PIC24FJ128GA Family Data Sheet

General Purpose, 16-Bit Flash Microcontrollers

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.

Trademarks

QUALITY MANAGEMENT SYSTEM

Microchip received ISO/TS-16949:2002 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona and Mountain View, California in October 2003. The Company's quality system processes and procedures are for its PICmicro® 8-bit MCUs, 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.

PIC24FJ128GA FAMILY

General Purpose, 16-bit Flash Microcontrollers

High-Performance CPU:

- Modified Harvard Architecture
- Up to 16 MIPS operation @ 32 MHz
- 8 MHz internal oscillator:
  - 4x PLL option
  - Multiple divide options
- 17-bit x 17-bit Single-Cycle Hardware Fractional/Integer Multiplier
- 32-bit by 16-bit Hardware Divider
- 16 x 16-bit Working Register Array
- C compiler Optimized Instruction Set Architecture:
  - 76 base instructions
  - Flexible addressing modes
- Linear Program Memory Addressing up to 12 Mbytes
- Linear Data Memory Addressing up to 64 Kbytes
- Two Address Generation Units for separate Read and Write Addressing of Data Memory

Special Microcontroller Features:

- Operating Voltage Range of 2.0V to 3.6V
- Flash Program Memory:
  - 1000 erase/write cycles, typical
  - Flash retention 20 years, typical
- Self-Reprogrammable under Software Control
- Selectable Power Management modes:
  - Sleep, Idle and Alternate Clock modes
- Fail-Safe Clock Monitor operation:
  - Detects clock failure and switches to on-chip, low-power RC oscillator
- On-Chip LDO Regulator
- JTAG Boundary Scan and Programming Support
- Power-on Reset (POR), Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Flexible Watchdog Timer (WDT) with On-Chip, Low-Power RC Oscillator for reliable operation
- In-Circuit Serial Programming™ (ICSP™) and In-Circuit Emulation (ICE) via 2 pins

Analog Features:

- 10-bit, up to 16-channel Analog-to-Digital Converter (A/D):
  - 500 ksps conversion rate
  - Conversion available during Sleep and Idle
- Dual Analog Comparators with Programmable Input/Output Configuration

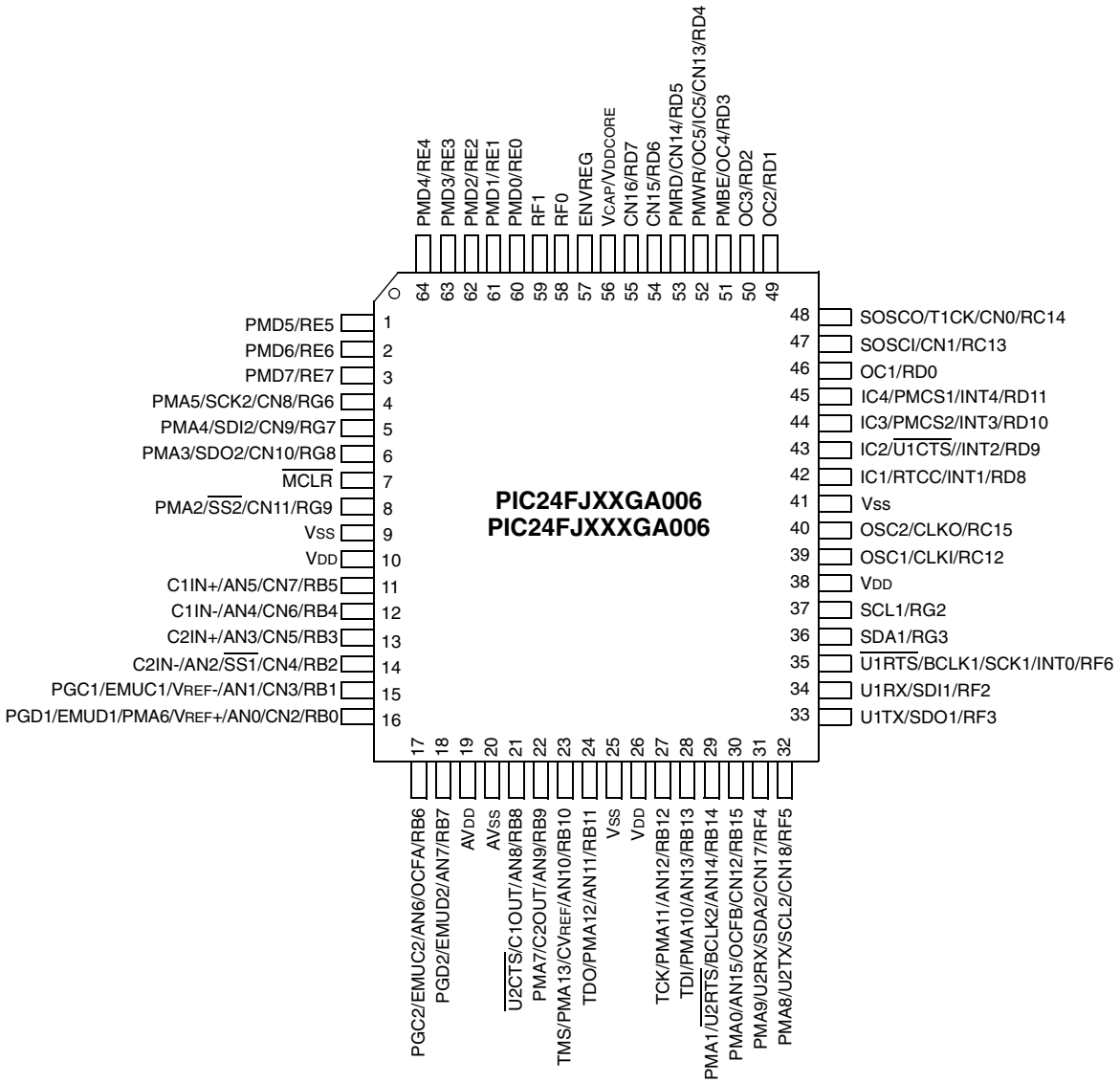
Peripheral Features:

- Two 3-wire/4-wire SPI modules, supporting 4 Frame modes with 4-level FIFO Buffer
- Two I^2C™ modules support Multi-Master/Slave mode and 7-bit/10-bit Addressing
- Two UART modules:
  - Supports RS-232, RS-485 and LIN 1.2
  - Supports IrDA® with on-chip hardware endec
  - Auto-Wake-up on Start bit
  - Auto-Baud Detect
  - 4-level FIFO buffer
- Parallel Master Slave Port (PMP/PSP):
  - Supports 8-bit or 16-bit data
  - Supports 16 address lines
- Hardware Real-Time Clock/Calendar (RTCC):
  - Provides clock, calendar and alarm functions
- Five 16-bit Timers/Counters with Programmable prescaler
- Five 16-bit Capture Inputs
- Five 16-bit Compare/PWM Outputs
- High-Current Sink/Source on select I/O pins: 18 mA/18 mA
- Configurable Open-Drain Output on Digital I/O pins
- Up to 5 External Interrupt Sources

Device	Pins	Program Memory (Bytes)	SRAM (Bytes)	Timers 16-bit	Capture Input	Compare/ PWM Output	UART	SPI	I^2C™	10-bit A/D (ch)	Comparators	PMP/PSP	JTAG
PIC24FJ64GA006	64	64K	8K	5	5	5	2	2	2	16	2	Y	Y
PIC24FJ96GA006	64	96K	8K	5	5	5	2	2	2	16	2	Y	Y
PIC24FJ128GA006	64	128K	8K	5	5	5	2	2	2	16	2	Y	Y
PIC24FJ64GA008	80	64K	8K	5	5	5	2	2	2	16	2	Y	Y
PIC24FJ96GA008	80	96K	8K	5	5	5	2	2	2	16	2	Y	Y
PIC24FJ128GA008	80	128K	8K	5	5	5	2	2	2	16	2	Y	Y
PIC24FJ64GA010	100	64K	8K	5	5	5	2	2	2	16	2	Y	Y
PIC24FJ96GA010	100	96K	8K	5	5	5	2	2	2	16	2	Y	Y
PIC24FJ128GA010	100	128K	8K	5	5	5	2	2	2	16	2	Y	Y

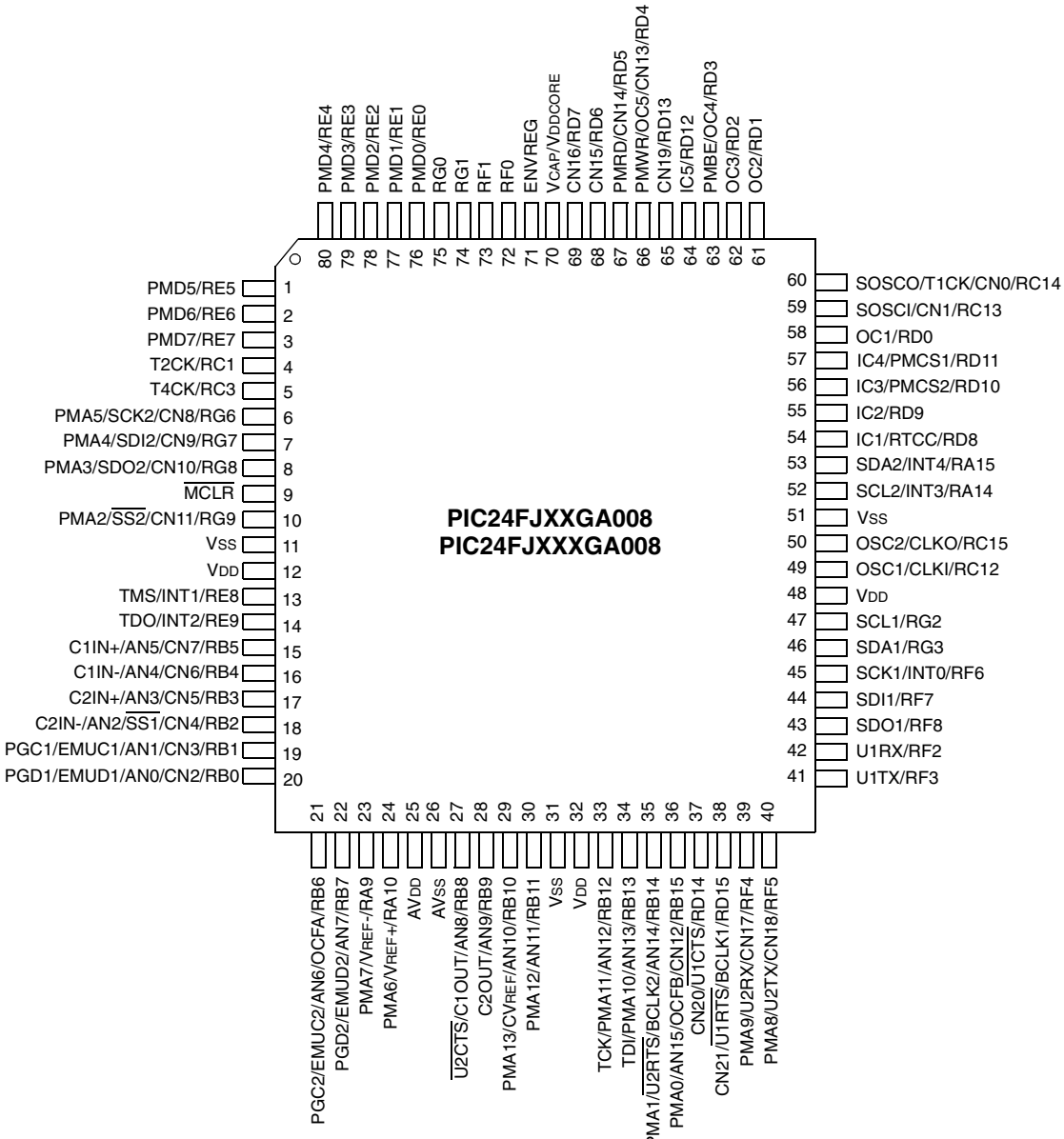
Pin Diagrams

64-Pin TQFP



Pin Diagrams (Continued)

80-Pin TQFP



Pin Diagrams (Continued))

100-Pin TQFP

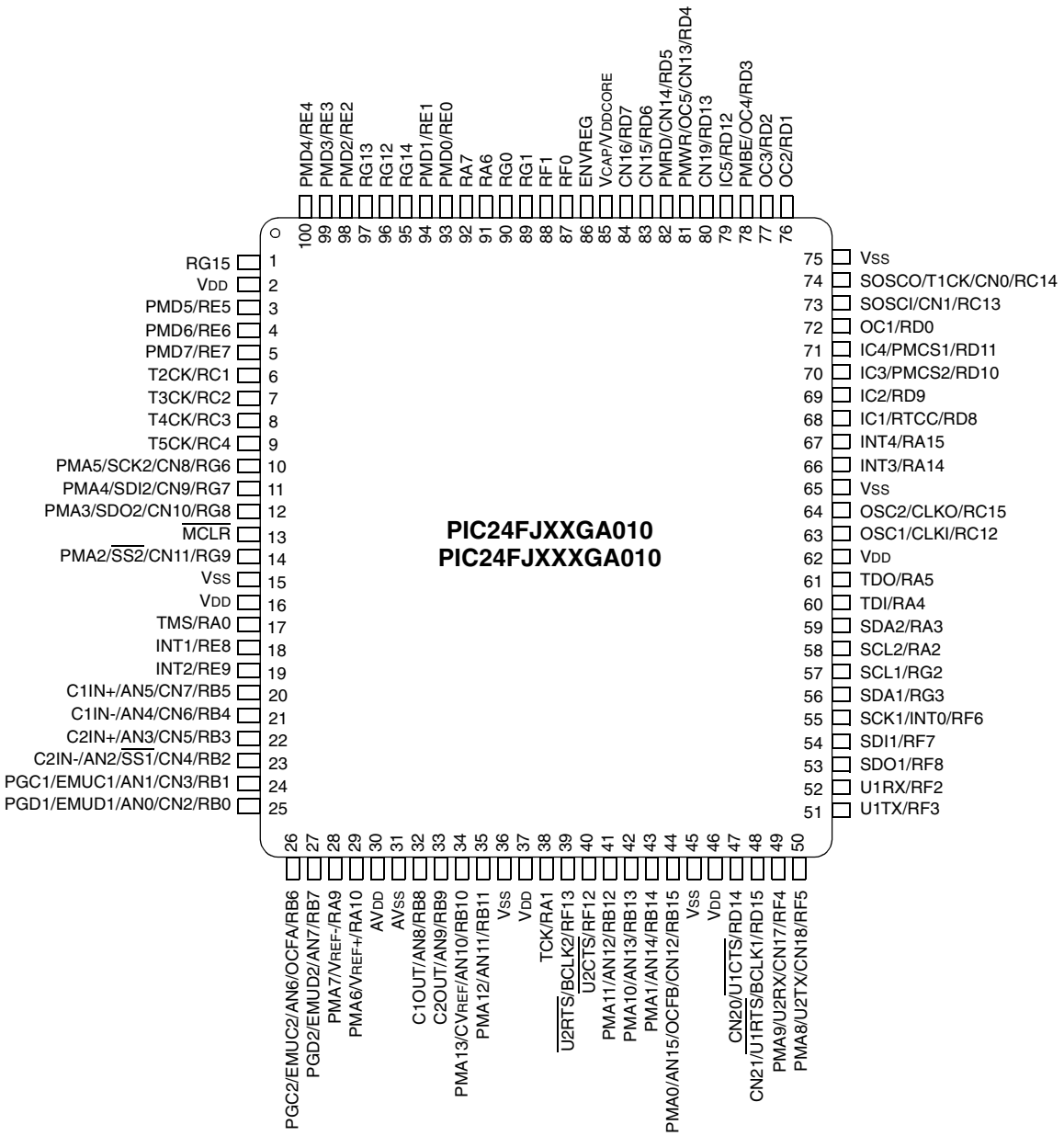


Table of Contents

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.

1.0 DEVICE OVERVIEW

This document contains device specific information for the following devices:

- PIC24FJ64GA006
- PIC24FJ64GA008
- PIC24FJ64GA010
- PIC24FJ96GA006
- PIC24FJ96GA008
- PIC24FJ96GA010
- PIC24FJ128GA006
- PIC24FJ128GA008
- PIC24FJ128GA010

This family introduces a new line of Microchip devices: a 16-bit RISC microcontroller family with a broad peripheral feature set and enhanced computational performance. The PIC24FJ128GA family offers a new migration option for those high-performance applications which may be outgrowing their 8-bit platforms, but don't require the numerical processing power of a digital signal processor.

1.1 Core Features

1.1.1 16-BIT ARCHITECTURE

Central to all PIC24 devices is the 16-bit modified Harvard architecture, first introduced with Microchip's dsPIC® digital signal controllers. The PIC24 CPU core offers a wide range of enhancements, such as:

- 16-bit data and 24-bit address paths, with the ability to move information between data and memory spaces
- Linear addressing of up to 8 Mbytes (program space) and 64 Kbytes (data)
- A 16-element working register array with built-in software stack support
- A 17 x 17 hardware multiplier with support for integer math
- Hardware support for 32 by 16-bit division
- An instruction set that supports multiple addressing modes and is optimized for high-level languages such as 'C'
- Operational performance up to 16 MIPS

1.1.2 POWER-SAVING TECHNOLOGY

All of the devices in the PIC24FJ128GA family incorporate a range of features that can significantly reduce power consumption during operation. Key items include:

- **On-the-Fly Clock Switching:** The device clock can be changed under software control to the Timer1 source or the internal low-power RC oscillator during operation, allowing the user to incorporate power-saving ideas into their software designs.
- **Doze Mode Operation:** When timing-sensitive applications, such as serial communications, require the uninterrupted operation of peripherals, the CPU clock speed can be selectively reduced, allowing incremental power savings without missing a beat.
- **Instruction-Based Power-Saving Modes:** The microcontroller can suspend all operations, or selectively shut down its core while leaving its peripherals active, with a single instruction in software.

1.1.3 OSCILLATOR OPTIONS AND FEATURES

All of the devices in the PIC24FJ128GA family offer five different oscillator options, allowing users a range of choices in developing application hardware. These include:

- Two Crystal modes, using crystals or ceramic resonators.
- Two External Clock modes, offering the option of a divide-by-2 clock output.
- A Fast Internal Oscillator (FRC) with a nominal 8 MHz output, which can also be divided under software control to provide clock speeds as low as 31 kHz.
- A Phase Lock Loop (PLL) frequency multiplier, available to the external oscillator modes and the FRC oscillator, which allows clock speeds of up to 32 MHz.
- A separate internal RC oscillator (LPRC) with a fixed 31 kHz output, which provides a low-power option for timing-insensitive applications.

The internal oscillator block also provides a stable reference source for the Fail-Safe Clock Monitor. This option constantly monitors the main clock source against a reference signal provided by the internal oscillator and enables the controller to switch to the internal oscillator, allowing for continued low-speed operation or a safe application shutdown.

1.1.4 EASY MIGRATION

Regardless of the memory size, all devices share the same rich set of peripherals, allowing for a smooth migration path as applications grow and evolve.

The consistent pinout scheme used throughout the entire family also aids in migrating to the next larger device. This is true when moving between devices with the same pin count, or even jumping from 64-pin to 80-pin to 100-pin devices.

The PIC24 family is pin-compatible with devices in the dsPIC33 family, and shares some compatibility with the pinout schema for PIC18 and dsPIC30. This extends the ability of applications to grow from the relatively simple to the powerful and complex, yet still select a Microchip device.

1.2 Other Special Features

- **Communications:** The PIC24FJ128GA family incorporates a range of serial communication peripherals to handle a range of application requirements. All devices are equipped with two independent UARTs with built-in IrDA encoder/decoders. There are also two independent SPI modules, and two independent I^2C modules that support both Master and Slave modes of operation.
- **Parallel Master/Enhanced Parallel Slave Port:** One of the general purpose I/O ports can be reconfigured for enhanced parallel data communications. In this mode, the port can be configured for both master and slave operations, and supports 8-bit and 16-bit data transfers with up to 16 external address lines in Master modes.
- **Real-Time Clock/Calendar:** This module implements a full-featured clock and calendar with alarm functions in hardware, freeing up timer resources and program memory space for use of the core application.
- **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, as well as faster sampling speeds.

1.3 Details on Individual Family Members

Devices in the PIC24FJ128GA family are available in 64-pin, 80-pin and 100-pin packages. The general block diagram for all devices is shown in Figure 1-1.

The devices are differentiated from each other in two ways:

1. Flash program memory (64 Kbytes for PIC24FJ64GA devices, 96 Kbytes for PIC24FJ96GA devices and 128 Kbytes for PIC24FJ128GA devices).
2. Available I/O pins and ports (53 pins on 6 ports for 64-pin devices, 69 pins on 7 ports for 80-pin devices and 84 pins on 7 ports for 100-pin devices).

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

A list of the pin features available on the PIC24FJ128GA family devices, sorted by function, is shown in Table 1-2. Note that this table shows the pin location of individual peripheral features and not how they are multiplexed on the same pin. This information is provided in the pinout diagrams in the beginning of the data sheet. Multiplexed features are sorted by the priority given to a feature, with the highest priority peripheral being listed first.

TABLE 1-1: DEVICE FEATURES FOR THE PIC24FJ128GA FAMILY

Features	PIC24FJ64GA006	PIC24FJ96GA006	PIC24FJ128GA006	PIC24FJ64GA008	PIC24FJ96GA008	PIC24FJ128GA008	PIC24FJ64GA010	PIC24FJ96GA010	PIC24FJ128GA010
Operating Frequency	DC – 32 MHz								
Program Memory (Bytes)	64K	96K	128K	64K	96K	128K	64K	96K	128K
Program Memory (Instructions)	22,016	32,768	44,032	22,016	32,768	44,032	22,016	32,768	44,032
Data Memory (Bytes)	8192								
Interrupt Sources (Soft Vectors/NMI Traps)	43 (39/4)								
I/O Ports	Ports B, C, D, E, F, G			Ports A, B, C, D, E, F, G			Ports A, B, C, D, E, F, G		
Total I/O Pins	53			69			84		
Timers:									
Total number (16-bit)	5								
32-bit (from paired 16-bit timers)	2								
Input Capture Channels	5								
Output Compare/PWM Channels	5								
Input Change Notification Interrupt	19			22					
Serial Communications:									
Enhanced UART	2								
SPI (3-wire/4-wire)	2								
I^2C™	2								
Parallel Communications (PMP/PSP)	Yes								
JTAG Boundary Scan	Yes								
10-bit Analog-to-Digital Module (input channels)	16								
Analog Comparators	2								
Resets (and Delays)	POR, BOR, RESET Instruction, $\overline{MCLR}$, WDT; Illegal Opcode, Repeat Hardware Traps, (PWRT, OST, PLL Lock)								
Instruction Set	76 Base Instructions, Multiple Addressing Mode Variations								
Packages	64-pin TQFP			80-pin TQFP			100-pin TQFP		

FIGURE 1-1: PIC24FJ128GA FAMILY GENERAL BLOCK DIAGRAM

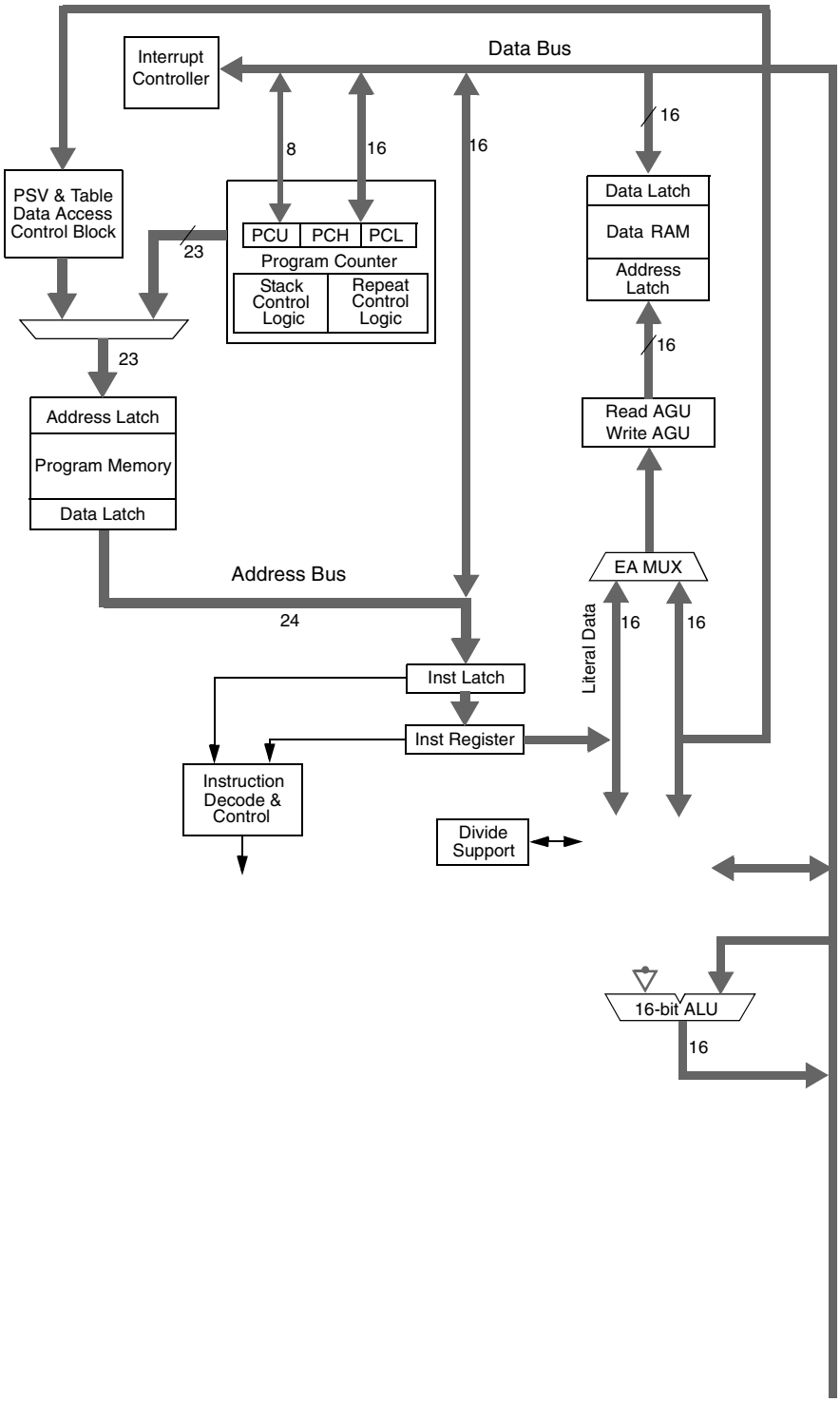


TABLE 1-2: PIC24FJ128GA FAMILY PINOUT DESCRIPTIONS

Function	Pin Number			I/O	Input Buffer	Description
	64-pin	80-pin	100-pin			
AN0	16	20	25	I	ANA	A/D Analog Inputs.
AN1	15	19	24	I	ANA	
AN2	14	18	23	I	ANA	
AN3	13	17	22	I	ANA	
AN4	12	16	21	I	ANA	
AN5	11	15	20	I	ANA	
AN6	17	21	26	I	ANA	
AN7	18	22	27	I	ANA	
AN8	21	27	32	I	ANA	
AN9	22	28	33	I	ANA	
AN10	23	29	34	I	ANA	
AN11	24	30	35	I	ANA	
AN12	27	33	41	I	ANA	
AN13	28	34	42	I	ANA	
AN14	29	35	43	I	ANA	
AN15	30	36	44	I	ANA	
AVDD	19	25	30	P	—	Positive Supply for Analog Modules.
AVSS	20	26	31	P	—	Ground Reference for Analog Modules.
BCLK1	35	38	48	O	—	UART1 IrDA® Baud Clock.
BCLK2	29	35	39	O	—	UART2 IrDA® Baud Clock.
C1IN-	12	16	21	I	ANA	Comparator 1 Negative Input.
C1IN+	11	15	20	I	ANA	Comparator 1 Positive Input.
C1OUT	21	27	32	O	—	Comparator 1 Output.
C2IN-	14	18	23	I	ANA	Comparator 2 Negative Input.
C2IN+	13	17	22	I	ANA	Comparator 2 Positive Input.
C2OUT	22	28	33	O	—	Comparator 2 Output.
CLKI	39	49	63	I	ANA	Main Clock Input Connection.
CLKO	40	50	64	O	—	System Clock Output.
CN0	48	60	74	I	ST	Interrupt-on-Change Inputs.
CN1	47	59	73	I	ST	
CN2	16	20	25	I	ST	
CN3	15	19	24	I	ST	
CN4	14	18	23	I	ST	
CN5	13	17	22	I	ST	
CN6	12	16	21	I	ST	
CN7	11	15	20	I	ST	
CN8	4	6	10	I	ST	
CN9	5	7	11	I	ST	
CN10	6	8	12	I	ST	
CN11	8	10	14	I	ST	
CN12	30	36	44	I	ST	
CN13	52	66	81	I	ST	
CN14	53	67	82	I	ST	
CN15	54	68	83	I	ST	
CN16	55	69	84	I	ST	
CN17	31	39	49	I	ST	

Legend: TTL = TTL input buffer
ANA = Analog level input/output
ST = Schmitt Trigger input buffer
I^2C™ = I^2C/SMBus input buffer

TABLE 1-2: PIC24FJ128GA FAMILY PINOUT DESCRIPTIONS (CONTINUED)

Function	Pin Number			I/O	Input Buffer	Description
	64-pin	80-pin	100-pin			
CN18	32	40	50	I	ST	Interrupt-on-Change Inputs.
CN19	—	65	80	I	ST	
CN20	—	37	47	I	ST	
CN21	—	38	48	I	ST	
CVREF	23	29	34	O	ANA	Comparator Voltage Reference Output.
EMUC1	15	19	24	I/O	ST	In-Circuit Emulator Clock Input/Output.
EMUD1	16	20	25	I/O	ST	In-Circuit Emulator Data Input/Output.
EMUC2	17	21	26	I/O	ST	In-Circuit Emulator Clock Input/Output.
EMUD2	18	22	27	I/O	ST	In-Circuit Emulator Data Input/Output.
ENVREG	57	71	86	I	ST	Enable for On-Chip Voltage Regulator.
IC1	42	54	68	I	ST	Input Capture Inputs.
IC2	43	55	69	I	ST	
IC3	44	56	70	I	ST	
IC4	45	57	71	I	ST	
IC5	52	64	79	I	ST	
INT0	35	45	55	I	ST	External Interrupt Inputs.
INT1	42	13	18	I	ST	
INT2	43	14	19	I	ST	
INT3	44	52	66	I	ST	
INT4	45	53	67	I	ST	
$\overline{MCLR}$	7	9	13	I	ST	Master Clear (Device Reset) Input. This line is brought low to cause a Reset.
OC1	46	58	72	O	—	Output Compare/PWM Outputs.
OC2	49	61	76	O	—	
OC3	50	62	77	O	—	
OC4	51	63	78	O	—	
OC5	52	66	81	O	—	
OCFA	17	21	26	I	ST	Output Compare Fault A Input.
OCFB	30	36	44	I	ST	Output Compare Fault B Input.
OSC1	39	49	63	I	ANA	Main Oscillator Input Connection.
OSC2	40	50	64	O	ANA	Main Oscillator Output Connection.
PGC1	15	19	24	I/O	ST	In-Circuit Debugger and ICSP™ Programming Clock
PGD1	16	20	25	I/O	ST	In-Circuit Debugger and ICSP Programming Data.
PGC2	17	21	26	I/O	ST	In-Circuit Debugger and ICSP™ Programming Clock.
PGD2	18	22	27	I/O	ST	In-Circuit Debugger and ICSP Programming Data.

Legend: TTL = TTL input buffer
ANA = Analog level input/output
ST = Schmitt Trigger input buffer
I^2C™ = I^2C/SMBus input buffer

TABLE 1-2: PIC24FJ128GA FAMILY PINOUT DESCRIPTIONS (CONTINUED)

Function	Pin Number			I/O	Input Buffer	Description
	64-pin	80-pin	100-pin			
PMA0	30	36	44	I/O	ST	Parallel Master Port Address Bit 0 Input (Buffered Slave modes) and Output (Master modes).
PMA1	29	35	43	I/O	ST	Parallel Master Port Address Bit 1 Input (Buffered Slave modes) and Output (Master modes).
PMA2	8	10	14	O	—	Parallel Master Port Address (Demultiplexed Master modes).
PMA3	6	8	12	O	—	
PMA4	5	7	11	O	—	
PMA5	4	6	10	O	—	
PMA6	16	24	29	O	—	
PMA7	22	23	28	O	—	
PMA8	32	40	50	O	—	
PMA9	31	39	49	O	—	
PMA10	28	34	42	O	—	
PMA11	27	33	41	O	—	
PMA12	24	30	35	O	—	
PMA13	23	29	34	O	—	
PMBE	51	63	78	O	—	Parallel Master Port Byte Enable Strobe.
PMCS1	45	57	71	O	—	Parallel Master Port Chip Select 1 Strobe/Address bit 14.
PMCS2	44	56	70	O	—	Parallel Master Port Chip Select 2 Strobe/Address bit 15.
PMD0	60	76	93	I/O	ST	Parallel Master Port Data (Demultiplexed Master mode) or Address/Data (Multiplexed Master modes).
PMD1	61	77	94	I/O	ST	
PMD2	62	78	98	I/O	ST	
PMD3	63	79	99	I/O	ST	
PMD4	64	80	100	I/O	ST	
PMD5	1	1	3	I/O	ST	
PMD6	2	2	4	I/O	ST	
PMD7	3	3	5	I/O	ST	
PMRD	53	67	82	O	—	Parallel Master Port Read Strobe.
PMWR	52	66	81	O	—	Parallel Master Port Write Strobe.

Legend: TTL = TTL input buffer
ANA = Analog level input/output
ST = Schmitt Trigger input buffer
I^2C™ = I^2C/SMBus input buffer

TABLE 1-2: PIC24FJ128GA FAMILY PINOUT DESCRIPTIONS (CONTINUED)

Function	Pin Number			I/O	Input Buffer	Description
	64-pin	80-pin	100-pin			
RA0	—	—	17	I/O	ST	PORTA Digital I/O.
RA1	—	—	38	I/O	ST	
RA2	—	—	58	I/O	ST	
RA3	—	—	59	I/O	ST	
RA4	—	—	60	I/O	ST	
RA5	—	—	61	I/O	ST	
RA6	—	—	91	I/O	ST	
RA7	—	—	92	I/O	ST	
RA9	—	23	28	I/O	ST	
RA10	—	24	29	I/O	ST	
RA14	—	52	66	I/O	ST	
RA15	—	53	67	I/O	ST	
RB0	16	20	25	I/O	ST	PORTB Digital I/O.
RB1	15	19	24	I/O	ST	
RB2	14	18	23	I/O	ST	
RB3	13	17	22	I/O	ST	
RB4	12	16	21	I/O	ST	
RB5	11	15	20	I/O	ST	
RB6	17	21	26	I/O	ST	
RB7	18	22	27	I/O	ST	
RB8	21	27	32	I/O	ST	
RB9	22	28	33	I/O	ST	
RB10	23	29	34	I/O	ST	
RB11	24	30	35	I/O	ST	
RB12	27	33	41	I/O	ST	
RB13	28	34	42	I/O	ST	
RB14	29	35	43	I/O	ST	
RB15	30	36	44	I/O	ST	
RC1	—	4	6	I/O	ST	PORTC Digital I/O.
RC2	—	—	7	I/O	ST	
RC3	—	5	8	I/O	ST	
RC4	—	—	9	I/O	ST	
RC12	39	49	63	I/O	ST	
RC13	47	59	73	I/O	ST	
RC14	48	60	74	I/O	ST	
RC15	40	50	64	I/O	ST	

Legend: TTL = TTL input buffer
ANA = Analog level input/output
ST = Schmitt Trigger input buffer
I^2C™ = I^2C/SMBus input buffer

TABLE 1-2: PIC24FJ128GA FAMILY PINOUT DESCRIPTIONS (CONTINUED)

Function	Pin Number			I/O	Input Buffer	Description
	64-pin	80-pin	100-pin			
RD0	46	58	72	I/O	ST	PORTD Digital I/O.
RD1	49	61	76	I/O	ST	
RD2	50	62	77	I/O	ST	
RD3	51	63	78	I/O	ST	
RD4	52	66	81	I/O	ST	
RD5	53	67	82	I/O	ST	
RD6	54	68	83	I/O	ST	
RD7	55	69	84	I/O	ST	
RD8	42	54	68	I/O	ST	
RD9	43	55	69	I/O	ST	
RD10	44	56	70	I/O	ST	
RD11	45	57	71	I/O	ST	
RD12	—	64	79	I/O	ST	
RD13	—	65	80	I/O	ST	
RD14	—	37	47	I/O	ST	
RD15	—	38	48	I/O	ST	
RE0	60	76	93	I/O	ST	PORTE Digital I/O.
RE1	61	77	94	I/O	ST	
RE2	62	78	98	I/O	ST	
RE3	63	79	99	I/O	ST	
RE4	64	80	100	I/O	ST	
RE5	1	1	3	I/O	ST	
RE6	2	2	4	I/O	ST	
RE7	3	3	5	I/O	ST	
RE8	—	13	18	I/O	ST	
RE9	—	14	19	I/O	ST	
RF0	58	72	87	I/O	ST	PORTF Digital I/O.
RF1	59	73	88	I/O	ST	
RF2	34	42	52	I/O	ST	
RF3	33	41	51	I/O	ST	
RF4	31	39	49	I/O	ST	
RF5	32	40	50	I/O	ST	
RF6	35	45	55	I/O	ST	
RF7	—	44	54	I/O	ST	
RF8	—	43	53	I/O	ST	
RF12	—	—	40	I/O	ST	
RF13	—	—	39	I/O	ST	

Legend: TTL = TTL input buffer
ANA = Analog level input/output
ST = Schmitt Trigger input buffer
I^2C™ = I^2C/SMBus input buffer

TABLE 1-2: PIC24FJ128GA FAMILY PINOUT DESCRIPTIONS (CONTINUED)

Function	Pin Number			I/O	Input Buffer	Description
	64-pin	80-pin	100-pin			
RG0	—	75	90	I/O	ST	PORTG Digital I/O.
RG1	—	74	89	I/O	ST	
RG2	37	47	57	I/O	ST	
RG3	36	46	56	I/O	ST	
RG6	4	6	10	I/O	ST	
RG7	5	7	11	I/O	ST	
RG8	6	8	12	I/O	ST	
RG9	8	10	14	I/O	ST	
RG12	—	—	96	I/O	ST	
RG13	—	—	97	I/O	ST	
RG14	—	—	95	I/O	ST	
RG15	—	—	1	I/O	ST	
RTCC	42	54	68	O	—	Real-Time Clock Alarm Output.
SCK1	35	45	55	O	—	SPI1 Serial Clock Output.
SCK2	4	6	10	I/O	ST	SPI2 Serial Clock Output.
SCL1	37	47	57	I/O	I^2C	I2C1 Synchronous Serial Clock Input/Output.
SCL2	32	52	58	I/O	I^2C	I2C2 Synchronous Serial Clock Input/Output.
SDA1	36	46	56	I/O	I^2C	I2C1 Data Input/Output.
SDA2	31	53	59	I/O	I^2C	I2C2 Data Input/Output.
SDI1	34	44	54	I	ST	SPI1 Serial Data Input.
SDI2	5	7	11	I	ST	SPI2 Serial Data Input.
SDO1	33	43	53	O	—	SPI1 Serial Data Output.
SDO2	6	8	12	O	—	SPI2 Serial Data Output.
SOSCI	47	59	73	I	ANA	Secondary Oscillator/Timer1 Clock Input.
SOSCO	48	60	74	O	ANA	Secondary Oscillator/Timer1 Clock Output.
$\overline{SS1}$	14	18	23	I/O	ST	Slave Select Input/Frame Select Output (SPI1).
$\overline{SS2}$	8	10	14	I/O	ST	Slave Select Input/Frame Select Output (SPI2).
T1CK	48	60	74	I	ST	Timer1 Clock.
T2CK	—	4	6	I	ST	Timer2 External Clock Input.
T3CK	—	—	7	I	ST	Timer3 External Clock Input.
T4CK	—	5	8	I	ST	Timer4 External Clock Input.
T5CK	—	—	9	I	ST	Timer5 External Clock Input.
TCK	27	33	38	I	ST	JTAG Test Clock/Programming Clock Input.
TDI	28	34	60	I	ST	JTAG Test Data/Programming Data Input.
TDO	24	14	61	O	—	JTAG Test Data Output.
TMS	23	13	17	I	ST	JTAG Test Mode Select Input.

Legend: TTL = TTL input buffer ST = Schmitt Trigger input buffer
ANA = Analog level input/output I^2C™ = I^2C/SMBus input buffer

TABLE 1-2: PIC24FJ128GA FAMILY PINOUT DESCRIPTIONS (CONTINUED)

Function	Pin Number			I/O	Input Buffer	Description
	64-pin	80-pin	100-pin			
$\overline{\text{U1CTS}}$	43	37	47	I	ST	UART1 Clear to Send Input.
$\overline{\text{U1RTS}}$	35	38	48	O	—	UART1 Request to Send Output.
U1RX	34	42	52	I	ST	UART1 Receive.
U1TX	33	41	51	O	DIG	UART1 Transmit Output.
$\overline{\text{U2CTS}}$	21	27	40	I	ST	UART2 Clear to Send Input.
$\overline{\text{U2RTS}}$	29	35	39	O	—	UART2 Request to Send Output.
U2RX	31	39	49	I	ST	UART 2 Receive Input.
U2TX	32	40	50	O	—	UART2 Transmit Output.
VDD	10, 26, 38	12, 32, 48	2, 16, 37, 46, 62	P	—	Positive Supply for Peripheral Digital Logic and I/O pins.
VDDCAP	56	70	85	P	—	External Filter Capacitor Connection (regulator enabled).
VDDCORE	56	70	85	P	—	Positive Supply for Microcontroller Core Logic (regulator disabled).
VREF-	15	23	28	I	ANA	A/D and Comparator Reference Voltage (Low) Input.
VREF+	16	24	29	I	ANA	A/D and Comparator Reference Voltage (High) Input.
VSS	9, 25, 41	11, 31, 51	15, 36, 45, 65, 75	P	—	Ground Reference for Logic and I/O pins.

Legend: TTL = TTL input buffer
ANA = Analog level input/output
ST = Schmitt Trigger input buffer
I^2C™ = I^2C/SMBus input buffer

NOTES:

2.0 CPU

The PIC24 CPU has a 16-bit (data) modified Harvard architecture with an enhanced instruction set, and a 23-bit instruction word with a variable length opcode field. The Program Counter (PC) is 24 bits wide and addresses up to 4M instructions of user program memory space. A single-cycle instruction prefetch mechanism is used to help maintain throughput and provides predictable execution. All instructions execute in a single cycle, with the exception of instructions that change the program flow, the double-word move (`MOV.D`) instruction and the table instructions. Overhead-free program loop constructs are supported using the `REPEAT` instructions, which are interruptible at any point.

PIC24 devices have sixteen 16-bit working registers in the programmer's model. Each of the working registers can act as a data, address or address offset register. The 16th working register (W15) operates as a software Stack Pointer for interrupts and calls.

The upper 32 Kbytes of the data space memory map can optionally be mapped into program space at any 16K word boundary defined by the 8-bit Program Space Visibility Page (PSVPAG) register. The program to data space mapping feature lets any instruction access program space as if it were data space.

The Instruction Set Architecture (ISA) has been significantly enhanced beyond that of the PIC18, but maintains an acceptable level of backward compatibility. All PIC18 instructions and addressing modes are supported either directly or through simple macros. Many of the ISA enhancements have been driven by compiler efficiency needs.

The core supports Inherent (no operand), Relative, Literal, Memory Direct and three groups of addressing modes. All modes support Register Direct and various Register Indirect modes. Each group offers up to 7 addressing modes. Instructions are associated with predefined addressing modes depending upon their functional requirements.

For most instructions, the core is capable of executing a data (or program data) memory read, a working register (data) read, a data memory write and a program (instruction) memory read per instruction cycle. As a result, three-parameter instructions can be supported, allowing trinary operations (that is, A + B = C) to be executed in a single cycle.

A high-speed 17-bit by 17-bit multiplier has been included to significantly enhance the core arithmetic capability and throughput. The multiplier supports signed, unsigned and mixed mode 16-bit by 16-bit or 8-bit by 8-bit integer multiplication. All multiply instructions execute in a single cycle.

The 16-bit ALU has been enhanced with integer divide assist hardware that supports an iterative non-restoring divide algorithm. It operates in conjunction with the `REPEAT` instruction looping mechanism, and a selection of iterative divide instructions, to support 32-bit (or 16-bit) divided by 16-bit integer signed and unsigned division. All divide operations require 19 cycles to complete but are interruptible at any cycle boundary.

The PIC24 has a vectored exception scheme with up to 8 sources of non-maskable traps and up to 118 interrupt sources. Each interrupt source can be assigned to one of seven priority levels.

A block diagram of the CPU is shown in Figure 2-1.

2.1 Programmer's Model

The programmer's model for the PIC24 is shown in Figure 2-2. All registers in the programmer's model are memory mapped and can be manipulated directly by instructions. A description of each register is provided in Table 2-1. All registers associated with the programmer's model are memory mapped.

FIGURE 2-1: PIC24 CPU CORE BLOCK DIAGRAM

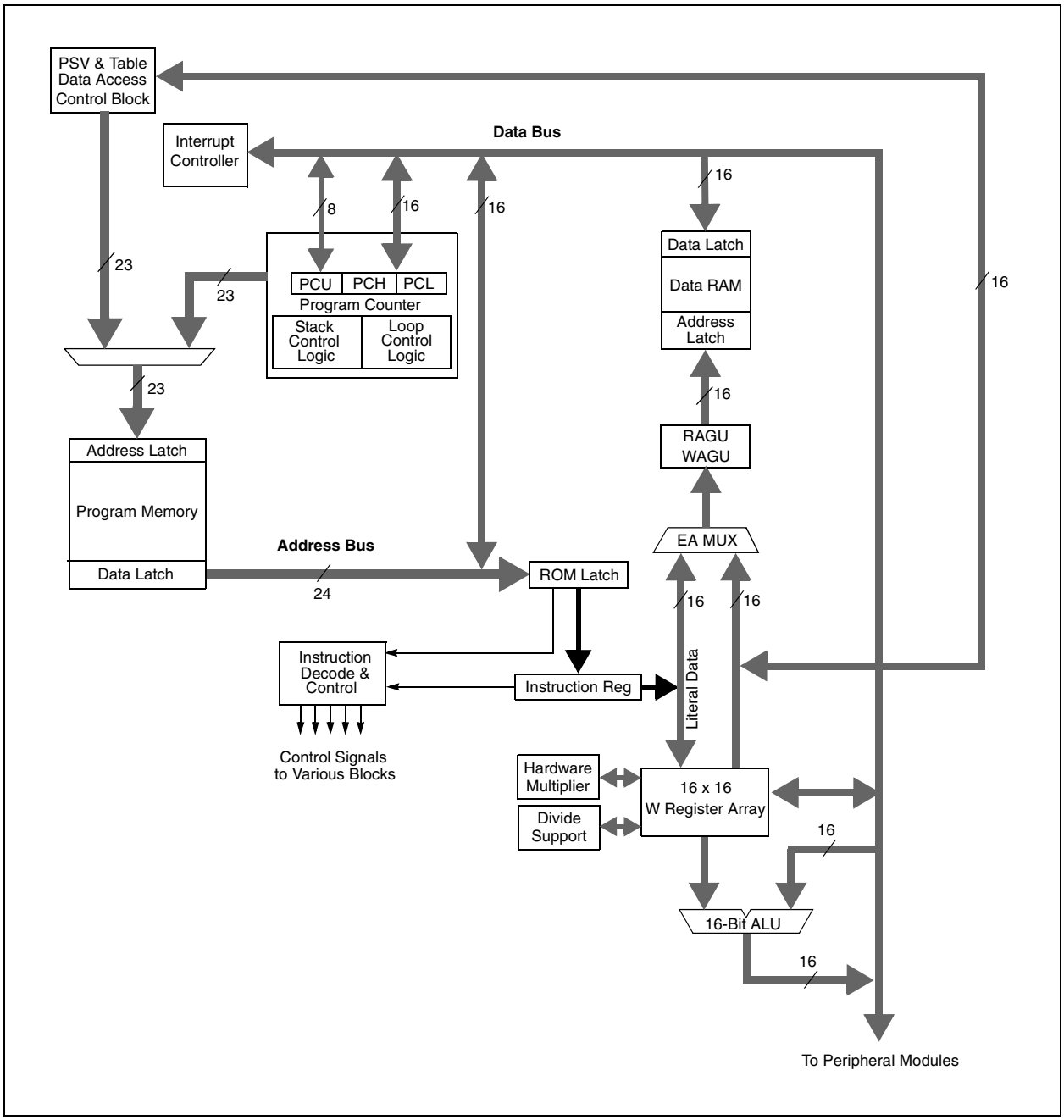
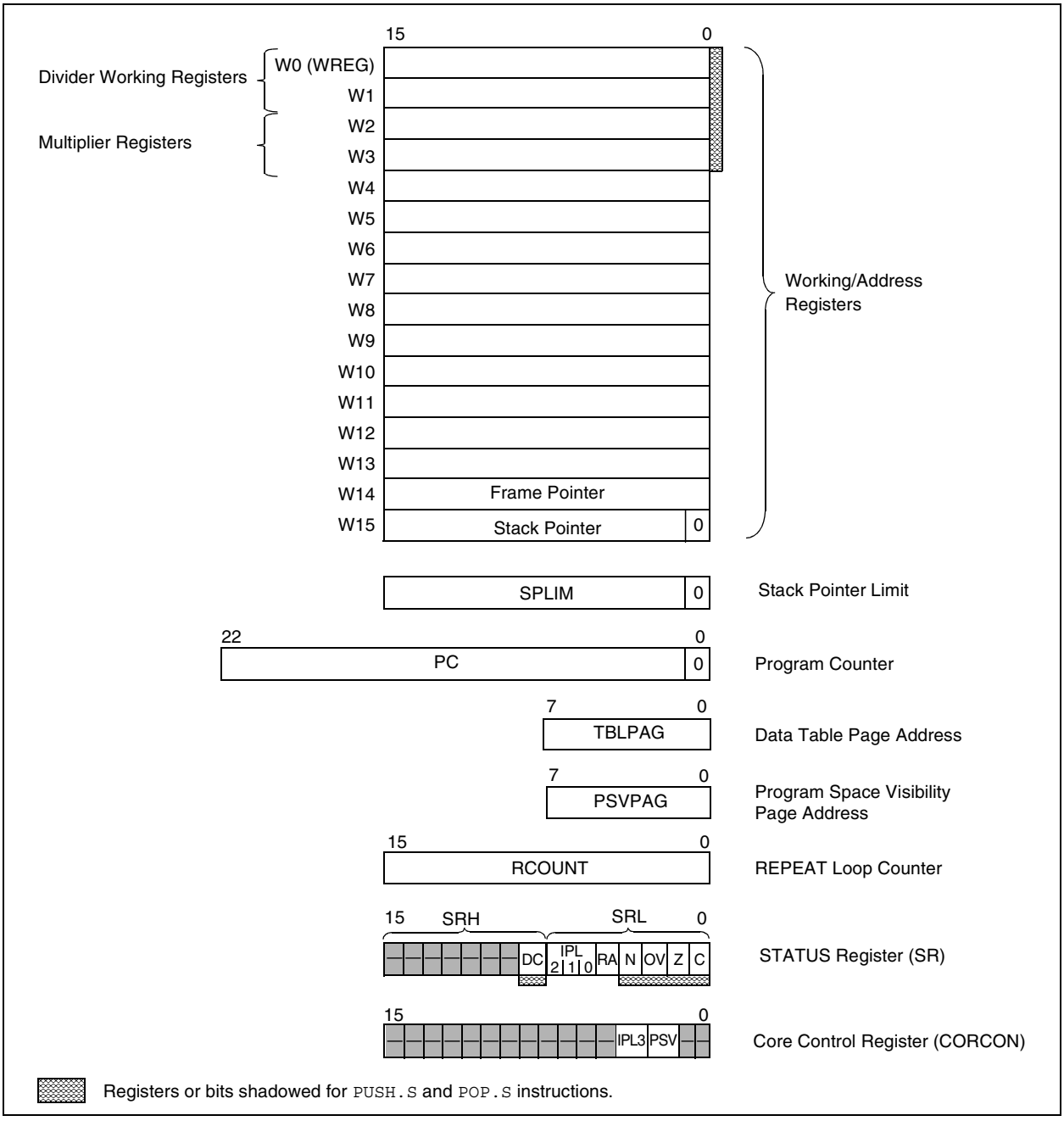


TABLE 2-1: CPU CORE REGISTERS

Register(s) Name	Description
W0 through W15	Working register array
PC	23-bit Program Counter
SR	ALU STATUS register
SPLIM	Stack Pointer Limit Value register
TBLPAG	Table Memory Page Address register
PSVPAG	Program Space Visibility Page Address register
RCOUNT	Repeat Loop Counter register
CORCON	CPU Control Register

FIGURE 2-2: PROGRAMMER'S MODEL



2.2 CPU Control Registers

REGISTER 2-1: SR: CPU STATUS REGISTER

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	U -0	R/W-0
—	—	—	—	—	—	—	DC
bit 15							bit 8

Lower Byte:							
R/W-0(1)	R/W-0(1)	R/W-0(1)	R-0	R/W-0	R/W-0	R/W-0	R/W-0
IPL2(2)	IPL1(2)	IPL0(2)	RA	N	OV	Z	C
bit 7							bit 0

bit 15-9 **Unimplemented:** Read as '0'

bit 8 **DC:** ALU Half Carry/Borrow bit
1 = A carry-out from the 4th low-order bit (for byte sized data) or 8th low-order bit (for word sized data) of the result occurred
0 = No carry-out from the 4th or 8th low-order bit of the result has occurred

bit 7-5 **IPL2:IPL0:** CPU Interrupt Priority Level Status bits(2)
111 = CPU interrupt priority level is 7 (15). User interrupts disabled.
110 = CPU interrupt priority level is 6 (14)
101 = CPU Interrupt Priority Level is 5 (13)
100 = CPU interrupt priority level is 4 (12)
011 = CPU interrupt priority level is 3 (11)
010 = CPU interrupt priority level is 2 (10)
001 = CPU interrupt priority level is 1 (9)
000 = CPU interrupt priority level is 0 (8)

Note 1: The IPL Status bits are read-only when NSTDIS (INTCON1<15>) = 1.

2: The IPL bits are concatenated with the IPL3 bit (CORCON<3>) to form the CPU interrupt priority level. The value in parentheses indicates the IPL when IPL3 = 1.

bit 4 **RA:** REPEAT Loop Active bit
1 = REPEAT loop in progress
0 = REPEAT loop not in progress

bit 3 **N:** ALU Negative bit
1 = Result was negative
0 = Result was non-negative (zero or positive)

bit 2 **OV:** ALU Overflow bit
1 = Overflow occurred for signed (2's complement) arithmetic in this arithmetic operation
0 = No overflow has occurred

bit 1 **Z:** ALU Zero bit
1 = An operation which effects the Z bit has set it at some time in the past
0 = The most recent operation which effects the Z bit has cleared it (i.e., a non-zero result)

bit 0 **C:** ALU Carry/$\overline{\text{Borrow}}$ bit
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

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

REGISTER 2-2: CORCON: CORE CONTROL REGISTER

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
U-0	U-0	U-0	U-0	R/C-0	R/W-0	U-0	U-0
—	—	—	—	IPL3	PSV	—	—
bit 7							bit 0

bit 15-4 **Unimplemented:** Read as '0'

bit 3 **IPL3:** CPU Interrupt Priority Level Status bit
1 = CPU interrupt priority level is greater than 7
0 = CPU interrupt priority level is 7 or less

Note: User interrupts are disabled when IPL3 = 1.

bit 2 **PSV:** Program Space Visibility in Data Space Enable bit
1 = Program space visible in data space
0 = Program space not visible in data space

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

2.3 Arithmetic Logic Unit (ALU)

The PIC24 ALU is 16 bits wide and is capable of addition, subtraction, bit shifts and logic operations. Unless otherwise mentioned, arithmetic operations are 2's complement in nature. Depending on the operation, the ALU may affect the values of the Carry (C), Zero (Z), Negative (N), Overflow (OV) and Digit Carry (DC) Status bits in the SR register. The C and DC Status bits operate as Borrow and Digit Borrow bits, respectively, for subtraction operations.

The ALU can perform 8-bit or 16-bit operations, depending on the mode of the instruction that is used. Data for the ALU operation can come from the W register array, or data memory, depending on the addressing mode of the instruction. Likewise, output data from the ALU can be written to the W register array or a data memory location.

The PIC24 CPU incorporates hardware support for both multiplication and division. This includes a dedicated hardware multiplier and support hardware for 16-bit divisor division.

2.3.1 MULTIPLIER

The ALU contains a high-speed 17-bit x 17-bit multiplier. It supports unsigned, signed or mixed sign operation in several multiplication modes:

1. 16-bit x 16-bit signed
2. 16-bit x 16-bit unsigned
3. 16-bit signed x 5-bit (literal) unsigned
4. 16-bit unsigned x 16-bit unsigned
5. 16-bit unsigned x 5-bit (literal) unsigned
6. 16-bit unsigned x 16-bit signed
7. 8-bit unsigned x 8-bit unsigned

2.3.2 DIVIDER

The divide block supports 32-bit/16-bit and 16-bit/16-bit signed and unsigned integer divide operation with the following data sizes:

1. 32-bit signed/16-bit signed divide
2. 32-bit unsigned/16-bit unsigned divide
3. 16-bit signed/16-bit signed divide
4. 16-bit unsigned/16-bit unsigned divide

The quotient for all divide instructions ends up in W0 and the remainder in W1. 16-bit signed and unsigned DIV instructions can specify any W register for both the 16-bit divisor (Wn) and any W register (aligned) pair (W(m+1):Wm) for the 32-bit dividend. The divide algorithm takes one cycle per bit of divisor, so both 32-bit/16-bit and 16-bit/16-bit instructions take the same number of cycles to execute.

2.3.3 MULTI-BIT SHIFT SUPPORT

The PIC24 ALU supports both single-bit and single-cycle, multi-bit arithmetic and logic shifts. Multi-bit shifts are implemented using a shifter block, capable of performing up to a 15-bit arithmetic right shift, or up to a 15-bit left shift, in a single cycle. All multi-bit shift instructions only support register direct addressing for both the operand source and result destination.

A full summary of instructions that use the shift operation is provided below in Table 2-2.

TABLE 2-2: INSTRUCTIONS THAT USE THE SINGLE AND MULTI-BIT SHIFT OPERATION

Instruction	Description
ASR	Arithmetic shift right source register by one bit.
ASRF	Arithmetic shift right the content of the register by one bit.
ASRW	Arithmetic shift right source register by up to 15 bits, value held in the W register referenced within instruction.
ASRK	Arithmetic shift right source register up to 15 bits. Shift value is literal.
SL	Shift left source register by one bit.
SLF	Shift left the content of the file register by one bit.
SLW	Shift left source register by up to 15 bits, value held in the W register referenced instruction.
SLK	Shift left source register up to 15 bits. Shift value is literal.
LSR	Logical shift right source register by one bit.
LSRF	Logical shift right the content of the register by one bit.
LSRW	Logical shift right source register by up to 15 bits, value held in the W register referenced within instruction.
LSRK	Logical shift right source register up to 15 bits. Shift value is literal.

3.0 MEMORY ORGANIZATION

As Harvard architecture devices, PIC24 microcontrollers feature separate program and data memory spaces and busses. This architecture also allows the direct access of program memory from the data space during code execution.

3.1 Program Address Space

The program address memory space of PIC24FJ128GA family devices is 4M instructions. The space is addressable by a 24-bit value derived from either the 23-bit Program Counter (PC) during program execution, or from table operation or data space remapping, as described in **Section 3.3 "Interfacing Program and Data Memory Spaces"**.

User access to the program memory space is restricted to the lower half of the address range (000000h to 7FFFFFh). The exception is the use of `TBLRD/TBLWT` operations, which use TBLPAG<7> to permit access to the Configuration bits and Device ID sections of the configuration memory space.

3.1.1 PROGRAM MEMORY ORGANIZATION

The program memory space is organized in word addressable blocks. Although it is treated as 24 bits wide, it is more appropriate to think of each address of the program memory as a lower and upper word, with the upper byte of the upper word being unimplemented. The lower word always has an even address, while the upper word has an odd address (Figure 3-2).

Program memory addresses are always word-aligned on the lower word, and addresses are incremented or decremented by two during code execution. This arrangement also provides compatibility with data memory space addressing and makes it possible to access data in the program memory space.

3.1.2 HARD MEMORY VECTORS

All PIC24 devices reserve the addresses between 00000h and 000200h for hard coded program execution vectors. A hardware Reset vector is provided to redirect code execution from the default value of the PC on device Reset to the actual start of code. A GOTO instruction is programmed by the user at 000000h, with the actual address for the start of code at 000002h.

PIC24 devices also have two interrupt vector tables, located from 000004h to 0000FFh and 000100h to 0001FFh. These vector tables allow each of the many device interrupt sources to be handled by separate ISRs. A more detailed discussion of the interrupt vector tables is provided in **Section 6.1 "Interrupt Vector Table"**.

