PIC24HJ32GP202/204 and PIC24HJ16GP304 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in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates.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, KEELOQ logo, microID, MPLAB, PIC, PICmicro, PICSTART, PRO MATE, rfPIC and SmartShunt are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

AmpLab, FilterLab, Linear Active Thermistor, 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, CodeGuard, dsPICDEM, dsPICDEM.net, dsPICworks, dsSPEAK, ECAN, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, Mindi, MiWi, MPASM, MPLAB Certified logo, MPLIB, MPLINK, PICkit, PICDEM, PICDEM.net, PICLAB, PICtail, PowerCal, PowerInfo, PowerMate, PowerTool, REAL ICE, rfLAB, 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.

© 2007, 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 design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

Downloaded from Elcodis.com electronic components distributor

PIC24HJ32GP202/204 AND PIC24HJ16GP304

High-Performance, 16-bit Microcontrollers

Operating Range:

- Up to 40 MIPS operation (@ 3.0-3.6V):
  - Industrial temperature range (-40°C to +85°C)
  - Extended temperature range (-40°C to +125°C)

High-Performance 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 addressing modes
- Software stack
- 16 x 16 multiply operations
- 32/16 and 16/16 divide operations
- Up to ±16-bit shifts for up to 40-bit data

Interrupt Controller:

- 5-cycle latency
- 118 interrupt vectors
- Up to 21 available interrupt sources
- Up to 3 external interrupts
- 7 programmable priority levels
- 4 processor exceptions

On-Chip Flash and SRAM:

- Flash program memory (up to 32 Kbytes)
- Data SRAM (2 Kbytes)
- Boot and General Security for Program Flash

Digital I/O:

- Peripheral Pin Select Functionality
- Up to 35 programmable digital I/O pins
- Wake-up/Interrupt-on-Change for up to 21 pins
- Output pins can drive from 3.0V to 3.6V
- Up to 5V output with open drain configuration
- All digital input pins are 5V tolerant
- 4 mA sink on all I/O pins

System Management:

- Flexible clock options:
  - External, crystal, resonator, internal RC
  - Fully integrated Phase-Locked Loop (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:

- Timer/Counters, up to three 16-bit timers:
  - Can pair up to make one 32-bit timer
  - 1 timer runs as Real-Time Clock with external 32.768 kHz oscillator
  - Programmable prescaler
- Input Capture (up to 4 channels):
  - Capture on up, down or both edges
  - 16-bit capture input functions
  - 4-deep FIFO on each capture
- Output Compare (up to 2 channels):
  - Single or Dual 16-Bit Compare mode
  - 16-bit Glitchless PWM Mode

Communication Modules:

- 4-wire SPI
  - Framing supports I/O interface to simple codecs
  - Supports 8-bit and 16-bit data
  - Supports all serial clock formats and sampling modes
- I²C™
  - Full Multi-Master Slave mode support
  - 7-bit and 10-bit addressing
  - Bus collision detection and arbitration
  - Integrated signal conditioning
  - Slave address masking
- UART
  - 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

Analog-to-Digital Converters (ADCs):

- 10-bit, 1.1 Msps or 12-bit, 500 Ksps conversion:
  - 2 and 4 simultaneous samples (10-bit ADC)
  - Up to 13 input channels with auto-scanning
  - Conversion start can be manual or synchronized with 1 of 4 trigger sources
  - Conversion possible in Sleep mode
  - ±2 LSb max integral nonlinearity
  - ±1 LSb max differential nonlinearity

CMOS Flash Technology:

- Low-power, high-speed Flash technology
- Fully static design
- 3.3V (±10%) operating voltage
- Industrial and extended temperature
- Low-power consumption

Packaging:

- 28-pin SDIP/SOIC/QFN-S
- 44-pin QFN/TQFP

Note: See the device variant tables for exact peripheral features per device.

PIC24HJ32GP202/204 and PIC24HJ16GP304 Product Families

The device names, pin counts, memory sizes and peripheral availability of each family are listed below, followed by their pinout diagrams.

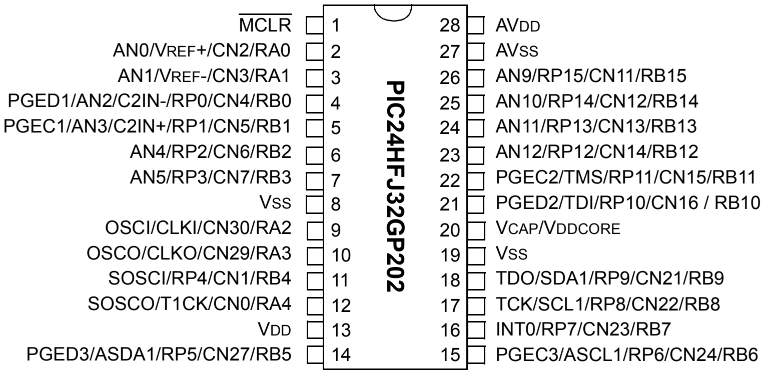
TABLE 1: PIC24HJ32GP202/204 AND PIC24HJ16GP304 CONTROLLER FAMILIES

Device	Pins	Program Flash Memory (Kbyte)	RAM	Remappable Peripherals						10-Bit/12-Bit ADC	I^{2}C™	I/O Pins (Max)	Packages
				Remappable Pins	16-bit Timer	Input Capture	Output Compare Std. PWM	UART	SPI				
PIC24HJ32GP202	28	32	2	16	3$^{(1)}$	4	2	1	1	1 ADC, 10 ch	1	21	SDIP SOIC QFN-S
PIC24HJ32GP204	44	32	2	26	3$^{(1)}$	4	2	1	1	1 ADC, 13 ch	1	35	QFN TQFP
PIC24HJ16GP304	44	16	2	26	3$^{(1)}$	4	2	1	1	1 ADC, 13 ch	1	35	QFN TQFP

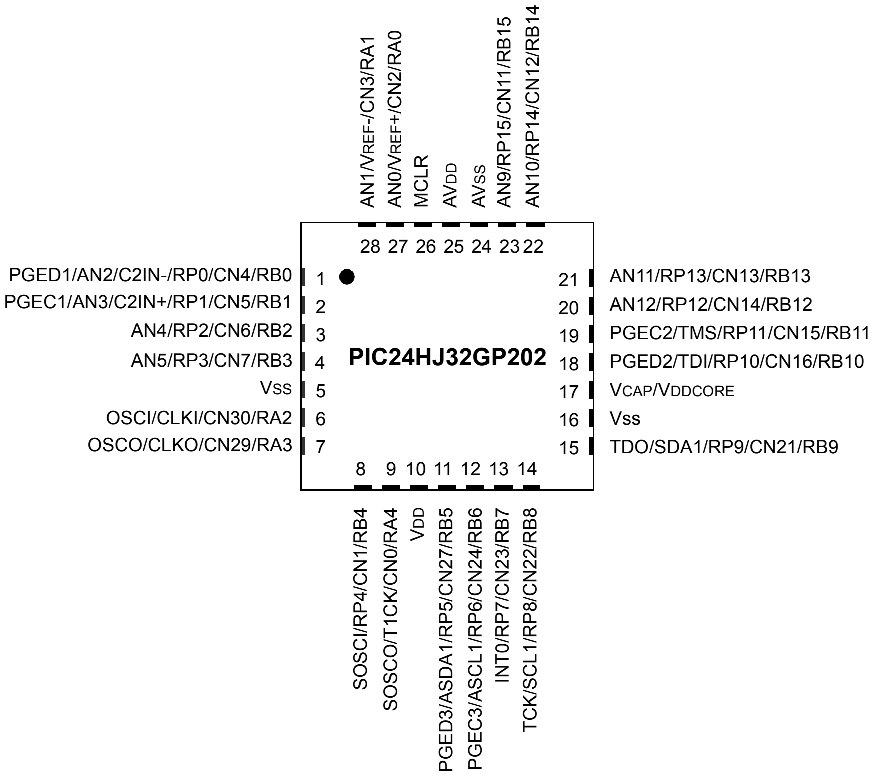
Note 1: Only 2 out of 3 timers are Remappable

Pin Diagrams

28-Pin SDIP, SOIC

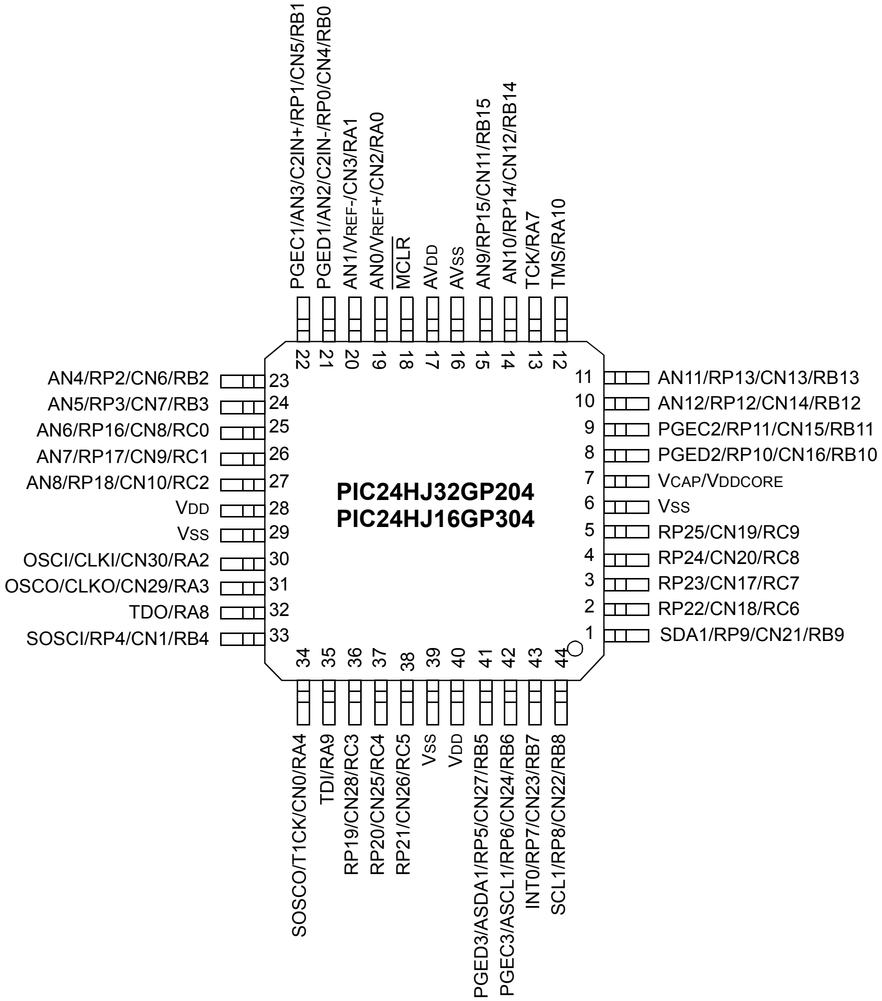


28-Pin QFN-S



Pin Diagrams (Continued)

44-Pin TQFP



Pin Diagrams (Continued)

44-Pin TQFP

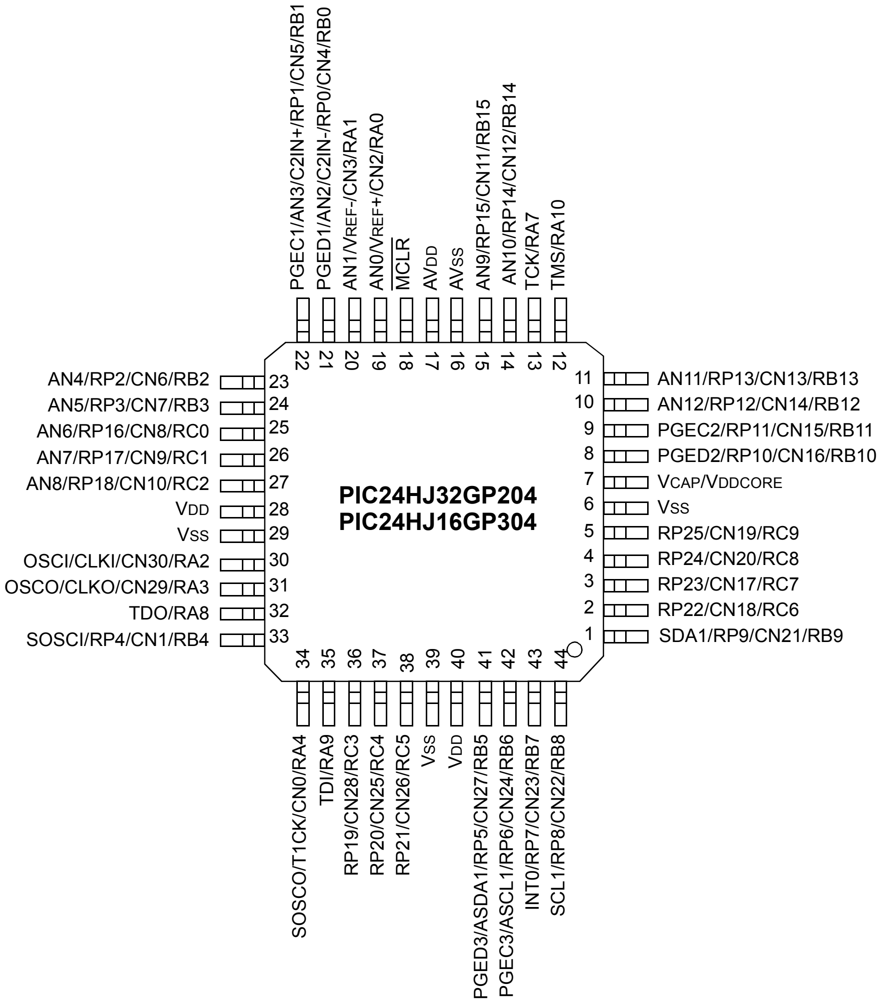


Table of Contents

TO OUR VALUED CUSTOMERS

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

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

Most Current Data Sheet

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

http://www.microchip.com

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

Errata

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

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

- Microchip's Worldwide Web site; http://www.microchip.com
- Your local Microchip sales office (see last page)

When contacting a sales office, please specify which device, revision of silicon and data sheet (include literature number) you are using.

Customer Notification System

Register on our web site at **www.microchip.com** to receive the most current information on all of our products.

NOTES:

1.0 DEVICE OVERVIEW

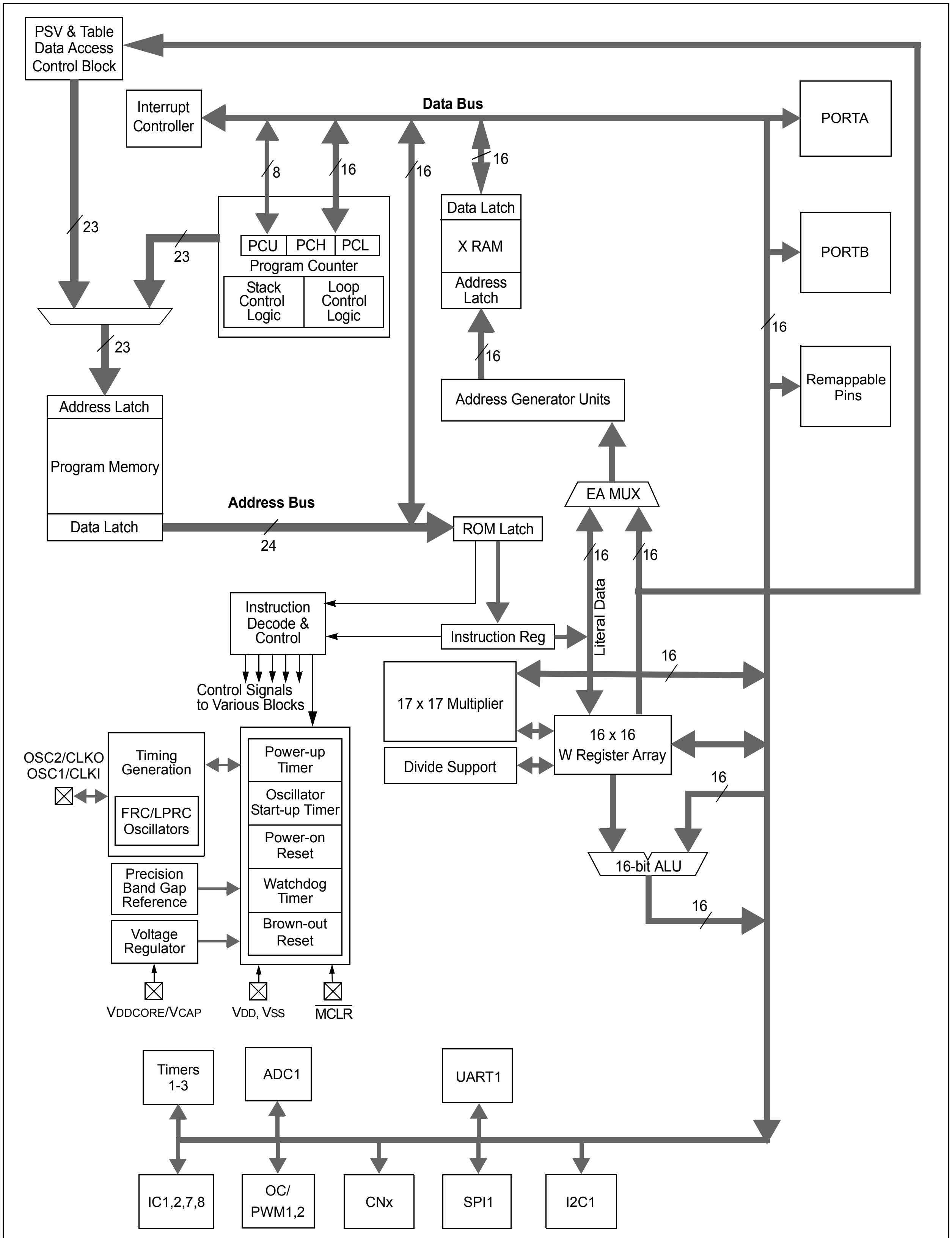
> **Note:** This data sheet summarizes the features of the PIC24HJ32GP202/204 and PIC24HJ16GP304 devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the *“PIC24H Family Reference Manual”*.

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

- PIC24HJ32GP202
- PIC24HJ32GP204
- PIC24HJ16GP304

Figure 1-1 shows a general block diagram of the core and peripheral modules in the PIC24HJ32GP202/204 and PIC24HJ16GP304 family of devices. Table 1-1 lists the functions of the various pins shown in the pinout diagrams.

FIGURE 1-1: PIC24HJ32GP202/204 AND PIC24HJ16GP304 BLOCK DIAGRAM



Note: Not all pins or features are implemented on all device pinout configurations. See pinout diagrams for the specific pins and features present on each device.

TABLE 1-1: PINOUT I/O DESCRIPTIONS

Pin Name	Pin Type	Buffer Type	Description
AN0-AN12	I	Analog	Analog input channels.
CLKI CLKO	I O	ST/CMOS —	External clock source input. Always associated with OSC1 pin function. 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.
OSC1 OSC2	I I/O	ST/CMOS —	Oscillator crystal input. ST buffer when configured in RC mode; CMOS otherwise. Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode. Optionally functions as CLKO in RC and EC modes.
SOSCI SOSCO	I O	ST/CMOS —	32.768 kHz low-power oscillator crystal input; CMOS otherwise. 32.768 kHz low-power oscillator crystal output.
CN0-CN30	I	ST	Change notification inputs. Can be software programmed for internal weak pull-ups on all inputs.
IC1-IC2 IC7-IC8	I	ST	Capture inputs 1/2 Capture inputs 7/8
OCFA OC1-OC2	I O	ST —	Compare Fault A input (for Compare Channels 1 and 2). Compare outputs 1 through 2.
INT0 INT1 INT2	I I I	ST ST ST	External interrupt 0. External interrupt 1. External interrupt 2.
RA0-RA4 RA7-RA15	I/O	ST	PORTA is a bidirectional I/O port.
RB0-RB15	I/O	ST	PORTB is a bidirectional I/O port.
RC0-RC9	I/O	ST	PORTC is a bidirectional I/O port.
T1CK T2CK T3CK	I I I	ST ST ST	Timer1 external clock input. Timer2 external clock input. Timer3 external clock input.
$\overline{\text{U1CTS}}$ $\overline{\text{U1RTS}}$ U1RX U1TX	I O I O	ST — ST —	UART1 clear to send. UART1 ready to send. UART1 receive. UART1 transmit.
SCK1 SDI1 SDO1 $\overline{\text{SS1}}$	I/O I O I/O	ST ST — ST	Synchronous serial clock input/output for SPI1. SPI1 data in. SPI1 data out. SPI1 slave synchronization or frame pulse I/O.
SCL1 SDA1 ASCL1 ASDA1	I/O I/O I/O I/O	ST ST ST ST	Synchronous serial clock input/output for I2C1. Synchronous serial data input/output for I2C1. Alternate synchronous serial clock input/output for I2C1. Alternate synchronous serial data input/output for I2C1.
TMS TCK TDI TDO	I I I O	ST ST ST —	JTAG Test mode select pin. JTAG test clock input pin. JTAG test data input pin. JTAG test data output pin.
PGD1/EMUD1 PGC1/EMUC1 PGD2/EMUD2 PGC2/EMUC2 PGD3/EMUD3 PGC3/EMUC3	I/O I I/O I I/O I	ST ST ST ST ST ST	Data I/O pin for programming/debugging communication channel 1. Clock input pin for programming/debugging communication channel 1. Data I/O pin for programming/debugging communication channel 2. Clock input pin for programming/debugging communication channel 2. Data I/O pin for programming/debugging communication channel 3. Clock input pin for programming/debugging communication channel 3.
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; O = Output
ST = Schmitt Trigger input with CMOS levels; I = Input; P = Power

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

Pin Name	Pin Type	Buffer Type	Description
AV_{DD}	P	P	Positive supply for analog modules.
$\overline{MCLR}$	I/P	ST	Master Clear (Reset) input. This pin is an active-low Reset to the device.
AV_{SS}	P	P	Ground reference for analog modules.
V_{DD}	P	—	Positive supply for peripheral logic and I/O pins.

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

2.0 CPU

> **Note:** This data sheet summarizes the features of this group of PIC24HJ32GP202/204 and PIC24HJ16GP304 devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the *"PIC24H Family Reference Manual"*.

The PIC24HJ32GP202/204 and PIC24HJ16GP304 CPU modules have 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 PIC24HJ32GP202/204 and PIC24HJ16GP304 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 PIC24HJ32GP202/204 and PIC24HJ16GP304 instruction set includes many addressing modes and is designed for optimum C compiler efficiency. For most instructions, the PIC24HJ32GP202/204 and PIC24HJ16GP304 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 PIC24HJ32GP202/204 and PIC24HJ16GP304 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 this may be used as general purpose RAM.

2.2 Special MCU Features

The PIC24HJ32GP202/204 and PIC24HJ16GP304 feature 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 PIC24HJ32GP202/204 and PIC24HJ16GP304 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.

FIGURE 2-1: PIC24HJ32GP202/204 AND PIC24HJ16GP304 CPU CORE BLOCK DIAGRAM

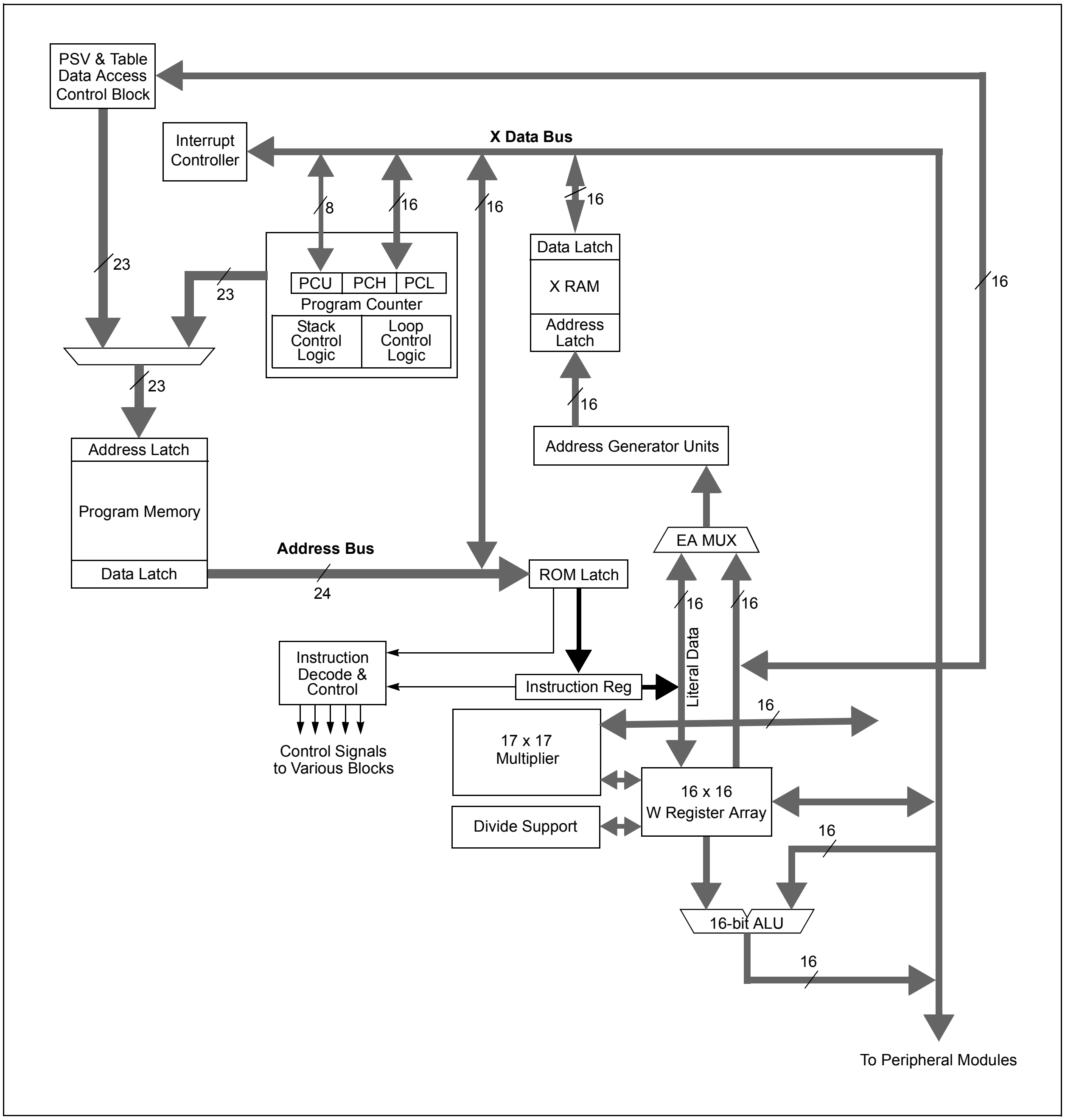
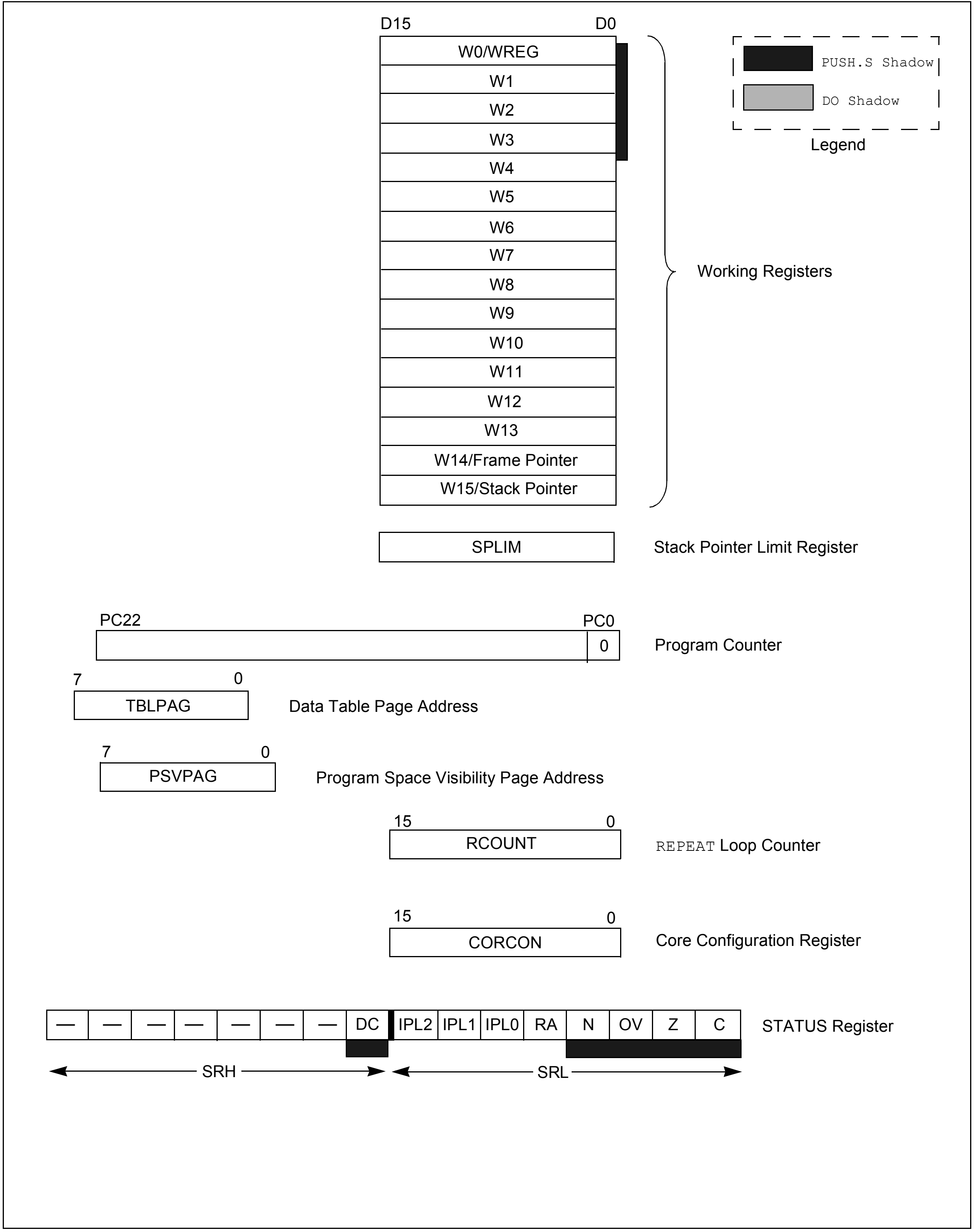


FIGURE 2-2: PIC24HJ32GP202/204 AND PIC24HJ16GP304 PROGRAMMER'S MODEL



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(1)	R/W-0(2)	R/W-0(2)	R-0	R/W-0	R/W-0	R/W-0	R/W-0
IPL<2:0>(2)			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(2)

111 = CPU Interrupt Priority Level is 7 (15), user interrupts disabled
110 = CPU Interrupt Priority Level is 6 (14)
101 = CPU Interrupt Priority Level is 5 (13)
100 = CPU Interrupt Priority Level is 4 (12)
011 = CPU Interrupt Priority Level is 3 (11)
010 = CPU Interrupt Priority Level is 2 (10)
001 = CPU Interrupt Priority Level is 1 (9)
000 = CPU Interrupt Priority Level is 0 (8)

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)

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>).

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(1)	PSV	—	—
bit 7							bit 0

Legend:	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(1) 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 PIC24HJ32GP202/204 and PIC24HJ16GP304 Arithmetic Logic Unit (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. 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 depending on the operation. The C and DC Status bits operate as $\overline{\text{Borrow}}$ and $\overline{\text{Digit Borrow}}$ 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 more information on the SR bits affected by each instruction.

The PIC24HJ32GP202/204 and PIC24HJ16GP304 CPU incorporates hardware support for both multiplication and division. This includes a dedicated hardware multiplier and a 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:

- 16-bit x 16-bit signed
- 16-bit x 16-bit unsigned
- 16-bit signed x 5-bit (literal) unsigned
- 16-bit unsigned x 16-bit unsigned
- 16-bit unsigned x 5-bit (literal) unsigned
- 16-bit unsigned x 16-bit signed
- 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. A 16-bit signed and unsigned `DIV` instructions can specify any W register for both the 16-bit divisor (Wn) and any W register (aligned) pair (W(m + 1):Wm) for the 32-bit dividend. The divide algorithm takes one cycle per bit of divisor, so both 32-bit/16-bit and 16-bit/16-bit instructions take the same number of cycles to execute.

2.4.3 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. and a negative value shifts the operand left. A value of '`0`' does not modify the operand.

3.0 MEMORY ORGANIZATION

Note: This data sheet summarizes the features of the PIC24HJ32GP202/204 and PIC24HJ16GP304 devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the *"PIC24H Family Reference Manual"*.

The PIC24HJ32GP202/204 and PIC24HJ16GP304 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.

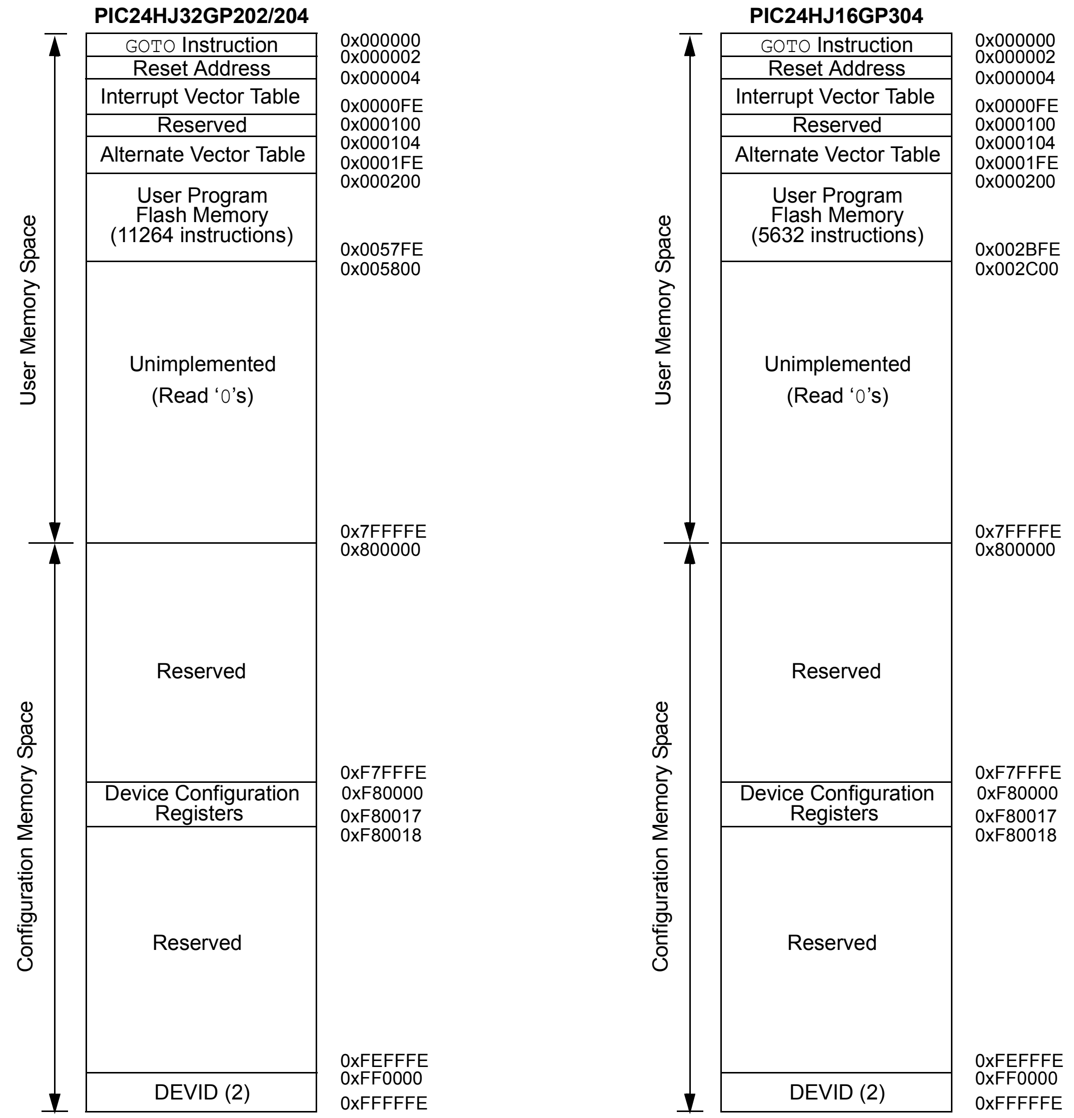
3.1 Program Address Space

The program address memory space of the PIC24HJ32GP202/204 and PIC24HJ16GP304 devices is 4M instructions. The space is addressable by a 24-bit value derived either from 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 application access to the program memory space is restricted to the lower half of the address range (0x000000 to 0x7FFFFF). 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.

The memory maps for the PIC24HJ32GP202/204 and PIC24HJ16GP304 devices are shown in Figure 3-1.

FIGURE 3-1: PROGRAM MEMORY FOR PIC24HJ32GP202/204 AND PIC24HJ16GP304 DEVICES



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 (See Figure 3-2).

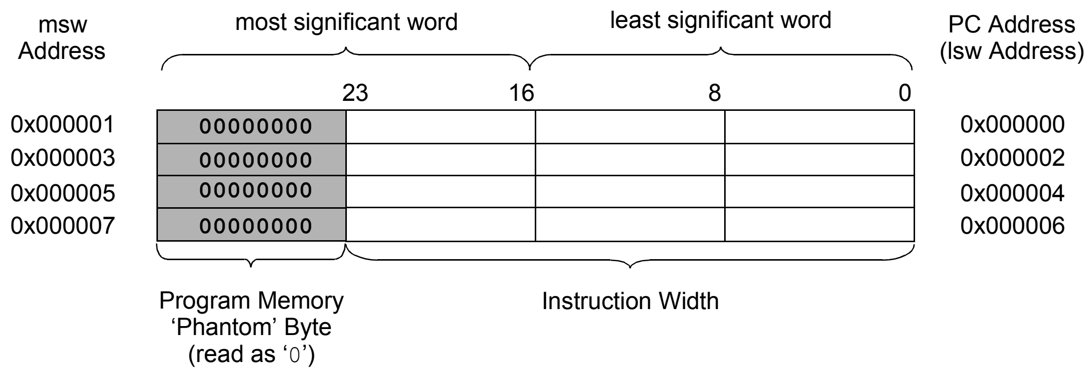
Program memory addresses are always word-aligned on the lower word, and addresses are incremented or decremented by two during code execution. This arrangement provides compatibility with data memory space addressing and makes data in the program memory space accessible.

3.1.2 INTERRUPT AND TRAP VECTORS

All PIC24HJ32GP202/204 and PIC24HJ16GP304 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 application at 0x000000, with the actual address for the start of code at 0x000002.

PIC24HJ32GP202/204 and PIC24HJ16GP304 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). **Section 6.1 “Interrupt Vector Table”** provides a more detailed discussion of the interrupt vector tables.

FIGURE 3-2: PROGRAM MEMORY ORGANIZATION



3.2 Data Address Space

The PIC24HJ32GP202/204 and PIC24HJ16GP304 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. The data memory maps is shown in Figure 3-3.

All Effective Addresses (EAs) in the data memory space are 16 bits wide and point to the 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<15> = `0`) is used for implemented memory addresses, while the upper half (EA<15> = `1`) is reserved for the Program Space Visibility area (see **Section 3.4.3 "Reading Data From Program Memory Using Program Space Visibility"**).

PIC24HJ32GP202/204 and PIC24HJ16GP304 devices implement up to 30 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 (LSBs) of each word have even addresses, while the Most Significant Bytes (MSBs) have odd addresses.

3.2.2 DATA MEMORY ORGANIZATION AND ALIGNMENT

To maintain backward compatibility with PIC® devices and improve data space memory usage efficiency, the PIC24HJ32GP202/204 and PIC24HJ16GP304 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 LSB of any EA to determine which byte to select. The selected byte is placed onto the LSB of the data path. That is, data memory and registers are organized as two parallel byte-wide entities with shared (word) address decode, but separate write lines. Data byte writes only write to the corresponding side of the array or register that 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 when 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 the instruction occurred on a write, the instruction is executed but the write does not occur. In either case, a trap is then executed, allowing the system and/or user application to examine the machine state prior to execution of the address Fault.

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

A sign-extend instruction (`SE`) is provided to allow users to translate 8-bit signed data to 16-bit signed values. Alternatively, for 16-bit unsigned data, user applications can clear the MSB 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 PIC24HJ32GP202/204 and PIC24HJ16GP304 core and peripheral modules to control 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 3-21.

Note:	The actual set of peripheral features and interrupts varies by the device. 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 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.

