURING DOZE

If an interrupt occurs during DOZE, system behavior can be configured using the Recover-On-Interrupt (ROI) bit and the Doze-On-Exit (DOE) bit. Refer to [Table 11-1](#) for details about system behavior in all cases for a transition from Main to ISR back to Main.

**TABLE 11-1: INTERRUPTS DURING DOZE**

DOZEN	ROI	Code Flow			
		Main	ISR <sup>(1)</sup>	Return to Main	
0	0	Normal operation	Normal operation and DOE = DOZEN (in hardware) DOZEN = 0 (unchanged)		
0	1	Normal operation	Normal operation and DOE = DOZEN (in hardware) DOZEN = 0 (unchanged)	If DOE = 1 when return from interrupt: DOZE operation and DOZEN = 1 (in hardware)	If DOE = 0 when return from interrupt: Normal operation and DOZEN = 0 (in hardware)
1	0	DOZE operation	DOZE operation and DOE = DOZEN (in hardware) DOZEN = 1 (unchanged)		
1	1	DOZE operation	Normal operation and DOE = DOZEN (in hardware) DOZEN = 0 (unchanged)		

**Note 1:** User software can change the DOE bit in the ISR.

## 11.2 Sleep Mode

Sleep mode is entered by executing the `SLEEP` instruction, while the Idle Enable (IDLEN) bit of the `CPUDOZE` register is clear (IDLEN = 0). If the `SLEEP` instruction is executed while the IDLEN bit is set (IDLEN = 1), the CPU will enter the IDLE mode ([Section 11.2.3 “Low-Power Sleep Mode”](#)).

Upon entering Sleep mode, the following conditions exist:

1. WDT will be cleared but keeps running if enabled for operation during Sleep
2. The  $\overline{PD}$  bit of the STATUS register is cleared
3. The  $\overline{TO}$  bit of the STATUS register is set
4. CPU Clock and System Clock
5. 31 kHz LFINTOSC, HFINTOSC and SOSC are unaffected and peripherals using them may continue operation in Sleep.
6. ADC is unaffected if the dedicated FRC oscillator is selected the conversion will be left abandoned if FOSC is selected and ADRES will have an incorrect value
7. I/O ports maintain the status they had before Sleep was executed (driving high, low, or high-impedance). This does not apply in the case of any asynchronous peripheral which is active and may affect the I/O port value
8. 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 any 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.

Any module with a clock source that is not FOSC can be enabled. Examples of internal circuitry that might be sourcing current include modules such as the DAC and FVR modules. See [Section 21.0 “5-Bit Digital-to-Analog Converter \(DAC1\) Module”](#), [Section 18.0 “Fixed Voltage Reference \(FVR\)”](#) for more information on these modules.



## 11.2.1 WAKE-UP FROM SLEEP

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

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

The first three events will cause a device Reset. The last three events are considered a continuation of program execution. To determine whether a device Reset or wake-up event occurred, refer to [Section 8.12 “Memory Execution Violation”](#).

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.

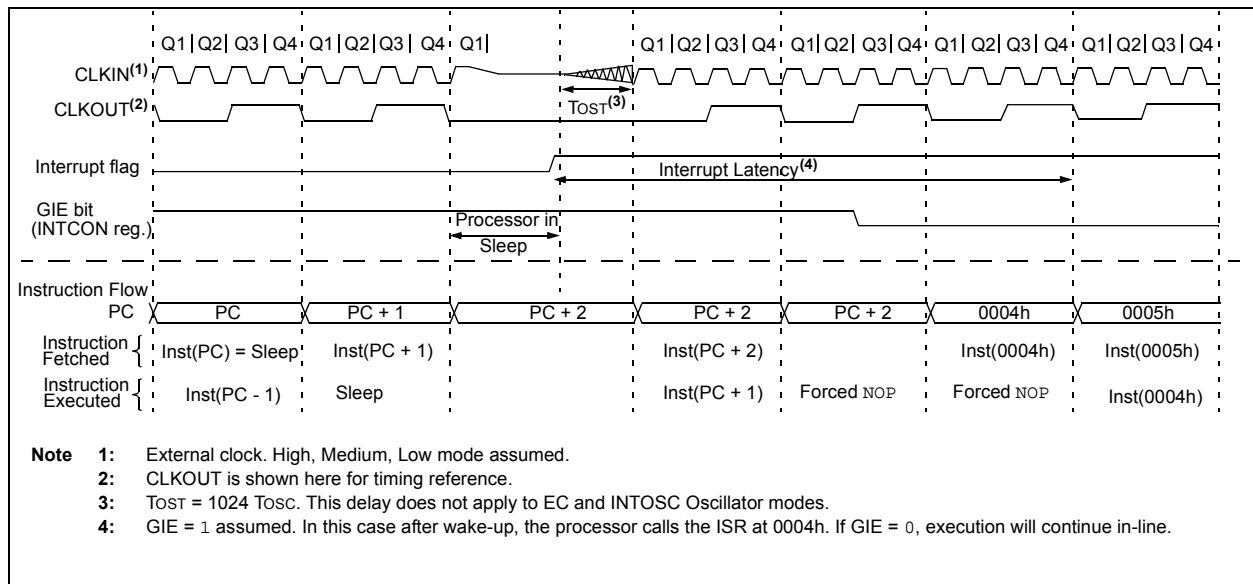
## 11.2.2 WAKE-UP USING INTERRUPTS

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

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

Even if the flag bits were checked before executing a `SLEEP` instruction, it may be possible for flag bits to become set before the `SLEEP` instruction completes. To determine whether a `SLEEP` instruction executed, test

**FIGURE 11-2: WAKE-UP FROM SLEEP THROUGH INTERRUPT**



## 11.2.3 LOW-POWER SLEEP MODE

The PIC16F15354/55 device contains an internal Low Dropout (LDO) voltage regulator, which allows the device I/O pins to operate at voltages up to 5.5V while the internal device logic operates at a lower voltage. The LDO and its associated reference circuitry must remain active when the device is in Sleep mode.

The PIC16F15354/55 allows the user to optimize the operating current in Sleep, depending on the application requirements.

Low-Power Sleep mode can be selected by setting the VREGPM bit of the VREGCON register. Depending on the configuration of these bits, the LDO and reference circuitry are placed in a low-power state when the device is in Sleep.

### 11.2.3.1 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.

### 11.2.3.2 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 these peripherals:

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

It is the responsibility of the end user to determine what is acceptable for their application when setting the VREGPM settings in order to ensure operation in Sleep.

**Note:** The PIC16LF15354/55 does not have a configurable Low-Power Sleep mode. PIC16LF15354/55 is an unregulated device and is always in the lowest power state when in Sleep, with no wake-up time penalty. This device has a lower maximum VDD and I/O voltage than the PIC16F15354/55. See [Section 37.0 “Electrical Specifications”](#) for more information.

## 11.3 IDLE Mode

When the Idle Enable (IDLEN) bit is clear (IDLEN = 0), the SLEEP instruction will put the device into full Sleep mode (see [Section 11.2 “Sleep Mode”](#)). When IDLEN is set (IDLEN = 1), the SLEEP instruction will put the device into IDLE mode. In IDLE mode, the CPU and memory operations are halted, but the peripheral clocks continue to run. This mode is similar to DOZE mode, except that in IDLE both the CPU and program memory are shut off.

**Note:** Peripherals using FOSC will continue running while in Idle (but not in Sleep). Peripherals using HFINTOSC, LFINTOSC, or SOSC will continue running in both Idle and Sleep.

**Note:** If CLKOUT is enabled (CLKOUT = 0, Configuration Word 1), the output will continue operating while in Idle.

### 11.3.1 IDLE AND INTERRUPTS

IDLE mode ends when an interrupt occurs (even if GIE = 0), but IDLEN is not changed. The device can re-enter IDLE by executing the SLEEP instruction.

If Recover-on-Interrupt is enabled (ROI = 1), the interrupt that brings the device out of Idle also restores full-speed CPU execution when doze is also enabled.

### 11.3.2 IDLE AND WDT

When in IDLE, the WDT Reset is blocked and will instead wake the device. The WDT wake-up is not an interrupt, therefore ROI does not apply.

**Note:** The WDT can bring the device out of IDLE, in the same way it brings the device out of Sleep. The DOZEN bit is not affected.

## 11.4 Register Definitions: Voltage Regulator and DOZE Control

### REGISTER 11-1: VREGCON: VOLTAGE REGULATOR CONTROL REGISTER<sup>(1)</sup>

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

#### Legend:

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

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

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

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

**Note 1:** PIC16F15354/55 only.

**2:** See [Section 37.0 "Electrical Specifications"](#).

## REGISTER 11-2: CPUDOZE: DOZE AND IDLE REGISTER

R/W-0/0	R/W/HC/HS-0/0	R/W-0/0	R/W/HS/HC-0/0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
IDLEN	DOZEN <sup>(1)</sup>	ROI <sup>(1)</sup>	DOE <sup>(1)</sup>	—	DOZE<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
HC = Bit is cleared by hardware		HS = Bit is set by hardware
'1' = Bit is set	'0' = Bit is cleared	

bit 7	<b>IDLEN:</b> Idle Enable bit 1 = A <i>SLEEP</i> instruction places the device into IDLE mode 0 = A <i>SLEEP</i> instruction places the device into Sleep mode
bit 6	<b>DOZEN:</b> Doze Enable bit 1 = Places the device into DOZE mode 0 = Places the device into Normal mode
bit 5	<b>ROI:</b> Recover-on-Interrupt bit 1 = Entering the Interrupt Service Routine (ISR) makes DOZEN = 0 0 = Entering the Interrupt Service Routine (ISR) does not change DOZEN
bit 4	<b>DOE:</b> Doze on Exit bit 1 = Exiting the ISR makes DOZEN = 1 0 = Exiting the ISR does not change DOZEN
bit 3	<b>Unimplemented:</b> Read as '0'
bit 2-0	<b>DOZE&lt;2:0&gt;:</b> Ratio of CPU Instruction Cycles to Peripheral Instruction Cycles 111 =1:256 110 =1:128 101 =1:64 100 =1:32 011 =1:16 010 =1:8 001 =1:4 000 =1:2

**Note 1:** Refer to [Table 11-1](#) for more details.

**TABLE 11-2: 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
STATUS	—	—	—	TO	PD	Z	DC	C	<a href="#">30</a>
VREGCON	—	—	—	—	—	—	VREGPM	—	<a href="#">143</a>
CPUDOZE	IDLEN	DOZEN	ROI	DOE	—	DOZE<2:0>			<a href="#">144</a>

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

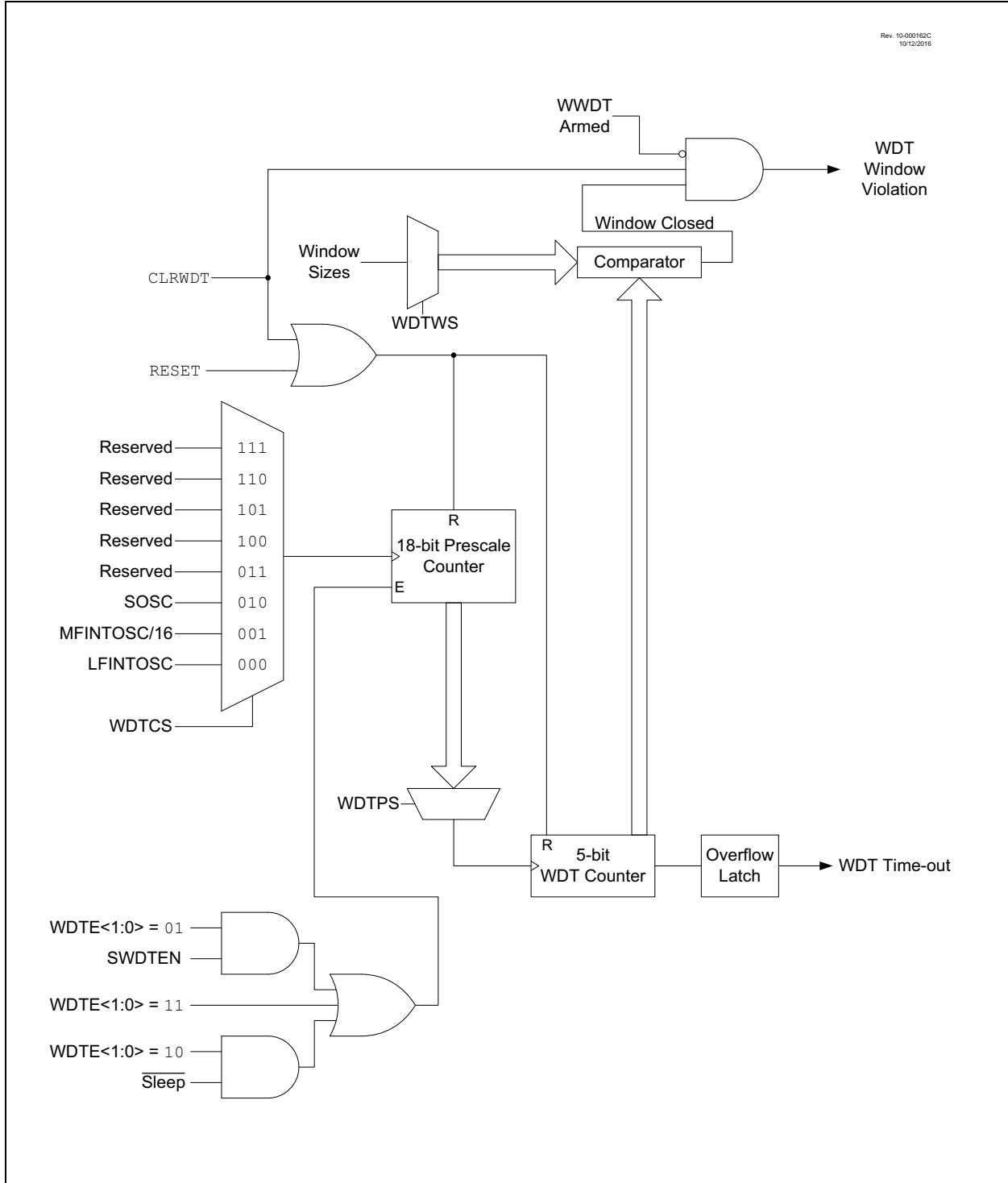
## 12.0 WINDOWED WATCHDOG TIMER (WWDT)

The Watchdog Timer (WDT) 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 Windowed Watchdog Timer (WWDT) differs in that CLRWDT instructions are only accepted when they are performed within a specific window during the time-out period.

The WDT has the following features:

- Selectable 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)
- Configurable window size from 12.5 to 100 percent of the time-out period
- Multiple Reset conditions
- Operation during Sleep

**FIGURE 12-1: WATCHDOG TIMER BLOCK DIAGRAM**



## 12.1 Independent Clock Source

The WDT can derive its time base from either the 31 kHz LFINTOSC or 31.25 kHz MFINTOSC internal oscillators, or the secondary oscillator SOSC, depending on the value of either the WDTCCS<2:0> Configuration bits or the WDTCS<2:0> bits of WDTCON1. Time intervals in this chapter are based on a minimum nominal interval of 1 ms. See [Section 37.0 “Electrical Specifications”](#) for LFINTOSC and MFINTOSC tolerances.

## 12.2 WDT Operating Modes

The Watchdog Timer module has four operating modes controlled by the WDTE<1:0> bits in Configuration Words. See [Table 12-1](#).

### 12.2.1 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.

### 12.2.2 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.

### 12.2.3 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 WDTCON0 register.

### 12.2.4 WDT IS OFF

When the WDTE bits of the Configuration Word are set to ‘00’, the WDT is always OFF.

WDT protection is unchanged by Sleep. See [Table 12-1](#) for more details.

**TABLE 12-1: WDT OPERATING MODES**

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

## 12.3 Time-Out Period

The WDTPS bits of the WDTCON0 register set the time-out period from 1 ms to 256 seconds (nominal). After a Reset, the default time-out period is two seconds.



## 12.4 Watchdog Window

The Watchdog Timer has an optional Windowed mode that is controlled by the WDT\_CWS<2:0> Configuration bits and WINDOW<2:0> bits of the WDTCON1 register. In the Windowed mode, the CLRWDT instruction must occur within the allowed window of the WDT period. Any CLRWDT instruction that occurs outside of this window will trigger a window violation and will cause a WDT Reset, similar to a WDT time out. See Figure 12-2 for an example.

The window size is controlled by the WDT\_CWS<2:0> Configuration bits, or the WINDOW<2:0> bits of WDTCON1, if WDT\_CWS<2:0> = 111.

In the event of a window violation, a Reset will be generated and the WDTWV bit of the PCON register will be cleared. This bit is set by a POR or can be set in firmware.

## 12.5 Clearing the WDT

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

- Any Reset
- Valid CLRWDT instruction is executed
- Device enters Sleep
- Device wakes up from Sleep
- WDT is disabled

- Oscillator Start-up Timer (OST) is running
- Any write to the WDTCON0 or WDTCON1 registers

### 12.5.1 CLRWDT CONSIDERATIONS (WINDOWED MODE)

When in Windowed mode, the WDT must be armed before a CLRWDT instruction will clear the timer. This is performed by reading the WDTCON0 register. Executing a CLRWDT instruction without performing such an arming action will trigger a window violation.

See Table 12-2 for more information.

## 12.6 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.

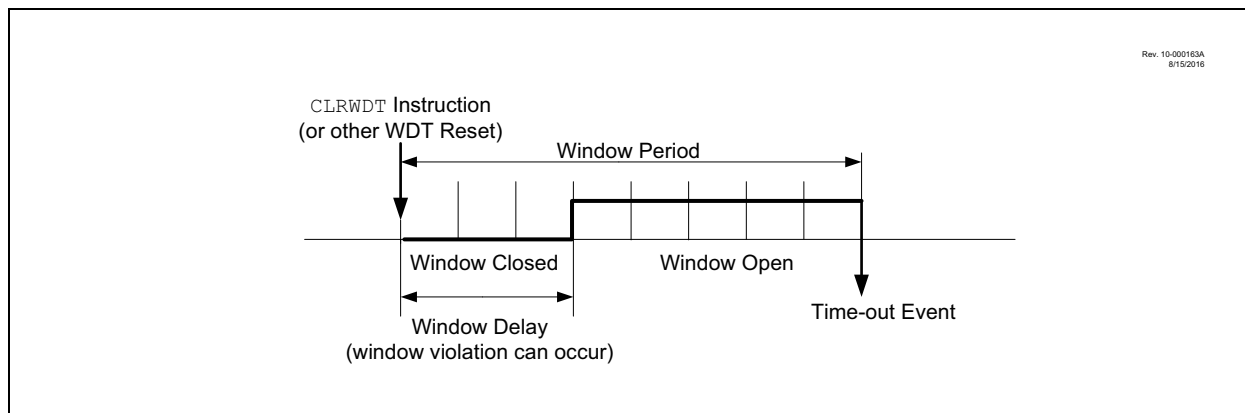
The WDT remains clear until the OST, if enabled, completes. See Section 9.0 “Oscillator Module (with Fail-Safe Clock Monitor)” for more information on the OST.

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. The RWDT bit in the PCON register can also be used. See Section 4.3.2.1 “STATUS Register” for more information.

**TABLE 12-2: 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 = SOSC, EXTOSC, INTOSC	
Change INTOSC divider (IRCF bits)	Unaffected

**FIGURE 12-2: WINDOW PERIOD AND DELAY**



## 12.7 Register Definitions: Windowed Watchdog Timer Control

**REGISTER 12-1: WDTCON0: WATCHDOG TIMER CONTROL REGISTER 0**

U-0	U-0	R/W <sup>(3)</sup> -q/q <sup>(2)</sup>	R/W <sup>(3)</sup> -q/q <sup>(2)</sup>	R/W <sup>(3)</sup> -q/q <sup>(2)</sup>	R/W <sup>(3)</sup> -q/q <sup>(2)</sup>	R/W <sup>(3)</sup> -q/q <sup>(2)</sup>	R/W-0/0
—	—	WDTPS<4:0> <sup>(1)</sup>					SWDTEN
bit 7							bit 0

**Legend:**

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

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

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

Bit Value = Prescale Rate

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

•  
•  
•

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

10010 =1:8388608 (2<sup>23</sup>) (Interval 256s nominal)

10001 =1:4194304 (2<sup>22</sup>) (Interval 128s nominal)

10000 =1:2097152 (2<sup>21</sup>) (Interval 64s nominal)

01111 =1:1048576 (2<sup>20</sup>) (Interval 32s nominal)

01110 =1:524288 (2<sup>19</sup>) (Interval 16s nominal)

01101 =1:262144 (2<sup>18</sup>) (Interval 8s nominal)

01100 =1:131072 (2<sup>17</sup>) (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.

**2:** When WDTCP5 <4:0> in CONFIG3 = 11111, the Reset value of WDTPS<4:0> is 01011. Otherwise, the Reset value of WDTPS<4:0> is equal to WDTCP5<4:0> in CONFIG3.

**3:** When WDTCP5 <4:0> in CONFIG3 ≠ 11111, these bits are read-only.

## REGISTER 12-2: WDTCON1: WATCHDOG TIMER CONTROL REGISTER 1

U-0	R/W <sup>(3)</sup> -q/q <sup>(1)</sup>	R/W <sup>(3)</sup> -q/q <sup>(1)</sup>	R/W <sup>(3)</sup> -q/q <sup>(1)</sup>	U-0	R/W <sup>(4)</sup> -q/q <sup>(2)</sup>	R/W <sup>(4)</sup> -q/q <sup>(2)</sup>	R/W <sup>(4)</sup> -q/q <sup>(2)</sup>
—	WDTCS<2:0>			—	WINDOW<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 **Unimplemented:** Read as '0'

bit 6-4 **WDTCS<2:0>:** Watchdog Timer Clock Select bits

111 =Reserved

•  
•  
•

010 =SOSC 32 kHz

001 =MFINTOSC 31.25 kHz

000 =LFINTOSC 31 kHz

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

bit 2-0 **WINDOW<2:0>:** Watchdog Timer Window Select bits

WINDOW<2:0>	Window delay Percent of time	Window opening Percent of time
111	N/A	100
110	12.5	87.5
101	25	75
100	37.5	62.5
011	50	50
010	62.5	37.5
001	75	25
000	87.5	12.5

**Note 1:** If WDTCCS <2:0> in CONFIG3 = 111, the Reset value of WDTCS<2:0> is 000.

**2:** The Reset value of WINDOW<2:0> is determined by the value of WDTCCS<2:0> in the CONFIG3 register.

**3:** If WDTCCS<2:0> in CONFIG3 ≠ 111, these bits are read-only.

**4:** If WDTCCS<2:0> in CONFIG3 ≠ 111, these bits are read-only.

## REGISTER 12-3: WDTPSL: WDT PRESCALE SELECT LOW BYTE REGISTER

R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0
PSCNT<7:0> <sup>(1)</sup>							
bit 7							bit 0

### Legend:

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

bit 7-0      **PSCNT<7:0>**: Prescale Select Low Byte bits<sup>(1)</sup>

**Note 1:** The 18-bit WDT prescale value, PSCNT<17:0> includes the WDTPSL, WDTPSH and the lower bits of the WDTTMR registers. PSCNT<17:0> is intended for debug operations and should be read during normal operation.

## REGISTER 12-4: WDTPSH: WDT PRESCALE SELECT HIGH BYTE REGISTER

R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0
PSCNT<15:8> <sup>(1)</sup>							
bit 7							bit 0

### Legend:

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

bit 7-0      **PSCNT<15:8>**: Prescale Select High Byte bits<sup>(1)</sup>

**Note 1:** The 18-bit WDT prescale value, PSCNT<17:0> includes the WDTPSL, WDTPSH and the lower bits of the WDTTMR registers. PSCNT<17:0> is intended for debug operations and should be read during normal operation.

## REGISTER 12-5: WDTTMR: WDT TIMER REGISTER

U-0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0
—	WDTTMR<3:0>				STATE	PSCNT<17:16> <sup>(1)</sup>	
bit 7							bit 0

### Legend:

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

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

bit 6-3      **WDTTMR<3:0>**: Watchdog Timer Value bits

bit 2      **STATE:** WDT Armed Status bit

1 = WDT is armed  
0 = WDT is not armed

bit 1-0      **PSCNT<17:16>**: Prescale Select Upper Byte bits<sup>(1)</sup>

**Note 1:** The 18-bit WDT prescale value, PSCNT<17:0> includes the WDTPSL, WDTPSH and the lower bits of the WDTTMR registers. PSCNT<17:0> is intended for debug operations and should be read during normal operation.

**TABLE 12-3: 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
OSCCON1	—	NOSC<2:0>			NDIV<3:0>				110
OSCCON2	—	COSC<2:0>			CDIV<3:0>				110
OSCCON3	CSWHOLD	SOSCPWR	—	ORDY	NOSCR	—	—	—	111
PCON0	STKOVF	STKUNF	WDTWV	RWDT	RMCLR	RI	POR	BOR	99
STATUS	—	—	—	TO	PD	Z	DC	C	30
WDTCON0	—	—	WDTPS<4:0>					SWDTEN	150
WDTCON1	—	WDTCS<2:0>			—	WINDOW<2:0>			151
WDTPSL	PSCNT<7:0>								152
WDTPSH	PSCNT<15:8>								152
WDTTMR	—	WDTTMR<4:0>				STATE	PSCNT<17:16>		152

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

**TABLE 12-4: 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	—	CSWEN	—	—	CLKOUTEN	76
	7:0	—	RSTOSC<2:0>			—	FEXTOSC<2:0>			

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

## 13.0 NONVOLATILE MEMORY (NVM) CONTROL

NVM consists of the Program Flash Memory.

NVM is accessible by using both the FSR and INDF registers, or through the NVMREG register interface.

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.

NVM can be protected in two ways; by either code protection or write protection.

Code protection ( $\overline{CP}$  bit in Configuration Word 5) disables access, reading and writing, to the 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 nonvolatile memory, Configuration bits, and User IDs.

Write protection prohibits self-write and erase to a portion or all of the program memory, as defined by the WRT<1:0> bits of Configuration Word 4. Write protection does not affect a device programmer's ability to read, write, or erase the device.

### 13.1 Program Flash Memory

The program memory consists of an array of 14-bit words as user memory, with additional words for User ID information, Configuration words, and interrupt vectors. The program memory provides storage locations for:

- User program instructions
- User defined data

Program memory data can be read and/or written to through:

- CPU instruction fetch (read-only)
- FSR/INDF indirect access (read-only) ([Section 13.2 “FSR and INDF Access”](#))
- NVMREG access ([Section 13.3 “NVMREG Access”](#))
- In-Circuit Serial Programming™ (ICSP™)

Read operations return a single word of memory. When write and erase operations are done on a row basis, the row size is defined in [Table 13-1](#). Program memory will erase to a logic '1' and program to a logic '0'.

**TABLE 13-1: FLASH MEMORY ORGANIZATION BY DEVICE**

Device	Row Erase (words)	Write Latches (words)	Total Program Flash (words)
PIC16(L)F15354	32	32	4096
PIC16(L)F15355			8192

It is important to understand the program memory structure for erase and programming operations. The program memory is arranged in rows. A row consists of 32 14-bit program memory words. A row is the minimum size that can be erased by user software.

All or a portion of a row can be programmed. Data to be written into the program memory row is written to 14-bit wide data write latches. These latches are not directly accessible, but may be loaded via sequential writes to the NVMDATH:NVMDATL register pair.

**Note:** To modify only a portion of a previously programmed row, the contents of the entire row must be read. Then, the new data and retained data can be written into the write latches to reprogram the row of 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

#### 13.1.1 PROGRAM MEMORY VOLTAGES

The program memory is readable and writable during normal operation over the full VDD range.

##### 13.1.1.1 Programming Externally

The program memory cell and control logic support write and Bulk Erase operations down to the minimum device operating voltage. Special BOR operation is enabled during Bulk Erase ([Section 8.2.4 “BOR is always OFF”](#)).

##### 13.1.1.2 Self-programming

The program memory cell and control logic will support write and row erase operations across the entire VDD range. Bulk Erase is not available when self-programming.

## 13.2 FSR and INDF Access

The FSR and INDF registers allow indirect access to the program memory.

### 13.2.1 FSR READ

With the intended address loaded into an FSR register a `MOVIW` instruction or read of INDF will read data from the program memory.

Reading from NVM requires one instruction cycle. The CPU operation is suspended during the read, and resumes immediately after. Read operations return a single byte of memory.

### 13.2.2 FSR WRITE

Writing/erasing the NVM through the FSR registers (ex. `MOVWI` instruction) is not supported in the PIC16(L)F15354/55 devices.

## 13.3 NVMREG Access

The NVMREG interface allows read/write access to all the locations accessible by FSRs, and also read/write access to the User ID locations, and read-only access to the device identification, revision, and Configuration data.

Writing or erasing of NVM via the NVMREG interface is prevented when the device is write-protected.

### 13.3.1 NVMREG READ OPERATION

To read a NVM location using the NVMREG interface, the user must:

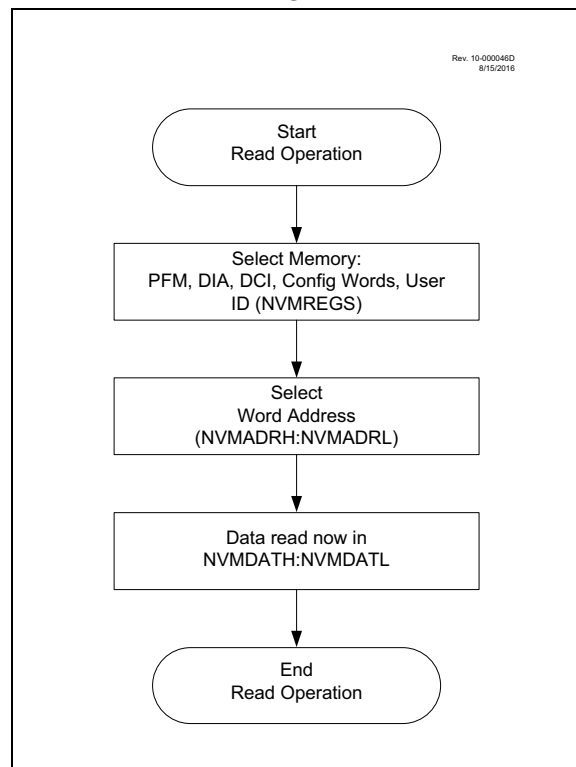
1. Clear the NVMREGS bit of the NVMCON1 register if the user intends to access the program memory locations, or set NVMREGS if the user intends to access User ID, or Configuration locations.
2. Write the desired address into the NVMADRH:NVMADRL register pair (Table 13-2).
3. Set the RD bit of the NVMCON1 register to initiate the read.

Once the read control bit is set, the CPU operation is suspended during the read, and resumes immediately after. The data is available in the very next cycle, in the NVMDATH:NVMDATL register pair; therefore, it can be read as two bytes in the following instructions.

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

Upon completion, the RD bit is cleared by hardware.

**FIGURE 13-1: FLASH PROGRAM MEMORY READ FLOWCHART**



## EXAMPLE 13-1: 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

BANKSELNVMADRL; Select Bank for NVMCON registers
MOVLWPROG_ADDR_LO;
MOVWFNVMADRL; Store LSB of address
MOVLWPROG_ADDR_HI;
MOVWFNVMADRH; Store MSB of address

BCF NVMCON1,NVMREGS; Do not select Configuration Space
BSF NVMCON1,RD; Initiate read

MOVFNVMDATL,W; Get LSB of word
MOVWFPROG_DATA_LO; Store in user location
MOVFNVMDATH,W; Get MSB of word
MOVWFPROG_DATA_HI; Store in user location
```



## 13.3.2 NVM UNLOCK SEQUENCE

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

- Program memory Row Erase
- Load of 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 and must be completed in order:

- Write 55h to NVMCON2
- Write AAh to NVMCON2
- Set the WR bit of NVMCON1

Once the WR bit is set, the processor will stall internal operations until the operation is complete and then resume with the next instruction.

**Note:** The two NOP instructions after setting the WR bit that were required in previous devices are not required for PIC16(L)F15354/55 devices. See [Figure 13-2](#).

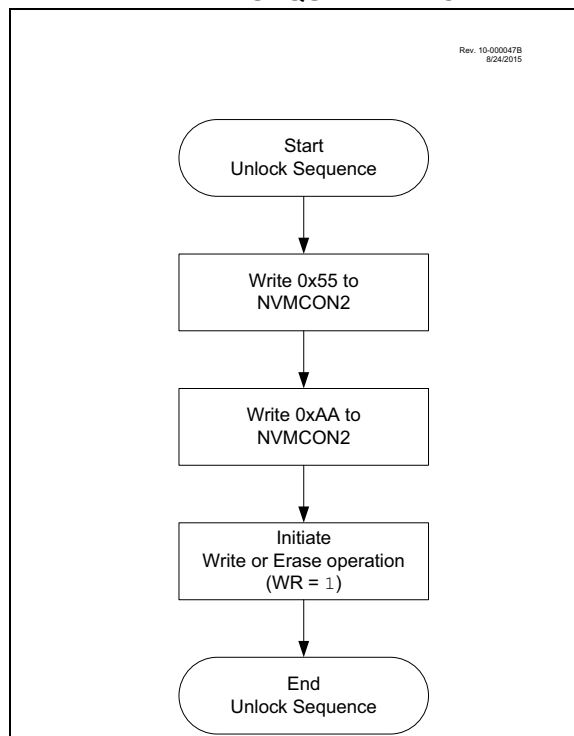
Since the unlock sequence must not be interrupted, global interrupts should be disabled prior to the unlock sequence and re-enabled after the unlock sequence is completed.

### EXAMPLE 13-2: NVM UNLOCK SEQUENCE

```
BCF INTCON, GIE; Recommended so sequence is not interrupted
BANKSELNVMCON1;
BSF NVMCON1, WREN; Enable write/erase
MOVLW55h; Load 55h
MOVWFNVMCON2; Step 1: Load 55h into NVMCON2
MOVLWAAh; Step 2: Load W with AAh
MOVWFNVMCON2; Step 3: Load AAH into NVMCON2
BSF NVMCON1, WR; Step 4: Set WR bit to begin write/erase
BSF INTCON, GIE; Re-enable interrupts
```

- Note 1:** Sequence begins when NVMCON2 is written; steps 1-4 must occur in the cycle-accurate order shown.  
**2:** Opcodes shown are illustrative; any instruction that has the indicated effect may be used.

**FIGURE 13-2: NVM UNLOCK SEQUENCE FLOWCHART**



## 13.3.3 NVMREG ERASE OF PROGRAM MEMORY

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

To erase a program memory row:

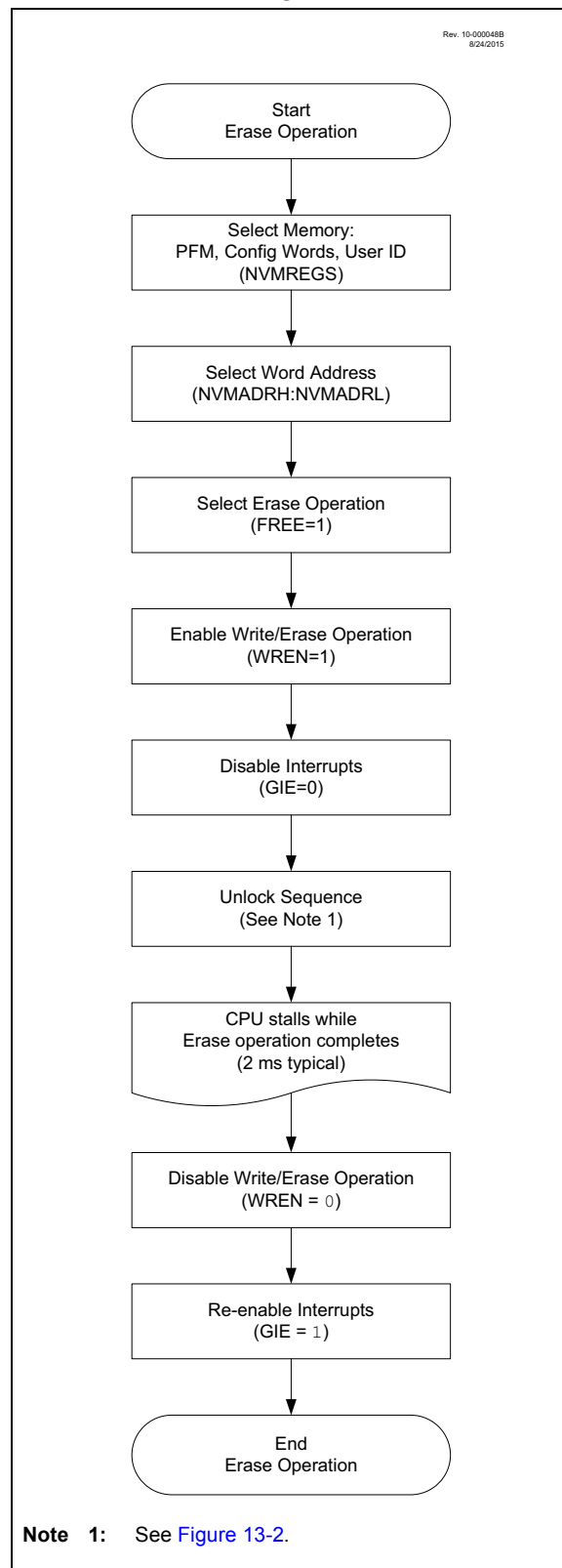
1. Clear the NVMREGS bit of the NVMCON1 register to erase program memory locations, or set the NMVREGS bit to erase User ID locations.
2. Write the desired address into the NVMADRH:NVMADRL register pair (Table 13-2).
3. Set the FREE and WREN bits of the NVMCON1 register.
4. Perform the unlock sequence as described in Section 13.3.2 “NVM Unlock Sequence”.

If the program memory address is write-protected, the WR bit will be cleared and the erase operation will not take place.

While erasing the program memory, CPU operation is suspended, and resumes when the operation is complete. Upon completion, the NVMIF is set, and an interrupt will occur if the NVMIE bit is also set.

Write latch data is not affected by erase operations, and WREN will remain unchanged.

**FIGURE 13-3: NVM ERASE FLOWCHART**



## EXAMPLE 13-3: ERASING ONE ROW OF PROGRAM FLASH MEMORY (PFM)

```

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

BANKSEL      NVMADRL
MOVF         ADDR, W
MOVWF       NVMADRL           ; Load lower 8 bits of erase address boundary
MOVF         ADDRH, W
MOVWF       NVMADRH         ; Load upper 6 bits of erase address boundary
BCF         NVMCON1, NVMREGS ; Choose PFM memory area
BSF         NVMCON1, FREE    ; Specify an erase operation
BSF         NVMCON1, WREN    ; Enable writes
BCF         INTCON, GIE      ; Disable interrupts during unlock sequence

; -----REQUIRED UNLOCK SEQUENCE:-----

MOVLW       55h              ; Load 55h to get ready for unlock sequence
MOVWF       NVMCON2         ; First step is to load 55h into NVMCON2
MOVLW       AAh              ; Second step is to load AAh into W
MOVWF       NVMCON2         ; Third step is to load AAh into NVMCON2
BSF         NVMCON1, WR     ; Final step is to set WR bit

; -----

BSF         INTCON, GIE      ; Re-enable interrupts, erase is complete
BCF         NVMCON1, WREN   ; Disable writes
    
```

**TABLE 13-2: NVM ORGANIZATION AND ACCESS INFORMATION**

Master Values			NVMREG Access			FSR Access	
Memory Function	Program Counter (PC), ICSP™ Address	Memory Type	NVMREGS bit (NVMCON1)	NVMADR<14:0>	Allowed Operations	FSR Address	FSR Programming Address
Reset Vector	0000h	Program Flash Memory	0	0000h	Read Write	8000h	Read-Only
User Memory	0001h		0	0001h		8001h	
	0003h		0	0003h		8003h	
INT Vector	0004h		0	0004h		8004h	
User Memory	0005h		0	0005h		8005h	
	0FFFh <sup>(1)</sup>			0FFFh		8FFFh	
	1FFFh <sup>(2)</sup>	1FFFh	9FFFh				
User ID	8000h	Program Flash Memory	1	0000h	Read Write	No Access	
	8003h			0003h			
Reserved	8004h	—	—	0004h	—		
Rev ID	8005h	Program Flash Memory	1	0005h	Read-Only		
Device ID	8006h		1	0006h	Read-Only		
CONFIG1	8007h		1	0007h	Read Write		
CONFIG2	8008h		1	0008h			
CONFIG3	8009h		1	0009h			
CONFIG4	800Ah		1	000Ah			
CONFIG5	800Bh		1	000Bh			
DIA and DCI	8100h-82FFh		Program Flash Memory and Hard coded	1	0100h-02FFh	Read-Only	

**Note 1:** PIC16(L)F15354 only.  
**Note 2:** PIC16(L)F15355 only.

## 13.3.4 NVMREG WRITE TO PROGRAM MEMORY

Program memory is programmed using the following steps:

1. Load the address of the row to be programmed into NVMADRH:NVMADRL.
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 13-4](#) (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 NVMADRH:NVMADRL, (NVMADRH<6:0>:NVMADRL<7:5>) with the lower five bits of NVMADRL, (NVMADRL<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 NVMDATH:NVMDATL 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.

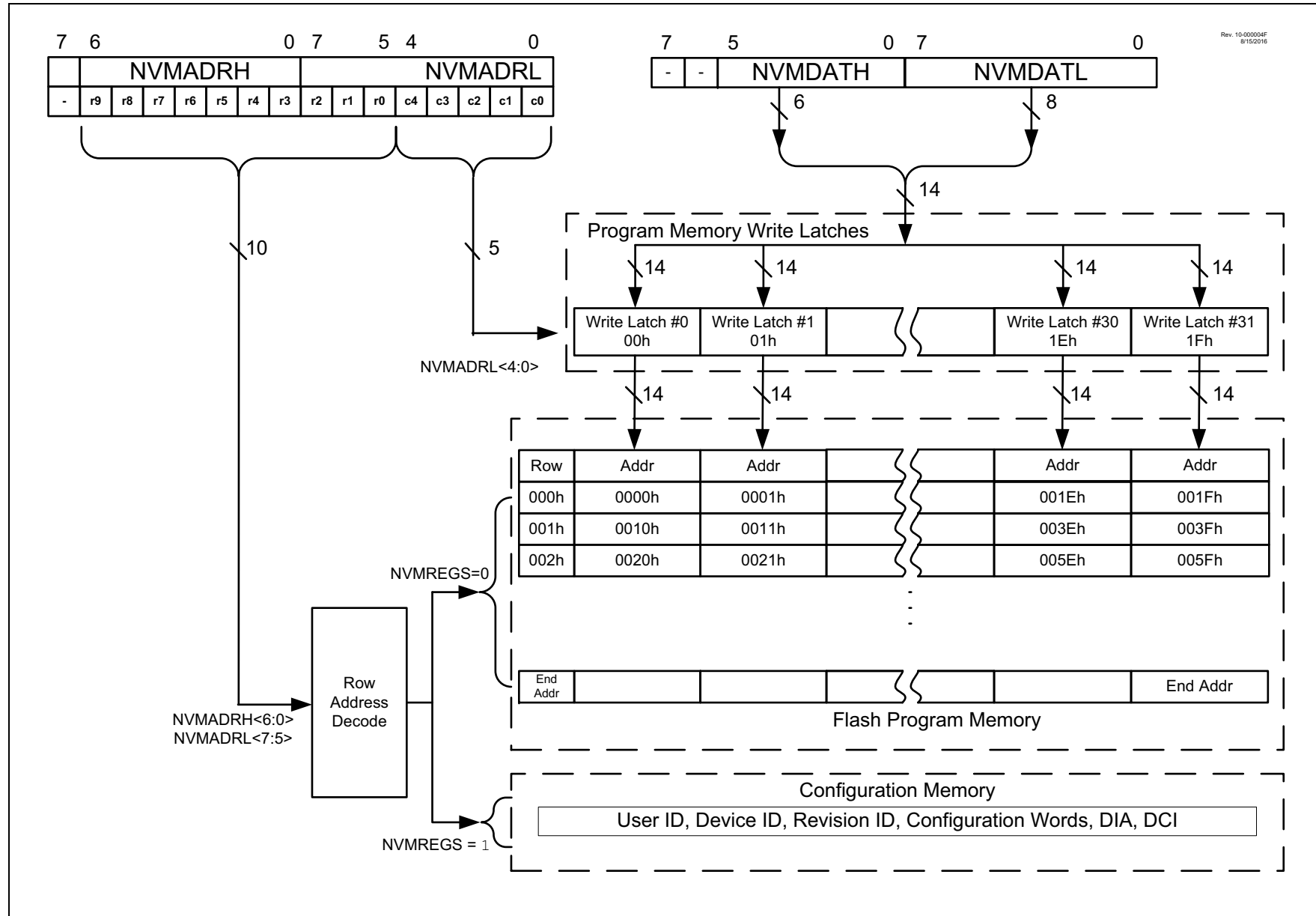
**Note:** The special unlock sequence is required to load a write latch with data or initiate a Flash programming operation. If the unlock sequence is interrupted, writing to the latches or program memory will not be initiated.

1. Set the WREN bit of the NVMCON1 register.
2. Clear the NVMREGS bit of the NVMCON1 register.
3. Set the LWLO bit of the NVMCON1 register. When the LWLO bit of the NVMCON1 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 NVMADRH:NVMADRL register pair with the address of the location to be written.
5. Load the NVMDATH:NVMDATL register pair with the program memory data to be written.
6. Execute the unlock sequence ([Section 13.3.2 "NVM Unlock Sequence"](#)). The write latch is now loaded.
7. Increment the NVMADRH:NVMADRL 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 NVMCON1 register. When the LWLO bit of the NVMCON1 register is '0', the write sequence will initiate the write to Flash program memory.
10. Load the NVMDATH:NVMDATL register pair with the program memory data to be written.
11. Execute the unlock sequence ([Section 13.3.2 "NVM Unlock Sequence"](#)). The entire program memory latch content is now written to Flash program memory.

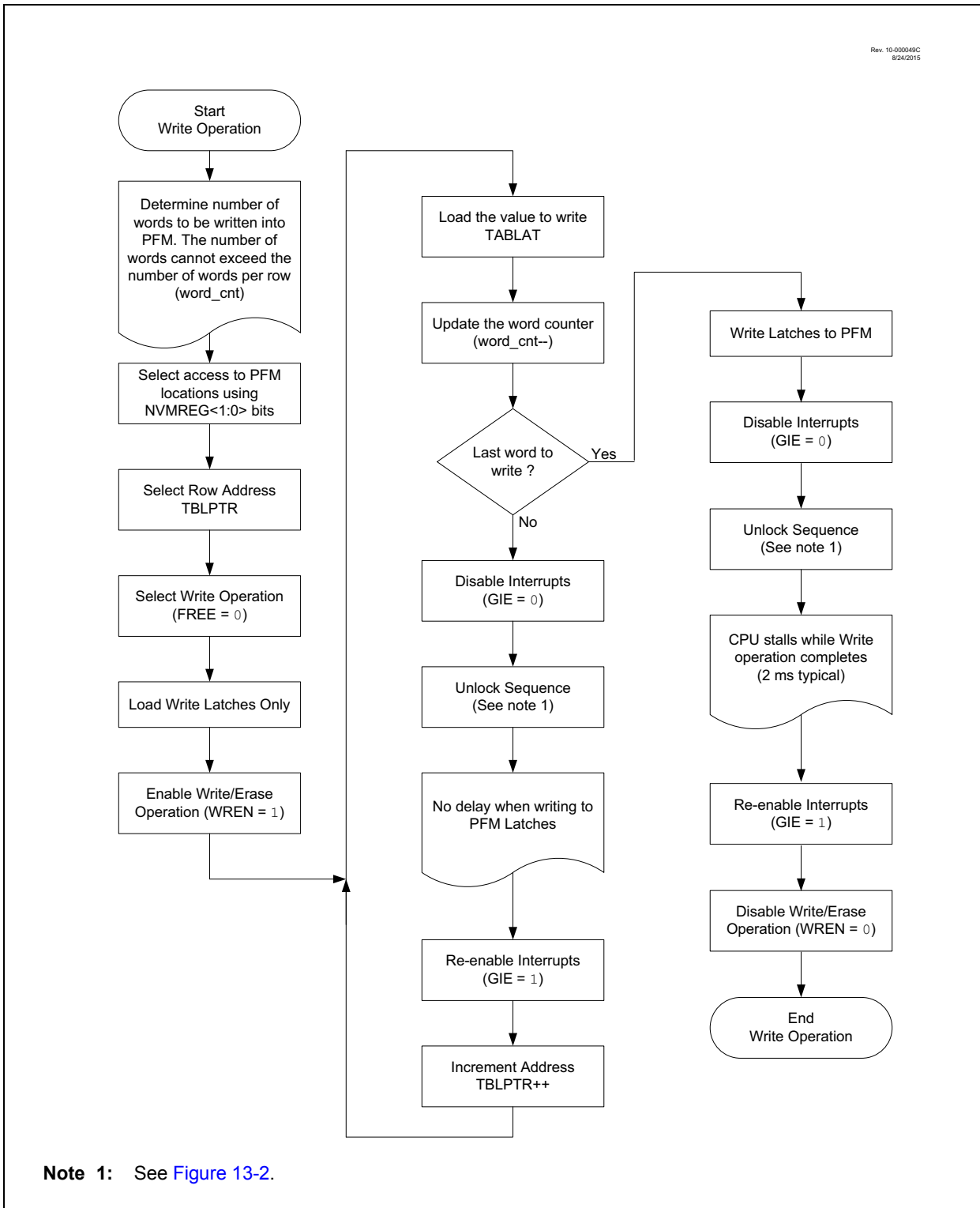
**Note:** The program memory write latches are reset to the blank state (0x3FFF) at the completion of every write or erase operation. As a result, it is not necessary to load all the program memory write latches. Unloaded latches will remain in the blank state.

An example of the complete write sequence is shown in [Example 13-4](#). The initial address is loaded into the NVMADRH:NVMADRL register pair; the data is loaded using indirect addressing.

**FIGURE 13-4: NVMREGS WRITES TO PROGRAM FLASH MEMORY WITH 32 WRITE LATCHES**



**FIGURE 13-5: PROGRAM FLASH MEMORY WRITE FLOWCHART**



## EXAMPLE 13-4: WRITING TO PROGRAM FLASH 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 common RAM (locations 0x70 - 0x7F)
; 5. NVM interrupts are not taken into account

        BANKSEL      NVMADRH
        MOVF          ADDRH,W
        MOVWF        NVMADRH           ; Load initial address
        MOVF          ADDRL,W
        MOVWF        NVMADRL
        MOVLW        LOW DATA_ADDR   ; Load initial data address
        MOVWF        FSR0L
        MOVLW        HIGH DATA_ADDR
        MOVWF        FSR0H
        BCF          NVMCON1,NVMREGS   ; Set Program Flash Memory as write location
        BSF          NVMCON1,WREN      ; Enable writes
        BSF          NVMCON1,LWLO     ; Load only write latches

LOOP
        MOVIW        FSR0++
        MOVWF        NVMDATL          ; Load first data byte
        MOVIW        FSR0++
        MOVWF        NVMDATH          ; Load second data byte

        MOVF          NVMADRL,W
        XORLW        0x1F             ; Check if lower bits of address are 00000
        ANDLW        0x1F             ; and if on last of 32 addresses
        BTFSCL       STATUS,Z         ; Last of 32 words?
        GOTO         START_WRITE      ; If so, go write latches into memory

        CALL         UNLOCK_SEQ       ; If not, go load latch
        INCF         NVMADRL,F        ; Increment address
        GOTO         LOOP

START_WRITE
        BCF          NVMCON1,LWLO     ; Latch writes complete, now write memory
        CALL         UNLOCK_SEQ       ; Perform required unlock sequence
        BCF          NVMCON1,WREN     ; Disable writes

UNLOCK_SEQ
        MOVLW        55h
        BCF          INTCON,GIE       ; Disable interrupts
        MOVWF        NVMCON2         ; Begin unlock sequence
        MOVLW        AAh
        MOVWF        NVMCON2
        BSF          NVMCON1,WR
        BSF          INTCON,GIE       ; Unlock sequence complete, re-enable interrupts
        return

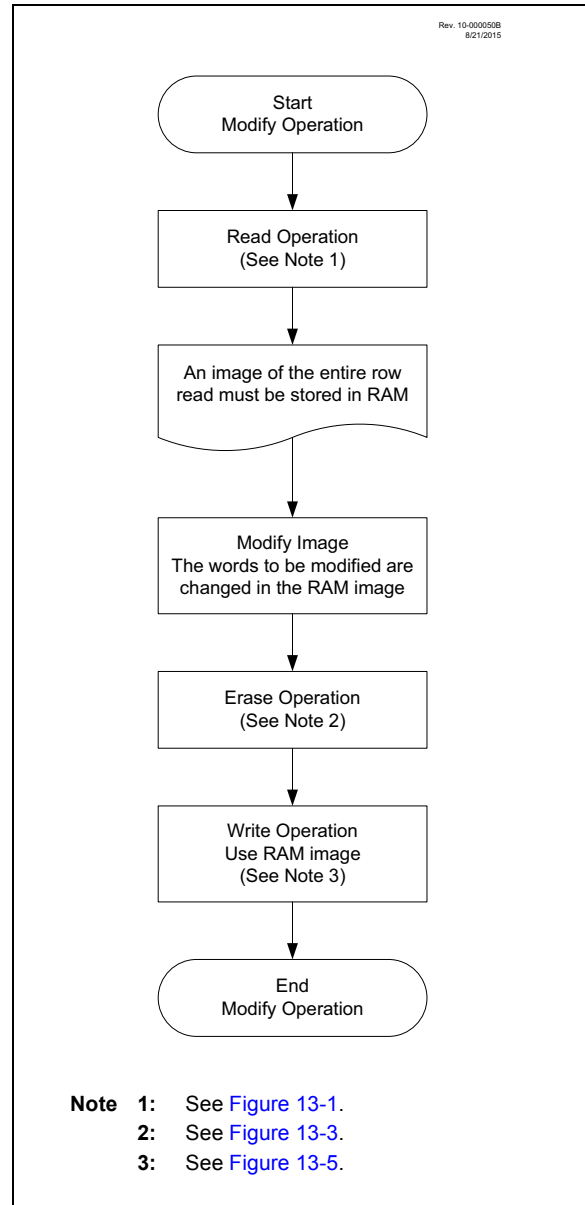
```

## 13.3.5 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 13-6: FLASH PROGRAM MEMORY MODIFY FLOWCHART**





## 13.3.6 NVMREG ACCESS TO DEVICE INFORMATION AREA, DEVICE CONFIGURATION AREA, USER ID, DEVICE ID AND CONFIGURATION WORDS

NVMREGS can be used to access the following memory regions:

- Device Information Area (DIA)
- Device Configuration Information (DCI)
- User ID region
- Device ID and Revision ID
- Configuration Words

The value of NVMREGS is set to '1' in the NVMCON1 register to access these regions. The memory regions listed above would be pointed to by PC<15> = 1, but not all addresses reference valid data. Different access may exist for reads and writes. Refer to [Table 13-3](#).

When read access is initiated on an address outside the parameters listed in [Table 13-3](#), the NVMDATH: NVMDATL register pair is cleared, reading back '0's.

**TABLE 13-3: NVMREGS ACCESS TO DEVICE INFORMATION AREA, DEVICE CONFIGURATION AREA, USER ID, DEVICE ID AND CONFIGURATION WORDS (NVMREGS = 1)**

Address	Function	Read Access	Write Access
8000h-8003h	User IDs	Yes	Yes
8005h-8006h	Device ID/Revision ID	Yes	No
8007h-800Bh	Configuration Words 1-5	Yes	No
8100h-82FFh	DIA and DCI	Yes	No

## EXAMPLE 13-5: DEVICE ID ACCESS

```

; This write routine assumes the following:
; 1. A full row 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 common RAM (locations 0x70 - 0x7F)
; 5. NVM interrupts are not taken into account

        BANKSEL          NVMADRH
        MOVF             ADDRH,W
        MOVWF           NVMADRH           ; Load initial address
        MOVF             ADDRL,W
        MOVWF           NVMADRL
        MOVLW           LOW DATA_ADDR   ; Load initial data address
        MOVWF           FSR0L
        MOVLW           HIGH DATA_ADDR
        MOVWF           FSR0H
        BCF             NVMCON1,NVMREGS  ; Set PFM as write location
        BSF             NVMCON1,WREN     ; Enable writes
        BSF             NVMCON1,LWLO    ; Load only write latches

LOOP
        MOVIW           FSR0++
        MOVWF           NVMDATL         ; Load first data byte
        MOVIW           FSR0++
        MOVWF           NVMDATAH       ; Load second data byte
        CALL            UNLOCK_SEQ      ; If not, go load latch
        INCF            NVMADRL,F       ; Increment address
        MOVF            NVMADRL,W
        XORLW           0x1F           ; Check if lower bits of address are 00000
        ANDLW           0x1F           ; and if on last of 32 addresses
        BTFSC           STATUS,Z       ; Last of 32 words?
        GOTO            START_WRITE     ; If so, go write latches into memory
        GOTO            LOOP

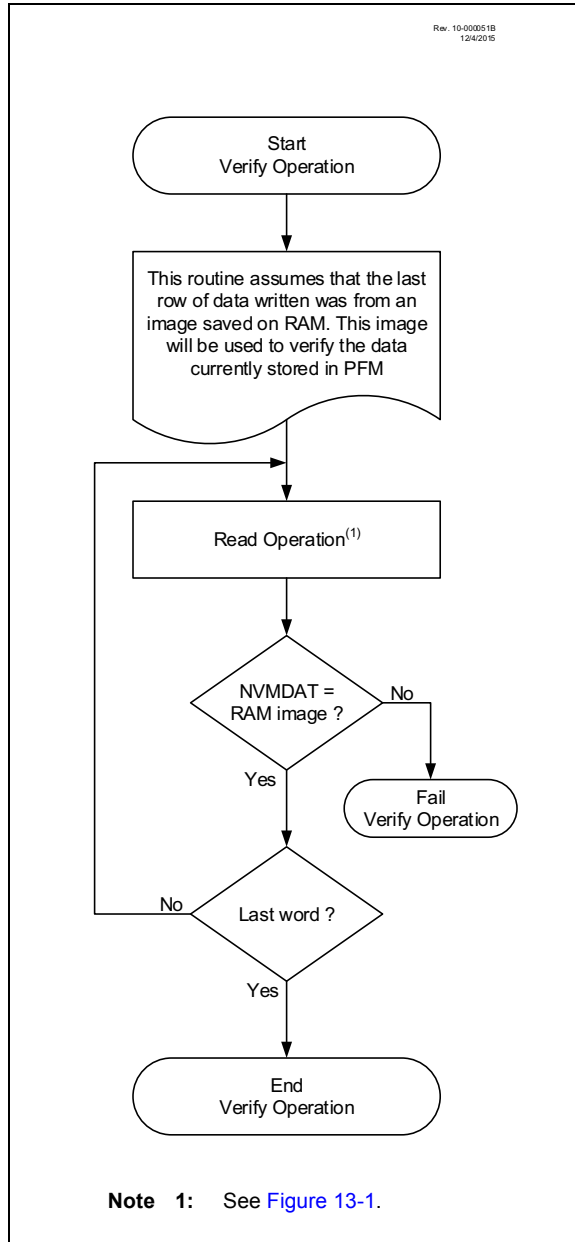
START_WRITE
        BCF             NVMCON1,LWLO    ; Latch writes complete, now write memory
        CALL            UNLOCK_SEQ      ; Perform required unlock sequence
        BCF             NVMCON1,LWLO    ; Disable writes

UNLOCK_SEQ
        MOVLW           55h
        BCF             INTCON,GIE     ; Disable interrupts
        MOVWF           NVMCON2        ; Begin unlock sequence
        MOVLW           AAh
        MOVWF           NVMCON2
        BSF             NVMCON1,WR
        BSF             INTCON,GIE     ; Unlock sequence complete, re-enable interrupts
        return
    
```

## 13.3.7 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 row then the stored program memory contents are compared with the intended data stored in RAM after the last write is complete.

**FIGURE 13-7: FLASH PROGRAM MEMORY VERIFY FLOWCHART**



## 13.3.8 WRERR BIT

The WRERR bit can be used to determine if a write error occurred.

WRERR will be set if one of the following conditions occurs:

- If WR is set while the NVMADRH:NMVADRL points to a write-protected address
- A Reset occurs while a self-write operation was in progress
- An unlock sequence was interrupted

The WRERR bit is normally set by hardware, but can be set by the user for test purposes. Once set, WRERR must be cleared in software.

**TABLE 13-4: ACTIONS FOR PFM WHEN WR = 1**

Free	LWLO	Actions for PFM when WR = 1	Comments
1	x	Erase the 32-word row of NVMADRH:NMVADRL location. See <a href="#">Section 13.3.3 “NVMREG Erase of Program Memory”</a>	<ul style="list-style-type: none"> <li>• If WP is enabled, WR is cleared and WRERR is set</li> <li>• All 32 words are erased</li> <li>• NVMDATH:NVMDATL is ignored</li> </ul>
0	1	Copy NVMDATH:NVMDATL to the write latch corresponding to NVMADR LSBs. See <a href="#">Section 13.3.3 “NVMREG Erase of Program Memory”</a>	<ul style="list-style-type: none"> <li>• Write protection is ignored</li> <li>• No memory access occurs</li> </ul>
0	0	Write the write-latch data to PFM row. See <a href="#">Section 13.3.3 “NVMREG Erase of Program Memory”</a>	<ul style="list-style-type: none"> <li>• If WP is enabled, WR is cleared and WRERR is set</li> <li>• Write latches are reset to 3FFh</li> <li>• NVMDATH:NVMDATL is ignored</li> </ul>

## 13.4 Register Definitions: Flash Program Memory Control

### REGISTER 13-1: NVMDATL: NONVOLATILE 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
NVMDAT<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                      **NVMDAT<7:0>**: Read/write value for Least Significant bits of program memory

### REGISTER 13-2: NVMDATH: NONVOLATILE 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
—		NVMDAT<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                      **NVMDAT<13:8>**: Read/write value for Most Significant bits of program memory

### REGISTER 13-3: NVMAADR: NONVOLATILE 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
NVMAADR<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                      **NVMAADR<7:0>**: Specifies the Least Significant bits for program memory address

### REGISTER 13-4: NVMAADR: NONVOLATILE 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)	NVMAADR<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                      **NVMAADR<14:8>**: Specifies the Most Significant bits for program memory address

**Note 1:** Bit is undefined while WR = 1

## REGISTER 13-5: NVMCON1: NONVOLATILE MEMORY CONTROL 1 REGISTER

U-0	R/W-0/0	R/W-0/0	R/W/HC-0/0	R/W/HC-x/q	R/W-0/0	R/S/HC-0/0	R/S/HC-0/0
—	NVMREGS	LWLO	FREE	WRERR <sup>(1,2,3)</sup>	WREN	WR <sup>(4,5,6)</sup>	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	<b>Unimplemented:</b> Read as '0'
bit 6	<b>NVMREGS:</b> Configuration Select bit 1 = Access DIA, DCI, Configuration, User ID and Device ID Registers 0 = Access Program Flash Memory
bit 5	<b>LWLO:</b> Load Write Latches Only bit <u>When FREE = 0:</u> 1 = The next WR command updates the write latch for this word within the row; no memory operation is initiated. 0 = The next WR command writes data or erases Otherwise: The bit is ignored
bit 4	<b>FREE:</b> Program Flash Memory Erase Enable bit 1 = Performs an erase operation with the next WR command; the 32-word pseudo-row containing the indicated address is erased (to all 1s) to prepare for writing. 0 = All erase operations have completed normally
bit 3	<b>WRERR:</b> Program/Erase Error Flag bit <sup>(1,2,3)</sup> This bit is normally set by hardware. 1 = A write operation was interrupted by a Reset, interrupted unlock sequence, or WR was written to one while NVMADR points to a write-protected address. 0 = The program or erase operation completed normally
bit 2	<b>WREN:</b> Program/Erase Enable bit 1 = Allows program/erase cycles 0 = Inhibits programming/erasing of program Flash
bit 1	<b>WR:</b> Write Control bit <sup>(4,5,6)</sup> <u>When NVMREG:NVMADR points to a Program Flash Memory location:</u> 1 = Initiates the operation indicated by <a href="#">Table 13-4</a> 0 = NVM program/erase operation is complete and inactive.
bit 0	<b>RD:</b> Read Control bit <sup>(7)</sup> 1 = Initiates a read at address = NVMADR1, and loads data to NVMDAT. Read takes one instruction cycle and the bit is cleared when the operation is complete. The bit can only be set (not cleared) in software. 0 = NVM read operation is complete and inactive

- Note**
- 1: Bit is undefined while WR = 1.
  - 2: Bit must be cleared by software; hardware will not clear this bit.
  - 3: Bit may be written to '1' by software in order to implement test sequences.
  - 4: This bit can only be set by following the unlock sequence of [Section 13.3.2 "NVM Unlock Sequence"](#).
  - 5: Operations are self-timed, and the WR bit is cleared by hardware when complete.
  - 6: Once a write operation is initiated, setting this bit to zero will have no effect.

# PIC16(L)F15354/55

**REGISTER 13-6: NVMCON2: NONVOLATILE 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
NVMCON2<7:0>							
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      **NVMCON2<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 NVMCON1 register. The value written to this register is used to unlock the writes.

**TABLE 13-5: SUMMARY OF REGISTERS ASSOCIATED WITH NONVOLATILE MEMORY (NVM)**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page	
INTCON	GIE	PEIE	—	—	—	—	—	INTEDG	121	
PIE7	—	—	NVMIE	NCO1IE	—	—	—	CWG1IE	129	
PIR7	—	—	NVMIF	NCO1IF	—	—	—	CWG1IF	137	
NVMCON1	—	NVMREGS	LWLO	FREE	WRERR	WREN	WR	RD	170	
NVMCON2	NVMCON2<7:0>								171	
NVMADRL	NVMADR<7:0>								169	
NVMADRH	— <sup>(1)</sup>	NVMADR<14:8>								169
NVMDATL	NVMDAT<7:0>								169	
NVMDATH	—	—	NVMDAT<13:8>						169	

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

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

## 14.0 I/O PORTS

**TABLE 14-1: PORT AVAILABILITY PER DEVICE**

Device	PORTA	PORTB	PORTC	PORTE
PIC16(L)F15354/55	•	•	•	•

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

- PORTx registers (reads the levels on the pins of the device)
- LATx registers (output latch)
- TRISx registers (data direction)
- ANSELx registers (analog select)
- WPUx registers (weak pull-up)
- INLVx registers (input level control)
- SLRCONx registers (slew rate)
- ODCONx registers (open-drain)

Most port pins share functions with device peripherals, both analog and digital. 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.

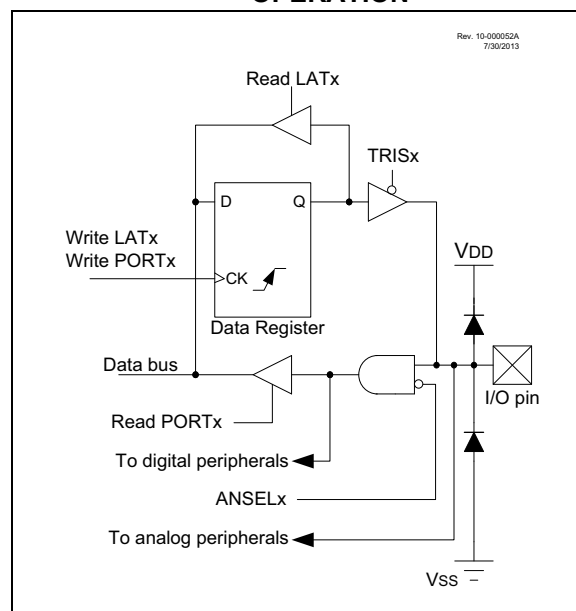
The Data Latch (LATx registers) is useful for read-modify-write operations on the value that the I/O pins are driving.

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 14-1](#).

**FIGURE 14-1: GENERIC I/O PORT OPERATION**



### 14.1 I/O Priorities

Each pin defaults to the PORT data latch after Reset. Other functions are selected with the peripheral pin select logic. See [Section 15.0 “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 ANSELx register. Digital output functions may continue to control the pin when it is in Analog mode.

Analog outputs, when enabled, take priority over the digital outputs and force the digital output driver to the high-impedance state.



## 14.2 PORTA Registers

### 14.2.1 DATA REGISTER

PORTA is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISA (Register 14-2). 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). Example 14.2.8 shows how to initialize PORTA.

Reading the PORTA register (Register 14-1) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch (LATA).

The PORT data latch LATA (Register 14-3) holds the output port data, and contains the latest value of a LATA or PORTA write.

#### EXAMPLE 14-1: 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
    
```

### 14.2.2 DIRECTION CONTROL

The TRISA register (Register 14-2) 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'.

### 14.2.3 OPEN-DRAIN CONTROL

The ODCONA register (Register 14-6) 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.

**Note:** It is not necessary to set open-drain control when using the pin for I<sup>2</sup>C; the I<sup>2</sup>C module controls the pin and makes the pin open-drain.

### 14.2.4 SLEW RATE CONTROL

The SLRCONA register (Register 14-7) 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.

### 14.2.5 INPUT THRESHOLD CONTROL

The INLVLA register (Register 14-8) 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 37-4 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.

## 14.2.6 ANALOG CONTROL

The ANSELA register ([Register 14-4](#)) 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 its TRIS bit clear and its ANSEL bit 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.

<p><b>Note:</b> 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.</p>
--

## 14.2.7 WEAK PULL-UP CONTROL

The WPUA register ([Register 14-5](#)) controls the individual weak pull-ups for each PORT pin.

## 14.2.8 PORTA FUNCTIONS AND OUTPUT PRIORITIES

Each PORTA pin is multiplexed with other functions.

Each pin defaults to the PORT latch data after Reset. Other output functions are selected with the peripheral pin select logic or by enabling an analog output, such as the DAC. See [Section 15.0 “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. Digital output functions may continue to control the pin when it is in Analog mode.

## 14.3 Register Definitions: PORTA

### REGISTER 14-1: PORTA: PORTA REGISTER

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

#### Legend:

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

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

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

### REGISTER 14-2: 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 14-3: LATA: PORTA DATA LATCH REGISTER

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-1/1	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<sup>(1)</sup>

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

## REGISTER 14-4: ANSELA: PORTA ANALOG SELECT REGISTER

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
ANSA7	ANSA6	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 7-0      **ANSA<7:0>**: Analog Select between Analog or Digital Function on pins RA<7:0>, respectively  
 1 = Analog input. Pin is assigned as analog input<sup>(1)</sup>. Digital input buffer disabled.  
 0 = Digital I/O. Pin is assigned to port or digital special function.

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

## REGISTER 14-5: WPUA: WEAK PULL-UP PORTA 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
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 = Pull-up enabled  
                   0 = Pull-up disabled

**Note 1:** The weak pull-up device is automatically disabled if the pin is configured as an output.

## REGISTER 14-6: 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
ODCA7	ODCA6	ODCA5	ODCA4	ODCA3	ODCA2	ODCA1	ODCA0
bit 7							bit 0

### Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit 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      **ODCA<7:0>**: PORTA Open-Drain Enable bits  
                   For RA<7:0> pins, respectively  
                   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 14-7: 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, respectively  
 1 = Port pin slew rate is limited  
 0 = Port pin slews at maximum rate

## REGISTER 14-8: INLVLA: PORTA 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
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, respectively  
 1 = ST input used for PORT reads and interrupt-on-change  
 0 = TTL input used for PORT reads and interrupt-on-change

**TABLE 14-2: 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
PORTA	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0	<a href="#">175</a>
TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	<a href="#">175</a>
LATA	LATA7	LATA6	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	<a href="#">176</a>
ANSELA	ANSA7	ANSA6	ANSA5	ANSA4	ANSA3	ANSA2	ANSA1	ANSA0	<a href="#">176</a>
WPUA	WPUA7	WPUA6	WPUA5	WPUA4	WPUA3	WPUA2	WPUA1	WPUA0	<a href="#">177</a>
ODCONA	ODCA7	ODCA6	ODCA5	ODCA4	ODCA3	ODCA2	ODCA1	ODCA0	<a href="#">177</a>
SLRCONA	SLRA7	SLRA6	SLRA5	SLRA4	SLRA3	SLRA2	SLRA1	SLRA0	<a href="#">178</a>
INLVLA	INLVLA7	INLVLA6	INLVLA5	INLVLA4	INLVLA3	INLVLA2	INLVLA1	INLVLA0	<a href="#">178</a>

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

## 14.4 PORTB Registers

### 14.4.1 DATA REGISTER

PORTB is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISB (Register 14-10). Setting a TRISB bit (= 1) will make the corresponding PORTB pin an input (i.e., disable the output driver). Clearing a TRISB bit (= 0) will make the corresponding PORTB pin an output (i.e., enables output driver and puts the contents of the output latch on the selected pin). Figure 14-1 shows how to initialize PORTB.

Reading the PORTB register (Register 14-9) 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).

The PORT data latch LATB (Register 14-11) holds the output port data, and contains the latest value of a LATB or PORTB write.

### 14.4.2 DIRECTION CONTROL

The TRISB register (Register 14-10) 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'.

### 14.4.3 OPEN-DRAIN CONTROL

The ODCONB register (Register 14-14) 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.

**Note:** It is not necessary to set open-drain control when using the pin for I<sup>2</sup>C; the I<sup>2</sup>C module controls the pin and makes the pin open-drain.

### 14.4.4 SLEW RATE CONTROL

The SLRCONB register (Register 14-15) 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.

### 14.4.5 INPUT THRESHOLD CONTROL

The INLVLB register (Register 14-8) 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 37-4 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.

### 14.4.6 ANALOG CONTROL

The ANSELB register (Register 14-12) 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 ANSELB bits has no effect on digital output functions. A pin with its TRIS bit clear and its ANSEL bit 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.

### 14.4.7 WEAK PULL-UP CONTROL

The WPUB register (Register 14-5) controls the individual weak pull-ups for each PORT pin.

### 14.4.8 PORTB FUNCTIONS AND OUTPUT PRIORITIES

Each PORTB pin is multiplexed with other functions.

Each pin defaults to the PORT latch data after Reset. Other output functions are selected with the peripheral pin select logic or by enabling an analog output, such as the DAC. See Section 15.0 "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. Digital output functions may continue to control the pin when it is in Analog mode.



## 14.5 Register Definitions: PORTB

### REGISTER 14-9: 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 I/O Value bits<sup>(1)</sup>  
1 = Port pin is  $\geq V_{IH}$   
0 = Port pin is  $\leq V_{IL}$

**Note 1:** Writes to PORTB are actually written to corresponding LATB register. The actual I/O pin values are read from the PORTB register.

### REGISTER 14-10: 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 bit  
1 = PORTB pin configured as an input (tri-stated)  
0 = PORTB pin configured as an output

## REGISTER 14-11: 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>**: RB<7:0> Output Latch Value bits<sup>(1)</sup>

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

## REGISTER 14-12: ANSELB: PORTB ANALOG SELECT REGISTER

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
ANSB7	ANSB6	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 7-0                  **ANSB<7:0>**: Analog Select between Analog or Digital Function on pins RB<7:0>, respectively  
1 = Analog input. Pin is assigned as analog input<sup>(1)</sup>. Digital input buffer disabled.  
0 = Digital I/O. Pin is assigned to port or digital special function.

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

## REGISTER 14-13: WPUB: WEAK PULL-UP PORTB 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
WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0
bit 7							bit 0

### Legend:

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

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

## REGISTER 14-14: 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
ODCB7	ODCB6	ODCB5	ODCB4	ODCB3	ODCB2	ODCB1	ODCB0
bit 7							bit 0

### Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit 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      **ODCB<7:0>**: PORTB Open-Drain Enable bits  
                   For RB<7:0> pins, respectively  
                   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 14-15: 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, respectively  
1 = Port pin slew rate is limited  
0 = Port pin slews at maximum rate

## REGISTER 14-16: INLVLB: PORTB 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
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, respectively  
1 = ST input used for PORT reads and interrupt-on-change  
0 = TTL input used for PORT reads and interrupt-on-change

## TABLE 14-3: 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
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	<a href="#">181</a>
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	<a href="#">181</a>
LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	<a href="#">182</a>
ANSELB	ANSB7	ANSB6	ANSB5	ANSB4	ANSB3	ANSB2	ANSB1	ANSB0	<a href="#">182</a>
WPUB	WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0	<a href="#">183</a>
ODCONB	ODCB7	ODCB6	ODCB5	ODCB4	ODCB3	ODCB2	ODCB1	ODCB0	<a href="#">183</a>
SLRCONB	SLRB7	SLRB6	SLRB5	SLRB4	SLRB3	SLRB2	SLRB1	SLRB0	<a href="#">184</a>
INLVLB	INLVLB7	INLVLB6	INLVLB5	INLVLB4	INLVLB3	INLVLB2	INLVLB1	INLVLB0	<a href="#">184</a>

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

## 14.6 PORTC Registers

### 14.6.1 DATA REGISTER

PORTC is an 8-bit wide bidirectional port. The corresponding data direction register is TRISC (Register 14-18). 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). Figure 14-1 shows how to initialize an I/O port.

Reading the PORTC register (Register 14-17) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch (LATC).

The PORT data latch LATC (Register 14-19) holds the output port data, and contains the latest value of a LATC or PORTC write.

### 14.6.2 DIRECTION CONTROL

The TRISC register (Register 14-18) 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'.

### 14.6.3 OPEN-DRAIN CONTROL

The ODCONC register (Register 14-22) 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.

**Note:** It is not necessary to set open-drain control when using the pin for I<sup>2</sup>C; the I<sup>2</sup>C module controls the pin and makes the pin open-drain.

### 14.6.4 SLEW RATE CONTROL

The SLRCONC register (Register 14-23) 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.

### 14.6.5 INPUT THRESHOLD CONTROL

The INLVLC register (Register 14-24) 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 37-4 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.

### 14.6.6 ANALOG CONTROL

The ANSEL register (Register 14-20) is used to configure the Input mode of an I/O pin to analog. Setting the appropriate ANSEL bit high will cause all digital reads on the pin to be read as '0' and allow analog functions on the pin to operate correctly.

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

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

### 14.6.7 WEAK PULL-UP CONTROL

The WPUC register (Register 14-21) controls the individual weak pull-ups for each port pin.

### 14.6.8 PORTC FUNCTIONS AND OUTPUT PRIORITIES

Each pin defaults to the PORT latch data after Reset. Other output functions are selected with the peripheral pin select logic. See Section 15.0 "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. Digital output functions may continue to control the pin when it is in Analog mode.

## 14.7 Register Definitions: PORTC

### REGISTER 14-17: 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<sup>(1)</sup>  
1 = Port pin is  $\geq V_{IH}$   
0 = Port pin is  $\leq V_{IL}$

**Note 1:** Writes to PORTC are actually written to corresponding LATC register. The actual I/O pin values are read from the PORTC register.

### REGISTER 14-18: 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 14-19: LATC: PORTC DATA LATCH REGISTER

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

#### Legend:

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

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

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

## REGISTER 14-20: 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	R/W-1/1	R/W-1/1
ANSC7	ANSC6	ANSC5	ANSC4	ANSC3	ANSC2	ANSC1	ANSC0
bit 7							bit 0

### Legend:

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

bit 7-0      **ANSC<7:0>**: Analog Select between Analog or Digital Function on Pins RC<7:0>, respectively<sup>(1)</sup>  
 1 = Analog input. Pin is assigned as analog input<sup>(1)</sup>. Digital input buffer disabled.  
 0 = Digital I/O. Pin is assigned to port or digital special function.

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

## REGISTER 14-21: WPUC: WEAK PULL-UP PORTC 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
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 = Pull-up enabled  
 0 = Pull-up disabled

## REGISTER 14-22: 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
ODCC7	ODCC6	ODCC5	ODCC4	ODCC3	ODCC2	ODCC1	ODCC0
bit 7							bit 0

### Legend:

R = Readable bit                      W = Writable bit                      U = Unimplemented bit, read as '0'  
u = Bit is unchanged                      x = Bit 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                      **ODCC<7:0>**: PORTC Open-Drain Enable bits  
For RC<7:0> pins, respectively  
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 14-23: 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, respectively  
1 = Port pin slew rate is limited  
0 = Port pin slews at maximum rate

## REGISTER 14-24: 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, respectively  
1 = ST input used for PORT reads and interrupt-on-change  
0 = TTL input used for PORT reads and interrupt-on-change



**TABLE 14-4: 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
PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	<a href="#">186</a>
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	<a href="#">186</a>
LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	<a href="#">186</a>
ANSEL	ANSEL7	ANSEL6	ANSEL5	ANSEL4	ANSEL3	ANSEL2	ANSEL1	ANSEL0	<a href="#">187</a>
WPUC	WPUC7	WPUC6	WPUC5	WPUC4	WPUC3	WPUC2	WPUC1	WPUC0	<a href="#">187</a>
ODCONC	ODCC7	ODCC6	ODCC5	ODCC4	ODCC3	ODCC2	ODCC1	ODCC0	<a href="#">188</a>
SLRCONC	SLRC7	SLRC6	SLRC5	SLRC4	SLRC3	SLRC2	SLRC1	SLRC0	<a href="#">188</a>
INLVLC	INLVLC7	INLVLC6	INLVLC5	INLVLC4	INLVLC3	INLVLC2	INLVLC1	INLVLC0	<a href="#">188</a>

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

## 14.8 PORTE Registers

### 14.8.1 DATA REGISTER

PORTE is a single bit-bit wide port. The corresponding data direction register is TRISE (Register 14-25). Setting a TRISE bit (= 1) will make the corresponding PORTE pin an input (i.e., disable the output driver). Clearing a TRISE bit (= 0) will make the corresponding PORTE pin an output (i.e., enables output driver and puts the contents of the output latch on the selected pin). Figure 14-1 shows how to initialize PORTE.

Reading the PORTE register (Register 14-25) reads the status of the pins, whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch (LATE).

### 14.8.2 DIRECTION CONTROL

The TRISE register (Register 14-26) controls the PORTE pin output drivers, even when they are being used as analog inputs. The user should ensure the bits in the TRISE register are maintained set when using them as analog inputs. I/O pins configured as analog inputs always read '0'.

**Note:** The TRISE3 bit is a read-only bit and it always reads a '1'.

### 14.8.3 INPUT THRESHOLD CONTROL

The INLVLE register (Register 14-28) 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 37-4 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.

### 14.8.4 WEAK PULL-UP CONTROL

The WPUE register (Register 14-27) controls the individual weak pull-ups for each port pin.

### 14.8.5 PORTE FUNCTIONS AND OUTPUT PRIORITIES

Each pin defaults to the PORT latch data after Reset. Other output functions are selected with the peripheral pin select logic. See Section 15.0 "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. Digital output functions may continue to control the pin when it is in Analog mode.

## 14.9 Register Definitions: PORTE

### REGISTER 14-25: 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      **RE<3>:** PORTE Input Pin bit  
 1 = Port pin is > VIH  
 0 = Port pin is < VIL

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

### REGISTER 14-26: TRISE: PORTE TRI-STATE REGISTER

U-0	U-0	U-0	U-0	U-1	U-0	U-0	U-0
—	—	—	—	—(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 7-4      **Unimplemented:** Read as '0'

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

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

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

## REGISTER 14-27: 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 <sup>(1)</sup>	—	—	—
bit 7							bit 0

### Legend:

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

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

bit 3      **WPUE3:** Weak Pull-up Register bit<sup>(1)</sup>  
             1 = Pull-up enabled  
             0 = Pull-up disabled

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

**Note 1:** If MCLRE = 1, the weak pull-up in RE3 is always enabled; bit WPUE3 is not affected.

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

## REGISTER 14-28: 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 bits  
             For RE3 pin,  
             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 14-5: 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
PORTE	—	—	—	—	RE3	—	—	—	191
TRISE	—	—	—	—	— <sup>(1)</sup>	—	—	—	191
WPUE	—	—	—	—	WPUE3	—	—	—	192
INLVLE	—	—	—	—	INLVLE3	—	—	—	192

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

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

**TABLE 14-6: SUMMARY OF CONFIGURATION WORD WITH PORTE**

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	—	—	$\overline{\text{DEBUG}}$	STVREN	PPS1WAY	ZCDDIS	BORV	—	77
	7:0	BOREN <1:0>		$\overline{\text{LPBOREN}}$	—	—	—	$\overline{\text{PWRTÉ}}$	MCLRE	

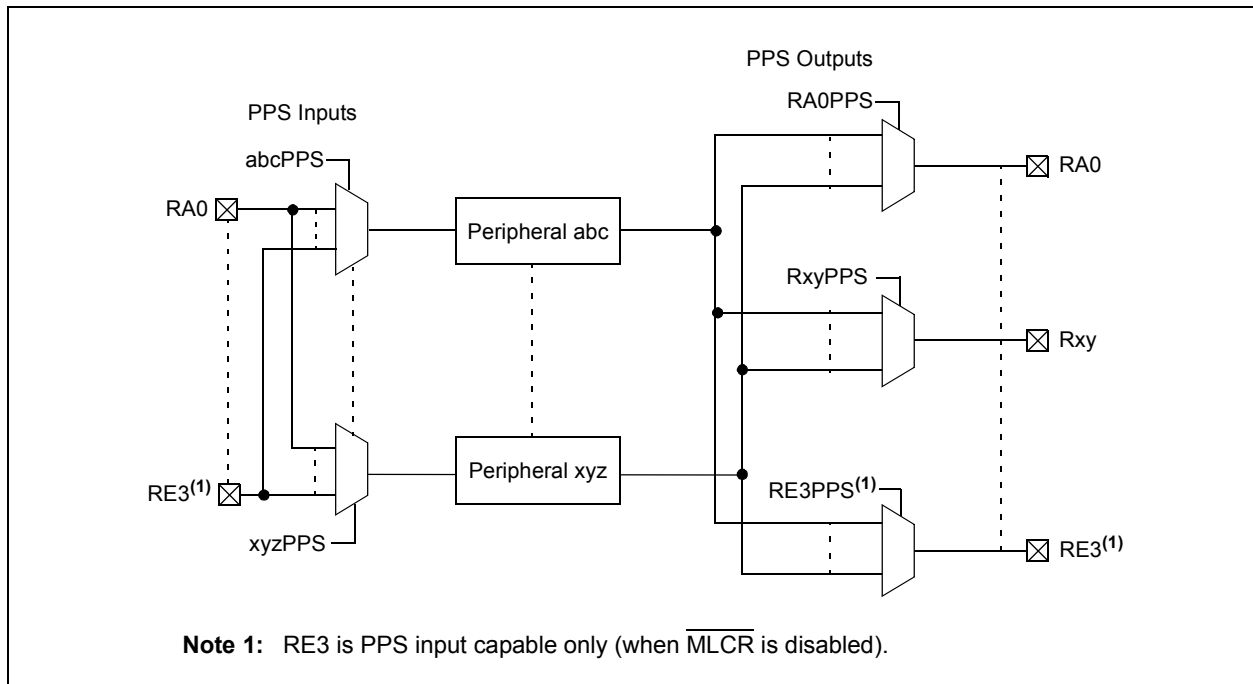
**Legend:** — = unimplemented location, read as '0'. Shaded cells are not used by PORTE.

## 15.0 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 15-1](#).

**FIGURE 15-1: SIMPLIFIED PPS BLOCK DIAGRAM**



### 15.1 PPS Inputs

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

Although every peripheral has its own PPS input selection register, the selections are identical for every peripheral as shown in [Register 15-1](#).

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

### 15.2 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 are (See [Section 15.3 “Bidirectional Pins”](#)):

- EUSART (synchronous operation)
- MSSP (I<sup>2</sup>C)

Although every pin has its own PPS peripheral selection register, the selections are identical for every pin as shown in [Register 15-2](#).

**Note:** The notation “Rxy” is a place holder for the pin port and bit identifiers. For example, x and y for PORTA bit 0 would be A and 0, respectively, resulting in the pin PPS output selection register RA0PPS.

# PIC16(L)F15354/55

**TABLE 15-1: PPS INPUT SIGNAL ROUTING OPTIONS (PIC16(L)F15354/55)**

INPUT SIGNAL NAME	Input Register Name	Default Location at POR	Reset Value (xxxPPS<5:0>)	Remappable to Pins of PORTx		
				PIC16(L)F15354/55		
				PORTA	PORTB	PORTC
INT	INTPPS	RB0	01000	•	•	
T0CKI	T0CKIPPS	RA4	00100	•	•	
T1CKI	T1CKIPSS	RC0	10000	•		•
T1G	T1GPPS	RB5	01101		•	•
T2IN	T2INPPS	RC3	10011	•		•
CCP1	CCP1PPS	RC2	10010		•	•
CCP2	CCP2PPS	RC1	10001		•	•
CWG1IN	CWG1PPS	RB0	01000		•	•
CLCIN0	CLCIN0PPS	RA0	00000	•		•
CLCIN1	CLCIN1PPS	RA1	00001	•		•
CLCIN2	CLCIN2PPS	RB6	01110		•	•
CLCIN3	CLCIN3PPS	RB7	01111		•	•
ADACT	ADACTPPS	RB4	01100		•	•
SCK1/SCL1	SSP1CLKPPS	RC3	10011		•	•
SDI1/SDA1	SSP1DATPPS	RC4	10100		•	•
SS1	SSP1SS1PPS	RA5	00101	•		•
SCK2/SCL2	SSP2CLKPPS	RB1	01001		•	•
SDI2/SDA2	SSP2DATPPS	RB2	01010		•	•
SS2	SSP2SSPPS	RB0	01000		•	•
RX1/DT1	RX1DTPPS	RC7	10111		•	•
CK1	TX1CKPPS	RC6	10110		•	•
RX2/DT2	RX2DTPPS	RB7	01111		•	•
CK2	TX2CKPPS	RB6	01110		•	•

**TABLE 15-2: PPS INPUT REGISTER VALUES**

Desired Input Pin	Value to Write to Register
RA0	0x00
RA1	0x01
RA2	0x02
RA3	0x03
RA4	0x04
RA5	0x05
RA6	0x06
RA7	0x07
RB0	0x08
RB1	0x09
RB2	0x0A
RB3	0x0B
RB4	0x0C
RB5	0x0D
RB6	0x0E
RB7	0x0F
RC0	0x10
RC1	0x11
RC2	0x12
RC3	0x13
RC4	0x14
RC5	0x15
RC6	0x16
RC7	0x17

**Note 1:** Only a few of the values in this column are valid for any given signal. For example, since the INT signal can only be mapped to PORTA or PORTB pins, only the register values 0x00-0x0F (corresponding to RA<7:0> and RB<7:0>) are valid values to write to the INTPPS register.



## 15.3 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<sup>2</sup>C)

**Note:** The I<sup>2</sup>C SCLx and SDAX functions can be remapped through PPS. However, only the RB1, RB2, RC3 and RC4 pins have the I<sup>2</sup>C and SMBus specific input buffers implemented (I<sup>2</sup>C mode disables INLVL and sets thresholds that are specific for I<sup>2</sup>C). If the SCLx or SDAX functions are mapped to some other pin (other than RB1, RB2, RC3 or RC4), the general purpose TTL or ST input buffers (as configured based on INLVL register setting) will be used instead. In most applications, it is therefore recommended only to map the SCLx and SDAX pin functions to the RB1, RB2, RC3 or RC4 pins.

## 15.4 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 15-1](#).

### EXAMPLE 15-1: 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
```

## 15.5 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.

## 15.6 Operation During Sleep

PPS input and output selections are unaffected by Sleep.

## 15.7 Effects of a Reset

A device Power-on-Reset (POR) clears all PPS input and output selections to their default values (Permanent Lock Removed). All other Resets leave the selections unchanged. Default input selections are shown in [Table 15-1](#).

**TABLE 15-3: PPS OUTPUT SIGNAL ROUTING OPTIONS (PIC16(L)F15354/55)**

Output Signal Name	RxyPPS Register Value	Remappable to Pins of PORTx		
		PIC16(L)F15354/55		
		PORTA	PORTB	PORTC
CLKR	0x1B		•	•
NCO1OUT	0x1A	•		•
TMR0	0x19		•	•
SDO2/SDA2	0x18		•	•
SCK2/SCL2	0x17		•	•
SDO1/SDA1	0x16		•	•
SCK1/SCL1	0x15		•	•
C2OUT	0x14	•		•
C1OUT	0x13	•		•
DT2	0x12		•	•
TX2/CK2	0x11		•	•
DT1	0x10		•	•
TX1/CK1	0x0F		•	•
PWM6OUT	0x0E	•		•
PWM5OUT	0x0D	•		•
PWM4OUT	0x0C		•	•
PWM3OUT	0x0B		•	•
CCP2	0x0A		•	•
CCP1	0x09		•	•
CWG1D	0x08		•	•
CWG1C	0x07		•	•
CWG1B	0x06		•	•
CWG1A	0x05		•	•
CLC4OUT	0x04		•	•
CLC3OUT	0x03		•	•
CLC2OUT	0x02	•		•
CLC1OUT	0x01	•		•

## 15.8 Register Definitions: PPS Input Selection

**REGISTER 15-1: xxxPPS: PERIPHERAL xxx INPUT SELECTION<sup>(1)</sup>**

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 5-0      **xxxPPS<5:0>:** Peripheral xxx Input Selection bits  
See [Table 15-1](#).

- Note 1:** The “xxx” in the register name “xxxPPS” represents the input signal function name, such as “INT”, “TOCKI”, “RX”, etc. This register summary shown here is only a prototype of the array of actual registers, as each input function has its own dedicated SFR (ex: INTPPS, T0CKIPPS, RXPPS, etc.).
- 2:** Each specific input signal may only be mapped to a subset of these I/O pins, as shown in [Table 15-1](#). Attempting to map an input signal to a non-supported I/O pin will result in undefined behavior. For example, the “INT” signal may be mapped to any PORTA or PORTB pin. Therefore, the INTPPS register may be written with values from 0x00-0x0F (corresponding to RA0-RB7). Attempting to write 0x10 or higher to the INTPPS register is not supported and will result in undefined behavior.

## REGISTER 15-2: RxyPPS: PIN Rxy OUTPUT SOURCE SELECTION REGISTER

U-0	U-0	U-0	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u
—	—	—	RxyPPS<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      **RxyPPS<4:0>:** Pin Rxy Output Source Selection bits  
See [Table 15-3](#).

## REGISTER 15-3: 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.

# PIC16(L)F15354/55

**TABLE 15-4: 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	200
INTPPS	—	—	INTPPS<5:0>						199
T0CKIPPS	—	—	T0CKIPPS<5:0>						199
T1CKIPPS	—	—	T1CKIPPS<5:0>						199
T1GPPS	—	—	T1GPPS<5:0>						199
T2INPPS			T2INPPS<5:0>						199
CCP1PPS	—	—	CCP1PPS<5:0>						199
CCP2PPS	—	—	CCP2PPS<5:0>						199
CWG1PPS	—	—	CWG1PPS<5:0>						199
SSP1CLKPPS	—	—	SSP1CLKPPS<5:0>						199
SSP1DATPPS	—	—	SSP1DATPPS<5:0>						199
SSP1SSPPS	—	—	SSP1SSPPS<5:0>						199
SSP2CLKPPS	—	—	SSP2CLKPPS<5:0>						199
SSP2DATPPS	—	—	SSP2DATPPS<5:0>						199
SSP2SSPPS	—	—	SSP2SSPPS<5:0>						199
RX1DTPPS	—	—	RX1DTPPS<5:0>						199
TX1CKPPS	—	—	TX1CKPPS<5:0>						199
CLCIN0PPS	—	—	CLCIN0PPS<5:0>						199
CLCIN1PPS	—	—	CLCIN1PPS<5:0>						199
CLCIN2PPS	—	—	CLCIN2PPS<5:0>						199
CLCIN3PPS	—	—	CLCIN3PPS<5:0>						199
RX2DTPPS	—	—	RX2DTPPS<5:0>						199
TX2CKPPS	—	—	TX2CKPPS<5:0>						199
ADACTPPS	—	—	ADACTPPS<5:0>						199
RA0PPS	—	—	—	RA0PPS<4:0>					200
RA1PPS	—	—	—	RA1PPS<4:0>					200
RA2PPS	—	—	—	RA2PPS<4:0>					200
RA3PPS	—	—	—	RA3PPS<4:0>					200
RA4PPS	—	—	—	RA4PPS<4:0>					200
RA5PPS	—	—	—	RA5PPS<4:0>					200
RA6PPS	—	—	—	RA6PPS<4:0>					200
RA7PPS	—	—	—	RA7PPS<4:0>					200
RB0PPS	—	—	—	RB0PPS<4:0>					200
RB1PPS	—	—	—	RB1PPS<4:0>					200
RB2PPS	—	—	—	RB2PPS<4:0>					200
RB3PPS	—	—	—	RB3PPS<4:0>					200
RB4PPS	—	—	—	RB4PPS<4:0>					200
RB5PPS	—	—	—	RB5PPS<4:0>					200
RB6PPS	—	—	—	RB6PPS<4:0>					200
RB7PPS	—	—	—	RB7PPS<4:0>					200
RC0PPS	—	—	—	RC0PPS<4:0>					200
RC1PPS	—	—	—	RC1PPS<4:0>					200

**Legend:** — = unimplemented, read as '0'. Shaded cells are unused by the PPS module.

**TABLE 15-4: SUMMARY OF REGISTERS ASSOCIATED WITH THE PPS MODULE (CONTINUED)**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
RC2PPS	—	—	—					RC2PPS<4:0>	200
RC3PPS	—	—	—					RC3PPS<4:0>	200
RC4PPS	—	—	—					RC4PPS<4:0>	200
RC5PPS	—	—	—					RC5PPS<4:0>	200
RC6PPS	—	—	—					RC6PPS<4:0>	200
RC7PPS	—	—	—					RC7PPS<4:0>	200

**Legend:** — = unimplemented, read as '0'. Shaded cells are unused by the PPS module.

## 16.0 PERIPHERAL MODULE DISABLE

The PIC16(L)F15354/55 provides the ability to disable selected modules, placing them into the lowest possible Power mode.

For legacy reasons, all modules are ON by default following any Reset.

### 16.1 Disabling a Module

Disabling a module has the following effects:

- All clock and control inputs to the module are suspended; there are no logic transitions, and the module will not function.
- The module is held in Reset:
  - Writing to SFRs is disabled
  - Reads return 00h

### 16.2 Enabling a module

When the register bit is cleared, the module is re-enabled and will be in its Reset state; SFR data will reflect the POR Reset values.

Depending on the module, it may take up to one full instruction cycle for the module to become active. There should be no interaction with the module (e.g., writing to registers) for at least one instruction after it has been re-enabled.

### 16.3 Disabling a Module

When a module is disabled, all the associated PPS selection registers (Registers xxxPPS [Register 15-1](#), [15-2](#), and [15-3](#)), are also disabled.

### 16.4 System Clock Disable

Setting SYSCMD (PMD0, [Register 16-1](#)) disables the system clock (FOSC) distribution network to the peripherals. Not all peripherals make use of SYSCLK, so not all peripherals are affected. Refer to the specific peripheral description to see if it will be affected by this bit.

## 16.5 Register Definitions: Peripheral Module Disable Control

### REGISTER 16-1: PMD0: PMD CONTROL REGISTER 0

R/W-0/0	R/W-0/0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
SYSCMD	FVRMD	—	—	—	NVMMD	CLKRMD	IOCMD
7							0

#### Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	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      **SYSCMD:** Disable Peripheral System Clock Network bit  
 See description in [Section 16.4 "System Clock Disable"](#).  
 1 = System clock network disabled (a.k.a. FOSC)  
 0 = System clock network enabled
- bit 6      **FVRMD:** Disable Fixed Voltage Reference (FVR) bit  
 1 = FVR module disabled  
 0 = FVR module enabled
- bit 5-3    **Unimplemented:** Read as '0'
- bit 2      **NVMMD:** NVM Module Disable bit<sup>(1)</sup>  
 1 = User memory reading and writing is disabled; NVMCON registers cannot be written; FSR access to these locations returns zero.  
 0 = NVM module enabled
- bit 1      **CLKRMD:** Disable Clock Reference CLKR bit  
 1 = CLKR module disabled  
 0 = CLKR module enabled
- bit 0      **IOCMD:** Disable Interrupt-on-Change bit, All Ports  
 1 = IOC module(s) disabled  
 0 = IOC module(s) enabled

**Note 1:** When enabling NVM, a delay of up to 1  $\mu$ s may be required before accessing data.



## REGISTER 16-2: PMD1: PMD CONTROL REGISTER 1

R/W-0/0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
NCO1MD	—	—	—	—	TMR2MD	TMR1MD	TMR0MD
bit 7						bit 0	

### Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	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      **NCO1MD:** Disable Numerically Control Oscillator bit  
           1 = NCO1 module disabled  
           0 = NCO1 module enabled
- bit 6-3    **Unimplemented:** Read as '0'
- bit 2      **TMR2MD:** Disable Timer TMR2 bit  
           1 = Timer2 module disabled  
           0 = Timer2 module enabled
- bit 1      **TMR1MD:** Disable Timer TMR1 bit  
           1 = Timer1 module disabled  
           0 = Timer1 module enabled
- bit 0      **TMR0MD:** Disable Timer TMR0 bit  
           1 = Timer0 module disabled  
           0 = Timer0 module enabled

**REGISTER 16-3: PMD2: PMD CONTROL REGISTER 2**

U-0	R/W-0/0	R/W-0/0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
—	DAC1MD	ADCMD	—	—	CMP2MD	CMP1MD	ZCDMD
bit 7							bit 0

**Legend:**

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	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        **Unimplemented:** Read as '0'
- bit 6        **DAC1MD:** Disable DAC1 bit  
1 = DAC module disabled  
0 = DAC module enabled
- bit 5        **ADCMD:** Disable ADC bit  
1 = ADC module disabled  
0 = ADC module enabled
- bit 4-3      **Unimplemented:** Read as '0'
- bit 2        **CMP2MD:** Disable Comparator C2 bit  
1 = C2 module disabled  
0 = C2 module enabled
- bit 1        **CMP1MD:** Disable Comparator C1 bit  
1 = C1 module disabled  
0 = C1 module enabled
- bit 0        **ZCDMD:** Disable ZCD bit  
1 = ZCD module disabled  
0 = ZCD module enabled

**REGISTER 16-4: PMD3: PMD CONTROL REGISTER 3**

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
—	—	PWM6MD	PWM5MD	PWM4MD	PWM3MD	CCP2MD	CCP1MD
bit 7							bit 0

**Legend:**

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	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       **PWM6MD:** Disable Pulse-Width Modulator PWM6 bit  
1 = PWM6 module disabled  
0 = PWM6 module enabled
- bit 4       **PWM5MD:** Disable Pulse-Width Modulator PWM5 bit  
1 = PWM5 module disabled  
0 = PWM5 module enabled
- bit 3       **PWM4MD:** Disable Pulse-Width Modulator PWM4 bit  
1 = PWM4 module disabled  
0 = PWM4 module enabled
- bit 2       **PWM3MD:** Disable Pulse-Width Modulator PWM3 bit  
1 = PWM3 module disabled  
0 = PWM3 module enabled
- bit 1       **CCP2MD:** Disable CCP2 bit  
1 = CCP2 module disabled  
0 = CCP2 module enabled
- bit 0       **CCP1MD:** Disable CCP1 bit  
1 = CCP1 module disabled  
0 = CCP1 module enabled

**REGISTER 16-5: PMD4: PMD CONTROL REGISTER 4**

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	U-0	U-0	R/W-0/0
UART2MD	UART1MD	MSSP2MD	MSSP1MD	—	—	—	CWG1MD
bit 7							bit 0

**Legend:**

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	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      **UART2MD:** Disable EUSART2 bit  
1 = EUSART2 module disabled  
0 = EUSART2 module enabled
- bit 6      **UART1MD:** Disable EUSART1 bit  
1 = EUSART1 module disabled  
0 = EUSART1 module enabled
- bit 5      **MSSP2MD:** Disable MSSP2 bit  
1 = MSSP2 module disabled  
0 = MSSP2 module enabled
- bit 4      **MSSP1MD:** Disable MSSP1 bit  
1 = MSSP1 module disabled  
0 = MSSP1 module enabled
- bit 3-1    **Unimplemented:** Read as '0'
- bit 0      **CWG1MD:** Disable CWG1 bit  
1 = CWG1 module disabled  
0 = CWG1 module enabled

**REGISTER 16-6: PMD5 – PMD CONTROL REGISTER 5**

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0
—	—	—	CLC4MD	CLC3MD	CLC2MD	CLC1MD	—
bit 7							bit 0

**Legend:**

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	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       **CLC4MD:** Disable CLC4 bit  
             1 = CLC4 module disabled  
             0 = CLC4 module enabled
- bit 3       **CLC3MD:** Disable CLC3 bit  
             1 = CLC3 module disabled  
             0 = CLC3 module enabled
- bit 2       **CLC2MD:** Disable CLC2 bit  
             1 = CLC2 module disabled  
             0 = CLC2 module enabled
- bit 1       **CLC1MD:** Disable CLC1 bit  
             1 = CLC1 module disabled  
             0 = CLC1 module enabled
- bit 0       **Unimplemented:** Read as '0'

**TABLE 16-1: SUMMARY OF REGISTERS ASSOCIATED WITH THE PMD MODULE**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
PMD0	SYSCMD	FVRMD	—	—	—	NVMMD	CLKRMD	IOCMD	<a href="#">204</a>
PMD1	NCO1MD	—	—	—	—	TMR2MD	TMR1MD	TMR0MD	<a href="#">205</a>
PMD2	—	DAC1MD	ADCMD	—	—	CMP2MD	CMP1MD	ZCDMD	<a href="#">206</a>
PMD3	—	—	PWM6MD	PWM5MD	PWM4MD	PWM3MD	CCP2MD	CCP1MD	<a href="#">207</a>
PMD4	UART2MD	UART1MD	MSSP2MD	MSSP1MD	—	—	—	CWG1MD	<a href="#">208</a>
PMD5	—	—	—	CLC4MD	CLC3MD	CLC2MD	CLC1MD	—	<a href="#">209</a>

**Legend:** — = unimplemented, read as '0'. Shaded cells are unused by the PMD module.

## 17.0 INTERRUPT-ON-CHANGE

All pins on ports A, B and C and lower four bits of PORTE 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 17-1 is a block diagram of the IOC module.

### 17.1 Enabling the Module

To allow individual pins to generate an interrupt, the IOCIE bit of the PIE0 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.

### 17.2 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.

## 17.3 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 PIR0 register reflects the status of all IOCxF bits.

### 17.3.1 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.

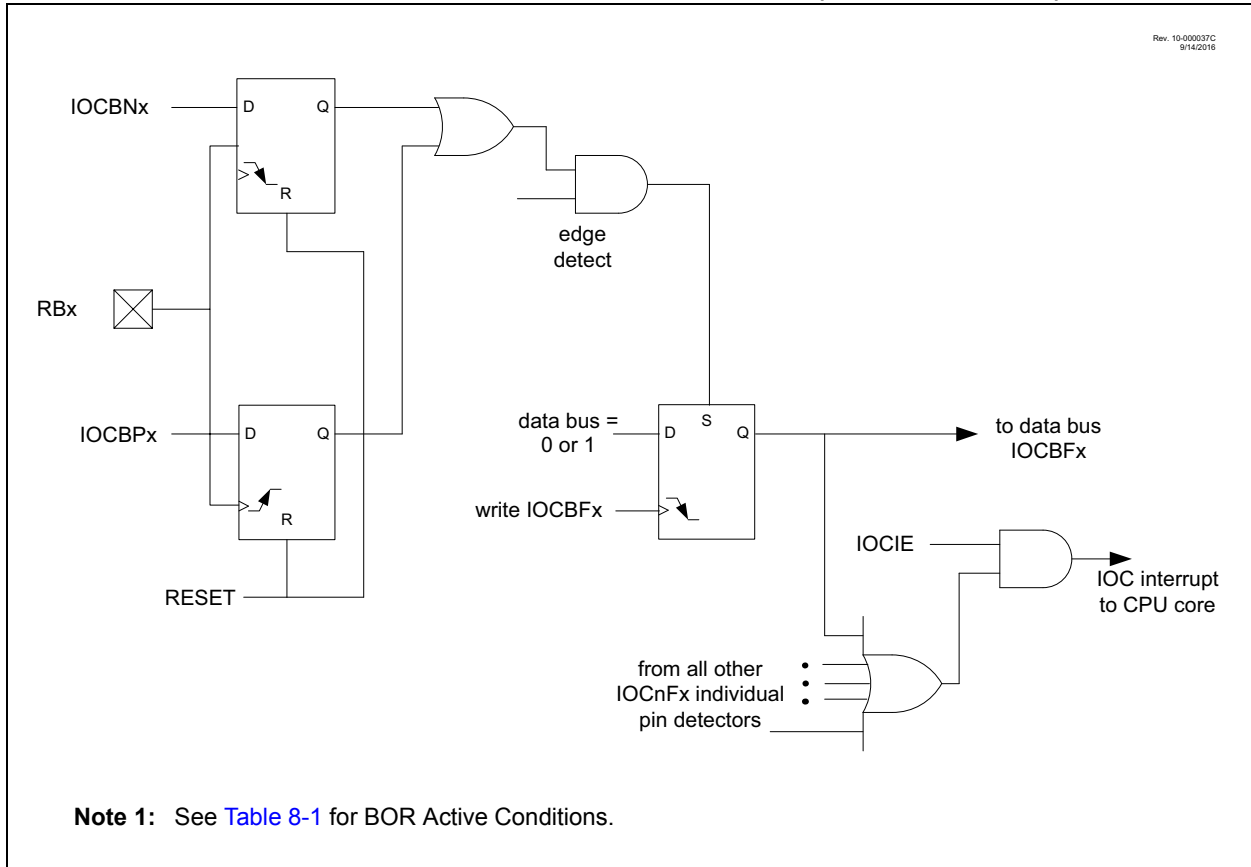
#### EXAMPLE 17-1: CLEARING INTERRUPT FLAGS (PORTA EXAMPLE)

```
MOVLW0xff
XORWFIOCAF, W
ANDWFIOCAF, F
```

### 17.4 Operation in Sleep

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

**FIGURE 17-1: INTERRUPT-ON-CHANGE BLOCK DIAGRAM (PORTB EXAMPLE)**





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

### REGISTER 17-1: 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. IOCAF<sub>x</sub> bit and IOCIF flag will be set upon detecting an edge.  
 0 = Interrupt-on-Change disabled for the associated pin.

### REGISTER 17-2: 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. IOCAF<sub>x</sub> bit and IOCIF flag will be set upon detecting an edge.  
 0 = Interrupt-on-Change disabled for the associated pin.

