PIC18F2331/2431/4331/4431
Data Sheet

28/40/44-Pin Enhanced Flash Microcontrollers with nanoWatt Technology, High-Performance PWM and A/D

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.

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
ISO/TS 16949:2002

Trademarks

Printed on recycled paper.

ISBN: 978-1-60932-490-2

Microchip received ISO/TS-16949:2002 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

PIC18F2331/2431/4331/4431

28/40/44-Pin Enhanced Flash Microcontrollers with nanoWatt Technology, High-Performance PWM and A/D

14-Bit Power Control PWM Module:

- Up to 4 Channels with Complementary Outputs
- Edge or Center-Aligned Operation
- Flexible Dead-Band Generator
- Hardware Fault Protection Inputs
- Simultaneous Update of Duty Cycle and Period:
  - Flexible Special Event Trigger output

Motion Feedback Module:

- Three Independent Input Capture Channels:
  - Flexible operating modes for period and pulse-width measurement
  - Special Hall sensor interface module
  - Special Event Trigger output to other modules
- Quadrature Encoder Interface:
  - 2-phase inputs and one index input from encoder
  - High and low position tracking with direction status and change of direction interrupt
  - Velocity measurement

High-Speed, 200 ksps 10-Bit A/D Converter:

- Up to 9 Channels
- Simultaneous, Two-Channel Sampling
- Sequential Sampling: 1, 2 or 4 Selected Channels
- Auto-Conversion Capability
- 4-Word FIFO with Selectable Interrupt Frequency
- Selectable External Conversion Triggers
- Programmable Acquisition Time

Flexible Oscillator Structure:

- Four Crystal modes up to 40 MHz
- Two External Clock modes up to 40 MHz
- Internal Oscillator Block:
  - 8 user-selectable frequencies: 31 kHz to 8 MHz
  - OSCTUNE can compensate for frequency drift
- Secondary Oscillator using Timer1 @ 32 kHz
- Fail-Safe Clock Monitor:
  - Allows for safe shutdown of device if clock fails

Power-Managed Modes:

- Run: CPU on, Peripherals on
- Idle: CPU off, Peripherals on
- Sleep: CPU off, Peripherals off
- Ultra Low, 50 nA Input Leakage
- Idle mode Currents Down to 5.8 μA, Typical
- Sleep Current Down to 0.1 μA, Typical
- Timer1 Oscillator, 1.8 μA, Typical, 32 kHz, 2V
- Watchdog Timer (WDT), 2.1 μA, typical
- Oscillator Two-Speed Start-up
  - Fast wake from Sleep and Idle, 1 μs, typical

Peripheral Highlights:

- High-Current Sink/Source 25 mA/25 mA
- Three External Interrupts
- Two Capture/Compare/PWM (CCP) modules
- Enhanced USART module:
  - Supports RS-485, RS-232 and LIN/J2602
  - Auto-wake-up on Start bit
  - Auto-Baud Detect

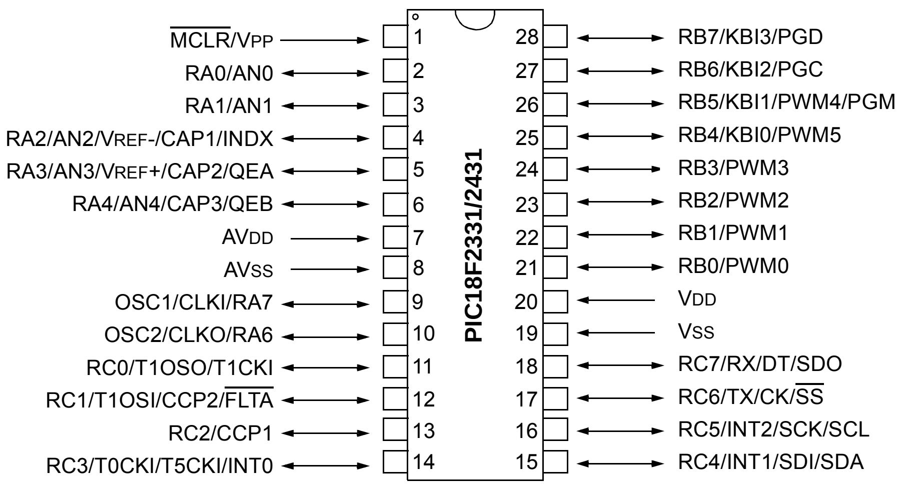
Special Microcontroller Features:

- 100,000 Erase/Write Cycle Enhanced Flash Program Memory, Typical
- 1,000,000 Erase/Write Cycle Data EEPROM Memory, Typical
- Flash/Data EEPROM Retention: 100 Years
- Self-Programmable under Software Control
- Priority Levels for Interrupts
- 8 x 8 Single-Cycle Hardware Multiplier
- Extended Watchdog Timer (WDT):
  - Programmable period from 41 ms to 131s
- Single-Supply In-Circuit Serial Programming™ (ICSP™) via Two Pins
- In-Circuit Debug (ICD) via Two Pins:
  - Drives PWM outputs safely when debugging

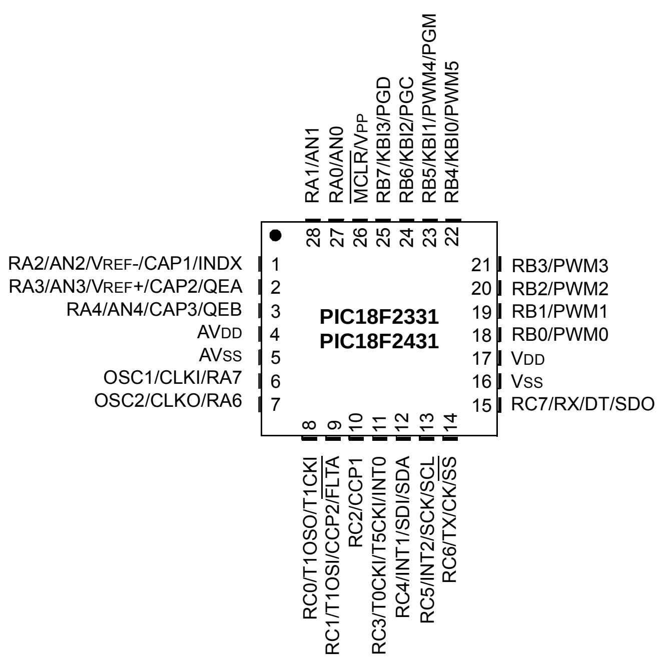
Device	Program Memory		Data Memory		I/O	10-Bit A/D (ch)	CCP	SSP		EUSART	Quadrature Encoder	14-Bit PWM (ch)	Timers 8/16-Bit
	Flash (bytes)	# Single-Word Instructions	SRAM (bytes)	EEPROM (bytes)				SPI	Slave I^2C™				
PIC18F2331	8192	4096	768	256	24	5	2	Y	Y	Y	Y	6	1/3
PIC18F2431	16384	8192	768	256	24	5	2	Y	Y	Y	Y	6	1/3
PIC18F4331	8192	4096	768	256	36	9	2	Y	Y	Y	Y	8	1/3
PIC18F4431	16384	8192	768	256	36	9	2	Y	Y	Y	Y	8	1/3

Pin Diagrams

28-Pin SPDIP, SOIC



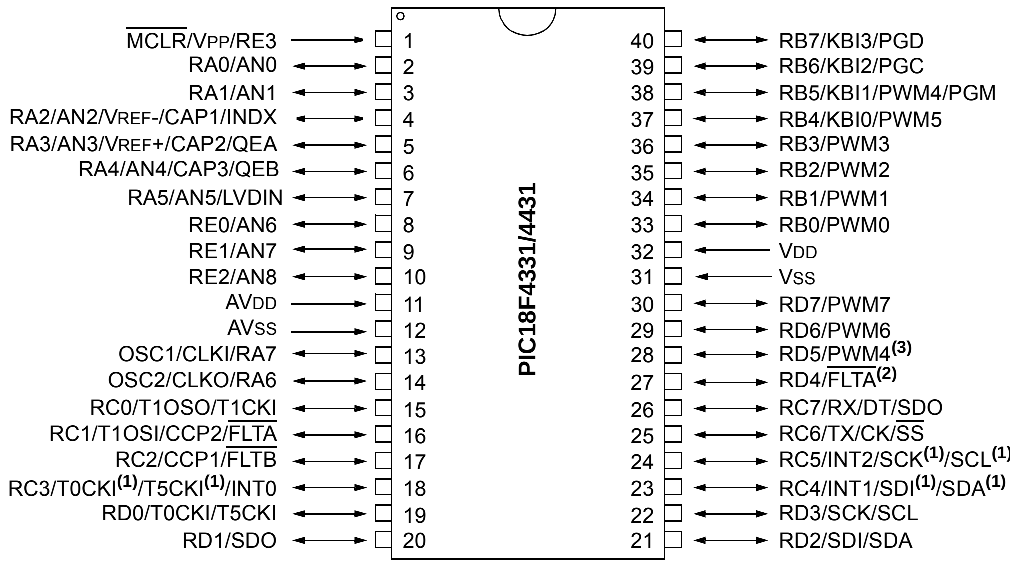
28-Pin QFN(1)



Note 1: For the QFN package, it is recommended that the bottom pad be connected to VSS.

Pin Diagrams (Continued)

40-Pin PDIP



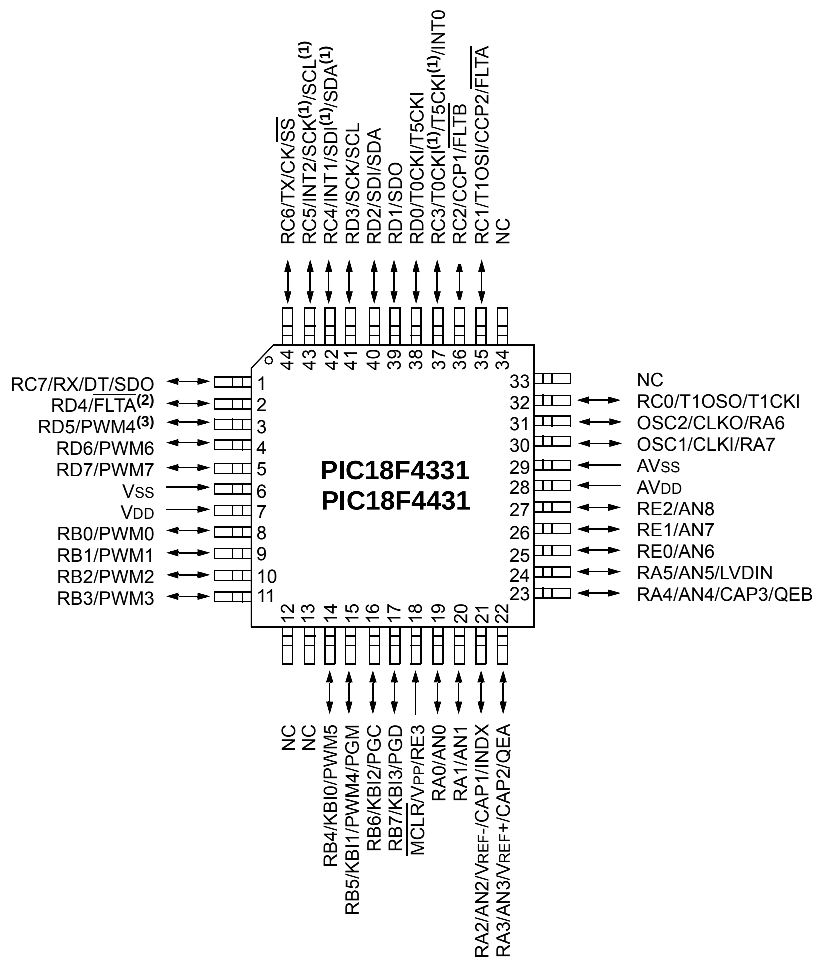
Note 1: RC3 is the alternate pin for T0CKI/T5CKI; RC4 is the alternate pin for SDI/SDA; RC5 is the alternate pin for SCK/SCL.

2: RD4 is the alternate pin for FLTA.

3: RD5 is the alternate pin for PWM4.

Pin Diagrams (Continued)

44-Pin TQFP



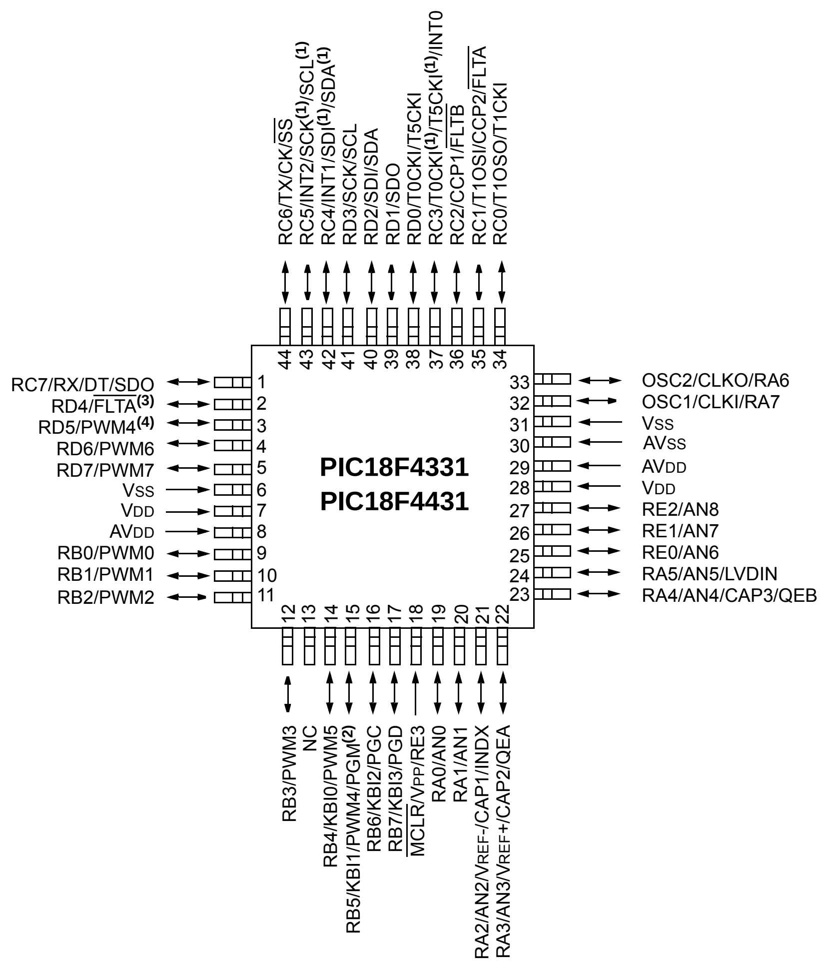
Note 1: RC3 is the alternate pin for T0CKI/T5CKI; RC4 is the alternate pin for SDI/SDA; RC5 is the alternate pin for SCK/SCL.

2: RD4 is the alternate pin for FLTA.

3: RD5 is the alternate pin for PWM4.

Pin Diagrams (Continued)

44-Pin QFN(2)



Note 1: RC3 is the alternate pin for T0CKI/T5CKI; RC4 is the alternate pin for SDI/SDA; RC5 is the alternate pin for SCK/SCL.

2: For the QFN package, it is recommended that the bottom pad be connected to VSS.

3: RD4 is the alternate pin for FLTA.

4: RD5 is the alternate pin for PWM4.

Table of Contents

TO OUR VALUED CUSTOMERS

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

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at **docerrors@microchip.com** or fax the **Reader Response Form** in the back of this data sheet to (480) 792-4150. We welcome your feedback.

Most Current Data Sheet

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

http://www.microchip.com

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000A is version A of document DS30000).

Errata

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

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

- Microchip's Worldwide Web site; 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 web site at **www.microchip.com** to receive the most current information on all of our products.

NOTES:

1.0 DEVICE OVERVIEW

This document contains device-specific information for the following devices:

- PIC18F2331
- PIC18F2431
- PIC18F4331
- PIC18F4431
- PIC18LF2331
- PIC18LF2431
- PIC18LF4331
- PIC18LF4431

This family offers the advantages of all PIC18 microcontrollers – namely, high computational performance at an economical price, with the addition of high-endurance enhanced Flash program memory and a high-speed 10-bit A/D Converter. On top of these features, the PIC18F2331/2431/4331/4431 family introduces design enhancements that make these microcontrollers a logical choice for many high-performance, power control and motor control applications. These special peripherals include:

- 14-Bit Resolution Power Control PWM module (PCPWM) with Programmable Dead-Time Insertion
- Motion Feedback Module (MFM), including a 3-Channel Input Capture (IC) module and Quadrature Encoder Interface (QEI)
- High-Speed 10-Bit A/D Converter (HSADC)

The PCPWM can generate up to eight complementary PWM outputs with dead-band time insertion. Overdrive current is detected by off-chip analog comparators or the digital Fault inputs ($\overline{FLTA}$, $\overline{FLTB}$).

The MFM Quadrature Encoder Interface provides precise rotor position feedback and/or velocity measurement. The MFM 3x input capture or external interrupts can be used to detect the rotor state for electrically commutated motor applications using Hall sensor feedback, such as BLDC motor drives.

PIC18F2331/2431/4331/4431 devices also feature Flash program memory and an internal RC oscillator with built-in LP modes.

1.1 New Core Features

1.1.1 nanoWatt Technology

All of the devices in the PIC18F2331/2431/4331/4431 family incorporate a range of features that can significantly reduce power consumption during operation. Key items include:

- **Alternate Run Modes:** By clocking the controller from the Timer1 source or the internal oscillator block, power consumption during code execution can be reduced by as much as 90%.
- **Multiple Idle Modes:** The controller can also run with its CPU core disabled, but the peripherals are still active. In these states, power consumption can be reduced even further, to as little as 4% of normal operation requirements.
- **On-the-Fly Mode Switching:** The power-managed modes are invoked by user code during operation, allowing the user to incorporate power-saving ideas into their application's software design.
- **Lower Consumption in Key Modules:** The power requirements for both Timer1 and the Watchdog Timer have been reduced by up to 80%, with typical values of 1.1 and 2.1 μA, respectively.

1.1.2 MULTIPLE OSCILLATOR OPTIONS AND FEATURES

All of the devices in the PIC18F2331/2431/4331/4431 family offer nine different oscillator options, allowing users a wide range of choices in developing application hardware. These include:

- Four Crystal modes, using crystals or ceramic resonators.
- Two External Clock modes, offering the option of using two pins (oscillator input and a divide-by-4 clock output) or one pin (oscillator input, with the second pin reassigned as general I/O).
- Two External RC Oscillator modes, with the same pin options as the External Clock modes.
- An internal oscillator block, which provides an 8 MHz clock and an INTRC source (approximately 31 kHz, stable over temperature and V_{DD}), as well as a range of 6 user-selectable clock frequencies (from 125 kHz to 4 MHz) for a total of 8 clock frequencies.
- A Phase Lock Loop (PLL) frequency multiplier, available to both the High-Speed Crystal and Internal Oscillator modes, which allows clock speeds of up to 40 MHz. Used with the internal oscillator, the PLL gives users a complete selection of clock speeds, from 31 kHz to 32 MHz – all without using an external crystal or clock circuit.
- **Fail-Safe Clock Monitor:** This option constantly monitors the main clock source against a reference signal provided by the internal oscillator. If a clock failure occurs, the controller is switched to the internal oscillator block, allowing for continued low-speed operation or a safe application shutdown.
- **Two-Speed Start-up:** This option allows the internal oscillator to serve as the clock source from Power-on Reset, or wake-up from Sleep mode, until the primary clock source is available.

1.2 Other Special Features

- **Memory Endurance:** The enhanced Flash cells for both program memory and data EEPROM are rated to last for many thousands of erase/write cycles – up to 100,000 for program memory and 1,000,000 for EEPROM. Data retention without refresh is conservatively estimated to be greater than 100 years.
- **Self-Programmability:** These devices can write to their own program memory spaces under internal software control. By using a bootloader routine located in the protected boot block at the top of program memory, it becomes possible to create an application that can update itself in the field.
- **Power Control PWM Module:** In PWM mode, this module provides 1, 2 or 4 modulated outputs for controlling half-bridge and full-bridge drivers. Other features include auto-shutdown on Fault detection and auto-restart to reactivate outputs once the condition has cleared.
- **Enhanced Addressable USART:** This serial communication module is capable of standard RS-232 operation and provides support for the LIN/J2602 bus protocol. Other enhancements include automatic baud rate detection and a 16-bit Baud Rate Generator for improved resolution. When the microcontroller is using the internal oscillator block, the EUSART provides stable operation for applications that talk to the outside world without using an external crystal (or its accompanying power requirement).
- **Extended Watchdog Timer (WDT):** This enhanced version incorporates a 16-bit prescaler, allowing an extended time-out range that is stable across operating voltage and temperature. See **Section 26.0 "Electrical Characteristics"** for time-out periods.
- **High-Speed 10-Bit A/D Converter:** This module incorporates programmable acquisition time, allowing for a channel to be selected and a conversion to be initiated without waiting for a sampling period and thus, reducing code overhead.
- **Motion Feedback Module** (MFM): This module features a Quadrature Encoder Interface (QEI) and an Input Capture (IC) module. The QEI accepts two phase inputs (QEA, QEB) and one index input (INDX) from an incremental encoder. The QEI supports high and low precision position tracking, direction status and change of direction interrupt and velocity measurement. The input capture features 3 channels of independent input capture with Timer5 as the time base, a Special Event Trigger to other modules and an adjustable noise filter on each IC input.
- **Extended Watchdog Timer (WDT):** This enhanced version incorporates a 16-bit prescaler, allowing a time-out range from 4 ms to over 2 minutes, that is stable across operating voltage and temperature.

1.3 Details on Individual Family Members

Devices in the PIC18F2331/2431/4331/4431 family are available in 28-pin (PIC18F2331/2431) and 40/44-pin (PIC18F4331/4431) packages. The block diagram for the two groups is shown in Figure 1-1.

The devices are differentiated from each other in three ways:

1. Flash program memory (8 Kbytes for PIC18F2331/4331 devices, 16 Kbytes for PIC18F2431/4431).
2. A/D channels (5 for PIC18F2331/2431 devices, 9 for PIC18F4331/4431 devices).
3. I/O ports (3 bidirectional ports on PIC18F2331/2431 devices, 5 bidirectional ports on PIC18F4331/4431 devices).

All other features for devices in this family are identical. These are summarized in Table 1-1.

The pinouts for all devices are listed in Table 1-2 and Table 1-3.

Like all Microchip PIC18 devices, members of the PIC18F2331/2431/4331/4431 family are available as both standard and low-voltage devices. Standard devices with Enhanced Flash memory, designated with an "F" in the part number (such as PIC18**F**2331), accommodate an operating VDD range of 4.2V to 5.5V. Low-voltage parts, designated by "LF" (such as PIC18**LF**2331), function over an extended VDD range of 2.0V to 5.5V.

TABLE 1-1: DEVICE FEATURES

Features	PIC18F2331	PIC18F2431	PIC18F4331	PIC18F4431
Operating Frequency	DC – 40 MHz	DC – 40 MHz	DC – 40 MHz	DC – 40 MHz
Program Memory (Bytes)	8192	16384	8192	16384
Program Memory (Instructions)	4096	8192	4096	8192
Data Memory (Bytes)	768	768	768	768
Data EEPROM Memory (Bytes)	256	256	256	256
Interrupt Sources	22	22	34	34
I/O Ports	Ports A, B, C	Ports A, B, C	Ports A, B, C, D, E	Ports A, B, C, D, E
Timers	4	4	4	4
Capture/Compare/PWM modules	2	2	2	2
14-Bit Power Control PWM	(6 Channels)	(6 Channels)	(8 Channels)	(8 Channels)
Motion Feedback Module (Input Capture/Quadrature Encoder Interface)	1 QEI or 3x IC	1 QEI or 3x IC	1 QEI or 3x IC	1 QEI or 3x IC
Serial Communications	SSP, Enhanced USART	SSP, Enhanced USART	SSP, Enhanced USART	SSP, Enhanced USART
10-Bit High-Speed Analog-to-Digital Converter module	5 Input Channels	5 Input Channels	9 Input Channels	9 Input Channels
Resets (and Delays)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), $\overline{MCLR}$ (optional), WDT	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), $\overline{MCLR}$ (optional), WDT	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), $\overline{MCLR}$ (optional), WDT	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), $\overline{MCLR}$ (optional), WDT
Programmable Low-Voltage Detect	Yes	Yes	Yes	Yes
Programmable Brown-out Reset	Yes	Yes	Yes	Yes
Instruction Set	75 Instructions	75 Instructions	75 Instructions	75 Instructions
Packages	28-pin SPDIP 28-pin SOIC 28-pin QFN	28-pin SPDIP 28-pin SOIC 28-pin QFN	40-pin PDIP 44-pin TQFP 44-pin QFN	40-pin PDIP 44-pin TQFP 44-pin QFN

FIGURE 1-1: PIC18F2331/2431 (28-PIN) BLOCK DIAGRAM

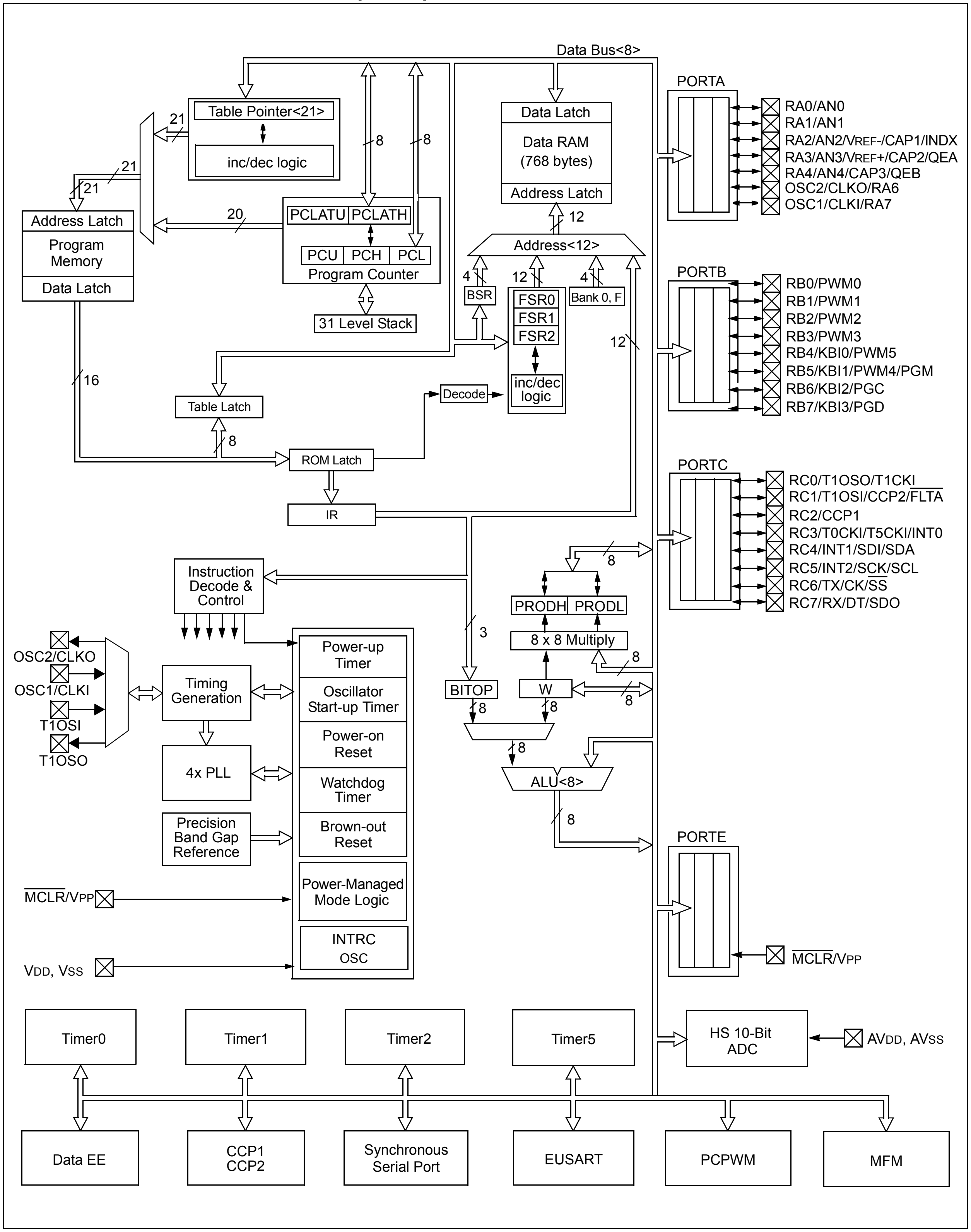
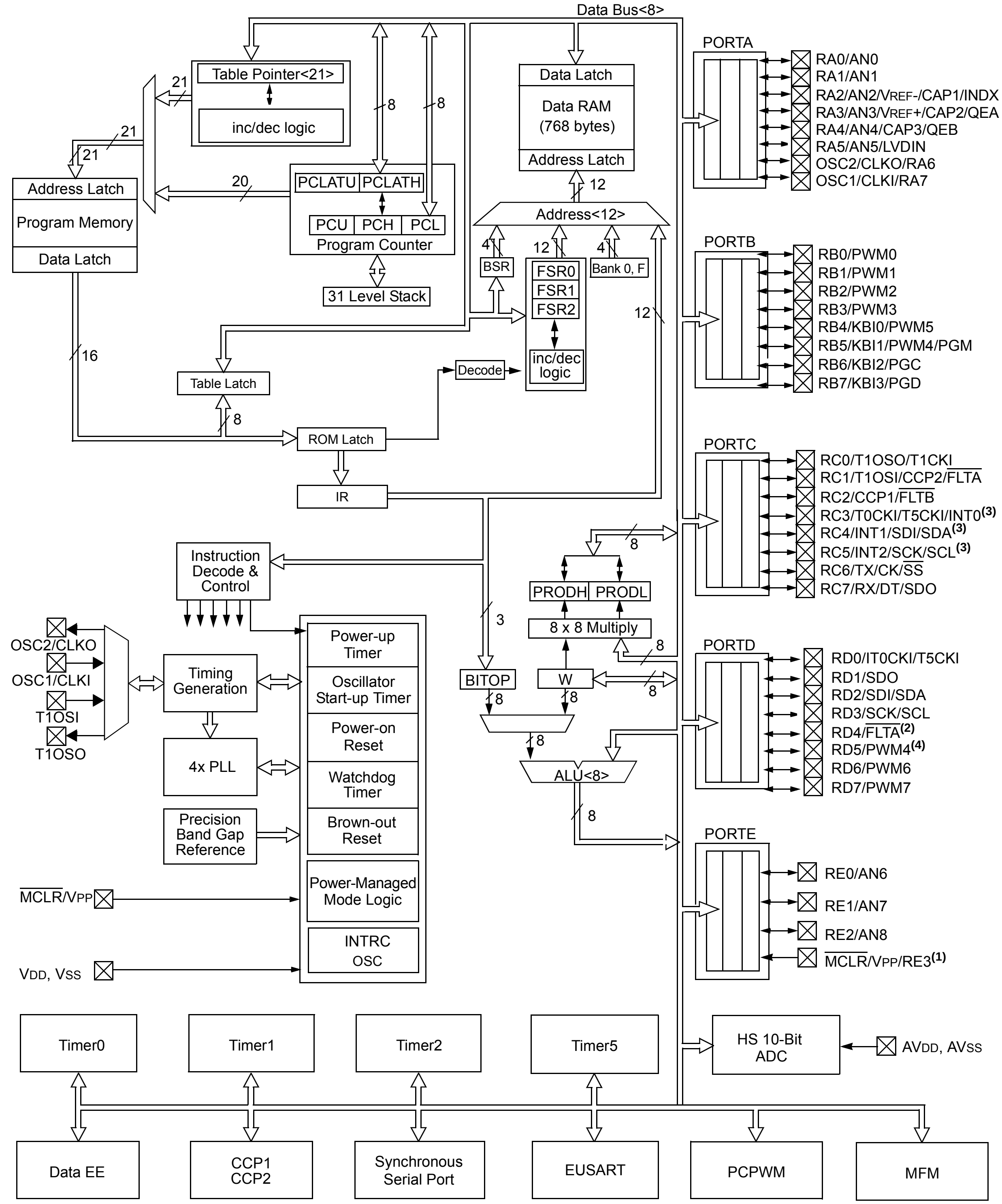


FIGURE 1-2: PIC18F4331/4431 (40/44-PIN) BLOCK DIAGRAM



Note 1: RE3 is available only when $\overline{MCLR}$ is disabled.

2: RD4 is the alternate pin for $\overline{FLTA}$.

3: RC3, RC4 and RC5 are alternate pins for T0CKI/T5CKI, SDI/SDA, SCK/SCL, respectively.

4: RD5 is the alternate pin for PWM4.

TABLE 1-2: PIC18F2331/2431 PINOUT I/O DESCRIPTIONS

Pin Name	Pin Number		Pin Type	Buffer Type	Description
	SPDIP, SOIC	QFN			
$\overline{MCLR}$/VPP	1	26			Master Clear (input) or programming voltage (input).
$\overline{MCLR}$			I	ST	Master Clear (Reset) input. This pin is an active-low Reset to the device.
VPP			P		High-voltage ICSP™ programming enable pin.
OSC1/CLKI/RA7	9	6			Oscillator crystal or external clock input.
OSC1			I	ST	Oscillator crystal input or external clock source input. ST buffer when configured in RC mode; CMOS otherwise.
CLKI			I	CMOS	External clock source input. Always associated with pin function OSC1. (See related OSC1/CLKI, OSC2/CLKO pins.)
RA7			I/O	TTL	General purpose I/O pin.
OSC2/CLKO/RA6	10	7			Oscillator crystal or clock output.
OSC2			O	—	Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode.
CLKO			O	—	In RC mode, OSC2 pin outputs CLKO, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate.
RA6			I/O	TTL	General purpose I/O pin.
					PORTA is a bidirectional I/O port.
RA0/AN0	2	27			
RA0			I/O	TTL	Digital I/O.
AN0			I	Analog	Analog Input 0.
RA1/AN1	3	28			
RA1			I/O	TTL	Digital I/O.
AN1			I	Analog	Analog Input 1.
RA2/AN2/VREF-/CAP1/INDX	4	1			
RA2			I/O	TTL	Digital I/O.
AN2			I	Analog	Analog Input 2.
VREF-			I	Analog	A/D reference voltage (low) input.
CAP1			I	ST	Input Capture Pin 1.
INDX			I	ST	Quadrature Encoder Interface index input pin.
RA3/AN3/VREF+/CAP2/QEA	5	2			
RA3			I/O	TTL	Digital I/O.
AN3			I	Analog	Analog Input 3.
VREF+			I	Analog	A/D reference voltage (high) input.
CAP2			I	ST	Input Capture Pin 2.
QEA			I	ST	Quadrature Encoder Interface Channel A input pin.
RA4/AN4/CAP3/QEB	6	3			
RA4			I/O	TTL	Digital I/O.
AN4			I	Analog	Analog Input 4.
CAP3			I	ST	Input Capture Pin 3.
QEB			I	ST	Quadrature Encoder Interface Channel B input pin.

Legend: TTL = TTL compatible input; ST = Schmitt Trigger input with CMOS levels; O = Output; CMOS = CMOS compatible input or output; I = Input; P = Power

TABLE 1-2: PIC18F2331/2431 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number SPDIP, SOIC	Pin Number QFN	Pin Type	Buffer Type	Description
					PORTB is a bidirectional I/O port. PORTB can be software programmed for internal weak pull-ups on all inputs.
RB0/PWM0	21	18			
RB0			I/O	TTL	Digital I/O.
PWM0			O	TTL	PWM Output 0.
RB1/PWM1	22	19			
RB1			I/O	TTL	Digital I/O.
PWM1			O	TTL	PWM Output 1.
RB2/PWM2	23	20			
RB2			I/O	TTL	Digital I/O.
PWM2			O	TTL	PWM Output 2.
RB3/PWM3	24	21			
RB3			I/O	TTL	Digital I/O.
PWM3			O	TTL	PWM Output 3.
RB4/KBI0/PWM5	25	22			
RB4			I/O	TTL	Digital I/O.
KBI0			I	TTL	Interrupt-on-change pin.
PWM5			O	TTL	PWM Output 5.
RB5/KBI1/PWM4/PGM	26	23			
RB5			I/O	TTL	Digital I/O.
KBI1			I	TTL	Interrupt-on-change pin.
PWM4			O	TTL	PWM Output 4.
PGM			I/O	ST	Single-Supply ICSP™ Programming entry pin.
RB6/KBI2/PGC	27	24			
RB6			I/O	TTL	Digital I/O.
KBI2			I	TTL	Interrupt-on-change pin.
PGC			I/O	ST	In-Circuit Debugger and ICSP programming clock pin.
RB7/KBI3/PGD	28	25			
RB7			I/O	TTL	Digital I/O.
KBI3			I	TTL	Interrupt-on-change pin.
PGD			I/O	ST	In-Circuit Debugger and ICSP programming data pin.

Legend: TTL = TTL compatible input; ST = Schmitt Trigger input with CMOS levels; O = Output; CMOS = CMOS compatible input or output; I = Input; P = Power

TABLE 1-2: PIC18F2331/2431 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number SPDIP, SOIC	Pin Number QFN	Pin Type	Buffer Type	Description
					PORTC is a bidirectional I/O port.
RC0/T1OSO/T1CKI	11	8			
RC0			I/O	ST	Digital I/O.
T1OSO			O	—	Timer1 oscillator output.
T1CKI			I	ST	Timer1 external clock input.
RC1/T1OSI/CCP2/$\overline{\text{FLTA}}$	12	9			
RC1			I/O	ST	Digital I/O.
T1OSI			I	Analog	Timer1 oscillator input.
CCP2			I/O	ST	Capture 2 input, Compare 2 output, PWM2 output.
$\overline{\text{FLTA}}$			I	ST	Fault interrupt input pin.
RC2/CCP1	13	10			
RC2			I/O	ST	Digital I/O.
CCP1			I/O	ST	Capture 1 input/Compare 1 output/PWM1 output.
RC3/T0CKI/T5CKI/INT0	14	11			
RC3			I/O	ST	Digital I/O.
T0CKI			I	ST	Timer0 alternate clock input.
T5CKI			I	ST	Timer5 alternate clock input.
INT0			I	ST	External Interrupt 0.
RC4/INT1/SDI/SDA	15	12			
RC4			I/O	ST	Digital I/O.
INT1			I	ST	External Interrupt 1.
SDI			I	ST	SPI data in.
SDA			I/O	I^2C	I^2C™ data I/O.
RC5/INT2/SCK/SCL	16	13			
RC5			I/O	ST	Digital I/O.
INT2			I	ST	External Interrupt 2.
SCK			I/O	ST	Synchronous serial clock input/output for SPI mode.
SCL			I/O	I^2C	Synchronous serial clock input/output for I^2C mode.
RC6/TX/CK/$\overline{\text{SS}}$	17	14			
RC6			I/O	ST	Digital I/O.
TX			O	—	EUSART asynchronous transmit.
CK			I/O	ST	EUSART synchronous clock (see related RX/DT).
$\overline{\text{SS}}$			I	TTL	SPI slave select input.
RC7/RX/DT/SDO	18	15			
RC7			I/O	ST	Digital I/O.
RX			I	ST	EUSART asynchronous receive.
DT			I/O	ST	EUSART synchronous data (see related TX/CK).
SDO			O	—	SPI data out.
VSS	8, 19	5, 16	P	—	Ground reference for logic and I/O pins.
VDD	7, 20	4, 17	P	—	Positive supply for logic and I/O pins.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; I = Input; O = Output; P = Power

TABLE 1-3: PIC18F4331/4431 PINOUT I/O DESCRIPTIONS

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	PDIP	TQFP	QFN			
$\overline{\text{MCLR}}$/V_{PP}/RE3	1	18	18			Master Clear (input) or programming voltage (input).
$\overline{\text{MCLR}}$				I	ST	Master Clear (Reset) input. This pin is an active-low Reset to the device.
V_{PP}				P		Programming voltage input.
RE3				I	ST	Digital input. Available only when $\overline{\text{MCLR}}$ is disabled.
OSC1/CLKI/RA7	13	30	32			Oscillator crystal or external clock input.
OSC1				I	ST	Oscillator crystal input or external clock source input. ST buffer when configured in RC mode; CMOS otherwise.
CLKI				I	CMOS	External clock source input. Always associated with pin function OSC1. (See related OSC1/CLKI, OSC2/CLKO pins.)
RA7				I/O	TTL	General purpose I/O pin.
OSC2/CLKO/RA6	14	31	33			Oscillator crystal or clock output.
OSC2				O	—	Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode.
CLKO				O	—	In RC mode, OSC2 pin outputs CLKO, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate.
RA6				I/O	TTL	General purpose I/O pin.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; I = Input; O = Output; P = Power

Note 1: RC3 is the alternate pin for T0CKI/T5CKI; RC4 is the alternate pin for SDI/SDA; RC5 is the alternate pin for SCK/SCL; RC7 is the alternate pin for SDO.

2: RD4 is the alternate pin for $\overline{\text{FLTA}}$.

3: RD5 is the alternate pin for PWM4.

TABLE 1-3: PIC18F4331/4431 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	PDIP	TQFP	QFN			
						PORTA is a bidirectional I/O port.
RA0/AN0	2	19	19			
RA0				I/O	TTL	Digital I/O.
AN0				I	Analog	Analog Input 0.
RA1/AN1	3	20	20			
RA1				I/O	TTL	Digital I/O.
AN1				I	Analog	Analog Input 1.
RA2/AN2/VREF-/CAP1/INDX	4	21	21			
RA2				I/O	TTL	Digital I/O.
AN2				I	Analog	Analog Input 2.
VREF-				I	Analog	A/D reference voltage (low) input.
CAP1				I	ST	Input Capture Pin 1.
INDX				I	ST	Quadrature Encoder Interface index input pin.
RA3/AN3/VREF+/CAP2/QEA	5	22	22			
RA3				I/O	TTL	Digital I/O.
AN3				I	Analog	Analog Input 3.
VREF+				I	Analog	A/D reference voltage (high) input.
CAP2				I	ST	Input Capture Pin 2.
QEA				I	ST	Quadrature Encoder Interface Channel A input pin.
RA4/AN4/CAP3/QEB	6	23	23			
RA4				I/O	TTL	Digital I/O.
AN4				I	Analog	Analog Input 4.
CAP3				I	ST	Input Capture Pin 3.
QEB				I	ST	Quadrature Encoder Interface Channel B input pin.
RA5/AN5/LVDIN	7	24	24			
RA5				I/O	TTL	Digital I/O.
AN5				I	Analog	Analog Input 5.
LVDIN				I	Analog	Low-Voltage Detect input.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; I = Input; O = Output; P = Power

Note 1: RC3 is the alternate pin for T0CKI/T5CKI; RC4 is the alternate pin for SDI/SDA; RC5 is the alternate pin for SCK/SCL; RC7 is the alternate pin for SDO.

2: RD4 is the alternate pin for $\overline{\text{FLTA}}$.

3: RD5 is the alternate pin for PWM4.

TABLE 1-3: PIC18F4331/4431 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number PDIP	Pin Number TQFP	Pin Number QFN	Pin Type	Buffer Type	Description
						PORTB is a bidirectional I/O port. PORTB can be software programmed for internal weak pull-ups on all inputs.
RB0/PWM0	33	8	9			
RB0				I/O	TTL	Digital I/O.
PWM0				O	TTL	PWM Output 0.
RB1/PWM1	34	9	10			
RB1				I/O	TTL	Digital I/O.
PWM1				O	TTL	PWM Output 1.
RB2/PWM2	35	10	11			
RB2				I/O	TTL	Digital I/O.
PWM2				O	TTL	PWM Output 2.
RB3/PWM3	36	11	12			
RB3				I/O	TTL	Digital I/O.
PWM3				O	TTL	PWM Output 3.
RB4/KBI0/PWM5	37	14	14			
RB4				I/O	TTL	Digital I/O.
KBI0				I	TTL	Interrupt-on-change pin.
PWM5				O	TTL	PWM Output 5.
RB5/KBI1/PWM4/PGM	38	15	15			
RB5				I/O	TTL	Digital I/O.
KBI1				I	TTL	Interrupt-on-change pin.
PWM4				O	TTL	PWM Output 4.
PGM				I/O	ST	Single-Supply ICSP™ Programming entry pin.
RB6/KBI2/PGC	39	16	16			
RB6				I/O	TTL	Digital I/O.
KBI2				I	TTL	Interrupt-on-change pin.
PGC				I/O	ST	In-Circuit Debugger and ICSP programming clock pin.
RB7/KBI3/PGD	40	17	17			
RB7				I/O	TTL	Digital I/O.
KBI3				I	TTL	Interrupt-on-change pin.
PGD				I/O	ST	In-Circuit Debugger and ICSP programming data pin.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; I = Input; O = Output; P = Power

Note 1: RC3 is the alternate pin for T0CKI/T5CKI; RC4 is the alternate pin for SDI/SDA; RC5 is the alternate pin for SCK/SCL; RC7 is the alternate pin for SDO.

2: RD4 is the alternate pin for $\overline{\text{FLTA}}$.

3: RD5 is the alternate pin for PWM4.

TABLE 1-3: PIC18F4331/4431 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	PDIP	TQFP	QFN			
						PORTC is a bidirectional I/O port.
RC0/T1OSO/T1CKI	15	32	34			
RC0				I/O	ST	Digital I/O.
T1OSO				O	—	Timer1 oscillator output.
T1CKI				I	ST	Timer1 external clock input.
RC1/T1OSI/CCP2/ $\overline{FLTA}$	16	35	35			
RC1				I/O	ST	Digital I/O.
T1OSI				I	CMOS	Timer1 oscillator input.
CCP2				I/O	ST	Capture 2 input, Compare 2 output, PWM2 output.
$\overline{FLTA}$				I	ST	Fault interrupt input pin.
RC2/CCP1/$\overline{FLTB}$	17	36	36			
RC2				I/O	ST	Digital I/O.
CCP1				I/O	ST	Capture 1 input/Compare 1 output/PWM1 output.
$\overline{FLTB}$				I	ST	Fault interrupt input pin.
RC3/T0CKI/T5CKI/ INT0	18	37	37			
RC3				I/O	ST	Digital I/O.
T0CKI(1)				I	ST	Timer0 alternate clock input.
T5CKI(1)				I	ST	Timer5 alternate clock input.
INT0				I	ST	External Interrupt 0.
RC4/INT1/SDI/SDA	23	42	42			
RC4				I/O	ST	Digital I/O.
INT1				I	ST	External Interrupt 1.
SDI(1)				I	ST	SPI data in.
SDA(1)				I/O	I^2C	I^2C™ data I/O.
RC5/INT2/SCK/SCL	24	43	43			
RC5				I/O	ST	Digital I/O.
INT2				I	ST	External Interrupt 2.
SCK(1)				I/O	ST	Synchronous serial clock input/output for SPI mode.
SCL(1)				I/O	I^2C	Synchronous serial clock input/output for I^2C mode.
RC6/TX/CK/$\overline{SS}$	25	44	44			
RC6				I/O	ST	Digital I/O.
TX				O	—	EUSART asynchronous transmit.
CK				I/O	ST	EUSART synchronous clock (see related RX/DT).
$\overline{SS}$				I	ST	SPI slave select input.
RC7/RX/DT/SDO	26	1	1			
RC7				I/O	ST	Digital I/O.
RX				I	ST	EUSART asynchronous receive.
DT				I/O	ST	EUSART synchronous data (see related TX/CK).
SDO(1)				O	—	SPI data out.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; I = Input; O = Output; P = Power

Note 1: RC3 is the alternate pin for T0CKI/T5CKI; RC4 is the alternate pin for SDI/SDA; RC5 is the alternate pin for SCK/SCL; RC7 is the alternate pin for SDO.

2: RD4 is the alternate pin for $\overline{FLTA}$.

3: RD5 is the alternate pin for PWM4.

TABLE 1-3: PIC18F4331/4431 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	PDIP	TQFP	QFN			
						PORTD is a bidirectional I/O port.
RD0/T0CKI/T5CKI	19	38	38			
RD0				I/O	ST	Digital I/O.
T0CKI				I	ST	Timer0 external clock input.
T5CKI				I	ST	Timer5 input clock.
RD1/SDO	20	39	39			
RD1				I/O	ST	Digital I/O.
SDO(1)				O	—	SPI data out.
RD2/SDI/SDA	21	40	40			
RD2				I/O	ST	Digital I/O.
SDI(1)				I	ST	SPI data in.
SDA(1)				I/O	ST	I^2C™ data I/O.
RD3/SCK/SCL	22	41	41			
RD3				I/O	ST	Digital I/O.
SCK(1)				I/O	ST	Synchronous serial clock input/output for SPI mode.
SCL(1)				I/O	ST	Synchronous serial clock input/output for I^2C mode.
RD4/$\overline{\text{FLTA}}$	27	2	2			
RD4				I/O	ST	Digital I/O.
$\overline{\text{FLTA}}$(2)				I	ST	Fault interrupt input pin.
RD5/PWM4	28	3	3			
RD5				I/O	ST	Digital I/O.
PWM4(3)				O	TTL	PWM Output 4.
RD6/PWM6	29	4	4			
RD6				I/O	ST	Digital I/O.
PWM6				O	TTL	PWM Output 6.
RD7/PWM7	30	5	5			
RD7				I/O	ST	Digital I/O.
PWM7				O	TTL	PWM Output 7.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output
ST = Schmitt Trigger input with CMOS levels; I = Input
O = Output; P = Power

Note 1: RC3 is the alternate pin for T0CKI/T5CKI; RC4 is the alternate pin for SDI/SDA; RC5 is the alternate pin for SCK/SCL; RC7 is the alternate pin for SDO.

2: RD4 is the alternate pin for $\overline{\text{FLTA}}$.

3: RD5 is the alternate pin for PWM4.

TABLE 1-3: PIC18F4331/4431 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number PDIP	TQFP	QFN	Pin Type	Buffer Type	Description
						PORTE is a bidirectional I/O port.
RE0/AN6	8	25	25			
RE0				I/O	ST	Digital I/O.
AN6				I	Analog	Analog Input 6.
RE1/AN7	9	26	26			
RE1				I/O	ST	Digital I/O.
AN7				I	Analog	Analog Input 7.
RE2/AN8	10	27	27			
RE2				I/O	ST	Digital I/O.
AN8				I	Analog	Analog Input 8.
VSS	12, 31	6, 29	6, 30, 31	P	—	Ground reference for logic and I/O pins.
VDD	11, 32	7, 28	7, 8, 28, 29	P	—	Positive supply for logic and I/O pins.
NC	—	12, 13, 33, 34	13	NC	NC	No connect.

Legend: TTL = TTL compatible input; CMOS = CMOS compatible input or output; ST = Schmitt Trigger input with CMOS levels; I = Input; O = Output; P = Power

Note 1: RC3 is the alternate pin for T0CKI/T5CKI; RC4 is the alternate pin for SDI/SDA; RC5 is the alternate pin for SCK/SCL; RC7 is the alternate pin for SDO.

2: RD4 is the alternate pin for $\overline{\text{FLTA}}$.

3: RD5 is the alternate pin for PWM4.

2.0 GUIDELINES FOR GETTING STARTED WITH PIC18F MICROCONTROLLERS

2.1 Basic Connection Requirements

Getting started with the PIC18F2331/2431/4331/4431 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"**)
- All AVDD and AVSS pins, regardless of whether or not the analog device features are used (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:

- PGC/PGD 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"**)

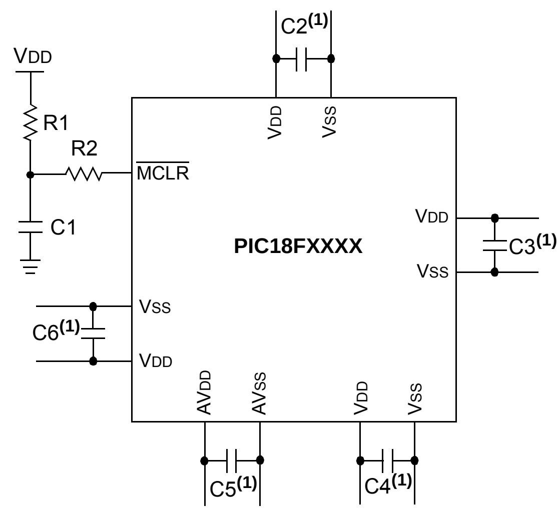
Additionally, the following pins may be required:

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

Note:	The AVDD and AVSS pins must always be connected, regardless of whether any of the analog modules are being used.

The minimum mandatory connections are shown in Figure 2-1.

FIGURE 2-1: RECOMMENDED MINIMUM CONNECTIONS



Key (all values are recommendations):

C1 through C6: 0.1 µF, 20V ceramic

R1: 10 kΩ

R2: 100Ω to 470Ω

Note 1: The example shown is for a PIC18F device with five VDD/VSS and AVDD/AVSS pairs. Other devices may have more or less pairs; adjust the number of decoupling capacitors appropriately.

2.2 Power Supply Pins

2.2.1 DECOUPLING CAPACITORS

The use of decoupling capacitors on every pair of power supply pins, such as VDD, VSS, AVDD and AVSS, is required.

Consider the following criteria when using decoupling capacitors:

- **Value and type of capacitor:** A 0.1 μF (100 nF), 10-20V 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 μF to 0.001 μF. 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 μF in parallel with 0.001 μF).
- **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 μF to 47 μF.

2.2.3 CONSIDERATIONS WHEN USING BOR

When the Brown-out Reset (BOR) feature is enabled, a sudden change in VDD may result in a spontaneous BOR event. This can happen when the microcontroller is operating under normal operating conditions, regardless of what the BOR set point has been programmed to, and even if VDD does not approach the set point. The precipitating factor in these BOR events is a rise or fall in VDD with a slew rate faster than 0.15V/μs.

An application that incorporates adequate decoupling between the power supplies will not experience such rapid voltage changes. Additionally, the use of an electrolytic tank capacitor across VDD and VSS, as described above, will be helpful in preventing high slew rate transitions.

If the application has components that turn on or off, and share the same VDD circuit as the microcontroller, the BOR can be disabled in software by using the SBOREN bit before switching the component. Afterwards, allow a small delay before re-enabling the BOR. By doing this, it is ensured that the BOR is disabled during the interval that might cause high slew rate changes of VDD.

Note:	Not all devices incorporate software BOR control. See **Section 5.0 "Reset"** for device-specific information.

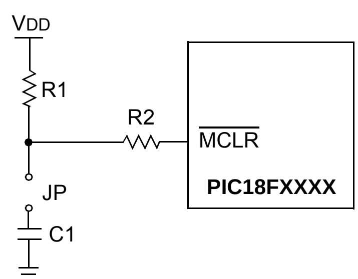
2.3 Master Clear ($\overline{MCLR}$) Pin

The $\overline{MCLR}$ 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 V_{DD} 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 $\overline{MCLR}$ 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 $\overline{MCLR}$ 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 $\overline{MCLR}$ pin should be placed within 0.25 inch (6 mm) of the pin.

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



Note 1: R1 ≤ 10 kΩ is recommended. A suggested starting value is 10 kΩ. Ensure that the $\overline{MCLR}$ pin V_{IH} and V_{IL} specifications are met.

2: R2 ≤ 470Ω will limit any current flowing into $\overline{MCLR}$ from the external capacitor, C, in the event of $\overline{MCLR}$ pin breakdown, due to Electrostatic Discharge (ESD) or Electrical Overstress (EOS). Ensure that the $\overline{MCLR}$ pin V_{IH} and V_{IL} specifications are met.

2.4 ICSP Pins

The PGC and PGD 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 PGC and PGD 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., PGCx/PGDx 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 25.0 "Development Support"**.

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 3.0 "Oscillator Configurations"** 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 2-4. 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).

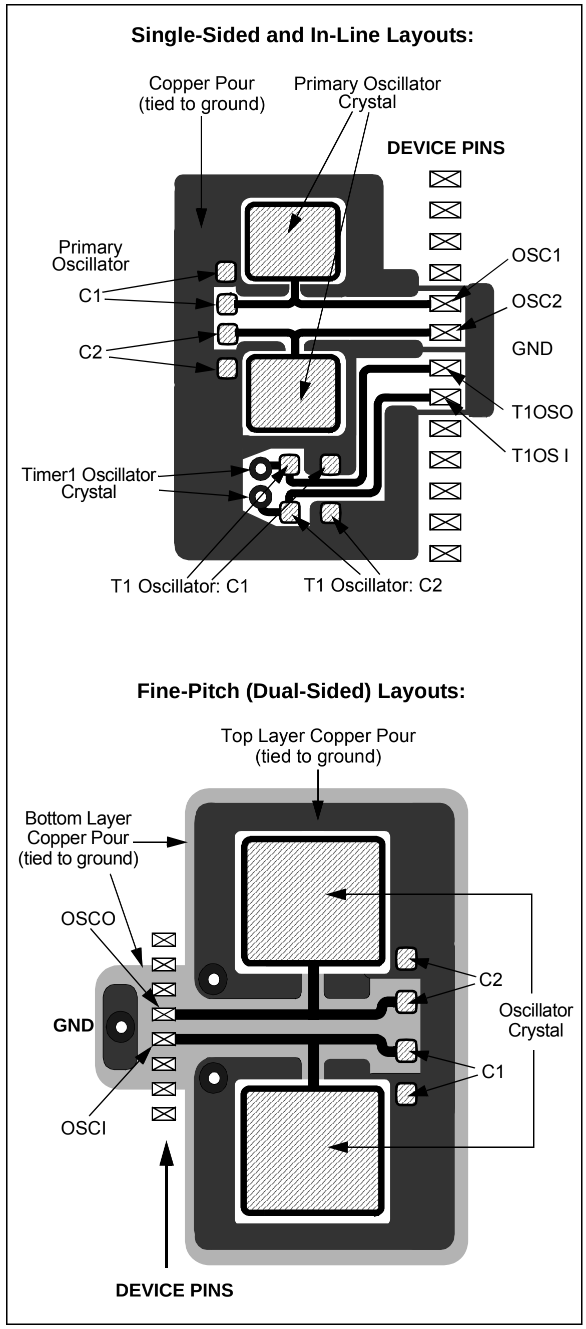
For additional information and design guidance on oscillator circuits, please refer to these Microchip Application Notes, available at the corporate web site (www.microchip.com):

- *AN826, "Crystal Oscillator Basics and Crystal Selection for rfPIC™ 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 VSS on unused pins and drive the output to logic low.

FIGURE 2-3: SUGGESTED PLACEMENT OF THE OSCILLATOR CIRCUIT



3.0 OSCILLATOR CONFIGURATIONS

3.1 Oscillator Types

The PIC18F2331/2431/4331/4431 devices can be operated in 10 different oscillator modes. The user can program the Configuration bits, FOSC<3:0>, in Configuration Register 1H to select one of these 10 modes:

1. LP — Low-Power Crystal
2. XT — Crystal/Resonator
3. HS — High-Speed Crystal/Resonator
4. HSPLL — High-Speed Crystal/Resonator with PLL Enabled
5. RC — External Resistor/Capacitor with FOSC/4 Output on RA6
6. RCIO — External Resistor/Capacitor with I/O on RA6
7. INTIO1 — Internal Oscillator with FOSC/4 Output on RA6 and I/O on RA7
8. INTIO2 — Internal Oscillator with I/O on RA6 and RA7
9. EC — External Clock with FOSC/4 Output
10. ECIO — External Clock with I/O on RA6

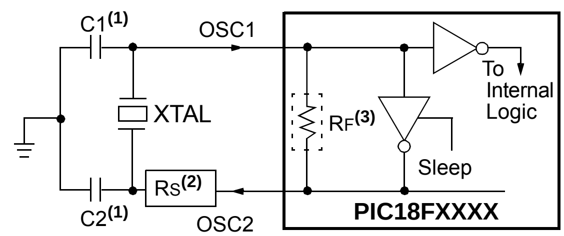
3.2 Crystal Oscillator/Ceramic Resonators

In XT, LP, HS or HSPLL Oscillator modes, a crystal or ceramic resonator is connected to the OSC1 and OSC2 pins to establish oscillation. Figure 3-1 shows the pin connections.

The oscillator design requires the use of a parallel resonant crystal.

Note: Use of a series resonant crystal may give a frequency out of the crystal manufacturers' specifications.

FIGURE 3-1: CRYSTAL/CERAMIC RESONATOR OPERATION (XT, LP, HS OR HSPLL CONFIGURATION)



Note 1: See Table 3-1 and Table 3-2 for initial values of C1 and C2.

2: A series resistor (RS) may be required for AT strip resonant crystals.

3: RF varies with the oscillator mode chosen.

TABLE 3-1: CAPACITOR SELECTION FOR CERAMIC RESONATORS

Typical Capacitor Values Used:			
Mode	**Freq**	**OSC1**	**OSC2**
XT	455 kHz 2.0 MHz 4.0 MHz	56 pF 47 pF 33 pF	56 pF 47 pF 33 pF
HS	8.0 MHz 16.0 MHz	27 pF 22 pF	27 pF 22 pF

Capacitor values are for design guidance only.

These capacitors were tested with the resonators listed below for basic start-up and operation. **These values are not optimized**.

Different capacitor values may be required to produce acceptable oscillator operation. The user should test the performance of the oscillator over the expected VDD and temperature range for the application.

See the notes following Table 3-2 for additional information.

Resonators Used:	
455 kHz	4.0 MHz
2.0 MHz	8.0 MHz
16.0 MHz	

TABLE 3-2: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR

Osc Type	Crystal Freq	Typical Capacitor Values Tested:	
		C1	C2
LP	32 kHz	33 pF	33 pF
	200 kHz	15 pF	15 pF
XT	1 MHz	33 pF	33 pF
	4 MHz	27 pF	27 pF
HS	4 MHz	27 pF	27 pF
	8 MHz	22 pF	22 pF
	20 MHz	15 pF	15 pF

Capacitor values are for design guidance only.

These capacitors were tested with the crystals listed below for basic start-up and operation. **These values are not optimized.**

Different capacitor values may be required to produce acceptable oscillator operation. The user should test the performance of the oscillator over the expected VDD and temperature range for the application.

See the notes following this table for additional information.

Crystals Used:	
32 kHz	4 MHz
200 kHz	8 MHz
1 MHz	20 MHz

Note 1: Higher capacitance increases the stability of oscillator, but also increases the start-up time.

2: When operating below 3V VDD, or when using certain ceramic resonators at any voltage, it may be necessary to use the HS mode or switch to a crystal oscillator.

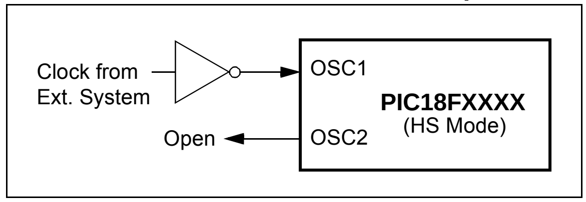
3: Since each resonator/crystal has its own characteristics, the user should consult the resonator/crystal manufacturer for appropriate values of external components.

4: RS may be required to avoid overdriving crystals with low drive level specification.

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

An external clock source may also be connected to the OSC1 pin in the HS mode, as shown in Figure 3-2.

FIGURE 3-2: EXTERNAL CLOCK INPUT OPERATION (HS OSC CONFIGURATION)



3.3 PLL Frequency Multiplier

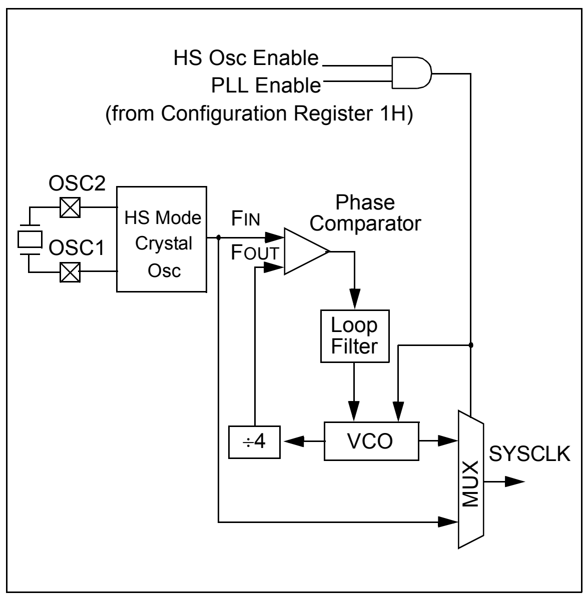
A Phase Locked Loop (PLL) circuit is provided as an option for users who wish to use a lower frequency oscillator circuit or to clock the device up to its highest rated frequency from a crystal oscillator. This may be useful for those concerned with EMI from high-frequency crystals or users requiring higher clock speeds from an internal oscillator.

3.3.1 HSPLL OSCILLATOR MODE

The HSPLL mode uses the HS Oscillator mode for frequencies up to 10 MHz. A PLL circuit then multiplies the oscillator output frequency by four to produce an internal clock frequency up to 40 MHz. The PLLEN bit is not available in this oscillator mode.

The PLL is only available to the crystal oscillator when the FOSC<3:0> Configuration bits are programmed for HSPLL mode (`0110`).

FIGURE 3-3: PLL BLOCK DIAGRAM

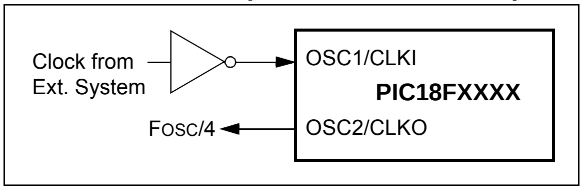


3.4 External Clock Input

The EC and ECIO Oscillator modes require an external clock source to be connected to the OSC1 pin. There is no oscillator start-up time required after a Power-on Reset or after an exit from Sleep mode.

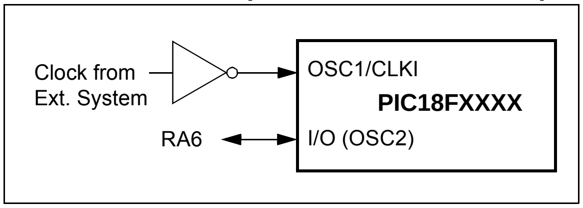
In the EC Oscillator mode, the oscillator frequency divided by 4 is available on the OSC2 pin. This signal may be used for test purposes or to synchronize other logic. Figure 3-4 shows the pin connections for the EC Oscillator mode.

FIGURE 3-4: EXTERNAL CLOCK INPUT OPERATION (EC CONFIGURATION)



The ECIO Oscillator mode functions like the EC mode, except that the OSC2 pin becomes an additional general purpose I/O pin. The I/O pin becomes bit 6 of PORTA (RA6). Figure 3-5 shows the pin connections for the ECIO Oscillator mode.

FIGURE 3-5: EXTERNAL CLOCK INPUT OPERATION (ECIO CONFIGURATION)



3.5 RC Oscillator

For timing-insensitive applications, the "RC" and "RCIO" device options offer additional cost savings. The actual oscillator frequency is a function of several factors:

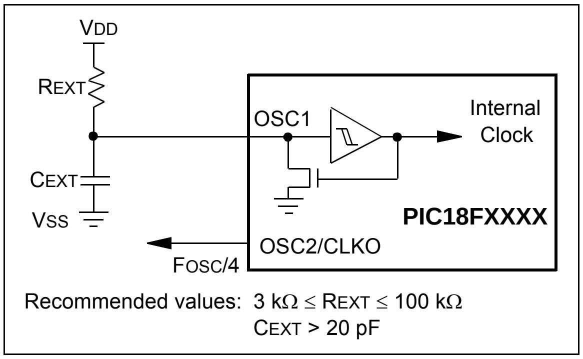
- Supply voltage
- Values of the external resistor (REXT) and capacitor (CEXT)
- Operating temperature

Given the same device, operating voltage and temperature, and component values, there will also be unit-to-unit frequency variations. These are due to factors, such as:

- Normal manufacturing variation
- Difference in lead frame capacitance between package types (especially for low CEXT values)
- Variations within the tolerance of limits of REXT and CEXT

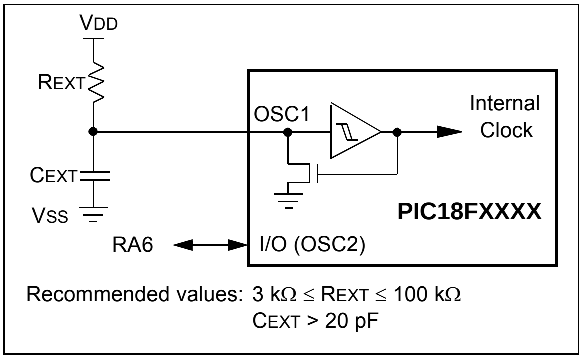
In the RC Oscillator mode (Figure 3-6), the oscillator frequency divided by 4 is available on the OSC2 pin. This signal may be used for test purposes or to synchronize other logic.

FIGURE 3-6: RC OSCILLATOR MODE



The RCIO Oscillator mode (Figure 3-7) functions like the RC mode, except that the OSC2 pin becomes an additional general purpose I/O pin. The I/O pin becomes bit 6 of PORTA (RA6).

FIGURE 3-7: RCIO OSCILLATOR MODE



3.6 Internal Oscillator Block

The PIC18F2331/2431/4331/4431 devices include an internal oscillator block, which generates two different clock signals; either can be used as the system's clock source. This can eliminate the need for external oscillator circuits on the OSC1 and/or OSC2 pins.

The main output (INTOSC) is an 8 MHz clock source, which can be used to directly drive the system clock. It also drives a postscaler, which can provide a range of clock frequencies from 125 kHz to 4 MHz. The INTOSC output is enabled when a system clock frequency from 125 kHz to 8 MHz is selected.

The other clock source is the internal RC oscillator (INTRC), which provides a 31 kHz output. The INTRC oscillator is enabled by selecting the internal oscillator block as the system clock source, or when any of the following are enabled:

- Power-up Timer
- Fail-Safe Clock Monitor
- Watchdog Timer
- Two-Speed Start-up

These features are discussed in greater detail in **Section 23.0 "Special Features of the CPU"**.

The clock source frequency (INTOSC direct, INTRC direct or INTOSC postscaler) is selected by configuring the IRCF bits of the OSCCON register (Register 3-2).

3.6.1 INTIO MODES

Using the internal oscillator as the clock source can eliminate the need for up to two external oscillator pins, which can then be used for digital I/O. Two distinct configurations are available:

- In INTIO1 mode, the OSC2 pin outputs FOSC/4, while OSC1 functions as RA7 for digital input and output.
- In INTIO2 mode, OSC1 functions as RA7 and OSC2 functions as RA6, both for digital input and output.

3.6.2 INTRC OUTPUT FREQUENCY

The internal oscillator block is calibrated at the factory to produce an INTOSC output frequency of 8.0 MHz. This changes the frequency of the INTRC source from its nominal 31.25 kHz. Peripherals and features that depend on the INTRC source will be affected by this shift in frequency.

3.6.3 OSCTUNE REGISTER

The internal oscillator's output has been calibrated at the factory, but can be adjusted in the user's application. This is done by writing to the OSCTUNE register (Register 3-1). Each increment may adjust the FRC frequency by varying amounts and may not be monotonic. The next closest frequency may be multiple steps apart.

When the OSCTUNE register is modified, the INTOSC and INTRC frequencies will begin shifting to the new frequency. Code execution continues during this shift. There is no indication that the shift has occurred. Operation of features that depend on the INTRC clock source frequency, such as the WDT, Fail-Safe Clock Monitor and peripherals, will also be affected by the change in frequency.

3.6.4 INTOSC FREQUENCY DRIFT

The factory calibrates the internal oscillator block output (INTOSC) for 8 MHz. This frequency, however, may drift as the VDD or temperature changes, which can affect the controller operation in a variety of ways.

The INTOSC frequency can be adjusted by modifying the value in the OSCTUNE register. This has no effect on the INTRC clock source frequency.

Tuning the INTOSC source requires knowing when to make an adjustment, in which direction it should be made, and in some cases, how large a change is needed. Three compensation techniques are discussed in **Section 3.6.4.1 "Compensating with the EUSART"**, **Section 3.6.4.2 "Compensating with the Timers"** and **Section 3.6.4.3 "Compensating with the CCP Module in Capture Mode"**, but other techniques may be used.

REGISTER 3-1: OSCTUNE: OSCILLATOR TUNING REGISTER

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	TUN5	TUN4	TUN3	TUN2	TUN1	TUN0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

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

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

`011111` = Maximum frequency
• •
• •
`000001`
`000000` = Center frequency. Oscillator module is running at the calibrated frequency.
`111111`
• •
• •
`100000` = Minimum frequency

3.6.4.1 Compensating with the EUSART

An adjustment may be required when the EUSART begins generating framing errors or receives data with errors while in Asynchronous mode. Framing errors frequently indicate that the device clock frequency is too high. To adjust for this, decrement the value in the OSCTUNE register to reduce the clock frequency.

Conversely, errors in data may suggest that the clock speed is too low; to compensate, increment the OSCTUNE register to increase the clock frequency.

3.6.4.2 Compensating with the Timers

This technique compares the device clock speed to that of a reference clock. Two timers may be used: one timer clocked by the peripheral clock and the other by a fixed reference source, such as the Timer1 oscillator.

Both timers are cleared, but the timer clocked by the reference generates interrupts. When an interrupt occurs, the internally clocked timer is read and both timers are cleared. If the internally clocked timer value is greater than expected, the internal oscillator block is running too fast. To adjust for this, decrement the OSCTUNE register.

3.6.4.3 Compensating with the CCP Module in Capture Mode

A CCP module can use free-running Timer1 (or Timer3), clocked by the internal oscillator block and an external event with a known period (such as the AC power frequency). The time of the first event is captured in the CCPRxH:CCPRxL registers and recorded for later use. When the second event causes a capture, the time of the first event is subtracted from the time of the second event. Since the period of the external event is known, the time difference between events can be calculated.

If the measured time is much greater than the calculated time, the internal oscillator block is running too fast. To compensate for this, decrement the OSCTUNE register. If the measured time is much less than the calculated time, the internal oscillator block is running too slow and the OSCTUNE register should be incremented.

3.7 Clock Sources and Oscillator Switching

Like previous PIC18 devices, the PIC18F2331/2431/4331/4431 devices include a feature that allows the system clock source to be switched from the main oscillator to an alternate low-frequency clock source. PIC18F2331/2431/4331/4431 devices offer two alternate clock sources. When enabled, these give additional options for switching to the various power-managed operating modes.

Essentially, there are three clock sources for these devices:

- Primary oscillators
- Secondary oscillators
- Internal oscillator block

The **primary oscillators** include the External Crystal and Resonator modes, the External RC modes, the External Clock modes and the internal oscillator block. The particular mode is defined on POR by the contents of Configuration Register 1H. The details of these modes are covered earlier in this chapter.

The **secondary oscillators** are those external sources not connected to the OSC1 or OSC2 pins. These sources may continue to operate even after the controller is placed in a power-managed mode.

PIC18F2331/2431/4331/4431 devices offer only the Timer1 oscillator as a secondary oscillator. This oscillator, in all power-managed modes, is often the time base for functions such as a Real-Time Clock (RTC).

Most often, a 32.768 kHz watch crystal is connected between the RC0/T1OSO/T1CKI and RC1/T1OSI/CCP2/$\overline{\text{FLTA}}$ pins. Like the LP Oscillator mode circuit, loading capacitors are also connected from each pin to ground.

The Timer1 oscillator is discussed in greater detail in **Section 13.2 "Timer1 Oscillator"**.

In addition to being a primary clock source, the **internal oscillator block** is available as a power-managed mode clock source. The INTRC source is also used as the clock source for several special features, such as the WDT and Fail-Safe Clock Monitor.

The clock sources for the PIC18F2331/2431/4331/4431 devices are shown in Figure 3-8. See **Section 13.0 "Timer1 Module"** for further details of the Timer1 oscillator. See **Section 23.1 "Configuration Bits"** for Configuration register details.

3.7.1 OSCILLATOR CONTROL REGISTER

The OSCCON register (Register 3-2) controls several aspects of the system clock's operation, both in full-power operation and in power-managed modes.

The System Clock Select bits, SCS<1:0>, select the clock source that is used when the device is operating in power-managed modes. The available clock sources are the primary clock (defined in Configuration Register 1H), the secondary clock (Timer1 oscillator) and the internal oscillator block. The clock selection has no effect until a SLEEP instruction is executed and the device enters a power-managed mode of operation. The SCS bits are cleared on all forms of Reset.

The Internal Oscillator Select bits, IRCF<2:0>, select the frequency output of the internal oscillator block that is used to drive the system clock. The choices are the INTRC source, the INTOSC source (8 MHz) or one of the six frequencies derived from the INTOSC postscaler (125 kHz to 4 MHz). If the internal oscillator block is supplying the system clock, changing the states of these bits will have an immediate change on the internal oscillator's output. On device Resets, the default output frequency of the internal oscillator block is set at 32 kHz.

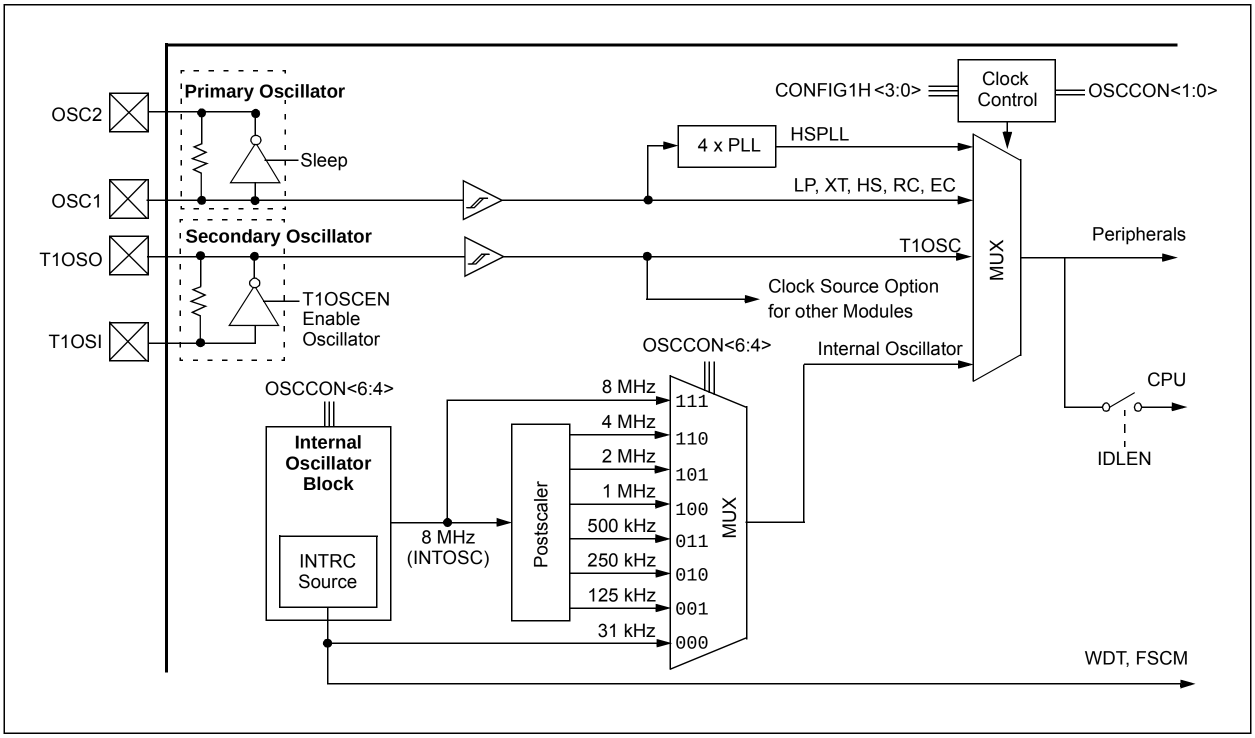
The OSTS, IOFS and T1RUN bits indicate which clock source is currently providing the system clock. The OSTS indicates that the Oscillator Start-up Timer has timed out, and the primary clock is providing the system clock in Primary Clock modes. The IOFS bit indicates when the internal oscillator block has stabilized, and is providing the system clock in RC Clock modes. The T1RUN bit (T1CON<6>) indicates when the Timer1 oscillator is providing the system clock in Secondary Clock modes. In power-managed modes, only one of these three bits will be set at any time. If none of these bits are set, the INTRC is providing the system clock, or the internal oscillator block has just started and is not yet stable.

The IDLEN bit controls the selective shutdown of the controller's CPU in power-managed modes. The use of these bits is discussed in more detail in **Section 4.0 "Power-Managed Modes"**

Note 1: The Timer1 oscillator must be enabled to select the secondary clock source. The Timer1 oscillator is enabled by setting the T1OSCEN bit in the Timer1 Control register (T1CON<3>). If the Timer1 oscillator is not enabled, then any attempt to select a secondary clock source, when executing a SLEEP instruction, will be ignored.

2: It is recommended that the Timer1 oscillator be operating and stable before executing the SLEEP instruction, or a very long delay may occur while the Timer1 oscillator starts.

FIGURE 3-8: PIC18F2331/2431/4331/4431 CLOCK DIAGRAM



REGISTER 3-2: OSCCON: OSCILLATOR CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R[(1)]	R-0	R/W-0	R/W-0
IDLEN	IRCF2	IRCF1	IRCF0	OSTS	IOFS	SCS1	SCS0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **IDLEN:** Idle Enable bit
- `1` = Idle mode enabled; CPU core is not clocked in power-managed modes
- `0` = Run mode enabled; CPU core is clocked in power-managed modes

bit 6-4 **IRCF<2:0>:** Internal Oscillator Frequency Select bits
- `111` = 8 MHz (8 MHz source drives clock directly)
- `110` = 4 MHz (default)
- `101` = 2 MHz
- `100` = 1 MHz
- `011` = 500 kHz
- `010` = 250 kHz
- `001` = 125 kHz
- `000` = 31 kHz (INTRC source drives clock directly)[2]

bit 3 **OSTS:** Oscillator Start-up Timer Time-out Status bit[1]
- `1` = Oscillator Start-up Timer time-out has expired; primary oscillator is running
- `0` = Oscillator Start-up Timer time-out is running; primary oscillator is not ready

bit 2 **IOFS:** INTOSC Frequency Stable bit
- `1` = INTOSC frequency is stable
- `0` = INTOSC frequency is not stable

bit 1-0 **SCS<1:0>:** System Clock Select bits
- `1x` = Internal oscillator block
- `01` = Secondary (Timer1) oscillator
- `00` = Primary oscillator

Note 1: Depends on the state of the IESO bit in Configuration Register 1H.
2: Default output frequency of INTOSC on Reset.

3.7.2 OSCILLATOR TRANSITIONS

The PIC18F2331/2431/4331/4431 devices contain circuitry to prevent clocking “glitches” when switching between clock sources. A short pause in the system clock occurs during the clock switch. The length of this pause is between 8 and 9 clock periods of the new clock source. This ensures that the new clock source is stable and that its pulse width will not be less than the shortest pulse width of the two clock sources.

Clock transitions are discussed in greater detail in **Section 4.1.2 “Entering Power-Managed Modes”**.

3.8 Effects of Power-Managed Modes on the Various Clock Sources

When PRI_IDLE mode is selected, the designated primary oscillator continues to run without interruption. For all other power-managed modes, the oscillator using the OSC1 pin is disabled. The OSC1 pin (and OSC2 pin, if used by the oscillator) will stop oscillating.

When the device executes a SLEEP instruction, the system is switched to one of the power-managed modes, depending on the state of the IDLEN and SCS<1:0> bits of the OSCCON register. See **Section 4.0 “Power-Managed Modes”** for details.

In secondary clock modes (SEC_RUN and SEC_IDLE), the Timer1 oscillator is operating and providing the system clock. The Timer1 oscillator may also run in all power-managed modes if required to clock Timer1.

In internal oscillator modes (RC_RUN and RC_IDLE), the internal oscillator block provides the system clock source. The INTRC output can be used directly to provide the system clock and may be enabled to support various special features, regardless of the power-managed mode (see **Section 23.2 “Watchdog Timer (WDT)”** through **Section 23.4 “Fail-Safe Clock Monitor”**). The INTOSC output at 8 MHz may be used directly to clock the system, or may be divided down first. The INTOSC output is disabled if the system clock is provided directly from the INTRC output.

If the Sleep mode is selected, all clock sources are stopped. Since all the transistor switching currents have been stopped, Sleep mode achieves the lowest current consumption of the device (only leakage currents).

Enabling any on-chip feature that will operate during Sleep will increase the current consumed during Sleep. The INTRC is required to support WDT operation. The Timer1 oscillator may be operating to support a Real-Time Clock. Other features may be operating that do not require a system clock source (i.e., SSP slave, INTx pins, A/D conversions and others).

3.9 Power-up Delays

Power-up delays are controlled by two timers, so that no external Reset circuitry is required for most applications. The delays ensure that the device is kept in Reset until the device power supply is stable under normal circumstances, and the primary clock is operating and stable. For additional information on power-up delays, see **Section 5.3 “Power-on Reset (POR)”** through **Section 5.4 “Brown-out Reset (BOR)”**.

The first timer is the Power-up Timer (PWRT), which provides a fixed delay on power-up (parameter 33, Table 26-8), if enabled, in Configuration Register 2L. The second timer is the Oscillator Start-up Timer (OST), intended to keep the chip in Reset until the crystal oscillator is stable (LP, XT and HS modes). The OST does this by counting 1024 oscillator cycles before allowing the oscillator to clock the device.

When the HSPLL Oscillator mode is selected, the device is kept in Reset for an additional 2 ms, following the HS mode OST delay, so the PLL can lock to the incoming clock frequency.

TABLE 3-3: OSC1 AND OSC2 PIN STATES IN SLEEP MODE

OSC Mode	OSC1 Pin	OSC2 Pin
RC, INTIO1	Floating, external resistor should pull high	At logic low (clock/4 output)
RCIO, INTIO2	Floating, external resistor should pull high	Configured as PORTA, bit 6
ECIO	Floating, pulled by external clock	Configured as PORTA, bit 6
EC	Floating, pulled by external clock	At logic low (clock/4 output)
LP, XT and HS	Feedback inverter disabled at quiescent voltage level	Feedback inverter disabled at quiescent voltage level

Note: See Table 5-1 in **Section 5.0 “Reset”** for time-outs due to Sleep and $\overline{\text{MCLR}}$ Reset.

NOTES:

4.0 POWER-MANAGED MODES

PIC18F2331/2431/4331/4431 devices offer a total of seven operating modes for more efficient power management. These modes provide a variety of options for selective power conservation in applications where resources may be limited (i.e., battery-powered devices).

There are three categories of power-managed modes:

- Run modes
- Idle modes
- Sleep mode

These categories define which portions of the device are clocked, and sometimes, what speed. The Run and Idle modes may use any of the three available clock sources (primary, secondary or internal oscillator block); the Sleep mode does not use a clock source.

The power-managed modes include several power-saving features offered on previous PIC® devices. One is the clock switching feature, offered in other PIC18 devices, allowing the controller to use the Timer1 oscillator in place of the primary oscillator. Also included is the Sleep mode, offered by all PIC devices, where all device clocks are stopped.

4.1 Selecting Power-Managed Modes

Selecting a power-managed mode requires two decisions: if the CPU is to be clocked or not and the selection of a clock source. The IDLEN bit (OSCCON<7>) controls CPU clocking, while the SCS<1:0> bits (OSCCON<1:0>) select the clock source. The individual modes, bit settings, clock sources and affected modules are summarized in Table 4-1.

4.1.1 CLOCK SOURCES

The SCS<1:0> bits allow the selection of one of three clock sources for power-managed modes. They are:

- the primary clock, as defined by the FOSC<3:0> Configuration bits
- the secondary clock (the Timer1 oscillator)
- the internal oscillator block (for RC modes)

4.1.2 ENTERING POWER-MANAGED MODES

Switching from one power-managed mode to another begins by loading the OSCCON register. The SCS<1:0> bits select the clock source and determine which Run or Idle mode is to be used. Changing these bits causes an immediate switch to the new clock source, assuming that it is running. The switch may also be subject to clock transition delays. These are discussed in **Section 4.1.3 "Clock Transitions and Status Indicators"** and subsequent sections.

Entry to the power-managed Idle or Sleep modes is triggered by the execution of a `SLEEP` instruction. The actual mode that results depends on the status of the IDLEN bit.

Depending on the current mode and the mode being switched to, a change to a power-managed mode does not always require setting all of these bits. Many transitions may be done by changing the oscillator select bits, or changing the IDLEN bit, prior to issuing a `SLEEP` instruction. If the IDLEN bit is already configured correctly, it may only be necessary to perform a `SLEEP` instruction to switch to the desired mode.

TABLE 4-1: POWER-MANAGED MODES

Mode	OSCCON Bits<7,1:0>		Module Clocking		Available Clock and Oscillator Source
	IDLEN[1]	SCS<1:0>	CPU	Peripherals	
Sleep	0	N/A	Off	Off	None – All clocks are disabled
PRI_RUN	N/A	00	Clocked	Clocked	Primary – LP, XT, HS, HSPLL, RC, EC and Internal Oscillator Block.[2] This is the normal, full-power execution mode.
SEC_RUN	N/A	01	Clocked	Clocked	Secondary – Timer1 Oscillator
RC_RUN	N/A	1x	Clocked	Clocked	Internal Oscillator Block[2]
PRI_IDLE	1	00	Off	Clocked	Primary – LP, XT, HS, HSPLL, RC, EC
SEC_IDLE	1	01	Off	Clocked	Secondary – Timer1 Oscillator
RC_IDLE	1	1x	Off	Clocked	Internal Oscillator Block[2]

Note 1: IDLEN reflects its value when the `SLEEP` instruction is executed.

2: Includes INTOSC and INTOSC postscaler, as well as the INTRC source.

4.1.3 CLOCK TRANSITIONS AND STATUS INDICATORS

The length of the transition between clock sources is the sum of two cycles of the old clock source and three to four cycles of the new clock source. This formula assumes that the new clock source is stable.

Three bits indicate the current clock source and its status. They are:

- OSTS (OSCCON<3>)
- IOFS (OSCCON<2>)
- T1RUN (T1CON<6>)

In general, only one of these bits will be set while in a given power-managed mode. When the OSTS bit is set, the primary clock is providing the device clock. When the IOFS bit is set, the INTOSC output is providing a stable, 8 MHz clock source to a divider that actually drives the device clock. When the T1RUN bit is set, the Timer1 oscillator is providing the clock. If none of these bits are set, then either the INTRC clock source is clocking the device, or the INTOSC source is not yet stable.

If the internal oscillator block is configured as the primary clock source by the FOSC<3:0> Configuration bits, then both the OSTS and IOFS bits may be set when in PRI_RUN or PRI_IDLE modes. This indicates that the primary clock (INTOSC output) is generating a stable, 8 MHz output. Entering another power-managed RC mode at the same frequency would clear the OSTS bit.

Note 1:	Caution should be used when modifying a single IRCF bit. If VDD is less than 3V, it is possible to select a higher clock speed than is supported by the low VDD. Improper device operation may result if the VDD/FOSC specifications are violated.
2:	Executing a SLEEP instruction does not necessarily place the device into Sleep mode. It acts as the trigger to place the controller into either the Sleep mode or one of the Idle modes, depending on the setting of the IDLEN bit.

4.1.4 MULTIPLE SLEEP COMMANDS

The power-managed mode that is invoked with the SLEEP instruction is determined by the setting of the IDLEN bit at the time the instruction is executed. If another SLEEP instruction is executed, the device will enter the power-managed mode specified by IDLEN at that time. If IDLEN has changed, the device will enter the new power-managed mode specified by the new setting.

4.2 Run Modes

In the Run modes, clocks to both the core and peripherals are active. The difference between these modes is the clock source.

4.2.1 PRI_RUN MODE

The PRI_RUN mode is the normal, full-power execution mode of the microcontroller. This is also the default mode upon a device Reset unless Two-Speed Start-up is enabled (see **Section 23.3 “Two-Speed Start-up”** for details). In this mode, the OSTS bit is set. The IOFS bit may be set if the internal oscillator block is the primary clock source (see **Section 3.7.1 “Oscillator Control Register”**).

4.2.2 SEC_RUN MODE

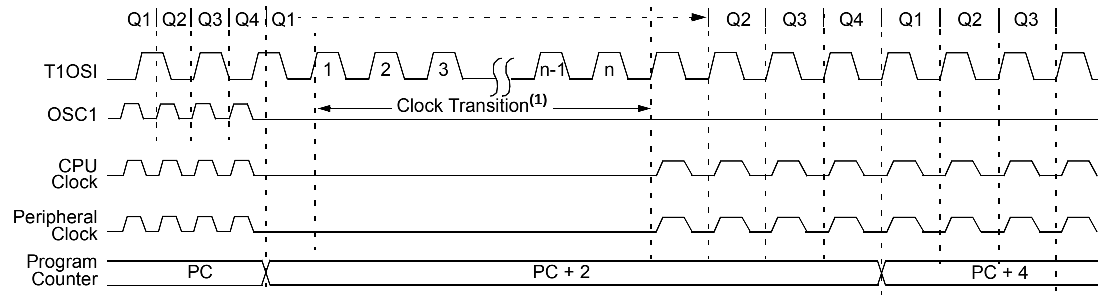
The SEC_RUN mode is the compatible mode to the “clock switching” feature offered in other PIC18 devices. In this mode, the CPU and peripherals are clocked from the Timer1 oscillator. This gives users the option of lower power consumption while still using a high-accuracy clock source.

SEC_RUN mode is entered by setting the SCS<1:0> bits to ‘01’. The device clock source is switched to the Timer1 oscillator (see Figure 4-1), the primary oscillator is shut down, the T1RUN bit (T1CON<6>) is set and the OSTS bit is cleared.

Note:	The Timer1 oscillator should already be running prior to entering SEC_RUN mode. If the T1OSCEN bit is not set when the SCS<1:0> bits are set to ‘01’, entry to SEC_RUN mode will not occur. If the Timer1 oscillator is enabled, but not yet running, device clocks will be delayed until the oscillator has started. In such situations, initial oscillator operation is far from stable and unpredictable operation may result.

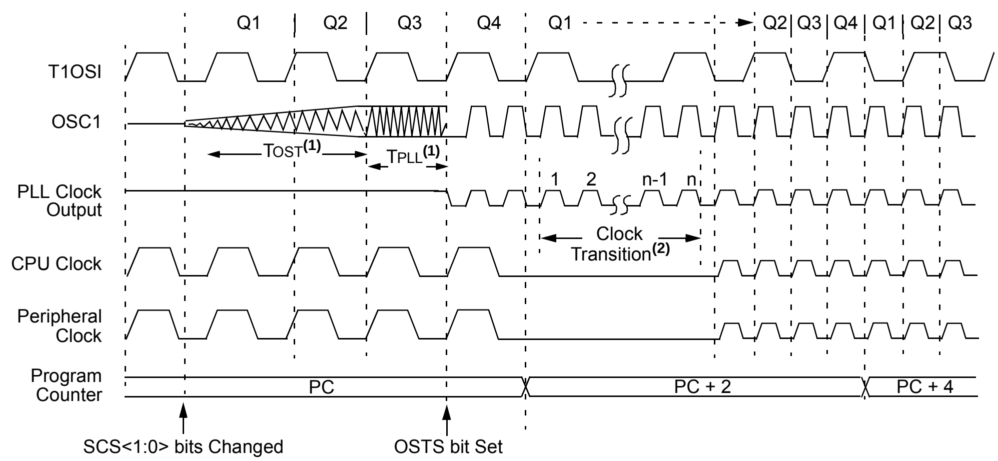
On transitions from SEC_RUN mode to PRI_RUN, the peripherals and CPU continue to be clocked from the Timer1 oscillator while the primary clock is started. When the primary clock becomes ready, a clock switch back to the primary clock occurs (see Figure 4-2). When the clock switch is complete, the T1RUN bit is cleared, the OSTS bit is set and the primary clock is providing the clock. The IDLEN and SCS bits are not affected by the wake-up; the Timer1 oscillator continues to run.

FIGURE 4-1: TRANSITION TIMING FOR ENTRY TO SEC_RUN MODE



Note 1: Clock transition typically occurs within 2-4 TOSC.

FIGURE 4-2: TRANSITION TIMING FROM SEC_RUN MODE TO PRI_RUN MODE (HSPLL)



Note 1: TOST = 1024 TOSC; TPLL = 2 ms (approx). These intervals are not shown to scale.
2: Clock transition typically occurs within 2-4 TOSC.

4.2.3 RC_RUN MODE

In RC_RUN mode, the CPU and peripherals are clocked from the internal oscillator block using the INTOSC multiplexer. In this mode, the primary clock is shut down. When using the INTRC source, this mode provides the best power conservation of all the Run modes, while still executing code. It works well for user applications which are not highly timing-sensitive or do not require high-speed clocks at all times.

If the primary clock source is the internal oscillator block (either INTRC or INTOSC), there are no distinguishable differences between PRI_RUN and RC_RUN modes during execution. However, a clock switch delay will occur during entry to and exit from RC_RUN mode. Therefore, if the primary clock source is the internal oscillator block, the use of RC_RUN mode is not recommended.

This mode is entered by setting the SCS1 bit to '1'. Although it is ignored, it is recommended that the SCS0 bit also be cleared; this is to maintain software compatibility with future devices. When the clock source is switched to the INTOSC multiplexer (see Figure 4-3), the primary oscillator is shut down and the OSTS bit is cleared. The IRCF bits may be modified at any time to immediately change the clock speed.

Note:	Caution should be used when modifying a single IRCF bit. If VDD is less than 3V, it is possible to select a higher clock speed than is supported by the low VDD. Improper device operation may result if the VDD/FOSC specifications are violated.

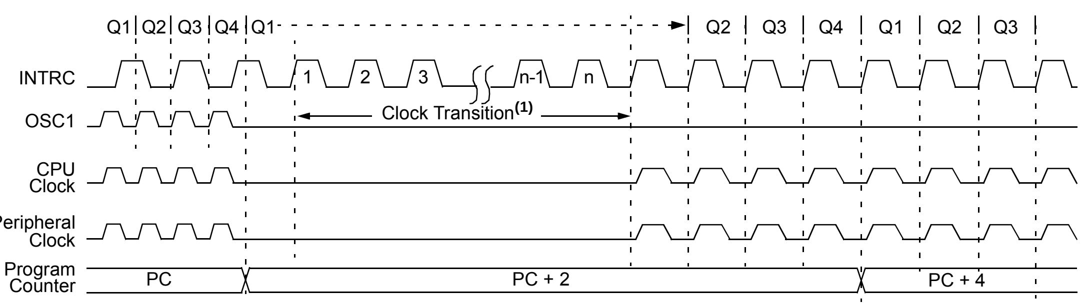
If the IRCF bits and the INTSRC bit are all clear, the INTOSC output is not enabled and the IOFS bit will remain clear; there will be no indication of the current clock source. The INTRC source is providing the device clocks.

If the IRCF bits are changed from all clear (thus, enabling the INTOSC output), or if INTSRC is set, the IOFS bit becomes set after the INTOSC output becomes stable. Clocks to the device continue while the INTOSC source stabilizes, after an interval of T_{IOBST}.

If the IRCF bits were previously at a non-zero value, or if INTSRC was set before setting SCS1 and the INTOSC source was already stable, the IOFS bit will remain set.

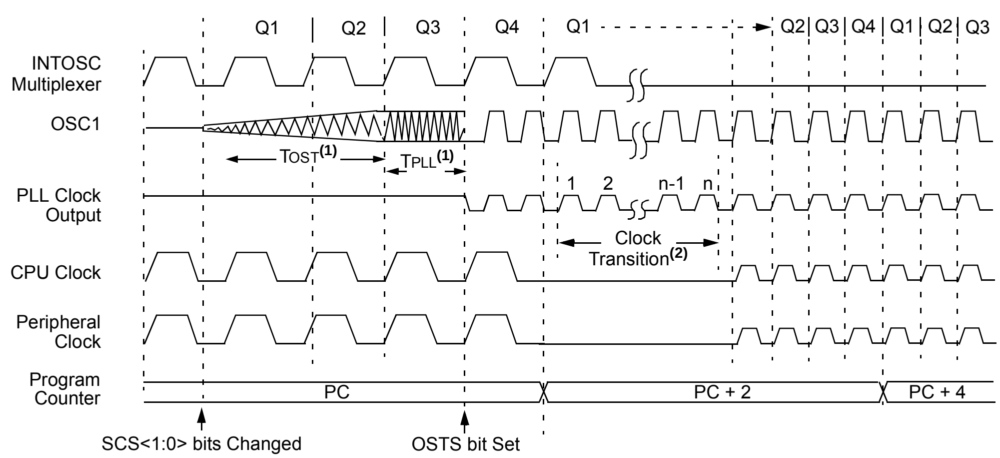
On transitions from RC_RUN mode to PRI_RUN mode, the device continues to be clocked from the INTOSC multiplexer while the primary clock is started. When the primary clock becomes ready, a clock switch to the primary clock occurs (see Figure 4-4). When the clock switch is complete, the IOFS bit is cleared, the OSTS bit is set and the primary clock is providing the device clock. The IDLEN and SCS bits are not affected by the switch. The INTRC source will continue to run if either the WDT or the Fail-Safe Clock Monitor is enabled.

FIGURE 4-3: TRANSITION TIMING TO RC_RUN MODE



Note 1: Clock transition typically occurs within 2-4 T_{OSC}.

FIGURE 4-4: TRANSITION TIMING FROM RC_RUN MODE TO PRI_RUN MODE



Note 1: T_{OST} = 1024 T_{OSC}; T_{PLL} = 2 ms (approx). These intervals are not shown to scale.
2: Clock transition typically occurs within 2-4 T_{OSC}.

4.3 Sleep Mode

The power-managed Sleep mode in the PIC18F2331/2431/4331/4431 devices is identical to the legacy Sleep mode offered in all other PIC devices. It is entered by clearing the IDLEN bit (the default state on device Reset) and executing the SLEEP instruction. This shuts down the selected oscillator (Figure 4-5). All clock source status bits are cleared.

Entering the Sleep mode from any other mode does not require a clock switch. This is because no clocks are needed once the controller has entered Sleep. If the WDT is selected, the INTRC source will continue to operate. If the Timer1 oscillator is enabled, it will also continue to run.

When a wake event occurs in Sleep mode (by interrupt, Reset or WDT time-out), the device will not be clocked until the clock source, selected by the SCS<1:0> bits, becomes ready (see Figure 4-6), or it will be clocked from the internal oscillator block if either the Two-Speed Start-up or the Fail-Safe Clock Monitor is enabled (see **Section 23.0 "Special Features of the CPU"**). In either case, the OSTS bit is set when the primary clock is providing the device clocks. The IDLEN and SCS bits are not affected by the wake-up.

4.4 Idle Modes

The Idle modes allow the controller's CPU to be selectively shut down while the peripherals continue to operate. Selecting a particular Idle mode allows users to further manage power consumption.

If the IDLEN bit is set to a '1' when a SLEEP instruction is executed, the peripherals will be clocked from the clock source selected using the SCS<1:0> bits; however, the CPU will not be clocked. The clock source status bits are not affected. Setting IDLEN and executing a SLEEP instruction provides a quick method of switching from a given Run mode to its corresponding Idle mode.

If the WDT is selected, the INTRC source will continue to operate. If the Timer1 oscillator is enabled, it will also continue to run.

Since the CPU is not executing instructions, the only exits from any of the Idle modes are by interrupt, WDT time-out or a Reset. When a wake event occurs, CPU execution is delayed by an interval of T_{CSD} (Parameter 38, Table 26-8) while it becomes ready to execute code. When the CPU begins executing code, it resumes with the same clock source for the current Idle mode. For example, when waking from RC_IDLE mode, the internal oscillator block will clock the CPU and peripherals (in other words, RC_RUN mode). The IDLEN and SCS bits are not affected by the wake-up.

While in any Idle mode or Sleep mode, a WDT time-out will result in a WDT wake-up to the Run mode currently specified by the SCS<1:0> bits.

FIGURE 4-5: TRANSITION TIMING FOR ENTRY TO SLEEP MODE

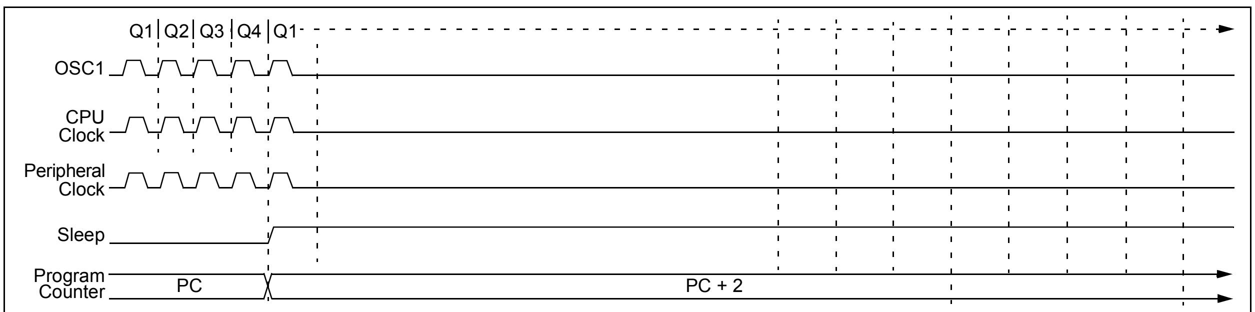
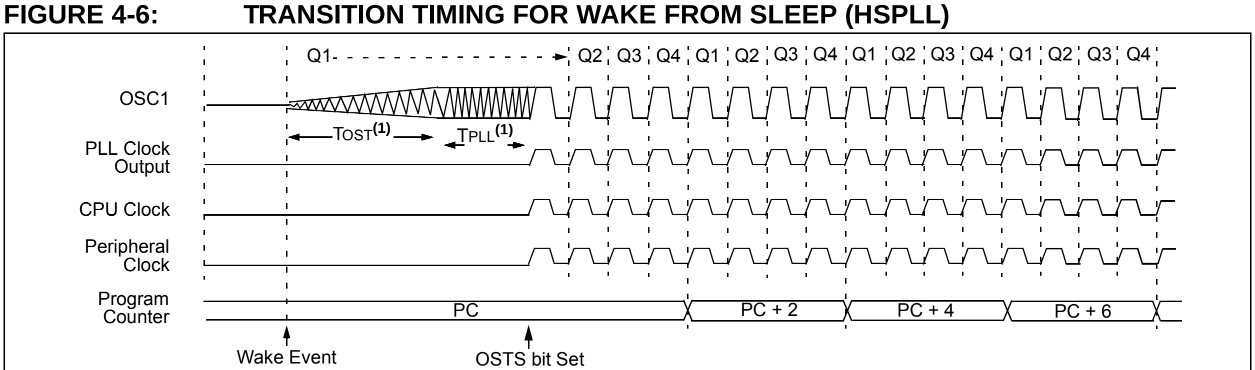


FIGURE 4-6: TRANSITION TIMING FOR WAKE FROM SLEEP (HSPLL)



Note 1: T_{OST} = 1024 T_{OSC}; T_{PLL} = 2 ms (approx). These intervals are not shown to scale.

4.4.1 PRI_IDLE MODE

This mode is unique among the three low-power Idle modes, in that it does not disable the primary device clock. For timing-sensitive applications, this allows for the fastest resumption of device operation with its more accurate primary clock source, since the clock source does not have to “warm-up” or transition from another oscillator.

PRI_IDLE mode is entered from PRI_RUN mode by setting the IDLEN bit and executing a SLEEP instruction. If the device is in another Run mode, set IDLEN first, then clear the SCS bits and execute SLEEP. Although the CPU is disabled, the peripherals continue to be clocked from the primary clock source specified by the FOSC<3:0> Configuration bits. The OSTS bit remains set (see Figure 4-7).

When a wake event occurs, the CPU is clocked from the primary clock source. A delay of interval, TCSD, is required between the wake event and when code execution starts. This is required to allow the CPU to become ready to execute instructions. After the wake-up, the OSTS bit remains set. The IDLEN and SCS bits are not affected by the wake-up (see Figure 4-8).

4.4.2 SEC_IDLE MODE

In SEC_IDLE mode, the CPU is disabled but the peripherals continue to be clocked from the Timer1 oscillator. This mode is entered from SEC_RUN by setting the IDLEN bit and executing a SLEEP instruction. If the device is in another Run mode, set the IDLEN bit first, then set the SCS<1:0> bits to ‘01’ and execute SLEEP. When the clock source is switched to the Timer1 oscillator, the primary oscillator is shut down, the OSTS bit is cleared and the T1RUN bit is set.

When a wake event occurs, the peripherals continue to be clocked from the Timer1 oscillator. After an interval of TCSD, following the wake event, the CPU begins executing code being clocked by the Timer1 oscillator. The IDLEN and SCS bits are not affected by the wake-up; the Timer1 oscillator continues to run (see Figure 4-8).

Note:	The Timer1 oscillator should already be running prior to entering SEC_IDLE mode. If the T1OSCEN bit is not set when the SLEEP instruction is executed, the SLEEP instruction will be ignored and entry to SEC_IDLE mode will not occur. If the Timer1 oscillator is enabled but not yet running, peripheral clocks will be delayed until the oscillator has started. In such situations, initial oscillator operation is far from stable and unpredictable operation may result.

FIGURE 4-7: TRANSITION TIMING FOR ENTRY TO IDLE MODE

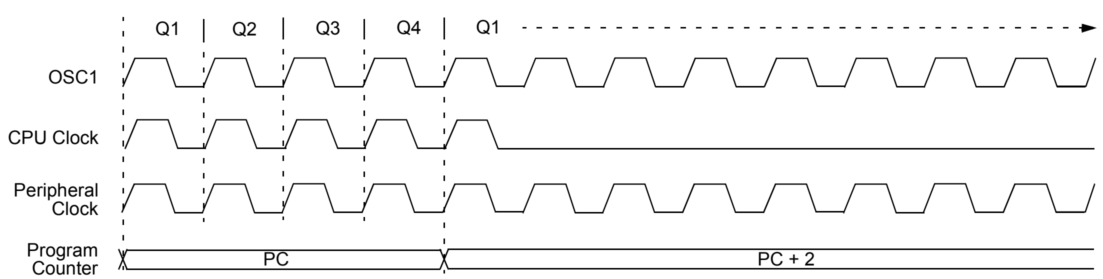
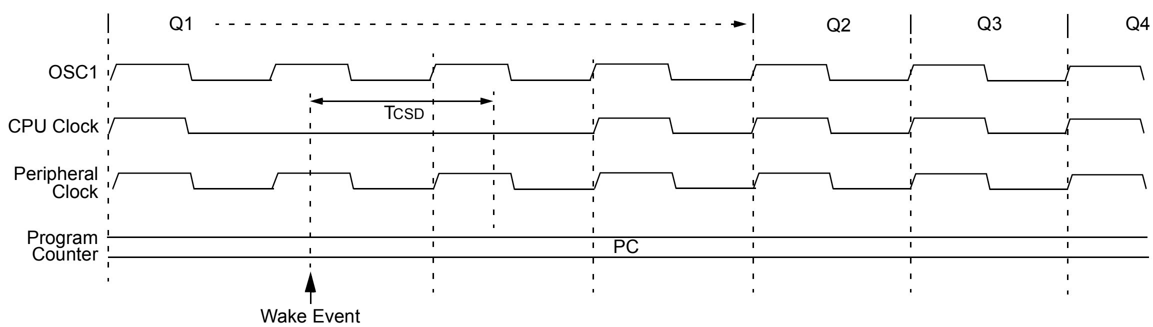


FIGURE 4-8: TRANSITION TIMING FOR WAKE FROM IDLE TO RUN MODE



4.4.3 RC_IDLE MODE

In RC_IDLE mode, the CPU is disabled but the peripherals continue to be clocked from the internal oscillator block using the INTOSC multiplexer. This mode allows for controllable power conservation during Idle periods.

From RC_RUN, this mode is entered by setting the IDLEN bit and executing a SLEEP instruction. If the device is in another Run mode, first set IDLEN, then set the SCS1 bit and execute SLEEP. Although its value is ignored, it is recommended that SCS0 also be cleared; this is to maintain software compatibility with future devices. The INTOSC multiplexer may be used to select a higher clock frequency by modifying the IRCF bits before executing the SLEEP instruction. When the clock source is switched to the INTOSC multiplexer, the primary oscillator is shut down and the OSTS bit is cleared.

If the IRCF bits are set to any non-zero value, or the INTSRC bit is set, the INTOSC output is enabled. The IOFS bit becomes set, after the INTOSC output becomes stable, after an interval of T_{IOBST} (Parameter 39, Table 26-8). Clocks to the peripherals continue while the INTOSC source stabilizes. If the IRCF bits were previously at a non-zero value, or INTSRC was set before the SLEEP instruction was executed, and the INTOSC source was already stable, the IOFS bit will remain set. If the IRCF bits and INTSRC are all clear, the INTOSC output will not be enabled, the IOFS bit will remain clear and there will be no indication of the current clock source.

When a wake event occurs, the peripherals continue to be clocked from the INTOSC multiplexer. After a delay of T_{CSD}, following the wake event, the CPU begins executing code being clocked by the INTOSC multiplexer. The IDLEN and SCS bits are not affected by the wake-up. The INTRC source will continue to run if either the WDT or the Fail-Safe Clock Monitor is enabled.

4.5 Exiting Idle and Sleep Modes

An exit from Sleep mode or any of the Idle modes, is triggered by an interrupt, a Reset or a WDT time-out. This section discusses the triggers that cause exits from power-managed modes. The clocking subsystem actions are discussed in more detail in each of the sections that relate to the power-managed modes (see **Section 4.2 "Run Modes"**, **Section 4.3 "Sleep Mode"** and **Section 4.4 "Idle Modes"**).

4.5.1 EXIT BY INTERRUPT

Any of the available interrupt sources can cause the device to exit from an Idle mode or Sleep mode to a Run mode. To enable this functionality, an interrupt source must be enabled by setting its enable bit in one of the INTCON or PIE registers. The exit sequence is initiated when the corresponding interrupt flag bit is set.

On all exits from Idle or Sleep modes by interrupt, code execution branches to the interrupt vector if the GIE/GIEH bit (INTCON<7>) is set. Otherwise, code execution continues or resumes without branching (see **Section 10.0 "Interrupts"**).

A fixed delay of interval, T_{CSD}, following the wake event, is required when leaving Sleep and Idle modes. This delay is required for the CPU to prepare for execution. Instruction execution resumes on the first clock cycle following this delay.

4.5.2 EXIT BY WDT TIME-OUT

A WDT time-out will cause different actions depending on which power-managed mode the device is in when the time-out occurs.

If the device is not executing code (all Idle modes and Sleep mode), the time-out will result in an exit from the power-managed mode (see **Section 4.2 "Run Modes"** and **Section 4.3 "Sleep Mode"**). If the device is executing code (all Run modes), the time-out will result in a WDT Reset (see **Section 23.2 "Watchdog Timer (WDT)"**).

The WDT timer and postscaler are cleared by executing a SLEEP or CLRWDT instruction, the loss of a currently selected clock source (if the Fail-Safe Clock Monitor is enabled) and modifying the IRCF bits in the OSCCON register if the internal oscillator block is the device clock source.

4.5.3 EXIT BY RESET

Normally, the device is held in Reset by the Oscillator Start-up Timer (OST) until the primary clock becomes ready. At that time, the OSTS bit is set and the device begins executing code. If the internal oscillator block is the new clock source, the IOFS bit is set instead.

The exit delay time from Reset to the start of code execution depends on both the clock sources before and after the wake-up, and the type of oscillator if the new clock source is the primary clock. Exit delays are summarized in Table 4-2.

Code execution can begin before the primary clock becomes ready. If either the Two-Speed Start-up (see **Section 23.3 "Two-Speed Start-up"**) or Fail-Safe Clock Monitor (see **Section 23.4 "Fail-Safe Clock Monitor"**) is enabled, the device may begin execution as soon as the Reset source has cleared. Execution is clocked by the INTOSC multiplexer driven by the internal oscillator block. Execution is clocked by the internal oscillator block until either the primary clock becomes ready or a power-managed mode is entered before the primary clock becomes ready; the primary clock is then shut down.

4.5.4 EXIT WITHOUT AN OSCILLATOR START-UP DELAY

Certain exits from power-managed modes do not invoke the OST at all. There are two cases:

- PRI_IDLE mode, where the primary clock source is not stopped; and
- the primary clock source is not any of the LP, XT, HS or HSPLL modes.

In these instances, the primary clock source either does not require an oscillator start-up delay since it is already running (PRI_IDLE), or normally does not require an oscillator start-up delay (RC, EC and INTIO Oscillator modes). However, a fixed delay of interval, TCSD, following the wake event, is still required when leaving Sleep and Idle modes to allow the CPU to prepare for execution. Instruction execution resumes on the first clock cycle following this delay.

TABLE 4-2: EXIT DELAY ON WAKE-UP BY RESET FROM SLEEP MODE OR ANY IDLE MODE (BY CLOCK SOURCES)

Clock Source Before Wake-up	Clock Source After Wake-up	Exit Delay	Clock Ready Status Bit (OSCCON)
Primary Device Clock (PRI_IDLE mode)	LP, XT, HS	TCSD(1)	OSTS
	HSPLL		
	EC, RC		
	INTOSC(2)		IOFS
T1OSC	LP, XT, HS	TOST(3)	OSTS
	HSPLL	TOST + t_{rc}(3)	
	EC, RC	TCSD(1)	
	INTOSC(2)	TIOBST(4)	IOFS
INTOSC(3)	LP, XT, HS	TOST(3)	OSTS
	HSPLL	TOST + t_{rc}(3)	
	EC, RC	TCSD(1)	
	INTOSC(2)	None	IOFS
None (Sleep mode)	LP, XT, HS	TOST(3)	OSTS
	HSPLL	TOST + t_{rc}(3)	
	EC, RC	TCSD(1)	
	INTOSC(2)	TIOBST(4)	IOFS

Note 1: TCSD (Parameter 38) is a required delay when waking from Sleep and all Idle modes, and runs concurrently with any other required delays (see **Section 4.4 "Idle Modes"**).

2: Includes both the INTOSC 8 MHz source and postscaler derived frequencies.

3: TOST is the Oscillator Start-up Timer (Parameter 32). t_{rc} is the PLL Lock-out Timer (Parameter F12); it is also designated as TPLL.

4: Execution continues during TIOBST (Parameter 39), the INTOSC stabilization period.

5.0 RESET

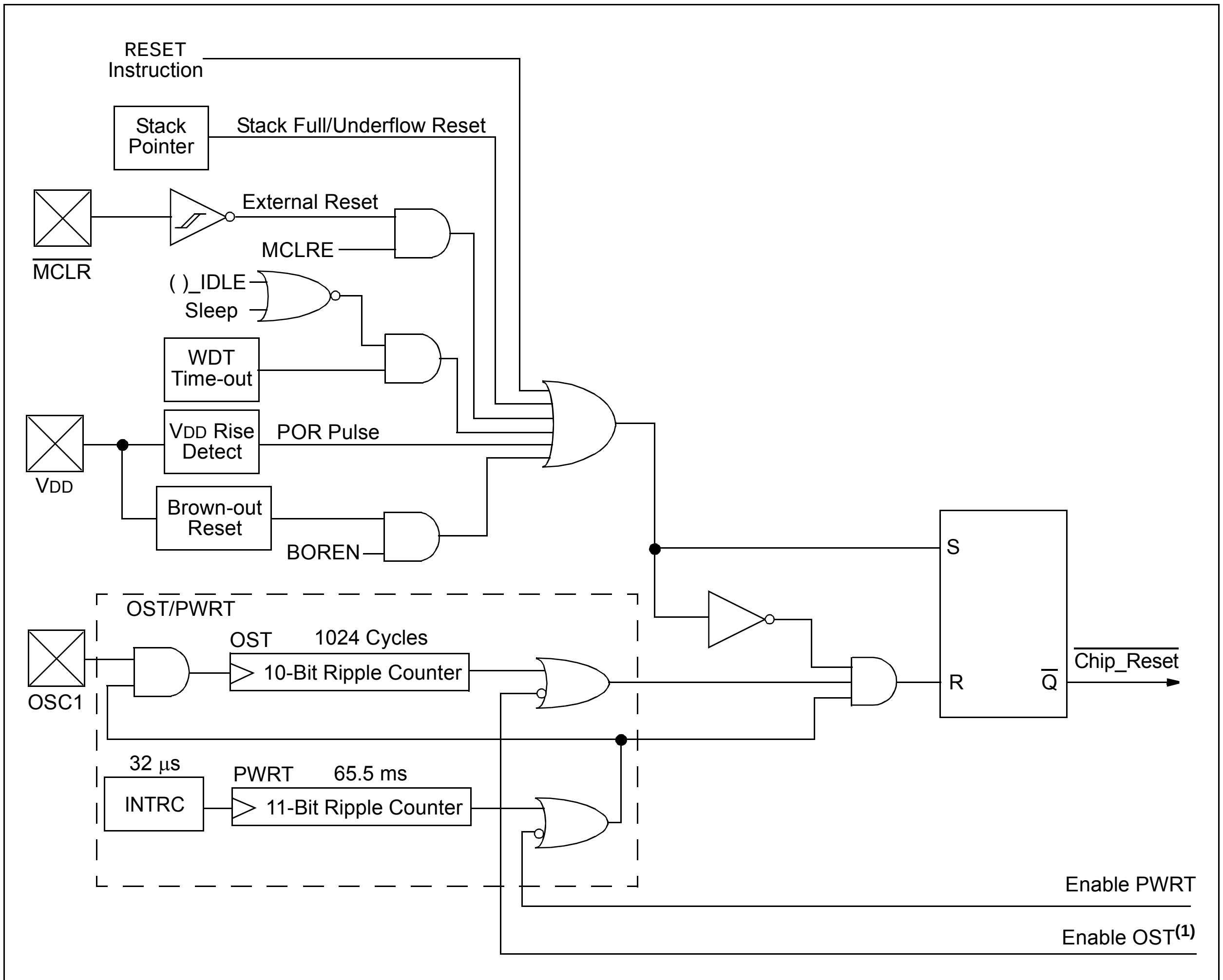
The PIC18F2331/2431/4331/4431 devices differentiate between various kinds of Reset:

a) Power-on Reset (POR)
b) MCLR Reset during normal operation
c) MCLR Reset during Sleep
d) Watchdog Timer (WDT) Reset (during execution)
e) Programmable Brown-out Reset (BOR)
f) RESET Instruction
g) Stack Full Reset
h) Stack Underflow Reset

This section discusses Resets generated by MCLR, POR and BOR, and the operation of the various start-up timers. Stack Reset events are covered in **Section 6.1.2.4 "Stack Full/Underflow Resets"**. WDT Resets are covered in **Section 23.2 "Watchdog Timer (WDT)"**.

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

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



Note 1: See Table 5-1 for time-out situations.

5.1 RCON Register

Device Reset events are tracked through the RCON register (Register 5-1). The lower five bits of the register indicate that a specific Reset event has occurred. In most cases, these bits can only be cleared by the event and must be set by the application after the event. The state of these flag bits, taken together, can be read to indicate the type of Reset that just occurred. This is described in more detail in **Section 5.6 "Reset State of Registers"**.