FIGURE 3-3: DATA MEMORY MAP FOR PIC24HJ32GP202/204 AND PIC24HJ16GP304 DEVICES WITH 2 KB RAM

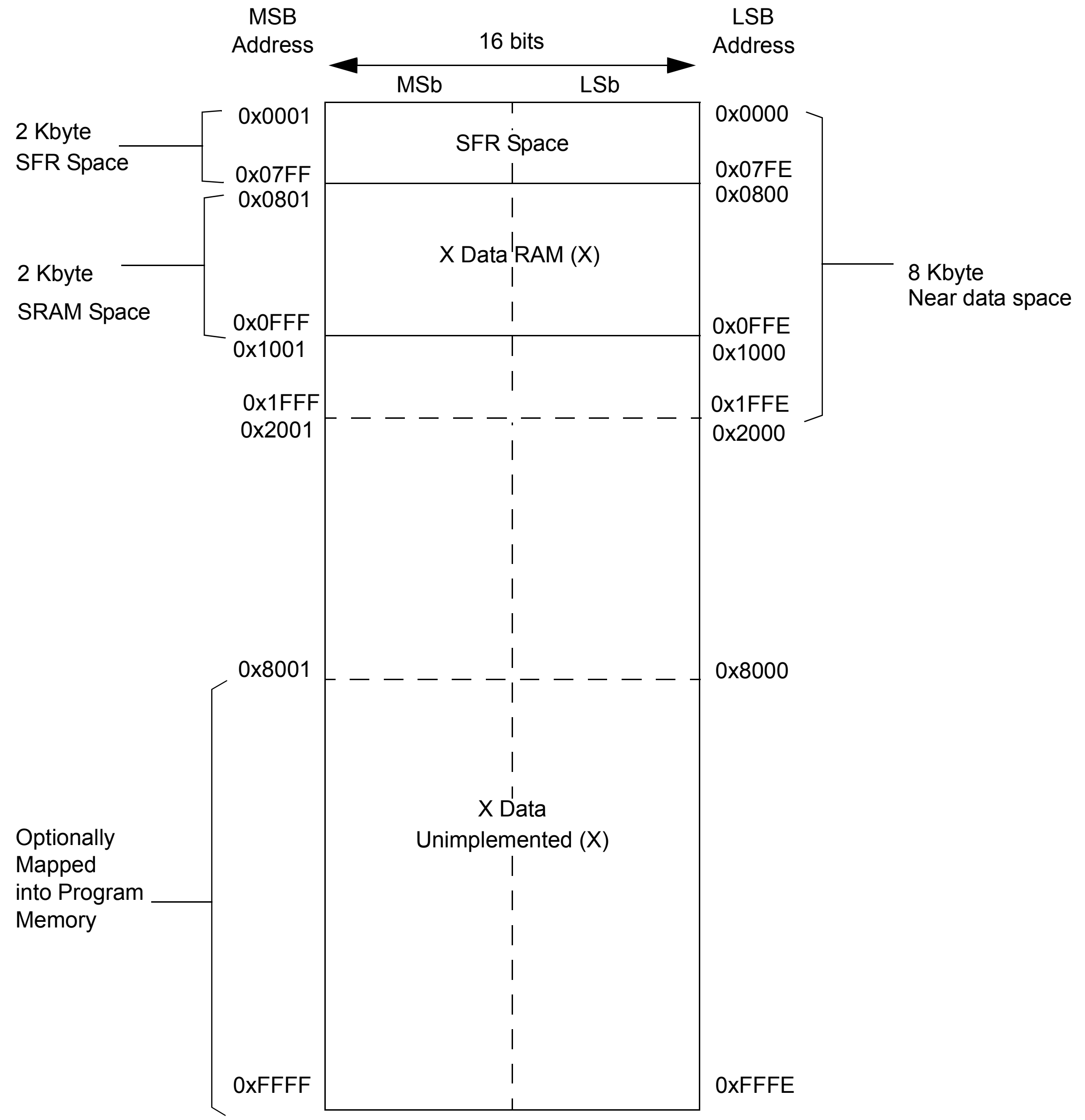


TABLE 3-1: CPU CORE REGISTERS MAP

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																xxxx
PCL	002E	Program Counter Low Word Register																0000
PCH	0030	—	—	—	—	—	—	—	—	Program Counter High Byte Register								0000
TBLPAG	0032	—	—	—	—	—	—	—	—	Table Page Address Pointer Register								0000
PSVPAG	0034	—	—	—	—	—	—	—	—	Program Memory Visibility Page Address Pointer Register								0000
RCOUNT	0036	Repeat Loop Counter Register																xxxx
SR	0042	—	—	—	—	—	—	—	DC	IPL2	IPL1	IPL0	RA	N	OV	Z	C	0000
CORCON	0044	—	—	—	—	—	—	—	—	—	—	—	—	IPL3	PSV	—	—	0000
DISICNT	0052	—	—	Disable Interrupts Counter Register														xxxx

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

TABLE 3-2: CHANGE NOTIFICATION REGISTER MAP FOR PIC24HJ32GP202

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	—	—	—	CN7IE	CN6IE	CN5IE	CN4IE	CN3IE	CN2IE	CN1IE	CN0IE	0000
CNEN2	0062	—	CN30IE	CN29IE	—	CN27IE	—	—	CN24IE	CN23IE	CN22IE	CN21IE	—	—	—	—	CN16IE	0000
CNPU1	0068	CN15PUE	CN14PUE	CN13PUE	CN12PUE	CN11PUE	—	—	—	CN7PUE	CN6PUE	CN5PUE	CN4PUE	CN3PUE	CN2PUE	CN1PUE	CN0PUE	0000
CNPU2	006A	—	CN30PUE	CN29PUE	—	CN27PUE	—	—	CN24PUE	CN23PUE	CN22PUE	CN21PUE	—	—	—	—	CN16PUE	0000

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

TABLE 3-3: CHANGE NOTIFICATION REGISTER MAP FOR PIC24HJ32GP204 AND PIC24HJ16GP304

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	—	CN30IE	CN29IE	CN28IE	CN27IE	CN26IE	CN25IE	CN24IE	CN23IE	CN22IE	CN21IE	CN20IE	CN19IE	CN18IE	CN17IE	CN16IE	0000
CNPU1	0068	CN15PUE	CN14PUE	CN13PUE	CN12PUE	CN11PUE	CN10PUE	CN9PUE	CN8PUE	CN7PUE	CN6PUE	CN5PUE	CN4PUE	CN3PUE	CN2PUE	CN1PUE	CN0PUE	0000
CNPU2	006A	—	CN30PUE	CN29PUE	CN28PUE	CN27PUE	CN26PUE	CN25PUE	CN24PUE	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 3-4: 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	—	—	—	—	—	—	—	—	DIV0ERR	—	MATHERR	ADDRERR	STKERR	OSCFAIL	—	0000
INTCON2	0082	ALTIVT	DISI	—	—	—	—	—	—	—	—	—	—	—	INT2EP	INT1EP	INT0EP	0000
IFS0	0084	—	—	AD1IF	U1TXIF	U1RXIF	SPI1IF	SPI1EIF	T3IF	T2IF	OC2IF	IC2IF	—	T1IF	OC1IF	IC1IF	INT0IF	0000
IFS1	0086	—	—	INT2IF	—	—	—	—	—	IC8IF	IC7IF	—	INT1IF	CNIF	—	MI2C1IF	SI2C1IF	0000
IFS4	008C	—	—	—	—	—	—	—	—	—	—	—	—	—	—	U1EIF	—	0000
IEC0	0094	—	—	AD1IE	U1TXIE	U1RXIE	SPI1IE	SPI1EIE	T3IE	T2IE	OC2IE	IC2IE	—	T1IE	OC1IE	IC1IE	INT0IE	0000
IEC1	0096	—	—	INT2IE	—	—	—	—	—	IC8IE	IC7IE	—	INT1IE	CNIE	—	MI2C1IE	SI2C1IE	0000
IEC4	009C	—	—	—	—	—	—	—	—	—	—	—	—	—	—	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>			—	—	—	—	4444
IPC2	00A8	—	U1RXIP<2:0>			—	SPI1IP<2:0>			—	SPI1EIP<2:0>			—	T3IP<2:0>			4444
IPC3	00AA	—	—	—	—	—	—	—	—	—	AD1IP<2:0>			—	U1TXIP<2:0>			4444
IPC4	00AC	—	CNIP<2:0>			—	—	—	—	—	MI2C1IP<2:0>			—	SI2C1IP<2:0>			4444
IPC5	00AE	—	IC8IP<2:0>			—	IC7IP<2:0>			—	—	—	—	—	INT1IP<2:0>			4444
IPC7	00B2	—	—	—	—	—	—	—	—	—	INT2IP<2:0>			—	—	—	—	4444
IPC16	00C4	—	—	—	—	—	—	—	—	—	U1EIP<2:0>			—	—	—	—	4444
INTTREG	00E0	—	—	—	—	ILR<3:0>>				—	VECNUM<6:0>							4444

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

TABLE 3-5: 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																xxxx
PR1	0102	Period Register 1																FFFF
T1CON	0104	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS<1:0>		—	TSYNC	TCS	—	0000
TMR2	0106	Timer2 Register																xxxx
TMR3HLD	0108	Timer3 Holding Register (for 32-bit timer operations only)																xxxx
TMR3	010A	Timer3 Register																xxxx
PR2	010C	Period Register 2																FFFF
PR3	010E	Period Register 3																FFFF
T2CON	0110	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS<1:0>		T32	—	TCS	—	0000
T3CON	0112	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 3-6: 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																xxxx
IC1CON	0142	—	—	ICSIDL	—	—	—	—	—	ICTMR	ICI<1:0>		ICOV	ICBNE	ICM<2:0>			0000
IC2BUF	0144	Input 2 Capture Register																xxxx
IC2CON	0146	—	—	ICSIDL	—	—	—	—	—	ICTMR	ICI<1:0>		ICOV	ICBNE	ICM<2:0>			0000
IC7BUF	0158	Input 7 Capture Register																xxxx
IC7CON	015A	—	—	ICSIDL	—	—	—	—	—	ICTMR	ICI<1:0>		ICOV	ICBNE	ICM<2:0>			0000
IC8BUF	015C	Input 8Capture Register																xxxx
IC8CON	015E	—	—	ICSIDL	—	—	—	—	—	ICTMR	ICI<1:0>		ICOV	ICBNE	ICM<2:0>			0000

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

TABLE 3-7: 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																xxxx
OC1R	0182	Output Compare 1 Register																xxxx
OC1CON	0184	—	—	OCSIDL	—	—	—	—	—	—	—	—	OCFLT	OCTSEL	OCM<2:0>			0000
OC2RS	0186	Output Compare 2 Secondary Register																xxxx
OC2R	0188	Output Compare 2 Register																xxxx
OC2CON	018A	—	—	OCSIDL	—	—	—	—	—	—	—	—	OCFLT	OCTSEL	OCM<2:0>			0000

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

TABLE 3-8: 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
I2C1CON	0206	I2CEN	—	I2CSIDL	SCLREL	IPMIEN	A10M	DISSLW	SMEN	GCEN	STREN	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	1000
I2C1STAT	0208	ACKSTAT	TRSTAT	—	—	—	BCL	GCSTAT	ADD10	IWCOL	I2COV	D_A	P	S	R_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 3-9: 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	—	—	—	—	—	—	—	UART Transmit Register									xxxx
U1RXREG	0226	—	—	—	—	—	—	—	UART Receive Register									0000
U1BRG	0228	Baud Rate Generator Prescaler																0000

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

TABLE 3-10: 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	—	SPISIDL	—	—	—	—	—	—	SPIROV	—	—	—	—	SPITBF	SPIRBF	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	SPI1 Transmit and Receive Buffer Register																0000

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

TABLE 3-11: PERIPHERAL PIN SELECT INPUT 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
RPINR0	0680	—	—	—	INT1R<4:0>					—	—	—	—	—	—	—	—	1F00
RPINR1	0682	—	—	—	—	—	—	—	—	—	—	—	INT2R<4:0>					001F
RPINR3	0686	—	—	—	T3CKR<4:0>					—	—	—	T2CKR<4:0>					1F1F
RPINR7	068E	—	—	—	IC2R<4:0>					—	—	—	IC1R<4:0>					1F1F
RPINR10	0694	—	—	—	IC8R<4:0>					—	—	—	IC7R<4:0>					1F1F
RPINR11	0696	—	—	—	—	—	—	—	—	—	—	—	OCFAR<4:0>					001F
RPINR18	06A4	—	—	—	U1CTSR<4:0>					—	—	—	U1RX<R4:0>					1F1F
RPINR20	06A8	—	—	—	SCK1R<4:0>					—	—	—	SDI1R<4:0>					1F1F
RPINR21	06AA	—	—	—	—	—	—	—	—	—	—	—	SS1R<4:0>					001F

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

TABLE 3-12: PERIPHERAL PIN SELECT OUTPUT REGISTER MAP FOR PIC24HJ32GP202

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
RPOR0	06C0	—	—	—	RP1R<4:0>					—	—	—	RP0R<4:0>					0000
RPOR1	06C2	—	—	—	RP3R<4:0>					—	—	—	RP2R<4:0>					0000
RPOR2	06C4	—	—	—	RP5R<4:0>					—	—	—	RP4R<4:0>					0000
RPOR3	06C6	—	—	—	RP7R<4:0>					—	—	—	RP6R<4:0>					0000
RPOR4	06C8	—	—	—	RP9R<4:0>					—	—	—	RP8R<4:0>					0000
RPOR5	06CA	—	—	—	RP11R<4:0>					—	—	—	RP10R<4:0>					0000
RPOR6	06CC	—	—	—	RP13R<4:0>					—	—	—	RP12R<4:0>					0000
RPOR7	06CE	—	—	—	RP15R<4:0>					—	—	—	RP14R<4:0>					0000

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

TABLE 3-13: PERIPHERAL PIN SELECT OUTPUT REGISTER MAP FOR PIC24HJ32GP204 AND PIC24HJ16GP304

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
RPOR0	06C0	—	—	—	RP1R<4:0>					—	—	—	RP0R<4:0>					0000
RPOR1	06C2	—	—	—	RP3R<4:0>					—	—	—	RP2R<4:0>					0000
RPOR2	06C4	—	—	—	RP5R<4:0>					—	—	—	RP4R<4:0>					0000
RPOR3	06C6	—	—	—	RP7R<4:0>					—	—	—	RP6R<4:0>					0000
RPOR4	06C8	—	—	—	RP9R<4:0>					—	—	—	RP8R<4:0>					0000
RPOR5	06CA	—	—	—	RP11R<4:0>					—	—	—	RP10R<4:0>					0000
RPOR6	06CC	—	—	—	RP13R<4:0>					—	—	—	RP12R<4:0>					0000
RPOR7	06CE	—	—	—	RP15R<4:0>					—	—	—	RP14R<4:0>					0000
RPOR8	06D0	—	—	—	RP17R<4:0>					—	—	—	RP16R<4:0>					0000
RPOR9	06D2	—	—	—	RP19R<4:0>					—	—	—	RP18R<4:0>					0000
RPOR10	06D4	—	—	—	RP21R<4:0>					—	—	—	RP20R<4:0>					0000
RPOR11	06D6	—	—	—	RP23R<4:0>					—	—	—	RP22R<4:0>					0000
RPOR12	06D8	—	—	—	RP25R<4:0>					—	—	—	RP24R<4:0>					0000

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

TABLE 3-14: ADC1 REGISTER MAP FOR PIC24HJ32GP204 AND PIC24HJ16GP304

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
ADC1BUF0	0300	ADC Data Buffer 0																xxxx
ADC1BUF1	0302	ADC Data Buffer 1																xxxx
ADC1BUF2	0304	ADC Data Buffer 2																xxxx
ADC1BUF3	0306	ADC Data Buffer 3																xxxx
ADC1BUF4	0308	ADC Data Buffer 4																xxxx
ADC1BUF5	030A	ADC Data Buffer 5																xxxx
ADC1BUF6	030C	ADC Data Buffer 6																xxxx
ADC1BUF7	030E	ADC Data Buffer 7																xxxx
ADC1BUF8	0310	ADC Data Buffer 8																xxxx
ADC1BUF9	0312	ADC Data Buffer 9																xxxx
ADC1BUFA	0314	ADC Data Buffer 10																xxxx
ADC1BUFB	0316	ADC Data Buffer 11																xxxx
ADC1BUFC	0318	ADC Data Buffer 12																xxxx
ADC1BUFD	031A	ADC Data Buffer 13																xxxx
ADC1BUFE	031C	ADC Data Buffer 14																xxxx
ADC1BUFE	031E	ADC Data Buffer 15																xxxx
AD1CON1	0320	ADON	—	ADSIDL	—	—	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<7:0>								0000
AD1CHS123	0326	—	—	—	—	—	CH123NB<1:0>		CH123SB	—	—	—	—	—	CH123NA<1:0>		CH123SA	0000
AD1CHS0	0328	CH0NB	—	—	CH0SB<4:0>					CH0NA	—	—	CH0SA<4:0>					0000
AD1PCFGL	032C	—	—	—	PCFG12	PCFG11	PCFG10	PCFG9	PCFG8	PCFG7	PCFG6	PCFG5	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0	0000
AD1CSSL	0330	—	—	—	CSS12	CSS11	CSS10	CSS9	CSS8	CSS7	CSS6	CSS5	CSS4	CSS3	CSS2	CSS1	CSS0	0000

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

TABLE 3-15: ADC1 REGISTER MAP FOR PIC24HJ32GP202

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
ADC1BUF0	0300	ADC Data Buffer 0																xxxx
ADC1BUF1	0302	ADC Data Buffer 1																xxxx
ADC1BUF2	0304	ADC Data Buffer 2																xxxx
ADC1BUF3	0306	ADC Data Buffer 3																xxxx
ADC1BUF4	0308	ADC Data Buffer 4																xxxx
ADC1BUF5	030A	ADC Data Buffer 5																xxxx
ADC1BUF6	030C	ADC Data Buffer 6																xxxx
ADC1BUF7	030E	ADC Data Buffer 7																xxxx
ADC1BUF8	0310	ADC Data Buffer 8																xxxx
ADC1BUF9	0312	ADC Data Buffer 9																xxxx
ADC1BUFA	0314	ADC Data Buffer 10																xxxx
ADC1BUFB	0316	ADC Data Buffer 11																xxxx
ADC1BUFC	0318	ADC Data Buffer 12																xxxx
ADC1BUFD	031A	ADC Data Buffer 13																xxxx
ADC1BUFE	031C	ADC Data Buffer 14																xxxx
ADC1BUFF	031E	ADC Data Buffer 15																xxxx
AD1CON1	0320	ADON	—	ADSIDL	—	—	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<7:0>								0000
AD1CHS123	0326	—	—	—	—	—	CH123NB<1:0>		CH123SB	—	—	—	—	—	CH123NA<1:0>		CH123SA	0000
AD1CHS0	0328	CH0NB	—	—	CH0SB<4:0>					CH0NA	—	—	CH0SA<4:0>					0000
AD1PCFGL	032C	—	—	—	PCFG12	PCFG11	PCFG10	PCFG9	—	—	—	PCFG5	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0	0000
AD1CSSL	0330	—	—	—	CSS12	CSS11	CSS10	CSS9	—	—	—	CSS5	CSS4	CSS3	CSS2	CSS1	CSS0	0000

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

TABLE 3-16: PORTA REGISTER MAP FOR PIC24HJ32GP202

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	—	—	—	—	—	—	—	—	—	—	—	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	001F
PORTA	02C2	—	—	—	—	—	—	—	—	—	—	—	RA4	RA3	RA2	RA1	RA0	xxxx
LATA	02C4	—	—	—	—	—	—	—	—	—	—	—	LATA4	LATA3	LATA2	LATA1	LATA0	xxxx
ODCA	02C6	—	—	—	—	—	—	—	—	—	—	—	ODCA4	ODCA3	ODCA2	ODCA1	ODCA0	xxxx

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

TABLE 3-17: PORTA REGISTER MAP FOR PIC24HJ32GP204 AND PIC24HJ16GP304

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	—	—	—	—	—	TRISA10	TRISA9	TRISA8	TRISA7	—	—	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	079F
PORTA	02C2	—	—	—	—	—	RA10	RA9	RA8	RA7	—	—	RA4	RA3	RA2	RA1	RA0	xxxx
LATA	02C4	—	—	—	—	—	LATA10	LATA9	LATA8	LATA7	—	—	LATA4	LATA3	LATA2	LATA1	LATA0	xxxx
ODCA	02C6	—	—	—	—	—	ODCA10	ODCA9	ODCA8	ODCA7	—	—	ODCA4	ODCA3	ODCA2	ODCA1	ODCA0	xxxx

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

TABLE 3-18: PORTB REGISTER MAP

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
TRISB	02C8	TRISB15	TRISB14	TRISB13	TRISB12	TRISB11	TRISB10	TRISB9	TRISB8	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	FFFF
PORTB	02CA	RB15	RB14	RB13	RB12	RB11	RB10	RB9	RB8	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx
LATB	02CC	LATB15	LATB14	LATB13	LATB12	LATB11	LATB10	LATB9	LATB8	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	xxxx
ODCB	02CE	ODCB15	ODCB14	ODCB13	ODCB12	ODCB11	ODCB10	ODCB9	ODCB8	ODCB7	ODCB6	ODCB5	ODCB4	ODCB3	ODCB2	ODCB1	ODCB0	xxxx

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

TABLE 3-19: PORTC REGISTER MAP FOR PIC24HJ32GP204 AND PIC24HJ16GP304

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	02D0	—	—	—	—	—	—	TRISC9	TRISC8	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	03FF
PORTC	02D2	—	—	—	—	—	—	RC9	RC8	RC7	RC6	RC5	RC4	RC4	RC2	RC1	RC0	xxxx
LATC	02D4	—	—	—	—	—	—	LATC9	LATC8	LATC7	LATC6	LATC5	LATC4	LATC4	LATC2	LATC1	LATC0	xxxx
ODCC	02D6	—	—	—	—	—	—	ODCC9	ODCC8	ODCC7	ODCC6	ODCC5	ODCC4	ODCC4	ODCC2	ODCC1	ODCC0	xxxx

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

TABLE 3-20: 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	—	—	NVMOP<3:0>				0000[1]
NVMKEY	0766	—	—	—	—	—	—	—	—	NVMKEY<7:0>								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 3-21: 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	—	—	T3MD	T2MD	T1MD	—	—	—	I2C1MD	—	U1MD	—	SPI1MD	—	—	AD1MD	0000
PMD2	0772	IC8MD	IC7MD	—	—	—	—	IC2MD	IC1MD	—	—	—	—	—	—	OC2MD	OC1MD	0000

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

3.2.5 SOFTWARE STACK

In addition to its use as a working register, the W15 register in the PIC24HJ32GP202/204 and PIC24HJ16GP304 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 3-4. 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.

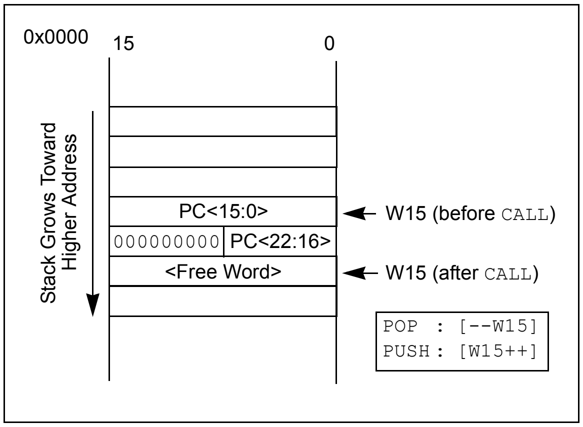
The Stack Pointer Limit register (SPLIM) associated with the Stack Pointer sets an upper address boundary for the stack. SPLIM is uninitialized at Reset. Similarly, the Stack Pointer, SPLIM<0> is forced to ‘`0`’ because all stack operations must be word aligned.

When 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. For example, 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 lesser 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 3-4: `CALL` STACK FRAME



3.2.6 DATA RAM PROTECTION FEATURE

The PIC24H product family supports Data RAM protection features that enable segments of RAM to be protected when used in conjunction with Boot and Secure Code Segment Security. BSRAM (Secure RAM segment for BS) is accessible only from the Boot Segment Flash code when enabled. SSRAM (Secure RAM segment for RAM) is accessible only from the Secure Segment Flash code when enabled. See Table 3-1 for an overview of the BSRAM and SSRAM SFRs.

3.3 Instruction Addressing Modes

The addressing modes shown in Table 3-22 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 differ 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 three-operand MCU instructions are of the form:

Operand 3 = Operand 1 <function> Operand 2

where, Operand 1 is always a working register (that is, 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 can support different subsets of these addressing modes.

TABLE 3-22: 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 Effective Address (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 (Register Indexed)	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 (`MOV`) INSTRUCTION

Move instructions provide a greater degree of addressing flexibility than the other instructions. In addition to the Addressing modes supported by most MCU instructions, `MOV` instructions also support Register Indirect with Register Offset Addressing mode. This is 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 the destination EA. However, the 4-bit Wb (Register Offset) field is shared by both source and destination (but typically only used by one).

In summary, move instructions support the following addressing modes:

- 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 addressing modes outlined previously, 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, such as `ADD Acc`, 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 PIC24HJ32GP202/204 and PIC24HJ16GP304 architecture uses a 24-bit-wide program space and a 16-bit wide data space. The architecture is also a modified Harvard scheme, which means that the 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 PIC24HJ32GP202/204 and PIC24HJ16GP304 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 periodically. 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. The application 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`).

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.

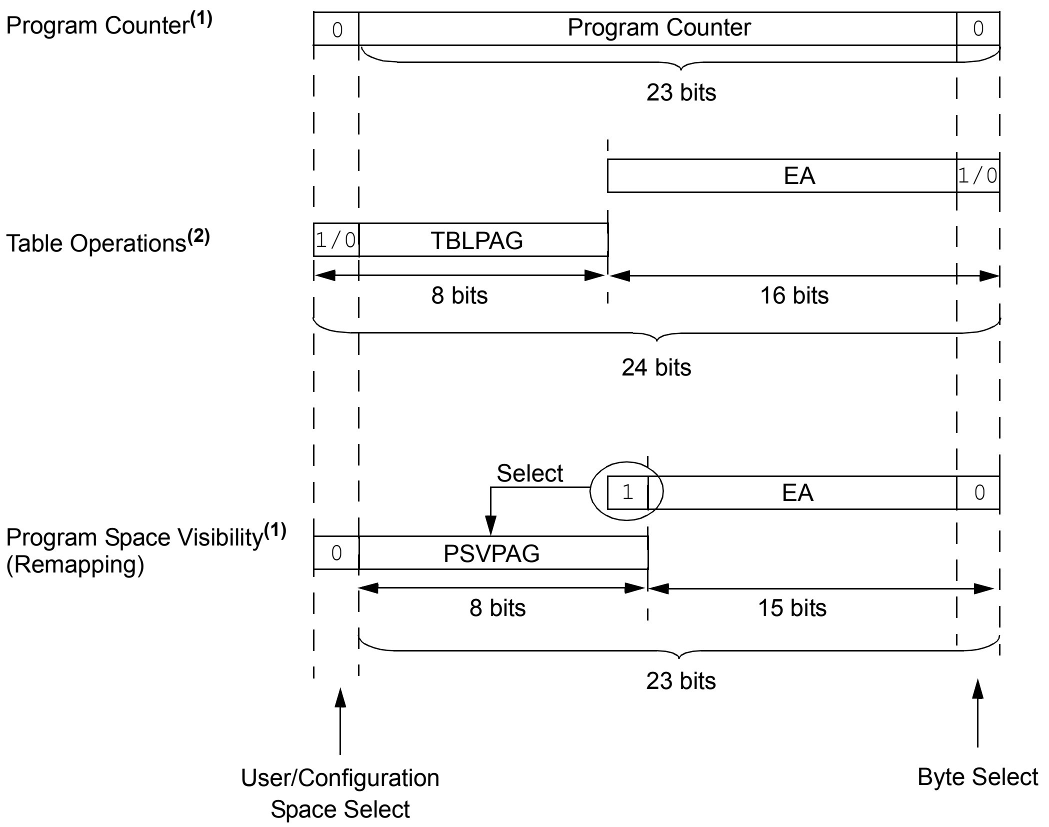
Table 3-23 and Figure 3-5 show how the program EA is created for table operations and remapping accesses from the data EA. Here, P<23:0> refers to a program space word, and D<15:0> refers to a data space word.

TABLE 3-23: PROGRAM SPACE ADDRESS CONSTRUCTION

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

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

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



Note 1: The Least Significant bit (LSb) of program space addresses is always fixed as '0' to maintain word alignment of data in the program and data spaces.

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

3.4.2 DATA ACCESS FROM PROGRAM MEMORY USING TABLE INSTRUCTIONS

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

The PC is incremented by 2 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 wide word address spaces, residing side by side, each with the same address range. `TBLRDL` and `TBLWTL` access the space that contains the least significant data word. `TBLRDH` and `TBLWTH` access the space that 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.

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

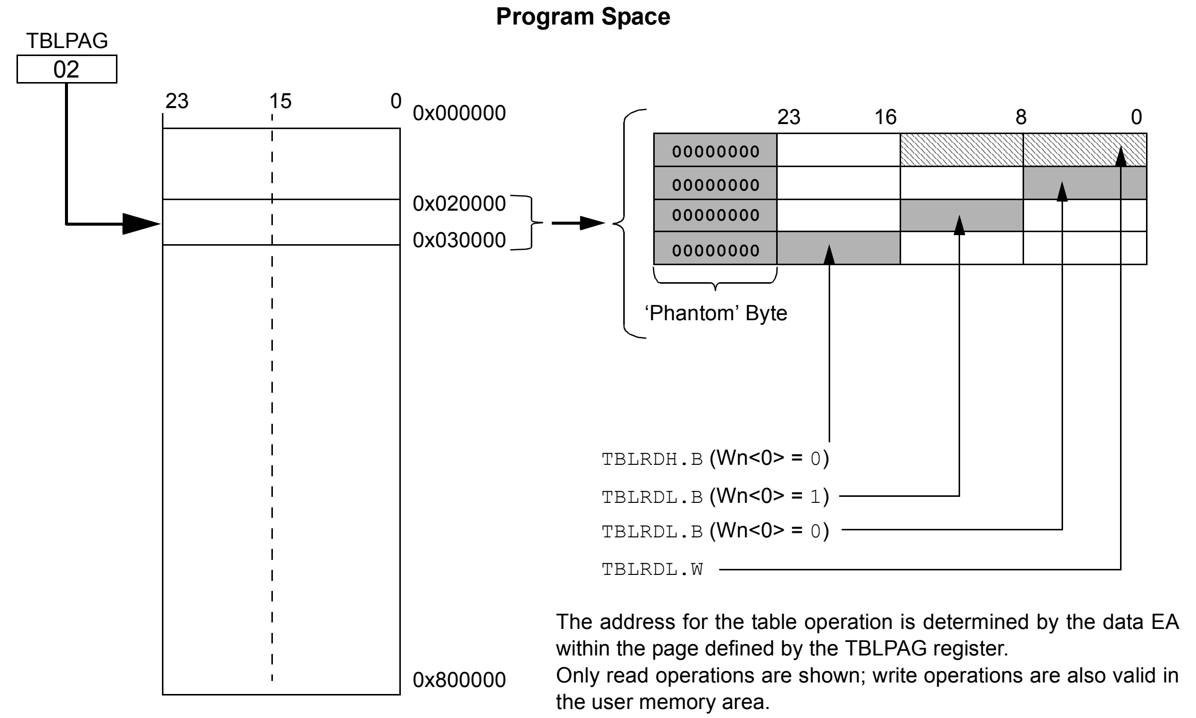
  In Byte mode, either the upper or lower byte of the lower program word is mapped to the lower byte of a data address. The upper byte is selected when Byte Select is ‘`1`’; the lower byte is selected when it is ‘`0`’.
- `TBLRDH` (Table Read High): In Word mode, this instruction maps the entire upper word 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, this instruction maps the upper or lower byte of the program word to D<7:0> of the data address, as in the `TBLRDL` instruction. 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 3-6: ACCESSING PROGRAM MEMORY WITH TABLE INSTRUCTIONS



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

3.4.3 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 to the stored constant data from the data space without the need to use special instructions (such as `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. 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 a cycle to the instruction being executed, since two program memory fetches are required.

Although each data space address 8000h and higher maps directly into a corresponding program memory address (see Figure 3-7), only the lower 16 bits of the 24-bit program word are used to contain the data. The upper 8 bits of any program space 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.

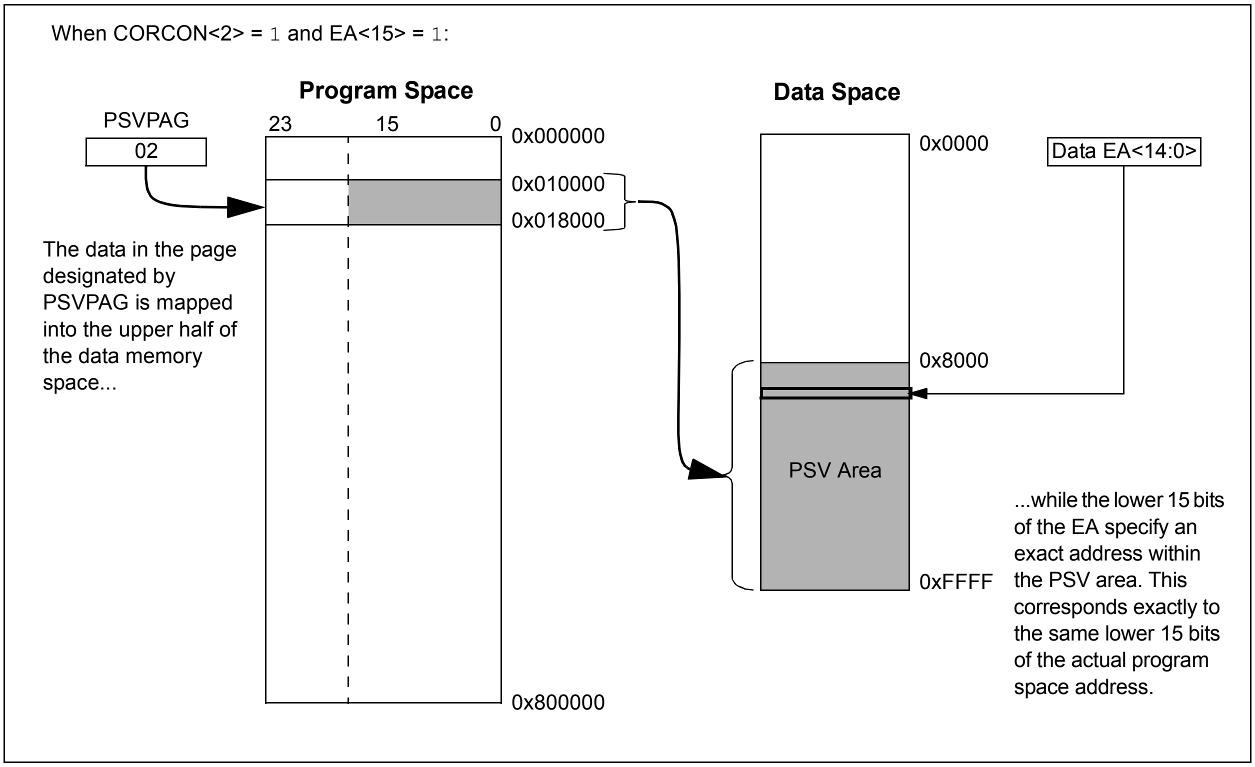
For operations that use PSV and are executed outside a REPEAT loop, the `MOV` and `MOV.D` instructions require one instruction cycle in addition to the specified execution time. All other instructions require two instruction cycles in addition to the specified execution time.

For operations that use PSV, and are executed inside a REPEAT loop, these instances 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 using PSV to access data to execute in a single cycle.

FIGURE 3-7: PROGRAM SPACE VISIBILITY OPERATION



NOTES:

4.0 FLASH PROGRAM MEMORY

Note: This data sheet summarizes the features of the PIC24HJ32GP202/204 and PIC24HJ16GP304 devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the *"PIC24H Family Reference Manual"*.

The PIC24HJ32GP202/204 and PIC24HJ16GP304 devices contain internal Flash program memory to store and execute 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:

- In-Circuit Serial Programming™ (ICSP™) programming capability
- Run-Time Self-Programming (RTSP)

ICSP allows a PIC24HJ32GP202/204 and PIC24HJ16GP304 device to be serially programmed while in the end application circuit. This is 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 ($\overline{\text{MCLR}}$). This allows customers to manufacture boards with unprogrammed devices and then program the microcontroller just before shipping the product. This also allows the most recent firmware or a custom firmware to be programmed.

RTSP is accomplished using `TBLRD` (table read) and `TBLWT` (table write) instructions. With RTSP, the user application can write program memory data either in 'blocks' or 'rows' of 64 instructions (192 bytes) at a time or a single program memory word, and erase program memory in blocks or 'pages' of 512 instructions (1536 bytes) at a time.

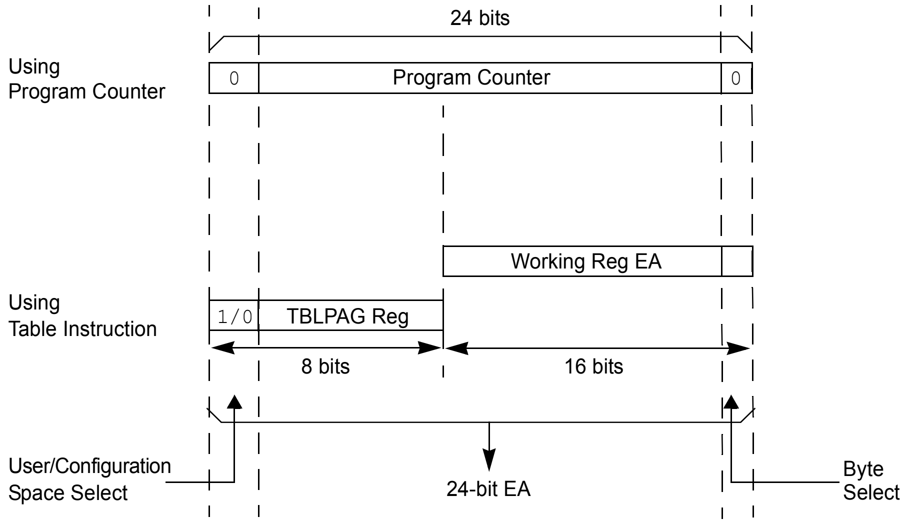
4.1 Table Instructions and Flash Programming

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 the 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 PIC24HJ32GP202/204 and PIC24HJ16GP304 Flash program memory array is organized into rows of 64 instructions or 192 bytes. RTSP allows the user application to erase a page of memory, which consists of eight rows (512 instructions) at a time, and to program one row or one word at a time. 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 sequentially. 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 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

Two SFRs are used to read and write the program Flash memory:

- **NVMCON: Flash Memory Control Register**
- **NVMKEY: Non-Volatile Memory Key Register**

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

NVMKEY (Register 4-2) is a write-only register that is used for write protection. To start a programming or erase sequence, the user application 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(1)	R/W-0(1)	R/W-0(1)	U-0	U-0	U-0	U-0	U-0
WR	WREN	WRERR	—	—	—	—	—
bit 15							bit 8

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

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

bit 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(2)

<u>If ERASE = `1`:</u>
`1111` = Memory bulk erase operation
`1101` = Erase General Segment
`1100` = Erase Secure Segment
`0011` = No operation
`0010` = Memory page erase operation
`0001` = No operation
`0000` = Erase a single Configuration register byte

<u>If ERASE = `0`:</u>
`1111` = No operation
`1101` = No operation
`1100` = No operation
`0011` = Memory word program operation
`0010` = No operation
`0001` = Memory row program operation
`0000` = Program 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.

REGISTER 4-2: NVMKEY: NON-VOLATILE MEMORY KEY REGISTER

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

W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
NVMKEY<7:0>							
bit 7							bit 0

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

bit 15-8	**Unimplemented:** Read as ‘0’
bit 7-0	**NVMKEY<7:0>:** Key Register (Write Only) bits

4.4.1 PROGRAMMING ALGORITHM FOR FLASH PROGRAM MEMORY

Programmers 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 block (see Example 4-1):
    a) Set the NVMOP bits (NVMCON<3:0>) to ‘`0010`’ to configure for block erase. Set 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) Write 55h to NVMKEY.
    d) Write AAh to NVMKEY.
    e) 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 55h to NVMKEY.
    c) Write AAh to NVMKEY.
    d) Set the WR bit. The programming cycle begins and the CPU stalls for the duration of the write cycle. When the write to Flash memory is done, the WR bit is cleared automatically.
6. Repeat steps 4 and 5, using the next available 64 instructions from the block in data RAM by incrementing the value in TBLPAG, until all 512 instructions are written back to Flash memory.