3.1.3 FLASH CONFIGURATION WORDS

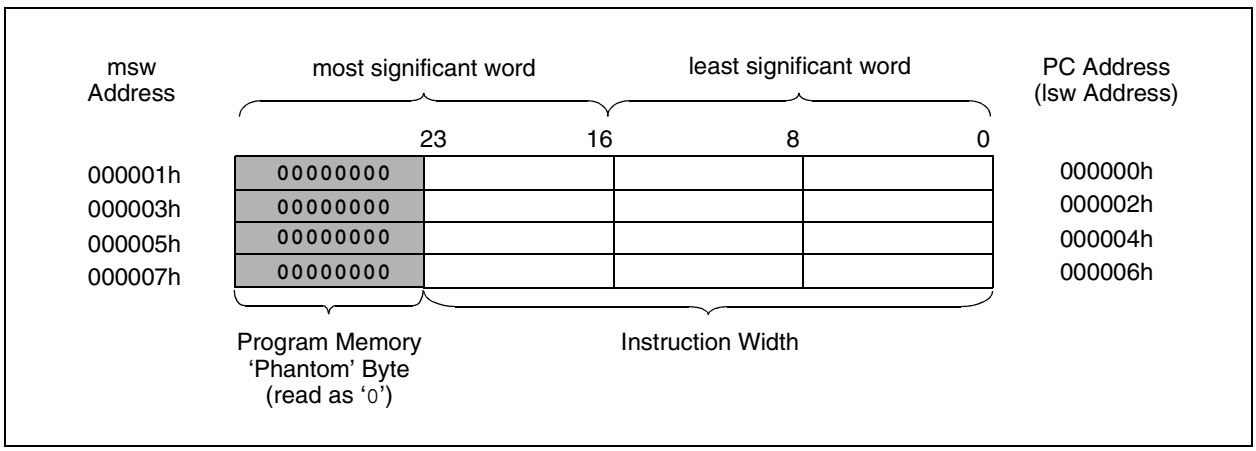
In PIC24FJ128GA family devices, the top two words of on-chip program memory are reserved for configuration information. On device Reset, the configuration information is copied into the appropriate Configuration registers. The addresses of the Flash Configuration Word for devices in the PIC24FJ128GA family are shown in Table 3-1. Their location in the memory map is shown with the other memory vectors in Figure 3-1.

The Configuration Words in program memory are a compact format. The actual Configuration bits are mapped in several different registers in the configuration memory space. Their order in the Flash Configuration Words do not reflect a corresponding arrangement in the configuration space. Additional details on the device Configuration Words are provided in **Section 23.1 "Configuration Bits"**.

TABLE 3-1: FLASH CONFIGURATION WORDS FOR PIC24FJ128GA FAMILY DEVICES

Device	Program Memory (K words)	Configuration Word Addresses
PIC24FJ64GA	22	00ABFCh: 00ABFEh
PIC24FJ96GA	32	00FFFCh: 00FFFEh
PIC24FJ128GA	44	0157FCh: 0157FEh

FIGURE 3-2: PROGRAM MEMORY ORGANIZATION



3.2 Data Address Space

The PIC24 core has a separate 16-bit wide data memory space, addressable as a single linear range. The data space is accessed using two Address Generation Units (AGUs), one each for read and write operations. The data space memory map is shown in Figure 3-3.

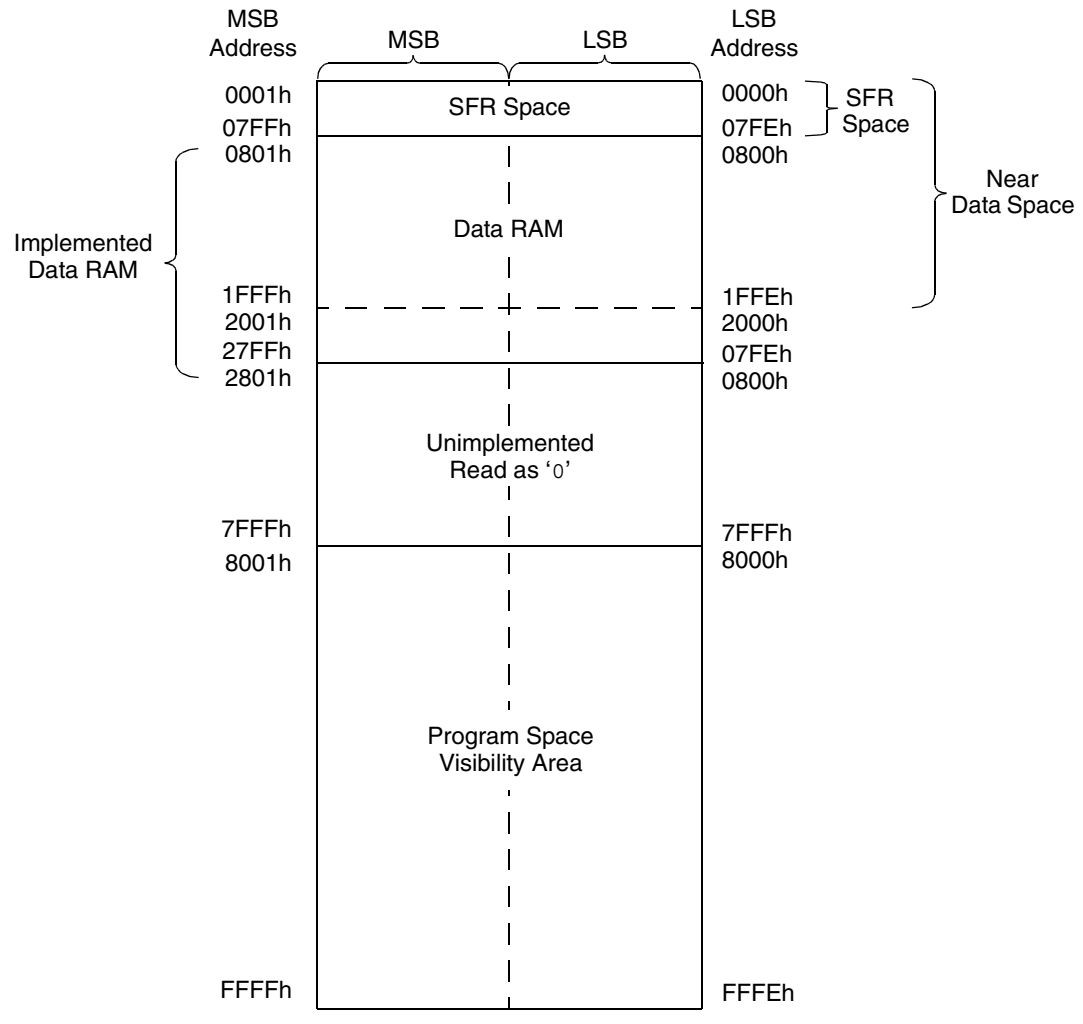
All Effective Addresses (EAs) in the data memory space are 16 bits wide, and point to bytes within the data space. This gives a data space address range of 64 Kbytes, or 32K words. The lower half of the data memory space (that is, when EA<15> = 0) is used for implemented memory addresses, while the upper half (EA<15> = 1) is reserved for the Program Space Visibility area (see **Section 3.3.3 "Reading Data from Program Memory Using Program Space Visibility"**).

PIC24FJ128GA family devices implement a total of 8 Kbytes of data memory. Should an EA point to a location outside of this area, an all zero word or byte will be returned.

3.2.1 DATA SPACE WIDTH

The data memory space is organized in byte addressable, 16-bit wide blocks. Data is aligned in data memory and registers as 16-bit words, but all data space EAs resolve to bytes. The Least Significant Bytes of each word have even addresses, while the Most Significant Bytes have odd addresses.

FIGURE 3-3: DATA SPACE MEMORY MAP FOR PIC24FJ128GA FAMILY DEVICES



Note: Data memory areas are not shown to scale.

3.2.2 DATA MEMORY ORGANIZATION AND ALIGNMENT

To maintain backward compatibility with PICmicro® devices and improve data space memory usage efficiency, the PIC24 instruction set supports both word and byte operations. As a consequence of byte accessibility, all effective address calculations are internally scaled to step through word-aligned memory. For example, the core recognizes that Post-Modified Register Indirect Addressing mode [Ws++] will result in a value of Ws + 1 for byte operations and Ws + 2 for word operations.

Data byte reads will read the complete word which contains the byte, using the LSb of any EA to determine which byte to select. The selected byte is placed onto the LSB of the data path. That is, data memory and registers are organized as two parallel byte-wide entities with shared (word) address decode but separate write lines. Data byte writes only write to the corresponding side of the array or register which matches the byte address.

All word accesses must be aligned to an even address. Misaligned word data fetches are not supported, so care must be taken when mixing byte and word operations, or translating from 8-bit MCU code. If a misaligned read or write is attempted, an address error trap will be generated. If the error occurred on a read, the instruction underway is completed; if it occurred on a write, the instruction will be executed but the write will not occur. In either case, a trap is then executed, allowing the system and/or user to examine the machine state prior to execution of the address Fault.

All byte loads into any W register are loaded into the Least Significant Byte. The Most Significant Byte is not modified.

A sign-extend instruction (SE) is provided to allow users to translate 8-bit signed data to 16-bit signed values. Alternatively, for 16-bit unsigned data, users can clear the MSB of any W register by executing a zero-extend (ZE) instruction on the appropriate address.

Although most instructions are capable of operating on word or byte data sizes, it should be noted that some instructions operate only on words.

3.2.3 NEAR DATA SPACE

The 8-Kbyte area between 0000h and 1FFFh is referred to as the near data space. Locations in this space are directly addressable via a 13-bit absolute address field within all memory direct instructions. The remainder of the data space is addressable indirectly. Additionally, the whole data space is addressable using MOV instructions, which support Memory Direct Addressing with a 16-bit address field.

3.2.4 SFR SPACE

The first 2 Kbytes of the near data space, from 0000h to 07FFh, are primarily occupied with Special Function Registers (SFRs). These are used by the PIC24 core and peripheral modules for controlling the operation of the device.

SFRs are distributed among the modules that they control, and are generally grouped together by module. Much of the SFR space contains unused addresses; these are read as '0'. A diagram of the SFR space, showing where SFRs are actually implemented, is shown in Table 3-2. Each implemented area indicates a 32-byte region where at least one address is implemented as an SFR. A complete listing of implemented SFRs, including their addresses, is shown in Tables 3-3 through 3-30.

TABLE 3-2: IMPLEMENTED REGIONS OF SFR DATA SPACE

	SFR Space Address							
	xx00	**xx20**	**xx40**	**xx60**	**xx80**	**xxA0**	**xxC0**	**xxE0**
000h	Core			ICN	Interrupts			—
100h	Timers		Capture	—	Compare	—	—	—
200h	I^2C™	UART	SPI		—	—	I/O	
300h	A/D		—	—	—	—	I/O	
400h	—	—	—	—	—	—	—	—
500h	—	—	—	—	—	—	—	—
600h	PMP	RTC/Comp	CRC	—	—	—	I/O	
700h	—	—	System	NVM/PMD	—	—	—	—

Legend: — = No implemented SFRs in this block

TABLE 3-3: CPU CORE REGISTERS MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
WREG0	0000	Working Register 0																0000
WREG1	0002	Working Register 1																0000
WREG2	0004	Working Register 2																0000
WREG3	0006	Working Register 3																0000
WREG4	0008	Working Register 4																0000
WREG5	000A	Working Register 5																0000
WREG6	000C	Working Register 6																0000
WREG7	000E	Working Register 7																0000
WREG8	0010	Working Register 8																0000
WREG9	0012	Working Register 9																0000
WREG10	0014	Working Register 10																0000
WREG11	0016	Working Register 11																0000
WREG12	0018	Working Register 12																0000
WREG13	001A	Working Register 13																0000
WREG14	001C	Working Register 14																0000
WREG15	001E	Working Register 15																0800
SPLIM	0020	Stack Pointer Limit																xxxx
PCL	002E	Program Counter, Low Word																0000
PCH	0030	—	—	—	—	—	—	—	—	Program Counter, High Byte								0000
TBLPAG	0032	—	—	—	—	—	—	—	—	Table Page Address Pointer								0000
PSVPAG	0034	—	—	—	—	—	—	—	—	Program Memory Visibility Page Address Pointer								0000
RCOUNT	0036	Repeat Loop Counter																xxxx
SR	0042	—	—	—	—	—	—	—	DC	IPL2	IPL1	IPL0	RA	N	OV	Z	C	0000
CORCON	0044	—	—	—	—	—	—	—	—	—	—	—	—	IPL3	PSV	—	—	0000
DISICNT	0052	—	—	Disable Interrupts Counter														xxxx

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 3-4: INTERRUPT CONTROLLER REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
INTCON1	0080	NSTDIS	—	—	—	—	—	—	—	—	—	—	MATHERR	ADDRERR	STKERR	OSCFAIL	—	0000
INTCON2	0082	ALTIVT	DISI	—	—	—	—	—	—	—	—	—	INT4EP	INT3EP	INT2EP	INT1EP	INT0EP	0000
IFS0	0084	—	—	AD1IF	U1TXIF	U1RXIF	SPI1IF	SPF1IF	T3IF	T2IF	OC2IF	IC2IF	—	T1IF	OC1IF	IC1IF	INT0IF	0000
IFS1	0086	U2TXIF	U2RXIF	INT2IF	T5IF	T4IF	OC4IF	OC3IF	—	—	—	—	INT1IF	CNIF	CMIF	MI2C1IF	SI2C1IF	0000
IFS2	0088	—	—	PMPIF	—	—	—	OC5IF	—	IC5IF	IC4IF	IC3IF	—	—	—	SPI2IF	SPF2IF	0000
IFS3	008A	—	RTCIF	—	—	—	—	—	—	—	INT4IF	INT3IF	—	—	MI2C2IF	SI2C2IF	—	0000
IFS4	008C	—	—	—	—	—	—	—	—	—	—	—	—	CRCIF	U2ERIF	U1ERIF	—	0000
IEC0	0094	—	—	AD1IE	U1TXIE	U1RXIE	SPI1IE	SPF1IE	T3IE	T2IE	OC2IE	IC2IE	—	T1IE	OC1IE	IC1IE	INT0IE	0000
IEC1	0096	U2TXIE	U2RXIE	INT2IE	T5IE	T4IE	OC4IE	OC3IE	—	—	—	—	INT1IE	CNIE	CMIE	MI2C1IE	SI2C1IE	0000
IEC2	0098	—	—	PMPIE	—	—	—	OC5IE	—	IC5IE	IC4IE	IC3IE	—	—	—	SPI2IE	SPF2IE	0000
IEC3	009A	—	RTCIE	—	—	—	—	—	—	—	INT4IE	INT3IE	—	—	MI2C2IE	SI2C2IE	—	0000
IEC4	009C	—	—	—	—	—	—	—	—	—	—	—	—	CRCIE	U2ERIE	U1ERIE	—	0000
IPC0	00A4	—	T1IP2	T1IP1	T1IP0	—	OC1IP2	OC1IP1	OC1IP0	—	IC1IP2	IC1IP1	IC1IP0	—	INT0IP2	INT0IP1	INT0IP0	4444
IPC1	00A6	—	T2IP2	T2IP1	T2IP0	—	OC2IP2	OC2IP1	OC2IP0	—	IC2IP2	IC2IP1	IC2IP0	—	—	—	—	4440
IPC2	00A8	—	U1RXIP2	U1RXIP1	U1RXIP0	—	SPI1IP2	SPI1IP1	SPI1IP0	—	SPF1IP2	SPF1IP1	SPF1IP0	—	T3IP2	T3IP1	T3IP0	4444
IPC3	00AA	—	—	—	—	—	—	—	—	—	AD1IP2	AD1IP1	AD1IP0	—	U1TXIP2	U1TXIP1	U1TXIP0	0044
IPC4	00AC	—	CNIP2	CNIP1	CNIP0	—	CMIP2	CMIP1	CMIP0	—	MI2C1P2	MI2C1P1	MI2C1P0	—	SI2C1P2	SI2C1P1	SI2C1P0	4444
IPC5	00AE	—	—	—	—	—	—	—	—	—	—	—	—	—	INT1IP2	INT1IP1	INT1IP0	0004
IPC6	00B0	—	T4IP2	T4IP1	T4IP0	—	OC4IP2	OC4IP1	OC4IP0	—	OC3IP2	OC3IP1	OC3IP0	—	—	—	—	4440
IPC7	00B2	—	U2TXIP2	U2TXIP1	U2TXIP0	—	U2RXIP2	U2RXIP1	U2RXIP0	—	INT2IP2	INT2IP1	INT2IP0	—	T5IP2	T5IP1	T5IP0	4444
IPC8	00B4	—	—	—	—	—	—	—	—	—	SPI2IP2	SPI2IP1	SPI2IP0	—	SPF2IP2	SPF2IP1	SPF2IP0	0044
IPC9	00B6	—	IC5IP2	IC5IP1	IC5IP0	—	IC4IP2	IC4IP1	IC4IP0	—	IC3IP2	IC3IP1	IC3IP0	—	—	—	—	4440
IPC10	00B8	—	—	—	—	—	—	—	—	—	OC5IP2	OC5IP1	OC5IP0	—	—	—	—	0040
IPC11	00BA	—	—	—	—	—	—	—	—	—	PMPIP2	PMPIP1	PMPIP0	—	—	—	—	0040
IPC12	00BC	—	—	—	—	—	MI2C2P2	MI2C2P1	MI2C2P0	—	SI2C2P2	SI2C2P1	SI2C2P0	—	—	—	—	0440
IPC13	00BE	—	—	—	—	—	INT4IP2	INT4IP1	INT4IP0	—	INT3IP2	INT3IP1	INT3IP0	—	—	—	—	0440
IPC15	00C2	—	—	—	—	—	RTCIP2	RTCIP1	RTCIP0	—	—	—	—	—	—	—	—	0400
IPC16	00C4	—	CRCIP2	CRCIP1	CRCIP0	—	U2ERIP2	U2ERIP1	U2ERIP0	—	U1ERIP2	U1ERIP1	U1ERIP0	—	—	—	—	4440

Legend: — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 3-5: ICN REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
CNEN1	0060	CN15IE	CN14IE	CN13IE	CN12IE	CN11IE	CN10IE	CN9IE	CN8IE	CN7IE	CN6IE	CN5IE	CN4IE	CN3IE	CN2IE	CN1IE	CN0IE	0000
CNEN2	0062	—	—	—	—	—	—	—	—	—	—	CN21IE	CN20IE	CN19IE	CN18IE	CN17IE	CN16IE	0000
CNPU1	0068	CN15PUE	CN14PUE	CN13PUE	CN12PUE	CN11PUE	CN10PUE	CN9PUE	CN8PUE	CN7PUE	CN6PUE	CN5PUE	CN4PUE	CN3PUE	CN2PUE	CN1PUE	CN0PUE	0000
CNPU2	006A	—	—	—	—	—	—	—	—	—	—	CN21PUE	CN20PUE	CN19PUE	CN18PUE	CN17PUE	CN16PUE	0000

Legend: — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 3-6: TIMER REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
TMR1	0100	Timer1 Register																xxxx
PR1	0102	Period Register 1																FFFF
T1CON	0104	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS1	TCKPS0	—	TSYNC	TCS	—	0000
TMR2	0106	Timer2 Register																xxxx
TMR3HLD	0108	Timer3 Holding Register (For 32-bit timer operations only)																xxxx
TMR3	010A	Timer3 Register																xxxx
PR2	010C	Period Register 2																FFFF
PR3	010E	Period Register 3																FFFF
T2CON	0110	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS1	TCKPS0	T32	—	TCS	—	0000
T3CON	0112	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS1	TCKPS0	—	—	TCS	—	0000
TMR4	0114	Timer4 Register																xxxx
TMR5HLD	0116	Timer5 Holding Register (For 32-bit operations only)																xxxx
TMR5	0118	Timer5 Register																xxxx
PR4	011A	Period Register 4																FFFF
PR5	011C	Period Register 5																FFFF
T4CON	011E	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS1	TCKPS0	T32	—	TCS	—	0000
T5CON	0120	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS1	TCKPS0	—	—	TCS	—	0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 3-7: INPUT CAPTURE REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
IC1BUF	0140	Input 1 Capture Register																xxxx
IC1CON	0142	—	—	ICSIDL	—	—	—	—	—	ICTMR	ICI1	ICI0	ICOV	ICBNE	ICM2	ICM1	ICM0	0000
IC2BUF	0144	Input 2 Capture Register																xxxx
IC2CON	0146	—	—	ICSIDL	—	—	—	—	—	ICTMR	ICI1	ICI0	ICOV	ICBNE	ICM2	ICM1	ICM0	0000
IC3BUF	0148	Input 3 Capture Register																xxxx
IC3CON	014A	—	—	ICSIDL	—	—	—	—	—	ICTMR	ICI1	ICI0	ICOV	ICBNE	ICM2	ICM1	ICM0	0000
IC4BUF	014C	Input 4 Capture Register																xxxx
IC4CON	014E	—	—	ICSIDL	—	—	—	—	—	ICTMR	ICI1	ICI0	ICOV	ICBNE	ICM2	ICM1	ICM0	0000
IC5BUF	0150	Input 5 Capture Register																xxxx
IC5CON	0152	—	—	ICSIDL	—	—	—	—	—	ICTMR	ICI1	ICI0	ICOV	ICBNE	ICM2	ICM1	ICM0	0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 3-8: OUTPUT COMPARE REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
OC1RS	0180	Output Compare 1 Secondary Register																xxxx
OC1R	0182	Output Compare 1 Register																xxxx
OC1CON	0184	—	—	OCSIDL	—	—	—	—	—	—	—	—	OCFLT	OCTSEL	OCM2	OCM1	OCM0	0000
OC2RS	0186	Output Compare 2 Secondary Register																xxxx
OC2R	0188	Output Compare 2 Register																xxxx
OC2CON	018A	—	—	OCSIDL	—	—	—	—	—	—	—	—	OCFLT	OCTSEL	OCM2	OCM1	OCM0	0000
OC3RS	018C	Output Compare 3 Secondary Register																xxxx
OC3R	018E	Output Compare 3 Register																xxxx
OC3CON	0190	—	—	OCSIDL	—	—	—	—	—	—	—	—	OCFLT	OCTSEL	OCM2	OCM1	OCM0	0000
OC4RS	0192	Output Compare 4 Secondary Register																xxxx
OC4R	0194	Output Compare 4 Register																xxxx
OC4CON	0196	—	—	OCSIDL	—	—	—	—	—	—	—	—	OCFLT	OCTSEL	OCM2	OCM1	OCM0	0000
OC5RS	0198	Output Compare 5 Secondary Register																xxxx
OC5R	019A	Output Compare 5 Register																xxxx
OC5CON	019C	—	—	OCSIDL	—	—	—	—	—	—	—	—	OCFLT	OCTSEL	OCM2	OCM1	OCM0	0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 3-9: I2C1 REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
I2C1RCV	0200	—	—	—	—	—	—	—	—	Receive Register								0000
I2C1TRN	0202	—	—	—	—	—	—	—	—	Transmit Register								00FF
I2C1BRG	0204	—	—	—	—	—	—	—	Baud Rate Generator									0000
I2C1CON	0206	I2CEN	—	I2CSIDL	SCLREL	IPMIEN	A10M	DISSLW	SMEN	GCEN	STREN	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	1000
I2C1STAT	0208	ACKSTAT	TRSTAT	—	—	—	BCL	GCSTAT	ADD10	IWCOL	I2COV	D/A	P	S	R/$\overline{W}$	RBF	TBF	0000
I2C1ADD	020A	—	—	—	—	—	—	Address Register										0000
I2C1MSK	020C	—	—	—	—	—	—	Address Mask										0000

Legend: — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 3-10: I2C2 REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
I2C2RCV	0210	—	—	—	—	—	—	—	—	Receive Register								0000
I2C2TRN	0212	—	—	—	—	—	—	—	—	Transmit Register								00FF
I2C2BRG	0214	—	—	—	—	—	—	—	Baud Rate Generator									0000
I2C2CON	0216	I2CEN	—	I2CSIDL	SCLREL	IPMIEN	A10M	DISSLW	SMEN	GCEN	STREN	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	1000
I2C2STAT	0218	ACKSTAT	TRSTAT	—	—	—	BCL	GCSTAT	ADD10	IWCOL	I2CPOV	D/A	P	S	R/$\overline{W}$	RBF	TBF	0000
I2C2ADD	021A	—	—	—	—	—	—	Address Register										0000
I2C2MSK	021C	—	—	—	—	—	—	Address Mask										0000

Legend: — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 3-11: UART1 REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
U1MODE	0220	UARTEN	—	USIDL	IREN	RTSMD	—	UEN1	UEN0	WAKE	LPBACK	ABAUD	RXINV	BRGH	PDSEL1	PDSEL0	STSEL	0000
U1STA	0222	UTXISEL1	UTXINV	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT	URXISEL1	URXISEL0	ADDEN	RIDLE	PERR	FERR	OERR	URXDA	0110
U1TXREG	0224	—	—	—	—	—	—	—	Transmit Register									xxxx
U1RXREG	0226	—	—	—	—	—	—	—	Receive Register									0000
U1BRG	0228	Baud Rate Generator Prescaler																0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 3-12: UART2 REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
U2MODE	0230	UARTEN	—	USIDL	IREN	RTSMD	—	UEN1	UEN0	WAKE	LPBACK	ABAUD	RXINV	BRGH	PDSEL1	PDSEL0	STSEL	0000
U2STA	0232	UTXISEL1	UTXINV	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT	URXISEL1	URXISEL0	ADDEN	RIDLE	PERR	FERR	OERR	URXDA	0110
U2TXREG	0234	—	—	—	—	—	—	—	Transmit Register									xxxx
U2RXREG	0236	—	—	—	—	—	—	—	Receive Register									0000
U2BRG	0238	Baud Rate Generator Prescaler																0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 3-13: SPI1 REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
SPI1STAT	0240	SPIEN	—	SPISIDL	—	—	SPIBEC2	SPIBEC1	SPIBEC0	—	SPIROV	—	—	—	—	SPITBF	SPIRBF	0000
SPI1CON1	0242	—	—	—	DISSCK	DISSDO	MODE16	SMP	CKE	SSEN	CKP	MSTEN	SPRE2	SPRE1	SPRE0	PPRE1	PPRE0	0000
SPI1CON2	0244	FRMEN	SPIFSD	SPIFPOL	—	—	—	—	—	—	—	—	—	—	—	SPIFE	SPIBEN	0000
SPI1BUF	0248	SPI1 Transmit and Receive Buffer																0000

Legend: — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 3-14: SPI2 REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
SPI2STAT	0260	SPIEN	—	SPISIDL	—	—	SPIBEC2	SPIBEC1	SPIBEC0	—	SPIROV	—	—	—	—	SPITBF	SPIRBF	0000
SPI2CON1	0262	—	—	—	DISSCK	DISSDO	MODE16	SMP	CKE	SSEN	CKP	MSTEN	SPRE2	SPRE1	SPRE0	PPRE1	PPRE0	0000
SPI2CON2	0264	FRMEN	SPIFSD	SPIFPOL	—	—	—	—	—	—	—	—	—	—	—	SPIFE	SPIBEN	0000
SPI2BUF	0268	SPI2 Transmit and Receive Buffer																0000

Legend: — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 3-15: ADC REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
ADC1BUF0	0300	ADC Data Buffer 0																xxxx
ADC1BUF1	0302	ADC Data Buffer 1																xxxx
ADC1BUF2	0304	ADC Data Buffer 2																xxxx
ADC1BUF3	0306	ADC Data Buffer 3																xxxx
ADC1BUF4	0308	ADC Data Buffer 4																xxxx
ADC1BUF5	030A	ADC Data Buffer 5																xxxx
ADC1BUF6	030C	ADC Data Buffer 6																xxxx
ADC1BUF7	030E	ADC Data Buffer 7																xxxx
ADC1BUF8	0310	ADC Data Buffer 8																xxxx
ADC1BUF9	0312	ADC Data Buffer 9																xxxx
ADC1BUFA	0314	ADC Data Buffer 10																xxxx
ADC1BUFB	0316	ADC Data Buffer 11																xxxx
ADC1BUFC	0318	ADC Data Buffer 12																xxxx
ADC1BUFD	031A	ADC Data Buffer 13																xxxx
ADC1BUFE	031C	ADC Data Buffer 14																xxxx
ADC1BUFF	031E	ADC Data Buffer 15																xxxx
AD1CON1	0320	ADON	—	ADSIDL	—	—	—	FORM1	FORM0	SSRC2	SSRC1	SSRC0	—	—	ASAM	SAMP	DONE	0000
AD1CON2	0322	VCFG2	VCFG1	VCFG0	OFFCAL	—	CSCNA	—	—	BUFS	—	SMPI3	SMPI2	SMPI1	SMPI0	BUFM	ALTS	0000
AD1CON3	0324	ADRC	—	—	SAMC4	SAMC3	SAMC2	SAMC1	SAMC0	ADCS7	ADCS6	ADCS5	ADCS4	ADCS3	ADCS2	ADCS1	ADCS0	0000
AD1CHS	0328	CH0NB	—	—	—	CH0SB3	CH0SB2	CH0SB1	CH0SB0	CH0NA	—	—	—	CH0SA3	CH0SA2	CH0SA1	CH0SA0	0000
AD1PCFG	032C	PCFG15	PCFG14	PCFG13	PCFG12	PCFG11	PCFG10	PCFG9	PCFG8	PCFG7	PCFG6	PCFG5	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0	0000
AD1CSSL	0330	CSSL15	CSSL14	CSSL13	CSSL12	CSSL11	CSSL10	CSSL9	CSSL8	CSSL7	CSSL6	CSSL5	CSSL4	CSSL3	CSSL2	CSSL1	CSSL0	0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 3-16: PORTA REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
TRISA	02C0	TRISA15(1)	TRISA14(1)	—	—	—	TRISA10(1)	TRISA9(1)	—	TRISA7(2)	TRISA6(2)	TRISA5(2)	TRISA4(2)	TRISA3(2)	TRISA2(2)	TRISA1(2)	TRISA0(2)	C6FF
PORTA	02C2	RA15(1)	RA14(1)	—	—	—	RA10(1)	RA9(1)	—	RA7(2)	RA6(2)	RA5(2)	RA4(2)	RA3(2)	RA2(2)	RA1(2)	RA0(2)	xxxx
LATA	02C4	LATA15(1)	LATA14(1)	—	—	—	LATA10(1)	LATA9(1)	—	LATA7(2)	LATA6(2)	LATA5(2)	LATA4(2)	LATA3(2)	LATA2(2)	LATA1(2)	LATA0(2)	xxxx
ODCA	06C0	ODA15(1)	ODA14(1)	—	—	—	ODA10(1)	ODA9(1)	—	ODA7(2)	ODA6(2)	ODA5(2)	ODA4(2)	ODA3(2)	ODA2(2)	ODA1(2)	ODA0(2)	0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal for 100-pin devices.

Note 1: Implemented in 80-pin and 100-pin devices only.

2: Implemented in 100-pin devices only

TABLE 3-17: PORTB REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
TRISB	02C6	TRISB15	TRISB14	TRISB13	TRISB12	TRISB11	TRISB10	TRISB9	TRISB8	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	FFFF
PORTB	02C8	RB15	RB14	RB13	RB12	RB11	RB10	RB9	RB8	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx
LATB	02CA	LATB15	LATB14	LATB13	LATB12	LATB11	LATB10	LATB9	LATB8	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	xxxx
ODCB	06C6	ODB15	ODB14	ODB13	ODB12	ODB11	ODB10	ODB9	ODB8	ODB7	ODB6	ODB5	ODB4	ODB3	ODB2	ODB1	ODB0	0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal for 100-pin devices.

TABLE 3-18: PORTC REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
TRISC	02CC	TRISC15	TRISC14	TRISC13	TRISC12	—	—	—	—	—	—	—	TRISC4(2)	TRISC3(1)	TRISC2(2)	TRISC1(1)	—	F01E
PORTC	02CE	RC15	RC14	RC13	RC12	—	—	—	—	—	—	—	RC4(2)	RC3(1)	RC2(2)	RC1(1)	—	xxxx
LATTC	02D0	LATC15	LATC14	LATC13	LATC12	—	—	—	—	—	—	—	LATC4(2)	LATC3(1)	LATC2(2)	LATC1(1)	—	xxxx
ODCC	06CC	ODC15	ODC14	ODC13	ODC12	—	—	—	—	—	—	—	ODC4(2)	ODC3(1)	ODC2(2)	ODC1(1)	—	0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal for 100-pin devices.
Note 1: Implemented in 80-pin and 100-pin devices only.
2: Implemented in 100-pin devices only

TABLE 3-19: PORTD REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
TRISD	02D2	TRISD15(1)	TRISD14(1)	TRISD13(1)	TRISD12(1)	TRISD11	TRISD10	TRISD9	TRISD8	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	FFFF
PORTD	02D4	RD15(1)	RD14(1)	RD13(1)	RD12(1)	RD11	RD10	RD9	RD8	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	xxxx
LATD	02D6	LATD15(1)	LATD14(1)	LATD13(1)	LATD12(1)	LATD11	LATD10	LATD9	LATD8	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0	xxxx
ODCD	06D2	ODD15(1)	ODD14(1)	ODD13(1)	ODD12(1)	ODD11	ODD10	ODD9	ODD8	ODD7	ODD6	ODD5	ODD4	ODD3	ODD2	ODD1	ODD0	0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal for 100-pin devices.
Note 1: Implemented in 80-pin and 100-pin devices only.

TABLE 3-20: PORTE REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
TRISE	02D8	—	—	—	—	—	—	TRISE9(1)	TRISE8(1)	TRISE7	TRISE6	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0	03FF
PORTE	02DA	—	—	—	—	—	—	RE9(1)	RE8(1)	RE7	RE6	RE5	RE4	RE3	RE2	RE1	RE0	xxxx
LATE	02DC	—	—	—	—	—	—	LATE9(1)	LATE8(1)	LATE7	LATE6	LATE5	LATE4	LATE3	LATE2	LATE1	LATE0	xxxx
ODCE	06D8	—	—	—	—	—	—	ODE9(1)	ODE8(1)	ODE7	ODE6	ODE5	ODE4	ODE3	ODE2	ODE1	ODE0	0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal for 100-pin devices.
Note 1: Implemented in 80-pin and 100-pin devices only.

TABLE 3-21: PORTF REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
TRISF	02DE	—	—	TRISF13(1)	TRISF12(1)	—	—	—	TRISF8(2)	TRISF7(2)	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0	31FF
PORTF	02E0	—	—	RG13(1)	RG12(1)	—	—	—	RF8(2)	RF7(2)	RF6	RF5	RF4	RF3	RF2	RF1	RF0	xxxx
LATF	02E2	—	—	LATF13(1)	LATF12(1)	—	—	—	LATF8(2)	LATF7(2)	LATF6	LATF5	LATF4	LATF3	LATF2	LATF1	LATF0	xxxx
ODCF	06DE	—	—	ODF13(1)	ODF12(1)	—	—	—	ODF8(2)	ODF7(2)	ODF6	ODF5	ODF4	ODF3	ODF2	ODF1	ODF0	0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal for 100-pin devices.
Note 1: Implemented in 100-pin devices only.
2: Implemented in 80-pin and 100-pin devices only.

TABLE 3-22: PORTG REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
TRISG	02E4	TRISG15(1)	TRISG14(1)	TRISG13(1)	TRISG12(1	—	—	TRISG9	TRISG8	TRISG7	TRISG6	—	—	TRISG3	TRISG2	TRISG1(2)	TRISG0(2)	F3CF
PORTG	02E6	RG15(1)	RG14(1)	RG13(1)	RG12(1)	—	—	RG9	RG8	RG7	RG6	—	—	RG3	RG2	RG1(2)	RG0(2)	xxxx
LATG	02E8	LATG15(1)	LATG14(1)	LATG13(1)	LATG12(1)	—	—	LATG9	LATG8	LATG7	LATG6	—	—	LATG3	LATG2	LATG1(2)	LATG0(2)	xxxx
ODCG	06E4	ODG15(1)	ODG14(1)	ODG13(1)	ODG12(1)	—	—	ODG9	ODG8	ODG7	ODG6	—	—	ODG3	ODG2	ODG1(2)	ODG0(2)	0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal for 100-pin devices.
Note 1: Implemented in 100-pin devices only
2: Implemented in 80-pin and 100-pin devices only.

TABLE 3-23: PAD CONFIGURATION MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
PADCFG1	02FC	—	—	—	—	—	—	—	—	—	—	—	—	—	—	RTSECSEL	PMPTTL	0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal for 100-pin devices.

TABLE 3-24: PARALLEL MASTER/SLAVE PORT REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
PMCON	0600	PMPEN	—	PSIDL	ADRMUX1	ADRMUX0	PTBEEN	PTWREN	PTRDEN	CSF1	CSF0	ALP	CS2P	CS1P	BEP	WRSP	RDSP	0000
PMMODE	0602	BUSY	IRQM1	IRQM0	INCM1	INCM0	MODE16	MODE1	MODE0	WAITB1	WAITB0	WAITM3	WAITM2	WAITM1	WAITM0	WAITE1	WAITE0	0000
PMADDR(1)	0604	CS2	CS1	Parallel Port Destination Address<13:0> (Master modes)														0000
PMDOUT1(1)		Parallel Port Data Out Register 1 (Buffers 0 and 1)																0000
PMDOUT2	0606	Parallel Port Data Out Register 2 (Buffers 2 and 3)																0000
PMDIN1	0608	Parallel Port Data In Register 1 (Buffers 0 and 1)																0000
PMPDIN2	060A	Parallel Port Data In Register 2 (Buffers 2 and 3)																0000
PMPEN	060C	PTEN15	PTEN14	PTEN13	PTEN12	PTEN11	PTEN10	PTEN9	PTEN8	PTEN7	PTEN6	PTEN5	PTEN4	PTEN3	PTEN2	PTEN1	PTEN0	0000
PMSTAT	060E	IBF	IBOV	—	—	IB3F	IB2F	IB1F	IB0F	OBE	OBUF	—	—	OB3E	OB2E	OB1E	OB0E	0000

Legend: — = unimplemented, read as '0'. Reset values are shown in hexadecimal.
Note 1: PMADDR and PMDOUT1 share the same physical register. The register functions as PMDOUT1 only in Slave modes, and as PMADDR only in Master modes.

TABLE 3-25: REAL-TIME CLOCK AND CALENDAR REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
ALRMVAL	0620	Alarm Value Register Window based on APTR<1:0>																xxxx
ALCFGRPT	0622	ALRMEN	CHIME	AMASK3	AMASK2	AMASK1	AMASK0	ALRMPTR1	ALRMPTR0	ARPT7	ARPT6	ARPT5	ARPT4	ARPT3	ARPT2	ARPT1	ARPT0	0000
RTCVAL	0624	RTCC Value Register Window based on RTCPTR<1:0>																xxxx
RCFGCAL(1)	0626	RTCEN	—	RTCWREN	RTCSYNC	HALFSEC	RTCOE	RTCPTR1	RTCPTR0	CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0	0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.
Note 1: RCFGCAL register Reset value dependent on type of Reset.

TABLE 3-26: DUAL COMPARATOR REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
CMCON	0630	CMIDL	—	C2EVT	C1EVT	C2EN	C1EN	C2OUTEN	C1OUTEN	C2OUT	C1OUT	C2INV	C1INV	C2NEG	C2POS	C1NEG	C1POS	0000
CVRCON	0632	—	—	—	—	—	—	—	—	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	0000

Legend: — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 3-27: CRC REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
CRCCON	0640	—	—	CSIDL	VWORD4	VWORD3	VWORD2	VWORD1	VWORD0	CRCFUL	CRCMPT	—	CRCGO	PLEN3	PLEN2	PLEN1	PLEN0	0000
CRCXOR	0642	CRC XOR Polynomial Register																0000
CRCDAT	0644	CRC Data Input Register																0000
CRCWDAT	0646	CRC Result Register																0000

Legend: — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

TABLE 3-28: SYSTEM REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
RCON	0740	TRAPR	IOPUWR	—	—	—	—	CM	VREGS	EXTR	SWR	SWDTEN	WDTO	SLEEP	IDLE	BOR	POR	xxxx(1)
OSCCON	0742	—	COSC2	COSC1	COSC0	—	NOSC2	NOSC1	NOSC0	CLKLOCK	—	LOCK	—	CF	—	SOSCEN	OSWEN	xxxx(2)
CLKDIV	0744	ROI	DOZE2	DOZE1	DOZE0	DOZEN	RCDIV2	RCDIV1	RCDIV0	—	—	—	—	—	—	—	—	0300
OSCTUN	0748	—	—	—	—	—	—	—	—	—	—	TUN<5:0>						0000

Legend: x = unknown value on Reset, — = unimplemented, read as '0'. Reset values are shown in hexadecimal.
Note 1: RCON register Reset values dependent on type of Reset.
2: OSCCON register Reset values dependent on the FOSC Configuration bits and by type of Reset.

TABLE 3-29: NVM REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
NVMCON	0760	WR	WREN	WRERR	—	—	—	—	—	—	ERASE	—	—	NVMOP3	NVMOP2	NVMOP1	NVMOP0	0000(1)
NVMKEY	0766	—	—	—	—	—	—	—	—	NVMKEY<7:0>								0000

Legend: — = unimplemented, read as '0'. Reset values are shown in hexadecimal.
Note 1: Reset value shown is for POR only. Value on other Reset states is dependent on the state of memory write or erase operations at the time of Reset.

TABLE 3-30: PMD REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
PMD1	0770	T5MD	T4MD	T3MD	T2MD	T1MD	—	—	—	I2C1MD	U2MD	U1MD	SPI2MD	SPI1MD	—	—	ADCMD	0000
PMD2	0772	—	—	—	IC5MD	IC4MD	IC3MD	IC2MD	IC1MD	—	—	—	OC5MD	OC4MD	OC3MD	OC2MD	OC1MD	0000
PMD3	0774	—	—	—	—	—	CMPMD	RTCCMD	PMPMD	CRCPMD	—	—	—	—	—	I2C2MD	—	0000

Legend: — = unimplemented, read as '0'. Reset values are shown in hexadecimal.

3.2.5 SOFTWARE STACK

In addition to its use as a working register, the W15 register in PIC24 devices is also used as a software Stack Pointer. The pointer always points to the first available free word and grows from lower to higher addresses. It pre-decrements for stack pops and post-increments for stack pushes, as shown in Figure 3-4. Note that for a PC push during any CALL instruction, the MSB of the PC is zero-extended before the push, ensuring that the MSB is always clear.

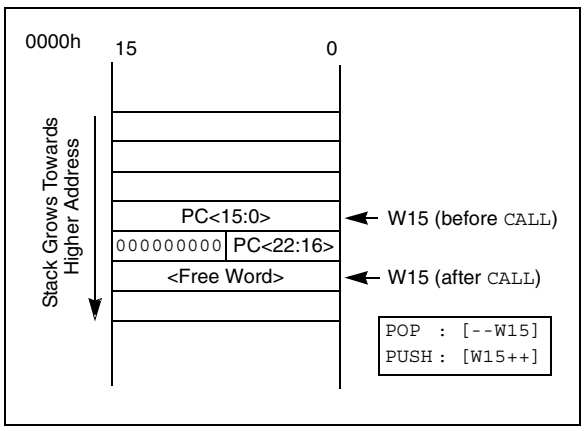
Note:	A PC push during exception processing will concatenate the SRL register to the MSB of the PC prior to the push.

The Stack Pointer Limit register (SPLIM) associated with the Stack Pointer sets an upper address boundary for the stack. SPLIM is uninitialized at Reset. As is the case for the Stack Pointer, SPLIM<0> is forced to '0' because all stack operations must be word-aligned. Whenever an EA is generated using W15 as a source or destination pointer, the resulting address is compared with the value in SPLIM. If the contents of the Stack Pointer (W15) and the SPLIM register are equal and a push operation is performed, a stack error trap will not occur. The stack error trap will occur on a subsequent push operation. Thus, for example, if it is desirable to cause a stack error trap when the stack grows beyond address 2000h in RAM, initialize the SPLIM with the value, 1FFEh.

Similarly, a Stack Pointer underflow (stack error) trap is generated when the Stack Pointer address is found to be less than 0800h. This prevents the stack from interfering with the Special Function Register (SFR) space.

A write to the SPLIM register should not be immediately followed by an indirect read operation using W15.

FIGURE 3-4: CALL STACK FRAME



3.3 Interfacing Program and Data Memory Spaces

The PIC24 architecture uses a 24-bit wide program space and 16-bit wide data space. The architecture is also a modified Harvard scheme, meaning that data can also be present in the program space. To use this data successfully, it must be accessed in a way that preserves the alignment of information in both spaces.

Aside from normal execution, the PIC24 architecture provides two methods by which program space can be accessed during operation:

- Using table instructions to access individual bytes or words anywhere in the program space
- Remapping a portion of the program space into the data space (Program Space Visibility)

Table instructions allow an application to read or write to small areas of the program memory. This makes the method ideal for accessing data tables that need to be updated from time to time. It also allows access to all bytes of the program word. The remapping method allows an application to access a large block of data on a read-only basis, which is ideal for look ups from a large table of static data. It can only access the least significant word of the program word.

3.3.1 ADDRESSING PROGRAM SPACE

Since the address ranges for the data and program spaces are 16 and 24 bits respectively, a method is needed to create a 23-bit or 24-bit program address from 16-bit data registers. The solution depends on the interface method to be used.

For table operations, the 8-bit Table Page register (TBLPAG) is used to define a 32K word region within the program space. This is concatenated with a 16-bit EA to arrive at a full 24-bit program space address. In this format, the Most Significant bit of TBLPAG is used to determine if the operation occurs in the user memory (TBLPAG<7> = 0) or the configuration memory (TBLPAG<7> = 1).

For remapping operations, the 8-bit Program Space Visibility register (PSVPAG) is used to define a 16K word page in the program space. When the Most Significant bit of the EA is '1', PSVPAG is concatenated with the lower 15 bits of the EA to form a 23-bit program space address. Unlike table operations, this limits remapping operations strictly to the user memory area.

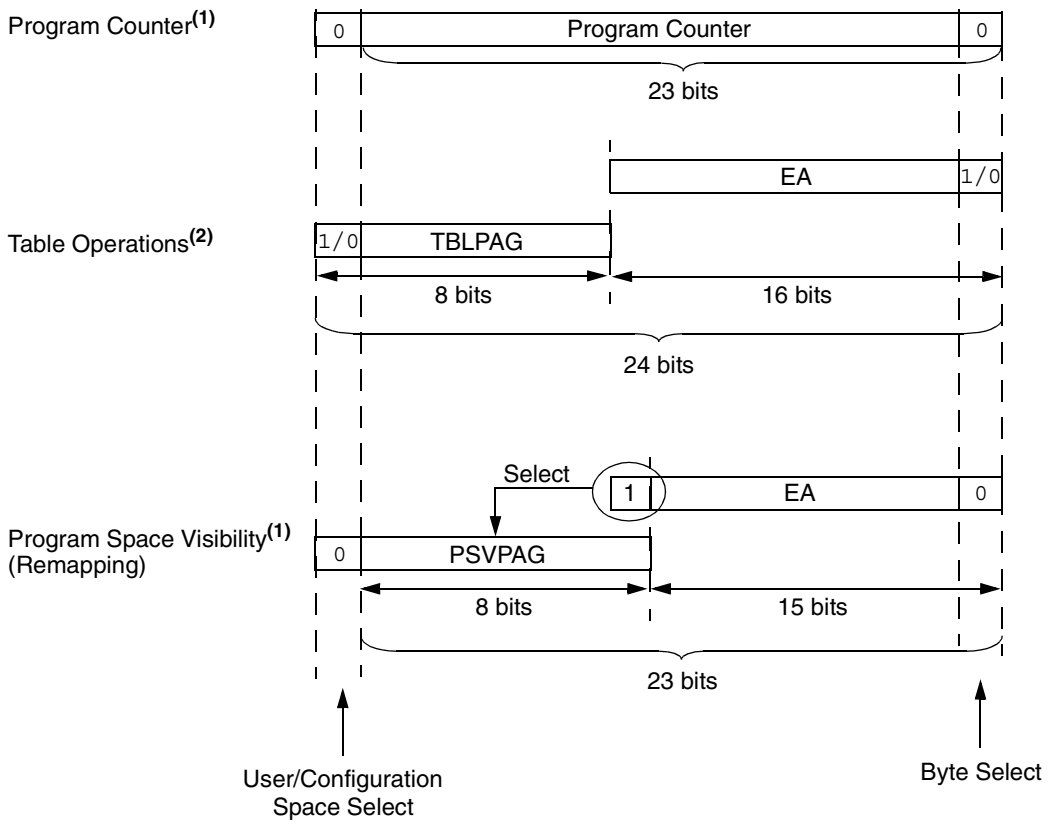
Table 3-31 and Figure 3-5 show how the program EA is created for table operations and remapping accesses from the data EA. Here, P<23:0> refers to a program space word, whereas D<15:0> refers to a data space word.

TABLE 3-31: PROGRAM SPACE ADDRESS CONSTRUCTION

Access Type	Access Space	Program Space Address				
		<23>	<22:16>	<15>	<14:1>	<0>
Instruction Access (Code Execution)	User	0	PC<22:1>			0
		0xx xxxx xxxx xxxx xxxx xxx0				
TBLRD/TBLWT (Byte/Word Read/Write)	User	TBLPAG<7:0>		Data EA<15:0>		
		0xxx xxxx		xxxx xxxx xxxx xxxx		
	Configuration	TBLPAG<7:0>		Data EA<15:0>		
		1xxx xxxx		xxxx xxxx xxxx xxxx		
Program Space Visibility (Block Remap/Read)	User	0	PSVPAG<7:0>		Data EA<14:0>(1)	
		0	xxxx xxxx		xxx xxxx xxxx xxxx	

Note 1: Data EA<15> is always '1' in this case, but is not used in calculating the program space address. Bit 15 of the address is PSVPAG<0>.

FIGURE 3-5: DATA ACCESS FROM PROGRAM SPACE ADDRESS GENERATION



Note 1: The LSb of program space addresses is always fixed as '0', in order to maintain word alignment of data in the program and data spaces.

2: Table operations are not required to be word-aligned. Table read operations are permitted in the configuration memory space.

3.3.2 DATA ACCESS FROM PROGRAM MEMORY USING TABLE INSTRUCTIONS

The TBLRDL and TBLWTL instructions offer a direct method of reading or writing the lower word of any address within the program space, without going through data space. The TBLRDH and TBLWTH instructions are the only method to read or write the upper 8 bits of a program space word as data.

The PC is incremented by two for each successive 24-bit program word. This allows program memory addresses to directly map to data space addresses. Program memory can thus be regarded as two 16-bit word wide address spaces, residing side by side, each with the same address range. TBLRDL and TBLWTL access the space which contains the least significant data word, and TBLRDH and TBLWTH access the space which contains the upper data byte.

Two table instructions are provided to move byte or word sized (16-bit) data to and from program space. Both function as either byte or word operations.

1. TBLRDL (Table Read Low): In Word mode, it maps the lower word of the program space location (P<15:0>) to a data address (D<15:0>).

   In Byte mode, either the upper or lower byte of the lower program word is mapped to the lower byte of a data address. The upper byte is selected when byte select is '1'; the lower byte is selected when it is '0'.

2. TBLRDH (Table Read High): In Word mode, it maps the entire upper word of a program address (P<23:16>) to a data address. Note that D<15:8>, the "phantom byte", will always be '0'.

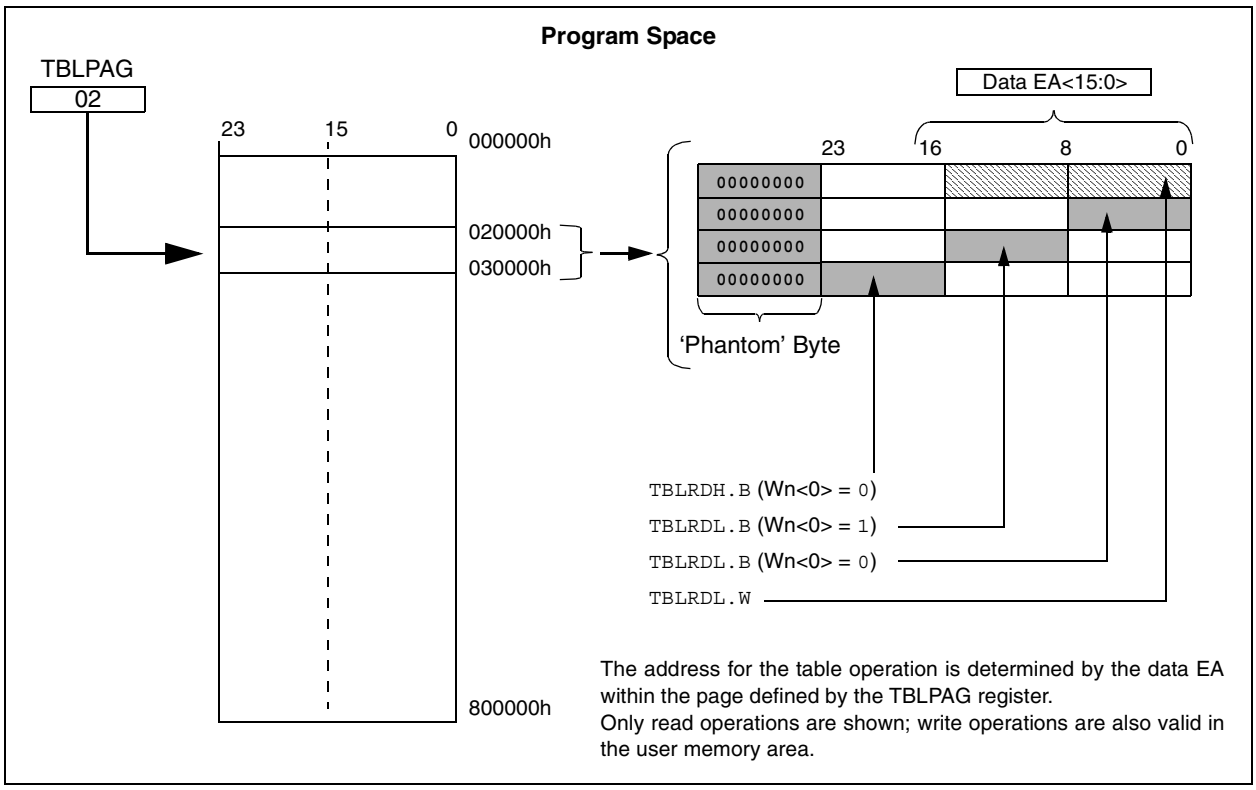
   In Byte mode, it maps the upper or lower byte of the program word to D<7:0> of the data address, as above. Note that the data will always be '0' when the upper "phantom" byte is selected (byte select = 1).

In a similar fashion, two table instructions, TBLWTH and TBLWTL, are used to write individual bytes or words to a program space address. The details of their operation are explained in **Section 4.0 "Flash Program Memory"**.

For all table operations, the area of program memory space to be accessed is determined by the Table Page register (TBLPAG). TBLPAG covers the entire program memory space of the device, including user and configuration spaces. When TBLPAG<7> = 0, the Table Page is located in the user memory space. When TBLPAG<7> = 1, the page is located in configuration space.

Note:	Only table read operations will execute in the configuration memory space and only then, in implemented areas such as the Device ID. Table write operations are not allowed.

FIGURE 3-6: ACCESSING PROGRAM MEMORY WITH TABLE INSTRUCTIONS



The address for the table operation is determined by the data EA within the page defined by the TBLPAG register.
Only read operations are shown; write operations are also valid in the user memory area.

3.3.3 READING DATA FROM PROGRAM MEMORY USING PROGRAM SPACE VISIBILITY

The upper 32 Kbytes of data space may optionally be mapped into any 16K word page of the program space. This provides transparent access of stored constant data from the data space without the need to use special instructions (i.e., TBLRDL/H).

Program space access through the data space occurs if the Most Significant bit of the data space EA is '1' and program space visibility is enabled by setting the PSV bit in the Core Control register (CORCON<2>). The location of the program memory space to be mapped into the data space is determined by the Program Space Visibility Page register (PSVPAG). This 8-bit register defines any one of 256 possible pages of 16K words in program space. In effect, PSVPAG functions as the upper 8 bits of the program memory address, with the 15 bits of the EA functioning as the lower bits. Note that by incrementing the PC by 2 for each program memory word, the lower 15 bits of data space addresses directly map to the lower 15 bits in the corresponding program space addresses.

Data reads to this area add an additional cycle to the instruction being executed, since two program memory fetches are required.

Although each data space address, 8000h and higher, maps directly into a corresponding program memory address (see Figure 3-7), only the lower 16 bits of the 24-bit program word are used to contain the data. The upper 8 bits of any program space locations used as data should be programmed with '1111 1111' or '0000 0000' to force a NOP. This prevents possible issues should the area of code ever be accidentally executed.

Note:	PSV access is temporarily disabled during table reads/writes.

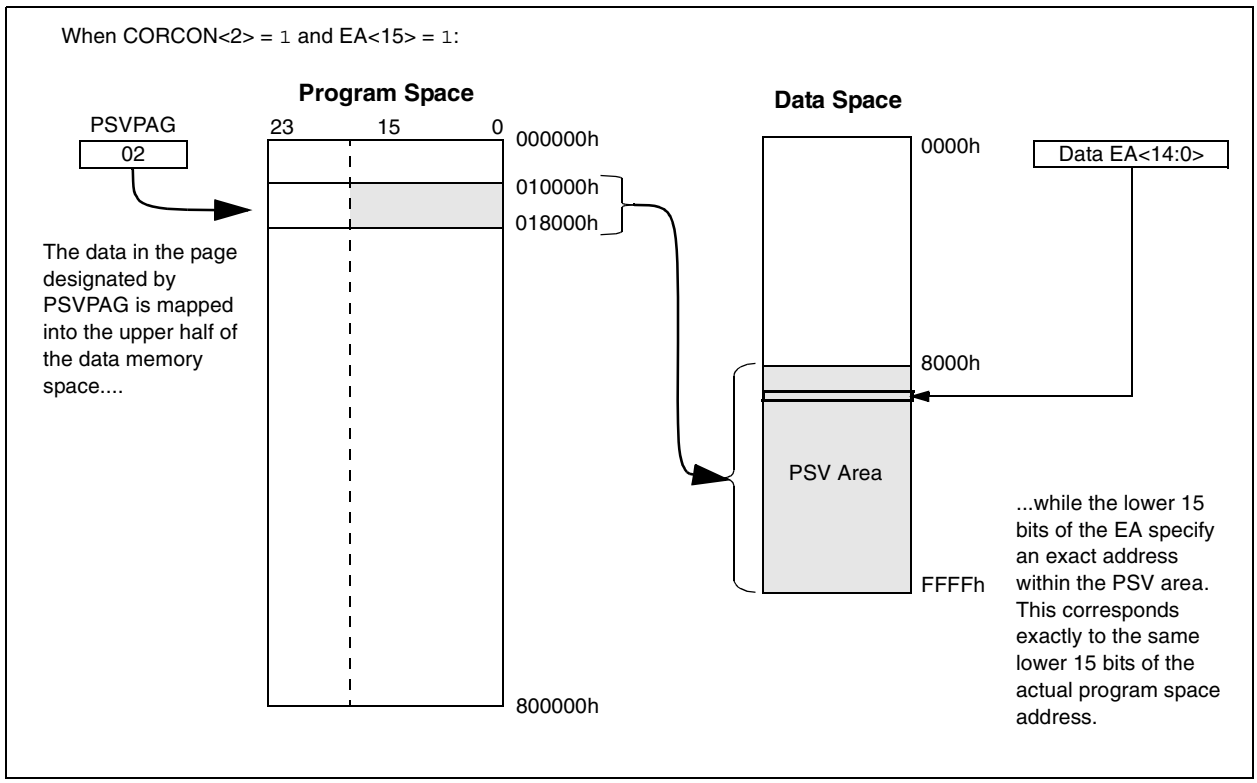
For operations that use PSV and are executed outside a REPEAT loop, the MOV and MOV.D instructions will require one instruction cycle in addition to the specified execution time. All other instructions will require two instruction cycles in addition to the specified execution time.

For operations that use PSV which are executed inside a REPEAT loop, there will be some instances that require two instruction cycles in addition to the specified execution time of the instruction:

- Execution in the first iteration
- Execution in the last iteration
- Execution prior to exiting the loop due to an interrupt
- Execution upon re-entering the loop after an interrupt is serviced

Any other iteration of the REPEAT loop will allow the instruction accessing data, using PSV, to execute in a single cycle.

FIGURE 3-7: PROGRAM SPACE VISIBILITY OPERATION



4.0 FLASH PROGRAM MEMORY

Note:	This data sheet summarizes the features of this group of PIC24FJ devices. It is not intended to be a comprehensive reference source.

The PIC24FJ128GA family of devices contains internal Flash program memory for storing and executing application code. The memory is readable, writable and erasable during normal operation over the entire VDD range.

Flash memory can be programmed in two ways:

1. In-Circuit Serial Programming (ICSP)
2. Run-Time Self-Programming (RTSP)

ICSP allows a PIC24FJ128GA family device to be serially programmed while in the end application circuit. This is simply done with two lines for Programming Clock and Programming Data (which are named PGCx and PGDx, respectively), and three other lines for power (VDD), ground (VSS) and Master Clear ($\overline{MCLR}$). 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.

RTSP is accomplished using TBLRD (table read) and TBLWT (table write) instructions. With RTSP, the user may write program memory data in blocks of 64 instructions (192 bytes) at a time, and erase program memory in blocks of 512 instructions (1536 bytes) at a time.

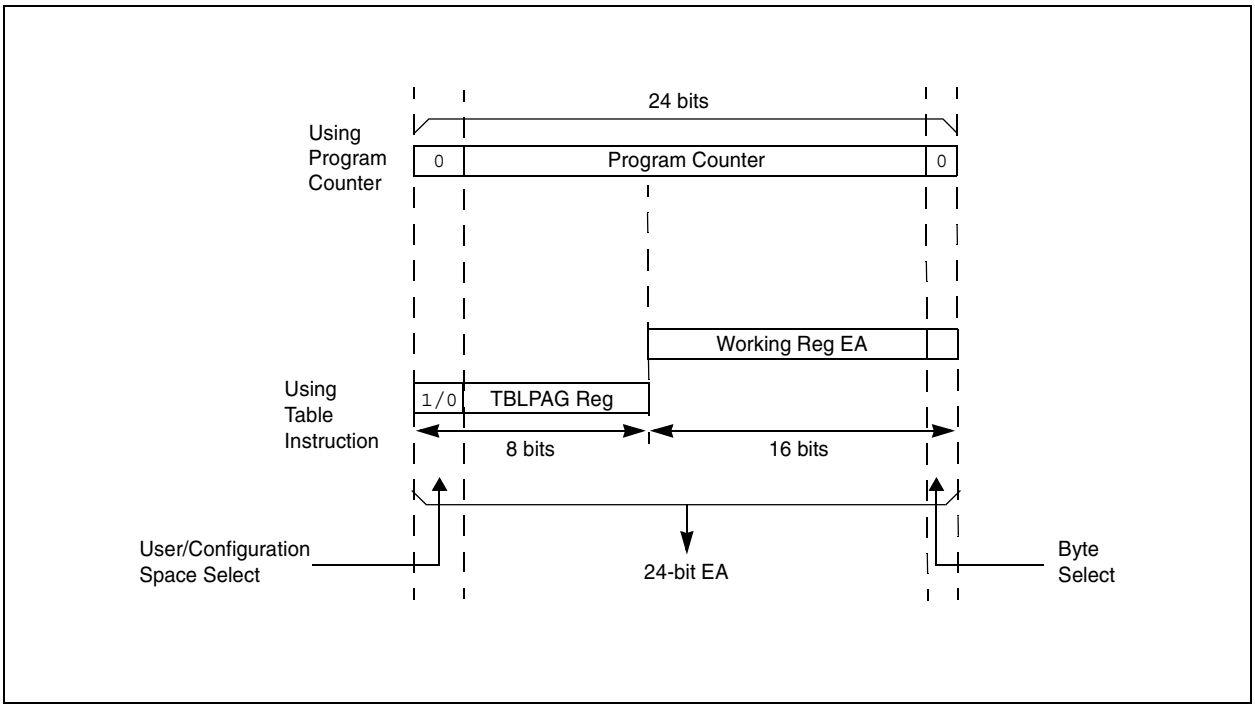
4.1 Table Instructions and Flash Programming

Regardless of the method used, all programming of Flash memory is done with the table read and table write instructions. These allow direct read and write access to the program memory space from the data memory while the device is in normal operating mode. The 24-bit target address in the program memory is formed using bits<7:0> of the TBLPAG register and the Effective Address (EA) from a W register specified in the table instruction, as shown in Figure 4-1.