The RCON register also has control bits for setting interrupt priority (IPEN) and software control of the BOR (SBOREN). Interrupt priority is discussed in **Section 10.0 "Interrupts"**. BOR is covered in **Section 5.4 "Brown-out Reset (BOR)"**.

> **Note 1:** If the BOREN Configuration bit is set (Brown-out Reset enabled), the $\overline{\text{BOR}}$ bit is '1' on a Power-on Reset. After a Brown-out Reset has occurred, the BOR bit will be cleared and must be set by firmware to indicate the occurrence of the next Brown-out Reset.
>
> **2:** It is recommended that the $\overline{\text{POR}}$ bit be set after a Power-on Reset has been detected, so that subsequent Power-on Resets may be detected.

REGISTER 5-1: RCON: RESET CONTROL REGISTER

R/W-0	U-0	U-0	R/W-1	R-1	R-1	R/W-0	R/W-0
IPEN	—	—	$\overline{\text{RI}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	$\overline{\text{POR}}$(2)	$\overline{\text{BOR}}$(1)
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **IPEN:** Interrupt Priority Enable bit
1 = Enable priority levels on interrupts
0 = Disable priority levels on interrupts (PIC16CXXX Compatibility mode)

bit 6-5 **Unimplemented:** Read as '0'

bit 4 **$\overline{\text{RI}}$:** RESET Instruction Flag bit
1 = The RESET instruction was not executed (set by firmware only)
0 = The RESET instruction was executed causing a device Reset (must be set in software after a Brown-out Reset occurs)

bit 3 **$\overline{\text{TO}}$:** Watchdog Time-out Flag bit
1 = Set by power-up, CLRWDT instruction or SLEEP instruction
0 = A WDT time-out occurred

bit 2 **$\overline{\text{PD}}$:** Power-Down Detection Flag bit
1 = Set by power-up or by the CLRWDT instruction
0 = Set by execution of the SLEEP instruction

bit 1 **$\overline{\text{POR}}$:** Power-on Reset Status bit(2)
1 = A Power-on Reset has not occurred (set by firmware only)
0 = A Power-on Reset occurred (must be set in software after a Power-on Reset occurs)

bit 0 **$\overline{\text{BOR}}$:** Brown-out Reset Status bit(1)
1 = A Brown-out Reset has not occurred (set by firmware only)
0 = A Brown-out Reset occurred (must be set in software after a Brown-out Reset occurs)

Note 1: If SBOREN is enabled, its Reset state is '1'; otherwise, it is '0'.

2: The actual Reset value of $\overline{\text{POR}}$ is determined by the type of device Reset. See the notes following this register and **Section 5.6 "Reset State of Registers"** for additional information.

> **Note 1:** It is recommended that the $\overline{\text{POR}}$ bit be set after a Power-on Reset has been detected so that subsequent Power-on Resets may be detected.
>
> **2:** Brown-out Reset is said to have occurred when $\overline{\text{BOR}}$ is '0' and $\overline{\text{POR}}$ is '1' (assuming that POR was set to '1' by software immediately after a Power-on Reset).

5.2 Master Clear (MCLR)

The MCLR pin can trigger an external Reset of the device by holding the pin low. These devices have a noise filter in the MCLR Reset path that detects and ignores small pulses.

The MCLR pin is not driven low by any internal Resets, including the Watchdog Timer.

In PIC18F2331/2431/4331/4431 devices, the MCLR input can be disabled with the MCLRE Configuration bit. When MCLR is disabled, the pin becomes a digital input. For more information, see **Section 11.5 "PORTE, TRISE and LATE Registers"**.

5.3 Power-on Reset (POR)

A Power-on Reset pulse is generated on-chip whenever VDD rises above a certain threshold. This allows the device to start in the initialized state when VDD is adequate for operation.

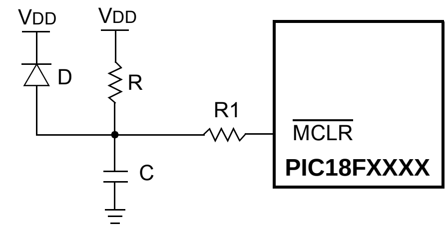
To take advantage of the POR circuitry, tie the MCLR pin through a resistor (1 kΩ to 10 kΩ) to VDD. This will eliminate external RC components usually needed to create a Power-on Reset delay. The minimum rise rate for VDD is specified (Parameter D004). For a slow rise time, see Figure 5-2.

When the device starts normal operation (i.e., exits the Reset condition), device operating parameters (such as voltage, frequency and temperature) must be met to ensure operation. If these conditions are not met, the device must be held in Reset until the operating conditions are met.

Power-on Reset events are captured by the POR bit (RCON<1>). The state of the bit is set to '0' whenever a POR occurs and does not change for any other Reset event. POR is not reset to '1' by any hardware event. To capture multiple events, the user manually resets the bit to '1' in software following any Power-on Reset.

Note: The following decoupling method is recommended:
1. A 1 μF capacitor should be connected across AVDD and AVSS.
2. A similar capacitor should be connected across VDD and VSS.

FIGURE 5-2: EXTERNAL POWER-ON RESET CIRCUIT (FOR SLOW VDD POWER-UP)



Note 1: External Power-on Reset circuit is required only if the VDD power-up slope is too slow. The diode, D, helps discharge the capacitor quickly when VDD powers down.

2: R < 40 kΩ is recommended to make sure that the voltage drop across R does not violate the device's electrical specification.

3: R1 ≥ 1 kΩ will limit any current flowing into MCLR from external capacitor, C, in the event of MCLR/VPP pin breakdown, due to Electrostatic Discharge (ESD) or Electrical Overstress (EOS).

5.4 Brown-out Reset (BOR)

A Configuration bit, BOREN, can disable (if clear/programmed) or enable (if set) the Brown-out Reset circuitry. If VDD falls below VBOR (Parameter D005A through D005F) for greater than TBOR (Parameter 35), the brown-out situation will reset the chip. A Reset may not occur if VDD falls below VBOR for less than TBOR. The chip will remain in Brown-out Reset until VDD rises above VBOR. If the Power-up Timer is enabled, it will be invoked after VDD rises above VBOR; it then will keep the chip in Reset for an additional time delay TPWRT (Parameter 33). If VDD drops below VBOR while the Power-up Timer is running, the chip will go back into a Brown-out Reset and the Power-up Timer will be initialized. Once VDD rises above VBOR, the Power-up Timer will execute the additional time delay. Enabling the Brown-out Reset does not automatically enable the PWRT.

5.5 Device Reset Timers

PIC18F2331/2431/4331/4431 devices incorporate three separate on-chip timers that help regulate the Power-on Reset process. Their main function is to ensure that the device clock is stable before code is executed. These timers are:

- Power-up Timer (PWRT)
- Oscillator Start-up Timer (OST)
- PLL Lock Time-out

5.5.1 POWER-UP TIMER (PWRT)

The Power-up Timer (PWRT) of PIC18F2331/2431/4331/4431 devices is an 11-bit counter that uses the INTRC source as the clock input. This yields an approximate time interval of 2,048 x 32 μs = 65.6 ms.

While the PWRT is counting, the device is held in Reset. The power-up time delay depends on the INTRC clock and will vary from chip to chip due to temperature and process variation. See DC Parameter 33 for details.

The PWRT is enabled by clearing the PWRTEN Configuration bit.

5.5.2 OSCILLATOR START-UP TIMER (OST)

The Oscillator Start-up Timer (OST) provides a 1,024 oscillator cycle (from OSC1 input) delay after the PWRT delay is over (Parameter 33). This ensures that the crystal oscillator or resonator has started and stabilized.

The OST time-out is invoked only for XT, LP, HS and HSPLL modes, and on Power-on Reset or on exit from most power-managed modes.

5.5.3 PLL LOCK TIME-OUT

With the PLL enabled in its PLL mode, the time-out sequence following a Power-on Reset is slightly different from other oscillator modes. A separate timer is used to provide a fixed time-out that is sufficient for the PLL to lock to the main oscillator frequency. This PLL Lock Time-out (T_{PLL}) is typically 2 ms and follows the oscillator start-up time-out.

5.5.4 TIME-OUT SEQUENCE

On power-up, the time-out sequence is as follows:

1. After the POR pulse has cleared, the PWRT time-out is invoked (if enabled).
2. Then, the OST is activated.

The total time-out will vary based on oscillator configuration and the status of the PWRT. Figure 5-3 through Figure 5-7 depict time-out sequences on power-up, with the Power-up Timer enabled and the device operating in HS Oscillator mode. Figure 5-3 through Figure 5-6 also apply to devices operating in XT or LP modes.

For devices in RC mode, and with the PWRT disabled, there will be no time-out at all. Since the time-outs occur from the POR pulse, if $\overline{MCLR}$ is kept low long enough, all time-outs will expire. Bringing $\overline{MCLR}$ high will begin execution immediately (Figure 5-5). This is useful for testing purposes or synchronization of more than one PIC18FXXXX device operating in parallel.

TABLE 5-1: TIME-OUT IN VARIOUS SITUATIONS

Oscillator Configuration	Power-up(2) and Brown-out		Exit From Power-Managed Mode
	$\overline{PWRTEN}$ = 0	$\overline{PWRTEN}$ = 1	
HSPLL	66 ms(1) + 1024 TOSC + 2 ms(2)	1024 TOSC + 2 ms(2)	1024 TOSC + 2 ms(2)
HS, XT, LP	66 ms(1) + 1024 TOSC	1024 TOSC	1024 TOSC
EC, ECIO	66 ms(1)	—	—
RC, RCIO	66 ms(1)	—	—
INTIO1, INTIO2	66 ms(1)	—	—

Note 1: 66 ms (65.5 ms) is the nominal Power-up Timer (PWRT) delay.
2: 2 ms is the nominal time required for the 4x PLL to lock.

5.6 Reset State of Registers

Most registers are unaffected by a Reset. Their status is unknown on POR and unchanged by all other Resets. The other registers are forced to a “Reset state” depending on the type of Reset that occurred.

Most registers are not affected by a WDT wake-up, since this is viewed as the resumption of normal operation.

Status bits from the RCON register ($\overline{RI}$, $\overline{TO}$, $\overline{PD}$, $\overline{POR}$ and $\overline{BOR}$) are set or cleared differently in different Reset situations, as indicated in Table 5-2. These bits are used in software to determine the nature of the Reset.

Table 5-3 describes the Reset states for all of the Special Function Registers. These are categorized by Power-on and Brown-out Resets, Master Clear and WDT Resets, and WDT wake-ups.

FIGURE 5-3: TIME-OUT SEQUENCE ON POWER-UP ($\overline{MCLR}$ TIED TO V_{DD}, V_{DD} RISE < T_{PWRT})

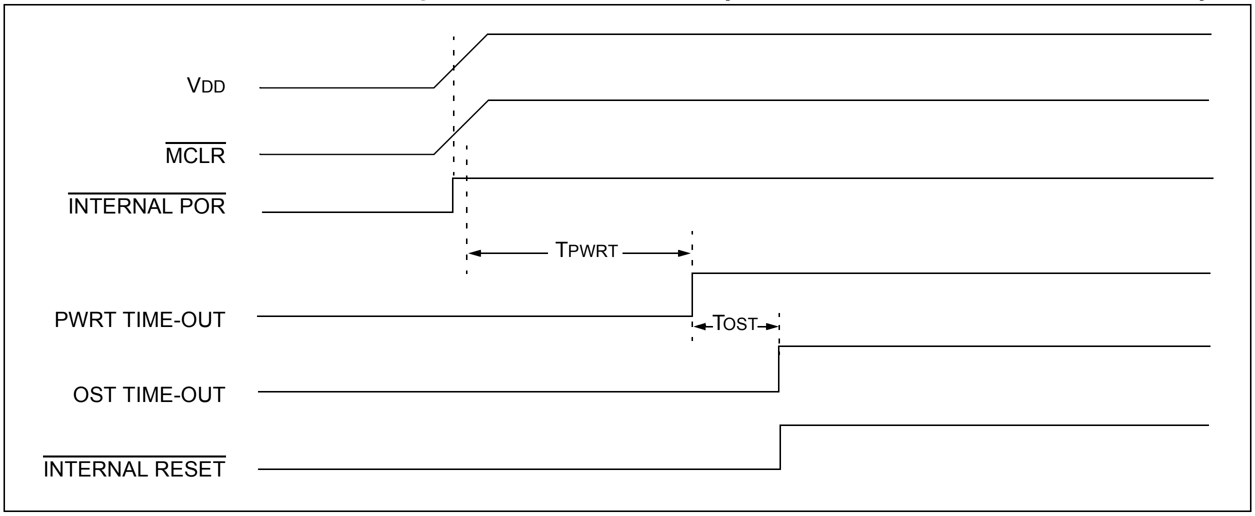


FIGURE 5-4: TIME-OUT SEQUENCE ON POWER-UP ($\overline{MCLR}$ NOT TIED TO V_{DD}): CASE 1

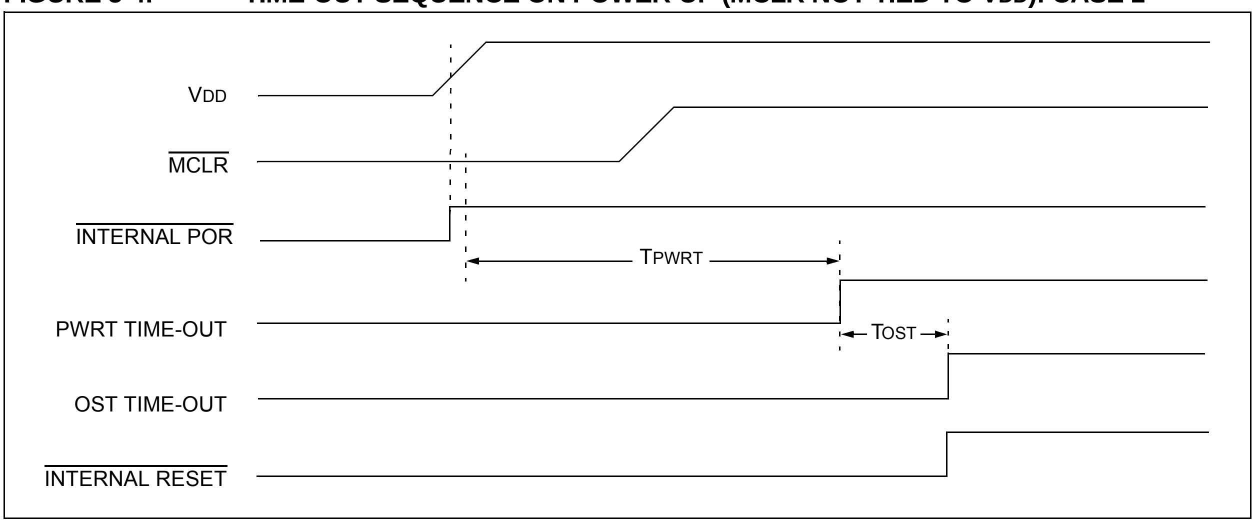


FIGURE 5-5: TIME-OUT SEQUENCE ON POWER-UP (MCLR NOT TIED TO VDD): CASE 2

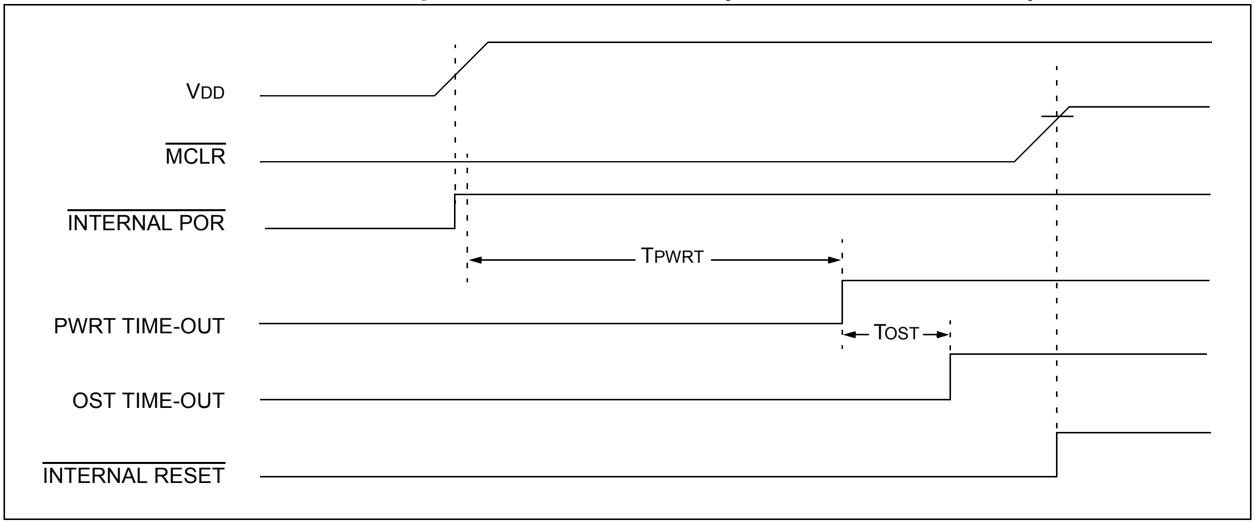


FIGURE 5-6: SLOW RISE TIME (MCLR TIED TO VDD, VDD RISE > TPWRT)

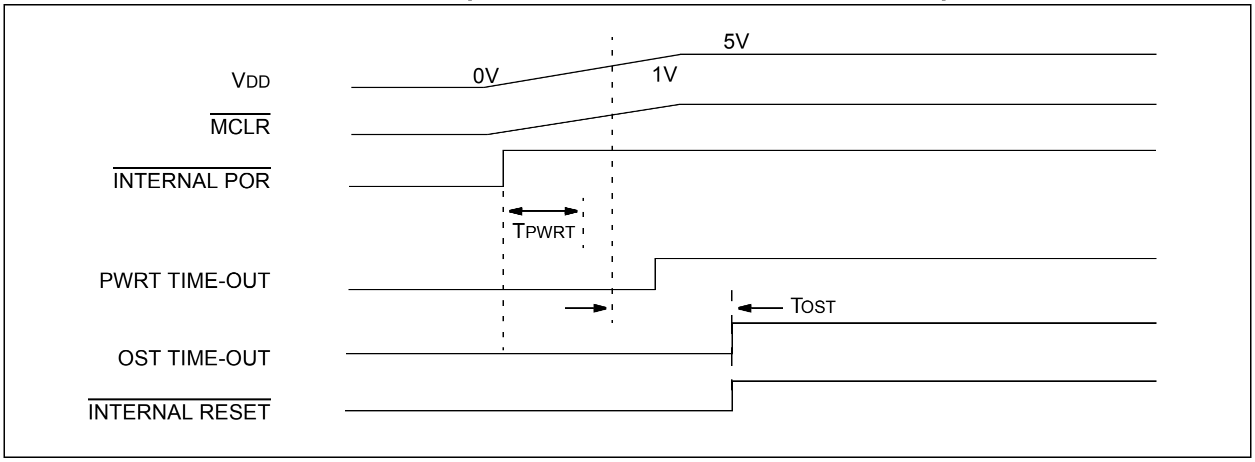
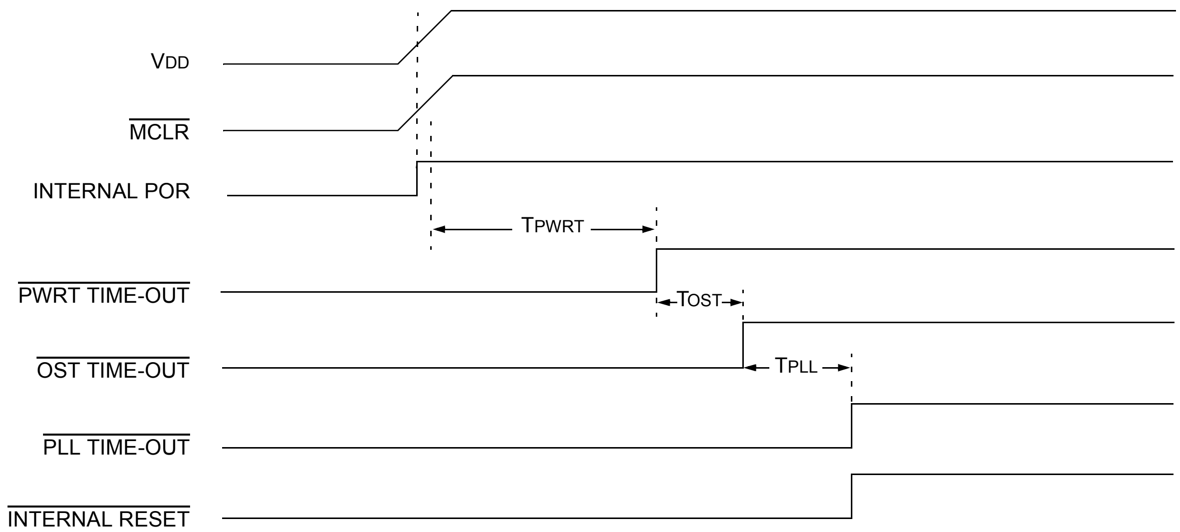


FIGURE 5-7: TIME-OUT SEQUENCE ON POR w/PLL ENABLED (MCLR TIED TO VDD)



Note: TOST = 1024 clock cycles.
TPLL ≈ 2 ms max. First three stages of the PWRT timer.

TABLE 5-2: STATUS BITS, THEIR SIGNIFICANCE AND THE INITIALIZATION CONDITION FOR RCON REGISTER

Condition	Program Counter	RCON Register	RI	TO	PD	POR	BOR	STKFUL	STKUNF
Power-on Reset	0000h	0--1 1100	1	1	1	0	0	0	0
RESET Instruction	0000h	0--0 uuuu	0	u	u	u	u	u	u
Brown-out	0000h	0--1 11u-	1	1	1	u	0	u	u
MCLR Reset during power-managed Run modes	0000h	0--u 1uuu	u	1	u	u	u	u	u
MCLR Reset during power-managed Idle and Sleep modes	0000h	0--u 10uu	u	1	0	u	u	u	u
WDT Time-out during full power or power-managed Run modes	0000h	0--u 0uuu	u	0	u	u	u	u	u
MCLR Reset during full-power execution								u	u
Stack Full Reset (STVREN = 1)	0000h	0--u uuuu	u	u	u	u	u	1	u
Stack Underflow Reset (STVREN = 1)								u	1
Stack Underflow Error (not an actual Reset, STVREN = 0)	0000h	u--u uuuu	u	u	u	u	u	u	1
WDT time-out during power-managed Idle or Sleep modes	PC + 2	u--u 00uu	u	0	0	u	u	u	u
Interrupt exit from power-managed modes	PC + 2[1]	u--u u0uu	u	u	0	u	u	u	u

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0'.
Note 1: When the wake-up is due to an interrupt and the GIEH or GIEL bits are set, the PC is loaded with the interrupt vector (0x000008h or 0x000018h).

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS

Register	Applicable Devices				Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets WDT Reset RESET Instruction Stack Resets	Wake-up via WDT or Interrupt
TOSU	2331	2431	4331	4431	---0 0000	---0 0000	---0 uuuu(3)
TOSH	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu(3)
TOSL	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu(3)
STKPTR	2331	2431	4331	4431	00-0 0000	uu-0 0000	uu-u uuuu(3)
PCLATU	2331	2431	4331	4431	---0 0000	---0 0000	---u uuuu
PCLATH	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
PCL	2331	2431	4331	4431	0000 0000	0000 0000	PC + 2(2)
TBLPTRU	2331	2431	4331	4431	--00 0000	--00 0000	--uu uuuu
TBLPTRH	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
TBLPTRL	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
TABLAT	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
PRODH	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
PRODL	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
INTCON	2331	2431	4331	4431	0000 000x	0000 000u	uuuu uuuu(1)
INTCON2	2331	2431	4331	4431	1111 -1-1	1111 -1-1	uuuu -u-u(1)
INTCON3	2331	2431	4331	4431	11-0 0-00	11-0 0-00	uu-u u-uu(1)
INDF0	2331	2431	4331	4431	N/A	N/A	N/A
POSTINC0	2331	2431	4331	4431	N/A	N/A	N/A
POSTDEC0	2331	2431	4331	4431	N/A	N/A	N/A
PREINC0	2331	2431	4331	4431	N/A	N/A	N/A
PLUSW0	2331	2431	4331	4431	N/A	N/A	N/A
FSR0H	2331	2431	4331	4431	---- xxxx	---- uuuu	---- uuuu
FSR0L	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
WREG	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF1	2331	2431	4331	4431	N/A	N/A	N/A
POSTINC1	2331	2431	4331	4431	N/A	N/A	N/A
POSTDEC1	2331	2431	4331	4431	N/A	N/A	N/A
PREINC1	2331	2431	4331	4431	N/A	N/A	N/A
PLUSW1	2331	2431	4331	4431	N/A	N/A	N/A

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition. Shaded cells indicate conditions do not apply for the designated device.

Note 1: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

4: See Table 5-2 for Reset value for specific condition.

5: Bits 6 and 7 of PORTA, LATA and TRISA are enabled depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.

6: Bit 3 of PORTE and LATE are enabled if $\overline{\text{MCLR}}$ functionality is disabled. When not enabled as the PORTE pin, they are disabled and read as '0'. The 28-pin devices do not have only RE3 implemented.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices				Power-on Reset, Brown-out Reset	MCLR Resets WDT Reset RESET Instruction Stack Resets	Wake-up via WDT or Interrupt
FSR1H	2331	2431	4331	4431	---- 0000	---- uuuu	---- uuuu
FSR1L	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
BSR	2331	2431	4331	4431	---- 0000	---- 0000	---- uuuu
INDF2	2331	2431	4331	4431	N/A	N/A	N/A
POSTINC2	2331	2431	4331	4431	N/A	N/A	N/A
POSTDEC2	2331	2431	4331	4431	N/A	N/A	N/A
PREINC2	2331	2431	4331	4431	N/A	N/A	N/A
PLUSW2	2331	2431	4331	4431	N/A	N/A	N/A
FSR2H	2331	2431	4331	4431	---- 0000	---- uuuu	---- uuuu
FSR2L	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
STATUS	2331	2431	4331	4431	---x xxxx	---u uuuu	---u uuuu
TMR0H	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
TMR0L	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
T0CON	2331	2431	4331	4431	1111 1111	1111 1111	uuuu uuuu
OSCCON	2331	2431	4331	4431	0000 q000	0000 q000	uuuu uuuu
LVDCON	2331	2431	4331	4431	--00 0101	--00 0101	--uu uuuu
WDTCON	2331	2431	4331	4431	0--- ---0	0--- ---0	u--- ---u
RCON(4)	2331	2431	4331	4431	0--1 11q0	0--q qquu	u--u qquu
TMR1H	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1L	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	2331	2431	4331	4431	0000 0000	u0uu uuuu	uuuu uuuu
TMR2	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
PR2	2331	2431	4331	4431	1111 1111	1111 1111	1111 1111
T2CON	2331	2431	4331	4431	-000 0000	-000 0000	-uuu uuuu
SSPBUF	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSPADD	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
SSPSTAT	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
SSPCON	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition. Shaded cells indicate conditions do not apply for the designated device.

Note 1: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

4: See Table 5-2 for Reset value for specific condition.

5: Bits 6 and 7 of PORTA, LATA and TRISA are enabled depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.

6: Bit 3 of PORTE and LATE are enabled if MCLR functionality is disabled. When not enabled as the PORTE pin, they are disabled and read as '0'. The 28-pin devices do not have only RE3 implemented.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices				Power-on Reset, Brown-out Reset	$\overline{MCLR}$ Resets WDT Reset RESET Instruction Stack Resets	Wake-up via WDT or Interrupt
ADRESH	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADRESL	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON0	2331	2431	4331	4431	--00 0000	--00 0000	--uu uuuu
ADCON1	2331	2431	4331	4431	00-0 0000	00-0 0000	uu-u uuuu
ADCON2	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
ADCON3	2331	2431	4331	4431	00-0 0000	00-0 0000	uu-u uuuu
ADCHS	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
CCPR1H	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1L	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP1CON	2331	2431	4331	4431	--00 0000	--00 0000	--uu uuuu
CCPR2H	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR2L	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP2CON	2331	2431	4331	4431	--00 0000	--00 0000	--uu uuuu
ANSEL1	2331	2431	4331	4431	---- ---1	---- ---1	---- ---u
ANSEL0	2331	2431	4331	4431	1111 1111	1111 1111	uuuu uuuu
T5CON	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
QEICON	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
SPBRGH	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
SPBRG	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
RCREG	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
TXREG	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
TXSTA	2331	2431	4331	4431	0000 -010	0000 -010	uuuu -uuu
RCSTA	2331	2431	4331	4431	0000 000x	0000 000x	uuuu uuuu
BAUDCON	2331	2431	4331	4431	-1-1 0-00	-1-1 0-00	-u-u u-uu
EEADR	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
EEDATA	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
EECON2	2331	2431	4331	4431	0000 0000	0000 0000	0000 0000
EECON1	2331	2431	4331	4431	xx-0 x000	uu-0 u000	uu-0 u000
IPR3	2331	2431	4331	4431	---1 1111	---1 1111	---u uuuu
PIE3	2331	2431	4331	4431	---0 0000	---0 0000	---u uuuu
PIR3	2331	2431	4331	4431	---0 0000	---0 0000	---u uuuu

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition. Shaded cells indicate conditions do not apply for the designated device.

Note 1: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

4: See Table 5-2 for Reset value for specific condition.

5: Bits 6 and 7 of PORTA, LATA and TRISA are enabled depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.

6: Bit 3 of PORTE and LATE are enabled if $\overline{MCLR}$ functionality is disabled. When not enabled as the PORTE pin, they are disabled and read as '0'. The 28-pin devices do not have only RE3 implemented.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices				Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets WDT Reset RESET Instruction Stack Resets	Wake-up via WDT or Interrupt
IPR2	2331	2431	4331	4431	1--1 -1-1	1--1 -1-1	u--u -u-u
PIR2	2331	2431	4331	4431	0--0 -0-0	0--0 -0-0	u--u -u-u
PIE2	2331	2431	4331	4431	0--0 -0-0	0--0 -0-0	u--u -u-u
IPR1	2331	2431	4331	4431	-111 1111	-111 1111	-uuu uuuu
PIR1	2331	2431	4331	4431	-000 0000	-000 0000	-uuu uuuu(1)
	2331	2431	4331	4431	-000 0000	-000 0000	-uuu uuuu(1)
PIE1	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
	2331	2431	4331	4431	-000 0000	-000 0000	-uuu uuuu
OSCTUNE	2331	2431	4331	4431	--00 0000	--00 0000	--uu uuuu
TRISE(6)	2331	2431	4331	4431	---- -111	---- -111	---- -uuu
TRISD	2331	2431	4331	4431	1111 1111	1111 1111	uuuu uuuu
TRISC	2331	2431	4331	4431	1111 1111	1111 1111	uuuu uuuu
TRISB	2331	2431	4331	4431	1111 1111	1111 1111	uuuu uuuu
TRISA(5)	2331	2431	4331	4431	1111 1111(5)	1111 1111(5)	uuuu uuuu(5)
PR5H	2331	2431	4331	4431	1111 1111	1111 1111	uuuu uuuu
PR5L	2331	2431	4331	4431	1111 1111	1111 1111	uuuu uuuu
LATE(6)	2331	2431	4331	4431	---- -xxx	---- -uuu	---- -uuu
LATD	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATC	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATB	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATA(5)	2331	2431	4331	4431	xxxx xxxx(5)	uuuu uuuu(5)	uuuu uuuu(5)
TMR5H	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR5L	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTE(6)	2331	2431	4331	4431	---- xxxx	---- xxxx	---- uuuu
PORTD	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTC	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTB	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA(5)	2331	2431	4331	4431	xx0x 0000(5)	uu0u 0000(5)	uuuu uuuu(5)

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition. Shaded cells indicate conditions do not apply for the designated device.

Note 1: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

4: See Table 5-2 for Reset value for specific condition.

5: Bits 6 and 7 of PORTA, LATA and TRISA are enabled depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.

6: Bit 3 of PORTE and LATE are enabled if $\overline{\text{MCLR}}$ functionality is disabled. When not enabled as the PORTE pin, they are disabled and read as '0'. The 28-pin devices do not have only RE3 implemented.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices				Power-on Reset, Brown-out Reset	$\overline{MCLR}$ Resets WDT Reset RESET Instruction Stack Resets	Wake-up via WDT or Interrupt
PTCON0	2331	2431	4331	4431	0000 0000	uuuu uuuu	uuuu uuuu
PTCON1	2331	2431	4331	4431	00-- ----	00-- ----	uu-- ----
PTMRL	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
PTMRH	2331	2431	4331	4431	---- 0000	---- 0000	---- uuuu
PTPERL	2331	2431	4331	4431	1111 1111	1111 1111	uuuu uuuu
PTPERH	2331	2431	4331	4431	---- 1111	---- 1111	---- uuuu
PDC0L	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
PDC0H	2331	2431	4331	4431	--00 0000	--00 0000	--uu uuuu
PDC1L	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
PDC1H	2331	2431	4331	4431	--00 0000	--00 0000	--uu uuuu
PDC2L	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
PDC2H	2331	2431	4331	4431	--00 0000	--00 0000	--uu uuuu
PDC3L	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
PDC3H	2331	2431	4331	4431	--00 0000	--00 0000	--uu uuuu
SEVTCMPL	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
SEVTCMPH	2331	2431	4331	4431	---- 0000	---- 0000	---- uuuu
PWMCON0	2331	2431	4331	4431	-111 0000	-111 0000	-uuu uuuu
PWMCON1	2331	2431	4331	4431	0000 0-00	0000 0-00	uuuu u-uu
DTCON	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
FLTCONFIG	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
OVDCOND	2331	2431	4331	4431	1111 1111	1111 1111	uuuu uuuu
OVDCONS	2331	2431	4331	4431	0000 0000	0000 0000	uuuu uuuu
CAP1BUFH/ VELRH	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
CAP1BUFL/ VELRL	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
CAP2BUFH/ POSCNTH	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
CAP2BUFL/ POSCNTL	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
CAP3BUFH/ MAXCNTH	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition. Shaded cells indicate conditions do not apply for the designated device.

Note 1: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

4: See Table 5-2 for Reset value for specific condition.

5: Bits 6 and 7 of PORTA, LATA and TRISA are enabled depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.

6: Bit 3 of PORTE and LATE are enabled if $\overline{MCLR}$ functionality is disabled. When not enabled as the PORTE pin, they are disabled and read as '0'. The 28-pin devices do not have only RE3 implemented.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices				Power-on Reset, Brown-out Reset	MCLR Resets WDT Reset RESET Instruction Stack Resets	Wake-up via WDT or Interrupt
CAP3BUFL/ MAXCNTL	2331	2431	4331	4431	xxxx xxxx	uuuu uuuu	uuuu uuuu
CAP1CON	2331	2431	4331	4431	-0-- 0000	-0-- 0000	-u-- uuuu
CAP2CON	2331	2431	4331	4431	-0-- 0000	-0-- 0000	-u-- uuuu
CAP3CON	2331	2431	4331	4431	-0-- 0000	-0-- 0000	-u-- uuuu
DFLTCON	2331	2431	4331	4431	-000 0000	-000 0000	-uuu uuuu

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition. Shaded cells indicate conditions do not apply for the designated device.

Note 1: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

4: See Table 5-2 for Reset value for specific condition.

5: Bits 6 and 7 of PORTA, LATA and TRISA are enabled depending on the oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.

6: Bit 3 of PORTE and LATE are enabled if MCLR functionality is disabled. When not enabled as the PORTE pin, they are disabled and read as '0'. The 28-pin devices do not have only RE3 implemented.

NOTES:

6.0 MEMORY ORGANIZATION

There are three memory types in enhanced MCU devices. These memory types are:

- Program Memory
- Data RAM
- Data EEPROM

As Harvard architecture devices, the data and program memories use separate buses, enabling concurrent access of the two memory spaces. The data EEPROM, for practical purposes, can be regarded as a peripheral device, since it is addressed and accessed through a set of control registers.

Additional detailed information on the operation of the Flash program memory is provided in **Section 8.0 "Flash Program Memory"**. Data EEPROM is discussed separately in **Section 7.0 "Data EEPROM Memory"**.

6.1 Program Memory Organization

PIC18 microcontrollers implement a 21-bit program counter that can address a 2-Mbyte program memory space. Accessing a location between the upper boundary of the physically implemented memory and the 2-Mbyte address will return all '0's (a NOP instruction).

The PIC18F2331/4331 devices each have 8 Kbytes of Flash memory and can store up to 4,096 single-word instructions.

The PIC18F2431/4431 devices each have 16 Kbytes of Flash memory and can store up to 8,192 single-word instructions.

PIC18 devices have two interrupt vectors. The Reset vector address is at 000000h and the interrupt vector addresses are at 000008h and 000018h.

The program memory maps for PIC18F2331/4331 and PIC18F2431/4431 devices are shown in Figure 6-1 and Figure 6-2, respectively.

FIGURE 6-1: PROGRAM MEMORY MAP AND STACK FOR PIC18F2331/4331

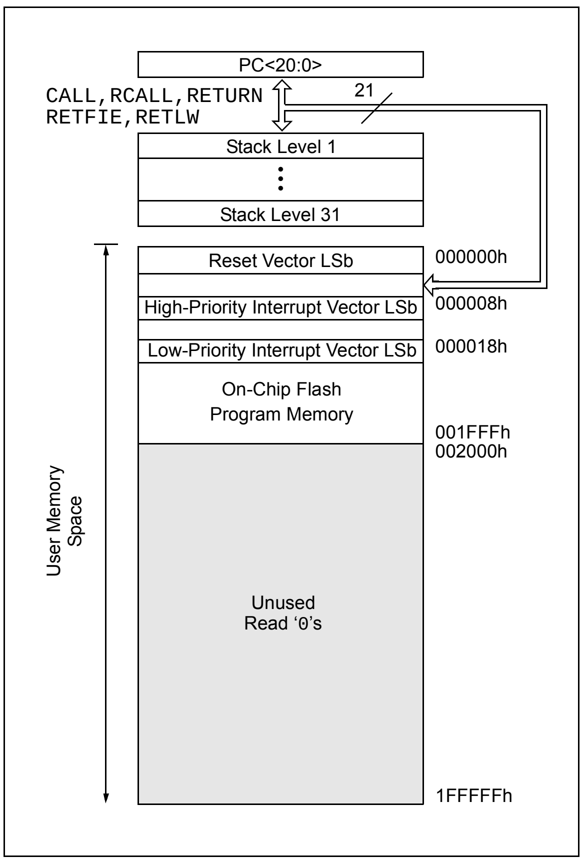
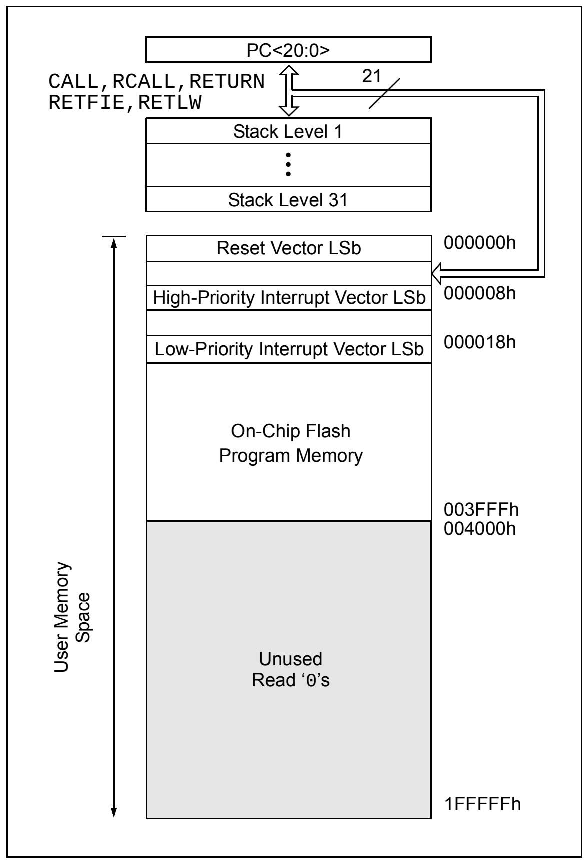


FIGURE 6-2: PROGRAM MEMORY MAP AND STACK FOR PIC18F2431/4431



6.1.1 PROGRAM COUNTER

The Program Counter (PC) specifies the address of the instruction to fetch for execution. The PC is 21 bits wide and contained in three 8-bit registers. The low byte, known as the PCL register, is both readable and writable. The high byte (PCH register) contains the PC<15:8> bits and is not directly readable or writable.

Updates to the PCH register are performed through the PCLATH register. The upper byte is the PCU register and contains the bits, PC<20:16>. This register is also not directly readable or writable. Updates to the PCU register are performed through the PCLATU register.

The contents of PCLATH and PCLATU are transferred to the program counter by any operation that writes to the PCL. Similarly, the upper two bytes of the program counter are transferred to PCLATH and PCLATU by an operation that reads PCL. This is useful for computed offsets to the PC (see **Section 6.1.4.1 "Computed GOTO"**).

The PC addresses bytes in the program memory. To prevent the PC from becoming misaligned with word instructions, the Least Significant bit of the PCL is fixed to a value of '0'. The PC increments by two to address sequential instructions in the program memory.

The CALL, RCALL, GOTO and program branch instructions write to the program counter directly. For these instructions, the contents of PCLATH and PCLATU are not transferred to the program counter.

6.1.2 RETURN ADDRESS STACK

The return address stack allows any combination of up to 31 program calls and interrupts to occur. The PC (Program Counter) is pushed onto the stack when a CALL or RCALL instruction is executed, or an interrupt is Acknowledged. The PC value is pulled off the stack on a RETURN, RETLW or a RETFIE instruction. PCLATU and PCLATH are not affected by any of the RETURN or CALL instructions.

The stack operates as a 31-word by 21-bit RAM and a 5-bit Stack Pointer, with the Stack Pointer initialized to 00000b after all Resets. There is no RAM associated with Stack Pointer, 00000b. This is only a Reset value. During a CALL type instruction, causing a push onto the stack, the Stack Pointer is first incremented and the RAM location pointed to by the Stack Pointer is written with the contents of the PC (already pointing to the instruction following the CALL). During a RETURN type instruction, causing a pop from the stack, the contents of the RAM location pointed to by the STKPTR are transferred to the PC and then the Stack Pointer is decremented.

The stack space is not part of either program or data space. The Stack Pointer is readable and writable, and the address on the top of the stack is readable and writable through the Top-of-Stack (TOS) Special Function Registers. Data can also be pushed to, or popped from, the stack using the Top-of-Stack SFRs. Status bits indicate if the stack is full, has overflowed or underflowed.

6.1.2.1 Top-of-Stack Access

The top of the stack is readable and writable. Three register locations, TOSU, TOSH and TOSL, hold the contents of the stack location pointed to by the STKPTR register (Figure 6-3). This allows users to implement a software stack if necessary. After a CALL, RCALL or interrupt, the software can read the pushed value by reading the TOSU, TOSH and TOSL registers. These values can be placed on a user-defined software stack. At return time, the software can replace the TOSU, TOSH and TOSL and do a return.

The user must disable the global interrupt enable bits while accessing the stack to prevent inadvertent stack corruption.

6.1.2.2 Return Stack Pointer (STKPTR)

The STKPTR register (Register 6-1) contains the Stack Pointer value, the STKFUL (Stack Full) status bit and the STKUNF (Stack Underflow) status bits. The value of the Stack Pointer can be 0 through 31. The Stack Pointer increments before values are pushed onto the stack and decrements after values are popped off the stack. At Reset, the Stack Pointer value will be zero. The user may read and write the Stack Pointer value. This feature can be used by a Real-Time Operating System (RTOS) for return stack maintenance.

After the PC is pushed onto the stack 31 times (without popping any values off the stack), the STKFUL bit is set. The STKFUL bit is cleared by software or by a POR.

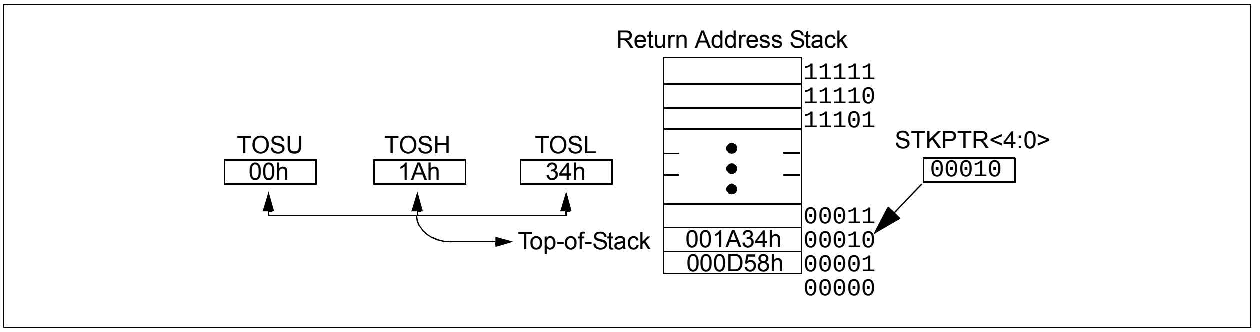
The action that takes place when the stack becomes full depends on the state of the STVREN (Stack Overflow Reset Enable) Configuration bit. (Refer to **Section 23.1 "Configuration Bits"** for a description of the device Configuration bits.) If STVREN is set (default), the 31st push will push the (PC + 2) value onto the stack, set the STKFUL bit and reset the device. The STKFUL bit will remain set and the Stack Pointer will be set to zero.

If STVREN is cleared, the STKFUL bit will be set on the 31st push and the Stack Pointer will increment to 31. Any additional pushes will not overwrite the 31st push and STKPTR will remain at 31.

When the stack has been popped enough times to unload the stack, the next pop will return a value of zero to the PC and set the STKUNF bit, while the Stack Pointer remains at zero. The STKUNF bit will remain set until cleared by software or a POR occurs.

Note:	Returning a value of zero to the PC on an underflow has the effect of vectoring the program to the Reset vector, where the stack conditions can be verified and appropriate actions can be taken. This is not the same as a Reset as the contents of the SFRs are not affected.

FIGURE 6-3: RETURN ADDRESS STACK AND ASSOCIATED REGISTERS



REGISTER 6-1: STKPTR: STACK POINTER REGISTER

R/C-0	R/C-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STKFUL(1)	STKUNF(1)	—	SP4	SP3	SP2	SP1	SP0
bit 7							bit 0

Legend:	C = Clearable bit		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **STKFUL:** Stack Full Flag bit(1)
1 = Stack became full or overflowed
0 = Stack has not become full or overflowed

bit 6 **STKUNF:** Stack Underflow Flag bit(1)
1 = Stack underflow occurred
0 = Stack underflow did not occur

bit 5 **Unimplemented:** Read as ‘0’

bit 4-0 **SP<4:0>:** Stack Pointer Location bits

Note 1: Bit 7 and bit 6 are cleared by user software or by a POR.

6.1.2.3 PUSH and POP Instructions

Since the Top-of-Stack (TOS) is readable and writable, the ability to push values onto the stack and pull values off the stack without disturbing normal program execution is a desirable option. To push the current PC value onto the stack, a PUSH instruction can be executed. This will increment the Stack Pointer and load the current PC value onto the stack. TOSU, TOSH and TOSL can then be modified to place data or a return address on the stack.

The PUSH instruction places the current PC value onto the stack. This increments the Stack Pointer and loads the current PC value onto the stack. The POP instruction discards the current TOS by decrementing the Stack Pointer. The previous value pushed onto the stack then becomes the TOS value.

6.1.2.4 Stack Full/Underflow Resets

These Resets are enabled by programming the STVREN bit in Configuration Register 4L. When the STVREN bit is cleared, a full or underflow condition will set the appropriate STKFUL or STKUNF bit, but not cause a device Reset. When the STVREN bit is set, a full or underflow condition will set the appropriate STKFUL or STKUNF bit and then cause a device Reset. The STKFUL or STKUNF bits are cleared by the user software or a Power-on Reset.

6.1.3 FAST REGISTER STACK

A Fast Register Stack is provided for the STATUS, WREG and BSR registers, to provide a "fast return" option for interrupts. The stack for each register is only one level deep and is neither readable nor writable. It is loaded with the current value of the corresponding register when the processor vectors for an interrupt. All interrupt sources will push values into the stack registers.

The values in the registers are then loaded back into their associated registers if the RETFIE, FAST instruction is used to return from the interrupt. If both low and high-priority interrupts are enabled, the stack registers cannot be used reliably to return from low-priority interrupts. If a high-priority interrupt occurs while servicing a low-priority interrupt, the stack register values stored by the low-priority interrupt will be overwritten. In these cases, users must save the key registers in software during a low-priority interrupt.

If interrupt priority is not used, all interrupts may use the Fast Register Stack for returns from interrupt. If no interrupts are used, the Fast Register Stack can be used to restore the STATUS, WREG and BSR registers at the end of a subroutine call. To use the Fast Register Stack for a subroutine call, a CALL label, FAST instruction must be executed to save the STATUS, WREG and BSR registers to the Fast Register Stack. A RETURN, FAST instruction is then executed to restore these registers from the Fast Register Stack.

Example 6-1 shows a source code example that uses the Fast Register Stack during a subroutine call and return.

EXAMPLE 6-1: FAST REGISTER STACK CODE EXAMPLE

```
CALL  SUB1, FAST    ;STATUS, WREG, BSR
                    ;SAVED IN FAST REGISTER
                    ;STACK
       •
       •

SUB1•
       •
    RETURN FAST      ;RESTORE VALUES SAVED
                    ;IN FAST REGISTER STACK
```

6.1.4 LOOK-UP TABLES IN PROGRAM MEMORY

There may be programming situations that require the creation of data structures, or look-up tables, in program memory. For PIC18 devices, look-up tables can be implemented two ways:

- Computed GOTO
- Table Reads

6.1.4.1 Computed GOTO

A computed GOTO is accomplished by adding an offset to the program counter. An example is shown in Example 6-2.

A look-up table can be formed with an ADDWF PCL instruction and a group of RETLW *nn* instructions. The W register is loaded with an offset into the table before executing a call to that table. The first instruction of the called routine is the ADDWF PCL instruction. The next instruction executed will be one of the RETLW *nn* instructions that returns the value "*nn*" to the calling function.

The offset value (in WREG) specifies the number of bytes that the program counter should advance and should be multiples of 2 (LSb = 0).

In this method, only one data byte can be stored in each instruction location and room on the return address stack is required.

EXAMPLE 6-2: COMPUTED GOTO USING AN OFFSET VALUE

```
        MOVFW     OFFSET
        CALL      TABLE
ORG     0xnn00
TABLE   ADDWF     PCL
        RETLW     0xnn
        RETLW     0xnn
        RETLW     0xnn
       .
       .
       .
```

6.1.4.2 Table Reads and Table Writes

A better method of storing data in program memory allows two bytes of data to be stored in each instruction location. Look-up table data may be stored, two bytes per program word, by using table reads and writes.

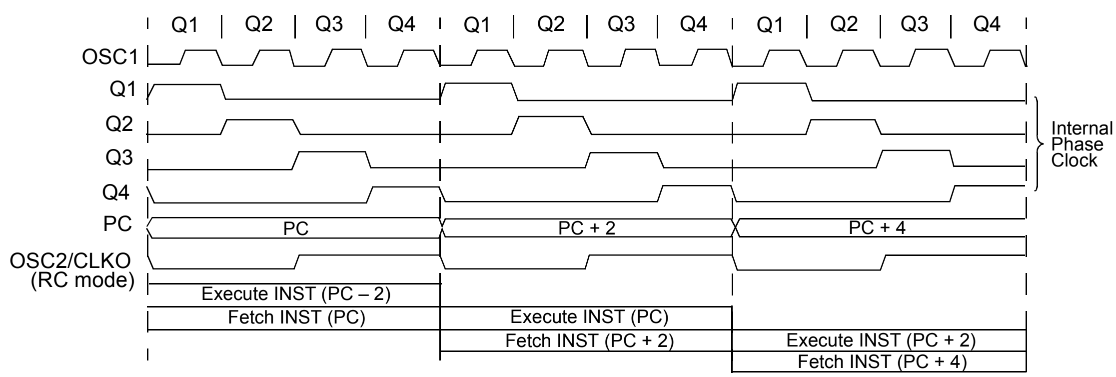
The Table Pointer register (TBLPTR) specifies the byte address and the Table Latch register (TABLAT) contains the data that is read from or written to program memory. Data is transferred to or from program memory, one byte at a time.

Table read and table write operations are discussed further in **Section 8.1 "Table Reads and Table Writes"**.

6.2 Clocking Scheme/Instruction Cycle

The clock input (from OSC1) is internally divided by four to generate four non-overlapping quadrature clocks, namely Q1, Q2, Q3 and Q4. Internally, the Program Counter (PC) is incremented every Q1, the instruction is fetched from the program memory and latched into the Instruction Register (IR) in Q4. The instruction is decoded and executed during the following Q1 through Q4. The clocks and instruction execution flow are shown in Figure 6-4.

FIGURE 6-4: CLOCK/INSTRUCTION CYCLE



6.3 Instruction Flow/Pipelining

An "Instruction Cycle" consists of four Q cycles (Q1, Q2, Q3 and Q4). The instruction fetch and execute are pipelined such that fetch takes one instruction cycle, while decode and execute take another instruction cycle. However, due to the pipelining, each instruction effectively executes in one cycle. If an instruction causes the program counter to change (e.g., `GOTO`), then two cycles are required to complete the instruction (Example 6-3).

A fetch cycle begins with the Program Counter (PC) incrementing in Q1.

In the execution cycle, the fetched instruction is latched into the "Instruction Register" (IR) in cycle, Q1. This instruction is then decoded and executed during the Q2, Q3 and Q4 cycles. Data memory is read during Q2 (operand read) and written during Q4 (destination write).

EXAMPLE 6-3: INSTRUCTION PIPELINE FLOW

	TCY0	TCY1	TCY2	TCY3	TCY4	TCY5
1. `MOVLW 55h`	Fetch 1	Execute 1				
2. `MOVWF PORTB`		Fetch 2	Execute 2			
3. `BRA SUB_1`			Fetch 3	Execute 3		
4. `BSF PORTA, BIT3 (Forced NOP)`				Fetch 4	Flush (NOP)	
5. `Instruction @ address SUB_1`					Fetch SUB_1	Execute SUB_1

All instructions are single cycle, except for any program branches. These take two cycles since the fetch instruction is "flushed" from the pipeline, while the new instruction is being fetched and then executed.

6.4 Instructions in Program Memory

The program memory is addressed in bytes. Instructions are stored as two bytes or four bytes in program memory. The Least Significant Byte of an instruction word is always stored in a program memory location with an even address (LSB = `0`). Figure 6-5 shows an example of how instruction words are stored in the program memory. To maintain alignment with instruction boundaries, the PC increments in steps of 2 and the LSB will always read '`0`'.

The `CALL` and `GOTO` instructions have the absolute program memory address embedded into the instruction. Since instructions are always stored on word boundaries, the data contained in the instruction is a word address. The word address is written to PC<20:1>, which accesses the desired byte address in program memory. Instruction 2 in Figure 6-5 shows how the instruction, '`GOTO 000006h`', is encoded in the program memory. Program branch instructions, which encode a relative address offset, operate in the same manner. The offset value stored in a branch instruction represents the number of single-word instructions that the PC will be offset by. **Section 24.0 "Instruction Set Summary"** provides further details of the instruction set.

6.4.1 TWO-WORD INSTRUCTIONS

The standard PIC18 instruction set has four two-word instructions: `CALL`, `MOVFF`, `GOTO` and `LSFR`. In all cases, the second word of the instructions always has '`1111`' as its four Most Significant bits; the other 12 bits are literal data, usually a data memory address.

The use of '`1111`' in the four MSbs of an instruction specifies a special form of `NOP`. If the instruction is executed in proper sequence, immediately after the first word, the data in the second word is accessed and used by the instruction sequence. If the first word is skipped for some reason and the second word is executed by itself, a `NOP` is executed instead. This is necessary for cases when the two-word instruction is preceded by a conditional instruction that changes the PC. Example 6-4 shows how this works.

> **Note:** For information on two-word instructions in the extended instruction set, see **Section 24.2 "Instruction Set"**.

FIGURE 6-5: INSTRUCTIONS IN PROGRAM MEMORY

Program Memory Byte Locations →			LSB = 1	LSB = 0	Word Address ↓
					000000h
					000002h
					000004h
					000006h
Instruction 1:	`MOVLW`	`055h`	0Fh	55h	000008h
Instruction 2:	`GOTO`	`000006h`	EFh	03h	00000Ah
			F0h	00h	00000Ch
Instruction 3:	`MOVFF`	`123h, 456h`	C1h	23h	00000Eh
			F4h	56h	000010h
					000012h
					000014h

EXAMPLE 6-4: TWO-WORD INSTRUCTIONS

CASE 1:

Object Code	Source Code
`0110 0110 0000 0000`	`TSTFSZ REG1 ; is RAM location 0?`
`1100 0001 0010 0011`	`MOVFF REG1, REG2 ; No, skip this word`
`1111 0100 0101 0110`	`; Execute this word as a NOP`
`0010 0100 0000 0000`	`ADDWF REG3 ; continue code`

CASE 2:

Object Code	Source Code
`0110 0110 0000 0000`	`TSTFSZ REG1 ; is RAM location 0?`
`1100 0001 0010 0011`	`MOVFF REG1, REG2 ; Yes, execute this word`
`1111 0100 0101 0110`	`; 2nd word of instruction`
`0010 0100 0000 0000`	`ADDWF REG3 ; continue code`

6.5 Data Memory Organization

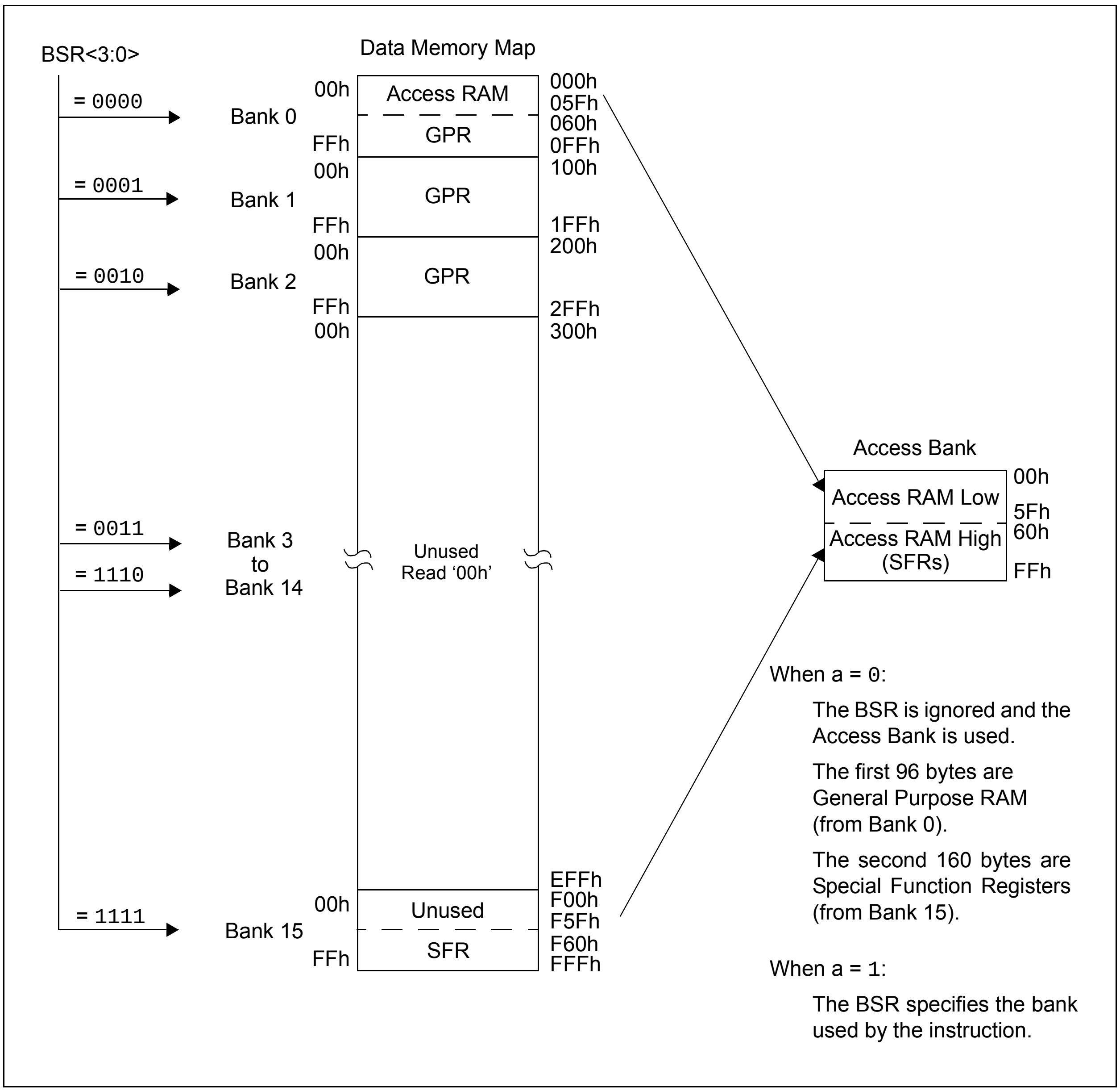
The data memory in PIC18 devices is implemented as static RAM. Each register in the data memory has a 12-bit address, allowing up to 4,096 bytes of data memory. The memory space is divided into as many as 16 banks that contain 256 bytes each. PIC18F2331/2431/4331/4431 devices implement all 16 banks.

Figure 6-6 shows the data memory organization for the PIC18F2331/2431/4331/4431 devices. The data memory contains Special Function Registers (SFRs) and General Purpose Registers (GPRs). The SFRs are used for control and status of the controller and peripheral functions, while GPRs are used for data storage and scratchpad operations in the user's application. Any read of an unimplemented location will read as '0's.

The instruction set and architecture allow operations across all banks. The entire data memory may be accessed by Direct, Indirect or Indexed Addressing modes. Addressing modes are discussed later in this subsection.

To ensure that commonly used registers (SFRs and select GPRs) can be accessed in a single cycle, PIC18 devices implement an Access Bank. This is a 256-byte memory space that provides fast access to SFRs and the lower portion of GPR Bank 0 without using the BSR. **Section 6.5.2 "Access Bank"** provides a detailed description of the Access RAM.

FIGURE 6-6: DATA MEMORY MAP FOR PIC18F2331/2431/4331/4431 DEVICES



6.5.1 BANK SELECT REGISTER (BSR)

Large areas of data memory require an efficient addressing scheme to make rapid access to any address possible. Ideally, this means that an entire address does not need to be provided for each read or write operation. For PIC18 devices, this is accomplished with a RAM banking scheme. This divides the memory space into 16 contiguous banks of 256 bytes.

Depending on the instruction, each location can be addressed directly by its full 12-bit address, or an 8-bit low-order address and a four-bit Bank Pointer. Most instructions in the PIC18 instruction set make use of the Bank Pointer, known as the Bank Select Register (BSR). This SFR holds the four Most Significant bits of a location's address; the instruction itself includes the eight Least Significant bits. Only the four lower bits of the BSR are implemented (BSR<3:0>). The upper four bits are unused; they will always read '0' and cannot be written to. The BSR can be loaded directly by using the MOVLB instruction.

The value of the BSR indicates the bank in data memory. The eight bits in the instruction show the location in the bank and can be thought of as an offset from the bank's lower boundary. The relationship between the BSR's value and the bank division in data memory is shown in Figure 6-6.

Since up to 16 registers may share the same low-order address, the user must always be careful to ensure that the proper bank is selected before performing a data read or write. For example, writing what should be program data to the eight-bit address of F9h, while the BSR is 0Fh, will end up resetting the program counter.

While any bank can be selected, only those banks that are actually implemented can be read or written to. Writes to unimplemented banks are ignored, while reads from unimplemented banks will return '0's. Even so, the STATUS register will still be affected as if the operation was successful. The data memory map in Figure 6-5 indicates which banks are implemented.

In the core PIC18 instruction set, only the MOVFF instruction fully specifies the 12-bit address of the source and target registers. This instruction ignores the BSR completely when it executes. All other instructions include only the low-order address as an operand and must use either the BSR or the Access Bank to locate their target registers.

6.5.2 ACCESS BANK

While the use of the BSR with an embedded 8-bit address allows users to address the entire range of data memory, it also means that the user must always ensure that the correct bank is selected; otherwise, data may be read from or written to the wrong location. This can be disastrous if a GPR is the intended target of an operation, but an SFR is written to instead. Verifying and/or changing the BSR for each read or write to data memory can become very inefficient.

To streamline access for the most commonly used data memory locations, the data memory is configured with an Access Bank, which allows users to access a mapped block of memory without specifying a BSR. The Access Bank consists of the first 128 bytes of memory (00h-7Fh) in Bank 0 and the last 128 bytes of memory (80h-FFh) in Block 15. The lower half is known as the "Access RAM" and is composed of GPRs. This upper half is also where the device's SFRs are mapped. These two areas are mapped contiguously in the Access Bank and can be addressed in a linear fashion by an 8-bit address (Figure 6-6).

The Access Bank is used by core PIC18 instructions that include the Access RAM bit (the 'a' parameter in the instruction). When 'a' is equal to '1', the instruction uses the BSR and the 8-bit address included in the opcode for the data memory address. When 'a' is '0', however, the instruction is forced to use the Access Bank address map; the current value of the BSR is ignored entirely.

Using this "forced" addressing allows the instruction to operate on a data address in a single cycle, without updating the BSR first. For 8-bit addresses of 80h and above, this means that users can evaluate and operate on SFRs more efficiently. The Access RAM below 80h is a good place for data values that the user might need to access rapidly, such as immediate computational results or common program variables. Access RAM also allows for faster and more code efficient context saving and switching of variables.

6.5.3 GENERAL PURPOSE REGISTER (GPR) FILE

PIC18 devices may have banked memory in the GPR area. This is data RAM, which is available for use by all instructions. GPRs start at the bottom of Bank 0 (address 000h) and grow upwards towards the bottom of the SFR area. GPRs are not initialized by a Power-on Reset and are unchanged on all other Resets.

6.5.4 SPECIAL FUNCTION REGISTERS

The Special Function Registers (SFRs) are registers used by the CPU and peripheral modules for controlling the desired operation of the device. These registers are implemented as static RAM. A list of these registers is given in Table 6-1 and Table 6-2.

The SFRs can be classified into two sets: those associated with the "core" function and those related to the peripheral functions. Those registers related to the "core" are described in this section, while those related to the operation of the peripheral features are described in the section of that peripheral feature.

The SFRs are typically distributed among the peripherals whose functions they control.

The unused SFR locations will be unimplemented and read as '0's.

TABLE 6-1: SPECIAL FUNCTION REGISTER MAP FOR PIC18F2331/2431/4331/4431 DEVICES

Address	Name	Address	Name	Address	Name	Address	Name	Address	Name
FFFh	TOSU	FDFh	INDF2(1)	FBFh	CCPR1H	F9Fh	IPR1	F7Fh	PTCON0
FFEh	TOSH	FDEh	POSTINC2(1)	FBEh	CCPR1L	F9Eh	PIR1	F7Eh	PTCON1
FFDh	TOSL	FDDh	POSTDEC2(1)	FBDh	CCP1CON	F9Dh	PIE1	F7Dh	PTMRL
FFCh	STKPTR	FDCh	PREINC2(1)	FBCh	CCPR2H	F9Ch	—(2)	F7Ch	PTMRH
FFBh	PCLATU	FDBh	PLUSW2(1)	FBBh	CCPR2L	F9Bh	OSCTUNE	F7Bh	PTPERL
FFAh	PCLATH	FDAh	FSR2H	FBAh	CCP2CON	F9Ah	ADCON3	F7Ah	PTPERH
FF9h	PCL	FD9h	FSR2L	FB9h	ANSEL1	F99h	ADCHS	F79h	PDC0L
FF8h	TBLPTRU	FD8h	STATUS	FB8h	ANSEL0	F98h	—(2)	F78h	PDC0H
FF7h	TBLPTRH	FD7h	TMR0H	FB7h	T5CON	F97h	—(2)	F77h	PDC1L
FF6h	TBLPTRL	FD6h	TMR0L	FB6h	QEICON	F96h	TRISE(3)	F76h	PDC1H
FF5h	TABLAT	FD5h	T0CON	FB5h	—(2)	F95h	TRISD(3)	F75h	PDC2L
FF4h	PRODH	FD4h	—(2)	FB4h	—(2)	F94h	TRISC	F74h	PDC2H
FF3h	PRODL	FD3h	OSCCON	FB3h	—(2)	F93h	TRISB	F73h	PDC3L(3)
FF2h	INTCON	FD2h	LVDCON	FB2h	—(2)	F92h	TRISA	F72h	PDC3H(3)
FF1h	INTCON2	FD1h	WDTCON	FB1h	—(2)	F91h	PR5H	F71h	SEVTCMPL
FF0h	INTCON3	FD0h	RCON	FB0h	SPBRGH	F90h	PR5L	F70h	SEVTCMPH
FEFh	INDF0(1)	FCFh	TMR1H	FAFh	SPBRG	F8Fh	—(2)	F6Fh	PWMCON0
FEEh	POSTINC0(1)	FCEh	TMR1L	FAEh	RCREG	F8Eh	—(2)	F6Eh	PWMCON1
FEDh	POSTDEC0(1)	FCDh	T1CON	FADh	TXREG	F8Dh	LATE(3)	F6Dh	DTCON
FECh	PREINC0(1)	FCCh	TMR2	FACh	TXSTA	F8Ch	LATD(3)	F6Ch	FLTCONFIG
FEBh	PLUSW0(1)	FCBh	PR2	FABh	RCSTA	F8Bh	LATC	F6Bh	OVDCOND
FEAh	FSR0H	FCAh	T2CON	FAAh	BAUDCON	F8Ah	LATB	F6Ah	OVDCONS
FE9h	FSR0L	FC9h	SSPBUF	FA9h	EEADR	F89h	LATA	F69h	CAP1BUFH
FE8h	WREG	FC8h	SSPADD	FA8h	EEDATA	F88h	TMR5H	F68h	CAP1BUFL
FE7h	INDF1(1)	FC7h	SSPSTAT	FA7h	EECON2	F87h	TMR5L	F67h	CAP2BUFH
FE6h	POSTINC1(1)	FC6h	SSPCON	FA6h	EECON1	F86h	—(2)	F66h	CAP2BUFL
FE5h	POSTDEC1(1)	FC5h	—(2)	FA5h	IPR3	F85h	—(2)	F65h	CAP3BUFH
FE4h	PREINC1(1)	FC4h	ADRESH	FA4h	PIR3	F84h	PORTE	F64h	CAP3BUFL
FE3h	PLUSW1(1)	FC3h	ADRESL	FA3h	PIE3	F83h	PORTD(3)	F63h	CAP1CON
FE2h	FSR1H	FC2h	ADCON0	FA2h	IPR2	F82h	PORTC	F62h	CAP2CON
FE1h	FSR1L	FC1h	ADCON1	FA1h	PIR2	F81h	PORTB	F61h	CAP3CON
FE0h	BSR	FC0h	ADCON2	FA0h	PIE2	F80h	PORTA	F60h	DFLTCON

Note 1: This is not a physical register.
2: Unimplemented registers are read as '0'.
3: This register is not available on 28-pin devices.

TABLE 6-2: REGISTER FILE SUMMARY (PIC18F2331/2431/4331/4431)

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR
TOSU	—	—	—	Top-of-Stack Upper Byte (TOS<20:16>)					---0 0000
TOSH	Top-of-Stack High Byte (TOS<15:8>)								0000 0000
TOSL	Top-of-Stack Low Byte (TOS<7:0>)								0000 0000
STKPTR	STKFUL	STKUNF	—	SP4	SP3	SP2	SP1	SP0	00-0 0000
PCLATU	—	—	bit 21(3)	Holding Register for PC<20:16>					---0 0000
PCLATH	Holding Register for PC<15:8>								0000 0000
PCL	PC Low Byte (PC<7:0>)								0000 0000
TBLPTRU	—	—	bit 21(3)	Program Memory Table Pointer Upper Byte (TBLPTR<20:16>)					--00 0000
TBLPTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)								0000 0000
TBLPTRL	Program Memory Table Pointer Low Byte (TBLPTR<7:0>)								0000 0000
TABLAT	Program Memory Table Latch								0000 0000
PRODH	Product Register High Byte								xxxx xxxx
PRODL	Product Register Low Byte								xxxx xxxx
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	—	TMR0IP	—	RBIP	1111 -1-1
INTCON3	INT2IP	INT1IP	—	INT2IE	INT1IE	—	INT2IF	INT1IF	11-0 0-00
INDF0	Uses contents of FSR0 to address data memory – value of FSR0 not changed (not a physical register)								N/A
POSTINC0	Uses contents of FSR0 to address data memory – value of FSR0 post-incremented (not a physical register)								N/A
POSTDEC0	Uses contents of FSR0 to address data memory – value of FSR0 post-decremented (not a physical register)								N/A
PREINC0	Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register)								N/A
PLUSW0	Uses contents of FSR0 to address data memory – value of FSR0 offset by W (not a physical register)								N/A
FSR0H	—	—	—	—	Indirect Data Memory Address Pointer 0 High				---- xxxx
FSR0L	Indirect Data Memory Address Pointer 0 Low Byte								xxxx xxxx
WREG	Working Register								xxxx xxxx
INDF1	Uses contents of FSR1 to address data memory – value of FSR1 not changed (not a physical register)								N/A
POSTINC1	Uses contents of FSR1 to address data memory – value of FSR1 post-incremented (not a physical register)								N/A
POSTDEC1	Uses contents of FSR1 to address data memory – value of FSR1 post-decremented (not a physical register)								N/A
PREINC1	Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register)								N/A
PLUSW1	Uses contents of FSR1 to address data memory – value of FSR1 offset by W (not a physical register)								N/A
FSR1H	—	—	—	—	Indirect Data Memory Address Pointer 1 High Byte				---- 0000
FSR1L	Indirect Data Memory Address Pointer 1 Low Byte								xxxx xxxx
BSR	—	—	—	—	Bank Select Register				---- 0000
INDF2	Uses contents of FSR2 to address data memory – value of FSR2 not changed (not a physical register)								N/A
POSTINC2	Uses contents of FSR2 to address data memory – value of FSR2 post-incremented (not a physical register)								N/A
POSTDEC2	Uses contents of FSR2 to address data memory – value of FSR2 post-decremented (not a physical register)								N/A
PREINC2	Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register)								N/A
PLUSW2	Uses contents of FSR2 to address data memory – value of FSR2 offset by W (not a physical register)								N/A
FSR2H	—	—	—	—	Indirect Data Memory Address Pointer 2 High Byte				---- 0000
FSR2L	Indirect Data Memory Address Pointer 2 Low Byte								xxxx xxxx
STATUS	—	—	—	N	OV	Z	DC	C	---x xxxx
TMR0H	Timer0 Register High Byte								0000 0000
TMR0L	Timer0 Register Low Byte								xxxx xxxx
T0CON	TMR0ON	T016BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	1111 1111

Legend: x = unknown, u = unchanged, - = unimplemented, q = value depends on condition. Shaded cells are unimplemented.

Note 1: RA6 and associated bits are configured as port pins in RCIO, ECIO and INTIO2 (with port function on RA6) Oscillator modes only and read '0' in all other oscillator modes.

2: RA7 and associated bits are configured as port pins in INTIO2 Oscillator mode only and read '0' in all other modes.

3: Bit 21 of the PC is only available in Test mode and Serial Programming modes.

4: These registers and/or bits are not implemented on the PIC18F2331/2431 devices and read as '0'.

5: The RE3 port bit is only available for PIC18F4331/4431 devices when the MCLRE fuse (CONFIG3H<7>) is programmed to '0'; otherwise, RE3 reads '0'. This bit is read-only.

TABLE 6-2: REGISTER FILE SUMMARY (PIC18F2331/2431/4331/4431) (CONTINUED)

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR
OSCCON	IDLEN	IRCF2	IRCF1	IRCF0	OSTS	IOFS	SCS1	SCS0	0000 q000
LVDCON	—	—	IRVST	LVDEN	LVDL3	LVDL2	LVDL1	LVDL0	--00 0101
WDTCON	WDTW	—	—	—	—	—	—	SWDTEN	0--- ---0
RCON	IPEN	—	—	RI	TO	PD	POR	BOR	0--1 11q0
TMR1H	Timer1 Register High Byte								xxxx xxxx
TMR1L	Timer1 Register Low Byte								xxxx xxxx
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON	0000 0000
TMR2	Timer2 Register								0000 0000
PR2	Timer2 Period Register								1111 1111
T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000
SSPBUF	SSP Receive Buffer/Transmit Register								xxxx xxxx
SSPADD	SSP Address Register in I^{2}C™ Slave mode. SSP Baud Rate Reload Register in I^{2}C Master mode.								0000 0000
SSPSTAT	SMP	CKE	D/A	P	S	R/W	UA	BF	0000 0000
SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000
ADRESH	A/D Result Register High Byte								xxxx xxxx
ADRESL	A/D Result Register Low Byte								xxxx xxxx
ADCON0	—	—	ACONV	ACSCH	ACMOD1	ACMOD0	GO/DONE	ADON	--00 0000
ADCON1	VCFG1	VCFG0	—	FIFOEN	BFEMT	BFOVFL	ADPNT1	ADPNT0	00-0 0000
ADCON2	ADFM	ACQT3	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	0000 0000
ADCON3	ADRS1	ADRS0	—	SSRC4	SSRC3	SSRC2	SSRC1	SSRC0	00-0 0000
ADCHS	GDSEL1	GDSEL0	GBSEL1	GBSEL0	GCSEL1	GCSEL0	GASEL1	GASEL0	0000 0000
CCPR1H	Capture/Compare/PWM Register 1 High Byte								xxxx xxxx
CCPR1L	Capture/Compare/PWM Register 1 Low Byte								xxxx xxxx
CCP1CON	—	—	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000
CCPR2H	Capture/Compare/PWM Register 2 High Byte								xxxx xxxx
CCPR2L	Capture/Compare/PWM Register 2 Low Byte								xxxx xxxx
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	--00 0000
ANSEL1	—	—	—	—	—	—	—	ANS8(4)	---- ---1
ANSEL0	ANS7(4)	ANS6(4)	ANS5(4)	ANS4	ANS3	ANS2	ANS1	ANS0	1111 1111
T5CON	T5SEN	RESEN(4)	T5MOD	T5PS1	T5PS0	T5SYNC	TMR5CS	TMR5ON	0000 0000
QEICON	VELM	QERR	UP/DOWN	QEIM2	QEIM1	QEIM0	PDEC1	PDEC0	0000 0000
SPBRGH	EUSART Baud Rate Generator Register High Byte								0000 0000
SPBRG	EUSART Baud Rate Generator Register Low Byte								0000 0000
RCREG	EUSART Receive Register								0000 0000
TXREG	EUSART Transmit Register								0000 0000
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000X
BAUDCON	—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	-1-1 0-00

Legend: x = unknown, u = unchanged, - = unimplemented, q = value depends on condition. Shaded cells are unimplemented.

Note 1: RA6 and associated bits are configured as port pins in RCIO, ECIO and INTIO2 (with port function on RA6) Oscillator modes only and read '0' in all other oscillator modes.

2: RA7 and associated bits are configured as port pins in INTIO2 Oscillator mode only and read '0' in all other modes.

3: Bit 21 of the PC is only available in Test mode and Serial Programming modes.

4: These registers and/or bits are not implemented on the PIC18F2331/2431 devices and read as '0'.

5: The RE3 port bit is only available for PIC18F4331/4431 devices when the MCLRE fuse (CONFIG3H<7>) is programmed to '0'; otherwise, RE3 reads '0'. This bit is read-only.

TABLE 6-2: REGISTER FILE SUMMARY (PIC18F2331/2431/4331/4431) (CONTINUED)

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR
EEADR	EEPROM Address Register								0000 0000
EEDATA	EEPROM Data Register								0000 0000
EECON2	EEPROM Control Register 2 (not a physical register)								0000 0000
EECON1	EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD	xx-0 x000
IPR3	—	—	—	PTIP	IC3DRIP	IC2QEIP	IC1IP	TMR5IP	---1 1111
PIR3	—	—	—	PTIF	IC3DRIF	IC2QEIF	IC1IF	TMR5IF	---0 0000
PIE3	—	—	—	PTIE	IC3DRIE	IC2QEIE	IC1IE	TMR5IE	---0 0000
IPR2	OSCFIP	—	—	EEIP	—	LVDIP	—	CCP2IP	1--1 -1-1
PIR2	OSCFIF	—	—	EEIF	—	LVDIF	—	CCP2IF	0--0 -0-0
PIE2	OSCFIE	—	—	EEIE	—	LVDIE	—	CCP2IE	0--0 -0-0
IPR1	—	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	-111 1111
PIR1	—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	-000 0000
PIE1	—	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	-000 0000
OSCTUNE	—	—	TUN5	TUN4	TUN3	TUN2	TUN1	TUN0	--00 0000
ADCON3	ADRS1	ADRS0	—	SSRC4	SSRC3	SSRC2	SSRC1	SSRC0	00-0 0000
ADCHS	GDSEL1	GDSEL0	GBSEL1	GBSEL0	GCSEL1	GCSEL0	GASEL1	GASEL0	0000 0000
TRISE(4)	—	—	—	—	—	PORTE Data Direction Register(4)			---- -111
TRISD(4)	PORTD Data Direction Register								1111 1111
TRISC	PORTC Data Direction Register								1111 1111
TRISB	PORTB Data Direction Register								1111 1111
TRISA	TRISA7(2)	TRISA6(1)	PORTA Data Direction Register						1111 1111
PR5H	Timer5 Period Register High Byte								1111 1111
PR5L	Timer5 Period Register Low Byte								1111 1111
LATE(4)	—	—	—	—	—	LATE Data Output Register			---- -xxx
LATD(4)	LATD Data Output Register								xxxx xxxx
LATC	LATC Data Output Register								xxxx xxxx
LATB	LATB Data Output Register								xxxx xxxx
LATA	LATA7(2)	LATA6(1)	LATA Data Output Register						xxxx xxxx
TMR5H	Timer5 Register High Byte								xxxx xxxx
TMR5L	Timer5 Register Low Byte								xxxx xxxx
PORTE	—	—	—	—	RE3(4,5)	RE2(4)	RE1(4)	RE0(4)	---- xxxx
PORTD(4)	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	xxxx xxxx
PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	xxxx xxxx
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx
PORTA	RA7(2)	RA6(1)	RA5	RA4	RA3	RA2	RA1	RA0	xx0x 0000
PTCON0	PTOPS3	PTOPS2	PTOPS1	PTOPS0	PTCKPS1	PTCKPS0	PTMOD1	PTMOD0	0000 0000
PTCON1	PTEN	PTDIR	—	—	—	—	—	—	00-- ----
PTMRL	PWM Time Base Register (lower 8 bits)								0000 0000
PTMRH	UNUSED				PWM Time Base Register (upper 4 bits)				---- 0000
PTPERL	PWM Time Base Period Register (lower 8 bits)								1111 1111
PTPERH	UNUSED				PWM Time Base Period Register (upper 4 bits)				---- 1111

Legend: x = unknown, u = unchanged, - = unimplemented, q = value depends on condition. Shaded cells are unimplemented.

Note 1: RA6 and associated bits are configured as port pins in RCIO, ECIO and INTIO2 (with port function on RA6) Oscillator modes only and read '0' in all other oscillator modes.

2: RA7 and associated bits are configured as port pins in INTIO2 Oscillator mode only and read '0' in all other modes.

3: Bit 21 of the PC is only available in Test mode and Serial Programming modes.

4: These registers and/or bits are not implemented on the PIC18F2331/2431 devices and read as '0'.

5: The RE3 port bit is only available for PIC18F4331/4431 devices when the MCLRE fuse (CONFIG3H<7>) is programmed to '0'; otherwise, RE3 reads '0'. This bit is read-only.

TABLE 6-2: REGISTER FILE SUMMARY (PIC18F2331/2431/4331/4431) (CONTINUED)

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR
PDC0L	PWM Duty Cycle #0L Register (lower 8 bits)								0000 0000
PDC0H	UNUSED		PWM Duty Cycle #0H Register (upper 6 bits)						--00 0000
PDC1L	PWM Duty Cycle #1L Register (lower 8 bits)								0000 0000
PDC1H	UNUSED		PWM Duty Cycle #1H Register (upper 6 bits)						--00 0000
PDC2L	PWM Duty Cycle #2L Register (lower 8 bits)								0000 0000
PDC2H	UNUSED		PWM Duty Cycle #2H Register (upper 6 bits)						--00 0000
PDC3L(4)	PWM Duty Cycle #3L Register (lower 8 bits)								0000 0000
PDC3H(4)	UNUSED		PWM Duty Cycle #3H Register (upper 6 bits)						--00 0000
SEVTCMPL	PWM Special Event Compare Register (lower 8 bits)								0000 0000
SEVTCMPH	UNUSED				PWM Special Event Compare Register (upper 4 bits)				---- 0000
PWMCON0	—	PWMEN2	PWMEN1	PWMEN0	PMOD3	PMOD2	PMOD1	PMOD0	-111 0000
PWMCON1	SEVOPS3	SEVOPS2	SEVOPS1	SEVOPS0	SEVTDIR	—	UDIS	OSYNC	0000 0-00
DTCON	DTPS1	DTPS0	DT5	DT4	DT3	DT2	DT1	DT0	0000 0000
FLTCONFIG	BRFEN	FLTBS(4)	FLTBMOD(4)	FLTBEN(4)	FLTCON	FLTAS	FLTAMOD	FLTAEN	0000 0000
OVDCOND	POVD7(4)	POVD6(4)	POVD5	POVD4	POVD3	POVD2	POVD1	POVD0	1111 1111
OVDCONS	POUT7(4)	POUT6(4)	POUT5	POUT4	POUT3	POUT2	POUT1	POUT0	0000 0000
CAP1BUFH/ VELRH	Capture 1 Register High Byte/Velocity Register High Byte								xxxx xxxx
CAP1BUFL/ VELRL	Capture 1 Register Low Byte/Velocity Register Low Byte								xxxx xxxx
CAP2BUFH/ POSCNTH	Capture 2 Register High Byte/QEI Position Counter Register High Byte								xxxx xxxx
CAP2BUFL/ POSCNTL	Capture 2 Register Low Byte/QEI Position Counter Register Low Byte								xxxx xxxx
CAP3BUFH/ MAXCNTH	Capture 3 Register High Byte/QEI Max. Count Limit Register High Byte								xxxx xxxx
CAP3BUFL/ MAXCNTL	Capture 3 Register Low Byte/QEI Max. Count Limit Register Low Byte								xxxx xxxx
CAP1CON	—	CAP1REN	—	—	CAP1M3	CAP1M2	CAP1M1	CAP1M0	-0-- 0000
CAP2CON	—	CAP2REN	—	—	CAP2M3	CAP2M2	CAP2M1	CAP2M0	-0-- 0000
CAP3CON	—	CAP3REN	—	—	CAP3M3	CAP3M2	CAP3M1	CAP3M0	-0-- 0000
DFLTCON	—	FLT4EN	FLT3EN	FLT2EN	FLT1EN	FLTCK2	FLTCK1	FLTCK0	-000 0000

Legend: x = unknown, u = unchanged, - = unimplemented, q = value depends on condition. Shaded cells are unimplemented.

Note 1: RA6 and associated bits are configured as port pins in RCIO, ECIO and INTIO2 (with port function on RA6) Oscillator modes only and read '0' in all other oscillator modes.

2: RA7 and associated bits are configured as port pins in INTIO2 Oscillator mode only and read '0' in all other modes.

3: Bit 21 of the PC is only available in Test mode and Serial Programming modes.

4: These registers and/or bits are not implemented on the PIC18F2331/2431 devices and read as '0'.

5: The RE3 port bit is only available for PIC18F4331/4431 devices when the MCLRE fuse (CONFIG3H<7>) is programmed to '0'; otherwise, RE3 reads '0'. This bit is read-only.

6.6 STATUS Register

The STATUS register, shown in Register 6-2, contains the arithmetic status of the ALU. The STATUS register can be the operand for any instruction, as with any other register. If the STATUS register is the destination for an instruction that affects the Z, DC, C, OV or N bits, then the write to these five bits is disabled. These bits are set or cleared according to the device logic. Therefore, the result of an instruction with the STATUS register as destination may be different than intended.

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

It is recommended, therefore, that only `BCF`, `BSF`, `SWAPF`, `MOVFF` and `MOVWF` instructions are used to alter the STATUS register, because these instructions do not affect the Z, C, DC, OV or N bits in the STATUS register. For other instructions not affecting any Status bits, see Table 24-2.

Note:	The C and DC bits operate as a $\overline{\text{Borrow}}$ and $\overline{\text{Digit Borrow}}$ bit respectively, in subtraction.

REGISTER 6-2: STATUS REGISTER

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	N	OV	Z	DC(1)	C(2)
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-5 **Unimplemented:** Read as ‘`0`’

bit 4 **N:** Negative bit

This bit is used for signed arithmetic (2’s complement). It indicates whether the result was negative (ALU MSB = `1`).

`1` = Result was negative
`0` = Result was positive

bit 3 **OV:** Overflow bit

This bit is used for signed arithmetic (2’s complement). It indicates an overflow of the 7-bit magnitude which causes the sign bit (bit 7) to change state.

`1` = Overflow occurred for signed arithmetic (in this arithmetic operation)
`0` = No overflow occurred

bit 2 **Z:** Zero bit

`1` = The result of an arithmetic or logic operation is zero
`0` = The result of an arithmetic or logic operation is not zero

bit 1 **DC:** Digit Carry/$\overline{\text{Borrow}}$ bit(1)

For `ADDWF, ADDLW, SUBLW` and `SUBWF` instructions:

`1` = A carry-out from the 4th low-order bit of the result occurred
`0` = No carry-out from the 4th low-order bit of the result

bit 0 **C:** Carry/$\overline{\text{Borrow}}$ bit(2)

For `ADDWF, ADDLW, SUBLW` and `SUBWF` instructions:

`1` = A carry-out from the Most Significant bit of the result occurred
`0` = No carry-out from the Most Significant bit of the result occurred

Note 1: For $\overline{\text{Borrow}}$, the polarity is reversed. A subtraction is executed by adding the 2’s complement of the second operand. For rotate (`RRF, RLF`) instructions, this bit is loaded with either bit 4 or bit 3 of the source register.

2: For $\overline{\text{Borrow}}$, the polarity is reversed. A subtraction is executed by adding the 2’s complement of the second operand. For rotate (`RRF, RLF`) instructions, this bit is loaded with either the high or low-order bit of the source register.

6.7 Data Addressing Modes

The data memory space can be addressed in several ways. For most instructions, the addressing mode is fixed. Other instructions may use up to three modes, depending on which operands are used and whether or not the extended instruction set is enabled.

The addressing modes are:

- Inherent
- Literal
- Direct
- Indirect

6.7.1 INHERENT AND LITERAL ADDRESSING

Many PIC18 control instructions do not need any argument at all. They either perform an operation that globally affects the device or they operate implicitly on one register. This addressing mode is known as Inherent Addressing. Examples include SLEEP, RESET and DAW.

Other instructions work in a similar way but require an additional explicit argument in the opcode. This is known as Literal Addressing mode because they require some literal value as an argument. Examples include ADDLW and MOVLW, which respectively, add or move a literal value to the W register. Other examples include CALL and GOTO, which include a 20-bit program memory address.

6.7.2 DIRECT ADDRESSING

Direct Addressing specifies all or part of the source and/or destination address of the operation within the opcode itself. The options are specified by the arguments accompanying the instruction.

In the core PIC18 instruction set, bit-oriented and byte-oriented instructions use some version of Direct Addressing by default. All of these instructions include some 8-bit literal address as their Least Significant Byte. This address specifies either a register address in one of the banks of data RAM (**Section 6.5.4 "Special Function Registers"**) or a location in the Access Bank (**Section 6.5.2 "Access Bank"**) as the data source for the instruction.

The Access RAM bit, 'a', determines how the address is interpreted. When 'a' is '1', the contents of the BSR (**Section 6.5.1 "Bank Select Register (BSR)"**) are used with the address to determine the complete 12-bit address of the register. When 'a' is '0', the address is interpreted as being a register in the Access Bank. Addressing that uses the Access RAM is sometimes also known as Direct Forced Addressing mode.

A few instructions, such as MOVFF, include the entire 12-bit address (either source or destination) in their op codes. In these cases, the BSR is ignored entirely.

The destination of the operation's results is determined by the destination bit, 'd'. When 'd' is '1', the results are stored back in the source register, overwriting its original contents. When 'd' is '0', the results are stored in the W register. Instructions without the 'd' argument have a destination that is implicit in the instruction; their destination is either the target register being operated on or the W register.

6.7.3 INDIRECT ADDRESSING

Indirect Addressing allows the user to access a location in data memory without giving a fixed address in the instruction. This is done by using File Select Registers (FSRs) as pointers to the locations to be read or written to. Since the FSRs are themselves located in RAM as Special Function Registers, they can also be directly manipulated under program control. This makes FSRs very useful in implementing data structures, such as tables and arrays in data memory.

The registers for Indirect Addressing are also implemented with Indirect File Operands (INDFs) that permit automatic manipulation of the pointer value with auto-incrementing, auto-decrementing or offsetting with another value. This allows for efficient code, using loops, such as the example of clearing an entire RAM bank in Example 6-5.

EXAMPLE 6-5: HOW TO CLEAR RAM (BANK 1) USING INDIRECT ADDRESSING

```
        LFSR   FSR0, 100h ;
NEXT    CLRF   POSTINC0   ; Clear INDF
                          ; register then
                          ; inc pointer
        BTFSS  FSR0H, 1   ; All done with
                          ; Bank1?
        BRA    NEXT       ; NO, clear next
CONTINUE                  ; YES, continue
```

6.7.3.1 FSR Registers and the INDF Operand

At the core of Indirect Addressing are three sets of registers: FSR0, FSR1 and FSR2. Each represents a pair of 8-bit registers, FSRnH and FSRnL. The four upper bits of the FSRnH register are not used so each FSR pair holds a 12-bit value. This represents a value that can address the entire range of the data memory in a linear fashion. The FSR register pairs, then, serve as pointers to data memory locations.

Indirect Addressing is accomplished with a set of Indirect File Operands: INDF0 through INDF2. These can be thought of as "virtual" registers; they are mapped in the SFR space but are not physically implemented. Reading or writing to a particular INDF register actually accesses its corresponding FSR register pair. A read from INDF1, for example, reads the data at the address indicated by FSR1H:FSR1L. Instructions that use the INDF registers as operands actually use the contents of their corresponding FSR as a pointer to the instruction's target. The INDF operand is just a convenient way of using the pointer.

Because Indirect Addressing uses a full 12-bit address, data RAM banking is not necessary. Thus, the current contents of the BSR and the Access RAM bit have no effect on determining the target address.

6.7.3.2 FSR Registers and POSTINC, POSTDEC, PREINC and PLUSW

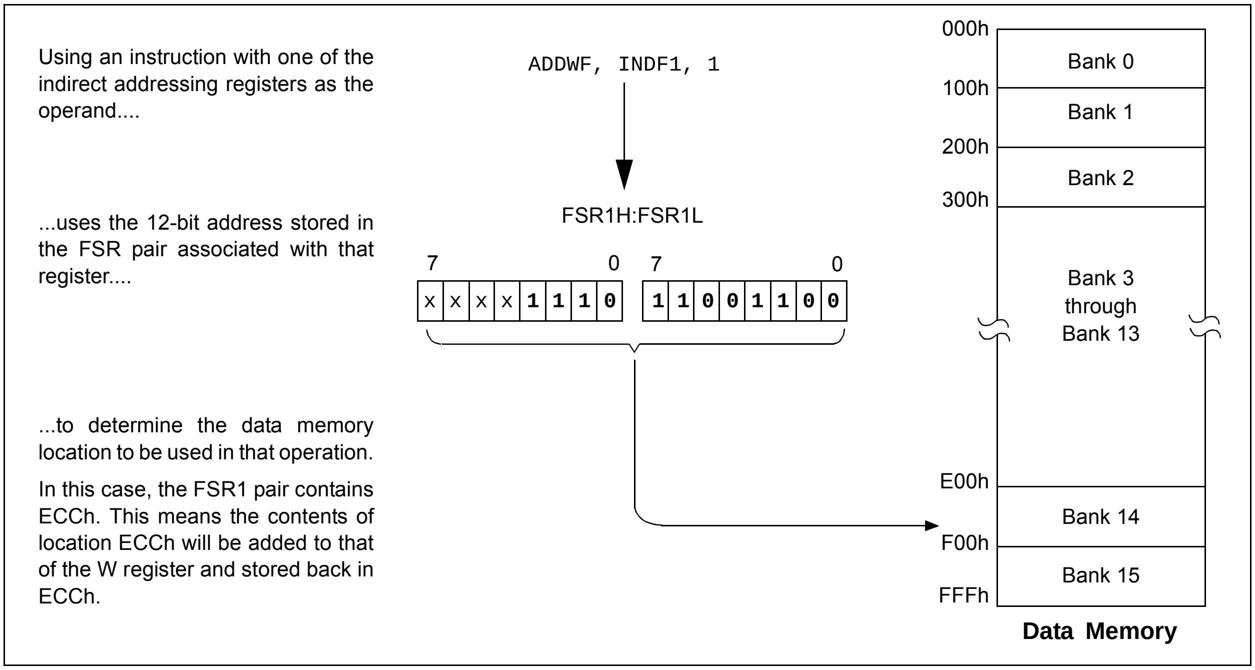
In addition to the INDF operand, each FSR register pair also has four additional indirect operands. Like INDF, these are "virtual" registers that cannot be indirectly read or written to. Accessing these registers actually accesses the associated FSR register pair, but also performs a specific action on its stored value. They are:

- POSTDEC: accesses the FSR value, then automatically decrements it by 1 afterwards
- POSTINC: accesses the FSR value, then automatically increments it by 1 afterwards
- PREINC: increments the FSR value by 1, then uses it in the operation
- PLUSW: adds the signed value of the W register (range of -127 to 128) to that of the FSR and uses the new value in the operation.

In this context, accessing an INDF register uses the value in the FSR registers without changing them. Similarly, accessing a PLUSW register gives the FSR value offset by that in the W register; neither value is actually changed in the operation. Accessing the other virtual registers changes the value of the FSR registers.

Operations on the FSRs with POSTDEC, POSTINC and PREINC affect the entire register pair; that is, rollovers of the FSRnL register from FFh to 00h carry over to the FSRnH register. On the other hand, results of these operations do not change the value of any flags in the STATUS register (e.g., Z, N, OV, etc.).

FIGURE 6-7: INDIRECT ADDRESSING



The PLUSW register can be used to implement a form of Indexed Addressing in the data memory space. By manipulating the value in the W register, users can reach addresses that are fixed offsets from pointer addresses. In some applications, this can be used to implement some powerful program control structure, such as software stacks, inside of data memory.

6.7.3.3 Operations by FSRs on FSRs

Indirect Addressing operations that target other FSRs or virtual registers represent special cases. For example, using an FSR to point to one of the virtual registers will not result in successful operations. As a specific case, assume that FSR0H:FSR0L contain FE7h, the address of INDF1. Attempts to read the value of the INDF1 using INDF0 as an operand will return 00h. Attempts to write to INDF1 using INDF0 as the operand will result in a `NOP`.

On the other hand, using the virtual registers to write to an FSR pair may not occur as planned. In these cases, the value will be written to the FSR pair but without any incrementing or decrementing. Thus, writing to INDF2 or POSTDEC2 will write the same value to the FSR2H:FSR2L.

Since the FSRs are physical registers mapped in the SFR space, they can be manipulated through all direct operations. Users should proceed cautiously when working on these registers, particularly if their code uses Indirect Addressing.

Similarly, operations by Indirect Addressing are generally permitted on all other SFRs. Users should exercise the appropriate caution that they do not inadvertently change settings that might affect the operation of the device.

NOTES:

7.0 DATA EEPROM MEMORY

The data EEPROM is readable and writable during normal operation over the entire VDD range. The data memory is not directly mapped in the register file space. Instead, it is indirectly addressed through the Special Function Registers (SFR).

There are four SFRs used to read and write the program and data EEPROM memory. These registers are:

- EECON1
- EECON2
- EEDATA
- EEADR

The EEPROM data memory allows byte read and write. When interfacing to the data memory block, EEDATA holds the 8-bit data for read/write and EEADR holds the address of the EEPROM location being accessed. These devices have 256 bytes of data EEPROM with an address range from 00h to FFh.

The EEPROM data memory is rated for high erase/write cycle endurance. A byte write automatically erases the location and writes the new data (erase-before-write). The write time is controlled by an on-chip timer. The write time will vary with voltage and temperature, as well as from chip-to-chip. Please refer to Parameter D122 (Table 26-1 in **Section 26.0 "Electrical Characteristics"**) for exact limits.

7.1 EEADR

The Address register can address 256 bytes of data EEPROM.

7.2 EECON1 and EECON2 Registers

Access to the data EEPROM is controlled by two registers: EECON1 and EECON2. These are the same registers which control access to the program memory and are used in a similar manner for the data EEPROM.

The EECON1 register (Register 7-1) is the control register for data and program memory access. Control bit, EEPGD, determines if the access will be to program or data EEPROM memory. When clear, operations will access the data EEPROM memory. When set, program memory is accessed.

Control bit, CFGS, determines if the access will be to the Configuration registers or to program memory/data EEPROM memory. When set, subsequent operations access Configuration registers. When CFGS is clear, the EEPGD bit selects either Flash program or data EEPROM memory.

The WREN bit, when set, will allow a write operation. On power-up, the WREN bit is clear. The WRERR bit is set in hardware when the WREN bit is set and cleared when the internal programming timer expires and the write operation is complete.

Note:	During normal operation, the WRERR bit is read as '1'. This can indicate that a write operation was prematurely terminated by a Reset or a write operation was attempted improperly.

The WR control bit initiates write operations. The bit cannot be cleared, only set, in software; it is cleared in hardware at the completion of the write operation.

Note:	The EEIF interrupt flag bit (PIR2<4>) is set when the write is complete. It must be cleared in software.

Control bits, RD and WR, start read and erase/write operations, respectively. These bits are set by firmware and cleared by hardware at the completion of the operation.

The RD bit cannot be set when accessing program memory (EEPGD = 1). Program memory is read using table read instructions. See **Section 7.3 "Reading the Data EEPROM Memory"** regarding table reads.

The EECON2 register is not a physical register. It is used exclusively in the memory write and erase sequences. Reading EECON2 will read all '0's.

REGISTER 7-1: EECON1: EEPROM CONTROL REGISTER 1

R/W-x	R/W-x	U-0	R/W-0	R/W-x	R/W-0	R/S-0	R/S-0
EEPGD	CFGS	—	FREE	WRERR[(1)]	WREN	WR	RD
bit 7							bit 0

Legend:	S = Settable bit (cannot be cleared in software)		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **EEPGD:** Flash Program or Data EEPROM Memory Select bit
1 = Access Flash program memory
0 = Access data EEPROM memory

bit 6 **CFGS:** Flash Program/Data EEPROM or Configuration Select bit
1 = Access Configuration registers
0 = Access Flash program or data EEPROM memory

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

bit 4 **FREE:** Flash Row Erase Enable bit
1 = Erase the program memory row addressed by TBLPTR on the next WR command (cleared by completion of erase operation)
0 = Perform write only

bit 3 **WRERR:** Flash Program/Data EEPROM Error Flag bit[(1)]
1 = A write operation is prematurely terminated (any Reset during self-timed programming in normal operation, or an improper write attempt)
0 = The write operation completed

bit 2 **WREN:** Flash Program/Data EEPROM Write Enable bit
1 = Allows write cycles to Flash program/data EEPROM
0 = Inhibits write cycles to Flash program/data EEPROM

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

bit 0 **RD:** Read Control bit
1 = Initiates an EEPROM read (Read takes one cycle. RD is cleared in hardware. The RD bit can only be set (not cleared) in software. RD bit cannot be set when EEPGD = 1 or CFGS = 1.)
0 = Does not initiate an EEPROM read

Note 1: When a WRERR occurs, the EEPGD and CFGS bits are not cleared. This allows tracing of the error condition.

7.3 Reading the Data EEPROM Memory

To read a data memory location, the user must write the address to the EEADR register, clear the EEPGD control bit (EECON1<7>) and then set control bit, RD (EECON1<0>). The data is available for the very next instruction cycle; therefore, the EEDATA register can be read by the next instruction. EEDATA will hold this value until another read operation, or until it is written to by the user (during a write operation).The basic process is shown in Example 7-1.

7.4 Writing to the Data EEPROM Memory

To write an EEPROM data location, the address must first be written to the EEADR register and the data written to the EEDATA register. The sequence in Example 7-2 must be followed to initiate the write cycle.

The write will not begin if this sequence is not exactly followed (write 55h to EECON2, write 0AAh to EECON2, then set WR bit) for each byte. It is strongly recommended that interrupts be disabled during this code segment.

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

After a write sequence has been initiated, EECON1, EEADR and EEDATA cannot be modified. The WR bit will be inhibited from being set unless the WREN bit is set. The WREN bit must be set on a previous instruction. Both WR and WREN cannot be set with the same instruction.

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

7.5 Write Verify

Depending on the application, good programming practice may dictate that the value written to the memory should be verified against the original value. This should be used in applications where excessive writes can stress bits near the specification limit.

7.6 Protection Against Spurious Write

There are conditions when the device may not want to write to the data EEPROM memory. To protect against spurious EEPROM writes, various mechanisms have been built-in. On power-up, the WREN bit is cleared. Also, the Power-up Timer (72 ms duration) prevents EEPROM write.

The write initiate sequence and the WREN bit together help prevent an accidental write during brown-out, power glitch, or software malfunction.

EXAMPLE 7-1: DATA EEPROM READ

```
        MOVLW   DATA_EE_ADDR      ;
        MOVWF   EEADR             ; Data Memory Address to read
        BCF     EECON1, EEPGD     ; Point to DATA memory
        BSF     EECON1, RD        ; EEPROM Read
        MOVF    EEDATA, W         ; W = EEDATA
```

EXAMPLE 7-2: DATA EEPROM WRITE

```
            MOVLW   DATA_EE_ADDR      ;
            MOVWF   EEADR             ; Data Memory Address to write
            MOVLW   DATA_EE_DATA      ;
            MOVWF   EEDATA            ; Data Memory Value to write
            BCF     EECON1, EEPGD     ; Point to DATA memory
            BCF     EECON1, CFGS      ; Access EEPROM
            BSF     EECON1, WREN      ; Enable writes
            BCF     INTCON, GIE       ; Disable Interrupts
            MOVLW   55h               ;
Required    MOVWF   EECON2            ; Write 55h
Sequence    MOVLW   0AAh              ;
            MOVWF   EECON2            ; Write 0AAh
            BSF     EECON1, WR        ; Set WR bit to begin write
            BTFSC   EECON1, WR        ; Wait for write to complete

            GOTO    $-2               ;
            BSF     INTCON, GIE       ; Enable interrupts
```

7.7 Operation During Code-Protect

Data EEPROM memory has its own code-protect bits in Configuration Words. External read and write operations are disabled if either of these mechanisms are enabled.

The microcontroller itself can both read and write to the internal data EEPROM, regardless of the state of the code-protect Configuration bit. Refer to **Section 23.0 "Special Features of the CPU"** for additional information.

7.8 Protection Against Spurious Write

There are conditions when the device may not want to write to the data EEPROM memory. To protect against spurious EEPROM writes, various mechanisms have been implemented. On power-up, the WREN bit is cleared. In addition, writes to the EEPROM memory are blocked during the Power-up Timer period (TPWRT, Parameter 33).

The write/initiate sequence, and the WREN bit together, help prevent an accidental write during Brown-out Reset, power glitch or software malfunction.

7.9 Using the Data EEPROM

The data EEPROM is a high-endurance, byte-addressable array that has been optimized for the storage of frequently changing information (e.g., program variables or other data that are updated often). Frequently changing values will typically be updated more often than Specification D124. If this is not the case, an array refresh must be performed. For this reason, variables that change infrequently (such as constants, IDs, calibration, etc.) should be stored in Flash program memory.

A simple data EEPROM refresh routine is shown in Example 7-3.

Note:	If data EEPROM is only used to store constants and/or data that changes rarely, an array refresh is likely not required. See Specification D124.