To protect 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 application must wait for the programming time until programming is complete. The two instructions following the start of the programming sequence should be `NOP`s, 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                                   ; Insert two NOPs after the erase
       NOP                                   ; command is asserted
```

EXAMPLE 4-2: LOADING THE WRITE BUFFERS

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

EXAMPLE 4-3: INITIATING A PROGRAMMING SEQUENCE

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

5.0 RESETS

Note: This data sheet summarizes the features of the PIC24HJ32GP202/204 and PIC24HJ16GP304 devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the *"PIC24H Family Reference Manual"*.

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
- WDTO: Watchdog Timer Reset
- TRAPR: Trap Conflict Reset
- IOPUWR: Illegal Opcode, Uninitialized W Register Reset, and Security Reset
- CM: Configuration Mismatch Reset

Figure 5-1 shows the simplified block diagram of the Reset module.

Any active source of Reset makes 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.

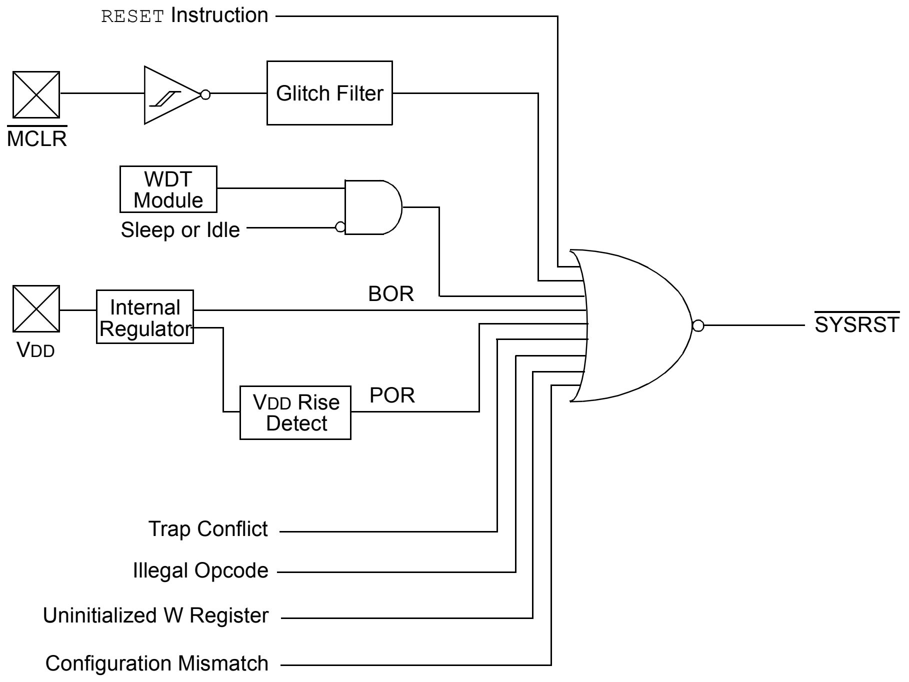
Note: Refer to the specific peripheral or CPU section of this manual for register Reset states.

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 application 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



REGISTER 5-1: RCON: RESET CONTROL REGISTER[1]

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

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[2]	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 13-10 **Unimplemented:** Read as ‘0’

bit 9 **CM:** Configuration Mismatch Flag bit

1 = A configuration mismatch Reset has occurred.
0 = A configuration mismatch Reset has NOT occurred.

bit 8 **VREGS:** Voltage Regulator Standby During Sleep bit

1 = Voltage regulator is active during Sleep
0 = Voltage regulator goes into Standby mode 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[2]

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

Note 1: All of the Reset status bits can 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[1]

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

Note 1: All of the Reset status bits can 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.

TABLE 5-1: RESET FLAG BIT OPERATION

Flag Bit	Setting Event	Clearing Event
TRAPR (RCON<15>)	Trap conflict event	POR, BOR
IOPUWR (RCON<14>)	Illegal opcode or uninitialized W register access	POR, BOR
CM (RCON<9>)	Configuration mismatch	POR, BOR
EXTR (RCON<7>)	$\overline{\text{MCLR}}$ Reset	POR
SWR (RCON<6>)	`RESET` instruction	POR, BOR
WDTO (RCON<4>)	WDT time-out	`PWRSAV` instruction, POR, BOR, `CLRWDT` instruction
SLEEP (RCON<3>)	`PWRSAV #SLEEP` instruction	POR, BOR
IDLE (RCON<2>)	`PWRSAV #IDLE` instruction	POR, BOR
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 7.0 "Oscillator Configuration"** for further details.

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

Reset Type	Clock Source Determinant
POR	Oscillator Configuration bits (FNOSC<2:0>)
BOR	
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, SYSRST, 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 SYSRST delay times.

The FSCM delay determines the time at which the FSCM begins to monitor the system clock source after the SYSRST signal is released.

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

Reset Type	Clock Source	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**
BOR	EC, FRC, LPRC	TSTARTUP + TRST	—	—	**3**
	ECPLL, FRCPLL	TSTARTUP + TRST	TLOCK	TFSCM	**3, 5, 6**
	XT, HS, SOSC	TSTARTUP + TRST	TOST	TFSCM	**3, 4, 6**
	XTPLL, HSPLL	TSTARTUP + TRST	TOST + TLOCK	TFSCM	**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 µs nominal).

2: TSTARTUP = Conditional POR delay of 20 µs 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 µs nominal).

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

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

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

5.2.1 POR AND LONG OSCILLATOR START-UP TIMES

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

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

The device will not begin to execute code until a valid clock source has been released to the system. Therefore, the oscillator and the 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 $\overline{\text{SYSRST}}$ is released. If a valid clock source is not available, the device automatically switches to the FRC oscillator and the user application 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 short delay, T_{FSCM}, is automatically inserted after the POR and PWRT delay times. The FSCM does not start to monitor the system clock source until this delay expires. The FSCM delay time is nominally 500 μs 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.

NOTES:

6.0 INTERRUPT CONTROLLER

> **Note:** This data sheet summarizes the features of the PIC24HJ32GP202/204 and PIC24HJ16GP304 devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the *"PIC24H Family Reference Manual"*.

The PIC24HJ32GP202/204 and PIC24HJ16GP304 interrupt controllers reduce the numerous peripheral interrupt request signals to a single interrupt request signal to the PIC24HJ32GP202/204 and PIC24HJ16GP304 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

Figure 6-1 shows the Interrrupt Vector Table. The IVT resides in program memory, starting at location 000004h. The IVT contains 126 vectors consisting of 8 nonmaskable trap vectors and 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. Lower addresses generally have a higher natural priority. For example, the interrupt associated with vector 0 will take priority over interrupts at any other vector address.

PIC24HJ32GP202/204 and PIC24HJ16GP304 devices implement up to 21 unique interrupts and 4 non-maskable traps. These are summarized in Table 6-1 and Table 6-2.

6.1.1 ALTERNATE INTERRUPT VECTOR TABLE

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

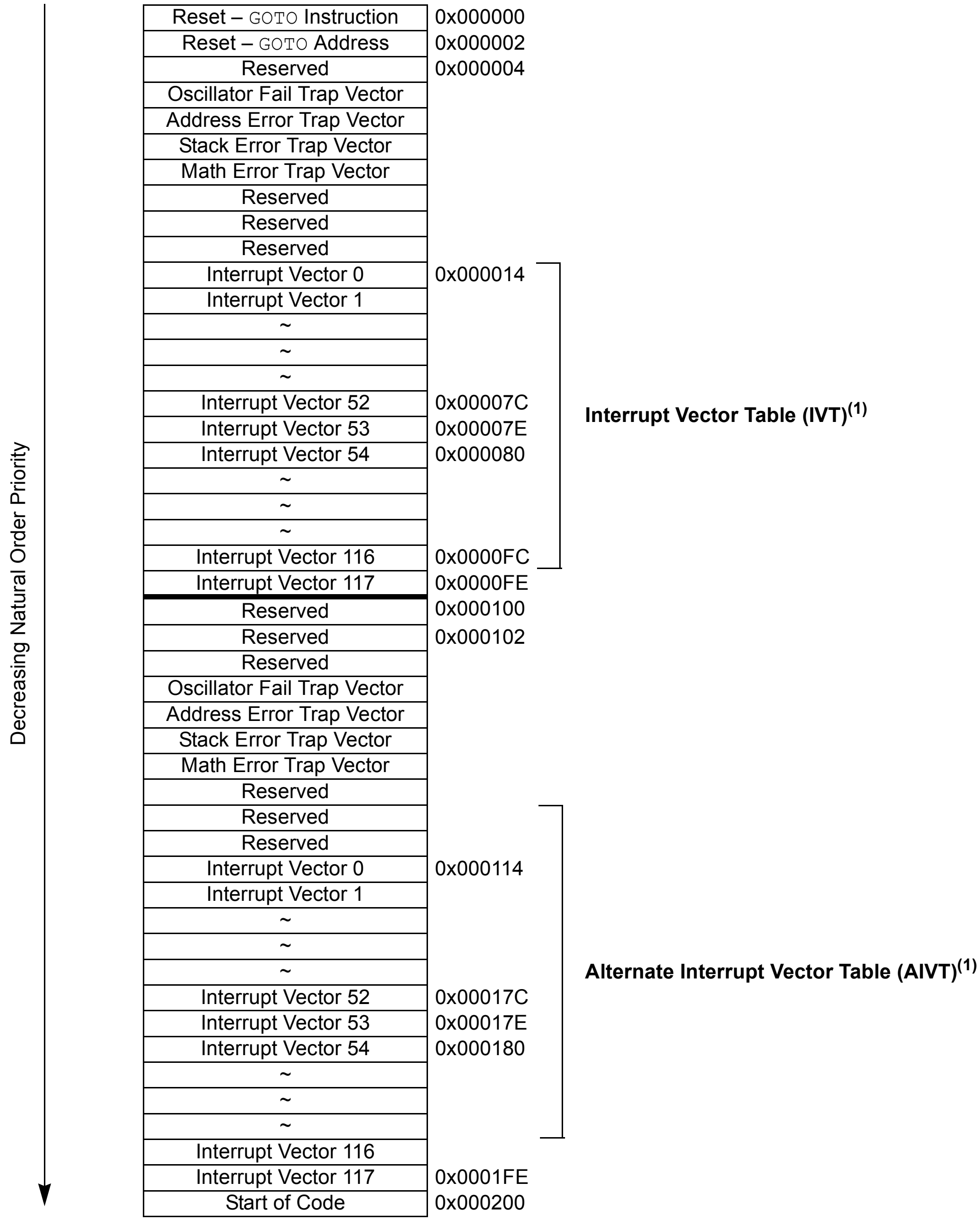
The AIVT supports debugging by providing a means to switch between an application and a support environment without requiring the interrupt vectors to be reprogrammed. This feature also enables switching between applications for evaluation of different software algorithms at run time. If the AIVT is 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 PIC24HJ32GP202/204 and PIC24HJ16GP304 device clear its registers in response to a Reset, which forces the PC to zero. The microcontroller then begins the program execution at location 0x000000. The user application can use 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.

FIGURE 6-1: PIC24HJ32GP202/204 AND PIC24HJ16GP304 INTERRUPT VECTOR TABLE



Reset – `GOTO` Instruction	0x000000
Reset – `GOTO` Address	0x000002
Reserved	0x000004
Oscillator Fail Trap Vector	
Address Error Trap Vector	
Stack Error Trap Vector	
Math Error Trap Vector	
Reserved	
Reserved	
Reserved	
Interrupt Vector 0	0x000014
Interrupt Vector 1	
~	
~	
~	
Interrupt Vector 52	0x00007C
Interrupt Vector 53	0x00007E
Interrupt Vector 54	0x000080
~	
~	
~	
Interrupt Vector 116	0x0000FC
Interrupt Vector 117	0x0000FE
Reserved	0x000100
Reserved	0x000102
Reserved	
Oscillator Fail Trap Vector	
Address Error Trap Vector	
Stack Error Trap Vector	
Math Error Trap Vector	
Reserved	
Reserved	
Reserved	
Interrupt Vector 0	0x000114
Interrupt Vector 1	
~	
~	
~	
Interrupt Vector 52	0x00017C
Interrupt Vector 53	0x00017E
Interrupt Vector 54	0x000180
~	
~	
~	
Interrupt Vector 116	
Interrupt Vector 117	0x0001FE
Start of Code	0x000200

Note 1: See Table 6-1 for the list of implemented interrupt vectors.

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	Reserved
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 – ADC 1
22	14	0x000030	0x000130	Reserved
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	Change Notification Interrupt
28	20	0x00003C	0x00013C	INT1 – External Interrupt 1
29	21	0x00003E	0x00013E	Reserved
30	22	0x000040	0x000140	IC7 – Input Capture 7
31	23	0x000042	0x000142	IC8 – Input Capture 8
32	24	0x000044	0x000144	Reserved
33	25	0x000046	0x000146	Reserved
34	26	0x000048	0x000148	Reserved
35	27	0x00004A	0x00014A	Reserved
36	28	0x00004C	0x00014C	Reserved
37	29	0x00004E	0x00014E	INT2 – External Interrupt 2
38	30	0x000050	0x000150	Reserved
39	31	0x000052	0x000152	Reserved
40	32	0x000054	0x000154	Reserved
41	33	0x000056	0x000156	Reserved
42	34	0x000058	0x000158	Reserved
43	35	0x00005A	0x00015A	Reserved
44	36	0x00005C	0x00015C	Reserved
45	37	0x00005E	0x00015E	Reserved
46	38	0x000060	0x000160	Reserved
47	39	0x000062	0x000162	Reserved
48	40	0x000064	0x000164	Reserved
49	41	0x000066	0x000166	Reserved
50	42	0x000068	0x000168	Reserved
51	43	0x00006A	0x00016A	Reserved
52	44	0x00006C	0x00016C	Reserved
53	45	0x00006E	0x00016E	Reserved

TABLE 6-1: INTERRUPT VECTORS (CONTINUED)

Vector Number	Interrupt Request (IRQ) Number	IVT Address	AIVT Address	Interrupt Source
54	46	0x000070	0x000170	Reserved
55	47	0x000072	0x000172	Reserved
56	48	0x000074	0x000174	Reserved
57	49	0x000076	0x000176	Reserved
58	50	0x000078	0x000178	Reserved
59	51	0x00007A	0x00017A	Reserved
60	52	0x00007C	0x00017C	Reserved
61	53	0x00007E	0x00017E	Reserved
62	54	0x000080	0x000180	Reserved
63	55	0x000082	0x000182	Reserved
64	56	0x000084	0x000184	Reserved
65	57	0x000086	0x000186	Reserved
66	58	0x000088	0x000188	Reserved
67	59	0x00008A	0x00018A	Reserved
68	60	0x00008C	0x00018C	Reserved
69	61	0x00008E	0x00018E	Reserved
70	62	0x000090	0x000190	Reserved
71	63	0x000092	0x000192	Reserved
72	64	0x000094	0x000194	Reserved
73	65	0x000096	0x000196	U1E – UART1 Error
74	66	0x000098	0x000198	Reserved
75	67	0x00009A	0x00019A	Reserved
76	68	0x00009C	0x00019C	Reserved
77	69	0x00009E	0x00019E	Reserved
78	70	0x0000A0	0x0001A0	Reserved
79	71	0x0000A2	0x0001A2	Reserved
80-125	72-117	0x0000A4- 0x0000FE	0x0001A4- 0x0001FE	Reserved

TABLE 6-2: TRAP VECTORS

Vector Number	IVT Address	AIVT Address	Trap Source
0	0x000004	0x000104	Reserved
1	0x000006	0x000106	Oscillator Failure
2	0x000008	0x000108	Address Error
3	0x00000A	0x00010A	Stack Error
4	0x00000C	0x00010C	Math Error
5	0x00000E	0x00010E	Reserved
6	0x000010	0x000110	Reserved
7	0x000012	0x000112	Reserved

6.3 Interrupt Control and Status Registers

PIC24HJ32GP202/204 and PIC24HJ16GP304 devices implement a total of 17 registers for the interrupt controller:

- Interrupt Control Register 1 (INTCON1)
- Interrupt Control Register 2 (INTCON2)
- Interrupt Flag Status Registers (IFSx)
- Interrupt Enable Control Registers (IECx)
- Interrupt Priority Control Registers (IPCx)
- Interrupt Control and Status Register (INTTREG)

6.3.1 INTCON1 AND INTCON2

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.

6.3.2 IFSx

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

6.3.3 IECx

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

6.3.4 IPCx

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 the eight priority levels.

6.3.5 INTTREG

The INTTREG register contains the associated interrupt vector number and the new CPU interrupt priority level, which are latched into vector number (VECNUM<6:0>) and Interrupt level (ILR<3:0>) bit fields in the INTTREG register. The new interrupt priority level is the priority of the pending interrupt.

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 INT0 (External Interrupt 0) is shown as having vector number 8 and a natural order priority of 0. Thus, the INT0IF bit is found in IFS0<0>, the INT0IE bit in IEC0<0>, and the INT0IP bits in the first position of IPC0 (IPC0<2:0>).

6.3.6 STATUS REGISTERS

Although these 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 through Register 6-19 in the following pages.

REGISTER 6-1: SR: CPU STATUS REGISTER[1]

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[3]	R/W-0[3]	R/W-0[3]	R-0	R/W-0	R/W-0	R/W-0	R/W-0
IPL2[2]	**IPL1**[2]	**IPL0**[2]	RA	N	OV	Z	C
bit 7							bit 0

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[1]

`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(1)

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(2)**	PSV	—	—
bit 7							bit 0

Legend:	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 3 **IPL3:** CPU Interrupt Priority Level Status bit 3(2)
`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	U-0	U-0	U-0	U-0	U-0	U-0	U-0
NSTDIS	—	—	—	—	—	—	—
bit 15							bit 8

U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0
—	DIV0ERR	—	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 14-7 **Unimplemented:** Read as ‘0’.

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 **Unimplemented:** Read as ‘0’

bit 4 **MATHERR:** Arithmetic Error Status bit

1 = Math error trap has occurred
0 = Math error trap has not occurred

bit 3 **ADDRERR:** Address Error Trap Status bit

1 = Address error trap has occurred
0 = Address error trap has not occurred

bit 2 **STKERR:** Stack Error Trap Status bit

1 = Stack error trap has occurred
0 = Stack error trap has not occurred

bit 1 **OSCFAIL:** Oscillator Failure Trap Status bit

1 = Oscillator failure trap has occurred
0 = Oscillator failure trap has not occurred

bit 0 **Unimplemented:** Read as ‘0’

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	U-0	U-0	R/W-0	R/W-0	R/W-0
—	—	—	—	—	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 13-3 **Unimplemented:** Read as ‘0’

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

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

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	AD1IF	U1TXIF	U1RXIF	SPI1IF	SPI1EIF	T3IF
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/W-0
T2IF	OC2IF	IC2IF	—	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 15-4 **Unimplemented:** Read as ‘0’

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 **Unimplemented:** Read as ‘0’

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

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

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

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

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

U-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
—	—	INT2IF	—	—	—	—	—
bit 15							bit 8

R/W-0	R/W-0	U-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
IC8IF	IC7IF	—	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 15-4	**Unimplemented:** Read as ‘0’
bit 13	**INT2IF:** External Interrupt 2 Flag Status bit 1 = Interrupt request has occurred 0 = Interrupt request has not occurred
bit 12-8	**Unimplemented:** Read as ‘0’
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	**Unimplemented:** Read as ‘0’
bit 4	**INT1IF:** External Interrupt 1 Flag Status bit 1 = Interrupt request has occurred 0 = Interrupt request has not occurred
bit 3	**CNIF:** Input Change Notification Interrupt Flag Status bit 1 = Interrupt request has occurred 0 = Interrupt request has not occurred
bit 2	**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

REGISTER 6-7: 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

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	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 15-2	**Unimplemented:** Read as ‘0’
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’

REGISTER 6-8: IEC0: INTERRUPT ENABLE CONTROL REGISTER 0

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	AD1IE	U1TXIE	U1RXIE	SPI1IE	SPI1EIE	T3IE
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/W-0
T2IE	OC2IE	IC2IE	—	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 15-4 **Unimplemented:** Read as ‘0’

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 **Unimplemented:** Read as ‘0’

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

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

REGISTER 6-8: IEC0: INTERRUPT ENABLE CONTROL REGISTER 0 (CONTINUED)

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

REGISTER 6-9: IEC1: INTERRUPT ENABLE CONTROL REGISTER 0

U-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
—	—	INT2IE	—	—	—	—	—
bit 15							bit 8

R/W-0	R/W-0	U-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
IC8IE	IC7IE	—	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 15-14	**Unimplemented:** Read as ‘0’
bit 13	**INT2IE:** External Interrupt 2 Enable bit 1 = Interrupt request enabled 0 = Interrupt request not enabled
bit 12-8	**Unimplemented:** Read as ‘0’
bit 7	**IC8IE:** Input Capture Channel 8 Interrupt Enable bit 1 = Interrupt request enabled 0 = Interrupt request not enabled
bit 6	**IC7IE:** Input Capture Channel 7 Interrupt Enable bit 1 = Interrupt request enabled 0 = Interrupt request not enabled
bit 5	**Unimplemented:** Read as ‘0’
bit 4	**INT1IE:** External Interrupt 1 Enable bit 1 = Interrupt request enabled 0 = Interrupt request not enabled
bit 3	**CNIE:** Input Change Notification Interrupt Enable bit 1 = Interrupt request enabled 0 = Interrupt request not enabled
bit 2	**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

REGISTER 6-10: IEC4: INTERRUPT ENABLE CONTROL REGISTER 0

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	U-0	R/W-0	U-0
—	—	—	—	—	—	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 15-2	**Unimplemented:** Read as ‘`0`’
bit 1	**U1EIE:** UART1 Error Interrupt Enable bit `1` = Interrupt request enabled `0` = Interrupt request not enabled
bit 0	**Unimplemented:** Read as ‘`0`’

REGISTER 6-11: 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 6-12: 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	U-0	U-0	U-0
—	IC2IP<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 3-0 **Unimplemented:** Read as ‘`0`’

REGISTER 6-13: 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 6-14: IPC3: INTERRUPT PRIORITY CONTROL REGISTER 3

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	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 15-7 **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

REGISTER 6-15: 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 6-16: 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	U-0	U-0	U-0	U-0	R/W-1	R/W-0	R/W-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 7-3 **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

REGISTER 6-17: IPC7: INTERRUPT PRIORITY CONTROL REGISTER 7

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
—	INT2IP<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 **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 3-0 **Unimplemented:** Read as ‘0’

REGISTER 6-18: IPC16: INTERRUPT PRIORITY CONTROL REGISTER 16

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
—	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 15-7	**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 6-19: INTTREG: INTERRUPT CONTROL AND STATUS REGISTER

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

U-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
—	VECNUM<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 15-12 **Unimplemented:** Read as ‘`0`’

bit 11-8 **ILR:** New CPU Interrupt Priority Level bits

`1111` = CPU Interrupt Priority Level is 15
•
•
•
`0001` = CPU Interrupt Priority Level is 1
`0000` = CPU Interrupt Priority Level is 0

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

bit 6-0 **VECNUM:** Vector Number of Pending Interrupt bits

`0111111` = Interrupt Vector pending is number 135
•
•
•
`0000001` = Interrupt Vector pending is number 9
`0000000` = Interrupt Vector pending is number 8

6.4 Interrupt Setup Procedures

6.4.1 INITIALIZATION

To configure an interrupt source at initialization:

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 can 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. Set the interrupt enable control bit associated with the source in the appropriate IECx register to enable the interrupt source.

6.4.2 INTERRUPT SERVICE ROUTINE

The method used to declare an ISR and initialize the IVT with the correct vector address depends on the programming language (C or Assembler) and the language development toolsuite used to develop the application.

In general, the user application must clear the interrupt flag in the appropriate IFSx register for the source of interrupt that the ISR handles. Otherwise, the program will re-enter the ISR 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 this procedure:

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

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

Note:	Only user interrupts with a priority level of 7 or lower 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.

NOTES:

7.0 OSCILLATOR CONFIGURATION

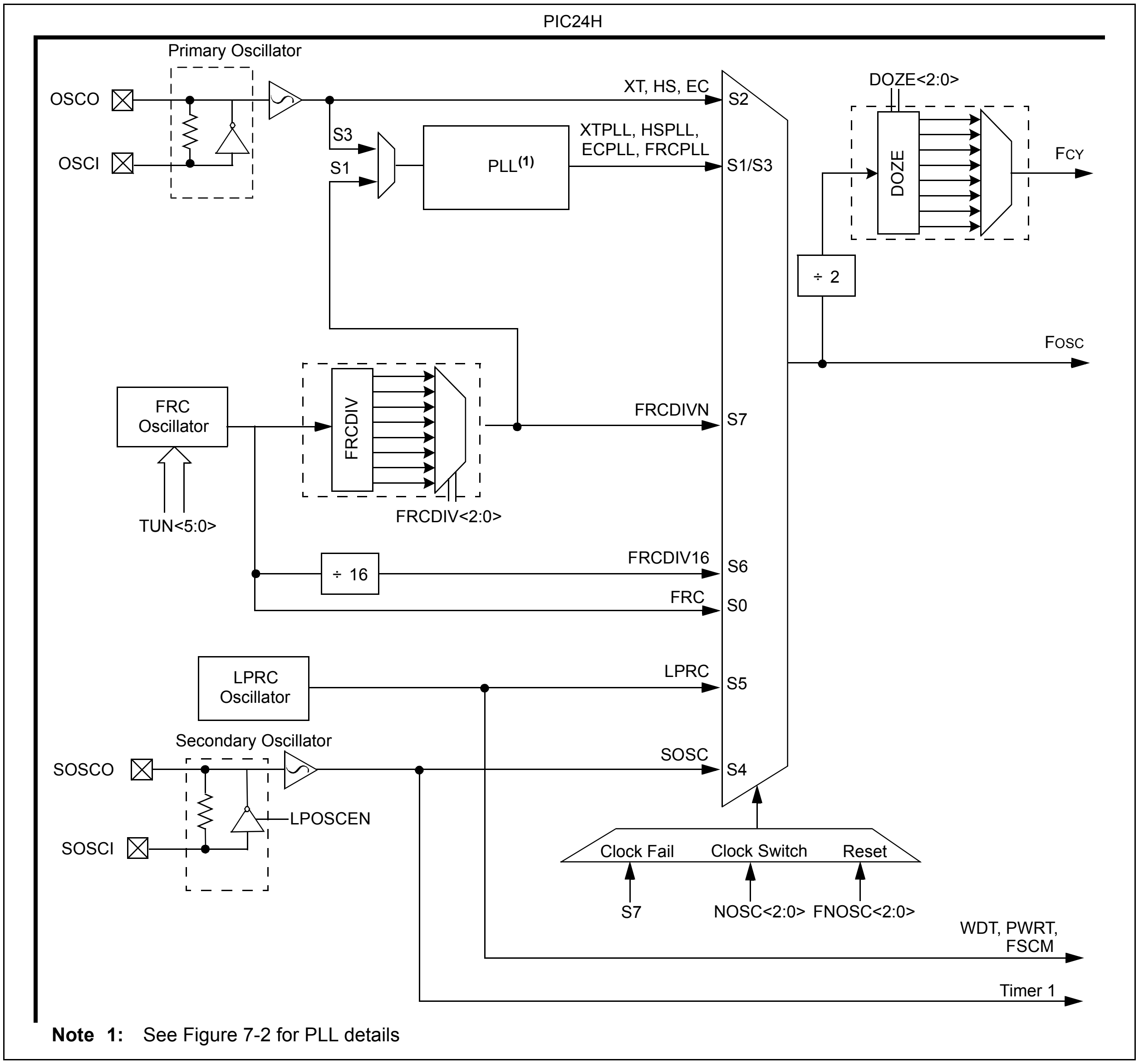
Note: This data sheet summarizes the features of the PIC24HJ32GP202/204 and PIC24HJ16GP304 devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the *"PIC24H Family Reference Manual"*.

The PIC24HJ32GP202/204 and PIC24HJ16GP304 oscillator system provides:

- External and internal oscillator options as clock sources
- An on-chip PLL to scale the internal operating frequency to the required system clock frequency
- An internal FRC oscillator that 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 7-1.

FIGURE 7-1: PIC24HJ32GP202/204 AND PIC24HJ16GP304 OSCILLATOR SYSTEM DIAGRAM



Note 1: See Figure 7-2 for PLL details

7.1 CPU Clocking System

The PIC24HJ32GP202/204 and PIC24HJ16GP304 devices provide the following seven system clock options.

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

7.1.1 SYSTEM CLOCK SOURCES

7.1.1.1 Fast RC

The Fast RC (FRC) internal oscillator runs at a nominal frequency of 7.37 MHz. 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 FRCDIV<2:0> (CLKDIV<10:8>) bits.

7.1.1.2 Primary

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

- Crystal (XT): Crystals and ceramic resonators in the range of 3 MHz to 10 MHz. The crystal is connected to the OSC1 and OSC2 pins.
- HS (High-Speed Crystal): Crystals in the range of 10 MHz to 40 MHz. The crystal is connected to the OSC1 and OSC2 pins.
- 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.

7.1.1.3 Secondary

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

7.1.1.4 Low-Power RC

The Low-Power RC (LPRC) 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).

7.1.1.5 FRC

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 7.1.3 “PLL Configuration”**.

7.1.2 SYSTEM CLOCK SELECTION

The oscillator source 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 18.1 “Configuration Bits”** for further details.) The Initial Oscillator Selection Configuration bits, FNOSC<2:0> (FOSCSEL<2:0>), and the Primary Oscillator Mode Select Configuration bits, POSCMD<1:0> (FOSC<1:0>), 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 among 12 different clock modes, shown in Table 7-1.

The output of the oscillator (or the output of the PLL if a PLL mode has been selected) FOSC is divided by 2 to generate the device instruction clock (FCY). FCY defines the operating speed of the device, and speeds up to 40 MHz are supported by the PIC24HJ32GP202/204 and PIC24HJ16GP304 architecture.

Instruction execution speed or device operating frequency, FCY, is given by:

EQUATION 7-1: DEVICE OPERATING FREQUENCY

$$F_{CY} = F_{OSC}/2$$

7.1.3 PLL CONFIGURATION

The primary oscillator and internal FRC oscillator can optionally use on-chip PLL to obtain higher speeds of operation. The PLL provides significant flexibility in selecting the device operating speed. A block diagram of the PLL is shown in Figure 7-2.

The output of the primary oscillator or FRC, denoted as ‘FIN’ is divided down by a prescale factor (N1) of 2, 3, ... or 33 before it is being provided to the PLL’s Voltage Controlled Oscillator (VCO). The input to the VCO must be selected in the range of 0.8 MHz to 8 MHz. The prescale factor ‘N1’ is selected using the PLLPRE<4:0> bits (CLKDIV<4:0>).

The PLL Feedback Divisor, selected using the PLLDIV<8:0> bits (PLLFBD<8:0>), 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 PLLPOST<1:0> bits (CLKDIV<7:6>). ‘N2’ can be 2, 4 or 8, and must be selected such that the PLL output frequency (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 7-2: FOSC CALCULATION

$$F_{OSC} = F_{IN}*\left(\frac{M}{N1*N2}\right)$$

For example, when 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 MHz to 200 MHz range, which is 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 7-3: 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 7-2: PIC24HJ32GP202/204 AND PIC24HJ16GP304 PLL BLOCK DIAGRAM

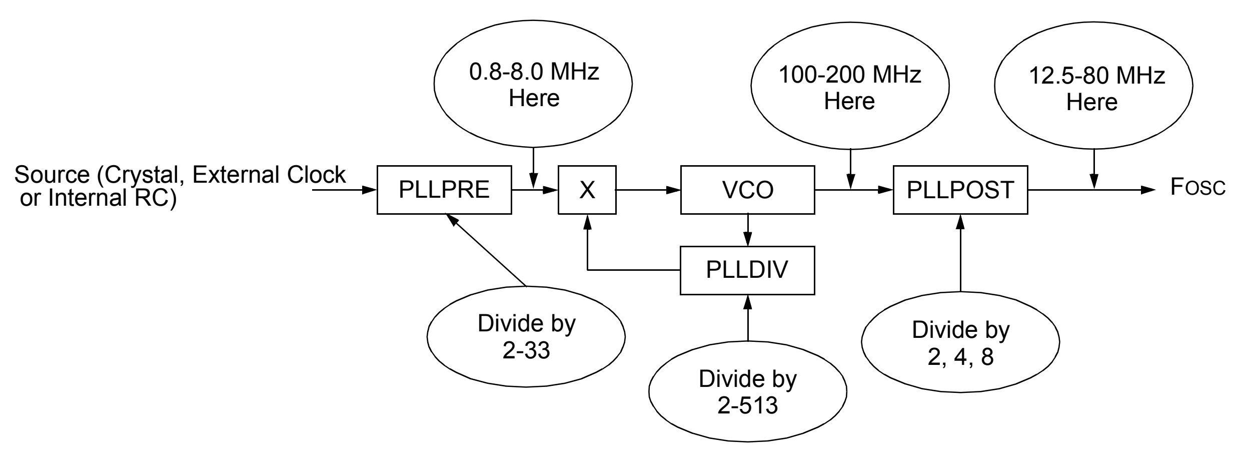


TABLE 7-1: CONFIGURATION BIT VALUES FOR CLOCK SELECTION

Oscillator Mode	Oscillator Source	POSCMD<1:0>	FNOSC<2:0>	Note
Fast RC Oscillator with Divide-by-N (FRCDIVN)	Internal	`xx`	`111`	**1, 2**
Fast RC Oscillator with Divide-by-16 (FRCDIV16)	Internal	`xx`	`110`	**1**
Low-Power RC Oscillator (LPRC)	Internal	`xx`	`101`	**1**
Secondary (Timer1) Oscillator (SOSC)	Secondary	`xx`	`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	`xx`	`001`	**1**
Fast RC Oscillator (FRC)	Internal	`xx`	`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.

REGISTER 7-1: 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	R/W-0	R-0	U-0	R/C-0	U-0	R/W-0	R/W-0
CLKLOCK	IOLOCK	LOCK	—	CF	—	LPOSCEN	OSWEN
bit 7							bit 0

Legend:	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` = Fast RC oscillator (FRC) with Divide-by-16
`111` = Fast RC oscillator (FRC) with Divide-by-n

bit 11 **Unimplemented:** Read as ‘`0`’

bit 10-8 **NOSC<2:0>:** New Oscillator Selection bits

`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` = Fast RC oscillator (FRC) with Divide-by-16
`111` = Fast RC oscillator (FRC) with Divide-by-n

bit 7 **CLKLOCK:** Clock Lock Enable bit

If clock switching is enabled and FSCM is disabled (FOSC<FCKSM> = 0b01)
`1` = Clock switching is disabled, system clock source is locked
`0` = Clock switching is enabled, system clock source can be modified by clock switching

bit 6 **IOLOCK:** Peripheral Pin Select Lock bit

`1` = Peripheral Pin Select is locked, write to peripheral pin select register is not allowed
`0` = Peripheral Pin Select is unlocked, write to peripheral pin select register is allowed

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`’

REGISTER 7-1: OSCCON: OSCILLATOR CONTROL REGISTER (CONTINUED)

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 7-2: 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(1)	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

Legend:	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
`001` = FCY/2
`010` = FCY/4
`011` = FCY/8 (default)
`100` = FCY/16
`101` = FCY/32
`110` = FCY/64
`111` = FCY/128

bit 11 **DOZEN:** DOZE Mode Enable bit(1)

`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 1 (default)
`001` = FRC divide by 2
`010` = FRC divide by 4
`011` = FRC divide by 8
`100` = FRC divide by 16
`101` = FRC divide by 32
`110` = FRC divide by 64
`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 (default)
`10` = Reserved
`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 (default)
`00001` = Input/3
•
•
•
`11111` = Input/33

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

REGISTER 7-3: PLLFBD: PLL FEEDBACK DIVISOR REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0[1]
—	—	—	—	—	—	—	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
•
•
•
`000110000` = 50 (default)
•
•
•
`111111111` = 513

REGISTER 7-4: 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
—	—	TUN<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 15-6	**Unimplemented:** Read as ‘`0`’
bit 5-0	**TUN<5:0>:** FRC Oscillator Tuning bits `011111` = Center frequency + 11.625% `011110` = Center frequency + 11.25% (8.23 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)

7.2 Clock Switching Operation

Applications are free to switch among any of the four clock sources (Primary, LP, FRC and LPRC) under software control at any time. To limit the possible side effects of this flexibility, PIC24HJ32GP202/204 and PIC24HJ16GP304 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 among the different primary submodes without reprogramming the device.

7.2.1 ENABLING CLOCK SWITCHING

To enable clock switching, the FCKSM1 Configuration bit in the Configuration register must be programmed to ‘`0`’. (Refer to **Section 18.1 “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.

7.2.2 OSCILLATOR SWITCHING SEQUENCE

Performing a clock switch requires the following basic sequence:

1. Read the COSC bits (OSCCON<14:12>) to determine the current oscillator source, if desired.
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 both of them are the same, 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 has to 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.

7.3 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.

If the PLL multiplier is used to scale the system clock, the internal FRC is also multiplied by the same factor on clock failure. Essentially, the device switches to FRC with PLL on a clock failure.

NOTES:

8.0 POWER-SAVING FEATURES

Note: This data sheet summarizes the features of the PIC24HJ32GP202/204 and PIC24HJ16GP304 devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the *"PIC24H Family Reference Manual"*.

The PIC24HJ32GP202/204 and PIC24HJ16GP304 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. PIC24HJ32GP202/204 and PIC24HJ16GP304 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 the above methods can be used to selectively customize an application's power consumption while still maintaining critical application features, such as timing-sensitive communications.

8.1 Clock Frequency and Clock Switching

PIC24HJ32GP202/204 and PIC24HJ16GP304 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 7.0 "Oscillator Configuration"**.

8.2 Instruction-Based Power-Saving Modes

PIC24HJ32GP202/204 and PIC24HJ16GP304 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. Example 8-1 shows the Assembler syntax of the `PWRSAV` instruction.

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

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

8.2.1 SLEEP MODE

In the Sleep mode,

- 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, since the system clock source is disabled.
- The LPRC clock continues to run 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. 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.

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

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

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

EXAMPLE 8-1: `PWRSAV` INSTRUCTION SYNTAX

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

8.2.2 IDLE MODE

The following occur in Idle mode:

- 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 8.4 "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 mode, the clock is reapplied to the CPU and instruction execution begins immediately, starting with the instruction following the `PWRSAV` instruction, or the first instruction in the ISR.

8.2.3 INTERRUPTS COINCIDENT WITH POWER SAVE INSTRUCTIONS

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

8.3 Doze Mode

The preferred strategies for reducing power consumption are changing clock speed and invoking one of the power-saving modes. In some circumstances, however, these are 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 can introduce communication errors, while using a power-saving mode can 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.

Programs can 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. An automatic return to full-speed CPU operation on interrupts can be 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 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.

8.4 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 using 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 enabled only if both the associated bit in the PMD register are cleared and the peripheral is supported by the specific PIC24H 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 one instruction cycle. Similarly, if a PMD bit is cleared, the corresponding module is enabled after a delay of one instruction cycle (assuming the module control registers are already configured to enable module operation).

9.0 I/O PORTS

Note:	This data sheet summarizes the features of the PIC24HJ32GP202/204 and PIC24HJ16GP304 devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the *"PIC24H Family Reference Manual"*.

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

9.1 Parallel I/O (PIO) Ports

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

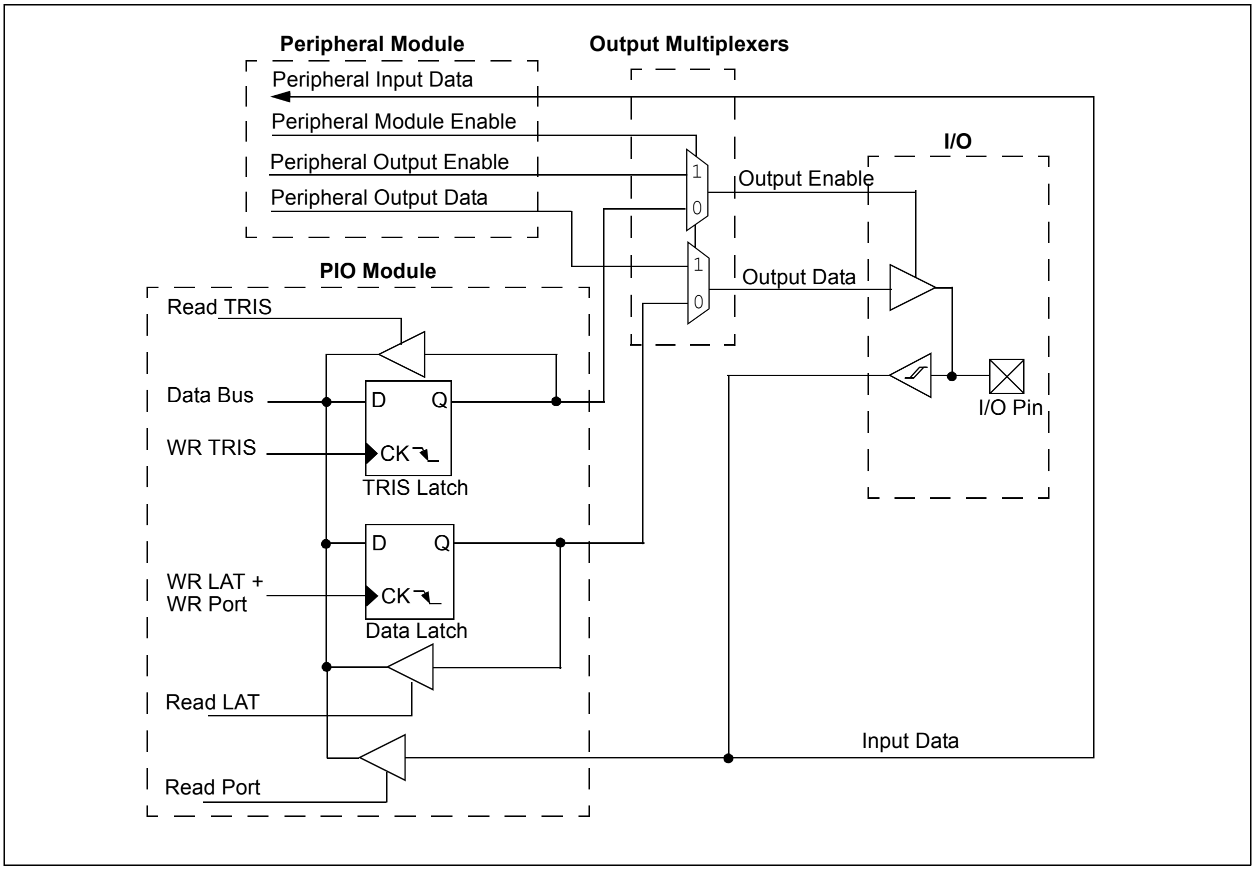
When a peripheral is enabled and the peripheral is actively driving an associated pin, the use of the pin as a general purpose output pin is disabled. The I/O pin can be read, but the output driver for the parallel port bit is disabled. If a peripheral is enabled, but the peripheral is not actively driving a pin, that pin can 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 '`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. This means that the corresponding LATx and TRISx registers and the port pin will read as zeros.

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

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



9.1.1 OPEN-DRAIN CONFIGURATION

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

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

9.2 Configuring Analog Port Pins

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

When the PORT register is read, 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.

9.2.1 I/O PORT WRITE/READ TIMING

One instruction cycle is required between a port direction change or port write operation and a read operation of the same port. Typically this instruction would be a `NOP`. An example is shown in **EXAMPLE 9-1: "Port Write/Read Example"**.

9.3 Input Change Notification

The input change notification function of the I/O ports allows the PIC24HJ32GP202/204 and PIC24HJ16GP304 devices to generate interrupt requests to the processor in response to a change-of-state on selected input pins. This feature can detect input change-of-states even in Sleep mode, when the clocks are disabled. Depending on the device pin count, up to 31 external signals (CNx pin) can be selected (enabled) for generating an interrupt request on a change-of-state.

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

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

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

EXAMPLE 9-1: PORT WRITE/READ EXAMPLE

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

9.4 Peripheral Pin Select

A major challenge in general purpose devices is providing the largest possible set of peripheral features while minimizing the conflict of features on I/O pins. The challenge is even greater on low-pin count devices. In an application where more than one peripheral must be assigned to a single pin, inconvenient workarounds in application code or a complete redesign may be the only option.

Peripheral pin select configuration enables peripheral set selection and placement on a wide range of I/O pins. By increasing the pinout options available on a particular device, programmers can better tailor the microcontroller to their entire application, rather than trimming the application to fit the device.

The peripheral pin select configuration feature operates over a fixed subset of digital I/O pins. Programmers can independently map the input and/or output of most digital peripherals to any one of these I/O pins. Peripheral pin select is performed in software, and generally does not require the device to be reprogrammed. Hardware safeguards are included that prevent accidental or spurious changes to the peripheral mapping, once it has been established.

9.4.1 AVAILABLE PINS

The peripheral pin select feature is used with a range of up to 26 pins. The number of available pins depends on the particular device and its pin count. Pins that support the peripheral pin select feature include the designation “RPn” in their full pin designation, where “RP” designates a remappable peripheral and “n” is the remappable pin number.

9.4.2 AVAILABLE PERIPHERALS

The peripheral pin select feature manages all digital-only peripherals. These include:

- General serial communications (UART and SPI)
- General-purpose timer clock inputs
- Timer-related peripherals (input capture and output compare)
- Interrupt-on-change inputs.

In comparison, some digital-only peripheral modules are never included in the peripheral pin select feature. This is because the peripheral’s function requires special I/O circuitry on a specific port and cannot be easily connected to multiple pins. These modules include I^{2}C. A similar requirement excludes all modules with analog inputs, such as the Analog-to-Digital Converter (ADC).

Remappable peripherals are not associated with a default I/O pin. The peripheral must always be assigned to a specific I/O pin before it can be used. In contrast, non remappable peripherals are always available on a default pin, assuming that the peripheral is active and not conflicting with another peripheral.

9.4.2.1 Peripheral Pin Select Function Priority

When a remappable peripheral is active on a given I/O pin, it takes priority over all other digital I/O and digital communication peripherals associated with the pin. Priority is given regardless of the type of peripheral that is mapped. Remappable peripherals never take priority over any analog functions associated with the pin.

9.4.3 CONTROLLING PERIPHERAL PIN SELECT

Peripheral pin select features are controlled through two sets of special function registers to map peripherals and to map outputs.

Since they are separately controlled, a particular peripheral’s input and output (if the peripheral has both) can be placed on any selectable function pin without constraint.

The association of a peripheral to a peripheral selectable pin is handled in two different ways, depending on whether an input or output is being mapped.

9.4.3.1 Input Mapping

The inputs of the peripheral pin select options are mapped on the basis of the peripheral. A control register associated with a peripheral dictates the pin it will be mapped to. The RPINRx registers are used to configure peripheral input mapping (see Register 9-1 through Register 9-9). Each register contains sets of 5-bit fields, with each set associated with one of the remappable peripherals. Programming a given peripheral’s bit field with an appropriate 5-bit value maps the RPn pin with that value to that peripheral. For any given device, the valid range of values for any bit field corresponds to the maximum number of peripheral pin selections supported by the device.

Figure 9-2 Illustrates remappable pin selection for U1RX input.

FIGURE 9-2: REMAPPABLE MUX INPUT FOR U1RX

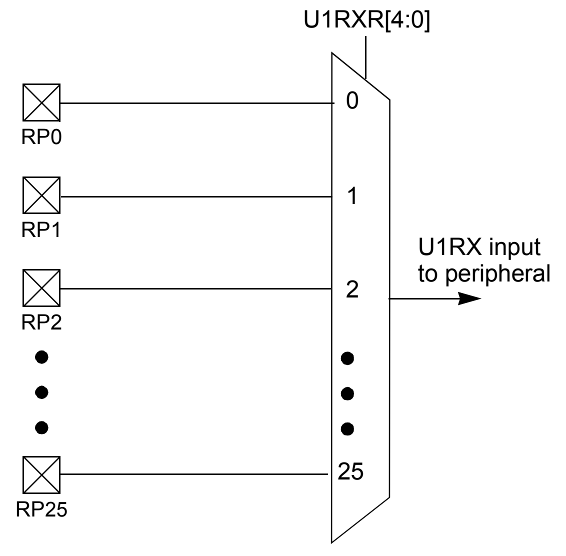


TABLE 9-1: REMAPPABLE PERIPHERAL INPUTS[1]

Input Name	Function Name	Register	Configuration Bits
External Interrupt 1	INT1	RPINR0	INT1R[4:0]
External Interrupt 2	INT2	RPINR1	INT2R[4:0]
Timer 2 External Clock	T2CK	RPINR3	T2CKR[4:0]
Timer 3 External Clock	T3CK	RPINR3	T3CKR[4:0]
Input Capture 1	IC1	RPINR7	IC1R[4:0]
Input Capture 2	IC2	RPINR7	IC2R[4:0]
Input Capture 7	IC7	RPINR10	IC7R[4:0]
Input Capture 8	IC8	RPINR10	IC8R[4:0]
Output Compare Fault A	OCFA	RPINR11	OCFAR[4:0]
UART 1 Receive	U1RX	RPINR18	U1RXR[4:0]
UART 1 Clear To Send	$\overline{\text{U1CTS}}$	RPINR18	U1CTSR[4:0]
SPI 1 Data Input	SDI1	RPINR20	SDI1R[4:0]
SPI 1 Clock Input	SCK1IN	RPINR20	SCK1R[4:0]
SPI 1 Slave Select Input	SS1IN	RPINR21	SS1R[4:0]

Note 1: Unless otherwise noted, all inputs use the Schmitt input buffers.

9.4.3.2 Output Mapping

In contrast to inputs, the outputs of the peripheral pin select options are mapped on the basis of the pin. In this case, a control register associated with a particular pin dictates the peripheral output to be mapped. The RPORx registers are used to control output mapping. Like the RPINRx registers, each register contains sets of 5-bit fields, with each set associated with one RPn pin (see Register 9-10 through Register 9-22). The value of the bit field corresponds to one of the peripherals, and that peripheral's output is mapped to the pin (see Table 9-2 and Figure 9-3).

The list of peripherals for output mapping also includes a null value of `00000` because of the mapping technique. This permits any given pin to remain unconnected from the output of any of the pin selectable peripherals.

FIGURE 9-3: MULTIPLEXING OF REMAPPABLE OUTPUT FOR RPn

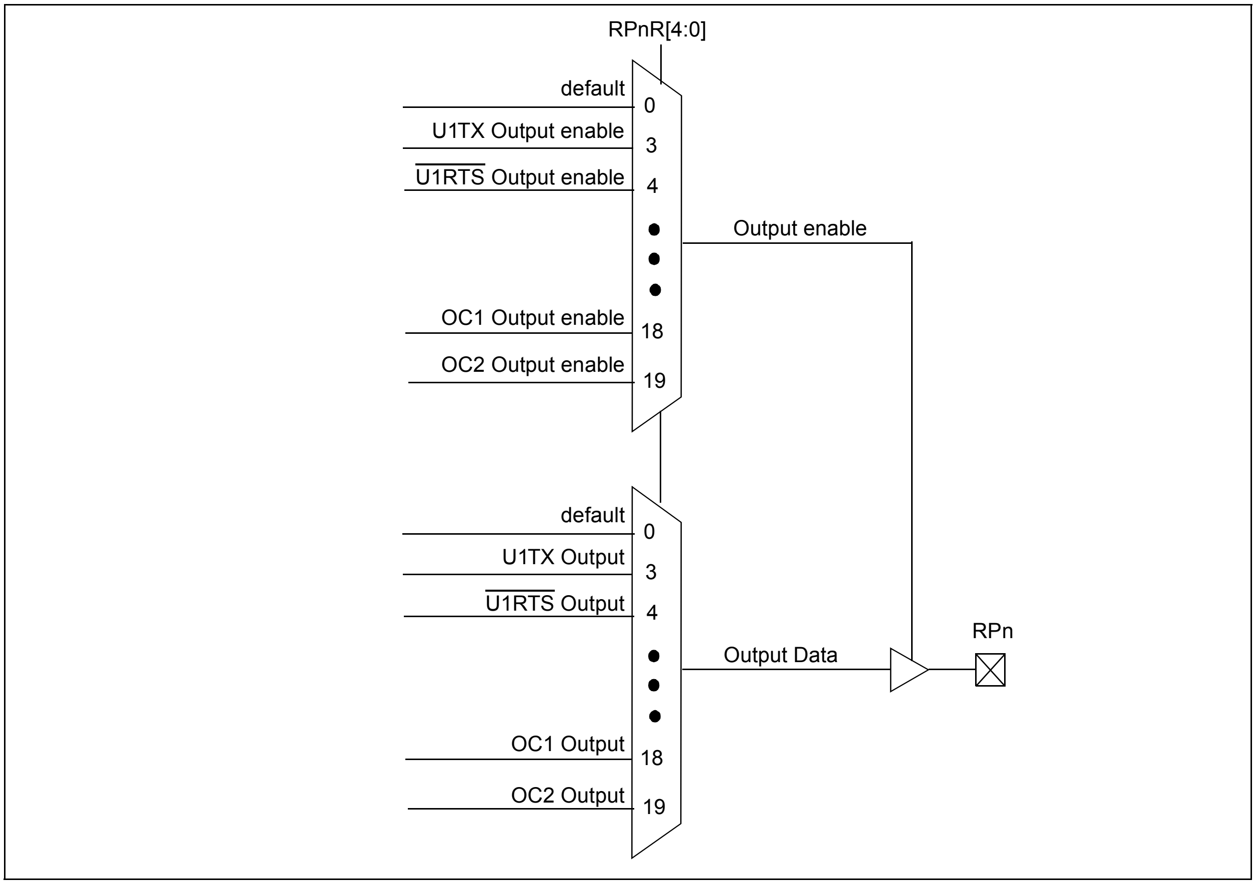


TABLE 9-2: OUTPUT SELECTION FOR REMAPPABLE PIN (RPn)

Function	RPnR<4:0>	Output Name
NULL	00000	RPn tied to default port pin
U1TX	00011	RPn tied to UART 1 Transmit
$\overline{\text{U1RTS}}$	00100	RPn tied to UART 1 Ready To Send
SDO1	00111	RPn tied to SPI 1 Data Output
SCK1OUT	01000	RPn tied to SPI 1 Clock Output
SS1OUT	01001	RPn tied to SPI 1 Slave Select Output
OC1	10010	RPn tied to Output Compare 1
OC2	10011	RPn tied to Output Compare 2

9.4.3.3 Mapping

The control schema of peripheral select pins is not limited to a small range of fixed peripheral configurations. There are no mutual or hardware-enforced lockouts between any of the peripheral mapping SFRs. Literally any combination of peripheral mappings across any or all of the RPn pins is possible. This includes both many-to-one and one-to-many mappings of peripheral inputs and outputs to pins.

While such mappings may be technically possible from a configuration point of view, they may not be supportable electrically.

9.4.4 CONTROLLING CONFIGURATION CHANGES

Because peripheral remapping can be changed during run time, some restrictions on peripheral remapping are needed to prevent accidental configuration changes. PIC24H devices include three features to prevent alterations to the peripheral map:

- Control register lock sequence
- Continuous state monitoring
- Configuration bit pin select lock

9.4.4.1 Control Register Lock

Under normal operation, writes to the RPINRx and RPORx registers are not allowed. Attempted writes appear to execute normally, but the contents of the registers remain unchanged. To change these registers, they must be unlocked in hardware. The register lock is controlled by the IOLOCK bit (OSCCON<6>). Setting IOLOCK prevents writes to the control registers; clearing IOLOCK allows writes.

To set or clear IOLOCK, a specific command sequence must be executed:

1. Write 46h to OSCCON<7:0>.
2. Write 57h to OSCCON<7:0>.
3. Clear (or set) IOLOCK as a single operation.

Note: MPLAB® C30 provides built-in C language functions for unlocking the OSCCON register:

```
__builtin_write_OSCCONL(value)
__builtin_write_OSCCONH(value)
```

See MPLAB Help for more information.

Unlike the similar sequence with the oscillator's LOCK bit, IOLOCK remains in one state until changed. This allows all the peripheral pin selects to be configured with a single unlock sequence followed by an update to all control registers, then locked with a second lock sequence.

9.4.4.2 Continuous State Monitoring

In addition to being protected from direct writes, the contents of the RPINRx and RPORx registers are constantly monitored in hardware by shadow registers. If an unexpected change in any of the registers occurs (such as cell disturbances caused by ESD or other external events), a configuration mismatch Reset will be triggered.

9.4.4.3 Configuration Bit Pin Select Lock

As an additional level of safety, the device can be configured to prevent more than one write session to the RPINRx and RPORx registers. The IOL1WAY (FOSC<IOL1WAY>) configuration bit blocks the IOLOCK bit from being cleared after it has been set once.

In the default (unprogrammed) state, IOL1WAY is set restricting the users to one write session. Programming IOL1WAY allows user applications unlimited access (with the proper use of the unlock sequence) to the peripheral pin select registers.

9.4.5 CONSIDERATIONS FOR PERIPHERAL PIN SELECTION

The ability to control peripheral pin selection introduces several considerations into application design, including several common peripherals that are only available as remappable peripherals.

9.4.5.1 Configuration

The peripheral pin selects are not available on default pins in the device's default (Reset) state. More specifically, since all RPINRx and RPORx registers reset to `0000h`, this means all peripheral pin select inputs are tied to RP0, while all peripheral pin select outputs are disconnected. This means that before any other application code is executed, the user application must initialize the device with the proper peripheral configuration.

Since the IOLOCK bit resets in the unlocked state, it is not necessary to execute the unlock sequence after the device has come out of Reset. For the sake of application safety, however, it is always a good idea to set IOLOCK and lock the configuration after writing to the control registers.

Because the unlock sequence is timing-critical, it must be executed as an assembly-language routine, in the same manner as changes to the oscillator configuration. If the bulk of the application is written in C or another high-level language, the unlock sequence should be performed by writing inline assembly.

9.4.5.2 Changing the Configuration

Choosing the configuration requires review of all peripheral pin selects and their pin assignments, especially those that will not be used in the application. In all cases, unused pin selectable peripherals should be disabled completely. Unused peripherals should have their inputs assigned to an unused RPn pin function. I/O pins with unused RPn functions should be configured with the null peripheral output.

The assignment of a peripheral to a particular pin does not automatically perform any other configuration of the pin's I/O circuitry. This means adding a pin selectable output to a pin can inadvertently drive an existing peripheral input when the output is driven. Programmers must be familiar with the behavior of other fixed peripherals that share a remappable pin, and know when to enable or disable them. To be safe, fixed digital peripherals that share the same pin should be disabled when not in use.

9.4.5.3 Pin Operation

Configuring a remappable pin for a specific peripheral does not automatically turn that feature on. The peripheral must be specifically configured for an operation and enabled, as if it were tied to a fixed pin. Where this happens in the application code (immediately following device Reset and peripheral configuration, or inside the main application routine) depends on the peripheral and its use in the application.

9.4.5.4 Analog Function

A final consideration is that peripheral pin select functions neither override analog inputs nor reconfigure pins with analog functions for digital I/O. If a pin is configured as an analog input on device Reset, it must be explicitly reconfigured as digital I/O when used with a peripheral pin select.

9.4.5.5 Configuration Example

Example 9-2 shows a configuration for bidirectional communication with flow control using UART1. The following input and output functions are used:

- Input Functions: U1RX, $\overline{\text{U1CTS}}$
- Output Functions: U1TX, $\overline{\text{U1RTS}}$

9.5 Peripheral Pin Select Registers

The PIC24HJ32GP202/204 and PIC24HJ16GP304 devices implement 17 registers for remappable peripheral configuration:

- Input Remappable Peripheral Registers (9)
- Output Remappable Peripheral Registers (8)

> **Note:** Input and Output Register values can only be changed if OSCCON[IOLOCK] = `0`. See **Section 9.4.4.1 "Control Register Lock"** for a specific command sequence.

EXAMPLE 9-2: CONFIGURING UART1 INPUT AND OUTPUT FUNCTIONS

```
//*************************************
// Unlock Registers
//*************************************
asm volatile ( "mov #OSCCONL, w1  \n"
               "mov #0x46, w2     \n"
               "mov #0x57, w3     \n"
               "mov.b w2, [w1]    \n"
               "mov.b w3, [w1]    \n"
               "bclr OSCCON, 6");

//***************************
// Configure Input Functions
// (See Table 9-1)
//***************************
    //***************************
    // Assign U1Rx To Pin RP0
    //***************************
    RPINR18bits.U1RXR = 0;

    //***************************
    // Assign U1CTS To Pin RP1
    //***************************
    RPINR18bits.U1CTSR = 1;

//***************************
// Configure Output Functions
// (See Table 9-2)
//***************************
    //***************************
    // Assign U1Tx To Pin RP2
    //***************************
    RPOR1bits.RP2R = 3;

    //***************************
    // Assign U1RTS To Pin RP3
    //***************************
    RPOR1bits.RP3R = 4;