The TBLRDL and the TBLWTL instructions are used to read or write to bits<15:0> of program memory. TBLRDL and TBLWTL can access program memory in both Word and Byte modes.

The TBLRDH and TBLWTH instructions are used to read or write to bits<23:16> of program memory. TBLRDH and TBLWTH can also access program memory in Word or Byte mode.

FIGURE 4-1: ADDRESSING FOR TABLE REGISTERS



4.2 RTSP Operation

The PIC24 Flash program memory array is organized into rows of 64 instructions or 192 bytes. RTSP allows the user to erase blocks of eight rows (512 instructions) at a time, and to program one row at a time. The 8-row erase blocks and single-row write blocks are edge-aligned, from the beginning of program memory, on boundaries of 1536 bytes and 192 bytes, respectively.

The program memory implements holding buffers that can contain 64 instructions of programming data. Prior to the actual programming operation, the write data must be loaded into the buffers in sequential order. The instructions words loaded must always be from a group of 64 boundaries.

The basic sequence for RTSP programming is to set up a Table Pointer, then do a series of `TBLWT` instructions to load the buffers. Programming is performed by setting the control bits in the NVMCON register. A total of 64 `TBLWTL` and `TBLWTH` instructions are required to load the instructions.

All of the table write operations are single-word writes (2 instruction cycles), because only the buffers are written. A programming cycle is required for programming each row.

4.3 Control Registers

There are two SFRs used to read and write the program Flash memory: NVMCON and NVMKEY.

The NVMCON register (Register 4-1) controls which blocks are to be erased, which memory type is to be programmed and the start of the programming cycle.

NVMKEY is a write-only register that is used for write protection. To start a programming or erase sequence, the user must consecutively write 55h and AAh to the NVMKEY register. Refer to **Section 4.4 "Programming Operations"** for further details.

4.4 Programming Operations

A complete programming sequence is necessary for programming or erasing the internal Flash in RTSP mode. A programming operation is nominally 4 ms in duration and the processor stalls (waits) until the operation is finished. Setting the WR bit (NVMCON<15>) starts the operation, and the WR bit is automatically cleared when the operation is finished.

REGISTER 4-1: NVMCOM: FLASH MEMORY CONTROL REGISTER

Upper Byte:							
R/SO-0(1)	R/W-0(1)	R/W-0(1)	U-0	U-0	U-0	U-0	U-0
WR	WREN	WRERR	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
U-0	R/W-0(1)	U-0	U-0	R/W-0(1)	R/W-0(1)	R/W-0(1)	R/W-0(1)
—	ERASE	—	—	NVMOP3(2)	NVMOP2(2)	NVMOP1(2)	NVMOP0(2)
bit 7							bit 0

bit 15 **WR:** Write Control bit
- 1 = Initiates a Flash memory program or erase operation
  The operation is self-timed and the bit is cleared by hardware once operation is complete.
- 0 = Program or erase operation is complete and inactive

bit 14 **WREN:** Write Enable bit
- 1 = Enable Flash program/erase operations
- 0 = Inhibit Flash program/erase operations

bit 13 **WRERR:** Write Sequence Error Flag bit
- 1 = An improper program or erase sequence attempt or termination has occurred (bit is set automatically on any set attempt of the WR bit)
- 0 = The program or erase operation completed normally

bit 12-7 **Unimplemented:** Read as '0'

bit 6 **ERASE:** Erase/Program Enable bit
- 1 = Perform the erase operation specified by NVMOP3:NVMOP0 on the next WR command
- 0 = Perform the program operation specified by NVMOP3:NVMOP0 on the next WR command

bit 5-4 **Unimplemented:** Read as '0'

bit 3-0 **NVMOP3:NVMOP0:** NVM Operation Select bits(2)
- 1111 = Memory bulk erase operation (ERASE = 1) or no operation (ERASE = 0)
- 0010 = Memory row erase operation (ERASE = 1) or no operation (ERASE = 0)
- 0001 = Memory row program operation (ERASE = 0) or no operation (ERASE = 1)

Note 1: These bits can only be reset on POR.
2: All other combinations of NVMOP3:NVMOP0 are unimplemented.

Legend:			
R = Readable bit	W = Writable bit	SO = Settable-Only bit	U = Unimplemented bit
-n = Value at Reset	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

4.4.1 PROGRAMMING ALGORITHM FOR FLASH PROGRAM MEMORY

The user can program one row of program Flash memory at a time. To do this, it is necessary to erase the 8-row erase block containing the desired row. The general process is:

1. Read eight rows of program memory (512 instructions) and store in data RAM.
2. Update the program data in RAM with the desired new data.
3. Erase the block (see Example 4-1):
   a) Set the NVMOP bits (NVMCOM<3:0>) to '0010' to configure for block erase. Set the ERASE (NVMCOM<6>) and WREN (NVMCOM<14>) bits.
   b) Write the starting address of the block to be erased into the TBLPAG and W registers.
   c) Write 55h to NVMKEY.
   d) Write AAh to NVMKEY.
   e) Set the WR bit (NVMCOM<15>). The erase cycle begins and the CPU stalls for the duration of the erase cycle. When the erase is done, the WR bit is cleared automatically.
4. Write the first 64 instructions from data RAM into the program memory buffers (see Example 4-2).
5. Write the program block to Flash memory:
   a) Set the NVMOP bits to '0001' to configure for row programming. Clear the ERASE bit and set the WREN bit.
   b) Write 55h to NVMKEY.
   c) Write AAh to NVMKEY.
   d) Set the WR bit. The programming cycle begins and the CPU stalls for the duration of the write cycle. When the write to Flash memory is done, the WR bit is cleared automatically.
6. Repeat steps 4 and 5, using the next available 64 instructions from the block in data RAM by incrementing the value in TBLPAG, until all 512 instructions are written back to Flash memory.

For protection against accidental operations, the write initiate sequence for NVMKEY must be used to allow any erase or program operation to proceed. After the programming command has been executed, the user must wait for the programming time until programming is complete. The two instructions following the start of the programming sequence should be NOPs, as shown in Example 4-3.

EXAMPLE 4-1: ERASING A PROGRAM MEMORY BLOCK

```
; Set up NVMCON for block erase operation
        MOV     #0x4042, W0                 ;
        MOV     W0, NVMCON                  ; Initialize NVMCON
; Init pointer to row to be ERASED
        MOV     #tblpage(PROG_ADDR), W0     ;
        MOV     W0, TBLPAG                  ; Initialize PM Page Boundary SFR
        MOV     #tbloffset(PROG_ADDR), W0   ; Initialize in-page EA[15:0] pointer
        TBLWTL  W0, [W0]                    ; Set base address of erase block
        DISI    #5                          ; Block all interrupts with priority <7
                                            ; for next 5 instructions
        MOV     #0x55, W0
        MOV     W0, NVMKEY                  ; Write the 55 key
        MOV     #0xAA, W1                   ;
        MOV     W1, NVMKEY                  ; Write the AA key
        BSET    NVMCON, #WR                 ; Start the erase sequence
        NOP                                 ; Insert two NOPs after the erase
        NOP                                 ; command is asserted
```


EXAMPLE 4-2: LOADING THE WRITE BUFFERS

```
; Set up NVMCON for row programming operations
       MOV       #0x4001, W0                    ;
       MOV       W0, NVMCON                     ; Initialize NVMCON
; Set up a pointer to the first program memory location to be written
; program memory selected, and writes enabled
       MOV       #0x0000, W0                    ;
       MOV       W0, TBLPAG                     ; Initialize PM Page Boundary SFR
       MOV       #0x6000, W0                    ; An example program memory address
; Perform the TBLWT instructions to write the latches
; 0th_program_word
       MOV       #LOW_WORD_0, W2                ;
       MOV       #HIGH_BYTE_0, W3               ;
       TBLWTL    W2, [W0]                       ; Write PM low word into program latch
       TBLWTH    W3, [W0++]                     ; Write PM high byte into program latch
; 1st_program_word
       MOV       #LOW_WORD_1, W2                ;
       MOV       #HIGH_BYTE_1, W3               ;
       TBLWTL    W2, [W0]                       ; Write PM low word into program latch
       TBLWTH    W3, [W0++]                     ; Write PM high byte into program latch
;  2nd_program_word
       MOV       #LOW_WORD_2, W2                ;
       MOV       #HIGH_BYTE_2, W3               ;
       TBLWTL    W2, [W0]                       ; Write PM low word into program latch
       TBLWTH    W3, [W0++]                     ; Write PM high byte into program latch
       •
       •
       •
; 63rd_program_word
       MOV       #LOW_WORD_31, W2               ;
       MOV       #HIGH_BYTE_31, W3              ;
       TBLWTL    W2, [W0]                       ; Write PM low word into program latch
       TBLWTH    W3, [W0++]                     ; Write PM high byte into program latch
```

EXAMPLE 4-3: INITIATING A PROGRAMMING SEQUENCE

```
       DISI    #5                          ; Block all interrupts with priority <7
                                           ; for next 5 instructions
       MOV     #0x55, W0
       MOV     W0, NVMKEY                  ; Write the 55 key
       MOV     #0xAA, W1                   ;
       MOV     W1, NVMKEY                  ; Write the AA key
       BSET    NVMCON, #WR                 ; Start the erase sequence
       NOP                                 ; Insert two NOPs after the
       NOP                                 ; erase command is asserted
```

NOTES:

5.0 RESETS

The Reset module combines all Reset sources and controls the device Master Reset Signal, $\overline{SYSRST}$. The following is a list of device Reset sources:

- POR: Power-on Reset
- $\overline{MCLR}$: Pin Reset
- SWR: RESET Instruction
- WDT: Watchdog Timer Reset
- BOR: Brown-out Reset
- TRAPR: Trap Conflict Reset
- IOPUWR: Illegal Opcode Reset
- UWR: Uninitialized W Register Reset

A simplified block diagram of the Reset module is shown in Figure 5-1.

Any active source of Reset will make the $\overline{SYSRST}$ signal active. Many registers associated with the CPU and peripherals are forced to a known Reset state. Most registers are unaffected by a Reset; their status is unknown on POR and unchanged by all other Resets.

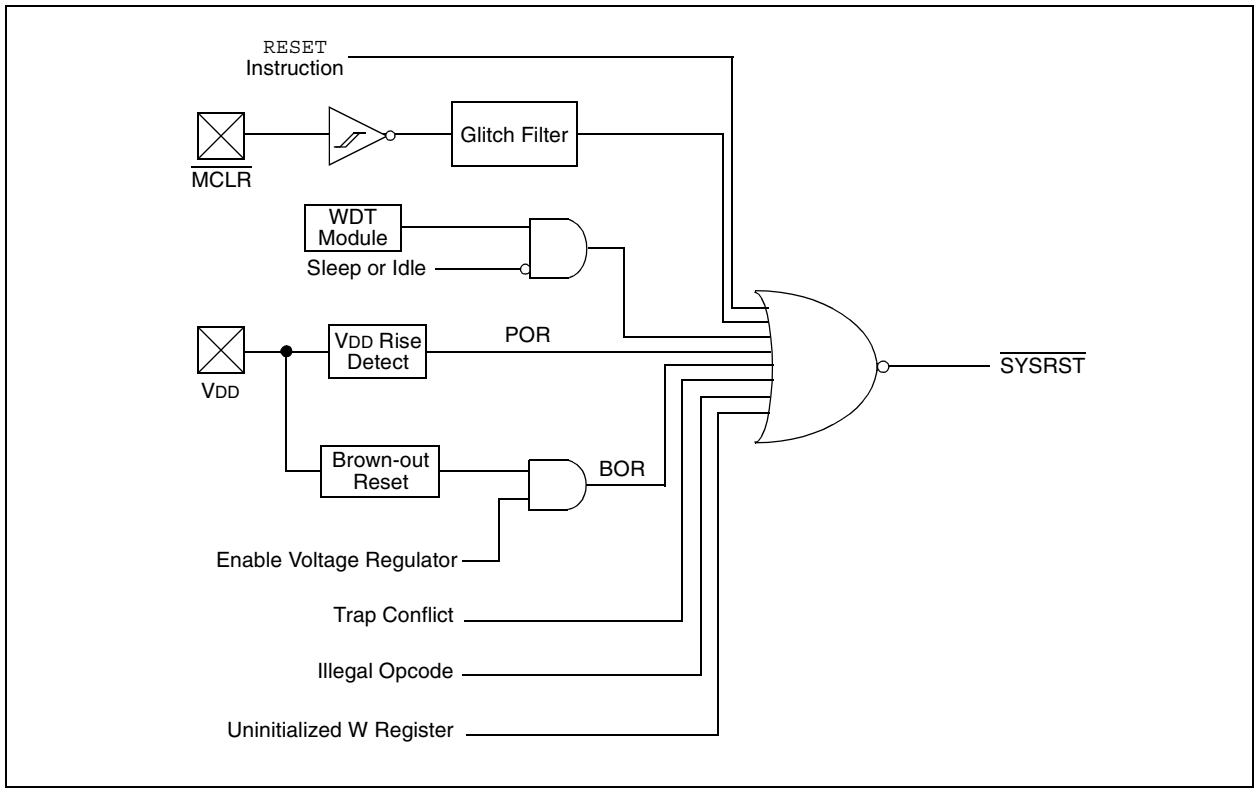
Note: Refer to the specific peripheral or CPU section of this manual for register Reset states.

All types of device Reset will set a corresponding status bit in the RCON register to indicate the type of Reset (see Register 5-1). A POR will clear all bits except for the BOR and POR bits (RCON<1:0>), which are set. The user may set or clear any bit at any time during code execution. The RCON bits only serve as status bits. Setting a particular Reset status bit in software will not cause a device Reset to occur.

The RCON register also has other bits associated with the Watchdog Timer and device power-saving states. The function of these bits is discussed in other sections of this manual.

Note: The status bits in the RCON register should be cleared after they are read so that the next RCON register value after a device Reset will be meaningful.

FIGURE 5-1: RESET SYSTEM BLOCK DIAGRAM



REGISTER 5-1: RCON: RESET CONTROL REGISTER

Upper Byte:							
R/W-0	R/W-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
TRAPR	IOPUWR	—	—	—	—	CM	VREGS
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-1
EXTR	SWR	SWDTEN	WDTO	SLEEP	IDLE	BOR	POR
bit 7							bit 0

bit 15 **TRAPR:** Trap Reset Flag bit
1 = A Trap Conflict Reset has occurred
0 = A Trap Conflict Reset has not occurred

bit 14 **IOPUWR:** Illegal Opcode or Uninitialized W Access Reset Flag bit
1 = An illegal opcode detection, an illegal address mode, or uninitialized W register used as an Address Pointer caused a Reset
0 = An illegal opcode or uninitialized W Reset has not occurred

bit 13-10 **Unimplemented:** Read as '0'

bit 9 **CM:** Configuration Word Mismatch Reset Flag bit
1 = A Configuration Word Mismatch Reset has occurred
0 = A Configuration Word Mismatch Reset has not occurred

bit 8 **VREGS:** Voltage Regulator Standby Enable bit
1 = Regulator remains active during Sleep
0 = Regulator goes to standby during Sleep

bit 7 **EXTR:** External Reset ($\overline{\text{MCLR}}$) Pin bit
1 = A Master Clear (pin) Reset has occurred
0 = A Master Clear (pin) Reset has not occurred

bit 6 **SWR:** Software Reset (Instruction) Flag bit
1 = A RESET instruction has been executed
0 = A RESET instruction has not been executed

bit 5 **SWDTEN:** Software Enable/Disable of WDT bit
1 = WDT is enabled
0 = WDT is disabled

Note: If the FWDTEN Configuration bit is '1' (unprogrammed), the WDT is always enabled, regardless of the SWDTEN bit setting.

bit 4 **WDTO:** Watchdog Timer Time-out Flag bit
1 = WDT time-out has occurred
0 = WDT time-out has not occurred

bit 3 **SLEEP:** Wake From Sleep Flag bit
1 = Device has been in Sleep mode
0 = Device has not been in Sleep mode

bit 2 **IDLE:** Wake-up From Idle Flag bit
1 = Device was in Idle mode
0 = Device was not in Idle mode

bit 1 **BOR:** Brown-out Reset Flag bit
1 = A Brown-out Reset has occurred. Note that BOR is also set after Power-on Reset.
0 = A Brown-out Reset has not occurred

bit 0 **POR:** Power-on Reset Flag bit
1 = A Power-up Reset has occurred
0 = A Power-up Reset has not occurred

Note: All of the Reset status bits may be set or cleared in software. Setting one of these bits in software does not cause a device Reset.

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

TABLE 5-1: RESET FLAG BIT OPERATION

Flag Bit	Setting Event	Clearing Event
TRAPR (RCON<15>)	Trap conflict event	POR
IOPR (RCON<14>)	Illegal opcode or uninitialized W register access	POR
EXTR (RCON<7>)	$\overline{\text{MCLR}}$ Reset	POR
SWR (RCON<6>)	RESET instruction	POR
WDTO (RCON<4>)	WDT time-out	PWRSAV instruction, POR
SLEEP (RCON<3>)	PWRSAV #SLEEP instruction	POR
IDLE (RCON<2>)	PWRSAV #IDLE instruction	POR
BOR (RCON<1>)	POR, BOR	—
POR (RCON<0>)	POR	—

Note: All Reset flag bits may be set or cleared by the user software.

5.1 Clock Source Selection at Reset

If clock switching is enabled, the system clock source at device Reset is chosen as shown in Table 5-2. If clock switching is disabled, the system clock source is always selected according to the oscillator Configuration bits. Refer to **7.0 "Oscillator Configuration"** for further details.

TABLE 5-2: OSCILLATOR SELECTION vs. TYPE OF RESET (CLOCK SWITCHING ENABLED)

Reset Type	Clock Source Determinant
POR	Oscillator Configuration Bits (FNOSC2:FNOSC0)
BOR	
$\overline{\text{MCLR}}$	COSC Control bits (OSCCON<14:12>)
WDTR	
SWR	

5.2 Device Reset Times

The Reset times for various types of device Reset are summarized in Table 5-3. Note that the system Reset signal, $\overline{\text{SYSRST}}$, is released after the POR and PWRT delay times expire.

The time that the device actually begins to execute code will also depend on the system oscillator delays, which include the Oscillator Start-up Timer (OST) and the PLL lock time. The OST and PLL lock times occur in parallel with the applicable $\overline{\text{SYSRST}}$ delay times.

The FSCM delay determines the time at which the FSCM begins to monitor the system clock source after the $\overline{\text{SYSRST}}$ signal is released.

TABLE 5-3: RESET DELAY TIMES FOR VARIOUS DEVICE RESETS

Reset Type	Clock Source	$\overline{\text{SYSRST}}$ Delay	System Clock Delay	FSCM Delay	Notes
POR	EC, FRC, FRCDIV, LPRC	TPOR + TSTARTUP + TRST	—	—	**1, 2, 3**
	ECPLL, FRCPLL	TPOR + TSTARTUP + TRST	TLOCK	TFSCM	**1, 2, 3, 5, 6**
	XT, HS, SOSC	TPOR + TSTARTUP + TRST	TOST	TFSCM	**1, 2, 3, 4, 6**
	XTPLL, HSPLL	TPOR + TSTARTUP + TRST	TOST + TLOCK	TFSCM	**1, 2, 3, 4, 5, 6**
BOR	EC, FRC, FRCDIV, LPRC	TSTARTUP + TRST	—	—	**2, 3**
	ECPLL, FRCPLL	TSTARTUP + TRST	TLOCK	TFSCM	**2, 3, 5, 6**
	XT, HS, SOSC	TSTARTUP + TRST	TOST	TFSCM	**2, 3, 4, 6**
	XTPLL, HSPLL	TSTARTUP + TRST	TOST + TLOCK	TFSCM	**2, 3, 4, 5, 6**
$\overline{\text{MCLR}}$	Any Clock	TRST	—	—	**3**
WDT	Any Clock	TRST	—	—	**3**
Software	Any clock	TRST	—	—	**3**
Illegal Opcode	Any Clock	TRST	—	—	**3**
Uninitialized W	Any Clock	TRST	—	—	**3**
Trap Conflict	Any Clock	TRST	—	—	**3**

Note 1: TPOR = Power-on Reset delay (10 μs nominal).

2: TSTARTUP = TVREG (10 μs nominal) if on-chip regulator enabled or TPWRT (64 ms nominal) if on-chip regulator disabled.

3: TRST = Internal state Reset time (20 μs nominal).

4: TOST = Oscillator Start-up Timer. A 10-bit counter counts 1024 oscillator periods before releasing the oscillator clock to the system.

5: TLOCK = PLL lock time (20 μs nominal).

6: TFSCM = Fail-Safe Clock Monitor delay (100 μs nominal).

5.2.1 POR AND LONG OSCILLATOR START-UP TIMES

The oscillator start-up circuitry and its associated delay timers are not linked to the device Reset delays that occur at power-up. Some crystal circuits (especially low-frequency crystals) will have a relatively long start-up time. Therefore, one or more of the following conditions is possible after $\overline{\text{SYSRST}}$ is released:

- The oscillator circuit has not begun to oscillate.
- The Oscillator Start-up Timer has NOT expired (if a crystal oscillator is used).
- The PLL has not achieved a LOCK (if PLL is used).

The device will not begin to execute code until a valid clock source has been released to the system. Therefore, the oscillator and PLL start-up delays must be considered when the Reset delay time must be known.

5.2.2 FAIL-SAFE CLOCK MONITOR (FSCM) AND DEVICE RESETS

If the FSCM is enabled, it will begin to monitor the system clock source when $\overline{\text{SYSRST}}$ is released. If a valid clock source is not available at this time, the device will automatically switch to the FRC oscillator and the user can switch to the desired crystal oscillator in the Trap Service Routine.

5.2.2.1 FSCM Delay for Crystal and PLL Clock Sources

When the system clock source is provided by a crystal oscillator and/or the PLL, a small delay, TFSCM, will automatically be inserted after the POR and PWRT delay times. The FSCM will not begin to monitor the system clock source until this delay expires. The FSCM delay time is nominally 100 μs and provides additional time for the oscillator and/or PLL to stabilize. In most cases, the FSCM delay will prevent an oscillator failure trap at a device Reset when the PWRT is disabled.

5.3 Special Function Register Reset States

Most of the Special Function Registers (SFRs) associated with the PIC24 CPU and peripherals are reset to a particular value at a device Reset. The SFRs are grouped by their peripheral or CPU function and their Reset values are specified in each section of this manual.

The Reset value for each SFR does not depend on the type of Reset, with the exception of four registers. The Reset value for the Reset Control register, RCON, will depend on the type of device Reset. The Reset value for the Oscillator Control register, OSCCON, will depend on the type of Reset and the programmed values of the oscillator Configuration bits in the FOSC Device Configuration register (see Table 5-2). The RCFGCAL and EECON1 registers are only affected by a POR.

NOTES:

6.0 INTERRUPT CONTROLLER

The PIC24 interrupt controller reduces the numerous peripheral interrupt request signals to a single interrupt request signal to the PIC24 CPU. It has the following features:

- Up to 8 processor exceptions and software traps
- 7 user-selectable priority levels
- Interrupt Vector Table (IVT) with up to 118 vectors
- A unique vector for each interrupt or exception source
- Fixed priority within a specified user priority level
- Alternate Interrupt Vector Table (AIVT) for debug support
- Fixed interrupt entry and return latencies

6.1 Interrupt Vector Table

The Interrupt Vector Table (IVT) is shown in Figure 6-1. The IVT resides in program memory, starting at location 000004h. The IVT contains 126 vectors, consisting of 8 non-maskable trap vectors, plus up to 118 sources of interrupt. In general, each interrupt source has its own vector. Each interrupt vector contains a 24-bit wide address. The value programmed into each interrupt vector location is the starting address of the associated Interrupt Service Routine (ISR).

Interrupt vectors are prioritized in terms of their natural priority; this is linked to their position in the vector table. All other things being equal, lower addresses have a higher natural priority. For example, the interrupt associated with vector 0 will take priority over interrupts at any other vector address.

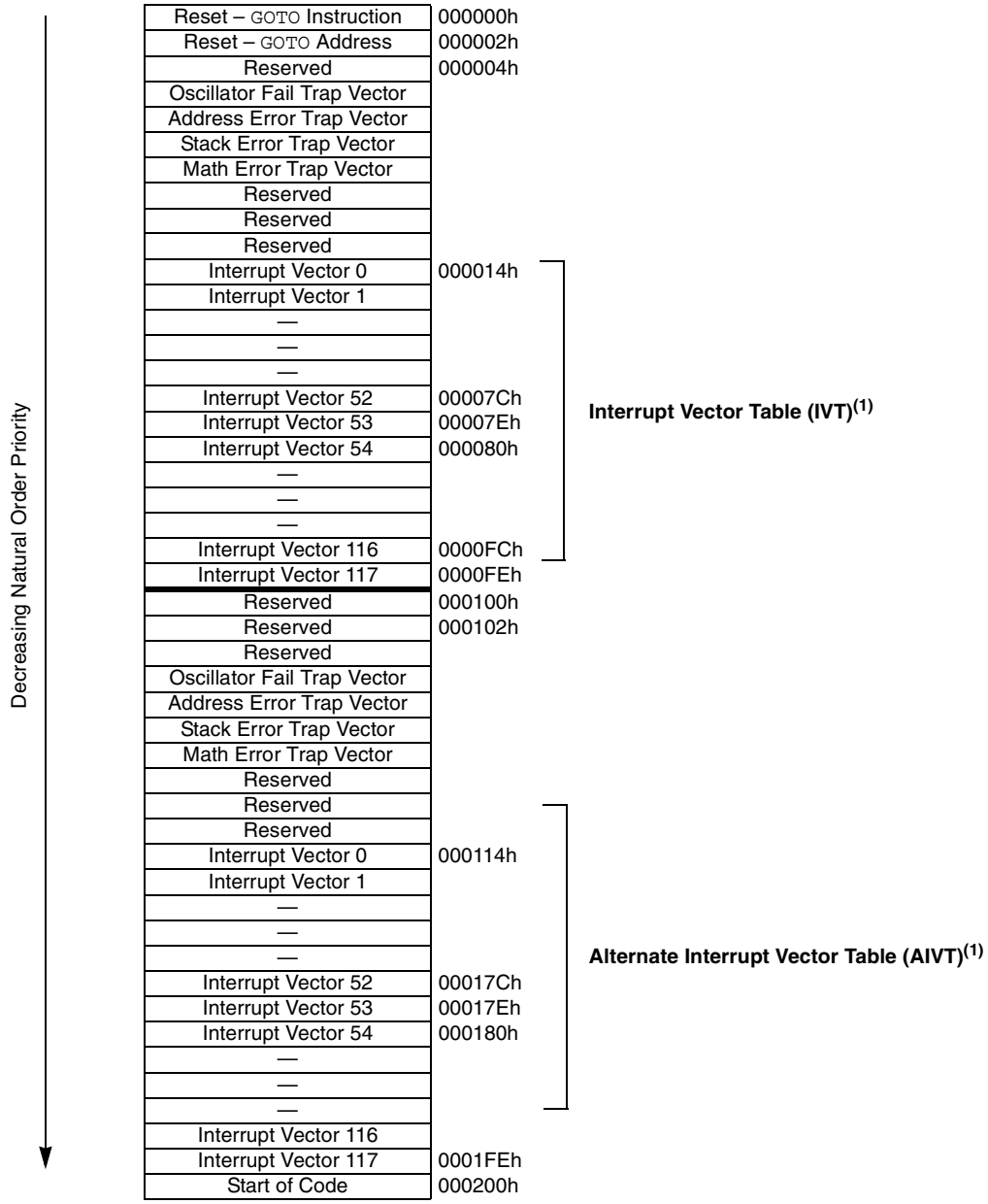
PIC24FJ128GA family devices implement non-maskable traps and unique interrupts. These are summarized in Table 6-1 and Table 6-2.

6.1.1 ALTERNATE INTERRUPT VECTOR TABLE

The Alternate Interrupt Vector Table (AIVT) is located after the IVT as shown in Figure 6-1. Access to the AIVT is provided by the ALTIVT control bit (INTCON2<15>). If the ALTIVT bit is set, all interrupt and exception processes will use the alternate vectors instead of the default vectors. The alternate vectors are organized in the same manner as the default vectors.

The AIVT supports emulation and debugging efforts by providing a means to switch between an application

FIGURE 6-1: PIC24 INTERRUPT VECTOR TABLE



Note 1: See Table 6-2 for the Interrupt Vector list.

TABLE 6-1: TRAP VECTOR DETAILS

Vector Number	IVT Address	AIVT Address	Trap Source
0	000004h	000104h	Reserved
1	000006h	000106h	Oscillator Failure
2	000008h	000108h	Address Error
3	00000Ah	00010Ah	Stack Error
4	00000Ch	00010Ch	Math Error
5	00000Eh	00010Eh	Reserved
6	000010h	000110h	Reserved
7	000012h	0001172h	Reserved

TABLE 6-2: IMPLEMENTED INTERRUPT VECTORS

Interrupt Source	Vector Number	IVT Address	AIVT Address	Interrupt Bit Locations		
				Flag	Enable	Priority
ADC1 Conversion Done	13	00002Eh	00012Eh	IFS0<13>	IEC0<13>	IPC3<6:4>
Comparator Event	18	000038h	000138h	IFS1<2>	IEC1<2>	IPC4<10:8>
CRC Generator	67	00009Ah	00019Ah	IFS4<3>	IEC4<3>	IPC16<14:12>
External Interrupt 0	0	000014h	000114h	IFS0<0>	IEC0<0>	IPC0<2:0>
External Interrupt 1	20	00003Ch	00013Ch	IFS1<4>	IEC1<4>	IPC5<2:0>
External Interrupt 2	29	00004Eh	00014Eh	IFS1<13>	IEC1<13>	IPC7<6:4>
External Interrupt 3	53	00007Eh	00017Eh	IFS3<5>	IEC3<5>	IPC13<6:4>
External Interrupt 4	54	000080h	000180h	IFS3<6>	IEC3<6>	IPC13<10:8>
I2C1 Master Event	17	000036h	000136h	IFS1<1>	IEC1<1>	IPC4<6:4>
I2C1 Slave Event	16	000034h	000034h	IFS1<0>	IEC1<0>	IPC4<2:0>
I2C2 Master Event	50	000078h	000178h	IFS3<2>	IEC3<2>	IPC12<10:8>
I2C2 Slave Event	49	000076h	000176h	IFS3<1>	IEC3<1>	IPC12<6:4>
Input Capture 1	1	000016h	000116h	IFS0<1>	IEC0<1>	IPC0<6:4>
Input Capture 2	5	00001Eh	00011Eh	IFS0<5>	IEC0<5>	IPC1<6:4>
Input Capture 3	37	00005Eh	00015Eh	IFS2<5>	IEC2<5>	IPC9<6:4>
Input Capture 4	38	000060h	000160h	IFS2<6>	IEC2<6>	IPC9<10:8>
Input Capture 5	39	000062h	000162h	IFS2<7>	IEC2<7>	IPC9<14:12>
Input Change Notification	19	00003Ah	00013Ah	IFS1<3>	IEC1<3>	IPC4<14:12>
Output Compare 1	2	000018h	000118h	IFS0<2>	IEC0<2>	IPC0<10:8>
Output Compare 2	6	000020h	000120h	IFS0<6>	IEC0<6>	IPC1<10:8>
Output Compare 3	25	000046h	000146h	IFS1<9>	IEC1<9>	IPC6<6:4>
Output Compare 4	26	000048h	000148h	IFS1<10>	IEC1<10>	IPC6<10:8>
Output Compare 5	41	000066h	000166h	IFS2<9>	IEC2<9>	IPC10<6:4>
Parallel Master Port	45	00006Eh	00016Eh	IFS2<13>	IEC2<13>	IPC11<6:4>
Real-Time Clock/Calendar	62	000090h	000190h	IFS3<14>	IEC3<13>	IPC15<10:8>
SPI1 Error	9	000026h	000126h	IFS0<9>	IEC0<9>	IPC2<6:4>
SPI1 Event	10	000028h	000128h	IFS0<10>	IEC0<10>	IPC2<10:8>
SPI2 Error	32	000054h	000154h	IFS2<0>	IEC0<0>	IPC8<2:0>
SPI2 Event	33	000056h	000156h	IFS2<1>	IEC2<1>	IPC8<6:4>
Timer1	3	00001Ah	00011Ah	IFS0<3>	IEC0<3>	IPC0<14:12>
Timer2	7	000022h	000122h	IFS0<7>	IEC0<7>	IPC1<14:12>
Timer3	8	000024h	000124h	IFS0<8>	IEC0<8>	IPC2<2:0>
Timer4	27	00004Ah	00014Ah	IFS1<11>	IEC1<11>	IPC6<14:12>
Timer5	28	00004Ch	00014Ch	IFS1<12>	IEC1<12>	IPC7<2:0>
UART1 Error	65	000096h	000196h	IFS4<1>	IEC4<1>	IPC16<6:4>
UART1 Receiver	11	00002Ah	00012Ah	IFS0<11>	IEC0<11>	IPC2<14:12>
UART1 Transmitter	12	00002Ch	00012Ch	IFS0<12>	IEC0<12>	IPC3<2:0>
UART2 Error	66	000098h	000198h	IFS4<2>	IEC4<2>	IPC16<10:8>
UART2 Receiver	30	000050h	000150h	IFS1<14>	IEC1<14>	IPC7<10:8>
UART2 Transmitter	31	000052h	000152h	IFS1<15>	IEC1<15>	IPC7<14:12>

6.3 Interrupt Control and Status Registers

The PIC24FJ128GA family devices implement a total of 28 registers for the interrupt controller:

- INTCON1
- INTCON2
- IFS0 through IFS4
- IEC0 through IEC4
- IPC0 through IPC14, and IPC16

Global interrupt control functions are controlled from INTCON1 and INTCON2. INTCON1 contains the Interrupt Nesting Disable (NSTDIS) bit, as well as the control and status flags for the processor trap sources. The INTCON2 register controls the external interrupt request signal behavior and the use of the Alternate Interrupt Vector Table.

The IFS registers maintain all of the interrupt request flags. Each source of interrupt has a status bit which is set by the respective peripherals, or external signal, and is cleared via software.

The IEC registers maintain all of the interrupt enable bits. These control bits are used to individually enable interrupts from the peripherals or external signals.

The IPC registers are used to set the interrupt priority level for each source of interrupt. Each user interrupt source can be assigned to one of eight priority levels.

The interrupt sources are assigned to the IFSx, IECx and IPCx registers in the same sequence that they are listed in Table 6-2. For example, the INT0 (External Interrupt 0) is shown as having a vector number and a natural order priority of 0. Thus, the INT0IF status bit is found in IFS0<0>, the enable bit in IEC0<0> and the priority bits in the first position of IPC0 (IPC0<2:0>).

Although they are not specifically part of the interrupt control hardware, two of the CPU control registers contain bits that control interrupt functionality. The CPU STATUS register (SR) contains the IPL2:IPL0 bits (SR<7:5>). These indicate the current CPU interrupt priority level. The user may change the current CPU priority level by writing to the IPL bits.

The CORCON register contains the IPL3 bit, which together with IPL2:IPL0, also indicates the current CPU priority level. IPL3 is a read-only bit so that trap events cannot be masked by the user software.

All interrupt registers are described in Register 6-1 through Register 6-30, in the following pages.

REGISTER 6-1: SR: STATUS REGISTER (IN CPU)

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	U-0	R-0
—	—	—	—	—	—	—	DC
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
IPL2(1,2)	IPL1(1,2)	IPL0(1,2)	RA	N	OV	Z	C
bit 7							bit 0

bit 7-5 **IPL2:IPL0:** CPU Interrupt Priority Level Status bits(1,2)

111 = CPU interrupt priority level is 7 (15). User interrupts disabled.
110 = CPU interrupt priority level is 6 (14)
101 = CPU interrupt priority level is 5 (13)
100 = CPU interrupt priority level is 4 (12)
011 = CPU interrupt priority level is 3 (11)
010 = CPU interrupt priority level is 2 (10)
001 = CPU interrupt priority level is 1 (9)
000 = CPU interrupt priority level is 0 (8)

Note 1: The IPL bits are concatenated with the IPL3 bit (CORCON<3>) to form the CPU interrupt priority level. The value in parentheses indicates the IPL if IPL3 = 1.

2: The IPL Status bits are read-only when NSTDIS (INTCON1<15>) = 1.

.

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

REGISTER 6-2: CORCON: CORE CONTROL REGISTER

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
U-0	U-0	U-0	U-0	R/C-0	R/W-0	U-0	U-0
—	—	—	—	IPL3(1)	PSV	—	—
bit 7							bit 0

bit 3 **IPL3:** CPU Interrupt Priority Level Status bit(1)

1 = CPU interrupt priority level is greater than 7; peripheral interrupts are disabled
0 = CPU interrupt priority level is 7 or less

Note 1: The IPL3 bit is concatenated with the IPL2:IPL0 bits (SR<7:5>) to form the CPU interrupt priority level.

.

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

REGISTER 6-3: INTCON1: INTERRUPT CONTROL REGISTER 1

Upper Byte:							
R/W-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
NSTDIS	—	—	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0
—	—	—	MATHERR	ADDRERR	STKERR	OSCFAIL	—
bit 7							bit 0

bit 15 **NSTDIS:** Interrupt Nesting Disable bit
1 = Interrupt nesting is disabled
0 = Interrupt nesting is enabled

bit 14-5 **Unimplemented:** Read as '0'

bit 4 **MATHERR:** Arithmetic Error Trap Status bit
1 = Overflow trap has occurred
0 = Overflow trap has not occurred

bit 3 **ADDRERR:** Address Error Trap Status bit
1 = Address error trap has occurred
0 = Address error trap has not occurred

bit 2 **STKERR:** Stack Error Trap Status bit
1 = Stack error trap has occurred
0 = Stack error trap has not occurred

bit 1 **OSCFAIL:** Oscillator Failure Trap Status bit
1 = Oscillator failure trap has occurred
0 = Oscillator failure trap has not occurred

bit 0 **Unimplemented:** Read as '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

REGISTER 6-4: INTCON2: INTERRUPT CONTROL REGISTER 2

Upper Byte:							
R/W-0	R-0	U-0	U-0	U-0	U-0	U-0	U-0
ALTIVT	DISI	—	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	INT4EP	INT3EP	INT2EP	INT1EP	INT0EP
bit 7							bit 0

bit 15 **ALTIVT:** Enable Alternate Interrupt Vector Table bit
1 = Use alternate vector table
0 = Use standard (default) vector table

bit 14 **DISI:** DISI Instruction Status bit
1 = DISI instruction is active
0 = DISI is not active

bit 13-5 **Unimplemented:** Read as '0'

bit 4 **INT4EP:** External Interrupt 4 Edge Detect Polarity Select bit
1 = Interrupt on negative edge
0 = Interrupt on positive edge

bit 3 **INT3EP:** External Interrupt 3 Edge Detect Polarity Select bit
1 = Interrupt on negative edge
0 = Interrupt on positive edge

bit 2 **INT2EP:** External Interrupt 2 Edge Detect Polarity Select bit
1 = Interrupt on negative edge
0 = Interrupt on positive edge

bit 1 **INT1EP:** External Interrupt 1 Edge Detect Polarity Select bit
1 = Interrupt on negative edge
0 = Interrupt on positive edge

bit 0 **INT0EP:** External Interrupt 0 Edge Detect Polarity Select bit
1 = Interrupt on negative edge
0 = Interrupt on positive edge

.

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

REGISTER 6-5: IFS0: INTERRUPT FLAG STATUS REGISTER 0

Upper Byte:							
U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	AD1IF	U1TXIF	U1RXIF	SPI1IF	SPF1IF	T3IF
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
T2IF	OC2IF	IC2IF	—	T1IF	OC1IF	IC1IF	INT0IF
bit 7							bit 0

bit 15,14 **Unimplemented:** Read as '0'

bit 13 **AD1IF:** A/D Conversion Complete Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 12 **U1TXIF:** UART1 Transmitter Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 11 **U1RXIF:** UART1 Receiver Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 10 **SPI1IF:** SPI1 Event Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 9 **SPF1IF:** SPI1 Fault Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 8 **T3IF:** Timer3 Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 7 **T2IF:** Timer2 Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 6 **OC2IF:** Output Compare Channel 2 Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 5 **IC2IF:** Input Capture Channel 2 Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

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

bit 3 **T1IF:** Timer1 Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 2 **OC1IF:** Output Compare Channel 1 Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 1 **IC1IF:** Input Capture Channel 1 Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 0 **INT0IF:** External Interrupt 0 Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

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

REGISTER 6-6: IFS1: INTERRUPT FLAG STATUS REGISTER 1

Upper Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0
U2TXIF	U2RXIF	INT2IF	T5IF	T4IF	OC4IF	OC3IF	—
bit 15							bit 8

Lower Byte:							
U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	INT1IF	CNIF	CMIF	MI2C1IF	SI2C1IF
bit 7							bit 0

bit 15 **U2TXIF:** UART2 Transmitter Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 14 **U2RXIF:** UART2 Receiver Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 13 **INT2IF:** External Interrupt 2 Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 12 **T5IF:** Timer5 Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 11 **T4IF:** Timer4 Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 10 **OC4IF:** Output Compare Channel 4 Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 9 **OC3IF:** Output Compare Channel 3 Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 8-5 **Unimplemented:** Read as '0'

bit 4 **INT1IF:** External Interrupt 1 Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 3 **CNIF:** Input Change Notification Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 2 **CMIF:** Comparator Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 1 **MI2C1IF:** Master I2C1 Event Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 0 **SI2C1IF:** Slave I2C1 Event Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

.

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

REGISTER 6-7: IFS2: INTERRUPT FLAG STATUS REGISTER 2

Upper Byte:							
U-0	U-0	R/W-0	U-0	U-0	U-0	R/W-0	U-0
—	—	PMPIF	—	—	—	OC5IF	—
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	R/W-0	R/W-0
IC5IF	IC4IF	IC3IF	—	—	—	SPI2IF	SPF2IF
bit 7							bit 0

bit 15-14 **Unimplemented:** Read as '0'

bit 13 **PMPIF:** Parallel Master Port Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 12-10 **Unimplemented:** Read as '0'

bit 9 **OC5IF:** Output Compare Channel 5 Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 8 **Unimplemented:** Read as '0'

bit 7 **IC5IF:** Input Capture Channel 5 Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 6 **IC4IF:** Input Capture Channel 4 Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 5 **IC3IF:** Input Capture Channel 3 Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 4-2 **Unimplemented:** Read as '0'

bit 1 **SPI2IF:** SPI2 Event Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 0 **SPI2IF:** SPI2 Fault Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

.

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

REGISTER 6-8: IFS3: INTERRUPT FLAG STATUS REGISTER 3

Upper Byte:							
U-0	R/W-0	U-0	U-0	U-0	U-0	U-0	U-0
—	RTCIF	—	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
U-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	U-0
—	INT4IF	INT3IF	—	—	MI2C2IF	SI2C2IF	—
bit 7							bit 0

bit 15 **Unimplemented:** Read as '0'

bit 14 **RTCIF:** Real-Time Clock/Calendar Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 13-7 **Unimplemented:** Read as '0'

bit 6 **INT4IF:** External Interrupt 4 Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 5 **INT3IF:** External Interrupt 3 Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 4-3 **Unimplemented:** Read as '0'

bit 2 **MI2C2IF:** Master I2C2 Event Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 1 **SI2C2IF:** Slave I2C2 Event Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 0 **Unimplemented:** Read as '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

REGISTER 6-9: IFS4: INTERRUPT FLAG STATUS REGISTER 4

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	U-0
—	—	—	—	CRCIF	U2ERIF	U1ERIF	—
bit 7							bit 0

bit 15-4 **Unimplemented:** Read as '0'

bit 3 **CRCIF:** CRC Generator Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 2 **U2ERIF:** UART2 Error Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 1 **U1ERIF:** UART1 Error Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 0 **Unimplemented:** Read as '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

REGISTER 6-10: IEC0: INTERRUPT ENABLE CONTROL REGISTER 0

Upper Byte:							
U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	AD1IE	U1TXIE	U1RXIE	SPI1IE	SPF1IE	T3IE
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
T2IE	OC2IE	IC2IE	—	T1IE	OC1IE	IC1IE	INT0IE
bit 7							bit 0

bit 15-14 **Unimplemented:** Read as '0'

bit 13 **AD1IE:** A/D Conversion Complete Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 12 **U1TXIE:** UART1 Transmitter Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 11 **U1RXIE:** UART1 Receiver Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 10 **SPI1IE:** SPI1 Transfer Complete Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 9 **SPF1IE:** SPI1 Fault Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 8 **T3IE:** Timer3 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 7 **T2IE:** Timer2 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 6 **OC2IE:** Output Compare Channel 2 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 5 **IC2IE:** Input Capture Channel 2 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

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

bit 3 **T1IE:** Timer1 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 2 **OC1IE:** Output Compare Channel 1 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 1 **IC1IE:** Input Capture Channel 1 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 0 **INT0IE:** External Interrupt 0 Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

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

REGISTER 6-11: IEC1: INTERRUPT ENABLE CONTROL REGISTER 1

Upper Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0
U2TXIE	U2RXIE	INT2IE	T5IE	T4IE	OC4IE	OC3IE	—
bit 15							bit 8

Lower Byte:							
U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	INT1IE	CNIE	CMIE	MI2C1IE	SI2C1IE
bit 7							bit 0

bit 15 **U2TXIE:** UART2 Transmitter Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 14 **U2RXIE:** UART2 Receiver Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 13 **INT2IE:** External Interrupt 2 Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 12 **T5IE:** Timer5 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 11 **T4IE:** Timer4 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 10 **OC4IE:** Output Compare Channel 4 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 9 **OC3IE:** Output Compare Channel 3 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 8-5 **Unimplemented:** Read as '0'

bit 4 **INT1IE:** External Interrupt 1 Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 3 **CNIE:** Input Change Notification Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 2 **CMIE:** Comparator Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 1 **MI2C1IE:** Master I2C1 Event Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 0 **SI2C1IE:** Slave I2C1 Event Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

.

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

REGISTER 6-12: IEC2: INTERRUPT ENABLE CONTROL REGISTER 2

Upper Byte:							
U-0	U-0	R/W-0	U-0	U-0	U-0	R/W-0	U-0
—	—	PMPIE	—	—	—	OC5IE	—
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	R/W-0	R/W-0
IC5IE	IC4IE	IC3IE	—	—	—	SPI2IE	SPF2IE
bit 7							bit 0

bit 15-14 **Unimplemented:** Read as '0'

bit 13 **PMPIE:** Parallel Master Port Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 12-10 **Unimplemented:** Read as '0'

bit 9 **OC5IE:** Output Compare Channel 5 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 8 **Unimplemented:** Read as '0'

bit 7 **IC5IE:** Input Capture Channel 5 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 6 **IC4IE:** Input Capture Channel 4 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 5 **IC3IE:** Input Capture Channel 3 Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 4-2 **Unimplemented:** Read as '0'

bit 1 **SPI2IE:** SPI2 Event Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 0 **SPF2IE:** SPI2 Fault Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

.

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

REGISTER 6-13: IEC3: INTERRUPT ENABLE CONTROL REGISTER 3

Upper Byte:							
U-0	R/W-0	U-0	U-0	U-0	U-0	U-0	U-0
—	RTCIE	—	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
U-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	U-0
—	INT4IE	INT3IE	—	—	MI2C2IE	SI2C2IE	—
bit 7							bit 0

bit 15 **Unimplemented:** Read as '0'

bit 14 **RTCIE:** Real-Time Clock/Calendar Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 13-7 **Unimplemented:** Read as '0'

bit 6 **INT4IE:** External Interrupt 4 Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 5 **INT3IE:** External Interrupt 3 Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 4-3 **Unimplemented:** Read as '0'

bit 2 **MI2C2IE:** Master I2C2 Event Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 1 **SI2C2IE:** Slave I2C2 Event Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 0 **Unimplemented:** Read as '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

REGISTER 6-14: IEC4: INTERRUPT ENABLE CONTROL REGISTER 4

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	U-0
—	—	—	—	CRCIE	U2ERIE	U1ERIE	—
bit 7							bit 0

bit 15-4 **Unimplemented:** Read as '0'

bit 3 **CRCIE:** CRC Generator Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 2 **U2ERIE:** UART2 Error Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 1 **U1ERIE:** UART1 Error Interrupt Enable bit
1 = Interrupt request enabled
0 = Interrupt request not enabled

bit 0 **Unimplemented:** Read as '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

REGISTER 6-15: IPC0: INTERRUPT PRIORITY CONTROL REGISTER 0

Upper Byte:							
U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	T1IP2	T1IP1	T1IP0	—	OC1IP2	OC1IP1	OC1IP0
bit 15							bit 8

Lower Byte:							
U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	IC1IP2	IC1IP1	IC1IP0	—	INT0IP2	INT0IP1	INT0IP0
bit 7							bit 0

bit 15 **Unimplemented:** Read as '0'

bit 14-12 **T1IP2:T1IP0:** Timer1 Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 11 **Unimplemented:** Read as '0'

bit 10-8 **OC1IP2:OC1IP0:** Output Compare Channel 1 Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

bit 6-4 **IC1IP2:IC1IP0:** Input Capture Channel 1 Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

bit 2-0 **INT0IP2:INT0IP0:** External Interrupt 0 Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

.

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

REGISTER 6-16: IPC1: INTERRUPT PRIORITY CONTROL REGISTER 1

Upper Byte:							
U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	T2IP2	T2IP1	T2IP0	—	OC2IP2	OC2IP1	OC2IP0
bit 15							bit 8

Lower Byte:							
U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	IC2IP2	IC2IP1	IC2IP0	—	—	—	—
bit 7							bit 0

bit 15 **Unimplemented:** Read as '0'

bit 14-12 **T2IP2:T2IP0:** Timer2 Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 11 **Unimplemented:** Read as '0'

bit 10-8 **OC2IP2:OC2IP0:** Output Compare Channel 2 Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

bit 6-4 **IC2IP2:IC2IP0:** Input Capture Channel 2 Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

REGISTER 6-17: IPC2: INTERRUPT PRIORITY CONTROL REGISTER 2

Upper Byte:							
U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	U1RXIP2	U1RXIP1	U1RXIP0	—	SPI1IP2	SPI1IP1	SPI1IP0
bit 15							bit 8

Lower Byte:							
U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	SPF1IP2	SPF1IP1	SPF1IP0	—	T3IP2	T3IP1	T3IP0
bit 7							bit 0

bit 15 **Unimplemented:** Read as '0'

bit 14-12 **U1RXIP2:U1RXIP0:** UART1 Receiver Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 11 **Unimplemented:** Read as '0'

bit 10-8 **SPI1IP2:SPI1IP0:** SPI1 Event Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

bit 6-4 **SPF1IP2:SPF1IP0:** SPI1 Fault Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

bit 2-0 **T3IP2:T3IP0:** Timer3 Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

.

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

REGISTER 6-18: IPC3: INTERRUPT PRIORITY CONTROL REGISTER 3

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	AD1IP2	AD1IP1	AD1IP0	—	U1TXIP2	U1TXIP1	U1TXIP0
bit 7							bit 0

bit 15-7 **Unimplemented:** Read as '0'

bit 6-4 **AD1IP2:AD1IP0:** A/D Conversion Complete Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

bit 2-0 **U1TXIP2:U1TXIP0:** UART1 Transmitter Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

.

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

REGISTER 6-19: IPC4: INTERRUPT PRIORITY CONTROL REGISTER 4

Upper Byte:							
U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	CNIP2	CNIP1	CNIP0	—	CMIP2	CMIP1	CMIP0
bit 15							bit 8

Lower Byte:							
U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	MI2C1P2	MI2C1P1	MI2C1P0	—	SI2C1P2	SI2C1P1	SI2C1P0
bit 7							bit 0

bit 15 **Unimplemented:** Read as '0'

bit 14-12 **CNIP2:CNIP0:** Input Change Notification Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 11 **Unimplemented:** Read as '0'

bit 10-8 **CMIP2:CMIP0:** Comparator Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

bit 6-4 **MI2C1P2:MI2C1P0:** Master I2C1 Event Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

bit 2-0 **SI2C1P2:SI2C1P0:** Slave I2C1 Event Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

.

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

REGISTER 6-20: IPC5: INTERRUPT PRIORITY CONTROL REGISTER 5

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
U-0	U-0	U-0	U-0	U-0	R/W-1	R/W-0	R/W-0
—	—	—	—	—	INT1IP2	INT1IP1	INT1IP0
bit 7							bit 0

bit 15-3 **Unimplemented:** Read as '0'

bit 2-0 **INT1IP2:INT1IP0:** External Interrupt 1 Priority bits

111 = Interrupt is priority 7 (highest priority interrupt)

•

•

•

001 = Interrupt is priority 1

000 = Interrupt source is disabled

.

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

REGISTER 6-21: IPC6: INTERRUPT PRIORITY CONTROL REGISTER 6

Upper Byte:							
U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	T4IP2	T4IP1	T4IP0	—	OC4IP2	OC4IP1	OC4IP0
bit 15							bit 8

Lower Byte:							
U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	OC3IP2	OC3IP1	OC3IP0	—	—	—	—
bit 7							bit 0

bit 15 **Unimplemented:** Read as '0'

bit 14-12 **T4IP2:T4IP0:** Timer4 Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 11 **Unimplemented:** Read as '0'

bit 10-8 **OC4IP2:OC4IP0:** Output Compare Channel 4 Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

bit 6-4 **OC3IP2:OC3IP0:** Output Compare Channel 3 Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

REGISTER 6-22: IPC7: INTERRUPT PRIORITY CONTROL REGISTER 7

Upper Byte:							
U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	U2TXIP2	U2TXIP1	U2TXIP0	—	U2RXIP2	U2RXIP1	U2RXIP0
bit 15							bit 8

Lower Byte:							
U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	INT2IP2	INT2IP1	INT2IP0	—	T5IP2	T5IP1	T51P0
bit 7							bit 0

bit 15 **Unimplemented:** Read as '0'

bit 14-12 **U2TXIP2:U2TXIP0:** UART2 Transmitter Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 11 **Unimplemented:** Read as '0'

bit 10-8 **U2RXIP2:U2RXIP0:** UART2 Receiver Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

bit 6-4 **INT2IP2:INT2IP0:** External Interrupt 2 Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

bit 2-0 **T5IP2:T5IP0:** Timer5 Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

.

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

REGISTER 6-23: IPC8: INTERRUPT PRIORITY CONTROL REGISTER 8

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	SPI2IP2	SPI2IP1	SPI2IP0	—	SPF2IP2	SPF2IP1	SPF2IP0
bit 7							bit 0

bit 15-7 **Unimplemented:** Read as '0'

bit 6-4 **SPI2IP2:SPI2IP0:** SPI2 Event Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

bit 2-0 **SPF2IP2:SPF2IP0:** SPI2 Fault Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

.

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

REGISTER 6-24: IPC9: INTERRUPT PRIORITY CONTROL REGISTER 9

Upper Byte:							
U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	IC5IP2	IC5IP1	IC5IP0	—	IC4IP2	IC4IP1	IC4IP0
bit 15							bit 8

Lower Byte:							
U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	IC3IP2	IC3IP1	IC3IP0	—	—	—	—
bit 7							bit 0

bit 15 **Unimplemented:** Read as '0'

bit 14-12 **IC5IP2:IC5IP0:** Input Capture Channel 5 Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 11 **Unimplemented:** Read as '0'

bit 10-8 **IC4IP2:IC4IP0:** Input Capture Channel 4 Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

bit 6-4 **IC3IP2:IC3IP0:** Input Capture Channel 3 Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

REGISTER 6-25: IPC10: INTERRUPT PRIORITY CONTROL REGISTER 10

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	OC5IP2	OC5IP1	OC5IP0	—	—	—	—
bit 7							bit 0

bit 15-7 **Unimplemented:** Read as '0'

bit 6-4 **OC5IP2:OC5IP0:** Output Compare Channel 5 Interrupt Priority bits

111 = Interrupt is priority 7 (highest priority interrupt)

•

•

•

001 = Interrupt is priority 1

000 = Interrupt source is disabled

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

REGISTER 6-26: IPC11: INTERRUPT PRIORITY CONTROL REGISTER 11

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	PMPIP2	PMPIP1	PMPIP0	—	—	—	—
bit 7							bit 0

bit 15-7 **Unimplemented:** Read as '0'

bit 6-4 **PMPIP2:PMPIP0:** Parallel Master Port Interrupt Priority bits

111 = Interrupt is priority 7 (highest priority interrupt)

•

•

•

001 = Interrupt is priority 1

000 = Interrupt source is disabled

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

REGISTER 6-27: IPC12: INTERRUPT PRIORITY CONTROL REGISTER 12

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	R/W-1	R/W-0	R/W-0
—	—	—	—	—	MI2C2P2	MI2C2P1	MI2C2P0
bit 15							bit 8

Lower Byte:							
U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	SI2C2P2	SI2C2P1	SI2C2P0	—	—	—	—
bit 7							bit 0

bit 15-11 **Unimplemented:** Read as '0'

bit 10-8 **MI2C2P2:MI2C2P0:** Master I2C2 Event Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

bit 6-4 **SI2C2P2:SI2C2P0:** Slave I2C2 Event Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

REGISTER 6-28: IPC13: INTERRUPT PRIORITY CONTROL REGISTER 13

Upper Byte:

U-0	U-0	U-0	U-0	U-0	R/W-1	R/W-0	R/W-0
—	—	—	—	—	INT4IP2	INT4IP1	INT4IP0
bit 15							bit 8

Lower Byte:

U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	INT3IP2	INT3IP1	INT3IP0	—	—		

REGISTER 6-29: IPC15: INTERRUPT PRIORITY CONTROL REGISTER 15

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	R/W-1	R/W-0	R/W-0
—	—	—	—	—	RTCIP2	RTCIP1	RTCIP0
bit 15							bit 8

Lower Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 7							bit 0

bit 15-11 **Unimplemented:** Read as '0'

bit 10-8 **RTCIP2:RTCIP0:** Real-Time Clock/Calendar Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

REGISTER 6-30: IPC16: INTERRUPT PRIORITY CONTROL REGISTER 16

Upper Byte:							
U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	CRCIP2	CRCIP1	CRCIP0	—	U2ERIP2	U2ERIP1	U2ERIP0
bit 15							bit 8

Lower Byte:							
U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	U1ERIP2	U1ERIP1	U1ERIP0	—	—	—	—
bit 7							bit 0

bit 15 **Unimplemented:** Read as '0'

bit 14-12 **CRCIP2:CRCIP0:** CRC Generator Error Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

bit 11 **Unimplemented:** Read as '0'

bit 10-8 **U2ERIP2:U2ERIP0:** UART2 Error Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

bit 6-4 **U1ERIP2:U1ERIP0:** UART1 Error Interrupt Priority bits
111 = Interrupt is priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is priority 1
000 = Interrupt source is disabled

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

6.4 Interrupt Setup Procedures

6.4.1 INITIALIZATION

To configure an interrupt source:

1. Set the NSTDIS Control bit (INTCON1<15>) if nested interrupts are not desired.
2. Select the user-assigned priority level for the interrupt source by writing the control bits in the appropriate IPCx Control register. The priority level will depend on the specific application and type of interrupt source. If multiple priority levels are not desired, the IPCx register control bits for all enabled interrupt sources may be programmed to the same non-zero value.

Note: At a device Reset, the IPC registers are initialized, such that all user interrupt sources are assigned to priority level 4.

3. Clear the interrupt flag status bit associated with the peripheral in the associated IFSx Status register.
4. Enable the interrupt source by setting the interrupt enable control bit associated with the source in the appropriate IECx Control register.

6.4.2 INTERRUPT SERVICE ROUTINE

The method that is used to declare an ISR and initialize the IVT with the correct vector address will depend on the programming language (i.e., 'C' or assembler) and the language development toolsuite that is used to develop the application. In general, the user must clear the interrupt flag in the appropriate IFSx register for the source of interrupt that the ISR handles. Otherwise, the ISR will be re-entered immediately after exiting the routine. If the ISR is coded in assembly language, it must be terminated using a `RETFIE` instruction to unstack the saved PC value, SRL value and old CPU priority level.

6.4.3 TRAP SERVICE ROUTINE

A Trap Service Routine (TSR) is coded like an ISR, except that the appropriate trap status flag in the INTCON1 register must be cleared to avoid re-entry into the TSR.

6.4.4 INTERRUPT DISABLE

All user interrupts can be disabled using the following procedure:

1. Push the current SR value onto the software stack using the `PUSH` instruction.
2. Force the CPU to priority level 7 by inclusive ORing the value OEh with SRL.

To enable user interrupts, the `POP` instruction may be used to restore the previous SR value.

Note that only user interrupts with a priority level of 7 or less can be disabled. Trap sources (level 8-15) cannot be disabled.

The `DISI` instruction provides a convenient way to disable interrupts of priority levels 1-6 for a fixed period of time. Level 7 interrupt sources are not disabled by the `DISI` instruction.