## REGISTER 17-3: 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 17-4: 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 17-5: 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 17-6: 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 17-7: 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 17-8: 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

# PIC16(L)F15354/55

## REGISTER 17-9: 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 17-10: IOCEP: INTERRUPT-ON-CHANGE PORTE POSITIVE EDGE REGISTER

U-0	U-0	U-0	U-0	R/W/HS-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	HS - Bit is set in hardware

bit 7-4 **Unimplemented**: Read as '0'  
 bit 3 **IOCEP3**: Interrupt-on-Change PORTE Positive Edge Enable bit  
 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  
 bit 2-0 **Unimplemented**: Read as '0'

## REGISTER 17-11: IOCN3: INTERRUPT-ON-CHANGE PORTE NEGATIVE EDGE REGISTER

U-0	U-0	U-0	U-0	R/W/HS-0/0	U-0	U-0	U-0
—	—	—	—	IOCN3	—	—	—
bit 7				bit 0			

### Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	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 **IOCN3:** Interrupt-on-Change PORTE Negative Edge Enable bit  
 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

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

## REGISTER 17-12: IOCF3: 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
—	—	—	—	IOCF3	—	—	—
bit 7				bit 0			

### Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	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 **IOCF3:** Interrupt-on-Change PORTE Flag bit  
 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

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

**TABLE 17-1: SUMMARY OF REGISTERS ASSOCIATED WITH INTERRUPT-ON-CHANGE**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
INTCON	GIE	PEIE	—	—	—	—	—	INTEDG	<a href="#">121</a>
PIE0	—	—	TMR0IE	IOCFE	—	—	—	INTE	<a href="#">122</a>
IOCAP	IOCAP7	IOCAP6	IOCAP5	IOCAP4	IOCAP3	IOCAP2	IOCAP1	IOCAP0	<a href="#">213</a>
IOCAN	IOCAN7	IOCAN6	IOCAN5	IOCAN4	IOCAN3	IOCAN2	IOCAN1	IOCAN0	<a href="#">213</a>
IOCAF	IOCAF7	IOCAF6	IOCAF5	IOCAF4	IOCAF3	IOCAF2	IOCAF1	IOCAF0	<a href="#">214</a>
IOCBP	IOCBP7	IOCBP6	IOCBP5	IOCBP4	IOCBP3	IOCBP2	IOCBP1	IOCBP0	<a href="#">215</a>
IOCBN	IOCBN7	IOCBN6	IOCBN5	IOCBN4	IOCBN3	IOCBN2	IOCBN1	IOCBN0	<a href="#">215</a>
IOCBF	IOCBF7	IOCBF6	IOCBF5	IOCBF4	IOCBF3	IOCBF2	IOCBF1	IOCBF0	<a href="#">216</a>
IOCCP	IOCCP7	IOCCP6	IOCCP5	IOCCP4	IOCCP3	IOCCP2	IOCCP1	IOCCP0	<a href="#">217</a>
IOCCN	IOCCN7	IOCCN6	IOCCN5	IOCCN4	IOCCN3	IOCCN2	IOCCN1	IOCCN0	<a href="#">217</a>
IOCCF	IOCCF7	IOCCF6	IOCCF5	IOCCF4	IOCCF3	IOCCF2	IOCCF1	IOCCF0	<a href="#">218</a>
IOCEP	—	—	—	—	IOCEP3	—	—	—	<a href="#">218</a>
IOCEN	—	—	—	—	IOCEN3	—	—	—	<a href="#">219</a>
IOCEF	—	—	—	—	IOCEF3	—	—	—	<a href="#">219</a>

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



## 18.0 FIXED VOLTAGE REFERENCE (FVR)

The Fixed Voltage Reference, or FVR, is a stable voltage reference, independent of  $V_{DD}$ , 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 and negative input
- Digital-to-Analog Converter (DAC)

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

**Note:** Fixed Voltage Reference output cannot exceed  $V_{DD}$ .

## 18.1 Independent Gain Amplifiers

The output of the FVR, which is connected 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 20.0 “Analog-to-Digital Converter \(ADC\) Module”](#) for additional information.

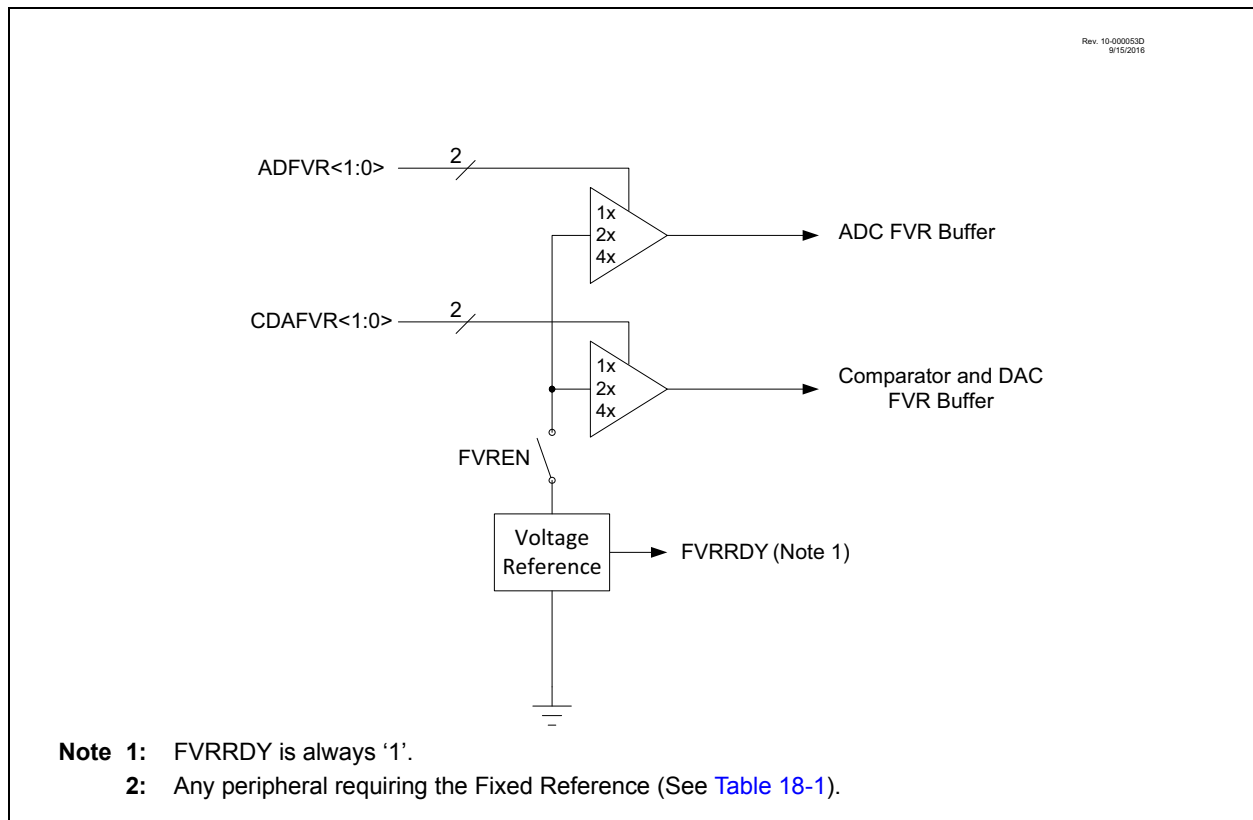
The CDAFVR<1:0> bits of the FVRCON register are used to enable and configure the gain amplifier settings for the reference supplied to the DAC and comparator module. Reference [Section 21.0 “5-Bit Digital-to-Analog Converter \(DAC1\) Module”](#) and [Section 23.0 “Comparator Module”](#) for additional information.

## 18.2 FVR Stabilization Period

When the Fixed Voltage Reference module is enabled, it requires time for the reference and amplifier circuits to stabilize.

FVRRDY is an indicator of the reference being ready. In the case of an LF device, or a device on which the BOR is enabled in the Configuration Word settings, then the FVRRDY bit will be high prior to setting FVREN as those module require the reference voltage.

**FIGURE 18-1: VOLTAGE REFERENCE BLOCK DIAGRAM**



## 18.3 Register Definitions: FVR Control

### REGISTER 18-1: 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 <sup>(1)</sup>	TSEN <sup>(3)</sup>	TSRNG <sup>(3)</sup>	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	<b>FVREN:</b> Fixed Voltage Reference Enable bit 1 = Fixed Voltage Reference is enabled 0 = Fixed Voltage Reference is disabled
bit 6	<b>FVRRDY:</b> Fixed Voltage Reference Ready Flag bit <sup>(1)</sup> 1 = Fixed Voltage Reference output is ready for use 0 = Fixed Voltage Reference output is not ready or not enabled
bit 5	<b>TSEN:</b> Temperature Indicator Enable bit <sup>(3)</sup> 1 = Temperature Indicator is enabled 0 = Temperature Indicator is disabled
bit 4	<b>TSRNG:</b> Temperature Indicator Range Selection bit <sup>(3)</sup> 1 = Temperature in High Range V <sub>OUT</sub> = 3VT 0 = Temperature in Low Range V <sub>OUT</sub> = 2VT
bit 3-2	<b>CDAFVR&lt;1:0&gt;:</b> Comparator FVR Buffer Gain Selection bits 11 =Comparator FVR Buffer Gain is 4x, (4.096V) <sup>(2)</sup> 10 =Comparator FVR Buffer Gain is 2x, (2.048V) <sup>(2)</sup> 01 =Comparator FVR Buffer Gain is 1x, (1.024V) 00 =Comparator FVR Buffer is off
bit 1-0	<b>ADFVR&lt;1:0&gt;:</b> ADC FVR Buffer Gain Selection bit 11 =ADC FVR Buffer Gain is 4x, (4.096V) <sup>(2)</sup> 10 =ADC FVR Buffer Gain is 2x, (2.048V) <sup>(2)</sup> 01 =ADC FVR Buffer Gain is 1x, (1.024V) 00 =ADC FVR Buffer is off

- Note 1:** FVRRDY is always '1'.  
**Note 2:** Fixed Voltage Reference output cannot exceed V<sub>DD</sub>.  
**Note 3:** See **Section 19.0 "Temperature Indicator Module"** for additional information.

**TABLE 18-1: 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>		<a href="#">222</a>
ADCON0	CHS<5:0>						GO/DONE	ADON	<a href="#">234</a>
ADCON1	ADFM	ADCS<2:0>			—	—	ADPREF<1:0>		<a href="#">235</a>
DAC1CON0	DAC1EN	—	DAC1OE1	DAC1OE2	DAC1PSS<1:0>		—	DAC1NSS	<a href="#">243</a>

**Legend:** — = unimplemented locations read as '0'. Shaded cells are not used with the Fixed Voltage Reference.

## 19.0 TEMPERATURE INDICATOR MODULE

This family of devices is equipped with a temperature circuit designed to measure the operating temperature of the silicon die.

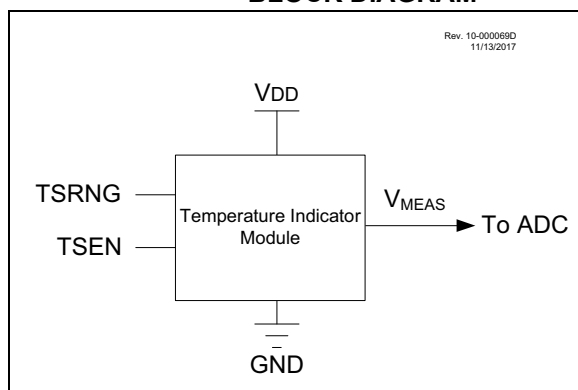
The circuit's range of operating temperature falls between  $-40^{\circ}\text{C}$  and  $+125^{\circ}\text{C}$ . A one-point calibration allows the circuit to indicate a temperature closely surrounding that point. A two-point calibration allows the circuit to sense the entire range of temperature more accurately.

### 19.1 Module Operation

The temperature indicator module consists of a temperature-sensing circuit that provides a voltage to the device ADC. The analog voltage output,  $V_{\text{MEAS}}$ , varies inversely to the device temperature. The output of the temperature indicator is referred to as  $V_{\text{MEAS}}$ .

Figure 19-1 shows a simplified block diagram of the temperature indicator module.

**FIGURE 19-1: TEMPERATURE INDICATOR MODULE BLOCK DIAGRAM**



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 20.0 “Analog-to-Digital Converter \(ADC\) Module”](#) for detailed information.

The ON/OFF bit for the module is located in the FVRCON register. See [Section 18.0 “Fixed Voltage Reference \(FVR\)”](#) for more information. The circuit is enabled by setting the TSEN bit of the FVRCON register. When the module is disabled, the circuit draws no current.

The circuit operates in either High or Low range. Refer to [Section 19.1.1 “Temperature Indicator Range”](#) for more details on the range settings.

### 19.1.1 TEMPERATURE INDICATOR RANGE

The temperature indicator 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. High range requires a higher-bias voltage to operate and thus, a higher  $V_{\text{DD}}$  is needed. The low range is selected by clearing the TSRNG bit of the FVRCON register. The low range generates a lower sensor voltage and thus, a lower  $V_{\text{DD}}$  voltage is needed to operate the circuit.

The output voltage of the sensor is the highest value at  $-40^{\circ}\text{C}$  and the lowest value at  $+125^{\circ}\text{C}$ .

- **High Range:** The High range is used in applications with the reference for the ADC,  $V_{\text{REF}} = 2.048\text{V}$ . This range may not be suitable for battery-powered applications. The ADC reading (in counts) at  $90^{\circ}\text{C}$  for the high range setting is stored in the DIA Table ([Table 6-1](#)) as parameter TSHR2.
- **Low Range:** This mode is useful in applications in which the  $V_{\text{DD}}$  is too low for high-range operation. The  $V_{\text{DD}}$  in this mode can be as low as 1.8V.  $V_{\text{DD}}$  must, however, be at least 0.5V higher than the maximum sensor voltage depending on the expected low operating temperature. The ADC reading (in counts) at  $90^{\circ}\text{C}$  for the Low range setting is stored in the DIA Table ([Table 6-1](#)) as parameter TSLR2.

### 19.1.2 MINIMUM OPERATING $V_{\text{DD}}$

When the temperature circuit is operated in low range, the device may be operated at any operating voltage that is within specifications. When the temperature circuit is operated in high range, the device operating voltage,  $V_{\text{DD}}$ , must be high enough to ensure that the temperature circuit is correctly biased.

[Table 19-1](#) shows the recommended minimum  $V_{\text{DD}}$  vs. Range setting.

**TABLE 19-1: RECOMMENDED  $V_{\text{DD}}$  vs. RANGE**

Min. $V_{\text{DD}}$ , TSRNG = 1 (High Range)	Min. $V_{\text{DD}}$ , TSRNG = 0 (Low Range)
$\geq 2.5$	$\geq 1.8$

## 19.2 Temperature Calculation

This section describes the steps involved in calculating the die temperature, TMEAS:

1. Obtain the ADC count value of the measured analog voltage: The analog output voltage, VMEAS is converted to a digital count value by the Analog to Digital Converter (ADC) and is referred to as ADCMEAS.
2. Obtain the ADC count value, ADCDIA at 90 degrees, in the DIA table (Table 6-1). This parameter is TSLR2 for the low range setting or TSHR2 for the high range setting of the temperature indicator module.
3. Obtain the output analog voltage (in mV) value of the Fixed Reference Voltage (FVR) for 2x setting, from the DIA Table. This parameter is FVRA2X in the DIA table (Table 5-3).
4. Obtain the value of the temperature indicator voltage sensitivity, parameter Mv, from Table 37-26 for the corresponding range setting.

Equation 19-1 provides an estimate for the die temperature based on the above parameters.

### EQUATION 19-1: SENSOR TEMPERATURE

$$T_{MEAS} = 90 + \frac{(ADC_{MEAS} - ADC_{DIA}) \times FVRA2X}{(2^N - 1) \times Mv}$$

Where:

ADCMEAS = ADC reading at temperature being estimated

ADCDIA = ADC reading stored in the DIA

FVRA2X = FVR value stored in the DIA for 2x setting

N = Resolution of the ADC

Mv = Temperature Indicator voltage sensitivity (mV/°C)

**Note 1:** It is recommended to take the average of ten measurements of ADCmeas to reduce noise and improve accuracy.

**2:** Refer to Section 37.0, Electrical Specifications for FVR reference voltage accuracy.

### 19.2.1 CALIBRATION

#### 19.2.1.1 Higher-Order Calibration

If the application requires more precise temperature measurement, additional calibrations steps will be necessary. For these applications, two-point or three-point calibration is recommended.

## 19.3 ADC Acquisition Time

To ensure accurate temperature measurements, the user must wait a certain minimum acquisition time (parameter TS01 in Table 37-26) for the ADC value to settle, after the ADC input multiplexer is connected to the temperature indicator output, before the conversion is performed.

**TABLE 19-2: SUMMARY OF REGISTERS ASSOCIATED WITH THE TEMPERATURE INDICATOR<sup>(1)</sup>**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
FVRCON	EN	RDY	TSEN	TSRNG	CDAFVR<1:0>		ADFVR<1:0>		222

**Legend:** — = Unimplemented location, read as '0'. Shaded cells are unused by the temperature indicator module.

**Note 1:** It is recommended to take the average of ten measurements of ADCMEAS to reduce noise and improve accuracy.

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

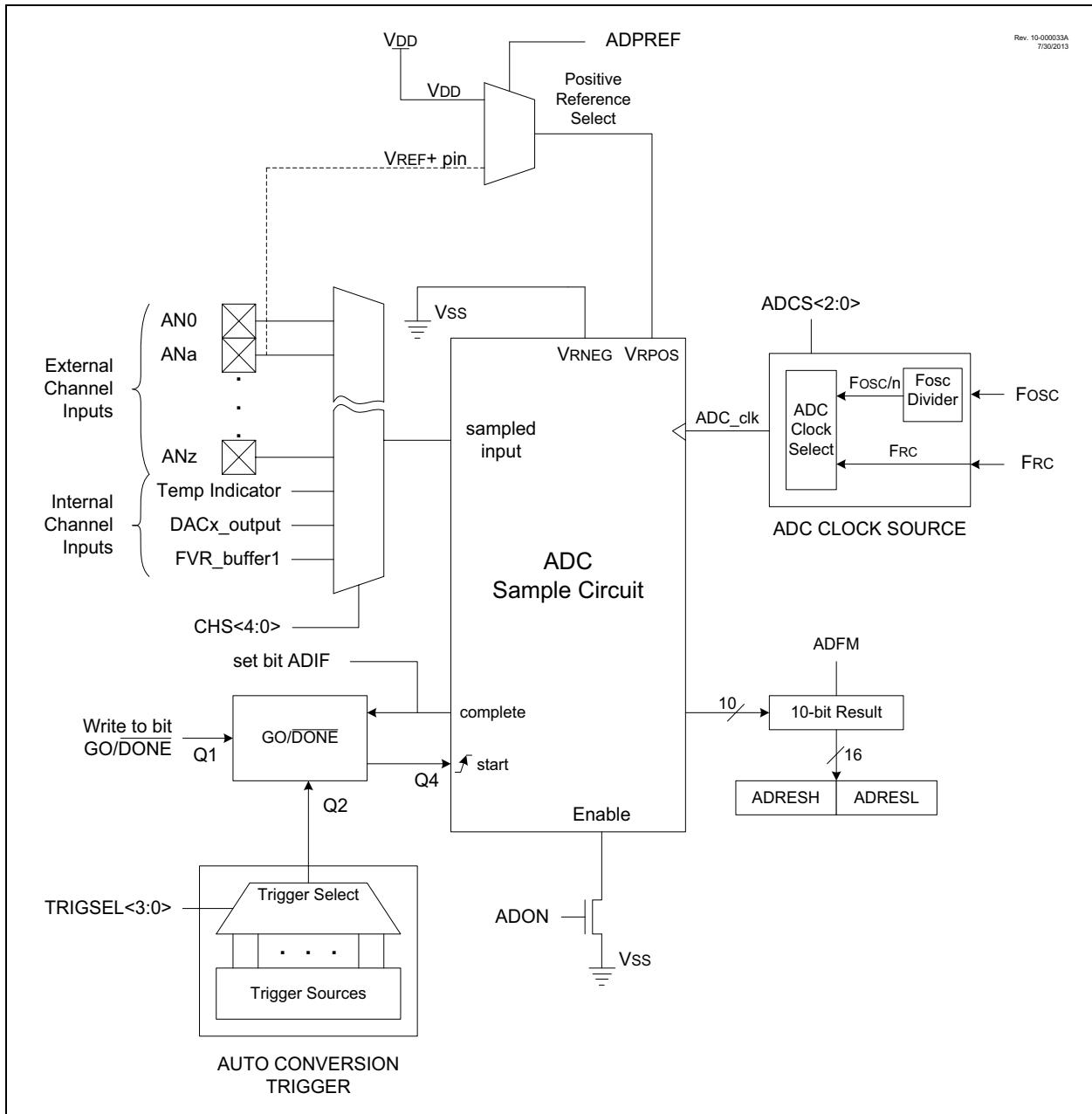
The Analog-to-Digital Converter (ADC) allows conversion of an analog input signal to a 10-bit binary representation of that signal. This device uses analog inputs, which are multiplexed into a single sample and hold circuit. The output of the sample and hold is connected to the input of the converter. 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 20-1 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 20-1: ADC BLOCK DIAGRAM**



## 20.1 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

### 20.1.1 PORT CONFIGURATION

The ADC can be used to convert both analog and digital signals. When converting analog signals, the I/O pin will be configured for analog by setting the associated TRIS and ANSEL bits. Refer to [Section 14.0 “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.

### 20.1.2 CHANNEL SELECTION

There are several channel selections available:

- Seven Port A channels
- Seven Port B channels
- Seven Port C channels
- Temperature Indicator
- DAC output
- Fixed Voltage Reference (FVR)
- AVss (Ground)

The CHS<5:0> bits of the ADCON0 register ([Register 20-1](#)) 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 20.2 “ADC Operation”](#) for more information.

**Note:** It is recommended that when switching from an ADC channel of a higher voltage to a channel of a lower voltage, that the user selects the Vss channel before connecting to the channel with the lower voltage. If the ADC does not have a dedicated Vss input channel, the Vss selection (DAC1R<4:0> = b'00000') through the DAC output channel can be used. If the DAC is in use, a free input channel can be connected to Vss, and can be used in place of the DAC.

### 20.1.3 ADC VOLTAGE REFERENCE

The ADPREF<1:0> bits of the ADCON1 register provides 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)

The ADPREF bit of the ADCON1 register provides control of the negative voltage reference. The negative voltage reference can be:

- VREF- pin
- Vss

See [Section 18.0 “Fixed Voltage Reference \(FVR\)”](#) for more details on the Fixed Voltage Reference.

### 20.1.4 CONVERSION CLOCK

The source of the conversion clock is software selectable via the ADCS<2:0> bits of the ADCON1 register. There are seven possible clock options:

- Fosc/2
- Fosc/4
- Fosc/8
- Fosc/16
- Fosc/32
- Fosc/64
- ADCRC (dedicated 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 20-2](#).

For correct conversion, the appropriate TAD specification must be met. Refer to [Table 37-13](#) for more information. [Table 20-1](#) gives examples of appropriate ADC clock selections.

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

**TABLE 20-1: ADC CLOCK PERIOD (TAD) Vs. DEVICE OPERATING FREQUENCIES**

ADC Clock Period (TAD)		Device Frequency (Fosc)					
ADC Clock Source	ADCS<2:0>	32 MHz	20 MHz	16 MHz	8 MHz	4 MHz	1 MHz
Fosc/2	000	62.5ns <sup>(2)</sup>	100 ns <sup>(2)</sup>	125 ns <sup>(2)</sup>	250 ns <sup>(2)</sup>	500 ns <sup>(2)</sup>	2.0 μs
Fosc/4	100	125 ns <sup>(2)</sup>	200 ns <sup>(2)</sup>	250 ns <sup>(2)</sup>	500 ns <sup>(2)</sup>	1.0 μs	4.0 μs
Fosc/8	001	0.5 μs <sup>(2)</sup>	400 ns <sup>(2)</sup>	0.5 μs <sup>(2)</sup>	1.0 μs	2.0 μs	8.0 μs <sup>(3)</sup>
Fosc/16	101	800 ns	800 ns	1.0 μs	2.0 μs	4.0 μs	16.0 μs <sup>(3)</sup>
Fosc/32	010	1.0 μs	1.6 μs	2.0 μs	4.0 μs	8.0 μs <sup>(3)</sup>	32.0 μs <sup>(2)</sup>
Fosc/64	110	2.0 μs	3.2 μs	4.0 μs	8.0 μs <sup>(3)</sup>	16.0 μs <sup>(2)</sup>	64.0 μs <sup>(2)</sup>
ADCRC	x11	1.0-6.0 μs <sup>(1,4)</sup>	1.0-6.0 μs <sup>(1,4)</sup>	1.0-6.0 μs <sup>(1,4)</sup>	1.0-6.0 μs <sup>(1,4)</sup>	1.0-6.0 μs <sup>(1,4)</sup>	1.0-6.0 μs <sup>(1,4)</sup>

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

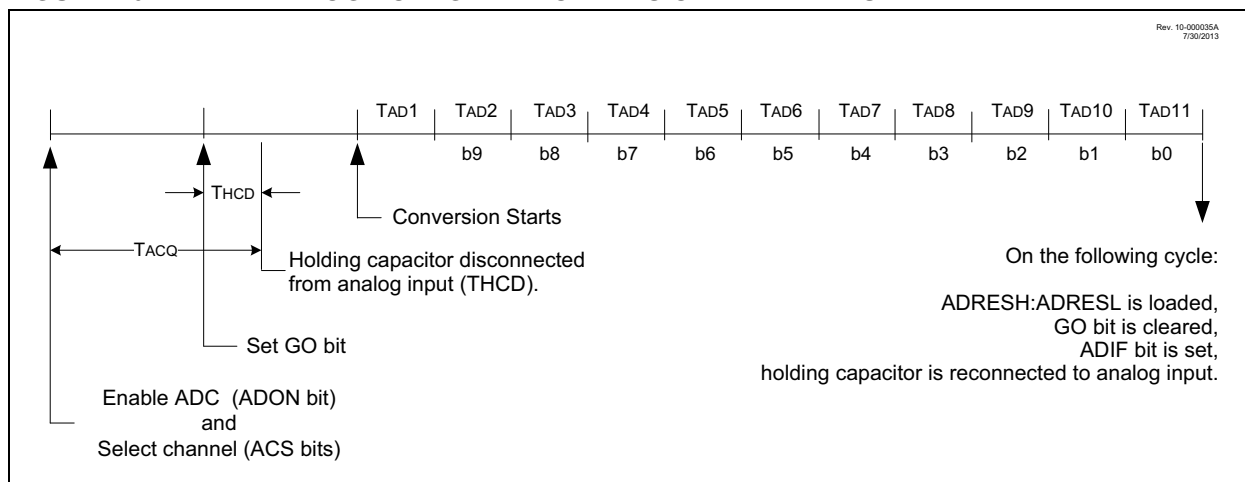
**Note 1:** See TAD parameter for ADCRC source typical TAD value.

**2:** These values violate the required TAD time.

**3:** Outside the recommended TAD time.

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

**FIGURE 20-2: ANALOG-TO-DIGITAL CONVERSION TAD CYCLES**





## 20.1.5 INTERRUPTS

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

**Note 1:** The ADIF bit is set at the completion of every conversion, regardless of whether or not the ADC interrupt is enabled.

**2:** The ADC operates during Sleep only when the ADCRC oscillator is selected.

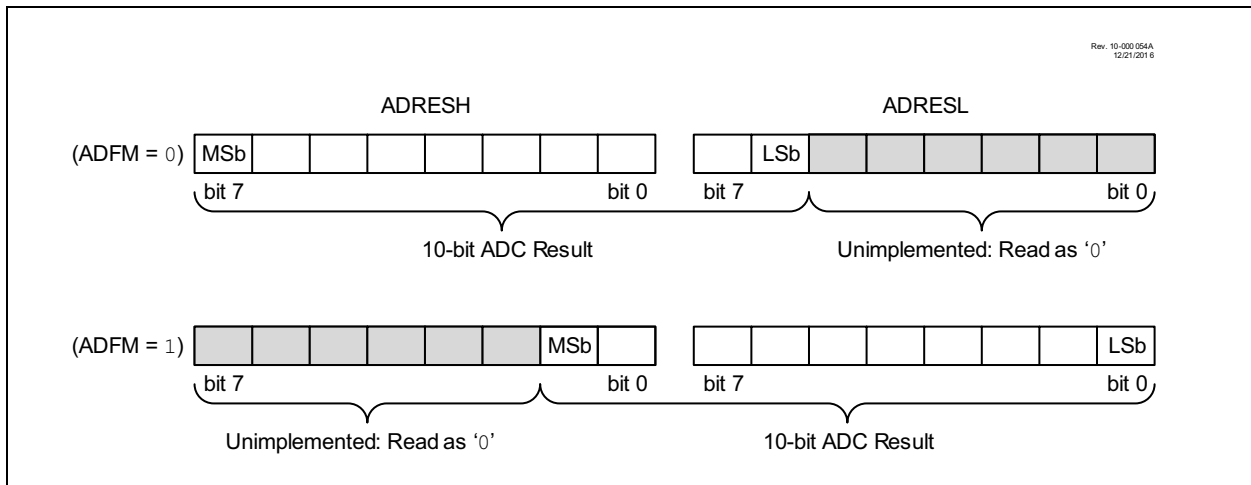
This interrupt can be generated while the device is operating or while in Sleep. If the device is in Sleep, the interrupt will wake-up the device. Upon waking from Sleep, the next instruction following the SLEEP instruction is always executed. If the user is attempting to wake-up from Sleep and resume 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 (ISR).

## 20.1.6 RESULT FORMATTING

The 10-bit ADC conversion result can be supplied in two formats, left justified or right justified. The ADFM bit of the ADCON1 register controls the output format.

Figure 20-3 shows the two output formats.

**FIGURE 20-3: 10-BIT ADC CONVERSION RESULT FORMAT**



## 20.2 ADC Operation

### 20.2.1 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 will not be set in the same instruction that turns on the ADC. Refer to [Section 20.2.5 “ADC Conversion Procedure”](#).

### 20.2.2 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

**Note:** A device Reset forces all registers to their Reset state. Thus, the ADC module is turned off and any pending conversion is terminated.

### 20.2.3 ADC OPERATION DURING SLEEP

The ADC module can operate during Sleep. This requires the ADC clock source to be set to the ADCRC option. When the ADCRC 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 ADCRC, a SLEEP instruction causes the present conversion to be aborted and the ADC module is turned off, although the ADON bit remains set.

### 20.2.4 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 ADOACT<4:0> bits of the ADOACT 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 20-2](#) for auto-conversion sources.

**Note:** The Auto-conversion feature is not available while the device is in Sleep mode.

**TABLE 20-2: ADC AUTO-CONVERSION TABLE**

ADACT VALUE	SOURCE/ PERIPHERAL	DESCRIPTION
0x00	Disabled	External Trigger Disabled
0x01	ADACTPPS	Pin Selected by ADACTPPS
0x02	TMR0	Timer0 overflow condition
0x03	TMR1	Timer1 overflow condition
0x04	TMR2	Match between Timer2 postscaled value and PR2
0x05	CCP1	CCP1 output
0x06	CCP2	CCP2 output
0x07	PWM3	PWM3 output
0x08	PWM4	PWM4 output
0x09	PWM5	PWM5 output
0x0A	PWM6	PWM6 output
0x0B	NCO1	NCO1 output
0x0C	C1OUT	Comparator C1 output
0x0D	C2OUT	Comparator C2 output
0x0E	IOCIF	Interrupt-on change flag trigger
0x0F	CLC1	CLC1 output
0x10	CLC2	CLC2 output
0x11	CLC3	CLC3 output
0x12	CLC4	CLC4 output
0x13-0xFF	Reserved	Reserved, do not use

## 20.2.5 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)
2. Configure the ADC module:
  - Select ADC conversion clock
  - Select voltage reference
  - Select ADC input channel
  - Turn on ADC module
3. Configure ADC interrupt (optional):
  - Clear ADC interrupt flag
  - Enable ADC interrupt
  - Enable peripheral interrupt
  - Enable global interrupt<sup>(1)</sup>
4. Wait the required acquisition time<sup>(2)</sup>.
5. Start conversion by setting the GO/DONE bit.
6. Wait for ADC conversion to complete by one of the following:
  - Polling the GO/DONE bit
  - Waiting for the ADC interrupt
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 20.3 “ADC Acquisition Requirements”](#).

## EXAMPLE 20-1: ADC CONVERSION

```
;This code block configures the ADC
;for polling, Vdd and Vss references, ADCRC
;oscillator and AN0 input.
;
;Conversion start & polling for completion ;
are included.
;
BANKSELADCON1;
MOVLWB'11110000';Right justify, ADCRC
;oscillator
MOVWFADCON1;Vdd and Vss Vref
BANKSELTRISA;
BSF TRISA,0;Set RA0 to input
BANKSELANSELA;
BSF ANSELA,0;Set RA0 to analog
BANKSELADCON0;
MOVLWB'00000001';Select channel AN0
MOVWFADCON0;Turn ADC On
CALLSampleTime;Acquisiton delay
BSF ADCON0,ADGO;Start conversion
BTFSCADCON0,ADGO;Is conversion done?
GOTO$-1 ;No, test again
BANKSELADRESH;
MOVFADRESH,W;Read upper 2 bits
MOVWFRESULTHI;store in GPR space
BANKSELADRESL;
MOVFADRESL,W;Read lower 8 bits
MOVWFRESULTLO;Store in GPR space
```

## 20.3 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 20-4. 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 20-4. **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 20-1 may be used. This equation assumes that 1/2 LSB error is used (1024 steps for the ADC). The 1/2 LSB error is the maximum error allowed for the ADC to meet its specified resolution.

### EQUATION 20-1: ACQUISITION TIME EXAMPLE

*Assumptions: Temperature = 50°C and external impedance of 10kΩ 5.0V VDD*

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

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

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

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

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

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

*Solving for TC:*

$$\begin{aligned} T_C &= -CHOLD(RIC + RSS + RS) \ln(1/2047) \\ &= -10pF(1k\Omega + 7k\Omega + 10k\Omega) \ln(0.0004885) \\ &= 1.37\mu s \end{aligned}$$

*Therefore:*

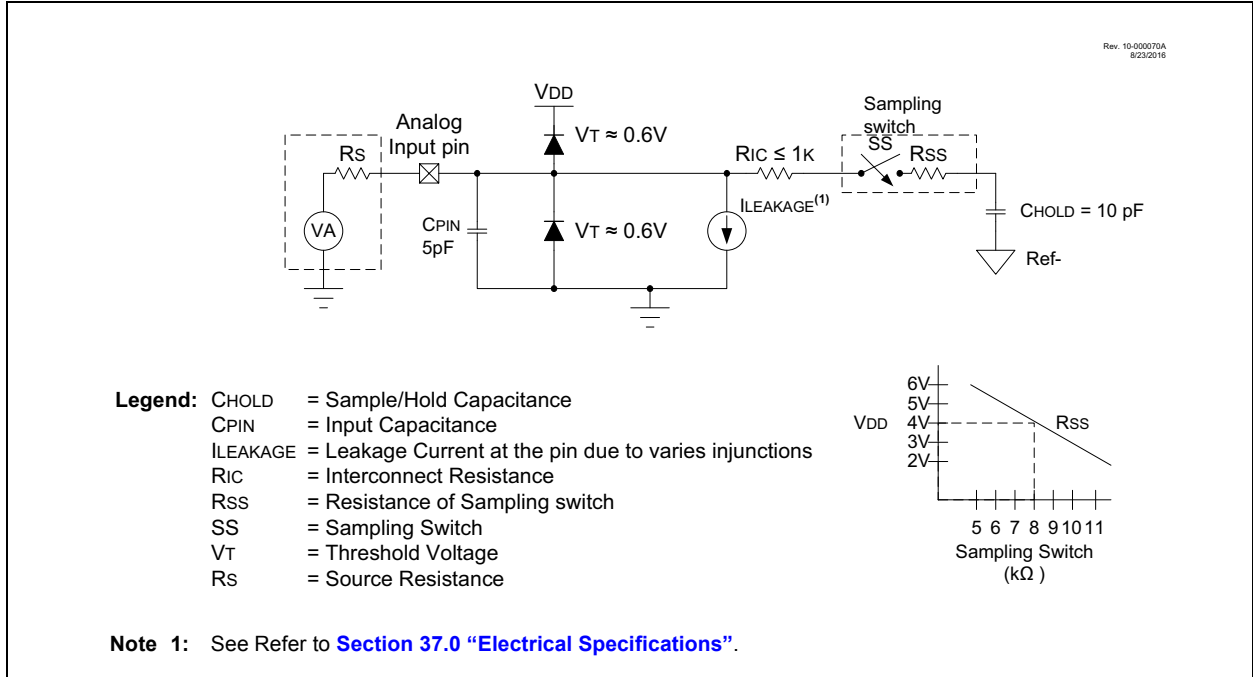
$$\begin{aligned} T_{ACQ} &= 2\mu s + 1.37 + [(50^\circ C - 25^\circ C)(0.05\mu s/^\circ C)] \\ &= 4.62\mu s \end{aligned}$$

**Note 1:** The VAPPLIED has no effect on the equation, since it cancels itself out.

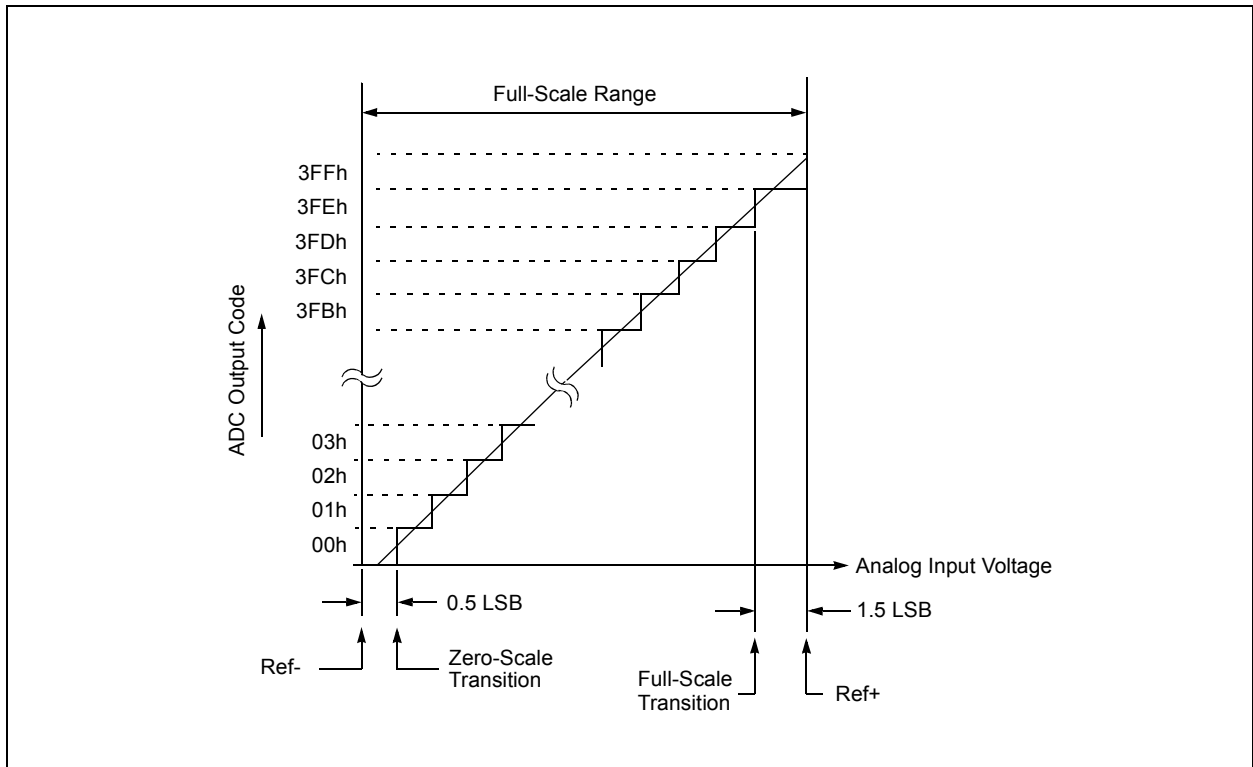
**2:** The charge holding capacitor (CHOLD) is not discharged after each conversion.

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

**FIGURE 20-4: ANALOG INPUT MODEL**



**FIGURE 20-5: ADC TRANSFER FUNCTION**



## 20.4 Register Definitions: ADC Control

### REGISTER 20-1: 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/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 Buffer 2 reference voltage<sup>(2)</sup>  
 111110 = FVR 1 Buffer 1 reference voltage<sup>(2)</sup>  
 111101 = DAC1 output voltage<sup>(1)</sup>  
 111100 = Temperature sensor output<sup>(3)</sup>  
 111011 = AVss (Analog Ground)  
 111010-011000 = Reserved. No channel connected  
 .  
 .  
 .  
 010111 = RC7  
 010110 = RC6  
 010101 = RC5  
 010100 = RC4  
 010011 = RC3  
 010010 = RC2  
 010001 = RC1  
 010000 = RC0  
 001111 = RB7  
 001110 = RB6  
 001101 = RB5  
 001100 = RB4  
 001011 = RB3  
 001010 = RB2  
 001001 = RB1  
 001000 = RB0  
 001011 = RA7<sup>(4)</sup>  
 000101 = RA5  
 000100 = RA4  
 000011 = RA3  
 000010 = RA2  
 000001 = RA1  
 000000 = RA0

bit 1      **GO/DONE**: ADC Conversion Status bit  
 1 = ADC conversion cycle in progress. Setting this bit starts an ADC conversion cycle.  
 This bit is automatically cleared by hardware when the ADC conversion has completed.  
 0 = ADC conversion completed/not in progress

bit 0      **ADON**: ADC Enable bit  
 1 = ADC is enabled  
 0 = ADC is disabled and consumes no operating current

- Note**
- 1: See [Section 21.0 "5-Bit Digital-to-Analog Converter \(DAC1\) Module"](#) for more information.
  - 2: See [Section 18.0 "Fixed Voltage Reference \(FVR\)"](#) for more information.
  - 3: See [Section 19.0 "Temperature Indicator Module"](#) for more information.
  - 4: The analog channel functionality on these pins is disabled when the system clock source is selected is external.

## REGISTER 20-2: ADCON1: ADC CONTROL REGISTER 1

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	U-0	R/W-0/0	R/W-0/0
ADFM	ADCS<2:0>		—	—	ADPREF<1:0>		
bit 7						bit 0	

### Legend:

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

- bit 7      **ADFM:** ADC Result Format Select bit  
 1 = Right justified. Six Most Significant bits of ADRESH are set to '0' when the conversion result is loaded.  
 0 = Left justified. Six Least Significant bits of ADRESL are set to '0' when the conversion result is loaded.
- bit 6-4    **ADCS<2:0>:** ADC Conversion Clock Select bits  
 111 =ADCRC (dedicated RC oscillator)  
 110 =Fosc/64  
 101 =Fosc/16  
 100 =Fosc/4  
 011 =ADCRC (dedicated RC oscillator)  
 010 =Fosc/32  
 001 =Fosc/8  
 000 =Fosc/2
- bit 3-2    **Unimplemented:** Read as '0'
- bit 1-0    **ADPREF<1:0>:** ADC Positive Voltage Reference Configuration bits  
 11 =VREF+ is connected to internal Fixed Voltage Reference (FVR) module<sup>(1)</sup>  
 10 =VREF+ is connected to external VREF+ pin<sup>(1)</sup>  
 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 37-14](#) for details.

## REGISTER 20-3: A/D AUTO-CONVERSION TRIGGER

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	
—	—	—	ADACT<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

**ADACT<4:0>:** Auto-Conversion Trigger Selection bits<sup>(1)</sup> (see [Table 20-2](#))

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



**REGISTER 20-4: 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 20-5: 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 20-6: 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 20-7: 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

**TABLE 20-3: 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
INTCON	GIE	PEIE	—	—	—	—	—	INTEDG	<a href="#">121</a>
PIE1	OSFIE	CSWIE	—	—	—	—	—	ADIE	<a href="#">123</a>
PIR1	OSFIF	CSWIF	—	—	—	—	—	ADIF	<a href="#">131</a>
TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	<a href="#">175</a>
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	<a href="#">181</a>
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	<a href="#">186</a>
ANSELA	ANSA7	ANSA6	ANSA5	ANSA4	ANSA3	ANSA2	ANSA1	ANSA0	<a href="#">176</a>
ANSELB	ANSB7	ANSB6	ANSB5	ANSB4	ANSB3	ANSB2	ANSB1	ANSB0	<a href="#">182</a>
ANSELC	ANSC7	ANSC6	ANSC5	ANSC4	ANSC3	ANSC2	ANSC1	ANSC0	<a href="#">187</a>
ADCON0	CHS<5:0>						GO/DONE	ADON	<a href="#">234</a>
ADCON1	ADFM	ADCS<2:0>			—	—	ADPREF<1:0>		<a href="#">235</a>
ADACT	—	—	—	ADACT<4:0>					<a href="#">236</a>
ADRESH	ADRESH<7:0>								<a href="#">237</a>
ADRESL	ADRESL<7:0>								<a href="#">237</a>
FVRCON	FVREN	FVRRDY	TSEN	TSRNG	CDAFVR<1:0>		ADFVR<1:0>		<a href="#">222</a>
DAC1CON1	—	—	—	DAC1R<4:0>					<a href="#">243</a>
OSCSTAT1	EXTOR	HFOR	MFOR	LFOR	SOR	ADOR	—	PLLR	<a href="#">112</a>

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

## 21.0 5-BIT DIGITAL-TO-ANALOG CONVERTER (DAC1) MODULE

The Digital-to-Analog Converter supplies a variable voltage reference, ratiometric with the input source, with 32 selectable output levels.

The input of the DAC can be connected to:

- External VREF pins
- VDD supply voltage
- FVR (Fixed Voltage Reference)

The output of the DAC can be configured to supply a reference voltage to the following:

- Comparator positive input
- ADC input channel
- DAC1OUT pin

The Digital-to-Analog Converter (DAC) is enabled by setting the DAC1EN bit of the DAC1CON0 register.

## 21.1 Output Voltage Selection

The DAC has 32 voltage level ranges. The 32 levels are set with the DAC1R<4:0> bits of the DAC1CON1 register.

The DAC output voltage is determined by [Equation 21-1](#):

### EQUATION 21-1: DAC OUTPUT VOLTAGE

$$V_{OUT} = \left\{ (V_{SOURCE+} - V_{SOURCE-}) \right\} \frac{DAC1R\langle 4:0 \rangle}{2^5} + V_{SOURCE-}$$

$$V_{SOURCE+} = V_{DD} \text{ or } V_{REF+} \text{ or } FVR$$

$$V_{SOURCE-} = V_{SS} \text{ or } V_{REF-}$$

## 21.2 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.

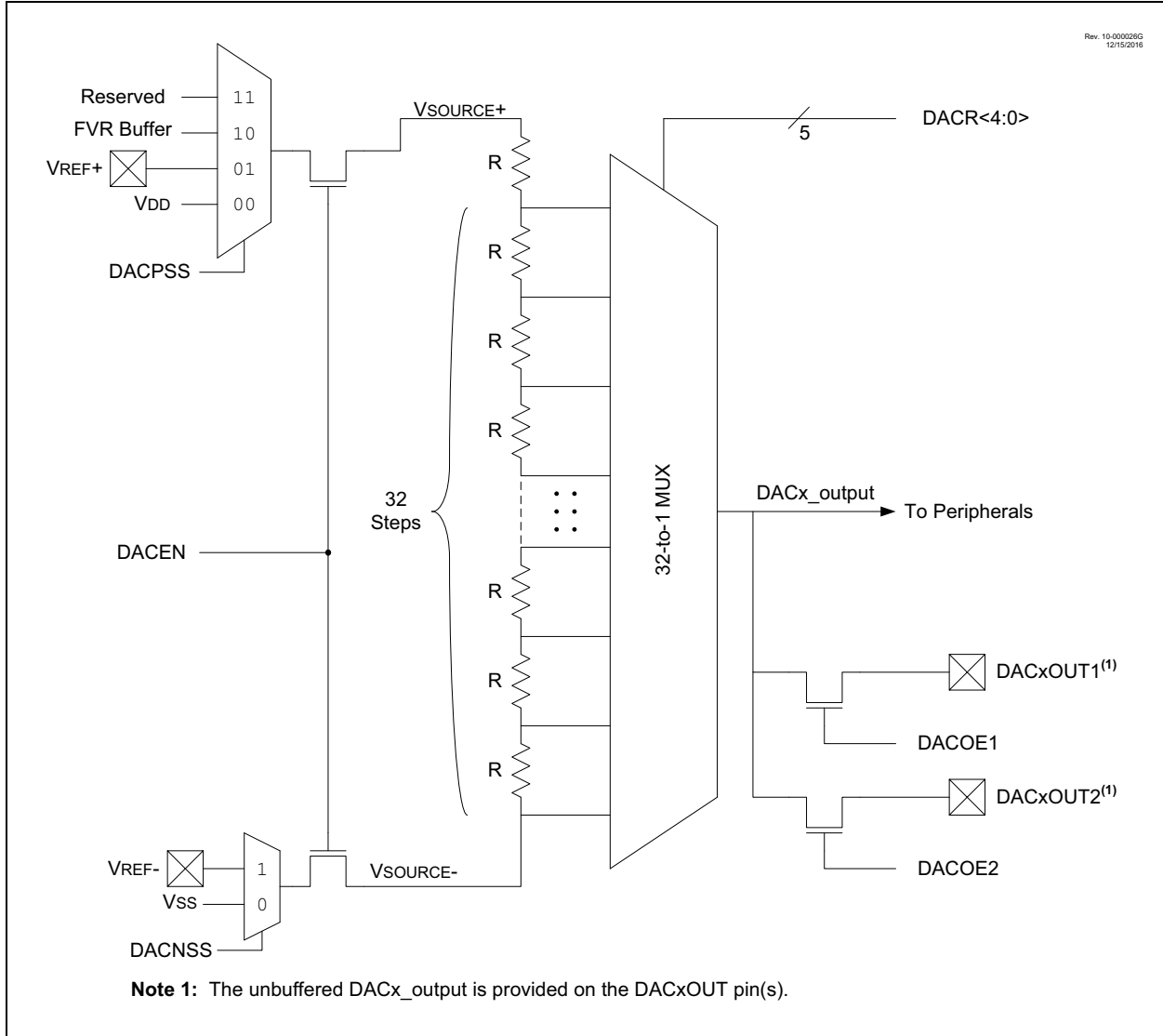
The value of the individual resistors within the ladder can be found in [Table 37-15](#).

## 21.3 DAC Voltage Reference Output

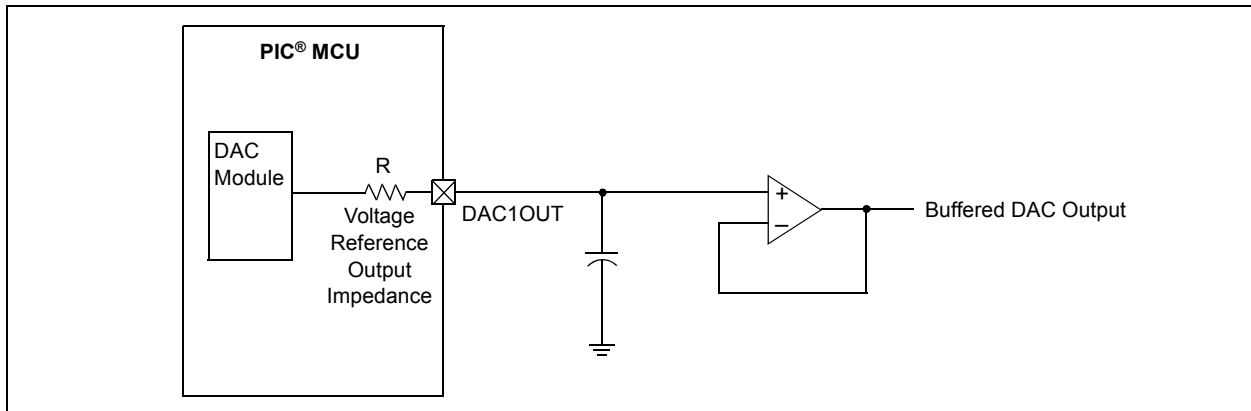
The DAC voltage can be output to the DAC1OUT1/2 pins by setting the DAC1OE1/2 bits of the DAC1CON0 register, respectively. Selecting the DAC reference voltage for output on the DAC1OUT1/2 pins automatically overrides the digital output buffer and digital input threshold detector functions, disables the weak pull-up, and disables the current-controlled drive function of that pin. Reading the DAC1OUT1/2 pin when it has been configured for DAC reference voltage output will always return a '0'.

Due to the limited current drive capability, a buffer must be used on the DAC voltage reference output for external connections to the DAC1OUT1/2 pins. [Figure 21-2](#) shows an example buffering technique.

**FIGURE 21-1: DIGITAL-TO-ANALOG CONVERTER BLOCK DIAGRAM**



**FIGURE 21-2: VOLTAGE REFERENCE OUTPUT BUFFER EXAMPLE**



## 21.4 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 DAC1CON0 register are not affected.

## 21.5 Effects of a Reset

A device Reset affects the following:

- DAC is disabled.
- DAC output voltage is removed from the DAC1OUT1/2 pins.
- The DAC1R<4:0> range select bits are cleared.

## 21.6 Register Definitions: DAC Control

### REGISTER 21-1: DAC1CON0: VOLTAGE REFERENCE 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
DAC1EN	—	DAC1OE1	DAC1OE2	DAC1PSS<1:0>		—	DAC1NSS
bit 7							bit 0

#### Legend:

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

bit 7      **DAC1EN:** DAC1 Enable bit  
1 = DAC is enabled  
0 = DAC is disabled

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

bit 5      **DAC1OE1:** DAC1 Voltage Output 1 Enable bit  
1 = DAC voltage level is an output on the DAC1OUT1 pin  
0 = DAC voltage level is disconnected from the DAC1OUT1 pin

bit 4      **DAC1OE2:** DAC1 Voltage Output 2 Enable bit  
1 = DAC voltage level is an output on the DAC1OUT2 pin  
0 = DAC voltage level is disconnected from the DAC1OUT2 pin

bit 3-2    **DAC1PSS<1:0>:** DAC1 Positive Source Select bits  
11 =Reserved, do not use  
10 =FVR output  
01 =VREF+ pin  
00 =VDD

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

bit 0      **DAC1NSS:** Read as '0'

### REGISTER 21-2: DAC1CON1: VOLTAGE REFERENCE CONTROL REGISTER 1

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	DAC1R<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    **DAC1R<4:0>:** DAC1 Voltage Output Select bits  
 $V_{OUT} = (V_{SRC+} - V_{SRC-}) * (DAC1R<4:0> / 32) + V_{SRC}$

**TABLE 21-1: SUMMARY OF REGISTERS ASSOCIATED WITH THE DAC1 MODULE**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
DAC1CON0	DAC1EN	—	DAC1OE1	DAC1OE2	DAC1PSS<1:0>		—	DAC1NSS	<a href="#">243</a>
DAC1CON1	—	—	—	DAC1R<4:0>					<a href="#">243</a>
CM1PSEL	—	—	—	—	—	PCH<2:0>			<a href="#">263</a>
CM2PSEL	—	—	—	—	—	PCH<2:0>			<a href="#">263</a>

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



## 22.0 NUMERICALLY CONTROLLED OSCILLATOR (NCO) MODULE

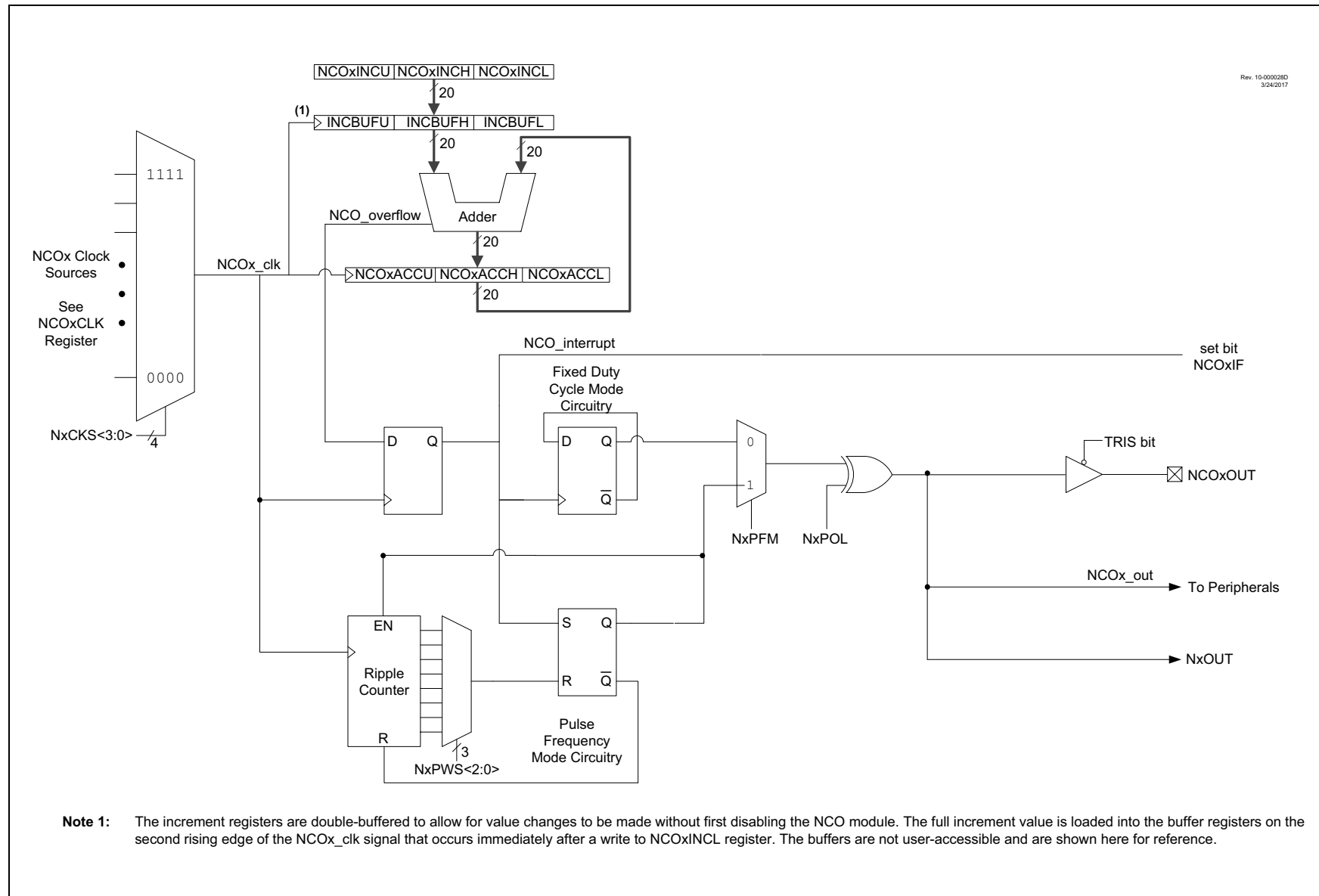
The Numerically Controlled Oscillator (NCO) module is a timer that uses overflow from the addition of an increment value to divide the input frequency. The advantage of the addition method over simple counter driven timer is that the output frequency resolution does not vary with the divider value. The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

Features of the NCO include:

- 20-bit Increment Function
- Fixed Duty Cycle mode (FDC) mode
- Pulse Frequency (PF) mode
- Output Pulse Width Control
- Multiple Clock Input Sources
- Output Polarity Control
- Interrupt Capability

Figure 22-1 is a simplified block diagram of the NCO module.

**FIGURE 22-1: NUMERICALLY CONTROLLED OSCILLATOR MODULE SIMPLIFIED BLOCK DIAGRAM**



Rev. 10-000280  
3/24/2017

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

## 22.1 NCO OPERATION

The NCO operates by repeatedly adding a fixed value to an accumulator. Additions occur at the input clock rate. The accumulator will overflow with a carry periodically, which is the raw NCO output (NCO\_overflow). This effectively reduces the input clock by the ratio of the addition value to the maximum accumulator value. See [Equation 22-1](#).

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

The NCO period changes in discrete steps to create an average frequency.

### EQUATION 22-1: NCO OVERFLOW FREQUENCY

$$F_{OVERFLOW} = \frac{NCO \text{ Clock Frequency} \times \text{Increment Value}}{2^{20}}$$

#### 22.1.1 NCO CLOCK SOURCES

Clock sources available to the NCO include:

- HFINTOSC
- Fosc
- LC1\_out
- LC2\_out
- LC3\_out
- LC4\_out
- MFINTOSC (500 kHz)
- MFINTOSC (32 kHz)
- SOSC
- CLKR

The NCO clock source is selected by configuring the N1CKS<2:0> bits in the NCO1CLK register.

#### 22.1.2 ACCUMULATOR

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

- NCO1ACCL
- NCO1ACCH
- NCO1ACCU

#### 22.1.3 ADDER

The NCO Adder is a full adder, which operates synchronously from the source clock. The addition of the previous result and the increment value replaces the accumulator value on the rising edge of each input clock.

#### 22.1.4 INCREMENT REGISTERS

The increment value is stored in three registers making up a 20-bit incrementer. In order of LSB to MSB they are:

- NCO1INCL
- NCO1INCH
- NCO1INCU

When the NCO module is enabled, the NCO1INCU and NCO1INCH registers should be written first, then the NCO1INCL register. Writing to the NCO1INCL register initiates the increment buffer registers to be loaded simultaneously on the second rising edge of the NCO\_clk signal.

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

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

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

## 22.2 FIXED DUTY CYCLE MODE

In Fixed Duty Cycle (FDC) mode, every time the accumulator overflows (NCO\_overflow), the output is toggled at a frequency rate half of the F\_OVERFLOW. This provides a 50% duty cycle, provided that the increment value remains constant. For more information, see [Figure 22-2](#).

The FDC mode is selected by clearing the N1PFM bit in the NCO1CON register.

## 22.3 PULSE FREQUENCY MODE

In Pulse Frequency (PF) mode, every time the Accumulator overflows, the output becomes active for one or more clock periods. Once the clock period expires, the output returns to an inactive state. This provides a pulsed output. The output becomes active on the rising clock edge immediately following the overflow event. For more information, see [Figure 22-2](#).

The value of the active and inactive states depends on the polarity bit, N1POL in the NCO1CON register.

The PF mode is selected by setting the N1PFM bit in the NCO1CON register.

### 22.3.1 OUTPUT PULSE WIDTH CONTROL

When operating in PF mode, the active state of the output can vary in width by multiple clock periods. Various pulse widths are selected with the N1PWS<2:0> bits in the NCO1CLK register.

When the selected pulse width is greater than the Accumulator overflow time frame, then NCO1 output does not toggle.

## 22.4 OUTPUT POLARITY CONTROL

The last stage in the NCO module is the output polarity. The N1POL bit in the NCO1CON register selects the output polarity. Changing the polarity while the interrupts are enabled will cause an interrupt for the resulting output transition.

The NCO output signal (NCO1\_out) is available to the following peripherals:

- CLC
- CWG
- Timer1
- Timer2
- CLKR

## 22.5 Interrupts

When the accumulator overflows (NCO\_overflow), the NCO Interrupt Flag bit, NCO1IF, of the PIR7 register is set. To enable the interrupt event (NCO\_interrupt), the following bits must be set:

- N1EN bit of the NCO1CON register
- NCO1IE bit of the PIE7 register
- PEIE bit of the INTCON register
- GIE bit of the INTCON register

The interrupt must be cleared by software by clearing the NCO1IF bit in the Interrupt Service Routine.

## 22.6 Effects of a Reset

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

## 22.7 Operation in Sleep

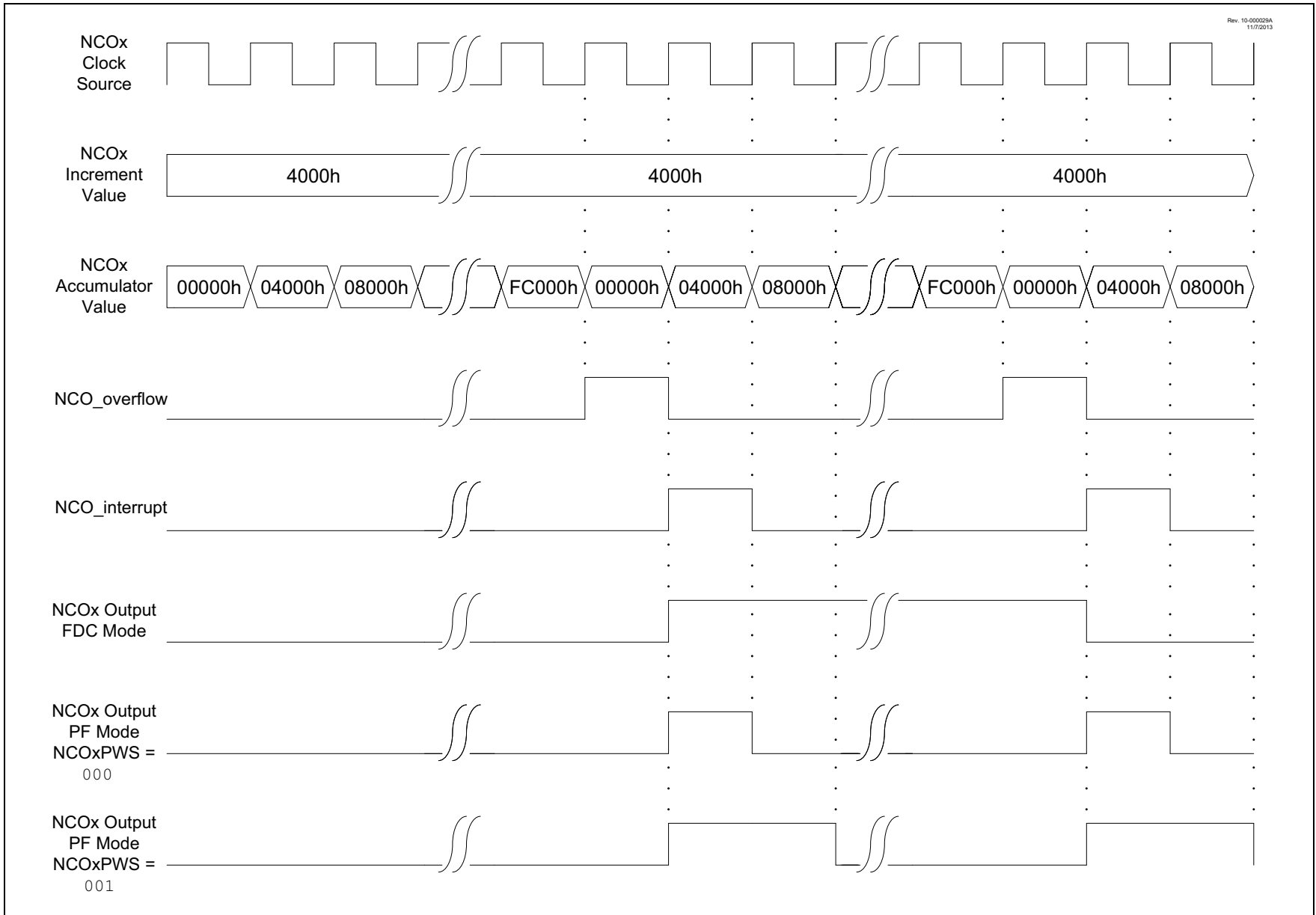
The NCO module operates independently from the system clock and will continue to run during Sleep, provided that the clock source selected remains active.

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

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

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

**FIGURE 22-2: FDC OUTPUT MODE OPERATION DIAGRAM**



## 22.8 NCO Control Registers

**REGISTER 22-1: NCO1CON: NCO CONTROL REGISTER**

R/W-0/0	U-0	R-0/0	R/W-0/0	U-0	U-0	U-0	R/W-0/0
N1EN	—	N1OUT	N1POL	—	—	—	N1PFM
bit 7							bit 0

**Legend:**

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

u = Bit is unchanged

x = Bit is unknown

-n/n = Value at POR and BOR/Value at all other Resets

'1' = Bit is set

'0' = Bit is cleared

- bit 7      **N1EN:** NCO1 Enable bit  
           1 = NCO1 module is enabled  
           0 = NCO1 module is disabled
- bit 6      **Unimplemented:** Read as '0'
- bit 5      **N1OUT:** NCO1 Output bit  
           Displays the current output value of the NCO1 module.
- bit 4      **N1POL:** NCO1 Polarity bit  
           1 = NCO1 output signal is inverted  
           0 = NCO1 output signal is not inverted
- bit 3-1    **Unimplemented:** Read as '0'
- bit 0      **N1PFM:** NCO1 Pulse Frequency Mode bit  
           1 = NCO1 operates in Pulse Frequency mode  
           0 = NCO1 operates in Fixed Duty Cycle mode, divide by 2

# PIC16(L)F15354/55

## REGISTER 22-2: NCO1CLK: NCO1 INPUT CLOCK CONTROL REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
N1PWS<2:0> <sup>(1,2)</sup>			—	N1CKS<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 7-5 **N1PWS<2:0>**: NCO1 Output Pulse Width Select bits<sup>(1)</sup>

- 111 = NCO1 output is active for 128 input clock periods
- 110 = NCO1 output is active for 64 input clock periods
- 101 = NCO1 output is active for 32 input clock periods
- 100 = NCO1 output is active for 16 input clock periods
- 011 = NCO1 output is active for 8 input clock periods
- 010 = NCO1 output is active for 4 input clock periods
- 001 = NCO1 output is active for 2 input clock periods
- 000 = NCO1 output is active for 1 input clock period

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

bit 3-0 **N1CKS<3:0>**: NCO1 Clock Source Select bits

- 1011-1111 = Reserved
- 1010 = LC4\_out
- 1001 = LC3\_out
- 1000 = LC2\_out
- 0111 = LC1\_out
- 0110 = CLKR
- 0101 = SOSC
- 0100 = MFINTOSC (32 kHz)
- 0011 = MFINTOSC (500 kHz)
- 0010 = LFINTOSC
- 0001 = HFINTOSC
- 0000 = Fosc

**Note 1:** N1PWS applies only when operating in Pulse Frequency mode.

# PIC16(L)F15354/55

## REGISTER 22-3: NCO1ACCL: NCO1 ACCUMULATOR REGISTER – LOW BYTE

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
NCO1ACC<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                  **NCO1ACC<7:0>**: NCO1 Accumulator, Low Byte

## REGISTER 22-4: NCO1ACCH: NCO1 ACCUMULATOR REGISTER – HIGH BYTE

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
NCO1ACC<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                  **NCO1ACC<15:8>**: NCO1 Accumulator, High Byte

## REGISTER 22-5: NCO1ACCU: NCO1 ACCUMULATOR REGISTER – UPPER BYTE<sup>(1)</sup>

U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	NCO1ACC<19:16>			
bit 7							bit 0

### Legend:

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

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

bit 3-0                  **NCO1ACC<19:16>**: NCO1 Accumulator, Upper Byte

**Note 1:** The accumulator spans registers NCO1ACCU:NCO1ACCH:NCO1ACCL. The 24 bits are reserved but not all are used. This register updates in real-time, asynchronously to the CPU; there is no provision to guarantee atomic access to this 24-bit space using an 8-bit bus. Writing to this register while the module is operating will produce undefined results.



# PIC16(L)F15354/55

## REGISTER 22-6: NCO1INCL: NCO1 INCREMENT REGISTER – LOW BYTE<sup>(1,2)</sup>

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-1/1
NCO1INC<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 **NCO1INC<7:0>**: NCO1 Increment, Low Byte

- Note 1:** The logical increment spans NCO1INC<sub>U</sub>:NCO1INC<sub>H</sub>:NCO1INCL.  
**Note 2:** NCO1INC is double-buffered as INCBUF; INCBUF is updated on the next falling edge of NCOCLK after writing to NCO1INCL; NCO1INC<sub>U</sub> and NCO1INC<sub>H</sub> should be written prior to writing NCO1INCL.

## REGISTER 22-7: NCO1INCH: NCO1 INCREMENT REGISTER – HIGH BYTE<sup>(1)</sup>

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
NCO1INC<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 **NCO1INC<15:8>**: NCO1 Increment, High Byte

- Note 1:** The logical increment spans NCO1INC<sub>U</sub>:NCO1INC<sub>H</sub>:NCO1INCL.

## REGISTER 22-8: NCO1INC<sub>U</sub>: NCO1 INCREMENT REGISTER – UPPER BYTE<sup>(1)</sup>

U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	NCO1INC<19:16>			
bit 7							bit 0

### Legend:

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

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

bit 3-0 **NCO1INC<19:16>**: NCO1 Increment, Upper Byte

- Note 1:** The logical increment spans NCO1INC<sub>U</sub>:NCO1INC<sub>H</sub>:NCO1INCL.

**TABLE 22-1: SUMMARY OF REGISTERS ASSOCIATED WITH NCO**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
INTCON	GIE	PEIE	—	—	—	—	—	INTEDG	121
PIR7	—	—	NVMIF	NCO1IF	—	—	—	CWG1IF	137
PIE7	—	—	NVMIE	NCO1IE	—	—	—	CWG1IE	129
NCO1CON	N1EN	—	N1OUT	N1POL	—	—	—	N1PFM	250
NCO1CLK	N1PWS<2:0>			—	N1CKS<3:0>				251
NCO1ACCL	NCO1ACC<7:0>								252
NCO1ACCH	NCO1ACC<15:8>								252
NCO1ACCU	—	—	—	—	NCO1ACC<19:16>				252
NCO1INCL	NCO1INC<7:0>								253
NCO1INCH	NCO1INC<15:8>								253
NCO1INCUC	—	—	—	—	NCO1AINC<19:16>				253
RxyPPS	—	—	—	RxyPPS<4:0>					200

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

## 23.0 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:

- Programmable input selection
- Selectable voltage reference
- Programmable output polarity
- Rising/falling output edge interrupts
- CWG1 Auto-shutdown source

### 23.1 Comparator Overview

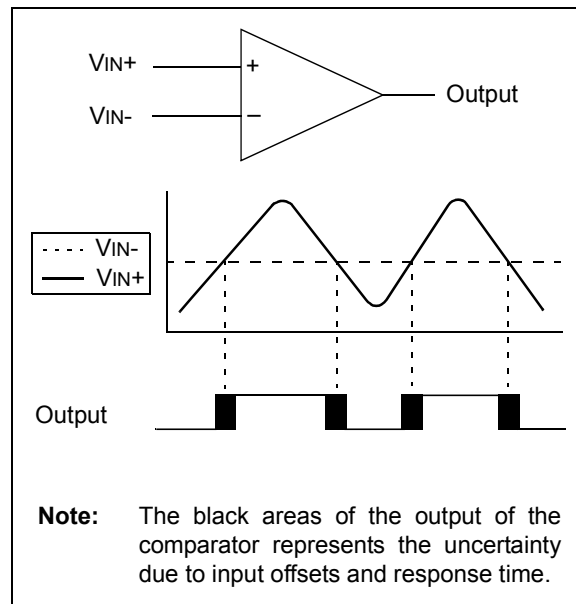
A single comparator is shown in [Figure 23-1](#) along with the relationship between the analog input levels and the digital output. When the analog voltage at  $V_{IN+}$  is less than the analog voltage at  $V_{IN-}$ , the output of the comparator is a digital low level. When the analog voltage at  $V_{IN+}$  is greater than the analog voltage at  $V_{IN-}$ , the output of the comparator is a digital high level.

The comparators available are shown in [Table 23-1](#).

**TABLE 23-1: AVAILABLE COMPARATORS**

Device	C1	C2
PIC16(L)F15354/55	•	•

**FIGURE 23-1: SINGLE COMPARATOR**



**FIGURE 23-2: COMPARATOR MODULE SIMPLIFIED BLOCK DIAGRAM**



## 23.2 Comparator Control

Each comparator has two control registers: CMxCON0 and CMxCON1.

The CMxCON0 register (see [Register 23-1](#)) contains Control and Status bits for the following:

- Enable
- Output
- Output polarity
- Hysteresis enable
- Timer1 output synchronization

The CMxCON1 register (see [Register 23-2](#)) contains Control bits for the following:

- Interrupt on positive/negative edge enables

### 23.2.1 COMPARATOR ENABLE

Setting the CxON bit of the CMxCON0 register enables the comparator for operation. Clearing the CxON bit disables the comparator resulting in minimum current consumption.

### 23.2.2 COMPARATOR OUTPUT

The output of the comparator can be monitored by reading either the CxOUT bit of the CMxCON0 register or the MCxOUT bit of the CMOUT register.

The comparator output can also be routed to an external pin through the RxyPPS register ([Register 15-2](#)). The corresponding TRIS bit must be clear to enable the pin as an output.

**Note 1:** The internal output of the comparator is latched with each instruction cycle. Unless otherwise specified, external outputs are not latched.

### 23.2.3 COMPARATOR OUTPUT POLARITY

Inverting the output of the comparator is functionally equivalent to swapping the comparator inputs. The polarity of the comparator output can be inverted by setting the CxPOL bit of the CMxCON0 register. Clearing the CxPOL bit results in a non-inverted output.

[Table 23-2](#) shows the output state versus input conditions, including polarity control.

**TABLE 23-2: 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

## 23.3 Comparator Hysteresis

A selectable amount of separation voltage can be added to the input pins of each comparator to provide a hysteresis function to the overall operation. Hysteresis is enabled by setting the CxHYS bit of the CMxCON0 register.

See Comparator Specifications in [Table 37-14](#) for more information.

## 23.4 Timer1 Gate Operation

The output resulting from a comparator operation can be used as a source for gate control of Timer1. See [Section 26.7 “Timer 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.

### 23.4.1 COMPARATOR OUTPUT SYNCHRONIZATION

The output from a comparator can be synchronized with Timer1 by setting the CxSYNC bit of the CMxCON0 register.

Once enabled, the comparator output is latched on the falling edge of the Timer1 source clock. If a prescaler is used with Timer1, the comparator output is latched after the prescaling function. To prevent a race condition, the comparator output is latched on the falling edge of the Timer1 clock source and Timer1 increments on the rising edge of its clock source. See the Comparator Block Diagram ([Figure 23-2](#)) and the Timer1 Block Diagram ([Figure 26-1](#)) for more information.

## 23.5 Comparator Interrupt

An interrupt can be generated upon a change in the output value of the comparator for each comparator, a rising edge detector and a falling edge detector are present.

When either edge detector is triggered and its associated enable bit is set (CxINTP and/or CxINTN bits of the CMxCON1 register), the Corresponding Interrupt Flag bit (CxIF bit of the PIR2 register) will be set.

To enable the interrupt, you must set the following bits:

- CxON, CxPOL and CxSP bits of the CMxCON0 register
- CxIE bit of the PIE2 register
- CxINTP bit of the CMxCON1 register (for a rising edge detection)
- CxINTN bit of the CMxCON1 register (for a falling edge detection)
- PEIE and GIE bits of the INTCON register

The associated interrupt flag bit, CxIF bit of the PIR2 register, must be cleared in software. If another edge is detected while this flag is being cleared, the flag will still be set at the end of the sequence.

**Note:** Although a comparator is disabled, an interrupt can be generated by changing the output polarity with the CxPOL bit of the CMxCON0 register, or by switching the comparator on or off with the CxON bit of the CMxCON0 register.

## 23.6 Comparator Positive Input Selection

Configuring the CxPCH<2:0> bits of the CMxPSEL register directs an internal voltage reference or an analog pin to the noninverting input of the comparator:

- CxIN0+ analog pin
- DAC output
- FVR (Fixed Voltage Reference)
- Vss (Ground)

See [Section 18.0 “Fixed Voltage Reference \(FVR\)”](#) for more information on the Fixed Voltage Reference module.

See [Section 21.0 “5-Bit Digital-to-Analog Converter \(DAC1\) Module”](#) for more information on the DAC input signal.

Any time the comparator is disabled (CxON = 0), all comparator inputs are disabled.

## 23.7 Comparator Negative Input Selection

The CxNCH<2:0> bits of the CMxNSEL register direct an analog input pin and internal reference voltage or analog ground to the inverting input of the comparator:

- CxINy - pin
- FVR (Fixed Voltage Reference)
- Analog Ground

**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.

## 23.8 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 37-14](#) for more details.

## 23.9 Analog Input Connection Considerations

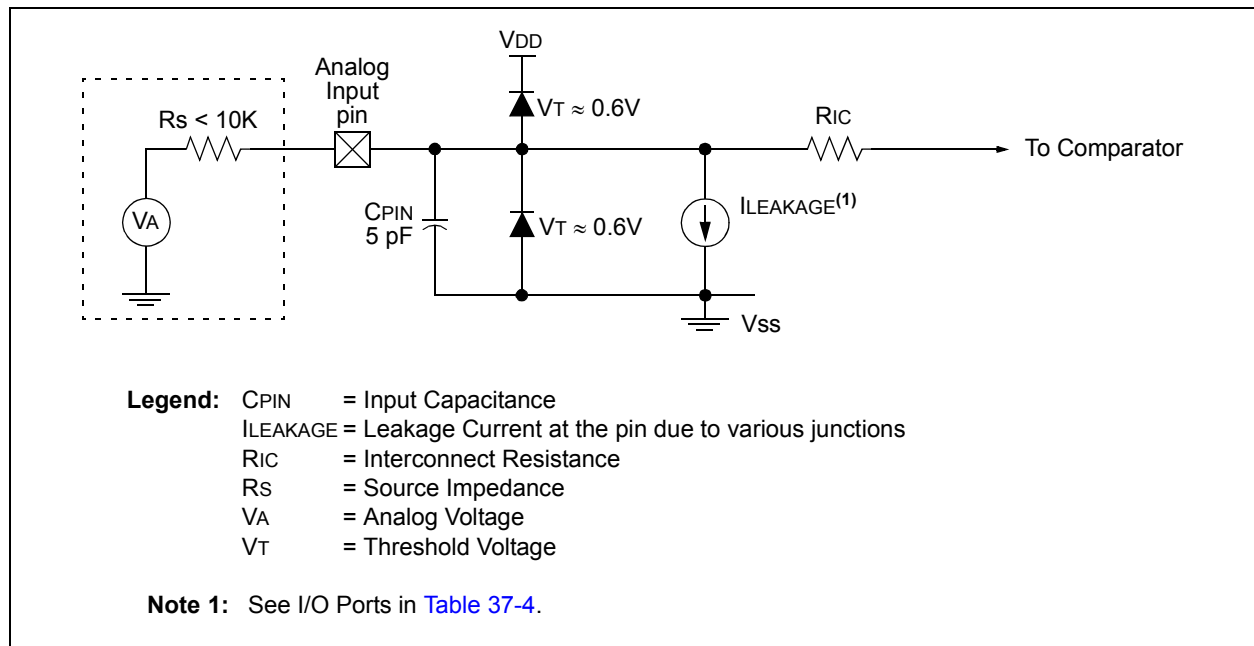
A simplified circuit for an analog input is shown in [Figure 23-3](#). Since the analog input pins share their connection with a digital input, they have reverse biased ESD protection diodes to  $V_{DD}$  and  $V_{SS}$ . The analog input, therefore, must be between  $V_{SS}$  and  $V_{DD}$ . If the input voltage deviates from this range by more than 0.6V in either direction, one of the diodes is forward biased and a latch-up may occur.

A maximum source impedance of 10 k $\Omega$  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 23-3: ANALOG INPUT MODEL**



## 23.10 CWG1 Auto-shutdown Source

The output of the comparator module can be used as an auto-shutdown source for the CWG1 module. When the output of the comparator is active and the corresponding ASxE is enabled, the CWG operation will be suspended immediately (see [Section 30.10 “Auto-Shutdown”](#)).

## 23.11 Operation in Sleep Mode

The comparator module can operate during Sleep. The comparator clock source is based on the Timer1 clock source. If the Timer1 clock source is either the system clock (Fosc) or the instruction clock (Fosc/4), Timer1 will not operate during Sleep, and synchronized comparator outputs will not operate.

A comparator interrupt will wake the device from Sleep. The CxIE bits of the PIE2 register must be set to enable comparator interrupts.



## 23.12 Register Definitions: Comparator Control

**REGISTER 23-1: CMxCON0: COMPARATOR Cx CONTROL REGISTER 0**

R/W-0/0	R-0/0	U-0	R/W-0/0	U-0	U-0	R/W-0/0	R/W-0/0
ON	OUT	—	POL	—	—	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 CxPOL = 1 (inverted polarity):  
 1 = CxVP < CxVN  
 0 = CxVP > CxVN  
If CxPOL = 0 (noninverted polarity):  
 1 = CxVP > CxVN  
 0 = CxVP < CxVN
- bit 5      **Unimplemented:** Read as '0'
- bit 4      **POL:** Comparator Output Polarity Select bit  
 1 = Comparator output is inverted  
 0 = Comparator output is not inverted
- bit 3-2    **Unimplemented:** Read as '0'
- 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 23-2: 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 23-3: CMxNSEL: COMPARATOR Cx NEGATIVE INPUT SELECT REGISTER

U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	—	NCH<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 7-3 **Unimplemented:** Read as '0'

bit 2-0 **NCH<2:0>:** Comparator Negative Input Channel Select bits

- 111 =CxVN connects to AVss
- 110 =CxVN connects to FVR Buffer 2
- 101 =CxVN unconnected
- 100 =CxVN unconnected
- 011 =CxVN connects to CxIN3- pin
- 010 =CxVN connects to CxIN2- pin
- 001 =CxVN connects to CxIN1- pin
- 000 =CxVN connects to CxIN0- pin

## REGISTER 23-4: CMxPSEL: COMPARATOR Cx POSITIVE INPUT SELECT REGISTER

U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	—	PCH<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 7-3 **Unimplemented:** Read as '0'

bit 2-0 **PCH<2:0>:** Comparator Positive Input Channel Select bits

- 111 =CxVP connects to AVss
- 110 =CxVP connects to FVR Buffer 2
- 101 =CxVP connects to DAC output
- 100 =CxVP unconnected
- 011 =CxVP unconnected
- 010 =CxVP unconnected
- 001 =CxVP connects to CxIN1+ pin
- 000 =CxVP connects to CxIN0+ pin

## REGISTER 23-5: CMOUT: COMPARATOR OUTPUT REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	R-0/0	R-0/0
—	—	—	—	—	—	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 7-2	<b>Unimplemented:</b> Read as '0'
bit 1	<b>MC2OUT:</b> Mirror Copy of C2OUT bit
bit 0	<b>MC1OUT:</b> Mirror Copy of C1OUT bit

## TABLE 23-3: SUMMARY OF REGISTERS ASSOCIATED WITH COMPARATOR MODULE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
CMxCON0	ON	OUT	—	POL	—	—	HYS	SYNC	261
CMxCON1	—	—	—	—	—	—	INTP	INTN	262
CMOUT	—	—	—	—	—	—	MC2OUT	MC1OUT	264
FVRCON	FVREN	FVRRDY	TSEN	TSRNG	CDAFVR<1:0>		ADFVR<1:0>		222
DAC1CON0	DAC1EN	—	DAC1OE1	DAC1OE2	DAC1PSS<1:0>		—	DAC1NSS	243
DAC1CON1	—	—	—	DAC1R<4:0>					243
INTCON	GIE	PEIE	—	—	—	—	—	INTEDG	121
PIE2	—	ZCDIE	—	—	—	—	C2IE	C1IE	124
PIR2	—	ZCDIF	—	—	—	—	C2IF	C1IF	132
RxyPPS	—	—	—	RxyPPS<4:0>					200
CLCINxPPS	—	—	CLCIN0PPS<5:0>						199
T1GPPS	—	—	T1GPPS<5:0>						199

**Legend:** — = unimplemented location, read as '0'. Shaded cells are unused by the comparator module.

## 24.0 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,  $V_{CPINV}$ , 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 24-2](#).

The ZCD module is useful when monitoring an A/C 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

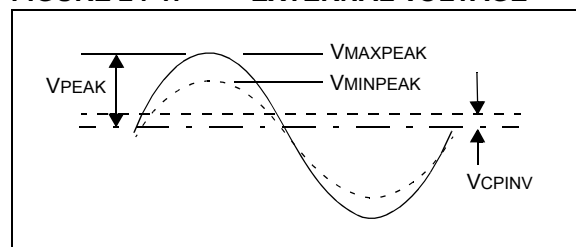
## 24.1 External Resistor Selection

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  $\mu\text{A}$ . Refer to [Equation 24-1](#) and [Figure 24-1](#). 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.

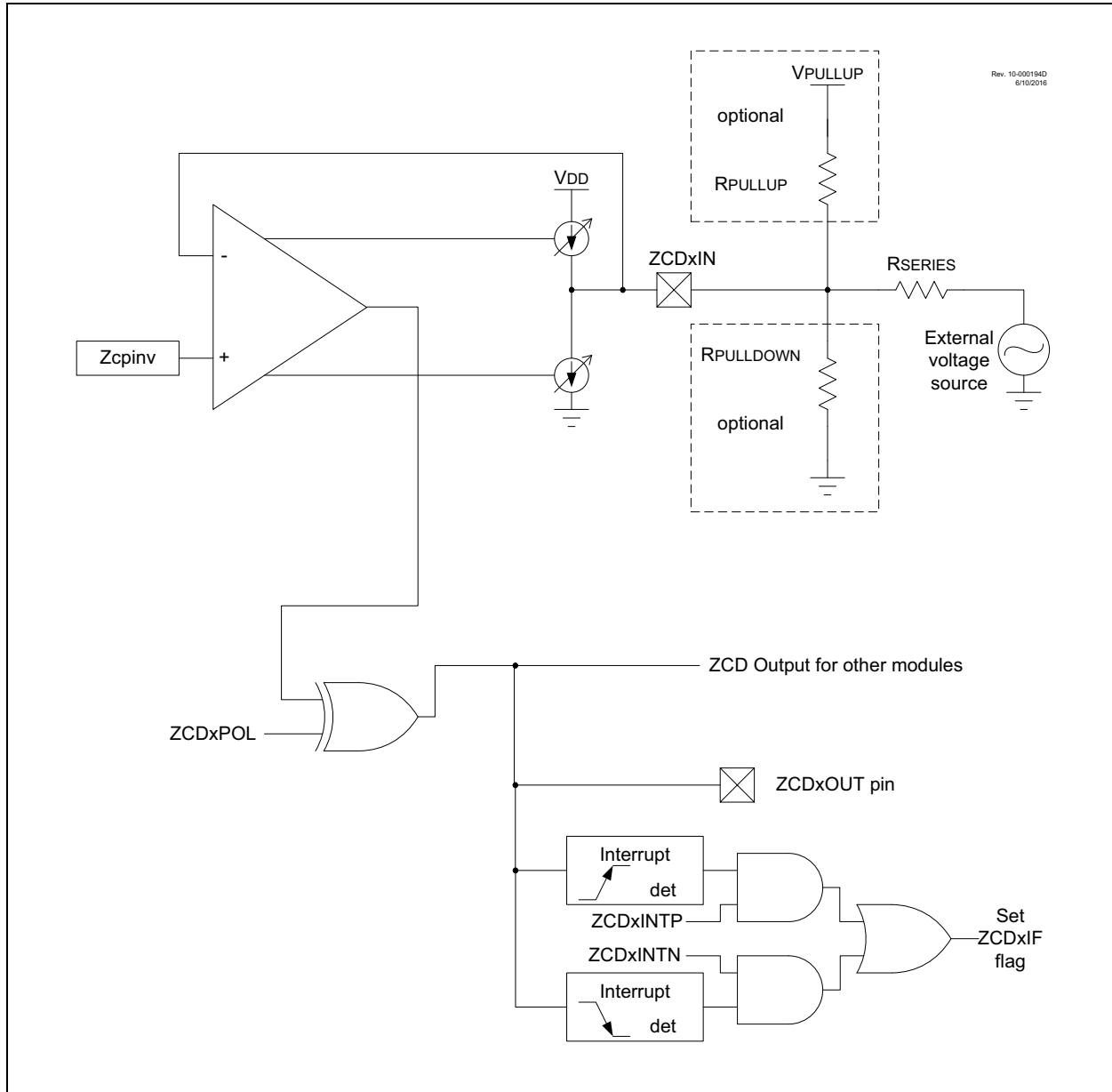
### EQUATION 24-1: EXTERNAL RESISTOR

$$R_{SERIES} = \frac{V_{PEAK}}{3 \times 10^{-4}}$$

FIGURE 24-1: EXTERNAL VOLTAGE



**FIGURE 24-2: SIMPLIFIED ZCD BLOCK DIAGRAM**



## 24.2 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 ZCDxCON 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 even if the module is disabled.

## 24.3 ZCD Logic Polarity

The POL bit of the ZCDxCON register inverts the ZCDxOUT bit relative to the current source and sink output. When the POL bit is set, a logic '1' on the OUT bit indicates that the current source is active, and a logic '0' on the OUT bit indicates that the current sink is active. When the POL bit is clear, a logic '1' on the OUT bit indicates that the current sink is active, and a logic '0' on the OUT bit indicates that the current source is active.

The POL bit affects the ZCD interrupts. See [Section 24.4 “ZCD Interrupts”](#).

## 24.4 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 PIR2 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 PIE2 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 can cause an interrupt, regardless of the level of the EN bit.

The ZCDIF bit of the PIR2 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.

## 24.5 Correcting for VCPINV offset

The actual voltage at which the ZCD switches is the reference voltage at the noninverting 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.

### 24.5.1 CORRECTION BY AC COUPLING

When the external voltage source is sinusoidal then the effects of the VCPINV 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  $\mu$ A. 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 24-2](#).

### EQUATION 24-2: 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 $Z = V_{PEAK}/3 \times 10^{-4}$ $X_c = 1/(2\pi f C)$ $R = \sqrt{Z^2 - X_c^2}$ $V_C = X_c(3 \times 10^{-4})$ $\Phi = \tan^{-1}(X_c/R)$ $T_\Phi = \Phi/(2\pi f)$
--

## EXAMPLE 24-1:

$$\begin{aligned} V_{RMS} &= 120 \\ V_{PEAK} &= V_{RMS} \cdot \sqrt{2} = 169.7 \\ f &= 60 \text{ Hz} \\ C &= 0.1 \text{ }\mu\text{F} \\ Z &= V_{PEAK} / 3 \times 10^{-4} = 169.7 / (3 \times 10^{-4}) = 565.7 \text{ k}\Omega \\ X_C &= 1 / (2\pi f C) = 1 / (2\pi \cdot 60 \cdot 1 \cdot 10^{-7}) = 26.53 \text{ k}\Omega \\ R &= \sqrt{Z^2 - X_C^2} = 565.1 \text{ k}\Omega \text{ (computed)} \\ R &= 560 \text{ k}\Omega \text{ (used)} \\ Z_R &= \sqrt{R^2 + X_C^2} = 560.6 \text{ k}\Omega \text{ (using actual resistor)} \\ I_{PEAK} &= V_{PEAK} / Z_R = 302.7 \cdot 10^{-6} \\ V_C &= X_C \cdot I_{PEAK} = 8.0 \text{ V} \\ \Phi &= \tan^{-1}(X_C / R) = 0.047 \text{ radians} \\ T_\Phi &= \Phi / (2\pi f) = 125.6 \text{ }\mu\text{s} \end{aligned}$$

### 24.5.2 CORRECTION BY OFFSET CURRENT

When the waveform is varying relative to V<sub>SS</sub>, 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 V<sub>DD</sub>, 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 24-3](#).

#### EQUATION 24-3: ZCD EVENT OFFSET

When External Voltage Source is relative to V<sub>SS</sub>:

$$T_{OFFSET} = \frac{\text{asin}\left(\frac{V_{cpinv}}{V_{PEAK}}\right)}{2\pi \cdot Freq}$$

When External Voltage Source is relative to V<sub>DD</sub>:

$$T_{OFFSET} = \frac{\text{asin}\left(\frac{V_{DD} - V_{cpinv}}{V_{PEAK}}\right)}{2\pi \cdot Freq}$$

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 V<sub>SS</sub>. A pull-down resistor is used when the voltage is varying relative to V<sub>DD</sub>. 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 V<sub>CPINV</sub> switching voltage. The pull-up or pull-down value can be determined with the equation shown in [Equation 24-4](#).

#### EQUATION 24-4: ZCD PULL-UP/DOWN

When External Signal is relative to V<sub>SS</sub>:

$$R_{PULLUP} = \frac{R_{SERIES}(V_{PULLUP} - V_{cpinv})}{V_{cpinv}}$$

When External Signal is relative to V<sub>DD</sub>:

$$\left( R_{PULLDOWN} = \frac{R_{SERIES} \times (V_{cpinv})}{(V_{DD} - V_{cpinv})} \right)$$

### 24.6 Handling V<sub>PEAK</sub> variations

If the peak amplitude of the external voltage is expected to vary, the series resistor must be selected to keep the ZCD current source and sink below the design maximum range of  $\pm 600 \mu\text{A}$  and above a reasonable minimum range. 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  $\pm 600 \mu\text{A}$  and the minimum is at least  $\pm 100 \mu\text{A}$ , compute the series resistance as shown in [Equation 24-5](#). The compensating pull-up for this series resistance can be determined with [Equation 24-4](#) because the pull-up value is not dependent from the peak voltage.

#### EQUATION 24-5: SERIES R FOR V RANGE

$$R_{SERIES} = \frac{V_{MAXPEAK} + V_{MINPEAK}}{7 \times 10^{-4}}$$



## 24.7 Operation During Sleep

The ZCD current sources and interrupts are unaffected by Sleep.

## 24.8 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 ZCDDIS Configuration bit is cleared, the ZCD circuit will be active at POR. When the  $\overline{\text{ZCD}}$  Configuration bit is set, the EN bit of the ZCDxCON register must be set to enable the ZCD module.

## 24.9 Register Definitions: ZCD Control

**REGISTER 24-1: ZCDCON: ZERO-CROSS DETECTION CONTROL REGISTER**

R/W-q/q	U-0	R-x/x	R/W-0/0	U-0	U-0	R/W-0/0	R/W-0/0
SEN	—	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	<b>SEN:</b> Zero-Cross Detection Enable bit 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	<b>Unimplemented:</b> Read as '0'
bit 5	<b>OUT:</b> Zero-Cross Detection Logic Level bit <u>POL bit = 1:</u> 1 = ZCD pin is sourcing current 0 = ZCD pin is sinking current <u>POL bit = 0:</u> 1 = ZCD pin is sinking current 0 = ZCD pin is sourcing current
bit 4	<b>POL:</b> Zero-Cross Detection Logic Output Polarity bit 1 = ZCD logic output is inverted 0 = ZCD logic output is not inverted
bit 3-2	<b>Unimplemented:</b> Read as '0'
bit 1	<b>INTP:</b> Zero-Cross Positive Edge Interrupt Enable bit 1 = ZCDIF bit is set on low-to-high ZCDx_output transition 0 = ZCDIF bit is unaffected by low-to-high ZCDx_output transition
bit 0	<b>INTN:</b> Zero-Cross Negative Edge Interrupt Enable bit 1 = ZCDIF bit is set on high-to-low ZCDx_output transition 0 = ZCDIF bit is unaffected by high-to-low ZCDx_output transition

**TABLE 24-1: 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
PIE2	—	ZCDIE	—	—	—	—	C2IE	C1IE	<a href="#">124</a>
PIR2	—	ZCDIF	—	—	—	—	C2IF	C1IF	<a href="#">132</a>
ZCDxCON	EN	—	OUT	POL	—	—	INTP	INTN	<a href="#">270</a>

**Legend:** — = unimplemented, read as '0'. Shaded cells are unused by the ZCD module.

**TABLE 24-2: 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	—	—	DEBUG	STVREN	PPS1WAY	ZCDDIS	BORV	—	<a href="#">77</a>
	7:0	BOREN <1:0>		LPBOREN	—	—	—	PWRTE	MCLRE	

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

## 25.0 TIMER0 MODULE

The Timer0 module is an 8/16-bit timer/counter with the following features:

- 16-bit timer/counter
- 8-bit timer/counter with programmable period
- Synchronous or asynchronous operation
- Selectable clock sources
- Programmable prescaler (independent of Watchdog Timer)
- Programmable postscaler
- Operation during Sleep mode
- Interrupt on match or overflow
- Output on I/O pin (via PPS) or to other peripherals

### 25.1 Timer0 Operation

Timer0 can operate as either an 8-bit timer/counter or a 16-bit timer/counter. The mode is selected with the T016BIT bit of the T0CON register.

#### 25.1.1 16-BIT MODE

In normal operation, TMR0 increments on the rising edge of the clock source. A 15-bit prescaler on the clock input gives several prescale options (see prescaler control bits, T0CKPS<3:0> in the T0CON1 register).

##### 25.1.1.1 Timer0 Reads and Writes in 16-Bit Mode

TMR0H is not the actual high byte of Timer0 in 16-bit mode. It is actually a buffered version of the real high byte of Timer0, which is neither directly readable nor writable (see [Figure 25-1](#)). TMR0H is updated with the contents of the high byte of Timer0 during a read of TMR0L. This provides the ability to read all 16 bits of Timer0 without having to verify that the read of the high and low byte was valid, due to a rollover between successive reads of the high and low byte.

Similarly, a write to the high byte of Timer0 must also take place through the TMR0H Buffer register. The high byte is updated with the contents of TMR0H when a write occurs to TMR0L. This allows all 16 bits of Timer0 to be updated at once.

#### 25.1.2 8-BIT MODE

In normal operation, TMR0 increments on the rising edge of the clock source. A 15-bit prescaler on the clock input gives several prescale options (see prescaler control bits, T0CKPS<3:0> in the T0CON1 register).

The value of TMR0L is compared to that of the Period buffer, a copy of TMR0H, on each clock cycle. When the two values match, the following events happen:

- TMR0\_out goes high for one prescaled clock period
- TMR0L is reset
- The contents of TMR0H are copied to the period buffer

In 8-bit mode, the TMR0L and TMR0H registers are both directly readable and writable. The TMR0L register is cleared on any device Reset, while the TMR0H register initializes at FFh.

Both the prescaler and postscaler counters are cleared on the following events:

- A write to the TMR0L register
- A write to either the T0CON0 or T0CON1 registers
- Any device Reset – Power-on Reset (POR), MCLR Reset, Watchdog Timer Reset (WDTR) or Brown-out Reset (BOR)

#### 25.1.3 COUNTER MODE

In Counter mode, the prescaler is normally disabled by setting the T0CKPS bits of the T0CON1 register to '0000'. Each rising edge of the clock input (or the output of the prescaler if the prescaler is used) increments the counter by '1'.

#### 25.1.4 TIMER MODE

In Timer mode, the Timer0 module will increment every instruction cycle as long as there is a valid clock signal and the T0CKPS bits of the T0CON1 register ([Register 25-2](#)) are set to '0000'. When a prescaler is added, the timer will increment at the rate based on the prescaler value.

#### 25.1.5 ASYNCHRONOUS MODE

When the T0ASYNC bit of the T0CON1 register is set (T0ASYNC = '1'), the counter increments with each rising edge of the input source (or output of the prescaler, if used). Asynchronous mode allows the counter to continue operation during Sleep mode provided that the clock also continues to operate during Sleep.

#### 25.1.6 SYNCHRONOUS MODE

When the T0ASYNC bit of the T0CON1 register is clear (T0ASYNC = 0), the counter clock is synchronized to the system oscillator (Fosc/4). When operating in Synchronous mode, the counter clock frequency cannot exceed Fosc/4.

## 25.2 Clock Source Selection

The T0CS<2:0> bits of the T0CON1 register are used to select the clock source for Timer0. [Register 25-2](#) displays the clock source selections.

### 25.2.1 INTERNAL CLOCK SOURCE

When the internal clock source is selected, Timer0 operates as a timer and will increment on multiples of the clock source, as determined by the Timer0 prescaler.

### 25.2.2 EXTERNAL CLOCK SOURCE

When an external clock source is selected, Timer0 can operate as either a timer or a counter. Timer0 will increment on multiples of the rising edge of the external clock source, as determined by the Timer0 prescaler.

## 25.3 Programmable Prescaler

A software programmable prescaler is available for exclusive use with Timer0. There are 16 prescaler options for Timer0 ranging in powers of two from 1:1 to 1:32768. The prescaler values are selected using the T0CKPS<3:0> bits of the T0CON1 register.

The prescaler is not directly readable or writable. Clearing the prescaler register can be done by writing to the TMR0L register or the T0CON1 register.

## 25.4 Programmable Postscaler

A software programmable postscaler (output divider) is available for exclusive use with Timer0. There are 16 postscaler options for Timer0 ranging from 1:1 to 1:16. The postscaler values are selected using the T0OUTPS<3:0> bits of the T0CON0 register.

The postscaler is not directly readable or writable. Clearing the postscaler register can be done by writing to the TMR0L register or the T0CON0 register.

In the 16-bit mode, if the postscaler option is selected to a ratio other than 1:1, the reload of the TMR0H and TMR0L registers is not possible inside the Interrupt Service Routine. The timer period must be calculated with the prescaler and postscaler factors selected.

## 25.5 Operation during Sleep

When operating synchronously, Timer0 will halt. When operating asynchronously, Timer0 will continue to increment and wake the device from Sleep (if Timer0 interrupts are enabled) provided that the input clock source is active.

## 25.6 Timer0 Interrupts

The Timer0 interrupt flag bit (TMR0IF) is set when either of the following conditions occur:

- 8-bit TMR0L matches the TMR0H value
- 16-bit TMR0 rolls over from 'FFFFh'

When the postscaler bits (T0OUTPS<3:0>) are set to 1:1 operation (no division), the T0IF flag bit will be set with every TMR0 match or rollover. In general, the TMR0IF flag bit will be set every T0OUTPS + 1 matches or rollovers.

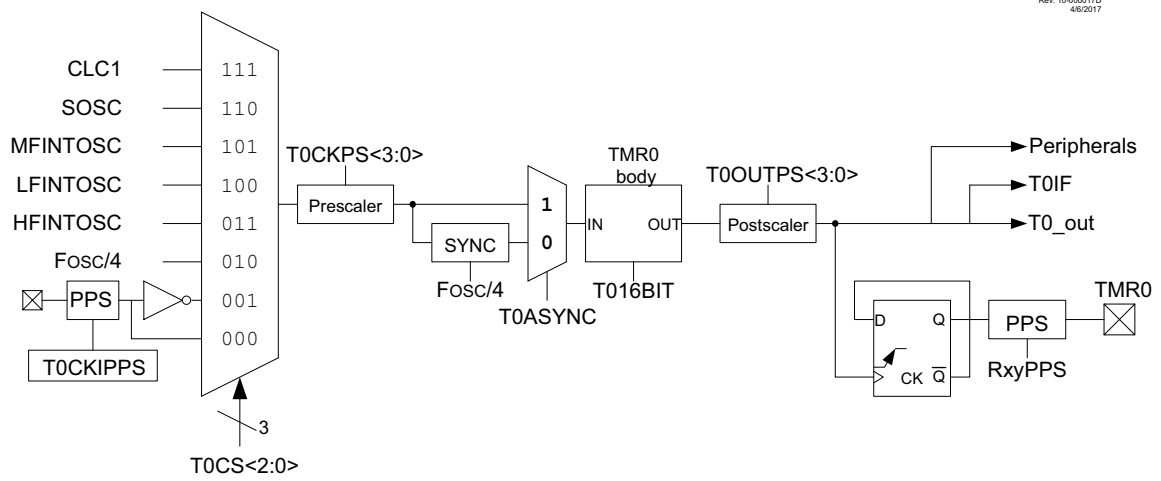
If Timer0 interrupts are enabled (TMR0IE bit of the PIE0 register = 1), the CPU will be interrupted and the device may wake from sleep (see [Section 25.2 “Clock Source Selection”](#) for more details).

## 25.7 Timer0 Output

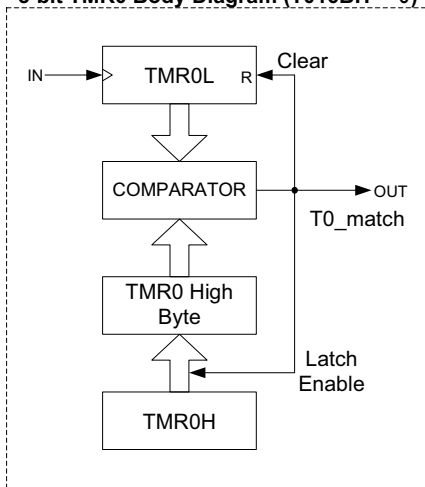
The Timer0 output can be routed to any I/O pin via the RxyPPS output selection register (see [Section 15.0 “Peripheral Pin Select \(PPS\) Module”](#) for additional information). The Timer0 output can also be used by other peripherals, such as the Auto-conversion Trigger of the Analog-to-Digital Converter. Finally, the Timer0 output can be monitored through software via the Timer0 output bit (T0OUT) of the T0CON0 register ([Register 25-1](#)).

TMR0\_out will be one postscaled clock period when a match occurs between TMR0L and TMR0H in 8-bit mode, or when TMR0 rolls over in 16-bit mode. The Timer0 output is a 50% duty cycle that toggles on each TMR0\_out rising clock edge.

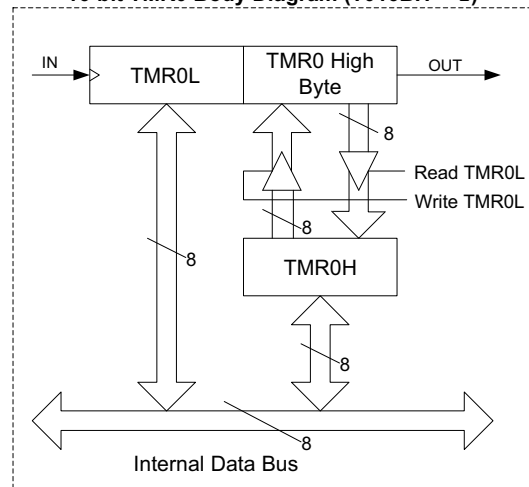
**FIGURE 25-1: BLOCK DIAGRAM OF TIMER0**



**8-bit TMR0 Body Diagram (T016BIT = 0)**



**16-bit TMR0 Body Diagram (T016BIT = 1)**



## 25.8 Register Definitions: Timer0 Control

### REGISTER 25-1: T0CON0: TIMER0 CONTROL REGISTER 0

R/W-0/0	U-0	R-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	
T0EN	—	T0OUT	T016BIT	T0OUTPS<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	<b>T0EN:</b> Timer0 Enable bit 1 = The module is enabled and operating 0 = The module is disabled and in the lowest power mode
bit 6	<b>Unimplemented:</b> Read as '0'
bit 5	<b>T0OUT:</b> Timer0 Output bit (read-only) Timer0 output bit
bit 4	<b>T016BIT:</b> Timer0 Operating as 16-bit Timer Select bit 1 = Timer0 is a 16-bit timer 0 = Timer0 is an 8-bit timer
bit 3-0	<b>T0OUTPS&lt;3:0&gt;:</b> Timer0 output postscaler (divider) 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

# PIC16(L)F15354/55

## REGISTER 25-2: T0CON1: TIMER0 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
T0CS<2:0>		T0ASYNC	T0CKPS<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 7-5      **T0CS<2:0>**: Timer0 Clock Source select bits

111 = LC1\_out  
 110 = SOSC  
 101 = MFINTOSC (500 kHz)  
 100 = LFINTOSC  
 011 = HFINTOSC  
 010 = Fosc/4  
 001 = T0CKIPPS (Inverted)  
 000 = T0CKIPPS (True)

bit 4      **T0ASYNC**: TMR0 Input Asynchronization Enable bit

1 = The input to the TMR0 counter is not synchronized to system clocks  
 0 = The input to the TMR0 counter is synchronized to Fosc/4

bit 3-0      **T0CKPS<3:0>**: Prescaler Rate Select bit

1111 = 1:32768  
 1110 = 1:16384  
 1101 = 1:8192  
 1100 = 1:4096  
 1011 = 1:2048  
 1010 = 1:1024  
 1001 = 1:512  
 1000 = 1:256  
 0111 = 1:128  
 0110 = 1:64  
 0101 = 1:32  
 0100 = 1:16  
 0011 = 1:8  
 0010 = 1:4  
 0001 = 1:2  
 0000 = 1:1

**TABLE 25-1: 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	
TMR0L	Holding Register for the Least Significant Byte of the 16-bit TMR0 Register								271*	
TMR0H	Holding Register for the Most Significant Byte of the 16-bit TMR0 Register								271*	
T0CON0	T0EN	—	T0OUT	T016BIT	T0OUTPS<3:0>					274
T0CON1	T0CS<2:0>			T0ASYNC	T0CKPS<3:0>					275
T0CKIPPS	—	—	T0CKIPPS<5:0>						199	
TMR0PPS	—	—	TMR0PPS<5:0>						199	
T1GCON	GE	GPOL	GTM	GSPM	GGO/DONE	GVAL	—	—	287	
INTCON	GIE	PEIE	—	—	—	—	—	INTEDG	121	
PIR0	—	—	TMR0IF	IOCIF	—	—	—	INTF	130	
PIE0	—	—	TMR0IE	IOCIE	—	—	—	INTE	122	

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

\* Page with Register information.