//*************************************
// Lock Registers
//*************************************
asm volatile ( "mov #OSCCONL, w1  \n"
               "mov #0x46, w2     \n"
               "mov #0x57, w3     \n"
               "mov.b w2, [w1]    \n"
               "mov.b w3, [w1]    \n"
               "bset OSCCON, 6");
```

REGISTER 9-1: RPINR0: PERIPHERAL PIN SELECT INPUT REGISTER 0

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	INT1R<4:0>				
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	**INT1R<4:0>:** Assign External Interrupt 1 (INTR1) to the corresponding RPn pin `11111` = Input tied to Vss `11001` = Input tied to RP25 • • • `00001` = Input tied to RP1 `00000` = Input tied to RP0
bit 7-0	**Unimplemented:** Read as ‘`0`’

REGISTER 9-2: RPINR1: PERIPHERAL PIN SELECT INPUT REGISTER 1

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/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	INT2R<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	**INT2R<4:0>:** Assign External Interrupt 2 (INTR2) to the corresponding RPn pin `11111` = Input tied to Vss `11001` = Input tied to RP25 • • • `00001` = Input tied to RP1 `00000` = Input tied to RP0

REGISTER 9-3: RPINR3: PERIPHERAL PIN SELECT INPUT REGISTER 3

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	T3CKR<4:0>				
bit 15							bit 8

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	T2CKR<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 **Unimplemented:** Read as ‘0’

bit 12-8 **T3CKR<4:0>:** Assign Timer3 External Clock (T3CK) to the Corresponding RPn pin

11111 = Input tied to Vss
11001 = Input tied to RP25
•
•
•
00001 = Input tied to RP1
00000 = Input tied to RP0

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

bit 4-0 **T2CKR<4:0>:** Assign Timer2 External Clock (T2CK) to the Corresponding RPn pin

11111 = Input tied to Vss
11001 = Input tied to RP25
•
•
•
00001 = Input tied to RP1
00000 = Input tied to RP0

REGISTER 9-4: RPINR7: PERIPHERAL PIN SELECT INPUT REGISTER 7

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	IC2R<4:0>				
bit 15							bit 8

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	IC1R<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 **Unimplemented:** Read as ‘0’

bit 12-8 **IC2R<4:0>:** Assign Input Capture 2 (IC2) to the corresponding RPn pin

T2CKR<4:0>: Assign Timer2 External Clock (T2CK) to the Corresponding RPn pin

`11111` = Input tied to Vss
`11001` = Input tied to RP25
•
•
•
`00001` = Input tied to RP1
`00000` = Input tied to RP0

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

bit 4-0 **IC1R<4:0>:** Assign Input Capture 1 (IC1) to the corresponding RPn pin

T2CKR<4:0>: Assign Timer2 External Clock (T2CK) to the Corresponding RPn pin

`11111` = Input tied to Vss
`11001` = Input tied to RP25
•
•
•
`00001` = Input tied to RP1
`00000` = Input tied to RP0

REGISTER 9-5: RPIR10: PERIPHERAL PIN SELECT INPUT REGISTERS 10

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	IC8R<4:0>				
bit 15							bit 8

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	IC7R<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 **Unimplemented:** Read as ‘0’

bit 12-8 **IC8R<4:0>:** Assign Input Capture 8 (IC8) to the corresponding pin RPn pin

`11111` = Input tied to Vss
`11001` = Input tied to RP25
•
•
•
`00001` = Input tied to RP1
`00000` = Input tied to RP0

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

bit 4-0 **IC7R<4:0>:** Assign Input Capture 7 (IC7) to the corresponding pin RPn pin

`11111` = Input tied to Vss
`11001` = Input tied to RP25
•
•
•
`00001` = Input tied to RP1
`00000` = Input tied to RP0

REGISTER 9-6: RPINR11: PERIPHERAL PIN SELECT INPUT REGISTER 11

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/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	OCFAR<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	**OCFAR<4:0>:** Assign Output Capture A (OCFA) to the corresponding RPn pin 11111 = Input tied to Vss 11001 = Input tied to RP25 • • • 00001 = Input tied to RP1 00000 = Input tied to RP0

REGISTER 9-7: RPINR18: PERIPHERAL PIN SELECT INPUT REGISTER 18

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	U1CTSR<4:0>				
bit 15							bit 8

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	U1RXR<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 **Unimplemented:** Read as ‘`0`’

bit 12-8 **U1CTSR<4:0>:** Assign UART 1 Clear to Send ($\overline{\text{U1CTS}}$) to the corresponding RPn pin

`11111` = Input tied to Vss
`11001` = Input tied to RP25
•
•
•
`00001` = Input tied to RP1
`00000` = Input tied to RP0

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

bit 4-0 **U1RXR<4:0>:** Assign UART 1 Receive (U1RX) to the corresponding RPn pin

`11111` = Input tied to Vss
`11001` = Input tied to RP25
•
•
•
`00001` = Input tied to RP1
`00000` = Input tied to RP0

REGISTER 9-8: RPINR20: PERIPHERAL PIN SELECT INPUT REGISTER 20

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	SCK1R<4:0>				
bit 15							bit 8

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	SDI1R<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 **Unimplemented:** Read as ‘`0`’

bit 12-8 **SCK1R<4:0>:** Assign SPI 1 Clock Input (SCK1IN) to the corresponding RPn pin

`11111` = Input tied to Vss
`11001` = Input tied to RP25
•
•
•
`00001` = Input tied to RP1
`00000` = Input tied to RP0

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

bit 4-0 **SDI1R<4:0>:** Assign SPI 1 Data Input (SDI1) to the corresponding RPn pin

`11111` = Input tied to Vss
`11001` = Input tied to RP25
•
•
•
`00001` = Input tied to RP1
`00000` = Input tied to RP0

REGISTER 9-9: RPINR21: PERIPHERAL PIN SELECT INPUT REGISTER 21

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/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	SS1R<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 **SS1R<4:0>:** Assign SPI1 Slave Select Input (SS1IN) to the Corresponding RPn pin

`11111` = Input tied to Vss
`11001` = Input tied to RP25
•
•
•
`00001` = Input tied to RP1
`00000` = Input tied to RP0

REGISTER 9-10: RPOR0: PERIPHERAL PIN SELECT OUTPUT REGISTERS 0

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP1R<4:0>				
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
—	—	—	RP0R<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	**Unimplemented:** Read as ‘0’
bit 12-8	**RP1R<4:0>:** Peripheral Output Function is Assigned to RP1 Output Pin (see Table 9-2 for peripheral function numbers)
bit 7-5	**Unimplemented:** Read as ‘0’
bit 4-0	**RP0R<4:0>:** Peripheral Output Function is Assigned to RP0 Output Pin (see Table 9-2 for peripheral function numbers)

REGISTER 9-11: RPOR1: PERIPHERAL PIN SELECT OUTPUT REGISTERS 1

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP3R<4:0>				
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
—	—	—	RP2R<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	**Unimplemented:** Read as ‘0’
bit 12-8	**RP3R<4:0>:** Peripheral Output Function is Assigned to RP3 Output Pin (see Table 9-2 for peripheral function numbers)
bit 7-5	**Unimplemented:** Read as ‘0’
bit 4-0	**RP2R<4:0>:** Peripheral Output Function is Assigned to RP2 Output Pin (see Table 9-2 for peripheral function numbers)

REGISTER 9-12: RPOR2: PERIPHERAL PIN SELECT OUTPUT REGISTERS 2

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP5R<4:0>				
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
—	—	—	RP4R<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	**Unimplemented:** Read as ‘0’
bit 12-8	**RP5R<4:0>:** Peripheral Output Function is Assigned to RP5 Output Pin (see Table 9-2 for peripheral function numbers)
bit 7-5	**Unimplemented:** Read as ‘0’
bit 4-0	**RP4R<4:0>:** Peripheral Output Function is Assigned to RP4 Output Pin (see Table 9-2 for peripheral function numbers)

REGISTER 9-13: RPOR3: PERIPHERAL PIN SELECT OUTPUT REGISTERS 3

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP7R<4:0>				
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
—	—	—	RP6R<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	**Unimplemented:** Read as ‘0’
bit 12-8	**RP7R<4:0>:** Peripheral Output Function is Assigned to RP7 Output Pin (see Table 9-2 for peripheral function numbers)
bit 7-5	**Unimplemented:** Read as ‘0’
bit 4-0	**RP6R<4:0>:** Peripheral Output Function is Assigned to RP6 Output Pin (see Table 9-2 for peripheral function numbers)

REGISTER 9-14: RPOR4: PERIPHERAL PIN SELECT OUTPUT REGISTERS 0

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP9R<4:0>				
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
—	—	—	RP8R<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	**Unimplemented:** Read as ‘0’
bit 12-8	**RP9R<4:0>:** Peripheral Output Function is Assigned to RP9 Output Pin (see Table 9-2 for peripheral function numbers)
bit 7-5	**Unimplemented:** Read as ‘0’
bit 4-0	**RP8R<4:0>:** Peripheral Output Function is Assigned to RP8 Output Pin (see Table 9-2 for peripheral function numbers)

REGISTER 9-15: RPOR5: PERIPHERAL PIN SELECT OUTPUT REGISTERS 5

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP11R<4:0>				
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
—	—	—	RP10R<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	**Unimplemented:** Read as ‘0’
bit 12-8	**RP11R<4:0>:** Peripheral Output Function is Assigned to RP11 Output Pin (see Table 9-2 for peripheral function numbers)
bit 7-5	**Unimplemented:** Read as ‘0’
bit 4-0	**RP10R<4:0>:** Peripheral Output Function is Assigned to RP10 Output Pin (see Table 9-2 for peripheral function numbers)

REGISTER 9-16: RPOR6: PERIPHERAL PIN SELECT OUTPUT REGISTERS 6

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP13R<4:0>				
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
—	—	—	RP12R<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	**Unimplemented:** Read as ‘0’
bit 12-8	**RP13R<4:0>:** Peripheral Output Function is Assigned to RP13 Output Pin (see Table 9-2 for peripheral function numbers)
bit 7-5	**Unimplemented:** Read as ‘0’
bit 4-0	**RP12R<4:0>:** Peripheral Output Function is Assigned to RP12 Output Pin (see Table 9-2 for peripheral function numbers)

REGISTER 9-17: RPOR7: PERIPHERAL PIN SELECT OUTPUT REGISTERS 7

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP15R<4:0>				
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
—	—	—	RP14R<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	**Unimplemented:** Read as ‘0’
bit 12-8	**RP15R<4:0>:** Peripheral Output Function is Assigned to RP15 Output Pin (see Table 9-2 for peripheral function numbers)
bit 7-5	**Unimplemented:** Read as ‘0’
bit 4-0	**RP14R<4:0>:** Peripheral Output Function is Assigned to RP14 Output Pin (see Table 9-2 for peripheral function numbers)

REGISTER 9-18: RPOR7: PERIPHERAL PIN SELECT OUTPUT REGISTERS 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP17R<4:0>				
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
—	—	—	RP16R<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	**Unimplemented:** Read as ‘0’
bit 12-8	**RP15R<4:0>:** Peripheral Output Function is Assigned to RP15 Output Pin (see Table 9-2 for peripheral function numbers)
bit 7-5	**Unimplemented:** Read as ‘0’
bit 4-0	**RP14R<4:0>:** Peripheral Output Function is Assigned to RP14 Output Pin (see Table 9-2 for peripheral function numbers)

REGISTER 9-19: RPOR7: PERIPHERAL PIN SELECT OUTPUT REGISTERS 9

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP19R<4:0>				
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
—	—	—	RP18R<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	**Unimplemented:** Read as ‘0’
bit 12-8	**RP19R<4:0>:** Peripheral Output Function is Assigned to RP19 Output Pin (see Table 9-2 for peripheral function numbers)
bit 7-5	**Unimplemented:** Read as ‘0’
bit 4-0	**RP18R<4:0>:** Peripheral Output Function is Assigned to RP18 Output Pin (see Table 9-2 for peripheral function numbers)

REGISTER 9-20: RPOR7: PERIPHERAL PIN SELECT OUTPUT REGISTERS 10

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP21R<4:0>				
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
—	—	—	RP20R<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	**Unimplemented:** Read as ‘0’
bit 12-8	**RP21R<4:0>:** Peripheral Output Function is Assigned to RP21 Output Pin (see Table 9-2 for peripheral function numbers)
bit 7-5	**Unimplemented:** Read as ‘0’
bit 4-0	**RP20R<4:0>:** Peripheral Output Function is Assigned to RP20 Output Pin (see Table 9-2 for peripheral function numbers)

REGISTER 9-21: RPOR7: PERIPHERAL PIN SELECT OUTPUT REGISTERS 11

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP23R<4:0>				
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
—	—	—	RP22R<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	**Unimplemented:** Read as ‘0’
bit 12-8	**RP23R<4:0>:** Peripheral Output Function is Assigned to RP23 Output Pin (see Table 9-2 for peripheral function numbers)
bit 7-5	**Unimplemented:** Read as ‘0’
bit 4-0	**RP22R<4:0>:** Peripheral Output Function is Assigned to RP22 Output Pin (see Table 9-2 for peripheral function numbers)

REGISTER 9-22: RPOR7: PERIPHERAL PIN SELECT OUTPUT REGISTERS 12

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP25R<4:0>				
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
—	—	—	RP24R<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	**Unimplemented:** Read as ‘0’
bit 12-8	**RP25R<4:0>:** Peripheral Output Function is Assigned to RP25 Output Pin (see Table 9-2 for peripheral function numbers)
bit 7-5	**Unimplemented:** Read as ‘0’
bit 4-0	**RP24R<4:0>:** Peripheral Output Function is Assigned to RP24 Output Pin (see Table 9-2 for peripheral function numbers)

NOTES:

10.0 TIMER1

Note:	This data sheet summarizes the features of the PIC24HJ32GP202/204 and PIC24HJ16GP304 devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the *“PIC24H Family Reference Manual”*.

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

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

Timer1 also supports these features:

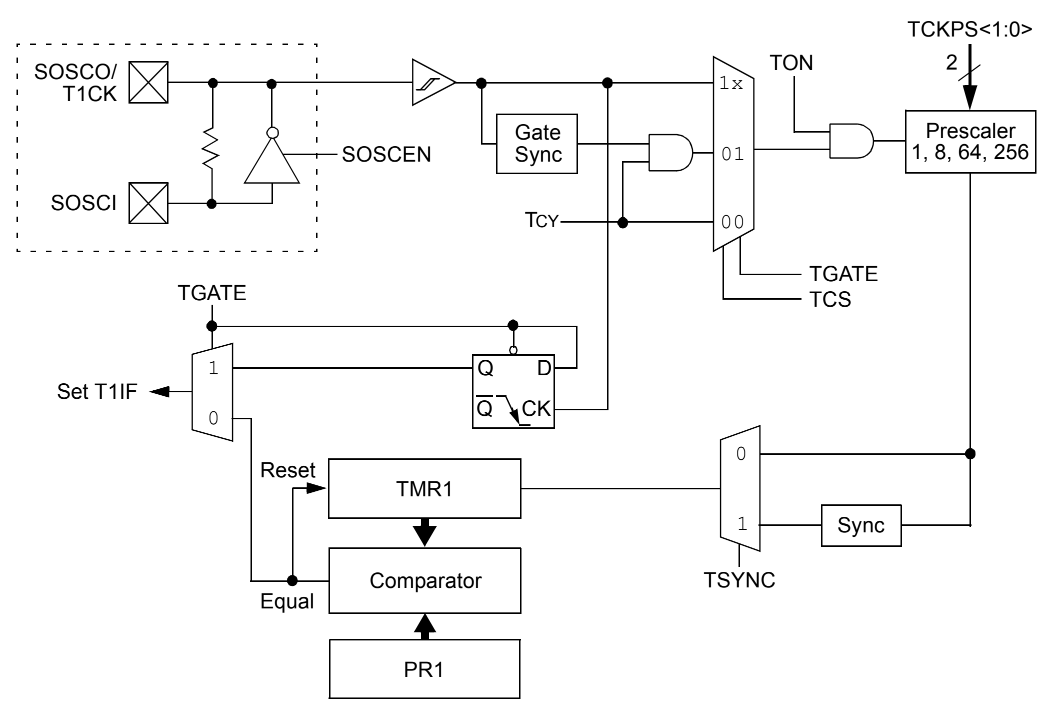
- Timer gate operation
- Selectable prescaler settings
- Timer operation during CPU Idle and Sleep modes
- Interrupt on 16-bit Period register match or falling edge of external gate signal

Figure 10-1 shows 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 10-1: 16-BIT TIMER1 MODULE BLOCK DIAGRAM



REGISTER 10-1: 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’

11.0 TIMER2/3 FEATURE

Note:	This data sheet summarizes the features of the PIC24HJ32GP202/204 and PIC24HJ16GP304 devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the *"PIC24H Family Reference Manual"*.

The Timer2/3 feature has 32-bit timers that can also be configured as two independent 16-bit timers with selectable operating modes.

As a 32-bit timer, the Timer2/3 feature permits operation in three modes:

- Two Independent 16-bit timers (Timer2 and Timer3) with all 16-bit operating modes (except Asynchronous Counter mode)
- Single 32-bit timer (Timer2/3)
- Single 32-bit synchronous counter (Timer2/3)

The Timer2/3 feature also supports:

- 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)
- ADC1 Event Trigger (Timer2/3 only)

Individually, all eight of the 16-bit timers can function as synchronous timers or counters. They also offer the features that are listed above, except for the event trigger. The operating modes and enabled features are determined by setting the appropriate bit(s) in the T2CON and T3CON registers. T2CON registers are shown in generic form in Register 11-1. T3CON registers are shown in Register 11-2.

For 32-bit timer/counter operation, Timer2 is the Least Significant word, and Timer3 is the Most Significant word of the 32-bit timers.

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

11.1 32-bit Operation

To configure the Timer2/3 feature for 32-bit operation:

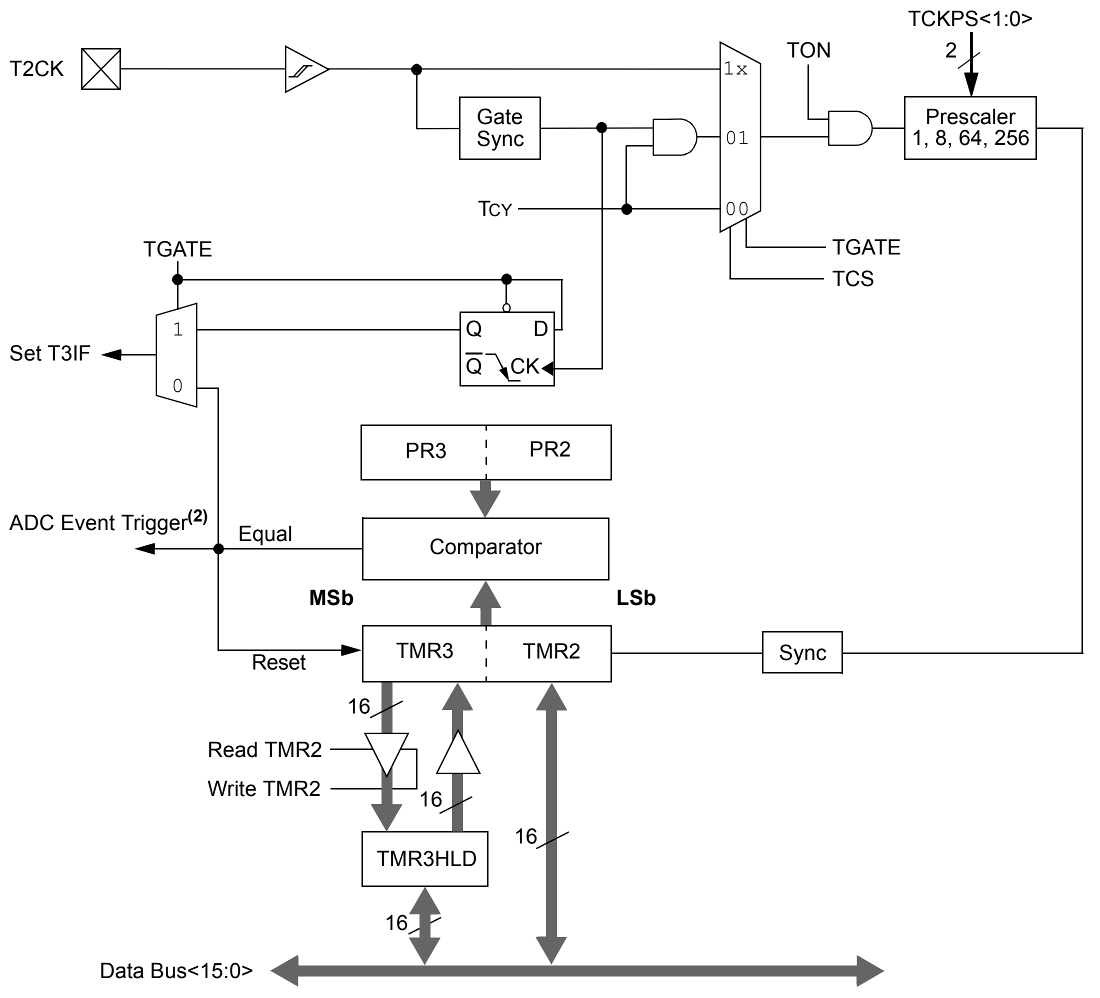
1. Set the corresponding T32 control bit.
2. Select the prescaler ratio for Timer2 using the TCKPS<1:0> bits.
3. Set the Clock and Gating modes using the corresponding TCS and TGATE bits.
4. Load the timer period value. PR3 contains the Most Significant word of the value, while PR2 contains the Least Significant word.
5. Set the interrupt enable bit T3IE, if interrupts are required. Use the priority bits T3IP<2:0> to set the interrupt priority. While Timer2 controls the timer, the interrupt appears as a Timer3 interrupt.
6. Set the corresponding TON bit.

The timer value at any point is stored in the register pair TMR3:TMR2. TMR3 always contains the Most Significant word of the count, while TMR2 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.

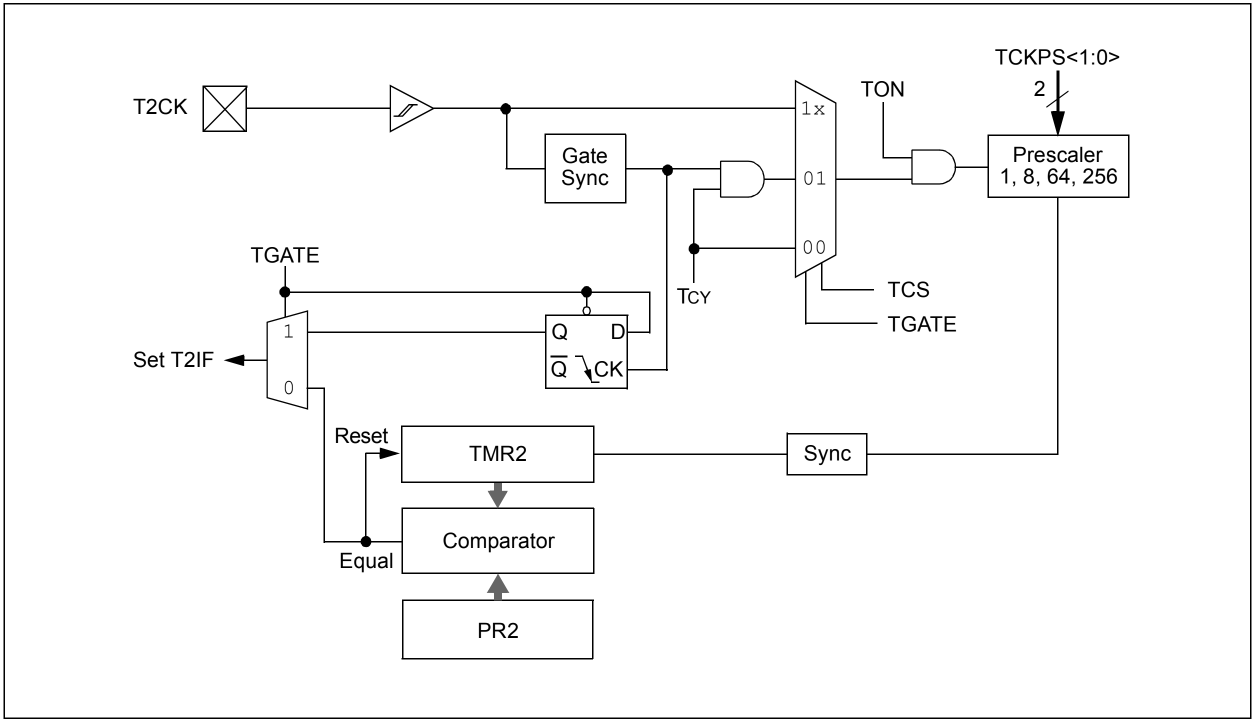
FIGURE 11-1: TIMER2/3 (32-BIT) BLOCK DIAGRAM(1)



Note 1: The 32-bit timer control bit, T32, must be set for 32-bit timer/counter operation. All control bits are respective to the T2CON register.

2: The ADC event trigger is available only on Timer2/3.

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



REGISTER 11-1: T2CON 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[1]	—	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:** Timer2 On bit

When T32 = `1`:
`1` = Starts 32-bit Timer2/3
`0` = Stops 32-bit Timer2/3

When T32 = `0`:
`1` = Starts 16-bit Timer2
`0` = Stops 16-bit Timer2

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:** Timer2 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>:** Timer2 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[1]
`1` = Timer2 and Timer3 form a single 32-bit timer
`0` = Timer2 and Timer3 act as two 16-bit timers

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

bit 1 **TCS:** Timer2 Clock Source Select bit
`1` = External clock from pin T2CK (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 11-2: T3CON CONTROL REGISTER

R/W-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
TON[1]	—	TSIDL[1]	—	—	—	—	—
bit 15							bit 8

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

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

bit 15 **TON:** Timer3 On bit[1]
`1` = Starts 16-bit Timer3
`0` = Stops 16-bit Timer3

bit 14 **Unimplemented:** Read as ‘`0`’

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

bit 12-7 **Unimplemented:** Read as ‘`0`’

bit 6 **TGATE:** Timer3 Gated Time Accumulation Enable bit[1]
When TCS = `1`:
This bit is ignored.
When TCS = `0`:
`1` = Gated time accumulation enabled
`0` = Gated time accumulation disabled

bit 5-4 **TCKPS<1:0>:** Timer3 Input Clock Prescale Select bits[1]
`11` = 1:256
`10` = 1:64
`01` = 1:8
`00` = 1:1

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

bit 1 **TCS:** Timer3 Clock Source Select bit[1]
`1` = External clock from pin T3CK (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 Timer3 operation; all timer functions are set through T2CON.

NOTES:

12.0 INPUT CAPTURE

Note: This data sheet summarizes the features of the PIC24HJ32GP202/204 and PIC24HJ16GP304 devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the *"PIC24H Family Reference Manual"*.

The input capture module is useful in applications requiring frequency (period) and pulse measurement. The PIC24HJ32GP202/204 and PIC24HJ16GP304 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:

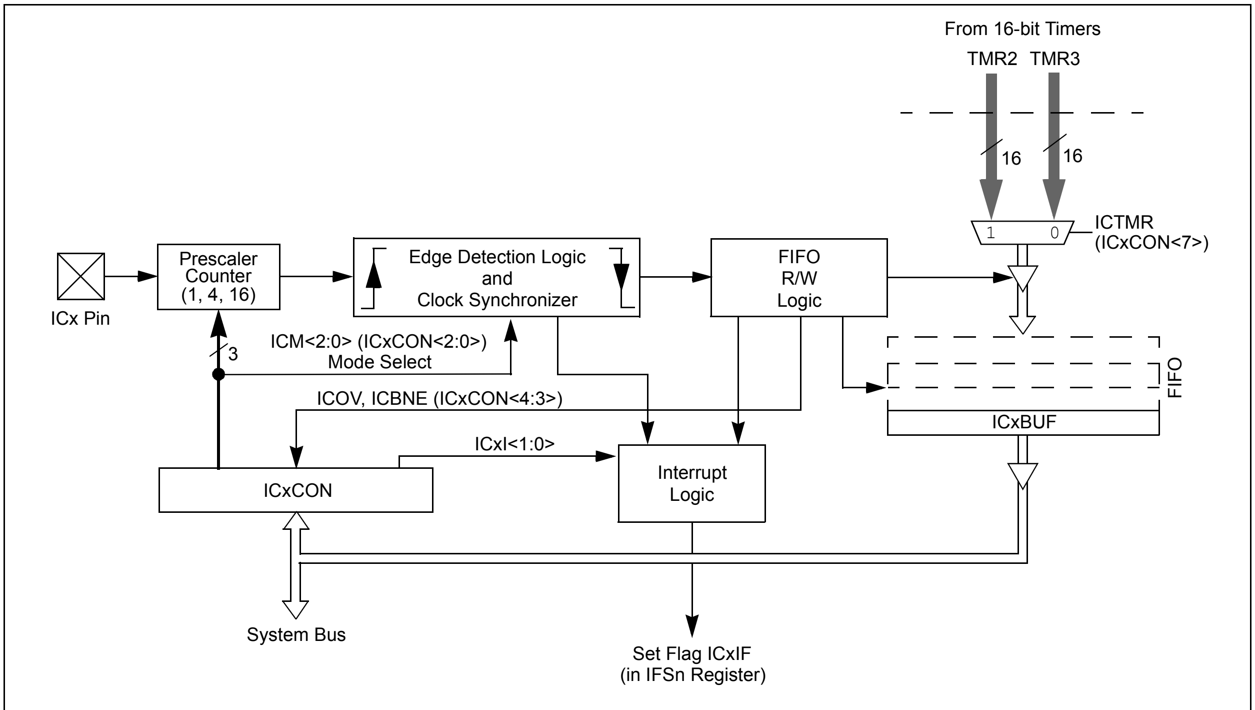
- Simple Capture Event modes:
  - Capture timer value on every falling edge of input at ICx pin
  - Capture timer value on every rising edge of input at ICx pin
- Capture timer value on every edge (rising and falling)
- Prescaler Capture Event modes:
  - Capture timer value on every 4th rising edge of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

Each input capture channel can select one of the 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
- Four-word FIFO buffer for capture values
  - Interrupt optionally generated after 1, 2, 3 or 4 buffer locations are filled
- Use of input capture to provide additional sources of external interrupts

FIGURE 12-1: INPUT CAPTURE BLOCK DIAGRAM



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

12.1 Input Capture Registers

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

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	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 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

13.0 OUTPUT COMPARE

Note: This data sheet summarizes the features of the PIC24HJ32GP202/204 and PIC24HJ16GP304 devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the *"PIC24H Family Reference Manual"*.

13.1 Setup for Single Output Pulse Generation

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

To generate a single output pulse, the following steps are required. These steps assume timer source is initially turned off though 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 value computed in step 2 into the Output Compare register, OCxR, and the value computed in step 3 into the Output Compare Secondary register, OCxRS.
5. Set Timer Period register, PRy, to a 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. Upon the first match between TMRy and OCxR, the OCx pin will be driven high.

   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. This results 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"**.
8. change the Timer and Compare register settings to initiate another single pulse output, if needed; and then issue a write to set the OCM bits to '`100`'. Disabling and re-enabling the timer, and clearing the TMRy register, are not required, but may be advantageous for defining a pulse from a known event time boundary.

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

13.2 Setup for Continuous Output Pulse Generation

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

To configure the module to generate a continuous stream of output pulses, the following steps are required. These steps assume that the timer source is initially turned off but this is not a requirement for the module operation.

1. Determine the instruction clock cycle time. Take into account the frequency of the external clock to the timer source (if one is used) and the timer prescaler settings.
2. Calculate time to the rising edge of the output pulse relative to the TMRy start value (0000h).
3. Calculate the time to the falling edge of the pulse, based on the desired pulse width and the time to the rising edge of the pulse.
4. Write the values computed in step 2 into the Output Compare register, OCxR, and value computed in step 3 into the Output Compare Secondary register, OCxRS.
5. Set Timer Period register, PRy, to a 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`'. Upon the first match between TMRy and OCxR, the OCx pin will be driven high.

   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.
8. As a result of the second compare match event, the OCxIF interrupt flag bit is set.

   When the compare time base and the value in its respective Timer Period register match, the TMRy register resets to 0x0000 and resumes counting.
9. 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.

13.3 Pulse-Width Modulation Mode

Use the following steps 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 the two PWM operation modes by writing to the Output Compare Mode bits, OCM<2:0> and (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.

13.3.1 PWM PERIOD

The PWM period is specified by writing to PRy, the Timer Period register. The PWM period can be calculated using Equation 13-1:

EQUATION 13-1: CALCULATING THE PWM PERIOD

$$\text{PWM Period} = [(\text{PRy}) + 1] \bullet T_{CY} \bullet (\text{Timer Prescale Value})$$

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.

13.3.2 PWM DUTY CYCLE

Specify the PWM duty cycle 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 13-1 for PWM mode timing details. Table 13-1 shows an example of PWM frequencies and resolutions for a device operating at 10 MIPS.

EQUATION 13-2: CALCULATION FOR MAXIMUM PWM RESOLUTION

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

EXAMPLE 13-1: 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.

   TCY = 62.5 ns
   PWM Period = 1/PWM Frequency = 1/52.08 kHz = 19.2 ms
   PWM Period = (PR2 + 1) • TCY • (Timer2 Prescale Value)
   19.2 ms = (PR2 + 1) • 62.5 ns • 1
   PR2 = 306

2. Find the maximum resolution of the duty cycle that can be used with a 52.08 kHz frequency and a 32 MHz device clock rate:

   PWM Resolution = $\log_{10}$(FCY/FPWM)/$\log_{10}2$) bits
   = ($\log_{10}$(16 MHz/52.08 kHz)/$\log_{10}2$) bits
   = 8.3 bits

TABLE 13-1: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 4 MIPS (FCY = 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

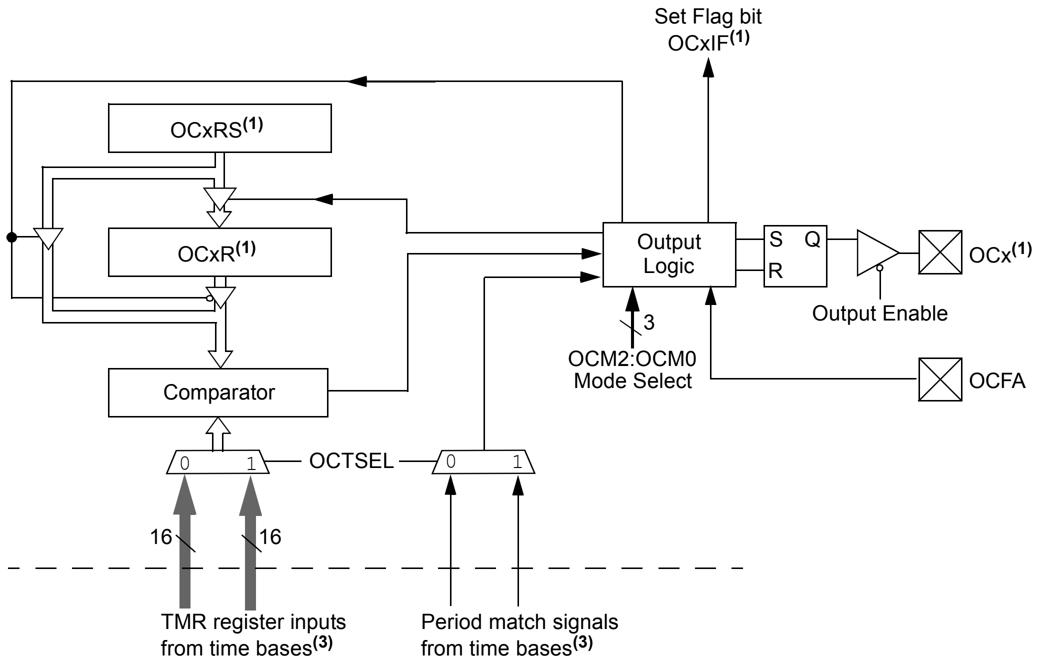
TABLE 13-2: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 16 MIPS (FCY = 16 MHz)

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 13-3: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 40 MIPS (FCY = 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 13-1: OUTPUT COMPARE MODULE BLOCK DIAGRAM



Note 1: Where 'x' is shown, reference is made to the registers associated with the respective output compare channels 1 through 8.

2: OCFA pin controls OC1-OC2 channels.

3: TMR2/TMR3 can be selected via OCTSEL(OCxOCN<3>) bit.

13.4 Output Compare Register

REGISTER 13-1: 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	OCM<2:0>		
bit 7							bit 0

Legend:	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 hardware 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 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

NOTES:

14.0 SERIAL PERIPHERAL INTERFACE (SPI)

Note:	This data sheet summarizes the features of the PIC24HJ32GP202/204 and PIC24HJ16GP304 devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the *"PIC24H Family Reference Manual"*.

The Serial Peripheral Interface (SPI) module is a synchronous serial interface useful for communicating with other peripheral or microcontroller devices. These peripheral devices can be serial EEPROMs, shift registers, display drivers, Analog-to-Digital Converters (ADC) and so on. The SPI module is compatible with SPI and SIOP from Motorola®.

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 status conditions.

The serial interface consists of 4 pins:

- SDIx (serial data input)
- SDOx (serial data output)
- SCKx (shift clock input or output)
- SSx (active low slave select).

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

14.1 Interrupts

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

14.2 Receive Operations

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 sets the SPIROV bit, indicating an overflow condition. The transfer of the data from SPIxSR to SPIxBUF is not completed, and the new data is lost. The module will not respond to SCL transitions while SPIROV is '1', effectively disabling the module until SPIxBUF is read by user software.

14.3 Transmit Operations

Transmit writes are also double-buffered. The user application 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. Do not perform read-modify-write operations (such as bit-oriented instructions) on the SPIxBUF register.

14.4 SPI Setup

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.

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 set the SSEN bit (SPIxCON1<7>) to enable the $\overline{\text{SSx}}$ 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.

FIGURE 14-1: SPI MODULE BLOCK DIAGRAM

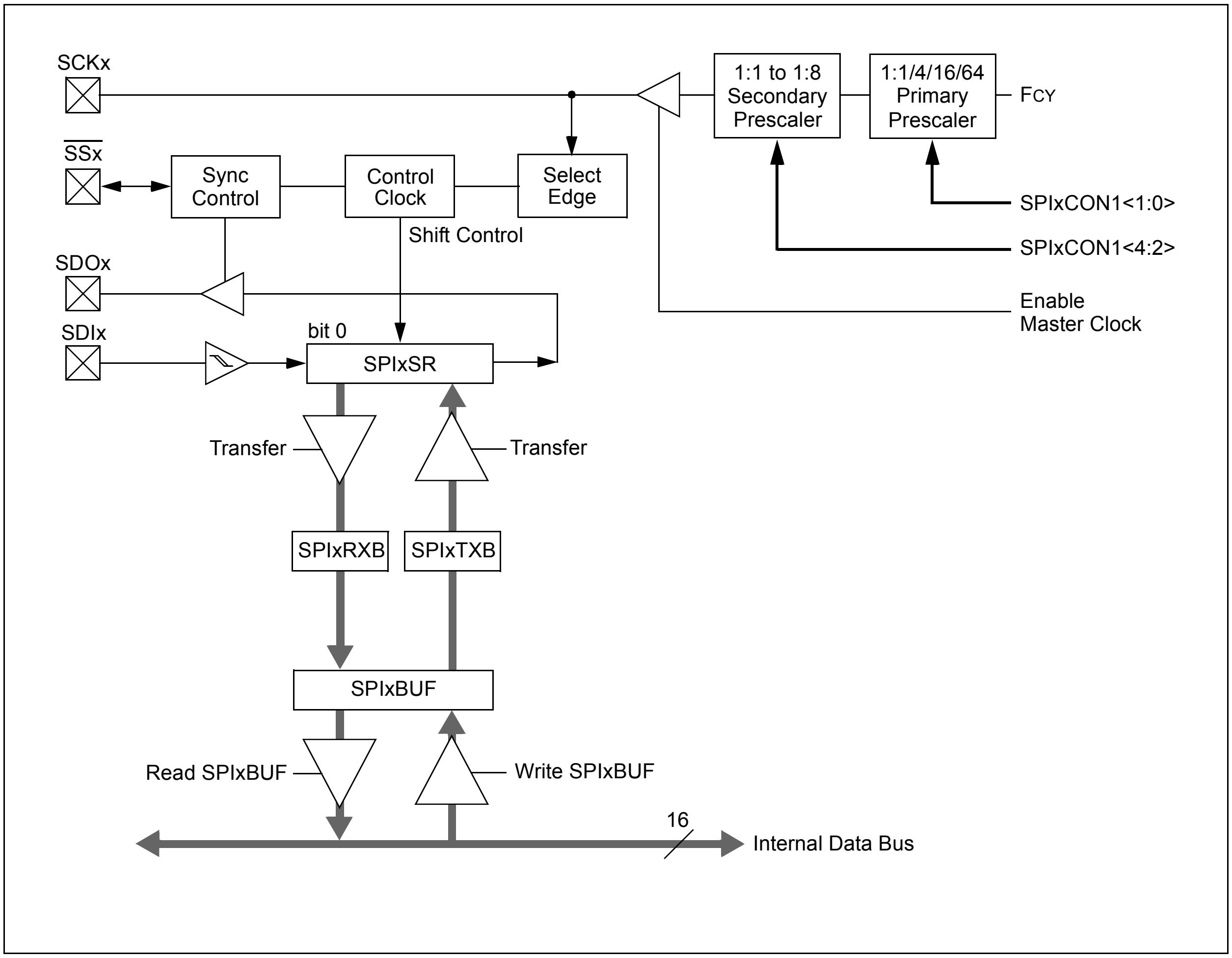
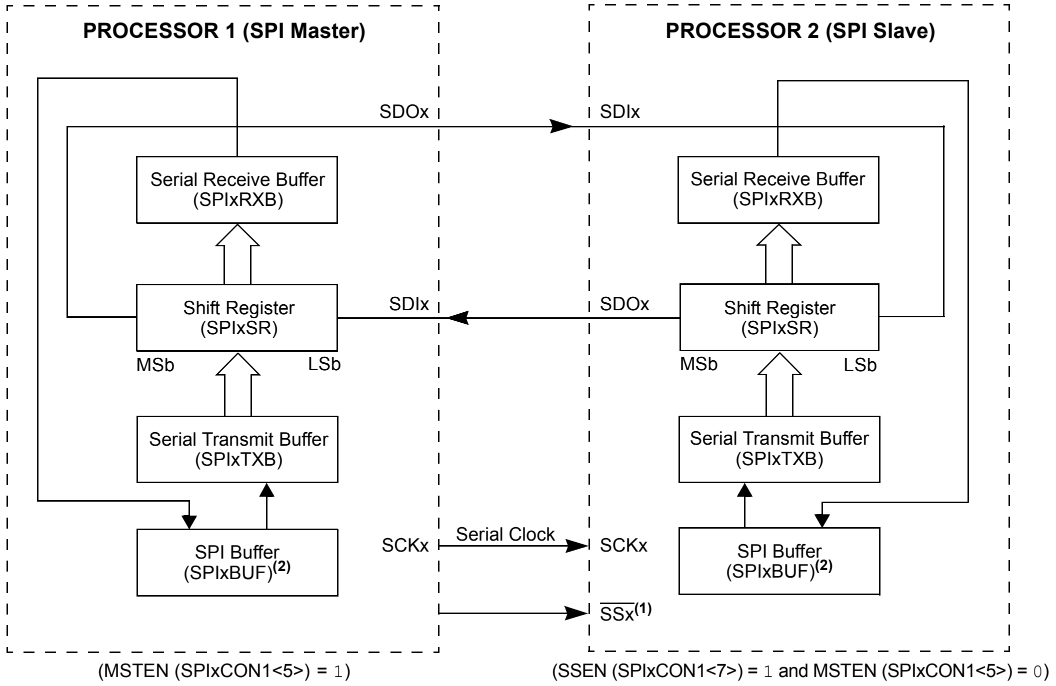


FIGURE 14-2: SPI MASTER/SLAVE CONNECTION



Note 1: Using the SSx pin in Slave mode of operation is optional.

2: User application must write transmit data to read received data from SPIxBUF. The SPIxTXB and SPIxRXB registers are memory mapped to SPIxBUF.

FIGURE 14-3: SPI MASTER AND FRAME MASTER CONNECTION DIAGRAM

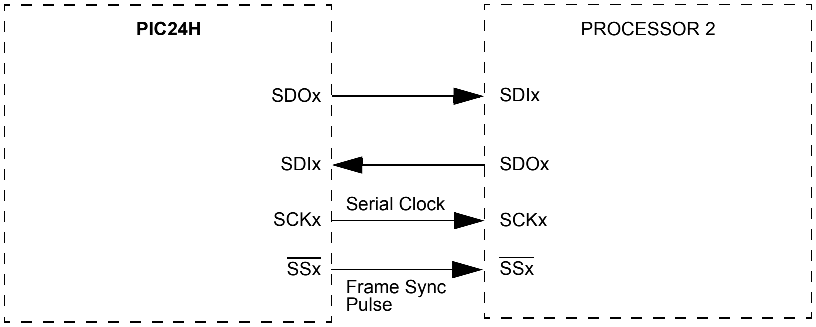


FIGURE 14-4: SPI MASTER AND FRAME SLAVE CONNECTION DIAGRAM

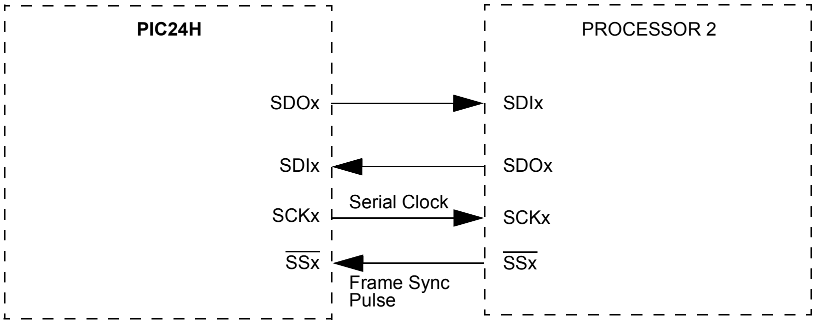


FIGURE 14-5: SPI SLAVE AND FRAME MASTER CONNECTION DIAGRAM

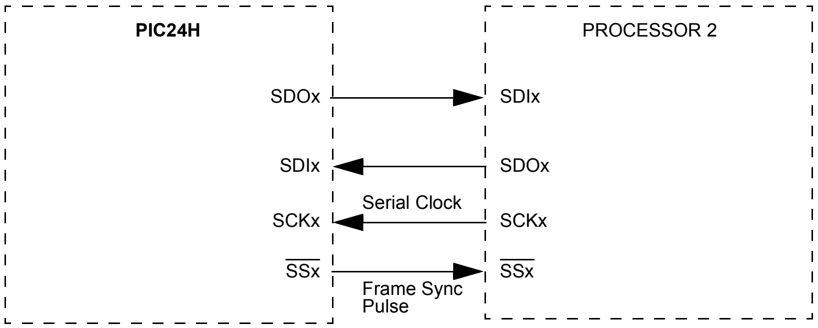
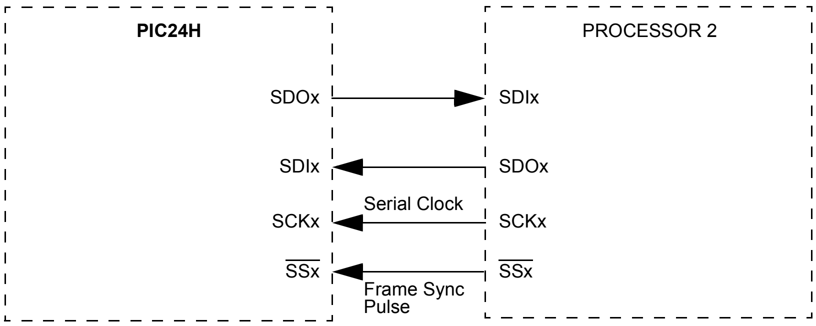


FIGURE 14-6: SPI SLAVE, FRAME SLAVE CONNECTION DIAGRAM



EQUATION 14-1: RELATIONSHIP BETWEEN DEVICE AND SPI CLOCK SPEED

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

TABLE 14-1: SAMPLE SCKx FREQUENCIES

FCY = 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
FCY = 5 MHz						
Primary Prescaler Settings	1:1	5000	2500	1250	833	625
	4:1	1250	625	313	208	156
	16:1	313	156	78	52	39
	64:1	78	39	20	13	10

Note: SCKx frequencies shown in kHz.

REGISTER 14-1: 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	—	SPISIDL	—	—	—	—	—
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

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

bit 15	**SPIEN:** SPIx Enable bit `1` = Enables module and configures SCKx, SDOx, SDIx and $\overline{\text{SSx}}$ as serial port pins `0` = Disables module
bit 14	**Unimplemented:** Read as ‘`0`’
bit 13	**SPISIDL:** Stop in Idle Mode bit `1` = Discontinue module operation when device enters Idle mode `0` = Continue module operation in Idle mode
bit 12-7	**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

REGISTER 14-2: 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(1)
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(1) 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 = $\overline{\text{SSx}}$ pin used for Slave mode 0 = $\overline{\text{SSx}}$ 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

Note 1: The CKE bit is not used in the Framed SPI modes. Program this bit to ‘0’ for the Framed SPI modes (FRMEN = 1).

REGISTER 14-2: SPIxCON1: SPIx CONTROL REGISTER 1 (CONTINUED)

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. Program this bit to ‘`0`’ for the Framed SPI modes (FRMEN = `1`).

REGISTER 14-3: 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 ($\overline{\text{SSx}}$ 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:** This bit must not be set to '`1`' by the user application.

15.0 INTER-INTEGRATED CIRCUIT (I^2C)

> **Note:** This data sheet summarizes the features of the PIC24HJ32GP202/204 and PIC24HJ16GP304 devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the *"PIC24H Family Reference Manual"*.

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

The I^2C module has a 2-pin interface:

- The SCLx pin is clock
- The SDAx pin is data.

The I^2C module offers the following key features:

- I^2C interface supporting both Master and Slave modes of operation.
- I^2C Slave mode supports 7- and 10-bit address.
- I^2C Master mode supports 7- and 10-bit address.
- I^2C port allows bidirectional transfers between master and slaves.
- Serial clock synchronization for I^2C port can be used as a handshake mechanism to suspend and resume serial transfer (SCLREL control).
- I^2C supports multi-master operation, detects bus collision and arbitrates accordingly.

15.1 Operating Modes

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

The I^2C module can operate either as a slave or a master on an I^2C bus.

The following types of I^2C operation are supported:

- I^2C slave operation with 7-bit address
