



# **PIC24H Family Data Sheet**

High-Performance, 16-bit  
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.

---

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights.

#### **Trademarks**

The Microchip name and logo, the Microchip logo, Accuron, dsPIC, KEELOQ, microID, MPLAB, PIC, PICmicro, PICSTART, PRO MATE, PowerSmart, rfPIC and SmartShunt are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.


AmpLab, FilterLab, Migratable Memory, MXDEV, MXLAB, SEEVAL, SmartSensor and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Analog-for-the-Digital Age, Application Maestro, dsPICDEM, dsPICDEM.net, dsPICworks, ECAN, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, Linear Active Thermistor, Mindi, MiWi, MPASM, MPLIB, MPLINK, PICKit, PICDEM, PICDEM.net, PICLAB, PICtail, PowerCal, PowerInfo, PowerMate, PowerTool, REAL ICE, rLAB, rfPICDEM, Select Mode, Smart Serial, SmartTel, Total Endurance, UNI/O, WiperLock and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2006, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

 Printed on recycled paper.

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

*Microchip received ISO/TS-16949:2002 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona, Gresham, Oregon and Mountain View, California. 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.*

---

---

**High-Performance, 16-bit Microcontrollers**

---

---

**Operating Range**

- DC – 40 MIPS (40 MIPS @ 3.0-3.6V, -40°C to +85°C)
- Industrial temperature range (-40°C to +85°C)

**High-Performance DSC CPU**

- Modified Harvard architecture
- C compiler optimized instruction set
- 16-bit wide data path
- 24-bit wide instructions
- Linear program memory addressing up to 4M instruction words
- Linear data memory addressing up to 64 Kbytes
- 71 base instructions: mostly 1 word/1 cycle
- Sixteen 16-bit General Purpose Registers
- Flexible and powerful Indirect Addressing modes
- Software stack
- 16 x 16 multiply operations
- 32/16 and 16/16 divide operations
- Up to  $\pm 16$ -bit data shifts

**Direct Memory Access (DMA)**

- 8-channel hardware DMA
- 2 Kbytes dual ported DMA buffer area (DMA RAM) to store data transferred via DMA:
  - Allows data transfer between RAM and a peripheral while CPU is executing code (no cycle stealing)
- Most peripherals support DMA

**Interrupt Controller**

- 5-cycle latency
- 118 interrupt vectors
- Up to 61 available interrupt sources
- Up to 5 external interrupts
- 7 programmable priority levels
- 5 processor exceptions

**Digital I/O**

- Up to 85 programmable digital I/O pins
- Wake-up/Interrupt-on-Change on up to 24 pins
- Output pins can drive from 3.0V to 3.6V
- All digital input pins are 5V tolerant
- 4 mA sink on all I/O pins

**On-Chip Flash and SRAM**

- Flash program memory, up to 256 Kbytes
- Data SRAM, up to 16 Kbytes (includes 2 Kbytes of DMA RAM)

**System Management**

- Flexible clock options:
  - External, crystal, resonator, internal RC
  - Fully integrated PLL
  - Extremely low jitter PLL
- Power-up Timer
- Oscillator Start-up Timer/Stabilizer
- Watchdog Timer with its own RC oscillator
- Fail-Safe Clock Monitor
- Reset by multiple sources

**Power Management**

- On-chip 2.5V voltage regulator
- Switch between clock sources in real time
- Idle, Sleep and Doze modes with fast wake-up

**Timers/Capture/Compare/PWM**

- Timer/Counters, up to nine 16-bit timers:
  - Can pair up to make four 32-bit timers
  - 1 timer runs as Real-Time Clock with external 32.768 kHz oscillator
  - Programmable prescaler
- Input Capture (up to 8 channels):
  - Capture on up, down or both edges
  - 16-bit capture input functions
  - 4-deep FIFO on each capture
- Output Compare (up to 8 channels):
  - Single or Dual 16-Bit Compare mode
  - 16-bit Glitchless PWM mode

# PIC24H

---

## Communication Modules

- 3-wire SPI (up to 2 modules):
  - Framing supports I/O interface to simple codecs
  - Supports 8-bit and 16-bit data
  - Supports all serial clock formats and sampling modes
- I<sup>2</sup>C™ (up to 2 modules):
  - Full Multi-Master Slave mode support
  - 7-bit and 10-bit addressing
  - Bus collision detection and arbitration
  - Integrated signal conditioning
  - Slave address masking
- UART (up to 2 modules):
  - Interrupt on address bit detect
  - Interrupt on UART error
  - Wake-up on Start bit from Sleep mode
  - 4-character TX and RX FIFO buffers
  - LIN bus support
  - IrDA® encoding and decoding in hardware
  - High-Speed Baud mode
  - Hardware Flow Control with CTS and RTS
- Enhanced CAN (ECAN™ module) 2.0B active (up to 2 modules):
  - Up to 8 transmit and up to 32 receive buffers
  - 16 receive filters and 3 masks
  - Loopback, Listen Only and Listen All Messages modes for diagnostics and bus monitoring
  - Wake-up on CAN message
  - Automatic processing of Remote Transmission Requests
  - FIFO mode using DMA
  - DeviceNet™ addressing support

## Analog-to-Digital Converters

- Up to two A/D modules in a device
- 10-bit, 1.1 Msps or 12-bit, 500 ksps conversion:
  - 2, 4 or 8 simultaneous samples
  - Up to 32 input channels with auto-scanning
  - Conversion start can be manual or synchronized with 1 of 4 trigger sources
  - Conversion possible in Sleep mode
  - $\pm 2$  LSB max integral nonlinearity
  - $\pm 1$  LSB max differential nonlinearity

## CMOS Flash Technology

- Low-power, high-speed Flash technology
- Fully static design
- 3.3V ( $\pm 10\%$ ) operating voltage
- Industrial temperature
- Low-power consumption

## Packaging:

- 100-pin TQFP (14x14x1 mm and 12x12x1 mm)
- 64-pin TQFP (10x10x1 mm)

<b>Note:</b> See the device variant tables for exact peripheral features per device.
--

## PIC24H PRODUCT FAMILIES

The PIC24H General Purpose Family is ideal for a wide variety of 16-bit MCU embedded applications. The device names, pin counts, memory sizes and peripheral availability of each family are listed below, followed by their pinout diagrams.

### PIC24H General Purpose Family Variants

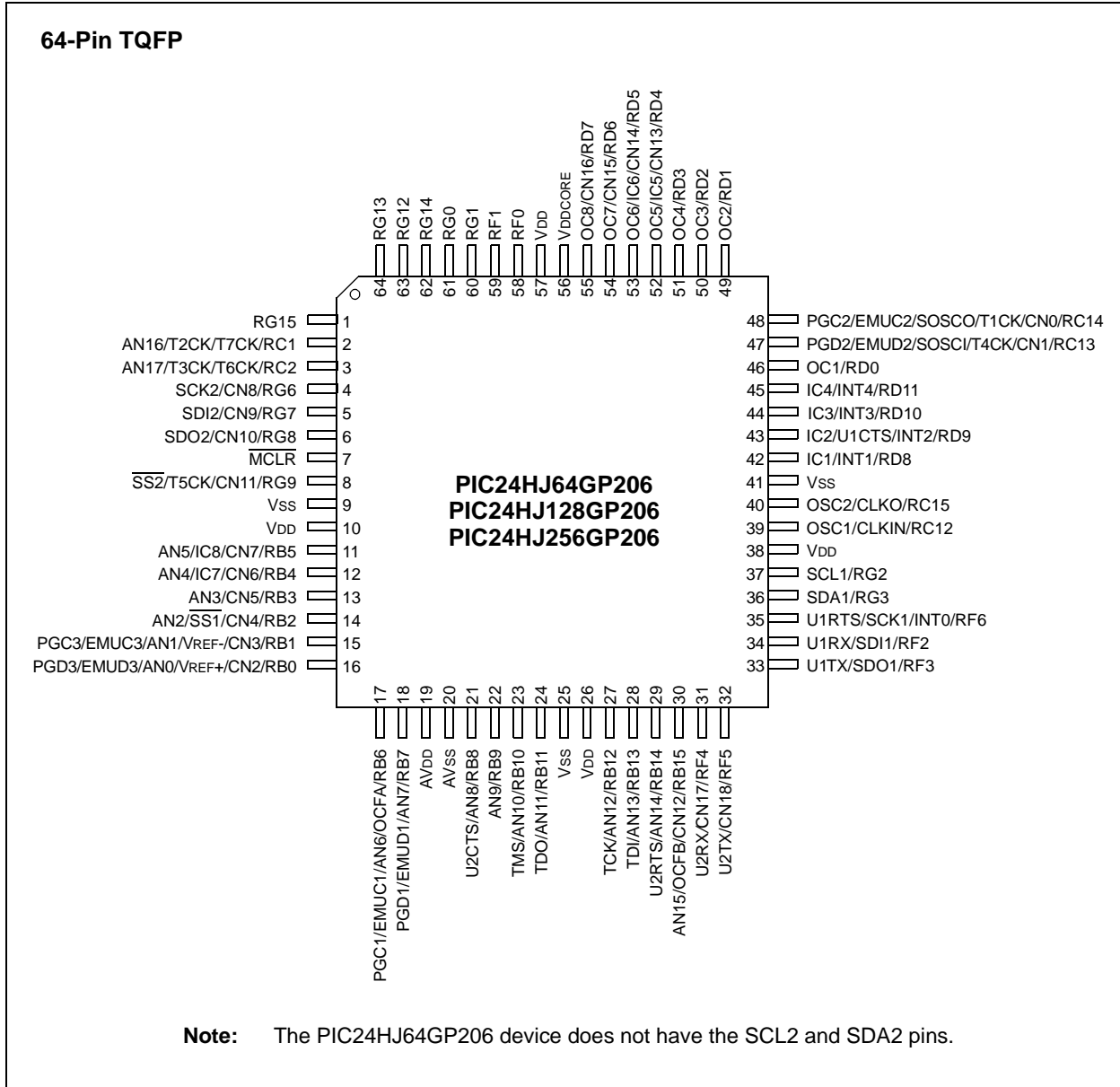
Device	Pins	Program Flash Memory (KB)	RAM <sup>(1)</sup> (KB)	DMA Channels	Timer 16-bit	Input Capture	Output Compare Std. PWM	Codec Interface	ADC	UART	SPI	I <sup>2</sup> C™	CAN	I/O Pins (Max) <sup>(2)</sup>	Packages
PIC24HJ64GP206	64	64	8	8	9	8	8	0	1 ADC, 18 ch	2	2	1	0	53	PT
PIC24HJ64GP210	100	64	8	8	9	8	8	0	1 ADC, 32 ch	2	2	2	0	85	PF, PT
PIC24HJ64GP506	64	64	8	8	9	8	8	0	1 ADC, 18 ch	2	2	2	1	53	PT
PIC24HJ64GP510	100	64	8	8	9	8	8	0	1 ADC, 32 ch	2	2	2	1	85	PF, PT
PIC24HJ128GP206	64	128	8	8	9	8	8	0	1 ADC, 18 ch	2	2	2	0	53	PT
PIC24HJ128GP210	100	128	8	8	9	8	8	0	1 ADC, 32 ch	2	2	2	0	85	PF, PT
PIC24HJ128GP506	64	128	8	8	9	8	8	0	1 ADC, 18 ch	2	2	2	1	53	PT
PIC24HJ128GP510	100	128	8	8	9	8	8	0	1 ADC, 32 ch	2	2	2	1	85	PF, PT
PIC24HJ128GP306	64	128	16	8	9	8	8	0	1 ADC, 18 ch	2	2	2	0	53	PT
PIC24HJ128GP310	100	128	16	8	9	8	8	0	1 ADC, 32 ch	2	2	2	0	85	PF, PT
PIC24HJ256GP206	64	256	16	8	9	8	8	0	1 ADC, 18 ch	2	2	2	0	53	PT
PIC24HJ256GP210	100	256	16	8	9	8	8	0	1 ADC, 32 ch	2	2	2	0	85	PF, PT
PIC24HJ256GP610	100	256	16	8	9	8	8	0	2 ADC, 32 ch	2	2	2	2	85	PF, PT

**Note 1:** RAM size is inclusive of 2 Kbytes DMA RAM.

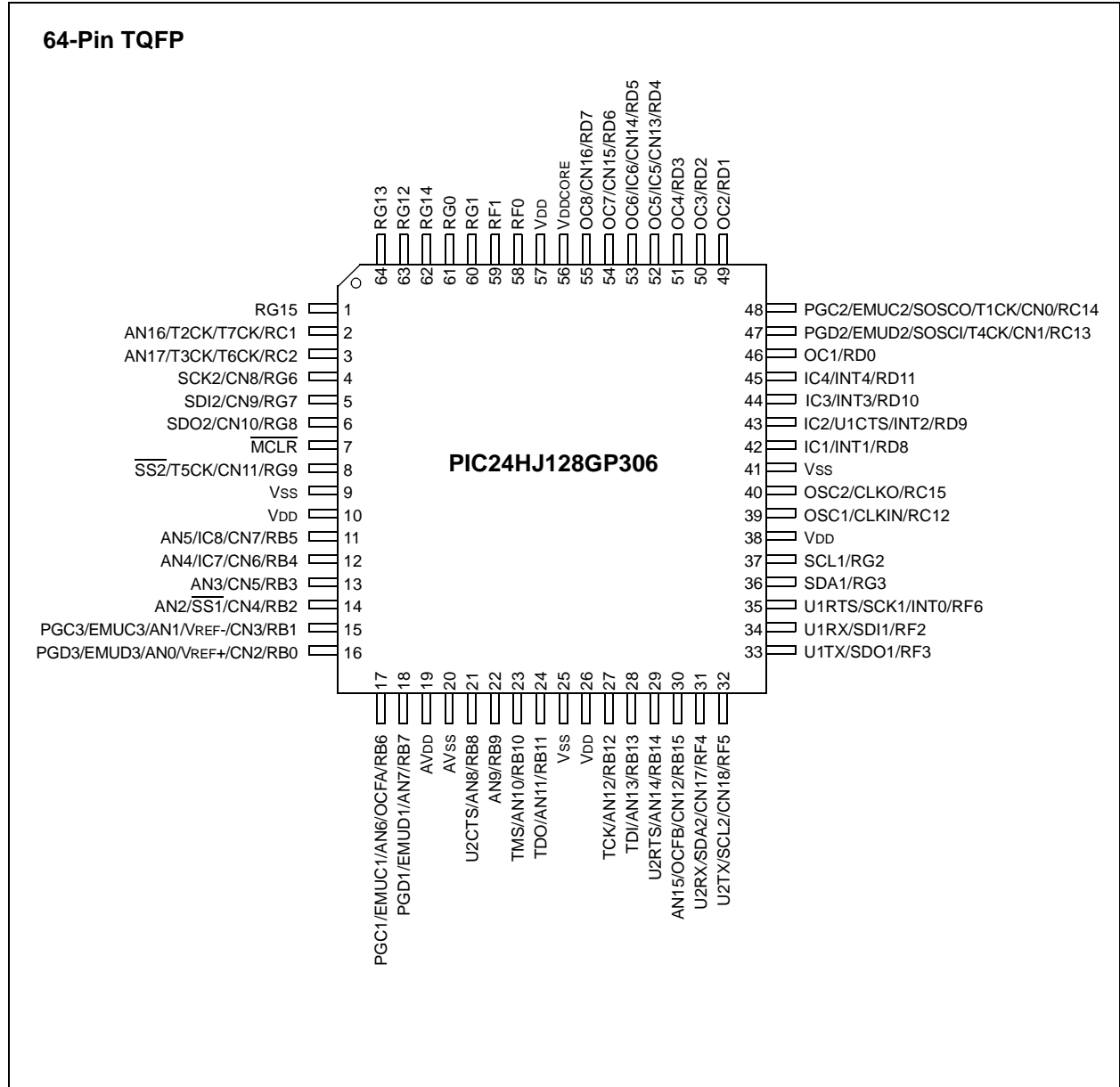
**Note 2:** Maximum I/O pin count includes pins shared by the peripheral functions.

# PIC24H

## Pin Diagrams

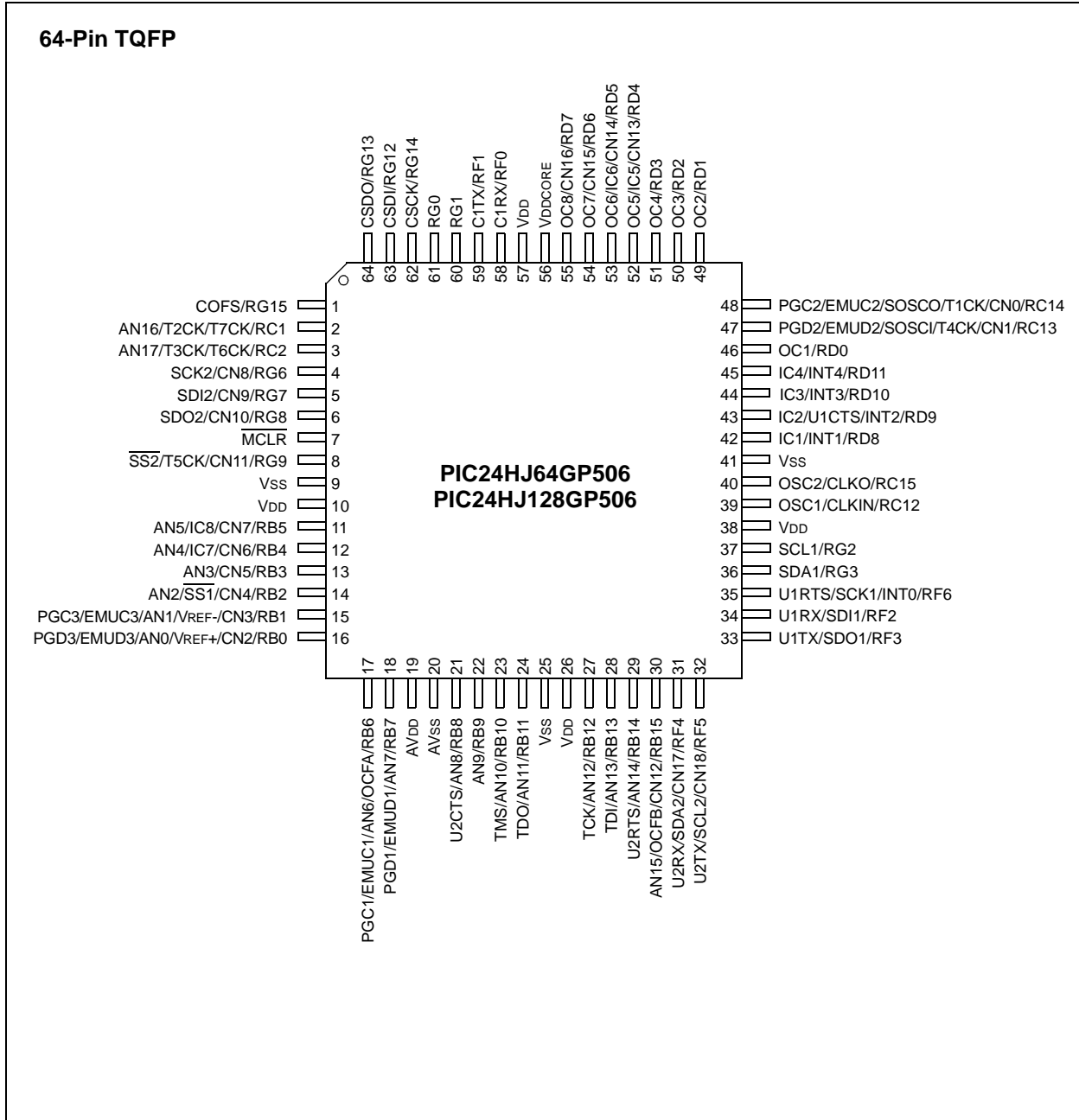


## Pin Diagrams (Continued)



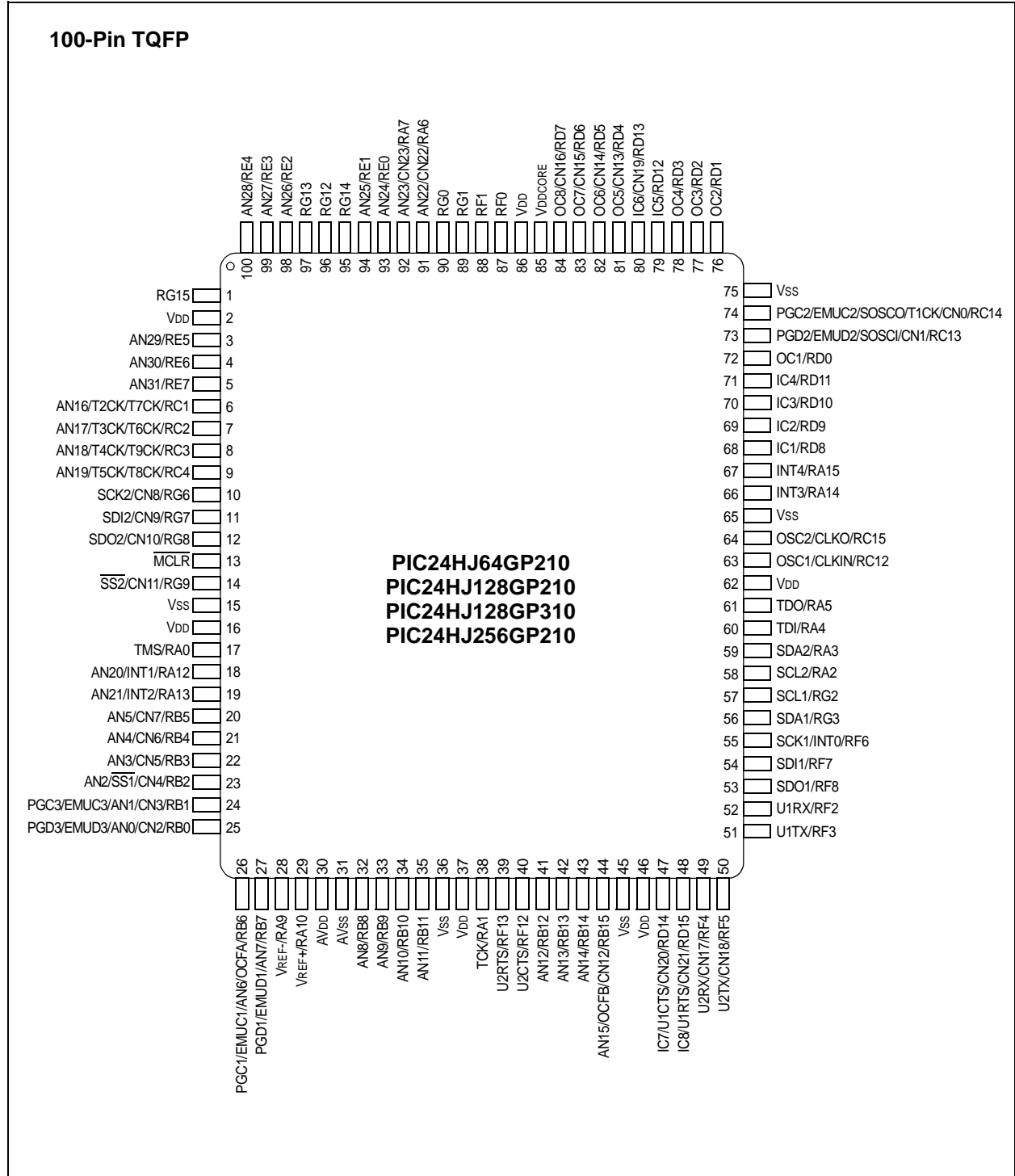
# PIC24H

## Pin Diagrams (Continued)



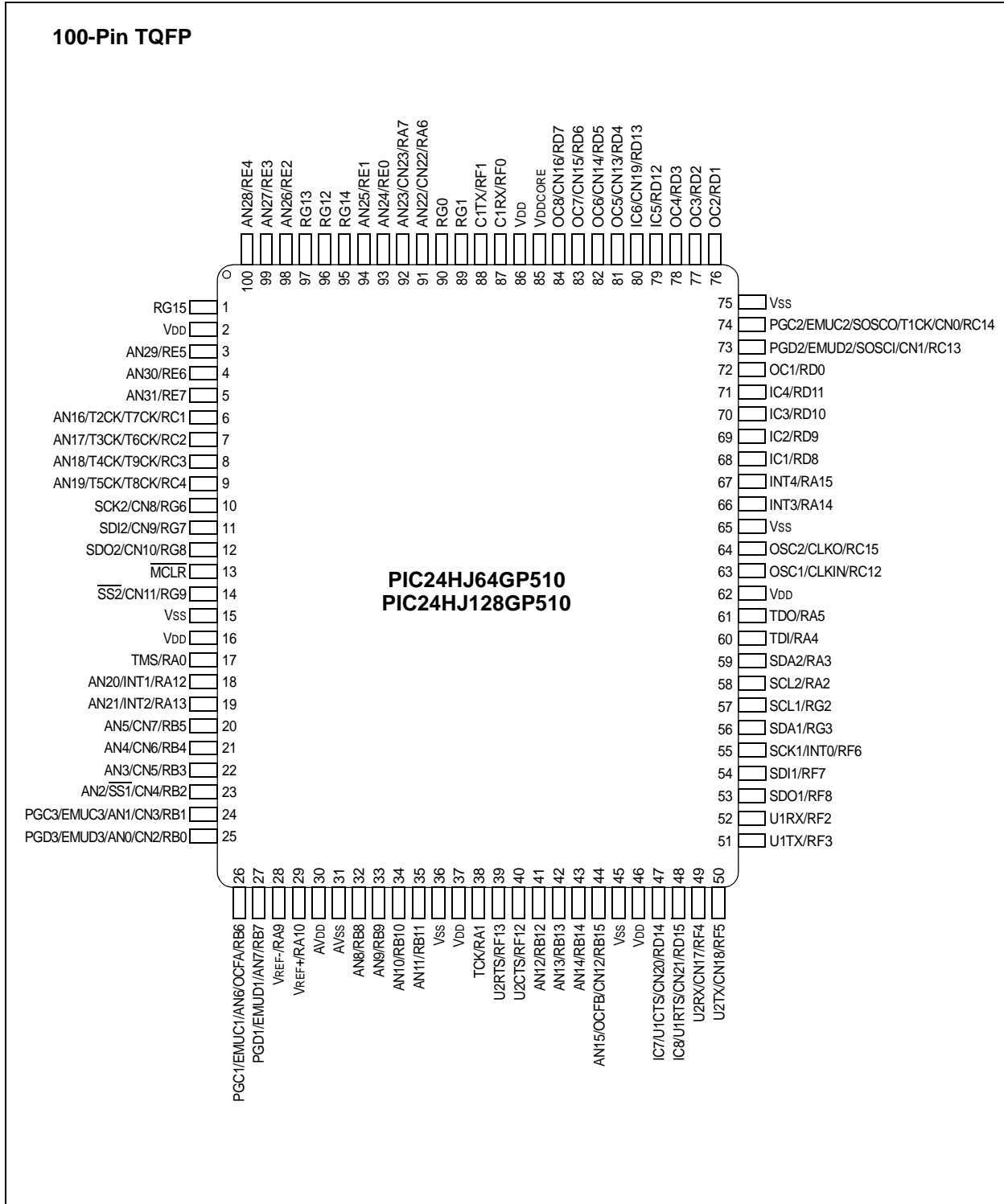


## Pin Diagrams (Continued)

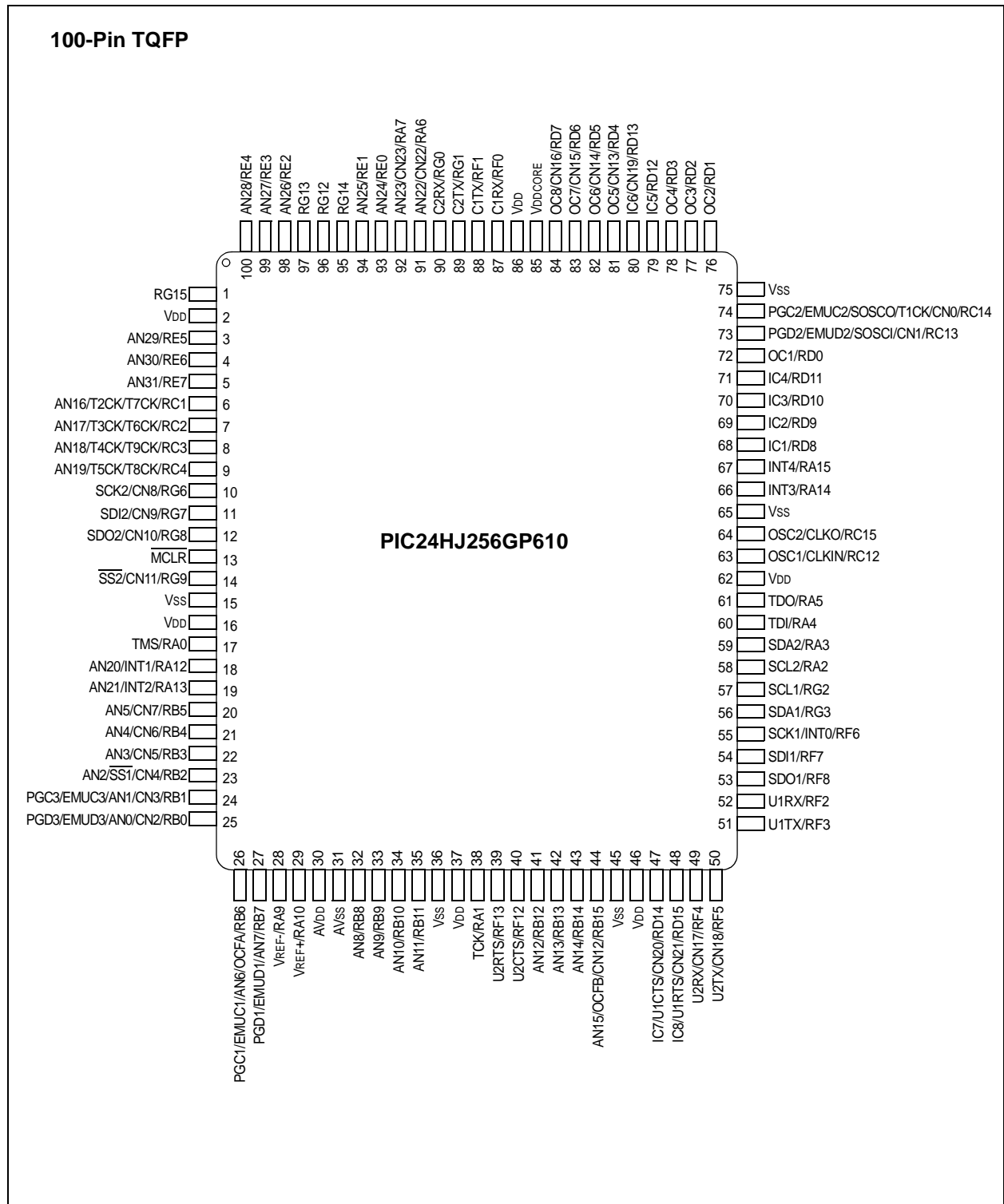


# PIC24H

## Pin Diagrams (Continued)



## Pin Diagrams (Continued)



# PIC24H

---

---

## Table of Contents

PIC24H Product Families .....	3
1.0 Device Overview .....	13
2.0 CPU .....	17
3.0 Memory Organization .....	25
4.0 Flash Program Memory .....	55
5.0 Resets .....	61
6.0 Interrupt Controller .....	67
7.0 Direct Memory Access (DMA) .....	111
8.0 Oscillator Configuration .....	125
9.0 Power-Saving Features .....	133
10.0 I/O Ports .....	135
11.0 Timer1 .....	137
12.0 Timer2/3, Timer4/5, Timer6/7 and Timer8/9 .....	139
13.0 Input Capture .....	145
14.0 Output Compare .....	147
15.0 Serial Peripheral Interface (SPI) .....	151
16.0 Inter-Integrated Circuit (I <sup>2</sup> C) .....	159
17.0 Universal Asynchronous Receiver Transmitter (UART) .....	169
18.0 Enhanced CAN Module .....	177
19.0 10-bit/12-bit A/D Converter .....	207
20.0 Special Features .....	221
21.0 Instruction Set Summary .....	227
22.0 Development Support .....	235
23.0 Electrical Characteristics .....	239
24.0 Packaging Information .....	273
Appendix A: Revision History .....	277
Index .....	279
The Microchip Web Site .....	283
Customer Change Notification Service .....	283
Customer Support .....	283
Reader Response .....	284
Product Identification System .....	285

## TO OUR VALUED CUSTOMERS

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

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

# PIC24H

---

NOTES:

## 1.0 DEVICE OVERVIEW

**Note:** This data sheet summarizes the features of this group of PIC24H devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the “*dsPIC30F Family Reference Manual*” (DS70046).

This document contains device specific information for the following devices:

- PIC24HJ64GP206
- PIC24HJ64GP210
- PIC24HJ64GP506
- PIC24HJ64GP510
- PIC24HJ128GP206
- PIC24HJ128GP210
- PIC24HJ128GP506
- PIC24HJ128GP510
- PIC24HJ128GP306
- PIC24HJ128GP310
- PIC24HJ256GP206
- PIC24HJ256GP210
- PIC24HJ256GP610

The PIC24H device family includes devices with different pin counts (64 and 100 pins), different program memory sizes (64 Kbytes, 128 Kbytes and 256 Kbytes) and different RAM sizes (8 Kbytes and 16 Kbytes).

This makes these families suitable for a wide variety of high-performance digital signal control applications. The devices are pin compatible with the dsPIC33F family of devices, and also share a very high degree of compatibility with the dsPIC30F family devices. This allows easy migration between device families as may be necessitated by the specific functionality, computational resource and system cost requirements of the application.

The PIC24H device family employs a powerful 16-bit architecture, ideal for applications that rely on high-speed, repetitive computations, as well as control.

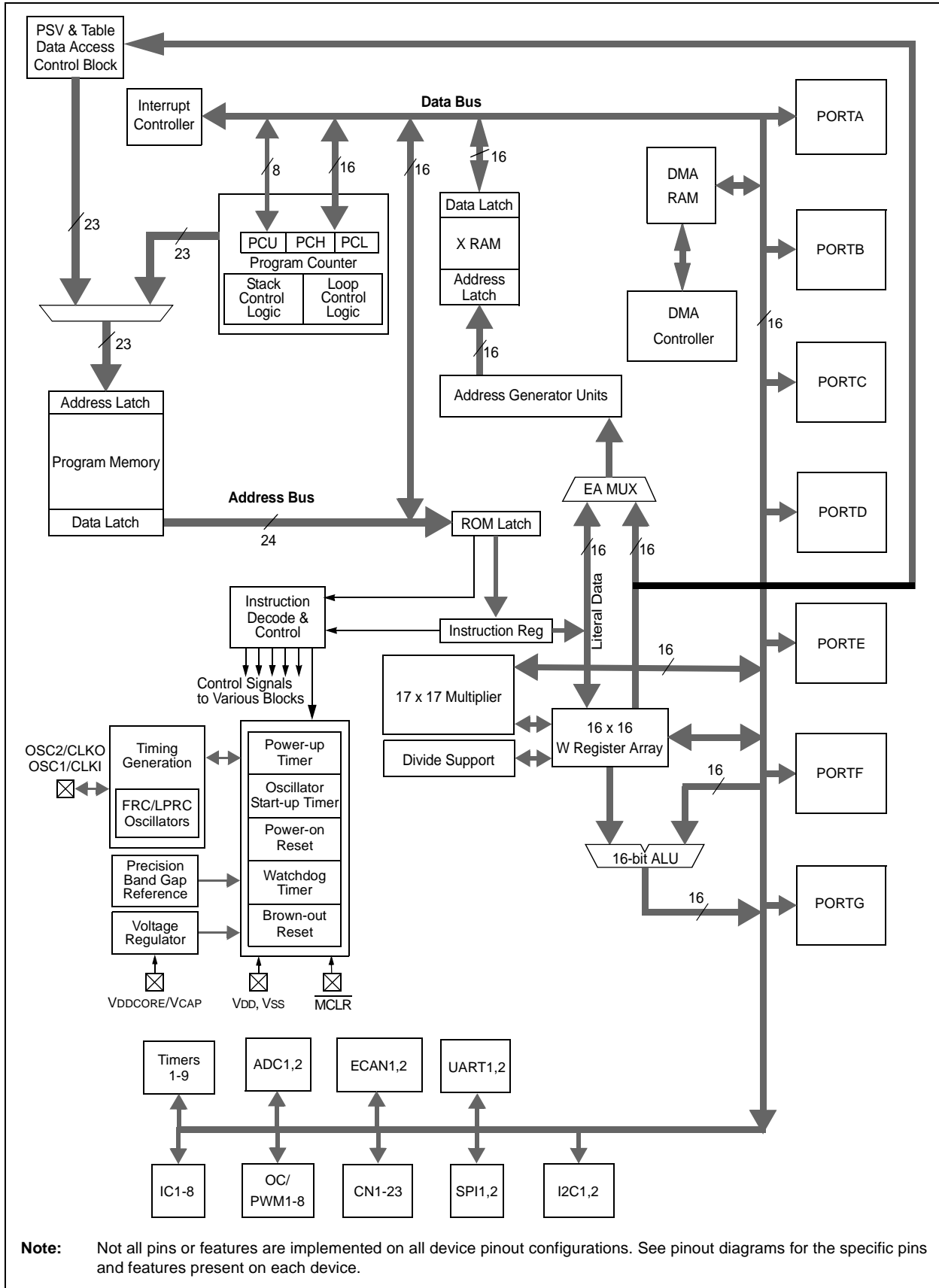
The 17 x 17 multiplier, hardware support for division operations, multi-bit data shifter, a large array of 16-bit working registers and a wide variety of data addressing modes, together provide the PIC24H Central Processing Unit (CPU) with extensive mathematical processing capability. Flexible and deterministic interrupt handling, coupled with a powerful array of peripherals, renders the PIC24H devices suitable for

control applications. Further, Direct Memory Access (DMA) enables overhead-free transfer of data between several peripherals and a dedicated DMA RAM. Reliable, field programmable Flash program memory ensures scalability of applications that use PIC24H devices.

Figure 1-1 shows a general block diagram of the various core and peripheral modules in the PIC24H family of devices, while Table 1-1 lists the functions of the various pins shown in the pinout diagrams.

# PIC24H

**FIGURE 1-1: PIC24H GENERAL BLOCK DIAGRAM**





**TABLE 1-1: PINOUT I/O DESCRIPTIONS**

Pin Name	Pin Type	Buffer Type	Description
AN0-AN31	I	Analog	Analog input channels.
AVDD	P	P	Positive supply for analog modules.
AVSS	P	P	Ground reference for analog modules.
CLKI	I	ST/CMOS	External clock source input. Always associated with OSC1 pin function.
CLKO	O	—	Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode. Optionally functions as CLKO in RC and EC modes. Always associated with OSC2 pin function.
CN0-CN23	I	ST	Input change notification inputs. Can be software programmed for internal weak pull-ups on all inputs.
C1RX	I	ST	ECAN1 bus receive pin.
C1TX	O	—	ECAN1 bus transmit pin.
C2RX	I	ST	ECAN2 bus receive pin.
C2TX	O	—	ECAN2 bus transmit pin.
PGD1/EMUD1	I/O	ST	Data I/O pin for programming/debugging communication channel 1.
PGC1/EMUC1	I	ST	Clock input pin for programming/debugging communication channel 1.
PGD2/EMUD2	I/O	ST	Data I/O pin for programming/debugging communication channel 2.
PGC2/EMUC2	I	ST	Clock input pin for programming/debugging communication channel 2.
PGD3/EMUD3	I/O	ST	Data I/O pin for programming/debugging communication channel 3.
PGC3/EMUC3	I	ST	Clock input pin for programming/debugging communication channel 3.
IC1-IC8	I	ST	Capture inputs 1 through 8.
INT0	I	ST	External interrupt 0.
INT1	I	ST	External interrupt 1.
INT2	I	ST	External interrupt 2.
INT3	I	ST	External interrupt 3.
INT4	I	ST	External interrupt 4.
MCLR	I/P	ST	Master Clear (Reset) input. This pin is an active-low Reset to the device.
OCFA	I	ST	Compare Fault A input (for Compare Channels 1, 2, 3 and 4).
OCFB	I	ST	Compare Fault B input (for Compare Channels 5, 6, 7 and 8).
OC1-OC8	O	—	Compare outputs 1 through 8.
OSC1	I	ST/CMOS	Oscillator crystal input. ST buffer when configured in RC mode; CMOS otherwise.
OSC2	I/O	—	Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode. Optionally functions as CLKO in RC and EC modes.
RA0-RA7	I/O	ST	PORTA is a bidirectional I/O port.
RA9-RA10	I/O	ST	
RA12-RA15	I/O	ST	
RB0-RB15	I/O	ST	PORTB is a bidirectional I/O port.
RC1-RC4	I/O	ST	PORTC is a bidirectional I/O port.
RC12-RC15	I/O	ST	
RD0-RD15	I/O	ST	PORTD is a bidirectional I/O port.
RE0-RE7	I/O	ST	PORTE is a bidirectional I/O port.
RF0-RF8	I/O	ST	PORTF is a bidirectional I/O port.
RF12-RF13	I/O	ST	
RG0-RG3	I/O	ST	PORTG is a bidirectional I/O port.
RG6-RG9	I/O	ST	
RG12-RG15	I/O	ST	

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

# PIC24H

**TABLE 1-1: PINOUT I/O DESCRIPTIONS (CONTINUED)**

Pin Name	Pin Type	Buffer Type	Description
SCK1	I/O	ST	Synchronous serial clock input/output for SPI1.
SDI1	I	ST	SPI1 data in.
SDO1	O	—	SPI1 data out.
SS1	I/O	ST	SPI1 slave synchronization or frame pulse I/O.
SCK2	I/O	ST	Synchronous serial clock input/output for SPI2.
SDI2	I	ST	SPI2 data in.
SDO2	O	—	SPI2 data out.
SS2	I/O	ST	SPI2 slave synchronization or frame pulse I/O.
SCL1	I/O	ST	Synchronous serial clock input/output for I2C1.
SDA1	I/O	ST	Synchronous serial data input/output for I2C1.
SCL2	I/O	ST	Synchronous serial clock input/output for I2C2.
SDA2	I/O	ST	Synchronous serial data input/output for I2C2.
SOSCI	I	ST/CMOS	32.768 kHz low-power oscillator crystal input; CMOS otherwise.
SOSCO	O	—	32.768 kHz low-power oscillator crystal output.
TMS	I	ST	JTAG Test mode select pin.
TCK	I	ST	JTAG test clock input pin.
TDI	I	ST	JTAG test data input pin.
TDO	O	—	JTAG test data output pin.
T1CK	I	ST	Timer1 external clock input.
T2CK	I	ST	Timer2 external clock input.
T3CK	I	ST	Timer3 external clock input.
T4CK	I	ST	Timer4 external clock input.
T5CK	I	ST	Timer5 external clock input.
T6CK	I	ST	Timer6 external clock input.
T7CK	I	ST	Timer7 external clock input.
T8CK	I	ST	Timer8 external clock input.
T9CK	I	ST	Timer9 external clock input.
U1CTS	I	ST	UART1 clear to send.
U1RTS	O	—	UART1 ready to send.
U1RX	I	ST	UART1 receive.
U1TX	O	—	UART1 transmit.
U2CTS	I	ST	UART2 clear to send.
U2RTS	O	—	UART2 ready to send.
U2RX	I	ST	UART2 receive.
U2TX	O	—	UART2 transmit.
VDD	P	—	Positive supply for peripheral logic and I/O pins.
VDDCORE	P	—	CPU logic filter capacitor connection.
VSS	P	—	Ground reference for logic and I/O pins.
VREF+	I	Analog	Analog voltage reference (high) input.
VREF-	I	Analog	Analog voltage reference (low) input.

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

## 2.0 CPU

**Note:** This data sheet summarizes the features of this group of PIC24H devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the “*dsPIC30F Family Reference Manual*” (DS70046).

The PIC24H CPU module has a 16-bit (data) modified Harvard architecture with an enhanced instruction set and addressing modes. The CPU has a 24-bit instruction word with a variable length opcode field. The Program Counter (PC) is 23 bits wide and addresses up to 4M x 24 bits of user program memory space. The actual amount of program memory implemented varies by device. 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, single-cycle program loop constructs are supported using the REPEAT instruction, which is interruptible at any point.

The PIC24H devices have sixteen, 16-bit working registers in the programmer’s model. Each of the working registers can serve as a data, address or address offset register. The 16th working register (W15) operates as a software Stack Pointer (SP) for interrupts and calls.

The PIC24H instruction set includes many addressing modes and is designed for optimum C compiler efficiency. For most instructions, the PIC24H 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  $A + B = C$  operations to be executed in a single cycle.

A block diagram of the CPU is shown in Figure 2-1, and the programmer’s model for the PIC24H is shown in Figure 2-2.

### 2.1 Data Addressing Overview

The data space can be linearly addressed as 32K words or 64 Kbytes using an Address Generation Unit (AGU). The upper 32 Kbytes of the data space memory map can optionally be mapped into program space at any 16K program 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 data space also includes 2 Kbytes of DMA RAM, which is primarily used for DMA data transfers, but may be used as general purpose RAM.

## 2.2 Special MCU Features

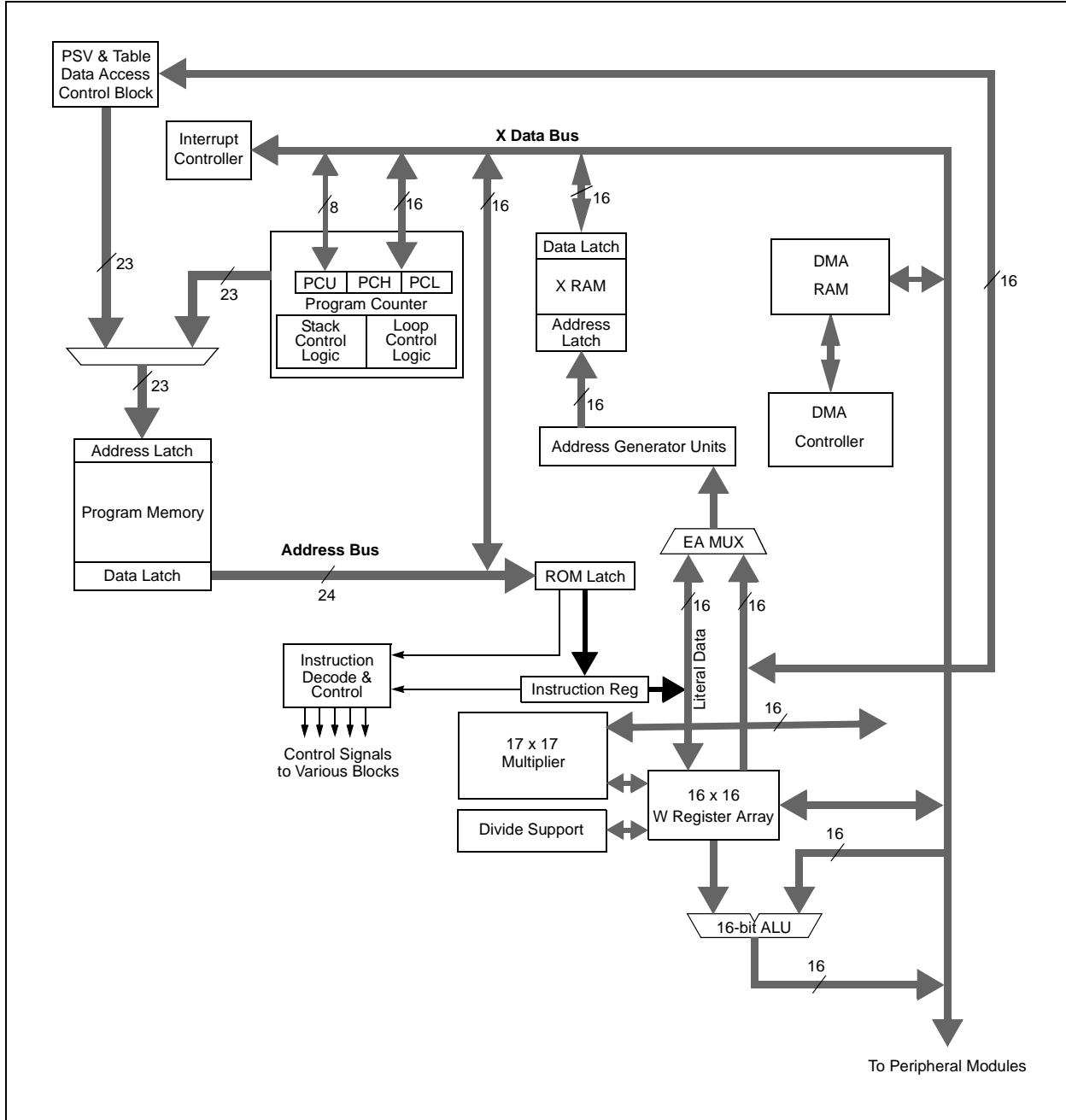
The PIC24H features a 17-bit by 17-bit, single-cycle multiplier. The multiplier can perform signed, unsigned and mixed-sign multiplication. Using a 17-bit by 17-bit multiplier for 16-bit by 16-bit multiplication makes mixed-sign multiplication possible.

The PIC24H supports 16/16 and 32/16 integer divide operations. All divide instructions are iterative operations. They must be executed within a REPEAT loop, resulting in a total execution time of 19 instruction cycles. The divide operation can be interrupted during any of those 19 cycles without loss of data.

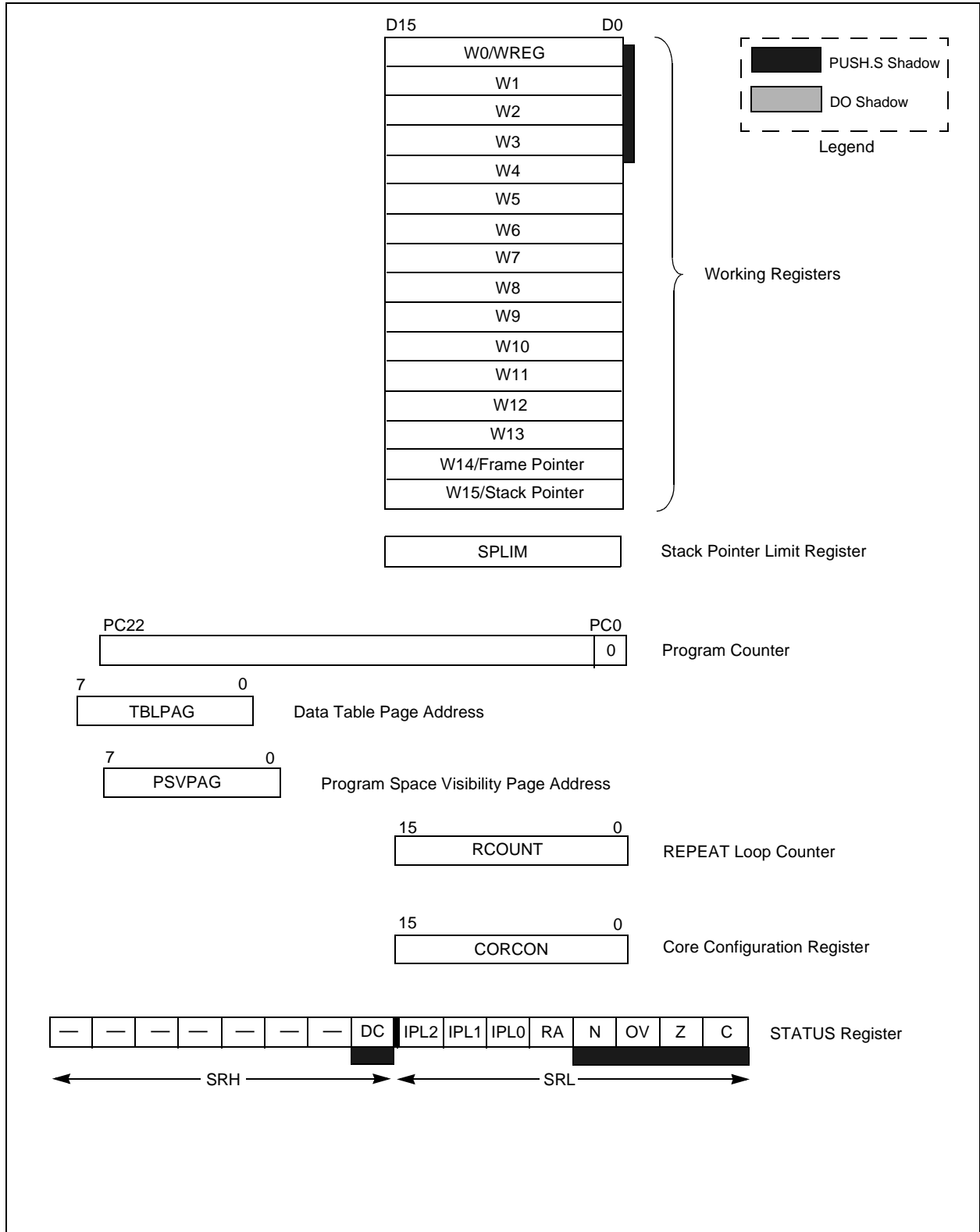
A multi-bit data shifter is used to perform up to a 16-bit, left or right shift in a single cycle.

# PIC24H

FIGURE 2-1: PIC24H CPU CORE BLOCK DIAGRAM



**FIGURE 2-2: PIC24H PROGRAMMER'S MODEL**



# PIC24H

## 2.3 CPU Control Registers

### REGISTER 2-1: SR: CPU STATUS REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
—	—	—	—	—	—	—	DC
bit 15							bit 8

R/W-0 <sup>(1)</sup>	R/W-0 <sup>(2)</sup>	R/W-0 <sup>(2)</sup>	R-0	R/W-0	R/W-0	R/W-0	R/W-0
IPL<2:0> <sup>(2)</sup>			RA	N	OV	Z	C
bit 7							bit 0

#### Legend:

C = Clear only bit	R = Readable bit	U = Unimplemented bit, read as '0'
S = Set only bit	W = Writable bit	-n = Value at POR
'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

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

bit 8 **DC:** MCU 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 low-order bit (for byte sized data) or 8th low-order bit (for word sized data) of the result occurred

bit 7-5 **IPL<2:0>:** CPU Interrupt Priority Level Status bits<sup>(2)</sup>

- 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)

bit 4 **RA:** REPEAT Loop Active bit

- 1 = REPEAT loop in progress
- 0 = REPEAT loop not in progress

bit 3 **N:** MCU ALU Negative bit

- 1 = Result was negative
- 0 = Result was non-negative (zero or positive)

bit 2 **OV:** MCU ALU Overflow bit

- This bit is used for signed arithmetic (2's complement). It indicates an overflow of the magnitude which causes the sign bit to change state.
- 1 = Overflow occurred for signed arithmetic (in this arithmetic operation)
- 0 = No overflow occurred

bit 1 **Z:** MCU ALU Zero bit

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

**Note 1:** The IPL<2:0> bits are concatenated with the IPL<3> bit (CORCON<3>) to form the CPU Interrupt Priority Level. The value in parentheses indicates the IPL if IPL<3> = 1. User interrupts are disabled when IPL<3> = 1.

**2:** The IPL<2:0> Status bits are read only when NSTDIS = 1 (INTCON1<15>).

## REGISTER 2-1: SR: CPU STATUS REGISTER (CONTINUED)

bit 0            **C:** MCU ALU Carry/Borrow bit

1 = A carry-out from the Most Significant bit (MSb) of the result occurred

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

**Note 1:** The IPL<2:0> bits are concatenated with the IPL<3> bit (CORCON<3>) to form the CPU Interrupt Priority Level. The value in parentheses indicates the IPL if IPL<3> = 1. User interrupts are disabled when IPL<3> = 1.

**2:** The IPL<2:0> Status bits are read only when NSTDIS = 1 (INTCON1<15>).

# PIC24H

## REGISTER 2-2: CORCON: CORE CONTROL REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15						bit 8	

U-0	U-0	U-0	U-0	R/C-0	R/W-0	U-0	U-0
—	—	—	—	IPL3 <sup>(1)</sup>	PSV	—	—
bit 7						bit 0	

<b>Legend:</b>	C = Clear only bit		
R = Readable bit	W = Writable bit	-n = Value at POR	'1' = Bit is set
0' = Bit is cleared	'x' = Bit is unknown	U = Unimplemented bit, read as '0'	

- bit 15-4     **Unimplemented:** Read as '0'
- bit 3       **IPL3:** CPU Interrupt Priority Level Status bit 3<sup>(1)</sup>
  - 1 = CPU interrupt priority level is greater than 7
  - 0 = CPU interrupt priority level is 7 or less
- 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'

**Note 1:** The IPL3 bit is concatenated with the IPL<2:0> bits (SR<7:5>) to form the CPU interrupt priority level.



## 2.4 Arithmetic Logic Unit (ALU)

The PIC24H 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.

Refer to the “*dsPIC30F/33F Programmer's Reference Manual*” (DS70157) for information on the SR bits affected by each instruction.

The PIC24H CPU incorporates hardware support for both multiplication and division. This includes a dedicated hardware multiplier and support hardware for 16-bit divisor division.

### 2.4.1 MULTIPLIER

Using the high-speed 17-bit x 17-bit multiplier, the ALU 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

### 2.4.2 DIVIDER

The divide block supports 32-bit/16-bit and 16-bit/16-bit signed and unsigned integer divide operations 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. Sixteen-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.

### 2.4.3 MULTI-BIT DATA SHIFTER

The multi-bit data shifter is capable of performing up to 16-bit arithmetic or logic right shifts, or up to 16-bit left shifts in a single cycle. The source can be either a working register or a memory location.

The shifter requires a signed binary value to determine both the magnitude (number of bits) and direction of the shift operation. A positive value shifts the operand right. A negative value shifts the operand left. A value of '0' does not modify the operand.

# PIC24H

---

NOTES:

## 3.0 MEMORY ORGANIZATION

**Note:** This data sheet summarizes the features of this group of PIC24H devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the “dsPIC30F Family Reference Manual” (DS70046).

The PIC24H architecture features separate program and data memory spaces and buses. This architecture also allows the direct access of program memory from the data space during code execution.

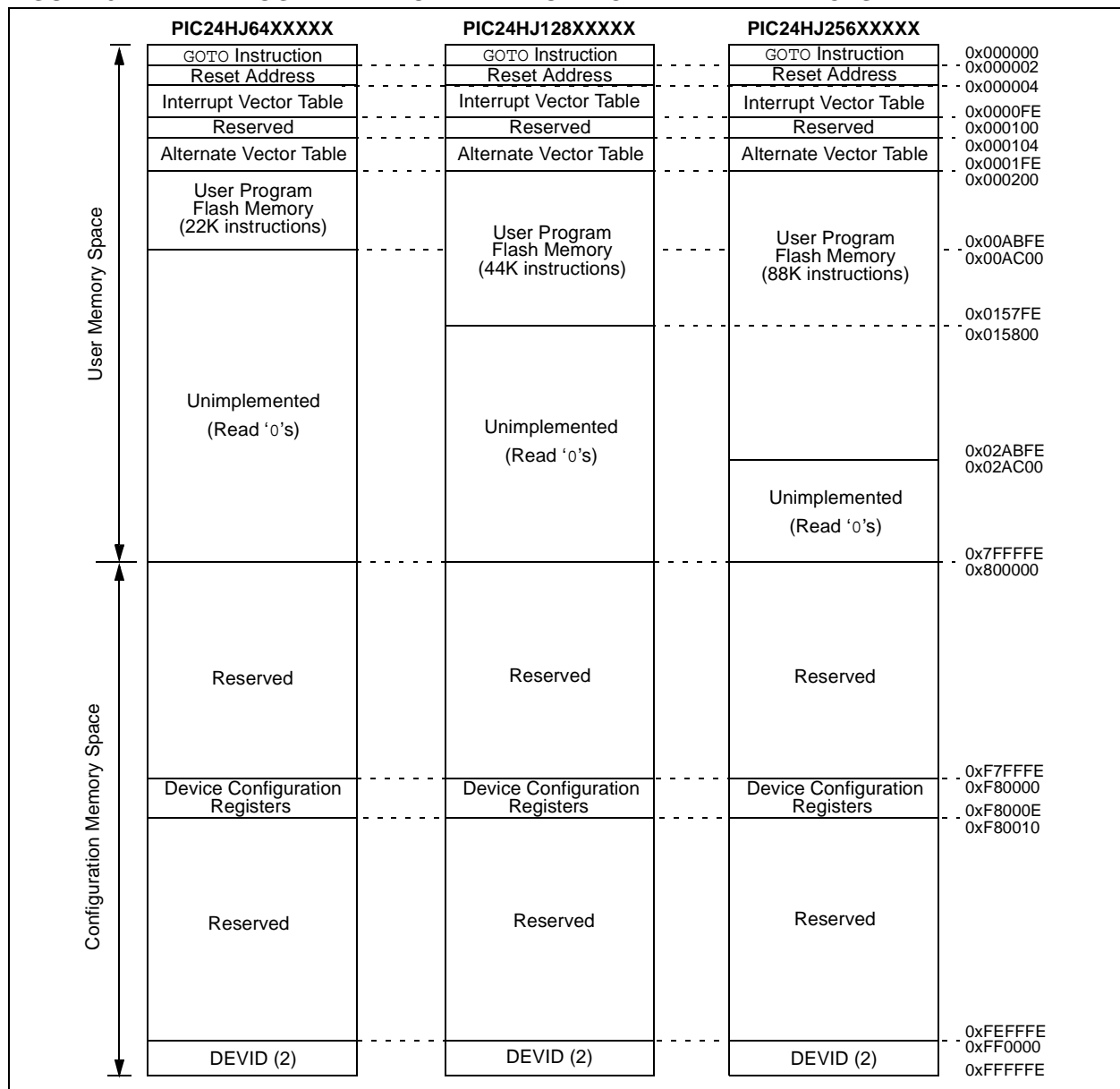
## 3.1 Program Address Space

The program address memory space of the PIC24H 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 3.4 “Interfacing Program and Data Memory Spaces”**.

User access to the program memory space is restricted to the lower half of the address range (0x000000 to 0x7FFFFFFF). 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.

Memory maps for the PIC24H family of devices are shown in Figure 3-1.

**FIGURE 3-1: PROGRAM MEMORY MAP FOR PIC24H FAMILY DEVICES**



# PIC24H

## 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).

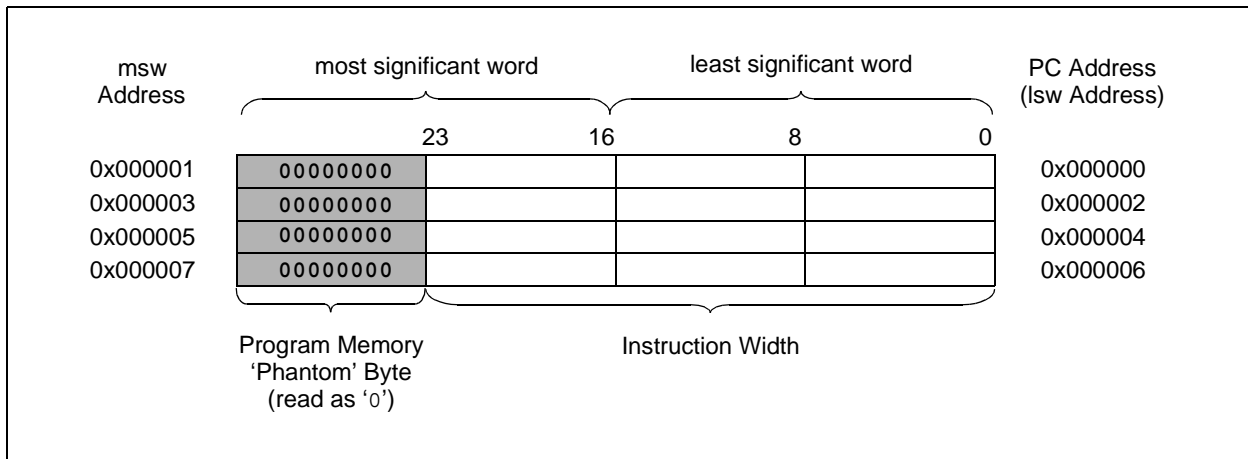
Program memory addresses are always word-aligned on the lower word, and addresses are incremented or decremented by two during code execution. This arrangement also provides compatibility with data memory space addressing and makes it possible to access data in the program memory space.

## 3.1.2 INTERRUPT AND TRAP VECTORS

All PIC24H devices reserve the addresses between 0x00000 and 0x000200 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 0x000000, with the actual address for the start of code at 0x000002.

PIC24H devices also have two interrupt vector tables, located from 0x000004 to 0x0000FF and 0x000100 to 0x0001FF. These vector tables allow each of the many device interrupt sources to be handled by separate Interrupt Service Routines (ISRs). A more detailed discussion of the interrupt vector tables is provided in **Section 6.1 “Interrupt Vector Table”**.

**FIGURE 3-2: PROGRAM MEMORY ORGANIZATION**



## 3.2 Data Address Space

The PIC24H CPU has a separate 16-bit wide data memory space. The data space is accessed using separate Address Generation Units (AGUs) for read and write operations. Data memory maps of devices with different RAM sizes are shown in Figure 3-3 and Figure 3-4.

All Effective Addresses (EAs) in the data memory space are 16 bits wide and point to bytes within the data space. This arrangement gives a data space address range of 64 Kbytes or 32K words. The lower half of the data memory space (that is, when  $EA_{\langle 15 \rangle} = 0$ ) is used for implemented memory addresses, while the upper half ( $EA_{\langle 15 \rangle} = 1$ ) is reserved for the Program Space Visibility area (see **Section 3.4.3 “Reading Data From Program Memory Using Program Space Visibility”**).

PIC24H devices implement up to 16 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.

### 3.2.2 DATA MEMORY ORGANIZATION AND ALIGNMENT

To maintain backward compatibility with PICmicro® MCU devices and improve data space memory usage efficiency, the PIC24H 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 that contains the byte, using the Least Significant bit (LSb) of any EA to determine which byte to select. The selected byte is placed onto the Least Significant Byte (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 is 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 does 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 (MSB) 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 Most Significant Byte of any W register by executing a zero-extend ( $ZE$ ) instruction on the appropriate address.

### 3.2.3 SFR SPACE

The first 2 Kbytes of the Near Data Space, from 0x0000 to 0x07FF, is primarily occupied by Special Function Registers (SFRs). These are used by the PIC24H 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 complete listing of implemented SFRs, including their addresses, is shown in Table 3-1 through Table 3-31.

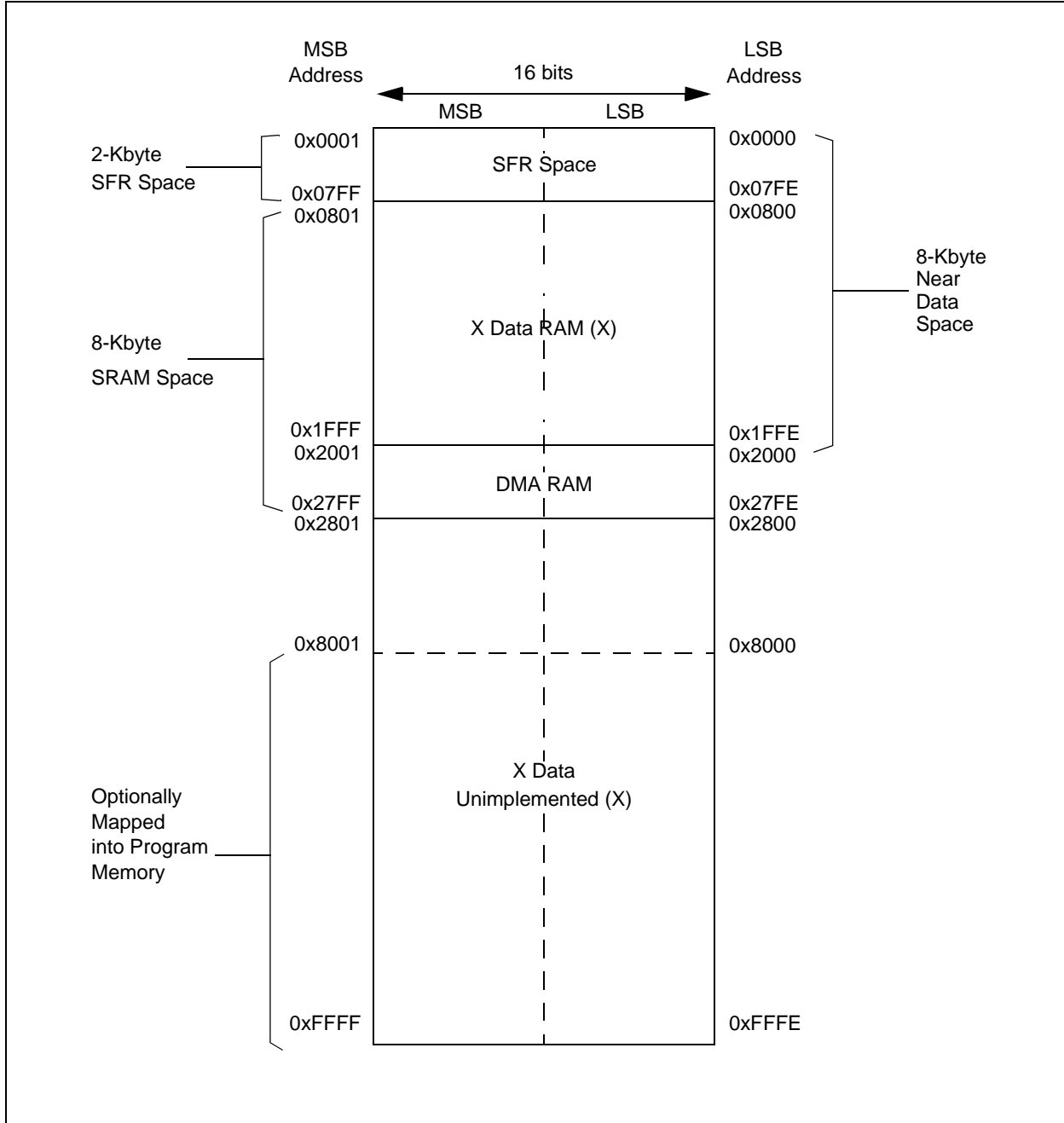
**Note:** The actual set of peripheral features and interrupts varies by the device. Please refer to the corresponding device tables and pinout diagrams for device-specific information.