## 26.0 TIMER1 MODULE WITH GATE CONTROL

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

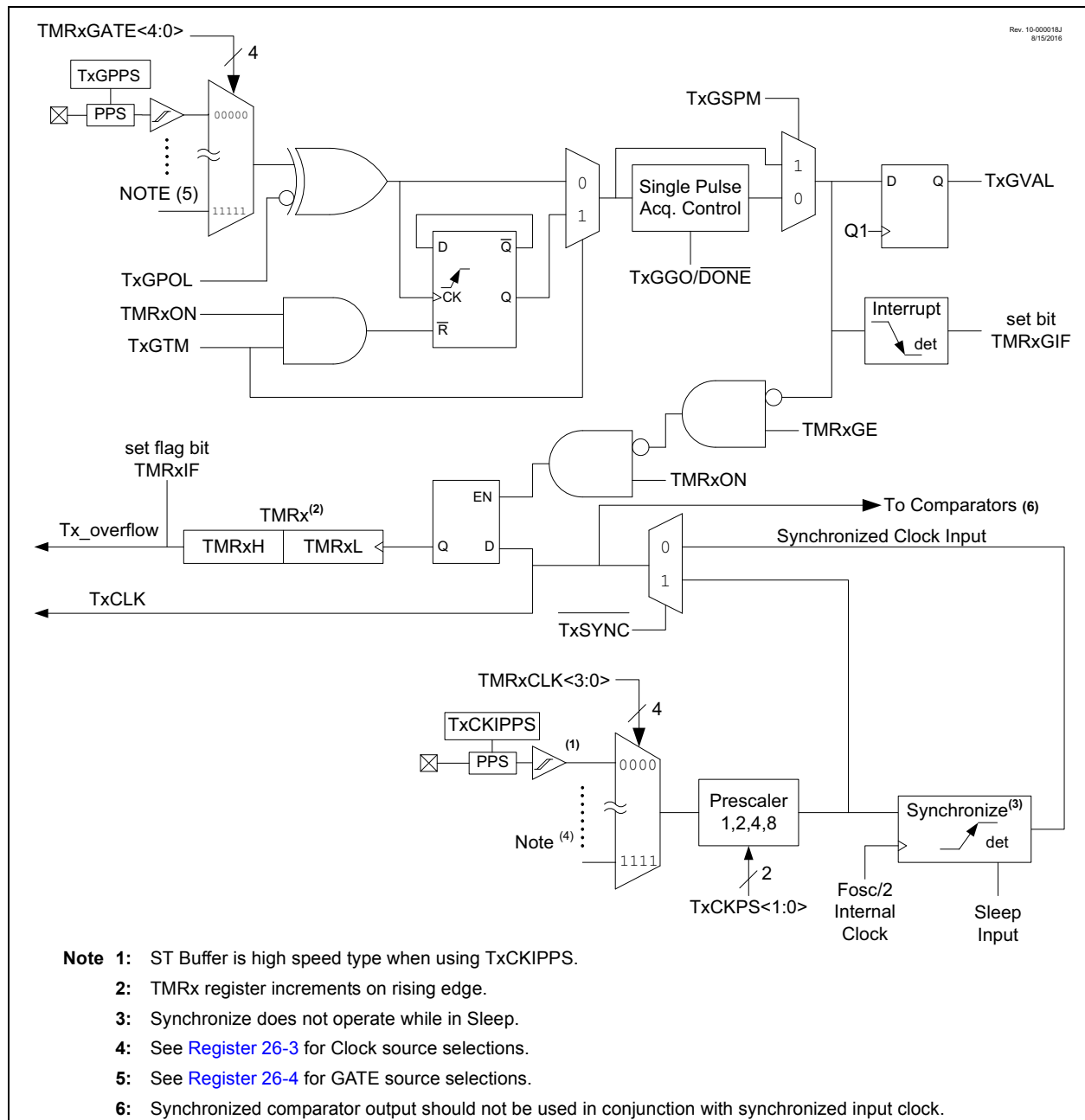
- 16-bit timer/counter register pair (TMR1H:TMR1L)
- Programmable internal or external clock source
- 2-bit prescaler
- Clock source for optional comparator synchronization
- Multiple Timer1 gate (count enable) sources
- Interrupt on overflow

- 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 26-1 is a block diagram of the Timer1 module.

This device has one instance of Timer1 type modules.

**FIGURE 26-1: TIMER1 BLOCK DIAGRAM**



## 26.1 Timer1 Operation

The Timer1 modules are 16-bit incrementing counters which are accessed through the TMR1H:TMR1L register pairs. 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.

The timer is enabled by configuring the TMR1ON and GE bits in the T1CON and T1GCON registers, respectively. [Table 26-1](#) displays the Timer1 enable selections.

**TABLE 26-1: TIMER1 ENABLE SELECTIONS**

TMR1ON	TMR1GE	Timer1 Operation
1	1	Count Enabled
1	0	Always On
0	1	Off
0	0	Off

## 26.2 Clock Source Selection

The T1CLK register is used to select the clock source for the timer. [Register 26-3](#) shows the possible clock sources that may be selected to make the timer increment.

### 26.2.1 INTERNAL CLOCK SOURCE

When the internal clock source FOSC is selected, the TMR1H:TMR1L register pair will increment on multiples of FOSC as determined by the respective Timer1 prescaler.

When the FOSC internal clock source is selected, the timer 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 TMR1H:TMR1L value. To utilize the full resolution of the timer in this mode, an asynchronous input signal must be used to gate the timer clock input.

Out of the total timer gate signal sources, the following subset of sources can be asynchronous and may be useful for this purpose:

- CLC4 output
- CLC3 output
- CLC2 output
- CLC1 output
- Zero-Cross Detect output
- Comparator2 output
- Comparator1 output
- TxG PPS remappable input pin

### 26.2.2 EXTERNAL CLOCK SOURCE

When the timer is enabled and the external clock input source (ex: T1CKI PPS remappable input) is selected as the clock source, the timer will increment on the rising edge of the external clock input.

When using an external clock source, the timer can be configured to run synchronously or asynchronously, as described in [Section 26.6 “Timer Operation in Asynchronous Mode”](#).

When used as a timer with a clock oscillator, an external 32.768 kHz crystal can be used connected to the SOSCI/SOSCO pins.

**Note:** When using Timer1 to count events, 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:

- The timer is first enabled after POR
- Firmware writes to TMR1H or TMR1L
- The timer is disabled
- The timer is re-enabled (e.g., TMR1ON-->1) when the T1CKI signal is currently logic low.

## 26.3 Timer 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.

## 26.4 Timer1 16-Bit Read/Write Mode

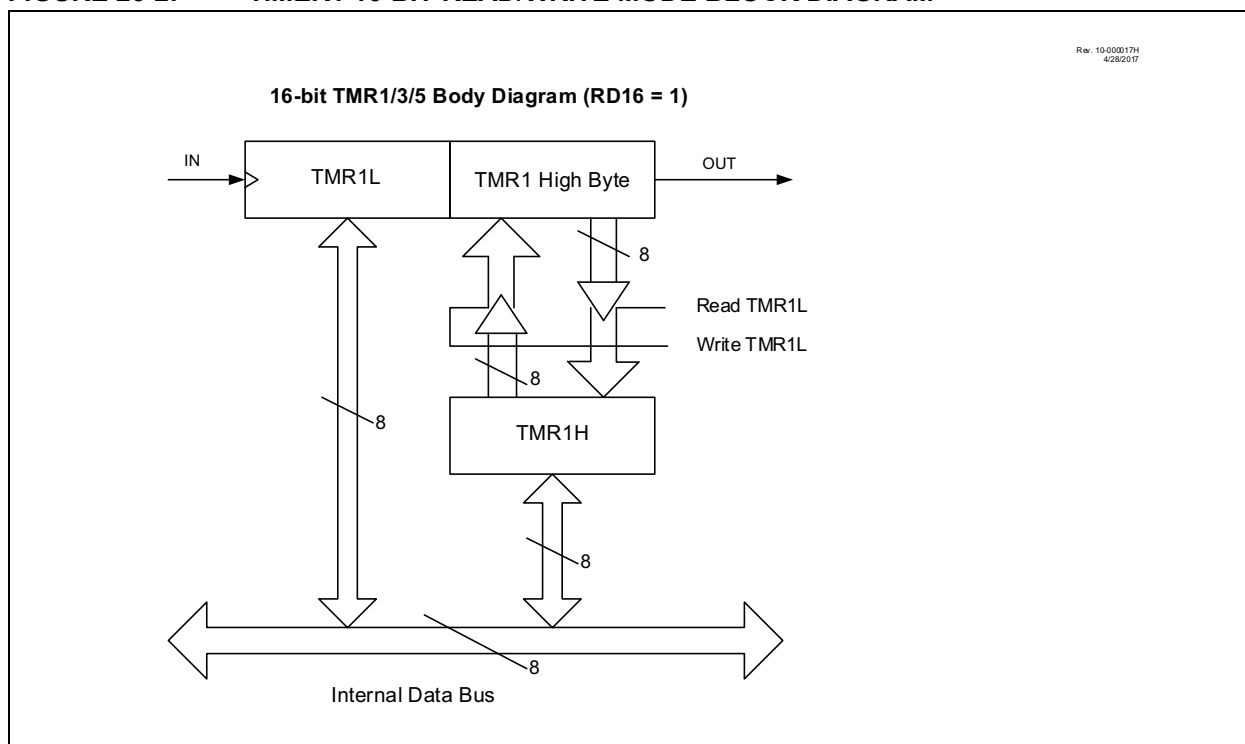
Timer1 can be configured for 16-bit reads and writes. When the RD16 control bit (T1CON<1>) is set, the address for TMR1H is mapped to a buffer register for the high byte of Timer1. A read from TMR1L loads the contents of the high byte of Timer1 into the Timer1 High Byte Buffer register. This provides the user with the

ability to accurately read all 16 bits of Timer1 without having to determine whether a read of the high byte, followed by a read of the low byte, has become invalid due to a rollover between reads.

A write to the high byte of Timer1 must also take place through the TMR1H Buffer register. The Timer1 high byte is updated with the contents of TMR1H when a write occurs to TMR1L. This allows a user to write all 16 bits at once to

both the high and low bytes of Timer1. The high byte of Timer1 is not directly readable or writable in this mode. All reads and writes must take place through the Timer1 High Byte Buffer register. Writes to TMR1H do not clear the Timer1 prescaler. The prescaler is only cleared on writes to TMR1L.

**FIGURE 26-2: TIMER1 16-BIT READ/WRITE MODE BLOCK DIAGRAM**



## 26.5 Secondary Oscillator

A dedicated low-power 32.768 kHz oscillator circuit is built-in between pins SOSC1 (input) and SOSCO (amplifier output). This internal circuit is designed to be used in conjunction with an external 32.768 kHz crystal.

The oscillator circuit is enabled by setting the SOSSEN bit of the OSCEN register. The oscillator will continue to run during Sleep.

**Note:** The oscillator requires a start-up and stabilization time before use. Thus, SOSSEN should be set and a suitable delay observed prior to using Timer1 with the SOSC source. 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.

## 26.6 Timer Operation in Asynchronous 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 26.6.1 “Reading and Writing Timer1 in Asynchronous 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.

### 26.6.1 READING AND WRITING TIMER1 IN ASYNCHRONOUS 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.

## 26.7 Timer Gate

Timer1 can be configured to count freely or the count can be enabled and disabled using the time gate circuitry. This is also referred to as Timer Gate Enable.

The timer gate can also be driven by multiple selectable sources.

### 26.7.1 TIMER GATE ENABLE

The Timer Gate Enable mode is enabled by setting the GE bit of the T1GCON register. The polarity of the Timer Gate Enable mode is configured using the GPOL bit of the T1GCON register.

When Timer Gate Enable signal is enabled, the timer will increment on the rising edge of the Timer1 clock source. When Timer Gate Enable signal is disabled, the timer always increments, regardless of the GE bit. See [Figure 26-4](#) for timing details.

**TABLE 26-2: TIMER GATE ENABLE SELECTIONS**

T1CLK	T1GPOL	T1G	Timer Operation
↑	1	1	Counts
↑	1	0	Holds Count
↑	0	1	Holds Count
↑	0	0	Counts

## 26.7.2 TIMER GATE SOURCE SELECTION

One of the several different external or internal signal sources may be chosen to gate the timer and allow the timer to increment. The gate input signal source can be selected based on the T1GATE register setting. See the T1GATE register ([Register 26-4](#)) description for a complete list of the available gate sources. The polarity for each available source is also selectable. Polarity selection is controlled by the GPOL bit of the T1GCON register.

### 26.7.2.1 T1G Pin Gate Operation

The T1G pin is one source for the timer gate control. It can be used to supply an external source to the time gate circuitry.

### 26.7.2.2 Timer0 Overflow Gate Operation

When Timer0 overflows, or a period register match condition occurs (in 8-bit mode), a low-to-high pulse will automatically be generated and internally supplied to the Timer1 gate circuitry.

### 26.7.2.3 Comparator C1 Gate Operation

The output resulting from a Comparator 1 operation can be selected as a source for the timer gate control. The Comparator 1 output can be synchronized to the timer clock or left asynchronous. For more information see [Section 23.4.1](#) “[Comparator Output Synchronization](#)”.

### 26.7.2.4 Comparator C2 Gate Operation

The output resulting from a Comparator 2 operation can be selected as a source for the timer gate control. The Comparator 2 output can be synchronized to the timer clock or left asynchronous. For more information see [Section 23.4.1](#) “[Comparator Output Synchronization](#)”.

## 26.7.3 TIMER1 GATE TOGGLE MODE

When Timer1 Gate Toggle mode is enabled, it is possible to measure the full-cycle length of a timer gate signal, as opposed to the duration of a single level pulse.

The timer gate source is routed through a flip-flop that changes state on every incrementing edge of the signal. See [Figure 26-5](#) for timing details.

Timer1 Gate Toggle mode is enabled by setting the GTM bit of the T1GCON register. When the GTM 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.

## 26.7.4 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 GSPM bit in the T1GCON register. Next, the GGO/DONE bit in the T1GCON register must be set. The timer will be fully enabled on the next incrementing edge. On the next trailing edge of the pulse, the GGO/DONE bit will automatically be cleared. No other gate events will be allowed to increment the timer until the GGO/DONE bit is once again set in software. See [Figure 26-6](#) for timing details.

If the Single-Pulse Gate mode is disabled by clearing the GSPM bit in the T1GCON register, the GGO/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 timer gate source to be measured. See [Figure 26-7](#) for timing details.

## 26.7.5 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 GVAL bit in the T1GCON register. The GVAL bit is valid even when the timer gate is not enabled (GE bit is cleared).

## 26.7.6 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 GVAL occurs, the TMR1GIF flag bit in the PIR5 register will be set. If the TMR1GIE bit in the PIE5 register is set, then an interrupt will be recognized.

The TMR1GIF flag bit operates even when the timer gate is not enabled (TMR1GE bit is cleared).

## 26.8 Timer1 Interrupts

The timer register pair (TMR1H:TMR1L) increments to FFFFh and rolls over to 0000h. When the timer rolls over, the respective timer interrupt flag bit of the PIR5 register is set. To enable the interrupt on rollover, you must set these bits:

- ON bit of the T1CON register
- TMR1IE bit of the PIE4 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:** To avoid immediate interrupt vectoring, the TMR1H:TMR1L register pair should be preloaded with a value that is not imminently about to rollover, and the TMR1IF flag should be cleared prior to enabling the timer interrupts.

## 26.9 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 PIE4 register must be set
- PEIE bit of the INTCON register must be set
- SYNC bit of the T1CON register must be set
- CS bits of the T1CLK register must be configured
- The timer clock source must be enabled and continue operation during sleep. When the SOSC is used for this purpose, the SOSSEN bit of the OSCEN register must be set.

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 SYNC bit setting.

## 26.10 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 CCPRxH:CCPRxL register pair on a configured event.

In Compare mode, an event is triggered when the value CCPRxH:CCPRxL register pair matches the value in the TMR1H:TMR1L register pair. This event can be an Auto-conversion Trigger.

For more information, see [Section 28.0 “Capture/Compare/PWM Modules”](#).

## 26.11 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 timer interrupt. The CCP module may still be configured to generate a CCP interrupt.

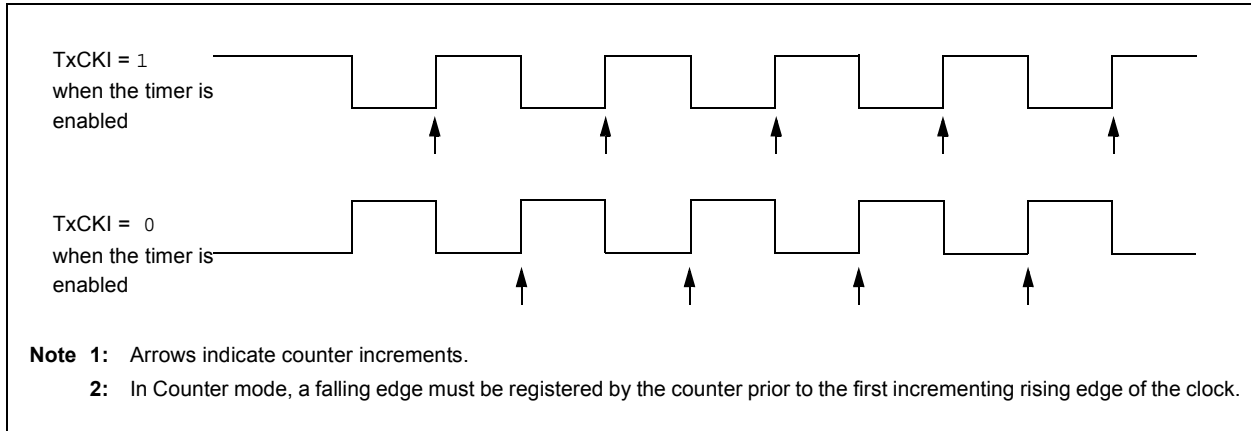
In this mode of operation, the CCPRxH:CCPRxL register pair becomes the period register for Timer1.

The timer should be synchronized and FOSC/4 should be selected as the clock source in order to utilize the Auto-conversion Trigger. Asynchronous operation of the timer can cause an Auto-conversion Trigger to be missed.

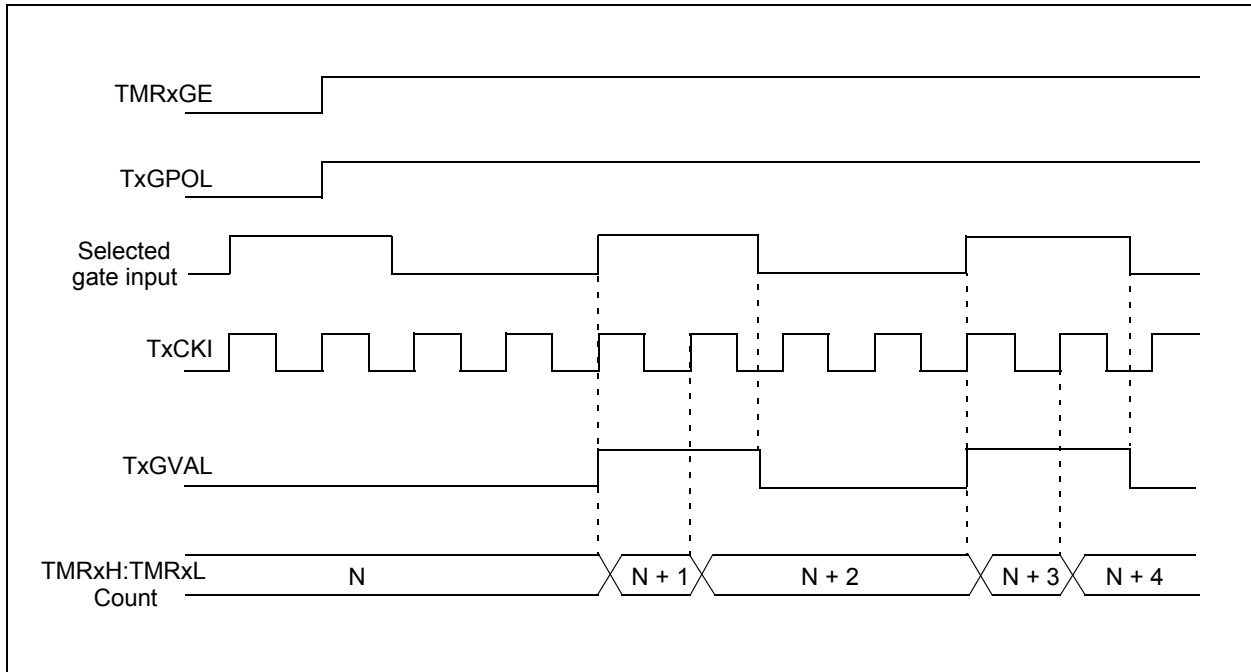
In the event that a write to TMR1H or TMR1L coincides with an Auto-conversion Trigger from the CCP, the write will take precedence.

For more information, see [Section 28.2.4 “Compare During Sleep”](#).

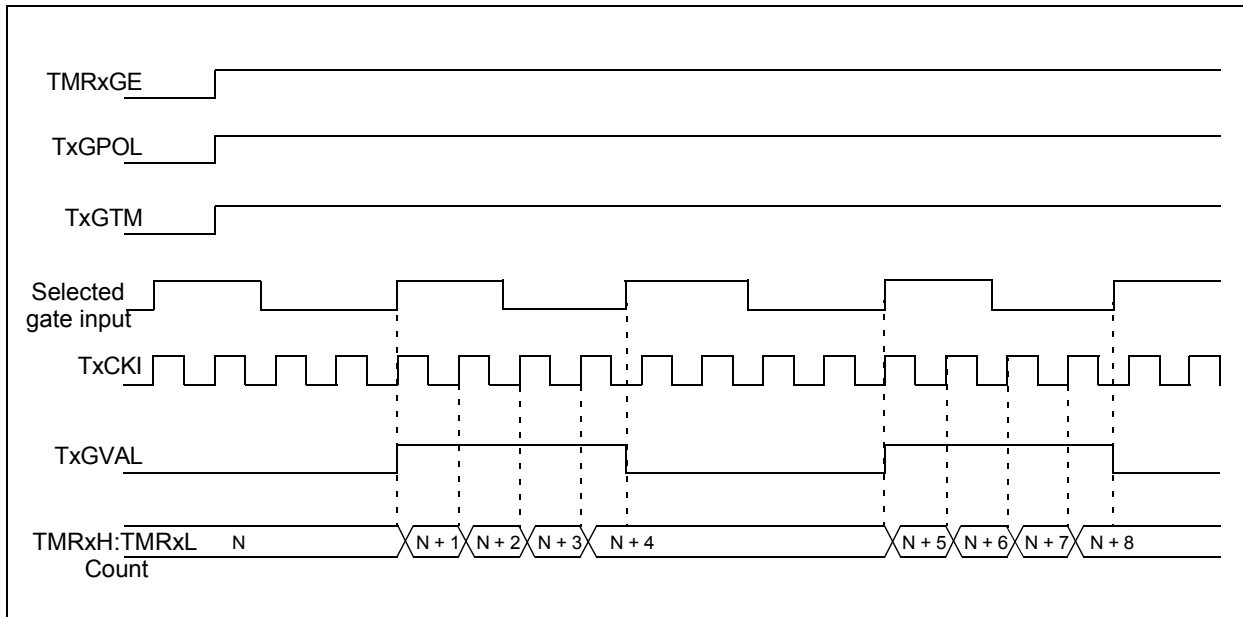
**FIGURE 26-3: TIMER1 INCREMENTING EDGE**



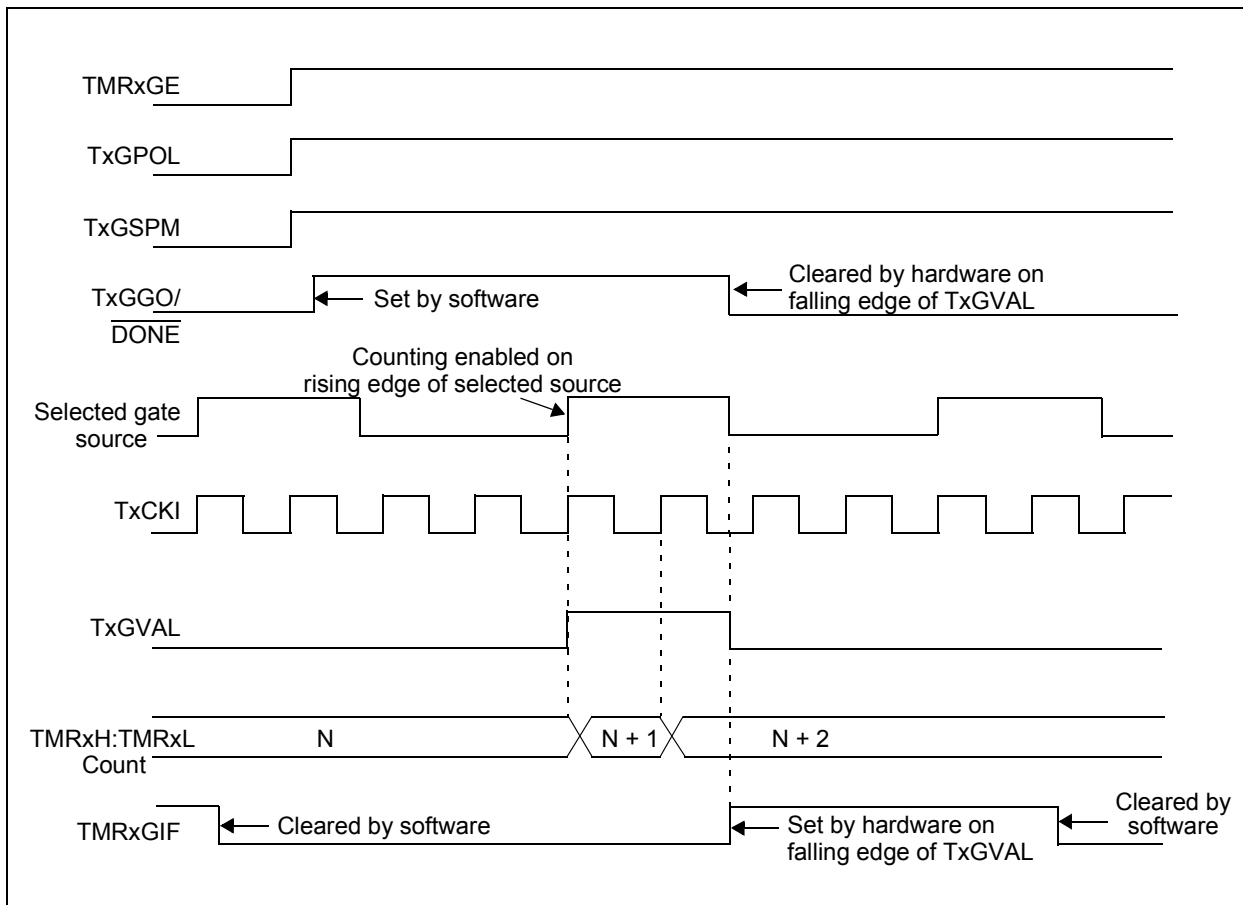
**FIGURE 26-4: TIMER1 GATE ENABLE MODE**



**FIGURE 26-5: TIMER1 GATE TOGGLE MODE**

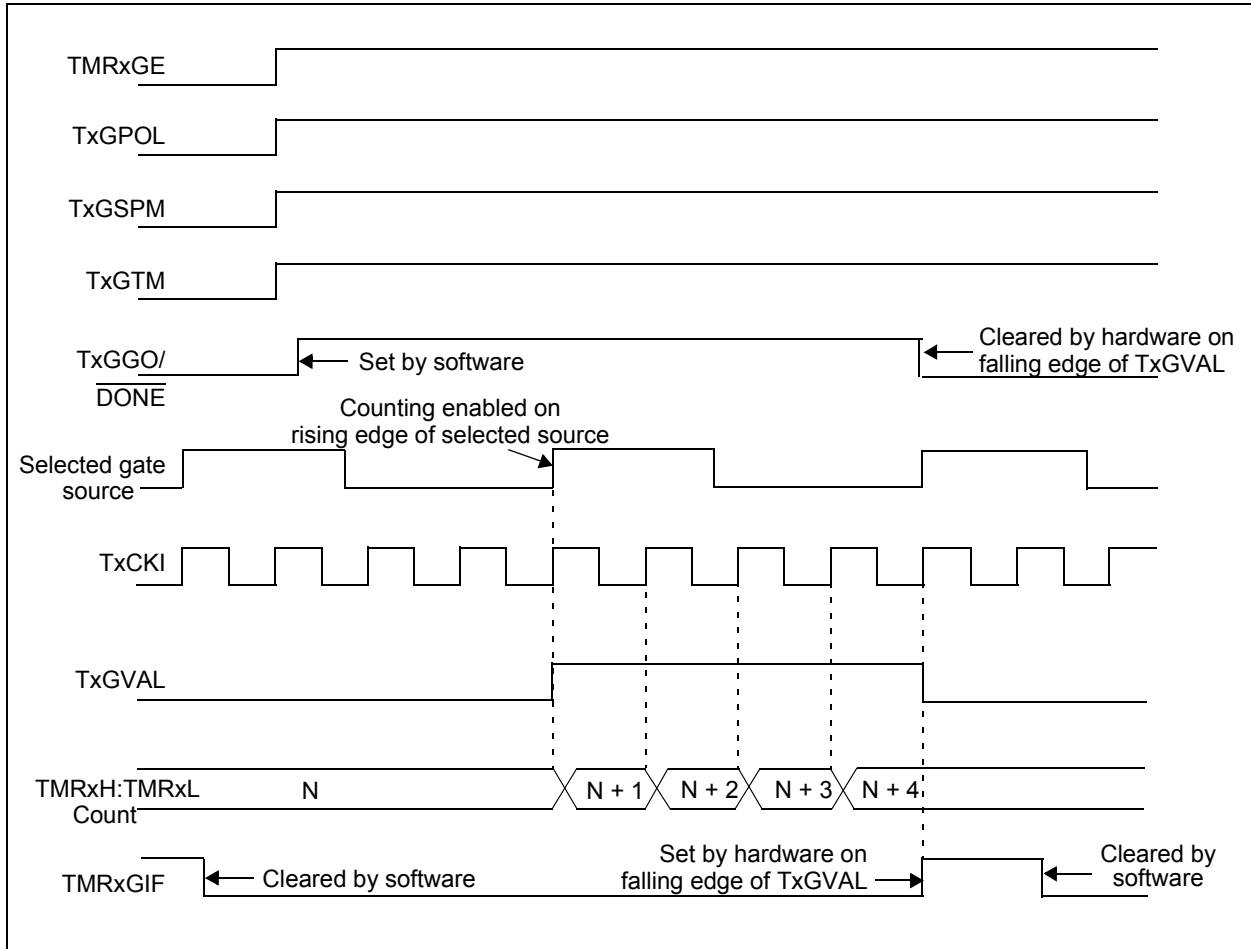


**FIGURE 26-6: TIMER1 GATE SINGLE-PULSE MODE**





**FIGURE 26-7: TIMER1 GATE SINGLE-PULSE AND TOGGLE COMBINED MODE**



## 26.12 Register Definitions: Timer1 Control

**REGISTER 26-1: T1CON: TIMER1 CONTROL REGISTER**

U-0	U-0	R/W-0/u	R/W-0/u	U-0	R/W-0/u	R/W-0/u	R/W-0/u
—	—	CKPS<1:0>		—	SYNC	RD16	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      **Unimplemented:** Read as '0'

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      **Unimplemented:** Read as '0'

bit 2      **SYNC:** Timer1 Synchronization Control bit

When TMR1CLK = Fosc or Fosc/4

This bit is ignored. The timer uses the internal clock and no additional synchronization is performed.

ELSE

0 = Synchronize external clock input with system clock

1 = Do not synchronize external clock input

bit 1      **RD16:** 16-bit Read/Write Mode Enable bit

0 = Enables register read/write of Timer1 in two 8-bit operation

1 = Enables register read/write of Timer1 in one 16-bit operation

bit 0      **ON:** Timer1 On bit

1 = Enables Timer1

0 = Stops Timer1 and clears Timer1 gate flip-flop

## REGISTER 26-2: T1GCON: TIMER1 GATE CONTROL REGISTER

R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R/W/HC-0/u	R-x/x	U-0	U-0
GE	GPOL	GTM	GSPM	GGO/DONE	GVAL	—	—
bit 7						bit 0	

### Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	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 ON = 0:  
This bit is ignored  
If ON = 1:  
1 = Timer1 counting is controlled by the Timer1 gate function  
0 = Timer1 is always counting
- 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  
0 = Timer1 Gate Single-Pulse mode is disabled
- bit 3      **GGO/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  
This bit is automatically cleared when GSPM is cleared
- 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 (GE)
- bit 1-0    **Unimplemented:** Read as '0'

## REGISTER 26-3: T1CLK TIMER1 CLOCK SELECT REGISTER

U-0	U-0	U-0	U-0	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u
—	—	—	—	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

HC = Bit is cleared by hardware

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

bit 3-0      **CS<3:0>:** Timer1 Clock Select bits

1111 = Reserved

1110 = Reserved

1101 = LC4\_out

1100 = LC3\_out

1011 = LC2\_out

1010 = LC1\_out

1001 = Timer0 overflow output

1000 = CLKR output

0111 = SOSC

0110 = MFINTOSC (32 kHz)

0101 = MFINTOSC (500 kHz)

0100 = LFINTOSC

0011 = HFINTOSC

0010 = Fosc

0001 = Fosc/4

0000 = T1CKIPPS

## REGISTER 26-4: T1GATE TIMER1 GATE SELECT REGISTER

U-0	U-0	U-0	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u
—	—	—	GSS<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	HC = Bit is cleared by hardware

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

bit 4-0      **GSS<4:0>:** Timer1 Gate Select bits

11111-10001 = Reserved

10000 = LC4\_out

01111 = LC3\_out

01110 = LC2\_out

01101 = LC1\_out

00100 = ZCD1\_output

01011 = C2OUT\_sync

01010 = C1OUT\_sync

01001 = NCO1\_out

01000 = PWM6\_out

00111 = PWM5\_out

00110 = PWM4\_out

00101 = PWM3\_out

00100 = CCP2\_out

00011 = CCP1\_out

00010 = TMR2\_postscaled

00001 = Timer0 overflow output

00000 = T1GPPS

**TABLE 26-3: SUMMARY OF REGISTERS ASSOCIATED WITH TIMER1**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
INTCON	GIE	PEIE	—	—	—	—	—	INTEDG	121
PIE4	—	—	—	—	—	—	TMR2IE	TMR1IE	126
PIR4	—	—	—	—	—	—	TMR2IF	TMR1IF	134
T1CON	—	—	CKPS<1:0>		—	SYNC	RD16	ON	286
T1GCON	GE	GPOL	GTM	GSPM	GGO/DONE	GVAL	—	—	287
T1GATE	—	—	—	GSS<4:0>					289
T1CLK	—	—	—	—	CS<3:0>				288
TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								277*
TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								277*
T1CKIPPS	—	—	T1CKIPPS<5:0>						199
T1GPPS	—	—	T1GPPS<5:0>						199
CCPxCON	CCPxEN	CCPxOE	CCPxOUT	CCPxFMT	CCPxMODE<3:0>				319
CLCxSELY	—	—	—	LCxDyS<4:0>					364
ADACT	—	—	—	ADACT<4:0>					236

**Legend:** — = Unimplemented location, read as '0'. Shaded cells are not used with the Timer1 modules.

\* Page with register information.

## 27.0 TIMER2 MODULE WITH HARDWARE LIMIT TIMER (HLT)

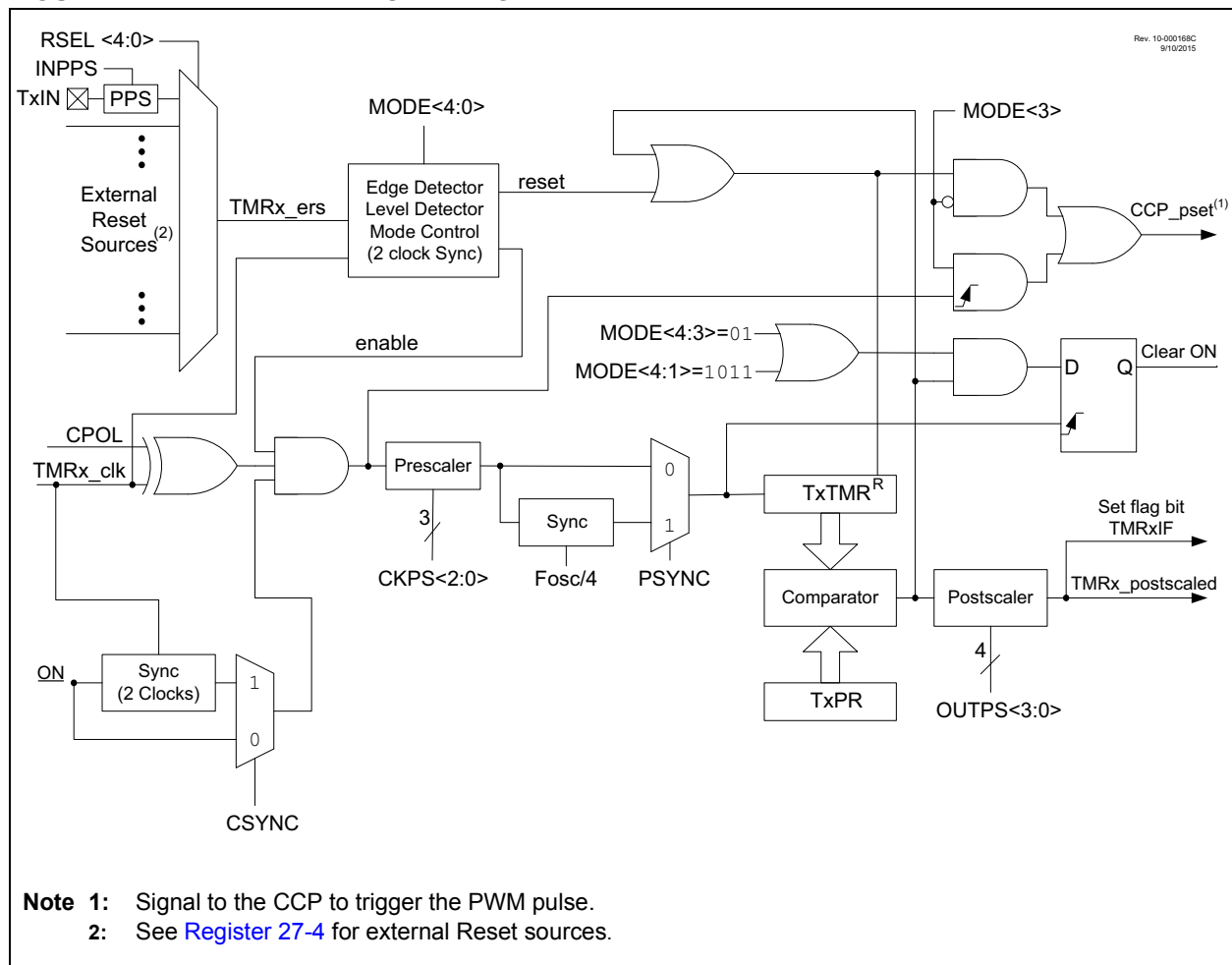
The Timer2 module is an 8-bit timer that can operate as free-running period counters or in conjunction with external signals that control start, run, freeze, and reset operation in One-Shot and Monostable modes of operation. Sophisticated waveform control such as pulse density modulation are possible by combining the operation of this timer with other internal peripherals such as the comparators and CCP modules. Features of the timer include:

- 8-bit timer register
- 8-bit period register

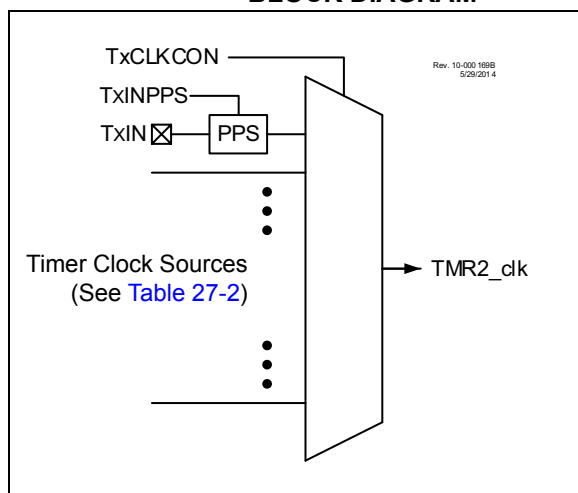
- Selectable external hardware timer Resets
- Programmable prescaler (1:1 to 1:128)
- Programmable postscaler (1:1 to 1:16)
- Selectable synchronous/asynchronous operation
- Alternate clock sources
- Interrupt-on-period
- Three modes of operation:
  - Free Running Period
  - One-shot
  - Monostable

See [Figure 27-1](#) for a block diagram of Timer2. See [Figure 27-2](#) for the clock source block diagram.

**FIGURE 27-1: TIMER2 BLOCK DIAGRAM**



**FIGURE 27-2: TIMER2 CLOCK SOURCE BLOCK DIAGRAM**



## 27.1 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 27-1](#) 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 PR2, 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 PR2 registers are both directly readable and writable. The TMR2 register is cleared and the PR2 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.

### 27.1.1 FREE RUNNING PERIOD MODE

The value of TMR2 is compared to that of the Period register, PR2, on each TMR2\_clk cycle. When the two values match, the comparator resets the value of TMR2 to 00h on the next rising TMR2\_clk edge and increments

the output postscaler counter. When the postscaler count equals the value in the OUTPS<4:0> bits of the TMR2CON1 register, a one TMR2\_clk period wide pulse occurs on the TMR2\_postscaled output, and the postscaler count is cleared.

### 27.1.2 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 PR2 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.

### 27.1.3 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.

## 27.2 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 TMR2CON register. The PR2 postscaler is incremented each time the TMR2 value matches the PR2 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 28.0 "Capture/Compare/PWM Modules"](#) for more details on setting up Timer2 for use with the CCP, as well as the timing diagrams in [Section 27.5 "Operation Examples"](#) for examples of how the varying Timer2 modes affect CCP PWM output.

## 27.3 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 with the T2RST register. 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 TMR2HLT 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 27-1: TIMER2 OPERATING MODES**

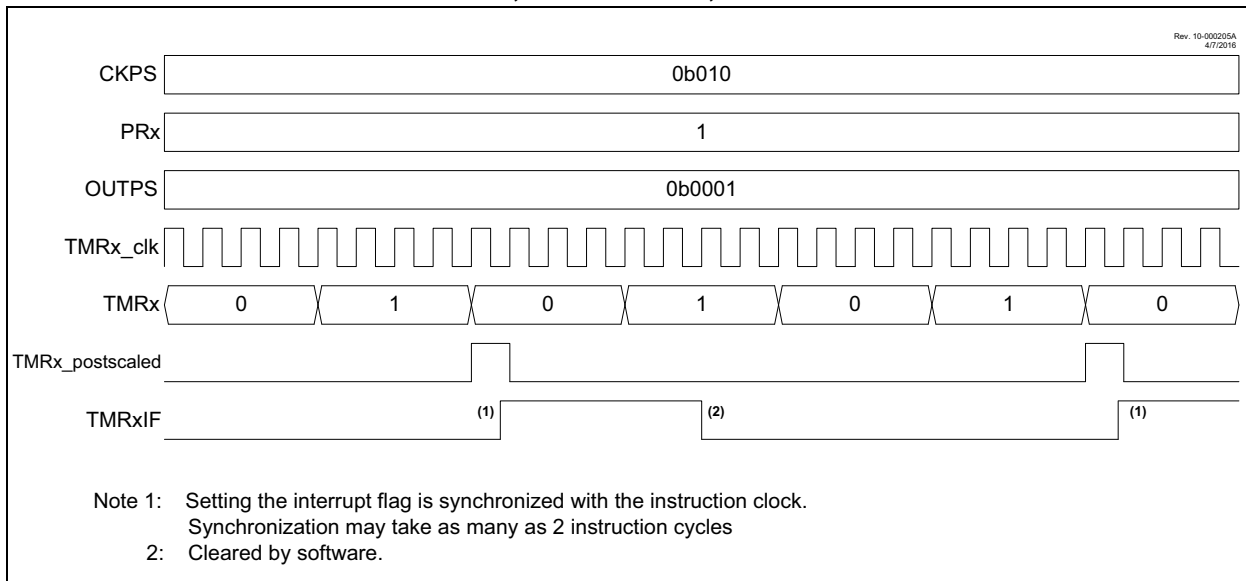
Mode	MODE<4:0>		Output Operation	Operation	Timer Control			
	<4:3>	<2:0>			Start	Reset	Stop	
Free Running Period	00	000	Period Pulse	Software gate (Figure 27-4)	ON = 1	—	ON = 0	
		001		Hardware gate, active-high (Figure 27-5)	ON = 1 and TMRx_ers = 1	—	ON = 0 or TMRx_ers = 0	
		010		Hardware gate, active-low	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 27-6)		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 27-7)		TMRx_ers = 1	ON = 0 or TMRx_ers = 1	
One-shot	01	000	One-shot	Software start (Figure 27-8)	ON = 1	—	ON = 0 or Next clock after TMRx = PRx (Note 2)	
		001	Edge triggered start (Note 1)	Rising edge start (Figure 27-9)	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 27-10)	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 27-11)	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 27-12)	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	11	110	Level triggered start and hardware Reset	High level start and Low level Reset (Figure 27-13)	ON = 1 and TMRx_ers = 1	TMRx_ers = 0
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.  
**Note 2:** When TMRx = PRx then the next clock clears ON and stops TMRx at 00h.  
**Note 3:** When TMRx = PRx then the next clock stops TMRx at 00h but does not clear ON.

## 27.4 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 PIE4 register. Interrupt timing is illustrated in Figure 27-3.

**FIGURE 27-3: TIMER2 PRESCALER, POSTSCALER, AND INTERRUPT TIMING DIAGRAM**



## 27.5 Operation Examples

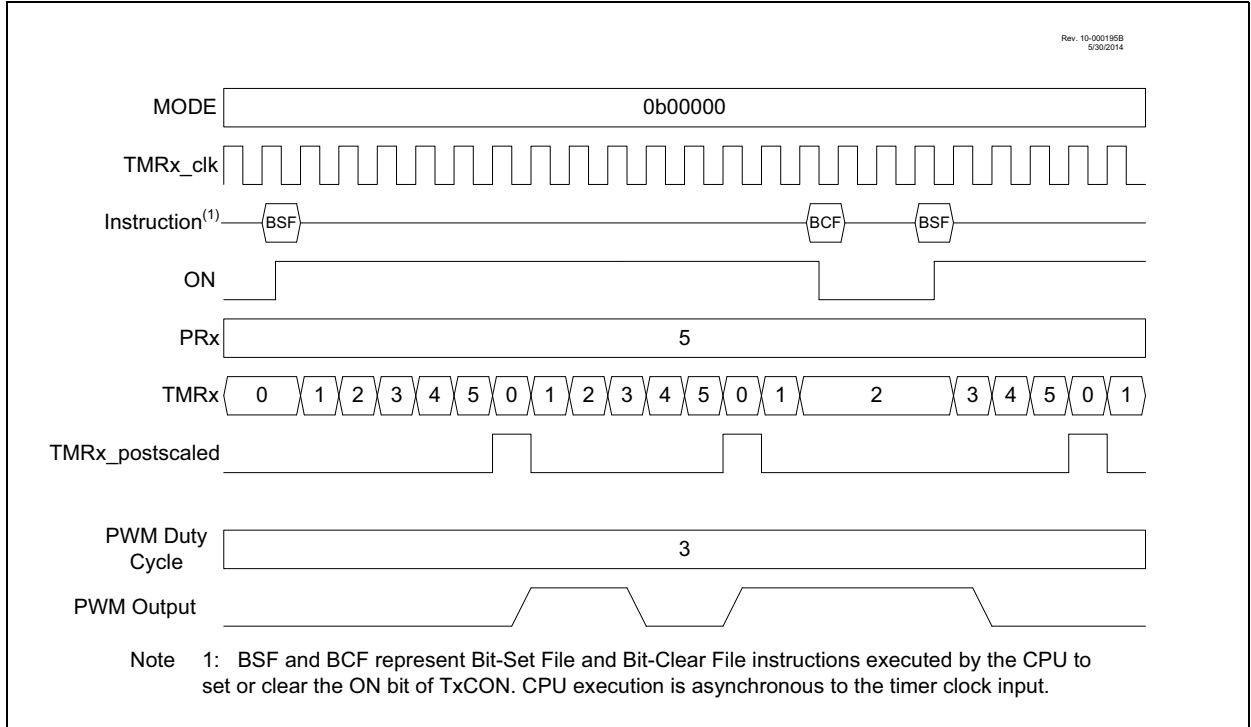
Unless otherwise specified, the following notes apply to the following timing diagrams:

- Both the prescaler and postscaler are set to 1:1 (both the CKPS and OUTPS bits in the T2CON register are cleared).
- The diagrams illustrate any clock except  $F_{osc}/4$  and show clock-sync delays of at least two full cycles for both ON and Timer2\_ers. When using  $F_{osc}/4$ , 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 28.0 “Capture/Compare/PWM Modules”](#). The signals are not a part of the Timer2 module.

### 27.5.1 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 TMR2 count equals the PR2 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 27-4](#). With  $PR2 = 5$ , the counter advances until  $TMR2 = 5$ , and goes to zero with the next clock.

**FIGURE 27-4: SOFTWARE GATE MODE TIMING DIAGRAM (MODE = 00000)**



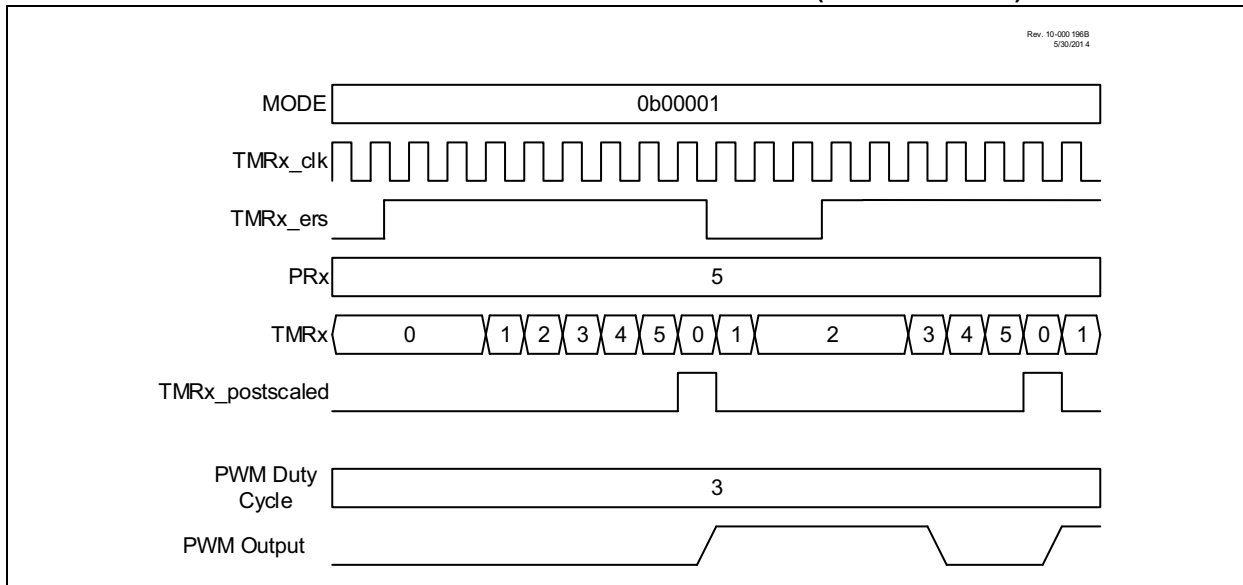
## 27.5.2 HARDWARE GATE MODE

The Hardware Gate modes operate the same as the Software Gate mode except the TMR2\_ers external signal gates 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 27-5 illustrates the Hardware Gating mode for MODE<4:0> = 00001 in which a high input level starts the counter.

**FIGURE 27-5: HARDWARE GATE MODE TIMING DIAGRAM (MODE = 00001)**



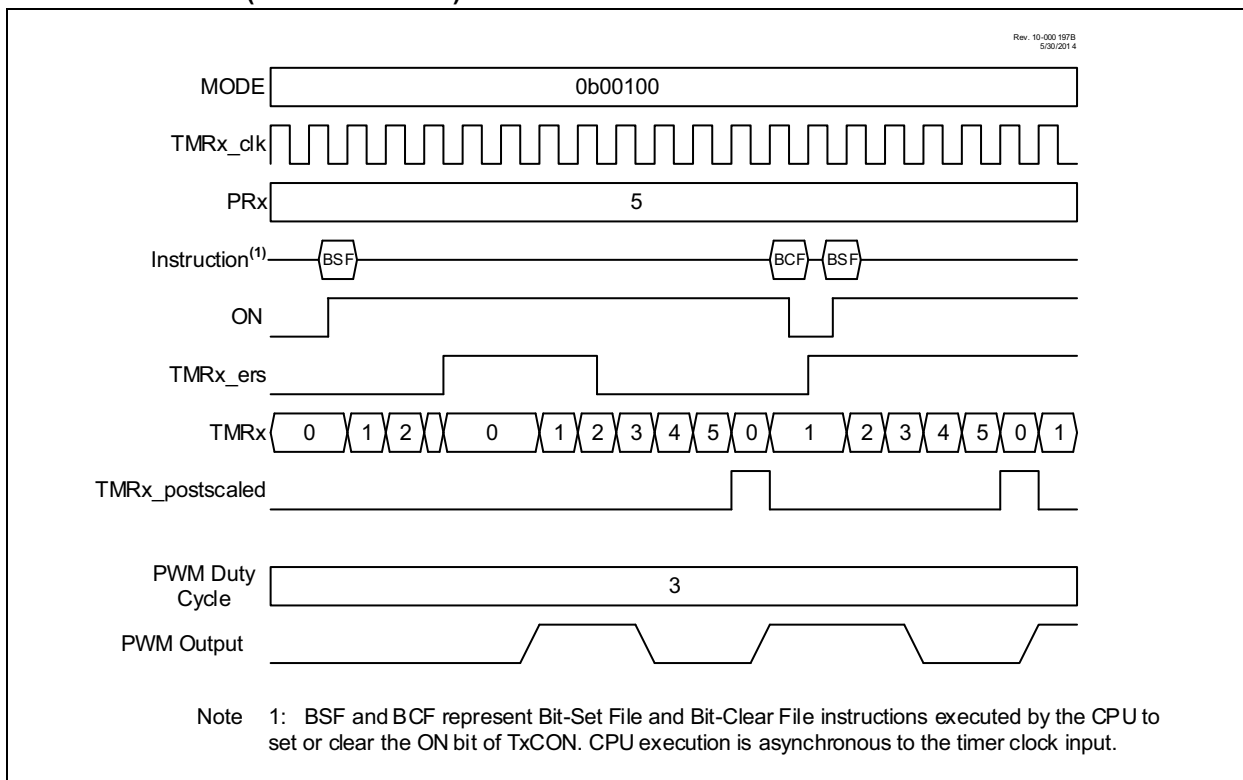
## 27.5.3 EDGE-TRIGGERED HARDWARE LIMIT MODE

In Hardware Limit mode the timer can be reset by the TMR2\_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)  
(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 27-6](#).

**FIGURE 27-6: EDGE-TRIGGERED HARDWARE LIMIT MODE TIMING DIAGRAM (MODE = 00100)**



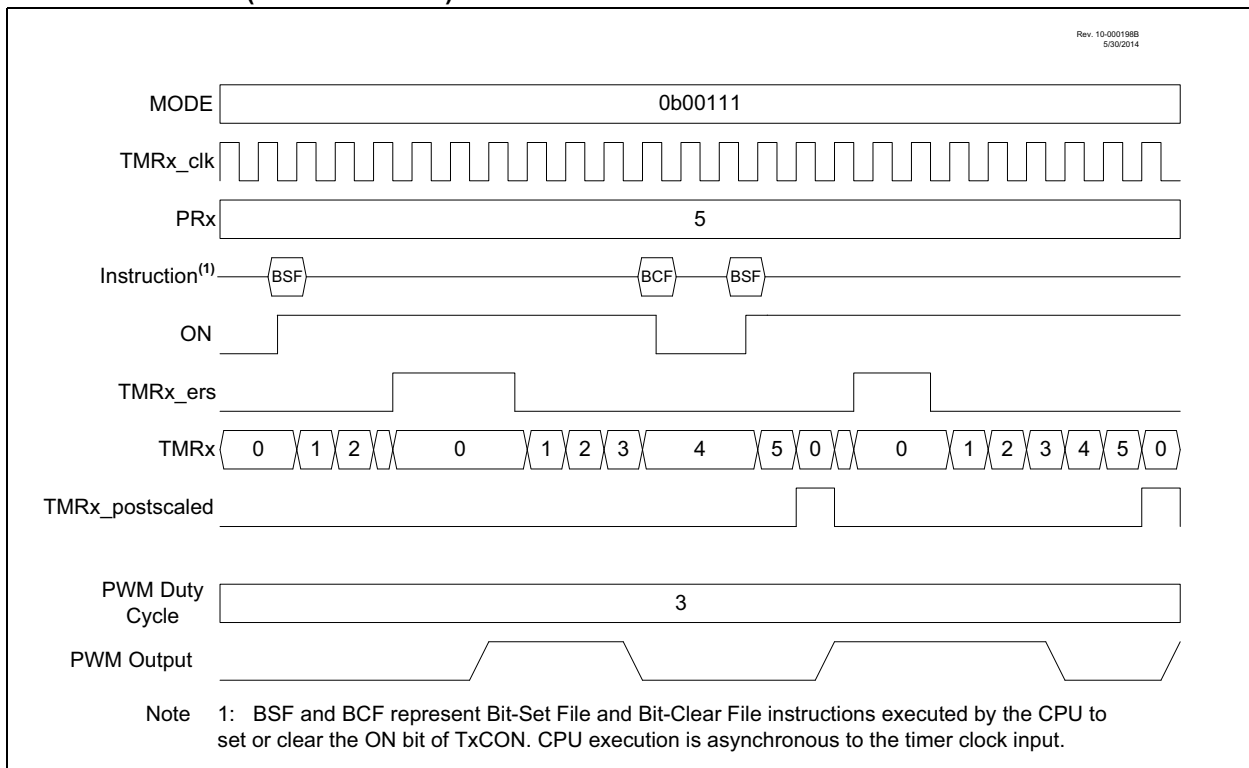
## 27.5.4 LEVEL-TRIGGERED HARDWARE LIMIT MODE

In the Level-Triggered Hardware Limit Timer modes the counter is reset by high or low levels of the external signal TMR2\_ers, as shown in Figure 27-7. 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 TMR2\_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 PR2 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 PR2 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 27-7: LEVEL-TRIGGERED HARDWARE LIMIT MODE TIMING DIAGRAM (MODE = 00111)**

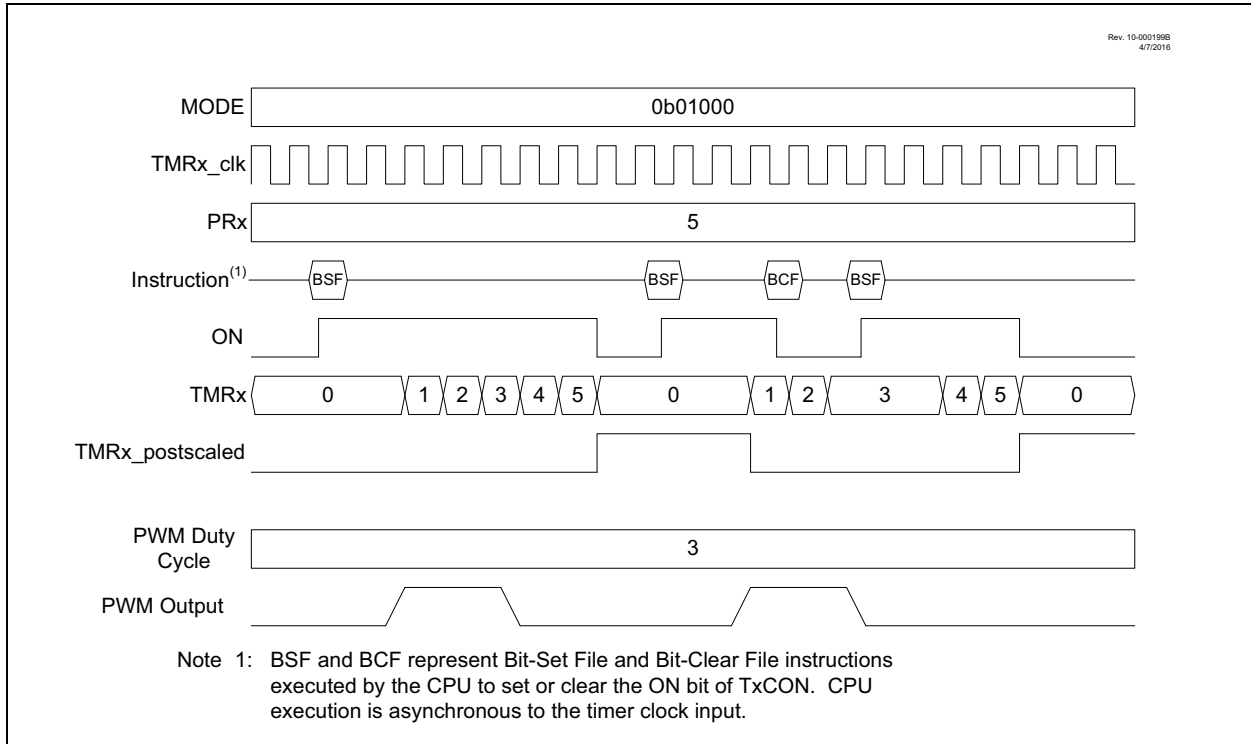


## 27.5.5 SOFTWARE START ONE-SHOT MODE

In One-Shot mode the timer resets and the ON bit is cleared when the timer value matches the PR2 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 27-8. 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 PR2 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 PR2 period count match.

**FIGURE 27-8: SOFTWARE START ONE-SHOT MODE TIMING DIAGRAM (MODE = 01000)**





## 27.5.6 EDGE-TRIGGERED ONE-SHOT MODE

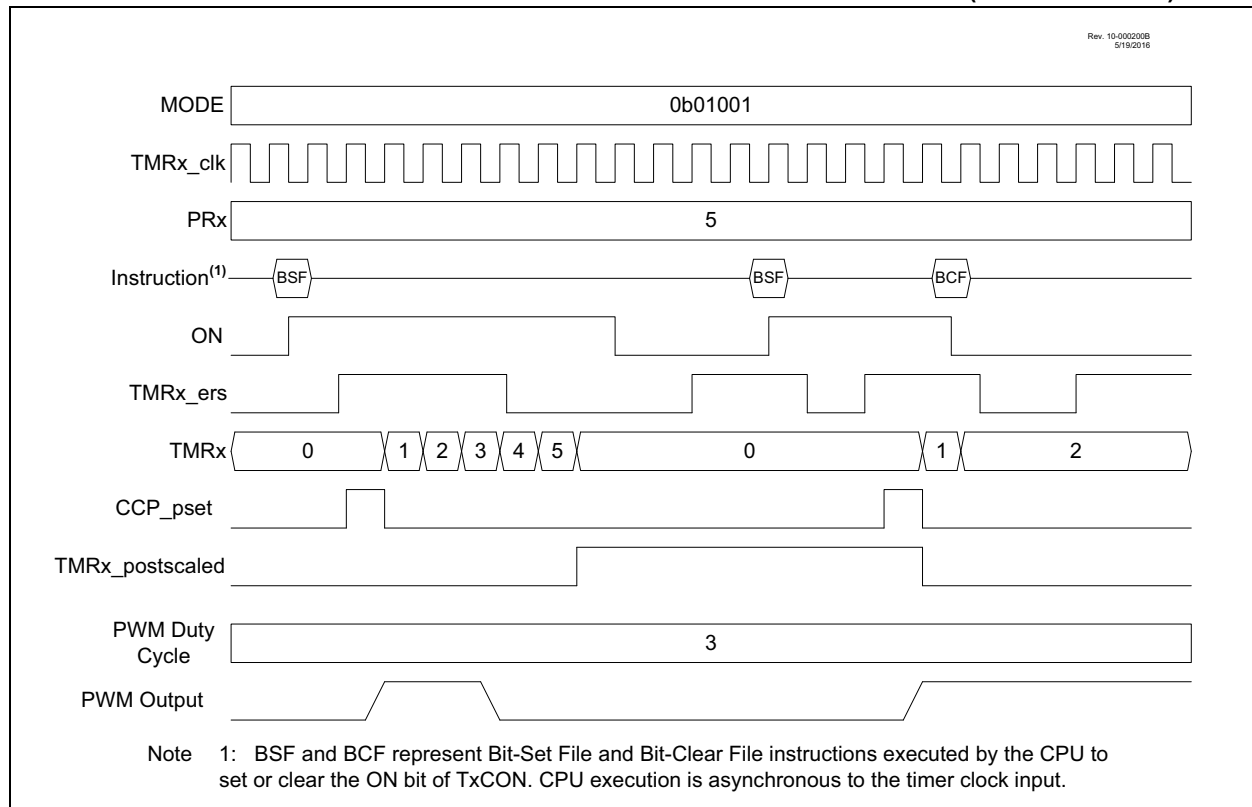
The Edge-Triggered One-Shot modes start the timer on an edge from the external signal input, after the ON bit is set, and clear the ON bit when the timer matches the PR2 period value. The following edges will start the timer:

- 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 TMR2\_ers edge is required after the ON bit is set to resume counting. [Figure 27-9](#) 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 PR2 period count match.

**FIGURE 27-9: EDGE-TRIGGERED ONE-SHOT MODE TIMING DIAGRAM (MODE = 01001)**



## 27.5.7 EDGE-TRIGGERED HARDWARE LIMIT ONE-SHOT MODE

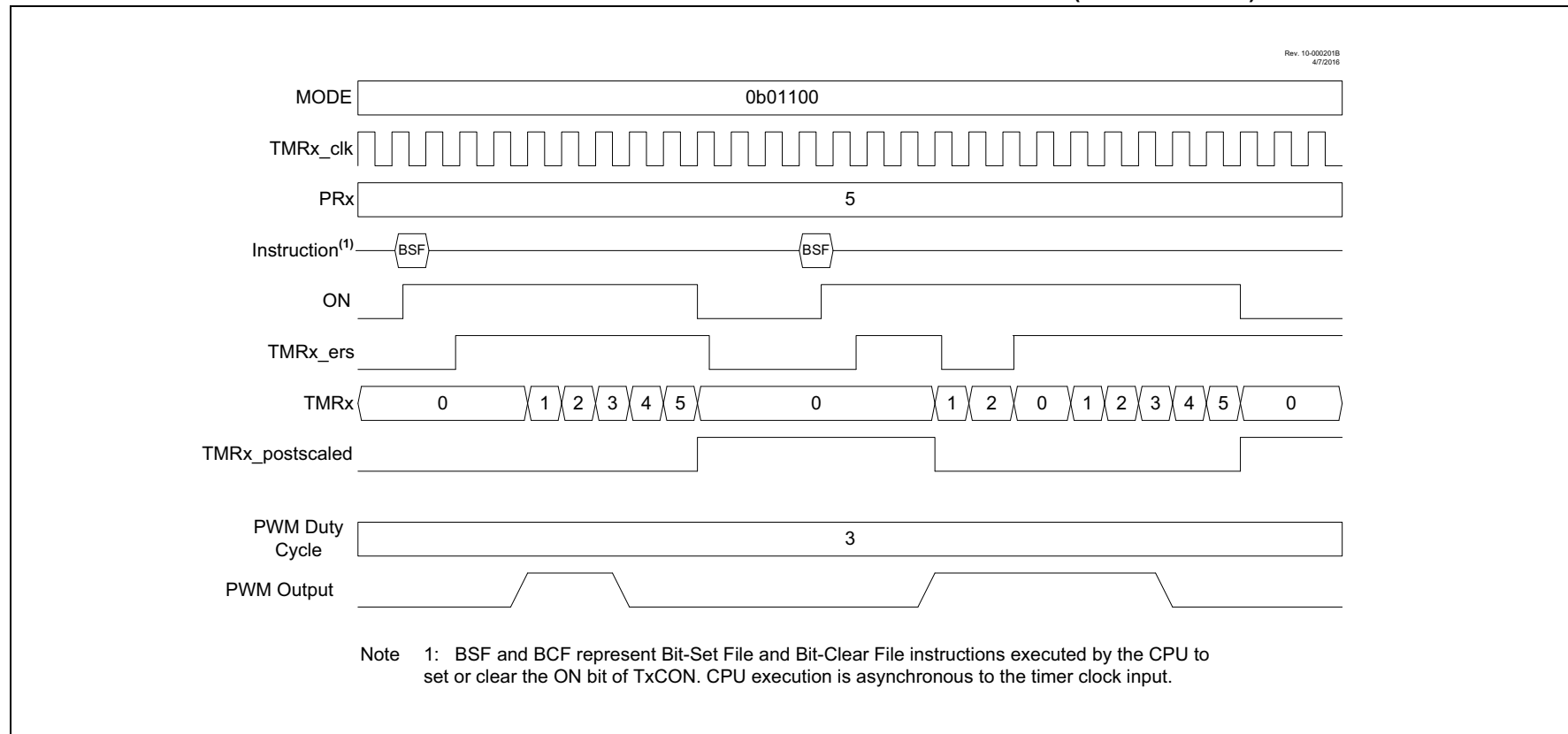
In Edge-Triggered Hardware Limit One-Shot modes the timer starts on the first external signal edge after the ON bit is set and resets on all subsequent edges. Only the first edge after the ON bit is set is needed to start the timer. The counter will resume counting automatically two clocks after all 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 PR2 period value. External signal edges will have no effect until after software sets the ON bit. Figure 27-10 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 PR2 period match unless an external signal edge resets the timer before the match occurs.

**FIGURE 27-10: EDGE-TRIGGERED HARDWARE LIMIT ONE-SHOT MODE TIMING DIAGRAM (MODE = 01100)**



### 27.5.8 LEVEL RESET, EDGE-TRIGGERED HARDWARE LIMIT ONE-SHOT MODES

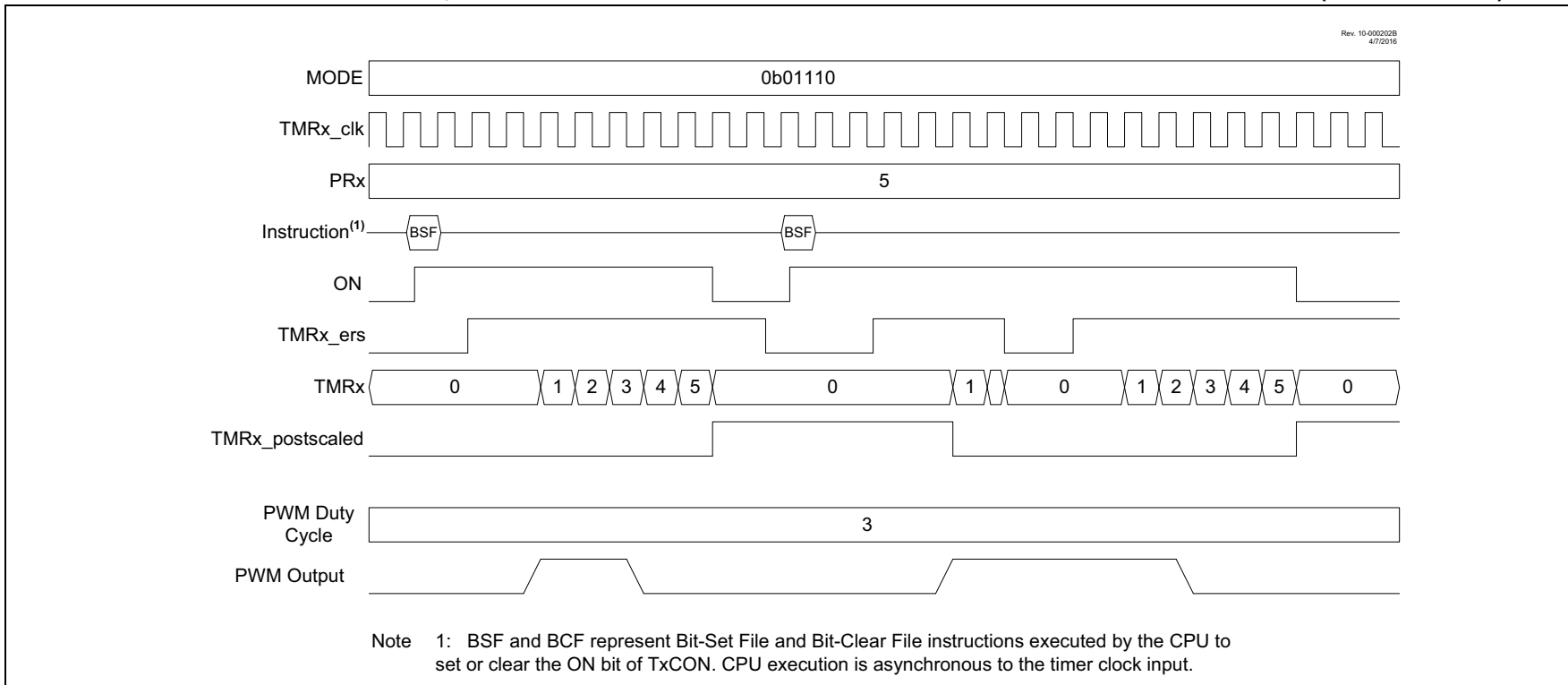
In Level -Triggered One-Shot mode the timer count is reset on the external signal level and starts counting on the rising/falling edge of the transition from Reset level to the active level while the ON bit is set. Reset levels are selected as follows:

- Low Reset level (MODE<4:0> = 01110)
- High Reset level (MODE<4:0> = 01111)

When the timer count matches the PR2 period count, the timer is reset and the ON bit is cleared. When the ON bit is cleared by either a PR2 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 PR2 period count match.

**FIGURE 27-11: LOW LEVEL RESET, EDGE-TRIGGERED HARDWARE LIMIT ONE-SHOT MODE TIMING DIAGRAM (MODE = 01110)**



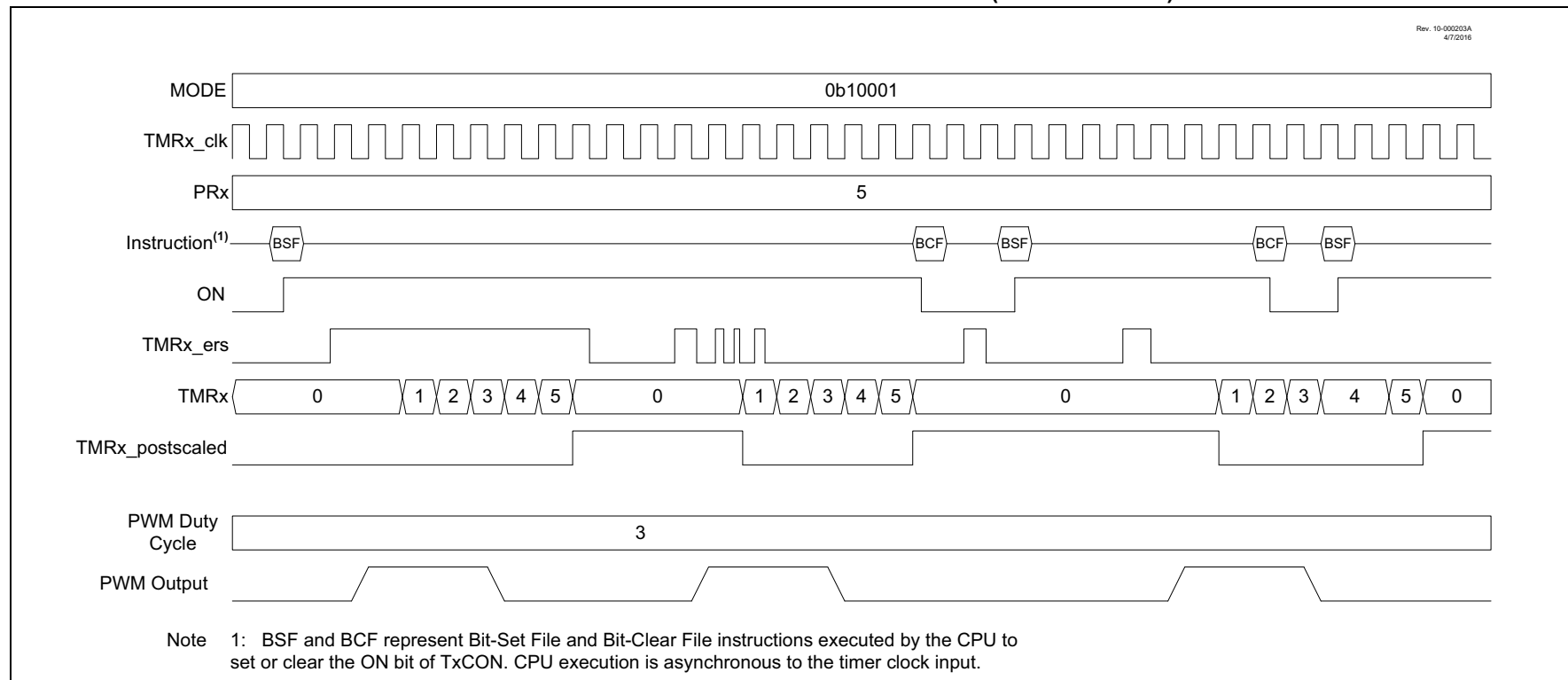
### 27.5.9 EDGE-TRIGGERED MONOSTABLE MODES

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 PR2 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 PR2 value. While the timer is incrementing, additional edges on the external Reset signal will not affect the CCP PWM.

**FIGURE 27-12: RISING EDGE-TRIGGERED MONOSTABLE MODE TIMING DIAGRAM (MODE = 10001)**



### 27.5.10 LEVEL-TRIGGERED HARDWARE LIMIT ONE-SHOT MODES

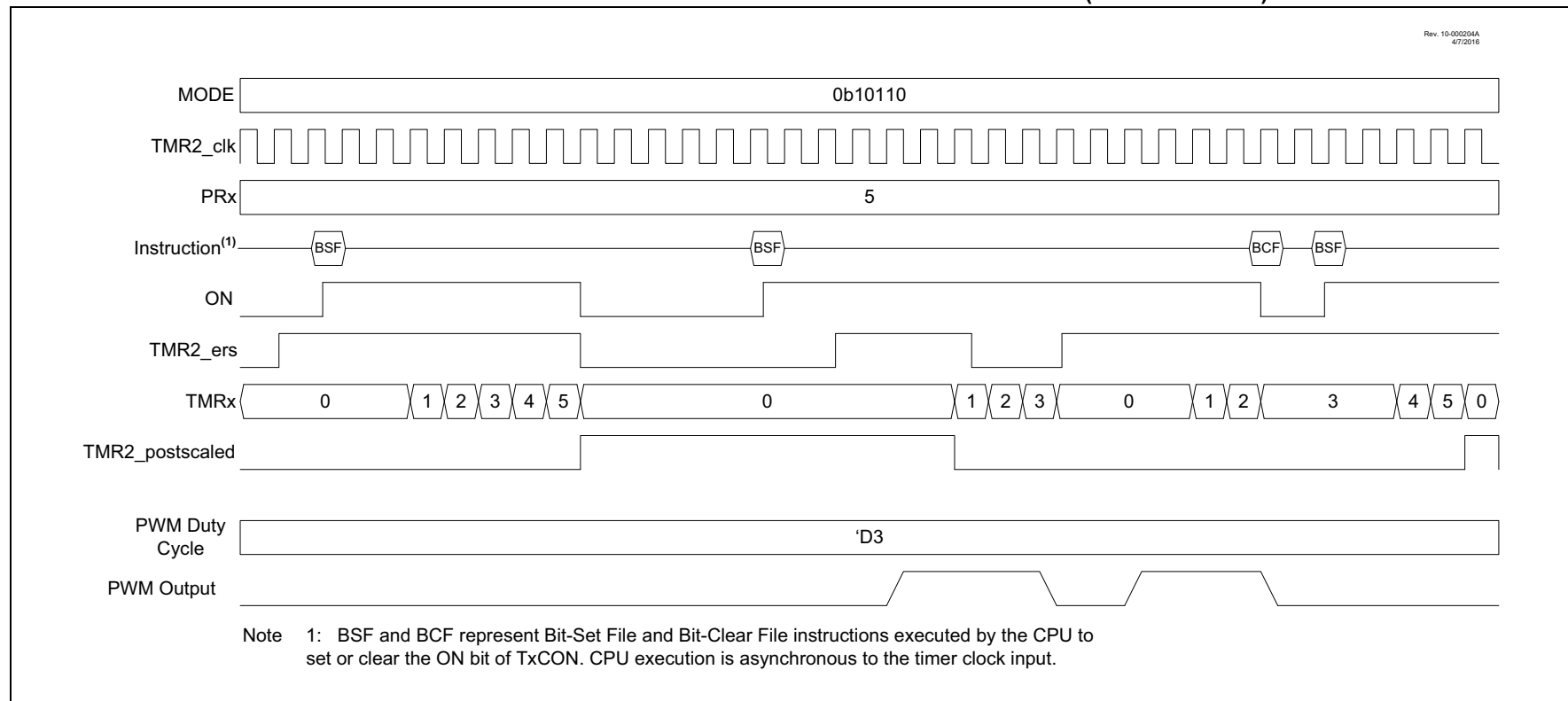
The Level-Triggered Hardware Limit One-Shot modes hold the timer in Reset on an external Reset level and start counting when both the ON bit is set and the external signal is not at the Reset level. If one of either the external signal is not in Reset or the ON bit is set then the other signal being 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 PR2 period count, the timer is reset and the ON bit is cleared. When the ON bit is cleared by either a PR2 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 27-13: LEVEL-TRIGGERED HARDWARE LIMIT ONE-SHOT MODE TIMING DIAGRAM (MODE = 10110)**



## 27.6 Timer2 Operation During Sleep

When PSYNC = 1, Timer2 cannot be operated while the processor is in Sleep mode. The contents of the TMR2 and PR2 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.

## 27.7 Register Definitions: Timer2 Control

**REGISTER 27-1: T2CLKCON: TIMER2 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	<b>Unimplemented:</b> Read as '0'
bit 3-0	<b>CS&lt;3:0&gt;:</b> Timer2 Clock Select bits
	1111 = Reserved
	1110 = LC4_out
	1101 = LC3_out
	1100 = LC2_out
	1011 = LC1_out
	1010 = ZCD1_output
	1001 = NCO1_out
	1000 = CLKR
	0111 = SOSC
	0110 = MFINTOSC (31.25 kHz)
	0101 = MFINTOSC (500 kHz)
	0100 = LFINTOSC
	0011 = HFINTOSC (32 MHz)
	0010 = Fosc
	0001 = Fosc/4
	0000 = T2CKIPPS

## REGISTER 27-2: T2CON: TIMER2 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 <sup>(1)</sup>	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	<b>ON:</b> Timer2 On bit 1 = Timer2 is on 0 = Timer2 is off: all counters and state machines are reset
bit 6-4	<b>CKPS&lt;2:0&gt;:</b> 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	<b>OUTPS&lt;3:0&gt;:</b> Timer2 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 27.5 “Operation Examples”](#).



## REGISTER 27-3: T2HLT: TIMER2 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 <sup>(1, 2)</sup>	CKPOL <sup>(3)</sup>	CKSYNC <sup>(4, 5)</sup>	MODE<4:0> <sup>(6, 7)</sup>				
bit 7							bit 0

### Legend:

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

- bit 7      **PSYNC:** Timer2 Prescaler Synchronization Enable bit<sup>(1, 2)</sup>  
 1 = TMR2 Prescaler Output is synchronized to Fosc/4  
 0 = TMR2 Prescaler Output is not synchronized to Fosc/4
- bit 6      **CKPOL:** Timer2 Clock Polarity Selection bit<sup>(3)</sup>  
 1 = Falling edge of input clock clocks timer/prescaler  
 0 = Rising edge of input clock clocks timer/prescaler
- bit 5      **CKSYNC:** Timer2 Clock Synchronization Enable bit<sup>(4, 5)</sup>  
 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>:** Timer2 Control Mode Selection bits<sup>(6, 7)</sup>  
 See [Table 27-1](#).

- Note 1:** Setting this bit ensures that reading TM2x 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 TMR2 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 TMR2).
- 7:** When TMR2 = PR2, the next clock clears TMR2, regardless of the operating mode.

## REGISTER 27-4: T2RST: TIMER2 EXTERNAL RESET SIGNAL 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
—	—	—	—	RSEL<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      **RSEL<3:0>:** Timer2 External Reset Signal Source Selection bits

- 1111 = Reserved
- 1101 = LC4\_out
- 1100 = LC3\_out
- 1011 = LC2\_out
- 1010 = LC1\_out
- 1001 = ZCD1\_output
- 1000 = C2OUT\_sync
- 0111 = C1OUT\_sync
- 0110 = PWM6\_out
- 0101 = PWM5\_out
- 0100 = PWM4\_out
- 0011 = PWM3\_out
- 0010 = CCP2\_out
- 0001 = CCP1\_out
- 0000 = T2INPPS

**TABLE 27-2: 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>				319
CCP2CON	EN	—	OUT	FMT	MODE<3:0>				319
INTCON	GIE	PEIE	—	—	—	—	—	INTEDG	121
PIE4	—	—	—	—	—	—	TMR2IE	TMR1IE	134
PIR4	—	—	—	—	—	—	TMR2IF	TMR1IF	134
PR2	Timer2 Module Period Register								291*
TMR2	Holding Register for the 8-bit TMR2 Register								292*
T2CON	ON	CKPS<2:0>			OUTPS<3:0>				308
T2CLKCON	—	—	—	—	CS<3:0>				307
T2RST	—	—	—	—	RSEL<3:0>				310
T2HLT	PSYNC	CKPOL	CKSYNC	MODE<4:0>					309

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

\* Page provides register information.

## 28.0 CAPTURE/COMPARE/PWM MODULES

The Capture/Compare/PWM module is a peripheral that 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.

The Capture/Compare/PWM modules available are shown in [Table 28-1](#).

**TABLE 28-1: AVAILABLE CCP MODULES**

Device	CCP1	CCP2
PIC16(L)F15354/55	•	•

The Capture and Compare functions are identical for all CCP modules.

**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 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.

## 28.1 Capture Mode

Capture mode makes use of the 16-bit Timer1 resource. When an event occurs on the capture source, 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 CCPxMODE<3:0> bits of the CCPxCON register:

- Every falling edge
- Every rising edge
- Every 4th rising edge
- Every 16th rising edge

When a capture is made, the Interrupt Request Flag bit CCPxIF of the PIR6 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 28-1 shows a simplified diagram of the capture operation.

### 28.1.1 CAPTURE SOURCES

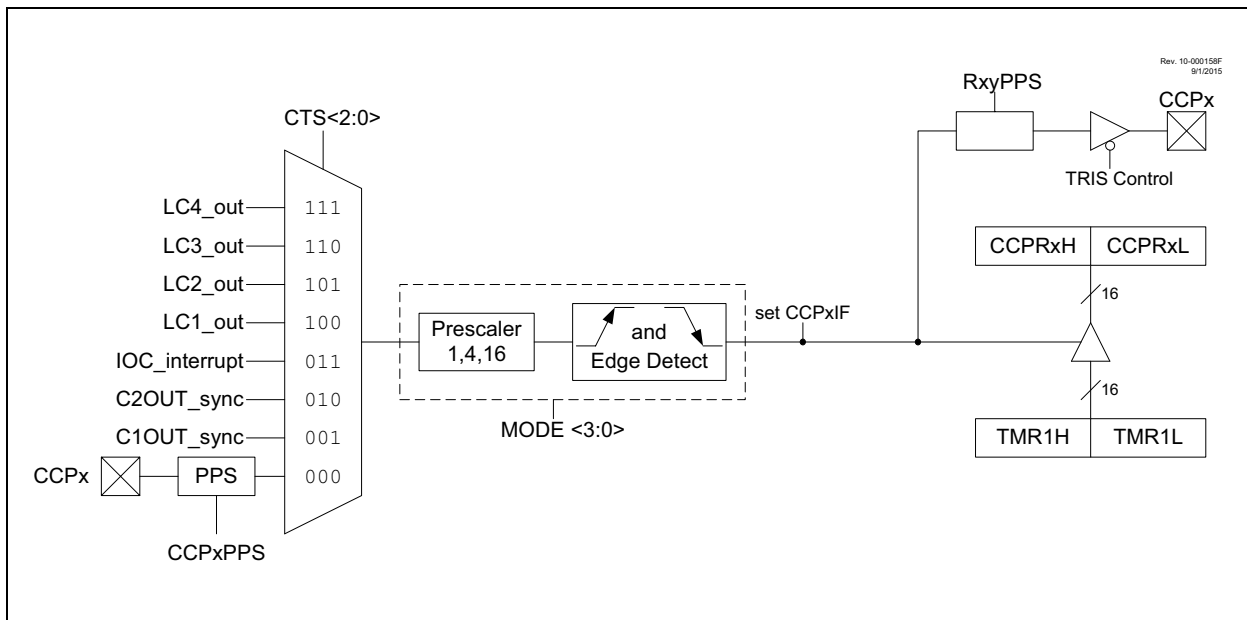
In Capture mode, the CCPx pin should be configured as an input by setting the associated TRIS control bit.

**Note:** If the CCPx pin is configured as an output, a write to the port can cause a capture condition.

The capture source is selected by configuring the CCPxCTS<2:0> bits of the CCPxCON register. The following sources can be selected:

- CCPxPPS input
- C1OUT\_sync
- C2OUT\_sync
- IOC\_interrupt
- LC1\_out
- LC2\_out
- LC3\_out
- LC4\_out

**FIGURE 28-1: CAPTURE MODE OPERATION BLOCK DIAGRAM**



## 28.1.2 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 26.0 “Timer1 Module with Gate Control”](#) for more information on configuring Timer1.

## 28.1.3 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 PIE6 register clear to avoid false interrupts. Additionally, the user should clear the CCPxIF interrupt flag bit of the PIR6 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).

## 28.1.4 CCP PRESCALER

There are four prescaler settings specified by the CCPxMODE<3:0> bits of the CCPxCON register. Whenever the CCP module is turned off, or the CCP module is not in Capture mode, the prescaler counter is cleared. Any Reset will clear the prescaler counter.

Switching from one capture prescaler to another does not clear the prescaler and may generate a false interrupt. To avoid this unexpected operation, turn the module off by clearing the CCPxCON register before changing the prescaler. [Example 28-1](#) demonstrates the code to perform this function.

### EXAMPLE 28-1: CHANGING BETWEEN CAPTURE PRESCALERS

```
BANKSEL CCPxCON    ;Set Bank bits to point
                   ;to CCPxCON
CLRF    CCPxCON    ;Turn CCP module off
MOVLW  NEW_CAPT_PS ;Load the W reg with
                   ;the new prescaler
MOVWF  CCPxCON    ;move value and CCP ON
                   ;Load CCPxCON with this
                   ;value
```

## 28.1.5 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.

## 28.2 Compare Mode

Compare mode makes use of the 16-bit Timer1 resource. The 16-bit value of the CCPRxH:CCPRxL register pair is constantly compared against the 16-bit value of the TMR1H:TMR1L register pair. When a match occurs, one of the following events can occur:

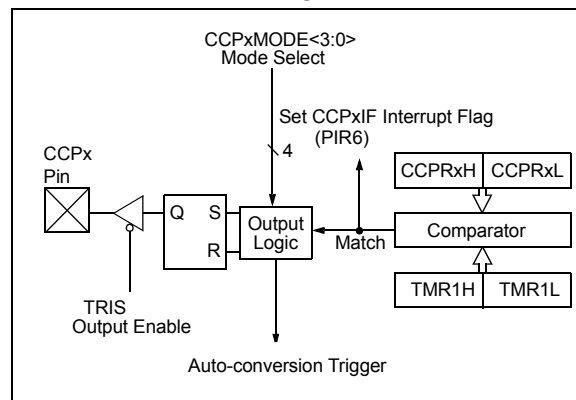
- Toggle the CCPx output
- Set the CCPx output
- Clear the CCPx output
- Generate an Auto-conversion Trigger
- Generate a Software Interrupt

The action on the pin is based on the value of the CCPxMODE<3:0> control bits of the CCPxCON register. At the same time, the interrupt flag CCPxIF bit is set, and an ADC conversion can be triggered, if selected.

All Compare modes can generate an interrupt and trigger and ADC conversion.

[Figure 28-2](#) shows a simplified diagram of the compare operation.

**FIGURE 28-2: COMPARE MODE OPERATION BLOCK DIAGRAM**



## 28.2.1 CCPX PIN CONFIGURATION

The software must configure the CCPx pin as an output by clearing the associated TRIS bit and defining the appropriate output pin through the RxyPPS registers. See [Section 15.0 “Peripheral Pin Select \(PPS\) Module”](#) for more details.

The CCP output can also be used as an input for other peripherals.

**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.

## 28.2.2 TIMER1 MODE RESOURCE

In Compare mode, Timer1 must be running in either Timer mode or Synchronized Counter mode. The compare operation may not work in Asynchronous Counter mode.

See [Section 26.0 “Timer1 Module with Gate Control”](#) for more information on configuring Timer1.

**Note:** Clocking Timer1 from the system clock (Fosc) should not be used in 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.

## 28.2.3 AUTO-CONVERSION TRIGGER

All CCPx modes set the CCP interrupt flag (CCPxF). When this flag is set and a match occurs, an Auto-conversion Trigger can take place if the CCP module is selected as the conversion trigger source.

Refer to [Section 20.2.4 “Auto-Conversion Trigger”](#) for more information.

**Note:** 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

## 28.2.4 COMPARE DURING SLEEP

Since Fosc is shut down during Sleep mode, the Compare mode will not function properly during Sleep, unless the timer is running. The device will wake on interrupt (if enabled).

## 28.3 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 28-3](#) shows a typical waveform of the PWM signal.

### 28.3.1 STANDARD PWM OPERATION

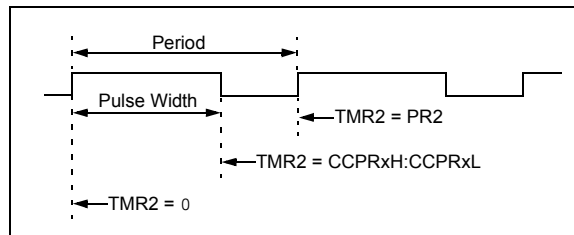
The standard PWM mode generates a Pulse-Width Modulation (PWM) signal on the CCPx pin with up to ten bits of resolution. The period, duty cycle, and resolution are controlled by the following registers:

- PR2 registers
- T2CON registers
- CCPRxL registers
- CCPxCON registers

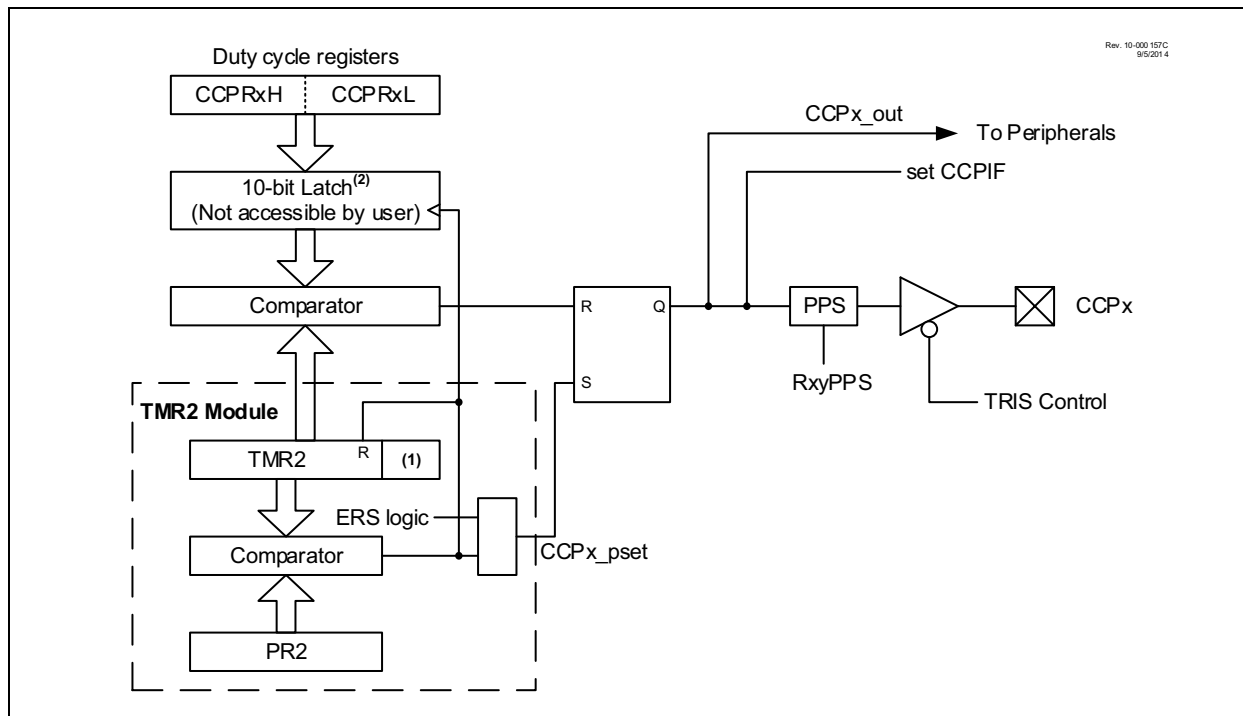
[Figure 28-4](#) shows a simplified block diagram of PWM operation.

**Note:** The corresponding TRIS bit must be cleared to enable the PWM output on the CCPx pin.

**FIGURE 28-3: CCP PWM OUTPUT SIGNAL**



**FIGURE 28-4: SIMPLIFIED PWM BLOCK DIAGRAM**



## 28.3.2 SETUP FOR PWM OPERATION

The following steps should be taken when configuring the CCP module for standard PWM operation:

1. Use the desired output pin RxyPPS control to select CCPx as the source and disable the CCPx pin output driver by setting the associated TRIS bit.
2. Load the PR2 register with the PWM period value.
3. Configure the CCP module for the PWM mode by loading the CCPxCON register with the appropriate values.
4. Load the CCPRxL register, and the CCPRxH register with the PWM duty cycle value and configure the CCPxFMT bit of the CCPxCON register to set the proper register alignment.
5. Configure and start Timer2:
  - Clear the TMR2IF interrupt flag bit of the PIR4 register. See Note below.
  - Configure the CKPS bits of the T2CON register with the Timer prescale value.
  - Enable the Timer by setting the Timer2 ON bit of the T2CON register.

6. Enable PWM output pin:

- Wait until the Timer overflows and the TMR2IF bit of the PIR4 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.

## 28.3.3 CCP/PWM CLOCK SELECTION

The PIC16(L)F15354/55 allows each individual CCP and PWM module to select the timer source that controls the module. Each module has an independent selection.



## 28.3.4 TIMER2 TIMER RESOURCE

This device has a newer version of the Timer2 module that has many new modes, which allow for greater customization and control of the PWM signals than on older parts. Refer to [Section 27.5 “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

## 28.3.5 PWM PERIOD

The PWM period is specified by the PR2 register of Timer2. The PWM period can be calculated using the formula of [Equation 28-1](#).

### EQUATION 28-1: PWM PERIOD

$$PWM\ Period = [(PR2) + 1] \cdot 4 \cdot T_{OSC} \cdot (TMR2\ Prescale\ Value)$$

**Note 1:**  $T_{OSC} = 1/F_{OSC}$

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

- TMR2 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 transferred from the CCPRxL/H register pair into a 10-bit buffer.

**Note:** The Timer postscaler (see [Section 27.4 “Timer2 Interrupt”](#)) is not used in the determination of the PWM frequency.

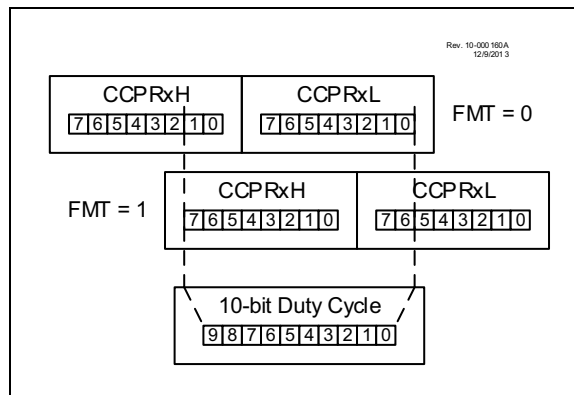
## 28.3.6 PWM DUTY CYCLE

The PWM duty cycle is specified by writing a 10-bit value to the CCPRxH:CCPRxL register pair. The alignment of the 10-bit value is determined by the CCPRxFMT bit of the CCPxCON register (see [Figure 28-5](#)). The CCPRxH:CCPRxL register pair can be written to at any time; however the duty cycle value is not latched into the 10-bit buffer until after a match between PR2 and TMR2.

[Equation 28-2](#) is used to calculate the PWM pulse width.

[Equation 28-3](#) is used to calculate the PWM duty cycle ratio.

**FIGURE 28-5: PWM 10-BIT ALIGNMENT**



### EQUATION 28-2: PULSE WIDTH

$$Pulse\ Width = (CCPRxH:CCPRxL\ register\ pair) \cdot T_{OSC} \cdot (TMR2\ Prescale\ Value)$$

### EQUATION 28-3: DUTY CYCLE RATIO

$$Duty\ Cycle\ Ratio = \frac{(CCPRxH:CCPRxL\ register\ pair)}{4(PR2 + 1)}$$

CCPRxH:CCPRxL register pair are used to double buffer the PWM duty cycle. This double buffering provides for glitchless PWM operation.

The 8-bit timer TMR2 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 prescaler is set to 1:1.

When the 10-bit time base matches the CCPRxH:CCPRxL register pair, then the CCPx pin is cleared (see [Figure 28-4](#)).

## 28.3.7 PWM RESOLUTION

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

The maximum PWM resolution is ten bits when PR2 is 255. The resolution is a function of the PR2 register value as shown by [Equation 28-4](#).

### EQUATION 28-4: 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 28-2: 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
PR2 Value	0xFF	0xFF	0xFF	0x3F	0x1F	0x17
Maximum Resolution (bits)	10	10	10	8	7	6.6

**TABLE 28-3: 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
PR2 Value	0x65	0x65	0x65	0x19	0x0C	0x09
Maximum Resolution (bits)	8	8	8	6	5	5

### 28.3.8 OPERATION IN SLEEP MODE

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

### 28.3.9 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 9.0 “Oscillator Module \(with Fail-Safe Clock Monitor\)”](#) for additional details.

### 28.3.10 EFFECTS OF RESET

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

## 28.4 Register Definitions: CCP Control

Long bit name prefixes for the CCP peripherals are shown in [Section 1.1 “Register and Bit Naming Conventions”](#).

**TABLE 28-4: LONG BIT NAMES PREFIXES FOR CCP PERIPHERALS**

Peripheral	Bit Name Prefix
CCP1	CCP1
CCP2	CCP2

**REGISTER 28-1: 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  
MODE = Capture mode  
 Unused  
MODE = Compare mode  
 Unused  
MODE = PWM mode  
 1 = Left-aligned format  
 0 = Right-aligned format

bit 3-0    **MODE<3:0>:** CCPx Mode Select bits<sup>(1)</sup>  
 1111 - 1100 = PWM mode (Timer2 as the timer source)  
 1110 = Reserved  
 1101 =Reserved  
 1100 = Reserved

1011 =Compare mode: output will pulse 0-1-0; Clears TMR1  
 1010 =Compare mode: output will pulse 0-1-0  
 1001 =Compare mode: clear output on compare match  
 1000 =Compare mode: set output on compare match

0111 =Capture mode: every 16th rising edge of CCPx input  
 0110 =Capture mode: every 4th rising edge of CCPx input  
 0101 =Capture mode: every rising edge of CCPx input  
 0100 =Capture mode: every falling edge of CCPx input

0011 =Capture mode: every edge of CCPx input  
 0010 =Compare mode: toggle output on match  
 0001 =Compare mode: toggle output on match; clear TMR1  
 0000 =Capture/Compare/PWM off (resets CCPx module)

**Note 1:** All modes will set the CCPxIF bit, and will trigger an ADC conversion if CCPx is selected as the ADC trigger source.

# PIC16(L)F15354/55

## REGISTER 28-2: CCPxCAP: CAPTURE INPUT SELECTION REGISTER

U-0	U-0	U-0	U-0	U-0	R/W-0/x	R/W-0/x	R/W-0/x
—	—	—	—	—	CTS<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 Reset  
'1' = Bit is set                              '0' = Bit is cleared

bit 7-3                      **Unimplemented:** Read as '0'  
bit 2-0                      **CTS<2:0>:** Capture Trigger Input Selection bits

CTS	CCP1.capture	CCP2.capture
111	LC4_out	
110	LC3_out	
101	LC2_out	
100	LC1_out	
011	IOC_interrupt	
010	C2OUT	
001	C1OUT	
000	CCP1PPS	CCP2PPS

## REGISTER 28-3: CCPRxL REGISTER: CCPx REGISTER LOW BYTE

R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x
CCPRx<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                      CCPxMODE = Capture mode  
**CCPRxL<7:0>:** Capture value of TMR1L  
CCPxMODE = Compare mode  
**CCPRxL<7:0>:** LS Byte compared to TMR1L  
CCPxMODE = PWM modes when CCPxFMT = 0:  
**CCPRxL<7:0>:** Pulse-width Least Significant eight bits  
CCPxMODE = PWM modes when CCPxFMT = 1:  
**CCPRxL<7:6>:** Pulse-width Least Significant two bits  
**CCPRxL<5:0>:** Not used.

## REGISTER 28-4: CCPRxH REGISTER: CCPx REGISTER HIGH BYTE

R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x
CCPRx<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      CCPxMODE = Capture mode  
**CCPRxH<7:0>**: Captured value of TMR1H  
CCPxMODE = Compare mode  
**CCPRxH<7:0>**: MS Byte compared to TMR1H  
CCPxMODE = PWM modes when CCPxFMT = 0:  
**CCPRxH<7:2>**: Not used  
**CCPRxH<1:0>**: Pulse-width Most Significant two bits  
CCPxMODE = PWM modes when CCPxFMT = 1:  
**CCPRxH<7:0>**: Pulse-width Most Significant eight bits

**TABLE 28-5: SUMMARY OF REGISTERS ASSOCIATED WITH CCPx**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page		
INTCON	GIE	PEIE	—	—	—	—	—	INTEDG	121		
PIR6	—	—	—	—	—	—	CCP2IF	CCP1IF	144		
PIE6	—	—	—	—	—	—	CCP2IE	CCP1IE	136		
CCP1CON	EN	—	OUT	FMT	MODE<3:0>				319		
CCP1CAP	—	—	—	—	—	CTS<2:0>				320	
CCPR1L	Capture/Compare/PWM Register 1 (LSB)								320		
CCPR1H	Capture/Compare/PWM Register 1 (MSB)								321		
CCP2CON	EN	—	OUT	FMT	MODE<3:0>				319		
CCP2CAP	—	—	—	—	—	CTS<2:0>				320	
CCPR2L	Capture/Compare/PWM Register 1 (LSB)								320		
CCPR2H	Capture/Compare/PWM Register 1 (MSB)								320		
CCP1PPS	—	—	CCP1PPS<5:0>							199	
CCP2PPS	—	—	CCP2PPS<5:0>							199	
RxyPPS	—	—	—	RxyPPS<4:0>							200
ADACT	—	—	—	ADACT<4:0>							236
CLCxSELY	—	—	—	LCxDyS<4:0>							364
CWG1DAT	—	—	—	—	DAT<3:0>				353		

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

## 29.0 PULSE-WIDTH MODULATION (PWM)

The PWMx modules generate Pulse-Width Modulated (PWM) signals of varying frequency and duty cycle.

In addition to the CCP modules, the PIC16(L)F15354/55 devices contain four 10-bit PWM modules (PWM3, PWM4, PWM5 and PWM6). The PWM modules reproduce the PWM capability of the CCP modules.

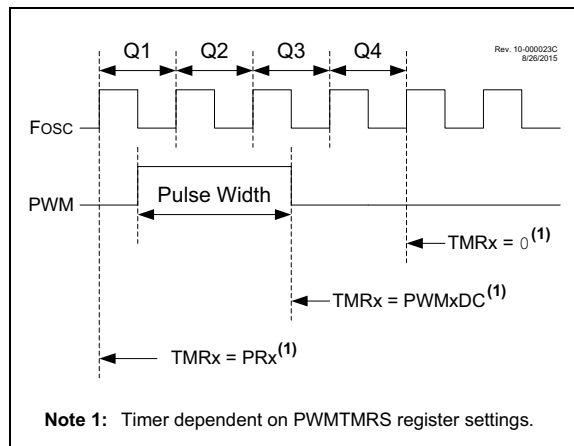
**Note:** The PWM3/4/5/6 modules are four instances of the same PWM module design. Throughout this section, the lower case 'x' in register and bit names is a generic reference to the PWM module number (which should be substituted with 3, or 4, or 5 or 6 during code development). For example, the control register is generically described in this chapter as PWMxCON, but the actual device registers are PWM3CON, PWM4CON, PWM5CON and PWM6CON. Similarly, the PWMxEN bit represents the PWM3EN, PWM4EN, PWM5EN and PWM6EN bits.

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 (pulse width), and the low portion of the signal is considered the 'off' state. 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. 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.

Figure 29-1 shows a typical waveform of the PWM signal.

FIGURE 29-1: PWM OUTPUT



Note 1: Timer dependent on PWMTMRS register settings.