- I^2C slave operation with 10-bit address
- I^2C master operation with 7- or 10-bit address

For details about the communication sequence in each of these modes, refer to the *"PIC24H Family Reference Manual"*.

15.2 I^2C 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 I2CSTAT are read/write.

- I2CxRSR is the shift register used for shifting data.
- I2CxRCV is the receive buffer and the register to which data bytes are written, or from which data bytes are read.
- I2CxTRN 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.
- 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.

15.3 I^2C Interrupts

The I^2C module generates two interrupt flags:

- MI2CxIF (I^2C Master Events Interrupt flag)
- SI2CxIF (I^2C Slave Events Interrupt flag).

A separate interrupt is generated for all I^2C error conditions.

15.4 Baud Rate Generator

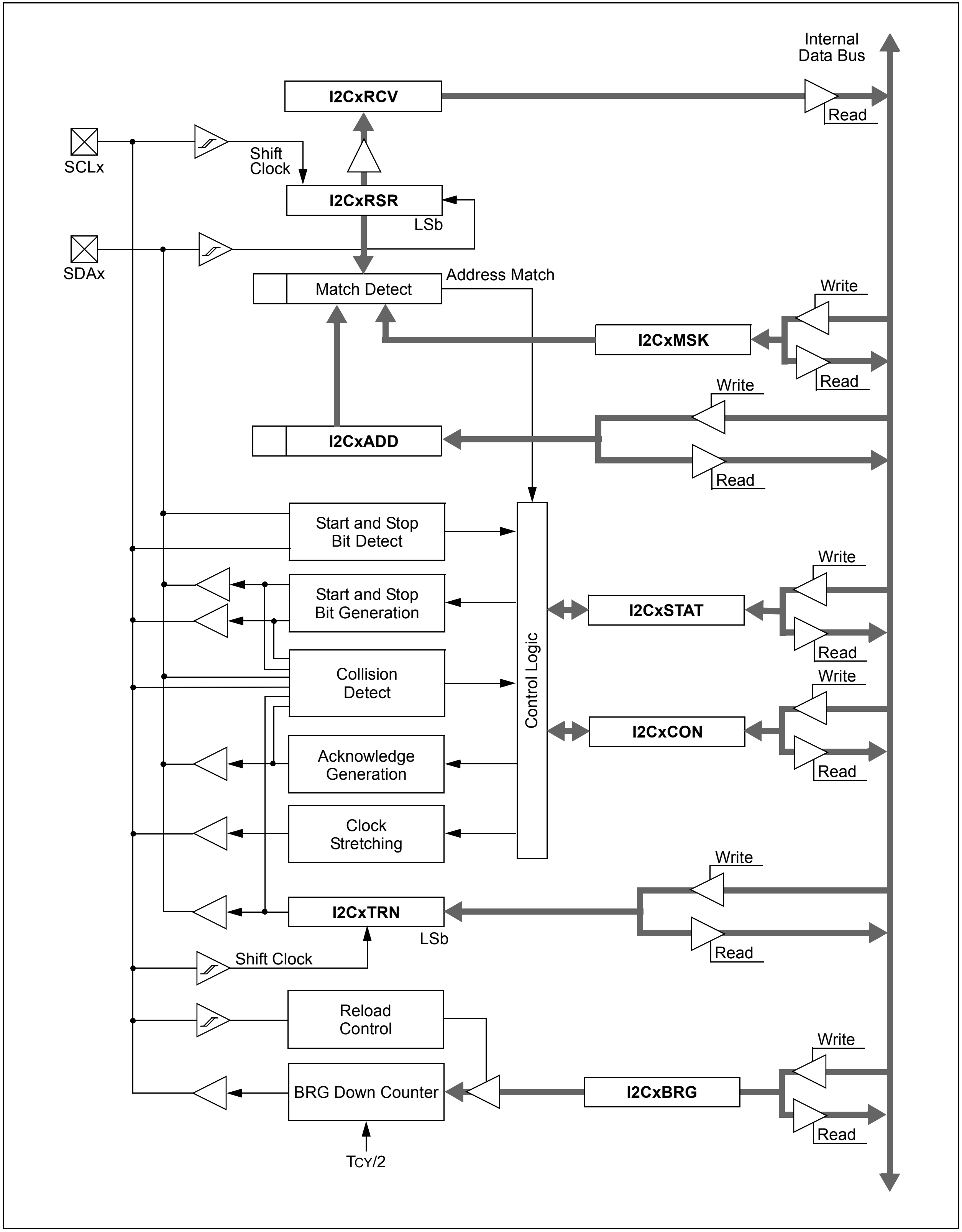
In I^2C Master mode, the reload value for the Baud Rate Generator (BRG) is located in the I2CxBRG register. When the BRG is loaded with this value, the BRG counts down to zero and stops until another reload has taken place. If clock arbitration is taking place, for example, the BRG is reloaded when the SCLx pin is sampled high.

As per the I^2C standard, F_{SCL} can be 100 kHz or 400 kHz. However, the user application can specify any baud rate up to 1 MHz. I2CxBRG values of '`0`' or '`1`' are illegal.

EQUATION 15-1: SERIAL CLOCK RATE

$$I2CxBRG = \left(\frac{F_{CY}}{F_{SCL}} - \frac{F_{CY}}{10{,}000{,}000}\right) - 1$$

FIGURE 15-1: I^{2}C™ BLOCK DIAGRAM (x = 1)



15.5 I^2C Module Addresses

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

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 seven 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 is 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 15-1: 7-BIT I^2C™ SLAVE ADDRESSES SUPPORTED BY PIC24HJ32GP202/204 AND PIC24HJ16GP304

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

15.6 Slave Address Masking

The I2CxMSK register (Register 15-3) designates address bit positions as “don’t care” for both 7-bit and 10-bit Address modes. Setting a particular bit location (= `1`) in the I2CxMSK register causes the slave module to respond, whether the corresponding address bit value is a ‘`0`’ or ‘`1`’. For example, when I2CxMSK is set to ‘`00100000`’, the Slave module will detect both addresses, ‘`0000000`’ and ‘`00100000`’.

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

15.7 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.

15.8 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^2C 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.

15.9 Automatic Clock Stretch

In Slave modes, the module can synchronize buffer reads and write to the master device by clock stretching.

15.9.1 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 application has time to service the ISR and load the contents of the I2CxTRN before the master device can initiate another transmit sequence.

15.9.2 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 application has time to service the ISR and read the contents of the I2CxRCV before the master device can initiate another receive sequence. This prevents buffer overruns.

15.10 Software Controlled Clock Stretching (STREN = `1`)

When the STREN bit is ‘`1`’, the software can clear the SCLREL bit to allow software to control the clock stretching.

If the STREN bit is ‘`0`’, a software write to the SCLREL bit is disregarded and has no effect on the SCLREL bit.

15.11 Slope Control

The I^2C standard requires slope control on the SDAx and SCLx signals for Fast mode (400 kHz). The control bit, DISSLW, enables the user application to disable slew rate control if desired. It is necessary to disable the slew rate control for 1 MHz mode.

15.12 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 BRG is reloaded with the contents of I2CxBRG and begins counting. This process 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.

15.13 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^2C master events interrupt flag and reset the master portion of the I^2C port to its Idle state.

15.14 Peripheral Pin Select Limitations

The I^2C module has limited peripheral pin select functionality. When the ACTI2C bit in the FPOR configuration register is set to '`1`', the module uses the SDAx/SCLx pins. If the ALTI2C bit is '`0`', the module uses the ASDAx/ASCLx pins.

REGISTER 15-1: 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

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

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^{2}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^{2}C slave)

1 = Release SCLx clock
0 = Hold SCLx clock low (clock stretch)

If STREN = 1:
Bit is R/W (i.e., software can 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 can 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^{2}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^{2}C slave)

Used in conjunction with SCLREL bit.
1 = Enable software or receive clock stretching
0 = Disable software or receive clock stretching

REGISTER 15-1: I2CxCON: I2Cx CONTROL REGISTER (CONTINUED)

bit 5	**ACKDT:** Acknowledge Data bit (when operating as I^{2}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^{2}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^{2}C master) `1` = Enables Receive mode for I^{2}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^{2}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^{2}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^{2}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 15-2: 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	I2COV	D_A	P	S	R_W	RBF	TBF
bit 7							bit 0

Legend:	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^{2}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^{2}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^{2}C module is busy
0 = No collision
Hardware set at occurrence of write to I2CxTRN while busy (cleared by software).

bit 6 **I2COV:** Receive Overflow Flag bit
1 = A byte was received while the I2CxRCV register is still holding the previous byte
0 = No overflow
Hardware set at attempt to transfer I2CxRSR to I2CxRCV (cleared by software).

bit 5 **D_A:** Data/Address bit (when operating as I^{2}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.

REGISTER 15-2: 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^{2}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^{2}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 15-3: 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

NOTES:

16.0 UNIVERSAL ASYNCHRONOUS RECEIVER TRANSMITTER (UART)

Note:	This data sheet summarizes the features of the PIC24HJ32GP202/204 and PIC24HJ16GP304 devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the *"PIC24H Family Reference Manual"*.

The Universal Asynchronous Receiver Transmitter (UART) module is one of the serial I/O modules available in the PIC24HJ32GP202/204 and PIC24HJ16GP304 device family. The UART is a full-duplex asynchronous system that can communicate with peripheral devices, such as personal computers, LIN, RS-232 and RS-485 interfaces. The module also supports a hardware flow control option with the $\overline{\text{UxCTS}}$ and $\overline{\text{UxRTS}}$ pins and also includes an IrDA® encoder and decoder.

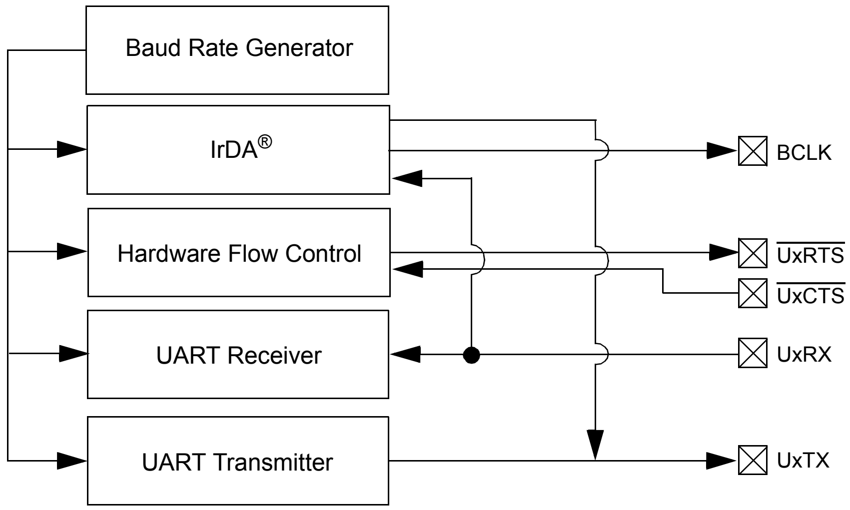
The primary features of the UART module are:

- Full-Duplex 8- or 9-bit Data Transmission through the UxTX and UxRX pins
- Even, odd or no parity options (for 8-bit data)
- One or two stop bits
- Hardware Flow Control Option with $\overline{\text{UxCTS}}$ and $\overline{\text{UxRTS}}$ pins
- Fully Integrated Baud Rate Generator with 16-bit prescaler
- Baud rates ranging from 1 Mbps to 15 Mbps 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
- Support for automatic baud rate detection
- IrDA encoder and decoder logic
- 16x baud clock output for IrDA support

A simplified block diagram of the UART module is shown in Figure 16-1. The UART module consists of the following key hardware elements:

- Baud Rate Generator
- Asynchronous Transmitter
- Asynchronous Receiver

FIGURE 16-1: UART SIMPLIFIED BLOCK DIAGRAM



16.1 UART Baud Rate Generator

The UART module includes a dedicated 16-bit Baud Rate Generator (BRG). The BRGx register controls the period of a free-running 16-bit timer. Equation 16-1 shows the formula for computation of the baud rate with BRGH = `0`.

EQUATION 16-1: UART BAUD RATE WITH BRGH = 0

$$\text{Baud Rate} = \frac{F_{CY}}{16 \bullet (\text{BRGx} + 1)}$$

$$\text{BRGx} = \frac{F_{CY}}{16 \bullet \text{Baud Rate}} - 1$$

Note: FCY denotes the instruction cycle clock frequency (FOSC/2).

Example 16-1 shows the calculation of the baud rate error for the following conditions:

- FCY = 4 MHz
- Desired Baud Rate = 9600

The maximum baud rate (BRGH = `0`) possible is FCY/16 (for BRGx = `0`), and the minimum baud rate possible is FCY/(16 * 65536).

Equation 16-2 shows the formula for computation of the baud rate with BRGH = `1`.

EQUATION 16-2: UART BAUD RATE WITH BRGH = 1

$$\text{Baud Rate} = \frac{F_{CY}}{4 \bullet (\text{BRGx} + 1)}$$

$$\text{BRGx} = \frac{F_{CY}}{4 \bullet \text{Baud Rate}} - 1$$

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.

EXAMPLE 16-1: 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%

16.2 Transmitting in 8-bit Data Mode

1. Set up the UART:
   a) Write appropriate values for data, parity and Stop bits.
   b) Write appropriate baud rate value to the BRGx register.
   c) Set up transmit and receive interrupt enable and priority bits.
2. Enable the UART.
3. Set the UTXEN bit (causes a transmit interrupt).

   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.
   Alternately, the data byte can be transferred while UTXEN = `0`, and the user application can set UTXEN. This causes the serial bit stream to begin immediately, because the baud clock starts from a cleared state.
4. A transmit interrupt will be generated as per interrupt control bits, UTXISEL<1:0>.

16.3 Transmitting in 9-bit Data Mode

1. Set up the UART (as described in **Section 16.2 "Transmitting in 8-bit Data Mode"**).
2. Enable the UART.
3. Set the UTXEN bit (causes a transmit interrupt).
4. Write UxTXREG as a 16-bit value only.

   A word write to UxTXREG triggers the transfer of the 9-bit data to the TSR. The serial bit stream will start shifting out with the first rising edge of the baud clock.
   A transmit interrupt will be generated as per the setting of control bits, UTXISEL<1:0>.

16.4 Break and Sync Transmit Sequence

The following sequence will send a message frame header made up of a Break, followed by an auto-baud Sync byte.

1. Configure the UART for the desired mode.
2. Set UTXEN and UTXBRK, which 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, which loads the Sync character into the transmit FIFO. After the Break has been sent, the UTXBRK bit is reset by hardware. The Sync character now starts transmitting.

16.5 Receiving in 8-bit or 9-bit Data Mode

1. Set up the UART (as described in **Section 16.2 "Transmitting in 8-bit Data Mode"**).
2. Enable the UART. A receive interrupt will be generated when one or more data characters have been received as per interrupt control bits, URXISEL<1:0>.
3. Read the OERR bit to determine if an overrun error has occurred. The OERR bit must be reset in software.
4. 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.

16.6 Flow Control Using $\overline{\text{UxCTS}}$ and UxRTS Pins

UARTx Clear to Send ($\overline{\text{UxCTS}}$) and Request to Send (UxRTS) are the two hardware controlled active-low pins associated with the UART module. The UEN<1:0> bits in the UxMODE register configure these pins.

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).

16.7 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.

16.7.1 EXTERNAL IrDA SUPPORT – IrDA CLOCK OUTPUT

To support external IrDA encoder and decoder devices, the BCLK 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. The pin can be used to support the IrDA codec chip.

16.7.2 BUILT-IN IrDA ENCODER AND DECODER

The UART module includes full implementation of the IrDA encoder and decoder. The built-in IrDA encoder and decoder functionality is enabled using the IREN bit (UxMODE<12>). When IREN = `1` is enabled, the receive pin (UxRX) acts as the input from the infrared receiver. The transmit pin (UxTX) acts as the output to the infrared transmitter.

REGISTER 16-1: UxMODE: UARTx MODE REGISTER

R/W-0	U-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
UARTEN	—	USIDL	IREN(1)	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

Legend:	HC = Hardware cleared		
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 **UARTEN:** UARTx Enable bit

1 = UARTx is enabled; all UARTx pins are controlled by UARTx as defined by UEN<1:0>
0 = UARTx is disabled; all UARTx pins are controlled by port latches; UARTx power consumption minimal

bit 14 **Unimplemented:** Read as ‘0’

bit 13 **USIDL:** Stop in Idle Mode bit

1 = Discontinue module operation when device enters Idle mode
0 = Continue module operation in Idle mode

bit 12 **IREN:** IrDA Encoder and Decoder Enable bit(1)

1 = IrDA encoder and decoder enabled
0 = IrDA encoder and decoder disabled

bit 11 **RTSMD:** Mode Selection for $\overline{\text{UxRTS}}$ Pin bit

1 = $\overline{\text{UxRTS}}$ pin in Simplex mode
0 = $\overline{\text{UxRTS}}$ pin in Flow Control mode

bit 10 **Unimplemented:** Read as ‘0’

bit 9-8 **UEN<1:0>:** UARTx Enable bits

11 = UxTX, UxRX and BCLK pins are enabled and used; $\overline{\text{UxCTS}}$ pin controlled by port latches
10 = UxTX, UxRX, $\overline{\text{UxCTS}}$ and $\overline{\text{UxRTS}}$ pins are enabled and used
01 = UxTX, UxRX and $\overline{\text{UxRTS}}$ pins are enabled and used; $\overline{\text{UxCTS}}$ pin controlled by port latches
00 = UxTX and UxRX pins are enabled and used; $\overline{\text{UxCTS}}$ and $\overline{\text{UxRTS}}$/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) before other data; 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).

REGISTER 16-1: 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`).

REGISTER 16-2: 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[1]	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

Legend:	HC = Hardware cleared		
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 **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[1]

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 16-2: 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`).

NOTES:

17.0 10-BIT/12-BIT ANALOG-TO-DIGITAL CONVERTER (ADC)

> **Note:** This data sheet summarizes the features of the PIC24HJ32GP202/204 and PIC24HJ16GP304 devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the *"PIC24H Family Reference Manual"*.

The PIC24HJ32GP202/204 and PIC24HJ16GP304 devices have up to 13 Analog-to-Digital Conversion (ADC) module input channels.

The AD12B bit (AD1CON1<10>) allows each of the ADC modules to be configured as either a 10-bit, 4-sample-and-hold ADC (default configuration), or a 12-bit, 1-sample-and-hold ADC.

> **Note:** The ADC module must be disabled before the AD12B bit can be modified.

17.1 Key Features

The 10-bit ADC configuration has the following key features:

- Successive Approximation (SAR) conversion
- Conversion speeds of up to 1.1 Msps
- Up to 13 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
- Operation during CPU Sleep and Idle modes
- 16-word conversion result buffer

The 12-bit ADC 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-and-hold amplifier in the 12-bit configuration, so simultaneous sampling of multiple channels is not supported.

Depending on the particular device pinout, the ADC can have up to 13 analog input pins, designated AN0 through AN12. In addition, there are two analog input pins for external voltage reference connections. These voltage reference inputs can 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.

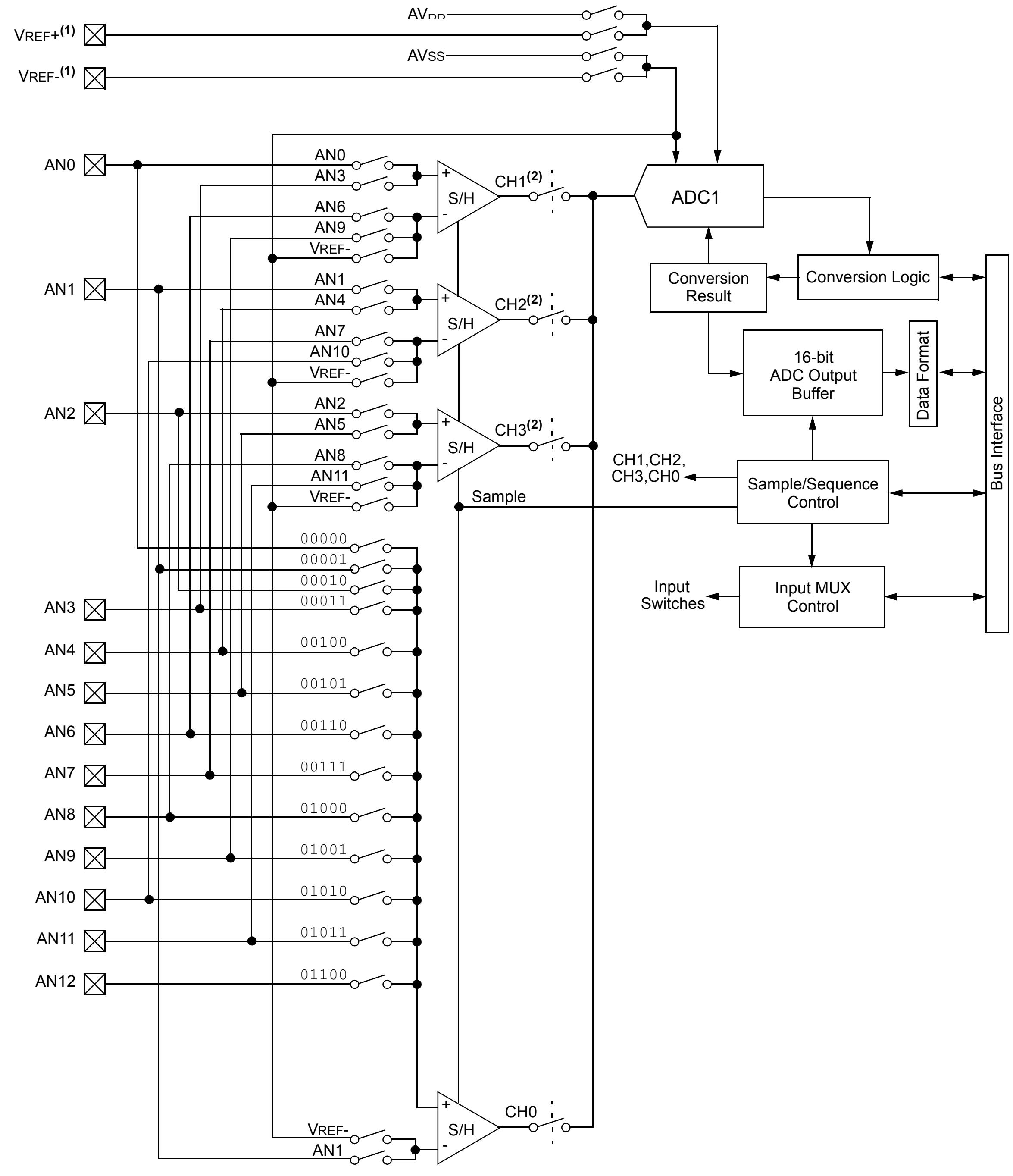
A block diagram of ADC for PIC24HJ16GP304 and PIC24HJ32GP204 devices is shown in Figure 17-1. A block diagram of the ADC for the PIC24HJ32GP202 device is shown in Figure 17-2.

17.2 ADC Initialization

To configure the ADC module:

1. Select port pins as analog inputs (AD1PCFGH<15:0> or AD1PCFGL<15:0>).
2. Select voltage reference source to match expected range on analog inputs (AD1CON2<15:13>).
3. Select the analog conversion clock to match desired data rate with processor clock (AD1CON3<7:0>).
4. Determine how many sample-and-hold channels will be used (AD1CON2<9:8> and AD1PCFGH<15:0> or AD1PCFGL<15:0>).
5. Select the appropriate sample/conversion sequence (AD1CON1<7:5> and AD1CON3<12:8>).
6. Select the way conversion results are presented in the buffer (AD1CON1<9:8>).
7. Turn on the ADC module (AD1CON1<15>).
8. Configure ADC interrupt (if required):
    a) Clear the AD1IF bit.
    b) Select ADC interrupt priority.

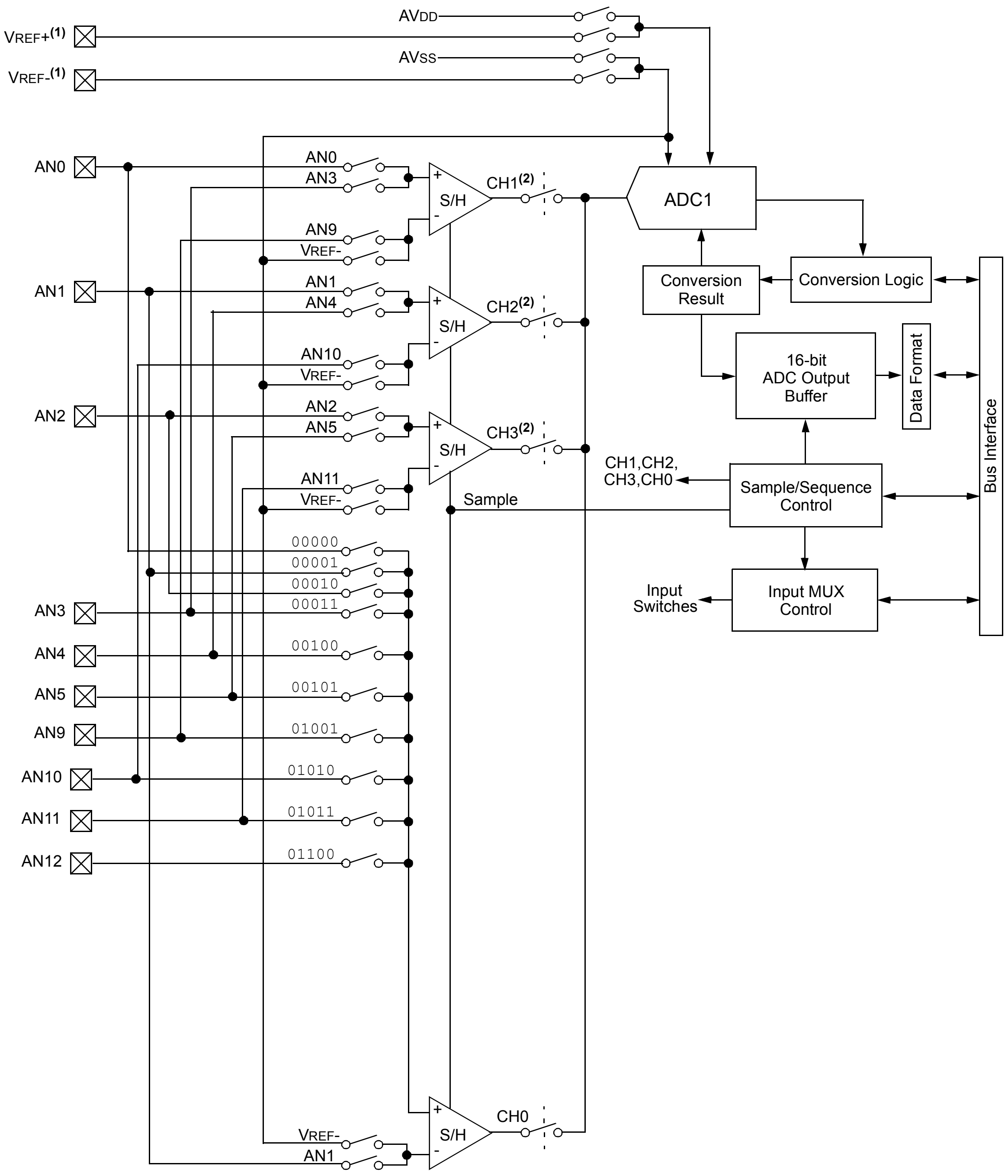
FIGURE 17-1: ADC1 MODULE BLOCK DIAGRAM FOR PIC24HFJ16GP304 AND PIC24HJ32GP204 DEVICES



Note 1: VREF+, VREF- inputs may be multiplexed with other analog inputs.

2: Channels 1, 2 and 3 are not applicable for the 12-bit mode of operation.

FIGURE 17-2: ADC1 MODULE BLOCK DIAGRAM FOR PIC24HJ32GP202 DEVICES



Note 1: VREF+, VREF- inputs may be multiplexed with other analog inputs.

2: Channels 1, 2 and 3 are not applicable for the 12-bit mode of operation.

EQUATION 17-1: ADC CONVERSION CLOCK PERIOD

$$T_{AD} = T_{CY}(ADCS + 1)$$

$$ADCS = \frac{T_{AD}}{T_{CY}} - 1$$

FIGURE 17-3: ADC TRANSFER FUNCTION (10-BIT EXAMPLE)

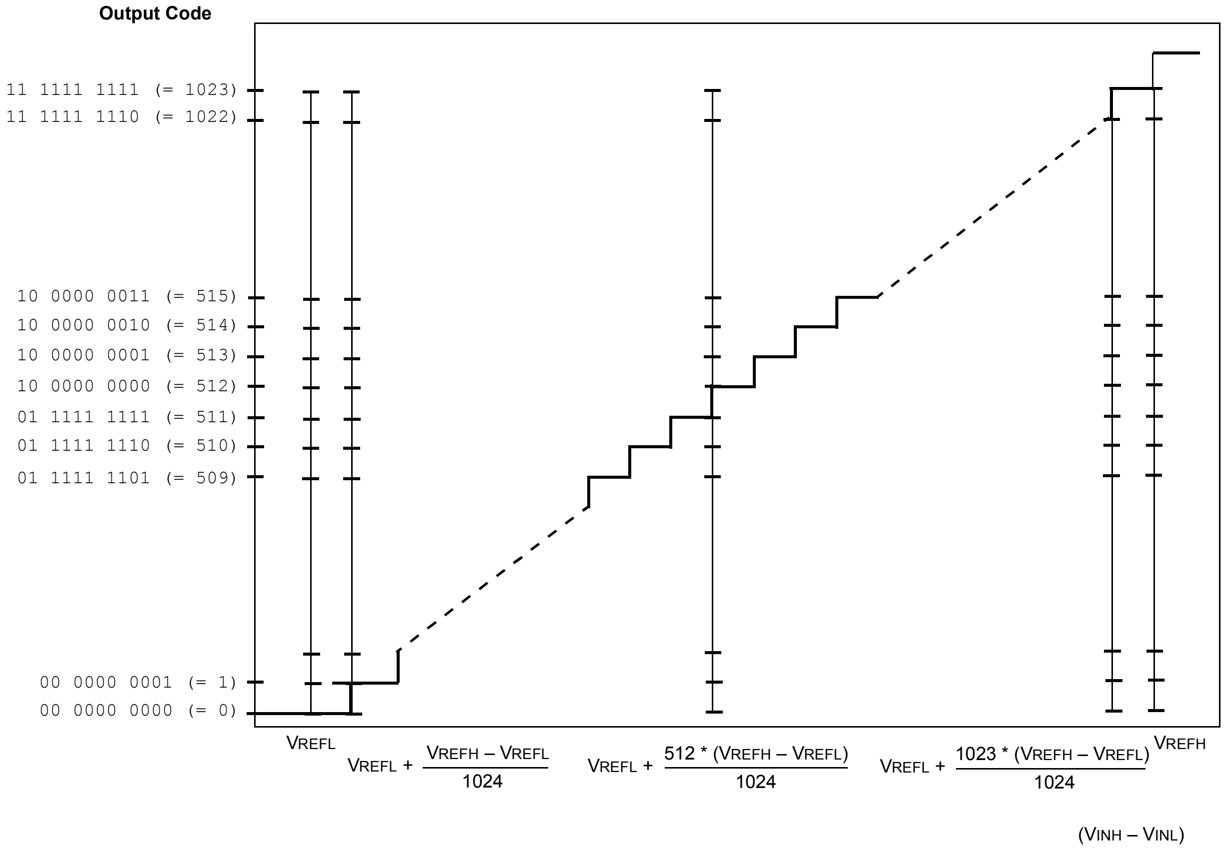
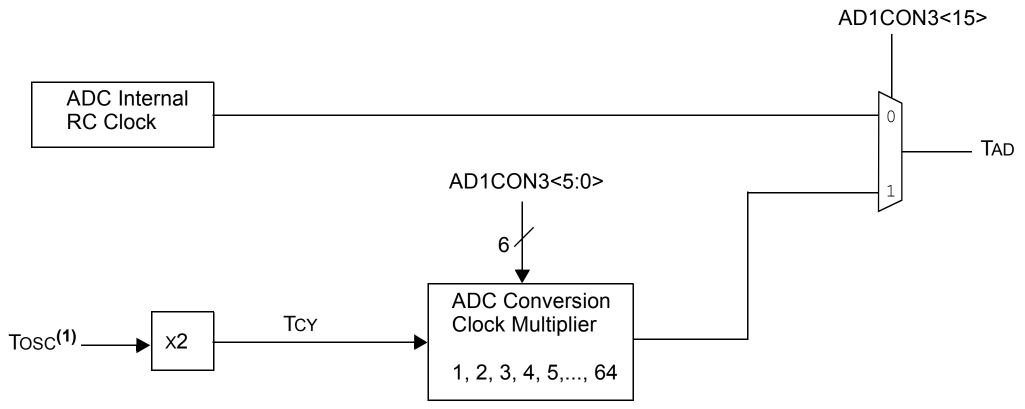


FIGURE 17-4: ADC CONVERSION CLOCK PERIOD BLOCK DIAGRAM



Note: Refer to Figure 7-2 for the derivation of FOSC when the PLL is enabled. If the PLL is not used, FOSC is equal to the clock frequency. TOSC = 1/FOSC.

REGISTER 17-1: AD1CON1: ADC1 CONTROL REGISTER 1

R/W-0	U-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0
ADON	—	ADSIDL	—	—	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 HC,HS	R/C-0 HC, HS
SSRC<2:0>			—	SIMSAM	ASAM	SAMP	DONE
bit 7							bit 0

Legend:	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:** ADC Operating Mode bit

1 = ADC module is operating
0 = ADC 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 12-11 **Unimplemented:** Read as ‘0’

bit 10 **AD12B:** 10-bit or 12-bit Operation Mode bit

1 = 12-bit, 1-channel ADC operation
0 = 10-bit, 4-channel ADC operation

bit 9-8 **FORM<1:0>:** Data Output Format bits

<u>For 10-bit operation:</u>
11 = Reserved
10 = Reserved
01 = Signed integer (DOUT = ssss sssd dddd dddd, where s = .NOT.d<9>)
00 = Integer (DOUT = 0000 00dd dddd dddd)

<u>For 12-bit operation:</u>
11 = Reserved
10 = Reserved
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 3 compare ends sampling and starts conversion
001 = Active transition on INT0 pin ends sampling and starts conversion
000 = Clearing sample bit ends sampling and starts conversion

bit 4 **Unimplemented:** Read as ‘0’

bit 3 **SIMSAM:** Simultaneous Sample Select bit (applicable only 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

REGISTER 17-1: AD1CON1: ADC1 CONTROL REGISTER 1 (CONTINUED)

bit 2	**ASAM:** ADC 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:** ADC Sample Enable bit `1` = ADC sample-and-hold amplifiers are sampling `0` = ADC sample-and-hold amplifiers are holding If ASAM = `0`, software can write '`1`' to begin sampling. Automatically set by hardware if ASAM = `1`. If SSRC = `000`, software can write '`0`' to end sampling and start conversion. If SSRC ≠ `000`, automatically cleared by hardware to end sampling and start conversion.
bit 0	**DONE:** ADC Conversion Status bit `1` = ADC conversion cycle is completed `0` = ADC conversion not started or in progress Automatically set by hardware when ADC conversion is complete. Software can 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 17-2: AD1CON2: ADC1 CONTROL REGISTER 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>:** Select 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 (valid only when BUFM = `1`)

`1` = ADC is currently filling second half of buffer, user application should access data in the first half
`0` = ADC is currently filling first half of buffer, user application should access data in the second half

bit 6 **Unimplemented:** Read as ‘`0`’

bit 5-2 **SMPI<3:0>:** Sample/Convert Sequences Per Interrupt Selection bits

`1111` =Interrupts at the completion of conversion for each 16th sample/convert sequence
`1110` =Interrupts at the completion of conversion for each 15th sample/convert sequence
•
•
•
`0001` =Interrupts at the completion of conversion for each 2nd sample/convert sequence
`0000` =Interrupts at the completion of conversion for each sample/convert sequence

bit 1 **BUFM:** Buffer Fill Mode Select bit

`1` = Starts filling first half of buffer on first interrupt and the second half of buffer on next interrupt
`0` = Always starts filling buffer from the beginning

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

REGISTER 17-3: AD1CON3: ADC1 CONTROL REGISTER 3

R/W-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	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADCS<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 **ADRC:** ADC Conversion Clock Source bit

1 = ADC 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 7-0 **ADCS<7:0>:** ADC Conversion Clock Select bits

11111111 = TCY · (ADCS<7:0> + 1) = 256 · TCY = TAD
•
•
•
00000010 = TCY · (ADCS<7:0> + 1) = 3 · TCY = TAD
00000001 = TCY · (ADCS<7:0> + 1) = 2 · TCY = TAD
00000000 = TCY · (ADCS<7:0> + 1) = 1 · TCY = TAD

REGISTER 17-4: AD1CHS123: ADC1 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

PIC24HJ32GP202 devices only:

If AD12B = `1`:
`11` = Reserved
`10` = Reserved
`01` = Reserved
`00` = Reserved

If AD12B = `0`:
`11` = CH1 negative input is AN9, CH2 negative input is AN10, CH3 negative input is AN11
`10` = Reserved
`01` = CH1, CH2, CH3 negative input is VREF-
`00` = CH1, CH2, CH3 negative input is VREF-

PIC24HJ32GP204 and PIC24HJ16GP304 devices only:

If AD12B = `1`:
`11` = Reserved
`10` = Reserved
`01` = Reserved
`00` = Reserved

If AD12B = `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
`01` = CH1, CH2, CH3 negative input is VREF-
`00` = CH1, CH2, CH3 negative input is VREF-

bit 8 **CH123SB:** Channel 1, 2, 3 Positive Input Select for Sample B bit

If AD12B = `1`:
`1` = Reserved
`0` = Reserved

If AD12B = `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`’

REGISTER 17-4: AD1CHS123: ADC1 INPUT CHANNEL 1, 2, 3 SELECT REGISTER (CONTINUED)

bit 2-1 **CH123NA<1:0>:** Channel 1, 2, 3 Negative Input Select for Sample A bits

PIC24HJ32GP202 devices only:

If AD12B = `1`:
`11` = Reserved
`10` = Reserved
`01` = Reserved
`00` = Reserved

If AD12B = `0`:
`11` = CH1 negative input is AN9, CH2 negative input is AN10, CH3 negative input is AN11
`10` = Reserved
`01` = CH1, CH2, CH3 negative input is VREF-
`00` = CH1, CH2, CH3 negative input is VREF-

PIC24HJ32GP204 and PIC24HJ16GP304 devices only:

If AD12B = `1`:
`11` = Reserved
`10` = Reserved
`01` = Reserved
`00` = Reserved

If AD12B = `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
`01` = CH1, CH2, CH3 negative input is VREF-
`00` = CH1, CH2, CH3 negative input is VREF-

bit 0 **CH123SA:** Channel 1, 2, 3 Positive Input Select for Sample A bit

If AD12B = `1`:
`1` = Reserved
`0` = Reserved

If AD12B = `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 17-5: AD1CHS0: ADC1 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
1 = Channel 0 negative input is AN1
0 = Channel 0 negative input is VREF-

bit 14-13 **Unimplemented:** Read as ‘0’

bit 12-8 **CH0SB<4:0>:** Channel 0 Positive Input Select for Sample B bits

PIC24HJ32GP204 and PIC24HJ16GP304 devices only:
01100 = Channel 0 positive input is AN12
•
•
•
00010 = Channel 0 positive input is AN2
00001 = Channel 0 positive input is AN1
00000 = Channel 0 positive input is AN0

PIC24HJ32GP202 devices only:
01100 = Channel 0 positive input is AN12
•
•
•
01000 = Reserved
00111 = Reserved
00110 = Reserved
•
•
•
00010 = Channel 0 positive input is AN2
00001 = Channel 0 positive input is AN1
00000 = Channel 0 positive input is AN0

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’

REGISTER 17-5: AD1CHS0: ADC1 INPUT CHANNEL 0 SELECT REGISTER (CONTINUED)

bit 4-0 **CH0SA<4:0>:** Channel 0 Positive Input Select for Sample A bits

PIC24HJ32GP204 and PIC24HJ16GP304 devices only:

`01100` = Channel 0 positive input is AN12
•
•
•
`00010` = Channel 0 positive input is AN2
`00001` = Channel 0 positive input is AN1
`00000` = Channel 0 positive input is AN0

PIC24HJ32GP202 devices only:

`01100` = Channel 0 positive input is AN12
•
•
•
`01000` = Reserved
`00111` = Reserved
`00110` = Reserved
•
•
•
`00010` = Channel 0 positive input is AN2
`00001` = Channel 0 positive input is AN1
`00000` = Channel 0 positive input is AN0

REGISTER 17-6: AD1CSSL: ADC1 INPUT SCAN SELECT REGISTER LOW[1]

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	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<12:0>:** ADC Input Scan Selection bits

`1` = Select ANx for input scan

`0` = Skip ANx for input scan

Note 1: On devices without nine analog inputs, all AD1CSSL bits can be selected. However, inputs selected for scan without a corresponding input on device will convert ADREF-.

REGISTER 17-7: AD1PCFGL: ADC1 PORT CONFIGURATION REGISTER LOW[1]

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	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<12:0>:** ADC Port Configuration Control bits

`1` = Port pin in Digital mode, port read input enabled, ADC input multiplexer connected to AVSS

`0` = Port pin in Analog mode, port read input disabled, ADC samples pin voltage

Note 1: On devices without nine analog inputs, all PCFG bits are R/W. However, PCFG bits are ignored on ports without a corresponding input on device.

NOTES:

18.0 SPECIAL FEATURES

> **Note:** This data sheet summarizes the features of the PIC24HJ32GP202/204 and PIC24HJ16GP304 devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the *"PIC24H Family Reference Manual"*.

PIC24HJ32GP202/204 and PIC24HJ16GP304 devices include several features that are intended to maximize application flexibility and reliability, and minimize cost through elimination of external components. These are:

- Flexible configuration
- Watchdog Timer (WDT)
- Code Protection and CodeGuard™ Security
- JTAG Boundary Scan Interface
- In-Circuit Serial Programming™ (ICSP™)
- In-Circuit emulation

18.1 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 18-1.

The individual Configuration bit descriptions for the FBS, FGS, FOSCSEL, FOSC, FWDT, FPOR and FICD Configuration registers are shown in Table 18-2.

Note that address 0xF80000 is beyond the user program memory space. It belongs to the configuration memory space (0x800000-0xFFFFFF), 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`' to these locations has no effect on device operation.

To prevent the 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 18-1: DEVICE CONFIGURATION REGISTER MAP

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0xF80000	FBS	—	—	—	—	BSS<2:0>			BWRP
0xF80002	Reserved	Reserved(1)							
0xF80004	FGS	—	—	—	—	—	GSS<1:0>		GWRP
0xF80006	FOSCSEL	IESO	—	—	—		FNOSC<2:0>		
0xF80008	FOSC	FCKSM<1:0>		IOL1WAY	—	—	OSCIOFNC	POSCMD<1:0>	
0xF8000A	FWDT	FWDTEN	WINDIS	—	WDTPRE	WDTPOST<3:0>			
0xF8000C	FPOR	—	—	—	ALTI2C	—	FPWRT<2:0>		
0xF8000E	Reserved	Reserved(1)							
0xF80010	FUID0	User Unit ID Byte 0							
0xF80012	FUID1	User Unit ID Byte 1							
0xF80014	FUID2	User Unit ID Byte 2							
0xF80016	FUID3	User Unit ID Byte 3							

Note 1: These reserved bits read as '1' and must be programmed as '1'.

TABLE 18-2: PIC24HJ32GP202/204 AND PIC24HJ16GP304 CONFIGURATION BITS DESCRIPTION

Bit Field	Register	Description
BWRP	FBS	Boot Segment Program Flash Write Protection `1` = Boot segment may be written `0` = Boot segment is write-protected
BSS<2:0>	FBS	**PIC24HJ32GP202 and PIC24HJ32GP204 Devices Only** Boot Segment Program Flash Code Protection Size `x11` = No Boot program Flash segment Boot space is 768 Instruction Words (except interrupt vectors) `110` = Standard security; boot program Flash segment ends at 0x0007FE `010` = High security; boot program Flash segment ends at 0x0007FE Boot space is 3840 Instruction Words (except interrupt vectors) `101` = Standard security; boot program Flash segment, ends at 0x001FFE `001` = High security; boot program Flash segment ends at 0x001FFE Boot space is 7936 Instruction Words (except interrupt vectors) `100` = Standard security; boot program Flash segment ends at 0x003FFE `000` = High security; boot program Flash segment ends at 0x003FFE
BSS<2:0>	FBS	**PIC24HJ16GP304 Devices Only** Boot Segment Program Flash Code Protection Size `x11` = No Boot program Flash segment Boot space is 768 Instruction Words (except interrupt vectors) `110` = Standard security; boot program Flash segment ends at 0x0007FE `010` = High security; boot program Flash segment ends at 0x0007FE Boot space is 3840 Instruction Words (except interrupt vectors) `101` = Standard security; boot program Flash segment, ends at 0x001FFE `001` = High security; boot program Flash segment ends at 0x001FFE Boot space is 5376 Instruction Words (except interrupt vectors) `100` = Standard security; boot program Flash segment ends at 0x002BFE `000` = High security; boot program Flash segment ends at 0x002BFE
GSS<1:0>	FGS	General Segment Code-Protect bit `11` = User program memory is not code-protected `10` = Standard security `0x` = High security
GWRP	FGS	General Segment Write-Protect bit `1` = User program memory is not write-protected `0` = User program memory is write-protected
IESO	FOSCSEL	Two-speed Oscillator Start-up Enable 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
FNOSC<2:0>	FOSCSEL	Initial Oscillator Source Selection bits `111` = Internal Fast RC (FRC) oscillator with postscaler `110` = Internal Fast RC (FRC) oscillator with divide-by-16 `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

TABLE 18-2: PIC24HJ32GP202/204 AND PIC24HJ16GP304 CONFIGURATION BITS DESCRIPTION (CONTINUED)

Bit Field	Register	Description
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
IOL1WAY	FOSC	Peripheral Pin Select Configuration `1` = Allow only one re-configuration `0` = Allow multiple re-configurations
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<3:0>	FWDT	Watchdog Timer Postscaler bits `1111` = 1:32,768 `1110` = 1:16,384 . . . `0001` = 1:2 `0000` = 1:1
ALTI2C	FPOR	Alternate I^2C pins 1 = I^2C mapped to SDA1/SCL1 pins 0 = I^2C mapped to ASDA1/ASCL1 pins
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

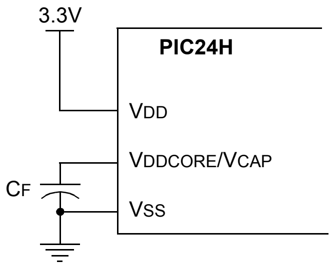
18.2 On-Chip Voltage Regulator