### 3.2.4 NEAR DATA SPACE

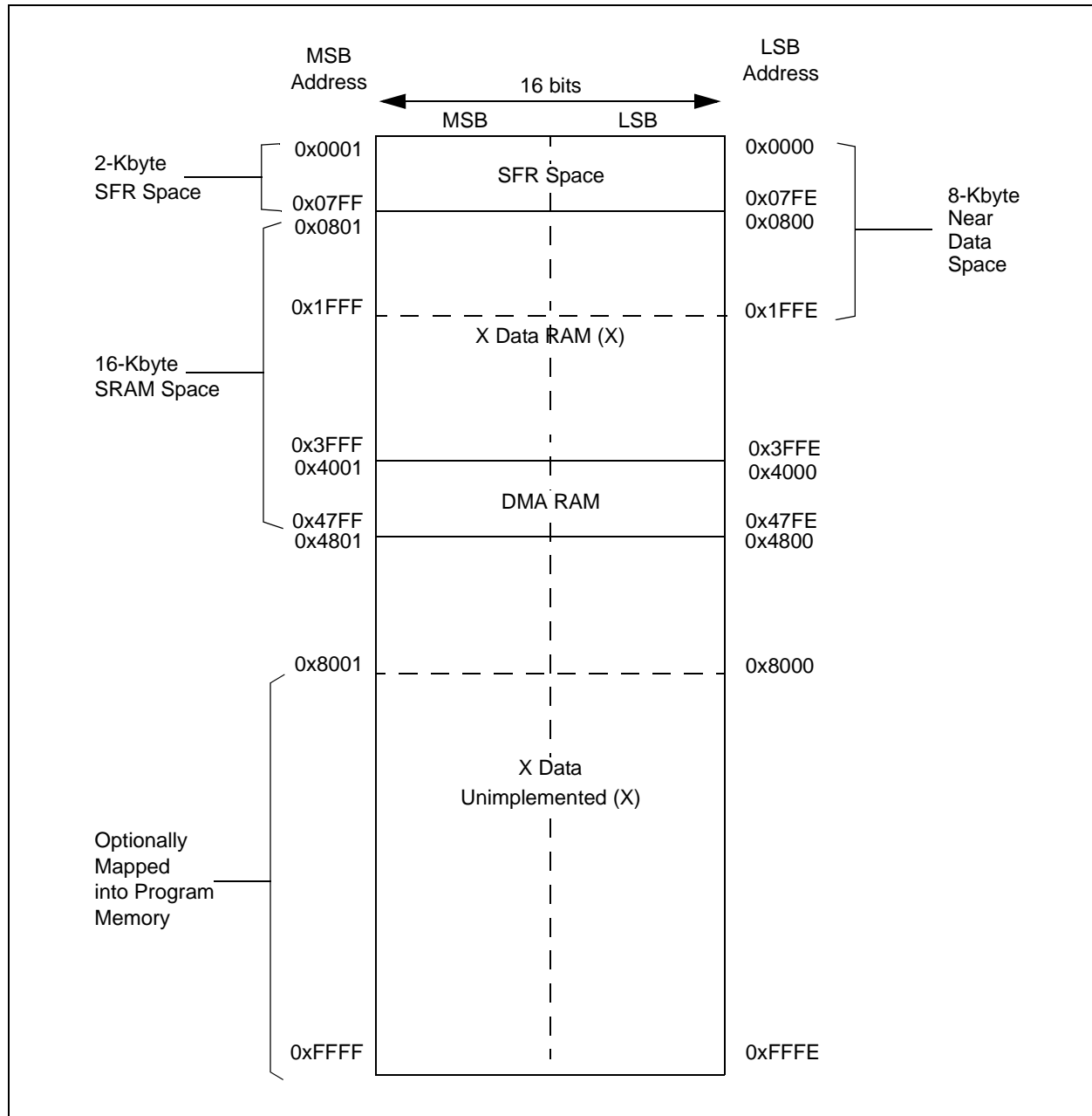
The 8-Kbyte area between 0x0000 and 0x1FFF is referred to as the Near Data Space. Locations in this space are directly addressable via a 13-bit absolute address field within all memory direct instructions. Additionally, the whole data space is addressable using  $MOV$  instructions, which support Memory Direct Addressing mode with a 16-bit address field, or by using Indirect Addressing mode using a working register as an Address Pointer.

# PIC24H

FIGURE 3-3: DATA MEMORY MAP FOR PIC24H DEVICES WITH 8 KBYTES RAM



**FIGURE 3-4: DATA MEMORY MAP FOR PIC24H DEVICES WITH 16 KBYTES RAM**



### 3.2.5 DMA RAM

Every PIC24H device contains 2 Kbytes of dual ported DMA RAM located at the end of data space. Memory locations in the DMA RAM space are accessible simultaneously by the CPU and the DMA controller module. DMA RAM is utilized by the DMA controller to store data to be transferred to various peripherals using DMA, as well as data transferred from various

peripherals using DMA. The DMA RAM can be accessed by the DMA controller without having to steal cycles from the CPU.

When the CPU and the DMA controller attempt to concurrently write to the same DMA RAM location, the hardware ensures that the CPU is given precedence in accessing the DMA RAM location. Therefore, the DMA RAM provides a reliable means of transferring DMA data without ever having to stall the CPU.

**Note:** DMA RAM can be used for general purpose data storage if the DMA function is not required in an application.

**TABLE 3-1: CPU CORE REGISTERS MAP**

SFR Name	SFR 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 Register																xxxxx
PCL	002E	Program Counter Low Word Register																0000
PCH	0030	Program Counter High Byte Register																0000
TBLPAG	0032	Table Page Address Pointer Register																0000
PSV/PAG	0034	Program Memory Visibility Page Address Pointer Register																0000
RCOUNT	0036	Repeat Loop Counter Register																xxxxx
SR	0042	Disable Interrupts Counter Register																0000
CORCON	0044	Disable Interrupts Counter Register																0000
DISICNT	0052	Disable Interrupts Counter Register																xxxxx

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

**TABLE 3-2: CHANGE NOTIFICATION REGISTER MAP**

SFR Name	SFR 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	—	—	—	—	—	—	—	—	CN23IE	CN22IE	CN21IE	CN20IE	CN19IE	CN18IE	CN17IE	CN16IE	0000
CNP1U	0068	CN15PUE	CN14PUE	CN13PUE	CN12PUE	CN11PUE	CN10PUE	CN9PUE	CN8PUE	CN7PUE	CN6PUE	CN5PUE	CN4PUE	CN3PUE	CN2PUE	CN1PUE	CN0PUE	0000
CNP2U	006A	—	—	—	—	—	—	—	—	CN23PUE	CN22PUE	CN21PUE	CN20PUE	CN19PUE	CN18PUE	CN17PUE	CN16PUE	0000

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



**TABLE 3-3: INTERRUPT CONTROLLER REGISTER MAP**

SFR Name	SFR 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	OVAERR	OVBERR	COVAERR	COVBERR	OVATE	OVBTE	COVTE	SFTACERR	DIV0ERR	DMACERR	MATHERR	ADDRERR	STKERR	OSCFAIL	—	0000
INTCON2	0082	ALTVT	DISI	—	—	—	—	—	—	—	—	—	INT4EP	INT3EP	INT2EP	INT1EP	INT0EP	0000
IFS0	0084	—	DMA1IF	AD1IF	U1TXIF	U1RXIF	SPI1EIF	SPI1EIF	T3IF	T2IF	OC2IF	IC2IF	DMA0IF	T1IF	OC1IF	IC1IF	INT0IF	0000
IFS1	0086	U2TXIF	U2RXIF	INT2IF	T5IF	T4IF	OC4IF	OC3IF	DMA2IF	IC8IF	IC7IF	AD2IF	INT1IF	CNIF	—	MIC2C1IF	SIZC1IF	0000
IFS2	0088	T6IF	DMA4IF	—	OC8IF	OC7IF	OC6IF	OC5IF	IC6IF	C2RXIF	IC4IF	IC3IF	DMA3IF	C1IF	C1RXIF	SPI2IF	SPI2EIF	0000
IFS3	008A	—	—	DMA5IF	—	—	—	—	C2IF	C2TXIF	INT4IF	INT3IF	T9IF	T8IF	MIC2C2IF	SIZC2IF	T7IF	0000
IFS4	008C	—	—	—	—	—	—	—	—	C2TXIF	C1TXIF	DMA7IF	DMA6IF	—	U2EIF	U1EIF	—	0000
IEC0	0094	—	DMA1IE	AD1IE	U1TXIE	U1RXIE	SPI1IE	SPI1IE	T3IE	T2IE	OC2IE	IC2IE	DMA0IE	T1IE	OC1IE	IC1IE	INT0IE	0000
IEC1	0096	U2TXIE	U2RXIE	INT2IE	T5IE	T4IE	OC4IE	OC3IE	DMA2IE	IC8IE	IC7IE	AD2IE	INT1IE	CNIE	—	MIC2C1IE	SIZC1IE	0000
IEC2	0098	T6IE	DMA4IE	—	OC8IE	OC7IE	OC6IE	OC5IE	IC6IE	C2RXIE	IC4IE	IC3IE	DMA3IE	C1IE	C1RXIE	SPI2IE	SPI2EIE	0000
IEC3	009A	—	—	DMA5IE	—	—	—	—	C2IE	C2TXIE	INT4IE	INT3IE	T9IE	T8IE	MIC2C2IE	SIZC2IE	T7IE	0000
IEC4	009C	—	—	—	—	—	—	—	—	C2TXIE	C1TXIE	DMA7IE	DMA6IE	—	U2EIE	U1EIE	—	0000
IPC0	00A4	—	—	T1IP<2:0>	—	—	OC1IP<2:0>	—	—	—	—	IC1IP<2:0>	—	—	INT0IP<2:0>	—	—	4444
IPC1	00A6	—	—	T2IP<2:0>	—	—	OC2IP<2:0>	—	—	—	—	IC2IP<2:0>	—	—	DMA0IP<2:0>	—	—	4444
IPC2	00A8	—	—	U1RXIP<2:0>	—	—	SPI1IP<2:0>	—	—	—	—	SPI1EIP<2:0>	—	—	T3IP<2:0>	—	—	4444
IPC3	00AA	—	—	—	—	—	DMA1IP<2:0>	—	—	—	—	AD1IP<2:0>	—	—	U1TXIP<2:0>	—	—	0444
IPC4	00AC	—	—	—	—	—	—	—	—	—	—	MIC2C1IP<2:0>	—	—	SIZC1IP<2:0>	—	—	0044
IPC5	00AE	—	—	IC8IP<2:0>	—	—	IC7IP<2:0>	—	—	—	—	AD2IP<2:0>	—	—	INT1IP<2:0>	—	—	4444
IPC6	00B0	—	—	T4IP<2:0>	—	—	OC4IP<2:0>	—	—	—	—	OC3IP<2:0>	—	—	DMA2IP<2:0>	—	—	4444
IPC7	00B2	—	—	U2TXIP<2:0>	—	—	U2RXIP<2:0>	—	—	—	—	INT2IP<2:0>	—	—	T5IP<2:0>	—	—	4444
IPC8	00B4	—	—	C1IP<2:0>	—	—	C1RXIP<2:0>	—	—	—	—	SPI2IP<2:0>	—	—	SPI2EIP<2:0>	—	—	4444
IPC9	00B6	—	—	IC5IP<2:0>	—	—	IC4IP<2:0>	—	—	—	—	IC3IP<2:0>	—	—	DMA3IP<2:0>	—	—	4444
IPC10	00B8	—	—	OC7IP<2:0>	—	—	OC6IP<2:0>	—	—	—	—	OC5IP<2:0>	—	—	IC6IP<2:0>	—	—	4444
IPC11	00BA	—	—	T6IP<2:0>	—	—	DMA4IP<2:0>	—	—	—	—	—	—	—	OC8IP<2:0>	—	—	4404
IPC12	00BC	—	—	T8IP<2:0>	—	—	MIC2C2IP<2:0>	—	—	—	—	SIZC2IP<2:0>	—	—	T7IP<2:0>	—	—	4444
IPC13	00BE	—	—	C2RXIP<2:0>	—	—	INT4IP<2:0>	—	—	—	—	INT3IP<2:0>	—	—	T9IP<2:0>	—	—	4444
IPC14	00C0	—	—	—	—	—	—	—	—	—	—	—	—	—	C2IP<2:0>	—	—	0004
IPC15	00C2	—	—	—	—	—	—	—	—	—	—	DMA5IP<2:0>	—	—	—	—	—	0040
IPC16	00C4	—	—	—	—	—	U2EIP<2:0>	—	—	—	—	U1EIP<2:0>	—	—	—	—	—	4440
IPC17	00C6	—	—	C2TXIP<2:0>	—	—	C1TXIP<2:0>	—	—	—	—	DMA7IP<2:0>	—	—	DMA6IP<2:0>	—	—	4444

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

**TABLE 3-4: TIMER REGISTER MAP**

SFR Name	SFR 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																	x000x
PR1	0102	Period Register 1																	FFFF
T1CON	0104	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS<1:0>	—	—	TSYNC	TCS	—	0000	
TMR2	0106	Timer2 Register																	x000x
TMR3HLD	0108	Timer3 Holding Register (for 32-bit timer operations only)																	x000x
TMR3	010A	Timer3 Register																	x000x
PR2	010C	Period Register 2																	FFFF
PR3	010E	Period Register 3																	FFFF
T2CON	0110	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS<1:0>	—	T32	—	TCS	—	0000	
T3CON	0112	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS<1:0>	—	—	—	TCS	—	0000	
TMR4	0114	Timer4 Register																	x000x
TMR5HLD	0116	Timer5 Holding Register (for 32-bit operations only)																	x000x
TMR5	0118	Timer5 Register																	x000x
PR4	011A	Period Register 4																	FFFF
PR5	011C	Period Register 5																	FFFF
T4CON	011E	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS<1:0>	—	T32	—	TCS	—	0000	
T5CON	0120	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS<1:0>	—	—	—	TCS	—	0000	
TMR6	0122	Timer6 Register																	x000x
TMR7HLD	0124	Timer7 Holding Register (for 32-bit operations only)																	x000x
TMR7	0126	Timer7 Register																	x000x
PR6	0128	Period Register 6																	FFFF
PR7	012A	Period Register 7																	FFFF
T6CON	012C	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS<1:0>	—	T32	—	TCS	—	0000	
T7CON	012E	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS<1:0>	—	—	—	TCS	—	0000	
TMR8	0130	Timer8 Register																	x000x
TMR9HLD	0132	Timer9 Holding Register (for 32-bit operations only)																	x000x
TMR9	0134	Timer9 Register																	x000x
PR8	0136	Period Register 8																	FFFF
PR9	0138	Period Register 9																	FFFF
T8CON	013A	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS<1:0>	—	T32	—	TCS	—	0000	
T9CON	013C	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS<1:0>	—	—	—	TCS	—	0000	

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

**TABLE 3-5: INPUT CAPTURE REGISTER MAP**

SFR Name	SFR 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										xxxxx
IC1CON	0142	—	—	ICSIDL	—	—	—	—	—	ICTMR	IC1<1:0>	ICOV	ICBNE	ICBNE	ICM<2:0>	ICM<2:0>	0000	0000	
IC2BUF	0144	—	—	—	—	—	—	—	Input 2 Capture Register										xxxxx
IC2CON	0146	—	—	ICSIDL	—	—	—	—	—	ICTMR	IC1<1:0>	ICOV	ICBNE	ICBNE	ICM<2:0>	ICM<2:0>	0000	0000	
IC3BUF	0148	—	—	—	—	—	—	—	Input 3 Capture Register										xxxxx
IC3CON	014A	—	—	ICSIDL	—	—	—	—	—	ICTMR	IC1<1:0>	ICOV	ICBNE	ICBNE	ICM<2:0>	ICM<2:0>	0000	0000	
IC4BUF	014C	—	—	—	—	—	—	—	Input 4 Capture Register										xxxxx
IC4CON	014E	—	—	ICSIDL	—	—	—	—	—	ICTMR	IC1<1:0>	ICOV	ICBNE	ICBNE	ICM<2:0>	ICM<2:0>	0000	0000	
IC5BUF	0150	—	—	—	—	—	—	—	Input 5 Capture Register										xxxxx
IC5CON	0152	—	—	ICSIDL	—	—	—	—	—	ICTMR	IC1<1:0>	ICOV	ICBNE	ICBNE	ICM<2:0>	ICM<2:0>	0000	0000	
IC6BUF	0154	—	—	—	—	—	—	—	Input 6 Capture Register										xxxxx
IC6CON	0156	—	—	ICSIDL	—	—	—	—	—	ICTMR	IC1<1:0>	ICOV	ICBNE	ICBNE	ICM<2:0>	ICM<2:0>	0000	0000	
IC7BUF	0158	—	—	—	—	—	—	—	Input 7 Capture Register										xxxxx
IC7CON	015A	—	—	ICSIDL	—	—	—	—	—	ICTMR	IC1<1:0>	ICOV	ICBNE	ICBNE	ICM<2:0>	ICM<2:0>	0000	0000	
IC8BUF	015C	—	—	—	—	—	—	—	Input 8 Capture Register										xxxxx
IC8CON	015E	—	—	ICSIDL	—	—	—	—	—	ICTMR	IC1<1:0>	ICOV	ICBNE	ICBNE	ICM<2:0>	ICM<2:0>	0000	0000	

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

**TABLE 3-6: OUTPUT COMPARE REGISTER MAP**

SFR Name	SFR 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																	xxxxx
OC1R	0182	Output Compare 1 Register																	xxxxx
OC1CON	0184	—	—	OCSIDL	—	—	—	—	—	—	—	—	OCFLT	OCTSEL	OCM<2:0>			0000	
OC2RS	0186	Output Compare 2 Secondary Register																	xxxxx
OC2R	0188	Output Compare 2 Register																	xxxxx
OC2CON	018A	—	—	OCSIDL	—	—	—	—	—	—	—	—	OCFLT	OCTSEL	OCM<2:0>			0000	
OC3RS	018C	Output Compare 3 Secondary Register																	xxxxx
OC3R	018E	Output Compare 3 Register																	xxxxx
OC3CON	0190	—	—	OCSIDL	—	—	—	—	—	—	—	—	OCFLT	OCTSEL	OCM<2:0>			0000	
OC4RS	0192	Output Compare 4 Secondary Register																	xxxxx
OC4R	0194	Output Compare 4 Register																	xxxxx
OC4CON	0196	—	—	OCSIDL	—	—	—	—	—	—	—	—	OCFLT	OCTSEL	OCM<2:0>			0000	
OC5RS	0198	Output Compare 5 Secondary Register																	xxxxx
OC5R	019A	Output Compare 5 Register																	xxxxx
OC5CON	019C	—	—	OCSIDL	—	—	—	—	—	—	—	—	OCFLT	OCTSEL	OCM<2:0>			0000	
OC6RS	019E	Output Compare 6 Secondary Register																	xxxxx
OC6R	01A0	Output Compare 6 Register																	xxxxx
OC6CON	01A2	—	—	OCSIDL	—	—	—	—	—	—	—	—	OCFLT	OCTSEL	OCM<2:0>			0000	
OC7RS	01A4	Output Compare 7 Secondary Register																	xxxxx
OC7R	01A6	Output Compare 7 Register																	xxxxx
OC7CON	01A8	—	—	OCSIDL	—	—	—	—	—	—	—	—	OCFLT	OCTSEL	OCM<2:0>			0000	
OC8RS	01AA	Output Compare 8 Secondary Register																	xxxxx
OC8R	01AC	Output Compare 8 Register																	xxxxx
OC8CON	01AE	—	—	OCSIDL	—	—	—	—	—	—	—	—	OCFLT	OCTSEL	OCM<2:0>			0000	

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

**TABLE 3-7: I2C1 REGISTER MAP**

SFR Name	SFR 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 Register				0000	
I2C1ON	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_W	RBF	TBF	0000
I2C1ADD	020A	—	—	—	—	—	—	—	—	—	—	—	Address Register				0000	
I2C1MSK	020C	—	—	—	—	—	—	—	—	—	—	—	Address Mask Register				0000	

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

**TABLE 3-8: I2C2 REGISTER MAP**

SFR Name	SFR 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 Register				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	I2COV	D_A	P	S	R_W	RBF	TBF	0000
I2C2ADD	021A	—	—	—	—	—	—	—	—	—	—	—	Address Register				0000	
I2C2MSK	021C	—	—	—	—	—	—	—	—	—	—	—	Address Mask Register				0000	

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

**TABLE 3-9: UART1 REGISTER MAP**

SFR Name	SFR 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	URXINV	BRGH	PDSEL<1:0>	—	STSEL	0000
U1STA	0222	UTXISEL1	UTXINV	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT	URXISEL<1:0>	—	ADDEN	RIDLE	PERR	FERR	OERR	URXDA	0110
U1TXREG	0224	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	xxxxx
U1RXREG	0226	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
U1BRG	0228	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000

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

Baud Rate Generator Prescaler

UART Transmit Register  
UART Receive Register

**TABLE 3-10: UART2 REGISTER MAP**

SFR Name	SFR 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	URXINV	BRGH	PDSEL<1:0>	—	STSEL	0000
U2STA	0232	UTXISEL1	UTXINV	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT	URXISEL<1:0>	—	ADDEN	RIDLE	PERR	FERR	OERR	URXDA	0110
U2TXREG	0234	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	xxxxx
U2RXREG	0236	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
U2BRG	0238	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000

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

Baud Rate Generator Prescaler

UART Transmit Register  
UART Receive Register

**TABLE 3-11: SPI1 REGISTER MAP**

SFR Name	SFR 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	—	SPIIDL	—	—	—	—	—	—	SPIROV	—	—	—	—	SPI1BF	SPI1RF	0000
SPI1CON1	0242	—	—	—	DISSCK	DISSDO	MODE16	SMP	CKE	SSEN	CKP	MSTEN	—	SPRE<2:0>	—	PPRE<1:0>	—	0000
SPI1CON2	0244	FRMEN	SPIFSD	FRMPOL	—	—	—	—	—	—	—	—	—	—	FRMDLY	—	—	0000
SPI1BUF	0248	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000

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

SPI1 Transmit and Receive Buffer Register

**TABLE 3-12: SPI2 REGISTER MAP**

SFR Name	SFR 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	—	SPIIDL	—	—	—	—	—	—	SPIROV	—	—	—	—	SPI1BF	SPI1RF	0000
SPI2CON1	0262	—	—	—	DISSCK	DISSDO	MODE16	SMP	CKE	SSEN	CKP	MSTEN	—	SPRE<2:0>	—	PPRE<1:0>	—	0000
SPI2CON2	0264	FRMEN	SPIFSD	FRMPOL	—	—	—	—	—	—	—	—	—	—	FRMDLY	—	—	0000
SPI2BUF	0268	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000

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

SPI2 Transmit and Receive Buffer Register

**TABLE 3-13: ADC1 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
ADC Data Buffer 0																		
ADC1BUF0	0300																	xxxx
AD1CON1	0320	ADON	—	ADSIDL	ADDMABM	—	AD12B	FORM<1:0>	—	SSRC<2:0>	—	—	—	SIMSAM	ASAM	SAMP	DONE	0000
AD1CON2	0322	—	VCFG<2:0>	—	—	—	CSCNA	CHPS<1:0>	—	BUFS	—	—	SMPI<3:0>	—	—	BUFM	ALTS	0000
AD1CON3	0324	ADRC	—	—	—	—	SAMC<4:0>	—	—	—	—	—	—	ADCS<5:0>	—	—	—	0000
AD1CHS123	0326	—	—	—	—	—	CH123NB<1:0>	CH123SB	—	—	—	—	—	—	CH123NA<1:0>	—	CH123SA	0000
AD1CHS0	0328	CH0NB	—	—	—	—	CH0SB<4:0>	—	CH0NA	—	—	—	—	—	CH0SA<4:0>	—	—	0000
AD1PCFGH	032A	PCFG31	PCFG30	PCFG29	PCFG28	PCFG27	PCFG26	PCFG25	PCFG24	PCFG23	PCFG22	PCFG21	PCFG20	PCFG19	PCFG18	PCFG17	PCFG16	0000
AD1PCFGL	032C	PCFG15	PCFG14	PCFG13	PCFG12	PCFG11	PCFG10	PCFG9	PCFG8	PCFG7	PCFG6	PCFG5	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0	0000
AD1CSSH	032E	CSS31	CSS30	CSS29	CSS28	CSS27	CSS26	CSS25	CSS24	CSS23	CSS22	CSS21	CSS20	CSS19	CSS18	CSS17	CSS16	0000
AD1CSSL	0330	CSS15	CSS14	CSS13	CSS12	CSS11	CSS10	CSS9	CSS8	CSS7	CSS6	CSS5	CSS4	CSS3	CSS2	CSS1	CSS0	0000
AD1CON4	0332	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	DMABL<2:0>	0000
Reserved	0334-033E	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000

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

**TABLE 3-14: ADC2 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
ADC Data Buffer 0																		
ADC2BUF0	0340																	xxxx
AD2CON1	0360	ADON	—	ADSIDL	ADDMABM	—	AD12B	FORM<1:0>	—	SSRC<2:0>	—	—	—	SIMSAM	ASAM	SAMP	DONE	0000
AD2CON2	0362	—	VCFG<2:0>	—	—	—	CSCNA	CHPS<1:0>	—	BUFS	—	—	—	SMPI<3:0>	—	BUFM	ALTS	0000
AD2CON3	0364	ADRC	—	—	—	—	SAMC<4:0>	—	—	—	—	—	—	ADCS<5:0>	—	—	—	0000
AD2CHS123	0366	—	—	—	—	—	CH123NB<1:0>	CH123SB	—	—	—	—	—	—	CH123NA<1:0>	—	CH123SA	0000
AD2CHS0	0368	CH0NB	—	—	—	—	CH0SB<3:0>	—	CH0NA	—	—	—	—	—	CH0SA<3:0>	—	—	0000
Reserved	036A	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
AD2PCFGL	036C	PCFG15	PCFG14	PCFG13	PCFG12	PCFG11	PCFG10	PCFG9	PCFG8	PCFG7	PCFG6	PCFG5	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0	0000
Reserved	036E	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
AD2CSSL	0370	CSS15	CSS14	CSS13	CSS12	CSS11	CSS10	CSS9	CSS8	CSS7	CSS6	CSS5	CSS4	CSS3	CSS2	CSS1	CSS0	0000
AD2CON4	0372	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	DMABL<2:0>	0000
Reserved	0374-037E	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000

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

**TABLE 3-15: DMA 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
DMA0CON	0380	CHEN	SIZE	DIR	HALF	NULLW	—	—	—	—	—	—	—	—	—	—	—	0000
DMA0REQ	0382	FORCE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA0STA	0384	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA0STB	0386	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA0PAD	0388	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA0CNT	038A	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA1CON	038C	CHEN	SIZE	DIR	HALF	NULLW	—	—	—	—	—	—	—	—	—	—	—	0000
DMA1REQ	038E	FORCE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA1STA	0390	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA1STB	0392	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA1PAD	0394	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA1CNT	0396	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA2CON	0398	CHEN	SIZE	DIR	HALF	NULLW	—	—	—	—	—	—	—	—	—	—	—	0000
DMA2REQ	039A	FORCE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA2STA	039C	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA2STB	039E	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA2PAD	03A0	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA2CNT	03A2	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA3CON	03A4	CHEN	SIZE	DIR	HALF	NULLW	—	—	—	—	—	—	—	—	—	—	—	0000
DMA3REQ	03A6	FORCE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA3STA	03A8	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA3STB	03AA	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA3PAD	03AC	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA3CNT	03AE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA4CON	03B0	CHEN	SIZE	DIR	HALF	NULLW	—	—	—	—	—	—	—	—	—	—	—	0000
DMA4REQ	03B2	FORCE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA4STA	03B4	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA4STB	03B6	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA4PAD	03B8	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA4CNT	03BA	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA5CON	03BC	CHEN	SIZE	DIR	HALF	NULLW	—	—	—	—	—	—	—	—	—	—	—	0000
DMA5REQ	03BE	FORCE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA5STA	03C0	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
DMA5STB	03C2	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000

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



**TABLE 3-15: DMA REGISTER MAP (CONTINUED)**

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	
DMA5PAD	03C4	PAD<15:0>																	0000
DMA5CNT	03C6	CNT<9:0>																	0000
DMA6CON	03C8	CHEN	SIZE	DIR	HALF	NULLW	—	—	—	—	—	—	—	—	—	—	MODE<1:0>	0000	
DMA6REQ	03CA	FORCE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	IRQSEL<6:0>	0000	
DMA6STA	03CC	STA<15:0>																	0000
DMA6STB	03CE	STB<15:0>																	0000
DMA6PAD	03D0	PAD<15:0>																	0000
DMA6CNT	03D2	CNT<9:0>																	0000
DMA7CON	03D4	CHEN	SIZE	DIR	HALF	NULLW	—	—	—	—	—	—	—	—	—	—	MODE<1:0>	0000	
DMA7REQ	03D6	FORCE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	IRQSEL<6:0>	0000	
DMA7STA	03D8	STA<15:0>																	0000
DMA7STB	03DA	STB<15:0>																	0000
DMA7PAD	03DC	PAD<15:0>																	0000
DMA7CNT	03DE	CNT<9:0>																	0000
DMA8S0	03E0	PWCOL7	PWCOL6	PWCOL5	PWCOL4	PWCOL3	PWCOL2	PWCOL1	PWCOL0	XWCOL7	XWCOL6	XWCOL5	XWCOL4	XWCOL3	XWCOL2	XWCOL1	XWCOL0	0000	
DMA8S1	03E2	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000	
DSADR	03E4	DSADR<15:0>																	0000

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

**TABLE 3-16: ECAN1 REGISTER MAP WHEN C1CTRL1.WIN = 0 OR 1**

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	
C1CTRL1	0400	—	—	CSIDL	ABAT	CANCKS	REQOP<2:0>	—	—	—	—	—	—	CANCAP	—	—	WIN	0480	
C1CTRL2	0402	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000	
C1VEC	0404	—	—	—	—	—	FILHIT<4:0>	—	—	—	—	—	—	—	—	—	—	0000	
C1FCTRL	0406	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000	
C1FIFO	0408	—	—	—	—	—	FBP<5:0>	—	—	—	—	—	—	—	—	—	—	0000	
C1INTF	040A	—	—	TXBO	TXBP	RXBP	TXWAR	RXWAR	EWARN	IVRIF	WAKIF	ERRIF	—	FIFOIF	RBOVIF	RBIF	TBIF	0000	
C1INTE	040C	—	—	—	—	—	—	—	—	IVRIE	WAKIE	ERRIE	—	FIFOIE	RBOVIE	RBIE	TBIE	0000	
C1EC	040E	TERRCNT<7:0>																	0000
C1CFG1	0410	—	—	—	—	—	—	—	—	SJW<1:0>	—	—	—	—	—	—	—	0000	
C1CFG2	0412	—	WAKFIL	—	—	—	SEG2PH<2:0>	—	—	SEG2PHTS	SAM	—	—	—	—	—	PRSEG<2:0>	0000	
C1FEN1	0414	FLTEN15	FLTEN14	FLTEN13	FLTEN12	FLTEN11	FLTEN10	FLTEN9	FLTEN8	FLTEN7	FLTEN6	FLTEN5	FLTEN4	FLTEN3	FLTEN2	FLTEN1	FLTEN0	0000	
C1FMSKSEL1	0418	F7MSK<1:0>	F6MSK<1:0>	F5MSK<1:0>	F4MSK<1:0>	F3MSK<1:0>	F2MSK<1:0>	F1MSK<1:0>	—	—	—	—	—	—	—	—	—	0000	
C1FMSKSEL2	041A	F15MSK<1:0>	F14MSK<1:0>	F13MSK<1:0>	F12MSK<1:0>	F11MSK<1:0>	F10MSK<1:0>	F9MSK<1:0>	F8MSK<1:0>	—	—	—	—	—	—	—	—	0000	

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

**TABLE 3-17: ECAN1 REGISTER MAP WHEN C1CTRL1.WIN = 0**

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
	0400-041E	See definition when WIN = x																
C1RXFUL1	0420	RXFUL15	RXFUL14	RXFUL13	RXFUL12	RXFUL11	RXFUL10	RXFUL9	RXFUL8	RXFUL7	RXFUL6	RXFUL5	RXFUL4	RXFUL3	RXFUL2	RXFUL1	RXFUL0	0000
C1RXFUL2	0422	RXFUL31	RXFUL30	RXFUL29	RXFUL28	RXFUL27	RXFUL26	RXFUL25	RXFUL24	RXFUL23	RXFUL22	RXFUL21	RXFUL20	RXFUL19	RXFUL18	RXFUL17	RXFUL16	0000
C1RXOVF1	0428	RXOVF15	RXOVF14	RXOVF13	RXOVF12	RXOVF11	RXOVF10	RXOVF9	RXOVF8	RXOVF7	RXOVF6	RXOVF5	RXOVF4	RXOVF3	RXOVF2	RXOVF1	RXOVF0	0000
C1RXOVF2	042A	RXOVF31	RXOVF30	RXOVF29	RXOVF28	RXOVF27	RXOVF26	RXOVF25	RXOVF24	RXOVF23	RXOVF22	RXOVF21	RXOVF20	RXOVF19	RXOVF18	RXOVF17	RXOVF16	0000
C1TR01CON	0430	TXEN1	TX ABT1	TX LARB1	TX ERR1	TX REQ1	RTREN1	TX1PRI<1:0>	TXEN0	TX ABAT0	TX LARB0	TX ERR0	TX REQ0	TX RTREN0	TX0PRI<1:0>		0000	
C1TR23CON	0432	TXEN3	TX ABT3	TX LARB3	TX ERR3	TX REQ3	RTREN3	TX3PRI<1:0>	TXEN2	TX ABAT2	TX LARB2	TX ERR2	TX REQ2	TX RTREN2	TX2PRI<1:0>		0000	
C1TR45CON	0434	TXEN5	TX ABT5	TX LARB5	TX ERR5	TX REQ5	RTREN5	TX5PRI<1:0>	TXEN4	TX ABAT4	TX LARB4	TX ERR4	TX REQ4	TX RTREN4	TX4PRI<1:0>		0000	
C1TR67CON	0436	TXEN7	TX ABT7	TX LARB7	TX ERR7	TX REQ7	RTREN7	TX7PRI<1:0>	TXEN6	TX ABAT6	TX LARB6	TX ERR6	TX REQ6	TX RTREN6	TX6PRI<1:0>		xxxx	
C1RXD	0440	Received Data Word																
C1TXD	0442	Transmit Data Word																

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

**TABLE 3-18: ECAN1 REGISTER MAP WHEN C1CTRL1.WIN = 1**

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
	0400-041E	See definition when WIN = x																
C1BUFNPNT1	0420		F3BP<3:0>				F2BP<3:0>				F1BP<3:0>				F0BP<3:0>			0000
C1BUFNPNT2	0422		F7BP<3:0>				F6BP<3:0>				F5BP<3:0>				F4BP<3:0>			0000
C1BUFNPNT3	0424		F11BP<3:0>				F10BP<3:0>				F9BP<3:0>				F8BP<3:0>			0000
C1BUFNPNT4	0426		F15BP<3:0>				F14BP<3:0>				F13BP<3:0>				F12BP<3:0>			0000
C1RXM0SID	0430			SID<10:3>							SID<2:0>			MIDE		EID<17:16>		xxxxx
C1RXM0EID	0432			EID<15:8>									EID<7:0>					xxxxx
C1RXM1SID	0434			SID<10:3>							SID<2:0>			MIDE		EID<17:16>		xxxxx
C1RXM1EID	0436			EID<15:8>									EID<7:0>					xxxxx
C1RXM2SID	0438			SID<10:3>							SID<2:0>			MIDE		EID<17:16>		xxxxx
C1RXM2EID	043A			EID<15:8>									EID<7:0>					xxxxx
C1RXF0SID	0440			SID<10:3>							SID<2:0>			EXIDE		EID<17:16>		xxxxx
C1RXF0EID	0442			EID<15:8>									EID<7:0>					xxxxx
C1RXF1SID	0444			SID<10:3>							SID<2:0>			EXIDE		EID<17:16>		xxxxx
C1RXF1EID	0446			EID<15:8>									EID<7:0>					xxxxx
C1RXF2SID	0448			SID<10:3>							SID<2:0>			EXIDE		EID<17:16>		xxxxx
C1RXF2EID	044A			EID<15:8>									EID<7:0>					xxxxx
C1RXF3SID	044C			SID<10:3>							SID<2:0>			EXIDE		EID<17:16>		xxxxx
C1RXF3EID	044E			EID<15:8>									EID<7:0>					xxxxx
C1RXF4SID	0450			SID<10:3>							SID<2:0>			EXIDE		EID<17:16>		xxxxx
C1RXF4EID	0452			EID<15:8>									EID<7:0>					xxxxx
C1RXF5SID	0454			SID<10:3>							SID<2:0>			EXIDE		EID<17:16>		xxxxx
C1RXF5EID	0456			EID<15:8>									EID<7:0>					xxxxx
C1RXF6SID	0458			SID<10:3>							SID<2:0>			EXIDE		EID<17:16>		xxxxx
C1RXF6EID	045A			EID<15:8>									EID<7:0>					xxxxx
C1RXF7SID	045C			SID<10:3>							SID<2:0>			EXIDE		EID<17:16>		xxxxx
C1RXF7EID	045E			EID<15:8>									EID<7:0>					xxxxx
C1RXF8SID	0460			SID<10:3>							SID<2:0>			EXIDE		EID<17:16>		xxxxx
C1RXF8EID	0462			EID<15:8>									EID<7:0>					xxxxx
C1RXF9SID	0464			SID<10:3>							SID<2:0>			EXIDE		EID<17:16>		xxxxx
C1RXF9EID	0466			EID<15:8>									EID<7:0>					xxxxx
C1RXF10SID	0468			SID<10:3>							SID<2:0>			EXIDE		EID<17:16>		xxxxx
C1RXF10EID	046A			EID<15:8>									EID<7:0>					xxxxx

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

**TABLE 3-18: ECAN1 REGISTER MAP WHEN C1CTRL1.WIN = 1 (CONTINUED)**

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
C1RXF11SID	046C				SID<10:3>						SID<2:0>			EXIDE		EID<17:16>		XXXXX
C1RXF11EID	046E				EID<15:8>								EID<7:0>					XXXXX
C1RXF12SID	0470				SID<10:3>						SID<2:0>			EXIDE		EID<17:16>		XXXXX
C1RXF12EID	0472				EID<15:8>								EID<7:0>					XXXXX
C1RXF13SID	0474				SID<10:3>						SID<2:0>			EXIDE		EID<17:16>		XXXXX
C1RXF13EID	0476				EID<15:8>								EID<7:0>					XXXXX
C1RXF14SID	0478				SID<10:3>						SID<2:0>			EXIDE		EID<17:16>		XXXXX
C1RXF14EID	047A				EID<15:8>								EID<7:0>					XXXXX
C1RXF15SID	047C				SID<10:3>						SID<2:0>			EXIDE		EID<17:16>		XXXXX
C1RXF15EID	047E				EID<15:8>								EID<7:0>					XXXXX

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

**TABLE 3-19: ECAN2 REGISTER MAP WHEN C2CTRL1.WIN = 0 OR 1**

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
C2CTRL1	0500	—	—	—	ABAT	CANCKS	REQOP<2:0>				OPMODE<2:0>			CANCAP			WIN	0480
C2CTRL2	0502	—	—	—	—	—	—	—	—	—	—	—	—	—	DNCNT<4:0>			0000
C2VEC	0504	—	—	—	—	—	FILHIT<4:0>				—	—	—	—	—	—	—	0000
C2FCTRL	0506	—	—	—	—	—	—	—	—	—	—	—	—	—	FSA<4:0>			0000
C2FIFO	0508	—	—	—	—	—	FBP<5:0>				—	—	—	—	FNRB<5:0>			0000
C2INTF	050A	—	—	TXBO	TXBP	RXBP	TXWAR	RXWAR	EWARN	IVRIF	WAKIF	ERRIF	—	FIFOIF	RBOVIF	RBIF	TBIF	0000
C2INTE	050C	—	—	—	—	—	—	—	—	IVRIE	WAKIE	ERRIE	—	FIFOIE	RBOVIE	RBIE	TBIE	0000
C2EC	050E	—	—	—	—	—	—	—	—	—	—	—	—	—	RERRCNT<7:0>			0000
C2CFG1	0510	—	—	—	—	—	—	—	—	SJW<1:0>	—	—	—	—	BRP<5:0>			0000
C2CFG2	0512	—	WAKFIL	—	—	—	—	—	—	SEG2PHTS	SAM	—	—	—	—	PRSEG<2:0>		0000
C2FEN1	0514	FLTEN15	FLTEN14	FLTEN13	FLTEN12	FLTEN11	FLTEN10	FLTEN9	FLTEN8	FLTEN7	FLTEN6	FLTEN5	FLTEN4	FLTEN3	FLTEN2	FLTEN1	FLTEN0	0000
C2FMSKSEL1	0518	F7MSK<1:0>	F6MSK<1:0>	F5MSK<1:0>	F4MSK<1:0>	F3MSK<1:0>	F2MSK<1:0>	F1MSK<1:0>	F0MSK<1:0>									0000
C2FMSKSEL2	051A	F15MSK<1:0>	F14MSK<1:0>	F13MSK<1:0>	F12MSK<1:0>	F11MSK<1:0>	F10MSK<1:0>	F9MSK<1:0>	F8MSK<1:0>									0000

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

**TABLE 3-20: ECAN2 REGISTER MAP WHEN C2CTRL1.WIN = 0**

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
	0500-051E	See definition when WIN = x																
C2RXFUL1	0520	RXFUL15	RXFUL14	RXFUL13	RXFUL12	RXFUL11	RXFUL10	RXFUL9	RXFUL8	RXFUL7	RXFUL6	RXFUL5	RXFUL4	RXFUL3	RXFUL2	RXFUL1	RXFUL0	0000
C2RXFUL2	0522	RXFUL31	RXFUL30	RXFUL29	RXFUL28	RXFUL27	RXFUL26	RXFUL25	RXFUL24	RXFUL23	RXFUL22	RXFUL21	RXFUL20	RXFUL19	RXFUL18	RXFUL17	RXFUL16	0000
C2RXOVF1	0528	RXOVF15	RXOVF14	RXOVF13	RXOVF12	RXOVF11	RXOVF10	RXOVF09	RXOVF08	RXOVF7	RXOVF6	RXOVF5	RXOVF4	RXOVF3	RXOVF2	RXOVF1	RXOVF0	0000
C2RXOVF2	052A	RXOVF31	RXOVF30	RXOVF29	RXOVF28	RXOVF27	RXOVF26	RXOVF25	RXOVF24	RXOVF23	RXOVF22	RXOVF21	RXOVF20	RXOVF19	RXOVF18	RXOVF17	RXOVF16	0000
C2TR01CON	0530	TXEN1	TX ABAT1	TX LARB1	TX ERR1	TX REQ1	RTREN1	TX1PRI<1:0>	TXEN0	TXEN0	TX ABAT0	TX LARB0	TX ERR0	TX REQ0	RTREN0	TX0PRI<1:0>	0000	
C2TR23CON	0532	TXEN3	TX ABAT3	TX LARB3	TX ERR3	TX REQ3	RTREN3	TX3PRI<1:0>	TXEN2	TXEN2	TX ABAT2	TX LARB2	TX ERR2	TX REQ2	RTREN2	TX2PRI<1:0>	0000	
C2TR45CON	0534	TXEN5	TX ABAT5	TX LARB5	TX ERR5	TX REQ5	RTREN5	TX5PRI<1:0>	TXEN4	TXEN4	TX ABAT4	TX LARB4	TX ERR4	TX REQ4	RTREN4	TX4PRI<1:0>	0000	
C2TR67CON	0536	TXEN7	TX ABAT7	TX LARB7	TX ERR7	TX REQ7	RTREN7	TX7PRI<1:0>	TXEN6	TXEN6	TX ABAT6	TX LARB6	TX ERR6	TX REQ6	RTREN6	TX6PRI<1:0>	xxxx	
C2RXD	0540	Received Data Word																
C2TXD	0542	Transmit Data Word																

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

**TABLE 3-21: ECAN2 REGISTER MAP WHEN C2CTRL1.WIN = 1**

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
	0500-051E	See definition when WIN = x																
C2BUFPNT1	0520		F3BP<3:0>				F2BP<3:0>				F1BP<3:0>				F0BP<3:0>			0000
C2BUFPNT2	0522		F7BP<3:0>				F6BP<3:0>				F5BP<3:0>				F4BP<3:0>			0000
C2BUFPNT3	0524		F12BP<3:0>				F10BP<3:0>				F9BP<3:0>				F8BP<3:0>			0000
C2BUFPNT4	0526		F15BP<3:0>				F14BP<3:0>				F13BP<3:0>				F12BP<3:0>			0000
C2RXM0SID	0530				SID<10:3>						SID<2:0>			MIDE		EID<17:16>		xxxx
C2RXM0EID	0532				EID<15:8>						EID<7:0>					EID<17:16>		xxxx
C2RXM1SID	0534				SID<10:3>						SID<2:0>			MIDE		EID<17:16>		xxxx
C2RXM1EID	0536				EID<15:8>						EID<7:0>					EID<17:16>		xxxx
C2RXM2SID	0538				SID<10:3>						SID<2:0>			MIDE		EID<17:16>		xxxx
C2RXM2EID	053A				EID<15:8>						EID<7:0>					EID<17:16>		xxxx
C2RXF0SID	0540				SID<10:3>						SID<2:0>			EXIDE		EID<17:16>		xxxx
C2RXF0EID	0542				EID<15:8>						EID<7:0>					EID<17:16>		xxxx

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

**TABLE 3-21: ECAN2 REGISTER MAP WHEN C2CTRL1.WIN = 1 (CONTINUED)**

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
C2RXF1SID	0544			SID<10:3>							SID<2:0>			EXIDE	—	EID<17:16>		xxxxx
C2RXF1EID	0546			EID<15:8>									EID<7:0>					xxxxx
C2RXF2SID	0548			SID<10:3>							SID<2:0>			EXIDE	—	EID<17:16>		xxxxx
C2RXF2EID	054A			EID<15:8>									EID<7:0>					xxxxx
C2RXF3SID	054C			SID<10:3>							SID<2:0>			EXIDE	—	EID<17:16>		xxxxx
C2RXF3EID	054E			EID<15:8>									EID<7:0>					xxxxx
C2RXF4SID	0550			SID<10:3>							SID<2:0>			EXIDE	—	EID<17:16>		xxxxx
C2RXF4EID	0552			EID<15:8>									EID<7:0>					xxxxx
C2RXF5SID	0554			SID<10:3>							SID<2:0>			EXIDE	—	EID<17:16>		xxxxx
C2RXF5EID	0556			EID<15:8>									EID<7:0>					xxxxx
C2RXF6SID	0558			SID<10:3>							SID<2:0>			EXIDE	—	EID<17:16>		xxxxx
C2RXF6EID	055A			EID<15:8>									EID<7:0>					xxxxx
C2RXF7SID	055C			SID<10:3>							SID<2:0>			EXIDE	—	EID<17:16>		xxxxx
C2RXF7EID	055E			EID<15:8>									EID<7:0>					xxxxx
C2RXF8SID	0560			SID<10:3>							SID<2:0>			EXIDE	—	EID<17:16>		xxxxx
C2RXF8EID	0562			EID<15:8>									EID<7:0>					xxxxx
C2RXF9SID	0564			SID<10:3>							SID<2:0>			EXIDE	—	EID<17:16>		xxxxx
C2RXF9EID	0566			EID<15:8>									EID<7:0>					xxxxx
C2RXF10SID	0568			SID<10:3>							SID<2:0>			EXIDE	—	EID<17:16>		xxxxx
C2RXF10EID	056A			EID<15:8>									EID<7:0>					xxxxx
C2RXF11SID	056C			SID<10:3>							SID<2:0>			EXIDE	—	EID<17:16>		xxxxx
C2RXF11EID	056E			EID<15:8>									EID<7:0>					xxxxx
C2RXF12SID	0570			SID<10:3>							SID<2:0>			EXIDE	—	EID<17:16>		xxxxx
C2RXF12EID	0572			EID<15:8>									EID<7:0>					xxxxx
C2RXF13SID	0574			SID<10:3>							SID<2:0>			EXIDE	—	EID<17:16>		xxxxx
C2RXF13EID	0576			EID<15:8>									EID<7:0>					xxxxx
C2RXF14SID	0578			SID<10:3>							SID<2:0>			EXIDE	—	EID<17:16>		xxxxx
C2RXF14EID	057A			EID<15:8>									EID<7:0>					xxxxx
C2RXF15SID	057C			SID<10:3>							SID<2:0>			EXIDE	—	EID<17:16>		xxxxx
C2RXF15EID	057E			EID<15:8>									EID<7:0>					xxxxx

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

**TABLE 3-22: PORTA REGISTER MAP(1)**

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	TRISA14	TRISA13	TRISA12	—	TRISA10	TRISA9	—	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	D6C0
PORTA	02C2	RA15	RA14	RA13	RA12	—	RA10	RA9	—	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0	xxxxx
LATA	02C4	LATA15	LATA14	LATA13	LATA12	—	LATA10	LATA9	—	LATA7	LATA6	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	xxxxx
ODCA(2)	06C0	ODCA15	ODCA14	ODCA13	ODCA12	—	—	—	—	—	—	ODCA5	ODCA4	ODCA3	ODCA2	ODCA1	ODCA0	xxxxx

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

**Note 1:** The actual set of I/O port pins varies from one device to another. Please refer to the corresponding pinout diagrams.

**TABLE 3-23: PORTB REGISTER MAP(1)**

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	xxxxx
LATB	02CA	LATB15	LATB14	LATB13	LATB12	LATB11	LATB10	LATB9	LATB8	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	xxxxx

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

**Note 1:** The actual set of I/O port pins varies from one device to another. Please refer to the corresponding pinout diagrams.

**TABLE 3-24: PORTC REGISTER MAP(1)**

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	TRISC3	TRISC2	TRISC1	—	F01E
PORTC	02CE	RC15	RC14	RC13	RC12	—	—	—	—	—	—	—	RC4	RC3	RC2	RC1	—	xxxxx
LATC	02D0	LATC15	LATC14	LATC13	LATC12	—	—	—	—	—	—	—	LATC4	LATC3	LATC2	LATC1	—	xxxxx

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

**Note 1:** The actual set of I/O port pins varies from one device to another. Please refer to the corresponding pinout diagrams.

**TABLE 3-25: PORTD REGISTER MAP(1)**

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	TRISD14	TRISD13	TRISD12	TRISD11	TRISD10	TRISD9	TRISD8	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	FFFF
PORTD	02D4	RD15	RD14	RD13	RD12	RD11	RD10	RD9	RD8	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	xxxxx
LATD	02D6	LATD15	LATD14	LATD13	LATD12	LATD11	LATD10	LATD9	LATD8	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0	xxxxx
ODCD(2)	06D2	ODCD15	ODCD14	ODCD13	ODCD12	ODCD11	ODCD10	ODCD9	ODCD8	ODCD7	ODCD6	ODCD5	ODCD4	ODCD3	ODCD2	ODCD1	ODCD0	xxxxx

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

**Note 1:** The actual set of I/O port pins varies from one device to another. Please refer to the corresponding pinout diagrams.

**TABLE 3-26: PORTE REGISTER MAP<sup>(1)</sup>**

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	—	—	—	—	—	—	—	—	TRISE7	TRISE6	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0	03FF
PORTE	02DA	—	—	—	—	—	—	—	—	RE7	RE6	RE5	RE4	RE3	RE2	RE1	RE0	xxxxx
LATE	02DC	—	—	—	—	—	—	—	—	LATE7	LATE6	LATE5	LATE4	LATE3	LATE2	LATE1	LATE0	xxxxx

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

**Note 1:** The actual set of I/O port pins varies from one device to another. Please refer to the corresponding pinout diagrams.

**TABLE 3-27: PORTF REGISTER MAP<sup>(1)</sup>**

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	TRISF12	—	—	—	TRISF8	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0	31FF
PORTF	02E0	—	—	RF13	RF12	—	—	—	RF8	RF7	RF6	RF5	RF4	RF3	RF2	RF1	RF0	xxxxx
LATF	02E2	—	—	LATF13	LATF12	—	—	—	LATF8	LATF7	LATF6	LATF5	LATF4	LATF3	LATF2	LATF1	LATF0	xxxxx
ODCF <sup>(2)</sup>	06DE	—	—	ODCF13	ODCF12	—	—	—	ODCF8	ODCF7	ODCF6	ODCF5	ODCF4	ODCF3	ODCF2	ODCF1	ODCF0	xxxxx

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

**Note 1:** The actual set of I/O port pins varies from one device to another. Please refer to the corresponding pinout diagrams.

**TABLE 3-28: PORTG REGISTER MAP<sup>(1)</sup>**

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	TRISG14	TRISG13	TRISG12	—	—	TRISG9	TRISG8	TRISG7	TRISG6	—	—	TRISG3	TRISG2	TRISG1	TRISG0	F3CF
PORTG	02E6	RG15	RG14	RG13	RG12	—	—	RG9	RG8	RG7	RG6	—	—	RG3	RG2	RG1	RG0	xxxxx
LATG	02E8	LATG15	LATG14	LATG13	LATG12	—	—	LATG9	LATG8	LATG7	LATG6	—	—	LATG3	LATG2	LATG1	LATG0	xxxxx
ODCG <sup>(2)</sup>	06E4	ODCG15	ODCG14	ODCG13	ODCG12	—	—	ODCG9	ODCG8	ODCG7	ODCG6	—	—	ODCG3	ODCG2	ODCG1	ODCG0	xxxxx

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

**Note 1:** The actual set of I/O port pins varies from one device to another. Please refer to the corresponding pinout diagrams.



**TABLE 3-29: SYSTEM CONTROL 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	—	—	—	—	—	VREGS	EXTR	SWR	SWDTEN	WDTO	SLEEP	IDLE	BOR	POR	xxxxx(1)
OSCCON	0742	—	—	COSC<2:0>	—	—	NOSC<2:0>	—	—	CLKLOCK	—	LOCK	—	CF	—	LPOSCEN	OSWEN	0300(2)
CLKDIV	0744	ROI	—	DOZE<2:0>	—	DOZEN	—	—	—	—	—	—	—	—	—	—	—	0040
PLLFBD	0746	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0030
OSCTUN	0748	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	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 3-30: 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	—	—	—	—	—	—	0000(1)
NVMKEY	0766	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000

**Legend:** x = unknown value on Reset, — = 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 3-31: 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	C2MD	C1MD	AD1MD	0000
PMD2	0772	IC8MD	IC7MD	IC6MD	IC5MD	IC4MD	IC3MD	IC2MD	IC1MD	OC8MD	OC7MD	OC6MD	OC5MD	OC4MD	OC3MD	OC2MD	OC1MD	0000
PMD3	0774	T9MD	T8MD	T7MD	T6MD	—	—	—	—	—	—	—	—	—	—	I2C2MD	AD2MD	0000

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

## 3.2.6 SOFTWARE STACK

In addition to its use as a working register, the W15 register in the PIC24H devices is also used as a software Stack Pointer. The Stack 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 3-5. 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.

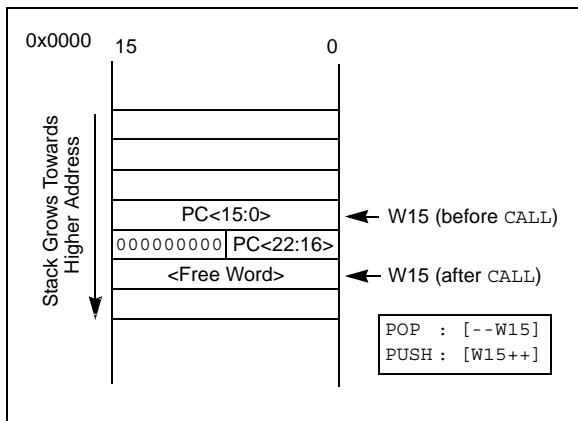
**Note:** A PC push during exception processing concatenates the SRL register to the MSB of the PC prior to the push.

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 0x2000 in RAM, initialize the SPLIM with the value 0x1FFE.

Similarly, a Stack Pointer underflow (stack error) trap is generated when the Stack Pointer address is found to be less than 0x0800. 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 3-5: CALL STACK FRAME**



## 3.3 Instruction Addressing Modes

The addressing modes in Table 3-32 form the basis of the addressing modes optimized to support the specific features of individual instructions. The addressing modes provided in the MAC class of instructions are somewhat different from those in the other instruction types.

### 3.3.1 FILE REGISTER INSTRUCTIONS

Most file register instructions use a 13-bit address field (f) to directly address data present in the first 8192 bytes of data memory (Near Data Space). Most file register instructions employ a working register, W0, which is denoted as WREG in these instructions. The destination is typically either the same file register or WREG (with the exception of the MUL instruction), which writes the result to a register or register pair. The MOV instruction allows additional flexibility and can access the entire data space.

### 3.3.2 MCU INSTRUCTIONS

The 3-operand MCU instructions are of the form:

Operand 3 = Operand 1 <function> Operand 2

where Operand 1 is always a working register (i.e., the addressing mode can only be Register Direct) which is referred to as Wb. Operand 2 can be a W register, fetched from data memory, or a 5-bit literal. The result location can be either a W register or a data memory location. The following addressing modes are supported by MCU instructions:

- Register Direct
- Register Indirect
- Register Indirect Post-Modified
- Register Indirect Pre-Modified
- 5-bit or 10-bit Literal

**Note:** Not all instructions support all the addressing modes given above. Individual instructions may support different subsets of these addressing modes.

**TABLE 3-32: FUNDAMENTAL ADDRESSING MODES SUPPORTED**

Addressing Mode	Description
File Register Direct	The address of the file register is specified explicitly.
Register Direct	The contents of a register are accessed directly.
Register Indirect	The contents of Wn forms the EA.
Register Indirect Post-Modified	The contents of Wn forms the EA. Wn is post-modified (incremented or decremented) by a constant value.
Register Indirect Pre-Modified	Wn is pre-modified (incremented or decremented) by a signed constant value to form the EA.
Register Indirect with Register Offset	The sum of Wn and Wb forms the EA.
Register Indirect with Literal Offset	The sum of Wn and a literal forms the EA.

### 3.3.3 MOVE INSTRUCTIONS

Move instructions provide a greater degree of addressing flexibility than other instructions. In addition to the Addressing modes supported by most MCU instructions, move instructions also support Register Indirect with Register Offset Addressing mode, also referred to as Register Indexed mode.

**Note:** For the MOV instructions, the Addressing mode specified in the instruction can differ for the source and destination EA. However, the 4-bit Wb (Register Offset) field is shared between both source and destination (but typically only used by one).

In summary, the following Addressing modes are supported by move instructions:

- Register Direct
- Register Indirect
- Register Indirect Post-modified
- Register Indirect Pre-modified
- Register Indirect with Register Offset (Indexed)
- Register Indirect with Literal Offset
- 8-bit Literal
- 16-bit Literal

**Note:** Not all instructions support all the Addressing modes given above. Individual instructions may support different subsets of these Addressing modes.

### 3.3.4 OTHER INSTRUCTIONS

Besides the various addressing modes outlined above, some instructions use literal constants of various sizes. For example, BRA (branch) instructions use 16-bit signed literals to specify the branch destination directly, whereas the DISI instruction uses a 14-bit unsigned literal field. In some instructions, the source of an operand or result is implied by the opcode itself. Certain operations, such as NOP, do not have any operands.

## 3.4 Interfacing Program and Data Memory Spaces

The PIC24H architecture uses a 24-bit wide program space and a 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 PIC24H 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 capability 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.

### 3.4.1 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).

# PIC24H

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.

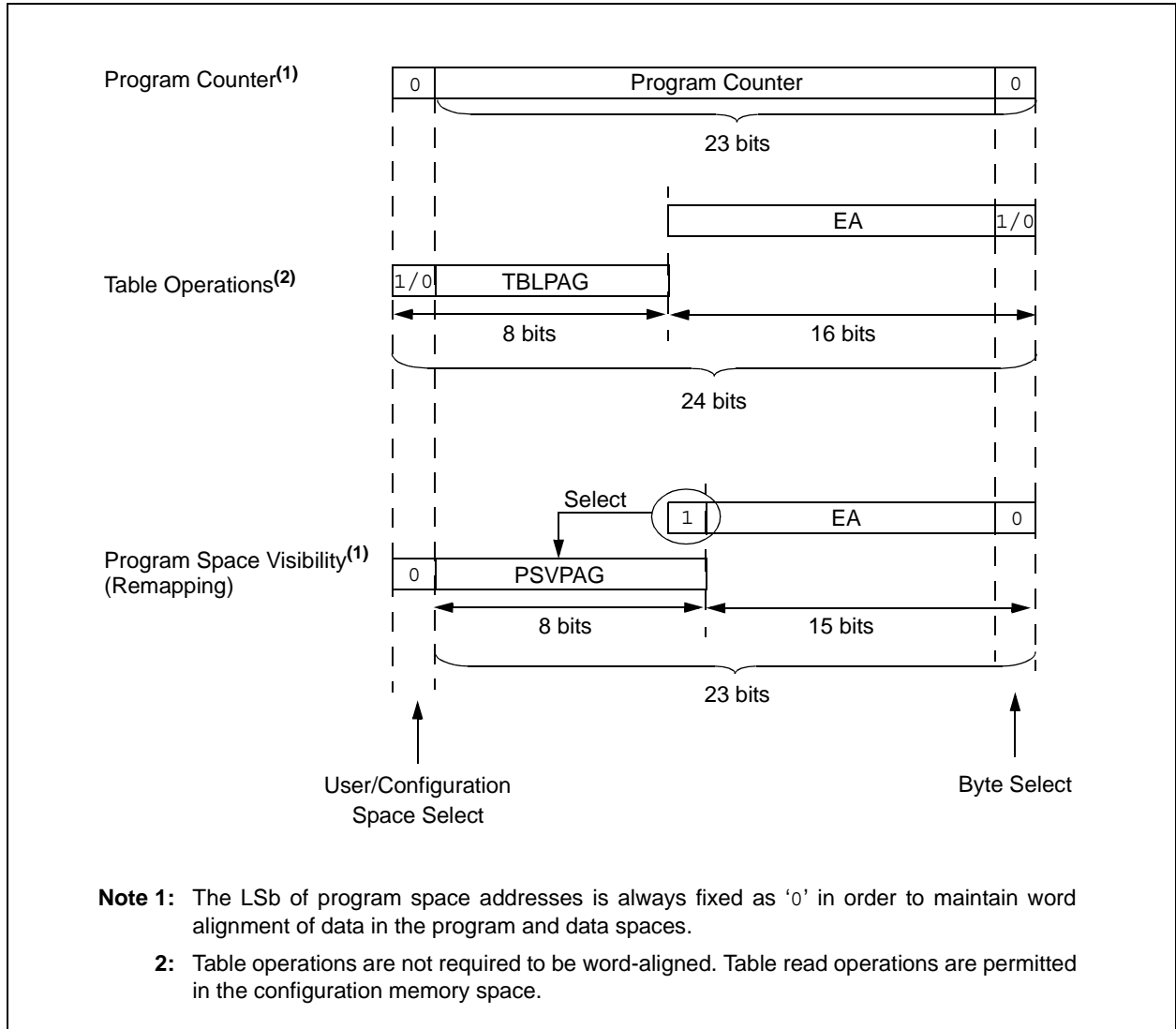
Table 3-33 and Figure 3-6 show how the program EA is created for table operations and remapping accesses from the data EA. Here, P<23:0> refers to a program space word, whereas D<15:0> refers to a data space word.

**TABLE 3-33: 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> <sup>(1)</sup>	
		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 3-6: DATA ACCESS FROM PROGRAM SPACE ADDRESS GENERATION**



# PIC24H

## 3.4.2 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'.

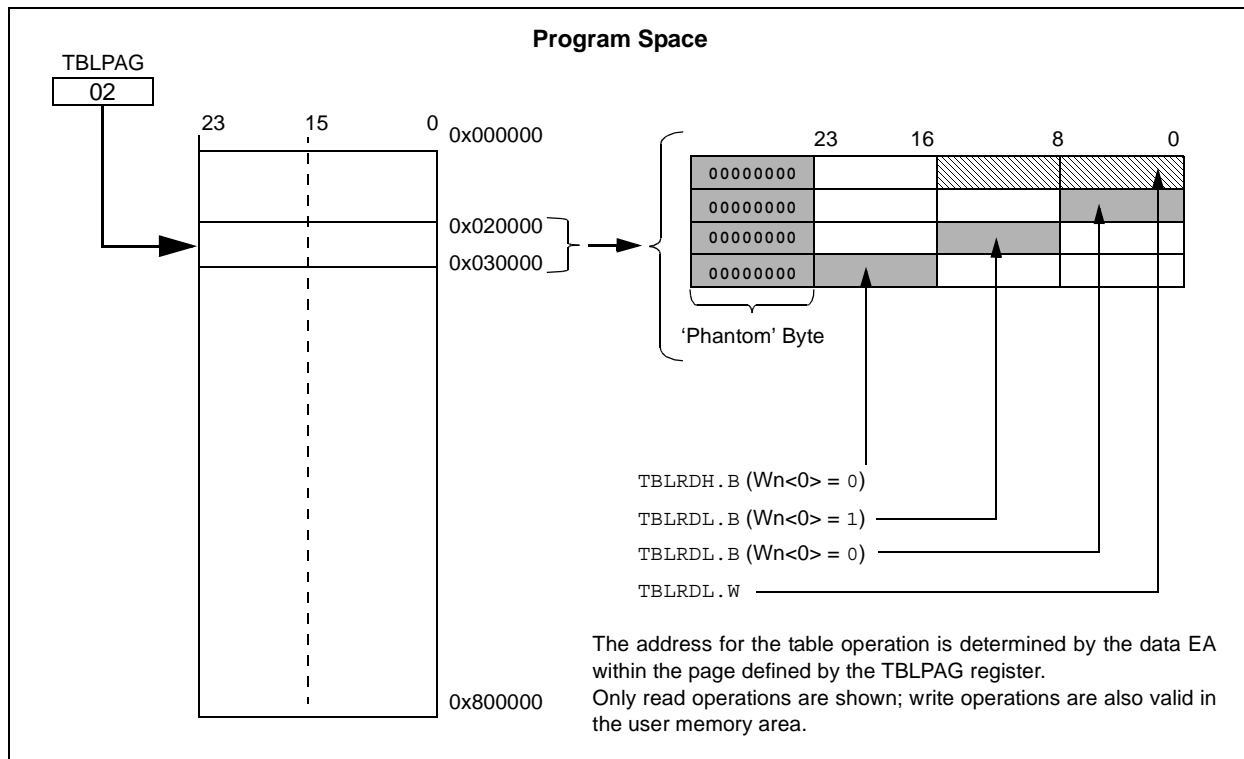
2. TBLRDH (Table Read High): In Word mode, it maps the entire upper word of a program address ( $P<23:16>$ ) to a data address. Note that  $D<15:8>$ , the 'phantom byte', will always be '0'.

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.

FIGURE 3-7: ACCESSING PROGRAM MEMORY WITH TABLE INSTRUCTIONS



### 3.4.3 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 option 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 3-8), 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 location 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.

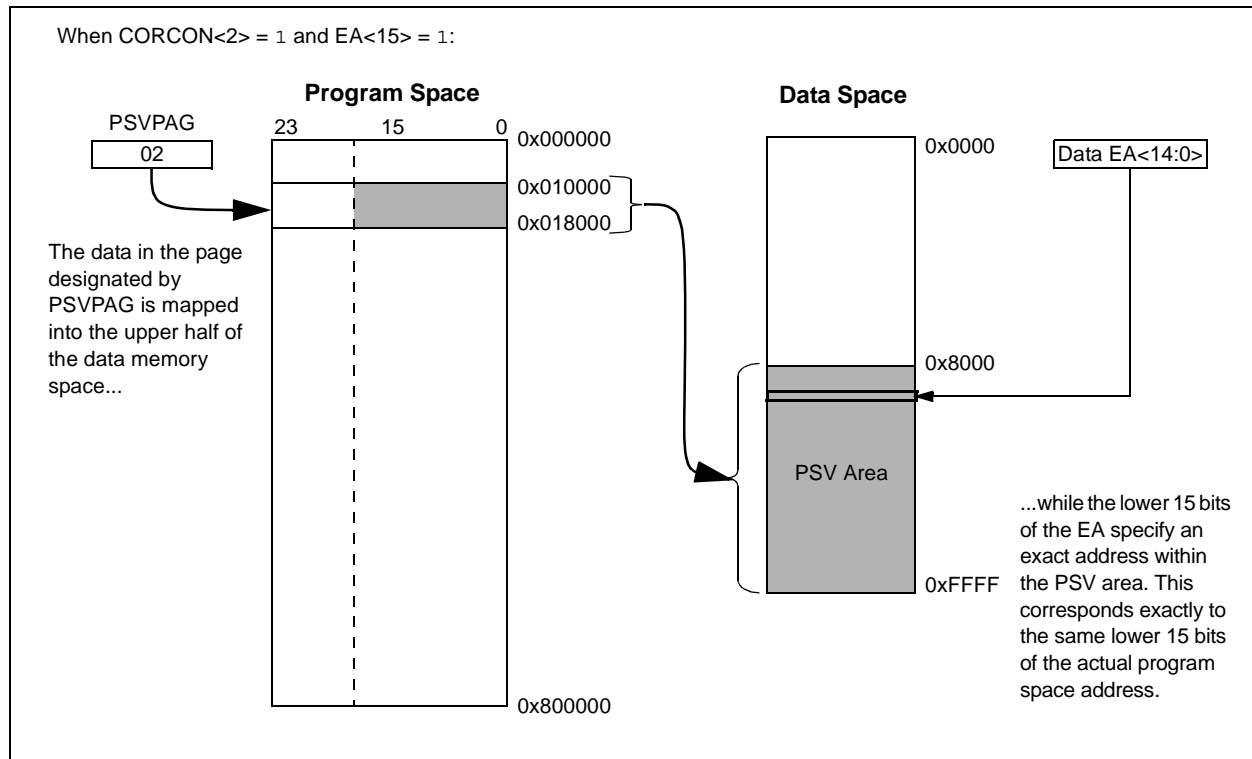
For operations that use PSV and are executed outside a REPEAT loop, the MOV and MOV.D instructions require one instruction cycle in addition to the specified execution time. All other instructions require two instruction cycles in addition to the specified execution time.

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 3-8: PROGRAM SPACE VISIBILITY OPERATION**



# PIC24H

---

NOTES:



## 4.0 FLASH PROGRAM MEMORY

**Note:** This data sheet summarizes the features of this group of PIC24H devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the “dsPIC30F Family Reference Manual” (DS70046).

The PIC24H devices contain 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™) programming capability
2. Run-Time Self-Programming (RTSP)

ICSP programming capability allows a PIC24H device to be serially programmed while in the end application circuit. This is simply done with two lines for programming clock and programming data (one of the alternate programming pin pairs: PGC1/PGD1, PGC2/PGD2 or PGC3/PGD3, and three other lines for power (VDD), ground (VSS) and Master Clear (MCLR). This allows customers to manufacture boards with unprogrammed devices and then program the digital signal controller 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 can write program memory data in blocks or ‘rows’ of 64 instructions (192 bytes) at a time, and erase program memory in blocks or ‘pages’ of 512 instructions (1536 bytes) at a time.

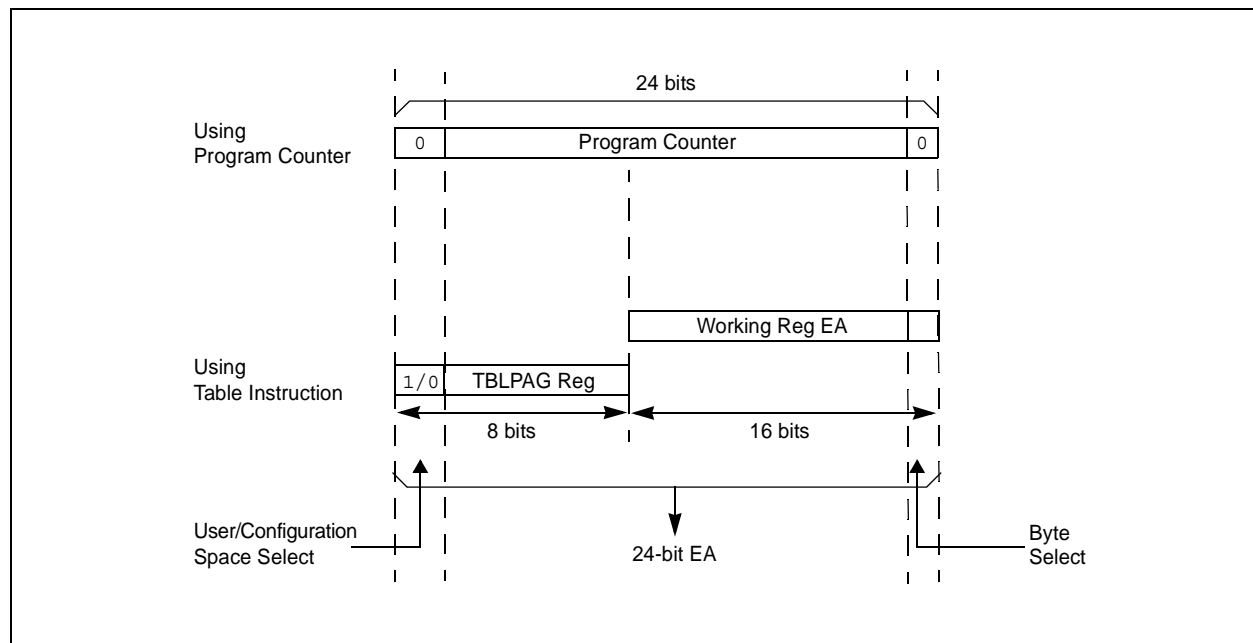
### 4.1 Table Instructions and Flash Programming

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 PIC24H Flash program memory array is organized into rows of 64 instructions or 192 bytes. RTSP allows the user to erase a page of memory, which consists of eight rows (512 instructions) at a time, and to program one row at a time. **TABLE 23-11: “DC Characteristics: Program Memory”** displays typical erase and programming times. The 8-row erase pages and single row write rows 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 instruction words loaded must always be from a group of 64 boundary.