## 29.1 Standard PWM Mode

The standard PWM mode generates a Pulse-Width Modulation (PWM) signal on the PWMx pin with up to ten bits of resolution. The period, duty cycle, and resolution are controlled by the following registers:

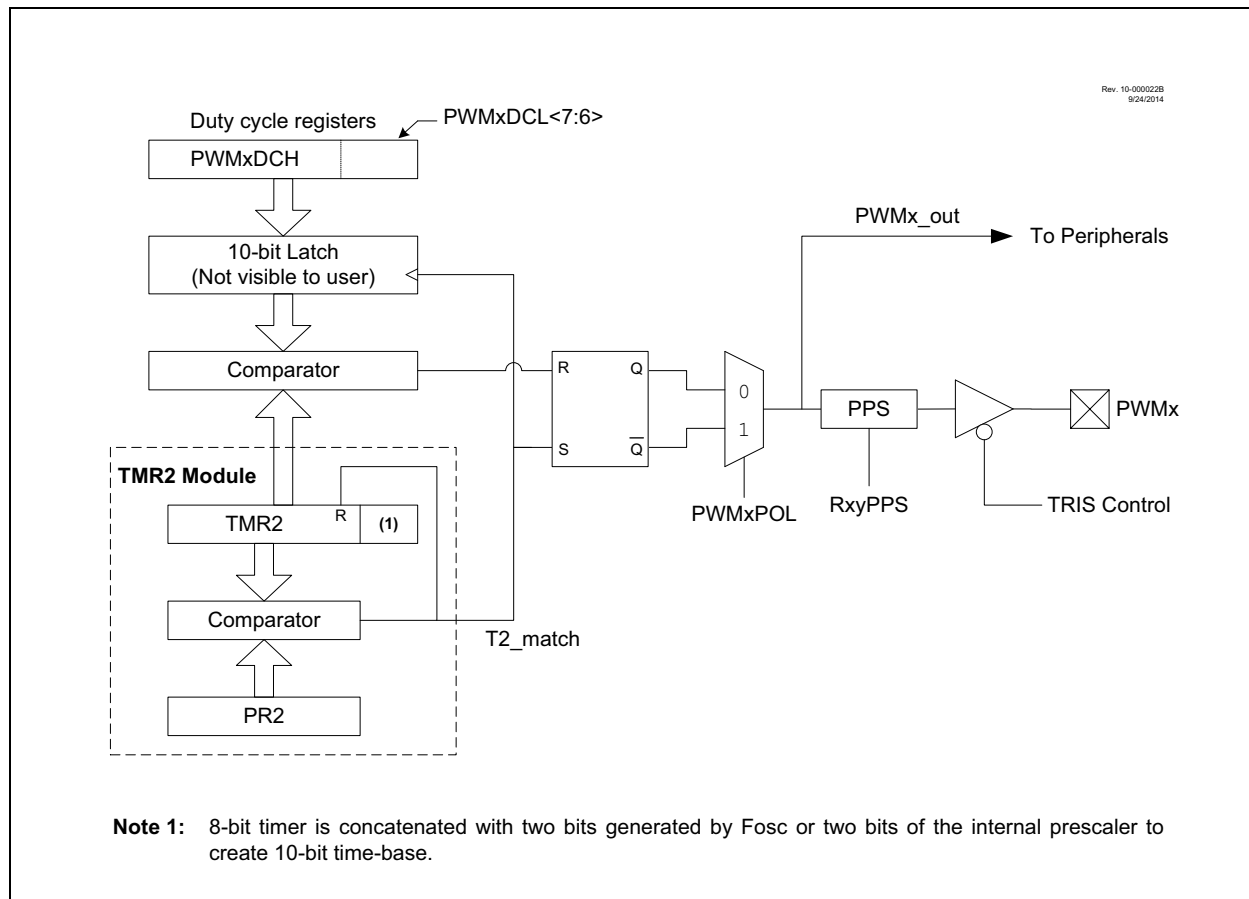
- TMR2 register
- PR2 register
- PWMxCON registers
- PWMxDCH registers
- PWMxDCL registers

Figure 29-2 shows a simplified block diagram of PWM operation.

If PWMPOL = 0, the default state of the output is '0'. If PWMPOL = 1, the default state is '1'. If PWMEN = 0, the output will be the default state.

**Note:** The corresponding TRIS bit must be cleared to enable the PWM output on the PWMx pin

**FIGURE 29-2: SIMPLIFIED PWM BLOCK DIAGRAM**





## 29.1.1 PWM CLOCK SELECTION

The PIC16(L)F15354/55 allows each individual CCP and PWM module to select the timer source that controls the module. Each module has an independent selection.

## 29.1.2 USING THE TMR2 WITH THE PWM MODULE

This device has a newer version of the TMR2 module that has many new modes, which allow for greater customization and control of the PWM signals than on older parts. Refer to [Section 27.5 “Operation Examples”](#) for examples of PWM signal generation using the different modes of Timer2.

**Note:** PWM operation requires that the timer used as the PWM time base has the FOSC/4 clock source selected.

## 29.1.3 PWM PERIOD

Referring to [Figure 29-1](#), the PWM output has a period and a pulse width. The frequency of the PWM is the inverse of the period (1/period).

The PWM period is specified by writing to the PR2 register. The PWM period can be calculated using the following formula:

### EQUATION 29-1: PWM PERIOD

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

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

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

- TMR2 is cleared
- The PWMx pin is set (Exception: If the PWM duty cycle = 0%, the pin will not be set.)
- The PWM pulse width is latched from PWMxDC.

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

## 29.1.4 PWM DUTY CYCLE

The PWM duty cycle is specified by writing a 10-bit value to the PWMxDC register. The PWMxDCH contains the eight MSBs and the PWMxDCL<7:6> bits contain the two LSBs.

The PWMDC register is double-buffered and can be updated at any time. This double buffering is essential for glitch-free PWM operation. New values take effect when TMR2 = PR2. Note that PWMDC is left-justified.

The 8-bit timer TMR2 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 prescaler is set to 1:1.

[Equation 29-2](#) is used to calculate the PWM pulse width.

[Equation 29-3](#) is used to calculate the PWM duty cycle ratio.

### EQUATION 29-2: PULSE WIDTH

$$Pulse\ Width = (PWMxDC) \cdot TOSC \cdot (TMR2\ Prescale\ Value)$$

### EQUATION 29-3: DUTY CYCLE RATIO

$$Duty\ Cycle\ Ratio = \frac{(PWMxDC)}{4(PR2 + 1)}$$

## 29.1.5 PWM RESOLUTION

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

The maximum PWM resolution is ten bits when PR2 is 255. The resolution is a function of the PR2 register value as shown by [Equation 29-4](#).

### EQUATION 29-4: PWM RESOLUTION

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

## 29.1.6 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.

## 29.1.7 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 9.0 “Oscillator Module \(with Fail-Safe Clock Monitor\)”](#) for additional details.

## 29.1.8 EFFECTS OF RESET

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

**TABLE 29-1: 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
PR2 Value	0xFF	0xFF	0xFF	0x3F	0x1F	0x17
Maximum Resolution (bits)	10	10	10	8	7	6.6

**TABLE 29-2: 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
PR2 Value	0xFF	0xFF	0xFF	0x3F	0x1F	0x17
Maximum Resolution (bits)	10	10	10	8	7	6.6

## 29.1.9 SETUP FOR PWM OPERATION

The following steps should be taken when configuring the module for using the PWMx outputs:

1. Disable the PWMx pin output driver(s) by setting the associated TRIS bit(s).
2. Configure the PWM output polarity by configuring the PWMxPOL bit of the PWMxCON register.
3. Load the PR2 register with the PWM period value, as determined by [Equation 29-1](#).
4. Load the PWMxDCH register and bits <7:6> of the PWMxDCL register with the PWM duty cycle value, as determined by [Equation 29-2](#).
5. Configure and start Timer2:
  - Clear the TMR2IF interrupt flag bit of the PIR4 register.
  - Select the Timer2 prescale value by configuring the CKPS<2:0> bits of the T2CON register.
  - Enable Timer2 by setting the Timer2 ON bit of the T2CON register.

6. Wait until the TMR2IF is set.

7. When the TMR2IF flag bit is set:

- Clear the associated TRIS bit(s) to enable the output driver.
- Route the signal to the desired pin by configuring the RxyPPS register.
- Enable the PWMx module by setting the PWMxEN bit of the PWMxCON register.

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 the PWM module can be enabled during Step 2 by setting the PWMxEN bit of the PWMxCON register.

## 29.2 Register Definitions: PWM Control

### REGISTER 29-1: PWMxCON: PWM CONTROL REGISTER

R/W-0/0	U-0	R-0	R/W-0/0	U-0	U-0	U-0	U-0
PWMxEN	—	PWMxOUT	PWMxPOL	—	—	—	—
bit 7							bit 0

#### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

u = Bit is unchanged

x = Bit is unknown

-n/n = Value at POR and BOR/Value at all other Resets

'1' = Bit is set

'0' = Bit is cleared

- bit 7      **PWMxEN:** PWM Module Enable bit  
           1 = PWM module is enabled  
           0 = PWM module is disabled
- bit 6      **Unimplemented:** Read as '0'
- bit 5      **PWMxOUT:** PWM Module Output Level when Bit is Read
- bit 4      **PWMxPOL:** PWMx Output Polarity Select bit  
           1 = PWM output is active-low  
           0 = PWM output is active-high
- bit 3-0    **Unimplemented:** Read as '0'

## REGISTER 29-2: 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
PWMxDC<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      **PWMxDC<9:2>**: PWM Duty Cycle Most Significant bits  
 These bits are the MSBs of the PWM duty cycle. The two LSBs are found in PWMxDCL Register.

## REGISTER 29-3: 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
PWMxDC<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      **PWMxDC<1:0>**: PWM Duty Cycle Least Significant bits  
 These bits are the LSBs of the PWM duty cycle. The MSBs are found in PWMxDCH Register.

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

**TABLE 29-3: SUMMARY OF REGISTERS ASSOCIATED WITH PWMx**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
T2CON	ON	CKPS<2:0>			OUTPS<3:0>				<a href="#">308</a>
T2TMR	Holding Register for the 8-bit TMR2 Register								<a href="#">292*</a>
T2PR	TMR2 Period Register								<a href="#">291*</a>
RxyPPS	—	—	—	RxyPPS<4:0>				<a href="#">200</a>	
CWG1DAT	—	—	—	—	DAT<3:0>				<a href="#">353</a>
CLCxSELy	—	—	LCxDyS<5:0>						<a href="#">364</a>

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

\* Page with Register information.

## 30.0 COMPLEMENTARY WAVEFORM GENERATOR (CWG) MODULE

The Complementary Waveform Generator (CWG) produces half-bridge, full-bridge, and steering of PWM waveforms. It is backwards compatible with previous ECCP functions.

The CWG has the following features:

- Six operating modes:
  - Synchronous Steering mode
  - Asynchronous Steering mode
  - Full-Bridge mode, Forward
  - Full-Bridge mode, Reverse
  - Half-Bridge mode
  - Push-Pull mode
- Output polarity control
- Output steering
  - Synchronized to rising event
  - Immediate effect
- Independent 6-bit rising and falling event dead-band timers
  - Clocked dead band
  - Independent rising and falling dead-band enables
- Auto-shutdown control with:
  - Selectable shutdown sources
  - Auto-restart enable
  - Auto-shutdown pin override control

The CWG modules available are shown in [Table 30-1](#).

**TABLE 30-1: AVAILABLE CWG MODULES**

Device	CWG1
PIC16(L)F15354/55	•

## 30.1 Fundamental Operation

The CWG module can operate in six different modes, as specified by MODE of the CWG1CON0 register:

- Half-Bridge mode ([Figure 30-9](#))
- Push-Pull mode ([Figure 30-2](#))
  - Full-Bridge mode, Forward ([Figure 30-3](#))
  - Full-Bridge mode, Reverse ([Figure 30-3](#))
- Steering mode ([Figure 30-10](#))
- Synchronous Steering mode ([Figure 30-11](#))

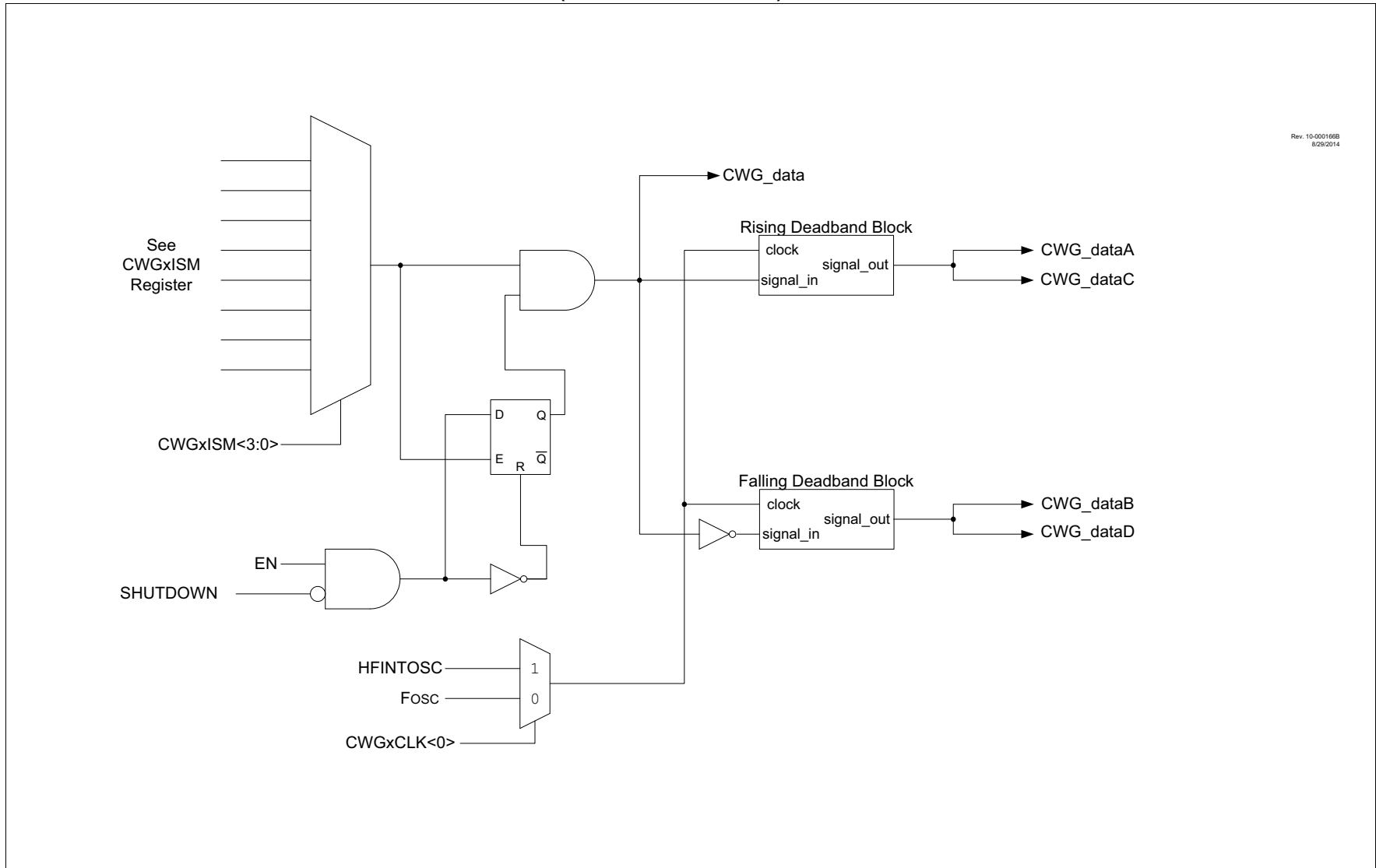
It may be necessary to guard against the possibility of circuit faults or a feedback event arriving too late or not at all. In this case, the active drive must be terminated before the Fault condition causes damage. Thus, all output modes support auto-shutdown, which is covered in [30.10 “Auto-Shutdown”](#).

### 30.1.1 HALF-BRIDGE MODE

In Half-Bridge mode, two output signals are generated as true and inverted versions of the input as illustrated in [Figure 30-9](#). A non-overlap (dead-band) time is inserted between the two outputs as described in [Section 30.5 “Dead-Band Control”](#).

The unused outputs CWG1C and CWG1D drive similar signals, with polarity independently controlled by the POLC and POLD bits of the CWG1CON1 register, respectively.

**FIGURE 30-1: SIMPLIFIED CWG BLOCK DIAGRAM (HALF-BRIDGE MODE)**



## 30.1.2 PUSH-PULL MODE

In Push-Pull mode, two output signals are generated, alternating copies of the input as illustrated in [Figure 30-2](#). This alternation creates the push-pull effect required for driving some transformer-based power supply designs.

The push-pull sequencer is reset whenever  $EN = 0$  or if an auto-shutdown event occurs. The sequencer is clocked by the first input pulse, and the first output appears on CWG1A.

The unused outputs CWG1C and CWG1D drive copies of CWG1A and CWG1B, respectively, but with polarity controlled by the POLC and POLD bits of the CWG1CON1 register, respectively.

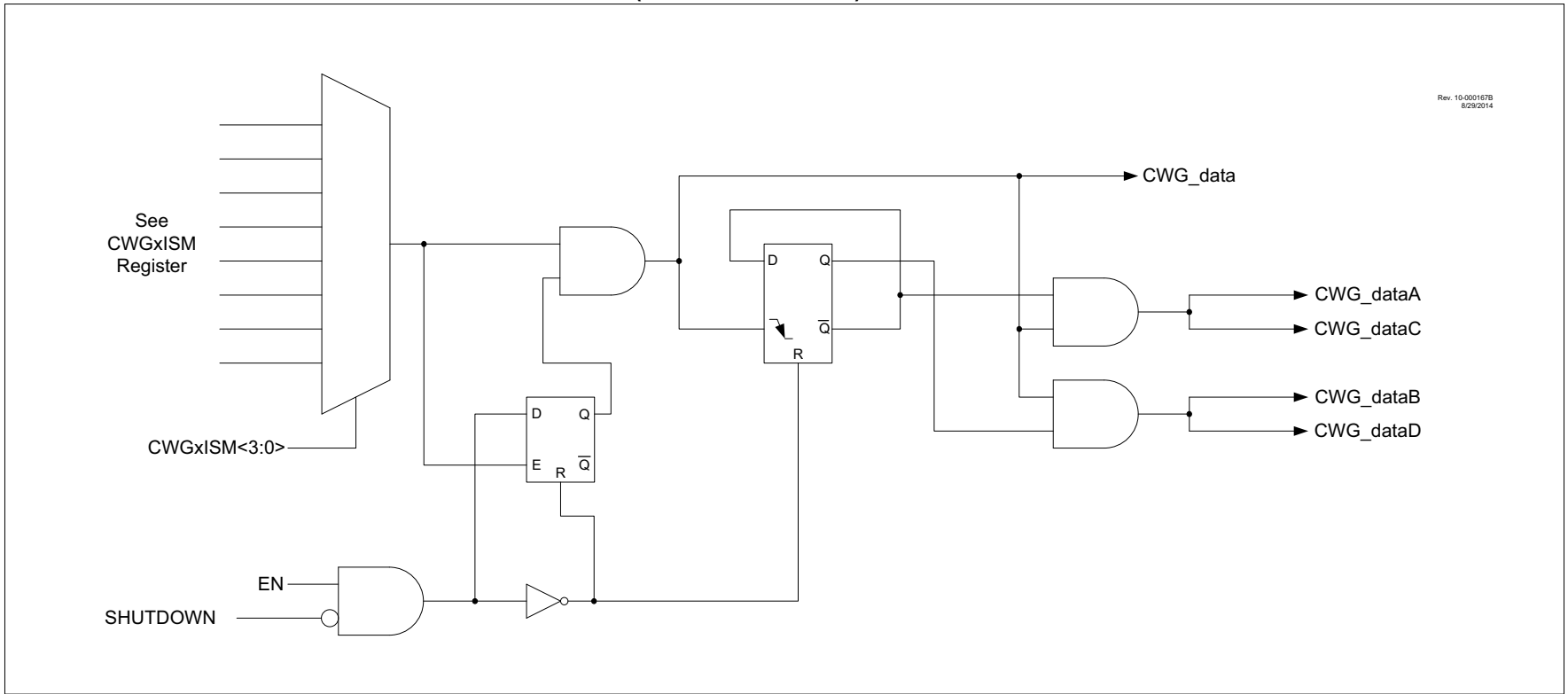
## 30.1.3 FULL-BRIDGE MODES

In Forward and Reverse Full-Bridge modes, three outputs drive static values while the fourth is modulated by the input data signal. In Forward Full-Bridge mode, CWG1A is driven to its active state, CWG1B and CWG1C are driven to their inactive state, and CWG1D is modulated by the input signal. In Reverse Full-Bridge mode, CWG1C is driven to its active state, CWG1A and CWG1D are driven to their inactive states, and CWG1B is modulated by the input signal. In Full-Bridge mode, the dead-band period is used when there is a switch from forward to reverse or vice-versa. This dead-band control is described in [Section 30.5 “Dead-Band Control”](#), with additional details in [Section 30.6 “Rising Edge and Reverse Dead Band”](#) and [Section 30.7 “Falling Edge and Forward Dead Band”](#).

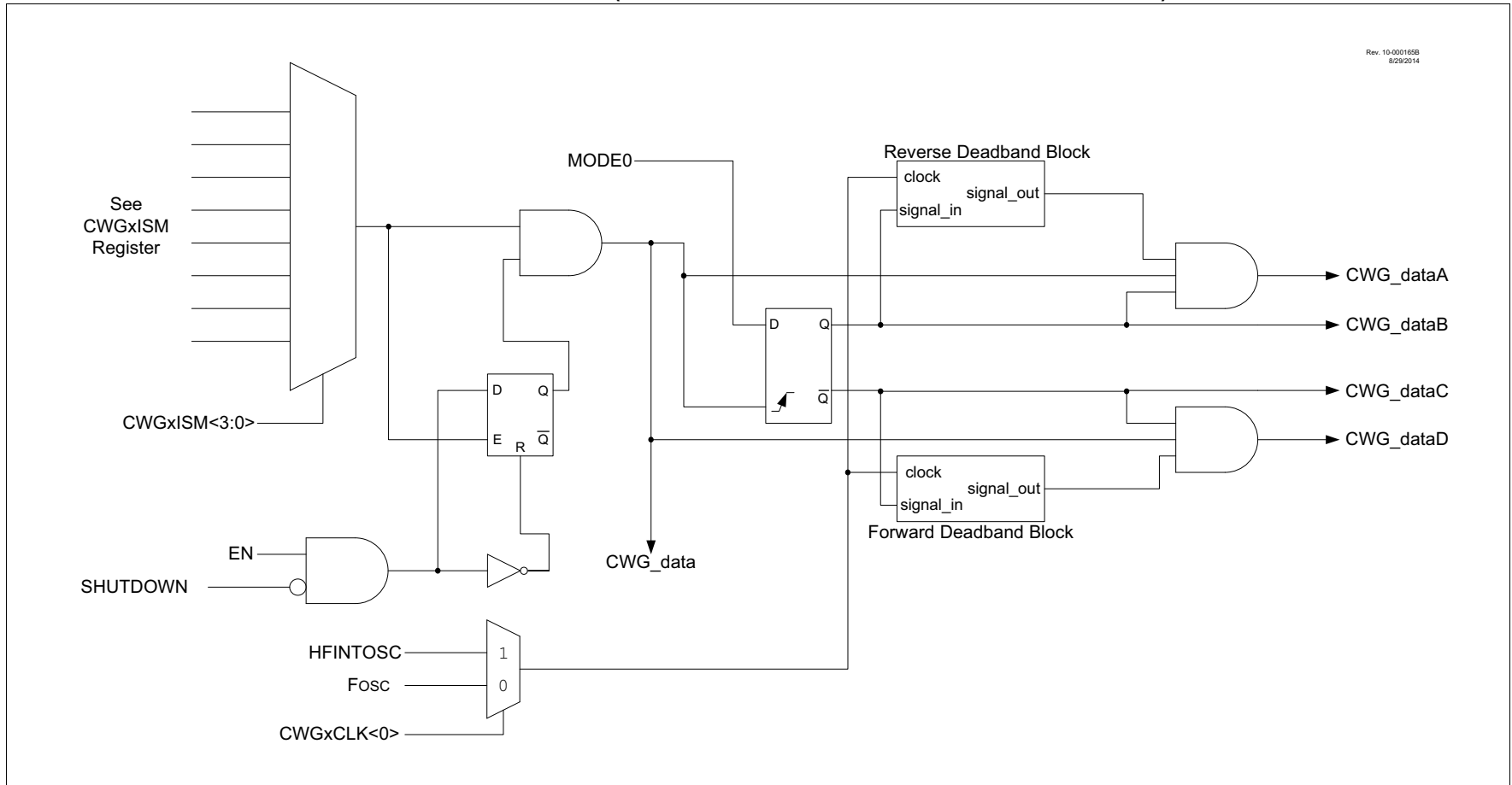
The mode selection may be toggled between forward and reverse toggling the MODE<0> bit of the CWG1CON0 while keeping MODE<2:1> static, without disabling the CWG module.



**FIGURE 30-2: SIMPLIFIED CWG BLOCK DIAGRAM (PUSH-PULL MODE)**



**FIGURE 30-3: SIMPLIFIED CWG BLOCK DIAGRAM (FORWARD AND REVERSE FULL-BRIDGE MODES)**



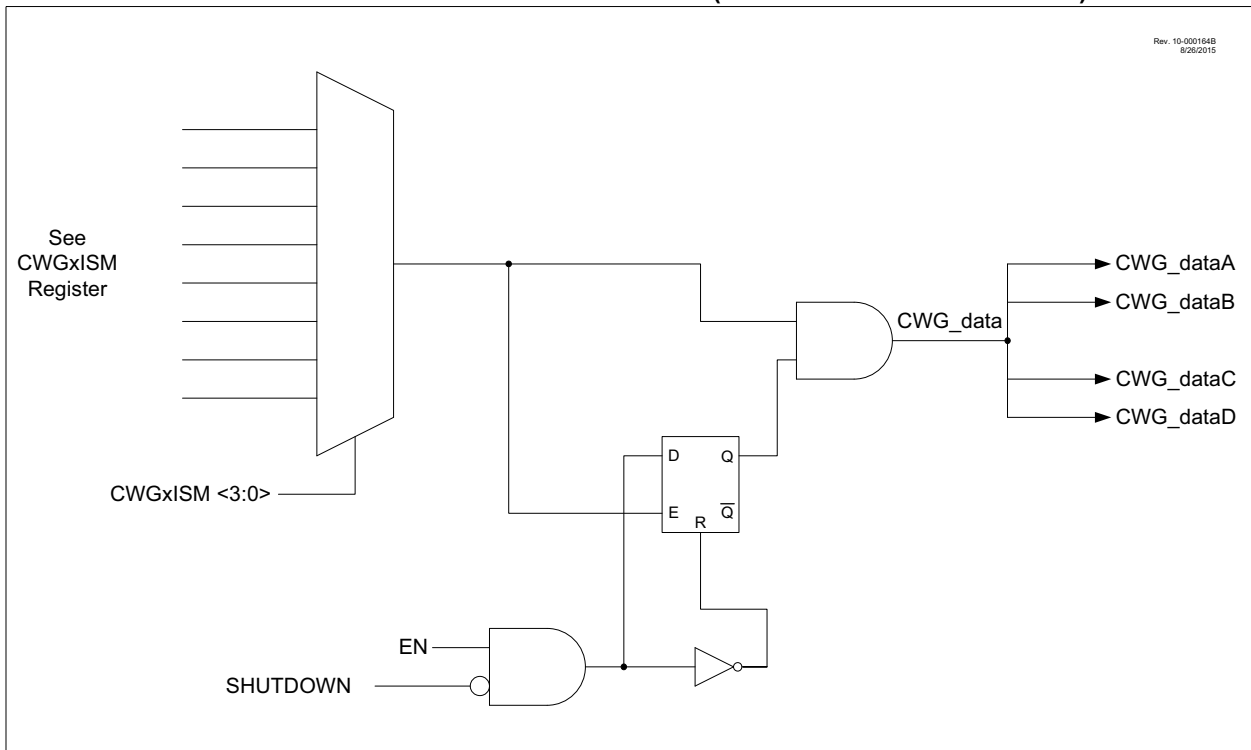
Rev. 10-000165B  
8/29/2014

## 30.1.4 STEERING MODES

In Steering modes, the data input can be steered to any or all of the four CWG output pins. In Synchronous Steering mode, changes to steering selection registers take effect on the next rising input.

In Non-Synchronous mode, steering takes effect on the next instruction cycle. Additional details are provided in [Section 30.9 “CWG Steering Mode”](#).

**FIGURE 30-4: SIMPLIFIED CWG BLOCK DIAGRAM (OUTPUT STEERING MODES)**



## 30.2 Clock Source

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

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

The clock sources are selected using the CS bit of the CWG1CLKCON register.

## 30.3 Selectable Input Sources

The CWG generates the output waveforms from the input sources in [Table 30-2](#).

**TABLE 30-2: SELECTABLE INPUT SOURCES**

Source Peripheral	Signal Name
CWG input PPS pin	CWG1PPS
CCP1	CCP1_out
CCP2	CCP2_out
PWM3	PWM3_out
PWM4	PWM4_out
PWM5	PWM5_out
PWM6	PWM6_out
NCO	NCO1_out
Comparator C1	C1OUT_sync
Comparator C2	C2OUT_sync
CLC1	LC1_out
CLC2	LC2_out
CLC3	LC3_out
CLC4	LC4_out

The input sources are selected using the CWG1DAT register.

## 30.4 Output Control

### 30.4.1 POLARITY CONTROL

The polarity of each CWG output can be selected independently. When the output polarity bit is set, the corresponding output is active-high. Clearing the output polarity bit configures the corresponding output as active-low. However, polarity does not affect the override levels. Output polarity is selected with the POLx bits of the CWG1CON1. Auto-shutdown and steering options are unaffected by polarity.

**FIGURE 30-5: CWG OUTPUT BLOCK DIAGRAM**



## 30.5 Dead-Band Control

The dead-band control provides non-overlapping PWM signals to prevent shoot-through current in PWM switches. Dead-band operation is employed for Half-Bridge and Full-Bridge modes. The CWG contains two 6-bit dead-band counters. One is used for the rising edge of the input source control in Half-Bridge mode or for reverse dead-band Full-Bridge mode. The other is used for the falling edge of the input source control in Half-Bridge mode or for forward dead band in Full-Bridge mode.

Dead band is timed by counting CWG clock periods from zero up to the value in the rising or falling dead-band counter registers. See CWG1DBR and CWG1DBF registers, respectively.

### 30.5.1 DEAD-BAND FUNCTIONALITY IN HALF-BRIDGE MODE

In Half-Bridge mode, the dead-band counters dictate the delay between the falling edge of the normal output and the rising edge of the inverted output. This can be seen in [Figure 30-9](#).

### 30.5.2 DEAD-BAND FUNCTIONALITY IN FULL-BRIDGE MODE

In Full-Bridge mode, the dead-band counters are used when undergoing a direction change. The MODE<0> bit of the CWG1CON0 register can be set or cleared while the CWG is running, allowing for changes from Forward to Reverse mode. The CWG1A and CWG1C signals will change upon the first rising input edge following a direction change, but the modulated signals (CWG1B or CWG1D, depending on the direction of the change) will experience a delay dictated by the dead-band counters. This is demonstrated in [Figure 30-3](#).

## 30.6 Rising Edge and Reverse Dead Band

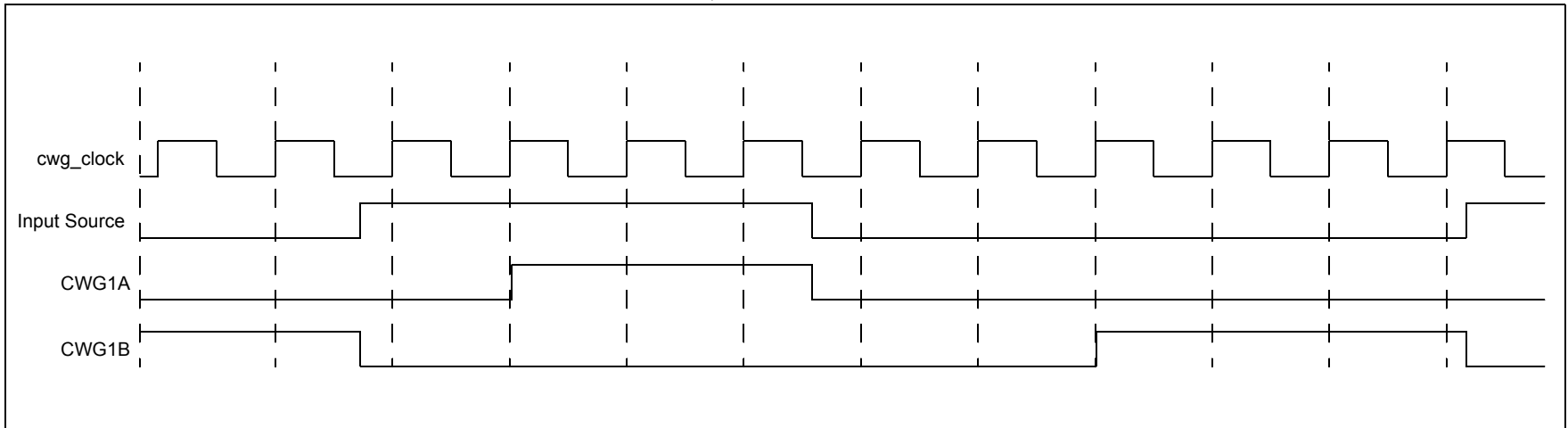
CWG1DBR controls the rising edge dead-band time at the leading edge of CWG1A (Half-Bridge mode) or the leading edge of CWG1B (Full-Bridge mode). The CWG1DBR value is double-buffered. When EN = 0, the CWG1DBR register is loaded immediately when CWG1DBR is written. When EN = 1, then software must set the LD bit of the CWG1CON0 register, and the buffer will be loaded at the next falling edge of the CWG input signal. If the input source signal is not present for enough time for the count to be completed, no output will be seen on the respective output.

## 30.7 Falling Edge and Forward Dead Band

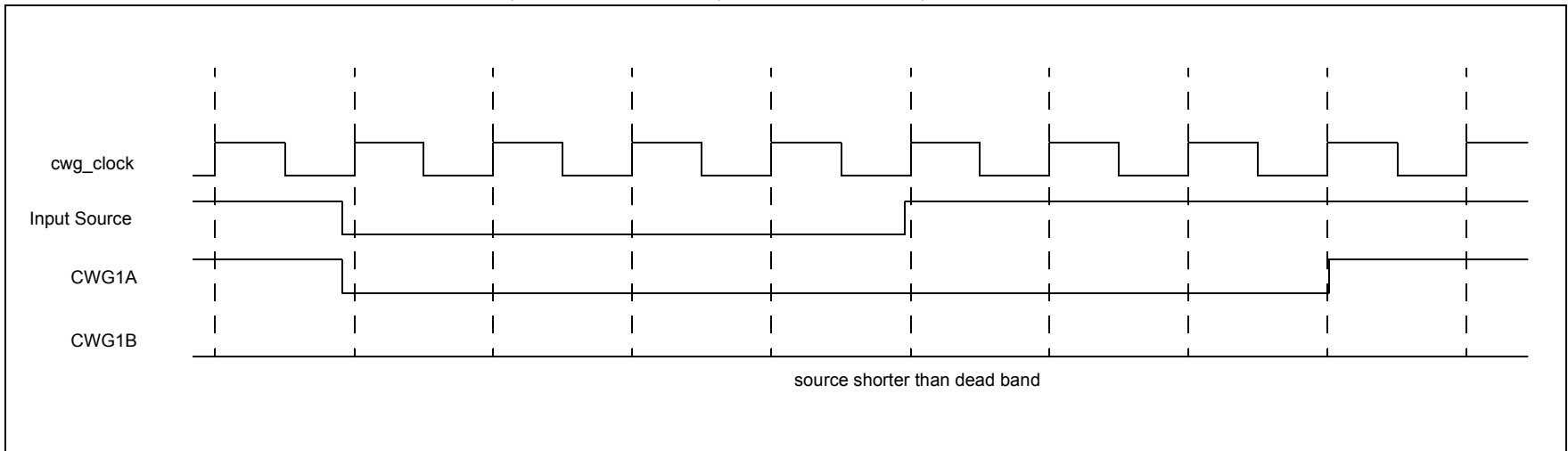
CWG1DBF controls the dead-band time at the leading edge of CWG1B (Half-Bridge mode) or the leading edge of CWG1D (Full-Bridge mode). The CWG1DBF value is double-buffered. When EN = 0, the CWG1DBF register is loaded immediately when CWG1DBF is written. When EN = 1 then software must set the LD bit of the CWG1CON0 register, and the buffer will be loaded at the next falling edge of the CWG input signal. If the input source signal is not present for enough time for the count to be completed, no output will be seen on the respective output.

Refer to [Figure 30-6](#) and [Figure 30-7](#) for examples.

**FIGURE 30-6: DEAD-BAND OPERATION CWG1DBR = 01H, CWG1DBF = 02H**



**FIGURE 30-7: DEAD-BAND OPERATION, CWG1DBR = 03H, CWG1DBF = 04H, SOURCE SHORTER THAN DEAD BAND**



## 30.8 Dead-Band Uncertainty

When the rising and falling edges of the input source are asynchronous to the CWG clock, it creates uncertainty in the dead-band time delay. The maximum uncertainty is equal to one CWG clock period. Refer to [Equation 30-1](#) for more details.

### EQUATION 30-1: DEAD-BAND UNCERTAINTY

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

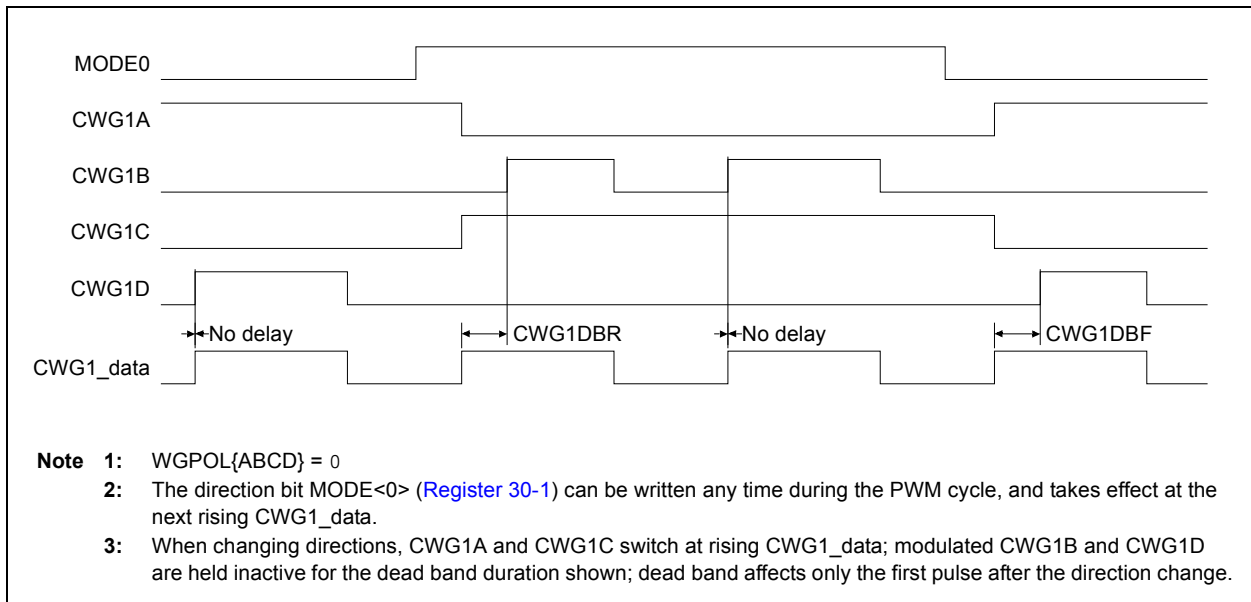
Example:

$$F_{CWG\_CLOCK} = 16 \text{ MHz}$$

Therefore:

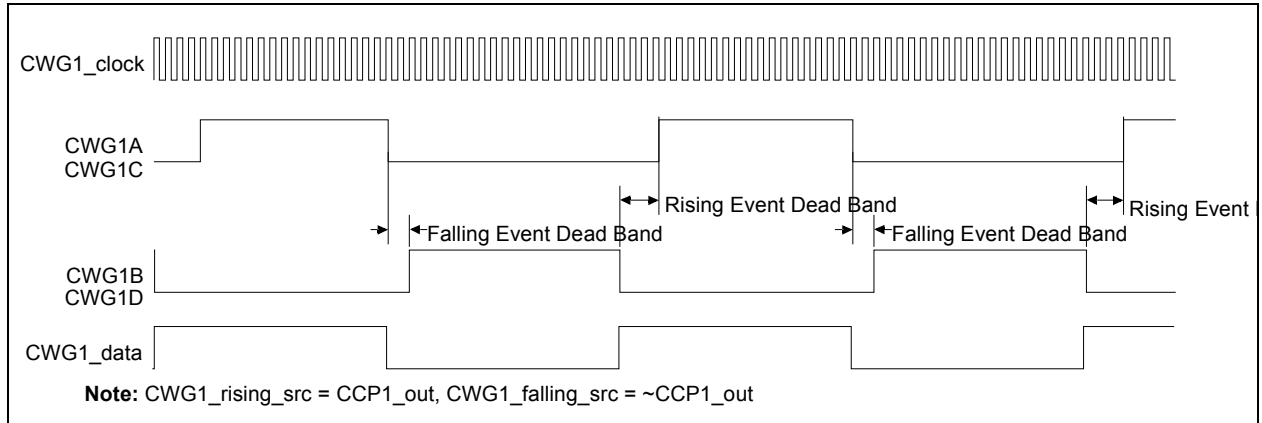
$$\begin{aligned} T_{DEADBAND\_UNCERTAINTY} &= \frac{1}{F_{cwg\_clock}} \\ &= \frac{1}{16 \text{ MHz}} \\ &= 62.5 \text{ ns} \end{aligned}$$

**FIGURE 30-8: EXAMPLE OF PWM DIRECTION CHANGE**





**FIGURE 30-9: CWG HALF-BRIDGE MODE OPERATION**



## 30.9 CWG Steering Mode

In Steering mode (MODE = 00x), the CWG allows any combination of the CWG1x pins to be the modulated signal. The same signal can be simultaneously available on multiple pins, or a fixed-value output can be presented.

When the respective STRx bit of CWG1OCON0 is '0', the corresponding pin is held at the level defined. When the respective STRx bit of CWG1OCON0 is '1', the pin is driven by the input data signal. The user can assign the input data signal to one, two, three, or all four output pins.

The POLx bits of the CWG1CON1 register control the signal polarity only when STRx = 1.

The CWG auto-shutdown operation also applies in Steering modes as described in [Section 30.10 "Auto-Shutdown"](#). An auto-shutdown event will only affect pins that have STRx = 1.

### 30.9.1 STEERING SYNCHRONIZATION

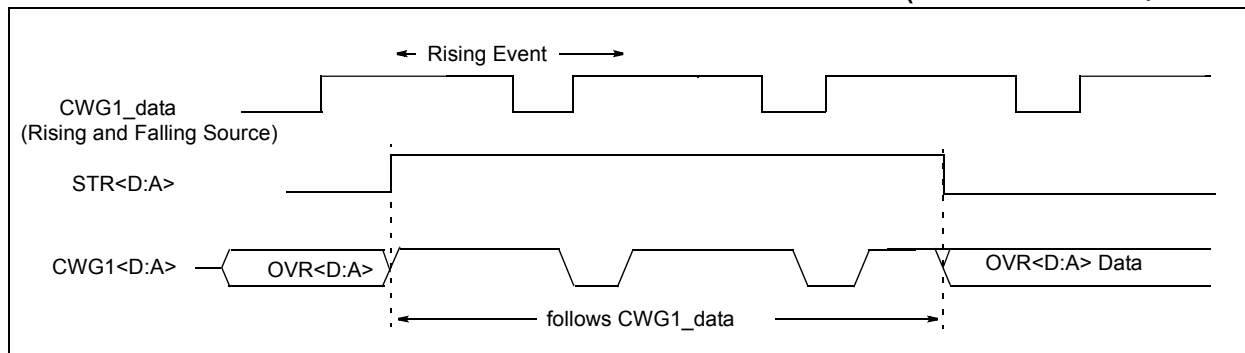
Changing the MODE bits allows for two modes of steering, synchronous and asynchronous.

When MODE = 000, the steering event is asynchronous and will happen at the end of the instruction that writes to STRx (that is, immediately). In this case, the output signal at the output pin may be an incomplete waveform. This can be useful for immediately removing a signal from the pin.

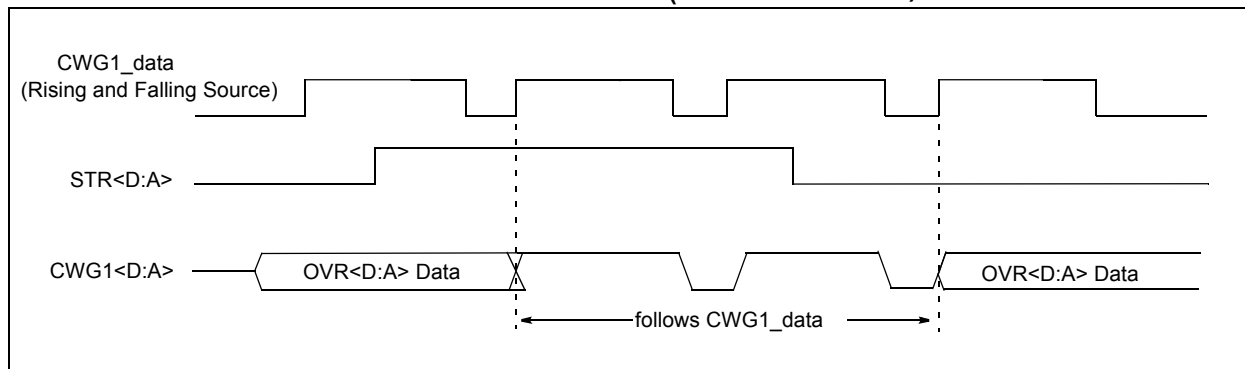
When MODE = 001, the steering update is synchronous and occurs at the beginning of the next rising edge of the input data signal. In this case, steering the output on/off will always produce a complete waveform.

[Figure 30-10](#) and [Figure 30-11](#) illustrate the timing of asynchronous and synchronous steering, respectively.

**FIGURE 30-10: EXAMPLE OF ASYNCHRONOUS STEERING EVENT (MODE<2:0> = 000)**



**FIGURE 30-11: EXAMPLE OF STEERING EVENT (MODE<2:0> = 001)**



## 30.10 Auto-Shutdown

Auto-shutdown is a method to immediately override the CWG output levels with specific overrides that allow for safe shutdown of the circuit. The shutdown state can be either cleared automatically or held until cleared by software. The auto-shutdown circuit is illustrated in [Figure 30-12](#).

### 30.10.1 SHUTDOWN

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

- Software generated
- External Input

#### 30.10.1.1 Software Generated Shutdown

Setting the SHUTDOWN bit of the CWG1AS0 register will force the CWG into the shutdown state.

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

When auto-restart is enabled, the SHUTDOWN bit will clear automatically and resume operation on the next rising edge event.

### 30.10.2 EXTERNAL INPUT SOURCE

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

- Comparator C1OUT\_sync
- Comparator C2OUT\_sync
- Timer2 – TMR2\_postscaled
- CWG1IN input pin

Shutdown inputs are selected using the CWG1AS1 register ([Register 30-6](#)).

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

## 30.11 Operation During Sleep

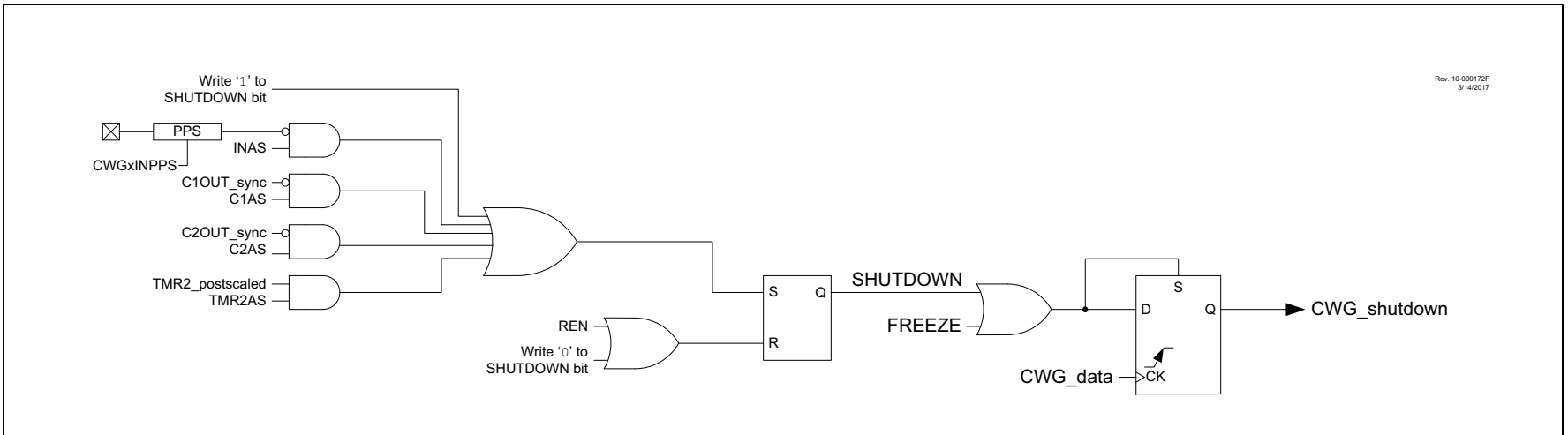
The CWG module operates independently from the system clock and will continue to run during Sleep, provided that the clock and input sources selected remain active.

The HFINTOSC remains active during Sleep when all the following conditions are met:

- CWG module is enabled
- Input source is active
- HFINTOSC is selected as the clock source, regardless of the system clock source selected.

In other words, if the HFINTOSC is simultaneously selected as the system clock and the CWG clock source, when the CWG is enabled and the input source is active, then the CPU will go idle during Sleep, but the HFINTOSC will remain active and the CWG will continue to operate. This will have a direct effect on the Sleep mode current.

**FIGURE 30-12: CWG SHUTDOWN BLOCK DIAGRAM**



## 30.12 Configuring the CWG

The following steps illustrate how to properly configure the CWG.

1. Ensure that the TRIS control bits corresponding to the desired CWG pins for your application are set so that the pins are configured as inputs.
2. Clear the EN bit, if not already cleared.
3. Set desired mode of operation with the MODE bits.
4. Set desired dead-band times, if applicable to mode, with the CWG1DBR and CWG1DBF registers.
5. Setup the following controls in the CWG1AS0 and CWG1AS1 registers.
  - a. Select the desired shutdown source.
  - b. Select both output overrides to the desired levels (this is necessary even if not using auto-shutdown because start-up will be from a shutdown state).
  - c. Set which pins will be affected by auto-shutdown with the CWG1AS1 register.
  - d. Set the SHUTDOWN bit and clear the REN bit.
6. Select the desired input source using the CWG1DAT register.
7. Configure the following controls.
  - a. Select desired clock source using the CWG1CLKCON register.
  - b. Select the desired output polarities using the CWG1CON1 register.
  - c. Set the output enables for the desired outputs.
8. Set the EN bit.
9. Clear TRIS control bits corresponding to the desired output pins to configure these pins as outputs.
10. If auto-restart is to be used, set the REN bit and the SHUTDOWN bit will be cleared automatically. Otherwise, clear the SHUTDOWN bit to start the CWG.

### 30.12.1 PIN OVERRIDE LEVELS

The levels driven to the output pins, while the shutdown input is true, are controlled by the LSB and LSAC bits of the CWG1AS0 register. LSB<1:0> controls the CWG1B and D override levels and LSAC<1:0> controls the CWG1A and C override levels. The control bit logic level corresponds to the output logic drive level while in the shutdown state. The polarity control does not affect the override level.

### 30.12.2 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 REN bit of the CWG1CON2 register. Waveforms of software controlled and automatic restarts are shown in [Figure 30-13](#) and [Figure 30-14](#).

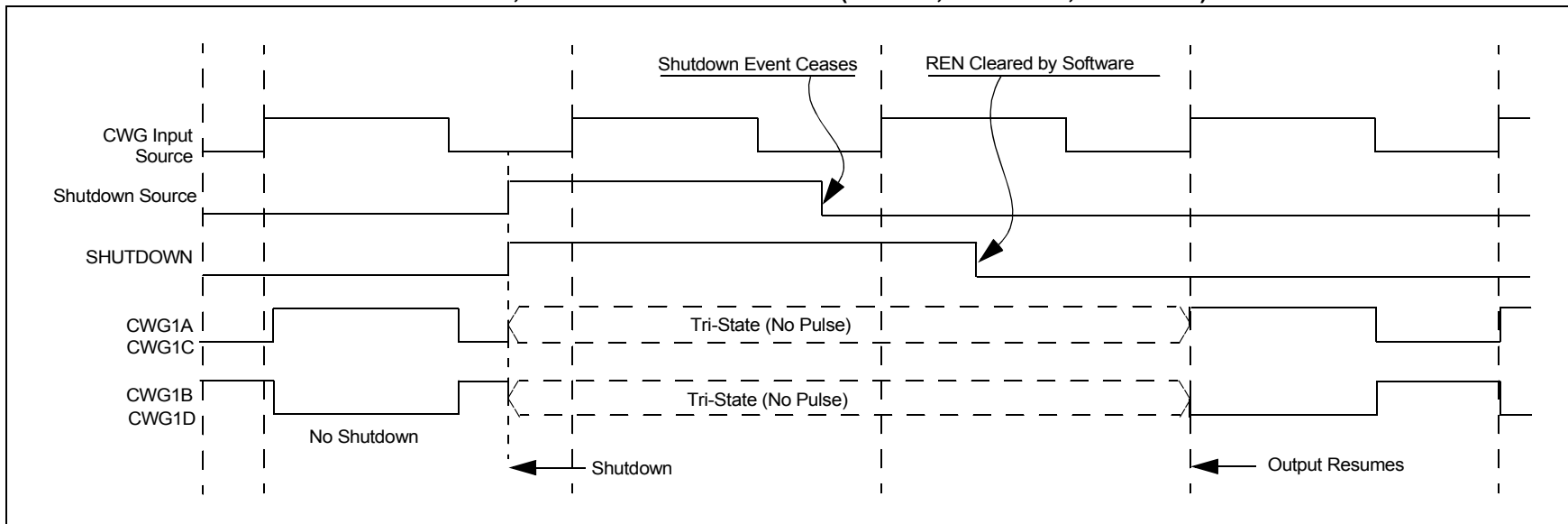
#### 30.12.2.1 Software Controlled Restart

When the REN bit of the CWG1AS0 register is cleared, the CWG must be restarted after an auto-shutdown event by software. Clearing the shutdown state requires all selected shutdown inputs to be low, otherwise the SHUTDOWN bit will remain set. The overrides will remain in effect until the first rising edge event after the SHUTDOWN bit is cleared. The CWG will then resume operation.

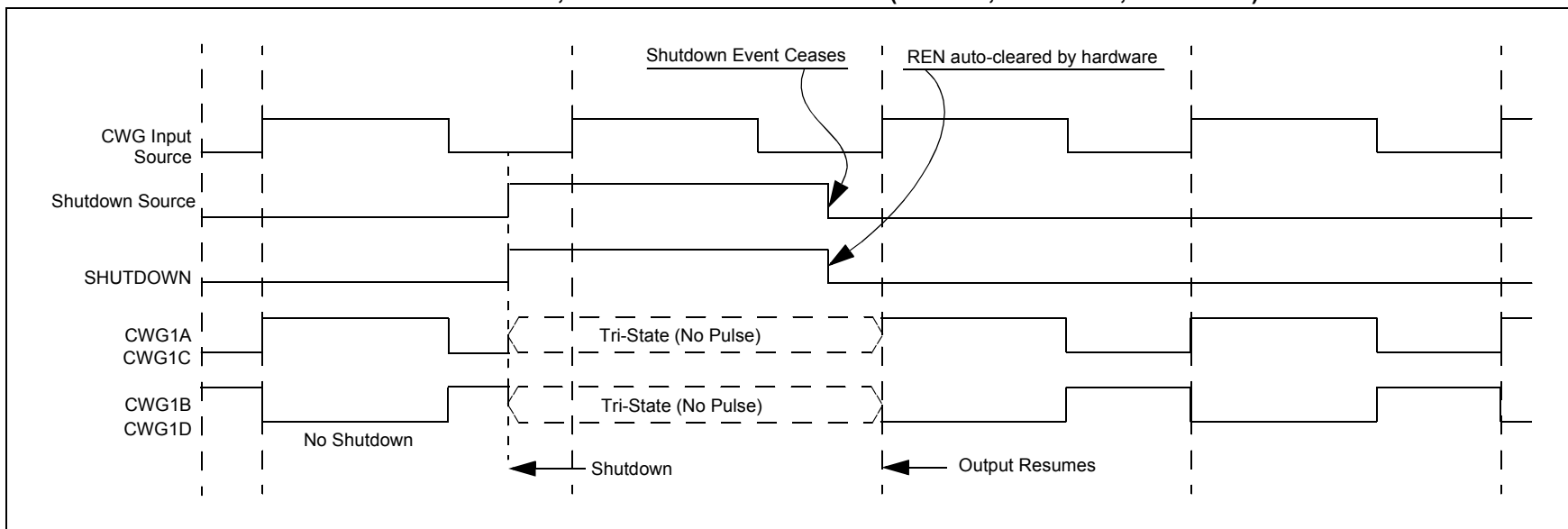
#### 30.12.2.2 Auto-Restart

When the REN bit of the CWG1CON2 register is set, the CWG will restart from the auto-shutdown state automatically. The SHUTDOWN bit will clear automatically when all shutdown sources go low. The overrides will remain in effect until the first rising edge event after the SHUTDOWN bit is cleared. The CWG will then resume operation.

**FIGURE 30-13: SHUTDOWN FUNCTIONALITY, AUTO-RESTART DISABLED (REN = 0, LSAC = 01, LSBD = 01)**



**FIGURE 30-14: SHUTDOWN FUNCTIONALITY, AUTO-RESTART ENABLED (REN = 1, LSAC = 01, LSBD = 01)**



## 30.13 Register Definitions: CWG Control

Long bit name prefixes for the CWG peripherals are shown in [Section 1.1 “Register and Bit Naming Conventions”](#).

### REGISTER 30-1: CWG1CON0: CWG1 CONTROL REGISTER 0

R/W-0/0	R/W/HC-0/0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
EN	LD <sup>(1)</sup>	—	—	—	MODE<2:0>		
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

q = Value depends on condition

bit 7	<b>EN:</b> CWG1 Enable bit 1 = Module is enabled 0 = Module is disabled
bit 6	<b>LD:</b> CWG1 Load Buffer bits <sup>(1)</sup> 1 = Buffers to be loaded on the next rising/falling event 0 = Buffers not loaded
bit 5-3	<b>Unimplemented:</b> Read as '0'
bit 2-0	<b>MODE&lt;2:0&gt;:</b> CWG1 Mode bits 111 = Reserved 110 = Reserved 101 = CWG outputs operate in Push-Pull mode 100 = CWG outputs operate in Half-Bridge mode 011 = CWG outputs operate in Reverse Full-Bridge mode 010 = CWG outputs operate in Forward Full-Bridge mode 001 = CWG outputs operate in Synchronous Steering mode 000 = CWG outputs operate in Steering mode

**Note 1:** This bit can only be set after EN = 1 and cannot be set in the same instruction that EN is set.

## REGISTER 30-2: CWG1CON1: CWG1 CONTROL REGISTER 1

U-0	U-0	R-x	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	IN	—	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 7-6	<b>Unimplemented:</b> Read as '0'
bit 5	<b>IN:</b> CWG Input Value bit
bit 4	<b>Unimplemented:</b> Read as '0'
bit 3	<b>POLD:</b> CWG1D Output Polarity bit 1 = Signal output is inverted polarity 0 = Signal output is normal polarity
bit 2	<b>POLC:</b> CWG1C Output Polarity bit 1 = Signal output is inverted polarity 0 = Signal output is normal polarity
bit 1	<b>POLB:</b> CWG1B Output Polarity bit 1 = Signal output is inverted polarity 0 = Signal output is normal polarity
bit 0	<b>POLA:</b> CWG1A Output Polarity bit 1 = Signal output is inverted polarity 0 = Signal output is normal polarity



## REGISTER 30-3: CWG1DBR: CWG1 RISING DEAD-BAND COUNTER 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
—	—	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 Value for Counter bits

## REGISTER 30-4: CWG1DBF: CWG1 FALLING DEAD-BAND COUNTER 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
—	—	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 Value for Counter bits

## REGISTER 30-5: CWG1AS0: CWG1 AUTO-SHUTDOWN CONTROL REGISTER 0

R/W/HS-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
SHUTDOWN <sup>(1,2)</sup>	REN	LSBD<1:0>		LSAC<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

q = Value depends on condition

- bit 7            **SHUTDOWN:** Auto-Shutdown Event Status bit<sup>(1, 2)</sup>  
 1 = An Auto-Shutdown state is in effect  
 0 = No Auto-shutdown event has occurred
- bit 6            **REN:** Auto-Restart Enable bit  
 1 = Auto-restart enabled  
 0 = Auto-restart disabled
- bit 5-4         **LSBD<1:0>:** CWG1B and CWG1D Auto-Shutdown State Control bits  
 11 =A logic '1' is placed on CWG1B/D when an auto-shutdown event is present  
 10 =A logic '0' is placed on CWG1B/D when an auto-shutdown event is present  
 01 =Pin is tri-stated on CWG1B/D when an auto-shutdown event is present  
 00 =The inactive state of the pin, including polarity, is placed on CWG1B/D after the required dead-band interval
- bit 3-2         **LSAC<1:0>:** CWG1A and CWG1C Auto-Shutdown State Control bits  
 11 =A logic '1' is placed on CWG1A/C when an auto-shutdown event is present  
 10 =A logic '0' is placed on CWG1A/C when an auto-shutdown event is present  
 01 =Pin is tri-stated on CWG1A/C when an auto-shutdown event is present  
 00 =The inactive state of the pin, including polarity, is placed on CWG1A/C after the required dead-band interval
- bit 1-0         **Unimplemented:** Read as '0'

**Note 1:** This bit may be written while EN = 0 (CWG1CON0 register) to place the outputs into the shutdown configuration.

**2:** The outputs will remain in auto-shutdown state until the next rising edge of the input signal after this bit is cleared.

## REGISTER 30-6: CWG1AS1: CWG1 AUTO-SHUTDOWN CONTROL REGISTER 1

U-1	U-1	U-1	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	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 7-5	<b>Unimplemented:</b> Read as '0'
bit 4	<b>AS4E:</b> CLC2 Output bit 1 = LC2_out shut-down is enabled 0 = LC2_out shut-down is disabled
bit 3	<b>AS3E:</b> Comparator C2 Output bit 1 = C2 output shut-down is enabled 0 = C2 output shut-down is disabled
bit 2	<b>AS2E:</b> Comparator C1 Output bit 1 = C1 output shut-down is enabled 0 = C1 output shut-down is disabled
bit 2	<b>AS1E:</b> TMR2 Postscale Output bit 1 = TMR2 Postscale shut-down is enabled 0 = TMR2 Postscale shut-down is disabled
bit 0	<b>AS0E:</b> CWG1 Input Pin bit 1 = Input pin selected by CWG1PPS shut-down is enabled 0 = Input pin selected by CWG1PPS shut-down is disabled

## REGISTER 30-7: CWG1STR: CWG1 STEERING CONTROL REGISTER<sup>(1)</sup>

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
OVRD	OVRC	OVRB	OVRA	STRD <sup>(2)</sup>	STRC <sup>(2)</sup>	STRB <sup>(2)</sup>	STRA <sup>(2)</sup>
bit 7							bit 0

### Legend:

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

bit 7	<b>OVRD:</b> Steering Data D bit
bit 6	<b>OVRC:</b> Steering Data C bit
bit 5	<b>OVRB:</b> Steering Data B bit
bit 4	<b>OVRA:</b> Steering Data A bit
bit 3	<b>STRD:</b> Steering Enable D bit <sup>(2)</sup> 1 = CWG1D output has the CWG1_data waveform with polarity control from POLD bit 0 = CWG1D output is assigned the value of OVRD bit
bit 2	<b>STRC:</b> Steering Enable C bit <sup>(2)</sup> 1 = CWG1C output has the CWG1_data waveform with polarity control from POLC bit 0 = CWG1C output is assigned the value of OVRC bit
bit 1	<b>STRB:</b> Steering Enable B bit <sup>(2)</sup> 1 = CWG1B output has the CWG1_data waveform with polarity control from POLB bit 0 = CWG1B output is assigned the value of OVRB bit
bit 0	<b>STRA:</b> Steering Enable A bit <sup>(2)</sup> 1 = CWG1A output has the CWG1_data waveform with polarity control from POLA bit 0 = CWG1A output is assigned the value of OVRA bit

**Note 1:** The bits in this register apply only when MODE<2:0> = 00x.

**2:** This bit is effectively double-buffered when MODE<2:0> = 001.

# PIC16(L)F15354/55

## REGISTER 30-8: CWG1CLK: CWG1 CLOCK SELECTION REGISTER

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

### Legend:

R = Readable bit                      W = Writable bit                      U = Unimplemented bit, read as '0'  
u = Bit is unchanged                      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 7-1                      **Unimplemented:** Read as '0'  
bit 0                      **CS:** CWG1 Clock Selection bit  
                                    1 = HFINTOSC 16 MHz is selected  
                                    0 = FOSC is selected

## REGISTER 30-9: CWG1DAT: CWG1 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
—	—	—	—	DAT<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                      **DAT<3:0>:** CWG1 Input Selection bits  
                                    1111 =Reserved. No channel connected.  
                                    1110 =Reserved. No channel connected.  
                                    1101 =LC4\_out  
                                    1100 =LC3\_out  
                                    1011 =LC2\_out  
                                    1010 =LC1\_out  
                                    1001 =Comparator C2 out  
                                    1000 =Comparator C1 out  
                                    0111 =NCO1 output  
                                    0110 =PWM6\_out  
                                    0101 =PWM5\_out  
                                    0100 =PWM4\_out  
                                    0011 =PWM3\_out  
                                    0010 =CCP2\_out  
                                    0001 =CCP1\_out  
                                    0000 =CWG11CLK

**TABLE 30-3: SUMMARY OF REGISTERS ASSOCIATED WITH CWG**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page	
CWG1CLKCON	—	—	—	—	—	—	—	CS	353	
CWG1DAT	—	—	—	—	DAT<3:0>				353	
CWG1DBR	—	—	DBR<5:0>						349	
CWG1DBF	—	—	DBF<5:0>						349	
CWG1CON0	EN	LD	—	—	—	MODE<2:0>			352	
CWG1CON1	—	—	IN	—	POLD	POLC	POLB	POLA	348	
CWG1AS0	SHUTDOWN	REN	LSBD<1:0>		LSAC<1:0>			—	—	350
CWG1AS1	—	—	—	AS4E	AS3E	AS2E	AS1E	AS0E	351	
CWG1STR	OVRD	OVRC	OVRB	OVRA	STRD	STRC	STRB	STRA	352	

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

## 31.0 CONFIGURABLE LOGIC CELL (CLC)

The Configurable Logic Cell (CLCx) module provides programmable logic that operates outside the speed limitations of software execution. The logic cell selects from 40 input signals and, through the use of configurable gates, reduces the 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.

The CLC modules available are shown in [Table 31-1](#).

Refer to [Figure 31-1](#) for a simplified diagram showing signal flow through the CLCx.

Possible configurations include:

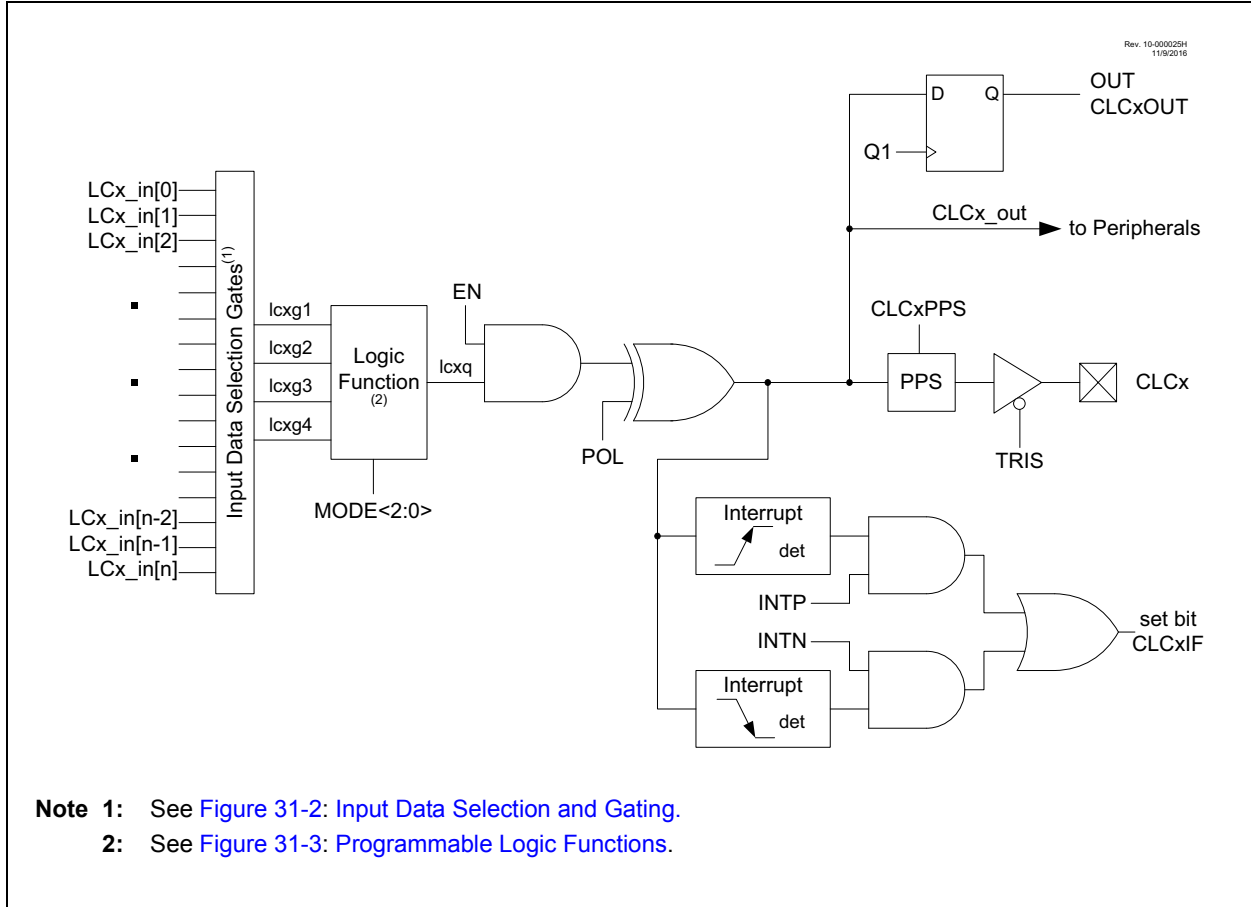
- Combinatorial Logic
  - AND
  - NAND
  - AND-OR
  - AND-OR-INVERT
  - OR-XOR
  - OR-XNOR
- Latches
  - S-R
  - Clocked D with Set and Reset
  - Transparent D with Set and Reset
  - Clocked J-K with Reset

**TABLE 31-1: AVAILABLE CLC MODULES**

Device	CLC1	CLC2	CLC3	CLC4
PIC16(L)F15354/55	•	•	•	•

**Note:** The CLC1, CLC2, CLC3 and CLC4 are four separate module instances of the same CLC module design. Throughout this section, the lower case 'x' in register and bit names is a generic reference to the CLC number (which should be substituted with 1, 2, 3, or 4 during code development). For example, the control register is generically described in this chapter as CLCxCON, but the actual device registers are CLC1CON, CLC2CON, CLC3CON and CLC4CON. Similarly, the LCxEN bit represents the LC1EN, LC2EN, LC3EN and LC4EN bits.

**FIGURE 31-1: CLCx SIMPLIFIED BLOCK DIAGRAM**



- Note 1:** See [Figure 31-2: Input Data Selection and Gating](#).  
**Note 2:** See [Figure 31-3: Programmable Logic Functions](#).



## 31.1 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.

### 31.1.1 DATA SELECTION

There are 40 signals available as inputs to the configurable logic. Four 40-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 31-2](#). Data inputs in the figure are identified by a generic numbered input name.

[Table 31-2](#) correlates the generic input name to the actual signal for each CLC module. The column labeled 'LCxDyS<5:0> Value' indicates the MUX selection code for the selected data input. LCxDyS is an abbreviation to identify specific multiplexers: LCxD1S<5:0> through LCxD4S<5:0>.

Data inputs are selected with CLCxSEL0 through CLCxSEL3 registers ([Register 31-3](#) through [Register 31-6](#)).

**TABLE 31-2: CLCx DATA INPUT SELECTION**

LCxDyS<5:0> Value	CLCx Input Source
101000 to 111111 [40+]	Reserved
100111 [39]	CWG1B output
100110 [38]	CWG1A output
100101 [37]	MSSP2 SCK output
100100 [36]	MSSP2 SDO output
100011 [35]	MSSP1 SCK output
100010 [34]	MSSP1 SDO output
100001 [33]	EUSART2 (TX/CK) output
100000 [32]	EUSART2 (DT) output
011111 [31]	EUSART1 (TX/CK) output
011110 [30]	EUSART1 (DT) output
011101 [29]	CLC4 output
011100 [28]	CLC3 output
011011 [27]	CLC2 output
011010 [26]	CLC1 output
011001 [25]	IOCIF
011000 [24]	ZCD output
010111 [23]	C2OUT
010110 [22]	C1OUT
010101 [21]	NCO1 output
010100 [20]	PWM6 output
010011 [19]	PWM5 output
010010 [18]	PWM4 output
010001 [17]	PWM3 output
010000 [16]	CCP2 output
001111 [15]	CCP1 output
001110 [14]	Timer2 overflow
001101 [13]	Timer1 overflow
001100 [12]	Timer0 overflow
001011 [11]	CLKR
001010 [10]	ADCRC
001001 [9]	SOSC
001000 [8]	MFINTOSC (32 kHz)
000111 [7]	MFINTOSC (500 kHz)
000110 [6]	LFINTOSC
000101 [5]	HFINTOSC
000100 [4]	Fosc
000011 [3]	CLCIN3PPS
000010 [2]	CLCIN2PPS
000001 [1]	CLCIN1PPS

## 31.1.2 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. The output of each gate can be inverted before going on to the logic function stage.

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

Table 31-3 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 31-3: DATA GATING LOGIC**

CLCxGLSy	LCxGyPOL	Gate Logic
0x55	1	4-input AND
0x55	0	4-input NAND
0xAA	1	4-input NOR
0xAA	0	4-input 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 31-7)
- Gate 2: CLCxGLS1 (Register 31-8)
- Gate 3: CLCxGLS2 (Register 31-9)
- Gate 4: CLCxGLS3 (Register 31-10)

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 31-2. 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.

## 31.1.3 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 31-2. 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.

## 31.1.4 OUTPUT POLARITY

The last stage in the Configurable Logic Cell is the output polarity. Setting the LCxPOL bit of the CLCxPOL 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.

## 31.2 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 PIR5 register will be set when either edge detector is triggered and its associated enable bit is set. The LCxINTP enables rising edge interrupts and the LCxINTN bit enables falling edge interrupts. Both are located in the CLCxCON register.

To fully enable the interrupt, set the following bits:

- CLCxIE bit of the PIE5 register
- LCxINTP bit of the CLCxCON register (for a rising edge detection)
- LCxINTN bit of the CLCxCON register (for a falling edge detection)
- PEIE and GIE bits of the INTCON register

The CLCxIF bit of the PIR5 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.

## 31.3 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 LCxOUT bits in the individual CLCxCON registers.

## 31.4 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.

## 31.5 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.

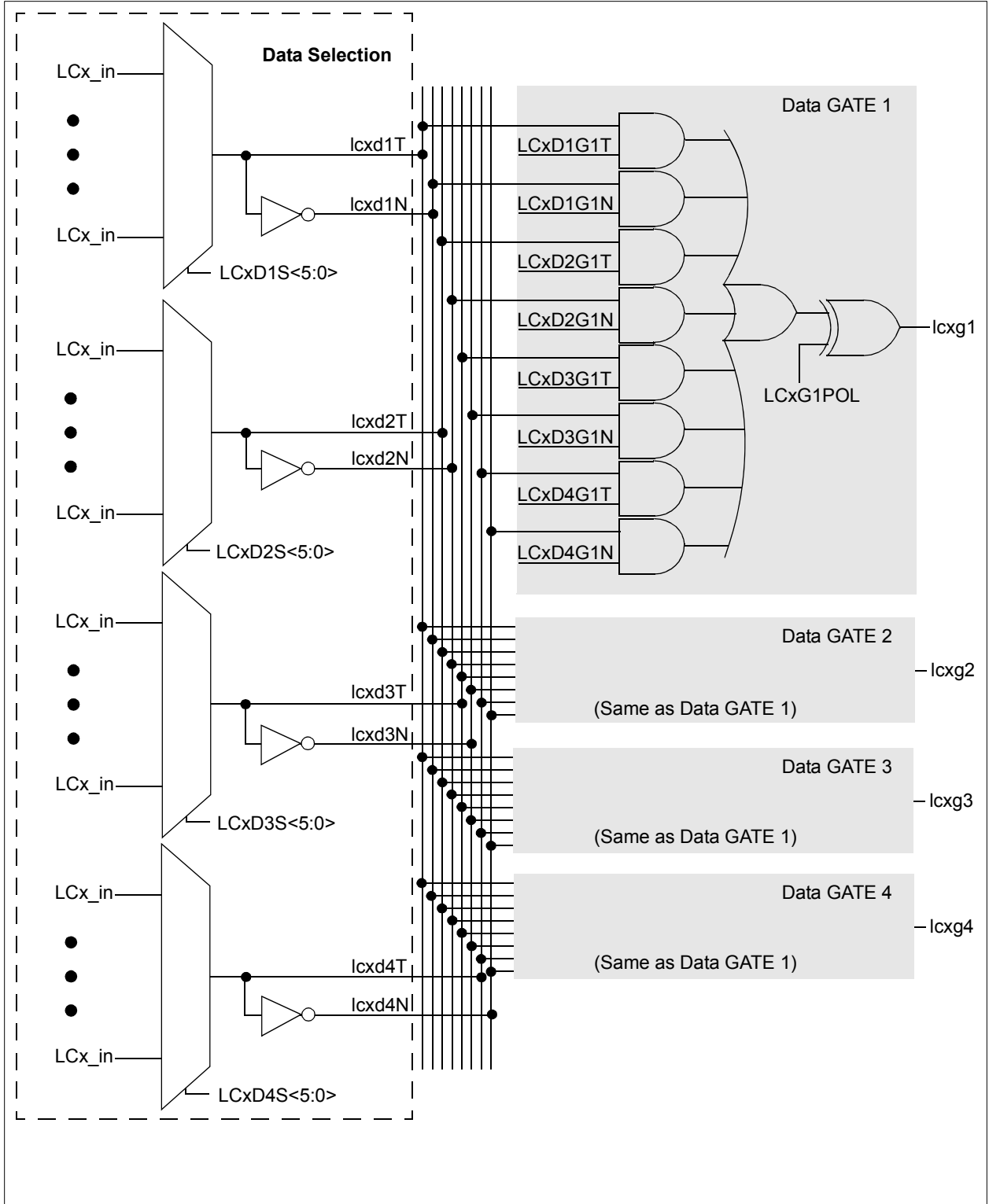
This will have a direct effect on the Sleep mode current.

## 31.6 CLCx Setup Steps

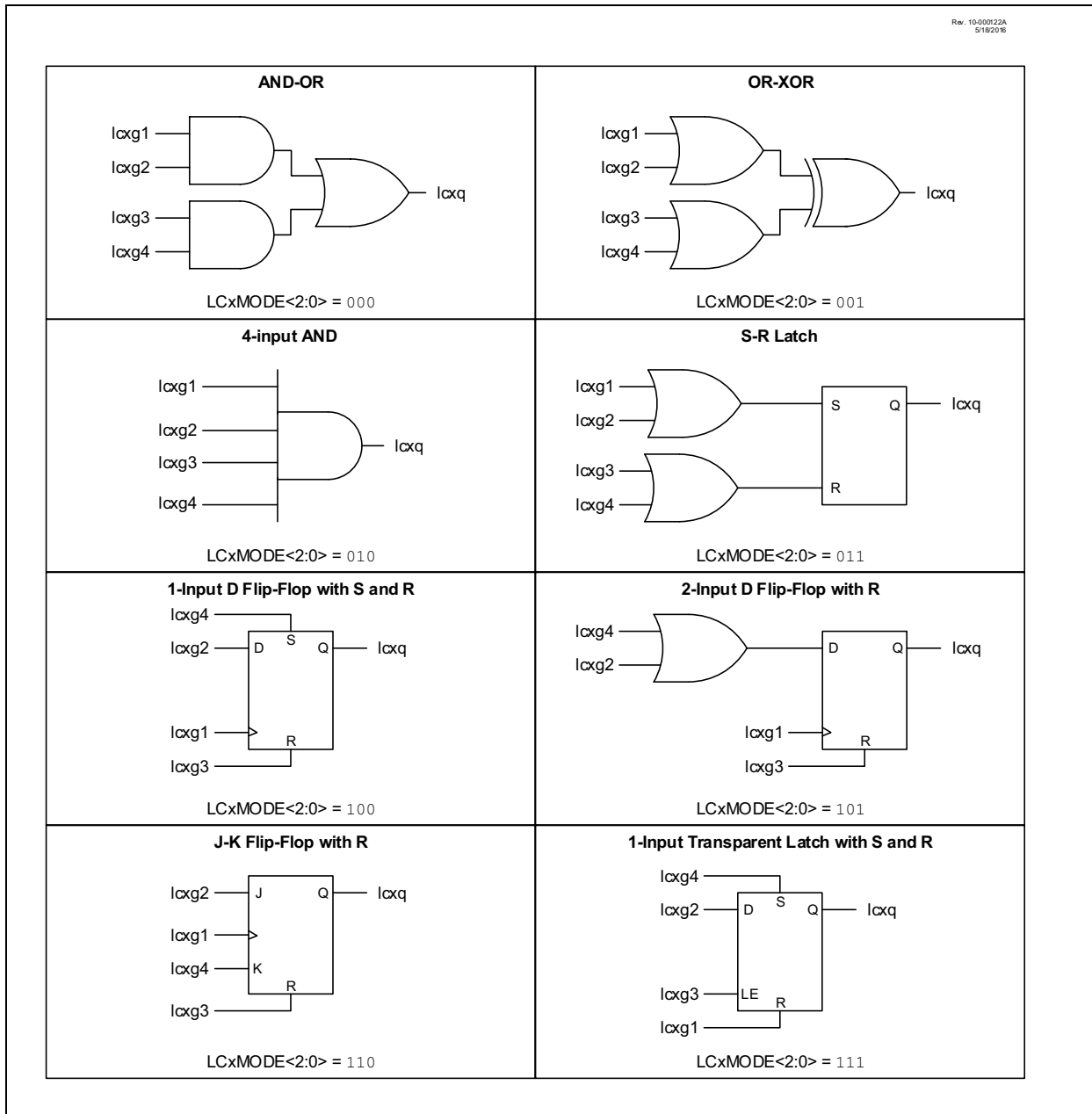
The following steps should be followed when setting up the CLCx:

- Disable CLCx by clearing the LCxEN bit.
- Select desired inputs using CLCxSEL0 through CLCxSEL3 registers (See [Table 31-2](#)).
- Clear any associated ANSEL bits.
- Enable the chosen inputs through the four gates using CLCxGLS0, CLCxGLS1, CLCxGLS2, and CLCxGLS3 registers.
- Select the gate output polarities with the LCxGyPOL bits of the CLCxPOL register.
- Select the desired logic function with the LCxMODE<2:0> bits of the CLCxCON register.
- Select the desired polarity of the logic output with the LCxPOL bit of the CLCxPOL register. (This step may be combined with the previous gate output polarity step).
- If driving a device 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 LCxINTP bit in the CLCxCON register for rising event.
  - Set the LCxINTN bit in the CLCxCON register for falling event.
  - Set the CLCxIE bit of the PIE5 register.
  - Set the GIE and PEIE bits of the INTCON register.
- Enable the CLCx by setting the LCxEN bit of the CLCxCON register.

**FIGURE 31-2: INPUT DATA SELECTION AND GATING**



**FIGURE 31-3: PROGRAMMABLE LOGIC FUNCTIONS**



## 31.7 Register Definitions: CLC Control

**REGISTER 31-1: 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
LCxEN	—	LCxOUT	LCxINTP	LCxINTN	LCxMODE<2:0>		
bit 7							bit 0

**Legend:**

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

- bit 7      **LCxEN:** Configurable Logic Cell Enable bit  
 1 = Configurable logic cell is enabled and mixing input signals  
 0 = Configurable logic cell is disabled and has logic zero output
- bit 6      **Unimplemented:** Read as '0'
- bit 5      **LCxOUT:** Configurable Logic Cell Data Output bit  
 Read-only: logic cell output data, after LCPOL; sampled from CLCxOUT
- bit 4      **LCxINTP:** Configurable Logic Cell Positive Edge Going Interrupt Enable bit  
 1 = CLCxIF will be set when a rising edge occurs on CLCxOUT  
 0 = CLCxIF will not be set
- bit 3      **LCxINTN:** Configurable Logic Cell Negative Edge Going Interrupt Enable bit  
 1 = CLCxIF will be set when a falling edge occurs on CLCxOUT  
 0 = CLCxIF will not be set
- bit 2-0    **LCxMODE<2:0>:** Configurable Logic Cell Functional Mode bits  
 111 =Cell is 1-input transparent latch with S and R  
 110 =Cell is J-K flip-flop with R  
 101 =Cell is 2-input D flip-flop with R  
 100 =Cell is 1-input D flip-flop with S and R  
 011 =Cell is S-R latch  
 010 =Cell is 4-input AND  
 001 =Cell is OR-XOR  
 000 =Cell is AND-OR

## REGISTER 31-2: CLCxPOL: SIGNAL POLARITY CONTROL REGISTER

R/W-0/0	U-0	U-0	U-0	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
LCxPOL	—	—	—	LCxG4POL	LCxG3POL	LCxG2POL	LCxG1POL
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

u = Bit is unchanged

x = Bit is unknown

-n/n = Value at POR and BOR/Value at all other Resets

'1' = Bit is set

'0' = Bit is cleared

- bit 7      **LCxPOL:** CLCxOUT Output Polarity Control bit  
 1 = The output of the logic cell is inverted  
 0 = The output of the logic cell is not inverted
- bit 6-4    **Unimplemented:** Read as '0'
- bit 3      **LCxG4POL:** Gate 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 2      **LCxG3POL:** 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 1      **LCxG2POL:** 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
- bit 0      **LCxG1POL:** Gate 0 Output Polarity Control bit  
 1 = The output of gate 0 is inverted when applied to the logic cell  
 0 = The output of gate 0 is not inverted

## REGISTER 31-3: CLCxSEL0: GENERIC CLCx DATA 0 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
—	—	LCxD1S<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                      **LCxD1S<5:0>:** CLCx Data1 Input Selection bits  
See [Table 31-2](#).

## REGISTER 31-4: CLCxSEL1: 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
—	—	LCxD2S<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                      **LCxD2S<5:0>:** CLCx Data 2 Input Selection bits  
See [Table 31-2](#).

## REGISTER 31-5: CLCxSEL2: 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
—	—	LCxD3S<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                      **LCxD3S<5:0>:** CLCx Data 3 Input Selection bits  
See [Table 31-2](#).

## REGISTER 31-6: CLCxSEL3: 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
—	—	LCxD4S<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                      **LCxD4S<5:0>:** CLCx Data 4 Input Selection bits  
See [Table 31-2](#).



## REGISTER 31-7: CLCxGLS0: GATE 0 LOGIC SELECT REGISTER

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
LCxG1D4T	LCxG1D4N	LCxG1D3T	LCxG1D3N	LCxG1D2T	LCxG1D2N	LCxG1D1T	LCxG1D1N
bit 7							bit 0

### Legend:

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

bit 7	<b>LCxG1D4T:</b> Gate 0 Data 4 True (non-inverted) bit 1 = CLCIN3 (true) is gated into CLCx Gate 0 0 = CLCIN3 (true) is not gated into CLCx Gate 0
bit 6	<b>LCxG1D4N:</b> Gate 0 Data 4 Negated (inverted) bit 1 = CLCIN3 (inverted) is gated into CLCx Gate 0 0 = CLCIN3 (inverted) is not gated into CLCx Gate 0
bit 5	<b>LCxG1D3T:</b> Gate 0 Data 3 True (non-inverted) bit 1 = CLCIN2 (true) is gated into CLCx Gate 0 0 = CLCIN2 (true) is not gated into CLCx Gate 0
bit 4	<b>LCxG1D3N:</b> Gate 0 Data 3 Negated (inverted) bit 1 = CLCIN2 (inverted) is gated into CLCx Gate 0 0 = CLCIN2 (inverted) is not gated into CLCx Gate 0
bit 3	<b>LCxG1D2T:</b> Gate 0 Data 2 True (non-inverted) bit 1 = CLCIN1 (true) is gated into CLCx Gate 0 0 = CLCIN1 (true) is not gated into CLCx Gate 0
bit 2	<b>LCxG1D2N:</b> Gate 0 Data 2 Negated (inverted) bit 1 = CLCIN1 (inverted) is gated into CLCx Gate 0 0 = CLCIN1 (inverted) is not gated into CLCx Gate 0
bit 1	<b>LCxG1D1T:</b> Gate 0 Data 1 True (non-inverted) bit 1 = CLCIN0 (true) is gated into CLCx Gate 0 0 = CLCIN0 (true) is not gated into CLCx Gate 0
bit 0	<b>LCxG1D1N:</b> Gate 0 Data 1 Negated (inverted) bit 1 = CLCIN0 (inverted) is gated into CLCx Gate 0 0 = CLCIN0 (inverted) is not gated into CLCx Gate 0

## REGISTER 31-8: CLCxGLS1: 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
LCxG2D4T	LCxG2D4N	LCxG2D3T	LCxG2D3N	LCxG2D2T	LCxG2D2N	LCxG2D1T	LCxG2D1N
bit 7							bit 0

### Legend:

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

bit 7	<b>LCxG2D4T:</b> Gate 1 Data 4 True (non-inverted) bit 1 = CLCIN3 (true) is gated into CLCx Gate 1 0 = CLCIN3 (true) is not gated into CLCx Gate 1
bit 6	<b>LCxG2D4N:</b> Gate 1 Data 4 Negated (inverted) bit 1 = CLCIN3 (inverted) is gated into CLCx Gate 1 0 = CLCIN3 (inverted) is not gated into CLCx Gate 1
bit 5	<b>LCxG2D3T:</b> Gate 1 Data 3 True (non-inverted) bit 1 = CLCIN2 (true) is gated into CLCx Gate 1 0 = CLCIN2 (true) is not gated into CLCx Gate 1
bit 4	<b>LCxG2D3N:</b> Gate 1 Data 3 Negated (inverted) bit 1 = CLCIN2 (inverted) is gated into CLCx Gate 1 0 = CLCIN2 (inverted) is not gated into CLCx Gate 1
bit 3	<b>LCxG2D2T:</b> Gate 1 Data 2 True (non-inverted) bit 1 = CLCIN1 (true) is gated into CLCx Gate 1 0 = CLCIN1 (true) is not gated into CLCx Gate 1
bit 2	<b>LCxG2D2N:</b> Gate 1 Data 2 Negated (inverted) bit 1 = CLCIN1 (inverted) is gated into CLCx Gate 1 0 = CLCIN1 (inverted) is not gated into CLCx Gate 1
bit 1	<b>LCxG2D1T:</b> Gate 1 Data 1 True (non-inverted) bit 1 = CLCIN0 (true) is gated into CLCx Gate 1 0 = CLCIN0 (true) is not gated into CLCx Gate 1
bit 0	<b>LCxG2D1N:</b> Gate 1 Data 1 Negated (inverted) bit 1 = CLCIN0 (inverted) is gated into CLCx Gate 1 0 = CLCIN0 (inverted) is not gated into CLCx Gate 1

## REGISTER 31-9: CLCxGLS2: 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
LCxG3D4T	LCxG3D4N	LCxG3D3T	LCxG3D3N	LCxG3D2T	LCxG3D2N	LCxG3D1T	LCxG3D1N
bit 7							bit 0

### Legend:

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

bit 7	<b>LCxG3D4T:</b> Gate 2 Data 4 True (non-inverted) bit 1 = CLCIN3 (true) is gated into CLCx Gate 2 0 = CLCIN3 (true) is not gated into CLCx Gate 2
bit 6	<b>LCxG3D4N:</b> Gate 2 Data 4 Negated (inverted) bit 1 = CLCIN3 (inverted) is gated into CLCx Gate 2 0 = CLCIN3 (inverted) is not gated into CLCx Gate 2
bit 5	<b>LCxG3D3T:</b> Gate 2 Data 3 True (non-inverted) bit 1 = CLCIN2 (true) is gated into CLCx Gate 2 0 = CLCIN2 (true) is not gated into CLCx Gate 2
bit 4	<b>LCxG3D3N:</b> Gate 2 Data 3 Negated (inverted) bit 1 = CLCIN2 (inverted) is gated into CLCx Gate 2 0 = CLCIN2 (inverted) is not gated into CLCx Gate 2
bit 3	<b>LCxG3D2T:</b> Gate 2 Data 2 True (non-inverted) bit 1 = CLCIN1 (true) is gated into CLCx Gate 2 0 = CLCIN1 (true) is not gated into CLCx Gate 2
bit 2	<b>LCxG3D2N:</b> Gate 2 Data 2 Negated (inverted) bit 1 = CLCIN1 (inverted) is gated into CLCx Gate 2 0 = CLCIN1 (inverted) is not gated into CLCx Gate 2
bit 1	<b>LCxG3D1T:</b> Gate 2 Data 1 True (non-inverted) bit 1 = CLCIN0 (true) is gated into CLCx Gate 2 0 = CLCIN0 (true) is not gated into CLCx Gate 2
bit 0	<b>LCxG3D1N:</b> Gate 2 Data 1 Negated (inverted) bit 1 = CLCIN0 (inverted) is gated into CLCx Gate 2 0 = CLCIN0 (inverted) is not gated into CLCx Gate 2

## REGISTER 31-10: CLCxGLS3: 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
LCxG4D4T	LCxG4D4N	LCxG4D3T	LCxG4D3N	LCxG4D2T	LCxG4D2N	LCxG4D1T	LCxG4D1N
bit 7							bit 0

### Legend:

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

bit 7	<b>LCxG4D4T:</b> Gate 3 Data 4 True (non-inverted) bit 1 = CLCIN3 (true) is gated into CLCx Gate 3 0 = CLCIN3 (true) is not gated into CLCx Gate 3
bit 6	<b>LCxG4D4N:</b> Gate 3 Data 4 Negated (inverted) bit 1 = CLCIN3 (inverted) is gated into CLCx Gate 3 0 = CLCIN3 (inverted) is not gated into CLCx Gate 3
bit 5	<b>LCxG4D3T:</b> Gate 3 Data 3 True (non-inverted) bit 1 = CLCIN2 (true) is gated into CLCx Gate 3 0 = CLCIN2 (true) is not gated into CLCx Gate 3
bit 4	<b>LCxG4D3N:</b> Gate 3 Data 3 Negated (inverted) bit 1 = CLCIN2 (inverted) is gated into CLCx Gate 3 0 = CLCIN2 (inverted) is not gated into CLCx Gate 3
bit 3	<b>LCxG4D2T:</b> Gate 3 Data 2 True (non-inverted) bit 1 = CLCIN1 (true) is gated into CLCx Gate 3 0 = CLCIN1 (true) is not gated into CLCx Gate 3
bit 2	<b>LCxG4D2N:</b> Gate 3 Data 2 Negated (inverted) bit 1 = CLCIN1 (inverted) is gated into CLCx Gate 3 0 = CLCIN1 (inverted) is not gated into CLCx Gate 3
bit 1	<b>LCxG4D1T:</b> Gate 4 Data 1 True (non-inverted) bit 1 = CLCIN0 (true) is gated into CLCx Gate 3 0 = CLCIN0 (true) is not gated into CLCx Gate 3
bit 0	<b>LCxG4D1N:</b> Gate 3 Data 1 Negated (inverted) bit 1 = CLCIN0 (inverted) is gated into CLCx Gate 3 0 = CLCIN0 (inverted) is not gated into CLCx Gate 3

## REGISTER 31-11: 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 31-4: SUMMARY OF REGISTERS ASSOCIATED WITH CLCx**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
INTCON	GIE	PEIE	—	—	—	—	—	INTEDG	121
PIR5	CLC4IF	CLC3IF	CLC2IF	CLC1IF	—	—	—	TMR1GIF	135
PIE5	CLC4IE	CLC4IE	CLC2IE	CLC1IE	—	—	—	TMR1GIE	127
CLC1CON	LC1EN	—	LC1OUT	LC1INTP	LC1INTN	LC1MODE<2:0>			362
CLC1POL	LC1POL	—	—	—	LC1G4POL	LC1G3POL	LC1G2POL	LC1G1POL	363
CLC1SEL0	—	—	LC1D1S<5:0>						364
CLC1SEL1	—	—	LC1D2S<5:0>						364
CLC1SEL2	—	—	LC1D3S<5:0>						364
CLC1SEL3	—	—	LC1D4S<5:0>						364
CLC1GLS0	—	—	LC1G1D3T	LC1G1D3N	LC1G1D2T	LC1G1D2N	LC1G1D1T	LC1G1D1N	365
CLC1GLS1	—	—	LC1G2D3T	LC1G2D3N	LC1G2D2T	LC1G2D2N	LC1G2D1T	LC1G2D1N	366
CLC1GLS2	—	—	LC1G3D3T	LC1G3D3N	LC1G3D2T	LC1G3D2N	LC1G3D1T	LC1G3D1N	367
CLC1GLS3	—	—	LC1G4D3T	LC1G4D3N	LC1G4D2T	LC1G4D2N	LC1G4D1T	LC1G4D1N	368
CLC2CON	LC2EN	—	LC2OUT	LC2INTP	LC2INTN	LC2MODE<2:0>			362
CLC2POL	LC2POL	—	—	—	LC2G4POL	LC2G3POL	LC2G2POL	LC2G1POL	363
CLC2SEL0	—	—	LC2D1S<5:0>						364
CLC2SEL1	—	—	LC2D2S<5:0>						364
CLC2SEL2	—	—	LC2D3S<5:0>						364
CLC2SEL3	—	—	LC2D4S<5:0>						364
CLC2GLS0	—	—	LC2G1D3T	LC2G1D3N	LC2G1D2T	LC2G1D2N	LC2G1D1T	LC2G1D1N	365
CLC2GLS1	—	—	LC2G2D3T	LC2G2D3N	LC2G2D2T	LC2G2D2N	LC2G2D1T	LC2G2D1N	366
CLC2GLS2	—	—	LC2G3D3T	LC2G3D3N	LC2G3D2T	LC2G3D2N	LC2G3D1T	LC2G3D1N	367
CLC2GLS3	—	—	LC2G4D3T	LC2G4D3N	LC2G4D2T	LC2G4D2N	LC2G4D1T	LC2G4D1N	368
CLC3CON	LC3EN	—	LC3OUT	LC3INTP	LC3INTN	LC3MODE<2:0>			362
CLC3POL	LC3POL	—	—	—	LC3G4POL	LC3G3POL	LC3G2POL	LC3G1POL	363
CLC3SEL0	—	—	LC3D1S<5:0>						364
CLC3SEL1	—	—	LC3D2S<5:0>						364
CLC3SEL2	—	—	LC3D3S<5:0>						364
CLC3SEL3	—	—	LC3D4S<5:0>						364
CLC3GLS0	—	—	LC3G1D3T	LC3G1D3N	LC3G1D2T	LC3G1D2N	LC3G1D1T	LC3G1D1N	365
CLC3GLS1	—	—	LC3G2D3T	LC3G2D3N	LC3G2D2T	LC3G2D2N	LC3G2D1T	LC3G2D1N	366
CLC3GLS2	—	—	LC3G3D3T	LC3G3D3N	LC3G3D2T	LC3G3D2N	LC3G3D1T	LC3G3D1N	367
CLC3GLS3	—	—	LC3G4D3T	LC3G4D3N	LC3G4D2T	LC3G4D2N	LC3G4D1T	LC3G4D1N	368
CLC4CON	LC4EN	—	LC4OUT	LC4INTP	LC4INTN	LC4MODE<2:0>			362
CLC4POL	LC4POL	—	—	—	LC4G4POL	LC4G3POL	LC4G2POL	LC4G1POL	363
CLC4SEL0	—	—	LC4D1S<5:0>						364
CLC4SEL1	—	—	LC4D2S<5:0>						364
CLC4SEL2	—	—	LC4D3S<5:0>						364
CLC4SEL3	—	—	LC4D4S<5:0>						364
CLC4GLS0	—	—	LC4G1D3T	LC4G1D3N	LC4G1D2T	LC4G1D2N	LC4G1D1T	LC4G1D1N	365

**Legend:** — = unimplemented, read as '0'. Shaded cells are unused by the CLCx modules.

**TABLE 31-4: SUMMARY OF REGISTERS ASSOCIATED WITH CLCx (continued)**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
CLC4GLS1	—	—	LC4G2D3T	LC4G2D3N	LC4G2D2T	LC4G2D2N	LC4G2D1T	LC4G2D1N	<a href="#">366</a>
CLC4GLS2	—	—	LC4G3D3T	LC4G3D3N	LC4G3D2T	LC4G3D2N	LC4G3D1T	LC4G3D1N	<a href="#">367</a>
CLC4GLS3	—	—	LC4G4D3T	LC4G4D3N	LC4G4D2T	LC4G4D2N	LC4G4D1T	LC4G4D1N	<a href="#">368</a>
CLCIN0PPS	—	—	CLCIN0PPS<5:0>						<a href="#">199</a>
CLCIN1PPS	—	—	CLCIN1PPS<5:0>						<a href="#">199</a>
CLCIN2PPS	—	—	CLCIN2PPS<5:0>						<a href="#">199</a>
CLCIN3PPS	—	—	CLCIN3PPS<5:0>						<a href="#">199</a>

**Legend:** — = unimplemented, read as '0'. Shaded cells are unused by the CLCx modules.

## 32.0 MASTER SYNCHRONOUS SERIAL PORT (MSSPx) MODULES

### 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:

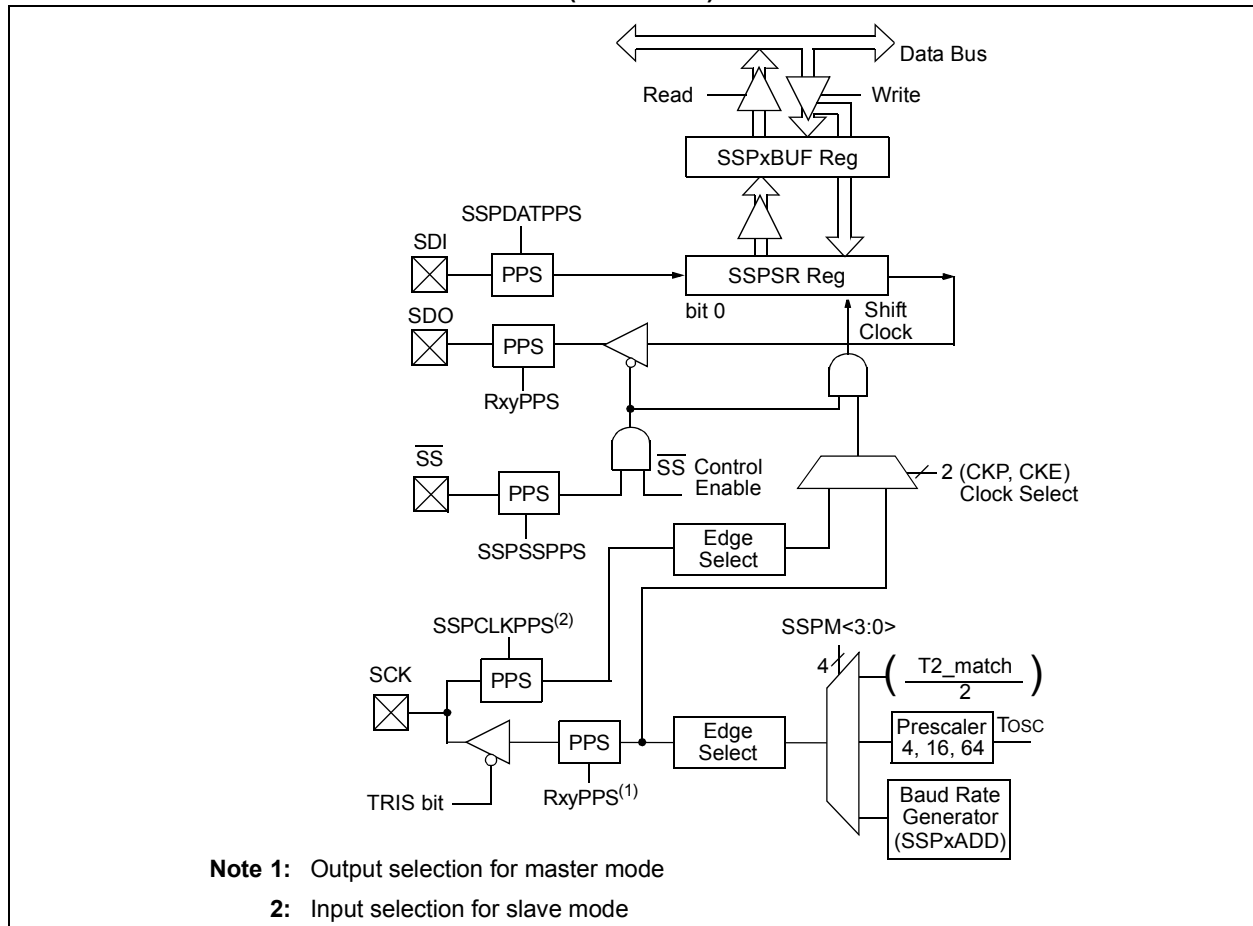
- Serial Peripheral Interface (SPI)
- Inter-Integrated Circuit (I<sup>2</sup>C)

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.

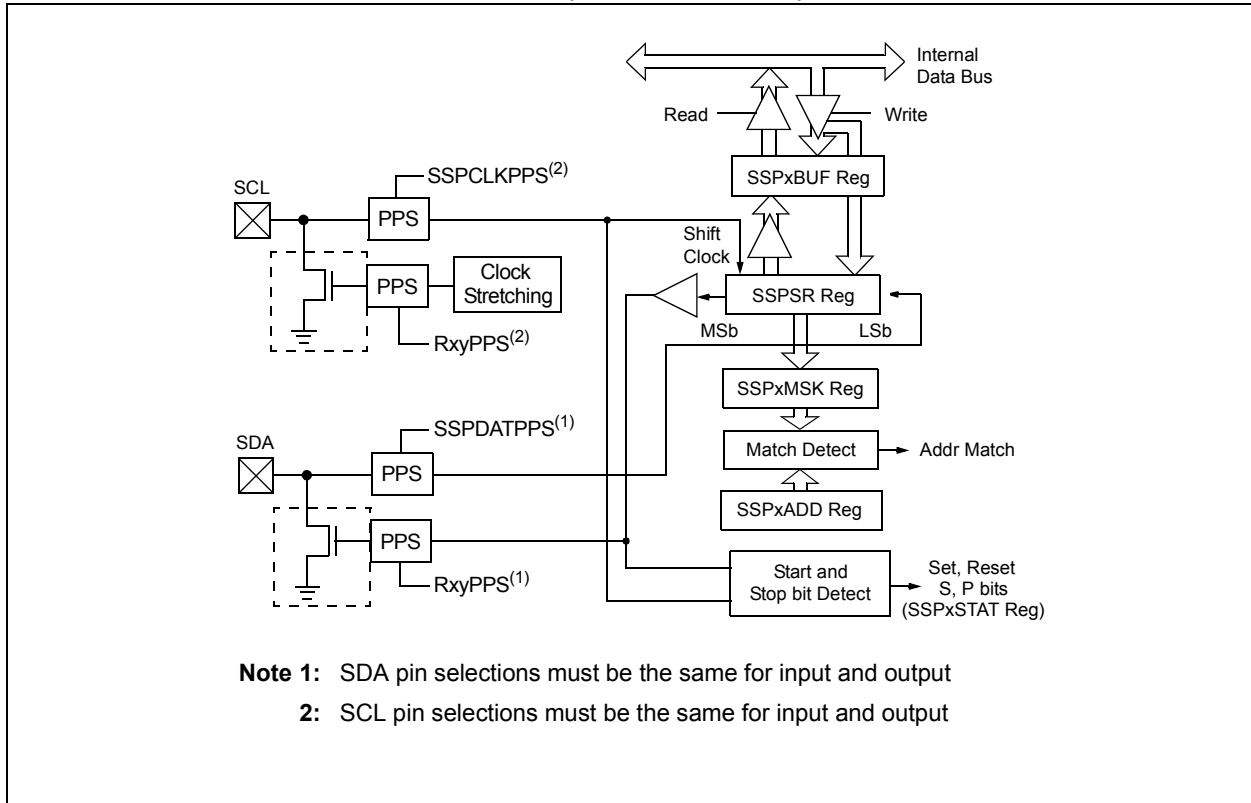
**FIGURE 32-1: MSSP BLOCK DIAGRAM (SPI MODE)**







**FIGURE 32-3: MSSP BLOCK DIAGRAM (I<sup>2</sup>C SLAVE MODE)**



## 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 can be used to address each slave individually.

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. 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.

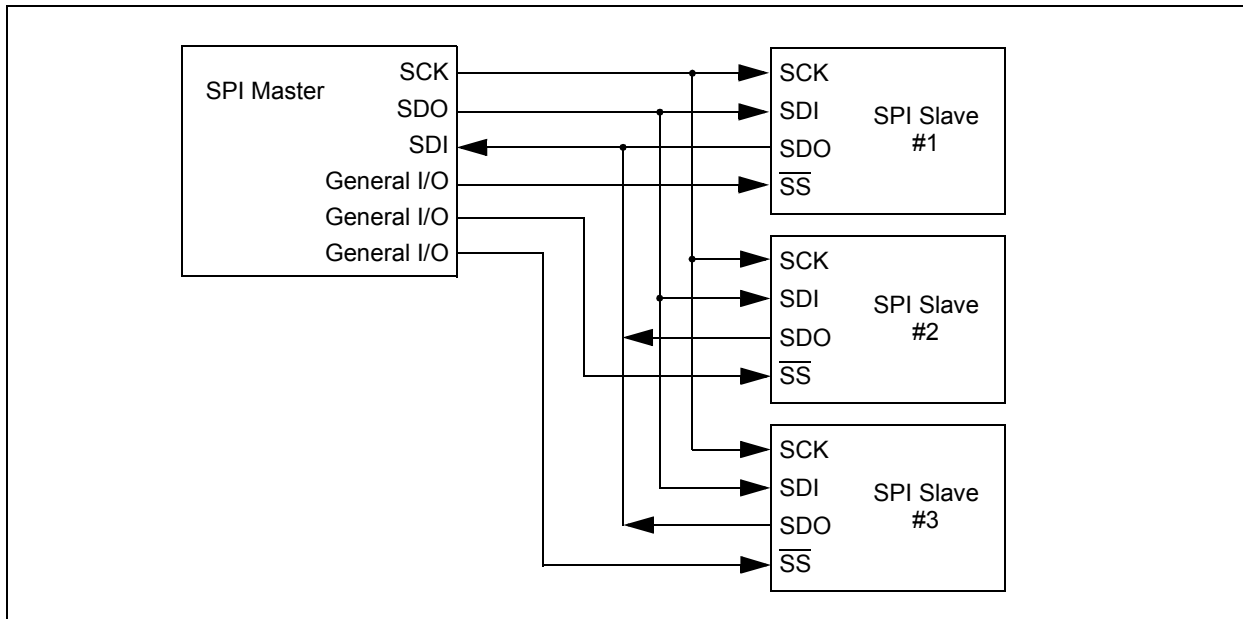
Whether the data is meaningful or not (dummy data), depends on the application software. This leads to three scenarios for data transmission:

- 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 must be performed in multiples of eight clock pulses. 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 (SSPxSR)  
(Not directly accessible)

SSPxCON1 and SSPxSTAT are the control and status registers in SPI mode operation. The SSPxCON1 register is readable and writable. The lower six bits of the SSPxSTAT are read-only. The upper two bits of the SSPxSTAT are read/write.