EXAMPLE 7-3: DATA EEPROM REFRESH ROUTINE

```
            CLRF    EEADR           ; Start at address 0
            BCF     EECON1, CFGS    ; Set for memory
            BCF     EECON1, EEPGD   ; Set for Data EEPROM
            BCF     INTCON, GIE     ; Disable interrupts
            BSF     EECON1, WREN    ; Enable writes
LOOP                                ; Loop to refresh array
            BSF     EECON1, RD      ; Read current address
            MOVLW   55h             ;
            MOVWF   EECON2          ; Write 55h
Required    MOVLW   0AAh            ;
Sequence    MOVWF   EECON2          ; Write 0AAh
            BSF     EECON1, WR      ; Set WR bit to begin write
            BTFSC   EECON1, WR      ; Wait for write to complete
            BRA     $-2
            INCFSZ  EEADR, F        ; Increment address
            BRA     LOOP            ; Not zero, do it again

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

TABLE 7-1: REGISTERS ASSOCIATED WITH DATA EEPROM MEMORY

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
EEADR	EEPROM Address Register								56
EEDATA	EEPROM Data Register								56
EECON2	EEPROM Control Register 2 (not a physical register)								56
EECON1	EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD	56
IPR2	OSCFIP	—	—	EEIP	—	LVDIP	—	CCP2IP	57
PIR2	OSCFIF	—	—	EEIF	—	LVDIF	—	CCP2IF	57
PIE2	OSCFIE	—	—	EEIE	—	LVDIE	—	CCP2IE	57

Legend: — = unimplemented, read as '0'. Shaded cells are not used during Flash/EEPROM access.

NOTES:

8.0 FLASH PROGRAM MEMORY

The Flash program memory is readable, writable and erasable during normal operation over the entire VDD range.

A read from program memory is executed on one byte at a time. A write to program memory is executed on blocks of 8 bytes at a time. Program memory is erased in blocks of 64 bytes at a time. A bulk erase operation may not be issued from user code.

While writing or erasing program memory, instruction fetches cease until the operation is complete. The program memory cannot be accessed during the write or erase, therefore, code cannot execute. An internal programming timer terminates program memory writes and erases.

A value written to program memory does not need to be a valid instruction. Executing a program memory location that forms an invalid instruction results in a NOP.

8.1 Table Reads and Table Writes

In order to read and write program memory, there are two operations that allow the processor to move bytes between the program memory space and the data RAM:

- Table Read (TBLRD)
- Table Write (TBLWT)

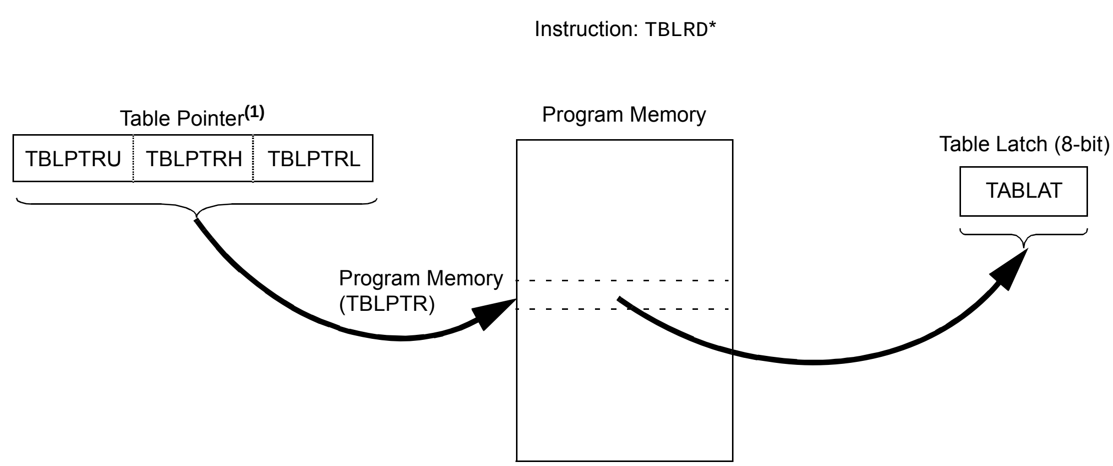
The program memory space is 16 bits wide, while the data RAM space is 8 bits wide. Table reads and table writes move data between these two memory spaces through an 8-bit register (TABLAT).

Table read operations retrieve data from program memory and place it into TABLAT in the data RAM space. Figure 8-1 shows the operation of a table read with program memory and data RAM.

Table write operations store data from TABLAT in the data memory space into holding registers in program memory. The procedure to write the contents of the holding registers into program memory is detailed in **Section 8.5 "Writing to Flash Program Memory"**. Figure 8-2 shows the operation of a table write with program memory and data RAM.

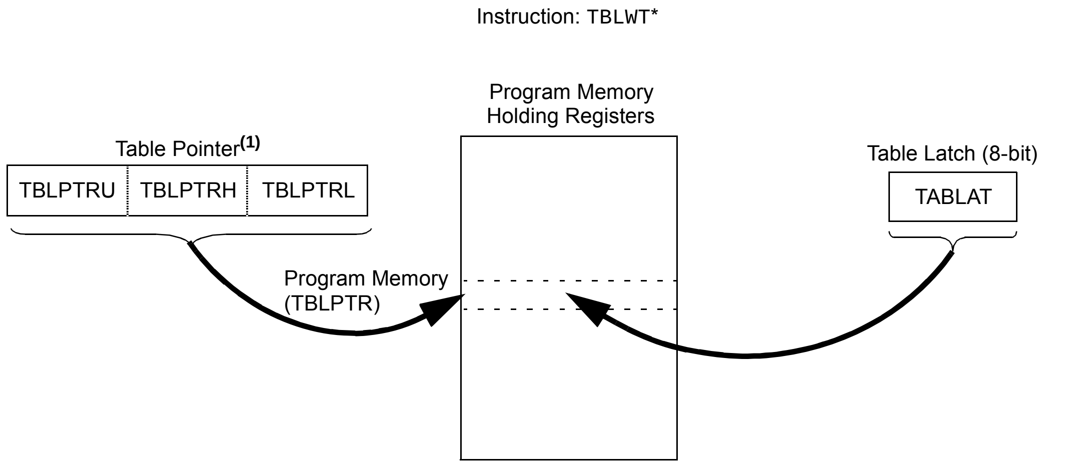
Table operations work with byte entities. A table block containing data, rather than program instructions, is not required to be word-aligned. Therefore, a table block can start and end at any byte address. If a table write is being used to write executable code into program memory, program instructions will need to be word-aligned, (TBLPTRL<0> = 0).

FIGURE 8-1: TABLE READ OPERATION



Note 1: The Table Pointer points to a byte in program memory.

FIGURE 8-2: TABLE WRITE OPERATION



Note 1: The Table Pointer actually points to one of eight holding registers, the address of which is determined by TBLPTRL<2:0>. The process for physically writing data to the program memory array is discussed in **Section 8.5 "Writing to Flash Program Memory"**.

8.2 Control Registers

Several control registers are used in conjunction with the `TBLRD` and `TBLWT` instructions. These include the:

- EECON1 register
- EECON2 register
- TABLAT register
- TBLPTR registers

8.2.1 EECON1 AND EECON2 REGISTERS

EECON1 is the control register for memory accesses.

EECON2 is not a physical register. Reading EECON2 will read all '0's. The EECON2 register is used exclusively in the memory write and erase sequences.

Control bit, EEPGD, determines if the access will be to program or data EEPROM memory. When clear, operations will access the data EEPROM memory. When set, program memory is accessed.

Control bit, CFGS, determines if the access will be to the Configuration registers or to program memory/data EEPROM memory. When set, subsequent operations access Configuration registers, regardless of EEPGD. (See **Section 23.0 "Special Features of the CPU"**.) When CFGS is clear, the EEPGD bit selects either program Flash or data EEPROM memory.

The FREE bit controls program memory erase operations. When the FREE bit is set, the erase operation is initiated on the next WR command. When FREE is clear, only writes are enabled.

A write operation is allowed when the WREN bit (EECON1<2>) is set. On power-up, the WREN bit is clear. The WRERR bit (EECON1<3>) is set in hardware when the WR bit (EECON1<1>) is set and cleared when the internal programming timer expires and the write operation is complete.

Note:	During normal operation, the WRERR may read as '1'. This can indicate that a write operation was prematurely terminated by a Reset or a write operation was attempted improperly.

The WR control bit initiates write operations. The bit cannot be cleared, only set, in software. The bit is cleared in hardware at the completion of the write operation.

Note:	The EEIF interrupt flag bit (PIR2<4>) is set when the write is complete. It must be cleared in software.

REGISTER 8-1: EECON1: DATA EEPROM CONTROL REGISTER 1

R/W-x	R/W-x	U-0	R/W-0	R/W-x	R/W-0	R/S-0	R/S-0
EEPGD	CFGS	—	FREE	WRERR[(1)]	WREN	WR	RD
bit 7							bit 0

Legend:	S = Settable bit (cannot be cleared in software)		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **EEPGD:** Flash Program or Data EEPROM Memory Select bit
1 = Access Flash program memory
0 = Access data EEPROM memory

bit 6 **CFGS:** Flash Program/Data EEPROM or Configuration Select bit
1 = Access Configuration registers
0 = Access Flash program or data EEPROM memory

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

bit 4 **FREE:** Flash Row Erase Enable bit
1 = Erase the program memory row addressed by TBLPTR on the next WR command (cleared by completion of erase operation)
0 = Perform write only

bit 3 **WRERR:** Flash Program/Data EEPROM Error Flag bit[(1)]
1 = A write operation is prematurely terminated (any Reset during self-timed programming in normal operation, or an improper write attempt)
0 = The write operation completed

bit 2 **WREN:** Flash Program/Data EEPROM Write Enable bit
1 = Allows write cycles to Flash program/data EEPROM
0 = Inhibits write cycles to Flash program/data EEPROM

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

bit 0 **RD:** Read Control bit
1 = Initiates an EEPROM read (Read takes one cycle. RD is cleared in hardware. The RD bit can only be set (not cleared) in software. RD bit cannot be set when EEPGD = 1 or CFGS = 1.)
0 = Does not initiate an EEPROM read

Note 1: When a WRERR occurs, the EEPGD and CFGS bits are not cleared. This allows tracing of the error condition.

8.2.2 TABLAT – TABLE LATCH REGISTER

The Table Latch (TABLAT) is an 8-bit register mapped into the SFR space. The Table Latch is used to hold 8-bit data during data transfers between program memory and data RAM.

8.2.3 TBLPTR – TABLE POINTER REGISTER

The Table Pointer (TBLPTR) addresses a byte within the program memory. The TBLPTR is comprised of three SFR registers: Table Pointer Upper Byte, Table Pointer High Byte and Table Pointer Low Byte (TBLPTRU:TBLPTRH:TBLPTRL). These three registers join to form a 22-bit wide pointer. The low-order 21 bits allow the device to address up to 2 Mbytes of program memory space. Setting the 22nd bit allows access to the Device ID, the User ID and the Configuration bits.

The TBLPTR is used by the TBLRD and TBLWT instructions. These instructions can update the TBLPTR in one of four ways based on the table operation. These operations are shown in Table 8-1. These operations on the TBLPTR only affect the low-order 21 bits.

8.2.4 TABLE POINTER BOUNDARIES

TBLPTR is used in reads, writes and erases of the Flash program memory.

When a TBLRD is executed, all 22 bits of the Table Pointer determine which byte is read from program or configuration memory into TABLAT.

When a TBLWT is executed, the three LSbs of the Table Pointer (TBLPTR<2:0>) determine which of the eight program memory holding registers is written to. When the timed write to program memory (long write) begins, the 19 MSbs of the Table Pointer, TBLPTR (TBLPTR<21:3>), will determine which program memory block of 8 bytes is written to (TBLPTR<2:0> are ignored). For more detail, see **Section 8.5 "Writing to Flash Program Memory"**.

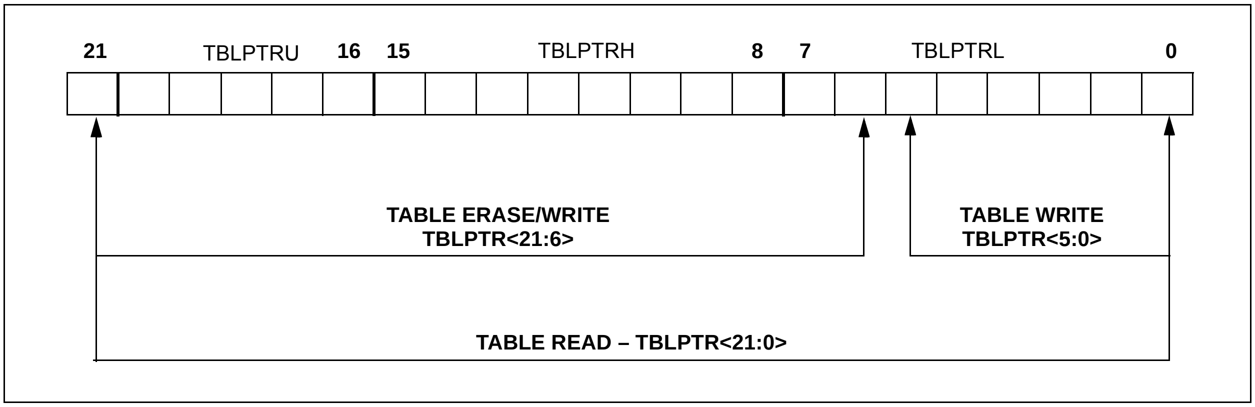
When an erase of program memory is executed, the 16 MSbs of the Table Pointer (TBLPTR<21:6>) point to the 64-byte block that will be erased. The Least Significant bits (TBLPTR<5:0>) are ignored.

Figure 8-3 describes the relevant boundaries of TBLPTR based on Flash program memory operations.

TABLE 8-1: TABLE POINTER OPERATIONS WITH TBLRD AND TBLWT INSTRUCTIONS

Example	Operation on Table Pointer
TBLRD* TBLWT*	TBLPTR is not modified
TBLRD*+ TBLWT*+	TBLPTR is incremented after the read/write
TBLRD*- TBLWT*-	TBLPTR is decremented after the read/write
TBLRD+* TBLWT+*	TBLPTR is incremented before the read/write

FIGURE 8-3: TABLE POINTER BOUNDARIES BASED ON OPERATION



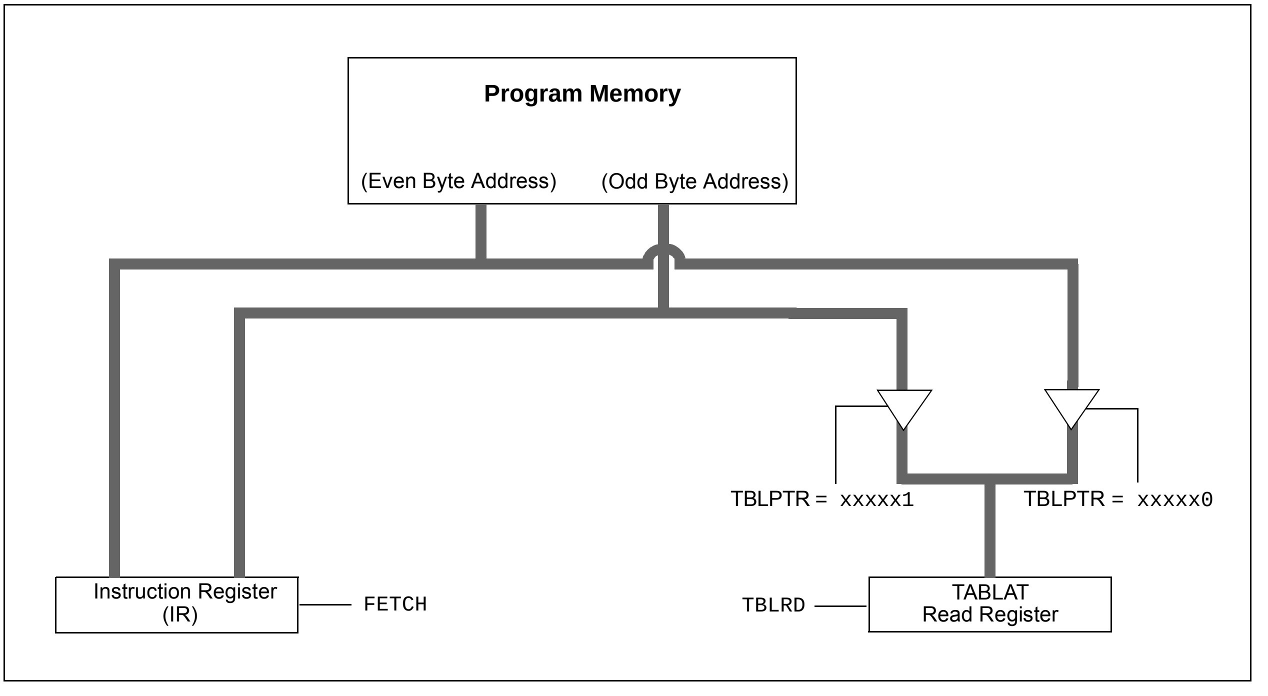
8.3 Reading the Flash Program Memory

The TBLRD instruction is used to retrieve data from program memory and place it into data RAM. Table reads from program memory are performed one byte at a time.

TBLPTR points to a byte address in program space. Executing a TBLRD instruction places the byte pointed to into TABLAT. In addition, TBLPTR can be modified automatically for the next table read operation.

The internal program memory is typically organized by words. The Least Significant bit of the address selects between the high and low bytes of the word. Figure 8-4 shows the interface between the internal program memory and the TABLAT.

FIGURE 8-4: READS FROM FLASH PROGRAM MEMORY



EXAMPLE 8-1: READING A FLASH PROGRAM MEMORY WORD

```
            MOVLW   CODE_ADDR_UPPER             ; Load TBLPTR with the base
            MOVWF   TBLPTRU                     ; address of the word
            MOVLW   CODE_ADDR_HIGH
            MOVWF   TBLPTRH
            MOVLW   CODE_ADDR_LOW
            MOVWF   TBLPTRL
READ_WORD
            TBLRD*+                             ; read into TABLAT and increment TBLPTR
            MOVF    TABLAT,W                    ; get data
            MOVWF   WORD_EVEN
            TBLRD*+                             ; read into TABLAT and increment TBLPTR
            MOVF    TABLAT,W                    ; get data
            MOVWF   WORD_ODD
```

8.4 Erasing Flash Program Memory

The minimum erase block is 32 words or 64 bytes. Larger blocks of program memory can be bulk erased only through the use of an external programmer or ICSP control. Word erase in the Flash array is not supported.

When initiating an erase sequence from the microcontroller itself, a block of 64 bytes of program memory is erased. The Most Significant 16 bits of the TBLPTR<21:6> point to the block being erased; TBLPTR<5:0> are ignored.

The EECON1 register commands the erase operation. The EEPGD bit (EECON1<7>) must be set to point to the Flash program memory. The WREN bit (EECON1<2>) must be set to enable write operations. The FREE bit (EECON1<4>) is set to select an erase operation.

For protection, the write initiate sequence using EECON2 must be used.

A long write is necessary for erasing the internal Flash. Instruction execution is halted while in a long write cycle. The long write will be terminated by the internal programming timer.

8.4.1 FLASH PROGRAM MEMORY ERASE SEQUENCE

The sequence of events for erasing a block of internal program memory location is:

1. Load the Table Pointer with the address of the row being erased.
2. Set the EECON1 register for the erase operation:
   - set the EEPGD bit to point to program memory;
   - clear the CFGS bit to access program memory;
   - set the WREN bit to enable writes;
   - set the FREE bit to enable the erase.
3. Disable interrupts.
4. Write 55h to EECON2.
5. Write 0AAh to EECON2.
6. Set the WR bit. This will begin the row erase cycle.
7. The CPU will stall for the duration of the erase (about 2 ms using internal timer).
8. Execute a `NOP`.
9. Re-enable interrupts.

EXAMPLE 8-2: ERASING A FLASH PROGRAM MEMORY ROW

```
                 MOVLW  CODE_ADDR_UPPER    ; load TBLPTR with the base
                 MOVWF  TBLPTRU            ; address of the memory block
                 MOVLW  CODE_ADDR_HIGH
                 MOVWF  TBLPTRH
                 MOVLW  CODE_ADDR_LOW
                 MOVWF  TBLPTRL
      ERASE_ROW
                 BSF    EECON1, EEPGD      ; point to Flash program memory
                 BCF    EECON1, CFGS       ; access Flash program memory
                 BSF    EECON1, WREN       ; enable write to memory
                 BSF    EECON1, FREE       ; enable Row Erase operation
                 BCF    INTCON, GIE        ; disable interrupts
                 MOVLW  55h
                 MOVWF  EECON2             ; write 55H
  Required       MOVLW  0AAh
  Sequence       MOVWF  EECON2             ; write 0AAH
                 BSF    EECON2, WR         ; start erase (CPU stall)
                 NOP
                 BSF    INTCON, GIE        ; re-enable interrupts
```

8.5 Writing to Flash Program Memory

The programming block size is 4 words or 8 bytes. Word or byte programming is not supported.

Table writes are used internally to load the holding registers needed to program the Flash memory. There are 8 holding registers used by the table writes for programming.

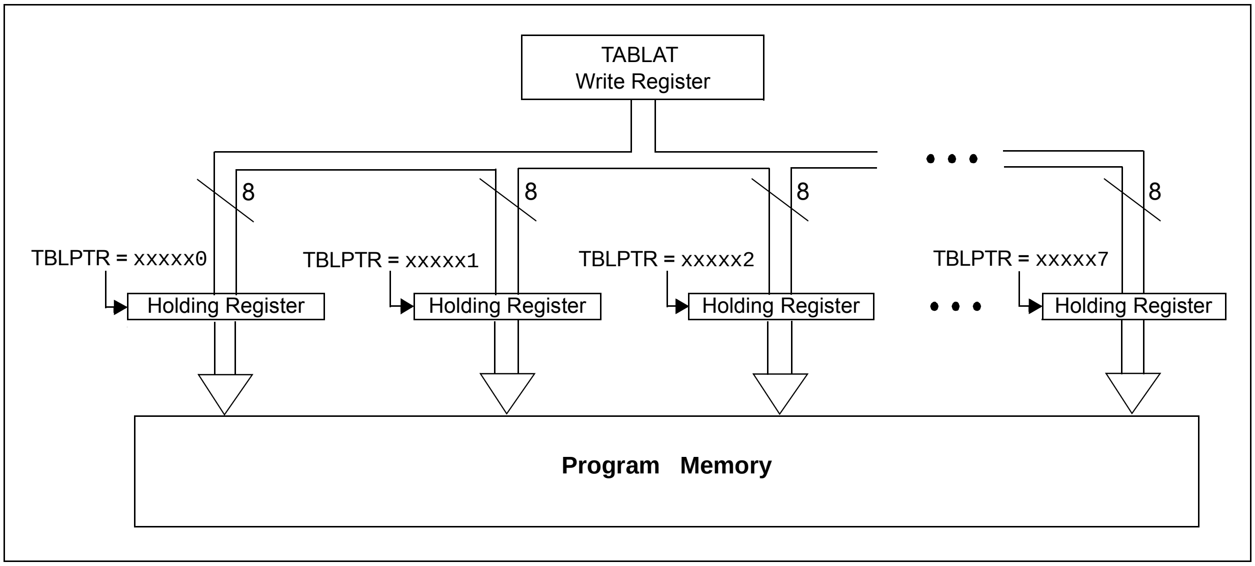
Since the Table Latch (TABLAT) is only a single byte, the `TBLWT` instruction has to be executed 8 times for each programming operation. All of the table write operations will essentially be short writes, because only the holding registers are written. At the end of updating 8 registers, the EECON1 register must be written to, to start the programming operation with a long write.

The long write is necessary for programming the internal Flash. Instruction execution is halted while in a long write cycle. The long write will be terminated by the internal programming timer.

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

Note:	The default value of the holding registers on device Resets and after write operations is FFh. A write of FFh to a holding register does not modify that byte. This means that individual bytes of program memory may be modified, provided that the modification does not attempt to change any bit from a ‘`0`’ to a ‘`1`’. When modifying individual bytes, it is not necessary to load all 64 holding registers before executing a write operation.

FIGURE 8-5: TABLE WRITES TO FLASH PROGRAM MEMORY



8.5.1 FLASH PROGRAM MEMORY WRITE SEQUENCE

The sequence of events for programming an internal program memory location should be:

1. Read 64 bytes into RAM.
2. Update data values in RAM as necessary.
3. Load Table Pointer with address being erased.
4. Do the row erase procedure (see **Section 8.4.1 "Flash Program Memory Erase Sequence"**).
5. Load Table Pointer with the address of the first byte being written.
6. Write the first 8 bytes into the holding registers with auto-increment.
7. Set the EECON1 register for the write operation by doing the following:
   - Set the EEPGD bit to point to program memory
   - Clear the CFGS bit to access program memory
   - Set the WREN bit to enable byte writes
8. Disable interrupts.
9. Write 55h to EECON2.
10. Write 0AAh to EECON2.
11. Set the WR bit. This will begin the write cycle.
12. The CPU will stall for the duration of the write (about 2 ms using internal timer).
13. Execute a `NOP`.
14. Re-enable interrupts.
15. Repeat Steps 6-14 seven times to write 64 bytes.
16. Verify the memory (table read).

This procedure will require about 18 ms to update one row of 64 bytes of memory. An example of the required code is given in Example 8-3.

EXAMPLE 8-3: WRITING TO FLASH PROGRAM MEMORY

```
            MOVLW   D'64'                     ; number of bytes in erase block
            MOVWF   COUNTER
            MOVLW   BUFFER_ADDR_HIGH          ; point to buffer
            MOVWF   FSR0H
            MOVLW   BUFFER_ADDR_LOW
            MOVWF   FSR0L
            MOVLW   CODE_ADDR_UPPER           ; Load TBLPTR with the base
            MOVWF   TBLPTRU                   ; address of the memory block
            MOVLW   CODE_ADDR_HIGH
            MOVWF   TBLPTRH
            MOVLW   CODE_ADDR_LOW             ; 6 LSB = 0
            MOVWF   TBLPTRL
READ_BLOCK
            TBLRD*+                           ; read into TABLAT, and inc
            MOVF    TABLAT,W                  ; get data
            MOVWF   POSTINC0                  ; store data and increment FSR0
            DECFSZ  COUNTER                   ; done?
            BRA     READ_BLOCK                ; repeat
MODIFY_WORD
            MOVLW   DATA_ADDR_HIGH            ; point to buffer
            MOVWF   FSR0H
            MOVLW   DATA_ADDR_LOW
            MOVWF   FSR0L
            MOVLW   NEW_DATA_LOW              ; update buffer word and increment FSR0
            MOVWF   POSTINC0
            MOVLW   NEW_DATA_HIGH             ; update buffer word
            MOVWF   INDF0
ERASE_BLOCK
            MOVLW   CODE_ADDR_UPPER           ; load TBLPTR with the base
            MOVWF   TBLPTRU                   ; address of the memory block
            MOVLW   CODE_ADDR_HIGH
            MOVWF   TBLPTRH
            MOVLW   CODE_ADDR_LOW             ; 6 LSB = 0
            MOVWF   TBLPTRL
            BCF     EECON1, CFGS              ; point to PROG/EEPROM memory
            BSF     EECON1, EEPGD             ; point to Flash program memory
            BSF     EECON1, WREN              ; enable write to memory
            BSF     EECON1, FREE              ; enable Row Erase operation
            BCF     INTCON, GIE               ; disable interrupts
            MOVLW   55h                       ; Required sequence
            MOVWF   EECON2                    ; write 55h
            MOVLW   0AAh
            MOVWF   EECON2                    ; write 0AAh
            BSF     EECON1, WR                ; start erase (CPU stall)
            NOP
            BSF     INTCON, GIE               ; re-enable interrupts
WRITE_BUFFER_BACK
            MOVLW   8                         ; number of write buffer groups of 8 bytes
            MOVWF   COUNTER_HI
            MOVLW   BUFFER_ADDR_HIGH          ; point to buffer
            MOVWF   FSR0H
            MOVLW   BUFFER_ADDR_LOW
            MOVWF   FSR0L
PROGRAM_LOOP
            MOVLW   8                         ; number of bytes in holding register
            MOVWF   COUNTER
WRITE_WORD_TO_HREGS
            MOVF    POSTINC0,F                ; get low byte of buffer data and increment FSR0
            MOVWF   TABLAT                    ; present data to table latch
            TBLWT+*                           ; short write
                                              ; to internal TBLWT holding register, increment
                                              ; TBLPTR
            DECFSZ  COUNTER                   ; loop until buffers are full
            GOTO    WRITE_WORD_TO_HREGS
```

EXAMPLE 8-3: WRITING TO FLASH PROGRAM MEMORY (CONTINUED)

```
PROGRAM_MEMORY
           BCF     INTCON, GIE           ; disable interrupts
           MOVLW   55h                   ; required sequence
           MOVWF   EECON2                ; write 55h
           MOVLW   0AAh
           MOVWF   EECON2                ; write 0AAh
           BSF     EECON1, WR            ; start program (CPU stall)
           NOP
           BSF     INTCON, GIE           ; re-enable interrupts
           DECFSZ  COUNTER_HI            ; loop until done
           GOTO    PROGRAM_LOOP
           BCF     EECON1, WREN          ; disable write to memory
```

8.5.2 WRITE VERIFY

Depending on the application, good programming practice may dictate that the value written to the memory should be verified against the original value. This should be used in applications where excessive writes can stress bits near the specification limit.

8.5.3 UNEXPECTED TERMINATION OF WRITE OPERATION

If a write is terminated by an unplanned event, such as loss of power or an unexpected Reset, the memory location just programmed should be verified and reprogrammed if needed. The WRERR bit is set when a write operation is interrupted by a MCLR Reset, or a WDT Time-out Reset during normal operation. In these situations, users can check the WRERR bit and rewrite the location.

8.6 Flash Program Operation During Code Protection

See **Section 23.5 "Program Verification and Code Protection"** for details on code protection of Flash program memory.

TABLE 8-2: REGISTERS ASSOCIATED WITH PROGRAM FLASH MEMORY

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
TBLPTRU	—	—	bit 21(1)	Program Memory Table Pointer Upper Byte (TBLPTR<20:16>)					54
TBPLTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)								54
TBLPTRL	Program Memory Table Pointer Low Byte (TBLPTR<7:0>)								54
TABLAT	Program Memory Table Latch								54
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
EECON2	EEPROM Control Register 2 (not a physical register)								56
EECON1	EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD	56
IPR2	OSCFIP	—	—	EEIP	—	LVDIP	—	CCP2IP	57
PIR2	OSCFIF	—	—	EEIF	—	LVDIF	—	CCP2IF	57
PIE2	OSCFIE	—	—	EEIE	—	LVDIE	—	CCP2IE	57

Legend: — = unimplemented, read as '0'. Shaded cells are not used during Flash/EEPROM access.

Note 1: Bit 21 of the PC is only available in Test mode and Serial Programming modes.

9.0 8 x 8 HARDWARE MULTIPLIER

9.1 Introduction

All PIC18 devices include an 8 x 8 hardware multiplier as part of the ALU. The multiplier performs an unsigned operation and yields a 16-bit result that is stored in the product register pair, PRODH:PRODL. The multiplier's operation does not affect any flags in the STATUS register.

Making multiplication a hardware operation allows it to be completed in a single instruction cycle. This has the advantages of higher computational throughput and reduced code size for multiplication algorithms, and allows the PIC18 devices to be used in many applications previously reserved for digital signal processors.

A comparison of various hardware and software multiply operations, along with the savings in memory and execution time, is shown in Table 9-1.

9.2 Operation

Example 9-1 shows the sequence to do an 8 x 8 unsigned multiply. Only one instruction is required when one argument of the multiply is already loaded in the WREG register.

Example 9-2 shows the sequence to do an 8 x 8 signed multiply. To account for the sign bits of the arguments, each argument's Most Significant bit (MSb) is tested and the appropriate subtractions are done.

EXAMPLE 9-1: 8 x 8 UNSIGNED MULTIPLY ROUTINE

```
MOVF    ARG1, W         ;
MULWF   ARG2            ; ARG1 * ARG2 ->
                        ; PRODH:PRODL
```

EXAMPLE 9-2: 8 x 8 SIGNED MULTIPLY ROUTINE

```
MOVF    ARG1, W
MULWF   ARG2            ; ARG1 * ARG2 ->
                        ; PRODH:PRODL
BTFSC   ARG2, SB        ; Test Sign Bit
SUBWF   PRODH, F        ; PRODH = PRODH
                        ;         - ARG1
MOVF    ARG2, W
BTFSC   ARG1, SB        ; Test Sign Bit
SUBWF   PRODH, F        ; PRODH = PRODH
                        ;         - ARG2
```

TABLE 9-1: PERFORMANCE COMPARISON

Routine	Multiply Method	Program Memory (Words)	Cycles (Max)	Time @ 40 MHz	Time @ 10 MHz	Time @ 4 MHz
8 x 8 Unsigned	Without Hardware Multiply	13	69	6.9 μs	27.6 μs	69 μs
	Hardware Multiply	1	1	100 ns	400 ns	1 μs
8 x 8 Signed	Without Hardware Multiply	33	91	9.1 μs	36.4 μs	91 μs
	Hardware Multiply	6	6	600 ns	2.4 μs	6 μs
16 x 16 Unsigned	Without Hardware Multiply	21	242	24.2 μs	96.8 μs	242 μs
	Hardware Multiply	24	24	2.4 μs	9.6 μs	24 μs
16 x 16 Signed	Without Hardware Multiply	52	254	25.4 μs	102.6 μs	254 μs
	Hardware Multiply	36	36	3.6 μs	14.4 μs	36 μs

Example 9-3 shows the sequence to do a 16 x 16 unsigned multiply. Equation 9-1 shows the algorithm that is used. The 32-bit result is stored in four registers, RES<3:0>.

EQUATION 9-1: 16 x 16 UNSIGNED MULTIPLICATION ALGORITHM

$$
\begin{aligned}
\text{RES<3:0>} &= \text{ARG1H:ARG1L} \bullet \text{ARG2H:ARG2L} \\
&= (\text{ARG1H} \bullet \text{ARG2H} \bullet 2^{16}) + \\
&\quad (\text{ARG1H} \bullet \text{ARG2L} \bullet 2^{8}) + \\
&\quad (\text{ARG1L} \bullet \text{ARG2H} \bullet 2^{8}) + \\
&\quad (\text{ARG1L} \bullet \text{ARG2L})
\end{aligned}
$$

EXAMPLE 9-3: 16 x 16 UNSIGNED MULTIPLY ROUTINE

```
    MOVF    ARG1L, W
    MULWF   ARG2L          ; ARG1L * ARG2L ->
                           ; PRODH:PRODL
    MOVFF   PRODH, RES1    ;
    MOVFF   PRODL, RES0    ;
;
    MOVF    ARG1H, W
    MULWF   ARG2H          ; ARG1H * ARG2H ->
                           ; PRODH:PRODL
    MOVFF   PRODH, RES3    ;
    MOVFF   PRODL, RES2    ;
;
    MOVF    ARG1L, W
    MULWF   ARG2H          ; ARG1L * ARG2H ->
                           ; PRODH:PRODL
    MOVF    PRODL, W       ;
    ADDWF   RES1, F        ; Add cross
    MOVF    PRODH, W       ; products
    ADDWFC  RES2, F        ;
    CLRF    WREG           ;
    ADDWFC  RES3, F        ;
;
    MOVF    ARG1H, W       ;
    MULWF   ARG2L          ; ARG1H * ARG2L ->
                           ; PRODH:PRODL
    MOVF    PRODL, W       ;
    ADDWF   RES1, F        ; Add cross
    MOVF    PRODH, W       ; products
    ADDWFC  RES2, F        ;
    CLRF    WREG           ;
    ADDWFC  RES3, F        ;
```

Example 9-4 shows the sequence to do a 16 x 16 signed multiply. Equation 9-2 shows the algorithm used. The 32-bit result is stored in four registers, RES<3:0>. To account for the sign bits of the arguments, each argument pair's Most Significant bit (MSb) is tested and the appropriate subtractions are done.

EQUATION 9-2: 16 x 16 SIGNED MULTIPLICATION ALGORITHM

$$
\begin{aligned}
&\text{RES<3:0>} \\
&\quad = \text{ARG1H:ARG1L} \bullet \text{ARG2H:ARG2L} \\
&\quad = (\text{ARG1H} \bullet \text{ARG2H} \bullet 2^{16}) + \\
&\qquad (\text{ARG1H} \bullet \text{ARG2L} \bullet 2^{8}) + \\
&\qquad (\text{ARG1L} \bullet \text{ARG2H} \text{ ² } 2^{8}) + \\
&\qquad (\text{ARG1L} \bullet \text{ARG2L}) + \\
&\qquad (-1 \bullet \text{ARG2H<7>} \bullet \text{ARG1H:ARG1L} \bullet 2^{16}) + \\
&\qquad (-1 \bullet \text{ARG1H<7>} \bullet \text{ARG2H:ARG2L} \bullet 2^{16})
\end{aligned}
$$

EXAMPLE 9-4: 16 x 16 SIGNED MULTIPLY ROUTINE

```
    MOVF    ARG1L, W
    MULWF   ARG2L          ; ARG1L * ARG2L ->
                           ; PRODH:PRODL
    MOVFF   PRODH, RES1    ;
    MOVFF   PRODL, RES0    ;
;
    MOVF    ARG1H, W
    MULWF   ARG2H          ; ARG1H * ARG2H ->
                           ; PRODH:PRODL
    MOVFF   PRODH, RES3    ;
    MOVFF   PRODL, RES2    ;
;
    MOVF    ARG1L,W
    MULWF   ARG2H          ; ARG1L * ARG2H ->
                           ; PRODH:PRODL
    MOVF    PRODL, W       ;
    ADDWF   RES1, F        ; Add cross
    MOVF    PRODH, W       ; products
    ADDWFC  RES2, F        ;
    CLRF    WREG           ;
    ADDWFC  RES3, F        ;
;
    MOVF    ARG1H, W       ;
    MULWF   ARG2L          ; ARG1H * ARG2L ->
                           ; PRODH:PRODL
    MOVF    PRODL, W       ;
    ADDWF   RES1, F        ; Add cross
    MOVF    PRODH, W       ; products
    ADDWFC  RES2, F        ;
    CLRF    WREG           ;
    ADDWFC  RES3, F        ;
;
    BTFSS   ARG2H, 7       ; ARG2H:ARG2L neg?
    BRA     SIGN_ARG1      ; no, check ARG1
    MOVF    ARG1L, W       ;
    SUBWF   RES2           ;
    MOVF    ARG1H, W       ;
    SUBWFB  RES3
;
SIGN_ARG1
    BTFSS   ARG1H, 7       ; ARG1H:ARG1L neg?
    BRA     CONT_CODE      ; no, done
    MOVF    ARG2L, W       ;
    SUBWF   RES2           ;
    MOVF    ARG2H, W       ;
    SUBWFB  RES3
;
CONT_CODE
    :
```

10.0 INTERRUPTS

The PIC18F2331/2431/4331/4431 devices have multiple interrupt sources and an interrupt priority feature that allows each interrupt source to be assigned a high-priority level or a low-priority level. The high-priority interrupt vector is at 000008h and the low-priority interrupt vector is at 000018h. High-priority interrupt events will interrupt any low-priority interrupts that may be in progress.

There are thirteen registers which are used to control interrupt operation. These registers are:

- RCON
- INTCON
- INTCON2
- INTCON3
- PIR1, PIR2, PIR3
- PIE1, PIE2, PIE3
- IPR1, IPR2, IPR3

It is recommended that the Microchip header files supplied with MPLAB® IDE be used for the symbolic bit names in these registers. This allows the assembler/compiler to automatically take care of the placement of these bits within the specified register.

In general, each interrupt source has three bits to control its operation. The functions of these bits are:

- Flag bit to indicate that an interrupt event occurred
- Enable bit that allows program execution to branch to the interrupt vector address when the flag bit is set
- Priority bit to select high priority or low priority (most interrupt sources have priority bits)

The interrupt priority feature is enabled by setting the IPEN bit (RCON<7>). When interrupt priority is enabled, there are two bits which enable interrupts globally. Setting the GIEH bit (INTCON<7>) enables all interrupts that have the priority bit set (high priority). Setting the GIEL bit (INTCON<6>) enables all interrupts that have the priority bit cleared (low priority). When the interrupt flag, enable bit and appropriate global interrupt enable bit are set, the interrupt will vector immediately to address 000008h or 000018h depending on the priority bit setting. Individual interrupts can be disabled through their corresponding enable bits.

When the IPEN bit is cleared (default state), the interrupt priority feature is disabled and interrupts are compatible with PIC® mid-range devices. In Compatibility mode, the interrupt priority bits for each source have no effect. INTCON<6> is the PEIE bit, which enables/disables all peripheral interrupt sources. INTCON<7> is the GIE bit, which enables/disables all interrupt sources. All interrupts branch to address 000008h in Compatibility mode.

When an interrupt is responded to, the global interrupt enable bit is cleared to disable further interrupts. If the IPEN bit is cleared, this is the GIE bit. If interrupt priority levels are used, this will be either the GIEH or GIEL bit. High-priority interrupt sources can interrupt a low-priority interrupt. Low-priority interrupts are not processed while high-priority interrupts are in progress.

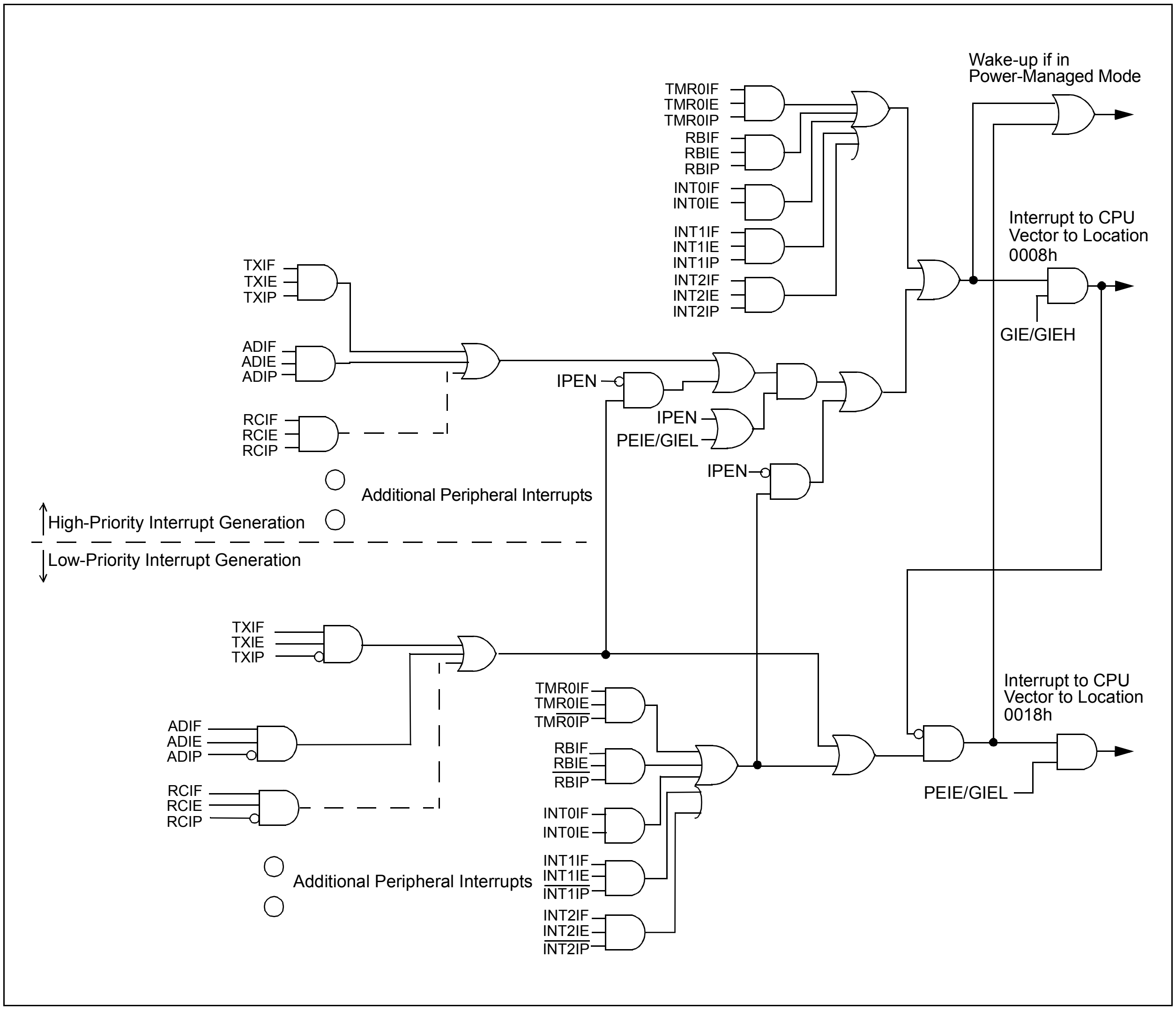
The return address is pushed onto the stack and the PC is loaded with the interrupt vector address (000008h or 000018h). Once in the Interrupt Service Routine, the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt flag bits must be cleared in software before re-enabling interrupts to avoid recursive interrupts.

The "return from interrupt" instruction, `RETFIE`, exits the interrupt routine and sets the GIE bit (GIEH or GIEL if priority levels are used), which re-enables interrupts.

For external interrupt events, such as the INTx pins or the PORTB input change interrupt, the interrupt latency will be three to four instruction cycles. The exact latency is the same for one or two-cycle instructions. Individual interrupt flag bits are set, regardless of the status of their corresponding enable bit or the GIE bit.

Note:	Do not use the `MOVFF` instruction to modify any of the Interrupt Control registers while **any** interrupt is enabled. Doing so may cause erratic microcontroller behavior.

FIGURE 10-1: INTERRUPT LOGIC



10.1 INTCON Registers

The INTCON registers are readable and writable registers which contain various enable, priority and flag 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. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. This feature allows for software polling.

REGISTER 10-1: INTCON: INTERRUPT CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **GIE/GIEH:** Global Interrupt Enable bit
When IPEN = 0:
1 = Enables all unmasked interrupts
0 = Disables all interrupts
When IPEN = 1:
1 = Enables all high-priority interrupts
0 = Disables all high-priority interrupts

bit 6 **PEIE/GIEL:** Peripheral Interrupt Enable bit
When IPEN = 0:
1 = Enables all unmasked peripheral interrupts
0 = Disables all peripheral interrupts
When IPEN = 1:
1 = Enables all low-priority peripheral interrupts
0 = Disables all low-priority peripheral interrupts

bit 5 **TMR0IE:** TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 overflow interrupt
0 = Disables the TMR0 overflow interrupt

bit 4 **INT0IE:** INT0 External Interrupt Enable bit
1 = Enables the INT0 external interrupt
0 = Disables the INT0 external interrupt

bit 3 **RBIE:** RB Port Change Interrupt Enable bit
1 = Enables the RB port change interrupt for RB<7:4> pins
0 = Disables the RB port change interrupt for RB<7:4> pins

bit 2 **TMR0IF:** TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software)
0 = TMR0 register did not overflow

bit 1 **INT0IF:** INT0 External Interrupt Flag bit
1 = The INT0 external interrupt occurred (must be cleared in software)
0 = The INT0 external interrupt did not occur

bit 0 **RBIF:** RB Port Change Interrupt Flag bit
1 = At least one of the RB<7:4> pins changed state (must be cleared in software)
0 = None of the RB<7:4> pins have changed state

REGISTER 10-2: INTCON2: INTERRUPT CONTROL REGISTER 2

R/W-1	R/W-1	R/W-1	R/W-1	U-0	R/W-1	U-0	R/W-1
$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	—	TMR0IP	—	RBIP
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **$\overline{\text{RBPU}}$:** PORTB Pull-up Enable bit
- 1 = All PORTB pull-ups are disabled
- 0 = PORTB pull-ups are enabled by individual port latch values

bit 6 **INTEDG0:** External Interrupt 0 Edge Select bit
- 1 = Interrupt on rising edge
- 0 = Interrupt on falling edge

bit 5 **INTEDG1:** External Interrupt 1 Edge Select bit
- 1 = Interrupt on rising edge
- 0 = Interrupt on falling edge

bit 4 **INTEDG2:** External Interrupt 2 Edge Select bit
- 1 = Interrupt on rising edge
- 0 = Interrupt on falling edge

bit 3 **Unimplemented:** Read as ‘0’

bit 2 **TMR0IP:** TMR0 Overflow Interrupt Priority bit
- 1 = High priority
- 0 = Low priority

bit 1 **Unimplemented:** Read as ‘0’

bit 0 **RBIP:** RB Port Change Interrupt Priority bit
- 1 = High priority
- 0 = Low priority

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. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. This feature allows for software polling.

REGISTER 10-3: INTCON3: INTERRUPT CONTROL REGISTER 3

R/W-1	R/W-1	U-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
INT2IP	INT1IP	—	INT2IE	INT1IE	—	INT2IF	INT1IF
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **INT2IP:** INT2 External Interrupt Priority bit
1 = High priority
0 = Low priority

bit 6 **INT1IP:** INT1 External Interrupt Priority bit
1 = High priority
0 = Low priority

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

bit 4 **INT2IE:** INT2 External Interrupt Enable bit
1 = Enables the INT2 external interrupt
0 = Disables the INT2 external interrupt

bit 3 **INT1IE:** INT1 External Interrupt Enable bit
1 = Enables the INT1 external interrupt
0 = Disables the INT1 external interrupt

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

bit 1 **INT2IF:** INT2 External Interrupt Flag bit
1 = The INT2 external interrupt occurred (must be cleared in software)
0 = The INT2 external interrupt did not occur

bit 0 **INT1IF:** INT1 External Interrupt Flag bit
1 = The INT1 external interrupt occurred (must be cleared in software)
0 = The INT1 external interrupt 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. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. This feature allows for software polling.

10.2 PIR Registers

The PIR registers contain the individual flag bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are three Peripheral Interrupt Request (Flag) Registers (PIR1, PIR2 and PIR3).

Note 1: Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the Global Interrupt Enable bit, GIE (INTCON<7>).

2: User software should ensure the appropriate interrupt flag bits are cleared prior to enabling an interrupt and after servicing that interrupt.

REGISTER 10-4: PIR1: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 1

U-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

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

bit 6 **ADIF:** A/D Converter Interrupt Flag bit
1 = An A/D conversion completed (must be cleared in software)
0 = The A/D conversion is not complete

bit 5 **RCIF:** EUSART Receive Interrupt Flag bit
1 = The EUSART receive buffer, RCREG, is full (cleared when RCREG is read)
0 = The EUSART receive buffer is empty

bit 4 **TXIF:** EUSART Transmit Interrupt Flag bit
1 = The EUSART transmit buffer, TXREG, is empty (cleared when TXREG is written)
0 = The EUSART transmit buffer is full

bit 3 **SSPIF:** Synchronous Serial Port Interrupt Flag bit
1 = The transmission/reception is complete (must be cleared in software)
0 = Waiting to transmit/receive

bit 2 **CCP1IF:** CCP1 Interrupt Flag bit
Capture mode:
1 = A TMR1 register capture occurred (must be cleared in software)
0 = No TMR1 register capture occurred
Compare mode:
1 = A TMR1 register compare match occurred (must be cleared in software)
0 = No TMR1 register compare match occurred
PWM mode:
Unused in this mode.

bit 1 **TMR2IF:** TMR2 to PR2 Match Interrupt Flag bit
1 = TMR2 to PR2 match occurred (must be cleared in software)
0 = No TMR2 to PR2 match occurred

bit 0 **TMR1IF:** TMR1 Overflow Interrupt Flag bit
1 = TMR1 register overflowed (must be cleared in software)
0 = TMR1 register did not overflow

REGISTER 10-5: PIR2: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 2

R/W-0	U-0	U-0	R/W-0	U-0	R/W-0	U-0	R/W-0
OSCFIF	—	—	EEIF	—	LVDIF	—	CCP2IF
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **OSCFIF:** Oscillator Fail Interrupt Flag bit
1 = Device oscillator failed, clock input has changed to INTOSC (must be cleared in software)
0 = Device clock operating

bit 6-5 **Unimplemented:** Read as ‘0’

bit 4 **EEIF:** EEPROM or Flash Write Operation Interrupt Flag bit
1 = The write operation is complete (must be cleared in software)
0 = The write operation is not complete or has not been started

bit 3 **Unimplemented:** Read as ‘0’

bit 2 **LVDIF:** Low-Voltage Detect Interrupt Flag bit
1 = The supply voltage has fallen below the specified LVD voltage (must be cleared in software)
0 = The supply voltage is greater than the specified LVD voltage

bit 1 **Unimplemented:** Read as ‘0’

bit 0 **CCP2IF:** CCP2 Interrupt Flag bit
Capture mode:
1 = A TMR1 register capture occurred (must be cleared in software)
0 = No TMR1 register capture occurred
Compare mode:
1 = A TMR1 register compare match occurred (must be cleared in software)
0 = No TMR1 register compare match occurred
PWM mode:
Not used in this mode.

REGISTER 10-6: PIR3: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 3

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	PTIF	IC3DRIF	IC2QEIF	IC1IF	TMR5IF
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

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

bit 4 **PTIF:** PWM Time Base Interrupt bit
1 = PWM time base matched the value in the PTPER registers. Interrupt is issued according to the postscaler settings. PTIF must be cleared in software.
0 = PWM time base has not matched the value in the PTPER registers

bit 3 **IC3DRIF:** IC3 Interrupt Flag/Direction Change Interrupt Flag bit
IC3 Enabled (CAP3CON<3:0>):
1 = TMR5 value was captured by the active edge on CAP3 input (must be cleared in software)
0 = TMR5 capture has not occurred
QEI Enabled (QEIM<2:0>):
1 = Direction of rotation has changed (must be cleared in software)
0 = Direction of rotation has not changed

bit 2 **IC2QEIF:** IC2 Interrupt Flag/QEI Interrupt Flag bit
IC2 Enabled (CAP2CON<3:0>):
1 = TMR5 value was captured by the active edge on CAP2 input (must be cleared in software)
0 = TMR5 capture has not occurred
QEI Enabled (QEIM<2:0>):
1 = The QEI position counter has reached the MAXCNT value, or the index pulse, INDX, has been detected. Depends on the QEI operating mode enabled. Must be cleared in software.
0 = The QEI position counter has not reached the MAXCNT value or the index pulse has not been detected

bit 1 IC1 Enabled (CAP1CON<3:0>):
1 = TMR5 value was captured by the active edge on CAP1 input (must be cleared in software)
0 = TMR5 capture has not occurred
QEI Enabled (QEIM<2:0>), Velocity Measurement Mode Enabled ($\overline{VELM}$ = 0 in QEICON register):
1 = Timer5 value was captured by the active velocity edge (based on PHA or PHB input). CAP1REN bit must be set in CAP1CON register. IC1IF must be cleared in software.
0 = Timer5 value was not captured by the active velocity edge

bit 0 **TMR5IF:** Timer5 Interrupt Flag bit
1 = Timer5 time base matched the PR5 value (must be cleared in software)
0 = Timer5 time base did not match the PR5 value

10.3 PIE Registers

The PIE registers contain the individual enable bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are three Peripheral Interrupt Enable Registers (PIE1, PIE2 and PIE3). When IPEN = 0, the PEIE bit must be set to enable any of these peripheral interrupts.

REGISTER 10-7: PIE1: PERIPHERAL INTERRUPT ENABLE REGISTER 1

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as ‘0’

bit 6 **ADIE:** A/D Converter Interrupt Enable bit
1 = Enables the A/D interrupt
0 = Disables the A/D interrupt

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

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

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

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

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

bit 0 **TMR1IE:** TMR1 Overflow Interrupt Enable bit
1 = Enables the TMR1 overflow interrupt
0 = Disables the TMR1 overflow interrupt

REGISTER 10-8: PIE2: PERIPHERAL INTERRUPT ENABLE REGISTER 2

R/W-0	U-0	U-0	R/W-0	U-0	R/W-0	U-0	R/W-0
OSCFIE	—	—	EEIE	—	LVDIE	—	CCP2IE
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **OSCFIE:** Oscillator Fail Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 6-5 **Unimplemented:** Read as '0'

bit 4 **EEIE:** Interrupt Enable bit
1 = Enabled
0 = Disabled

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

bit 2 **LVDIE:** Low-Voltage Detect Interrupt Enable bit
1 = Enabled
0 = Disabled

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

bit 0 **CCP2IE:** CCP2 Interrupt Enable bit
1 = Enabled
0 = Disabled

REGISTER 10-9: PIE3: PERIPHERAL INTERRUPT ENABLE REGISTER 3

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	PTIE	IC3DRIE	IC2QEIE	IC1IE	TMR5IE
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-5 **Unimplemented:** Read as ‘0’

bit 4 **PTIE:** PWM Time Base Interrupt Enable bit
1 = PTIF enabled
0 = PTIF disabled

bit 3 **IC3DRIE:** IC3 Interrupt Enable/Direction Change Interrupt Enable bit
<u>IC3 Enabled (CAP3CON<3:0>):</u>
1 = IC3 interrupt enabled
0 = IC3 interrupt disabled
<u>QEI Enabled (QEIM<2:0>):</u>
1 = Change of direction interrupt enabled
0 = Change of direction interrupt disabled

bit 2 **IC2QEIE:** IC2 Interrupt Flag/QEI Interrupt Flag Enable bit
<u>IC2 Enabled (CAP2CON<3:0>):</u>
1 = IC2 interrupt enabled)
0 = IC2 interrupt disabled
<u>QEI Enabled (QEIM<2:0>):</u>
1 = QEI interrupt enabled
0 = QEI interrupt disabled

bit 1 **IC1IE:** IC1 Interrupt Enable bit
1 = IC1 interrupt enabled
0 = IC1 interrupt disabled

bit 0 **TMR5IE:** Timer5 Interrupt Enable bit
1 = Timer5 interrupt enabled
0 = Timer5 interrupt disabled

10.4 IPR Registers

The IPR registers contain the individual priority bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are three peripheral interrupt priority registers (IPR1, IPR2 and IPR3). Using the priority bits requires that the Interrupt Priority Enable (IPEN) bit be set.

REGISTER 10-10: IPR1: PERIPHERAL INTERRUPT PRIORITY REGISTER 1

U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	ADIP	RCIP	TXIP	SSPIP	CCPIP	TMR2IP	TMR1IP
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as ‘0’

bit 6 **ADIP:** A/D Converter Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 **RC1IP:** EUSART Receive Interrupt Priority bit
1 = High priority
0 = Low priority

bit 4 **TX1IP:** EUSART Transmit Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **SSP1IP:** Synchronous Serial Port Interrupt Priority bit
1 = High priority
0 = Low priority

bit 2 **CCP1IP:** CCP1 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **TMR2IP:** TMR2 to PR2 Match Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **TMR1IP:** TMR1 Overflow Interrupt Priority bit
1 = High priority
0 = Low priority

REGISTER 10-11: IPR2: PERIPHERAL INTERRUPT PRIORITY REGISTER 2

R/W-1	U-0	U-0	R/W-1	U-0	R/W-1	U-0	R/W-1
OSCFIP	—	—	EEIP	—	LVDIP	—	CCP2IP
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **OSCFIP:** Oscillator Fail Interrupt Priority bit
1 = High priority
0 = Low priority

bit 6-5 **Unimplemented:** Read as ‘0’

bit 4 **EEIP:** Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **Unimplemented:** Read as ‘0’

bit 2 **LVDIP:** Low-Voltage Detect Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **Unimplemented:** Read as ‘0’

bit 0 **CCP2IP:** CCP2 Interrupt Priority bit
1 = High priority
0 = Low priority

REGISTER 10-12: IPR3: PERIPHERAL INTERRUPT PRIORITY REGISTER 3

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	PTIP	IC3DRIP	IC2QEIP	IC1IP	TMR5IP
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-5 **Unimplemented:** Read as ‘0’

bit 4 **PTIP:** PWM Time Base Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **IC3DRIP:** IC3 Interrupt Priority/Direction Change Interrupt Priority bit
IC3 Enabled (CAP3CON<3:0>):
1 = IC3 interrupt high priority
0 = IC3 interrupt low priority
QEI Enabled (QEIM<2:0>):
1 = Change of direction interrupt high priority
0 = Change of direction interrupt low priority

bit 2 **IC2QEIP:** IC2 Interrupt Priority/QEI Interrupt Priority bit
IC2 Enabled (CAP2CON<3:0>):
1 = IC2 interrupt high priority
0 = IC2 interrupt low priority
QEI Enabled (QEIM<2:0>):
1 = High priority
0 = Low priority

bit 1 **IC1IP:** IC1 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **TMR5IP:** Timer5 Interrupt Priority bit
1 = High priority
0 = Low priority

10.5 RCON Register

The RCON register contains bits used to determine the cause of the last Reset or wake-up from a power-managed mode. RCON also contains the bit that enables interrupt priorities (IPEN).

REGISTER 10-13: RCON: RESET CONTROL REGISTER

R/W-0	U-0	U-0	R/W-1	R-1	R-1	R/W-0	R/W-0
IPEN	—	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **IPEN:** Interrupt Priority Enable bit
1 = Enable priority levels on interrupts
0 = Disable priority levels on interrupts (PIC16CXXX Compatibility mode)

bit 6-5 **Unimplemented:** Read as ‘0’

bit 4 **$\overline{RI}$:** RESET Instruction Flag bit
For details of bit operation, see Register 5-1.

bit 3 **$\overline{TO}$:** Watchdog Timer Time-out Flag bit
For details of bit operation, see Register 5-1.

bit 2 **$\overline{PD}$:** Power-Down Detection Flag bit
For details of bit operation, see Register 5-1.

bit 1 **$\overline{POR}$:** Power-on Reset Status bit
For details of bit operation, see Register 5-1.

bit 0 **$\overline{BOR}$:** Brown-out Reset Status bit
For details of bit operation, see Register 5-1.

10.6 INTx Pin Interrupts

External interrupts on the INT0, INT1 and INT2 pins are edge-triggered. If the corresponding INTEDGx bit in the INTCON2 register is set (= 1), the interrupt is triggered by a rising edge. If the bit is clear, the trigger is on the falling edge.

When a valid edge appears on the INTx pin, the corresponding flag bit, INTxIF, is set. This interrupt can be disabled by clearing the corresponding enable bit, INTxIE. Before re-enabling the interrupt, the flag bit, INTxIF, must be cleared in software in the Interrupt Service Routine.

All external interrupts (INT0, INT1 and INT2) can wake-up the processor from the Idle or Sleep modes if bit, INTxIE, was set prior to going into those modes. If the Global Interrupt Enable bit, GIE, is set, the processor will branch to the interrupt vector following wake-up.

Interrupt priority for INT1 and INT2 is determined by the value contained in the Interrupt Priority bits, INT1IP (INTCON3<6>) and INT2IP (INTCON3<7>). There is no priority bit associated with INT0. It is always a high-priority interrupt source.

10.7 TMR0 Interrupt

In 8-bit mode (which is the default), an overflow (FFh → 00h) in the TMR0 register will set flag bit, TMR0IF. In 16-bit mode, an overflow (FFFFh → 0000h) in the TMR0H:TMR0L registers will set flag bit, TMR0IF. The interrupt can be enabled/disabled by setting/clearing enable bit, TMR0IE (INTCON<5>). Interrupt priority for Timer0 is determined by the value contained in the interrupt priority bit, TMR0IP (INTCON2<2>). See **Section 12.0 "Timer0 Module"** for further details.

10.8 PORTB Interrupt-on-Change

An input change on PORTB<7:4> sets flag bit, RBIF (INTCON<0>). The interrupt can be enabled/disabled by setting/clearing enable bit, RBIE (INTCON<3>). Interrupt priority for PORTB interrupt-on-change is determined by the value contained in the interrupt priority bit, RBIP (INTCON2<0>).

10.9 Context Saving During Interrupts

During interrupts, the return PC address is saved on the stack. Additionally, the WREG, STATUS and BSR registers are saved on the fast return stack. If a fast return from interrupt is not used (see **Section 6.1.3 "Fast Register Stack"**), the user may need to save the WREG, STATUS and BSR registers on entry to the Interrupt Service Routine. Depending on the user's application, other registers may also need to be saved. Example 10-1 saves and restores the WREG, STATUS and BSR registers during an Interrupt Service Routine.

EXAMPLE 10-1: SAVING STATUS, WREG AND BSR REGISTERS IN RAM

```
MOVWF   W_TEMP                  ; W_TEMP is in virtual bank
MOVFF   STATUS, STATUS_TEMP     ; STATUS_TEMP located anywhere
MOVFF   BSR, BSR_TEMP           ; BSR_TMEP located anywhere
;
; USER ISR CODE
;
MOVFF   BSR_TEMP, BSR           ; Restore BSR
MOVF    W_TEMP, W               ; Restore WREG
MOVFF   STATUS_TEMP, STATUS     ; Restore STATUS
```

11.0 I/O PORTS

Depending on the device selected and features enabled, there are up to five ports available. Some pins of the I/O ports are multiplexed with an alternate function from the peripheral features on the device. In general, when a peripheral is enabled, that pin may not be used as a general purpose I/O pin.

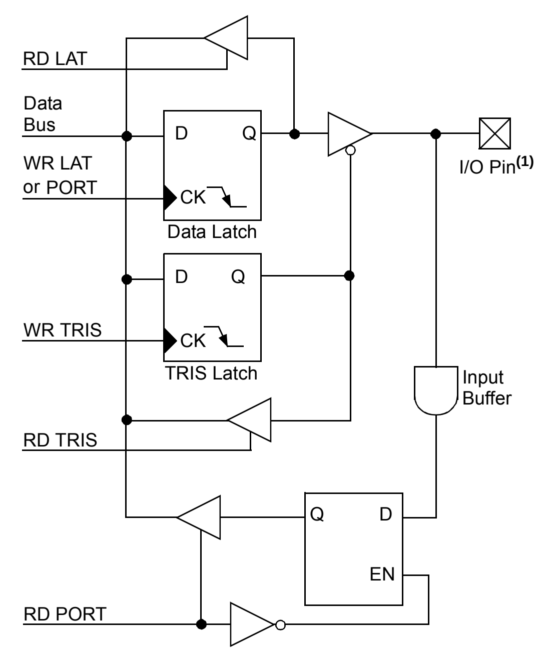
Each port has three registers for its operation. These registers are:

- TRIS register (Data Direction register)
- PORT register (reads the levels on the pins of the device)
- LAT register (Data Latch)

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

A simplified model of a generic I/O port without the interfaces to other peripherals is shown in Figure 11-1.

FIGURE 11-1: GENERIC I/O PORT OPERATION



Note 1: I/O pins have diode protection to VDD and VSS.

11.1 PORTA, TRISA and LATA Registers

PORTA is an 8-bit wide, bidirectional port. The corresponding data direction register is TRISA. Setting a TRISA bit (= 1) will make the corresponding PORTA pin an input (i.e., put the corresponding output driver in a high-impedance mode). Clearing a TRISA bit (= 0) will make the corresponding PORTA pin an output (i.e., put the contents of the output latch on the selected pin).

Reading the PORTA register reads the status of the pins, whereas writing to it, will write to the port latch.

The Data Latch register (LATA) is also memory mapped. Read-modify-write operations on the LATA register read and write the latched output value for PORTA.

The RA<4:2> pins are multiplexed with three input capture pins and Quadrature Encoder Interface pins. Pins, RA6 and RA7, are multiplexed with the main oscillator pins. They are enabled as oscillator or I/O pins by the selection of the main oscillator in Configuration Register 1H (see **Section 23.1 "Configuration Bits"** for details). When they are not used as port pins, RA6 and RA7 and their associated TRIS and LAT bits are read as '0'.

The other PORTA pins are multiplexed with analog inputs, the analog VREF+ and VREF- inputs and the comparator voltage reference output. The operation of pins RA<3:0> and RA5 as A/D Converter inputs is selected by clearing/setting the control bits in the ANSEL0 and ANSEL1 registers.

> **Note 1:** On a Power-on Reset, RA<5:0> are configured as analog inputs and read as '0'.
>
> **2:** RA5 I/F is available only on 40-pin devices (PIC18F4331/4431).

The TRISA register controls the direction of the RA pins, even when they are being used as analog inputs. The user must ensure the bits in the TRISA register are maintained set when using them as analog inputs.

EXAMPLE 11-1: INITIALIZING PORTA

```
CLRF    PORTA   ; Initialize PORTA by
                ; clearing output
                ; data latches
CLRF    LATA    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0x3F    ; Configure A/D
MOVWF   ANSEL0  ; for digital inputs
MOVLW   0xCF    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISA   ; Set RA<3:0> as inputs
                ; RA<5:4> as outputs
```

TABLE 11-1: PORTA I/O SUMMARY

Pin	Function	TRIS Setting	I/O	I/O Type	Description
RA0/AN0	RA0	0	O	DIG	LATA<0> data output; not affected by analog input.
		1	I	TTL	PORTA<0> data input; disabled when analog input is enabled.
	AN0	1	I	ANA	A/D Input Channel 0. Default input configuration on POR; does not affect digital output.
RA1/AN1	RA1	0	O	DIG	LATA<1> data output; not affected by analog input.
		1	I	TTL	PORTA<1> data input; disabled when analog input is enabled.
	AN1	1	I	ANA	A/D Input Channel 1. Default input configuration on POR; does not affect digital output.
RA2/AN2/VREF-/CAP1/INDX	RA2	0	O	DIG	LATA<2> data output; not affected by analog input.
		1	I	TTL	PORTA<2> data input. Disabled when analog input is enabled.
	AN2	1	I	ANA	A/D Input Channel 2. Default input configuration on POR.
	VREF-	1	I	ANA	A/D voltage reference low input.
	CAP1	1	I	ST	Input Capture Pin 1. Disabled when analog input is enabled.
	INDX	1	I	ST	Quadrature Encoder Interface index input pin. Disabled when analog input is enabled.
RA3/AN3/VREF+/CAP2/QEA	RA3	0	O	DIG	LATA<3> data output; not affected by analog input.
		1	I	TTL	PORTA<3> data input; disabled when analog input is enabled.
	AN3	1	I	ANA	A/D Input Channel 3. Default input configuration on POR.
	VREF+	1	I	ANA	A/D voltage reference high input.
	CAP2	1	I	ST	Input Capture Pin 2. Disabled when analog input is enabled.
	QEA	1	I	ST	Quadrature Encoder Interface Channel A input pin. Disabled when analog input is enabled.
RA4/AN4/CAP3/QEB	RA4	0	O	DIG	LATA<4> data output; not affected by analog input.
		1	I	ST	PORTA<4> data input; disabled when analog input is enabled.
	AN4	1	I	ANA	A/D Input Channel 4. Default input configuration on POR.
	CAP3	1	I	ST	Input Capture Pin 3. Disabled when analog input is enabled.
	QEB	1	I	ST	Quadrature Encoder Interface Channel B input pin. Disabled when analog input is enabled.
RA5/AN5/LVDIN	RA5	0	O	DIG	LATA<5> data output; not affected by analog input.
		1	I	TTL	PORTA<5> data input; disabled when analog input is enabled.
	AN5	1	I	ANA	A/D Input Channel 5. Default configuration on POR.
	LVDIN	1	I	ANA	Low-Voltage Detect external trip point input.
OSC2/CLKO/RA6	OSC2	x	O	ANA	Main oscillator feedback output connection (XT, HS and LP modes).
	CLKO	x	O	DIG	System cycle clock output (FOSC/4) in RC, INTIO1 and EC Oscillator modes.
	RA6	0	O	DIG	LATA<6> data output. Enabled in RCIO, INTIO2 and ECIO modes only.
		1	I	TTL	PORTA<6> data input. Enabled in RCIO, INTIO2 and ECIO modes only.
OSC1/CLKI/RA7	OSC1	x	I	ANA	Main oscillator input connection.
	CLKI	x	I	ANA	Main clock input connection.
	RA7	0	O	DIG	LATA<7> data output. Disabled in external oscillator modes.
		1	I	TTL	PORTA<7> data input. Disabled in external oscillator modes.

Legend: DIG = Digital level output; TTL = TTL input buffer; ST = Schmitt Trigger input buffer; ANA = Analog level input/output; x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

TABLE 11-2: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
PORTA	RA7(1)	RA6(1)	RA5	RA4	RA3	RA2	RA1	RA0	57
LATA	LATA7(1)	LATA6(1)	LATA Data Output Register						57
TRISA	TRISA7(1)	TRISA6(1)	PORTA Data Direction Register						57
ADCON1	VCFG1	VCFG0	—	FIFOEN	BFEMT	BFOVFL	ADPNT1	ADPNT0	56
ANSEL0	ANS7(2)	ANS6(2)	ANS5(2)	ANS4	ANS3	ANS2	ANS1	ANS0	56
ANSEL1	—	—	—	—	—	—	—	ANS8(2)	56

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

Note 1: RA<7:6> and their associated latch and data direction bits are enabled as I/O pins based on oscillator configuration; otherwise, they are read as ‘0’.

2: ANS5 through ANS8 are available only on the PIC18F4331/4431 devices.

11.2 PORTB, TRISB and LATB Registers

PORTB is an 8-bit wide, bidirectional port. The corresponding Data Direction register is TRISB. Setting a TRISB bit (= 1) will make the corresponding PORTB pin an input (i.e., put the corresponding output driver in a high-impedance mode). Clearing a TRISB bit (= 0) will make the corresponding PORTB pin an output (i.e., put the contents of the output latch on the selected pin).

The Data Latch register (LATB) is also memory mapped. Read-modify-write operations on the LATB register read and write the latched output value for PORTB.

EXAMPLE 11-2: INITIALIZING PORTB

```
CLRF    PORTB   ; Initialize PORTB by
                ; clearing output
                ; data latches
CLRF    LATB    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0xCF    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISB   ; Set RB<3:0> as inputs
                ; RB<5:4> as outputs
                ; RB<7:6> as inputs
```

Each of the PORTB pins has a weak internal pull-up. A single control bit can turn on all the pull-ups. This is performed by clearing bit $\overline{RBPU}$ (INTCON2<7>). The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on a Power-on Reset.

Four of the PORTB pins (RB<7:4>) have an interrupt-on-change feature. Only pins configured as inputs can cause this interrupt to occur (i.e., any RB<7:4> pin configured as an output is excluded from the interrupt-on-change comparison). The input pins (of RB<7:4>) are compared with the old value latched on the last read of PORTB. The "mismatch" outputs of RB<7:4> are ORed together to generate the RB port change interrupt with flag bit, RBIF (INTCON<0>).

This interrupt can wake the device from Sleep. The user, in the Interrupt Service Routine, can clear the interrupt in the following manner:

a) Any read or write of PORTB (except with the `MOVFF (ANY), PORTB` instruction).
b) `NOP` (or any 1 T_{CY} delay).
c) Clear flag bit, RBIF.

A mismatch condition will continue to set flag bit, RBIF. Reading PORTB and waiting 1 T_{CY} will end the mismatch condition and allow flag bit, RBIF, to be cleared. Also, if the port pin returns to its original state, the mismatch condition will be cleared.

The interrupt-on-change feature is recommended for wake-up on key depression operation and operations where PORTB is only used for the interrupt-on-change feature. Polling of PORTB is not recommended while using the interrupt-on-change feature.

RB<3:0> and RB4 pins are multiplexed with the 14-bit PWM module for PWM<3:0> and PWM5 output. The RB5 pin can be configured by the Configuration bit, PWM4MX, as the alternate pin for PWM4 output.

TABLE 11-3: PORTB I/O SUMMARY

Pin	Function	TRIS Setting	I/O	I/O Type	Description
RB0/PWM0	RB0	0	O	DIG	LATB<0> data output; not affected by analog input.
		1	I	TTL	PORTB<0> data input; weak pull-up when $\overline{RBPU}$ bit is cleared. Disabled when analog input is enabled.
	PWM0	0	O	DIG	PWM Output 0.
RB1/PWM1	RB1	0	O	DIG	LATB<1> data output; not affected by analog input.
		1	I	TTL	PORTB<1> data input; weak pull-up when $\overline{RBPU}$ bit is cleared. Disabled when analog input is enabled.
	PWM1	0	O	DIG	PWM Output 1.
RB2/PWM2	RB2	0	O	DIG	LATB<2> data output; not affected by analog input.
		1	I	TTL	PORTB<2> data input; weak pull-up when $\overline{RBPU}$ bit is cleared. Disabled when analog input is enabled.
	PWM2	0	O	DIG	PWM Output 2.
RB3/PWM3	RB3	0	O	DIG	LATB<3> data output; not affected by analog input.
		1	I	TTL	PORTB<3> data input; weak pull-up when $\overline{RBPU}$ bit is cleared. Disabled when analog input is enabled.
	PWM3	0	O	DIG	PWM Output 3.
RB4/KBI0/PWM5	RB4	0	O	DIG	LATB<4> data output; not affected by analog input.
		1	I	TTL	PORTB<4> data input; weak pull-up when $\overline{RBPU}$ bit is cleared. Disabled when analog input is enabled.
	KBI0	1	I	TTL	Interrupt-on-change pin.
	PWM5	0	O	DIG	PWM Output 5.
RB5/KBI1/ PWM4/PGM	RB5	0	O	DIG	LATB<5> data output.
		1	I	TTL	PORTB<5> data input; weak pull-up when $\overline{RBPU}$ bit is cleared.
	KBI1	1	I	TTL	Interrupt-on-change pin.
	PWM4(3)	0	O	DIG	PWM Output 4; takes priority over port data.
	PGM(2)	x	I	ST	Single-Supply Programming mode entry (ICSP™). Enabled by LVP Configuration bit; all other pin functions are disabled.
RB6/KBI2/PGC	RB6	0	O	DIG	LATB<6> data output.
		1	I	TTL	PORTB<6> data input; weak pull-up when $\overline{RBPU}$ bit is cleared.
	KBI2	1	I	TTL	Interrupt-on-change pin.
	PGC	x	I	ST	Serial execution (ICSP™) clock input for ICSP and ICD operation.(1)
RB7/KBI3/PGD	RB7	0	O	DIG	LATB<7> data output.
		1	I	TTL	PORTB<7> data input; weak pull-up when $\overline{RBPU}$ bit is cleared.
	KBI3	1	I	TTL	Interrupt-on-change pin.
	PGD	x	O	DIG	Serial execution data output for ICSP and ICD operation.(1)
		x	I	ST	Serial execution data input for ICSP and ICD operation.(1)

Legend: DIG = Digital level output; TTL = TTL input buffer; ST = Schmitt Trigger input buffer; ANA = Analog level input/output; x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

Note 1: All other pin functions are disabled when ICSP or ICD is enabled.
2: Single-Supply Programming must be enabled.
3: RD5 is the alternate pin for PWM4.

TABLE 11-4: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	57
LATB	LATB Data Output Register								57
TRISB	PORTB Data Direction Register								57
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	—	TMR0IP	—	RBIP	54
INTCON3	INT2IP	INT1IP	—	INT2IE	INT1IE	—	INT2IF	INT1IF	54

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

11.3 PORTC, TRISC and LATC Registers

PORTC is an 8-bit wide, bidirectional port. The corresponding Data Direction register is TRISC. 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., put the contents of the output latch on the selected pin).

The Data Latch register (LATC) is also memory mapped. Read-modify-write operations on the LATC register read and write the latched output value for PORTC.

PORTC is multiplexed with several peripheral functions (Table 11-5). The pins have Schmitt Trigger input buffers.

When enabling peripheral functions, care should be taken in defining TRIS bits for each PORTC pin. Some peripherals override the TRIS bit to make a pin an output, while other peripherals override the TRIS bit to make a pin an input. The user should refer to the corresponding peripheral section for the correct TRIS bit settings.

Note:	On a Power-on Reset, these pins are configured as digital inputs.

The contents of the TRISC register are affected by peripheral overrides. Reading TRISC always returns the current contents, even though a peripheral device may be overriding one or more of the pins.

External interrupts, IN0, INT1 and INT2, are placed on RC3, RC4 and RC5 pins, respectively.

SSP alternate interface pins, SDI/SDA, SCK/SCL and SDO are placed on RC4, RC5 and RC7 pins, respectively.

These pins are multiplexed on PORTC and PORTD by using the SSPMX bit in the CONFIG3L register.

EUSART pins RX/DT and TX/CK are placed on RC7 and RC6 pins, respectively.

The alternate Timer5 external clock input, T5CKI, and the alternate TMR0 external clock input, T0CKI, are placed on RC3 and are multiplexed with the PORTD (RD0) pin using the EXCLKMX Configuration bit in CONFIG3H. Fault inputs to the 14-bit PWM module, $\overline{FLTA}$ and $\overline{FLTB}$, are located on RC1 and RC2. $\overline{FLTA}$ input on RC1 is multiplexed with RD4 using the FLTAMX bit.

The contents of the TRISC register are affected by peripheral overrides. Reading TRISC always returns the current contents, even though a peripheral device may be overriding one or more of the pins.

EXAMPLE 11-3: INITIALIZING PORTC

```
CLRF    PORTC   ; Initialize PORTC by
                ; clearing output
                ; data latches
CLRF    LATC    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0xCF    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISC   ; Set RC<3:0> as inputs
                ; RC<5:4> as outputs
                ; RC<7:6> as inputs
```

TABLE 11-5: PORTC I/O SUMMARY

Pin	Function	TRIS Setting	I/O	I/O Type	Description
RC0/T1OSO/T1CKI	RC0	0	O	DIG	LATC<0> data output.
		1	I	ST	PORTC<0> data input.
	T1OSO	x	O	ANA	Timer1 oscillator output; enabled when Timer1 oscillator is enabled. Disables digital I/O.
	T1CKI	1	I	ST	Timer1/Timer3 counter input.
RC1/T1OSI/CCP2/$\overline{FLTA}$	RC1	0	O	DIG	LATC<1> data output.
		1	I	ST	PORTC<1> data input.
	T1OSI	x	I	ANA	Timer1 oscillator input; enabled when Timer1 oscillator is enabled. Disables digital I/O.
	CCP2	0	O	DIG	CCP2 compare and PWM output; takes priority over port data.
		1	I	ST	CCP2 capture input.
	$\overline{FLTA}$	1	I	ST	Fault Interrupt Input Pin A.
RC2/CCP1/$\overline{FLTB}$	RC2	0	O	DIG	LATC<2> data output.
		1	I	ST	PORTC<2> data input.
	CCP1	0	O	DIG	CCP1 compare or PWM output; takes priority over port data.
		1	I	ST	CCP1 capture input.
	$\overline{FLTB}$	1	I	ST	Fault Interrupt Input Pin B.
RC3/T0CKI/T5CKI/INT0	RC3	0	O	DIG	LATC<3> data output.
		1	I	ST	PORTC<3> data input.
	T0CKI(1)	1	I	ST	Timer0 alternate clock input.
	T5CKI(1)	1	I	ST	Timer5 alternate clock input.
	INT0	1	I	ST	External Interrupt 0.
RC4/INT1/SDI/SDA	RC4	0	O	DIG	LATC<4> data output.
		1	I	ST	PORTC<4> data input.
	INT1	1	I	ST	External Interrupt 1.
	SDI(1)	1	I	ST	SPI data input (SSP module).
	SDA(1)	0	O	DIG	I^2C™ data output (SSP module); takes priority over port data.
		1	I	I^2C	I^2C data input (SSP module).
RC5/INT2/SCK/SCL	RC5	0	O	DIG	LATC<5> data output.
		1	I	ST	PORTC<5> data input.
	INT2	1	I	ST	External Interrupt 2.
	SCK(1)	0	O	DIG	SPI clock output (SSP module); takes priority over port data.
		1	I	ST	SPI clock input (SSP module).
	SCL(1)	0	O	DIG	I^2C clock output (SSP module); takes priority over port data.
		1	I	I^2C	I^2C clock input (SSP module); input type depends on module setting.
RC6/TX/CK/$\overline{SS}$	RC6	0	O	DIG	LATC<6> data output.
		1	I	ST	PORTC<6> data input.
	TX	0	O	DIG	Asynchronous serial transmit data output (EUSART module); takes priority over port data. User must configure as an output.
	CK	0	O	DIG	Synchronous serial clock output (EUSART module); takes priority over port data.
		1	I	ST	Synchronous serial clock input (EUSART module).
	$\overline{SS}$	1	I	ST	SPI slave select input.

Legend: DIG = Digital level output; TTL = TTL input buffer; ST = Schmitt Trigger input buffer; ANA = Analog level input/output; x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

Note 1: RD0 is the alternate pin for T0CKI/T5CKI; RD2 is the alternate pin for SDI/SDA; RD3 is the alternate pin for SCK/SCL; RD1 is the alternate pin for SDO.

TABLE 11-5: PORTC I/O SUMMARY (CONTINUED)

Pin	Function	TRIS Setting	I/O	I/O Type	Description
RC7/RX/DT/SDO	RC7	0	O	DIG	LATC<7> data output.
		1	I	ST	PORTC<7> data input.
	RX	1	I	ST	Asynchronous serial receive data input (EUSART module).
	DT	0	O	DIG	Synchronous serial data output (EUSART module); takes priority over port data.
		1	I	ST	Synchronous serial data input (EUSART module). User must configure as an input.
	SDO(1)	0	O	DIG	SPI data out; takes priority over port data.

Legend: DIG = Digital level output; TTL = TTL input buffer; ST = Schmitt Trigger input buffer; ANA = Analog level input/output; x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

Note 1: RD0 is the alternate pin for T0CKI/T5CKI; RD2 is the alternate pin for SDI/SDA; RD3 is the alternate pin for SCK/SCL; RD1 is the alternate pin for SDO.

TABLE 11-6: SUMMARY OF REGISTERS ASSOCIATED WITH PORTC

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	57
LATC	LATC Data Output Register								57
TRISC	PORTC Data Direction Register								57
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	—	TMR0IP	—	RBIP	54
INTCON3	INT2IP	INT1IP	—	INT2IE	INT1IE	—	INT2IF	INT1IF	54

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

11.4 PORTD, TRISD and LATD Registers

Note:	PORTD is only available on PIC18F4331/4431 devices.

PORTD is an 8-bit wide, bidirectional port. The corresponding Data Direction register is TRISD. Setting a TRISD bit (= 1) will make the corresponding PORTD pin an input (i.e., put the corresponding output driver in a high-impedance mode). Clearing a TRISD bit (= 0) will make the corresponding PORTD pin an output (i.e., put the contents of the output latch on the selected pin).

The Data Latch register (LATD) is also memory mapped. Read-modify-write operations on the LATD register read and write the latched output value for PORTD.

All pins on PORTD are implemented with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output.

Note:	On a Power-on Reset, these pins are configured as digital inputs.

PORTD includes PWM<7:6> complementary fourth channel PWM outputs. PWM4 is the complementary output of PWM5 (the third channel), which is multiplexed with the RB5 pin. This output can be used as the alternate output using the PWM4MX Configuration bit in CONFIG3H when the Single-Supply Programming pin (PGM) is used on RB5.

RD1, RD2 and RD3 can be used as the alternate output for SDO, SDI/SDA and SCK/SCL using the SSPMX Configuration bit in CONFIG3H.

RD4 an be used as the alternate output for $\overline{\text{FLTA}}$ using the FLTAMX Configuration bit in CONFIG3H.

EXAMPLE 11-4: INITIALIZING PORTD

```
CLRF    PORTD   ; Initialize PORTD by
                ; clearing output
                ; data latches
CLRF    LATD    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0xCF    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISD   ; Set RD<3:0> as inputs
                ; RD<5:4> as outputs
                ; RD<7:6> as inputs
```

TABLE 11-7: PORTD I/O SUMMARY

Pin	Function	TRIS Setting	I/O	I/O Type	Description
RD0/T0CKI/T5CKI	RD0	0	O	DIG	LATD<0> data output.
		1	I	ST	PORTD<0> data input.
	T0CKI(1)	1	I	ST	Timer0 alternate clock input.
	T5CKI(1)	1	I	ST	Timer5 alternate clock input.
RD1/SDO	RD1	0	O	DIG	LATD<1> data output.
		1	I	ST	PORTD<1> data input.
	SDO(1)	0	O	DIG	SPI data out; takes priority over port data.
RD2/SDI/SDA	RD2	0	O	DIG	LATD<2> data output.
		1	I	ST	PORTD<2> data input.
	SDI(1)	1	I	ST	SPI data input (SSP module).
	SDA(1)	0	O	DIG	I^{2}C™ data output (SSP module); takes priority over port data.
		1	I	I^{2}C	I^{2}C data input (SSP module).
RD3/SCK/SCL	RD3	0	O	DIG	LATD<3> data output.
		1	I	ST	PORTD<3> data input.
	SCK(1)	0	O	DIG	SPI clock output (SSP module); takes priority over port data.
		1	I	ST	SPI clock input (SSP module).
	SCL(1)	0	O	DIG	I^{2}C clock output (SSP module); takes priority over port data.
		1	I	I^{2}C	I^{2}C clock input (SSP module); input type depends on module setting.
RD4/$\overline{\text{FLTA}}$	RD4	0	O	DIG	LATD<4> data output.
		1	I	ST	PORTD<4> data input.
	$\overline{\text{FLTA}}$(2)	1	I	ST	Fault Interrupt Input Pin A.
RD5/PWM4	RD5	0	O	DIG	LATD<5> data output.
		1	I	ST	PORTD<5> data input.
	PWM4(3)	0	O	DIG	PWM Output 4; takes priority over port data.
RD6/PWM6	RD6	0	O	DIG	LATD<6> data output.
		1	I	ST	PORTD<6> data input.
	PWM6	0	O	DIG	PWM Output 6; takes priority over port data.
RD7/PWM7	RD7	0	O	DIG	LATD<7> data output.
		1	I	ST	PORTD<7> data input.
	PWM7	0	O	DIG	PWM Output 7; takes priority over port data.

Legend: DIG = Digital level output; TTL = TTL input buffer; ST = Schmitt Trigger input buffer;
x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

Note 1: RC3 is the alternate pin for T0CKI/T5CKI; RC4 is the alternate pin for SDI/SDA; RC5 is the alternate pin for SCK/SCL; RC7 is the alternate pin for $\overline{\text{SDO}}$.

2: RC1 is the alternate pin for $\overline{\text{FLTA}}$.

3: RB5 is the alternate pin for PWM4.

TABLE 11-8: SUMMARY OF REGISTERS ASSOCIATED WITH PORTD

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
PORTD	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	57
LATD	LATD Data Output Register								57
TRISD	PORTD Data Direction Register								57

11.5 PORTE, TRISE and LATE Registers

Note:	PORTE is only available on PIC18F4331/4431 devices.

PORTE is a 4-bit wide, bidirectional port. Three pins (RE0/AN6, RE1/AN7 and RE2/AN8) are individually configurable as inputs or outputs. These pins have Schmitt Trigger input buffers. When selected as an analog input, these pins will read as ‘0’s.

The corresponding Data Direction register is TRISE. Setting a TRISE bit (= 1) will make the corresponding PORTE pin an input (i.e., put the corresponding output driver in a high-impedance mode). Clearing a TRISE bit (= 0) will make the corresponding PORTE pin an output (i.e., put the contents of the output latch on the selected pin).

TRISE controls the direction of the RE pins, even when they are being used as analog inputs. The user must make sure to keep the pins configured as inputs when using them as analog inputs.

Note:	On a Power-on Reset, RE<2:0> are configured as analog inputs.

The Data Latch register (LATE) is also memory mapped. Read-modify-write operations on the LATE register read and write the latched output value for PORTE.

The fourth pin of PORTE ($\overline{MCLR}$/VPP/RE3) is an input only pin available for PIC18F4331/4431 devices. Its operation is controlled by the MCLRE Configuration bit in Configuration Register 3H (CONFIG3H<7>). When selected as a port pin (MCLRE = 0), it functions as a digital input-only pin. As such, it does not have TRIS or LAT bits associated with its operation. Otherwise, it functions as the device’s master clear input. In either configuration, RE3 also functions as the programming voltage input during programming.

Note:	On a Power-on Reset, RE3 is enabled as a digital input only if Master Clear functionality is disabled.

EXAMPLE 11-5: INITIALIZING PORTE

```
CLRF    PORTE       ; Initialize PORTE by
                    ; clearing output
                    ; data latches
CLRF    LATE        ; Alternate method
                    ; to clear output
                    ; data latches
MOVLW   0x3F        ; Configure A/D
MOVWF   ANSEL0      ; for digital inputs
BCF     ANSEL1, 0   ;
MOVLW   0x03        ; Value used to
                    ; initialize data
                    ; direction
MOVWF   TRISE       ; Set RE<0> as input
                    ; RE<1> as output
                    ; RE<2> as input
```

11.5.1 PORTE IN 28-PIN DEVICES

For PIC18F2331/2431 devices, PORTE is not available. It is only available for PIC18F4331/4431 devices.

REGISTER 11-1: TRISE REGISTER

U-0	U-0	U-0	U-0	U-0	R/W-1	R/W-1	R/W-1
—	—	—	—	—	TRISE2	TRISE1	TRISE0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-3 **Unimplemented:** Read as ‘0’

bit 2 **TRISE2**: RE2 Direction Control bit
1 = Input
0 = Output

bit 1 **TRISE1**: RE1 Direction Control bit
1 = Input
0 = Output

bit 0 **TRISE0**: RE0 Direction Control bit
1 = Input
0 = Output

TABLE 11-9: PORTE I/O SUMMARY

Pin	Function	TRIS Setting	I/O	I/O Type	Description
RE0/AN6	RE0	0	O	DIG	LATE<0> data output; not affected by analog input.
		1	I	ST	PORTE<0> data input; disabled when analog input is enabled.
	AN6	1	I	ANA	A/D Input Channel 6. Default input configuration on POR.
RE1/AN7	RE1	0	O	DIG	LATE<1> data output; not affected by analog input.
		1	I	ST	PORTE<1> data input; disabled when analog input is enabled.
	AN7	1	I	ANA	A/D Input Channel 7. Default input configuration on POR.
RE2/AN8	RE2	0	O	DIG	LATE<2> data output; not affected by analog input.
		1	I	ST	PORTE<2> data input; disabled when analog input is enabled.
	AN8	1	I	ANA	A/D Input Channel 8. Default input configuration on POR.
$\overline{\text{MCLR}}$/VPP/RE3[1]	$\overline{\text{MCLR}}$	—	I	ST	External Master Clear input; enabled when MCLRE Configuration bit is set.
	VPP	—	I	ANA	High-Voltage Detection; used for ICSP™ mode entry detection. Always available, regardless of pin mode.
	RE3	—[2]	I	ST	PORTE<3> data input; enabled when MCLRE Configuration bit is clear.

Legend: DIG = Digital level output; TTL = TTL input buffer; ST = Schmitt Trigger input buffer; ANA = Analog level input/output; x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

Note 1: All PORTE pins are only implemented on 40/44-pin devices.

2: RE3 does not have a corresponding TRIS bit to control data direction.

TABLE 11-10: SUMMARY OF REGISTERS ASSOCIATED WITH PORTE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
PORTE	—	—	—	—	RE3[1]	RE2	RE1	RE0	57
LATE	—	—	—	—	—	LATE Data Output Register			57
TRISE	—	—	—	—	—	PORTE Data Direction Register			57
ANSEL0	ANS7[2]	ANS6[2]	ANS5[2]	ANS4	ANS3	ANS2	ANS1	ANS0	56
ANSEL1	—	—	—	—	—	—	—	ANS8[2]	56

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

Note 1: Implemented only when Master Clear functionality is disabled (CONFIG3H<7> = 0). It is available for PIC18F4331/4431 devices only.

2: ANS5 through ANS8 are available only on PIC18F4331/4431 devices.

NOTES:

12.0 TIMER0 MODULE

The Timer0 module has the following features:

- Software selectable as an 8-bit or 16-bit timer/counter
- Readable and writable
- Dedicated 8-bit software programmable prescaler
- Clock source selectable to be external or internal
- Interrupt-on-overflow from FFh to 00h in 8-bit mode and FFFFh to 0000h in 16-bit mode
- Edge select for external clock

Figure 12-1 shows a simplified block diagram of the Timer0 module in 8-bit mode and Figure 12-2 shows a simplified block diagram of the Timer0 module in 16-bit mode.

The T0CON register (Register 12-1) is a readable and writable register that controls all the aspects of Timer0, including the prescale selection.

REGISTER 12-1: T0CON: TIMER0 CONTROL REGISTER

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TMR0ON	T016BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **TMR0ON:** Timer0 On/Off Control bit
1 = Enables Timer0
0 = Stops Timer0

bit 6 **T016BIT:** Timer0 16-Bit Control bit
1 = Timer0 is configured as an 8-bit timer/counter
0 = Timer0 is configured as a 16-bit timer/counter

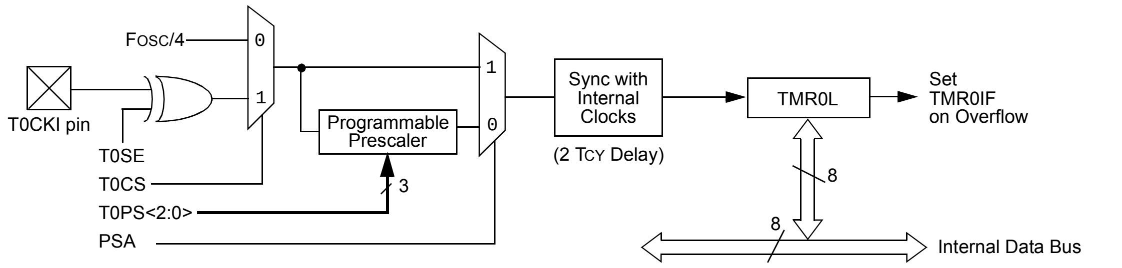
bit 5 **T0CS:** Timer0 Clock Source Select bit
1 = Transition on T0CKI pin input edge
0 = Internal clock (FOSC/4)

bit 4 **T0SE:** Timer0 Source Edge Select bit
1 = Increment on high-to-low transition on T0CKI pin
0 = Increment on low-to-high transition on T0CKI pin

bit 3 **PSA:** Timer0 Prescaler Assignment bit
1 = TImer0 prescaler is not assigned. Timer0 clock input bypasses prescaler.
0 = Timer0 prescaler is assigned. Timer0 clock input comes from prescaler output.

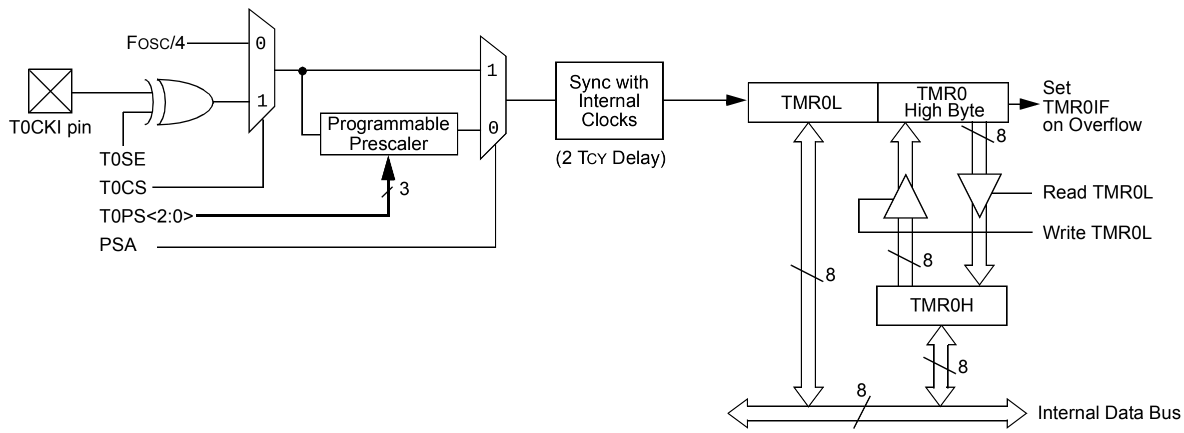
bit 2-0 **T0PS<2:0>:** Timer0 Prescaler Select bits
111 = 1:256 Prescale value
110 = 1:128 Prescale value
101 = 1:64 Prescale value
100 = 1:32 Prescale value
011 = 1:16 Prescale value
010 = 1:8 Prescale value
001 = 1:4 Prescale value
000 = 1:2 Prescale value

FIGURE 12-1: TIMER0 BLOCK DIAGRAM (8-BIT MODE)



Note: Upon Reset, Timer0 is enabled in 8-bit mode with clock input from T0CKI max. prescale.

FIGURE 12-2: TIMER0 BLOCK DIAGRAM (16-BIT MODE)



Note: Upon Reset, Timer0 is enabled in 8-bit mode with clock input from T0CKI max. prescale.

12.1 Timer0 Operation

Timer0 can operate as a timer or as a counter.

Timer mode is selected by clearing the T0CS bit. In Timer mode, the Timer0 module will increment every instruction cycle (without prescaler). If the TMR0 register is written, the increment is inhibited for the following two instruction cycles. The user can work around this by writing an adjusted value to the TMR0 register.

Counter mode is selected by setting the T0CS bit. In Counter mode, Timer0 will increment, either on every rising or falling edge of pin, RC3/T0CKI/T5CKI/INT0. The incrementing edge is determined by the Timer0 Source Edge Select bit (T0SE). Clearing the T0SE bit selects the rising edge.

When an external clock input is used for Timer0, it must meet certain requirements. The requirements ensure the external clock can be synchronized with the internal phase clock (TOSC). Also, there is a delay in the actual incrementing of Timer0 after synchronization.

12.2 Prescaler

An 8-bit counter is available as a prescaler for the Timer0 module. The prescaler is not readable or writable.

The PSA and T0PS<2:0> bits determine the prescaler assignment and prescale ratio.

Clearing bit, PSA, will assign the prescaler to the Timer0 module. When the prescaler is assigned to the Timer0 module, prescale values of 1:2, 1:4, ..., 1:256 are selectable.

When assigned to the Timer0 module, all instructions writing to the TMR0 register (e.g., `CLRF TMR0, MOVWF TMR0, BSF TMR0, x`..., etc.) will clear the prescaler count.

> **Note:** Writing to TMR0 when the prescaler is assigned to Timer0 will clear the prescaler count, but will not change the prescaler assignment.

12.2.1 SWITCHING PRESCALER ASSIGNMENT

The prescaler assignment is fully under software control (i.e., it can be changed "on-the-fly" during program execution).

12.3 Timer0 Interrupt

The TMR0 interrupt is generated when the TMR0 register overflows from FFh to 00h in 8-bit mode, or FFFFh to 0000h in 16-bit mode. This overflow sets the TMR0IF bit. The interrupt can be masked by clearing the TMR0IE bit. The TMR0IF bit must be cleared in software by the Timer0 module Interrupt Service Routine before re-enabling this interrupt. The TMR0 interrupt cannot awaken the processor from Sleep mode, since the timer requires clock cycles, even when T0CS is set.

12.4 16-Bit Mode Timer Reads and Writes

TMR0H is not the high byte of the timer/counter in 16-bit mode, but is actually a buffered version of the high byte of Timer0 (refer to Figure 12-2). The high byte of the Timer0 counter/timer is not directly readable nor writable. 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 were valid due to a rollover between successive reads of the high and low byte.

A write to the high byte of Timer0 must also take place through the TMR0H Buffer register. Timer0 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.

TABLE 12-1: REGISTERS ASSOCIATED WITH TIMER0

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
TMR0L	Timer0 Register Low Byte								55
TMR0H	Timer0 Register High Byte								55
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
T0CON	TMR0ON	T016BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	55
TRISA	TRISA7(1)	TRISA6(1)	PORTA Data Direction Register						57

Legend: Shaded cells are not used by Timer0.

Note 1: RA6 and RA7 are enabled as I/O pins depending on the oscillator mode selected in Configuration Word 1H.

NOTES:

13.0 TIMER1 MODULE

The Timer1 timer/counter module has the following features:

- 16-bit timer/counter (two 8-bit registers; TMR1H and TMR1L)
- Readable and writable (both registers)
- Internal or external clock select
- Interrupt-on-overflow from FFFFh to 0000h
- Reset from CCP module Special Event Trigger
- Status of system clock operation

Figure 13-1 is a simplified block diagram of the Timer1 module.

Register 13-1 details the Timer1 Control register. This register controls the operating mode of the Timer1 module and contains the Timer1 Oscillator Enable bit (T1OSCEN). Timer1 can be enabled or disabled by setting or clearing control bit, TMR1ON (T1CON<0>).

The Timer1 oscillator can be used as a secondary clock source in power-managed modes. When the T1RUN bit is set, the Timer1 oscillator provides the system clock. If the Fail-Safe Clock Monitor is enabled and the Timer1 oscillator fails while providing the system clock, polling the T1RUN bit will indicate whether the clock is being provided by the Timer1 oscillator or another source.

Timer1 can also be used to provide Real-Time Clock (RTC) functionality to applications with only a minimal addition of external components and code overhead.

REGISTER 13-1: T1CON: TIMER1 CONTROL REGISTER

R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **RD16:** 16-Bit Read/Write Mode Enable bit
1 = Enables register read/write of TImer1 in one 16-bit operation
0 = Enables register read/write of Timer1 in two 8-bit operations

bit 6 **T1RUN:** Timer1 System Clock Status bit
1 = Device clock is derived from Timer1 oscillator
0 = Device clock is derived from another source

bit 5-4 **T1CKPS<1:0>:** Timer1 Input Clock Prescale Select bits
11 = 1:8 Prescale value
10 = 1:4 Prescale value
01 = 1:2 Prescale value
00 = 1:1 Prescale value

bit 3 **T1OSCEN:** Timer1 Oscillator Enable bit
1 = Timer1 oscillator is enabled
0 = Timer1 oscillator is shut off
The oscillator inverter and feedback resistor are turned off to eliminate power drain.

bit 2 **$\overline{\text{T1SYNC}}$:** Timer1 External Clock Input Synchronization Select bit
When TMR1CS = 1 (External Clock):
1 = Do not synchronize external clock input
0 = Synchronize external clock input
When TMR1CS = 0 (Internal Clock):
This bit is ignored. Timer1 uses the internal clock when TMR1CS = 0.

bit 1 **TMR1CS:** Timer1 Clock Source Select bit
1 = External clock from pin RC0/T1OSO/T1CKI (on the rising edge)
0 = Internal clock (FOSC/4)

bit 0 **TMR1ON:** Timer1 On bit
1 = Enables Timer1
0 = Stops Timer1

13.1 Timer1 Operation

Timer1 can operate in one of these modes:

- As a timer
- As a synchronous counter
- As an asynchronous counter

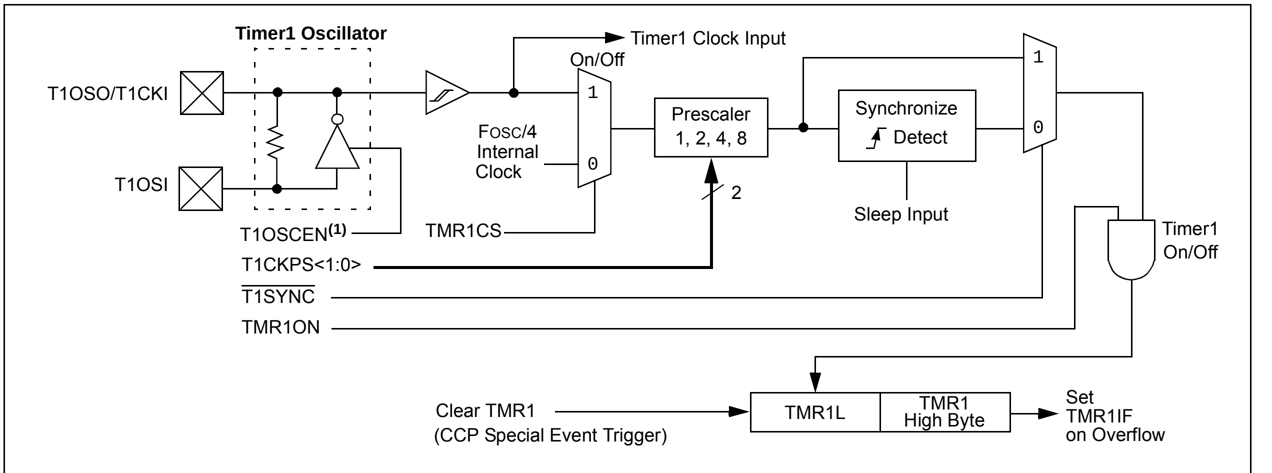
The operating mode is determined by the Timer1 Clock Select bit, TMR1CS (T1CON<1>).

When TMR1CS = 0, Timer1 increments every instruction cycle. When TMR1CS = 1, Timer1 increments on every rising edge of the external clock input or the Timer1 oscillator, if enabled.

When the Timer1 oscillator is enabled (T1OSCEN is set), the RC1/T1OSI/CCP2/FLTA and RC0/T1OSO/T1CKI pins become inputs. That is, the TRISC<1:0> value is ignored and the pins are read as '0'.

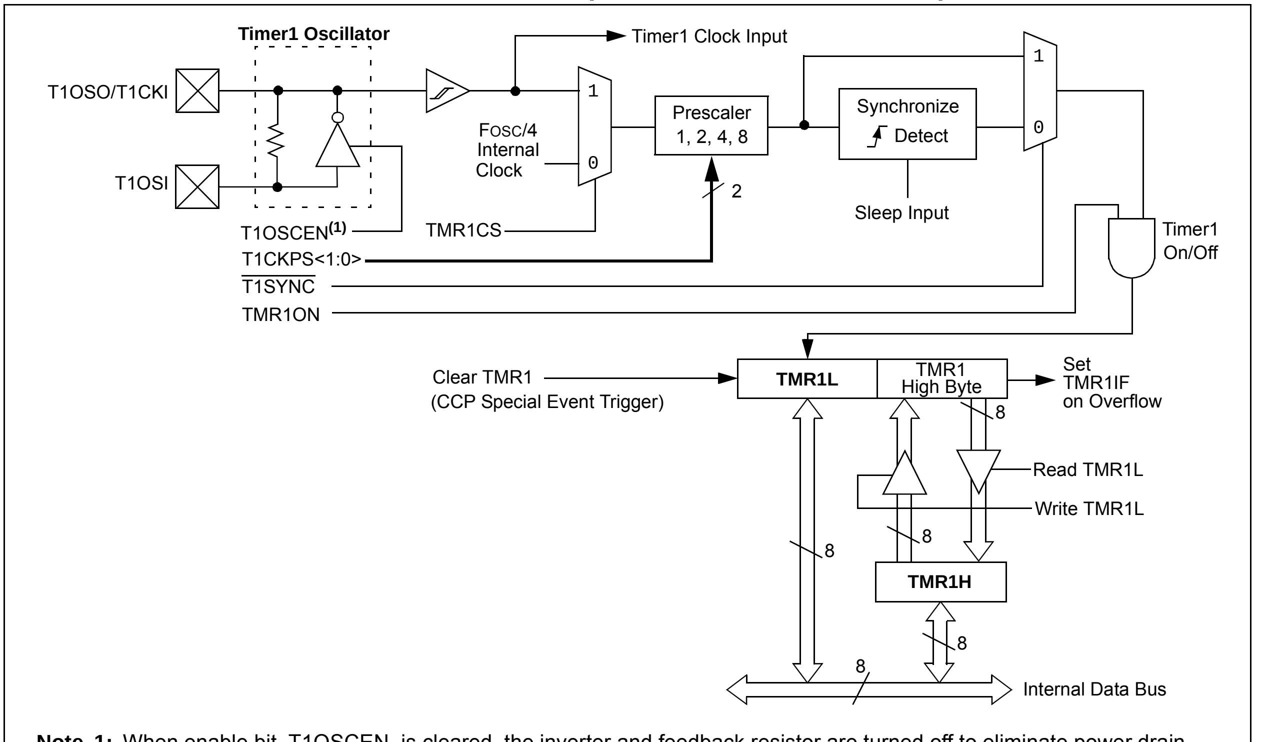
Timer1 also has an internal "Reset input". This Reset can be generated by the CCP module (see **Section 16.4.4 "Special Event Trigger"**).

FIGURE 13-1: TIMER1 BLOCK DIAGRAM



Note 1: When enable bit, T1OSCEN, is cleared, the inverter and feedback resistor are turned off to eliminate power drain.

FIGURE 13-2: TIMER1 BLOCK DIAGRAM (16-BIT READ/WRITE MODE)



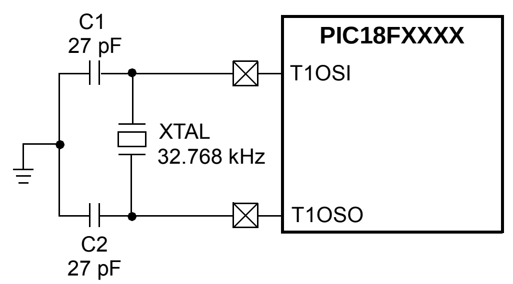
Note 1: When enable bit, T1OSCEN, is cleared, the inverter and feedback resistor are turned off to eliminate power drain.

13.2 Timer1 Oscillator

A crystal oscillator circuit is built in-between pins, T1OSI (input) and T1OSO (amplifier output). It is enabled by setting control bit, T1OSCEN (T1CON<3>). The oscillator is a low-power oscillator rated for 32 kHz crystals. It will continue to run during all power-managed modes. The circuit for a typical LP oscillator is shown in Figure 13-3. Table 13-1 shows the capacitor selection for the Timer1 oscillator.

The user must provide a software time delay to ensure proper start-up of the Timer1 oscillator.

FIGURE 13-3: EXTERNAL COMPONENTS FOR THE TIMER1 LP OSCILLATOR



Note: See the notes with Table 13-1 for additional information about capacitor selection.

TABLE 13-1: CAPACITOR SELECTION FOR THE TIMER OSCILLATOR

Osc Type	Freq	C1	C2
LP	32 kHz	27 pF(1)	27 pF(1)

Note 1: Microchip suggests this value as a starting point in validating the oscillator circuit.

2: Higher capacitance increases the stability of the oscillator, but also increases the start-up time.

3: Since each resonator/crystal has its own characteristics, the user should consult the resonator/crystal manufacturer for appropriate values of external components.

4: Capacitor values are for design guidance only.

13.3 Timer1 Oscillator Layout Considerations

The Timer1 oscillator for PIC18F2331/2431/4331/4431 devices incorporates an additional low-power feature. When this option is selected, it allows the oscillator to automatically reduce its power consumption when the microcontroller is in Sleep mode. During normal device operation, the oscillator draws full current. As high noise environments may cause excessive oscillator instability in Sleep mode, this option is best suited for low noise applications, where power conservation is an important design consideration.

The low-power option is enabled by clearing the $\overline{\text{T1OSCMX}}$ bit (CONFIG3L<5>). By default, the option is disabled, which results in a more or less constant current draw for the Timer1 oscillator.

Due to the low-power nature of the oscillator, it may also be sensitive to rapidly changing signals in close proximity.

The oscillator circuit, shown in Figure 13-3, should be located as close as possible to the microcontroller. There should be no circuits passing within the oscillator circuit boundaries other than VSS or VDD. Refer to **Section 2.0 "Guidelines for Getting Started with PIC18F Microcontrollers"** for additional information

13.4 Timer1 Interrupt

The TMR1 register pair (TMR1H:TMR1L) increments from 0000h to FFFFh and rolls over to 0000h. The Timer1 interrupt, if enabled, is generated on overflow, which is latched in Timer1 Interrupt Flag bit, TMR1IF (PIR1<0>). This interrupt can be enabled/disabled by setting/clearing Timer1 Interrupt Enable bit, TMR1IE (PIE1<0>).

13.5 Resetting Timer1 Using a CCP Trigger Output

If the CCP1 module is configured in Compare mode to generate a “Special Event Trigger” (CCP1M<3:0> = `1011`), this signal will reset Timer1 and start an A/D conversion if the A/D module is enabled (see **Section 16.4.4 “Special Event Trigger”** for more information).

Note:	The Special Event Triggers from the CCP1 module will not set interrupt flag bit, TMR1IF (PIR1<0>).

Timer1 must be configured for either Timer or Synchronized Counter mode to take advantage of this feature. If Timer1 is running in Asynchronous Counter mode, this Reset operation may not work.

In the event that a write to Timer1 coincides with a Special Event Trigger from CCP1, the write will take precedence.

In this mode of operation, the CCPR1H:CCPR1L register pair effectively becomes the Period register for Timer1.

13.6 Timer1 16-Bit Read/Write Mode

Timer1 can be configured for 16-bit reads and writes (see Figure 13-2). When the RD16 control bit (T1CON<7>) is set, the address for TMR1H is mapped to a buffer register for the high byte of Timer1. A read from TMR1L will load 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, is valid due to a rollover between reads.

A write to the high byte of Timer1 must also take place through the TMR1H Buffer register. 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 to both the high and low bytes of Timer1 at once.

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.

13.7 Using Timer1 as a Real-Time Clock (RTC)

Adding an external LP oscillator to Timer1 (such as the one described in **Section 13.2 “Timer1 Oscillator”**) gives users the option to include RTC functionality to their applications. This is accomplished with an inexpensive watch crystal to provide an accurate time base, and several lines of application code to calculate the time. When operating in Sleep mode and using a battery or supercapacitor as a power source, it can completely eliminate the need for a separate RTC device and battery backup.

The application code routine, `RTCisr`, shown in Example 13-1, demonstrates a simple method to increment a counter at one-second intervals using an Interrupt Service Routine. Incrementing the TMR1 register pair to overflow triggers the interrupt and calls the routine, which increments the seconds counter by one. Additional counters for minutes and hours are incremented as the previous counter overflow.

Since the register pair is 16 bits wide, counting up to overflow the register directly from a 32.768 kHz clock would take 2 seconds. To force the overflow at the required one-second intervals, it is necessary to preload it. The simplest method is to set the MSb of TMR1H with a `BSF` instruction. Note that the TMR1L register is never preloaded or altered; doing so may introduce cumulative error over many cycles.

For this method to be accurate, Timer1 must operate in Asynchronous mode and the Timer1 overflow interrupt must be enabled (PIE1<0> = `1`) as shown in the routine, `RTCinit`. The Timer1 oscillator must also be enabled and running at all times.

EXAMPLE 13-1: IMPLEMENTING A REAL-TIME CLOCK USING A TIMER1 INTERRUPT SERVICE

```
RTCinit
           MOVLW   0x80          ; Preload TMR1 register pair
           MOVWF   TMR1H         ; for 1 second overflow
           CLRF    TMR1L
           MOVLW   b'00001111'   ; Configure for external clock,
           MOVWF   T1CON         ; Asynchronous operation, external oscillator
           CLRF    secs          ; Initialize timekeeping registers
           CLRF    mins          ;
           MOVLW   .12
           MOVWF   hours
           BSF     PIE1, TMR1IE  ; Enable Timer1 interrupt
           RETURN
RTCisr
           BSF     TMR1H, 7      ; Preload for 1 sec overflow
           BCF     PIR1, TMR1IF  ; Clear interrupt flag
           INCF    secs, F       ; Increment seconds
           MOVLW   .59           ; 60 seconds elapsed?
           CPFSGT  secs
           RETURN                ; No, done
           CLRF    secs          ; Clear seconds
           INCF    mins, F       ; Increment minutes
           MOVLW   .59           ; 60 minutes elapsed?
           CPFSGT  mins
           RETURN                ; No, done
           CLRF    mins          ; clear minutes
           INCF    hours, F      ; Increment hours
           MOVLW   .23           ; 24 hours elapsed?
           CPFSGT  hours
           RETURN                ; No, done
           MOVLW   .01           ; Reset hours to 1
           MOVWF   hours
           RETURN                ; Done
```

TABLE 13-2: REGISTERS ASSOCIATED WITH TIMER1 AS A TIMER/COUNTER

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
PIR1	—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	57
PIE1	—	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	57
IPR1	—	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	57
TMR1L	Timer1 Register Low Byte								55
TMR1H	Timer1 Register High Byte								55
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	55

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

14.0 TIMER2 MODULE

The Timer2 module has the following features:

- 8-bit Timer register (TMR2)
- 8-bit Period register (PR2)
- Readable and writable (both registers)
- Software programmable prescaler (1:1, 1:4, 1:16)
- Software programmable postscaler (1:1 to 1:16)
- Interrupt on TMR2 match with PR2
- SSP module optional use of TMR2 output to generate clock shift

Timer2 has a control register, shown in Register 14-1. TMR2 can be shut off by clearing control bit, TMR2ON (T2CON<2>), to minimize power consumption. Figure 14-1 is a simplified block diagram of the Timer2 module. Register 14-1 shows the Timer2 Control register. The prescaler and postscaler selection of Timer2 are controlled by this register.

14.1 Timer2 Operation

Timer2 can be used as the PWM time base for the PWM mode of the CCP module. The TMR2 register is readable and writable, and is cleared on any device Reset. The input clock (FOSC/4) has a prescale option of 1:1, 1:4 or 1:16, selected by control bits, T2CKPS<1:0> (T2CON<1:0>). The match output of TMR2 goes through a 4-bit postscaler (which gives a 1:1 to 1:16 scaling inclusive) to generate a TMR2 interrupt, latched in flag bit, TMR2IF (PIR1<1>).

The TMR2 and PR2 registers are both directly readable and writable. The TMR2 register is cleared on any device Reset, while the PR2 register initializes at FFh.

The prescaler and postscaler counters are cleared when any of the following occurs:

- A write to the TMR2 register
- A write to the T2CON register
- Any device Reset (Power-on Reset, $\overline{\text{MCLR}}$ Reset, Watchdog Timer Reset or Brown-out Reset)

TMR2 is not cleared when T2CON is written.

REGISTER 14-1: T2CON: TIMER2 CONTROL REGISTER

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as ‘0’

bit 6-3 **TOUTPS<3:0>:** Timer2 Output Postscale Select bits
`0000` = 1:1 Postscale
`0001` = 1:2 Postscale
•
•
•
`1111` = 1:16 Postscale

bit 2 **TMR2ON:** Timer2 On bit
`1` = Timer2 is on
`0` = Timer2 is off

bit 1-0 **T2CKPS<1:0>:** Timer2 Clock Prescale Select bits
`00` = Prescaler is 1
`01` = Prescaler is 4
`1x` = Prescaler is 16

14.2 Timer2 Interrupt

Timer2 can also generate an optional device interrupt. The Timer2 output signal (TMR2 to PR2 match) provides the input for the 4-bit output counter/postscaler. This counter generates the TMR2 match interrupt flag which is latched in TMR2IF (PIR1<1>).

The interrupt is enabled by setting the TMR2 Match Interrupt Enable bit, TMR2IE (PIE1<1>). A range of 16 postscale options (from 1:1 through 1:16 inclusive) can be selected with the postscaler control bits, T2OUTPS<3:0> (T2CON<6:3>).

14.3 Output of TMR2

The unscaled output of TMR2 is available primarily to the CCP modules, where it is used as a time base for operations in PWM mode. Timer2 can be optionally used as the shift clock source for the SSP module operating in SPI mode.

For additional information, see **Section 19.0 "Synchronous Serial Port (SSP) Module"**.

FIGURE 14-1: TIMER2 BLOCK DIAGRAM

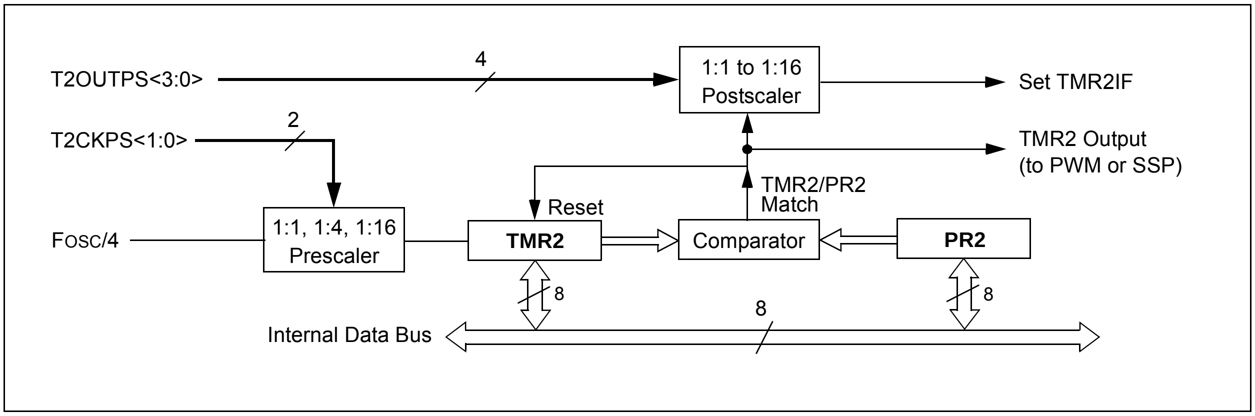


TABLE 14-1: REGISTERS ASSOCIATED WITH TIMER2 AS A TIMER/COUNTER

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
PIR1	—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	57
PIE1	—	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	57
IPR1	—	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	57
TMR2	Timer2 Register								55
T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	55
PR2	Timer2 Period Register								55

Legend: — = unimplemented, read as '0'. Shaded cells are not used by the Timer2 module.

NOTES:

15.0 TIMER5 MODULE

The Timer5 module implements these features:

- 16-bit timer/counter operation
- Synchronous and Asynchronous Counter modes
- Continuous Count and Single-Shot Operating modes
- Four programmable prescaler values (1:1 to 1:8)
- Interrupt generated on period match
- Special Event Trigger Reset function
- Double-buffered registers
- Operation during Sleep
- CPU wake-up from Sleep
- Selectable hardware Reset input with a wake-up feature

Timer5 is a general purpose timer/counter that incorporates additional features for use with the Motion Feedback Module (see **Section 17.0 "Motion Feedback Module"**). It may also be used as a general purpose timer or a Special Event Trigger delay timer. When used as a general purpose timer, it can be configured to generate a delayed Special Event Trigger (e.g., an ADC Special Event Trigger) using a preprogrammed period delay.

Timer5 is controlled through the Timer5 Control register (T5CON), shown in Register 15-1. The timer can be enabled or disabled by setting or clearing the control bit TMR5ON (T5CON<0>).

A block diagram of Timer5 is shown in Figure 15-1.

REGISTER 15-1: T5CON: TIMER5 CONTROL 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
T5SEN	$\overline{\text{RESEN}}$(1)	T5MOD	T5PS1	T5PS0	$\overline{\text{T5SYNC}}$(2)	TMR5CS	TMR5ON
bit 7							bit 0

Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **T5SEN:** Timer5 Sleep Enable bit
1 = Timer5 is enabled during Sleep
0 = Timer5 is disabled during Sleep

bit 6 **$\overline{\text{RESEN}}$:** Special Event Trigger Reset Enable bit(1)
1 = Special Event Trigger Reset is disabled
0 = Special Event Trigger Reset is enabled

bit 5 **T5MOD:** Timer5 Mode bit
1 = Single-Shot mode is enabled
0 = Continuous Count mode is enabled

bit 4-3 **T5PS<1:0>:** Timer5 Input Clock Prescale Select bits
11 = 1:8
10 = 1:4
01 = 1:2
00 = 1:1

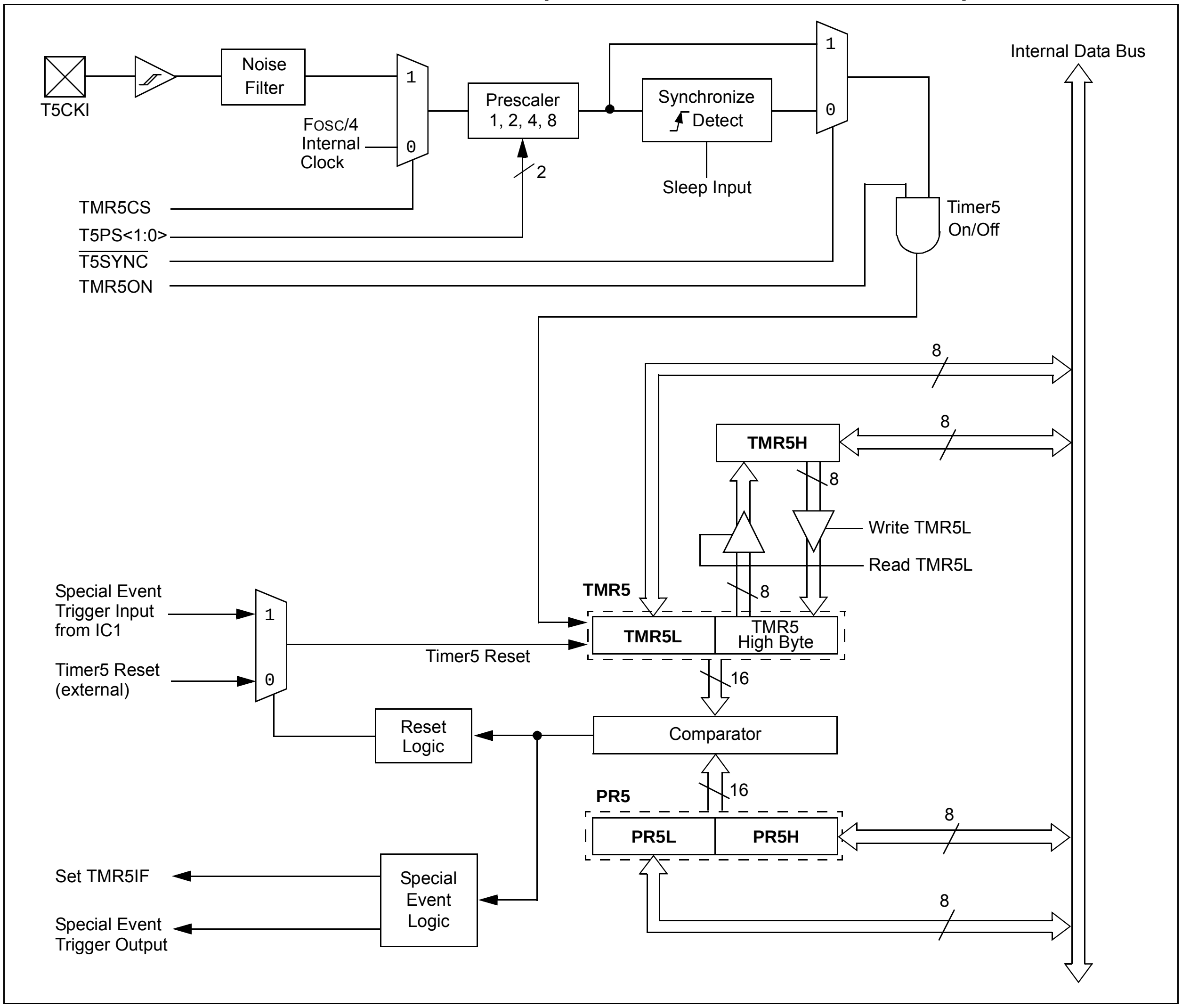
bit 2 **$\overline{\text{T5SYNC}}$:** Timer5 External Clock Input Synchronization Select bit(2)
When TMR5CS = 1:
1 = Do not synchronize external clock input
0 = Synchronize external clock input
When TMR5CS = 0:
This bit is ignored. Timer5 uses the internal clock when TMR5CS = 0.

bit 1 **TMR5CS:** Timer5 Clock Source Select bit
1 = External clock from the T5CKI pin
0 = Internal clock (T_{CY})

bit 0 **TMR5ON:** Timer5 On bit
1 = Timer5 is enabled
0 = Timer5 is disabled

Note 1: These bits are not implemented on PIC18F2331/2431 devices and read as '0'.
2: For Timer5 to operate during Sleep mode, $\overline{\text{T5SYNC}}$ must be set.

FIGURE 15-1: TIMER5 BLOCK DIAGRAM (16-BIT READ/WRITE MODE SHOWN)



15.1 Timer5 Operation

Timer5 combines two 8-bit registers to function as a 16-bit timer. The TMR5L register is the actual low byte of the timer; it can be read and written to directly. The high byte is contained in an unmapped register; it is read and written to through TMR5H, which serves as a buffer. Each register increments from 00h to FFh.

A second register pair, PR5H and PR5L, serves as the Period register; it sets the maximum count for the TMR5 register pair. When TMR5 reaches the value of PR5, the timer rolls over to 00h and sets the TMR5IF interrupt flag. A simplified block diagram of the Timer5 module is shown in Figure 2-1.

Note:	The Timer5 may be used as a general purpose timer and as the time base resource to the Motion Feedback Module (Input Capture or Quadrature Encoder Interface).

Timer5 supports three configurations:

- 16-Bit Synchronous Timer
- 16-Bit Synchronous Counter
- 16-Bit Asynchronous Counter

In Synchronous Timer configuration, the timer is clocked by the internal device clock. The optional Timer5 prescaler divides the input by 2, 4, 8 or not at all (1:1). The TMR5 register pair increments on Q1. Clearing TMR5CS (= `0`) selects the internal device clock as the timer sampling clock.

In Synchronous Counter mode configuration, the timer is clocked by the external clock (T5CKI) with the optional prescaler. The external T5CKI is selected by setting the TMR5CS bit (TMR5CS = `1`); the internal clock is selected by clearing TMR5CS. The external clock is synchronized to the internal clock by clearing the $\overline{\text{T5SYNC}}$ bit. The input on T5CKI is sampled on every Q2 and Q4 of the internal clock. The low to rise transition is decoded on three adjacent samples and the Timer5 is incremented on the next Q1. The T5CKI minimum pulse-width high and low time must be greater than TCY/2.

In Asynchronous Counter mode configuration, Timer5 is clocked by the external clock (T5CKI) with the optional prescaler. In this mode, T5CKI is not synchronized to the internal clock. By setting TMR5CS, the external input clock (T5CKI) can be used as the counter sampling clock. When $\overline{\text{T5SYNC}}$ is set, the external clock is not synchronized to the internal device clock.

The timer count is not reset automatically when the module is disabled. The user may write the Counter register to initialize the counter.

Note:	The Timer5 module does NOT prevent writes to the PR5 registers (PR5H:PR5L) while the timer is enabled. Writing to PR5 while the timer is enabled may result in unexpected period match events.

15.1.1 CONTINUOUS COUNT AND SINGLE-SHOT OPERATION

Timer5 has two operating modes: Continuous Count and Single-Shot.

Continuous Count mode is selected by clearing the T5MOD control bit (= `0`). In this mode, the Timer5 time base will start incrementing according to the prescaler settings until a TMR5/PR5 match occurs, or until TMR5 rolls over (FFFFh to 0000h). The TMR5IF interrupt flag is set, the TMR5 register is reset on the following input clock edge and the timer continues to count for as long as the TMR5ON bit remains set.

Single-Shot mode is selected by setting T5MOD (= `1`). In this mode, the Timer5 time base begins to increment according to the prescaler settings until a TMR5/PR5 match occurs. This causes the TMR5IF interrupt flag to be set, the TMR5 register pair to be cleared on the following input clock edge and the TMR5ON bit to be cleared by the hardware to halt the timer.

The Timer5 time base can only start incrementing in Single-Shot mode under two conditions:

1. Timer5 is enabled (TMR5ON is set), or
2. Timer5 is disabled and a Special Event Trigger Reset is present on the Timer5 Reset input. (See **Section 15.7 "Timer5 Special Event Trigger Reset Input"** for additional information.)

15.2 16-Bit Read/Write and Write Modes

As noted, the actual high byte of the Timer5 register pair is mapped to TMR5H, which serves as a buffer. Reading TMR5L will load the contents of the high byte of the register pair into the TMR5H register. This allows the user to accurately read all 16 bits of the register pair without having to determine whether a read of the high byte, followed by the low byte, is valid due to a rollover between reads.

Since the actual high byte of the Timer5 register pair is not directly readable or writable, it must be read and written to through the Timer5 High Byte Buffer register (TMR5H). The T5 high byte is updated with the contents of TMR5H when a write occurs to TMR5L. This allows a user to write all 16 bits to both the high and low bytes of Timer5 at once. Writes to TMR5H do not clear the Timer5 prescaler. The prescaler is only cleared on writes to TMR5L.

15.2.1 16-BIT READ-MODIFY-WRITE

Read-modify-write instructions, like `BSF` and `BCF`, will read the contents of a register, make the appropriate changes and place the result back into the register. The write portion of a read-modify-write instruction of TMR5H will not update the contents of the high byte of TMR5 until a write of TMR5L takes place. Only then will the contents of TMR5H be placed into the high byte of TMR5.

15.3 Timer5 Prescaler

The Timer5 clock input (either TCY or the external clock) may be divided by using the Timer5 programmable prescaler. The prescaler control bits, T5PS<1:0> (T5CON<4:3>), select a prescale factor of 2, 4, 8 or no prescale.

The Timer5 prescaler is cleared by any of the following:

- A write to the Timer5 register
- Disabling Timer5 (TMR5ON = `0`)
- A device Reset such as Master Clear, POR or BOR

Note:	Writing to the T5CON register does not clear the Timer5.

15.4 Noise Filter

The Timer5 module includes an optional input noise filter, designed to reduce spurious signals in noisy operating environments. The filter ensures that the input is not permitted to change until a stable value has been registered for three consecutive sampling clock cycles.

The noise filter is part of the input filter network associated with the Motion Feedback Module (see **Section 17.0 "Motion Feedback Module"**). All of the filters are controlled using the Digital Filter Control (DFLTCON) register (Register 17-3). The Timer5 filter can be individually enabled or disabled by setting or clearing the FLT4EN bit (DFLTCON<6>). It is disabled on all Brown-out Resets.

For additional information, refer to **Section 17.3 "Noise Filters"** in the Motion Feedback Module.

15.5 Timer5 Interrupt

Timer5 has the ability to generate an interrupt on a period match. When the PR5 register is loaded with a new period value (00FFh), the Timer5 time base increments until its value is equal to the value of PR5. When a match occurs, the Timer5 interrupt is generated on the rising edge of Q4; TMR5IF is set on the next TCY.

The interrupt latency (i.e., the time elapsed from the moment Timer5 rolls over until TMR5IF is set) will not exceed 1 TCY. When the Timer5 clock input is prescaled and a TMR5/PR5 match occurs, the interrupt will be generated on the first Q4 rising edge after TMR5 resets.

15.6 Timer5 Special Event Trigger Output

A Timer5 Special Event Trigger is generated on a TMR5/PR5 match. The Special Event Trigger is generated on the falling edge of Q3.

Timer5 must be configured for either Synchronous mode (Counter or Timer) to take advantage of the Special Event Trigger feature. If Timer5 is running in Asynchronous Counter mode, the Special Event Trigger may not work and should not be used.

15.7 Timer5 Special Event Trigger Reset Input

In addition to the Special Event Trigger output, Timer5 has a Special Event Trigger Reset input that may be used with Input Capture Channel 1 (IC1) of the Motion Feedback Module. To use the Special Event Trigger Reset, the Capture 1 Control register, CAP1CON, must be configured for one of the Special Event Trigger modes (CAP1M<3:0> = `1110` or `1111`). The Special Event Trigger Reset can be disabled by setting the RESEN control bit (T5CON<6>).

The Special Event Trigger Reset resets the Timer5 time base. This Reset occurs in either Continuous Count or Single-Shot modes.

15.7.1 WAKE-UP ON IC1 EDGE

The Timer5 Special Event Trigger Reset input can act as a Timer5 wake-up and a start-up pulse. Timer5 must be in Single-Shot mode and disabled (TMR5ON = `0`). An active edge on the CAP1 input pin will set TMR5ON. The timer is subsequently incremented on the next following clock according to the prescaler and the Timer5 clock settings. A subsequent hardware time-out (such as TMR5/PR5 match) will clear the TMR5ON bit and stop the timer.

15.7.2 DELAYED ACTION EVENT TRIGGER

An active edge on CAP1 can also be used to initiate some later action delayed by the Timer5 time base. In this case, Timer5 increments as before after being triggered. When the hardware time-out occurs, the Special Event Trigger output is generated and used to trigger another action, such as an A/D conversion. This allows a given hardware action to be referenced from a capture edge on CAP1 and delayed by the timer.

The event timing for the delayed action event trigger is discussed further in **Section 17.1 "Input Capture"**.

15.7.3 SPECIAL EVENT TRIGGER RESET WHILE TIMER5 IS INCREMENTING

In the event that a bus write to Timer5 coincides with a Special Event Trigger Reset, the bus write will always take precedence over the Special Event Trigger Reset.

15.8 Operation in Sleep Mode

When Timer5 is configured for asynchronous operation, it will continue to increment each timer clock (or prescale multiple of clocks). Executing the `SLEEP` instruction will either stop the timer or let the timer continue, depending on the setting of the Timer5 Sleep Enable bit, T5SEN. If T5SEN is set (= `1`), the timer continues to run when the `SLEEP` instruction is executed and the external clock is selected (TMR5CS = `1`). If T5SEN is cleared, the timer stops when a `SLEEP` instruction is executed, regardless of the state of the TMR5CS bit.

To summarize, Timer5 will continue to increment when a `SLEEP` instruction is executed only if all of these bits are set:

- TMR5ON
- T5SEN
- TMR5CS
- $\overline{\text{T5SYNC}}$

15.8.1 INTERRUPT DETECT IN SLEEP MODE

When configured as described above, Timer5 will continue to increment on each rising edge on T5CKI while in Sleep mode. When a TMR5/PR5 match occurs, an interrupt is generated which can wake the part.

TABLE 15-1: REGISTERS ASSOCIATED WITH TIMER5

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
IPR3	—	—	—	PTIP	IC3DRIP	IC2QEIP	IC1IP	TMR5IP	56
PIE3	—	—	—	PTIE	IC3DRIE	IC2QEIE	IC1IE	TMR5IE	56
PIR3	—	—	—	PTIF	IC3DRIF	IC2QEIF	IC1IF	TMR5IF	56
TMR5H	Timer5 Register High Byte								57
TMR5L	TImer5 Register Low Byte								57
PR5H	Timer5 Period Register High Byte								57
PR5L	Timer5 Period Register Low Byte								57
T5CON	T5SEN	$\overline{RESEN}$	T5MOD	T5PS1	T5PS0	$\overline{T5SYNC}$	TMR5CS	TMR5ON	56
CAP1CON	—	CAP1REN	—	—	CAP1M3	CAP1M2	CAP1M1	CAP1M0	59
DFLTCON	—	FLT4EN	FLT3EN	FLT2EN	FLT1EN	FLTCK2	FLTCK1	FLTCK0	59

Legend: — = unimplemented. Shaded cells are not used by the Timer5 module.

NOTES:

16.0 CAPTURE/COMPARE/PWM (CCP) MODULES

The CCP (Capture/Compare/PWM) module contains a 16-bit register that can operate as a 16-bit Capture register, a 16-bit Compare register or a PWM Master/Slave Duty Cycle register. Table 16-1 shows the timer resources required for each of the CCP module modes.

The operation of CCP1 is identical to that of CCP2, with the exception of the Special Event Trigger. Therefore, operation of a CCP module is described with respect to CCP1, except where noted.

16.1 CCP1 Module

Capture/Compare/PWM Register 1 (CCPR1) is comprised of two 8-bit registers: CCPR1L (low byte) and CCPR1H (high byte). The CCP1CON register controls the operation of CCP1. All are readable and writable.

TABLE 16-1: CCP MODE – TIMER RESOURCES

CCP Mode	Timer Resources
Capture	Timer1
Compare	Timer1
PWM	Timer2

16.2 CCP2 Module

Capture/Compare/PWM Register 2 (CCPR2) is comprised of two 8-bit registers: CCPR2L (low byte) and CCPR2H (high byte). The CCP2CON register controls the operation of CCP2. All are readable and writable.

REGISTER 16-1: CCPxCON: CCPx CONTROL REGISTER

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

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

bit 5-4 **DCxB<1:0>**: PWM Duty Cycle bit 1 and bit 0

Capture mode:
Unused.

Compare mode:
Unused.

PWM mode:
These bits are the two LSBs (bit 1 and bit 0) of the 10-bit PWM duty cycle. The upper eight bits (DCxB<9:2>) of the duty cycle are found in CCPRxL.

bit 3-0 **CCPxM<3:0>**: CCPx Mode Select bits

0000 = Capture/Compare/PWM disabled (resets CCPx module)
0001 = Reserved
0010 = Compare mode; toggle output on match (CCPxIF bit is set)
0011 = Reserved
0100 = Capture mode; every falling edge
0101 = Capture mode; every rising edge
0110 = Capture mode; every 4th rising edge
0111 = Capture mode; every 16th rising edge
1000 = Compare mode; initialize CCPx pin low; on compare match, force CCPx pin high (CCPxIF bit is set)
1001 = Compare mode; initialize CCPx pin high; on compare match, force CCPx pin low (CCPxIF bit is set)
1010 = Compare mode; generate software interrupt on compare match (CCPxIF bit is set, CCPx pin is unaffected)
1011 = Compare mode; Special Event Trigger (CCPxIF bit is set)
11xx = PWM mode

16.3 Capture Mode

In Capture mode, CCPR1H:CCPR1L captures the 16-bit value of the TMR1 register when an event occurs on pin RC2/CCP1. An event is defined as one of the following:

- every falling edge
- every rising edge
- every 4th rising edge
- every 16th rising edge

The event is selected by control bits, CCP1M<3:0> (CCP1CON<3:0>). When a capture is made, the interrupt request flag bit, CCP1IF (PIR1<2>), is set; it must be cleared in software. If another capture occurs before the value in register CCPR1 is read, the old captured value is overwritten by the new captured value.

16.3.1 CCP PIN CONFIGURATION

In Capture mode, the RC2/CCP1 pin should be configured as an input by setting the TRISC<2> bit.

Note: If the RC2/CCP1 pin is configured as an output, a write to the port can cause a capture condition.

16.3.2 TIMER1 MODE SELECTION

Timer1 must be running in Timer mode or Synchronized Counter mode to be used with the capture feature. In Asynchronous Counter mode, the capture operation may not work.

16.3.3 SOFTWARE INTERRUPT

When the Capture mode is changed, a false capture interrupt may be generated. The user should keep bit, CCP1IE (PIE1<2>), clear to avoid false interrupts and should clear the flag bit, CCP1IF, following any such change in operating mode.

16.3.4 CCP PRESCALER

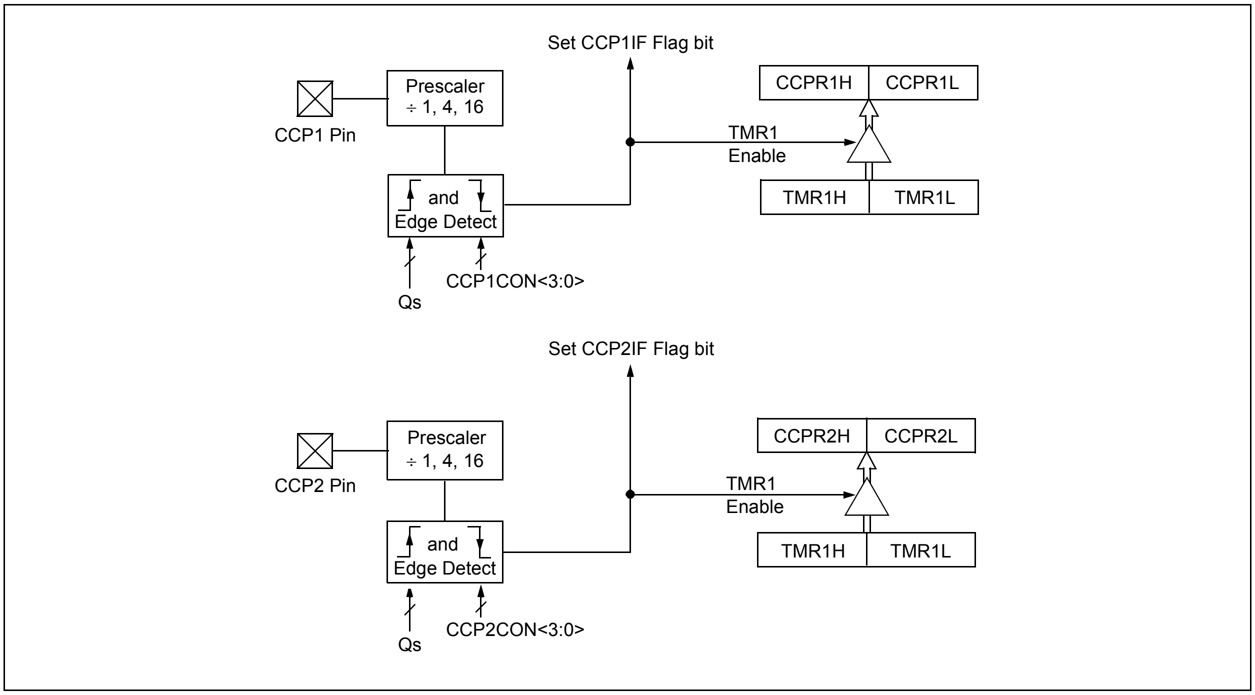
There are four prescaler settings specified by bits CCP1M<3:0>. Whenever the CCP module is turned off, or the CCP module is not in Capture mode, the prescaler counter is cleared. This means that any Reset will clear the prescaler counter.

Switching from one capture prescaler to another may generate an interrupt. Also, the prescaler counter will not be cleared, therefore, the first capture may be from a non-zero prescaler. Example 16-1 shows the recommended method for switching between capture prescalers. This example also clears the prescaler counter and will not generate the “false” interrupt.

EXAMPLE 16-1: CHANGING BETWEEN CAPTURE PRESCALERS