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 (two 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: NVMCON: FLASH MEMORY CONTROL REGISTER

R/SO-0 <sup>(1)</sup>	R/W-0 <sup>(1)</sup>	R/W-0 <sup>(1)</sup>	U-0	U-0	U-0	U-0	U-0
WR	WREN	WRERR	—	—	—	—	—
bit 15							bit 8

U-0	R/W-0 <sup>(1)</sup>	U-0	U-0	R/W-0 <sup>(1)</sup>	R/W-0 <sup>(1)</sup>	R/W-0 <sup>(1)</sup>	R/W-0 <sup>(1)</sup>
—	ERASE	—	—	NVMOP<3:0> <sup>(2)</sup>			
bit 7							bit 0

<b>Legend:</b>	SO = Satiabie only bit
R = Readable bit	W = Writable bit
-n = Value at POR	'1' = Bit is set
	U = Unimplemented bit, read as '0'
	'0' = Bit is cleared
	x = Bit is unknown

- 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 NVMOP<3:0> on the next WR command  
 0 = Perform the program operation specified by NVMOP<3:0> on the next WR command
- bit 5-4      **Unimplemented:** Read as '0'
- bit 3-0      **NVMOP<3:0>:** NVM Operation Select bits<sup>(2)</sup>  
 1111 = Memory bulk erase operation (ERASE = 1) or no operation (ERASE = 0)  
 0011 = Memory word program operation (ERASE = 0) or no operation (ERASE = 1)  
 0010 = Memory page erase operation (ERASE = 1) or no operation (ERASE = 0)  
 0001 = Memory row program operation (ERASE = 0) or no operation (ERASE = 1)  
 0000 = Program or erase a single Configuration register byte

**Note 1:** These bits can only be reset on POR.

**2:** All other combinations of NVMOP<3:0> are unimplemented.

# PIC24H

## 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 page that contains 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 page (see Example 4-1):
  - a) Set the NVMOP bits (NVMCON<3:0>) to '0010' to configure for block erase. Set the ERASE (NVMCON<6>) and WREN (NVMCON<14>) bits.
  - b) Write the starting address of the page to be erased into the TBLPAG and W registers.
  - c) Perform a dummy table write operation (TBLWTL) to any address within the page that needs to be erased.
  - d) Write 0x55 to NVMKEY.
  - e) Write 0xAA to NVMKEY.
  - f) Set the WR bit (NVMCON<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 0x55 to NVMKEY.
  - c) Write 0xAA 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 PAGE

```
; 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
NOP                               ; Insert two NOPs after the erase
                                           ; command is asserted
```

**Note:** A program memory page erase operation is set up by performing a dummy table write (TBLWTL) operation to any address within the page. This methodology is different from the page erase operation on dsPIC30F devices in which the erase page was selected using a dedicated pair of registers (NVMADRU and NVMADR).

## 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

```

# PIC24H

---

NOTES:

## 5.0 RESETS

**Note:** This data sheet summarizes the features of this group of PIC24H devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the “*dsPIC30F Family Reference Manual*” (DS70046).

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

- POR: Power-on Reset
- BOR: Brown-out Reset
- $\overline{\text{MCLR}}$ : Master Clear Pin Reset
- SWR: RESET Instruction
- WDT: Watchdog Timer Reset
- TRAPR: Trap Conflict Reset
- IOPUWR: Illegal Opcode and 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  $\overline{\text{SYSRST}}$  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.

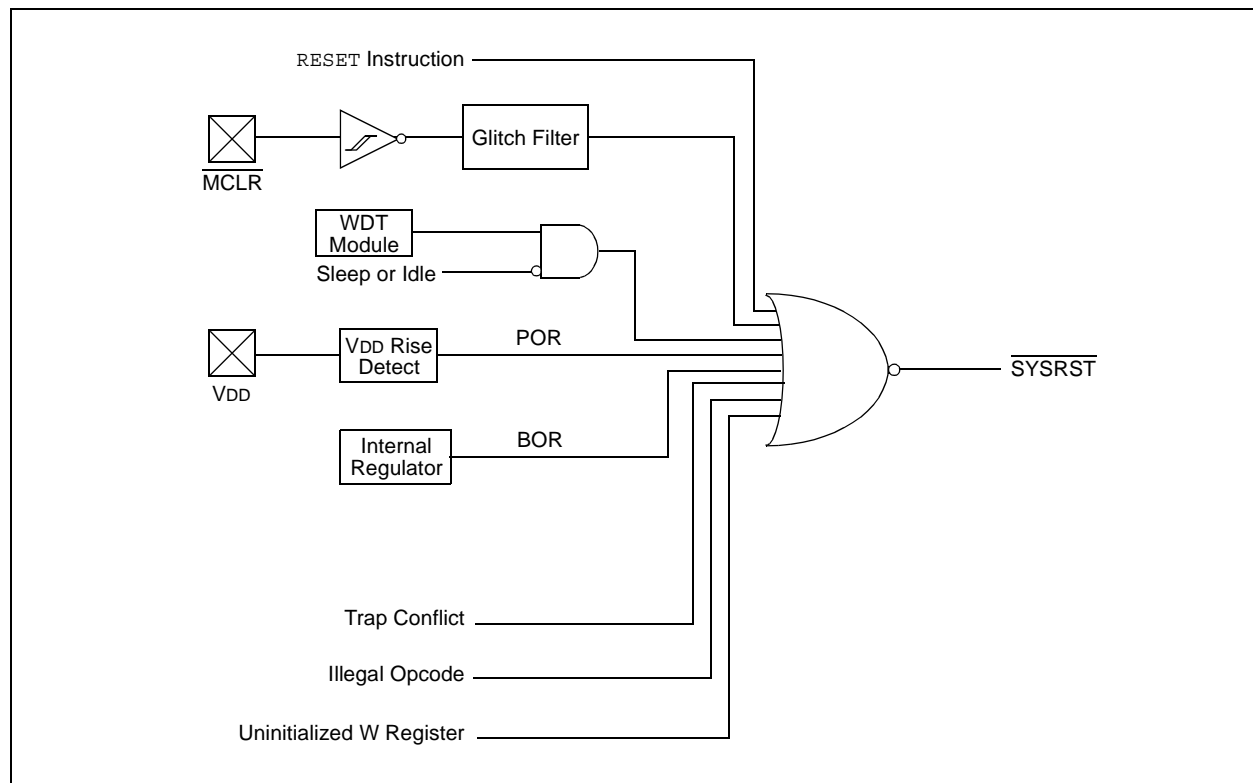
**Note:** Refer to the specific peripheral or CPU section of this manual for register Reset states.

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 POR bit (RCON<0>), that are set. The user can 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 does 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**



# PIC24H

## REGISTER 5-1: RCON: RESET CONTROL REGISTER<sup>(1)</sup>

R/W-0	R/W-0	U-0	U-0	U-0	U-0	U-0	R/W-0
TRAPR	IOPUWR	—	—	—	—	—	VREGS
bit 15							bit 8

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 <sup>(2)</sup>	WDTO	SLEEP	IDLE	BOR	POR
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 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 13-9    **Unimplemented:** Read as '0'
- bit 8        **VREGS:** Voltage Regulator Standby During Sleep bit  
 1 = Voltage regulator goes into Standby mode during Sleep  
 0 = Voltage regulator is active 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<sup>(2)</sup>  
 1 = WDT is enabled  
 0 = WDT is disabled
- 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-up 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  
 0 = A Brown-out Reset has not occurred

**Note 1:** 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.

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



## REGISTER 5-1: RCON: RESET CONTROL REGISTER<sup>(1)</sup>

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

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

# PIC24H

**TABLE 5-1: RESET FLAG BIT OPERATION**

Flag Bit	Setting Event	Clearing Event
TRAPR (RCON<15>)	Trap conflict event	POR
IOPUWR (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	PWRSV instruction, POR
SLEEP (RCON<3>)	PWRSV #SLEEP instruction	POR
IDLE (RCON<2>)	PWRSV #IDLE instruction	POR
BOR (RCON<1>)	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 **Section 8.0 “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 (FNOSC<2:0>)
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. The system Reset signal is released after the POR and PWRT delay times expire.

The time at which the device actually begins to execute code also depends 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 reset delay times.

The FSCM delay determines the time at which the FSCM begins to monitor the system clock source after the reset 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, 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
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  $\mu$ s nominal).

**2:** TSTARTUP = Conditional POR delay of 20  $\mu$ s nominal (if on-chip regulator is enabled) or 64 ms nominal Power-up Timer delay (if regulator is disabled). TSTARTUP is also applied to all returns from powered-down states, including waking from Sleep mode, only if the regulator is enabled.

**3:** TRST = Internal state Reset time (20  $\mu$ s 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  $\mu$ s nominal).

**6:** TFSCM = Fail-Safe Clock Monitor delay (100  $\mu$ s 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) have a relatively long start-up time. Therefore, one or more of the following conditions is possible after the Reset signal 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 begins to monitor the system clock source when the Reset signal is released. If a valid clock source is not available at this time, the device automatically switches 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, is automatically inserted after the POR and PWRT delay times. The FSCM does not begin to monitor the system clock source until this delay expires. The FSCM delay time is nominally 100  $\mu$ s and provides additional time for the oscillator and/or PLL to stabilize. In most cases, the FSCM delay prevents 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 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 two registers. The Reset value for the Reset Control register, RCON, depends on the type of device Reset. The Reset value for the Oscillator Control register, OSCCON, depends on the type of Reset and the programmed values of the oscillator Configuration bits in the FOSC Configuration register.

# PIC24H

---

NOTES:

## 6.0 INTERRUPT CONTROLLER

**Note:** This data sheet summarizes the features of this group of PIC24H devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the “*dsPIC30F Family Reference Manual*” (DS70046).

The PIC24H interrupt controller reduces the numerous peripheral interrupt request signals to a single interrupt request signal to the PIC24H 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 nonmaskable 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 priority 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.

PIC24H devices implement up to 61 unique interrupts and 5 nonmaskable traps. These are summarized in Table 6-1 and Table 6-2.

### 6.1.1 ALTERNATE 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 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 debugging by providing a means to switch between an application and a support environment without requiring the interrupt vectors to be reprogrammed. This feature also enables switching between applications for evaluation of different software algorithms at run time. If the AIVT is not needed, the AIVT should be programmed with the same addresses used in the IVT.

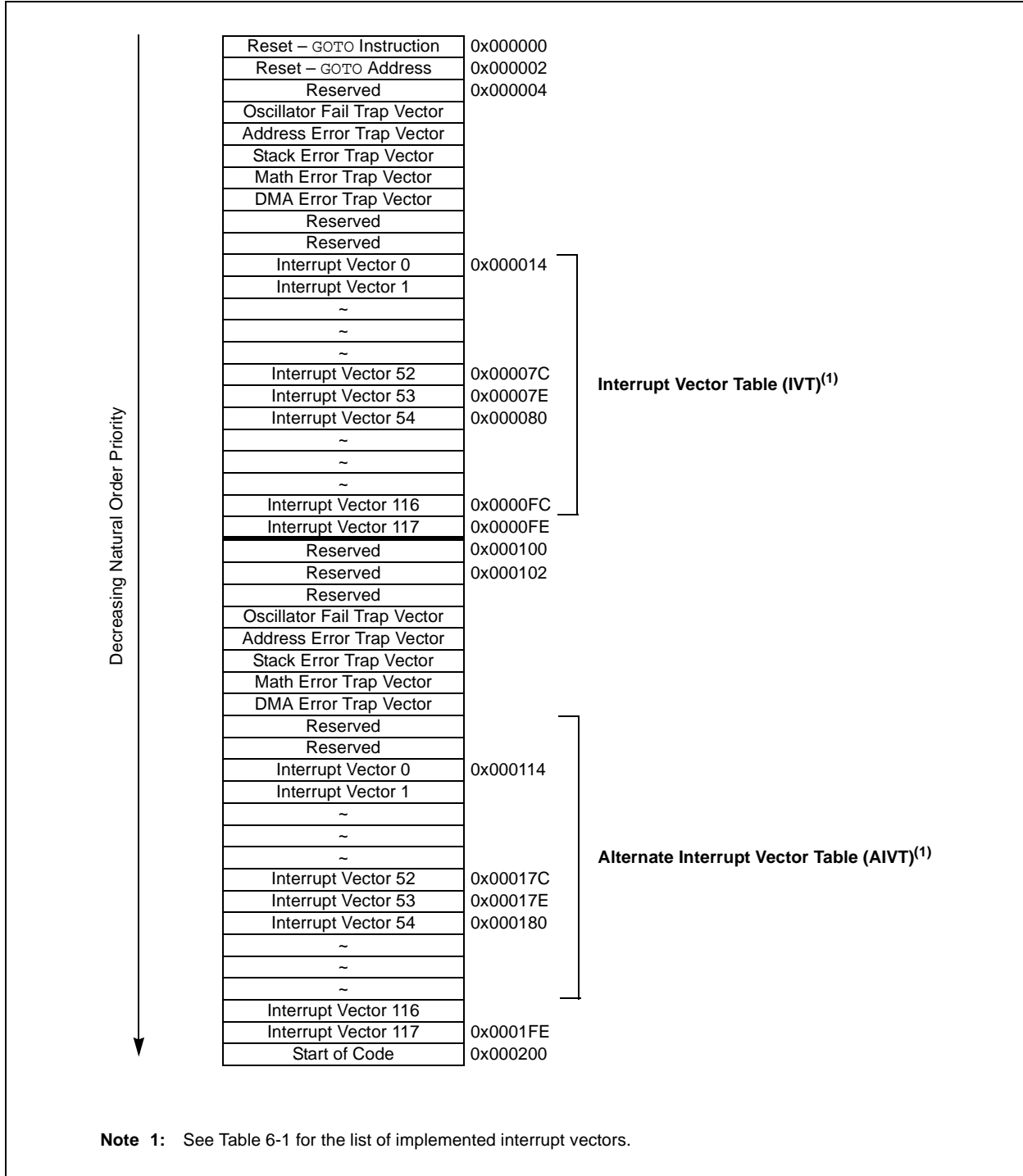
### 6.2 Reset Sequence

A device Reset is not a true exception because the interrupt controller is not involved in the Reset process. The PIC24H device clears its registers in response to a Reset which forces the PC to zero. The digital signal controller then begins program execution at location 0x000000. The user programs a GOTO instruction at the Reset address which redirects program execution to the appropriate start-up routine.

**Note:** Any unimplemented or unused vector locations in the IVT and AIVT should be programmed with the address of a default interrupt handler routine that contains a RESET instruction.

# PIC24H

**FIGURE 6-1: PIC24H INTERRUPT VECTOR TABLE**



**TABLE 6-1: INTERRUPT VECTORS**

Vector Number	Interrupt Request (IRQ) Number	IVT Address	AIVT Address	Interrupt Source
8	0	0x000014	0x000114	INT0 – External Interrupt 0
9	1	0x000016	0x000116	IC1 – Input Compare 1
10	2	0x000018	0x000118	OC1 – Output Compare 1
11	3	0x00001A	0x00011A	T1 – Timer1
12	4	0x00001C	0x00011C	DMA0 – DMA Channel 0
13	5	0x00001E	0x00011E	IC2 – Input Capture 2
14	6	0x000020	0x000120	OC2 – Output Compare 2
15	7	0x000022	0x000122	T2 – Timer2
16	8	0x000024	0x000124	T3 – Timer3
17	9	0x000026	0x000126	SPI1E – SPI1 Error
18	10	0x000028	0x000128	SPI1 – SPI1 Transfer Done
19	11	0x00002A	0x00012A	U1RX – UART1 Receiver
20	12	0x00002C	0x00012C	U1TX – UART1 Transmitter
21	13	0x00002E	0x00012E	ADC1 – A/D Converter 1
22	14	0x000030	0x000130	DMA1 – DMA Channel 1
23	15	0x000032	0x000132	Reserved
24	16	0x000034	0x000134	SI2C1 – I2C1 Slave Events
25	17	0x000036	0x000136	MI2C1 – I2C1 Master Events
26	18	0x000038	0x000138	Reserved
27	19	0x00003A	0x00013A	CN - Change Notification Interrupt
28	20	0x00003C	0x00013C	INT1 – External Interrupt 1
29	21	0x00003E	0x00013E	ADC2 – A/D Converter 2
30	22	0x000040	0x000140	IC7 – Input Capture 7
31	23	0x000042	0x000142	IC8 – Input Capture 8
32	24	0x000044	0x000144	DMA2 – DMA Channel 2
33	25	0x000046	0x000146	OC3 – Output Compare 3
34	26	0x000048	0x000148	OC4 – Output Compare 4
35	27	0x00004A	0x00014A	T4 – Timer4
36	28	0x00004C	0x00014C	T5 – Timer5
37	29	0x00004E	0x00014E	INT2 – External Interrupt 2
38	30	0x000050	0x000150	U2RX – UART2 Receiver
39	31	0x000052	0x000152	U2TX – UART2 Transmitter
40	32	0x000054	0x000154	SPI2E – SPI2 Error
41	33	0x000056	0x000156	SPI1 – SPI1 Transfer Done
42	34	0x000058	0x000158	C1RX – ECAN1 Receive Data Ready
43	35	0x00005A	0x00015A	C1 – ECAN1 Event
44	36	0x00005C	0x00015C	DMA3 – DMA Channel 3
45	37	0x00005E	0x00015E	IC3 – Input Capture 3
46	38	0x000060	0x000160	IC4 – Input Capture 4
47	39	0x000062	0x000162	IC5 – Input Capture 5
48	40	0x000064	0x000164	IC6 – Input Capture 6
49	41	0x000066	0x000166	OC5 – Output Compare 5
50	42	0x000068	0x000168	OC6 – Output Compare 6
51	43	0x00006A	0x00016A	OC7 – Output Compare 7
52	44	0x00006C	0x00016C	OC8 – Output Compare 8
53	45	0x00006E	0x00016E	Reserved

# PIC24H

**TABLE 6-1: INTERRUPT VECTORS (CONTINUED)**

Vector Number	Interrupt Request (IRQ) Number	IVT Address	AIVT Address	Interrupt Source
54	46	0x000070	0x000170	DMA4 – DMA Channel 4
55	47	0x000072	0x000172	T6 – Timer6
56	48	0x000074	0x000174	T7 – Timer7
57	49	0x000076	0x000176	SI2C2 – I2C2 Slave Events
58	50	0x000078	0x000178	MI2C2 – I2C2 Master Events
59	51	0x00007A	0x00017A	T8 – Timer8
60	52	0x00007C	0x00017C	T9 – Timer9
61	53	0x00007E	0x00017E	INT3 – External Interrupt 3
62	54	0x000080	0x000180	INT4 – External Interrupt 4
63	55	0x000082	0x000182	C2RX – ECAN2 Receive Data Ready
64	56	0x000084	0x000184	C2 – ECAN2 Event
65-68	57-60	0x000086- 0x00008C	0x000186- 0x00018C	Reserved
69	61	0x00008E	0x00018E	DMA5 – DMA Channel 5
70-72	62-64	0x000090- 0x000094	0x000190- 0x000194	Reserved
73	65	0x000096	0x000196	U1E – UART1 Error
74	66	0x000098	0x000198	U2E – UART2 Error
75	67	0x00009A	0x00019A	Reserved
76	68	0x00009C	0x00019C	DMA6 – DMA Channel 6
77	69	0x00009E	0x00019E	DMA7 – DMA Channel 7
78	70	0x0000A0	0x0001A0	C1TX – ECAN1 Transmit Data Request
79	71	0x0000A2	0x0001A2	C2TX – ECAN2 Transmit Data Request
80-125	72-117	0x0000A4- 0x0000FE	0x0001A4- 0x0001FE	Reserved

**TABLE 6-2: TRAP VECTORS**

Vector Number	IVT Address	AIVT Address	Trap Source
0	0x000004	0x000084	Reserved
1	0x000006	0x000086	Oscillator Failure
2	0x000008	0x000088	Address Error
3	0x00000A	0x00008A	Stack Error
4	0x00000C	0x00008C	Math Error
5	0x00000E	0x00008E	DMA Error Trap
6	0x000010	0x000090	Reserved
7	0x000012	0x000092	Reserved



## 6.3 Interrupt Control and Status Registers

PIC24H devices implement a total of 30 registers for the interrupt controller:

- INTCON1
- INTCON2
- IFS0 through IFS4
- IEC0 through IEC4
- IPC0 through IPC17

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 6-1. For example, the INTO (External Interrupt 0) is shown as having vector number 8 and a natural order priority of 0. Thus, the INTOIF bit is found in IFS0<0>, the INTOIE bit in IEC0<0>, and the INTOIP 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 IPL<2:0> bits (SR<7:5>). These bits indicate the current CPU interrupt priority level. The user can change the current CPU priority level by writing to the IPL bits.

The CORCON register contains the IPL3 bit which, together with IPL<2:0>, 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, SR: CPU STATUS Register<sup>(1)</sup>** through **Register 6-32, IPC17: Interrupt Priority Control Register 17**, in the following pages.

# PIC24H

## REGISTER 6-1: SR: CPU STATUS REGISTER<sup>(1)</sup>

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
—	—	—	—	—	—	—	DC
bit 15							bit 8
R/W-0 <sup>(3)</sup>	R/W-0 <sup>(3)</sup>	R/W-0 <sup>(3)</sup>	R-0	R/W-0	R/W-0	R/W-0	R/W-0
IPL2 <sup>(2)</sup>	IPL1 <sup>(2)</sup>	IPL0 <sup>(2)</sup>	RA	N	OV	Z	C
bit 7							bit 0

### Legend:

C = Clear only bit	R = Readable bit	U = Unimplemented bit, read as '0'
S = Set only bit	W = Writable bit	-n = Value at POR
'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-5 **IPL<2:0>**: CPU Interrupt Priority Level Status bits<sup>(2)</sup>

- 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:** For complete register details, see **Register 2-1, SR: CPU STATUS Register**.

**2:** The IPL<2:0> bits are concatenated with the IPL<3> bit (CORCON<3>) to form the CPU Interrupt Priority Level. The value in parentheses indicates the IPL if IPL<3> = 1. User interrupts are disabled when IPL<3> = 1.

**3:** The IPL<2:0> Status bits are read-only when NSTDIS (INTCON1<15>) = 1.

## REGISTER 6-2: CORCON: CORE CONTROL REGISTER<sup>(1)</sup>

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8
U-0	U-0	U-0	U-0	R/C-0	R/W-0	U-0	U-0
—	—	—	—	IPL3 <sup>(2)</sup>	PSV	—	—
bit 7							bit 0

### Legend:

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

bit 3 **IPL3**: CPU Interrupt Priority Level Status bit 3<sup>(2)</sup>

- 1 = CPU interrupt priority level is greater than 7
- 0 = CPU interrupt priority level is 7 or less

**Note 1:** For complete register details, see **Register 2-2, CORCON: CORE Control Register**.

**2:** The IPL3 bit is concatenated with the IPL<2:0> bits (SR<7:5>) to form the CPU Interrupt Priority Level.

## REGISTER 6-3: INTCON1: INTERRUPT CONTROL REGISTER 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
NSTDIS	OVAERR	OVBERR	COVAERR	COVBERR	OVATE	OVBTE	COVTE
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0
SFTACERR	DIV0ERR	DMACERR	MATHERR	ADDRERR	STKERR	OSCFAIL	—
bit 7							bit 0

### Legend:

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

- bit 15      **NSTDIS:** Interrupt Nesting Disable bit  
1 = Interrupt nesting is disabled  
0 = Interrupt nesting is enabled
- bit 14      **OVAERR:** Accumulator A Overflow Trap Flag bit  
1 = Trap was caused by overflow of Accumulator A  
0 = Trap was not caused by overflow of Accumulator A
- bit 13      **OVBERR:** Accumulator B Overflow Trap Flag bit  
1 = Trap was caused by overflow of Accumulator B  
0 = Trap was not caused by overflow of Accumulator B
- bit 12      **COVAERR:** Accumulator A Catastrophic Overflow Trap Enable bit  
1 = Trap was caused by catastrophic overflow of Accumulator A  
0 = Trap was not caused by catastrophic overflow of Accumulator A
- bit 11      **COVBERR:** Accumulator B Catastrophic Overflow Trap Enable bit  
1 = Trap was caused by catastrophic overflow of Accumulator B  
0 = Trap was not caused by catastrophic overflow of Accumulator B
- bit 10      **OVATE:** Accumulator A Overflow Trap Enable bit  
1 = Trap overflow of Accumulator A  
0 = Trap disabled
- bit 9        **OVBTE:** Accumulator B Overflow Trap Enable bit  
1 = Trap overflow of Accumulator B  
0 = Trap disabled
- bit 8        **COVTE:** Catastrophic Overflow Trap Enable bit  
1 = Trap on catastrophic overflow of Accumulator A or B enabled  
0 = Trap disabled
- bit 7        **SFTACERR:** Shift Accumulator Error Status bit  
1 = Math error trap was caused by an invalid accumulator shift  
0 = Math error trap was not caused by an invalid accumulator shift
- bit 6        **DIV0ERR:** Arithmetic Error Status bit  
1 = Math error trap was caused by a divide by zero  
0 = Math error trap was not caused by a divide by zero
- bit 5        **DMACERR:** DMA Controller Error Status bit  
1 = DMA controller error trap has occurred  
0 = DMA controller error trap has not occurred
- bit 4        **MATHERR:** Arithmetic Error Status bit  
1 = Math error trap has occurred  
0 = Math error trap has not occurred

# PIC24H

---

## REGISTER 6-3: INTCON1: INTERRUPT CONTROL REGISTER 1 (CONTINUED)

- 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'

## REGISTER 6-4: INTCON2: INTERRUPT CONTROL REGISTER 2

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

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

### Legend:

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

- bit 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 instruction 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

# PIC24H

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

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	DMA1IF	AD1IF	U1TXIF	U1RXIF	SPI1IF	SPI1EIF	T3IF
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
T2IF	OC2IF	IC2IF	DMA01IF	T1IF	OC1IF	IC1IF	INT0IF
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 15      **Unimplemented:** Read as '0'
- bit 14      **DMA1IF:** DMA Channel 1 Data Transfer Complete Interrupt Flag Status bit  
             1 = Interrupt request has occurred  
             0 = Interrupt request has not occurred
- bit 13      **AD1IF:** ADC1 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        **SPI1EIF:** 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        **DMA0IF:** DMA Channel 0 Data Transfer Complete Interrupt Flag Status bit  
             1 = Interrupt request has occurred  
             0 = Interrupt request has not occurred

## REGISTER 6-5: IFS0: INTERRUPT FLAG STATUS REGISTER 0 (CONTINUED)

- 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

# PIC24H

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

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
U2TXIF	U2RXIF	INT2IF	T5IF	T4IF	OC4IF	OC3IF	DMA21IF
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
IC8IF	IC7IF	AD2IF	INT1IF	CNIF	—	MI2C1IF	SI2C1IF
bit 7							bit 0

### Legend:

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

- bit 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             **DMA2IF:** DMA Channel 2 Data Transfer Complete Interrupt Flag Status bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 7             **IC8IF:** Input Capture Channel 8 Interrupt Flag Status bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 6             **IC7IF:** Input Capture Channel 7 Interrupt Flag Status bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 5             **AD2IF:** ADC2 Conversion Complete Interrupt Flag Status bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 4             **INT1IF:** External Interrupt 1 Flag Status bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred



## REGISTER 6-6: IFS1: INTERRUPT FLAG STATUS REGISTER 1 (CONTINUED)

- bit 3        **CNIF**: Input Change Notification Interrupt Flag Status bit  
             1 = Interrupt request has occurred  
             0 = Interrupt request has not occurred
- bit 2        **Unimplemented**: Read as '0'
- bit 1        **MI2C1IF**: I2C1 Master Events Interrupt Flag Status bit  
             1 = Interrupt request has occurred  
             0 = Interrupt request has not occurred
- bit 0        **SI2C1IF**: I2C1 Slave Events Interrupt Flag Status bit  
             1 = Interrupt request has occurred  
             0 = Interrupt request has not occurred

# PIC24H

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

R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
T6IF	DMA4IF	—	OC8IF	OC7IF	OC6IF	OC5IF	IC6IF
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
IC5IF	IC4IF	IC3IF	DMA3IF	C1IF	C1RXIF	SPI2IF	SPI2EIF
bit 7							bit 0

### Legend:

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

- bit 15      **T6IF:** Timer6 Interrupt Flag Status bit  
1 = Interrupt request has occurred  
0 = Interrupt request has not occurred
- bit 14      **DMA4IF:** DMA Channel 4 Data Transfer Complete Interrupt Flag Status bit  
1 = Interrupt request has occurred  
0 = Interrupt request has not occurred
- bit 13      **Unimplemented:** Read as '0'
- bit 12      **OC8IF:** Output Compare Channel 8 Interrupt Flag Status bit  
1 = Interrupt request has occurred  
0 = Interrupt request has not occurred
- bit 11      **OC7IF:** Output Compare Channel 7 Interrupt Flag Status bit  
1 = Interrupt request has occurred  
0 = Interrupt request has not occurred
- bit 10      **OC6IF:** Output Compare Channel 6 Interrupt Flag Status bit  
1 = Interrupt request has occurred  
0 = Interrupt request has not occurred
- bit 9        **OC5IF:** Output Compare Channel 5 Interrupt Flag Status bit  
1 = Interrupt request has occurred  
0 = Interrupt request has not occurred
- bit 8        **IC6IF:** Input Capture Channel 6 Interrupt Flag Status bit  
1 = Interrupt request has occurred  
0 = Interrupt request has not occurred
- 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        **DMA3IF:** DMA Channel 3 Data Transfer Complete Interrupt Flag Status bit  
1 = Interrupt request has occurred  
0 = Interrupt request has not occurred

## REGISTER 6-7: IFS2: INTERRUPT FLAG STATUS REGISTER 2 (CONTINUED)

- bit 3      **C1IF:** ECAN1 Event Interrupt Flag Status bit  
            1 = Interrupt request has occurred  
            0 = Interrupt request has not occurred
- bit 2      **C1RXIF:** ECAN1 Receive Data Ready Interrupt Flag Status bit  
            1 = Interrupt request has occurred  
            0 = Interrupt request has not occurred
- bit 1      **SPI2IF:** SPI2 Event Interrupt Flag Status bit  
            1 = Interrupt request has occurred  
            0 = Interrupt request has not occurred
- bit 0      **SPI2EIF:** SPI2 Error Interrupt Flag Status bit  
            1 = Interrupt request has occurred  
            0 = Interrupt request has not occurred

# PIC24H

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

U-0	U-0	R/W-0	U-0	U-0	U-0	U-0	R/W-0
—	—	DMA5IF	—	—	—	—	C2IF
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
C2RXIF	INT4IF	INT3IF	T9IF	T8IF	MI2C2IF	SI2C2IF	T7IF
bit 7							bit 0

### Legend:

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

- bit 15-14     **Unimplemented:** Read as '0'
- bit 13        **DMA5IF:** DMA Channel 5 Data Transfer Complete Interrupt Flag Status bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 12-9     **Unimplemented:** Read as '0'
- bit 8         **C2IF:** ECAN2 Event Interrupt Flag Status bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 7         **C2RXIF:** ECAN2 Receive Data Ready Interrupt Flag Status bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- 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         **T9IF:** Timer9 Interrupt Flag Status bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 3         **T8IF:** Timer8 Interrupt Flag Status bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 2         **MI2C2IF:** I2C2 Master Events Interrupt Flag Status bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 1         **SI2C2IF:** I2C2 Slave Events Interrupt Flag Status bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 0         **T7IF:** Timer7 Interrupt Flag Status bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred

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

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15						bit 8	

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	U-0
C2TXIF	C1TXIF	DMA7IF	DMA6IF	—	U2EIF	U1EIF	—
bit 7						bit 0	

### Legend:

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

- bit 15-8        **Unimplemented:** Read as '0'
- bit 7            **C2TXIF:** ECAN2 Transmit Data Request Interrupt Flag Status bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 6            **C1TXIF:** ECAN1 Transmit Data Request Interrupt Flag Status bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 5            **DMA7IF:** DMA Channel 7 Data Transfer Complete Interrupt Flag Status bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 4            **DMA6IF:** DMA Channel 6 Data Transfer Complete Interrupt Flag Status bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 3            **Unimplemented:** Read as '0'
- bit 2            **U2EIF:** UART2 Error Interrupt Flag Status bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 1            **U1EIF:** UART1 Error Interrupt Flag Status bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 0            **Unimplemented:** Read as '0'

# PIC24H

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

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	DMA1IE	AD1IE	U1TXIE	U1RXIE	SPI1IE	SPI1EIE	T3IE
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
T2IE	OC2IE	IC2IE	DMA0IE	T1IE	OC1IE	IC1IE	INT0IE
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 15      **Unimplemented:** Read as '0'
- bit 14      **DMA1IE:** DMA Channel 1 Data Transfer Complete Interrupt Enable bit  
             1 = Interrupt request enabled  
             0 = Interrupt request not enabled
- bit 13      **AD1IE:** ADC1 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 Event Interrupt Enable bit  
             1 = Interrupt request enabled  
             0 = Interrupt request not enabled
- bit 9        **SPI1EIE:** SPI1 Error 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        **DMA0IE:** DMA Channel 0 Data Transfer Complete Interrupt Enable bit  
             1 = Interrupt request enabled  
             0 = Interrupt request not enabled
- bit 3        **T1IE:** Timer1 Interrupt Enable bit  
             1 = Interrupt request enabled  
             0 = Interrupt request not enabled

## REGISTER 6-10: IEC0: INTERRUPT ENABLE CONTROL REGISTER 0 (CONTINUED)

- 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

# PIC24H

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

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
U2TXIE	U2RXIE	INT2IE	T5IE	T4IE	OC4IE	OC3IE	DMA2IE
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
IC8IE	IC7IE	AD2IE	INT1IE	CNIE	—	MI2C1IE	SI2C1IE
bit 7						bit 0	

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 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        **DMA2IE:** DMA Channel 2 Data Transfer Complete Interrupt Enable bit  
1 = Interrupt request has occurred  
0 = Interrupt request has not occurred
- bit 7        **IC8IE:** Input Capture Channel 8 Interrupt Enable bit  
1 = Interrupt request has occurred  
0 = Interrupt request has not occurred
- bit 6        **IC7IE:** Input Capture Channel 7 Interrupt Enable bit  
1 = Interrupt request has occurred  
0 = Interrupt request has not occurred
- bit 5        **AD2IE:** ADC2 Conversion Complete Interrupt Enable bit  
1 = Interrupt request has occurred  
0 = Interrupt request has not occurred
- bit 4        **INT1IE:** External Interrupt 1 Enable bit  
1 = Interrupt request enabled  
0 = Interrupt request not enabled



## REGISTER 6-11: IEC1: INTERRUPT ENABLE CONTROL REGISTER 1 (CONTINUED)

- bit 3        **CNIE:** Input Change Notification Interrupt Enable bit  
             1 = Interrupt request enabled  
             0 = Interrupt request not enabled
- bit 2        **Unimplemented:** Read as '0'
- bit 1        **MI2C1IE:** I2C1 Master Events Interrupt Enable bit  
             1 = Interrupt request enabled  
             0 = Interrupt request not enabled
- bit 0        **SI2C1IE:** I2C1 Slave Events Interrupt Enable bit  
             1 = Interrupt request enabled  
             0 = Interrupt request not enabled

# PIC24H

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

R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
T6IE	DMA4IE	—	OC8IE	OC7IE	OC6IE	OC5IE	IC6IE
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
IC5IE	IC4IE	IC3IE	DMA3IE	C1IE	C1RXIE	SPI2IE	SPI2EIE
bit 7							bit 0

### Legend:

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

- bit 15            **T6IE:** Timer6 Interrupt Enable bit  
                   1 = Interrupt request enabled  
                   0 = Interrupt request not enabled
- bit 14            **DMA4IE:** DMA Channel 4 Data Transfer Complete Interrupt Enable bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 13            **Unimplemented:** Read as '0'
- bit 12            **OC8IE:** Output Compare Channel 8 Interrupt Enable bit  
                   1 = Interrupt request enabled  
                   0 = Interrupt request not enabled
- bit 11            **OC7IE:** Output Compare Channel 7 Interrupt Enable bit  
                   1 = Interrupt request enabled  
                   0 = Interrupt request not enabled
- bit 10            **OC6IE:** Output Compare Channel 6 Interrupt Enable bit  
                   1 = Interrupt request enabled  
                   0 = Interrupt request not enabled
- bit 9             **OC5IE:** Output Compare Channel 5 Interrupt Enable bit  
                   1 = Interrupt request enabled  
                   0 = Interrupt request not enabled
- bit 8             **IC6IE:** Input Capture Channel 6 Interrupt Enable bit  
                   1 = Interrupt request enabled  
                   0 = Interrupt request not enabled
- 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             **DMA3IE:** DMA Channel 3 Data Transfer Complete Interrupt Enable bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 3             **C1IE:** ECAN1 Event Interrupt Enable bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred

---

**REGISTER 6-12: IEC2: INTERRUPT ENABLE CONTROL REGISTER 2 (CONTINUED)**

- bit 2      **C1RXIE:** ECAN1 Receive Data Ready Interrupt Enable bit  
            1 = Interrupt request has occurred  
            0 = Interrupt request has not occurred
- bit 1      **SPI2IE:** SPI2 Event Interrupt Enable bit  
            1 = Interrupt request enabled  
            0 = Interrupt request not enabled
- bit 0      **SPI2EIE:** SPI2 Error Interrupt Enable bit  
            1 = Interrupt request enabled  
            0 = Interrupt request not enabled

# PIC24H

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

U-0	U-0	R/W-0	U-0	U-0	U-0	U-0	R/W-0
—	—	DMA5IE	—	—	—	—	C2IE
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
C2RXIE	INT4IE	INT3IE	T9IE	T8IE	MI2C2IE	SI2C2IE	T7IE
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 15-14     **Unimplemented:** Read as '0'
- bit 13        **DMA5IE:** DMA Channel 5 Data Transfer Complete Interrupt Enable bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 12-9      **Unimplemented:** Read as '0'
- bit 8          **C2IE:** ECAN2 Event Interrupt Enable bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 7          **C2RXIE:** ECAN2 Receive Data Ready Interrupt Enable bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 6          **INT4IE:** External Interrupt 4 Enable bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 5          **INT3IE:** External Interrupt 3 Enable bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 4          **T9IE:** Timer9 Interrupt Enable bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 3          **T8IE:** Timer8 Interrupt Enable bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 2          **MI2C2IE:** I2C2 Master Events Interrupt Enable bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 1          **SI2C2IE:** I2C2 Slave Events Interrupt Enable bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 0          **T7IE:** Timer7 Interrupt Enable bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred

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

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15						bit 8	

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	U-0
C2TXIE	C1TXIE	DMA7IE	DMA6IE	—	U2EIE	U1EIE	—
bit 7						bit 0	

### Legend:

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

- bit 15-8        **Unimplemented:** Read as '0'
- bit 7            **C2TXIE:** ECAN2 Transmit Data Request Interrupt Enable bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 6            **C1TXIE:** ECAN1 Transmit Data Request Interrupt Enable bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 5            **DMA7IE:** DMA Channel 7 Data Transfer Complete Enable Status bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 4            **DMA6IE:** DMA Channel 6 Data Transfer Complete Enable Status bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 3            **Unimplemented:** Read as '0'
- bit 2            **U2EIE:** UART2 Error Interrupt Enable bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 1            **U1EIE:** UART1 Error Interrupt Enable bit  
                   1 = Interrupt request has occurred  
                   0 = Interrupt request has not occurred
- bit 0            **Unimplemented:** Read as '0'

# PIC24H

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

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	T1IP<2:0>			—	OC1IP<2:0>		
bit 15				bit 8			

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	IC1IP<2:0>			—	INT0IP<2:0>		
bit 7				bit 0			

### Legend:

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

- bit 15                      **Unimplemented:** Read as '0'
- bit 14-12                **T1IP<2:0>:** 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                **OC1IP<2:0>:** 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                 **IC1IP<2:0>:** 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                 **INT0IP<2:0>:** External Interrupt 0 Priority bits
  - 111 = Interrupt is priority 7 (highest priority interrupt)
  - 
  - 
  - 
  - 001 = Interrupt is priority 1
  - 000 = Interrupt source is disabled

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

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	T2IP<2:0>			—	OC2IP<2:0>		
bit 15				bit 8			

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	IC2IP<2:0>			—	DMA0IP<2:0>		
bit 7				bit 0			

### Legend:

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

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

bit 14-12 **T2IP<2:0>:** 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 **OC2IP<2:0>:** 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 **IC2IP<2:0>:** 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 **Unimplemented:** Read as '0'

bit 2-0 **DMA0IP<2:0>:** DMA Channel 0 Data Transfer Complete Interrupt Priority bits  
 111 = Interrupt is priority 7 (highest priority interrupt)  
 •  
 •  
 •  
 001 = Interrupt is priority 1  
 000 = Interrupt source is disabled

# PIC24H

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

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	U1RXIP<2:0>			—	SPI1IP<2:0>		
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	SPI1EIP<2:0>			—	T3IP<2:0>		
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 15      **Unimplemented:** Read as '0'
- bit 14-12    **U1RXIP<2:0>:** 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    **SPI1IP<2:0>:** 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    **SPI1EIP<2:0>:** SPI1 Error 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    **T3IP<2:0>:** Timer3 Interrupt Priority bits  
 111 = Interrupt is priority 7 (highest priority interrupt)  
 •  
 •  
 •  
 001 = Interrupt is priority 1  
 000 = Interrupt source is disabled



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

U-0	U-0	U-0	U-0	U-0	R/W-1	R/W-0	R/W-0
—	—	—	—	—	DMA1IP<2:0>		
bit 15					bit 8		

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	AD1IP<2:0>			—	U1TXIP<2:0>		
bit 7				bit 0			

### Legend:

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

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

bit 10-8 **DMA1IP<2:0>:** DMA Channel 1 Data Transfer Complete 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 **AD1IP<2:0>:** ADC1 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 **U1TXIP<2:0>:** UART1 Transmitter Interrupt Priority bits

111 = Interrupt is priority 7 (highest priority interrupt)

- 
- 
- 

001 = Interrupt is priority 1

000 = Interrupt source is disabled

# PIC24H

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

U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	CNIP<2:0>			—	—	—	—
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	MI2C1IP<2:0>			—	SI2C1IP<2:0>		
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 15      **Unimplemented:** Read as '0'
- bit 14-12    **CNIP<2:0>:** Change Notification Interrupt Priority bits  
 111 = Interrupt is priority 7 (highest priority interrupt)  
 •  
 •  
 •  
 001 = Interrupt is priority 1  
 000 = Interrupt source is disabled
- bit 11-7    **Unimplemented:** Read as '0'
- bit 6-4      **MI2C1IP<2:0>:** I2C1 Master Events 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      **SI2C1IP<2:0>:** I2C1 Slave Events Interrupt Priority bits  
 111 = Interrupt is priority 7 (highest priority interrupt)  
 •  
 •  
 •  
 001 = Interrupt is priority 1  
 000 = Interrupt source is disabled

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

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	IC8IP<2:0>			—	IC7IP<2:0>		
bit 15				bit 8			

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	AD2IP<2:0>			—	INT1IP<2:0>		
bit 7				bit 0			

### Legend:

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

- bit 15      **Unimplemented:** Read as '0'
- bit 14-12    **IC8IP<2:0>:** Input Capture Channel 8 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    **IC7IP<2:0>:** Input Capture Channel 7 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    **AD2IP<2:0>:** ADC2 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    **INT1IP<2:0>:** External Interrupt 1 Priority bits
  - 111 = Interrupt is priority 7 (highest priority interrupt)
  - 
  - 
  - 
  - 001 = Interrupt is priority 1
  - 000 = Interrupt source is disabled

# PIC24H

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

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	T4IP<2:0>			—	OC4IP<2:0>		
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	OC3IP<2:0>			—	DMA2IP<2:0>		
bit 7							bit 0

### Legend:

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

- bit 15                      **Unimplemented:** Read as '0'
- bit 14-12                **T4IP<2:0>:** 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                **OC4IP<2:0>:** 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                 **OC3IP<2:0>:** 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                    **Unimplemented:** Read as '0'
- bit 2-0                 **DMA2IP<2:0>:** DMA Channel 2 Data Transfer Complete Interrupt Priority bits
  - 111 = Interrupt is priority 7 (highest priority interrupt)
  - 
  - 
  - 
  - 001 = Interrupt is priority 1
  - 000 = Interrupt source is disabled

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

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	U2TXIP<2:0>			—	U2RXIP<2:0>		
bit 15				bit 8			

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	INT2IP<2:0>			—	T5IP<2:0>		
bit 7				bit 0			

**Legend:**

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

- bit 15      **Unimplemented:** Read as '0'
- bit 14-12    **U2TXIP<2:0>:** 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    **U2RXIP<2:0>:** 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     **INT2IP<2:0>:** 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     **T5IP<2:0>:** Timer5 Interrupt Priority bits
  - 111 = Interrupt is priority 7 (highest priority interrupt)
  - 
  - 
  - 
  - 001 = Interrupt is priority 1
  - 000 = Interrupt source is disabled

# PIC24H

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

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	C1IP<2:0>			—	C1RXIP<2:0>		
bit 15				bit 8			

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	SPI2IP<2:0>			—	SPI2EIP<2:0>		
bit 7				bit 0			

### Legend:

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

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

bit 14-12        **C1IP<2:0>:** ECAN1 Event 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         **C1RXIP<2:0>:** ECAN1 Receive Data Ready 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          **SPI2IP<2:0>:** 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          **SPI2EIP<2:0>:** SPI2 Error Interrupt Priority bits  
                   111 = Interrupt is priority 7 (highest priority interrupt)  
                   •  
                   •  
                   •  
                   001 = Interrupt is priority 1  
                   000 = Interrupt source is disabled

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

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	IC5IP<2:0>			—	IC4IP<2:0>		
bit 15				bit 8			

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	IC3IP<2:0>			—	DMA3IP<2:0>		
bit 7				bit 0			

### Legend:

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

- bit 15                      **Unimplemented:** Read as '0'
- bit 14-12                **IC5IP<2:0>:** 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                **IC4IP<2:0>:** 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                **IC3IP<2:0>:** 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                    **Unimplemented:** Read as '0'
- bit 2-0                **DMA3IP<2:0>:** DMA Channel 3 Data Transfer Complete Interrupt Priority bits
  - 111 = Interrupt is priority 7 (highest priority interrupt)
  - 
  - 
  - 
  - 001 = Interrupt is priority 1
  - 000 = Interrupt source is disabled

# PIC24H

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

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	OC7IP<2:0>			—	OC6IP<2:0>		
bit 15				bit 8			

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	OC5IP<2:0>			—	IC6IP<2:0>		
bit 7				bit 0			

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 15           **Unimplemented:** Read as '0'
- bit 14-12       **OC7IP<2:0>:** Output Compare Channel 7 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        **OC6IP<2:0>:** Output Compare Channel 6 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         **OC5IP<2:0>:** 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            **Unimplemented:** Read as '0'
- bit 2-0         **IC6IP<2:0>:** Input Capture Channel 6 Interrupt Priority bits  
111 = Interrupt is priority 7 (highest priority interrupt)  
•  
•  
•  
001 = Interrupt is priority 1  
000 = Interrupt source is disabled



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

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	T6IP<2:0>			—	DMA4IP<2:0>		
bit 15				bit 8			

U-0	U-0	U-0	U-0	U-0	R/W-1	R/W-0	R/W-0
—	—	—	—	—	OC8IP<2:0>		
bit 7				bit 0			

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

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

bit 14-12 **T6IP<2:0>:** Timer6 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 **DMA4IP<2:0>:** DMA Channel 4 Data Transfer Complete Interrupt Priority bits

111 = Interrupt is priority 7 (highest priority interrupt)

- 
- 
- 

001 = Interrupt is priority 1

000 = Interrupt source is disabled

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

bit 2-0 **OC8IP<2:0>:** Output Compare Channel 8 Interrupt Priority bits

111 = Interrupt is priority 7 (highest priority interrupt)

- 
- 
- 

001 = Interrupt is priority 1

000 = Interrupt source is disabled

# PIC24H

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

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	T8IP<2:0>			—	MI2C2IP<2:0>		
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	SI2C2IP<2:0>			—	T7IP<2:0>		
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 15           **Unimplemented:** Read as '0'
- bit 14-12       **T8IP<2:0>:** Timer8 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        **MI2C2IP<2:0>:** I2C2 Master Events 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         **SI2C2IP<2:0>:** I2C2 Slave Events 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         **T7IP<2:0>:** Timer7 Interrupt Priority bits  
 111 = Interrupt is priority 7 (highest priority interrupt)  
 •  
 •  
 •  
 001 = Interrupt is priority 1  
 000 = Interrupt source is disabled

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

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	C2RXIP<2:0>			—	INT4IP<2:0>		
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	INT3IP<2:0>			—	T9IP<2:0>		
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 15      **Unimplemented:** Read as '0'
- bit 14-12    **C2RXIP<2:0>:** ECAN2 Receive Data Ready 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    **INT4IP<2:0>:** External Interrupt 4 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     **INT3IP<2:0>:** External Interrupt 3 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     **T9IP<2:0>:** Timer9 Interrupt Priority bits  
 111 = Interrupt is priority 7 (highest priority interrupt)  
 •  
 •  
 •  
 001 = Interrupt is priority 1  
 000 = Interrupt source is disabled

# PIC24H

## REGISTER 6-29: IPC14: INTERRUPT PRIORITY CONTROL REGISTER 14

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	U-0	U-0	R/W-1	R/W-0	R/W-0
—	—	—	—	—	C2IP<2:0>		
bit 7					bit 0		

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15-3

**Unimplemented:** Read as '0'

bit 2-0

**C2IP<2:0>:** ECAN2 Event Interrupt Priority bits

111 = Interrupt is priority 7 (highest priority interrupt)

- 
- 
- 

001 = Interrupt is priority 1

000 = Interrupt source is disabled

## REGISTER 6-30: IPC15: INTERRUPT PRIORITY CONTROL REGISTER 15

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	DMA5IP<2:0>			—	—	—	—
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

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

bit 6-4      **DMA5IP<2:0>:** DMA Channel 5 Data Transfer Complete 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'

# PIC24H

## REGISTER 6-31: IPC16: INTERRUPT PRIORITY CONTROL REGISTER 16

U-0	U-0	U-0	U-0	U-0	R/W-1	R/W-0	R/W-0
—	—	—	—	—	U2EIP<2:0>		
bit 15						bit 8	

U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	U1EIP<2:0>			—	—	—	—
bit 7						bit 0	

### Legend:

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

- bit 15-11      **Unimplemented:** Read as '0'
- bit 10-8      **U2EIP<2:0>:** 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        **U1EIP<2:0>:** 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'

## REGISTER 6-32: IPC17: INTERRUPT PRIORITY CONTROL REGISTER 17

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	C2TXIP<2:0>			—	C1TXIP<2:0>		
bit 15				bit 8			

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	DMA7IP<2:0>			—	DMA6IP<2:0>		
bit 7				bit 0			

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

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

bit 14-12 **C2TXIP<2:0>:** ECAN2 Transmit Data Request 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 **C1TXIP<2:0>:** ECAN1 Transmit Data Request 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 **DMA7IP<2:0>:** DMA Channel 7 Data Transfer 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 **DMA6IP<2:0>:** DMA Channel 6 Data Transfer Complete Interrupt Priority bits

111 = Interrupt is priority 7 (highest priority interrupt)

•  
•  
•

001 = Interrupt is priority 1

000 = Interrupt source is disabled

# PIC24H

---

## 6.4 Interrupt Setup Procedures

### 6.4.1 INITIALIZATION

To configure an interrupt source:

1. Set the NSTDIS 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 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 IPCx 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 register.
4. Enable the interrupt source by setting the interrupt enable control bit associated with the source in the appropriate IECx 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 0x0E 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-level 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.



## 7.0 DIRECT MEMORY ACCESS (DMA)

**Note:** This data sheet summarizes the features of this group of PIC24H devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the “*dsPIC30F Family Reference Manual*” (DS70046).

Direct Memory Access (DMA) is a very efficient mechanism of copying data between peripheral SFRs (e.g., UART Receive register, Input Capture 1 buffer), and buffers or variables stored in RAM, with minimal CPU intervention. The DMA controller can automatically copy entire blocks of data without requiring the user software to read or write the peripheral Special Function Registers (SFRs) every time a peripheral interrupt occurs. The DMA controller uses a dedicated bus for data transfers and, therefore, does not steal cycles from the code execution flow of the CPU. To exploit the DMA capability, the corresponding user buffers or variables must be located in DMA RAM.

The PIC24H peripherals that can utilize DMA are listed in Table 7-1 along with their associated Interrupt Request (IRQ) numbers.

**TABLE 7-1: PERIPHERALS WITH DMA SUPPORT**

Peripheral	IRQ Number
INT0	0
Input Capture 1	1
Input Capture 2	5
Output Compare 1	2
Output Compare 2	6
Timer2	7
Timer3	8
SPI1	10
SPI2	33
UART1 Reception	11
UART1 Transmission	12
UART2 Reception	30
UART2 Transmission	31
ADC1	13
ADC2	21
ECAN1 Reception	34
ECAN1 Transmission	70
ECAN2 Reception	55
ECAN2 Transmission	71

The DMA controller features eight identical data transfer channels.

Each channel has its own set of control and status registers. Each DMA channel can be configured to copy data either from buffers stored in dual port DMA RAM to peripheral SFRs, or from peripheral SFRs to buffers in DMA RAM.

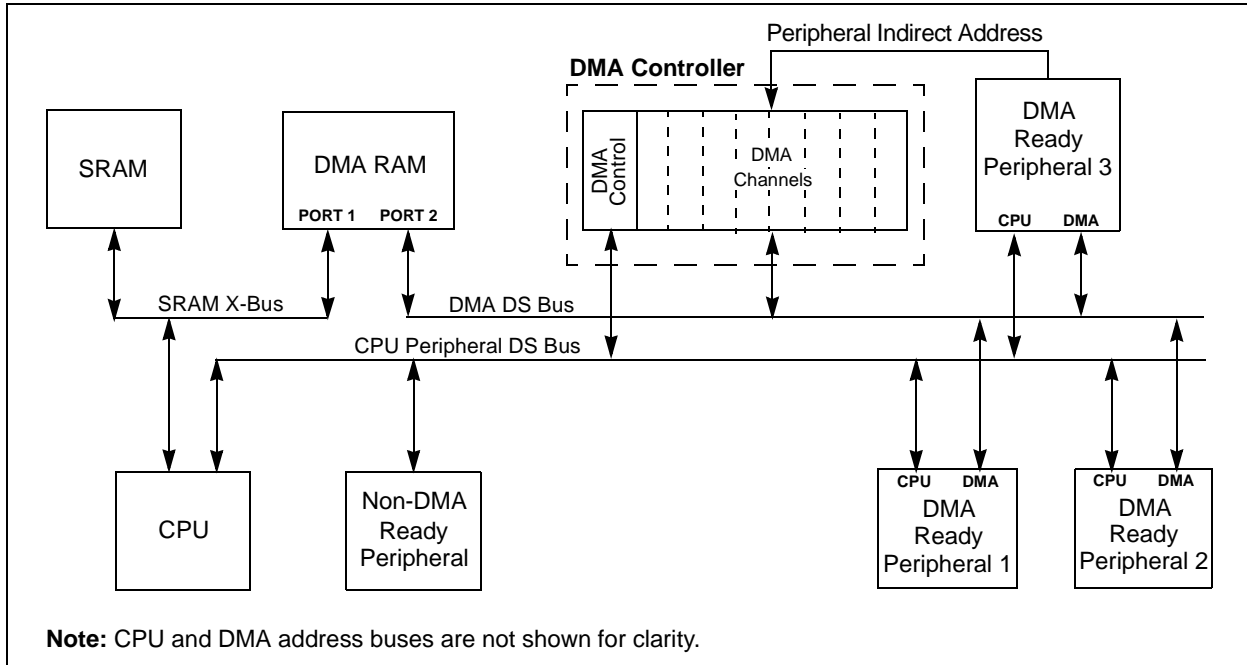
The DMA controller supports the following features:

- Word or byte sized data transfers.
- Transfers from peripheral to DMA RAM or DMA RAM to peripheral.
- Indirect Addressing of DMA RAM locations with or without automatic post-increment.
- Peripheral Indirect Addressing – In some peripherals, the DMA RAM read/write addresses may be partially derived from the peripheral.
- One-Shot Block Transfers – Terminating DMA transfer after one block transfer.
- Continuous Block Transfers – Reloading DMA RAM buffer start address after every block transfer is complete.
- Ping-Pong Mode – Switching between two DMA RAM start addresses between successive block transfers, thereby filling two buffers alternately.
- Automatic or manual initiation of block transfers
- Each channel can select from 19 possible sources of data sources or destinations.

For each DMA channel, a DMA interrupt request is generated when a block transfer is complete. Alternatively, an interrupt can be generated when half of the block has been filled.

# PIC24H

FIGURE 7-1: TOP LEVEL SYSTEM ARCHITECTURE USING A DEDICATED TRANSACTION BUS



## 7.1 DMAC Registers

Each DMAC Channel  $x$  ( $x = 0, 1, 2, 3, 4, 5, 6$  or  $7$ ) contains the following registers:

- A 16-bit DMA Channel Control register (DMAxCON)
- A 16-bit DMA Channel IRQ Select register (DMAxREQ)
- A 16-bit DMA RAM Primary Start Address register (DMAxSTA)
- A 16-bit DMA RAM Secondary Start Address register (DMAxSTB)
- A 16-bit DMA Peripheral Address register (DMAxPAD)
- A 10-bit DMA Transfer Count register (DMAxCNT)

An additional status register, DMACS, is common to all DMAC channels. It contains the DMA RAM and SFR write collision flags, XWCOL $x$  and PWCOL $x$ , respectively.

The DMAxCON, DMAxREQ, DMAxPAD and DMAxCNT are all conventional read/write registers. Reads of DMAxSTA or DMAxSTB will read the contents of the DMA RAM Address register. Writes to DMAxSTA or DMAxSTB write to the registers. This allows the user to determine the DMA buffer pointer value (address) at any time.

The interrupt flags (DMAxIF) are located in an IFS $x$  register in the interrupt controller. The corresponding interrupt enable control bits (DMAxIE) are located in an IEC $x$  register in the interrupt controller, and the corresponding interrupt priority control bits (DMAxIP) are located in an IPC $x$  register in the interrupt controller.

## 7.2 DMAC Operating Modes

Each DMA channel has its own status and control register (DMAxCON) that is used to configure the channel to support the following operating modes:

- Word or byte size data transfers
- Peripheral to DMA RAM or DMA RAM to peripheral transfers
- Post-increment or static DMA RAM address
- One-shot or continuous block transfers
- Auto-switch between two start addresses after each transfer complete (Ping-Pong mode)
- Force a single DMA transfer (Manual mode)

Each DMA channel can be independently configured to:

- Select from one of 19 DMA request sources
- Manually enable or disable the DMA channel
- Interrupt the CPU when the transfer is half or fully complete

DMA channel interrupts are routed to the interrupt controller module and enabled through associated enable flags.

The channel DMA RAM and peripheral write collision Faults are combined into a single DMAC error trap (Level 10) and are not maskable. Each channel has DMA RAM write collision (XWCOL $x$ ) and peripheral write collision (PWCOL $x$ ) status bits in the DMAC Status register (DMACS) to allow the DMAC error trap handler to determine the source of the Fault condition.

## 7.2.1 BYTE OR WORD TRANSFER

Each DMA channel can be configured to transfer words or bytes. As usual, words can only be moved to and from aligned (even) addresses. Bytes can be moved to or from any (legal) address.

If the SIZE bit (DMAxCON<14>) is clear, word sized data is transferred. The LSb of the DMA RAM Address register (DMAxSTA or DMAxSTB) is ignored. If Post-Increment Addressing mode is enabled, the DMA RAM Address register is incremented by 2 after every word transfer.

If the SIZE bit is set, byte sized data is transferred. If Post-Increment Addressing is enabled, the DMA RAM Address register is incremented by 1 after every byte transfer.

**Note:** DMAxCNT value is independent of data transfer size (byte/word). If an address offset is required, a 1-bit left shift of the counter is required to generate the correct offset for (aligned) word transfers.

## 7.2.2 ADDRESSING MODES

The DMAC supports Register Indirect and Register Indirect Post-Increment Addressing modes for DMA RAM addresses (source or destination). Each channel can select the DMA RAM Addressing mode independently. The Peripheral SFR is always accessed using Register Indirect Addressing.

If the AMODE<1:0> bits (DMAxCON<5:4>) are set to '01', Register Indirect Addressing without Post-Increment is used, which implies that the DMA RAM address remains constant.

If the AMODE<1:0> bits are clear, DMA RAM is accessed using Register Indirect Addressing with Post-Increment, which means the DMA RAM address will be incremented after every access.

Any DMA channel can be configured to operate in Peripheral Indirect Addressing mode by setting the AMODE<1:0> bits to '10'. In this mode, the DMA RAM source or destination address is partially derived from the peripheral as well as the DMA Address registers. Each peripheral module has a pre-assigned peripheral indirect address which is logically ORed with the DMA Start Address register to obtain the effective DMA RAM address. The DMA RAM Start Address register value must be aligned to a power-of-two boundary.

**Note:** Only the ECAN and A/D modules can utilize Peripheral Indirect Addressing.

## 7.2.3 DMA TRANSFER DIRECTION

Each DMA channel can be configured to transfer data from a peripheral to DMA RAM, or from DMA RAM to a peripheral.

If the DIR bit (DMAxCON<13>) is clear, the reads occur from a peripheral SFR (using the DMA Peripheral Address register, DMAxPAD) and the writes are directed to the DMA RAM (using the DMA RAM Address register).

If the DIR bit (DMAxCON<13>) is set, the reads occur from the DMA RAM (using the DMA RAM Address register) and the writes are directed to the peripheral (using the DMA Peripheral Address register, DMAxPAD).

## 7.2.4 NULL DATA PERIPHERAL WRITE MODE

If the NULLW bit (DMAxCON<11>) is set, a null data write to the peripheral SFR is performed in addition to a data transfer from the peripheral SFR to DMA RAM (assuming the DIR bit is clear). This mode is most useful in applications in which sequential reception of data is required without any data transmission.

## 7.2.5 CONTINUOUS OR ONE-SHOT OPERATION

Each DMA channel can be configured for One-Shot or Continuous mode operation.

If MODE<0> (DMAxCON<0>) is clear, the channel operates in Continuous mode.

When all data has been moved (i.e., buffer end has been detected), the channel is automatically reconfigured for subsequent use. During the last data transfer, the next Effective Address generated will be the original start address (from the selected DMAxSTA or DMAxSTB register). If the HALF bit (DMAxCON<12>) is clear, the transfer complete interrupt flag (DMAxIF) is set. If the HALF bit is set, DMAxIF will not be set at this time and the channel will remain enabled.

If MODE<0> is set, the channel operates in One-Shot mode. When all data has been moved (i.e., buffer end has been detected), the channel is automatically disabled. During the last data transfer, no new Effective Address is generated and the DMA RAM Address register retains the last DMA RAM address that was accessed. If the HALF bit is clear, the DMAxIF bit is set. If the HALF bit is set, the DMAxIF will not be set at this time and the channel is automatically disabled.

## 7.2.6 PING-PONG MODE

When the MODE<1> bit (DMAxCON<1>) is set by the user, Ping-Pong mode is enabled.

In this mode, successive block transfers alternately select DMAxSTA and DMAxSTB as the DMA RAM start address. In this way, a single DMA channel can be used to support two buffers of the same length in DMA RAM. Using this technique maximizes data throughput by allowing the CPU time to process one buffer while the other is being loaded.

## 7.2.7 MANUAL TRANSFER MODE

A manual DMA request can be created by setting the FORCE bit (DMAxREQ<15>) in software. If already enabled, the corresponding DMA channel executes a single data element transfer rather than a block transfer.

The FORCE bit is cleared by hardware when the forced DMA transfer is complete and cannot be cleared by the user. Any attempt to set this bit prior to completion of a DMA request that is underway will have no effect.

The manual DMA transfer function is a one-time event. The DMA channel always reverts to normal operation (i.e., based on hardware DMA requests) after a forced (manual) transfer.

This mode provides the user a straightforward method of initiating a block transfer. For example, using Manual mode to transfer the first data element into a serial peripheral allows subsequent data within the buffer to be moved automatically by the DMAC using a 'transmit buffer empty' DMA request.

## 7.2.8 DMA REQUEST SOURCE SELECTION

Each DMA channel can select between one of 128 interrupt sources to be a DMA request for that channel, based on the contents of the IRQSEL<6:0> bits (DMAxREQ<6:0>). The available interrupt sources are device dependent. Please refer to Table 7-1 for IRQ numbers associated with each of the interrupt sources that can generate a DMA transfer.

## 7.3 DMA Interrupts and Traps

Each DMA channel can generate an independent 'block transfer complete' (HALF = 0) or 'half block transfer complete' (HALF = 1) interrupt. Every DMA channel has its own interrupt vector and therefore, does not use the interrupt vector of the peripheral to which it is assigned. If a peripheral contains multi-word buffers, the buffering function must be disabled in the peripheral in order to use DMA. DMA interrupt requests are only generated by data transfers and not by peripheral error conditions.

The DMA controller can also react to peripheral and DMA RAM write collision error conditions through a nonmaskable CPU trap event. A DMA error trap is generated in either of the following Fault conditions:

- DMA RAM data write collision between the CPU and a peripheral
  - This condition occurs when the CPU and a peripheral attempt to write to the same DMA RAM address simultaneously
- Peripheral SFR data write collision between the CPU and the DMA controller
  - This condition occurs when the CPU and the DMA controller attempt to write to the same peripheral SFR simultaneously

The channel DMA RAM and peripheral write collision Faults are combined into a single DMAC error trap (Level 10) and are nonmaskable. Each channel has DMA RAM Write Collision (XWCOLx) and Peripheral Write Collision (PWCOLx) status bits in the DMAC Status register (DMACS) to allow the DMAC error trap handler to determine the source of the Fault condition.

## 7.4 DMA Initialization Example

The following is a DMA initialization example:

### EXAMPLE 7-1: DMA SAMPLE INITIALIZATION METHOD

```
// Clear all DMA controller status bits to a known state
DMACS0 = 0;

// Set up DMA Channel 0: Word mode, Read from Peripheral & Write to DMA; Interrupt when all the
data has been moved; Indirect with post-increment; Continuous mode with Ping-Pong Disabled
DMA0CON = 0x000D;

//Automatic DMA transfer initiation by DMA request; DMA Peripheral IRQ Number set up for ADC1
DMA0REQ = 0x0000;

// Set up offset into DMA RAM so that the buffer that collects A/D result data starts at the base
of DMA RAM
DMA0STA = 0x0000;

// DMA0PAD should be loaded with the address of the A/D conversion result register
DMA0PAD = (volatile unsigned int) &ADC1BUF0;

// DMA transfer of 256 words of data
DMA0CNT = 0x0100 ;

//Clear the DMA0 Interrupt Flag
IFS0bits.DMA0IF = 0;

//Enable DMA0 Interrupts
IEC0bits.DMA0IE = 1;