In 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”](#).

SSPxSR is the shift register used for shifting data in and out. SSPxBUF provides indirect access to the SSPxSR register. SSPxBUF is the buffer register to which data bytes are written, and from which data bytes are read.

In receive operations, SSPxSR and SSPxBUF together create a buffered receiver. When SSPxSR receives a complete byte, it is transferred to SSPxBUF and the SSPxIF interrupt is set.

During transmission, the SSPxBUF is not buffered. A write to SSPxBUF will write to both SSPxBUF and SSPxSR.

## 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<3:0> 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 TRISx 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

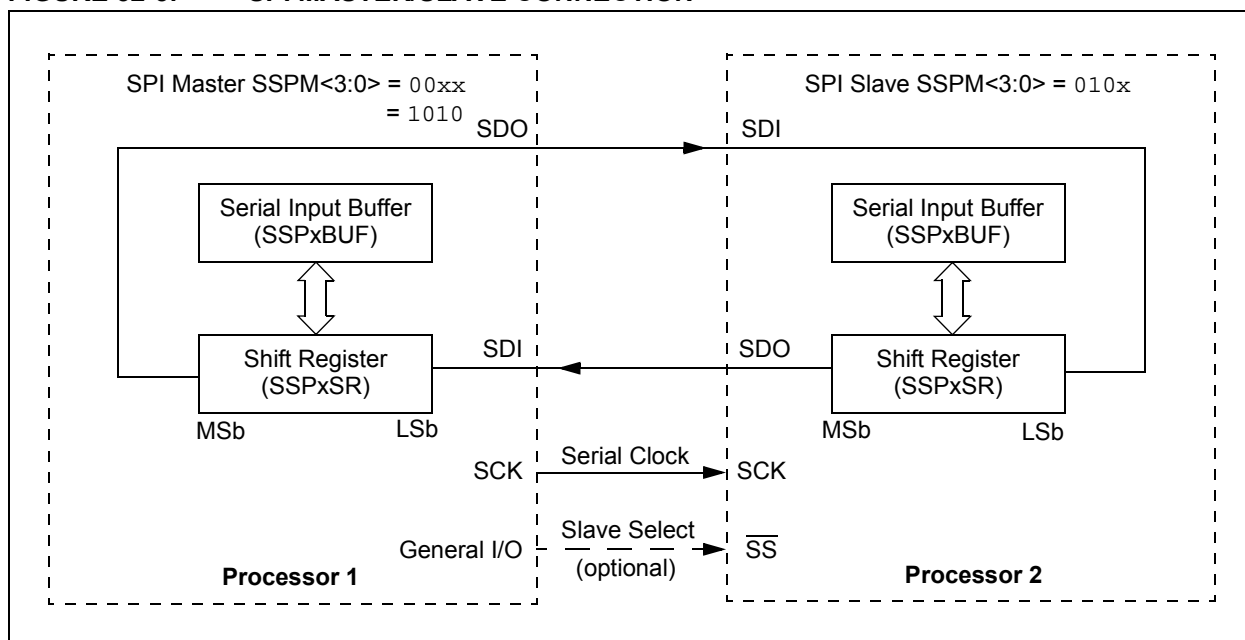
Any serial port function that is not desired may be overridden by programming the corresponding data direction (TRIS) register to the opposite value.

The MSSP consists of a transmit/receive shift register (SSPxSR) and a buffer register (SSPxBUF). The SSPxSR shifts the data in and out of the device, MSb first. The SSPxBUF holds the data that was written to the SSPxSR until the received data is ready. Once the eight bits of data have been received, that byte is moved to the SSPxBUF register. Then, the Buffer Full Detect bit, BF of the SSPxSTAT register, and the interrupt flag bit, SSPxIF, are set. 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 SSPxSR is not directly readable or writable and can only be accessed by addressing the SSPxBUF register.

**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 SSPxSR 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).

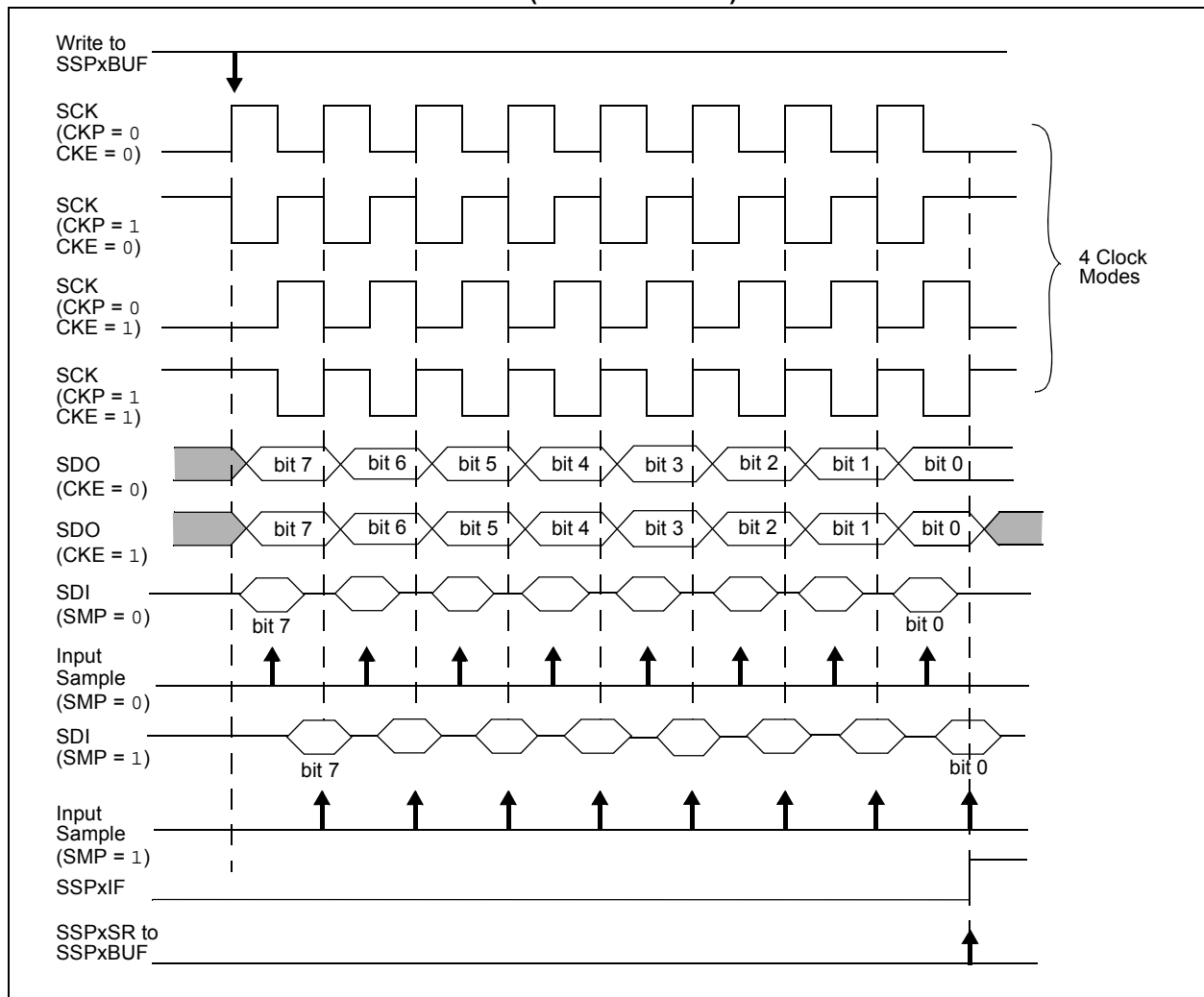
The clock polarity is selected by appropriately programming the CKP bit of the SSPxCON1 register and the CKE bit of the SSPxSTAT register. This then, would give waveforms for SPI communication as shown in [Figure 32-6](#), [Figure 32-8](#), [Figure 32-9](#) and [Figure 32-10](#), where the MSB is transmitted first. In Master mode, the SPI clock rate (bit rate) is user programmable to be one of the following:

- $F_{osc}/4$  (or  $T_{CY}$ )
- $F_{osc}/16$  (or  $4 * T_{CY}$ )
- $F_{osc}/64$  (or  $16 * T_{CY}$ )
- Timer2 output/2
- $F_{osc}/(4 * (SSPxADD + 1))$

[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.

**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).

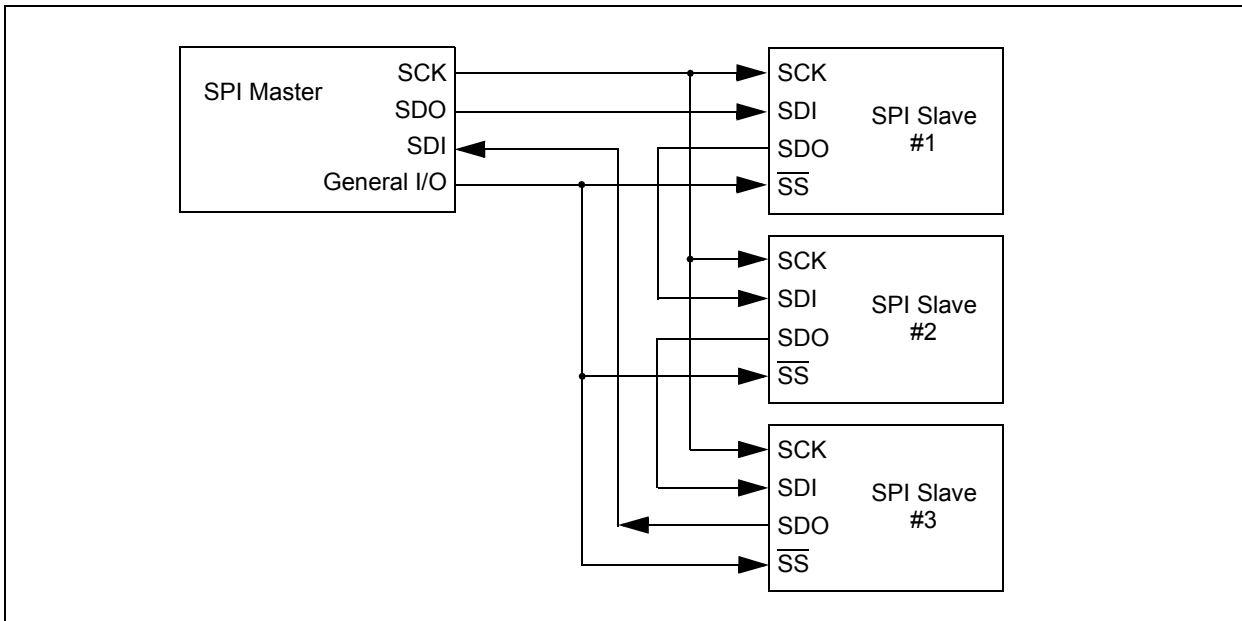
When the  $\overline{SS}$  pin is low, transmission and reception are enabled and the SDO pin is driven.

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.

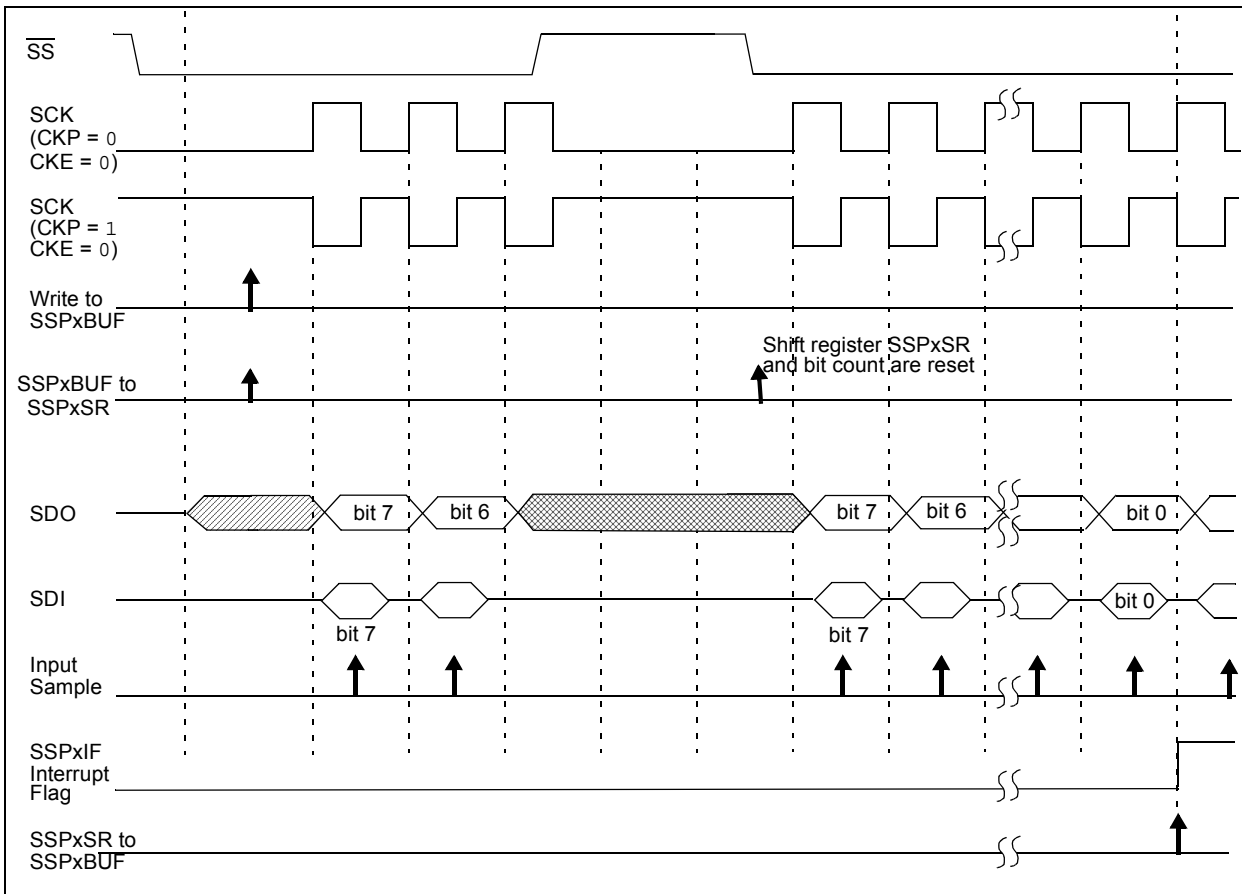
- Note 1:** When the SPI is in Slave mode with  $\overline{SS}$  pin control enabled (SSPxCON1<3:0> = 0100), the SPI module will reset if the  $\overline{SS}$  pin is set to VDD.
- 2:** When the SPI is used in Slave mode with CKE set; the user must enable  $\overline{SS}$  pin control.
- 3:** While operated 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**

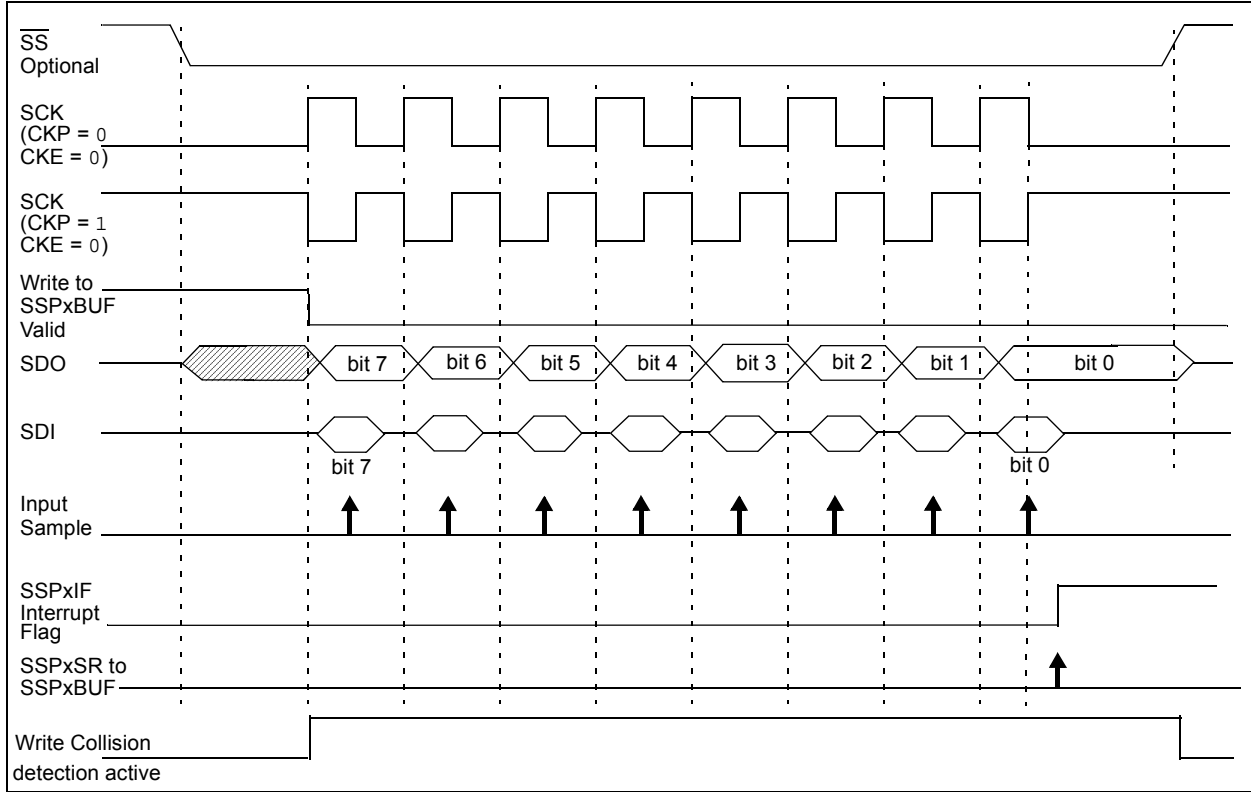


**FIGURE 32-8: SLAVE SELECT SYNCHRONOUS WAVEFORM**

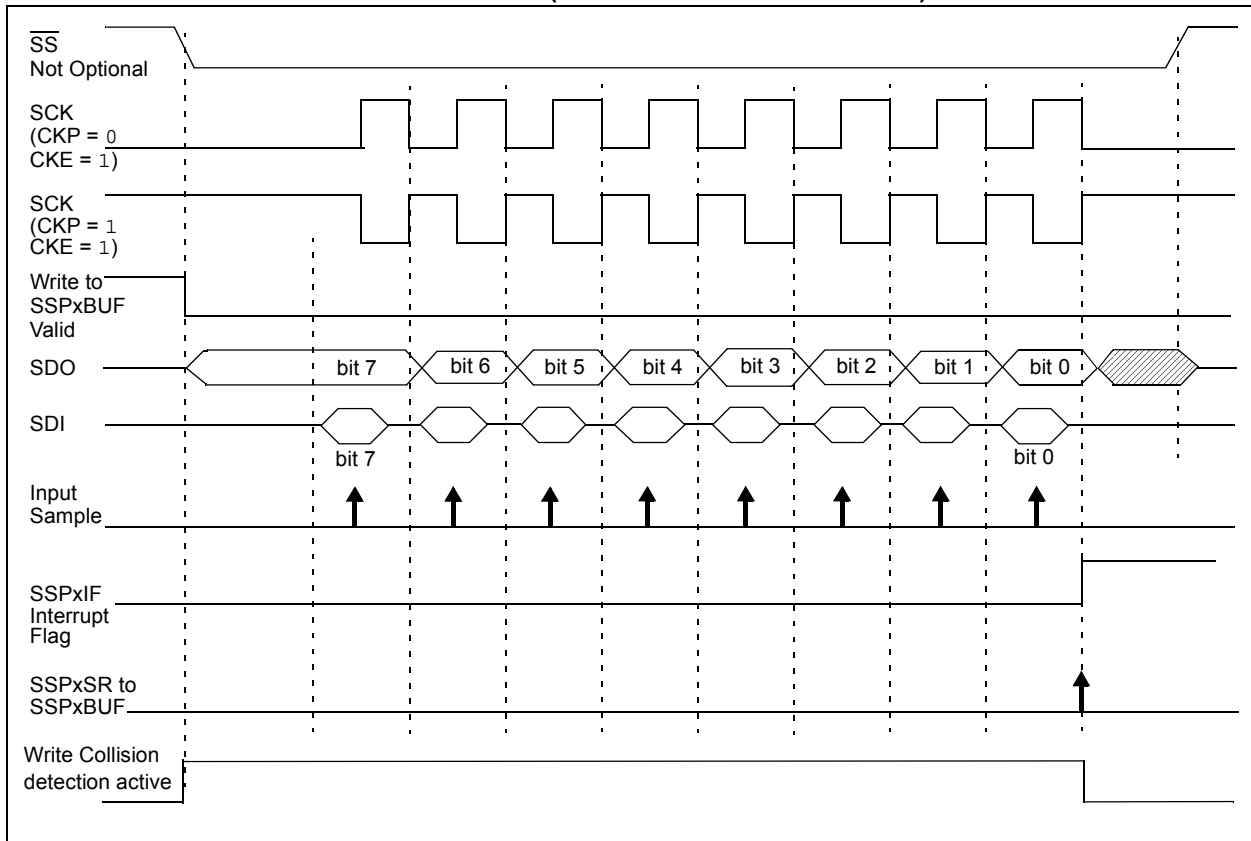




**FIGURE 32-9: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 0)**



**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.

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.

## 32.3 I<sup>2</sup>C MODE OVERVIEW

The Inter-Integrated Circuit (I<sup>2</sup>C) 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 I<sup>2</sup>C 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 I<sup>2</sup>C 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 I<sup>2</sup>C 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:

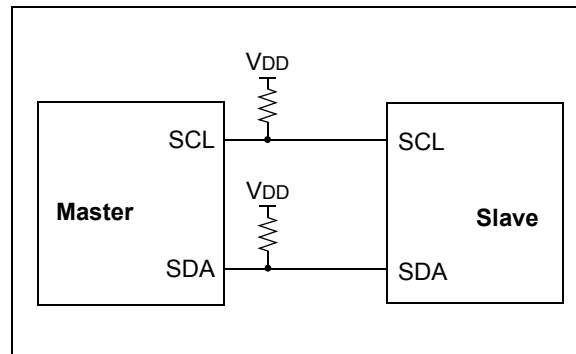
- Master Transmit mode  
(master is transmitting data to a slave)
- Master Receive mode  
(master is receiving data from a slave)
- Slave Transmit mode  
(slave is transmitting data to a master)
- Slave Receive mode  
(slave is receiving data from the master)

To begin communication, the master device sends out a Start condition 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 to either transmit or receive data from the slave device.

**FIGURE 32-11: I<sup>2</sup>C MASTER/SLAVE CONNECTION**



The line is held high to indicate Start and Stop bits.

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 a Restart condition in place of the Stop condition or last  $\overline{\text{ACK}}$  bit when it is in Receive mode.

The I<sup>2</sup>C 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.

## 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.

Slave Transmit mode can also be arbitrated, when a master addresses multiple slaves, but this is less common.

## 32.4 I<sup>2</sup>C MODE OPERATION

All MSSP I<sup>2</sup>C communication is byte oriented and shifted out MSb first. Six SFR registers and two interrupt flags interface the module with the PIC<sup>®</sup> microcontroller and user software. Two pins, SDA and SCL, are exercised by the module to communicate with other external I<sup>2</sup>C devices.

### 32.4.1 BYTE FORMAT

All communication in I<sup>2</sup>C 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 I<sup>2</sup>C TERMINOLOGY

There is language and terminology in the description of I<sup>2</sup>C communication that have definitions specific to I<sup>2</sup>C. 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 I<sup>2</sup>C specification.

### 32.4.3 SDA AND SCL PINS

Selection of any I<sup>2</sup>C 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:** 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 32-1: I<sup>2</sup>C 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/W bit clear, and is ready to clock in data.
Read Request	Master sends an address byte with the R/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<sup>2</sup>C 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.

## 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

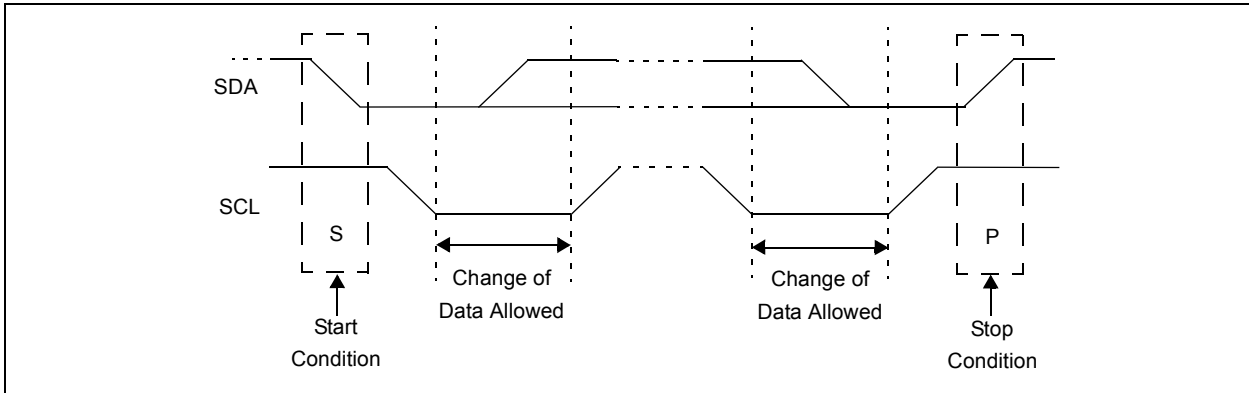
A Restart is valid any time that a Stop would be valid. A master can issue a Restart if it wishes to hold the bus after terminating the current transfer. A Restart has the same effect on the slave that a Start would, resetting all slave logic and preparing it to clock in an address. The master may want 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/W bit set. The slave logic will then hold the clock and prepare to clock out data.

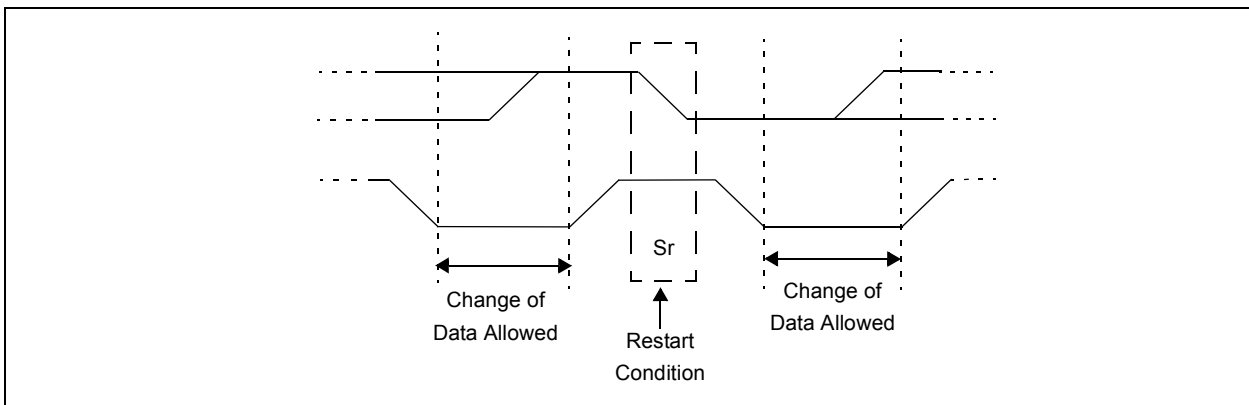
## 32.4.8 START/STOP CONDITION INTERRUPT MASKING

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<sup>2</sup>C START AND STOP CONDITIONS**



**FIGURE 32-13: I<sup>2</sup>C RESTART CONDITION**



### 32.4.9 ACKNOWLEDGE SEQUENCE

The 9th SCL pulse for any transferred byte in I<sup>2</sup>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.

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<sup>2</sup>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 32.5.9 “SSP Mask Register”](#) for more information.

#### 32.5.1.1 I<sup>2</sup>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<sup>2</sup>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 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/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/W bit of a matching received address byte is clear, the R/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.

#### 32.5.2.1 7-bit Addressing Reception

This section describes a standard sequence of events for the MSSP module configured as an I<sup>2</sup>C 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<sup>2</sup>C 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/W bit clear is received.
4. The slave pulls SDA low sending an  $\overline{\text{ACK}}$  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  $\overline{\text{ACK}}$  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 allows time for the slave software to decide whether it wants to ACK the receive address or data byte.

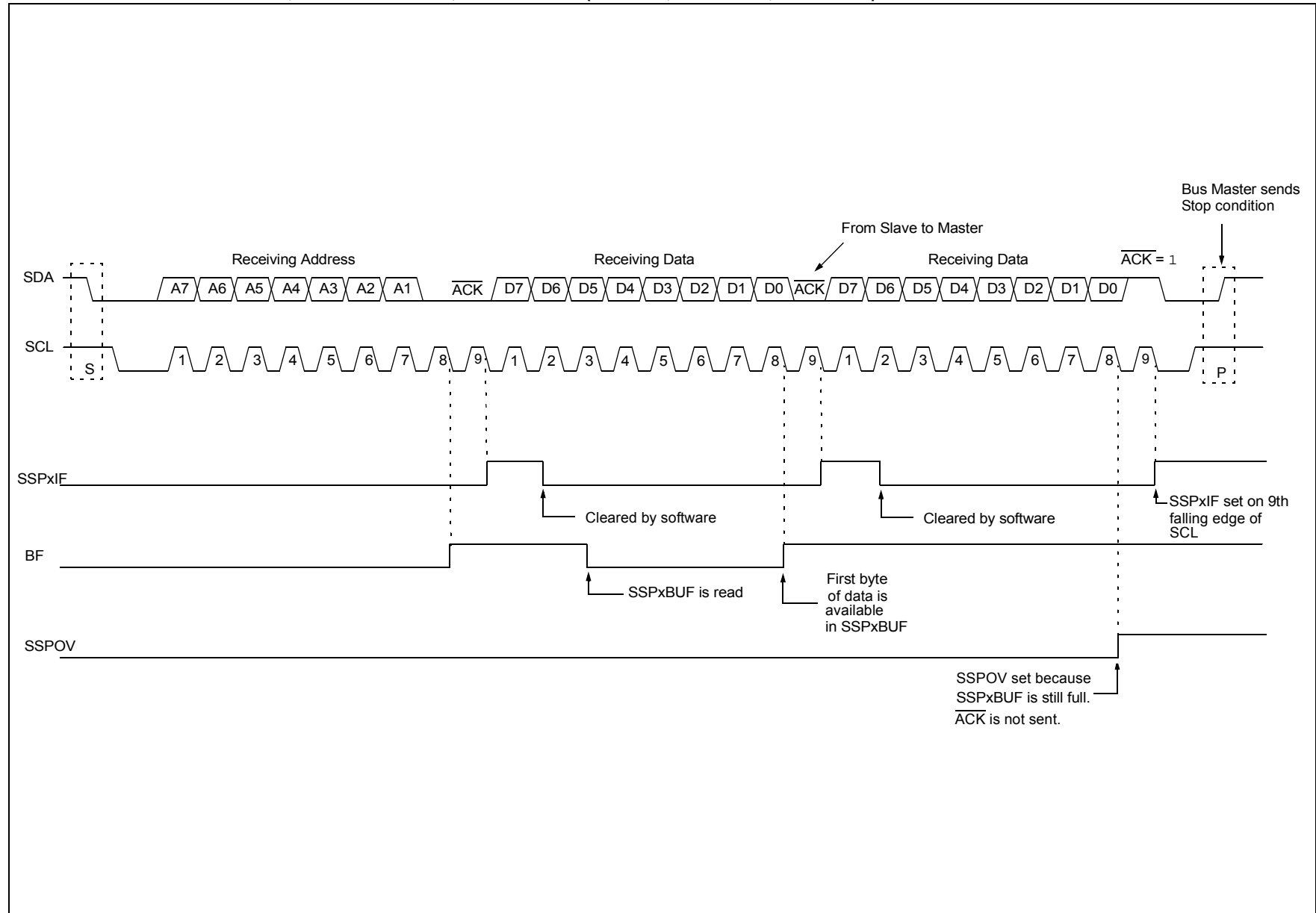
This list describes the steps that need to be taken by slave software to use these options for I<sup>2</sup>C communication. Figure 32-16 displays a module using both address and data holding. Figure 32-17 includes the operation with the SEN bit of the SSPxCON2 register set.

1. S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
2. Matching address with R/W bit clear is clocked in. SSPxIF is set, and CKP is cleared in hardware 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 ACK.
5. Slave reads the address value from SSPxBUF, clearing the BF flag.
6. Slave sets ACK value clocked out to the master by setting ACKDT.
7. Slave releases the clock by setting CKP in software.
8. SSPxIF is set after an ACK, not after a NACK.
9. If SEN = 1 the slave hardware will stretch the clock after the 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

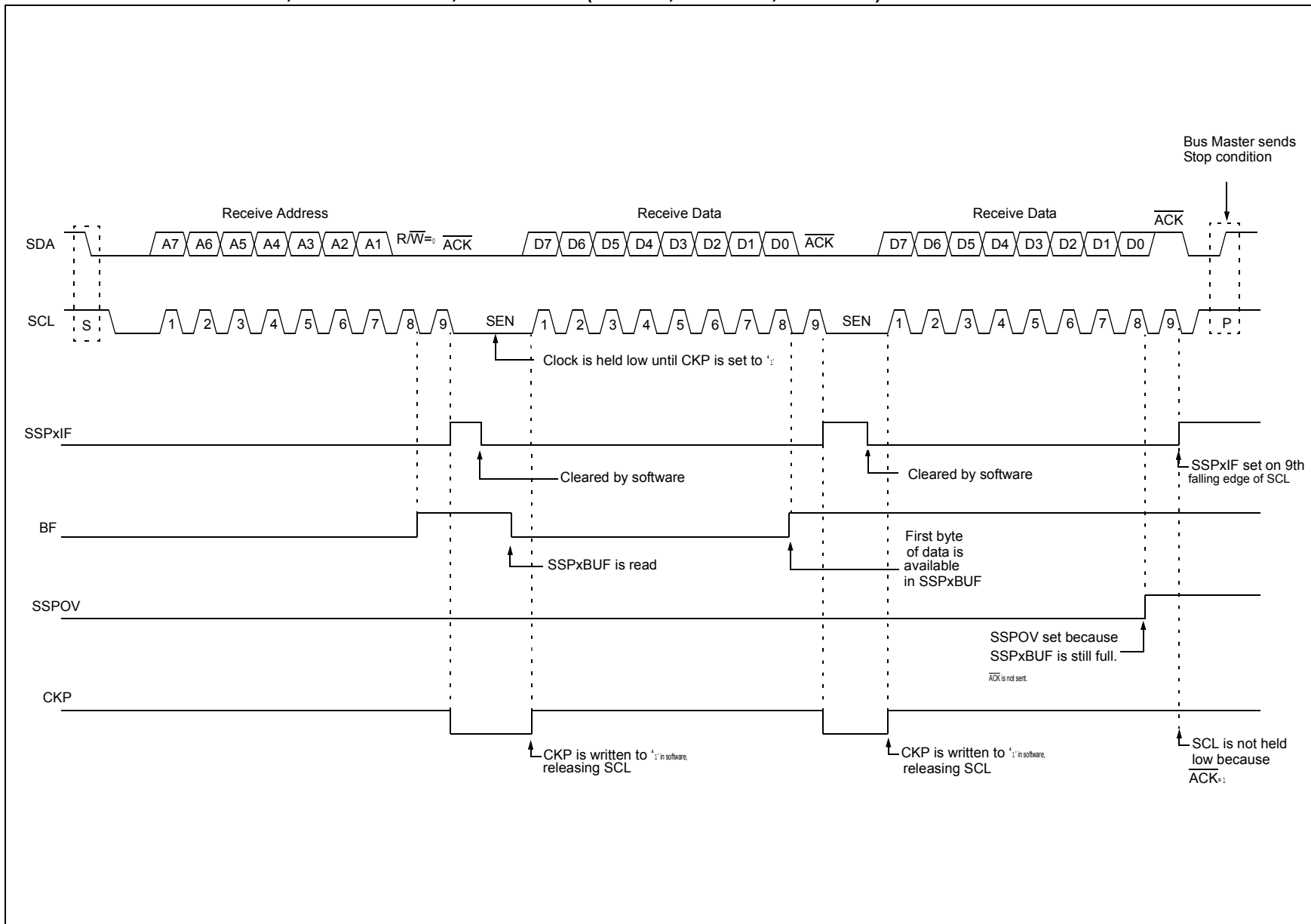
11. SSPxIF set, and CKP is cleared in hardware after eighth falling edge of SCL for a received data byte.
12. Slave looks at ACKTIM bit of SSPxCON3 to determine the source of the interrupt.
13. Slave reads the received data from SSPxBUF clearing BF.
14. Steps 7-14 are the same for each received data byte.
15. Communication is ended by either the slave sending an 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 SSPxSTAT register.

**FIGURE 32-14: I<sup>2</sup>C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 0, AHEN = 0, DHEN = 0)**

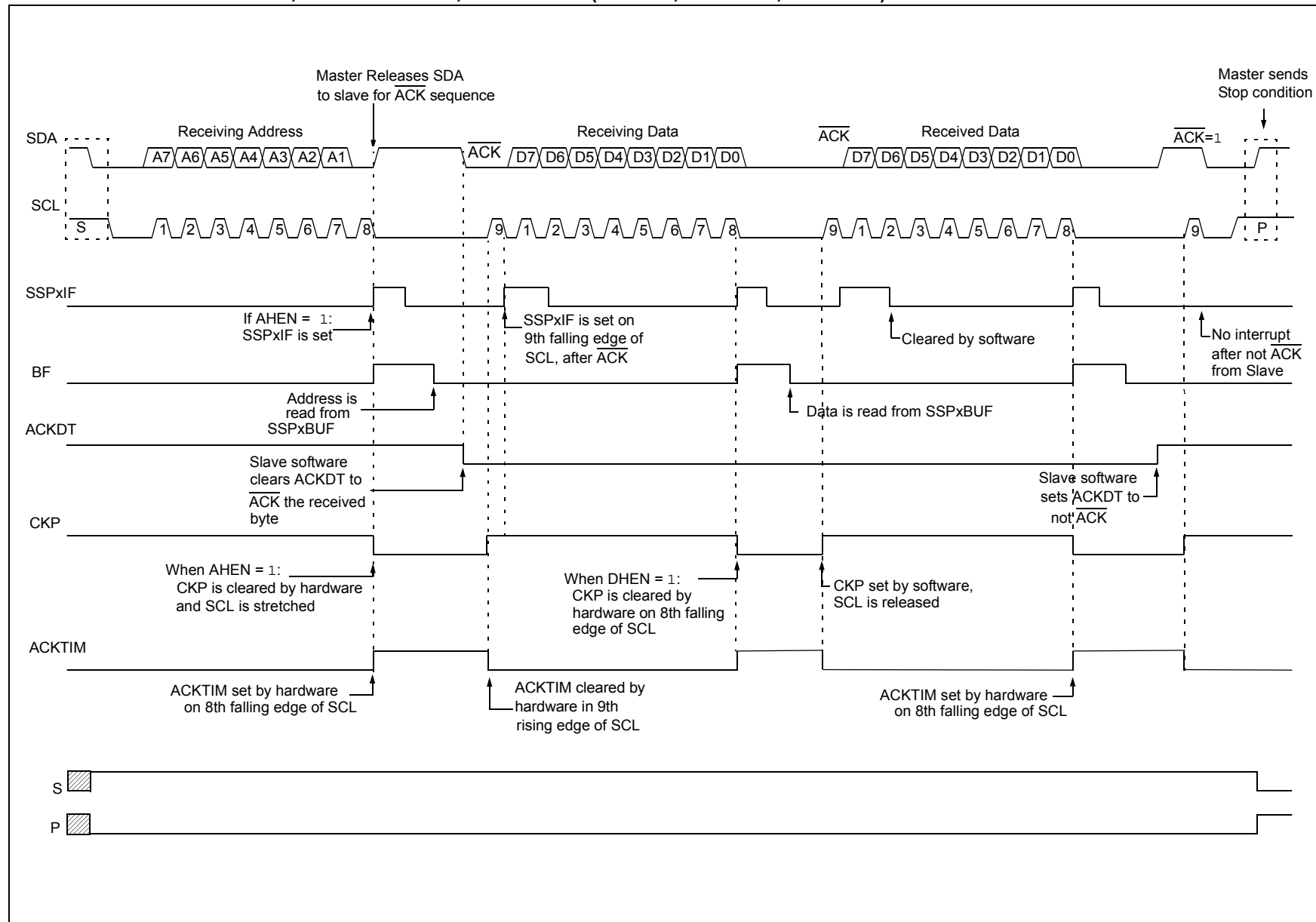




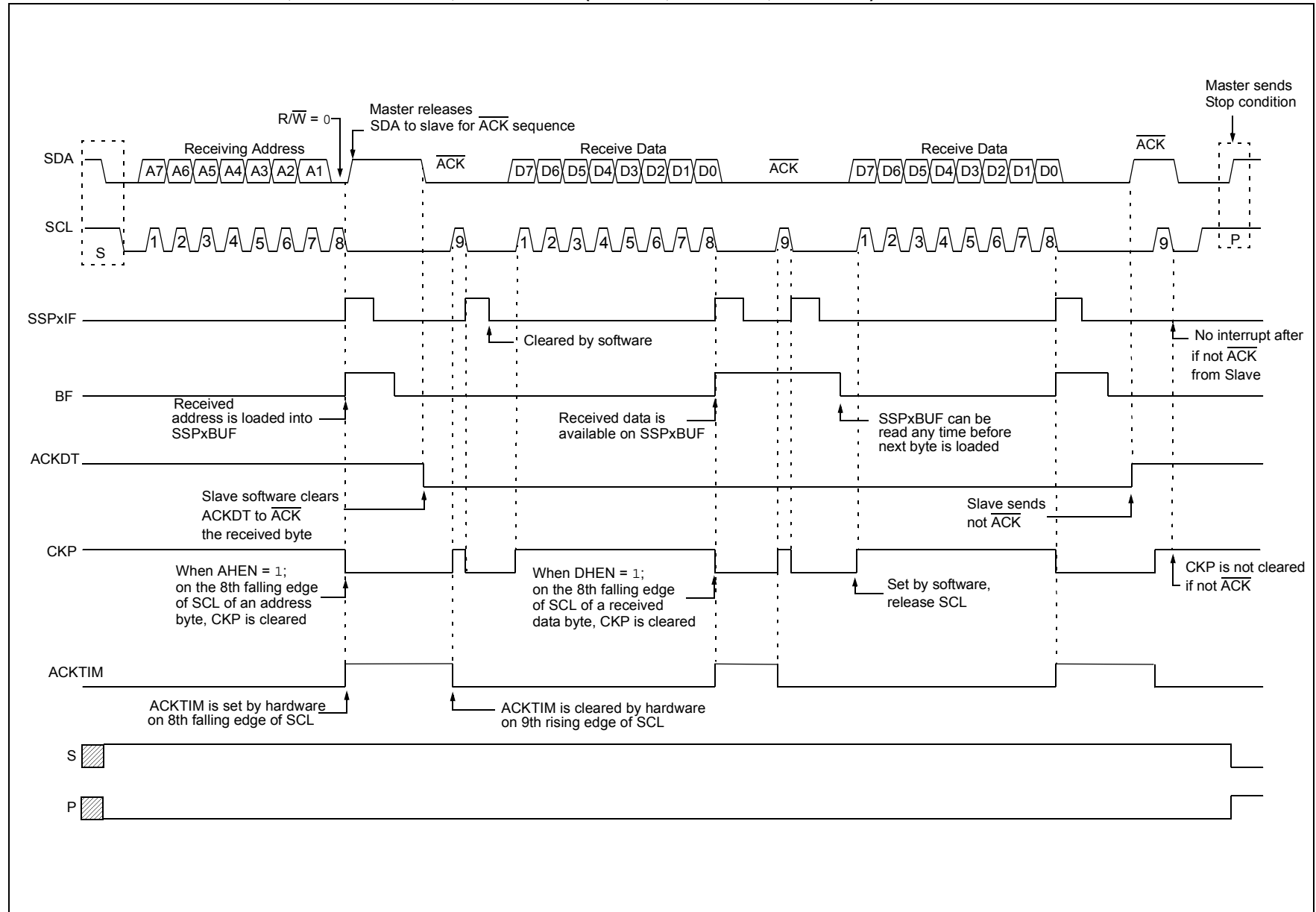
**FIGURE 32-15: I<sup>2</sup>C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 1, AHEN = 0, DHEN = 0)**



**FIGURE 32-16: I<sup>2</sup>C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 0, AHEN = 1, DHEN = 1)**



**FIGURE 32-17: I<sup>2</sup>C SLAVE, 7-BIT ADDRESS, RECEPTION (SEN = 1, AHEN = 1, DHEN = 1)**



## 32.5.3 SLAVE TRANSMISSION

When the  $\overline{R/W}$  bit of the incoming address byte is set and an address match occurs, the  $\overline{R/W}$  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 SSPxSR 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 BCL1IF bit of the PIR3 register is set. Once a bus collision is detected, the slave goes idle and waits to be addressed again. User software can use the BCL1IF 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.

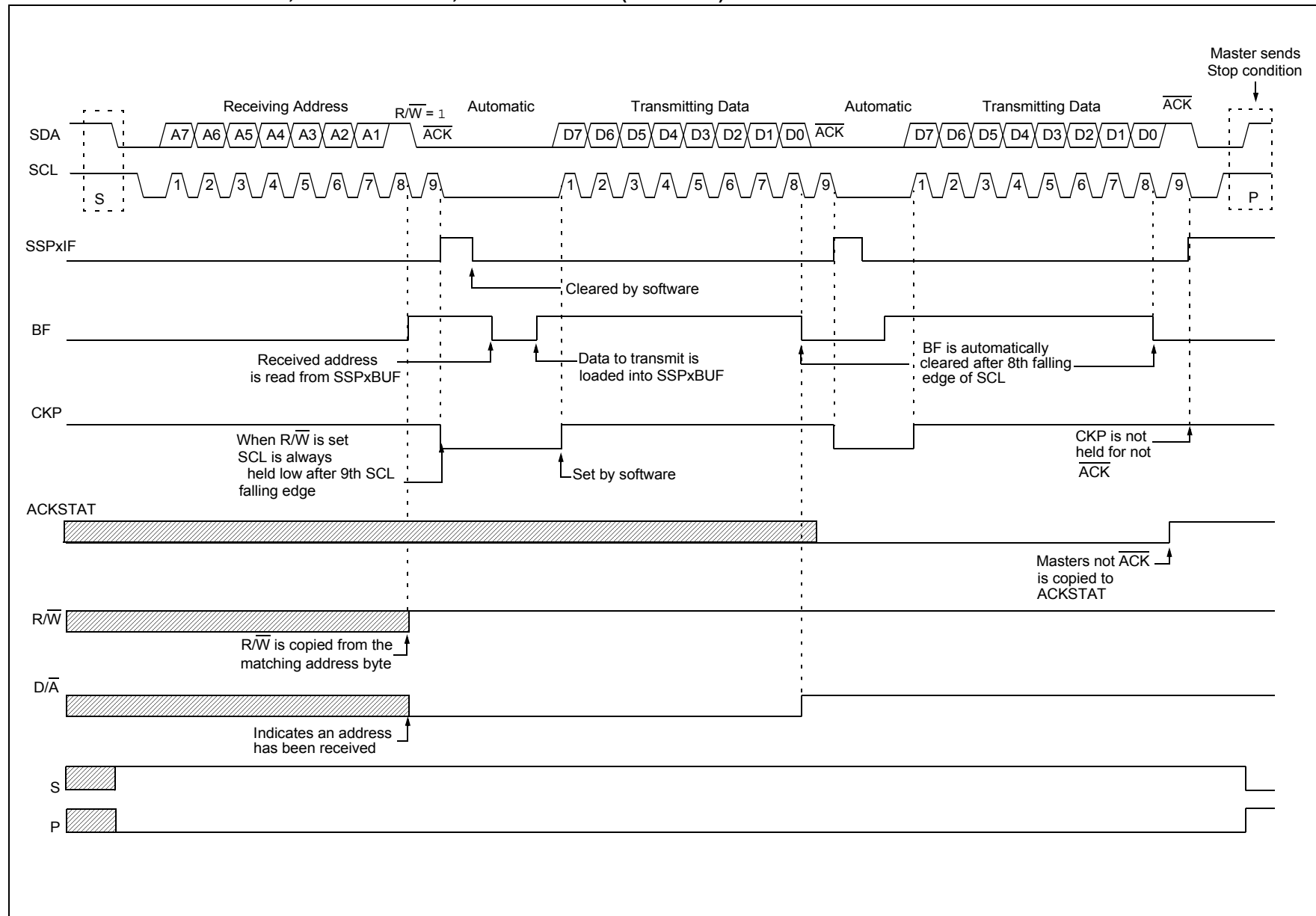
1. Master sends a Start condition on SDA and SCL.
2. S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
3. Matching address with  $\overline{R/W}$  bit set is received by the Slave setting SSPxIF bit.
4. Slave hardware generates an  $\overline{ACK}$  and sets SSPxIF.
5. SSPxIF bit is cleared by software.
6. Software reads the received address from SSPxBUF, clearing BF.
7.  $\overline{R/W}$  is set so CKP was automatically cleared by hardware after the  $\overline{ACK}$ .
8. The slave software loads the transmit data into SSPxBUF.
9. CKP bit is set in software, 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 bit.
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: I<sup>2</sup>C 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.

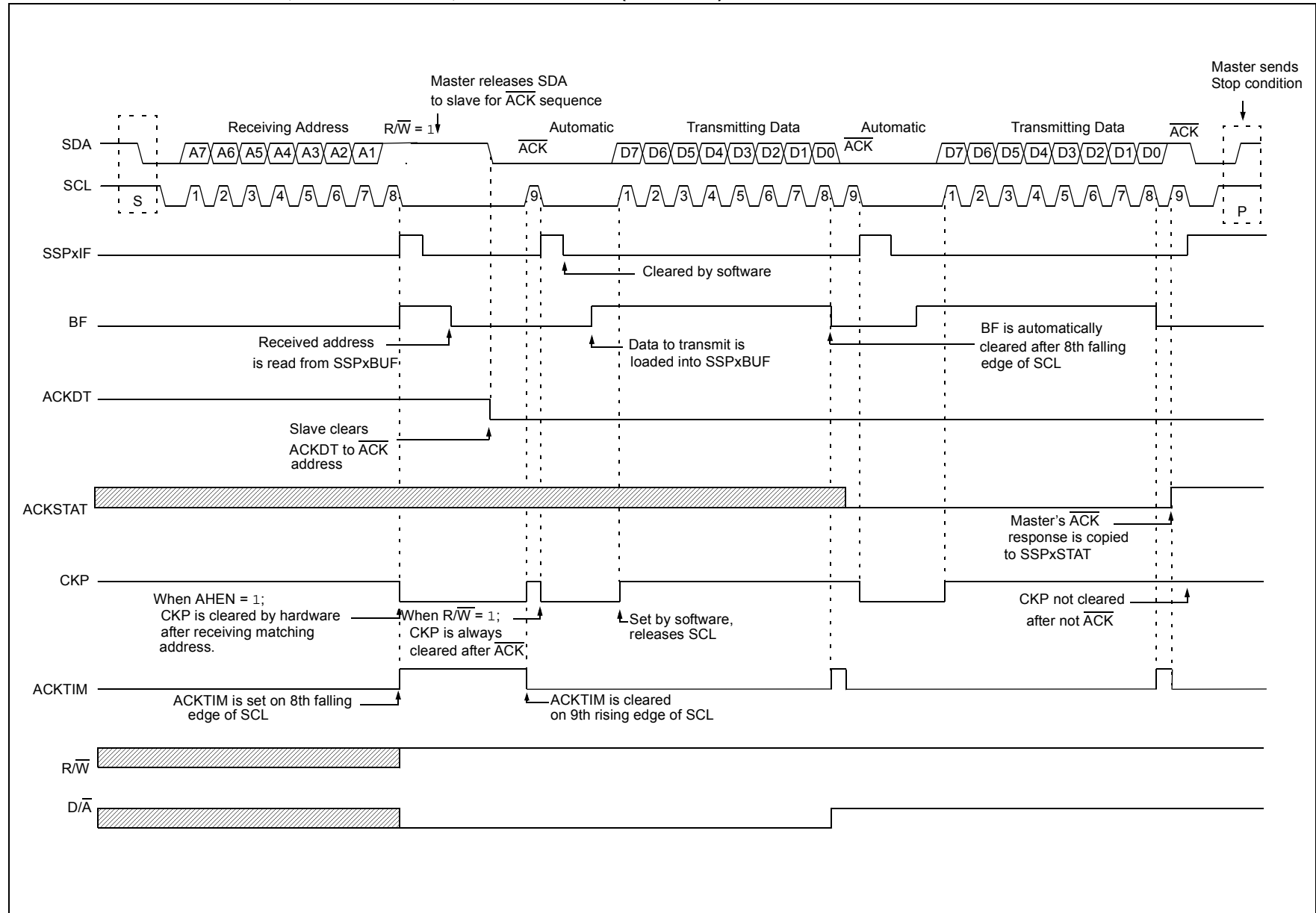
1. Master sends Start condition; the S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
2. Master sends matching address with  $\overline{R/W}$  bit set. After the eighth falling edge of the SCL line the CKP bit is cleared by hardware and SSPxIF interrupt is generated.
3. Slave software clears SSPxIF.
4. Slave software reads ACKTIM bit of SSPxCON3 register, and  $\overline{R/W}$  and D/A of the SSPxSTAT register to determine the source of the interrupt.
5. Slave reads the address value from the SSPxBUF register clearing the BF bit.
6. 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.
7. Slave software sets the CKP bit releasing SCL.
8. Master clocks in the  $\overline{ACK}$  value from the slave.
9. Slave hardware automatically clears the CKP bit and sets SSPxIF after the  $\overline{ACK}$  if the  $\overline{R/W}$  bit is set.
10. Slave software clears SSPxIF.
11. Slave loads value to transmit to the master into SSPxBUF setting the BF bit.

**Note:** SSPxBUF cannot be loaded until after the  $\overline{ACK}$ .

12. Slave sets the CKP bit releasing the clock.
13. Master clocks out the data from the slave and sends an  $\overline{ACK}$  value on the ninth SCL pulse.
14. Slave hardware copies the  $\overline{ACK}$  value into the ACKSTAT bit of the SSPxCON2 register.
15. Steps 10-15 are repeated for each byte transmitted to the master from the slave.
16. 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: I<sup>2</sup>C 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<sup>2</sup>C 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<sup>2</sup>C communication.

1. Master sends Start condition; S bit of SSPxSTAT is set; SSPxIF is set if interrupt on Start detect is enabled.
2. Master sends matching high address with R/W bit clear; UA bit of the SSPxSTAT register is set.
3. Slave sends  $\overline{\text{ACK}}$  and SSPxIF is set.
4. Software clears the SSPxIF bit.
5. Software reads received address from SSPxBUF clearing the BF flag.
6. Slave loads low address into SSPxADD, releasing SCL.
7. 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  $\overline{\text{ACK}}$  sequence.

8. Slave sends  $\overline{\text{ACK}}$  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.

9. Slave clears SSPxIF.
10. Slave reads the received matching address from SSPxBUF clearing BF.
11. Slave loads high address into SSPxADD.
12. Master clocks a data byte to the slave and clocks out the slaves  $\overline{\text{ACK}}$  on the ninth SCL pulse; SSPxIF is set.
13. If SEN bit of SSPxCON2 is set, CKP is cleared by hardware and the clock is stretched.
14. Slave clears SSPxIF.
15. Slave reads the received byte from SSPxBUF clearing BF.
16. If SEN is set the slave software sets CKP to release the SCL.
17. Steps 13-17 repeat for each received byte.
18. Master sends Stop to end the transmission.

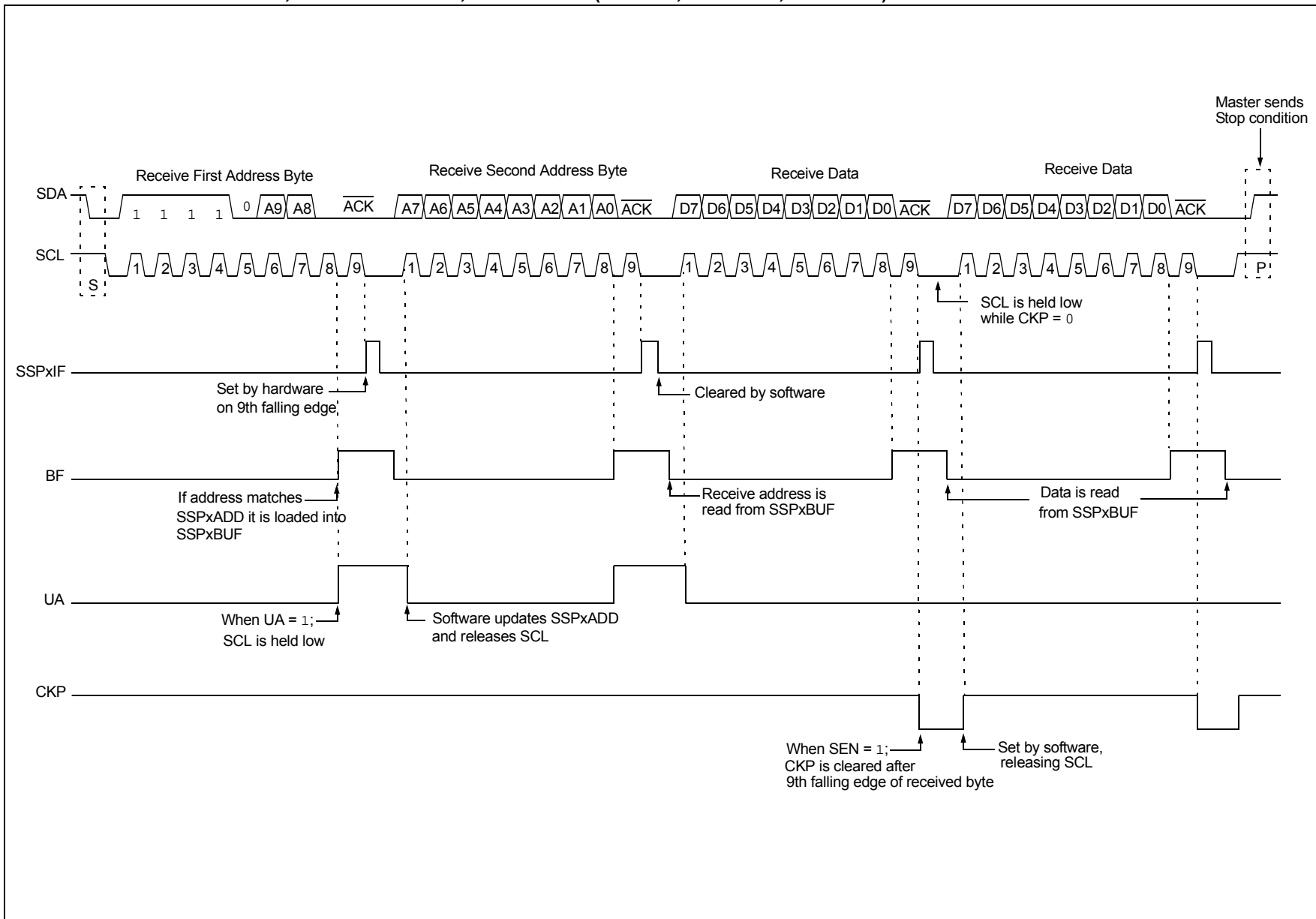
## 32.5.5 10-BIT ADDRESSING WITH ADDRESS OR DATA HOLD

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.

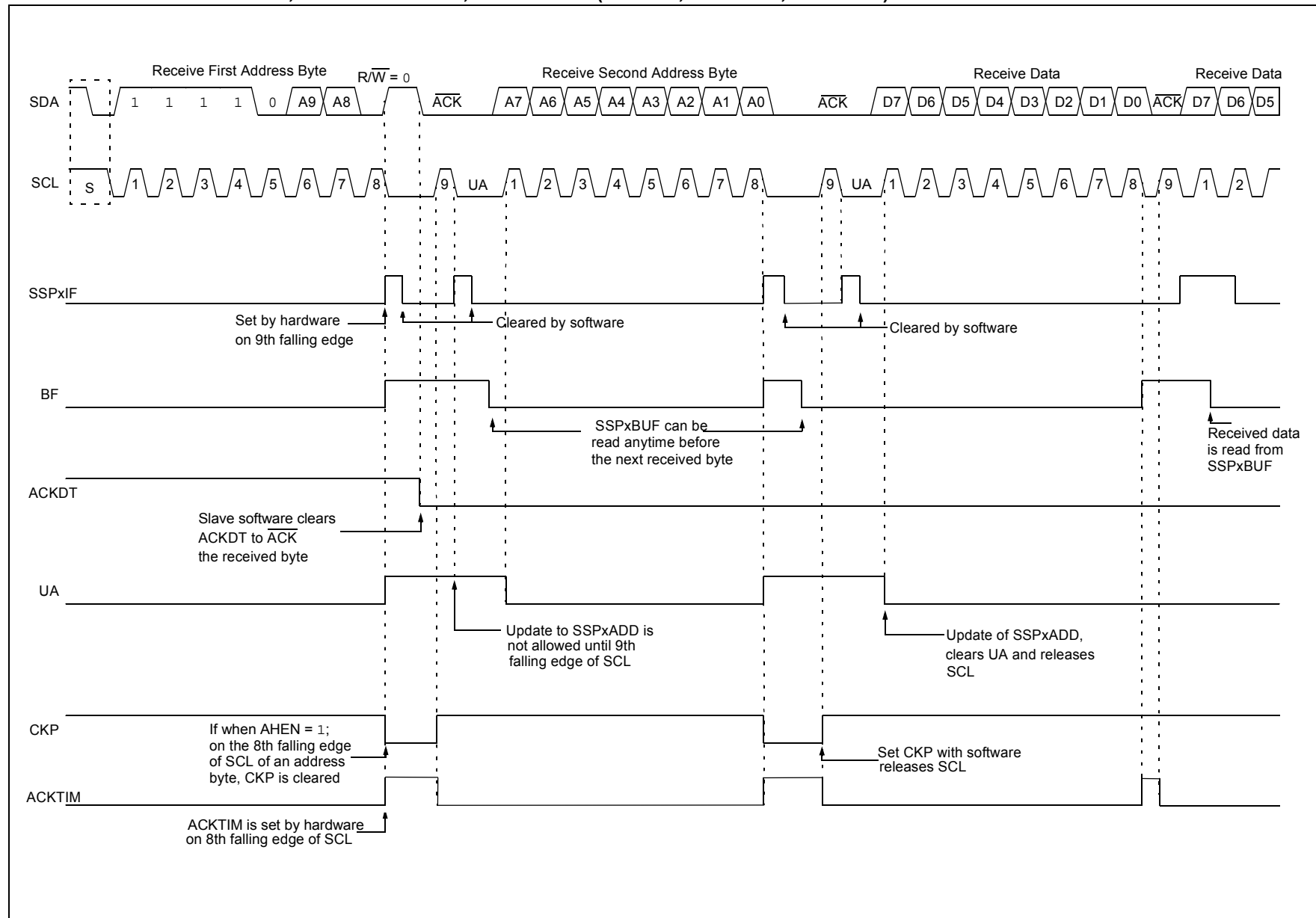
Figure 32-22 shows a standard waveform for a slave transmitter in 10-bit Addressing mode.



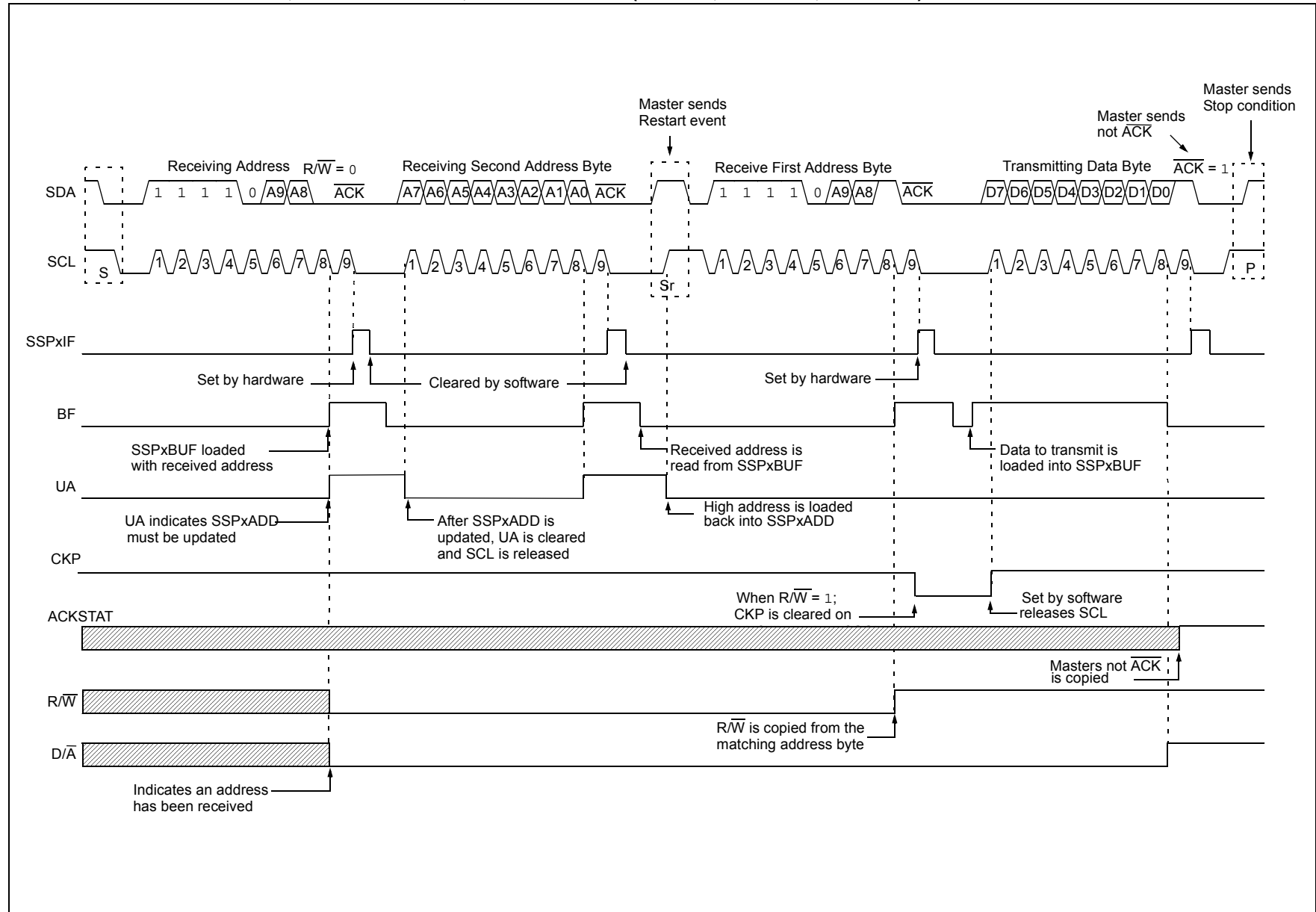
**FIGURE 32-20: I<sup>2</sup>C SLAVE, 10-BIT ADDRESS, RECEPTION (SEN = 1, AHEN = 0, DHEN = 0)**



**FIGURE 32-21: I<sup>2</sup>C SLAVE, 10-BIT ADDRESS, RECEPTION (SEN = 0, AHEN = 1, DHEN = 0)**



**FIGURE 32-22: I<sup>2</sup>C 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. 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  $\overline{\text{ACK}}$  if the  $\overline{\text{R/W}}$  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  $\overline{\text{ACK}}$  sequence. Once the slave is ready; CKP is set by software and communication resumes.

### 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.

### 32.5.6.3 Byte NACKing

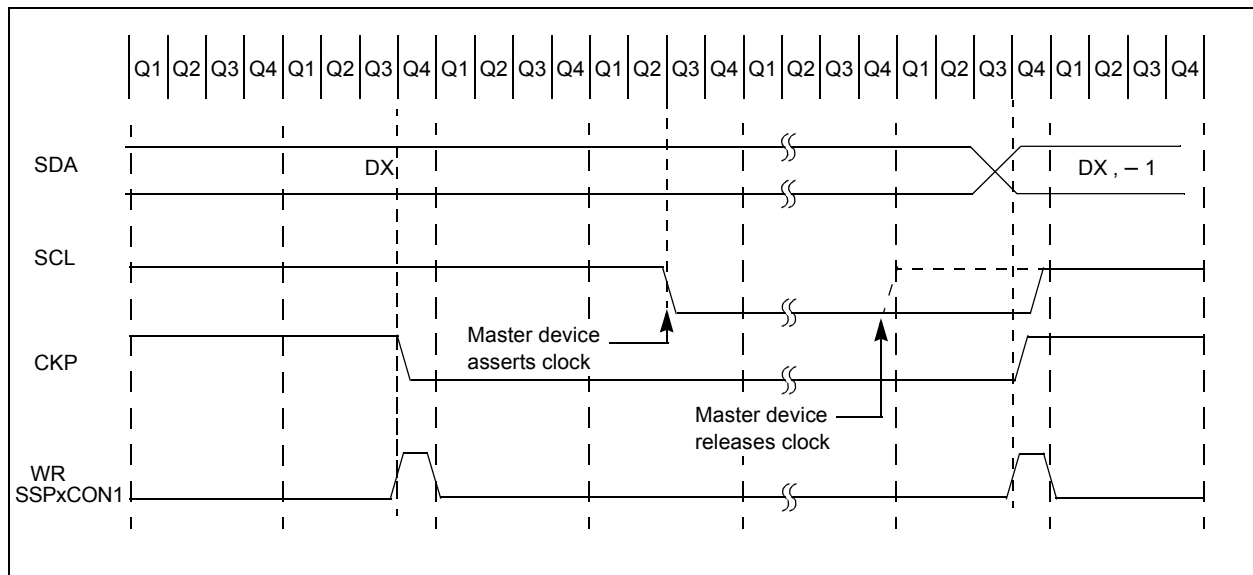
When AHEN bit of SSPxCON3 is set; CKP is cleared by hardware after the eighth falling edge of SCL for a received matching address byte. When DHEN bit of SSPxCON3 is set; CKP is cleared after the eighth falling edge of SCL for received data.

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.

### 32.5.7 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<sup>2</sup>C 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<sup>2</sup>C 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**



## 32.5.8 GENERAL CALL ADDRESS SUPPORT

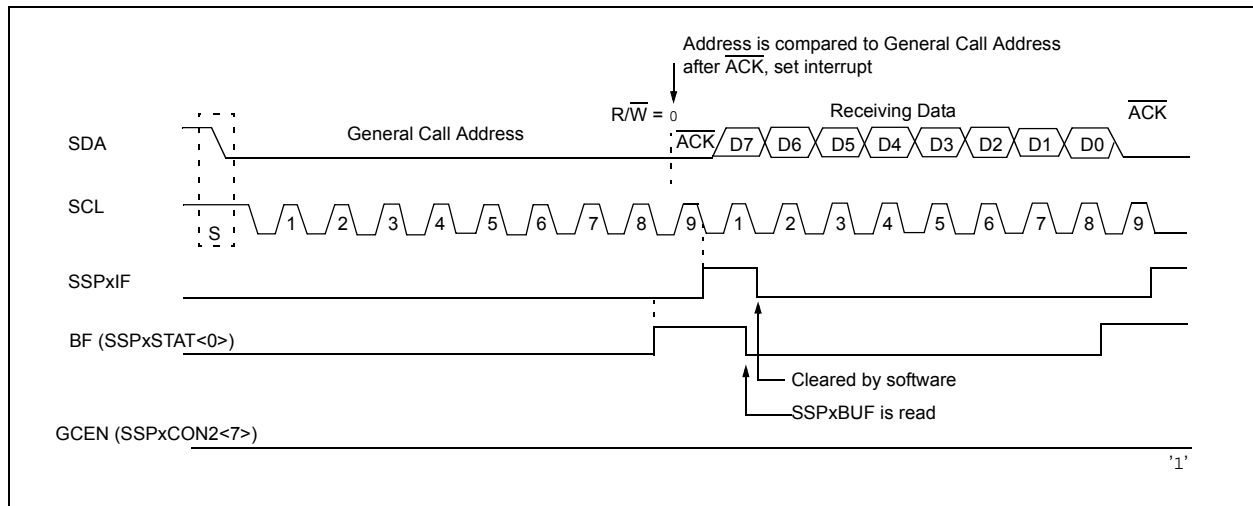
The addressing procedure for the I<sup>2</sup>C bus is such that the first byte after the Start condition usually determines which device will be the slave addressed by the master device. The exception is the general call address which can address all devices. When this address is used, all devices should, in theory, respond with an acknowledge.

The general call address is a reserved address in the I<sup>2</sup>C protocol, defined as address 0x00. When the GCEN bit of the SSPxCON2 register is set, the slave module will automatically ACK the reception of this address regardless of the value stored in SSPxADD. After the slave clocks in an address of all zeros with the R/W bit clear, an interrupt is generated and slave software can read SSPxBUF and respond. Figure 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.

**FIGURE 32-24: SLAVE MODE GENERAL CALL ADDRESS SEQUENCE**



## 32.5.9 SSP MASK REGISTER

An SSP Mask (SSPxMSK) register (Register 32-5) is available in I<sup>2</sup>C Slave mode as a mask for the value held in the SSPxSR register during an address comparison operation. A zero ('0') bit in the SSPxMSK register has the effect of making the corresponding bit of the received address a "don't care".

This register is reset to all '1's upon any Reset condition and, therefore, has no effect on standard 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.

## 32.6 I<sup>2</sup>C 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 I<sup>2</sup>C bus may be taken when the P bit is set, or the bus is Idle.

In Firmware Controlled Master mode, user code conducts all I<sup>2</sup>C 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 generated
- Stop condition generated
- Data transfer byte transmitted/received
- Acknowledge transmitted/received
- Repeated Start generated

**Note 1:** The MSSP module, when configured in I<sup>2</sup>C 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 I<sup>2</sup>C 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 I<sup>2</sup>C 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/W) bit. In this case, the R/W bit will be logic '0'. Serial data is transmitted eight bits at a time. After each byte is transmitted, an Acknowledge bit is received. Start and Stop conditions are output to indicate the beginning and the end of a serial transfer.

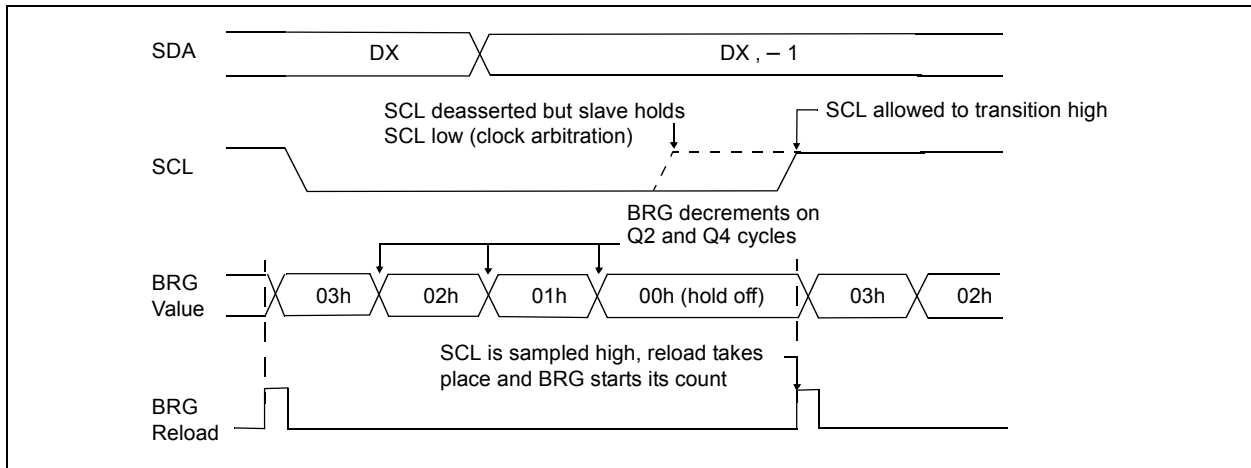
In Master Receive mode, the first byte transmitted contains the slave address of the transmitting device (7 bits) and the R/W bit. In this case, the R/W bit will be logic '1'. Thus, the first byte transmitted is a 7-bit slave address followed by a '1' to indicate the receive bit. Serial data is received via 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 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 I<sup>2</sup>C MASTER MODE START CONDITION TIMING

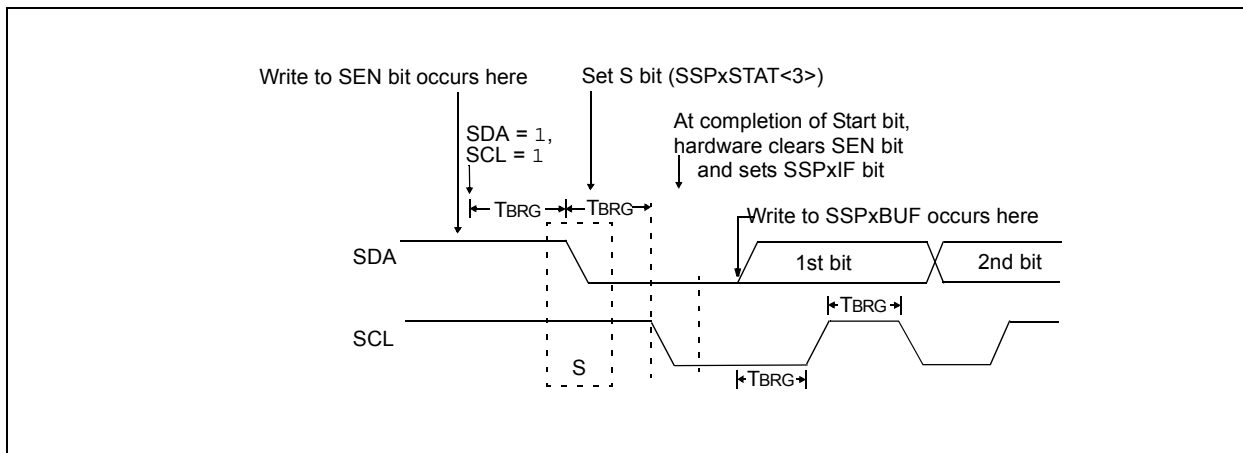
To initiate a Start condition (Figure 32-26), the user sets the Start Enable bit, SEN bit of the SSPxCON2 register. If the SDA and SCL pins are sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<7:0> and starts its count. If SCL and SDA are both sampled high when the Baud 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 I<sup>2</sup>C module is reset into its Idle state.

**2:** The Philips I<sup>2</sup>C specification states that a bus collision cannot occur on a Start.

**FIGURE 32-26: FIRST START BIT TIMING**





## 32.6.5 I<sup>2</sup>C 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 automati-

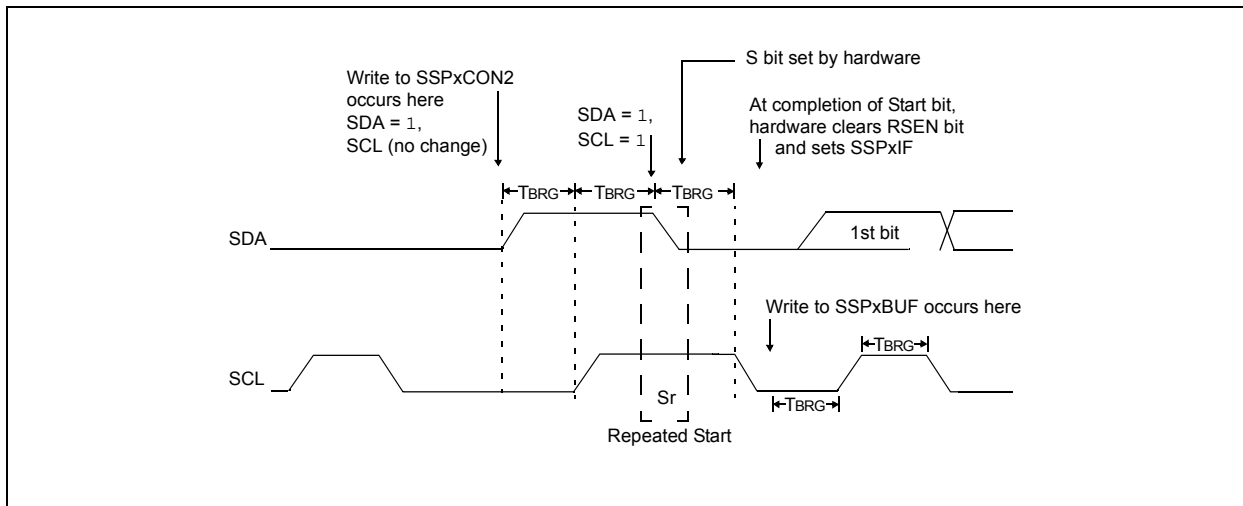
cally cleared and the Baud Rate Generator will not be reloaded, leaving the SDA pin held low. As soon as a Start condition is detected on the SDA and SCL pins, the S bit of the SSPxSTAT register will be set. The SSPxIF bit will not be set until the Baud Rate Generator has timed out.

**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 I<sup>2</sup>C 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 ACK bit during the ninth bit time if an address match occurred, or if data was received properly. The status of  $\overline{\text{ACK}}$  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/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 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., SSPxSR is still shifting out a data byte), the WCOL bit is set and the contents of the buffer are unchanged (the write does not occur).

WCOL must be cleared by software before the next transmission.

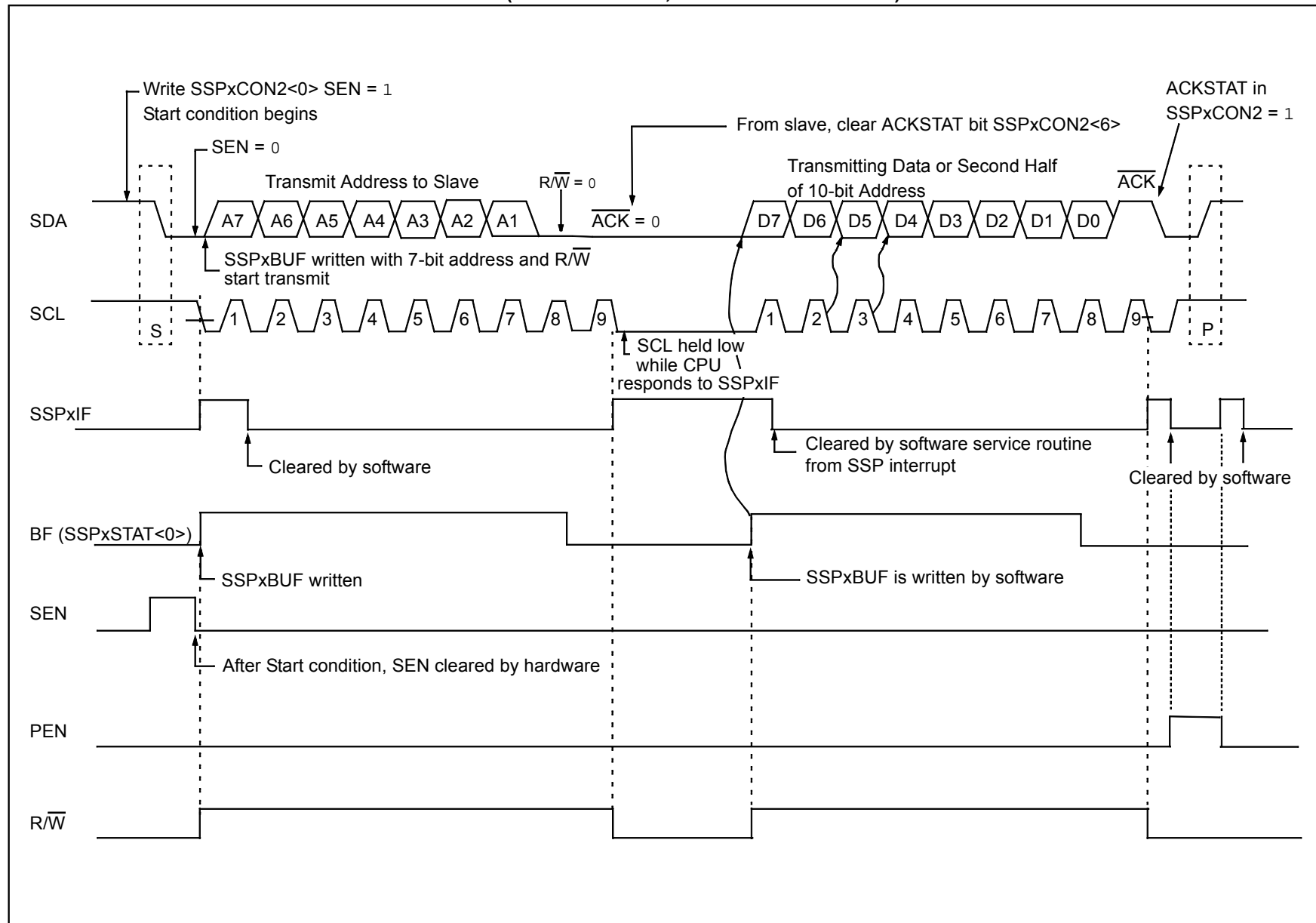
### 32.6.6.3 ACKSTAT Status Flag

In Transmit mode, the ACKSTAT bit of the SSPxCON2 register is cleared when the slave has sent an Acknowledge ( $\text{ACK} = 0$ ) and is set when the slave does not Acknowledge ( $\text{ACK} = 1$ ). 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  $\overline{\text{ACK}}$  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  $\overline{\text{ACK}}$  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<sup>2</sup>C MASTER MODE WAVEFORM (TRANSMISSION, 7 OR 10-BIT ADDRESS)**



## 32.6.7 I<sup>2</sup>C 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 SSPxSR. After the falling edge of the eighth clock, the receive enable flag is automatically cleared, the contents of the SSPxSR are loaded into the SSPxBUF, the BF flag bit is set, the SSPxIF flag bit is set and the Baud Rate Generator is suspended from counting, holding 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 SSPxSR. 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 SSPxSR and the BF flag bit is already set from a previous reception.

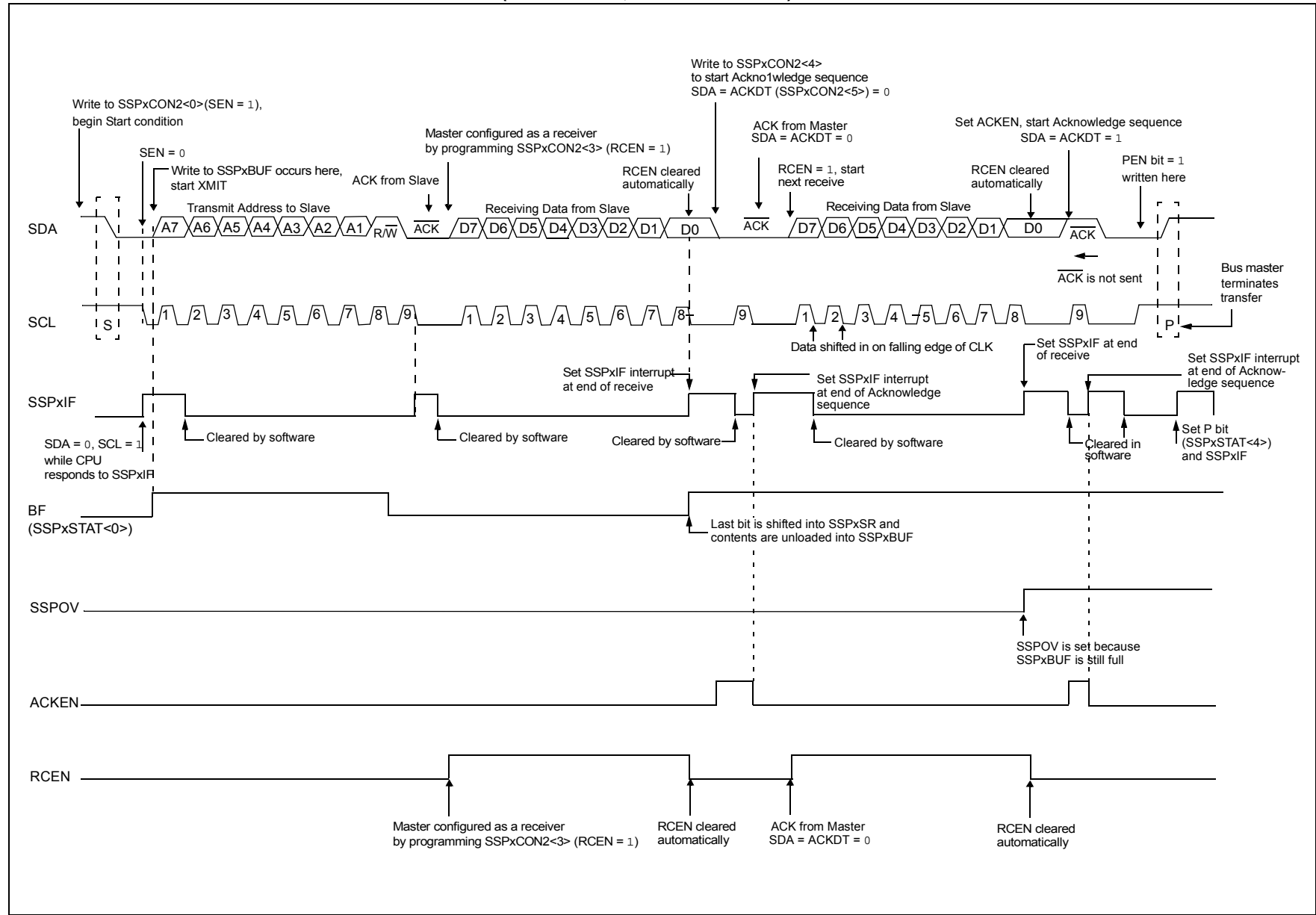
### 32.6.7.3 WCOL Status Flag

If the user writes the SSPxBUF when a receive is already in progress (i.e., SSPxSR is still shifting in a data byte), the WCOL bit is set and the contents of the buffer are unchanged (the write does not occur).

### 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/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 ACK by setting the ACKEN bit.
12. Master's 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<sup>2</sup>C 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 WCOL bit is set and the contents of the buffer are unchanged (the write does not occur).

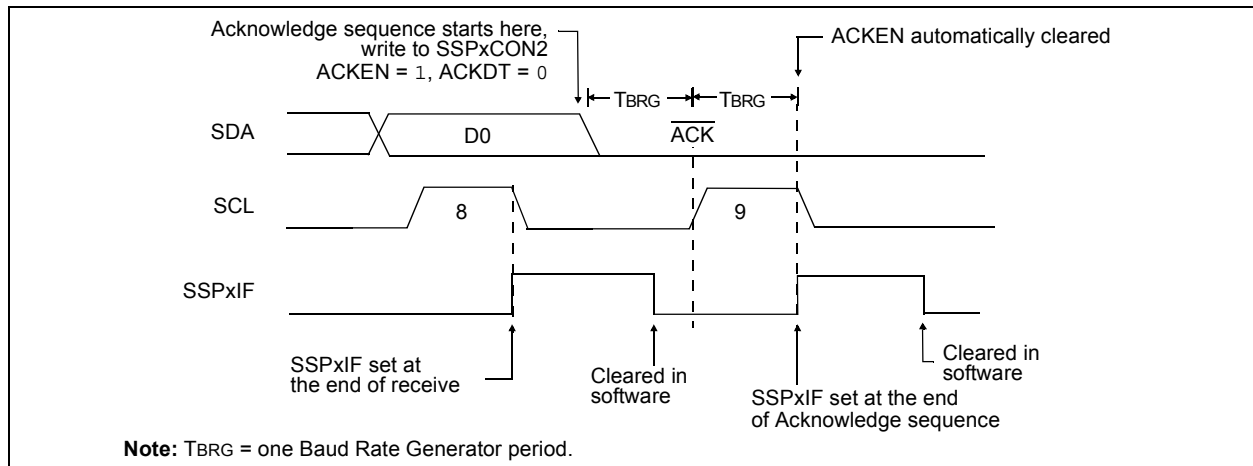
## 32.6.9 STOP CONDITION TIMING

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).

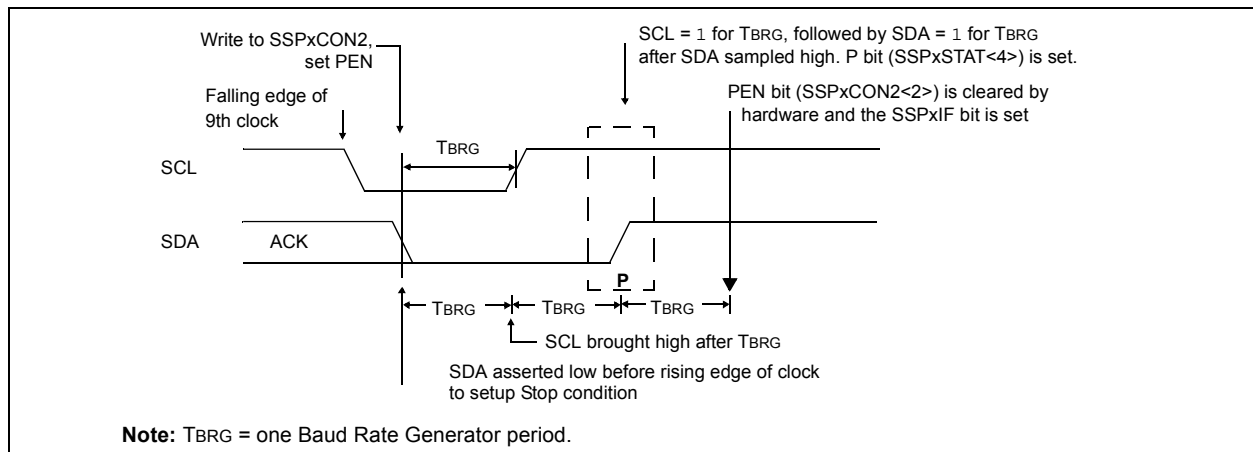
### 32.6.9.1 WCOL Status Flag

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**



**FIGURE 32-31: STOP CONDITION RECEIVE OR TRANSMIT MODE**



## 32.6.10 SLEEP OPERATION

While in Sleep mode, the I<sup>2</sup>C 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<sup>2</sup>C bus may be taken when the P bit of the SSPxSTAT register is set, or the bus is Idle, with both the S and P bits clear. When the bus is busy, enabling the 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 BCL1IF 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, BCL1IF and reset the I<sup>2</sup>C 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<sup>2</sup>C bus is free, the user can resume communication by asserting a Start condition.

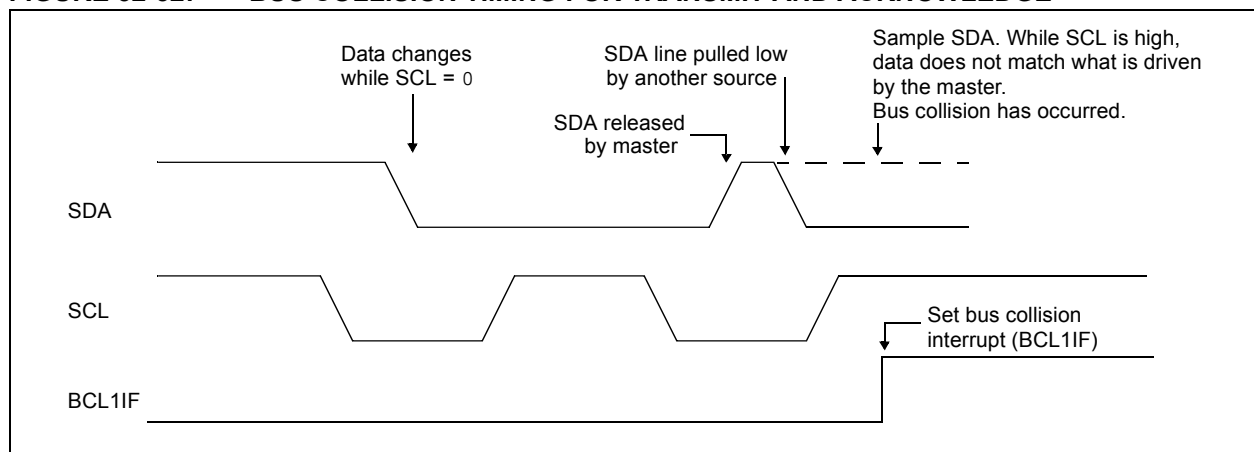
If a Start, Repeated Start, Stop or Acknowledge condition was in progress when the bus collision occurred, the condition is aborted, the SDA and SCL lines are deasserted and the respective control bits in the SSPxCON2 register are cleared. When the user services the bus collision Interrupt Service Routine and if the I<sup>2</sup>C 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<sup>2</sup>C bus can be taken when the P bit is set in the SSPxSTAT register, or the bus is Idle and the S and P bits are cleared.

**FIGURE 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:

- SDA or SCL are sampled low at the beginning of the Start condition (Figure 32-33).
- 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:

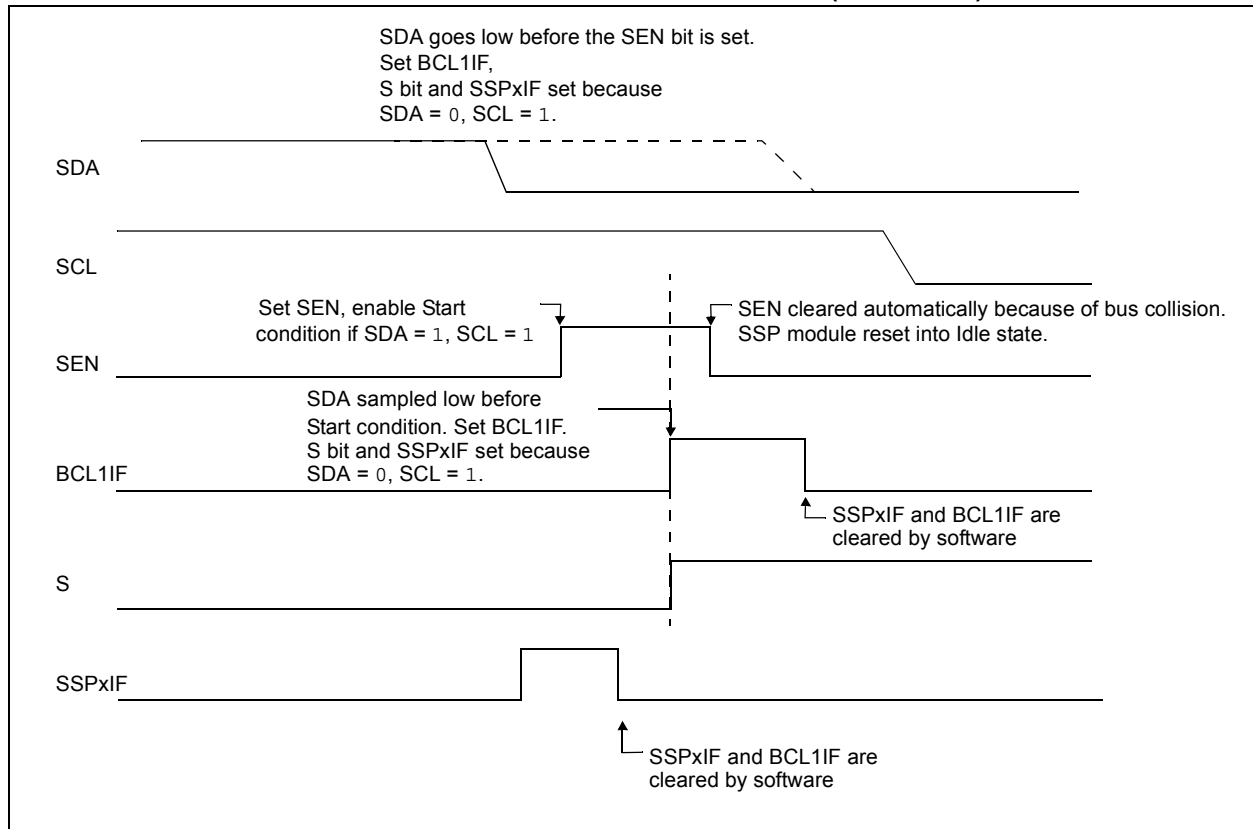
- the Start condition is aborted,
- the BCL1IF flag is set and
- the MSSP module is reset to its Idle state (Figure 32-33).

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.

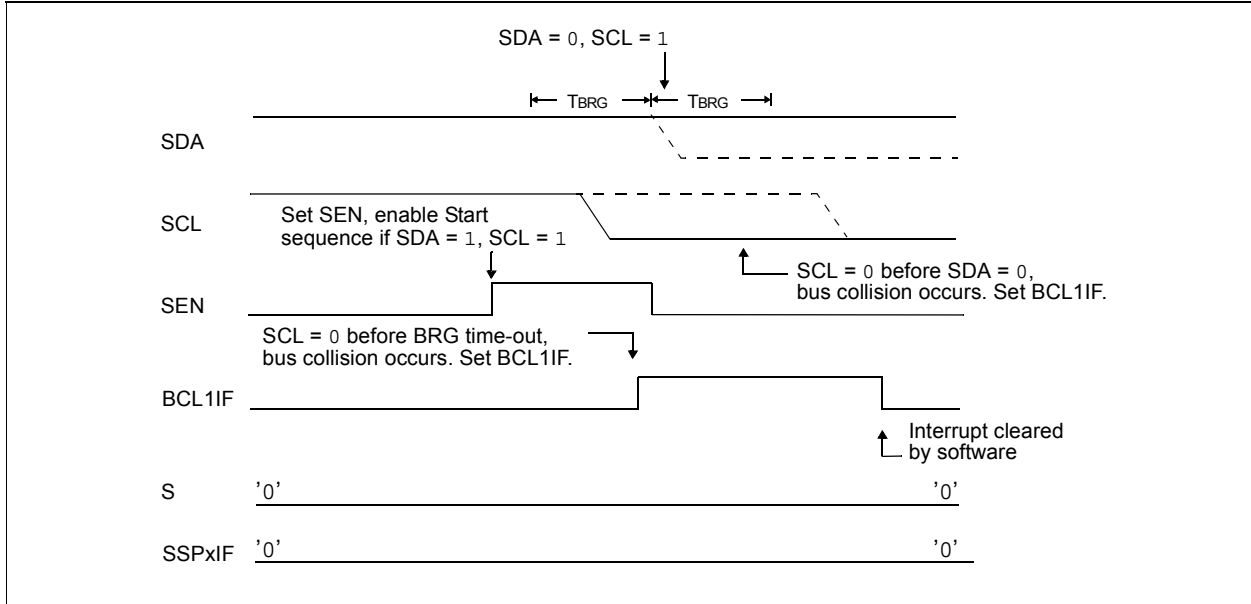
**Note:** The reason that bus collision is not a factor during a Start condition is that no two bus masters can assert a Start condition at the exact same time. Therefore, one master will always assert SDA before the other. This condition does not cause a bus collision because the two masters must be 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)**

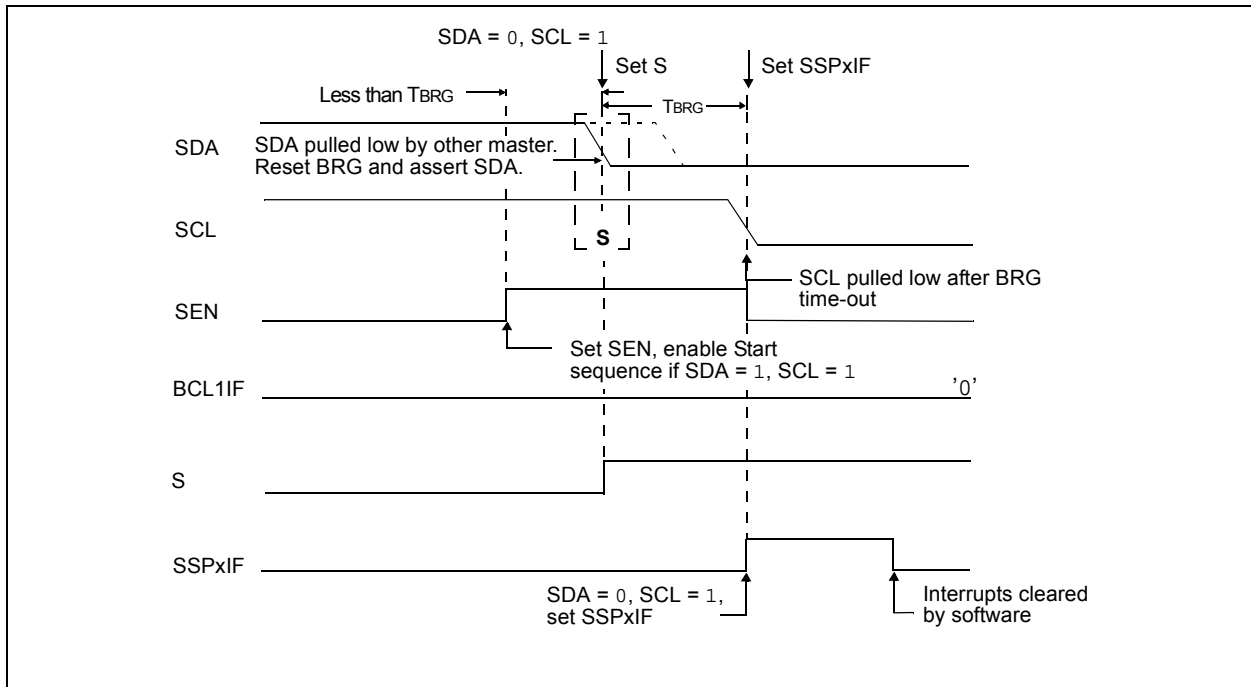




**FIGURE 32-34: BUS COLLISION DURING START CONDITION (SCL = 0)**



**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 low level is sampled on SDA when SCL goes from low level to high level (Case 1).
- 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.

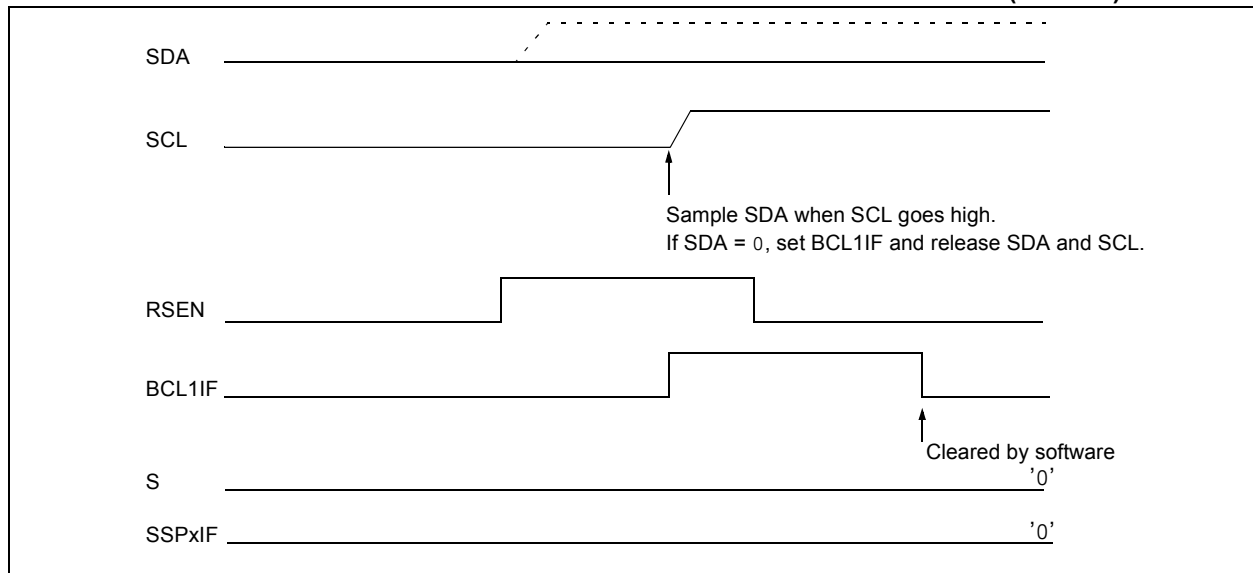
If SDA is low, a bus collision has occurred (i.e., another master is attempting to transmit a data '0', [Figure 32-36](#)). If SDA is sampled high, the BRG is reloaded and begins

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.

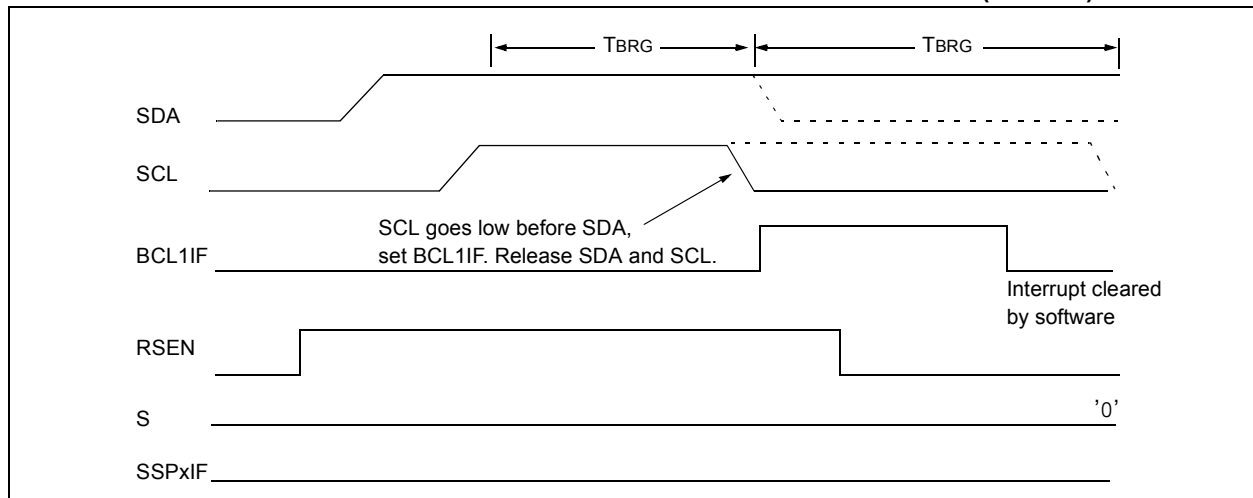
If SCL goes from high-to-low before the BRG times out and SDA has not already been asserted, a bus collision occurs. In this case, another master is attempting to transmit a data '1' during the Repeated Start condition, see [Figure 32-37](#).