```
CLRF    CCP1CON          ; Turn CCP module off
MOVLW   NEW_CAPT_PS      ; Load WREG with the
                         ; new prescaler mode
                         ; value and CCP ON
MOVWF   CCP1CON          ; Load CCP1CON with
                         ; this value
```

FIGURE 16-1: CAPTURE MODE OPERATION BLOCK DIAGRAM



16.4 Compare Mode

In Compare mode, the 16-bit CCPR1 (CCPR2) register value is constantly compared against the TMR1 register pair value. When a match occurs, the RC2/CCP1 (RC1/CCP2) pin:

- is driven high
- is driven low
- toggles output (high-to-low or low-to-high)
- remains unchanged (interrupt only)

The action on the pin is based on the value of control bits, CCP1M<3:0> (CCP2M<3:0>). At the same time, interrupt flag bit CCP1IF (CCP2IF) is set.

16.4.1 CCP PIN CONFIGURATION

The user must configure the CCP1 pin as an output by clearing the appropriate TRISC bit.

Note:	Clearing the CCPxCON register will force the RC1 or RC2 compare output latch to the default low level. This is not the PORTC I/O data latch.

16.4.2 TIMER1 MODE SELECTION

Timer1 must be running in Timer mode or Synchronized Counter mode if the CCP module is using the compare feature. In Asynchronous Counter mode, the compare operation may not work.

16.4.3 SOFTWARE INTERRUPT MODE

When generate software interrupt is chosen, the CCP1 pin is not affected. Only a CCP interrupt is generated (if enabled).

16.4.4 SPECIAL EVENT TRIGGER

In this mode, an internal hardware trigger is generated which may be used to initiate an action.

The Special Event Trigger output of CCP1 resets the TMR1 register pair. This allows the CCPR1 register to effectively be a 16-bit programmable period register for Timer1.

The Special Event Trigger output of CCP2 resets the TMR1 register pair. Additionally, the CCP2 Special Event Trigger will start an A/D conversion if the A/D module is enabled.

Note:	The Special Event Trigger from the CCP2 module will not set the Timer1 interrupt flag bit.

FIGURE 16-2: COMPARE MODE OPERATION BLOCK DIAGRAM

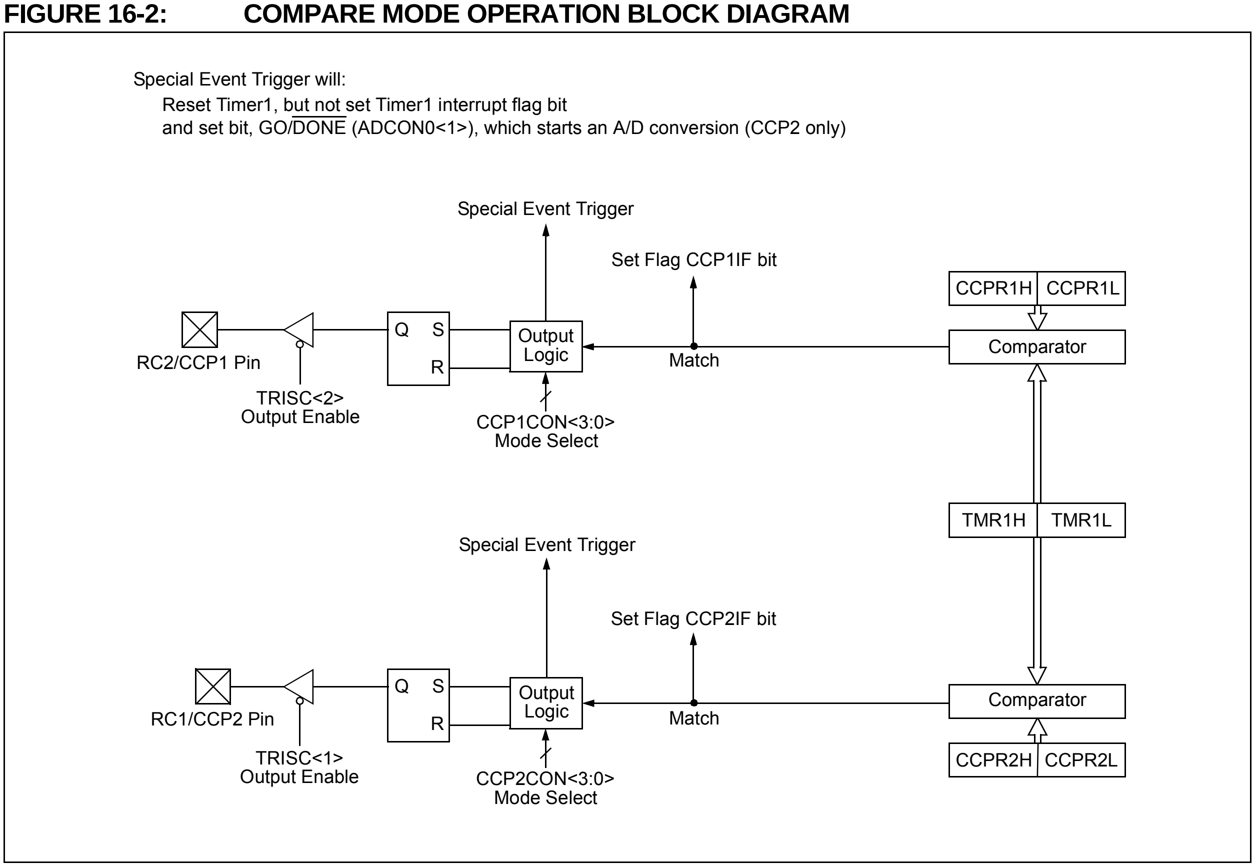


TABLE 16-2: REGISTERS ASSOCIATED WITH CAPTURE, COMPARE AND TIMER1

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
PIR1	—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	57
PIE1	—	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	57
IPR1	—	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	57
TRISC	PORTC Data Direction Register								57
TMR1L	Timer1 Register Low Byte								55
TMR1H	Timer1 Register High Byte								55
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	55
CCPR1L	Capture/Compare/PWM Register 1 Low Byte								56
CCPR1H	Capture/Compare/PWM Register 1 High Byte								56
CCP1CON	—	—	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	56
CCPR2L	Capture/Compare/PWM Register 2 Low Byte								56
CCPR2H	Capture/Compare/PWM Register 2 High Byte								56
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	56
PIR2	OSCFIF	—	—	EEIF	—	LVDIF	—	CCP2IF	57
PIE2	OSCFIE	—	—	EEIE	—	LVDIE	—	CCP2IE	57
IPR2	OSCFIP	—	—	EEIP	—	LVDIP	—	CCP2IP	57

Legend: — = unimplemented, read as '0'. Shaded cells are not used by Capture, Compare and Timer1.

16.5 PWM Mode

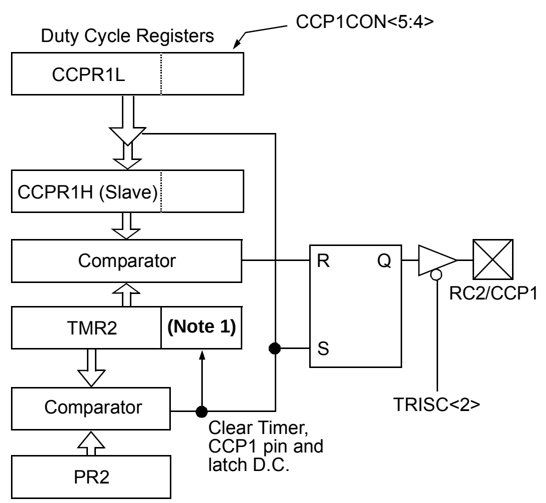
In Pulse-Width Modulation (PWM) mode, the CCP1 pin produces up to a 10-bit resolution PWM output. Since the CCP1 pin is multiplexed with the PORTC data latch, the TRISC<2> bit must be cleared to make the CCP1 pin an output.

> **Note:** Clearing the CCP1CON register will force the CCP1 PWM output latch to the default low level. This is not the PORTC I/O data latch.

Figure 16-3 shows a simplified block diagram of the CCP1 module in PWM mode.

For a step-by-step procedure on how to set up the CCP1 module for PWM operation, see **Section 16.5.3 "Setup for PWM Operation"**.

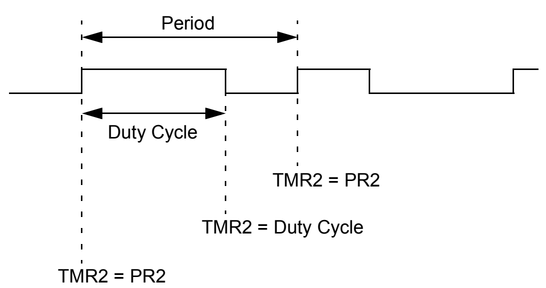
FIGURE 16-3: SIMPLIFIED PWM BLOCK DIAGRAM



Note 1: 8-bit timer is concatenated with 2-bit internal Q clock or 2 bits of the prescaler to create 10-bit time base.

A PWM output (Figure 16-4) has a time base (period) and a time that the output is high (duty cycle). The frequency of the PWM is the inverse of the period (1/period).

FIGURE 16-4: PWM OUTPUT



16.5.1 PWM PERIOD

The PWM period is specified by writing to the PR2 register. The PWM period can be calculated using the following equation:

EQUATION 16-1:

$$\text{PWM Period} = [(\text{PR2}) + 1] \bullet 4 \bullet T_{OSC} \bullet (\text{TMR2 Prescale Value})$$

PWM frequency is defined as 1/[PWM period]. When TMR2 is equal to PR2, the following three events occur on the next increment cycle:

- TMR2 is cleared
- The CCP1 pin is set (if PWM duty cycle = 0%, the CCP1 pin will not be set)
- The PWM duty cycle is copied from CCPR1L into CCPR1H

> **Note:** The Timer2 postscaler (see **Section 14.0 "Timer2 Module"**) is not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output.

16.5.2 PWM DUTY CYCLE

The PWM duty cycle is specified by writing to the CCPR1L register and to the CCP1CON<5:4> bits. Up to 10-bit resolution is available. The CCPR1L contains the eight MSbs and the CCP1CON<5:4> contains the two LSbs. This 10-bit value is represented by CCPR1L:CCP1CON<5:4>. The PWM duty cycle is calculated by the following equation:

EQUATION 16-2:

$$\text{PWM Duty Cycle} = (\text{CCPR1L:CCP1CON<5:4>}) \bullet T_{OSC} \bullet (\text{TMR2 Prescale Value})$$

CCPR1L and CCP1CON<5:4> can be written to at any time, but the duty cycle value is not copied into CCPR1H until a match between PR2 and TMR2 occurs (i.e., the period is complete). In PWM mode, CCPR1H is a read-only register.

The CCPR1H register and a 2-bit internal latch are used to double buffer the PWM duty cycle. This double buffering is essential for glitchless PWM operation. When the CCPR1H and 2-bit latch match TMR2, concatenated with an internal 2-bit Q clock or two bits of the TMR2 prescaler, the CCP1 pin is cleared. The maximum PWM resolution (bits) for a given PWM frequency is given by the following equation:

EQUATION 16-3:

$$\text{PWM Resolution (max)} = \frac{\log\left(\frac{F_{OSC}}{F_{PWM}}\right)}{\log(2)} \text{ bits}$$

Note:	If the PWM duty cycle value is longer than the PWM period, the CCP1 pin will not be cleared.

16.5.3 SETUP FOR PWM OPERATION

The following steps should be taken when configuring the CCP1 module for PWM operation:

1. Set the PWM period by writing to the PR2 register.
2. Set the PWM duty cycle by writing to the CCPR1L register and CCP1CON<5:4> bits.
3. Make the CCP1 pin an output by clearing the TRISC<2> bit.
4. Set the TMR2 prescale value and enable Timer2 by writing to T2CON.
5. Configure the CCP1 module for PWM operation.

TABLE 16-3: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 40 MHz

PWM Frequency	2.44 kHz	9.77 kHz	39.06 kHz	156.25 kHz	312.50 kHz	416.67 kHz
Timer Prescaler (1, 4, 16)	16	4	1	1	1	1
PR2 Value	FFh	FFh	FFh	3Fh	1Fh	17h
Maximum Resolution (bits)	10	10	10	8	7	6.58

TABLE 16-4: REGISTERS ASSOCIATED WITH PWM AND TIMER2

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
PIR1	—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	57
PIE1	—	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	57
IPR1	—	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	57
TRISC	PORTC Data Direction Register								57
TMR2	Timer2 Register								55
PR2	Timer2 Period Register								55
T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	55
CCPR1L	Capture/Compare/PWM Register 1 Low Byte								56
CCPR1H	Capture/Compare/PWM Register 1 High Byte								56
CCP1CON	—	—	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	56
CCPR2L	Capture/Compare/PWM Register 2 Low Byte								56
CCPR2H	Capture/Compare/PWM Register 2 High Byte								56
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	56

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used by PWM and Timer2.

17.0 MOTION FEEDBACK MODULE

The Motion Feedback Module (MFM) is a special purpose peripheral designed for motion feedback applications. Together with the Power Control PWM (PCPWM) module (see **Section 18.0 "Power Control PWM Module"**), it provides a variety of control solutions for a wide range of electric motors.

The module actually consists of two hardware submodules:

- Input Capture (IC)
- Quadrature Encoder Interface (QEI)

Together with Timer5 (see **Section 15.0 "Timer5 Module"**), these modules provide a number of options for motion and control applications.

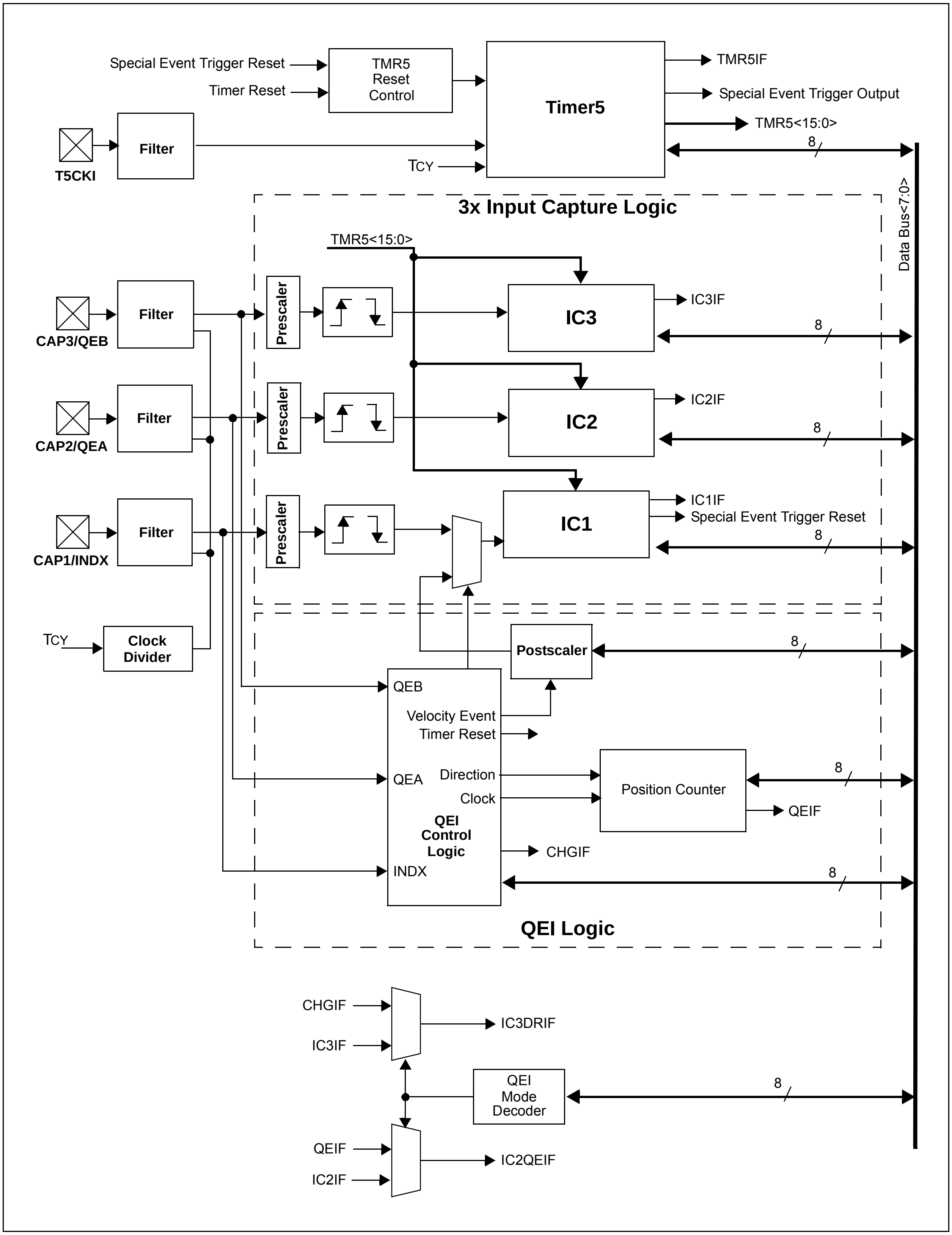
Many of the features for the IC and QEI submodules are fully programmable, creating a flexible peripheral structure that can accommodate a wide range of in-system uses. An overview of the available features is presented in Table 17-1. A simplified block diagram of the entire Motion Feedback Module is shown in Figure 17-1.

Note:	Because the same input pins are common to the IC and QEI submodules, only one of these two submodules may be used at any given time. If both modules are on, the QEI submodule will take precedence.

TABLE 17-1: SUMMARY OF MOTION FEEDBACK MODULE FEATURES

Submodule	Mode(s)	Features	Timer	Function
IC (3x)	• Synchronous • Input Capture	• Flexible Input Capture modes • Available Prescaler • Selectable Time Base Reset • Special Event Trigger for ADC Sampling/Conversion or Optional TMR5 Reset Feature (CAP1 only) • Wake-up from Sleep function • Selectable Interrupt Frequency • Optional Noise Filter	TMR5	• 3x Input Capture (edge capture, pulse width, period measurement, capture on change) • Special Event Triggers the A/D Conversion on the CAP1 Input
QEI	QEI	• Detect Position • Detect Direction of Rotation • Large Bandwidth (FCY/16) • Optional Noise Filter	16-Bit Position Counter	• Position Measurement • Direction of Rotation Status
	Velocity Measurement	• 2x and 4x Update modes • Velocity Event Postscaler • Counter Overflow Flag for Low Rotation Speed • Utilizes Input Capture 1 Logic (IC1) • High and Low Velocity Support	TMR5	• Precise Velocity Measurement • Direction of Rotation Status

FIGURE 17-1: MOTION FEEDBACK MODULE BLOCK DIAGRAM



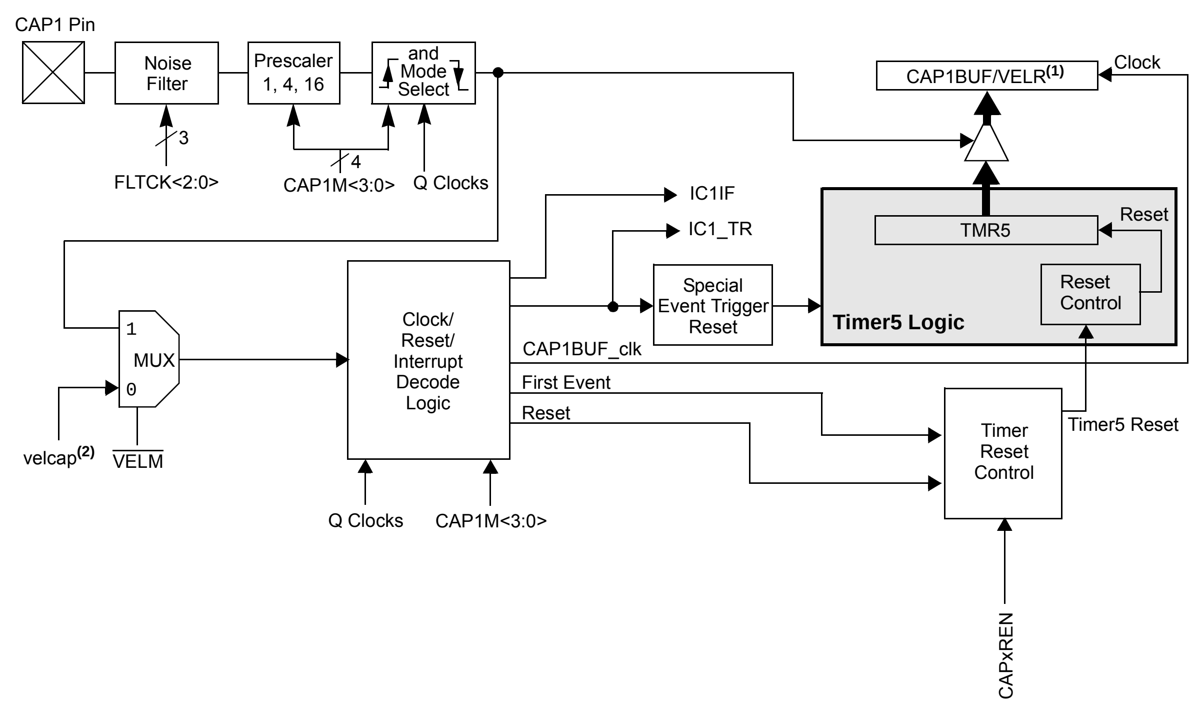
17.1 Input Capture

The Input Capture (IC) submodule implements the following features:

- Three channels of independent input capture (16-bits/channel) on the CAP1, CAP2 and CAP3 pins
- Edge-Trigger, Period or Pulse-Width Measurement Operating modes for each channel
- Programmable prescaler on every input capture channel
- Special Event Trigger output (IC1 only)
- Selectable noise filters on each capture input

Input Channel 1 (IC1) includes a Special Event Trigger that can be configured for use in Velocity Measurement mode. Its block diagram is shown in Figure 17-2. IC2 and IC3 are similar, but lack the Special Event Trigger features or additional velocity measurement logic. A representative block diagram is shown in Figure 17-3. Please note that the time base is Timer5.

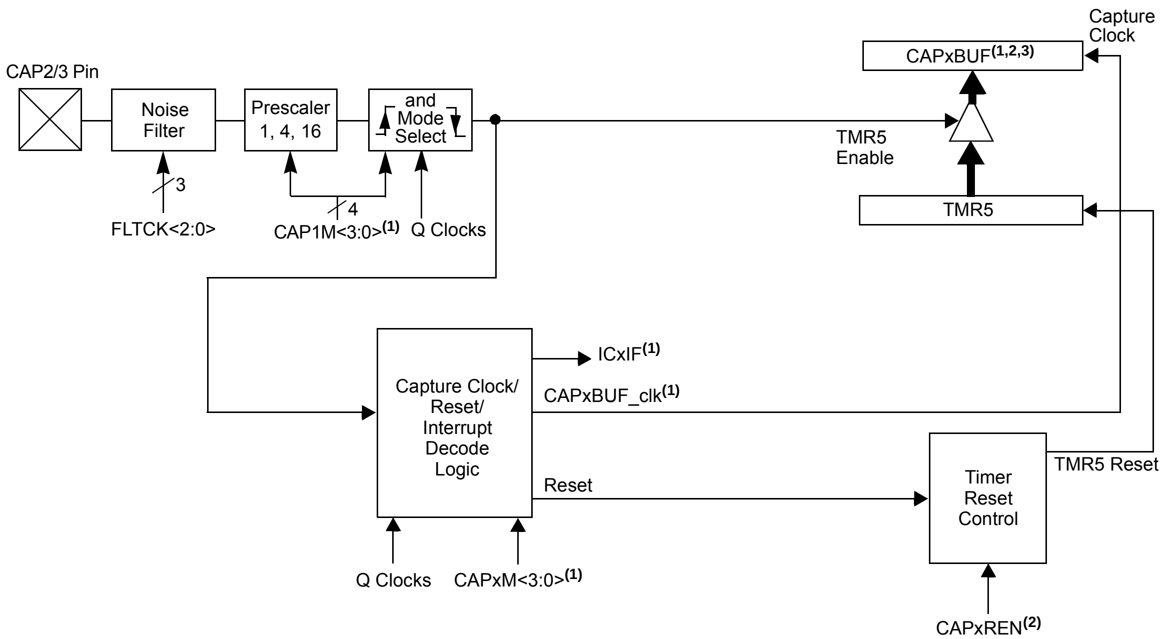
FIGURE 17-2: INPUT CAPTURE BLOCK DIAGRAM FOR IC1



Note 1: CAP1BUF register is reconfigured as VELR register when QEI mode is active.

2: QEI generated velocity pulses, vel_out, are downsampled to produce this velocity capture signal.

FIGURE 17-3: INPUT CAPTURE BLOCK DIAGRAM FOR IC2 AND IC3



Note 1: IC2 and IC3 are denoted as x = 2 and 3.

2: CAP2BUF is enabled as POSCNT when QEI mode is active.

3: CAP3BUF is enabled as MAXCNT when QEI mode is active.

The three input capture channels are controlled through the Input Capture Control registers, CAP1CON, CAP2CON and CAP3CON. Each channel is configured independently with its dedicated register. The implementation of the registers is identical except for the Special Event Trigger (see **Section 17.1.8 "Special Event Trigger (CAP1 Only)"**). The typical Capture Control register is shown in Register 17-1.

Note:	Throughout this section, references to registers and bit names that may be associated with a specific capture channel will be referred to generically by the use of the term 'x' in place of the channel number. For example, 'CAPxREN' may refer to the Capture Reset Enable bit in CAP1CON, CAP2CON or CAP3CON.

REGISTER 17-1: CAPxCON: INPUT CAPTURE x CONTROL REGISTER

U-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
—	CAPxREN	—	—	CAPxM3	CAPxM2	CAPxM1	CAPxM0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

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

bit 6 **CAPxREN:** Time Base Reset Enable bit
1 = Enabled
0 = Disable selected time base Reset on capture

bit 5-4 **Unimplemented:** Read as '0'

bit 3-0 **CAPxM<3:0>:** Input Capture x (ICx) Mode Select bits
1111 = Special Event Trigger mode; the trigger occurs on every rising edge on CAP1 input(1)
1110 = Special Event Trigger mode; the trigger occurs on every falling edge on CAP1 input(1)
1101 = Unused
1100 = Unused
1011 = Unused
1010 = Unused
1001 = Unused
1000 = Capture on every CAPx input state change
0111 = Pulse-Width Measurement mode, every rising to falling edge
0110 = Pulse-Width Measurement mode, every falling to rising edge
0101 = Frequency Measurement mode, every rising edge
0100 = Capture mode, every 16th rising edge
0011 = Capture mode, every 4th rising edge
0010 = Capture mode, every rising edge
0001 = Capture mode, every falling edge
0000 = Input Capture x (ICx) off

Note 1: Special Event Trigger is only available on CAP1. For CAP2 and CAP3, this configuration is unused.

When in Counter mode, the counter must be configured as the synchronous counter only ($\overline{T5SYNC}$ = 0). When configured in Asynchronous mode, the IC module will not work properly.

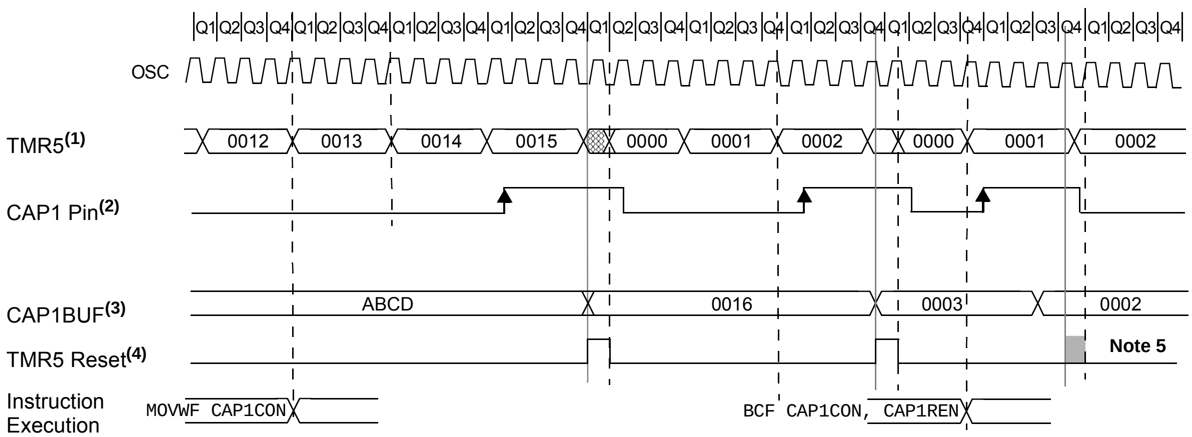
> **Note 1:** Input capture prescalers are reset (cleared) when the input capture module is disabled (CAPxM = 0000).
>
> **2:** When the Input Capture mode is changed, without first disabling the module and entering the new Input Capture mode, a false interrupt (or Special Event Trigger on IC1) may be generated. The user should either: (1) disable the input capture before entering another mode, or (2) disable IC interrupts to avoid false interrupts during IC mode changes.
>
> **3:** During IC mode changes, the prescaler count will not be cleared, therefore, the first capture in the new IC mode may be from the non-zero prescaler.

17.1.1 EDGE CAPTURE MODE

In this mode, the value of the time base is captured either on every rising edge, every falling edge, every 4th rising edge, or every 16th rising edge. The edge present on the input capture pin (CAP1, CAP2 or CAP3) is sampled by the synchronizing latch. The signal is used to load the Input Capture Buffer (ICxBUF register) on the following Q1 clock (see Figure 17-4). Consequently, Timer5 is either reset to '0' (Q1 immediately following the capture event) or left free running, depending on the setting of the Capture Reset Enable bit, CAPxREN, in the CAPxCON register.

> **Note:** On the first capture edge following the setting of the Input Capture mode (i.e., `MOVWF CAP1CON`), Timer5 contents are always captured into the corresponding Input Capture Buffer (i.e., CAPxBUF). Timer5 can optionally be reset; however, this is dependent on the setting of the Capture Reset Enable bit, CAPxREN (see Figure 17-4).

FIGURE 17-4: EDGE CAPTURE MODE TIMING



Note 1: TMR5 is a synchronous time base input to the input capture; prescaler = 1:1. It increments on the Q1 rising edge.

2: IC1 is configured in Edge Capture mode (CAP1M<3:0> = 0010) with the time base reset upon edge capture (CAP1REN = 1) and no noise filter.

3: TMR5 value is latched by CAP1BUF on TCY. In the event that a write to TMR5 coincides with an input capture event, the write will always take precedence. All Input Capture Buffers, CAP1BUF, CAP2BUF and CAP3BUF, are updated with the incremented value of the time base on the next TCY clock edge when the capture event takes place (see **Note 4** when Reset occurs).

4: TMR5 Reset is normally an asynchronous Reset signal to TMR5. When used with the input capture, it is active immediately after the time base value is captured.

5: TMR5 Reset pulse is disabled by clearing the CAP1REN bit (e.g., `BCF CAP1CON, CAP1REN`).

17.1.2 PERIOD MEASUREMENT MODE

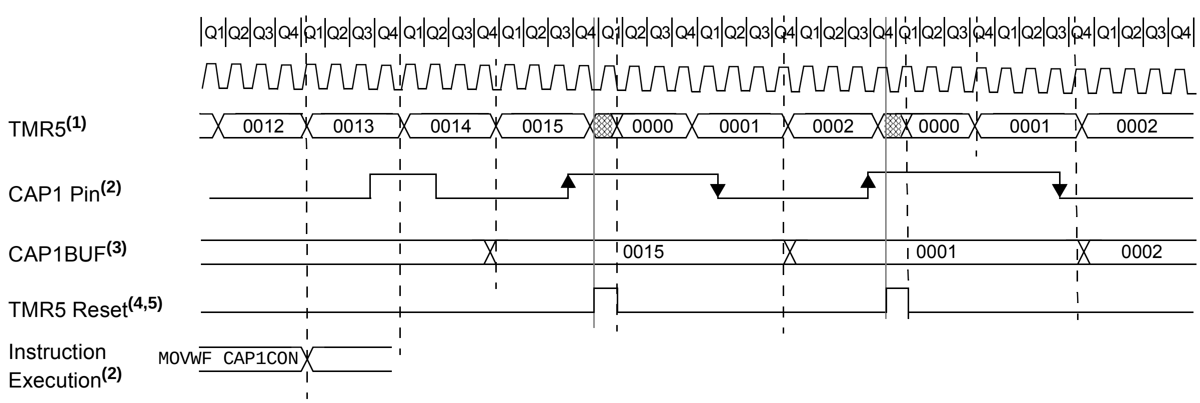
The Period Measurement mode is selected by setting CAPxM<3:0> = `0101`. In this mode, the value of Timer5 is latched into the CAPxBUF register on the rising edge of the input capture trigger and Timer5 is subsequently reset to 0000h (optional by setting CAPxREN = `1`) on the next TCY (see capture and Reset relationship in Figure 17-4).

17.1.3 PULSE-WIDTH MEASUREMENT MODE

The Pulse-Width Measurement mode can be configured for two different edge sequences, such that the pulse width is based on either the falling to rising edge of the CAPx input pin (CAPxM<3:0> = `0110`), or on the rising to falling edge (CAPxM<3:0> = `0111`).

Timer5 is always reset on the edge when the measurement is first initiated. For example, when the measurement is based on the falling to rising edge, Timer5 is first reset on the falling edge, and thereafter, the timer value is captured on the rising edge. Upon entry into the Pulse-Width Measurement mode, the very first edge detected on the CAPx pin is always captured. The TMR5 value is reset on the first active edge (see Figure 17-5).

FIGURE 17-5: PULSE-WIDTH MEASUREMENT MODE TIMING



Note 1: TMR5 is a synchronous time base input to the input capture; prescaler = 1:1. It increments on every Q1 rising edge.

2: IC1 is configured in Pulse-Width Measurement mode (CAP1M<3:0> = `0111`, rising to falling pulse-width measurement). No noise filter on CAP1 input is used. The `MOVWF` instruction loads CAP1CON when W = `0111`.

3: TMR5 value is latched by CAP1BUF on TCY rising edge. In the event that a write to TMR5 coincides with an input capture event, the write will always take precedence. All Input Capture Buffers, CAP1BUF, CAP2BUF and CAP3BUF, are updated with the incremented value of the time base on the next TCY clock edge when the capture event takes place (see **Note 4** when Reset occurs).

4: TMR5 Reset is normally an asynchronous Reset signal to TMR5. When used in Pulse-Width Measurement mode, it is always present on the edge that first initiates the pulse-width measurement (i.e., when configured in the rising to falling Pulse-Width Measurement mode); it is active on each rising edge detected. In the falling to rising Pulse-Width Measurement mode, it is active on each falling edge detected.

5: TMR5 Reset pulse is activated on the capture edge. The CAP1REN bit has no bearing in this mode.

17.1.3.1 Pulse-Width Measurement Timing

Pulse-width measurement accuracy can only be ensured when the pulse-width high and low present on the CAPx input exceeds one TCY clock cycle. The limitations depend on the mode selected:

- When CAPxM<3:0> = 0110 (rising to falling edge delay), the CAPx input high pulse width (TCCH) must exceed TCY + 10 ns.
- When CAPxM<3:0> = 0111 (falling to rising edge delay), the CAPx input low pulse width (TCCL) must exceed TCY + 10 ns.

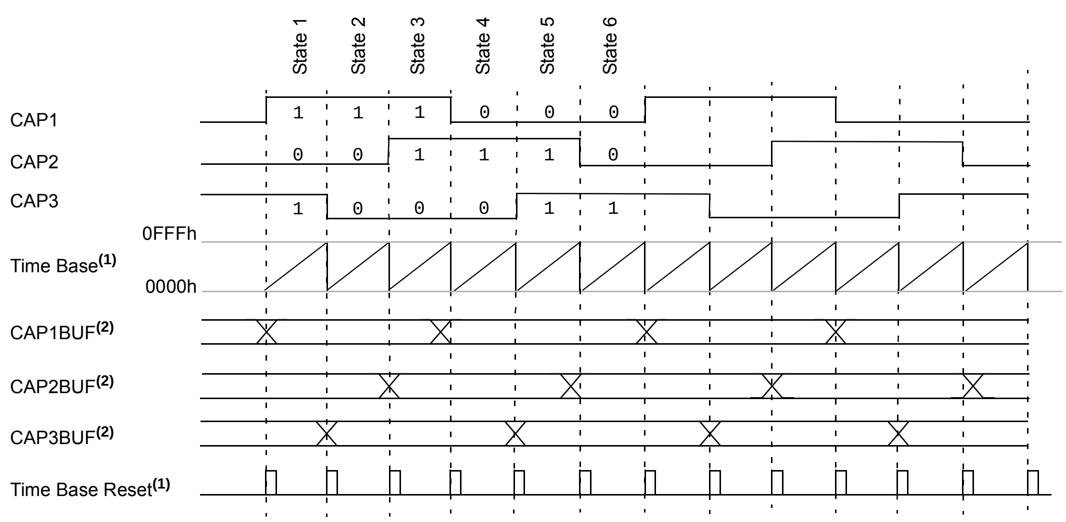
> **Note 1:** The Period Measurement mode will produce valid results upon sampling of the second rising edge of the input capture. CAPxBUF values latched during the first active edge after initialization are invalid.
>
> **2:** The Pulse-Width Measurement mode will latch the value of the timer upon sampling of the first input signal edge by the input capture.

17.1.4 INPUT CAPTURE ON STATE CHANGE

When CAPxM<3:0> = 1000, the value is captured on every signal change on the CAPx input. If all three capture channels are configured in this mode, the three input captures can be used as the Hall effect sensor state transition detector. The value of Timer5 can be captured, Timer5 reset and the interrupt generated. Any change on CAP1, CAP2 or CAP3 is detected and the associated time base count is captured.

For position and velocity measurement in this mode, the timer can be optionally reset (see **Section 17.1.6 "Timer5 Reset"** for Reset options).

FIGURE 17-6: INPUT CAPTURE ON STATE CHANGE (HALL EFFECT SENSOR MODE)



Note 1: TMR5 can be selected as the time base for input capture. The time base can be optionally reset when the Capture Reset Enable bit is set (CAPxREN = 1).

2: Detailed CAPxBUF event timing (all modes reflect the same capture and Reset timing) is shown in Figure 17-4. There are six commutation BLDC Hall effect sensor states shown. The other two remaining states (i.e., 000h and 111h) are invalid in the normal operation. They remain to be decoded by the CPU firmware in BLDC motor application.

17.1.5 ENTERING INPUT CAPTURE MODE AND CAPTURE TIMING

The following is a summary of functional operation upon entering any of the Input Capture modes:

1. After the module is configured for one of the Capture modes by setting the Capture Mode Select bits (CAPxM<3:0>), the first detected edge captures the Timer5 value and stores it in the CAPxBUF register. The timer is then reset (depending on the setting of CAPxREN bit) and starts to increment according to its settings (see Figure 17-4, Figure 17-5 and Figure 17-6).
2. On all edges, the capture logic performs the following:
   a) Input Capture mode is decoded and the active edge is identified.
   b) The CAPxREN bit is checked to determine whether Timer5 is reset or not.
   c) On every active edge, the Timer5 value is recorded in the Input Capture Buffer (CAPxBUF).
   d) Reset Timer5 after capturing the value of the timer when the CAPxREN bit is enabled. Timer5 is reset on every active capture edge in this case.
   e) On all continuing capture edge events, repeat steps (a) through (d) until the operational mode is terminated, either by user firmware, POR or BOR.
   f) The timer value is not affected when switching into and out of various Input Capture modes.

17.1.6 TIMER5 RESET

Every input capture trigger can optionally reset (TMR5). The Capture Reset Enable bit, CAPxREN, gates the automatic Reset of the time base of the capture event with this enable Reset signal. All capture events reset the selected timer when CAPxREN is set. Resets are disabled when CAPxREN is cleared (see Figure 17-4, Figure 17-5 and Figure 17-6).

Note:	The CAPxREN bit has no effect in Pulse-Width Measurement mode.

17.1.7 IC INTERRUPTS

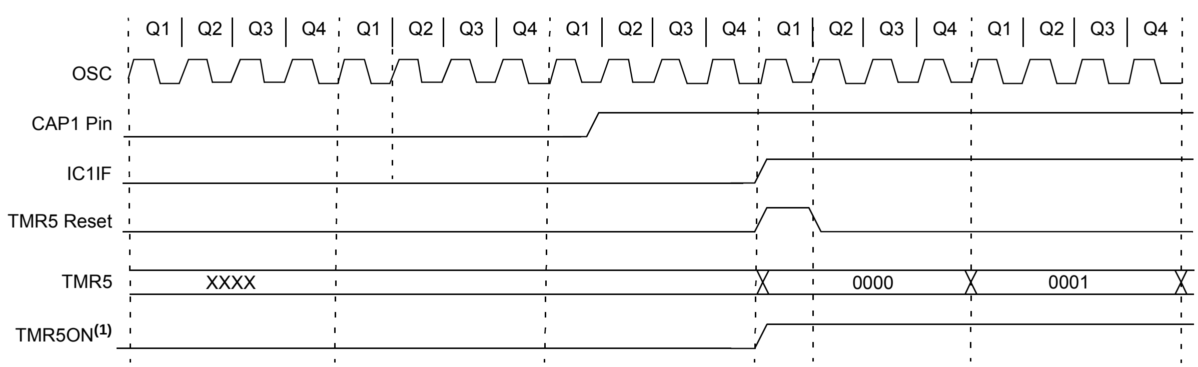
There are four operating modes for which the IC module can generate an interrupt and set one of the Interrupt Capture Flag bits (IC1IF, IC2QEIF or IC3DRIF). The interrupt flag that is set depends on the channel in which the event occurs. The modes are:

- Edge Capture (CAPxM<3:0> = `0001`, `0010`, `0011` or `0100`)
- Period Measurement Event (CAPxM<3:0> = `0101`)
- Pulse-Width Measurement Event (CAPxM<3:0> = `0110` or `0111`)
- State Change Event (CAPxM<3:0> = `1000`)

Note:	The Special Event Trigger is generated only in the Special Event Trigger mode on the CAP1 input (CAP1M<3:0> = `1110` and `1111`). IC1IF interrupt is not set in this mode.

The timing of interrupt and Special Event Trigger events is shown in Figure 17-7. Any active edge is detected on the rising edge of Q2 and propagated on the rising edge of Q4 rising edge. If an active edge happens to occur any later than this (on the falling edge of Q2, for example), then it will be recognized on the next Q2 rising edge.

FIGURE 17-7: CAPx INTERRUPTS AND IC1 SPECIAL EVENT TRIGGER



Note 1: Timer5 is only reset and enabled (assuming TMR5ON = `0` and T5MOD = `1`) when the Special Event Trigger Reset is enabled for the Timer5 Reset input. The TMR5ON bit is asserted and Timer5 is reset on the Q1 rising edge following the event capture. With the Special Event Trigger Reset disabled, Timer5 cannot be reset by the Special Event Trigger Reset on the CAP1 input. In order for the Special Event Trigger Reset to work as the Reset trigger to Timer5, IC1 must be configured in the Special Event Trigger mode (CAP1M<3:0> = `1110` or `1111`).

17.1.8 SPECIAL EVENT TRIGGER (CAP1 ONLY)

The Special Event Trigger mode of IC1 (CAP1M<3:0> = 1110 or 1111) enables the Special Event Trigger signal. The trigger signal can be used as the Special Event Trigger Reset input to TMR5, resetting the timer when the specific event happens on IC1. The events are summarized in Table 17-2.

TABLE 17-2: SPECIAL EVENT TRIGGER

CAP1M<3:0>	Description
1110	The trigger occurs on every **falling** edge on the CAP1 input.
1111	The trigger occurs on every **rising** edge on the CAP1 input.

17.1.9 OPERATING MODES SUMMARY

Table 17-3 shows a summary of the input capture configuration when used in conjunction with the TMR5 timer resource.

17.1.10 OTHER OPERATING MODES

Although the IC and QEI submodules are mutually exclusive, the IC can be reconfigured to work with the QEI module to perform specific functions. In effect, the QEI "borrows" hardware from the IC to perform these operations.

For velocity measurement, the QEI uses dedicated hardware in channel IC1. The CAP1BUF registers are remapped, becoming the VELR registers. Its operation and use are described in **Section 17.2.6 "Velocity Measurement"**.

While in QEI mode, the CAP2BUF and CAP3BUF registers of channel IC2 and IC3 are used for position determination. They are remapped as the POSCNT and MAXCNT Buffer registers, respectively.

TABLE 17-3: INPUT CAPTURE TIME BASE RESET SUMMARY

Pin	CAPxM	Mode	Timer	Reset Timer on Capture	Description
CAP1	0001-0100	Edge Capture	TMR5	optional(1)	Simple Edge Capture mode (includes a selectable prescaler).
	0101	Period Measurement	TMR5	optional(1)	Captures Timer5 on period boundaries.
	0110-0111	Pulse-Width Measurement	TMR5	always	Captures Timer5 on pulse boundaries.
	1000	Input Capture on State Change	TMR5	optional(1)	Captures Timer5 on state change.
	1110-1111	Special Event Trigger (rising or falling edge)	TMR5	optional(2)	Used as a Special Event Trigger to be used with the Timer5 or other peripheral modules.
CAP2	0001-0100	Edge Capture	TMR5	optional(1)	Simple Edge Capture mode (includes a selectable prescaler).
	0101	Period Measurement	TMR5	optional(1)	Captures Timer5 on period boundaries.
	0110-0111	Pulse-Width Measurement	TMR5	always	Captures Timer5 on pulse boundaries.
	1000	Input Capture on State Change	TMR5	optional(1)	Captures Timer5 on state change.
CAP3	0001-0100	Edge Capture	TMR5	optional(1)	Simple Edge Capture mode (includes a selectable prescaler).
	0101	Period Measurement	TMR5	optional(1)	Captures Timer5 on period boundaries.
	0110-0111	Pulse-Width Measurement	TMR5	always	Captures Timer5 on pulse boundaries.
	1000	Input Capture on State Change	TMR5	optional(1)	Captures Timer5 on state change.

Note 1: Timer5 may be reset on capture events only when CAPxREN = 1.
2: Trigger mode will not reset Timer5 unless $\overline{\text{RESEN}}$ = 0 in the T5CON register.

17.2 Quadrature Encoder Interface

The Quadrature Encoder Interface (QEI) decodes speed and motion sensor information. It can be used in any application that uses a quadrature encoder for feedback. The interface implements these features:

- Three QEI inputs: two phase signals (QEA and QEB) and one index signal (INDX)
- Direction of movement detection with a direction change interrupt (IC3DRIF)
- 16-bit up/down position counter
- Standard and High-Precision Position Tracking modes
- Two Position Update modes (x2 and x4)
- Velocity measurement with a programmable postscaler for high-speed velocity measurement
- Position counter interrupt (IC2QEIF in the PIR3 register)
- Velocity control interrupt (IC1IF in the PIR3 register)

The QEI submodule has three main components: the QEI control logic block, the position counter and velocity postscaler.

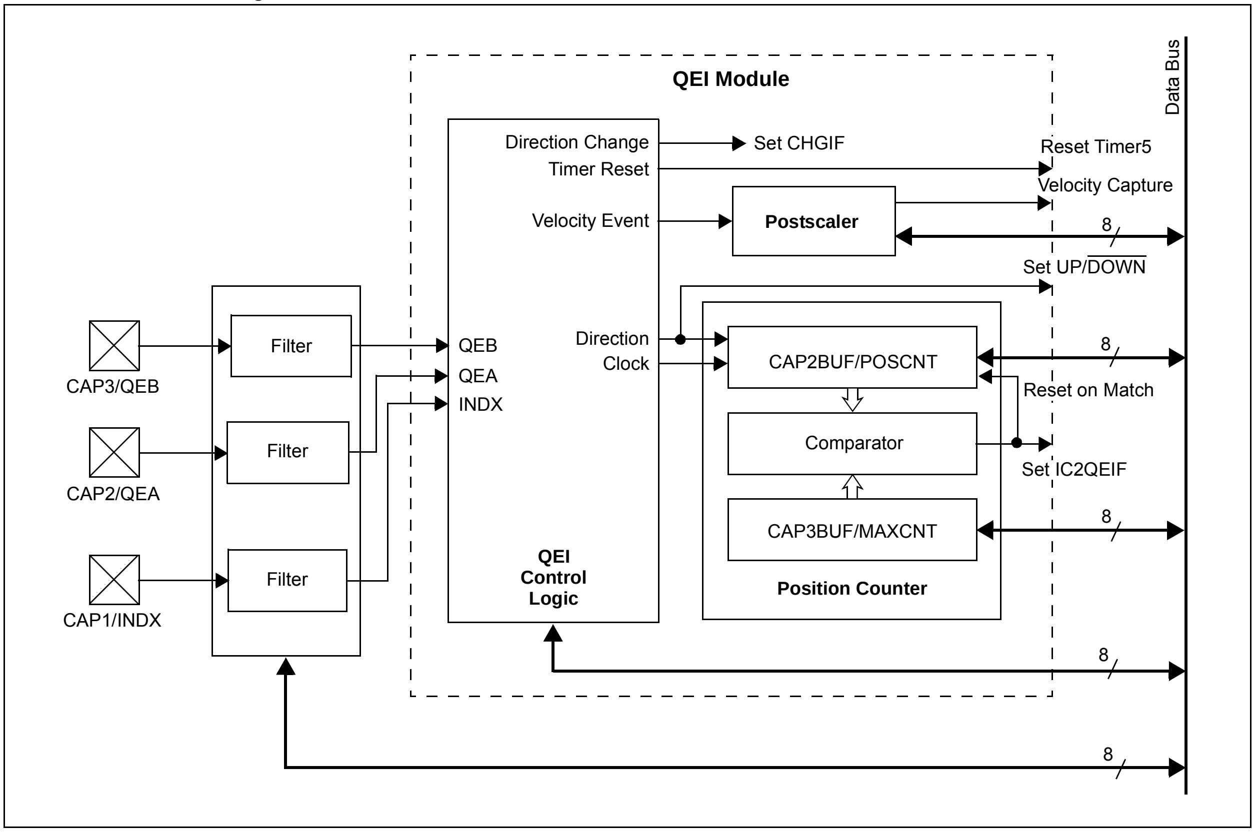
The QEI control logic detects the leading edge on the QEA or QEB phase input pins and generates the count pulse, which is sent to the position counter logic. It also samples the index input signal (INDX) and generates the direction of the rotation signal (up/down) and the velocity event signals.

The position counter acts as an integrator for tracking distance traveled. The QEA and QEB input edges serve as the stimulus to create the input clock which advances the Position Counter register (POSCNT). The register is incremented on either the QEA input edge, or the QEA and QEB input edges, depending on the operating mode. It is reset either by a rollover on match to the Period register, MAXCNT, or on the external index pulse input signal (INDX). An interrupt is generated on a Reset of POSCNT if the position counter interrupt is enabled.

The velocity postscaler down samples the velocity pulses used to increment the velocity counter by a specified ratio. It essentially divides down the number of velocity pulses to one output per so many inputs, preserving the pulse width in the process.

A simplified block-diagram of the QEI module is shown in Figure 17-8.

FIGURE 17-8: QEI BLOCK DIAGRAM



17.2.1 QEI CONFIGURATION

The QEI module shares its input pins with the Input Capture (IC) module. The inputs are mutually exclusive; only the IC module or the QEI module (but not both) can be enabled at one time. Also, because the IC and QEI are multiplexed to the same input pins, the programmable noise filters can be dedicated to one module only.

The operation of the QEI is controlled by the QEICON Configuration register (see Register 17-2).

> **Note:** In the event that both QEI and IC are enabled, QEI will take precedence and IC will remain disabled.

REGISTER 17-2: QEICON: QUADRATURE ENCODER INTERFACE CONTROL REGISTER

R/W-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
$\overline{\text{VELM}}$	QERR[1]	UP/$\overline{\text{DOWN}}$	QEIM2[2,3]	QEIM1[2,3]	QEIM0[2,3]	PDEC1	PDEC0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **$\overline{\text{VELM}}$:** Velocity Mode bit
1 = Velocity mode disabled
0 = Velocity mode enabled

bit 6 **QERR:** QEI Error bit[1]
1 = Position counter overflow or underflow[4]
0 = No overflow or underflow

bit 5 **UP/$\overline{\text{DOWN}}$:** Direction of Rotation Status bit
1 = Forward
0 = Reverse

bit 4-2 **QEIM<2:0>:** QEI Mode bits[2,3]
111 = Unused
110 = QEI enabled in 4x Update mode; position counter is reset on period match (POSCNT = MAXCNT)
101 = QEI enabled in 4x Update mode; INDX resets the position counter
100 = Unused
010 = QEI enabled in 2x Update mode; position counter is reset on period match (POSCNT = MAXCNT)
001 = QEI enabled in 2x Update mode; INDX resets the position counter
000 = QEI off

bit 1-0 **PDEC<1:0>:** Velocity Pulse Reduction Ratio bits
11 = 1:64
10 = 1:16
01 = 1:4
00 = 1:1

Note 1: QEI must be enabled and in Index mode.
2: QEI mode select must be cleared (= 000) to enable CAP1, CAP2 or CAP3 inputs. If QEI and IC modules are both enabled, QEI will take precedence.
3: Enabling one of the QEI operating modes remaps the IC Buffer registers, CAP1BUFH, CAP1BUFL, CAP2BUFH, CAP2BUFL, CAP3BUFH and CAP3BUFL, as the VELRH, VELRL, POSCNTH, POSCNTL, MAXCNTH and MAXCNTL registers (respectively) for the QEI.
4: The QERR bit must be cleared in software.

17.2.2 QEI MODES

Position measurement resolution depends on how often the Position Counter register, POSCNT, is incremented. There are two QEI Update modes to measure the rotor's position: QEI x2 and QEI x4.

TABLE 17-4: QEI MODES

QEIM<2:0>	Mode/ Reset	Description
`000`	—	QEI disabled.(1)
`001`	x2 update/ index pulse	Two clocks per QEA pulse. INDX resets POSCNT.
`010`	x2 update/ period match	Two clocks per QEA pulse. POSCNT is reset by the period match (MAXCNT).
`011`	—	Unused.
`100`	—	Unused.
`101`	x4 update/ index pulse	Four clocks per QEA and QEB pulse pair. INDX resets POSCNT.
`110`	x4 update/ period match	Four clocks per QEA and QEB pulse pair. POSCNT is reset by the period match (MAXCNT).
`111`	—	Unused.

Note 1: QEI module is disabled. The position counter and the velocity measurement functions are fully disabled in this mode.

17.2.2.1 QEI x2 Update Mode

QEI x2 Update mode is selected by setting the QEI Mode Select bits (QEIM<2:0>) to '`001`' or '`010`'. In this mode, the QEI logic detects every edge on the QEA input only. Every rising and falling edge on the QEA signal clocks the position counter.

The position counter can be reset by either an input on the INDX pin (QEIM<2:0> = `001`), or by a period match, even when the POSCNT register pair equals MAXCNT (QEIM<2:0> = `010`).

17.2.2.2 QEI x4 Update Mode

QEI x4 Update mode provides for a finer resolution of the rotor position, since the counter increments or decrements more frequently for each QEA/QEB input pulse pair than in QEI x2 mode. This mode is selected by setting the QEI mode select bits to '`101`' or '`110`'. In QEI x4, the phase measurement is made on the rising and the falling edges of both QEA and QEB inputs. The position counter is clocked on every QEA and QEB edge.

Like QEI x2 mode, the position counter can be reset by an input on the pin (QEIM<2:0> = `101`), or by the period match event (QEIM<2:0> = `010`).

17.2.3 QEI OPERATION

The Position Counter register pair (POSCNTH: POSCNTL) acts as an integrator, whose value is proportional to the position of the sensor rotor that corresponds to the number of active edges detected. POSCNT can either increment or decrement, depending on a number of selectable factors which are decoded by the QEI logic block. These include the Count mode selected, the phase relationship of QEA to QEB ("lead/lag"), the direction of rotation and if a Reset event occurs. The logic is detailed in the sections that follow.

17.2.3.1 Edge and Phase Detect

In the first step, the active edges of QEA and QEB are detected, and the phase relationship between them is determined. The position counter is changed based on the selected QEI mode.

In QEI x2 Update mode, the position counter increments or decrements on every QEA edge based on the phase relationship of the QEA and QEB signals.

In QEI x4 Update mode, the position counter increments or decrements on every QEA and QEB edge based on the phase relationship of the QEA and QEB signals. For example, if QEA leads QEB, the position counter is incremented by '`1`'. If QEB lags QEA, the position counter is decremented by '`1`'.

17.2.3.2 Direction of Count

The QEI control logic generates a signal that sets the UP/$\overline{\text{DOWN}}$ bit (QEICON<5>); this, in turn, determines the direction of the count. When QEA leads QEB, UP/$\overline{\text{DOWN}}$ is set (= `1`) and the position counter increments on every active edge. When QEA lags QEB, UP/$\overline{\text{DOWN}}$ is cleared and the position counter decrements on every active edge.

TABLE 17-5: DIRECTION OF ROTATION

Current Signal Detected	Previous Signal Detected				Pos. Cntrl.(1)
	Rising		Falling		
	QEA	QEB	QEA	QEB	
QEA Rising				x	INC
		x			DEC
QEA Falling				x	DEC
		x			INC
QEB Rising	x				INC
			x		DEC
QEB Falling			x		INC
	x				DEC

Note 1: When UP/$\overline{\text{DOWN}}$ = `1`, the position counter is incremented. When UP/$\overline{\text{DOWN}}$ = `0`, the position counter is decremented.

17.2.3.3 Reset and Update Events

The position counter will continue to increment or decrement until one of the following events takes place. The type of event and the direction of rotation when it happens determines if a register Reset or update occurs.

1. An index pulse is detected on the INDX input (QEIM<2:0> = `001`).

   If the encoder is traveling in the **forward** direction, POSCNT is reset (00h) on the next clock edge after the index marker, INDX, has been detected. The position counter resets on the QEA or QEB edge once the INDX rising edge has been detected.

   If the encoder is traveling in the **reverse** direction, the value in the MAXCNT register is loaded into POSCNT on the next quadrature pulse edge (QEA or QEB) after the falling edge on INDX has been detected.

2. A POSTCNT/MAXCNT period match occurs (QEIM<2:0> = `010`).

   If the encoder is traveling in the **forward** direction, POSCNT is reset (00h) on the next clock edge when POSCNT = MAXCNT. An interrupt event is triggered on the next TCY after the Reset (see Figure 17-10)

   If the encoder is traveling in the **reverse** direction and the value of POSCNT reaches 00h, POSCNT is loaded with the contents of the MAXCNT register on the next clock edge. An interrupt event is triggered on the next TCY after the load operation (see Figure 17-10).

The value of the position counter is not affected during QEI mode changes, nor when the QEI is disabled altogether.

17.2.4 QEI INTERRUPTS

The position counter interrupt occurs and the interrupt flag (IC2QEIF) is set, based on the following events:

- A POSCNT/MAXCNT period match event (QEIM<2:0> = `010` or `110`)
- A POSCNT rollover (FFFFh to 0000h) in Period mode only (QEIM<2:0> = `010` or `110`)
- An index pulse detected on INDX

The interrupt timing diagrams for IC2QEIF are shown in Figure 17-10 and Figure 17-11.

When the direction has changed, the direction change interrupt flag (IC3DRIF) is set on the following TCY clock (see Figure 17-10).

If the position counter rolls over in Index mode, the QERR bit will be set.

17.2.5 QEI SAMPLE TIMING

The quadrature input signals, QEA and QEB, may vary in quadrature frequency. The minimum quadrature input period, TQEI, is 16 TCY.

The position count rate, FPOS, is directly proportional to the rotor's RPM, line count D and QEI Update mode (x2 versus x4); that is,

EQUATION 17-1:

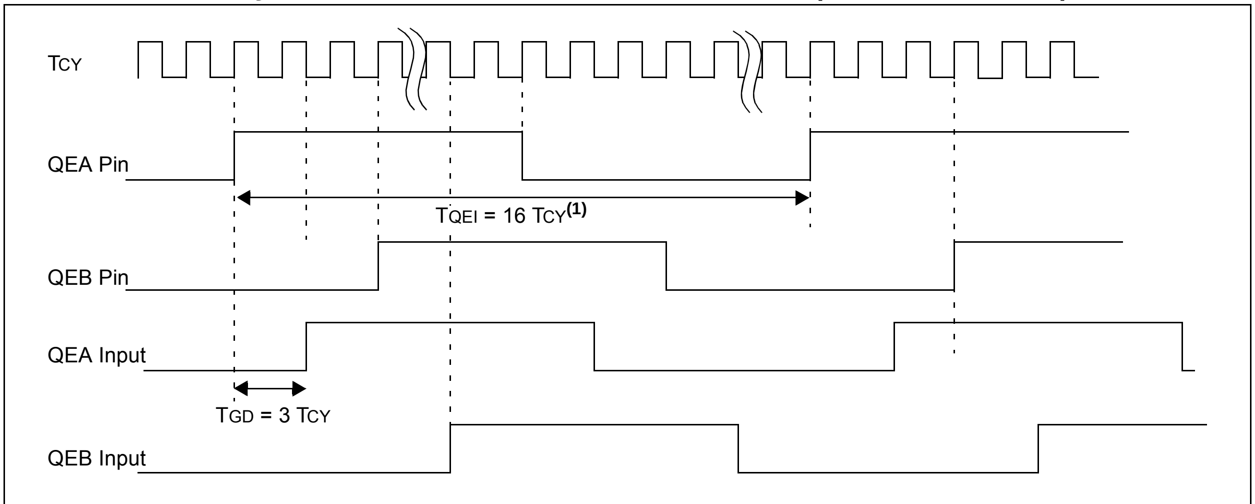
$$F_{POS} = \frac{4D \cdot RPM}{60}$$

Note:	The number of incremental lines in the position encoder is typically set at D = 1024 and the QEI Update mode = x4.

The maximum position count rate (i.e., x4 QEI Update mode, D = 1024) with FCY = 10 MIPS is equal to 2.5 MHz, which corresponds to FQEI of 625 kHz.

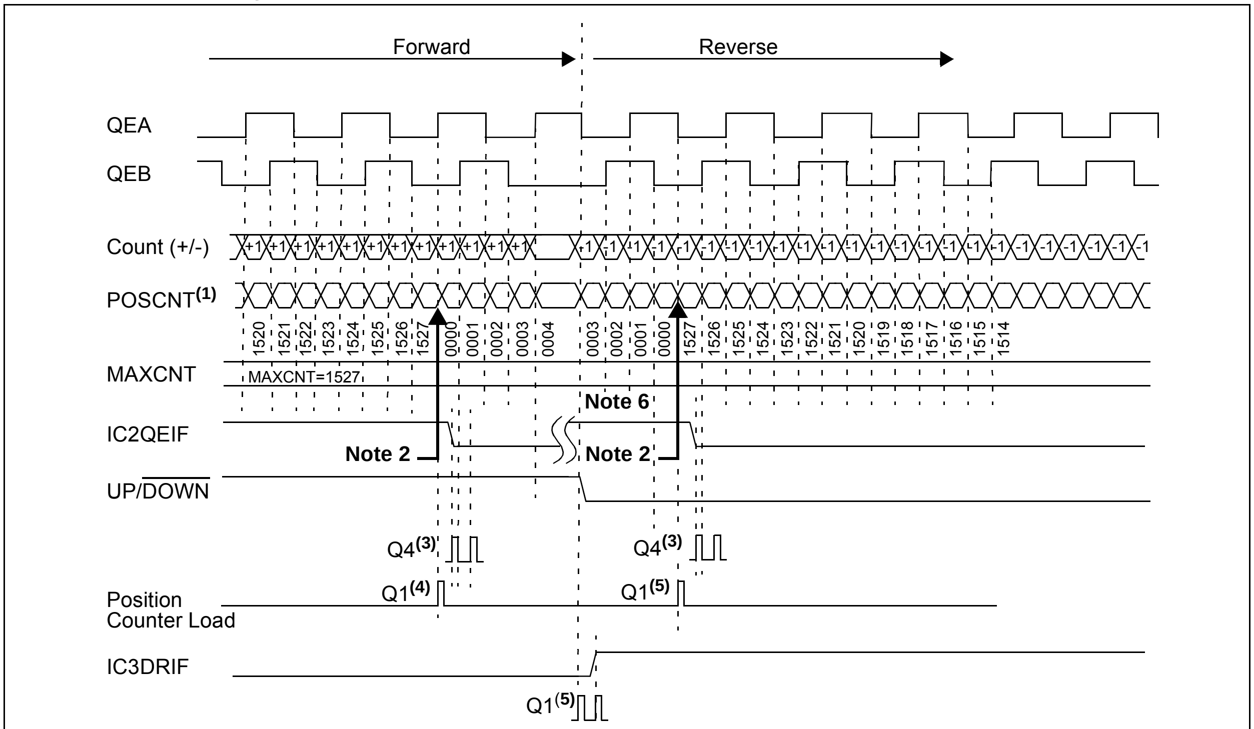
Figure 17-9 shows QEA and QEB quadrature input timing when sampled by the noise filter.

FIGURE 17-9: QEI INPUTS WHEN SAMPLED BY THE FILTER (DIVIDE RATIO = 1:1)



Note 1: The module design allows a quadrature frequency of up to FQEI = FCY/16.

FIGURE 17-10: QEI MODULE RESET TIMING ON PERIOD MATCH



Note 1: The POSCNT register is shown in QEI x4 Update mode (POSCNT increments on every rising and every falling edge of QEA and QEB input signals). Asynchronous external QEA and QEB inputs are synchronized to the TCY clock by the input sampling FF in the noise filter (see Figure 17-14).

2: When POSCNT = MAXCNT, POSCNT is reset to '0' on the next QEA rising edge. POSCNT is set to MAXCNT when POSCNT = 0 (when decrementing), which occurs on the next QEA falling edge.

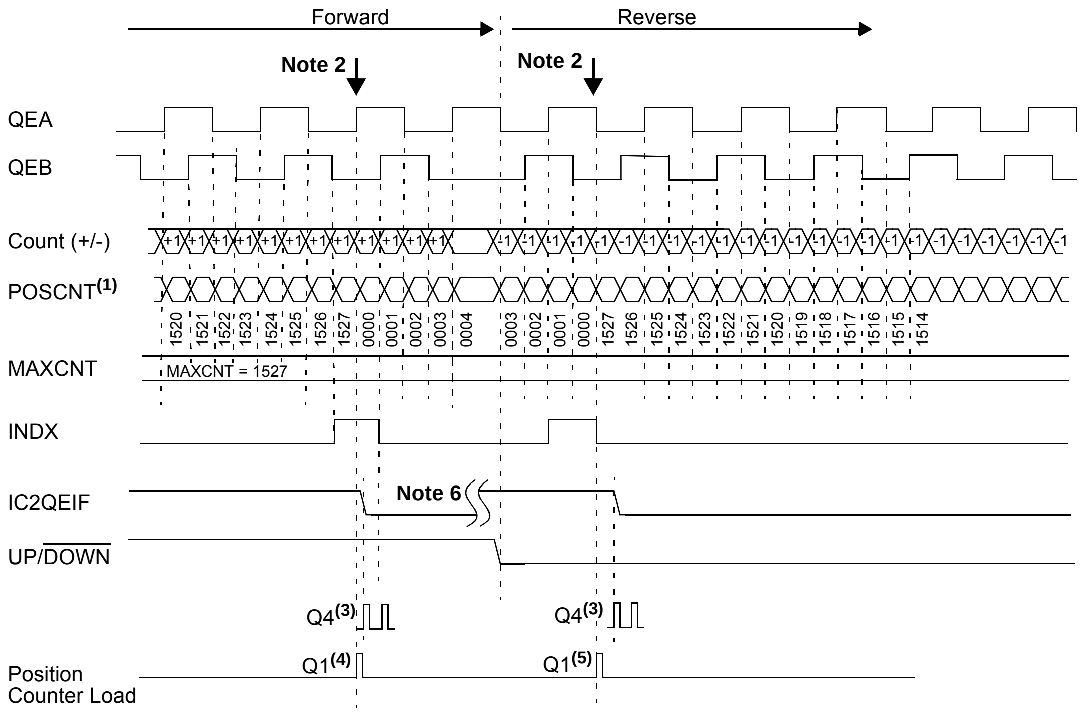
3: IC2QEIF is generated on the Q4 rising edge.

4: Position counter is loaded with '0' (which is a rollover event in this case) on POSCNT = MAXCNT.

5: Position counter is loaded with MAXCNT value (1527h) on underflow.

6: IC2QEIF must be cleared in software.

FIGURE 17-11: QEI MODULE RESET TIMING WITH THE INDEX INPUT



Note 1: POSCNT register is shown in QEI x4 Update mode (POSCNT increments on every rising and every falling edge of QEA and QEB input signals).

2: When an INDX Reset pulse is detected, POSCNT is reset to '0' on the next QEA or QEB edge. POSCNT is set to MAXCNT when POSCNT = 0 (when decrementing), which occurs on the next QEA or QEB edge. a similar Reset sequence occurs for the reverse direction, except that the INDX signal is recognized on its falling edge. The Reset is generated on the next QEA or QEB edge.

3: IC2QEIF is enabled for one TCY clock cycle.

4: The position counter is loaded with 0000h (i.e., Reset) on the next QEA or QEB edge when the INDX is high.

5: The position counter is loaded with a MAXCNT value (e.g., 1527h) on the next QEA or QEB edge following the INDX falling edge input signal detect).

6: IC2QEIF must be cleared in software.

17.2.6 VELOCITY MEASUREMENT

The velocity pulse generator, in conjunction with the IC1 and the synchronous TMR5 (in synchronous operation), provides a method for high accuracy speed measurements at both low and high mechanical motor speeds. The Velocity mode is enabled when the $\overline{\text{VELM}}$ bit is cleared (= `0`) and QEI is set to one of its operating modes (see Table 17-6).

To optimize register space, the Input Capture Channel 1 (IC1) is used to capture TMR5 counter values. Input Capture Buffer register, CAP1BUF, is redefined in Velocity Measurement mode, $\overline{\text{VELM}}$ = `0`, as the Velocity Register Buffer (VELRH, VELRL).

TABLE 17-6: VELOCITY PULSES

QEIM<2:0>	Velocity Event Mode
001 010	x2 Velocity Event mode. The velocity pulse is generated on every QEA edge.
101 110	x4 Velocity Event mode. The velocity pulse is generated on every QEA and QEB active edge.

17.2.6.1 Velocity Event Timing

The event pulses are reduced by a fixed ratio by the velocity pulse divider. The divider is useful for high-speed measurements where the velocity events happen frequently. By producing a single output pulse for a given number of input event pulses, the counter can track larger pulse counts (i.e., distance travelled) for a given time interval. Time is measured by utilizing the TMR5 time base.

Each velocity pulse serves as a capture pulse. With the TMR5 in Synchronous Timer mode, the value of TMR5 is captured on every output pulse of the postscaler. The counter is subsequently reset to '`0`'. TMR5 is reset upon a capture event.

Figure 17-13 shows the velocity measurement timing diagram.

FIGURE 17-12: VELOCITY MEASUREMENT BLOCK DIAGRAM

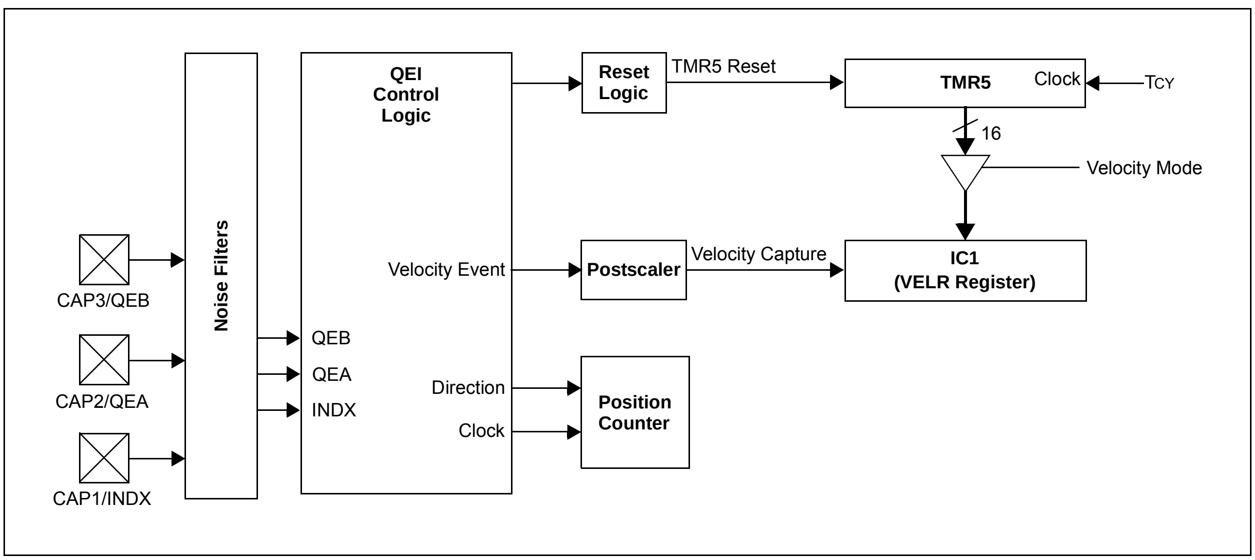
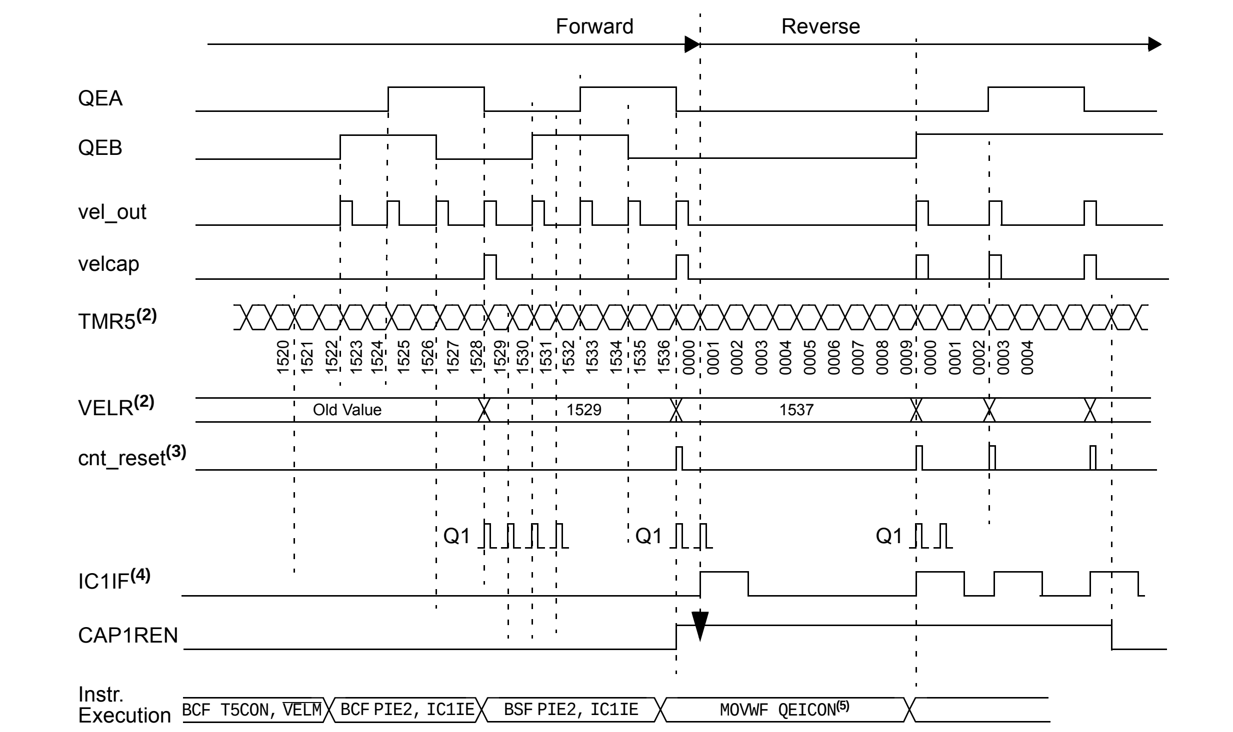


FIGURE 17-13: VELOCITY MEASUREMENT TIMING[1]



Note 1: Timing shown is for QEIM<2:0> = `101`, `110` or `111` (x4 Update mode enabled) and the velocity postscaler divide ratio is set to divide-by-4 (PDEC<1:0> = `01`).

2: The VELR register latches the TMR5 count on the "velcap" capture pulse. Timer5 must be set to the Synchronous Timer or Counter mode. In this example, it is set to the Synchronous Timer mode, where the TMR5 prescaler divide ratio = `1` (i.e., Timer5 Clock = TCY).

3: The TMR5 counter is reset on the next Q1 clock cycle following the "velcap" pulse. The TMR5 value is unaffected when the Velocity Measurement mode is first enabled (VELM = `0`). The velocity postscaler values must be reconfigured to their previous settings when re-entering Velocity Measurement mode. While making speed measurements of very slow rotational speeds (e.g., servo-controller applications), the Velocity Measurement mode may not provide sufficient precision. The Pulse-Width Measurement mode may have to be used to provide the additional precision. In this case, the input pulse is measured on the CAP1 input pin.

4: IC1IF interrupt is enabled by setting IC1IE as follows: `BSF PIE2, IC1IE`. Assume IC1E bit is placed in the PIE2 (Peripheral Interrupt Enable 2) register in the target device. The actual IC1IF bit is written on the Q2 rising edge.

5: The post decimation value is changed from PDEC = `01` (decimate by 4) to PDEC = `00` (decimate by 1).

17.2.6.2 Velocity Postscaler

The velocity event pulse (velcap, see Figure 17-12) serves as the TMR5 capture trigger to IC1 while in the Velocity mode. The number of velocity events are reduced by the velocity postscaler before they are used as the input capture clock. The velocity event reduction ratio can be set with the PDEC<1:0> control bits (QEICON<1:0>) to 1:4, 1:16, 1:64 or no reduction (1:1).

The velocity postscaler settings are automatically reloaded from their previous values as the Velocity mode is re-enabled.

17.2.6.3 CAP1REN in Velocity Mode

The TMR5 value can be reset (TMR5 register pair = 0000h) on a velocity event capture by setting the CAP1REN bit (CAP1CON<6>). When CAP1REN is cleared, the TMR5 time base will not be reset on any velocity event capture pulse. The VELR register pair, however, will continue to be updated with the current TMR5 value.

17.3 Noise Filters

The Motion Feedback Module includes three noise rejection filters on RA2/AN2/VREF-/CAP1/INDX, RA3/AN3/VREF+/CAP2/QEA and RA4/AN4/CAP3/QEB. The filter block also includes a fourth filter for the T5CKI pin. They are intended to help reduce spurious noise spikes which may cause the input signals to become corrupted at the inputs. The filter ensures that the input signals are not permitted to change until a stable value has been registered for three consecutive sampling clock cycles.

The filters are controlled using the Digital Filter Control (DFLTCON) register (see Register 17-3). The filters can be individually enabled or disabled by setting or clearing the corresponding FLTxEN bit in the DFLTCON register. The sampling frequency, which must be the same for all three noise filters, can be programmed by the FLTCK<2:0> Configuration bits. TCY is used as the clock reference to the clock divider block.

The noise filters can either be added or removed from the input capture, or QEI signal path, by setting or clearing the appropriate FLTxEN bit, respectively. Each capture channel provides for individual enable control of the filter output. The FLT4EN bit enables or disables the noise filter available on the T5CKI input in the Timer5 module.

The filter network for all channels is disabled on Power-on and Brown-out Resets, as the DFLTCON register is cleared on Resets. The operation of the filter is shown in the timing diagram in Figure 17-14.

REGISTER 17-3: DFLTCON: DIGITAL FILTER CONTROL REGISTER

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	FLT4EN	FLT3EN(1)	FLT2EN(1)	FLT1EN(1)	FLTCK2	FLTCK1	FLTCK0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as ‘0’

bit 6 **FLT4EN:** Noise Filter Output Enable bit (T5CKI input)
1 = Enabled
0 = Disabled

bit 5 **FLT3EN:** Noise Filter Output Enable bit (CAP3/QEB input)(1)
1 = Enabled
0 = Disabled

bit 4 **FLT2EN:** Noise Filter Output Enable bit (CAP2/QEA input)(1)
1 = Enabled
0 = Disabled

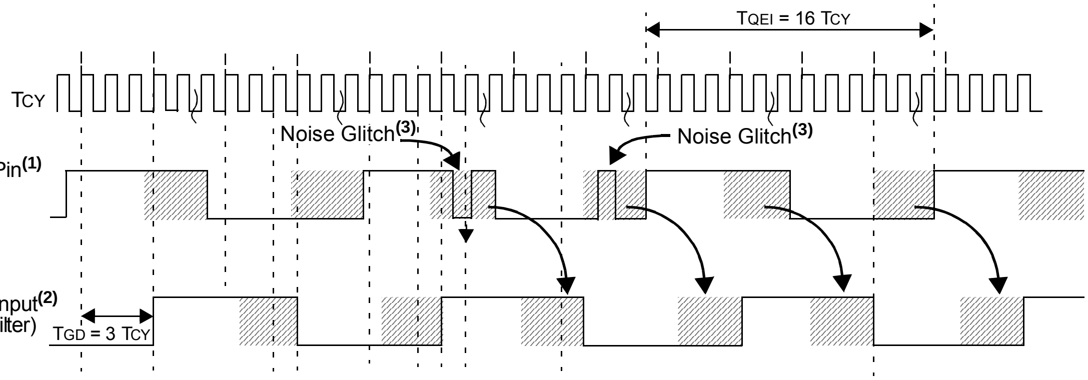
bit 3 **FLT1EN:** Noise Filter Output Enable bit (CAP1/INDX Input)(1)
1 = Enabled
0 = Disabled

bit 2-0 **FLTCK<2:0>:** Noise Filter Clock Divider Ratio bits
111 = Unused
110 = 1:128
101 = 1:64
100 = 1:32
011 = 1:16
010 = 1:4
001 = 1:2
000 = 1:1

Note 1: The noise filter output enables are functional in both QEI and IC Operating modes.

Note: The noise filter is intended for random high-frequency filtering and not continuous high-frequency filtering.

FIGURE 17-14: NOISE FILTER TIMING DIAGRAM (CLOCK DIVIDER = 1:1)



Note 1: Only the CAP1/INDX pin input is shown for simplicity. Similar event timing occurs on the CAP2/QEA and CAP3/QEB pins.

2: Noise filtering occurs in the shaded portions of the CAP1 input.

3: Filter's group delay: $T_{GD} = 3\ T_{CY}$.

17.4 IC and QEI Shared Interrupts

The IC and QEI submodules can each generate three distinct interrupt signals; however, they share the use of the same three interrupt flags in register, PIR3. The meaning of a particular interrupt flag at any given time depends on which module is active at the time the interrupt is set. The meaning of the flags in context are summarized in Table 17-7.

When the IC submodule is active, the three flags (IC1IF, IC2QEIF and IC3DRIF) function as interrupt-on-capture event flags for their respective input capture channels. The channel must be configured for one of the events that will generate an interrupt (see **Section 17.1.7 "IC Interrupts"** for more information).

When the QEI is enabled, the IC1IF interrupt flag indicates an interrupt caused by a velocity measurement event, usually an update of the VELR register. The IC2QEIF interrupt indicates that a position measurement event has occurred. IC3DRIF indicates that a direction change has been detected.

TABLE 17-7: MEANING OF IC AND QEI INTERRUPT FLAGS

Interrupt Flag	Meaning	
	IC Mode	QEI Mode
IC1IF	IC1 Capture Event	Velocity Register Update
IC2QEIF	IC2 Capture Event	Position Measurement Update
IC3DRIF	IC3 Capture Event	Direction Change

17.5 Operation in Sleep Mode

17.5.1 3x INPUT CAPTURE IN SLEEP MODE

Since the input capture can operate only when its time base is configured in a Synchronous mode, the input capture will not capture any events. This is because the device's internal clock has been stopped and any internal timers in Synchronous modes will not increment. The prescaler will continue to count the events (not synchronized).

When the specified capture event occurs, the CAPx interrupt will be set. The Capture Buffer register will be updated upon wake-up from sleep to the current TMR5 value. If the CAPx interrupt is enabled, the device will wake-up from Sleep. This effectively enables all input capture channels to be used as the external interrupts.

17.5.2 QEI IN SLEEP MODE

All QEI functions are halted in Sleep mode.

TABLE 17-8: REGISTERS ASSOCIATED WITH THE MOTION FEEDBACK MODULE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
IPR3	—	—	—	PTIP	IC3DRIP	IC2QEIP	IC1IP	TMR5IP	56
PIE3	—	—	—	PTIE	IC3DRIE	IC2QEIE	IC1IE	TMR5IE	56
PIR3	—	—	—	PTIF	IC3DRIF	IC2QEIF	IC1IF	TMR5IF	56
TMR5H	Timer5 Register High Byte								57
TMR5L	Timer5 Register Low Byte								57
PR5H	Timer5 Period Register High Byte								57
PR5L	Timer5 Period Register Low Byte								57
T5CON	T5SEN	RESEN	T5MOD	T5PS1	T5PS0	T5SYNC	TMR5CS	TMR5ON	57
CAP1BUFH/ VELRH	Capture 1 Register High Byte/Velocity Register High Byte(1)								58
CAP1BUFL/ VELRL	Capture 1 Register Low Byte/Velocity Register Low Byte(1)								58
CAP2BUFH/ POSCNTH	Capture 2 Register High Byte/QEI Position Counter Register High Byte(1)								58
CAP2BUFL/ POSCNTL	Capture 2 Register Low Byte/QEI Position Counter Register Low Byte(1)								58
CAP3BUFH/ MAXCNTH	Capture 3 Register High Byte/QEI Max. Count Limit Register High Byte(1)								58
CAP3BUFL/ MAXCNTL	Capture 3 Register Low Byte/QEI Max. Count Limit Register Low Byte(1)								58
CAP1CON	—	CAP1REN	—	—	CAP1M3	CAP1M2	CAP1M1	CAP1M0	59
CAP2CON	—	CAP2REN	—	—	CAP2M3	CAP2M2	CAP2M1	CAP2M0	59
CAP3CON	—	CAP3REN	—	—	CAP3M3	CAP3M2	CAP3M1	CAP3M0	59
DFLTCON	—	FLT4EN	FLT3EN	FLT2EN	FLT1EN	FLTCK2	FLTCK1	FLTCK0	59
QEICON	VELM	QERR	UP/DOWN	QEIM2	QEIM1	QEIM0	PDEC1	PDEC0	56

Legend: — = unimplemented. Shaded cells are not used by the Motion Feedback Module.

Note 1: Register name and function determined by which submodule is selected (IC/QEI, respectively). See **Section 17.1.10 "Other Operating Modes"** for more information.

NOTES:

18.0 POWER CONTROL PWM MODULE

The Power Control PWM module simplifies the task of generating multiple, synchronized Pulse-Width Modulated (PWM) outputs for use in the control of motor controllers and power conversion applications. In particular, the following power and motion control applications are supported by the PWM module:

- Three-Phase and Single-Phase AC Induction Motors
- Switched Reluctance Motors
- Brushless DC (BLDC) Motors
- Uninterruptible Power Supplies (UPS)
- Multiple DC Brush Motors

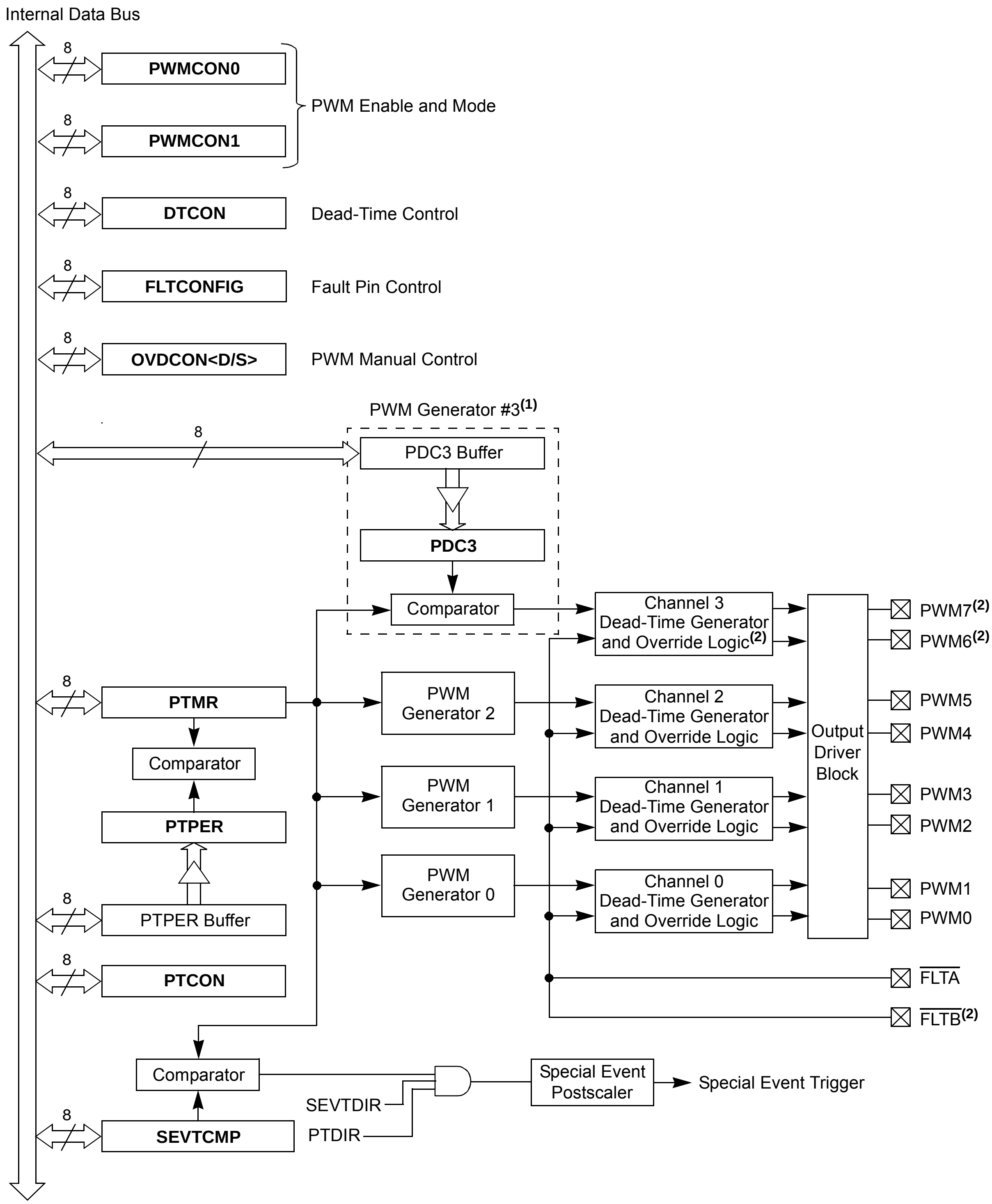
The PWM module has the following features:

- Up to eight PWM I/O pins with four duty cycle generators. Pins can be paired to get a complete half-bridge control.
- Up to 14-bit resolution, depending upon the PWM period.
- “On-the-fly” PWM frequency changes.
- Edge and Center-Aligned Output modes.
- Single-Pulse Generation mode.
- Programmable dead-time control between paired PWMs.
- Interrupt support for asymmetrical updates in Center-Aligned mode.
- Output override for Electrically Commutated Motor (ECM) operation; for example, BLDC.
- Special Event Trigger comparator for scheduling other peripheral events.
- PWM outputs disable feature sets PWM outputs to their inactive state when in Debug mode.

The Power Control PWM module supports three PWM generators and six output channels on PIC18F2331/2431 devices, and four generators and eight channels on PIC18F4331/4431 devices. A simplified block diagram of the module is shown in Figure 18-1. Figure 18-2 and Figure 18-3 show how the module hardware is configured for each PWM output pair for the Complementary and Independent Output modes.

Each functional unit of the PWM module will be discussed in subsequent sections.

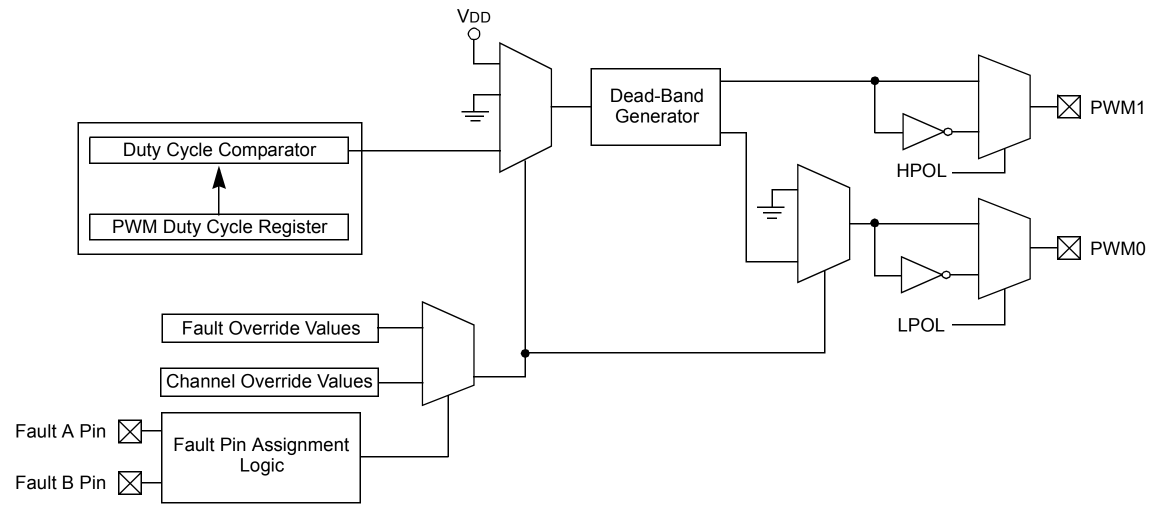
FIGURE 18-1: POWER CONTROL PWM MODULE BLOCK DIAGRAM



Note 1: Only PWM Generator 3 is shown in detail. The other generators are identical; their details are omitted for clarity.

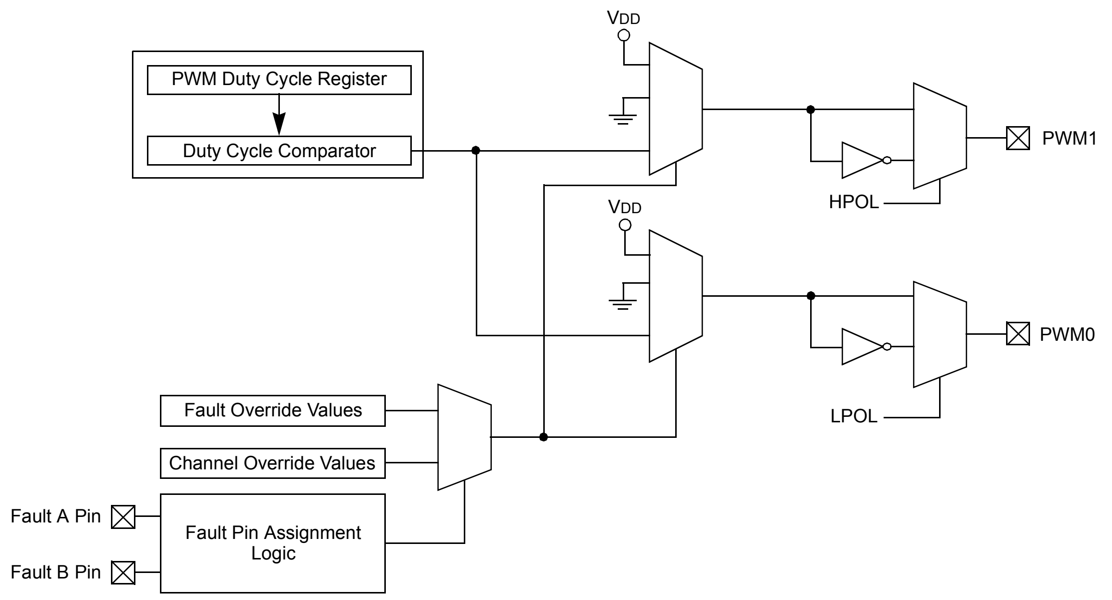
2: PWM Generator 3 and its logic, PWM Channels 6 and 7, and FLTB and its associated logic are not implemented on PIC18F2331/2431 devices.

FIGURE 18-2: PWM MODULE BLOCK DIAGRAM, ONE OUTPUT PAIR, COMPLEMENTARY MODE



Note: In Complementary mode, the even channel cannot be forced active by a Fault or override event when the odd channel is active. The even channel is always the complement of the odd channel and is inactive, with dead time inserted, before the odd channel is driven to its active state.

FIGURE 18-3: PWM MODULE BLOCK DIAGRAM, ONE OUTPUT PAIR, INDEPENDENT MODE



This module contains four duty cycle generators, numbered 0 through 3. The module has eight PWM output pins, numbered 0 through 7. The eight PWM outputs are grouped into output pairs of even and odd numbered outputs. In Complementary modes, the even PWM pins must always be the complement of the corresponding odd PWM pin. For example, PWM0 will be the complement of PWM1, PWM2 will be the complement of PWM3 and so on. The dead-time generator inserts an OFF period called "dead time" between the going OFF of one pin to the going ON of the complementary pin of the paired pins. This is to prevent damage to the power switching devices that will be connected to the PWM output pins.

The time base for the PWM module is provided by its own 12-bit timer, which also incorporates selectable prescaler and postscaler options.

18.1 Control Registers

The operation of the PWM module is controlled by a total of 22 registers. Eight of these are used to configure the features of the module:

- PWM Timer Control Register 0 (PTCON0)
- PWM Timer Control Register 1 (PTCON1)
- PWM Control Register 0 (PWMCON0)
- PWM Control Register 1 (PWMCON1)
- Dead-Time Control Register (DTCON)
- Output Override Control Register (OVDCOND)
- Output State Register (OVDCONS)
- Fault Configuration Register (FLTCONFIG)

There are also 14 registers that are configured as seven register pairs of 16 bits. These are used for the configuration values of specific features. They are:

- PWM Time Base Registers (PTMRH and PTMRL)
- PWM Time Base Period Registers (PTPERH and PTPERL)
- PWM Special Event Trigger Compare Registers (SEVTCMPH and SEVTCMPL)
- PWM Duty Cycle #0 Registers (PDC0H and PDC0L)
- PWM Duty Cycle #1 Registers (PDC1H and PDC1L)
- PWM Duty Cycle #2 Registers (PDC2H and PDC2L)
- PWM Duty Cycle #3 Registers (PDC3H and PDC3L)

All of these register pairs are double-buffered.

18.2 Module Functionality

The PWM module supports several modes of operation that are beneficial for specific power and motor control applications. Each mode of operation is described in subsequent sections.

The PWM module is composed of several functional blocks. The operation of each is explained separately in relation to the several modes of operation:

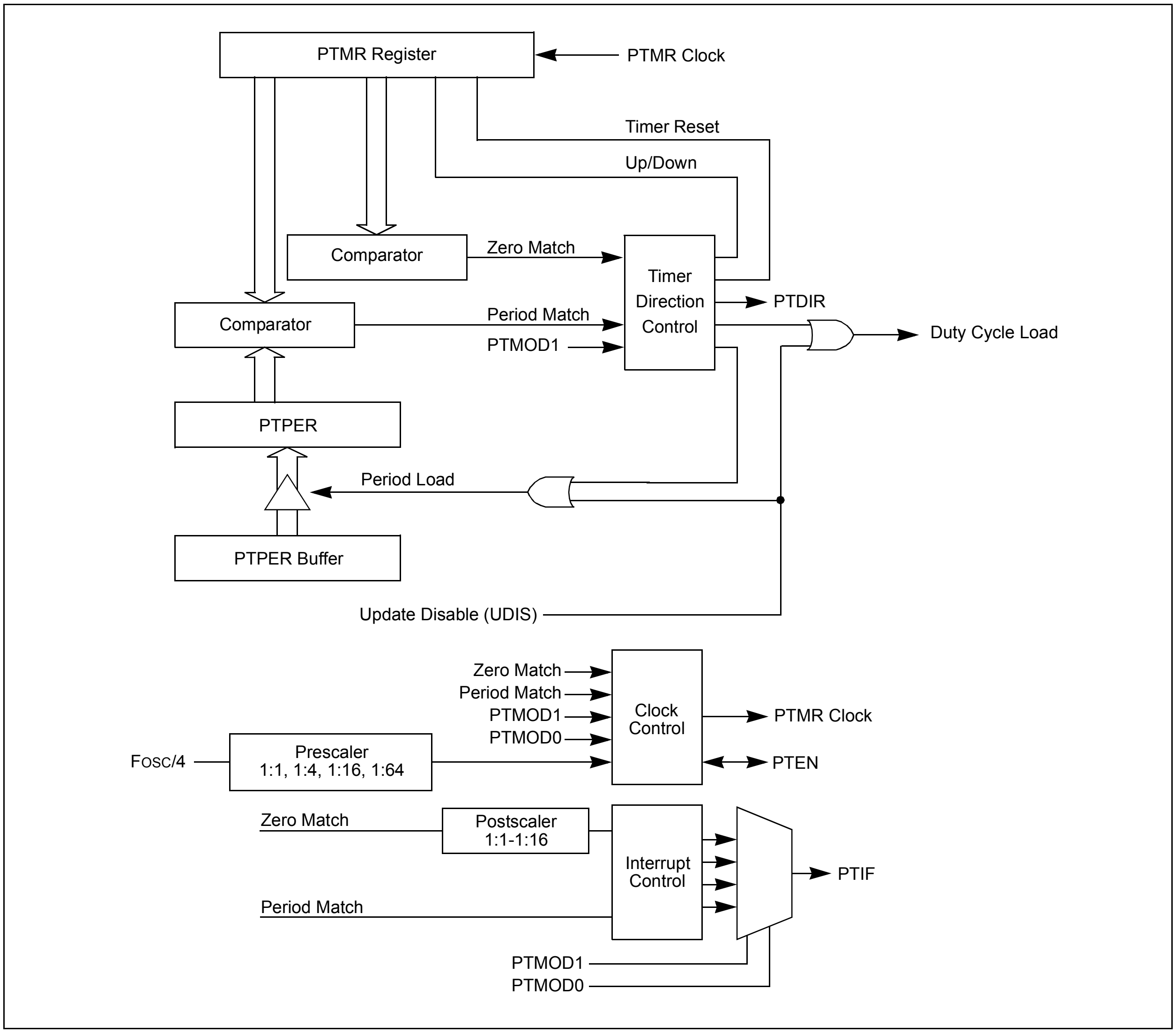
- PWM Time Base
- PWM Time Base Interrupts
- PWM Period
- PWM Duty Cycle
- Dead-Time Generators
- PWM Output Overrides
- PWM Fault Inputs
- PWM Special Event Trigger

18.3 PWM Time Base

The PWM time base is provided by a 12-bit timer with prescaler and postscaler functions. A simplified block diagram of the PWM time base is shown in Figure 18-4. The PWM time base is configured through the PTCON0 and PTCON1 registers. The time base is enabled or disabled by respectively setting or clearing the PTEN bit in the PTCON1 register.

Note:	The PTMR register pair (PTMRL:PTMRH) is not cleared when the PTEN bit is cleared in software.

FIGURE 18-4: PWM TIME BASE BLOCK DIAGRAM



The PWM time base can be configured for four different modes of operation:

- Free-Running mode
- Single-Shot mode
- Continuous Up/Down Count mode
- Continuous Up/Down Count mode with interrupts for double updates

These four modes are selected by the PTMOD<1:0> bits in the PTCON0 register. The Free-Running mode produces edge-aligned PWM generation. The Continuous Up/Down Count modes produce center-aligned PWM generation. The Single-Shot mode allows the PWM module to support pulse control of certain Electronically Commutated Motors (ECMs) and produces edge-aligned operation.

REGISTER 18-1: PTCON0: PWM TIMER CONTROL REGISTER 0

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTOPS3	PTOPS2	PTOPS1	PTOPS0	PTCKPS1	PTCKPS0	PTMOD1	PTMOD0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-4 **PTOPS<3:0>:** PWM Time Base Output Postscale Select bits
`0000` = 1:1 Postscale
`0001` = 1:2 Postscale
.
.
.
`1111` = 1:16 Postscale

bit 3-2 **PTCKPS<1:0>:** PWM Time Base Input Clock Prescale Select bits
`00` = PWM time base input clock is FOSC/4 (1:1 prescale)
`01` = PWM time base input clock is FOSC/16 (1:4 prescale)
`10` = PWM time base input clock is FOSC/64 (1:16 prescale)
`11` = PWM time base input clock is FOSC/256 (1:64 prescale)

bit 1-0 **PTMOD<1:0>:** PWM Time Base Mode Select bits
`11` = PWM time base operates in a Continuous Up/Down Count mode with interrupts for double PWM updates
`10` = PWM time base operates in a Continuous Up/Down Count mode
`01` = PWM time base configured for Single-Shot mode
`00` = PWM time base operates in a Free-Running mode

REGISTER 18-2: PTCON1: PWM TIMER CONTROL REGISTER 1

R/W-0	R-0	U-0	U-0	U-0	U-0	U-0	U-0
PTEN	PTDIR	—	—	—	—	—	—
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **PTEN:** PWM Time Base Timer Enable bit
`1` = PWM time base is on
`0` = PWM time base is off

bit 6 **PTDIR:** PWM Time Base Count Direction Status bit
`1` = PWM time base counts down
`0` = PWM time base counts up

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

REGISTER 18-3: PWMCON0: PWM CONTROL REGISTER 0

U-0	R/W-1[1]	R/W-1[1]	R/W-1[1]	R/W-0	R/W-0	R/W-0	R/W-0
—	PWMEN2	PWMEN1	PWMEN0	PMOD3[3]	PMOD2	PMOD1	PMOD0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

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

bit 6-4 **PWMEN<2:0>:** PWM Module Enable bits[1]

- 111 = All odd PWM I/O pins are enabled for PWM output[2]
- 110 = PWM1, PWM3 pins are enabled for PWM output
- 101 = All PWM I/O pins are enabled for PWM output[2]
- 100 = PWM0, PWM1, PWM2, PWM3, PWM4 and PWM5 pins are enabled for PWM output
- 011 = PWM0, PWM1, PWM2 and PWM3 I/O pins are enabled for PWM output
- 010 = PWM0 and PWM1 pins are enabled for PWM output
- 001 = PWM1 pin is enabled for PWM output
- 000 = PWM module is disabled; all PWM I/O pins are general purpose I/O

bit 3-0 **PMOD<3:0>:** PWM Output Pair Mode bits

For PMOD0:
- 1 = PWM I/O pin pair (PWM0, PWM1) is in the Independent mode
- 0 = PWM I/O pin pair (PWM0, PWM1) is in the Complementary mode

For PMOD1:
- 1 = PWM I/O pin pair (PWM2, PWM3) is in the Independent mode
- 0 = PWM I/O pin pair (PWM2, PWM3) is in the Complementary mode

For PMOD2:
- 1 = PWM I/O pin pair (PWM4, PWM5) is in the Independent mode
- 0 = PWM I/O pin pair (PWM4, PWM5) is in the Complementary mode

For PMOD3:[3]
- 1 = PWM I/O pin pair (PWM6, PWM7) is in the Independent mode
- 0 = PWM I/O pin pair (PWM6, PWM7) is in the Complementary mode

Note 1: Reset condition of the PWMEN bits depends on the PWMPIN Configuration bit.

2: When PWMEN<2:0> = 101, PWM<5:0> outputs are enabled for PIC18F2331/2431 devices; PWM<7:0> outputs are enabled for PIC18F4331/4431 devices.
When PWMEN<2:0> = 111, PWM Outputs 1, 3 and 5 are enabled in PIC18F2331/2431 devices; PWM Outputs 1, 3, 5 and 7 are enabled in PIC18F4331/4431 devices.

3: Unimplemented in PIC18F2331/2431 devices; maintain these bits clear.

REGISTER 18-4: PWMCON1: PWM CONTROL REGISTER 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
SEVOPS3	SEVOPS2	SEVOPS1	SEVOPS0	SEVTDIR	—	UDIS	OSYNC
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-4 **SEVOPS<3:0>:** PWM Special Event Trigger Output Postscale Select bits
`0000` = 1:1 Postscale
`0001` = 1:2 Postscale
.
.
.
`1111` = 1:16 Postscale

bit 3 **SEVTDIR:** Special Event Trigger Time Base Direction bit
`1` = A Special Event Trigger will occur when the PWM time base is counting downwards
`0` = A Special Event Trigger will occur when the PWM time base is counting upwards

bit 2 **Unimplemented:** Read as ‘0’

bit 1 **UDIS:** PWM Update Disable bit
`1` = Updates from Duty Cycle and Period Buffer registers are disabled
`0` = Updates from Duty Cycle and Period Buffer registers are enabled

bit 0 **OSYNC:** PWM Output Override Synchronization bit
`1` = Output overrides via the OVDCON register are synchronized to the PWM time base
`0` = Output overrides via the OVDCON register are asynchronous

18.3.1 FREE-RUNNING MODE

In the Free-Running mode, the PWM Time Base registers (PTMRL and PTMRH) will begin counting upwards until the value in the PWM Time Base Period register, PTPER (PTPERL and PTPERH), is matched. The PTMR registers will be reset on the following input clock edge and the time base will continue counting upwards as long as the PTEN bit remains set.

18.3.2 SINGLE-SHOT MODE

In the Single-Shot mode, the PWM time base will begin counting upwards when the PTEN bit is set. When the value in the PTMR register matches the PTPER register, the PTMR register will be reset on the following input clock edge and the PTEN bit will be cleared by the hardware to halt the time base.

18.3.3 CONTINUOUS UP/DOWN COUNT MODES

In Continuous Up/Down Count modes, the PWM time base counts upwards until the value in the PTPER register matches with the PTMR register. On the following input clock edge, the timer counts downwards. The PTDIR bit in the PTCON1 register is read-only and indicates the counting direction. The PTDIR bit is set when the timer counts downwards.

Note:	Since the PWM compare outputs are driven to the active state when the PWM time base is counting downwards and matches the duty cycle value, the PWM outputs are held inactive during the first half of the first period of the Continuous Up/Down Count mode until PTMR begins to count down from the PTPER value.

18.3.4 PWM TIME BASE PRESCALER

The input clock to PTMR (FOSC/4) has prescaler options of 1:1, 1:4, 1:16 or 1:64. These are selected by control bits, PTCKPS<1:0>, in the PTCON0 register. The prescaler counter is cleared when any of the following occurs:

- Write to the PTMR register
- Write to the PTCON (PTCON0 or PTCON1) register
- Any device Reset

Note:	The PTMR register is not cleared when PTCONx is written.

Table 18-1 shows the minimum PWM frequencies that can be generated with the PWM time base and the prescaler. An operating frequency of 40 MHz (FCYC = 10 MHz) and PTPER = 0xFFF is assumed in the table. The PWM module must be capable of generating PWM signals at the line frequency (50 Hz or 60 Hz) for certain power control applications.

TABLE 18-1: MINIMUM PWM FREQUENCY

Minimum PWM Frequencies vs. Prescaler Value for FCYC = 10 MIPS (PTPER = 0FFFh)		
Prescale	**PWM Frequency Edge-Aligned**	**PWM Frequency Center-Aligned**
1:1	2441 Hz	1221 Hz
1:4	610 Hz	305 Hz
1:16	153 Hz	76 Hz
1:64	38 Hz	19 Hz

18.3.5 PWM TIME BASE POSTSCALER

The match output of PTMR can optionally be postscaled through a 4-bit postscaler (which gives a 1:1 to 1:16 scaling inclusive) to generate an interrupt. The postscaler counter is cleared when any of the following occurs:

- Write to the PTMR register
- Write to the PTCON register
- Any device Reset

The PTMR register is not cleared when PTCON is written.

18.4 PWM Time Base Interrupts

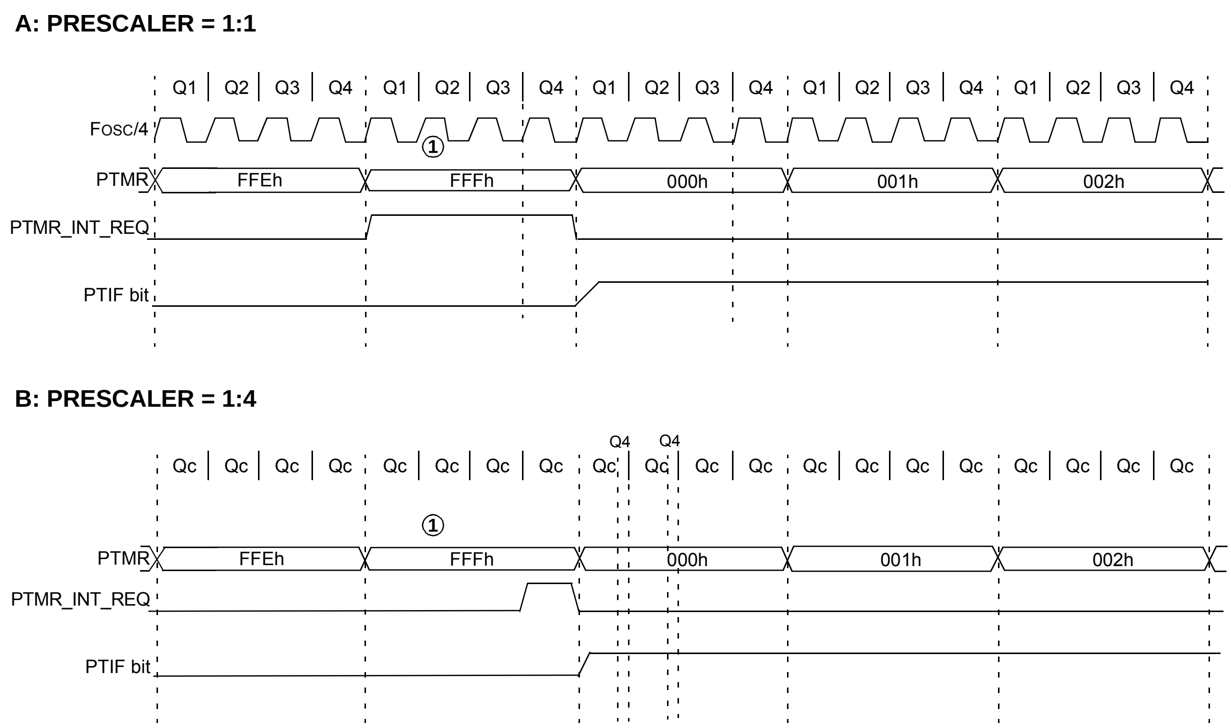
The PWM timer can generate interrupts based on the modes of operation selected by the PTMOD<1:0> bits and the postscaler bits (PTOPS<3:0>).

18.4.1 INTERRUPTS IN FREE-RUNNING MODE

When the PWM time base is in the Free-Running mode (PTMOD<1:0> = 00), an interrupt event is generated each time a match with the PTPER register occurs. The PTMR register is reset to zero in the following clock edge.

Using a postscaler selection other than 1:1 will reduce the frequency of interrupt events.

FIGURE 18-5: PWM TIME BASE INTERRUPT TIMING, FREE-RUNNING MODE



Note 1: PWM Time Base Period register, PTPER, is loaded with the value, FFFh, for this example.

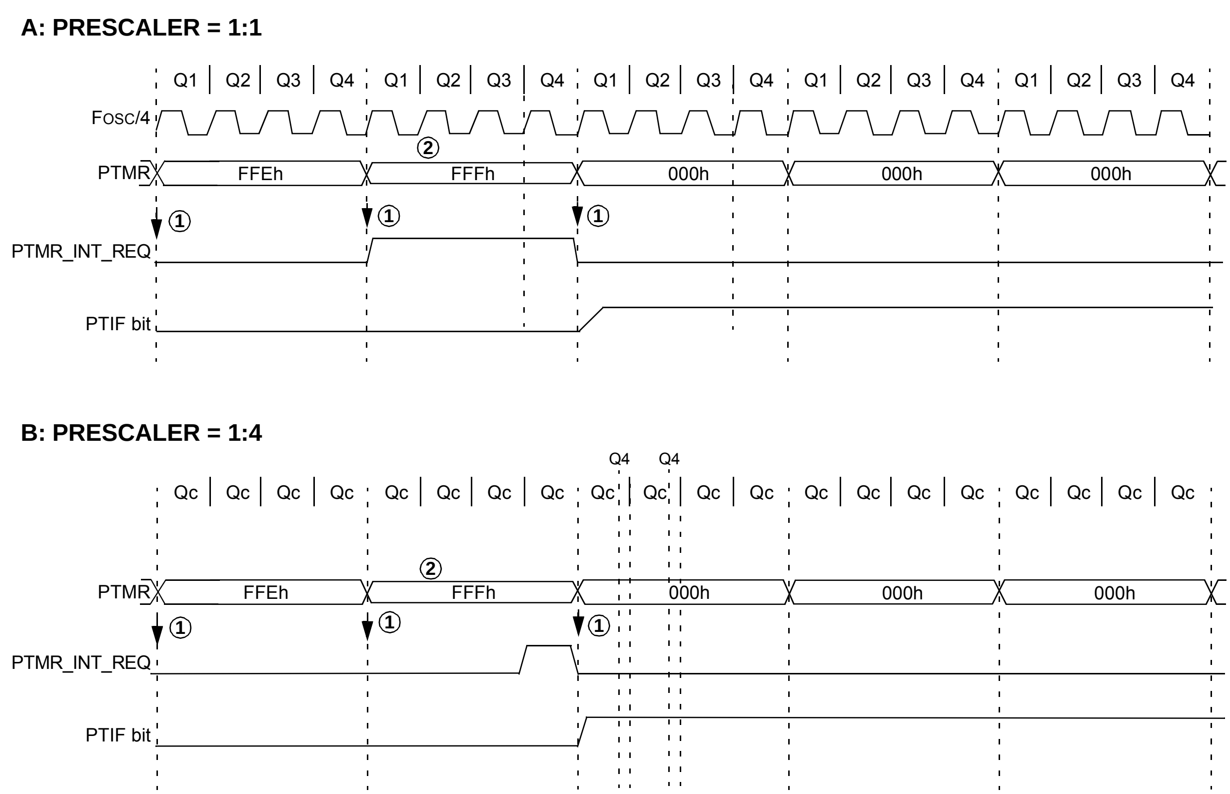
18.4.2 INTERRUPTS IN SINGLE-SHOT MODE

When the PWM time base is in the Single-Shot mode (PTMOD<1:0> = `01`), an interrupt event is generated when a match with the PTPER register occurs. The PWM Time Base register (PTMR) is reset to zero on the following input clock edge and the PTEN bit is cleared. The postscaler selection bits have no effect in this Timer mode.

18.4.3 INTERRUPTS IN CONTINUOUS UP/DOWN COUNT MODE

In the Continuous Up/Down Count mode (PTMOD<1:0> = `10`), an interrupt event is generated each time the value of the PTMR register becomes zero and the PWM time base begins to count upwards. The postscaler selection bits may be used in this mode of the timer to reduce the frequency of the interrupt events. Figure 18-7 shows the interrupts in Continuous Up/Down Count mode.

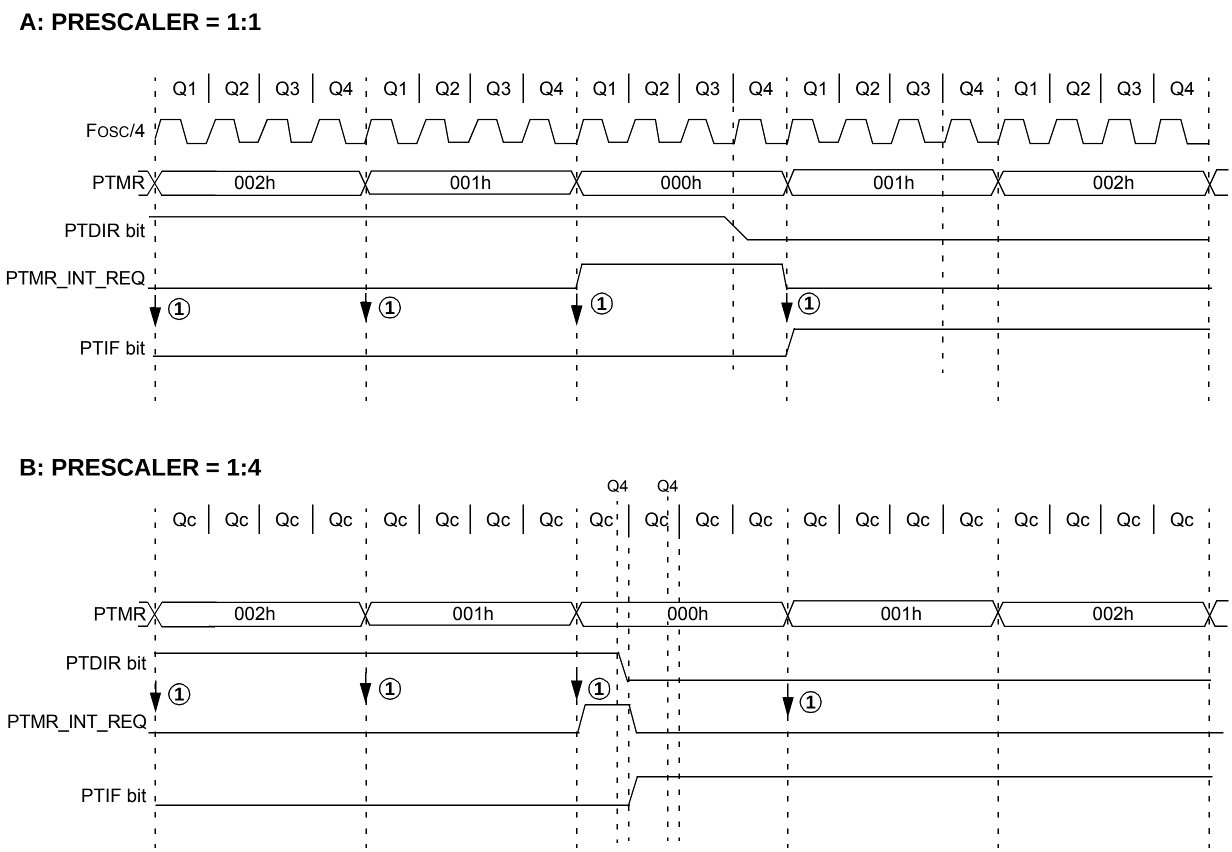
FIGURE 18-6: PWM TIME BASE INTERRUPT TIMING, SINGLE-SHOT MODE



Note 1: Interrupt flag bit, PTIF, is sampled here (every Q1).

2: PWM Time Base Period register, PTPER, is loaded with the value, FFFh, for this example.

FIGURE 18-7: PWM TIME BASE INTERRUPT, CONTINUOUS UP/DOWN COUNT MODE



Note 1: Interrupt flag bit, PTIF, is sampled here (every Q1).

18.4.4 INTERRUPTS IN DOUBLE UPDATE MODE

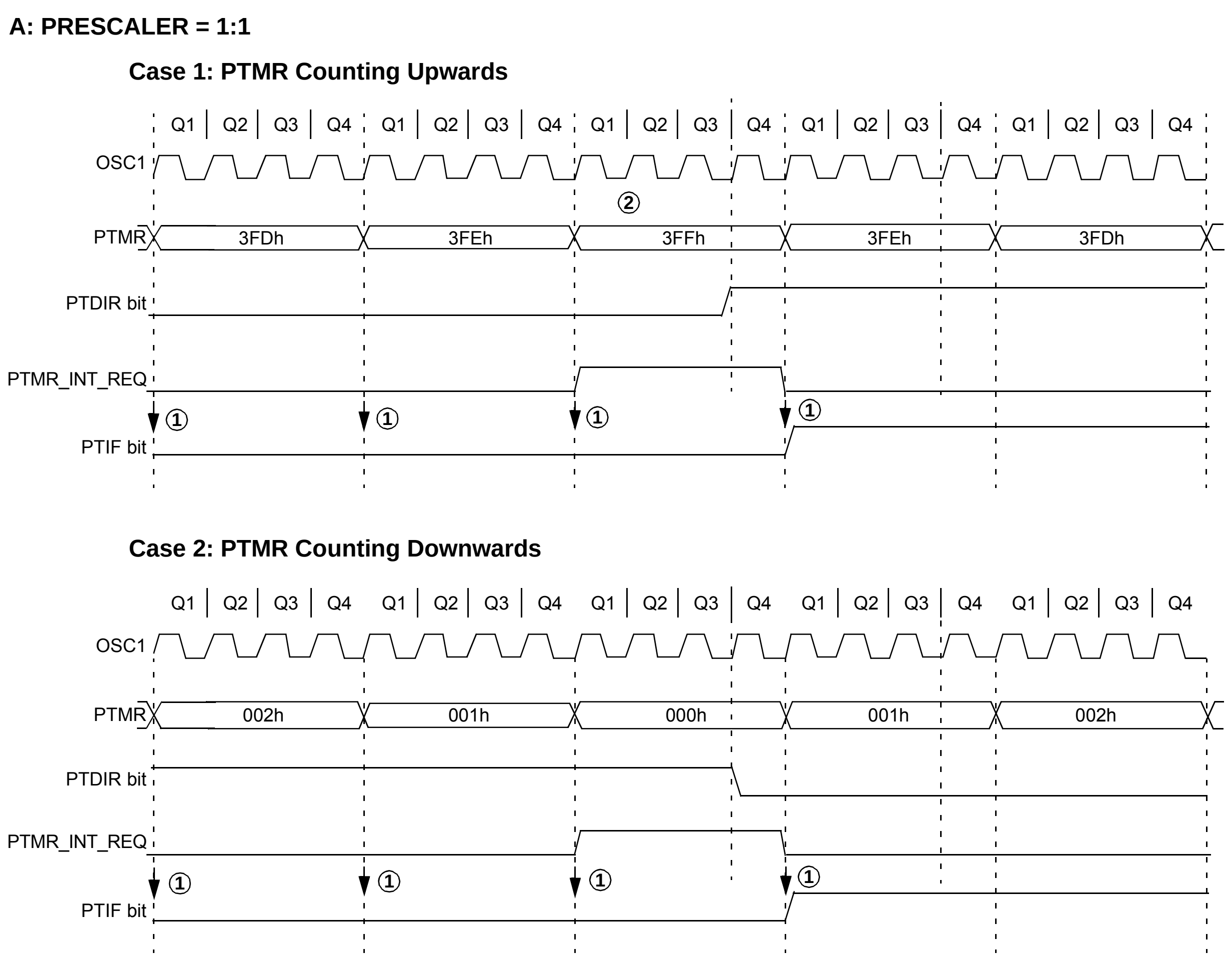
This mode is available in Continuous Up/Down Count mode. In the Double Update mode (PTMOD<1:0> = 11), an interrupt event is generated each time the PTMR register is equal to zero and each time the PTMR matches with PTPER register. Figure 18-8 shows the interrupts in Continuous Up/Down Count mode with double updates.

The Double Update mode provides two additional functions to the user in Center-Aligned mode.

1. The control loop bandwidth is doubled because the PWM duty cycles can be updated twice per period.
2. Asymmetrical center-aligned PWM waveforms can be generated, which are useful for minimizing output waveform distortion in certain motor control applications.

Note:	Do not change the PTMOD bits while PTEN is active; it will yield unexpected results. To change the PWM Timer mode of operation, first clear the PTEN bit, load the PTMOD bits with the required data and then set PTEN.

FIGURE 18-8: PWM TIME BASE INTERRUPT, CONTINUOUS UP/DOWN COUNT MODE WITH DOUBLE UPDATES



Note 1: Interrupt flag bit, PTIF, is sampled here (every Q1).

2: PWM Time Base Period register, PTPER, is loaded with the value, 3FFh, for this example.

18.5 PWM Period

The PWM period is defined by the PTPER register pair (PTPERL and PTPERH). The PWM period has 12-bit resolution by combining 4 LSBs of PTPERH and 8 bits of PTPERL. PTPER is a double-buffered register used to set the counting period for the PWM time base.

The PTPER register contents are loaded into the PTPER register at the following times:

- Free-Running and Single-Shot modes: When the PTMR register is reset to zero after a match with the PTPER register.
- Continuous Up/Down Count modes: When the PTMR register is zero. The value held in the PTPER register is automatically loaded into the PTPER register when the PWM time base is disabled (PTEN = 0). Figure 18-9 and Figure 18-10 indicate the times when the contents of the PTPER register are loaded into the actual PTPER register.

The PWM period can be calculated from the following formulas:

EQUATION 18-1: PWM PERIOD FOR FREE-RUNNING MODE

$$T_{PWM} = \frac{(PTPER + 1) \times PTMRPS}{F_{OSC}/4}$$

EQUATION 18-2: PWM PERIOD FOR UP/DOWN COUNT MODE

$$T_{PWM} = \frac{(2 \times PTPER) \times PTMRPS}{\frac{F_{OSC}}{4}}$$

The PWM frequency is the inverse of period; or:

EQUATION 18-3: PWM FREQUENCY

$$\text{PWM Frequency} = \frac{1}{\text{PWM Period}}$$

The maximum resolution (in bits) for a given device oscillator and PWM frequency can be determined from the following formula:

EQUATION 18-4: PWM RESOLUTION

$$\text{Resolution} = \frac{\log\left(\frac{F_{OSC}}{F_{PWM}}\right)}{\log(2)}$$

The PWM resolutions and frequencies are shown for a selection of execution speeds and PTPER values in Table 18-2. The PWM frequencies in Table 18-2 are calculated for Edge-Aligned PWM mode. For Center-Aligned mode, the PWM frequencies will be approximately one-half the values indicated in this table.

TABLE 18-2: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS

PWM Frequency = 1/TPWM				
Fosc	MIPS	PTPER Value	PWM Resolution	PWM Frequency
40 MHz	10	0FFFh	14 bits	2.4 kHz
40 MHz	10	07FFh	13 bits	4.9 kHz
40 MHz	10	03FFh	12 bits	9.8 kHz
40 MHz	10	01FFh	11 bits	19.5 kHz
40 MHz	10	FFh	10 bits	39.0 kHz
40 MHz	10	7Fh	9 bits	78.1 kHz
40 MHz	10	3Fh	8 bits	156.2 kHz
40 MHz	10	1Fh	7 bits	312.5 kHz
40 MHz	10	0Fh	6 bits	625 kHz
25 MHz	6.25	0FFFh	14 bits	1.5 kHz
25 MHz	6.25	03FFh	12 bits	6.1 kHz
25 MHz	6.25	FFh	10 bits	24.4 kHz
10 MHz	2.5	0FFFh	14 bits	610 Hz
10 MHz	2.5	03FFh	12 bits	2.4 kHz
10 MHz	2.5	FFh	10 bits	9.8 kHz
5 MHz	1.25	0FFFh	14 bits	305 Hz
5 MHz	1.25	03FFh	12 bits	1.2 kHz
5 MHz	1.25	FFh	10 bits	4.9 kHz
4 MHz	1	0FFFh	14 bits	244 Hz
4 MHz	1	03FFh	12 bits	976 Hz
4 MHz	1	FFh	10 bits	3.9 kHz

Note: For center-aligned operation, PWM frequencies will be approximately 1/2 the value indicated in the table.

FIGURE 18-9: PWM PERIOD BUFFER UPDATES IN FREE-RUNNING MODE

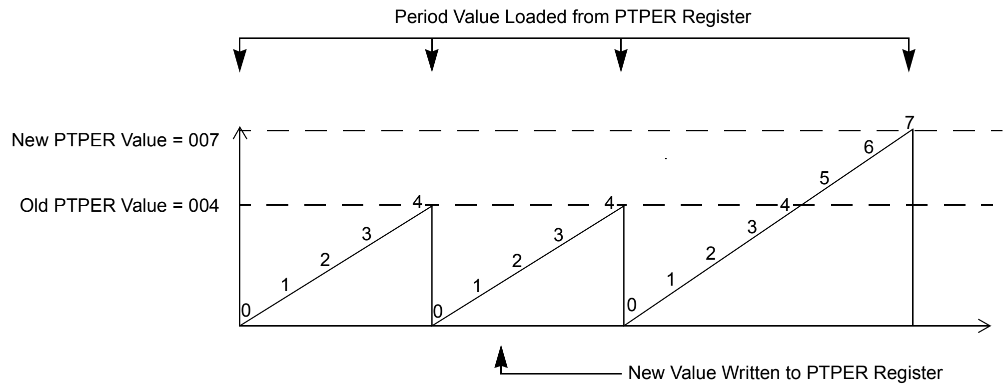
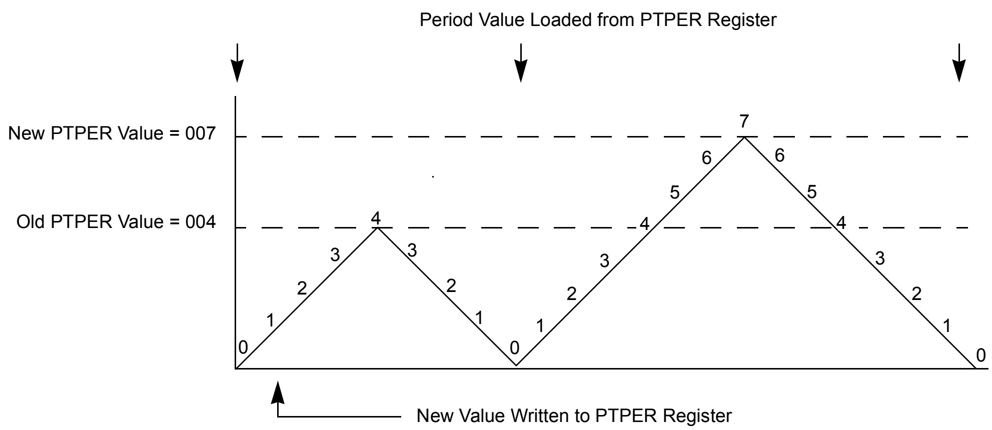


FIGURE 18-10: PWM PERIOD BUFFER UPDATES IN CONTINUOUS UP/DOWN COUNT MODE



18.6 PWM Duty Cycle

PWM duty cycle is defined by the PDCx (PDCxL and PDCxH) registers. There are a total of four PWM Duty Cycle registers for four pairs of PWM channels. The Duty Cycle registers have 14-bit resolution by combining six LSbs of PDCxH with the 8 bits of PDCxL. PDCx is a double-buffered register used to set the counting period for the PWM time base.

18.6.1 PWM DUTY CYCLE REGISTERS

There are four 14-bit Special Function Registers used to specify duty cycle values for the PWM module:

- PDC0 (PDC0L and PDC0H)
- PDC1 (PDC1L and PDC1H)
- PDC2 (PDC2L and PDC2H)
- PDC3 (PDC3L and PDC3H)

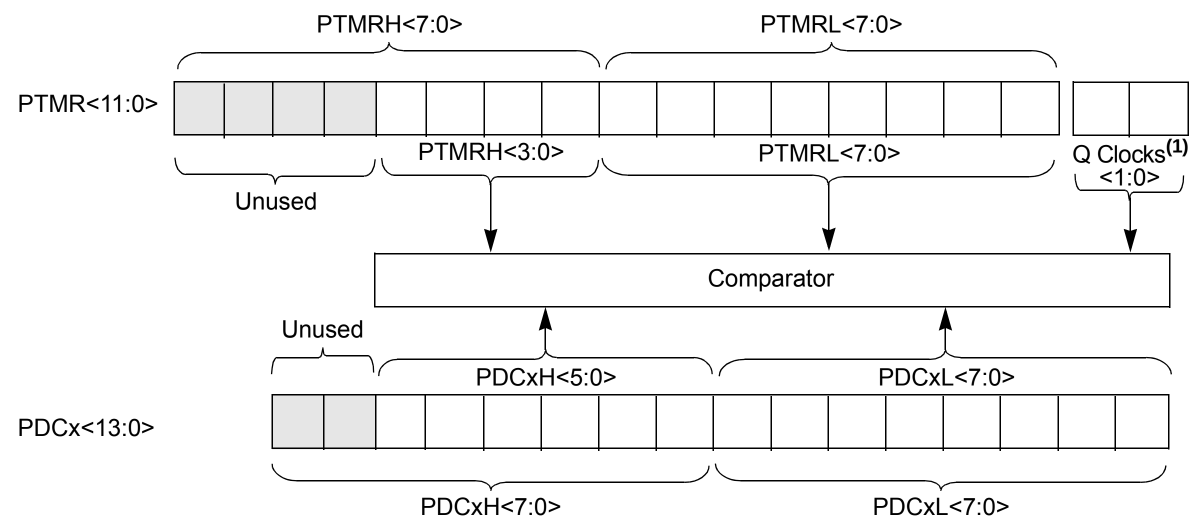
The value in each Duty Cycle register determines the amount of time that the PWM output is in the active state. The upper 12 bits of PDCx holds the actual duty cycle value from PTMRH/L<11:0>, while the lower 2 bits control which internal Q clock the duty cycle match will occur. This 2-bit value is decoded from the Q clocks as shown in Figure 18-11 (when the prescaler is 1:1 or PTCKPS<1:0> = `00`).

In Edge-Aligned mode, the PWM period starts at Q1 and ends when the Duty Cycle register matches the PTMR register as follows. The duty cycle match is considered when the upper 12 bits of the PDCx are equal to the PTMR and the lower 2 bits are equal to Q1, Q2, Q3 or Q4, depending on the lower two bits of the PDCx (when the prescaler is 1:1 or PTCKPS<1:0> = `00`).

Note: When the prescaler is not 1:1 (PTCKPS<1:0> ≠ ~`00`), the duty cycle match occurs at the Q1 clock of the instruction cycle when the PTMR and PDCx match occurs.

Each compare unit has logic that allows override of the PWM signals. This logic also ensures that the PWM signals will complement each other (with dead-time insertion) in Complementary mode (see **Section 18.7 "Dead-Time Generators"**).

FIGURE 18-11: DUTY CYCLE COMPARISON



Note 1: This value is decoded from the Q clocks:
`00` = duty cycle match occurs on Q1
`01` = duty cycle match occurs on Q2
`10` = duty cycle match occurs on Q3
`11` = duty cycle match occurs on Q4

18.6.2 DUTY CYCLE REGISTER BUFFERS

The four PWM Duty Cycle registers are double-buffered to allow glitchless updates of the PWM outputs. For each duty cycle block, there is a Duty Cycle Buffer register that is accessible by the user and a second Duty Cycle register that holds the actual compare value used in the present PWM period.

In Edge-Aligned PWM Output mode, a new duty cycle value will be updated whenever a PTMR match with the PTPER register occurs and PTMR is reset as shown in Figure 18-12. Also, the contents of the duty cycle buffers are automatically loaded into the Duty Cycle registers when the PWM time base is disabled (PTEN = 0).

When the PWM time base is in the Continuous Up/Down Count mode, new duty cycle values will be updated when the value of the PTMR register is zero and the PWM time base begins to count upwards. The contents of the duty cycle buffers are automatically loaded into the Duty Cycle registers when the PWM time base is disabled (PTEN = 0). Figure 18-13 shows the timings when the duty cycle update occurs for the Continuous Up/Down Count mode. In this mode, up to one entire PWM period is available for calculating and loading the new PWM duty cycle before changes take effect.

When the PWM time base is in the Continuous Up/Down Count mode with double updates, new duty cycle values will be updated when the value of the PTMR register is zero and when the value of the PTMR register matches the value in the PTPER register. The contents of the duty cycle buffers are automatically loaded into the Duty Cycle registers during both of the previously described conditions. Figure 18-14 shows the duty cycle updates for Continuous Up/Down Count mode with double updates. In this mode, only up to half of a PWM period is available for calculating and loading the new PWM duty cycle before changes take effect.

18.6.3 EDGE-ALIGNED PWM

Edge-aligned PWM signals are produced by the module when the PWM time base is in the Free-Running mode or the Single-Shot mode. For edge-aligned PWM outputs, the output for a given PWM channel has a period specified by the value loaded in PTPER and a duty cycle specified by the appropriate Duty Cycle register (see Figure 18-12). The PWM output is driven active at the beginning of the period (PTMR = 0) and is driven inactive when the value in the Duty Cycle register matches PTMR. A new cycle is started when PTMR matches the PTPER as explained in the PWM period section.

If the value in a particular Duty Cycle register is zero, then the output on the corresponding PWM pin will be inactive for the entire PWM period. In addition, the output on the PWM pin will be active for the entire PWM period if the value in the Duty Cycle register is greater than the value held in the PTPER register.

FIGURE 18-12: EDGE-ALIGNED PWM

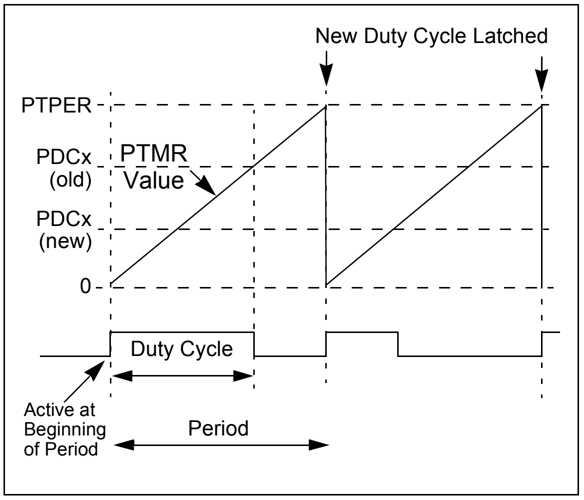


FIGURE 18-13: DUTY CYCLE UPDATE TIMES IN CONTINUOUS UP/DOWN COUNT MODE

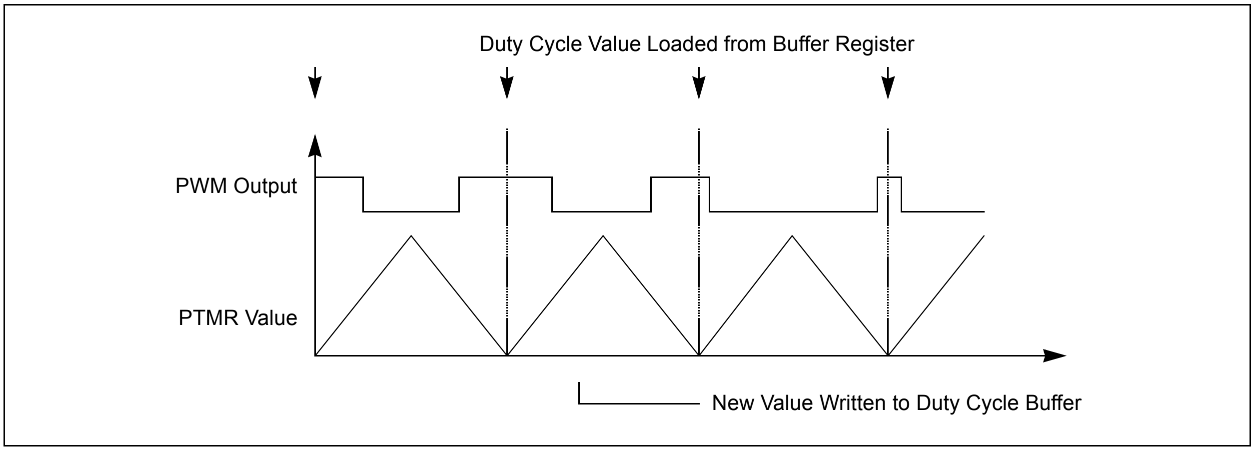
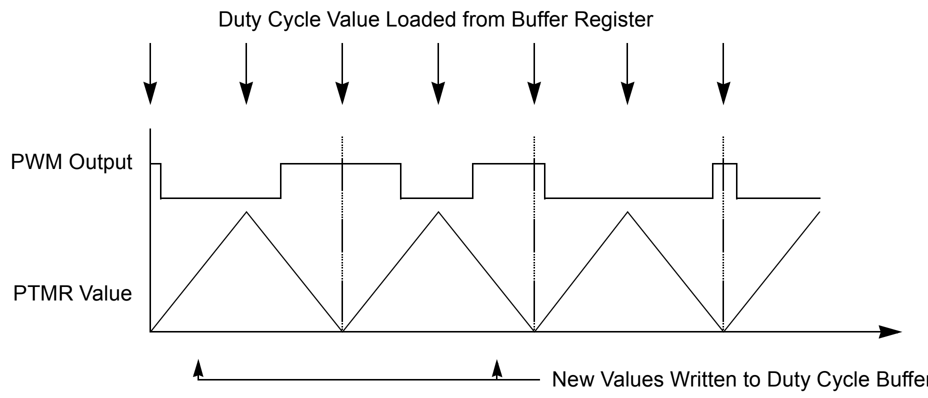


FIGURE 18-14: DUTY CYCLE UPDATE TIMES IN CONTINUOUS UP/DOWN COUNT MODE WITH DOUBLE UPDATES

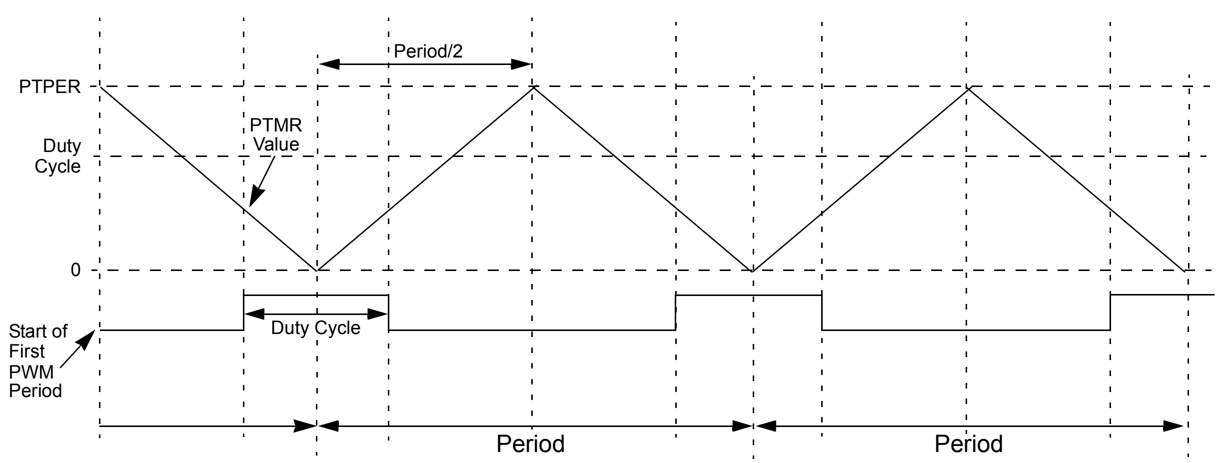


18.6.4 CENTER-ALIGNED PWM

Center-aligned PWM signals are produced by the module when the PWM time base is configured in a Continuous Up/Down Count mode (see Figure 18-15). The PWM compare output is driven to the active state when the value of the Duty Cycle register matches the value of PTMR and the PWM time base is counting downwards (PTDIR = `1`). The PWM compare output will be driven to the inactive state when the PWM time base is counting upwards (PTDIR = `0`) and the value in the PTMR register matches the duty cycle value. If the value in a particular Duty Cycle register is zero, then the output on the corresponding PWM pin will be inactive for the entire PWM period. In addition, the output on the PWM pin will be active for the entire PWM period if the value in the Duty Cycle register is equal to or greater than the value in the PTPER register.

> **Note:** When the PWM is started in Center-Aligned mode, the PWM Time Base Period register (PTPER) is loaded into the PWM Time Base register (PTMR) and the PTMR is configured automatically to start down counting. This is done to ensure that all the PWM signals don't start at the same time.

FIGURE 18-15: START OF CENTER-ALIGNED PWM



18.6.5 COMPLEMENTARY PWM OPERATION

The Complementary mode of PWM operation is useful to drive one or more power switches in half-bridge configuration as shown in Figure 18-16. This inverter topology is typical for a 3-phase induction motor, brushless DC motor or a 3-phase Uninterruptible Power Supply (UPS) control applications.

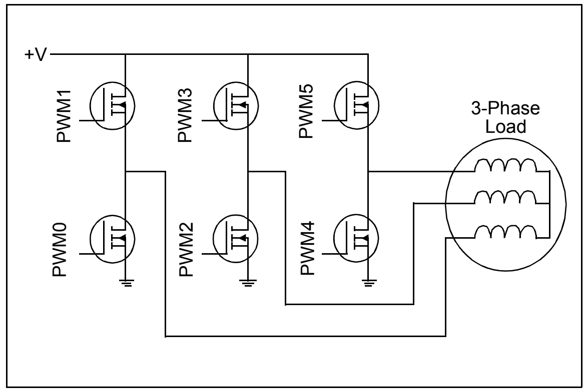
Each upper/lower power switch pair is fed by a complementary PWM signal. Dead time may be optionally inserted during device switching, where both outputs are inactive for a short period (see **Section 18.7 "Dead-Time Generators"**).

In Complementary mode, the duty cycle comparison units are assigned to the PWM outputs as follows:

- PDC0 register controls PWM1/PWM0 outputs
- PDC1 register controls PWM3/PWM2 outputs
- PDC2 register controls PWM5/PWM4 outputs
- PDC3 register controls PWM7/PWM6 outputs

PWM1/3/5/7 are the main PWMs that are controlled by the PDCx registers and PWM0/2/4/6 are the complemented outputs. When using the PWMs to control the half bridge, the odd numbered PWMs can be used to control the upper power switch and the even numbered PWMs used for the lower switches.

FIGURE 18-16: TYPICAL LOAD FOR COMPLEMENTARY PWM OUTPUTS



The Complementary mode is selected for each PWM I/O pin pair by clearing the appropriate PMODx bit in the PWMCON0 register. The PWM I/O pins are set to Complementary mode by default upon all kinds of device Resets.

18.7 Dead-Time Generators

In power inverter applications, where the PWMs are used in Complementary mode to control the upper and lower switches of a half-bridge, a dead-time insertion is highly recommended. The dead-time insertion keeps both outputs in inactive state for a brief time. This avoids any overlap in the switching during the state change of the power devices due to TON and TOFF characteristics.

Because the power output devices cannot switch instantaneously, some amount of time must be provided between the turn-off event of one PWM output in a complementary pair and the turn-on event of the other transistor. The PWM module allows dead time to be programmed. The following sections explain the dead-time block in detail.

18.7.1 DEAD-TIME INSERTION

Each complementary output pair for the PWM module has a 6-bit down counter used to produce the dead-time insertion. As shown in Figure 18-17, each dead-time unit has a rising and falling edge detector connected to the duty cycle comparison output. The dead time is loaded into the timer on the detected PWM edge event. Depending on whether the edge is rising or falling, one of the transitions on the complementary outputs is delayed until the timer counts down to zero. A timing diagram, indicating the dead-time insertion for one pair of PWM outputs, is shown in Figure 18-18.

FIGURE 18-17: DEAD-TIME CONTROL UNIT BLOCK DIAGRAM FOR ONE PWM OUTPUT PAIR

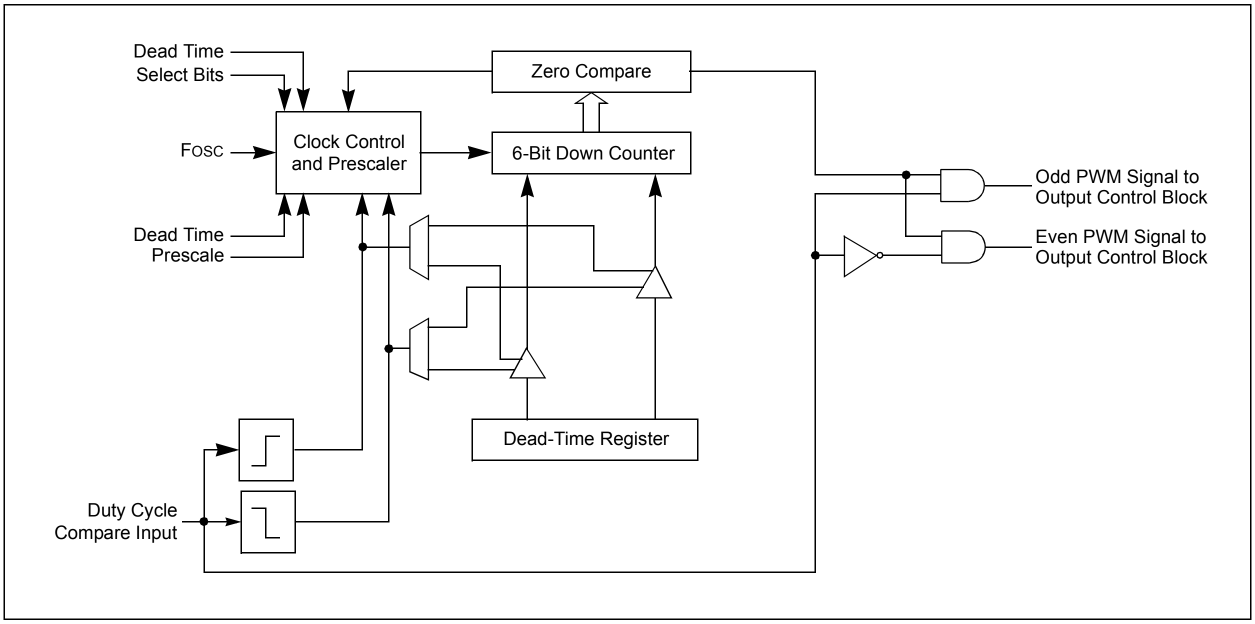
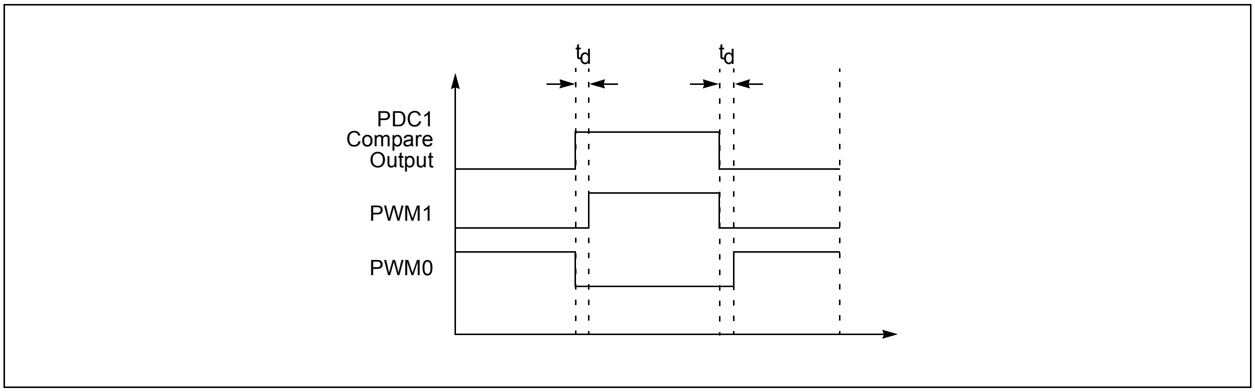


FIGURE 18-18: DEAD-TIME INSERTION FOR COMPLEMENTARY PWM



REGISTER 18-5: DTCON: DEAD-TIME CONTROL 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
DTPS1	DTPS0	DT5	DT4	DT3	DT2	DT1	DT0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-6 **DTPS<1:0>:** Dead-Time Unit A Prescale Select bits

11 = Clock source for dead-time unit is FOSC/16
10 = Clock source for dead-time unit is FOSC/8
01 = Clock source for dead-time unit is FOSC/4
00 = Clock source for dead-time unit is FOSC/2

bit 5-0 **DT<5:0>:** Unsigned 6-Bit Dead-Time Value for Dead-Time Unit bits

18.7.2 DEAD-TIME RANGES

The amount of dead time provided by the dead-time unit is selected by specifying the input clock prescaler value and a 6-bit unsigned value defined in the DTCON register. Four input clock prescaler selections have been provided to allow a suitable range of dead times based on the device operating frequency. FOSC/2, FOSC/4, FOSC/8 and FOSC/16 are the clock prescaler options available using the DTPS<1:0> control bits in the DTCON register.

After selecting an appropriate prescaler value, the dead time is adjusted by loading a 6-bit unsigned value into DTCON<5:0>. The dead-time unit prescaler is cleared on any of the following events:

- On a load of the down timer due to a duty cycle comparison edge event;
- On a write to the DTCON register; or
- On any device Reset.

18.7.3 DECREMENTING THE DEAD-TIME COUNTER

The dead-time counter is clocked from any of the Q clocks based on the following conditions.

1. The dead-time counter is clocked on Q1 when:
   - The DTPS bits are set to any of the following dead-time prescaler settings: FOSC/4, FOSC/8, FOSC/16
   - The PWM Time Base Prescale bits (PTCKPS) are set to any of the following prescale ratios: FOSC/16, FOSC/64, FOSC/256
2. The dead-time counter is clocked by a pair of Q clocks when the PWM Time Base Prescale bits are set to 1:1 (PTCKPS<1:0> = 00, FOSC/4) and the dead-time counter is clocked by the FOSC/2 (DTPS<1:0> = 00).
3. The dead-time counter is clocked using every other Q clock, depending on the two LSbs in the Duty Cycle registers:
   - If the PWM duty cycle match occurs on Q1 or Q3, then the dead-time counter is clocked using every Q1 and Q3.
   - If the PWM duty cycle match occurs on Q2 or Q4, then the dead-time counter is clocked using every Q2 and Q4.
4. When the DTPS<1:0> bits are set to any of the other dead-time prescaler settings (i.e., FOSC/4, FOSC/8 or FOSC/16) and the PWM time base prescaler is set to 1:1, the dead-time counter is clocked by the Q clock corresponding to the Q clocks on which the PWM duty cycle match occurs.

The actual dead time is calculated from the DTCON register as follows:

Dead Time = Dead-Time Value/(FOSC/Prescaler)

Table 18-3 shows example dead-time ranges as a function of the input clock prescaler selected and the device operating frequency.

TABLE 18-3: EXAMPLE DEAD-TIME RANGES

FOSC (MHz)	MIPS	Prescaler Selection	Dead-Time Min	Dead-Time Max
40	10	FOSC/2	50 ns	3.2 μs
40	10	FOSC/4	100 ns	6.4 μs
40	10	FOSC/8	200 ns	12.8 μs
40	10	FOSC/16	400 ns	25.6 μs
32	8	FOSC/2	62.5 ns	4 μs
32	8	FOSC/4	125 ns	8 μs
32	8	FOSC/8	250 ns	16 μs
32	8	FOSC/16	500 ns	32 μs
25	6.25	FOSC/2	80 ns	5.12 μs
25	6.25	FOSC/4	160 ns	10.2 μs
25	6.25	FOSC/8	320 ns	20.5 μs
25	6.25	FOSC/16	640 ns	41 μs
20	5	FOSC/2	100 ns	6.4 μs
20	5	FOSC/4	200 ns	12.8 μs
20	5	FOSC/8	400 ns	25.6 μs
20	5	FOSC/16	800 ns	51.2 μs
10	2.5	FOSC/2	200 ns	12.8 μs
10	2.5	FOSC/4	400 ns	25.6 μs
10	2.5	FOSC/8	800 ns	51.2 μs
10	2.5	FOSC/16	1.6 μs	102.4 μs
5	1.25	FOSC/2	400 ns	25.6 μs
5	1.25	FOSC/4	800 ns	51.2 μs
5	1.25	FOSC/8	1.6 μs	102.4 μs
5	1.25	FOSC/16	3.2 μs	204.8 μs
4	1	FOSC/2	0.5 μs	32 μs
4	1	FOSC/4	1 μs	64 μs
4	1	FOSC/8	2 μs	128 μs
4	1	FOSC/16	4 μs	256 μs

18.7.4 DEAD-TIME DISTORTION

Note 1: For small PWM duty cycles, the ratio of dead time to the active PWM time may become large. In this case, the inserted dead time will introduce distortion into waveforms produced by the PWM module. The user can ensure that dead-time distortion is minimized by keeping the PWM duty cycle at least three times larger than the dead time. A similar effect occurs for duty cycles at or near 100%. The maximum duty cycle used in the application should be chosen such that the minimum inactive time of the signal is at least three times larger than the dead time. If the dead time is greater or equal to the duty cycle of one of the PWM output pairs, then that PWM pair will be inactive for the whole period.

2: Changing the dead-time values in DTCON when the PWM is enabled may result in an undesired situation. Disable the PWM (PTEN = `0`) before changing the dead-time value

18.8 Independent PWM Output

Independent PWM mode is used for driving the loads (as shown in Figure 18-19) for driving one winding of a switched reluctance motor. A particular PWM output pair is configured in the Independent Output mode when the corresponding PMOD bit in the PWMCON0 register is set. No dead-time control is implemented between the PWM I/O pins when the module is operating in the Independent PWM mode and both I/O pins are allowed to be active simultaneously. This mode can also be used to drive stepper motors.

18.8.1 DUTY CYCLE ASSIGNMENT IN THE INDEPENDENT PWM MODE

In the Independent PWM mode, each duty cycle generator is connected to both PWM output pins in a given PWM output pair. The odd and even PWM output pins are driven with a single PWM duty cycle generator. PWM1 and PWM0 are driven by the PWM channel which uses the PDC0 register to set the duty cycle, PWM3 and PWM2 with PDC1, PWM5 and PWM4 with PDC2, and PWM7 and PWM6 with PDC3 (see Figure 18-3 and Register 18-4).

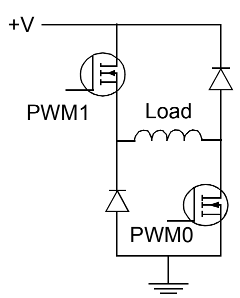
18.8.2 PWM CHANNEL OVERRIDE

PWM output may be manually overridden for each PWM channel by using the appropriate bits in the OVDCOND and OVDCONS registers. The user may select the following signal output options for each PWM output pin operating in the Independent PWM mode:

- I/O pin outputs PWM signal
- I/O pin inactive
- I/O pin active

Refer to **Section 18.10 "PWM Output Override"** for details for all the override functions.

FIGURE 18-19: CENTER CONNECTED LOAD



18.9 Single-Pulse PWM Operation

The single-pulse PWM operation is available only in Edge-Aligned mode. In this mode, the PWM module will produce single-pulse output. Single-pulse operation is configured when the PTMOD<1:0> bits are set to '01' in the PTCON0 register. This mode of operation is useful for driving certain types of ECMs.

In Single-Pulse mode, the PWM I/O pin(s) are driven to the active state when the PTEN bit is set. When the PWM timer match with the Duty Cycle register occurs, the PWM I/O pin is driven to the inactive state. When the PWM timer match with the PTPER register occurs, the PTMR register is cleared, all active PWM I/O pins are driven to the inactive state, the PTEN bit is cleared and an interrupt is generated if the corresponding interrupt bit is set.

Note:	PTPER and PDCx values are held as they are after the single-pulse output. To have another cycle of single pulse, only PTEN has to be enabled.

18.10 PWM Output Override

The PWM output override bits allow the user to manually drive the PWM I/O pins to specified logic states, independent of the duty cycle comparison units. The PWM override bits are useful when controlling various types of ECMs like a BLDC motor.

OVDCOND and OVDCONS registers are used to define the PWM override options. The OVDCOND register contains eight bits, POVD<7:0>, that determine which PWM I/O pins will be overridden. The OVDCONS register contains eight bits, POUT<7:0>, that determine the state of the PWM I/O pins when a particular output is overridden via the POVD bits.

The POVD bits are active-low control bits. When the POVD bits are set, the corresponding POUT bit will have no effect on the PWM output. In other words, the pins corresponding to POVD bits that are set will have the duty PWM cycle set by the PDCx registers. When one of the POVD bits is cleared, the output on the corresponding PWM I/O pin will be determined by the state of the POUT bit. When a POUT bit is set, the PWM pin will be driven to its active state. When the POUT bit is cleared, the PWM pin will be driven to its inactive state.

18.10.1 COMPLEMENTARY OUTPUT MODE

The even numbered PWM I/O pins have override restrictions when a pair of PWM I/O pins are operating in the Complementary mode (PMODx = `0`). In Complementary mode, if the even numbered pin is driven active by clearing the corresponding POVD bit and by setting POUT bits in the OVDCOND and OVDCONS registers, the output signal is forced to be the complement of the odd numbered I/O pin in the pair (see Figure 18-2 for details).

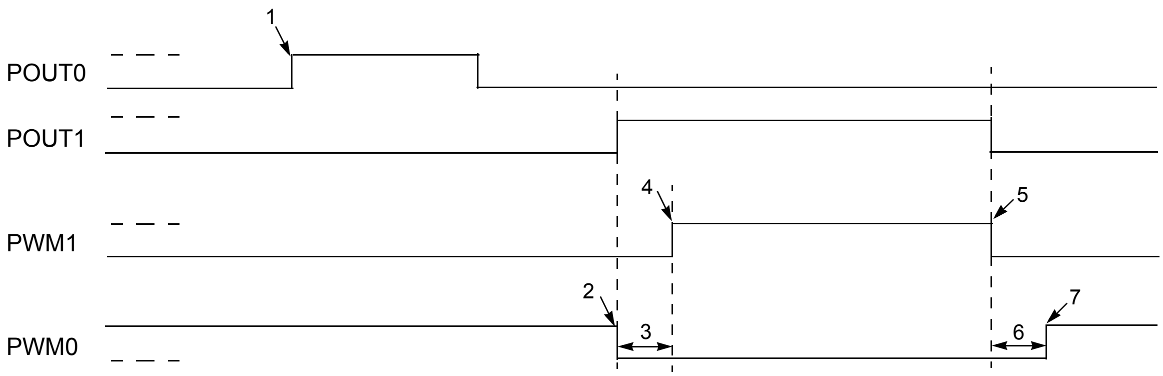
18.10.2 OVERRIDE SYNCHRONIZATION

If the OSYNC bit in the PWMCON1 register is set, all output overrides performed via the OVDCOND and OVDCONS registers will be synchronized to the PWM time base. Synchronous output overrides will occur on the following conditions:

- When the PWM is in Edge-Aligned mode, synchronization occurs when PTMR is zero.
- When the PWM is in Center-Aligned mode, synchronization occurs when PTMR is zero and when the value of PTMR matches PTPER.

Note 1:	In the Complementary mode, the even channel cannot be forced active by a Fault or override event when the odd channel is active. The even channel is always the complement of the odd channel with dead time inserted, before the odd channel can be driven to its active state, as shown in Figure 18-20.
2:	Dead time is inserted in the PWM channels even when they are in Override mode.

FIGURE 18-20: PWM OVERRIDE BITS IN COMPLEMENTARY MODE



Assume: POVD0 = 0; POVD1 = 0; PMOD0 = 0

1. Even override bits have no effect in Complementary mode.
2. Odd override bit is activated, which causes the even PWM to deactivate.
3. Dead-time insertion.
4. Odd PWM activated after the dead time.
5. Odd override bit is deactivated, which causes the odd PWM to deactivate.
6. Dead-time insertion.
7. Even PWM is activated after the dead time.

18.10.3 OUTPUT OVERRIDE EXAMPLES

Figure 18-21 shows an example of a waveform that might be generated using the PWM output override feature. The figure shows a six-step commutation sequence for a BLDC motor. The motor is driven through a 3-phase inverter as shown in Figure 18-16. When the appropriate rotor position is detected, the PWM outputs are switched to the next commutation state in the sequence. In this example, the PWM outputs are driven to specific logic states. The OVDCOND and OVDCONS register values used to generate the signals in Figure 18-21 are given in Table 18-4.

The PWM Duty Cycle registers may be used in conjunction with the OVDCOND and OVDCONS registers. The Duty Cycle registers control the average voltage across the load and the OVDCOND and OVDCONS registers control the commutation sequence. Figure 18-22 shows the waveforms, while Table 18-4 and Table 18-5 show the OVDCOND and OVDCONS register values used to generate the signals.

REGISTER 18-6: OVDCOND: OUTPUT OVERRIDE CONTROL REGISTER

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
POVD7[(1)]	POVD6[(1)]	POVD5	POVD4	POVD3	POVD2	POVD1	POVD0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-0 **POVD<7:0>:** PWM Output Override bits
1 = Output on PWM I/O pin is controlled by the value in the Duty Cycle register and the PWM time base
0 = Output on PWM I/O pin is controlled by the value in the corresponding POUT bit

Note 1: Unimplemented in PIC18F2331/2431 devices; maintain these bits clear.

REGISTER 18-7: OVDCONS: OUTPUT STATE REGISTER[(1,2)]

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
POUT7[(1)]	POUT6[(1)]	POUT5	POUT4	POUT3	POUT2	POUT1	POUT0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-0 **POUT<7:0>:** PWM Manual Output bits
1 = Output on PWM I/O pin is active when the corresponding PWM output override bit is cleared
0 = Output on PWM I/O pin is inactive when the corresponding PWM output override bit is cleared

Note 1: Unimplemented in PIC18F2331/2431 devices; maintain these bits clear.
2: With PWMs configured in Complementary mode, the output of even numbered PWM (PM0,2,4) will be complementary of the output of odd PWM (PWM1,3,5), irrespective of the POUT bit setting.

FIGURE 18-21: PWM OUTPUT OVERRIDE EXAMPLE #1

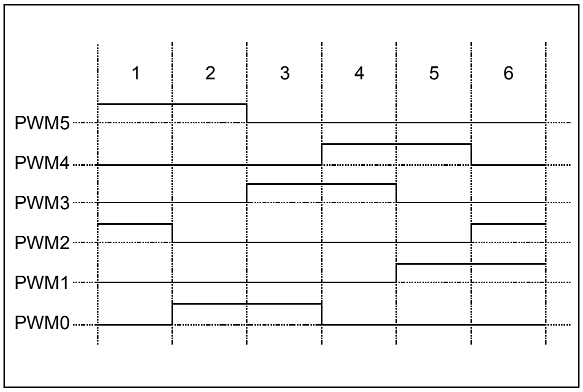


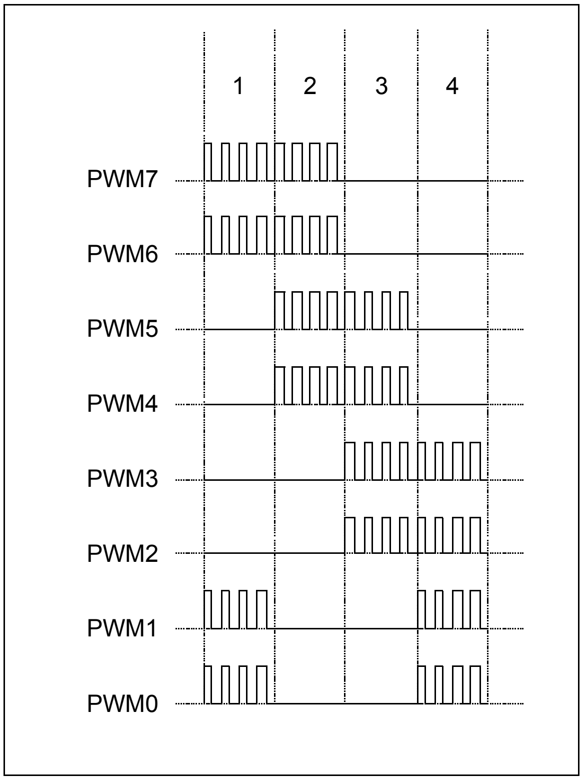
TABLE 18-4: PWM OUTPUT OVERRIDE EXAMPLE #1

State	OVDCOND (POVD)	OVDCONS (POUT)
1	00000000b	00100100b
2	00000000b	00100001b
3	00000000b	00001001b
4	00000000b	00011000b
5	00000000b	00010010b
6	00000000b	00000110b

TABLE 18-5: PWM OUTPUT OVERRIDE EXAMPLE #2

State	OVDCOND (POVD)	OVDCONS (POUT)
1	11000011b	00000000b
2	11110000b	00000000b
3	00111100b	00000000b
4	00001111b	00000000b

FIGURE 18-22: PWM OUTPUT OVERRIDE EXAMPLE #2



18.11 PWM Output and Polarity Control

There are three device Configuration bits associated with the PWM module that provide PWM output pin control defined in the CONFIG3L Configuration register. They are:

- HPOL
- LPOL
- PWMPIN

These three Configuration bits work in conjunction with the three PWM Enable bits (PWMEN<2:0>) in the PWMCON0 register. The Configuration bits and PWM enable bits ensure that the PWM pins are in the correct states after a device Reset occurs.

18.11.1 OUTPUT PIN CONTROL

The PWMEN<2:0> control bits enable each PWM output pin as required in the application.

All PWM I/O pins are general purpose I/O. When a pair of pins are enabled for PWM output, the PORT and TRIS registers controlling the pins are disabled. Refer to Figure 18-23 for details.

18.11.2 OUTPUT POLARITY CONTROL

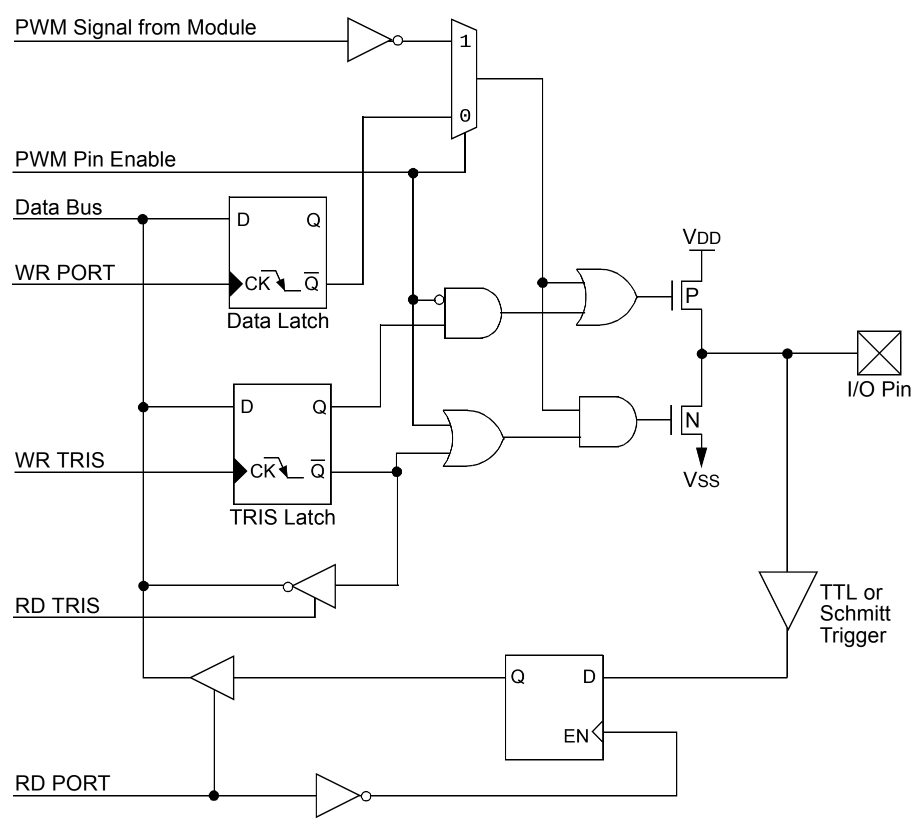
The polarity of the PWM I/O pins is set during device programming via the HPOL and LPOL Configuration bits in the CONFIG3L Configuration register. The HPOL Configuration bit sets the output polarity for the high side PWM outputs: PWM1, PWM3, PWM5 and PWM7. The polarity is active-low when HPOL is cleared (= 0), and active-high when it is set (= 1).

The LPOL Configuration bit sets the output polarity for the low side PWM outputs: PWM0, PWM2, PWM4 and PWM6. As with HPOL, they are active-low when LPOL is cleared and active-high when it is set.

All output signals generated by the PWM module are referenced to the polarity control bits, including those generated by Fault inputs or manual override (see **Section 18.10 "PWM Output Override"**).

The default polarity Configuration bits have the PWM I/O pins in active-high output polarity.

FIGURE 18-23: PWM I/O PIN BLOCK DIAGRAM



Note: I/O pin has protection diodes to VDD and VSS. PWM polarity selection logic not shown for clarity.

18.11.3 PWM OUTPUT PIN RESET STATES

The PWMPIN Configuration bit determines the PWM output pins to be PWM output pins or digital I/O pins, after the device comes out of Reset. If the PWMPIN Configuration bit is unprogrammed (default), the PWMEN<2:0> control bits will be cleared on a device Reset. Consequently, all PWM outputs will be tri-stated and controlled by the corresponding PORT and TRIS registers. If the PWMPIN Configuration bit is programmed low, the PWMEN<2:0> control bits will be set, as follows, on a device Reset:

- PWMEN<2:0> = `101` if device has 8 PWM pins (PIC18F4331/4431 devices)
- PWMEN<2:0> = `100` if device has 6 PWM pins (PIC18F2331/2431 devices)

All PWM pins will be enabled for PWM output and will have the output polarity defined by the HPOL and LPOL Configuration bits.

18.12 PWM Fault Inputs

There are two Fault inputs associated with the PWM module. The main purpose of the input Fault pins is to disable the PWM output signals and drive them into an inactive state. The action of the Fault inputs is performed directly in hardware so that when a Fault occurs, it can be managed quickly and the PWM outputs are put into an inactive state to save the power devices connected to the PWMs.

The PWM Fault inputs are FLTA and FLTB, which can come from I/O pins, the CPU or another module. The FLTA and FLTB pins are active-low inputs so it is easy to "OR" many sources to the same input. FLTB and its associated logic are not implemented on PIC18F2331/2431 devices.

The FLTCONFIG register (Register 18-8) defines the settings of FLTA and FLTB inputs.

Note:	The inactive state of the PWM pins are dependent on the HPOL and LPOL Configuration bit settings, which define the active and inactive state for PWM outputs.

18.12.1 FAULT PIN ENABLE BITS

By setting the bits, FLTAEN and FLTBEN in the FLTCONFIG register, the corresponding Fault inputs are enabled. If both bits are cleared, then the Fault inputs have no effect on the PWM module.

18.12.2 MFAULT INPUT MODES

The FLTAMOD and FLTBMOD bits in the FLTCONFIG register determine the modes of PWM I/O pins that are deactivated when they are overridden by Fault input.

The FLTAS and FLTBS bits in the FLTCONFIG register give the status of Fault A and Fault B inputs.

Each of the Fault inputs have two modes of operation:

- **Inactive Mode (FLTxMOD = `0`)**

  This is a Catastrophic Fault Management mode. When the Fault occurs in this mode, the PWM outputs are deactivated. The PWM pins will remain in Inactivate mode until the Fault is cleared (Fault input is driven high) and the corresponding Fault Status bit has been cleared in software. The PWM outputs are enabled immediately at the beginning of the following PWM period, after the Fault Status bit (FLTxS) is cleared.

- **Cycle-by-Cycle Mode (FLTxMOD = `1`)**

  When the Fault occurs in this mode, the PWM outputs are deactivated. The PWM outputs will remain in the defined Fault states (all PWM outputs inactive) for as long as the Fault pin is held low. After the Fault pin is driven high, the PWM outputs will return to normal operation at the beginning of the following PWM period and the FLTxS bit is automatically cleared.

18.12.3 PWM OUTPUTS WHILE IN FAULT CONDITION

While in the Fault state (i.e., one or both $\overline{\text{FLTA}}$ and $\overline{\text{FLTB}}$ inputs are active), the PWM output signals are driven into their inactive states. The selection of which PWM outputs are deactivated (while in the Fault state) is determined by the FLTCON bit in the FLTCONFIG register as follows:

- FLTCON = `1`: When $\overline{\text{FLTA}}$ or $\overline{\text{FLTB}}$ is asserted, the PWM outputs (i.e., PWM<7:0>) are driven into their inactive state.
- FLTCON = `0`: When $\overline{\text{FLTA}}$ or $\overline{\text{FLTB}}$ is asserted, only PWM<5:0> outputs are driven inactive, leaving PWM<7:6> activated.

> **Note:** Disabling only three PWM channels and leaving one PWM channel enabled when in the Fault state, allows the flexibility to have at least one PWM channel enabled. None of the PWM outputs can be enabled (driven with the PWM Duty Cycle registers) while FLTCON = `1` and the Fault condition is present.

18.12.4 PWM OUTPUTS IN DEBUG MODE

The BRFEN bit in the FLTCONFIG register controls the simulation of a Fault condition, when a breakpoint is hit, while debugging the application using an In-Circuit Emulator (ICE) or an In-Circuit Debugger (ICD). Setting the BRFEN to high, enables the Fault condition on breakpoint, thus driving the PWM outputs to the inactive state. This is done to avoid any continuous keeping of status on the PWM pin, which may result in damage of the power devices connected to the PWM outputs.

If BRFEN = `0`, the Fault condition on breakpoint is disabled.

> **Note:** It is highly recommended to enable the Fault condition on breakpoint if a debugging tool is used while developing the firmware and high-power circuitry. When the device is ready to program after debugging the firmware, the BRFEN bit can be disabled.

REGISTER 18-8: FLTCONFIG: FAULT CONFIGURATION 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
BRFEN	FLTBS(1)	FLTBMOD(1)	FLTBEN(1)	FLTCON(2)	FLTAS	FLTAMOD	FLTAEN
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **BRFEN:** Breakpoint Fault Enable bit
1 = Enable Fault condition on a breakpoint (i.e., only when PWMPIN = 1)
0 = Disable Fault condition

bit 6 **FLTBS:** Fault B Status bit(1)
1 = $\overline{\text{FLTB}}$ is asserted:
if FLTBMOD = 0, cleared by the user;
if FLTBMOD = 1, cleared automatically at beginning of the new period when $\overline{\text{FLTB}}$ is deasserted
0 = No Fault

bit 5 **FLTBMOD:** Fault B Mode bit(1)
1 = Cycle-by-Cycle mode: Pins are inactive for the remainder of the current PWM period or until $\overline{\text{FLTB}}$ is deasserted; FLTBS is cleared automatically when $\overline{\text{FLTB}}$ is inactive (no Fault present)
0 = Inactive mode: Pins are deactivated (catastrophic failure) until $\overline{\text{FLTB}}$ is deasserted and FLTBS is cleared by the user only

bit 4 **FLTBEN:** Fault B Enable bit(1)
1 = Enable Fault B
0 = Disable Fault B

bit 3 **FLTCON:** Fault Configuration bit(2)
1 = $\overline{\text{FLTA}}$, $\overline{\text{FLTB}}$ or both deactivates all PWM outputs
0 = $\overline{\text{FLTA}}$ or $\overline{\text{FLTB}}$ deactivates PWM<5:0>

bit 2 **FLTAS:** Fault A Status bit
1 = $\overline{\text{FLTA}}$ is asserted:
if FLTAMOD = 0, cleared by the user;
if FLTAMOD = 1, cleared automatically at beginning of the new period when $\overline{\text{FLTA}}$ is deasserted
0 = No Fault

bit 1 **FLTAMOD:** Fault A Mode bit
1 = Cycle-by-Cycle mode: Pins are inactive for the remainder of the current PWM period or until $\overline{\text{FLTA}}$ is deasserted; FLTAS is cleared automatically
0 = Inactive mode: Pins are deactivated (catastrophic failure) until $\overline{\text{FLTA}}$ is deasserted and FLTAS is cleared by the user only

bit 0 **FLTAEN:** Fault A Enable bit
1 = Enable Fault A
0 = Disable Fault A

Note 1: Unimplemented in PIC18F2331/2431 devices; maintain these bits clear.
2: PWM<7:6> are implemented only on PIC18F4331/4431 devices. On PIC18F2331/2431 devices, setting or clearing FLTCON has no effect.

18.13 PWM Update Lockout

For a complex PWM application, the user may need to write up to four Duty Cycle registers and the PWM Time Base Period register, PTPER, at a given time. In some applications, it is important that all buffer registers be written before the new duty cycle and period values are loaded for use by the module.

A PWM update lockout feature may optionally be enabled so the user may specify when new duty cycle buffer values are valid. The PWM update lockout feature is enabled by setting the control bit, UDIS, in the PWMCON1 register. This bit affects all Duty Cycle Buffer registers and the PWM Time Base Period register, PTPER.

To perform a PWM update lockout:

1. Set the UDIS bit.
2. Write all Duty Cycle registers and PTPER, if applicable.
3. Clear the UDIS bit to re-enable updates.
4. With this, when UDIS bit is cleared, the buffer values will be loaded to the actual registers. This makes a synchronous loading of the registers.

18.14 PWM Special Event Trigger

The PWM module has a Special Event Trigger capability that allows A/D conversions to be synchronized to the PWM time base. The A/D sampling and conversion time may be programmed to occur at any point within the PWM period. The Special Event Trigger allows the user to minimize the delay between the time when A/D conversion results are acquired and the time when the duty cycle value is updated.

The PWM 16-bit Special Event Trigger register, SEVTCMP (high and low), and five control bits in the PWMCON1 register are used to control its operation.

The PTMR value for which a Special Event Trigger should occur is loaded into the SEVTCMP register pair. The SEVTDIR bit in the PWMCON1 register specifies the counting phase when the PWM time base is in a Continuous Up/Down Count mode.

If the SEVTDIR bit is cleared, the Special Event Trigger will occur on the upward counting cycle of the PWM time base. If SEVTDIR is set, the Special Event Trigger will occur on the downward count cycle of the PWM time base. The SEVTDIR bit has effect only when the PWM timer is in the Continuous Up/Down Count mode.

18.14.1 SPECIAL EVENT TRIGGER ENABLE

The PWM module will always produce Special Event Trigger pulses. This signal may optionally be used by the A/D module. Refer to **Section 21.0 "10-Bit High-Speed Analog-to-Digital Converter (A/D) Module"** for details.

18.14.2 SPECIAL EVENT TRIGGER POSTSCALER

The PWM Special Event Trigger has a postscaler that allows a 1:1 to 1:16 postscale ratio. The postscaler is configured by writing the SEVOPS<3:0> control bits in the PWMCON1 register.

The Special Event Trigger output postscaler is cleared on any write to the SEVTCMP register pair, or on any device Reset.

TABLE 18-6: REGISTERS ASSOCIATED WITH THE POWER CONTROL PWM MODULE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
IPR3	—	—	—	PTIP	IC3DRIP	IC2QEIP	IC1IP	TMR5IP	56
PIE3	—	—	—	PTIE	IC3DRIE	IC2QEIE	IC1IE	TMR5IE	56
PIR3	—	—	—	PTIF	IC3DRIF	IC2QEIF	IC1IF	TMR5IF	56
PTCON0	PTOPS3	PTOPS2	PTOPS1	PTOPS0	PTCKPS1	PTCKPS0	PTMOD1	PTMOD0	58
PTCON1	PTEN	PTDIR	—	—	—	—	—	—	58
PTMRL[1]	PWM Time Base Register (lower 8 bits)								58
PTMRH[1]	**UNUSED**				PWM Time Base Register (upper 4 bits)				58
PTPERL[1]	PWM Time Base Period Register (lower 8 bits)								58
PTPERH[1]	**UNUSED**				PWM Time Base Period Register (upper 4 bits)				58
SEVTCMPL[1]	PWM Special Event Compare Register (lower 8 bits)								58
SEVTCMPH[1]	**UNUSED**				PWM Special Event Compare Register (upper 4 bits)				58
PWMCON0	—	PWMEN2	PWMEN1	PWMEN0	PMOD3[2]	PMOD2	PMOD1	PMOD0	58
PWMCON1	SEVOPS3	SEVOPS2	SEVOPS1	SEVOPS0	SEVTDIR	—	UDIS	OSYNC	58
DTCON	DTPS1	DTPS0	DT5	DT4	DT3	DT2	DT1	DT0	58
FLTCONFIG	BRFEN	FLTBS[2]	FLTBMOD[2]	FLTBEN[2]	FLTCON	FLTAS	FLTAMOD	FLTAEN	58
OVDCOND	POVD7[2]	POVD6[2]	POVD5	POVD4	POVD3	POVD2	POVD1	POVD0	58
OVDCONS	POUT7[2]	POUT6[2]	POUT5	POUT4	POUT3	POUT2	POUT1	POUT0	58
PDC0L[1]	PWM Duty Cycle #0L Register (lower 8 bits)								58
PDC0H[1]	**UNUSED**		PWM Duty Cycle #0H Register (upper 6 bits)						58
PDC1L[1]	PWM Duty Cycle #1L register (lower 8 bits)								58
PDC1H[1]	**UNUSED**		PWM Duty Cycle #1H Register (upper 6 bits)						58
PDC2L[1]	PWM Duty Cycle #2L Register (lower 8 bits)								58
PDC2H[1]	**UNUSED**		PWM Duty Cycle #2H Register (upper 6 bits)						58
PDC3L[1,2]	PWM Duty Cycle #3L Register (lower 8 bits)								58
PDC3H[1,2]	**UNUSED**		PWM Duty Cycle #3H Register (upper 6 bits)						58

Legend: — = Unimplemented, read as '0'. Shaded cells are not used with the power control PWM.

Note 1: Double-buffered register pairs. Refer to text for explanation of how these registers are read and written to.

2: Unimplemented in PIC18F2331/2431 devices; maintain these bits clear. Reset values shown are for PIC18F4331/4431 devices.

NOTES:

19.0 SYNCHRONOUS SERIAL PORT (SSP) MODULE

19.1 SSP Module Overview

The Synchronous Serial Port (SSP) 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 SSP module can operate in one of two modes:

- Serial Peripheral Interface (SPI)
- Inter-Integrated Circuit (I^2C™)

An overview of I^2C operations and additional information on the SSP module can be found in the *"PIC® Mid-Range MCU Family Reference Manual"* (DS33023).

Refer to application note *AN578, "Use of the SSP Module in the I^2C™ Multi-Master Environment"* (DS00578).

19.2 SPI Mode

This section contains register definitions and operational characteristics of the SPI module. Additional information on the SPI module can be found in the *"PIC® Mid-Range MCU Family Reference Manual"* (DS33023).

SPI mode allows 8 bits of data to be synchronously transmitted and received simultaneously. To accomplish communication, typically three pins are used:

- Serial Data Out (SDO)
- Serial Data In (SDI)
- Serial Clock (SCK)

Additionally, a fourth pin may be used when in a Slave mode of operation:

- Slave Select ($\overline{SS}$)

When initializing the SPI, several options need to be specified. This is done by programming the appropriate control bits in the SSPCON (SSPCON<5:0>) and SSPSTAT<7:6> registers. 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)
- Clock edge (output data on rising/falling edge of SCK)
- Clock rate (Master mode only)
- Slave Select mode (Slave mode only)

REGISTER 19-1: SSPSTAT: SYNCHRONOUS SERIAL PORT STATUS REGISTER

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **SMP:** Sample bit
SPI Master mode:
1 = Input data sampled at end of data output time
0 = Input data sampled at middle of data output time
SPI Slave mode:
SMP must be cleared when SPI is used in Slave mode.

bit 6 **CKE**: SPI Clock Edge Select bit (Figure 19-2, Figure 19-3 and Figure 19-4)
SPI mode, CKP = 0:
1 = Data transmitted on rising edge of SCK
0 = Data transmitted on falling edge of SCK
SPI mode, CKP = 1:
1 = Data transmitted on falling edge of SCK
0 = Data transmitted on rising edge of SCK
I^2C™ mode:
This bit must be maintained clear.

bit 5 **D/$\overline{A}$**: Data/$\overline{\text{Address}}$ bit (I^2C mode only)
1 = Indicates that the last byte received or transmitted was data
0 = Indicates that the last byte received or transmitted was address

bit 4 **P**: Stop bit (I^2C mode only)
This bit is cleared when the SSP module is disabled or when the Start bit is detected last; SSPEN is cleared.
1 = Indicates that a Stop bit has been detected last (this bit is '0' on Reset)
0 = Stop bit was not detected last

bit 3 **S**: Start bit (I^2C mode only)
This bit is cleared when the SSP module is disabled or when the Stop bit is detected last; SSPEN is cleared.
1 = Indicates that a Start bit has been detected last (this bit is '0' on Reset)
0 = Start bit was not detected last

bit 2 **R/$\overline{W}$**: Read/$\overline{\text{Write}}$ Information bit (I^2C mode only)
This bit holds the R/$\overline{W}$ bit information following the last address match. This bit is only valid from the address match to the next Start bit, Stop bit or $\overline{\text{ACK}}$ bit.
1 = Read
0 = Write

bit 1 **UA**: Update Address bit (10-Bit I^2C mode only)
1 = Indicates that the user needs to update the address in the SSPADD register
0 = Address does not need to be updated

bit 0 **BF**: Buffer Full Status bit
Receive (SPI and I^2C modes):
1 = Receive complete, SSPBUF is full
0 = Receive not complete, SSPBUF is empty
Transmit (I^2C mode only):
1 = Transmit in progress, SSPBUF is full
0 = Transmit complete, SSPBUF is empty

REGISTER 19-2: SSPCON: SYNCHRONOUS SERIAL PORT CONTROL 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
WCOL	SSPOV(1)	SSPEN(2)	CKP	SSPM3(3)	SSPM2(3)	SSPM1(3)	SSPM0(3)
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **WCOL:** Write Collision Detect bit
1 = The SSPBUF register is written while it is still transmitting the previous word (must be cleared in software)
0 = No collision

bit 6 **SSPOV**: Receive Overflow Indicator bit(1)
In SPI mode:
1 = A new byte is received while the SSPBUF register is still holding the previous data. In case of overflow, the data in SSPSR is lost. Overflow can only occur in Slave mode. The user must read the SSPBUF, 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 SSPBUF register.
0 = No overflow
In I^2C™ mode:
1 = A byte is received while the SSPBUF register is still holding the previous byte. SSPOV is a “don’t care” in Transmit mode. SSPOV must be cleared in software in either mode.
0 = No overflow

bit 5 **SSPEN**: Synchronous Serial Port Enable bit(2)
In SPI mode:
1 = Enables serial port and configures SCK, SDO and SDI as serial port pins
0 = Disables serial port and configures these pins as I/O port pins
In I^2C mode:
1 = Enables the serial port and configures the SDA and SCL pins as serial port pins
0 = Disables serial port and configures these pins as I/O port pins
In both modes, when enabled, these pins must be properly configured as input or output.

bit 4 **CKP**: Clock Polarity Select bit
In SPI mode:
1 = Idle state for clock is a high level
0 = Idle state for clock is a low level
In I^2C mode:
SCK release control.
1 = Enables clock
0 = Holds clock low (clock stretch). (Used to ensure data setup time.)

Note 1: In Master mode, the overflow bit is not set since each new reception (and transmission) is initiated by writing to the SSPBUF register.
2: When enabled, these pins must be properly configured as inputs or outputs.
3: Bit combinations not specifically listed here are either reserved or implemented in I^2C™ mode only.

REGISTER 19-2: SSPCON: SYNCHRONOUS SERIAL PORT CONTROL REGISTER (CONTINUED)

bit 3-0 **SSPM<3:0>**: Synchronous Serial Port Mode Select bits(3)

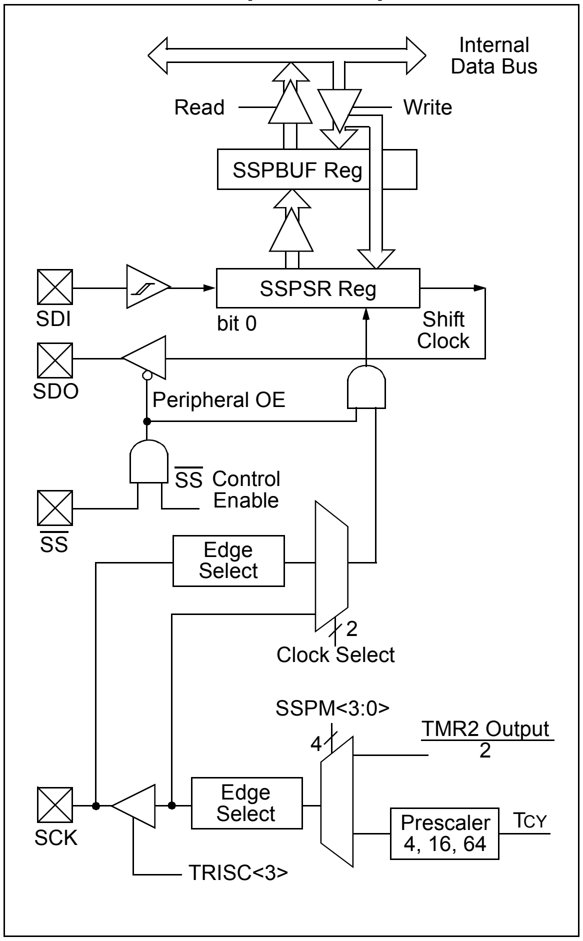
`0000` = SPI Master mode, Clock = FOSC/4
`0001` = SPI Master mode, Clock = FOSC/16
`0010` = SPI Master mode, Clock = FOSC/64
`0011` = SPI Master mode, Clock = TMR2 output/2
`0100` = SPI Slave mode, Clock = SCK pin, $\overline{SS}$ pin control enabled
`0101` = SPI Slave mode, Clock = SCK pin, $\overline{SS}$ pin control disabled, $\overline{SS}$ can be used as I/O pin
`0110` = I^{2}C Slave mode, 7-bit address
`0111` = I^{2}C Slave mode, 10-bit address
`1011` = I^{2}C Firmware Controlled Master mode (slave Idle)
`1110` = I^{2}C Slave mode, 7-bit address with Start and Stop bit interrupts enabled
`1111` = I^{2}C Slave mode, 10-bit address with Start and Stop bit interrupts enabled

Note 1: In Master mode, the overflow bit is not set since each new reception (and transmission) is initiated by writing to the SSPBUF register.

2: When enabled, these pins must be properly configured as inputs or outputs.

3: Bit combinations not specifically listed here are either reserved or implemented in I^{2}C™ mode only.

FIGURE 19-1: SSP BLOCK DIAGRAM (SPI MODE)



To enable the serial port, SSP Enable bit, SSPEN (SSPCON<5>), must be set. To reset or reconfigure SPI mode, clear bit SSPEN, reinitialize the SSPCON register and then set bit SSPEN. This configures the SDI, SDO, SCK and $\overline{SS}$ pins as serial port pins. For the pins to behave as the serial port function, they must have their data direction bits (in the TRISC register) appropriately programmed. That is:

- Serial Data Out (SDO) – RC7/RX/DT/SDO or RD1/SDO
- SDI must have TRISC<4> or TRISD<2> set
- SDO must have TRISC<7> or TRISD<1> cleared
- SCK (Master mode) must have TRISC<5> or TRISD<3> cleared
- SCK (Slave mode) must have TRISC<5> or TRISD<3> set
- $\overline{SS}$ must have TRISA<6> set

Note 1: When the SPI is in Slave mode, with the $\overline{SS}$ pin control enabled, (SSPCON<3:0> = `0100`), the SPI module will reset if the $\overline{SS}$ pin is set to VDD.

2: If the SPI is used in Slave mode with CKE = `1`, then the $\overline{SS}$ pin control must be enabled.

3: When the SPI is in Slave mode with $\overline{SS}$ pin control enabled (SSPCON<3:0> = `0100`), the state of the $\overline{SS}$ pin can affect the state read back from the TRISC<6> bit. The peripheral OE signal from the SSP module into PORTC controls the state that is read back from the TRISC<6> bit (see **Section 11.3 "PORTC, TRISC and LATC Registers"** for information on PORTC). If Read-Modify-Write instructions, such as BSF, are performed on the TRISC register while the $\overline{SS}$ pin is high, this will cause the TRISC<6> bit to be set, thus disabling the SDO output.

FIGURE 19-2: SPI MODE TIMING, MASTER MODE

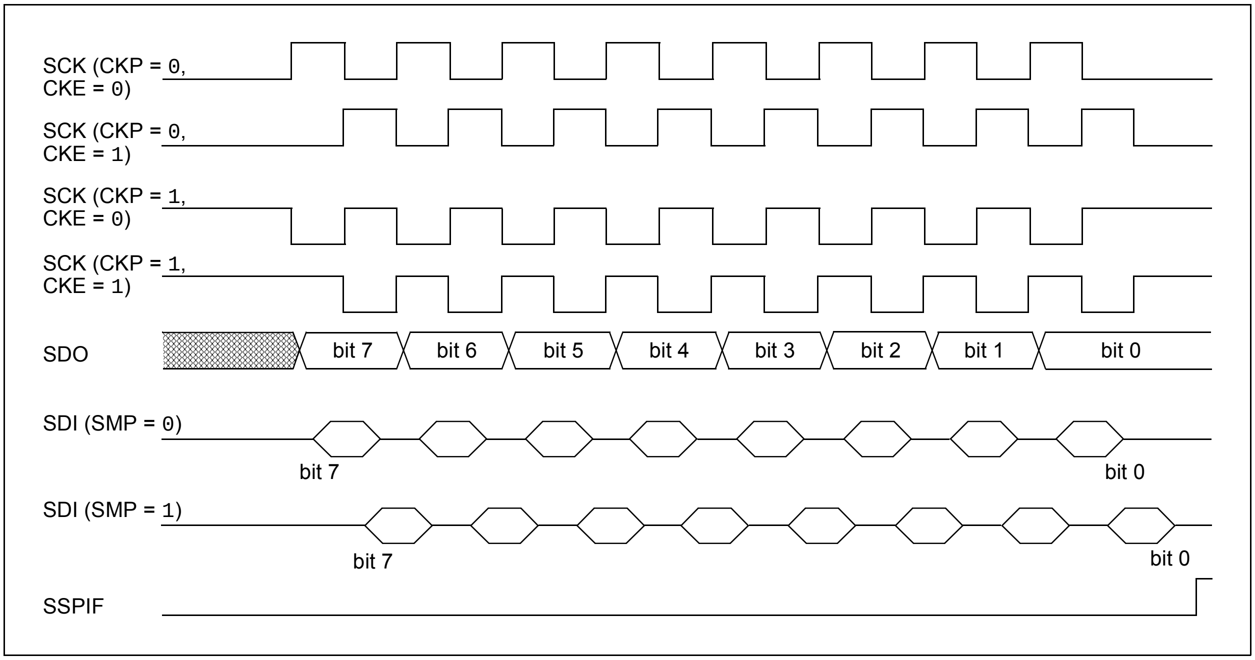


FIGURE 19-3: SPI MODE TIMING (SLAVE MODE WITH CKE = 0)

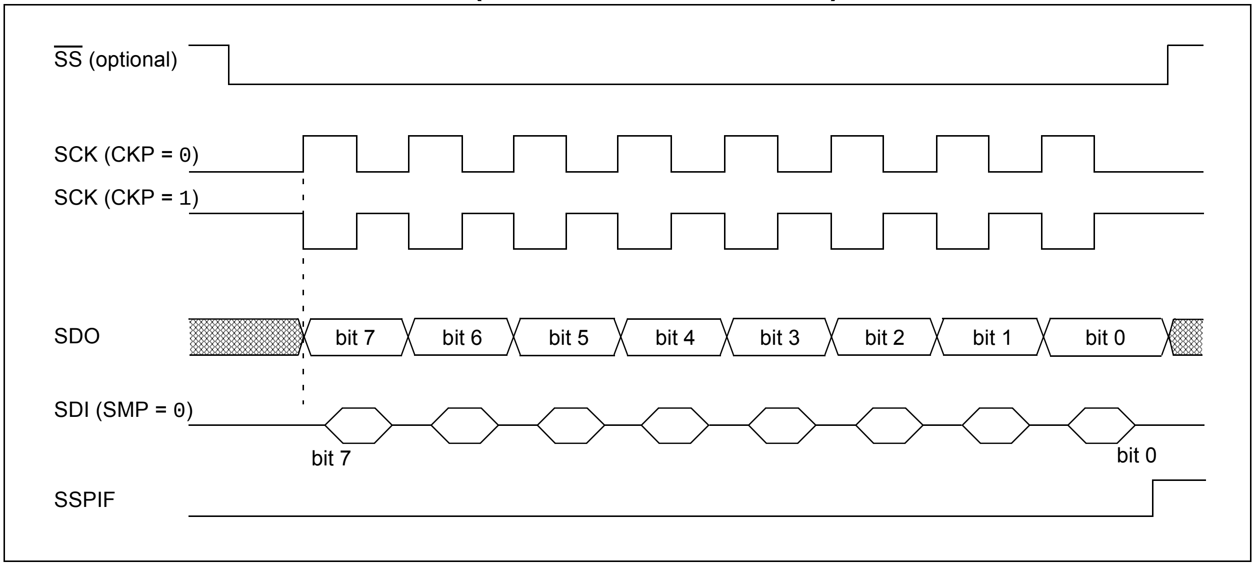


FIGURE 19-4: SPI MODE TIMING (SLAVE MODE WITH CKE = 1)

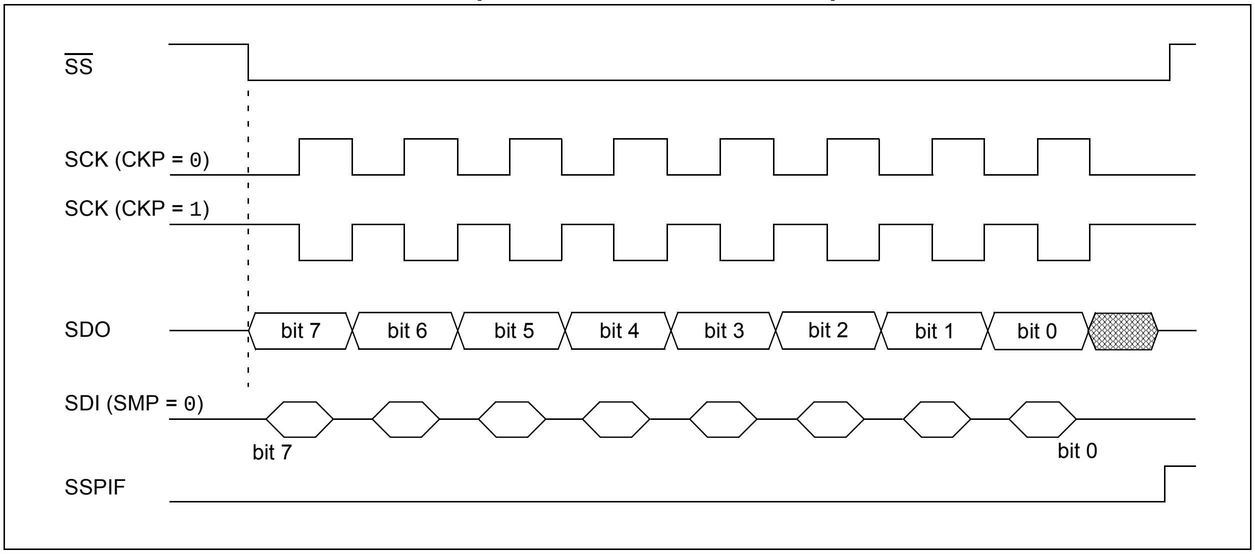


TABLE 19-1: REGISTERS ASSOCIATED WITH SPI OPERATION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
PIR1	—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	57
PIE1	—	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	57
TRISC	PORTC Data Direction Register								57
SSPBUF	SSP Receive Buffer/Transmit Register								55
SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	55
TRISA	TRISA7[(1)]	TRISA6[(2)]	PORTA Data Direction Register						57
SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	55

Legend: — = unimplemented, read as '0'. Shaded cells are not used by the SSP in SPI mode.

Note 1: RA7 and associated bits are configured as port pins in INTIO2 Oscillator mode only and read '0' in all other oscillator modes.

2: RA6 and associated bits are configured as port pins in RCIO, ECIO and INTIO2 (with port function on RA6) Oscillator modes only and read '0' in all other oscillator modes.

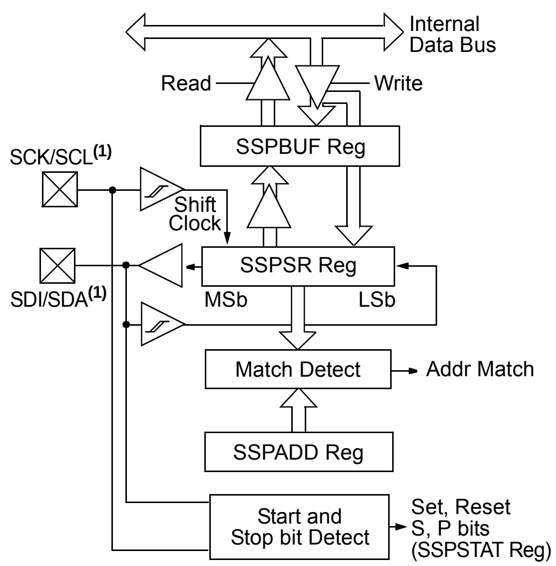
19.3 SSP I²C Operation

The SSP module, in I²C mode, fully implements all slave functions except general call support and provides interrupts on Start and Stop bits in hardware to facilitate firmware implementations of the master functions. The SSP module implements the standard mode specifications, as well as 7-bit and 10-bit addressing.

Two pins are used for data transfer. These are the SCK/SCL pin, which is the clock (SCL), and the SDI/SDA pin, which is the data (SDA). The user must configure these pins as inputs or outputs through the TRISC<5:4> or TRISD<3:2> bits.

The SSP module functions are enabled by setting SSP Enable bit SSPEN (SSPCON<5>).

FIGURE 19-5: SSP BLOCK DIAGRAM (I²C™ MODE)



Note 1: When SSPMX = `1` in CONFIG3H:
SCK/SCL is multiplexed to the RC5 pin, SDA/SDI is multiplexed to the RC4 pin and SDO is multiplexed to pin, RC7.

When SSPMX = `0` in CONFIG3H:
SCK/SCL is multiplexed to the RD3 pin, SDA/SDI is multiplexed to the RD2 pin and SDO is multiplexed to pin, RD1.

The SSP module has five registers for I²C operation. These are the:

- SSP Control Register (SSPCON)
- SSP Status Register (SSPSTAT)
- Serial Receive/Transmit Buffer (SSPBUF)
- SSP Shift Register (SSPSR) – Not directly accessible
- SSP Address Register (SSPADD)

The SSPCON register allows control of the I²C operation. Four mode selection bits (SSPCON<3:0>) allow one of the following I²C modes to be selected:

- I²C Slave mode (7-bit address)
- I²C Slave mode (10-bit address)
- I²C Slave mode (7-bit address), with Start and Stop bit interrupts enabled to support Firmware Controlled Master mode
- I²C Slave mode (10-bit address), with Start and Stop bit interrupts enabled to support Firmware Controlled Master mode
- I²C Start and Stop bit interrupts enabled to support Firmware Controlled Master mode; Slave is Idle

Selection of any I²C mode, with the SSPEN bit set, forces the SCL and SDA pins to be open-drain, provided these pins are programmed as inputs by setting the appropriate TRISC or TRISD bits. Pull-up resistors must be provided externally to the SCL and SDA pins for proper operation of the I²C module.

Additional information on SSP I²C operation can be found in the *"PIC® Mid-Range MCU Family Reference Manual"* (DS33023).

19.3.1 SLAVE MODE

In Slave mode, the SCL and SDA pins must be configured as inputs (TRISC<5:4> or TRISD<3:2> set). The SSP module will override the input state with the output data when required (slave-transmitter).

When an address is matched, or the data transfer after an address match is received, the hardware automatically will generate the Acknowledge ($\overline{ACK}$) pulse and then load the SSPBUF register with the received value currently in the SSPSR register.

There are certain conditions that will cause the SSP module not to give this $\overline{ACK}$ pulse. They include (either or both):

a) The Buffer Full bit, BF (SSPSTAT<0>), was set before the transfer was received.

b) The SSP Overflow bit, SSPOV (SSPCON<6>), was set before the transfer was received.

In this case, the SSPSR register value is not loaded into the SSPBUF, but bit, SSPIF (PIR1<3>), is set. Table 19-2 shows what happens when a data transfer byte is received, given the status of bits BF and SSPOV. The shaded cells show the condition where user software did not properly clear the overflow condition. Flag bit, BF, is cleared by reading the SSPBUF register, while bit, SSPOV, is cleared through software.

The SCL clock input must have a minimum high and low for proper operation. The high and low times of the I²C specification, as well as the requirements of the SSP module, are shown in timing Parameter 100 and Parameter 101.

19.3.1.1 Addressing

Once the SSP module has been enabled, it waits for a Start condition to occur. Following the Start condition, the 8 bits are shifted into the SSPSR register. All incoming bits are sampled with the rising edge of the clock (SCL) line. The value of register SSPSR<7:1> is compared to the value of the SSPADD register. The address is compared on the falling edge of the eighth clock (SCL) pulse. If the addresses match, and the BF and SSPOV bits are clear, the following events occur:

a) The SSPSR register value is loaded into the SSPBUF register.
b) The Buffer Full bit, BF, is set.
c) An $\overline{\text{ACK}}$ pulse is generated.
d) SSP Interrupt Flag bit, SSPIF (PIR1<3>), is set (interrupt is generated if enabled) on the falling edge of the ninth SCL pulse.

In 10-Bit Addressing mode, two address bytes need to be received by the slave (Figure 19-7). The five Most Significant bits (MSbs) of the first address byte specify if this is a 10-bit address. Bit R/$\overline{\text{W}}$ (SSPSTAT<2>) must specify a write so the slave device will receive the second address byte. For a 10-bit address, the first byte would equal '`1111 0 A9 A8 0`', where `A9` and `A8` are the two MSbs of the address.

The sequence of events for 10-Bit Addressing mode is as follows, with Steps 7-9 for slave-transmitter:

1. Receive first (high) byte of address (SSPIF, BF and UA bits are set).
2. Update the SSPADD register with second (low) byte of address (clears bit, UA, and releases the SCL line).
3. Read the SSPBUF register (clears bit, BF) and clear flag bit, SSPIF.
4. Receive second (low) byte of address (SSPIF, BF and UA bits are set).
5. Update the SSPADD register with the first (high) byte of address. If match releases SCL line, this will clear bit, UA.
6. Read the SSPBUF register (clears bit, BF) and clear flag bit, SSPIF.
7. Receive Repeated Start condition.
8. Receive first (high) byte of address (SSPIF and BF bits are set).
9. Read the SSPBUF register (clears bit, BF) and clear flag bit, SSPIF.

TABLE 19-2: DATA TRANSFER RECEIVED BYTE ACTIONS

Status Bits as Data Transfer is Received		SSPSR → SSPBUF	Generate $\overline{\text{ACK}}$ Pulse	Set SSPIF Bit (SSP interrupt occurs if enabled)
BF	SSPOV			
0	0	Yes	Yes	Yes
1	0	No	No	Yes
1	1	No	No	Yes
0	1	No	No	Yes

Note: Shaded cells show the conditions where the user software did not properly clear the overflow condition.

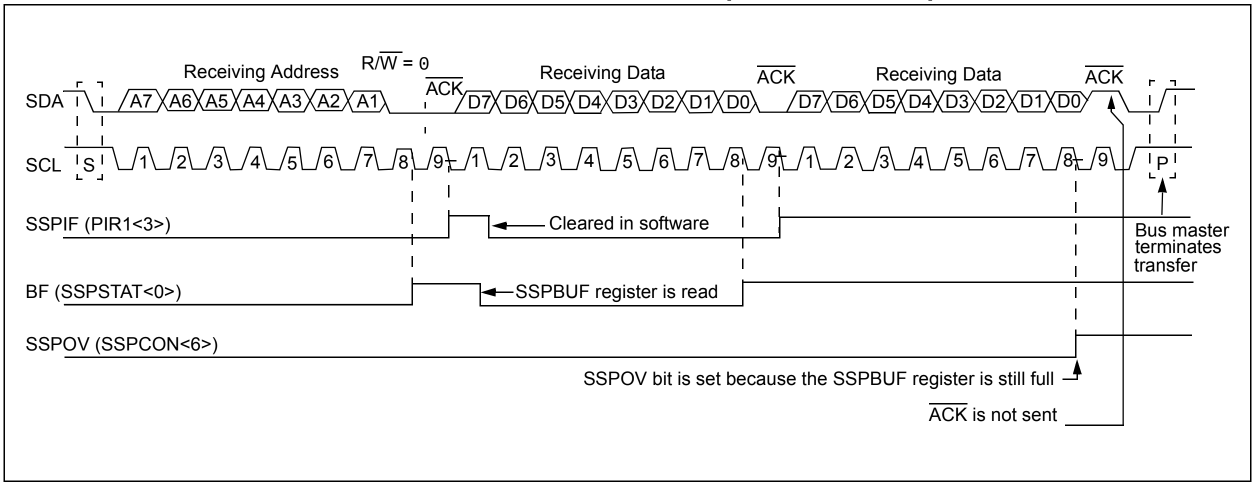
19.3.1.2 Reception

When the R/$\overline{W}$ bit of the address byte is clear and an address match occurs, the R/$\overline{W}$ bit of the SSPSTAT register is cleared. The received address is loaded into the SSPBUF register.

When the address byte overflow condition exists, then the no Acknowledge ($\overline{ACK}$) pulse is given. An overflow condition is defined as either bit BF (SSPSTAT<0>) is set, or bit SSPOV (SSPCON<6>) is set. This is an error condition due to the user's firmware.

An SSP interrupt is generated for each data transfer byte. Flag bit, SSPIF (PIR1<3>), must be cleared in software. The SSPSTAT register is used to determine the status of the byte.

FIGURE 19-6: I²C™ WAVEFORMS FOR RECEPTION (7-BIT ADDRESS)



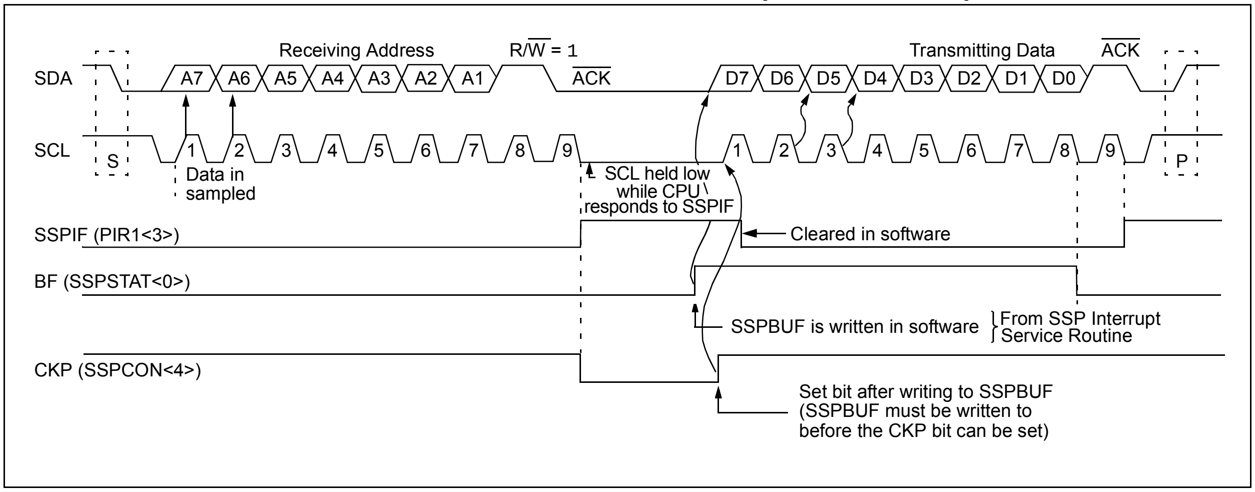
19.3.1.3 Transmission

When the R/$\overline{W}$ bit of the incoming address byte is set and an address match occurs, the R/$\overline{W}$ bit of the SSPSTAT register is set. The received address is loaded into the SSPBUF register. The $\overline{ACK}$ pulse will be sent on the ninth bit and pin, SCK/SCL, is held low. The transmit data must be loaded into the SSPBUF register, which also loads the SSPSR register. Then, pin, SCK/SCL, should be enabled by setting bit, CKP (SSPCON<4>). The master must monitor the SCL pin prior to asserting another clock pulse. The slave devices may be holding off the master by stretching the clock. 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 (Figure 19-7).

An SSP interrupt is generated for each data transfer byte. Flag bit, SSPIF, must be cleared in software and the SSPSTAT register is used to determine the status of the byte. Flag bit, SSPIF, is set on the falling edge of the ninth clock pulse.

As a slave-transmitter, the $\overline{ACK}$ pulse from the master-receiver is latched on the rising edge of the ninth SCL input pulse. If the SDA line was high (not $\overline{ACK}$), then the data transfer is complete. When the $\overline{ACK}$ is latched by the slave, the slave logic is reset and the slave then monitors for another occurrence of the Start bit. If the SDA line was low ($\overline{ACK}$), the transmit data must be loaded into the SSPBUF register, which also loads the SSPSR register. Then pin, SCK/SCL, should be enabled by setting bit CKP.

FIGURE 19-7: I²C™ WAVEFORMS FOR TRANSMISSION (7-BIT ADDRESS)



19.3.2 MASTER MODE

Master mode of operation is supported in firmware using 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 SSP module is disabled. The Stop (P) and Start (S) bits will toggle based on the Start and Stop conditions. Control of the I^2C bus may be taken when the P bit is set, or the bus is Idle and both the S and P bits are clear.

In Master mode, the SCL and SDA lines are manipulated by clearing the corresponding TRISC<5:4> or TRISD<3:2> bits. The output level is always low, regardless of the value(s) in PORTC<5:4> or PORTD<3:2>. So when transmitting data, a '1' data bit must have the TRISC<4> bit set (input) and a '0' data bit must have the TRISC<4> bit cleared (output). The same scenario is true for the SCL line with the TRISC<4> or TRISD<2> bit. Pull-up resistors must be provided externally to the SCL and SDA pins for proper operation of the I^2C module.

The following events will cause the SSP Interrupt Flag bit, SSPIF, to be set (SSP interrupt will occur if enabled):

- Start condition
- Stop condition
- Data transfer byte transmitted/received

Master mode of operation can be done with either the Slave mode Idle (SSPM<3:0> = 1011) or with the Slave active. When both Master and Slave modes are enabled, the software needs to differentiate the source(s) of the interrupt.

19.3.3 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 SSP module is disabled. The Stop (P) and Start (S) bits will toggle based on the Start and Stop conditions. Control of the I^2C bus may be taken when bit P (SSPSTAT<4>) is set, or the bus is Idle and 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 mode, the SDA line must be monitored to see if the signal level is the expected output level. This check only needs to be done when a high level is output. If a high level is expected and a low level is present, the device needs to release the SDA and SCL lines (set TRISC<5:4> or TRISD<3:2>). There are two stages where this arbitration can be lost, these are:

- Address Transfer
- Data Transfer

When the slave logic is enabled, the slave continues to receive. If arbitration was lost during the address transfer stage, communication to the device may be in progress. If addressed, an $\overline{ACK}$ pulse will be generated. If arbitration was lost during the data transfer stage, the device will need to retransfer the data at a later time.

TABLE 19-3: REGISTERS ASSOCIATED WITH I^2C™ OPERATION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
PIR1	—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	57
PIE1	—	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	57
SSPBUF	SSP Receive Buffer/Transmit Register								55
SSPADD	SSP Address Register (I^2C mode)								55
SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	55
SSPSTAT	SMP(1)	CKE(1)	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	55
TRISC(2)	PORTC Data Direction Register								57
TRISD(2)	PORTD Data Direction Register								57

Legend: — = unimplemented, read as '0'. Shaded cells are not used by the SSP module in I^2C mode.

Note 1: Maintain these bits clear in I^2C mode.

2: Depending upon the setting of SSPMX in CONFIG3H, these pins are multiplexed to PORTC or PORTD.

20.0 ENHANCED UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (EUSART)

The Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART) module is one of the two serial I/O modules available in the PIC18F2331/2431/4331/4431 family of microcontrollers. EUSART is also known as a Serial Communications Interface or SCI.

The EUSART can be configured as a full-duplex asynchronous system that can communicate with peripheral devices, such as CRT terminals and personal computers. It can also be configured as a half-duplex synchronous system that can communicate with peripheral devices, such as A/D or D/A integrated circuits, serial EEPROMs, etc.

The EUSART module implements additional features, including automatic baud rate detection and calibration, automatic wake-up on Sync Break reception and 12-bit Break character transmit. These features make it ideally suited for use in Local Interconnect Network (LIN/J2602) bus systems.

The EUSART can be configured in the following modes:

- Asynchronous (full-duplex) with:
  - Auto-wake-up on character reception
  - Auto-baud calibration
  - 12-bit Break character transmission
- Synchronous – Master (half-duplex) with selectable clock polarity
- Synchronous – Slave (half-duplex) with selectable clock polarity

In order to configure pins, TX and RX, as the Enhanced Universal Synchronous Asynchronous Receiver Transmitter:

- SPEN (RCSTA<7>) bit must be set (= 1),
- TRISC<6> bit must be set (= 1), and
- TRISC<7> bit must be set (= 1).

Note:	The EUSART control will automatically reconfigure the pin from input to output as needed.

The operation of the Enhanced USART module is controlled through three registers:

- Transmit Status and Control (TXSTA)
- Receive Status and Control (RCSTA)
- Baud Rate Control (BAUDCON)

These are detailed on the following pages in Register 20-1, Register 20-2 and Register 20-3, respectively.

20.1 Asynchronous Operation in Power-Managed Modes

The EUSART may operate in Asynchronous mode while the peripheral clocks are being provided by the internal oscillator block. This makes it possible to remove the crystal or resonator that is commonly connected as the primary clock on the OSC1 and OSC2 pins.

The factory calibrates the internal oscillator block output (INTOSC) for 8 MHz (see Table 26-6). However, this frequency may drift as VDD or temperature changes, and this directly affects the asynchronous baud rate. Two methods may be used to adjust the baud rate clock, but both require a reference clock source of some kind.

The first (preferred) method uses the OSCTUNE register to adjust the INTOSC output back to 8 MHz. Adjusting the value in the OSCTUNE register allows for fine resolution changes to the system clock source (see **Section 3.6.4 “INTOSC Frequency Drift”** for more information).

The other method adjusts the value in the Baud Rate Generator (BRG). There may not be fine enough resolution when adjusting the Baud Rate Generator to compensate for a gradual change in the peripheral clock frequency.

REGISTER 20-1: TXSTA: TRANSMIT STATUS AND CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN(1)	SYNC	SENDB	BRGH	TRMT	TX9D
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **CSRC:** Clock Source Select bit
Asynchronous mode:
Don’t care.
Synchronous mode:
1 = Master mode (clock generated internally from BRG)
0 = Slave mode (clock from external source)

bit 6 **TX9:** 9-Bit Transmit Enable bit
1 = Selects 9-bit transmission
0 = Selects 8-bit transmission

bit 5 **TXEN:** Transmit Enable bit(1)
1 = Transmit enabled
0 = Transmit disabled

bit 4 **SYNC:** EUSART Mode Select bit
1 = Synchronous mode
0 = Asynchronous mode

bit 3 **SENDB:** Send Break Character bit
Asynchronous mode:
1 = Send Sync Break on next transmission (cleared by hardware upon completion)
0 = Sync Break transmission completed
Synchronous mode:
Don’t care.

bit 2 **BRGH:** High Baud Rate Select bit
Asynchronous mode:
1 = High speed
0 = Low speed
Synchronous mode:
Unused in this mode.

bit 1 **TRMT:** Transmit Shift Register Status bit
1 = TSR is empty
0 = TSR is full

bit 0 **TX9D:** 9th Bit of Transmit Data
Can be address/data bit or a parity bit.

Note 1: SREN/CREN overrides TXEN in Sync mode.

REGISTER 20-2: RCSTA: RECEIVE STATUS AND CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **SPEN:** Serial Port Enable bit
1 = Serial port enabled
0 = Serial port disabled

bit 6 **RX9:** 9-Bit Receive Enable bit
1 = Selects 9-bit reception
0 = Selects 8-bit reception

bit 5 **SREN:** Single Receive Enable bit
Asynchronous mode:
Don’t care.
Synchronous mode – Master:
1 = Enables single receive
0 = Disables single receive
This bit is cleared after reception is complete.
Synchronous mode – Slave:
Don’t care.

bit 4 **CREN:** Continuous Receive Enable bit
Asynchronous mode:
1 = Enables receiver
0 = Disables receiver
Synchronous mode:
1 = Enables continuous receive until enable bit, CREN, is cleared (CREN overrides SREN)
0 = Disables continuous receive

bit 3 **ADDEN:** Address Detect Enable bit
Asynchronous mode 9-Bit (RX9 = 1):
1 = Enables address detection, enables interrupt and loads the receive buffer when RSR<8> is set
0 = Disables address detection, all bytes are received and ninth bit can be used as parity bit
Asynchronous mode 8-Bit (RX9 = 0):
Don’t care.

bit 2 **FERR:** Framing Error bit
1 = Framing error (can be cleared by reading RCREGx register and receiving 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:** 9th Bit of Received Data
This can be address/data bit or a parity bit and must be calculated by user firmware.

REGISTER 20-3: BAUDCON: BAUD RATE CONTROL REGISTER

U-0	R-1	U-0	R/W-1	R/W-0	U-0	R/W-0	R/W-0
—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as ‘0’

bit 6 **RCIDL:** Receive Operation Idle Status bit
1 = Receiver is Idle
0 = Receive in progress

bit 5 **Unimplemented:** Read as ‘0’

bit 4 **SCKP:** Synchronous Clock Polarity Select bit
Asynchronous mode:
Unused in this mode.
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 Register Enable bit
1 = 16-bit Baud Rate Generator – SPBRGH and SPBRG
0 = 8-bit Baud Rate Generator – SPBRG only (Compatible mode), SPBRGH value ignored

bit 2 **Unimplemented:** Read as ‘0’

bit 1 **WUE:** Wake-up Enable bit
Asynchronous mode:
1 = EUSART 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 or rising edge detected
Synchronous mode:
Unused in this mode.

bit 0 **ABDEN:** Auto-Baud Detect Enable bit
Asynchronous mode:
1 = Enable baud rate measurement on the next character – requires reception of a Sync field (55h); cleared in hardware upon completion.
0 = Baud rate measurement disabled or completed
Synchronous mode:
Unused in this mode.

20.2 EUSART Baud Rate Generator (BRG)

The BRG is a dedicated 8-bit or 16-bit generator, that supports both the Asynchronous and Synchronous modes of the EUSART. By default, the BRG operates in 8-bit mode. Setting the BRG16 bit (BAUDCON<3>) selects 16-bit mode.

The SPBRGH:SPBRG register pair controls the period of a free-running timer. In Asynchronous mode, bits BRGH (TXSTA<2>) and BRG16 also control the baud rate. In Synchronous mode, bit BRGH is ignored. Table 20-1 shows the formula for computation of the baud rate for different EUSART modes, which only apply in Master mode (internally generated clock).

Given the desired baud rate and FOSC, the nearest integer value for the SPBRGH:SPBRG registers can be calculated using the formulas in Table 20-1. From this, the error in baud rate can be determined. An example calculation is shown in Example 20-1. Typical baud rates and error values for the various Asynchronous modes are shown in Table 20-2. It may be advantageous to use the high baud rate (BRGH = 1), or the 16-bit BRG, to reduce the baud rate error or achieve a slow baud rate for a fast oscillator frequency.

Writing a new value to the SPBRGH:SPBRG registers causes the BRG timer to be reset (or cleared). This ensures the BRG does not wait for a timer overflow before outputting the new baud rate.

20.2.1 POWER-MANAGED MODE OPERATION

The system clock is used to generate the desired baud rate. However, when a power-managed mode is entered, the clock source may be operating at a different frequency than in PRI_RUN mode. In Sleep mode, no clocks are present and in PRI_IDLE, the primary clock source continues to provide clocks to the Baud Rate Generator. However, in other power-managed modes, the clock frequency will probably change. This may require the value in SPBRG to be adjusted.

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 and make sure that the receive operation is Idle before changing the system clock.

20.2.2 SAMPLING

The data on the RC7/RX/DT/SDO pin is sampled three times by a majority detect circuit to determine if a high or a low level is present at the RX pin.