//Enable the DMA0 Channel
DMA0CONbits.CHEN = 1;
```

# PIC24H

## REGISTER 7-1: DMAxCON: DMA CHANNEL x CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0	U-0
CHEN	SIZE	DIR	HALF	NULLW	—	—	—
bit 15							bit 8

U-0	U-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0
—	—	AMODE<1:0>		—	—	MODE<1:0>	
bit 7							bit 0

### Legend:

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

- bit 15            **CHEN:** Channel Enable bit  
                   1 = Channel enabled  
                   0 = Channel disabled
- bit 14            **SIZE:** Data Transfer Size bit  
                   1 = Byte  
                   0 = Word
- bit 13            **DIR:** Transfer Direction bit (source/destination bus select)  
                   1 = Read from DMA RAM address, write to peripheral address  
                   0 = Read from peripheral address, write to DMA RAM address
- bit 12            **HALF:** Early Block Transfer Complete Interrupt Select bit  
                   1 = Initiate block transfer complete interrupt when half of the data has been moved  
                   0 = Initiate block transfer complete interrupt when all of the data has been moved
- bit 11            **NULLW:** Null Data Peripheral Write Mode Select bit  
                   1 = Null data write to peripheral in addition to DMA RAM write (DIR bit must also be clear)  
                   0 = Normal operation
- bit 10-6         **Unimplemented:** Read as '0'
- bit 5-4           **AMODE<1:0>:** DMA Channel Operating Mode Select bits  
                   11 = Reserved (will act as Peripheral Indirect Addressing mode)  
                   10 = Peripheral Indirect Addressing mode  
                   01 = Register Indirect without Post-Increment mode  
                   00 = Register Indirect with Post-Increment mode
- bit 3-2           **Unimplemented:** Read as '0'
- bit 1-0           **MODE<1:0>:** DMA Channel Operating Mode Select bits  
                   11 = One-Shot, Ping-Pong modes enabled (one block transfer from/to each DMA RAM buffer)  
                   10 = Continuous, Ping-Pong modes enabled  
                   01 = One-Shot, Ping-Pong modes disabled  
                   00 = Continuous, Ping-Pong modes disabled

## REGISTER 7-2: DMAxREQ: DMA CHANNEL x IRQ SELECT REGISTER

R/W-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
FORCE <sup>(1)</sup>	—	—	—	—	—	—	—
bit 15							bit 8

U-0	R/W-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0
—	IRQSEL6 <sup>(2)</sup>	IRQSEL5 <sup>(2)</sup>	IRQSEL4 <sup>(2)</sup>	IRQSEL3 <sup>(2)</sup>	IRQSEL2 <sup>(2)</sup>	IRQSEL1 <sup>(2)</sup>	IRQSEL0 <sup>(2)</sup>
bit 7							bit 0

### Legend:

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

bit 15                      **FORCE:** Force DMA Transfer bit<sup>(1)</sup>  
                                     1 = Force a single DMA transfer (Manual mode)  
                                     0 = Automatic DMA transfer initiation by DMA request

bit 14-7                      **Unimplemented:** Read as '0'

bit 6-0                      **IRQSEL<6:0>:** DMA Peripheral IRQ Number Select bits<sup>(2)</sup>  
                                     0000000-1111111 = DMAIRQ0-DMAIRQ127 selected to be Channel DMAREQ

**Note 1:** The FORCE bit cannot be cleared by the user. The FORCE bit is cleared by hardware when the forced DMA transfer is complete.

**2:** Please see Table 6-1 for a complete listing of IRQ numbers for all interrupt sources.

# PIC24H

## REGISTER 7-3: DMAxSTA: DMA CHANNEL x RAM START ADDRESS REGISTER A<sup>(1)</sup>

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STA<15:8>							
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STA<7:0>							
bit 7							bit 0

### Legend:

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

bit 15-0                      **STA<15:0>**: Primary DMA RAM Start Address bits (source or destination)

**Note 1:** A read of this address register will return the current contents of the DMA RAM Address register, not the contents written to STA<15:0>. If the channel is enabled (i.e., active), writes to this register may result in unpredictable behavior of the DMA channel and should be avoided.

## REGISTER 7-4: DMAxSTB: DMA CHANNEL x RAM START ADDRESS REGISTER B<sup>(1)</sup>

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STB<15:8>							
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STB<7:0>							
bit 7							bit 0

### Legend:

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

bit 15-0                      **STB<15:0>**: Secondary DMA RAM Start Address bits (source or destination)

**Note 1:** A read of this address register will return the current contents of the DMA RAM Address register, not the contents written to STB<15:0>. If the channel is enabled (i.e., active), writes to this register may result in unpredictable behavior of the DMA channel and should be avoided.



## REGISTER 7-5: DMAxPAD: DMA CHANNEL x PERIPHERAL ADDRESS REGISTER<sup>(1)</sup>

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PAD<15:8>							
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PAD<7:0>							
bit 7							bit 0

### Legend:

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

bit 15-0                      **PAD<15:0>**: Peripheral Address Register bits

**Note 1:** If the channel is enabled (i.e., active), writes to this register may result in unpredictable behavior of the DMA channel and should be avoided.

## REGISTER 7-6: DMAxCNT: DMA CHANNEL x TRANSFER COUNT REGISTER<sup>(1)</sup>

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
—	—	—	—	—	—	—	CNT<9:8> <sup>(2)</sup>
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CNT<7:0>							
bit 7							bit 0

### Legend:

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

bit 15-10                      **Unimplemented:** Read as '0'

bit 9-0                      **CNT<9:0>**: DMA Transfer Count Register bits<sup>(2)</sup>

**Note 1:** If the channel is enabled (i.e., active), writes to this register may result in unpredictable behavior of the DMA channel and should be avoided.

**2:** Number of DMA transfers = CNT<9:0> + 1.

# PIC24H

## REGISTER 7-7: DMACS0: DMA CONTROLLER STATUS REGISTER 0

R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
PWCOL7	PWCOL6	PWCOL5	PWCOL4	PWCOL3	PWCOL2	PWCOL1	PWCOL0
bit 15							bit 8

R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
XWCOL7	XWCOL6	XWCOL5	XWCOL4	XWCOL3	XWCOL2	XWCOL1	XWCOL0
bit 7							bit 0

### Legend:

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

- bit 15            **PWCOL7:** Channel 7 Peripheral Write Collision Flag bit  
                   1 = Write collision detected  
                   0 = No write collision detected
- bit 14            **PWCOL6:** Channel 6 Peripheral Write Collision Flag bit  
                   1 = Write collision detected  
                   0 = No write collision detected
- bit 13            **PWCOL5:** Channel 5 Peripheral Write Collision Flag bit  
                   1 = Write collision detected  
                   0 = No write collision detected
- bit 12            **PWCOL4:** Channel 4 Peripheral Write Collision Flag bit  
                   1 = Write collision detected  
                   0 = No write collision detected
- bit 11            **PWCOL3:** Channel 3 Peripheral Write Collision Flag bit  
                   1 = Write collision detected  
                   0 = No write collision detected
- bit 10            **PWCOL2:** Channel 2 Peripheral Write Collision Flag bit  
                   1 = Write collision detected  
                   0 = No write collision detected
- bit 9             **PWCOL1:** Channel 1 Peripheral Write Collision Flag bit  
                   1 = Write collision detected  
                   0 = No write collision detected
- bit 8             **PWCOL0:** Channel 0 Peripheral Write Collision Flag bit  
                   1 = Write collision detected  
                   0 = No write collision detected
- bit 7             **XWCOL7:** Channel 7 DMA RAM Write Collision Flag bit  
                   1 = Write collision detected  
                   0 = No write collision detected
- bit 6             **XWCOL6:** Channel 6 DMA RAM Write Collision Flag bit  
                   1 = Write collision detected  
                   0 = No write collision detected
- bit 5             **XWCOL5:** Channel 5 DMA RAM Write Collision Flag bit  
                   1 = Write collision detected  
                   0 = No write collision detected
- bit 4             **XWCOL4:** Channel 4 DMA RAM Write Collision Flag bit  
                   1 = Write collision detected  
                   0 = No write collision detected

---

**REGISTER 7-7: DMACS0: DMA CONTROLLER STATUS REGISTER 0 (CONTINUED)**

---

- bit 3      **XWCOL3:** Channel 3 DMA RAM Write Collision Flag bit  
            1 = Write collision detected  
            0 = No write collision detected
- bit 2      **XWCOL2:** Channel 2 DMA RAM Write Collision Flag bit  
            1 = Write collision detected  
            0 = No write collision detected
- bit 1      **XWCOL1:** Channel 1 DMA RAM Write Collision Flag bit  
            1 = Write collision detected  
            0 = No write collision detected
- bit 0      **XWCOL0:** Channel 0 DMA RAM Write Collision Flag bit  
            1 = Write collision detected  
            0 = No write collision detected

# PIC24H

## REGISTER 7-8: DMACS1: DMA CONTROLLER STATUS REGISTER 1

U-0	U-0	U-0	U-0	R-1	R-1	R-1	R-1
—	—	—	—	LSTCH<3:0>			
bit 15				bit 8			

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
PPST7	PPST6	PPST5	PPST4	PPST3	PPST2	PPST1	PPST0
bit 7							bit 0

### Legend:

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

- bit 15-12      **Unimplemented:** Read as '0'
- bit 11-8      **LSTCH<3:0>:** Last DMA Channel Active bits
  - 1111 = No DMA transfer has occurred since system Reset
  - 1110-1000 = Reserved
  - 0111 = Last data transfer was by DMA Channel 7
  - 0110 = Last data transfer was by DMA Channel 6
  - 0101 = Last data transfer was by DMA Channel 5
  - 0100 = Last data transfer was by DMA Channel 4
  - 0011 = Last data transfer was by DMA Channel 3
  - 0010 = Last data transfer was by DMA Channel 2
  - 0001 = Last data transfer was by DMA Channel 1
  - 0000 = Last data transfer was by DMA Channel 0
- bit 7          **PPST7:** Channel 7 Ping-Pong Mode Status Flag bit
  - 1 = DMA7STB register selected
  - 0 = DMA7STA register selected
- bit 6          **PPST6:** Channel 6 Ping-Pong Mode Status Flag bit
  - 1 = DMA6STB register selected
  - 0 = DMA6STA register selected
- bit 5          **PPST5:** Channel 5 Ping-Pong Mode Status Flag bit
  - 1 = DMA5STB register selected
  - 0 = DMA5STA register selected
- bit 4          **PPST4:** Channel 4 Ping-Pong Mode Status Flag bit
  - 1 = DMA4STB register selected
  - 0 = DMA4STA register selected
- bit 3          **PPST3:** Channel 3 Ping-Pong Mode Status Flag bit
  - 1 = DMA3STB register selected
  - 0 = DMA3STA register selected
- bit 2          **PPST2:** Channel 2 Ping-Pong Mode Status Flag bit
  - 1 = DMA2STB register selected
  - 0 = DMA2STA register selected
- bit 1          **PPST1:** Channel 1 Ping-Pong Mode Status Flag bit
  - 1 = DMA1STB register selected
  - 0 = DMA1STA register selected
- bit 0          **PPST0:** Channel 0 Ping-Pong Mode Status Flag bit
  - 1 = DMA0STB register selected
  - 0 = DMA0STA register selected

## REGISTER 7-9: DSADR: MOST RECENT DMA RAM ADDRESS

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
DSADR<15:8>							
bit 15				bit 8			

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
DSADR<7:0>							
bit 7				bit 0			

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15-0      **DSADR<15:0>**: Most Recent DMA RAM Address Accessed by DMA Controller bits

# PIC24H

---

NOTES:

## 8.0 OSCILLATOR CONFIGURATION

**Note:** This data sheet summarizes the features of this group of PIC24H devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the “dsPIC30F Family Reference Manual” (DS70046).

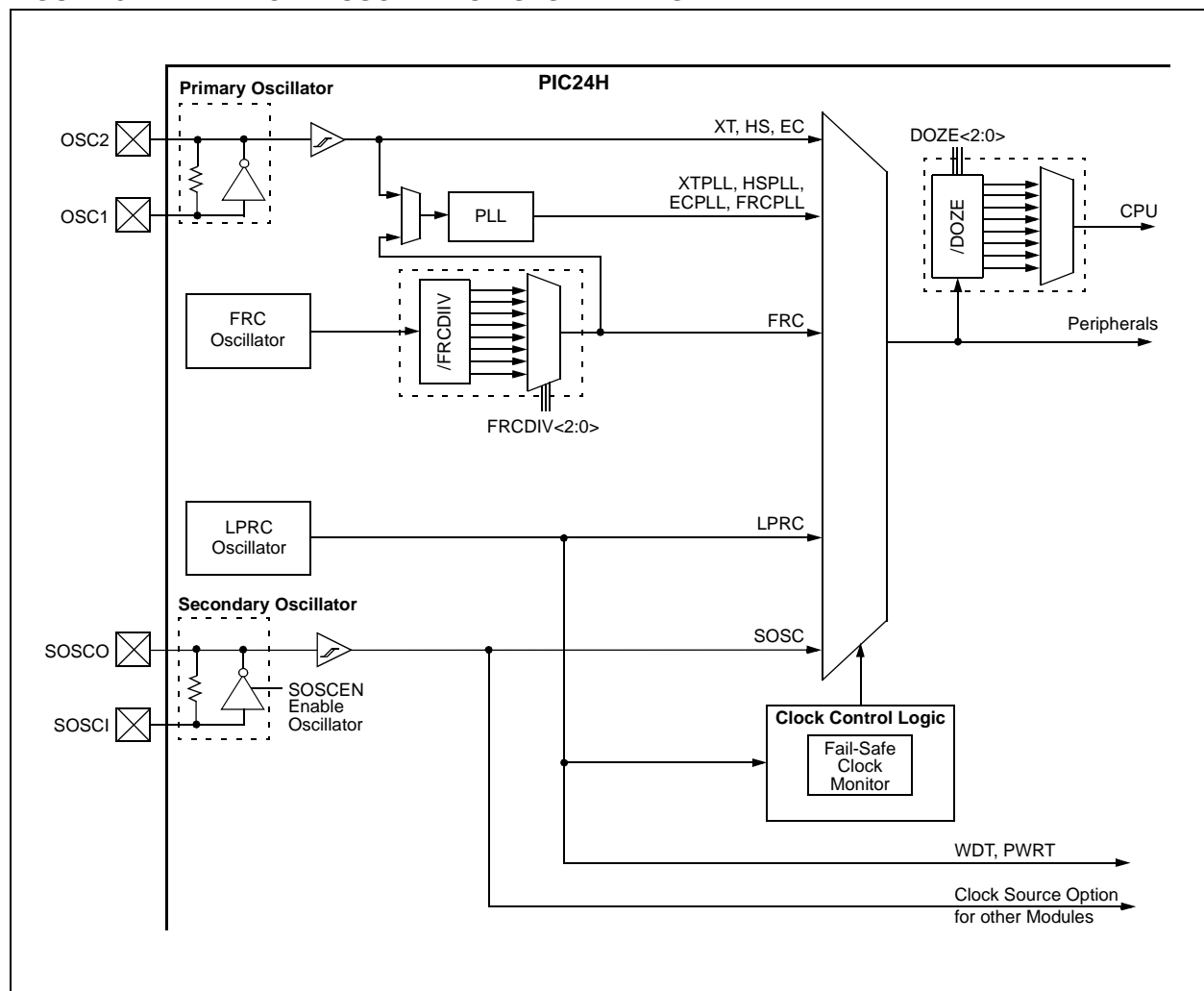
The PIC24H oscillator system provides:

- Various external and internal oscillator options as clock sources
- An on-chip PLL to scale the internal operating frequency to the required system clock frequency

- The internal FRC oscillator can also be used with the PLL, thereby allowing full-speed operation without any external clock generation hardware
- Clock switching between various clock sources
- Programmable clock postscaler for system power savings
- A Fail-Safe Clock Monitor (FSCM) that detects clock failure and takes fail-safe measures
- A Clock Control register (OSCCON)
- Nonvolatile Configuration bits for main oscillator selection.

A simplified diagram of the oscillator system is shown in Figure 8-1.

**FIGURE 8-1: PIC24H OSCILLATOR SYSTEM DIAGRAM**



## 8.1 CPU Clocking System

There are seven system clock options provided by the PIC24H:

- FRC Oscillator
- FRC Oscillator with PLL
- Primary (XT, HS or EC) Oscillator
- Primary Oscillator with PLL
- Secondary (LP) Oscillator
- LPRC Oscillator
- FRC Oscillator with postscaler

### 8.1.1 SYSTEM CLOCK SOURCES

The FRC (Fast RC) internal oscillator runs at a nominal frequency of 7.37 MHz. The user software can tune the FRC frequency. User software can optionally specify a factor (ranging from 1:2 to 1:256) by which the FRC clock frequency is divided. This factor is selected using the  $\text{FRCDIV}\langle 2:0 \rangle$  ( $\text{CLKDIV}\langle 10:8 \rangle$ ) bits.

The primary oscillator can use one of the following as its clock source:

1. XT (Crystal): Crystals and ceramic resonators in the range of 3 MHz to 10 MHz. The crystal is connected to the OSC1 and OSC2 pins.
2. HS (High-Speed Crystal): Crystals in the range of 10 MHz to 40 MHz. The crystal is connected to the OSC1 and OSC2 pins.
3. EC (External Clock): External clock signal in the range of 0.8 MHz to 64 MHz. The external clock signal is directly applied to the OSC1 pin.

The secondary (LP) oscillator is designed for low power and uses a 32.768 kHz crystal or ceramic resonator. The LP oscillator uses the SOSC1 and SOSCO pins.

The LPRC (Low-Power RC) internal oscillator runs at a nominal frequency of 32.768 kHz. It is also used as a reference clock by the Watchdog Timer (WDT) and Fail-Safe Clock Monitor (FSCM).

The clock signals generated by the FRC and primary oscillators can be optionally applied to an on-chip Phase Locked Loop (PLL) to provide a wide range of output frequencies for device operation. PLL configuration is described in **Section 8.1.3 “PLL Configuration”**.

### 8.1.2 SYSTEM CLOCK SELECTION

The oscillator source 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 20.1 “Configuration Bits”** for further details.) The Initial Oscillator Selection Configuration bits,  $\text{FNOSC}\langle 2:0 \rangle$  ( $\text{FOSCSEL}\langle 2:0 \rangle$ ), and the Primary Oscillator Mode Select Configuration bits,  $\text{POSCMD}\langle 1:0 \rangle$

( $\text{FOSC}\langle 1:0 \rangle$ ), select the oscillator source that is used at a Power-on Reset. The FRC primary oscillator is the default (unprogrammed) selection.

The Configuration bits allow users to choose between twelve different clock modes, shown in Table 8-1.

The output of the oscillator (or the output of the PLL if a PLL mode has been selected)  $\text{FOSC}$  is divided by 2 to generate the device instruction clock ( $\text{FCY}$ ).  $\text{FCY}$  defines the operating speed of the device, and speeds up to 40 MHz are supported by the PIC24H architecture.

Instruction execution speed or device operating frequency,  $\text{FCY}$ , is given by:

#### EQUATION 8-1: DEVICE OPERATING FREQUENCY

$$\text{FCY} = \text{FOSC}/2$$

### 8.1.3 PLL CONFIGURATION

The primary oscillator and internal FRC oscillator can optionally use an on-chip PLL to obtain higher speeds of operation. The PLL provides a significant amount of flexibility in selecting the device operating speed. A block diagram of the PLL is shown in Figure 8-2.

The output of the primary oscillator or FRC, denoted as 'FIN', is divided down by a prescale factor ( $\text{N1}$ ) of 2, 3, ... or 33 before being provided to the PLL's Voltage Controlled Oscillator (VCO). The input to the VCO must be selected to be in the range of 0.8 MHz to 8 MHz. Since the minimum prescale factor is 2, this implies that FIN must be chosen to be in the range of 1.6 MHz to 16 MHz. The prescale factor 'N1' is selected using the  $\text{PLLPRE}\langle 4:0 \rangle$  bits ( $\text{CLKDIV}\langle 4:0 \rangle$ ).

The PLL Feedback Divisor, selected using the  $\text{PLLDIV}\langle 8:0 \rangle$  bits ( $\text{PLLFBD}\langle 8:0 \rangle$ ), provides a factor 'M', by which the input to the VCO is multiplied. This factor must be selected such that the resulting VCO output frequency is in the range of 100 MHz to 200 MHz.

The VCO output is further divided by a postscale factor 'N2'. This factor is selected using the  $\text{PLLPOST}\langle 1:0 \rangle$  bits ( $\text{CLKDIV}\langle 7:6 \rangle$ ). 'N2' can be either 2, 4 or 8, and must be selected such that the PLL output frequency ( $\text{FOSC}$ ) is in the range of 12.5 MHz to 80 MHz, which generates device operating speeds of 6.25-40 MIPS.

For a primary oscillator or FRC oscillator, output 'FIN', the PLL output 'FOSC' is given by:

#### EQUATION 8-2: Fosc CALCULATION

$$\text{FOSC} = \text{FIN} * \left( \frac{\text{M}}{\text{N1} * \text{N2}} \right)$$



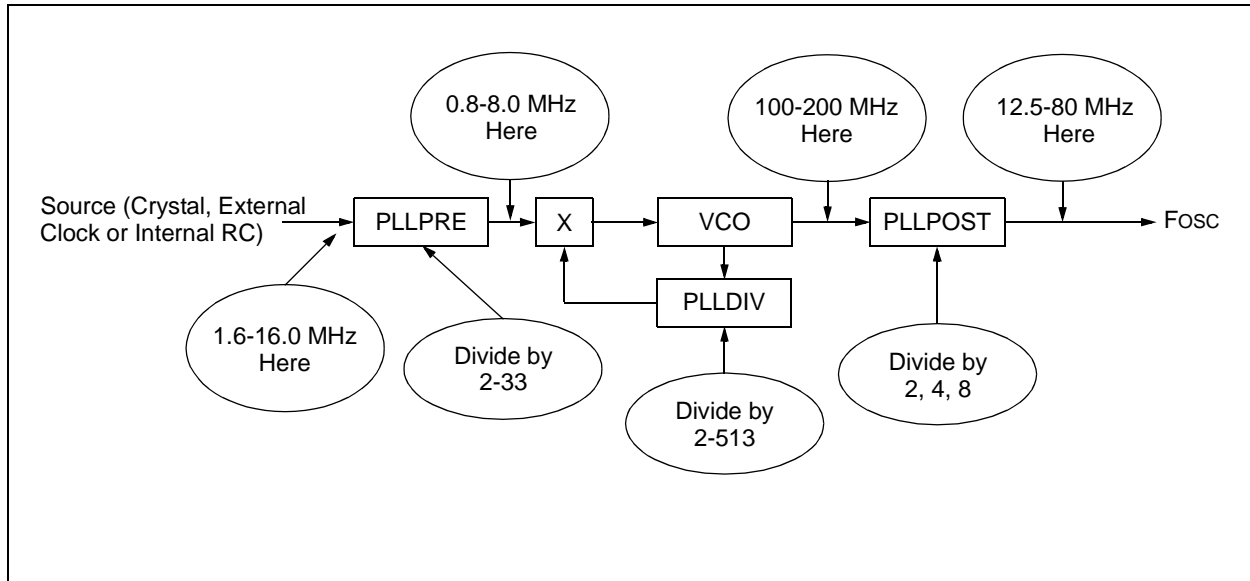
For example, suppose a 10 MHz crystal is being used, with "XT with PLL" being the selected oscillator mode. If PLLPRE<4:0> = 0, then N1 = 2. This yields a VCO input of 10/2 = 5 MHz, which is within the acceptable range of 0.8-8 MHz. If PLLDIV<8:0> = 0x1E, then M = 32. This yields a VCO output of 5 x 32 = 160 MHz, which is within the 100-200 MHz ranged needed.

If PLLPOST<1:0> = 0, then N2 = 2. This provides a Fosc of 160/2 = 80 MHz. The resultant device operating speed is 80/2 = 40 MIPS.

**EQUATION 8-3: XT WITH PLL MODE EXAMPLE**

$$F_{CY} = \frac{F_{OSC}}{2} = \frac{1}{2} \left( \frac{10000000 * 32}{2 * 2} \right) = 40 \text{ MIPS}$$

**FIGURE 8-2: PIC24H PLL BLOCK DIAGRAM**



**TABLE 8-1: CONFIGURATION BIT VALUES FOR CLOCK SELECTION**

Oscillator Mode	Oscillator Source	POSCMD<1:0>	FNOSC<2:0>	Note
Fast RC Oscillator (FRC)	Internal	11	111	1, 2
Reserved	Internal	11	110	1
Low-Power RC Oscillator (LPRC)	Internal	11	101	1
Secondary (Timer1) Oscillator (SOSC)	Secondary	11	100	1
Primary Oscillator (HS) with PLL (HSPLL)	Primary	10	011	
Primary Oscillator (XT) with PLL (XTPLL)	Primary	01	011	
Primary Oscillator (EC) with PLL (ECPLL)	Primary	00	011	1
Primary Oscillator (HS)	Primary	10	010	
Primary Oscillator (XT)	Primary	01	010	
Primary Oscillator (EC)	Primary	00	010	1
Fast RC Oscillator with PLL (FRCPLL)	Internal	11	001	1
Reserved	Internal	11	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.

# PIC24H

## REGISTER 8-1: OSCCON: OSCILLATOR CONTROL REGISTER

U-0	R-0	R-0	R-0	U-0	R/W-y	R/W-y	R/W-y
—	COSC<2:0>			—	NOSC<2:0>		
bit 15				bit 8			

R/W-0	U-0	R-0	U-0	R/C-0	U-0	R/W-0	R/W-0
CLKLOCK	—	LOCK	—	CF	—	LPOSCEN	OSWEN
bit 7						bit 0	

<b>Legend:</b>	y = Value set from Configuration bits on POR		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

- bit 15      **Unimplemented:** Read as '0'
- bit 14-12    **COSC<2:0>:** Current Oscillator Selection bits (read-only)  
 000 = Fast RC oscillator (FRC)  
 001 = Fast RC oscillator (FRC) with PLL  
 010 = Primary oscillator (XT, HS, EC)  
 011 = Primary oscillator (XT, HS, EC) with PLL  
 100 = Secondary oscillator (SOSC)  
 101 = Low-Power RC oscillator (LPRC)  
 110 = Reserved  
 111 = Fast RC oscillator (FRC)
- bit 11      **Unimplemented:** Read as '0'
- bit 10-8    **NOSC<2:0>:** New Oscillator Selection bits  
 000 = Reserved  
 001 = Fast RC oscillator (FRC) with PLL  
 010 = Primary oscillator (XT, HS, EC)  
 011 = Primary oscillator (XT, HS, EC) with PLL  
 100 = Secondary oscillator (SOSC)  
 101 = Low-Power RC oscillator (LPRC)  
 110 = Reserved  
 111 = Fast RC oscillator (FRC)
- bit 7      **CLKLOCK:** Clock Lock Enable bit  
 1 = If (FCKSM1 = 1), then clock and PLL configurations are locked.  
       If (FCKSM1 = 0), then clock and PLL configurations may be modified.  
 0 = Clock and PLL selections are not locked, configurations may be modified
- bit 6      **Unimplemented:** Read as '0'
- bit 5      **LOCK:** PLL Lock Status bit (read-only)  
 1 = Indicates that PLL is in lock, or PLL start-up timer is satisfied  
 0 = Indicates that PLL is out of lock, start-up timer is in progress or PLL is disabled
- bit 4      **Unimplemented:** Read as '0'
- bit 3      **CF:** Clock Fail Detect bit (read/clear by application)  
 1 = FSCM has detected clock failure  
 0 = FSCM has not detected clock failure
- bit 2      **Unimplemented:** Read as '0'
- bit 1      **LPOSCEN:** Secondary (LP) Oscillator Enable bit  
 1 = Enable secondary oscillator  
 0 = Disable secondary oscillator
- bit 0      **OSWEN:** Oscillator Switch Enable bit  
 1 = Request oscillator switch to selection specified by NOSC<2:0> bits  
 0 = Oscillator switch is complete

## REGISTER 8-2: CLKDIV: CLOCK DIVISOR REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0
ROI	DOZE<2:0>			DOZEN <sup>(1)</sup>	FRCDIV<2:0>		
bit 15							bit 8

R/W-0	R/W-1	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PLLPOST<1:0>		—	PLLPRE<4:0>				
bit 7							bit 0

<b>Legend:</b>	y = Value set from Configuration bits on POR		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

- bit 15      **ROI:** Recover on Interrupt bit  
 1 = Interrupts will clear the DOZEN bit and the processor clock/peripheral clock ratio is set to 1:1  
 0 = Interrupts have no effect on the DOZEN bit
- bit 14-12    **DOZE<2:0>:** Processor Clock Reduction Select bits  
 000 = Fcy/1 (default)  
 001 = Fcy/2  
 010 = Fcy/4  
 011 = Fcy/8  
 100 = Fcy/16  
 101 = Fcy/32  
 110 = Fcy/64  
 111 = Fcy/128
- bit 11      **DOZEN:** DOZE Mode Enable bit<sup>(1)</sup>  
 1 = DOZE<2:0> field specifies the ratio between the peripheral clocks and the processor clocks  
 0 = Processor clock/peripheral clock ratio forced to 1:1
- bit 10-8    **FRCDIV<2:0>:** Internal Fast RC Oscillator Postscaler bits  
 000 = FRC divide by 2  
 001 = FRC divide by 4  
 010 = FRC divide by 8  
 011 = FRC divide by 16 (default)  
 100 = FRC divide by 32  
 101 = FRC divide by 64  
 110 = FRC divide by 128  
 111 = FRC divide by 256
- bit 7-6    **PLLPOST<1:0>:** PLL VCO Output Divider Select bits (also denoted as 'N2', PLL postscaler)  
 00 = Output/2  
 01 = Output/4  
 10 = Reserved (defaults to output/4)  
 11 = Output/8
- bit 5      **Unimplemented:** Read as '0'
- bit 4-0    **PLLPRE<4:0>:** PLL Phase Detector Input Divider bits (also denoted as 'N1', PLL prescaler)  
 00000 = Input/2  
 00001 = Input/3  
 ...  
 11111 = Input/33

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

# PIC24H

## REGISTER 8-3: PLLFBD: PLL FEEDBACK DIVISOR REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
—	—	—	—	—	—	—	PLLDIV<8>
bit 15							bit 8

R/W-0	R/W-0	R/W-1	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0
PLLDIV<7:0>							
bit 7							bit 0

**Legend:**

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

bit 15-9                      **Unimplemented:** Read as '0'  
 bit 8-0                      **PLLDIV<8:0>:** PLL Feedback Divisor bits (also denoted as 'M', PLL multiplier)  
                                   000000000 = 2  
                                   000000001 = 3  
                                   000000010 = 4  
                                   •  
                                   •  
                                   •  
                                   111111111 = 513

## REGISTER 8-4: OSCTUN: FRC OSCILLATOR TUNING REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

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

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15-6

**Unimplemented:** Read as '0'

bit 5-0

**TUN<5:0>:** FRC Oscillator Tuning bits

011111 = Center frequency + 11.625% (8.23 MHz)

000110 = Center frequency + 11.25% (8.20 MHz)

•

•

•

000001 = Center frequency + 0.375% (7.40 MHz)

000000 = Center frequency (7.37 MHz nominal)

111111 = Center frequency – 0.375% (7.345 MHz)

•

•

•

100001 = Center frequency – 11.625% (6.52 MHz)

100000 = Center frequency – 12% (6.49 MHz)

## 8.2 Clock Switching Operation

Applications are free to switch between any of the four clock sources (Primary, LP, FRC and LPRC) under software control at any time. To limit the possible side effects that could result from this flexibility, PIC24H devices have a safeguard lock built into the switch process.

**Note:** Primary Oscillator mode has three different submodes (XT, HS and EC) which are determined by the POSCMD<1:0> 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.

### 8.2.1 ENABLING CLOCK SWITCHING

To enable clock switching, the FCKSM1 Configuration bit in the Configuration register must be programmed to '0'. (Refer to **Section 20.1 "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>) 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.

### 8.2.2 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 the 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 waits until the Oscillator Start-up Timer (OST) expires. If the new source is using the PLL, 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 LP (if LPOSCEN remains set).

**Note 1:** The processor continues 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.

## 8.3 Fail-Safe Clock Monitor (FSCM)

The Fail-Safe Clock Monitor (FSCM) allows the device to continue to operate even in the event of an oscillator failure. The FSCM function is enabled by programming. If the FSCM function is enabled, the LPRC internal oscillator runs at all times (except during Sleep mode) and is not subject to control by the Watchdog Timer.

In the event of an oscillator failure, the FSCM generates a clock failure trap event and switches the system clock over to the FRC oscillator. Then the application program can either attempt to restart the oscillator or execute a controlled shutdown. The trap can be treated as a warm Reset by simply loading the Reset address into the oscillator fail trap vector.

## 9.0 POWER-SAVING FEATURES

**Note:** This data sheet summarizes the features of this group of PIC24H devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the “*dsPIC30F Family Reference Manual*” (DS70046).

The PIC24H 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. PIC24H devices can 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.

### 9.1 Clock Frequency and Clock Switching

PIC24H devices allow 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 bits (OSCCON<10:8>). The process of changing a system clock during operation, as well as limitations to the process, are discussed in more detail in **Section 8.0 “Oscillator Configuration”**.

### 9.2 Instruction-Based Power-Saving Modes

PIC24H 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 9-1.

**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”.

#### 9.2.1 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 is 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 continues 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 is disabled in Sleep mode.

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

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

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

#### EXAMPLE 9-1: PWRSAV INSTRUCTION SYNTAX

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

# PIC24H

## 9.2.2 IDLE MODE

Idle mode has these features:

- The CPU stops 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 9.4 “Peripheral Module Disable”**).
- If the WDT or FSCM is enabled, the LPRC also remains 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 reapplied to the CPU and instruction execution begins immediately, starting with the instruction following the `PWRSVAV` instruction, or the first instruction in the ISR.

## 9.2.3 INTERRUPTS COINCIDENT WITH POWER SAVE INSTRUCTIONS

Any interrupt that coincides with the execution of a `PWRSVAV` instruction is held off until entry into Sleep or Idle mode has completed. The device then wakes up from Sleep or Idle mode.

## 9.3 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 `DOZE<2:0>` bits (`CLKDIV<14:12>`). There are eight possible configurations, from 1:1 to 1:128, with 1:1 being the default setting.

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.

For example, suppose the device is operating at 20 MIPS and the CAN module has been configured for 500 kbps based on this device operating speed. If the device is now placed in Doze mode with a clock frequency ratio of 1:4, the CAN module continues to communicate at the required bit rate of 500 kbps, but the CPU now starts executing instructions at a frequency of 5 MIPS.

## 9.4 Peripheral Module Disable

The Peripheral Module Disable (PMD) registers provide a method to disable a peripheral module by stopping all clock sources supplied to that module. When a peripheral is disabled via the appropriate PMD control bit, the peripheral is in a minimum power consumption state. The control and status registers associated with the peripheral are also disabled, so writes to those registers will have no effect and read values will be invalid.

A peripheral module is only enabled if both the associated bit in the PMD register is cleared and the peripheral is supported by the specific dsPIC® DSC variant. If the peripheral is present in the device, it is enabled in the PMD register by default.

**Note:** If a PMD bit is set, the corresponding module is disabled after a delay of 1 instruction cycle. Similarly, if a PMD bit is cleared, the corresponding module is enabled after a delay of 1 instruction cycle (assuming the module control registers are already configured to enable module operation).



## 10.0 I/O PORTS

**Note:** This data sheet summarizes the features of this group of PIC24H devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the “dsPIC30F Family Reference Manual” (DS70046).

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

### 10.1 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 10-1 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 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.

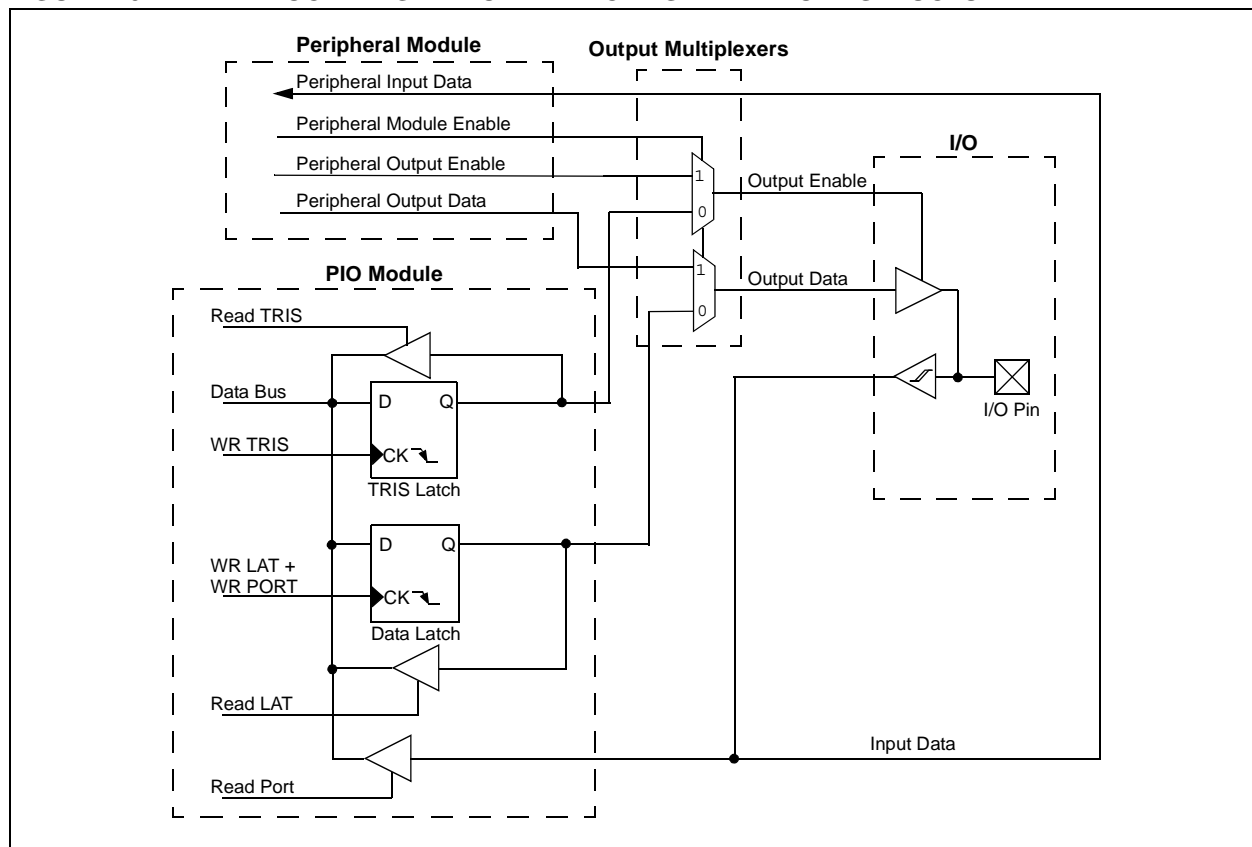
All port pins have three registers directly associated with their operation as digital I/O. The data direction register (TRISx) determines whether the pin is an input or an output. If the data direction bit is a '1', then the pin is an input. All port pins are defined as inputs after a Reset. Reads from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

**Note:** The voltage on a digital input pin can be between -0.3V to 5.6V.

**FIGURE 10-1: BLOCK DIAGRAM OF A TYPICAL SHARED PORT STRUCTURE**



# PIC24H

## 10.2 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 open-drain I/O feature is not supported on pins that have analog functionality multiplexed on the pin.) The maximum open-drain voltage allowed is the same as the maximum VIH specification. The open-drain output feature is supported for both port pin and peripheral configurations.

## 10.3 Configuring Analog Port Pins

The use of the ADxPCFGH, ADxPCFGL 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) is converted.

Clearing any bit in the ADxPCFGH or ADxPCFGL register configures the corresponding bit to be an analog pin. This is also the Reset state of any I/O pin that has an analog (ANx) function associated with it.

**Note:** In devices with two A/D modules, if the corresponding PCFG bit in either AD1PCFGH(L) and AD2PCFGH(L) is cleared, the pin is configured as an analog input.

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) can cause the input buffer to consume current that exceeds the device specifications.

**Note:** The voltage on an analog input pin can be between -0.3V to (VDD + 0.3 V).

### EXAMPLE 10-1: 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
```

## 10.4 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.

## 10.5 Input Change Notification

The input change notification function of the I/O ports allows the PIC24H 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 24 external signals (CN0 through CN23) that can 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 CN interrupt enable (CNxIE) 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 weak pull-up enable (CNxPUE) 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.

## 11.0 TIMER1

**Note:** This data sheet summarizes the features of this group of PIC24H devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the “dsPIC30F Family Reference Manual” (DS70046).

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:

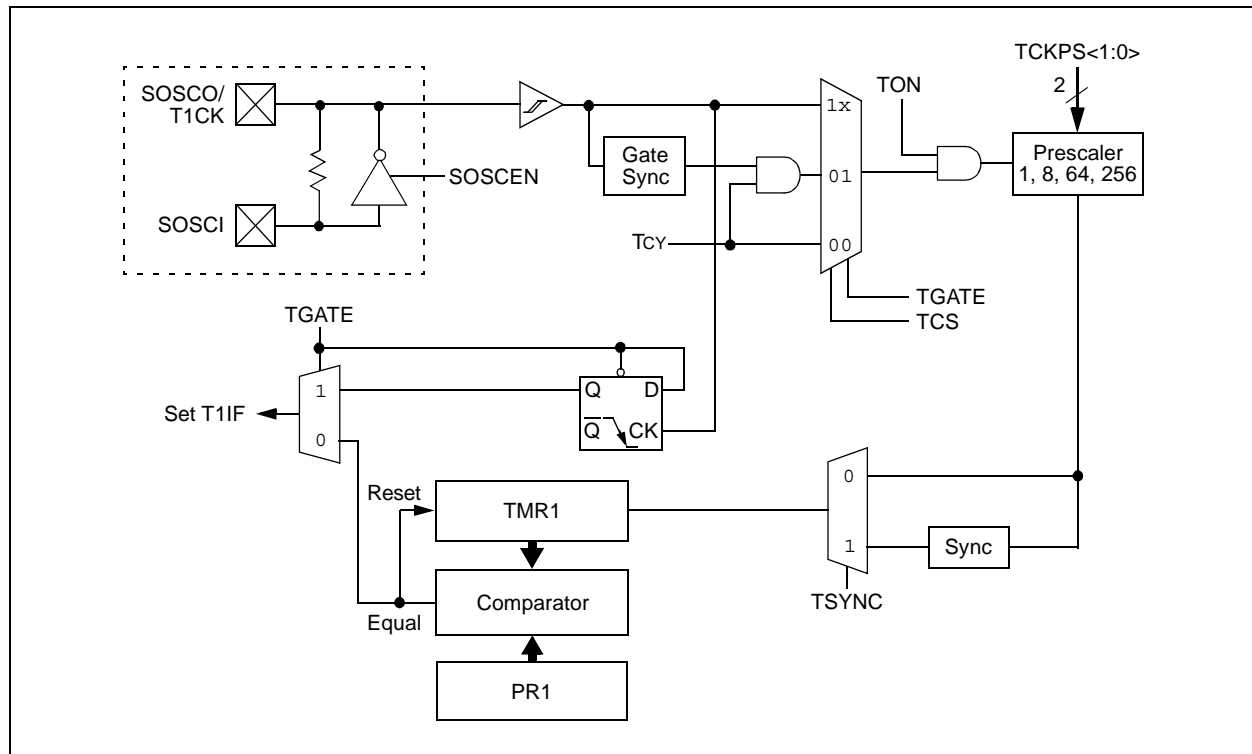
- Timer gate operation
- Selectable prescaler settings
- Timer operation during CPU Idle and Sleep modes
- Interrupt on 16-bit Period register match or falling edge of external gate signal

Figure 11-1 presents a block diagram of the 16-bit timer module.

To configure Timer1 for operation:

1. Set the TON bit (= 1) in the T1CON register.
2. Select the timer prescaler ratio using the TCKPS<1:0> bits in the T1CON register.
3. Set the Clock and Gating modes using the TCS and TGATE bits in the T1CON register.
4. Set or clear the TSYNC bit in T1CON to select 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, T1IP<2:0>, to set the interrupt priority.

**FIGURE 11-1: 16-BIT TIMER1 MODULE BLOCK DIAGRAM**



# PIC24H

## REGISTER 11-1: T1CON: TIMER1 CONTROL REGISTER

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

U-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	U-0
—	TGATE	TCKPS<1:0>		—	TSYNC	TCS	—
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 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 T1CS = 1:  
             This bit is ignored.  
             When T1CS = 0:  
             1 = Gated time accumulation enabled  
             0 = Gated time accumulation disabled
- bit 5-4     **TCKPS<1:0>:** 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 (Fcy)
- bit 0      **Unimplemented:** Read as '0'

## 12.0 TIMER2/3, TIMER4/5, TIMER6/7 AND TIMER8/9

**Note:** This data sheet summarizes the features of this group of PIC24H devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the “dsPIC30F Family Reference Manual” (DS70046).

The Timer2/3, Timer4/5, Timer6/7 and Timer8/9 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, Timer4/5, Timer6/7 and Timer8/9 operate in three modes:

- Two Independent 16-bit Timers (e.g., 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
- Time Base for Input Capture and Output Compare Modules (Timer2 and Timer3 only)
- ADC Event Trigger (Timer2/3 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 event trigger; this is implemented only with Timer2/3. The operating modes and enabled features are determined by setting the appropriate bit(s) in the T2CON, T3CON, T4CON, T5CON, T6CON, T7CON, T8CON and T9CON registers. T2CON, T4CON, T6CON and T8CON are shown in generic form in Register 12-1. T3CON, T5CON, T7CON and T9CON are shown in Register 12-2.

For 32-bit timer/counter operation, Timer2, Timer4, Timer6 or Timer8 is the least significant word; Timer3, Timer5, Timer7 or Timer9 is the most significant word of the 32-bit timers.

**Note:** For 32-bit operation, T3CON, T5CON, T7CON and T9CON control bits are ignored. Only T2CON, T4CON, T6CON and T8CON control bits are used for setup and control. Timer2, Timer4, Timer6 and Timer8 clock and gate inputs are utilized for the 32-bit timer modules, but an interrupt is generated with the Timer3, Timer5, Timer7 and Timer9 interrupt flags.

To configure Timer2/3, Timer4/5, Timer6/7 or Timer8/9 for 32-bit operation:

1. Set the corresponding T32 control bit.
2. Select the prescaler ratio for Timer2, Timer4, Timer6 or Timer8 using the TCKPS<1:0> bits.
3. Set the Clock and Gating modes using the corresponding TCS and TGATE bits.
4. Load the timer period value. PR3, PR5, PR7 or PR9 contains the most significant word of the value, while PR2, PR4, PR6 or PR8 contains the least significant word.
5. If interrupts are required, set the interrupt enable bit, T3IE, T5IE, T7IE or T9IE. Use the priority bits, T3IP<2:0>, T5IP<2:0>, T7IP<2:0> or T9IP<2:0>, to set the interrupt priority. While Timer2, Timer4, Timer6 or Timer8 control the timer, the interrupt appears as a Timer3, Timer5, Timer7 or Timer9 interrupt.
6. Set the corresponding TON bit.

The timer value at any point is stored in the register pair, TMR3:TMR2, TMR5:TMR4, TMR7:TMR6 or TMR9:TMR8. TMR3, TMR5, TMR7 or TMR9 always contains the most significant word of the count, while TMR2, TMR4, TMR6 or TMR8 contains the least significant word.

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

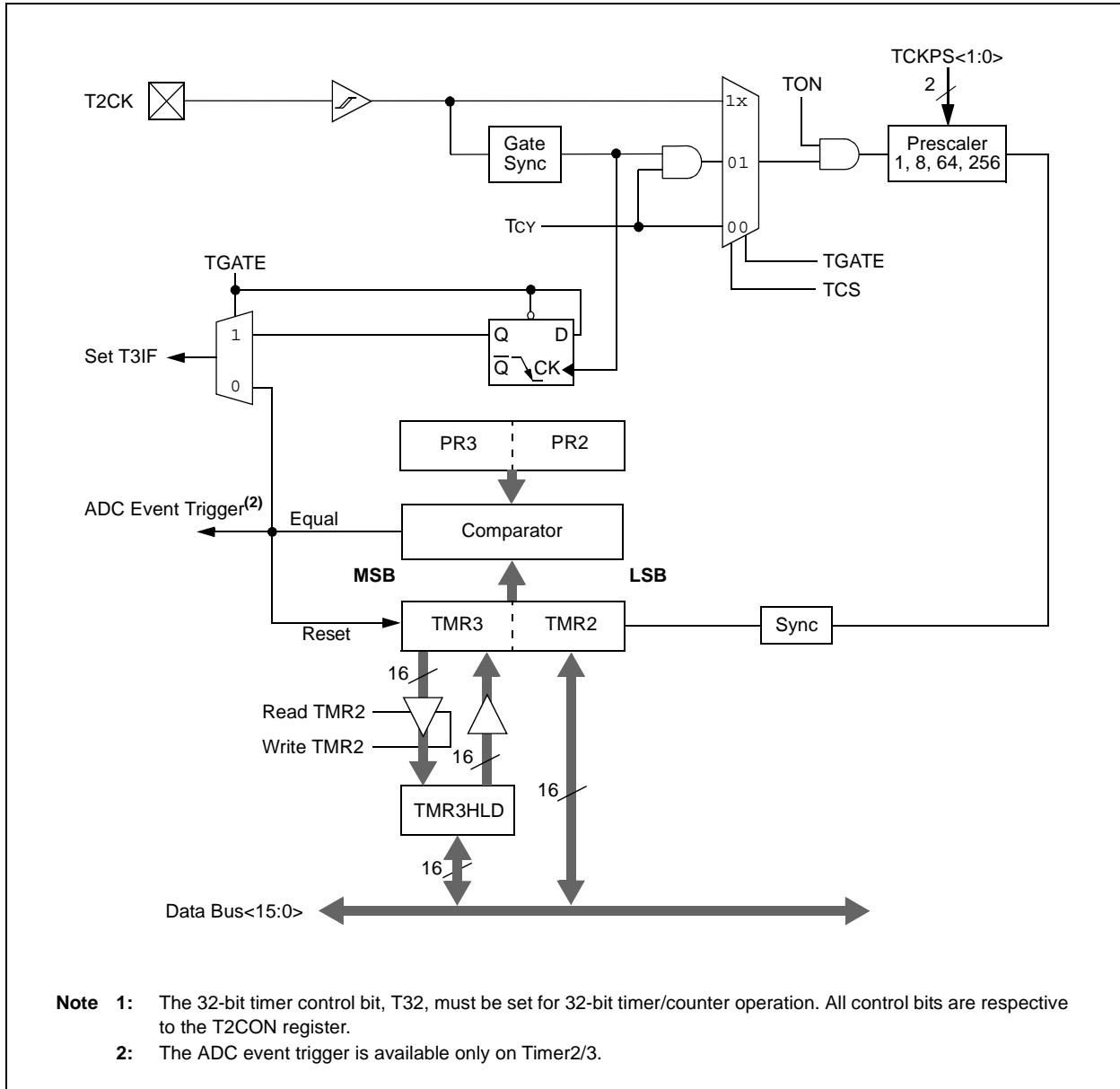
1. Clear the T32 bit corresponding to that timer.
2. Select the timer prescaler ratio using the TCKPS<1:0> bits.
3. Set the Clock and Gating modes using the TCS and TGATE bits.
4. Load the timer period value into the PRx register.
5. If interrupts are required, set the interrupt enable bit, TxIE. Use the priority bits, TxIP<2:0>, to set the interrupt priority.
6. Set the TON bit.

A block diagram for a 32-bit timer pair (Timer4/5) example is shown in Figure 12-1 and a timer (Timer4) operating in 16-bit mode example is shown in Figure 12-2.

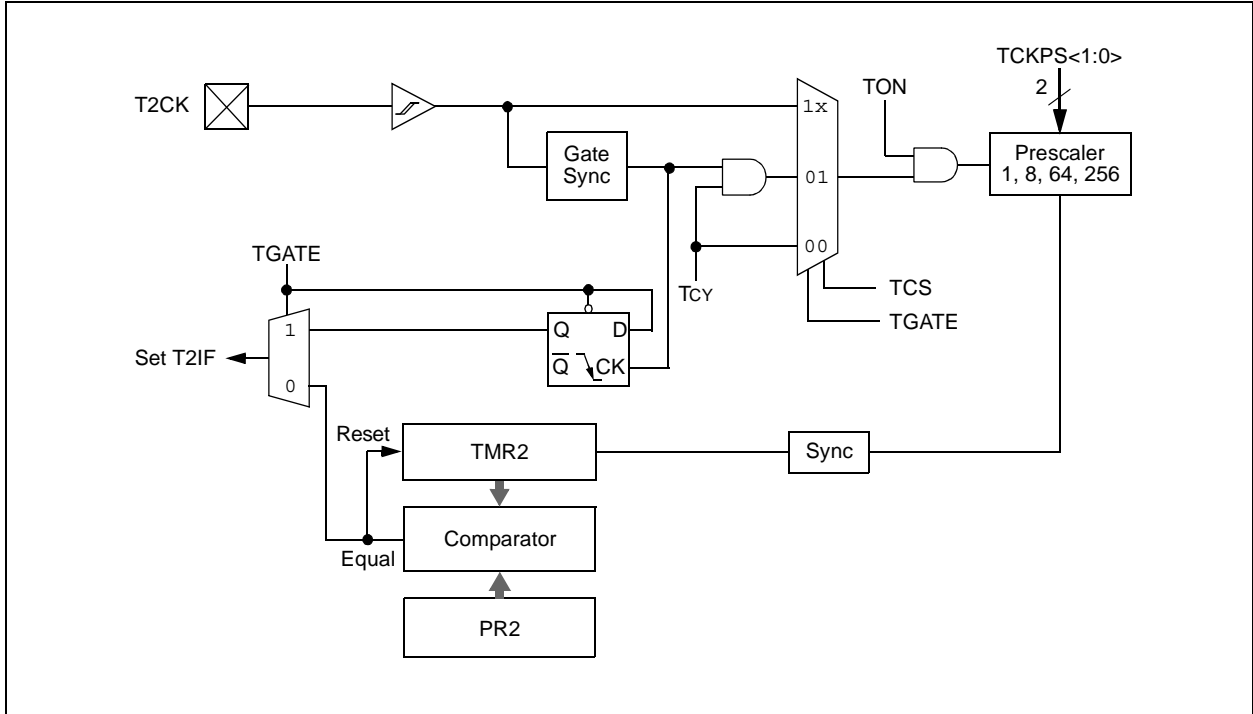
**Note:** Only Timer2 and Timer3 can trigger a DMA data transfer.

# PIC24H

FIGURE 12-1: TIMER2/3 (32-BIT) BLOCK DIAGRAM<sup>(1)</sup>



**FIGURE 12-2: TIMER2 (16-BIT) BLOCK DIAGRAM**



# PIC24H

## REGISTER 12-1: TxCON (T2CON, T4CON, T6CON OR T8CON) CONTROL REGISTER

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

U-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	U-0
—	TGATE	TCKPS<1:0>		T32 <sup>(1)</sup>	—	TCS	—
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 15      **TON:** Timerx On bit  
             When T32 = 1:  
             1 = Starts 32-bit Timerx/y  
             0 = Stops 32-bit Timerx/y  
             When T32 = 0:  
             1 = Starts 16-bit Timerx  
             0 = Stops 16-bit Timerx
- 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:** Timerx 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     **TCKPS<1:0>:** Timerx Input Clock Prescale Select bits  
             11 = 1:256  
             10 = 1:64  
             01 = 1:8  
             00 = 1:1
- bit 3      **T32:** 32-bit Timer Mode Select bit<sup>(1)</sup>  
             1 = Timerx and Timery form a single 32-bit timer  
             0 = Timerx and Timery act as two 16-bit timers
- bit 2      **Unimplemented:** Read as '0'
- bit 1      **TCS:** Timerx Clock Source Select bit  
             1 = External clock from pin TxCK (on the rising edge)  
             0 = Internal clock (FCY)
- bit 0      **Unimplemented:** Read as '0'

**Note 1:** In 32-bit mode, T3CON control bits do not affect 32-bit timer operation.



## REGISTER 12-2: TyCON (T3CON, T5CON, T7CON OR T9CON) CONTROL REGISTER

R/W-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
TON <sup>(1)</sup>	—	TSIDL <sup>(1)</sup>	—	—	—	—	—
bit 15							bit 8

U-0	R/W-0	R/W-0	R/W-0	U-0	U-0	R/W-0	U-0
—	TGATE <sup>(1)</sup>	TCKPS<1:0> <sup>(1)</sup>		—	—	TCS <sup>(1)</sup>	—
bit 7							bit 0

### Legend:

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

- bit 15        **TON:** Timery On bit<sup>(1)</sup>  
               1 = Starts 16-bit Timery  
               0 = Stops 16-bit Timery
- bit 14        **Unimplemented:** Read as '0'
- bit 13        **TSIDL:** Stop in Idle Mode bit<sup>(1)</sup>  
               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<sup>(1)</sup>  
               When TCS = 1:  
               This bit is ignored.  
               When TCS = 0:  
               1 = Gated time accumulation enabled  
               0 = Gated time accumulation disabled
- bit 5-4     **TCKPS<1:0>:** Timer3 Input Clock Prescale Select bits<sup>(1)</sup>  
               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<sup>(1)</sup>  
               1 = External clock from pin TyCK (on the rising edge)  
               0 = Internal clock (Fcy)
- 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.

# PIC24H

---

NOTES:

## 13.0 INPUT CAPTURE

**Note:** This data sheet summarizes the features of this group of PIC24H devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the “dsPIC30F Family Reference Manual” (DS70046).

The input capture module is useful in applications requiring frequency (period) and pulse measurement. The PIC24H devices support up to eight input capture channels.

The input capture module captures the 16-bit value of the selected Time Base register when an event occurs at the ICx pin. The events that cause a capture event are listed below in three categories:

1. Simple Capture Event modes
  - Capture timer value on every falling edge of input at ICx pin
  - Capture timer value on every rising edge of input at ICx pin
2. Capture timer value on every edge (rising and falling)
3. Prescaler Capture Event modes
  - Capture timer value on every 4th rising edge of input at ICx pin
  - Capture timer value on every 16th rising edge of input at ICx pin

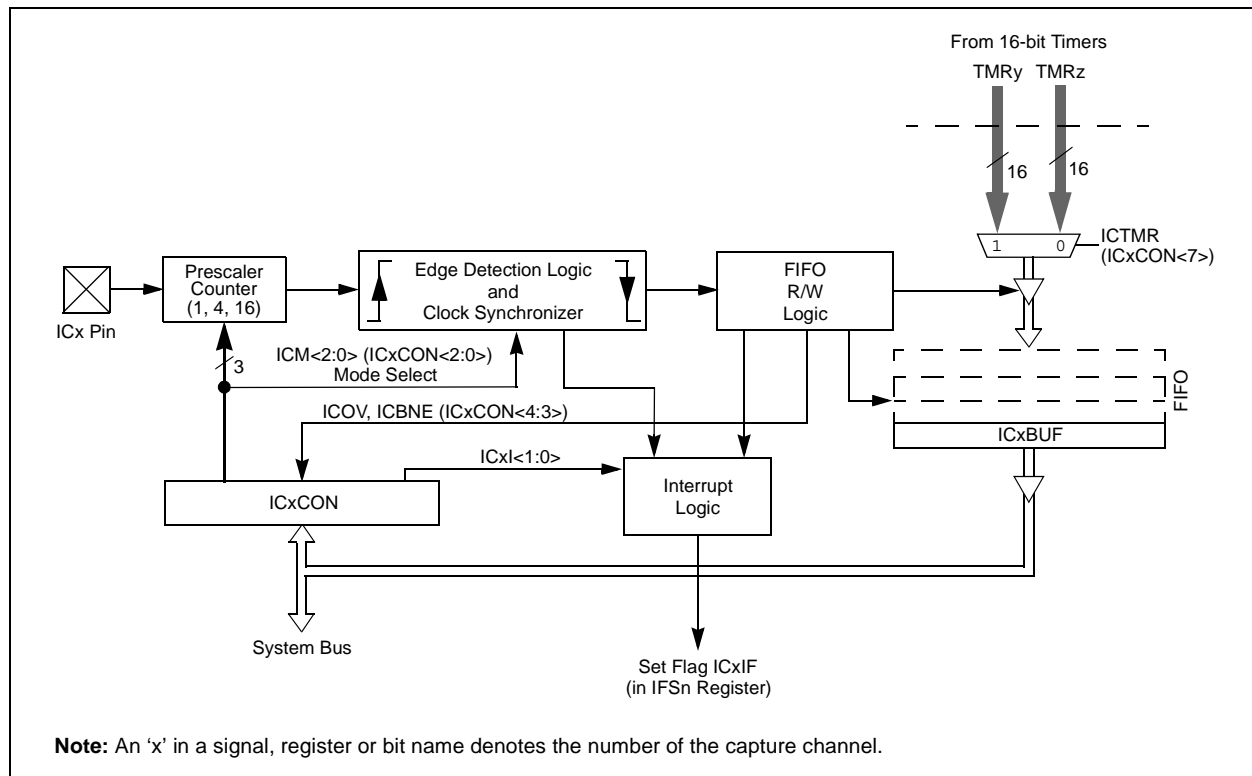
Each input capture channel can select between one of two 16-bit timers (Timer2 or Timer3) for the time base. The selected timer can use either an internal or external clock.

Other operational features include:

- Device wake-up from capture pin during CPU Sleep and Idle modes
- Interrupt on input capture event
- 4-word FIFO buffer for capture values
  - Interrupt optionally generated after 1, 2, 3 or 4 buffer locations are filled
- Input capture can also be used to provide additional sources of external interrupts

**Note:** Only IC1 and IC2 can trigger a DMA data transfer. If DMA data transfers are required, the FIFO buffer size must be set to 1 (IC1<1:0> = 00).

**FIGURE 13-1: INPUT CAPTURE BLOCK DIAGRAM**



# PIC24H

## 13.1 Input Capture Registers

**REGISTER 13-1: ICxCON: INPUT CAPTURE x CONTROL REGISTER**

U-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
—	—	ICSIDL	—	—	—	—	—
bit 15							bit 8

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 <sup>(1)</sup>	ICI<1:0>		ICOV	ICBNE	ICM<2:0>		
bit 7							bit 0

**Legend:**

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

- bit 15-14     **Unimplemented:** Read as '0'
- bit 13       **ICSIDL:** Input Capture 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 Timer Select bits<sup>(1)</sup>  
               1 = TMR2 contents are captured on capture event  
               0 = TMR3 contents are captured on capture event
- bit 6-5      **ICI<1:0>:** 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 Overflow Status Flag bit (read-only)  
               1 = Input capture overflow occurred  
               0 = No input capture overflow occurred
- bit 3         **ICBNE:** Input Capture Buffer Empty Status bit (read-only)  
               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      **ICM<2:0>:** Input Capture 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> bits do not control interrupt generation for this mode.)  
               000 = Input capture module turned off

**Note 1:** Timer selections may vary. Refer to the device data sheet for details.

## 14.0 OUTPUT COMPARE

**Note:** This data sheet summarizes the features of this group of PIC24H devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the “*dsPIC30F Family Reference Manual*” (DS70046).

### 14.1 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 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 Output Compare register, OCxR, and the Output Compare Secondary register, OCxRS, respectively.
5. Set Timer Period register, PRy, to value equal to or greater than value in OCxRS, the Output Compare Secondary 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 Output Compare Secondary 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.

### 14.2 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 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 Output Compare register, OCxR, and the Output Compare Secondary register, OCxRS, respectively.
5. Set Timer Period register, PRy, to value equal to or greater than value in OCxRS, the Output Compare Secondary 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.
7. Enable the compare time base by setting the TON (TyCON<15>) bit to '1'.
8. Upon the first match between TMRy and OCxR, the OCx pin will be driven high.
9. When the compare time base, TMRy, matches the Output Compare Secondary register, OCxRS, the second and trailing edge (high-to-low) of the pulse is driven onto the OCx pin.
10. As a result of the second compare match event, the OCxIF interrupt flag bit set.
11. When the compare time base and the value in its respective Timer Period register match, the TMRy register resets to 0x0000 and resumes counting.
12. Steps 8 through 11 are repeated and a continuous stream of pulses is generated, indefinitely. The OCxIF flag is set on each OCxRS-TMRy compare match event.

# PIC24H

## 14.3 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 OxCR 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 = 1 (TxCON<15>).

**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 Output Compare Secondary register, OCxRS, will not be transferred into OCxR until a time base period match occurs.

### 14.3.1 PWM PERIOD

The PWM period is specified by writing to PRy, the Timer Period register. The PWM period can be calculated using Equation 14-1:

### EQUATION 14-1: CALCULATING THE PWM PERIOD

$$\text{PWM Period} = [\text{PRy} + 1] \cdot \text{TCY} \cdot (\text{Timer Prescale Value})$$

where:  
 $\text{PWM Frequency} = 1/[\text{PWM Period}]$

**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 eight time base cycles.

### 14.3.2 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 glitchless PWM operation. In the PWM mode, OCxR is a read-only register.

Some important boundary parameters of the PWM duty cycle include:

- If the Output Compare 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 14-1 for PWM mode timing details. Table 14-1 shows example PWM frequencies and resolutions for a device operating at 10 MIPS.

### EQUATION 14-2: CALCULATION FOR MAXIMUM PWM RESOLUTION

$$\text{Maximum PWM Resolution (bits)} = \frac{\log_{10}\left(\frac{\text{FCY}}{\text{FPWM}}\right)}{\log_{10}(2)} \text{ bits}$$

### EXAMPLE 14-1: PWM PERIOD AND DUTY CYCLE CALCULATIONS

1. Find the Timer Period register value for a desired PWM frequency that is 52.08 kHz, where FCY = 16 MHz and a Timer2 prescaler setting of 1:1.  
 $\text{TCY} = 62.5 \text{ ns}$   
 $\text{PWM Period} = 1/\text{PWM Frequency} = 1/52.08 \text{ kHz} = 19.2 \mu\text{s}$   
 $\text{PWM Period} = (\text{PR2} + 1) \cdot \text{TCY} \cdot (\text{Timer2 Prescale Value})$   
 $19.2 \mu\text{s} = (\text{PR2} + 1) \cdot 62.5 \text{ ns} \cdot 1$   
 $\text{PR2} = 306$
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:  
 $\text{PWM Resolution} = \log_{10}(\text{FCY}/\text{FPWM})/\log_{10}(2) \text{ bits}$   
 $= (\log_{10}(16 \text{ MHz}/52.08 \text{ kHz})/\log_{10}(2)) \text{ bits}$   
 $= 8.3 \text{ bits}$

**TABLE 14-1: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 4 MIPS (F<sub>CY</sub> = 4 MHz)**

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

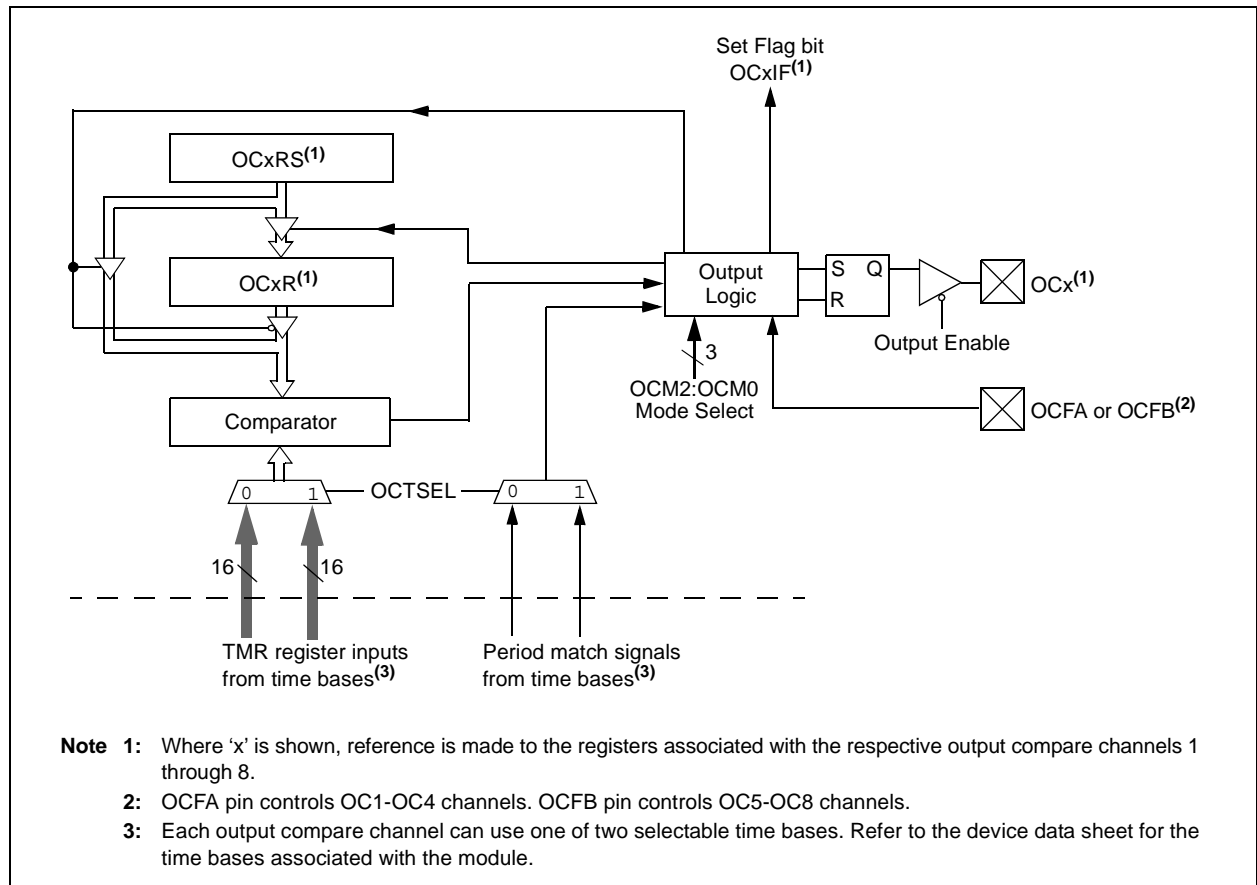
**TABLE 14-2: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 16 MIPS (F<sub>CY</sub> = 16 MHz)**

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

**TABLE 14-3: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 40 MIPS (F<sub>CY</sub> = 40 MHz)**

PWM Frequency	76 Hz	610 Hz	1.22 Hz	9.77 kHz	39 kHz	313 kHz	1.25 MHz
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

**FIGURE 14-1: OUTPUT COMPARE MODULE BLOCK DIAGRAM**



**Note:** Only OC1 and OC2 can trigger a DMA data transfer.

# PIC24H

## 14.4 Output Compare Register

REGISTER 14-1: OCxCON: OUTPUT COMPARE x CONTROL REGISTER

U-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
—	—	OCSIDL	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	R-0 HC	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	OCFLT	OCTSEL <sup>(1)</sup>	OCM<2:0>		
bit 7							bit 0

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

- bit 15-14     **Unimplemented:** Read as '0'
- bit 13       **OCSIDL:** Stop Output Compare 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 Timer Select bit<sup>(1)</sup>
  - 1 = Timer3 is the clock source for Compare x
  - 0 = Timer2 is the clock source for Compare x
- bit 2-0     **OCM<2:0>:** Output Compare 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

**Note 1:** Refer to the device data sheet for specific time bases available to the output compare module.



## 15.0 SERIAL PERIPHERAL INTERFACE (SPI)

**Note:** This data sheet summarizes the features of this group of PIC24H devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the “*dsPIC30F Family Reference Manual*” (DS70046).

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 SPI and SIOP from Motorola®.

**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.

### 15.1 Operating Function Description

Each SPI module consists of a 16-bit shift register, SPIxSR (where x = 1 or 2), used for shifting data in and out, and a buffer register, SPIxBUF. A control register, SPIxCON, configures the module. Additionally, a status register, SPIxSTAT, indicates various status conditions.

The serial interface consists of 4 pins: SDIx (serial data input), SDOx (serial data output), SCKx (shift clock input or output), and SSx (active low slave select).

In Master mode operation, SCK is a clock output but in Slave mode, it is a clock input.

A series of eight (8) or sixteen (16) clock pulses shift out bits from the SPIxSR to SDOx pin and simultaneously shift in data from SDIx pin. An interrupt is generated when the transfer is complete and the corresponding interrupt flag bit (SPI1IF or SPI2IF) is set. This interrupt can be disabled through an interrupt enable bit (SPI1IE or SPI2IE).

The receive operation is double-buffered. When a complete byte is received, it is transferred from SPIxSR to SPIxBUF.

If the receive buffer is full when new data is being transferred from SPIxSR to SPIxBUF, the module will set the SPIROV bit indicating an overflow condition. The transfer of the data from SPIxSR to SPIxBUF will not be completed and the new data will be lost. The module will not respond to SCL transitions while SPIROV is '1', effectively disabling the module until SPIxBUF is read by user software.

Transmit writes are also double-buffered. The user writes to SPIxBUF. When the master or slave transfer is completed, the contents of the shift register (SPIxSR) are moved to the receive buffer. If any transmit data has

been written to the buffer register, the contents of the transmit buffer are moved to SPIxSR. The received data is thus placed in SPIxBUF and the transmit data in SPIxSR is ready for the next transfer.

**Note:** Both the transmit buffer (SPIxTXB) and the receive buffer (SPIxRXB) are mapped to the same register address, SPIxBUF.

**Note:** Do not perform read-modify-write operations (such as bit-oriented instructions) on the SPIxBUF register.

The module supports a basic framed SPI protocol while operating in either Master or Slave mode. 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
- 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, SSx is not used. In the 2-pin mode, both SDOx and SSx are not used.

A block diagram of an SPI module is shown in Figure 15-1. All PIC24H devices contain two SPI modules on a single device.

The SPI module contains an 8-word deep FIFO buffer; the top of the buffer is denoted as SPIxBUF. If DMA transfers are enabled, the FIFO buffer must be disabled by clearing the ENHBUF bit (SPIxCON2<0>).

To set up the SPI module for the 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.