NOTES:

7.0 OSCILLATOR CONFIGURATION

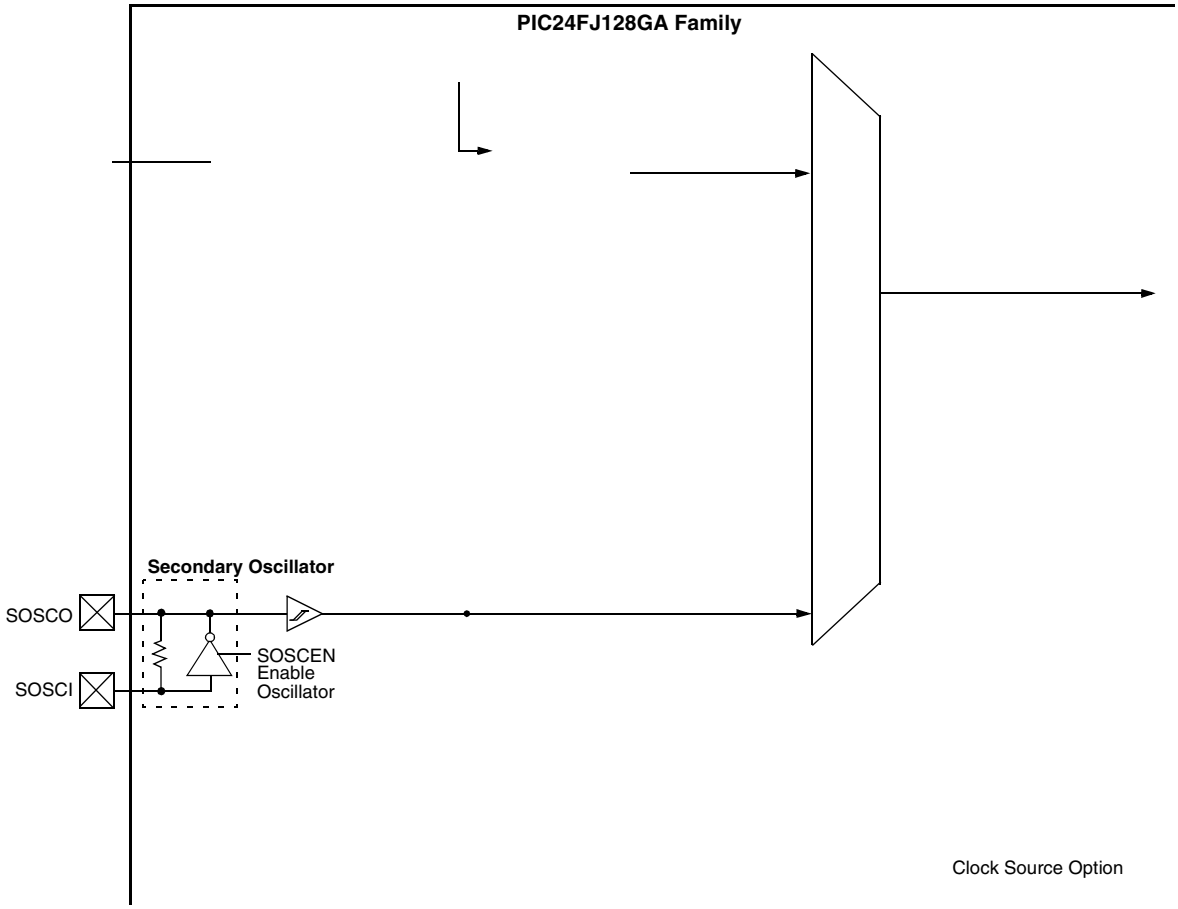
Note:	This data sheet summarizes the features of this group of PIC24FJ devices. It is not intended to be a comprehensive reference source.

The oscillator system for PIC24FJ128GA family devices has the following features:

- A total of four external and internal oscillator options as clock sources, providing 11 different clock modes
- On-chip 4x PLL to boost internal operating frequency on select internal and external oscillator sources
- Software-controllable switching between various clock sources
- Software-controllable postscaler for selective clocking of CPU for system power savings
- A Fail-Safe Clock Monitor (FSCM) that detects clock failure and permits safe application recovery or shutdown

A simplified diagram of the oscillator system is shown in Figure 7-1.

FIGURE 7-1: PIC24FJ128GA FAMILY CLOCK DIAGRAM



7.1 CPU Clocking Scheme

The system clock source can be provided by one of four sources:

- Primary Oscillator (POSC) on the OSC1 and OSC2 pins
- Secondary Oscillator (SOSC) on the SOSCI and SOSCO pins
- Fast Internal RC (FRC) Oscillator
- Low-Power Internal RC (LPRC) Oscillator

The primary oscillator and FRC sources have the option of using the internal 4x PLL. The frequency of the FRC clock source can optionally be reduced by the programmable clock divider. The selected clock source generates the processor and peripheral clock sources.

The processor clock source is divided by two to produce the internal instruction cycle clock, FCY. In this document, the instruction cycle clock is also denoted by FOSC/2. The internal instruction cycle clock, FOSC/2, can be provided on the OSC2 I/O pin for some operating modes of the primary oscillator.

7.2 Oscillator Configuration

The oscillator source (and operating mode) that is used at a device Power-on Reset event is selected using Configuration bit settings. The oscillator Configuration bit settings are located in the Configuration registers in the program memory (refer to **Section 23.1 "Configuration Bits"** for further details.) The Primary Oscillator Configuration bits, POSCMD1:POSCMD0 (Configuration Word 2<1:0>), and the Initial Oscillator Select Configuration bits, FNOSC2:FNOSC0 (Configuration Word 2<10:8>), select the oscillator source that is used at a Power-on Reset. The FRC primary oscillator with postscaler (FRCDIV) is the default (unprogrammed) selection. The secondary oscillator, or one of the internal oscillators, may be chosen by programming these bit locations.

The Configuration bits allow users to choose between the various clock modes, shown in Table 7-1.

7.2.1 CLOCK SWITCHING MODE CONFIGURATION BITS

The FCKSM Configuration bits (Configuration Word 2<7:6>) are used to jointly configure device clock switching and the Fail-Safe Clock Monitor (FSCM). Clock switching is enabled only when FCKSM1 is programmed ('0'). The FSCM is enabled only when FCKSM1:FCKSM0 are both programmed ('00').

TABLE 7-1: CONFIGURATION BIT VALUES FOR CLOCK SELECTION

Oscillator Mode	Oscillator Source	POSCMD1: POSCMD0	FNOSC2: FNOSC0	Note
Fast RC Oscillator with Postscaler (FRCDIV)	Internal	00	111	**1, 2**
(Reserved)	Internal	00	110	**1**
Low-Power RC Oscillator (LPRC)	Internal	00	101	**1**
Secondary (Timer1) Oscillator (SOSC)	Secondary	00	100	**1**
Primary Oscillator (HS) with PLL Module (HSPLL)	Primary	10	011	
Primary Oscillator (XT) with PLL Module (ECPLL)	Primary	01	011	
Primary Oscillator (EC) with PLL Module (XTPLL)	Primary	00	011	
Primary Oscillator (HS)	Primary	10	010	
Primary Oscillator (XT)	Primary	01	010	
Primary Oscillator (EC)	Primary	00	010	
Fast RC Oscillator with PLL Module (FRCPLL)	Internal	00	001	**1**
Fast RC Oscillator (FRC)	Internal	00	000	**1**

Note 1: OSC2 pin function is determined by the OSCIOFNC Configuration bit.

2: This is the default oscillator mode for an unprogrammed (erased) device.

7.3 Control Registers

The operation of the oscillator is controlled by three Special Function Registers:

- OSCCON
- CLKDIV
- OSCTUN

The OSCCON register (Register 7-1) is the main control register for the oscillator. It controls clock source switching, and allows the monitoring of clock sources.

The Clock Divider register (Register 7-2) controls the features associated with Doze mode, as well as the postscaler for the FRC oscillator.

The FRC Oscillator Tune register (Register 7-3) allows the user to fine tune the FRC oscillator over a range of approximately ±12%. Each bit increment or decrement changes the factory calibrated frequency of the FRC oscillator by a fixed amount.

REGISTER 7-1: OSCCON: OSCILLATOR CONTROL REGISTER

Upper Byte:							
U-0	R-0	R-0	R-0	U-0	R/W-x(1)	R/W-x(1)	R/W-x(1)
—	COSC2	COSC1	COSC0	—	NOSC2	NOSC1	NOSC0
bit 15							bit 8

Lower Byte:							
R/SO-0	U-0	R-0(2)	U-0	R/CO-0	U-0	R/W-0	R/W-0
CLKLOCK	—	LOCK	—	CF	—	SOSCEN	OSWEN
bit 7							bit 0

bit 15 **Unimplemented:** Read as '0'

bit 14-12 **COSC2:COSC0:** Current Oscillator Selection bits
- 111 = Fast RC Oscillator with Postscaler (FRCDIV)
- 110 = Reserved
- 101 = Low-Power RC Oscillator (LPRC)
- 100 = Secondary Oscillator (SOSC)
- 011 = Primary Oscillator with PLL module (HSPLL, ECPLL)
- 010 = Primary Oscillator (XT, HS, EC)
- 001 = Fast RC Oscillator with postscaler and PLL module (FRCPLL)
- 000 = Fast RC Oscillator (FRC)

bit 11 **Unimplemented:** Read as '0'

bit 10-8 **NOSC2:NOSC0:** New Oscillator Selection bits
- 111 = Fast RC Oscillator with Postscaler (FRCDIV)
- 110 = Reserved
- 101 = Low-Power RC Oscillator (LPRC)
- 100 = Secondary Oscillator (SOSC)
- 011 = Primary Oscillator with PLL module (HSPLL, ECPLL)
- 010 = Primary Oscillator (XT, HS, EC)
- 001 = Fast RC Oscillator with postscaler and PLL module (FRCPLL)
- 000 = Fast RC Oscillator (FRC)

bit 7 **CLKLOCK:** Clock Selection Lock Enabled bit

If FSCM is enabled (FCKSM1 = 1):
- 1 = Clock and PLL selections are locked
- 0 = Clock and PLL selections are not locked and may be modified by setting the OSWEN bit

If FSCM is disabled (FCKSM1 = 0):
Clock and PLL selections are never locked and may be modified by setting the OSWEN bit.

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

bit 5 **LOCK:** PLL Lock Status bit
- 1 = PLL module is in lock or PLL module start-up timer is satisfied
- 0 = PLL module is out of lock, PLL start-up timer is running or PLL is disabled

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

bit 3 **CF:** Clock Fail Detect bit
- 1 = FSCM has detected a clock failure
- 0 = No clock failure has been detected

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

bit 1 **SOSCEN:** 32 kHz Secondary Oscillator (SOSC) Enable bit
- 1 = Enable secondary oscillator
- 0 = Disable secondary oscillator

bit 0 **OSWEN:** Oscillator Switch Enable bit
- 1 = Initiate an oscillator switch to clock source specified by NOSC2:NOSC0 bits
- 0 = Oscillator switch is complete

Note 1: Reset values for these bits are determined by the FNOSC Configuration bits.
2: Also resets to '0' during any valid clock switch, or whenever a non-PLL Clock mode is selected.

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

REGISTER 7-2: CLKDIV: CLOCK DIVIDER REGISTER

Upper Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-1
ROI	DOZE2	DOZE1	DOZE0	DOZEN(1)	RCDIV2	RCDIV1	RCDIV0
bit 15							bit 8

Lower Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 7							bit 0

bit 15 **ROI:** Recover on Interrupt bit
1 = Interrupts clear the DOZEN bit and reset the CPU peripheral clock ratio to 1:1
0 = Interrupts have no effect on the DOZEN bit

bit 14-12 **DOZE2:DOZE0:** CPU Peripheral Clock Ratio Select bits
111 = 1:128
110 = 1:64
101 = 1:32
100 = 1:16
011 = 1:8
010 = 1:4
001 = 1:2
000 = 1:1

bit 11 **DOZEN:** DOZE Enable bit(1)
1 = DOZE2:DOZE0 bits specify the CPU peripheral clock ratio
0 = CPU peripheral clock ratio set to 1:1

Note 1: This bit is automatically cleared when the ROI bit is set and an interrupt occurs.

bit 10-8 **RCDIV2:RCDIV0:** FRC Postscaler Select bits
111 = 31.25 kHz (divide by 256)
110 = 125 kHz (divide by 64)
101 = 250 kHz (divide by 32)
100 = 500 kHz (divide by 16)
011 = 1 MHz (divide by 8)
010 = 2 MHz (divide by 4)
001 = 4 MHz (divide by 2)
000 = 8 MHz (divide by 1)

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

REGISTER 7-3: OSCTUN: FRC OSCILLATOR TUNE REGISTER

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
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

bit 15-6 **Unimplemented:** Read as '0'

bit 5-0 **TUN5:TUN0:** FRC Oscillator Tuning bits

011111 = Maximum frequency deviation
011110 =
•
•
•
000001 =
000000 = Center frequency, oscillator is running at factory calibrated frequency
111111 =
•
•
•
100001 =
100000 = Minimum frequency deviation

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

7.4 Clock Switching Operation

With few limitations, applications are free to switch between any of the four clock sources (POSC, SOSC, FRC and LPRC) under software control and at any time. To limit the possible side effects that could result from this flexibility, PIC24 devices have a safeguard lock built into the switching process.

Note: Primary Oscillator mode has three different submodes (XT, HS and EC) which are determined by the POSCMD Configuration bits. While an application can switch to and from Primary Oscillator mode in software, it cannot switch between the different primary submodes without reprogramming the device.

7.4.1 ENABLING CLOCK SWITCHING

To enable clock switching, the FCKSM1 Configuration bit in the Configuration register must be programmed to '0'. (Refer to **Section 23.1 "Configuration Bits"** for further details.) If the FCKSM1 Configuration bit is unprogrammed ('1'), the clock switching function and Fail-Safe Clock Monitor function are disabled. This is the default setting.

The NOSC control bits (OSCCON<10:8>) do not control the clock selection when clock switching is disabled. However, the COSC bits (OSCCON<14:12>) will reflect the clock source selected by the FNOSC Configuration bits.

The OSWEN control bit (OSCCON<0>) has no effect when clock switching is disabled. It is held at '0' at all times.

7.4.2 OSCILLATOR SWITCHING SEQUENCE

At a minimum, performing a clock switch requires this basic sequence:

1. If desired, read the COSC bits (OSCCON<14:12>), to determine the current oscillator source.
2. Perform the unlock sequence to allow a write to the OSCCON register high byte.
3. Write the appropriate value to the NOSC control bits (OSCCON<10:8>) for the new oscillator source.
4. Perform the unlock sequence to allow a write to the OSCCON register low byte.
5. Set the OSWEN bit to initiate the oscillator switch.

Once the basic sequence is completed, the system clock hardware responds automatically as follows:

1. The clock switching hardware compares the COSC status bits with the new value of the NOSC control bits. If they are the same, then the clock switch is a redundant operation. In this case, the OSWEN bit is cleared automatically and the clock switch is aborted.
2. If a valid clock switch has been initiated, the LOCK (OSCCON<5>) and CF (OSCCON<3>) status bits are cleared.
3. The new oscillator is turned on by the hardware if it is not currently running. If a crystal oscillator must be turned on, the hardware will wait until the OST expires. If the new source is using the PLL, then the hardware waits until a PLL lock is detected (LOCK = 1).
4. The hardware waits for 10 clock cycles from the new clock source and then performs the clock switch.
5. The hardware clears the OSWEN bit to indicate a successful clock transition. In addition, the NOSC bit values are transferred to the COSC status bits.
6. The old clock source is turned off at this time, with the exception of LPRC (if WDT or FSCM are enabled) or SOSC (if SOSCEN remains set).

Note 1: The processor will continue to execute code throughout the clock switching sequence. Timing sensitive code should not be executed during this time.

2: Direct clock switches between any Primary Oscillator mode with PLL and FRCPLL mode are not permitted. This applies to clock switches in either direction. In these instances, the application must switch to FRC mode as a transition clock source between the two PLL modes.

A recommended code sequence for a clock switch includes the following:

1. Disable interrupts during the OSCCON register unlock and write sequence.
2. Execute the unlock sequence for the OSCCON high byte, by writing 78h and 9Ah to OSCCON<15:8> in two back-to-back instructions.
3. Write new oscillator source to the NOSC control bits in the instruction immediately following the unlock sequence.
4. Execute the unlock sequence for the OSCCON low byte by writing 46h and 57h to OSCCON<7:0> in two back-to-back instructions.
5. Set the OSWEN bit in the instruction immediately following the unlock sequence.
6. Continue to execute code that is not clock sensitive (optional).
7. Invoke an appropriate amount of software delay (cycle counting) to allow the selected oscillator and/or PLL to start and stabilize.
8. Check to see if OSWEN is '0'. If it is, the switch was successful. If OSWEN is still set, then check the LOCK bit to determine cause of failure.

The core sequence for unlocking the OSCCON register and initiating a clock switch is shown in Example 7-1.

EXAMPLE 7-1: BASIC CODE SEQUENCE FOR CLOCK SWITCHING

```
;Place the new oscillator selection in W0
;OSCCONH (high byte) Unlock Sequence
MOV         #OSCCONH, w1
MOV         #0x78, w2
MOV         #0x9A, w3
MOV.b       w2, [w1]
MOV.b       w3, [w1]
;Set new oscillator selection
MOV.b       WREG, OSCCONH
;OSCCONL (low byte) unlock sequence
MOV         #OSCCONL, w1
MOV.b       #0x01, w0
MOV         #0x46, w2
MOV         #0x57, w3
MOV.b       w2, [w1]
MOV.b       w3, [w1]
;Start oscillator switch operation
MOV.b       w0, [w1]
```


8.0 POWER-SAVING FEATURES

Note: This data sheet summarizes the features of this group of PIC24FJ devices. It is not intended to be a comprehensive reference source.

The PIC24FJ128GA family of devices provide the ability to manage power consumption by selectively managing clocking to the CPU and the peripherals. In general, a lower clock frequency and a reduction in the number of circuits being clocked constitutes lower consumed power. All PIC24F devices manage power consumption in four different ways:

- Clock frequency
- Instruction-based Sleep and Idle modes
- Software-controlled Doze mode
- Selective peripheral control in software

Combinations of these methods can be used to selectively tailor an application's power consumption, while still maintaining critical application features, such as timing sensitive communications.

8.1 Clock Frequency and Clock Switching

PIC24F devices allow for a wide range of clock frequencies to be selected under application control. If the system clock configuration is not locked, users can choose low-power or high-precision oscillators by simply changing the NOSC Configuration bits. The process of changing a system clock during operation, as well as limitations to the process, are discussed in more detail in **Section 7.0 "Oscillator Configuration"**.

8.2 Instruction-Based Power-Saving Modes

PIC24F devices have two special power-saving modes that are entered through the execution of a special `PWRSAV` instruction. Sleep mode stops clock operation and halts all code execution; Idle mode halts the CPU and code execution, but allows peripheral modules to continue operation. The assembly syntax of the `PWRSAV` instruction is shown in Example 8-1.

Note: `SLEEP_MODE` and `IDLE_MODE` are constants defined in the assembler include file for the selected device.

Sleep and Idle modes can be exited as a result of an enabled interrupt, WDT time-out or a device Reset. When the device exits these modes, it is said to "wake-up".

8.2.1 SLEEP MODE

Sleep mode has these features:

- The system clock source is shut down. If an on-chip oscillator is used, it is turned off.
- The device current consumption will be reduced to a minimum provided that no I/O pin is sourcing current.
- The Fail-Safe Clock Monitor does not operate during Sleep mode since the system clock source is disabled.
- The LPRC clock will continue to run in Sleep mode if the WDT is enabled.
- The WDT, if enabled, is automatically cleared prior to entering Sleep mode.
- Some device features or peripherals may continue to operate in Sleep mode. This includes items such as the input change notification on the I/O ports, or peripherals that use an external clock input. Any peripheral that requires the system clock source for its operation will be disabled in Sleep mode.

The device will wake-up from Sleep mode on any of the these events:

- On any interrupt source that is individually enabled
- On any form of device Reset
- On a WDT time-out

On wake-up from Sleep, the processor will restart with the same clock source that was active when Sleep mode was entered.

EXAMPLE 8-1: `PWRSAV` INSTRUCTION SYNTAX

```
PWRSAV    #SLEEP_MODE       ; Put the device into SLEEP mode
PWRSAV    #IDLE_MODE        ; Put the device into IDLE mode
```

8.2.2 IDLE MODE

Idle mode has these features:

- The CPU will stop executing instructions.
- The WDT is automatically cleared.
- The system clock source remains active. By default, all peripheral modules continue to operate normally from the system clock source, but can also be selectively disabled (see **Section 8.4 "Selective Peripheral Module Control"**).
- If the WDT or FSCM is enabled, the LPRC will also remain active.

The device will wake from Idle mode on any of these events:

- Any interrupt that is individually enabled.
- Any device Reset.
- A WDT time-out.

On wake-up from Idle, the clock is re-applied to the CPU and instruction execution begins immediately, starting with the instruction following the `PWRSAV` instruction, or the first instruction in the ISR.

8.2.3 INTERRUPTS COINCIDENT WITH POWER SAVE INSTRUCTIONS

Any interrupt that coincides with the execution of a `PWRSAV` instruction will be held off until entry into Sleep or Idle mode has completed. The device will then wake-up from Sleep or Idle mode.

8.3 Doze Mode

Generally, changing clock speed and invoking one of the power-saving modes are the preferred strategies for reducing power consumption. There may be circumstances, however, where this is not practical. For example, it may be necessary for an application to maintain uninterrupted synchronous communication, even while it is doing nothing else. Reducing system clock speed may introduce communication errors, while using a power-saving mode may stop communications completely.

Doze mode is a simple and effective alternative method to reduce power consumption while the device is still executing code. In this mode, the system clock continues to operate from the same source and at the same speed. Peripheral modules continue to be clocked at the same speed, while the CPU clock speed is reduced. Synchronization between the two clock domains is maintained, allowing the peripherals to access the SFRs while the CPU executes code at a slower rate.

Doze mode is enabled by setting the DOZEN bit (CLKDIV<11>). The ratio between peripheral and core clock speed is determined by the DOZE2:DOZE0 bits (CLKDIV<14:12>). There are eight possible configurations, from 1:1 to 1:256, with 1:1 being the default.

It is also possible to use Doze mode to selectively reduce power consumption in event driven applications. This allows clock sensitive functions, such as synchronous communications, to continue without interruption while the CPU idles, waiting for something to invoke an interrupt routine. Enabling the automatic return to full-speed CPU operation on interrupts is enabled by setting the ROI bit (CLKDIV<15>). By default, interrupt events have no effect on Doze mode operation.

8.4 Selective Peripheral Module Control

Idle and Doze modes allow users to substantially reduce power consumption by slowing or stopping the CPU clock. Even so, peripheral modules still remain clocked and thus consume power. There may be cases where the application needs what these modes do not provide: the allocation of power resources to CPU processing with minimal power consumption from the peripherals.

PIC24F devices address this requirement by allowing peripheral modules to be selectively disabled, reducing or eliminating their power consumption. This can be done with two control bits:

- The Peripheral Enable bit, generically named "XXXEN", located in the module's main control SFR.
- The Peripheral Module Disable (PMD) bit, generically named "XXXMD", located in one of the PMD control registers.

Both bits have similar functions in enabling or disabling its associated module. Setting the PMD bit for a module disables all clock sources to that module, reducing its power consumption to an absolute minimum. In this state, the control and status registers associated with the peripheral will also be disabled, so writes to those registers will have no effect and read values will be invalid. Many peripheral modules have a corresponding PMD bit.

In contrast, disabling a module by clearing its XXXEN bit disables its functionality, but leaves its registers available to be read and written to. Power consumption is reduced, but not by as much as the PMD bit does. Most peripheral modules have an enable bit; exceptions include Capture, Compare and RTCC.

To achieve more selective power savings, peripheral modules can also be selectively disabled when the device enters Idle mode. This is done through the control bit of the generic name format "XXXIDL". By default, all modules that can operate during Idle mode will do so. Using the disable on Idle feature allows further reduction of power consumption during Idle mode, enhancing power savings for extremely critical power applications.

9.0 I/O PORTS

Note:	This data sheet summarizes the features of this group of PIC24FJ devices. It is not intended to be a comprehensive reference source.

All of the device pins (except VDD, VSS, MCLR and OSC1/CLKI) are shared between the peripherals and the parallel I/O ports. All I/O input ports feature Schmitt Trigger inputs for improved noise immunity.

9.1 Parallel I/O (PIO) Ports

A parallel I/O port that shares a pin with a peripheral is, in general, subservient to the peripheral. The peripheral's output buffer data and control signals are provided to a pair of multiplexers. The multiplexers select whether the peripheral or the associated port has ownership of the output data and control signals of the I/O pin. The logic also prevents "loop through", in which a port's digital output can drive the input of a peripheral that shares the same pin. Figure 9-1 shows how ports are shared with other peripherals and the associated I/O pin to which they are connected.

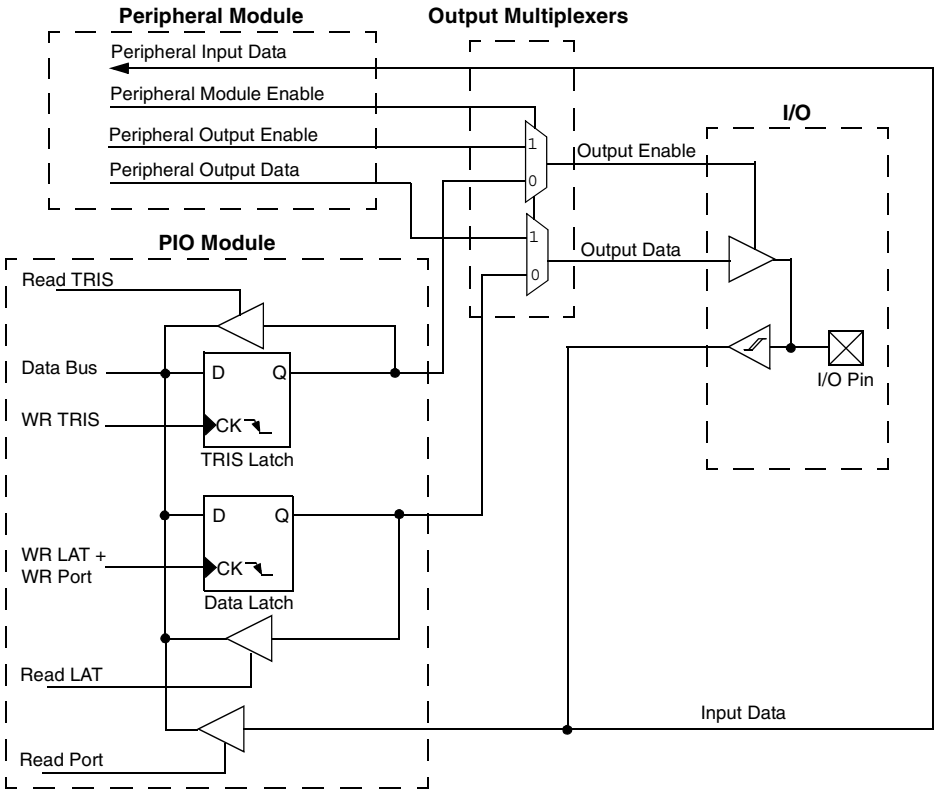
When a peripheral is enabled and the peripheral is actively driving an associated pin, the use of the pin as a general purpose output pin is disabled. The I/O pin may be read, but the output driver for the parallel port bit will be disabled. If a peripheral is enabled, but the peripheral is not actively driving a pin, that pin may be driven by a port.

All port pins have three registers directly associated with their operation as digital I/O. The data direction register (TRISx) determines whether the pin is an input or an output. If the data direction bit is a '1', then the pin is an input. All port pins are defined as inputs after a Reset. Reads from the latch (LATx), read the latch. Writes to the latch, write the latch. Reads from the port (PORTx), read the port pins, while writes to the port pins, write the latch.

Any bit and its associated data and control registers that are not valid for a particular device will be disabled. That means the corresponding LATx and TRISx registers and the port pin will read as zeros.

When a pin is shared with another peripheral or function that is defined as an input only, it is nevertheless regarded as a dedicated port because there is no other competing source of outputs. An example is the INT4 pin.

FIGURE 9-1: BLOCK DIAGRAM OF A TYPICAL SHARED PORT STRUCTURE



9.1.1 OPEN-DRAIN CONFIGURATION

In addition to the PORT, LAT and TRIS registers for data control, each port pin can also be individually configured for either digital or open-drain output. This is controlled by the Open-Drain Control register, ODCx, associated with each port. Setting any of the bits configures the corresponding pin to act as an open-drain output.

The open-drain feature allows the generation of outputs higher than VDD (e.g., 5V) on any desired digital-only pins by using external pull-up resistors. The maximum open-drain voltage allowed is the same as the maximum VIH specification.

9.2 Configuring Analog Port Pins

The use of the AD1PCFG and TRIS registers control the operation of the A/D port pins. The port pins that are desired as analog inputs must have their corresponding TRIS bit set (input). If the TRIS bit is cleared (output), the digital output level (VOH or VOL) will be converted.

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 not convert an analog input. Analog levels on any pin that is defined as a digital input (including the ANx pins) may cause the input buffer to consume current that exceeds the device specifications.

9.2.1 I/O PORT WRITE/READ TIMING

One instruction cycle is required between a port direction change or port write operation and a read operation of the same port. Typically this instruction would be a NOP.

EXAMPLE 9-1: PORT WRITE/READ EXAMPLE

```
MOV   0xFF00, W0          ; Configure PORTB<15:8> as inputs
MOV   W0, TRISBB          ; and PORTB<7:0> as outputs
NOP                       ; Delay 1 cycle
btss  PORTB, #13          ; Next Instruction
```

9.3 Input Change Notification

The input change notification function of the I/O ports allows the PIC24FJ128GA family of devices to generate interrupt requests to the processor in response to a change-of-state on selected input pins. This feature is capable of detecting input change-of-states even in Sleep mode, when the clocks are disabled. Depending on the device pin count, there are up to 22 external signals (CN0 through CN21) that may be selected (enabled) for generating an interrupt request on a change-of-state.

There are four control registers associated with the CN module. The CNEN1 and CNEN2 registers contain the interrupt enable control bits for each of the CN input pins. Setting any of these bits enables a CN interrupt for the corresponding pins.

Each CN pin also has a weak pull-up connected to it. The pull-ups act as a current source that is connected to the pin, and eliminate the need for external resistors when push button or keypad devices are connected. The pull-ups are enabled separately using the CNPU1 and CNPU2 registers, which contain the control bits for each of the CN pins. Setting any of the control bits enables the weak pull-ups for the corresponding pins.

Note:	Pull-ups on change notification pins should always be disabled whenever the port pin is configured as a digital output.

10.0 TIMER1

Note:	This data sheet summarizes the features of this group of PIC24FJ devices. It is not intended to be a comprehensive reference source.

The Timer1 module is a 16-bit timer which can serve as the time counter for the Real-Time Clock, or operate as a free-running interval timer/counter. Timer1 can operate in three modes:

- 16-bit Timer
- 16-bit Synchronous Counter
- 16-bit Asynchronous Counter

Timer1 also supports these features:

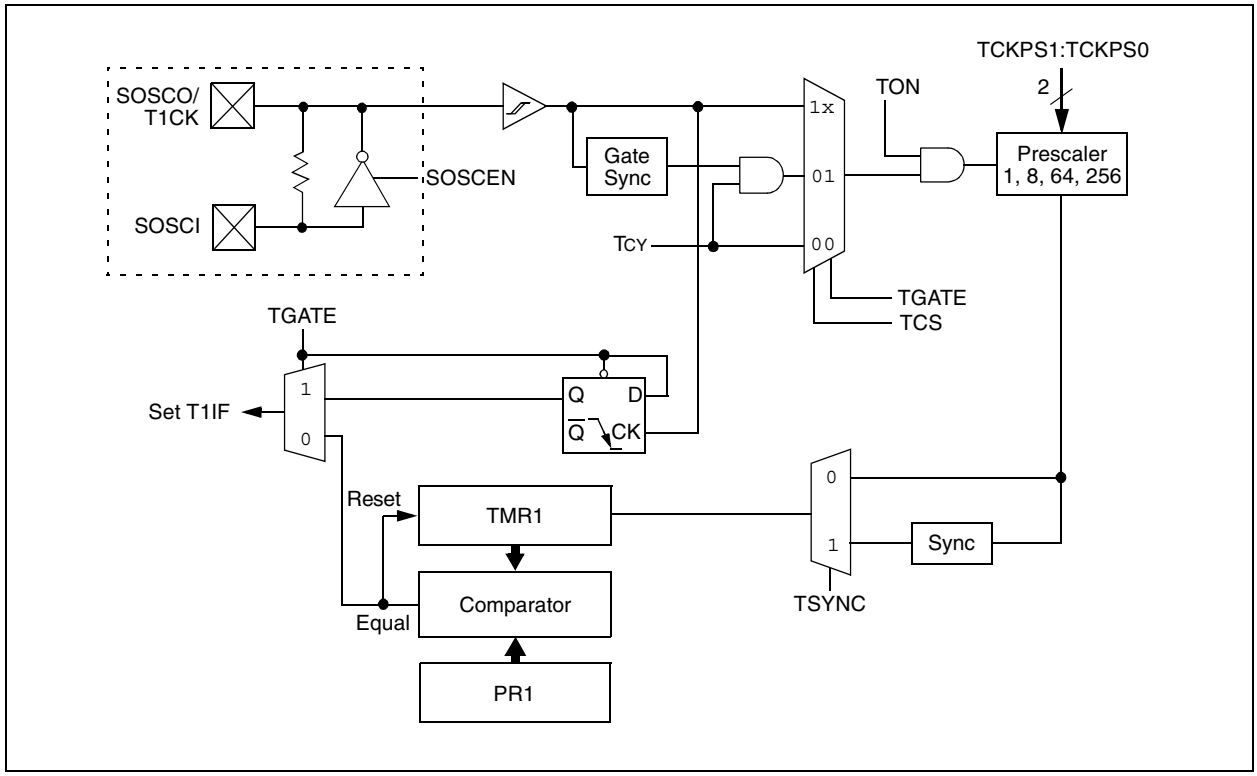
- Timer gate operation
- Selectable prescaler settings
- Timer operation during CPU Idle and Sleep modes
- Interrupt on 16-bit period register match or falling edge of external gate signal

Figure 10-1 presents a block diagram of the 16-bit timer module.

To configure Timer1 for operation:

1. Set the TON bit (= 1).
2. Select the timer prescaler ratio using the TCKPS1:TCKPS0 bits.
3. Set the Clock and Gating modes using the TCS and TGATE bits.
4. Set or clear the TSYNC bit to configure synchronous or asynchronous operation.
5. Load the timer period value into the PR1 register.
6. If interrupts are required, set the interrupt enable bit, T1IE. Use the priority bits, T1IP2:T1IP0, to set the interrupt priority.

FIGURE 10-1: 16-BIT TIMER1 MODULE BLOCK DIAGRAM



REGISTER 10-1: T1CON: TIMER1 CONTROL REGISTER

Upper Byte:							
R/W-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
TON	—	TSIDL	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
U-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	U-0
—	TGATE	TCKPS1	TCKPS0	—	TSYNC	TCS	—
bit 7							bit 0

bit 15 **TON:** Timer1 On bit
1 = Starts 16-bit Timer1
0 = Stops 16-bit Timer1

bit 14 **Unimplemented:** Read as '0'

bit 13 **TSIDL:** Stop in Idle Mode bit
1 = Discontinue module operation when device enters Idle mode
0 = Continue module operation in Idle mode

bit 12-7 **Unimplemented:** Read as '0'

bit 6 **TGATE:** Timer1 Gated Time Accumulation Enable bit
When TCS = 1:
This bit is ignored.
When TCS = 0:
1 = Gated time accumulation enabled
0 = Gated time accumulation disabled

bit 5-4 **TCKPS1:TCKPS0:** Timer1 Input Clock Prescale Select bits
11 = 1:256
10 = 1:64
01 = 1:8
00 = 1:1

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

bit 2 **TSYNC:** Timer1 External Clock Input Synchronization Select bit
When TCS = 1:
1 = Synchronize external clock input
0 = Do not synchronize external clock input
When TCS = 0:
This bit is ignored.

bit 1 **TCS:** Timer1 Clock Source Select bit
1 = External clock from pin T1CK (on the rising edge)
0 = Internal clock (FOSC/2)

bit 0 **Unimplemented:** Read as '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

11.0 TIMER2/3 AND TIMER4/5

Note:	This data sheet summarizes the features of this group of PIC24FJ devices. It is not intended to be a comprehensive reference source.

The Timer2/3 and Timer4/5 modules are 32-bit timers, which can also be configured as four independent 16-bit timers with selectable operating modes.

As a 32-bit timer, Timer2/3 and Timer4/5 operate in three modes:

- Two independent 16-bit timers (Timer2 and Timer3) with all 16-bit operating modes (except Asynchronous Counter mode)
- Single 32-bit Timer
- Single 32-bit Synchronous Counter

They also support these features:

- Timer gate operation
- Selectable prescaler settings
- Timer operation during Idle and Sleep modes
- Interrupt on a 32-bit period register match
- ADC Event Trigger (Timer4/5 only)

Individually, all four of the 16-bit timers can function as synchronous timers or counters. They also offer the features listed above, except for the ADC Event Trigger; this is implemented only with Timer5. The operating modes and enabled features are determined by setting the appropriate bit(s) in the T2CON, T3CON, T4CON and T5CON registers. T2CON and T4CON are shown in generic form in Register 11-1; T3CON and T5CON are shown in Register 11-2.

For 32-bit timer/counter operation, Timer2 and Timer4 are the least significant word; Timer3 and Timer4 are the most significant word of the 32-bit timers.

Note:	For 32-bit operation, T3CON and T5CON control bits are ignored. Only T2CON and T4CON control bits are used for setup and control. Timer2 and Timer4 clock and gate inputs are utilized for the 32-bit timer modules, but an interrupt is generated with the Timer3 or Timer5 interrupt flags.

To configure Timer2/3 or Timer4/5 for 32-bit operation:

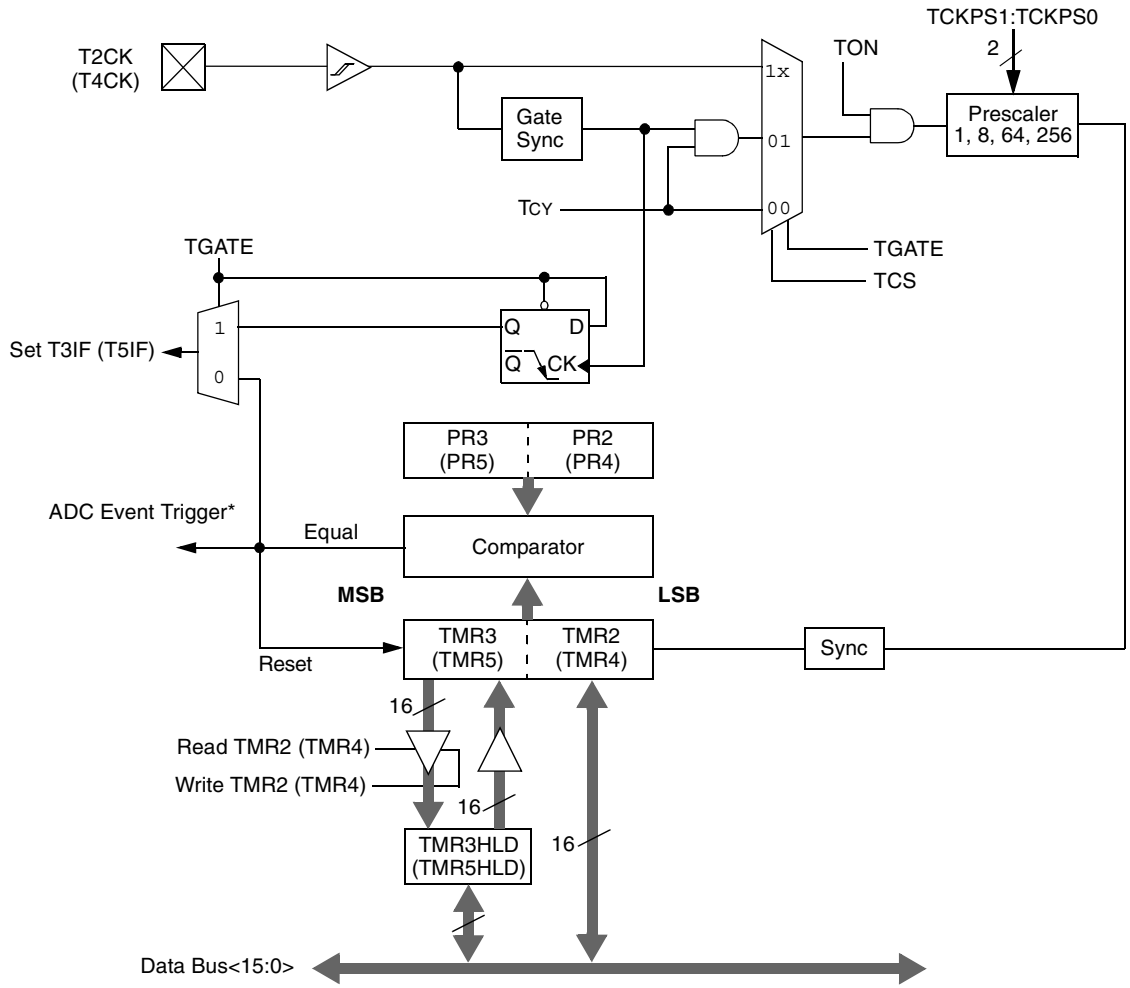
1. Set the T32 bit (T2CON<3> or T4CON<3> = 1).
2. Select the prescaler ratio for Timer2 or Timer4 using the TCKPS1:TCKPS0 bits.
3. Set the Clock and Gating modes using the TCS and TGATE bits.
4. Load the timer period value. PR3 (or PR5) will contain the most significant word of the value, while PR2 (or PR4) contains the least significant word.
5. If interrupts are required, set the interrupt enable bit T3IE or T5IE; use the priority bits, T3IP2:T3IP0 or T5IP2:T5IP0, to set the interrupt priority. Note that while Timer2 or Timer4 controls the timer, the interrupt appears as a Timer3 or Timer5 interrupt.
6. Set the TON bit (= 1).

The timer value at any point is stored in the register pair, TMR3:TMR2 (or TMR5:TMR4). TMR3 (TMR5) always contains the most significant word of the count, while TMR2 (TMR4) contains the least significant word.

To configure any of the timers for individual 16-bit operation:

1. Clear the T32 bit corresponding to that timer (T2CON<3> for Timer2 and Timer3 or T4CON<3> for Timer4 and Timer5).
2. Select the timer prescaler ratio using the TCKPS1:TCKPS0 bits.
3. Set the Clock and Gating modes using the TCS and TGATE bits.
4. Load the timer period value into the PRx register.
5. If interrupts are required, set the interrupt enable bit, TxIE; use the priority bits, TxIP2:TxIP0, to set the interrupt priority.
6. Set the TON bit (TxCON<15> = 1).

FIGURE 11-1: TIMER2/3 AND TIMER4/5 (32-BIT) BLOCK DIAGRAM



Note: The 32-bit Timer Configuration bit, T32, must be set for 32-bit timer/counter operation. All control bits are respective to the T2CON and T4CON registers.

* The ADC Event Trigger is available only on Timer4/5.

FIGURE 11-2: TIMER2 AND TIMER4 (16-BIT SYNCHRONOUS) BLOCK DIAGRAM

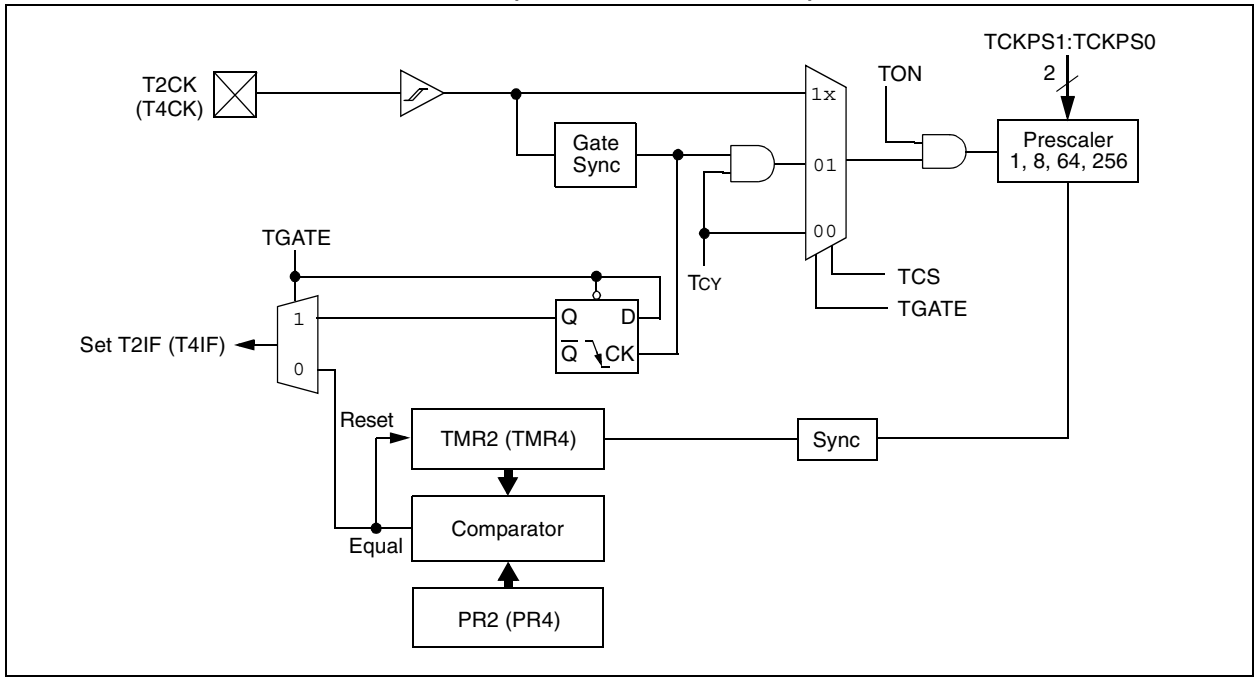
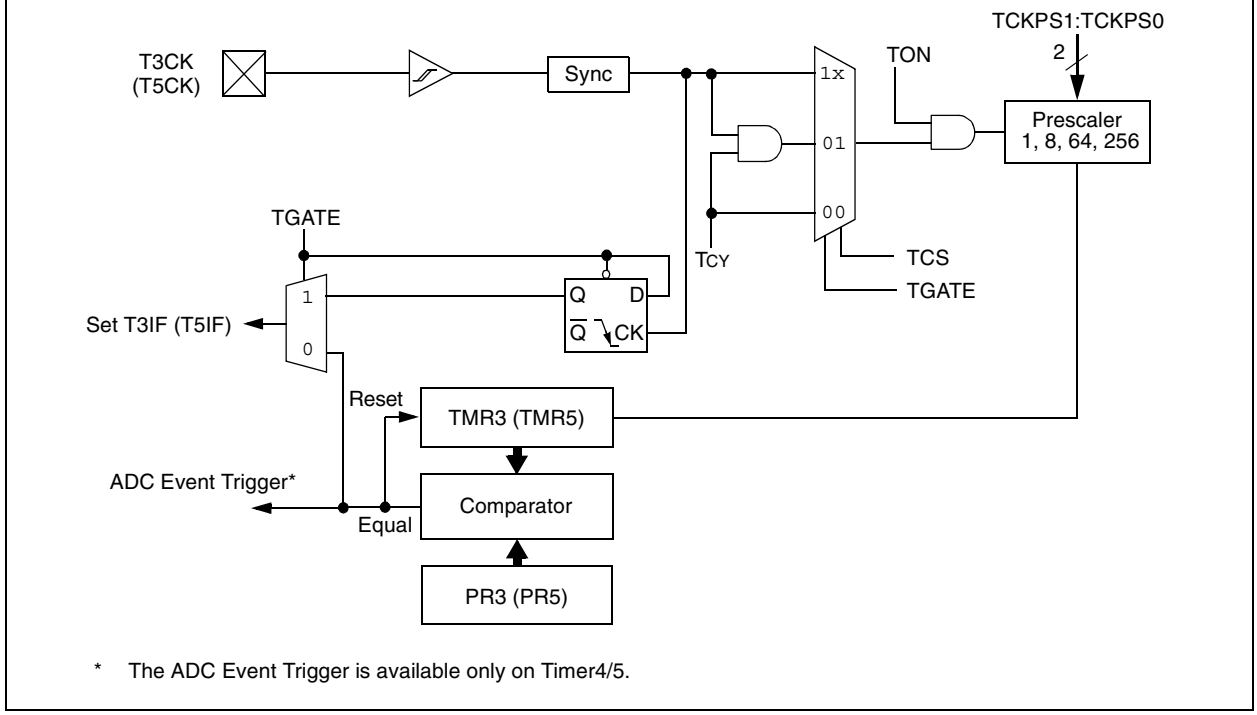


FIGURE 11-3: TIMER3 AND TIMER5 (16-BIT ASYNCHRONOUS) BLOCK DIAGRAM



* The ADC Event Trigger is available only on Timer4/5.

REGISTER 11-1: TxCON: TIMER2 AND TIMER4 CONTROL REGISTER

Upper Byte:

R/W-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
TON	—	TSIDL	—	—	—	—	—
bit 15							bit 8

Lower Byte:

U-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	U-0
—	TGATE	TCKPS1	TCKPS0	T32	—	TCS	—
bit 7							bit 0

bit 15 **TON:** Timerx On bit

When TxCON<3> = 1:

1

REGISTER 11-2: TyCON: TIMER3 AND TIMER5 CONTROL REGISTER

Upper Byte:							
R/W-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
TON(1)	—	TSIDL(1)	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
U-0	R/W-0	R/W-0	R/W-0	U-0	U-0	R/W-0	U-0
—	TGATE(1)	TCKPS1(1)	TCKPS0(1)	—	—	TCS(1)	—
bit 7							bit 0

bit 15 **TON:** Timery On bit(1)
1 = Starts 16-bit Timery
0 = Stops 16-bit Timery

bit 14 **Unimplemented:** Read as '0'

bit 13 **TSIDL:** Stop in Idle Mode bit(1)
1 = Discontinue module operation when device enters Idle mode
0 = Continue module operation in Idle mode

bit 12-7 **Unimplemented:** Read as '0'

bit 6 **TGATE:** Timery Gated Time Accumulation Enable bit(1)
When TCS = 1:
This bit is ignored.
When TCS = 0:
1 = Gated time accumulation enabled
0 = Gated time accumulation disabled

bit 5-4 **TCKPS1:TCKPS0:** Timery Input Clock Prescale Select bits(1)
11 = 1:256
10 = 1:64
01 = 1:8
00 = 1:1

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

bit 1 **TCS:** Timery Clock Source Select bit(1)
1 = External clock from pin TyCK (on the rising edge)
0 = Internal clock (FOSC/2)

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

Note 1: When 32-bit operation is enabled (T2CON<3> = 1), these bits have no effect on Timery operation; all timer functions are set through T2CON.

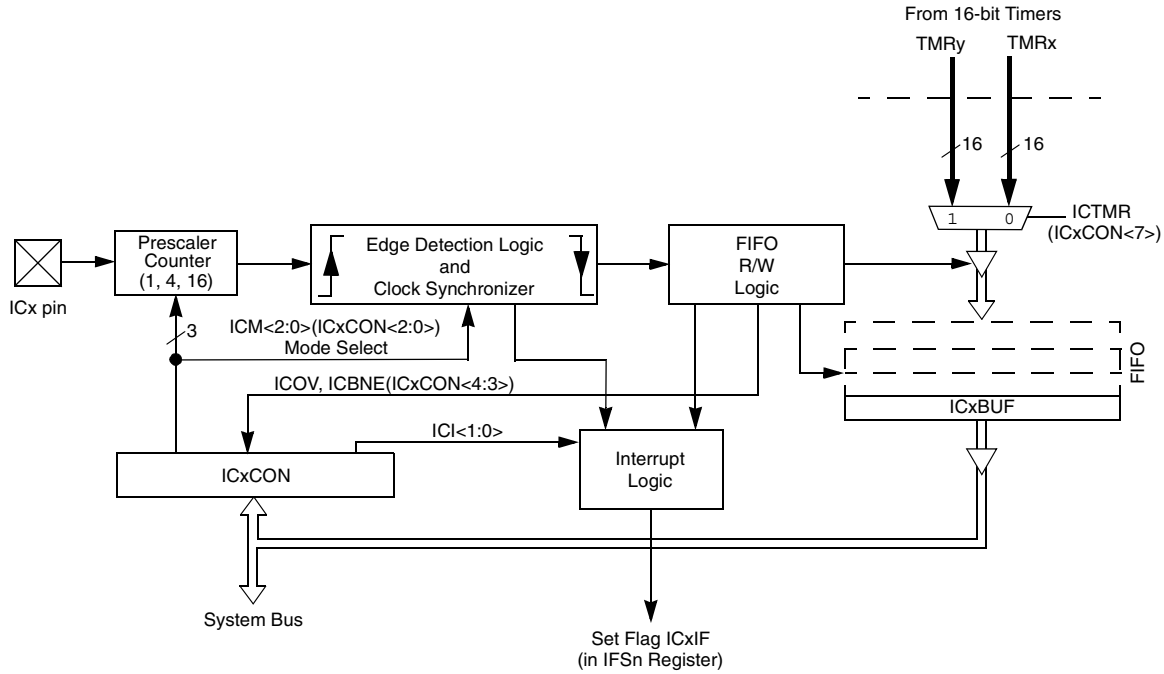
Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

NOTES:

12.0 INPUT CAPTURE

Note:	This data sheet summarizes the features of this group of PIC24FJ devices. It is not intended to be a comprehensive reference source.

FIGURE 12-1: INPUT CAPTURE BLOCK DIAGRAM



Note: An 'x' in a signal, register or bit name denotes the number of the capture channel.

12.1 Input Capture Registers

REGISTER 12-1: ICxCON: INPUT CAPTURE x CONTROL REGISTER

Upper Byte:							
U-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
—	—	ICSIDL	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R-0, HC	R-0, HC	R/W-0	R/W-0	R/W-0
ICTMR	ICI1	ICI0	ICOV	ICBNE	ICM2	ICM1	ICM0
bit 7							bit 0

bit 15-14 **Unimplemented:** Read as '0'

bit 13 **ICSIDL:** Input Capture x Module Stop in Idle Control bit
1 = Input capture module will halt in CPU Idle mode
0 = Input capture module will continue to operate in CPU Idle mode

bit 12-8 **Unimplemented:** Read as '0'

bit 7 **ICTMR:** Input Capture x Timer Select bit
1 = TMR2 contents are captured on capture event
0 = TMR3 contents are captured on capture event

Note: Timer selections may vary. Refer to the device data sheet for details.

bit 6-5 **ICI1:ICI0:** Select Number of Captures per Interrupt bits
11 = Interrupt on every fourth capture event
10 = Interrupt on every third capture event
01 = Interrupt on every second capture event
00 = Interrupt on every capture event

bit 4 **ICOV:** Input Capture x Overflow Status Flag (Read-Only) bit
1 = Input capture overflow occurred
0 = No input capture overflow occurred

bit 3 **ICBNE:** Input Capture x Buffer Empty Status (Read-Only) bit
1 = Input capture buffer is not empty, at least one more capture value can be read
0 = Input capture buffer is empty

bit 2-0 **ICM2:ICM0:** Input Capture x Mode Select bits
111 = Input capture functions as interrupt pin only when device is in Sleep or Idle mode
(rising edge detect only, all other control bits are not applicable)
110 = Unused (module disabled)
101 = Capture mode, every 16th rising edge
100 = Capture mode, every 4th rising edge
011 = Capture mode, every rising edge
010 = Capture mode, every falling edge
001 = Capture mode, every edge (rising and falling) – ICI<1:0> does not control interrupt generation for this mode
000 = Input capture module turned off

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

13.0 OUTPUT COMPARE

13.1 Setup for Single Output Pulse Generation

When the OCM control bits (OCxCON<2:0>) are set to '100', the selected output compare channel initializes the OCx pin to the low state and generates a single output pulse.

To generate a single output pulse, the following steps are required (these steps assume the timer source is initially turned off, but this is not a requirement for the module operation):

1. Determine the instruction clock cycle time. Take into account the frequency of the external clock to the timer source (if one is used) and the timer prescaler settings.
2. Calculate time to the rising edge of the output pulse relative to the TMRy start value (0000h).
3. Calculate the time to the falling edge of the pulse based on the desired pulse width and the time to the rising edge of the pulse.
4. Write the values computed in steps 2 and 3 above into the Compare register, OCxR, and the Secondary Compare register, OCxRS, respectively.
5. Set Timer Period register, PRy, to value equal to or greater than value in OCxRS, the Secondary Compare register.
6. Set the OCM bits to '100' and the OCTSEL (OCxCON<3>) bit to the desired timer source. The OCx pin state will now be driven low.
7. Set the TON (TyCON<15>) bit to '1', which enables the compare time base to count.
8. Upon the first match between TMRy and OCxR, the OCx pin will be driven high.
9. When the incrementing timer, TMRy, matches the Secondary Compare register, OCxRS, the second and trailing edge (high-to-low) of the pulse is driven onto the OCx pin. No additional pulses are driven onto the OCx pin and it remains at low. As a result of the second compare match event, the OCxIF interrupt flag bit is set, which will result in an interrupt if it is enabled, by setting the OCxIE bit. For further information on peripheral interrupts, refer to **Section 6.0 "Interrupt Controller"**.
10. To initiate another single pulse output, change the Timer and Compare register settings, if needed, and then issue a write to set the OCM bits to '100'. Disabling and re-enabling of the timer and clearing the TMRy register are not required, but may be advantageous for defining a pulse from a known event time boundary.

The output compare module does not have to be disabled after the falling edge of the output pulse. Another pulse can be initiated by rewriting the value of the OCxCON register.

13.2 Setup for Continuous Output Pulse Generation

When the OCM control bits (OCxCON<2:0>) are set to '101', the selected output compare channel initializes the OCx pin to the low state and generates output pulses on each and every compare match event.

For the user to configure the module for the generation of a continuous stream of output pulses, the following steps are required (these steps assume the timer source is initially turned off, but this is not a requirement for the module operation):

1. Determine the instruction clock cycle time. Take into account the frequency of the external clock to the timer source (if one is used) and the timer prescaler settings.
2. Calculate time to the rising edge of the output pulse relative to the TMRy start value (0000h).
3. Calculate the time to the falling edge of the pulse, based on the desired pulse width and the time to the rising edge of the pulse.
4. Write the values computed in step 2 and 3 above into the Compare register, OCxR, and the Secondary Compare register, OCxRS, respectively.
5. Set Timer Period register, PRy, to value equal to or greater than value in OCxRS, the Secondary Compare register.
6. Set the OCM bits to '101' and the OCTSEL bit to the desired timer source. The OCx pin state will now be driven low.

13.3 Pulse-Width Modulation Mode

The following steps should be taken when configuring the output compare module for PWM operation:

1. Set the PWM period by writing to the selected Timer Period register (PRy).
2. Set the PWM duty cycle by writing to the OCxRS register.
3. Write the OCxR register with the initial duty cycle.
4. Enable interrupts, if required, for the timer and output compare modules. The output compare interrupt is required for PWM Fault pin utilization.
5. Configure the output compare module for one of two PWM operation modes by writing to the Output Compare mode bits OCM<2:0> (OCxCON<2:0>).
6. Set the TMRy prescale value and enable the time base by setting TON (TxCON<15>) = 1.

Note: The OCxR register should be initialized before the output compare module is first enabled. The OCxR register becomes a Read-Only Duty Cycle register when the module is operated in the PWM modes. The value held in OCxR will become the PWM duty cycle for the first PWM period. The contents of the Duty Cycle Buffer register, OCxRS, will not be transferred into OCxR until a time base period match occurs.

13.3.1 PWM PERIOD

The PWM period is specified by writing to PRy, the Timer Period register. The PWM period can be calculated using Equation 13-1.

EQUATION 13-1: CALCULATING THE PWM PERIOD(1)

PWM Period = [(PRy) + 1] • TCY • (Timer Prescale Value)
where:
PWM Frequency = 1/[PWM Period]

Note 1: Based on TCY = FOSC/2, Doze mode and PLL are disabled.

Note: A PRy value of N will produce a PWM period of N + 1 time base count cycles. For example, a value of 7 written into the PRy register will yield a period consisting of 8 time base cycles.

13.3.2 PWM DUTY CYCLE

The PWM duty cycle is specified by writing to the OCxRS register. The OCxRS register can be written to at any time, but the duty cycle value is not latched into OCxR until a match between PRy and TMRy occurs (i.e., the period is complete). This provides a double buffer for the PWM duty cycle and is essential for glitch-less PWM operation. In the PWM mode, OCxR is a read-only register.

Some important boundary parameters of the PWM duty cycle include:

- If the Duty Cycle register, OCxR, is loaded with 0000h, the OCx pin will remain low (0% duty cycle).
- If OCxR is greater than PRy (Timer Period register), the pin will remain high (100% duty cycle).
- If OCxR is equal to PRy, the OCx pin will be low for one time base count value and high for all other count values.

See Example 13-1 for PWM mode timing details. Table 13-1 shows example PWM frequencies and resolutions for a device operating at 10 MIPS.

EQUATION 13-2: CALCULATION FOR MAXIMUM PWM RESOLUTION(1)

$$\text{Maximum PWM Resolution (bits)} = \frac{\log_{10}\left(\frac{F_{CY}}{F_{PWM} \bullet (\text{Timer Prescale Value})}\right)}{\log_{10}(2)} \text{ bits}$$

Note 1: Based on TCY = FOSC/2, Doze mode and PLL are disabled.

EXAMPLE 13-1: PWM PERIOD AND DUTY CYCLE CALCULATIONS(1)

1. Find the Period register value for a desired PWM frequency of 52.08 kHz, where FOSC = 8 MHz with PLL (32 MHz device clock rate) and a Timer2 prescaler setting of 1:1.

   TCY = 2/FOSC = 62.5 ns

   PWM Period = 1/PWM Frequency = 1/52.08 kHz = 19.2 µs

   PWM Period = (PR2 + 1) • TCY • (Timer 2 Prescale Value)

   19.2 µs = (PR2 + 1) • 62.5 ns • 1

   PR2 = 306

2. Find the maximum resolution of the duty cycle that can be used with a 52.08 kHz frequency and a 32 MHz device clock rate:

   PWM Resolution = $\log_{10}(FCY/FPWM)/\log_{10}2)$ bits

   = $(\log_{10}(16\ MHz/52.08\ kHz)/\log_{10}2)$ bits

   = 8.3 bits

Note 1: Based on TCY = FOSC/2, Doze mode and PLL are disabled.