TABLE 20-1: BAUD RATE FORMULAS

Configuration Bits			BRG/EUSART Mode	Baud Rate Formula
SYNC	BRG16	BRGH		
0	0	0	8-Bit/Asynchronous	FOSC/[64 (n + 1)]
0	0	1	8-Bit/Asynchronous	FOSC/[16 (n + 1)]
0	1	0	16-Bit/Asynchronous	
0	1	1	16-Bit/Asynchronous	FOSC/[4 (n + 1)]
1	0	x	8-Bit/Synchronous	
1	1	x	16-Bit/Synchronous	

Legend: x = Don't care, n = value of SPBRGH:SPBRG register pair

EXAMPLE 20-1: CALCULATING BAUD RATE ERROR

For a device with FOSC of 16 MHz, desired baud rate of 9600, Asynchronous mode, 8-bit BRG:
Desired Baud Rate = FOSC/(64 ([SPBRGH:SPBRG] + 1))
Solving for SPBRGH:SPBRG:
X = ((FOSC/Desired Baud Rate)/64) – 1
= ((16000000/9600)/64) – 1
= [25.042] = 25
Calculated Baud Rate = 16000000/(64 (25 + 1))
= 9615
Error = (Calculated Baud Rate – Desired Baud Rate)/Desired Baud Rate
= (9615 – 9600)/9600 = 0.16%

TABLE 20-2: REGISTERS ASSOCIATED WITH BAUD RATE GENERATOR

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	56
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	56
BAUDCON	—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	56
SPBRGH	EUSART Baud Rate Generator Register High Byte								56
SPBRG	EUSART Baud Rate Generator Register Low Byte								56

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

TABLE 20-3: BAUD RATES FOR ASYNCHRONOUS MODES

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 0											
	FOSC = 40.000 MHz			FOSC = 20.000 MHz			FOSC = 10.000 MHz			FOSC = 8.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	—	—	—	—	—	—	—	—	—	—	—	—
1.2	—	—	—	1.221	1.73	255	1.202	0.16	129	1.201	-0.16	103
2.4	2.441	1.73	255	2.404	0.16	129	2.404	0.16	64	2.403	-0.16	51
9.6	9.615	0.16	64	9.766	1.73	31	9.766	1.73	15	9.615	-0.16	12
19.2	19.531	1.73	31	19.531	1.73	15	19.531	1.73	7	—	—	—
57.6	56.818	-1.36	10	62.500	8.51	4	52.083	-9.58	2	—	—	—
115.2	125.000	8.51	4	104.167	-9.58	2	78.125	-32.18	1	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 0								
	FOSC = 4.000 MHz			FOSC = 2.000 MHz			FOSC = 1.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.16	207	0.300	-0.16	103	0.300	-0.16	51
1.2	1.202	0.16	51	1.201	-0.16	25	1.201	-0.16	12
2.4	2.404	0.16	25	2.403	-0.16	12	—	—	—
9.6	8.929	-6.99	6	—	—	—	—	—	—
19.2	20.833	8.51	2	—	—	—	—	—	—
57.6	62.500	8.51	0	—	—	—	—	—	—
115.2	62.500	-45.75	0	—	—	—	—	—	—

TABLE 20-3: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)

BAUD RATE (K)	SYNC = 0, BRGH = 1, BRG16 = 0											
	FOSC = 40.000 MHz			FOSC = 20.000 MHz			FOSC = 10.000 MHz			FOSC = 8.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
2.4	—	—	—	—	—	—	2.441	1.73	255	2.403	-0.16	207
9.6	9.766	1.73	255	9.615	0.16	129	9.615	0.16	64	9.615	-0.16	51
19.2	19.231	0.16	129	19.231	0.16	64	19.531	1.73	31	19.230	-0.16	25
57.6	58.140	0.94	42	56.818	-1.36	21	56.818	-1.36	10	55.555	3.55	8
115.2	113.636	-1.36	21	113.636	-1.36	10	125.000	8.51	4	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 1, BRG16 = 0								
	FOSC = 4.000 MHz			FOSC = 2.000 MHz			FOSC = 1.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	—	—	—	—	—	—	0.300	-0.16	207
1.2	1.202	0.16	207	1.201	-0.16	103	1.201	-0.16	51
2.4	2.404	0.16	103	2.403	-0.16	51	2.403	-0.16	25
9.6	9.615	0.16	25	9.615	-0.16	12	—	—	—
19.2	19.231	0.16	12	—	—	—	—	—	—
57.6	62.500	8.51	3	—	—	—	—	—	—
115.2	125.000	8.51	1	—	—	—	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 1											
	FOSC = 40.000 MHz			FOSC = 20.000 MHz			FOSC = 10.000 MHz			FOSC = 8.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.00	8332	0.300	0.02	4165	0.300	0.02	2082	0.300	-0.04	1665
1.2	1.200	0.02	2082	1.200	-0.03	1041	1.200	-0.03	520	1.201	-0.16	415
2.4	2.402	0.06	1040	2.399	-0.03	520	2.404	0.16	259	2.403	-0.16	207
9.6	9.615	0.16	259	9.615	0.16	129	9.615	0.16	64	9.615	-0.16	51
19.2	19.231	0.16	129	19.231	0.16	64	19.531	1.73	31	19.230	-0.16	25
57.6	58.140	0.94	42	56.818	-1.36	21	56.818	-1.36	10	55.555	3.55	8
115.2	113.636	-1.36	21	113.636	-1.36	10	125.000	8.51	4	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 1								
	FOSC = 4.000 MHz			FOSC = 2.000 MHz			FOSC = 1.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.04	832	0.300	-0.16	415	0.300	-0.16	207
1.2	1.202	0.16	207	1.201	-0.16	103	1.201	-0.16	51
2.4	2.404	0.16	103	2.403	-0.16	51	2.403	-0.16	25
9.6	9.615	0.16	25	9.615	-0.16	12	—	—	—
19.2	19.231	0.16	12	—	—	—	—	—	—
57.6	62.500	8.51	3	—	—	—	—	—	—
115.2	125.000	8.51	1	—	—	—	—	—	—

TABLE 20-3: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)

BAUD RATE (K)	SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1											
	FOSC = 40.000 MHz			FOSC = 20.000 MHz			FOSC = 10.000 MHz			FOSC = 8.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.00	33332	0.300	0.00	16665	0.300	0.00	8332	0.300	-0.01	6665
1.2	1.200	0.00	8332	1.200	0.02	4165	1.200	0.02	2082	1.200	-0.04	1665
2.4	2.400	0.02	4165	2.400	0.02	2082	2.402	0.06	1040	2.400	-0.04	832
9.6	9.606	0.06	1040	9.596	-0.03	520	9.615	0.16	259	9.615	-0.16	207
19.2	19.193	-0.03	520	19.231	0.16	259	19.231	0.16	129	19.230	-0.16	103
57.6	57.803	0.35	172	57.471	-0.22	86	58.140	0.94	42	57.142	0.79	34
115.2	114.943	-0.22	86	116.279	0.94	42	113.636	-1.36	21	117.647	-2.12	16

BAUD RATE (K)	SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1								
	FOSC = 4.000 MHz			FOSC = 2.000 MHz			FOSC = 1.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.01	3332	0.300	-0.04	1665	0.300	-0.04	832
1.2	1.200	0.04	832	1.201	-0.16	415	1.201	-0.16	207
2.4	2.404	0.16	415	2.403	-0.16	207	2.403	-0.16	103
9.6	9.615	0.16	103	9.615	-0.16	51	9.615	-0.16	25
19.2	19.231	0.16	51	19.230	-0.16	25	19.230	-0.16	12
57.6	58.824	2.12	16	55.555	3.55	8	—	—	—
115.2	111.111	-3.55	8	—	—	—	—	—	—

20.2.3 AUTO-BAUD RATE DETECT

The Enhanced USART module supports the automatic detection and calibration of baud rate. This feature is active only in Asynchronous mode and while the WUE bit is clear.

The automatic baud rate measurement sequence (Figure 20-1) begins whenever a Start bit is received and the ABDEN bit is set. The calculation is self-averaging.

In the Auto-Baud Rate 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. In ABD mode, the internal Baud Rate Generator is used as a counter to time the bit period of the incoming serial byte stream.

Once the ABDEN bit is set, the state machine will clear the BRG and look for a Start bit. The Auto-Baud Detect must receive a byte with the value of 55h (ASCII “U”, which is also the LIN/J2602 bus Sync character) in order to calculate the proper bit rate. The measurement takes over both a low and a high bit time in order to minimize any effects caused by asymmetry of the incoming signal. After a Start bit, the SPBRG begins counting up, using the preselected clock source on the first rising edge of RX. After eight bits on the RX pin, or the fifth rising edge, an accumulated value totalling the proper BRG period is left in the SPBRGH:SPBRG registers. Once the 5th edge is seen (should correspond to the Stop bit), the ABDEN bit is automatically cleared.

While calibrating the baud rate period, the BRG registers are clocked at 1/8th the preconfigured clock rate. The BRG clock can be configured by the BRG16 and BRGH bits. The BRG16 bit must be set to use both SPBRG and SPBRGH as a 16-bit counter.

This allows the user to verify that no carry occurred for 8-bit modes by checking for 00h in the SPBRGH register. Refer to Table 20-4 for counter clock rates to the BRG.

While the ABD sequence takes place, the EUSART state machine is held in Idle. The RCIF interrupt is set once the fifth rising edge on RX is detected. The value in the RCREG needs to be read to clear the RCIF interrupt. RCREG content should be discarded.

Note 1: If the WUE bit is set with the ABDEN bit, Auto-Baud Rate Detection will occur on the byte *following* the Break character (see **Section 20.3.4 “Auto-Wake-up on Sync Break Character”**).

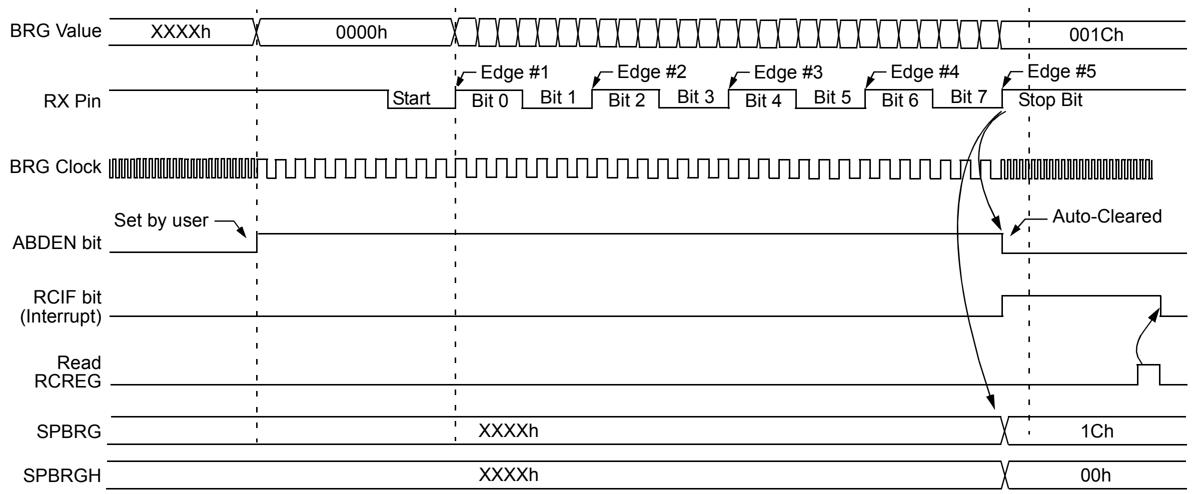
2: It is up to the user to determine that the incoming character baud rate is within the range of the selected BRG clock source. Some combinations of oscillator frequency and EUSART baud rates are not possible due to bit error rates. Overall system timing and communication baud rates must be taken into consideration when using the Auto-Baud Rate Detection feature.

3: To maximize baud rate range, setting the BRG16 bit is recommended if the auto-baud feature is used.

TABLE 20-4: BRG COUNTER CLOCK RATES

BRG16	BRGH	BRG Counter Clock
0	0	FOSC/512
0	1	FOSC/256
1	0	FOSC/128
1	1	FOSC/32

FIGURE 20-1: AUTOMATIC BAUD RATE CALCULATION(1)



Note 1: The ABD sequence requires the EUSART module to be configured in Asynchronous mode and WUE = 0.

20.3 EUSART Asynchronous Mode

The Asynchronous mode of operation is selected by clearing the SYNC bit (TXSTA<4>). In this mode, the EUSART uses standard Non-Return-to-Zero (NRZ) format (one Start bit, eight or nine data bits and one Stop bit). The most common data format is 8 bits. An on-chip dedicated 8-bit/16-bit Baud Rate Generator can be used to derive standard baud rate frequencies from the oscillator.

The EUSART transmits and receives the LSb first. The EUSART's transmitter and receiver are functionally independent, but use the same data format and baud rate. The Baud Rate Generator produces a clock, either x16 or x64 of the bit shift rate, depending on the BRGH and BRG16 bits (TXSTA<2> and BAUDCON<3>). Parity is not supported by the hardware but can be implemented in software and stored as the 9th data bit.

Asynchronous mode is available in all Low-Power modes; it is available in Sleep mode only when Auto-Wake-up on Sync Break is enabled. When in PRI_IDLE mode, no changes to the Baud Rate Generator values are required; however, other Low-Power mode clocks may operate at another frequency than the primary clock. Therefore, the Baud Rate Generator values may need to be adjusted.

When operating in Asynchronous mode, the EUSART module consists of the following important elements:

- Baud Rate Generator
- Sampling Circuit
- Asynchronous Transmitter
- Asynchronous Receiver
- Auto-Wake-up on Sync Break Character
- 12-Bit Break Character Transmit
- Auto-Baud Rate Detection

20.3.1 EUSART ASYNCHRONOUS TRANSMITTER

The EUSART transmitter block diagram is shown in Figure 20-2. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The Shift register obtains its data from the Read/Write Transmit Buffer register, TXREG. The TXREG register is loaded with data in software. The TSR register is not loaded until the Stop bit has been transmitted from the previous load. As soon as the Stop bit is transmitted, the TSR is loaded with new data from the TXREG register (if available).

Once the TXREG register transfers the data to the TSR register (occurs in one TCY), the TXREG register is empty and flag bit, TXIF (PIR1<4>), is set. This interrupt can be enabled/disabled by setting/clearing enable bit, TXIE (PIE1<4>). Flag bit, TXIF, will be set, regardless of the state of enable bit TXIE and cannot be cleared in software. Flag bit, TXIF, is not cleared immediately upon loading the Transmit Buffer register, TXREG. TXIF becomes valid in the second instruction cycle following the load instruction. Polling TXIF immediately following a load of TXREG will return invalid results.

While flag bit, TXIF, indicates the status of the TXREG register, another bit, TRMT (TXSTA<1>), shows the status of the TSR register. Status bit, TRMT, is a read-only bit, which is set when the TSR register is empty. No interrupt logic is tied to this bit, so the user has to poll this bit in order to determine if the TSR register is empty.

> **Note 1:** The TSR register is not mapped in data memory, so it is not available to the user.
>
> **2:** Flag bit, TXIF, is set when enable bit, TXEN, is set.

To set up an Asynchronous Transmission:

1. Initialize the SPBRGH:SPBRG registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing bit, SYNC, and setting bit, SPEN.
3. If interrupts are desired, set enable bit, TXIE.
4. If 9-bit transmission is desired, set transmit bit, TX9. Can be used as address/data bit.
5. Enable the transmission by setting bit, TXEN, which will also set bit, TXIF.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit, TX9D.
7. Load data to the TXREG register (starts transmission).

If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

FIGURE 20-2: EUSART TRANSMIT BLOCK DIAGRAM

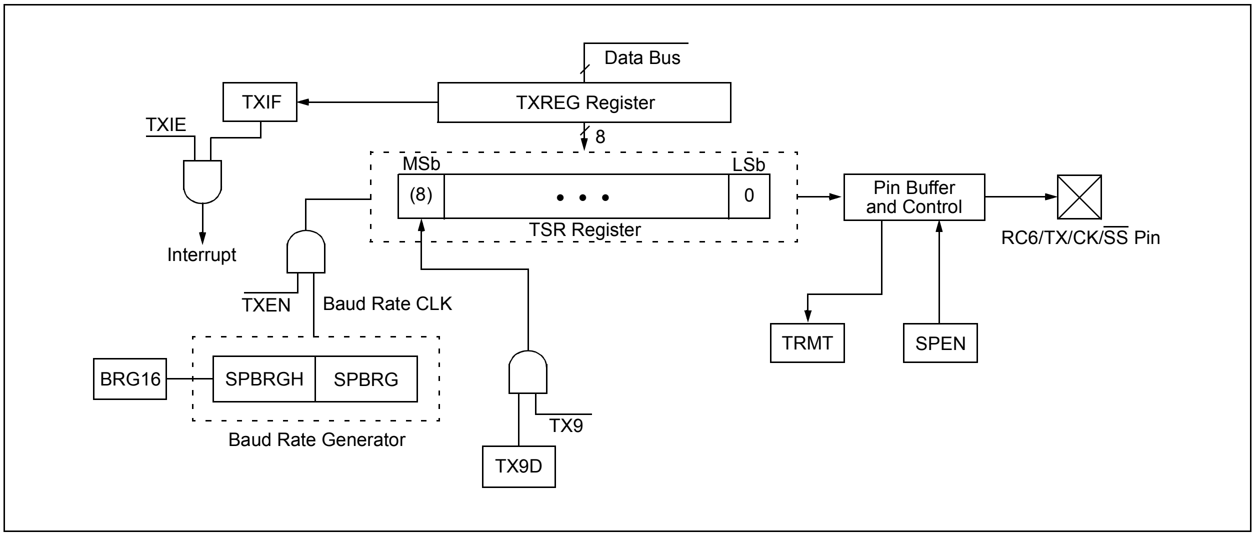


FIGURE 20-3: ASYNCHRONOUS TRANSMISSION

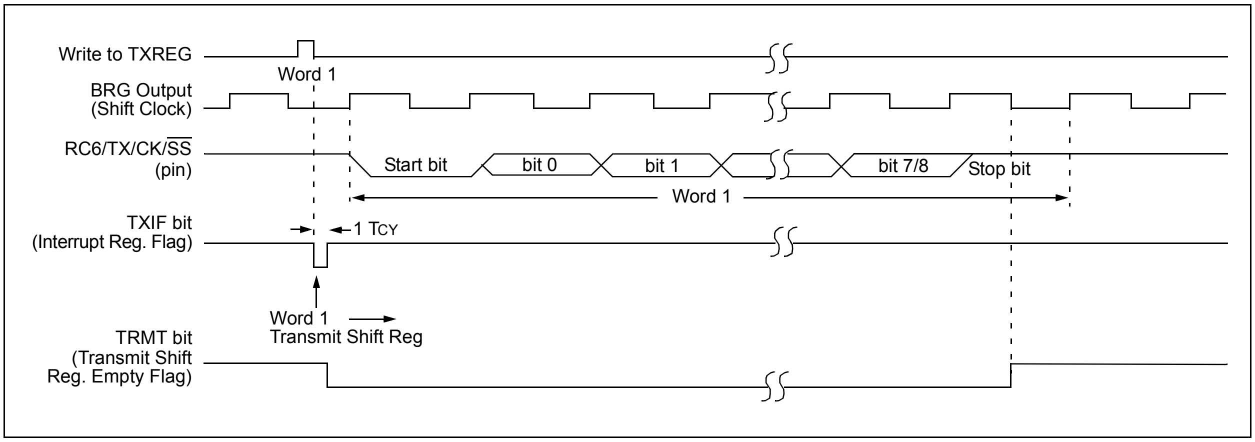
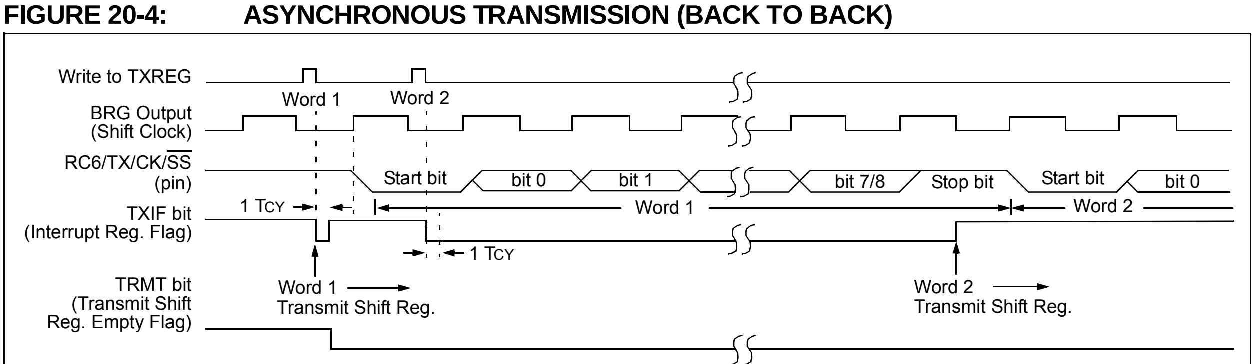


FIGURE 20-4: ASYNCHRONOUS TRANSMISSION (BACK TO BACK)



Note: This timing diagram shows two consecutive transmissions.

TABLE 20-5: REGISTERS ASSOCIATED WITH ASYNCHRONOUS TRANSMISSION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
PIR1	—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	57
PIE1	—	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	57
IPR1	—	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	57
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	56
TXREG	EUSART Transmit Register								56
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	56
BAUDCON	—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	56
SPBRGH	EUSART Baud Rate Generator Register High Byte								56
SPBRG	EUSART Baud Rate Generator Register Low Byte								56

Legend: — = unimplemented, read as ‘0’. Shaded cells are not used for asynchronous transmission.

20.3.2 EUSART ASYNCHRONOUS RECEIVER

The receiver block diagram is shown in Figure 20-5. The data is received on the RC7/RX/DT/SDO pin and drives the data recovery block. The data recovery block is actually a high-speed shifter operating at x16 times the baud rate, whereas the main receive serial shifter operates at the bit rate or at FOSC. This mode would typically be used in RS-232 systems.

To set up an Asynchronous Reception:

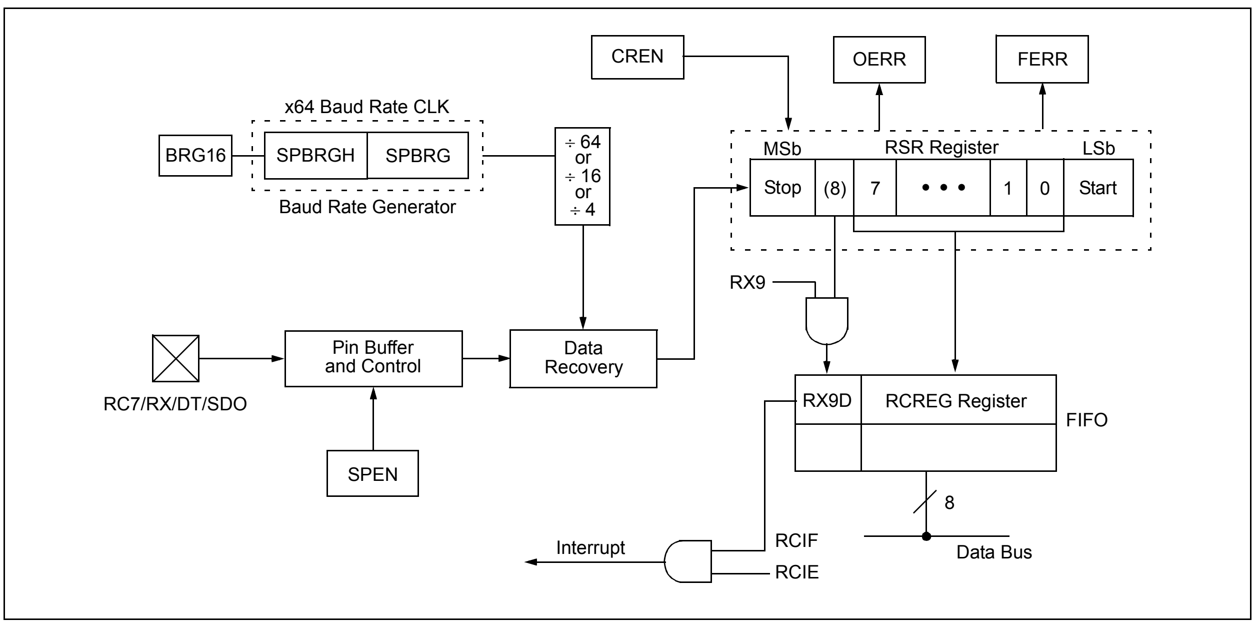
1. Initialize the SPBRGH:SPBRG registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing bit, SYNC, and setting bit, SPEN.
3. If interrupts are desired, set enable bit, RCIE.
4. If 9-bit reception is desired, set bit, RX9.
5. Enable the reception by setting bit, CREN.
6. Flag bit, RCIF, will be set when reception is complete and an interrupt will be generated if enable bit, RCIE, was set.
7. Read the RCSTA register to get the 9th bit (if enabled) and determine if any error occurred during reception.
8. Read the 8-bit received data by reading the RCREG register.
9. If any error occurred, clear the error by clearing enable bit, CREN.
10. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

20.3.3 SETTING UP 9-BIT MODE WITH ADDRESS DETECT

This mode would typically be used in RS-485 systems. To set up an Asynchronous Reception with Address Detect Enable:

1. Initialize the SPBRGH:SPBRG registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing the SYNC bit and setting the SPEN bit.
3. If interrupts are required, set the RCEN bit and select the desired priority level with the RCIP bit.
4. Set the RX9 bit to enable 9-bit reception.
5. Set the ADDEN bit to enable address detect.
6. Enable reception by setting the CREN bit.
7. The RCIF bit will be set when reception is complete. The interrupt will be Acknowledged if the RCIE and GIE bits are set.
8. Read the RCSTA register to determine if any error occurred during reception, as well as read bit 9 of data (if applicable).
9. Read RCREG to determine if the device is being addressed.
10. If any error occurred, clear the CREN bit.
11. If the device has been addressed, clear the ADDEN bit to allow all received data into the receive buffer and interrupt the CPU.

FIGURE 20-5: EUSART RECEIVE BLOCK DIAGRAM

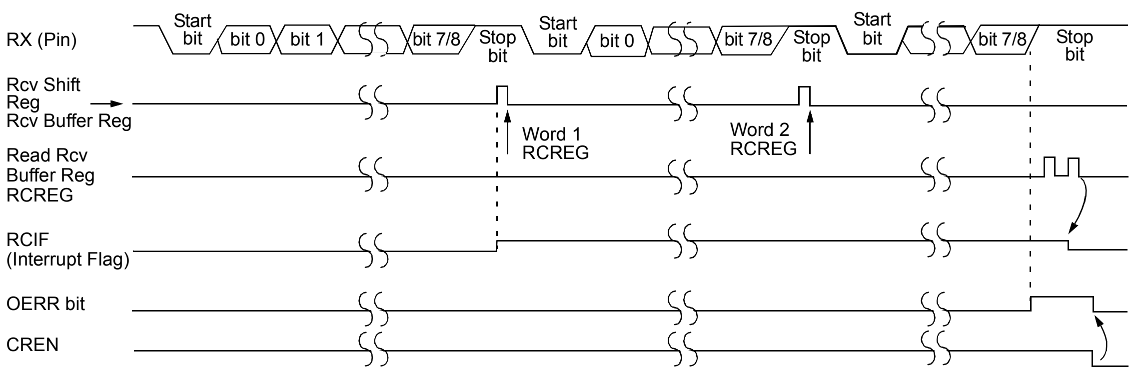


To set up an Asynchronous Transmission:

1. Initialize the SPBRG register for the appropriate baud rate. If a high-speed baud rate is desired, set bit, BRGH (see **Section 20.2 "EUSART Baud Rate Generator (BRG)"**).
2. Enable the asynchronous serial port by clearing bit, SYNC, and setting bit, SPEN.
3. If interrupts are desired, set enable bit, TXIE.
4. If 9-bit transmission is desired, set transmit bit, TX9. Can be used as address/data bit.
5. Enable the transmission by setting bit, TXEN, which will also set bit, TXIF.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit, TX9D.
7. Load data to the TXREG register (starts transmission).

If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

FIGURE 20-6: ASYNCHRONOUS RECEPTION



Note: This timing diagram shows three words appearing on the RX input. The RCREG (Receive Buffer) is read after the third word, causing the OERR (Overrun) bit to be set.

TABLE 20-6: REGISTERS ASSOCIATED WITH ASYNCHRONOUS RECEPTION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
PIR1	—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	57
PIE1	—	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	57
IPR1	—	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	57
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	56
RCREG	EUSART Receive Register								56
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	56
BAUDCON	—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	56
SPBRGH	EUSART Baud Rate Generator Register High Byte								56
SPBRG	EUSART Baud Rate Generator Register Low Byte								56

Legend: — = unimplemented, read as '0'. Shaded cells are not used for asynchronous reception.

20.3.4 AUTO-WAKE-UP ON SYNC BREAK CHARACTER

During Sleep mode, all clocks to the EUSART are suspended. Because of this, the Baud Rate Generator is inactive and a proper byte reception cannot be performed. The auto-wake-up feature allows the controller to wake-up due to activity on the RX/DT line, while the EUSART is operating in Asynchronous mode.

The auto-wake-up feature is enabled by setting the WUE bit (BAUDCON<1>). Once set, the typical 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/J2602 protocol.)

Following a wake-up event, the module generates an RCIF interrupt. The interrupt is generated synchronously to the Q clocks in normal operating modes (Figure 20-7), and asynchronously if the device is in Sleep mode (Figure 20-8). The interrupt condition is cleared by reading the RCREG register.

The WUE bit is automatically cleared once a low-to-high transition is observed on the RX line following the wake-up event. At this point, the EUSART module is in Idle mode and returns to normal operation. This signals to the user that the Sync Break event is over.

20.3.4.1 Special Considerations Using Auto-Wake-up

Since Auto-Wake-up functions by sensing rising edge transitions on RX/DT, information with any state changes before the Stop bit may signal a false end-of-character and cause data or framing errors. To work properly, therefore, the initial characters in the transmission must be all '0's. This can be 00h (8 bits) for standard RS-232 devices, or 000h (12 bits) for LIN/J2602 bus.

Oscillator start-up time must also 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.

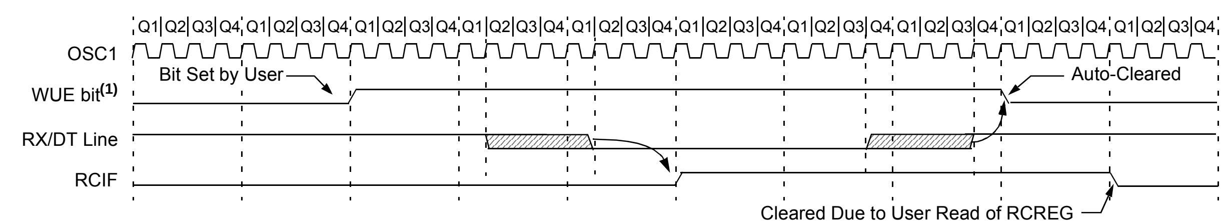
20.3.4.2 Special Considerations Using the WUE Bit

The timing of WUE and RCIF events may cause some confusion when it comes to determining the validity of received data. As noted, setting the WUE bit places the EUSART in an Idle mode. The wake-up event causes a receive interrupt by setting the RCIF bit. The WUE bit is cleared after this when a rising edge is seen on RX/DT. The interrupt condition is then cleared by reading the RCREG register. Ordinarily, the data in RCREG will be dummy data and should be discarded.

The fact that the WUE bit has been cleared (or is still set), and the RCIF flag is set, should not be used as an indicator of the integrity of the data in RCREG. Users should consider implementing a parallel method in firmware to verify received data integrity.

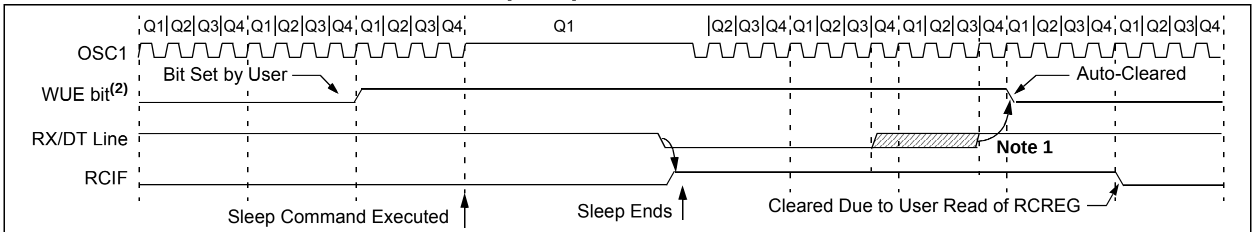
To assure that no actual data is lost, check the RCIDL bit to verify that a receive operation is not in process. If a receive operation is not occurring, the WUE bit may then be set just prior to entering the Sleep mode.

FIGURE 20-7: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING NORMAL OPERATION



Note 1: The EUSART remains in Idle while the WUE bit is set.

FIGURE 20-8: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING SLEEP



Note 1: If the wake-up event requires long oscillator warm-up time, the auto-clear of the WUE bit can occur while the *stposc* signal is still active. This sequence should not depend on the presence of Q clocks.

2: The EUSART remains in Idle while the WUE bit is set.

20.3.5 BREAK CHARACTER SEQUENCE

The Enhanced USART module has the capability of sending the special Break character sequences that are required by the LIN/J2602 bus standard. The Break character transmit consists of a Start bit, followed by twelve '0' bits and a Stop bit. The Frame Break character is sent whenever the SENDB and TXEN bits (TXSTA<3> and TXSTA<5>) are set while the Transmit Shift register is loaded with data. Note that the value of data written to TXREG 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/J2602 specification).

Note that the data value written to the TXREG for the Break character is ignored. The write simply serves the purpose of initiating the proper sequence.

The TRMT bit indicates when the transmit operation is active or Idle, just as it does during normal transmission. See Figure 20-9 for the timing of the Break character sequence.

20.3.5.1 Break and Sync Transmit Sequence

The following sequence will send a message frame header made up of a Break, followed by an Auto-Baud Sync byte. This sequence is typical of a LIN/J2602 bus master.

1. Configure the EUSART for the desired mode.
2. Set the TXEN and SENDB bits to setup the Break character.
3. Load the TXREG with a dummy character to initiate transmission (the value is ignored).
4. Write '55h' to TXREG to load the Sync character into the transmit FIFO buffer.
5. After the Break has been sent, the SENDB bit is reset by hardware. The Sync character now transmits in the preconfigured mode.

When the TXREG becomes empty, as indicated by the TXIF, the next data byte can be written to TXREG.

20.3.6 RECEIVING A BREAK CHARACTER

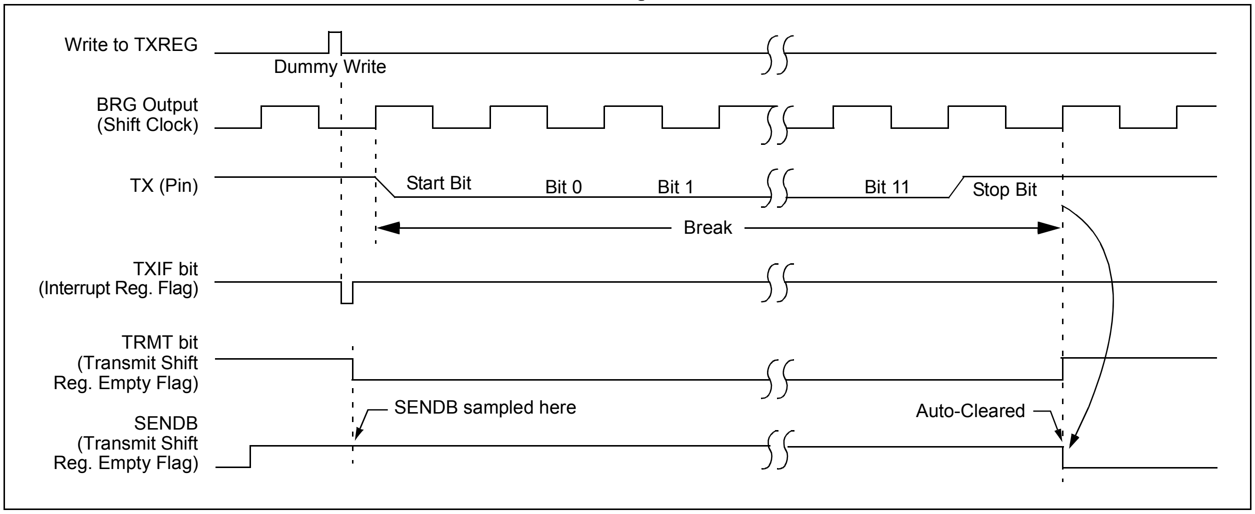
The Enhanced USART module can receive a Break character in two ways.

The first method forces configuration of the baud rate at a frequency of 9/13 of the typical speed. This allows for the Stop bit transition to be at the correct sampling location (13 bits for Break versus Start bit and 8 data bits for typical data).

The second method uses the auto-wake-up feature described in **Section 20.3.4 "Auto-Wake-up on Sync Break Character"**. By enabling this feature, the EUSART will sample the next two transitions on RX/DT, cause an RCIF interrupt and receive the next data byte followed by another interrupt.

Note that following a Break character, the user will typically want to enable the Auto-Baud Rate Detect feature. For both methods, the user can set the ABD bit before placing the EUSART in its Sleep mode.

FIGURE 20-9: SEND BREAK CHARACTER SEQUENCE



20.4 EUSART Synchronous Master Mode

The Synchronous Master mode is entered by setting the CSRC bit (TXSTA<7>). In this mode, the data is transmitted in a half-duplex manner (i.e., transmission and reception do not occur at the same time). When transmitting data, the reception is inhibited and vice versa. Synchronous mode is entered by setting bit SYNC (TXSTA<4>). In addition, enable bit SPEN (RCSTA<7>) is set in order to configure the RC6/TX/CK/SS and RC7/RX/DT/SDO I/O pins to CK (clock) and DT (data) lines, respectively.

The Master mode indicates that the processor transmits the master clock on the CK line. Clock polarity is selected with the SCKP bit (BAUDCON<4>). Setting SCKP sets the Idle state on CK as high, while clearing the bit, sets the Idle state low. This option is provided to support Microwire devices with this module.

20.4.1 EUSART SYNCHRONOUS MASTER TRANSMISSION

The EUSART transmitter block diagram is shown in Figure 20-2. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The Shift register obtains its data from the Read/Write Transmit Buffer register, TXREG. The TXREG register is loaded with data in software. The TSR register is not loaded until the last bit has been transmitted from the previous load. As soon as the last bit is transmitted, the TSR is loaded with new data from the TXREG (if available).

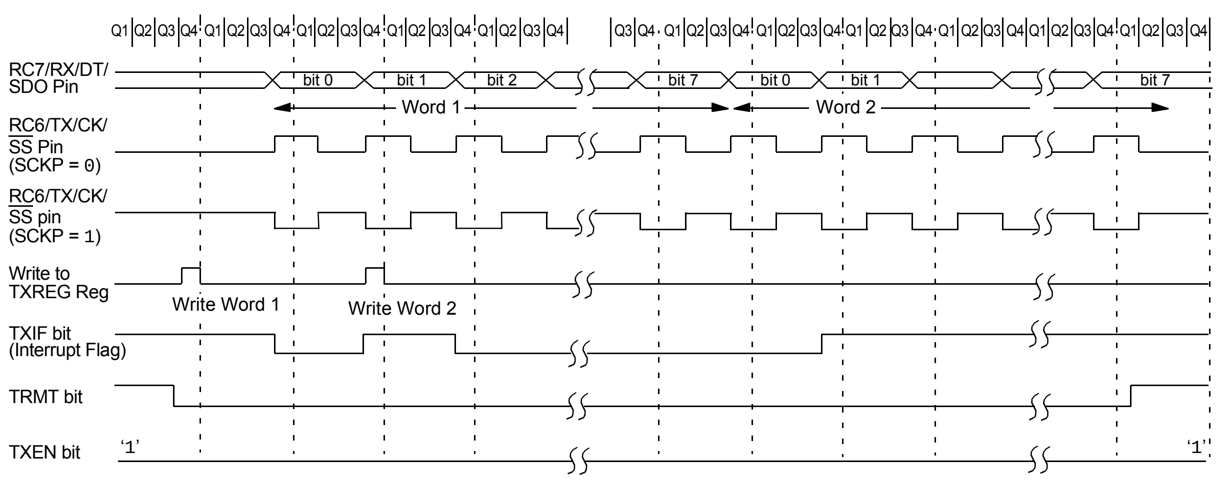
Once the TXREG register transfers the data to the TSR register (occurs in one TCYCLE), the TXREG is empty and interrupt bit, TXIF (PIR1<4>), is set. The interrupt can be enabled/disabled by setting/clearing enable bit, TXIE (PIE1<4>). Flag bit, TXIF, will be set, regardless of the state of enable bit, TXIE, and cannot be cleared in software. It will reset only when new data is loaded into the TXREG register.

While flag bit, TXIF, indicates the status of the TXREG register, another bit, TRMT (TXSTA<1>), shows the status of the TSR register. TRMT is a read-only bit which is set when the TSR is empty. No interrupt logic is tied to this bit, so the user must poll this bit in order to determine if the TSR register is empty. The TSR is not mapped in data memory, so it is not available to the user.

To set up a Synchronous Master Transmission:

1. Initialize the SPBRGH:SPBRG registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the synchronous master serial port by setting bits, SYNC, SPEN and CSRC.
3. If interrupts are desired, set enable bit, TXIE.
4. If 9-bit transmission is desired, set bit, TX9.
5. Enable the transmission by setting bit, TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit, TX9D.
7. Start transmission by loading data to the TXREG register.
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

FIGURE 20-10: SYNCHRONOUS TRANSMISSION



Note: Sync Master mode, SPBRG = 0, continuous transmission of two 8-bit words.

FIGURE 20-11: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)

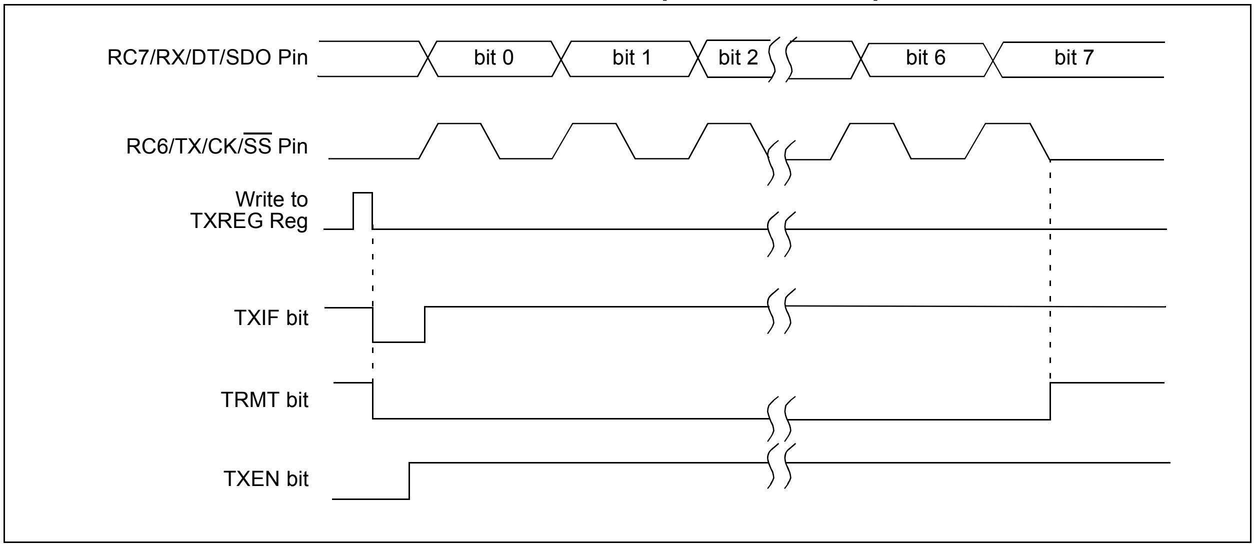


TABLE 20-7: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER TRANSMISSION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
PIR1	—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	57
PIE1	—	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	57
IPR1	—	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	57
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	56
TXREG	EUSART Transmit Register								56
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	56
BAUDCON	—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	56
SPBRGH	EUSART Baud Rate Generator Register High Byte								56
SPBRG	EUSART Baud Rate Generator Register Low Byte								56

Legend: — = unimplemented, read as '0'. Shaded cells are not used for synchronous master transmission.

20.4.2 EUSART SYNCHRONOUS MASTER RECEPTION

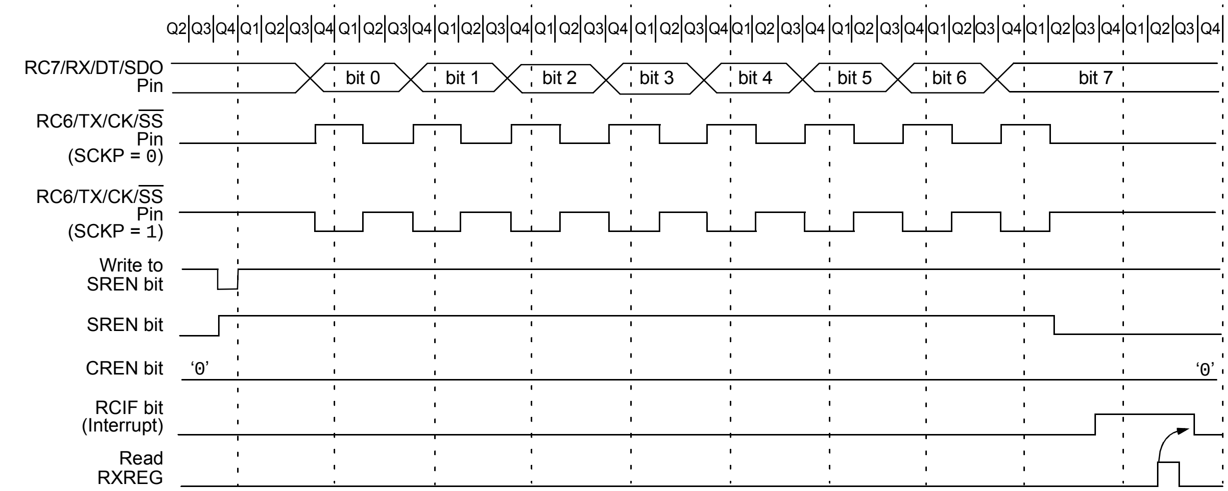
Once Synchronous mode is selected, reception is enabled by setting either the Single Receive Enable bit, SREN (RCSTA<5>), or the Continuous Receive Enable bit, CREN (RCSTA<4>). Data is sampled on the RC7/RX/DT/SDO pin on the falling edge of the clock.

If enable bit SREN is set, only a single word is received. If enable bit CREN is set, the reception is continuous until CREN is cleared. If both bits are set, then CREN takes precedence.

To set up a Synchronous Master Reception:

1. Initialize the SPBRGH:SPBRG registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the synchronous master serial port by setting bits, SYNC, SPEN and CSRC.
3. Ensure bits, CREN and SREN, are clear.
4. If interrupts are desired, set enable bit, RCIE.
5. If 9-bit reception is desired, set bit, RX9.
6. If a single reception is required, set bit, SREN. For continuous reception, set bit, CREN.
7. Interrupt flag bit, RCIF, will be set when reception is complete and an interrupt will be generated if the enable bit, RCIE, was set.
8. Read the RCSTA register to get the 9th bit (if enabled) and determine if any error occurred during reception.
9. Read the 8-bit received data by reading the RCREG register.
10. If any error occurred, clear the error by clearing bit, CREN.
11. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

FIGURE 20-12: SYNCHRONOUS RECEPTION (MASTER MODE, SREN)



Note: Timing diagram demonstrates Sync Master mode with SREN bit = 1 and BRGH bit = 0.

TABLE 20-8: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER RECEPTION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
PIR1	—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	57
PIE1	—	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	57
IPR1	—	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	57
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	56
RCREG	EUSART Receive Register								56
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	56
BAUDCON	—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	56
SPBRGH	EUSART Baud Rate Generator Register High Byte								56
SPBRG	EUSART Baud Rate Generator Register Low Byte								56

Legend: — = unimplemented, read as '0'. Shaded cells are not used for synchronous master reception.

20.5 EUSART Synchronous Slave Mode

Synchronous Slave mode is entered by clearing bit, CSRC (TXSTA<7>). This mode differs from the Synchronous Master mode in that the shift clock is supplied externally at the RC6/TX/CK/$\overline{SS}$ pin (instead of being supplied internally in Master mode). This allows the device to transfer or receive data while in any low-power mode.

20.5.1 EUSART SYNCHRONOUS SLAVE TRANSMIT

The operation of the Synchronous Master and Slave modes are identical, except in the case of Sleep mode.

If two words are written to the TXREG and then the `SLEEP` instruction is executed, the following will occur:

a) The first word will immediately transfer to the TSR register and transmit.
b) The second word will remain in TXREG register.
c) Flag bit, TXIF, will not be set.
d) When the first word has been shifted out of TSR, the TXREG register will transfer the second word to the TSR and flag bit, TXIF, will now be set.
e) If enable bit, TXIE, is set, the interrupt will wake the chip from Sleep. If the global interrupt is enabled, the program will branch to the interrupt vector.

To set up a Synchronous Slave Transmission:

1. Enable the synchronous slave serial port by setting bits, SYNC and SPEN, and clearing bit, CSRC.
2. Clear bits, CREN and SREN.
3. If interrupts are desired, set enable bit, TXIE.
4. If 9-bit transmission is desired, set bit, TX9.
5. Enable the transmission by setting enable bit, TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit, TX9D.
7. Start transmission by loading data to the TXREG register.
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

TABLE 20-9: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE TRANSMISSION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
PIR1	—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	57
PIE1	—	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	57
IPR1	—	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	57
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	56
TXREG	EUSART Transmit Register								56
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	56
BAUDCON	—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	56
SPBRGH	EUSART Baud Rate Generator Register High Byte								56
SPBRG	EUSART Baud Rate Generator Register Low Byte								56

Legend: — = unimplemented, read as ‘0’. Shaded cells are not used for synchronous slave transmission.

20.5.2 EUSART SYNCHRONOUS SLAVE RECEPTION

The operation of the Synchronous Master and Slave modes is identical, except in the case of Sleep, or any Idle mode and bit SREN, which is a “don’t care” in Slave mode.

If receive is enabled by setting the CREN bit prior to entering Sleep or any Idle mode, then a word may be received while in this Low-Power mode. Once the word is received, the RSR register will transfer the data to the RCREG register. If the RCIE enable bit is set, the interrupt generated will wake the chip from Low-Power mode. If the global interrupt is enabled, the program will branch to the interrupt vector.

To set up a Synchronous Slave Reception:

1. Enable the synchronous master serial port by setting bits, SYNC and SPEN, and clearing bit, CSRC.
2. If interrupts are desired, set enable bit, RCIE.
3. If 9-bit reception is desired, set bit, RX9.
4. To enable reception, set enable bit, CREN.
5. Flag bit, RCIF, will be set when reception is complete. An interrupt will be generated if enable bit, RCIE, was set.
6. Read the RCSTA register to get the 9th bit (if enabled) and determine if any error occurred during reception.
7. Read the 8-bit received data by reading the RCREG register.
8. If any error occurred, clear the error by clearing bit, CREN.
9. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

TABLE 20-10: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE RECEPTION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
PIR1	—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	57
PIE1	—	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	57
IPR1	—	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	57
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	56
RCREG	EUSART Receive Register								56
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	56
BAUDCON	—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	56
SPBRGH	EUSART Baud Rate Generator Register High Byte								56
SPBRG	EUSART Baud Rate Generator Register Low Byte								56

Legend: — = unimplemented, read as ‘0’. Shaded cells are not used for synchronous slave reception.

21.0 10-BIT HIGH-SPEED ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

The high-speed Analog-to-Digital (A/D) Converter module allows conversion of an analog signal to a corresponding 10-bit digital number.

The A/D module supports up to 5 input channels on PIC18F2331/2431 devices, and up to 9 channels on the PIC18F4331/4431 devices.

This high-speed 10-bit A/D module offers the following features:

- Up to 200K samples per second
- Two sample and hold inputs for dual-channel simultaneous sampling
- Selectable Simultaneous or Sequential Sampling modes
- 4-word data buffer for A/D results
- Selectable data acquisition timing
- Selectable A/D event trigger
- Operation in Sleep using internal oscillator

These features lend themselves to many applications including motor control, sensor interfacing, data acquisition and process control. In many cases, these features will reduce the software overhead associated with standard A/D modules.

The module has 9 registers:

- A/D Result High Register (ADRESH)
- A/D Result Low Register (ADRESL)
- A/D Control Register 0 (ADCON0)
- A/D Control Register 1 (ADCON1)
- A/D Control Register 2 (ADCON2)
- A/D Control Register 3 (ADCON3)
- A/D Channel Select Register (ADCHS)
- Analog I/O Select Register 0 (ANSEL0)
- Analog I/O Select Register 1 (ANSEL1)

REGISTER 21-1: ADCON0: A/D CONTROL REGISTER 0

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	ACONV	ACSCH	ACMOD1	ACMOD0	GO/$\overline{\text{DONE}}$	ADON
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

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

bit 5 **ACONV**: Auto-Conversion Continuous Loop or Single-Shot Mode Select bit
1 = Continuous Loop mode enabled
0 = Single-Shot mode enabled

bit 4 **ACSCH**: Auto-Conversion Single or Multi-Channel Mode bit
1 = Multi-Channel mode enabled, Single Channel mode disabled
0 = Single Channel mode enabled, Multi-Channel mode disabled

bit 3-2 **ACMOD<1:0>:** Auto-Conversion Mode Sequence Select bits

If ACSCH = 1:
00 = Sequential Mode 1 (SEQM1); two samples are taken in sequence:
  1st sample: Group A(1)
  2nd sample: Group B(1)
01 = Sequential Mode 2 (SEQM2); four samples are taken in sequence:
  1st sample: Group A(1)
  2nd sample: Group B(1)
  3rd sample: Group C(1)
  4th sample: Group D(1)
10 = Simultaneous Mode 1 (STNM1); two samples are taken simultaneously:
  1st sample: Group A and Group B(1)
11 = Simultaneous Mode 2 (STNM2); two samples are taken simultaneously:
  1st sample: Group A and Group B(1)
  2nd sample: Group C and Group D(1)

If ACSCH = 0, Auto-Conversion Single Channel Sequence Mode Enabled:
00 = Single Channel Mode 1 (SCM1); Group A is taken and converted(1)
01 = Single Channel Mode 2 (SCM2); Group B is taken and converted(1)
10 = Single Channel Mode 3 (SCM3); Group C is taken and converted(1)
11 = Single Channel Mode 4 (SCM4); Group D is taken and converted(1)

bit 1 **GO/$\overline{\text{DONE}}$**: A/D Conversion Status bit
1 = A/D conversion cycle in progress. Setting this bit starts the A/D conversion cycle. If Auto-Conversion Single-Shot mode is enabled (ACONV = 0), this bit is automatically cleared by hardware when the A/D conversion (single or multi-channel depending on ACMOD settings) has completed. If Auto-Conversion Continuous Loop mode is enabled (ACONV = 1), this bit remains set after the user/trigger has set it (continuous conversions). It may be cleared manually by the user to stop the conversions.
0 = A/D conversion or multiple conversions completed/not in progress

bit 0 **ADON:** A/D On bit
1 = A/D Converter module is enabled (after brief power-up delay, starts continuous sampling)
0 = A/D Converter module is disabled

Note 1: Groups A, B, C, and D refer to the ADCHS register.

REGISTER 21-2: ADCON1: A/D CONTROL REGISTER 1

R/W-0	R/W-0	U-0	R/W-0	R-0	R-0	R-0	R-0
VCFG1	VCFG0	—	FIFOEN	BFEMT	BFOVL	ADPNT1	ADPNT0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-6 **VCFG<1:0>:** A/D VREF+ and A/D VREF- Source Selection bits
00 = VREF+ = AVDD, VREF- = AVSS (AN2 and AN3 are analog inputs or digital I/O)
01 = VREF+ = External VREF+, VREF- = AVSS (AN2 is an analog input or digital I/O)
10 = VREF+ = AVDD, VREF- = External VREF- (AN3 is an analog input or digital I/O)
11 = VREF+ = External VREF-, VREF- = External VREF-

bit 5 **Unimplemented:** Read as ‘0’

bit 4 **FIFOEN**: FIFO Buffer Enable bit
1 = FIFO is enabled
0 = FIFO is disabled

bit 3 **BFEMT**: Buffer Empty bit
1 = FIFO is empty
0 = FIFO is not empty (at least one of four locations has unread A/D result data)

bit 2 **BFOVFL**: Buffer Overflow bit
1 = A/D result has overwritten a buffer location that has unread data
0 = A/D result has not overflowed

bit 1-0 **ADPNT<1:0>**: Buffer Read Pointer Location bits
Designates the location to be read next.
00 = Buffer Address 0
01 = Buffer Address 1
10 = Buffer Address 2
11 = Buffer Address 3

REGISTER 21-3: ADCON2: A/D CONTROL REGISTER 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	ACQT3	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **ADFM:** A/D Result Format Select bit
1 = Right justified
0 = Left justified

bit 6-3 **ACQT<3:0>**: A/D Acquisition Time Select bits
0000 = No delay (conversion starts immediately when GO/$\overline{\text{DONE}}$ is set)(1)
0001 = 2 TAD
0010 = 4 TAD
0011 = 6 TAD
0100 = 8 TAD
0101 = 10 TAD
0110 = 12 TAD
0111 = 16 TAD
1000 = 20 TAD
1001 = 24 TAD
1010 = 28 TAD
1011 = 32 TAD
1100 = 36 TAD
1101 = 40 TAD
1110 = 48 TAD
1111 = 64 TAD

bit 2-0 **ADCS<2:0>:** A/D Conversion Clock Select bits
000 = FOSC/2
001 = FOSC/8
010 = FOSC/32
011 = FRC/4
100 = FOSC/4
101 = FOSC/16
110 = FOSC/64
111 = FRC (Internal A/D RC Oscillator)

Note 1: If the A/D RC clock source is selected, a delay of one TCY (instruction cycle) is added before the A/D clock starts. This allows the SLEEP instruction to be executed before starting a conversion.

REGISTER 21-4: ADCON3: A/D CONTROL REGISTER 3

R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADRS1	ADRS0	—	SSRC4(1)	SSRC3(1)	SSRC2(1)	SSRC1(1)	SSRC0(1)
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-6 **ADRS<1:0>**: A/D Result Buffer Depth Interrupt Select Control for Continuous Loop Mode bits
The ADRS bits are ignored in Single-Shot mode.
00 = Interrupt is generated when each word is written to the buffer
01 = Interrupt is generated when the 2nd and 4th words are written to the buffer
10 = Interrupt is generated when the 4th word is written to the buffer
11 = Unimplemented

bit 5 **Unimplemented:** Read as ‘0’

bit 4-0 **SSRC<4:0>:** A/D Trigger Source Select bits(1)
00000 = All triggers disabled
xxxx1 = External interrupt RC3/INT0 starts A/D sequence
xxx1x = Timer5 starts A/D sequence
xx1xx = Input Capture 1 (IC1) starts A/D sequence
x1xxx = CCP2 compare match starts A/D sequence
1xxxx = Power Control PWM module rising edge starts A/D sequence

Note 1: The SSRC<4:0> bits can be set such that any of the triggers will start a conversion (e.g., SSRC<4:0> = 00101 will trigger the A/D conversion sequence when RC3/INT0 or Input Capture 1 event occurs).

REGISTER 21-5: ADCHS: A/D CHANNEL SELECT 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
GDSEL1	GDSEL0	GBSEL1	GBSEL0	GCSEL1	GCSEL0	GASEL1	GASEL0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-6 **GDSEL<1:0>**: Group D Select bits
S/H-2 positive input.
`00` = AN3
`01` = AN7(1)
`1x` = Reserved

bit 5-4 **GBSEL<1:0>**: Group B Select bits
S/H-2 positive input.
`00` = AN1
`01` = AN5(1)
`1x` = Reserved

bit 3-2 **GCSEL<1:0>**: Group C Select bits
S/H-1 positive input.
`00` = AN2
`01` = AN6(1)
`1x` = Reserved

bit 1-0 **GASEL<1:0>**: Group A Select bits
S/H-1 positive input.
`00` = AN0
`01` = AN4
`10` = AN8(1)
`11` = Reserved

Note 1: AN5 through AN8 are available only in PIC18F4331/4431 devices.

REGISTER 21-6: ANSEL0: ANALOG SELECT REGISTER 0(1)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
ANS7(2)	ANS6(2)	ANS5(2)	ANS4	ANS3	ANS2	ANS1	ANS0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-0 **ANS<7:0>:** Analog Input Function Select bits
Correspond to pins, AN<7:0>.
1 = Analog input
0 = Digital I/O

Note 1: Setting a pin to an analog input disables the digital input buffer. The corresponding TRIS bit should be set for an input and cleared for an output (analog or digital). The ANSx bits directly correspond to the ANx pins (e.g., ANS0 = AN0, ANS1 = AN1, etc.). Unused ANSx bits are read as '0'.
2: ANS7 through ANS5 are available only on PIC18F4331/4431 devices.

REGISTER 21-7: ANSEL1: ANALOG SELECT REGISTER 1(1)

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-1
—	—	—	—	—	—	—	ANS8(2)
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-1 **Unimplemented:** Read as '0'
bit 0 **ANS8:** Analog Input Function Select bit(2)
1 = Analog input
0 = Digital I/O

Note 1: Setting a pin to an analog input disables the digital input buffer. The corresponding TRIS bit should be set for an input and cleared for an output (analog or digital). The ANSx bits directly correspond to the ANx pins (e.g., ANS8 = AN8, ANS9 = AN9, etc.). Unused ANSx bits are read as '0'.
2: ANS8 is available only on PIC18F4331/4431 devices.

The A/D channels are grouped into four sets of 2 or 3 channels. For the PIC18F2331/2431 devices, AN0 and AN4 are in Group A, AN1 is in Group B, AN2 is in Group C and AN3 is in Group D. For the PIC18F4331/4431 devices, AN0, AN4 and AN8 are in Group A, AN1 and AN5 are in Group B, AN2 and AN6 are in Group C and AN3 and AN7 are in Group D. The selected channel in each group is selected by configuring the A/D Channel Select Register, ADCHS.

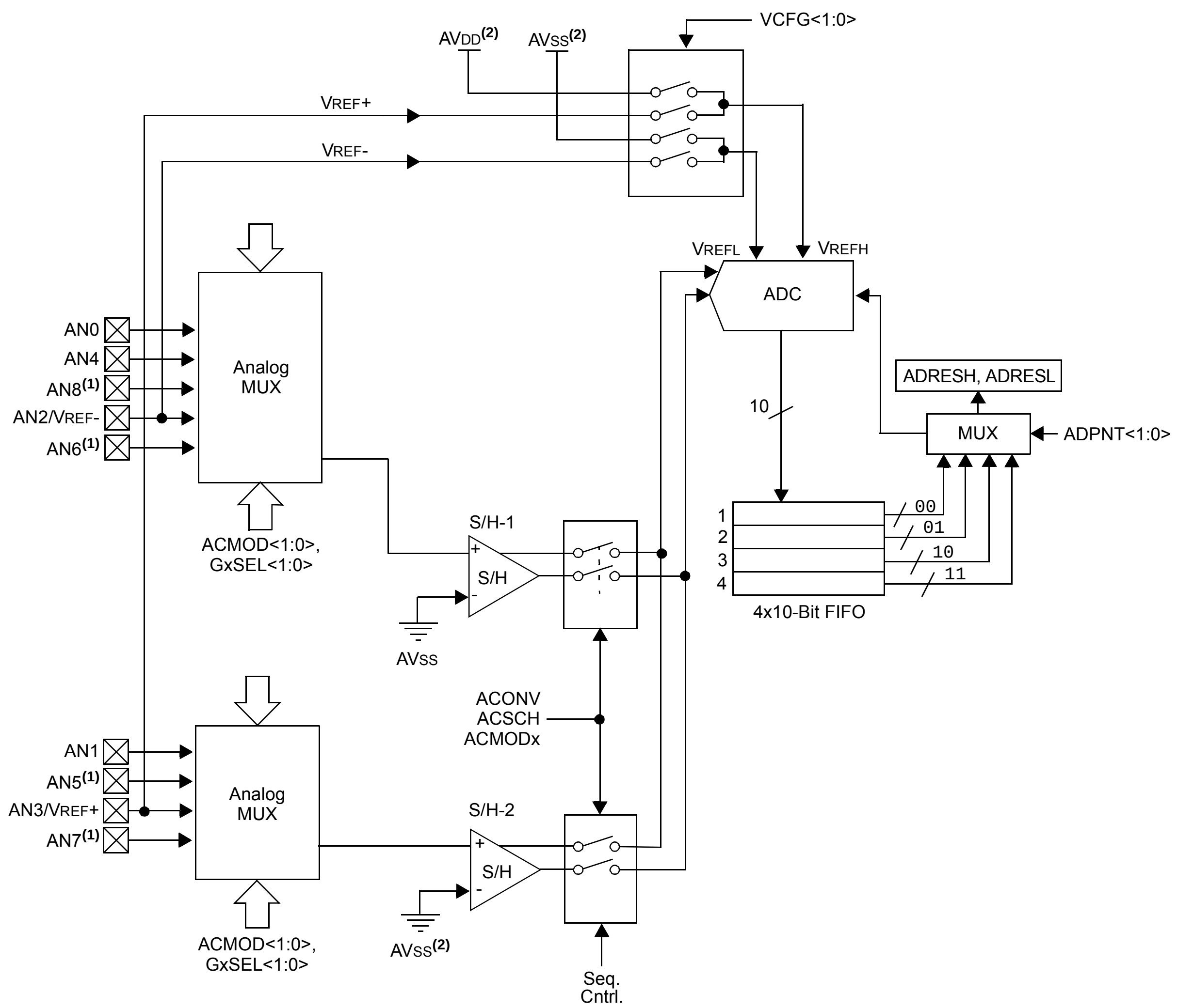
The analog voltage reference is software selectable to either the device's positive and negative analog supply voltage (AVDD and AVSS), or the voltage level on the RA3/AN3/VREF+/CAP2/QEA and RA2/AN2/VREF-/CAP1/INDX, or some combination of supply and external sources. Register ADCON1 controls the voltage reference settings.

The A/D Converter has a unique feature of being able to operate while the device is in Sleep mode. To operate in Sleep, the A/D conversion clock must be derived from the A/D's internal RC oscillator.

A device Reset forces all registers to their Reset state. This forces the A/D module to be turned off and any conversion in progress is aborted.

Each port pin associated with the A/D Converter can individually be configured as an analog input or digital I/O using the ANSEL0 and ANSEL1 registers. The ADRESH and ADRESL registers contain the value in the result buffer pointed to by ADPNT<1:0> (ADCON1<1:0>). The result buffer is a 4-deep circular buffer that has a Buffer Empty status bit, BFEMT (ADCON1<3>), and a Buffer Overflow status bit, BFOVFL (ADCON1<2>).

FIGURE 21-1: A/D BLOCK DIAGRAM



Note 1: AN5 through AN8 are available only on PIC18F4331/4431 devices.

2: I/O pins have diode protection to VDD and VSS.

21.1 Configuring the A/D Converter

The A/D Converter has two types of conversions, two modes of operation and eight different Sequencing modes. These features are controlled by the ACONV bit (ADCON0<5>), ACSCH bit (ADCON0<4>) and ACMOD<1:0> bits (ADCON0<3:2>). In addition, the A/D channels are divided into four groups as defined in the ADCHS register. Table 21-1 shows the sequence configurations as controlled by the ACSCH and ACMOD<1:0> bits.

21.1.1 CONVERSION TYPE

Two types of conversions exist in the high-speed 10-bit A/D Converter module that are selected using the ACONV bit. Single-Shot mode allows a single conversion or sequence to be enabled when ACONV = 0. At the end of the sequence, the GO/DONE bit will be automatically cleared and the interrupt flag, ADIF, will be set. When using Single-Shot mode and configured for Simultaneous mode, STNM2, acquisition time must be used to ensure proper conversion of the analog input signals.

Continuous Loop mode allows the defined sequence to be executed in a continuous loop when ACONV = 1. In this mode, either the user can trigger the start of conversion by setting the GO/DONE bit, or one of the A/D triggers can start the conversion. The interrupt flag, ADIF, is set based on the configuration of the bits, ADRS<1:0> (ADCON3<7:6>). In Simultaneous modes, STNM1 and STNM2 acquisition time must be configured to ensure proper conversion of the analog input signals.

21.1.2 CONVERSION MODE

The ACSCH bit (ADCON0<4>) controls how many channels are used in the configured sequence. When clear, the A/D is configured for single channel conversion and will convert the group selected by the ACMOD<1:0> bits and the channel selected by the GxSEL<1:0> bits (ADCHS register). When ACSCH = 1, the A/D is configured for multiple channel conversion and the sequence is defined by ACMOD<1:0>.

TABLE 21-1: AUTO-CONVERSION SEQUENCE CONFIGURATIONS

Mode	ACSCH	ACMOD<1:0>	Description
Multi-Channel Sequential Mode 1 (SEQM1)	1	00	Groups A and B are sampled and converted sequentially.
Multi-Channel Sequential Mode 2 (SEQM2)	1	01	Groups A, B, C and D are sampled and converted sequentially.
Multi-Channel Simultaneous Mode 1 (STNM1)	1	10	Groups A and B are sampled simultaneously and converted sequentially.
Multi-Channel Simultaneous Mode 2 (STNM2)	1	11	Groups A and B are sampled simultaneously, then converted sequentially. Then, Group C and D are sampled simultaneously, then converted sequentially.
Single Channel Mode 1 (SCM1)	0	00	Group A is sampled and converted.
Single Channel Mode 2 (SCM2)	0	01	Group B is sampled and converted.
Single Channel Mode 3 (SCM3)	0	10	Group C is sampled and converted.
Single Channel Mode 4 (SCM4)	0	11	Group D is sampled and converted.

21.1.3 CONVERSION SEQUENCING

The ACMOD<1:0> bits control the sequencing of the A/D conversions. When ACSCH = `0`, the A/D is configured to sample and convert a single channel. The ACMOD bits select which group to perform the conversions and the GxSEL<1:0> bits select which channel in the group is to be converted. If Single-Shot mode is enabled, the A/D interrupt flag will be set after the channel is converted. If Continuous Loop mode is enabled, the A/D interrupt flag will be set according to the ADRS<1:0> bits.

When ACSCH = `1`, multiple channel sequencing is enabled and two submodes can be selected. The first mode is Sequential mode with two settings. The first setting is called SEQM1, and first samples and converts the selected Group A channel, and then samples and converts the selected Group B channel. The second mode is called SEQM2, and it samples and converts a Group A channel, Group B channel, Group C channel and finally, a Group D channel.

The second multiple channel sequencing submode is Simultaneous Sampling mode. In this mode, there are also two settings. The first setting is called STNM1, and uses the two sample and hold circuits on the A/D module. The selected Group A and B channels are simultaneously sampled and then the Group A channel is converted followed by the conversion of the Group B channel. The second setting is called STNM2, and starts the same as STNM1, but follows it with a simultaneous sample of Group C and D channels. The A/D module will then convert the Group C channel followed by the Group D channel.

21.1.4 TRIGGERING A/D CONVERSIONS

The PIC18F2331/2431/4331/4431 devices are capable of triggering conversions from many different sources. The same method used by all other microcontrollers of setting the GO/$\overline{\text{DONE}}$ bit still works. The other trigger sources are:

- RC3/INT0 Pin
- Timer5 Overflow
- Input Capture 1 (IC1)
- CCP2 Compare Match
- Power Control PWM Rising Edge

These triggers are enabled using the SSRC<4:0> bits (ADCON3<4:0>). Any combination of the five sources can trigger a conversion by simply setting the corresponding bit in ADCON3. When the trigger occurs, the GO/$\overline{\text{DONE}}$ bit is automatically set by the hardware and then cleared once the conversion completes.

21.1.5 A/D MODULE INITIALIZATION STEPS

The following steps should be followed to initialize the A/D module:

1. Configure the A/D module:
   a) Configure the analog pins, voltage reference and digital I/O.
   b) Select the A/D input channels.
   c) Select the A/D Auto-Conversion mode (Single-Shot or Continuous Loop).
   d) Select the A/D conversion clock.
   e) Select the A/D conversion trigger.
2. Configure the A/D interrupt (if required):
   a) Set the GIE bit.
   b) Set the PEIE bit.
   c) Set the ADIE bit.
   d) Clear the ADIF bit.
   e) Select the A/D trigger setting.
   f) Select the A/D interrupt priority.
3. Turn on ADC:
   a) Set the ADON bit in the ADCON0 register.
   b) Wait the required power-up setup time, about 5-10 μs.
4. Start the sample/conversion sequence:
   a) Sample for a minimum of 2 T_{AD} and start the conversion by setting the GO/$\overline{\text{DONE}}$ bit. The GO/$\overline{\text{DONE}}$ bit is set by the user in software or by the module if initiated by a trigger.
   b) If T_{ACQ} is assigned a value (multiple of T_{AD}), then setting the GO/$\overline{\text{DONE}}$ bit starts a sample period of the T_{ACQ} value, then starts a conversion.
5. Wait for A/D conversion/conversions to complete using one of the following options:
   a) Poll for the GO/$\overline{\text{DONE}}$ bit to be cleared if in Single-Shot mode.
   b) Wait for the A/D Interrupt Flag (ADIF) to be set.
   c) Poll for the BFEMT bit to be cleared to signify that at least the first conversion has completed.
6. Read the A/D results, clear the ADIF flag, reconfigure the trigger.

21.2 A/D Result Buffer

The A/D module has a 4-level result buffer with an address range of 0 to 3, enabled by setting the FIFOEN bit in the ADCON1 register. This buffer is implemented in a circular fashion, where the A/D result is stored in one location and the address is incremented. If the address is greater than 3, the pointer is wrapped back around to 0. The result buffer has a Buffer Empty Flag, BFEMT, indicating when any data is in the buffer. It also has a Buffer Overflow Flag, BFOVFL, which indicates when a new sample has overwritten a location that was not previously read.

Associated with the buffer is a pointer to the address for the next read operation. The ADPNT<1:0> bits configure the address for the next read operation. These bits are read-only.

The Result Buffer also has a configurable interrupt trigger level that is configured by the ADRS<1:0> bits. The user has three selections: interrupt flag set on every write to the buffer, interrupt on every second write to the buffer, or interrupt on every fourth write to the buffer. ADPNT<1:0> are reset to '00' every time a conversion sequence is started (either by setting the GO/$\overline{\text{DONE}}$ bit or on a trigger).

Note:	When right justified, reading ADRESL increments the ADPNT<1:0> bits. When left justified, reading ADRESH increments the ADPNT<1:0> bits.

21.3 A/D Acquisition Requirements

For the A/D Converter to meet its specified accuracy, the charge holding capacitor (C_{HOLD}) must be allowed to fully charge to the input channel voltage level. The analog input model is shown in Figure 21-2. The source impedance (R_S) and the internal sampling switch (R_{SS}) impedance directly affect the time required to charge the capacitor C_{HOLD}. The sampling switch (R_{SS}) impedance varies over the device voltage (V_{DD}). The source impedance affects the offset voltage at the analog input (due to pin leakage current). **The maximum recommended impedance for analog sources is 2.5 kΩ**. After the analog input channel is selected (changed), the channel must be sampled for at least the minimum acquisition time before starting a conversion.

Note:	When the conversion is started, the holding capacitor is disconnected from the input pin.

To calculate the minimum acquisition time, Equation 21-1 may be used. This equation assumes that 1/2 LSb error is used (1024 steps for the A/D). The 1/2 LSb error is the maximum error allowed for the A/D to meet its specified resolution.

Example 21-1 shows the calculation of the minimum required acquisition time T_{ACQ}. In this case, the converter module is fully powered up at the outset and therefore, the amplifier settling time, T_{AMP}, is negligible. This calculation is based on the following application system assumptions:

C_{HOLD}	=	9 pF
Rs	=	100Ω
Conversion Error	≤	1/2 LSb
V_{DD}	=	5V → Rss = 6 kΩ
Temperature	=	50°C (system max.)
V_{HOLD}	=	0V @ time = 0

EQUATION 21-1: ACQUISITION TIME

$$T_{ACQ} = \text{Amplifier Settling Time} + \text{Holding Capacitor Charging Time} + \text{Temperature Coefficient}$$

$$= T_{AMP} + T_C + T_{COFF}$$

EQUATION 21-2: MINIMUM A/D HOLDING CAPACITOR CHARGING TIME

$$V_{HOLD} = (V_{REF} - (V_{REF}/2048)) \cdot (1 - e^{(-T_C/C_{HOLD}(R_{IC} + R_{SS} + R_S))})$$

or

$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S)\ \ln(1/2048)$$

EXAMPLE 21-1: CALCULATING THE MINIMUM REQUIRED ACQUISITION TIME

$T_{ACQ} = T_{AMP} + T_C + T_{COFF}$

T_{AMP} = Negligible

T_{COFF} = (Temp – 25°C)(0.005 µs/°C)
(50°C – 25°C)(0.005 µs/°C) = .13 µs

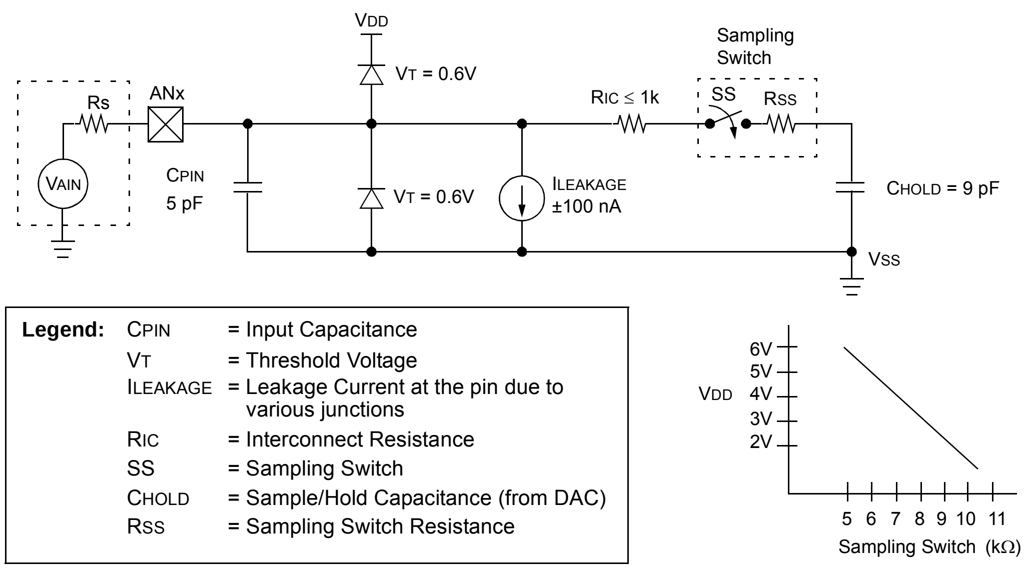
Temperature coefficient is only required for temperatures > 25°C. Below 25°C, T_{COFF} = 0 µs.

T_C = $-(C_{HOLD})(R_{IC} + R_{SS} + R_S)\ln(1/2047)$ µs
-(9 pF) (1 kΩ + 6 kΩ + 100Ω) ln(0.0004883) µs = .49 µs

T_{ACQ} = 0 + .49 µs + .13 µs = .62 µs

Note: If the converter module has been in Sleep mode, T_{AMP} is 2.0 µs from the time the part exits Sleep mode.

FIGURE 21-2: ANALOG INPUT MODEL



Note: For V_{DD} < 2.7V and temperatures below 0°C, V_{AIN} should be restricted to range: $V_{AIN} < V_{DD}/2$.

21.4 A/D Voltage References

If external voltage references are used instead of the internal AVDD and AVSS sources, the source impedance of the VREF+ and VREF- voltage sources must be considered. During acquisition, currents supplied by these sources are insignificant. However, during conversion, the A/D module sinks and sources current through the reference sources.

In order to maintain the A/D accuracy, the voltage reference source impedances should be kept low to reduce voltage changes. These voltage changes occur as reference currents flow through the reference source impedance.

Note:	When using external references, the source impedance of the external voltage references must be less than 75Ω in order to achieve the specified ADC resolution. A higher reference source impedance will increase the ADC offset and gain errors. Resistive voltage dividers will not provide a low enough source impedance. To ensure the best possible ADC performance, external VREF inputs should be buffered with an op amp or other low-impedance circuit.

21.5 Selecting and Configuring Automatic Acquisition Time

The ADCON2 register allows the user to select an acquisition time that occurs each time an A/D conversion is triggered.

When the GO/$\overline{\text{DONE}}$ bit is set, sampling is stopped and a conversion begins. The user is responsible for ensuring the required acquisition time has passed between selecting the desired input channel and the start of conversion. This occurs when the ACQT<3:0> bits (ADCON2<6:3>) remain in their Reset state (‘0000’).

If desired, the ACQT bits can be set to select a programmable acquisition time for the A/D module. When triggered, the A/D module continues to sample the input for the selected acquisition time, then automatically begins a conversion. Since the acquisition time is programmed, there may be no need to wait for an acquisition time between selecting a channel and triggering the A/D. If an acquisition time is programmed, there is nothing to indicate if the acquisition time has ended or if the conversion has begun.

21.6 Selecting the A/D Conversion Clock

The A/D conversion time per bit is defined as TAD. The A/D conversion requires 12 TAD per 10-bit conversion. The source of the A/D conversion clock is software selectable. There are eight possible options for TAD:

- 2 TOSC
- 4 TOSC
- 8 TOSC
- 16 TOSC
- 32 TOSC
- 64 TOSC
- Internal RC Oscillator
- Internal RC Oscillator/4

For correct A/D conversions, the A/D conversion clock (TAD) must be as short as possible, but greater than the minimum TAD (approximately 416 ns, see parameter A11 for more information).

Table 21-2 shows the resultant TAD times derived from the device operating frequencies and the A/D clock source selected.

TABLE 21-2: TAD VS. DEVICE OPERATING FREQUENCIES

AD Clock Source (TAD)		Maximum Device Frequency	
Operation	**ADCS<2:0>**	**PIC18FXX31**	**PIC18LFXX31(4)**
2 TOSC	000	4.8 MHz	666 kHz
4 TOSC	100	9.6 MHz	1.33 MHz
8 TOSC	001	19.2 MHz	2.66 MHz
16 TOSC	101	38.4 MHz	5.33 MHz
32 TOSC	010	40.0 MHz	10.65 MHz
64 TOSC	110	40.0 MHz	21.33 MHz
RC/4(3)	011	1.00 MHz(1)	1.00 MHz(2)
RC(3)	111	4.0 MHz(2)	4.0 MHz(2)

Note 1: The RC source has a typical TAD time of 2-6 μs.
2: The RC source has a typical TAD time of 0.5-1.5 μs.
3: For device frequencies above 1 MHz, the device must be in Sleep for the entire conversion or the A/D accuracy may be out of specification unless in Single-Shot mode.
4: Low-power devices only.

21.7 Operation in Power-Managed Modes

The selection of the automatic acquisition time and A/D conversion clock is determined in part by the clock source and frequency while in a power-managed mode.

If the A/D is expected to operate while the device is in a power-managed mode, the ACQT<3:0> and ADCS<2:0> bits in ADCON2 should be updated in accordance with the power-managed mode clock that will be used. After the power-managed mode is entered (either of the power-managed Run modes), an A/D acquisition or conversion may be started. Once an acquisition or conversion is started, the device should continue to be clocked by the same power-managed mode clock source until the conversion has been completed. If desired, the device may be placed into the corresponding power-managed Idle mode during the conversion.

If the power-managed mode clock frequency is less than 1 MHz, the A/D RC clock source should be selected.

Operation in Sleep mode requires the A/D RC clock to be selected. If bits, ACQT<3:0>, are set to '0000' and a conversion is started, the conversion will be delayed one instruction cycle to allow execution of the SLEEP instruction and entry to Sleep mode. The IDLEN and SCS bits in the OSCCON register must have already been cleared prior to starting the conversion.

Note:	The A/D can operate in Sleep mode only when configured for Single-Shot mode. If the part is in Sleep mode, and it is possible for a source other than the A/D module to wake the part, the user must poll ADCON0<GO/$\overline{\text{DONE}}$> to ensure it is clear before reading the result.

21.8 Configuring Analog Port Pins

The ANSEL0, ANSEL1, TRISA and TRISE registers all configure the A/D port pins. The port pins needed as analog inputs must have their corresponding TRIS bits set (input). If the TRIS bit is cleared (output), the digital output level (V_{OH} or V_{OL}) will be converted.

The A/D operation is independent of the state of the ANSEL0, ANSEL1 and TRIS bits.

Note 1:	When reading the PORT register, all pins configured as analog input channels will read as cleared (a low level). Pins configured as digital inputs will convert an analog input. Analog levels on a digitally configured input will be accurately converted.
2:	Analog levels on any pin defined as a digital input may cause the digital input buffer to consume current out of the device's specification limits.

21.9 A/D Conversions

Figure 21-3 shows the operation of the A/D Converter after the GO/$\overline{\text{DONE}}$ bit has been set and the ACQT<2:0> bits are cleared. A conversion is started after the following instruction to allow entry into Sleep mode before the conversion begins. The internal A/D RC oscillator must be selected to perform a conversion in Sleep.

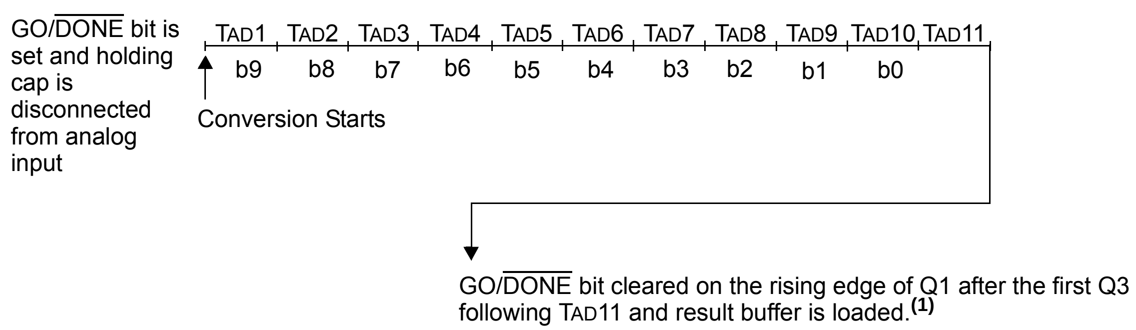
Figure 21-4 shows the operation of the A/D Converter after the GO/$\overline{\text{DONE}}$ bit has been set, the ACQT<3:0> bits are set to '010' and a 4 TAD acquisition time is selected before the conversion starts.

Clearing the GO/$\overline{\text{DONE}}$ bit during a conversion will abort the current conversion. The resulting buffer location will contain the partially completed A/D conversion sample. This will not set the ADIF flag, therefore, the user must read the buffer location before a conversion sequence overwrites it.

After the A/D conversion is completed or aborted, a 2 TAD wait is required before the next acquisition can be started. After this wait, acquisition on the selected channel is automatically started.

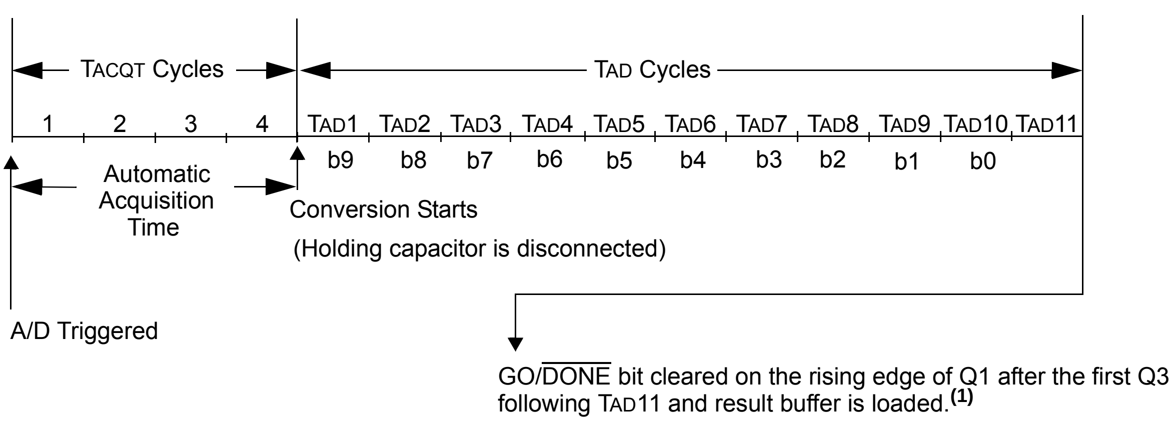
Note:	The GO/$\overline{\text{DONE}}$ bit should **NOT** be set in the same instruction that turns on the A/D.

FIGURE 21-3: A/D CONVERSION TAD CYCLES (ACQT<2:0> = 000, TACQ = 0)



Note 1: Conversion time is a minimum of 11 TAD + 2 TCY and a maximum of 11 TAD + 6 TCY.

FIGURE 21-4: A/D CONVERSION TAD CYCLES (ACQT<3:0> = 0010, TACQ = 4 TAD)

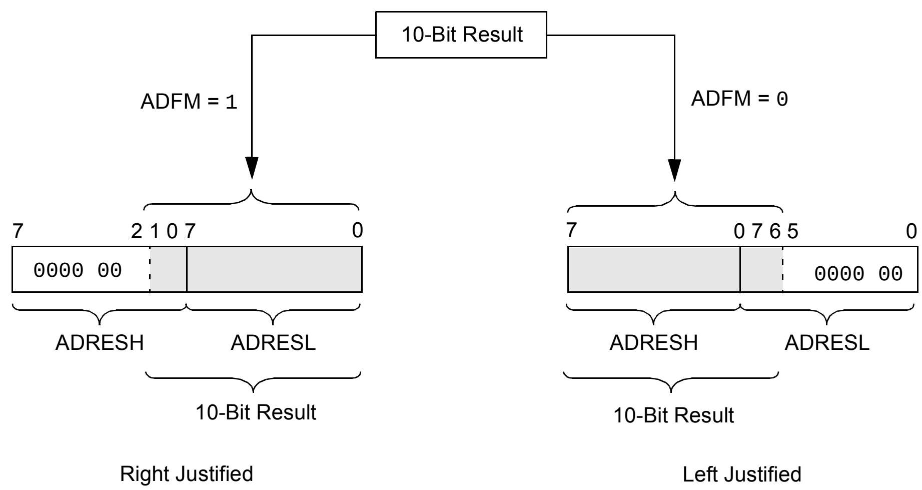


Note 1: In Continuous modes, next conversion starts at the end of TAD12.

21.9.1 A/D RESULT REGISTER

The ADRESH:ADRESL register pair is the location where the 10-bit A/D result is loaded at the completion of the A/D conversion. This register pair is 16 bits wide. The A/D module gives the flexibility to left or right justify the 10-bit result in the 16-bit result register. The A/D Format Select bit (ADFM) controls this justification. Figure 21-5 shows the operation of the A/D result justification. The extra bits are loaded with '0's. When an A/D result will not overwrite these locations (A/D disable), these registers may be used as two general purpose 8-bit registers.

FIGURE 21-5: A/D RESULT JUSTIFICATION



EQUATION 21-3: CONVERSION TIME FOR MULTI-CHANNEL MODES

Sequential Mode:

$$T = (T_{ACQ})_A + (T_{CON})_A + [(T_{ACQ})_B - 12\ T_{AD}] + (T_{CON})_B + [(T_{ACQ})_C - 12\ T_{AD}] + (T_{CON})_C + [(T_{ACQ})_D - 12\ T_{AD}] + (T_{CON})_D$$

Simultaneous Mode:

$$T = T_{ACQ} + (T_{CON})_A + (T_{CON})_B + T_{ACQ} + (T_{CON})_C + (T_{CON})_D$$

TABLE 21-3: SUMMARY OF A/D REGISTERS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on Page:
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	54
PIR1	—	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	57
PIE1	—	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	57
IPR1	—	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	57
PIR2	OSCFIF	—	—	EEIF	—	LVDIF	—	CCP2IF	57
PIE2	OSCFIE	—	—	EEIE	—	LVDIE	—	CCP2IE	57
IPR2	OSCFIP	—	—	EEIP	—	LVDIP	—	CCP2IP	57
ADRESH	A/D Result Register High Byte								56
ADRESL	A/D Result Register Low Byte								56
ADCON0	—	—	ACONV	ACSCH	ACMOD1	ACMOD0	GO/$\overline{\text{DONE}}$	ADON	56
ADCON1	VCFG1	VCFG0	—	FIFOEN	BFEMT	BFOVFL	ADPNT1	ADPNT0	56
ADCON2	ADFM	ACQT3	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	56
ADCON3	ADRS1	ADRS0	—	SSRC4	SSRC3	SSRC2	SSRC1	SSRC0	56
ADCHS	GDSEL1	GDSEL0	GBSEL1	GBSEL0	GCSEL1	GCSEL0	GASEL1	GASEL0	56
ANSEL0	ANS7(6)	ANS6(6)	ANS5(6)	ANS4	ANS3	ANS2	ANS1	ANS0	56
ANSEL1	—	—	—	—	—	—	—	ANS8(5)	56
PORTA	RA7(4)	RA6(4)	RA5	RA4	RA3	RA2	RA1	RA0	57
TRISA	TRISA7(4)	TRISA6(4)	PORTA Data Direction Register						57
PORTE(2)	—	—	—	—	RE3(1,3)	RA2(3)	RA1(3)	RA0(3)	57
TRISE(3)	—	—	—	—	—	PORTE Data Direction Register			57
LATE(3)	—	—	—	—	—	LATE Data Output Register			57

Legend: — = unimplemented, read as '0'. Shaded cells are not used for A/D conversion.

Note 1: The RE3 port bit is available only as an input pin when the MCLRE bit in the CONFIG3H register is '0'.

2: This register is not implemented on PIC18F2331/2431 devices.

3: These bits are not implemented on PIC18F2331/2431 devices.

4: These pins may be configured as port pins depending on the oscillator mode selected.

5: ANS5 through ANS8 are available only on the PIC18F4331/4431 devices.

6: Not available on 28-pin devices.

NOTES:

22.0 LOW-VOLTAGE DETECT (LVD)

PIC18F2331/2431/4331/4431 devices have a Low-Voltage Detect module (LVD), a programmable circuit that enables the user to specify a device voltage trip point. If the device experiences an excursion below the trip point, an interrupt flag is set. If the interrupt is enabled, the program execution will branch to the interrupt vector address and the software can then respond to the interrupt.

The Low-Voltage Detect Control register (Register 22-1) completely controls the operation of the LVD module. This allows the circuitry to be "turned off" by the user under software control, which minimizes the current consumption for the device.

The block diagram for the LVD module is shown in Figure 22-1.

The module is enabled by setting the LVDEN bit, but the circuitry requires some time to stabilize each time that it is enabled. The IRVST bit is a read-only bit used to indicate when the circuit is stable. The module can only generate an interrupt after the circuit is stable and the IRVST bit is set. The module monitors for drops in VDD below a predetermined set point.

REGISTER 22-1: LVDCON: LOW-VOLTAGE DETECT CONTROL REGISTER

U-0	U-0	R-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-1
—	—	IRVST	LVDEN	LVDL3[(1)]	LVDL2[(1)]	LVDL1[(1)]	LVDL0[(1)]
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

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

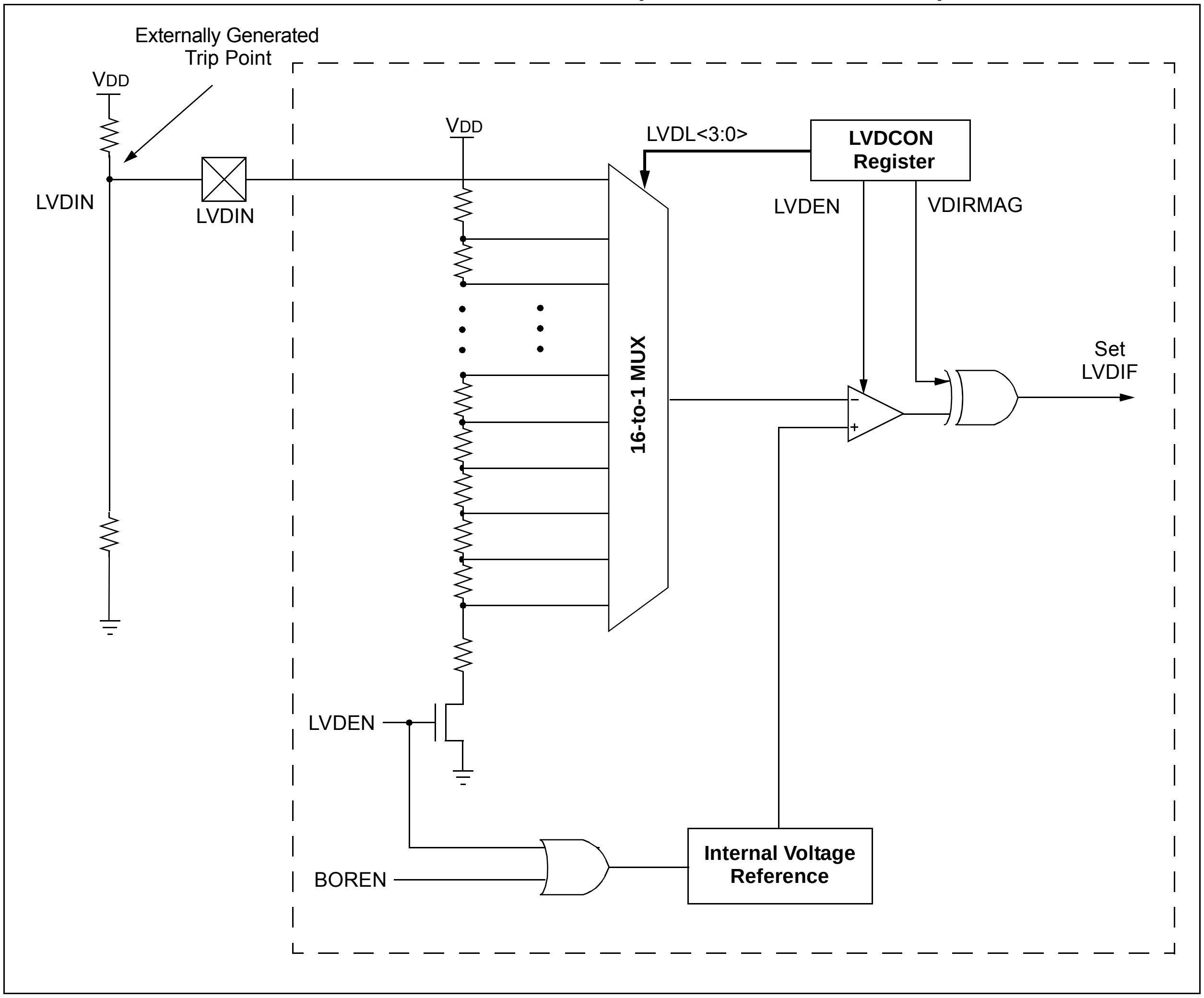
bit 5 **IRVST:** Internal Reference Voltage Stable Flag bit
1 = Indicates that the Low-Voltage Detect logic will generate the interrupt flag at the specified voltage range
0 = Indicates that the Low-Voltage Detect logic will not generate the interrupt flag at the specified voltage range and the LVD interrupt should not be enabled

bit 4 **LVDEN:** Low-Voltage Detect Power Enable bit
1 = Enables LVD, powers up LVD circuit
0 = Disables LVD, powers down LVD circuit

bit 3-0 **LVDL<3:0>:** Low-Voltage Detection Limit bits[(1)]
1111 = External analog input is used (input comes from the LVDIN pin)
1110 = Maximum setting
.
.
.
0010 = Minimum setting
0001 = Reserved
0000 = Reserved

Note 1: LVDL<3:0> bit modes, which result in a trip point below the valid operating voltage of the device, are not tested.

FIGURE 22-1: LVD MODULE BLOCK DIAGRAM (WITH EXTERNAL INPUT)



22.1 Operation

When the LVD module is enabled, a comparator uses an internally generated reference voltage as the set point. The set point is compared with the trip point, where each node in the resistor divider represents a trip point voltage. The "trip point" voltage is the voltage level at which the device detects a low-voltage event, depending on the configuration of the module. When the supply voltage is equal to the trip point, the voltage tapped off of the resistor array is equal to the internal reference voltage generated by the voltage reference module. The comparator then generates an interrupt signal by setting the LVDIF bit.

The trip point voltage is software programmable to any one of 16 values, selected by programming the LVDL<3:0> bits (LVDCON<3:0>).

The LVD module has an additional feature that allows the user to supply the trip voltage to the module from an external source. This mode is enabled when bits, LVDL<3:0>, are set to '1111'. In this state, the comparator input is multiplexed from the external input pin, LVDIN. This gives users flexibility because it allows them to configure the Low-Voltage Detect interrupt to occur at any voltage in the valid operating range.

22.2 LVD Setup

The following steps are needed to set up the LVD module:

1. Disable the module by clearing the LVDEN bit (LVDCON<4>).
2. Write the value to the LVDL<3:0> bits that selects the desired LVD trip point.
3. Enable the LVD module by setting the LVDEN bit.
4. Clear the LVD interrupt flag (PIR2<2>), which may have been set from a previous interrupt.
5. Enable the LVD interrupt, if interrupts are desired, by setting the LVDIE and GIE bits (PIE<2> and INTCON<7>).

An interrupt will not be generated until the IRVST bit is set.

22.3 Current Consumption

When the module is enabled, the LVD comparator and voltage divider are enabled and will consume static current. The total current consumption, when enabled, is specified in electrical specification Parameter D022B.

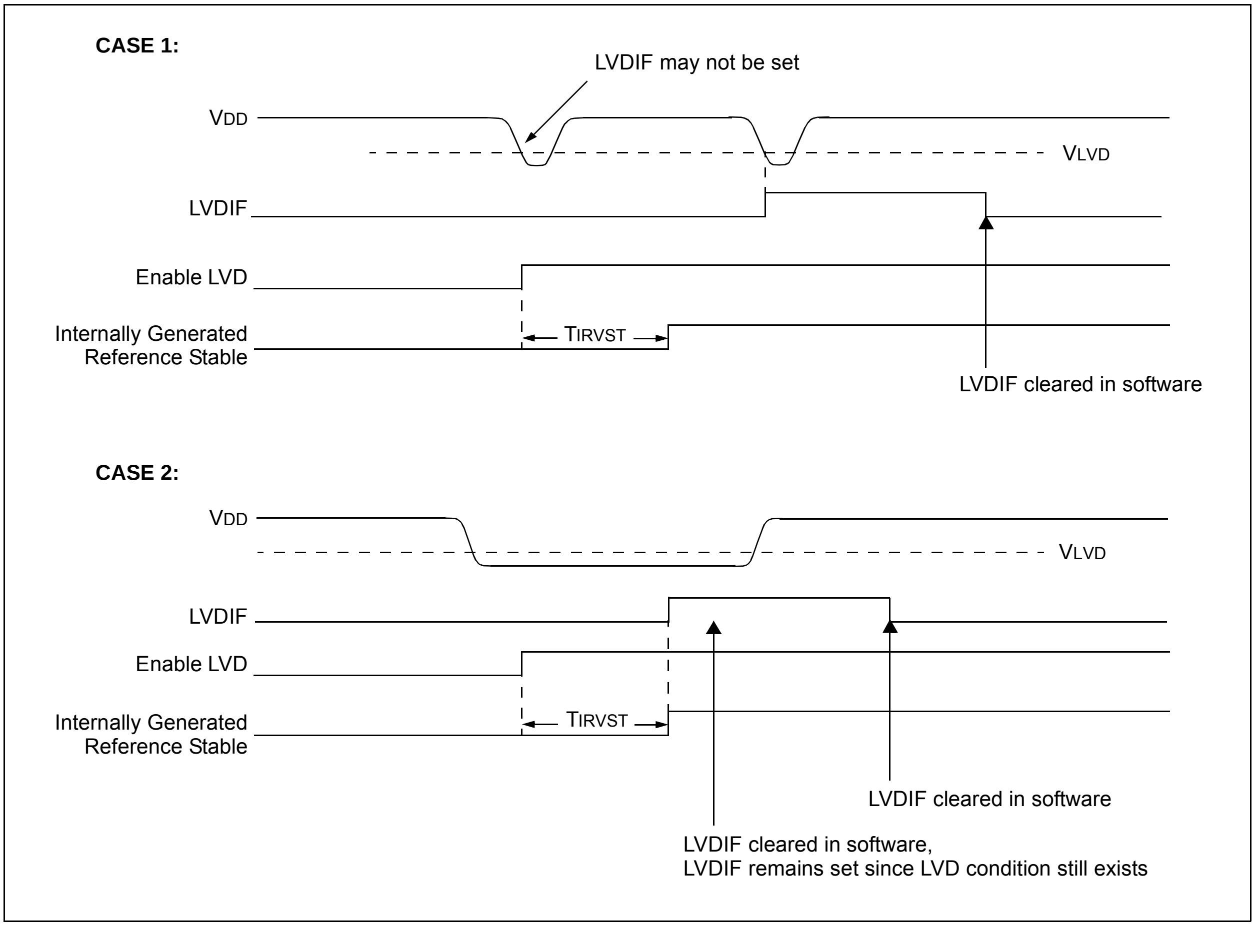
Depending on the application, the LVD module does not need to be operating constantly. To decrease the current requirements, the LVD circuitry may only need to be enabled for short periods where the voltage is checked. After doing the check, the LVD module may be disabled.

22.4 LVD Start-up Time

The internal reference voltage of the LVD module, specified in electrical specification Parameter D420, may be used by other internal circuitry, such as the Programmable Brown-out Reset. If the LVD, or other circuits using the voltage reference, are disabled to lower the device's current consumption, the reference voltage circuit will require time to become stable before a low-voltage condition can be reliably detected. This start-up time, TIRVST, is an interval that is independent of device clock speed. It is specified in electrical specification Parameter 36.

The LVD interrupt flag is not enabled until TIRVST has expired and a stable reference voltage is reached. For this reason, brief excursions beyond the set point may not be detected during this interval (refer to Figure 22-2).

FIGURE 22-2: LOW-VOLTAGE DETECT WAVEFORMS



22.5 Operation During Sleep

When enabled, the LVD circuitry continues to operate during Sleep. If the device voltage crosses the trip point, the LVDIF bit will be set and the device will wake-up from Sleep. Device execution will continue from the interrupt vector address if interrupts have been globally enabled.

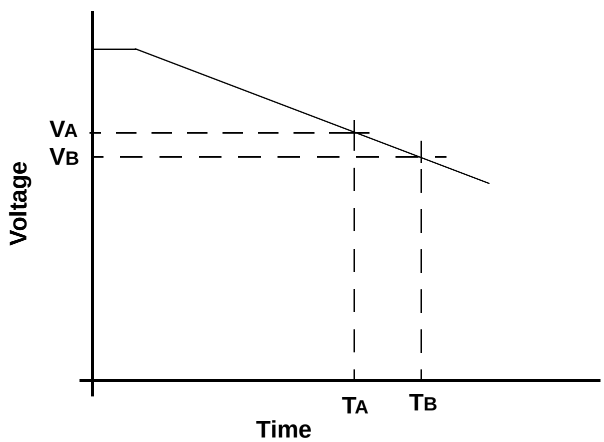
22.6 Effects of a Reset

A device Reset forces all registers to their Reset state. This forces the LVD module to be turned off.

22.7 Applications

Figure 22-3 shows a possible application voltage curve (typically for batteries). Over time, the device voltage decreases. When the device voltage equals voltage, V_A, the LVD logic generates an interrupt. This occurs at time, T_A. The application software then has the time, until the device voltage is no longer in valid operating range, to perform “housekeeping tasks” and to shut down the system. Voltage point, V_B, is the minimum valid operating voltage specification. This occurs at time, T_B. The difference, $T_B - T_A$, is the total time for shutdown.

FIGURE 22-3: TYPICAL LOW-VOLTAGE DETECT APPLICATION



Legend: V_A = LVD trip point
V_B = Minimum valid device operating voltage

TABLE 22-1: REGISTERS ASSOCIATED WITH LOW-VOLTAGE DETECT MODULE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
LVDCON	—	—	IRVST	LVDEN	LVDL3	LVDL2	LVDL1	LVDL0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
IPR2	OSCFIP	—	—	EEIP	—	LVDIP	—	CCP2IP
PIR2	OSCFIF	—	—	EEIF	—	LVDIF	—	CCP2IF
PIE2	OSCFIE	—	—	EEIE	—	LVDIE	—	CCP2IE

Legend: — = unimplemented, read as ‘0’. Shaded cells are unused by the LVD module.

NOTES:

23.0 SPECIAL FEATURES OF THE CPU

PIC18F2331/2431/4331/4431 devices include several features intended to maximize system reliability and minimize cost through elimination of external components. These are:

- Oscillator Selection
- Resets:
  - Power-on Reset (POR)
  - Power-up Timer (PWRT)
  - Oscillator Start-up Timer (OST)
  - Brown-out Reset (BOR)
- Interrupts
- Watchdog Timer (WDT)
- Fail-Safe Clock Monitor
- Two-Speed Start-up
- Code Protection
- ID Locations
- In-Circuit Serial Programming™ (ICSP™)

The oscillator can be configured for the application depending on frequency, power, accuracy and cost. All of the options are discussed in detail in **Section 3.0 "Oscillator Configurations"**.

A complete discussion of device Resets and interrupts is available in previous sections of this data sheet.

In addition to their Power-up and Oscillator Start-up Timers provided for Resets, PIC18F2331/2431/4331/4431 devices have a Watchdog Timer, which is either permanently enabled via the Configuration bits, or software-controlled (if configured as disabled).

The inclusion of an internal RC oscillator also provides the additional benefits of a Fail-Safe Clock Monitor (FSCM) and Two-Speed Start-up. FSCM provides for background monitoring of the peripheral clock and automatic switchover in the event of its failure. Two-Speed Start-up enables code to be executed almost immediately on start-up, while the primary clock source completes its start-up delays.

All of these features are enabled and configured by setting the appropriate Configuration register bits.

23.1 Configuration Bits

The Configuration bits can be programmed (read as '0'), or left unprogrammed (read as '1'), to select various device configurations. These bits are mapped starting at program memory location 300000h.

The user will note that address 300000h is beyond the user program memory space. In fact, it belongs to the configuration memory space (300000h-3FFFFFh), which can only be accessed using table reads and table writes.

Programming the Configuration registers is done in a manner similar to programming the Flash memory. The EECON1 register WR bit starts a self-timed write to the Configuration register. In normal operation mode, a TBLWT instruction with the TBLPTR pointing to the Configuration register sets up the address and the data for the Configuration register write. Setting the WR bit starts a long write to the Configuration register. The Configuration registers are written a byte at a time. To write or erase a configuration cell, a TBLWT instruction can write a '1' or a '0' into the cell. For additional details on Flash programming, refer to **Section 8.5 "Writing to Flash Program Memory"**.

TABLE 23-1: CONFIGURATION BITS AND DEVICE IDs

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default/ Unprogrammed Value
300000h	CONFIG1L	—	—	—	—	—	—	—	—	`---- ----`
300001h	CONFIG1H	IESO	FCMEN	—	—	FOSC3	FOSC2	FOSC1	FOSC0	`11-- 1111`
300002h	CONFIG2L	—	—	—	—	BORV1	BORV0	BOREN	$\overline{\text{PWRTEN}}$	`---- 1111`
300003h	CONFIG2H	—	—	$\overline{\text{WINEN}}$	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDTEN	`--11 1111`
300004h	CONFIG3L	—	—	$\overline{\text{T1OSCMX}}$	HPOL	LPOL	PWMPIN	—	—	`--11 11--`
300005h	CONFIG3H	MCLRE[(1)]	—	—	EXCLKMX[(1)]	PWM4MX[(1)]	SSPMX[(1)]	—	FLTAMX[(1)]	`1--1 11-1`
300006h	CONFIG4L	$\overline{\text{DEBUG}}$	—	—	—	—	LVP	—	STVREN	`1--- -1-1`
300007h	CONFIG4H	—	—	—	—	—	—	—	—	`---- ----`
300008h	CONFIG5L	—	—	—	—	CP3[(1)]	CP2[(1)]	CP1	CP0	`---- 1111`
300009h	CONFIG5H	CPD	CPB	—	—	—	—	—	—	`11-- ----`
30000Ah	CONFIG6L	—	—	—	—	WRT3[(1)]	WRT2[(1)]	WRT1	WRT0	`---- 1111`
30000Bh	CONFIG6H	WRTD	WRTB	WRTC	—	—	—	—	—	`111- ----`
30000Ch	CONFIG7L	—	—	—	—	EBTR3[(1)]	EBTR2[(1)]	EBTR1	EBTR0	`---- 1111`
30000Dh	CONFIG7H	—	EBTRB	—	—	—	—	—	—	`-1-- ----`
3FFFFEh	DEVID1[(2)]	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	`xxxx xxxx`[(2)]
3FFFFFh	DEVID2[(2)]	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	`0000 0101`

Legend: x = unknown, u = unchanged, - = unimplemented. Shaded cells are unimplemented, read as ‘0’.

Note 1: Unimplemented in PIC18F2331/4331 devices; maintain this bit set.

2: See Register 23-13 for DEVID1 values. DEVID registers are read-only and cannot be programmed by the user.

REGISTER 23-1: CONFIG1H: CONFIGURATION REGISTER 1 HIGH (BYTE ADDRESS 300001h)

R/P-1	R/P-1	U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1
IESO	FCMEN	—	—	FOSC3	FOSC2	FOSC1	FOSC0
bit 7							bit 0

Legend:
R = Readable bit | P = Programmable bit | U = Unimplemented bit, read as ‘0’
-n = Value when device is unprogrammed | U = Unchanged from programmed state

bit 7 **IESO:** Internal External Switchover bit
`1` = Internal External Switchover mode enabled
`0` = Internal External Switchover mode disabled

bit 6 **FCMEN:** Fail-Safe Clock Monitor Enable bit
`1` = Fail-Safe Clock Monitor enabled
`0` = Fail-Safe Clock Monitor disabled

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

bit 3-0 **FOSC<3:0>:** Oscillator Selection bits
`11xx` = External RC oscillator, CLKO function on RA6
`1001` = Internal oscillator block, CLKO function on RA6 and port function on RA7 (INTIO1)
`1000` = Internal oscillator block, port function on RA6 and port function on RA7 (INTIO2)
`0111` = External RC oscillator, port function on RA6
`0110` = HS oscillator, PLL enabled (clock frequency = 4 x FOSC1)
`0101` = EC oscillator, port function on RA6 (ECIO)
`0100` = EC oscillator, CLKO function on RA6 (EC)
`0010` = HS oscillator
`0001` = XT oscillator
`0000` = LP oscillator

REGISTER 23-2: CONFIG2L: CONFIGURATION REGISTER 2 LOW (BYTE ADDRESS 300002h)

U-0	U-0	U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1
—	—	—	—	BORV1	BORV0	BOREN[1]	$\overline{\text{PWRTEN}}$[1]
bit 7							bit 0

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as ‘0’
-n = Value when device is unprogrammed		U = Unchanged from programmed state

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

bit 3-2 **BORV<1:0>:** Brown-out Reset Voltage bits
11 = Reserved
10 = V_{BOR} set to 2.7V
01 = V_{BOR} set to 4.2V
00 = V_{BOR} set to 4.5V

bit 1 **BOREN:** Brown-out Reset Enable bit[1]
1 = Brown-out Reset is enabled
0 = Brown-out Reset is disabled

bit 0 **$\overline{\text{PWRTEN}}$:** Power-up Timer Enable bit[1]
1 = PWRT is disabled
0 = PWRT is enabled

Note 1: Having BOREN = 1 does not automatically override the $\overline{\text{PWRTEN}}$ to ‘0’, nor automatically enables the Power-up Timer.

REGISTER 23-3: CONFIG2H: CONFIGURATION REGISTER 2 HIGH (BYTE ADDRESS 300003h)

U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1
—	—	$\overline{\text{WINEN}}$	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDTEN
bit 7							bit 0

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
-n = Value when device is unprogrammed		U = Unchanged from programmed state

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

bit 5 **$\overline{\text{WINEN}}$:** Watchdog Timer Window Enable bit
1 = WDT window is disabled
0 = WDT window is enabled

bit 4-1 **WDTPS<3:0>:** Watchdog Timer Postscale Select bits
1111 = 1:32,768
1110 = 1:16,384
1101 = 1:8,192
1100 = 1:4,096
1011 = 1:2,048
1010 = 1:1,024
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

bit 0 **WDTEN:** Watchdog Timer Enable bit
1 = WDT is enabled
0 = WDT is disabled (control is placed on the SWDTEN bit)

REGISTER 23-4: CONFIG3L: CONFIGURATION REGISTER 3 LOW (BYTE ADDRESS 300004h)

U-0	U	R/P-1	R/P-1	R/P-1	R/P-1	U	U
—	—	$\overline{\text{T1OSCMX}}$	HPOL(1)	LPOL(1)	PWMPIN(3)	—	—
bit 7							bit 0

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
-n = Value when device is unprogrammed		U = Unchanged from programmed state

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

bit 5 **$\overline{\text{T1OSCMX}}$:** Timer1 Oscillator Mode bit
1 = Low-power Timer1 operation when microcontroller is in Sleep mode
0 = Standard (legacy) Timer1 oscillator operation

bit 4 **HPOL:** High Side Transistors Polarity bit (i.e., Odd PWM Output Polarity Control bit)(1)
1 = PWM1, 3, 5 and 7 are active-high (default)(2)
0 = PWM1, 3, 5 and 7 are active-low(2)

bit 3 **LPOL:** Low Side Transistors Polarity bit (i.e., Even PWM Output Polarity Control bit)(1)
1 = PWM0, 2, 4 and 6 are active-high (default)(2)
0 = PWM0, 2, 4 and 6 are active-low(2)

bit 2 **PWMPIN:** PWM Output Pins Reset State Control bit(3)
1 = PWM outputs are disabled upon Reset (default)
0 = PWM outputs drive active states upon Reset

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

Note 1: Polarity control bits, HPOL and LPOL, define PWM signal output active and inactive states; PWM states generated by the Fault inputs or PWM manual override.

2: PWM6 and PWM7 output channels are only available on PIC18F4331/4431 devices.

3: When PWMPIN = 0, PWMEN<2:0> = 101 if the device has eight PWM output pins (40 and 44-pin devices) and PWMEN<2:0> = 100 if the device has six PWM output pins (28-pin devices). PWM output polarity is defined by HPOL and LPOL.

REGISTER 23-5: CONFIG3H: CONFIGURATION REGISTER 3 HIGH (BYTE ADDRESS 300005h)

R/P-1	U	U	R/P-1	R/P-1	R/P-1	U	R/P-1
MCLRE[(1)]	—	—	EXCLKMX[(1)]	PWM4MX[(1)]	SSPMX[(1)]	—	FLTAMX[(1)]
bit 7							bit 0

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as ‘0’
-n = Value when device is unprogrammed		U = Unchanged from programmed state

bit 7 **MCLRE:** $\overline{\text{MCLR}}$ Pin Enable bit[(1)]
1 = $\overline{\text{MCLR}}$ pin is enabled; RE3 input pin is disabled
0 = RE3 input pin is enabled; $\overline{\text{MCLR}}$ is disabled

bit 6-5 **Unimplemented:** Read as ‘0’

bit 4 **EXCLKMX:** TMR0/T5CKI External Clock MUX bit[(1)]
1 = TMR0/T5CKI external clock input is multiplexed with RC3
0 = TMR0/T5CKI external clock input is multiplexed with RD0

bit 3 **PWM4MX:** PWM4 MUX bit[(1)]
1 = PWM4 output is multiplexed with RB5
0 = PWM4 output is multiplexed with RD5

bit 2 **SSPMX:** SSP I/O MUX bit[(1)]
1 = SCK/SCL clocks and SDA/SDI data are multiplexed with RC5 and RC4, respectively. SDO output is multiplexed with RC7.
0 = SCK/SCL clocks and SDA/SDI data are multiplexed with RD3 and RD2, respectively. SDO output is multiplexed with RD1.

bit 1 **Unimplemented:** Read as ‘0’

bit 0 **FLTAMX:** $\overline{\text{FLTA}}$ MUX bit[(1)]
1 = $\overline{\text{FLTA}}$ input is multiplexed with RC1
0 = $\overline{\text{FLTA}}$ input is multiplexed with RD4

Note 1: Unimplemented in PIC18F2331/2431 devices; maintain this bit set.

REGISTER 23-6: CONFIG4L: CONFIGURATION REGISTER 4 LOW (BYTE ADDRESS 300006h)

R/P-1	U-0	U-0	U-0	U-0	R/P-1	U-0	R/P-1
$\overline{DEBUG}$	—	—	—	—	LVP	—	STVREN
bit 7							bit 0

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
-n = Value when device is unprogrammed		U = Unchanged from programmed state

bit 7 **$\overline{DEBUG}$:** Background Debugger Enable bit
1 = Background debugger is disabled; RB6 and RB7 are configured as general purpose I/O pins
0 = Background debugger is enabled; RB6 and RB7 are dedicated to In-Circuit Debug

bit 6-3 **Unimplemented:** Read as '0'

bit 2 **LVP:** Single-Supply ICSP™ Enable bit
1 = Single-Supply ICSP is enabled
0 = Single-Supply ICSP is disabled

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

bit 0 **STVREN:** Stack Full/Underflow Reset Enable bit
1 = Stack full/underflow will cause Reset
0 = Stack full/underflow will not cause Reset

REGISTER 23-7: CONFIG5L: CONFIGURATION REGISTER 5 LOW (BYTE ADDRESS 300008h)

U-0	U-0	U-0	U-0	R/C-1	R/C-1	R/C-1	R/C-1
—	—	—	—	CP3[1,2]	CP2[1,2]	CP1[2]	CP0[2]
bit 7							bit 0

Legend:		
R = Readable bit	C = Clearable bit	U = Unimplemented bit, read as ‘0’
-n = Value when device is unprogrammed		U = Unchanged from programmed state

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

bit 3 **CP3:** Code Protection bit[1,2]
1 = Block 3 is not code-protected
0 = Block 3 is code-protected

bit 2 **CP2:** Code Protection bit[1,2]
1 = Block 2 is not code-protected
0 = Block 2 is code-protected

bit 1 **CP1:** Code Protection bit[2]
1 = Block 1 is not code-protected
0 = Block 1 is code-protected

bit 0 **CP0:** Code Protection bit[2]
1 = Block 0 is not code-protected
0 = Block 0 is code-protected

Note 1: Unimplemented in PIC18F2331/4331 devices; maintain this bit set.
2: Refer to Figure 23-5 for block boundary addresses.

REGISTER 23-8: CONFIG5H: CONFIGURATION REGISTER 5 HIGH (BYTE ADDRESS 300009h)

R/C-1	R/C-1	U-0	U-0	U-0	U-0	U-0	U-0
CPD[1]	CPB[1]	—	—	—	—	—	—
bit 7							bit 0

Legend:		
R = Readable bit	C = Clearable bit	U = Unimplemented bit, read as ‘0’
-n = Value when device is unprogrammed		U = Unchanged from programmed state

bit 7 **CPD:** Data EEPROM Code Protection bit[1]
1 = Data EEPROM is not code-protected
0 = Data EEPROM is code-protected

bit 6 **CPB:** Boot Block Code Protection bit[1]
1 = Boot Block is not code-protected
0 = Boot Block is code-protected

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

Note 1: Refer to Figure 23-5 for block boundary addresses.

REGISTER 23-9: CONFIG6L: CONFIGURATION REGISTER 6 LOW (BYTE ADDRESS 30000Ah)

U-0	U-0	U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1
—	—	—	—	WRT3[(1,2)]	WRT2[(1,2)]	WRT1[(2)]	WRT0[(2)]
bit 7							bit 0

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
-n = Value when device is unprogrammed		U = Unchanged from programmed state

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

bit 3 **WRT3:** Write Protection bit[(1,2)]
1 = Block 3 is not write-protected
0 = Block 3 is write-protected

bit 2 **WRT2:** Write Protection bit[(1,2)]
1 = Block 2 is not write-protected
0 = Block 2 is write-protected

bit 1 **WRT1:** Write Protection bit[(2)]
1 = Block 1 is not write-protected
0 = Block 1 is write-protected

bit 0 **WRT0:** Write Protection bit[(2)]
1 = Block 0 is not write-protected
0 = Block 0 is write-protected

Note 1: Unimplemented in PIC18F2331/4331 devices; maintain this bit set.
2: Refer to Figure 23-5 for block boundary addresses.

REGISTER 23-10: CONFIG6H: CONFIGURATION REGISTER 6 HIGH (BYTE ADDRESS 30000Bh)

R/P-1	R/P-1	R-1	U-0	U-0	U-0	U-0	U-0
WRTD[(2)]	WRTB[(2)]	WRTC[(1,2)]	—	—	—	—	—
bit 7							bit 0

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
-n = Value when device is unprogrammed		U = Unchanged from programmed state

bit 7 **WRTD:** Data EEPROM Write Protection bit[(2)]
1 = Data EEPROM is not write-protected
0 = Data EEPROM is write-protected

bit 6 **WRTB:** Boot Block Write Protection bit[(2)]
1 = Boot block is not write-protected
0 = Boot block is write-protected

bit 5 **WRTC:** Configuration Register Write Protection bit[(1,2)]
1 = Configuration registers are not write-protected
0 = Configuration registers are write-protected

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

Note 1: This bit is read-only in normal execution mode; it can be written only in Program mode.
2: Refer to Figure 23-5 for block boundary addresses.

REGISTER 23-11: CONFIG7L: CONFIGURATION REGISTER 7 LOW (BYTE ADDRESS 30000Ch)

U-0	U-0	U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1
—	—	—	—	EBTR3(1,2,3)	EBTR2(1,2,3)	EBTR1(2,3)	EBTR0(2,3)
bit 7							bit 0

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as ‘0’
-n = Value when device is unprogrammed		U = Unchanged from programmed state

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

bit 3 **EBTR3:** Table Read Protection bit(1,2,3)
1 = Block 3 is not protected from table reads executed in other blocks
0 = Block 3 is protected from table reads executed in other blocks

bit 2 **EBTR2:** Table Read Protection bit(1,2,3)
1 = Block 2 is not protected from table reads executed in other blocks
0 = Block 2 is protected from table reads executed in other blocks

bit 1 **EBTR1:** Table Read Protection bit(2,3)
1 = Block 1 is not protected from table reads executed in other blocks
0 = Block 1 is protected from table reads executed in other blocks

bit 0 **EBTR0:** Table Read Protection bit(2,3)
1 = Block 0 is not protected from table reads executed in other blocks
0 = Block 0 is protected from table reads executed in other blocks

Note 1: Unimplemented in PIC18F2331/4331 devices; maintain this bit set.
2: Refer to Figure 23-5 for block boundary addresses.
3: Enabling the corresponding CPx bit is recommended to protect the block from external read operations.

REGISTER 23-12: CONFIG7H: CONFIGURATION REGISTER 7 HIGH (BYTE ADDRESS 30000Dh)

U-0	R/P-1	U-0	U-0	U-0	U-0	U-0	U-0
—	EBTRB(1,2)	—	—	—	—	—	—
bit 7							bit 0

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as ‘0’
-n = Value when device is unprogrammed		U = Unchanged from programmed state

bit 7 **Unimplemented:** Read as ‘0’

bit 6 **EBTRB:** Boot Block Table Read Protection bit(1,2)
1 = Boot block is not protected from table reads executed in other blocks
0 = Boot block is protected from table reads executed in other blocks

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

Note 1: Enabling the corresponding CPx bit is recommended to protect the block from external read operations.
2: Refer to Figure 23-5 for block boundary addresses.

REGISTER 23-13: DEVID1: DEVICE ID REGISTER 1 FOR PIC18F2331/2431/4331/4431 DEVICES

R	R	R	R	R	R	R	R
DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0
bit 7							bit 0

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as ‘0’
-n = Value when device is unprogrammed		U = Unchanged from programmed state

bit 7-5 **DEV<2:0>:** Device ID bits
These bits are used with the DEV<10:3> bits in the Device ID Register 2 to identify the part number.
`000` = PIC18F4331
`001` = PIC18F4431
`100` = PIC18F2331
`101` = PIC18F2431

bit 4-0 **REV<4:0>:** Revision ID bits
These bits are used to indicate the device revision.

REGISTER 23-14: DEVID2: DEVICE ID REGISTER 2 FOR PIC18F2331/2431/4331/4431 DEVICES

R	R	R	R	R	R	R	R
DEV10(1)	DEV9(1)	DEV8(1)	DEV7(1)	DEV6(1)	DEV5(1)	DEV4(1)	DEV3(1)
bit 7							bit 0

Legend:		
R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as ‘0’
-n = Value when device is unprogrammed		U = Unchanged from programmed state

bit 7-0 **DEV<10:3>:** Device ID bits(1)
These bits are used with the DEV<2:0> bits in the Device ID Register 1 to identify the part number
`0000 0101` = PIC18F2331/2431/4331/4431 devices

Note 1: These values for DEV<10:3> may be shared with other devices. The specific device is always identified by using the entire DEV<10:0> bit sequence.

23.2 Watchdog Timer (WDT)

For PIC18F2331/2431/4331/4431 devices, the WDT is driven by the INTRC source. When the WDT is enabled, the clock source is also enabled. The nominal WDT period is 4 ms and has the same stability as the INTRC oscillator.

The 4 ms period of the WDT is multiplied by a 16-bit postscaler. Any output of the WDT postscaler is selected by a multiplexer, controlled by bits in Configuration Register 2H (see Register 23-3). Available periods range from 4 ms to 131.072 seconds (2.18 minutes). The WDT and postscaler are cleared when any of the following events occur: execute a `SLEEP` or `CLRWDT` instruction, the IRCF bits (OSCCON<6:4>) are changed or a clock failure has occurred (see **Section 23.4.1 "FSCM and the Watchdog Timer"**).

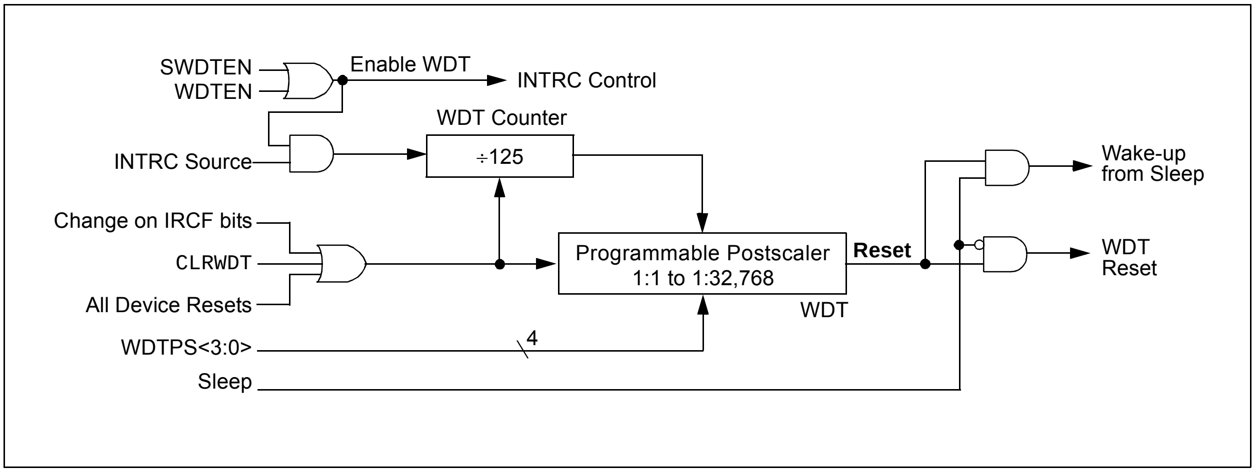
Adjustments to the internal oscillator clock period using the OSCTUNE register also affect the period of the WDT by the same factor. For example, if the INTRC period is increased by 3%, then the WDT period is increased by 3%.

> **Note 1:** The `CLRWDT` and `SLEEP` instructions clear the WDT and postscaler counts when executed.
>
> **2:** Changing the setting of the IRCF bits (OSCCON<6:4>) clears the WDT and postscaler counts.
>
> **3:** When a `CLRWDT` instruction is executed, the postscaler count will be cleared.
>
> **4:** If $\overline{\text{WINEN}}$ = `0`, then `CLRWDT` must be executed only when WDTW = `1`; otherwise, a device Reset will result.

23.2.1 CONTROL REGISTER

Register 23-15 shows the WDTCON register. This is a readable and writable register. The SWDTEN bit allows software to enable or disable the WDT, but only if the Configuration bit has disabled the WDT. The WDTW bit is a read-only bit that indicates when the WDT count is in the fourth quadrant (i.e., when the 8-bit WDT value is b'`11000000`' or greater).

FIGURE 23-1: WDT BLOCK DIAGRAM



REGISTER 23-15: WDTCON: WATCHDOG TIMER CONTROL REGISTER

R-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
WDTW	—	—	—	—	—	—	SWDTEN[(1)]
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **WDTW:** Watchdog Timer Window bit
1 = WDT count is in fourth quadrant
0 = WDT count is not in fourth quadrant

bit 6-1 **Unimplemented**: Read as '0'

bit 0 **SWDTEN:** Software Enable/Disable for Watchdog Timer bit[(1)]
1 = WDT is turned on
0 = WDT is turned off

Note 1: If the WDTEN Configuration bit = 1, then WDT is always enabled, irrespective of this control bit. If WDTEN Configuration bit = 0, then it is possible to turn WDT on/off with this control bit.

TABLE 23-2: SUMMARY OF WATCHDOG TIMER REGISTERS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CONFIG2H	—	—	$\overline{\text{WINEN}}$	WDTPS3	WDTPS2	WDTPS2	WDTPS0	WDTEN
RCON	IPEN	—	—	$\overline{\text{RI}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$
WDTCON	WDTW	—	—	—	—	—	—	SWDTEN

Legend: Shaded cells are not used by the Watchdog Timer.

23.3 Two-Speed Start-up

The Two-Speed Start-up feature helps to minimize the latency period from oscillator start-up to code execution by allowing the microcontroller to use the INTRC oscillator as a clock source until the primary clock source is available. It is enabled by setting the IESO bit in Configuration Register 1H (CONFIG1H<7>).

Two-Speed Start-up is available only if the primary oscillator mode is LP, XT, HS or HSPLL (Crystal-Based modes). Other sources do not require a OST start-up delay; for these, Two-Speed Start-up is disabled.

When enabled, Resets and wake-ups from Sleep mode cause the device to configure itself to run from the internal oscillator block as the clock source, following the time-out of the Power-up Timer after a Power-on Reset is enabled. This allows almost immediate code execution while the primary oscillator starts and the OST is running. Once the OST times out, the device automatically switches to PRI_RUN mode.

Because the OSCCON register is cleared on Reset events, the INTOSC (or postscaler) clock source is not initially available after a Reset event; the INTRC clock is used directly at its base frequency. To use a higher clock speed on wake-up, the INTOSC or postscaler clock sources can be selected to provide a higher clock speed by setting bits IRCF<2:0> immediately after Reset. For wake-ups from Sleep, the INTOSC or postscaler clock sources can be selected by setting IRCF<2:0> prior to entering Sleep mode.

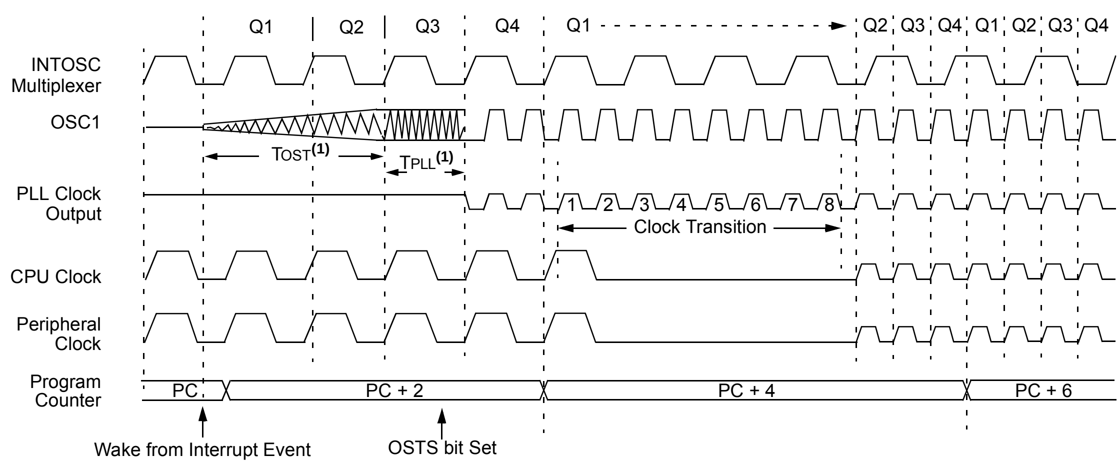
In all other power-managed modes, Two-Speed Start-up is not used. The device will be clocked by the currently selected clock source until the primary clock source becomes available. The setting of the IESO Configuration bit is ignored.

23.3.1 SPECIAL CONSIDERATIONS FOR USING TWO-SPEED START-UP

While using the INTRC oscillator in Two-Speed Start-up, the device still obeys the normal command sequences for entering power-managed modes, including serial SLEEP instructions (refer to **Section 4.1.4 "Multiple Sleep Commands"**). In practice, this means that user code can change the SCS<1:0> bit settings and issue SLEEP commands before the OST times out. This would allow an application to briefly wake-up, perform routine "housekeeping" tasks and return to Sleep before the device starts to operate from the primary oscillator.

User code can also check if the primary clock source is currently providing the system clocking by checking the status of the OSTS bit (OSCCON<3>). If the bit is set, the primary oscillator is providing the system clock. Otherwise, the internal oscillator block is providing the clock during wake-up from Reset or Sleep mode.

FIGURE 23-2: TIMING TRANSITION FOR TWO-SPEED START-UP (INTOSC TO HSPLL)



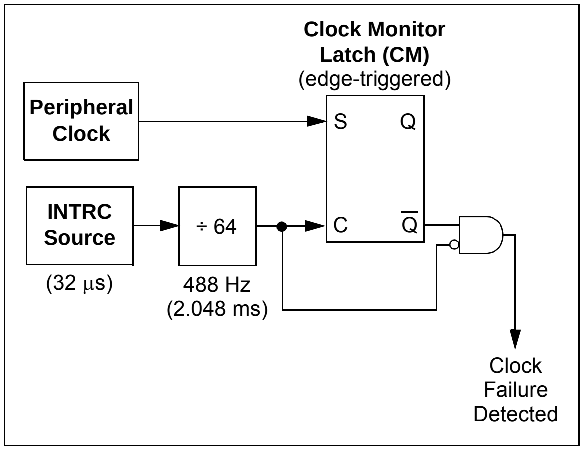
Note 1: TOST = 1024 TOSC; TPLL = 2 ms (approx). These intervals are not shown to scale.

23.4 Fail-Safe Clock Monitor

The Fail-Safe Clock Monitor (FSCM) allows the microcontroller to continue operation, in the event of an external oscillator failure, by automatically switching the system clock to the internal oscillator block. The FSCM function is enabled by setting the Fail-Safe Clock Monitor Enable bit, FCMEN (CONFIG1H<6>).

When FSCM is enabled, the INTRC oscillator runs at all times to monitor clocks to peripherals and provide an instant backup clock in the event of a clock failure. Clock monitoring (shown in Figure 23-3) is accomplished by creating a sample clock signal, which is the INTRC output divided by 64. This allows ample time between FSCM sample clocks for a peripheral clock edge to occur. The peripheral system clock and the sample clock are presented as inputs to the Clock Monitor latch (CM). The CM is set on the falling edge of the system clock source, but cleared on the rising edge of the sample clock.

FIGURE 23-3: FSCM BLOCK DIAGRAM



Clock failure is tested for on the falling edge of the sample clock. If a sample clock falling edge occurs while the CM is still set, a clock failure has been detected (Figure 23-4). This causes the following:

- the FSCM generates an oscillator fail interrupt by setting bit, OSCFIF (PIR2<7>);
- the system clock source is switched to the internal oscillator block (OSCCON is not updated to show the current clock source – this is the fail-safe condition); and
- the WDT is reset.

Since the postscaler frequency from the internal oscillator block may not be sufficiently stable, it may be desirable to select another clock configuration and enter an alternate power-managed mode (see **Section 23.3.1 "Special Considerations for Using Two-Speed Start-up"** and **Section 4.1.4 "Multiple Sleep Commands"** for more details). This can be done to attempt a partial recovery or execute a controlled shutdown.

To use a higher clock speed on wake-up, the INTOSC or postscaler clock sources can be selected to provide a higher clock speed by setting bits, IRCF<2:0>, immediately after Reset. For wake-ups from Sleep, the INTOSC or postscaler clock sources can be selected by setting the IRCF<2:0> bits prior to entering Sleep mode.

Adjustments to the internal oscillator block using the OSCTUNE register also affect the period of the FSCM by the same factor. This can usually be neglected, as the clock frequency being monitored is generally much higher than the sample clock frequency.

The FSCM will detect failures of the primary or secondary clock sources only. If the internal oscillator block fails, no failure would be detected, nor would any action be possible.

23.4.1 FSCM AND THE WATCHDOG TIMER

Both the FSCM and the WDT are clocked by the INTRC oscillator. Since the WDT operates with a separate divider and counter, disabling the WDT has no effect on the operation of the INTRC oscillator when the FSCM is enabled.

As already noted, the clock source is switched to the INTOSC clock when a clock failure is detected. Depending on the frequency selected by the IRCF<2:0> bits, this may mean a substantial change in the speed of code execution. If the WDT is enabled with a small prescale value, a decrease in clock speed allows a WDT time-out to occur and a subsequent device Reset. For this reason, Fail-Safe Clock Monitor events also reset the WDT and postscaler, allowing it to start timing from when execution speed was changed and decreasing the likelihood of an erroneous time-out.

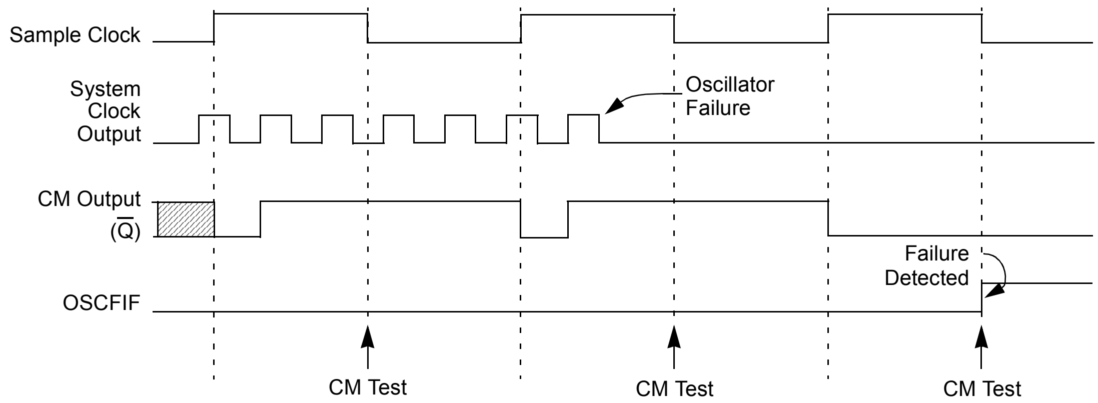
23.4.2 EXITING FAIL-SAFE OPERATION

The fail-safe condition is terminated by either a device Reset, or by entering a power-managed mode. On Reset, the controller starts the primary clock source specified in Configuration Register 1H (with any required start-up delays that are required for the oscillator mode, such as the OST or PLL timer). The INTOSC multiplexer provides the system clock until the primary clock source becomes ready (similar to a Two-Speed Start-up). The clock system source is then switched to the primary clock (indicated by the OSTS bit in the OSCCON register becoming set). The Fail-Safe Clock Monitor then resumes monitoring the peripheral clock.

The primary clock source may never become ready during start-up. In this case, operation is clocked by the INTOSC multiplexer. The OSCCON register will remain in its Reset state until a power-managed mode is entered.

Entering a power-managed mode by loading the OSCCON register and executing a SLEEP instruction will clear the fail-safe condition. When the fail-safe condition is cleared, the clock monitor will resume monitoring the peripheral clock.

FIGURE 23-4: FSCM TIMING DIAGRAM



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

23.4.3 FSCM INTERRUPTS IN POWER-MANAGED MODES

As previously mentioned, entering a power-managed mode clears the fail-safe condition. By entering a power-managed mode, the clock multiplexer selects the clock source selected by the OSCCON register. Fail-safe monitoring of the power-managed clock source resumes in the power-managed mode.

If an oscillator failure occurs during power-managed operation, the subsequent events depend on whether or not the oscillator failure interrupt is enabled. If enabled (OSCFIF = 1), code execution will be clocked by the INTOSC multiplexer. An automatic transition back to the failed clock source will not occur.

If the interrupt is disabled, the device will not exit the power-managed mode on oscillator failure. Instead, the device will continue to operate as before, but clocked by the INTOSC multiplexer. While in Idle mode, subsequent interrupts will cause the CPU to begin executing instructions while being clocked by the INTOSC multiplexer. The device will not transition to a different clock source until the fail-safe condition is cleared.

23.4.4 POR OR WAKE FROM SLEEP

The FSCM is designed to detect oscillator failure at any point after the device has exited Power-on Reset (POR) or low-power Sleep mode. When the primary system clock is EC, RC or INTRC modes, monitoring can begin immediately following these events.

For oscillator modes involving a crystal or resonator (HS, HSPLL, LP or XT), the situation is somewhat different. Since the oscillator may require a start-up time considerably longer than the FCSM sample clock time, a false clock failure may be detected. To prevent this, the internal oscillator block is automatically configured as the system clock and functions until the primary clock is stable (the OST and PLL timers have timed out). This is identical to Two-Speed Start-up mode. Once the primary clock is stable, the INTRC returns to its role as the FSCM source.

Note:	The same logic that prevents false oscillator failure interrupts on POR or wake from Sleep will also prevent the detection of the oscillator's failure to start at all following these events. This can be avoided by monitoring the OSTS bit and using a timing routine to determine if the oscillator is taking too long to start. Even so, no oscillator failure interrupt will be flagged.

As noted in **Section 23.3.1 "Special Considerations for Using Two-Speed Start-up"**, it is also possible to select another clock configuration, and enter an alternate power-managed mode, while waiting for the primary system clock to become stable. When the new powered-managed mode is selected, the primary clock is disabled.

23.5 Program Verification and Code Protection

The overall structure of the code protection on the PIC18 Flash devices differs significantly from other PIC® devices.

The user program memory is divided into five blocks. One of these is a Boot Block of 512 bytes. The remainder of the memory is divided into four blocks on binary boundaries.

Each of the five blocks has three code protection bits associated with them. They are:

- Code-Protect bit (CPn)
- Write-Protect bit (WRTn)
- External Block Table Read bit (EBTRn)

Figure 23-5 shows the program memory organization for 8 and 16-Kbyte devices, and the specific code protection bit associated with each block. The actual locations of the bits are summarized in Table 23-3.

FIGURE 23-5: CODE-PROTECTED PROGRAM MEMORY FOR PIC18F2331/2431/4331/4431

MEMORY SIZE/DEVICE

8 Kbytes (PIC18F2331/4331)	Address Range	16 Kbytes (PIC18F2431/4431)	Address Range	Block Code Protection Controlled By:
Boot Block	0000h 0FFFh	Boot Block	0000h 01FFh	CPB, WRTB, EBTRB
Block 0	0200h 0FFFh	Block 0	0200h 0FFFh	CP0, WRT0, EBTR0
Block 1	1000h 1FFFh	Block 1	1000h 1FFFh	CP1, WRT1, EBTR1
Unimplemented Read '0's		Block 2	2000h 2FFFh	CP2, WRT2, EBTR2
	3FFFh	Block 3	3000h 3FFFh	CP3, WRT3, EBTR3

TABLE 23-3: SUMMARY OF CODE PROTECTION REGISTERS

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
300008h	CONFIG5L	—	—	—	—	CP3(1)	CP2(1)	CP1	CP0
300009h	CONFIG5H	CPD	CPB	—	—	—	—	—	—
30000Ah	CONFIG6L	—	—	—	—	WRT3(1)	WRT2(1)	WRT1	WRT0
30000Bh	CONFIG6H	WRTD	WRTB	WRTC	—	—	—	—	—
30000Ch	CONFIG7L	—	—	—	—	EBTR3(1)	EBTR2(1)	EBTR1	EBTR0
30000Dh	CONFIG7H	—	EBTRB	—	—	—	—	—	—

Legend: Shaded cells are unimplemented.

Note 1: Unimplemented in PIC18F2331/4331 devices; maintain this bit set.

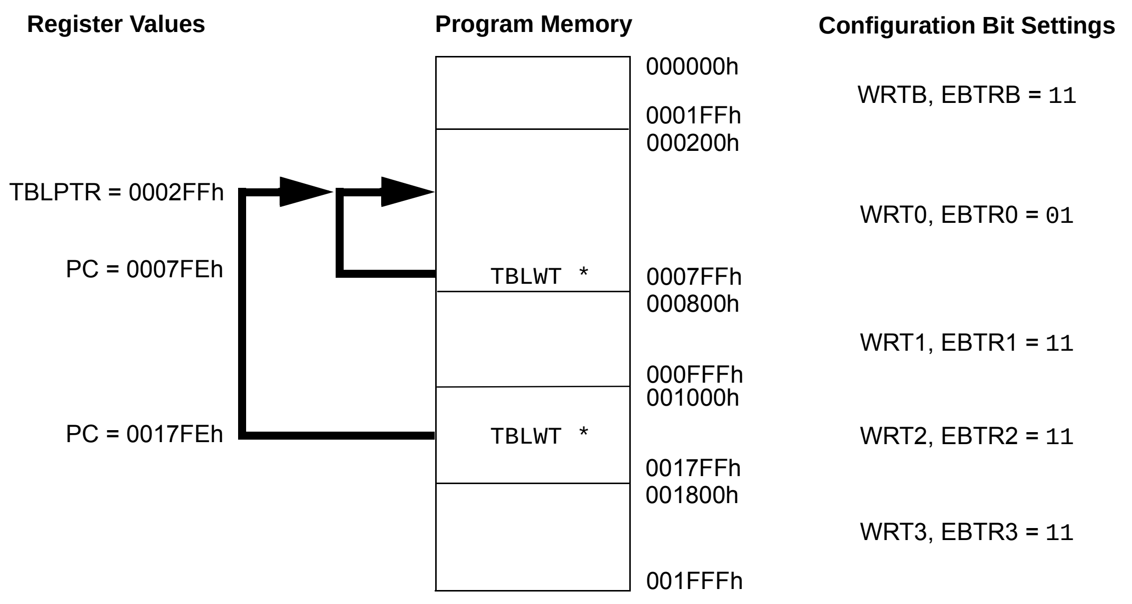
23.5.1 PROGRAM MEMORY CODE PROTECTION

The program memory may be read to, or written from, any location using the table read and table write instructions. The Device ID may be read with table reads. The Configuration registers may be read and written with the table read and table write instructions.

In normal execution mode, the CPn bits have no direct effect. CPn bits inhibit external reads and writes. A block of user memory may be protected from table writes if the WRTn Configuration bit is '0'. The EBTRn bits control table reads. For a block of user memory with the EBTRn bit set to '0', a table read instruction that executes from within that block is allowed to read. A table read instruction that executes from a location outside of that block is not allowed to read, and will result in reading '0's. Figures 23-6 through 23-8 illustrate table write and table read protection.

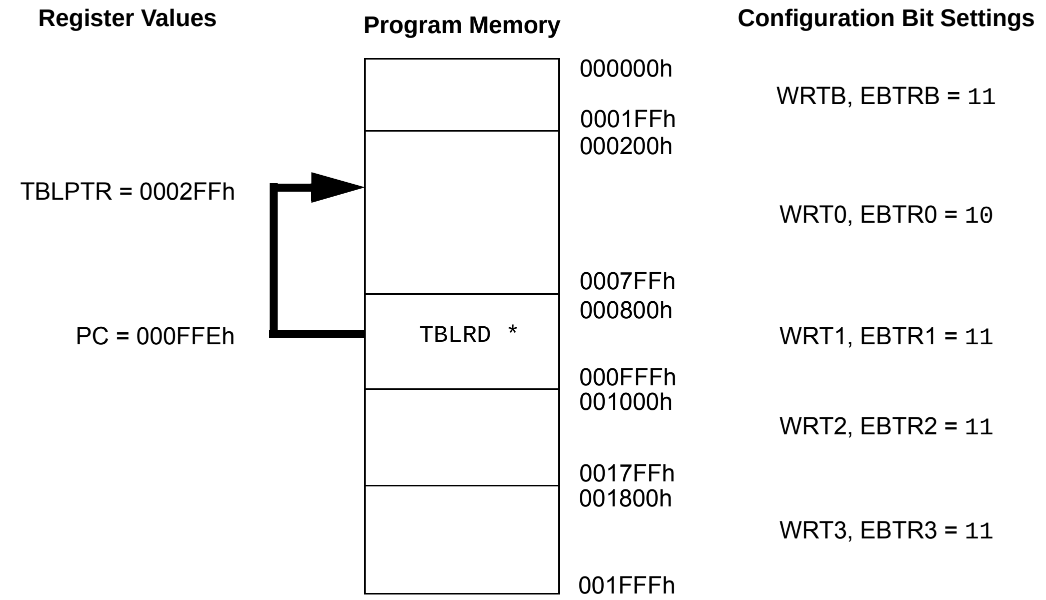
Note:	Code protection bits may only be written to a '0' from a '1' state. It is not possible to write a '1' to a bit in the '0' state. Code protection bits are only set to '1' by a full chip erase or block erase function. The full chip erase and block erase functions can only be initiated via ICSP or an external programmer.

FIGURE 23-6: TABLE WRITE (WRTn) DISALLOWED



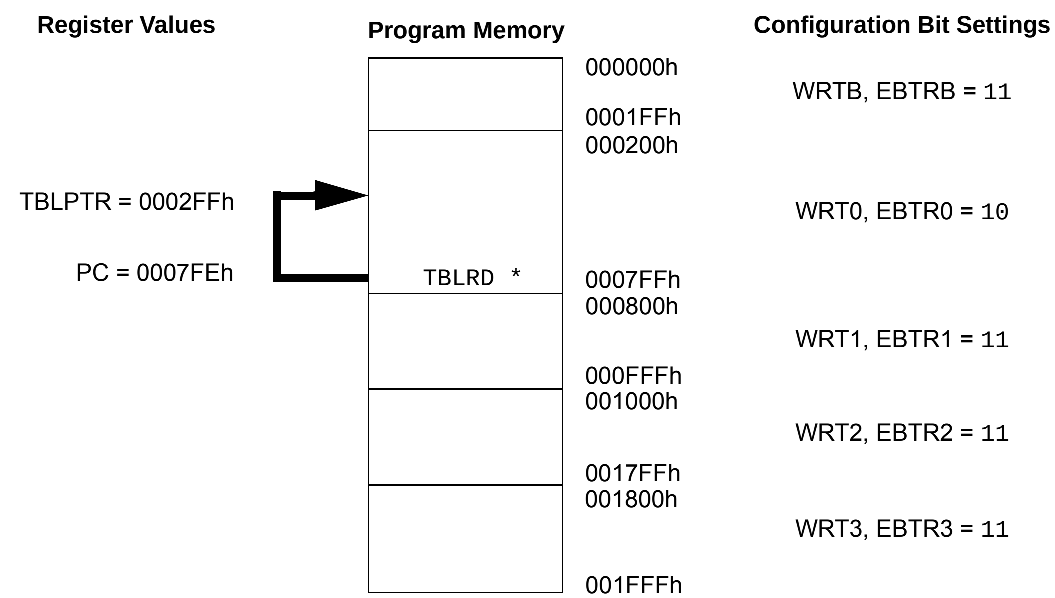
Results: All table writes are disabled to Blockn whenever WRTn = 0.

FIGURE 23-7: EXTERNAL BLOCK TABLE READ (EBTRn) DISALLOWED



Results: All table reads from external blocks to Blockn are disabled whenever EBTRn = 0.
The TABLAT register returns a value of '0'.

FIGURE 23-8: EXTERNAL BLOCK TABLE READ (EBTRn) ALLOWED



Results: Table reads permitted within Blockn, even when EBTRBn = 0.
The TABLAT register returns the value of the data at the location TBLPTR.

23.5.2 DATA EEPROM CODE PROTECTION

The entire data EEPROM is protected from external reads and writes by two bits: CPD and WRTD. CPD inhibits external reads and writes of data EEPROM. WRTD inhibits external writes to data EEPROM. The CPU can continue to read and write data EEPROM regardless of the protection bit settings.

23.5.3 CONFIGURATION REGISTER PROTECTION

The Configuration registers can be write-protected. The WRTC bit controls protection of the Configuration registers. In normal execution mode, the WRTC bit is readable only. WRTC can only be written via ICSP or an external programmer.

23.6 ID Locations

Eight memory locations (200000h-200007h) are designated as ID locations, where the user can store checksum or other code identification numbers. These locations are both readable and writable during normal execution through the TBLRD and TBLWT instructions, or during program/verify. The ID locations can be read when the device is code-protected.

23.7 In-Circuit Serial Programming

PIC18F2331/2431/4331/4431 microcontrollers can be serially programmed while in the end application circuit. This is simply done with two lines for clock and data, and three other lines for power, ground and the programming voltage. This allows customers to manufacture boards with unprogrammed devices, and then program the microcontroller just before shipping the product. This also allows the most recent firmware or a custom firmware to be programmed.

23.8 In-Circuit Debugger

When the $\overline{\text{DEBUG}}$ bit in the CONFIG4L Configuration register is programmed to a '0', the In-Circuit Debugger functionality is enabled. This function allows simple debugging functions when used with MPLAB® IDE. When the microcontroller has this feature enabled, some resources are not available for general use. Table 23-4 shows which resources are required by the background debugger.

TABLE 23-4: DEBUGGER RESOURCES

I/O pins:	RB6, RB7
Stack:	2 levels
Program Memory:	<1 Kbytes
Data Memory:	16 bytes

To use the In-Circuit Debugger function of the microcontroller, the design must implement In-Circuit Serial Programming connections to $\overline{\text{MCLR}}$/VPP, VDD, VSS, RB7 and RB6. This will interface to the In-Circuit Debugger module available from Microchip or one of the third party development tool companies.

23.9 Single-Supply ICSP™ Programming

The LVP bit in Configuration Register 4L (CONFIG4L<2>) enables Single-Supply ICSP Programming. When LVP is enabled, the microcontroller can be programmed without requiring high voltage being applied to the $\overline{\text{MCLR}}$/VPP pin, but the RB5/PGM pin is then dedicated to controlling Program mode entry and is not available as a general purpose I/O pin.

LVP is enabled in erased devices.

While programming, using Single-Supply Programming, VDD is applied to the $\overline{\text{MCLR}}$/VPP pin as in normal execution mode. To enter Programming mode, VDD is applied to the PGM pin.

> **Note 1:** High-voltage programming is always available, regardless of the state of the LVP bit or the PGM pin, by applying VIHH to the $\overline{\text{MCLR}}$ pin.
>
> **2:** When Single-Supply Programming is enabled, the RB5 pin can no longer be used as a general purpose I/O pin.
>
> **3:** When LVP is enabled externally, pull the PGM pin to VSS to allow normal program execution.

If Single-Supply ICSP Programming mode will not be used, the LVP bit can be cleared and RB5/PGM becomes available as the digital I/O pin RB5. The LVP bit may be set or cleared only when using standard high-voltage programming (VIHH applied to the $\overline{\text{MCLR}}$/VPP pin). Once LVP has been disabled, only the standard high-voltage programming is available and must be used to program the device.

Memory that is not code-protected can be erased using either a block erase, or erased row by row, then written at any specified VDD. If code-protected memory is to be erased, a block erase is required. If a block erase is to be performed when using Single-Supply Programming, the device must be supplied with VDD of 4.5V to 5.5V.

24.0 INSTRUCTION SET SUMMARY

The PIC18 instruction set adds many enhancements to the previous PIC® instruction sets, while maintaining an easy migration from these PIC instruction sets.

Most instructions are a single program memory word (16 bits), but there are three instructions that require two program memory locations.

Each single-word instruction is a 16-bit word divided into an opcode, which specifies the instruction type and one or more operands, which further specify the operation of the instruction.

The instruction set is highly orthogonal and is grouped into four basic categories:

- **Byte-oriented** operations
- **Bit-oriented** operations
- **Literal** operations
- **Control** operations

The PIC18 instruction set summary in Table 24-2 lists **byte-oriented**, **bit-oriented**, **literal** and **control** operations. Table 24-1 shows the opcode field descriptions.

Most **byte-oriented** instructions have three operands:

1. The file register (specified by 'f')
2. The destination of the result (specified by 'd')
3. The accessed memory (specified by 'a')

The file register designator, 'f', specifies which file register is to be used by the instruction.

The destination designator, 'd', specifies where the result of the operation is to be placed. If 'd' is '0', the result is placed in the WREG register. If 'd' is '1', the result is placed in the file register specified in the instruction.

All **bit-oriented** instructions have three operands:

1. The file register (specified by 'f')
2. The bit in the file register (specified by 'b')
3. The accessed memory (specified by 'a')

The bit field designator, 'b', selects the number of the bit affected by the operation, while the file register designator, 'f', represents the number of the file in which the bit is located.

The **literal** instructions may use some of the following operands:

- A literal value to be loaded into a file register (specified by 'k')
- The desired FSR register to load the literal value into (specified by 'f')
- No operand required (specified by '—')

The **control** instructions may use some of the following operands:

- A program memory address (specified by 'n')
- The mode of the call or return instructions (specified by 's')
- The mode of the table read and table write instructions (specified by 'm')
- No operand required (specified by '—')

All instructions are a single word, except for three double-word instructions. These three instructions were made double word instructions so that all the required information is available in these 32 bits. In the second word, the 4 MSbs are '1's. If this second word is executed as an instruction (by itself), it will execute as a NOP.

All single-word instructions are executed in a single instruction cycle, unless a conditional test is true or the program counter is changed as a result of the instruction. In these cases, the execution takes two instruction cycles with the additional instruction cycle(s) executed as a NOP.

The double word instructions execute in two instruction cycles.

One instruction cycle consists of four oscillator periods. Thus, for an oscillator frequency of 4 MHz, the normal instruction execution time is 1 μs. If a conditional test is true or the program counter is changed as a result of an instruction, the instruction execution time is 2 μs. Two-word branch instructions (if true) would take 3 μs.

Figure 24-1 shows the general formats that the instructions can have.

All examples use the format 'nnh' to represent a hexadecimal number, where 'h' signifies a hexadecimal digit.

The Instruction Set Summary, shown in Table 24-2, lists the instructions recognized by the Microchip Assembler (MPASM™ Assembler). **Section 24.2 "Instruction Set"** provides a description of each instruction.

24.1 Read-Modify-Write Operations

Any instruction that specifies a file register as part of the instruction performs a Read-Modify-Write (R-M-W) operation. The register is read, the data is modified, and the result is stored according to either the instruction or the destination designator, 'd'. A read operation is performed on a register even if the instruction writes to that register.

For example, a "BCF PORTB, 1" instruction will read PORTB, clear bit 1 of the data, then write the result back to PORTB. The read operation would have the unintended result that any condition that sets the RBIF flag would be cleared. The R-M-W operation may also copy the level of an input pin to its corresponding output latch.

TABLE 24-1: OPCODE FIELD DESCRIPTIONS

Field	Description
a	RAM access bit: a = 0: RAM location in Access RAM (BSR register is ignored) a = 1: RAM bank is specified by BSR register
bbb	Bit address within an 8-bit file register (0 to 7).
BSR	Bank Select Register. Used to select the current RAM bank.
d	Destination select bit: d = 0: store result in WREG d = 1: store result in file register f
dest	Destination either the WREG register or the specified register file locations.
f	8-bit register file address (0x00 to 0xFF).
fs	12-bit register file address (0x000 to 0xFFF). This is the source address.
fd	12-bit register file address (0x000 to 0xFFF). This is the destination address.
k	Literal field, constant data or label (may be either an 8-bit, 12-bit or a 20-bit value).
label	Label name.
mm * *+ *- +*	The mode of the TBLPTR register for the table read and table write instructions. Only used with table read and table write instructions: No Change to register (such as TBLPTR with table reads and writes). Post-Increment register (such as TBLPTR with table reads and writes). Post-Decrement register (such as TBLPTR with table reads and writes). Pre-Increment register (such as TBLPTR with table reads and writes).
n	The relative address (2's complement number) for relative branch instructions, or the direct address for Call/Branch and Return instructions.
PRODH	Product of Multiply High Byte.
PRODL	Product of Multiply Low Byte.
s	Fast Call/Return Mode Select bit: s = 0: do not update into/from Shadow registers s = 1: certain registers loaded into/from shadow registers (Fast mode)
u	Unused or Unchanged.
WREG	Working register (accumulator).
x	Don't care ('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.
TBLPTR	21-bit Table Pointer (points to a Program Memory location).
TABLAT	8-bit Table Latch.
TOS	Top-of-Stack.
PC	Program Counter.
PCL	Program Counter Low Byte.
PCH	Program Counter High Byte.
PCLATH	Program Counter High Byte Latch.
PCLATU	Program Counter Upper Byte Latch.
GIE	Global Interrupt Enable bit.
WDT	Watchdog Timer.
$\overline{TO}$	Time-out bit.
$\overline{PD}$	Power-Down bit.
C, DC, Z, OV, N	ALU Status bits: Carry, Digit Carry, Zero, Overflow, Negative.
[ ]	Optional.
( )	Contents.
→	Assigned to.
< >	Register bit field.
∈	In the set of.
italics	User-defined term (font is Courier New).

FIGURE 24-1: GENERAL FORMAT FOR INSTRUCTIONS

Byte-oriented file register operations — **Example Instruction**

15 … 10	9	8	7 … 0
OPCODE	d	a	f (FILE #)

Example: `ADDWF MYREG, W, B`

d = `0` for result destination to be WREG register
d = `1` for result destination to be file register (f)
a = `0` to force Access Bank
a = `1` for BSR to select bank
f = 8-bit file register address

Byte to Byte move operations (2-word)

15 … 12	11 … 0
OPCODE	f (Source FILE #)
`1111`	f (Destination FILE #)

Example: `MOVFF MYREG1, MYREG2`

f = 12-bit file register address

Bit-oriented file register operations

15 … 12	11 … 9	8	7 … 0
OPCODE	b (BIT #)	a	f (FILE #)

Example: `BSF MYREG, bit, B`

b = 3-bit position of bit in file register (f)
a = `0` to force Access Bank
a = `1` for BSR to select bank
f = 8-bit file register address

Literal operations

15 … 8	7 … 0
OPCODE	k (literal)

Example: `MOVLW 0x7F`

k = 8-bit immediate value

Control operations

`CALL`, `GOTO` and Branch operations

15 … 8	7 … 0
OPCODE	n<7:0> (literal)

15 … 12	11 … 0
`1111`	n<19:8> (literal)

Example: `GOTO Label`

n = 20-bit immediate value

15 … 9	8	7 … 0
OPCODE	S	n<7:0> (literal)

15 … 12	11 … 0
	n<19:8> (literal)

Example: `CALL MYFUNC`

S = Fast bit

15 … 11	10 … 0
OPCODE	n<10:0> (literal)

Example: `BRA MYFUNC`

15 … 8	7 … 0
OPCODE	n<7:0> (literal)

Example: `BC MYFUNC`

TABLE 24-2: PIC18FXXXX INSTRUCTION SET

Mnemonic,	Operands	Description	Cycles	16-Bit Instruction Word MSb			LSb	Status Affected	Notes
BYTE-ORIENTED FILE REGISTER OPERATIONS									
ADDWF	f, d, a	Add WREG and f	1	0010	01da	ffff	ffff	C, DC, Z, OV, N	1, 2
ADDWFC	f, d, a	Add WREG and Carry bit to f	1	0010	00da	ffff	ffff	C, DC, Z, OV, N	1, 2
ANDWF	f, d, a	AND WREG with f	1	0001	01da	ffff	ffff	Z, N	1,2
CLRF	f, a	Clear f	1	0110	101a	ffff	ffff	Z	2
COMF	f, d, a	Complement f	1	0001	11da	ffff	ffff	Z, N	1, 2
CPFSEQ	f, a	Compare f with WREG, Skip =	1 (2 or 3)	0110	001a	ffff	ffff	None	4
CPFSGT	f, a	Compare f with WREG, Skip >	1 (2 or 3)	0110	010a	ffff	ffff	None	4
CPFSLT	f, a	Compare f with WREG, Skip <	1 (2 or 3)	0110	000a	ffff	ffff	None	1, 2
DECF	f, d, a	Decrement f	1	0000	01da	ffff	ffff	C, DC, Z, OV, N	1, 2, 3, 4
DECFSZ	f, d, a	Decrement f, Skip if 0	1 (2 or 3)	0010	11da	ffff	ffff	None	1, 2, 3, 4
DCFSNZ	f, d, a	Decrement f, Skip if Not 0	1 (2 or 3)	0100	11da	ffff	ffff	None	1, 2
INCF	f, d, a	Increment f	1	0010	10da	ffff	ffff	C, DC, Z, OV, N	1, 2, 3, 4
INCFSZ	f, d, a	Increment f, Skip if 0	1 (2 or 3)	0011	11da	ffff	ffff	None	4
INFSNZ	f, d, a	Increment f, Skip if Not 0	1 (2 or 3)	0100	10da	ffff	ffff	None	1, 2
IORWF	f, d, a	Inclusive OR WREG with f	1	0001	00da	ffff	ffff	Z, N	1, 2
MOVF	f, d, a	Move f	1	0101	00da	ffff	ffff	Z, N	1
MOVFF	f_s, f_d	Move f_s (source) to 1st word	2	1100	ffff	ffff	ffff	None	
		f_d (destination) 2nd word		1111	ffff	ffff	ffff		
MOVWF	f, a	Move WREG to f	1	0110	111a	ffff	ffff	None	
MULWF	f, a	Multiply WREG with f	1	0000	001a	ffff	ffff	None	
NEGF	f, a	Negate f	1	0110	110a	ffff	ffff	C, DC, Z, OV, N	1, 2
RLCF	f, d, a	Rotate Left f through Carry	1	0011	01da	ffff	ffff	C, Z, N	
RLNCF	f, d, a	Rotate Left f (No Carry)	1	0100	01da	ffff	ffff	Z, N	1, 2
RRCF	f, d, a	Rotate Right f through Carry	1	0011	00da	ffff	ffff	C, Z, N	
RRNCF	f, d, a	Rotate Right f (No Carry)	1	0100	00da	ffff	ffff	Z, N	
SETF	f, a	Set f	1	0110	100a	ffff	ffff	None	
SUBFWB	f, d, a	Subtract f from WREG with Borrow	1	0101	01da	ffff	ffff	C, DC, Z, OV, N	1, 2
SUBWF	f, d, a	Subtract WREG from f	1	0101	11da	ffff	ffff	C, DC, Z, OV, N	
SUBWFB	f, d, a	Subtract WREG from f with Borrow	1	0101	10da	ffff	ffff	C, DC, Z, OV, N	1, 2
SWAPF	f, d, a	Swap Nibbles in f	1	0011	10da	ffff	ffff	None	4
TSTFSZ	f, a	Test f, Skip if 0	1 (2 or 3)	0110	011a	ffff	ffff	None	1, 2
XORWF	f, d, a	Exclusive OR WREG with f	1	0001	10da	ffff	ffff	Z, N	
BIT-ORIENTED FILE REGISTER OPERATIONS									
BCF	f, b, a	Bit Clear f	1	1001	bbba	ffff	ffff	None	1, 2
BSF	f, b, a	Bit Set f	1	1000	bbba	ffff	ffff	None	1, 2
BTFSC	f, b, a	Bit Test f, Skip if Clear	1 (2 or 3)	1011	bbba	ffff	ffff	None	3, 4
BTFSS	f, b, a	Bit Test f, Skip if Set	1 (2 or 3)	1010	bbba	ffff	ffff	None	3, 4
BTG	f, b, a	Bit Toggle f	1	0111	bbba	ffff	ffff	None	1, 2

Note 1: When a PORT register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is '`1`' for a pin configured as an input and is driven low by an external device, the data will be written back with a '`0`'.

2: If this instruction is executed on the TMR0 register (and, where applicable, d = `1`), the prescaler will be cleared if assigned.

3: 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`.

4: Some instructions are 2-word instructions. The second word of these instructions will be executed as a `NOP` unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.

5: If the table write starts the write cycle to internal memory, the write will continue until terminated.

TABLE 24-2: PIC18FXXXX INSTRUCTION SET (CONTINUED)

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word MSb			LSb	Status Affected	Notes
CONTROL OPERATIONS									
BC	n	Branch if Carry	1 (2)	1110	0010	nnnn	nnnn	None	
BN	n	Branch if Negative	1 (2)	1110	0110	nnnn	nnnn	None	
BNC	n	Branch if Not Carry	1 (2)	1110	0011	nnnn	nnnn	None	
BNN	n	Branch if Not Negative	1 (2)	1110	0111	nnnn	nnnn	None	
BNOV	n	Branch if Not Overflow	1 (2)	1110	0101	nnnn	nnnn	None	
BNZ	n	Branch if Not Zero	2	1110	0001	nnnn	nnnn	None	
BOV	n	Branch if Overflow	1 (2)	1110	0100	nnnn	nnnn	None	
BRA	n	Branch Unconditionally	1 (2)	1101	0nnn	nnnn	nnnn	None	
BZ	n	Branch if Zero	1 (2)	1110	0000	nnnn	nnnn	None	
CALL	n, s	Call Subroutine 1st word	2	1110	110s	kkkk	kkkk	None	
		2nd word		1111	kkkk	kkkk	kkkk		
CLRWDT	—	Clear Watchdog Timer	1	0000	0000	0000	0100	$\overline{TO}$, $\overline{PD}$	
DAW	—	Decimal Adjust WREG	1	0000	0000	0000	0111	C, DC	
GOTO	n	Go to Address 1st word	2	1110	1111	kkkk	kkkk	None	
		2nd word		1111	kkkk	kkkk	kkkk		
NOP	—	No Operation	1	0000	0000	0000	0000	None	
NOP	—	No Operation	1	1111	xxxx	xxxx	xxxx	None	4
POP	—	Pop Top of Return Stack (TOS)	1	0000	0000	0000	0110	None	
PUSH	—	Push Top of Return Stack (TOS)	1	0000	0000	0000	0101	None	
RCALL	n	Relative Call	2	1101	1nnn	nnnn	nnnn	None	
RESET		Software Device Reset	1	0000	0000	1111	1111	All	
RETFIE	s	Return from Interrupt Enable	2	0000	0000	0001	000s	GIE/GIEH, PEIE/GIEL	
RETLW	k	Return with Literal in WREG	2	0000	1100	kkkk	kkkk	None	
RETURN	s	Return from Subroutine	2	0000	0000	0001	001s	None	
SLEEP	—	Go into Standby mode	1	0000	0000	0000	0011	$\overline{TO}$, $\overline{PD}$	

Note 1: When a PORT register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as an input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and, where applicable, d = 1), the prescaler will be cleared if assigned.

3: 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`.

4: Some instructions are 2-word instructions. The second word of these instructions will be executed as a `NOP` unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.

5: If the table write starts the write cycle to internal memory, the write will continue until terminated.

TABLE 24-2: PIC18FXXXX INSTRUCTION SET (CONTINUED)

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word				Status Affected	Notes
				MSb			LSb		
LITERAL OPERATIONS									
ADDLW	k	Add Literal and WREG	1	0000	1111	kkkk	kkkk	C, DC, Z, OV, N	
ANDLW	k	AND Literal with WREG	1	0000	1011	kkkk	kkkk	Z, N	
IORLW	k	Inclusive OR Literal with WREG	1	0000	1001	kkkk	kkkk	Z, N	
LFSR	f, k	Load Literal (12-bit) 2nd word	2	1110	1110	00ff	kkkk	None	
		to FSRx 1st word		1111	0000	kkkk	kkkk		
MOVLB	k	Move Literal to BSR<3:0>	1	0000	0001	0000	kkkk	None	
MOVLW	k	Move Literal to WREG	1	0000	1110	kkkk	kkkk	None	
MULLW	k	Multiply Literal with WREG	1	0000	1101	kkkk	kkkk	None	
RETLW	k	Return with Literal in WREG	2	0000	1100	kkkk	kkkk	None	
SUBLW	k	Subtract WREG from Literal	1	0000	1000	kkkk	kkkk	C, DC, Z, OV, N	
XORLW	k	Exclusive OR Literal with WREG	1	0000	1010	kkkk	kkkk	Z, N	
DATA MEMORY ↔ PROGRAM MEMORY OPERATIONS									
TBLRD*		Table Read	2	0000	0000	0000	1000	None	
TBLRD*+		Table Read with Post-Increment		0000	0000	0000	1001	None	
TBLRD*-		Table Read with Post-Decrement		0000	0000	0000	1010	None	
TBLRD+*		Table Read with Pre-Increment		0000	0000	0000	1011	None	
TBLWT*		Table Write	2 (5)	0000	0000	0000	1100	None	
TBLWT*+		Table Write with Post-Increment		0000	0000	0000	1101	None	
TBLWT*-		Table Write with Post-Decrement		0000	0000	0000	1110	None	
TBLWT+*		Table Write with Pre-Increment		0000	0000	0000	1111	None	

Note 1: When a PORT register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as an input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and, where applicable, d = 1), the prescaler will be cleared if assigned.

3: 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`.

4: Some instructions are 2-word instructions. The second word of these instructions will be executed as a `NOP` unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.

5: If the table write starts the write cycle to internal memory, the write will continue until terminated.

24.2 Instruction Set

ADDLW — ADD Literal to W

Syntax:	[*label*] ADDLW k
Operands:	$0 \le k \le 255$
Operation:	(W) + k → W
Status Affected:	N, OV, C, DC, Z
Encoding:	`0000` `1111` `kkkk` `kkkk`
Description:	The contents of W are added to the 8-bit literal 'k' and the result is placed in W.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example: `ADDLW 0x15`

Before Instruction
W = 0x10
After Instruction
W = 0x25

ADDWF — ADD W to f

Syntax:	[*label*] ADDWF f [,d [,a]]
Operands:	$0 \le f \le 255$ $d \in [0,1]$ $a \in [0,1]$
Operation:	(W) + (f) → dest
Status Affected:	N, OV, C, DC, Z
Encoding:	`0010` `01da` `ffff` `ffff`
Description:	Add W to register, 'f'. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register, 'f'. If 'a' is '0', the Access Bank will be selected. If 'a' is '1', the BSR is used.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `ADDWF REG, W`

Before Instruction
W = 0x17
REG = 0xC2
After Instruction
W = 0xD9
REG = 0xC2

ADDWFC — ADD W and Carry bit to f

Syntax: [*label*] ADDWFC f [,d [,a]]

Operands: $0 \leq f \leq 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (W) + (f) + (C) → dest

Status Affected: N, OV, C, DC, Z

Encoding:

`0010`	`00da`	`ffff`	`ffff`

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'. If 'a' is '0', the Access Bank will be selected. If 'a' is '1', the BSR will not be overridden.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `ADDWFC REG, W`

Before Instruction
Carry bit = 1
REG = 0x02
W = 0x4D

After Instruction
Carry bit = 0
REG = 0x02
W = 0x50

ANDLW — AND Literal with W

Syntax: [*label*] ANDLW k

Operands: $0 \leq k \leq 255$

Operation: (W) .AND. k → W

Status Affected: N, Z

Encoding:

`0000`	`1011`	`kkkk`	`kkkk`

Description: The contents of W are ANDed with the 8-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example: `ANDLW 0x5F`

Before Instruction
W = 0xA3

After Instruction
W = 0x03

ANDWF — AND W with f

Syntax:	[*label*] ANDWF f [,d [,a]]
Operands:	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
Operation:	(W) .AND. (f) → dest
Status Affected:	N, Z
Encoding:	`0001` `01da` `ffff` `ffff`
Description:	The contents of W are ANDed with register, 'f'. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register, 'f'. If 'a' is '0', the Access Bank will be selected. If 'a' is '1', the BSR will not be overridden.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `ANDWF REG, W`

Before Instruction
W = 0x17
REG = 0xC2

After Instruction
W = 0x02
REG = 0xC2

BC — Branch if Carry

Syntax:	[*label*] BC n
Operands:	-128 ≤ n ≤ 127
Operation:	if Carry bit is '1', (PC) + 2 + 2n → PC
Status Affected:	None
Encoding:	`1110` `0010` `nnnn` `nnnn`
Description:	If the Carry bit is '1', then the program will branch. The 2's complement number, '2n', is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.
Words:	1
Cycles:	1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: `HERE BC JUMP`

Before Instruction
PC = address `(HERE)`

After Instruction
If Carry = 1;
PC = address `(JUMP)`
If Carry = 0;
PC = address `(HERE + 2)`

BCF — Bit Clear f

Syntax: [*label*] BCF f,b[,a]

Operands: $0 \le f \le 255$
$0 \le b \le 7$
$a \in [0,1]$

Operation: 0 → f<b>

Status Affected: None

Encoding:

`1001`	`bbba`	`ffff`	`ffff`

Description: Bit 'b' in register, 'f', is cleared. If 'a' is '`0`', the Access Bank will be selected, overriding the BSR value. If 'a' = `1`, then the bank will be selected as per the BSR value.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example: `BCF FLAG_REG, 7`

Before Instruction
FLAG_REG = 0xC7

After Instruction
FLAG_REG = 0x47

BN — Branch if Negative

Syntax: [*label*] BN n

Operands: $-128 \le n \le 127$

Operation: if Negative bit is '`1`',
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

`1110`	`0110`	`nnnn`	`nnnn`

Description: If the Negative bit is '`1`', then the program will branch.
The 2's complement number, '2n', is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: `HERE BN Jump`

Before Instruction
PC = address (`HERE`)

After Instruction
If Negative = 1;
PC = address (`Jump`)
If Negative = 0;
PC = address (`HERE + 2`)

BNC — Branch if Not Carry

Syntax:	[*label*] BNC n
Operands:	$-128 \leq n \leq 127$
Operation:	if Carry bit is ‘0’, (PC) + 2 + 2n → PC
Status Affected:	None
Encoding:	`1110` `0011` `nnnn` `nnnn`
Description:	If the Carry bit is ‘0’, then the program will branch. The 2’s complement number, ‘2n’, is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.
Words:	1
Cycles:	1(2)

Q Cycle Activity:

If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal ‘n’	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal ‘n’	Process Data	No operation

Example: `HERE BNC Jump`

Before Instruction
PC = address (HERE)

After Instruction
If Carry = 0;
PC = address (Jump)
If Carry = 1;
PC = address (HERE + 2)

BNN — Branch if Not Negative

Syntax:	[*label*] BNN n
Operands:	$-128 \leq n \leq 127$
Operation:	if Negative bit is ‘0’, (PC) + 2 + 2n → PC
Status Affected:	None
Encoding:	`1110` `0111` `nnnn` `nnnn`
Description:	If the Negative bit is ‘0’, then the program will branch. The 2’s complement number, ‘2n’, is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.
Words:	1
Cycles:	1(2)

Q Cycle Activity:

If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal ‘n’	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal ‘n’	Process Data	No operation

Example: `HERE BNN Jump`

Before Instruction
PC = address (HERE)

After Instruction
If Negative = 0;
PC = address (Jump)
If Negative = 1;
PC = address (HERE + 2)

BNOV — Branch if Not Overflow

Syntax:	[*label*] BNOV n
Operands:	-128 ≤ n ≤ 127
Operation:	if Overflow bit is '0', (PC) + 2 + 2n → PC
Status Affected:	None
Encoding:	`1110` `0101` `nnnn` `nnnn`
Description:	If the Overflow bit is '0', then the program will branch. The 2's complement number, '2n', is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.
Words:	1
Cycles:	1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: `HERE BNOV Jump`

Before Instruction
PC = address (`HERE`)
After Instruction
If Overflow = 0;
PC = address (`Jump`)
If Overflow = 1;
PC = address (`HERE + 2`)

BNZ — Branch if Not Zero

Syntax:	[*label*] BNZ n
Operands:	-128 ≤ n ≤ 127
Operation:	if Zero bit is '0', (PC) + 2 + 2n → PC
Status Affected:	None
Encoding:	`1110` `0001` `nnnn` `nnnn`
Description:	If the Zero bit is '0', then the program will branch. The 2's complement number, '2n', is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.
Words:	1
Cycles:	1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: `HERE BNZ Jump`

Before Instruction
PC = address (`HERE`)
After Instruction
If Zero = 0;
PC = address (`Jump`)
If Zero = 1;
PC = address (`HERE + 2`)

BRA — Unconditional Branch

Syntax:	[*label*] BRA n
Operands:	-1024 ≤ n ≤ 1023
Operation:	(PC) + 2 + 2n → PC
Status Affected:	None
Encoding:	`1101` `0nnn` `nnnn` `nnnn`
Description:	Add the 2's complement number, '2n', to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is a two-cycle instruction.
Words:	1
Cycles:	2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

<u>Example:</u> `HERE BRA Jump`

Before Instruction
PC = address (`HERE`)

After Instruction
PC = address (`Jump`)

BSF — Bit Set f

Syntax:	[*label*] BSF f,b[,a]
Operands:	0 ≤ f ≤ 255 0 ≤ b ≤ 7 a ∈ [0,1]
Operation:	1 → f<b>
Status Affected:	None
Encoding:	`1000` `bbba` `ffff` `ffff`
Description:	Bit 'b' in register, 'f', is set. If 'a' is '`0`', Access Bank will be selected, overriding the BSR value. If 'a' = `1`, then the bank will be selected as per the BSR value.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

<u>Example:</u> `BSF FLAG_REG, 7`

Before Instruction
FLAG_REG = 0x0A

After Instruction
FLAG_REG = 0x8A

BTFSC — Bit Test File, Skip if Clear

Syntax: [*label*] BTFSC f,b[,a]

Operands: $0 \le f \le 255$
$0 \le b \le 7$
$a \in [0,1]$

Operation: skip if (f<b>) = `0`

Status Affected: None

Encoding:

1011	bbba	ffff	ffff

Description: If bit 'b' in register, 'f', is '`0`', then the next instruction is skipped.
If bit 'b' is '`0`', then the next instruction fetched during the current instruction execution is discarded, and a `NOP` is executed instead, making this a two-cycle instruction. If 'a' is '`0`', the Access Bank will be selected, overriding the BSR value. If 'a' = `1`, then the bank will be selected as per the BSR value.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:

```
HERE    BTFSC   FLAG, 1
FALSE   :
TRUE    :
```

Before Instruction
PC = address (`HERE`)

After Instruction
If FLAG<1> = 0;
PC = address (`TRUE`)
If FLAG<1> = 1;
PC = address (`FALSE`)

BTFSS — Bit Test File, Skip if Set

Syntax: [*label*] BTFSS f,b[,a]

Operands: $0 \le f \le 255$
$0 \le b < 7$
$a \in [0,1]$

Operation: skip if (f<b>) = `1`

Status Affected: None

Encoding:

1010	bbba	ffff	ffff

Description: If bit 'b' in register, 'f', is '`1`', then the next instruction is skipped.
If bit 'b' is '`1`', then the next instruction fetched during the current instruction execution, is discarded and a `NOP` is executed instead, making this a two-cycle instruction. If 'a' is '`0`', the Access Bank will be selected, overriding the BSR value. If 'a' = `1`, then the bank will be selected as per the BSR value.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:

```
HERE    BTFSS   FLAG, 1
FALSE   :
TRUE    :
```

Before Instruction
PC = address (`HERE`)

After Instruction
If FLAG<1> = 0;
PC = address (`FALSE`)
If FLAG<1> = 1;
PC = address (`TRUE`)

BTG — Bit Toggle f

Syntax: [*label*] BTG f,b[,a]

Operands: $0 \le f \le 255$
$0 \le b < 7$
$a \in [0,1]$

Operation: $(\overline{f<b>}) \rightarrow f<b>$

Status Affected: None

Encoding:

0111	bbba	ffff	ffff

Description: Bit 'b' in data memory location, 'f', is inverted. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example:
```
BTG     PORTC,  4
```
Before Instruction:
PORTC = 0111 0101 [0x75]

After Instruction:
PORTC = 0110 0101 [0x65]

BOV — Branch if Overflow

Syntax: [*label*] BOV n

Operands: $-128 \le n \le 127$

Operation: if Overflow bit is '1',
$(PC) + 2 + 2n \rightarrow PC$

Status Affected: None

Encoding:

1110	0100	nnnn	nnnn

Description: If the Overflow bit is '1', then the program will branch.
The 2's complement number, '2n', is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example:
```
HERE      BOV   JUMP
```
Before Instruction
PC = address (HERE)

After Instruction
If Overflow = 1;
PC = address (JUMP)
If Overflow = 0;
PC = address (HERE + 2)

BZ — Branch if Zero

Syntax: [*label*] BZ n

Operands: $-128 \le n \le 127$

Operation: if Zero bit is '1',
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0000	nnnn	nnnn

Description: If the Zero bit is '1', then the program will branch.
The 2's complement number, '2n', is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a two-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: `HERE BZ Jump`

Before Instruction
PC = address (HERE)

After Instruction
If Zero = 1;
PC = address (Jump)
If Zero = 0;
PC = address (HERE + 2)

CALL — Subroutine Call

Syntax: [*label*] CALL k [,s]

Operands: $0 \le k \le 1048575$
$s \in [0,1]$

Operation: (PC) + 4 → TOS,
k → PC<20:1>;
if s = 1:
(W) → WS,
(STATUS) → STATUSS,
(BSR) → BSRS

Status Affected: None

Encoding:
1st word (k<7:0>)
2nd word(k<19:8>)

1110	110s	$k_7$kkk	kkkk$_0$
1111	k_{19}kkk	kkkk	kkkk$_8$

Description: Subroutine call of entire 2-Mbyte memory range. First, the return address (PC + 4) is pushed onto the return stack. If 's' = 1, the W, STATUS and BSR registers are also pushed into their respective shadow registers, WS, STATUSS and BSRS. If 's' = 0, no update occurs. Then, the 20-bit value, 'k', is loaded into PC<20:1>. CALL is a two-cycle instruction.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'<7:0>,	Push PC to Stack	Read literal 'k'<19:8>, Write to PC
No operation	No operation	No operation	No operation

Example: `HERE CALL THERE,FAST`

Before Instruction
PC = address (HERE)

After Instruction
PC = address (THERE)
TOS = address (HERE + 4)
WS = W
BSRS = BSR
STATUSS = STATUS

CLRF — Clear f

Syntax:	[*label*] CLRF f [,a]
Operands:	$0 \le f \le 255$ $a \in [0,1]$
Operation:	000h → f, 1 → Z
Status Affected:	Z
Encoding:	`0110` `101a` `ffff` `ffff`
Description:	Clears the contents of the specified register. If 'a' is '`0`', the Access Bank will be selected, overriding the BSR value. If 'a' = `1`, then the bank will be selected as per the BSR value.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example: `CLRF FLAG_REG`

Before Instruction
FLAG_REG = 0x5A

After Instruction
FLAG_REG = 0x00

CLRWDT — Clear Watchdog Timer

Syntax:	[*label*] CLRWDT
Operands:	None
Operation:	000h → WDT, 000h → WDT postscaler, 1 → $\overline{TO}$, 1 → $\overline{PD}$
Status Affected:	$\overline{TO}$, $\overline{PD}$
Encoding:	`0000` `0000` `0000` `0100`
Description:	`CLRWDT` instruction resets the Watchdog Timer. It also resets the postscaler of the WDT. Status bits $\overline{TO}$ and $\overline{PD}$ are set.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	No operation

Example: `CLRWDT`

Before Instruction
WDT Counter = ?

After Instruction
WDT Counter = 0x00
WDT Postscaler = 0
$\overline{TO}$ = 1
$\overline{PD}$ = 1

COMF — Complement f

Syntax: [*label*] COMF f [,d [,a]]

Operands: $0 \leq f \leq 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: $(\bar{f}) \rightarrow$ dest

Status Affected: N, Z

Encoding:

0001	11da	ffff	ffff

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'. If 'a' is 0, the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `COMF REG, W`

Before Instruction
REG = 0x13

After Instruction
REG = 0x13
W = 0xEC

CPFSEQ — Compare f with W, Skip if f = W

Syntax: [*label*] CPFSEQ f [,a]

Operands: $0 \leq f \leq 255$
$a \in [0,1]$

Operation: (f) – (W),
skip if (f) = (W)
(unsigned comparison)

Status Affected: None

Encoding:

0110	001a	ffff	ffff

Description: Compares the contents of data memory location, 'f', to the contents of W by performing an unsigned subtraction.
If 'f' = W, then the fetched instruction is discarded and a NOP is executed instead, making this a two-cycle instruction. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:

```
HERE     CPFSEQ REG
NEQUAL   :
EQUAL    :
```

Before Instruction
PC Address = HERE
W = ?
REG = ?

After Instruction
If REG = W;
PC = Address (EQUAL)
If REG ≠ W;
PC = Address (NEQUAL)

CPFSGT — Compare f with W, Skip if f > W

Syntax: [*label*] CPFSGT f [,a]

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: (f) – (W),
skip if (f) > (W)
(unsigned comparison)

Status Affected: None

Encoding:

0110	010a	ffff	ffff

Description: Compares the contents of data memory location, 'f', to the contents of the W by performing an unsigned subtraction.
If the contents of 'f' are greater than the contents of WREG, then the fetched instruction is discarded and a `NOP` is executed instead, making this a two-cycle instruction. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:

```
HERE      CPFSGT REG
NGREATER  :
GREATER   :
```

Before Instruction
PC = Address (`HERE`)
W = ?

After Instruction
If REG > W;
PC = Address (`GREATER`)
If REG ≤ W;
PC = Address (`NGREATER`)

CPFSLT — Compare f with W, Skip if f < W

Syntax: [*label*] CPFSLT f [,a]

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: (f) – (W),
skip if (f) < (W)
(unsigned comparison)

Status Affected: None

Encoding:

0110	000a	ffff	ffff

Description: Compares the contents of data memory location, 'f', to the contents of W by performing an unsigned subtraction.
If the contents of 'f' are less than the contents of W, then the fetched instruction is discarded and a `NOP` is executed instead, making this a two-cycle instruction. If 'a' is '0', the Access Bank will be selected. If 'a' is '1', the BSR will not be overridden.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:

```
HERE   CPFSLT REG
NLESS  :
LESS   :
```

Before Instruction
PC = Address (`HERE`)
W = ?

After Instruction
If REG < W;
PC = Address (`LESS`)
If REG ≥ W;
PC = Address (`NLESS`)

DAW — Decimal Adjust W Register

Syntax: [*label*] DAW

Operands: None

Operation:
If [W<3:0> > 9] or [DC = 1] then,
(W<3:0>) + 6 → W<3:0>;
else,
(W<3:0>) → W<3:0>;

If [W<7:4> 9] or [C = 1] then,
(W<7:4>) + 6 → W<7:4>;
else,
(W<7:4>) → W<7:4>

Status Affected: C, DC

Encoding:

0000	0000	0000	0111

Description: `DAW` adjusts the 8-bit value in W, resulting from the earlier addition of two variables (each in packed BCD format) and produces a correct packed BCD result. The Carry bit may be set by `DAW` regardless of its setting prior to the `DAW` instruction.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register W	Process Data	Write W

Example 1: `DAW`

Before Instruction
W = 0xA5
C = 0
DC = 0

After Instruction
W = 0x05
C = 1
DC = 0

Example 2:

Before Instruction
W = 0xCE
C = 0
DC = 0

After Instruction
W = 0x34
C = 1
DC = 0

DECF — Decrement f

Syntax: [*label*] DECF f [,d [,a]]

Operands:
$0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f) – 1 → dest

Status Affected: C, DC, N, OV, Z

Encoding:

0000	01da	ffff	ffff

Description: Decrement register, ‘f’,. If ‘d’ is ‘0’, the result is stored in W. If ‘d’ is ‘1’, the result is stored back in register, ‘f’. If ‘a’ is ‘0’, the Access Bank will be selected, overriding the BSR value. If ‘a’ = 1, then the bank will be selected as per the BSR value.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write to destination

Example: `DECF CNT,`

Before Instruction
CNT = 0x01
Z = 0

After Instruction
CNT = 0x00
Z = 1

DECFSZ — Decrement f, Skip if 0

Syntax:	[*label*] DECFSZ f [,d [,a]]
Operands:	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
Operation:	(f) – 1 → dest, skip if result = `0`
Status Affected:	None
Encoding:	`0010` `11da` `ffff` `ffff`
Description:	The contents of register, 'f', are decremented. If 'd' is '`0`', the result is placed in W. If 'd' is '`1`', the result is placed back in register, 'f'. If the result is '`0`', the next instruction, which is already fetched, is discarded, and a `NOP` is executed instead, making it a two-cycle instruction. If 'a' is '`0`', the Access Bank will be selected, overriding the BSR value. If 'a' = `1`, then the bank will be selected as per the BSR value.
Words:	1
Cycles:	1(2) **Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

<u>Example:</u>

```
HERE      DECFSZ   CNT
          GOTO     LOOP
CONTINUE
```

Before Instruction
PC = Address (`HERE`)

After Instruction
CNT = CNT – 1
If CNT = 0;
PC = Address (`CONTINUE`)
If CNT ≠ 0;
PC = Address (`HERE + 2`)

DCFSNZ — Decrement f, Skip if Not 0

Syntax:	[*label*] DCFSNZ f [,d [,a]]
Operands:	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
Operation:	(f) – 1 → dest, skip if result ≠ `0`
Status Affected:	None
Encoding:	`0100` `11da` `ffff` `ffff`
Description:	The contents of register, 'f', are decremented. If 'd' is '`0`', the result is placed in W. If 'd' is '`1`', the result is placed back in register, 'f'. If the result is not '`0`', the next instruction, which is already fetched, is discarded, and a `NOP` is executed instead, making it a two-cycle instruction. If 'a' is '`0`', the Access Bank will be selected, overriding the BSR value. If 'a' = `1`, then the bank will be selected as per the BSR value.
Words:	1
Cycles:	1(2) **Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

<u>Example:</u>

```
HERE    DCFSNZ  TEMP
ZERO    :
NZERO   :
```

Before Instruction
TEMP = ?

After Instruction
TEMP = TEMP – 1,
If TEMP = 0;
PC = Address (`ZERO`)
If TEMP ≠ 0;
PC = Address (`NZERO`)

GOTO — Unconditional Branch

Syntax: [*label*] GOTO k

Operands: 0 ≤ k ≤ 1048575

Operation: k → PC<20:1>

Status Affected: None

Encoding:

1st word (k<7:0>)	1110	1111	$k_7$kkk	kkkk$_0$
2nd word(k<19:8>)	1111	k_{19}kkk	kkkk	kkkk$_8$

Description: `GOTO` allows an unconditional branch anywhere within entire 2-Mbyte memory range. The 20-bit value, 'k', is loaded into PC<20:1>. `GOTO` is always a two-cycle instruction.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'<7:0>,	No operation	Read literal 'k'<19:8>, Write to PC
No operation	No operation	No operation	No operation

Example: `GOTO THERE`

After Instruction
PC = Address (`THERE`)

INCF — Increment f

Syntax: [*label*] INCF f [,d [,a]]

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

Operation: (f) + 1 → dest

Status Affected: C, DC, N, OV, Z

Encoding:

0010	10da	ffff	ffff

Description: The contents of register, 'f', are incremented. If 'd' is '`0`', the result is placed in W. If 'd' is '`1`', the result is placed back in register, 'f'. If 'a' is '`0`', the Access Bank will be selected, over-riding the BSR value. If 'a' = `1`, then the bank will be selected as per the BSR value.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `INCF CNT,`

Before Instruction
CNT = 0xFF
Z = 0
C = ?
DC = ?

After Instruction
CNT = 0x00
Z = 1
C = 1
DC = 1

INCFSZ — Increment f, Skip if 0

Syntax:	[*label*] INCFSZ f [,d [,a]]
Operands:	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
Operation:	(f) + 1 → dest, skip if result = 0
Status Affected:	None
Encoding:	0011 | 11da | ffff | ffff
Description:	The contents of register, 'f', are incremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register, 'f'. If the result is '0', the next instruction, which is already fetched, is discarded, and a NOP is executed instead, making it a two-cycle instruction. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value.
Words:	1
Cycles:	1(2) **Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:

```
HERE     INCFSZ   CNT
NZERO    :
ZERO     :
```

Before Instruction
- PC = Address (HERE)

After Instruction
- CNT = CNT + 1
- If CNT = 0;
- PC = Address (ZERO)
- If CNT ≠ 0;
- PC = Address (NZERO)

INFSNZ — Increment f, Skip if Not 0

Syntax:	[*label*] INFSNZ f [,d [,a]]
Operands:	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
Operation:	(f) + 1 → dest, skip if result ≠ 0
Status Affected:	None
Encoding:	0100 | 10da | ffff | ffff
Description:	The contents of register, 'f', are incremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register, 'f'. If the result is not '0', the next instruction, which is already fetched, is discarded, and a NOP is executed instead, making it a two-cycle instruction. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value.
Words:	1
Cycles:	1(2) **Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:

```
HERE     INFSNZ   REG
ZERO
NZERO
```

Before Instruction
- PC = Address (HERE)

After Instruction
- REG = REG + 1
- If REG ≠ 0;
- PC = Address (NZERO)
- If REG = 0;
- PC = Address (ZERO)

IORLW **Inclusive OR Literal with W**

Syntax: [*label*] IORLW k

Operands: $0 \le k \le 255$

Operation: (W) .OR. k → W

Status Affected: N, Z

Encoding:

`0000`	`1001`	`kkkk`	`kkkk`

Description: The contents of W are ORed with the 8-bit literal, 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example: `IORLW 0x35`

Before Instruction
W = 0x9A

After Instruction
W = 0xBF

IORWF **Inclusive OR W with f**

Syntax: [*label*] IORWF f [,d [,a]]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (W) .OR. (f) → dest

Status Affected: N, Z

Encoding:

`0001`	`00da`	`ffff`	`ffff`

Description: Inclusive OR W with register, 'f'. If 'd' is '`0`', the result is placed in W. If 'd' is '`1`', the result is placed back in register, 'f'. If 'a' is '`0`', the Access Bank will be selected, overriding the BSR value. If 'a' = `1`, then the bank will be selected as per the BSR value.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `IORWF RESULT, W`

Before Instruction
RESULT = 0x13
W = 0x91

After Instruction
RESULT = 0x13
W = 0x93

LFSR — Load FSR

Syntax: [*label*] LFSR f,k

Operands: $0 \le f \le 2$
$0 \le k \le 4095$

Operation: $k \rightarrow FSRf$

Status Affected: None

Encoding:

1110	1110	00ff	k_{11}kkk
1111	0000	$k_7$kkk	kkkk

Description: The 12-bit literal 'k' is loaded into the file select register pointed to by 'f'.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k' MSB	Process Data	Write literal 'k' MSB to FSRfH
Decode	Read literal 'k' LSB	Process Data	Write literal 'k' to FSRfL

Example: `LFSR 2, 0x3AB`

After Instruction
FSR2H = 0x03
FSR2L = 0xAB

MOVF — Move f

Syntax: [*label*] MOVF f [,d [,a]]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: $f \rightarrow dest$

Status Affected: N, Z

Encoding:

0101	00da	ffff	ffff

Description: The contents of register, 'f', are moved to a destination dependent upon the status of 'd'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register, 'f'. Location, 'f', can be anywhere in the 256-byte bank. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write W

Example: `MOVF REG, W`

Before Instruction
REG = 0x22
W = 0xFF

After Instruction
REG = 0x22
W = 0x22

MOVFF — Move f to f

Syntax: [*label*] MOVFF f_s,f_d

Operands: $0 \le f_s \le 4095$
$0 \le f_d \le 4095$

Operation: $(f_s) \rightarrow f_d$

Status Affected: None

Encoding:

1st word (source)	`1100`	`ffff`	`ffff`	`ffff`$_s$
2nd word (destin.)	`1111`	`ffff`	`ffff`	`ffff`$_d$

Description: The contents of source register, 'f_s', are moved to destination register, 'f_d'. Location of source, 'f_s', can be anywhere in the 4096-byte data space (000h to FFFh) and location of destination, 'f_d', can also be anywhere from 000h to FFFh.
Either source or destination can be W (a useful special situation).
`MOVFF` is particularly useful for transferring a data memory location to a peripheral register (such as the transmit buffer or an I/O port).
The `MOVFF` instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register.
The `MOVFF` instruction should not be used to modify interrupt settings while any interrupt is enabled (see the note on page 97).

Words: 2

Cycles: 2 (3)

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f' (src)	Process Data	No operation
Decode	No operation No dummy read	No operation	Write register 'f' (dest)

Example: `MOVFF REG1, REG2`

Before Instruction
REG1 = 0x33
REG2 = 0x11

After Instruction
REG1 = 0x33
REG2 = 0x33

MOVLB — Move Literal to Low Nibble in BSR

Syntax: [*label*] MOVLB k

Operands: $0 \le k \le 255$

Operation: $k \rightarrow BSR$

Status Affected: None

Encoding:

`0000`	`0001`	`0000`	`kkkk`

Description: The 8-bit literal, 'k', is loaded into the Bank Select Register (BSR).

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write literal 'k' to BSR

Example: `MOVLB 5`

Before Instruction
BSR register = 0x02

After Instruction
BSR register = 0x05

MOVLW — Move Literal to W

Syntax:	[*label*] MOVLW k
Operands:	0 ≤ k ≤ 255
Operation:	k → W
Status Affected:	None
Encoding:	`0000` `1110` `kkkk` `kkkk`
Description:	The 8-bit literal, 'k', is loaded into W.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example: `MOVLW 0x5A`

After Instruction
W = 0x5A

MOVWF — Move W to f

Syntax:	[*label*] MOVWF f [,a]
Operands:	0 ≤ f ≤ 255 a ∈ [0,1]
Operation:	(W) → f
Status Affected:	None
Encoding:	`0110` `111a` `ffff` `ffff`
Description:	Move data from W to register, 'f'. Location, 'f', can be anywhere in the 256-byte bank. If 'a' is '`0`', the Access Bank will be selected, overriding the BSR value. If 'a' = `1`, then the bank will be selected as per the BSR value.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example: `MOVWF REG`

Before Instruction
W = 0x4F
REG = 0xFF
After Instruction
W = 0x4F
REG = 0x4F

MULLW — Multiply Literal with W

Syntax: [*label*] MULLW k

Operands: $0 \leq k \leq 255$

Operation: (W) x k → PRODH:PRODL

Status Affected: None

Encoding:

0000	1101	kkkk	kkkk

Description: An unsigned multiplication is carried out between the contents of W and the 8-bit literal, 'k'. The 16-bit result is placed in PRODH:PRODL register pair. PRODH contains the high byte. W is unchanged. None of the Status flags are affected. Note that neither Overflow nor Carry is possible in this operation. A Zero result is possible but not detected.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write registers PRODH: PRODL

Example: `MULLW 0xC4`

Before Instruction
- W = 0xE2
- PRODH = ?
- PRODL = ?

After Instruction
- W = 0xE2
- PRODH = 0xAD
- PRODL = 0x08

MULWF — Multiply W with f

Syntax: [*label*] MULWF f [,a]

Operands: $0 \leq f \leq 255$
$a \in [0,1]$

Operation: (W) x (f) → PRODH:PRODL

Status Affected: None

Encoding:

0000	001a	ffff	ffff

Description: An unsigned multiplication is carried out between the contents of W and the register file location, 'f'. The 16-bit result is stored in the PRODH:PRODL register pair. PRODH contains the high byte. Both W and 'f' are unchanged. None of the Status flags are affected. Note that neither Overflow nor Carry is possible in this operation. A Zero result is possible but not detected. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a'= 1, then the bank will be selected as per the BSR value.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write registers PRODH: PRODL

Example: `MULWF REG`

Before Instruction
- W = 0xC4
- REG = 0xB5
- PRODH = ?
- PRODL = ?

After Instruction
- W = 0xC4
- REG = 0xB5
- PRODH = 0x8A
- PRODL = 0x94

NEGF **Negate f**

Syntax:	[*label*] NEGF f [,a]
Operands:	$0 \leq f \leq 255$ $a \in [0,1]$
Operation:	$(\overline{f}) + 1 \rightarrow f$
Status Affected:	N, OV, C, DC, Z
Encoding:	`0110` | `110a` | `ffff` | `ffff`
Description:	Location, 'f', is negated using two's complement. The result is placed in the data memory location, 'f'. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example: `NEGF REG, 1`

Before Instruction
REG = `0011 1010` [0x3A]

After Instruction
REG = `1100 0110` [0xC6]

NOP **No Operation**

Syntax:	[*label*] NOP
Operands:	None
Operation:	No operation
Status Affected:	None
Encoding:	`0000` | `0000` | `0000` | `0000` `1111` | `xxxx` | `xxxx` | `xxxx`
Description:	No operation.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	No operation

Example:

None.

POP — Pop Top of Return Stack

Syntax:	[*label*] POP
Operands:	None
Operation:	(TOS) → bit bucket
Status Affected:	None
Encoding:	`0000` `0000` `0000` `0110`
Description:	The TOS value is pulled off the return stack and is discarded. The TOS value then becomes the previous value that was pushed onto the return stack. This instruction is provided to enable the user to properly manage the return stack to incorporate a software stack.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	POP TOS value	No operation

Example:

```
POP
GOTO    NEW
```

Before Instruction
- TOS = 0x0031A2
- Stack (1 level down) = 0x014332

After Instruction
- TOS = 0x014332
- PC = NEW

PUSH — Push Top of Return Stack

Syntax:	[*label*] PUSH
Operands:	None
Operation:	(PC + 2) → TOS
Status Affected:	None
Encoding:	`0000` `0000` `0000` `0101`
Description:	The PC + 2 is pushed onto the top of the return stack. The previous TOS value is pushed down on the stack. This instruction allows to implement a software stack by modifying TOS, and then push it onto the return stack.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	PUSH PC + 2 onto return stack	No operation	No operation

Example:

```
PUSH
```

Before Instruction
- TOS = 0x00345A
- PC = 0x000124

After Instruction
- PC = 0x000126
- TOS = 0x000126
- Stack (1 level down) = 0x00345A

RCALL — Relative Call

Syntax:	[*label*] RCALL n
Operands:	$-1024 \leq n \leq 1023$
Operation:	(PC) + 2 → TOS, (PC) + 2 + 2n → PC
Status Affected:	None
Encoding:	`1101` `1nnn` `nnnn` `nnnn`
Description:	Subroutine call with a jump up to 1K from the current location. First, return address (PC + 2) is pushed onto the stack. Then, add the 2's complement number '2n' to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is a two-cycle instruction.
Words:	1
Cycles:	2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'n' PUSH PC to stack	Process Data	Write to PC
No operation	No operation	No operation	No operation

Example: `HERE RCALL Jump`

Before Instruction
PC = Address (`HERE`)

After Instruction
PC = Address (`Jump`)
TOS = Address (`HERE + 2`)

RESET — Reset

Syntax:	[*label*] RESET
Operands:	None
Operation:	Reset all registers and flags that are affected by a $\overline{\text{MCLR}}$ Reset.
Status Affected:	All
Encoding:	`0000` `0000` `1111` `1111`
Description:	This instruction provides a way to execute a $\overline{\text{MCLR}}$ Reset in software.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Start Reset	No operation	No operation

Example: `RESET`

After Instruction
Registers = Reset Value
Flags* = Reset Value

RETFIE — Return from Interrupt

Syntax: [*label*] RETFIE [s]

Operands: s ∈ [0,1]

Operation: (TOS) → PC,
1 → GIE/GIEH or PEIE/GIEL;
if s = 1:
(WS) → W,
(STATUSS) → STATUS,
(BSRS) → BSR,
PCLATU, PCLATH are unchanged

Status Affected: GIE/GIEH, PEIE/GIEL.

Encoding:

0000	0000	0001	000s

Description: Return from interrupt. Stack is popped and Top-of-Stack (TOS) is loaded into the PC. Interrupts are enabled by setting either the high or low-priority global interrupt enable bit. If 's' = 1, the contents of the shadow registers, WS, STATUSS and BSRS, are loaded into their corresponding registers, W, STATUS and BSR. If 's' = 0, no update of these registers occurs.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	POP PC from stack Set GIEH or GIEL
No operation	No operation	No operation	No operation

Example: RETFIE 1

After Interrupt
- PC = TOS
- W = WS
- BSR = BSRS
- STATUS = STATUSS
- GIE/GIEH, PEIE/GIEL = 1

RETLW — Return Literal to W

Syntax: [*label*] RETLW k

Operands: 0 ≤ k ≤ 255

Operation: k → W,
(TOS) → PC,
PCLATU, PCLATH are unchanged

Status Affected: None

Encoding:

0000	1100	kkkk	kkkk

Description: W is loaded with the 8-bit literal, 'k'. The program counter is loaded from the top of the stack (the return address). The high address latch (PCLATH) remains unchanged.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	POP PC from stack, Write to W
No operation	No operation	No operation	No operation

Example:

```
    CALL TABLE ; W contains table
               ; offset value
               ; W now has
               ; table value
   :
TABLE
    ADDWF PCL  ; W = offset
    RETLW k0   ; Begin table
    RETLW k1   ;
   :
   :
    RETLW kn   ; End of table
```

Before Instruction
- W = 0x07

After Instruction
- W = value of kn

RETURN — Return from Subroutine

Syntax:	[*label*] RETURN [s]
Operands:	s ∈ [0,1]
Operation:	(TOS) → PC; if s = 1: (WS) → W, (STATUSS) → STATUS, (BSRS) → BSR, PCLATU, PCLATH are unchanged
Status Affected:	None

Encoding:

0000	0000	0001	001s

Description: Return from subroutine. The stack is popped and the top of the stack (TOS) is loaded into the program counter. If ‘s’= 1, the contents of the shadow registers, WS, STATUSS and BSRS, are loaded into their corresponding registers, W, STATUS and BSR. If ‘s’ = 0, no update of these registers occurs.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	POP PC from stack
No operation	No operation	No operation	No operation

Example: `RETURN`

After Interrupt
PC = TOS

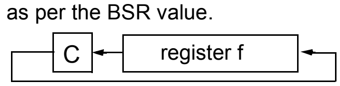
RLCF — Rotate Left f through Carry

Syntax:	[*label*] RLCF f [,d [,a]]
Operands:	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
Operation:	(f<n>) → dest<n + 1>, (f<7>) → C, (C) → dest<0>
Status Affected:	C, N, Z

Encoding:

0011	01da	ffff	ffff

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 W. If ‘d’ is ‘1’, the result is stored back in register, ‘f’. If ‘a’ is ‘0’, the Access Bank will be selected, overriding the BSR value. If ‘a’ = 1, then the bank will be selected as per the BSR value.



Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write to destination

Example: `RLCF REG, W`

Before Instruction
REG = 1110 0110
C = 0

After Instruction
REG = 1110 0110
W = 1100 1100
C = 1

RLNCF — Rotate Left f (No Carry)

Syntax: [*label*] RLNCF f [,d [,a]]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f<n>) → dest<n + 1>,
(f<7>) → dest<0>

Status Affected: N, Z

Encoding:

`0100`	`01da`	`ffff`	`ffff`

Description: The contents of register, ‘f’, are rotated one bit to the left. If ‘d’ is ‘0’, the result is placed in W. If ‘d’ is ‘1’, the result is stored back in register, ‘f’. If ‘a’ is ‘0’, the Access Bank will be selected, overriding the BSR value. If ‘a’ is ‘1’, then the bank will be selected as per the BSR value.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write to destination

Example: `RLNCF REG`

Before Instruction
REG = `1010 1011`

After Instruction
REG = `0101 0111`

RRCF — Rotate Right f through Carry

Syntax: [*label*] RRCF f [,d [,a]]

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f<n>) → dest<n – 1>,
(f<0>) → C,
(C) → dest<7>

Status Affected: C, N, Z

Encoding:

`0011`	`00da`	`ffff`	`ffff`

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 W. If ‘d’ is ‘1’, the result is placed back in register, ‘f’. If ‘a’ is ‘0’, the Access Bank will be selected, overriding the BSR value. If ‘a’ is ‘1’, then the bank will be selected as per the BSR value.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write to destination

Example: `RRCF REG, W`

Before Instruction
REG = `1110 0110`
C = `0`

After Instruction
REG = `1110 0110`
W = `0111 0011`
C = `0`

RRNCF — Rotate Right f (No Carry)

Syntax:	[*label*] RRNCF f [,d [,a]]
Operands:	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
Operation:	(f<n>) → dest<n – 1>, (f<0>) → dest<7>
Status Affected:	N, Z
Encoding:	`0100` `00da` `ffff` `ffff`
Description:	The contents of register, 'f', are rotated one bit to the right. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register, 'f'. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value.

Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example 1: `RRNCF REG, 1, 0`

Before Instruction
REG = 1101 0111
After Instruction
REG = 1110 1011

Example 2: `RRNCF REG, W`

Before Instruction
W = ?
REG = 1101 0111
After Instruction
W = 1110 1011
REG = 1101 0111

SETF — Set f

Syntax:	[*label*] SETF f [,a]
Operands:	0 ≤ f ≤ 255 a ∈ [0,1]
Operation:	FFh → f
Status Affected:	None
Encoding:	`0110` `100a` `ffff` `ffff`
Description:	The contents of the specified register are set to FFh. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example: `SETF REG`

Before Instruction
REG = 0x5A
After Instruction
REG = 0xFF

SLEEP — Enter Sleep Mode

Syntax:	[*label*] SLEEP
Operands:	None
Operation:	00h → WDT, 0 → WDT postscaler, 1 → $\overline{TO}$, 0 → $\overline{PD}$
Status Affected:	$\overline{TO}$, $\overline{PD}$
Encoding:	`0000` `0000` `0000` `0011`
Description:	The Power-Down status bit ($\overline{PD}$) is cleared. The Time-out status bit ($\overline{TO}$) is set. Watchdog Timer and its post-scaler are cleared. The processor is put into Sleep mode with the oscillator stopped.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	Go to Sleep

Example: `SLEEP`

Before Instruction
$\overline{TO}$ = ?
$\overline{PD}$ = ?

After Instruction
$\overline{TO}$ = 1 †
$\overline{PD}$ = 0

† If WDT causes wake-up, this bit is cleared.

SUBFWB — Subtract f from W with Borrow

Syntax:	[*label*] SUBFWB f [,d [,a]]
Operands:	$0 \le f \le 255$ $d \in [0,1]$ $a \in [0,1]$
Operation:	(W) – (f) – ($\overline{C}$) → dest
Status Affected:	N, OV, C, DC, Z
Encoding:	`0101` `01da` `ffff` `ffff`
Description:	Subtract register, 'f', and the Carry flag (borrow) from W (2's complement method). If 'd' is '`0`', the result is stored in W. If 'd' is '`1`', the result is stored in register, 'f'. If 'a' is '`0`', the Access Bank will be selected, overriding the BSR value. If 'a' is '`1`', then the bank will be selected as per the BSR value.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example 1: `SUBFWB REG`

Before Instruction
REG = 0x03
W = 0x02
C = 0x01

After Instruction
REG = 0xFF
W = 0x02
C = 0x00
Z = 0x00
N = 0x01 ; result is negative

Example 2: `SUBFWB REG, 0, 0`

Before Instruction
REG = 2
W = 5
C = 1

After Instruction
REG = 2
W = 3
C = 1
Z = 0
N = 0 ; result is positive

Example 3: `SUBFWB REG, 1, 0`

Before Instruction
REG = 1
W = 2
C = 0

After Instruction
REG = 0
W = 2
C = 1
Z = 1 ; result is zero
N = 0

SUBLW — Subtract W from Literal

Syntax:	[*label*] SUBLW k
Operands:	0 ≤ k ≤ 255
Operation:	k – (W) → W
Status Affected:	N, OV, C, DC, Z
Encoding:	`0000` `1000` `kkkk` `kkkk`
Description:	W is subtracted from the 8-bit literal, 'k'. The result is placed in W.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example 1: `SUBLW 0x02`

Before Instruction
W = 1
C = ?
After Instruction
W = 1
C = 1 ; result is positive
Z = 0
N = 0

Example 2: `SUBLW 0x02`

Before Instruction
W = 2
C = ?
After Instruction
W = 0
C = 1 ; result is zero
Z = 1
N = 0

Example 3: `SUBLW 0x02`

Before Instruction
W = 3
C = ?
After Instruction
W = FF ; (2's complement)
C = 0 ; result is negative
Z = 0
N = 1

SUBWF — Subtract W from f

Syntax:	[*label*] SUBWF f [,d [,a]]
Operands:	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
Operation:	(f) – (W) → dest
Status Affected:	N, OV, C, DC, Z
Encoding:	`0101` `11da` `ffff` `ffff`
Description:	Subtract W 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'. If = 'a' is '`0`', the Access Bank will be selected, overriding the BSR value. If 'a' is '`1`', then the bank will be selected as per the BSR value.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example 1: `SUBWF REG`

Before Instruction
REG = 3
W = 2
C = ?
After Instruction
REG = 1
W = 2
C = 1 ; result is positive
Z = 0
N = 0

Example 2: `SUBWF REG, W`

Before Instruction
REG = 2
W = 2
C = ?
After Instruction
REG = 2
W = 0
C = 1 ; result is zero
Z = 1
N = 0

Example 3: `SUBWF REG`

Before Instruction
REG = 0x01
W = 0x02
C = ?
After Instruction
REG = 0xFFh ; (2's complement)
W = 0x02
C = 0x00 ; result is negative
Z = 0x00
N = 0x01

SUBWFB — Subtract W from f with Borrow

Syntax:	[*label*] SUBWFB f [,d [,a]]
Operands:	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
Operation:	(f) – (W) – ($\overline{C}$) → dest
Status Affected:	N, OV, C, DC, Z
Encoding:	0101 | 10da | ffff | ffff
Description:	Subtract W and the Carry flag (borrow) 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'. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example 1: `SUBWFB REG, 1, 0`

Before Instruction
REG = 0x19 (0001 1001)
W = 0x0D (0000 1101)
C = 0x01
After Instruction
REG = 0x0C (0000 1011)
W = 0x0D (0000 1101)
C = 0x01
Z = 0x00
N = 0x00 ; result is positive

Example 2: `SUBWFB REG, 0, 0`

Before Instruction
REG = 0x1B (0001 1011)
W = 0x1A (0001 1010)
C = 0x00
After Instruction
REG = 0x1B (0001 1011)
W = 0x00
C = 0x01
Z = 0x01 ; result is zero
N = 0x00

Example 3: `SUBWFB REG, 1, 0`

Before Instruction
REG = 0x03 (0000 0011)
W = 0x0E (0000 1101)
C = 0x01
After Instruction
REG = 0xF5 (1111 0100)
; [2's comp]
W = 0x0E (0000 1101)
C = 0x00
Z = 0x00
N = 0x01 ; result is negative

SWAPF — Swap f

Syntax:	[*label*] SWAPF f [,d [,a]]
Operands:	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
Operation:	(f<3:0>) → dest<7:4>, (f<7:4>) → dest<3:0>
Status Affected:	None
Encoding:	0011 | 10da | ffff | ffff
Description:	The upper and lower nibbles of register, 'f', are exchanged. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed in register, 'f'. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `SWAPF REG`

Before Instruction
REG = 0x53
After Instruction
REG = 0x35

TBLRD — Table Read

Syntax:	[*label*] TBLRD (*; *+; *-; +*)
Operands:	None
Operation:	if TBLRD *, (Prog Mem (TBLPTR)) → TABLAT, TBLPTR – No Change; if TBLRD *+, (Prog Mem (TBLPTR)) → TABLAT, (TBLPTR) + 1 → TBLPTR; if TBLRD *-, (Prog Mem (TBLPTR)) → TABLAT, (TBLPTR) – 1 → TBLPTR; if TBLRD +*, (TBLPTR) + 1 → TBLPTR, (Prog Mem (TBLPTR)) → TABLAT
Status Affected:	None

Encoding:

0000	0000	0000	10nn nn = 0 * =1 *+ =2 *- =3 +*

Description: This instruction is used to read the contents of Program Memory (P.M.). To address the program memory, a pointer called Table Pointer (TBLPTR) is used.

The TBLPTR (a 21-bit pointer) points to each byte in the program memory. TBLPTR has a 2-Mbyte address range.

TBLPTR[0] = 0: Least Significant Byte of Program Memory Word

TBLPTR[0] = 1: Most Significant Byte of Program Memory Word

The TBLRD instruction can modify the value of TBLPTR as follows:

- no change
- post-increment
- post-decrement
- pre-increment

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	No operation
No operation	No operation (Read Program Memory)	No operation	No operation (Write TABLAT)

TBLRD — Table Read (cont'd)

Example 1: `TBLRD *+ ;`

Before Instruction

TABLAT	=	0x55
TBLPTR	=	0x00A356
MEMORY(0x00A356)	=	0x34

After Instruction

TABLAT	=	0x34
TBLPTR	=	0x00A357

Example 2: `TBLRD +* ;`

Before Instruction

TABLAT	=	0xAA
TBLPTR	=	0x01A357
MEMORY(0x01A357)	=	0x12
MEMORY(0x01A358)	=	0x34

After Instruction

TABLAT	=	0x34
TBLPTR	=	0x01A358

TBLWT Table Write

Syntax:	[*label*] TBLWT (*; *+; *-; +*)
Operands:	None
Operation:	if TBLWT*, (TABLAT) → Holding Register, TBLPTR – No Change; if TBLWT*+, (TABLAT) → Holding Register, (TBLPTR) + 1 → TBLPTR; if TBLWT*-, (TABLAT) → Holding Register, (TBLPTR) – 1 → TBLPTR; if TBLWT+*, (TBLPTR) + 1 → TBLPTR, (TABLAT) → Holding Register
Status Affected:	None

Encoding:

0000	0000	0000	11nn nn = 0 * =1 *+ =2 *- =3 +*

Description: This instruction uses the 3 LSBs of TBLPTR to determine which of the 8 holding registers the TABLAT is written to. The holding registers are used to program the contents of Program Memory (P.M.). (Refer to **Section 8.0 "Flash Program Memory"** for additional details on programming Flash memory.)

The TBLPTR (a 21-bit pointer) points to each byte in the program memory. TBLPTR has a 2-Mbyte address range. The LSb of the TBLPTR selects which byte of the program memory location to access.

TBLPTR[0] = `0`: Least Significant Byte of Program Memory Word

TBLPTR[0] = `1`: Most Significant Byte of Program Memory Word

The `TBLWT` instruction can modify the value of TBLPTR as follows:

- no change
- post-increment
- post-decrement
- pre-increment

TBLWT Table Write (Continued)

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	No operation
No operation	No operation (Read TABLAT)	No operation	No operation (Write to Holding Register)

Example 1: `TBLWT *+;`

Before Instruction

TABLAT	=	0x55
TBLPTR	=	0x00A356
HOLDING REGISTER (0x00A356)	=	0xFF

After Instructions (table write completion)

TABLAT	=	0x55
TBLPTR	=	0x00A357
HOLDING REGISTER (0x00A356)	=	0x55

Example 2: `TBLWT +*;`

Before Instruction

TABLAT	=	0x34
TBLPTR	=	0x01389A
HOLDING REGISTER (0x01389A)	=	0xFF
HOLDING REGISTER (0x01389B)	=	0xFF

After Instruction (table write completion)

TABLAT	=	0x34
TBLPTR	=	0x01389B
HOLDING REGISTER (0x01389A)	=	0xFF
HOLDING REGISTER (0x01389B)	=	0x34

TSTFSZ — Test f, Skip if 0

Syntax:	[*label*] TSTFSZ f [,a]
Operands:	0 ≤ f ≤ 255 a ∈ [0,1]
Operation:	skip if f = `0`
Status Affected:	None
Encoding:	`0110` `011a` `ffff` `ffff`
Description:	If 'f' = `0`, the next instruction, fetched during the current instruction execution, is discarded and a `NOP` is executed, making this a two-cycle instruction. If 'a' is '`0`', the Access Bank will be selected, overriding the BSR value. If 'a' is '`1`', then the bank will be selected as per the BSR value.
Words:	1
Cycles:	1(2) **Note:** 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:

```
HERE     TSTFSZ  CNT
NZERO      :
ZERO     :
```

Before Instruction
PC = Address (`HERE`)

After Instruction
If CNT = 0x00,
PC = Address (`ZERO`)
If CNT ≠ 0x00,
PC = Address (`NZERO`)

XORLW — Exclusive OR Literal with W

Syntax:	[*label*] XORLW k
Operands:	0 ≤ k ≤ 255
Operation:	(W) .XOR. k → W
Status Affected:	N, Z
Encoding:	`0000` `1010` `kkkk` `kkkk`
Description:	The contents of W are XORed with the 8-bit literal, 'k'. The result is placed in W.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example:

```
XORLW   0xAF
```

Before Instruction
W = 0xB5

After Instruction
W = 0x1A

XORWF — Exclusive OR W with f

Syntax:	[*label*] XORWF f [,d [,a]]
Operands:	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
Operation:	(W) .XOR. (f) → dest
Status Affected:	N, Z
Encoding:	`0001` `10da` `ffff` `ffff`
Description:	Exclusive OR the contents of W with register, 'f'. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in the register, 'f'. If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `XORWF REG`

Before Instruction
REG = 0xAF
W = 0xB5

After Instruction
REG = 0x1A
W = 0xB5

25.0 DEVELOPMENT SUPPORT

The PIC® microcontrollers and dsPIC® digital signal controllers are supported with a full range of software and hardware development tools:

- Integrated Development Environment
  - MPLAB® IDE Software
- Compilers/Assemblers/Linkers
  - MPLAB C Compiler for Various Device Families
  - HI-TECH C for Various Device Families
  - MPASM™ Assembler
  - MPLINK™ Object Linker/ MPLIB™ Object Librarian
  - MPLAB Assembler/Linker/Librarian for Various Device Families
- Simulators
  - MPLAB SIM Software Simulator
- Emulators
  - MPLAB REAL ICE™ In-Circuit Emulator
- In-Circuit Debuggers
  - MPLAB ICD 3
  - PICkit™ 3 Debug Express
- Device Programmers
  - PICkit™ 2 Programmer
  - MPLAB PM3 Device Programmer
- Low-Cost Demonstration/Development Boards, Evaluation Kits, and Starter Kits

25.1 MPLAB Integrated Development Environment Software

The MPLAB IDE software brings an ease of software development previously unseen in the 8/16/32-bit microcontroller market. The MPLAB IDE is a Windows® operating system-based application that contains:

- A single graphical interface to all debugging tools
  - Simulator
  - Programmer (sold separately)
  - In-Circuit Emulator (sold separately)
  - In-Circuit Debugger (sold separately)
- A full-featured editor with color-coded context
- A multiple project manager
- Customizable data windows with direct edit of contents
- High-level source code debugging
- Mouse over variable inspection
- Drag and drop variables from source to watch windows
- Extensive on-line help
- Integration of select third party tools, such as IAR C Compilers

The MPLAB IDE allows you to:

- Edit your source files (either C or assembly)
- One-touch compile or assemble, and download to emulator and simulator tools (automatically updates all project information)
- Debug using:
  - Source files (C or assembly)
  - Mixed C and assembly
  - Machine code

MPLAB IDE supports multiple debugging tools in a single development paradigm, from the cost-effective simulators, through low-cost in-circuit debuggers, to full-featured emulators. This eliminates the learning curve when upgrading to tools with increased flexibility and power.

25.2 MPLAB C Compilers for Various Device Families

The MPLAB C Compiler code development systems are complete ANSI C compilers for Microchip's PIC18, PIC24 and PIC32 families of microcontrollers and the dsPIC30 and dsPIC33 families of digital signal controllers. These compilers provide powerful integration capabilities, superior code optimization and ease of use.

For easy source level debugging, the compilers provide symbol information that is optimized to the MPLAB IDE debugger.

25.3 HI-TECH C for Various Device Families

The HI-TECH C Compiler code development systems are complete ANSI C compilers for Microchip's PIC family of microcontrollers and the dsPIC family of digital signal controllers. These compilers provide powerful integration capabilities, omniscient code generation and ease of use.

For easy source level debugging, the compilers provide symbol information that is optimized to the MPLAB IDE debugger.

The compilers include a macro assembler, linker, preprocessor, and one-step driver, and can run on multiple platforms.

25.4 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 IDE projects
- User-defined macros to streamline assembly code
- Conditional assembly for multi-purpose source files
- Directives that allow complete control over the assembly process

25.5 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK Object Linker combines relocatable objects created by the MPASM Assembler and the MPLAB C18 C Compiler. It can link relocatable objects from precompiled libraries, using directives from a linker script.

The MPLIB Object Librarian manages the creation and modification of library files of precompiled code. When a routine from a library is called from a source file, only the modules that contain that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications.

The object linker/library features include:

- Efficient linking of single libraries instead of many smaller files
- Enhanced code maintainability by grouping related modules together
- Flexible creation of libraries with easy module listing, replacement, deletion and extraction

25.6 MPLAB Assembler, Linker and Librarian for Various Device Families

MPLAB Assembler produces relocatable machine code from symbolic assembly language for PIC24, PIC32 and dsPIC devices. MPLAB C 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 IDE compatibility

25.7 MPLAB SIM Software Simulator

The MPLAB 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 SIM Software Simulator fully supports symbolic debugging using the MPLAB C 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.

25.8 MPLAB REAL ICE In-Circuit Emulator System

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 PIC® Flash MCUs and dsPIC® Flash DSCs with the easy-to-use, powerful graphical user interface of the MPLAB Integrated Development Environment (IDE), included with each kit.

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 (RJ11) or with the new high-speed, noise tolerant, Low-Voltage Differential Signal (LVDS) interconnection (CAT5).

The emulator is field upgradable through future firmware downloads in MPLAB IDE. In upcoming releases of MPLAB IDE, new devices will be supported, and new features will be added. MPLAB REAL ICE offers significant advantages over competitive emulators including low-cost, full-speed emulation, run-time variable watches, trace analysis, complex breakpoints, a ruggedized probe interface and long (up to three meters) interconnection cables.

25.9 MPLAB ICD 3 In-Circuit Debugger System

MPLAB ICD 3 In-Circuit Debugger System is Microchip's most cost effective high-speed hardware debugger/programmer for Microchip Flash Digital Signal Controller (DSC) and microcontroller (MCU) devices. It debugs and programs PIC® Flash microcontrollers and dsPIC® DSCs with the powerful, yet easy-to-use graphical user interface of MPLAB Integrated Development Environment (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.

25.10 PICkit 3 In-Circuit Debugger/Programmer and PICkit 3 Debug Express

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 Integrated Development Environment (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 an 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™.

The PICkit 3 Debug Express include the PICkit 3, demo board and microcontroller, hookup cables and CDROM with user's guide, lessons, tutorial, compiler and MPLAB IDE software.

25.11 PICkit 2 Development Programmer/Debugger and PICkit 2 Debug Express

The PICkit™ 2 Development Programmer/Debugger is a low-cost development tool with an easy to use interface for programming and debugging Microchip's Flash families of microcontrollers. The full featured Windows® programming interface supports baseline (PIC10F, PIC12F5xx, PIC16F5xx), midrange (PIC12F6xx, PIC16F), PIC18F, PIC24, dsPIC30, dsPIC33, and PIC32 families of 8-bit, 16-bit, and 32-bit microcontrollers, and many Microchip Serial EEPROM products. With Microchip's powerful MPLAB Integrated Development Environment (IDE) the PICkit™ 2 enables in-circuit debugging on most PIC® microcontrollers. In-Circuit-Debugging runs, halts and single steps the program while the PIC microcontroller is embedded in the application. When halted at a breakpoint, the file registers can be examined and modified.

The PICkit 2 Debug Express include the PICkit 2, demo board and microcontroller, hookup cables and CDROM with user's guide, lessons, tutorial, compiler and MPLAB IDE software.

25.12 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.

25.13 Demonstration/Development Boards, Evaluation Kits, and Starter Kits

A wide variety of demonstration, development and evaluation boards for various PIC MCUs and dsPIC DSCs allows quick application development on fully functional systems. Most boards include prototyping areas for adding custom circuitry and provide application firmware and source code for examination and modification.

The boards support a variety of features, including LEDs, temperature sensors, switches, speakers, RS-232 interfaces, LCD displays, potentiometers and additional EEPROM memory.

The demonstration and development boards can be used in teaching environments, for prototyping custom circuits and for learning about various microcontroller applications.

In addition to the PICDEM™ and dsPICDEM™ demonstration/development board series of circuits, Microchip has a line of evaluation kits and demonstration software for analog filter design, KEELOQ® security ICs, CAN, IrDA®, PowerSmart battery management, SEEVAL® evaluation system, Sigma-Delta ADC, flow rate sensing, plus many more.

Also available are starter kits that contain everything needed to experience the specified device. This usually includes a single application and debug capability, all on one board.

Check the Microchip web page (www.microchip.com) for the complete list of demonstration, development and evaluation kits.

26.0 ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings(†)

Ambient temperature under bias .... -55°C to +125°C
Storage temperature .... -65°C to +150°C
Voltage on any pin with respect to V_{SS} (except V_{DD}, $\overline{MCLR}$, and RA4) .... -0.3V to (V_{DD} + 0.3V)
Voltage on V_{DD} with respect to V_{SS} .... -0.3V to +7.5V
Voltage on $\overline{MCLR}$ with respect to V_{SS} **(Note 2)** .... 0V to +13.25V
Voltage on RA4 with respect to Vss .... 0V to +8.5V
Total power dissipation **(Note 1)** 1.0W
Maximum current out of V_{SS} pin .... 300 mA
Maximum current into V_{DD} pin .... 250 mA
Input clamp current, I_{IK} ($V_I < 0$ or $V_I > V_{DD}$) .... ±20 mA
Output clamp current, I_{OK} ($V_O < 0$ or $V_O > V_{DD}$) .... ±20 mA
Maximum output current sunk by any I/O pin .... 25 mA
Maximum output current sourced by any I/O pin .... 25 mA
Maximum current sunk by all ports .... 200 mA
Maximum current sourced by all ports .... 200 mA

Note 1: Power dissipation is calculated as follows:
$P_{dis} = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD} - V_{OH}) \times I_{OH}\} + \sum(V_{OL} \times I_{OL})$

2: Voltage spikes below V_{SS} at the $\overline{MCLR}/V_{PP}$ pin, inducing currents greater than 80 mA, may cause latch-up. Thus, a series resistor of 50-100Ω should be used when applying a "low" level to the $\overline{MCLR}/V_{PP}$ pin, rather than pulling this pin directly to V_{SS}.

† NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

FIGURE 26-1: PIC18F2331/2431/4331/4431 VOLTAGE-FREQUENCY GRAPH (INDUSTRIAL)

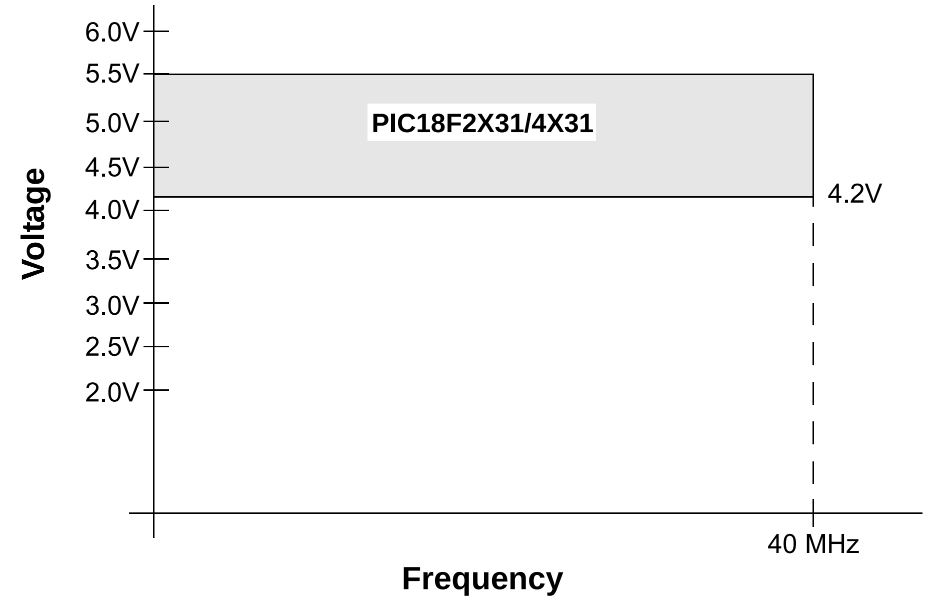
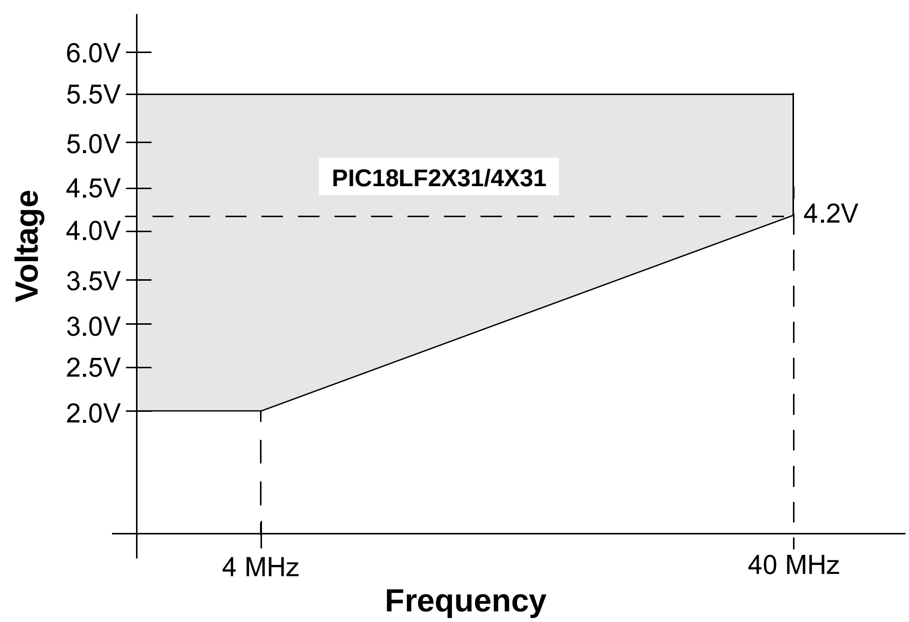


FIGURE 26-2: PIC18LF2331/2431/4331/4431 VOLTAGE-FREQUENCY GRAPH (INDUSTRIAL)



F_{MAX} = (16.36 MHz/V) ($V_{DDAPPMIN}$ – 2.0V) + 4 MHz

Note: $V_{DDAPPMIN}$ is the minimum voltage of the PIC® device in the application.

26.1 DC Characteristics: Supply Voltage PIC18F2331/2431/4331/4431 (Industrial, Extended) PIC18LF2331/2431/4331/4431 (Industrial)

PIC18LF2331/2431/4331/4431 (Industrial)			**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial				
PIC18F2331/2431/4331/4431 (Industrial, Extended)			**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended				
Param No.	**Symbol**	**Characteristic**	**Min**	**Typ**	**Max**	**Units**	**Conditions**
D001	VDD	**Supply Voltage**					
		PIC18LF2X31/4X31	2.0	—	5.5	V	
		PIC18F2X31/4X31	4.2	—	5.5	V	
D001C	AVDD	**Analog Supply Voltage**	VDD – 0.3	—	VDD + 0.3	V	
D001D	AVSS	**Analog Ground Voltage**	VSS – 0.3	—	VSS + 0.3	V	
D002	VDR	**RAM Data Retention Voltage**(1)	1.5	—	—	V	
D003	VPOR	**VDD Start Voltage** to Ensure Internal Power-on Reset Signal	—	—	0.7	V	See section on Power-on Reset for details
D004	SVDD	**VDD Rise Rate** to Ensure Internal Power-on Reset Signal	0.05	—	—	V/ms	See section on Power-on Reset for details
D005A	VBOR	**Brown-out Reset Voltage**					
		PIC18LF2X31/4X31	Industrial Low Voltage (-10°C to +85°C)				
		BORV<1:0> = 11	N/A	N/A	N/A	V	Reserved
		BORV<1:0> = 10	2.50	2.72	2.94	V	
		BORV<1:0> = 01	3.88	4.22	4.56	V	
		BORV<1:0> = 00	4.18	4.54	4.90	V	
D005B		PIC18LF2X31/4X31	Industrial Low Voltage (-40°C to -10°C)				
		BORV<1:0> = 11	N/A	N/A	N/A	V	Reserved
		BORV<1:0>= 10	2.34	2.72	3.10	V	
		BORV<1:0> = 01	3.63	4.22	4.81	V	
		BORV<1:0> = 00	3.90	4.54	5.18	V	
D005C		PIC18F2X31/4X31	Industrial (-10°C to +85°C)				
		BORV<1:0>= 1x	N/A	N/A	N/A	V	Reserved
		BORV<1:0> = 01	3.88	4.22	4.56	V	**(Note 2)**
		BORV<1:0> = 00	4.18	4.54	4.90	V	**(Note 2)**
D005D		PIC18F2X31/4X31	Industrial (-40°C to -10°C)				
		BORV<1:0>= 1x	N/A	N/A	N/A	V	Reserved
		BORV<1:0> = 01	N/A	N/A	N/A	V	Reserved
		BORV<1:0> = 00	3.90	4.54	5.18	V	**(Note 2)**
D005E		PIC18F2X31/4X31	Extended (-10°C to +85°C)				
		BORV<1:0> = 1x	N/A	N/A	N/A	V	Reserved
		BORV<1:0> = 01	3.88	4.22	4.56	V	**(Note 2)**
		BORV<1:0> = 00	4.18	4.54	4.90	V	**(Note 2)**
D005F		PIC18F2X31/4X31	Extended (-40°C to -10°C, +85°C to +125°C)				
		BORV<1:0> = 1x	N/A	N/A	N/A	V	Reserved
		BORV<1:0> = 01	N/A	N/A	N/A	V	Reserved
		BORV<1:0> = 00	3.90	4.54	5.18	V	**(Note 2)**

Legend: Shading of rows is to assist in readability of the table.

Note 1: This is the limit to which VDD can be lowered in Sleep mode, or during a device Reset, without losing RAM data.

2: When BOR is on and BORV<1:0> = 0x, the device will operate correctly at 40 MHz for any VDD at which the BOR allows execution.

26.2 DC Characteristics: Power-Down and Supply Current PIC18F2331/2431/4331/4431 (Industrial, Extended) PIC18LF2331/2431/4331/4431 (Industrial)

PIC18LF2331/2431/4331/4431 (Industrial)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial
PIC18F2331/2431/4331/4431 (Industrial, Extended)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended

Param No.	Device	Typ	Max	Units	Conditions	
	Power-Down Current (IPD)[1]					
	PIC18LF2X31/4X31	0.1	0.5	μA	-40°C	VDD = 2.0V (**Sleep** mode)
		0.1	0.5	μA	+25°C	
		0.2	1.9	μA	+85°C	
	PIC18LF2X31/4X31	0.1	0.5	μA	-40°C	VDD = 3.0V (**Sleep** mode)
		0.1	0.5	μA	+25°C	
		0.3	1.9	μA	+85°C	
	All devices	0.1	2.0	μA	-40°C	VDD = 5.0V (**Sleep** mode)
		0.1	2.0	μA	+25°C	
		0.4	6.5	μA	+85°C	
		5	33	μA	+125°C	

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS, and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
MCLR = VDD; WDT enabled/disabled as specified.

3: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be estimated by the formula: Ir = VDD/2REXT (mA) with REXT in kΩ.

4: Standard, low-cost 32 kHz crystals have an operating temperature range of -10°C to +70°C. Extended temperature crystals are available at a much higher cost.

26.2 DC Characteristics: Power-Down and Supply Current PIC18F2331/2431/4331/4431 (Industrial, Extended) PIC18LF2331/2431/4331/4431 (Industrial) (Continued)

PIC18LF2331/2431/4331/4431 (Industrial)					**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial		
PIC18F2331/2431/4331/4431 (Industrial, Extended)					**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended		
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**		
	Supply Current (IDD)[2,3]						
	PIC18LF2X31/4X31	8	40	μA	-40°C	VDD = 2.0V	FOSC = 31 kHz (**RC_RUN** mode, Internal oscillator source)
		9	40	μA	+25°C		
		11	40	μA	+85°C		
	PIC18LF2X31/4X31	25	68	μA	-40°C	VDD = 3.0V	
		25	68	μA	+25°C		
		20	68	μA	+85°C		
	All devices	55	180	μA	-40°C	VDD = 5.0V	
		55	180	μA	+25°C		
		50	180	μA	+85°C		
		0.25	1	mA	+125°C		
	PIC18LF2X31/4X31	140	220	μA	-40°C	VDD = 2.0V	FOSC = 1 MHz (**RC_RUN** mode, Internal oscillator source)
		145	220	μA	+25°C		
		155	220	μA	+85°C		
	PIC18LF2X31/4X31	215	330	μA	-40°C	VDD = 3.0V	
		225	330	μA	+25°C		
		235	330	μA	+85°C		
	All devices	385	550	μA	-40°C	VDD = 5.0V	
		390	550	μA	+25°C		
		405	550	μA	+85°C		
		0.7	2.8	mA	+125°C		
	PIC18LF2X31/4X31	410	600	μA	-40°C	VDD = 2.0V	FOSC = 4 MHz (**RC_RUN** mode, Internal oscillator source)
		425	600	μA	+25°C		
		435	600	μA	+85°C		
	PIC18LF2X31/4X31	650	900	μA	-40°C	VDD = 3.0V	
		670	900	μA	+25°C		
		680	900	μA	+85°C		
	All devices	1.2	1.8	mA	-40°C	VDD = 5.0V	
		1.2	1.8	mA	+25°C		
		1.2	1.8	mA	+85°C		
		2.2	6	mA	+125°C		

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS, and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
MCLR = VDD; WDT enabled/disabled as specified.

3: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be estimated by the formula: Ir = VDD/2REXT (mA) with REXT in kΩ.

4: Standard, low-cost 32 kHz crystals have an operating temperature range of -10°C to +70°C. Extended temperature crystals are available at a much higher cost.

26.2 DC Characteristics: Power-Down and Supply Current PIC18F2331/2431/4331/4431 (Industrial, Extended) PIC18LF2331/2431/4331/4431 (Industrial) (Continued)

PIC18LF2331/2431/4331/4431 (Industrial)					**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial		
PIC18F2331/2431/4331/4431 (Industrial, Extended)					**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial; -40°C ≤ TA ≤ +125°C for extended		
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**		
	Supply Current (IDD)(2,3)						
	PIC18LF2X31/4X31	4.7	8	μA	-40°C	VDD = 2.0V	FOSC = 31 kHz (**RC_IDLE** mode, Internal oscillator source)
		5.0	8	μA	+25°C		
		5.8	11	μA	+85°C		
	PIC18LF2X31/4X31	7.0	11	μA	-40°C	VDD = 3.0V	
		7.8	11	μA	+25°C		
		8.7	15	μA	+85°C		
	All devices	12	16	μA	-40°C	VDD = 5.0V	
		14	16	μA	+25°C		
		14	22	μA	+85°C		
		200	850	μA	+125°C		
	PIC18LF2X31/4X31	75	150	μA	-40°C	VDD = 2.0V	FOSC = 1 MHz (**RC_IDLE** mode, Internal oscillator source)
		85	150	μA	+25°C		
		95	150	μA	+85°C		
	PIC18LF2X31/4X31	110	180	μA	-40°C	VDD = 3.0V	
		125	180	μA	+25°C		
		135	180	μA	+85°C		
	All devices	180	300	μA	-40°C	VDD = 5.0V	
		195	300	μA	+25°C		
		200	300	μA	+85°C		
		300	750	μA	+125°C		
	PIC18LF2X31/4X31	175	275	μA	-40°C	VDD = 2.0V	FOSC = 4 MHz (**RC_IDLE** mode, Internal oscillator source)
		185	275	μA	+25°C		
		195	275	μA	+85°C		
	PIC18LF2X31/4X31	265	375	μA	-40°C	VDD = 3.0V	
		280	375	μA	+25°C		
		300	375	μA	+85°C		
	All devices	475	800	μA	-40°C	VDD = 5.0V	
		500	800	μA	+25°C		
		505	800	μA	+85°C		
		0.7	1.6	mA	+125°C		

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS, and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
MCLR = VDD; WDT enabled/disabled as specified.

3: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be estimated by the formula: Ir = VDD/2REXT (mA) with REXT in kΩ.

4: Standard, low-cost 32 kHz crystals have an operating temperature range of -10°C to +70°C. Extended temperature crystals are available at a much higher cost.

26.2 DC Characteristics: Power-Down and Supply Current PIC18F2331/2431/4331/4431 (Industrial, Extended) PIC18LF2331/2431/4331/4431 (Industrial) (Continued)

PIC18LF2331/2431/4331/4431 (Industrial)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial
PIC18F2331/2431/4331/4431 (Industrial, Extended)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended

Param No.	Device	Typ	Max	Units	Conditions		
	Supply Current (IDD)(2,3)						
	PIC18LF2X31/4X31	150	250	μA	-40°C	VDD = 2.0V	FOSC = 1 MHz (**PRI_RUN**, EC oscillator)
		150	250	μA	+25°C		
		160	250	μA	+85°C		
	PIC18LF2X31/4X31	340	350	μA	-40°C	VDD = 3.0V	
		300	350	μA	+25°C		
		280	350	μA	+85°C		
	All devices	0.72	1.0	mA	-40°C	VDD = 5.0V	
		0.63	1.0	mA	+25°C		
		0.57	1.0	mA	+85°C		
		0.9	2.1	mA	+125°C		
	PIC18LF2X31/4X31	440	600	μA	-40°C	VDD = 2.0V	FOSC = 4 MHz (**PRI_RUN**, EC oscillator)
		450	600	μA	+25°C		
		460	600	μA	+85°C		
	PIC18LF2X31/4X31	0.80	1.0	mA	-40°C	VDD = 3.0V	
		0.78	1.0	mA	+25°C		
		0.77	1.0	mA	+85°C		
	All devices	1.6	2.0	mA	-40°C	VDD = 5.0V	
		1.5	2.0	mA	+25°C		
		1.5	2.0	mA	+85°C		
		2.0	4.2	mA	+125°C		
	All devices	10	28	mA	+125°C	VDD = 5.0V	FOSC = 25 MHz (**PRI_RUN**, EC oscillator)
	All devices	9.5	12	mA	-40°C	VDD = 4.2V	FOSC = 40 MHz (**PRI_RUN**, EC oscillator)
		9.7	12	mA	+25°C		
		9.9	12	mA	+85°C		
	All devices	11.9	15	mA	-40°C	VDD = 5.0V	
		12.1	15	mA	+25°C		
		12.3	15	mA	+85°C		

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS, and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
MCLR = VDD; WDT enabled/disabled as specified.

3: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be estimated by the formula: Ir = VDD/2REXT (mA) with REXT in kΩ.

4: Standard, low-cost 32 kHz crystals have an operating temperature range of -10°C to +70°C. Extended temperature crystals are available at a much higher cost.

26.2 DC Characteristics: Power-Down and Supply Current PIC18F2331/2431/4331/4431 (Industrial, Extended) PIC18LF2331/2431/4331/4431 (Industrial) (Continued)

PIC18LF2331/2431/4331/4431 (Industrial)					**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial		
PIC18F2331/2431/4331/4431 (Industrial, Extended)					**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial; -40°C ≤ TA ≤ +125°C for extended		
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**		
	Supply Current (IDD)(2,3)						
	PIC18LF2X31/4X31	35	50	μA	-40°C	VDD = 2.0V	FOSC = 1 MHz (**PRI_IDLE** mode, EC oscillator)
		35	50	μA	+25°C		
		35	60	μA	+85°C		
	PIC18LF2X31/4X31	55	80	μA	-40°C	VDD = 3.0V	
		50	80	μA	+25°C		
		60	100	μA	+85°C		
	All devices	105	150	μA	-40°C	VDD = 5.0V	
		110	150	μA	+25°C		
		115	150	μA	+85°C		
		300	400	μA	+125°C		
	PIC18LF2X31/4X31	135	180	μA	-40°C	VDD = 2.0V	FOSC = 4 MHz (**PRI_IDLE** mode, EC oscillator)
		140	180	μA	+25°C		
		140	180	μA	+85°C		
	PIC18LF2X31/4X31	215	280	μA	-40°C	VDD = 3.0V	
		225	280	μA	+25°C		
		230	280	μA	+85°C		
	All devices	410	525	μA	-40°C	VDD = 5.0V	
		420	525	μA	+25°C		
		430	525	μA	+85°C		
		1.2	1.7	mA	+125°C		
	All devices	18	22	mA	+125°C	VDD = 5.0V	FOSC = 25 MHz (**PRI_IDLE** mode, EC oscillator)
	All devices	3.2	4.1	mA	-40°C	VDD = 4.2 V	FOSC = 40 MHz (**PRI_IDLE** mode, EC oscillator)
		3.2	4.1	mA	+25°C		
		3.3	4.1	mA	+85°C		
	All devices	4.0	5.1	mA	-40°C	VDD = 5.0V	
		4.1	5.1	mA	+25°C		
		4.1	5.1	mA	+85°C		

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS, and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
MCLR = VDD; WDT enabled/disabled as specified.

3: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be estimated by the formula: Ir = VDD/2REXT (mA) with REXT in kΩ.

4: Standard, low-cost 32 kHz crystals have an operating temperature range of -10°C to +70°C. Extended temperature crystals are available at a much higher cost.

26.2 DC Characteristics: Power-Down and Supply Current PIC18F2331/2431/4331/4431 (Industrial, Extended) PIC18LF2331/2431/4331/4431 (Industrial) (Continued)

PIC18LF2331/2431/4331/4431 (Industrial)			**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ T_A ≤ +85°C for industrial				
PIC18F2331/2431/4331/4431 (Industrial, Extended)			**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ T_A ≤ +85°C for industrial; -40°C ≤ T_A ≤ +125°C for extended				
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**		
	Supply Current (I_{DD})(2,3)						
	PIC18LF2X31/4X31	5.1	9	μA	-10°C	V_{DD} = 2.0V	F_{OSC} = 32 kHz(4) (**SEC_RUN** mode, Timer1 as clock)
		5.8	9	μA	+25°C	V_{DD} = 2.0V	
		7.9	11	μA	+70°C	V_{DD} = 2.0V	
	PIC18LF2X31/4X31	7.9	12	μA	-10°C	V_{DD} = 3.0V	
		8.9	12	μA	+25°C	V_{DD} = 3.0V	
		10.5	14	μA	+70°C	V_{DD} = 3.0V	
	All devices	12.5	20	μA	-10°C	V_{DD} = 5.0V	
	All devices	16.3	20	μA	+25°C	V_{DD} = 5.0V	
	All devices	18.9	25	μA	+70°C	V_{DD} = 5.0V	
	All devices	150	850	μA	+125°C	V_{DD} = 5.0V	
	PIC18LF2X31/4X31	9.2	15	μA	-10°C	V_{DD} = 2.0V	F_{OSC} = 32 kHz(4) (**SEC_IDLE** mode, Timer1 as clock)
		9.6	15	μA	+25°C	V_{DD} = 2.0V	
		12.7	18	μA	+70°C	V_{DD} = 2.0V	
	PIC18LF2X31/4X31	22.0	30	μA	-10°C	V_{DD} = 3.0V	
		21.0	30	μA	+25°C	V_{DD} = 3.0V	
		20.0	35	μA	+70°C	V_{DD} = 3.0V	
	All devices	30	80	μA	-10°C	V_{DD} = 5.0V	
	All devices	45	80	μA	+25°C	V_{DD} = 5.0V	
	All devices	45	85	μA	+70°C	V_{DD} = 5.0V	
	All devices	250	850	μA	+125°C	V_{DD} = 5.0V	

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to V_{DD} or V_{SS}, and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all I_{DD} measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to V_{DD};
MCLR = V_{DD}; WDT enabled/disabled as specified.

3: For RC oscillator configurations, current through R_{EXT} is not included. The current through the resistor can be estimated by the formula: Ir = $V_{DD}/2R_{EXT}$ (mA) with R_{EXT} in kΩ.

4: Standard, low-cost 32 kHz crystals have an operating temperature range of -10°C to +70°C. Extended temperature crystals are available at a much higher cost.

26.2 DC Characteristics: Power-Down and Supply Current PIC18F2331/2431/4331/4431 (Industrial, Extended) PIC18LF2331/2431/4331/4431 (Industrial) (Continued)

PIC18LF2331/2431/4331/4431 (Industrial)					Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial		
PIC18F2331/2431/4331/4431 (Industrial, Extended)					**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended		
Param No.	**Device**	**Typ**	**Max**	**Units**	**Conditions**		
	Module Differential Currents (ΔIWDT, ΔIBOR, ΔILVD, ΔIOSCB, ΔIAD)						
D022 (ΔIWDT)	**Watchdog Timer**	1.5	4.0	μA	-40°C	VDD = 2.0V	
		2.2	4.0	μA	+25°C		
		3.1	5.0	μA	+85°C		
		2.5	6.0	μA	-40°C	VDD = 3.0V	
		3.3	6.0	μA	+25°C		
		4.7	7.0	μA	+85°C		
		3.7	10.0	μA	-40°C	VDD = 5.0V	
		4.5	10.0	μA	+25°C		
		6.1	13.0	μA	+85°C		
		22	44	μA	+125°C		
D022A (ΔIBOR)	**Brown-out Reset**	19	35.0	μA	-40°C to +85°C	VDD = 3.0V	
		24	45.0	μA	-40°C to +85°C	VDD = 5.0V	
		40	75	μA	+125°C		
D022B (ΔILVD)	**Low-Voltage Detect**	8.5	25.0	μA	-40°C to +85°C	VDD = 2.0V	
		16	35.0	μA	-40°C to +85°C	VDD = 3.0V	
		20	45.0	μA	-40°C to +85°C	VDD = 5.0V	
		35	66	μA	+125°C		
D025 (ΔIOSCB)	**Timer1 Oscillator**	1.7	3.5	μA	-40°C	VDD = 2.0V	32 kHz on Timer1[4]
		1.8	3.5	μA	+25°C		
		2.1	4.5	μA	+85°C		
		2.2	4.5	μA	-40°C	VDD = 3.0V	32 kHz on Timer1[4]
		2.6	4.5	μA	+25°C		
		2.8	5.5	μA	+85°C		
		3.0	6.0	μA	-40°C	VDD = 5.0V	32 kHz on Timer1[4]
		3.3	6.0	μA	+25°C		
		3.6	7.0	μA	+85°C		
		42	70	μA	+125°C		
D026 (ΔIAD)	**A/D Converter**	1.0	3.0	μA	-40°C to +85°C	VDD = 2.0V	A/D on, not converting
		1.0	4.0	μA	-40°C to +85°C	VDD = 3.0V	
		2.0	10.0	μA	-40°C to +85°C	VDD = 5.0V	
		150	950	μA	+125°C		

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS, and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
MCLR = VDD; WDT enabled/disabled as specified.

3: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be estimated by the formula: Ir = VDD/2REXT (mA) with REXT in kΩ.

4: Standard, low-cost 32 kHz crystals have an operating temperature range of -10°C to +70°C. Extended temperature crystals are available at a much higher cost.

26.3 DC Characteristics: PIC18F2331/2431/4331/4431 (Industrial, Extended) PIC18LF2331/2431/4331/4431 (Industrial)

DC CHARACTERISTICS			Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended			
Param No.	**Symbol**	**Characteristic**	**Min**	**Max**	**Units**	**Conditions**
	VIL	**Input Low Voltage**				
		I/O Ports:				
D030		with TTL Buffer	VSS	0.15 VDD	V	VDD < 4.5V
D030A			—	0.8	V	4.5V ≤ VDD ≤ 5.5V
D031		with Schmitt Trigger Buffer RC3 and RC4	VSS VSS	0.2 VDD 0.3 VDD	V V	 I²C™ enabled
D032		MCLR	VSS	0.2 VDD	V	
D032A		OSC1 and T1OSI	VSS	0.3 VDD	V	LP, XT, HS, HSPLL modes(1)
D033		OSC1	VSS	0.2 VDD	V	EC mode(1)
	VIH	**Input High Voltage**				
		I/O Ports:				
D040		with TTL Buffer	0.25 VDD + 0.8V	VDD	V	VDD < 4.5V
D040A			2.0	VDD	V	4.5V ≤ VDD ≤ 5.5V
D041		with Schmitt Trigger Buffer RC3 and RC4	0.8 VDD 0.7 VDD	VDD VDD	V V	 I²C™ enabled
D042		MCLR	0.8 VDD	VDD	V	
D042A		OSC1 and T1OSI	0.7 VDD	VDD	V	LP, XT, HS, HSPLL modes(1)
D043		OSC1	0.8 VDD	VDD	V	EC mode(1)
	IIL	**Input Leakage Current(2,3)**				
D060		I/O Ports	—	+200 nA	μA	VDD < 5.5V, VSS ≤ VPIN ≤ VDD, Pin at high-impedance
			—	+50 nA		VDD < 3V, VSS ≤ VPIN ≤ VDD, Pin at high-impedance
D061		MCLR	—	±1	μA	Vss ≤ VPIN ≤ VDD
D063		OSC1	—	±1	μA	Vss ≤ VPIN ≤ VDD
	IPU	**Weak Pull-up Current**				
D070	IPURB	PORTB Weak Pull-up Current	50	400	μA	VDD = 5V, VPIN = VSS

Note 1: In RC oscillator configuration, the OSC1/CLKI pin is a Schmitt Trigger input. It is not recommended that the PIC® device be driven with an external clock while in RC mode.

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.

3: Negative current is defined as current sourced by the pin.

26.3 DC Characteristics: PIC18F2331/2431/4331/4431 (Industrial, Extended) PIC18LF2331/2431/4331/4431 (Industrial) (Continued)

DC CHARACTERISTICS			Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended			
Param No.	Symbol	Characteristic	Min	Max	Units	Conditions
	V_{OL}	**Output Low Voltage**				
D080		I/O Ports	—	0.6	V	I_{OL} = 8.5 mA, V_{DD} = 4.5V, -40°C to +85°C
D083		OSC2/CLKO (RC, RCIO, EC, ECIO modes)	—	0.6	V	I_{OL} = 1.6 mA, V_{DD} = 4.5V, -40°C to +85°C
	V_{OH}	**Output High Voltage**(3)				
D090		I/O Ports	V_{DD} – 0.7	—	V	I_{OH} = -3.0 mA, V_{DD} = 4.5V, -40°C to +85°C
D092		OSC2/CLKO (RC, RCIO, EC, ECIO modes)	V_{DD} – 0.7	—	V	I_{OH} = -1.3 mA, V_{DD} = 4.5V, -40°C to +85°C
		Capacitive Loading Specs on Output Pins				
D100	C_{OSC2}	OSC2 Pin	—	15	pF	In XT, HS and LP modes when external clock is used to drive OSC1
D101	C_{IO}	All I/O Pins and OSC2 (in RC mode)	—	50	pF	To meet the AC Timing Specifications
D102	C_B	SCL, SDA	—	400	pF	I^2C™ Specification

Note 1: In RC oscillator configuration, the OSC1/CLKI pin is a Schmitt Trigger input. It is not recommended that the PIC® device be driven with an external clock while in RC mode.

2: The leakage current on the $\overline{MCLR}$ pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

3: Negative current is defined as current sourced by the pin.

TABLE 26-1: MEMORY PROGRAMMING REQUIREMENTS

DC CHARACTERISTICS			**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended				
Param No.	**Sym**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
		Internal Program Memory Programming Specifications(1)					
D110	VPP	Voltage on $\overline{\text{MCLR}}$/VPP pin	9.00	—	13.25	V	**(Note 3)**
D112	IPP	Current into $\overline{\text{MCLR}}$/VPP pin	—	—	300	μA	
D113	IDDP	Supply Current during Programming	—	—	1	mA	
		Data EEPROM Memory					
D120	ED	Byte Endurance	100K	1M	—	E/W	-40°C to +85°C
D121	VDRW	VDD for Read/Write	VMIN	—	5.5	V	Using EECON to read/write VMIN = Minimum operating voltage
D122	TDEW	Erase/Write Cycle Time	—	4	—	ms	
D123	TRETD	Characteristic Retention	40	—	—	Year	Provided no other specifications are violated
D124	TREF	Number of Total Erase/Write Cycles before Refresh(2)	1M	10M	—	E/W	-40°C to +85°C
		Program Flash Memory					
D130	EP	Cell Endurance	10K	100K	—	E/W	-40°C to +85°C
D131	VPR	VDD for Read	VMIN	—	5.5	V	VMIN = Minimum operating voltage
D132	VIE	VDD for Block Erase	4.5	—	5.5	V	Using ICSP™ port
D132A	VIW	VDD for Externally Timed Erase or Write	4.5	—	5.5	V	Using ICSP port
D132B	VPEW	VDD for Self-Timed Write	VMIN	—	5.5	V	VMIN = Minimum operating voltage
D133	TIE	ICSP™ Block Erase Cycle Time	—	4	—	ms	VDD > 4.5V
D133A	TIW	ICSP Erase or Write Cycle Time (externally timed)	1	—	—	ms	VDD > 4.5V
D133A	TIW	Self-Timed Write Cycle Time	—	2	—	ms	
D134	TRETD	Characteristic Retention	40	100	—	Year	Provided no other specifications are violated

† Data in “Typ” column is at 5.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: These specifications are for programming the on-chip program memory through the use of table write instructions.

2: Refer to **Section 7.9 “Using the Data EEPROM”** for a more detailed discussion on data EEPROM endurance.

3: Required only if Single-Supply Programming is disabled.

FIGURE 26-3: LOW-VOLTAGE DETECT CHARACTERISTICS

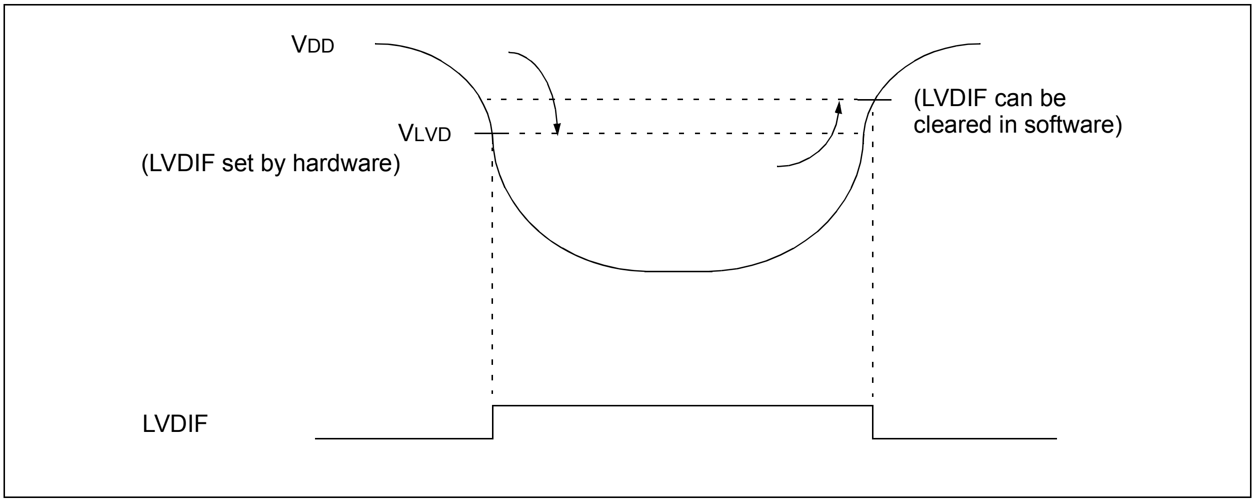


TABLE 26-2: LOW-VOLTAGE DETECT CHARACTERISTICS

PIC18LF2331/2431/4331/4431 (Industrial)				Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial				
PIC18F2331/2431/4331/4431 (Industrial, Extended)				**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended				
Param No.	**Symbol**	**Characteristic**		**Min**	**Typ†**	**Max**	**Units**	**Conditions**
D420A	VLVD	LVD Voltage on VDD Transition High-to-Low		Industrial Low Voltage (-10°C to +85°C)				
		PIC18LF2X31/4X31	LVDL<3:0> = `0000`	N/A	N/A	N/A	V	Reserved
			LVDL<3:0> = `0001`	N/A	N/A	N/A	V	Reserved
			LVDL<3:0> = `0010`	2.08	2.26	2.44	V	
			LVDL<3:0> = `0011`	2.26	2.45	2.65	V	
			LVDL<3:0> = `0100`	2.35	2.55	2.76	V	
			LVDL<3:0> = `0101`	2.55	2.77	2.99	V	
			LVDL<3:0> = `0110`	2.64	2.87	3.10	V	
			LVDL<3:0> = `0111`	2.82	3.07	3.31	V	
			LVDL<3:0> = `1000`	3.09	3.36	3.63	V	
			LVDL<3:0> = `1001`	3.29	3.57	3.86	V	
			LVDL<3:0> = `1010`	3.38	3.67	3.96	V	
			LVDL<3:0> = `1011`	3.56	3.87	4.18	V	
			LVDL<3:0> = `1100`	3.75	4.07	4.40	V	
			LVDL<3:0> = `1101`	3.93	4.28	4.62	V	
			LVDL<3:0> = `1110`	4.23	4.60	4.96	V	

Legend: Shading of rows is to assist in readability of the table.
† Production tested at TAMB = 25°C. Specifications over temperature limits ensured by characterization.

TABLE 26-2: LOW-VOLTAGE DETECT CHARACTERISTICS (CONTINUED)

PIC18LF2331/2431/4331/4431 (Industrial)				**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ T_A ≤ +85°C for industrial				
PIC18F2331/2431/4331/4431 (Industrial, Extended)				**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ T_A ≤ +85°C for industrial -40°C ≤ T_A ≤ +125°C for extended				
Param No.	**Symbol**	**Characteristic**		**Min**	**Typ†**	**Max**	**Units**	**Conditions**
D420B	V_{LVD}	LVD Voltage on V_{DD} Transition High-to-Low		Industrial Low Voltage (-40°C to -10°C)				
		PIC18LF2X31/4X31	LVDL<3:0> = `0000`	N/A	N/A	N/A	V	Reserved
			LVDL<3:0> = `0001`	N/A	N/A	N/A	V	Reserved
			LVDL<3:0> = `0010`	1.99	2.26	2.53	V	
			LVDL<3:0> = `0011`	2.16	2.45	2.75	V	
			LVDL<3:0> = `0100`	2.25	2.55	2.86	V	
			LVDL<3:0> = `0101`	2.43	2.77	3.10	V	
			LVDL<3:0> = `0110`	2.53	2.87	3.21	V	
			LVDL<3:0> = `0111`	2.70	3.07	3.43	V	
			LVDL<3:0> = `1000`	2.96	3.36	3.77	V	
			LVDL<3:0> = `1001`	3.14	3.57	4.00	V	
			LVDL<3:0> = `1010`	3.23	3.67	4.11	V	
			LVDL<3:0> = `1011`	3.41	3.87	4.34	V	
			LVDL<3:0> = `1100`	3.58	4.07	4.56	V	
			LVDL<3:0> = `1101`	3.76	4.28	4.79	V	
			LVDL<3:0> = `1110`	4.04	4.60	5.15	V	
D420C	V_{LVD}	LVD Voltage on V_{DD} Transition High-to-Low		Industrial (-10°C to +85°C)				
		PIC18F2X31/4X31	LVDL<3:0> = `1101`	3.93	4.28	4.62	V	
			LVDL<3:0> = `1110`	4.23	4.60	4.96	V	
D420D	V_{LVD}	LVD Voltage on V_{DD} Transition High-to-Low		Industrial (-40°C to -10°C)				
		PIC18F2X31/4X31	LVDL<3:0> = `1101`	3.76	4.28	4.79	V	Reserved
			LVDL<3:0> = `1110`	4.04	4.60	5.15	V	
D420E	V_{LVD}	LVD Voltage on V_{DD} Transition High-to-Low		Extended (-10°C to +85°C)				
		PIC18F2X31/4X31	LVDL<3:0> = `1101`	3.94	4.28	4.62	V	
			LVDL<3:0> = `1110`	4.23	4.60	4.96	V	
D420F	V_{LVD}	LVD Voltage on V_{DD} Transition High-to-Low		Extended (-40°C to -10°C, +85°C to +125°C)				
		PIC18F2X31/4X31	LVDL<3:0> = `1101`	3.77	4.28	4.79	V	Reserved
			LVDL<3:0> = `1110`	4.05	4.60	5.15	V	

Legend: Shading of rows is to assist in readability of the table.

† Production tested at T_{AMB} = 25°C. Specifications over temperature limits ensured by characterization.

26.4 AC (Timing) Characteristics

26.4.1 TIMING PARAMETER SYMBOLOGY

The timing parameter symbols have been created following one of the following formats:

1. TppS2ppS		3. TCC:ST	(I^2C specifications only)
2. TppS		4. Ts	(I^2C specifications only)
T			
F	Frequency	T	Time
Lowercase letters (pp) and their meanings:			
pp			
cc	CCP1	osc	OSC1
ck	CLKO	rd	$\overline{\text{RD}}$
cs	$\overline{\text{CS}}$	rw	$\overline{\text{RD}}$ or $\overline{\text{WR}}$
di	SDI	sc	SCK
do	SDO	ss	$\overline{\text{SS}}$
dt	Data in	t0	T0CKI
io	I/O port	t1	T1CKI
mc	$\overline{\text{MCLR}}$	wr	$\overline{\text{WR}}$
Uppercase letters and their meanings:			
S			
F	Fall	P	Period
H	High	R	Rise
I	Invalid (High-Impedance)	V	Valid
L	Low	Z	High-Impedance
I^2C only			
AA	output access	High	High
BUF	Bus free	Low	Low
TCC:ST (I^2C specifications only)			
CC			
HD	Hold	SU	Setup
ST			
DAT	DATA input hold	STO	Stop condition
STA	Start condition		

26.4.2 TIMING CONDITIONS

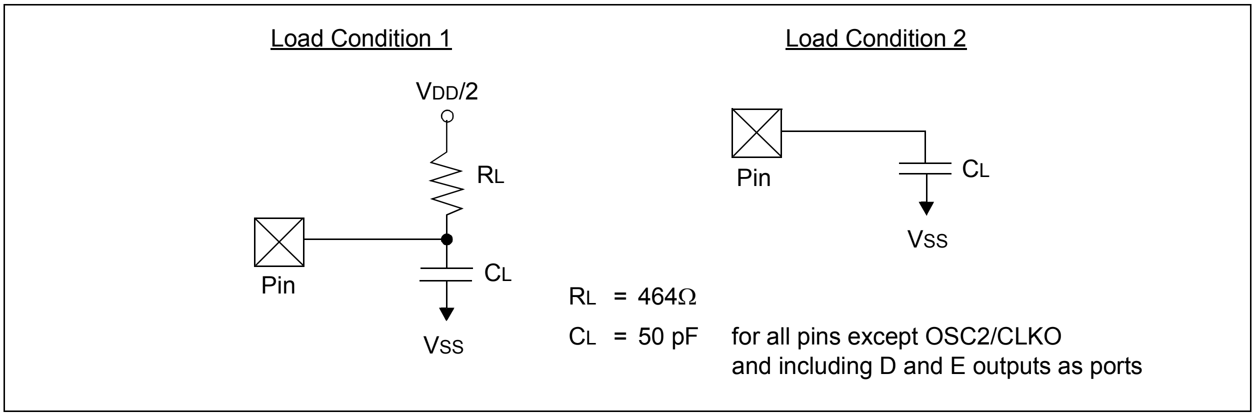
The temperature and voltages specified in Table 26-3 apply to all timing specifications unless otherwise noted. Figure 26-4 specifies the load conditions for the timing specifications.

Note: Because of space limitations, the generic terms "PIC18FXX31" and "PIC18LFXX31" are used throughout this section to refer to the PIC18F2331/2431/4331/4431 and PIC18LF2331/2431/4331/4431 families of devices specifically, and only those devices.

TABLE 26-3: TEMPERATURE AND VOLTAGE SPECIFICATIONS – AC

AC CHARACTERISTICS	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended Operating voltage VDD range as described in DC spec **Section 26.1** and **Section 26.3**. LF parts operate for industrial temperatures only.

FIGURE 26-4: LOAD CONDITIONS FOR DEVICE TIMING SPECIFICATIONS



26.4.3 TIMING DIAGRAMS AND SPECIFICATIONS

FIGURE 26-5: EXTERNAL CLOCK TIMING (ALL MODES EXCEPT PLL)

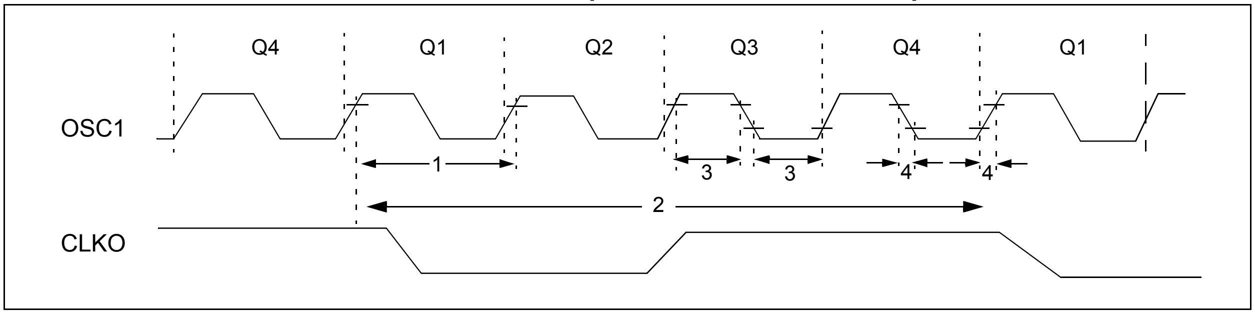


TABLE 26-4: EXTERNAL CLOCK TIMING REQUIREMENTS

Param. No.	Symbol	Characteristic	Min	Max	Units	Conditions
1A	FOSC	External CLKI Frequency(1)	DC	40	MHz	EC, ECIO
		Oscillator Frequency(1)	DC	4	MHz	RC osc
			0.1	4	MHz	XT osc
			4	25	MHz	HS osc
			4	10	MHz	HS + PLL osc
			5	200	kHz	LP Osc mode
1	TOSC	External CLKI Period(1)	25	—	ns	EC, ECIO
		Oscillator Period(1)	250	—	ns	RC osc
			250	10,000	ns	XT osc
			25	250	ns	HS osc
			100	250	ns	HS + PLL osc
			25	—	μs	LP osc
2	TCY	Instruction Cycle Time(1)	100	—	ns	TCY = 4/FOSC
3	TosL, TosH	External Clock in (OSC1) High or Low Time	30	—	ns	XT osc
			2.5	—	μs	LP osc
			10	—	ns	HS osc
4	TosR, TosF	External Clock in (OSC1) Rise or Fall Time	—	20	ns	XT osc
			—	50	ns	LP osc
			—	7.5	ns	HS osc

Note 1: Instruction cycle period (TCY) equals four times the input oscillator time base period for all configurations except PLL. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at “min.” values with an external clock applied to the OSC1/CLKI pin. When an external clock input is used, the “max.” cycle time limit is “DC” (no clock) for all devices.

TABLE 26-5: PLL CLOCK TIMING SPECIFICATIONS (V_{DD} = 4.2V TO 5.5V)

Param No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
F10	FOSC	Oscillator Frequency Range	4	—	10	MHz	HS mode only
F11	FSYS	On-Chip VCO System Frequency	16	—	40	MHz	HS mode only
F12	TPLL	PLL Start-up Time (Lock Time)	—	—	2	ms	
F13	ΔCLK	CLKO Stability (Jitter)	-2	—	+2	%	

† Data in “Typ” column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

TABLE 26-6: INTERNAL RC ACCURACY

PIC18LF2331/2431/4331/4431 (Industrial)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial
PIC18F2331/2431/4331/4431 (Industrial)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended

Param No.	Device	Min	Typ	Max	Units	Conditions	
	INTOSC Accuracy @ Freq = 8 MHz, 4 MHz, 2 MHz, 1 MHz, 500 kHz, 250 kHz, 125 kHz(1)						
F2	PIC18LF2331/2431/4331/4431	-15	+/-5	+15	%	25°C	V_{DD} = 3.0V
F3	All devices	-15	+/-5	+15	%	25°C	V_{DD} = 5.0V
	INTRC Accuracy @ Freq = 31 kHz(2)						
F5	PIC18LF2331/2431/4331/4431	26.562	—	35.938	kHz	25°C	V_{DD} = 3.0V
F6	All devices	26.562	—	35.938	kHz	25°C	V_{DD} = 5.0V

Legend: Shading of rows is to assist in readability of the table.

Note 1: Frequency calibrated at 25°C. OSCTUNE register can be used to compensate for temperature drift.

2: INTRC frequency after calibration.

FIGURE 26-6: CLKO AND I/O TIMING

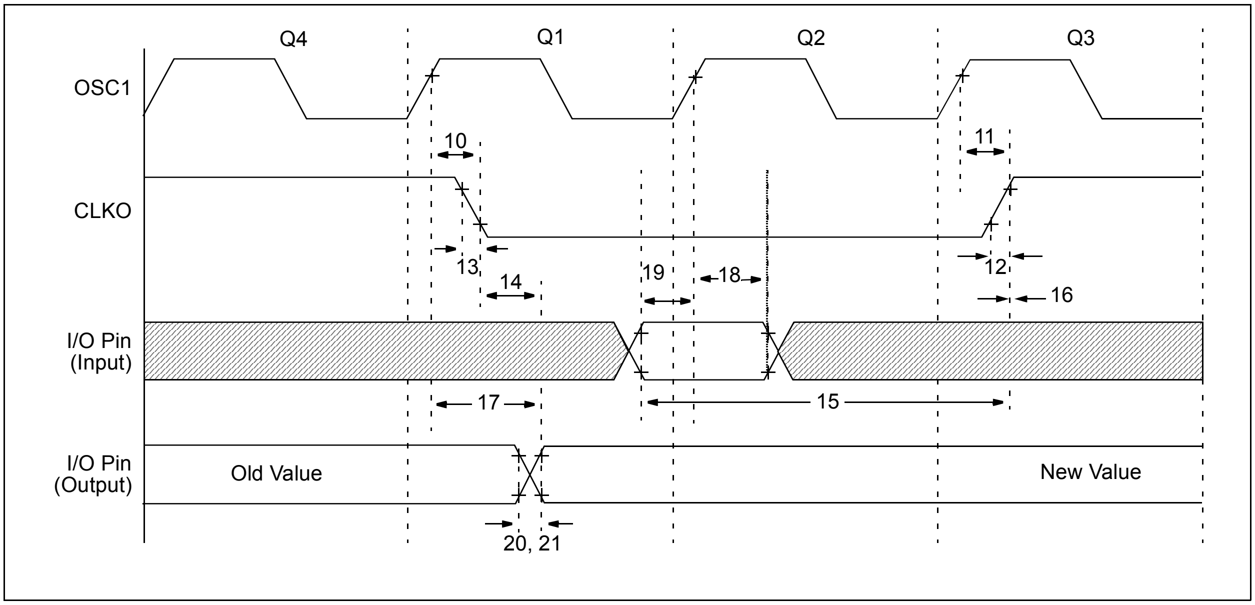


TABLE 26-7: CLKO AND I/O TIMING REQUIREMENTS

Param No.	Symbol	Characteristic		Min	Typ†	Max	Units	Conditions
10	TosH2ckL	OSC1 ↑ to CLKO ↓		—	75	200	ns	**(Note 1)**
11	TosH2ckH	OSC1 ↑ to CLKO ↑		—	75	200	ns	**(Note 1)**
12	TckR	CLKO Rise Time		—	35	100	ns	**(Note 1)**
13	TckF	CLKO Fall Time		—	35	100	ns	**(Note 1)**
14	TckL2ioV	CLKO ↓ to Port Out Valid		—	—	0.5 TCY + 20	ns	**(Note 1)**
15	TioV2ckH	Port In Valid before CLKO ↑		0.25 TCY + 25	—	—	ns	**(Note 1)**
16	TckH2ioI	Port In Hold after CLKO ↑		0	—	—	ns	**(Note 1)**
17	TosH2ioV	OSC1 ↑ (Q1 cycle) to Port Out Valid		—	50	150	ns	
18	TosH2ioI	OSC1 ↑ (Q2 cycle) to Port Input Invalid (I/O in hold time)	PIC18FXX31	100	—	—	ns	
18A			PIC18LFXX31	200	—	—	ns	
19	TioV2osH	Port Input Valid to OSC1 ↑ (I/O in setup time)		0	—	—	ns	
20	TioR	Port Output Rise Time	PIC18FXX31	—	10	25	ns	
20A			PIC18LFXX31	—	—	60	ns	
21	TioF	Port Output Fall Time	PIC18FXX31	—	10	25	ns	
21A			PIC18LFXX31	—	—	60	ns	
22†	TINP	INTx Pin High or Low Time		TCY	—	—	ns	
23†	TRBP	RB<7:4> Change INTx High or Low Time		TCY	—	—	ns	

† These parameters are asynchronous events not related to any internal clock edges.

Note 1: Measurements are taken in RC mode, where CLKO output is 4 x TOSC.

FIGURE 26-7: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING

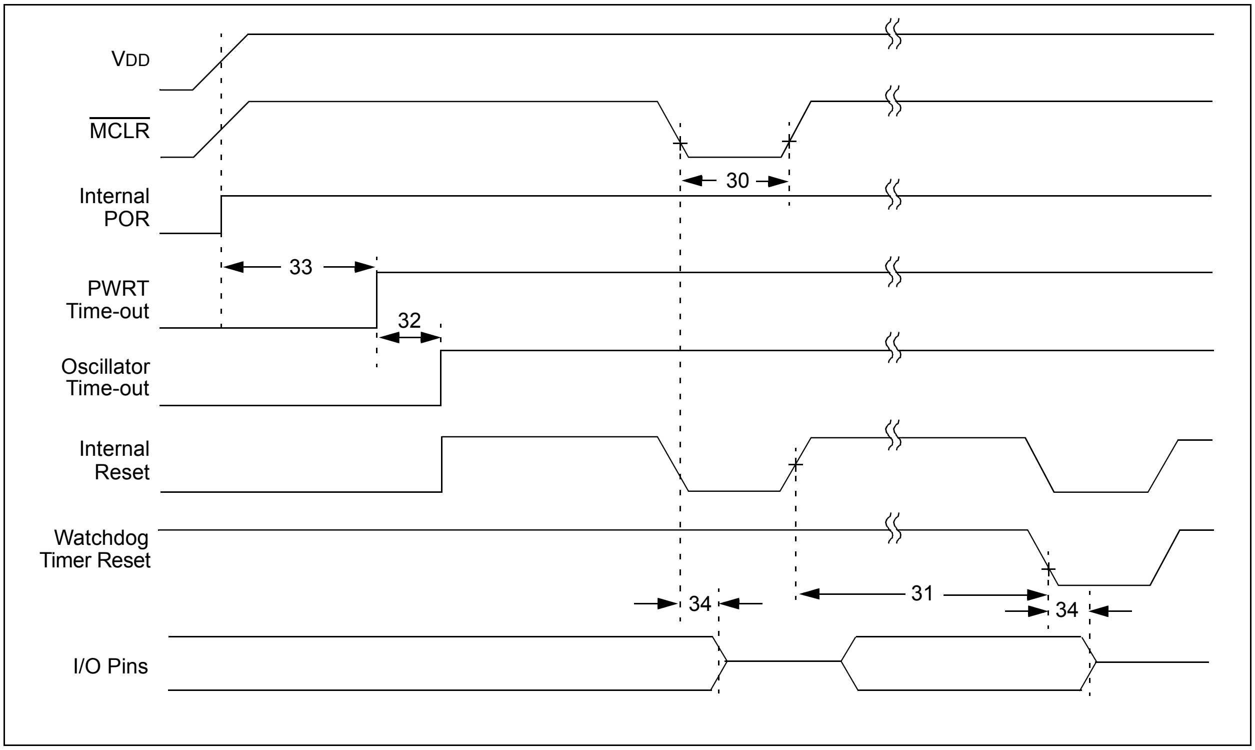


FIGURE 26-8: BROWN-OUT RESET TIMING

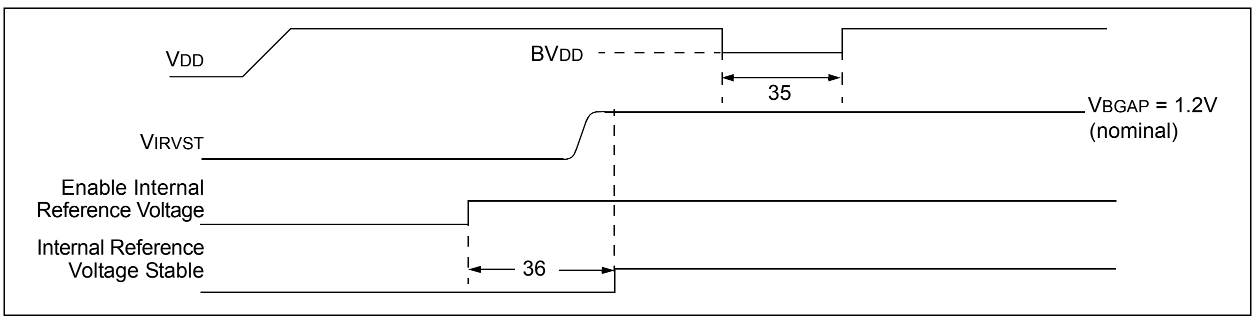


TABLE 26-8: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER AND BROWN-OUT RESET REQUIREMENTS

Param. No.	Symbol	Characteristic	Min	Typ	Max	Units	Conditions
30	TMCL	$\overline{MCLR}$ Pulse Width (low)	2	—	—	μs	
31	TWDT	Watchdog Timer Time-out Period (no postscaler)	—	4.00	—	ms	
32	TOST	Oscillation Start-up Timer Period	1024 TOSC	—	1024 TOSC	—	TOSC = OSC1 period
33	TPWRT	Power-up Timer Period	—	65.5	—	ms	
34	TIOZ	I/O High-impedance from $\overline{MCLR}$ Low or Watchdog Timer Reset	—	2	—	μs	
35	TBOR	Brown-out Reset Pulse Width	200	—	—	μs	VDD ≤ BVDD (see D005)
36	TIRVST	Time for Internal Reference Voltage to become Stable	—	20	50	μs	
37	TLVD	Low-Voltage Detect Pulse Width	200	—	—	μs	VDD ≤ VLVD
38	TCSD	CPU Start-up Time	—	10	—	μs	
39	TIOBST	Time for INTOSC to Stabilize	—	1	—	ms	

FIGURE 26-9: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS

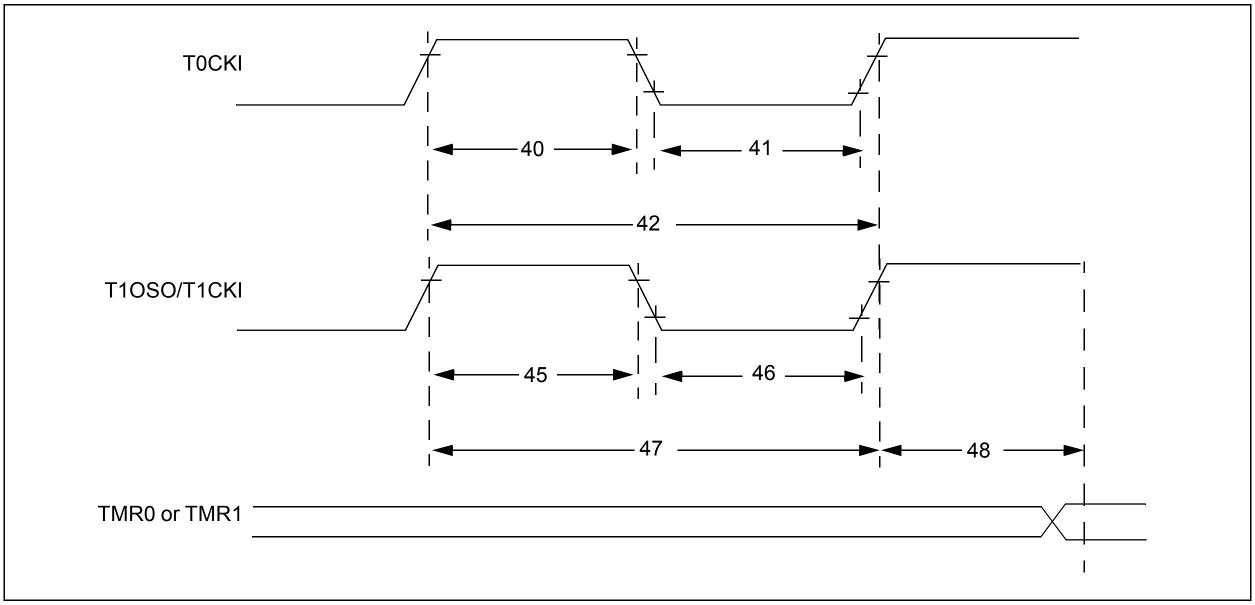


TABLE 26-9: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS

Param No.	Symbol	Characteristic			Min	Max	Units	Conditions
40	Tt0H	T0CKI High Pulse Width		No prescaler	0.5 TCY + 20	—	ns	VDD = 2V
				With prescaler	10	—	ns	
41	Tt0L	T0CKI Low Pulse Width		No prescaler	0.5 TCY + 20	—	ns	
				With prescaler	10	—	ns	
42	Tt0P	T0CKI Period		No prescaler	TCY + 10	—	ns	
				With prescaler	Greater of: 20 ns or (TCY + 40)/N	—	ns	N = prescale value (1, 2, 4,..., 256)
45	Tt1H	T1CKI High Time	Synchronous, no prescaler		0.5 TCY + 20	—	ns	
			Synchronous, with prescaler	PIC18FXX31	10	—	ns	
				PIC18LFXX31	25	—	ns	
			Asynchronous	PIC18FXX31	30	—	ns	
				PIC18LFXX31	50	—	ns	
46	Tt1L	T1CKI Low Time	Synchronous, no prescaler		0.5 TCY + 5	—	ns	
			Synchronous, with prescaler	PIC18FXX31	10	—	ns	
				PIC18LFXX31	25	—	ns	
			Asynchronous	PIC18FXX31	30	—	ns	
				PIC18LFXX31	50	—	ns	
47	Tt1P	T1CKI Input Period	Synchronous		Greater of: 20 ns or (TCY + 40)/N	—	ns	N = prescale value (1, 2, 4, 8)
			Asynchronous		60	—	ns	
	Ft1	T1CKI Oscillator Input Frequency Range			DC	50	kHz	
48	Tcke2tmrI	Delay from External T1CKI Clock Edge to Timer Increment			2 TOSC	7 TOSC	—	

FIGURE 26-10: CAPTURE/COMPARE/PWM TIMINGS (ALL CCP MODULES)

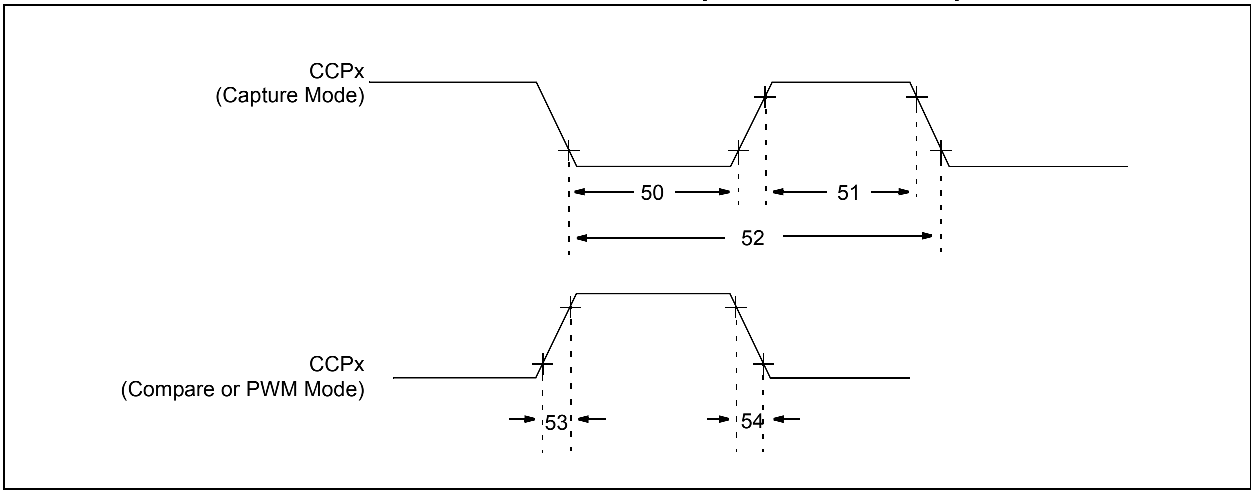


TABLE 26-10: CAPTURE/COMPARE/PWM REQUIREMENTS (ALL CCP MODULES)

Param No.	Symbol	Characteristic			Min	Max	Units	Conditions
50	TccL	CCPx Input Low Time	No prescaler		0.5 T_{CY} + 20	—	ns	
			With prescaler	PIC18FXX31	10	—	ns	
				PIC18LFXX31	20	—	ns	
51	TccH	CCPx Input High Time	No prescaler		0.5 T_{CY} + 20	—	ns	
			With prescaler	PIC18FXX31	10	—	ns	
				PIC18LFXX31	20	—	ns	
52	TccP	CCPx Input Period			$\frac{3 T_{CY} + 40}{N}$	—	ns	N = prescale value (1, 4 or 16)
53	TccR	CCPx Output Fall Time		PIC18FXX31	—	25	ns	
				PIC18LFXX31	—	45	ns	
54	TccF	CCPx Output Fall Time		PIC18FXX31	—	25	ns	
				PIC18LFXX31	—	45	ns	

FIGURE 26-11: EXAMPLE SPI MASTER MODE TIMING (CKE = 0)

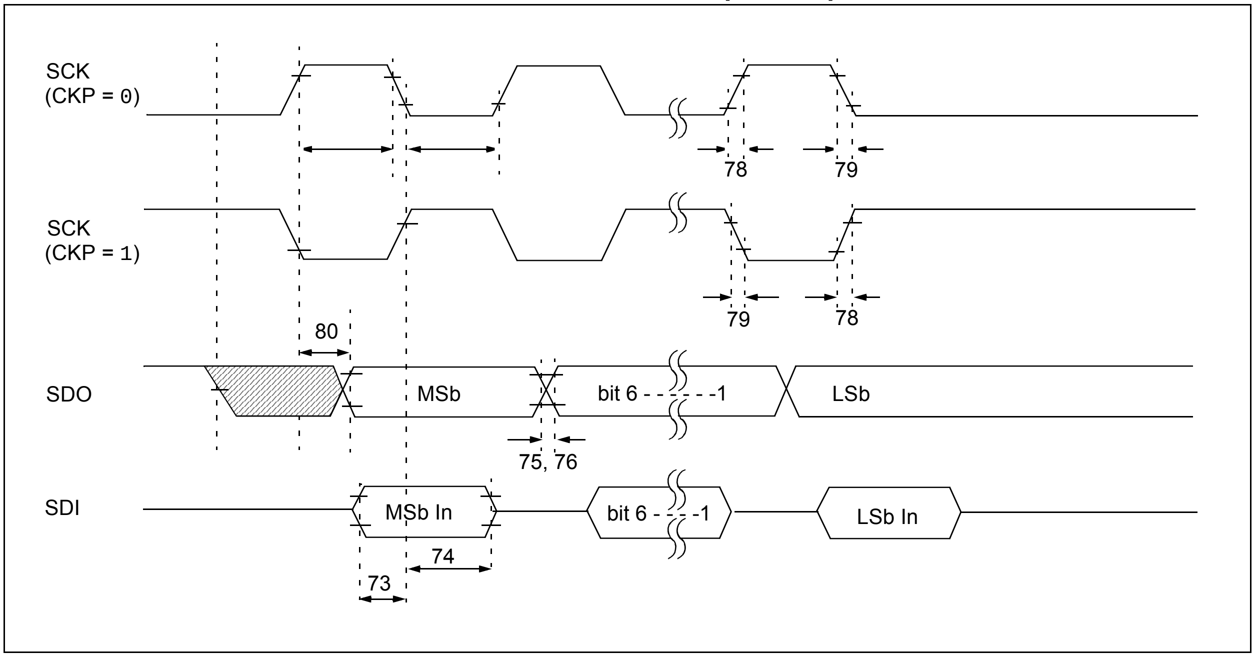


TABLE 26-11: EXAMPLE SPI MODE REQUIREMENTS (MASTER MODE, CKE = 0)

Param No.	Symbol	Characteristic		Min	Max	Units	Conditions
73	TdiV2scH, TdiV2scL	Setup Time of SDI Data Input to SCK Edge		20	—	ns	
73A	Tb2b	Last Clock Edge of Byte 1 to the 1st Clock Edge of Byte 2		1.5 TCY + 40	—	ns	
74	TscH2diL, TscL2diL	Hold Time of SDI Data Input to SCK Edge		40	—	ns	
75	TdoR	SDO Data Output Rise Time	PIC18FXX31	—	25	ns	
			PIC18LFXX31	—	45	ns	
76	TdoF	SDO Data Output Fall Time		—	25	ns	
78	TscR	SCK Output Rise Time	PIC18FXX31	—	25	ns	
			PIC18LFXX31	—	45	ns	
79	TscF	SCK Output Fall Time		—	25	ns	
80	TscH2doV, TscL2doV	SDO Data Output Valid after SCK Edge	PIC18FXX31	—	50	ns	
			PIC18LFXX31	—	100	ns	

FIGURE 26-12: EXAMPLE SPI MASTER MODE TIMING (CKE = 1)

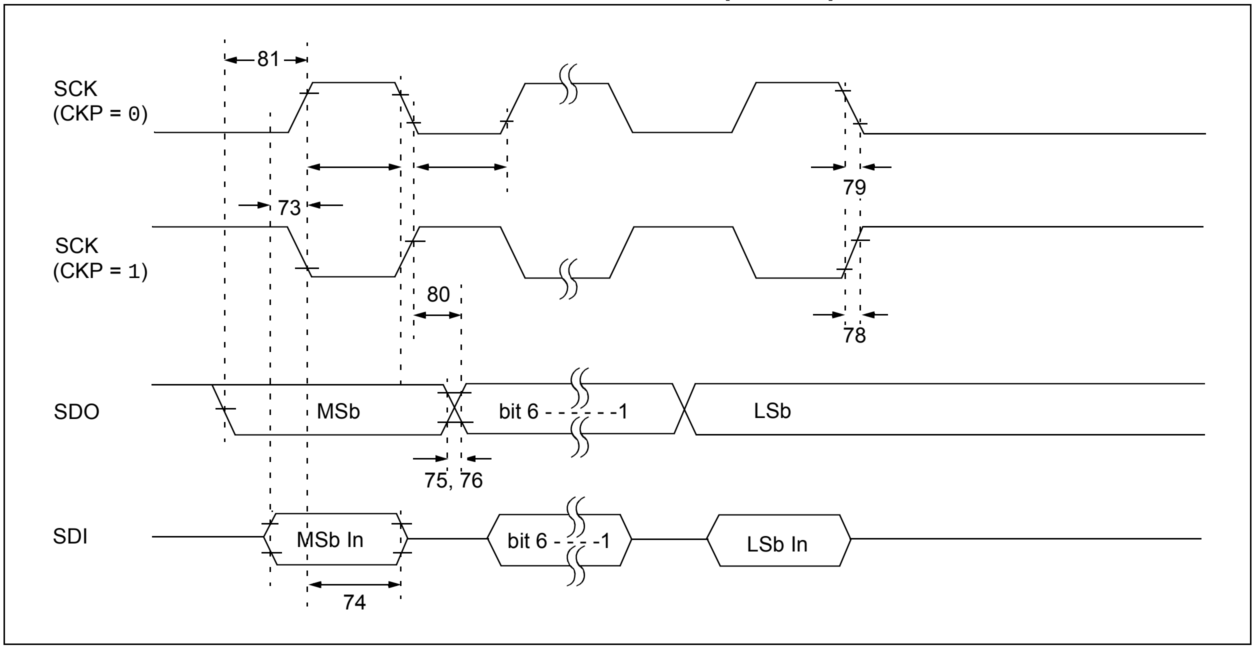


TABLE 26-12: EXAMPLE SPI MODE REQUIREMENTS (MASTER MODE, CKE = 1)

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
73	TdiV2scH, TdiV2scL	Setup Time of SDI Data Input to SCK Edge		20	—	ns	
73A	Tb2b	Last Clock Edge of Byte 1 to the 1st Clock Edge of Byte 2		1.5 TCY + 40	—	ns	
74	TscH2diL, TscL2diL	Hold Time of SDI Data Input to SCK Edge		40	—	ns	
75	TdoR	SDO Data Output Rise Time	PIC18FXX31	—	25	ns	
			PIC18LFXX31	—	45	ns	
76	TdoF	SDO Data Output Fall Time		—	25	ns	
78	TscR	SCK Output Rise Time	PIC18FXX31	—	25	ns	
			PIC18LFXX31	—	45	ns	
79	TscF	SCK Output Fall Time		—	25	ns	
80	TscH2doV, TscL2doV	SDO Data Output Valid after SCK Edge	PIC18FXX31	—	50	ns	
			PIC18LFXX31	—	100	ns	
81	TdoV2scH, TdoV2scL	SDO Data Output Setup to SCK Edge		TCY	—	ns	

FIGURE 26-13: EXAMPLE SPI SLAVE MODE TIMING (CKE = 0)

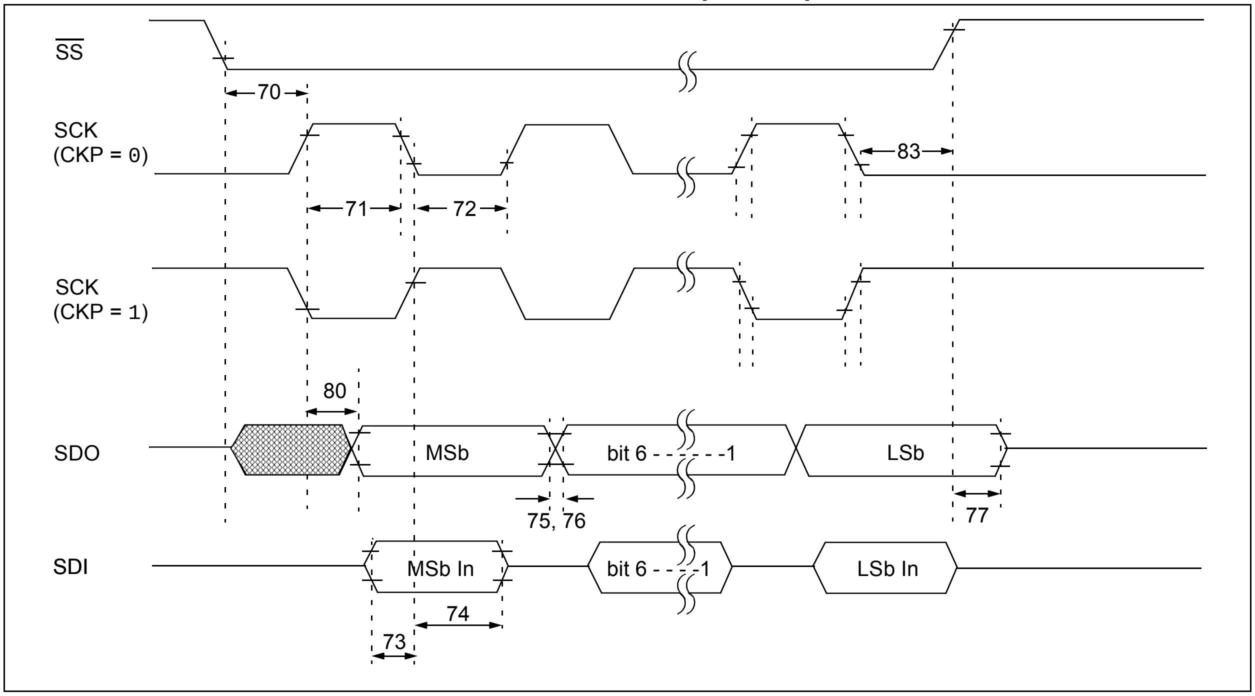


TABLE 26-13: EXAMPLE SPI MODE REQUIREMENTS (SLAVE MODE, CKE = 0)

Param No.	Symbol	Characteristic		Min	Max	Units	Conditions
70	TssL2scH, TssL2scL	$\overline{SS}$ ↓ to SCK ↓ or SCK ↑ Input		TCY	—	ns	
71	TscH	SCK Input High Time	Continuous	1.25 TCY + 30	—	ns	
71A			Single byte	40	—	ns	**(Note 1)**
72	TscL	SCK Input Low Time	Continuous	1.25 TCY + 30	—	ns	
72A			Single byte	40	—	ns	**(Note 1)**
73	TdiV2scH, TdiV2scL	Setup Time of SDI Data Input to SCK Edge		20	—	ns	
73A	TB2B	Last Clock Edge of Byte 1 to the First Clock Edge of Byte 2		1.5 TCY + 40	—	ns	**(Note 2)**
74	TscH2diL, TscL2diL	Hold Time of SDI Data Input to SCK Edge		40	—	ns	
75	TdoR	SDO Data Output Rise Time	PIC18FXX31	—	25	ns	
			PIC18LFXX31	—	45	ns	
76	TdoF	SDO Data Output Fall Time		—	25	ns	
77	TssH2doZ	$\overline{SS}$ ↑ to SDO Output High-Impedance		10	50	ns	
80	TscH2doV, TscL2doV	SDO Data Output Valid after SCK Edge	PIC18FXX31	—	50	ns	
			PIC18LFXX31	—	100	ns	
83	TscH2ssH, TscL2ssH	$\overline{SS}$ ↑ after SCK Edge		1.5 TCY + 40	—	ns	

Note 1: Requires the use of Parameter 73A.

2: Only if Parameter 71A and 72A are used.

FIGURE 26-14: EXAMPLE SPI SLAVE MODE TIMING (CKE = 1)

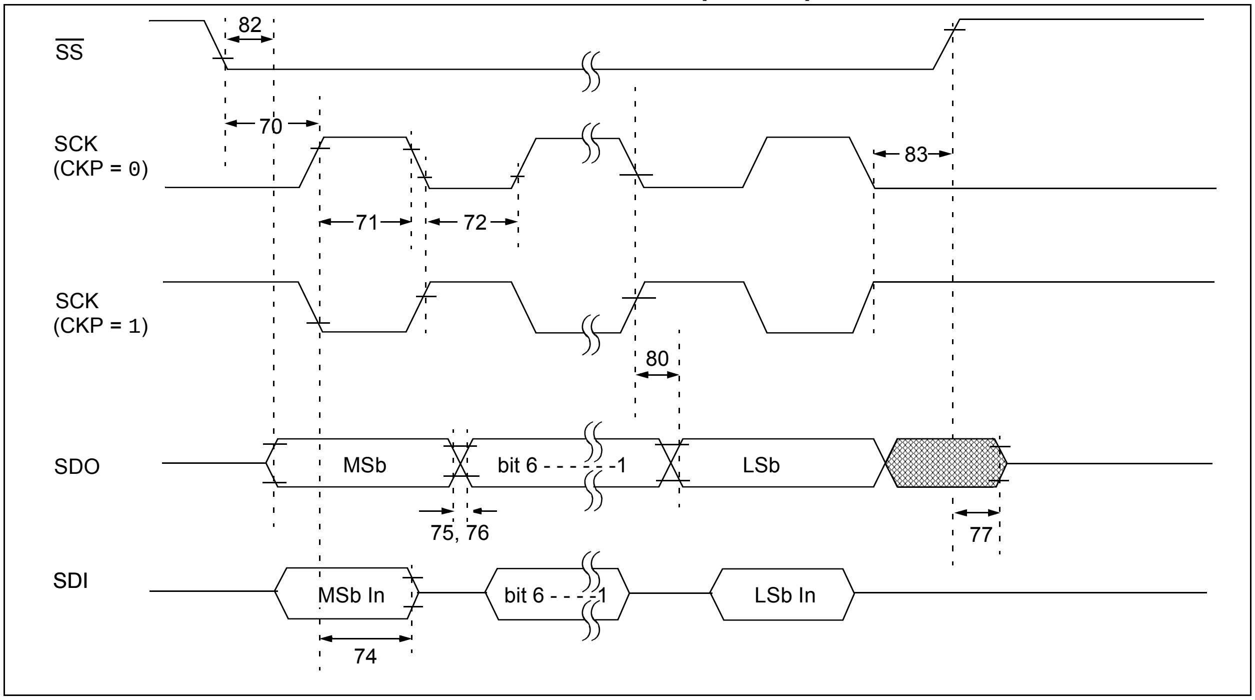


TABLE 26-14: EXAMPLE SPI SLAVE MODE REQUIREMENTS (CKE = 1)

Param No.	Symbol	Characteristic		Min	Max	Units	Conditions
70	TssL2scH, TssL2scL	$\overline{SS}$ ↓ to SCK ↓ or SCK ↑ Input		TCY	—	ns	
71	TscH	SCK Input High Time	Continuous	1.25 TCY + 30	—	ns	
71A			Single byte	40	—	ns	**(Note 1)**
72	TscL	SCK Input Low Time	Continuous	1.25 TCY + 30	—	ns	
72A			Single byte	40	—	ns	**(Note 1)**
73A	TB2B	Last Clock Edge of Byte 1 to the First Clock Edge of Byte 2		1.5 TCY + 40	—	ns	**(Note 2)**
74	TscH2diL, TscL2diL	Hold Time of SDI Data Input to SCK Edge		40	—	ns	
75	TdoR	SDO Data Output Rise Time	PIC18FXX31	—	25	ns	
			PIC18LFXX31	—	45	ns	
76	TdoF	SDO Data Output Fall Time		—	25	ns	
77	TssH2doZ	$\overline{SS}$ ↑ to SDO Output High-Impedance		10	50	ns	
80	TscH2doV, TscL2doV	SDO Data Output Valid after SCK Edge	PIC18FXX31	—	50	ns	
			PIC18LFXX31	—	100	ns	
82	TssL2doV	SDO Data Output Valid after $\overline{SS}$ ↓ Edge	PIC18FXX31	—	50	ns	
			PIC18LFXX31	—	100	ns	
83	TscH2ssH, TscL2ssH	$\overline{SS}$ ↑ after SCK Edge		1.5 TCY + 40	—	ns	

Note 1: Requires the use of Parameter 73A.

2: Only if Parameter 71A and 72A are used.

FIGURE 26-15: I^2C™ BUS START/STOP BITS TIMING

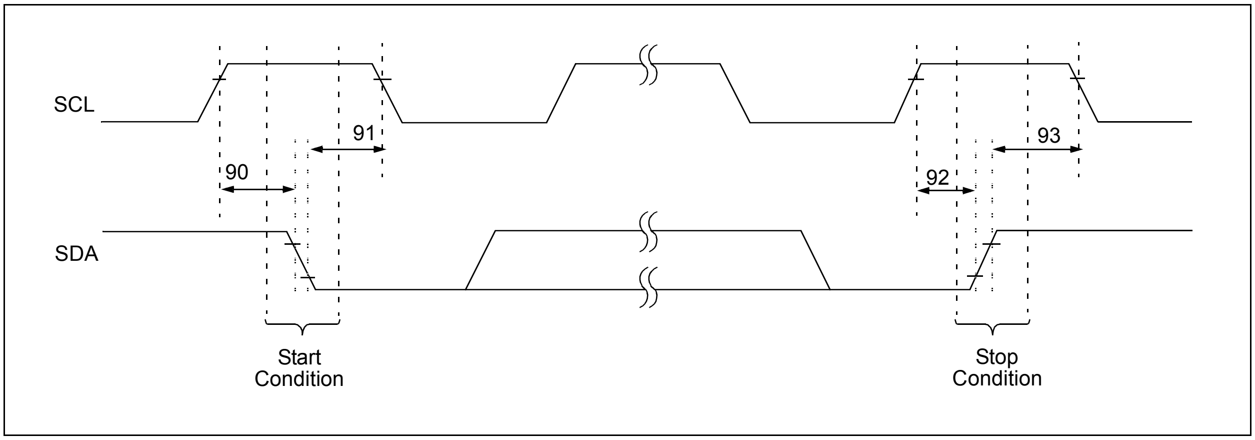


TABLE 26-15: I^2C™ BUS START/STOP BITS REQUIREMENTS (SLAVE MODE)

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
90	TSU:STA	Start Condition	100 kHz mode	4700	—	ns	Only relevant for repeated Start condition
		Setup Time	400 kHz mode	600	—		
91	THD:STA	Start Condition	100 kHz mode	4000	—	ns	After this period, the first clock pulse is generated
		Hold Time	400 kHz mode	600	—		
92	TSU:STO	Stop Condition	100 kHz mode	4000	—	ns	
		Setup Time	400 kHz mode	600	—		
93	THD:STO	Stop Condition	100 kHz mode	4700	—	ns	
		Hold Time	400 kHz mode	600	—		

FIGURE 26-16: I^2C™ BUS DATA TIMING

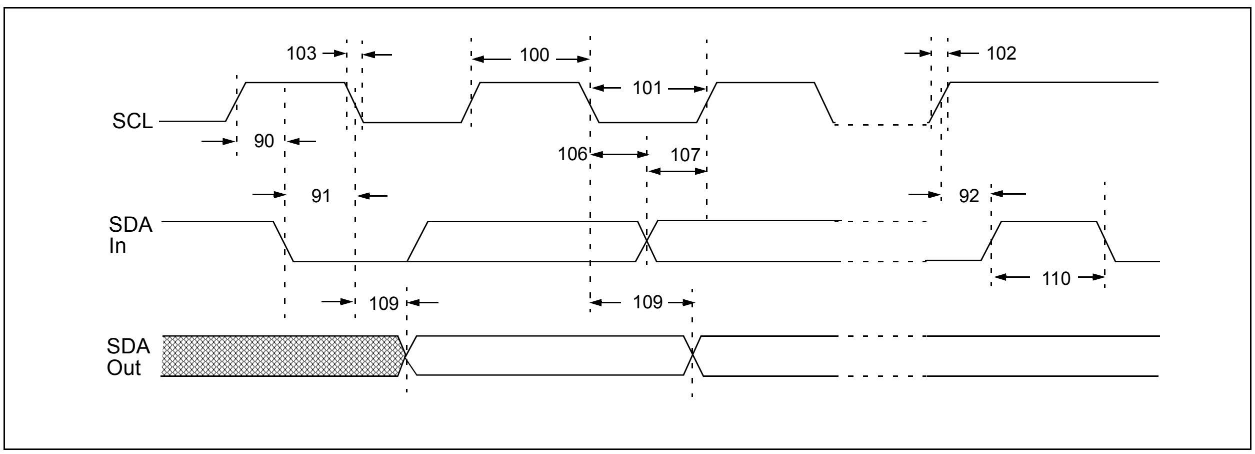


TABLE 26-16: I²C™ BUS DATA REQUIREMENTS (SLAVE MODE)

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
100	THIGH	Clock High Time	100 kHz mode	4.0	—	μs	PIC18FXX31 must operate at a minimum of 1.5 MHz
			400 kHz mode	0.6	—	μs	PIC18FXX31 must operate at a minimum of 10 MHz
			SSP module	1.5 TCY	—		
101	TLOW	Clock Low Time	100 kHz mode	4.7	—	μs	PIC18FXX31 must operate at a minimum of 1.5 MHz
			400 kHz mode	1.3	—	μs	PIC18FXX31 must operate at a minimum of 10 MHz
			SSP Module	1.5 TCY	—		
102	TR	SDA and SCL Rise Time	100 kHz mode	—	1000	ns	
			400 kHz mode	20 + 0.1 CB	300	ns	CB is specified to be from 10 to 400 pF
103	TF	SDA and SCL Fall Time	100 kHz mode	—	300	ns	
			400 kHz mode	20 + 0.1 CB	300	ns	CB is specified to be from 10 to 400 pF
90	TSU:STA	Start Condition Setup Time	100 kHz mode	4.7	—	μs	Only relevant for Repeated Start condition
			400 kHz mode	0.6	—	μs	
91	THD:STA	Start Condition Hold Time	100 kHz mode	4.0	—	μs	After this period, the first clock pulse is generated
			400 kHz mode	0.6	—	μs	
106	THD:DAT	Data Input Hold Time	100 kHz mode	0	—	ns	
			400 kHz mode	0	0.9	μs	
107	TSU:DAT	Data Input Setup Time	100 kHz mode	250	—	ns	**(Note 2)**
			400 kHz mode	100	—	ns	
92	TSU:STO	Stop Condition Setup Time	100 kHz mode	4.7	—	μs	
			400 kHz mode	0.6	—	μs	
109	TAA	Output Valid From Clock	100 kHz mode	—	3500	ns	**(Note 1)**
			400 kHz mode	—	—	ns	
110	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	
D102	CB	Bus Capacitive Loading		—	400	pF	

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 I²C bus device can be used in a Standard mode I²C bus system, but the requirement, TSU:DAT ≥ 250 ns, must then be met. This will automatically be the case if the device does not stretch the LOW period of the SCL signal. If such a device does stretch the LOW period of the SCL signal, it must output the next data bit to the SDA line,.
TR max. + TSU:DAT = 1000 + 250 = 1250 ns (according to the Standard mode I²C bus specification), before the SCL line is released.

TABLE 26-17: SSP I²C™ BUS DATA REQUIREMENTS

Param. No.	Symbol	Characteristic		Min	Max	Units	Conditions
100	THIGH	Clock High Time	100 kHz mode	2(TOSC)(BRG + 1)	—	ms	
			400 kHz mode	2(TOSC)(BRG + 1)	—	ms	
101	TLOW	Clock Low Time	100 kHz mode	2(TOSC)(BRG + 1)	—	ms	
			400 kHz mode	2(TOSC)(BRG + 1)	—	ms	
102	TR	SDA and SCL Rise Time	100 kHz mode	—	1000	ns	CB is specified to be from 10 to 400 pF
			400 kHz mode	20 + 0.1 CB	300	ns	
103	TF	SDA and SCL Fall Time	100 kHz mode	—	300	ns	CB is specified to be from 10 to 400 pF
			400 kHz mode	20 + 0.1 CB	300	ns	
90	TSU:STA	Start Condition Setup Time	100 kHz mode	2(TOSC)(BRG + 1)	—	ms	Only relevant for Repeated Start condition
			400 kHz mode	2(TOSC)(BRG + 1)	—	ms	
91	THD:STA	Start Condition Hold Time	100 kHz mode	2(TOSC)(BRG + 1)	—	ms	After this period, the first clock pulse is generated
			400 kHz mode	2(TOSC)(BRG + 1)	—	ms	
106	THD:DAT	Data Input Hold Time	100 kHz mode	0	—	ns	
			400 kHz mode	0	0.9	ms	
107	TSU:DAT	Data Input Setup Time	100 kHz mode	250	—	ns	
			400 kHz mode	100	—	ns	
92	TSU:STO	Stop Condition Setup Time	100 kHz mode	2(TOSC)(BRG + 1)	—	ms	
			400 kHz mode	2(TOSC)(BRG + 1)	—	ms	
109	TAA	Output Valid from Clock	100 kHz mode	—	3500	ns	
			400 kHz mode	—	1000	ns	
110	TBUF	Bus Free Time	100 kHz mode	4.7	—	ms	Time the bus must be free before a new transmission can start
			400 kHz mode	1.3	—	ms	
D102	CB	Bus Capacitive Loading		—	400	pF	

FIGURE 26-17: EUSART SYNCHRONOUS TRANSMISSION (MASTER/SLAVE) TIMING

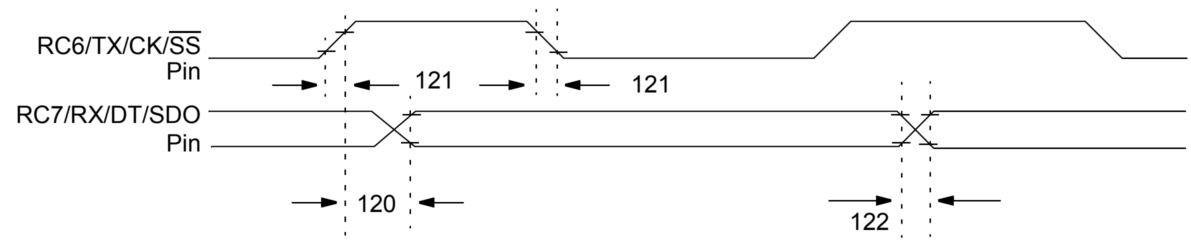


TABLE 26-18: EUSART SYNCHRONOUS TRANSMISSION REQUIREMENTS

Param No.	Symbol	Characteristic		Min	Max	Units	Conditions
120	TckH2dtV	SYNC XMIT (MASTER & SLAVE) Clock High to Data Out Valid	PIC18FXX31	—	40	ns	
			PIC18LFXX31	—	100	ns	
121	Tckrf	Clock Out Rise Time and Fall Time (Master mode)	PIC18FXX31	—	20	ns	
			PIC18LFXX31	—	50	ns	
122	Tdtrf	Data Out Rise Time and Fall Time	PIC18FXX31	—	20	ns	
			PIC18LFXX31	—	50	ns	

FIGURE 26-18: EUSART SYNCHRONOUS RECEIVE (MASTER/SLAVE) TIMING

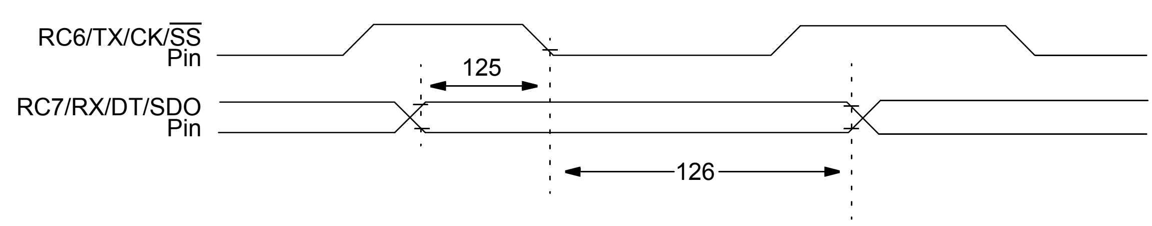


TABLE 26-19: EUSART SYNCHRONOUS RECEIVE REQUIREMENTS

Param. No.	Symbol	Characteristic	Min	Max	Units	Conditions
125	TdtV2ckl	SYNC RCV (MASTER & SLAVE) Data Hold before CK ↓ (DT hold time)	10	—	ns	
126	TckL2dtl	Data Hold after CK ↓ (DT hold time)	15	—	ns	

TABLE 26-20: A/D CONVERTER CHARACTERISTICS

PIC18LF2331/2431/4331/4431 (Industrial)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial
PIC18F2331/2431/4331/4431 (Industrial)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial; -40°C ≤ TA ≤ +125°C for extended

Param No.	Symbol	Characteristic	Min	Typ	Max	Units	Conditions
Device Supply							
	AVDD	Analog VDD Supply	VDD – 0.3	—	VDD + 0.3	V	
	AVSS	Analog VSS Supply	VSS – 0.3	—	VSS + 0.3	V	
	IAD	Module Current (during conversion)	— —	500 250	— —	μA μA	VDD = 5V VDD = 2.5V
	IADO	Module Current Off	—	—	1.0	μA	
AC Timing Parameters							
A10	FTHR	Throughput Rate	— —	— —	200 75	ksps ksps	VDD = 5V, single channel VDD < 3V, single channel
A11	TAD	A/D Clock Period	385 1000	— —	20,000 20,000	ns ns	VDD = 5V VDD = 3V
A12	TRC	A/D Internal RC Oscillator Period	— — —	500 750 10000	1500 2250 20000	ns ns ns	PIC18F parts PIC18LF parts AVDD < 3.0V
A13	TCNV	Conversion Time(1)	12	12	12	TAD	
A14	TACQ	Acquisition Time(2)	2(2)	—	—	TAD	
A16	TTC	Conversion Start from External	1/4 TCY	—	—		
Reference Inputs							
A20	VREF	Reference Voltage for 10-Bit Resolution (VREF+ – VREF-)	1.5 1.8	— —	AVDD – AVSS AVDD – AVSS	V V	VDD ≥ 3V VDD < 3V
A21	VREFH	Reference Voltage High (AVDD or VREF+)	1.5V	—	AVDD	V	VDD ≥ 3V
A22	VREFL	Reference Voltage Low (AVSS or VREF-)	AVSS	—	VREFH – 1.5V	V	
A23	IREF	Reference Current	— —	150 μA 75 μA	— —		VDD = 5V VDD = 2.5V
Analog Input Characteristics							
A26	VAIN	Input Voltage(3)	AVSS – 0.3	—	AVDD + 0.3	V	
A30	ZAIN	Recommended Impedance of Analog Voltage Source	—	—	2.5	kΩ	
A31	ZCHIN	Analog Channel Input Impedance	—	—	10.0	kΩ	VDD = 3.0V
DC Performance							
A41	NR	Resolution	10 bits			—	
A42	EIL	Integral Nonlinearity	—	—	<±1	LSb	VDD ≥ 3.0V VREFH ≥ 3.0V
A43	EIL	Differential Nonlinearity	—	—	<±1	LSb	VDD ≥ 3.0V VREFH ≥ 3.0V
A45	EOFF	Offset Error	—	±0.5	<±1.5	LSb	VDD ≥ 3.0V VREFH ≥ 3.0V
A46	EGA	Gain Error	—	±0.5	<±1.5	LSb	VDD ≥ 3.0V VREFH ≥ 3.0V
A47	—	Monotonicity(4)	guaranteed			—	VDD ≥ 3.0V VREFH ≥ 3.0V

Note 1: Conversion time does not include acquisition time. See **Section 21.0 "10-Bit High-Speed Analog-to-Digital Converter (A/D) Module"** for a full discussion of acquisition time requirements.

2: In Sequential modes, TACQ should be 12 TAD or greater.

3: For VDD < 2.7V and temperature below 0°C, VAIN should be limited to range < VDD/2.

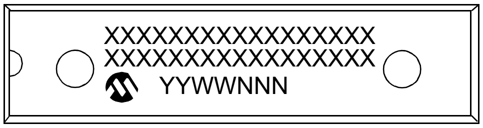
4: The A/D conversion result never decreases with an increase in the input voltage and has no missing codes.

NOTES:

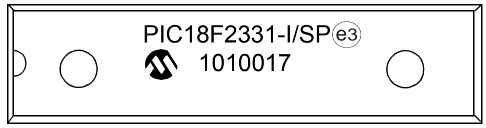
27.0 PACKAGING INFORMATION

27.1 Package Marking Information

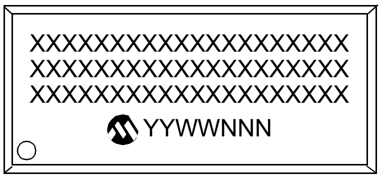
28-Lead SPDIP (Skinny PDIP)



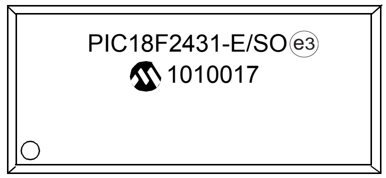
Example



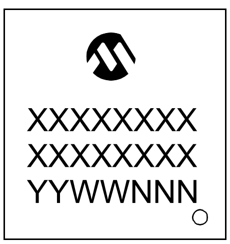
28-Lead SOIC



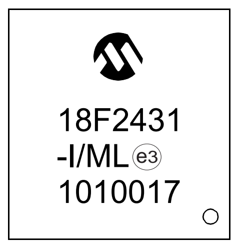
Example



28-Lead QFN



Example

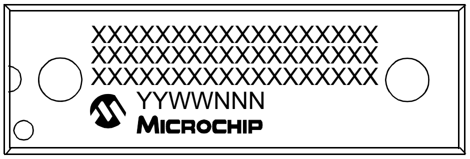


Legend:	XX...X	Customer-specific information
	Y	Year code (last digit of calendar year)
	YY	Year code (last 2 digits of calendar year)
	WW	Week code (week of January 1 is week '01')
	NNN	Alphanumeric traceability code
	e3	Pb-free JEDEC designator for Matte Tin (Sn)
	*	This package is Pb-free. The Pb-free JEDEC designator (e3) can be found on the outer packaging for this package.

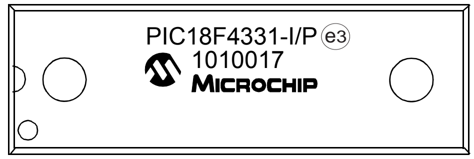
Note: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.

27.1 Package Marking Information (Continued)

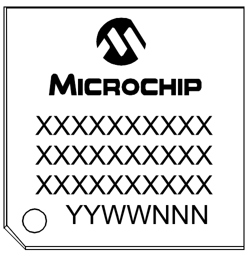
40-Lead PDIP



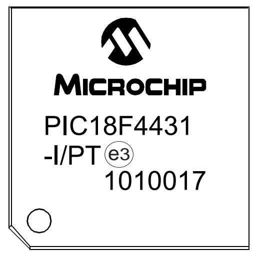
Example



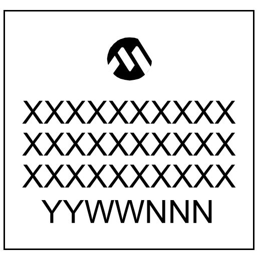
44-Lead TQFP



Example



44-Lead QFN



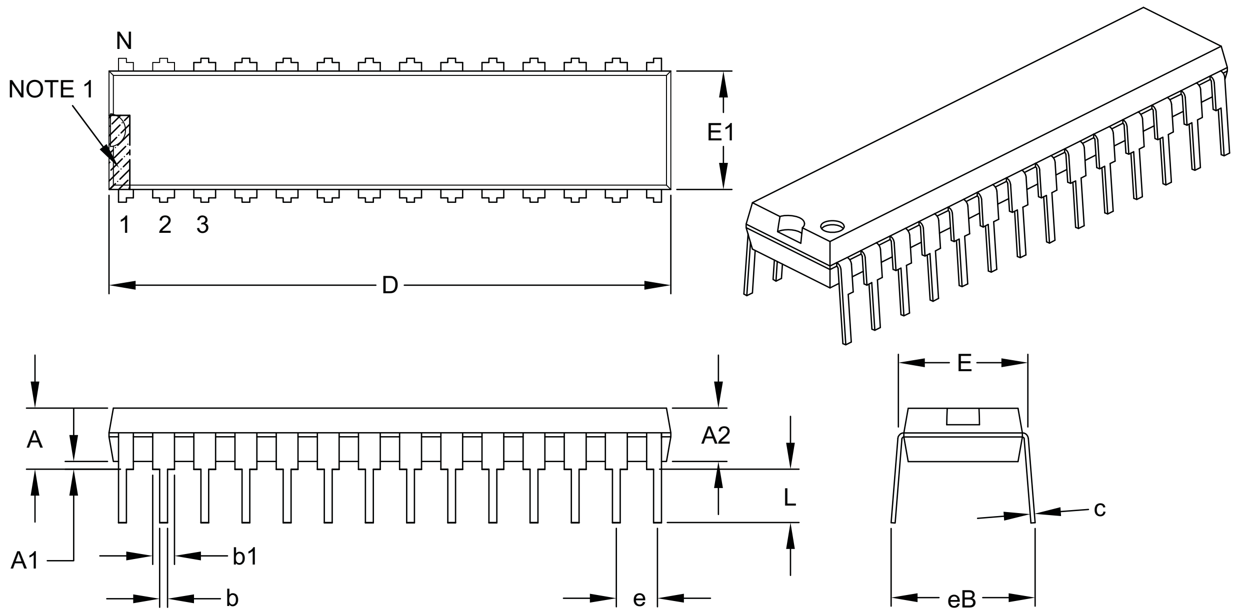
Example

27.2 Package Details

The following sections give the technical details of the packages.

28-Lead Skinny Plastic Dual In-Line (SP) – 300 mil Body [SPDIP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		INCHES		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		28	
Pitch	e		.100 BSC	
Top to Seating Plane	A	–	–	.200
Molded Package Thickness	A2	.120	.135	.150
Base to Seating Plane	A1	.015	–	–
Shoulder to Shoulder Width	E	.290	.310	.335
Molded Package Width	E1	.240	.285	.295
Overall Length	D	1.345	1.365	1.400
Tip to Seating Plane	L	.110	.130	.150
Lead Thickness	c	.008	.010	.015
Upper Lead Width	b1	.040	.050	.070
Lower Lead Width	b	.014	.018	.022
Overall Row Spacing §	eB	–	–	.430

Notes:

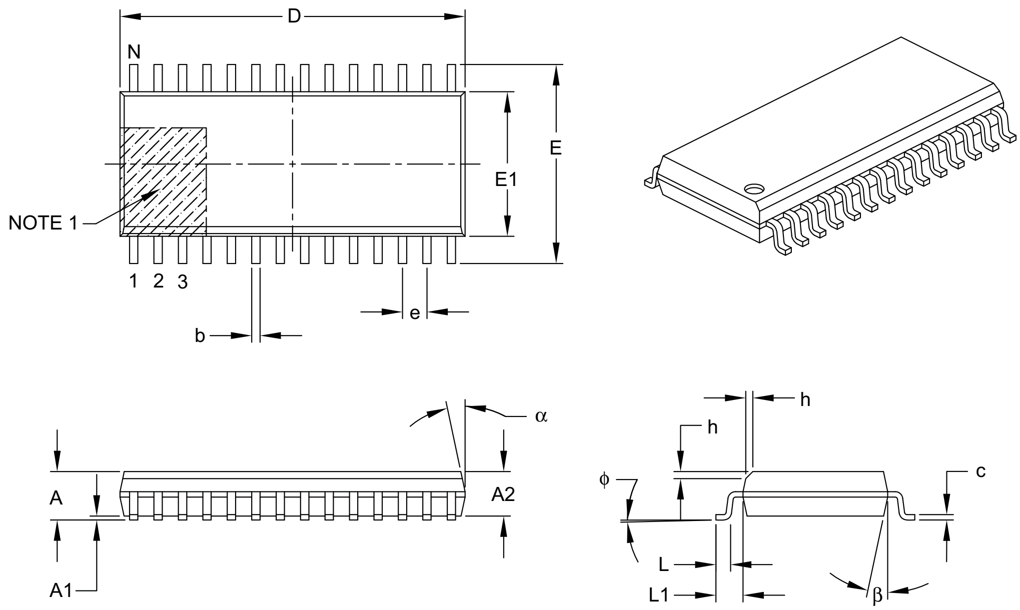
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-070B

28-Lead Plastic Small Outline (SO) – Wide, 7.50 mm Body [SOIC]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



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	
Foot Angle Top	ϕ	0°	–	8°
Lead Thickness	c	0.18	–	0.33
Lead Width	b	0.31	–	0.51
Mold Draft Angle Top	α	5°	–	15°
Mold Draft Angle Bottom	β	5°	–	15°

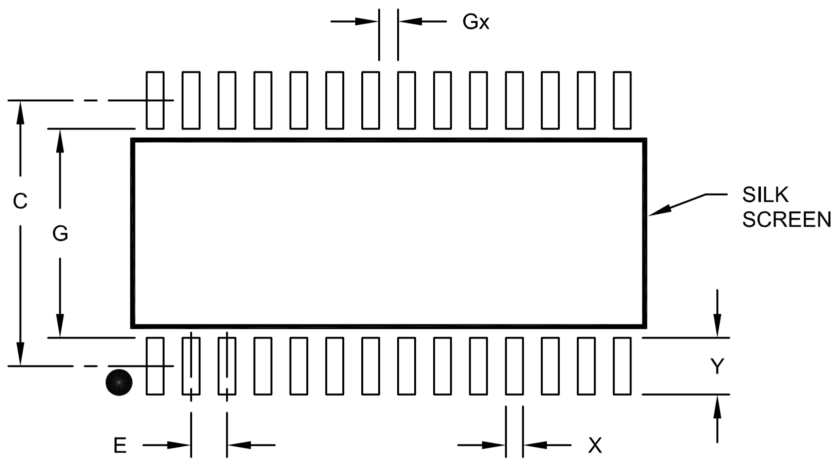
Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.15 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-052B

28-Lead Plastic Small Outline (SO) - Wide, 7.50 mm Body [SOIC]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

	Units	MILLIMETERS		
	Dimension Limits	MIN	NOM	MAX
Contact Pitch	E	1.27 BSC		
Contact Pad Spacing	C		9.40	
Contact Pad Width (X28)	X			0.60
Contact Pad Length (X28)	Y			2.00
Distance Between Pads	Gx	0.67		
Distance Between Pads	G	7.40		

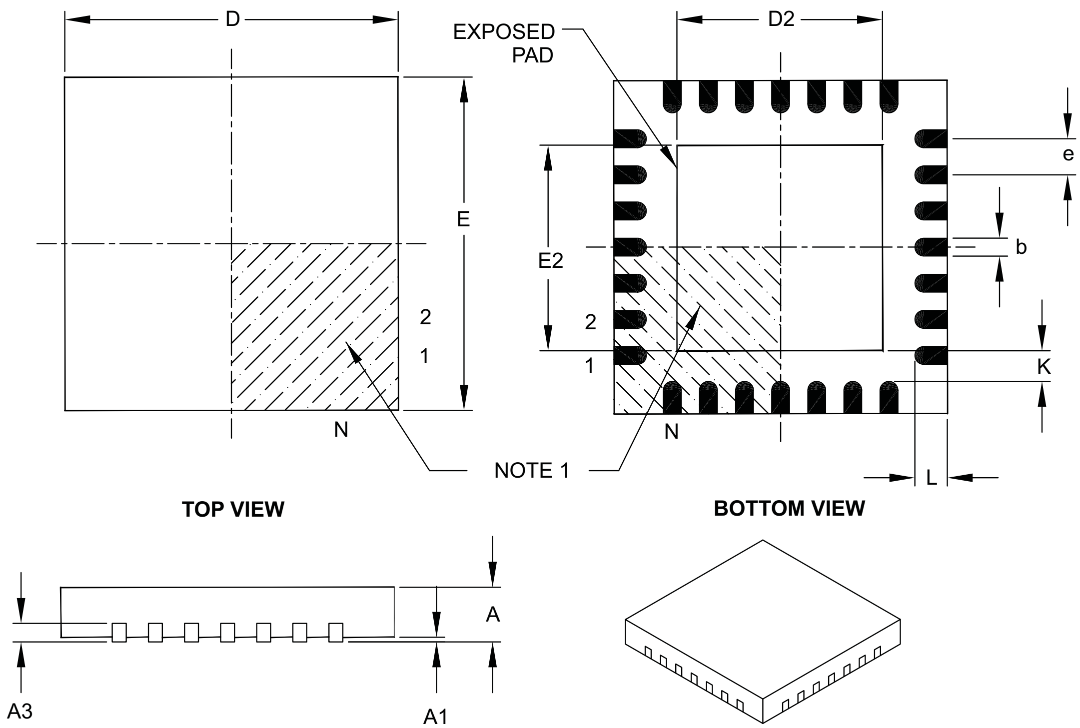
Notes:

1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2052A

28-Lead Plastic Quad Flat, No Lead Package (ML) – 6x6 mm Body [QFN] with 0.55 mm Contact Length

Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



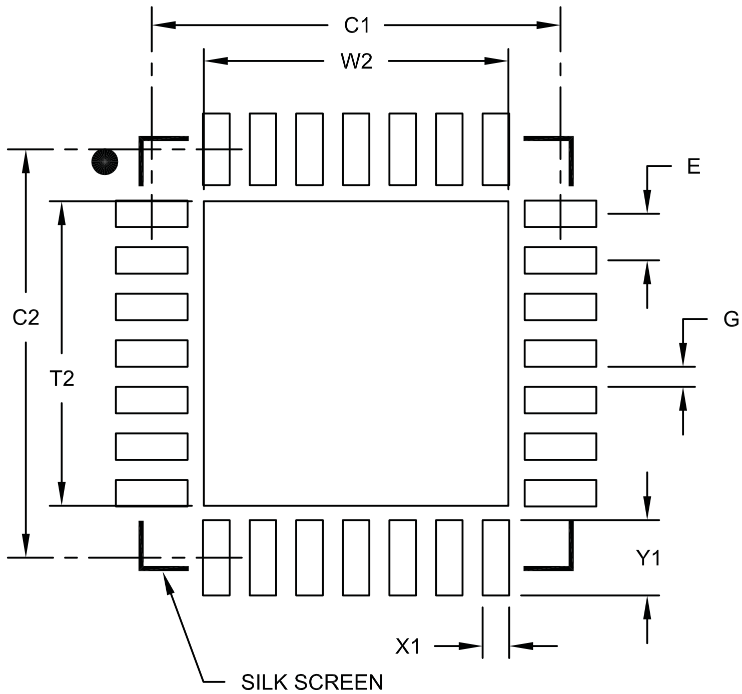
Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N	28		
Pitch	e	0.65 BSC		
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Contact Thickness	A3	0.20 REF		
Overall Width	E	6.00 BSC		
Exposed Pad Width	E2	3.65	3.70	4.20
Overall Length	D	6.00 BSC		
Exposed Pad Length	D2	3.65	3.70	4.20
Contact Width	b	0.23	0.30	0.35
Contact Length	L	0.50	0.55	0.70
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 C04-105B

28-Lead Plastic Quad Flat, No Lead Package (ML) – 6x6 mm Body [QFN] with 0.55 mm Contact Length

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

		Units	MILLIMETERS	
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.65 BSC	
Optional Center Pad Width	W2			4.25
Optional Center Pad Length	T2			4.25
Contact Pad Spacing	C1		5.70	
Contact Pad Spacing	C2		5.70	
Contact Pad Width (X28)	X1			0.37
Contact Pad Length (X28)	Y1			1.00
Distance Between Pads	G	0.20		

Notes:

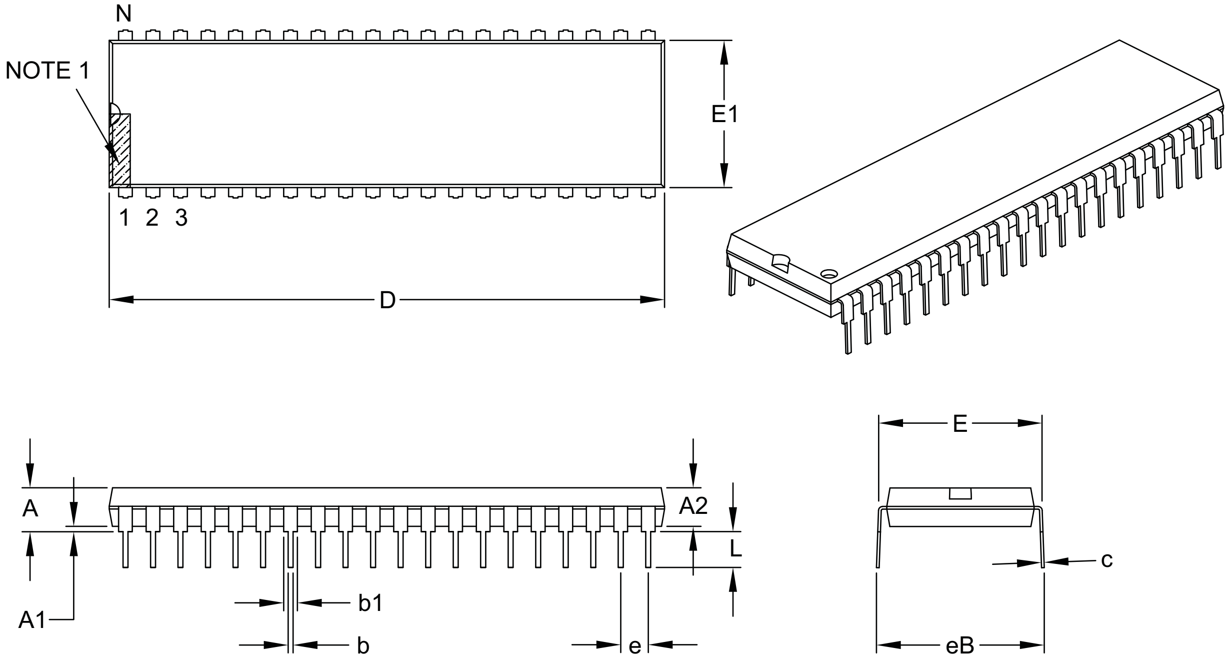
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2105A

40-Lead Plastic Dual In-Line (P) – 600 mil Body [PDIP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



		INCHES		
	Dimension Limits	MIN	NOM	MAX
Number of Pins	N		40	
Pitch	e		.100 BSC	
Top to Seating Plane	A	–	–	.250
Molded Package Thickness	A2	.125	–	.195
Base to Seating Plane	A1	.015	–	–
Shoulder to Shoulder Width	E	.590	–	.625
Molded Package Width	E1	.485	–	.580
Overall Length	D	1.980	–	2.095
Tip to Seating Plane	L	.115	–	.200
Lead Thickness	c	.008	–	.015
Upper Lead Width	b1	.030	–	.070
Lower Lead Width	b	.014	–	.023
Overall Row Spacing §	eB	–	–	.700

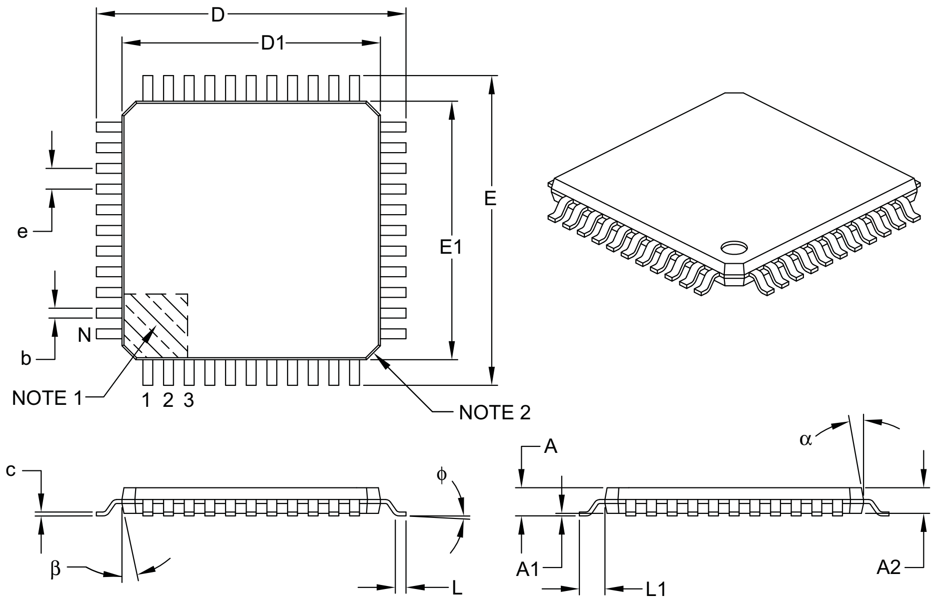
Notes:
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-016B

44-Lead Plastic Thin Quad Flatpack (PT) – 10x10x1 mm Body, 2.00 mm [TQFP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



	Units	MILLIMETERS		
	Dimension Limits	MIN	NOM	MAX
Number of Leads	N	44		
Lead Pitch	e	0.80 BSC		
Overall Height	A	–	–	1.20
Molded Package Thickness	A2	0.95	1.00	1.05
Standoff	A1	0.05	–	0.15
Foot Length	L	0.45	0.60	0.75
Footprint	L1	1.00 REF		
Foot Angle	φ	0°	3.5°	7°
Overall Width	E	12.00 BSC		
Overall Length	D	12.00 BSC		
Molded Package Width	E1	10.00 BSC		
Molded Package Length	D1	10.00 BSC		
Lead Thickness	c	0.09	–	0.20
Lead Width	b	0.30	0.37	0.45
Mold Draft Angle Top	α	11°	12°	13°
Mold Draft Angle Bottom	β	11°	12°	13°

Notes:

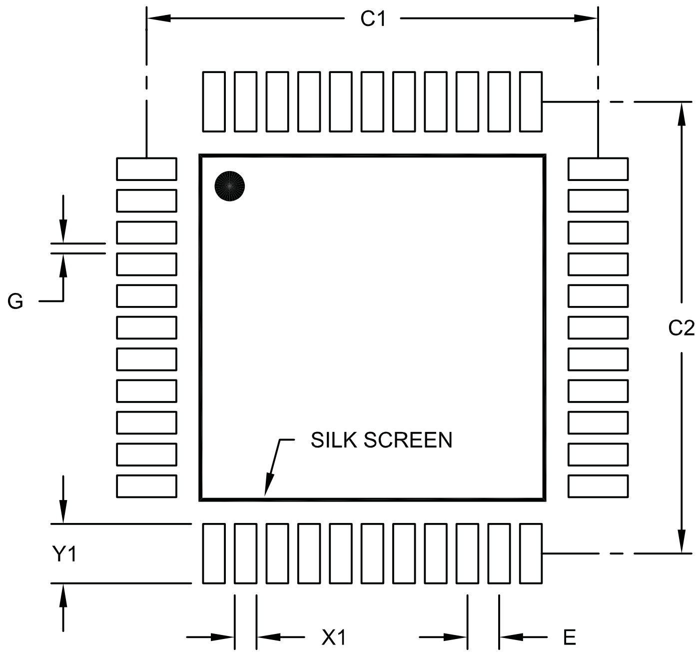
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Chamfers at corners are optional; size may vary.
3. Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-076B

44-Lead Plastic Thin Quad Flatpack (PT) – 10x10x1 mm Body, 2.00 mm [TQFP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

		Units	MILLIMETERS		
		Dimension Limits	MIN	NOM	MAX
Contact Pitch	E			0.80 BSC	
Contact Pad Spacing	C1			11.40	
Contact Pad Spacing	C2			11.40	
Contact Pad Width (X44)	X1				0.55
Contact Pad Length (X44)	Y1				1.50
Distance Between Pads	G		0.25		

Notes:

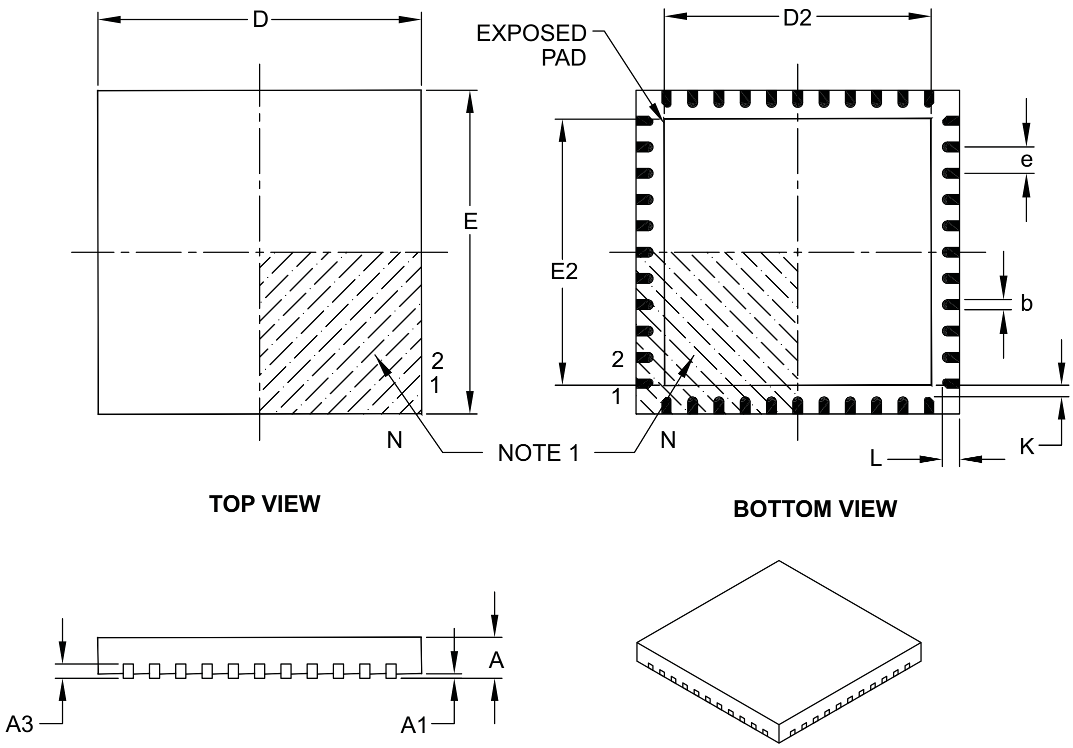
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2076A

44-Lead Plastic Quad Flat, No Lead Package (ML) – 8x8 mm Body [QFN]

Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		44	
Pitch	e		0.65 BSC	
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Contact Thickness	A3		0.20 REF	
Overall Width	E		8.00 BSC	
Exposed Pad Width	E2	6.30	6.45	6.80
Overall Length	D		8.00 BSC	
Exposed Pad Length	D2	6.30	6.45	6.80
Contact Width	b	0.25	0.30	0.38
Contact Length	L	0.30	0.40	0.50
Contact-to-Exposed Pad	K	0.20	–	–

Notes:

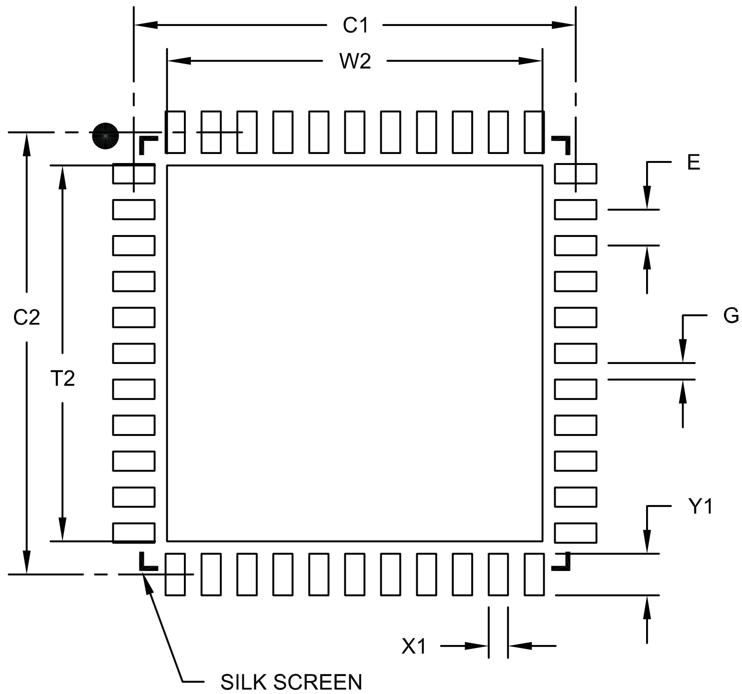
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated.
3. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-103B

44-Lead Plastic Quad Flat, No Lead Package (ML) – 8x8 mm Body [QFN]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

		Units	MILLIMETERS	
	Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.65 BSC	
Optional Center Pad Width	W2			6.80
Optional Center Pad Length	T2			6.80
Contact Pad Spacing	C1		8.00	
Contact Pad Spacing	C2		8.00	
Contact Pad Width (X44)	X1			0.35
Contact Pad Length (X44)	Y1			0.80
Distance Between Pads	G	0.25		

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2103A

APPENDIX A: REVISION HISTORY

Revision A (June 2003)

Original data sheet for PIC18F2331/2431/4331/4431 devices.

Revision B (December 2003)

The Electrical Specifications in **Section 26.0 "Electrical Characteristics"** have been updated and there have been minor corrections to the data sheet text.

Revision C (June 2007)

The data sheet has been updated with all known Data Sheet Errata items and there have been minor corrections made to the data sheet text. Also, the packaging diagrams have been updated in **Section 27.0 "Packaging Information"**.

Revision D (September 2010)

Section 2.0 "Guidelines for Getting Started with PIC18F Microcontrollers" has been updated with more detailed explanations. Changes have been made to the port summary tables in **Section 11.0 "I/O Ports"**. **Section 26.0 "Electrical Characteristics"** has been updated to include extended temperature data. Packaging diagrams have been replaced with new diagrams in **Section 27.0 "Packaging Information"**. There have been minor text edits throughout the document.

APPENDIX B: DEVICE DIFFERENCES

The differences between the devices listed in this data sheet are shown in Table B-1.

TABLE B-1: DEVICE DIFFERENCES

Features	PIC18F2331	PIC18F2431	PIC18F4331	PIC18F4431
Program Memory (Bytes)	4096	8192	4096	8192
Program Memory (Instructions)	2048	4096	2048	4096
Interrupt Sources	22	22	34	34
I/O Ports	Ports A, B, C, D, E	Ports A, B, C, D, E	Ports A, B, C, D, E	Ports A, B, C, D, E
Capture/Compare/PWM Modules	2	2	2	2
Enhanced Capture/Compare/ PWM Modules	1	1	1	1
10-Bit Analog-to-Digital Module	5 Input Channels	5 Input Channels	9 Input Channels	9 Input Channels
Packages	28-Pin SPDIP 28-Pin SOIC 28-Pin QFN	28-Pin SPDIP 28-Pin SOIC 28-Pin QFN	40-Pin PDIP 44-Pin TQFP 44-Pin QFN	40-Pin PDIP 44-Pin TQFP 44-Pin QFN

APPENDIX C: CONVERSION CONSIDERATIONS

This appendix discusses the considerations for converting from previous versions of a device to the ones listed in this data sheet. Typically, these changes are due to the differences in the process technology used. An example of this type of conversion is from a PIC16C74A to a PIC16C74B.

Not Applicable

APPENDIX D: MIGRATION FROM BASELINE TO ENHANCED DEVICES

This section discusses how to migrate from a baseline device (i.e., PIC16C5X) to an enhanced MCU device (i.e., PIC18FXXX).

The following are the list of modifications over the PIC16C5X microcontroller family:

Not Currently Available

APPENDIX E: MIGRATION FROM MID-RANGE TO ENHANCED DEVICES

A detailed discussion of the differences between the mid-range MCU devices (i.e., PIC16CXXX) and the enhanced devices (i.e., PIC18FXXX) is provided in AN716, “Migrating Designs from PIC16C74A/74B to PIC18F442.” The changes discussed, while device-specific, are generally applicable to all mid-range to enhanced device migrations.

This Application Note is available on Microchip’s web site: www.Microchip.com.

APPENDIX F: MIGRATION FROM HIGH-END TO ENHANCED DEVICES

A detailed discussion of the migration pathway and differences between the high-end MCU devices (i.e., PIC17CXXX) and the enhanced devices (i.e., PIC18FXXX) is provided in AN726, “PIC17CXXX to PIC18FXXX Migration.”

This Application Note is available on Microchip’s web site: www.Microchip.com.

NOTES:

INDEX

D

E

Q

R

S

T

U

W

X

NOTES:

THE MICROCHIP WEB SITE

Microchip provides online support via our WWW site at www.microchip.com. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site 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 web site at www.microchip.com. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support
- Development Systems Information Line

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 web site at: http://support.microchip.com

READER RESPONSE

It is our intention to provide you with the best documentation possible to ensure successful use of your Microchip product. If you wish to provide your comments on organization, clarity, subject matter, and ways in which our documentation can better serve you, please FAX your comments to the Technical Publications Manager at (480) 792-4150.

Please list the following information, and use this outline to provide us with your comments about this document.

TO: Technical Publications Manager

Total Pages Sent ________

RE: Reader Response

From: Name ____

Company ____

Address ____

City / State / ZIP / Country ____

Telephone: (_______) _________ - _________

FAX: (______) _________ - _________

Application (optional):

Would you like a reply? ___Y ___N

Device: PIC18F2331/2431/4331/4431

Literature Number: DS39616D

Questions:

1. What are the best features of this document?

____

____

2. How does this document meet your hardware and software development needs?

____

____

3. Do you find the organization of this document easy to follow? If not, why?

____

____

4. What additions to the document do you think would enhance the structure and subject?

____

____

5. What deletions from the document could be made without affecting the overall usefulness?

____

____

6. Is there any incorrect or misleading information (what and where)?

____

____

7. How would you improve this document?

____

____

PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.

PART NO.	X	/XX	XXX
Device	Temperature Range	Package	Pattern

Device	PIC18F2331/2431/4331/4431[1], PIC18F2331/2431/4331/4431T[1,2]; VDD range 4.2V to 5.5V PIC18LF2331/2431/4331/4431[1], PIC18LF2331/2431/4331/44310T[1,2]; VDD range 2.0V to 5.5V
Temperature Range	I = -40°C to +85°C (Industrial) E = -40°C to +125°C (Extended)
Package	PT = TQFP (Thin Quad Flatpack) SO = SOIC SP = Skinny Plastic DIP P = PDIP ML = QFN
Pattern	QTP, SQTP, Code or Special Requirements (blank otherwise)

Examples:

a) PIC18LF4431-I/P 301 = Industrial temp., PDIP package, Extended VDD limits, QTP pattern #301.

b) PIC18LF2331-I/SO = Industrial temp., SOIC package, Extended VDD limits.

c) PIC18F4331-I/P = Industrial temp., PDIP package, normal VDD limits.

Note 1: F = Standard Voltage Range
LF = Wide Voltage Range

2: T = in Tape and Reel – SOIC and TQFP Packages only.

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://support.microchip.com
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Cleveland
Independence, OH
Tel: 216-447-0464
Fax: 216-447-0643

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Farmington Hills, MI
Tel: 248-538-2250
Fax: 248-538-2260

Kokomo
Kokomo, IN
Tel: 765-864-8360
Fax: 765-864-8387

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

Santa Clara
Santa Clara, CA
Tel: 408-961-6444
Fax: 408-961-6445

Toronto
Mississauga, Ontario,
Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8528-2100
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

China - Chongqing
Tel: 86-23-8980-9588
Fax: 86-23-8980-9500

China - Hong Kong SAR
Tel: 852-2401-1200
Fax: 852-2401-3431

China - Nanjing
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

China - Qingdao
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

China - Shanghai
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

China - Shenyang
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

China - Shenzhen
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

China - Wuhan
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

China - Xian
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

China - Xiamen
Tel: 86-592-2388138
Fax: 86-592-2388130

China - Zhuhai
Tel: 86-756-3210040
Fax: 86-756-3210049

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

India - Pune
Tel: 91-20-2566-1512
Fax: 91-20-2566-1513

Japan - Yokohama
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Malaysia - Kuala Lumpur
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

Malaysia - Penang
Tel: 60-4-227-8870
Fax: 60-4-227-4068

Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Hsin Chu
Tel: 886-3-6578-300
Fax: 886-3-6578-370

Taiwan - Kaohsiung
Tel: 886-7-213-7830
Fax: 886-7-330-9305

Taiwan - Taipei
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

UK - Wokingham
Tel: 44-118-921-5869
Fax: 44-118-921-5820

08/04/10