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)**



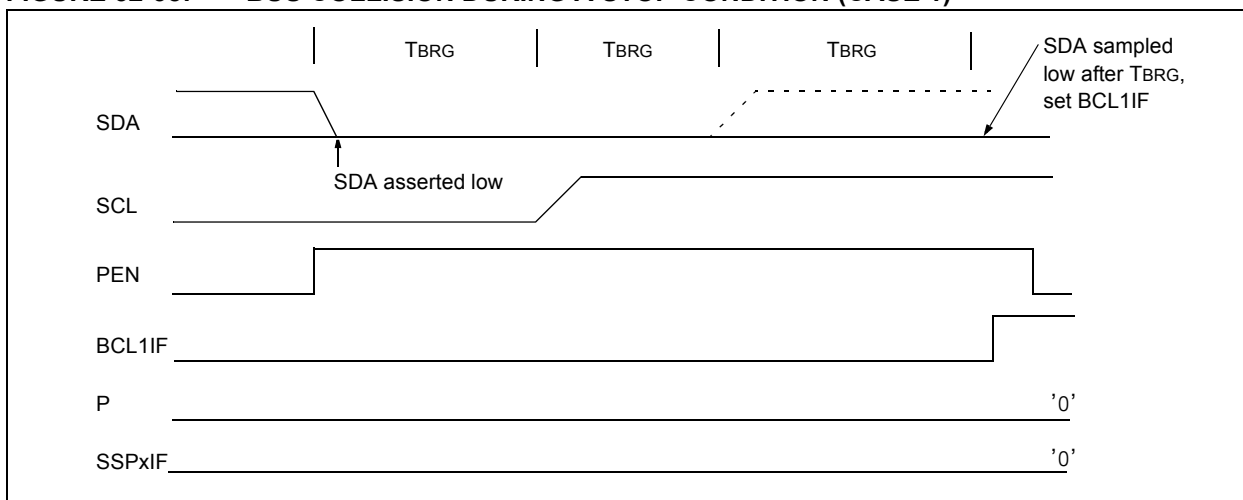
### 32.6.13.3 Bus Collision During a Stop Condition

Bus collision occurs during a Stop condition if:

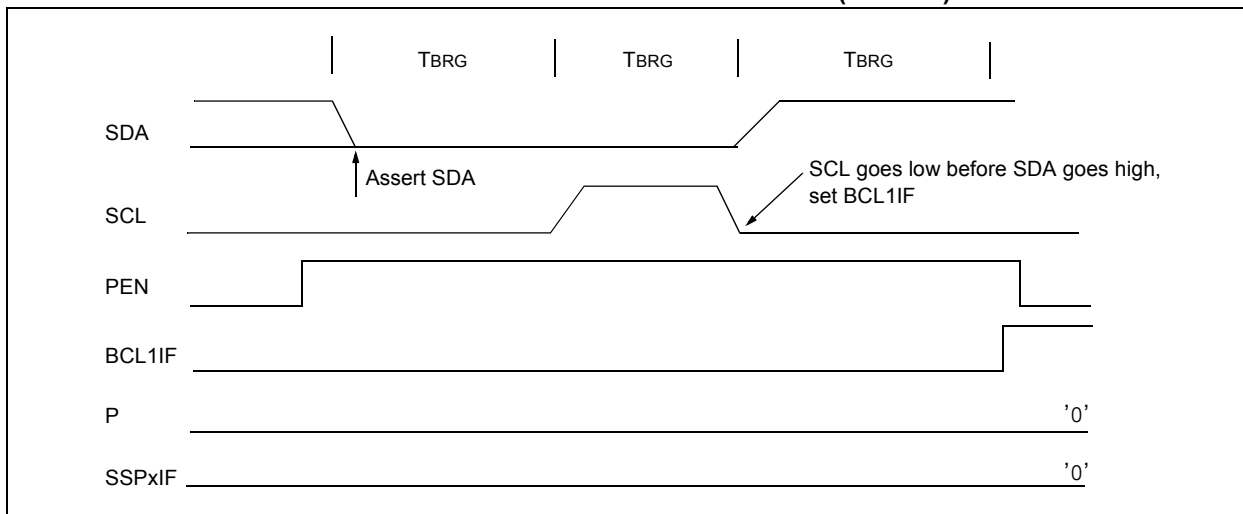
- After the SDA pin has been deasserted and allowed to float high, SDA is sampled low after the BRG has timed out (Case 1).
- After the SCL pin is deasserted, SCL is sampled low before SDA goes high (Case 2).

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)**



## 32.7 BAUD RATE GENERATOR

The MSSP module has a Baud Rate Generator available for clock generation in both I<sup>2</sup>C 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

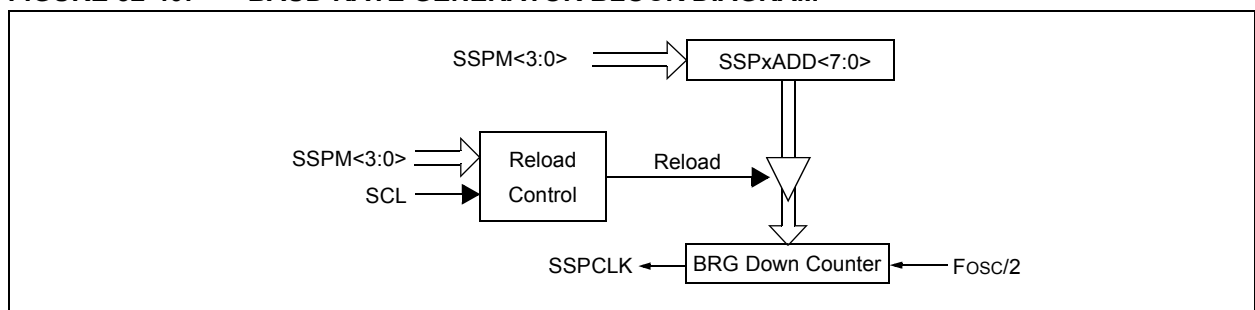
module clock line. The logic dictating when the reload signal is asserted depends on the mode the MSSP is being operated in.

Table 32-4 demonstrates clock rates based on instruction cycles and the BRG value loaded into SSPxADD.

**EQUATION 32-1:**

$$F_{CLOCK} = \frac{F_{OSC}}{(SSP1ADD + 1)(4)}$$

**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<sup>2</sup>C. This is an implementation limitation.

**TABLE 32-2: MSSP CLOCK RATE W/BRG**

Fosc	Fcy	BRG Value	F <sub>CLOCK</sub> (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 37-4 to ensure the system is designed to support IOL requirements.

## 32.8 Register Definitions: MSSPx Control

### REGISTER 32-1: SSPxSTAT: SSPx STATUS REGISTER

R/W-0/0	R/W-0/0	R/HS/HC-0	R/HS/HC-0	R/HS/HC-0	R/HS/HC-0	R/HS/HC-0	R/HS/HC-0
SMP	CKE <sup>(1)</sup>	D/A	P <sup>(2)</sup>	S <sup>(2)</sup>	R/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	HS/HC = Hardware set/clear

bit 7	<p><b>SMP:</b> SPI Data Input Sample bit</p> <p><u>SPI Master mode:</u>            1 = Input data sampled at end of data output time            0 = Input data sampled at middle of data output time</p> <p><u>SPI Slave mode:</u>            SMP must be cleared when SPI is used in Slave mode</p> <p><u>In I<sup>2</sup>C Master or Slave mode:</u>            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)</p>
bit 6	<p><b>CKE:</b> SPI Clock Edge Select bit (SPI mode only)<sup>(1)</sup></p> <p><u>In SPI Master or Slave mode:</u>            1 = Transmit occurs on transition from active to Idle clock state            0 = Transmit occurs on transition from Idle to active clock state</p> <p><u>In I<sup>2</sup>C mode only:</u>            1 = Enable input logic so that thresholds are compliant with SMBus specification            0 = Disable SMBus specific inputs</p>
bit 5	<p><b>D/A:</b> Data/Address bit (I<sup>2</sup>C mode only)</p> <p>1 = Indicates that the last byte received or transmitted was data            0 = Indicates that the last byte received or transmitted was address</p>
bit 4	<p><b>P:</b> Stop bit<sup>(2)</sup></p> <p>(I<sup>2</sup>C 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</p>
bit 3	<p><b>S:</b> Start bit <sup>(2)</sup></p> <p>(I<sup>2</sup>C 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</p>
bit 2	<p><b>R/W:</b> Read/Write bit information (I<sup>2</sup>C mode only)</p> <p>This bit holds the R/W bit information following the last address match. This bit is only valid from the address match to the next Start bit, Stop bit, or not ACK bit.</p> <p><u>In I<sup>2</sup>C Slave mode:</u>            1 = Read            0 = Write</p> <p><u>In I<sup>2</sup>C Master mode:</u>            1 = Transmit is in progress            0 = Transmit is not in progress</p> <p>OR-ing this bit with SEN, RSEN, PEN, RCEN or ACKEN will indicate if the MSSP is in IDLE mode.</p>
bit 1	<p><b>UA:</b> Update Address bit (10-bit I<sup>2</sup>C mode only)</p> <p>1 = Indicates that the user needs to update the address in the SSPxADD register            0 = Address does not need to be updated</p>
bit 0	<p><b>BF:</b> Buffer Full Status bit</p> <p><u>Receive (SPI and I<sup>2</sup>C modes):</u>            1 = Receive complete, SSPxBUF is full            0 = Receive not complete, SSPxBUF is empty</p> <p><u>Transmit (I<sup>2</sup>C mode only):</u>            1 = Data transmit in progress (does not include the ACK and Stop bits), SSPxBUF is full            0 = Data transmit complete (does not include the ACK and Stop bits), SSPxBUF is empty</p>

- Note** 1: Polarity of clock state is set by the CKP bit of the SSPxCON register.  
 2: This bit is cleared on Reset and when SSPEN is cleared.

## REGISTER 32-2: SSPxCON1: SSPx CONTROL REGISTER 1

R/C/HS-0/0	R/C/HS-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
WCOL	SSPOV <sup>(1)</sup>	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 (Transmit mode only)  
 1 = The SSPxBUF register is written while it is still transmitting the previous word (must be cleared in software)  
 0 = No collision
- bit 6      **SSPOV:** Receive Overflow Indicator bit<sup>(1)</sup>  
In SPI mode:  
 1 = A new byte is received while the SSPxBUF register is still holding the previous data. In case of overflow, the data in SSPxSR is lost. Overflow can only occur in Slave mode. In Slave mode, the user must read the SSPxBUF, even if only transmitting data, to avoid setting overflow. In Master mode, the overflow bit is not set since each new reception (and transmission) is initiated by writing to the SSPxBUF register (must be cleared in software).  
 0 = No overflow  
In I<sup>2</sup>C mode:  
 1 = A byte is received while the SSPxBUF register is still holding the previous byte. SSPOV is a "don't care" in Transmit mode (must be cleared in software).  
 0 = No overflow
- bit 5      **SSPEN:** Synchronous Serial Port Enable bit  
 In both modes, when enabled, the following 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<sup>(2)</sup>  
 0 = Disables serial port and configures these pins as I/O port pins  
In I<sup>2</sup>C mode:  
 1 = Enables the serial port and configures the SDA and SCL pins as the source of the serial port pins<sup>(3)</sup>  
 0 = Disables serial port and configures these pins as I/O port pins
- bit 4      **CKP:** Clock Polarity Select bit  
In SPI mode:  
 1 = Idle state for clock is a high level  
 0 = Idle state for clock is a low level  
In I<sup>2</sup>C Slave mode:  
 SCL release control  
 1 = Enable clock  
 0 = Holds clock low (clock stretch). (Used to ensure data setup time.)  
In I<sup>2</sup>C Master mode:  
 Unused in this mode
- bit 3-0      **SSPM<3:0>:** Synchronous Serial Port Mode Select bits  
 1111 = I<sup>2</sup>C Slave mode, 10-bit address with Start and Stop bit interrupts enabled  
 1110 = I<sup>2</sup>C Slave mode, 7-bit address with Start and Stop bit interrupts enabled  
 1101 = Reserved  
 1100 = Reserved  
 1011 = I<sup>2</sup>C firmware controlled Master mode (slave idle)  
 1010 = SPI Master mode, clock = Fosc/(4 \* (SSPxADD+1))<sup>(5)</sup>  
 1001 = Reserved  
 1000 = I<sup>2</sup>C Master mode, clock = Fosc / (4 \* (SSPxADD+1))<sup>(4)</sup>  
 0111 = I<sup>2</sup>C Slave mode, 10-bit address  
 0110 = I<sup>2</sup>C 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 SSPxBUF register.
  - 2: When enabled, these pins must be properly configured as input or output. Use SSPxSSPPS, SSPxCLKPPS, SSPxDATPPS, and RxyPPS to select the pins.
  - 3: When enabled, the SDA and SCL pins must be configured as inputs. Use SSPxCLKPPS, SSPxDATPPS, and RxyPPS to select the pins.
  - 4: SSPxADD values of 0, 1 or 2 are not supported for I<sup>2</sup>C mode.
  - 5: SSPxADD value of '0' is not supported. Use SSPM = 0000 instead.

## REGISTER 32-3: SSPxCON2: SSPx CONTROL REGISTER 2 (I<sup>2</sup>C MODE ONLY)<sup>(1)</sup>

R/W-0/0	R/HS/HC-0	R/W-0/0	R/S/HC-0/0	R/S/HC-0/0	R/S/HC-0/0	R/S/HC-0/0	R/S/HC-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<sup>2</sup>C Slave mode only)  
1 = Enable interrupt when a general call address (0x00 or 00h) is received in the SSPxSR  
0 = General call address disabled
- bit 6      **ACKSTAT:** Acknowledge Status bit (in I<sup>2</sup>C mode only)  
1 = Acknowledge was not received  
0 = Acknowledge was received
- bit 5      **ACKDT:** Acknowledge Data bit (in I<sup>2</sup>C 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<sup>2</sup>C 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<sup>2</sup>C Master mode only)  
1 = Enables Receive mode for I<sup>2</sup>C  
0 = Receive idle
- bit 2      **PEN:** Stop Condition Enable bit (in I<sup>2</sup>C 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<sup>2</sup>C 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<sup>2</sup>C module is not in the IDLE mode, this bit may not be set (no spooling) and the SSPxBUF may not be written (or writes to the SSPxBUF are disabled).

## REGISTER 32-4: SSPxCON3: SSPx CONTROL REGISTER 3

R-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
ACKTIM <sup>(3)</sup>	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<sup>2</sup>C mode only)<sup>(3)</sup>  
 1 = Indicates the I<sup>2</sup>C bus is in an Acknowledge sequence, set on 8<sup>th</sup> falling edge of SCL clock  
 0 = Not an Acknowledge sequence, cleared on 9<sup>th</sup> rising edge of SCL clock
- bit 6 **PCIE:** Stop Condition Interrupt Enable bit (I<sup>2</sup>C mode only)  
 1 = Enable interrupt on detection of Stop condition  
 0 = Stop detection interrupts are disabled<sup>(2)</sup>
- bit 5 **SCIE:** Start Condition Interrupt Enable bit (I<sup>2</sup>C mode only)  
 1 = Enable interrupt on detection of Start or Restart conditions  
 0 = Start detection interrupts are disabled<sup>(2)</sup>
- bit 4 **BOEN:** Buffer Overwrite Enable bit  
In SPI Slave mode:<sup>(1)</sup>  
 1 = SSPxBUF updates every time that a new data byte is shifted in ignoring the BF bit  
 0 = If new byte is received with BF bit of the SSPxSTAT register already set, SSPOV bit of the SSPxCON1 register is set, and the buffer is not updated  
In I<sup>2</sup>C Master mode and SPI Master mode:  
 This bit is ignored.  
In I<sup>2</sup>C Slave mode:  
 1 = SSPxBUF is updated and ACK is generated for a received address/data byte, ignoring the state of the SSPOV bit only if the BF bit = 0.  
 0 = SSPxBUF is only updated when SSPOV is clear
- bit 3 **SDAHT:** SDA Hold Time Selection bit (I<sup>2</sup>C 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<sup>2</sup>C 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 PIR3 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<sup>2</sup>C Slave mode only)  
 1 = Following the eighth falling edge of SCL for a matching received address byte; CKP bit of the SSPxCON1 register will be cleared and the SCL will be held low.  
 0 = Address holding is disabled
- bit 0 **DHEN:** Data Hold Enable bit (I<sup>2</sup>C Slave mode only)  
 1 = Following the eighth falling edge of SCL for a received data byte; slave hardware clears the CKP bit of the SSPxCON1 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 SSPxBUF.
- 2:** This bit has no effect in Slave modes that Start and Stop condition detection is explicitly listed as enabled.
- 3:** The ACKTIM Status bit is only active when the AHEN bit or DHEN bit is set.



## REGISTER 32-5: SSPxMSK: SSPx MASK REGISTER

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
SSPxMSK<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      **SSPxMSK<7:1>**: Mask bits  
 1 = The received address bit n is compared to SSPxADD<n> to detect I<sup>2</sup>C address match  
 0 = The received address bit n is not used to detect I<sup>2</sup>C address match
- bit 0      **SSPxMSK<0>**: Mask bit for I<sup>2</sup>C Slave mode, 10-bit Address  
I<sup>2</sup>C Slave mode, 10-bit address (SSPM<3:0> = 0111 or 1111):  
 1 = The received address bit 0 is compared to SSPxADD<0> to detect I<sup>2</sup>C address match  
 0 = The received address bit 0 is not used to detect I<sup>2</sup>C address match  
I<sup>2</sup>C Slave mode, 7-bit address:  
 MSK0 bit is ignored.

## REGISTER 32-6: SSPxADD: MSSPx ADDRESS AND BAUD RATE REGISTER (I<sup>2</sup>C 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
SSPxADD<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      **SSPxADD<7:0>**: Baud Rate Clock Divider bits  
 $SCL \text{ pin clock period} = ((ADD<7:0> + 1) * 4) / F_{osc}$

### 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<sup>2</sup>C 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      **SSPxADD<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      **SSPxADD<7:0>**: Eight Least Significant bits of 10-bit address

### 7-Bit Slave mode:

- bit 7-1      **SSPxADD<7:1>**: 7-bit address
- bit 0      **Not used:** Unused in this mode. Bit state is a “don't care”.

## REGISTER 32-7: SSPxBUF: MSSPx BUFFER REGISTER

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SSPxBUF<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      **SSPxBUF<7:0>**: MSSP Buffer bits

**TABLE 32-3: SUMMARY OF REGISTERS ASSOCIATED WITH MSSPx**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
INTCON	GIE	PEIE	—	—	—	—	—	INTEDG	121
PIR3	RC2IF	TX2IF	RC1IF	TX1IF	BCL2IF	SSP2IF	BCL1IF	SSP1IF	141
PIE3	RC2IE	TX2IE	RC1IE	TX1IE	BCL2IE	SSP2IE	BCL1IE	SSP1IE	133
SSP1STAT	SMP	CKE	D $\bar{A}$	P	S	R $\bar{W}$	UA	BF	417
SSP1CON1	WCOL	SSPOV	SSPEN	CKP	SSPM<3:0>				418
SSP1CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	419
SSP1CON3	ACKTIM	PCIE	SCIE	BOEN	SDAHT	SBCDE	AHEN	DHEN	417
SSP1MSK	SSPMSK<7:0>								421
SSP1ADD	SSPADD<7:0>								421
SSP1BUF	SSPBUF<7:0>								422
SSP2STAT	SMP	CKE	D $\bar{A}$	P	S	R $\bar{W}$	UA	BF	417
SSP2CON1	WCOL	SSPOV	SSPEN	CKP	SSPM<3:0>				418
SSP2CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	419
SSP2CON3	ACKTIM	PCIE	SCIE	BOEN	SDAHT	SBCDE	AHEN	DHEN	417
SSP2MSK	SSPMSK<7:0>								421
SSP2ADD	SSPADD<7:0>								421
SSP2BUF	SSPBUF<7:0>								422
SSP1CLKPPS	—	—	SSP1CLKPPS<5:0>						199
SSP1DATPPS	—	—	SSP1DATPPS<5:0>						199
SSP1SSPPS	—	—	SSP1SSPPS<5:0>						199
SSP2CLKPPS	—	—	SSP2CLKPPS<5:0>						199
SSP2DATPPS	—	—	SSP2DATPPS<5:0>						199
SSP2SSPPS	—	—	SSP2SSPPS<5:0>						199
RxyPPS	—	—	RxyPPS<4:0>						200

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

**Note 1:** When using designated I<sup>2</sup>C pins, the associated pin values in INLVLx will be ignored.

## 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.

**Note:** Two identical EUSART modules are implemented on this device, EUSART1 and EUSART2. All references to EUSART1 apply to EUSART2 as well.

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:

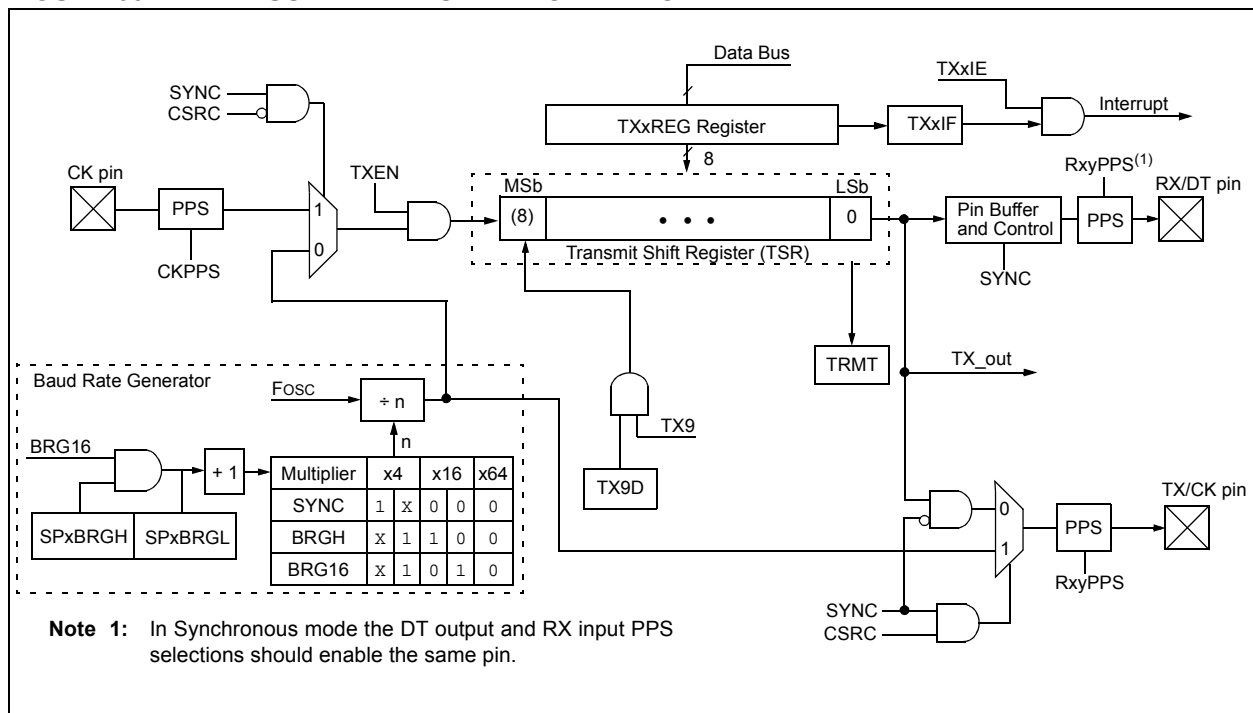
- Automatic detection and calibration of the baud rate
- Wake-up on Break reception
- 13-bit Break character transmit

Block diagrams of the EUSART transmitter and receiver are shown in [Figure 33-1](#) and [Figure 33-2](#).

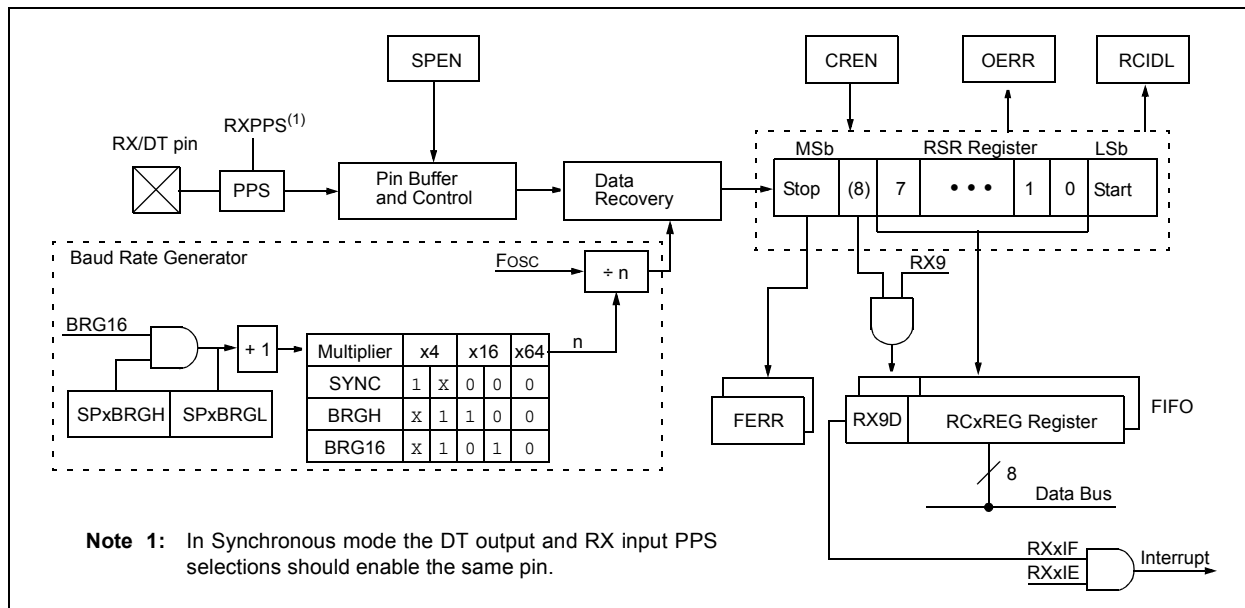
The EUSART transmit output (TX\_out) is available to the TX/CK pin and internally to the following peripherals:

- Configurable Logic Cell (CLC)

**FIGURE 33-1: EUSART TRANSMIT BLOCK DIAGRAM**



**FIGURE 33-2: EUSART RECEIVE BLOCK DIAGRAM**



The operation of the EUSART module is controlled through three registers:

- Transmit Status and Control (TXxSTA)
- Receive Status and Control (RCxSTA)
- Baud Rate Control (BAUDxCON)

These registers are detailed in [Register 33-1](#), [Register 33-2](#) and [Register 33-3](#), respectively.

The RX input pin is selected with the RXxPPS. The CK input is selected with the TXxPPS register. 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 V<sub>OH</sub> Mark state which represents a '1' data bit, and a V<sub>OL</sub> 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 33-3](#) 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 TXxIF 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 T<sub>cy</sub> 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 33.4.1.2 "Clock Polarity"](#).

#### 33.1.1.4 Transmit Interrupt Flag

The TXxIF interrupt flag bit of the PIR3 register is set whenever the EUSART transmitter is enabled and no character is being held for transmission in the TXxREG. In other words, the TXxIF bit is only clear when the TSR is busy with a character and a new character has been queued for transmission in the TXxREG. The TXxIF flag bit is not cleared immediately upon writing TXxREG. TXxIF becomes valid in the second instruction cycle following the write execution. Polling TXxIF immediately following the TXxREG write will return invalid results. The TXxIF bit is read-only, it cannot be set or cleared by software.

The TXxIF interrupt can be enabled by setting the TXxIE interrupt enable bit of the PIE3 register. However, the TXxIF flag bit will be set whenever the TXxREG is empty, regardless of the state of TXxIE enable bit.

To use interrupts when transmitting data, set the TXxIE bit only when there is more data to send. Clear the TXxIE interrupt enable bit upon writing the last character of the transmission to the TXxREG.

## 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

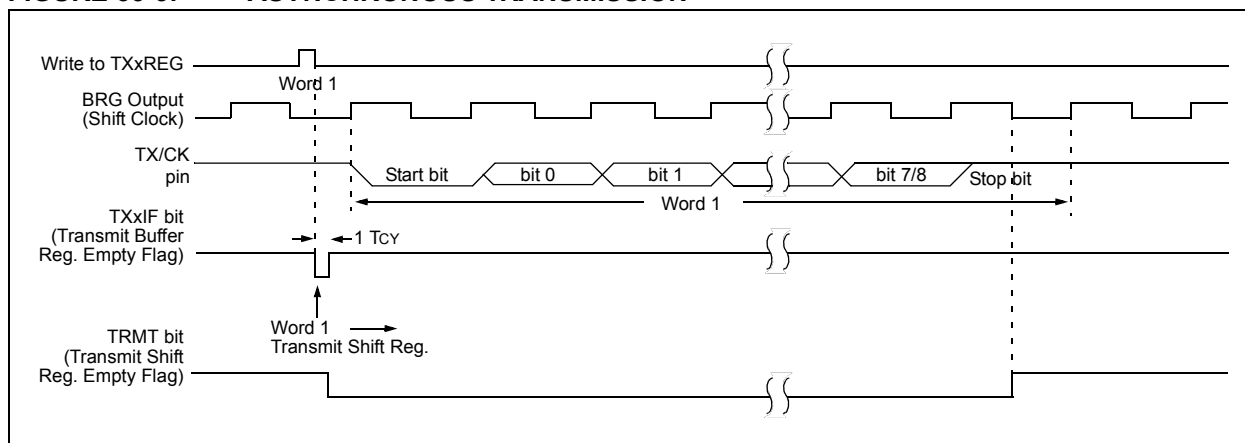
The EUSART supports 9-bit character transmissions. When the TX9 bit of the TXxSTA register is set, the EUSART will shift nine bits out for each character transmitted. The TX9D bit of the TXxSTA register is the ninth, and Most Significant data bit. When transmitting 9-bit data, the TX9D data bit must 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.

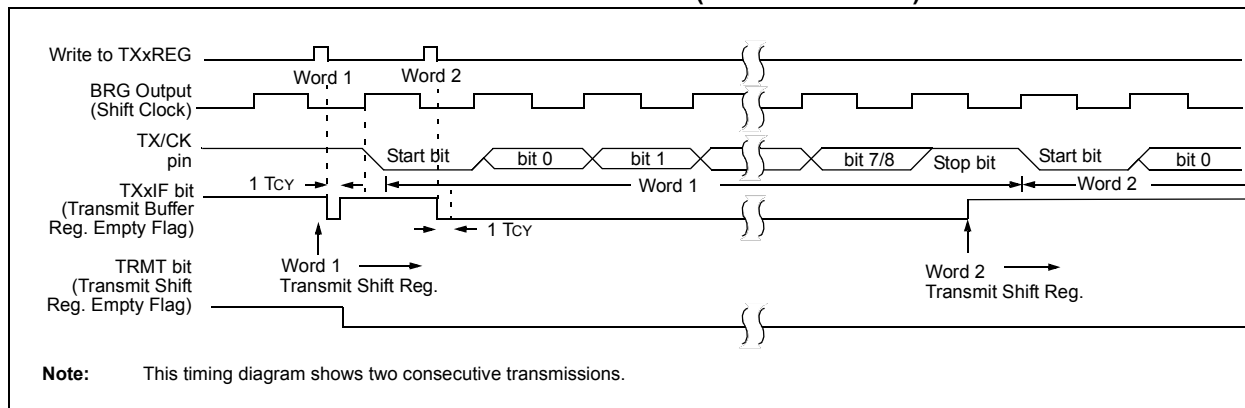
## 33.1.1.7 Asynchronous Transmission Set-up:

1. Initialize the SPxBRGH, SPxBRGL register pair and the BRGH and BRG16 bits to achieve the desired baud rate (see [Section 33.3 “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 SCKP bit if inverted transmit is desired.
5. Enable the transmission by setting the TXEN control bit. This will cause the TXxIF interrupt bit to be set.
6. If interrupts are desired, set the TXxIE interrupt enable bit of the PIE3 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)**



### 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 RXxIF interrupt flag bit of the PIR3 register is set. The top character in the FIFO is transferred out of the FIFO by reading the RCxREG register.

**Note:** If the receive FIFO is overrun, no additional characters will be received until the overrun condition is cleared. See Section 33.1.2.5 "Receive Overrun Error" for more information on overrun errors.

## 33.1.2.3 Receive Interrupts

The RXxIF interrupt flag bit of the PIR3 register is set whenever the EUSART receiver is enabled and there is an unread character in the receive FIFO. The RXxIF interrupt flag bit is read-only, it cannot be set or cleared by software.

RXxIF interrupts are enabled by setting all of the following bits:

- RXxIE, Interrupt Enable bit of the PIE3 register
- PEIE, Peripheral Interrupt Enable bit of the INTCON register
- GIE, Global Interrupt Enable bit of the INTCON register

The RXxIF 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.

<b>Note:</b> 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 RXxIF 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 Setup:

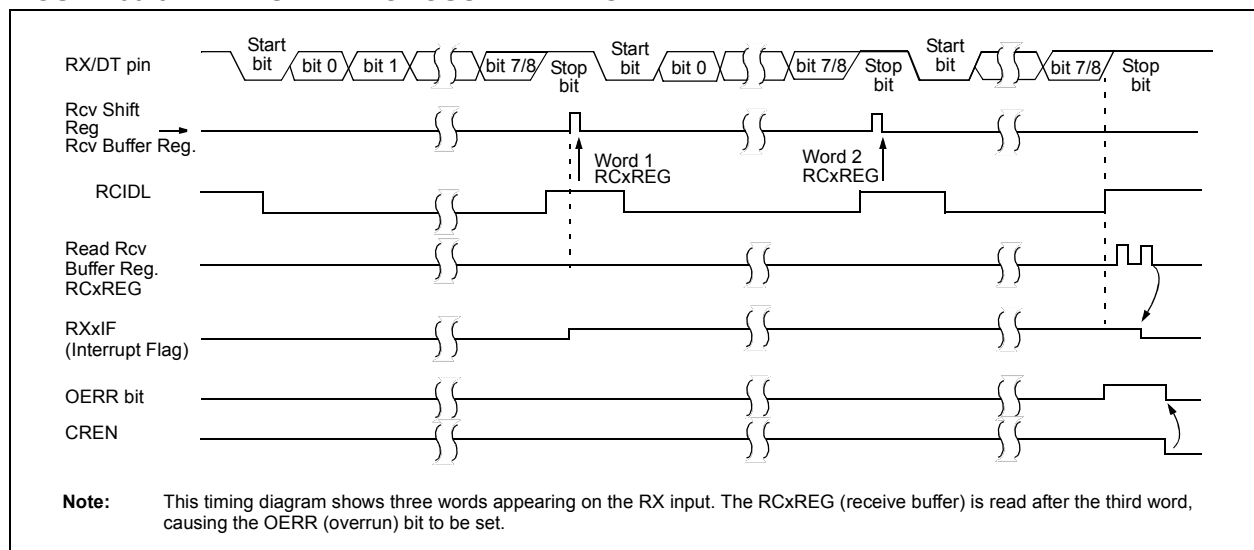
1. Initialize the SPxBRGH, SPxBRGL register pair and the BRGH and BRG16 bits to achieve the desired baud rate (see [Section 33.3 “EUSART Baud Rate Generator \(BRG\)”](#)).
2. Clear the ANSEL bit for the RX pin (if applicable).
3. Enable the serial port by setting the SPEN bit. The SYNC bit must be clear for asynchronous operation.
4. If interrupts are desired, set the RXxIE bit of the PIE3 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 RXxIF 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 RXxIE 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 Setup

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 33.3 “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 RXxIE bit of the PIE3 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 RXxIF 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 RXxIE 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**



## 33.2 Clock Accuracy with Asynchronous Operation

The factory calibrates the internal oscillator block output (INTOSC). However, the HFINTOSC frequency may drift as  $V_{DD}$  or temperature changes, and this directly affects the asynchronous baud rate. Two methods may be used to adjust the baud rate clock, but both require a reference clock source of some kind.

The first (preferred) method uses the OSCTUNE register to adjust the HFINTOSC output. Adjusting the value in the OSCTUNE register allows for fine resolution changes to the system clock source. See [Section 9.2.2.2 “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 33.3.1 “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 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 33-1 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 33-3. 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:

$$\text{Desired Baud Rate} = \frac{F_{OSC}}{64(SPBRGH:SPBRGL) + 1}$$

Solving for SPxBRGH:SPxBRGL:

$$X = \frac{F_{OSC}}{\text{Desired Baud Rate} \cdot 64} - 1$$

$$= \frac{16000000}{9600 \cdot 64} - 1$$

$$= [25.042] = 25$$

$$\text{Calculated Baud Rate} = \frac{16000000}{64(25 + 1)}$$

$$= 9615$$

$$\text{Error} = \frac{\text{Calc. Baud Rate} - \text{Desired Baud Rate}}{\text{Desired Baud Rate}}$$

$$= \frac{(9615 - 9600)}{9600} = 0.16\%$$

## 33.3.1 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 RXxIF interrupt flag is set. The value in the RCxREG needs to be read to clear the RXxIF 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 33-1. 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 33.3.3 “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.

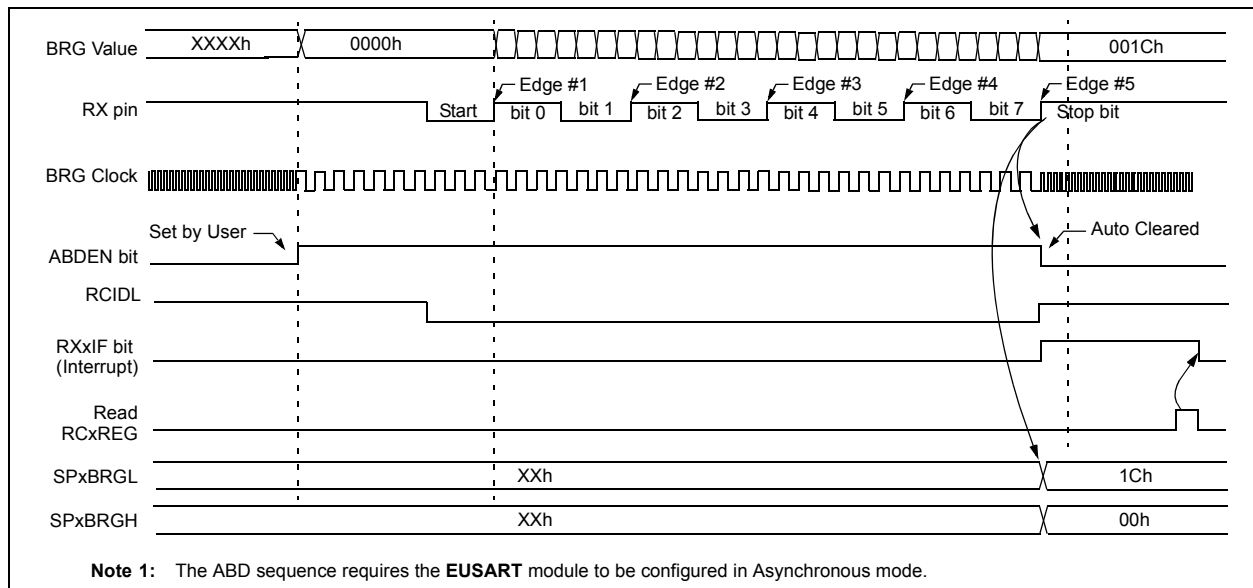
**3:** During the auto-baud process, the auto-baud counter starts counting at one. Upon completion of the auto-baud sequence, to achieve maximum accuracy, subtract 1 from the SPxBRGH:SPxBRGL register pair.

**TABLE 33-1: 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**



## 33.3.2 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 RXxIF 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 RCxREG is read after the overflow occurs but before the fifth rising edge then the fifth rising edge will set the RXxIF 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 RCxREG to clear RXxIF.
2. If RCIDL is '0' then wait for RDCIF and repeat step 1.
3. Clear the ABDOVF bit.

## 33.3.3 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 RXxIF 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.

## 33.3.3.1 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

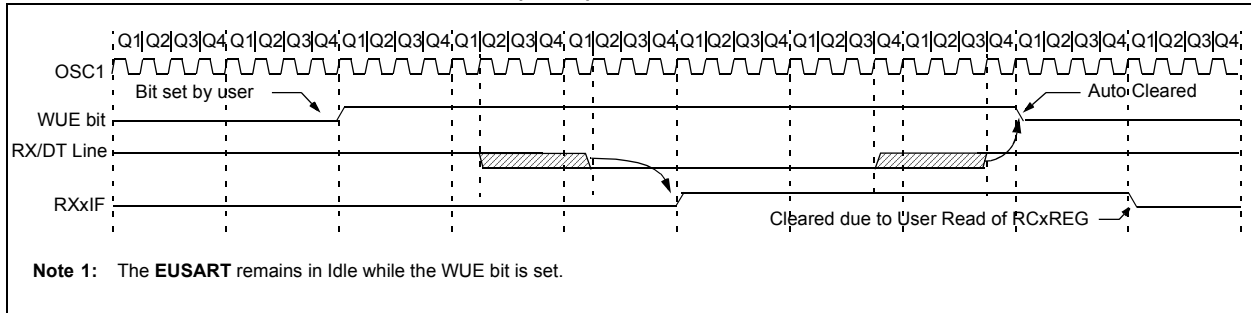
Oscillator start-up time must be considered, especially in applications using oscillators with longer start-up intervals (i.e., LP, XT or HS/PLL mode). The Sync Break (or wake-up signal) character must be of sufficient length, and be followed by a sufficient interval, to allow enough time for the selected oscillator to start and provide proper initialization of the EUSART.

### WUE Bit

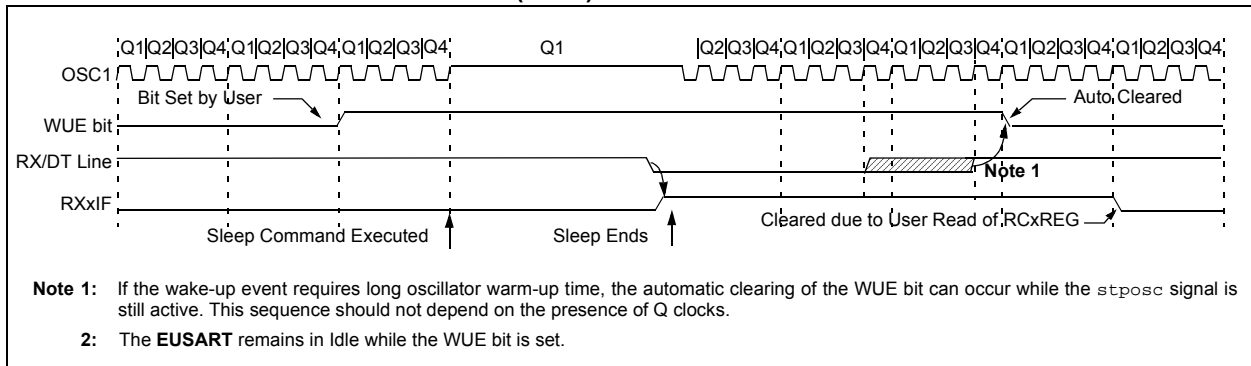
The wake-up event causes a receive interrupt by setting the RXxIF bit. The WUE bit is cleared in hardware by a rising edge on RX/DT. The interrupt condition is then cleared in software by reading the RCxREG register and discarding its contents.

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**



**FIGURE 33-8: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING SLEEP**



### 33.3.4 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.

#### 33.3.4.1 Break and Sync Transmit Sequence

The following sequence will start a message frame header made up of a Break, followed by an auto-baud Sync byte. This sequence is typical of a LIN bus master.

1. Configure the EUSART for the desired mode.
2. Set the TXEN and SENDB bits to enable the Break sequence.
3. Load the TXxREG with a dummy character to initiate transmission (the value is ignored).
4. Write '55h' to TXxREG to load the Sync character into the transmit FIFO buffer.
5. After the Break has been sent, the SENDB bit is reset by hardware and the Sync character is then transmitted.

When the TXxREG becomes empty, as indicated by the TXxIF, the next data byte can be written to TXxREG.

## 33.3.5 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.

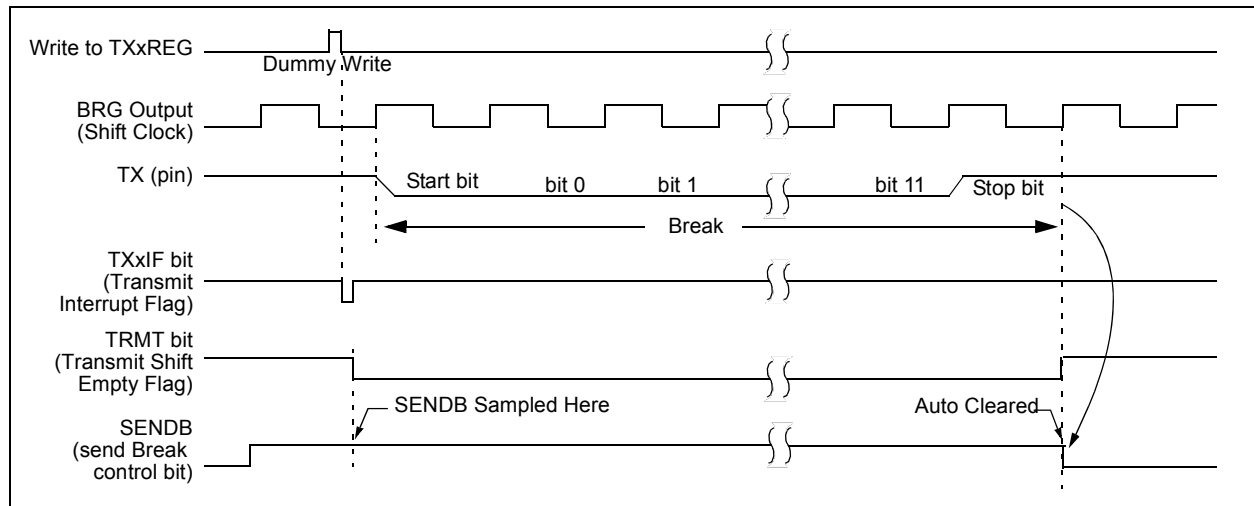
A Break character has been received when:

- RXxIF bit is set
- FERR bit is set
- RCxREG = 00h

The second method uses the Auto-Wake-up feature described in [Section 33.3.3 “Auto-Wake-up on Break”](#). By enabling this feature, the EUSART will sample the next two transitions on RX/DT, cause an RXxIF 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**





## 33.4 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.

### 33.4.1 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.

#### 33.4.1.1 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.

#### 33.4.1.2 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.

#### 33.4.1.3 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.

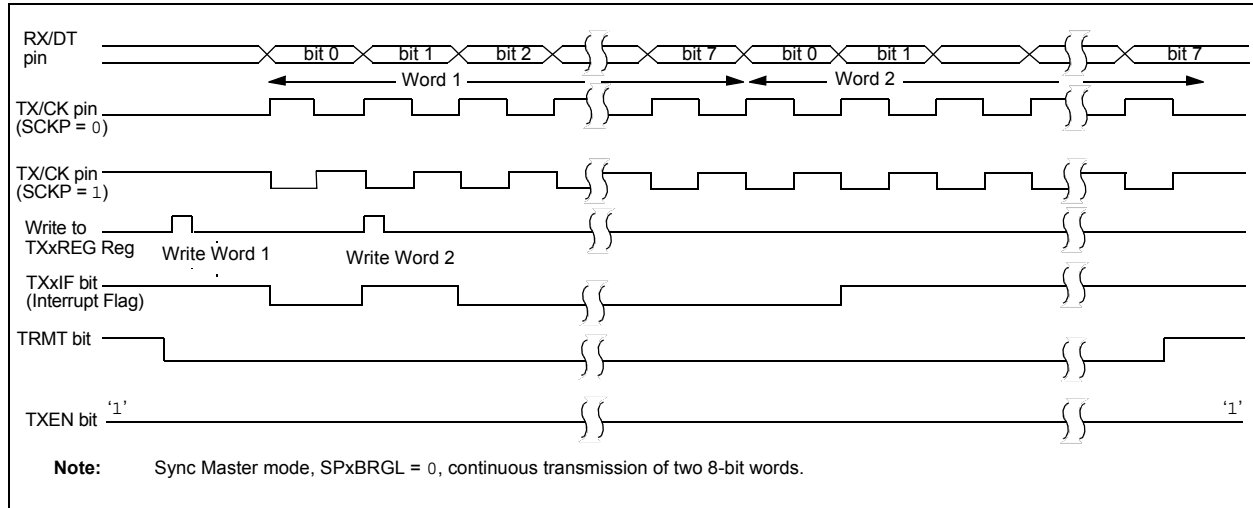
**Note:** The TSR register is not mapped in data memory, so it is not available to the user.

#### 33.4.1.4 Synchronous Master Transmission Set-up:

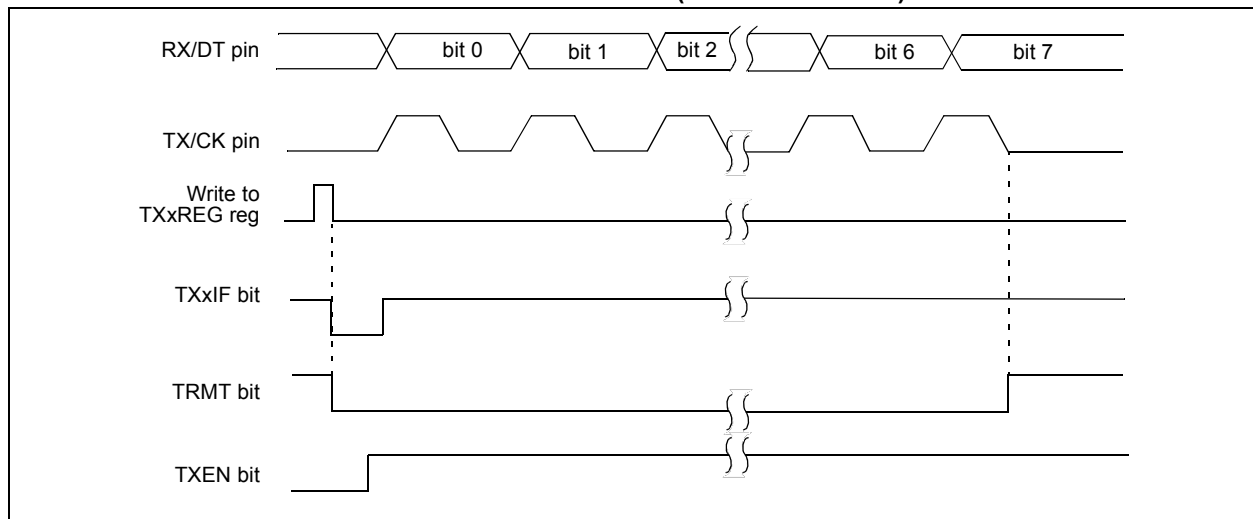
1. Initialize the SPxBRGH, SPxBRGL register pair and the BRGH and BRG16 bits to achieve the desired baud rate (see [Section 33.3 “EUSART Baud Rate Generator \(BRG\)”](#)).
2. Enable the synchronous master serial port by setting bits SYNC, SPEN and CSRC.
3. Disable Receive mode by clearing bits SREN and 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 TXXIE bit of the PIE3 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**



**FIGURE 33-11: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)**



### 33.4.1.5 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 RXxIF 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 RXxIF 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.

## 33.4.1.6 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.

## 33.4.1.7 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.

## 33.4.1.8 Receiving 9-bit Characters

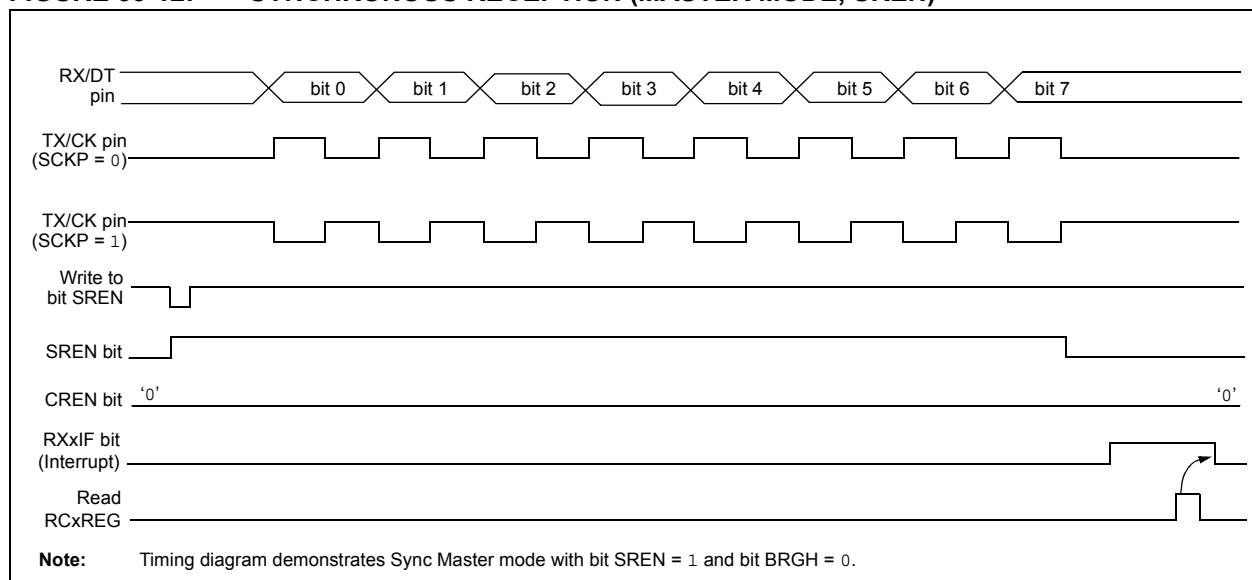
The EUSART supports 9-bit character reception. When the RX9 bit of the RCxSTA register is set the EUSART will shift nine bits into the RSR for each character

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.

## 33.4.1.9 Synchronous Master Reception Setup:

1. Initialize the SPxBRGH, SPxBRGL register pair for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. 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 RXxIE bit of the PIE3 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 RXxIF will be set when reception of a character is complete. An interrupt will be generated if the enable bit RXxIE 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)**



## 33.4.2 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.

### 33.4.2.1 EUSART Synchronous Slave Transmit

The operation of the Synchronous Master and Slave modes are identical (see [Section 33.4.1.3 “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 TXxIF bit will not be set.
4. After the first character has been shifted out of TSR, the TXxREG register will transfer the second character to the TSR and the TXxIF bit will now be set.
5. If the PEIE and TXxIE bits are set, the interrupt will wake the device from Sleep and execute the next instruction. If the GIE bit is also set, the program will call the Interrupt Service Routine.

### 33.4.2.2 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 TXxIE bit of the PIE3 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.

## 33.4.2.3 EUSART Synchronous Slave Reception

The operation of the Synchronous Master and Slave modes is identical ([Section 33.4.1.5 “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 RXxIE 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.

## 33.4.2.4 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 RXxIE bit of the PIE3 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 RXxIF bit will be set when reception is complete. An interrupt will be generated if the RXxIE 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.

## 33.5 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.

### 33.5.1 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 33.4.2.4 “Synchronous Slave Reception Set-up:”](#)).
- If interrupts are desired, set the RXxIE bit of the PIE3 register and the GIE and PEIE bits of the INTCON register.
- The RXxIF 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 RXxIF interrupt flag bit of the PIR3 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.

### 33.5.2 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 33.4.2.2 “Synchronous Slave Transmission Set-up:”](#)).
- The TXxIF 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 TXxIE bit of the PIE3 register and the PEIE bit of the INTCON register.
- Interrupt enable bits TXxIE of the PIE3 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 TXxIF 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 TXxIF 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.

## 33.6 Register Definitions: EUSART Control

### REGISTER 33-1: TXxSTA: 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 <sup>(1)</sup>	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:  
 Unused in this mode – value ignored  
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<sup>(1)</sup>  
 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 SYNCH BREAK on next transmission – Start bit, followed by 12 '0' bits, followed by Stop bit; cleared by hardware upon completion  
 0 = SYNCH BREAK transmission disabled or completed  
Synchronous mode:  
 Unused in this mode – value ignored
- bit 2      **BRGH:** High Baud Rate Select bit  
Asynchronous mode:  
 1 = High speed  
 0 = Low speed  
Synchronous mode:  
 Unused in this mode – value ignored
- 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: RCxSTA: 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 <sup>(1)</sup>	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<sup>(1)</sup>  
             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:  
 Unused in this mode – value ignored  
Synchronous mode – Master:  
             1 = Enables single receive  
             0 = Disables single receive  
 This bit is cleared after reception is complete.  
Synchronous mode – Slave  
 Unused in this mode – value ignored
- bit 4      **CREN:** Continuous Receive Enable bit  
Asynchronous mode:  
             1 = Enables continuous receive until enable bit CREN is cleared  
             0 = Disables continuous receive  
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 of the receive buffer when the ninth bit in  
                     the receive buffer 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):  
 Unused in this mode – value ignored
- 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.

**Note 1:** The EUSART module automatically changes the pin from tri-state to drive as needed. Configure the associated TRIS bits for TX/CK and RX/DT to 1.

## REGISTER 33-3: BAUDxCON: BAUD RATE CONTROL REGISTER

R/W-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:** Clock/Transmit Polarity Select bit  
Asynchronous mode:  
 1 = Idle state for transmit (TX) is a low level  
 0 = Idle state for transmit (TX) is a high level  
Synchronous mode:  
 1 = Idle state for clock (CK) is a high level  
 0 = Idle state for clock (CK) is a low level
- 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 = USART will continue to sample the Rx pin – interrupt generated on falling edge; bit cleared in hardware on following rising edge.  
 0 = RX pin not monitored nor rising edge detected  
Synchronous mode:  
 Unused in this mode – value ignored
- bit 0      **ABDEN:** Auto-Baud Detect Enable bit  
Asynchronous mode:  
 1 = Enable baud rate measurement on the next character – requires reception of a SYNCH field (55h);  
     cleared in hardware upon completion  
 0 = Baud rate measurement disabled or completed  
Synchronous mode:  
 Unused in this mode – value ignored



## REGISTER 33-4: RCxREG<sup>(1)</sup>: RECEIVE DATA REGISTER

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
RCxREG<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      **RCxREG<7:0>**: Lower eight bits of the received data; read-only; see also RX9D ([Register 33-2](#))

**Note 1:** RCxREG (including the 9<sup>th</sup> bit) is double buffered, and data is available while new data is being received.

## REGISTER 33-5: TXxREG<sup>(1)</sup>: TRANSMIT DATA REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
TXxREG<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      **TXxREG<7:0>**: Lower eight bits of the received data; read-only; see also RX9D ([Register 33-1](#))

**Note 1:** TXxREG (including the 9<sup>th</sup> bit) is double buffered, and can be written when previous data has started shifting.

## REGISTER 33-6: SPxBRGL<sup>(1)</sup>: BAUD RATE GENERATOR REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SPxBRG<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      **SPxBRG<7:0>**: Lower eight bits of the Baud Rate Generator

**Note 1:** Writing to SP1BRG resets the BRG counter.

---



---

**REGISTER 33-7: SPxBRGH<sup>(1, 2)</sup>: BAUD RATE GENERATOR HIGH REGISTER**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SPxBRG<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            **SPxBRG<15:8>**: Upper eight bits of the Baud Rate Generator

**Note 1:** SPxBRGH value is ignored for all modes unless BAUDxCON<BRG16> is active.

**2:** Writing to SPxBRGH resets the BRG counter.

**TABLE 33-2: SUMMARY OF REGISTERS ASSOCIATED WITH EUSART**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
INTCON	GIE	PEIE	—	—	—	—	—	INTEDG	121
PIE3	RC2IE	TX2IE	RC1IE	TX1IE	BCL2IE	SSP2IE	BCL1IE	SSP1IE	125
PIR3	RC2IF	TX2IF	RC1IF	TX1IF	BCL2IF	SSP2IF	BCL1IF	SSP1IF	133
RCxSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	443
TXxSTA	CSRC	TX9	TXEN	SYNC	SEnDB	BRGH	TRMT	TX9D	442
BAUDxCON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	444
RCxREG	EUSART Receive Data Register								445*
TXxREG	EUSART Transmit Data Register								445*
SPxBRGL	SPxBRG<7:0>								445*
SPxBRGH	SPxBRG<15:8>								446*
RXPPS	—	—	RXPPS<5:0>						199
CKPPS	—	—	CXPPS<5:0>						199
RxyPPS	—	—	—	RxyPPS<4:0>					200
CLCxSELY	—	—	LCxDyS<5:0>						364

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

\* Page with register information.

**TABLE 33-3: BAUD RATE FORMULAS**

Configuration Bits			BRG/EUSART Mode	Baud Rate Formula
SYNC	BRG16	BRGH		
0	0	0	8-bit/Asynchronous	$F_{osc}/[64(n+1)]$
0	0	1	8-bit/Asynchronous	$F_{osc}/[16(n+1)]$
0	1	0	16-bit/Asynchronous	
0	1	1	16-bit/Asynchronous	$F_{osc}/[4(n+1)]$
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: BAUD RATE 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	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG 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	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)
300	—	—	—	300	0.16	207	300	0.00	191	300	0.16	51
1200	1202	0.16	103	1202	0.16	51	1200	0.00	47	1202	0.16	12
2400	2404	0.16	51	2404	0.16	25	2400	0.00	23	—	—	—
9600	9615	0.16	12	—	—	—	9600	0.00	5	—	—	—
10417	10417	0.00	11	10417	0.00	5	—	—	—	—	—	—
19.2k	—	—	—	—	—	—	19.20k	0.00	2	—	—	—
57.6k	—	—	—	—	—	—	57.60k	0.00	0	—	—	—
115.2k	—	—	—	—	—	—	—	—	—	—	—	—

**TABLE 33-4: BAUD RATE FOR ASYNCHRONOUS MODES (CONTINUED)**

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	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG 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

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	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG 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	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG 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

**TABLE 33-4: BAUD RATE FOR ASYNCHRONOUS MODES (CONTINUED)**

BAUD RATE	SYNC = 0, BRGH = 0, BRG16 = 1											
	Fosc = 8.000 MHz			Fosc = 4.000 MHz			Fosc = 3.6864 MHz			Fosc = 1.000 MHz		
	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG 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	—	—	—

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	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG 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	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG value (decimal)	Actual Rate	% Error	SPBRG 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	—	—	—

## 34.0 REFERENCE CLOCK OUTPUT MODULE

The reference clock output module provides the ability to send a clock signal to the clock reference output pin (CLKR).

The reference clock output module has the following features:

- Selectable input clock
- Programmable clock divider
- Selectable duty cycle

### 34.1 CLOCK SOURCE

The reference clock output module has a selectable clock source. The CLKRCLK register ([Register 34-2](#)) controls which input is used.

#### 34.1.1 CLOCK SYNCHRONIZATION

Once the reference clock enable (CLKREN) is set, the module is ensured to be glitch-free at start-up.

When the reference clock output is disabled, the output signal will be disabled immediately.

Clock dividers and clock duty cycles can be changed while the module is enabled, but glitches may occur on the output. To avoid possible glitches, clock dividers and clock duty cycles should be changed only when the CLKREN is clear.

### 34.2 PROGRAMMABLE CLOCK DIVIDER

The module takes the system clock input and divides it based on the value of the CLKRDIV<2:0> bits of the CLKRCON register ([Register 34-1](#)).

The following configurations can be made based on the CLKRDIV<2:0> bits:

- Base clock value
- Base clock value divided by 2
- Base clock value divided by 4
- Base clock value divided by 8
- Base clock value divided by 16
- Base clock value divided by 32
- Base clock value divided by 64
- Base clock value divided by 128

The clock divider values can be changed while the module is enabled; however, in order to prevent glitches on the output, the CLKRDIV<2:0> bits should only be changed when the module is disabled (CLKREN = 0).

### 34.3 SELECTABLE DUTY CYCLE

The CLKRDC<1:0> bits of the CLKRCON register can be used to modify the duty cycle of the output clock. A duty cycle of 25%, 50%, or 75% can be selected for all clock rates, with the exception of the undivided base FOSC value.

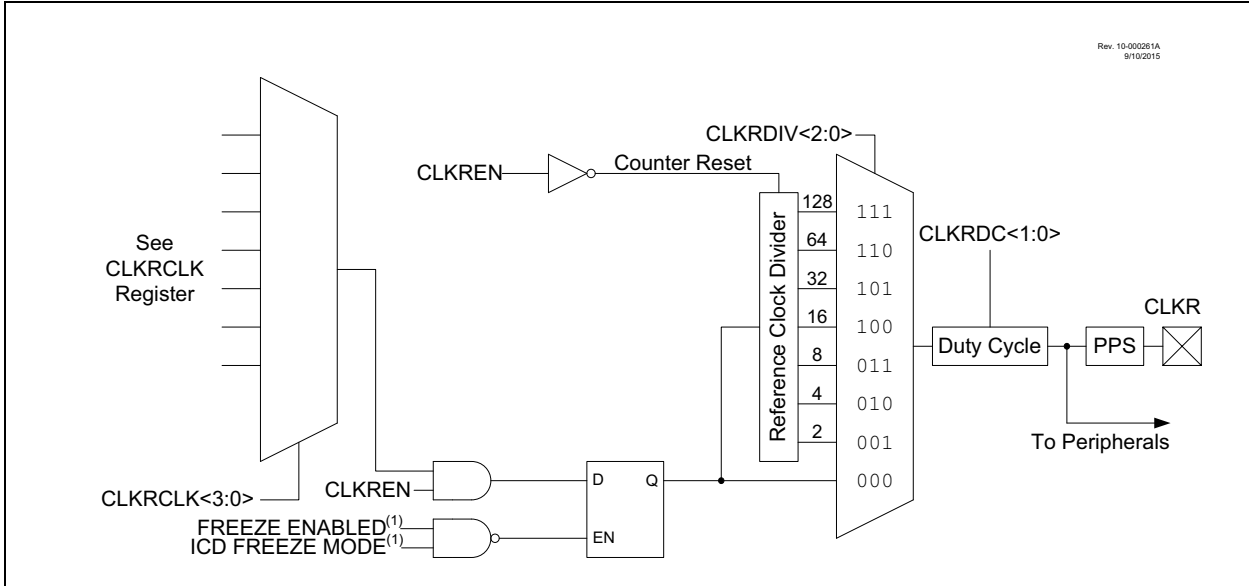
The duty cycle can be changed while the module is enabled; however, in order to prevent glitches on the output, the CLKRDC<1:0> bits should only be changed when the module is disabled (CLKREN = 0).

<b>Note:</b> The CLKRDC1 bit is reset to '1'. This makes the default duty cycle 50% and not 0%.
---

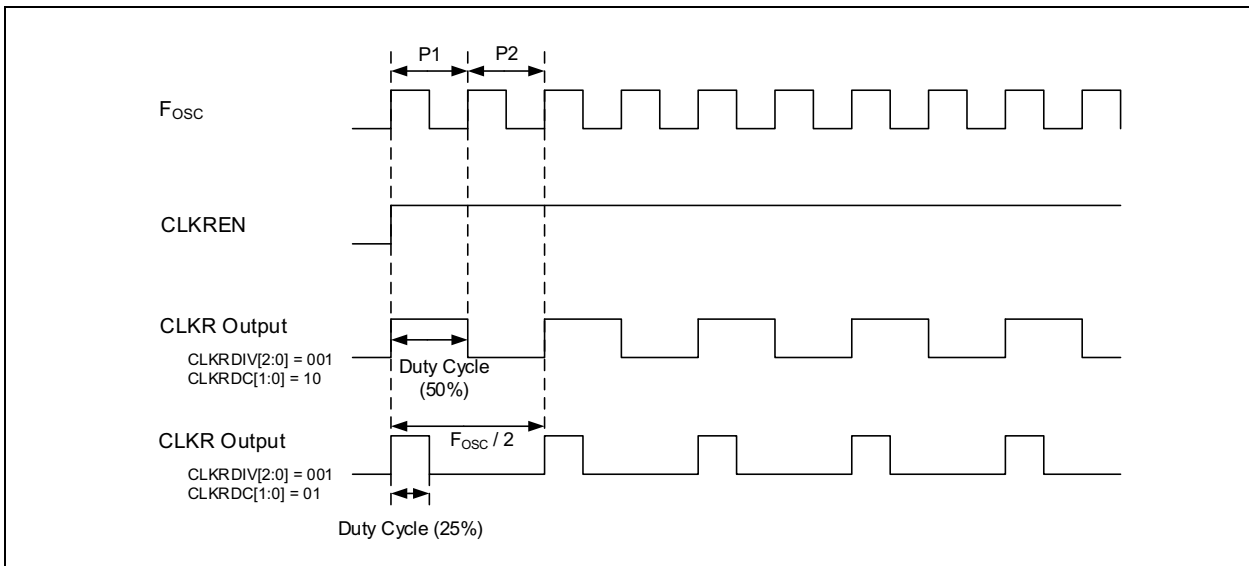
### 34.4 OPERATION IN SLEEP MODE

The reference clock output module clock is based on the system clock. When the device goes to Sleep, the module outputs will remain in their current state. This will have a direct effect on peripherals using the reference clock output as an input signal.

**FIGURE 34-1: CLOCK REFERENCE BLOCK DIAGRAM**



**FIGURE 34-2: CLOCK REFERENCE TIMING**





## 34.5 Register Definition: Reference Clock Output Control

**REGISTER 34-1: CLKRCON: REFERENCE CLOCK CONTROL REGISTER**

R/W-0/0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
CLKREN	—	—	CLKRDC<1:0>			CLKRDIV<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      **CLKREN:** Reference Clock Module Enable bit  
           1 = Reference clock module enabled  
           0 = Reference clock module is disabled
- bit 6-5    **Unimplemented:** Read as '0'
- bit 4-3    **CLKRDC<1:0>:** Reference Clock Duty Cycle bits <sup>(1)</sup>  
           11 = Clock outputs duty cycle of 75%  
           10 = Clock outputs duty cycle of 50%  
           01 = Clock outputs duty cycle of 25%  
           00 = Clock outputs duty cycle of 0%
- bit 2-0    **CLKRDIV<2:0>:** Reference Clock Divider bits  
           111 = Base clock value divided by 128  
           110 = Base clock value divided by 64  
           101 = Base clock value divided by 32  
           100 = Base clock value divided by 16  
           011 = Base clock value divided by 8  
           010 = Base clock value divided by 4  
           001 = Base clock value divided by 2  
           000 = Base clock value

**Note 1:** Bits are valid for reference clock divider values of two or larger, the base clock cannot be further divided.

# PIC16(L)F15354/55

**REGISTER 34-2: CLKRCLK: CLOCK REFERENCE 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
—	—	—	—	CLKRCLK<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                      **CLKRCLK<3:0>:** CLKR Input bits  
Clock Selection  
1111 = Reserved  
. . .  
1011 = Reserved  
1010 = LC4\_out  
1001 = LC3\_out  
1000 = LC2\_out  
0111 = LC1\_out  
0110 = NCO1\_out  
0101 = SOSC  
0100 = MFINTOSC (31.25 kHz)  
0011 = MFINTOSC (500 kHz)  
0010 = LFINTOSC  
0001 = HFINTOSC  
0000 = Fosc

**TABLE 34-1: SUMMARY OF REGISTERS ASSOCIATED WITH CLOCK REFERENCE OUTPUT**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
CLKRCON	CLKREN	—	—	CLKRDC<1:0>		CLKRDIV<2:0>			453
CLKRCLK	—	—	—	—	CLKRCLK<3:0>				454
CLCxSELY	—	—	LCxDyS<5:0>						364
RxyPPS	—	—	—	RxyPPS<4:0>					200

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

## 35.0 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
- MCLR/VPP
- VDD
- VSS

In Program/Verify mode the program memory, User IDs and the Configuration Words are programmed through serial communications. The ICSPDAT pin is a bidirectional I/O used for transferring the serial data and the ICSPCLK pin is the clock input. For more information on ICSP™ refer to the “PIC16(L)F153XX Memory Programming Specification” (DS40001838).

### 35.1 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 MCLR/VPP to VIH.

### 35.2 Low-Voltage Programming Entry Mode

The Low-Voltage Programming Entry mode allows the PIC® Flash MCUs to be programmed using VDD only, without high voltage. When the LVP bit of Configuration Words is set to ‘1’, the low-voltage ICSP programming entry is enabled. To disable the Low-Voltage ICSP mode, the LVP bit must be programmed to ‘0’. The LVP bit can only be reprogrammed to ‘0’ by using the High-Voltage Programming mode.

Entry into the Low-Voltage Programming Entry mode requires the following steps:

1. MCLR is brought to VIL.
2. A 32-bit key sequence is presented on ICSPDAT, while clocking ICSPCLK.

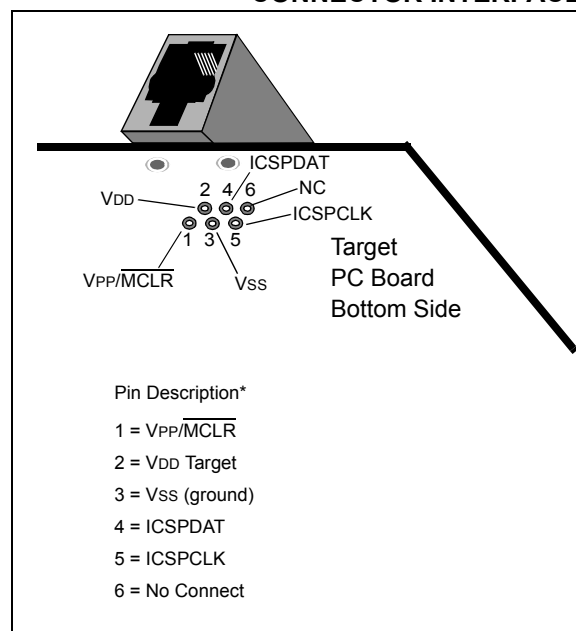
Once the key sequence is complete, MCLR must be held at VIL for as long as Program/Verify mode is to be maintained.

If low-voltage programming is enabled (LVP = 1), the MCLR Reset function is automatically enabled and cannot be disabled. See [Section 8.5 “MCLR”](#) for more information.

## 35.3 Common Programming Interfaces

Connection to a target device is typically done through an ICSP™ header. A commonly found connector on development tools is the RJ-11 in the 6P6C (6-pin, 6-conductor) configuration. See [Figure 35-1](#).

**FIGURE 35-1: ICD RJ-11 STYLE CONNECTOR INTERFACE**

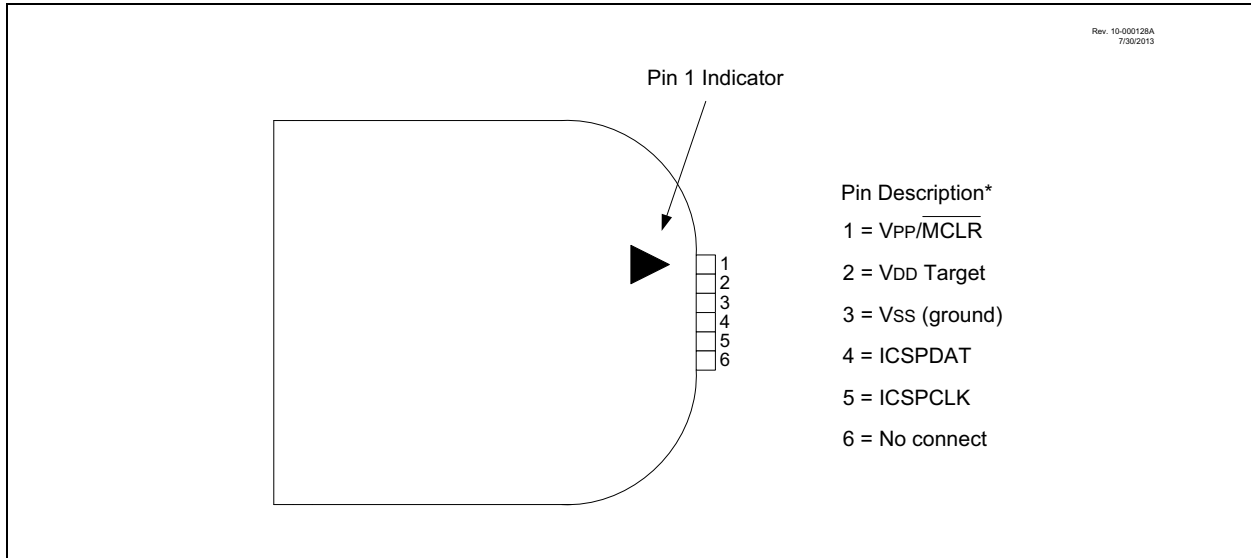


Another connector often found in use with the PICkit™ programmers is a standard 6-pin header with 0.1 inch spacing. Refer to [Figure 35-2](#).

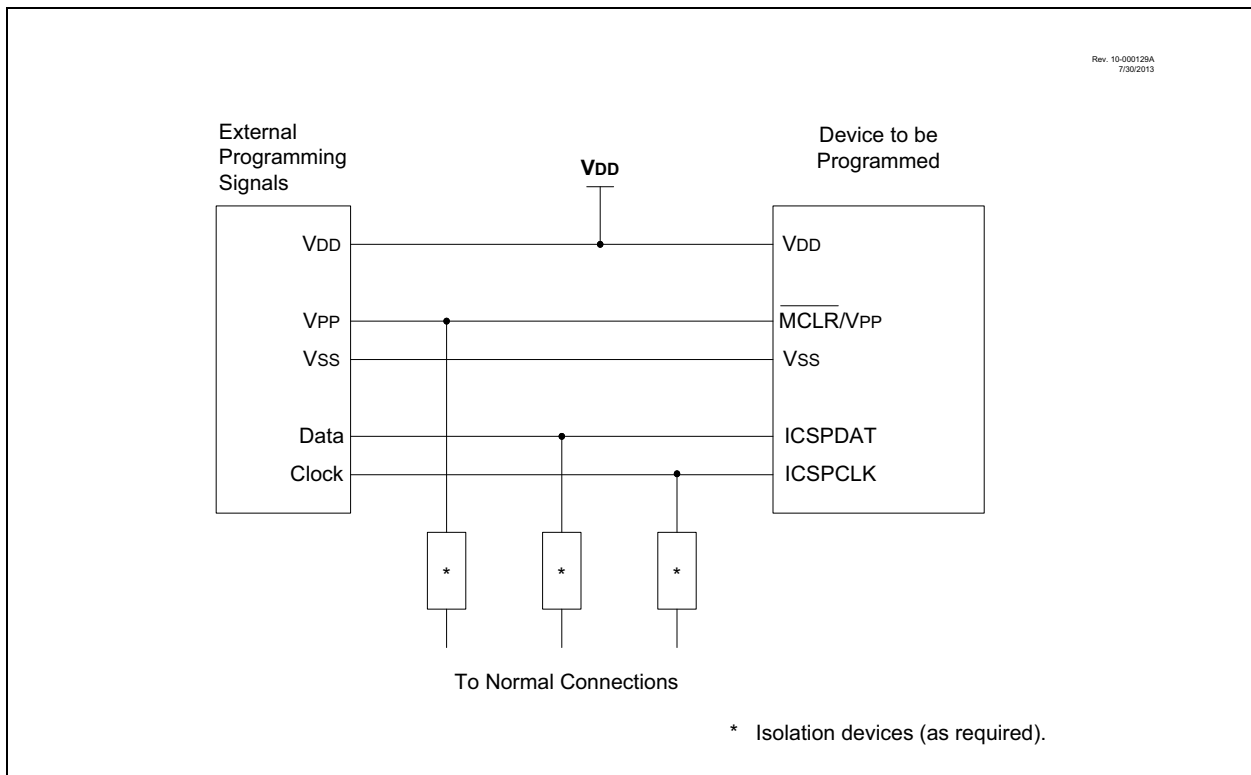
For additional interface recommendations, refer to your specific device programmer manual prior to PCB design.

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 35-3](#) for more information.

**FIGURE 35-2: PICKIT™ PROGRAMMER STYLE CONNECTOR INTERFACE**



**FIGURE 35-3: TYPICAL CONNECTION FOR ICSP™ PROGRAMMING**



## 36.0 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 36-3 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 entry takes two cycles (CALL, CALLW)
- Returns from interrupts or subroutines take two cycles (RETURN, RETLW, RETFIE)
- Program branching takes two cycles (GOTO, BRA, BRW, BTFSS, BTFSC, DECFSZ, INCSFZ)
- One additional instruction cycle will be used when any instruction references an indirect file register and the file select register is pointing to program memory.

One instruction cycle consists of 4 oscillator cycles; for an oscillator frequency of 4 MHz, this gives a nominal instruction execution rate of 1 MHz.

All instruction examples use the format '0xhh' to represent a hexadecimal number, where 'h' signifies a hexadecimal digit.

## 36.1 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 in either the working (W) register, or the originating file register, depending on the state of the destination designator 'd' (see Table 36-1 for more information). A read operation is performed on a register even if the instruction writes to that register.

**TABLE 36-1: OPCODE FIELD DESCRIPTIONS**

Field	Description
f	Register file address (0x00 to 0x7F)
W	Working register (accumulator)
b	Bit address within an 8-bit file register
k	Literal field, constant data or label
x	Don't care location (= 0 or 1). The assembler will generate code with x = 0. It is the recommended form of use for compatibility with all Microchip software tools.
d	Destination select; d = 0: store result in W, d = 1: store result in file register f.
n	FSR or INDF number. (0-1)
mm	Prepost increment-decrement mode selection

**TABLE 36-2: 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

## 36.2 General Format for Instructions

**TABLE 36-3: INSTRUCTION SET**

Mnemonic, Operands	Description	Cycles	14-Bit Opcode				Status Affected	Notes	
			MSb	LSb					
<b>BYTE-ORIENTED FILE REGISTER OPERATIONS</b>									
ADDWF	f, d	Add W and f	1	00	0111	dfff	ffff	C, DC, Z	2
ADDWFC	f, d	Add with Carry W and f	1	11	1101	dfff	ffff	C, DC, Z	2
ANDWF	f, d	AND W with f	1	00	0101	dfff	ffff	Z	2
ASRF	f, d	Arithmetic Right Shift	1	11	0111	dfff	ffff	C, Z	2
LSLF	f, d	Logical Left Shift	1	11	0101	dfff	ffff	C, Z	2
LSRF	f, d	Logical Right Shift	1	11	0110	dfff	ffff	C, Z	2
CLRF	f	Clear f	1	00	0001	1fff	ffff	Z	2
CLRW	–	Clear W	1	00	0001	0000	00xx	Z	
COMF	f, d	Complement f	1	00	1001	dfff	ffff	Z	2
DECF	f, d	Decrement f	1	00	0011	dfff	ffff	Z	2
INCF	f, d	Increment f	1	00	1010	dfff	ffff	Z	2
IORWF	f, d	Inclusive OR W with f	1	00	0100	dfff	ffff	Z	2
MOVF	f, d	Move f	1	00	1000	dfff	ffff	Z	2
MOVWF	f	Move W to f	1	00	0000	1fff	ffff		2
RLF	f, d	Rotate Left f through Carry	1	00	1101	dfff	ffff	C	2
RRF	f, d	Rotate Right f through Carry	1	00	1100	dfff	ffff	C	2
SUBWF	f, d	Subtract W from f	1	00	0010	dfff	ffff	C, DC, Z	2
SUBWFB	f, d	Subtract with Borrow W from f	1	11	1011	dfff	ffff	C, DC, Z	2
SWAPF	f, d	Swap nibbles in f	1	00	1110	dfff	ffff		2
XORWF	f, d	Exclusive OR W with f	1	00	0110	dfff	ffff	Z	2
<b>BYTE ORIENTED SKIP OPERATIONS</b>									
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	00	1011	dfff	ffff		1, 2
INCFSZ	f, d	Increment f, Skip if 0	1(2)	00	1111	dfff	ffff		1, 2
<b>BIT-ORIENTED FILE REGISTER OPERATIONS</b>									
BCF	f, b	Bit Clear f	1	01	00bb	bfff	ffff		2
BSF	f, b	Bit Set f	1	01	01bb	bfff	ffff		2
<b>BIT-ORIENTED SKIP OPERATIONS</b>									
BTFSC	f, b	Bit Test f, Skip if Clear	1 (2)	01	10bb	bfff	ffff		1, 2
BTFSS	f, b	Bit Test f, Skip if Set	1 (2)	01	11bb	bfff	ffff		1, 2
<b>LITERAL OPERATIONS</b>									
ADDLW	k	Add literal and W	1	11	1110	kkkk	kkkk	C, DC, Z	
ANDLW	k	AND literal with W	1	11	1001	kkkk	kkkk	Z	
IORLW	k	Inclusive OR literal with W	1	11	1000	kkkk	kkkk	Z	
MOVLB	k	Move literal to BSR	1	00	000	0k	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.
- Note 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 36-3: INSTRUCTION SET (CONTINUED)**

Mnemonic, Operands	Description	Cycles	14-Bit Opcode				Status Affected	Notes
			MSb			LSb		
<b>CONTROL OPERATIONS</b>								
BRA	k	Relative Branch	2	11	001k	kkkk	kkkk	
BRW	–	Relative Branch with W	2	00	0000	0000	1011	
CALL	k	Call Subroutine	2	10	0kkk	kkkk	kkkk	
CALLW	–	Call Subroutine with W	2	00	0000	0000	1010	
GOTO	k	Go to address	2	10	1kkk	kkkk	kkkk	
RETFIE	k	Return from interrupt	2	00	0000	0000	1001	
RETLW	k	Return with literal in W	2	11	0100	kkkk	kkkk	
RETURN	–	Return from Subroutine	2	00	0000	0000	1000	
<b>INHERENT OPERATIONS</b>								
CLRWDT	–	Clear Watchdog Timer	1	00	0000	0110	0100	$\overline{TO}, \overline{PD}$
NOP	–	No Operation	1	00	0000	0000	0000	
RESET	–	Software device Reset	1	00	0000	0000	0001	
SLEEP	–	Go into Standby or IDLE mode	1	00	0000	0110	0011	$\overline{TO}, \overline{PD}$
TRIS	f	Load TRIS register with W	1	00	0000	0110	0fff	
<b>C-COMPILER OPTIMIZED</b>								
ADDFSR	n, k	Add Literal k to FSRn	1	11	0001	0nkk	kkkk	
MOVIW	n mm	Move Indirect FSRn to W with pre/post inc/dec modifier, mm	1	00	0000	0001	0nmm	Z
	k[n]	Move INDFn to W, Indexed Indirect.	1	11	1111	0nkk	kkkk	Z
MOVWI	n mm	Move W to Indirect FSRn with pre/post inc/dec modifier, mm	1	00	0000	0001	1nmm	
	k[n]	Move W to INDFn, Indexed Indirect.	1	11	1111	1nkk	kkkk	

**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 [Section 36.3 “Instruction Descriptions”](#) for detailed MOVIW and MOVWI instruction descriptions.

## 36.3 Instruction Descriptions

### **ADDFSR**      **Add Literal to FSRn**

**Syntax:**             $[label] \text{ ADDFSR } FSRn, k$

**Operands:**         $-32 \leq k \leq 31$   
 $n \in [0, 1]$

**Operation:**         $FSR(n) + k \rightarrow FSR(n)$

**Status Affected:**    None

**Description:**      The signed 6-bit literal 'k' is added to the contents of the FSRnH:FSRnL register pair.

FSRn is limited to the range 0000h-FFFFh. Moving beyond these bounds will cause the FSR to wrap-around.

### **ANDLW**        **AND literal with W**

**Syntax:**             $[label] \text{ ANDLW } k$

**Operands:**         $0 \leq k \leq 255$

**Operation:**         $(W) \text{ .AND. } (k) \rightarrow (W)$

**Status Affected:**    Z

**Description:**      The contents of W register are AND'ed with the 8-bit literal 'k'. The result is placed in the W register.

### **ADDLW**        **Add literal and W**

**Syntax:**             $[label] \text{ ADDLW } k$

**Operands:**         $0 \leq k \leq 255$

**Operation:**         $(W) + k \rightarrow (W)$

**Status Affected:**    C, DC, Z

**Description:**      The contents of the W register are added to the 8-bit literal 'k' and the result is placed in the W register.

### **ANDWF**        **AND W with f**

**Syntax:**             $[label] \text{ ANDWF } f, d$

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

**Operation:**         $(W) \text{ .AND. } (f) \rightarrow (\text{destination})$

**Status Affected:**    Z

**Description:**      AND the W register with register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.

### **ADDWF**        **Add W and f**

**Syntax:**             $[label] \text{ ADDWF } f, d$

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

**Operation:**         $(W) + (f) \rightarrow (\text{destination})$

**Status Affected:**    C, DC, Z

**Description:**      Add the contents of the W register with register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.

### **ASRF**            **Arithmetic Right Shift**

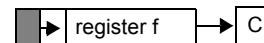
**Syntax:**             $[label] \text{ ASRF } f \{, d\}$

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

**Operation:**         $(f<7>) \rightarrow \text{dest}<7>$   
 $(f<7:1>) \rightarrow \text{dest}<6:0>$ ,  
 $(f<0>) \rightarrow C$ ,

**Status Affected:**    C, Z

**Description:**      The contents of register 'f' are shifted one bit to the right through the Carry flag. The MSb remains unchanged. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is stored back in register 'f'.



### **ADDWFC**      **ADD W and CARRY bit to f**

**Syntax:**             $[label] \text{ ADDWFC } f \{, d\}$

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

**Operation:**         $(W) + (f) + (C) \rightarrow \text{dest}$

**Status Affected:**    C, DC, Z

**Description:**      Add W, the Carry flag and data memory location 'f'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed in data memory location 'f'.



<b>BCF</b>	<b>Bit Clear f</b>
Syntax:	[ <i>label</i> ] BCF f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b \leq 7$
Operation:	$0 \rightarrow (f<b>)$
Status Affected:	None
Description:	Bit 'b' in register 'f' is cleared.

<b>BTFSC</b>	<b>Bit Test f, Skip if Clear</b>
Syntax:	[ <i>label</i> ] BTFSC f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b \leq 7$
Operation:	skip if (f<b>) = 0
Status Affected:	None
Description:	If bit 'b' in register 'f' is '1', the next instruction is executed. If bit 'b', in register 'f', is '0', the next instruction is discarded, and a NOP is executed instead, making this a 2-cycle instruction.

<b>BRA</b>	<b>Relative Branch</b>
Syntax:	[ <i>label</i> ] BRA label [ <i>label</i> ] BRA \$+k
Operands:	$-256 \leq \text{label} - \text{PC} + 1 \leq 255$ $-256 \leq k \leq 255$
Operation:	$(\text{PC}) + 1 + k \rightarrow \text{PC}$
Status Affected:	None
Description:	Add the signed 9-bit literal 'k' to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be $\text{PC} + 1 + k$ . This instruction is a 2-cycle instruction. This branch has a limited range.

<b>BTFSS</b>	<b>Bit Test f, Skip if Set</b>
Syntax:	[ <i>label</i> ] BTFSS f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b < 7$
Operation:	skip if (f<b>) = 1
Status Affected:	None
Description:	If bit 'b' in register 'f' is '0', the next instruction is executed. If bit 'b' is '1', then the next instruction is discarded and a NOP is executed instead, making this a 2-cycle instruction.

<b>BRW</b>	<b>Relative Branch with W</b>
Syntax:	[ <i>label</i> ] BRW
Operands:	None
Operation:	$(\text{PC}) + (W) \rightarrow \text{PC}$
Status Affected:	None
Description:	Add the contents of W (unsigned) to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be $\text{PC} + 1 + (W)$ . This instruction is a 2-cycle instruction.

<b>BSF</b>	<b>Bit Set f</b>
Syntax:	[ <i>label</i> ] BSF f,b
Operands:	$0 \leq f \leq 127$ $0 \leq b \leq 7$
Operation:	$1 \rightarrow (f<b>)$
Status Affected:	None
Description:	Bit 'b' in register 'f' is set.

## CALL Call Subroutine

**Syntax:** [ *label* ] CALL *k*

**Operands:**  $0 \leq k \leq 2047$

**Operation:** (PC)+ 1 → TOS,  
 $k \rightarrow PC<10:0>$ ,  
(PCLATH<6:3>) → PC<14:11>

**Status Affected:** None

**Description:** Call Subroutine. First, return address (PC + 1) is pushed onto the stack. The 11-bit immediate address is loaded into PC bits <10:0>. The upper bits of the PC are loaded from PCLATH. CALL is a 2-cycle instruction.

## CLRWDT Clear Watchdog Timer

**Syntax:** [ *label* ] CLRWDT

**Operands:** None

**Operation:** 00h → WDT  
0 → WDT prescaler,  
1 →  $\overline{TO}$   
1 →  $\overline{PD}$

**Status Affected:**  $\overline{TO}$ ,  $\overline{PD}$

**Description:** CLRWDT instruction resets the Watchdog Timer. It also resets the prescaler of the WDT. Status bits  $\overline{TO}$  and  $\overline{PD}$  are set.

## CALLW Subroutine Call With W

**Syntax:** [ *label* ] CALLW

**Operands:** None

**Operation:** (PC) + 1 → TOS,  
(W) → PC<7:0>,  
(PCLATH<6:0>) → PC<14:8>

**Status Affected:** None

**Description:** Subroutine call with W. First, the return address (PC + 1) is pushed onto the return stack. Then, the contents of W is loaded into PC<7:0>, and the contents of PCLATH into PC<14:8>. CALLW is a 2-cycle instruction.

## COMF Complement f

**Syntax:** [ *label* ] COMF *f,d*

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

**Operation:** ( $\bar{f}$ ) → (destination)

**Status Affected:** Z

**Description:** The contents of register 'f' are complemented. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f'.

## CLRF Clear f

**Syntax:** [ *label* ] CLRF *f*

**Operands:**  $0 \leq f \leq 127$

**Operation:** 00h → (f)  
1 → Z

**Status Affected:** Z

**Description:** The contents of register 'f' are cleared and the Z bit is set.

## DECF Decrement f

**Syntax:** [ *label* ] DECF *f,d*

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

**Operation:** (f) - 1 → (destination)

**Status Affected:** Z

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

## CLRW Clear W

**Syntax:** [ *label* ] CLRW

**Operands:** None

**Operation:** 00h → (W)  
1 → Z

**Status Affected:** Z

**Description:** W register is cleared. Zero bit (Z) is set.

## **DECFSZ**      **Decrement f, Skip if 0**

**Syntax:**            `[ label ] DECFSZ f,d`

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

**Operation:**         $(f) - 1 \rightarrow (\text{destination});$   
skip if result = 0

**Status Affected:**    None

**Description:**      The contents of register 'f' are decremented. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'. If the result is '1', the next instruction is executed. If the result is '0', then a NOP is executed instead, making it a 2-cycle instruction.

## **INCFSZ**        **Increment f, Skip if 0**

**Syntax:**            `[ label ] INCFSZ f,d`

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

**Operation:**         $(f) + 1 \rightarrow (\text{destination});$   
skip if result = 0

**Status Affected:**    None

**Description:**      The contents of register 'f' are incremented. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'. If the result is '1', the next instruction is executed. If the result is '0', a NOP is executed instead, making it a 2-cycle instruction.

## **GOTO**            **Unconditional Branch**

**Syntax:**            `[ label ] GOTO k`

**Operands:**         $0 \leq k \leq 2047$

**Operation:**         $k \rightarrow \text{PC}<10:0>$   
 $\text{PCLATH}<6:3> \rightarrow \text{PC}<14:11>$

**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.

## **IORLW**          **Inclusive OR literal with W**

**Syntax:**            `[ label ] IORLW k`

**Operands:**         $0 \leq k \leq 255$

**Operation:**         $(W) .\text{OR. } k \rightarrow (W)$

**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.

## **INCF**            **Increment f**

**Syntax:**            `[ label ] INCF f,d`

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

**Operation:**         $(f) + 1 \rightarrow (\text{destination})$

**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'.

## **IORWF**          **Inclusive OR W with f**

**Syntax:**            `[ label ] IORWF f,d`

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

**Operation:**         $(W) .\text{OR. } (f) \rightarrow (\text{destination})$

**Status Affected:**    Z

**Description:**      Inclusive OR the W register with register 'f'. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'.

**LSLF**                      **Logical Left Shift**

---

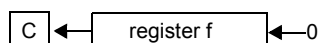
Syntax:                      [ *label* ] LSLF f {,d}

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

Operation:                    (f<7>) → 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**

---

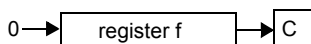
Syntax:                      [ *label* ] LSRF f {,d}

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

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

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:**  $n \in [0,1]$   
 $mm \in [00,01, 10, 11]$   
 $-32 \leq k \leq 31$

**Operation:** INDFn  $\rightarrow$  W  
 Effective address is determined by

- FSR + 1 (preincrement)
- FSR - 1 (predecrement)
- FSR + k (relative offset)

After the Move, the FSR value will be either:

- FSR + 1 (all increments)
- FSR - 1 (all decrements)
- Unchanged

**Status Affected:** Z

Mode	Syntax	mm
Preincrement	++FSRn	00
Predecrement	--FSRn	01
Postincrement	FSRn++	10
Postdecrement	FSRn--	11

**Description:** This instruction is used to move data between W and one of the indirect registers (INDFn). Before/after this move, the pointer (FSRn) is updated by pre/post incrementing/decrementing it.

**Note:** The INDFn registers are not physical registers. Any instruction that accesses an INDFn register actually accesses the register at the address specified by the FSRn.

FSRn is limited to the range 0000h - FFFFh. Incrementing/decrementing it beyond these bounds will cause it to wrap-around.