TABLE 13-1: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 4 MIPS (FCY = 4 MHz)(1)

PWM Frequency	7.6 Hz	61 Hz	122 Hz	977 Hz	3.9 kHz	31.3 kHz	125 kHz
Timer Prescaler Ratio	8	1	1	1	1	1	1
Period Register Value	FFFFh	FFFFh	7FFFh	0FFFh	03FFh	007Fh	001Fh
Resolution (bits)	16	16	15	12	10	7	5

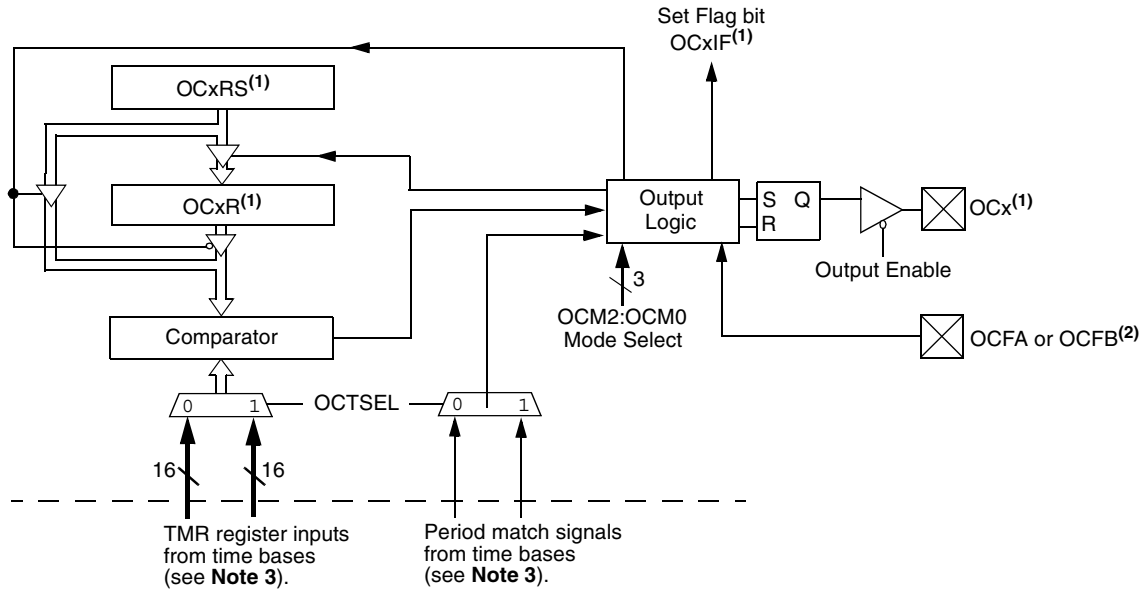
Note 1: Based on TCY = FOSC/2, Doze mode and PLL are disabled.

TABLE 13-2: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 16 MIPS (FCY = 16 MHz)(1)

PWM Frequency	30.5 Hz	244 Hz	488 Hz	3.9 kHz	15.6 kHz	125 kHz	500 kHz
Timer Prescaler Ratio	8	1	1	1	1	1	1
Period Register Value	FFFFh	FFFFh	7FFFh	0FFFh	03FFh	007Fh	001Fh
Resolution (bits)	16	16	15	12	10	7	5

Note 1: Based on TCY = FOSC/2, Doze mode and PLL are disabled.

FIGURE 13-1: OUTPUT COMPARE MODULE BLOCK DIAGRAM



Note 1: Where 'x' is shown, reference is made to the registers associated with the respective output compare channels 1 through 8.

2: OCFA pin controls OC1-OC4 channels. OCFB pin controls OC5-OC8 channels.

3: Each output compare channel can use one of two selectable time bases. Refer to the device data sheet for the time bases associated with the module.

13.4 Output Compare Register

REGISTER 13-1: OCxCON: OUTPUT COMPARE x CONTROL REGISTER

Upper Byte:							
U-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
—	—	OCSIDL	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
U-0	U-0	U-0	R-0 HC	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	OCFLT	OCTSEL	OCM2	OCM1	OCM0
bit 7							bit 0

bit 15-14 **Unimplemented:** Read as '0'

bit 13 **OCSIDL:** Stop Output Compare x in Idle Mode Control bit
1 = Output Compare x will halt in CPU Idle mode
0 = Output Compare x will continue to operate in CPU Idle mode

bit 12-5 **Unimplemented:** Read as '0'

bit 4 **OCFLT:** PWM Fault Condition Status bit
1 = PWM Fault condition has occurred (cleared in HW only)
0 = No PWM Fault condition has occurred (this bit is only used when OCM<2:0> = 111)

bit 3 **OCTSEL:** Output Compare x Timer Select bit
1 = Timer3 is the clock source for Output Compare x
0 = Timer2 is the clock source for Output Compare x

Note: Refer to the device data sheet for specific time bases available to the output compare module.

bit 2-0 **OCM2:OCM0:** Output Compare x Mode Select bits
111 = PWM mode on OCx, Fault pin enabled
110 = PWM mode on OCx, Fault pin disabled
101 = Initialize OCx pin low, generate continuous output pulses on OCx pin
100 = Initialize OCx pin low, generate single output pulse on OCx pin
011 = Compare event toggles OCx pin
010 = Initialize OCx pin high, compare event forces OCx pin low
001 = Initialize OCx pin low, compare event forces OCx pin high
000 = Output compare channel is disabled

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

14.0 SERIAL PERIPHERAL INTERFACE (SPI)

Note:	This data sheet summarizes the features of this group of PIC24FJ devices. It is not intended to be a comprehensive reference source.

The Serial Peripheral Interface (SPI) module is a synchronous 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 SPI module is compatible with Motorola's SPI and SIOP interfaces.

The module supports operation in two buffer modes. In Standard mode, data is shifted through a single serial buffer. In Enhanced Buffer mode, data is shifted through an 8-level FIFO buffer.

Note:	Do not perform read-modify-write operations (such as bit-oriented instructions) on the SPIxBUF register, in either Standard or Enhanced Buffer mode.

The module also supports a basic framed SPI protocol while operating in either Master or Slave modes. A total of four framed SPI configurations are supported.

The SPI serial interface consists of four pins:

- SDIx: Serial Data Input
- SDOx: Serial Data Output
- SCKx: Shift Clock Input or Output
- $\overline{SSx}$: Active-Low Slave Select or Frame Synchronization I/O Pulse

The SPI module can be configured to operate using 2, 3 or 4 pins. In the 3-pin mode, $\overline{SSx}$ is not used. In the 2-pin mode, both SDOx and $\overline{SSx}$ are not used.

A block diagram of the module is shown in Figure 14-1.

Depending on the pin count, devices of the PIC24FJ128GA family offer one or two SPI modules on a single device.

Note:	In this section, the SPI modules are referred to together as SPIx or separately as SPI1 and SPI2. Special Function Registers will follow a similar notation. For example, SPIxCON refers to the control register for the SPI1 or SPI2 module.

To set up the SPI module for the Standard Master mode of operation:

1. If using interrupts:
   a) Clear the SPIxIF bit in the respective IFSn register.
   b) Set the SPIxIE bit in the respective IECn register.
   c) Write the SPIxIP bits in the respective IPCn register to set the interrupt priority.
2. Write the desired settings to the SPIxCON register with MSTEN (SPIxCON1<5>) = 1.
3. Clear the SPIROV bit (SPIxSTAT<6>).
4. Enable SPI operation by setting the SPIEN bit (SPIxSTAT<15>).
5. Write the data to be transmitted to the SPIxBUF register. Transmission (and reception) will start as soon as data is written to the SPIxBUF register.

To set up the SPI module for the Standard Slave mode of operation:

1. Clear the SPIxBUF register.
2. If using interrupts:
   a) Clear the SPIxIF bit in the respective IFSn register.
   b) Set the SPIxIE bit in the respective IECn register.
   c) Write the SPIxIP bits in the respective IPCn register to set the interrupt priority.
3. Write the desired settings to the SPIxCON1 and SPIxCON2 registers with MSTEN (SPIxCON1<5>) = 0.
4. Clear the SMP bit.
5. If the CKE bit is set, then the SSEN bit (SPIxCON1<7>) must be set to enable the $\overline{SSx}$ pin.
6. Clear the SPIROV bit (SPIxSTAT<6>).
7. Enable SPI operation by setting the SPIEN bit (SPIxSTAT<15>).

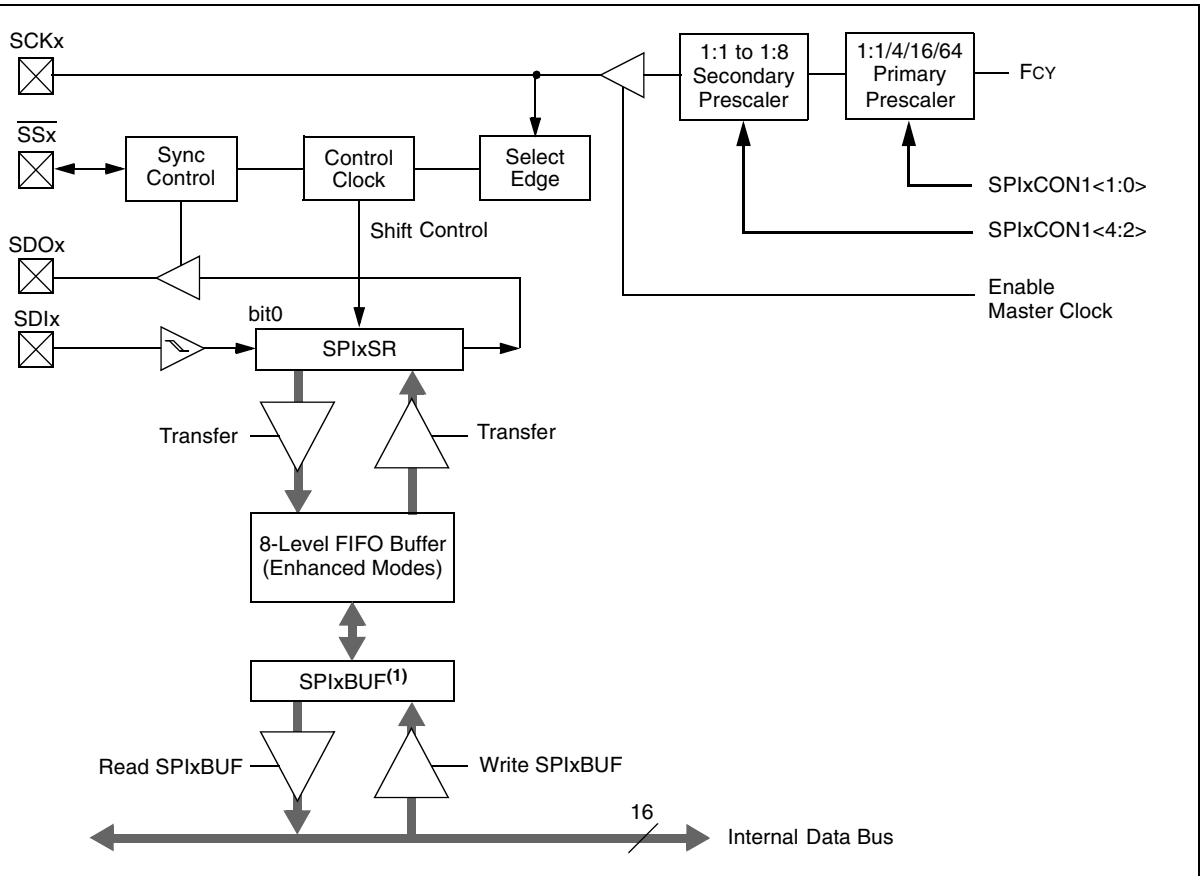
To set up the SPI module for the Enhanced Buffer Master mode of operation:

1. If using interrupts:
   a) Clear the SPIxIF bit in the respective IFSn register.
   b) Set the SPIxIE bit in the respective IECn register.
   c) Write the SPIxIP bits in the respective IPCn register.
2. Write the desired settings to the SPIxCON1 and SPIxCON2 registers with MSTEN (SPIxCON1<5>) = 1.
3. Clear the SPIROV bit (SPIxSTAT<6>).
4. Select Enhanced Buffer mode by setting the SPIBEN bit (SPIxCON2<0>).
5. Enable SPI operation by setting the SPIEN bit (SPIxSTAT<15>).
6. Write the data to be transmitted to the SPIxBUF register. Transmission (and reception) will start as soon as data is written to the SPIxBUF register.

To set up the SPI module for the Enhanced Buffer Slave mode of operation:

1. Clear the SPIxBUF register.
2. If using interrupts:
   - Clear the SPIxIF bit in the respective IFSn register.
   - Set the SPIxIE bit in the respective IECn register.
   - Write the SPIxIP bits in the respective IPCn register to set the interrupt priority.
3. Write the desired settings to the SPIxCON1 and SPIxCON2 registers with MSTEN (SPIxCON1<5>) = 0.
4. Clear the SMP bit.
5. If the CKE bit is set, then the SSEN bit must be set, thus enabling the $\overline{SSx}$ pin.
6. Clear the SPIROV bit (SPIxSTAT<6>).
7. Select Enhanced Buffer mode by setting the SPIBEN bit (SPIxCON2<0>).
8. Enable SPI operation by setting the SPIEN bit (SPIxSTAT<15>).

FIGURE 14-1: SPI MODULE BLOCK DIAGRAM



Note 1: In Standard modes, data is transferred directly between SPIxSR and SPIxBUF.

REGISTER 14-1: SPIxSTAT: SPIx STATUS AND CONTROL REGISTER

Upper Byte:							
R/W-0	U-0	R/W-0	U-0	U-0	R-0	R-0	R-0
SPIEN	—	SPISIDL	—	—	SPIBEC2	SPIBEC1	SPIBEC0
bit 15							bit 8

Lower Byte:							
U-0	R/C-0	U-0	U-0	U-0	U-0	R-0	R-0
—	SPIROV	—	—	—	—	SPITBF	SPIRBF
bit 7							bit 0

bit 15 **SPIEN:** SPIx Enable bit
1 = Enables module and configures SCKx, SDOx, SDIx and SSx as serial port pins
0 = Disables module

bit 14 **Unimplemented:** Read as '0'

bit 13 **SPISIDL:** Stop in Idle Mode bit
1 = Discontinue module operation when device enters Idle mode
0 = Continue module operation in Idle mode

bit 12-11 **Unimplemented:** Read as '0'

bit 10-8 **SPIBEC2:SPIBEC0:** SPIx Buffer Element Count bits
Master mode:
Number of SPI transfers pending.
Slave mode:
Number of SPI transfers unread.

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

bit 6 **SPIROV:** Receive Overflow Flag bit
1 = A new byte/word is completely received and discarded. The user software has not read the previous data in the SPIxBUF register.
0 = No overflow has occurred

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

bit 1 **SPITBF:** SPIx Transmit Buffer Full Status bit
1 = Transmit not yet started, SPIxTXB is full
0 = Transmit started, SPIxTXB is empty
In Standard Buffer mode:
Automatically set in hardware when CPU writes SPIxBUF location, loading SPIxTXB.
Automatically cleared in hardware when SPIx module transfers data from SPIxTXB to SPIxSR.
In Enhanced Buffer mode:
Automatically set in hardware when CPU writes SPIxBUF location, loading the last available buffer location.
Automatically cleared in hardware when a buffer location is available for a CPU write.

bit 0 **SPIRBF:** SPIx Receive Buffer Full Status bit
1 = Receive complete, SPIxRXB is full
0 = Receive is not complete, SPIxRXB is empty
In Standard Buffer mode:
Automatically set in hardware when SPIx transfers data from SPIxSR to SPIxRXB.
Automatically cleared in hardware when core reads SPIxBUF location, reading SPIxRXB.
In Enhanced Buffer mode:
Automatically set in hardware when SPIx transfers data from SPIxSR to buffer, filling the last unread buffer location.
Automatically cleared in hardware when a buffer location is available for a transfer from SPIxSR.

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

REGISTER 14-2: SPIxCON1: SPIx CONTROL REGISTER 1

Upper Byte:							
U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	DISSCK	DISSDO	MODE16	SMP	CKE
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SSEN	CKP	MSTEN	SPRE2	SPRE1	SPRE0	PPRE1	PPRE0
bit 7							bit 0

bit 15-13 **Unimplemented:** Read as '0'

bit 12 **DISSCK:** Disable SCKx pin bit (SPI Master modes only)
1 = Internal SPI clock is disabled, pin functions as I/O
0 = Internal SPI clock is enabled

bit 11 **DISSDO:** Disable SDOx pin bit
1 = SDOx pin is not used by module; pin functions as I/O
0 = SDOx pin is controlled by the module

bit 10 **MODE16:** Word/Byte Communication Select bit
1 = Communication is word-wide (16 bits)
0 = Communication is byte-wide (8 bits)

bit 9 **SMP:** SPIx Data Input Sample Phase bit
Master mode:
1 = Input data sampled at end of data output time
0 = Input data sampled at middle of data output time
Slave mode:
SMP must be cleared when SPIx is used in Slave mode.

bit 8 **CKE:** SPIx Clock Edge Select bit
1 = Serial output data changes on transition from active clock state to Idle clock state (see bit 6)
0 = Serial output data changes on transition from Idle clock state to active clock state (see bit 6)

Note: The CKE bit is not used in the Framed SPI modes. The user should program this bit to '0' for the Framed SPI modes (FRMEN = 1).

bit 7 **SSEN:** Slave Select Enable (Slave mode) bit
1 = $\overline{SSx}$ pin used for Slave mode
0 = $\overline{SSx}$ pin not used by module. Pin controlled by port function.

bit 6 **CKP:** Clock Polarity Select bit
1 = Idle state for clock is a high level; active state is a low level
0 = Idle state for clock is a low level; active state is a high level

bit 5 **MSTEN:** Master Mode Enable bit
1 = Master mode
0 = Slave mode

bit 4-2 **SPRE2:SPRE0:** Secondary Prescale (Master mode) bits
111 = Secondary prescale 1:1
110 = Secondary prescale 2:1
...
000 = Secondary prescale 8:1

bit 1-0 **PPRE1:PPRE0:** Primary Prescale (Master mode) bits
11 = Primary prescale 1:1
10 = Primary prescale 4:1
01 = Primary prescale 16:1
00 = Primary prescale 64:1

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

REGISTER 14-3: SPIxCON2: SPIx CONTROL REGISTER 2

Upper Byte:							
R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0	U-0
FRMEN	SPIFSD	SPIFPOL	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	—	—	—	SPIFE	SPIBEN
bit 7							bit 0

bit 15 **FRMEN:** Framed SPIx Support bit
1 = Framed SPIx support enabled
0 = Framed SPIx support disabled

bit 14 **SPIFSD:** Frame Sync Pulse Direction Control on $\overline{SSx}$ pin bit
1 = Frame sync pulse input (slave)
0 = Frame sync pulse output (master)

bit 13 **SPIFPOL:** Frame Sync Pulse Polarity bit (Frame mode only)
1 = Frame sync pulse is active-high
0 = Frame sync pulse is active-low

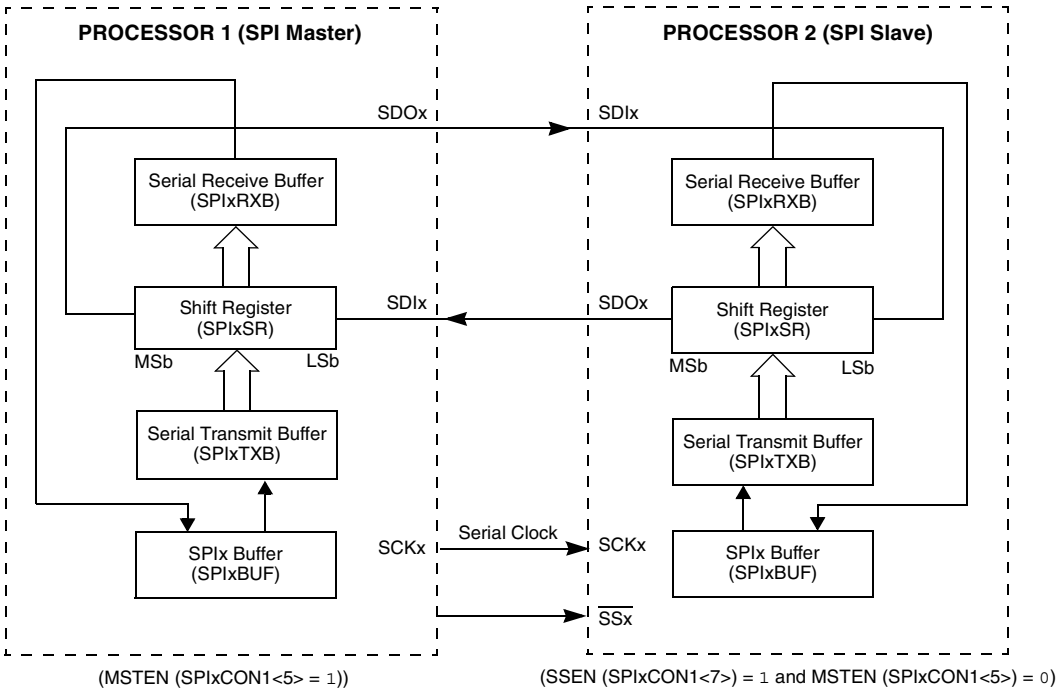
bit 12-2 **Unimplemented:** Read as '0'

bit 1 **SPIFE:** Frame Sync Pulse Edge Select bit
1 = Frame sync pulse coincides with first bit clock
0 = Frame sync pulse precedes first bit clock

bit 0 **SPIBEN:** Enhanced Buffer Enable bit
1 = Enhanced Buffer enabled
0 = Enhanced Buffer disabled (legacy mode)

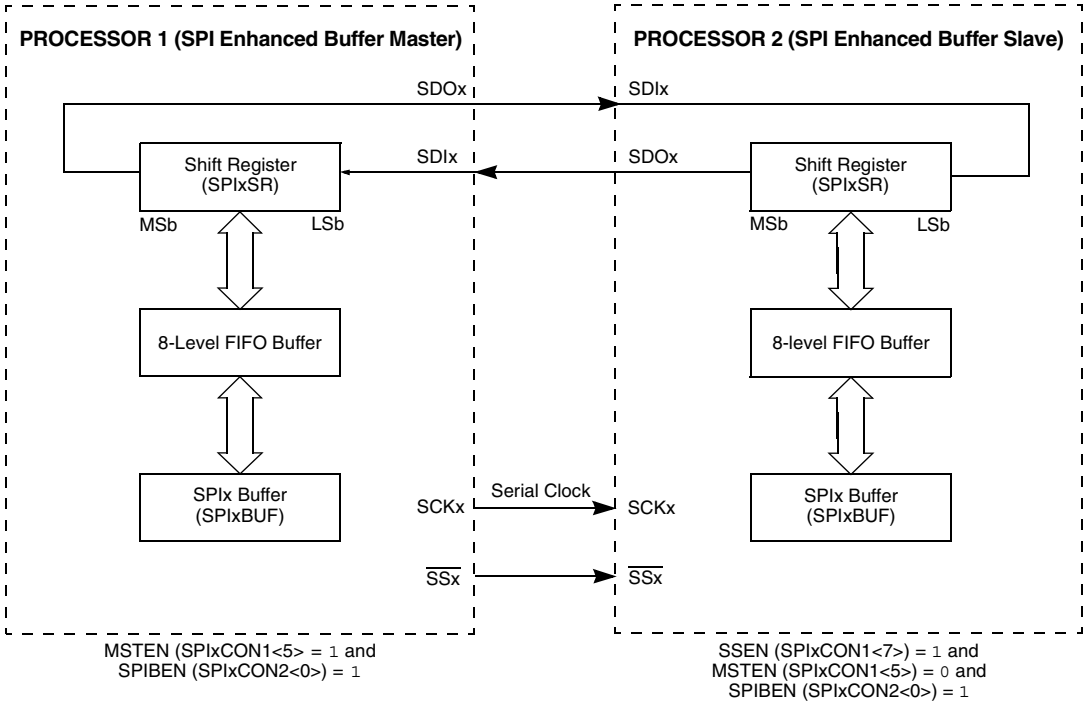
Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

FIGURE 14-2: SPI MASTER/SLAVE CONNECTION (STANDARD MODE)



Note 1: Using the $\overline{SSx}$ pin in Slave mode of operation is optional.
2: User must write transmit data to read received data from SPIxBUF. The SPIxTXB and SPIxRXB registers are memory mapped to SPIxBUF.

FIGURE 14-3: SPI MASTER/SLAVE CONNECTION (ENHANCED BUFFER MODES)



Note 1: Using the $\overline{SSx}$ pin in Slave mode of operation is optional.
2: User must write transmit data to read received data from SPIxBUF. The SPIxTXB and SPIxRXB registers are memory mapped to SPIxBUF.

FIGURE 14-4: SPI MASTER, FRAME MASTER CONNECTION DIAGRAM

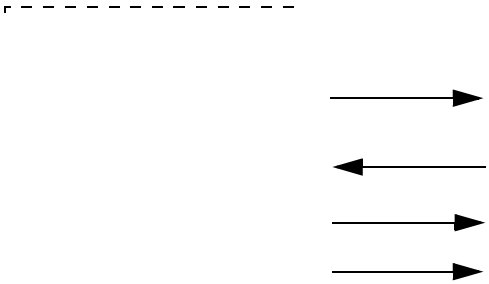


FIGURE 14-5: SPI MASTER, FRAME SLAVE CONNECTION DIAGRAM

FIGURE 14-6: SPI SLAVE, FRAME MASTER CONNECTION DIAGRAM

FIGURE 14-7: SPI SLAVE, FRAME SLAVE CONNECTION DIAGRAM

EQUATION 14-1: RELATIONSHIP BETWEEN DEVICE AND SPI CLOCK SPEED(1)

$$FSCK = \frac{FCY}{\text{Primary Prescaler} * \text{Secondary Prescaler}}$$

Note 1: Based on TCY = FOSC/2, Doze mode and PLL are disabled.

TABLE 14-1: SAMPLE SCK FREQUENCIES(1,2)

FCY = 16 MHz		Secondary Prescaler Settings				
		1:1	2:1	4:1	6:1	8:1
Primary Prescaler Settings	1:1	16000	8000	4000	2667	2000
	4:1	4000	2000	1000	667	500
	16:1	1000	500	250	167	125
	64:1	250	125	63	42	31
FCY = 5 MHz						
Primary Prescaler Settings	1:1	5000	2500	1250	833	625
	4:1	1250	625	313	208	156
	16:1	313	156	78	52	39
	64:1	78	39	20	13	10

Note 1: Based on TCY = FOSC/2, Doze mode and PLL are disabled.

2: SCKx frequencies shown in kHz.

15.0 INTER-INTEGRATED CIRCUIT (I^2C™)

Note: This data sheet summarizes the features of this group of PIC24FJ devices. It is not intended to be a comprehensive reference source.

The Inter-Integrated Circuit (I^2C) module is a serial interface useful for communicating with other peripheral or microcontroller devices. These peripheral devices may be serial EEPROMs, display drivers, A/D converters, etc.

The I^2C module supports these features:

- Independent master and slave logic
- 7-bit and 10-bit device addresses
- General call address, as defined in the I^2C protocol
- Clock stretching to provide delays for the processor to respond to a slave data request
- Both 100 kHz and 400 kHz bus specifications.
- Configurable address masking
- Multi-Master modes to prevent loss of messages in arbitration
- Bus Repeater mode, allowing the acceptance of all messages as a slave regardless of the address
- Automatic SCL

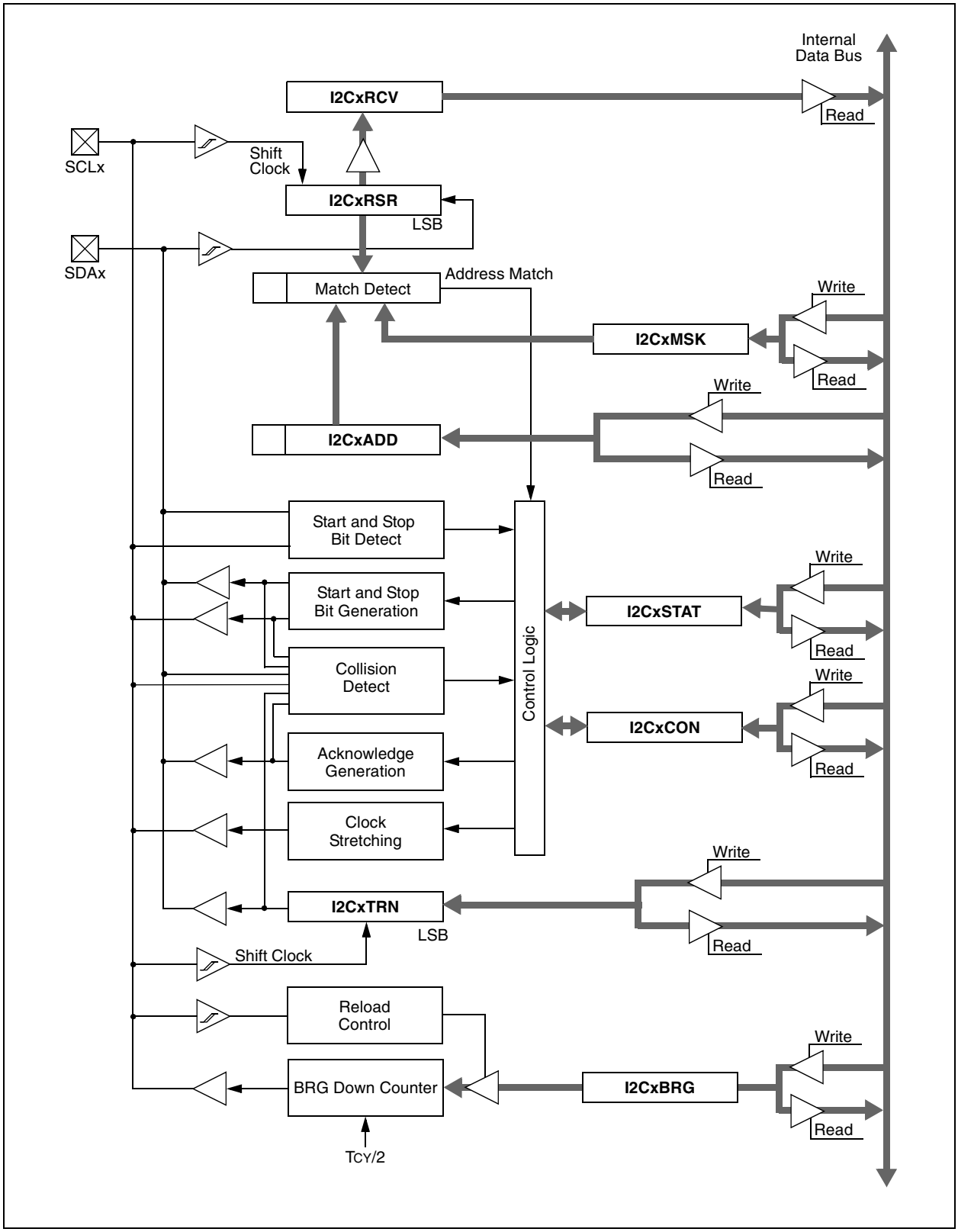
A block diagram of the module is shown in Figure 15-1.

15.1 Communicating as a Master in a Single Master Environment

The details of sending a message in Master mode depends on the communications protocol for the device being communicated with. Typically, the sequence of events is as follows:

1. Assert a Start condition on SDAx and SCLx.
2. Send the I^2C device address byte to the slave with a write indication.
3. Wait for and verify an Acknowledge from the slave.
4. Send the first data byte (sometimes known as the command) to the slave.
5. Wait for and verify an Acknowledge from the slave.
6. Send the serial memory address low byte to the slave.
7. Repeat steps 4 and 5 until all data bytes are sent.
8. Assert a Repeated Start condition on SDAx and SCLx.
9. Send the device address byte to the slave with a read indication.
10. Wait for and verify an Acknowledge from the slave.
11. Enable master reception to receive serial memory data.
12. Generate an ACK or NACK condition at the end of a received byte of data.
13. Generate a Stop condition on SDAx and SCLx.

FIGURE 15-1: I²C™ BLOCK DIAGRAM



15.2 Setting Baud Rate When Operating as a Bus Master

To compute the Baud Rate Generator reload value, use the following equation:

EQUATION 15-1:(1)

$$FSCL = \frac{FCY}{2 \cdot (I2CxBRG + 1)}$$

or

$$I2CxBRG = \left(\frac{FCY}{2 \cdot FSCL}\right) - 1$$

Note 1: Based on TCY = FOSC/2, Doze mode and PLL are disabled.

15.3 Slave Address Masking

The I2CxMSK register (Register 15-3) designates address bit positions as "don't care" for both 7-bit and 10-bit Address modes. Setting a particular bit location (= 1) in the I2CxMSK register causes the slave module to respond whether the corresponding address bit value is a '0' or '1'. For example, when I2CxMSK is set to '00100000', the slave module will detect both addresses '0000000' and '00100000'.

To enable address masking, the IPMI (Intelligent Peripheral Management Interface) must be disabled by clearing the IPMIEN bit (I2CxCON<11>).

TABLE 15-1: I²C™ CLOCK RATES(1)

Required System FSCL	FCY	I2CxBRG Value		Actual FSCL
		(Decimal)	(Hexadecimal)	
100 kHz	16 MHz	79	4F	100 kHz
100 kHz	8 MHz	39	27	100 kHz
100 kHz	4 MHz	19	13	100 kHz
400 kHz	16 MHz	19	13	400 kHz
400 kHz	8 MHz	9	9	400 kHz
400 kHz	4 MHz	4	4	400 kHz
400 kHz	2 MHz	2	2	333 kHz(2)

REGISTER 15-1: I2CxCON: I2Cx CONTROL REGISTER

Upper Byte:							
R/W-0	U-0	R/W-0	R/W-1 HC	R/W-0	R/W-0	R/W-0	R/W-0
I2CEN	—	I2CSIDL	SCLREL	IPMIEN	A10M	DISSLW	SMEN
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R/W-0 HC	R/W-0 HC	R/W-0 HC	R/W-0 HC	R/W-0 HC
GCEN	STREN	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN
bit 7							bit 0

bit 15 **I2CEN:** I2Cx Enable bit
1 = Enables the I2Cx module and configures the SDAx and SCLx pins as serial port pins
0 = Disables I2Cx module. All I^2C pins are controlled by port functions.

bit 14 **Unimplemented:** Read as '0'

bit 13 **I2CSIDL:** Stop in Idle Mode bit
1 = Discontinue module operation when device enters an Idle mode
0 = Continue module operation in Idle mode

bit 12 **SCLREL:** SCLx Release Control bit (when operating as I^2C Slave)
1 = Release SCLx clock
0 = Hold SCLx clock low (clock stretch)
If STREN = 1:
Bit is R/W (i.e., software may write '0' to initiate stretch and write '1' to release clock).
Hardware clear at beginning of slave transmission.
Hardware clear at end of slave reception.
If STREN = 0:
Bit is R/S (i.e., software may only write '1' to release clock).
Hardware clear at beginning of slave transmission.

bit 11 **IPMIEN:** Intelligent Peripheral Management Interface (IPMI) Enable bit
1 = IPMI Support mode is enabled; all addresses Acknowledged
0 = IPMI mode disabled

bit 10 **A10M:** 10-bit Slave Address bit
1 = I2CxADD is a 10-bit slave address
0 = I2CxADD is a 7-bit slave address

bit 9 **DISSLW:** Disable Slew Rate Control bit
1 = Slew rate control disabled
0 = Slew rate control enabled

bit 8 **SMEN:** SMBus Input Levels bit
1 = Enable I/O pin thresholds compliant with SMBus specification
0 = Disable SMBus input thresholds

bit 7 **GCEN:** General Call Enable bit (when operating as I^2C slave)
1 = Enable interrupt when a general call address is received in the I2CRSR (module is enabled for reception)
0 = General call address disabled

.

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

REGISTER 15-1: I2CxCON: I2Cx CONTROL REGISTER (CONTINUED)

Upper Byte:							
R/W-0	U-0	R/W-0	R/W-1 HC	R/W-0	R/W-0	R/W-0	R/W-0
I2CEN	—	I2CSIDL	SCLREL	IPMIEN	A10M	DISSLW	SMEN
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R/W-0 HC	R/W-0 HC	R/W-0 HC	R/W-0 HC	R/W-0 HC
GCEN	STREN	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN
bit 7							bit 0

bit 6 **STREN:** SCLx Clock Stretch Enable bit (when operating as I^2C slave)
Used in conjunction with SCLREL bit.
1 = Enable software or receive clock stretching
0 = Disable software or receive clock stretching

bit 5 **ACKDT:** Acknowledge Data bit (When operating as I^2C master. Applicable during master receive.)
Value that will be transmitted when the software initiates an Acknowledge sequence.
1 = Send NACK during Acknowledge
0 = Send ACK during Acknowledge

bit 4 **ACKEN:** Acknowledge Sequence Enable bit
(When operating as I^2C master. Applicable during master receive.)
1 = Initiate Acknowledge sequence on SDAx and SCLx pins and transmit ACKDT data bit
Hardware clear at end of master Acknowledge sequence.
0 = Acknowledge sequence not in progress

bit 3 **RCEN:** Receive Enable bit (when operating as I^2C master)
1 = Enables Receive mode for I^2C
Hardware clear at end of eighth bit of master receive data byte.
0 = Receive sequence not in progress

bit 2 **PEN:** Stop Condition Enable bit (when operating as I^2C master)
1 = Initiate Stop condition on SDAx and SCLx pins
Hardware clear at end of master Stop sequence.
0 = Stop condition not in progress

bit 1 **RSEN:** Repeated Start Condition Enabled bit (when operating as I^2C master)
1 = Initiate Repeated Start condition on SDAx and SCLx pins
Hardware clear at end of master Repeated Start sequence.
0 = Repeated Start condition not in progress

bit 0 **SEN:** Start Condition Enabled bit (when operating as I^2C master)
1 = Initiate Start condition on SDA and SCL pins
Hardware clear at end of master Start sequence.
0 = Start condition not in progress

.

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

REGISTER 15-2: I2CxSTAT: I2Cx STATUS REGISTER

Upper Byte:

R-0 HSC	R-0 HSC	U-0	U-0	U-0	R/C-0 HS	R-0 HSC	R-0 HSC
ACKSTAT	TRSTAT	—	—	—	BCL	GCSTAT	ADD10
bit 15							bit 8

Lower Byte:

R/C-0 HS	R/C-0 HS	R-0 HSC	R/C-0 HSC	R/C-0 HSC	R-0 HSC	R-0 HSC	R-0 HSC
IWCOL	I2COV	D/$\overline{A}$	P	S	R/$\overline{W}$	RBF	TBF
bit 7							bit 0

bit 15 **ACKSTAT:** Acknowledge Status bit
(When operating as I^2C master. Applicable to master transmit operation.)
1 = NACK received from slave
0 = ACK received from slave
Hardware set or clear at end of slave Acknowledge.

bit 14 **TRSTAT:** Transmit Status bit
(When operating as I^2C master. Applicable to master transmit operation.)
1 = Master transmit is in progress (8 bits + ACK)
0 = Master transmit is not in progress
Hardware set at beginning of master transmission.
Hardware clear at end of slave Acknowledge.

bit 13-11 **Unimplemented:** Read as '0'

bit 10 **BCL:** Master Bus Collision Detect bit
1 = A bus collision has been detected during a master operation
0 = No collision
Hardware set at detection of bus collision.

bit 9 **GCSTAT:** General Call Status bit
1 = General call address was received
0 = General call address was not received
Hardware set when address matches general call address.
Hardware clear at Stop detection.

bit 8 **ADD10:** 10-bit Address Status bit
1 = 10-bit address was matched
0 = 10-bit address was not matched
Hardware set at match of 2nd byte of matched 10-bit address.
Hardware clear at Stop detection.

bit 7 **IWCOL:** Write Collision Detect bit
1 = An attempt to write the I2CxTRN register failed because the I^2C module is busy
0 = No collision
Hardware set at occurrence of write to I2CxTRN while busy (cleared by software).

bit 6 **I2COV:** Receive Overflow Flag bit
1 = A byte was received while the I2CxRCV register is still holding the previous byte
0 = No overflow
Hardware set at attempt to transfer I2CRSR to I2CxRCV (cleared by software).

.

Legend:		U = Unimplemented bit, read as '0'	
R = Readable bit	C = Clearable bit	HS = Set in Hardware	HSC = Hardware Set/Cleared
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 15-2: I2CxSTAT: I2Cx STATUS REGISTER (CONTINUED)

Upper Byte:

R-0 HSC	R-0 HSC	U-0	U-0	U-0	R/C-0 HS	R-0 HSC	R-0 HSC
ACKSTAT	TRSTAT	—	—	—	BCL	GCSTAT	ADD10
bit 15							bit 8

Lower Byte:

R/C-0 HS	R/C-0 HS	R-0 HSC	R/C-0 HSC	R/C-0 HSC	R-0 HSC	R-0 HSC	R-0 HSC
IWCOL	I2COV	D/$\overline{A}$	P	S	R/$\overline{W}$	RBF	TBF
bit 7							bit 0

bit 5 **D/$\overline{A}$:** Data/$\overline{\text{Address}}$ bit (when operating as I^2C slave)
1 = Indicates that the last byte received was data
0 = Indicates that the last byte received was device address
Hardware clear at device address match.
Hardware set by write to I2CxTRN or by reception of slave byte.

bit 4 **P:** Stop bit
1 = Indicates that a Stop bit has been detected last
0 = Stop bit was not detected last
Hardware set or clear when Start, Repeated Start or Stop detected.

bit 3 **S:** Start bit
1 = Indicates that a Start (or Repeated Start) bit has been detected last
0 = Start bit was not detected last
Hardware set or clear when Start, Repeated Start or Stop detected.

bit 2 **R/$\overline{W}$:** Read/$\overline{\text{Write}}$ bit Information (when operating as I^2C slave)
1 = Read – indicates data transfer is output from slave
0 = Write – indicates data transfer is input to slave
Hardware set or clear after reception of I^2C device address byte.

bit 1 **RBF:** Receive Buffer Full Status bit
1 = Receive complete, I2CxRCV is full
0 = Receive not complete, I2CxRCV is empty
Hardware set when I2CxRCV written with received byte.
Hardware clear when software reads I2CxRCV.

bit 0 **TBF:** Transmit Buffer Full Status bit
1 = Transmit in progress, I2CxTRN is full
0 = Transmit complete, I2CxTRN is empty
Hardware set when software writes I2CxTRN.
Hardware clear at completion of data transmission.

.

Legend:		U = Unimplemented bit, read as ‘0’	
R = Readable bit	C = Clearable bit	HS = Set in Hardware	HSC = Hardware Set/Cleared
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

REGISTER 15-3: I2CxMSK: I2Cx SLAVE MODE ADDRESS MASK REGISTER

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	—	—	—	AMSK9	AMSK8
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
AMSK7	AMSK6	AMSK5	AMSK4	AMSK3	AMSK2	AMSK1	AMSK0
bit 7							bit 0

bit 15-10 **Unimplemented:** Read as '0'

bit 9-0 **AMSKx:** Mask for Address Bit x Select bit
1 = Enable masking for bit x of incoming message address; bit match not required in this position
0 = Disable masking for bit x; bit match required in this position

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

16.0 UNIVERSAL ASYNCHRONOUS RECEIVER TRANSMITTER (UART)

Note: This data sheet summarizes the features of this group of PIC24FJ devices. It is not intended to be a comprehensive reference source.

The Universal Asynchronous Receiver Transmitter (UART) module is one of the serial I/O modules available in the PIC24 device family. The UART is a full-duplex asynchronous system that can communicate with peripheral devices, such as personal computers, LIN, RS-232 and RS-485 interfaces. The module also supports a hardware flow control option with the $\overline{\text{UxCTS}}$ and $\overline{\text{UxRTS}}$ pins and also includes an IrDA encoder and decoder.

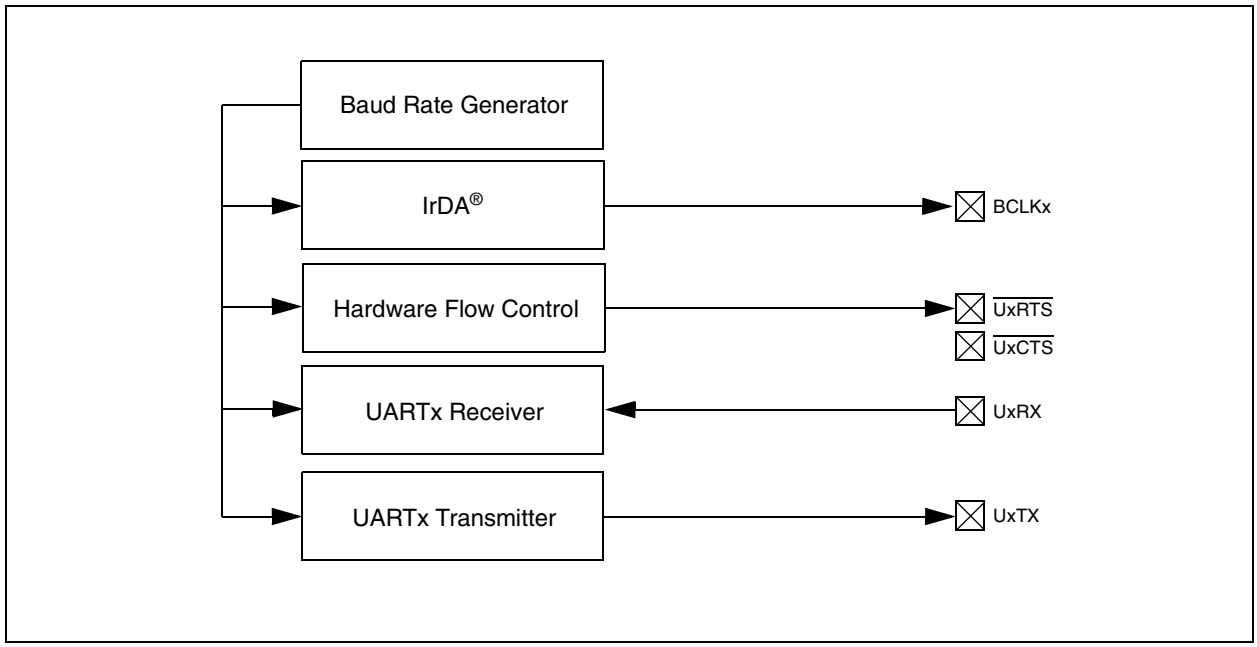
The primary features of the UART module are:

- Full-Duplex 8 or 9-bit Data Transmission through the UxTX and UxRX pins
- Even, Odd or No Parity Options (for 8-bit data)
- One or Two Stop bits
- Hardware Flow Control Option with $\overline{\text{UxCTS}}$ and $\overline{\text{UxRTS}}$ pins
- Fully Integrated Baud Rate Generator with 16-bit Prescaler
- Baud Rates Ranging from 1 Mbps to 15 bps at 16 MIPS
- 4-Deep First-In-First-Out (FIFO) Transmit Data Buffer
- 4-Deep FIFO Receive Data Buffer
- Parity, Framing and Buffer Overrun Error Detection
- Support for 9-bit mode with Address Detect (9th bit = 1)
- Transmit and Receive Interrupts
- Loopback mode for Diagnostic Support
- Support for Sync and Break Characters
- Supports Automatic Baud Rate Detection
- IrDA Encoder and Decoder Logic
- 16x Baud Clock Output for IrDA Support

A simplified block diagram of the UART is shown in Figure 16-1. The UART module consists of these key important hardware elements:

- Baud Rate Generator
- Asynchronous Transmitter
- Asynchronous Receiver

FIGURE 16-1: UART SIMPLIFIED BLOCK DIAGRAM



16.1 UART Baud Rate Generator (BRG)

The UART module includes a dedicated 16-bit Baud Rate Generator. The BRGx register controls the period of a free-running 16-bit timer. Equation 16-1 shows the formula for computation of the baud rate with BRGH = 0.

EQUATION 16-1: UART BAUD RATE WITH BRGH = 0(1,2)

$$\text{Baud Rate} = \frac{FCY}{16 \cdot (BRGx + 1)}$$

$$BRGx = \frac{FCY}{16 \cdot \text{Baud Rate}} - 1$$

Note 1: FCY denotes the instruction cycle clock frequency (FOSC/2).

2: Based on TCY = FOSC/2, Doze mode and PLL are disabled.

Example 16-1 shows the calculation of the baud rate error for the following conditions:

- FCY = 4 MHz
- Desired Baud Rate = 9600

The maximum baud rate (BRGH = 0) possible is FCY/16 (for BRGx = 0), and the minimum baud rate possible is FCY/(16 * 65536).

Equation 16-2 shows the formula for computation of the baud rate with BRGH = 1.

EQUATION 16-2: UART BAUD RATE WITH BRGH = 1(1,2)

$$\text{Baud Rate} = \frac{FCY}{4 \cdot (BRGx + 1)}$$

$$BRGx = \frac{FCY}{4 \cdot \text{Baud Rate}} - 1$$

Note 1: FCY denotes the instruction cycle clock frequency.

2: Based on TCY = FOSC/2, Doze mode and PLL are disabled.

The maximum baud rate (BRGH = 1) possible is FCY/4 (for BRGx = 0) and the minimum baud rate possible is FCY/(4 * 65536).

Writing a new value to the BRGx register causes the BRG timer to be reset (cleared). This ensures the BRG does not wait for a timer overflow before generating the new baud rate.

EXAMPLE 16-1: BAUD RATE ERROR CALCULATION (BRGH = 0)(1)

Desired Baud Rate	=	FCY/(16 (BRGx + 1))
Solving for BRGx value:		
BRGx	=	((FCY/Desired Baud Rate)/16) – 1
BRGx	=	((4000000/9600)/16) – 1
BRGx	=	25
Calculated Baud Rate	=	4000000/(16 (25 + 1))
	=	9615
Error	=	(Calculated Baud Rate – Desired Baud Rate) Desired Baud Rate
	=	(9615 – 9600)/9600
	=	0.16%

Note 1: Based on TCY = FOSC/2, Doze mode and PLL are disabled.

16.2 Transmitting in 8-Bit Data Mode

1. Set up the UART:
   a) Write appropriate values for data, parity and Stop bits.
   b) Write appropriate baud rate value to the BRGx register.
   c) Set up transmit and receive interrupt enable and priority bits.
2. Enable the UART.
3. Set the UTXEN bit (causes a transmit interrupt).
4. Write data byte to lower byte of TXxREG word. The value will be immediately transferred to the Transmit Shift Register (TSR), and the serial bit stream will start shifting out with next rising edge of the baud clock.
5. Alternately, the data byte may be transferred while UTXEN = 0, and then the user may set UTXEN. This will cause the serial bit stream to begin immediately because the baud clock will start from a cleared state.
6. A transmit interrupt will be generated as per interrupt control bit, UTXISELx.

16.3 Transmitting in 9-Bit Data Mode

1. Set up the UART (as described in **Section 16.2 "Transmitting in 8-Bit Data Mode"**).
2. Enable the UART.
3. Set the UTXEN bit (causes a transmit interrupt).
4. Write TXxREG as a 16-bit value only.
5. A word write to TXxREG triggers the transfer of the 9-bit data to the TSR. Serial bit stream will start shifting out with the first rising edge of the baud clock.
6. A transmit interrupt will be generated as per the setting of control bit, UTXISELx.

16.4 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.

1. Configure the UART for the desired mode.
2. Set UTXEN and UTXBRK – sets up the Break character,
3. Load the TXxREG with a dummy character to initiate transmission (value is ignored).
4. Write '55h' to TXxREG – loads Sync character into the transmit FIFO.
5. After the Break has been sent, the UTXBRK bit is reset by hardware. The Sync character now transmits.

16.5 Receiving in 8-Bit or 9-Bit Data Mode

1. Set up the UART (as described in **Section 16.2 "Transmitting in 8-Bit Data Mode"**).
2. Enable the UART.
3. A receive interrupt will be generated when one or more data characters have been received as per interrupt control bit, URXISELx.
4. Read the OERR bit to determine if an overrun error has occurred. The OERR bit must be reset in software.
5. Read RXxREG.

The act of reading the RXxREG character will move the next character to the top of the receive FIFO, including a new set of PERR and FERR values.

16.6 Operation of $\overline{\text{UxCTS}}$ and $\overline{\text{UxRTS}}$ Control Pins

UARTx Clear to Send ($\overline{\text{UxCTS}}$) and Request to Send ($\overline{\text{UxRTS}}$) are the two hardware controlled pins that are associated with the UART module. These two pins allow the UART to operate in Simplex and Flow Control mode. They are implemented to control the transmission and reception between the Data Terminal Equipment (DTE). The UEN<1:0> bits in the UxMODE register configure these pins.

16.7 Infrared Support

The UART module provides two types of infrared UART support: one is the IrDA clock output to support external IrDA encoder and decoder device (legacy module support) and the other is the full implementation of the IrDA encoder and decoder.

16.8 External IrDA Support – IrDA Clock Output

To support external IrDA encoder and decoder devices, the BCLKx pin (same as the $\overline{\text{UxRTS}}$ pin) can be configured to generate the 16x baud clock. With UEN<1:0> = 11, the BCLKx pin will output the 16x baud clock if the UART module is enabled. It can be used to support the IrDA codec chip.

16.9 Built-in IrDA Encoder and Decoder

The UART has full implementation of the IrDA encoder and decoder as part of the UART module. The built-in IrDA encoder and decoder functionality is enabled using the IREN bit UxMODE<12>. When enabled (IREN = 1), the receive pin (UxRX) acts as the input from the infrared receiver. The transmit pin (UxTX) acts as the output to the infrared transmitter.

REGISTER 16-1: UxMODE: UARTx MODE REGISTER

Upper Byte:							
R/W-0	U-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0(1)	R/W-0(1)
UARTEN	—	USIDL	IREN	RTSMD	—	UEN1	UEN0
bit 15							bit 8

Lower Byte:							
R/W-0 HC	R/W-0	R/W-0 HC	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WAKE	LPBACK	ABAUD	RXINV	BRGH	PDSEL1	PDSEL0	STSEL
bit 7							bit 0

bit 15 **UARTEN:** UARTx Enable bit
1 = UARTx is enabled; all UARTx pins are controlled by UARTx as defined by UEN<1:0>
0 = UARTx is disabled; all UARTx pins are controlled by PORT latches; UARTx power consumption minimal

bit 14 **Unimplemented:** Read as '0'

bit 13 **USIDL:** Stop in Idle Mode bit
1 = Discontinue module operation when device enters Idle mode
0 = Continue module operation in Idle mode

bit 12 **IREN:** IrDA Encoder and Decoder Enable bit
1 = IrDA encoder and decoder enabled
0 = IrDA encoder and decoder disabled

Note: This feature is only available for the 16x BRG mode (BRGH = 0).

bit 11 **RTSMD:** Mode Selection for $\overline{\text{UxRTS}}$ Pin bit
1 = $\overline{\text{UxRTS}}$ pin in Simplex mode
0 = $\overline{\text{UxRTS}}$ pin in Flow Control mode

bit 10 **Unimplemented:** Read as '0'

bit 9-8 **UEN1:UEN0:** UARTx Enable bits
11 = UxTX, UxRX and BCLKx pins are enabled and used; $\overline{\text{UxCTS}}$ pin controlled by PORT latches
10 = UxTX, UxRX, $\overline{\text{UxCTS}}$ and $\overline{\text{UxRTS}}$ pins are enabled and used
01 = UxTX, UxRX and $\overline{\text{UxRTS}}$ pins are enabled and used; $\overline{\text{UxCTS}}$ pin controlled by PORT latches
00 = UxTX and UxRX pins are enabled and used; $\overline{\text{UxCTS}}$ and $\overline{\text{UxRTS}}$/BCLKx pins controlled by PORT latches

Note 1: Bit availability depends on pin availability.

bit 7 **WAKE:** Wake-up on Start bit Detect During Sleep Mode Enable bit
1 = UARTx will continue to sample the UxRX pin; interrupt generated on falling edge, bit cleared in hardware on following rising edge
0 = No wake-up enabled

bit 6 **LPBACK:** UARTx Loopback Mode Select bit
1 = Enable Loopback mode
0 = Loopback mode is disabled

bit 5 **ABAUD:** Auto-Baud Enable bit
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

bit 4 **RXINV:** Receive Polarity Inversion bit
1 = UxRX Idle state is '0'
0 = UxRX Idle state is '1'

Legend:	U = Unimplemented bit, read as '0'		
R = Readable bit	W = Writable bit	HC = Hardware Cleared	HS = Hardware Set
-n = Value at Reset	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 16-1: UxMODE: UARTx MODE REGISTER (CONTINUED)

Upper Byte:							
R/W-0	U-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0(1)	R/W-0(1)
UARTEN	—	USIDL	IREN	RTSMD	—	UEN1	UEN0
bit 15							bit 8

Lower Byte:							
R/W-0 HC	R/W-0	R/W-0 HC	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WAKE	LPBACK	ABAUD	RXINV	BRGH	PDSEL1	PDSEL0	STSEL
bit 7							bit 0

bit 3 **BRGH:** High Baud Rate Enable bit
1 = BRG generates 4 clocks per bit period (4x Baud Clock, High-Speed mode)
0 = BRG generates 16 clocks per bit period (16x Baud Clock, Standard mode)

bit 2-1 **PDSEL1:PDSEL0:** Parity and Data Selection bits
11 = 9-bit data, no parity
10 = 8-bit data, odd parity
01 = 8-bit data, even parity
00 = 8-bit data, no parity

bit 0 **STSEL:** Stop Bit Selection bit
1 = Two Stop bits
0 = One Stop bit

Legend:	U = Unimplemented bit, read as '0'		
R = Readable bit	W = Writable bit	HC = Hardware Cleared	HS = Hardware Set
-n = Value at Reset	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 16-2: UxSTA: UARTx STATUS AND CONTROL REGISTER

Upper Byte:							
R/W-0	R/W-0	R/W-0	U-0	R/W-0 HC	R/W-0	R-0	R-1
UTXISEL1	UTXINV(1)	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R-1	R-0	R-0	R/C-0	R-0
URXISEL1	URXISEL0	ADDEN	RIDLE	PERR	FERR	OERR	URXDA
bit 7							bit 0

bit 15,13 **UTXISEL1:UTXISEL0:** Transmission Interrupt Mode Selection bits

11 = Reserved; do not use
10 = Interrupt when a character is transferred to the Transmit Shift Register and as a result, the transmit buffer becomes empty
01 = Interrupt when the last character is shifted out of the Transmit Shift Register; all transmit operations are completed
00 = Interrupt when a character is transferred to the Transmit Shift Register (this implies there is at least one character open in the transmit buffer)

bit 14 **UTXINV:** IrDA Encoder Transmit Polarity Inversion bit(1)

1 = IrDA encoded UxTX idle state is '1'
0 = IrDA encoded UxTX idle state is '0'

Note 1: Value of bit only affects the transmit properties of the module when the IrDA encoder is enabled (IREN = 1).

bit 12 **Unimplemented:** Read as '0'

bit 11 **UTXBRK:** Transmit Break bit

1 = Send Sync Break on next transmission – Start bit, followed by twelve '0' bits, followed by Stop bit; cleared by hardware upon completion
0 = Sync Break transmission disabled or completed

bit 10 **UTXEN:** Transmit Enable bit

1 = Transmit enabled, UxTX pin controlled by UARTx
0 = Transmit disabled, any pending transmission is aborted and buffer is reset. UxTX pin controlled by PORT.

bit 9 **UTXBF:** Transmit Buffer Full Status bit (Read-Only)

1 = Transmit buffer is full
0 = Transmit buffer is not full, at least one more character can be written

bit 8 **TRMT:** Transmit Shift Register Empty bit (Read-Only)

1 = Transmit Shift Register is empty and transmit buffer is empty (the last transmission has completed)
0 = Transmit Shift Register is not empty, a transmission is in progress or queued

bit 7-6 **URXISEL1:URXISEL0:** Receive Interrupt Mode Selection bits

11 = Interrupt is set on RSR transfer, making the receive buffer full (i.e., has 4 data characters)
10 = Interrupt is set on RSR transfer, making the receive buffer 3/4 full (i.e., has 3 data characters)
0x = Interrupt is set when any character is received and transferred from the RSR to the receive buffer. Receive buffer has one or more characters.

bit 5 **ADDEN:** Address Character Detect bit (bit 8 of received data = 1)

1 = Address Detect mode enabled. If 9-bit mode is not selected, this does not take effect.
0 = Address Detect mode disabled

Legend:	U = Unimplemented bit, read as '0'		
R = Readable bit	W = Writable bit	HS = Hardware Set	HC = Hardware Cleared
-n = Value at Reset	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 16-2: UxSTA: UARTx STATUS AND CONTROL REGISTER (CONTINUED)

Upper Byte:							
R/W-0	R/W-0	R/W-0	U-0	R/W-0 HC	R/W-0	R-0	R-1
UTXISEL1	UTXINV(1)	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R-1	R-0	R-0	R/C-0	R-0
URXISEL1	URXISEL0	ADDEN	RIDLE	PERR	FERR	OERR	URXDA
bit 7							bit 0

bit 4 **RIDLE:** Receiver Idle bit (Read-Only)
1 = Receiver is Idle
0 = Receiver is active

bit 3 **PERR:** Parity Error Status bit (Read-Only)
1 = Parity error has been detected for the current character (character at the top of the receive FIFO)
0 = Parity error has not been detected

bit 2 **FERR:** Framing Error Status bit (Read-Only)
1 = Framing error has been detected for the current character (character at the top of the receive FIFO)
0 = Framing error has not been detected

bit 1 **OERR:** Receive Buffer Overrun Error Status bit (Read/Clear-Only)
1 = Receive buffer has overflowed
0 = Receive buffer has not overflowed (clearing a previously set OERR bit (1 → 0 transition) will reset the receiver buffer and the RSR to the empty state)

bit 0 **URXDA:** Receive Buffer Data Available bit (Read-Only)
1 = Receive buffer has data, at least one more character can be read
0 = Receive buffer is empty

Legend:	U = Unimplemented bit, read as ‘0’		
R = Readable bit	W = Writable bit	HS = Hardware Set	HC = Hardware Cleared
-n = Value at Reset	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

NOTES:

17.0 PARALLEL MASTER PORT

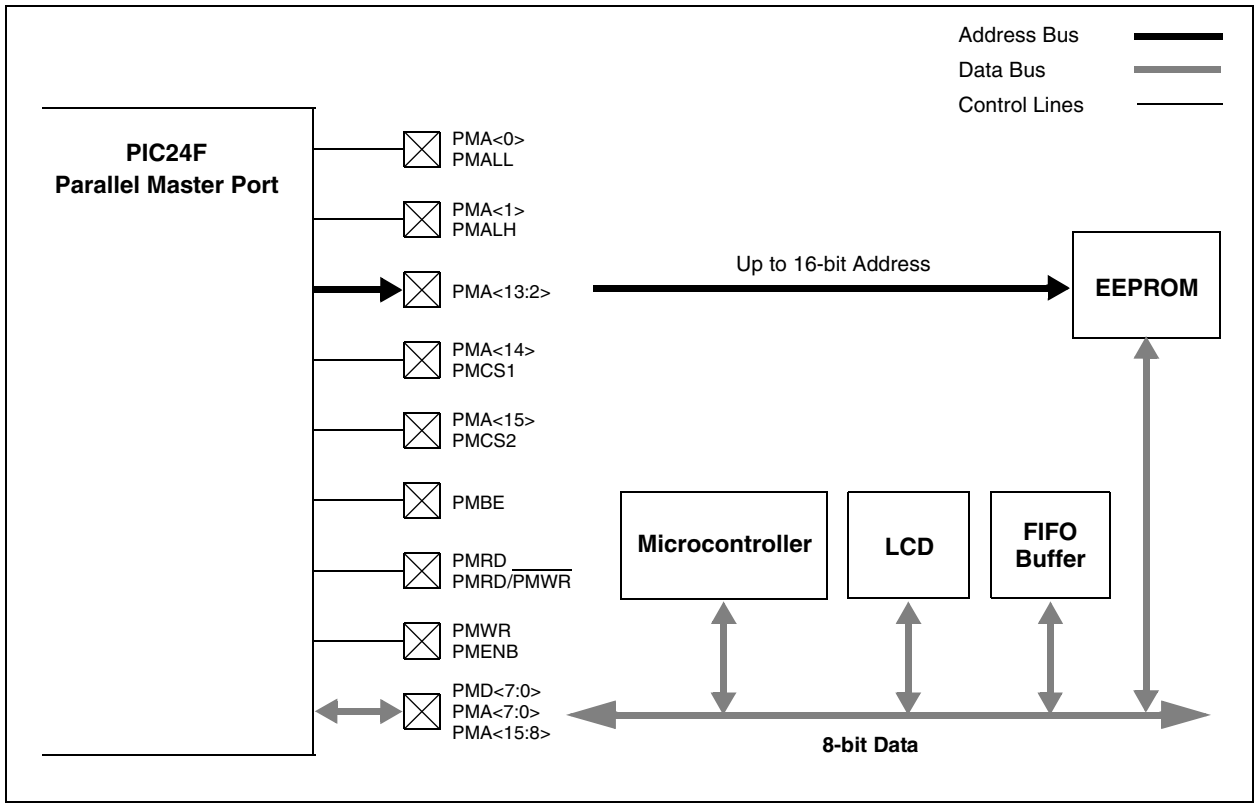
Note:	This data sheet summarizes the features of this group of PIC24FJ devices. It is not intended to be a comprehensive reference source.