# PIC24H

To set up the SPI module for the Slave mode of operation:

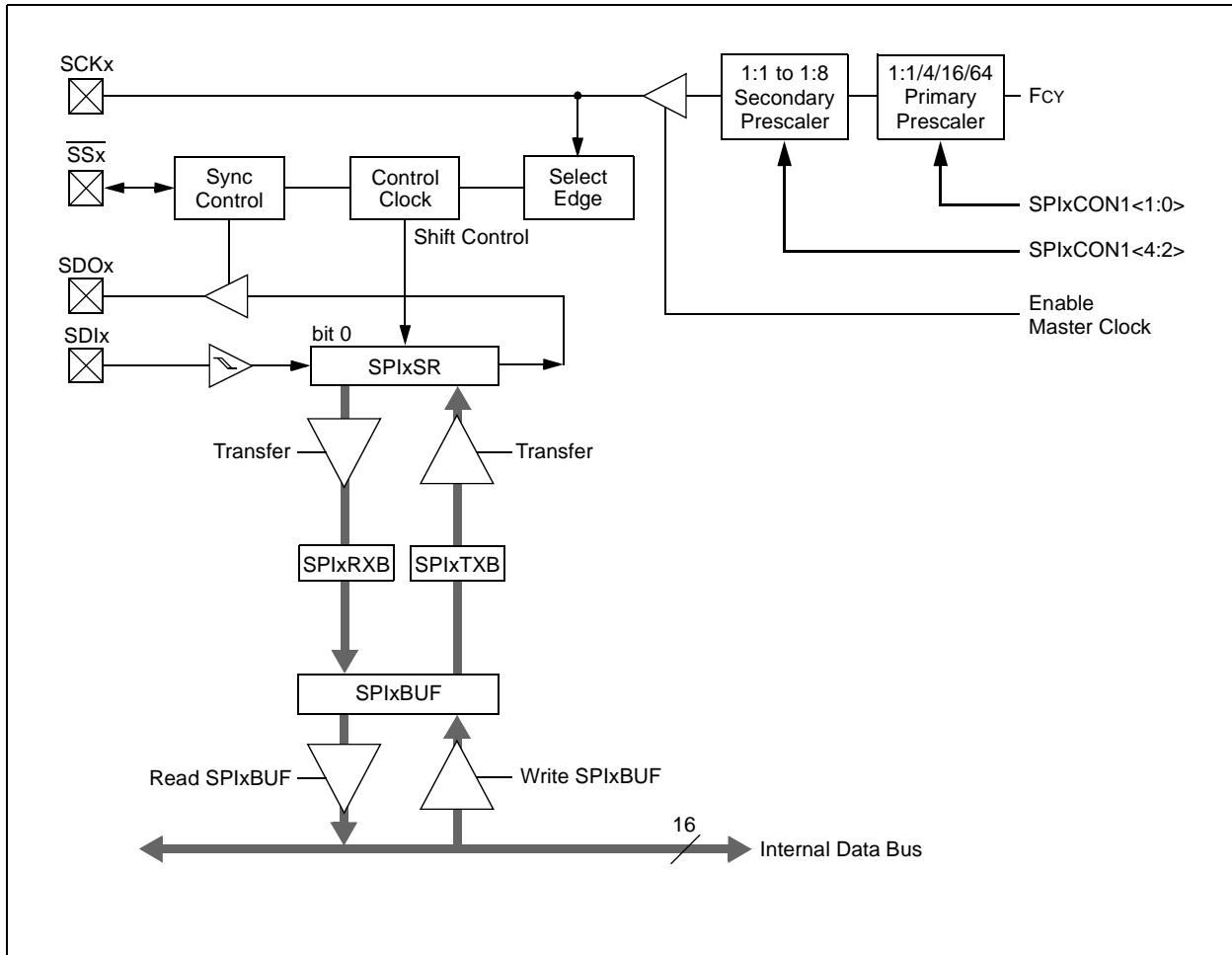
1. Clear the SPIxBUF register.
2. If using interrupts:
  - a) Clear the SPIxIF bit in the respective IFSn register.
  - b) Set the SPIxIE bit in the respective IECn register.
  - c) Write the SPIxIP bits in the respective IPCn register to set the interrupt priority.
3. Write the desired settings to the 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>).

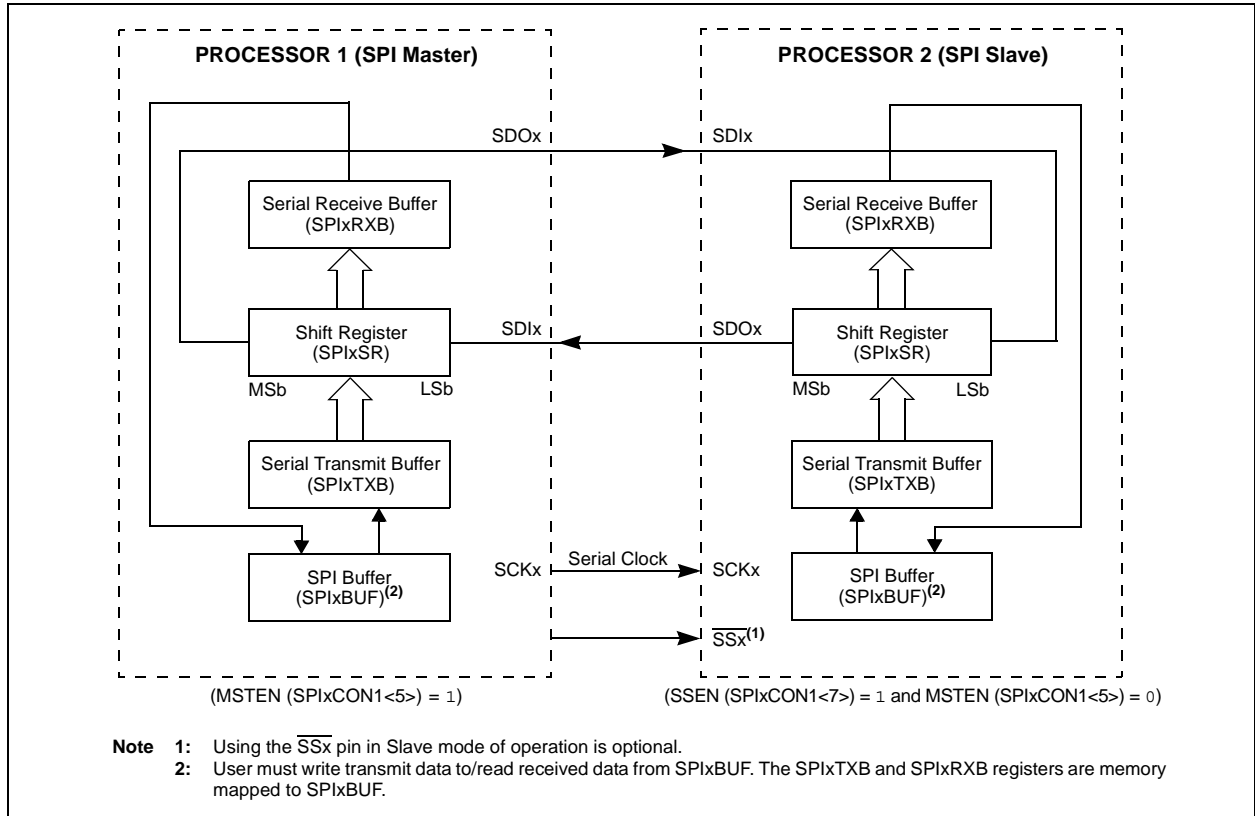
The SPI module generates an interrupt indicating completion of a byte or word transfer, as well as a separate interrupt for all SPI error conditions.

**Note:** Both SPI1 and SPI2 can trigger a DMA data transfer. If SPI1 or SPI2 is selected as the DMA IRQ source, a DMA transfer occurs when the SPI1IF or SPI2IF bit gets set as a result of an SPI1 or SPI2 byte or word transfer.

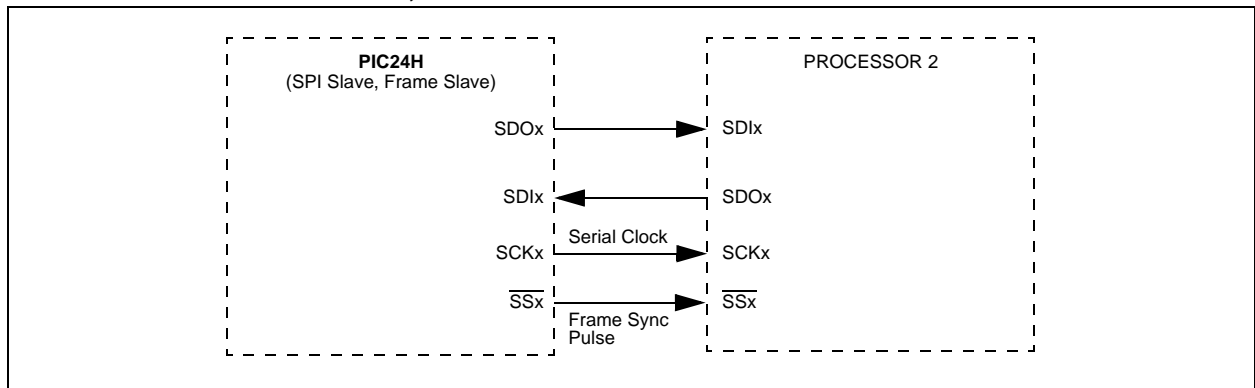
**FIGURE 15-1: SPI MODULE BLOCK DIAGRAM**



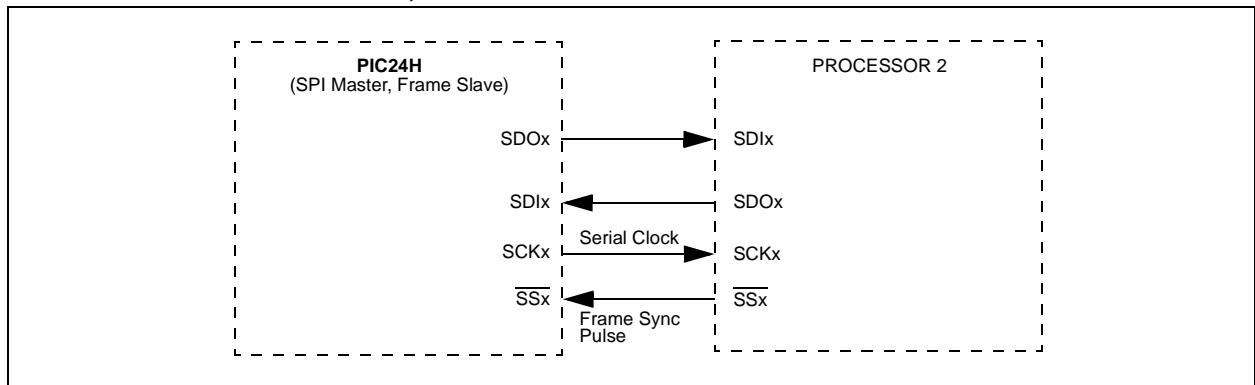
**FIGURE 15-2: SPI MASTER/SLAVE CONNECTION**



**FIGURE 15-3: SPI MASTER, FRAME MASTER CONNECTION DIAGRAM**

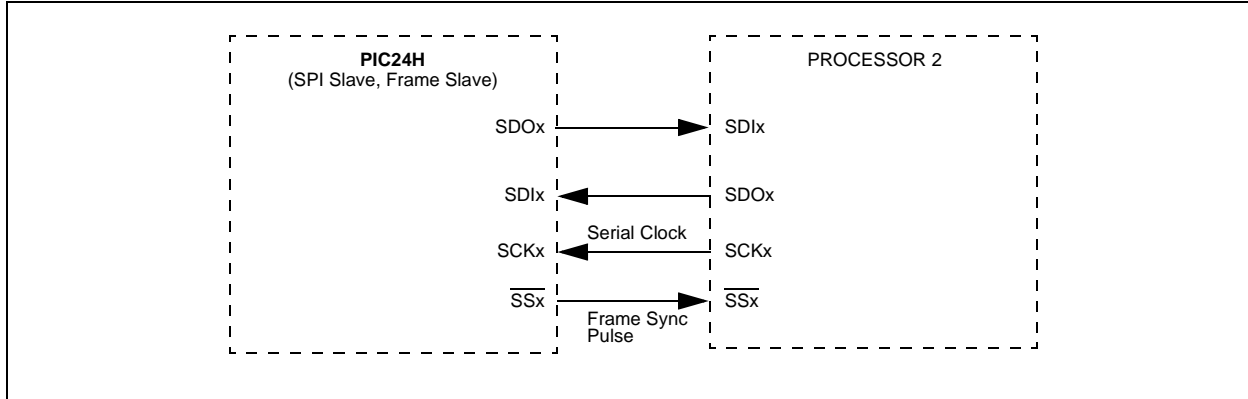


**FIGURE 15-4: SPI MASTER, FRAME SLAVE CONNECTION DIAGRAM**

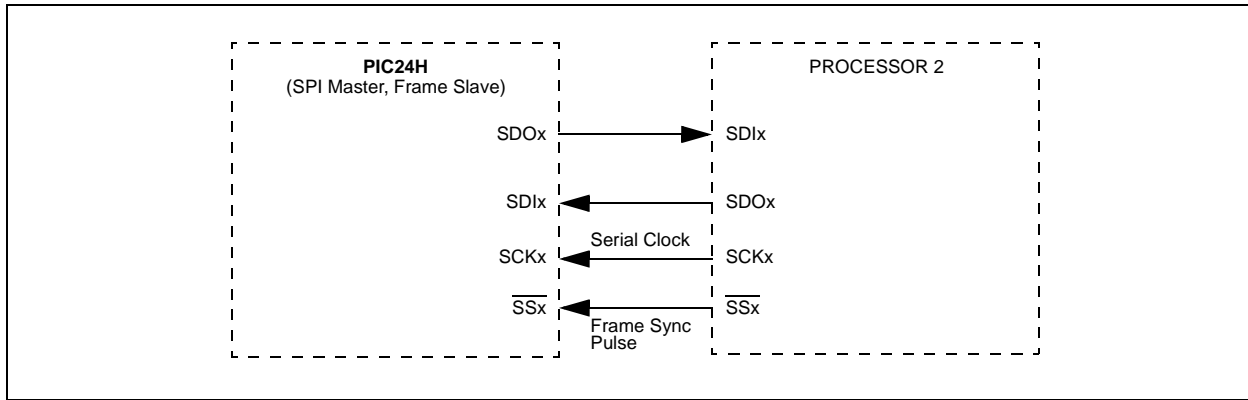


# PIC24H

**FIGURE 15-5: SPI SLAVE, FRAME MASTER CONNECTION DIAGRAM**



**FIGURE 15-6: SPI SLAVE, FRAME SLAVE CONNECTION DIAGRAM**



**EQUATION 15-1: RELATIONSHIP BETWEEN DEVICE AND SPI CLOCK SPEED**

$$F_{SCK} = \frac{F_{CY}}{\text{Primary Prescaler} * \text{Secondary Prescaler}}$$

**TABLE 15-1: SAMPLE SCKx FREQUENCIES**

F <sub>CY</sub> = 40 MHz		Secondary Prescaler Settings				
		1:1	2:1	4:1	6:1	8:1
Primary Prescaler Settings	1:1	Invalid	Invalid	10000	6666.67	5000
	4:1	10000	5000	2500	1666.67	1250
	16:1	2500	1250	625	416.67	312.50
	64:1	625	312.5	156.25	104.17	78.125
F <sub>CY</sub> = 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:** SCKx frequencies shown in kHz.

## REGISTER 15-1: SPIxSTAT: SPIx STATUS AND CONTROL REGISTER

R/W-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
SPIEN	—	SPIIDL	—	—	—	—	—
bit 15							bit 8

U-0	R/C-0	U-0	U-0	U-0	U-0	R-0	R-0
—	SPIROV	—	—	—	—	SPITBF	SPIRBF
bit 7							bit 0

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

- bit 15      **SPIEN:** SPIx Enable bit  
1 = Enables module and configures SCKx, SDOx, SDIx and  $\overline{SSx}$  as serial port pins  
0 = Disables module
- bit 14      **Unimplemented:** Read as '0'
- bit 13      **SPIIDL:** 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        **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  
Automatically set in hardware when CPU writes SPIxBUF location, loading SPIxTXB.  
Automatically cleared in hardware when SPIx module transfers data from SPIxTXB to SPIxSR.
- bit 0        **SPIRBF:** SPIx Receive Buffer Full Status bit  
1 = Receive complete, SPIxRXB is full  
0 = Receive is not complete, SPIxRXB is empty  
Automatically set in hardware when SPIx transfers data from SPIxSR to SPIxRXB.  
Automatically cleared in hardware when core reads SPIxBUF location, reading SPIxRXB.

# PIC24H

## REGISTER 15-2: SPIxCON1: SPIx CONTROL REGISTER 1

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 <sup>(1)</sup>	
bit 15								bit 8
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	SPRE<2:0>			PPRE<1:0>		
bit 7								bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 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<sup>(1)</sup>  
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)
- bit 7         **SSEN:** Slave Select Enable bit (Slave mode)  
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       **SPRE<2:0>:** Secondary Prescale bits (Master mode)  
111 = Secondary prescale 1:1  
110 = Secondary prescale 2:1  
...  
000 = Secondary prescale 8:1
- bit 1-0       **PPRE<1:0>:** Primary Prescale bits (Master mode)  
11 = Primary prescale 1:1  
10 = Primary prescale 4:1  
01 = Primary prescale 16:1  
00 = Primary prescale 64:1

**Note 1:** 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).

## REGISTER 15-3: SPIxCON2: SPIx CONTROL REGISTER 2

R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0	U-0
FRMEN	SPIFSD	FRMPOL	—	—	—	—	—
bit 15						bit 8	

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	U-0
—	—	—	—	—	—	FRMDLY	—
bit 7						bit 0	

### Legend:

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

- bit 15                      **FRMEN:** Framed SPIx Support bit  
                                  1 = Framed SPIx support enabled ( $\overline{SSx}$  pin used as frame sync pulse input/output)  
                                  0 = Framed SPIx support disabled
- bit 14                      **SPIFSD:** Frame Sync Pulse Direction Control bit  
                                  1 = Frame sync pulse input (slave)  
                                  0 = Frame sync pulse output (master)
- bit 13                      **FRMPOL:** Frame Sync Pulse Polarity bit  
                                  1 = Frame sync pulse is active-high  
                                  0 = Frame sync pulse is active-low
- bit 12-2                      **Unimplemented:** Read as '0'
- bit 1                        **FRMDLY:** 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                        **Unimplemented:** Read as '0'  
                                  This bit must not be set to '1' by the user application.

# PIC24H

---

NOTES:



## 16.0 INTER-INTEGRATED CIRCUIT (I<sup>2</sup>C)

**Note:** This data sheet summarizes the features of this group of PIC24H devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the “*dsPIC30F Family Reference Manual*” (DS70046).

The Inter-Integrated Circuit (I<sup>2</sup>C) module provides complete hardware support for both Slave and Multi-Master modes of the I<sup>2</sup>C serial communication standard, with a 16-bit interface.

The PIC24H devices have up to two I<sup>2</sup>C interface modules, denoted as I2C1 and I2C2. Each I<sup>2</sup>C module has a 2-pin interface: the SCLx pin is clock and the SDAX pin is data.

Each I<sup>2</sup>C module ‘x’ (x = 1 or 2) offers the following key features:

- I<sup>2</sup>C interface supporting both master and slave operation.
- I<sup>2</sup>C Slave mode supports 7 and 10-bit address.
- I<sup>2</sup>C Master mode supports 7 and 10-bit address.
- I<sup>2</sup>C port allows bidirectional transfers between master and slaves.
- Serial clock synchronization for I<sup>2</sup>C port can be used as a handshake mechanism to suspend and resume serial transfer (SCLREL control).
- I<sup>2</sup>C supports multi-master operation; detects bus collision and will arbitrate accordingly.

### 16.1 Operating Modes

The hardware fully implements all the master and slave functions of the I<sup>2</sup>C Standard and Fast mode specifications, as well as 7 and 10-bit addressing.

The I<sup>2</sup>C module can operate either as a slave or a master on an I<sup>2</sup>C bus.

The following types of I<sup>2</sup>C operation are supported:

- I<sup>2</sup>C slave operation with 7-bit address
- I<sup>2</sup>C slave operation with 10-bit address
- I<sup>2</sup>C master operation with 7 or 10-bit address

For details about the communication sequence in each of these modes, please refer to the “*dsPIC30F Family Reference Manual*” (DS70046).

### 16.2 I<sup>2</sup>C Registers

I2CxCON and I2CxSTAT are control and status registers, respectively. The I2CxCON register is readable and writable. The lower six bits of I2CxSTAT are read-only. The remaining bits of the I2CxSTAT are read/write.

I2CxRSR is the shift register used for shifting data, whereas I2CxRCV is the buffer register to which data bytes are written, or from which data bytes are read. I2CxRTRN is the transmit register to which bytes are written during a transmit operation.

The I2CxADD register holds the slave address. A status bit, ADD10, indicates 10-bit Address mode. The I2CxBRG acts as the Baud Rate Generator (BRG) reload value.

In receive operations, I2CxRSR and I2CxRCV together form a double-buffered receiver. When I2CxRSR receives a complete byte, it is transferred to I2CxRCV and an interrupt pulse is generated.

### 16.3 I<sup>2</sup>C Interrupts

The I<sup>2</sup>C module generates two interrupt flags, MI2CxIF (I<sup>2</sup>C Master Events Interrupt Flag) and SI2CxIF (I<sup>2</sup>C Slave Events Interrupt Flag). A separate interrupt is generated for all I<sup>2</sup>C error conditions.

### 16.4 Baud Rate Generator

In I<sup>2</sup>C Master mode, the reload value for the BRG is located in the I2CxBRG register. When the BRG is loaded with this value, the BRG counts down to ‘0’ and stops until another reload has taken place. If clock arbitration is taking place, for instance, the BRG is reloaded when the SCLx pin is sampled high.

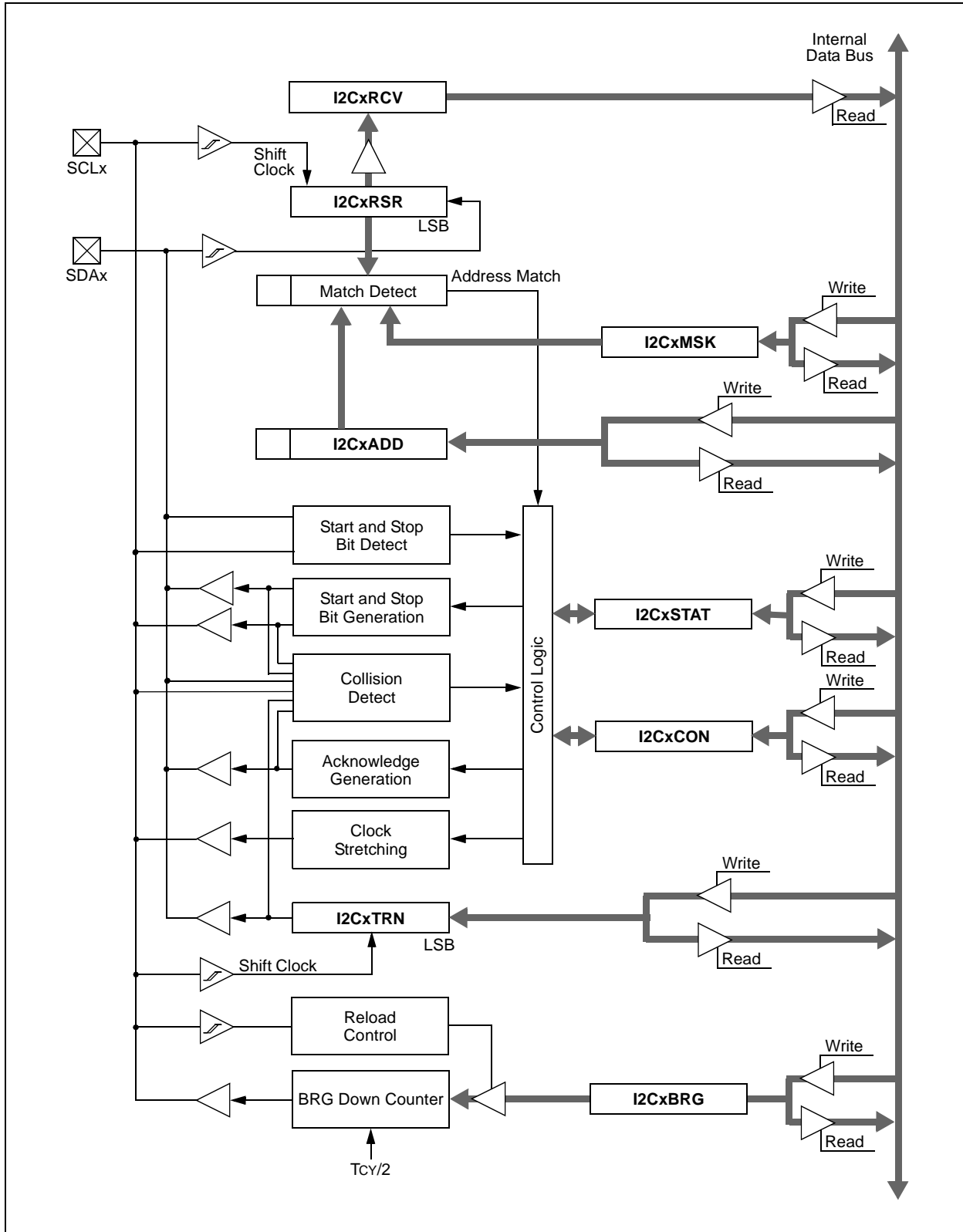
As per the I<sup>2</sup>C standard, FSCL may be 100 kHz or 400 kHz. However, the user can specify any baud rate up to 1 MHz. I2CxBRG values of ‘0’ or ‘1’ are illegal.

#### EQUATION 16-1: SERIAL CLOCK RATE

$$I2CxBRG = \left( \frac{FCY}{FSCL} - \frac{FCY}{1,111,111} \right) - 1$$

# PIC24H

FIGURE 16-1: I<sup>2</sup>C™ BLOCK DIAGRAM (x = 1 OR 2)



## 16.5 I<sup>2</sup>C Module Addresses

The I2CxADD register contains the Slave mode addresses. The register is a 10-bit register.

If the A10M bit (I2CxCON<10>) is '0', the address is interpreted by the module as a 7-bit address. When an address is received, it is compared to the 7 Least Significant bits of the I2CxADD register.

If the A10M bit is '1', the address is assumed to be a 10-bit address. When an address is received, it will be compared with the binary value, '11110 A9 A8' (where A9 and A8 are two Most Significant bits of I2CxADD). If that value matches, the next address will be compared with the Least Significant 8 bits of I2CxADD, as specified in the 10-bit addressing protocol.

**TABLE 16-1: 7-BIT I<sup>2</sup>C™ SLAVE ADDRESSES SUPPORTED BY PIC24H**

0x00	General call address or Start byte
0x01-0x03	Reserved
0x04-0x07	Hs mode Master codes
0x08-0x77	Valid 7-bit addresses
0x78-0x7b	Valid 10-bit addresses (lower 7 bits)
0x7c-0x7f	Reserved

## 16.6 Slave Address Masking

The I2CxMSK register (Register 16-3) 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, '00000000' and '00100000'.

To enable address masking, the Intelligent Peripheral Management Interface (IPMI) must be disabled by clearing the IPMIEN bit (I2CxCON<11>).

## 16.7 IPMI Support

The control bit, IPMIEN, enables the module to support the Intelligent Peripheral Management Interface (IPMI). When this bit is set, the module accepts and acts upon all addresses.

## 16.8 General Call Address Support

The general call address can address all devices. When this address is used, all devices should, in theory, respond with an Acknowledgement.

The general call address is one of eight addresses reserved for specific purposes by the I<sup>2</sup>C protocol. It consists of all '0's with R\_W = 0.

The general call address is recognized when the General Call Enable (GCEN) bit is set (I2CxCON<7> = 1). When the interrupt is serviced, the source for the interrupt can be checked by reading the contents of the I2CxRCV to determine if the address was device-specific or a general call address.

## 16.9 Automatic Clock Stretch

In Slave modes, the module can synchronize buffer reads and writes to the master device by clock stretching.

### 16.9.1 TRANSMIT CLOCK STRETCHING

Both 10-bit and 7-bit Transmit modes implement clock stretching by asserting the SCLREL bit after the falling edge of the ninth clock, if the TBF bit is cleared, indicating the buffer is empty.

In Slave Transmit modes, clock stretching is always performed, irrespective of the STREN bit. The user's ISR must set the SCLREL bit before transmission is allowed to continue. By holding the SCLx line low, the user has time to service the ISR and load the contents of the I2CxTRN before the master device can initiate another transmit sequence.

### 16.9.2 RECEIVE CLOCK STRETCHING

The STREN bit in the I2CxCON register can be used to enable clock stretching in Slave Receive mode. When the STREN bit is set, the SCLx pin will be held low at the end of each data receive sequence.

The user's ISR must set the SCLREL bit before reception is allowed to continue. By holding the SCLx line low, the user has time to service the ISR and read the contents of the I2CxRCV before the master device can initiate another receive sequence. This will prevent buffer overruns from occurring.

## 16.10 Software Controlled Clock Stretching (STREN = 1)

When the STREN bit is '1', the SCLREL bit may be cleared by software to allow software to control the clock stretching.

If the STREN bit is '0', a software write to the SCLREL bit will be disregarded and have no effect on the SCLREL bit.

## 16.11 Slope Control

The I<sup>2</sup>C standard requires slope control on the SDAx and SCLx signals for Fast mode (400 kHz). The control bit, DISSLW, enables the user to disable slew rate control if desired. It is necessary to disable the slew rate control for 1 MHz mode.

## 16.12 Clock Arbitration

Clock arbitration occurs when the master deasserts the SCLx pin (SCLx allowed to float high) during any receive, transmit or Restart/Stop condition. When the SCLx pin is allowed to float high, the Baud Rate Generator (BRG) is suspended from counting until the SCLx pin is actually sampled high. When the SCLx pin is sampled high, the Baud Rate Generator is reloaded with the contents of I2CxBRG and begins counting. This ensures that the SCLx high time will always be at least one BRG rollover count in the event that the clock is held low by an external device.

## 16.13 Multi-Master Communication, Bus Collision and Bus Arbitration

Multi-Master mode support is achieved by bus arbitration. When the master outputs address/data bits onto the SDAx pin, arbitration takes place when the master outputs a '1' on SDAx by letting SDAx float high while another master asserts a '0'. When the SCLx pin floats high, data should be stable. If the expected data on SDAx is a '1' and the data sampled on the SDAx pin = 0, then a bus collision has taken place. The master will set the I<sup>2</sup>C master events interrupt flag and reset the master portion of the I<sup>2</sup>C port to its Idle state.

## REGISTER 16-1: I2CxCON: I2Cx CONTROL REGISTER

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

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

<b>Legend:</b>	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

- bit 15      **I2CEN:** I2Cx Enable bit  
 1 = Enables the I2Cx module and configures the SDAx and SCLx pins as serial port pins  
 0 = Disables the I2Cx module. All I<sup>2</sup>C 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<sup>2</sup>C 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 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<sup>2</sup>C slave)  
 1 = Enable interrupt when a general call address is received in the I2CxRSR (module is enabled for reception)  
 0 = General call address disabled
- bit 6        **STREN:** SCLx Clock Stretch Enable bit (when operating as I<sup>2</sup>C slave)  
 Used in conjunction with SCLREL bit.  
 1 = Enable software or receive clock stretching  
 0 = Disable software or receive clock stretching

# PIC24H

---

## REGISTER 16-1: I2CxCON: I2Cx CONTROL REGISTER (CONTINUED)

- bit 5      **ACKDT:** Acknowledge Data bit (when operating as I<sup>2</sup>C 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<sup>2</sup>C 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<sup>2</sup>C master)  
1 = Enables Receive mode for I<sup>2</sup>C. 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<sup>2</sup>C 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 Enable bit (when operating as I<sup>2</sup>C 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 Enable bit (when operating as I<sup>2</sup>C master)  
1 = Initiate Start condition on SDAx and SCLx pins. Hardware clear at end of master Start sequence.  
0 = Start condition not in progress

## REGISTER 16-2: I2CxSTAT: I2Cx STATUS REGISTER

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	

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	I2CPOV	D_A	P	S	R_W	RBF	TBF
bit 7						bit 0	

<b>Legend:</b>	U = Unimplemented bit, read as '0'		
R = Readable bit	W = Writable 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

- bit 15      **ACKSTAT:** Acknowledge Status bit  
(when operating as I<sup>2</sup>C 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<sup>2</sup>C 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<sup>2</sup>C module is busy  
0 = No collision  
Hardware set at occurrence of write to I2CxTRN while busy (cleared by software).
- bit 6        **I2CPOV:** 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 I2CxRSR to I2CxRCV (cleared by software).
- bit 5        **D\_A:** Data/Address bit (when operating as I<sup>2</sup>C 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 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.

# PIC24H

---

## REGISTER 16-2: I2CxSTAT: I2Cx STATUS REGISTER (CONTINUED)

- 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\_W:** Read/Write Information bit (when operating as I<sup>2</sup>C 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<sup>2</sup>C 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 is 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.



## REGISTER 16-3: I2CxMSK: I2Cx SLAVE MODE ADDRESS MASK REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	—	—	—	AMSK9	AMSK8
bit 15						bit 8	

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	

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 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

# PIC24H

---

NOTES:

## 17.0 UNIVERSAL ASYNCHRONOUS RECEIVER TRANSMITTER (UART)

**Note:** This data sheet summarizes the features of this group of PIC24H devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the “dsPIC30F Family Reference Manual” (DS70046).

The Universal Asynchronous Receiver Transmitter (UART) module is one of the serial I/O modules available in the PIC24H 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 UxCTS and UxRTS pins and also includes an IrDA® encoder and decoder.

The primary features of the UART module are:

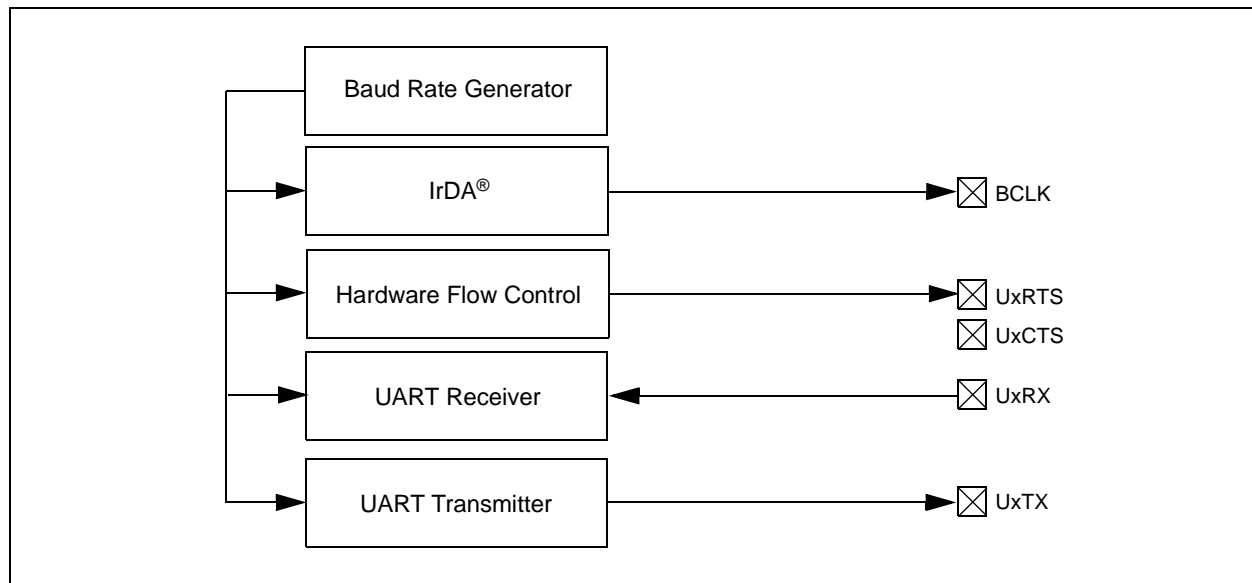
- Full-Duplex, 8 or 9-bit Data Transmission through the UxTX and UxRX pins
- Even, Odd or No Parity Options (for 8-bit data)
- One or Two Stop bits
- Hardware Flow Control Option with UxCTS and UxRTS pins

- 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
- A Separate Interrupt for all UART Error Conditions
- 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 17-1. The UART module consists of the key important hardware elements:

- Baud Rate Generator
- Asynchronous Transmitter
- Asynchronous Receiver

**FIGURE 17-1: UART SIMPLIFIED BLOCK DIAGRAM**



**Note 1:** Both UART1 and UART2 can trigger a DMA data transfer. If U1TX, U1RX, U2TX or U2RX is selected as a DMA IRQ source, a DMA transfer occurs when the U1TXIF, U1RXIF, U2TXIF or U2RXIF bit gets set as a result of a UART1 or UART2 transmission or reception.

**2:** If DMA transfers are required, the UART TX/RX FIFO buffer must be set to a size of 1 byte/word (i.e., UTXISEL<1:0> = 00 and URXISEL<1:0> = 00).

# PIC24H

## 17.1 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 17-1 shows the formula for computation of the baud rate with BRGH = 0.

### EQUATION 17-1: UART BAUD RATE WITH BRGH = 0

$$\text{Baud Rate} = \frac{\text{FCY}}{16 \cdot (\text{BRGx} + 1)}$$

$$\text{BRGx} = \frac{\text{FCY}}{16 \cdot \text{Baud Rate}} - 1$$

**Note:** FCY denotes the instruction cycle clock frequency (FOSC/2).

Example 17-1 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).

### EXAMPLE 17-1: BAUD RATE ERROR CALCULATION (BRGH = 0)

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%

Equation 17-2 shows the formula for computation of the baud rate with BRGH = 1.

### EQUATION 17-2: UART BAUD RATE WITH BRGH = 1

$$\text{Baud Rate} = \frac{\text{FCY}}{4 \cdot (\text{BRGx} + 1)}$$

$$\text{BRGx} = \frac{\text{FCY}}{4 \cdot \text{Baud Rate}} - 1$$

**Note:** FCY denotes the instruction cycle clock frequency (FOSC/2).

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.

## 17.2 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 UxTXREG word. The value will be immediately transferred to the Transmit Shift Register (TSR) and the serial bit stream will start shifting out with the 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 bits, UTXISEL<1:0>.

## 17.3 Transmitting in 9-bit Data Mode

1. Set up the UART (as described in **Section 17.2 “Transmitting in 8-bit Data Mode”**).
2. Enable the UART.
3. Set the UTXEN bit (causes a transmit interrupt).
4. Write UxTXREG as a 16-bit value only.
5. A word write to UxTXREG 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 bits, UTXISEL<1:0>.

## 17.4 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 UxTXREG register with a dummy character to initiate transmission (value is ignored).
4. Write 0x55 to UxTXREG – 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.

## 17.5 Receiving in 8-bit or 9-bit Data Mode

1. Set up the UART (as described in **Section 17.2 “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 bits, URXISEL<1:0>.
4. Read the OERR bit to determine if an overrun error has occurred. The OERR bit must be reset in software.
5. Read UxRXREG.

The act of reading the UxRXREG character will move the next character to the top of the receive FIFO, including a new set of PERR and FERR values.

## 17.6 Flow Control Using UxCTS and UxRTS Pins

UARTx Clear to Send (UxCTS) and Request to Send (UxRTS) are the two hardware controlled active-low pins that are associated with the UART module. These two pins allow the UART to operate in Simplex and Flow Control modes. They are implemented to control the transmission and the reception between the Data Terminal Equipment (DTE). The UEN<1:0> bits in the UxMODE register configures these pins.

## 17.7 Infrared Support

The UART module provides two types of infrared UART support:

- IrDA clock output to support external IrDA encoder and decoder device (legacy module support)
- Full implementation of the IrDA encoder and decoder.

### 17.7.1 EXTERNAL IrDA SUPPORT – IrDA CLOCK OUTPUT

To support external IrDA encoder and decoder devices, the BCLK pin (same as the UxRTS pin) can be configured to generate the 16x baud clock. With UEN<1:0> = 11, the BCLK pin will output the 16x baud clock if the UART module is enabled; it can be used to support the IrDA codec chip.

### 17.7.2 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.

# PIC24H

## REGISTER 17-1: UxMODE: UARTx MODE REGISTER

R/W-0	U-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0 <sup>(2)</sup>	R/W-0 <sup>(2)</sup>
UARTEN	—	USIDL	IREN <sup>(1)</sup>	RTSMD	—	UEN<1:0>	
bit 15							bit 8

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	URXINV	BRGH	PDSEL<1:0>		STSEL
bit 7							bit 0

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

- 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<sup>(1)</sup>  
 1 = IrDA encoder and decoder enabled  
 0 = IrDA encoder and decoder disabled
- bit 11      **RTSMD:** Mode Selection for UxRTS Pin bit  
 1 = UxRTS pin in Simplex mode  
 0 = UxRTS pin in Flow Control mode
- bit 10      **Unimplemented:** Read as '0'
- bit 9-8     **UEN<1:0>:** UARTx Enable bits  
 11 = UxTX, UxRX and BCLK pins are enabled and used; UxCTS pin controlled by port latches  
 10 = UxTX, UxRX, UxCTS and UxRTS pins are enabled and used  
 01 = UxTX, UxRX and UxRTS pins are enabled and used; UxCTS pin controlled by port latches  
 00 = UxTX and UxRX pins are enabled and used; UxCTS and UxRTS/BCLK pins controlled by port latches
- 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      **URXINV:** Receive Polarity Inversion bit  
 1 = UxRX Idle state is '0'  
 0 = UxRX Idle state is '1'

**Note 1:** This feature is only available for the 16x BRG mode (BRGH = 0).  
**2:** Bit availability depends on pin availability.

## REGISTER 17-1: UxMODE: UARTx MODE REGISTER (CONTINUED)

- 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     **PDSEL<1:0>:** 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

**Note 1:** This feature is only available for the 16x BRG mode (BRGH = 0).

**2:** Bit availability depends on pin availability.

# PIC24H

## REGISTER 17-2: UxSTA: UARTx STATUS AND CONTROL REGISTER

R/W-0	R/W-0	R/W-0	U-0	R/W-0 HC	R/W-0	R-0	R-1
UTXISEL1	UTXINV <sup>(1)</sup>	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R-1	R-0	R-0	R/C-0	R-0
URXISEL<1:0>		ADDEN	RIDLE	PERR	FERR	OERR	URXDA
bit 7							bit 0

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

- bit 15,13 **UTXISEL<1:0>**: 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<sup>(1)</sup>  
 1 = IrDA encoded, UxTX Idle state is '1'  
 0 = IrDA encoded, UxTX Idle state is '0'
- 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 **URXISEL<1:0>**: Receive Interrupt Mode Selection bits  
 11 = Interrupt is set on UxRSR transfer making the receive buffer full (i.e., has 4 data characters)  
 10 = Interrupt is set on UxRSR 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 UxRSR 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

**Note 1:** Value of bit only affects the transmit properties of the module when the IrDA encoder is enabled (IREN = 1).



---

---

## REGISTER 17-2: UxSTA: UARTx STATUS AND CONTROL REGISTER (CONTINUED)

- 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 UxRSR 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

**Note 1:** Value of bit only affects the transmit properties of the module when the IrDA encoder is enabled (IREN = 1).

# PIC24H

---

NOTES:

## 18.0 ENHANCED CAN MODULE

**Note:** This data sheet summarizes the features of this group of PIC24H devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the “*dsPIC30F Family Reference Manual*” (DS70046).

### 18.1 Overview

The Enhanced Controller Area Network (ECAN) module is a serial interface, useful for communicating with other CAN modules or microcontroller devices. This interface/protocol was designed to allow communications within noisy environments. The PIC24H devices contain up to two ECAN modules.

The CAN module is a communication controller implementing the CAN 2.0 A/B protocol, as defined in the BOSCH specification. The module will support CAN 1.2, CAN 2.0A, CAN 2.0B Passive and CAN 2.0B Active versions of the protocol. The module implementation is a full CAN system. The CAN specification is not covered within this data sheet. The reader may refer to the BOSCH CAN specification for further details.

The module features are as follows:

- Implementation of the CAN protocol, CAN 1.2, CAN 2.0A and CAN 2.0B
- Standard and extended data frames
- 0-8 bytes data length
- Programmable bit rate up to 1 Mbit/sec
- Automatic response to remote transmission requests
- Up to 8 transmit buffers with application specified prioritization and abort capability (each buffer may contain up to 8 bytes of data)
- Up to 32 receive buffers (each buffer may contain up to 8 bytes of data)
- Up to 16 full (standard/extended identifier) acceptance filters
- 3 full acceptance filter masks
- DeviceNet™ addressing support
- Programmable wake-up functionality with integrated low-pass filter
- Programmable Loopback mode supports self-test operation
- Signaling via interrupt capabilities for all CAN receiver and transmitter error states
- Programmable clock source
- Programmable link to input capture module (IC2 for both CAN1 and CAN2) for time-stamping and network synchronization
- Low-power Sleep and Idle mode

The CAN bus module consists of a protocol engine and message buffering/control. The CAN protocol engine handles all functions for receiving and transmitting messages on the CAN bus. Messages are transmitted by first loading the appropriate data registers. Status and errors can be checked by reading the appropriate registers. Any message detected on the CAN bus is checked for errors and then matched against filters to see if it should be received and stored in one of the receive registers.

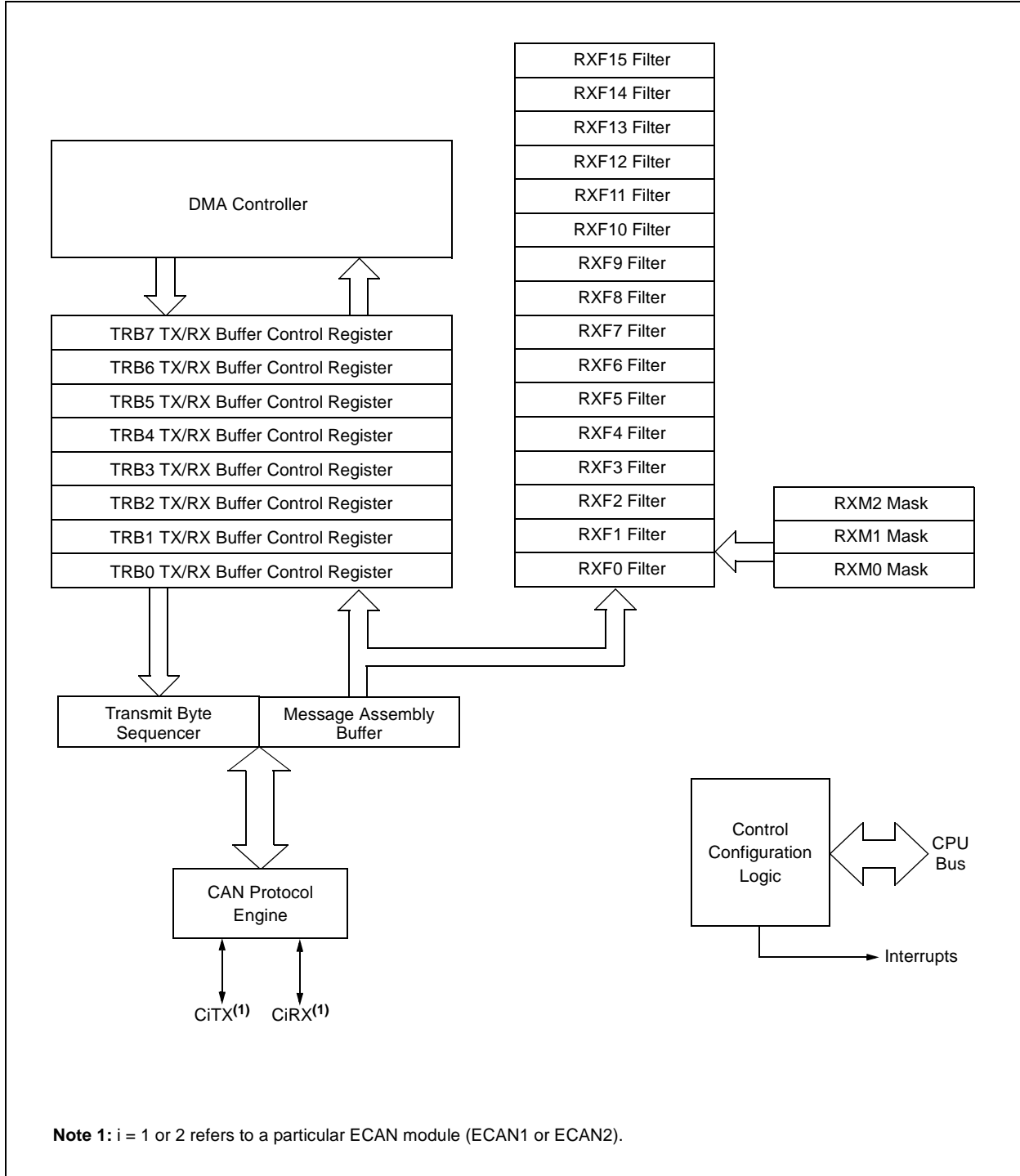
### 18.2 Frame Types

The CAN module transmits various types of frames which include data messages, remote transmission requests and as other frames that are automatically generated for control purposes. The following frame types are supported:

- **Standard Data Frame:**  
A standard data frame is generated by a node when the node wishes to transmit data. It includes an 11-bit standard identifier (SID) but not an 18-bit extended identifier (EID).
- **Extended Data Frame:**  
An extended data frame is similar to a standard data frame but includes an extended identifier as well.
- **Remote Frame:**  
It is possible for a destination node to request the data from the source. For this purpose, the destination node sends a remote frame with an identifier that matches the identifier of the required data frame. The appropriate data source node will then send a data frame as a response to this remote request.
- **Error Frame:**  
An error frame is generated by any node that detects a bus error. An error frame consists of two fields: an error flag field and an error delimiter field.
- **Overload Frame:**  
An overload frame can be generated by a node as a result of two conditions. First, the node detects a dominant bit during interframe space which is an illegal condition. Second, due to internal conditions, the node is not yet able to start reception of the next message. A node may generate a maximum of 2 sequential overload frames to delay the start of the next message.
- **Interframe Space:**  
Interframe space separates a proceeding frame (of whatever type) from a following data or remote frame.

# PIC24H

**FIGURE 18-1: ECAN™ MODULE BLOCK DIAGRAM**



**Note 1:** i = 1 or 2 refers to a particular ECAN module (ECAN1 or ECAN2).

## 18.3 Modes of Operation

The CAN module can operate in one of several operation modes selected by the user. These modes include:

- Initialization Mode
- Disable Mode
- Normal Operation Mode
- Listen Only Mode
- Listen All Messages Mode
- Loopback Mode

Modes are requested by setting the REQOP<2:0> bits (CiCTRL1<10:8>). Entry into a mode is Acknowledged by monitoring the OPMODE<2:0> bits (CiCTRL1<7:5>). The module will not change the mode and the OPMODE bits until a change in mode is acceptable, generally during bus Idle time, which is defined as at least 11 consecutive recessive bits.

### 18.3.1 INITIALIZATION MODE

In the Initialization mode, the module will not transmit or receive. The error counters are cleared and the interrupt flags remain unchanged. The programmer will have access to Configuration registers that are access restricted in other modes. The module will protect the user from accidentally violating the CAN protocol through programming errors. All registers which control the configuration of the module can not be modified while the module is on-line. The CAN module will not be allowed to enter the Configuration mode while a transmission is taking place. The Configuration mode serves as a lock to protect the following registers.

- All Module Control Registers
- Baud Rate and Interrupt Configuration Registers
- Bus Timing Registers
- Identifier Acceptance Filter Registers
- Identifier Acceptance Mask Registers

### 18.3.2 DISABLE MODE

In Disable mode, the module will not transmit or receive. The module has the ability to set the WAKIF bit due to bus activity, however, any pending interrupts will remain and the error counters will retain their value.

If the REQOP<2:0> bits (CiCTRL1<10:8>) = 001, the module will enter the Module Disable mode. If the module is active, the module will wait for 11 recessive bits on the CAN bus, detect that condition as an Idle bus, then accept the module disable command. When the OPMODE<2:0> bits (CiCTRL1<7:5>) = 001, that indicates whether the module successfully went into Module Disable mode. The I/O pins will revert to normal I/O function when the module is in the Module Disable mode.

The module can be programmed to apply a low-pass filter function to the CiRX input line while the module or the CPU is in Sleep mode. The WAKFIL bit (CiCFG2<14>) enables or disables the filter.

**Note:** Typically, if the CAN module is allowed to transmit in a particular mode of operation and a transmission is requested immediately after the CAN module has been placed in that mode of operation, the module waits for 11 consecutive recessive bits on the bus before starting transmission. If the user switches to Disable mode within this 11-bit period, then this transmission is aborted and the corresponding TXABT bit is set and TXREQ bit is cleared.

### 18.3.3 NORMAL OPERATION MODE

Normal Operation mode is selected when REQOP<2:0> = 000. In this mode, the module is activated and the I/O pins will assume the CAN bus functions. The module will transmit and receive CAN bus messages via the CiTX and CiRX pins.

### 18.3.4 LISTEN ONLY MODE

If the Listen Only mode is activated, the module on the CAN bus is passive. The transmitter buffers revert to the port I/O function. The receive pins remain inputs. For the receiver, no error flags or Acknowledge signals are sent. The error counters are deactivated in this state. The Listen Only mode can be used for detecting the baud rate on the CAN bus. To use this, it is necessary that there are at least two further nodes that communicate with each other.

### 18.3.5 LISTEN ALL MESSAGES MODE

The module can be set to ignore all errors and receive any message. The Listen All Messages mode is activated by setting REQOP<2:0> = '111'. In this mode, the data which is in the message assembly buffer, until the time an error occurred, is copied in the receive buffer and can be read via the CPU interface.

### 18.3.6 LOOPBACK MODE

If the Loopback mode is activated, the module will connect the internal transmit signal to the internal receive signal at the module boundary. The transmit and receive pins revert to their port I/O function.

## 18.4 Message Reception

### 18.4.1 RECEIVE BUFFERS

The CAN bus module has up to 32 receive buffers, located in DMA RAM. The first 8 buffers need to be configured as receive buffers by clearing the corresponding TX/RX buffer selection (TXENn) bit in a CiTRmnCON register. The overall size of the CAN buffer area in DMA RAM is selectable by the user and is defined by the DMABS<2:0> bits (CiFCTRL<15:13>). The first 16 buffers can be assigned to receive filters, while the rest can be used only as a FIFO buffer.

An additional buffer is always committed to monitoring the bus for incoming messages. This buffer is called the Message Assembly Buffer (MAB).

All messages are assembled by the MAB and are transferred to the buffers only if the acceptance filter criterion are met. When a message is received, the RBIF flag (CiINTF<1>) will be set. The user would then need to inspect the CiVEC and/or CiRXFUL1 register to determine which filter and buffer caused the interrupt to get generated. The RBIF bit can only be set by the module when a message is received. The bit is cleared by the user when it has completed processing the message in the buffer. If the RBIE bit is set, an interrupt will be generated when a message is received.

## 18.4.2 FIFO BUFFER MODE

The ECAN module provides FIFO buffer functionality if the buffer pointer for a filter has a value of '1111'. In this mode, the results of a hit on that buffer will write to the next available buffer location within the FIFO.

The CiFCTRL register defines the size of the FIFO. The FSA<4:0> bits in this register define the start of the FIFO buffers. The end of the FIFO is defined by the DMABS<2:0> bits if DMA is enabled. Thus, FIFO sizes up to 32 buffers are supported.

## 18.4.3 MESSAGE ACCEPTANCE FILTERS

The message acceptance filters and masks are used to determine if a message in the message assembly buffer should be loaded into either of the receive buffers. Once a valid message has been received into the Message Assembly Buffer (MAB), the identifier fields of the message are compared to the filter values. If there is a match, that message will be loaded into the appropriate receive buffer. Each filter is associated with a buffer pointer (FnBP<3:0>), which is used to link the filter to one of 16 receive buffers.

The acceptance filter looks at incoming messages for the IDE bit (CiTRBnSID<0>) to determine how to compare the identifiers. If the IDE bit is clear, the message is a standard frame and only filters with the EXIDE bit (CiRXFnSID<3>) clear are compared. If the IDE bit is set, the message is an extended frame, and only filters with the EXIDE bit set are compared.

## 18.4.4 MESSAGE ACCEPTANCE FILTER MASKS

The mask bits essentially determine which bits to apply the filter to. If any mask bit is set to a zero, then that bit will automatically be accepted regardless of the filter bit. There are three programmable acceptance filter masks associated with the receive buffers. Any of these three masks can be linked to each filter by selecting the desired mask in the FnMSK<1:0> bits in the appropriate CiFMSKSELn register.

## 18.4.5 RECEIVE ERRORS

The CAN module will detect the following receive errors:

- Cyclic Redundancy Check (CRC) Error
- Bit Stuffing Error
- Invalid Message Receive Error

These receive errors do not generate an interrupt. However, the receive error counter is incremented by one in case one of these errors occur. The RXWAR bit (CiINTF<9>) indicates that the receive error counter has reached the CPU warning limit of 96 and an interrupt is generated.

## 18.4.6 RECEIVE INTERRUPTS

Receive interrupts can be divided into 3 major groups, each including various conditions that generate interrupts:

- Receive Interrupt:
  - A message has been successfully received and loaded into one of the receive buffers. This interrupt is activated immediately after receiving the End-of-Frame (EOF) field. Reading the RXnIF flag will indicate which receive buffer caused the interrupt.
- Wake-up Interrupt:
  - The CAN module has woken up from Disable mode or the device has woken up from Sleep mode.
- Receive Error Interrupts:
  - A receive error interrupt will be indicated by the ERRIF bit. This bit shows that an error condition occurred. The source of the error can be determined by checking the bits in the CAN Interrupt Flag register, CiINTF.
    - Invalid Message Received:
      - If any type of error occurred during reception of the last message, an error will be indicated by the IVRIF bit.
    - Receiver Overrun:
      - The RBOVIF bit (CiINTF<2>) indicates that an overrun condition occurred.
    - Receiver Warning:
      - The RXWAR bit indicates that the receive error counter (RERRCNT<7:0>) has reached the warning limit of 96.
    - Receiver Error Passive:
      - The RXEP bit indicates that the receive error counter has exceeded the error passive limit of 127 and the module has gone into error passive state.

## 18.5 Message Transmission

### 18.5.1 TRANSMIT BUFFERS

The CAN module has up to eight transmit buffers, located in DMA RAM. These 8 buffers need to be configured as transmit buffers by setting the corresponding TX/RX buffer selection (TXENn or TXENm) bit in a CiTRmnCON register. The overall size of the CAN buffer area in DMA RAM is selectable by the user and is defined by the DMABS<2:0> bits (CiFCTRL<15:13>).

Each transmit buffer occupies 16 bytes of data. Eight of the bytes are the maximum 8 bytes of the transmitted message. Five bytes hold the standard and extended identifiers and other message arbitration information. The last byte is unused.

### 18.5.2 TRANSMIT MESSAGE PRIORITY

Transmit priority is a prioritization within each node of the pending transmittable messages. There are four levels of transmit priority. If the TXnPRI<1:0> bits (in CiTRmnCON) for a particular message buffer are set to '11', that buffer has the highest priority. If the TXnPRI<1:0> bits for a particular message buffer are set to '10' or '01', that buffer has an intermediate priority. If the TXnPRI<1:0> bits for a particular message buffer are '00', that buffer has the lowest priority. If two or more pending messages have the same priority, the messages are transmitted in decreasing order of buffer index.

### 18.5.3 TRANSMISSION SEQUENCE

To initiate transmission of the message, the TXREQn bit (in CiTRmnCON) must be set. The CAN bus module resolves any timing conflicts between the setting of the TXREQn bit and the Start-of-Frame (SOF), ensuring that if the priority was changed, it is resolved correctly before the SOF occurs. When TXREQn is set, the TXABTn, TXLARBn and TXERRn flag bits are automatically cleared.

Setting the TXREQn bit simply flags a message buffer as enqueued for transmission. When the module detects an available bus, it begins transmitting the message which has been determined to have the highest priority.

If the transmission completes successfully on the first attempt, the TXREQn bit is cleared automatically and an interrupt is generated if TXnIE was set.

If the message transmission fails, one of the error condition flags will be set and the TXREQn bit will remain set, indicating that the message is still pending for transmission. If the message encountered an error condition during the transmission attempt, the TXERRn bit will be set and the error condition may cause an interrupt. If the message loses arbitration during the transmission attempt, the TXLARBn bit is set. No interrupt is generated to signal the loss of arbitration.

### 18.5.4 AUTOMATIC PROCESSING OF REMOTE TRANSMISSION REQUESTS

If the RTRENn bit (in the CiTRmnCON register) for a particular transmit buffer is set, the hardware automatically transmits the data in that buffer in response to remote transmission requests matching the filter that points to that particular buffer. The user does not need to manually initiate a transmission in this case.

### 18.5.5 ABORTING MESSAGE TRANSMISSION

The system can also abort a message by clearing the TXREQ bit associated with each message buffer. Setting the ABAT bit (CiCTRL1<12>) will request an abort of all pending messages. If the message has not yet started transmission, or if the message started but is interrupted by loss of arbitration or an error, the abort will be processed. The abort is indicated when the module sets the TXABT bit and the TXnIF flag is not automatically set.

### 18.5.6 TRANSMISSION ERRORS

The CAN module will detect the following transmission errors:

- Acknowledge Error
- Form Error
- Bit Error

These transmission errors will not necessarily generate an interrupt, but are indicated by the transmission error counter. However, each of these errors will cause the transmission error counter to be incremented by one. Once the value of the error counter exceeds the value of 96, the ERRIF (CiINTF<5>) and the TXWAR bit (CiINTF<10>) are set. Once the value of the error counter exceeds the value of 96, an interrupt is generated and the TXWAR bit in the Interrupt Flag register is set.

# PIC24H

## 18.5.7 TRANSMIT INTERRUPTS

Transmit interrupts can be divided into 2 major groups, each including various conditions that generate interrupts:

- **Transmit Interrupt:**

At least one of the three transmit buffers is empty (not scheduled) and can be loaded to schedule a message for transmission. Reading the TXnIF flags will indicate which transmit buffer is available and caused the interrupt.

- **Transmit Error Interrupts:**

A transmission error interrupt will be indicated by the ERRIF flag. This flag shows that an error condition occurred. The source of the error can be determined by checking the error flags in the CAN Interrupt Flag register, CiINTF. The flags in this register are related to receive and transmit errors.

- **Transmitter Warning Interrupt:**

The TXWAR bit indicates that the transmit error counter has reached the CPU warning limit of 96.

- **Transmitter Error Passive:**

The TXEP bit (CiINTF<12>) indicates that the transmit error counter has exceeded the error passive limit of 127 and the module has gone to error passive state.

- **Bus Off:**

The TXBO bit (CiINTF<13>) indicates that the transmit error counter has exceeded 255 and the module has gone to the bus off state.

**Note:** Both ECAN1 and ECAN2 can trigger a DMA data transfer. If C1TX, C1RX, C2TX or C2RX is selected as a DMA IRQ source, a DMA transfer occurs when the C1TXIF, C1RXIF, C2TXIF or C2RXIF bit gets set as a result of an ECAN1 or ECAN2 transmission or reception.

## 18.6 Baud Rate Setting

All nodes on any particular CAN bus must have the same nominal bit rate. In order to set the baud rate, the following parameters have to be initialized:

- Synchronization Jump Width
- Baud Rate Prescaler
- Phase Segments
- Length Determination of Phase Segment 2
- Sample Point
- Propagation Segment bits

### 18.6.1 BIT TIMING

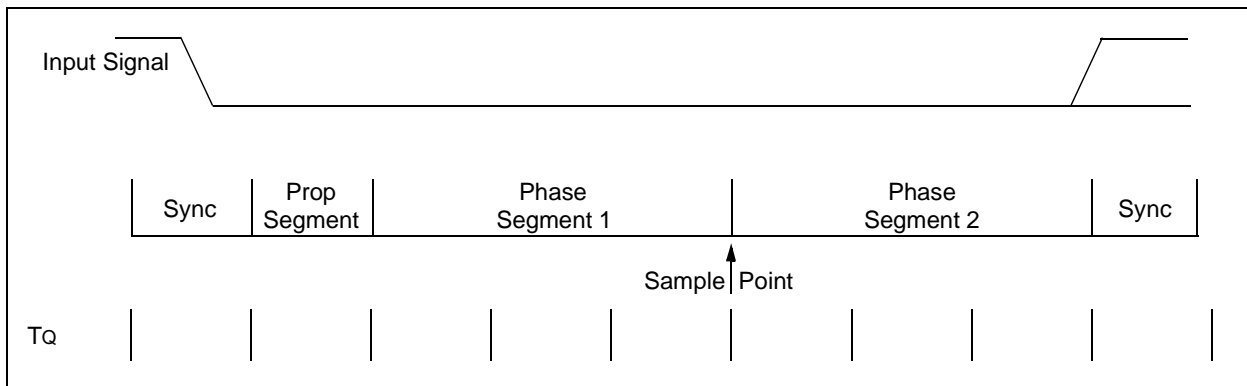
All controllers on the CAN bus must have the same baud rate and bit length. However, different controllers are not required to have the same master oscillator clock. At different clock frequencies of the individual controllers, the baud rate has to be adjusted by adjusting the number of time quanta in each segment.

The nominal bit time can be thought of as being divided into separate non-overlapping time segments. These segments are shown in Figure 18-2.

- Synchronization Segment (Sync Seg)
- Propagation Time Segment (Prop Seg)
- Phase Segment 1 (Phase1 Seg)
- Phase Segment 2 (Phase2 Seg)

The time segments and also the nominal bit time are made up of integer units of time called time quanta or T<sub>Q</sub>. By definition, the nominal bit time has a minimum of 8 T<sub>Q</sub> and a maximum of 25 T<sub>Q</sub>. Also, by definition, the minimum nominal bit time is 1 μsec corresponding to a maximum bit rate of 1 MHz.

**FIGURE 18-2: ECAN™ MODULE BIT TIMING**





## 18.6.2 PRESCALER SETTING

There is a programmable prescaler with integral values ranging from 1 to 64, in addition to a fixed divide-by-2 for clock generation. The time quantum (T<sub>Q</sub>) is a fixed unit of time derived from the oscillator period and is given by Equation 18-1.

**Note:** F<sub>CAN</sub> must not exceed 40 MHz. If CANCKS = 0, then F<sub>CY</sub> must not exceed 20 MHz.

### EQUATION 18-1: TIME QUANTUM FOR CLOCK GENERATION

$$TQ = 2 (BRP<5:0> + 1) / F_{CAN}$$

## 18.6.3 PROPAGATION SEGMENT

This part of the bit time is used to compensate physical delay times within the network. These delay times consist of the signal propagation time on the bus line and the internal delay time of the nodes. The Prop Seg can be programmed from 1 T<sub>Q</sub> to 8 T<sub>Q</sub> by setting the PRSEG<2:0> bits (CiCFG2<2:0>).

## 18.6.4 PHASE SEGMENTS

The phase segments are used to optimally locate the sampling of the received bit within the transmitted bit time. The sampling point is between Phase1 Seg and Phase2 Seg. These segments are lengthened or shortened by resynchronization. The end of the Phase1 Seg determines the sampling point within a bit period. The segment is programmable from 1 T<sub>Q</sub> to 8 T<sub>Q</sub>. Phase2 Seg provides delay to the next transmitted data transition. The segment is programmable from 1 T<sub>Q</sub> to 8 T<sub>Q</sub>, or it may be defined to be equal to the greater of Phase1 Seg or the information processing time (2 T<sub>Q</sub>). The Phase1 Seg is initialized by setting bits SEG1PH<2:0> (CiCFG2<5:3>) and Phase2 Seg is initialized by setting SEG2PH<2:0> (CiCFG2<10:8>).

The following requirement must be fulfilled while setting the lengths of the phase segments:

$$\text{Prop Seg} + \text{Phase1 Seg} \geq \text{Phase2 Seg}$$

## 18.6.5 SAMPLE POINT

The sample point is the point of time at which the bus level is read and interpreted as the value of that respective bit. The location is at the end of Phase1 Seg. If the bit timing is slow and contains many T<sub>Q</sub>, it is possible to specify multiple sampling of the bus line at the sample point. The level determined by the CAN bus then corresponds to the result from the majority decision of three values. The majority samples are taken at the sample point and twice before with a distance of T<sub>Q</sub>/2. The CAN module allows the user to choose between sampling three times at the same point or once at the same point, by setting or clearing the SAM bit (CiCFG2<6>).

Typically, the sampling of the bit should take place at about 60-70% through the bit time, depending on the system parameters.

## 18.6.6 SYNCHRONIZATION

To compensate for phase shifts between the oscillator frequencies of the different bus stations, each CAN controller must be able to synchronize to the relevant signal edge of the incoming signal. When an edge in the transmitted data is detected, the logic will compare the location of the edge to the expected time (Synchronous Segment). The circuit will then adjust the values of Phase1 Seg and Phase2 Seg. There are two mechanisms used to synchronize.

### 18.6.6.1 Hard Synchronization

Hard synchronization is only done whenever there is a 'recessive' to 'dominant' edge during bus Idle, indicating the start of a message. After hard synchronization, the bit time counters are restarted with the Sync Seg. Hard synchronization forces the edge which has caused the hard synchronization to lie within the synchronization segment of the restarted bit time. If a hard synchronization is done, there will not be a resynchronization within that bit time.

### 18.6.6.2 Resynchronization

As a result of resynchronization, Phase1 Seg may be lengthened or Phase2 Seg may be shortened. The amount of lengthening or shortening of the phase buffer segment has an upper boundary known as the synchronization jump width, and is specified by the SJW<1:0> bits (CiCFG1<7:6>). The value of the synchronization jump width will be added to Phase1 Seg or subtracted from Phase2 Seg. The resynchronization jump width is programmable between 1 T<sub>Q</sub> and 4 T<sub>Q</sub>.

The following requirement must be fulfilled while setting the SJW<1:0> bits:

$$\text{Phase2 Seg} > \text{Synchronization Jump Width}$$

**Note:** In the register descriptions that follow, 'i' in the register identifier denotes the specific ECAN module (ECAN1 or ECAN2).  
 'n' in the register identifier denotes the buffer, filter or mask number.  
 'm' in the register identifier denotes the word number within a particular CAN data field.

# PIC24H

## REGISTER 18-1: CiCTRL1: ECAN MODULE CONTROL REGISTER 1

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0
—	—	CSIDL	ABAT	CANCKS	REQOP<2:0>		
bit 15						bit 8	

R-1	R-0	R-0	U-0	R/W-0	U-0	U-0	R/W-0
OPMODE<2:0>			—	CANCAP	—	—	WIN
bit 7						bit 0	

### Legend:

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

- bit 15-14      **Unimplemented:** Read as '0'
- bit 13        **CSIDL:** Stop in Idle Mode bit  
                   1 = Discontinue module operation when device enters Idle mode  
                   0 = Continue module operation in Idle mode
- bit 12        **ABAT:** Abort All Pending Transmissions bit  
                   Signal all transmit buffers to abort transmission. Module will clear this bit when all transmissions are aborted
- bit 11        **CANCKS:** CAN Master Clock Select bit  
                   1 = CAN FCAN clock is FCY  
                   0 = CAN FCAN clock is FOSC
- bit 10-8      **REQOP<2:0>:** Request Operation Mode bits  
                   000 = Set Normal Operation mode  
                   001 = Set Disable mode  
                   010 = Set Loopback mode  
                   011 = Set Listen Only Mode  
                   100 = Set Configuration mode  
                   101 = Reserved – do not use  
                   110 = Reserved – do not use  
                   111 = Set Listen All Messages mode
- bit 7-5       **OPMODE<2:0>:** Operation Mode bits  
                   000 = Module is in Normal Operation mode  
                   001 = Module is in Disable mode  
                   010 = Module is in Loopback mode  
                   011 = Module is in Listen Only mode  
                   100 = Module is in Configuration mode  
                   101 = Reserved  
                   110 = Reserved  
                   111 = Module is in Listen All Messages mode
- bit 4         **Unimplemented:** Read as '0'
- bit 3         **CANCAP:** CAN Message Receive Timer Capture Event Enable bit  
                   1 = Enable input capture based on CAN message receive  
                   0 = Disable CAN capture
- bit 2-1      **Unimplemented:** Read as '0'
- bit 0         **WIN:** SFR Map Window Select bit  
                   1 = Use filter window  
                   0 = Use buffer window

## REGISTER 18-2: CICTRL2: ECAN MODULE CONTROL REGISTER 2

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	R-0	R-0	R-0	R-0	R-0
—	—	—	DNCNT<4:0>				
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15-5

**Unimplemented:** Read as '0'

bit 4-0

**DNCNT<4:0>:** DeviceNet™ Filter Bit Number bits

10010-11111 = Invalid selection

10001 = Compare up to data byte 3, bit 6 with EID<17>

....