All of the PIC24HJ32GP202/204 and PIC24HJ16GP304 devices power their core digital logic at a nominal 2.5V. This can create a conflict for designs that are required to operate at a higher typical voltage, such as 3.3V. To simplify system design, all devices in the PIC24HJ32GP202/204 and PIC24HJ16GP304 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 18-1). This helps to maintain the stability of the regulator. The recommended value for the filter capacitor is provided in **Section TABLE 21-13: "Internal Voltage Regulator Specifications"** located in **Section 21.1 "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 TSTARTUP, code execution is disabled. TSTARTUP is applied every time the device resumes operation after any power-down.

FIGURE 18-1: CONNECTIONS FOR THE ON-CHIP VOLTAGE REGULATOR(1)



Note 1: These are typical operating voltages. Refer to **Section TABLE 21-13: "Internal Voltage Regulator Specifications"** located in **Section 21.1 "DC Characteristics"** for the full operating ranges of VDD and VDDCORE.

18.3 BOR: Brown-Out Reset

The Brown-out Reset (BOR) module is based on an internal voltage reference circuit that monitors the regulated voltage VDDCORE. The main purpose of the BOR module is to generate a device Reset when a brown-out condition occurs. Brown-out conditions are generally caused by glitches on the AC mains (for example, missing portions of the AC cycle waveform due to bad power transmission lines, or voltage sags due to excessive current draw when a large inductive load is turned on).

A BOR generates a Reset pulse, which resets the device. The BOR selects the clock source, based on the device Configuration bit values (FNOSC<2:0> and POSCMD<1:0>).

If an oscillator mode is selected, the BOR activates the Oscillator Start-up Timer (OST). The system clock is held until OST expires. If the PLL is used, the clock is held until the LOCK bit (OSCCON<5>) is '`1`'.

Concurrently, the PWRT time-out (TPWRT) will be applied before the internal Reset is released. If TPWRT = `0` and a crystal oscillator is being used, a nominal delay of TFSCM = `100` is applied. The total delay in this case is TFSCM.

The BOR Status bit (RCON<1>) is set to indicate that a BOR has occurred. If the BOR circuit is enabled, it continues to operate while in Sleep or Idle mode and resets the device in case VDD falls below the BOR threshold voltage.

18.4 Watchdog Timer (WDT)

For PIC24HJ32GP202/204 and PIC24HJ16GP304 devices, the WDT is driven by the LPRC oscillator. When the WDT is enabled, the clock source is also enabled.

18.4.1 PRESCALER/POSTSCALER

The nominal WDT clock source from LPRC is 32 kHz. This feeds a prescaler than 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 (T_{WDT}) 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 allows the selection 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 (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

Note: The `CLRWDT` and `PWRSAV` instructions clear the prescaler and postscaler counts when executed.

18.4.2 SLEEP AND IDLE MODES

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.

18.4.3 ENABLING WDT

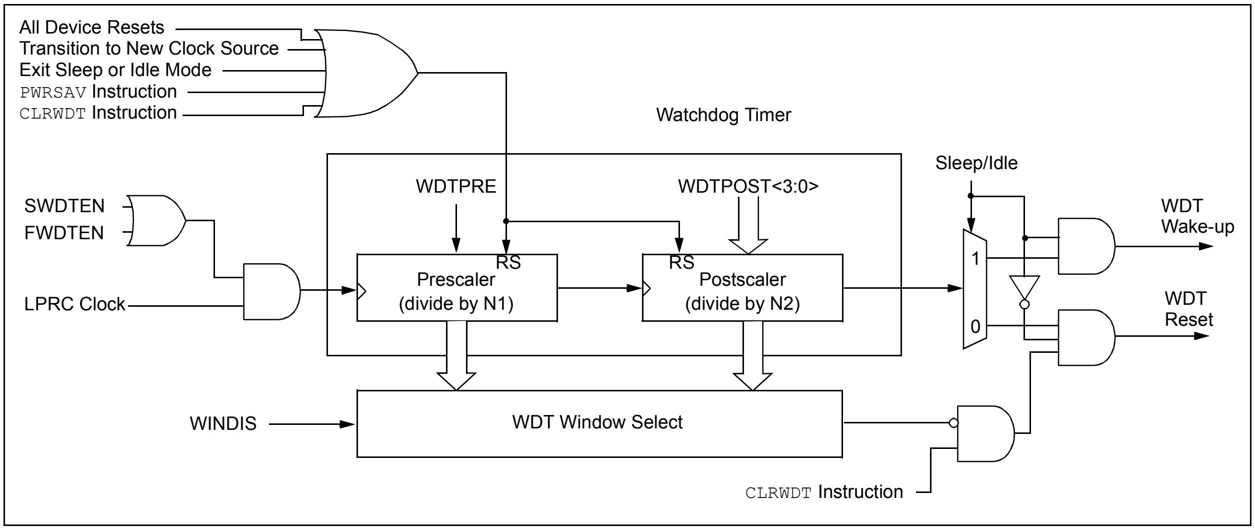
The WDT is enabled or disabled by the FWDTEN Configuration bit in the FWDT Configuration register. When the FWDTEN Configuration bit is set, the WDT is always enabled.

The WDT 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.

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 application 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 18-2: WDT BLOCK DIAGRAM



18.5 JTAG Interface

PIC24HJ32GP202/204 and PIC24HJ16GP304 devices implement a JTAG interface, which supports boundary scan device testing, as well as in-circuit programming. Detailed information on this interface will be provided in future revisions of the document.

18.6 Code Protection and CodeGuard™ Security

The PIC24HJ32GP202/204 and PIC24HJ16GP304 product families offer the intermediate implementation of CodeGuard Security. CodeGuard Security allows multiple parties to securely share resources (memory, interrupts and peripherals) on a single chip. This feature helps to protect individual Intellectual Property in collaborative system designs.

When coupled with software encryption libraries, CodeGuard Security can be used to securely update Flash even when multiple IPs reside on the single chip.

The code protection features are controlled by the Configuration registers: FBS and FGS. The Secure segment and RAM is not implemented.

Note: Refer to *CodeGuard Security Reference Manual (DS70180)* for further information on usage, configuration and operation of CodeGuard Security.

TABLE 18-3: CODE FLASH SECURITY SEGMENT SIZES FOR 32 KBYTE DEVICES

CONFIG BITS	
BSS<2:0>=x11 **0K**	VS = 256 IW GS = 3840 IW 000000h 0001FEh 000200h 0007FEh 000800h 001FFEh 002000h 003FFEh 004000h 0057FEh
BSS<2:0>=x10 **256**	VS = 256 IW BS = 768 IW GS = 10249 IW 000000h 0001FEh 000200h 0007FEh 000800h 001FFEh 002000h 003FFEh 004000h 0057FEh
BSS<2:0>=x01 **768**	VS = 256 IW BS = 3840 IW GS = 7168 IW 000000h 0001FEh 000200h 0007FEh 000800h 001FFEh 002000h 003FFEh 004000h 0057FEh
BSS<2:0>=x00 **1792**	VS = 256 IW BS = 7936 IW GS = 3072 IW 000000h 0001FEh 000200h 0007FEh 000800h 001FFEh 002000h 003FFEh 004000h 0057FEh

TABLE 18-4: CODE FLASH SECURITY SEGMENT SIZES FOR 16 KBYTE DEVICES

CONFIG BITS	
BSS<2:0>=x11 **0K**	VS = 256 IW GS = 3840 IW 000000h 0001FEh 000200h 0007FEh 000800h 001FFEh 002000h 002BFEh
BSS<2:0>=x10 **256**	VS = 256 IW BS = 768 IW GS = 4608 IW 000000h 0001FEh 000200h 0007FEh 000800h 001FFEh 002000h 002BFEh
BSS<2:0>=x01 **768**	VS = 256 IW BS = 3840 IW GS = 1536 IW 000000h 0001FEh 000200h 0007FEh 000800h 001FFEh 002000h 002BFEh
BSS<2:0>=x00 **1792**	VS = 256 IW BS = 5376 IW 000000h 0001FEh 000200h 0007FEh 000800h 001FFEh 002000h 002BFEh

18.7 In-Circuit Serial Programming

PIC24HJ32GP202/204 and PIC24HJ16GP304 family microcontrollers can be serially programmed while in the end application circuit. This is done with two lines for clock and data, and three other lines for power, ground and the programming sequence. Serial programming allows customers to manufacture boards with unprogrammed devices and then program the microcontroller just before shipping the product. Serial programming also allows the most recent firmware or a custom firmware to be programmed. Refer to the "*dsPIC30F/33F Flash Programming Specification*" (DS70152) document for details about In-Circuit Serial Programming (ICSP).

Any of the following three pairs of programming clock/data pins can be used:

- PGC1/EMUC1 and PGD1/EMUD1
- PGC2/EMUC2 and PGD2/EMUD2
- PGC3/EMUC3 and PGD3/EMUD3

18.8 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 Emulation/Debug Clock (EMUCx) and Emulation/Debug Data (EMUDx) pin functions.

Any of the following three pairs of debugging clock/data pins can be used:

- PGC1/EMUC1 and PGD1/EMUD1
- PGC2/EMUC2 and PGD2/EMUD2
- PGC3/EMUC3 and PGD3/EMUD3

To make use of the in-circuit debugger function of the device, the design must implement ICSP connections to $\overline{\text{MCLR}}$, VDD, VSS, 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.

NOTES:

19.0 INSTRUCTION SET SUMMARY

Note: This data sheet summarizes the features of this group of PIC24HJ32GP202/204 and PIC24HJ16GP304 devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the *"PIC24H Family Reference Manual"*.

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 19-1 shows the general symbols used in describing the instructions.

The PIC24H instruction set summary in Table 19-2 lists all the instructions, along with the status flags affected by each instruction.

Most word or byte-oriented W register instructions (including barrel shift instructions) have three operands:

- The first source operand which is typically a register 'Wb' without any address modifier
- The second source operand which is typically a register 'Ws' with or without an address modifier
- The destination of the result which is typically a register 'Wd' with or without an address modifier

However, word or byte-oriented file register instructions have two operands:

- The file register specified by the value 'f'
- The destination, which could either be the file register 'f' or the W0 register, which is denoted as 'WREG'

Most bit-oriented instructions (including simple rotate/shift instructions) have two operands:

- The W register (with or without an address modifier) or file register (specified by the value of 'Ws' or 'f')
- The bit in the W register or file register (specified by a literal value or indirectly by the contents of register 'Wb')

The literal instructions that involve data movement may use some of the following operands:

- A literal value to be loaded into a W register or file register (specified by the value of 'k')
- The W register or file register where the literal value is to be loaded (specified by 'Wb' or 'f')

However, literal instructions that involve arithmetic or logical operations use some of the following operands:

- The first source operand which is a register 'Wb' without any address modifier
- The second source operand which is a literal value
- The destination of the result (only if not the same as the first source operand) which is typically a register 'Wd' with or without an address modifier

The control instructions may use some of the following operands:

- A program memory address
- The mode of the table read and table write instructions

All instructions are single word. Ceratian of them 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 19-1: SYMBOLS USED IN OPCODE DESCRIPTIONS

Field	Description
#text	Means literal defined by "`text`"
(text)	Means "content of `text`"
[text]	Means "the location addressed by `text`"
{ }	Optional field or operation
<n:m>	Register bit field
.b	Byte mode selection
.d	Double Word mode selection
.S	Shadow register select
.w	Word mode selection (default)
bit4	4-bit bit selection field (used in word addressed instructions) ∈ {0...15}
C, DC, N, OV, Z	MCU Status bits: Carry, Digit Carry, Negative, Overflow, Sticky Zero
Expr	Absolute address, label or expression (resolved by the linker)
f	File register address ∈ {0x0000...0x1FFF}
lit1	1-bit unsigned literal ∈ {0,1}
lit4	4-bit unsigned literal ∈ {0...15}
lit5	5-bit unsigned literal ∈ {0...31}
lit8	8-bit unsigned literal ∈ {0...255}
lit10	10-bit unsigned literal ∈ {0...255} for Byte mode, {0:1023} for Word mode
lit14	14-bit unsigned literal ∈ {0...16384}
lit16	16-bit unsigned literal ∈ {0...65535}
lit23	23-bit unsigned literal ∈ {0...8388608}; LSB must be '`0`'
None	Field does not require an entry, may be blank
PC	Program Counter
Slit10	10-bit signed literal ∈ {-512...511}
Slit16	16-bit signed literal ∈ {-32768...32767}
Slit6	6-bit signed literal ∈ {-16...16}
Wb	Base W register ∈ {W0..W15}
Wd	Destination W register ∈ { Wd, [Wd], [Wd++], [Wd--], [++Wd], [--Wd] }
Wdo	Destination W register ∈ { Wnd, [Wnd], [Wnd++], [Wnd--], [++Wnd], [--Wnd], [Wnd+Wb] }
Wm,Wn	Dividend, Divisor working register pair (direct addressing)
Wm*Wm	Multiplicand and Multiplier working register pair for Square instructions ∈ {W4 * W4,W5 * W5,W6 * W6,W7 * W7}
Wn	One of 16 working registers ∈ {W0..W15}

TABLE 19-1: SYMBOLS USED IN OPCODE DESCRIPTIONS (CONTINUED)

Field	Description
Wnd	One of 16 destination working registers ∈ {W0..W15}
Wns	One of 16 source working registers ∈ {W0..W15}
WREG	W0 (working register used in file register instructions)
Ws	Source W register ∈ { Ws, [Ws], [Ws++], [Ws--], [++Ws], [--Ws] }
Wso	Source W register ∈ { Wns, [Wns], [Wns++], [Wns--], [++Wns], [--Wns], [Wns+Wb] }

TABLE 19-2: 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 = Arithmetic Right Shift f	1	1	C,N,OV,Z
		`ASR f,WREG`	WREG = Arithmetic Right Shift f	1	1	C,N,OV,Z
		`ASR Ws,Wd`	Wd = Arithmetic Right Shift Ws	1	1	C,N,OV,Z
		`ASR Wb,Wns,Wnd`	Wnd = Arithmetic Right Shift Wb by Wns	1	1	N,Z
		`ASR Wb,#lit5,Wnd`	Wnd = Arithmetic Right Shift Wb by lit5	1	1	N,Z
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 19-2: 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 = $\overline{f}$	1	1	N,Z
		`COM f,WREG`	WREG = $\overline{f}$	1	1	N,Z
		`COM Ws,Wd`	Wd = $\overline{Ws}$	1	1	N,Z
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 – $\overline{C}$)	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 ¼	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

TABLE 19-2: 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 = Logical Right Shift f	1	1	C,N,OV,Z
		LSR f,WREG	WREG = Logical Right Shift f	1	1	C,N,OV,Z
		LSR Ws,Wd	Wd = Logical Right Shift Ws	1	1	C,N,OV,Z
		LSR Wb,Wns,Wnd	Wnd = Logical Right Shift Wb by Wns	1	1	N,Z
		LSR Wb,#lit5,Wnd	Wnd = Logical Right Shift Wb by lit5	1	1	N,Z
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} = signed(Wb) * signed(Ws)	1	1	None
		MUL.SU Wb,Ws,Wnd	{Wnd + 1, Wnd} = signed(Wb) * unsigned(Ws)	1	1	None
		MUL.US Wb,Ws,Wnd	{Wnd + 1, Wnd} = unsigned(Wb) * signed(Ws)	1	1	None
		MUL.UU Wb,Ws,Wnd	{Wnd + 1, Wnd} = unsigned(Wb) * unsigned(Ws)	1	1	None
		MUL.SU Wb,#lit5,Wnd	{Wnd + 1, Wnd} = signed(Wb) * unsigned(lit5)	1	1	None
		MUL.UU Wb,#lit5,Wnd	{Wnd + 1, Wnd} = unsigned(Wb) * unsigned(lit5)	1	1	None
		MUL f	W3:W2 = f * WREG	1	1	None
42	NEG	NEG f	f = $\overline{f}$ + 1	1	1	C,DC,N,OV,Z
		NEG f,WREG	WREG = $\overline{f}$ + 1	1	1	C,DC,N,OV,Z
		NEG Ws,Wd	Wd = $\overline{Ws}$ + 1	1	1	C,DC,N,OV,Z
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	PWRSAV	PWRSAV #lit1	Go into Sleep or Idle mode	1	1	WDTO,Sleep

TABLE 19-2: 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 – $(\overline{C})$	1	1	C,DC,N,OV,Z
		SUBB f,WREG	WREG = f – WREG – $(\overline{C})$	1	1	C,DC,N,OV,Z
		SUBB #lit10,Wn	Wn = Wn – lit10 – $(\overline{C})$	1	1	C,DC,N,OV,Z
		SUBB Wb,Ws,Wd	Wd = Wb – Ws – $(\overline{C})$	1	1	C,DC,N,OV,Z
		SUBB Wb,#lit5,Wd	Wd = Wb – lit5 – $(\overline{C})$	1	1	C,DC,N,OV,Z
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 – $(\overline{C})$	1	1	C,DC,N,OV,Z
		SUBBR f,WREG	WREG = WREG – f – $(\overline{C})$	1	1	C,DC,N,OV,Z
		SUBBR Wb,Ws,Wd	Wd = Ws – Wb – $(\overline{C})$	1	1	C,DC,N,OV,Z
		SUBBR Wb,#lit5,Wd	Wd = lit5 – Wb – $(\overline{C})$	1	1	C,DC,N,OV,Z
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

TABLE 19-2: INSTRUCTION SET OVERVIEW (CONTINUED)

Base Instr #	Assembly Mnemonic	Assembly Syntax	Description	# of Words	# of Cycles	Status Flags Affected
66	`TBLRDL`	`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

20.0 DEVELOPMENT SUPPORT

The PIC® microcontrollers are supported with a full range of hardware and software development tools:

- Integrated Development Environment
  - MPLAB® IDE Software
- Assemblers/Compilers/Linkers
  - MPASM™ Assembler
  - MPLAB C18 and MPLAB C30 C Compilers
  - MPLINK™ Object Linker/
    MPLIB™ Object Librarian
  - MPLAB ASM30 Assembler/Linker/Library
- Simulators
  - MPLAB SIM Software Simulator
- Emulators
  - MPLAB ICE 2000 In-Circuit Emulator
  - MPLAB REAL ICE™ In-Circuit Emulator
- In-Circuit Debugger
  - MPLAB ICD 2
- Device Programmers
  - PICSTART® Plus Development Programmer
  - MPLAB PM3 Device Programmer
  - PICkit™ 2 Development Programmer
- Low-Cost Demonstration and Development Boards and Evaluation Kits

20.1 MPLAB Integrated Development Environment Software

The MPLAB IDE software brings an ease of software development previously unseen in the 8/16-bit microcontroller market. The MPLAB IDE is a Windows® operating system-based application that contains:

- A single graphical interface to all debugging tools
  - Simulator
  - Programmer (sold separately)
  - Emulator (sold separately)
  - In-Circuit Debugger (sold separately)
- A full-featured editor with color-coded context
- A multiple project manager
- Customizable data windows with direct edit of contents
- High-level source code debugging
- Visual device initializer for easy register initialization
- Mouse over variable inspection
- Drag and drop variables from source to watch windows
- Extensive on-line help
- Integration of select third party tools, such as HI-TECH Software C Compilers and IAR C Compilers

The MPLAB IDE allows you to:

- Edit your source files (either assembly or C)
- One touch assemble (or compile) and download to PIC 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.

20.2 MPASM Assembler

The MPASM Assembler is a full-featured, universal macro assembler for all PIC 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

20.3 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 and PIC24 families of microcontrollers and the dsPIC30 and dsPIC33 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.

20.4 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK Object Linker combines relocatable objects created by the MPASM Assembler and the MPLAB C18 C Compiler. It can link relocatable objects from precompiled libraries, using directives from a linker script.

The MPLIB Object Librarian manages the creation and modification of library files of precompiled code. When a routine from a library is called from a source file, only the modules that contain that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications.

The object linker/library features include:

- Efficient linking of single libraries instead of many smaller files
- Enhanced code maintainability by grouping related modules together
- Flexible creation of libraries with easy module listing, replacement, deletion and extraction

20.5 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

20.6 MPLAB SIM Software Simulator

The MPLAB SIM Software Simulator allows code development in a PC-hosted environment by simulating the PIC MCUs and dsPIC® DSCs on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a comprehensive stimulus controller. Registers can be logged to files for further run-time analysis. The trace buffer and logic analyzer display extend the power of the simulator to record and track program execution, actions on I/O, most peripherals and internal registers.

The MPLAB SIM Software Simulator fully supports symbolic debugging using the MPLAB 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.

20.7 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 PIC 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 PIC 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.

20.8 MPLAB REAL ICE In-Circuit Emulator System

MPLAB REAL ICE In-Circuit Emulator System is Microchip's next generation high-speed emulator for Microchip Flash DSC and MCU devices. It debugs and programs PIC® Flash MCUs and dsPIC® Flash DSCs with the easy-to-use, powerful graphical user interface of the MPLAB Integrated Development Environment (IDE), included with each kit.

The MPLAB REAL ICE probe is connected to the design engineer's PC using a high-speed USB 2.0 interface and is connected to the target with either a connector compatible with the popular MPLAB ICD 2 system (RJ11) or with the new high-speed, noise tolerant, Low-Voltage Differential Signal (LVDS) interconnection (CAT5).

MPLAB REAL ICE is field upgradeable through future firmware downloads in MPLAB IDE. In upcoming releases of MPLAB IDE, new devices will be supported, and new features will be added, such as software breakpoints and assembly code trace. MPLAB REAL ICE offers significant advantages over competitive emulators including low-cost, full-speed emulation, real-time variable watches, trace analysis, complex breakpoints, a ruggedized probe interface and long (up to three meters) interconnection cables.

20.9 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 PIC MCUs and can be used to develop for these and other PIC 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 PIC devices.

20.10 MPLAB PM3 Device Programmer

The MPLAB PM3 Device Programmer is a universal, CE compliant device programmer with programmable voltage verification at VDDMIN and VDDMAX for maximum reliability. It features a large LCD display (128 x 64) for menus and error messages and a modular, detachable socket assembly to support various package types. The ICSP™ cable assembly is included as a standard item. In Stand-Alone mode, the MPLAB PM3 Device Programmer can read, verify and program PIC devices without a PC connection. It can also set code protection in this mode. The MPLAB PM3 connects to the host PC via an RS-232 or USB cable. The MPLAB PM3 has high-speed communications and optimized algorithms for quick programming of large memory devices and incorporates an SD/MMC card for file storage and secure data applications.

20.11 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 PIC 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.

20.12 PICkit 2 Development Programmer

The PICkit™ 2 Development Programmer is a low-cost programmer and selected Flash device debugger 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.

20.13 Demonstration, Development and Evaluation Boards

A wide variety of demonstration, development and evaluation boards for various PIC MCUs and dsPIC DSCs allows quick application development on fully functional systems. Most boards include prototyping areas for adding custom circuitry and provide application firmware and source code for examination and modification.

The boards support a variety of features, including LEDs, temperature sensors, switches, speakers, RS-232 interfaces, LCD displays, potentiometers and additional EEPROM memory.

The demonstration and development boards can be used in teaching environments, for prototyping custom circuits and for learning about various microcontroller applications.

In addition to the PICDEM™ and dsPICDEM™ demonstration/development board series of circuits, Microchip has a line of evaluation kits and demonstration software for analog filter design, KEELOQ® security ICs, CAN, IrDA®, PowerSmart battery management, SEEVAL® evaluation system, Sigma-Delta ADC, flow rate sensing, plus many more.

Check the Microchip web page (www.microchip.com) for the complete list of demonstration, development and evaluation kits.

21.0 ELECTRICAL CHARACTERISTICS

This section provides an overview of PIC24HJ32GP202/204 and PIC24HJ16GP304 electrical characteristics. Additional information will be provided in future revisions of this document as it becomes available.

Absolute maximum ratings for the PIC24HJ32GP202/204 and PIC24HJ16GP304 family are listed below. Exposure to these maximum rating conditions for extended periods can 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(1)

Ambient temperature under bias	-40°C to +125°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(2)	250 mA
Maximum output current sunk by any I/O pin(3)	4 mA
Maximum output current sourced by any I/O pin(3)	4 mA
Maximum current sunk by all ports	200 mA
Maximum current sourced by all ports(2)	200 mA

Note 1: Stresses above those listed under “Absolute Maximum Ratings” can 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 can affect device reliability.

2: Maximum allowable current is a function of device maximum power dissipation (see Table 21-2).

3: Exceptions are CLKOUT, which is able to sink/source 25 mA, and the VREF+, VREF-, SCLx, SDAx, PGCx and PGDx pins, which are able to sink/source 12 mA.

21.1 DC Characteristics

TABLE 21-1: OPERATING MIPS VS. VOLTAGE

Characteristic	V_{DD} Range (in Volts)	Temp Range (in °C)	Max MIPS
			PIC24HJ32GP202/204 and PIC24HJ16GP304
	3.0-3.6V	-40°C to +85°C	40
	3.0-3.6V	-40°C to +125°C	40

TABLE 21-2: THERMAL OPERATING CONDITIONS

Rating	Symbol	Min	Typ	Max	Unit
Industrial Temperature Devices					
Operating Junction Temperature Range	T_J	-40	—	+125	°C
Operating Ambient Temperature Range	T_A	-40	—	+85	°C
Extended Temperature Devices					
Operating Junction Temperature Range	T_J	-40	—	+140	°C
Operating Ambient Temperature Range	T_A	-40	—	+125	°C
Power Dissipation: Internal chip power dissipation: $P_{INT} = V_{DD} \text{ x } (I_{DD} - \Sigma I_{OH})$ I/O Pin Power Dissipation: $I/O = \Sigma (\{V_{DD} - V_{OH}\} \text{ x } I_{OH}) + \Sigma (V_{OL} \text{ x } I_{OL})$	P_D	$P_{INT} + P_{I/O}$			W
Maximum Allowed Power Dissipation	P_{DMAX}	$(T_J - T_A)/\theta_{JA}$			W

TABLE 21-3: THERMAL PACKAGING CHARACTERISTICS

Characteristic	Symbol	Typ	Max	Unit	Notes
Package Thermal Resistance, 44-pin QFN	θ_{JA}	62.4	—	°C/W	**1**
Package Thermal Resistance, 44-pin TFQP	θ_{JA}	60	—	°C/W	**1**
Package Thermal Resistance, 28-pin SPDIP	θ_{JA}	108	—	°C/W	**1**
Package Thermal Resistance, 28-pin SOIC	θ_{JA}	80.2	—	°C/W	**1**
Package Thermal Resistance, 28-pin QFN-S	θ_{JA}	32	—	°C/W	**1**

Note 1: Junction to ambient thermal resistance, Theta-JA (θ_{JA}) numbers are achieved by package simulations.

TABLE 21-4: DC TEMPERATURE AND VOLTAGE SPECIFICATIONS

DC CHARACTERISTICS			**Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic**	**Min**	**Typ(1)**	**Max**	**Units**	**Conditions**
Operating Voltage							
DC10	**Supply Voltage**						
	VDD		3.0	—	3.6	V	Industrial and Extended
DC12	VDR	**RAM Data Retention Voltage(2)**	1.1	1.3	1.8	V	
DC16	VPOR	**VDD Start Voltage** to ensure internal Power-on Reset signal	—	—	VSS	V	
DC17	SVDD	**VDD Rise Rate** to ensure internal Power-on Reset signal	0.03	—	—	V/ms	0-3.0V in 0.1s
DC18	VCORE	**VDD Core(3)** Internal regulator voltage	2.25	—	2.75	V	Voltage is dependent on load, temperature and VDD

Note 1: Data in “Typ” column is at 3.3V, 25°C unless otherwise stated.

2: This is the limit to which VDD can be lowered without losing RAM data.

3: These parameters are characterized but not tested in manufacturing.

TABLE 21-5: DC CHARACTERISTICS: OPERATING CURRENT (IDD)

DC CHARACTERISTICS			**Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended			
Parameter No.	**Typical[1]**	**Max**	**Units**	**Conditions**		
Operating Current (IDD)[2]						
DC20d	24	30	mA	-40°C	3.3V	10 MIPS
DC20a	27	30	mA	+25°C		
DC20b	27	30	mA	+85°C		
DC20c	27	35	mA	+125°C		
DC21d	30	40	mA	-40°C	3.3V	16 MIPS
DC21a	37	40	mA	+25°C		
DC21b	32	45	mA	+85°C		
DC21c	33	45	mA	+125°C		
DC22d	35	50	mA	-40°C	3.3V	20 MIPS
DC22a	38	50	mA	+25°C		
DC22b	38	55	mA	+85°C		
DC22c	39	55	mA	+125°C		
DC23d	47	70	mA	-40°C	3.3V	30 MIPS
DC23a	48	70	mA	+25°C		
DC23b	48	70	mA	+85°C		
DC23c	48	70	mA	+125°C		
DC24d	56	90	mA	-40°C	3.3V	40 MIPS
DC24a	56	90	mA	+25°C		
DC24b	54	90	mA	+85°C		
DC24c	54	80	mA	+125°C	3.3V	35 MIPS

Note 1: Data in “Typical” column is at 3.3V, 25°C unless otherwise stated.

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. $\overline{\text{MCLR}}$ = 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).

TABLE 21-6: DC CHARACTERISTICS: IDLE CURRENT (IIDLE)

DC CHARACTERISTICS			**Standard Operating Conditions: 3.0V to 3.6V** **(unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended			
Parameter No.	**Typical(1)**	**Max**	**Units**	**Conditions**		
Idle Current (IIDLE): Core OFF Clock ON Base Current(2)						
DC40d	3	25	mA	-40°C	3.3V	10 MIPS
DC40a	3	25	mA	+25°C		
DC40b	3	25	mA	+85°C		
DC40c	3	25	mA	+125°C		
DC41d	4	25	mA	-40°C	3.3V	16 MIPS
DC41a	4	25	mA	+25°C		
DC41b	5	25	mA	+85°C		
DC41c	5	25	mA	+125°C		
DC42d	6	25	mA	-40°C	3.3V	20 MIPS
DC42a	6	25	mA	+25°C		
DC42b	7	25	mA	+85°C		
DC42c	7	25	mA	+125°C		
DC43d	9	25	mA	-40°C	3.3V	30 MIPS
DC43a	9	25	mA	+25°C		
DC43b	9	25	mA	+85°C		
DC43c	9	25	mA	+125°C		
DC44d	10	25	mA	-40°C	3.3V	40 MIPS
DC44a	10	25	mA	+25°C		
DC44b	16	25	mA	+85°C		
DC44c	10	25	mA	+125°C	3.3V	35 MIPS

Note 1: Data in “Typical” column is at 3.3V, 25°C unless otherwise stated.

2: Base IIDLE 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 VSS.

TABLE 21-7: DC CHARACTERISTICS: POWER-DOWN CURRENT (IPD)

DC CHARACTERISTICS				**Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended		
Parameter No.	**Typical(1)**	**Max**	**Units**	**Conditions**		
Power-Down Current (IPD)(2)						
DC60d	55	500	μA	-40°C	3.3V	Base Power-Down Current(3,4)
DC60a	63	500	μA	+25°C		
DC60b	85	500	μA	+85°C		
DC60c	146	1	mA	+125°C		
DC61d	8	12	μA	-40°C	3.3V	Watchdog Timer Current: ΔIWDT(3)
DC61a	10	15	μA	+25°C		
DC61b	12	20	μA	+85°C		
DC61c	13	25	μA	+125°C		

Note 1: Data in the Typical column is at 3.3V, 25°C unless otherwise stated.

2: Base IPD is measured with all peripherals and clocks shut down. All I/Os are configured as inputs and pulled to VSS. 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 IPD current.

4: These currents are measured on the device containing the most memory in this family.

TABLE 21-8: DC CHARACTERISTICS: DOZE CURRENT (IDOZE)

DC CHARACTERISTICS			**Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Parameter No.	**Typical(1)**	**Max**	**Doze Ratio**	**Units**	**Conditions**		
DC73a	25	32	1:2	mA	-40°C	3.3V	40 MIPS
DC73f	23	27	1:64	mA			
DC73g	23	26	1:128	mA			
DC70a	42	47	1:2	mA	+25°C	3.3V	40 MIPS
DC70f	26	27	1:64	mA			
DC70g	25	27	1:128	mA			
DC71a	41	48	1:2	mA	+85°C	3.3V	40 MIPS
DC71f	25	28	1:64	mA			
DC71g	24	28	1:128	mA			
DC72a	42	49	1:2	mA	+125°C	3.3V	35 MIPS
DC72f	26	29	1:64	mA			
DC72g	25	28	1:128	mA			

Note 1: Data in the Typical column is at 3.3V, 25°C unless otherwise stated.

TABLE 21-9: DC CHARACTERISTICS: I/O PIN INPUT SPECIFICATIONS

DC CHARACTERISTICS			**Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic**	**Min**	**Typ(1)**	**Max**	**Units**	**Conditions**
	VIL	**Input Low Voltage**					
DI10		I/O pins	VSS	—	0.2 VDD	V	
DI15		$\overline{\text{MCLR}}$	VSS	—	0.2 VDD	V	
DI16		OSC1 (XT mode)	VSS	—	0.2 VDD	V	
DI17		OSC1 (HS mode)	VSS	—	0.2 VDD	V	
DI18		SDAx, SCLx	VSS	—	0.3 VDD	V	SMBus disabled
DI19		SDAx, SCLx	VSS	—	0.2 VDD	V	SMBus enabled
	VIH	**Input High Voltage**					
DI20		I/O pins:					
		with analog functions	0.8 VDD	—	VDD	V	
		digital-only	0.8 VDD	—	5.5	V	
DI25		$\overline{\text{MCLR}}$	0.8 VDD	—	VDD	V	
DI26		OSC1 (XT mode)	0.7 VDD	—	VDD	V	
DI27		OSC1 (HS mode)	0.7 VDD	—	VDD	V	
DI28		SDAx, SCLx	0.7 VDD	—	VDD	V	SMBus disabled
DI29		SDAx, SCLx	0.8 VDD	—	VDD	V	SMBus enabled
	ICNPU	**CNx Pull-up Current**					
DI30			50	250	400	µA	VDD = 3.3V, VPIN = VSS
	IIL	**Input Leakage Current(2)(3)**					
DI50		I/O ports	—	—	±2	µA	VSS ≤ VPIN ≤ VDD, Pin at high-impedance
DI51		Analog Input Pins	—	—	±1	µA	VSS ≤ VPIN ≤ VDD, Pin at high-impedance, 40°C ≤ TA ≤ +85°C
DI51a		Analog Input Pins	—	—	±2	µA	Analog pins shared with external reference pins, 40°C ≤ TA ≤ +85°C
DI51b		Analog Input Pins	—	—	±3.5	µA	VSS ≤ VPIN ≤ VDD, Pin at high-impedance, -40°C ≤ TA ≤ +125°C
DI51c		Analog Input Pins	—	—	±8	µA	Analog pins shared with external reference pins, -40°C ≤ TA ≤ +125°C
DI55		$\overline{\text{MCLR}}$	—	—	±2	µA	VSS ≤ VPIN ≤ VDD
DI56		OSC1	—	—	±2	µA	VSS ≤ VPIN ≤ VDD, XT and HS modes

Note 1: Data in “Typ” column is at 3.3V, 25°C unless otherwise stated.

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 21-10: DC CHARACTERISTICS: I/O PIN OUTPUT SPECIFICATIONS

DC CHARACTERISTICS			**Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)** Operating temperature -40°C ≤ T_A ≤ +85°C for Industrial -40°C ≤ T_A ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic**	**Min**	**Typ**	**Max**	**Units**	**Conditions**
	V_{OL}	**Output Low Voltage**					
DO10		I/O ports	—	—	0.4	V	I_{OL} = 2mA, V_{DD} = 3.3V
DO16		OSC2/CLKO	—	—	0.4	V	I_{OL} = 2mA, V_{DD} = 3.3V
	V_{OH}	**Output High Voltage**					
DO20		I/O ports	2.40	—	—	V	I_{OH} = -2.3 mA, V_{DD} = 3.3V
DO26		OSC2/CLKO	2.41	—	—	V	I_{OH} = -1.3 mA, V_{DD} = 3.3V

TABLE 21-11: ELECTRICAL CHARACTERISTICS: BOR