The Parallel Master Port module (PMP) is a parallel 8-bit I/O module, specifically designed to communicate with a wide variety of parallel devices, such as communications peripherals, LCDs, external memory devices and microcontrollers. Because the interface to parallel peripherals varies significantly, the PMP is highly configurable.

Key features of the PMP module include:

- Up to 16 Programmable Address Lines
- Up to Two Chip Select Lines
- Programmable Strobe Options
  - Individual Read and Write Strobes or;
  - Read/$\overline{\text{Write}}$ Strobe with Enable Strobe
- Address Auto-Increment/Auto-Decrement
- Programmable Address/Data Multiplexing
- Programmable Polarity on Control Signals
- Legacy Parallel Slave Port Support
- Enhanced Parallel Slave Support
  - Address Support
  - 4-byte Deep Auto-Incrementing Buffer
- Programmable Wait States
- Selectable Input Voltage Levels

FIGURE 17-1: PMP MODULE OVERVIEW



REGISTER 17-1: PMCON: PARALLEL PORT CONTROL REGISTER

Upper Byte:							
R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PMPEN	—	PSIDL	ADRMUX1	ADRMUX0	PTBEEN	PTWREN	PTRDEN
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0(1)	R/W-0(1)	R/W-0(1)	R/W-0	R/W-0	R/W-0
CSF1	CSF0	ALP	CS2P	CS1P	BEP	WRSP	RDSP
bit 7							bit 0

bit 15 **PMPEN:** Parallel Master Port Enable bit
1 = PMP enabled
0 = PMP disabled, no off-chip access performed

bit 14 **Unimplemented:** Read as '0'

bit 13 **PSIDL:** Stop in Idle Mode bit
1 = Discontinue module operation when device enters Idle mode
0 = Continue module operation in Idle mode

bit 12-11 **ADRMUX1:ADRMUX0:** Address/Data Multiplexing Selection bits
11 = Reserved
10 = All 16 bits of address are multiplexed on PMD<7:0> pins
01 = Lower 8 bits of address are multiplexed on PMD<7:0> pins, upper 8 bits are on PMA<15:8>
00 = Address and data appear on separate pins

bit 10 **PTBEEN:** Byte Enable Port Enable bit (16-bit Master mode)
1 = PMBE port enabled
0 = PMBE port disabled

bit 9 **PTWREN:** Write Enable Strobe Port Enable bit
1 = PMWR/PMENB port enabled
0 = PMWR/PMENB port disabled

bit 8 **PTRDEN:** Read/Write Strobe Port Enable bit
1 = PMRD/$\overline{\text{PMWR}}$ port enabled
0 = PMRD/$\overline{\text{PMWR}}$ port disabled

bit 7-6 **CSF1:CSF0:** Chip Select Function bits
11 = Reserved
10 = PMCS1 and PMCS2 function as chip select
01 = PMCS2 functions as chip select, PMCS1 functions as address bit 14
00 = PMCS1 and PMCS2 function as address bits 15 and 14

bit 5 **ALP:** Address Latch Polarity bit(1)
1 = Active-high (PMALL and PMALH)
0 = Active-low ($\overline{\text{PMALL}}$ and $\overline{\text{PMALH}}$)

Note 1: These bits have no effect when their corresponding pins are used as address lines.

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

REGISTER 17-1: PMCON: PARALLEL PORT CONTROL REGISTER (CONTINUED)

Upper Byte:							
R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PMPEN	—	PSIDL	ADRMUX1	ADRMUX0	PTBEEN	PTWREN	PTRDEN
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0(1)	R/W-0(1)	R/W-0(1)	R/W-0	R/W-0	R/W-0
CSF1	CSF0	ALP	CS2P	CS1P	BEP	WRSP	RDSP
bit 7							bit 0

bit 4 **CS2P:** Chip Select 2 Polarity bit(1)
1 = Active-high (PMCS2)
0 = Active-low ($\overline{\text{PMCS2}}$)

bit 3 **CS1P:** Chip Select 1 Polarity bit(1)
1 = Active-high (PMCS1/PMCS)
0 = Active-low ($\overline{\text{PMCS1}}$/$\overline{\text{PMCS}}$)

bit 2 **BEP:** Byte Enable Polarity bit
1 = Byte enable active-high (PMBE)
0 = Byte enable active-low ($\overline{\text{PMBE}}$)

bit 1 **WRSP:** Write Strobe Polarity bit
For Slave modes and Master mode 2 (PMMODE<9:8> = 00, 01, 10):
1 = Write strobe active-high (PMWR)
0 = Write strobe active-low ($\overline{\text{PMWR}}$)
For Master mode 1 (PMMODE<9:8> = 11):
1 = Enable strobe active-high (PMENB)
0 = Enable strobe active-low ($\overline{\text{PMENB}}$)

bit 0 **RDSP:** Read Strobe Polarity bit
For Slave modes and Master mode 2 (PMMODE<9:8> = 00, 01, 10):
1 = Read strobe active-high (PMRD)
0 = Read strobe active-low ($\overline{\text{PMRD}}$)
For Master mode 1 (PMMODE<9:8> = 11):
1 = Read/write strobe active-high (PMRD/$\overline{\text{PMWR}}$)
0 = Read/write strobe active-low ($\overline{\text{PMRD}}$/PMWR)

Note 1: These bits have no effect when their corresponding pins are used as address lines.

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

REGISTER 17-2: PMMODE: PARALLEL PORT MODE REGISTER

bit 15 **BUSY:** Busy bit (Master mode only)
1 = Port is busy (not useful when the processor stall is active)
0 = Port is not busy

bit 14-13 **IRQM1:IRQM0:** Interrupt Request Mode bits
11 = Interrupt generated when Read Buffer 3 is read or Write Buffer 3 is written (Buffered PSP mode)
or on a read or write operation when PMA<1:0> = 11 (Addressable PSP mode only)
10 = No interrupt generated, processor stall activated
01 = Interrupt generated at the end of the read/write cycle
00 = No interrupt generated

bit 12-11 **INCM1:INCM0:** Increment Mode bits
11 = PSP read and write buffers auto-increment (Legacy PSP mode only)
10 = Decrement ADDR<15,13:0> by 1 every read/write cycle
01 = Increment ADDR<15,13:0> by 1 every read/write cycle
00 = No increment or decrement of address

bit 10 **MODE16:** 8/16-bit Mode bit
1 = 16-bit mode: data register is 16 bits, a read or write to the data register invokes two 8-bit transfers
0 = 8-bit mode: data register is 8 bits, a read or .01187d or .01187dt mb(pt 2-12.7(3.3(b(R(t)0.6(M)8.9(o)-1.4(d)11.9(e)-1.4(bi

REGISTER 17-3: PMADDR: PARALLEL PORT ADDRESS REGISTER

Upper Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CS2	CS1	ADDR<13:8>					
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADDR<7:0>							
bit 7							bit 0

bit 15 **CS2:** Chip Select 2 bit
1 = Chip select 2 is active
0 = Chip select 2 is inactive (pin functions as PMA<15>)

bit 14 **CS1:** Chip Select 1 bit
1 = Chip select 1 is active
0 = Chip select 1 is inactive (pin functions as PMA<14>)

bit 13-0 **ADDR13:ADDR0:** Parallel Port Destination Address bits

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

REGISTER 17-4: PMPEN: PARALLEL PORT ENABLE REGISTER

Upper Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTEN15	PTEN14	PTEN13	PTEN12	PTEN11	PTEN10	PTEN9	PTEN8
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTEN7	PTEN6	PTEN5	PTEN4	PTEN3	PTEN2	PTEN1	PTEN0
bit 7							bit 0

bit 15-14 **PTEN15:PTEN14:** PMCSx Strobe Enable bits
1 = PMA15 and PMA14 function as either PMA<15:14> or PMCS2 and PMCS1
0 = PMA15 and PMA14 function as port I/O

bit 13-2 **PTEN13:PTEN2:** PMP Address Port Enable bits
1 = PMA<13:2> function as PMP address lines
0 = PMA<13:2> function as port I/O

bit 1-0 **PTEN1:PTEN0:** PMALH/PMALL Strobe Enable bits
1 = PMA1 and PMA0 function as either PMA<1:0> or PMALH and PMALL
0 = PMA1 and PMA0 pads functions as port I/O

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

REGISTER 17-5: PMSTAT: PARALLEL PORT STATUS REGISTER

Upper Byte:							
R-0	R/W-0 HS	U-0	U-0	R-0	R-0	R-0	R-0
IBF	IBOV	—	—	IB3F	IB2F	IB1F	IB0F
bit 15							bit 8

Lower Byte:							
R-1	R/W-0 HS	U-0	U-0	R-1	R-1	R-1	R-1
OBE	OBUF	—	—	OB3E	OB2E	OB1E	OB0E
bit 7							bit 0

bit 15 **IBF:** Input Buffer Full Status bit
1 = All writable input buffer registers are full
0 = Some or all of the writable input buffer registers are empty

bit 14 **IBOV:** Input Buffer Overflow Status bit
1 = A write attempt to a full input byte register occurred (must be cleared in software)
0 = No overflow occurred

bit 13-12 **Unimplemented:** Read as '0'

bit 11-8 **IBnF:** Input Buffer n Status Full bit
1 = Input buffer contains data that has not been read (reading buffer will clear this bit)
0 = Input buffer does not contain any unread data

bit 7 **OBE:** Output buffer Empty Status bit
1 = All readable output buffer registers are empty
0 = Some or all of the readable output buffer registers are full

bit 6 **OBUF:** Output Buffer Underflow Status bit
1 = A read occurred from an empty output byte register (must be cleared in software)
0 = No underflow occurred

bit 5-4 **Unimplemented:** Read as '0'

bit 3-0 **OBnE:** Output Buffer n Status Empty bit
1 = Output buffer is empty (writing data to the buffer will clear this bit)
0 = Output buffer contains data that has not been transmitted

REGISTER 17-6: PADCFG1: PAD CONFIGURATION CONTROL REGISTER

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	—	—	—	RTSECSEL	PMPTTL
bit 7							bit 0

bit 15-2 **Unimplemented:** Read as '0'

bit 1 **RTSECSEL:** RTCC Seconds Clock Output Select bit
1 = RTCC Seconds Clock is selected for the RTCC pin
0 = RTCC Alarm Pulse is selected for the RTCC pin

Note: To enable the actual RTCC output, the RTCCFG (RTCOE) bit needs to be set.

bit 0 **PMPTTL:** PMP Module TTL Input Buffer Select bit
1 = PMP module uses TTL input buffers
0 = PMP module uses Schmitt input buffers

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

FIGURE 17-2: LEGACY PARALLEL SLAVE PORT EXAMPLE

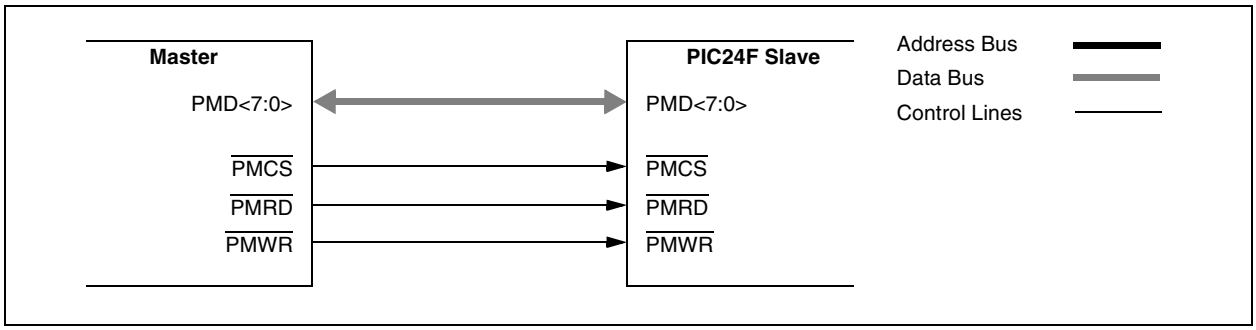


FIGURE 17-3: ADDRESSABLE PARALLEL SLAVE PORT EXAMPLE

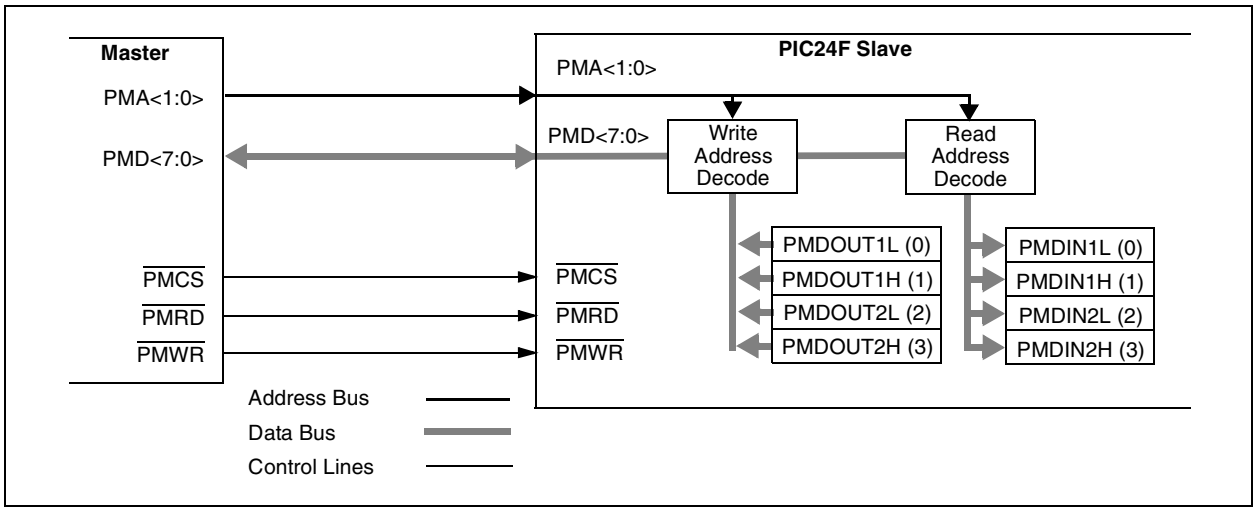


TABLE 17-1: SLAVE MODE ADDRESS RESOLUTION

PMA<1:0>	Output Register (Buffer)	Input Register (Buffer)
00	PMDOUT1<7:0> (0)	PMDIN1<7:0> (0)
01	PMDOUT1<15:8> (1)	PMDIN1<15:8> (1)
10	PMDOUT2<7:0> (2)	PMDIN2<7:0> (2)
11	PMDOUT2<15:8> (3)	PMDIN2<15:8> (3)

FIGURE 17-4: MASTER MODE, DEMULTIPLEXED ADDRESSING (SEPARATE READ AND WRITE STROBES, TWO CHIP SELECTS)

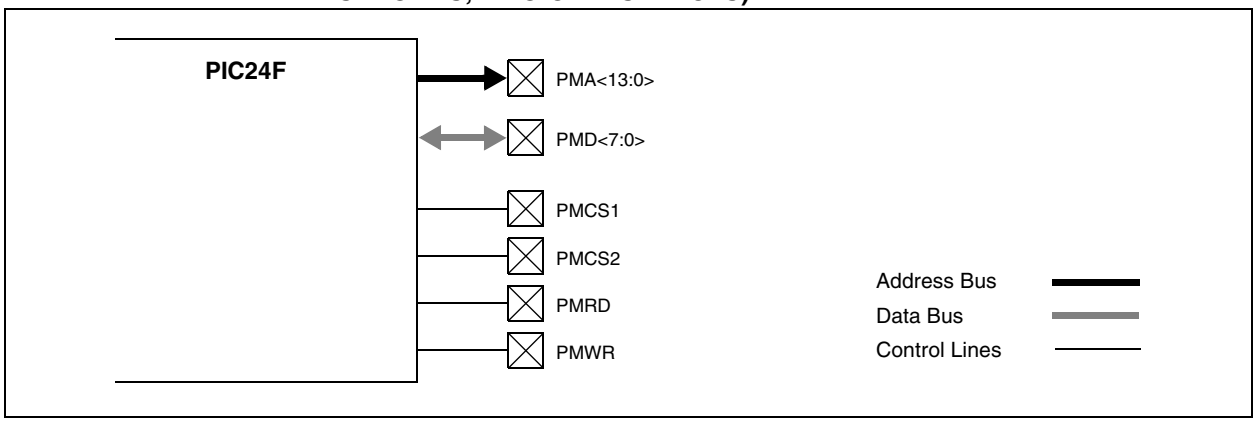


FIGURE 17-5: MASTER MODE, PARTIALLY MULTIPLEXED ADDRESSING (SEPARATE READ AND WRITE STROBES, TWO CHIP SELECTS)

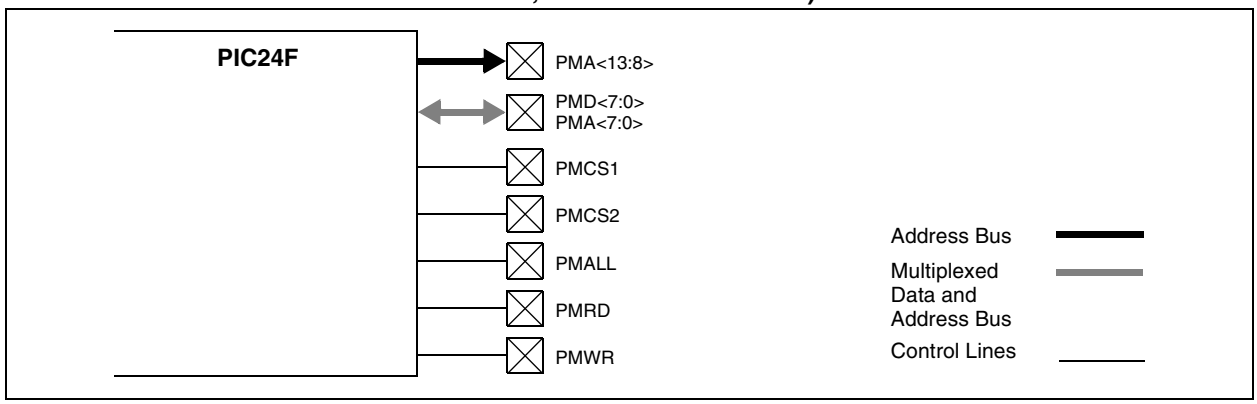


FIGURE 17-6: MASTER MODE, FULLY MULTIPLEXED ADDRESSING (SEPARATE READ AND WRITE STROBES, TWO CHIP SELECTS)

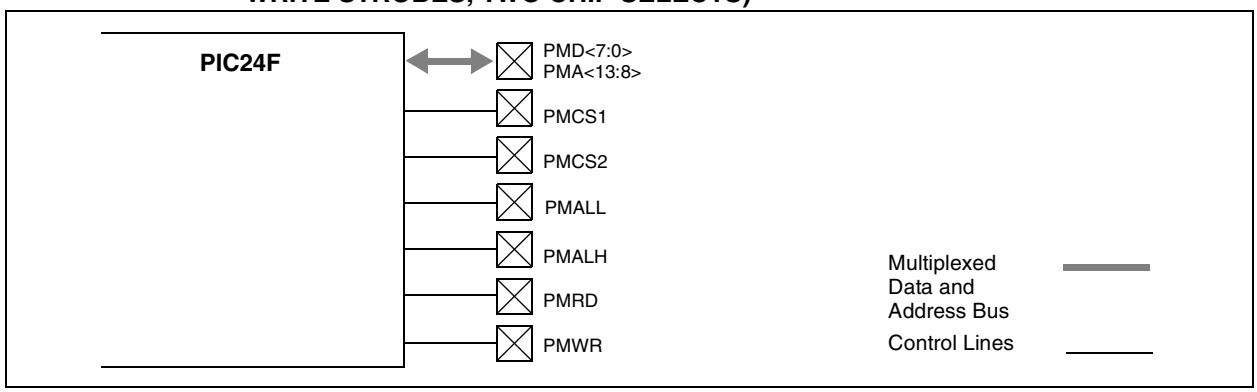


FIGURE 17-7: EXAMPLE OF A MULTIPLEXED ADDRESSING APPLICATION

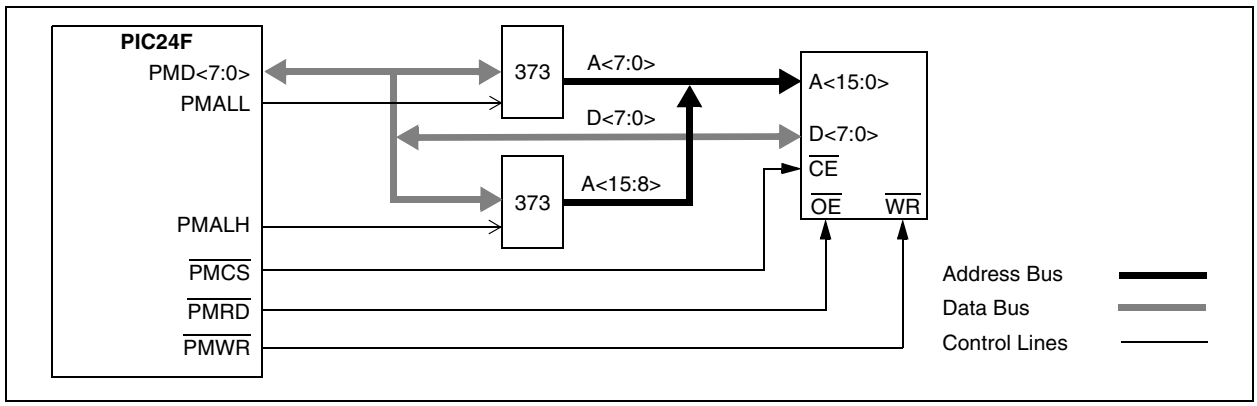


FIGURE 17-8: EXAMPLE OF A PARTIALLY MULTIPLEXED ADDRESSING APPLICATION

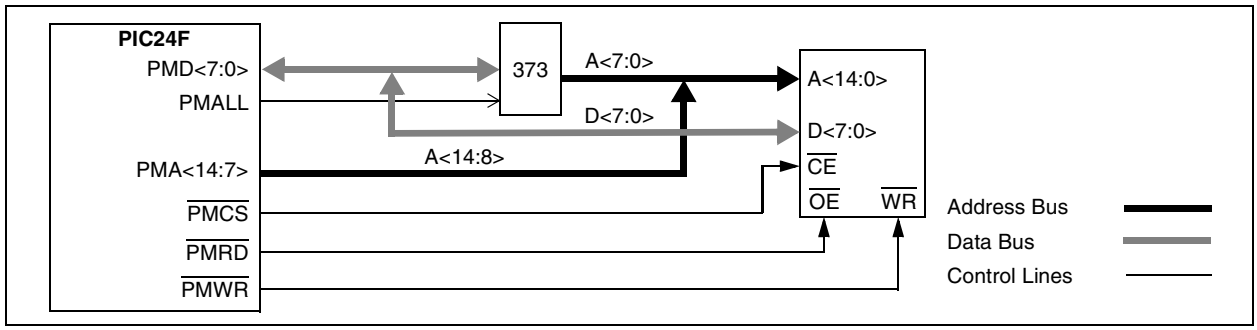


FIGURE 17-9: EXAMPLE OF AN 8-BIT MULTIPLEXED ADDRESS AND DATA APPLICATION

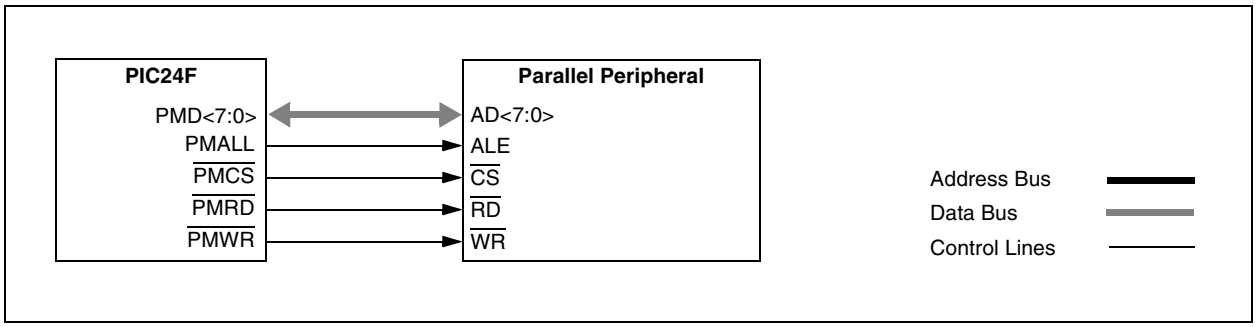


FIGURE 17-10: PARALLEL EEPROM EXAMPLE (UP TO 15-BIT ADDRESS, 8-BIT DATA)

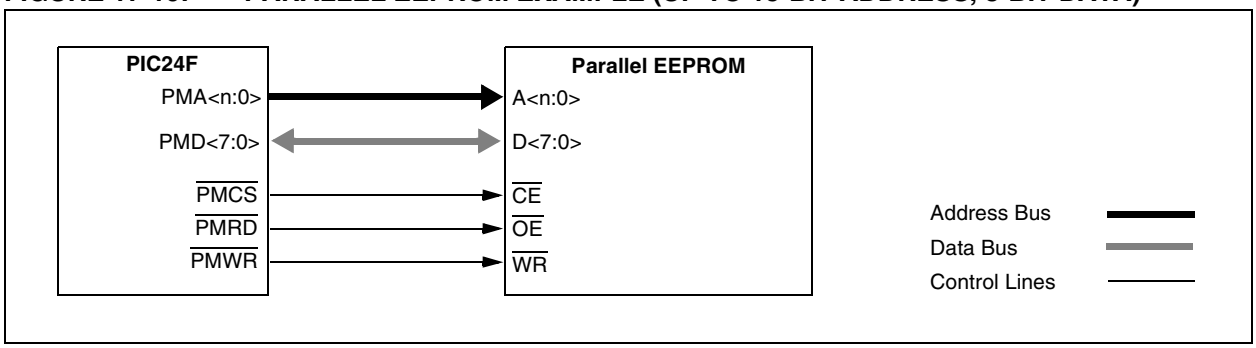


FIGURE 17-11: PARALLEL EEPROM EXAMPLE (UP TO 15-BIT ADDRESS, 16-BIT DATA)

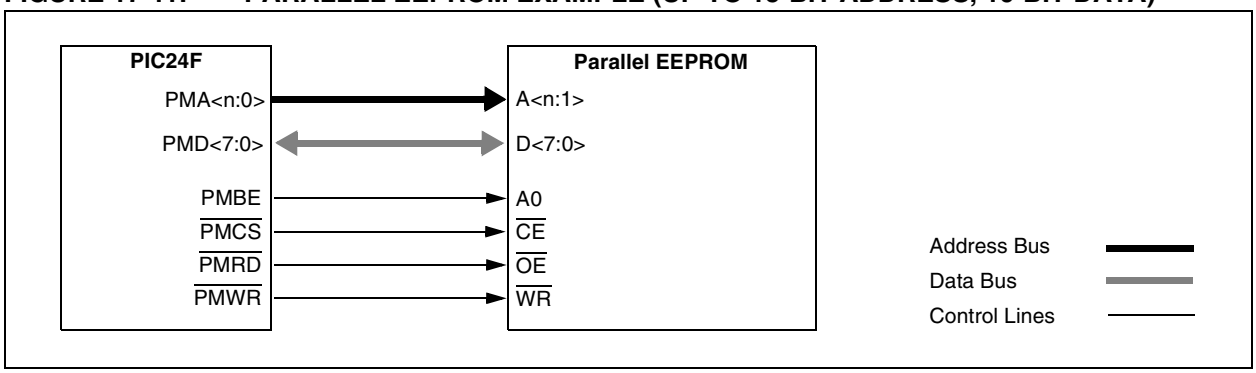
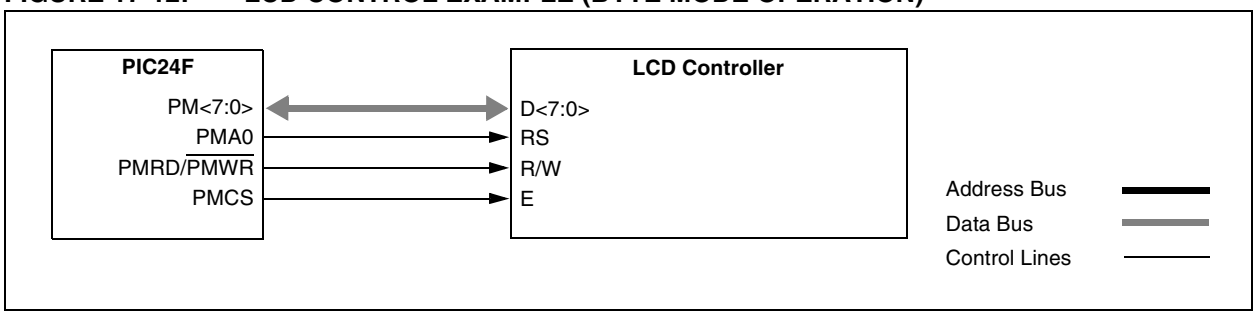


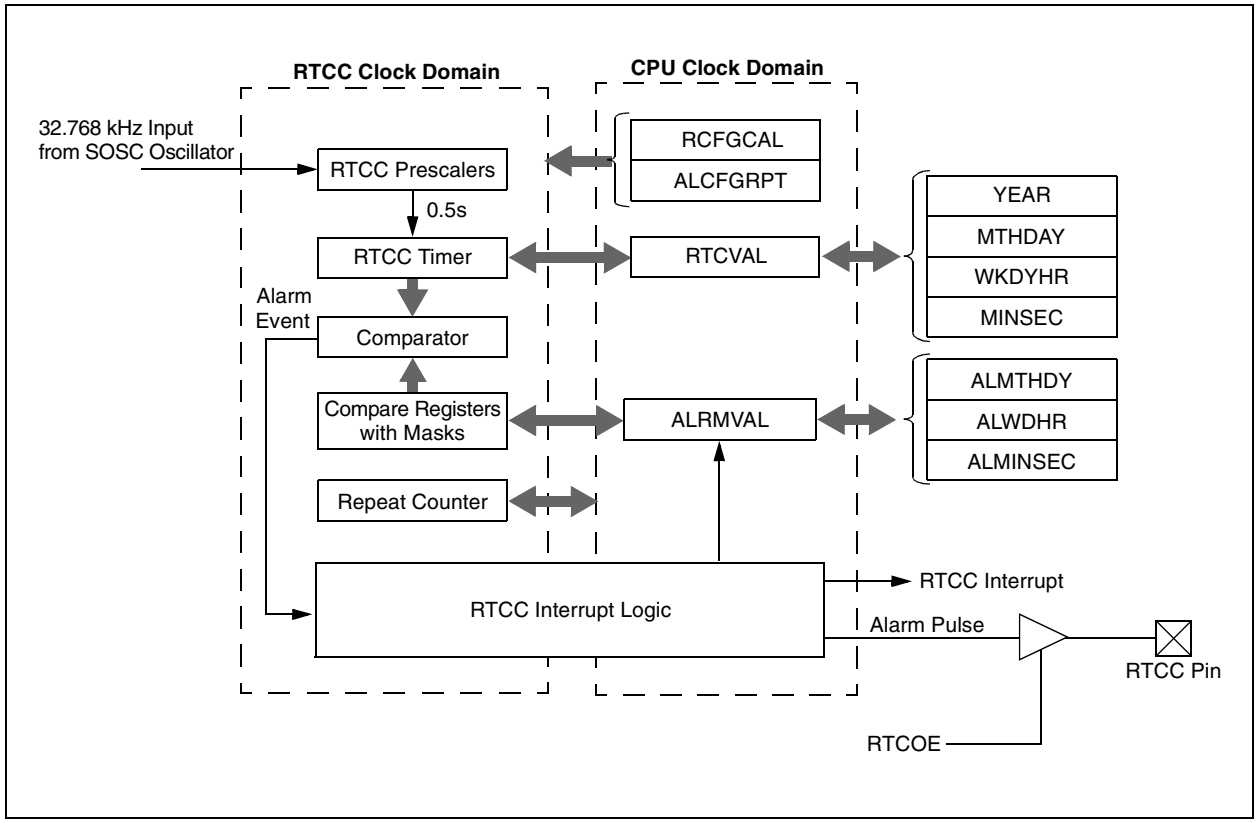
FIGURE 17-12: LCD CONTROL EXAMPLE (BYTE MODE OPERATION)



18.0 REAL-TIME CLOCK AND CALENDAR

Note:	This data sheet summarizes the features of this group of PIC24FJ devices. It is not intended to be a comprehensive reference source.

FIGURE 18-1: RTCC BLOCK DIAGRAM



18.1 RTCC Module Registers

The RTCC module registers are organized into three categories:

- RTCC Control Registers
- RTCC Value Registers
- Alarm Value Registers

18.1.1 REGISTER MAPPING

To limit the register interface, the RTCC Timer and Alarm Time registers are accessed through corresponding register pointers. The RTCC Value register window (RTCVALH and RTCVALL) uses the RTCPTR bits (RCFGCAL<9:8>) to select the desired timer register pair (see Table 18-1).

By writing the RTCVALH byte, the RTCC pointer value RTCPTR<1:0> decrements by one until it reaches '00'. Once it reaches '00', the MINUTES and SECONDS value will be accessible through RTCVALH and RTCVALL until the pointer value is manually changed.

TABLE 18-1: RTCVAL REGISTER MAPPING

RTCPTR <1:0>	RTCC Value Register Window	
	RTCVAL<15:8>	RTCVAL<7:0>
00	MINUTES	SECONDS
01	WEEKDAY	HOURS
10	MONTH	DAY
11	—	YEAR

The Alarm Value register window (ALRMVALH and ALRMVALL) uses the ALRMPTR bits (ALCFGRPT<9:8>) to select the desired alarm register pair (see Table 18-2).

By writing the ALRMVALH byte, the alarm pointer value ALRMPTR<1:0> decrements by one until it reaches '00'. Once it reaches '00', the ALRMMIN and ALRMSEC value will be accessible through ALRMVALH and ALRMVALL until the pointer value is manually changed.

TABLE 18-2: ALRMVAL REGISTER MAPPING

ALRMPTR <1:0>	Alarm Value Register Window	
	ALRMVAL<15:8>	ALRMVAL<7:0>
00	ALRMMIN	ALRMSEC
01	ALRMWD	ALRMHR
10	ALRMMNTH	ALRMDAY
11	—	—

Considering that the 16-bit core does not distinguish between 8-bit and 16-bit read operations, the user must be aware that when reading either the ALRMVALH or ALRMVALL bytes will decrement the ALRMPTR<1:0> value. The same applies to the RTCVALH or RTCVALL bytes with the RTCPTR<1:0> being decremented.

Note:	This only applies to read operations and not write operations.

18.1.2 RTCC CONTROL REGISTERS

REGISTER 18-1: RCFGCAL: RTCC CALIBRATION AND CONFIGURATION REGISTER(1)

Upper Byte:							
R/W-0	U-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0
RTCEN(2)	—	RTCWREN	RTCSYNC	HALFSEC	RTCOE	RTCPTR1	RTCPTR0
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0
bit 7							bit 0

bit 15 **RTCEN:** RTCC Enable bit(2)
- 1 = RTCC module is enabled
- 0 = RTCC module is disabled

bit 14 **Unimplemented:** Read as '0'

bit 13 **RTCWREN:** RTCC Value Registers Write Enable bit
- 1 = RTCVALH and RTCVALL registers can be written to by the user
- 0 = RTCVALH and RTCVALL registers are locked out from being written to by the user

bit 12 **RTCSYNC:** RTCC Value Registers Read Synchronization bit
- 1 = RTCVALH, RTCVALL and ALCFGRPT registers can change while reading due to a rollover ripple resulting in an invalid data read. If the register is read twice and results in the same data, the data can be assumed to be valid.
- 0 = RTCVALH, RTCVALL or ALCFGRPT registers can be read without concern over a rollover ripple

bit 11 **HALFSEC:** Half-Second Status bit
- 1 = Second half period of a second
- 0 = First half period of a second

Note: This bit is read-only. It is cleared to '0' on a write to the lower half of the MINSEC register.

bit 10 **RTCOE:** RTCC Output Enable bit
- 1 = RTCC output enabled
- 0 = RTCC output disabled

Note 1: The RCFGCAL register is only affected by a POR.

2: A write to the RTCEN bit is only allowed when RTCWREN = 1.

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

REGISTER 18-1: RCFGCAL: RTCC CALIBRATION AND CONFIGURATION REGISTER(1) (CONTINUED)

Upper Byte:							
R/W-0	U-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0
RTCEN(2)	—	RTCWREN	RTCSYNC	HALFSEC	RTCOE	RTCPTR1	RTCPTR0
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0
bit 7							bit 0

bit 9-8 **RTCPTR1:RTCPTR0:** RTCC Value Register Window Pointer bits

Points to the corresponding RTCC Value registers when reading RTCVALH and RTCVALL registers; the RTCPTR<1:0> value decrements on every read or write of RTCVALH until it reaches '00'.

RTCVAL<15:8>:
00 = MINUTES
01 = WEEKDAY
10 = MONTH
11 = Reserved

RTCVAL<7:0>:
00 = SECONDS
01 = HOURS
10 = DAY
11 = YEAR

bit 7-0 **CAL7:CAL0:** RTC Drift Calibration bits

01111111 = Maximum positive adjustment; adds 508 RTC clock pulses every one minute
...
01111111 = Minimum positive adjustment; adds 4 RTC clock pulses every one minute
00000000 = No adjustment
11111111 = Minimum negative adjustment; subtracts 4 RTC clock pulses every one minute
...
10000000 = Maximum negative adjustment; subtracts 512 RTC clock pulses every one minute

Note 1: The RCFGCAL register is only affected by a POR.

2: A write to the RTCEN bit is only allowed when RTCWREN = 1.

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

REGISTER 18-2: PADCFG1: PAD CONFIGURATION CONTROL REGISTER

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	—	—	—	RTSECSEL	PMPTTL
bit 7							bit 0

bit 15-2 **Unimplemented:** Read as '0'

bit 1 **RTSECSEL:** RTCC Seconds Clock Output Select bit
1 = RTCC seconds clock is selected for the RTCC pin
0 = RTCC alarm pulse is selected for the RTCC pin

Note: To enable the actual RTCC output, the RTCCFG (RTCOE) bit needs to be set.

bit 0 **PMPTTL:** PMP Module TTL Input Buffer Select bit
1 = PMP module uses TTL input buffers
0 = PMP module uses Schmitt input buffers

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

REGISTER 18-3: ALCFGRPT: ALARM CONFIGURATION REGISTER

Upper Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ALRMEN	CHIME	AMASK3	AMASK2	AMASK1	AMASK0	ALRMPTR1	ALRMPTR0
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ARPT7	ARPT6	ARPT5	ARPT4	ARPT3	ARPT2	ARPT1	ARPT0
bit 7							bit 0

bit 15 **ALRMEN:** Alarm Enable bit

1 = Alarm is enabled (cleared automatically after an alarm event whenever ARPT<7:0> = 00 and CHIME = 0)

0 = Alarm is disabled

bit 14 **CHIME:** Chime Enable bit

1 = Chime is enabled; ARPT<7:0> is allowed to roll over from 00h to FFh

0 = Chime is disabled; ARPT<7:0> stops once it reaches 00h

bit 13-10 **AMASK3:AMASK0:** Alarm Mask Configuration bits

0000 = Every half second
0001 = Every second
0010 = Every 10 seconds
0011 = Every minute
0100 = Every 10 minutes
0101 = Every hour
0110 = Once a day
0111 = Once a week
1000 = Once a month
1001 = Once a year (except when configured for February 29th, once every 4 years)
101x = Reserved – do not use
11xx = Reserved – do not use

bit 9-8 **ALRMPTR1:ALRMPTR0:** Alarm Value Register Window Pointer bits

Points to the corresponding Alarm Value registers when reading ALRMVALH and ALRMVALL registers; the ALRMPTR<1:0> value decrements on every read or write of ALRMVALH until it reaches '00'.

ALRMVAL<15:8>:
00 = ALRMMIN
01 = ALRMWD
10 = ALRMMNTH
11 = Unimplemented

ALRMVAL<7:0>:
00 = ALRMSEC
01 = ALRMHR
10 = ALRMDAY
11 = Unimplemented

bit 7-0 **ARPT7:ARPT0:** Alarm Repeat Counter Value bits

11111111 = Alarm will repeat 255 more times
...
00000000 = Alarm will not repeat

The counter decrements on any alarm event. The counter is prevented from rolling over from 00h to FFh unless CHIME = 1.

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

18.1.3 RTCVAL REGISTER MAPPINGS

REGISTER 18-4: YEAR: YEAR VALUE REGISTER(1)

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
YRTEN3	YRTEN2	YRTEN1	YRTEN0	YRONE3	YRONE2	YRONE1	YRONE0
bit 7							bit 0

bit 15-8 **Unimplemented:** Read as '0'

bit 7-4 **YRTEN3:YRTEN0:** Binary Coded Decimal Value of Year's Tens Digit; Contains a value from 0 to 9

bit 3-0 **YRONE3:YRONE0:** Binary Coded Decimal Value of Year's Ones Digit; Contains a value from 0 to 9

Note 1: A write to the YEAR register is only allowed when RTCWREN = 1.

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

REGISTER 18-5: MTHDY: MONTH AND DAY VALUE REGISTER(1)

Upper Byte:							
U-0	U-0	U-0	R-x	R-x	R-x	R-x	R-x
—	—	—	MTHTEN0	MTHONE3	MTHONE2	MTHONE1	MTHONE0
bit 15							bit 8

Lower Byte:							
U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	DAYTEN1	DAYTEN0	DAYONE3	DAYONE2	DAYONE1	DAYONE0
bit 7							bit 0

bit 15-13 **Unimplemented:** Read as '0'

bit 12 **MTHTEN0:** Binary Coded Decimal Value of Month's Tens Digit; Contains a value of 0 or 1

bit 11-8 **MTHONE3:MTHONE0:** Binary Coded Decimal Value of Month's Ones Digit; Contains a value from 0 to 9

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

bit 5-4 **DAYTEN1:DAYTEN0:** Binary Coded Decimal Value of Day's Tens Digit; Contains a value from 0 to 3

bit 3-0 **DAYONE3:DAYONE0:** Binary Coded Decimal Value of Day's Ones Digit; Contains a value from 0 to 9

Note 1: A write to this register is only allowed when RTCWREN = 1.

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

REGISTER 18-6: WKDYHR: WEEKDAY AND HOURS VALUE REGISTER(1)

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x
—	—	—	—	—	WDAY2	WDAY1	WDAY0
bit 15							bit 8

Lower Byte:							
U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	HRTEN1	HRTEN0	HRONE3	HRONE2	HRONE1	HRONE0
bit 7							bit 0

bit 15-11 **Unimplemented:** Read as '0'

bit 10-8 **WDAY2:WDAY0:** Binary Coded Decimal Value of Weekday Digit; Contains a value from 0 to 6

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

bit 5-4 **HRTEN1:HRTEN0:** Binary Coded Decimal Value of Hour's Tens Digit; Contains a value from 0 to 2

bit 3-0 **HRONE3:HRONE0:** Binary Coded Decimal Value of Hour's Ones Digit; Contains a value from 0 to 9

Note 1: A write to this register is only allowed when RTCWREN = 1.

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

REGISTER 18-7: MINSEC: MINUTES AND SECONDS VALUE REGISTER

Upper Byte:							
U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	MINTEN2	MINTEN1	MINTEN0	MINONE3	MINONE2	MINONE1	MINONE0
bit 15							bit 8

Lower Byte:							
U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	SECTEN2	SECTEN1	SECTEN0	SECONE3	SECONE2	SECONE1	SECONE0
bit 7							bit 0

bit 15 **Unimplemented:** Read as '0'

bit 14-12 **MINTEN2:MINTEN0:** Binary Coded Decimal Value of Minute's Tens Digit; Contains a value from 0 to 5

bit 11-8 **MINONE3:MINONE0:** Binary Coded Decimal Value of Minute's Ones Digit; Contains a value from 0 to 9

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

bit 6-4 **SECTEN2:SECTEN0:** Binary Coded Decimal Value of Second's Tens Digit; Contains a value from 0 to 5

bit 3-0 **SECONE3:SECONE0:** Binary Coded Decimal Value of Second's Ones Digit; Contains a value from 0 to 9

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

18.1.4 ALRMVAL REGISTER MAPPINGS

REGISTER 18-8: ALMTHDY: ALARM MONTH AND DAY VALUE REGISTER(1)

Upper Byte:							
U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	MTHTEN0	MTHONE3	MTHONE2	MTHONE1	MTHONE0
bit 15							bit 8

Lower Byte:							
U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	DAYTEN1	DAYTEN0	DAYONE3	DAYONE2	DAYONE1	DAYONE0
bit 7							bit 0

bit 15-13 **Unimplemented:** Read as '0'

bit 12 **MTHTEN0:** Binary Coded Decimal Value of Month's Tens Digit; Contains a value of 0 or 1

bit 11-8 **MTHONE3:MTHONE0:** Binary Coded Decimal Value of Month's Ones Digit; Contains a value from 0 to 9

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

bit 5-4 **DAYTEN1:DAYTEN0:** Binary Coded Decimal Value of Day's Tens Digit; Contains a value from 0 to 3

bit 3-0 **DAYONE3:DAYONE0:** Binary Coded Decimal Value of Day's Ones Digit; Contains a value from 0 to 9

Note 1: A write to this register is only allowed when RTCWREN = 1.

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

REGISTER 18-9: ALWDHR: ALARM WEEKDAY AND HOURS VALUE REGISTER(1)

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x
—	—	—	—	—	WDAY2	WDAY1	WDAY0
bit 15							bit 8

Lower Byte:							
U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	HRTEN1	HRTEN0	HRONE3	HRONE2	HRONE1	HRONE0
bit 7							bit 0

bit 15-11 **Unimplemented:** Read as '0'

bit 10-8 **WDAY2:WDAY0:** Binary Coded Decimal Value of Weekday Digit; Contains a value from 0 to 6

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

bit 5-4 **HRTEN1:HRTEN0:** Binary Coded Decimal Value of Hour's Tens Digit; Contains a value from 0 to 2

bit 3-0 **HRONE3:HRONE0:** Binary Coded Decimal Value of Hour's Ones Digit; Contains a value from 0 to 9

Note 1: A write to this register is only allowed when RTCWREN = 1.

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

REGISTER 18-10: ALMINSEC: ALARM MINUTES AND SECONDS VALUE REGISTER

Upper Byte:							
U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	MINTEN2	MINTEN1	MINTEN0	MINONE3	MINONE2	MINONE1	MINONE0
bit 15							bit 8

Lower Byte:							
U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	SECTEN2	SECTEN1	SECTEN0	SECONE3	SECONE2	SECONE1	SECONE0
bit 7							bit 0

bit 15 **Unimplemented:** Read as '0'

bit 14-12 **MINTEN2:MINTEN0:** Binary Coded Decimal Value of Minute's Tens Digit; Contains a value from 0 to 5

bit 11-8 **MINONE3:MINONE0:** Binary Coded Decimal Value of Minute's Ones Digit; Contains a value from 0 to 9

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

bit 6-4 **SECTEN2:SECTEN0:** Binary Coded Decimal Value of Second's Tens Digit; Contains a value from 0 to 5

bit 3-0 **SECONE3:SECONE0:** Binary Coded Decimal Value of Second's Ones Digit; Contains a value from 0 to 9

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

18.2 Calibration

The real-time crystal input can be calibrated using the periodic auto-adjust feature. When properly calibrated, the RTCC can provide an error of less than 3 seconds per month. This is accomplished by finding the number of error clock pulses and storing the value into the lower half of the RCFGCAL register. The 8-bit signed value loaded into the lower half of RCFGCAL is multiplied by four and will be either added or subtracted from the RTCC timer, once every minute. Refer to the steps below for RTCC calibration:

1. Using another timer resource on the device, the user must find the error of the 32.768 kHz crystal.
2. Once the error is known, it must be converted to the number of error clock pulses per minute.

   Formula box:
   (Ideal frequency (32,768) – measured frequency) * 60 = clocks per minute

3. a) If the oscillator is faster then ideal (negative result form step 2), the RCFGCAL register value needs to be negative. This causes the specified number of clock pulses to be subtracted from the timer counter once every minute.

   b) If the oscillator is slower then ideal (positive result from step 2) the RCFGCAL register value needs to be positive. This causes the specified number of clock pulses to be subtracted from the timer counter once every minute.
4. Load the RCFGCAL register with the correct value.

Writes to the lower half of the RCFGCAL register should only occur when the timer is turned off, or immediately after the rising edge of the seconds pulse.

Note: It is up to the user to include in the error value the initial error of the crystal, drift due to temperature and drift due to crystal aging.

18.3 Alarm

- Configurable from half second to one year
- Enabled using the ALRMEN bit (ALCFGRPT<7>, Register 18-3)
- One-time alarm and repeat alarm options available

18.3.1 CONFIGURING THE ALARM

The alarm feature is enabled using the ALRMEN bit. This bit is cleared when an alarm is issued. Writes to ALRMVALH:ALRMVALL should only take place when ALRMEN = 0.

As shown in Figure 18-2, the interval selection of the alarm is configured through the AMASK bits (ALCFGRPT<13:10>). These bits determine which and how many digits of the alarm must match the clock value for the alarm to occur. The alarm can also be configured to repeat based on a preconfigured interval. The amount of times this occurs once the alarm is enabled is stored in the lower half of the ALCFGRPT register.

When ALCFGRPT = 00 and CHIME bit = 0 (ALCFGRPT<14>), the repeat function is disabled and only a single alarm will occur. The alarm can be repeated up to 255 times by loading the lower half of the ALCFGRPT register with FFh.

After each alarm is issued, the ALCFGRPT register is decremented by one. Once the register has reached '00', the alarm will be issued one last time, after which the ALRMEN bit will be cleared automatically and the alarm will turn off. Indefinite repetition of the alarm can occur if the CHIME bit = 1. Instead of the alarm being disabled when the ALCFGRPT register reaches '00', it will roll over to FF and continue counting indefinitely when CHIME = 1.

18.3.2 ALARM INTERRUPT

At every alarm event an interrupt is generated. In addition, an alarm pulse output is provided that operates at half the frequency of the alarm. This output is completely synchronous to the RTCC clock and can be used as a trigger clock to other peripherals.

Note:	Changing any of the registers, other then the RCFGCAL and ALCFGRPT registers and the CHIME bit while the alarm is enabled (ALRMEN = 1), can result in a false alarm event leading to a false alarm interrupt. To avoid a false alarm event, the timer and alarm values should only be changed while the alarm is disabled (ALRMEN = 0). It is recommended that the ALCFGRPT register and CHIME bit be changed when RTCSYNC = 0.

FIGURE 18-2: ALARM MASK SETTINGS

Alarm Mask Setting (AMASK3:AMASK0)	Day of the Week	Month	Day	Hours	Minutes	Seconds
0000 – Every half second 0001 – Every second						
0010 – Every 10 seconds						s
0011 – Every minute						s s
0100 – Every 10 minutes					m	s s
0101 – Every hour					m m	s s
0110 – Every day				h h	m m	s s
0111 – Every week	d			h h	m m	s s
1000 – Every month			d d	h h	m m	s s
1001 – Every year(1)		m m	d d	h h	m m	s s

Note 1: Annually, except when configured for February 29.

NOTES:

19.0 PROGRAMMABLE CYCLIC REDUNDANCY CHECK (CRC) GENERATOR

The programmable CRC generator offers the following features:

- User-programmable polynomial CRC equation
- Interrupt output
- Data FIFO

19.1 Registers

There are four registers used to control programmable CRC operation:

- CRCCON
- CRCXOR
- CRCDAT
- CRCWDAT

REGISTER 19-1: CRCCON: CRC CONTROL REGISTER

Upper Byte:							
U-0	U-0	R/W-0	R-0	R-0	R-0	R-0	R-0
—	—	CSIDL	VWORD4	VWORD3	VWORD2	VWORD1	VWORD0
bit 15							bit 8

Lower Byte:							
R-0	R-1	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CRCFUL	CRCMPT	—	CRCGO	PLEN3	PLEN2	PLEN1	PLEN0
bit 7							bit 0

bit 15-14 **Unimplemented:** Read as '0'

bit 13 **CSIDL:** CRC Stop in Idle Mode bit
1 = Discontinue module operation when device enters Idle mode
0 = Continue module operation in Idle mode

bit 12-8 **VWORD4:VWORD0:** Pointer Value bits
Indicates the number of valid words in the FIFO. Has a maximum value of 8 when PLEN3:PLEN0 > 7, or 16 when PLEN3:PLEN0 ≤ 7.

bit 7 **CRCFUL**: FIFO Full bit
1 = FIFO is full
0 = FIFO is not full

bit 6 **CRCMPT:** FIFO Empty Bit
1 = FIFO is empty
0 = FIFO is not empty

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

bit 4 **CRCGO:** Start CRC bit
1 = Start CRC serial shifter
0 = CRC serial shifter turned off

bit 3-0 **PLEN3:PLEN0:** Polynomial Length bits
Denotes the length of the polynomial to be generated minus 1.

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

19.2 Overview

The module implements a software configurable CRC generator. The terms of the polynomial and its length can be programmed using the CRCXOR (X<15:1>) bits and the CRCCON (PLEN3:PLEN0) bits, respectively.

Consider the CRC equation:

$$x^{16} + x^{12} + x^5 + 1$$

To program this polynomial into the CRC generator, the CRC register bits should be set as shown in Table 19-1.

TABLE 19-1: EXAMPLE CRC SETUP

Bit Name	Bit Value
PLEN3:PLEN0	1111
X<15:1>	000100000010000

Note that for the value of X<15:1>, the 12th bit and the 5th bit are set to '1', as required by the equation. The 0th bit required by the equation is always XORed. For a 16-bit polynomial, the 16th bit is also always assumed to be XORed; therefore, the X<15:1> bits do not have the 0th bit or the 16th bit.

The topology of a standard CRC generator is shown in Figure 19-2.

FIGURE 19-1: CRC SHIFTER DETAILS

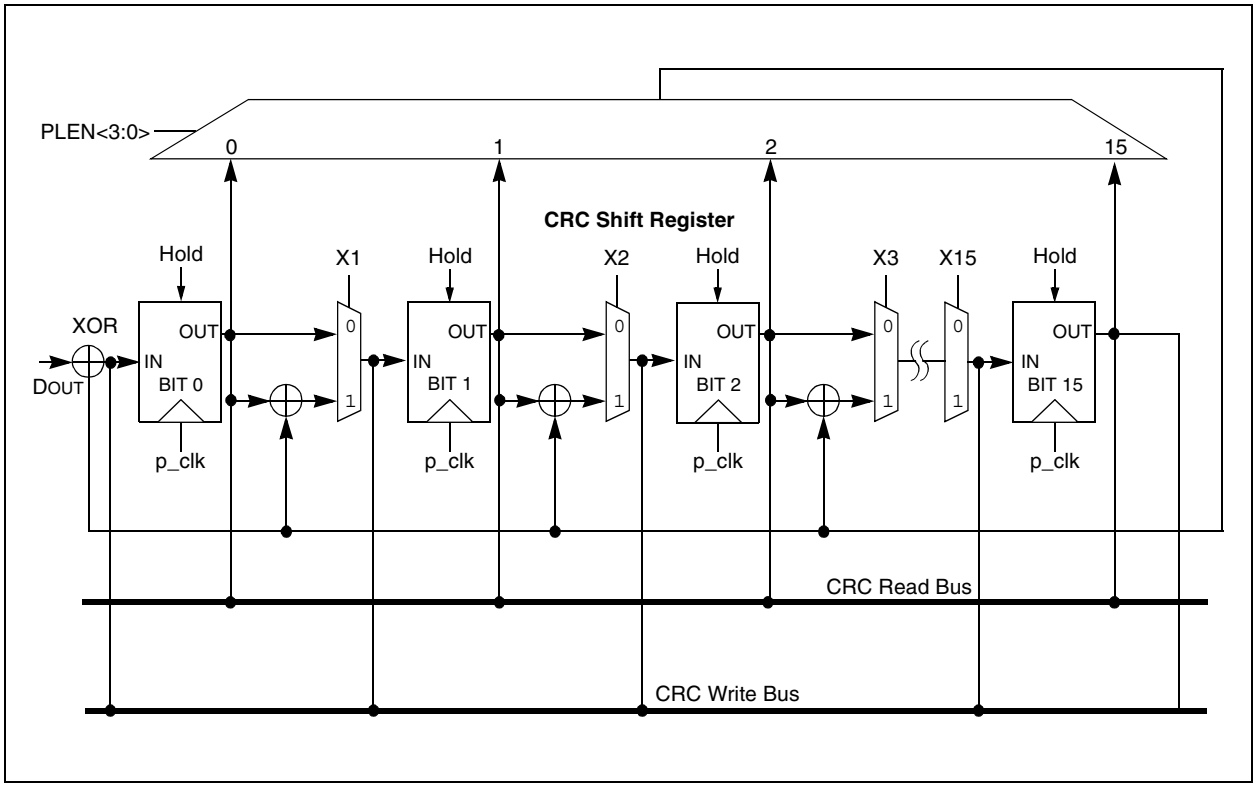
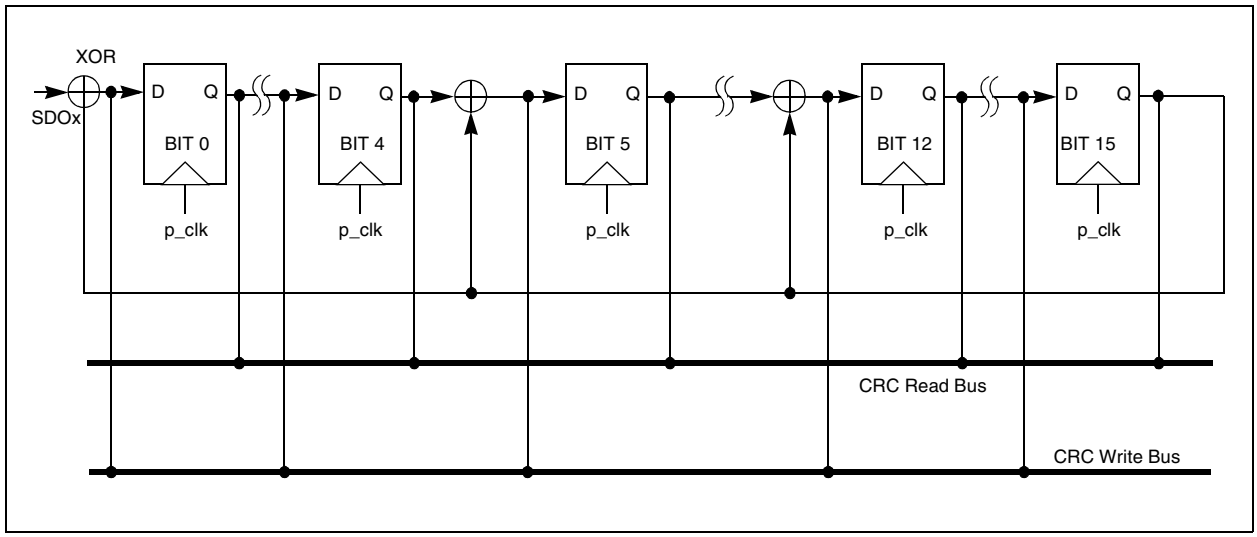


FIGURE 19-2: CRC GENERATOR RECONFIGURED FOR $x^{16} + x^{12} + x^5 + 1$



19.3 User Interface

19.3.1 DATA INTERFACE

To start serial shifting, a '1' must be written to the CRCGO bit.