00001 = Compare up to data byte 1, bit 7 with EID<0>

00000 = Do not compare data bytes

# PIC24H

## REGISTER 18-3: CIVEC: ECAN MODULE INTERRUPT CODE REGISTER

U-0	U-0	U-0	R-0	R-0	R-0	R-0	R-0
—	—	—	FILHIT<4:0>				
bit 15							bit 8

U-0	R-1	R-0	R-0	R-0	R-0	R-0	R-0
—	ICODE<6:0>						
bit 7							bit 0

### Legend:

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

bit 15-13                      **Unimplemented:** Read as '0'

bit 12-8                      **FILHIT<4:0>:** Filter Hit Number bits

10000-11111 = Reserved

01111 = Filter 15

....

00001 = Filter 1

00000 = Filter 0

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

bit 6-0                      **ICODE<6:0>:** Interrupt Flag Code bits

1000101-1111111 = Reserved

1000100 = FIFO almost full interrupt

1000011 = Receiver overflow interrupt

1000010 = Wake-up interrupt

1000001 = Error interrupt

1000000 = No interrupt

0010000-0111111 = Reserved

0001111 = RB15 buffer Interrupt

....

0001001 = RB9 buffer interrupt

0001000 = RB8 buffer interrupt

0000111 = TRB7 buffer interrupt

0000110 = TRB6 buffer interrupt

0000101 = TRB5 buffer interrupt

0000100 = TRB4 buffer interrupt

0000011 = TRB3 buffer interrupt

0000010 = TRB2 buffer interrupt

0000001 = TRB1 buffer interrupt

0000000 = TRB0 Buffer interrupt

## REGISTER 18-4: CifCTRL: ECAN MODULE FIFO CONTROL REGISTER

R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0	U-0	
DMABS<2:0>			—	—	—	—	—	
bit 15								bit 8
U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	
—	—	—	FSA<4:0>					
bit 7								bit 0

### Legend:

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

bit 15-13      **DMABS<2:0>**: DMA Buffer Size bits

111 = Reserved  
 110 = 32 buffers in DMA RAM  
 101 = 24 buffers in DMA RAM  
 100 = 16 buffers in DMA RAM  
 011 = 12 buffers in DMA RAM  
 010 = 8 buffers in DMA RAM  
 001 = 6 buffers in DMA RAM  
 000 = 4 buffers in DMA RAM

bit 12-5      **Unimplemented**: Read as '0'

bit 4-0      **FSA<4:0>**: FIFO Area Starts with Buffer bits

11111 = RB31 buffer  
 11110 = RB30 buffer  
 . . . .  
 00001 = TRB1 buffer  
 00000 = TRB0 buffer

# PIC24H

## REGISTER 18-5: CIFIFO: ECAN MODULE FIFO STATUS REGISTER

U-0	U-0	R-0	R-0	R-0	R-0	R-0	R-0	
—	—	FBP<5:0>						
bit 15							bit 8	

U-0	U-0	R-0	R-0	R-0	R-0	R-0	R-0	
—	—	FNRB<5:0>						
bit 7							bit 0	

### Legend:

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

- bit 15-14      **Unimplemented:** Read as '0'
- bit 13-8      **FBP<5:0>:** FIFO Write Buffer Pointer bits
  - 011111 = RB31 buffer
  - 011110 = RB30 buffer
  - ....
  - 000001 = TRB1 buffer
  - 000000 = TRB0 buffer
- bit 7-6      **Unimplemented:** Read as '0'
- bit 5-0      **FNRB<5:0>:** FIFO Next Read Buffer Pointer bits
  - 011111 = RB31 buffer
  - 011110 = RB30 buffer
  - ....
  - 000001 = TRB1 buffer
  - 000000 = TRB0 buffer

## REGISTER 18-6: CiINTF: ECAN MODULE INTERRUPT FLAG REGISTER

U-0	U-0	R-0	R-0	R-0	R-0	R-0	R-0
—	—	TXBO	TXBP	RXBP	TXWAR	RXWAR	EWARN
bit 15							bit 8

R/C-0	R/C-0	R/C-0	U-0	R/C-0	R/C-0	R/C-0	R/C-0
IVRIF	WAKIF	ERRIF	—	FIFOIF	RBOVIF	RBIF	TBIF
bit 7							bit 0

### Legend:

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

- bit 15-14      **Unimplemented:** Read as '0'
- bit 13        **TXBO:** Transmitter in Error State Bus Off bit
- bit 12        **TXBP:** Transmitter in Error State Bus Passive bit
- bit 11        **RXBP:** Receiver in Error State Bus Passive bit
- bit 10        **TXWAR:** Transmitter in Error State Warning bit
- bit 9         **RXWAR:** Receiver in Error State Warning bit
- bit 8         **EWARN:** Transmitter or Receiver in Error State Warning bit
- bit 7         **IVRIF:** Invalid Message Received Interrupt Flag bit
- bit 6         **WAKIF:** Bus Wake-up Activity Interrupt Flag bit
- bit 5         **ERRIF:** Error Interrupt Flag bit (multiple sources in CiINTF<13:8> register)
- bit 4         **Unimplemented:** Read as '0'
- bit 3         **FIFOIF:** FIFO Almost Full Interrupt Flag bit
- bit 2         **RBOVIF:** RX Buffer Overflow Interrupt Flag bit
- bit 1         **RBIF:** RX Buffer Interrupt Flag bit
- bit 0         **TBIF:** TX Buffer Interrupt Flag bit

# PIC24H

## REGISTER 18-7: CIINTE: ECAN MODULE INTERRUPT ENABLE REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
IVRIE	WAKIE	ERRIE	—	FIFOIE	RBOVIE	RBIE	TBIE
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

- bit 15-8      **Unimplemented:** Read as '0'
- bit 7        **IVRIE:** Invalid Message Received Interrupt Enable bit
- bit 6        **WAKIE:** Bus Wake-up Activity Interrupt Flag bit
- bit 5        **ERRIE:** Error Interrupt Enable bit
- bit 4        **Unimplemented:** Read as '0'
- bit 3        **FIFOIE:** FIFO Almost Full Interrupt Enable bit
- bit 2        **RBOVIE:** RX Buffer Overflow Interrupt Enable bit
- bit 1        **RBIE:** RX Buffer Interrupt Enable bit
- bit 0        **TBIE:** TX Buffer Interrupt Enable bit



## REGISTER 18-8: CiEC: ECAN MODULE TRANSMIT/RECEIVE ERROR COUNT REGISTER

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	
TERRCNT<7:0>								
bit 15								bit 8

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	
RERRCNT<7:0>								
bit 7								bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15-8      **TERRCNT<7:0>**: Transmit Error Count bits

bit 7-0      **RERRCNT<7:0>**: Receive Error Count bits

# PIC24H

## REGISTER 18-9: CiCFG1: ECAN MODULE BAUD RATE CONFIGURATION REGISTER 1

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SJW<1:0>		BRP<5:0>					
bit 7							bit 0

### Legend:

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

- bit 15-8            **Unimplemented:** Read as '0'
- bit 7-6            **SJW<1:0>:** Synchronization Jump Width bits
- 11 = Length is 4 x TQ
  - 10 = Length is 3 x TQ
  - 01 = Length is 2 x TQ
  - 00 = Length is 1 x TQ
- bit 5-0            **BRP<5:0>:** Baud Rate Prescaler bits
- 11 1111 = TQ = 2 x 64 x 1/FCAN
  - 00 0010 = TA = 2 x 3 x 1/FCAN
  - 00 0001 = TA = 2 x 2 x 1/FCAN
  - 00 0000 = TQ = 2 x 1 x 1/FCAN

## REGISTER 18-10: CiCFG2: ECAN MODULE BAUD RATE CONFIGURATION REGISTER 2

U-0	R/W-x	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x
—	WAKFIL	—	—	—	SEG2PH<2:0>		
bit 15						bit 8	

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SEG2PHTS	SAM	SEG1PH<2:0>			PRSEG<2:0>		
bit 7						bit 0	

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

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

bit 14      **WAKFIL:** Select CAN bus Line Filter for Wake-up bit

1 = Use CAN bus line filter for wake-up

0 = CAN bus line filter is not used for wake-up

bit 13-11    **Unimplemented:** Read as '0'

bit 10-8     **SEG2PH<2:0>:** Phase Buffer Segment 2 bits

111 = Length is 8 x TQ

000 = Length is 1 x TQ

bit 7        **SEG2PHTS:** Phase Segment 2 Time Select bit

1 = Freely programmable

0 = Maximum of SEG1PH bits or Information Processing Time (IPT), whichever is greater

bit 6        **SAM:** Sample of the CAN bus Line bit

1 = Bus line is sampled three times at the sample point

0 = Bus line is sampled once at the sample point

bit 5-3      **SEG1PH<2:0>:** Phase Buffer Segment 1 bits

111 = Length is 8 x TQ

000 = Length is 1 x TQ

bit 2-0      **PRSEG<2:0>:** Propagation Time Segment bits

111 = Length is 8 x TQ

000 = Length is 1 x TQ

# PIC24H

## REGISTER 18-11: CIFEN1: ECAN MODULE ACCEPTANCE FILTER ENABLE REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
FLTEN15	FLTEN14	FLTEN13	FLTEN12	FLTEN11	FLTEN10	FLTEN9	FLTEN8
bit 15							bit 8

R/W-0	R/W-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
FLTEN7	FLTEN6	FLTEN5	FLTEN4	FLTEN3	FLTEN2	FLTEN1	FLTEN0
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15-0 **FLTENn**: Enable Filter n to Accept Messages bits

1 = Enable Filter n

0 = Disable Filter n

## REGISTER 18-12: CIBUFPNT1: ECAN MODULE FILTER 0-3 BUFFER POINTER REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
F3BP<3:0>				F2BP<3:0>			
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
F1BP<3:0>				F0BP<3:0>			
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15-12 **F3BP<3:0>**: RX Buffer Written when Filter 3 Hits bits

bit 11-8 **F2BP<3:0>**: RX Buffer Written when Filter 2 Hits bits

bit 7-4 **F1BP<3:0>**: RX Buffer Written when Filter 1 Hits bits

bit 3-0 **F0BP<3:0>**: RX Buffer Written when Filter 0 Hits bits

1111 = Filter hits received in RX FIFO buffer

1110 = Filter hits received in RX Buffer 14

....

0001 = Filter hits received in RX Buffer 1

0000 = Filter hits received in RX Buffer 0

## REGISTER 18-13: CiBUFNT2: ECAN MODULE FILTER 4-7 BUFFER POINTER REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
F7BP<3:0>				F6BP<3:0>			
bit 15				bit 8			

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
F5BP<3:0>				F4BP<3:0>			
bit 7				bit 0			

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

- bit 15-12     **F7BP<3:0>**: RX Buffer Written when Filter 7 Hits bits
- bit 11-8     **F6BP<3:0>**: RX Buffer Written when Filter 6 Hits bits
- bit 7-4      **F5BP<3:0>**: RX Buffer Written when Filter 5 Hits bits
- bit 3-0      **F4BP<3:0>**: RX Buffer Written when Filter 4 Hits bits

## REGISTER 18-14: CiBUFNT3: ECAN MODULE FILTER 8-11 BUFFER POINTER REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
F11BP<3:0>				F10BP<3:0>			
bit 15				bit 8			

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
F9BP<3:0>				F8BP<3:0>			
bit 7				bit 0			

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

- bit 15-12     **F11BP<3:0>**: RX Buffer Written when Filter 11 Hits bits
- bit 11-8     **F10BP<3:0>**: RX Buffer Written when Filter 10 Hits bits
- bit 7-4      **F9BP<3:0>**: RX Buffer Written when Filter 9 Hits bits
- bit 3-0      **F8BP<3:0>**: RX Buffer Written when Filter 8 Hits bits

# PIC24H

## REGISTER 18-15: CiBUFPNT4: ECAN MODULE FILTER 12-15 BUFFER POINTER REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
F15BP<3:0>				F14BP<3:0>			
bit 15				bit 8			

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
F13BP<3:0>				F12BP<3:0>			
bit 7				bit 0			

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15-12     **F15BP<3:0>**: RX Buffer Written when Filter 15 Hits bits

bit 11-8     **F14BP<3:0>**: RX Buffer Written when Filter 14 Hits bits

bit 7-4     **F13BP<3:0>**: RX Buffer Written when Filter 13 Hits bits

bit 3-0     **F12BP<3:0>**: RX Buffer Written when Filter 12 Hits bits

## REGISTER 18-16: C<sub>i</sub>RXF<sub>n</sub>SID: ECAN MODULE ACCEPTANCE FILTER *n* STANDARD IDENTIFIER (*n* = 0, 1, ..., 15)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3
bit 15							bit 8

R/W-x	R/W-x	R/W-x	U-0	R/W-x	U-0	R/W-x	R/W-x
SID2	SID1	SID0	—	EXIDE	—	EID17	EID16
bit 7							bit 0

### Legend:

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

- bit 15-5        **SID<10:0>**: Standard Identifier bits  
                   1 = Message address bit SID<sub>x</sub> must be '1' to match filter  
                   0 = Message address bit SID<sub>x</sub> must be '0' to match filter
- bit 4            **Unimplemented**: Read as '0'
- bit 3            **EXIDE**: Extended Identifier Enable bit  
                   If MIDE = 1 then:  
                   1 = Match only messages with extended identifier addresses  
                   0 = Match only messages with standard identifier addresses  
                   If MIDE = 0 then:  
                   Ignore EXIDE bit.
- bit 2            **Unimplemented**: Read as '0'
- bit 1-0        **EID<17:16>**: Extended Identifier bits  
                   1 = Message address bit EID<sub>x</sub> must be '1' to match filter  
                   0 = Message address bit EID<sub>x</sub> must be '0' to match filter

## REGISTER 18-17: C<sub>i</sub>RXF<sub>n</sub>EID: ECAN MODULE ACCEPTANCE FILTER *n* EXTENDED IDENTIFIER (*n* = 0, 1, ..., 15)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8
bit 15							bit 8

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0
bit 7							bit 0

### Legend:

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

- bit 15-0        **EID<15:0>**: Extended Identifier bits  
                   1 = Message address bit EID<sub>x</sub> must be '1' to match filter  
                   0 = Message address bit EID<sub>x</sub> must be '0' to match filter

# PIC24H

## REGISTER 18-18: C1FMSKSEL1: ECAN MODULE FILTER 7-0 MASK SELECTION REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
F7MSK<1:0>		F6MSK<1:0>		F5MSK<1:0>		F4MSK<1:0>	
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
F3MSK<1:0>		F2MSK<1:0>		F1MSK<1:0>		F0MSK<1:0>	
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15-14     **F7MSK<1:0>**: Mask Source for Filter 7 bit

bit 13-12     **F6MSK<1:0>**: Mask Source for Filter 6 bit

bit 11-10     **F5MSK<1:0>**: Mask Source for Filter 5 bit

bit 9-8        **F4MSK<1:0>**: Mask Source for Filter 4 bit

bit 7-6        **F3MSK<1:0>**: Mask Source for Filter 3 bit

bit 5-4        **F2MSK<1:0>**: Mask Source for Filter 2 bit

bit 3-2        **F1MSK<1:0>**: Mask Source for Filter 1 bit

bit 1-0        **F0MSK<1:0>**: Mask Source for Filter 0 bit

11 = No mask

10 = Acceptance Mask 2 registers contain mask

01 = Acceptance Mask 1 registers contain mask

00 = Acceptance Mask 0 registers contain mask



## REGISTER 18-19: C<sub>i</sub>RX<sub>M</sub>nSID: ECAN MODULE ACCEPTANCE FILTER MASK n STANDARD IDENTIFIER

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3
bit 15							bit 8

R/W-x	R/W-x	R/W-x	U-0	R/W-x	U-0	R/W-x	R/W-x
SID2	SID1	SID0	—	MIDE	—	EID17	EID16
bit 7							bit 0

### Legend:

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

- bit 15-5        **SID<10:0>**: Standard Identifier bits
  - 1 = Include bit SID<sub>x</sub> in filter comparison
  - 0 = Bit SID<sub>x</sub> is don't care in filter comparison
- bit 4            **Unimplemented**: Read as '0'
- bit 3            **MIDE**: Identifier Receive Mode bit
  - 1 = Match only message types (standard or extended address) that correspond to EXIDE bit in filter
  - 0 = Match either standard or extended address message if filters match  
     (i.e., if (Filter SID) = (Message SID) or if (Filter SID/EID) = (Message SID/EID))
- bit 2            **Unimplemented**: Read as '0'
- bit 1-0        **EID<17:16>**: Extended Identifier bits
  - 1 = Include bit EID<sub>x</sub> in filter comparison
  - 0 = Bit EID<sub>x</sub> is don't care in filter comparison

## REGISTER 18-20: C<sub>i</sub>RX<sub>M</sub>nEID: ECAN TECHNOLOGY ACCEPTANCE FILTER MASK n EXTENDED IDENTIFIER

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8
bit 15							bit 8

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0
bit 7							bit 0

### Legend:

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

- bit 15-0        **EID<15:0>**: Extended Identifier bits
  - 1 = Include bit EID<sub>x</sub> in filter comparison
  - 0 = Bit EID<sub>x</sub> is don't care in filter comparison

# PIC24H

## REGISTER 18-21: C<sub>i</sub>RXFUL1: ECAN MODULE RECEIVE BUFFER FULL REGISTER 1

R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
RXFUL15	RXFUL14	RXFUL13	RXFUL12	RXFUL11	RXFUL10	RXFUL9	RXFUL8
bit 15							bit 8

R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
RXFUL7	RXFUL6	RXFUL5	RXFUL4	RXFUL3	RXFUL2	RXFUL1	RXFUL0
bit 7							bit 0

### Legend:

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

bit 15-0      **RXFUL<15:0>**: Receive Buffer n Full bits  
1 = Buffer is full (set by module)  
0 = Buffer is empty (clear by application software)

## REGISTER 18-22: C<sub>i</sub>RXFUL2: ECAN MODULE RECEIVE BUFFER FULL REGISTER 2

R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
RXFUL31	RXFUL30	RXFUL29	RXFUL28	RXFUL27	RXFUL26	RXFUL25	RXFUL24
bit 15							bit 8

R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
RXFUL23	RXFUL22	RXFUL21	RXFUL20	RXFUL19	RXFUL18	RXFUL17	RXFUL16
bit 7							bit 0

### Legend:

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

bit 15-0      **RXFUL<31:16>**: Receive Buffer n Full bits  
1 = Buffer is full (set by module)  
0 = Buffer is empty (clear by application software)

## REGISTER 18-23: C<sub>i</sub>RXOVF1: ECAN MODULE RECEIVE BUFFER OVERFLOW REGISTER 1

R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
RXOVF15	RXOVF14	RXOVF13	RXOVF12	RXOVF11	RXOVF10	RXOVF9	RXOVF8
bit 15							bit 8

R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
RXOVF7	RXOVF6	RXOVF5	RXOVF4	RXOVF3	RXOVF2	RXOVF1	RXOVF0
bit 7							bit 0

### Legend:

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

bit 15-0                      **RXOVF<15:0>**: Receive Buffer n Overflow bits  
    1 = Module pointed a write to a full buffer (set by module)  
    0 = Overflow is cleared (clear by application software)

## REGISTER 18-24: C<sub>i</sub>RXOVF2: ECAN MODULE RECEIVE BUFFER OVERFLOW REGISTER 2

R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
RXOVF31	RXOVF30	RXOVF29	RXOVF28	RXOVF27	RXOVF26	RXOVF25	RXOVF24
bit 15							bit 8

R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
RXOVF23	RXOVF22	RXOVF21	RXOVF20	RXOVF19	RXOVF18	RXOVF17	RXOVF16
bit 7							bit 0

### Legend:

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

bit 15-0                      **RXOVF<31:16>**: Receive Buffer n Overflow bits  
    1 = Module pointed a write to a full buffer (set by module)  
    0 = Overflow is cleared (clear by application software)

# PIC24H

## REGISTER 18-25: CnTRmnCON: ECAN MODULE TX/RX BUFFER m CONTROL REGISTER (m = 0,2,4,6; n = 1,3,5,7)

R/W-0	R-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
TXENn	TXABTn	TXLARBn	TXERRn	TXREQn	RTRENn	TXnPRI<1:0>	
bit 15							bit 8

R/W-0	R-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
TXENm	TXABTm <sup>(1)</sup>	TXLARBm <sup>(1)</sup>	TXERRm <sup>(1)</sup>	TXREQm	RTRENm	TXmPRI<1:0>	
bit 7							bit 0

### Legend:

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

bit 15-8                      **See Definition for Bits 7-0, Controls Buffer n**

bit 7                      **TXENm:** TX/RX Buffer Selection bit

1 = Buffer TRBn is a transmit buffer  
 0 = Buffer TRBn is a receive buffer

bit 6                      **TXABTm:** Message Aborted bit<sup>(1)</sup>

1 = Message was aborted  
 0 = Message completed transmission successfully

bit 5                      **TXLARBm:** Message Lost Arbitration bit<sup>(1)</sup>

1 = Message lost arbitration while being sent  
 0 = Message did not lose arbitration while being sent

bit 4                      **TXERRm:** Error Detected During Transmission bit<sup>(1)</sup>

1 = A bus error occurred while the message was being sent  
 0 = A bus error did not occur while the message was being sent

bit 3                      **TXREQm:** Message Send Request bit

Setting this bit to '1' requests sending a message. The bit will automatically clear when the message is successfully sent. Clearing the bit to '0' while set will request a message abort.

bit 2                      **RTRENm:** Auto-Remote Transmit Enable bit

1 = When a remote transmit is received, TXREQ will be set  
 0 = When a remote transmit is received, TXREQ will be unaffected

bit 1-0                      **TXmPRI<1:0>:** Message Transmission Priority bits

11 = Highest message priority  
 10 = High intermediate message priority  
 01 = Low intermediate message priority  
 00 = Lowest message priority

**Note 1:** This bit is cleared when TXREQ is set.

**Note:** The buffers, SID, EID, DLC, Data Field and Receive Status registers are stored in DMA RAM. These are not Special Function Registers.

**REGISTER 18-26: CiTRBnSID: ECAN MODULE BUFFER n STANDARD IDENTIFIER**  
(n = 0, 1, ..., 31)

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	SID10	SID9	SID8	SID7	SID6
bit 15							bit 8

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SID5	SID4	SID3	SID2	SID1	SID0	SRR	IDE
bit 7							bit 0

**Legend:**

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

- bit 15-13      **Unimplemented:** Read as '0'
- bit 12-2      **SID<10:0>:** Standard Identifier bits
- bit 1          **SRR:** Substitute Remote Request bit  
                  1 = Message will request remote transmission  
                  0 = Normal message
- bit 0          **IDE:** Extended Identifier bit  
                  1 = Message will transmit extended identifier  
                  0 = Message will transmit standard identifier

**REGISTER 18-27: CiTRBnEID: ECAN MODULE BUFFER n EXTENDED IDENTIFIER**  
(n = 0, 1, ..., 31)

U-0	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	—	EID17	EID16	EID15	EID14
bit 15							bit 8

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID13	EID12	EID11	EID10	EID9	EID8	EID7	EID6
bit 7							bit 0

**Legend:**

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

- bit 15-12      **Unimplemented:** Read as '0'
- bit 11-0      **EID<17:6>:** Extended Identifier bits

# PIC24H

## REGISTER 18-28: CiTRBnDLC: ECAN MODULE BUFFER n DATA LENGTH CONTROL (n = 0, 1, ..., 31)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID5	EID4	EID3	EID2	EID1	EID0	RTR	RB1
bit 15							bit 8

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	RB0	DLC3	DLC2	DLC1	DLC0
bit 7							bit 0

### Legend:

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

- bit 15-10      **EID<5:0>**: Extended Identifier bits
- bit 9          **RTR**: Remote Transmission Request bit  
                  1 = Message will request remote transmission  
                  0 = Normal message
- bit 8          **RB1**: Reserved Bit 1  
                  User must set this bit to '0' per CAN protocol.
- bit 7-5        **Unimplemented**: Read as '0'
- bit 4          **RB0**: Reserved Bit 0  
                  User must set this bit to '0' per CAN protocol.
- bit 3-0        **DLC<3:0>**: Data Length Code bits

## REGISTER 18-29: CiTRBnDm: ECAN MODULE BUFFER n DATA FIELD BYTE m (n = 0, 1, ..., 31; m = 0, 1, ..., 7)<sup>(1)</sup>

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
TRBnDm7	TRBnDm6	TRBnDm5	TRBnDm4	TRBnDm3	TRBnDm2	TRBnDm1	TRBnDm0
bit 7							bit 0

### Legend:

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

- bit 7-0        **TRnDm<7:0>**: Data Field Buffer 'n' Byte 'm' bits

**Note 1:** The Most Significant Byte contains byte (m + 1) of the buffer.

**REGISTER 18-30: CiTRBnSTAT: ECAN MODULE RECEIVE BUFFER n STATUS  
(n = 0, 1, ..., 31)**

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	
—	—	—	FILHIT4	FILHIT3	FILHIT2	FILHIT1	FILHIT0	
bit 15								bit 8

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0	
—	—	—	—	—	—	—	—	
bit 7								bit 0

**Legend:**

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

bit 15-13      **Unimplemented:** Read as '0'

bit 12-8      **FILHIT<4:0>:** Filter Hit Code bits (only written by module for receive buffers, unused for transmit buffers)  
Encodes number of filter that resulted in writing this buffer.

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

# PIC24H

---

NOTES:



## 19.0 10-BIT/12-BIT A/D CONVERTER

**Note:** This data sheet summarizes the features of this group of PIC24H devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the “*dsPIC30F Family Reference Manual*” (DS70046).

The PIC24H devices have up to 32 A/D input channels. These devices also have up to 2 A/D converter modules (ADC<sub>x</sub>, where ‘x’ = 1 or 2), each with its own set of Special Function Registers.

The AD12B bit (AD<sub>x</sub>CON1<10>) allows each of the A/D modules to be configured by the user as either a 10-bit, 4-sample/hold A/D (default configuration) or a 12-bit, 1-sample/hold A/D.

**Note:** The A/D module needs to be disabled before modifying the AD12B bit.

### 19.1 Key Features

The 10-bit A/D configuration has the following key features:

- Successive Approximation (SAR) conversion
- Conversion speeds of up to 1.1 Msps
- Up to 32 analog input pins
- External voltage reference input pins
- Simultaneous sampling of up to four analog input pins
- Automatic Channel Scan mode
- Selectable conversion trigger source
- Selectable Buffer Fill modes
- Four result alignment options (signed/unsigned, fractional/integer)
- Operation during CPU Sleep and Idle modes

The 12-bit A/D configuration supports all the above features, except:

- In the 12-bit configuration, conversion speeds of up to 500 ksps are supported
- There is only 1 sample/hold amplifier in the 12-bit configuration, so simultaneous sampling of multiple channels is not supported.

Depending on the particular device pinout, the A/D converter can have up to 32 analog input pins, designated AN0 through AN31. 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 19-1.

## 19.2 A/D Initialization

The following configuration steps should be performed.

1. Configure the A/D module:
  - a) Select port pins as analog inputs (AD<sub>x</sub>PCFGH<15:0> or AD<sub>x</sub>PCFGL<15:0>)
  - b) Select voltage reference source to match expected range on analog inputs (AD<sub>x</sub>CON2<15:13>)
  - c) Select the analog conversion clock to match desired data rate with processor clock (AD<sub>x</sub>CON3<5:0>)
  - d) Determine how many S/H channels will be used (AD<sub>x</sub>CON2<9:8> and AD<sub>x</sub>PCFGH<15:0> or AD<sub>x</sub>PCFGL<15:0>)
  - e) Select the appropriate sample/conversion sequence (AD<sub>x</sub>CON1<7:5> and AD<sub>x</sub>CON3<12:8>)
  - f) Select how conversion results are presented in the buffer (AD<sub>x</sub>CON1<9:8>)
  - g) Turn on A/D module (AD<sub>x</sub>CON1<15>)
2. Configure A/D interrupt (if required):
  - a) Clear the AD<sub>x</sub>IF bit
  - b) Select A/D interrupt priority

## 19.3 ADC and DMA

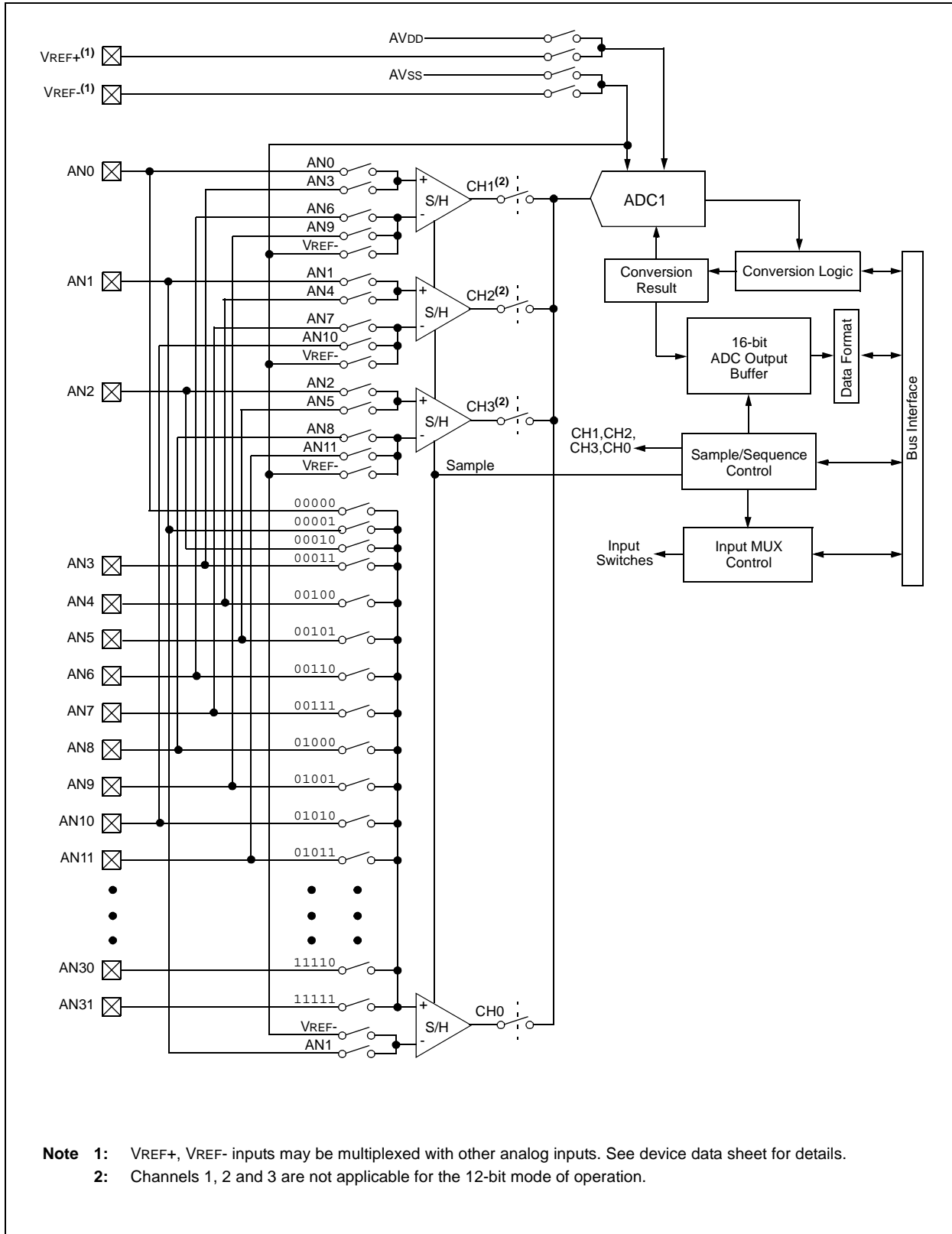
If more than one conversion result needs to be buffered before triggering an interrupt, DMA data transfers can be used. Both ADC1 and ADC2 can trigger a DMA data transfer. If ADC1 or ADC2 is selected as the DMA IRQ source, a DMA transfer occurs when the AD1IF or AD2IF bit gets set as a result of an ADC1 or ADC2 sample conversion sequence.

The SMPI<3:0> bits (AD<sub>x</sub>CON2<5:2>) are used to select how often the DMA RAM buffer pointer is incremented.

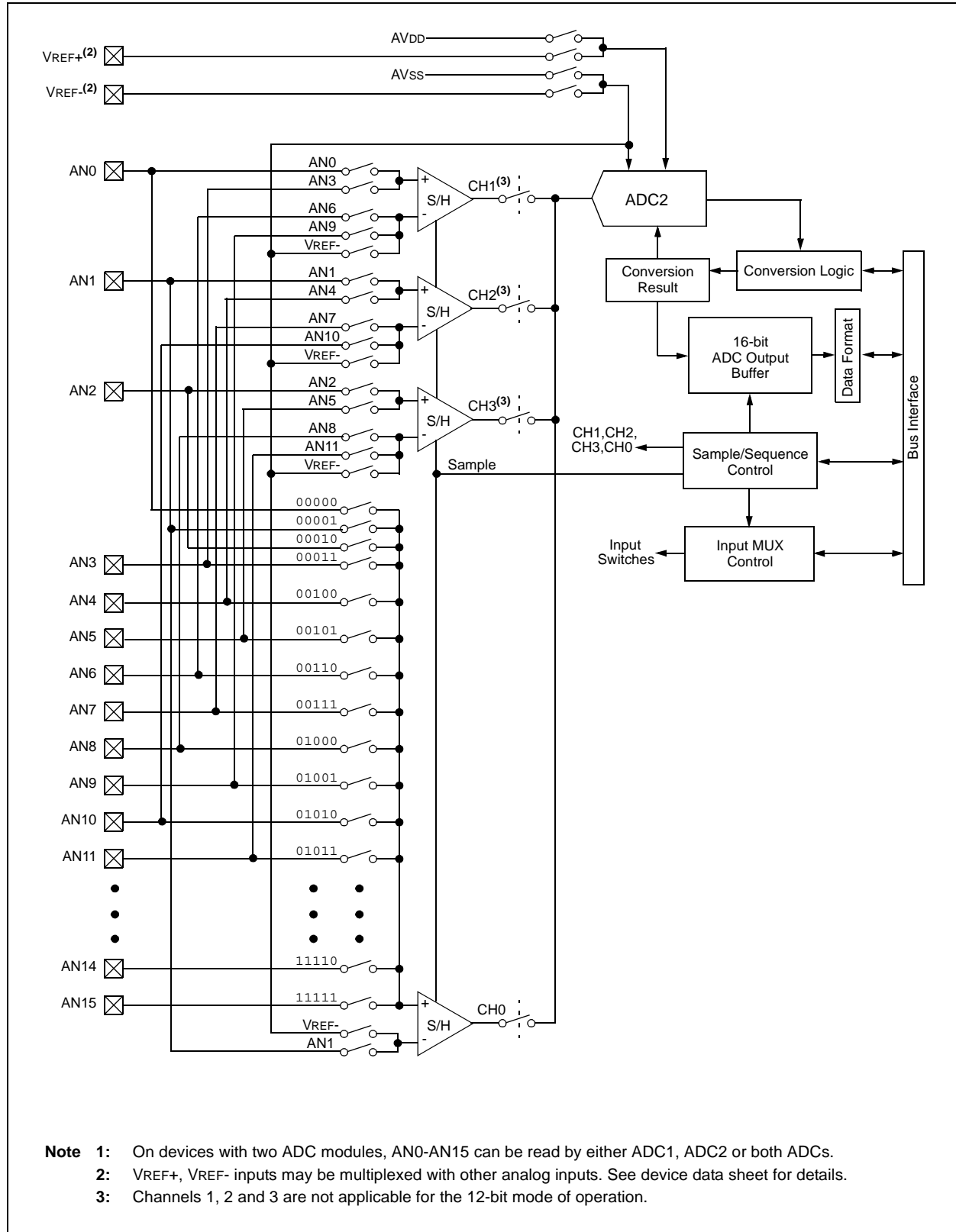
The ADDMABM bit (AD<sub>x</sub>CON1<12>) determines how the conversion results are filled in the DMA RAM buffer area being used for ADC. If this bit is set, DMA buffers are written in the order of conversion. The module will provide an address to the DMA channel that is the same as the address used for the non-DMA stand-alone buffer. If the ADDMABM bit is cleared, then DMA buffers are written in Scatter/Gather mode. The module will provide a scatter/gather address to the DMA channel, based on the index of the analog input and the size of the DMA buffer.

# PIC24H

**FIGURE 19-1: ADC1 MODULE BLOCK DIAGRAM**



**FIGURE 19-2: ADC2 MODULE BLOCK DIAGRAM<sup>(1)</sup>**



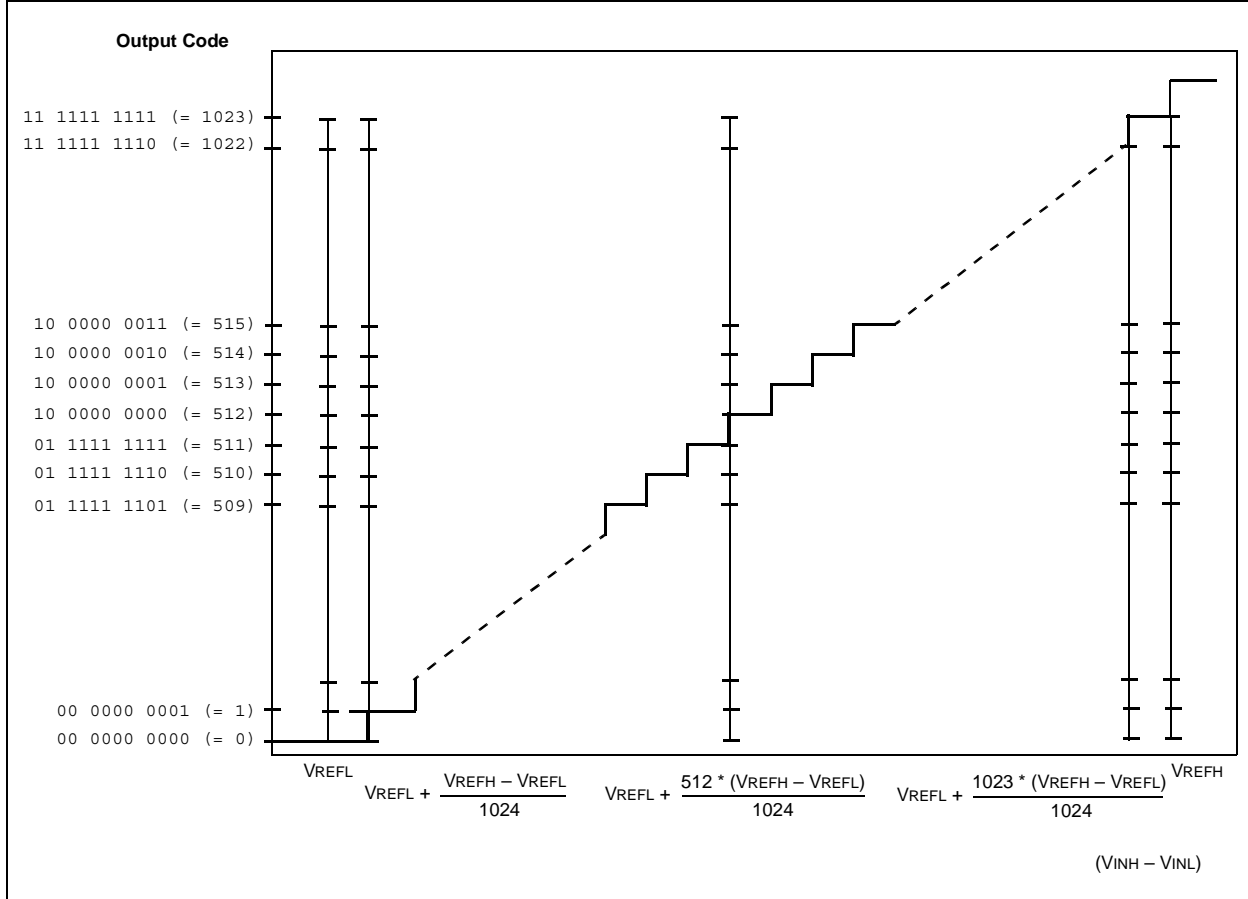
# PIC24H

**EQUATION 19-1: A/D CONVERSION CLOCK PERIOD**

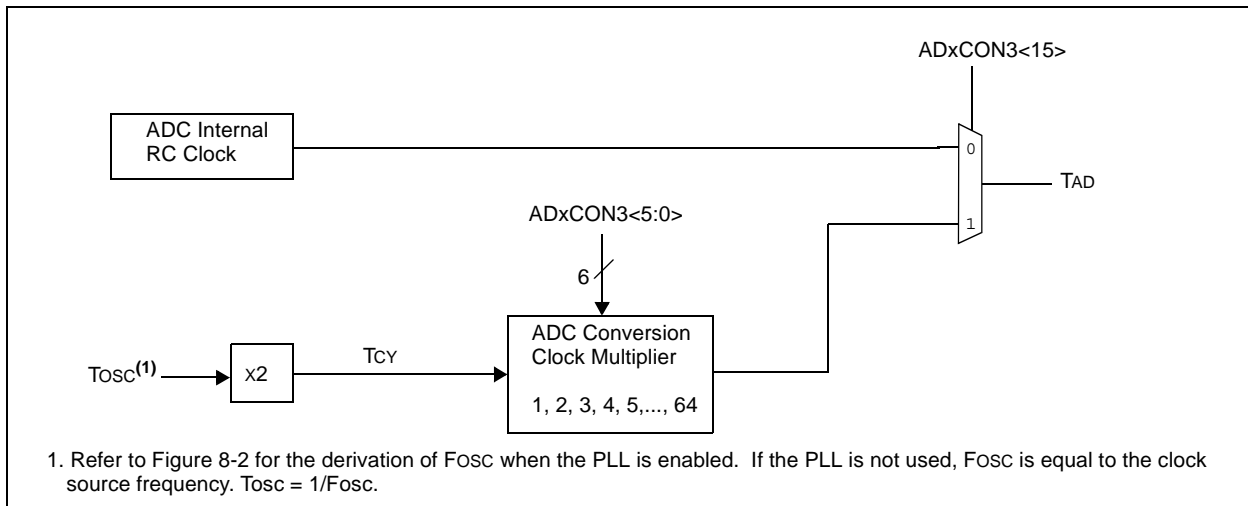
$$T_{AD} = T_{CY}(ADCS + 1)$$

$$ADCS = \frac{T_{AD}}{T_{CY}} - 1$$

**FIGURE 19-3: A/D TRANSFER FUNCTION (10-BIT EXAMPLE)**



**FIGURE 19-4: ADC CONVERSION CLOCK PERIOD BLOCK DIAGRAM**



**REGISTER 19-1: ADxCON1: ADCx CONTROL REGISTER 1 (where x = 1 or 2)**

R/W-0	U-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
ADON	—	ADSIDL	ADDMABM	—	AD12B	FORM<1:0>	
bit 15						bit 8	

R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/C-0
SSRC<2:0>			—	SIMSAM	ASAM	SAMP	HC, HS HC, HS
bit 7						bit 0	

<b>Legend:</b>	HC = Cleared by hardware	HS = Set by hardware
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared
		x = Bit is unknown

- bit 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      **ADDMABM:** DMA Buffer Build Mode bit  
 1 = DMA buffers are written in the order of conversion. The module will provide an address to the DMA channel that is the same as the address used for the non-DMA stand-alone buffer.  
 0 = DMA buffers are written in Scatter/Gather mode. The module will provide a scatter/gather address to the DMA channel, based on the index of the analog input and the size of the DMA buffer.
- bit 11      **Unimplemented:** Read as '0'
- bit 10      **AD12B:** 10-bit or 12-bit Operation Mode bit  
 1 = 12-bit, 1-channel A/D operation  
 0 = 10-bit, 4-channel A/D operation
- bit 9-8     **FORM<1:0>:** Data Output Format bits  
For 10-bit operation:  
 11 = Signed fractional (DOUT = sddd dddd dd00 0000, where s = .NOT.d<9>)  
 10 = Fractional (DOUT = dddd dddd dd00 0000)  
 01 = Signed integer (DOUT = ssss sssd dddd dddd, where s = .NOT.d<9>)  
 00 = Integer (DOUT = 0000 00dd dddd dddd)  
For 12-bit operation:  
 11 = Signed fractional (DOUT = sddd dddd dddd 0000, where s = .NOT.d<11>)  
 10 = Fractional (DOUT = dddd dddd dddd 0000)  
 01 = Signed Integer (DOUT = ssss sddd dddd dddd, where s = .NOT.d<11>)  
 00 = Integer (DOUT = 0000 dddd dddd dddd)
- bit 7-5     **SSRC<2:0>:** Sample Clock Source Select bits  
 111 = Internal counter ends sampling and starts conversion (auto-convert)  
 110 = Reserved  
 101 = Reserved  
 100 = Reserved  
 011 = Reserved  
 010 = GP timer (Timer3 for ADC1, Timer5 for ADC2) compare ends sampling and starts conversion  
 001 = Active transition on INTx pin ends sampling and starts conversion  
 000 = Clearing sample bit ends sampling and starts conversion
- bit 4      **Unimplemented:** Read as '0'

# PIC24H

---

**REGISTER 19-1: ADxCON1: ADCx CONTROL REGISTER 1 (CONTINUED)(where x = 1 or 2)**

- bit 3      **SIMSAM:** Simultaneous Sample Select bit (only applicable when CHPS<1:0> = 01 or 1x)  
**When AD12B = 1, SIMSAM is: U-0, Unimplemented, Read as '0'**  
1 = Samples CH0, CH1, CH2, CH3 simultaneously (when CHPS<1:0> = 1x); or  
    Samples CH0 and CH1 simultaneously (when CHPS<1:0> = 01)  
0 = Samples multiple channels individually in sequence
- bit 2      **ASAM:** A/D Sample Auto-Start bit  
1 = Sampling begins immediately after last conversion. 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 amplifiers are sampling  
0 = A/D sample/hold amplifiers are holding  
If ASAM = 0, software may write '1' to begin sampling. Automatically set by hardware if ASAM = 1.  
If SSRC = 000, software may write '0' to end sampling and start conversion. If SSRC ≠ 000,  
automatically cleared by hardware to end sampling and start conversion.
- bit 0      **DONE:** A/D Conversion Status bit  
1 = A/D conversion cycle is completed.  
0 = A/D conversion not started or in progress  
Automatically set by hardware when A/D conversion is complete. Software may write '0' to clear  
DONE status (software not allowed to write '1'). Clearing this bit will NOT affect any operation in  
progress. Automatically cleared by hardware at start of a new conversion.

## REGISTER 19-2: ADxCON2: ADCx CONTROL REGISTER 2 (where x = 1 or 2)

R/W-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	
VCFG<2:0>			—	—	CSCNA	CHPS<1:0>		
bit 15								bit 8

R-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	
BUFS	—	SMPI<3:0>				BUFM	ALTS	
bit 7								bit 0

### Legend:

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

bit 15-13      **VCFG<2:0>**: Converter Voltage Reference Configuration bits

	ADREF+	ADREF-
000	AVDD	AVss
001	External VREF+	AVss
010	AVDD	External VREF-
011	External VREF+	External VREF-
1xx	AVDD	AVss

bit 12-11      **Unimplemented**: Read as '0'

bit 10      **CSCNA**: Scan Input Selections for CH0+ during Sample A bit

1 = Scan inputs  
 0 = Do not scan inputs

bit 9-8      **CHPS<1:0>**: Selects Channels Utilized bits

**When AD12B = 1, CHPS<1:0> is: U-0, Unimplemented, Read as '0'**

1x = Converts CH0, CH1, CH2 and CH3  
 01 = Converts CH0 and CH1  
 00 = Converts CH0

bit 7      **BUFS**: Buffer Fill Status bit (only valid when BUFM = 1)

1 = A/D is currently filling buffer 0x8-0xF, user should access data in 0x0-0x7  
 0 = A/D is currently filling buffer 0x0-0x7, user should access data in 0x8-0xF

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

bit 5-2      **SMPI<3:0>**: Selects Increment Rate for DMA Addresses bits

1111 = Increments the DMA address after completion of every 16th sample/conversion operation  
 1110 = Increments the DMA address after completion of every 15th sample/conversion operation  
 •••

0001 = Increments the DMA address after completion of every 2nd sample/conversion operation  
 0000 = Increments the DMA address after completion of every sample/conversion operation

bit 1      **BUFM**: Buffer Fill Mode Select bit

1 = Starts buffer filling at address 0x0 on first interrupt and 0x8 on next interrupt  
 0 = Always starts filling buffer at address 0x0

bit 0      **ALTS**: Alternate Input Sample Mode Select bit

1 = Uses channel input selects for Sample A on first sample and Sample B on next sample  
 0 = Always uses channel input selects for Sample A

# PIC24H

## REGISTER 19-3: ADxCON3: ADCx CONTROL REGISTER 3

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADRC	—	—	SAMC<4:0>				
bit 15							bit 8

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	ADCS<5:0>					
bit 7							bit 0

### Legend:

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

bit 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                      **SAMC<4:0>:** Auto Sample Time bits  
                                  11111 = 31 TAD  
                                  •••  
                                  00001 = 1 TAD  
                                  00000 = 0 TAD

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

bit 5-0                      **ADCS<5:0>:** A/D Conversion Clock Select bits  
                                  111111 =  $T_{CY} \cdot (ADCS<7:0> + 1) = 64 \cdot T_{CY} = TAD$   
                                  •••  
                                  000010 =  $T_{CY} \cdot (ADCS<7:0> + 1) = 3 \cdot T_{CY} = TAD$   
                                  000001 =  $T_{CY} \cdot (ADCS<7:0> + 1) = 2 \cdot T_{CY} = TAD$   
                                  000000 =  $T_{CY} \cdot (ADCS<7:0> + 1) = 1 \cdot T_{CY} = TAD$



## REGISTER 19-4: ADxCON4: ADCx CONTROL REGISTER 4

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
—	—	—	—	—	DMABL<2:0>		
bit 7						bit 0	

### Legend:

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

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

bit 2-0                      **DMABL<2:0>:** Selects Number of DMA Buffer Locations per Analog Input bits

- 111 = Allocates 128 words of buffer to each analog input
- 110 = Allocates 64 words of buffer to each analog input
- 101 = Allocates 32 words of buffer to each analog input
- 100 = Allocates 16 words of buffer to each analog input
- 011 = Allocates 8 words of buffer to each analog input
- 010 = Allocates 4 words of buffer to each analog input
- 001 = Allocates 2 words of buffer to each analog input
- 000 = Allocates 1 word of buffer to each analog input

# PIC24H

**REGISTER 19-5: ADxCHS123: ADCx INPUT CHANNEL 1, 2, 3 SELECT REGISTER**

U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
—	—	—	—	—	CH123NB<1:0>		CH123SB
bit 15							bit 8

U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
—	—	—	—	—	CH123NA<1:0>		CH123SA
bit 7							bit 0

**Legend:**

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

- bit 15-11      **Unimplemented:** Read as '0'
- bit 10-9      **CH123NB<1:0>:** Channel 1, 2, 3 Negative Input Select for Sample B bits  
**When AD12B = 1, CHxNB is: U-0, Unimplemented, Read as '0'**  
 11 = CH1 negative input is AN9, CH2 negative input is AN10, CH3 negative input is AN11  
 10 = CH1 negative input is AN6, CH2 negative input is AN7, CH3 negative input is AN8  
 0x = CH1, CH2, CH3 negative input is VREF-
- bit 8          **CH123SB:** Channel 1, 2, 3 Positive Input Select for Sample B bit  
**When AD12B = 1, CHxSA is: U-0, Unimplemented, Read as '0'**  
 1 = CH1 positive input is AN3, CH2 positive input is AN4, CH3 positive input is AN5  
 0 = CH1 positive input is AN0, CH2 positive input is AN1, CH3 positive input is AN2
- bit 7-3      **Unimplemented:** Read as '0'
- bit 2-1      **CH123NA<1:0>:** Channel 1, 2, 3 Negative Input Select for Sample A bits  
**When AD12B = 1, CHxNA is: U-0, Unimplemented, Read as '0'**  
 11 = CH1 negative input is AN9, CH2 negative input is AN10, CH3 negative input is AN11  
 10 = CH1 negative input is AN6, CH2 negative input is AN7, CH3 negative input is AN8  
 0x = CH1, CH2, CH3 negative input is VREF-
- bit 0         **CH123SA:** Channel 1, 2, 3 Positive Input Select for Sample A bit  
**When AD12B = 1, CHxSA is: U-0, Unimplemented, Read as '0'**  
 1 = CH1 positive input is AN3, CH2 positive input is AN4, CH3 positive input is AN5  
 0 = CH1 positive input is AN0, CH2 positive input is AN1, CH3 positive input is AN2

## REGISTER 19-6: ADxCHS0: ADCx INPUT CHANNEL 0 SELECT REGISTER

R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CH0NB	—	—	CH0SB<4:0>				
bit 15							bit 8

R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CH0NA	—	—	CH0SA<4:0>				
bit 7							bit 0

### Legend:

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

- bit 15      **CH0NB:** Channel 0 Negative Input Select for Sample B bit  
Same definition as bit 7.
- bit 14-13    **Unimplemented:** Read as '0'
- bit 12-8    **CH0SB<4:0>:** Channel 0 Positive Input Select for Sample B bits  
Same definition as bit<4:0>.
- bit 7        **CH0NA:** Channel 0 Negative Input Select for Sample A bit  
1 = Channel 0 negative input is AN1  
0 = Channel 0 negative input is VREF-
- bit 6-5     **Unimplemented:** Read as '0'
- bit 4-0     **CH0SA<4:0>:** Channel 0 Positive Input Select for Sample A bits  
11111 = Channel 0 positive input is AN31  
11110 = Channel 0 positive input is AN30  
•••  
00010 = Channel 0 positive input is AN2  
00001 = Channel 0 positive input is AN1  
00000 = Channel 0 positive input is AN0

# PIC24H

## REGISTER 19-7: ADxCSSH: ADCx INPUT SCAN SELECT REGISTER HIGH<sup>(1)</sup>

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CSS31	CSS30	CSS29	CSS28	CSS27	CSS26	CSS25	CSS24
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CSS23	CSS22	CSS21	CSS20	CSS19	CSS18	CSS17	CSS16
bit 7							bit 0

### Legend:

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

bit 15-0                      **CSS<31:16>**: A/D Input Scan Selection bits

- 1 = Select ANx for input scan
- 0 = Skip ANx for input scan

**Note 1:** On devices without 32 analog inputs, all ADxCSSL bits may be selected by user. However, inputs selected for scan without a corresponding input on device will convert ADREF-.

## REGISTER 19-8: ADxCSSL: ADCx INPUT SCAN SELECT REGISTER LOW<sup>(1)</sup>

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CSS15	CSS14	CSS13	CSS12	CSS11	CSS10	CSS9	CSS8
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CSS7	CSS6	CSS5	CSS4	CSS3	CSS2	CSS1	CSS0
bit 7							bit 0

### Legend:

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

bit 15-0                      **CSS<15:0>**: A/D Input Scan Selection bits

- 1 = Select ANx for input scan
- 0 = Skip ANx for input scan

**Note 1:** On devices without 16 analog inputs, all ADxCSSL bits may be selected by user. However, inputs selected for scan without a corresponding input on device will convert ADREF-.

## REGISTER 19-9: AD1PCFGH: ADC1 PORT CONFIGURATION REGISTER HIGH<sup>(1,2)</sup>

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PCFG31	PCFG30	PCFG29	PCFG28	PCFG27	PCFG26	PCFG25	PCFG24
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PCFG23	PCFG22	PCFG21	PCFG20	PCFG19	PCFG18	PCFG17	PCFG16
bit 7							bit 0

### Legend:

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

bit 15-0                      **PCFG<31:16>**: A/D Port Configuration Control bits

- 1 = Port pin in Digital mode, port read input enabled, A/D input multiplexor connected to AVSS
- 0 = Port pin in Analog mode, port read input disabled, A/D samples pin voltage

- Note 1:** On devices without 32 analog inputs, all PCFG bits are R/W by user. However, PCFG bits are ignored on ports without a corresponding input on device.
- 2:** ADC2 only supports analog inputs AN0-AN15; therefore, no ADC2 high port Configuration register exists.

## REGISTER 19-10: ADxPCFGL: ADCx PORT CONFIGURATION REGISTER LOW<sup>(1,2)</sup>

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

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

### Legend:

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

bit 15-0                      **PCFG<15:0>**: A/D Port Configuration Control bits

- 1 = Port pin in Digital mode, port read input enabled, A/D input multiplexor connected to AVSS
- 0 = Port pin in Analog mode, port read input disabled, A/D samples pin voltage

- Note 1:** On devices without 16 analog inputs, all PCFG bits are R/W by user. However, PCFG bits are ignored on ports without a corresponding input on device.
- 2:** On devices with 2 analog-to-digital modules, both AD1PCFGL and AD2PCFGL will affect the configuration of port pins multiplexed with AN0-AN15.

# PIC24H

---

NOTES:

## 20.0 SPECIAL FEATURES

**Note:** This data sheet summarizes the features of this group of PIC24H devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the “*dsPIC30F Family Reference Manual*” (DS70046).

PIC24H 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™ (ICSP™) programming capability
- In-Circuit Emulation

### 20.1 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 0xF80000.

The device Configuration register map is shown in Table 20-1.

The individual Configuration bit descriptions for the RESERVED1, RESERVED2, FGS, FOSCSEL, FOSC, FWDT, FPOR and RESERVED3 Configuration registers are shown in Table 20-2.

Note that address F8000h is beyond the user program memory space. In fact, it belongs to the configuration memory space (0x800000-0xFFFFF), which can only be accessed using table reads and table writes.

The upper byte of all device Configuration registers 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.

**TABLE 20-1: DEVICE CONFIGURATION REGISTER MAP**

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
0xF8000	RESERVED1	Reserved <sup>(1)</sup>								
0xF8002	RESERVED2	Reserved <sup>(1)</sup>								
0xF8004	FGS	—					Reserved <sup>(2)</sup>	GCP	GWRP	
0xF8006	FOSCSEL	IESO	—	TEMP	—		FNOSC<2:0>			
0xF8008	FOSC	FCKSM<1:0>					OSCI0FNC	POSCMD<1:0>		
0xF800A	FWDT	FWDTEN	WINDIS	Reserved	WDTPRE	WDTPOST<3:0>				
0xF800C	FPOR	Reserved <sup>(1)</sup>			—		FPWRT<2:0>			
0xF800E	RESERVED3	Reserved								
0xF8010	FUID0	User Unit ID Byte 0								
0xF8012	FUID1	User Unit ID Byte 1								
0xF8014	FUID2	User Unit ID Byte 2								
0xF8016	FUID3	User Unit ID Byte 3								

**Note 1:** Reserved bits are read as ‘1’ and must be programmed as ‘1’.

**2:** This reserved bit is a read-only copy of the GCP bit.

**3:** Unimplemented bits are read as ‘0’.

# PIC24H

**TABLE 20-2: PIC24H CONFIGURATION BITS DESCRIPTION**

Bit Field	Register	Description
GCP	FGS	General Segment Code-Protect bit 1 = User program memory is not code-protected 0 = User program memory is code-protected
GWRP	FGS	General Segment Write-Protect bit 1 = User program memory is not write-protected 0 = User program memory is write-protected
IESO	FOSCSEL	Internal External Start-up Option bit 1 = Start-up device with FRC, then automatically switch to the user-selected oscillator source when ready 0 = Start-up device with user-selected oscillator source
TEMP	FOSCSEL	Temperature Protection Enable bit 1 = Temperature protection disabled 0 = Temperature protection enabled
FNOSC<2:0>	FOSCSEL	Initial Oscillator Source Selection bits 111 = Internal Fast RC (FRC) oscillator with postscaler 110 = Reserved 101 = LPRC oscillator 100 = Secondary (LP) oscillator 011 = Primary (XT, HS, EC) oscillator with PLL 010 = Primary (XT, HS, EC) oscillator 001 = Internal Fast RC (FRC) oscillator with PLL 000 = FRC oscillator
FCKSM<1:0>	FOSC	Clock Switching Mode bits 1x = Clock switching is disabled, Fail-Safe Clock Monitor is disabled 01 = Clock switching is enabled, Fail-Safe Clock Monitor is disabled 00 = Clock switching is enabled, Fail-Safe Clock Monitor is enabled
OSCIOFNC	FOSC	OSC2 Pin Function bit (except in XT and HS modes) 1 = OSC2 is clock output 0 = OSC2 is general purpose digital I/O pin
POSCMD<1:0>	FOSC	Primary Oscillator Mode Select bits 11 = Primary oscillator disabled 10 = HS Crystal Oscillator mode 01 = XT Crystal Oscillator mode 00 = EC (External Clock) mode
FWDTEN	FWDT	Watchdog Timer Enable bit 1 = Watchdog Timer always enabled (LPRC oscillator cannot be disabled. Clearing the SWDTEN bit in the RCON register will have no effect.) 0 = Watchdog Timer enabled/disabled by user software (LPRC can be disabled by clearing the SWDTEN bit in the RCON register)
WINDIS	FWDT	Watchdog Timer Window Enable bit 1 = Watchdog Timer in Non-Window mode 0 = Watchdog Timer in Window mode
WDTPRE	FWDT	Watchdog Timer Prescaler bit 1 = 1:128 0 = 1:32
WDTPOST	FWDT	Watchdog Timer Postscaler bits 1111 = 1:32,768 1110 = 1:16,384 . . . 0001 = 1:2 0000 = 1:1



**TABLE 20-2: PIC24H CONFIGURATION BITS DESCRIPTION (CONTINUED)**

Bit Field	Register	Description
FPWRT<2:0>	FPOR	Power-on Reset Timer Value Select bits 111 = PWRT = 128 ms 110 = PWRT = 64 ms 101 = PWRT = 32 ms 100 = PWRT = 16 ms 011 = PWRT = 8 ms 010 = PWRT = 4 ms 001 = PWRT = 2 ms 000 = PWRT = Disabled
Reserved	RESERVED1, RESERVED2, FPOR, RESERVED3	Reserved (read as '1' and must be programmed as '1')
—	FGS, FOSCSEL, FOSC, FWDT, FPOR	Unimplemented (read as '0', write as '0')

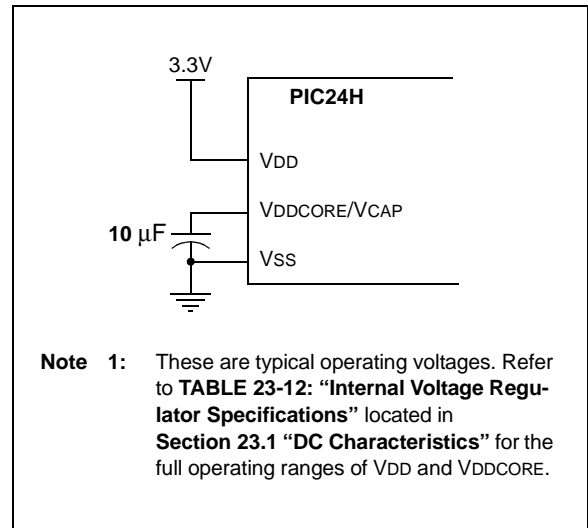
## 20.2 On-Chip Voltage Regulator

All of the PIC24H 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 PIC24H family incorporate an on-chip regulator that allows the device to run its core logic from VDD.

The regulator provides power to the core from the other VDD pins. When the regulator is enabled, a low-ESR (less than 5 ohms) capacitor (such as tantalum or ceramic) must be connected to the VDDCORE/VCAP pin (Figure 20-1). This helps to maintain the stability of the regulator. The recommended value for the filter capacitor is provided in **TABLE 23-12: “Internal Voltage Regulator Specifications”** located in **Section 23.1 “DC Characteristics”**.

On a POR, it takes approximately 20 μs for the on-chip voltage regulator to generate an output voltage. During this time, designated as T<sub>STARTUP</sub>, code execution is disabled. T<sub>STARTUP</sub> is applied every time the device resumes operation after any power-down.

**FIGURE 20-1: CONNECTIONS FOR THE ON-CHIP VOLTAGE REGULATOR<sup>(1)</sup>**



# PIC24H

## 20.3 Watchdog Timer (WDT)

For PIC24H 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 WDTPRE 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 WDTPOST<3:0> Configuration bits (FWDT<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.

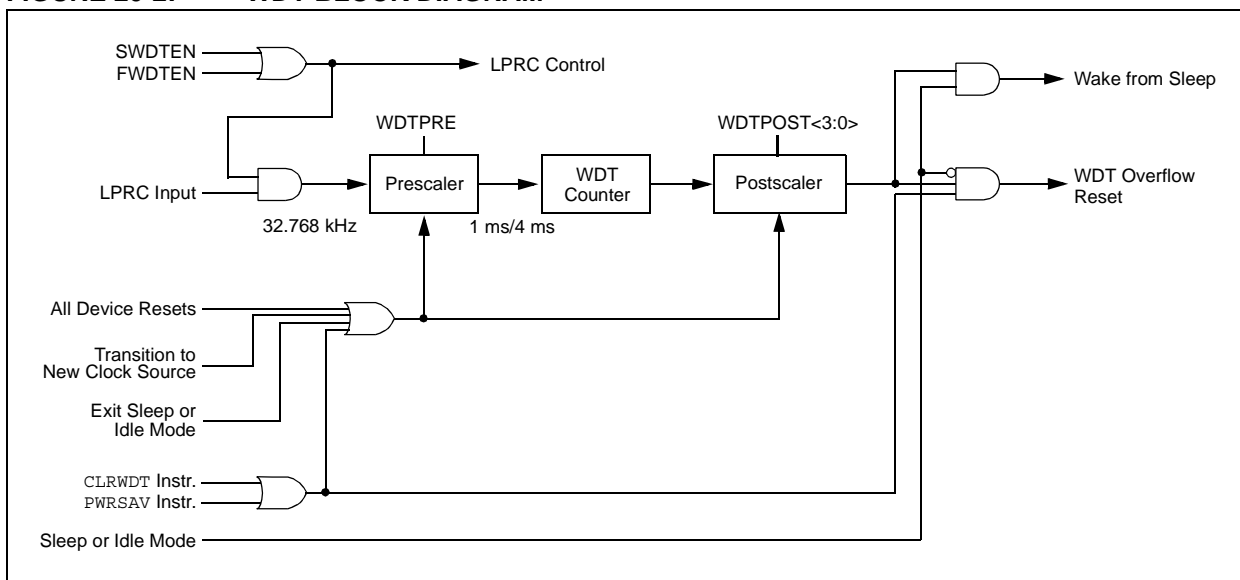
**Note:** The CLRWDT and PWRSAV instructions clear the prescaler and postscaler counts when executed.

The WDT is enabled or disabled by the FWDTEN Configuration bit in the FWDT Configuration register. 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.

**Note:** If the WINDIS bit (FWDT<6>) is cleared, the CLRWDT instruction should be executed by the application software only during the last 1/4 of the WDT period. This CLRWDT window can be determined by using a timer. If a CLRWDT instruction is executed before this window, a WDT Reset occurs.

FIGURE 20-2: WDT BLOCK DIAGRAM



## 20.4 JTAG Interface

PIC24H devices implement a JTAG interface, which supports boundary scan device testing, as well as in-circuit programming. Detailed information on the interface will be provided in future revisions of the document.

## 20.5 Code Protection

For all devices in the PIC24H family of devices, the on-chip program memory space is treated as a single block. Code protection for this block is controlled by a pair of Configuration bits, GCP and GWRP. These bits inhibit program memory reads and writes, respectively.

## 20.6 In-Circuit Serial Programming Programming Capability

PIC24H family digital signal controllers can be serially programmed while in the end application circuit. This is simply done with two lines for clock and data and three other lines for power, ground and the programming sequence. This allows customers to manufacture boards with unprogrammed devices and then program the digital signal controller just before shipping the product. This also allows the most recent firmware or a custom firmware, to be programmed. Please refer to the “*dsPIC33F Flash Programming Specification*” (DS70152) document for details about ICSP programming capability.

Any 1 out of 3 pairs of programming clock/data pins may be used:

- PGC1/EMUC1 and PGD1/EMUD1
- PGC2/EMUC2 and PGD2/EMUD2
- PGC3/EMUC3 and PGD3/EMUD3

## 20.7 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) pin functions.

Any 1 out of 3 pairs of debugging clock/data pins may be used:

- PGC1/EMUC1 and PGD1/EMUD1
- PGC2/EMUC2 and PGD2/EMUD2
- PGC3/EMUC3 and PGD3/EMUD3

To use the in-circuit debugger function of the device, the design must implement ICSP programming capability connections to  $\overline{MCLR}$ ,  $V_{DD}$ ,  $V_{SS}$ , PGC, PGD 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.

# PIC24H

---

NOTES:

## 21.0 INSTRUCTION SET SUMMARY

**Note:** This data sheet summarizes the features of this group of PIC24H devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the “*dsPIC30F Family Reference Manual*” (DS70046).

The PIC24H instruction set is identical to that of the PIC24F, and is a subset of the dsPIC30F/33F instruction set.

Most instructions are a single program memory word (24 bits). 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 five basic categories:

- Word or byte-oriented operations
- Bit-oriented operations
- Literal operations
- DSP operations
- Control operations

Table 21-1 shows the general symbols used in describing the instructions.

The PIC24H instruction set summary in Table 21-2 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

# PIC24H

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 double word instruction. Moreover, double word moves require two cycles. The double word instructions execute in two instruction cycles.