## MOVLB Move literal to BSR

**Syntax:** [ *label* ] MOVLB k

**Operands:**  $0 \leq k \leq$

**Operation:**  $k \rightarrow$  BSR

**Status Affected:** None

**Description:** The 6-bit literal 'k' is loaded into the Bank Select Register (BSR).

## MOVLP Move literal to PCLATH

**Syntax:** [ *label* ] MOVLP k

**Operands:**  $0 \leq k \leq 127$

**Operation:**  $k \rightarrow$  PCLATH

**Status Affected:** None

**Description:** The 7-bit literal 'k' is loaded into the PCLATH register.

## MOVLW Move literal to W

**Syntax:** [ *label* ] MOVLW k

**Operands:**  $0 \leq k \leq 255$

**Operation:**  $k \rightarrow$  (W)

**Status Affected:** None

**Description:** The 8-bit literal 'k' is loaded into W register. The "don't cares" will assemble as '0's.

**Words:** 1

**Cycles:** 1

**Example:** MOVLW 0x5A  
 After Instruction  
 W = 0x5A

## MOVWF Move W to f

**Syntax:** [ *label* ] MOVWF f

**Operands:**  $0 \leq f \leq 127$

**Operation:** (W)  $\rightarrow$  (f)

**Status Affected:** None

**Description:** Move data from W register to register 'f'.

**Words:** 1

**Cycles:** 1

**Example:** MOVWF LATA  
 Before Instruction  
 LATA = 0xFF  
 W = 0x4F  
 After Instruction  
 LATA = 0x4F  
 W = 0x4F

MOVWI	Move W to INDFn
Syntax:	[ <i>label</i> ] MOVWI ++FSRn [ <i>label</i> ] MOVWI --FSRn [ <i>label</i> ] MOVWI FSRn++ [ <i>label</i> ] MOVWI FSRn-- [ <i>label</i> ] MOVWI k[FSRn]
Operands:	n ∈ [0,1] mm ∈ [00,01, 10, 11] -32 ≤ k ≤ 31
Operation:	W → INDFn Effective address is determined by <ul style="list-style-type: none"> <li>• FSR + 1 (preincrement)</li> <li>• FSR - 1 (predecrement)</li> <li>• FSR + k (relative offset)</li> </ul> After the Move, the FSR value will be either: <ul style="list-style-type: none"> <li>• FSR + 1 (all increments)</li> <li>• FSR - 1 (all decrements)</li> </ul> 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:	[ <i>label</i> ] NOP
Operands:	None
Operation:	No operation
Status Affected:	None
Description:	No operation.
Words:	1
Cycles:	1
<b>Example:</b>	NOP

RESET	Software Reset
Syntax:	[ <i>label</i> ] RESET
Operands:	None
Operation:	Execute a device Reset. Resets the 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:	[ <i>label</i> ] 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
<b>Example:</b>	RETFIE
	After Interrupt
	PC = TOS
	GIE = 1

# PIC16(L)F15354/55

## RETLW Return with literal in W

Syntax: `[label] RETLW k`

Operands:  $0 \leq k \leq 255$

Operation:  $k \rightarrow (W)$ ;  
TOS  $\rightarrow$  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
```

## RETURN Return from Subroutine

Syntax: `[label] RETURN`

Operands: None

Operation: TOS  $\rightarrow$  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.

Before Instruction  
W = 0x07

After Instruction  
W = value of k8

## RLF Rotate Left f through Carry

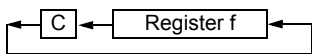
Syntax: `[label] RLF f,d`

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

Operation: See description below

Status Affected: C

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



Words: 1

Cycles: 1

**Example:**