The module incorporates a FIFO that is 8 deep when PLEN (PLEN<3:0>) > 7, and 16 deep otherwise. The data for which the CRC is to be calculated must first be written into the FIFO. The smallest data element that can be written into the FIFO is one byte. For example, if PLEN = 5, then the size of the data is PLEN + 1 = 6. The data must be written as follows:

data[5:0] = crc_input[5:0]

data[7:6] = 'bxx

Once data is written into the CRCWDAT MSb (as defined by PLEN), the value of VWORD (VWORD<4:0>) increments by one. The serial shifter starts shifting data into the CRC engine when CRCGO = 1 and VWORD > 0. When the MSb is shifted out, VWORD decrements by one. The serial shifter continues shifting until the VWORD reaches 0. Therefore, for a given value of PLEN, it will take (PLEN + 1) * VWORD number of clock cycles to complete the CRC calculations.

When VWORD reaches 8 (or 16), the CRCFUL bit will be set. When VWORD reaches 0, the CRCMPT bit will be set.

To continually feed data into the CRC engine, the recommended mode of operation is to initially "prime" the FIFO with a sufficient number of words so no interrupt is generated before the next word can be written. Once that is done, start the CRC by setting the CRCGO bit to '1'. From that point onward, the VWORD bits should be polled. If they read less than 8 or 16, another word can be written into the FIFO.

To empty words already written into a FIFO, the CRCGO bit must be set to '1' and the CRC shifter allowed to run until the CRCMPT bit is set.

Also, to get the correct CRC reading, it will be necessary to wait for the CRCMPT bit to go high before reading the CRCWDAT register.

If a word is written when the CRCFUL bit is set, the VWORD pointer will roll over to 0. The hardware will then behave as if the FIFO is empty. However, the condition to generate an interrupt will not be met; therefore, no interrupt will be generated (See **Section 19.3.2 "Interrupt Operation"**).

At least one instruction cycle must pass after a write to CRCWDAT before a read of the VWORD bits is done.

19.3.2 INTERRUPT OPERATION

When VWORD4:VWORD0 makes a transition from a value of '1' to '0', an interrupt will be generated.

19.4 Operation in Power Save Modes

19.4.1 SLEEP MODE

If Sleep mode is entered while the module is operating, the module will be suspended in its current state until clock execution resumes.

19.4.2 IDLE MODE

To continue full module operation in Idle mode, the CSIDL bit must be cleared prior to entry into the mode.

If CSIDL = 1, the module will behave the same way as it does in Sleep mode; pending interrupt events will be passed on, even though the module clocks are not available.

NOTES:

20.0 10-BIT HIGH-SPEED A/D CONVERTER

Note:	This data sheet summarizes the features of this group of PIC24FJ devices. It is not intended to be a comprehensive reference source.

The 10-bit A/D converter has the following key features:

- Successive Approximation (SAR) conversion
- Conversion speeds of up to 500 ksps
- Up to 16 analog input pins
- External voltage reference input pins
- Automatic Channel Scan mode
- Selectable conversion trigger source
- 16-word conversion result buffer
- Selectable Buffer Fill modes
- Four result alignment options
- Operation during CPU Sleep and Idle modes

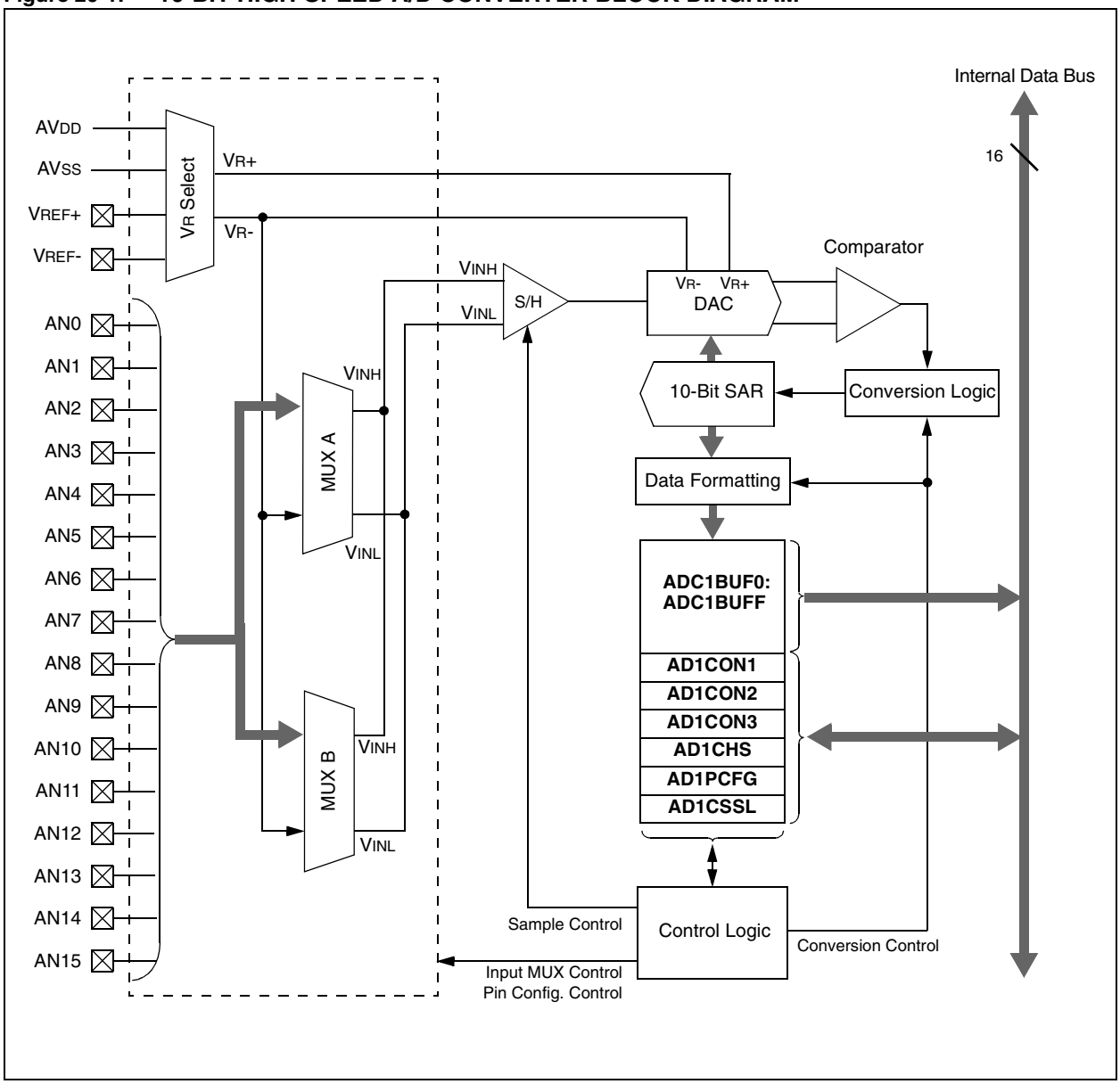
Depending on the particular device pinout, the 10-bit A/D converter can have up to 16 analog input pins, designated AN0 through AN15. In addition, there are two analog input pins for external voltage reference connections. These voltage reference inputs may be shared with other analog input pins. The actual number of analog input pins and external voltage reference input configuration will depend on the specific device. Refer to the device data sheet for further details.

A block diagram of the A/D converter is shown in Figure 20-1.

To perform an A/D conversion:

1. Configure the A/D module:
   a) Select port pins as analog inputs (AD1PCFG<15:0>).
   b) Select voltage reference source to match expected range on analog inputs (AD1CON2<15:13>).
   c) Select the analog conversion clock to match desired data rate with processor clock (AD1CON3<7:0>).
   d) Select the appropriate sample/conversion sequence (AD1CON1<7:0> and AD1CON3<12:8>).
   e) Select how conversion results are presented in the buffer (AD1CON1<9:8>).
   f) Select interrupt rate (AD1CON2<5:2>).
   g) Turn on A/D module (AD1CON1<15>).
2. Configure A/D interrupt (if required):
   a) Clear the AD1IF bit.
   b) Select A/D interrupt priority.

Figure 20-1: 10-BIT HIGH-SPEED A/D CONVERTER BLOCK DIAGRAM



REGISTER 20-1: AD1CON1: A/D CONTROL REGISTER 1

Upper Byte:							
R/W-0	U-0	R/W-0	U-0	U-0	U-0	R/W-0	R/W-0
ADON	—	ADSIDL	—	—	—	FORM1	FORM0
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0 HCS	R/C-0 HCS
SSRC2	SSRC1	SSRC0	—	—	ASAM	SAMP	DONE
bit 7							bit 0

bit 15 **ADON:** A/D Operating Mode bit
1 = A/D converter module is operating
0 = A/D converter is off

bit 14 **Unimplemented:** Read as '0'

bit 13 **ADSIDL:** Stop in Idle Mode bit
1 = Discontinue module operation when device enters Idle mode
0 = Continue module operation in Idle mode

bit 12-10 **Unimplemented:** Read as '0'

bit 9-8 **FORM1:FORM0:** Data Output Format bits
11 = Signed fractional (sddd dddd dd00 0000)
10 = Fractional (dddd dddd dd00 0000)
01 = Signed integer (ssss sssd dddd dddd)
00 = Integer (0000 00dd dddd dddd)

bit 7-5 **SSRC2:SSRC0:** Conversion Trigger Source Select bits
111 = Internal counter ends sampling and starts conversion (auto-convert)
110 = Reserved
10x = Reserved
011 = Reserved
010 = Timer3 compare ends sampling and starts conversion
001 = Active transition on INT0 pin ends sampling and starts conversion
000 = Clearing SAMP bit ends sampling and starts conversion

bit 4-3 **Unimplemented:** Read as '0'

bit 2 **ASAM:** A/D Sample Auto-Start bit
1 = Sampling begins immediately after last conversion completes. SAMP bit is auto-set.
0 = Sampling begins when SAMP bit is set

bit 1 **SAMP:** A/D Sample Enable bit
1 = A/D sample/hold amplifier is sampling input
0 = A/D sample/hold amplifier is holding

bit 0 **DONE:** A/D Conversion Status bit
1 = A/D conversion is done
0 = A/D conversion is NOT done

Legend:	U = Unimplemented bit, read as '0'		
R = Readable bit	W = Writable bit	C = Clear-Only bit	HCS = Hardware Cleared/Set
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 20-2: AD1CON2: A/D CONTROL REGISTER 2

Upper Byte:							
R/W-0	R/W-0	R/W-0	U-0	U-0	R/W-0	U-0	U-0
VCFG2	VCFG1	VCFG0	r	—	CSCNA	—	—
bit 15							bit 8

Lower Byte:							
R-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
BUFS	—	SMPI3	SMPI2	SMPI1	SMPI0	BUFM	ALTS
bit 7							bit 0

bit 15-13 **VCFG2:VCFG0:** Voltage Reference Configuration bits:

VCFG2:VCFG0	VR+	VR-
000	AVDD	AVSS
001	External VREF+ pin	AVSS
010	AVDD	External VREF- pin
011	External VREF+ pin	External VREF- pin
1xx	AVDD	AVSS

bit 12 **Reserved:** User should write '0' to this location

bit 11 **Unimplemented:** Read as '0'

bit 10 **CSCNA:** Scan Input Selections for CH0+ S/H Input for MUX A Input Multiplexer Setting bit
1 = Scan inputs
0 = Do not scan inputs

bit 9-8 **Unimplemented:** Read as '0'

bit 7 **BUFS:** Buffer Fill Status bit (Valid only when BUFM = 1)
1 = A/D is currently filling buffer 08-0F, user should access data in 00-07
0 = A/D is currently filling buffer 00-07, user should access data in 08-0F

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

bit 5-2 **SMPI3:SMPI0:** Sample/Convert Sequences Per Interrupt Selection bits
1111 = Interrupts at the completion of conversion for each 16th sample/convert sequence
1110 = Interrupts at the completion of conversion for each 15th sample/convert sequence
.....
0001 = Interrupts at the completion of conversion for each 2nd sample/convert sequence
0000 = Interrupts at the completion of conversion for each sample/convert sequence

bit 1 **BUFM:** Buffer Mode Select bit
1 = Buffer configured as two 8-word buffers (ADC1BUFn<15:8> and ADC1BUFn<7:0>)
0 = Buffer configured as one 16-word buffer (ADC1BUFn<15:0>)

bit 0 **ALTS:** Alternate Input Sample Mode Select bit
1 = Uses MUX A input multiplexer settings for first sample, then alternates between MUX B and MUX A input multiplexer settings for all subsequent samples
0 = Always use MUX A input multiplexer settings

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

REGISTER 20-3: AD1CON3: A/D CONTROL REGISTER 3

Upper Byte:							
R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADRC	—	—	SAMC4	SAMC3	SAMC2	SAMC1	SAMC0
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADCS7	ADCS6	ADCS5	ADCS4	ADCS3	ADCS2	ADCS1	ADCS0
bit 7							bit 0

bit 15 **ADRC:** A/D Conversion Clock Source bit
1 = A/D internal RC clock
0 = Clock derived from system clock

bit 14-13 **Unimplemented:** Read as '0'

bit 12-8 **SAMC4:SAMC0:** Auto-Sample Time bits
11111 = 31 TAD
.....
00001 = 1 TAD
00000 = 0 TAD (not recommended)

bit 7-0 **ADCS7:ADCS0:** A/D Conversion Clock Select bits
11111111 = 128 • TCY
.....
00000001 = TCY
00000000 = TCY/2

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

REGISTER 20-4: AD1CHS: A/D INPUT SELECT REGISTER

Upper Byte:							
R/W-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
CH0NB	—	—	—	CH0SB3	CH0SB2	CH0SB1	CH0SB0
bit 15							bit 8

Lower Byte:							
R/W-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
CH0NA	—	—	—	CH0SA3	CH0SA2	CH0SA1	CH0SA0
bit 7							bit 0

bit 15 **CH0NB:** Channel 0 Negative Input Select for MUX B Multiplexer Setting bit
1 = Channel 0 negative input is AN1
0 = Channel 0 negative input is VR-

bit 14-12 **Unimplemented:** Read as '0'

bit 11-8 **CH0SB3:CH0SB0:** Channel 0 Positive Input Select for MUX B Multiplexer Setting bits
1111 = Channel 0 positive input is AN15
1110 = Channel 0 positive input is AN14
.....
0001 = Channel 0 positive input is AN1
0000 = Channel 0 positive input is AN0

bit 7 **CH0NA:** Channel 0 Negative Input Select for MUX A Multiplexer Setting bit
1 = Channel 0 negative input is AN1
0 = Channel 0 negative input is VR-

bit 6-4 **Unimplemented:** Read as '0'

bit 3-0 **CH0SA3:CH0SA0:** Channel 0 Positive Input Select for MUX A Multiplexer Setting bits
1111 = Channel 0 positive input is AN15
1110 = Channel 0 positive input is AN14
.....
0001 = Channel 0 positive input is AN1
0000 = Channel 0 positive input is AN0

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

REGISTER 20-5: AD1PCFG: A/D PORT CONFIGURATION REGISTER

Upper Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PCFG15	PCFG14	PCFG13	PCFG12	PCFG11	PCFG10	PCFG9	PCFG8
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PCFG7	PCFG6	PCFG5	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

bit 15-0 **PCFG15:PCFG0:** Analog Input Pin Configuration Control bits

1 = Pin for corresponding analog channel is configured in Digital mode; I/O port read enabled

0 = Pin configured in Analog mode; I/O port read disabled, A/D samples pin voltage

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

REGISTER 20-6: AD1CSSL: A/D INPUT SCAN SELECT REGISTER

Upper Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CSSL15	CSSL14	CSSL13	CSSL12	CSS1L1	CSSL10	CSSL9	CSSL8
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CSSL7	CSSL6	CSSL5	CSSL4	CSSL3	CSSL2	CSSL1	CSSL0
bit 7							bit 0

bit 15-0 **CSSL15:CSSL0:** A/D Input Pin Scan Selection bits

1 = Corresponding analog channel selected for input scan

0 = Analog channel omitted from input scan

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

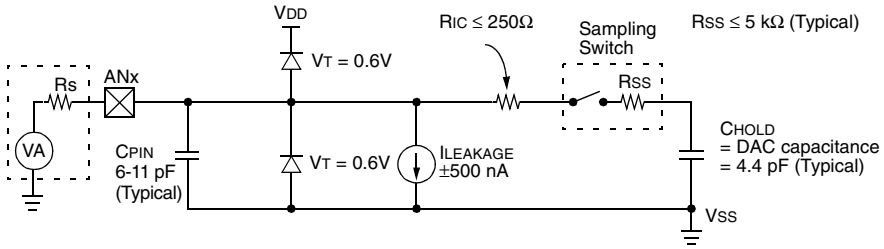
EQUATION 20-1: A/D CONVERSION CLOCK PERIOD(1)

$$T_{AD} = T_{CY}(ADCS + 1)$$

$$ADCS = \frac{T_{AD}}{T_{CY}} - 1$$

Note 1: Based on $T_{CY} = F_{OSC}/2$, Doze mode and PLL are disabled.

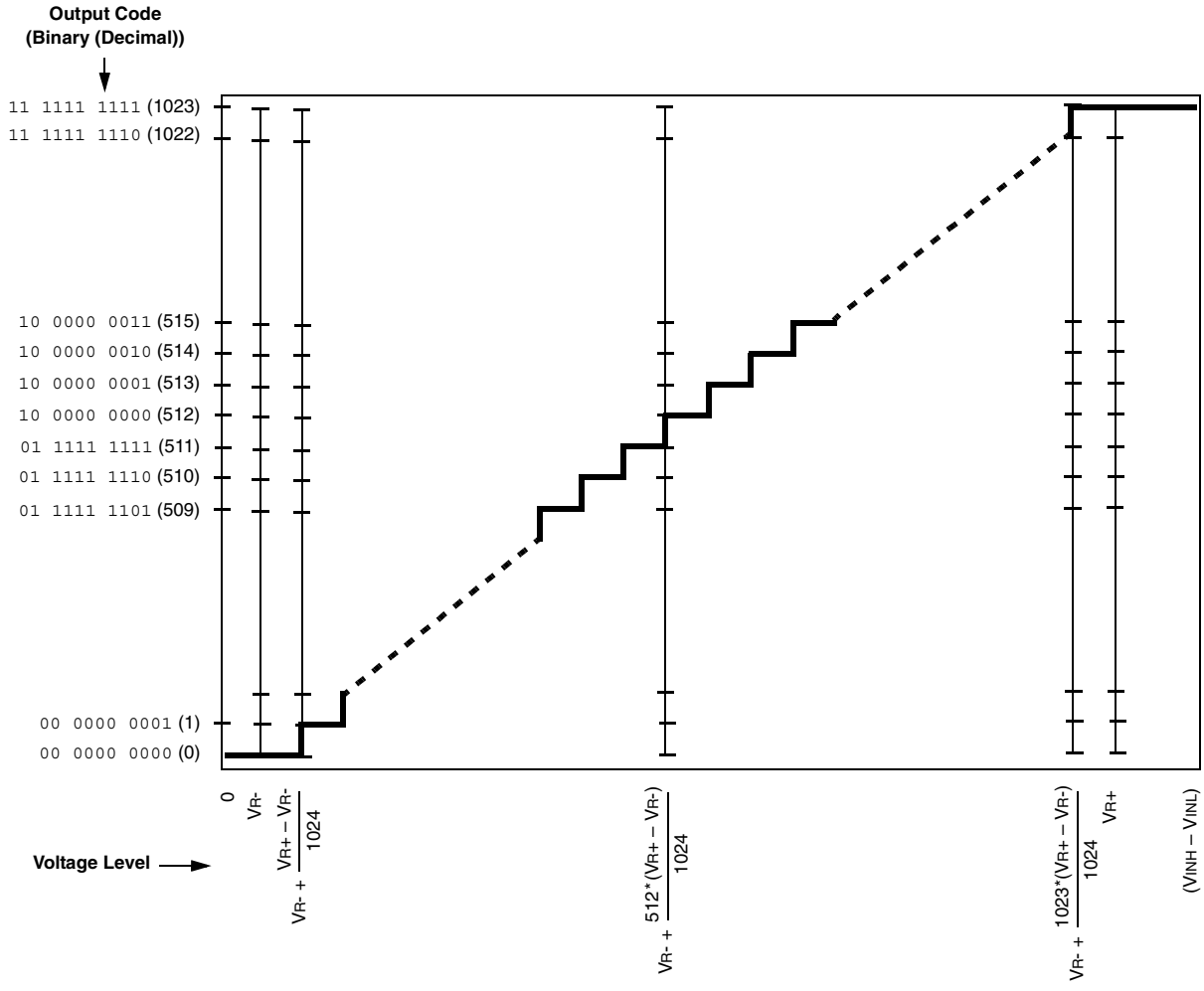
FIGURE 20-2: 10-BIT A/D CONVERTER ANALOG INPUT MODEL



Legend:		
	CPIN	= Input Capacitance
	VT	= Threshold Voltage
	ILEAKAGE	= Leakage Current at the pin due to various junctions
	RIC	= Interconnect Resistance
	RSS	= Sampling Switch Resistance
	CHOLD	= Sample/Hold Capacitance (from DAC)

Note: CPIN value depends on device package and is not tested. Effect of CPIN negligible if Rs ≤ 5 kΩ.

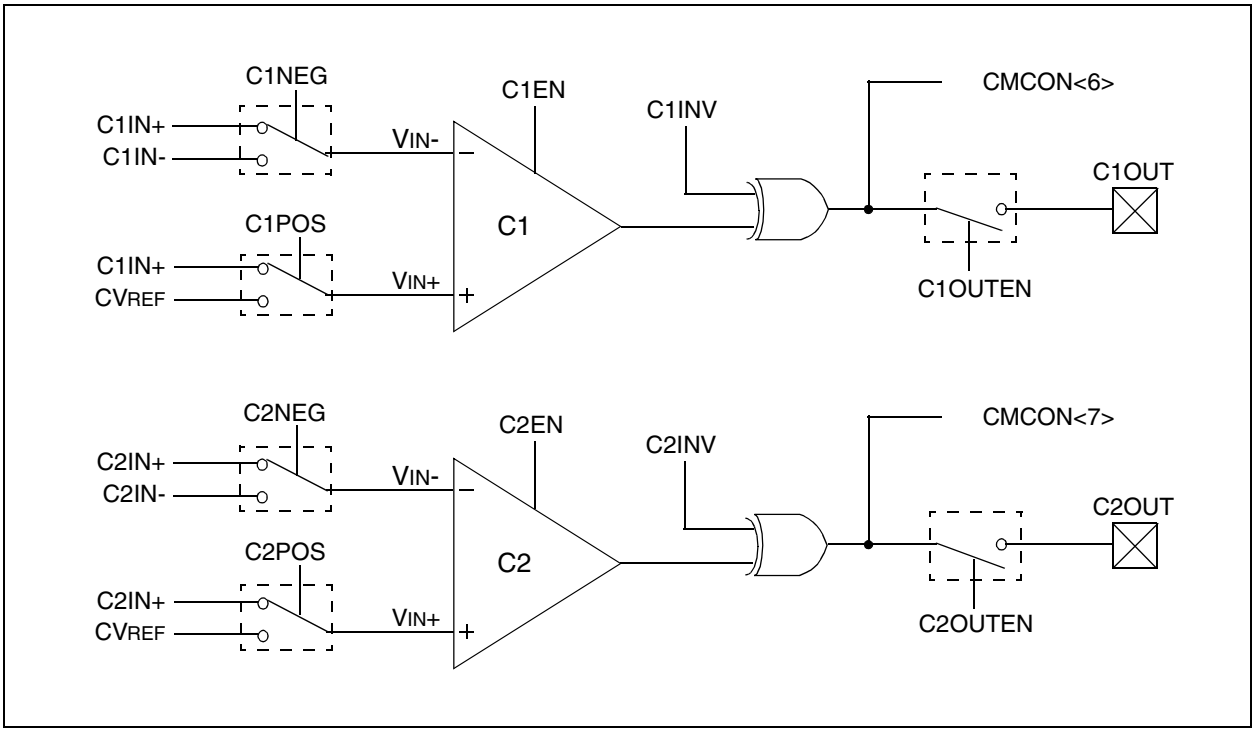
FIGURE 20-3: A/D TRANSFER FUNCTION



21.0 COMPARATOR MODULE

Note:	This data sheet summarizes the features of this group of PIC24FJ devices. It is not intended to be a comprehensive reference source.

FIGURE 21-1: COMPARATOR I/O OPERATING MODES



REGISTER 21-1: CMCON: COMPARATOR CONTROL REGISTER

Upper Byte:							
R/W-0	U-0	R/C-0	R/C-0	R/W-0	R/W-0	R/W-0	R/W-0
CMIDL	—	C2EVT	C1EVT	C2EN	C1EN	C2OUTEN	C1OUTEN
bit 15							bit 8

Lower Byte:							
R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
C2OUT	C1OUT	C2INV	C1INV	C2NEG	C2POS	C1NEG	C1POS
bit 7							bit 0

bit 15 **CMIDL:** Stop in Idle Mode
1 = When device enters Idle mode, module does not generate interrupts. Module is still enabled.
0 = Continue normal module operation in Idle mode

bit 14 **Unimplemented:** Read as '0'

bit 13 **C2EVT:** Comparator 2 Event
1 = Comparator output changed states
0 = Comparator output did not change states

bit 12 **C1EVT:** Comparator 1 Event
1 = Comparator output changed states
0 = Comparator output did not change states

bit 11 **C2EN:** Comparator 2 Enable
1 = Comparator is enabled
0 = Comparator is disabled

bit 10 **C1EN:** Comparator 1 Enable
1 = Comparator is enabled
0 = Comparator is disabled

bit 9 **C2OUTEN:** Comparator 2 Output Enable
1 = Comparator output is driven on the output pad
0 = Comparator output is not driven on the output pad

bit 8 **C1OUTEN:** Comparator 1 Output Enable
1 = Comparator output is driven on the output pad
0 = Comparator output is not driven on the output pad

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

REGISTER 21-1: CMCON: COMPARATOR CONTROL REGISTER (CONTINUED)

Upper Byte:							
R/W-0	U-0	R/C-0	R/C-0	R/W-0	R/W-0	R/W-0	R/W-0
CMIDL	—	C2EVT	C1EVT	C2EN	C1EN	C2OUTEN	C1OUTEN
bit 15							bit 8

Lower Byte:							
R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
C2OUT	C1OUT	C2INV	C1INV	C2NEG	C2POS	C1NEG	C1POS
bit 7							bit 0

bit 7 **C2OUT:** Comparator 2 Output bit

When C2INV = 0:
1 = C2 VIN+ > C2 VIN-
0 = C2 VIN+ < C2 VIN-

When C2INV = 1:
0 = C2 VIN+ > C2 VIN-
1 = C2 VIN+ < C2 VIN-

bit 6 **C1OUT:** Comparator 1 Output bit

When C1INV = 0:
1 = C1 VIN+ > C1 VIN-
0 = C1 VIN+ < C1 VIN-

When C1INV = 1:
0 = C1 VIN+ > C1 VIN-
1 = C1 VIN+ < C1 VIN-

bit 5 **C2INV:** Comparator 2 Output Inversion bit

1 = C2 output inverted
0 = C2 output not inverted

bit 4 **C1INV:** Comparator 1 Output Inversion bit

1 = C1 output inverted
0 = C1 output not inverted

bit 3 **C2NEG:** Comparator 2 Negative Input Configure bit

1 = Input is connected to VIN+
0 = Input is connected to VIN-
See Figure 21-1 for the comparator modes.

bit 2 **C2POS:** Comparator 2 Positive Input Configure bit

1 = Input is connected to VIN+
0 = Input is connected to CVREF
See Figure 21-1 for the comparator modes.

bit 1 **C1NEG:** Comparator 1 Negative Input Configure bit

1 = Input is connected to VIN+
0 = Input is connected to VIN-
See Figure 21-1 for the comparator modes.

bit 0 **C1POS:** Comparator 1 Positive Input Configure bit

1 = Input is connected to VIN+
0 = Input is connected to CVREF
See Figure 21-1 for the comparator modes.

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

NOTES:

22.0 COMPARATOR VOLTAGE REFERENCE

Note:	This data sheet summarizes the features of this group of PIC24FJ devices. It is not intended to be a comprehensive reference source.

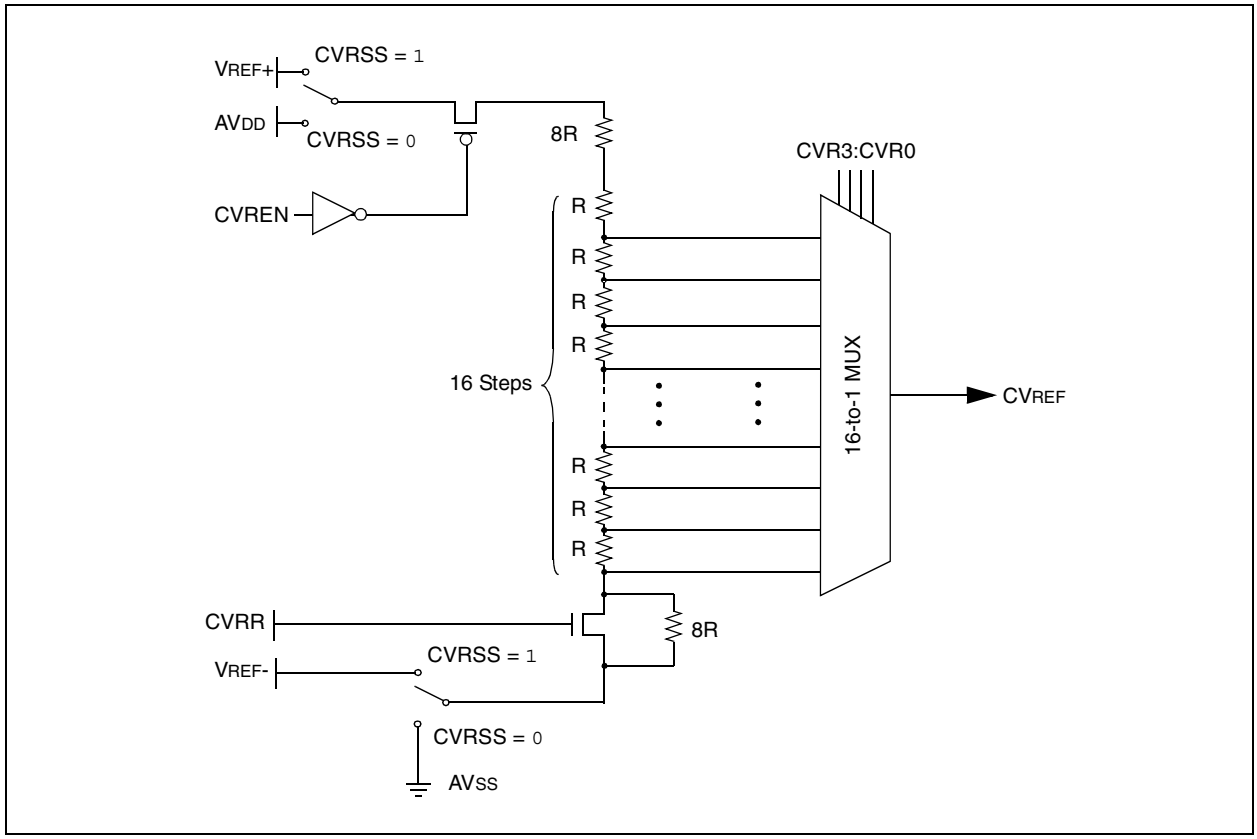
22.1 Configuring the Comparator Voltage Reference

The voltage reference module is controlled through the CVRCON register (Register 22-1). The comparator voltage reference provides two ranges of output voltage, each with 16 distinct levels. The range to be used is selected by the CVRR bit (CVRCON<5>). The primary difference between the ranges is the size of the steps selected by the CVREF Selection bits (CVR3:CVR0), with one range offering finer resolution.

The comparator reference supply voltage can come from either VDD and VSS, or the external VREF+ and VREF-. The voltage source is selected by the CVRSS bit (CVRCON<4>).

The settling time of the comparator voltage reference must be considered when changing the CVREF output.

FIGURE 22-1: COMPARATOR VOLTAGE REFERENCE BLOCK DIAGRAM



REGISTER 22-1: CVRCON: COMPARATOR VOLTAGE REFERENCE CONTROL REGISTER

Upper Byte:							
U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0
bit 7							bit 0

bit 15-8 **Unimplemented:** Read as '0'

bit 7 **CVREN:** Comparator Voltage Reference Enable bit
1 = CVREF circuit powered on
0 = CVREF circuit powered down

bit 6 **CVROE:** Comparator VREF Output Enable bit
1 = CVREF voltage level is output on CVREF pin
0 = CVREF voltage level is disconnected from CVREF pin

bit 5 **CVRR:** Comparator VREF Range Selection bit
1 = 0 to 0.67 CVRSRC, with CVRSRC/24 step size
0 = 0.25 CVRSRC to 0.75 CVRSRC, with CVRSRC/32 step size

bit 4 **CVRSS:** Comparator VREF Source Selection bit
1 = Comparator reference source CVRSRC = VREF+ – VREF-
0 = Comparator reference source CVRSRC = AVDD – AVSS

bit 3-0 **CVR3:CVR0:** Comparator VREF Value Selection 0 ≤ CVR3:CVR0 ≤ 15 bits
When CVRR = 1:
CVREF = (CVR<3:0>/ 24) • (CVRSRC)
When CVRR = 0:
CVREF = 1/4 • (CVRSRC) + (CVR<3:0>/32) • (CVRSRC)

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

23.0 SPECIAL FEATURES

Note:	This data sheet summarizes the features of this group of PIC24FJ devices. It is not intended to be a comprehensive reference source.

PIC24FJ128GA family devices include several features intended to maximize application flexibility and reliability, and minimize cost through elimination of external components. These are:

- Flexible Configuration
- Watchdog Timer (WDT)
- Code Protection
- JTAG Boundary Scan Interface
- In-Circuit Serial Programming
- In-Circuit Emulation

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 F80000h. A complete list is shown in Table 23-1. A detailed explanation of the various bit functions is provided in Register 23-1 through Register 23-4.

Note that address F80000h is beyond the user program memory space. In fact, it belongs to the configuration memory space (800000h-FFFFFFh) which can only be accessed using table reads and table writes.

23.1.1 CONSIDERATIONS FOR CONFIGURING PIC24FJ128GA FAMILY DEVICES

In PIC24FJ128GA family devices, the configuration bytes are implemented as volatile memory. This means that configuration data must be programmed each time the device is powered up. Configuration data is stored in the two words at the top of the on-chip program memory space, known as the Flash Configuration Words. Their specific locations are shown in Table 23-1. These are packed representations of the actual device Configuration bits, whose actual locations are distributed among five locations in configuration space. The configuration data is automatically loaded from the Flash Configuration Words to the proper Configuration registers during device Resets.

TABLE 23-1: FLASH CONFIGURATION WORDS LOCATIONS FOR PIC24FJ128GA FAMILY DEVICES

Device	Configuration Word Addresses	
	1	2
PIC24FJ64GA	00ABFEh	00ABFCh
PIC24FJ96GA	00FFFEh	00FFFCh
PIC24FJ128GA	0157FEh	0157FCh

When creating applications for these devices, users should always specifically allocate the location of the Flash Configuration Word for configuration data. This is to make certain that program code is not stored in this address when the code is compiled.

The volatile memory cells used for the Configuration bits always reset to '1' on Power-on Resets. For all other type of Reset events, the previously programmed values are maintained and used without reloading from program memory.

The upper byte of both Flash Configuration Words in program memory should always be '1111 1111'. This makes them appear to be NOP instructions in the remote event that their locations are ever executed by accident. Since Configuration bits are not implemented in the corresponding locations, writing '1's to these locations has no effect on device operation.

To prevent inadvertent configuration changes during code execution, all programmable Configuration bits are write-once. After a bit is initially programmed during a power cycle, it cannot be written to again. Changing a device configuration requires that power to the device be cycled.

REGISTER 23-1: FLASH CONFIGURATION WORD 1

Upper Third:							
U-1	U-1	U-1	U-1	U-1	U-1	U-1	U-1
—	—	—	—	—	—	—	—
bit 23							bit 16

Middle Third:							
r-0	R/PO-1	R/PO-1	R/PO-1	R/PO-1	R/PO-1	U-1	R/PO-1
r	JTAGEN	GCP	GWRP	$\overline{\text{DEBUG}}$	COE	—	ICS
bit 15							bit 8

Lower Third:							
R/PO-1	R/PO-1	U-1	R/PO-1	R/PO-1	R/PO-1	R/PO-1	R/PO-1
FWDTEN	WINDIS	—	FWPSA	WDTPS3	WDTPS2	WDTPS1	WDTPS0
bit 7							bit 0

bit 23-16 **Unimplemented:** Read as '0'

bit 15 **Reserved:** Maintain as '1'

bit 14 **JTAGEN:** JTAG Port Enable bit
1 = JTAG port is enabled
0 = JTAG port is disabled

bit 13 **GCP:** General Segment Program Memory Code Protection bit
1 = Code protection is disabled
0 = Code protection is enabled for the entire program memory space

bit 12 **GWRP:** General Segment Code Flash Write Protection bit
1 = Writes to program memory are allowed
0 = Writes to program memory are disabled

bit 11 **$\overline{\text{DEBUG}}$:** Background Debugger Enable bit
1 = Device resets into Operational mode
0 = Device resets into Debug mode

bit 10 **COE:** Set Clip On Emulation bit
1 = Device resets into Operational mode
0 = Device resets into Clip On Emulation mode

bit 9 **Unimplemented:** Read as '1'

bit 8 **ICS: ICD Pin Placement Select bit**
1 = ICD uses EMUC2/EMUD2
0 = ICD uses EMUC1/EMUD1

bit 7 **FWDTEN:** Watchdog Timer Enable bit
1 = Watchdog Timer is enabled
0 = Watchdog Timer is disabled

bit 6 **WINDIS:** Windowed Watchdog Timer Disable bit
1 = Standard Watchdog Timer enabled
0 = Windowed Watchdog Timer enabled; FWDTEN must be '1'

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

Legend:			
R = Readable bit	PO = Program-Once bit	U = Unimplemented, read as '1'	
-n = Value when unprogrammed	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 23-1: FLASH CONFIGURATION WORD 1 (CONTINUED)

Upper Third:							
U-1	U-1	U-1	U-1	U-1	U-1	U-1	U-1
—	—	—	—	—	—	—	—
bit 23							bit 16

Middle Third:							
r-0	R/PO-1	R/PO-1	R/PO-1	R/PO-1	R/PO-1	U-1	R/PO-1
r	JTAGEN	GCP	GWRP	$\overline{\text{DEBUG}}$	COE	—	ICS
bit 15							bit 8

Lower Third:							
R/PO-1	R/PO-1	U-1	R/PO-1	R/PO-1	R/PO-1	R/PO-1	R/PO-1
FWDTEN	WINDIS	—	FWPSA	WDTPS3	WDTPS2	WDTPS1	WDTPS0
bit 7							bit 0

bit 4 **FWPSA:** WDT Prescaler Ratio Select bit
1 = Prescaler ratio of 1:128
0 = Prescaler ratio of 1:32

bit 3-0 **WDTPS3:WDTPS0:** Watchdog Timer Postscaler 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

Legend:			
R = Readable bit	PO = Program-Once bit	U = Unimplemented, read as '1'	
-n = Value when unprogrammed	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 23-2: FLASH CONFIGURATION WORD 2

Upper Third:							
U-1	U-1	U-1	U-1	U-1	U-1	U-1	U-1
—	—	—	—	—	—	—	—
bit 23							bit 16

Middle Third:							
R/PO-1	U-1	U-1	U-1	U-1	R/PO-1	R/PO-1	R/PO-1
IESO	—	—	—	—	FNOSC2	FNOSC1	FNOSC0
bit 15							bit 8

Lower Third:							
R/PO-1	R/PO-1	R/PO-1	U-1	U-1	U-1	R/PO-1	R/PO-1
FCKSM1	FCKSM0	OSCIOFCN	—	—	—	POSCMD1	POSCMD0
bit 7							bit 0

bit 23-16 **Unimplemented:** Read as '1'

bit 15 **IESO:** Internal External Switchover bit
1 = IESO mode (Two-Speed Start-up) enabled
0 = IESO mode (Two-Speed Start-up) disabled

bit 14-11 **Unimplemented:** Read as '1'

bit 10-8 **FNOSC2:FNOSC0:** Initial Oscillator Select bits
111 = Fast RC Oscillator with Postscaler (FRCDIV)
110 = Reserved
101 = Low-Power RC Oscillator (LPRC)
100 = Secondary Oscillator (SOSC)
011 = Primary Oscillator with PLL module (HSPLL, ECPLL)
010 = Primary Oscillator (XT, HS, EC)
001 = Fast RC Oscillator with postscaler and PLL module (FRCPLL)
000 = Fast RC Oscillator (FRC)

bit 7-6 **FCKSM1:FCKSM0:** Clock Switching and Fail-Safe Clock Monitor Configuration bits
1x = Clock switching and Fail-Safe Clock Monitor are disabled
01 = Clock switching is enabled, Fail-Safe Clock Monitor is disabled
00 = Clock switching is enabled, Fail-Safe Clock Monitor is enabled

bit 5 **OSCIOFCN:** OSC2 Pin Configuration bit
If POSCMD1:POSCMD0 = 11 or 00:
1 = OSC2/CLKO/RC15 functions as CLKO (FOSC/2)
0 = OSC2/CLKO/RC15 functions as port I/O (RC15)
If POSCMD1:POSCMD0 = 10 or 01:
OSCIOFCN has no effect on OSC2/CLKO/RC15.

bit 4-2 **Unimplemented:** Read as '1'

bit 1-0 **POSCMD1:POSCMD0:** Primary Oscillator Configuration bits
11 = Primary oscillator disabled
10 = HS Oscillator mode selected
01 = XT Oscillator mode selected
00 = EC Oscillator mode selected

.

Legend:			
R = Readable bit	PO = Program-Once bit	U = Unimplemented bit, read as '1'	
-n = Value when unprogrammed	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 23-3: DEVID: DEVICE ID REGISTER

Upper Third:							
U	U	U	U	U	U	U	U
—	—	—	—	—	—	—	—
bit 23							bit 16

Middle Third:							
U	U	R	R	R	R	R	R
—	—	FAMID7	FAMID6	FAMID5	FAMID4	FAMID3	FAMID2
bit 15							bit 8

Lower Third:							
R	R	R	R	R	R	R	R
FAMID1	FAMID0	DEV5	DEV4	DEV3	DEV2	DEV1	DEV0
bit 7							bit 0

bit 23-14 **Unimplemented:** Read as '0'

bit 13-6 **FAMID7:FAMID0:** Device Family Identifier bits

00010000 = PIC24FJ128GA family

bit 5-0 **DEV5:DEV0:** Individual Device Identifier bits

000101 = PIC24FJ64GA006
000110 = PIC24FJ96GA006
000111 = PIC24FJ128GA006
001000 = PIC24FJ64GA008
001001 = PIC24FJ96GA008
001010 = PIC24FJ128GA008
001011 = PIC24FJ64GA010
001100 = PIC24FJ96GA010
001101 = PIC24FJ128GA010

Legend:	
R = Readable bit	U = Unimplemented bit, read as '0'

REGISTER 23-4: DEVREV: DEVICE REVISION REGISTER

Upper Third:

U	U	U	U	U	U	U	U
—	—	—	—	—	—	—	—

bit 23 bit 16

Middle Third:

R	R	R	R	U	U	U	R
r	r	r	r	—	—	—	MAJRV2

bit 15 bit 8

Lower Third:

R	R	U	U	U	R	R	R
MAJRV1	MAJRV0	—	—	—	DOT2	DOT1	DOT0

bit 7 bit 0

bit 23-16 **Unimplemented:** Read as '0'

bit 15-12 **Reserbit 8**

R R R

23.2 On-Chip Voltage Regulator

All of the PIC24FJ128GA family devices power their core digital logic at a nominal 2.5V. This may create an issue for designs that are required to operate at a higher typical voltage, such as 3.3V. To simplify system design, all devices in the PIC24FJ128GA family incorporate an on-chip regulator that allows the device to run its core logic from VDD.

The regulator is controlled by the ENVREG pin. Tying VDD to the pin enables the regulator, which in turn, provides power to the core from the other VDD pins. When the regulator is enabled, a low ESR capacitor (such as tantalum) must be connected to the VDDCORE/VCAP pin (Figure 23-1). This helps to maintain the stability of the regulator. The recommended value for the filer capacitor is provided in **Section 26.1 "DC Characteristics"**.

If ENVREG is tied to VSS, the regulator is disabled. In this case, separate power for the core logic at a nominal 2.5V must be supplied to the device on the VDDCORE/VCAP pin to run the I/O pins at higher voltage levels, typically 3.3V. Alternatively, the VDDCORE/VCAP and VDD pins can be tied together to operate at a lower nominal voltage. Refer to Figure 23-1 for possible configurations.

23.2.1 ON-CHIP REGULATOR AND POR

When the voltage regulator is enabled, it takes approximately 20 μs for it to generate output. During this time, designated as TSTARTUP, code execution is disabled. TSTARTUP is applied every time the device resumes operation after any power-down, including Sleep mode.

If the regulator is disabled, a separate Power-up Timer (PWRT) is automatically enabled. The PWRT adds a fixed delay of 64 ms nominal delay at device start-up.

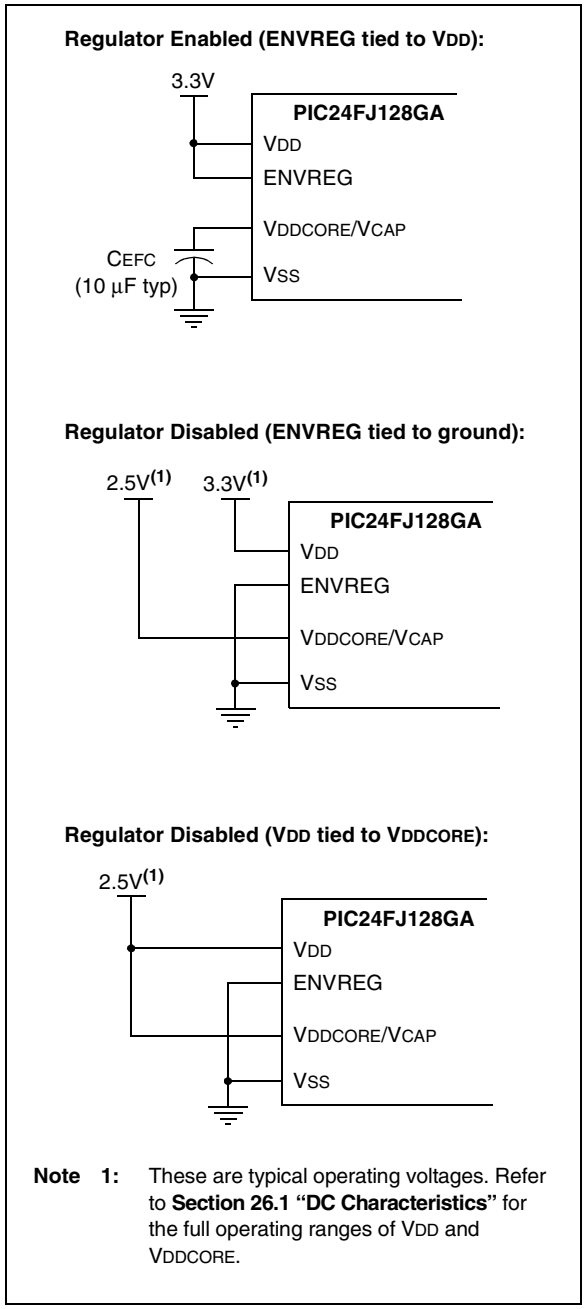
23.2.2 ON-CHIP REGULATOR AND BOR

When the on-chip regulator is enabled, PIC24FJ128GA family devices also have a simple brown-out capability. If the voltage supplied to the regulator is inadequate to maintain a regulated level, the regulator Reset circuitry will generate a Brown-out Reset. This event is captured by the $\overline{BOR}$ flag bit (RCON<0>). The brown-out voltage levels are specific in **Section 26.1 "DC Characteristics"**.

23.2.3 POWER-UP REQUIREMENTS

The on-chip regulator is designed to meet the power-up requirements for the device. If the application does not use the regulator, then strict power-up conditions must be adhered to. While powering up, VDDCORE must never exceed VDD by 0.3 volts.

FIGURE 23-1: CONNECTIONS FOR THE ON-CHIP REGULATOR



Note 1: These are typical operating voltages. Refer to **Section 26.1 "DC Characteristics"** for the full operating ranges of VDD and VDDCORE.

23.3 Watchdog Timer (WDT)

For PIC24FJ128GA family devices, the WDT is driven by the LPRC oscillator. When the WDT is enabled, the clock source is also enabled.

The nominal WDT clock source from LPRC is 32 kHz. This feeds a prescaler that can be configured for either 5-bit (divide-by-32) or 7-bit (divide-by-128) operation. The prescaler is set by the FWPSA Configuration bit. With a 32 kHz input, the prescaler yields a nominal WDT time-out period (TWDT) of 1 ms in 5-bit mode, or 4 ms in 7-bit mode.

A variable postscaler divides down the WDT prescaler output and allows for a wide range of time-out periods. The postscaler is controlled by the WDTPS3:WDTPS0 Configuration bits (Flash Configuration Word 1<3:0>), which allow the selection of a total of 16 settings, from 1:1 to 1:32,768. Using the prescaler and postscaler, time-out periods ranging from 1 ms to 131 seconds can be achieved.

The WDT, prescaler and postscaler are reset:

- On any device Reset
- On the completion of a clock switch, whether invoked by software (i.e., setting the OSWEN bit after changing the NOSC bits), or by hardware (i.e., Fail-Safe Clock Monitor)
- When a PWRSAV instruction is executed (i.e., Sleep or Idle mode is entered)
- When the device exits Sleep or Idle mode to resume normal operation
- By a CLRWDT instruction during normal execution

If the WDT is enabled, it will continue to run during Sleep or Idle modes. When the WDT time-out occurs, the device will wake the device and code execution will continue from where the PWRSAV instruction was executed. The corresponding SLEEP or IDLE bits (RCON<3:2>) will need to be cleared in software after the device wakes up.

The WDT Flag bit, WDTO (RCON<4>), is not automatically cleared following a WDT time-out. To detect subsequent WDT events, the flag must be cleared in software.

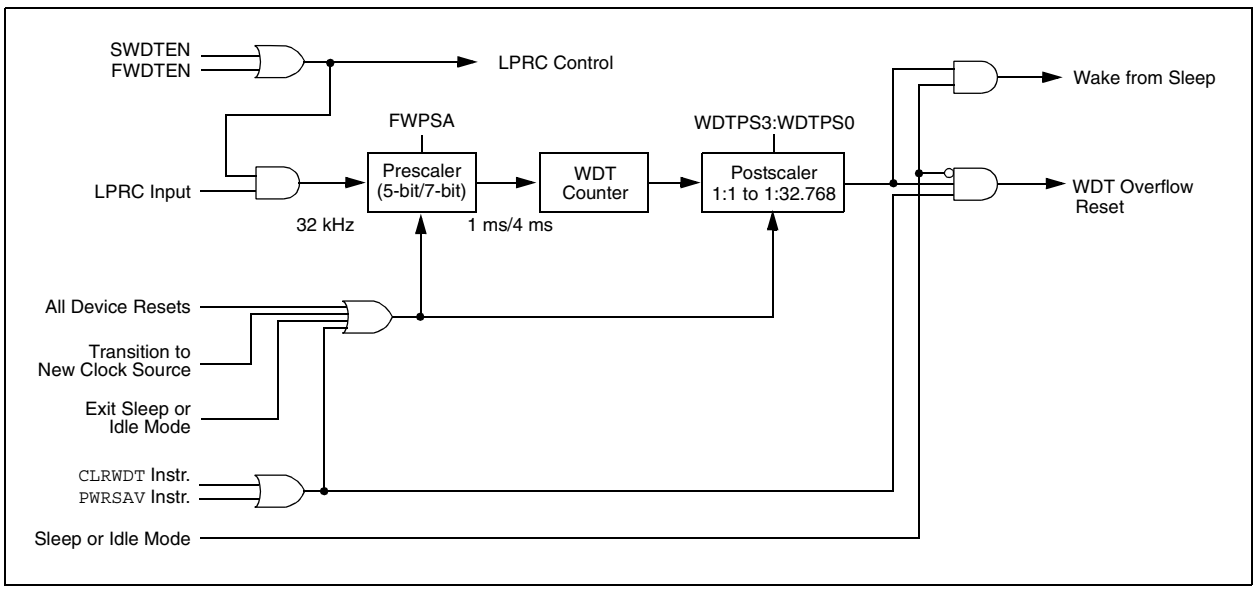
Note:	The CLRWDT and PWRSAV instructions clear the prescaler and postscaler counts when executed.

23.3.1 CONTROL REGISTER

The WDT is enabled or disabled by the FWDTEN device Configuration bit. When the FWDTEN Configuration bit is set, the WDT is always enabled.

The WDT can be optionally controlled in software when the FWDTEN Configuration bit has been programmed to '0'. The WDT is enabled in software by setting the SWDTEN control bit (RCON<5>). The SWDTEN control bit is cleared on any device Reset. The software WDT option allows the user to enable the WDT for critical code segments and disable the WDT during non-critical segments for maximum power savings.

FIGURE 23-2: WDT BLOCK DIAGRAM



23.4 JTAG Interface

PIC24FJ128GA family devices implement a JTAG interface, which supports boundary scan device testing as well as in-circuit programming.

23.5 Program Verification and Code Protection

For all devices in the PIC24FJ128GA family of devices, the on-chip program memory space is treated as a single block. Code protection for this block is controlled by one Configuration bit, GCP. This bit inhibits external reads and writes to the program memory space. It has no direct effect in normal execution mode.

23.5.1 CONFIGURATION REGISTER PROTECTION

The Configuration registers are protected against inadvertent or unwanted changes or reads in two ways. The primary protection is the write-once feature of the Configuration bits which prevents reconfiguration once the bit has been programmed during a power cycle. To safeguard against unpredictable events, Configuration bit changes resulting from individual cell level disruptions (such as ESD events) will cause a parity error and trigger a device Reset.

The data for the Configuration registers is derived from the Flash Configuration Words in program memory. When the GCP bit is set, the source data for device configuration is also protected as a consequence.

23.6 In-Circuit Serial Programming

PIC24FJ128GA family microcontrollers can be serially programmed while in the end application circuit. This is simply done with two lines for clock (PGCx) and data (PGDx) 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.

23.7 In-Circuit Debugger

When MPLAB® ICD 2 is selected as a debugger, the In-Circuit Debugging functionality is enabled. This function allows simple debugging functions when used with MPLAB IDE. Debugging functionality is controlled through the EMUCx (Emulation/Debug Clock) and EMUDx (Emulation/Debug Data) pins.

To use the In-Circuit Debugger function of the device, the design must implement ICSP connections to $\overline{MCLR}$, VDD, VSS, PGCx, PGDx and the EMUDx/EMUCx pin pair. In addition, when the feature is enabled, some of the resources are not available for general use. These resources include the first 80 bytes of data RAM and two I/O pins.

NOTES:

24.0 INSTRUCTION SET SUMMARY

Note:	This chapter is a brief summary of the PIC24 instruction set architecture, and is not intended to be a comprehensive reference source. For detailed information on programming

The PIC24 instruction set adds many enhancements to the previous PICmicro® MCU instruction sets, while maintaining an easy migration from previous PICmicro MCU instruction sets. Most instructions are a single program memory word. Only three instructions require two program memory locations.

Each single-word instruction is a 24-bit word divided into an 8-bit 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:

- Word or byte-oriented operations
- Bit-oriented operations
- Literal operations
- Control operations

Table 24-1 shows the general symbols used in describing the instructions. The PIC24 instruction set summary in Table 24-2 lists all the instructions, along with the status flags affected by each instruction.

Most word or byte-oriented W register instructions (including barrel shift instructions) have three operands:

- The first source operand which is typically a register ‘Wb’ without any address modifier
- The second source operand which is typically a register ‘Ws’ with or without an address modifier
- The destination of the result which is typically a register ‘Wd’ with or without an address modifier

However, word or byte-oriented file register instructions have two operands:

- The file register specified by the value ‘f’
- The destination, which could either be the file register ‘f’ or the W0 register, which is denoted as ‘WREG’

Most bit-oriented instructions (including simple rotate/shift instructions) have two operands:

- The W register (with or without an address modifier) or file register (specified by the value of ‘Ws’ or ‘f’)
- The bit in the W register or file register (specified by a literal value or indirectly by the contents of register ‘Wb’)

The literal instructions that involve data movement may use some of the following operands:

- A literal value to be loaded into a W register or file register (specified by the value of ‘k’)
- The W register or file register where the literal value is to be loaded (specified by ‘Wb’ or ‘f’)

However, literal instructions that involve arithmetic or logical operations use some of the following operands:

- The first source operand which is a register ‘Wb’ without any address modifier
- The second source operand which is a literal value
- The destination of the result (only if not the same as the first source operand) which is typically a register ‘Wd’ with or without an address modifier

The control instructions may use some of the following operands:

- A program memory address
- The mode of the table read and table write instructions

All instructions are a single word, except for certain double-word instructions, which were made double-word instructions so that all the required information is available in these 48 bits. In the second word, the 8 MSbs are ‘0’s. If this second word is executed as an instruction (by itself), it will execute as a NOP.

Most 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. Notable exceptions are the BRA (unconditional/computed branch), indirect CALL/GOTO, all table reads and writes, and RETURN/RETFIE instructions, which are single-word instructions but take two or three cycles.

Certain instructions that involve skipping over the subsequent instruction require either two or three cycles if the skip is performed, depending on whether the instruction being skipped is a single-word or two-word instruction. Moreover, double-word moves require two cycles. The double-word instructions execute in two instruction cycles.