**Note:** For more details on the instruction set, refer to the “dsPIC30F/33F Programmer’s Reference Manual” (DS70157).

**TABLE 21-1: 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) $\in \{0..15\}$
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 $\in \{0x0000..0x1FFF\}$
lit1	1-bit unsigned literal $\in \{0,1\}$
lit4	4-bit unsigned literal $\in \{0..15\}$
lit5	5-bit unsigned literal $\in \{0..31\}$
lit8	8-bit unsigned literal $\in \{0..255\}$
lit10	10-bit unsigned literal $\in \{0..255\}$ for Byte mode, $\{0:1023\}$ for Word mode
lit14	14-bit unsigned literal $\in \{0..16384\}$
lit16	16-bit unsigned literal $\in \{0..65535\}$
lit23	23-bit unsigned literal $\in \{0..8388608\}$ ; LSB must be ‘0’
None	Field does not require an entry, may be blank
PC	Program Counter
Slit10	10-bit signed literal $\in \{-512..511\}$
Slit16	16-bit signed literal $\in \{-32768..32767\}$
Slit6	6-bit signed literal $\in \{-16..16\}$
Wb	Base W register $\in \{W0..W15\}$
Wd	Destination W register $\in \{Wd, [Wd], [Wd++] , [Wd--], [++Wd], [--Wd] \}$
Wdo	Destination W register $\in \{Wnd, [Wnd], [Wnd++] , [Wnd--], [++Wnd], [--Wnd], [Wnd+Wb] \}$
Wm,Wn	Dividend, Divisor working register pair (direct addressing)
Wm*Wm	Multiplicand and Multiplier working register pair for Square instructions $\in \{W4 * W4, W5 * W5, W6 * W6, W7 * W7\}$
Wm*Wn	Multiplicand and Multiplier working register pair for DSP instructions $\in \{W4 * W5, W4 * W6, W4 * W7, W5 * W6, W5 * W7, W6 * W7\}$
Wn	One of 16 working registers $\in \{W0..W15\}$

**TABLE 21-1: SYMBOLS USED IN OPCODE DESCRIPTIONS (CONTINUED)**

Field	Description
Wnd	One of 16 destination working registers $\in \{W0..W15\}$
Wns	One of 16 source working registers $\in \{W0..W15\}$
WREG	W0 (working register used in file register instructions)
Ws	Source W register $\in \{Ws, [Ws], [Ws++] , [Ws-], [++Ws], [--Ws] \}$
Wso	Source W register $\in \{ Wns, [Wns], [Wns++] , [Wns-], [++Wns], [--Wns], [Wns+Wb] \}$

# PIC24H

**TABLE 21-2: INSTRUCTION SET OVERVIEW**

Base Instr #	Assembly Mnemonic	Assembly Syntax	Description	# of Words	# of Cycles	Status Flags Affected
1	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
2	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
3	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
4	ASR	ASR f	$f = \text{Arithmetic Right Shift } f$	1	1	C,N,OV,Z
		ASR f,WREG	$WREG = \text{Arithmetic Right Shift } f$	1	1	C,N,OV,Z
		ASR Ws,Wd	$Wd = \text{Arithmetic Right Shift } Ws$	1	1	C,N,OV,Z
		ASR Wb,Wns,Wnd	$Wnd = \text{Arithmetic Right Shift } Wb \text{ by } Wns$	1	1	N,Z
		ASR Wb,#lit5,Wnd	$Wnd = \text{Arithmetic Right Shift } Wb \text{ by } lit5$	1	1	N,Z
5	BCLR	BCLR f,#bit4	Bit Clear f	1	1	None
		BCLR Ws,#bit4	Bit Clear Ws	1	1	None
6	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
		BRA LE,Expr	Branch if less than or equal	1	1 (2)	None
		BRA LEU,Expr	Branch if unsigned less than or equal	1	1 (2)	None
		BRA LT,Expr	Branch if less than	1	1 (2)	None
		BRA LTU,Expr	Branch if unsigned less than	1	1 (2)	None
		BRA N,Expr	Branch if Negative	1	1 (2)	None
		BRA NC,Expr	Branch if Not Carry	1	1 (2)	None
		BRA NN,Expr	Branch if Not Negative	1	1 (2)	None
		BRA NZ,Expr	Branch if Not Zero	1	1 (2)	None
		BRA Expr	Branch Unconditionally	1	2	None
		BRA Z,Expr	Branch if Zero	1	1 (2)	None
BRA Wn	Computed Branch	1	2	None		
7	BSET	BSET f,#bit4	Bit Set f	1	1	None
		BSET Ws,#bit4	Bit Set Ws	1	1	None
8	BSW	BSW.C Ws,Wb	Write C bit to $Ws<Wb>$	1	1	None
		BSW.Z Ws,Wb	Write Z bit to $Ws<Wb>$	1	1	None
9	BTG	BTG f,#bit4	Bit Toggle f	1	1	None
		BTG Ws,#bit4	Bit Toggle Ws	1	1	None
10	BTSC	BTSC f,#bit4	Bit Test f, Skip if Clear	1	1 (2 or 3)	None
		BTSC Ws,#bit4	Bit Test Ws, Skip if Clear	1	1 (2 or 3)	None
11	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



**TABLE 21-2: INSTRUCTION SET OVERVIEW (CONTINUED)**

Base Instr #	Assembly Mnemonic	Assembly Syntax	Description	# of Words	# of Cycles	Status Flags Affected
12	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
13	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
14	CALL	CALL lit23	Call subroutine	2	2	None
		CALL Wn	Call indirect subroutine	1	2	None
15	CLR	CLR f	f = 0x0000	1	1	None
		CLR WREG	WREG = 0x0000	1	1	None
		CLR Ws	Ws = 0x0000	1	1	None
16	CLRWDT	CLRWDT	Clear Watchdog Timer	1	1	WDTO,Sleep
17	COM	COM f	f = $\bar{f}$	1	1	N,Z
		COM f,WREG	WREG = $\bar{f}$	1	1	N,Z
		COM Ws,Wd	Wd = $\overline{Ws}$	1	1	N,Z
18	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
19	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
20	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 – C)	1	1	C,DC,N,OV,Z
21	CPSEQ	CPSEQ Wb, Wn	Compare Wb with Wn, skip if =	1	1 (2 or 3)	None
22	CPSGT	CPSGT Wb, Wn	Compare Wb with Wn, skip if >	1	1 (2 or 3)	None
23	CPSLT	CPSLT Wb, Wn	Compare Wb with Wn, skip if <	1	1 (2 or 3)	None
24	CPSNE	CPSNE Wb, Wn	Compare Wb with Wn, skip if ≠	1	1 (2 or 3)	None
25	DAW	DAW Wn	Wn = decimal adjust Wn	1	1	C
26	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
27	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
28	DISI	DISI #lit14	Disable Interrupts for k instruction cycles	1	1	None
29	DIV	DIV.S 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.U 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
30	EXCH	EXCH Wns,Wnd	Swap Wns with Wnd	1	1	None
31	FBCL	FBCL Ws,Wnd	Find Bit Change from Left (MSb) Side	1	1	C
32	FF1L	FF1L Ws,Wnd	Find First One from Left (MSb) Side	1	1	C
33	FF1R	FF1R Ws,Wnd	Find First One from Right (LSb) Side	1	1	C
34	GOTO	GOTO Expr	Go to address	2	2	None
		GOTO Wn	Go to indirect	1	2	None

# PIC24H

**TABLE 21-2: INSTRUCTION SET OVERVIEW (CONTINUED)**

Base Instr #	Assembly Mnemonic	Assembly Syntax	Description	# of Words	# of Cycles	Status Flags Affected
35	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
36	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
37	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
38	LNK	LNK #lit14	Link Frame Pointer	1	1	None
39	LSR	LSR f	$f = \text{Logical Right Shift } f$	1	1	C,N,OV,Z
		LSR f,WREG	$WREG = \text{Logical Right Shift } f$	1	1	C,N,OV,Z
		LSR Ws,Wd	$Wd = \text{Logical Right Shift } Ws$	1	1	C,N,OV,Z
		LSR Wb,Wns,Wnd	$Wnd = \text{Logical Right Shift } Wb \text{ by } Wns$	1	1	N,Z
		LSR Wb,#lit5,Wnd	$Wnd = \text{Logical Right Shift } Wb \text{ by } lit5$	1	1	N,Z
40	MOV	MOV f,Wn	Move f to Wn	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 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
41	MUL	MUL.SS Wb,Ws,Wnd	$\{Wnd + 1, Wnd\} = \text{signed}(Wb) * \text{signed}(Ws)$	1	1	None
		MUL.SU Wb,Ws,Wnd	$\{Wnd + 1, Wnd\} = \text{signed}(Wb) * \text{unsigned}(Ws)$	1	1	None
		MUL.US Wb,Ws,Wnd	$\{Wnd + 1, Wnd\} = \text{unsigned}(Wb) * \text{signed}(Ws)$	1	1	None
		MUL.UU Wb,Ws,Wnd	$\{Wnd + 1, Wnd\} = \text{unsigned}(Wb) * \text{unsigned}(Ws)$	1	1	None
		MUL.SU Wb,#lit5,Wnd	$\{Wnd + 1, Wnd\} = \text{signed}(Wb) * \text{unsigned}(lit5)$	1	1	None
		MUL.UU Wb,#lit5,Wnd	$\{Wnd + 1, Wnd\} = \text{unsigned}(Wb) * \text{unsigned}(lit5)$	1	1	None
		MUL f	$W3:W2 = f * WREG$	1	1	None
42	NEG	NEG f	$f = \bar{f} + 1$	1	1	C,DC,N,OV,Z
		NEG f,WREG	$WREG = \bar{f} + 1$	1	1	C,DC,N,OV,Z
		NEG Ws,Wd	$Wd = \bar{Ws} + 1$	1	1	C,DC,N,OV,Z
43	NOP	NOP	No Operation	1	1	None
		NOPR	No Operation	1	1	None
44	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
45	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
46	PWRSVAV	PWRSVAV #lit1	Go into Sleep or Idle mode	1	1	WDTO,Sleep

**TABLE 21-2: INSTRUCTION SET OVERVIEW (CONTINUED)**

Base Instr #	Assembly Mnemonic	Assembly Syntax	Description	# of Words	# of Cycles	Status Flags Affected
47	RCALL	RCALL Expr	Relative Call	1	2	None
		RCALL Wn	Computed Call	1	2	None
48	REPEAT	REPEAT #lit14	Repeat Next Instruction lit14 + 1 times	1	1	None
		REPEAT Wn	Repeat Next Instruction (Wn) + 1 times	1	1	None
49	RESET	RESET	Software device Reset	1	1	None
50	RETFIE	RETFIE	Return from interrupt	1	3 (2)	None
51	RETLW	RETLW #lit10,Wn	Return with literal in Wn	1	3 (2)	None
52	RETURN	RETURN	Return from Subroutine	1	3 (2)	None
53	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
54	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
55	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
56	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
57	SE	SE Ws,Wnd	Wnd = sign-extended Ws	1	1	C,N,Z
58	SETM	SETM f	f = 0xFFFF	1	1	None
		SETM WREG	WREG = 0xFFFF	1	1	None
		SETM Ws	Ws = 0xFFFF	1	1	None
59	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
60	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
61	SUBB	SUBB f	f = f - WREG - ( $\bar{C}$ )	1	1	C,DC,N,OV,Z
		SUBB f,WREG	WREG = f - WREG - ( $\bar{C}$ )	1	1	C,DC,N,OV,Z
		SUBB #lit10,Wn	Wn = Wn - lit10 - ( $\bar{C}$ )	1	1	C,DC,N,OV,Z
		SUBB Wb,Ws,Wd	Wd = Wb - Ws - ( $\bar{C}$ )	1	1	C,DC,N,OV,Z
		SUBB Wb,#lit5,Wd	Wd = Wb - lit5 - ( $\bar{C}$ )	1	1	C,DC,N,OV,Z
62	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
63	SUBBR	SUBBR f	f = WREG - f - ( $\bar{C}$ )	1	1	C,DC,N,OV,Z
		SUBBR f,WREG	WREG = WREG - f - ( $\bar{C}$ )	1	1	C,DC,N,OV,Z
		SUBBR Wb,Ws,Wd	Wd = Ws - Wb - ( $\bar{C}$ )	1	1	C,DC,N,OV,Z
		SUBBR Wb,#lit5,Wd	Wd = lit5 - Wb - ( $\bar{C}$ )	1	1	C,DC,N,OV,Z
64	SWAP	SWAP.b Wn	Wn = nibble swap Wn	1	1	None
		SWAP Wn	Wn = byte swap Wn	1	1	None
65	TBLRDH	TBLRDH Ws,Wd	Read Prog<23:16> to Wd<7:0>	1	2	None

# PIC24H

**TABLE 21-2: INSTRUCTION SET OVERVIEW (CONTINUED)**

Base Instr #	Assembly Mnemonic	Assembly Syntax	Description	# of Words	# of Cycles	Status Flags Affected
66	TBLRD	TBLRDL Ws,Wd	Read Prog<15:0> to Wd	1	2	None
67	TBLWTH	TBLWTH Ws,Wd	Write Ws<7:0> to Prog<23:16>	1	2	None
68	TBLWTL	TBLWTL Ws,Wd	Write Ws to Prog<15:0>	1	2	None
69	ULNK	ULNK	Unlink Frame Pointer	1	1	None
70	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
71	ZE	ZE Ws,Wnd	Wnd = Zero-extend Ws	1	1	C,Z,N

## 22.0 DEVELOPMENT SUPPORT

The PICmicro<sup>®</sup> microcontrollers are supported with a full range of hardware and software development tools:

- Integrated Development Environment
  - MPLAB<sup>®</sup> IDE Software
- Assemblers/Compilers/Linkers
  - MPASM<sup>™</sup> Assembler
  - MPLAB C18 and MPLAB C30 C Compilers
  - MPLINK<sup>™</sup> Object Linker/  
MPLIB<sup>™</sup> 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<sup>®</sup> Plus Development Programmer
  - MPLAB PM3 Device Programmer
  - PICKit<sup>™</sup> 2 Development Programmer
- Low-Cost Demonstration and Development Boards and Evaluation Kits

## 22.1 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<sup>®</sup> 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.

## 22.2 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

## 22.3 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 the dsPIC30, dsPIC33 and PIC24 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.

## 22.4 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/librarian features include:

- Efficient linking of single libraries instead of many smaller files
- Enhanced code maintainability by grouping related modules together
- Flexible creation of libraries with easy module listing, replacement, deletion and extraction

## 22.5 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

## 22.6 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, most peripherals and 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 hardware laboratory environment, making it an excellent, economical software development tool.

## 22.7 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.

## 22.8 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.

## 22.9 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 MCU devices.

## 22.10 MPLAB PM3 Device Programmer

The MPLAB PM3 Device Programmer is a universal, CE compliant device programmer with programmable voltage verification at VDDMIN and VDDMAX for maximum reliability. It features a large LCD display (128 x 64) for menus and error messages and a modular, detachable socket assembly to support various package types. The ICSP™ programming capability cable assembly is included as a standard item. In Stand-Alone mode, the MPLAB PM3 Device Programmer can read, verify and program PICmicro MCU 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.

## 22.11 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 MCU 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.

## 22.12 PICkit 2 Development Programmer

The PICkit™ 2 Development Programmer is a low-cost programmer with an easy-to-use interface for programming many of Microchip's baseline, mid-range and PIC18F families of Flash memory microcontrollers. The PICkit 2 Starter Kit includes a prototyping development board, twelve sequential lessons, software and HI-TECH's PICC Lite C compiler, and is designed to help get up to speed quickly using PIC® microcontrollers. The kit provides everything needed to program, evaluate and develop applications using Microchip's powerful, mid-range Flash memory family of microcontrollers.

## 22.13 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](http://www.microchip.com)) and the latest *"Product Selector Guide"* (DS00148) for the complete list of demonstration, development and evaluation kits.



## 23.0 ELECTRICAL CHARACTERISTICS

This section provides an overview of PIC24H electrical characteristics. Additional information will be provided in future revisions of this document as it becomes available.

Absolute maximum ratings for the PIC24H 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<sup>(†)</sup>

Ambient temperature under bias .....	-40°C to +85°C
Storage temperature .....	-65°C to +150°C
Voltage on VDD with respect to VSS .....	-0.3V to +4.0V
Voltage on any combined analog and digital pin and $\overline{\text{MCLR}}$ , with respect to VSS .....	-0.3V to (VDD + 0.3V)
Voltage on any digital-only pin with respect to VSS .....	-0.3V to +5.6V
Voltage on VDDCORE with respect to VSS .....	2.25V to 2.75V
Maximum current out of VSS pin .....	300 mA
Maximum current into VDD pin ( <b>Note 1</b> ).....	250 mA
Maximum output current sunk by any I/O pin.....	4 mA
Maximum output current sourced by any I/O pin .....	4 mA
Maximum current sunk by all ports .....	200 mA
Maximum current sourced by all ports ( <b>Note 1</b> ).....	200 mA

**Note 1:** Maximum allowable current is a function of device maximum power dissipation (see Table 23-2).

<sup>†</sup>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.

# PIC24H

## 23.1 DC Characteristics

**TABLE 23-1: OPERATING MIPS VS. VOLTAGE**

Characteristic	VDD Range (in Volts)	Temp Range (in °C)	Max MIPS
			PIC24H
DC5	3.0-3.6V	-40°C to +85°C	40

**TABLE 23-2: THERMAL OPERATING CONDITIONS**

Rating	Symbol	Min	Typ	Max	Unit
PIC24H					
Operating Junction Temperature Range	TJ	-40	—	+125	°C
Operating Ambient Temperature Range	TA	-40	—	+85	°C
Power Dissipation: Internal chip power dissipation: $P_{INT} = V_{DD} \times (I_{DD} - \Sigma I_{OH})$ I/O Pin Power Dissipation: $I/O = \Sigma (\{V_{DD} - V_{OH}\} \times I_{OH}) + \Sigma (V_{OL} \times I_{OL})$	PD	PINT + PI/O			W
Maximum Allowed Power Dissipation	PDMAX	$(T_J - T_A) / \theta_{JA}$			W

**TABLE 23-3: THERMAL PACKAGING CHARACTERISTICS**

Characteristic	Symbol	Typ	Max	Unit	Notes
Package Thermal Resistance, 100-pin TQFP (14x14x1 mm)	$\theta_{JA}$	48.4	—	°C/W	1
Package Thermal Resistance, 100-pin TQFP (12x12x1 mm)	$\theta_{JA}$	52.3	—	°C/W	1
Package Thermal Resistance, 80-pin TQFP (12x12x1 mm)	$\theta_{JA}$	38.7	—	°C/W	1
Package Thermal Resistance, 64-pin TQFP (10x10x1 mm)	$\theta_{JA}$	38.3	—	°C/W	1

**Legend:** TBD = To Be Determined

**Note 1:** Junction to ambient thermal resistance, Theta-JA ( $\theta_{JA}$ ) numbers are achieved by package simulations.

**TABLE 23-4: DC TEMPERATURE AND VOLTAGE SPECIFICATIONS**

DC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^\circ\text{C} \leq T_A \leq +85^\circ\text{C}$ for Industrial				
Param No.	Symbol	Characteristic	Min	Typ <sup>(1)</sup>	Max	Units	Conditions
<b>Operating Voltage</b>							
DC10	<b>Supply Voltage</b>						
	VDD		3.0	—	3.6	V	
DC12	VDR	<b>RAM Data Retention Voltage<sup>(2)</sup></b>	—	2.8	—	V	
DC16	VPOR	<b>VDD Start Voltage</b> to ensure internal Power-on Reset signal	—	VSS	—	V	
DC17	SVDD	<b>VDD Rise Rate</b> 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 23-5: DC CHARACTERISTICS: OPERATING CURRENT (IDD)**

DC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial			
Parameter No.	Typical <sup>(1)</sup>	Max	Units	Conditions		
<b>Operating Current (IDD)<sup>(2)</sup></b>						
DC20	27	—	mA	+25°C	3.3V	10 MIPS
DC20a	26	—	mA	+85°C		
DC21	33	—	mA	+25°C	3.3V	16 MIPS
DC21a	32	—	mA	+85°C		
DC22	44	—	mA	+25°C	3.3V	20 MIPS
DC22a	43	—	mA	+85°C		
DC23	60	—	mA	+25°C	3.3V	30 MIPS
DC23a	58	—	mA	+85°C		
DC24	74	—	mA	+25°C	3.3V	40 MIPS
DC24a	72	—	mA	+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 VSS. MCLR = VDD, WDT and FSCM are disabled. CPU, SRAM, program memory and data memory are operational. No peripheral modules are operating; however, every peripheral is being clocked (PMD bits are all zeroed).

# PIC24H

**TABLE 23-6: DC CHARACTERISTICS: IDLE CURRENT (I<sub>IDLE</sub>)**

DC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial			
Parameter No.	Typical <sup>(1)</sup>	Max	Units	Conditions		
<b>Idle Current (I<sub>IDLE</sub>): Core OFF Clock ON Base Current<sup>(2)</sup></b>						
DC40	TBD	—	mA	+25°C	3.3V	10 MIPS
DC40a	TBD	—	mA	+85°C		
DC41	TBD	—	mA	+25°C	3.3V	16 MIPS
DC41a	TBD	—	mA	+85°C		
DC42	TBD	—	mA	+25°C	3.3V	20 MIPS
DC42a	TBD	—	mA	+85°C		
DC43	TBD	—	mA	+25°C	3.3V	30 MIPS
DC43a	TBD	—	mA	+85°C		
DC44	16.5	—	mA	+25°C	3.3V	40 MIPS
DC44a	16	—	mA	+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 I<sub>IDLE</sub> current is measured with core off, clock on and all modules turned off. Peripheral Module Disable SFR registers are zeroed. All I/O pins are configured as inputs and pulled to V<sub>SS</sub>.

**TABLE 23-7: DC CHARACTERISTICS: POWER-DOWN CURRENT (I<sub>PD</sub>)**

DC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial			
Parameter No.	Typical <sup>(1)</sup>	Max	Units	Conditions		
<b>Power-Down Current (I<sub>PD</sub>)<sup>(2)</sup></b>						
DC60	200	—	μA	+25°C	3.0V	Base Power-Down Current <sup>(3,4)</sup>
DC60a	TBD	—	μA	+85°C		
DC61	TBD	—	μA	+25°C	3.0V	Watchdog Timer Current: ΔI <sub>WDT</sub> <sup>(3)</sup>
DC61a	TBD	—	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 I<sub>PD</sub> is measured with all peripherals and clocks shut down. All I/O pins are configured as inputs and pulled to V<sub>SS</sub>. WDT, etc., are all switched off.

**3:** The Δ current is the additional current consumed when the module is enabled. This current should be added to the base I<sub>PD</sub> current.

**4:** These currents are measured on the device containing the most memory in this family.

**TABLE 23-8: DC CHARACTERISTICS:DOZE CURRENT (IDoZE)**

DC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial		
Parameter No.	Typical <sup>(1)</sup>	Max	Doze Ratio	Units	Conditions
DC70a	42	—	1:2	mA 25°C	3.3V 40 MIPS
DC70f	26	—	1:64		
DC70g	25	—	1:128		
DC71a	41	—	1:2	mA 85°C	
DC71f	25	—	1:64		
DC71g	24	—	1:128		

**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.

# PIC24H

**TABLE 23-9: DC CHARACTERISTICS: I/O PIN INPUT SPECIFICATIONS**

DC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial				
Param No.	Symbol	Characteristic	Min	Typ <sup>(1)</sup>	Max	Units	Conditions
	$V_{IL}$	<b>Input Low Voltage</b>					
DI10		I/O pins	$V_{SS}$	—	$0.2 V_{DD}$	V	
DI15		$\overline{\text{MCLR}}$	$V_{SS}$	—	$0.2 V_{DD}$	V	
DI16		OSC1 (XT mode)	$V_{SS}$	—	$0.2 V_{DD}$	V	
DI17		OSC1 (HS mode)	$V_{SS}$	—	$0.2 V_{DD}$	V	
DI18		SDAx, SCLx	$V_{SS}$	—	$0.3 V_{DD}$	V	SMBus disabled
DI19		SDAx, SCLx	$V_{SS}$	—	$0.2 V_{DD}$	V	SMBus enabled
	$V_{IH}$	<b>Input High Voltage</b>					
DI20		I/O pins: with analog functions digital-only	$0.8 V_{DD}$ $0.8 V_{DD}$	— —	$V_{DD}$ 5.5	V V	
DI25		$\overline{\text{MCLR}}$	$0.8 V_{DD}$	—	$V_{DD}$	V	
DI26		OSC1 (XT mode)	$0.7 V_{DD}$	—	$V_{DD}$	V	
DI27		OSC1 (HS mode)	$0.7 V_{DD}$	—	$V_{DD}$	V	
DI28		SDAx, SCLx	$0.7 V_{DD}$	—	$V_{DD}$	V	SMBus disabled
DI29		SDAx, SCLx	$0.8 V_{DD}$	—	$V_{DD}$	V	SMBus enabled
	ICNPU	<b>CNx Pull-up Current</b>					
DI30			50	250	400	$\mu\text{A}$	$V_{DD} = 3.3\text{V}$ , $V_{PIN} = V_{SS}$
	$I_{IL}$	<b>Input Leakage Current<sup>(2)(3)</sup></b>					
DI50		I/O ports	—	TBD	TBD	$\mu\text{A}$	$V_{SS} \leq V_{PIN} \leq V_{DD}$ , Pin at high-impedance
DI51		Analog Input Pins	—	TBD	TBD	$\mu\text{A}$	$V_{SS} \leq V_{PIN} \leq V_{DD}$ , Pin at high-impedance
DI55		$\overline{\text{MCLR}}$	—	TBD	TBD	$\mu\text{A}$	$V_{SS} \leq V_{PIN} \leq V_{DD}$
DI56		OSC1	—	TBD	TBD	$\mu\text{A}$	$V_{SS} \leq V_{PIN} \leq V_{DD}$ , 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  $\overline{\text{MCLR}}$  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 23-10: DC CHARACTERISTICS: I/O PIN OUTPUT SPECIFICATIONS**

DC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial				
Param No.	Symbol	Characteristic	Min	Typ <sup>(1)</sup>	Max	Units	Conditions
DO10 DO16	VOL	<b>Output Low Voltage</b>					
		I/O ports	—	—	0.4	V	IOL = TBD mA, VDD = 3.3V
		OSC2/CLKO	—	—	0.4	V	IOL = TBD mA, VDD = 3.3V
DO20 DO26	VOH	<b>Output High Voltage</b>					
		I/O ports	2.4	—	—	V	IOH = -3.0 mA, VDD = 3.3V
		OSC2/CLKO	2.4	—	—	V	IOH = -1.3 mA, VDD = 3.3V

**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 23-11: DC CHARACTERISTICS: PROGRAM MEMORY**

DC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial				
Param No.	Symbol	Characteristic	Min	Typ <sup>(1)</sup>	Max	Units	Conditions
		<b>Program Flash Memory</b>					
D130	EP	Cell Endurance	100	1000	—	E/W	-40°C to +85°C
D131	VPR	VDD for Read	VMIN	—	3.6	V	
D132B	VPEW	VDD for Self-Timed Write	VMIN	—	3.6	V	VMIN = Minimum operating voltage
D133A	TIW	Self-Timed Write Cycle Time	—	1.5	—	ms	Provided no other specifications are violated
D134	TRETD	Characteristic Retention	20	—	—	Year	
D135	IDDP	Supply Current during Programming	—	10	—	mA	
D136	TRW	Self-Timed Row Write Cycle Time	—	1.6	—	ms	
D137	TPE	Self-Timed Page Erase Cycle Time	—	20.5	—	ms	

**Note 1:** Data in “Typ” column is at 3.3V, 25°C unless otherwise stated.

**TABLE 23-12: INTERNAL VOLTAGE REGULATOR SPECIFICATIONS**

Operating Conditions: $-40^{\circ}\text{C} < T_A < +85^{\circ}\text{C}$ (unless otherwise stated)							
Param No.	Symbol	Characteristics	Min	Typ	Max	Units	Comments
	CEFC	External Filter Capacitor Value	1	10	—	μF	Capacitor must be low series resistance (< 5 ohms)

# PIC24H

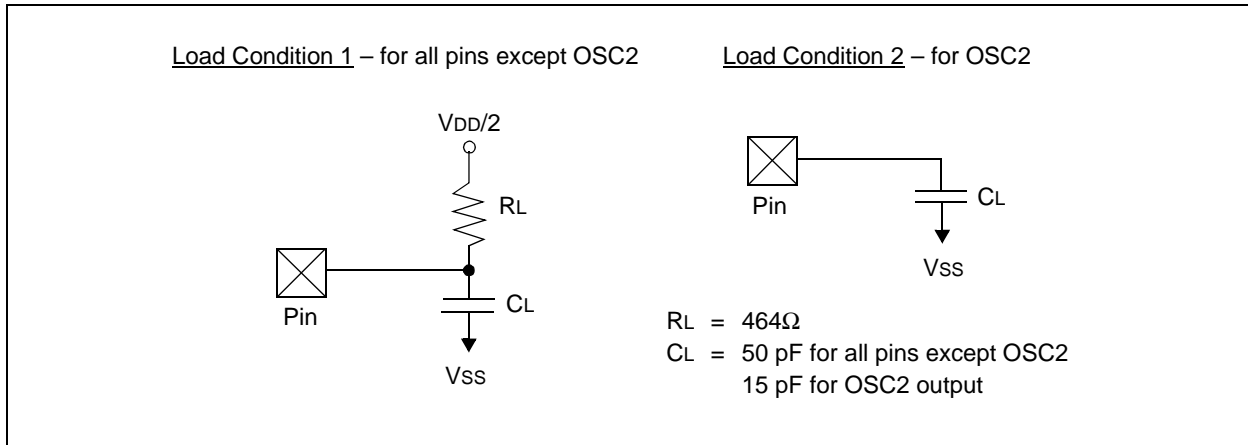
## 23.2 AC Characteristics and Timing Parameters

The information contained in this section defines PIC24H AC characteristics and timing parameters.

**TABLE 23-13: TEMPERATURE AND VOLTAGE SPECIFICATIONS – AC**

<b>AC CHARACTERISTICS</b>	<b>Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)</b> Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial Operating voltage $V_{DD}$ range as described in <b>Section 23.0 “Electrical Characteristics”</b> .
---------------------------	---

**FIGURE 23-1: LOAD CONDITIONS FOR DEVICE TIMING SPECIFICATIONS**



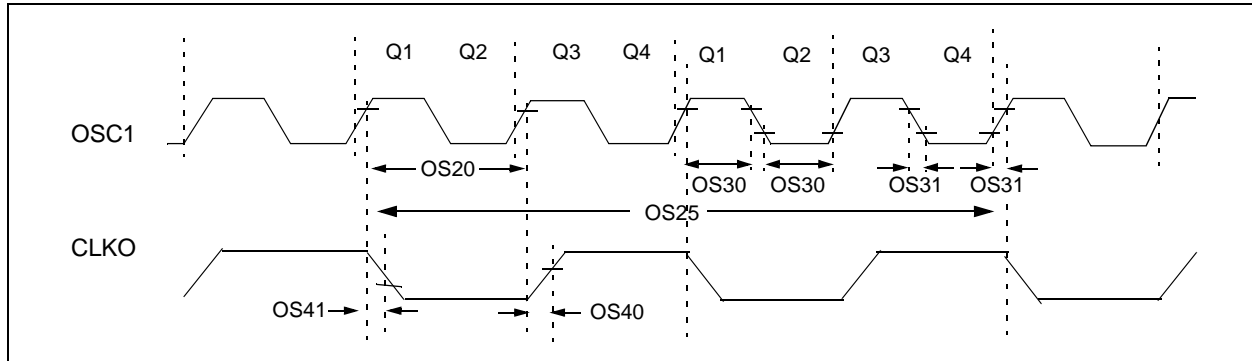
**TABLE 23-14: CAPACITIVE LOADING REQUIREMENTS ON OUTPUT PINS**

Param No.	Symbol	Characteristic	Min	Typ <sup>(1)</sup>	Max	Units	Conditions
DO50	Cosc2	OSC2/SOSC2 pin	—	—	15	pF	In XT and HS modes when external clock is used to drive OSC1
DO56	Cio	All I/O pins and OSC2	—	—	50	pF	EC mode
DO58	Cb	SCLx, SDAx	—	—	400	pF	In I <sup>2</sup> C™ mode

**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.



**FIGURE 23-2: EXTERNAL CLOCK TIMING**



**TABLE 23-15: EXTERNAL CLOCK TIMING REQUIREMENTS**

AC CHARACTERISTICS			Standard Operating Conditions: 2.5V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial				
Param No.	Symb	Characteristic	Min	Typ <sup>(1)</sup>	Max	Units	Conditions
OS10	FIN	External CLKI Frequency	0.8	—	64	MHz	EC
		(External clocks allowed only in EC and ECPLL modes)	4	—	8	MHz	ECPLL
		Oscillator Crystal Frequency	3	—	10	MHz	XT
			3	—	10	MHz	XTPLL
10	—		40	MHz	HS		
		10	—	40	MHz	HSPLL	
			—	33	kHz	SOSC	
OS20	Tosc	$T_{osc} = 1/F_{osc}$	12.5	—	DC	ns	
OS25	Tcy	Instruction Cycle Time <sup>(2)</sup>	25	—	DC	ns	
OS30	TosL, TosH	External Clock in (OSC1) High or Low Time	$0.625 \times T_{osc}$	—	—	ns	EC
OS31	TosR, TosF	External Clock in (OSC1) Rise or Fall Time	—	—	TBD	ns	EC
OS40	TckR	CLKO Rise Time <sup>(3)</sup>	—	6	TBD	ns	
OS41	TckF	CLKO Fall Time <sup>(3)</sup>	—	6	TBD	ns	

**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:** 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.

# PIC24H

**TABLE 23-16: PLL CLOCK TIMING SPECIFICATIONS (V<sub>DD</sub> = 3.0V TO 3.6V)**

AC CHARACTERISTICS		Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial					
Param No.	Symbol	Characteristic <sup>(1)</sup>	Min	Typ <sup>(2)</sup>	Max	Units	Conditions
OS50	FPLLI	PLL Voltage Controlled Oscillator (VCO) Input Frequency Range	0.8	—	8	MHz	ECPLL, HSPLL, XTPLL modes
OS51	FSYS	On-Chip VCO System Frequency	100	—	200	MHz	
OS52	TLOC	PLL Start-up Time (Lock Time)	TBD	100	TBD	μs	
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.

**Note 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 23-17: AC CHARACTERISTICS: INTERNAL RC ACCURACY**

AC CHARACTERISTICS		Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
Param No.	Characteristic	Min	Typ	Max	Units	Conditions	
<b>Internal FRC Accuracy @ 7.3728 MHz<sup>(1)</sup></b>							
F20	FRC	TBD	—	TBD	%	+25°C	V <sub>DD</sub> = 3.0-3.6V
		TBD	—	TBD	%	-40°C ≤ TA ≤ +85°C	V <sub>DD</sub> = 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.

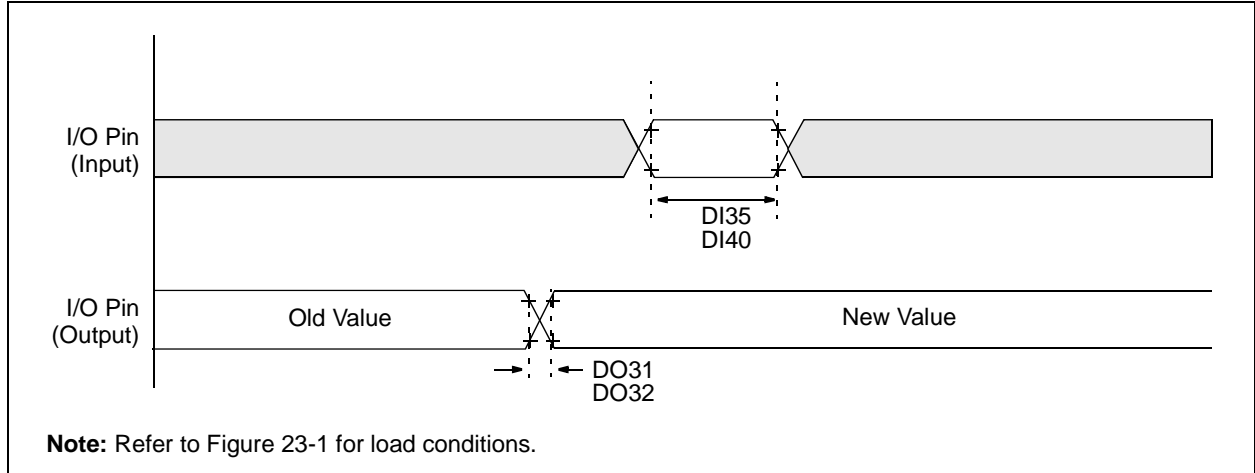
**TABLE 23-18: INTERNAL RC ACCURACY**

AC CHARACTERISTICS		Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial					
Param No.	Characteristic	Min	Typ	Max	Units	Conditions	
<b>LPRC @ 32.768 kHz<sup>(1)</sup></b>							
F21		TBD	—	TBD	%	+25°C	V <sub>DD</sub> = 3.0-3.6V
		TBD	—	TBD	%	-40°C ≤ TA ≤ +85°C	V <sub>DD</sub> = 3.0-3.6V

**Legend:** TBD = To Be Determined

**Note 1:** Change of LPRC frequency as V<sub>DD</sub> changes.

**FIGURE 23-3: CLKO AND I/O TIMING CHARACTERISTICS**



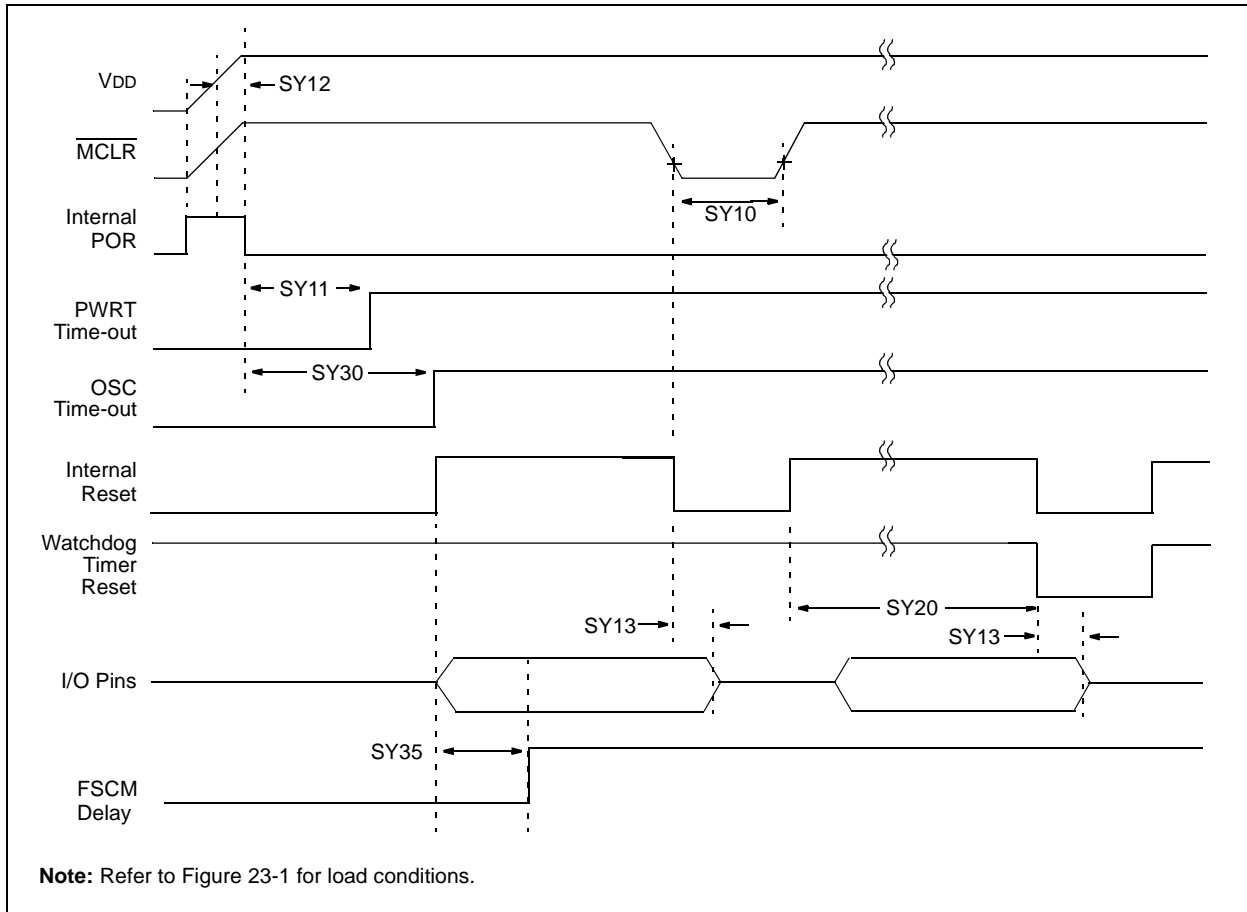
**TABLE 23-19: CLKO AND I/O TIMING REQUIREMENTS**

AC CHARACTERISTICS		Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial					
Param No.	Symbol	Characteristic	Min	Typ <sup>(1)</sup>	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.

# PIC24H

**FIGURE 23-4: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING CHARACTERISTICS**



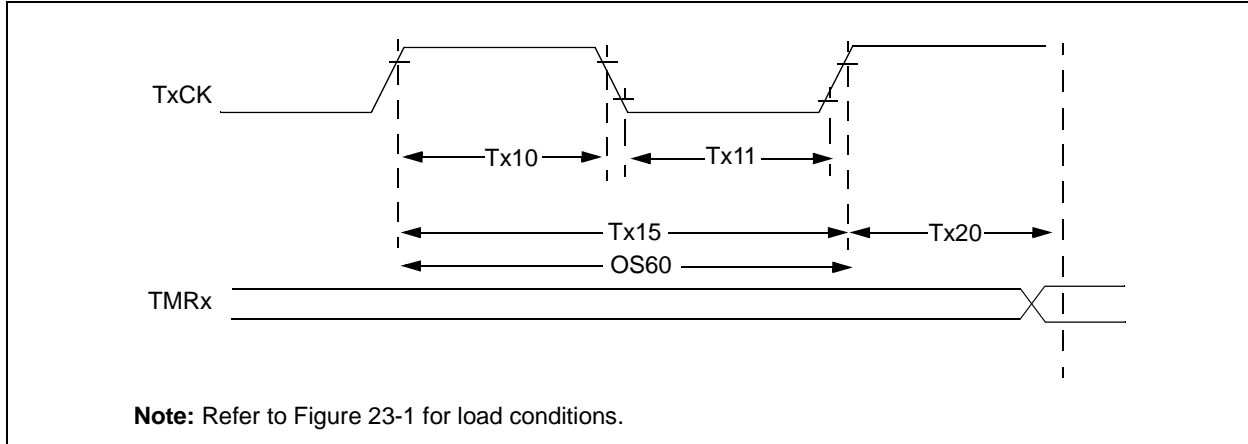
**TABLE 23-20: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER AND BROWN-OUT RESET TIMING REQUIREMENTS**

AC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$				
Param No.	Symbol	Characteristic <sup>(1)</sup>	Min	Typ <sup>(2)</sup>	Max	Units	Conditions
SY10	TMCL	MCLR Pulse Width (low)	2	—	—	μs	-40°C to +85°C
SY11	TPWRT	Power-up Timer Period	0.75 1.5 3 6 12 24 48 96	1 2 4 8 16 32 64 128	1.25 2.5 5 10 20 40 80 160	ms	-40°C to +85°C User programmable
SY12	TPOR	Power-on Reset Delay	3	10	30	μs	-40°C to +85°C
SY13	TIOZ	I/O High-Impedance from MCLR Low or Watchdog Timer Reset	—	0.8	1.0	μs	
SY20	TWDT1	Watchdog Timer Time-out Period (No Prescaler)	1.8	2.0	2.2	ms	VDD = 5V, -40°C to +85°C
	TWDT2		1.9	2.1	2.3	ms	VDD = 3V, -40°C to +85°C
SY30	TOST	Oscillator Start-up Timer Period	—	1024 TOSC	—	—	TOSC = OSC1 period
SY35	TFSCM	Fail-Safe Clock Monitor Delay	—	500	900	μs	-40°C to +85°C

- Note 1:** These parameters are characterized but not tested in manufacturing.  
**Note 2:** Data in "Typ" column is at 5V, 25°C unless otherwise stated.  
**Note 3:** Characterized by design but not tested.

# PIC24H

**FIGURE 23-5: TIMER1, 2, 3, 4, 5, 6, 7, 8 AND 9 EXTERNAL CLOCK TIMING CHARACTERISTICS**



**TABLE 23-21: TIMER1 EXTERNAL CLOCK TIMING REQUIREMENTS<sup>(1)</sup>**

AC CHARACTERISTICS		Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$						
Param No.	Symbol	Characteristic		Min	Typ	Max	Units	Conditions
TA10	T <sub>TXH</sub>	TxCK High Time	Synchronous, no prescaler	$0.5 T_{CY} + 20$	—	—	ns	Must also meet parameter TA15
			Synchronous, with prescaler	10	—	—	ns	
			Asynchronous	10	—	—	ns	
TA11	T <sub>TXL</sub>	TxCK Low Time	Synchronous, no prescaler	$0.5 T_{CY} + 20$	—	—	ns	Must also meet parameter TA15
			Synchronous, with prescaler	10	—	—	ns	
			Asynchronous	10	—	—	ns	
TA15	T <sub>TXP</sub>	TxCK Input Period	Synchronous, no prescaler	$T_{CY} + 10$	—	—	ns	N = prescale value (1, 8, 64, 256)
			Synchronous, with prescaler	Greater of: 20 ns or $(T_{CY} + 40)/N$	—	—	—	
			Asynchronous	20	—	—	ns	
OS60	F <sub>t1</sub>	SOSC1/T1CK Oscillator Input frequency Range (oscillator enabled by setting bit TCS (T1CON<1>))		DC	—	50	kHz	
TA20	T <sub>CKEXTMRL</sub>	Delay from External TxCK Clock Edge to Timer Increment		$0.5 T_{CY}$		$1.5 T_{CY}$	—	

**Note 1:** Timer1 is a Type A.

**TABLE 23-22: TIMER2, TIMER4, TIMER6 AND TIMER8 EXTERNAL CLOCK TIMING REQUIREMENTS**

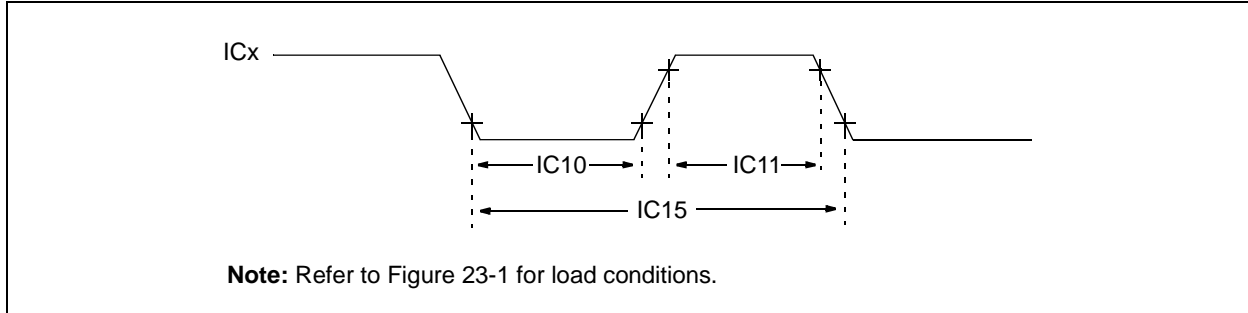
AC CHARACTERISTICS				Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$				
Param No.	Symbol	Characteristic		Min	Typ	Max	Units	Conditions
TB10	TtxH	TxCK High Time	Synchronous, no prescaler	$0.5 T_{CY} + 20$	—	—	ns	Must also meet parameter TB15
			Synchronous, with prescaler	10	—	—	ns	
TB11	TtxL	TxCK Low Time	Synchronous, no prescaler	$0.5 T_{CY} + 20$	—	—	ns	Must also meet parameter TB15
			Synchronous, with prescaler	10	—	—	ns	
TB15	TtxP	TxCK Input Period	Synchronous, no prescaler	$T_{CY} + 10$	—	—	ns	N = prescale value (1, 8, 64, 256)
			Synchronous, with prescaler	Greater of: 20 ns or $(T_{CY} + 40)/N$				
TB20	TCKEXTMRL	Delay from External TxCK Clock Edge to Timer Increment		$0.5 T_{CY}$	—	$1.5 T_{CY}$	—	

**TABLE 23-23: TIMER3, TIMER5, TIMER7 AND TIMER9 EXTERNAL CLOCK TIMING REQUIREMENTS**

AC CHARACTERISTICS				Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$				
Param No.	Symbol	Characteristic		Min	Typ	Max	Units	Conditions
TC10	TtxH	TxCK High Time	Synchronous	$0.5 T_{CY} + 20$	—	—	ns	Must also meet parameter TC15
TC11	TtxL	TxCK Low Time	Synchronous	$0.5 T_{CY} + 20$	—	—	ns	Must also meet parameter TC15
TC15	TtxP	TxCK Input Period	Synchronous, no prescaler	$T_{CY} + 10$	—	—	ns	N = prescale value (1, 8, 64, 256)
			Synchronous, with prescaler	Greater of: 20 ns or $(T_{CY} + 40)/N$				
TC20	TCKEXTMRL	Delay from External TxCK Clock Edge to Timer Increment		$0.5 T_{CY}$	—	$1.5 T_{CY}$	—	

# PIC24H

**FIGURE 23-6: INPUT CAPTURE (CAPx) TIMING CHARACTERISTICS**

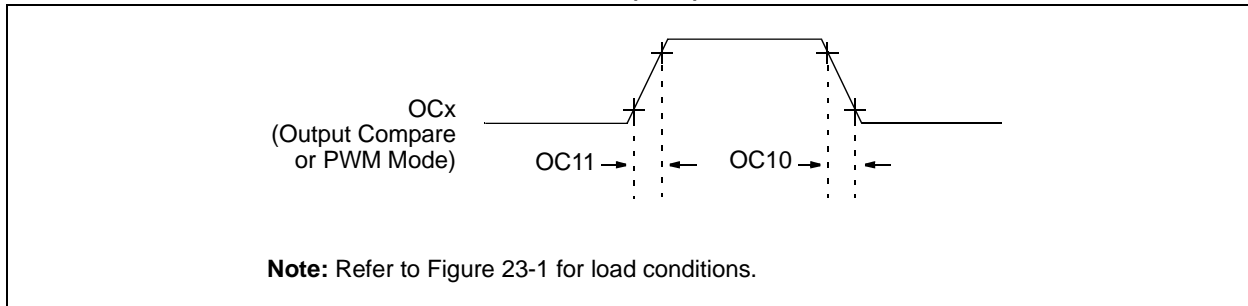


**TABLE 23-24: INPUT CAPTURE TIMING REQUIREMENTS**

AC CHARACTERISTICS		Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$				
Param No.	Symbol	Characteristic <sup>(1)</sup>	Min	Max	Units	Conditions
IC10	TccL	ICx Input Low Time	No Prescaler	$0.5 T_{CY} + 20$	—	ns
			With Prescaler	10	—	ns
IC11	TccH	ICx Input High Time	No Prescaler	$0.5 T_{CY} + 20$	—	ns
			With Prescaler	10	—	ns
IC15	TccP	ICx Input Period	$(2 T_{CY} + 40)/N$	—	ns	N = prescale value (1, 4, 16)

**Note 1:** These parameters are characterized but not tested in manufacturing.

**FIGURE 23-7: OUTPUT COMPARE MODULE (OCx) TIMING CHARACTERISTICS**



**TABLE 23-25: OUTPUT COMPARE MODULE TIMING REQUIREMENTS**

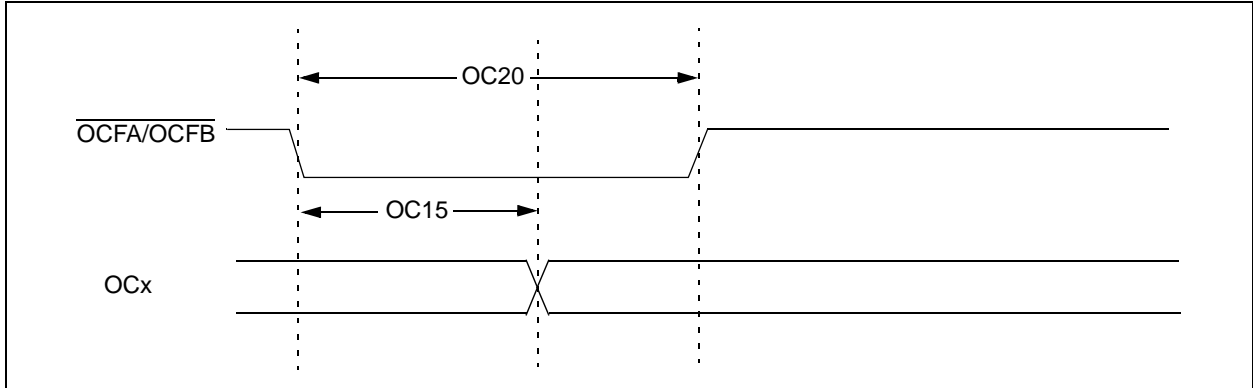
AC CHARACTERISTICS		Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$					
Param No.	Symbol	Characteristic <sup>(1)</sup>	Min	Typ <sup>(2)</sup>	Max	Units	Conditions
OC10	TccF	OCx Output Fall Time	—	—	—	ns	See parameter D032
OC11	TccR	OCx Output Rise Time	—	—	—	ns	See parameter D031

**Note 1:** These parameters are characterized but not tested in manufacturing.

**Note 2:** Data in “Typ” column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.



**FIGURE 23-8: OC/PWM MODULE TIMING CHARACTERISTICS**



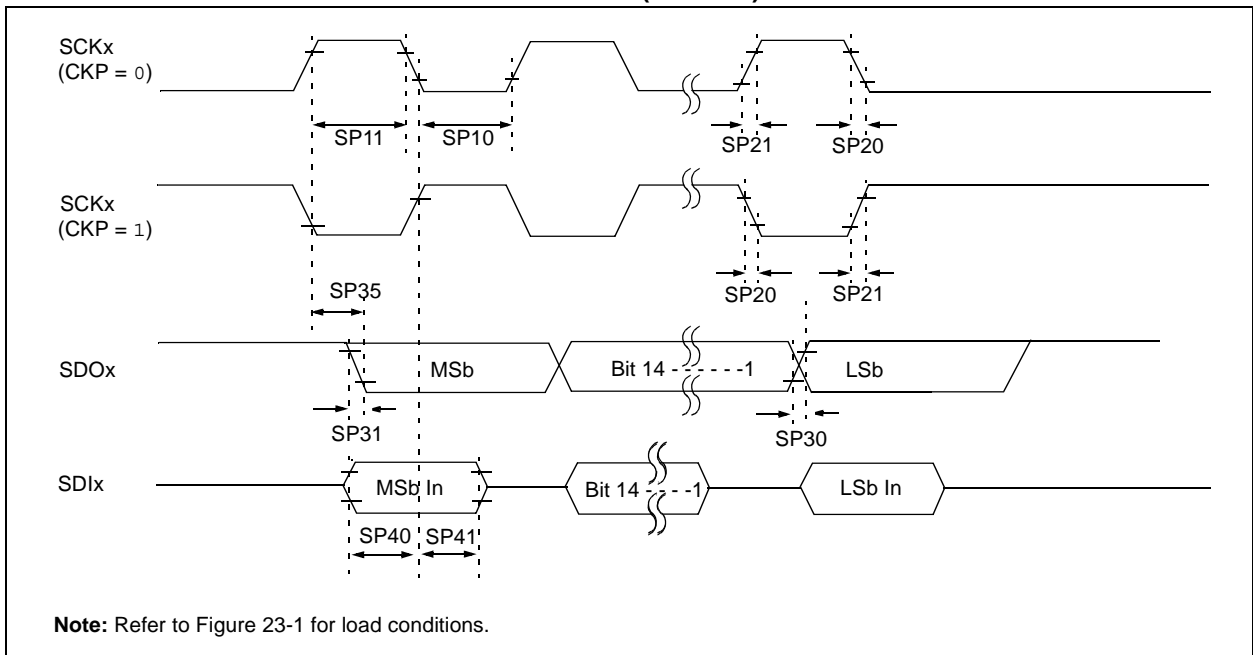
**TABLE 23-26: SIMPLE OC/PWM MODE TIMING REQUIREMENTS**

AC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$				
Param No.	Symbol	Characteristic <sup>(1)</sup>	Min	Typ <sup>(2)</sup>	Max	Units	Conditions
OC15	TFD	Fault Input to PWM I/O Change	—	—	50	ns	—
OC20	TFLT	Fault Input Pulse Width	50	—	—	ns	—

**Note 1:** These parameters are characterized but not tested in manufacturing.

**2:** Data in "Typ" column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

**FIGURE 23-9: SPIx MODULE MASTER MODE (CKE = 0) TIMING CHARACTERISTICS**



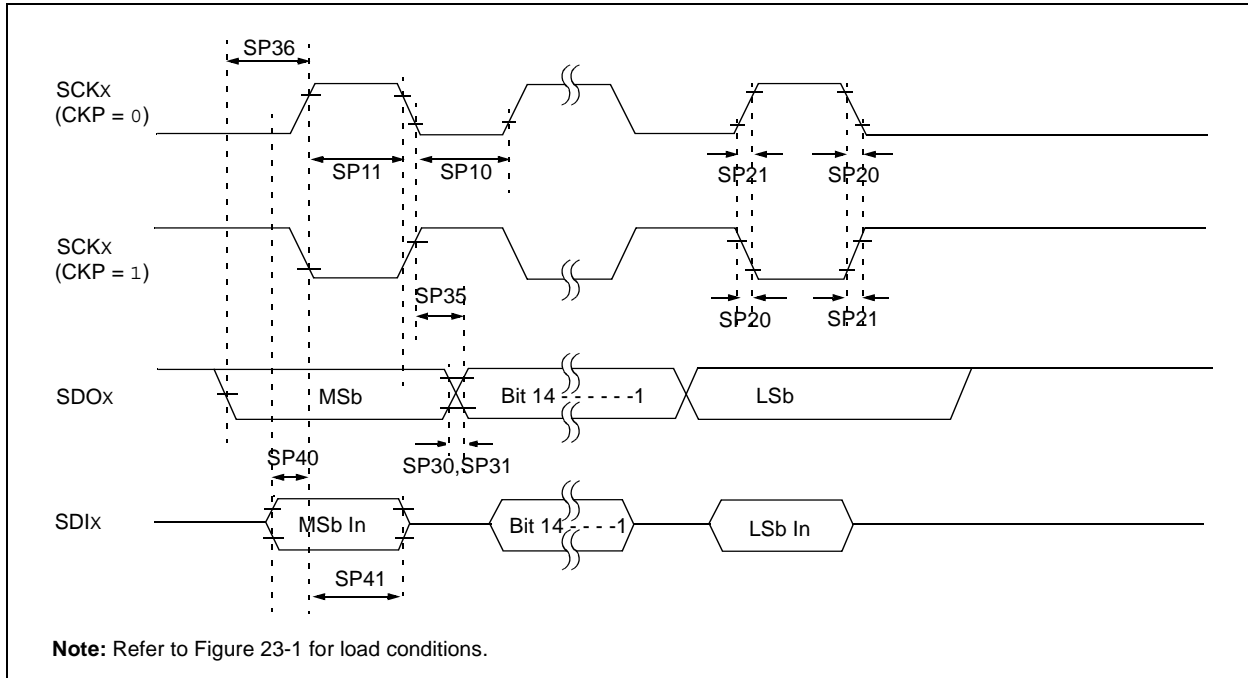
# PIC24H

**TABLE 23-27: SPIx MASTER MODE (CKE = 0) TIMING REQUIREMENTS**

AC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$				
Param No.	Symbol	Characteristic <sup>(1)</sup>	Min	Typ <sup>(2)</sup>	Max	Units	Conditions
SP10	TscL	SCKx Output Low Time <sup>(3)</sup>	Tcy/2	—	—	ns	—
SP11	TscH	SCKx Output High Time <sup>(3)</sup>	Tcy/2	—	—	ns	—
SP20	TscF	SCKx Output Fall Time <sup>(4)</sup>	—	—	—	ns	See parameter D032
SP21	TscR	SCKx Output Rise Time <sup>(4)</sup>	—	—	—	ns	See parameter D031
SP30	TdoF	SDOx Data Output Fall Time <sup>(4)</sup>	—	—	—	ns	See parameter D032
SP31	TdoR	SDOx Data Output Rise Time <sup>(4)</sup>	—	—	—	ns	See parameter D031
SP35	TscH2doV, TscL2doV	SDOx Data Output Valid after SCKx Edge	—	—	30	ns	—
SP40	TdiV2scH, TdiV2scL	Setup Time of SDIx Data Input to SCKx Edge	20	—	—	ns	—
SP41	Tsch2diL, TscL2diL	Hold Time of SDIx Data Input to SCKx Edge	20	—	—	ns	—

- Note 1:** These parameters are characterized but not tested in manufacturing.
- 2:** Data in “Typ” column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.
- 3:** The minimum clock period for SCKx is 100 ns. Therefore, the clock generated in Master mode must not violate this specification.
- 4:** Assumes 50 pF load on all SPIx pins.

**FIGURE 23-10: SPIx MODULE MASTER MODE (CKE = 1) TIMING CHARACTERISTICS**

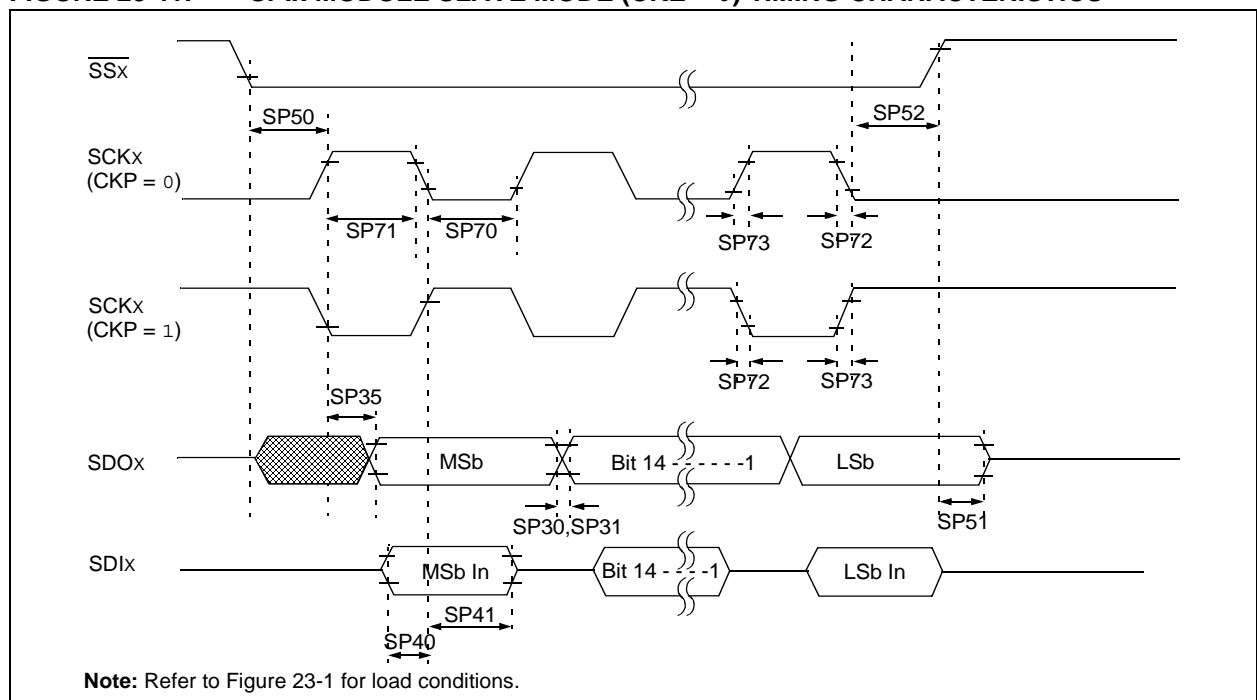


**TABLE 23-28: SPIx MODULE MASTER MODE (CKE = 1) TIMING REQUIREMENTS**

AC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$				
Param No.	Symbol	Characteristic <sup>(1)</sup>	Min	Typ <sup>(2)</sup>	Max	Units	Conditions
SP10	TscL	SCKx Output Low Time <sup>(3)</sup>	$T_{CY}/2$	—	—	ns	—
SP11	TscH	SCKx Output High Time <sup>(3)</sup>	$T_{CY}/2$	—	—	ns	—
SP20	TscF	SCKx Output Fall Time <sup>(4)</sup>	—	—	—	ns	See parameter D032
SP21	TscR	SCKx Output Rise Time <sup>(4)</sup>	—	—	—	ns	See parameter D031
SP30	TdoF	SDOx Data Output Fall Time <sup>(4)</sup>	—	—	—	ns	See parameter D032
SP31	TdoR	SDOx Data Output Rise Time <sup>(4)</sup>	—	—	—	ns	See parameter D031
SP35	Tsch2doV, TscL2doV	SDOx Data Output Valid after SCKx Edge	—	—	—	ns	—
SP36	TdoV2sc, TdoV2scL	SDOx Data Output Setup to First SCKx Edge	30	—	—	ns	—
SP40	TdiV2sch, TdiV2scL	Setup Time of SDIx Data Input to SCKx Edge	20	—	—	ns	—
SP41	Tsch2diL, TscL2diL	Hold Time of SDIx Data Input to SCKx Edge	20	—	—	ns	—

- Note 1:** These parameters are characterized but not tested in manufacturing.
- 2:** Data in “Typ” column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.
- 3:** The minimum clock period for SCKx is 100 ns. Therefore, the clock generated in Master mode must not violate this specification.
- 4:** Assumes 50 pF load on all SPIx pins.

**FIGURE 23-11: SPIx MODULE SLAVE MODE (CKE = 0) TIMING CHARACTERISTICS**



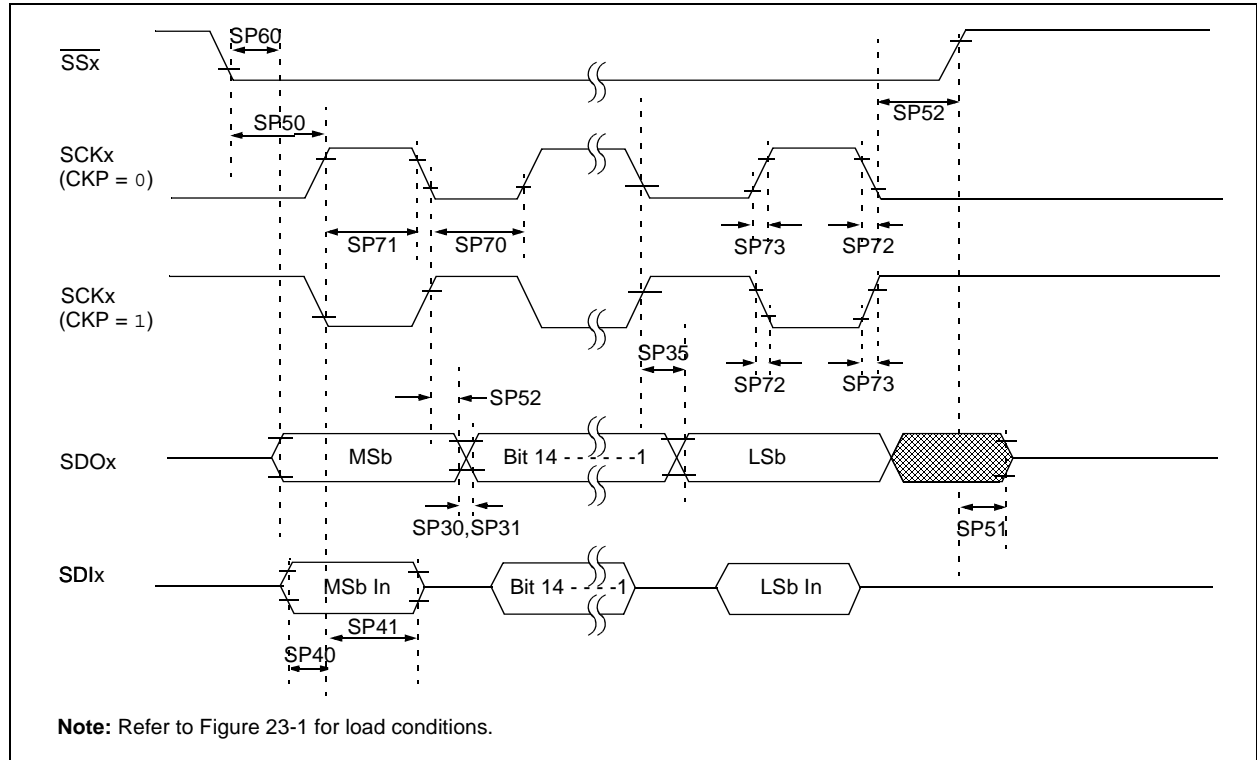
# PIC24H

**TABLE 23-29: SPIx MODULE SLAVE MODE (CKE = 0) TIMING REQUIREMENTS**

AC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$				
Param No.	Symbol	Characteristic <sup>(1)</sup>	Min	Typ <sup>(2)</sup>	Max	Units	Conditions
SP70	TscL	SCKx Input Low Time	30	—	—	ns	—
SP71	TscH	SCKx Input High Time	30	—	—	ns	—
SP72	TscF	SCKx Input Fall Time <sup>(3)</sup>	—	10	25	ns	—
SP73	TscR	SCKx Input Rise Time <sup>(3)</sup>	—	10	25	ns	—
SP30	TdoF	SDOx Data Output Fall Time <sup>(3)</sup>	—	—	—	ns	See parameter D032
SP31	TdoR	SDOx Data Output Rise Time <sup>(3)</sup>	—	—	—	ns	See parameter D031
SP35	Tsch2doV, TscL2doV	SDOx Data Output Valid after SCKx Edge	—	—	30	ns	—
SP40	TdiV2scH, TdiV2scL	Setup Time of SDIx Data Input to SCKx Edge	20	—	—	ns	—
SP41	Tsch2diL, TscL2diL	Hold Time of SDIx Data Input to SCKx Edge	20	—	—	ns	—
SP50	TssL2scH, TssL2scL	$\overline{\text{SS}}_x \downarrow$ to SCKx $\uparrow$ or SCKx Input	120	—	—	ns	—
SP51	TssH2doZ	$\overline{\text{SS}}_x \uparrow$ to SDOx Output High-Impedance <sup>(3)</sup>	10	—	50	ns	—
SP52	Tsch2ssH TscL2ssH	$\overline{\text{SS}}_x$ after SCKx Edge	$1.5 T_{CY} + 40$	—	—	ns	—

- Note 1:** These parameters are characterized but not tested in manufacturing.
- 2:** Data in “Typ” column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.
- 3:** Assumes 50 pF load on all SPIx pins.

**FIGURE 23-12: SPIx MODULE SLAVE MODE (CKE = 1) TIMING CHARACTERISTICS**



# PIC24H

**TABLE 23-30: SPIx MODULE SLAVE MODE (CKE = 1) TIMING REQUIREMENTS**

AC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$				
Param No.	Symbol	Characteristic <sup>(1)</sup>	Min	Typ <sup>(2)</sup>	Max	Units	Conditions
SP70	TscL	SCKx Input Low Time	30	—	—	ns	—
SP71	TscH	SCKx Input High Time	30	—	—	ns	—
SP72	TscF	SCKx Input Fall Time <sup>(3)</sup>	—	10	25	ns	—
SP73	TscR	SCKx Input Rise Time <sup>(3)</sup>	—	10	25	ns	—
SP30	TdoF	SDOx Data Output Fall Time <sup>(3)</sup>	—	—	—	ns	See parameter D032
SP31	TdoR	SDOx Data Output Rise Time <sup>(3)</sup>	—	—	—	ns	See parameter D031
SP35	Tsch2doV, TscL2doV	SDOx Data Output Valid after SCKx Edge	—	—	30	ns	—
SP40	TdiV2sch, TdiV2scL	Setup Time of SDIx Data Input to SCKx Edge	20	—	—	ns	—
SP41	Tsch2diL, TscL2diL	Hold Time of SDIx Data Input to SCKx Edge	20	—	—	ns	—
SP50	TssL2sch, TssL2scL	$\overline{\text{SSx}}$ ↓ to SCKx ↓ or SCKx ↑ Input	120	—	—	ns	—
SP51	TssH2doZ	$\overline{\text{SSx}}$ ↑ to SDOx Output High-Impedance <sup>(4)</sup>	10	—	50	ns	—
SP52	Tsch2ssH, TscL2ssH	$\overline{\text{SSx}}$ ↑ after SCKx Edge	$1.5 T_{CY} + 40$	—	—	ns	—
SP60	TssL2doV	SDOx Data Output Valid after $\overline{\text{SSx}}$ Edge	—	—	50	ns	—

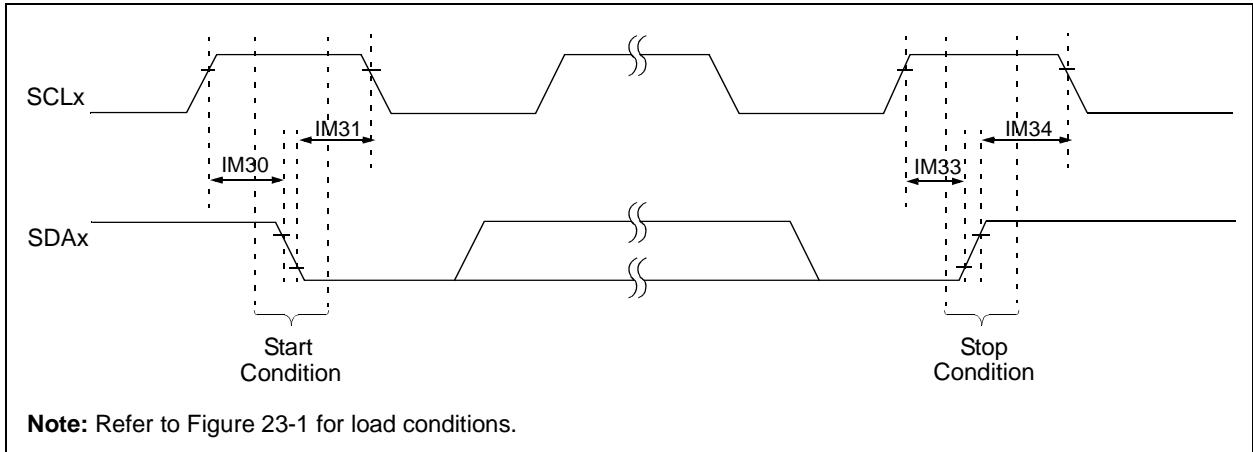
**Note 1:** These parameters are characterized but not tested in manufacturing.