```
RLF REG1,0
```

Before Instruction

```
REG1 = 1110 0110
C = 0
```

After Instruction

```
REG1 = 1110 0110
W = 1100 1100
C = 1
```

## RRF Rotate Right f through Carry

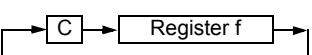
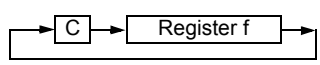
Syntax: `[label] RRF f,d`

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

Operation: See description below

Status Affected: C

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

**SLEEP**                      **Enter Sleep mode**

---

Syntax:                      [ *label* ] SLEEP

Operands:                    None

Operation:                   00h → WDT,  
                                   0 → WDT prescaler,  
                                   1 →  $\overline{TO}$ ,  
                                   0 →  $\overline{PD}$

Status Affected:            $\overline{TO}$ ,  $\overline{PD}$

Description:                The power-down Status bit,  $\overline{PD}$  is cleared. Time-out Status bit,  $\overline{TO}$  is set. Watchdog Timer and its prescaler are cleared. See [Section 11.2 “Sleep Mode”](#) for more information.

**SUBLW**                      **Subtract W from literal**

---

Syntax:                      [ *label* ] SUBLW k

Operands:                     $0 \leq k \leq 255$

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	$W \leq k$
DC = 0	$W\langle 3:0 \rangle > k\langle 3:0 \rangle$
DC = 1	$W\langle 3:0 \rangle \leq k\langle 3:0 \rangle$

**SUBWF**                      **Subtract W from f**

---

Syntax:                      [ *label* ] SUBWF f,d

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

Operation:                   (f) - (W) → (destination)

Status Affected:            C, DC, Z

Description:                Subtract (2's complement method) W register from register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.

C = 0	$W > f$
C = 1	$W \leq f$
DC = 0	$W\langle 3:0 \rangle > f\langle 3:0 \rangle$
DC = 1	$W\langle 3:0 \rangle \leq f\langle 3:0 \rangle$

**SUBWFB**                      **Subtract W from f with Borrow**

---

Syntax:                      SUBWFB f {,d}

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

Operation:                   (f) - (W) - ( $\overline{B}$ ) → 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:                     $0 \leq f \leq 127$   
                                    $d \in [0,1]$

Operation:                   (f<3:0>) → (destination<7:4>),  
                                   (f<7:4>) → (destination<3:0>)

Status Affected:            None

Description:                The upper and lower nibbles of register 'f' are exchanged. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed in register 'f'.



## **TRIS**                      **Load TRIS Register with W**

---

Syntax:                    [ *label* ] TRIS *f*  
Operands:                 $5 \leq f \leq 7$   
Operation:                (W) → TRIS register 'f'  
Status Affected:        None  
Description:             Move data from W register to TRIS register.  
                              When 'f' = 5, TRISA is loaded.  
                              When 'f' = 6, TRISB is loaded.  
                              When 'f' = 7, TRISC is loaded.

## **XORLW**                    **Exclusive OR literal with W**

---

Syntax:                    [ *label* ] XORLW *k*  
Operands:                 $0 \leq k \leq 255$   
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:                 $0 \leq f \leq 127$   
                               $d \in [0,1]$   
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'.

## 37.0 ELECTRICAL SPECIFICATIONS

### 37.1 Absolute Maximum Ratings<sup>(†)</sup>

Ambient temperature under bias .....	-40°C to +125°C
Storage temperature .....	-65°C to +150°C
Voltage on pins with respect to V <sub>SS</sub>	
on V <sub>DD</sub> pin	
PIC16F15354/55 .....	-0.3V to +6.5V
PIC16LF15354/55 .....	-0.3V to +4.0V
on $\overline{\text{MCLR}}$ pin .....	-0.3V to +9.0V
on all other pins .....	-0.3V to (V <sub>DD</sub> + 0.3V)
Maximum current	
on V <sub>SS</sub> pin <sup>(1)</sup>	
-40°C ≤ T <sub>A</sub> ≤ +85°C .....	350 mA
85°C < T <sub>A</sub> ≤ +125°C .....	120 mA
on V <sub>DD</sub> pin for 28-Pin devices <sup>(1)</sup>	
-40°C ≤ T <sub>A</sub> ≤ +85°C .....	250 mA
85°C < T <sub>A</sub> ≤ +125°C .....	85 mA
on V <sub>DD</sub> pin for 40-Pin devices <sup>(1)</sup>	
-40°C ≤ T <sub>A</sub> ≤ +85°C .....	350 mA
85°C < T <sub>A</sub> ≤ +125°C .....	120 mA
on any standard I/O pin .....	±50 mA
Clamp current, I <sub>K</sub> (V <sub>PIN</sub> < 0 or V <sub>PIN</sub> > V <sub>DD</sub> ) .....	±20 mA
Total power dissipation <sup>(2)</sup> .....	800 mW

**Note 1:** Maximum current rating requires even load distribution across I/O pins. Maximum current rating may be limited by the device package power dissipation characterizations, see [Table 37-6](#) to calculate device specifications.

**2:** Power dissipation is calculated as follows:

$$P_{DIS} = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD} - V_{OH}) \times I_{OH}\} + \sum (V_{OI} \times I_{OL})$$

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

## 37.2 Standard Operating Conditions

The standard operating conditions for any device are defined as:

Operating Voltage:  $V_{DDMIN} \leq V_{DD} \leq V_{DDMAX}$

Operating Temperature:  $T_{A\_MIN} \leq T_A \leq T_{A\_MAX}$

### V<sub>DD</sub> — Operating Supply Voltage<sup>(1)</sup>

PIC16LF15354/55

V<sub>DDMIN</sub> (F<sub>osc</sub> ≤ 16 MHz) ..... +1.8V

V<sub>DDMIN</sub> (F<sub>osc</sub> ≤ 32 MHz) ..... +2.5V

V<sub>DDMAX</sub> ..... +3.6V

PIC16F15354/55

V<sub>DDMIN</sub> (F<sub>osc</sub> ≤ 16 MHz) ..... +2.3V

V<sub>DDMIN</sub> (F<sub>osc</sub> ≤ 32 MHz) ..... +2.5V

V<sub>DDMAX</sub> ..... +5.5V

### T<sub>A</sub> — Operating Ambient Temperature Range

Industrial Temperature

T<sub>A\\_MIN</sub> ..... -40°C

T<sub>A\\_MAX</sub> ..... +85°C

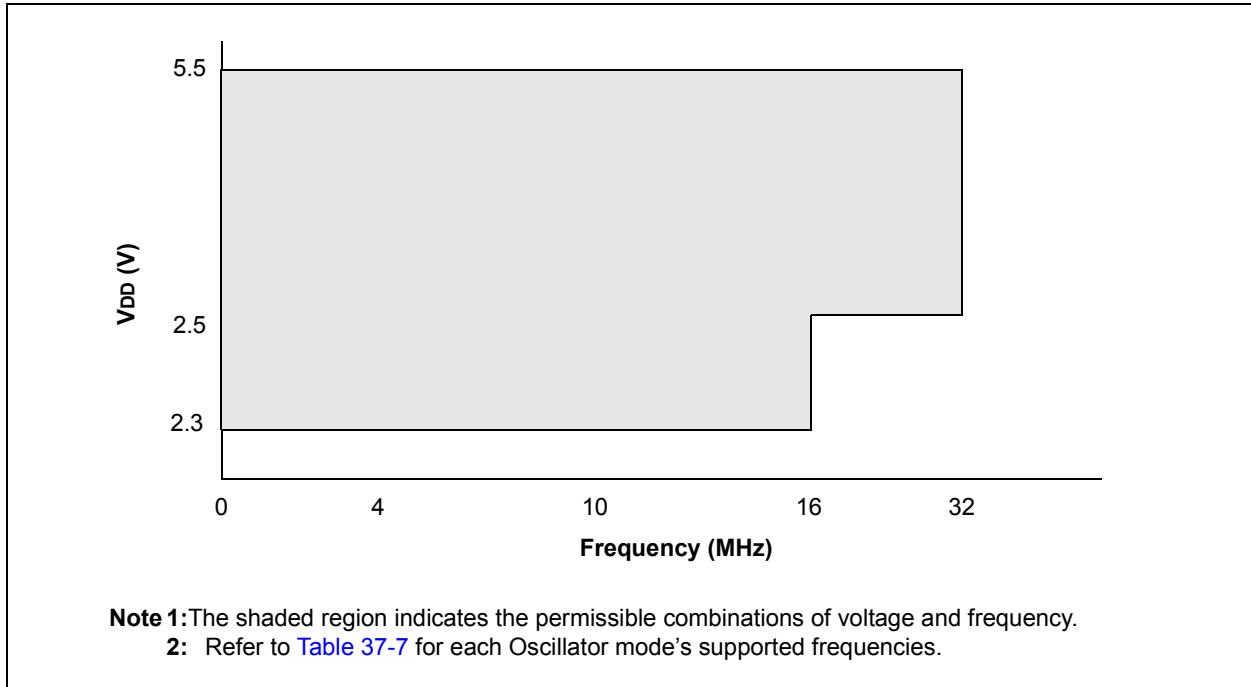
Extended Temperature

T<sub>A\\_MIN</sub> ..... -40°C

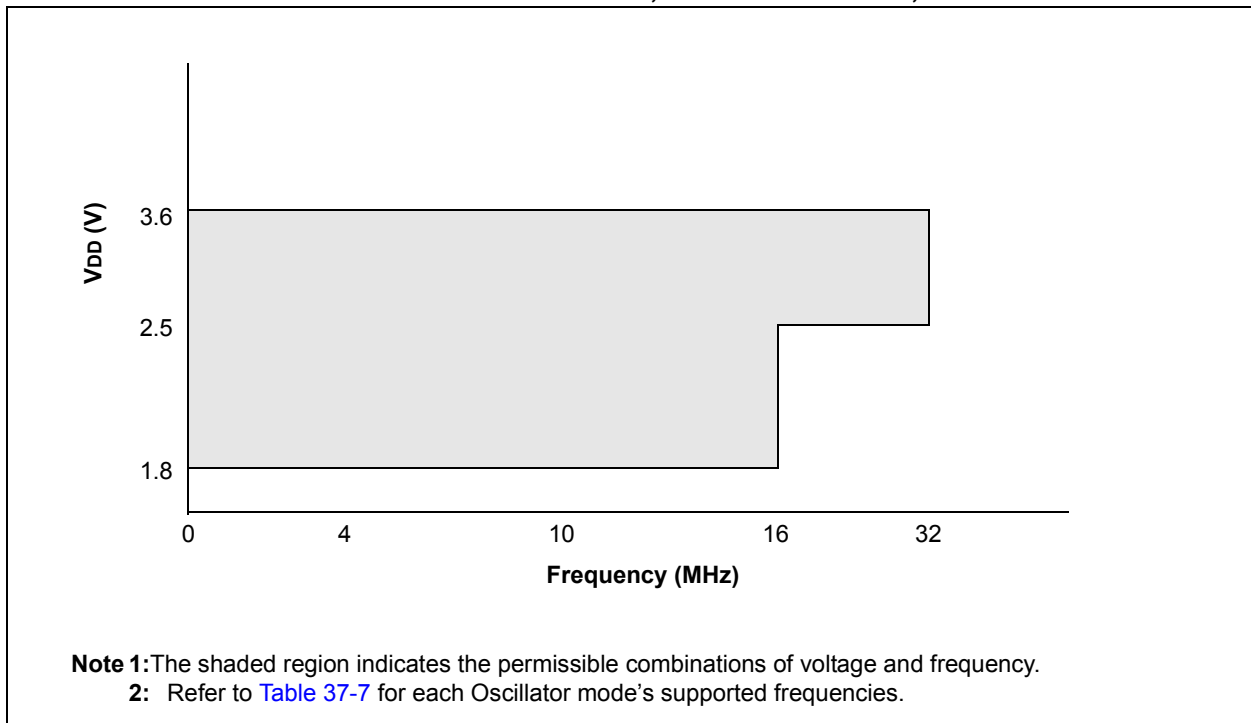
T<sub>A\\_MAX</sub> ..... +125°C

**Note 1:** See Parameter [Supply Voltage](#), DS Characteristics: Supply Voltage.

**FIGURE 37-1: VOLTAGE FREQUENCY GRAPH,  $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ , PIC16F15354/55 ONLY**



**FIGURE 37-2: VOLTAGE FREQUENCY GRAPH,  $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ , PIC16LF15354/55 ONLY**



## 37.3 DC Characteristics

**TABLE 37-1: SUPPLY VOLTAGE**

PIC16LF15354/55		Standard Operating Conditions (unless otherwise stated)					
PIC16F15354/55		Standard Operating Conditions (unless otherwise stated)					
Param. No.	Sym.	Characteristic	Min.	Typ.†	Max.	Units	Conditions
<b>Supply Voltage</b>							
D002	VDD		1.8	—	3.6	V	Fosc ≤ 16 MHz
			2.5	—	3.6	V	Fosc > 16 MHz
D002	VDD		2.3	—	5.5	V	Fosc ≤ 16 MHz
			2.5	—	5.5	V	Fosc > 16 MHz
<b>RAM Data Retention<sup>(1)</sup></b>							
D003	VDR		1.5	—	—	V	Device in Sleep mode
D003	VDR		1.7	—	—	V	Device in Sleep mode
<b>Power-on Reset Release Voltage<sup>(2)</sup></b>							
D004	VPOR		—	1.6	—	V	BOR or LPBOR disabled <sup>(3)</sup>
D004	VPOR		—	1.6	—	V	BOR or LPBOR disabled <sup>(3)</sup>
<b>Power-on Reset Rearm Voltage<sup>(2)</sup></b>							
D005	VPORR		—	0.8	—	V	BOR or LPBOR disabled <sup>(3)</sup>
D005	VPORR		—	1.5	—	V	BOR or LPBOR disabled <sup>(3)</sup>
<b>VDD Rise Rate to ensure internal Power-on Reset signal<sup>(2)</sup></b>							
D006	SVDD		0.05	—	—	V	BOR or LPBOR disabled <sup>(3)</sup>
D006	SVDD		0.05	—	—	V	BOR or LPBOR disabled <sup>(3)</sup>

† Data in "Typ." column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

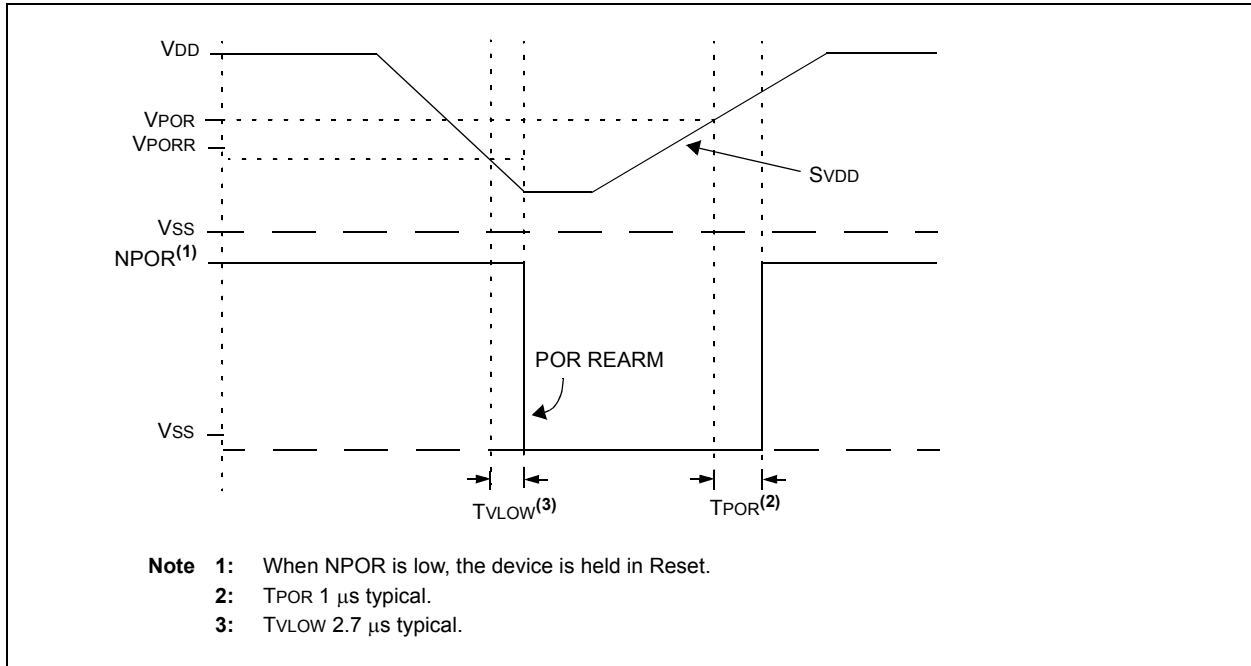
**Note 1:** This is the limit to which VDD can be lowered in Sleep mode without losing RAM data.

**2:** See [Figure 37-3](#), POR and POR REARM with Slow Rising VDD.

**3:** See [Table 37-11](#) for BOR and LPBOR trip point information.

**4:** ■ = F device

**FIGURE 37-3: POR AND POR REARM WITH SLOW RISING V<sub>DD</sub>**



# PIC16(L)F15354/55

**TABLE 37-2: SUPPLY CURRENT (IDD)<sup>(1,2,4)</sup>**

PIC16LF15354/55			Standard Operating Conditions (unless otherwise stated)					
PIC16F15354/55								
Param. No.	Symbol	Device Characteristics	Min.	Typ.†	Max.	Units	Conditions	
							VDD	Note
D100	IDD <sub>XT4</sub>	XT = 4 MHz	—	360	470	μA	3.0V	
D100	IDD <sub>XT4</sub>	XT = 4 MHz	—	380	480	μA	3.0V	
D101	IDD <sub>HFO16</sub>	HFINTOSC = 16 MHz	—	1.4	2.3	mA	3.0V	
D101	IDD <sub>HFO16</sub>	HFINTOSC = 16 MHz	—	1.5	2.3	mA	3.0V	
D102	IDD <sub>HFOPLL</sub>	HFINTOSC = 32 MHz	—	2.6	3.6	mA	3.0V	32 MHz PIC16
D102	IDD <sub>HFOPLL</sub>	HFINTOSC = 32 MHz	—	2.7	3.7	mA	3.0V	32 MHz PIC16
D103	IDD <sub>HSPLL32</sub>	HS+PLL = 32 MHz	—	2.6	3.6	mA	3.0V	32 MHz PIC16
D103	IDD <sub>HSPLL32</sub>	HS+PLL = 32 MHz	—	2.7	3.7	mA	3.0V	32 MHz PIC16
D104	IDD <sub>IDLE</sub>	IDLE mode, HFINTOSC = 16 MHz	—	—	—	mA	3.0V	
D104	IDD <sub>IDLE</sub>	IDLE mode, HFINTOSC = 16 MHz	—	—	—	mA	3.0V	
D105	IDD <sub>DOZE</sub> <sup>(3)</sup>	DOZE mode, HFINTOSC = 16 MHz, Doze Ratio = 16	—	—	—	mA	3.0V	Typical value only.
D105	IDD <sub>DOZE</sub> <sup>(3)</sup>	DOZE mode, HFINTOSC = 16 MHz, Doze Ratio = 16	—	—	—	mA	3.0V	Typical value only.

† Data in “Typ.” column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

- Note**
- 1: The test conditions for all IDD measurements in active operation mode are: OSC1 = external square wave, from rail-to-rail; all I/O pins are outputs driven low; MCLR = VDD; WDT disabled.
  - 2: The supply current is mainly a function of the operating voltage and frequency. Other factors, such as I/O pin loading and switching rate, oscillator type, internal code execution pattern and temperature, also have an impact on the current consumption.
  - 3:  $IDD_{DOZE} = [IDD_{IDLE} * (N-1)/N] + IDD_{HFO} / N$  where N = DOZE Ratio ([Register 11-2](#)).
  - 4: PMD bits are all in the default state, no modules are disabled.
  - 5:  = F device

# PIC16(L)F15354/55

**TABLE 37-3: POWER-DOWN CURRENT (IPD)<sup>(1,2,3)</sup>**

PIC16LF15354/55				Standard Operating Conditions (unless otherwise stated)					
PIC16F15354/55				Standard Operating Conditions (unless otherwise stated) VREGPM = 1					
Param. No.	Symbol	Device Characteristics	Min.	Typ.†	Max. +85°C	Max. +125°C	Units	Conditions	
								VDD	Note
D200	IPD	IPD Base	—	0.05	2	6	μA	3.0V	
D200	IPD	IPD Base	—	0.4	2.5	9	μA	3.0V	
D200A			—	18	22	27	μA	3.0V	VREGPM = 0
D201	IPD_WDT	Low-Frequency Internal Oscillator/WDT	—	0.4	2.9	9	μA	3.0V	
D201	IPD_WDT	Low-Frequency Internal Oscillator/WDT	—	0.5	3.3	13	μA	3.0V	
D202	IPD_SOSC	Secondary Oscillator (SOSC)	—	0.6	2.8	13	μA	3.0V	
D202	IPD_SOSC	Secondary Oscillator (SOSC)	—	0.8	3.2	15	μA	3.0V	
D203	IPD_FVR	FVR	—	33	47	47	μA	3.0V	
D203	IPD_FVR	FVR	—	28	44	44	μA	3.0V	
D204	IPD_BOR	Brown-out Reset (BOR)	—	10	17	19	μA	3.0V	
D204	IPD_BOR	Brown-out Reset (BOR)	—	14	18	20	μA	3.0V	
D205	IPD_LPBOR	Low-Power Brown-out Reset (LPBOR)	—	0.5	4	10	μA	3.0V	
D205	IPD_LPBOR	Low-Power Brown-out Reset (LPBOR)	—	0.7	5	11	μA	3.0V	
D206	IPD_ADCA	ADC - Active	—	250	—	—	μA	3.0V	ADC is converting <sup>(4)</sup>
D206	IPD_ADCA	ADC - Active	—	280	—	—	μA	3.0V	ADC is converting <sup>(4)</sup>
D207	IPD_CMP	Comparator	—	30	90	93	μA	3.0V	
D207	IPD_CMP	Comparator	—	33	93	98	μA	3.0V	

† Data in "Typ." column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

- Note**
- 1: The peripheral current is the sum of the base IDD and the additional current consumed when this peripheral is enabled. The peripheral Δ current can be determined by subtracting the base IDD or IPD current from this limit. Max. values should be used when calculating total current consumption.
  - 2: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode with all I/O pins in high-impedance state and tied to VSS.
  - 3: All peripheral currents listed are on a per-peripheral basis if more than one instance of a peripheral is available.
  - 4: ADC clock source is FRC.
  - 5:  = F device



**TABLE 37-4: I/O PORTS**

Standard Operating Conditions (unless otherwise stated)								
Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions	
D300 D301 D302 D303 D304 D305	V <sub>IL</sub>	<b>Input Low Voltage</b>						
		I/O PORT:						
		with TTL buffer	—	—	0.8	V	4.5V ≤ V <sub>DD</sub> ≤ 5.5V	
		with Schmitt Trigger buffer	—	—	0.15 V <sub>DD</sub>	V	1.8V ≤ V <sub>DD</sub> ≤ 4.5V	
		with I <sup>2</sup> C levels	—	—	0.2 V <sub>DD</sub>	V	2.0V ≤ V <sub>DD</sub> ≤ 5.5V	
		with SMBus levels	—	—	0.8	V	2.7V ≤ V <sub>DD</sub> ≤ 5.5V	
		MCLR	—	—	0.2 V <sub>DD</sub>	V		
D320 D321 D322 D323 D324 D325	V <sub>IH</sub>	<b>Input High Voltage</b>						
		I/O PORT:						
		with TTL buffer	2	—	—	V	4.5V ≤ V <sub>DD</sub> ≤ 5.5V	
		with Schmitt Trigger buffer	0.25 V <sub>DD</sub> + 0.8	—	—	V	1.8V ≤ V <sub>DD</sub> ≤ 4.5V	
		with I <sup>2</sup> C levels	0.8 V <sub>DD</sub>	—	—	V	2.0V ≤ V <sub>DD</sub> ≤ 5.5V	
		with SMBus levels	0.7 V <sub>DD</sub>	—	—	V		
		MCLR	2.1	—	—	V	2.7V ≤ V <sub>DD</sub> ≤ 5.5V	
		MCLR	0.7 V <sub>DD</sub>	—	—	V		
D340 D341 D342	I <sub>IL</sub>	<b>Input Leakage Current<sup>(1)</sup></b>						
		I/O Ports	—	± 5	± 125	nA	V <sub>SS</sub> ≤ V <sub>PIN</sub> ≤ V <sub>DD</sub> , Pin at high-impedance, 85°C	
			—	± 5	± 1000	nA	V <sub>SS</sub> ≤ V <sub>PIN</sub> ≤ V <sub>DD</sub> , Pin at high-impedance, 125°C	
		MCLR <sup>(2)</sup>	—	± 50	± 200	nA	V <sub>SS</sub> ≤ V <sub>PIN</sub> ≤ V <sub>DD</sub> , Pin at high-impedance, 85°C	
D350	I <sub>PUR</sub>	<b>Weak Pull-up Current</b>		25	100	200	μA	V <sub>DD</sub> = 3.0V, V <sub>PIN</sub> = V <sub>SS</sub>
D360	V <sub>OL</sub>	<b>Output Low Voltage</b>						
		I/O ports	—	—	0.6	V	I <sub>OL</sub> = 10.0 mA, V <sub>DD</sub> = 3.0V	
D370	V <sub>OH</sub>	<b>Output High Voltage</b>						
		I/O ports	V <sub>DD</sub> - 0.7	—	—	V	I <sub>OH</sub> = 6.0 mA, V <sub>DD</sub> = 3.0V	
D380	C <sub>IO</sub>	<b>All I/O pins</b>	—	5	50	pF		

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note 1:** Negative current is defined as current sourced by the pin.

**Note 2:** The leakage current on the MCLR pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

**TABLE 37-5: MEMORY PROGRAMMING SPECIFICATIONS**

Standard Operating Conditions (unless otherwise stated)							
Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions
<b>High Voltage Entry Programming Mode Specifications</b>							
MEM01	V <sub>IHH</sub>	Voltage on $\overline{\text{MCLR}}/\text{VPP}$ pin to enter programming mode	8	—	9	V	(Note 2, Note 3)
MEM02	I <sub>PPGM</sub>	Current on $\overline{\text{MCLR}}/\text{VPP}$ pin during programming mode	—	1	—	mA	(Note 2)
<b>Programming Mode Specifications</b>							
MEM10	V <sub>BE</sub>	VDD for Bulk Erase	—	2.7	—	V	
MEM11	I <sub>DDPGM</sub>	Supply Current during Programming operation	—	—	10	mA	
<b>Program Flash Memory Specifications</b>							
MEM30	E <sub>P</sub>	Flash Memory Cell Endurance	10k	—	—	E/W	-40°C ≤ T <sub>A</sub> ≤ +85°C (Note 1)
MEM32	T <sub>P_RET</sub>	Characteristic Retention	—	40	—	Year	Provided no other specifications are violated
MEM33	V <sub>P_RD</sub>	VDD for Read operation	VDDMIN	—	VDDMAX	V	
MEM34	V <sub>P_REW</sub>	VDD for Row Erase or Write operation	VDDMIN	—	VDDMAX	V	
MEM35	T <sub>P_REW</sub>	Self-Timed Row Erase or Self-Timed Write	—	2.0	2.5	ms	

† 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:** Flash Memory Cell Endurance for the Flash memory is defined as: One Row Erase operation and one Self-Timed Write. HEF feature applies only to the last 128 words of the Program Flash Memory.
- 2:** Required only if CONFIG4, bit LVP is disabled.
- 3:** The MPLAB<sup>®</sup> ICD2 does not support variable VPP output. Circuitry to limit the ICD2 VPP voltage must be placed between the ICD2 and target system when programming or debugging with the ICD2.

**TABLE 37-6: THERMAL CHARACTERISTICS**

**Standard Operating Conditions** (unless otherwise stated)  
**Operating Temperature**  $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$

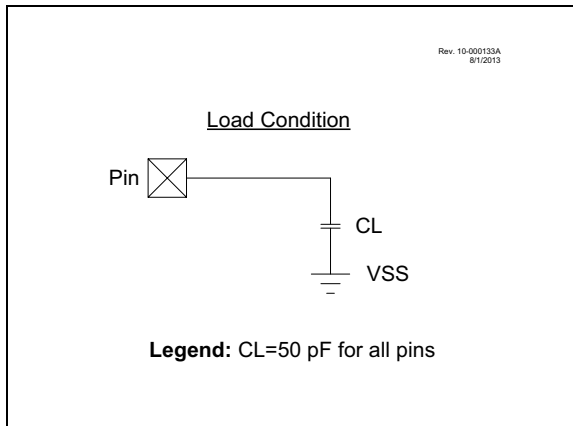
Param. No.	Sym.	Characteristic	Typ.	Units	Conditions
TH01	$\theta_{JA}$	Thermal Resistance Junction to Ambient	60	$^{\circ}\text{C}/\text{W}$	28-pin SPDIP package
			80	$^{\circ}\text{C}/\text{W}$	28-pin SOIC package
			90	$^{\circ}\text{C}/\text{W}$	28-pin SSOP package
			48	$^{\circ}\text{C}/\text{W}$	28-pin UQFN 4x4 mm package
TH02	$\theta_{JC}$	Thermal Resistance Junction to Case	31.4	$^{\circ}\text{C}/\text{W}$	28-pin SPDIP package
			24	$^{\circ}\text{C}/\text{W}$	28-pin SOIC package
			24	$^{\circ}\text{C}/\text{W}$	28-pin SSOP package
			12	$^{\circ}\text{C}/\text{W}$	28-pin UQFN 4x4 mm package
TH03	$T_{JMAX}$	Maximum Junction Temperature	150	$^{\circ}\text{C}$	
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	$P_{I/O} = \sum (I_{OL} \times V_{OL}) + \sum (I_{OH} \times (V_{DD} - V_{OH}))$
TH07	PDER	Derated Power	—	W	$P_{DER} = P_{D_{MAX}} (T_J - T_A) / \theta_{JA}^{(2)}$

**Note 1:**  $I_{DD}$  is current to run the chip alone without driving any load on the output pins.

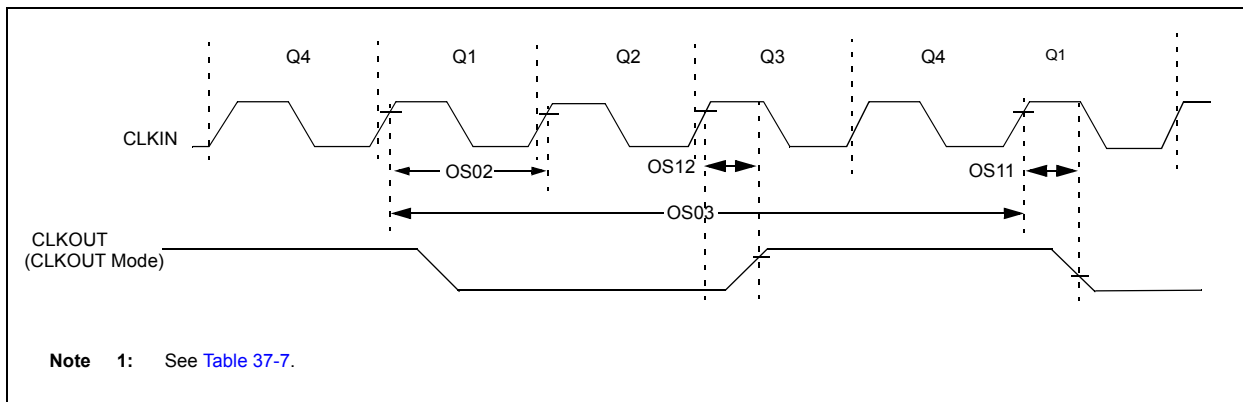
**2:**  $T_A$  = Ambient Temperature,  $T_J$  = Junction Temperature

## 37.4 AC Characteristics

**FIGURE 37-4: LOAD CONDITIONS**



**FIGURE 37-5: CLOCK TIMING**



**TABLE 37-7: EXTERNAL CLOCK/OSCILLATOR TIMING REQUIREMENTS**

Standard Operating Conditions (unless otherwise stated)							
Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions
<b>ECL Oscillator</b>							
OS1	$F_{ECL}$	Clock Frequency	—	—	500	kHz	
OS2	$T_{ECL\_DC}$	Clock Duty Cycle	40	—	60	%	
<b>ECM Oscillator</b>							
OS3	$F_{ECM}$	Clock Frequency	—	—	4	MHz	
OS4	$T_{ECM\_DC}$	Clock Duty Cycle	40	—	60	%	
<b>ECH Oscillator</b>							
OS5	$F_{ECH}$	Clock Frequency	—	—	32	MHz	
OS6	$T_{ECH\_DC}$	Clock Duty Cycle	40	—	60	%	
<b>LP Oscillator</b>							
OS7	$F_{LP}$	Clock Frequency	—	—	100	kHz	<b>Note 4</b>
<b>XT Oscillator</b>							
OS8	$F_{XT}$	Clock Frequency	—	—	4	MHz	<b>Note 4</b>
<b>HS Oscillator</b>							
OS9	$F_{HS}$	Clock Frequency	—	—	20	MHz	
<b>System Oscillator</b>							
OS20	$F_{OSC}$	System Clock Frequency	—	—	32	MHz	<b>(Note 2, Note 3)</b>
OS21	$F_{CY}$	Instruction Frequency	—	$F_{OSC}/4$	—	MHz	
OS22	$T_{CY}$	Instruction Period	125	$1/F_{CY}$	—	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.

- Note 1:** Instruction cycle period ( $T_{CY}$ ) 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.
- 2:** The system clock frequency ( $F_{OSC}$ ) is selected by the "main clock switch controls" as described in [Section 9.0 "Oscillator Module \(with Fail-Safe Clock Monitor\)"](#).
- 3:** The system clock frequency ( $F_{OSC}$ ) must meet the voltage requirements defined in the [Section 37.2 "Standard Operating Conditions"](#).
- 4:** LP, XT and HS oscillator modes require an appropriate crystal or resonator to be connected to the device. For clocking the device with the external square wave, one of the EC mode selections must be used.

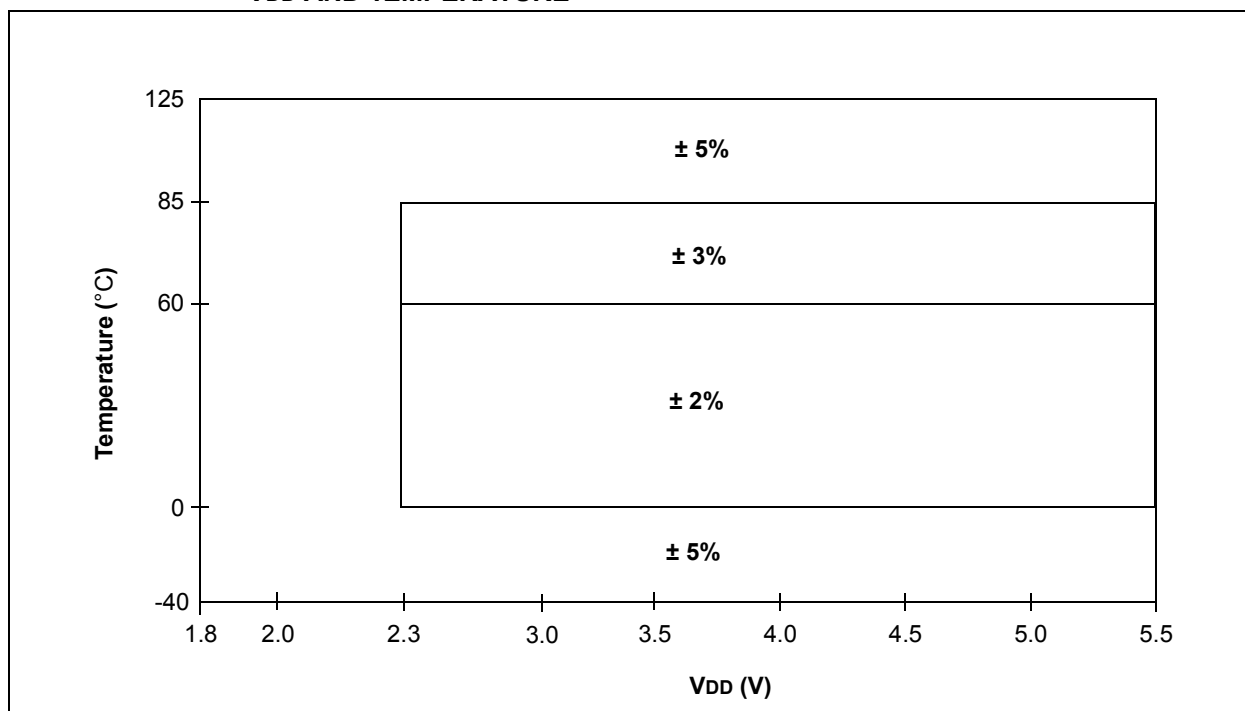
**TABLE 37-8: INTERNAL OSCILLATOR PARAMETERS<sup>(1)</sup>**

Standard Operating Conditions (unless otherwise stated)							
Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions
OS50	FHFOSC	Precision Calibrated HFINTOSC Frequency	—	4 8 12 16 32	—	MHz	(Note 2)
OS51	FHFOSCLP	Low-Power Optimized HFINTOSC Frequency	0.93 1.86	1 2	1.07 2.14	MHz MHz	
OS52	FMFOSC	Internal Calibrated MFINTOSC Frequency	—	500	—	kHz	(Note 3)
OS53	FLFOSC	Internal LFINTOSC Frequency	—	31	—	kHz	
OS54	THFOSCST	HFINTOSC Wake-up from Sleep Start-up Time	— —	11 50	20 —	μs μs	VREGPM = 0 VREGPM = 1
OS56	TLFOSCST	LFINTOSC Wake-up from Sleep Start-up Time	—	0.2	—	ms	

† Data in “Typ” column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

- Note 1:** To ensure these oscillator frequency tolerances, V<sub>DD</sub> and V<sub>SS</sub> must be capacitively decoupled as close to the device as possible. 0.1 μF and 0.01 μF values in parallel are recommended.
- 2:** See [Figure 37-6: Precision Calibrated HFINTOSC Frequency Accuracy Over Device V<sub>DD</sub> and Temperature](#).

**FIGURE 37-6: PRECISION CALIBRATED HFINTOSC FREQUENCY ACCURACY OVER DEVICE V<sub>DD</sub> AND TEMPERATURE**



**TABLE 37-9: PLL SPECIFICATIONS**

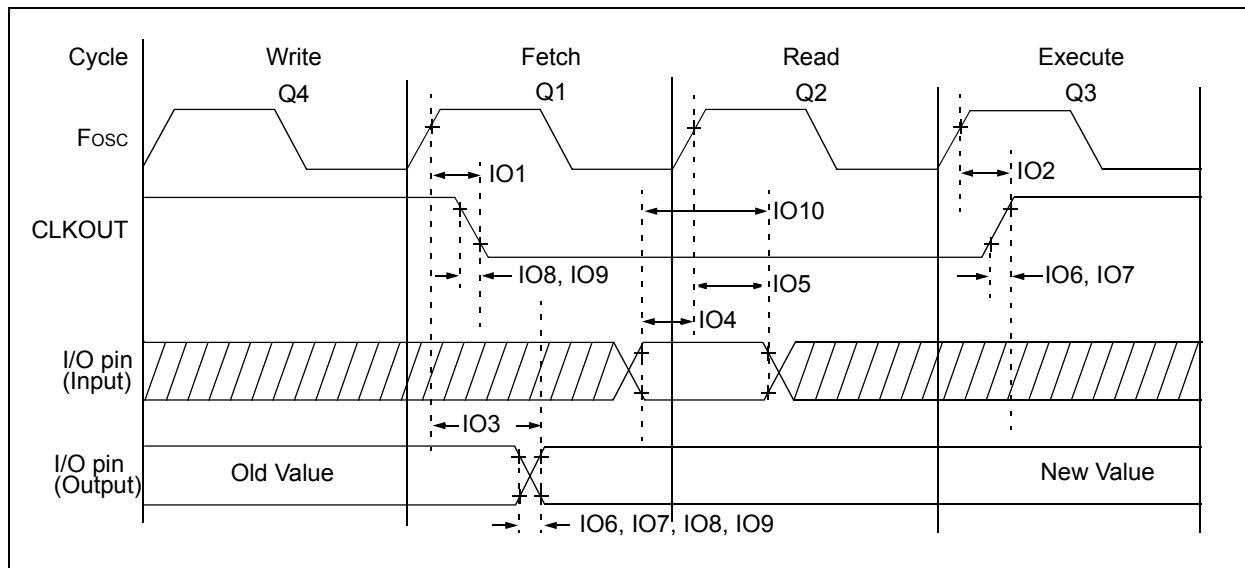
Standard Operating Conditions (unless otherwise stated) $V_{DD} \geq 2.5V$							
Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions
PLL01	FPLLIN	PLL Input Frequency Range	4	—	8	MHz	
PLL02	FPLLOUT	PLL Output Frequency Range	16	—	32	MHz	<b>Note 1</b>
PLL03	TPLLST	PLL Lock Time from Start-up	—	200	—	$\mu s$	
PLL04	FPLLJIT	PLL Output Frequency 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.

**Note 1:** The output frequency of the PLL must meet the FOSC requirements listed in Parameter D002.

**FIGURE 37-7: CLKOUT AND I/O TIMING**



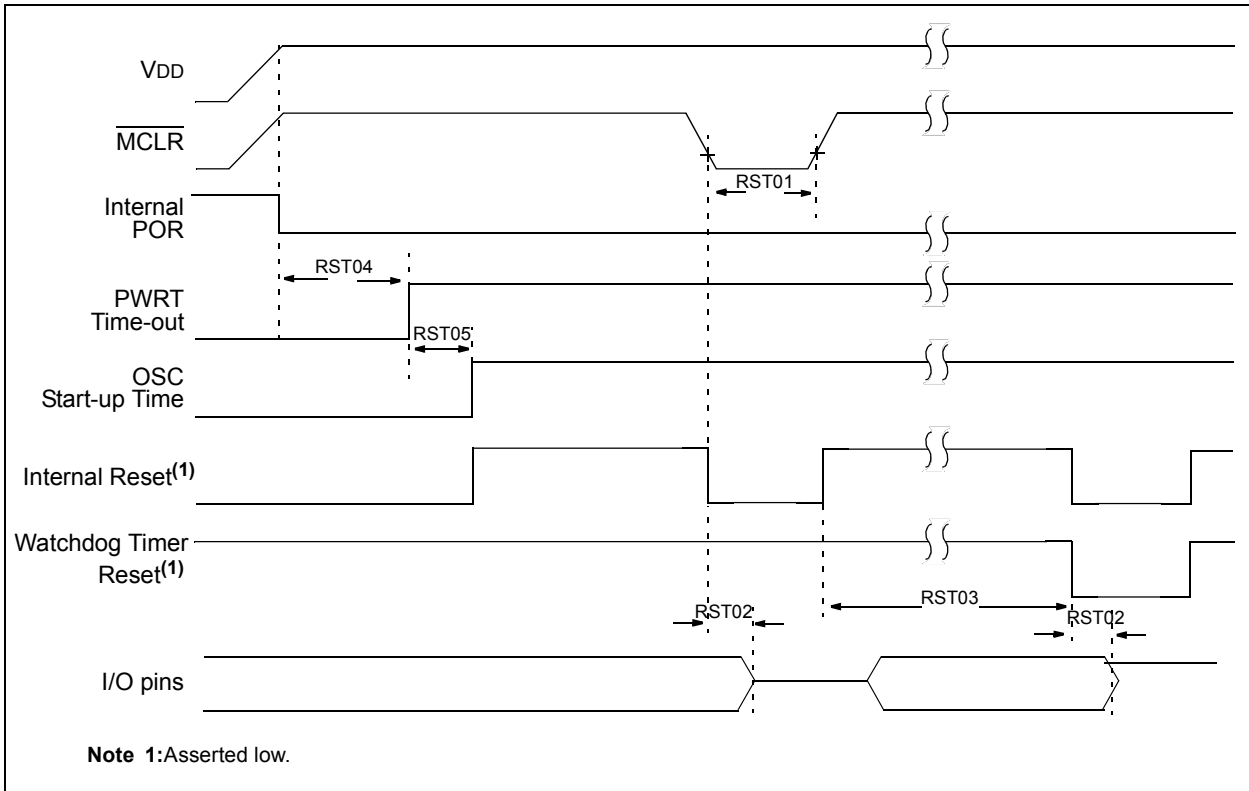
**TABLE 37-10: I/O AND CLKOUT TIMING SPECIFICATIONS**

Standard Operating Conditions (unless otherwise stated)							
Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions
IO1*	$T_{CLKOUTH}$	CLKOUT rising edge delay (rising edge Fosc (Q1 cycle) to falling edge CLKOUT)	—	—	70	ns	
IO2*	$T_{CLKOUTL}$	CLKOUT falling edge delay (rising edge Fosc (Q3 cycle) to rising edge CLKOUT)	—	—	72	ns	
IO3*	$T_{IO\_VALID}$	Port output valid time (rising edge Fosc (Q1 cycle) to port valid)	—	50	70	ns	
IO4*	$T_{IO\_SETUP}$	Port input setup time (Setup time before rising edge Fosc – Q2 cycle)	20	—	—	ns	
IO5*	$T_{IO\_HOLD}$	Port input hold time (Hold time after rising edge Fosc – Q2 cycle)	50	—	—	ns	
IO6*	$T_{IOR\_SLREN}$	Port I/O rise time, slew rate enabled	—	25	—	ns	$V_{DD} = 3.0V$
IO7*	$T_{IOR\_SLRDIS}$	Port I/O rise time, slew rate disabled	—	5	—	ns	$V_{DD} = 3.0V$
IO8*	$T_{IOF\_SLREN}$	Port I/O fall time, slew rate enabled	—	25	—	ns	$V_{DD} = 3.0V$
IO9*	$T_{IOF\_SLRDIS}$	Port I/O fall time, slew rate disabled	—	5	—	ns	$V_{DD} = 3.0V$
IO10*	$T_{INT}$	INT pin high or low time to trigger an interrupt	25	—	—	ns	
IO11*	$T_{IOC}$	Interrupt-on-Change minimum high or low time to trigger interrupt	25	—	—	ns	

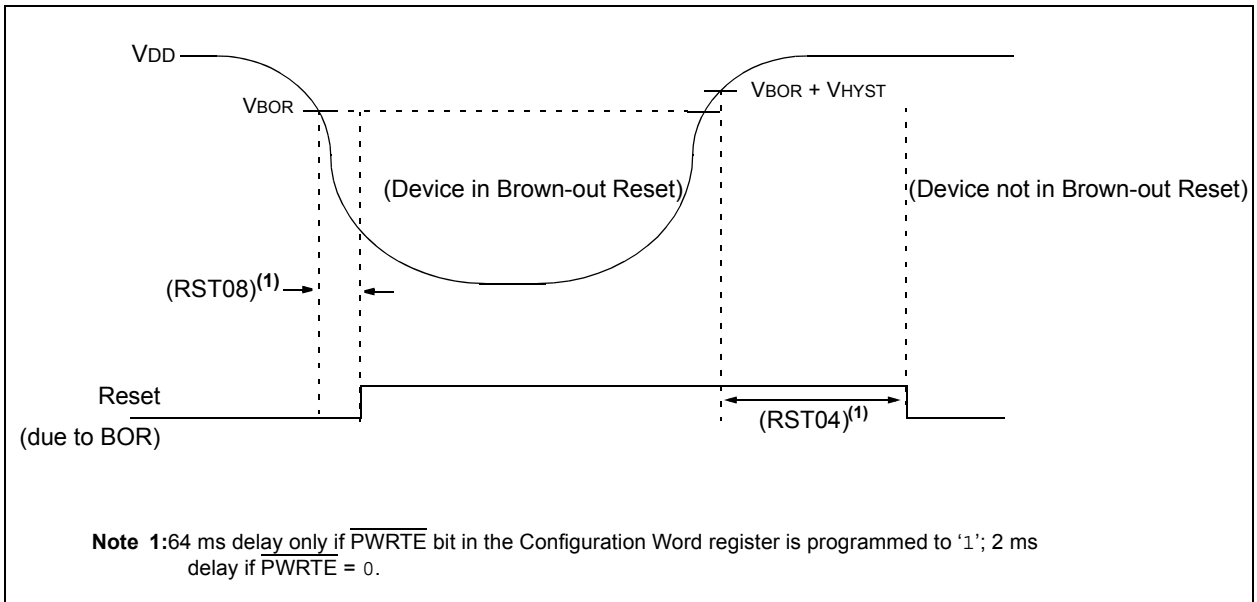
\*These parameters are characterized but not tested.



**FIGURE 37-8: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING**



**FIGURE 37-9: BROWN-OUT RESET TIMING AND CHARACTERISTICS**



**TABLE 37-11: RESET, WDT, OSCILLATOR START-UP TIMER, POWER-UP TIMER, BROWN-OUT RESET AND LOW-POWER BROWN-OUT RESET SPECIFICATIONS**

Standard Operating Conditions (unless otherwise stated)							
Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions
RST01*	TMCLR	MCLR Pulse Width Low to ensure Reset	2	—	—	μs	
RST02*	TIOZ	I/O high-impedance from Reset detection	—	—	2	μs	
RST03	TWDT	Watchdog Timer Time-out Period	—	16	—	ms	16 ms Nominal Reset Time
RST04*	TPWRT	Power-up Timer Period	—	65	—	ms	
RST05	TOST	Oscillator Start-up Timer Period <sup>(1,2)</sup>	—	1024	—	T <sub>OSC</sub>	
RST06	VBOR	Brown-out Reset Voltage	2.55 2.30 1.80	2.70 2.45 1.90	2.85 2.60 2.10	V	BORV = 0 BORV = 1 (F devices) BORV = 1 (LF devices)
RST07	VBORHYS	Brown-out Reset Hysteresis	—	40	—	mV	
RST08	TBORDC	Brown-out Reset Response Time	—	3	—	μs	
RST09	VLPBOR	Low-Power Brown-out Reset Voltage	1.8	2.0	2.2	V	LF Devices Only

\* 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.

**Note 2:** To ensure these voltage tolerances, V<sub>DD</sub> and V<sub>SS</sub> must be capacitively decoupled as close to the device as possible. 0.1 μF and 0.01 μF values in parallel are recommended.

**TABLE 37-12: ANALOG-TO-DIGITAL CONVERTER (ADC) ACCURACY SPECIFICATIONS<sup>(1,2)</sup>:**

Standard Operating Conditions (unless otherwise stated)							
V <sub>DD</sub> = 3.0V, T <sub>A</sub> = 25°C							
Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions
AD01	NR	Resolution	—	—	10	bit	
AD02	EIL	Integral Error	—	±0.1	±1.0	LSb	ADCREf+ = 3.0V, ADCREf- = 0V
AD03	EDL	Differential Error	—	±0.1	±1.0	LSb	ADCREf+ = 3.0V, ADCREf- = 0V
AD04	E <sub>OFF</sub>	Offset Error	—	0.5	2.0	LSb	ADCREf+ = 3.0V, ADCREf- = 0V
AD05	E <sub>GN</sub>	Gain Error	—	±0.2	±1.0	LSb	ADCREf+ = 3.0V, ADCREf- = 0V
AD06	V <sub>ADREF</sub>	ADC Reference Voltage (ADREF+) <sup>(3)</sup>	1.8	—	V <sub>DD</sub>	V	
AD07	V <sub>AIN</sub>	Full-Scale Range	ADREF-	—	ADREF+	V	
AD08	Z <sub>AIN</sub>	Recommended Impedance of Analog Voltage Source	—	10	—	kΩ	
AD09	R <sub>VREF</sub>	ADC Voltage Reference Ladder Impedance	—	50	—	kΩ	<b>Note 3</b>

\* 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 is the sum of the offset, gain and integral non-linearity (INL) errors.

**Note 2:** The ADC conversion result never decreases with an increase in the input and has no missing codes.

**Note 3:** This is the impedance seen by the V<sub>REF</sub> pads when the external reference pads are selected.

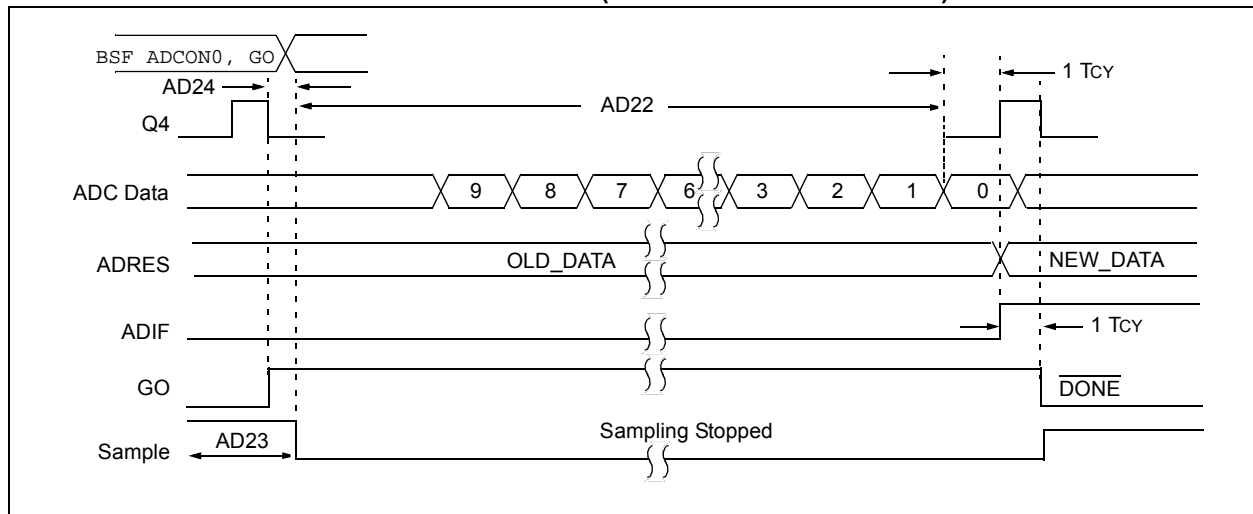
**TABLE 37-13: ANALOG-TO-DIGITAL CONVERTER (ADC) CONVERSION TIMING SPECIFICATIONS**

Standard Operating Conditions (unless otherwise stated)							
Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions
AD20	TAD	ADC Clock Period	1	—	9	μs	The requirement is to set ADCCS correctly to produce this period/frequency.
AD21			1	2	6	μs	Using FRC as the ADC clock source ADOSC = 1
AD22	TCNV	Conversion Time	—	11	—	TAD	Set of GO/DONE bit to Clear of GO/DONE bit
AD23	TACQ	Acquisition Time	—	2	—	μs	
AD24	THCD	Sample and Hold Capacitor Disconnect Time	—	—	—	μs	FOSC-based clock source FRC-based clock source

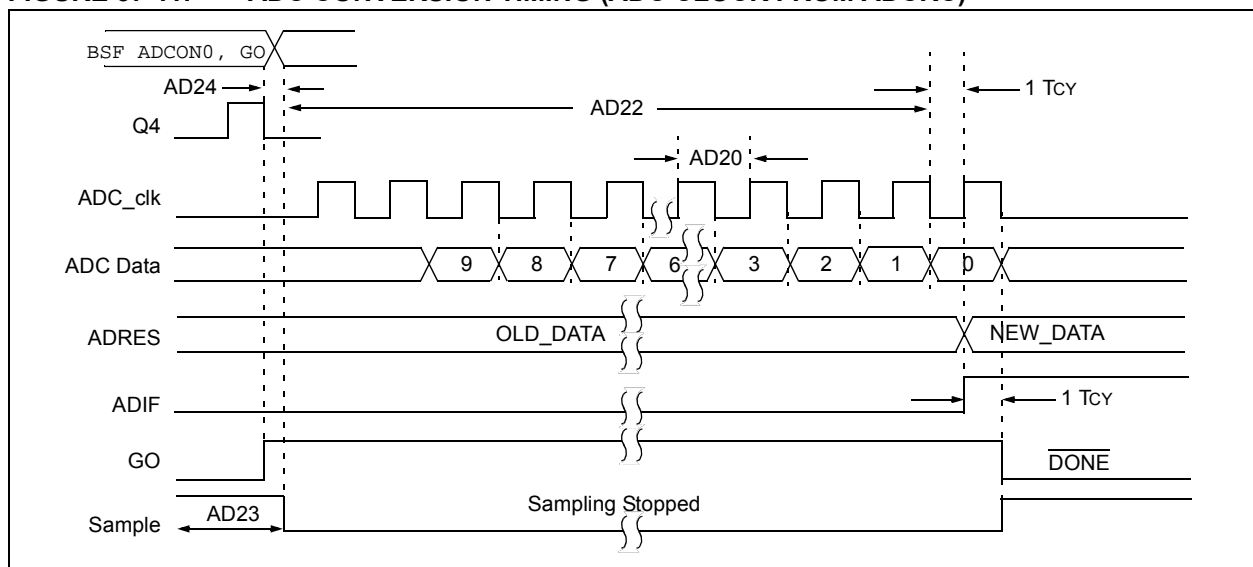
\* These parameters are characterized but not tested.

† 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 37-10: ADC CONVERSION TIMING (ADC CLOCK FOSC-BASED)**



**FIGURE 37-11: ADC CONVERSION TIMING (ADC CLOCK FROM ADCRC)**



**TABLE 37-14: COMPARATOR SPECIFICATIONS**

Standard Operating Conditions (unless otherwise stated) V <sub>DD</sub> = 3.0V, T <sub>A</sub> = 25°C							
Param. No.	Sym.	Characteristics	Min.	Typ.	Max.	Units	Comments
CM01	V <sub>IOFF</sub>	Input Offset Voltage	—	—	±50	mV	V <sub>ICM</sub> = V <sub>DD</sub> /2
CM02	V <sub>ICM</sub>	Input Common Mode Range	GND	—	V <sub>DD</sub>	V	
CM03	CMRR	Common Mode Input Rejection Ratio	—	50	—	dB	
CM04	V <sub>HYST</sub>	Comparator Hysteresis	15	25	35	mV	
CM05	T <sub>RESP</sub> <sup>(1)</sup>	Response Time, Rising Edge	—	300	600	ns	
		Response Time, Falling Edge	—	220	500	ns	
CMOS6	T <sub>MCV2VO</sub> <sup>(2)</sup>	Mode Change to Valid Output	—	—	10	µs	

\* These parameters are characterized but not tested.

**Note 1:** Response time measured with one comparator input at V<sub>DD</sub>/2, while the other input transitions from V<sub>SS</sub> to V<sub>DD</sub>.

**2:** A mode change includes changing any of the control register values, including module enable.

**TABLE 37-15: 5-BIT DAC SPECIFICATIONS**

Standard Operating Conditions (unless otherwise stated) V <sub>DD</sub> = 3.0V, T <sub>A</sub> = 25°C							
Param. No.	Sym.	Characteristics	Min.	Typ.	Max.	Units	Comments
DSB01	V <sub>LSB</sub>	Step Size	—	(V <sub>DACREF+</sub> - V <sub>DACREF-</sub> ) / 32	—	V	
DSB01	V <sub>ACC</sub>	Absolute Accuracy	—	—	± 0.5	LSb	
DSB03*	R <sub>UNIT</sub>	Unit Resistor Value	—	5000	—	Ω	
DSB04*	T <sub>ST</sub>	Settling Time <sup>(1)</sup>	—	—	10	µs	

\* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**Note 1:** Settling time measured while DACR<4:0> transitions from '00000' to '01111'.

**TABLE 37-16: FIXED VOLTAGE REFERENCE (FVR) SPECIFICATIONS**

Standard Operating Conditions (unless otherwise stated)							
Param. No.	Symbol	Characteristic	Min.	Typ.	Max.	Units	Conditions
FVR01	V <sub>FVR1</sub>	1x Gain (1.024V)	-4	—	+4	%	V <sub>DD</sub> ≥ 2.5V, -40°C to 85°C
FVR02	V <sub>FVR2</sub>	2x Gain (2.048V)	-4	—	+4	%	V <sub>DD</sub> ≥ 2.5V, -40°C to 85°C
FVR03	V <sub>FVR4</sub>	4x Gain (4.096V)	-6	—	+6	%	V <sub>DD</sub> ≥ 4.75V, -40°C to 85°C
FVR04	T <sub>FVRST</sub>	FVR Start-up Time	—	25	—	µs	
FVR05	FVRA1x/FVRC1x	FVR output voltage for 1x setting stored in the DIA	—	1024	—	mV	
FVR06	FVRA2x/FVRC2x	FVR output voltage for 2x setting stored in the DIA	—	2048	—	mV	
FVR07	FVRA4x/FVRC4x	FVR output voltage for 4x setting stored in the DIA	—	4096	—	mV	<b>Note 1</b>

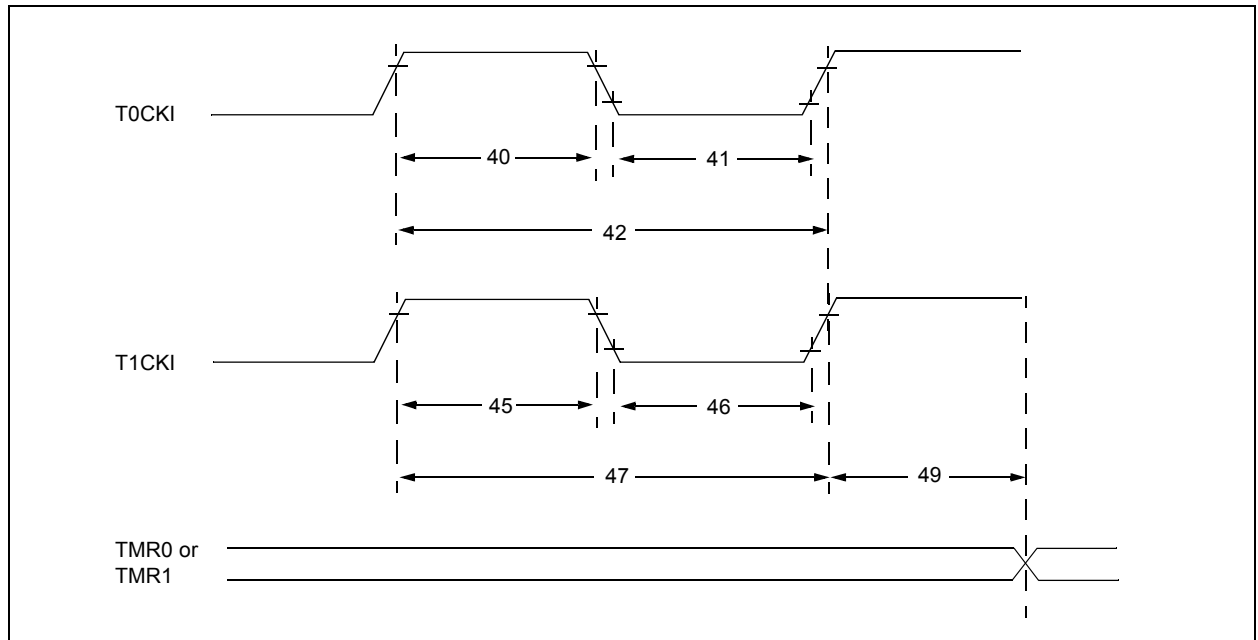
**Note 1:** Available only on PIC16F15354/55.

**TABLE 37-17: ZERO CROSS DETECT (ZCD) SPECIFICATIONS**

Standard Operating Conditions (unless otherwise stated) V <sub>DD</sub> = 3.0V, T <sub>A</sub> = 25°C							
Param. No.	Sym.	Characteristics	Min.	Typ†	Max.	Units	Comments
ZC01	ZPCINV	Voltage on Zero Cross Pin	—	0.75	—	V	
ZC02	ZCDRV	Maximum source or sink current	—	—	600	μA	
ZC04	ZCISW	Response Time, Rising Edge	—	1	—	μs	
		Response Time, Falling Edge	—	1	—	μs	
ZC05	ZCOUT	Response Time, Rising Edge	—	1	—	μs	
		Response Time, Falling Edge	—	1	—	μs	

† 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 37-12: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS**



# PIC16(L)F15354/55

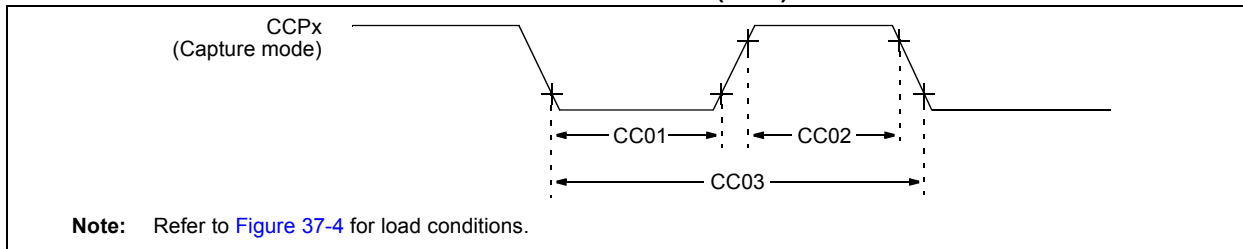
**TABLE 37-18: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS**

Standard Operating Conditions (unless otherwise stated)								
Operating Temperature $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$								
Param. No.	Sym.	Characteristic		Min.	Typ†	Max.	Units	Conditions
40*	Tt0H	T0CKI High-Pulse Width	No Prescaler	$0.5 T_{CY} + 20$	—	—	ns	
			With Prescaler	10	—	—	ns	
41*	Tt0L	T0CKI Low-Pulse Width	No Prescaler	$0.5 T_{CY} + 20$	—	—	ns	
			With Prescaler	10	—	—	ns	
42*	Tt0P	T0CKI Period		Greater of: 20 or $(T_{CY} + 40) * N$	—	—	ns	N = prescale value (2, 4,...256)
45*	Tt1H	T1CKI High Time	Synchronous, No Prescaler	$0.5 T_{CY} + 20$	—	—	ns	
			Synchronous, with Prescaler	15	—	—	ns	
			Asynchronous	30	—	—	ns	
46*	Tt1L	T1CKI Low Time	Synchronous, No Prescaler	$0.5 T_{CY} + 20$	—	—	ns	
			Synchronous, with Prescaler	15	—	—	ns	
			Asynchronous	30	—	—	ns	
47*	Tt1P	T1CKI Input Period	Synchronous	Greater of: 30 or $(T_{CY} + 40) * N$	—	—	ns	N = prescale value (2, 4,...256)
			Asynchronous	60	—	—	ns	
48	Ft1	Timer1 Oscillator Input Frequency Range (oscillator enabled by setting bit T1OSCEN)		32.4	32.768	33.1	kHz	
49*	TCKEZTMR1	Delay from External Clock Edge to Timer Increment		2 TOSC	—	7 TOSC	—	Timers in Sync mode

\* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**FIGURE 37-13: CAPTURE/COMPARE/PWM TIMINGS (CCP)**



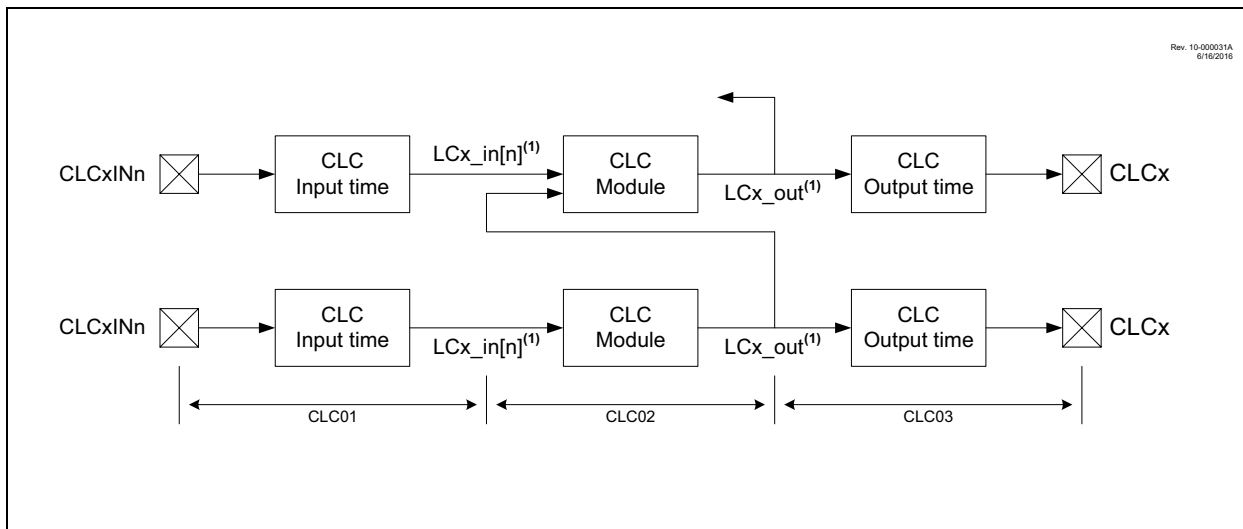
**TABLE 37-19: CAPTURE/COMPARE/PWM REQUIREMENTS (CCP)**

Standard Operating Conditions (unless otherwise stated)								
Operating Temperature $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$								
Param. No.	Sym.	Characteristic		Min.	Typ†	Max.	Units	Conditions
CC01*	TccL	CCPx Input Low Time	No Prescaler	$0.5T_{CY} + 20$	—	—	ns	
			With Prescaler	20	—	—	ns	
CC02*	TccH	CCPx Input High Time	No Prescaler	$0.5T_{CY} + 20$	—	—	ns	
			With Prescaler	20	—	—	ns	
CC03*	TccP	CCPx Input Period		$(3T_{CY} + 40) \cdot N$	—	—	ns	N = prescale value (1,4 or 16)

\* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

**FIGURE 37-14: CLC PROPAGATION TIMING**



**TABLE 37-20: CONFIGURABLE LOGIC CELL (CLC) CHARACTERISTICS**

Standard Operating Conditions (unless otherwise stated)							
Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$							
Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions
CLC01*	TCLCIN	CLC input pin (CKCxIN) to CKC Module Input select (LCx_IN) propagation time	—	7	IO5	ns	(Note 1)
CLC02*	TCLC	CLC Module input to output propagation delay	—	24 12	—	ns ns	VDD = 1.8V VDD > 3.6V
CLC03*	TCLCOUT	CLC Module output time	—	IO7	—	—	Rise Time (Note 1)
			—	IO8	—	—	Fall Time (Note 1)
CLC04*	FCLCMAX	CLC Maximum switching frequency	—	32	Fosc	MHz	

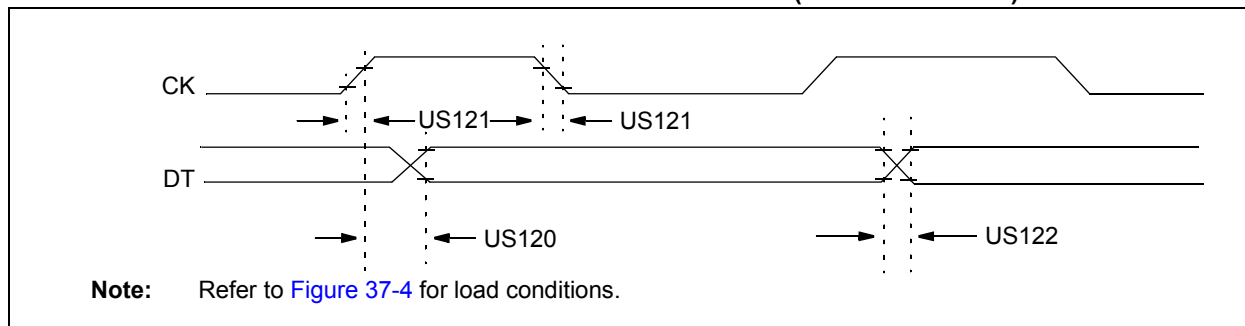
\* These parameters are characterized but not tested.

† 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 37-10 for IO5, IO7 and IO8 rise and fall times.



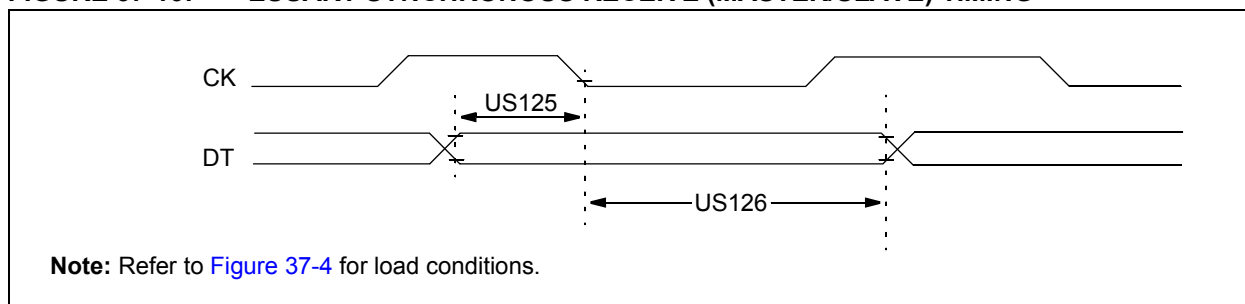
**FIGURE 37-15: EUSART SYNCHRONOUS TRANSMISSION (MASTER/SLAVE) TIMING**



**TABLE 37-21: EUSART SYNCHRONOUS TRANSMISSION CHARACTERISTICS**

Standard Operating Conditions (unless otherwise stated) Operating Temperature $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$						
Param. No.	Symbol	Characteristic	Min.	Max.	Units	Conditions
US120	TCKH2DTV	SYNC XMIT (Master and Slave) Clock high to data-out valid	—	80	ns	$3.0\text{V} \leq V_{DD} \leq 5.5\text{V}$
			—	100	ns	$1.8\text{V} \leq V_{DD} \leq 5.5\text{V}$
US121	TCKRF	Clock out rise time and fall time (Master mode)	—	45	ns	$3.0\text{V} \leq V_{DD} \leq 5.5\text{V}$
			—	50	ns	$1.8\text{V} \leq V_{DD} \leq 5.5\text{V}$
US122	TDTRF	Data-out rise time and fall time	—	45	ns	$3.0\text{V} \leq V_{DD} \leq 5.5\text{V}$
			—	50	ns	$1.8\text{V} \leq V_{DD} \leq 5.5\text{V}$

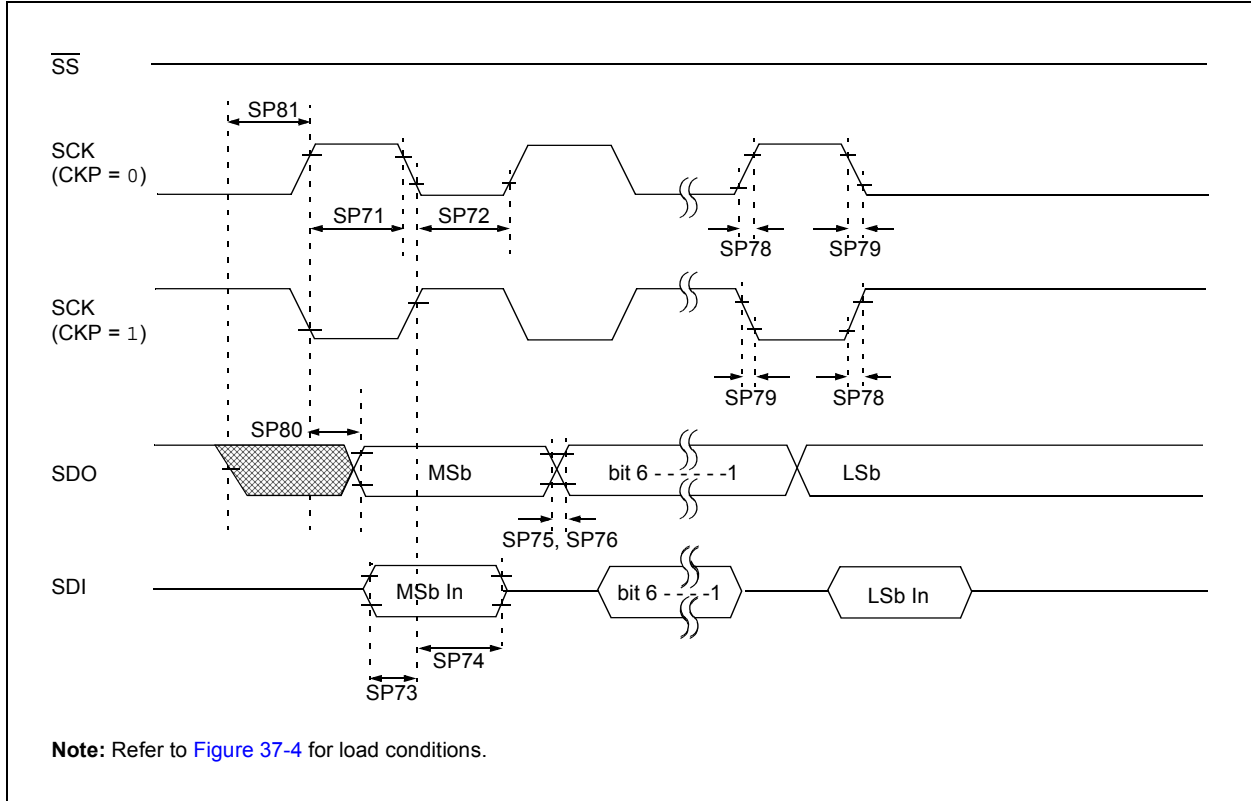
**FIGURE 37-16: EUSART SYNCHRONOUS RECEIVE (MASTER/SLAVE) TIMING**



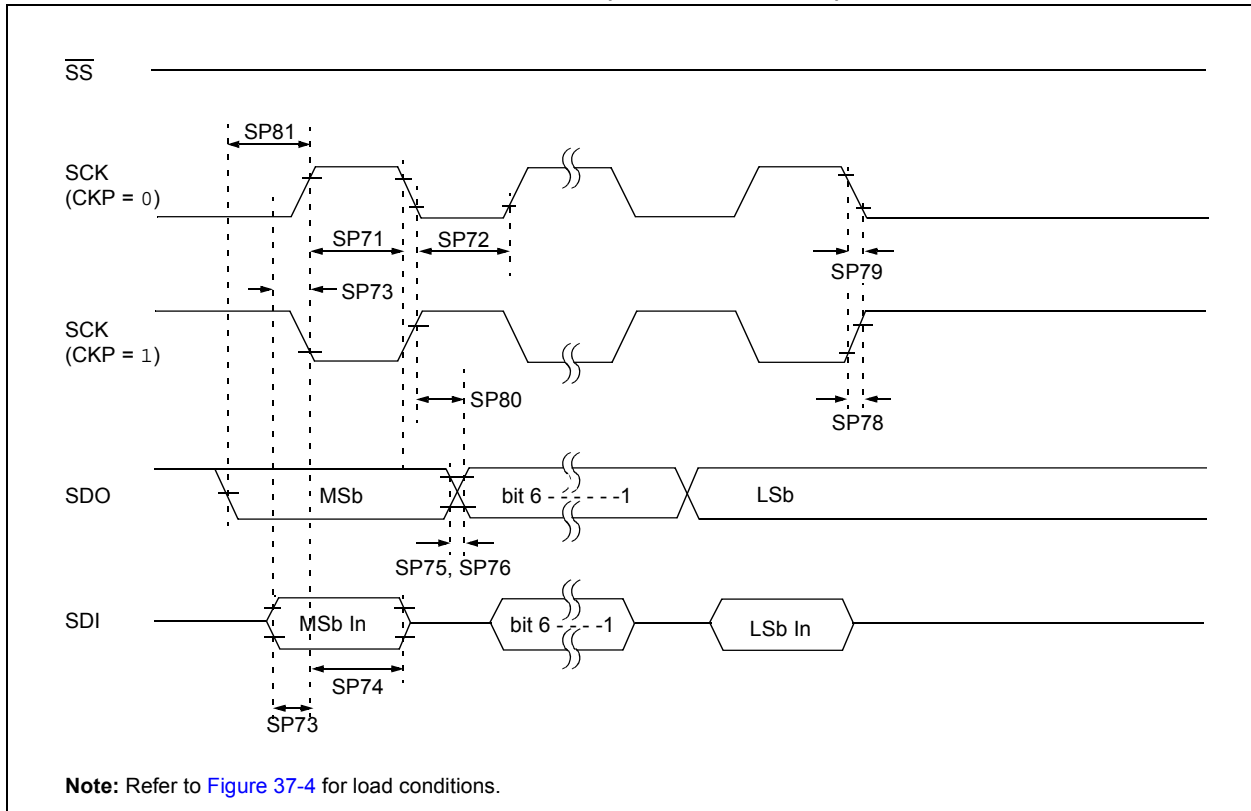
**TABLE 37-22: EUSART SYNCHRONOUS RECEIVE REQUIREMENTS**

Standard Operating Conditions (unless otherwise stated) Operating Temperature $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$						
Param. No.	Symbol	Characteristic	Min.	Max.	Units	Conditions
US125	TDTV2CKL	SYNC RCV (Master and Slave) Data-setup before CK $\downarrow$ (DT hold time)	10	—	ns	
US126	TCKL2DTL	Data-hold after CK $\downarrow$ (DT hold time)	15	—	ns	

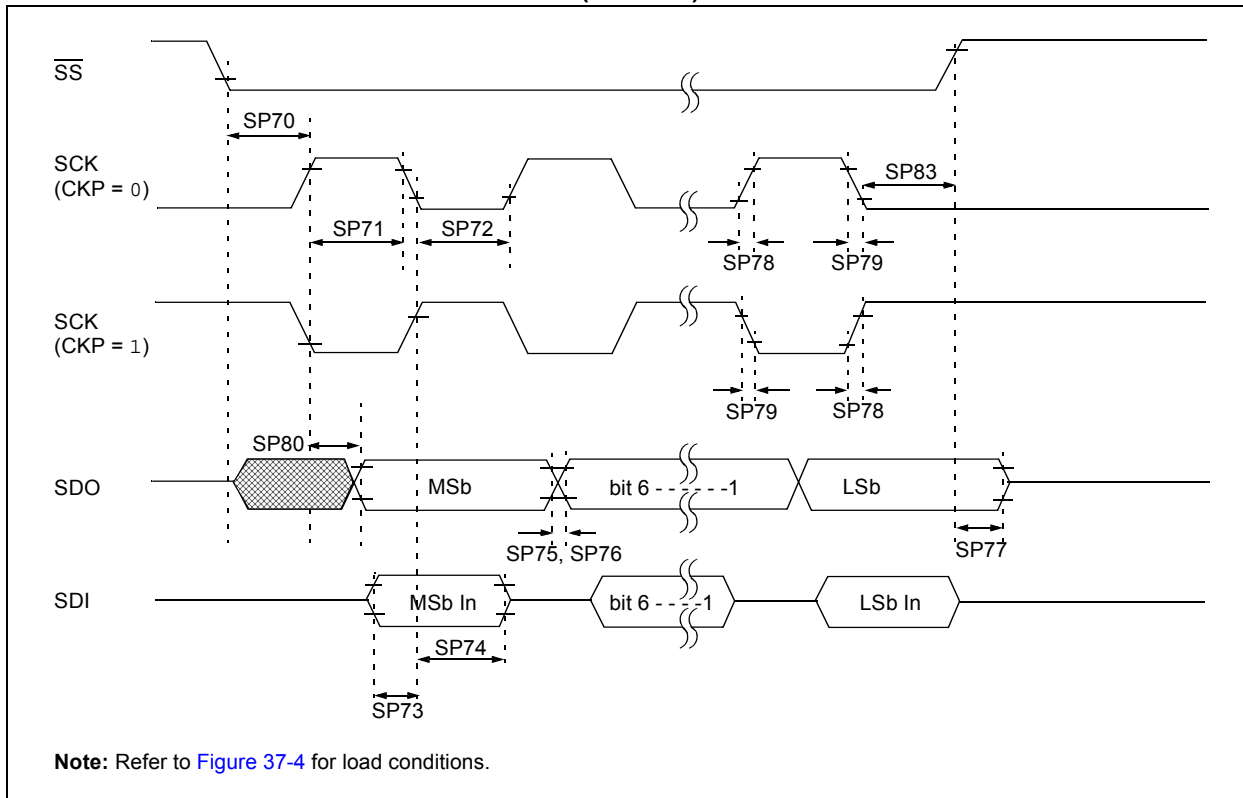
**FIGURE 37-17: SPI MASTER MODE TIMING (CKE = 0, SMP = 0)**



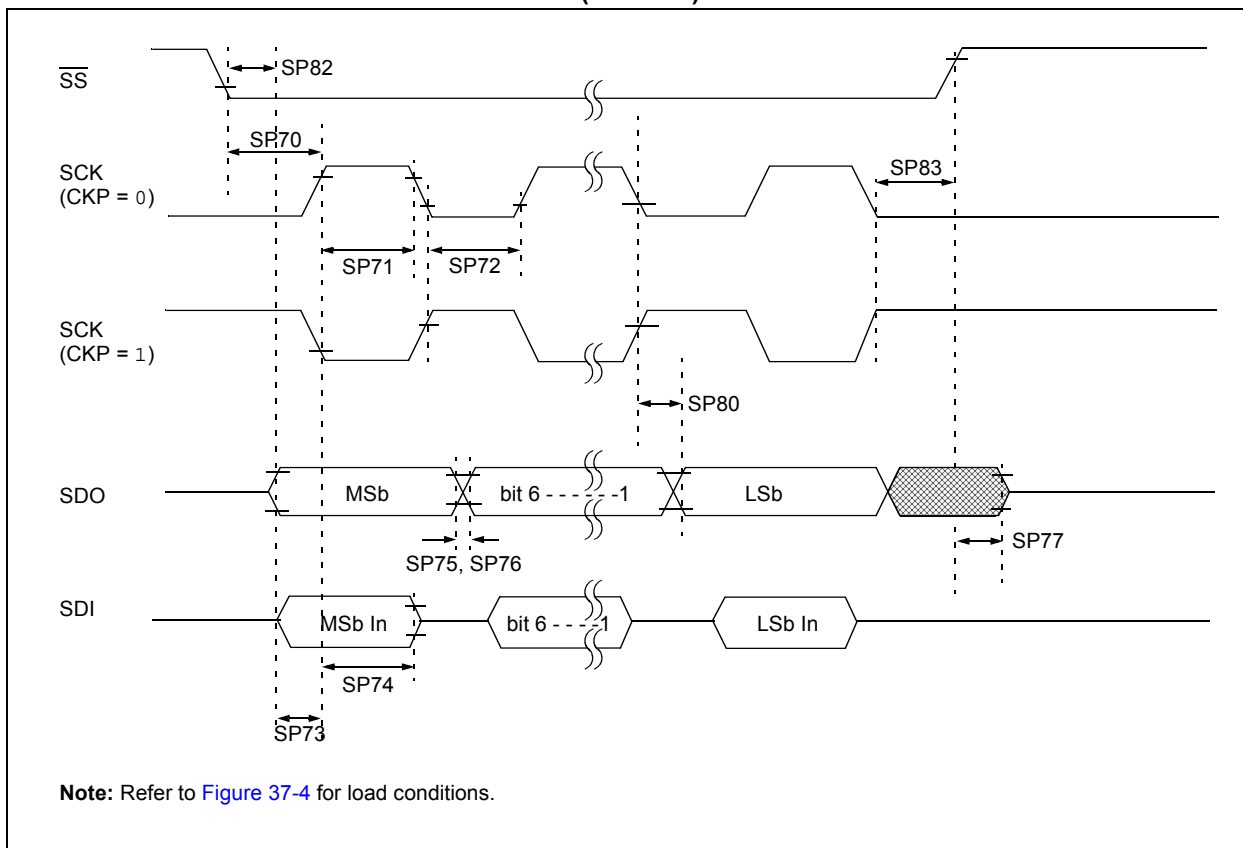
**FIGURE 37-18: SPI MASTER MODE TIMING (CKE = 1, SMP = 1)**



**FIGURE 37-19: SPI SLAVE MODE TIMING (CKE = 0)**



**FIGURE 37-20: SPI SLAVE MODE TIMING (CKE = 1)**



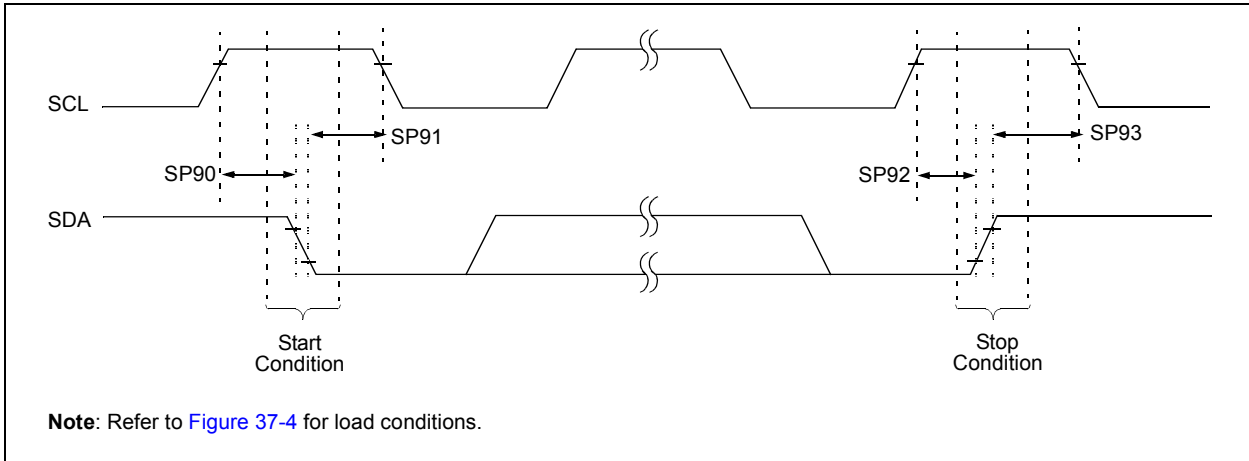
**TABLE 37-23: SPI MODE REQUIREMENTS**

Standard Operating Conditions (unless otherwise stated)							
Param. No.	Symbol	Characteristic	Min.	Typ†	Max.	Units	Conditions
SP70*	TssL2scH, TssL2scL	$\overline{SS}\downarrow$ to SCK $\downarrow$ or SCK $\uparrow$ 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	$\overline{SS}\uparrow$ 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 $\overline{SS}\downarrow$ edge	—	—	50	ns	
SP83*	Tsch2ssH, TscL2ssH	$\overline{SS}\uparrow$ 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 37-21: I<sup>2</sup>C BUS START/STOP BITS TIMING**

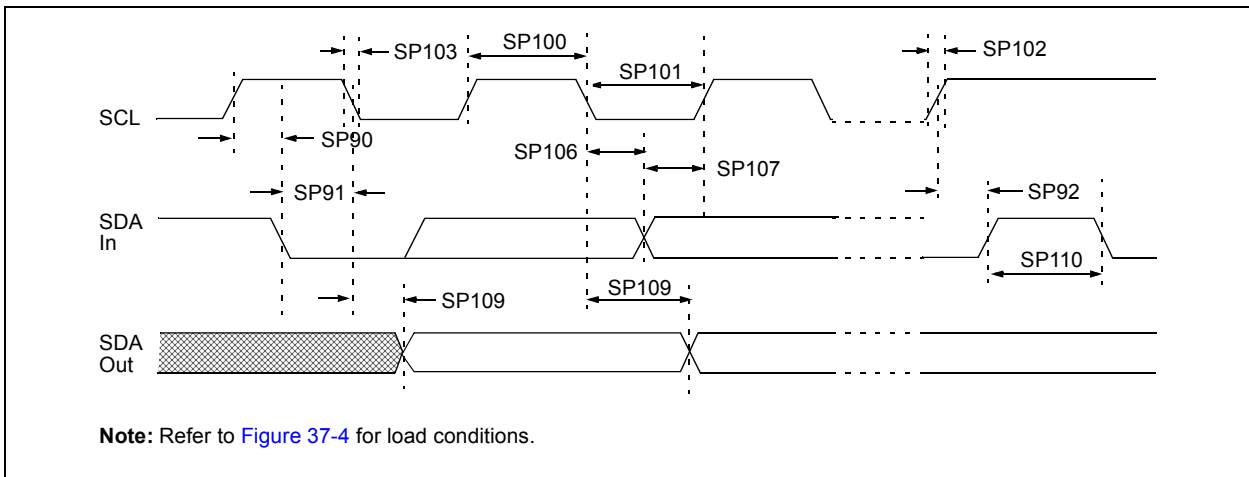


**TABLE 37-24: I<sup>2</sup>C BUS START/STOP BITS REQUIREMENTS**

Standard Operating Conditions (unless otherwise stated)								
Param. No.	Symbol	Characteristic		Min.	Typ	Max.	Units	Conditions
SP90*	TSU:STA	Start condition	100 kHz mode	4700	—	—	ns	Only relevant for Repeated Start condition
		Setup time	400 kHz mode	600	—	—		
SP91*	THD:STA	Start condition	100 kHz mode	4000	—	—	ns	After this period, the first clock pulse is generated
		Hold time	400 kHz mode	600	—	—		
SP92*	TSU:STO	Stop condition	100 kHz mode	4700	—	—	ns	
		Setup time	400 kHz mode	600	—	—		
SP93	THD:STO	Stop condition	100 kHz mode	4000	—	—	ns	
		Hold time	400 kHz mode	600	—	—		

\* These parameters are characterized but not tested.

**FIGURE 37-22: I<sup>2</sup>C BUS DATA TIMING**



**TABLE 37-25: I<sup>2</sup>C 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	1.5T <sub>CY</sub>	—		
SP101*	TLOW	Clock low time	100 kHz mode	4.7	—	μs	Device must operate at a minimum of 1.5 MHz
			400 kHz mode	1.3	—	μs	Device must operate at a minimum of 10 MHz
			SSP module	1.5 T <sub>CY</sub>	—		
SP102*	TR	SDA and SCL rise time	100 kHz mode	—	1000	ns	
			400 kHz mode	20 + 0.1 C <sub>B</sub>	300	ns	C <sub>B</sub> is specified to be from 10-400 pF
SP103*	TF	SDA and SCL fall time	100 kHz mode	—	250	ns	
			400 kHz mode	20 + 0.1 C <sub>B</sub>	250	ns	C <sub>B</sub> 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	<b>(Note 2)</b>
			400 kHz mode	100	—	ns	
SP109*	TAA	Output valid from clock	100 kHz mode	—	3500	ns	<b>(Note 1)</b>
			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	C <sub>B</sub>	Bus capacitive loading		—	400	pF	

\* These parameters are characterized but not tested.

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

**2:** A Fast mode (400 kHz) I<sup>2</sup>C bus device can be used in a Standard mode (100 kHz) I<sup>2</sup>C 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 I<sup>2</sup>C bus specification), before the SCL line is released.

**TABLE 37-26: TEMPERATURE INDICATOR REQUIREMENTS**

Standard Operating Conditions (unless otherwise stated)								
Param No.	Symbol	Characteristic		Min.	Typ†	Max.	Units	Conditions
TS01	TACQMIN	Minimum ADC Acquisition Time Delay		—	25	—	μs	
TS02	MV	Voltage Sensitivity	High Range	—	-3.684	—	mV/°C	TSRNG = 1
			Low Range	—	-2.456	—	mV/°C	TSRNG = 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.

## 38.0 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<sub>DD</sub> range). This is for **information only** and devices are ensured to operate properly only within the specified range.

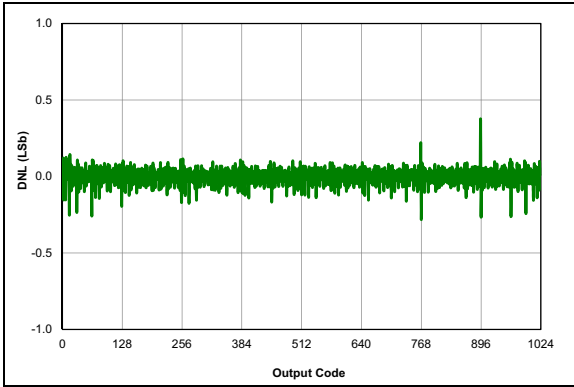
Unless otherwise noted, all graphs apply to both the L and LF devices.

<p><b>Note:</b> The graphs and tables provided following this note are a statistical summary based on a limited number of samples and are provided for informational purposes only. The performance characteristics listed herein are not tested or guaranteed. In some graphs or tables, the data presented may be outside the specified operating range (e.g., outside specified power supply range) and therefore, outside the warranted range.</p>
--

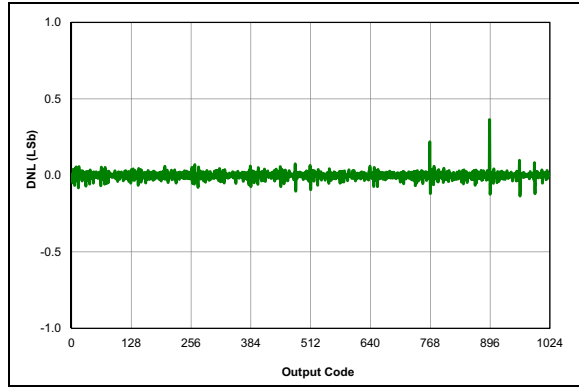
**“Typical”** represents the mean of the distribution at 25°C. **“Maximum”, “Max.”, “Minimum” or “Min.”** represents (mean + 3σ) or (mean - 3σ) respectively, where σ is a standard deviation, over each temperature range.

# PIC16LF15354/55

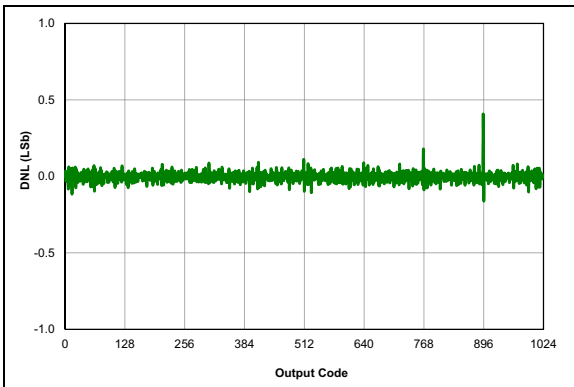
**Note:** Unless otherwise noted,  $V_{IN} = 5V$ ,  $F_{OSC} = 300\text{ kHz}$ ,  $C_{IN} = 0.1\ \mu\text{F}$ ,  $T_A = 25^\circ\text{C}$ .



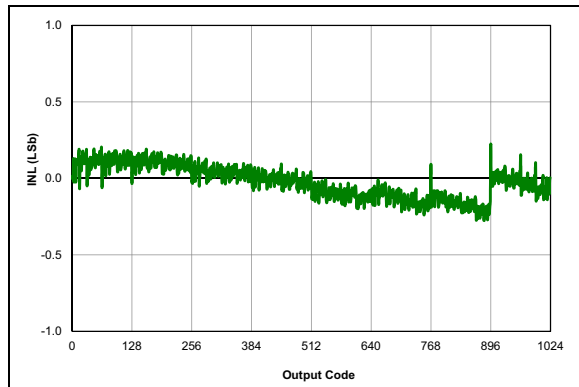
**FIGURE 38-1:** ADC 10-bit Mode, Single-Ended DNL,  $V_{DD} = 3.0V$ ,  $V_{REF} = 3.0V$ ,  $T_{AD} = 1\ \mu\text{S}$ ,  $25^\circ\text{C}$ .



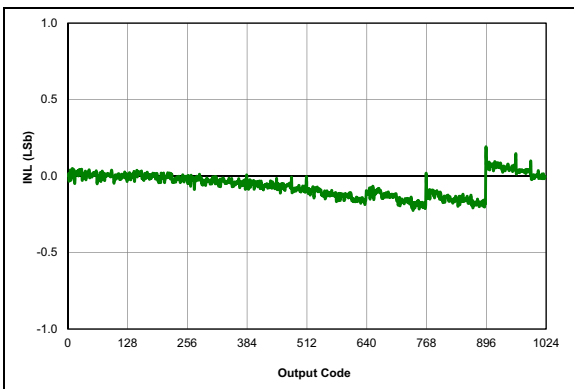
**FIGURE 38-2:** ADC 10-bit Mode, Single-Ended DNL,  $V_{DD} = 3.0V$ ,  $V_{REF} = 3.0V$ ,  $T_{AD} = 4\ \mu\text{S}$ ,  $25^\circ\text{C}$ .



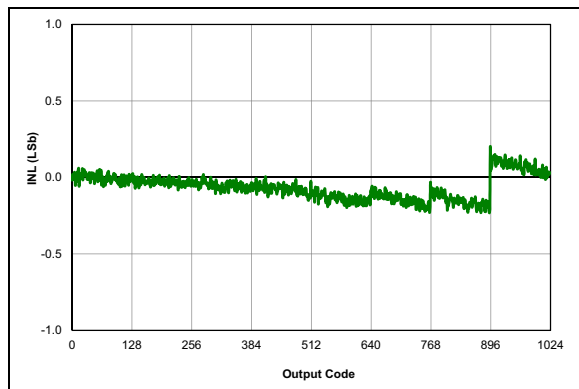
**FIGURE 38-3:** ADC 10-bit Mode, Single-Ended DNL,  $V_{DD} = 3.0V$ ,  $V_{REF} = 3.0V$ ,  $T_{AD} = 8\ \mu\text{S}$ ,  $25^\circ\text{C}$ .



**FIGURE 38-4:** ADC 10-bit Mode, Single-Ended INL,  $V_{DD} = 3.0V$ ,  $V_{REF} = 3.0V$ ,  $T_{AD} = 1\ \mu\text{S}$ ,  $25^\circ\text{C}$ .



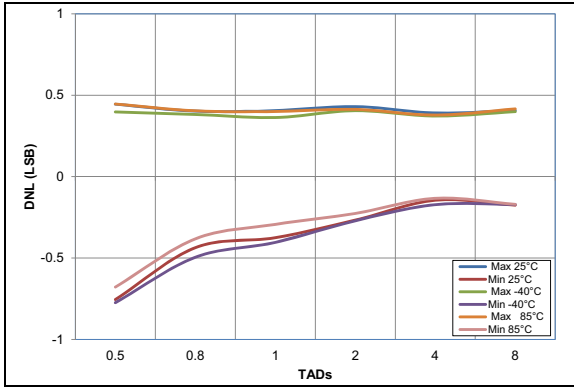
**FIGURE 38-5:** ADC 10-bit Mode, Single-Ended INL,  $V_{DD} = 3.0V$ ,  $V_{REF} = 3.0V$ ,  $T_{AD} = 4\ \mu\text{S}$ ,  $25^\circ\text{C}$ .



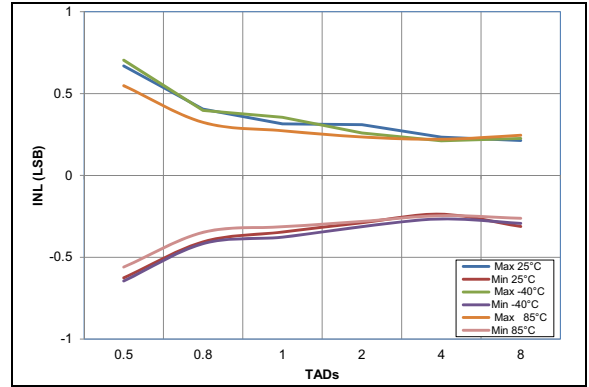
**FIGURE 38-6:** ADC 10-bit Mode, Single-Ended INL,  $V_{DD} = 3.0V$ ,  $V_{REF} = 3.0V$ ,  $T_{AD} = 8\ \mu\text{S}$ ,  $25^\circ\text{C}$ .



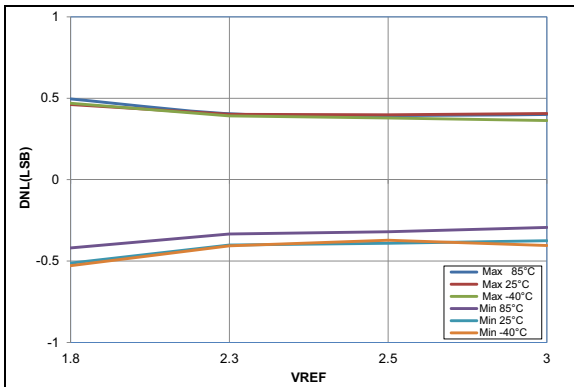
**Note:** Unless otherwise noted,  $V_{IN} = 5V$ ,  $F_{OSC} = 300\text{ kHz}$ ,  $C_{IN} = 0.1\ \mu\text{F}$ ,  $T_A = 25^\circ\text{C}$ .



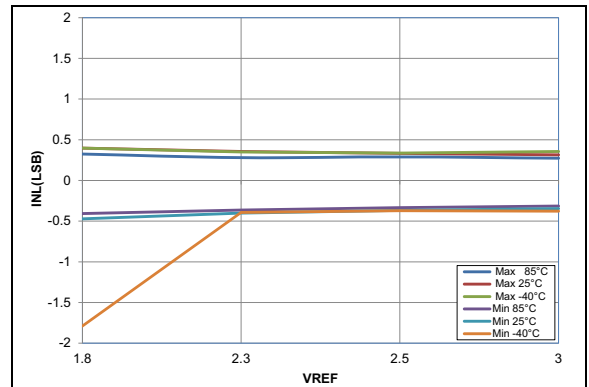
**FIGURE 38-7:** ADC 10-bit Mode, Single-Ended DNL,  $V_{DD} = 3.0V$ ,  $V_{REF} = 3.0V$



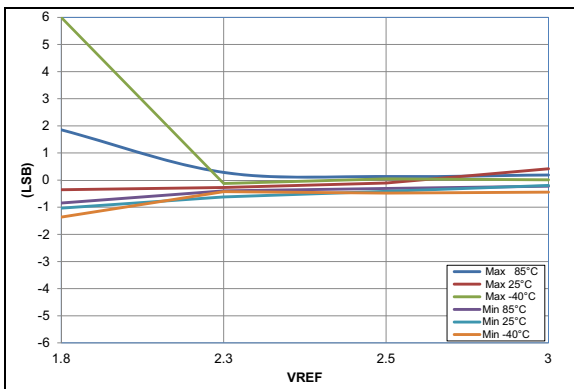
**FIGURE 38-8:** ADC 10-bit Mode, Single-Ended INL,  $V_{DD} = 3.0V$ ,  $V_{REF} = 3.0V$



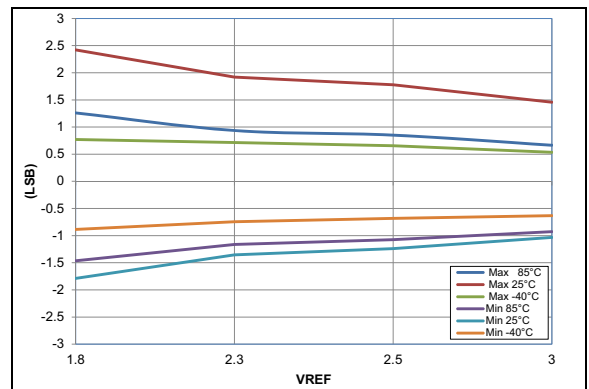
**FIGURE 38-9:** ADC 10-bit Mode, Single-Ended DNL,  $V_{DD} = 3.0V$ ,  $T_{AD} = 1\ \mu\text{s}$



**FIGURE 38-10:** ADC 10-bit Mode, Single-Ended INL,  $V_{DD} = 3.0V$ ,  $T_{AD} = 1\ \mu\text{s}$



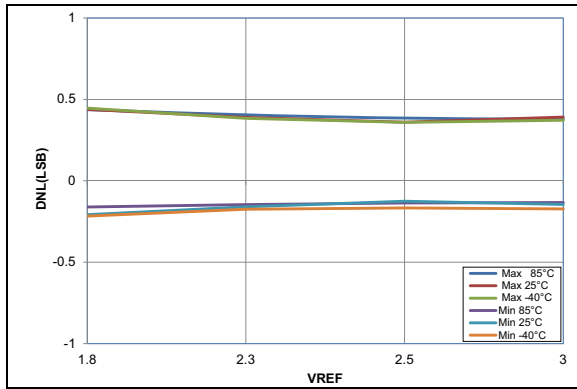
**FIGURE 38-11:** ADC 10-bit Mode, Single-Ended Gain Error,  $V_{DD} = 3.0V$ ,  $T_{AD} = 1\ \mu\text{s}$



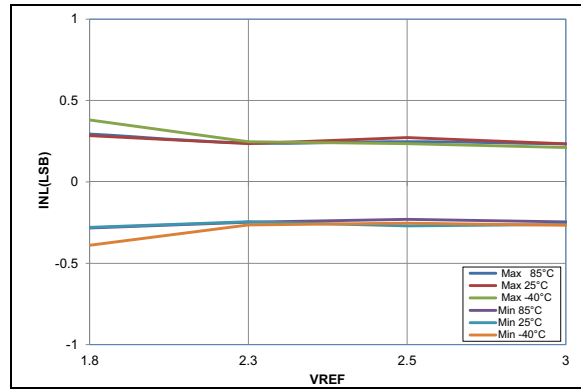
**FIGURE 38-12:** ADC 10-bit Mode, Single-Ended Offset Error,  $V_{DD} = 3.0V$ ,  $T_{AD} = 1\ \mu\text{s}$

# PIC16LF15354/55

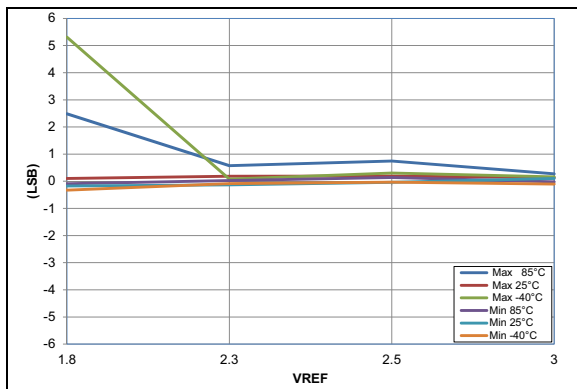
**Note:** Unless otherwise noted,  $V_{IN} = 5V$ ,  $F_{OSC} = 300\text{ kHz}$ ,  $C_{IN} = 0.1\ \mu F$ ,  $T_A = 25^\circ C$ .



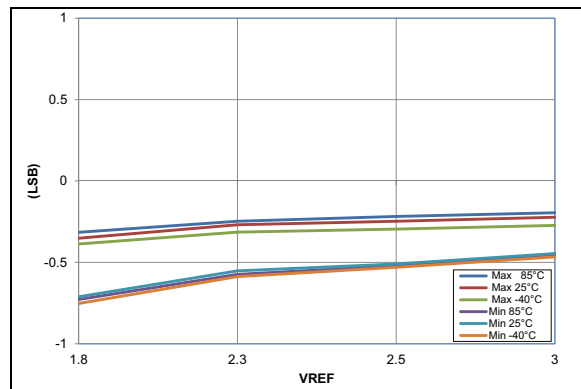
**FIGURE 38-13:** ADC 10-bit Mode, Single-Ended DNL,  $V_{DD} = 3.0V$ ,  $T_{AD} = 4\ \mu S$ .



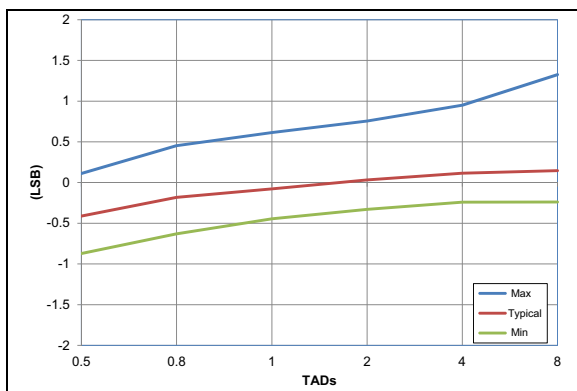
**FIGURE 38-14:** ADC 10-bit Mode, Single-Ended INL,  $V_{DD} = 3.0V$ ,  $T_{AD} = 4\ \mu S$ .



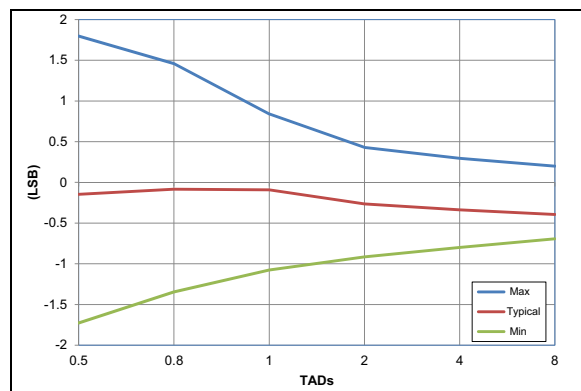
**FIGURE 38-15:** ADC 10-bit Mode, Single-Ended Gain Error,  $V_{DD} = 3.0V$ ,  $T_{AD} = 4\ \mu S$



**FIGURE 38-16:** ADC 10-bit Mode, Single-Ended Offset Error,  $V_{DD} = 3.0V$ ,  $T_{AD} = 4\ \mu S$

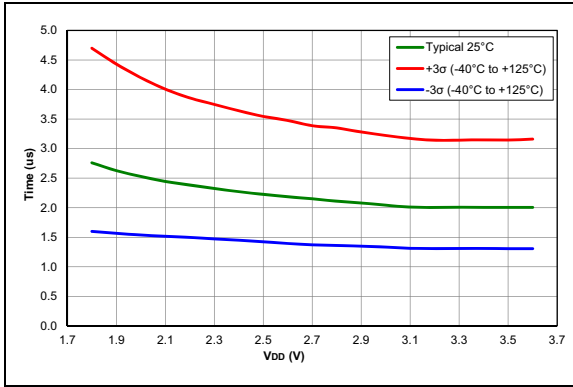


**FIGURE 38-17:** ADC 10-bit Mode, Single-Ended Gain Error,  $V_{DD} = 3.0V$ ,  $V_{REF} = 3.0V$ ,  $-40^\circ C$  to  $85^\circ C$ .

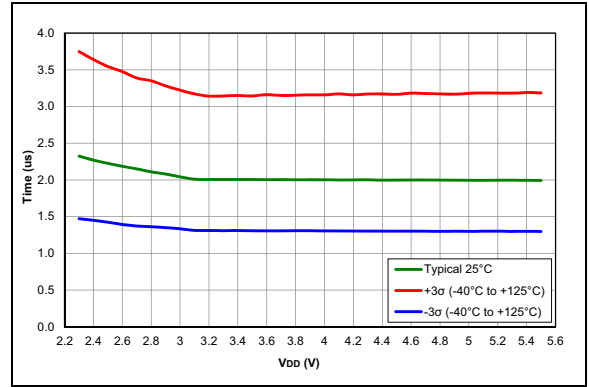


**FIGURE 38-18:** ADC 10-bit Mode, Single-Ended Offset Error,  $V_{DD} = 3.0V$ ,  $V_{REF} = 3.0V$ ,  $-40^\circ C$  to  $85^\circ C$

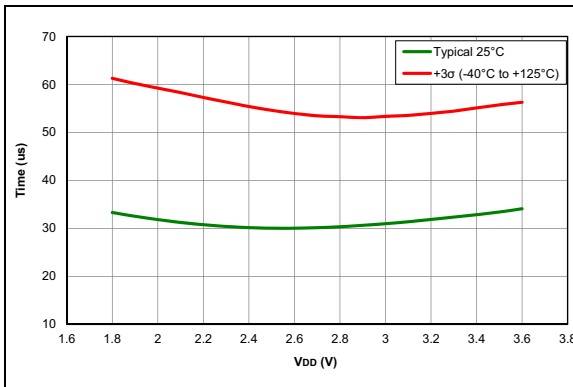
**Note:** Unless otherwise noted,  $V_{IN} = 5V$ ,  $F_{OSC} = 300\text{ kHz}$ ,  $C_{IN} = 0.1\ \mu F$ ,  $T_A = 25^\circ C$ .



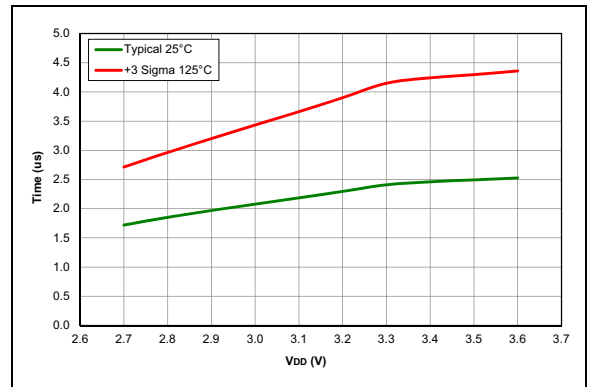
**FIGURE 38-19:** ADC RC Oscillator Period, PIC16F15354/55 devices only.



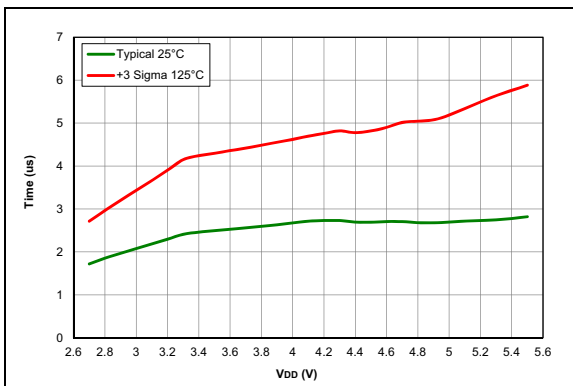
**FIGURE 38-20:** ADC RC Oscillator Period, PIC16F15354/55 devices only.



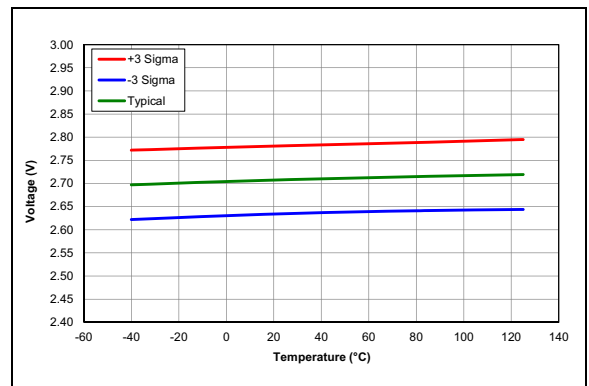
**FIGURE 38-21:** Band Gap Ready



**FIGURE 38-22:** Brown-out Reset Response Time, PIC16F15354/55 devices only.



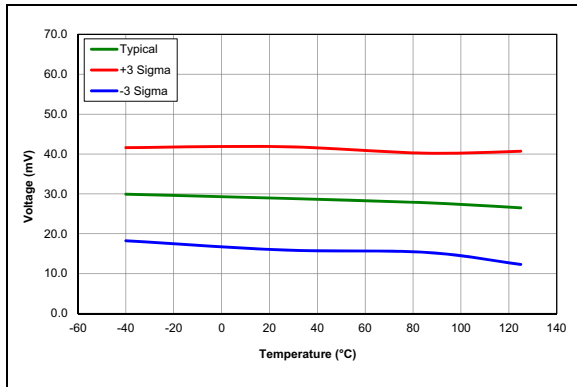
**FIGURE 38-23:** Brown-out Reset Response Time, PIC16F15354/55 devices only.



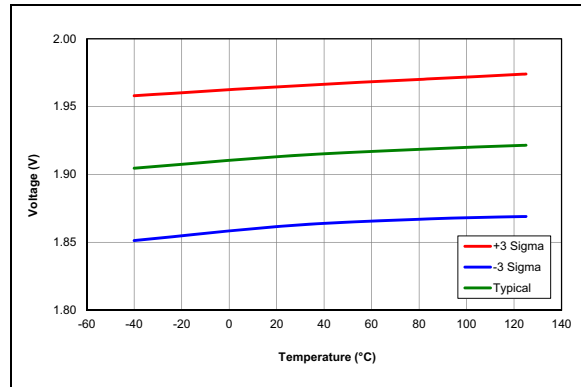
**FIGURE 38-24:** Brown-out Reset Voltage, Trip Point (BORV = 00)

# PIC16LF15354/55

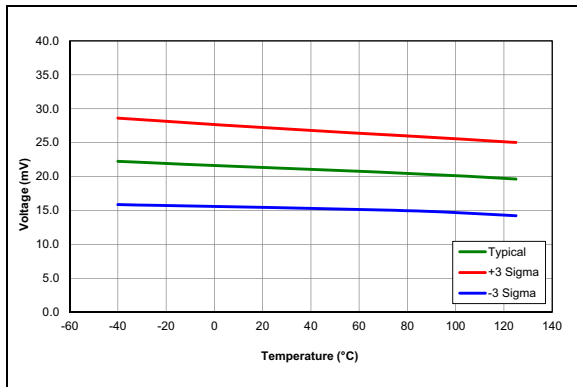
Note: Unless otherwise noted,  $V_{IN} = 5V$ ,  $F_{OSC} = 300\text{ kHz}$ ,  $C_{IN} = 0.1\ \mu\text{F}$ ,  $T_A = 25^\circ\text{C}$ .



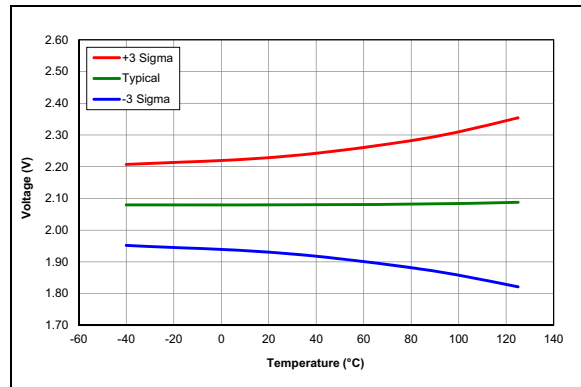
**FIGURE 38-25:** Brown-out Reset Hysteresis, Low-Trip Point (BORV = 00)



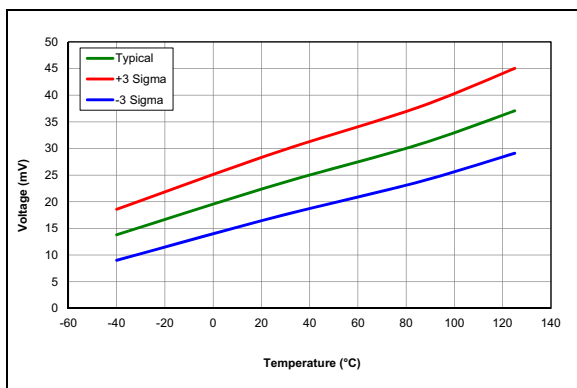
**FIGURE 38-26:** Brown-out Reset Voltage, Trip Point (BORV = 01)



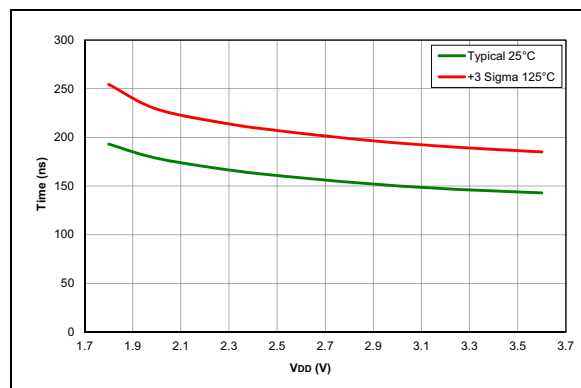
**FIGURE 38-27:** Brown-out Reset Hysteresis, Trip Point (BORV = 01)



**FIGURE 38-28:** LPBOR Reset Voltage

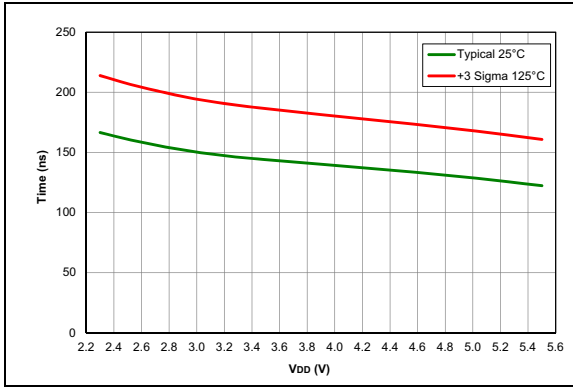


**FIGURE 38-29:** LPBOR Reset Hysteresis

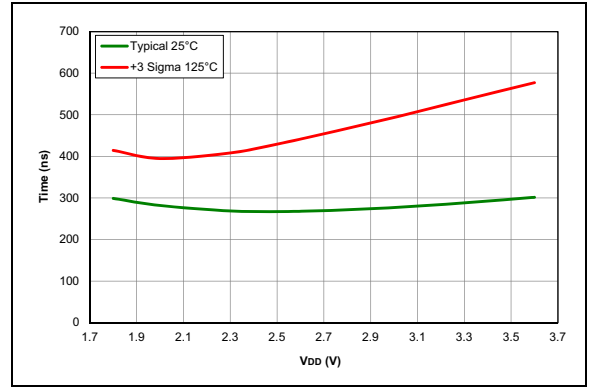


**FIGURE 38-30:** Comparator Response Time Falling Edge, PIC16LF15354/55 devices only.

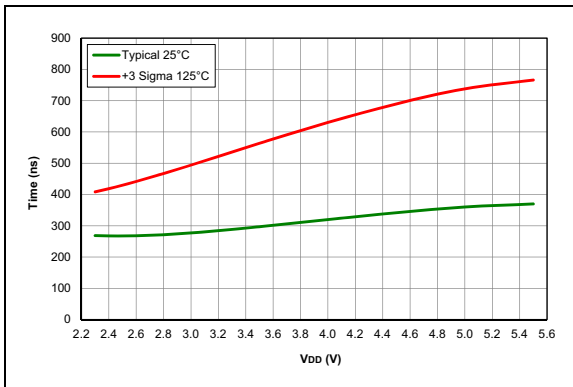
**Note:** Unless otherwise noted,  $V_{IN} = 5V$ ,  $F_{OSC} = 300\text{ kHz}$ ,  $C_{IN} = 0.1\ \mu F$ ,  $T_A = 25^\circ C$ .



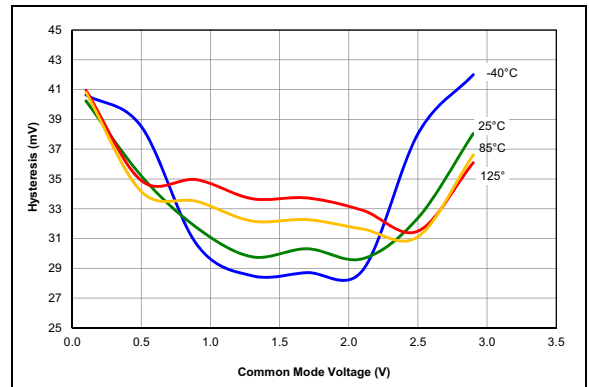
**FIGURE 38-31:** Comparator Response Time Falling Edge, PIC16F15354/55 devices only.



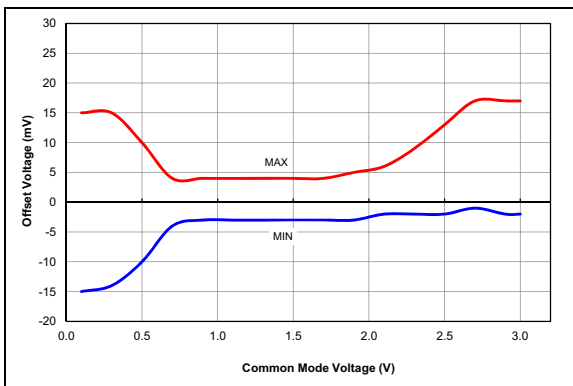
**FIGURE 38-32:** Comparator Response Time Rising Edge, PIC16F15354/55 devices only.



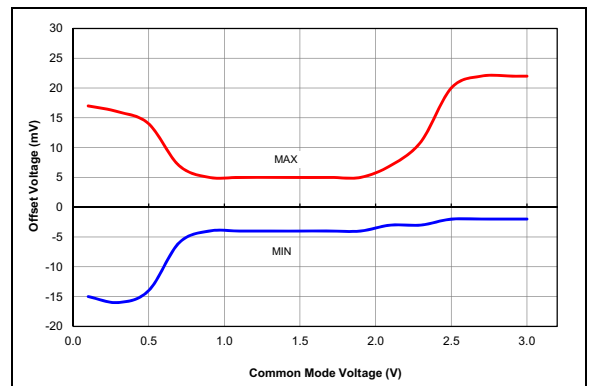
**FIGURE 38-33:** Comparator Response Time Rising Edge, PIC16F15354/55 devices only.



**FIGURE 38-34:** Comparator Hysteresis, Normal Power Mode ( $CxSP = 1$ ),  $V_{DD} = 3.0V$ , Typical Measured Values



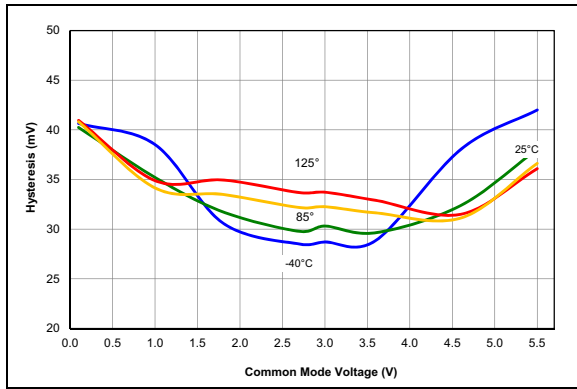
**FIGURE 38-35:** Comparator Offset, Normal Power Mode ( $CxSP = 1$ ),  $V_{DD} = 3.0V$ , Typical Measured Values at  $25^\circ C$ .



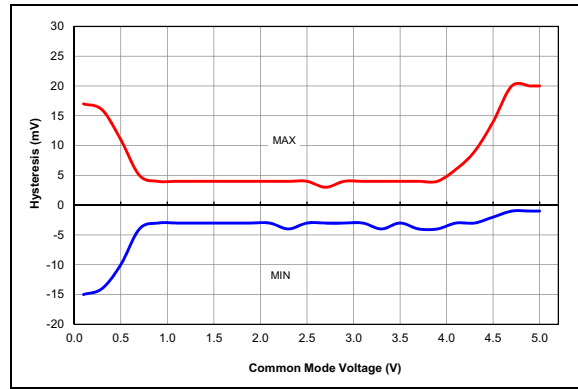
**FIGURE 38-36:** Comparator Offset, Normal Power Mode ( $CxSP = 1$ ),  $V_{DD} = 3.0V$ , Typical Measured Values from  $-40^\circ C$  to  $125^\circ C$ .

# PIC16LF15354/55

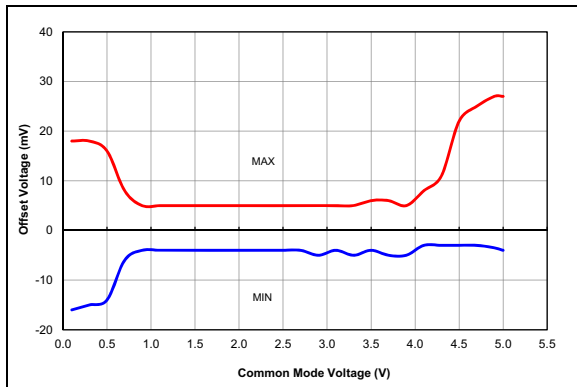
**Note:** Unless otherwise noted,  $V_{IN} = 5V$ ,  $F_{OSC} = 300\text{ kHz}$ ,  $C_{IN} = 0.1\ \mu\text{F}$ ,  $T_A = 25^\circ\text{C}$ .



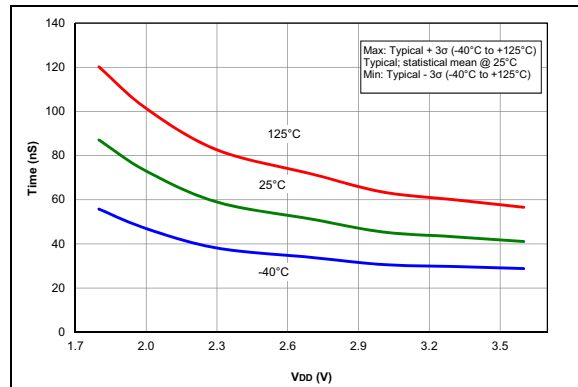
**FIGURE 38-37:** Comparator Hysteresis, Normal Power Mode ( $CxSP = 1$ ),  $V_{DD} = 5.5V$ , Typical Measured Values, PIC16F15354/55 devices only.



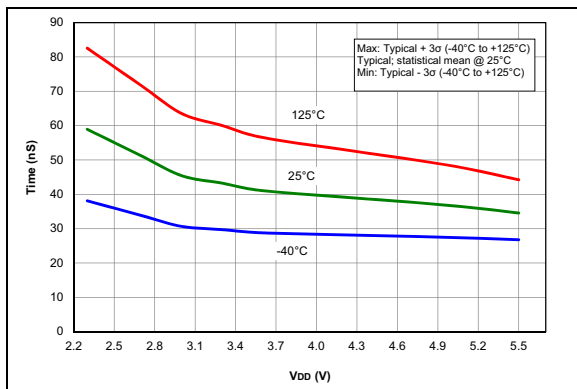
**FIGURE 38-38:** Comparator Offset, Normal Power Mode ( $CxSP = 1$ ),  $V_{DD} = 5.0V$ , Typical Measured Values at  $25^\circ\text{C}$ , PIC16F15354/55 devices only.



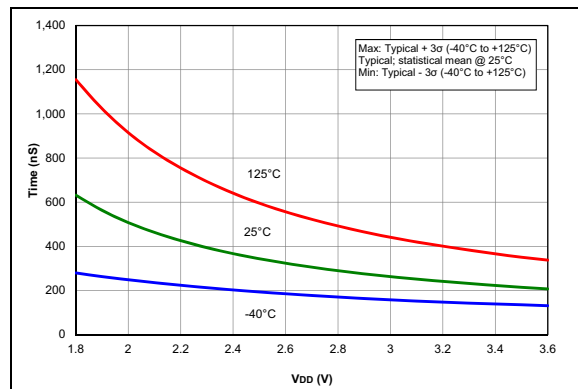
**FIGURE 38-39:** Comparator Offset, Normal Power Mode ( $CxSP = 1$ ),  $V_{DD} = 5.5V$ , Typical Measured Values from  $-40^\circ\text{C}$  to  $125^\circ\text{C}$ , PIC16F15354/55 devices only.



**FIGURE 38-40:** Comparator Response Time Over Voltage, Normal Power Mode ( $CxSP = 1$ ), Typical Measured Values, PIC16F15354/55 devices only.



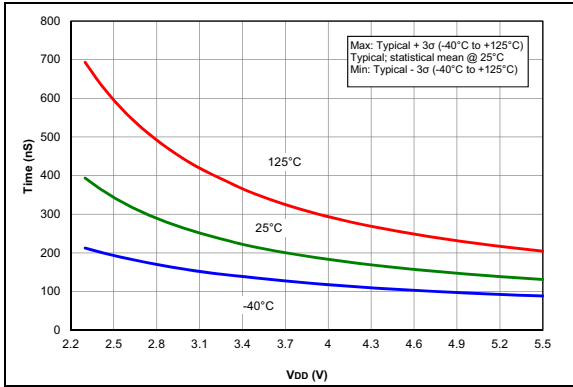
**FIGURE 38-41:** Comparator Response Time Over Voltage, Normal Power Mode ( $CxSP = 1$ ), Typical Measured Values, PIC16F15354/55 devices only.



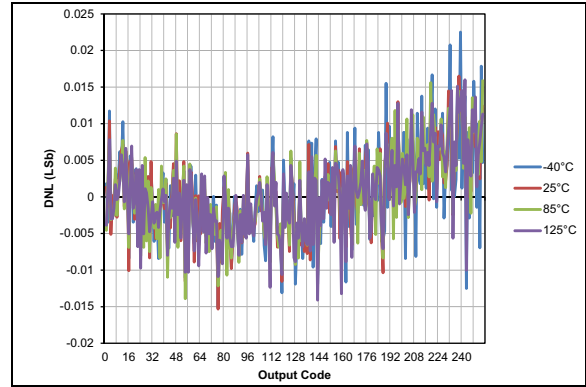
**FIGURE 38-42:** Comparator Output Filter Delay Time Over Temperature, Normal Power Mode ( $CxSP = 1$ ), Typical Measured Values, PIC16F15354/55 devices only.

# PIC16LF15354/55

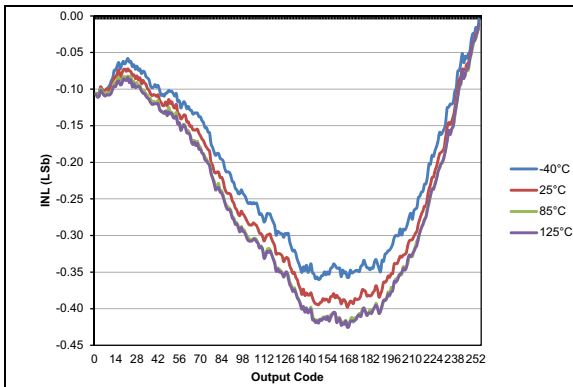
**Note:** Unless otherwise noted,  $V_{IN} = 5V$ ,  $F_{OSC} = 300\text{ kHz}$ ,  $C_{IN} = 0.1\ \mu\text{F}$ ,  $T_A = 25^\circ\text{C}$ .



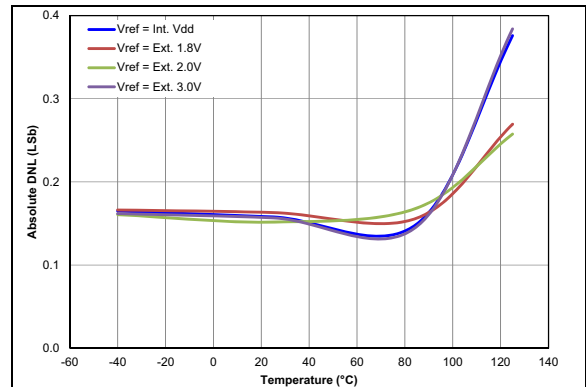
**FIGURE 38-43:** Comparator Output Filter Delay Time Over Temperature, Normal Power Mode ( $CxSP = 1$ ), Typical Measured Values, PIC16F15354/55 devices only



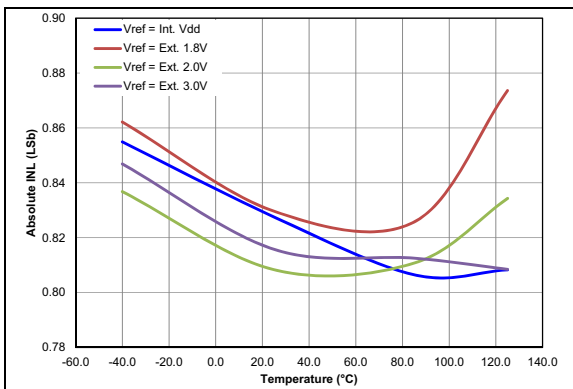
**FIGURE 38-44:** Typical DAC DNL Error,  $V_{DD} = 3.0V$ ,  $V_{REF} = \text{External } 3V$



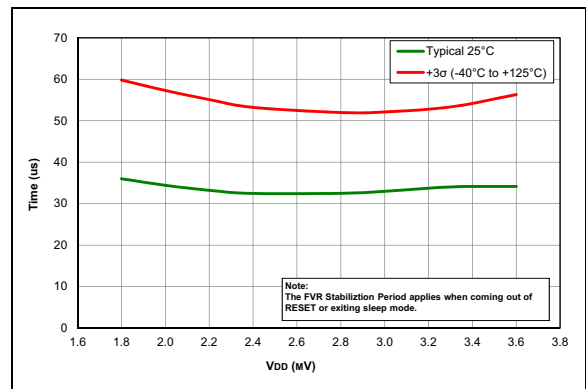
**FIGURE 38-45:** Typical DAC INL Error,  $V_{DD} = 3.0V$ ,  $V_{REF} = \text{External } 3V$



**FIGURE 38-46:** Absolute Value of DAC DNL Error,  $V_{DD} = 3.0V$ ,  $V_{REF} = V_{DD}$



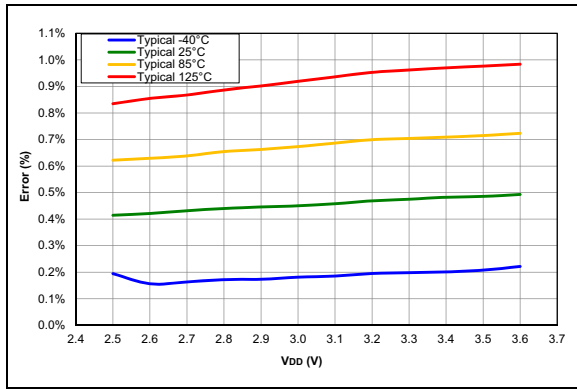
**FIGURE 38-47:** Absolute Value of DAC INL Error,  $V_{DD} = 3.0V$ ,  $V_{REF} = V_{DD}$



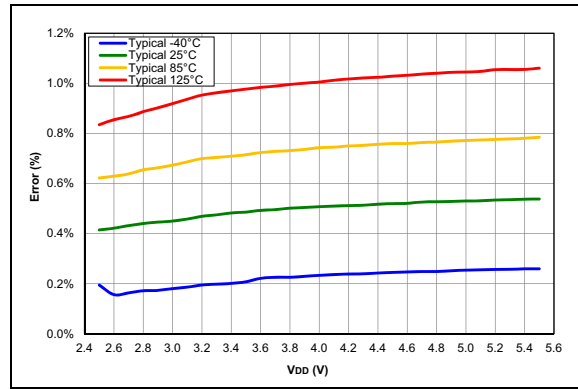
**FIGURE 38-48:** FVR Stabilization Period, PIC16F15354/55 devices only

# PIC16LF15354/55

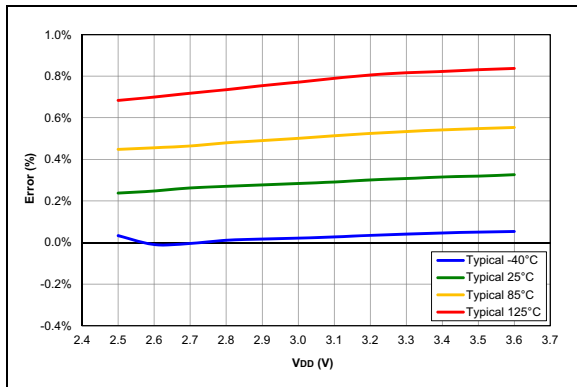
Note: Unless otherwise noted,  $V_{IN} = 5V$ ,  $F_{OSC} = 300\text{ kHz}$ ,  $C_{IN} = 0.1\ \mu F$ ,  $T_A = 25^\circ C$ .



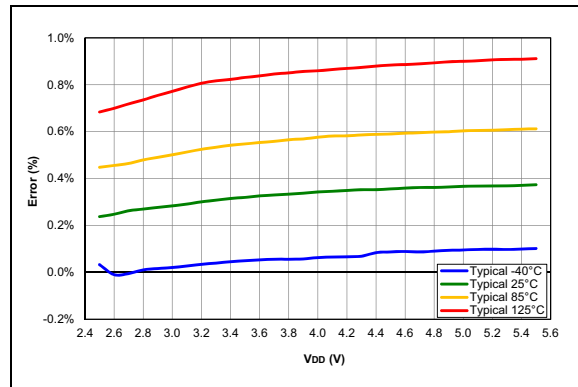
**FIGURE 38-49:** Typical FVR Voltage 1X, PIC16LF15354/55 devices only



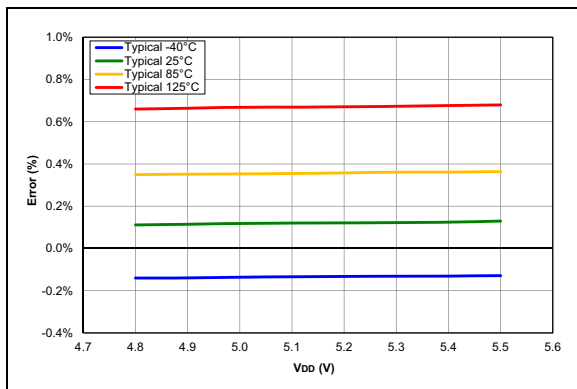
**FIGURE 38-50:** FVR Voltage Error 1X, PIC16LF15354/55 devices only



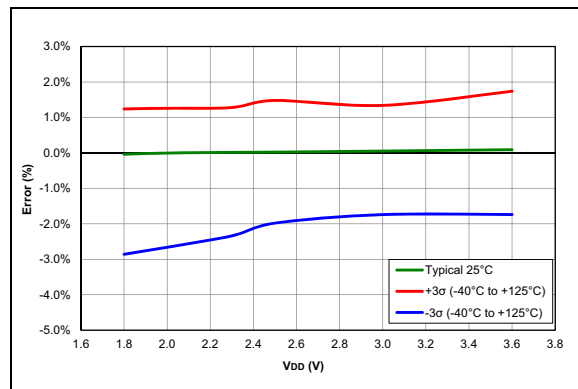
**FIGURE 38-51:** FVR Voltage Error 2X, PIC16LF15354/55 devices only



**FIGURE 38-52:** FVR Voltage Error 2X, PIC16LF15354/55 devices only



**FIGURE 38-53:** FVR Voltage Error 4X, PIC16LF15354/55 devices only

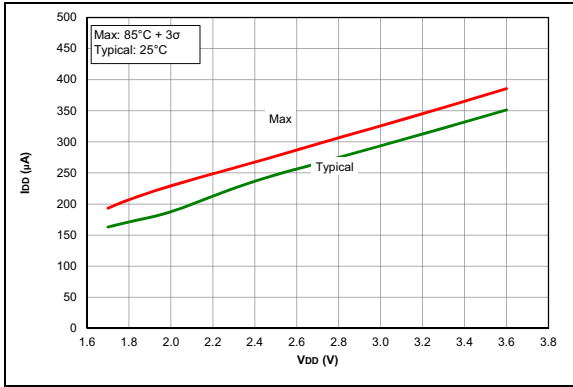


**FIGURE 38-54:** HFINTOSC Typical Frequency Error, PIC16LF15354/55 devices only

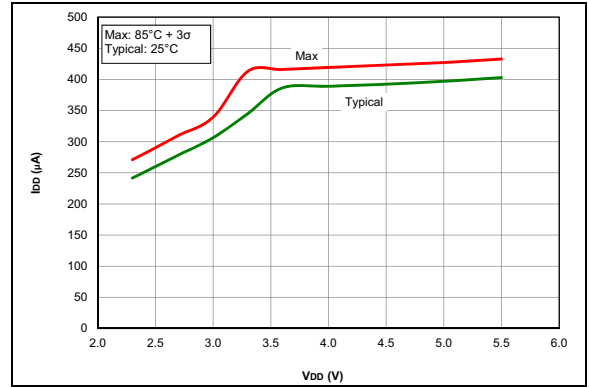


# PIC16LF15354/55

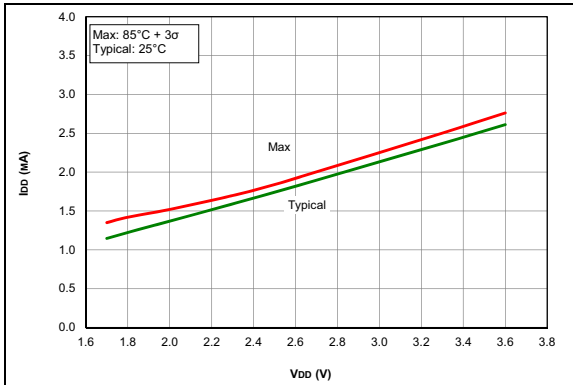
**Note:** Unless otherwise noted,  $V_{IN} = 5V$ ,  $F_{OSC} = 300\text{ kHz}$ ,  $C_{IN} = 0.1\ \mu F$ ,  $T_A = 25^\circ C$ .



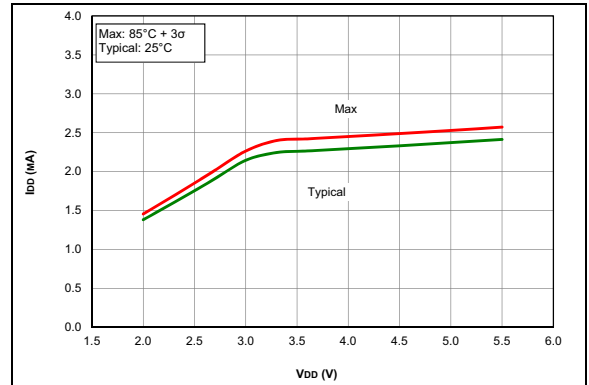
**FIGURE 38-55:**  $I_{DD}$ , XT Oscillator 4 MHz, PIC16LF15354/55 devices only



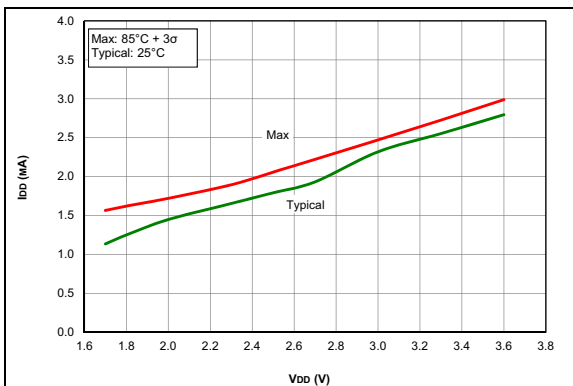
**FIGURE 38-56:**  $I_{DD}$ , XT Oscillator 4 MHz, PIC16LF15354/55 devices only



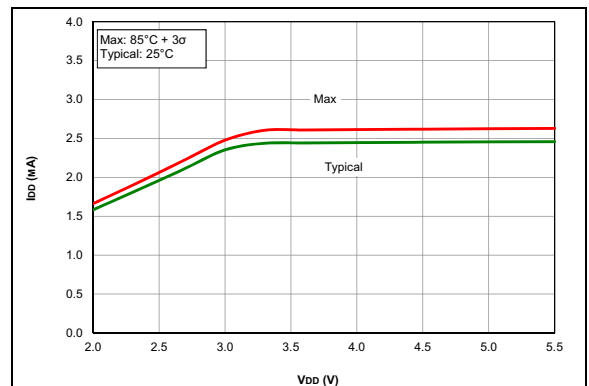
**FIGURE 38-57:**  $I_{DD}$ , HS Oscillator 32 MHz, PIC16LF15354/55 devices only



**FIGURE 38-58:**  $I_{DD}$ , HS Oscillator 32 MHz, PIC16LF15354/55 devices only



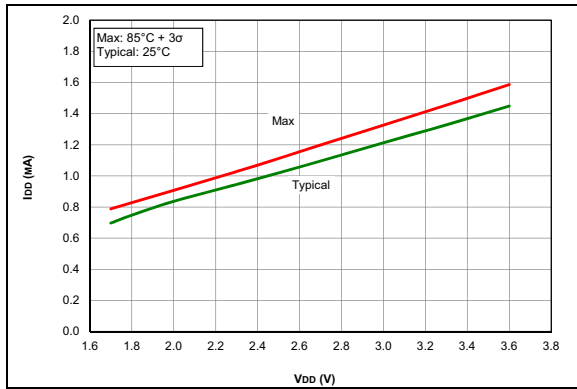
**FIGURE 38-59:**  $I_{DD}$ , HFINTOSC Mode,  $F_{OSC} = 32\text{ MHz}$ , PIC16LF15354/55 devices only



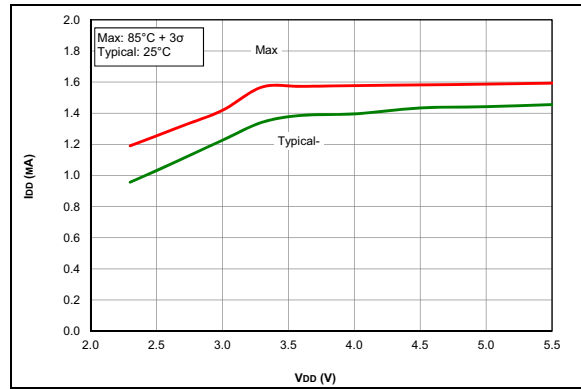
**FIGURE 38-60:**  $I_{DD}$ , HFINTOSC Mode,  $F_{OSC} = 32\text{ MHz}$ , PIC16LF15354/55 devices only

# PIC16LF15354/55

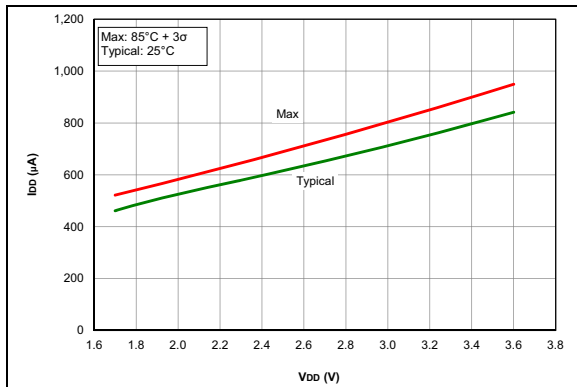
Note: Unless otherwise noted,  $V_{IN} = 5V$ ,  $F_{OSC} = 300\text{ kHz}$ ,  $C_{IN} = 0.1\ \mu\text{F}$ ,  $T_A = 25^\circ\text{C}$ .



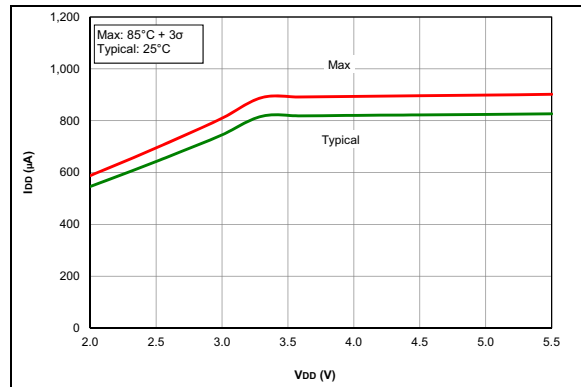
**FIGURE 38-61:**  $I_{DD}$ , HFINTOSC Mode,  $F_{OSC} = 16\text{ MHz}$ , PIC16LF15354/55 devices only



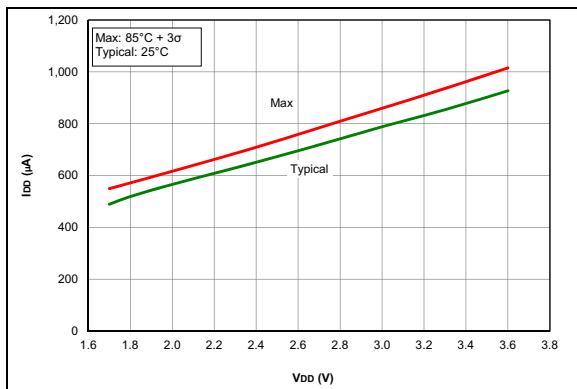
**FIGURE 38-62:**  $I_{DD}$ , HFINTOSC Mode,  $F_{OSC} = 16\text{ MHz}$ , PIC16LF15354/55 devices only



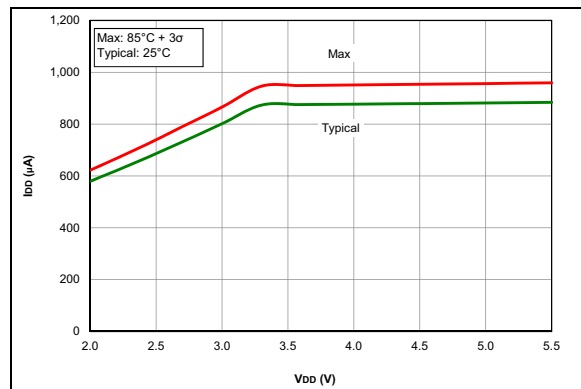
**FIGURE 38-63:**  $I_{DD}$ , HFINTOSC Idle Mode,  $F_{OSC} = 16\text{ MHz}$ , PIC16LF15354/55 devices only



**FIGURE 38-64:**  $I_{DD}$ , HFINTOSC Idle Mode,  $F_{OSC} = 16\text{ MHz}$ , PIC16LF15354/55 devices only

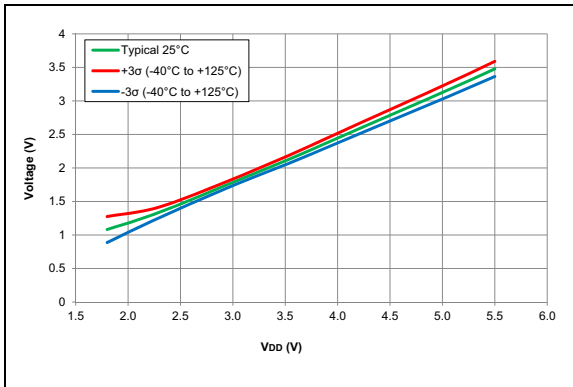


**FIGURE 38-65:**  $I_{DD}$ , HFINTOSC Doze Mode,  $F_{OSC} = 16\text{ MHz}$ , PIC16LF15354/55 devices only

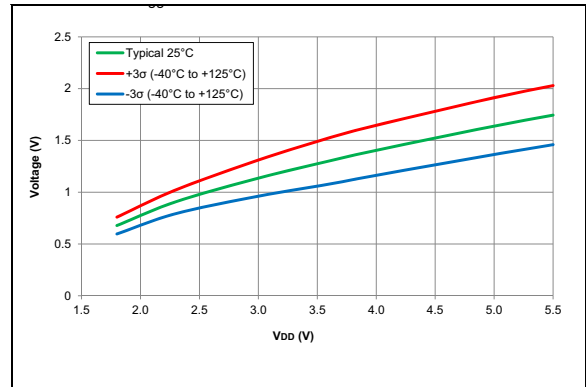


**FIGURE 38-66:**  $I_{DD}$ , HFINTOSC Doze Mode,  $F_{OSC} = 16\text{ MHz}$ , PIC16LF15354/55 devices only

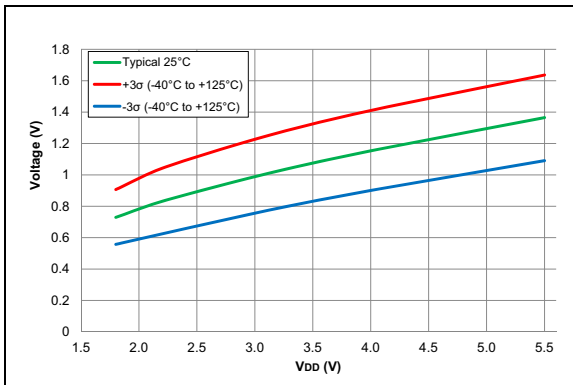
**Note:** Unless otherwise noted,  $V_{IN} = 5V$ ,  $F_{OSC} = 300\text{ kHz}$ ,  $C_{IN} = 0.1\ \mu\text{F}$ ,  $T_A = 25^\circ\text{C}$ .



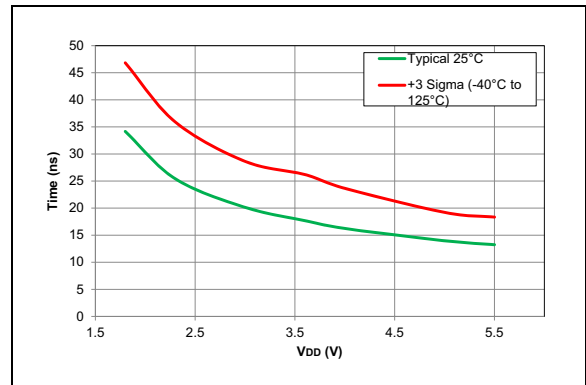
**FIGURE 38-67:** Schmitt Trigger High Values



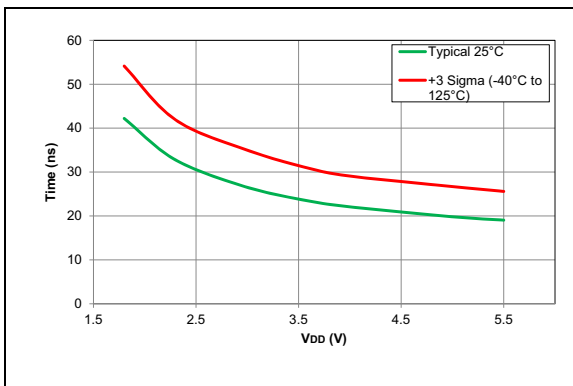
**FIGURE 38-68:** Schmitt Trigger Low Values



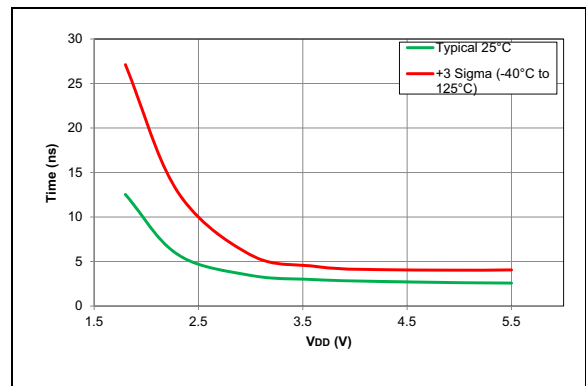
**FIGURE 38-69:** Input Level TTL Trip Thresholds



**FIGURE 38-70:** Rise Time, Slew Rate Control Enabled



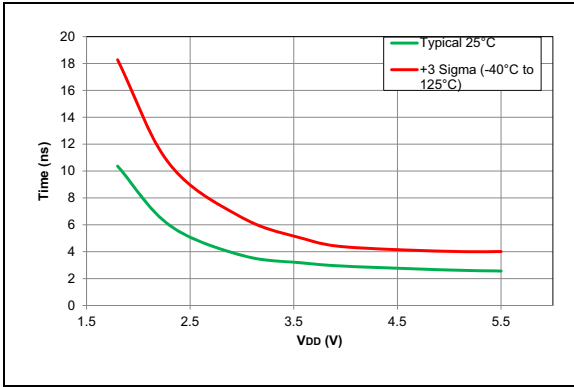
**FIGURE 38-71:** Fall Time, Slew Rate Control Enabled



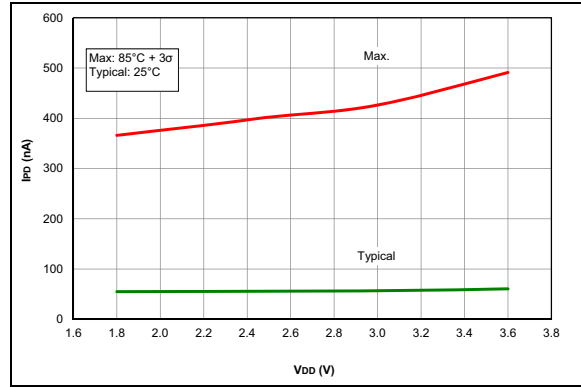
**FIGURE 38-72:** Rise Time, Slew Rate Control Disabled

# PIC16LF15354/55

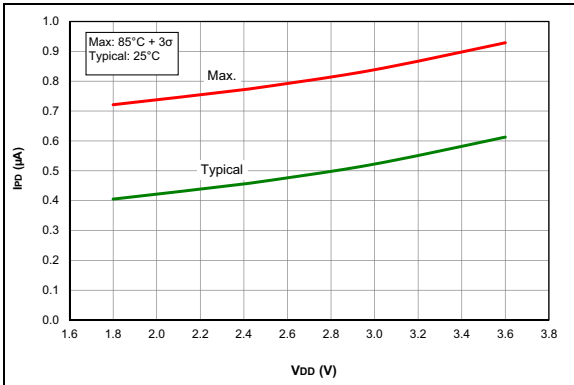
Note: Unless otherwise noted,  $V_{IN} = 5V$ ,  $F_{OSC} = 300\text{ kHz}$ ,  $C_{IN} = 0.1\ \mu\text{F}$ ,  $T_A = 25^\circ\text{C}$ .



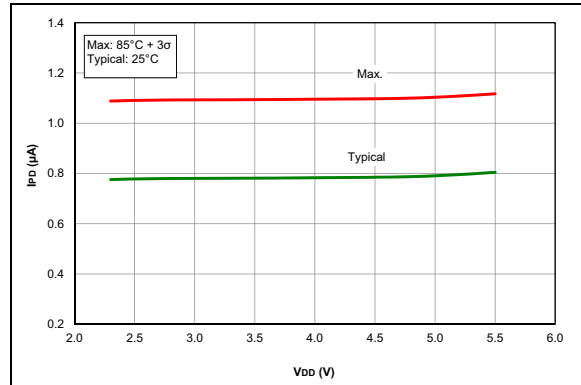
**FIGURE 38-73:** Rise Time, Slew Rate Control Disabled



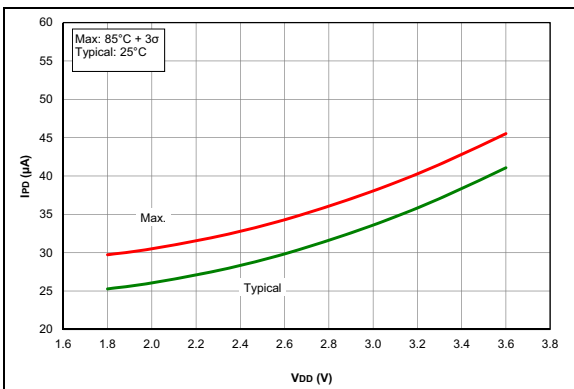
**FIGURE 38-74:** IPD Base, Low-Power Sleep Mode, PIC16LF15354/55 devices only



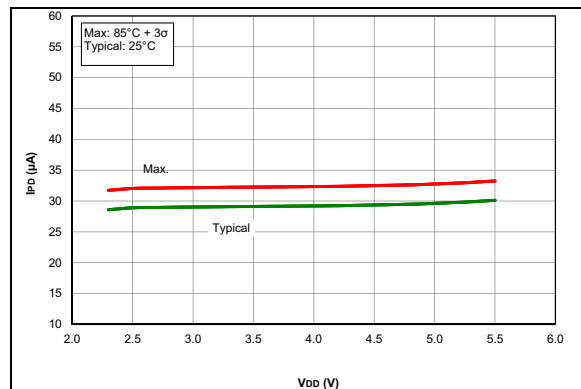
**FIGURE 38-75:** IPD, Watchdog Timer (WDT), PIC16LF15354/55 devices only



**FIGURE 38-76:** IPD, Watchdog Timer (WDT), PIC16LF15354/55 devices only



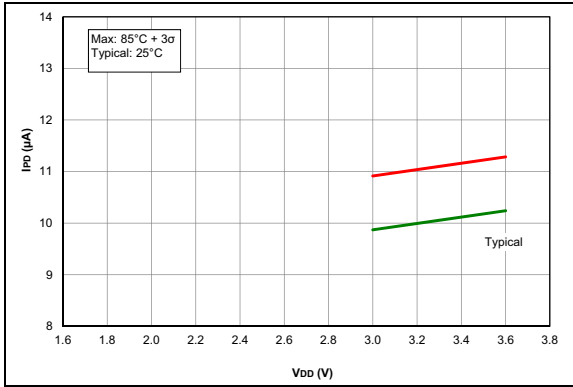
**FIGURE 38-77:** IPD, Fixed Voltage Reference (FVR), PIC16LF15354/55 devices only



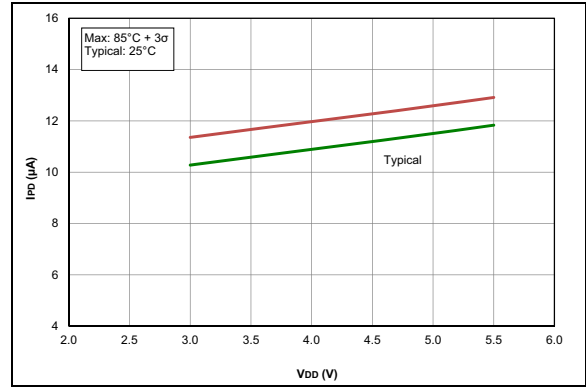
**FIGURE 38-78:** IPD, Fixed Voltage Reference (FVR), PIC16LF15354/55 devices only

# PIC16LF15354/55

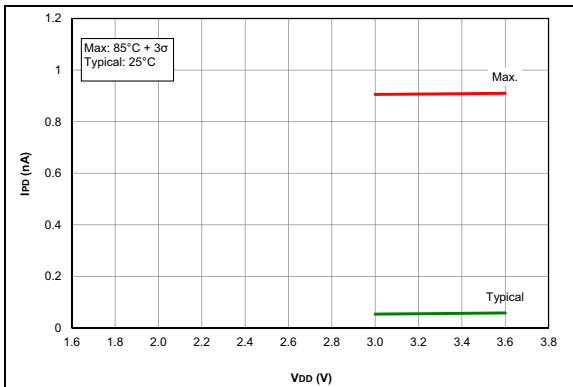
**Note:** Unless otherwise noted,  $V_{IN} = 5V$ ,  $F_{OSC} = 300\text{ kHz}$ ,  $C_{IN} = 0.1\ \mu\text{F}$ ,  $T_A = 25^\circ\text{C}$ .



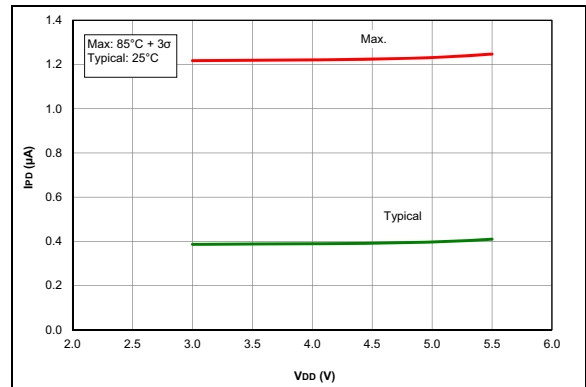
**FIGURE 38-79:**  $I_{PD}$ , Brown-out Reset (BOR), BORV = 1, PIC16LF15354/55 devices only



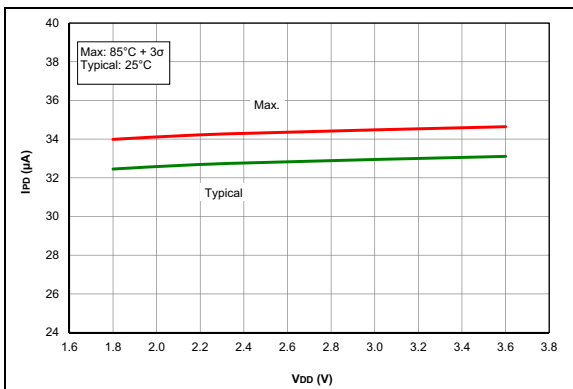
**FIGURE 38-80:**  $I_{PD}$ , Brown-out Reset (BOR), BORV = 1, PIC16F15354/55 devices only



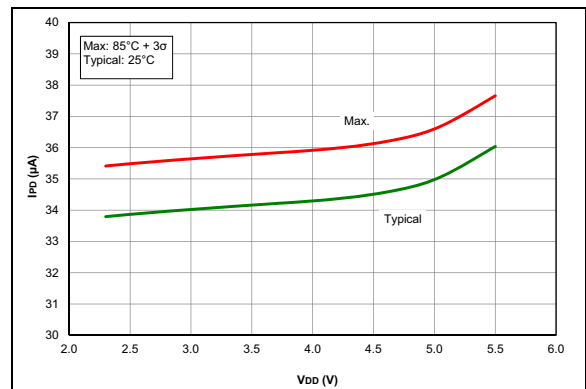
**FIGURE 38-81:**  $I_{PD}$ , Low-Power Brown-out Reset (LPBOR = 0), PIC16LF15354/55 devices only



**FIGURE 38-82:**  $I_{PD}$ , Low-Power Brown-out Reset (LPBOR = 0), PIC16F15354/55 devices only



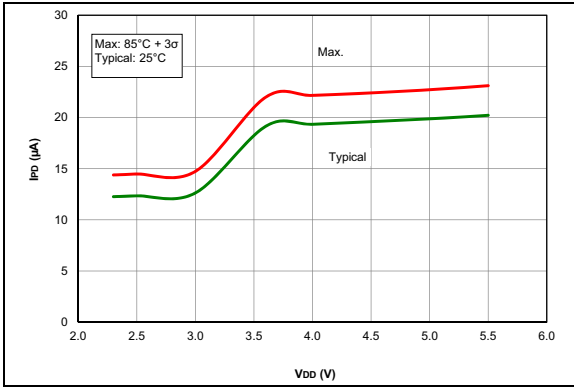
**FIGURE 38-83:**  $I_{PD}$ , Comparator, PIC16LF15354/55 devices only



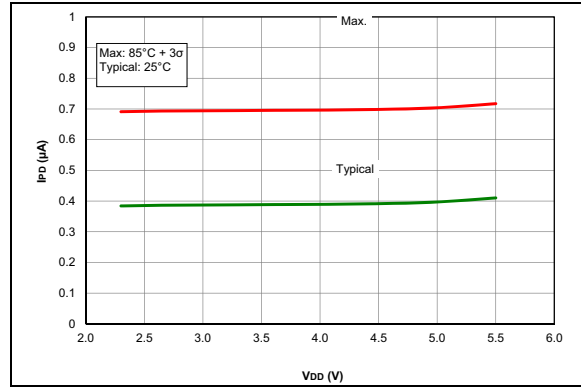
**FIGURE 38-84:**  $I_{PD}$ , Comparator, PIC16F15354/55 devices only

# PIC16LF15354/55

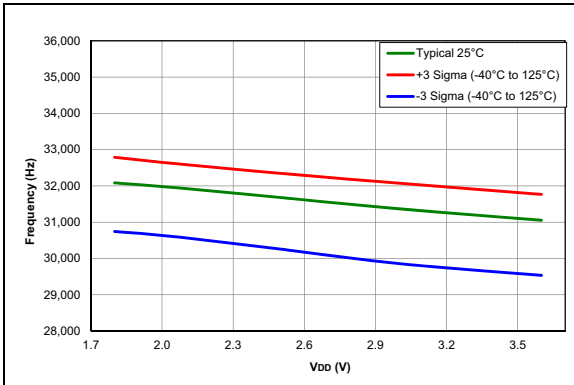
Note: Unless otherwise noted,  $V_{IN} = 5V$ ,  $F_{OSC} = 300\text{ kHz}$ ,  $C_{IN} = 0.1\ \mu\text{F}$ ,  $T_A = 25^\circ\text{C}$ .



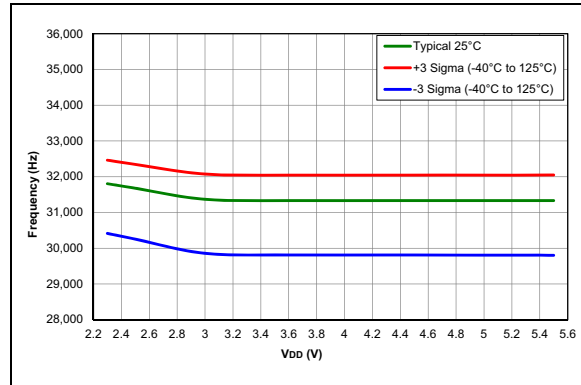
**FIGURE 38-85:** *Ipd Base, 01, PIC16F15354/55 devices only*



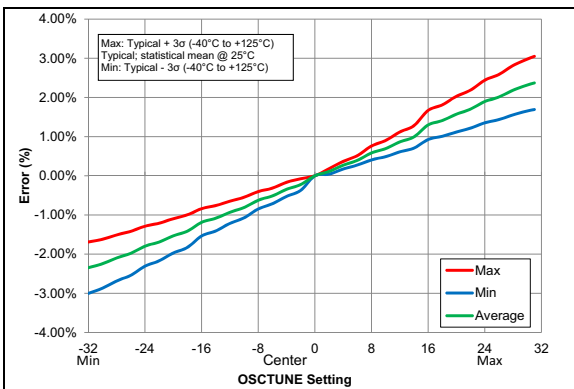
**FIGURE 38-86:** *Ipd Base, 11, PIC16F15354/55 devices only*



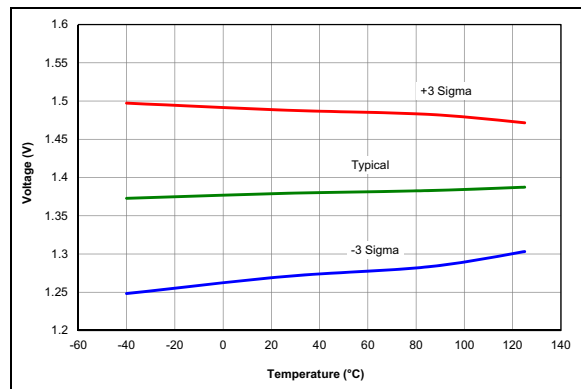
**FIGURE 38-87:** *LFINTOSC Frequency, PIC16F15354/55 devices only*



**FIGURE 38-88:** *LFINTOSC Frequency, PIC16F15354/55 devices only*



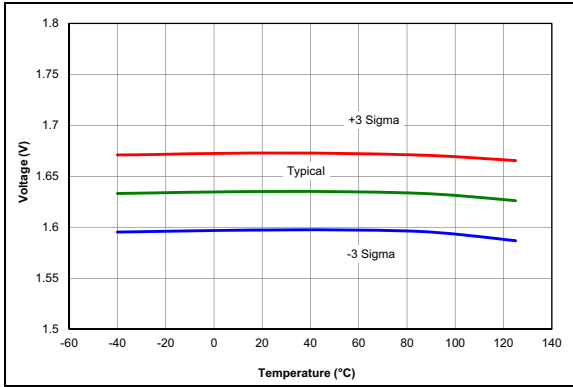
**FIGURE 38-89:** *OCSTUNE Center Frequency, PIC16F15354/55 devices only*



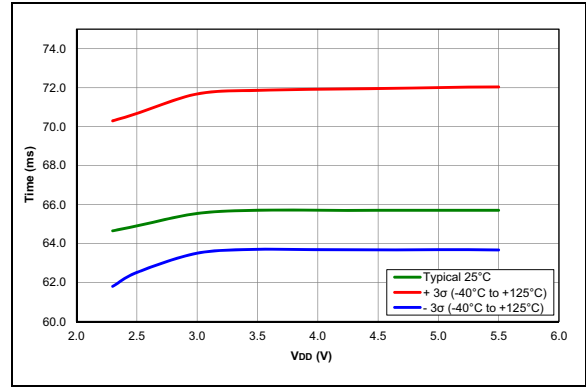
**FIGURE 38-90:** *Power-on Reset Release Voltage*

# PIC16LF15354/55

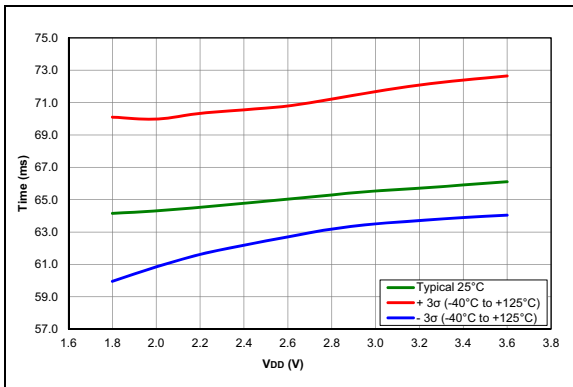
**Note:** Unless otherwise noted,  $V_{IN} = 5V$ ,  $F_{OSC} = 300\text{ kHz}$ ,  $C_{IN} = 0.1\ \mu\text{F}$ ,  $T_A = 25^\circ\text{C}$ .



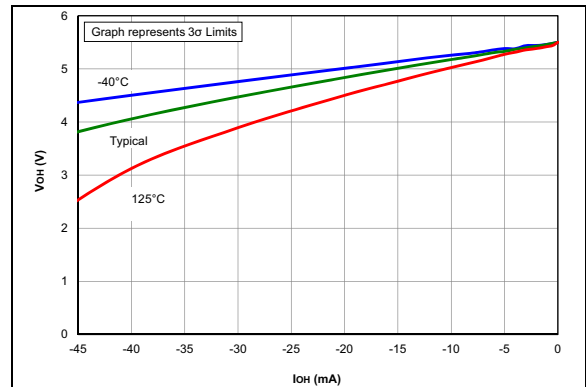
**FIGURE 38-91:** Power-on Reset Rearm Voltage, Normal Power Mode, PIC16F15354/55 devices only



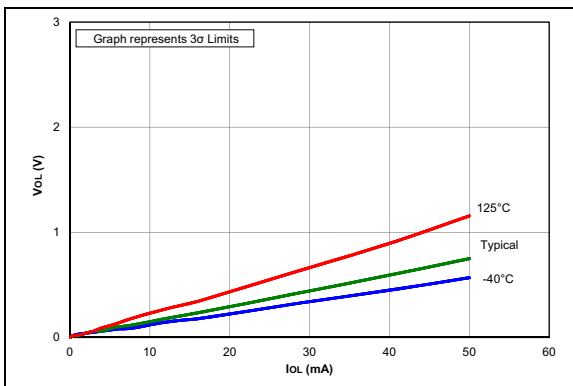
**FIGURE 38-92:** PWRT Period, PIC16F15354/55 devices only



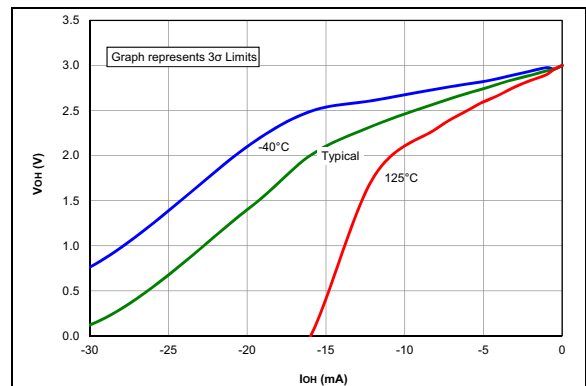
**FIGURE 38-93:** PWRT Period, PIC16F15354/55 devices only



**FIGURE 38-94:**  $V_{OH}$  Vs.  $I_{OH}$  Over Temperature,  $V_{DD} = 5.5V$ , PIC16F15354/55 devices only



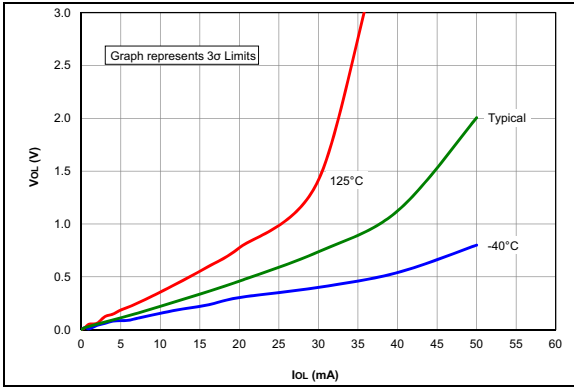
**FIGURE 38-95:**  $V_{OL}$  Vs.  $I_{OL}$  Over Temperature,  $V_{DD} = 5.5V$ , PIC16F15354/55 devices only



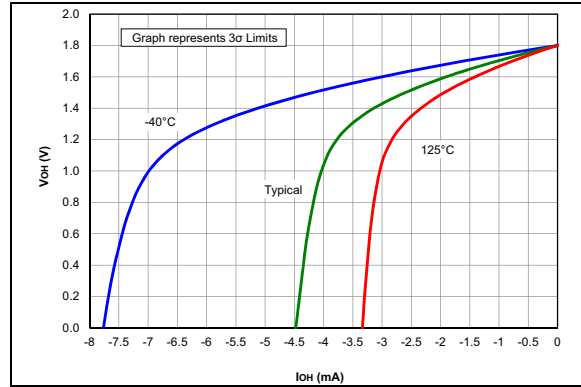
**FIGURE 38-96:**  $V_{OH}$  Vs.  $I_{OH}$  Over Temperature,  $V_{DD} = 3.0V$

# PIC16LF15354/55

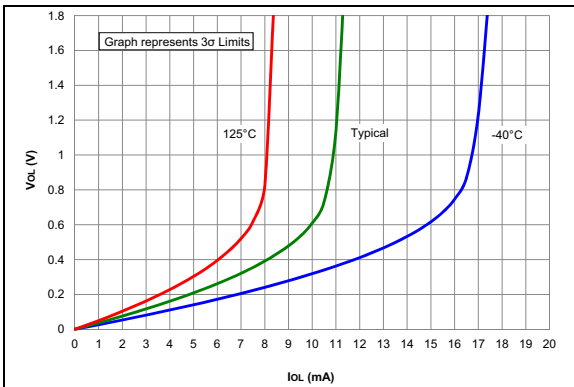
Note: Unless otherwise noted,  $V_{IN} = 5V$ ,  $F_{OSC} = 300\text{ kHz}$ ,  $C_{IN} = 0.1\ \mu\text{F}$ ,  $T_A = 25^\circ\text{C}$ .



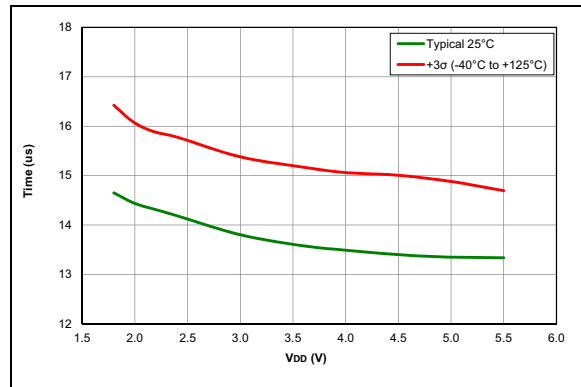
**FIGURE 38-97:**  $V_{OL}$  Vs.  $I_{OL}$  Over Temperature,  $V_{DD} = 3.0V$



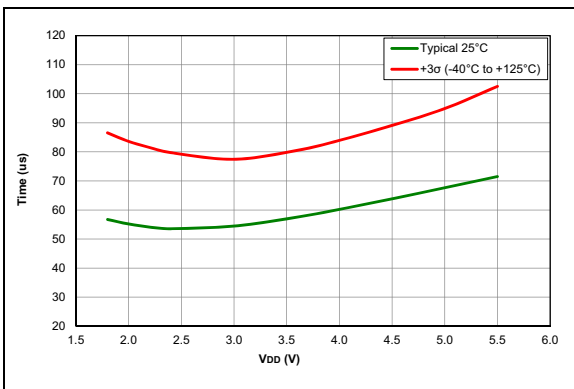
**FIGURE 38-98:**  $V_{OH}$  Vs.  $I_{OH}$  Over Temperature,  $V_{DD} = 1.8V$ , PIC16LF15354/55 devices only



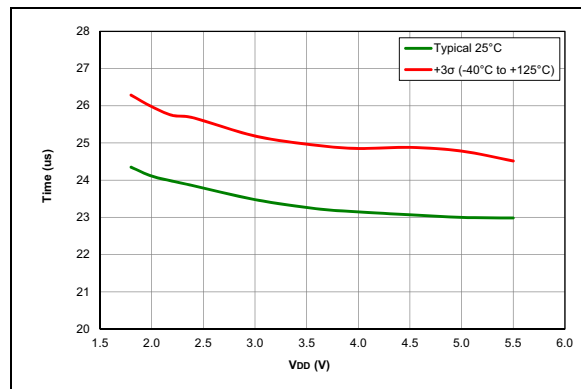
**FIGURE 38-99:**  $V_{OL}$  Vs.  $I_{OL}$  Over Temperature,  $V_{DD} = 3.0V$ , PIC16LF15354/55 devices only



**FIGURE 38-100:** Wake From Sleep,  $V_{REGPM} = 0$ ,  $HFINTOSC = 4\text{ MHz}$ , PIC16LF15354/55 devices only



**FIGURE 38-101:** Wake from Sleep,  $V_{REGPM} = 1$ ,  $HFINTOSC = 4\text{ MHz}$ , PIC16LF15354/55 devices only

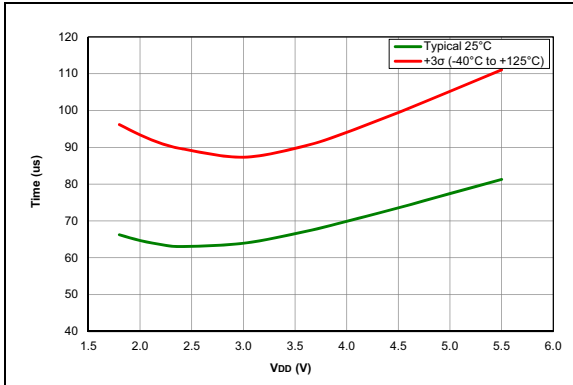


**FIGURE 38-102:** Wake from Sleep,  $V_{REGPM} = 1$ ,  $HFINTOSC = 16\text{ MHz}$ , PIC16LF15354/55 devices only

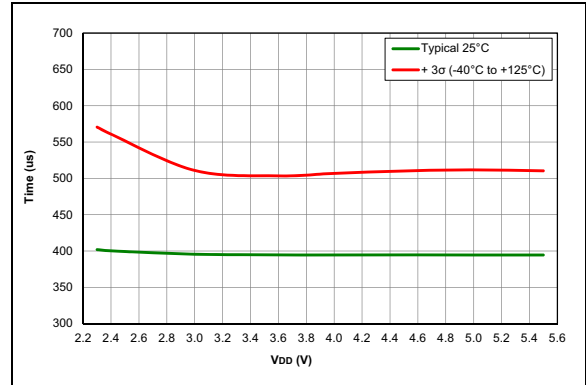


# PIC16LF15354/55

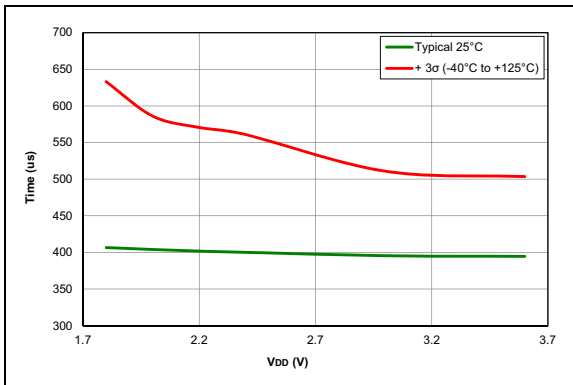
**Note:** Unless otherwise noted,  $V_{IN} = 5V$ ,  $F_{OSC} = 300\text{ kHz}$ ,  $C_{IN} = 0.1\ \mu F$ ,  $T_A = 25^\circ C$ .



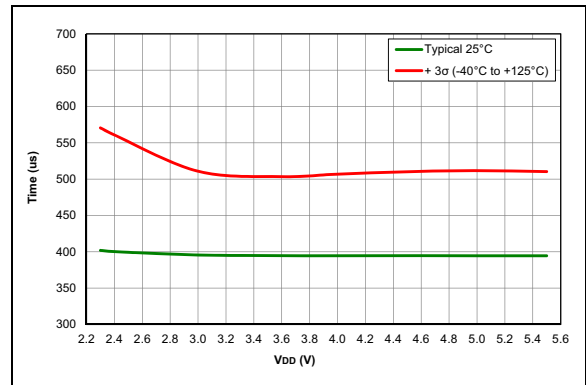
**FIGURE 38-103:** Wake from Sleep,  $V_{REGPM} = 1$ ,  $HFINTOSC = 16\text{ MHz}$ , PIC16F15354/55 devices only



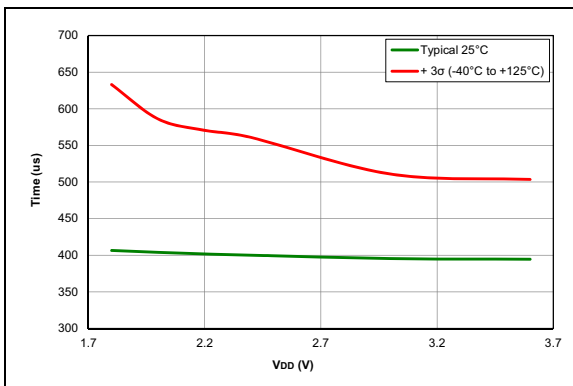
**FIGURE 38-104:** Wake from Sleep,  $V_{REGPM} = 1$ , PIC16F15354/55 devices only



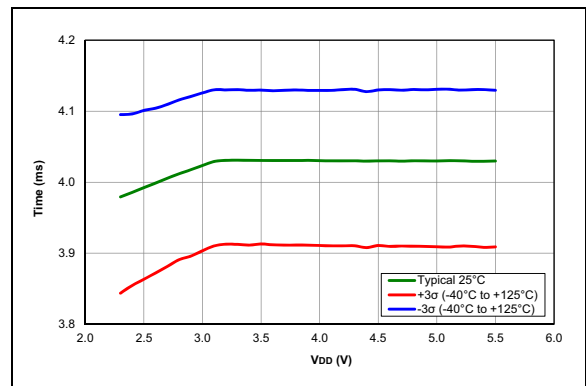
**FIGURE 38-105:** Wake from Sleep, PIC16F15354/55 devices only



**FIGURE 38-106:** Wake from Sleep,  $V_{REGPM} = 1$ ,  $LFINTOSC$ , PIC16F15354/55 devices only



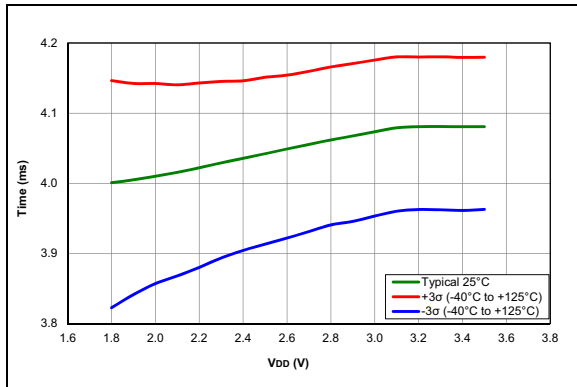
**FIGURE 38-107:** Wake from Sleep,  $LFINTOSC$ , PIC16F15354/55 devices only



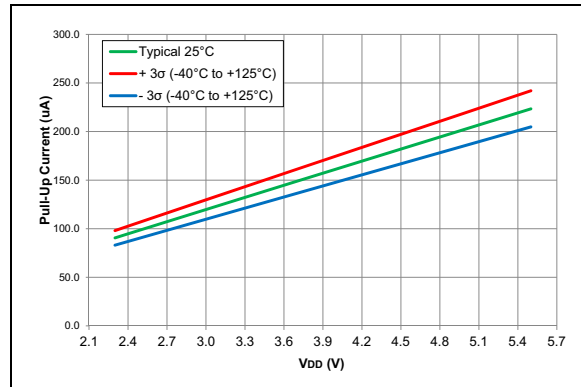
**FIGURE 38-108:** Watchdog Timer Time-out Period, PIC16F15354/55 devices only

# PIC16LF15354/55

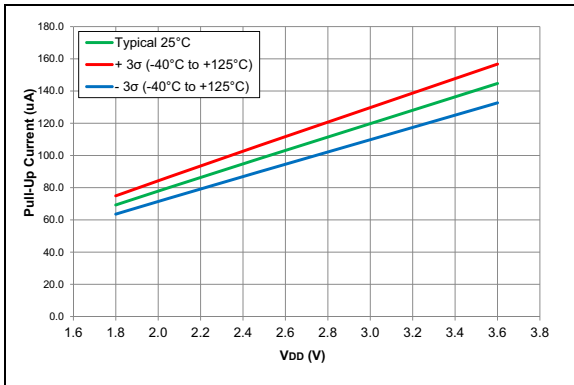
Note: Unless otherwise noted,  $V_{IN} = 5V$ ,  $F_{OSC} = 300\text{ kHz}$ ,  $C_{IN} = 0.1\ \mu\text{F}$ ,  $T_A = 25^\circ\text{C}$ .



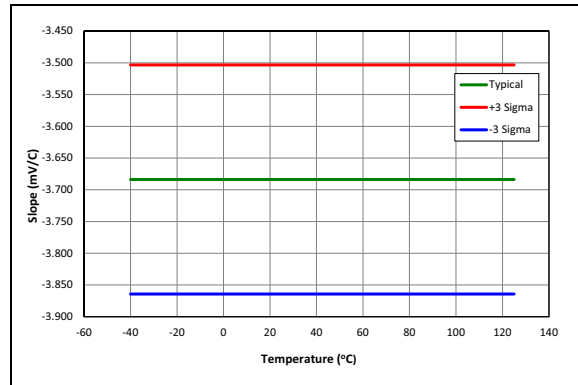
**FIGURE 38-109:** Watchdog Timer Time-out Period, PIC16LF15354/55 devices only



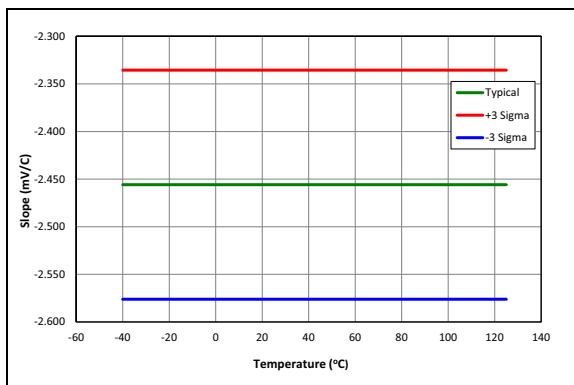
**FIGURE 38-110:** Weak Pull-up Current, PIC16LF15354/55 devices only



**FIGURE 38-111:** Weak Pull-up Current, PIC16LF15354/55 devices only



**FIGURE 38-112:** High Range Temperature Indicator Voltage Sensitivity Across Temperature



**FIGURE 38-113:** Low Range Temperature Indicator Voltage Sensitivity Across Temperature

## 39.0 DEVELOPMENT SUPPORT

The PIC<sup>®</sup> microcontrollers (MCU) and dsPIC<sup>®</sup> digital signal controllers (DSC) are supported with a full range of software and hardware development tools:

- Integrated Development Environment
  - MPLAB<sup>®</sup> X IDE Software
  - MPLAB<sup>®</sup> XPRESS IDE Software
- Compilers/Assemblers/Linkers
  - MPLAB XC Compiler
  - MPASM<sup>™</sup> Assembler
  - MPLINK<sup>™</sup> Object Linker/  
MPLIB<sup>™</sup> Object Librarian
  - MPLAB Assembler/Linker/Librarian for  
Various Device Families
- Simulators
  - MPLAB X SIM Software Simulator
- Emulators
  - MPLAB REAL ICE<sup>™</sup> In-Circuit Emulator
- In-Circuit Debuggers/Programmers
  - MPLAB ICD 3
  - PICKit<sup>™</sup> 3
- Device Programmers
  - MPLAB PM3 Device Programmer
- Low-Cost Demonstration/Development Boards,  
Evaluation Kits and Starter Kits
- Third-party development tools

## 39.1 MPLAB X Integrated Development Environment Software

The MPLAB X IDE is a single, unified graphical user interface for Microchip and third-party software, and hardware development tool that runs on Windows<sup>®</sup>, Linux and Mac OS<sup>®</sup> X. Based on the NetBeans IDE, MPLAB X IDE is an entirely new IDE with a host of free software components and plug-ins for high-performance application development and debugging. Moving between tools and upgrading from software simulators to hardware debugging and programming tools is simple with the seamless user interface.

With complete project management, visual call graphs, a configurable watch window and a feature-rich editor that includes code completion and context menus, MPLAB X IDE is flexible and friendly enough for new users. With the ability to support multiple tools on multiple projects with simultaneous debugging, MPLAB X IDE is also suitable for the needs of experienced users.

Feature-Rich Editor:

- Color syntax highlighting
- Smart code completion makes suggestions and provides hints as you type
- Automatic code formatting based on user-defined rules
- Live parsing

User-Friendly, Customizable Interface:

- Fully customizable interface: toolbars, toolbar buttons, windows, window placement, etc.
- Call graph window

Project-Based Workspaces:

- Multiple projects
- Multiple tools
- Multiple configurations
- Simultaneous debugging sessions

File History and Bug Tracking:

- Local file history feature
- Built-in support for Bugzilla issue tracker

## 39.2 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

## 39.3 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

## 39.4 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/librarian 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

## 39.5 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

## 39.6 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.

## 39.7 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.

## 39.8 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.

## 39.9 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™).

## 39.10 MPLAB PM3 Device Programmer

The MPLAB PM3 Device Programmer is a universal, CE compliant device programmer with programmable voltage verification at VDDMIN and VDDMAX for maximum reliability. It features a large LCD display (128 x 64) for menus and error messages, and a modular, detachable socket assembly to support various package types. The ICSP cable assembly is included as a standard item. In Stand-Alone mode, the MPLAB PM3 Device Programmer can read, verify and program PIC devices without a PC connection. It can also set code protection in this mode. The MPLAB PM3 connects to the host PC via an RS-232 or USB cable. The MPLAB PM3 has high-speed communications and optimized algorithms for quick programming of large memory devices, and incorporates an MMC card for file storage and data applications.

## 39.11 Demonstration/Development Boards, Evaluation Kits, and Starter Kits

A wide variety of demonstration, development and evaluation boards for various PIC MCUs and dsPIC DSCs allows quick application development on fully functional systems. Most boards include prototyping areas for adding custom circuitry and provide application firmware and source code for examination and modification.

The boards support a variety of features, including LEDs, temperature sensors, switches, speakers, RS-232 interfaces, LCD displays, potentiometers and additional EEPROM memory.

The demonstration and development boards can be used in teaching environments, for prototyping custom circuits and for learning about various microcontroller applications.

In addition to the PICDEM™ and dsPICDEM™ demonstration/development board series of circuits, Microchip has a line of evaluation kits and demonstration software for analog filter design, KEELOQ® security ICs, CAN, IrDA®, PowerSmart battery management, SEEVAL® evaluation system, Sigma-Delta ADC, flow rate sensing, plus many more.

Also available are starter kits that contain everything needed to experience the specified device. This usually includes a single application and debug capability, all on one board.

Check the Microchip web page ([www.microchip.com](http://www.microchip.com)) for the complete list of demonstration, development and evaluation kits.

## 39.12 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.

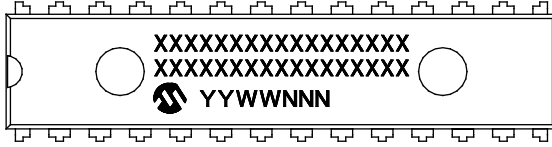
- Device Programmers and Gang Programmers from companies, such as SoftLog and CCS
- Software Tools from companies, such as Gimpel and Trace Systems
- Protocol Analyzers from companies, such as Saleae and Total Phase
- Demonstration Boards from companies, such as MikroElektronika, Digilent® and Olimex
- Embedded Ethernet Solutions from companies, such as EZ Web Lynx, WIZnet and IPLogika®

# PIC16(L)F15354/55

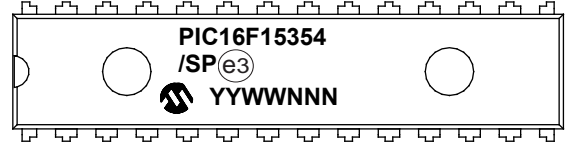
## 40.0 PACKAGING INFORMATION

### 40.1 Package Marking Information

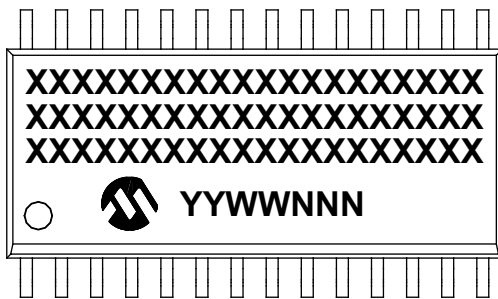
28-Lead SPDIP (.300")



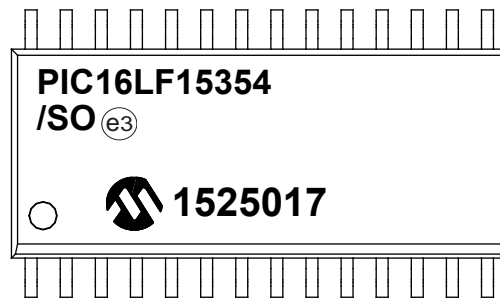
Example



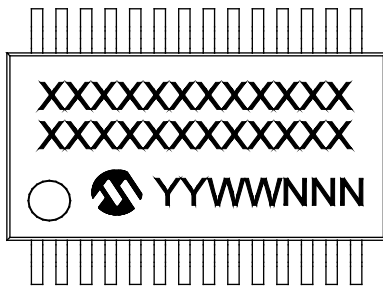
28-Lead SOIC (7.50 mm)



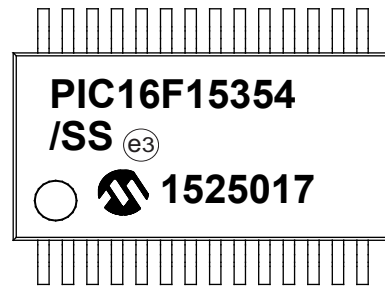
Example



28-Lead SSOP (5.30 mm)



Example



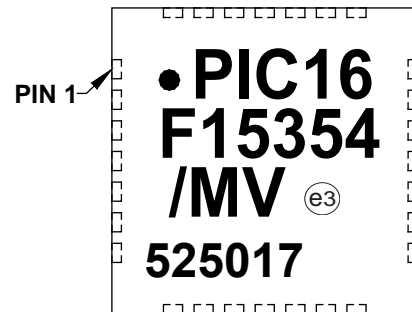
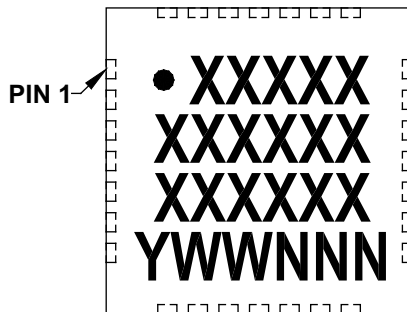
<b>Legend:</b>	XX...X	Customer-specific information
	Y	Year code (last digit of calendar year)
	YY	Year code (last 2 digits of calendar year)
	WW	Week code (week of January 1 is week '01')
	NNN	Alphanumeric traceability code
	*	Pb-free JEDEC® designator for Matte Tin (Sn)
		This package is Pb-free. The 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.

## 40.1 Package Marking Information (Continued)

28-Lead UQFN (4x4x0.5 mm) and (6x6 mm)

Example



<b>Legend:</b>	XX...X	Customer-specific information
	Y	Year code (last digit of calendar year)
	YY	Year code (last 2 digits of calendar year)
	WW	Week code (week of January 1 is week '01')
	NNN	Alphanumeric traceability code
		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.

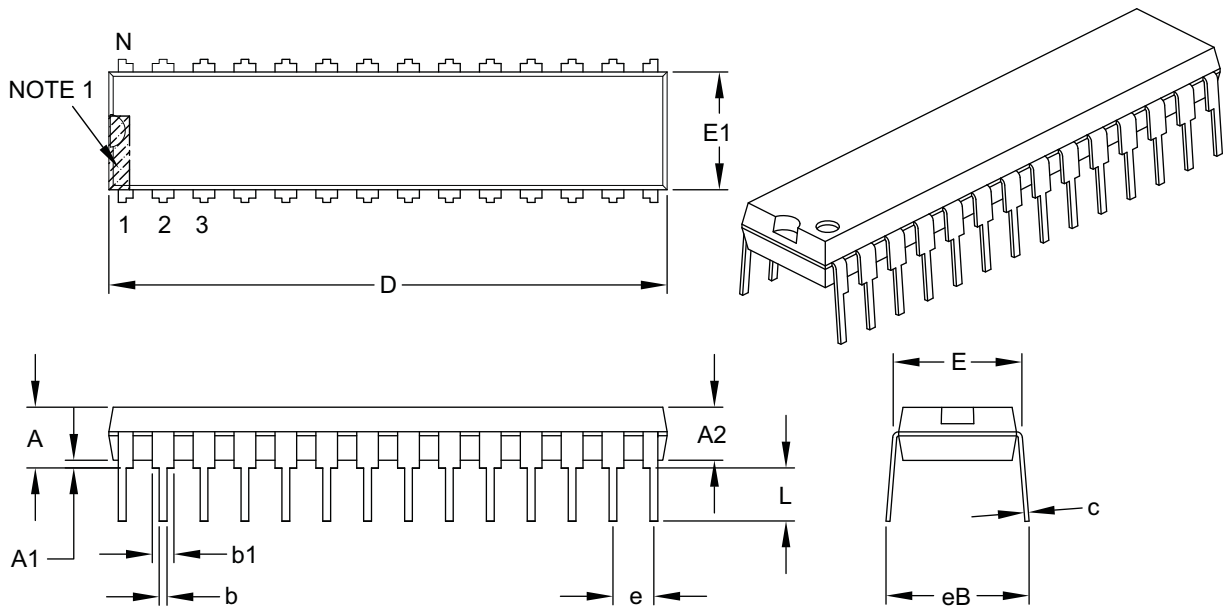


# PIC16(L)F15354/55

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>



Dimension Limits	Units	INCHES		
		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.

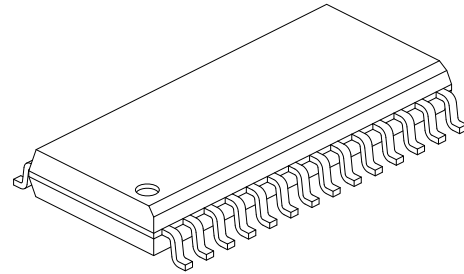
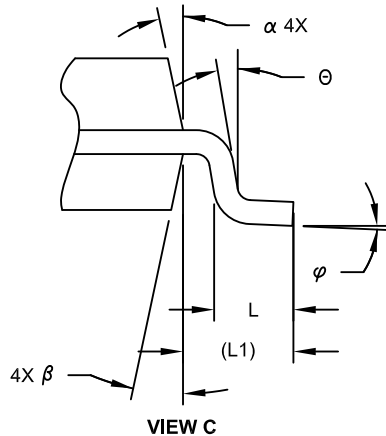
Microchip Technology Drawing C04-070B



# PIC16(L)F15354/55

## 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>



		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:**

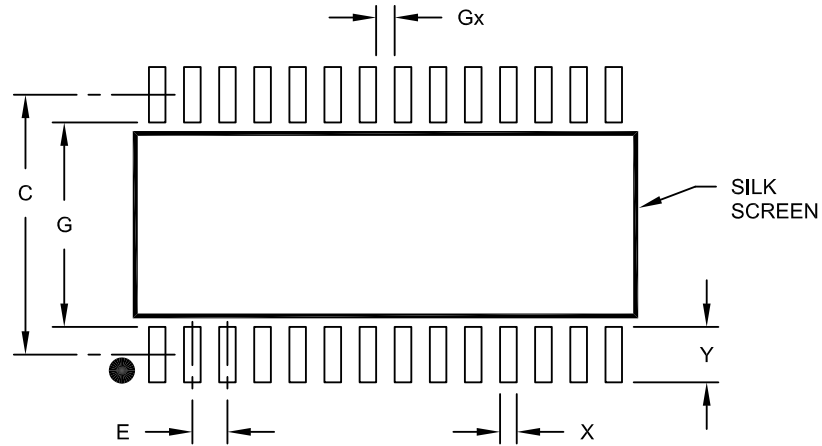
- Pin 1 visual index feature may vary, but must be located within the hatched area.
- § Significant Characteristic
- Dimension D does not include mold flash, protrusions or gate burrs, which shall not exceed 0.15 mm per end. Dimension E1 does not include interlead flash or protrusion, which shall not exceed 0.25 mm per side.
- Dimensioning and tolerancing per ASME Y14.5M  
 BSC: Basic Dimension. Theoretically exact value shown without tolerances.  
 REF: Reference Dimension, usually without tolerance, for information purposes only.
- Datums A & B to be determined at Datum H.

Microchip Technology Drawing C04-052C Sheet 2 of 2

# PIC16(L)F15354/55

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		
		MIN	NOM	MAX
Dimension Limits				
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

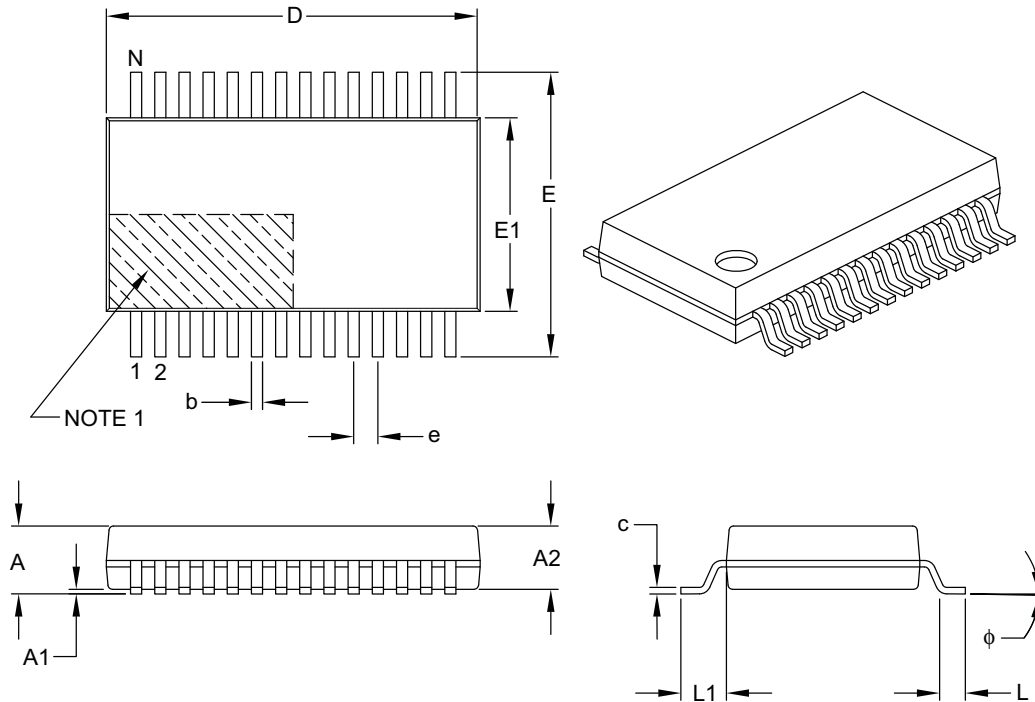
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2052A

# PIC16(L)F15354/55

## 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>



Dimension Limits	Units	MILLIMETERS		
		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	$\phi$	0°	4°	8°
Lead Width	b	0.22	–	0.38

**Notes:**

- Pin 1 visual index feature may vary, but must be located within the hatched area.
- Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.20 mm per side.
- Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

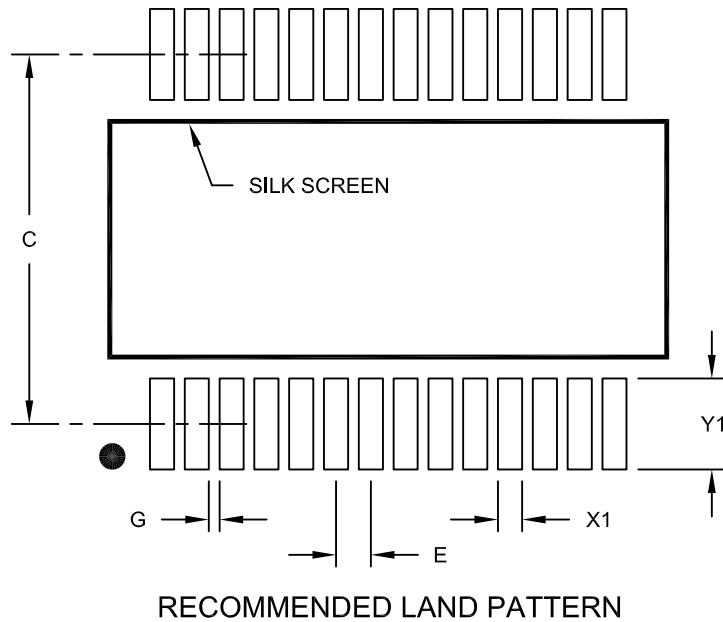
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-073B

# PIC16(L)F15354/55

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>



Dimension Limits	Units	MILLIMETERS		
		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

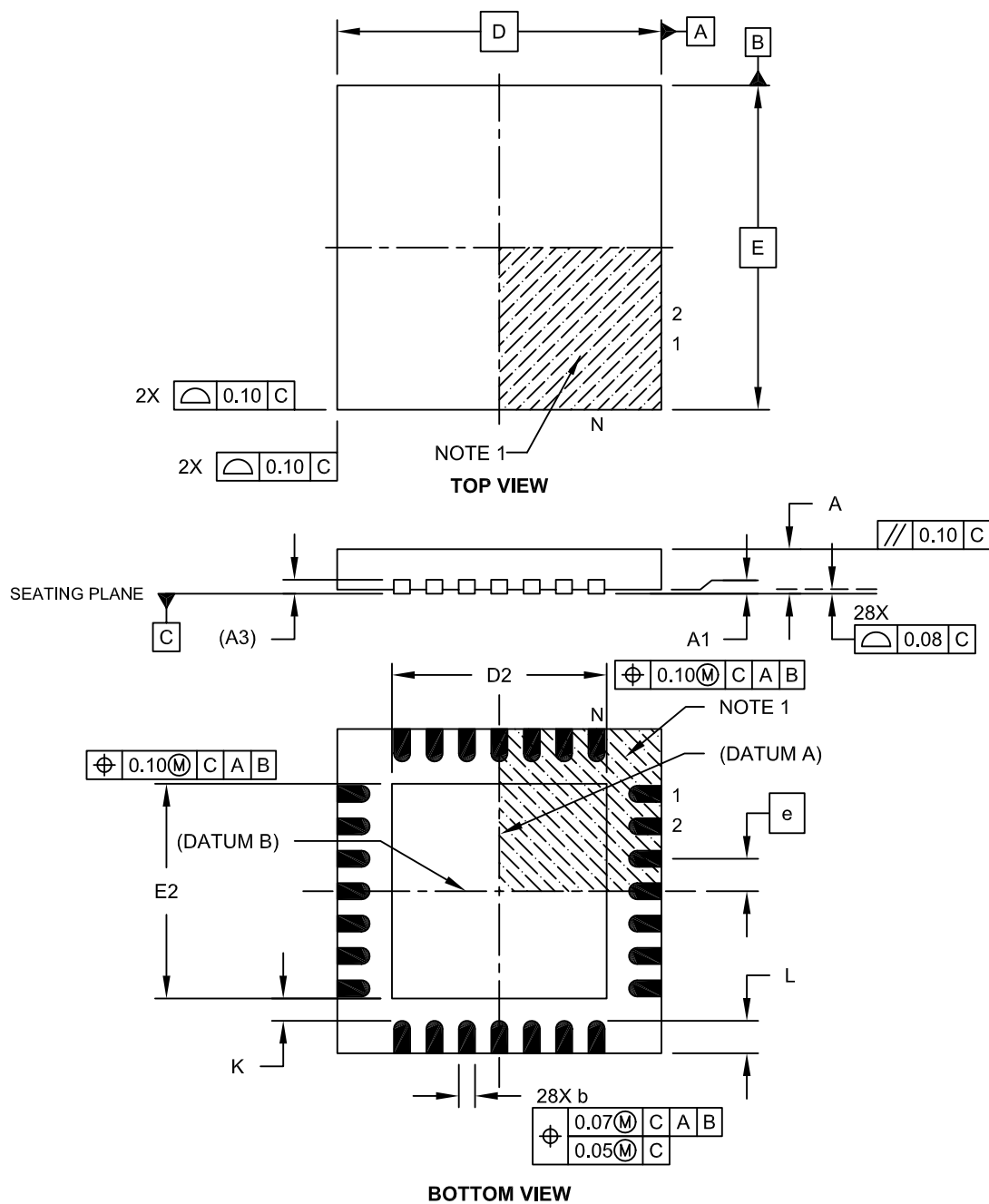
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2073A

# PIC16(L)F15354/55

## 28-Lead Plastic Ultra Thin Quad Flat, No Lead Package (MV) – 4x4x0.5 mm Body [UQFN]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>

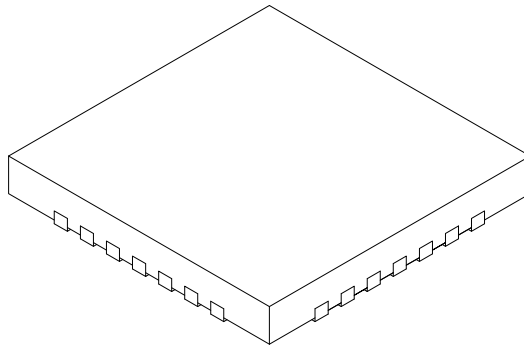


Microchip Technology Drawing C04-152A Sheet 1 of 2

# PIC16(L)F15354/55

## 28-Lead Plastic Ultra Thin Quad Flat, No Lead Package (MV) – 4x4x0.5 mm Body [UQFN]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N	28		
Pitch	e	0.40 BSC		
Overall Height	A	0.45	0.50	0.55
Standoff	A1	0.00	0.02	0.05
Contact Thickness	A3	0.127 REF		
Overall Width	E	4.00 BSC		
Exposed Pad Width	E2	2.55	2.65	2.75
Overall Length	D	4.00 BSC		
Exposed Pad Length	D2	2.55	2.65	2.75
Contact Width	b	0.15	0.20	0.25
Contact Length	L	0.30	0.40	0.50
Contact-to-Exposed Pad	K	0.20	-	-

**Notes:**

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated.
3. Dimensioning and tolerancing per ASME Y14.5M.  
BSC: Basic Dimension. Theoretically exact value shown without tolerances.  
REF: Reference Dimension, usually without tolerance, for information purposes only.

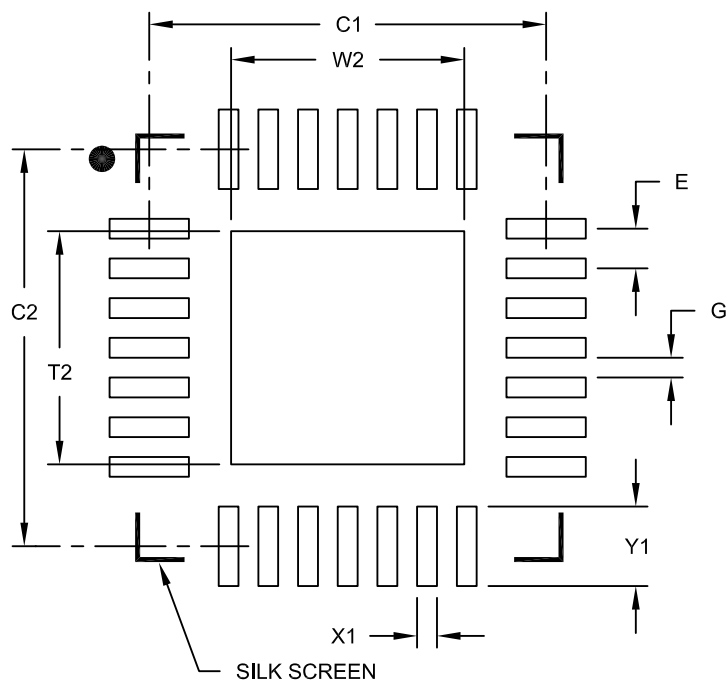
Microchip Technology Drawing C04-152A Sheet 2 of 2



# PIC16(L)F15354/55

28-Lead Ultra Thin Plastic Quad Flat, No Lead Package (MV) - 4x4 mm Body [UQFN]  
With 0.40 mm Contact Length

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



RECOMMENDED LAND PATTERN

Dimension Limits	Units	MILLIMETERS		
		MIN	NOM	MAX
Contact Pitch	E	0.40 BSC		
Optional Center Pad Width	W2			2.35
Optional Center Pad Length	T2			2.35
Contact Pad Spacing	C1		4.00	
Contact Pad Spacing	C2		4.00	
Contact Pad Width (X28)	X1			0.20
Contact Pad Length (X28)	Y1			0.80
Distance Between Pads	G	0.20		

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2152A

## APPENDIX A: DATA SHEET REVISION HISTORY

### **Revision A (07/2016)**

Initial release of the document.

### **Revision B (09/2016)**

Updated SFR table. Updated cover page. Added Figure 4-2. Updated Example 13-5; Figures 4-1, 4-3, 7-1, 8-3, 13-4, 13-8, 17-1, 18-1 and 22-1; Registers 5-3, 5-4, 5-7, 9-5, 10-13, 11-4, 13-1, 13-3, 13-5, 14-5, 14-9, 14-11, 14-13, 14-26, 15-3, and 20-3; Sections 1.0, 3.0, 3.4, 4.1.1, 4.2, 4.2.5, 4.3, 4.3.1, 4.3.2, 4.3.2.1, 5.1, 5.4, 6.1, 6.2, 6.3, 7.0, 7.1, 7.14, 8.4, 8.4.2, 8.5.1, 8.11, 8.12, 10.1, 13.2.2., 13.3.2, 13.3.6, 14.8, 14.8.1, 14.8.2, 16.0, 18.2, 19.5, 29.0, 29.1.1, and 33.0; Tables 3-4, 4-2, 4-4, 4-5, 4-6, 4-7, 4-8, 4-10, 5-1, 7-3, 7-4, 13-2, 13-3, 15-2, 15-3 and 37-6. Added Section 3.2.5. Removed Figure 13-7. General typo and formatting corrections.

### **Revision C (01/2018)**

Updated Table 3, 4-5 and 6-1. Updated Register 5-4. Added second Indirect addressing figure in memory chapter. Updated Equation 19-1 (sensor Temperature) Updated Register 18-1 (FVRCON), Updated 19.2.1.1, Removed Example 19-1 (Temp Sens)

Replaced PGC/PGD with ICSPCLK/ICSPDAT; Revised Section 9.2.2.3 LFINSTOSC; Revised Table 15-1 and 15-2 (PPS Input Signal Routing Options); Table 15-6 Summary of Registers/PPS Module; Revised Example 20-1 ADC Conversion; Added note to Section 20.1.2 Channel Selection; Revised Section 27.0 Timer2 Module; Table 37-11, revised RST06.

Removed Section 20.2.3 (Terminating a Conversation)

Added char graphs. Updated the Electrical Specs chapter: Absolute Maximum Ratings and Tables 37-1, 37-2, 37-3, 37-4, 37-5, 37-6, 37-7, 37-8, 37-11, 37-12, 37-14, 37-16, 37-17, 37-18, 37-19, 37-20, 37-21, 37-22. General typo and formatting corrections.

## THE MICROCHIP WEBSITE

Microchip provides online support via our website at [www.microchip.com](http://www.microchip.com). This website is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the website contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

## CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip website at [www.microchip.com](http://www.microchip.com). Under "Support", click on "Customer Change Notification" and follow the registration instructions.

## CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or Field Application Engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

**Technical support is available through the website at: <http://www.microchip.com/support>**

## PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.

<u>PART NO.</u>	<u>[X]<sup>(1)</sup></u>	-	<u>X</u>	<u>/XX</u>	<u>XXX</u>
Device	Tape and Reel Option		Temperature Range	Package	Pattern
<b>Device:</b>	PIC16F15354, PIC16LF15354 PIC16F15355, PIC16LF15355				
<b>Tape and Reel Option:</b>	Blank = Standard packaging (tube or tray) T = Tape and Reel <sup>(1)</sup>				
<b>Temperature Range:</b>	I = -40°C to +85°C (Industrial) E = -40°C to +125°C (Extended)				
<b>Package:<sup>(2)</sup></b>	MV = 28-lead UQFN 4x4mm SO = 28-lead SOIC SP = 28-lead SPDIP SS = 28-lead SSOP				
<b>Pattern:</b>	QTP, SQTP, Code or Special Requirements (blank otherwise)				

### Examples:

- a) PIC16F15354- E/P  
Extended temperature  
PDIP 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. Check [www.microchip.com/packaging](http://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.

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

*Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELoC® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.*

**QUALITY MANAGEMENT SYSTEM  
CERTIFIED BY DNV  
= ISO/TS 16949 =**

## Trademarks

The Microchip name and logo, the Microchip logo, AnyRate, AVR, AVR logo, AVR Freaks, BeaconThings, BitCloud, chipKIT, chipKIT logo, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, Helder, JukeBlox, KEELoQ, KEELoQ logo, Kleer, LANCheck, LINK MD, maXStylus, maXTouch, MedialB, megaAVR, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, Prochip Designer, QTouch, RightTouch, SAM-BA, SpyNIC, SST, SST Logo, SuperFlash, tinyAVR, UNI/O, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

ClockWorks, The Embedded Control Solutions Company, EtherSynch, Hyper Speed Control, HyperLight Load, IntelliMOS, mTouch, Precision Edge, and Quiet-Wire are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, BodyCom, CodeGuard, CryptoAuthentication, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, EtherGREEN, In-Circuit Serial Programming, ICSP, Inter-Chip Connectivity, JitterBlocker, KleerNet, KleerNet logo, Mindi, MiWi, motorBench, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICKit, PICtail, PureSilicon, QMatrix, RightTouch logo, REAL ICE, Ripple Blocker, SAM-ICE, Serial Quad I/O, SMART-I.S., SQI, SuperSwitcher, SuperSwitcher II, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

Silicon Storage Technology is a registered trademark of Microchip Technology Inc. in other countries.

GestIC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

All other trademarks mentioned herein are property of their respective companies.

© 2018, Microchip Technology Incorporated, All Rights Reserved.

ISBN: 978-1-5224-2562-5



# MICROCHIP

## Worldwide Sales and Service

### AMERICAS

**Corporate Office**  
2355 West Chandler Blvd.  
Chandler, AZ 85224-6199  
Tel: 480-792-7200  
Fax: 480-792-7277  
Technical Support:  
<http://www.microchip.com/support>  
Web Address:  
[www.microchip.com](http://www.microchip.com)

#### Atlanta

Duluth, GA  
Tel: 678-957-9614  
Fax: 678-957-1455

#### Austin, TX

Tel: 512-257-3370

#### Boston

Westborough, MA  
Tel: 774-760-0087  
Fax: 774-760-0088

#### Chicago

Itasca, IL  
Tel: 630-285-0071  
Fax: 630-285-0075

#### Dallas

Addison, TX  
Tel: 972-818-7423  
Fax: 972-818-2924

#### Detroit

Novi, MI  
Tel: 248-848-4000

#### Houston, TX

Tel: 281-894-5983

#### Indianapolis

Noblesville, IN  
Tel: 317-773-8323  
Fax: 317-773-5453  
Tel: 317-536-2380

#### Los Angeles

Mission Viejo, CA  
Tel: 949-462-9523  
Fax: 949-462-9608  
Tel: 951-273-7800

#### Raleigh, NC

Tel: 919-844-7510

#### New York, NY

Tel: 631-435-6000

#### San Jose, CA

Tel: 408-735-9110  
Tel: 408-436-4270

#### Canada - Toronto

Tel: 905-695-1980  
Fax: 905-695-2078

### ASIA/PACIFIC

**Australia - Sydney**  
Tel: 61-2-9868-6733

**China - Beijing**  
Tel: 86-10-8569-7000

**China - Chengdu**  
Tel: 86-28-8665-5511

**China - Chongqing**  
Tel: 86-23-8980-9588

**China - Dongguan**  
Tel: 86-769-8702-9880

**China - Guangzhou**  
Tel: 86-20-8755-8029

**China - Hangzhou**  
Tel: 86-571-8792-8115

**China - Hong Kong SAR**  
Tel: 852-2943-5100

**China - Nanjing**  
Tel: 86-25-8473-2460

**China - Qingdao**  
Tel: 86-532-8502-7355

**China - Shanghai**  
Tel: 86-21-3326-8000

**China - Shenyang**  
Tel: 86-24-2334-2829

**China - Shenzhen**  
Tel: 86-755-8864-2200

**China - Suzhou**  
Tel: 86-186-6233-1526

**China - Wuhan**  
Tel: 86-27-5980-5300

**China - Xian**  
Tel: 86-29-8833-7252

**China - Xiamen**  
Tel: 86-592-2388138

**China - Zhuhai**  
Tel: 86-756-3210040

### ASIA/PACIFIC

**India - Bangalore**  
Tel: 91-80-3090-4444

**India - New Delhi**  
Tel: 91-11-4160-8631

**India - Pune**  
Tel: 91-20-4121-0141

**Japan - Osaka**  
Tel: 81-6-6152-7160

**Japan - Tokyo**  
Tel: 81-3-6880-3770

**Korea - Daegu**  
Tel: 82-53-744-4301

**Korea - Seoul**  
Tel: 82-2-554-7200

**Malaysia - Kuala Lumpur**  
Tel: 60-3-7651-7906

**Malaysia - Penang**  
Tel: 60-4-227-8870

**Philippines - Manila**  
Tel: 63-2-634-9065

**Singapore**  
Tel: 65-6334-8870

**Taiwan - Hsin Chu**  
Tel: 886-3-577-8366

**Taiwan - Kaohsiung**  
Tel: 886-7-213-7830

**Taiwan - Taipei**  
Tel: 886-2-2508-8600

**Thailand - Bangkok**  
Tel: 66-2-694-1351

**Vietnam - Ho Chi Minh**  
Tel: 84-28-5448-2100

### EUROPE

**Austria - Wels**  
Tel: 43-7242-2244-39  
Fax: 43-7242-2244-393

**Denmark - Copenhagen**  
Tel: 45-4450-2828  
Fax: 45-4485-2829

**Finland - Espoo**  
Tel: 358-9-4520-820

**France - Paris**  
Tel: 33-1-69-53-63-20  
Fax: 33-1-69-30-90-79

**Germany - Garching**  
Tel: 49-8931-9700

**Germany - Haan**  
Tel: 49-2129-3766400

**Germany - Heilbronn**  
Tel: 49-7131-67-3636

**Germany - Karlsruhe**  
Tel: 49-721-625370

**Germany - Munich**  
Tel: 49-89-627-144-0  
Fax: 49-89-627-144-44

**Germany - Rosenheim**  
Tel: 49-8031-354-560

**Israel - Ra'anana**  
Tel: 972-9-744-7705

**Italy - Milan**  
Tel: 39-0331-742611  
Fax: 39-0331-466781

**Italy - Padova**  
Tel: 39-049-7625286

**Netherlands - Drunen**  
Tel: 31-416-690399  
Fax: 31-416-690340

**Norway - Trondheim**  
Tel: 47-7289-7561

**Poland - Warsaw**  
Tel: 48-22-3325737

**Romania - Bucharest**  
Tel: 40-21-407-87-50

**Spain - Madrid**  
Tel: 34-91-708-08-90  
Fax: 34-91-708-08-91

**Sweden - Gothenberg**  
Tel: 46-31-704-60-40

**Sweden - Stockholm**  
Tel: 46-8-5090-4654

**UK - Wokingham**  
Tel: 44-118-921-5800  
Fax: 44-118-921-5820