TABLE 24-1: SYMBOLS USED IN OPCODE DESCRIPTIONS

Field	Description
#text	Means literal defined by "text"
(text)	Means "content of text"
[text]	Means "the location addressed by text"
{ }	Optional field or operation
<n:m>	Register bit field
.b	Byte mode selection
.d	Double-Word mode selection
.S	Shadow register select
.w	Word mode selection (default)
bit4	4-bit bit selection field (used in word addressed instructions) ∈ {0...15}
C, DC, N, OV, Z	MCU Status bits: Carry, Digit Carry, Negative, Overflow, Sticky Zero
Expr	Absolute address, label or expression (resolved by the linker)
f	File register address ∈ {0000h...1FFFh}
lit1	1-bit unsigned literal ∈ {0,1}
lit4	4-bit unsigned literal ∈ {0...15}
lit5	5-bit unsigned literal ∈ {0...31}
lit8	8-bit unsigned literal ∈ {0...255}
lit10	10-bit unsigned literal ∈ {0...255} for Byte mode, {0:1023} for Word mode
lit14	14-bit unsigned literal ∈ {0...16384}
lit16	16-bit unsigned literal ∈ {0...65535}
lit23	23-bit unsigned literal ∈ {0...8388608}; LSB must be '0'
None	Field does not require an entry, may be blank
PC	Program Counter
Slit10	10-bit signed literal ∈ {-512...511}
Slit16	16-bit signed literal ∈ {-32768...32767}
Slit6	6-bit signed literal ∈ {-16...16}
Wb	Base W register ∈ {W0..W15}
Wd	Destination W register ∈ { Wd, [Wd], [Wd++], [Wd--], [++Wd], [--Wd] }
Wdo	Destination W register ∈ { Wnd, [Wnd], [Wnd++], [Wnd--], [++Wnd], [--Wnd], [Wnd+Wb] }
Wm,Wn	Dividend, Divisor working register pair (direct addressing)
Wn	One of 16 working registers ∈ {W0..W15}
Wnd	One of 16 destination working registers ∈ {W0..W15}
Wns	One of 16 source working registers ∈ {W0..W15}
WREG	W0 (working register used in file register instructions)
Ws	Source W register ∈ { Ws, [Ws], [Ws++], [Ws--], [++Ws], [--Ws] }
Wso	Source W register ∈ { Wns, [Wns], [Wns++], [Wns--], [++Wns], [--Wns], [Wns+Wb] }

TABLE 24-2: INSTRUCTION SET OVERVIEW

Assembly Mnemonic		Assembly Syntax	Description	# of Words	# of Cycles	Status Flags Affected
ADD	ADD	f	f = f + WREG	1	1	C, DC, N, OV, Z
	ADD	f,WREG	WREG = f + WREG	1	1	C, DC, N, OV, Z
	ADD	#lit10,Wn	Wd = lit10 + Wd	1	1	C, DC, N, OV, Z
	ADD	Wb,Ws,Wd	Wd = Wb + Ws	1	1	C, DC, N, OV, Z
	ADD	Wb,#lit5,Wd	Wd = Wb + lit5	1	1	C, DC, N, OV, Z
ADDC	ADDC	f	f = f + WREG + (C)	1	1	C, DC, N, OV, Z
	ADDC	f,WREG	WREG = f + WREG + (C)	1	1	C, DC, N, OV, Z
	ADDC	#lit10,Wn	Wd = lit10 + Wd + (C)	1	1	C, DC, N, OV, Z
	ADDC	Wb,Ws,Wd	Wd = Wb + Ws + (C)	1	1	C, DC, N, OV, Z
	ADDC	Wb,#lit5,Wd	Wd = Wb + lit5 + (C)	1	1	C, DC, N, OV, Z
AND	AND	f	f = f .AND. WREG	1	1	N, Z
	AND	f,WREG	WREG = f .AND. WREG	1	1	N, Z
	AND	#lit10,Wn	Wd = lit10 .AND. Wd	1	1	N, Z
	AND	Wb,Ws,Wd	Wd = Wb .AND. Ws	1	1	N, Z
	AND	Wb,#lit5,Wd	Wd = Wb .AND. lit5	1	1	N, Z
ASR	ASR	f	f = Arithmetic Right Shift f	1	1	C, N, OV, Z
	ASR	f,WREG	WREG = Arithmetic Right Shift f	1	1	C, N, OV, Z
	ASR	Ws,Wd	Wd = Arithmetic Right Shift Ws	1	1	C, N, OV, Z
	ASR	Wb,Wns,Wnd	Wnd = Arithmetic Right Shift Wb by Wns	1	1	N, Z
	ASR	Wb,#lit5,Wnd	Wnd = Arithmetic Right Shift Wb by lit5	1	1	N, Z
BCLR	BCLR	f,#bit4	Bit Clear f	1	1	None
	BCLR	Ws,#bit4	Bit Clear Ws	1	1	None
BRA	BRA	C,Expr	Branch if Carry	1	1 (2)	None
	BRA	GE,Expr	Branch if Greater than or Equal	1	1 (2)	None
	BRA	GEU,Expr	Branch if Unsigned Greater than or Equal	1	1 (2)	None
	BRA	GT,Expr	Branch if Greater than	1	1 (2)	None
	BRA	GTU,Expr	Branch if Unsigned Greater than	1	1 (2)	None

TABLE 24-2: INSTRUCTION SET OVERVIEW (CONTINUED)

Assembly Mnemonic	Assembly Syntax		Description	# of Words	# of Cycles	Status Flags Affected
BTSS	BTSS	f,#bit4	Bit Test f, Skip if Set	1	1 (2 or 3)	None
	BTSS	Ws,#bit4	Bit Test Ws, Skip if Set	1	1 (2 or 3)	None
BTST	BTST	f,#bit4	Bit Test f	1	1	Z
	BTST.C	Ws,#bit4	Bit Test Ws to C	1	1	C
	BTST.Z	Ws,#bit4	Bit Test Ws to Z	1	1	Z
	BTST.C	Ws,Wb	Bit Test Ws<Wb> to C	1	1	C
	BTST.Z	Ws,Wb	Bit Test Ws<Wb> to Z	1	1	Z
BTSTS	BTSTS	f,#bit4	Bit Test then Set f	1	1	Z
	BTSTS.C	Ws,#bit4	Bit Test Ws to C, then Set	1	1	C
	BTSTS.Z	Ws,#bit4	Bit Test Ws to Z, then Set	1	1	Z
CALL	CALL	lit23	Call Subroutine	2	2	None
	CALL	Wn	Call Indirect Subroutine	1	2	None
CLR	CLR	f	f = 0x0000	1	1	None
	CLR	WREG	WREG = 0x0000	1	1	None
	CLR	Ws	Ws = 0x0000	1	1	None
CLRWDT	CLRWDT		Clear Watchdog Timer	1	1	WDTO, Sleep
COM	COM	f	$f = \overline{f}$	1	1	N, Z
	COM	f,WREG	$WREG = \overline{f}$	1	1	N, Z
	COM	Ws,Wd	$Wd = \overline{Ws}$	1	1	N, Z
CP	CP	f	Compare f with WREG	1	1	C, DC, N, OV, Z
	CP	Wb,#lit5	Compare Wb with lit5	1	1	C, DC, N, OV, Z
	CP	Wb,Ws	Compare Wb with Ws (Wb – Ws)	1	1	C, DC, N, OV, Z
CP0	CP0	f	Compare f with 0x0000	1	1	C, DC, N, OV, Z
	CP0	Ws	Compare Ws with 0x0000	1	1	C, DC, N, OV, Z
CP1	CP1	f	Compare f with 0xFFFF	1	1	C, DC, N, OV, Z
	CP1	Ws	Compare Ws with 0xFFFF	1	1	C, DC, N, OV, Z
CPB	CPB	f	Compare f with WREG, with Borrow	1	1	C, DC, N, OV, Z
	CPB	Wb,#lit5	Compare Wb with lit5, with Borrow	1	1	C, DC, N, OV, Z
	CPB	Wb,Ws	Compare Wb with Ws, with Borrow ($Wb – Ws – \overline{C}$)	1	1	C, DC, N, OV, Z
CPSEQ	CPSEQ	Wb,Wn	Compare Wb with Wn, Skip if =	1	1 (2 or 3)	None
CPSGT	CPSGT	Wb,Wn	Compare Wb with Wn, Skip if >	1	1 (2 or 3)	None
CPSLT	CPSLT	Wb,Wn	Compare Wb with Wn, Skip if <	1	1 (2 or 3)	None
CPSNE	CPSNE	Wb,Wn	Compare Wb with Wn, Skip if ≠	1	1 (2 or 3)	None
DAW	DAW	Wn	Wn = Decimal Adjust Wn	1	1	C
DEC	DEC	f	f = f –1	1	1	C, DC, N, OV, Z
	DEC	f,WREG	WREG = f –1	1	1	C, DC, N, OV, Z
	DEC	Ws,Wd	Wd = Ws – 1	1	1	C, DC, N, OV, Z
DEC2	DEC2	f	f = f – 2	1	1	C, DC, N, OV, Z
	DEC2	f,WREG	WREG = f – 2	1	1	C, DC, N, OV, Z
	DEC2	Ws,Wd	Wd = Ws – 2	1	1	C, DC, N, OV, Z
DISI	DISI	#lit14	Disable Interrupts for k Instruction Cycles	1	1	None
DIV	DIV.SW	Wm,Wn	Signed 16/16-bit Integer Divide	1	18	N, Z, C, OV
	DIV.SD	Wm,Wn	Signed 32/16-bit Integer Divide	1	18	N, Z, C, OV
	DIV.UW	Wm,Wn	Unsigned 16/16-bit Integer Divide	1	18	N, Z, C, OV
	DIV.UD	Wm,Wn	Unsigned 32/16-bit Integer Divide	1	18	N, Z, C, OV
EXCH	EXCH	Wns,Wnd	Swap Wns with Wnd	1	1	None

TABLE 24-2: INSTRUCTION SET OVERVIEW (CONTINUED)

Assembly Mnemonic	Assembly Syntax		Description	# of Words	# of Cycles	Status Flags Affected
FF1L	FF1L	Ws,Wnd	Find First One from Left (MSb) Side	1	1	C
FF1R	FF1R	Ws,Wnd	Find First One from Right (LSb) Side	1	1	C
GOTO	GOTO	Expr	Go to Address	2	2	None
	GOTO	Wn	Go to Indirect	1	2	None
INC	INC	f	f = f + 1	1	1	C, DC, N, OV, Z
	INC	f,WREG	WREG = f + 1	1	1	C, DC, N, OV, Z
	INC	Ws,Wd	Wd = Ws + 1	1	1	C, DC, N, OV, Z
INC2	INC2	f	f = f + 2	1	1	C, DC, N, OV, Z
	INC2	f,WREG	WREG = f + 2	1	1	C, DC, N, OV, Z
	INC2	Ws,Wd	Wd = Ws + 2	1	1	C, DC, N, OV, Z
IOR	IOR	f	f = f .IOR. WREG	1	1	N, Z
	IOR	f,WREG	WREG = f .IOR. WREG	1	1	N, Z
	IOR	#lit10,Wn	Wd = lit10 .IOR. Wd	1	1	N, Z
	IOR	Wb,Ws,Wd	Wd = Wb .IOR. Ws	1	1	N, Z
	IOR	Wb,#lit5,Wd	Wd = Wb .IOR. lit5	1	1	N, Z
LNK	LNK	#lit14	Link Frame Pointer	1	1	None
LSR	LSR	f	f = Logical Right Shift f	1	1	C, N, OV, Z
	LSR	f,WREG	WREG = Logical Right Shift f	1	1	C, N, OV, Z
	LSR	Ws,Wd	Wd = Logical Right Shift Ws	1	1	C, N, OV, Z
	LSR	Wb,Wns,Wnd	Wnd = Logical Right Shift Wb by Wns	1	1	N, Z
	LSR	Wb,#lit5,Wnd	Wnd = Logical Right Shift Wb by lit5	1	1	N, Z
MOV	MOV	f,Wn	Move f to Wn	1	1	None
	MOV	[Wns+Slit10],Wnd	Move [Wns+Slit10] to Wnd	1	1	None
	MOV	f	Move f to f	1	1	N, Z
	MOV	f,WREG	Move f to WREG	1	1	N, Z
	MOV	#lit16,Wn	Move 16-bit Literal to Wn	1	1	None
	MOV.b	#lit8,Wn	Move 8-bit Literal to Wn	1	1	None
	MOV	Wn,f	Move Wn to f	1	1	None
	MOV	Wns,[Wns+Slit10]	Move Wns to [Wns+Slit10]	1	1	
	MOV	Wso,Wdo	Move Ws to Wd	1	1	None
	MOV	WREG,f	Move WREG to f	1	1	N, Z
	MOV.D	Wns,Wd	Move Double from W(ns):W(ns+1) to Wd	1	2	None
	MOV.D	Ws,Wnd	Move Double from Ws to W(nd+1):W(nd)	1	2	None
MUL	MUL.SS	Wb,Ws,Wnd	{Wnd+1, Wnd} = Signed(Wb) * Signed(Ws)	1	1	None
	MUL.SU	Wb,Ws,Wnd	{Wnd+1, Wnd} = Signed(Wb) * Unsigned(Ws)	1	1	None
	MUL.US	Wb,Ws,Wnd	{Wnd+1, Wnd} = Unsigned(Wb) * Signed(Ws)	1	1	None
	MUL.UU	Wb,Ws,Wnd	{Wnd+1, Wnd} = Unsigned(Wb) * Unsigned(Ws)	1	1	None
	MUL.SU	Wb,#lit5,Wnd	{Wnd+1, Wnd} = Signed(Wb) * Unsigned(lit5)	1	1	None
	MUL.UU	Wb,#lit5,Wnd	{Wnd+1, Wnd} = Unsigned(Wb) * Unsigned(lit5)	1	1	None
	MUL	f	W3:W2 = f * WREG	1	1	None
NEG	NEG	f	$f = \overline{f} + 1$	1	1	C, DC, N, OV, Z
	NEG	f,WREG	$WREG = \overline{f} + 1$	1	1	C, DC, N, OV, Z
	NEG	Ws,Wd	$Wd = \overline{Ws} + 1$	1	1	C, DC, N, OV, Z
NOP	NOP		No Operation	1	1	None
	NOPR		No Operation	1	1	None
POP	POP	f	Pop f from Top-of-Stack (TOS)	1	1	None
	POP	Wdo	Pop from Top-of-Stack (TOS) to Wdo	1	1	None
	POP.D	Wnd	Pop from Top-of-Stack (TOS) to W(nd):W(nd+1)	1	2	None
	POP.S		Pop Shadow Registers	1	1	All

TABLE 24-2: INSTRUCTION SET OVERVIEW (CONTINUED)

Assembly Mnemonic	Assembly Syntax		Description	# of Words	# of Cycles	Status Flags Affected
PUSH	PUSH	f	Push f to Top-of-Stack (TOS)	1	1	None
	PUSH	Wso	Push Wso to Top-of-Stack (TOS)	1	1	None
	PUSH.D	Wns	Push W(ns):W(ns+1) to Top-of-Stack (TOS)	1	2	None
	PUSH.S		Push Shadow Registers	1	1	None
PWRSAV	PWRSAV	#lit1	Go into Sleep or Idle mode	1	1	WDTO, Sleep
RCALL	RCALL	Expr	Relative Call	1	2	None
	RCALL	Wn	Computed Call	1	2	None
REPEAT	REPEAT	#lit14	Repeat Next Instruction lit14 + 1 times	1	1	None
	REPEAT	Wn	Repeat Next Instruction (Wn) + 1 times	1	1	None
RESET	RESET		Software Device Reset	1	1	None
RETFIE	RETFIE		Return from Interrupt	1	3 (2)	None
RETLW	RETLW	#lit10,Wn	Return with Literal in Wn	1	3 (2)	None
RETURN	RETURN		Return from Subroutine	1	3 (2)	None
RLC	RLC	f	f = Rotate Left through Carry f	1	1	C, N, Z
	RLC	f,WREG	WREG = Rotate Left through Carry f	1	1	C, N, Z
	RLC	Ws,Wd	Wd = Rotate Left through Carry Ws	1	1	C, N, Z
RLNC	RLNC	f	f = Rotate Left (No Carry) f	1	1	N, Z
	RLNC	f,WREG	WREG = Rotate Left (No Carry) f	1	1	N, Z
	RLNC	Ws,Wd	Wd = Rotate Left (No Carry) Ws	1	1	N, Z
RRC	RRC	f	f = Rotate Right through Carry f	1	1	C, N, Z
	RRC	f,WREG	WREG = Rotate Right through Carry f	1	1	C, N, Z
	RRC	Ws,Wd	Wd = Rotate Right through Carry Ws	1	1	C, N, Z
RRNC	RRNC	f	f = Rotate Right (No Carry) f	1	1	N, Z
	RRNC	f,WREG	WREG = Rotate Right (No Carry) f	1	1	N, Z
	RRNC	Ws,Wd	Wd = Rotate Right (No Carry) Ws	1	1	N, Z
SE	SE	Ws,Wnd	Wnd = Sign-Extended Ws	1	1	C, N, Z
SETM	SETM	f	f = FFFFh	1	1	None
	SETM	WREG	WREG = FFFFh	1	1	None
	SETM	Ws	Ws = FFFFh	1	1	None
SL	SL	f	f = Left Shift f	1	1	C, N, OV, Z
	SL	f,WREG	WREG = Left Shift f	1	1	C, N, OV, Z
	SL	Ws,Wd	Wd = Left Shift Ws	1	1	C, N, OV, Z
	SL	Wb,Wns,Wnd	Wnd = Left Shift Wb by Wns	1	1	N, Z
	SL	Wb,#lit5,Wnd	Wnd = Left Shift Wb by lit5	1	1	N, Z
SUB	SUB	f	f = f – WREG	1	1	C, DC, N, OV, Z
	SUB	f,WREG	WREG = f – WREG	1	1	C, DC, N, OV, Z
	SUB	#lit10,Wn	Wn = Wn – lit10	1	1	C, DC, N, OV, Z
	SUB	Wb,Ws,Wd	Wd = Wb – Ws	1	1	C, DC, N, OV, Z
	SUB	Wb,#lit5,Wd	Wd = Wb – lit5	1	1	C, DC, N, OV, Z
SUBB	SUBB	f	f = f – WREG – ($\overline{C}$)	1	1	C, DC, N, OV, Z
	SUBB	f,WREG	WREG = f – WREG – ($\overline{C}$)	1	1	C, DC, N, OV, Z
	SUBB	#lit10,Wn	Wn = Wn – lit10 – ($\overline{C}$)	1	1	C, DC, N, OV, Z
	SUBB	Wb,Ws,Wd	Wd = Wb – Ws – ($\overline{C}$)	1	1	C, DC, N, OV, Z
	SUBB	Wb,#lit5,Wd	Wd = Wb – lit5 – ($\overline{C}$)	1	1	C, DC, N, OV, Z
SUBR	SUBR	f	f = WREG – f	1	1	C, DC, N, OV, Z
	SUBR	f,WREG	WREG = WREG – f	1	1	C, DC, N, OV, Z
	SUBR	Wb,Ws,Wd	Wd = Ws – Wb	1	1	C, DC, N, OV, Z
	SUBR	Wb,#lit5,Wd	Wd = lit5 – Wb	1	1	C, DC, N, OV, Z

TABLE 24-2: INSTRUCTION SET OVERVIEW (CONTINUED)

Assembly Mnemonic	Assembly Syntax		Description	# of Words	# of Cycles	Status Flags Affected
SUBBR	SUBBR	f	f = WREG – f – ($\overline{C}$)	1	1	C, DC, N, OV, Z
	SUBBR	f,WREG	WREG = WREG – f – ($\overline{C}$)	1	1	C, DC, N, OV, Z
	SUBBR	Wb,Ws,Wd	Wd = Ws – Wb – ($\overline{C}$)	1	1	C, DC, N, OV, Z
	SUBBR	Wb,#lit5,Wd	Wd = lit5 – Wb – ($\overline{C}$)	1	1	C, DC, N, OV, Z
SWAP	SWAP.b	Wn	Wn = Nibble Swap Wn	1	1	None
	SWAP	Wn	Wn = Byte Swap Wn	1	1	None
TBLRDH	TBLRDH	Ws,Wd	Read Prog<23:16> to Wd<7:0>	1	2	None
TBLRDL	TBLRDL	Ws,Wd	Read Prog<15:0> to Wd	1	2	None
TBLWTH	TBLWTH	Ws,Wd	Write Ws<7:0> to Prog<23:16>	1	2	None
TBLWTL	TBLWTL	Ws,Wd	Write Ws to Prog<15:0>	1	2	None
ULNK	ULNK		Unlink Frame Pointer	1	1	None
XOR	XOR	f	f = f .XOR. WREG	1	1	N, Z
	XOR	f,WREG	WREG = f .XOR. WREG	1	1	N, Z
	XOR	#lit10,Wn	Wd = lit10 .XOR. Wd	1	1	N, Z
	XOR	Wb,Ws,Wd	Wd = Wb .XOR. Ws	1	1	N, Z
	XOR	Wb,#lit5,Wd	Wd = Wb .XOR. lit5	1	1	N, Z
ZE	ZE	Ws,Wnd	Wnd = Zero-Extend Ws	1	1	C, Z, N

NOTES:

25.0 DEVELOPMENT SUPPORT

The PICmicro® microcontrollers are supported with a full range of hardware and software development tools:

- Integrated Development Environment
  - MPLAB® IDE Software
- Assemblers/Compilers/Linkers
  - MPASM™ Assembler
  - MPLAB C18 and MPLAB C30 C Compilers
  - MPLINK™ Object Linker/ MPLIB™ Object Librarian
  - MPLAB ASM30 Assembler/Linker/Library
- Simulators
  - MPLAB SIM Software Simulator
- Emulators
  - MPLAB ICE 2000 In-Circuit Emulator
  - MPLAB ICE 4000 In-Circuit Emulator
- In-Circuit Debugger
  - MPLAB ICD 2
- Device Programmers
  - PICSTART® Plus Development Programmer
  - MPLAB PM3 Device Programmer
- Low-Cost Demonstration and Development Boards and Evaluation Kits

25.1 MPLAB Integrated Development Environment Software

The MPLAB IDE software brings an ease of software development previously unseen in the 8/16-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)
  - 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
- Visual device initializer for easy register initialization
- 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 HI-TECH Software C Compilers and IAR C Compilers

The MPLAB IDE allows you to:

- Edit your source files (either assembly or C)
- One touch assemble (or compile) and download to PICmicro MCU emulator and simulator tools (automatically updates all project information)
- Debug using:
  - Source files (assembly or C)
  - Mixed assembly and C
  - 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 MPASM Assembler

The MPASM Assembler is a full-featured, universal macro assembler for all PICmicro 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

25.3 MPLAB C18 and MPLAB C30 C Compilers

The MPLAB C18 and MPLAB C30 Code Development Systems are complete ANSI C compilers for Microchip's PIC18 family of microcontrollers and dsPIC30F family of digital signal controllers. These compilers provide powerful integration capabilities, superior code optimization and ease of use not found with other compilers.

For easy source level debugging, the compilers provide symbol information that is optimized to the MPLAB IDE debugger.

25.4 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

25.5 MPLAB ASM30 Assembler, Linker and Librarian

MPLAB ASM30 Assembler produces relocatable machine code from symbolic assembly language for dsPIC30F devices. MPLAB C30 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 dsPIC30F instruction set
- Support for fixed-point and floating-point data
- Command line interface
- Rich directive set
- Flexible macro language
- MPLAB IDE compatibility

25.6 MPLAB SIM Software Simulator

The MPLAB SIM Software Simulator allows code development in a PC-hosted environment by simulating the PICmicro 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, as well as internal registers.

The MPLAB SIM Software Simulator fully supports symbolic debugging using the MPLAB C18 and MPLAB C30 C Compilers, and the MPASM and MPLAB ASM30 Assemblers. The software simulator offers the flexibility to develop and debug code outside of the laboratory environment, making it an excellent, economical software development tool.

25.7 MPLAB ICE 2000 High-Performance In-Circuit Emulator

The MPLAB ICE 2000 In-Circuit Emulator is intended to provide the product development engineer with a complete microcontroller design tool set for PICmicro microcontrollers. Software control of the MPLAB ICE 2000 In-Circuit Emulator is advanced by the MPLAB Integrated Development Environment, which allows editing, building, downloading and source debugging from a single environment.

The MPLAB ICE 2000 is a full-featured emulator system with enhanced trace, trigger and data monitoring features. Interchangeable processor modules allow the system to be easily reconfigured for emulation of different processors. The architecture of the MPLAB ICE 2000 In-Circuit Emulator allows expansion to support new PICmicro microcontrollers.

The MPLAB ICE 2000 In-Circuit Emulator system has been designed as a real-time emulation system with advanced features that are typically found on more expensive development tools. The PC platform and Microsoft® Windows® 32-bit operating system were chosen to best make these features available in a simple, unified application.

25.8 MPLAB ICE 4000 High-Performance In-Circuit Emulator

The MPLAB ICE 4000 In-Circuit Emulator is intended to provide the product development engineer with a complete microcontroller design tool set for high-end PICmicro MCUs and dsPIC DSCs. Software control of the MPLAB ICE 4000 In-Circuit Emulator is provided by the MPLAB Integrated Development Environment, which allows editing, building, downloading and source debugging from a single environment.

The MPLAB ICE 4000 is a premium emulator system, providing the features of MPLAB ICE 2000, but with increased emulation memory and high-speed performance for dsPIC30F and PIC18XXXX devices. Its advanced emulator features include complex triggering and timing, and up to 2 Mb of emulation memory.

The MPLAB ICE 4000 In-Circuit Emulator system has been designed as a real-time emulation system with advanced features that are typically found on more expensive development tools. The PC platform and Microsoft Windows 32-bit operating system were chosen to best make these features available in a simple, unified application.

25.9 MPLAB ICD 2 In-Circuit Debugger

Microchip's In-Circuit Debugger, MPLAB ICD 2, is a powerful, low-cost, run-time development tool, connecting to the host PC via an RS-232 or high-speed USB interface. This tool is based on the Flash PICmicro MCUs and can be used to develop for these and other PICmicro MCUs and dsPIC DSCs. The MPLAB ICD 2 utilizes the in-circuit debugging capability built into the Flash devices. This feature, along with Microchip's In-Circuit Serial Programming™ (ICSP™) protocol, offers cost-effective, in-circuit Flash debugging from the graphical user interface of the MPLAB Integrated Development Environment. This enables a designer to develop and debug source code by setting breakpoints, single stepping and watching variables, and CPU status and peripheral registers. Running at full speed enables testing hardware and applications in real time. MPLAB ICD 2 also serves as a development programmer for selected PICmicro devices.

25.10 MPLAB PM3 Device Programmer

The MPLAB PM3 Device Programmer is a universal, CE compliant device programmer with programmable voltage verification at V_{DDMIN} and V_{DDMAX} for maximum reliability. It features a large LCD display (128 x 64) for menus and error messages and a modular, detachable socket assembly to support various package types. The ICSP™ cable assembly is included as a standard item. In Stand-Alone mode, the MPLAB PM3 Device Programmer can read, verify and program PICmicro 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 SD/MMC card for file storage and secure data applications.

25.11 PICSTART Plus Development Programmer

The PICSTART Plus Development Programmer is an easy-to-use, low-cost, prototype programmer. It connects to the PC via a COM (RS-232) port. MPLAB Integrated Development Environment software makes using the programmer simple and efficient. The PICSTART Plus Development Programmer supports most PICmicro devices in DIP packages up to 40 pins. Larger pin count devices, such as the PIC16C92X and PIC17C76X, may be supported with an adapter socket. The PICSTART Plus Development Programmer is CE compliant.

25.12 Demonstration, Development and Evaluation Boards

A wide variety of demonstration, development and evaluation boards for various PICmicro 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.

Check the Microchip web page (www.microchip.com) and the latest *"Product Selector Guide"* (DS00148) for the complete list of demonstration, development and evaluation kits.

26.0 ELECTRICAL CHARACTERISTICS

This section provides an overview of the PIC24FJ128GA family electrical characteristics. Additional information will be provided in future revisions of this document as it becomes available.

Absolute maximum ratings for the PIC24FJ128GA family are listed below. Exposure to these maximum rating conditions for extended periods may affect device reliability. Functional operation of the device at these, or any other conditions above the parameters indicated in the operation listings of this specification, is not implied.

Absolute Maximum Ratings(†)

Ambient temperature under bias ..... -40°C to +85°C
Storage temperature ..... -65°C to +150°C
Voltage on V_{DD} with respect to V_{SS} ..... -0.3V to +4.0V
Voltage on any combined analog and digital pin and $\overline{MCLR}$, with respect to V_{SS} ..... -0.3V to (V_{DD} + 0.3V)
Voltage on any digital-only pin with respect to V_{SS} ..... -0.3V to +6.0V
Voltage on V_{DDCORE} with respect to V_{SS} ..... -0.3V to +3.0V
Maximum current out of V_{SS} pin ..... 300 mA
Maximum current into V_{DD} pin **(Note 1)** ..... 250 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 **(Note 1)** ..... 200 mA

Note 1: Maximum allowable current is a function of device maximum power dissipation (see Table 26-2).

†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.

26.1 DC Characteristics

TABLE 26-1: OPERATING MIPS VS. VOLTAGE

VDD Range (in Volts)	Temp Range (in °C)	Max MIPS
		PIC24FJ128GA
2.0-3.6V	-40°C to +85°C	16

TABLE 26-2: THERMAL OPERATING CONDITIONS

Rating	Symbol	Min	Typ	Max	Unit
PIC24FJ128GA:					
Operating Junction Temperature Range	TJ	-40	—	+125	°C
Operating Ambient Temperature Range	TA	-40	—	+85	°C
Power Dissipation: Internal Chip Power Dissipation: PINT = VDD x (IDD – Σ IOH) I/O Pin Power Dissipation: PI/O = Σ ({VDD – VOH} x IOH) + Σ (VOL x IOL)	PD	PINT + PI/O			W
Maximum Allowed Power Dissipation	PDMAX	(TJ – TA)/θJA			W

TABLE 26-3: THERMAL PACKAGING CHARACTERISTICS

Characteristic	Symbol	Typ	Max	Unit	Notes
Package Thermal Resistance, 14x14x1 mm TQFP	θJA	50	—	°C/W	**(Note 1)**
Package Thermal Resistance, 12x12x1 mm TQFP	θJA	69.4	—	°C/W	**(Note 1)**
Package Thermal Resistance, 10x10x1 mm TQFP	θJA	76.6	—	°C/W	**(Note 1)**

Note 1: Junction to ambient thermal resistance, Theta-JA (θJA) numbers are achieved by package simulations.

TABLE 26-4: DC TEMPERATURE AND VOLTAGE SPECIFICATIONS

DC CHARACTERISTICS			Standard Operating Conditions: 2.5V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial				
Param No.	Symbol	Characteristic	Min	Typ(1)	Max	Units	Conditions
Operating Voltage							
DC10	**Supply Voltage**						
	VDD		2.7	—	3.6	V	Regulator enabled
	VDD		VDDCORE	—	3.6	V	Regulator disabled
	VDDCORE		2.0	—	2.75	V	Regulator disabled
DC12	VDR	**RAM Data Retention Voltage(2)**	1.5	—	—	V	
DC16	VPOR	**VDD Start Voltage** to ensure internal Power-on Reset signal	—	VSS	—	V	
DC17	SVDD	**VDD Rise Rate** to ensure internal Power-on Reset signal	0.05	—	—	V/ms	0-3.3V in 0.1s 0-2.5V in 60 ms

Note 1: Data in "Typ" column is at 3.3V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

2: This is the limit to which VDD can be lowered without losing RAM data.

TABLE 26-5: DC CHARACTERISTICS: OPERATING CURRENT (IDD)

DC CHARACTERISTICS			Standard Operating Conditions: 2.5V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial			
Parameter No.	Typical(1)	Max	Units	Conditions		
Operating Current (IDD)(2)						
DC20	—	—	mA	-40°C	2.5V(3)	1 MIPS
DC20a	TBD	—	mA	+25°C		
DC20b	—	—	mA	+85°C		
DC20d	—	—	mA	-40°C	3.3V(4)	
DC20e	TBD	—	mA	+25°C		
DC20f	—	—	mA	+85°C		
DC23	—	—	mA	-40°C	2.5V(3)	4 MIPS
DC23a	TBD	—	mA	+25°C		
DC23b	—	—	mA	+85°C		
DC23d	—	—	mA	-40°C	3.3V(4)	
DC23e	TBD	—	mA	+25°C		
DC23f	—	—	mA	+85°C		
DC24	—	—	mA	-40°C	2.5V(3)	16 MIPS
DC24a	TBD	—	mA	+25°C		
DC24b	—	—	mA	+85°C		
DC24d	—	—	mA	-40°C	3.3V(4)	
DC24e	TBD	—	mA	+25°C		
DC24f	—	—	mA	+85°C		
DC31	—	—	µA	-40°C	2.5V(3)	LPRC (31 kHz)
DC31a	TBD	—	µA	+25°C		
DC31b	—	—	µA	+85°C		
DC31d	—	—	µA	-40°C	3.3V(4)	
DC31e	TBD	—	µA	+25°C		
DC31f	—	—	µA	+85°C		

Legend: TBD = To Be Determined

Note 1: Data in "Typical" column is at 3.3V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

2: The supply current is mainly a function of the operating voltage and frequency. Other factors, such as I/O pin loading and switching rate, oscillator type, internal code execution pattern and temperature, also have an impact on the current consumption. The test conditions for all IDD measurements are as follows: OSC1 driven with external square wave from rail to rail. All I/O pins are configured as inputs and pulled to VDD. $\overline{\text{MCLR}}$ = VDD; WDT and FSCM are disabled. CPU, SRAM, program memory and data memory are operational. No peripheral modules are operating.

3: On-chip voltage regulator disabled (ENVREG tied to VSS).

4: On-chip voltage regulator enabled (ENVREG tied to VDD).

TABLE 26-6: DC CHARACTERISTICS: IDLE CURRENT (IIDLE)

DC CHARACTERISTICS			Standard Operating Conditions: 2.5V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial			
Parameter No.	Typical(1)	Max	Units	Conditions		
Idle Current (IIDLE): Core Off, Clock On Base Current(2)						
DC40	—	—	mA	-40°C	2.5V(3)	1 MIPS
DC40a	TBD	—	mA	+25°C		
DC40b	—	—	mA	+85°C		
DC40d	—	—	mA	-40°C	3.3V(4)	
DC40e	TBD	—	mA	+25°C		
DC40f	—	—	mA	+85°C		
DC43	—	—	mA	-40°C	2.5V(3)	4 MIPS
DC43a	TBD	—	mA	+25°C		
DC43b	—	—	mA	+85°C		
DC43d	—	—	mA	-40°C	3.3V(4)	
DC43e	TBD	—	mA	+25°C		
DC43f	—	—	mA	+85°C		
DC47	—	—	mA	-40°C	2.5V(3)	16 MIPS
DC47a	TBD	—	mA	+25°C		
DC47b	—	—	mA	+85°C		
DC47c	—	—	mA	-40°C	3.3V(4)	
DC47d	TBD	—	mA	+25°C		
DC47e	—	—	mA	+85°C		
DC50	—	—	mA	-40°C	2.5V(3)	FRC (4 MIPS)
DC50a	TBD	—	mA	+25°C		
DC50b	—	—	mA	+85°C		
DC50d	—	—	mA	-40°C	3.3V(4)	
DC50e	TBD	—	mA	+25°C		
DC50f	—	—	mA	+85°C		
DC51	—	—	µA	-40°C	2.5V(3)	LPRC (31 kHz)
DC51a	TBD	—	µA	+25°C		
DC51b	—	—	µA	+85°C		
DC51d	—	—	µA	-40°C	3.3V(4)	
DC51e	TBD	—	µA	+25°C		
DC51f	—	—	µA	+85°C		

Legend: TBD = To Be Determined

Note 1: Data in "Typical" column is at 3.3V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

2: Base IIDLE current is measured with core off, clock on and all modules turned off.

3: On-chip voltage regulator disabled (ENVREG tied to VSS).

4: On-chip voltage regulator enabled (ENVREG tied to VDD).

TABLE 26-7: DC CHARACTERISTICS: POWER-DOWN CURRENT (IPD)

DC CHARACTERISTICS			Standard Operating Conditions: 2.5V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial			
Parameter No.	Typical(1)	Max	Units	Conditions		
Power-Down Current (IPD)(2)						
DC60	—	—	µA	-40°C	2.0V(3)	Base Power-Down Current(5)
DC60a	TBD	—	µA	+25°C		
DC60b	—	—	µA	+85°C		
DC60c	—	—	µA	-40°C	2.5V(3)	
DC60d	TBD	—	µA	+25°C		
DC60e	—	—	µA	+85°C		
DC60f	—	—	µA	-40°C	3.3V(4)	
DC60g	TBD	—	µA	+25°C		
DC60h	—	—	µA	+85°C		
DC61	—	—	µA	-40°C	2.0V(3)	Watchdog Timer Current: ΔIWDT(5)
DC61a	TBD	—	µA	+25°C		
DC61b	—	—	µA	+85°C		
DC61c	—	—	µA	-40°C	2.5V(3)	
DC61d	TBD	—	µA	+25°C		
DC61e	—	—	µA	+85°C		
DC61f	—	—	µA	-40°C	3.3V(4)	
DC61g	TBD	—	µA	+25°C		
DC61h	—	—	µA	+85°C		
DC62	—	—	µA	-40°C	2.0V(3)	Timer1 w/32 kHz Crystal: ΔITI32(5)
DC62a	TBD	—	µA	+25°C		
DC62b	—	—	µA	+85°C		
DC62c	—	—	µA	-40°C	2.5V(3)	
DC62d	TBD	—	µA	+25°C		
DC62e	—	—	µA	+85°C		
DC62f	—	—	µA	-40°C	3.3V(4)	
DC62g	TBD	—	µA	+25°C		
DC62h	—	—	µA	+85°C		

Legend: TBD = To Be Determined

Note 1: Data in the Typical column is at 3.3V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

2: Base IPD is measured with all peripherals and clocks shut down. All I/Os are configured as inputs and pulled high. WDT, etc., are all switched off.

3: On-chip voltage regulator disabled (ENVREG tied to VSS).

4: On-chip voltage regulator enabled (ENVREG tied to VDD).

5: The Δ current is the additional current consumed when the module is enabled. This current should be added to the base IPD current.

TABLE 26-8: DC CHARACTERISTICS: I/O PIN INPUT SPECIFICATIONS

DC CHARACTERISTICS			Standard Operating Conditions: 2.5V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial				
Param No.	Sym	Characteristic	Min	Typ(1)	Max	Units	Conditions
	VIL	**Input Low Voltage**					
DI10		I/O pins	VSS	—	0.2 VDD	V	
DI11		PMP pins	VSS	—	0.15 VDD	V	PMPTTL = 1
DI15		$\overline{MCLR}$	VSS	—	0.2 VDD	V	
DI16		OSC1 (XT mode)	VSS	—	0.2 VDD	V	
DI17		OSC1 (HS mode)	VSS	—	0.2 VDD	V	
DI18		SDAx, SCLx	VSS	—	0.3 VDD	V	SMBus disabled
DI19		SDAx, SCLx	VSS	—	0.8	V	SMBus enabled
	VIH	**Input High Voltage**					
DI20		I/O pins:					
		With Analog Functions	0.8 VDD	—	VDD	V	
		Digital-Only	0.8 VDD	—	5.5	V	
DI21		PMP pins	0.25 VDD + 0.8	—	VDD	V	PMPTTL = 1
DI25		$\overline{MCLR}$	0.8 VDD	—	VDD	V	
DI26		OSC1 (XT mode)	0.7 VDD	—	VDD	V	
DI27		OSC1 (HS mode)	0.7 VDD	—	VDD	V	
DI28		SDAx, SCLx	0.7 VDD	—	VDD	V	SMBus disabled
DI29		SDAx, SCLx	2.1	—	VDD	V	SMBus enabled, 2.5V ≤ VPIN ≤ VDD
DI30	ICNPU	**CNxx Pull-up Current**	50	250	400	µA	VDD = 3.3V, VPIN = VSS
	IIL	**Input Leakage Current(2,3)**					
DI50		I/O Ports	—	TBD	TBD	µA	VSS ≤ VPIN ≤ VDD, Pin at high-impedance
DI51		Analog Input pins	—	TBD	TBD	µA	VSS ≤ VPIN ≤ VDD, Pin at high-impedance
DI55		$\overline{MCLR}$	—	TBD	TBD	µA	VSS ≤ VPIN ≤ VDD
DI56		OSC1	—	TBD	TBD	µA	VSS ≤ VPIN ≤ VDD, XT and HS modes

Legend: TBD = To Be Determined

Note 1: Data in "Typ" column is at 3.3V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

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 26-9: DC CHARACTERISTICS: I/O PIN OUTPUT SPECIFICATIONS

DC CHARACTERISTICS			Standard Operating Conditions: 2.5V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial				
Param No.	Sym	Characteristic	Min	Typ(1)	Max	Units	Conditions
	VOL	**Output Low Voltage**					
DO10		I/O Ports	—	—	0.4	V	IOL = 8.5 mA, VDD = 3.6V
			—	—	0.4	V	IOL = 6.0 mA, VDD = 2.0V
DO16		OSC2/CLKO	—	—	0.4	V	IOL = 1.6 mA, VDD = 3.6V
			—	—	TBD	V	IOL = 2.0 mA, VDD = 2.5V
	VOH	**Output High Voltage**					
DO20		I/O Ports	2.4	—	—	V	IOH = -6.0 mA, VDD = 3.6V
			1.4	—	—	V	IOH = -3.0 mA, VDD = 2.0V
DO26		OSC2/CLKO	2.4	—	—	V	IOH = -1.3 mA, VDD = 3.6V
			TBD	—	—	V	IOH = -2.0 mA, VDD = 2.5V

Legend: TBD = To Be Determined

Note 1: Data in "Typ" column is at 3.3V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

TABLE 26-10: DC CHARACTERISTICS: PROGRAM MEMORY

DC CHARACTERISTICS			Standard Operating Conditions: 2.5V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial				
Param No.	Sym	Characteristic	Min	Typ(1)	Max	Units	Conditions
		Program Flash Memory					
D130	EP	Cell Endurance	100	1K	—	E/W	-40°C to +85°C
D131	VPR	VDD for Read	VMIN	—	3.6	V	VMIN = Minimum operating voltage
D132B	VPEW	VDD for Self-Timed Write	VMIN	—	3.6	V	VMIN = Minimum operating voltage
D133A	TIW	Self-Timed Write Cycle Time	—	3	—	ms	
D134	TRETD	Characteristic Retention	20	—	—	Year	Provided no other specifications are violated
D135	IDDP	Supply Current during Programming	—	10	—	mA	

Note 1: Data in "Typ" column is at 3.3V, 25°C unless otherwise stated.

TABLE 26-11: INTERNAL VOLTAGE REGULATOR SPECIFICATIONS

Operating Conditions: -40°C < TA < +85°C (unless otherwise stated)							
Param No.	Symbol	Characteristics	Min	Typ	Max	Units	Comments
	VRGOUT	Regulator Output Voltage	—	2.5	—	V	
	CEFC	External Filter Capacitor Value	1	10	—	µF	Capacitor must be low series resistance
	TVREG		—	10	—	µs	ENVREG = 1
	TPWRT		—	64	—	ms	ENVREG = 0

FIGURE 26-2: EXTERNAL CLOCK TIMING

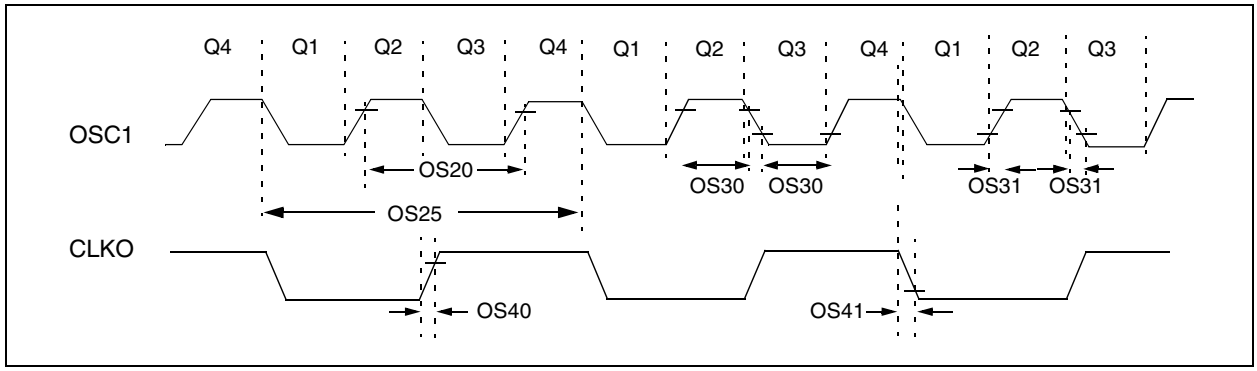


TABLE 26-14: EXTERNAL CLOCK TIMING REQUIREMENTS

AC CHARACTERISTICS			Standard Operating Conditions: 2.5V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial				
Param No.	**Sym**	**Characteristic**	**Min**	**Typ(1)**	**Max**	**Units**	**Conditions**
OS10	FOSC	External CLKI Frequency (External clocks allowed only in EC mode)	DC 4	— —	32 8	MHz MHz	EC ECPLL
		Oscillator Frequency	0.2 2 4 4 31	— — — — —	4 4 10 8 33	MHz MHz MHz MHz kHz	XT XTPLL HS HSPLL SOSC
OS20	TOSC	TOSC = 1/FOSC	—	—	—	—	See parameter OS10 for FOSC value
OS25	TCY	Instruction Cycle Time(2)	33	—	DC	ns	
OS30	TosL, TosH	External Clock in (OSC1) High or Low Time	0.45 x TOSC	—	—	ns	EC
OS31	TosR, TosF	External Clock in (OSC1) Rise or Fall Time	—	—	20	ns	EC
OS40	TckR	CLKO Rise Time(3)	—	6	10	ns	
OS41	TckF	CLKO Fall Time(3)	—	6	10	ns	

Note 1: Data in "Typ" column is at 3.3V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

2: Instruction cycle period (TCY) equals two times the input oscillator time base period. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at "Min." values with an external clock applied to the OSC1/CLKI pin. When an external clock input is used, the "Max." cycle time limit is "DC" (no clock) for all devices.

3: Measurements are taken in EC mode. The CLKO signal is measured on the OSC2 pin. CLKO is low for the Q1-Q2 period (1/2 TCY) and high for the Q3-Q4 period (1/2 TCY).

TABLE 26-15: PLL CLOCK TIMING SPECIFICATIONS (V_{DD} = 2.5V TO 3.6V)

AC CHARACTERISTICS			Standard Operating Conditions: 2.5V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial				
Param No.	Sym	Characteristic(1)	Min	Typ(2)	Max	Units	Conditions
OS50	FPLLI	PLL Input Frequency Range(2)	2	—	8	MHz	ECPLL, HSPLL, XTPLL modes
OS51	FSYS	On-Chip VCO System Frequency	8	—	32	MHz	
OS52	TLOC	PLL Start-up Time (Lock Time)	—	—	2	ms	
OS53	DCLK	CLKO Stability (Jitter)	TBD	1	TBD	%	Measured over 100 ms period

Legend: TBD = To Be Determined

Note 1: These parameters are characterized but not tested in manufacturing.

2: Data in "Typ" column is at 3.3V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

TABLE 26-16: AC CHARACTERISTICS: INTERNAL RC ACCURACY

AC CHARACTERISTICS		Standard Operating Conditions: 2.5V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial					
Param No.	Characteristic	Min	Typ	Max	Units	Conditions	
F20	**Internal FRC Accuracy @ 8 MHz(1)**						
	FRC	TBD	—	TBD	%	+25°C	VDD = 3.0-3.6V
		TBD	—	TBD	%	-40°C ≤ TA ≤ +85°C	VDD = 3.0-3.6V

Legend: TBD = To Be Determined

Note 1: Frequency calibrated at 25°C and 3.3V. TUN bits can be used to compensate for temperature drift.

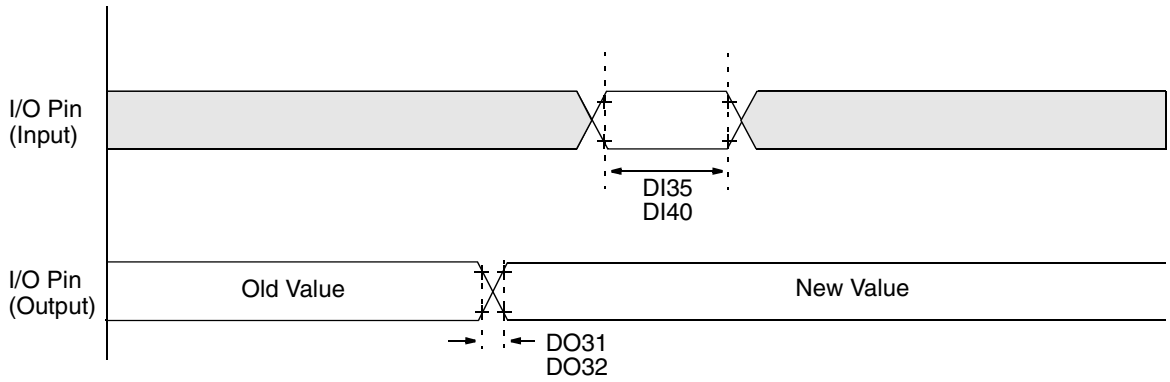
TABLE 26-17: INTERNAL RC ACCURACY

AC CHARACTERISTICS		Standard Operating Conditions: 2.5V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial					
Param No.	Characteristic	Min	Typ	Max	Units	Conditions	
F21	**LPRC @ 31 kHz(1)**						
		TBD	—	TBD	%	+25°C	VDD = 3.0-3.6V
		TBD	—	TBD	%	-40°C ≤ TA ≤ +85°C	VDD = 3.0-3.6V

Legend: TBD = To Be Determined

Note 1: Change of LPRC frequency as VDD changes.

FIGURE 26-3: CLKO AND I/O TIMING CHARACTERISTICS



Note: Refer to Figure 26-1 for load conditions.

TABLE 26-18: CLKO AND I/O TIMING REQUIREMENTS

AC CHARACTERISTICS			Standard Operating Conditions: 2.5V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial				
Param No.	**Sym**	**Characteristic**	**Min**	**Typ(1)**	**Max**	**Units**	**Conditions**
DO31	TIOR	Port Output Rise Time	—	10	25	ns	
DO32	TIOF	Port Output Fall Time	—	10	25	ns	
DI35	TINP	INTx pin High or Low Time (output)	20	—	—	ns	
DI40	TRBP	CNx High or Low Time (input)	2	—	—	TCY	

Note 1: Data in "Typ" column is at 3.3V, 25°C unless otherwise stated.

NOTES:

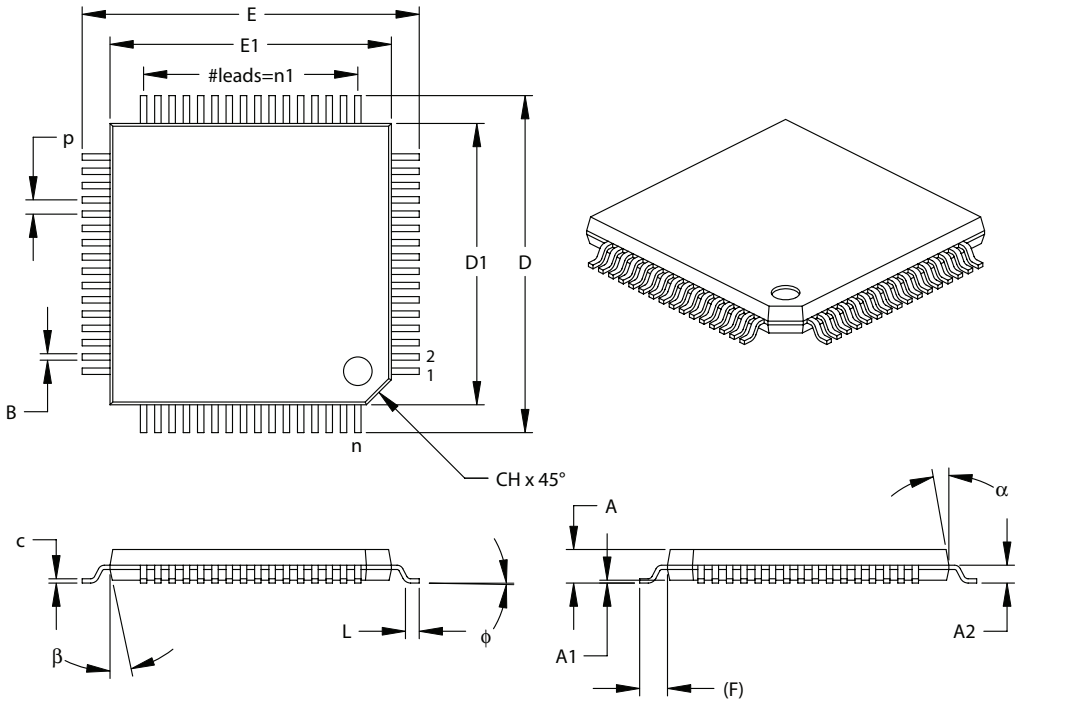
[redacted]

Legend:	XX...X	Cu ner-specific information
	Y	(last digit of calendar year)
	YY	t 2 digits of calendar year)
	(e	k of January 1 is week ‘01’)
		bility code
		or for Matte Tin (Sn)
		e Pb-free JEDEC designator ((e3))
		aging for this package.
	t be marked on one line, it will the number of available	

27.2 Package Details

The following sections give the technical details of the packages.

64-Lead Plastic Thin Quad Flatpack (PT) 10x10x1 mm Body, 1.0/0.10 mm Lead Form (TQFP)



Units		INCHES			MILLIMETERS*		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		64			64	
Pitch	p		.020			0.50	
Pins per Side	n1		16			16	
Overall Height	A	.039	.043	.047	1.00	1.10	1.20
Molded Package Thickness	A2	.037	.039	.041	0.95	1.00	1.05
Standoff	A1	.002	.006	.010	0.05	0.15	0.25
Foot Length	L	.018	.024	.030	0.45	0.60	0.75
Footprint (Reference)	(F)		.039			1.00	
Foot Angle	φ	0	3.5	7	0	3.5	7
Overall Width	E	.463	.472	.482	11.75	12.00	12.25
Overall Length	D	.463	.472	.482	11.75	12.00	12.25
Molded Package Width	E1	.390	.394	.398	9.90	10.00	10.10
Molded Package Length	D1	.390	.394	.398	9.90	10.00	10.10
Lead Thickness	c	.005	.007	.009	0.13	0.18	0.23
Lead Width	B	.007	.009	.011	0.17	0.22	0.27
Pin 1 Corner Chamfer	CH	.025	.035	.045	0.64	0.89	1.14
Mold Draft Angle Top	α	5	10	15	5	10	15
Mold Draft Angle Bottom	β	5	10	15	5	10	15

*Controlling Parameter

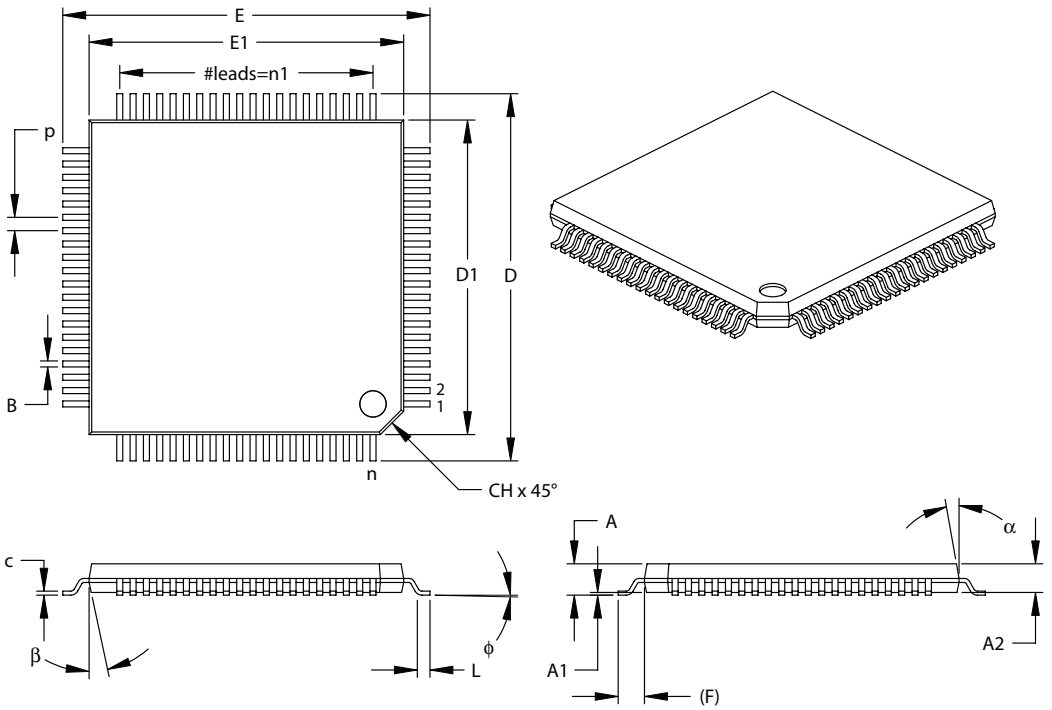
Notes:

Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC Equivalent: MS-026

Drawing No. C04-085

80-Lead Plastic Thin Quad Flatpack (PT) 12x12x1 mm Body, 1.0/0.10 mm Lead Form (TQFP)



Units		INCHES			MILLIMETERS*		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		80			80	
Pitch	p		.020			0.50	
Pins per Side	n1		20			20	
Overall Height	A	.039	.043	.047	1.00	1.10	1.20
Molded Package Thickness	A2	.037	.039	.041	0.95	1.00	1.05
Standoff	A1	.002	.004	.006	0.05	0.10	0.15
Foot Length	L	.018	.024	.030	0.45	0.60	0.75
Footprint (Reference)	(F)		.039			1.00	
Foot Angle	ϕ	0	3.5	7	0	3.5	7
Overall Width	E	.541	.551	.561	13.75	14.00	14.25
Overall Length	D	.541	.551	.561	13.75	14.00	14.25
Molded Package Width	E1	.463	.472	.482	11.75	12.00	12.25
Molded Package Length	D1	.463	.472	.482	11.75	12.00	12.25
Lead Thickness	c	.004	.006	.008	0.09	0.15	0.20
Lead Width	B	.007	.009	.011	0.17	0.22	0.27
Pin 1 Corner Chamfer	CH	.025	.035	.045	0.64	0.89	1.14
Mold Draft Angle Top	α	5	10	15	5	10	15
Mold Draft Angle Bottom	β	5	10	15	5	10	15

*Controlling Parameter

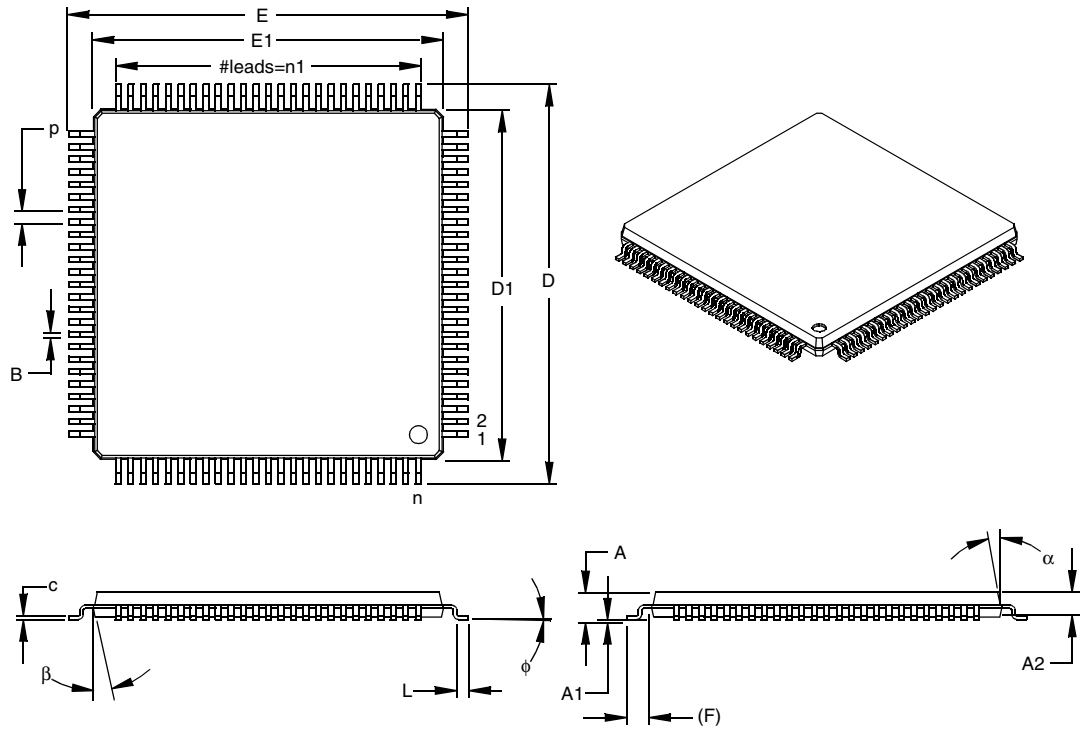
Notes:

Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

JEDEC Equivalent: MS-026

Drawing No. C04-092

100-Lead Plastic Thin Quad Flatpack (PF) 14x14x1 mm Body, 1.0/0.10 mm Lead Form (TQFP)



Units		INCHES			MILLIMETERS*		
Dimension Limits		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		100			100	
Pitch	p		.020			0.50	
Pins per Side	n1		25			25	
Overall Height	A			.047			1.20
Molded Package Thickness	A2	.037	.039	.041	0.95	1.00	1.05
Standoff §	A1	.002		.006	0.05		0.15
Foot Length	L	.018	.024	.030	0.45	0.60	0.75
Footprint (Reference)	(F)		.039			1.00	
Foot Angle	φ	0	3.5	7	0	3.5	7
Overall Width	E		.630 BSC			16.00 BSC	
Overall Length	D		.630 BSC			16.00 BSC	
Molded Package Width	E1		.551 BSC			14.00 BSC	
Molded Package Length	D1		.551 BSC			14.00 BSC	
Lead Thickness	c	.004		.008	0.09		0.20
Lead Width	B	.007	.009	.011	0.17	0.22	0.27
Mold Draft Angle Top	α	11	12	13	11	12	13
Mold Draft Angle Bottom	β	11	12	13	11	12	13

*Controlling Parameter
§ Significant Characteristic

Notes:
Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010” (0.254mm) per side.

JEDEC Equivalent: MS-026
Drawing No. C04-110

NOTES:

APPENDIX A: REVISION HISTORY

Revision A (September 2005)

Original data sheet for PIC24FJ128GA family devices.

Revision B (March 2006)

Update of electrical specifications.

INDEX

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, 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: PIC24FJ128GA Family Literature Number: DS39747B

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?

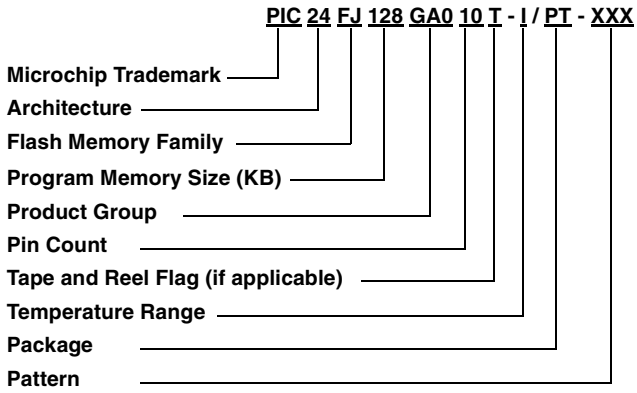
5. What deletions from the document could be made without affecting the overall usefulness?

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.



Architecture	24	= 16-bit modified Harvard without DSP
Flash Memory Family	FJ	= Flash program memory
Product Group	GA0	= General purpose microcontrollers
Pin Count	06 08 10	= 64-pin = 80-pin = 100-pin
Temperature Range	I	= -40°C to +85°C (Industrial)
Package	PT PF	= 64-Lead, 80-Lead, 100-Lead (12x12x1 mm) TQFP (Thin Quad Flatpack) = 100-Lead (14x14x1 mm) TQFP (Thin Quad Flatpack)
Pattern	Three-digit QTP, SQTP, Code or Special Requirements (blank otherwise) ES = Engineering Sample	

Examples:

a) PIC24FJ128GA008-I/PT 301:
General purpose PIC24, 96 KB program memory, 80-pin, Industrial temp., TQFP package, QTP pattern #301.

b) PIC24FJ128GA010-I/PT:
General purpose PIC24, 128 KB program memory, 100-pin, Industrial temp., TQFP package.

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
Alpharetta, GA
Tel: 770-640-0034
Fax: 770-640-0307

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
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

San Jose
Mountain View, CA
Tel: 650-215-1444
Fax: 650-961-0286

Toronto
Mississauga, Ontario,
Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

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: 86-28-8676-6200
Fax: 86-28-8676-6599

China - Fuzhou
Tel: 86-591-8750-3506
Fax: 86-591-8750-3521

China - Hong Kong SAR
Tel: 852-2401-1200
Fax: 852-2401-3431

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 - Shunde
Tel: 86-757-2839-5507
Fax: 86-757-2839-5571

China - Wuhan
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

China - Xian
Tel: 86-29-8833-7250
Fax: 86-29-8833-7256

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-4182-8400
Fax: 91-80-4182-8422

India - New Delhi
Tel: 91-11-5160-8631
Fax: 91-11-5160-8632

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 - Gumi
Tel: 82-54-473-4301
Fax: 82-54-473-4302

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Malaysia - Penang
Tel: 60-4-646-8870
Fax: 60-4-646-5086

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: 886-3-572-9526
Fax: 886-3-572-6459

Taiwan - Kaohsiung
Tel: 886-7-536-4818
Fax: 886-7-536-4803

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: 43-7242-2244-399
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

02/16/06