**2:** Data in “Typ” column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

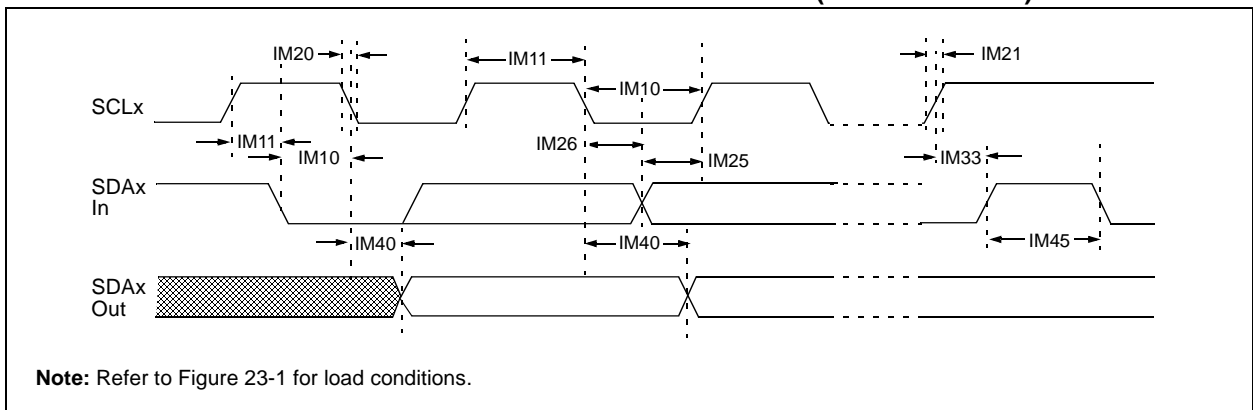
**3:** The minimum clock period for SCKx is 100 ns. Therefore, the clock generated in Master mode must not violate this specification.

**4:** Assumes 50 pF load on all SPIx pins.

**FIGURE 23-13: I2Cx BUS START/STOP BITS TIMING CHARACTERISTICS (MASTER MODE)**



**FIGURE 23-14: I2Cx BUS DATA TIMING CHARACTERISTICS (MASTER MODE)**



# PIC24H

**TABLE 23-31: I2Cx BUS DATA TIMING REQUIREMENTS (MASTER MODE)**

AC CHARACTERISTICS				Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C			
Param No.	Symbol	Characteristic		Min <sup>(1)</sup>	Max	Units	Conditions
IM10	TLO:SCL	Clock Low Time	100 kHz mode	$T_{CY}/2 (BRG + 1)$	—	μs	—
			400 kHz mode	$T_{CY}/2 (BRG + 1)$	—	μs	
			1 MHz mode <sup>(2)</sup>	$T_{CY}/2 (BRG + 1)$	—	μs	
IM11	THI:SCL	Clock High Time	100 kHz mode	$T_{CY}/2 (BRG + 1)$	—	μs	—
			400 kHz mode	$T_{CY}/2 (BRG + 1)$	—	μs	
			1 MHz mode <sup>(2)</sup>	$T_{CY}/2 (BRG + 1)$	—	μs	
IM20	TF:SCL	SDAx and SCLx Fall Time	100 kHz mode	—	300	ns	Cb is specified to be from 10 to 400 pF
			400 kHz mode	$20 + 0.1 C_b$	300	ns	
			1 MHz mode <sup>(2)</sup>	—	100	ns	
IM21	TR:SCL	SDAx and SCLx Rise Time	100 kHz mode	—	1000	ns	Cb is specified to be from 10 to 400 pF
			400 kHz mode	$20 + 0.1 C_b$	300	ns	
			1 MHz mode <sup>(2)</sup>	—	300	ns	
IM25	TSU:DAT	Data Input Setup Time	100 kHz mode	250	—	ns	—
			400 kHz mode	100	—	ns	
			1 MHz mode <sup>(2)</sup>	TBD	—	ns	
IM26	THD:DAT	Data Input Hold Time	100 kHz mode	0	—	ns	—
			400 kHz mode	0	0.9	μs	
			1 MHz mode <sup>(2)</sup>	TBD	—	ns	
IM30	TSU:STA	Start Condition Setup Time	100 kHz mode	$T_{CY}/2 (BRG + 1)$	—	μs	Only relevant for Repeated Start condition
			400 kHz mode	$T_{CY}/2 (BRG + 1)$	—	μs	
			1 MHz mode <sup>(2)</sup>	$T_{CY}/2 (BRG + 1)$	—	μs	
IM31	THD:STA	Start Condition Hold Time	100 kHz mode	$T_{CY}/2 (BRG + 1)$	—	μs	After this period the first clock pulse is generated
			400 kHz mode	$T_{CY}/2 (BRG + 1)$	—	μs	
			1 MHz mode <sup>(2)</sup>	$T_{CY}/2 (BRG + 1)$	—	μs	
IM33	TSU:STO	Stop Condition Setup Time	100 kHz mode	$T_{CY}/2 (BRG + 1)$	—	μs	—
			400 kHz mode	$T_{CY}/2 (BRG + 1)$	—	μs	
			1 MHz mode <sup>(2)</sup>	$T_{CY}/2 (BRG + 1)$	—	μs	
IM34	THD:STO	Stop Condition Hold Time	100 kHz mode	$T_{CY}/2 (BRG + 1)$	—	ns	—
			400 kHz mode	$T_{CY}/2 (BRG + 1)$	—	ns	
			1 MHz mode <sup>(2)</sup>	$T_{CY}/2 (BRG + 1)$	—	ns	
IM40	TAA:SCL	Output Valid From Clock	100 kHz mode	—	3500	ns	—
			400 kHz mode	—	1000	ns	
			1 MHz mode <sup>(2)</sup>	—	—	ns	
IM45	TBF:SDA	Bus Free Time	100 kHz mode	4.7	—	μs	Time the bus must be free before a new transmission can start
			400 kHz mode	1.3	—	μs	
			1 MHz mode <sup>(2)</sup>	TBD	—	μs	
IM50	CB	Bus Capacitive Loading		—	400	pF	

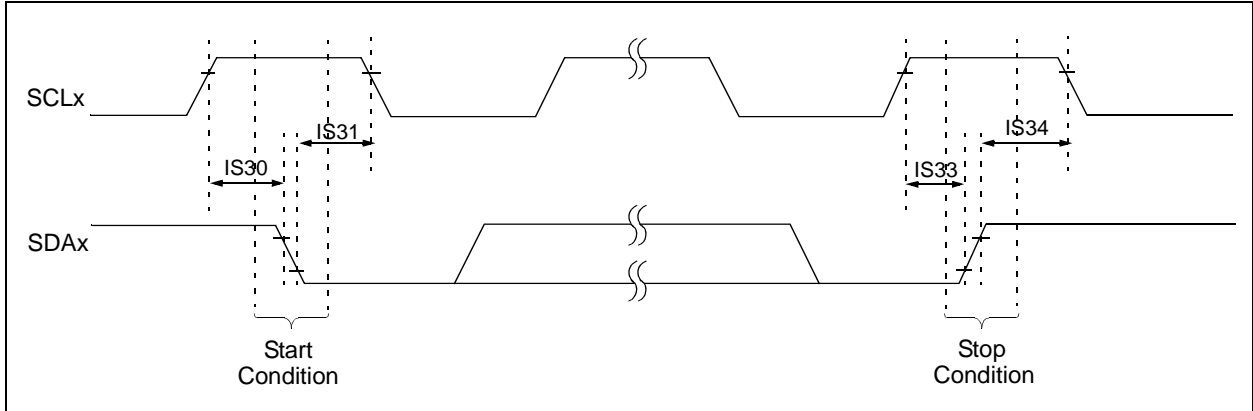
**Legend:** TBD = To Be Determined

**Note 1:** BRG is the value of the I<sup>2</sup>C Baud Rate Generator. Refer to **Section 21. “Inter-Integrated Circuit (I<sup>2</sup>C™)”** in the “dsPIC30F Family Reference Manual” (DS70046).

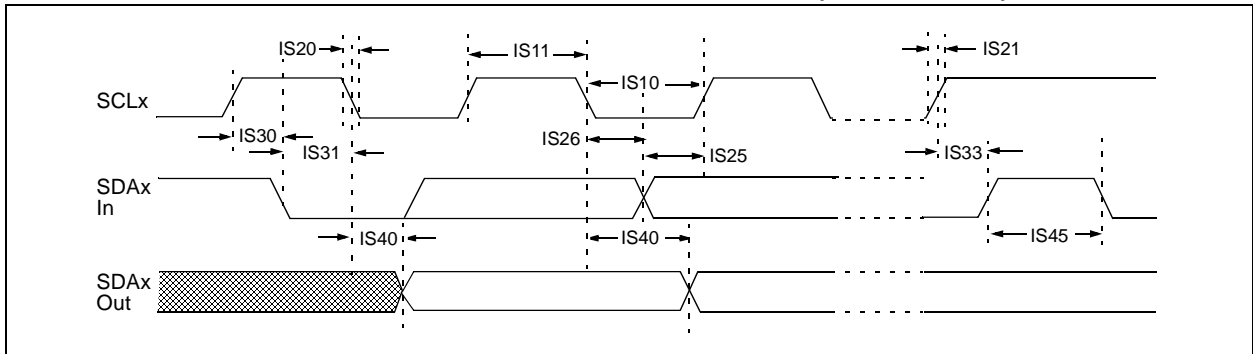
**2:** Maximum pin capacitance = 10 pF for all I2Cx pins (for 1 MHz mode only).



**FIGURE 23-15: I2Cx BUS START/STOP BITS TIMING CHARACTERISTICS (SLAVE MODE)**



**FIGURE 23-16: I2Cx BUS DATA TIMING CHARACTERISTICS (SLAVE MODE)**



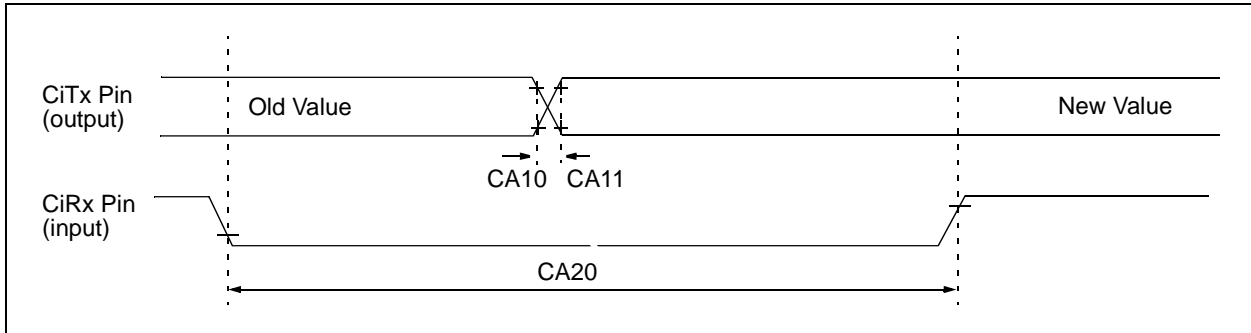
# PIC24H

**TABLE 23-32: I2Cx BUS DATA TIMING REQUIREMENTS (SLAVE MODE)**

AC CHARACTERISTICS				Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$			
Param No.	Symbol	Characteristic		Min	Max	Units	Conditions
IS10	TLO:SCL	Clock Low Time	100 kHz mode	4.7	—	$\mu\text{s}$	Device must operate at a minimum of 1.5 MHz
			400 kHz mode	1.3	—	$\mu\text{s}$	Device must operate at a minimum of 10 MHz
			1 MHz mode <sup>(1)</sup>	0.5	—	$\mu\text{s}$	—
IS11	THI:SCL	Clock High Time	100 kHz mode	4.0	—	$\mu\text{s}$	Device must operate at a minimum of 1.5 MHz
			400 kHz mode	0.6	—	$\mu\text{s}$	Device must operate at a minimum of 10 MHz
			1 MHz mode <sup>(1)</sup>	0.5	—	$\mu\text{s}$	—
IS20	TF:SCL	SDAx and SCLx Fall Time	100 kHz mode	—	300	ns	Cb is specified to be from 10 to 400 pF
			400 kHz mode	$20 + 0.1 C_B$	300	ns	
			1 MHz mode <sup>(1)</sup>	—	100	ns	
IS21	TR:SCL	SDAx and SCLx Rise Time	100 kHz mode	—	1000	ns	Cb is specified to be from 10 to 400 pF
			400 kHz mode	$20 + 0.1 C_B$	300	ns	
			1 MHz mode <sup>(1)</sup>	—	300	ns	
IS25	TSU:DAT	Data Input Setup Time	100 kHz mode	250	—	ns	—
			400 kHz mode	100	—	ns	
			1 MHz mode <sup>(1)</sup>	100	—	ns	
IS26	THD:DAT	Data Input Hold Time	100 kHz mode	0	—	ns	—
			400 kHz mode	0	0.9	$\mu\text{s}$	
			1 MHz mode <sup>(1)</sup>	0	0.3	$\mu\text{s}$	
IS30	TSU:STA	Start Condition Setup Time	100 kHz mode	4.7	—	$\mu\text{s}$	Only relevant for Repeated Start condition
			400 kHz mode	0.6	—	$\mu\text{s}$	
			1 MHz mode <sup>(1)</sup>	0.25	—	$\mu\text{s}$	
IS31	THD:STA	Start Condition Hold Time	100 kHz mode	4.0	—	$\mu\text{s}$	After this period, the first clock pulse is generated
			400 kHz mode	0.6	—	$\mu\text{s}$	
			1 MHz mode <sup>(1)</sup>	0.25	—	$\mu\text{s}$	
IS33	TSU:STO	Stop Condition Setup Time	100 kHz mode	4.7	—	$\mu\text{s}$	—
			400 kHz mode	0.6	—	$\mu\text{s}$	
			1 MHz mode <sup>(1)</sup>	0.6	—	$\mu\text{s}$	
IS34	THD:STO	Stop Condition Hold Time	100 kHz mode	4000	—	ns	—
			400 kHz mode	600	—	ns	
			1 MHz mode <sup>(1)</sup>	250	—	ns	
IS40	TAA:SCL	Output Valid From Clock	100 kHz mode	0	3500	ns	—
			400 kHz mode	0	1000	ns	
			1 MHz mode <sup>(1)</sup>	0	350	ns	
IS45	TBF:SDA	Bus Free Time	100 kHz mode	4.7	—	$\mu\text{s}$	Time the bus must be free before a new transmission can start
			400 kHz mode	1.3	—	$\mu\text{s}$	
			1 MHz mode <sup>(1)</sup>	0.5	—	$\mu\text{s}$	
IS50	CB	Bus Capacitive Loading		—	400	pF	—

**Note 1:** Maximum pin capacitance = 10 pF for all I2Cx pins (for 1 MHz mode only).

**FIGURE 23-17: ECAN™ MODULE I/O TIMING CHARACTERISTICS**



**TABLE 23-33: ECAN™ MODULE I/O TIMING REQUIREMENTS**

AC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$				
Param No.	Symbol	Characteristic <sup>(1)</sup>	Min	Typ <sup>(2)</sup>	Max	Units	Conditions
CA10	TioF	Port Output Fall Time	—	—	—	ns	See parameter D032
CA11	TioR	Port Output Rise Time	—	—	—	ns	See parameter D031
CA20	Tcwf	Pulse Width to Trigger CAN Wake-up Filter	500			ns	—

**Note 1:** These parameters are characterized but not tested in manufacturing.

**Note 2:** Data in “Typ” column is at 5V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

# PIC24H

**TABLE 23-34: A/D MODULE SPECIFICATIONS**

AC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$				
Param No.	Symbol	Characteristic	Min.	Typ	Max.	Units	Conditions
<b>Device Supply</b>							
AD01	AVDD	Module VDD Supply	Greater of VDD - 0.3 or 3.0	—	Lesser of VDD + 0.3 or 3.6	V	—
AD02	AVSS	Module VSS Supply	VSS - 0.3	—	VSS + 0.3	V	—
<b>Reference Inputs</b>							
AD05	VREFH	Reference Voltage High	AVSS + 1.7	—	AVDD	V	—
AD06	VREFL	Reference Voltage Low	AVSS	—	AVDD - 1.7	V	—
AD07	VREF	Absolute Reference Voltage	AVSS - 0.3	—	AVDD + 0.3	V	—
AD08	IREF	Current Drain	—	200 .001	300 3	$\mu\text{A}$ $\mu\text{A}$	A/D operating A/D off
<b>Analog Input</b>							
AD10	VINH-VINL	Full-Scale Input Span	VREFL		VREFH	V	See Note
AD11	VIN	Absolute Input Voltage	AVSS - 0.3		AVDD + 0.3	V	—
AD12	—	Leakage Current	—	$\pm 0.001$	$\pm 0.610$	$\mu\text{A}$	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 5V Source Impedance = 2.5 K $\Omega$
AD13	—	Leakage Current	—	$\pm 0.001$	$\pm 0.610$	$\mu\text{A}$	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3V Source Impedance = 2.5 K $\Omega$
AD17	RIN	Recommended Impedance of Analog Voltage Source	—	—	1K 2.5K	$\Omega$ $\Omega$	10-bit 12-bit
<b>ADC Accuracy (12-bit Mode)</b>							
AD20a	Nr	Resolution	12 data bits			bits	
AD21a	INL	Integral Nonlinearity <sup>(2)</sup>	—	—	$< \pm 2$	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3V
AD22a	DNL	Differential Nonlinearity <sup>(2)</sup>	—	—	$< \pm 1$	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3V
AD23a	GERR	Gain Error <sup>(2)</sup>	TBD	TBD	$\pm 3$	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3V
AD24a	EOFF	Offset Error <sup>(2)</sup>	TBD	TBD	$\pm 2$	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3V
AD25a	—	Monotonicity <sup>(1)</sup>	—	—	—	—	Guaranteed
AD26a	CMRR	Common-Mode Rejection	—	TBD	—	dB	—
AD27a	PSRR	Power Supply Rejection Ratio	—	TBD	—	dB	—
AD28a	CTLK	Channel to Channel Crosstalk	—	TBD	—	dB	—

**Legend:** TBD = To Be Determined

**Note 1:** The A/D conversion result never decreases with an increase in the input voltage, and has no missing codes.

**2:** Measurements taken with external VREF+ and VREF- used as the ADC voltage reference.

**TABLE 23-34: A/D MODULE SPECIFICATIONS (CONTINUED)**

AC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$				
Param No.	Symbol	Characteristic	Min.	Typ	Max.	Units	Conditions
<b>Dynamic Performance (12-bit Mode)</b>							
AD30a	THD	Total Harmonic Distortion	—	TBD	—	dB	—
AD31a	SINAD	Signal to Noise and Distortion	—	TBD	—	dB	—
AD32a	SFDR	Spurious Free Dynamic Range	—	TBD	—	dB	—
AD33a	FNYQ	Input Signal Bandwidth	—	—	250	kHz	—
AD34a	ENOB	Effective Number of Bits	—	TBD	TBD	bits	—
<b>ADC Accuracy (10-bit Mode)</b>							
AD20b	Nr	Resolution	10 data bits			bits	
AD21b	INL	Integral Nonlinearity	—	TBD	$\leq \pm 2$	LSb	$V_{INL} = AV_{SS} = V_{REFL} = 0V, AV_{DD} = V_{REFH} = 3V$
AD22b	DNL	Differential Nonlinearity	—	TBD	$\leq \pm 1$	LSb	$V_{INL} = AV_{SS} = V_{REFL} = 0V, AV_{DD} = V_{REFH} = 3V$
AD23b	GERR	Gain Error	—	TBD	$\leq \pm 3$	LSb	$V_{INL} = AV_{SS} = V_{REFL} = 0V, AV_{DD} = V_{REFH} = 3V$
AD24b	E <sub>OFF</sub>	Offset Error	—	TBD	$\leq \pm 2$	LSb	$V_{INL} = AV_{SS} = V_{REFL} = 0V, AV_{DD} = V_{REFH} = 3V$
AD25b	—	Monotonicity <sup>(1)</sup>	—	—	—	—	Guaranteed
AD26b	CMRR	Common-Mode Rejection	—	TBD	—	dB	—
AD27b	PSRR	Power Supply Rejection Ratio	—	TBD	—	dB	—
AD28b	CTLK	Channel to Channel Crosstalk	—	TBD	—	dB	—
<b>Dynamic Performance (10-bit Mode)</b>							
AD30b	THD	Total Harmonic Distortion	—	—	—	dB	—
AD31b	SINAD	Signal to Noise and Distortion	—	TBD	—	dB	—
AD32b	SFDR	Spurious Free Dynamic Range	—	TBD	—	dB	—
AD33b	FNYQ	Input Signal Bandwidth	—	—	550	kHz	—
AD34b	ENOB	Effective Number of Bits	—	TBD	TBD	bits	—

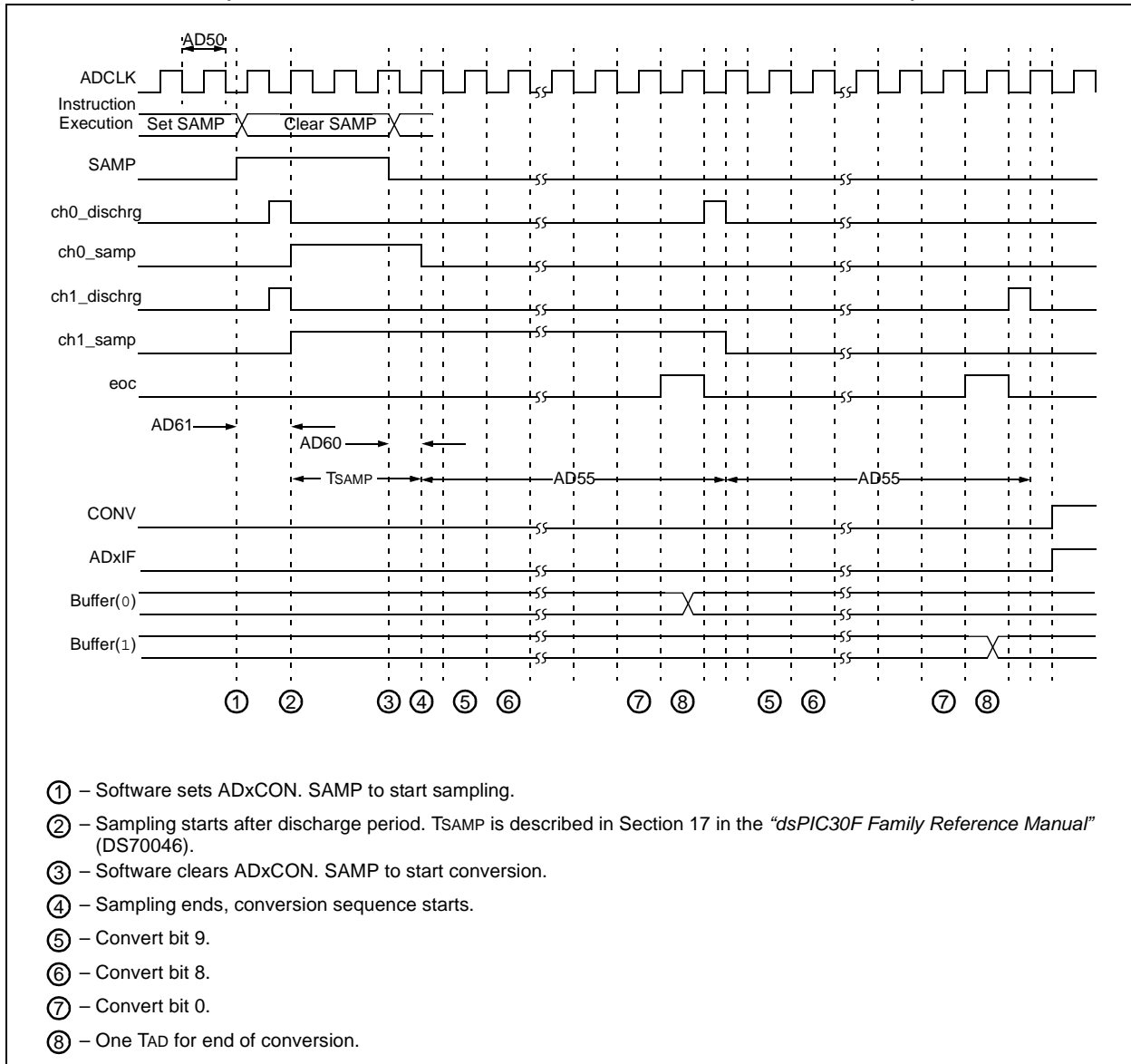
**Legend:** TBD = To Be Determined

**Note 1:** The A/D conversion result never decreases with an increase in the input voltage, and has no missing codes.

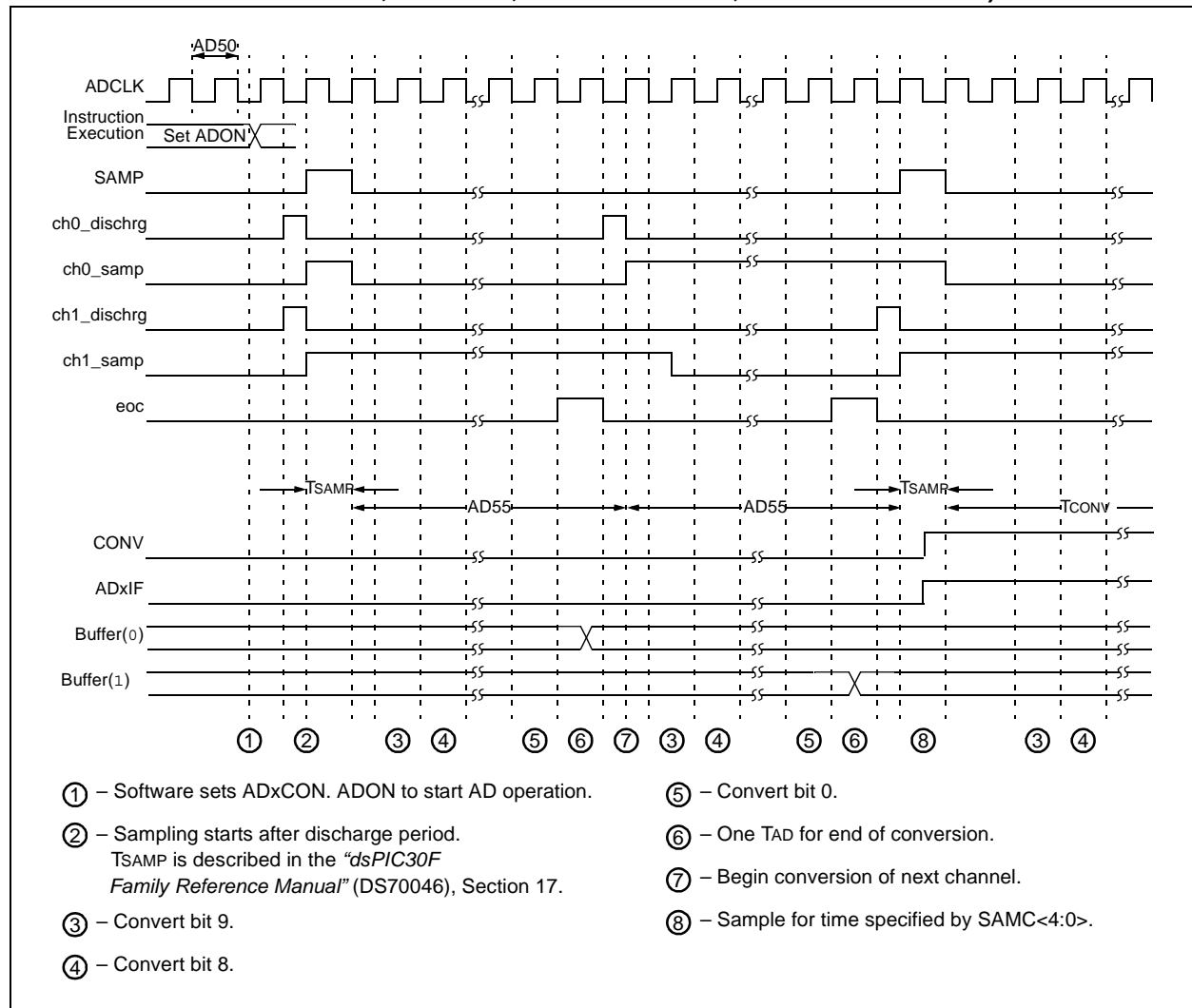
**2:** Measurements taken with external VREF+ and VREF- used as the ADC voltage reference.

# PIC24H

**FIGURE 23-18: A/D CONVERSION (10-BIT MODE) TIMING CHARACTERISTICS**  
**(CHPS<1:0> = 01, SIMSAM = 0, ASAM = 0, SSRC<2:0> = 000)**



**FIGURE 23-19: A/D CONVERSION (10-BIT MODE) TIMING CHARACTERISTICS (CHPS<1:0> = 01, SIMSAM = 0, ASAM = 1, SSRC<2:0> = 111, SAMC<4:0> = 00001)**



# PIC24H

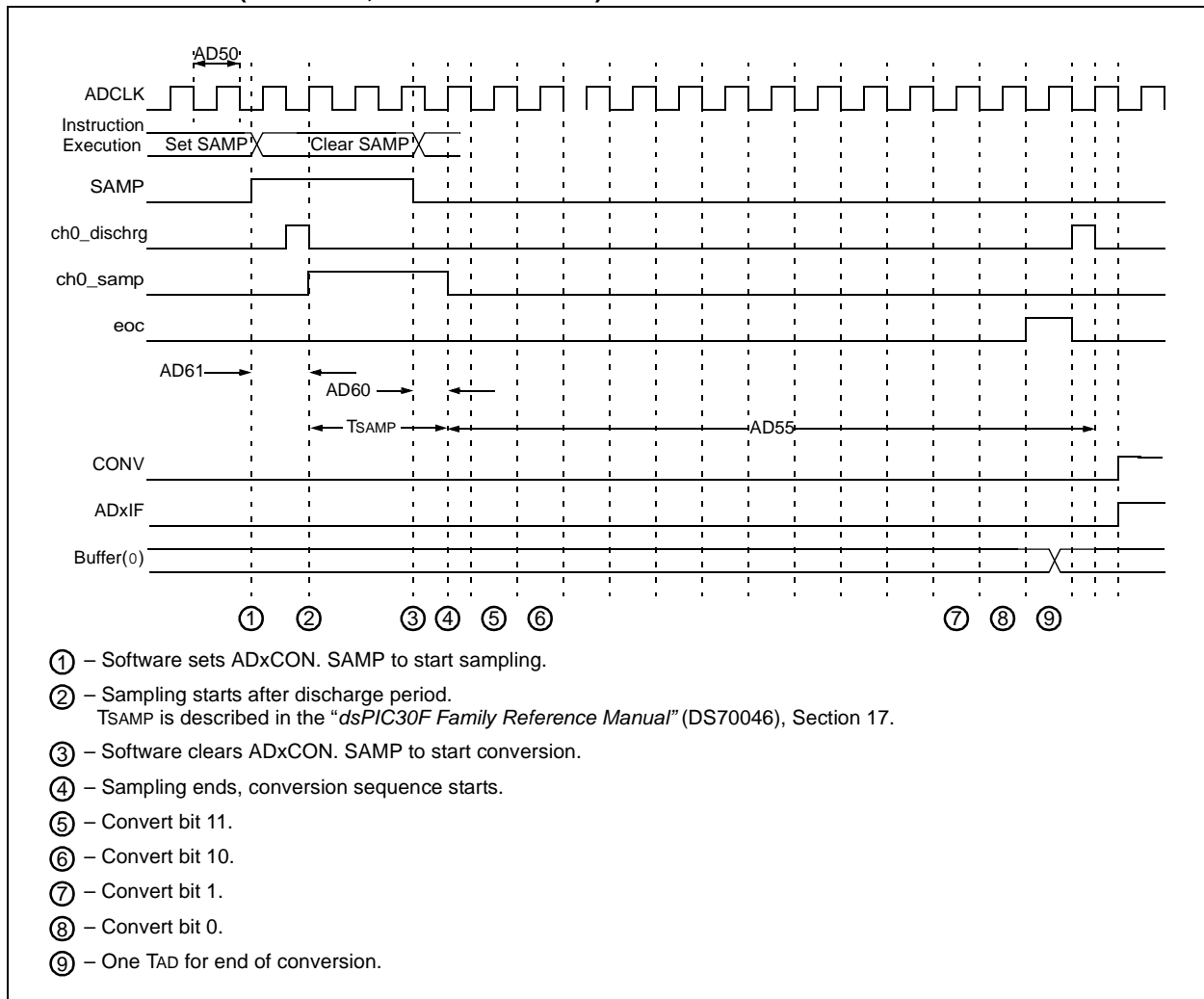
**TABLE 23-35: A/D CONVERSION (10-BIT MODE) TIMING REQUIREMENTS**

AC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$				
Param No.	Symbol	Characteristic	Min.	Typ <sup>(1)</sup>	Max.	Units	Conditions
<b>Clock Parameters</b>							
AD50	TAD	A/D Clock Period <sup>(2)</sup>	70	—	—	ns	T <sub>CY</sub> = 70 ms, ADxCON3 in default state
AD51	tRC	A/D Internal RC Oscillator Period	—	250	—	ns	—
<b>Conversion Rate</b>							
AD55	tCONV	Conversion Time	—	12 TAD	—	—	—
AD56	FCNV	Throughput Rate	—	—	1.1	Msp/s	—
AD57	TSAMP	Sample Time	—	1 TAD	—	—	—
<b>Timing Parameters</b>							
AD60	tPCS	Conversion Start from Sample Trigger <sup>(3)</sup>	—	1.0 TAD	—	—	Auto-Convert Trigger (SSRC<2:0> = 111) not selected
AD61	tPSS	Sample Start from Setting Sample (SAMP) bit	0.5 TAD	—	1.5 TAD	—	—
AD62	tcSS	Conversion Completion to Sample Start (ASAM = 1) <sup>(3)</sup>	—	0.5 TAD	—	—	—
AD63	tDPU	Time to Stabilize Analog Stage from A/D Off to A/D On <sup>(3)</sup>	—	20	—	μs	—

- Note 1:** These parameters are characterized but not tested in manufacturing.  
**Note 2:** Because the sample caps will eventually lose charge, clock rates below 10 kHz can affect linearity performance, especially at elevated temperatures.  
**Note 3:** Characterized by design but not tested.



**FIGURE 23-20: A/D CONVERSION (12-BIT MODE) TIMING CHARACTERISTICS**  
**(ASAM = 0, SSRC<2:0> = 000)**



# PIC24H

**TABLE 23-36: A/D CONVERSION (12-BIT MODE) TIMING REQUIREMENTS)**

AC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$				
Param No.	Symbol	Characteristic	Min.	Typ	Max.	Units	Conditions
<b>Clock Parameters</b>							
AD50	TAD	A/D Clock Period <sup>(1)</sup>	133	—	—	ns	Tcy = 133 ns, ADxCON3 in default state
AD51	tRC	A/D Internal RC Oscillator Period	—	250	—	ns	—
<b>Conversion Rate</b>							
AD55	tCONV	Conversion Time	—	14 TAD	—	ns	—
AD56	FCNV	Throughput Rate	—	—	500	ksps	—
AD57	TSAMP	Sample Time	—	1 TAD	—	ns	—
<b>Timing Parameters</b>							
AD60	tPCS	Conversion Start from Sample Trigger	—	1.0 TAD	—	ns	—
AD61	tPSS	Sample Start from Setting Sample (SAMP) bit	0.5 TAD	—	1.5 TAD	ns	—
AD62	tCSS	Conversion Completion to Sample Start (ASAM = 1)	—	—	—	ns	—
AD63	tDPU	Time to Stabilize Analog Stage from A/D Off to A/D On	—	250	—	$\mu\text{s}$	—

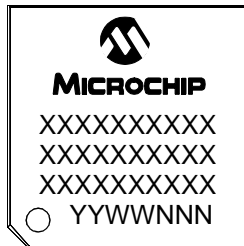
**Legend:** TBD = To Be Determined

**Note 1:** Because the sample caps will eventually lose charge, clock rates below 10 kHz can affect linearity performance, especially at elevated temperatures.

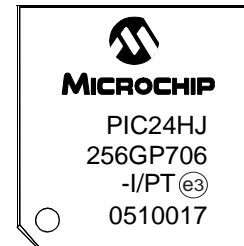
## 24.0 PACKAGING INFORMATION

### 24.1 Package Marking Information

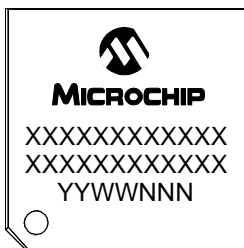
64-Lead TQFP (10x10x1 mm)



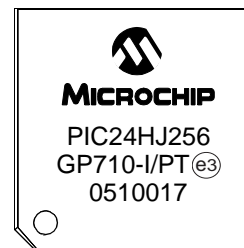
Example



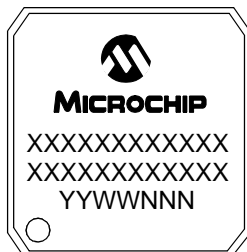
100-Lead TQFP (12x12x1 mm)



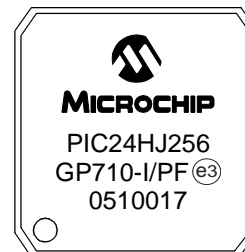
Example



100-Lead TQFP (14x14x1mm)



100-Lead TQFP (14x14x1mm)



<b>Legend:</b>	XX...X	Customer-specific information
	Y	Year code (last digit of calendar year)
	YY	Year code (last 2 digits of calendar year)
	WW	Week code (week of January 1 is week '01')
	NNN	Alphanumeric traceability code
	e3	Pb-free JEDEC designator for Matte Tin (Sn)
	*	This package is Pb-free. The Pb-free JEDEC designator (e3) can be found on the outer packaging for this package.

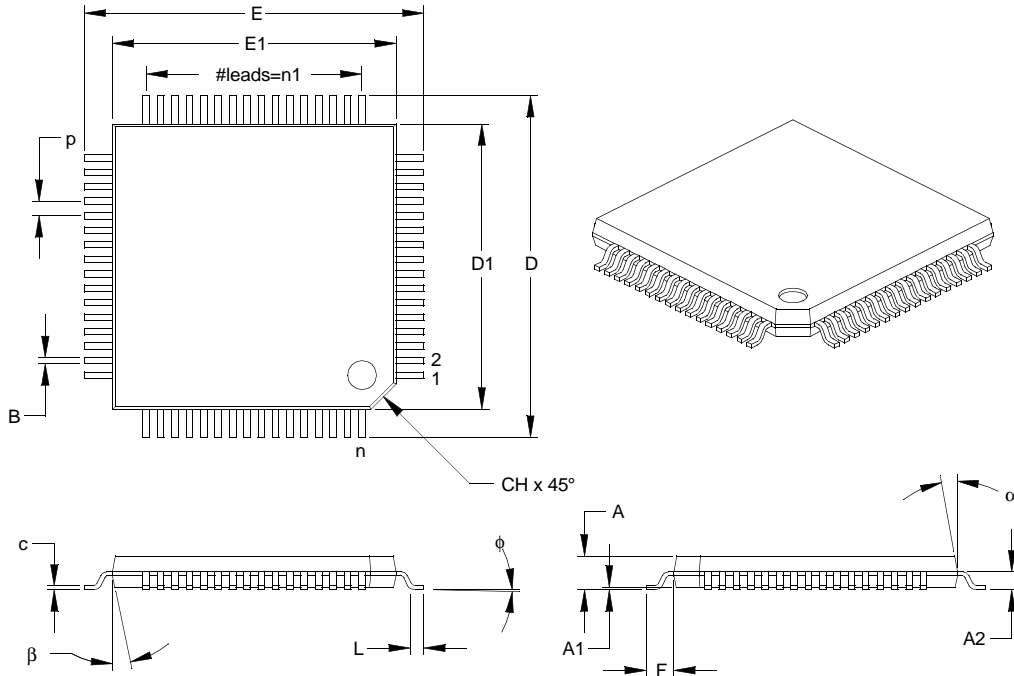
**Note:** In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.

# PIC24H

## 24.2 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	F	.039 REF.			1.00 REF.		
Foot Angle	phi	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	alpha	5	10	15	5	10	15
Mold Draft Angle Bottom	beta	5	10	15	5	10	15

\* Controlling Parameter

#### Notes:

Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

REF: Reference Dimension, usually without tolerance, for information purposes only.

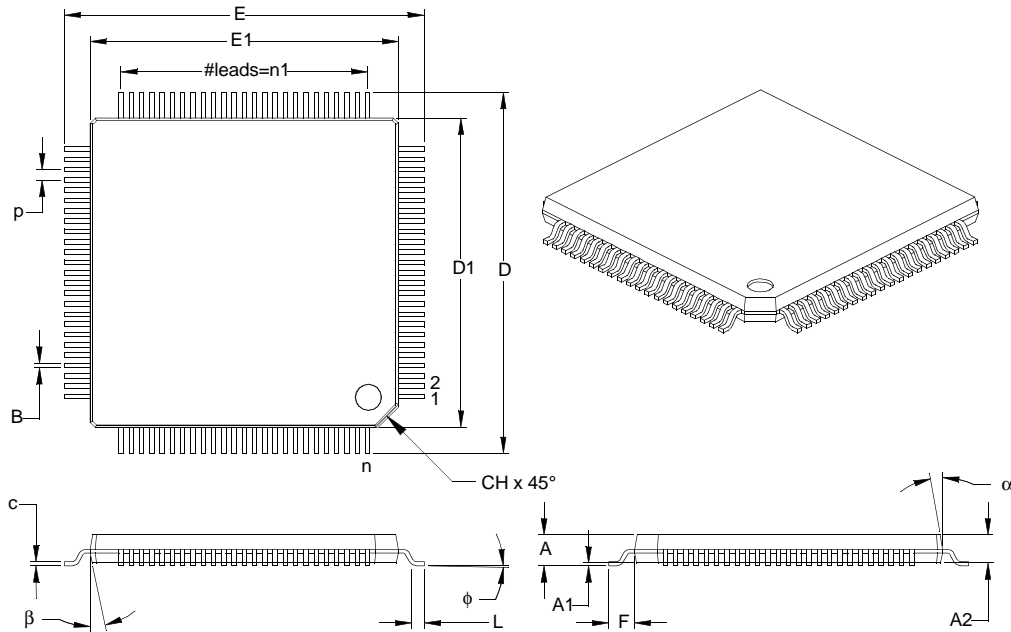
See ASME Y14.5M

JEDEC Equivalent: MS-026

Drawing No. C04-085

Revised 07-22-05

## 100-Lead Plastic Thin-Quad Flatpack (PT) 12x12x1 mm Body, 1.0/0.10 mm Lead Form (TQFP)



Dimension Limits	Units	INCHES			MILLIMETERS*		
		MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	n		100			100	
Pitch	p	.016 BSC			0.40 BSC		
Pins per Side	n1		25			25	
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 REF.			1.00 REF.		
Foot Angle	$\phi$	0°	3.5°	7°	0°	3.5°	7°
Overall Width	E	.551 BSC			14.00 BSC		
Overall Length	D	.551 BSC			14.00 BSC		
Molded Package Width	E1	.472 BSC			12.00 BSC		
Molded Package Length	D1	.472 BSC			12.00 BSC		
Lead Thickness	c	.004	.006	.008	0.09	0.15	0.20
Lead Width	B	.005	.007	.009	0.13	0.18	0.23
Mold Draft Angle Top	$\alpha$	5°	10°	15°	5°	10°	15°
Mold Draft Angle Bottom	$\beta$	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.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

See ASME Y14.5M

REF: Reference Dimension, usually without tolerance, for information purposes only.

See ASME Y14.5M

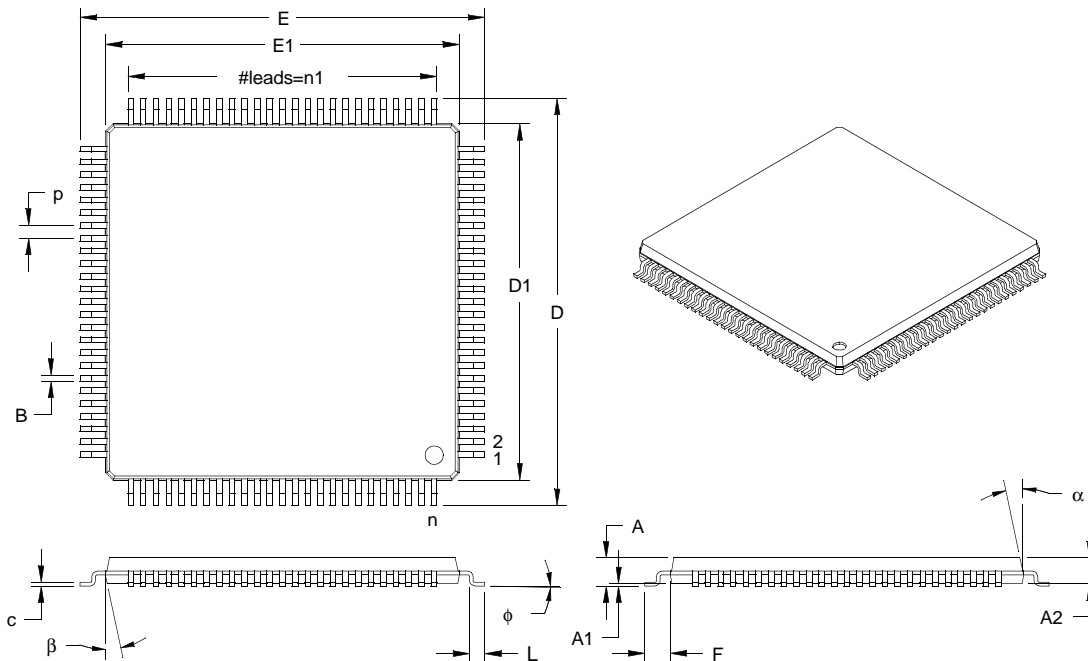
JEDEC Equivalent: MS-026

Drawing No. C04-100

Revised 07-22-05

# PIC24H

## 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 BSC			0.50 BSC		
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	F	.039 REF			1.00 REF		
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

### Notes:

Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" (0.254mm) per side.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

See ASME Y14.5M

REF: Reference Dimension, usually without tolerance, for information purposes only.

See ASME Y14.5M

JEDEC Equivalent: MS-026

Drawing No. C04-110

Revised 07-21-05

## APPENDIX A: REVISION HISTORY

### Revision A (February 2006)

- Initial release of this document

### Revision B (March 2006)

- Updated the Configuration Bits Description table (Table 20.1)
- Updated registers and register maps
- Updated **Section 15.0 “Serial Peripheral Interface (SPI)”**
- Updated **Section 23.0 “Electrical Characteristics”**
- Updated pinout diagrams
- Additional minor corrections throughout document text

### Revision C (May 2006)

- Updated **Section 23.0 “Electrical Characteristics”**
- Updated the Configuration Bits Description table (Table 20.1)
- Additional minor corrections throughout document text

# PIC24H

---

NOTES:



## INDEX

### A

A/D Converter .....	207
DMA .....	207
Initialization .....	207
Key Features.....	207
AC Characteristics .....	246
Internal RC Accuracy .....	248
Load Conditions .....	246
ADC Module	
ADC1 Register Map .....	37
ADC2 Register Map .....	37
Alternate Vector Table (AIVT) .....	67
Arithmetic Logic Unit (ALU).....	23
Assembler	
MPASM Assembler.....	236
Automatic Clock Stretch.....	161
Receive Mode .....	161
Transmit Mode.....	161

### B

Block Diagrams	
16-bit Timer1 Module .....	137
A/D Module .....	208, 209
Connections for On-Chip Voltage Regulator.....	223
ECAN Module .....	178
Input Capture .....	145
Output Compare .....	149
PIC24H .....	14
PIC24H CPU Core .....	18
PIC24H Oscillator System Diagram.....	125
PIC24H PLL.....	127
Reset System.....	61
Shared Port Structure .....	135
SPI .....	152
Timer2 (16-bit) .....	141
Timer2/3 (32-bit) .....	140
UART .....	169
Watchdog Timer (WDT) .....	224

### C

C Compilers	
MPLAB C18 .....	236
MPLAB C30 .....	236
Clock Switching.....	132
Enabling .....	132
Sequence.....	132
Code Examples	
DMA Sample Initialization Method.....	115
Erasing a Program Memory Page.....	58
Initiating a Programming Sequence.....	59
Loading Write Buffers .....	59
Port Write/Read .....	136
PWRSV Instruction Syntax.....	133
Code Protection .....	221, 225
Configuration Bits.....	221
Description (Table).....	222
Configuration Register Map .....	221
Configuring Analog Port Pins.....	136
CPU	
Control Register .....	20
CPU Clocking System.....	126
PLL Configuration .....	126
Selection .....	126
Sources.....	126
Customer Change Notification Service .....	283
Customer Support.....	283

### D

Data Address Space.....	27
Alignment.....	27
Memory Map for PIC24H Devices with 16 KBs RAM ..	29
Memory Map for PIC24H Devices with 8 KBs RAM ...	28
Near Data Space .....	27
Software Stack .....	48
Width .....	27
DC Characteristics.....	240
I/O Pin Input Specifications .....	244
I/O Pin Output Specifications.....	245
Idle Current (I <sub>DOZE</sub> ) .....	243
Idle Current (I <sub>IDLE</sub> ) .....	242
Operating Current (I <sub>DD</sub> ) .....	241
Power-Down Current (I <sub>PD</sub> ).....	242
Program Memory.....	245
Temperature and Voltage Specifications.....	240
Development Support.....	235
DMA	
Interrupts and Traps .....	114
Request Source Selection .....	114
DMA Module	
DMA Register Map .....	38
DMAC Operating Modes.....	112
Addressing.....	113
Byte or Word Transfer .....	113
Continuous or One-Shot.....	114
Manual Transfer .....	114
Null Data Peripheral Write .....	113
Ping-Pong.....	114
Transfer Direction .....	113
DMAC Registers .....	112
DMAxCNT .....	112
DMAxCON .....	112
DMAxPAD .....	112
DMAxREQ .....	112
DMAxSTA.....	112
DMAxSTB.....	112

### E

ECAN Module	
Baud Rate Setting .....	182
ECAN1 Register Map (C1CTRL1.WIN = 0 or 1).....	39
ECAN1 Register Map (C1CTRL1.WIN = 0).....	40
ECAN1 Register Map (C1CTRL1.WIN = 1).....	41
ECAN2 Register Map (C2CTRL1.WIN = 0 or 1).....	42
ECAN2 Register Map (C2CTRL1.WIN = 0).....	43
ECAN2 Register Map (C2CTRL1.WIN = 1).....	43
Frame Types .....	177
Message Reception.....	179
Message Transmission.....	181
Modes of Operation .....	179
Overview.....	177
Electrical Characteristics .....	239
AC.....	246
Enhanced CAN Module .....	177

# PIC24H

Equations			
A/D Conversion Clock Period .....	210	Instruction Addressing Modes .....	48
Calculating the PWM Period .....	148	File Register Instructions .....	48
Calculation for Maximum PWM Resolution .....	148	Fundamental Modes Supported .....	49
Device Operating Frequency .....	126	MCU Instructions .....	48
FOSC Calculation .....	126	Move and Accumulator Instructions.....	49
Relationship Between Device and SPI Clock		Other Instructions .....	49
Speed.....	154	Instruction Set	
Serial Clock Rate .....	159	Overview.....	230
Time Quantum for Clock Generation .....	183	Summary .....	227
UART Baud Rate with BRGH = 0 .....	170	Instruction-Based Power-Saving Modes.....	133
UART Baud Rate with BRGH = 1 .....	170	Idle .....	134
XT with PLL Mode Example.....	127	Sleep .....	133
Errata .....	11	Internal RC Oscillator	
<b>F</b>		Use with WDT.....	224
Flash Program Memory.....	55	Internet Address .....	283
Control Registers .....	56	Interrupt Control and Status Registers .....	71
Operations .....	56	IECx.....	71
Programming Algorithm .....	58	IFSx .....	71
RTSP Operation.....	56	INTCON1.....	71
Table Instructions.....	55	INTCON2.....	71
Flexible Configuration .....	221	IPCx.....	71
FSCM		Interrupt Setup Procedures.....	110
Delay for Crystal and PLL Clock Sources.....	65	Initialization.....	110
Device Resets.....	65	Interrupt Disable .....	110
<b>I</b>		Interrupt Service Routine.....	110
I/O Ports .....	135	Trap Service Routine.....	110
Parallel I/O (PIO).....	135	Interrupt Vector Table (IVT).....	67
Write/Read Timing .....	136	Interrupts Coincident with Power Save Instructions .....	134
<b>I<sup>2</sup>C</b>		<b>J</b>	
Addresses .....	161	JTAG Boundary Scan Interface .....	221
Baud Rate Generator.....	159	<b>M</b>	
General Call Address Support .....	161	Memory Organization .....	25
Interrupts.....	159	Microchip Internet Web Site.....	283
IPMI Support.....	161	Modes of Operation	
Master Mode Operation		Disable.....	179
Clock Arbitration.....	162	Initialization.....	179
Multi-Master Communication, Bus Collision		Listen All Messages.....	179
and Bus Arbitration .....	162	Listen Only.....	179
Operating Modes .....	159	Loopback .....	179
Registers.....	159	Normal Operation .....	179
Slave Address Masking .....	161	MPLAB ASM30 Assembler, Linker, Librarian .....	236
Slope Control .....	162	MPLAB ICD 2 In-Circuit Debugger .....	237
Software Controlled Clock Stretching (STREN = 1)..	161	MPLAB ICE 2000 High-Performance Universal	
<b>I<sup>2</sup>C Module</b>		In-Circuit Emulator.....	237
I2C1 Register Map .....	35	MPLAB ICE 4000 High-Performance Universal	
I2C2 Register Map .....	35	In-Circuit Emulator.....	237
In-Circuit Debugger.....	225	MPLAB Integrated Development Environment Software..	235
In-Circuit Emulation.....	221	MPLAB PM3 Device Programmer .....	237
In-Circuit Serial Programming (ICSP) .....	221, 225	MPLINK Object Linker/MPLIB Object Librarian .....	236
Infrared Support		Multi-Bit Data Shifter.....	23
Built-in IrDA Encoder and Decoder.....	171	<b>N</b>	
External IrDA, IrDA Clock Output.....	171	NVM Module	
Input Capture		Register Map .....	47
Registers.....	146	<b>O</b>	
Input Change Notification Module .....	136	Open-Drain Configuration.....	136
		Output Compare .....	147
		Registers .....	150

<b>P</b>	
Packaging .....	273
Details .....	274
Marking .....	273
Peripheral Module Disable (PMD) .....	134
PICSTART Plus Development Programmer .....	238
Pinout I/O Descriptions (table) .....	15
PMD Module	
Register Map.....	47
POR and Long Oscillator Start-up Times.....	65
PORTA	
Register Map.....	45
PORTB	
Register Map.....	45
PORTC	
Register Map.....	45
PORTD	
Register Map.....	45
PORTE	
Register Map.....	46
PORTF	
Register Map.....	46
PORTG	
Register Map.....	46
Power-Saving Features .....	133
Clock Frequency and Switching.....	133
Program Address Space .....	25
Construction.....	50
Data Access from Program Memory Using	
Program Space Visibility .....	53
Data Access from Program Memory Using Table	
Instructions .....	52
Data Access from, Address Generation.....	51
Memory Map .....	25
Table Read Instructions	
TBLRDH .....	52
TBLRDL .....	52
Visibility Operation .....	53
Program Memory	
Interrupt Vector .....	26
Organization.....	26
Reset Vector .....	26
Pulse-Width Modulation Mode .....	148
PWM	
Duty Cycle.....	148
Period.....	148
<b>R</b>	
Reader Response .....	284
Registers	
ADxCHS0 (ADCx Input Channel 0 Select).....	217
ADxCHS123 (ADCx Input Channel 1, 2, 3 Select) ..	216
ADxCON1 (ADCx Control 1) .....	211
ADxCON2 (ADCx Control 2) .....	213
ADxCON3 (ADCx Control 3) .....	214
ADxCON4 (ADCx Control 4) .....	215
ADxCSSH (ADCx Input Scan Select High) .....	218
ADxCSSL (ADCx Input Scan Select Low) .....	218
ADxPCFGH (ADCx Port Configuration High) .....	219
ADxPCFGL (ADCx Port Configuration Low) .....	219
CiBUFNT1 (ECAN Filter 0-3 Buffer Pointer).....	194
CiBUFNT2 (ECAN Filter 4-7 Buffer Pointer).....	195
CiBUFNT3 (ECAN Filter 8-11 Buffer Pointer).....	195
CiBUFNT4 (ECAN Filter 12-15 Buffer Pointer).....	196
CiCFG1 (ECAN Baud Rate Configuration 1) .....	192
CiCFG2 (ECAN Baud Rate Configuration 2).....	193
CiCTRL1 (ECAN Control 1) .....	184
CiCTRL2 (ECAN Control 2) .....	185
CiEC (ECAN Transmit/Receive Error Count) .....	191
CiFCTRL (ECAN FIFO Control) .....	187
CiFEN1 (ECAN Acceptance Filter Enable).....	194
CiFIFO (ECAN FIFO Status) .....	188
CiFMSKSEL1 (ECAN Filter 7-0 Mask Selection) .....	198
CiINTE (ECAN Interrupt Enable) .....	190
CiINTF (ECAN Interrupt Flag) .....	189
CiRXFnEID (ECAN Acceptance Filter n Extended	
Identifier) .....	197
CiRXFnSID (ECAN Acceptance Filter n Standard	
Identifier) .....	197
CiRXFUL1 (ECAN Receive Buffer Full 1).....	200
CiRXFUL2 (ECAN Receive Buffer Full 2).....	200
CiRXMnEID (ECAN Acceptance Filter Mask n	
Extended Identifier) .....	199
CiRXMnSID (ECAN Acceptance Filter Mask n	
Standard Identifier) .....	199
CiRXOVF1 (ECAN Receive Buffer Overflow 1).....	201
CiRXOVF2 (ECAN Receive Buffer Overflow 2).....	201
CiTRBnDLC (ECAN Buffer n Data Length Control)..	204
CiTRBnEID (ECAN Buffer n Extended Identifier) .....	203
CiTRBnSID (ECAN Buffer n Standard Identifier).....	203
CiTRBnSTAT (ECAN Receive Buffer n Status).....	205
CiTRmnCON (ECAN TX/RX Buffer m Control) .....	202
CiVEC (ECAN Interrupt Code) .....	186
CLKDIV (Clock Divisor) .....	129
CORCON (Core Control) .....	22, 72
DMACS0 (DMA Controller Status 0) .....	120
DMACS1 (DMA Controller Status 1) .....	122
DMAxCNT (DMA Channel x Transfer Count).....	119
DMAxCON (DMA Channel x Control).....	116
DMAxPAD (DMA Channel x Peripheral Address) ....	119
DMAxREQ (DMA Channel x IRQ Select) .....	117
DMAxSTA (DMA Channel x RAM Start Address A) .	118
DMAxSTB (DMA Channel x RAM Start Address B) .	118
DSADR (Most Recent DMA RAM Address) .....	123
I2CxCON (I2Cx Control) .....	163
I2CxMSK (I2Cx Slave Mode Address Mask).....	167
I2CxSTAT (I2Cx Status) .....	165
ICxCON (Input Capture x Control).....	146
IEC0 (Interrupt Enable Control 0) .....	84
IEC1 (Interrupt Enable Control 1) .....	86
IEC2 (Interrupt Enable Control 2) .....	88
IEC3 (Interrupt Enable Control 3) .....	90
IEC4 (Interrupt Enable Control 4) .....	91
IFS0 (Interrupt Flag Status 0) .....	76
IFS1 (Interrupt Flag Status 1) .....	78
IFS2 (Interrupt Flag Status 2) .....	80
IFS3 (Interrupt Flag Status 3) .....	82
IFS4 (Interrupt Flag Status 4) .....	83
INTCON1 (Interrupt Control 1) .....	73
INTCON2 (Interrupt Control 2) .....	75
IPC0 (Interrupt Priority Control 0) .....	92
IPC1 (Interrupt Priority Control 1) .....	93
IPC10 (Interrupt Priority Control 10) .....	102
IPC11 (Interrupt Priority Control 11) .....	103
IPC12 (Interrupt Priority Control 12) .....	104
IPC13 (Interrupt Priority Control 13) .....	105
IPC14 (Interrupt Priority Control 14) .....	106
IPC15 (Interrupt Priority Control 15) .....	107
IPC16 (Interrupt Priority Control 16) .....	108
IPC17 (Interrupt Priority Control 17) .....	109

# PIC24H

IPC2 (Interrupt Priority Control 2) .....	94
IPC3 (Interrupt Priority Control 3) .....	95
IPC4 (Interrupt Priority Control 4) .....	96
IPC5 (Interrupt Priority Control 5) .....	97
IPC6 (Interrupt Priority Control 6) .....	98
IPC7 (Interrupt Priority Control 7) .....	99
IPC8 (Interrupt Priority Control 8) .....	100
IPC9 (Interrupt Priority Control 9) .....	101
NVMCON (Flash Memory Control) .....	57
OCxCON (Output Compare x Control) .....	150
OSCCON (Oscillator Control) .....	128
OSCTUN (FRC Oscillator Tuning) .....	131
PLLFBF (PLL Feedback Divisor) .....	130
RCON (Reset Control) .....	62
SPIxCON1 (SPIx Control 1) .....	156
SPIxCON2 (SPIx Control 2) .....	157
SPIxSTAT (SPIx Status and Control) .....	155
SR (CPU Status) .....	20, 72
T1CON (Timer1 Control) .....	138
TxCON (T2CON, T4CON, T6CON or T8CON Control) .....	142
TyCON (T3CON, T5CON, T7CON or T9CON Control) .....	143
UxMODE (UARTx Mode) .....	172
UxSTA (UARTx Status and Control) .....	174
<b>Reset</b>	
Clock Source Selection .....	64
Special Function Register Reset States .....	65
Times .....	64
Reset Sequence .....	67
Resets .....	61
<b>S</b>	
Serial Peripheral Interface (SPI) .....	151
Setup for Continuous Output Pulse Generation .....	147
Setup for Single Output Pulse Generation .....	147
Software Simulator (MPLAB SIM) .....	236
Software Stack Pointer, Frame Pointer	
CALL Stack Frame .....	48
Special Features .....	221
SPI	
Master, Frame Master Connection .....	153
Master/Slave Connection .....	153
Slave, Frame Master Connection .....	154
Slave, Frame Slave Connection .....	154
SPI Module	
Operating Function Description .....	151
SPI1 Register Map .....	36
SPI2 Register Map .....	36
Symbols Used in Opcode Descriptions .....	228
System Control	
Register Map .....	47
<b>T</b>	
Temperature and Voltage Specifications	
AC .....	246
Timer1 .....	137
Timer2/3, Timer4/5, Timer6/7 and Timer8/9 .....	139
Timing Characteristics	
CLKO and I/O .....	249
Timing Diagrams	
10-bit A/D Conversion (CHPS = 01, SIMSAM = 0, ASAM = 0, SSRC = 000) .....	268
10-bit A/D Conversion (CHPS = 01, SIMSAM = 0, ASAM = 1, SSRC = 111, SAMC = 00001) .....	269
12-bit A/D Conversion (ASAM = 0, SSRC = 000) .....	271
CAN I/O .....	265
ECAN Bit .....	182
External Clock .....	247
I2Cx Bus Data (Master Mode) .....	261
I2Cx Bus Data (Slave Mode) .....	263
I2Cx Bus Start/Stop Bits (Master Mode) .....	261
I2Cx Bus Start/Stop Bits (Slave Mode) .....	263
Input Capture (CAPx) .....	254
OC/PWM .....	255
Output Compare (OCx) .....	254
Reset, Watchdog Timer, Oscillator Start-up Timer and Power-up Timer .....	250
SPIx Master Mode (CKE = 0) .....	255
SPIx Master Mode (CKE = 1) .....	256
SPIx Slave Mode (CKE = 0) .....	257
SPIx Slave Mode (CKE = 1) .....	259
Timer1, 2, 3, 4, 5, 6, 7, 8, 9 External Clock .....	252
<b>Timing Requirements</b>	
CLKO and I/O .....	249
External Clock .....	247
Input Capture .....	254
<b>Timing Specifications</b>	
10-bit A/D Conversion Requirements .....	270
12-bit A/D Conversion Requirements .....	272
CAN I/O Requirements .....	265
I2Cx Bus Data Requirements (Master Mode) .....	262
I2Cx Bus Data Requirements (Slave Mode) .....	264
Output Compare Requirements .....	254
PLL Clock .....	248
Reset, Watchdog Timer, Oscillator Start-up Timer, Power-up Timer and Brown-out Reset Requirements .....	251
Simple OC/PWM Mode Requirements .....	255
SPIx Master Mode (CKE = 0) Requirements .....	256
SPIx Master Mode (CKE = 1) Requirements .....	257
SPIx Slave Mode (CKE = 0) Requirements .....	258
SPIx Slave Mode (CKE = 1) Requirements .....	260
Timer1 External Clock Requirements .....	252
Timer2, Timer4, Timer6 and Timer8 External Clock Requirements .....	253
Timer3, Timer5, Timer7 and Timer9 External Clock Requirements .....	253
<b>U</b>	
UART	
Baud Rate	
Generator (BRG) .....	170
Break and Sync Transmit Sequence .....	171
Flow Control Using UxCTS and UxRTS Pins .....	171
Receiving in 8-bit or 9-bit Data Mode .....	171
Transmitting in 8-bit Data Mode .....	171
Transmitting in 9-bit Data Mode .....	171
UART Module	
UART1 Register Map .....	36
UART2 Register Map .....	36
<b>V</b>	
Voltage Regulator (On-Chip) .....	223
<b>W</b>	
Watchdog Timer (WDT) .....	221, 224
Programming Considerations .....	224
WWW Address .....	283
WWW, On-Line Support .....	11

## THE MICROCHIP WEB SITE

Microchip provides online support via our WWW site at [www.microchip.com](http://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](http://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

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>**

# PIC24H

---

---

## 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: PIC24H

Literature Number: DS70175C

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.

<p style="text-align: center;"><b>PIC 24 HJ 256 GP6 10 I ! / PT - XXX</b></p> <p><b>Microchip Trademark</b> _____</p> <p><b>Architecture</b> _____</p> <p><b>Flash Memory Family</b> _____</p> <p><b>Program Memory Size (KB)</b> _____</p> <p><b>Product Group</b> _____</p> <p><b>Pin Count</b> _____</p> <p><b>Tape and Reel Flag (if applicable)</b> _____</p> <p><b>Temperature Range</b> _____</p> <p><b>Package</b> _____</p> <p><b>Pattern</b> _____</p>	<p><b>Examples:</b></p> <p>a) PIC24HJ256GP210I/PT: General-purpose PIC24H, 256 KB program memory, 100-pin, Industrial temp., TQFP package.</p> <p>b) PIC24HJ64GP506I/PT-ES: General-purpose PIC24H, 64 KB program memory, 64-pin, Industrial temp., TQFP package, Engineering Sample.</p>																																																				
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 15%;">Architecture</td> <td style="width: 10%;">24</td> <td style="width: 10%;">=</td> <td>16-bit Microcontroller</td> </tr> <tr> <td>Flash Memory Family</td> <td>HJ</td> <td>=</td> <td>Flash program memory, 3.3V, High-speed</td> </tr> <tr> <td>Product Group</td> <td>GP2</td> <td>=</td> <td>General purpose family</td> </tr> <tr> <td></td> <td>GP3</td> <td>=</td> <td>General purpose family</td> </tr> <tr> <td></td> <td>GP5</td> <td>=</td> <td>General purpose family</td> </tr> <tr> <td></td> <td>GP6</td> <td>=</td> <td>General purpose family</td> </tr> <tr> <td>Pin Count</td> <td>06</td> <td>=</td> <td>64-pin</td> </tr> <tr> <td></td> <td>10</td> <td>=</td> <td>100-pin</td> </tr> <tr> <td>Temperature Range</td> <td>I</td> <td>=</td> <td>-40°C to +85°C (Industrial)</td> </tr> <tr> <td>Package</td> <td>PT</td> <td>=</td> <td>10x10 or 12x12 mmTQFP (Thin Quad Flatpack)</td> </tr> <tr> <td></td> <td>PF</td> <td>=</td> <td>14x14 mmTQFP (Thin Quad Flatpack)</td> </tr> <tr> <td>Pattern</td> <td colspan="3">Three-digit QTP, SQTP, Code or Special Requirements (blank otherwise)</td> </tr> <tr> <td></td> <td>ES</td> <td>=</td> <td>Engineering Sample</td> </tr> </table>		Architecture	24	=	16-bit Microcontroller	Flash Memory Family	HJ	=	Flash program memory, 3.3V, High-speed	Product Group	GP2	=	General purpose family		GP3	=	General purpose family		GP5	=	General purpose family		GP6	=	General purpose family	Pin Count	06	=	64-pin		10	=	100-pin	Temperature Range	I	=	-40°C to +85°C (Industrial)	Package	PT	=	10x10 or 12x12 mmTQFP (Thin Quad Flatpack)		PF	=	14x14 mmTQFP (Thin Quad Flatpack)	Pattern	Three-digit QTP, SQTP, Code or Special Requirements (blank otherwise)				ES	=	Engineering Sample
Architecture	24	=	16-bit Microcontroller																																																		
Flash Memory Family	HJ	=	Flash program memory, 3.3V, High-speed																																																		
Product Group	GP2	=	General purpose family																																																		
	GP3	=	General purpose family																																																		
	GP5	=	General purpose family																																																		
	GP6	=	General purpose family																																																		
Pin Count	06	=	64-pin																																																		
	10	=	100-pin																																																		
Temperature Range	I	=	-40°C to +85°C (Industrial)																																																		
Package	PT	=	10x10 or 12x12 mmTQFP (Thin Quad Flatpack)																																																		
	PF	=	14x14 mmTQFP (Thin Quad Flatpack)																																																		
Pattern	Three-digit QTP, SQTP, Code or Special Requirements (blank otherwise)																																																				
	ES	=	Engineering Sample																																																		



---

---

## 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](http://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