DC CHARACTERISTICS			**Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)** Operating temperature -40°C ≤ T_A ≤ +85°C for Industrial -40°C ≤ T_A ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic**	**Min**(1)	**Typ**	**Max**	**Units**	**Conditions**
BO10	V_{BOR}	BOR Event on V_{DD} transition high-to-low BOR event is tied to V_{DD} core voltage decrease	2.40	—	2.55	V	

Note 1: Parameters are for design guidance only and are not tested in manufacturing.

TABLE 21-12: DC CHARACTERISTICS: PROGRAM MEMORY

DC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	Symbol	Characteristic	Min	Typ[1]	Max	Units	Conditions
		Program Flash Memory					
D130	EP	Cell Endurance	10,000	—	—	E/W	-40°C to +125°C
D131	VPR	VDD for Read	VMIN	—	3.6	V	VMIN = Minimum operating voltage
D132B	VPEW	VDD for Self-Timed Write	VMIN	—	3.6	V	VMIN = Minimum operating voltage
D134	TRETD	Characteristic Retention	20	—	—	Year	Provided no other specifications are violated, -40°C to +125°C
D135	IDDP	Supply Current during Programming	—	10	—	mA	
D136	TRW	Row Write Time	—	1.6	—	ms	
D137	TPE	Page Erase Time	—	20	—	ms	
D138	TWW	Word Write Cycle Time	20	—	40	µs	

Note 1: Data in “Typ” column is at 3.3V, 25°C unless otherwise stated.

TABLE 21-13: INTERNAL VOLTAGE REGULATOR SPECIFICATIONS

Operating Conditions: -40°C < TA < +85°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)

21.2 AC Characteristics and Timing Parameters

The information contained in this section defines PIC24HJ32GP202/204 and PIC24HJ16GP304 AC characteristics and timing parameters.

TABLE 21-14: TEMPERATURE AND VOLTAGE SPECIFICATIONS – AC

AC CHARACTERISTICS	**Standard Operating Conditions: 3.0V to 3.6V** **(unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended Operating voltage VDD range as described in **Section 21.0 “Electrical Characteristics”**.

FIGURE 21-1: LOAD CONDITIONS FOR DEVICE TIMING SPECIFICATIONS

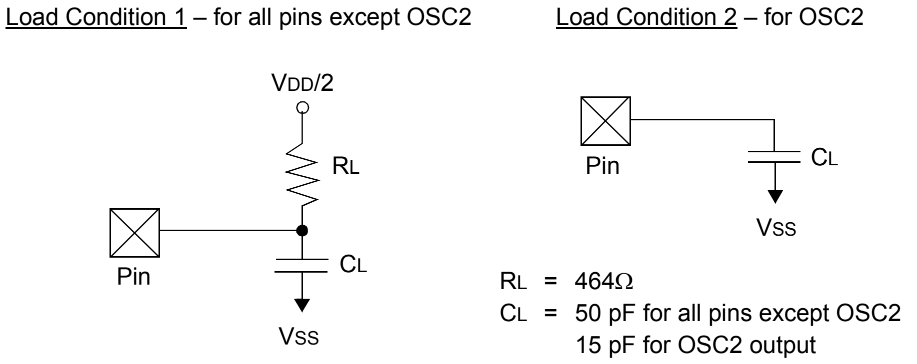


TABLE 21-15: CAPACITIVE LOADING REQUIREMENTS ON OUTPUT PINS

Param No.	Symbol	Characteristic	Min	Typ	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^{2}C™ mode

FIGURE 21-2: EXTERNAL CLOCK TIMING

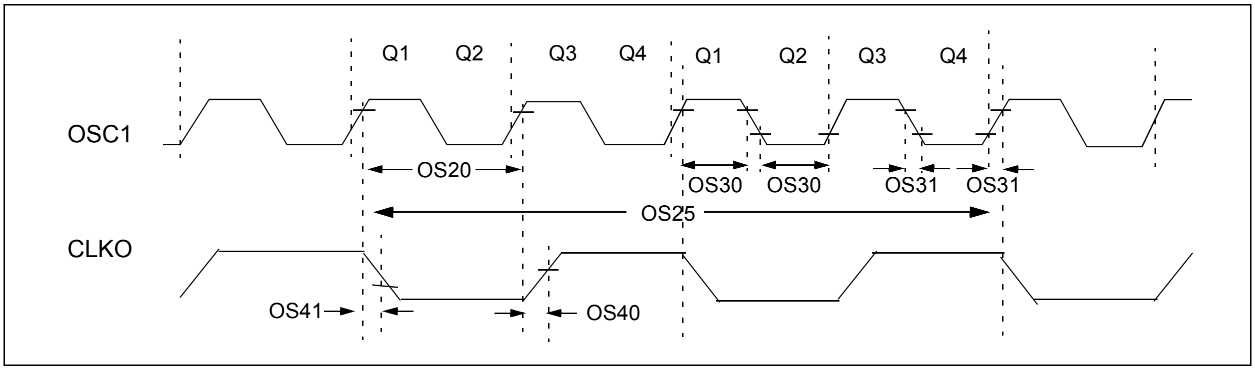


TABLE 21-16: EXTERNAL CLOCK TIMING REQUIREMENTS

AC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symb**	**Characteristic**	**Min**	**Typ(1)**	**Max**	**Units**	**Conditions**
OS10	FIN	External CLKI Frequency (External clocks allowed only in EC and ECPLL modes)	DC	—	40	MHz	EC
		Oscillator Crystal Frequency	3.5 10	— — —	10 40 33	MHz MHz kHz	XT HS SOSC
OS20	TOSC	TOSC = 1/FOSC	12.5	—	DC	ns	
OS25	TCY	Instruction Cycle Time(2)	25	—	DC	ns	
OS30	TosL, TosH	External Clock in (OSC1) High or Low Time	0.375 x TOSC	—	0.625 x TOSC	ns	EC
OS31	TosR, TosF	External Clock in (OSC1) Rise or Fall Time	—	—	20	ns	EC
OS40	TckR	CLKO Rise Time(3)	—	5.2	—	ns	
OS41	TckF	CLKO Fall Time(3)	—	5.2	—	ns	

Note 1: Data in “Typ” column is at 3.3V, 25°C unless otherwise stated.

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 can 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.

TABLE 21-17: PLL CLOCK TIMING SPECIFICATIONS (VDD = 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 -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic**	**Min**	**Typ(1)**	**Max**	**Units**	**Conditions**
OS50	FPLLI	PLL Voltage Controlled Oscillator (VCO) Input Frequency Range	0.8	—	8	MHz	ECPLL, XTPLL modes
OS51	FSYS	On-Chip VCO System Frequency	100	—	200	MHz	
OS52	TLOCK	PLL Start-up Time (Lock Time)	0.9	1.5	3.1	ms	
OS53	DCLK	CLKO Stability (Jitter)	-3	0.5	3	%	Measured over 100 ms period

Note 1: Data in “Typ” column is at 3.3V, 25°C unless otherwise stated.

TABLE 21-18: 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 -40°C ≤ TA ≤ +125°C for Extended					
Param No.	**Characteristic**	**Min**	**Typ**	**Max**	**Units**	**Conditions**	
	Internal FRC Accuracy @ FRC Frequency = 7.37 MHz(1,2)						
F20	FRC	-2	—	+2	%	-40°C ≤ TA ≤ +85°C	VDD = 3.0-3.6V
	FRC	-5	—	+5	%	-40°C ≤ TA ≤ +125°C	VDD = 3.0-3.6V

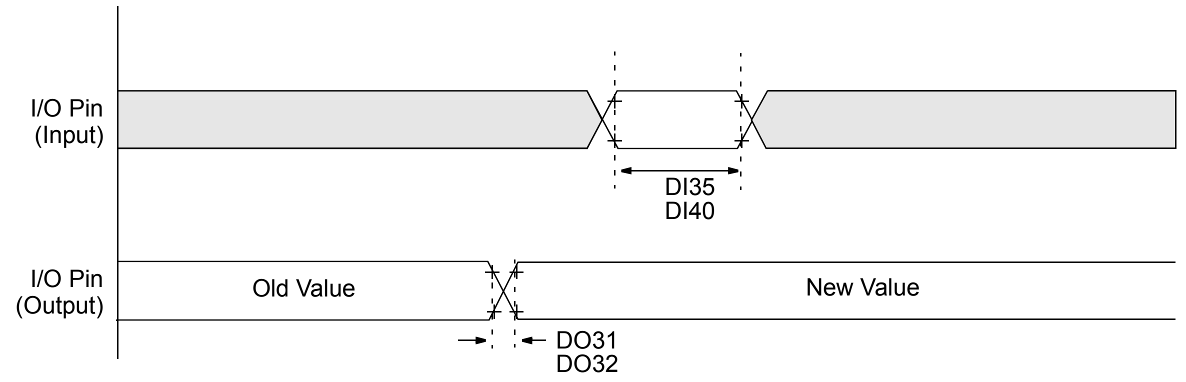
Note 1: Frequency calibrated at 25°C and 3.3V. TUN bits can be used to compensate for temperature drift.
2: FRC is set to initial frequency of 7.37 MHz (±2%) at 25°C.

TABLE 21-19: 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 -40°C ≤ TA ≤ +125°C for Extended					
Param No.	**Characteristic**	**Min**	**Typ**	**Max**	**Units**	**Conditions**	
	LPRC @ 32.768 kHz(1)						
F21	LPRC	-20	±6	+20	%	-40°C ≤ TA ≤ +85°C	VDD = 3.0-3.6V
	LPRC	-70	—	+20	%	-40°C ≤ TA ≤ +125°C	VDD = 3.0-3.6V

Note 1: Change of LPRC frequency as VDD changes.

FIGURE 21-3: I/O TIMING CHARACTERISTICS



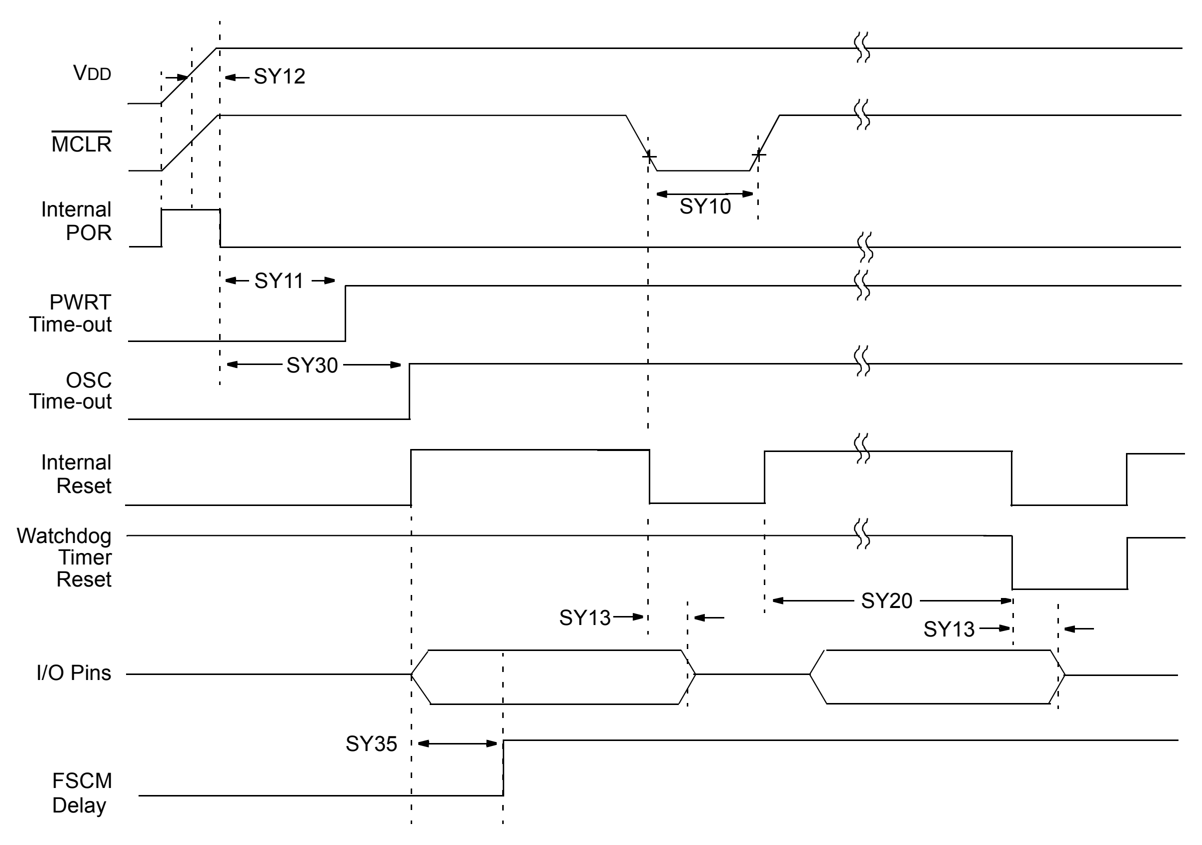
Note: Refer to Figure 21-1 for load conditions.

TABLE 21-20: I/O TIMING REQUIREMENTS

AC CHARACTERISTICS			**Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic**	**Min**	**Typ[1]**	**Max**	**Units**	**Conditions**
DO31	TIOR	Port Output Rise Time	—	10	25	ns	—
DO32	TIOF	Port Output Fall Time	—	10	25	ns	—
DI35	TINP	INTx Pin High or Low Time (output)	20	—	—	ns	—
DI40	TRBP	CNx High or Low Time (input)	2	—	—	TCY	—

Note 1: Data in “Typ” column is at 3.3V, 25°C unless otherwise stated.

FIGURE 21-4: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING CHARACTERISTICS



Note: Refer to Figure 21-1 for load conditions.

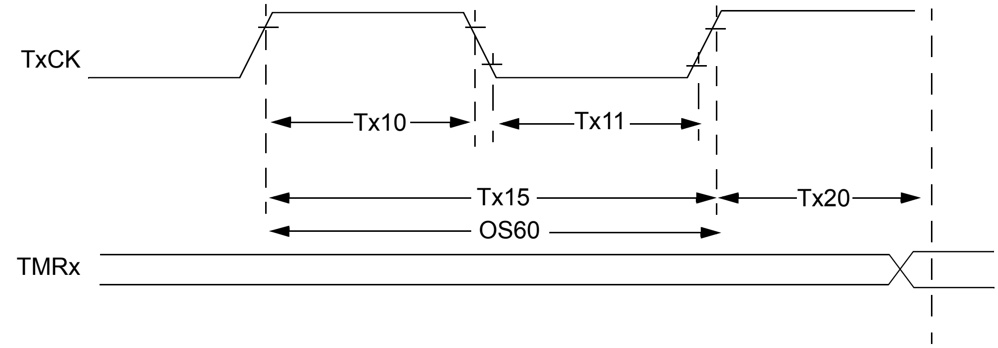
TABLE 21-21: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER TIMING REQUIREMENTS

AC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic[1]**	**Min**	**Typ[2]**	**Max**	**Units**	**Conditions**
SY10	TMCL	$\overline{\text{MCLR}}$ Pulse Width (low)	2	—	—	μs	-40°C to +85°C
SY11	TPWRT	Power-up Timer Period	—	2 4 8 16 32 64 128	—	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 $\overline{\text{MCLR}}$ Low or Watchdog Timer Reset	0.68	0.72	1.2	μs	
SY20	TWDT1	Watchdog Timer Time-out Period (No Prescaler)	1.7	2.1	2.6	ms	VDD = 3V, -40°C to +85°C
SY30	TOST	Oscillator Start-up Time	—	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.

2: Data in “Typ” column is at 3.3V, 25°C unless otherwise stated.

FIGURE 21-5: TIMER1, 2 AND 3 EXTERNAL CLOCK TIMING CHARACTERISTICS



Note: Refer to Figure 21-1 for load conditions.

TABLE 21-22: TIMER1 EXTERNAL CLOCK TIMING REQUIREMENTS[1]

AC CHARACTERISTICS				Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic**		**Min**	**Typ**	**Max**	**Units**	**Conditions**
TA10	TTXH	TxCK High Time	Synchronous, no prescaler	0.5 TCY + 20	—	—	ns	Must also meet parameter TA15
			Synchronous, with prescaler	10	—	—	ns	
			Asynchronous	10	—	—	ns	
TA11	TTXL	TxCK Low Time	Synchronous, no prescaler	0.5 TCY + 20	—	—	ns	Must also meet parameter TA15
			Synchronous, with prescaler	10	—	—	ns	
			Asynchronous	10	—	—	ns	
TA15	TTXP	TxCK Input Period	Synchronous, no prescaler	TCY + 40	—	—	ns	
			Synchronous, with prescaler	Greater of: 20 ns or (TCY + 40)/N	—	—	—	N = prescale value (1, 8, 64, 256)
			Asynchronous	20	—	—	ns	
OS60	Ft1	SOSC1/T1CK Oscillator Input frequency Range (oscillator enabled by setting bit TCS (T1CON<1>))		DC	—	50	kHz	
TA20	TCKEXTMRL	Delay from External TxCK Clock Edge to Timer Increment		0.5 TCY		1.5 TCY	—	

Note 1: Timer1 is a Type A.

TABLE 21-23: TIMER2 EXTERNAL CLOCK TIMING REQUIREMENTS

AC CHARACTERISTICS	**Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended

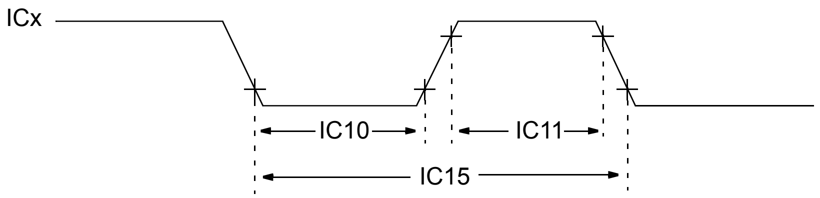
Param No.	Symbol	Characteristic		Min	Typ	Max	Units	Conditions
TB10	TtxH	TxCK High Time	Synchronous, no prescaler	0.5 TCY + 20	—	—	ns	Must also meet parameter TB15
			Synchronous, with prescaler	10	—	—	ns	
TB11	TtxL	TxCK Low Time	Synchronous, no prescaler	0.5 TCY + 20	—	—	ns	Must also meet parameter TB15
			Synchronous, with prescaler	10	—	—	ns	
TB15	TtxP	TxCK Input Period	Synchronous, no prescaler	TCY + 40	—	—	ns	N = prescale value (1, 8, 64, 256)
			Synchronous, with prescaler	Greater of: 20 ns or (TCY + 40)/N				
TB20	TCKEXT-MRL	Delay from External TxCK Clock Edge to Timer Increment		0.5 TCY	—	1.5 TCY	—	

TABLE 21-24: TIMER3 EXTERNAL CLOCK TIMING REQUIREMENTS

AC CHARACTERISTICS	**Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended

Param No.	Symbol	Characteristic		Min	Typ	Max	Units	Conditions
TC10	TtxH	TxCK High Time	Synchronous	0.5 TCY + 20	—	—	ns	Must also meet parameter TC15
TC11	TtxL	TxCK Low Time	Synchronous	0.5 TCY + 20	—	—	ns	Must also meet parameter TC15
TC15	TtxP	TxCK Input Period	Synchronous, no prescaler	TCY + 40	—	—	ns	N = prescale value (1, 8, 64, 256)
			Synchronous, with prescaler	Greater of: 20 ns or (TCY + 40)/N				
TC20	TCKEXT-MRL	Delay from External TxCK Clock Edge to Timer Increment		0.5 TCY	—	1.5 TCY	—	

FIGURE 21-6: INPUT CAPTURE (CAPx) TIMING CHARACTERISTICS



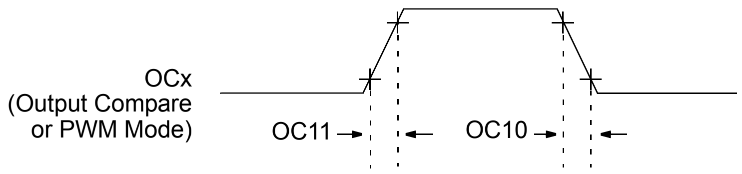
Note: Refer to Figure 21-1 for load conditions.

TABLE 21-25: INPUT CAPTURE TIMING REQUIREMENTS

AC CHARACTERISTICS				**Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended			
Param No.	**Symbol**	**Characteristic(1)**		**Min**	**Max**	**Units**	**Conditions**
IC10	TccL	ICx Input Low Time	No Prescaler	0.5 TCY + 20	—	ns	
			With Prescaler	10	—	ns	
IC11	TccH	ICx Input High Time	No Prescaler	0.5 TCY + 20	—	ns	
			With Prescaler	10	—	ns	
IC15	TccP	ICx Input Period		(TCY + 40)/N	—	ns	N = prescale value (1, 4, 16)

Note 1: These parameters are characterized but not tested in manufacturing.

FIGURE 21-7: OUTPUT COMPARE MODULE (OCx) TIMING CHARACTERISTICS



Note: Refer to Figure 21-1 for load conditions.

TABLE 21-26: OUTPUT COMPARE MODULE TIMING REQUIREMENTS

AC CHARACTERISTICS			**Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic(1)**	**Min**	**Typ**	**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.

FIGURE 21-8: OC/PWM MODULE TIMING CHARACTERISTICS

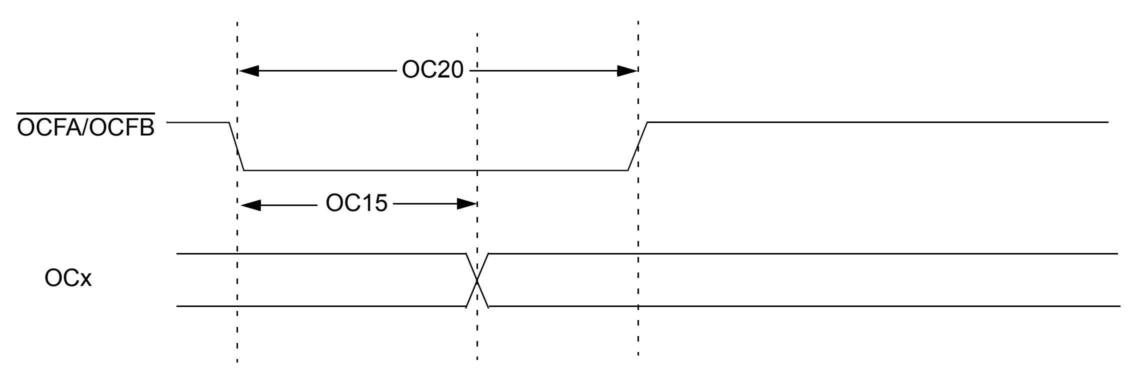
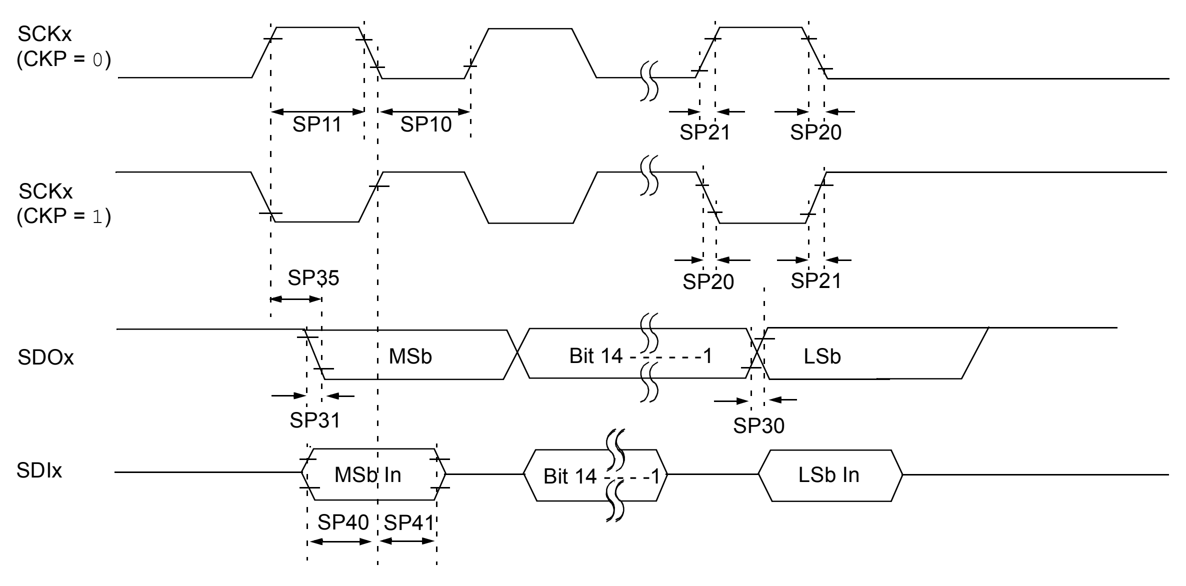


TABLE 21-27: SIMPLE OC/PWM MODE TIMING REQUIREMENTS

AC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic[1]**	**Min**	**Typ**	**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.

FIGURE 21-9: SPIx MODULE MASTER MODE (CKE = 0) TIMING CHARACTERISTICS



Note: Refer to Figure 21-1 for load conditions.

TABLE 21-28: SPIx MASTER MODE (CKE = 0) TIMING REQUIREMENTS

AC CHARACTERISTICS			**Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic[1]**	**Min**	**Typ[2]**	**Max**	**Units**	**Conditions**
SP10	TscL	SCKx Output Low Time	TCY/2	—	—	ns	See **Note 3**
SP11	TscH	SCKx Output High Time	TCY/2	—	—	ns	See **Note 3**
SP20	TscF	SCKx Output Fall Time	—	—	—	ns	See parameter D032 and **Note 4**
SP21	TscR	SCKx Output Rise Time	—	—	—	ns	See parameter D031 and **Note 4**
SP30	TdoF	SDOx Data Output Fall Time	—	—	—	ns	See parameter D032 and **Note 4**
SP31	TdoR	SDOx Data Output Rise Time	—	—	—	ns	See parameter D031 and **Note 4**
SP35	TscH2doV, TscL2doV	SDOx Data Output Valid after SCKx Edge	—	6	20	ns	—
SP40	TdiV2scH, TdiV2scL	Setup Time of SDIx Data Input to SCKx Edge	23	—	—	ns	—
SP41	TscH2diL, TscL2diL	Hold Time of SDIx Data Input to SCKx Edge	30	—	—	ns	—

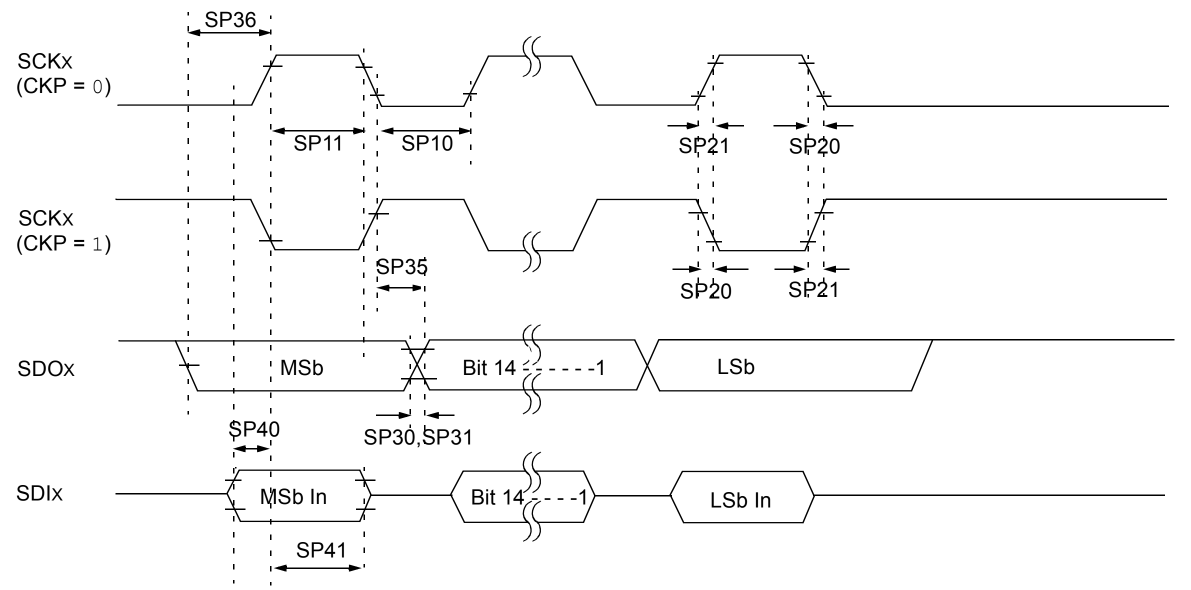
Note 1: These parameters are characterized but not tested in manufacturing.

2: Data in "Typ" column is at 3.3V, 25°C unless otherwise stated.

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 21-10: SPIx MODULE MASTER MODE (CKE = 1) TIMING CHARACTERISTICS



Note: Refer to Figure 21-1 for load conditions.

TABLE 21-29: SPIx MODULE MASTER MODE (CKE = 1) TIMING REQUIREMENTS

AC CHARACTERISTICS			**Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic**[1]	**Min**	**Typ**[2]	**Max**	**Units**	**Conditions**
SP10	TscL	SCKx Output Low Time	TCY/2	—	—	ns	See **Note 3**
SP11	TscH	SCKx Output High Time	TCY/2	—	—	ns	See **Note 3**
SP20	TscF	SCKx Output Fall Time	—	—	—	ns	See parameter D032 and **Note 4**
SP21	TscR	SCKx Output Rise Time	—	—	—	ns	See parameter D031 and **Note 4**
SP30	TdoF	SDOx Data Output Fall Time	—	—	—	ns	See parameter D032 and **Note 4**
SP31	TdoR	SDOx Data Output Rise Time	—	—	—	ns	See parameter D031 and **Note 4**
SP35	TscH2doV, TscL2doV	SDOx Data Output Valid after SCKx Edge	—	6	20	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	23	—	—	ns	—
SP41	TscH2diL, TscL2diL	Hold Time of SDIx Data Input to SCKx Edge	30	—	—	ns	—

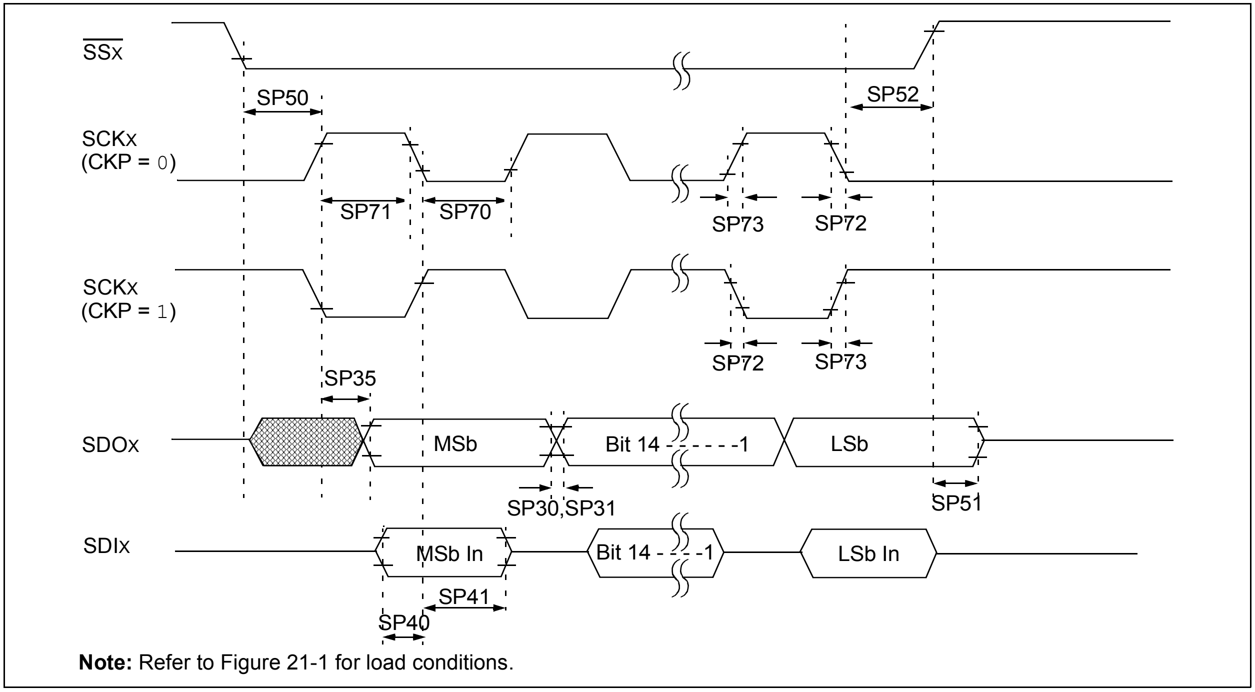
Note 1: These parameters are characterized but not tested in manufacturing.

2: Data in "Typ" column is at 3.3V, 25°C unless otherwise stated.

3: The minimum clock period for SCKx is 100 ns. The clock generated in Master mode must not violate this specification.

4: Assumes 50 pF load on all SPIx pins.

FIGURE 21-11: SPIx MODULE SLAVE MODE (CKE = 0) TIMING CHARACTERISTICS



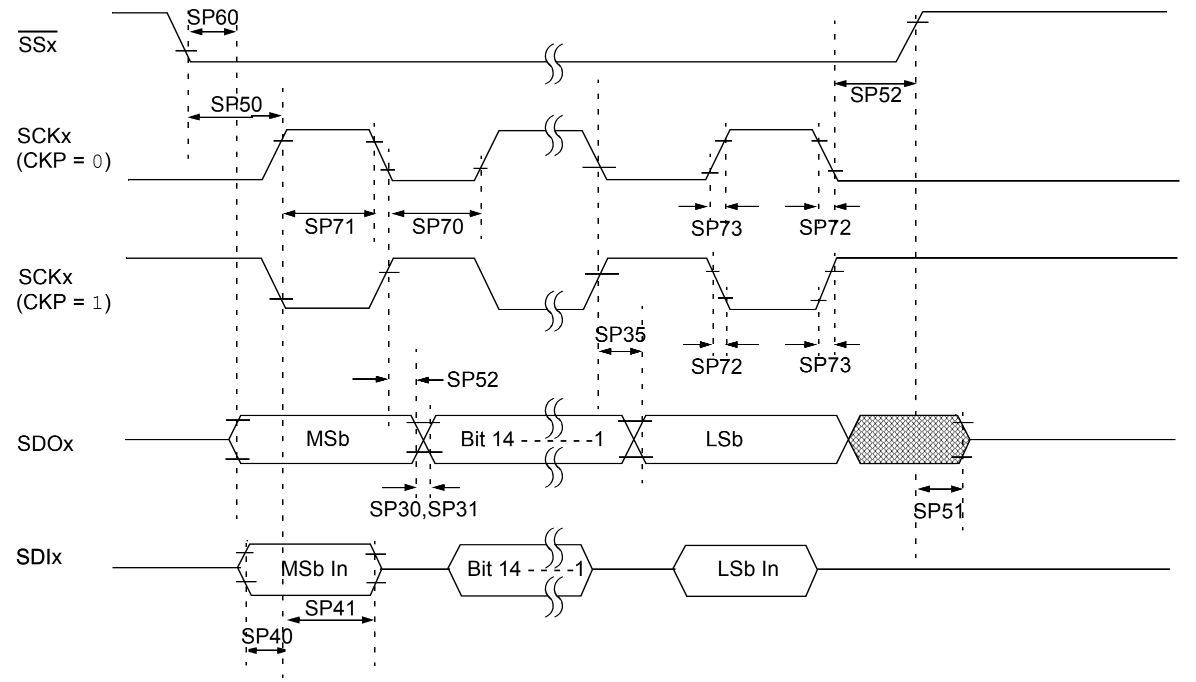
Note: Refer to Figure 21-1 for load conditions.

TABLE 21-30: SPIx MODULE SLAVE MODE (CKE = 0) TIMING REQUIREMENTS

AC CHARACTERISTICS			**Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic[1]**	**Min**	**Typ[2]**	**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	—	10	25	ns	See **Note 3**
SP73	TscR	SCKx Input Rise Time	—	10	25	ns	See **Note 3**
SP30	TdoF	SDOx Data Output Fall Time	—	—	—	ns	See parameter D032 and **Note 3**
SP31	TdoR	SDOx Data Output Rise Time	—	—	—	ns	See parameter D031 and **Note 3**
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	10	—	50	ns	See **Note 3**
SP52	TscH2ssH TscL2ssH	$\overline{\text{SSx}}$ after SCKx Edge	1.5 TCY +40	—	—	ns	—

Note 1: These parameters are characterized but not tested in manufacturing.
2: Data in “Typ” column is at 3.3V, 25°C unless otherwise stated.
3: Assumes 50 pF load on all SPIx pins.

FIGURE 21-12: SPIx MODULE SLAVE MODE (CKE = 1) TIMING CHARACTERISTICS



Note: Refer to Figure 21-1 for load conditions.

TABLE 21-31: SPIx MODULE SLAVE MODE (CKE = 1) TIMING REQUIREMENTS

AC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic[1]**	**Min**	**Typ[2]**	**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	—	10	25	ns	See **Note 3**
SP73	TscR	SCKx Input Rise Time	—	10	25	ns	See **Note 3**
SP30	TdoF	SDOx Data Output Fall Time	—	—	—	ns	See parameter D032 and **Note 3**
SP31	TdoR	SDOx Data Output Rise Time	—	—	—	ns	See parameter D031 and **Note 3**
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	10	—	50	ns	See **Note 4**
SP52	TscH2ssH TscL2ssH	$\overline{\text{SSx}}$ ↑ after SCKx Edge	1.5 TCY + 40	—	—	ns	—
SP60	TssL2doV	SDOx Data Output Valid after $\overline{\text{SSx}}$ Edge	—	—	50	ns	—

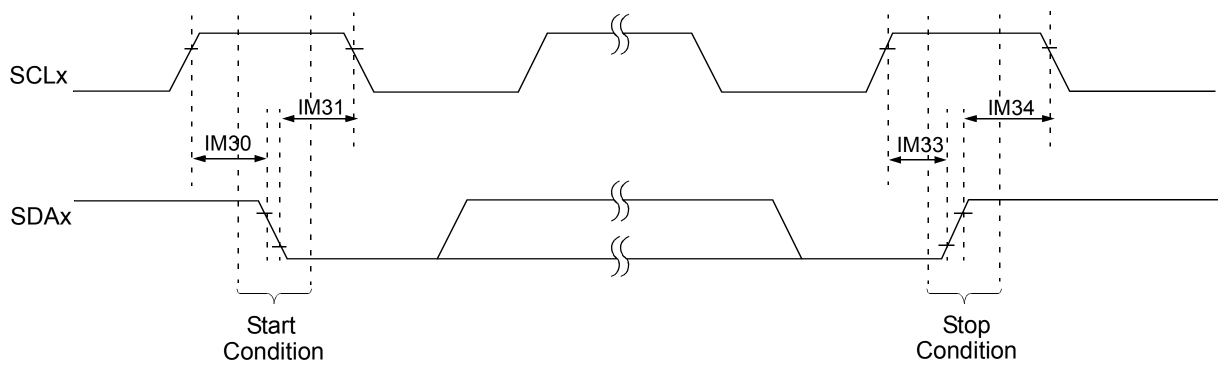
Note 1: These parameters are characterized but not tested in manufacturing.

2: Data in “Typ” column is at 3.3V, 25°C unless otherwise stated.

3: The minimum clock period for SCKx is 100 ns. The clock generated in Master mode must not violate this specification.

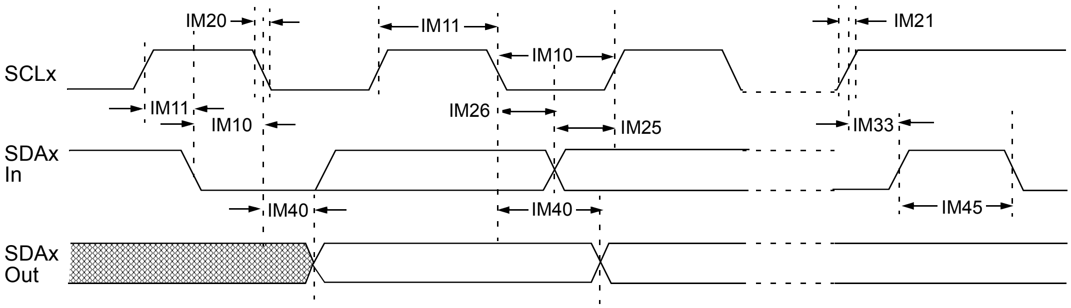
4: Assumes 50 pF load on all SPIx pins.

FIGURE 21-13: I2Cx BUS START/STOP BITS TIMING CHARACTERISTICS (MASTER MODE)



Note: Refer to Figure 21-1 for load conditions.

FIGURE 21-14: I2Cx BUS DATA TIMING CHARACTERISTICS (MASTER MODE)



Note: Refer to Figure 21-1 for load conditions.

TABLE 21-32: 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 for Industrial -40°C ≤ TA ≤ +125°C for Extended			
Param No.	**Symbol**	**Characteristic**		**Min(1)**	**Max**	**Units**	**Conditions**
IM10	TLO:SCL	Clock Low Time	100 kHz mode	TCY/2 (BRG + 1)	—	μs	—
			400 kHz mode	TCY/2 (BRG + 1)	—	μs	—
			1 MHz mode(2)	TCY/2 (BRG + 1)	—	μs	—
IM11	THI:SCL	Clock High Time	100 kHz mode	TCY/2 (BRG + 1)	—	μs	—
			400 kHz mode	TCY/2 (BRG + 1)	—	μs	—
			1 MHz mode(2)	TCY/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 CB	300	ns	
			1 MHz mode(2)	—	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 CB	300	ns	
			1 MHz mode(2)	—	300	ns	
IM25	TSU:DAT	Data Input Setup Time	100 kHz mode	250	—	ns	—
			400 kHz mode	100	—	ns	
			1 MHz mode(2)	40	—	ns	
IM26	THD:DAT	Data Input Hold Time	100 kHz mode	0	—	μs	—
			400 kHz mode	0	0.9	μs	
			1 MHz mode(2)	0.2	—	μs	
IM30	TSU:STA	Start Condition Setup Time	100 kHz mode	TCY/2 (BRG + 1)	—	μs	Only relevant for Repeated Start condition
			400 kHz mode	TCY/2 (BRG + 1)	—	μs	
			1 MHz mode(2)	TCY/2 (BRG + 1)	—	μs	
IM31	THD:STA	Start Condition Hold Time	100 kHz mode	TCY/2 (BRG + 1)	—	μs	After this period the first clock pulse is generated
			400 kHz mode	TCY/2 (BRG + 1)	—	μs	
			1 MHz mode(2)	TCY/2 (BRG + 1)	—	μs	
IM33	TSU:STO	Stop Condition Setup Time	100 kHz mode	TCY/2 (BRG + 1)	—	μs	—
			400 kHz mode	TCY/2 (BRG + 1)	—	μs	
			1 MHz mode(2)	TCY/2 (BRG + 1)	—	μs	
IM34	THD:STO	Stop Condition Hold Time	100 kHz mode	TCY/2 (BRG + 1)	—	ns	—
			400 kHz mode	TCY/2 (BRG + 1)	—	ns	
			1 MHz mode(2)	TCY/2 (BRG + 1)	—	ns	
IM40	TAA:SCL	Output Valid From Clock	100 kHz mode	—	3500	ns	—
			400 kHz mode	—	1000	ns	—
			1 MHz mode(2)	—	400	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(2)	0.5	—	μs	
IM50	CB	Bus Capacitive Loading		—	400	pF	

Note 1: BRG is the value of the I^2C Baud Rate Generator. Refer to **Section 19. "Inter-Integrated Circuit (I2C™)"** in the *"PIC24H Family Reference Manual"*.

2: Maximum pin capacitance = 10 pF for all I2Cx pins (for 1 MHz mode only).

FIGURE 21-15: I2Cx BUS START/STOP BITS TIMING CHARACTERISTICS (SLAVE MODE)

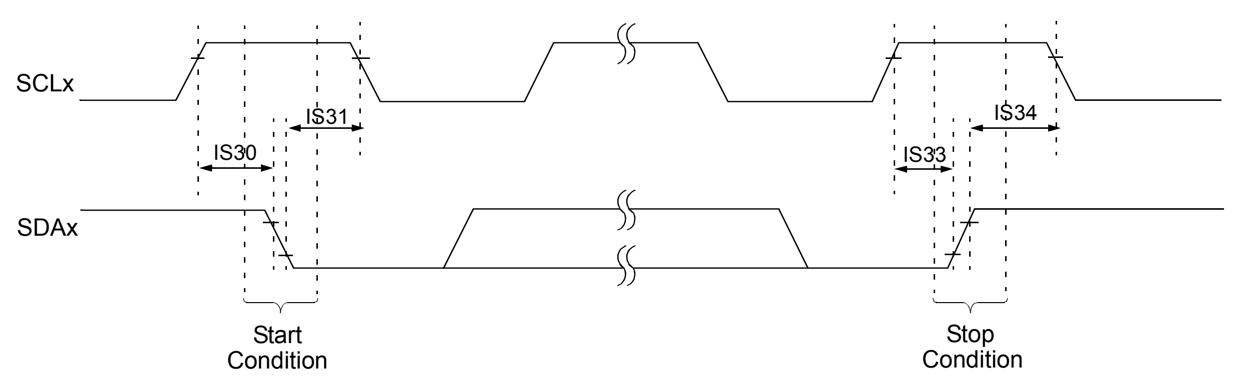


FIGURE 21-16: I2Cx BUS DATA TIMING CHARACTERISTICS (SLAVE MODE)

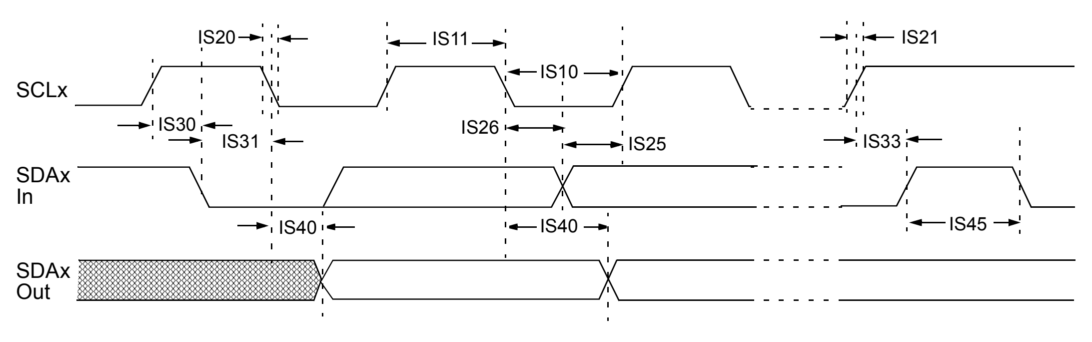


TABLE 21-33: I2Cx BUS DATA TIMING REQUIREMENTS (SLAVE MODE)

AC CHARACTERISTICS				**Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended			
Param	**Symbol**	**Characteristic**		**Min**	**Max**	**Units**	**Conditions**
IS10	TLO:SCL	Clock Low Time	100 kHz mode	4.7	—	μs	Device must operate at a minimum of 1.5 MHz
			400 kHz mode	1.3	—	μs	Device must operate at a minimum of 10 MHz
			1 MHz mode(1)	0.5	—	μs	—
IS11	THI:SCL	Clock High Time	100 kHz mode	4.0	—	μs	Device must operate at a minimum of 1.5 MHz
			400 kHz mode	0.6	—	μs	Device must operate at a minimum of 10 MHz
			1 MHz mode(1)	0.5	—	μs	—
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 CB	300	ns	
			1 MHz mode(1)	—	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 CB	300	ns	
			1 MHz mode(1)	—	300	ns	
IS25	TSU:DAT	Data Input Setup Time	100 kHz mode	250	—	ns	—
			400 kHz mode	100	—	ns	
			1 MHz mode(1)	100	—	ns	
IS26	THD:DAT	Data Input Hold Time	100 kHz mode	0	0	μs	—
			400 kHz mode	0	0.9	μs	
			1 MHz mode(1)	0	0.3	μs	
IS30	TSU:STA	Start Condition Setup Time	100 kHz mode	4.7	—	μs	Only relevant for Repeated Start condition
			400 kHz mode	0.6	—	μs	
			1 MHz mode(1)	0.25	—	μs	
IS31	THD:STA	Start Condition Hold Time	100 kHz mode	4.0	—	μs	After this period, the first clock pulse is generated
			400 kHz mode	0.6	—	μs	
			1 MHz mode(1)	0.25	—	μs	
IS33	TSU:STO	Stop Condition Setup Time	100 kHz mode	4.7	—	μs	—
			400 kHz mode	0.6	—	μs	
			1 MHz mode(1)	0.6	—	μs	
IS34	THD:STO	Stop Condition Hold Time	100 kHz mode	4000	—	ns	—
			400 kHz mode	600	—	ns	
			1 MHz mode(1)	250		ns	
IS40	TAA:SCL	Output Valid From Clock	100 kHz mode	0	3500	ns	—
			400 kHz mode	0	1000	ns	
			1 MHz mode(1)	0	350	ns	
IS45	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(1)	0.5	—	μs	
IS50	CB	Bus Capacitive Loading		—	400	pF	—

Note 1: Maximum pin capacitance = 10 pF for all I2Cx pins (for 1 MHz mode only).

TABLE 21-34: ADC MODULE SPECIFICATIONS

AC CHARACTERISTICS			**Standard Operating Conditions: 3.0V to 3.6V** **(unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic**	**Min.**	**Typ**	**Max.**	**Units**	**Conditions**
Device Supply							
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	—
Reference Inputs							
AD05	VREFH	Reference Voltage High	AVSS + 2.7	—	AVDD	V	See **Note 1**
AD05a			3.0	—	3.6	V	VREFH = AVDD VREFL = AVSS = 0
AD06	VREFL	Reference Voltage Low	AVSS	—	AVDD – 2.7	V	See **Note 1**
AD06a			0	—	0	V	VREFH = AVDD VREFL = AVSS = 0
AD07	VREF	Absolute Reference Voltage	2.7	—	3.6	V	VREF = VREFH - VREFL
AD08	IREF	Current Drain	—	400 —	550 10	µA µA	ADC operating ADC off
Analog Input							
AD12	VINH	Input Voltage Range VINH	VINL	—	VREFH	V	This voltage reflects Sample and Hold Channels 0, 1, 2, and 3 (CH0-CH3), positive input
AD13	VINL	Input Voltage Range VINL	VREFL	—	AVSS + 1V	V	This voltage reflects Sample and Hold Channels 0, 1, 2, and 3 (CH0-CH3), negative input
AD17	RIN	Recommended Impedance of Analog Voltage Source	— —	— —	200 200	Ω Ω	10-bit ADC 12-bit ADC

Note 1: These parameters are not characterized or tested in manufacturing.

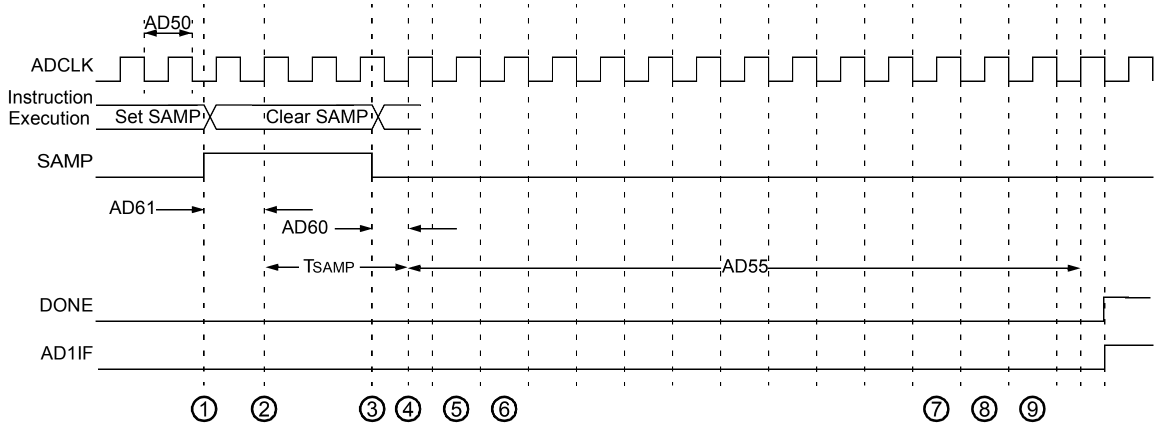
TABLE 21-35: ADC MODULE SPECIFICATIONS (12-BIT MODE)

AC CHARACTERISTICS			**Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic**	**Min.**	**Typ**	**Max.**	**Units**	**Conditions**
ADC Accuracy (12-bit Mode) – Measurements with external VREF+/VREF-							
AD20a	Nr	Resolution	12 data bits			bits	
AD21a	INL	Integral Nonlinearity	-2	—	+2	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3.6V
AD22a	DNL	Differential Nonlinearity	>-1	—	<1	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3.6V
AD23a	GERR	Gain Error	1.25	1.5	3	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3.6V
AD24a	EOFF	Offset Error	1.25	1.52	2	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3.6V
AD25a	—	Monotonicity	—	—	—	—	Guaranteed
ADC Accuracy (12-bit Mode) – Measurements with internal VREF+/VREF-							
AD20a	Nr	Resolution	12 data bits			bits	
AD21a	INL	Integral Nonlinearity	-2	—	+2	LSb	VINL = AVSS = 0V, AVDD = 3.6V
AD22a	DNL	Differential Nonlinearity	>-1	—	<1	LSb	VINL = AVSS = 0V, AVDD = 3.6V
AD23a	GERR	Gain Error	2	3	7	LSb	VINL = AVSS = 0V, AVDD = 3.6V
AD24a	EOFF	Offset Error	2	3	5	LSb	VINL = AVSS = 0V, AVDD = 3.6V
AD25a	—	Monotonicity	—	—	—	—	Guaranteed
Dynamic Performance (12-bit Mode)							
AD30a	THD	Total Harmonic Distortion	-77	-69	-61	dB	—
AD31a	SINAD	Signal to Noise and Distortion	59	63	64	dB	—
AD32a	SFDR	Spurious Free Dynamic Range	63	72	74	dB	—
AD33a	FNYQ	Input Signal Bandwidth	—	—	250	kHz	—
AD34a	ENOB	Effective Number of Bits	10.95	11.1	—	bits	—

TABLE 21-36: ADC MODULE SPECIFICATIONS (10-BIT MODE)

AC CHARACTERISTICS			**Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic**	**Min.**	**Typ**	**Max.**	**Units**	**Conditions**
ADC Accuracy (10-bit Mode) – Measurements with external VREF+/VREF-							
AD20b	Nr	Resolution	10 data bits			bits	
AD21b	INL	Integral Nonlinearity	-1.5	—	+1.5	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3.6V
AD22b	DNL	Differential Nonlinearity	>-1	—	<1	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3.6V
AD23b	GERR	Gain Error	1	3	6	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3.6V
AD24b	EOFF	Offset Error	1	2	5	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3.6V
AD25b	—	Monotonicity	—	—	—	—	Guaranteed
ADC Accuracy (10-bit Mode) – Measurements with internal VREF+/VREF-							
AD20b	Nr	Resolution	10 data bits			bits	
AD21b	INL	Integral Nonlinearity	-1	—	+1	LSb	VINL = AVSS = 0V, AVDD = 3.6V
AD22b	DNL	Differential Nonlinearity	>-1	—	<1	LSb	VINL = AVSS = 0V, AVDD = 3.6V
AD23b	GERR	Gain Error	1	5	6	LSb	VINL = AVSS = 0V, AVDD = 3.6V
AD24b	EOFF	Offset Error	1	2	3	LSb	VINL = AVSS = 0V, AVDD = 3.6V
AD25b	—	Monotonicity	—	—	—	—	Guaranteed
Dynamic Performance (10-bit Mode)							
AD30b	THD	Total Harmonic Distortion	—	-64	-67	dB	—
AD31b	SINAD	Signal to Noise and Distortion	—	57	58	dB	—
AD32b	SFDR	Spurious Free Dynamic Range	—	60	62	dB	—
AD33b	FNYQ	Input Signal Bandwidth	—	—	550	kHz	—
AD34b	ENOB	Effective Number of Bits	9.1	9.7	9.8	bits	—

FIGURE 21-17: ADC CONVERSION (12-BIT MODE) TIMING CHARACTERISTICS (ASAM = `0`, SSRC<2:0> = `000`)



① – Software sets AD1CON. SAMP to start sampling.

② – Sampling starts after discharge period. TSAMP is described in **Section 28. “10/12-bit ADC without DMA”** in the *“PIC24H Family Reference Manual”*.

③ – Software clears AD1CON. SAMP to start conversion.

④ – Sampling ends, conversion sequence starts.

⑤ – Convert bit `11`.

⑥ – Convert bit `10`.

⑦ – Convert bit `1`.

⑧ – Convert bit `0`.

⑨ – One TAD for end of conversion.

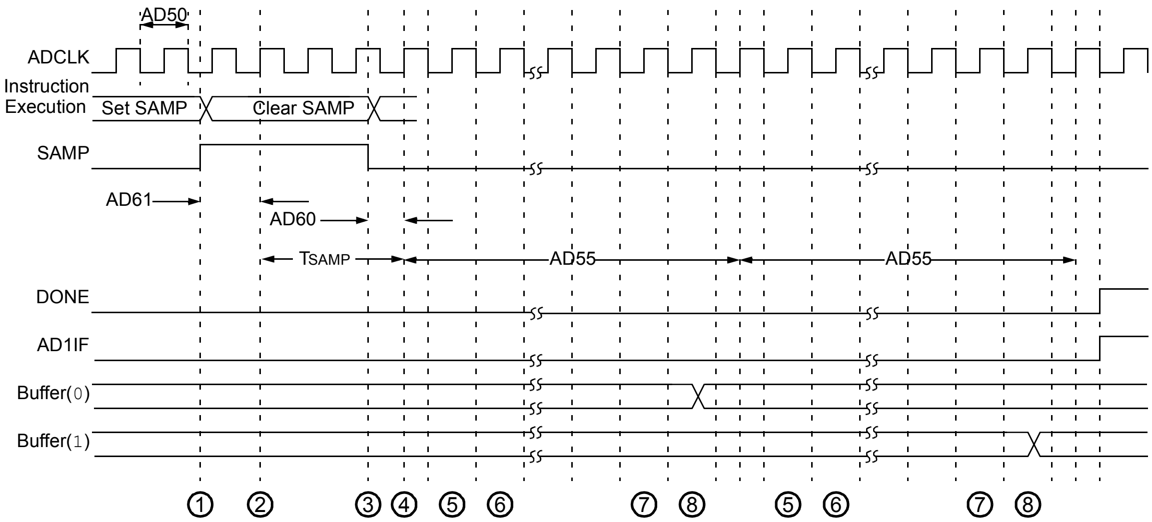
TABLE 21-37: ADC CONVERSION (12-BIT MODE) TIMING REQUIREMENTS

AC CHARACTERISTICS			**Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic**	**Min.**	**Typ**	**Max.**	**Units**	**Conditions**
Clock Parameters							
AD50	TAD	ADC Clock Period	117.6	—	—	ns	
AD51	tRC	ADC Internal RC Oscillator Period	—	250	—	ns	
Conversion Rate							
AD55	tCONV	Conversion Time	—	14 TAD		ns	
AD56	FCNV	Throughput Rate	—	—	500	Ksps	
AD57	TSAMP	Sample Time	3 TAD	—	—	—	
Timing Parameters							
AD60	tPCS	Conversion Start from Sample Trigger[(2)]	—	1.0 TAD	—	—	Auto Convert Trigger not selected
AD61	tPSS	Sample Start from Setting Sample (SAMP) bit[(2)]	0.5 TAD	—	1.5 TAD	—	—
AD62	tCSS	Conversion Completion to Sample Start (ASAM = `1`)[(2)]	—	0.5 TAD	—	—	—
AD63	tDPU	Time to Stabilize Analog Stage from ADC Off to ADC On[(2)]	1	—	5	µs	—

Note 1: Because the sample caps will eventually lose charge, clock rates below 10 kHz can affect linearity performance, especially at elevated temperatures.

2: These parameters are characterized but not tested in manufacturing.

FIGURE 21-18: ADC CONVERSION (10-BIT MODE) TIMING CHARACTERISTICS (CHPS<1:0> = 01, SIMSAM = 0, ASAM = 0, SSRC<2:0> = 000)



① – Software sets AD1CON. SAMP to start sampling.

② – Sampling starts after discharge period. TSAMP is described in **Section 28. “10/12-bit ADC without DMA”** in the *“PIC24H Family Reference Manual”*.

③ – Software clears AD1CON. SAMP to start conversion.

④ – Sampling ends, conversion sequence starts.

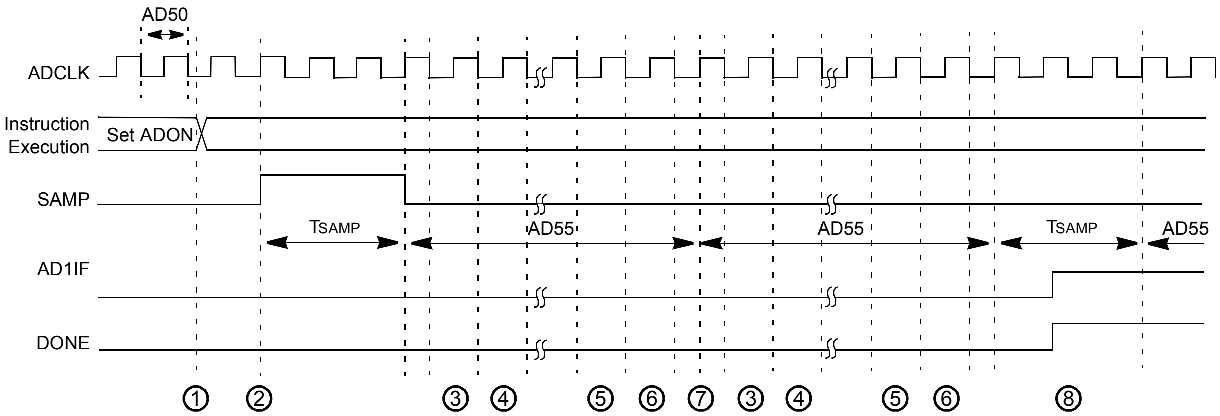
⑤ – Convert bit 9.

⑥ – Convert bit 8.

⑦ – Convert bit 0.

⑧ – One TAD for end of conversion.

FIGURE 21-19: ADC CONVERSION (10-BIT MODE) TIMING CHARACTERISTICS (CHPS<1:0> = 01, SIMSAM = 0, ASAM = 1, SSRC<2:0> = 111, SAMC<4:0> = 00001)



① – Software sets AD1CON. ADON to start AD operation.

② – Sampling starts after discharge period. TSAMP is described in **Section 28. “10/12-bit ADC without DMA”** in the *“PIC24H Family Reference Manual”*.

③ – Convert bit 9.

④ – Convert bit 8.

⑤ – Convert bit 0.

⑥ – One TAD for end of conversion.

⑦ – Begin conversion of next channel.

⑧ – Sample for time specified by SAMC<4:0>.

TABLE 21-38: ADC CONVERSION (10-BIT MODE) TIMING REQUIREMENTS

AC CHARACTERISTICS			**Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	**Symbol**	**Characteristic**	**Min.**	**Typ(1)**	**Max.**	**Units**	**Conditions**
Clock Parameters							
AD50	TAD	ADC Clock Period	76	—	—	ns	
AD51	tRC	ADC Internal RC Oscillator Period	—	250	—	ns	
Conversion Rate							
AD55	tCONV	Conversion Time	—	12 TAD	—	—	
AD56	FCNV	Throughput Rate	—	—	1.1	Msps	
AD57	TSAMP	Sample Time	2 TAD	—	—	—	
Timing Parameters							
AD60	tPCS	Conversion Start from Sample Trigger(1)	—	1.0 TAD	—	—	Auto-Convert Trigger not selected
AD61	tPSS	Sample Start from Setting Sample (SAMP) bit(1)	0.5 TAD	—	1.5 TAD	—	—
AD62	tCSS	Conversion Completion to Sample Start (ASAM = `1`)(1)	—	0.5 TAD	—	—	—
AD63	tDPU	Time to Stabilize Analog Stage from ADC Off to ADC On(1)	1	—	5	µs	—

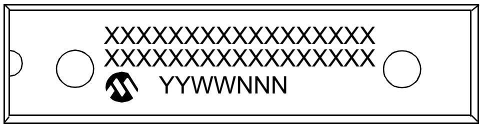
Note 1: These parameters are characterized but not tested in manufacturing.

2: Because the sample caps will eventually lose charge, clock rates below 10 kHz can affect linearity performance, especially at elevated temperatures.

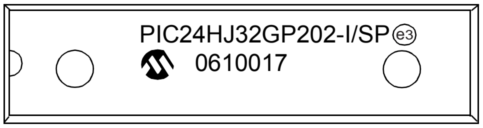
22.0 PACKAGING INFORMATION

22.1 Package Marking Information

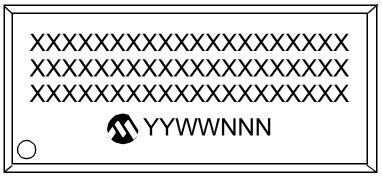
28-Lead SPDIP



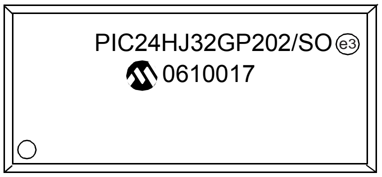
Example



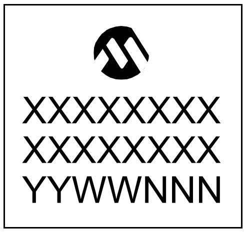
28-Lead SOIC (.300”)



Example

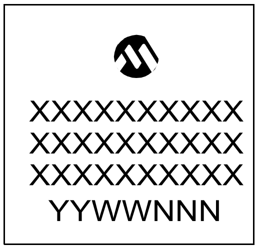


28-Lead QFN-S



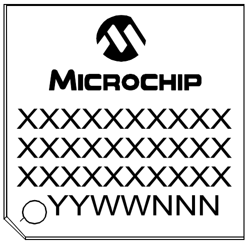
Example

44-Lead QFN

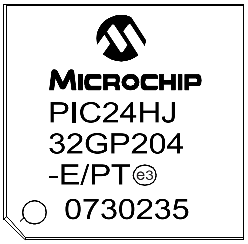


Example

44-Lead TQFP



Example



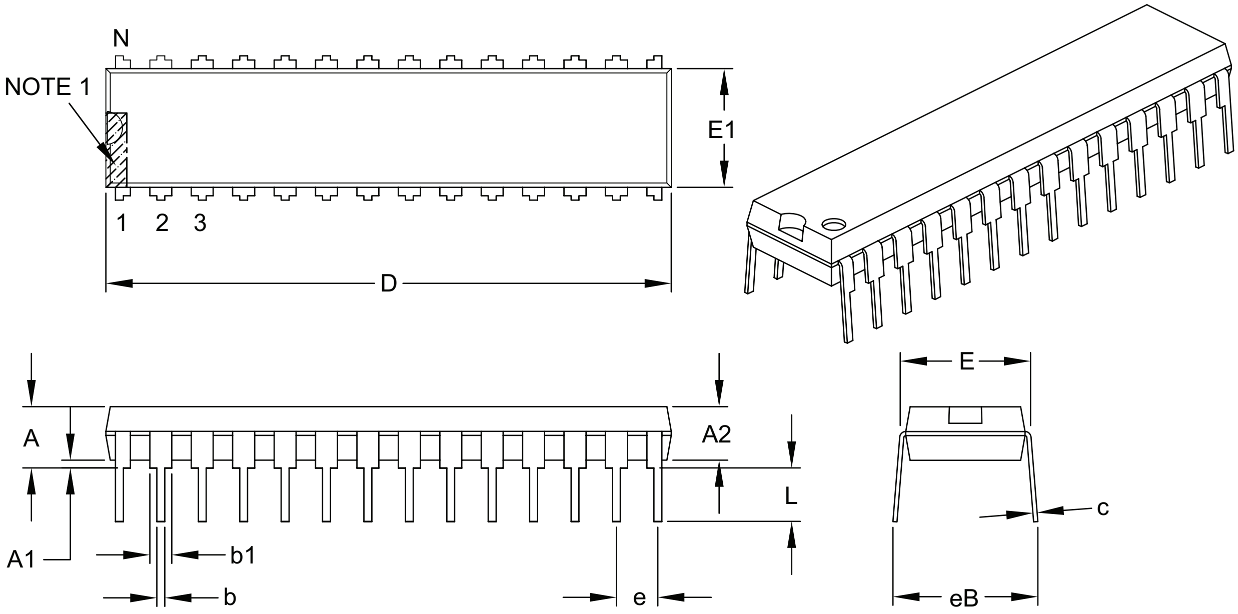
Legend:	XX...X	Customer-specific information
	Y	Year code (last digit of calendar year)
	YY	Year code (last 2 digits of calendar year)
	WW	Week code (week of January 1 is week ‘01’)
	NNN	Alphanumeric traceability code
	(e3)	Pb-free JEDEC designator for Matte Tin (Sn)
	*	This package is Pb-free. The Pb-free JEDEC designator ((e3)) can be found on the outer packaging for this package.

Note: If the full Microchip part number cannot be marked on one line, it is carried over to the next line, thus limiting the number of available characters for customer-specific information.

22.2 Package Details

28-Lead Skinny Plastic Dual In-Line (SP) – 300 mil Body [SPDIP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



	Units	INCHES		
	Dimension Limits	MIN	NOM	MAX
Number of Pins	N	28		
Pitch	e	.100 BSC		
Top to Seating Plane	A	–	–	.200
Molded Package Thickness	A2	.120	.135	.150
Base to Seating Plane	A1	.015	–	–
Shoulder to Shoulder Width	E	.290	.310	.335
Molded Package Width	E1	.240	.285	.295
Overall Length	D	1.345	1.365	1.400
Tip to Seating Plane	L	.110	.130	.150
Lead Thickness	c	.008	.010	.015
Upper Lead Width	b1	.040	.050	.070
Lower Lead Width	b	.014	.018	.022
Overall Row Spacing §	eB	–	–	.430

Notes:

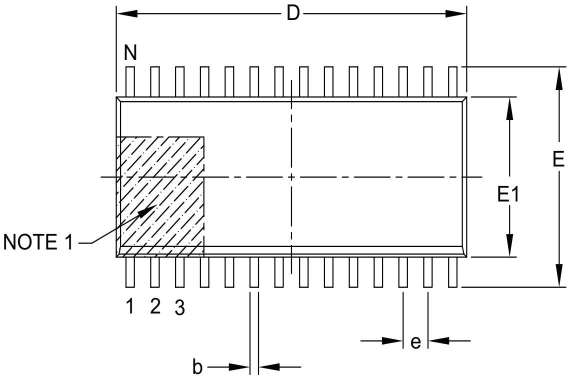
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

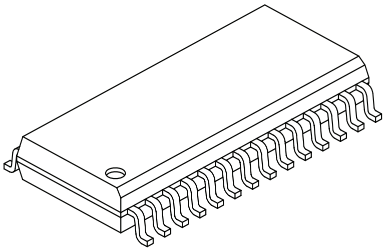
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

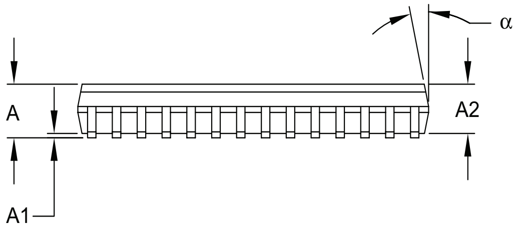
Microchip Technology Drawing C04-070B

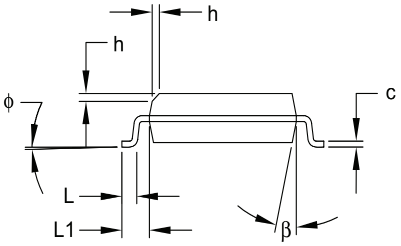
28-Lead Plastic Small Outline (SO) – Wide, 7.50 mm Body [SOIC]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging









Units		MILLMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N	28		
Pitch	e	1.27 BSC		
Overall Height	A	–	–	2.65
Molded Package Thickness	A2	2.05	–	–
Standoff §	A1	0.10	–	0.30
Overall Width	E	10.30 BSC		
Molded Package Width	E1	7.50 BSC		
Overall Length	D	17.90 BSC		
Chamfer (optional)	h	0.25	–	0.75
Foot Length	L	0.40	–	1.27
Footprint	L1	1.40 REF		
Foot Angle Top	φ	0°	–	8°
Lead Thickness	c	0.18	–	0.33
Lead Width	b	0.31	–	0.51
Mold Draft Angle Top	α	5°	–	15°
Mold Draft Angle Bottom	β	5°	–	15°

Notes:

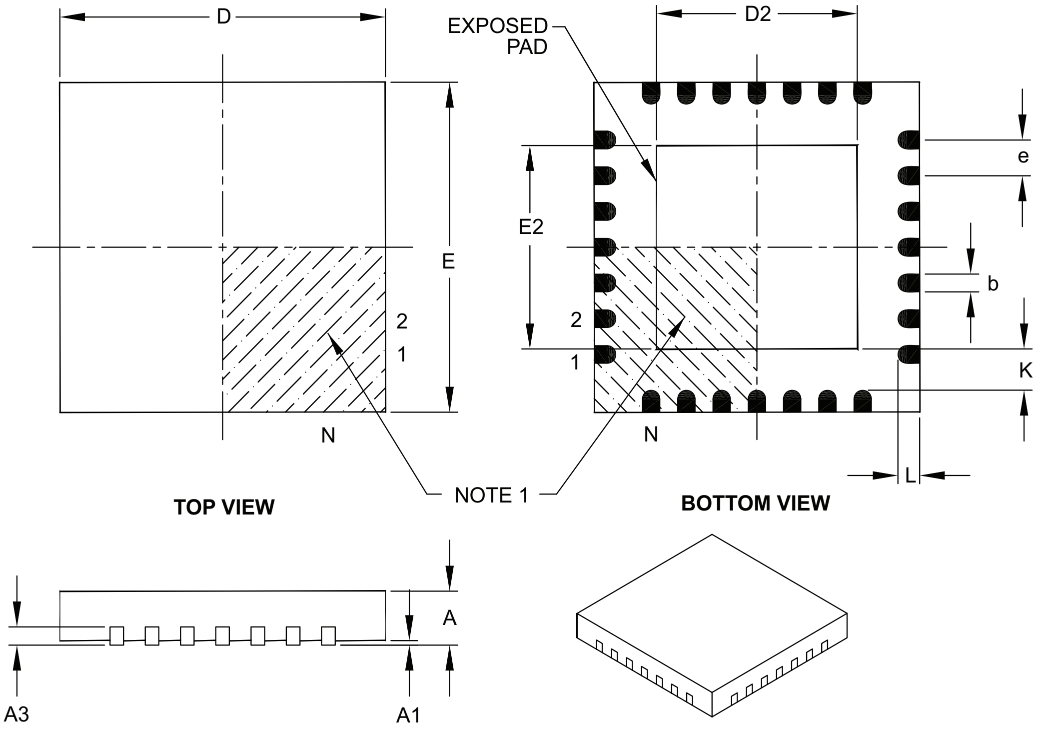
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.15 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-052B

28-Lead Plastic Quad Flat, No Lead Package (MM) – 6x6x0.9 mm Body [QFN-S] with 0.40 mm Contact Length

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N	28		
Pitch	e	0.65 BSC		
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Contact Thickness	A3	0.20 REF		
Overall Width	E	6.00 BSC		
Exposed Pad Width	E2	3.65	3.70	4.70
Overall Length	D	6.00 BSC		
Exposed Pad Length	D2	3.65	3.70	4.70
Contact Width	b	0.23	0.38	0.43
Contact Length	L	0.30	0.40	0.50
Contact-to-Exposed Pad	K	0.20	–	–

Notes:

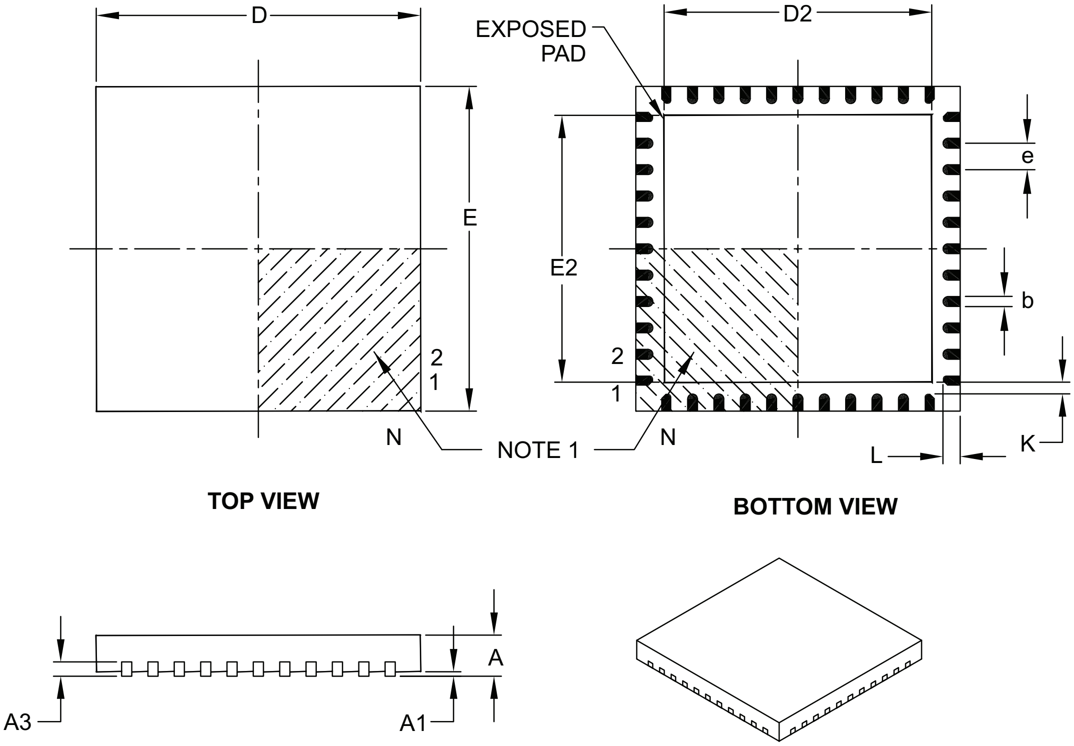
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated.
3. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-124B

44-Lead Plastic Quad Flat, No Lead Package (ML) – 8x8 mm Body [QFN]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



TOP VIEW

BOTTOM VIEW

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		44	
Pitch	e		0.65 BSC	
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Contact Thickness	A3		0.20 REF	
Overall Width	E		8.00 BSC	
Exposed Pad Width	E2	6.30	6.45	6.80
Overall Length	D		8.00 BSC	
Exposed Pad Length	D2	6.30	6.45	6.80
Contact Width	b	0.25	0.30	0.38
Contact Length	L	0.30	0.40	0.50
Contact-to-Exposed Pad	K	0.20	–	–

Notes:

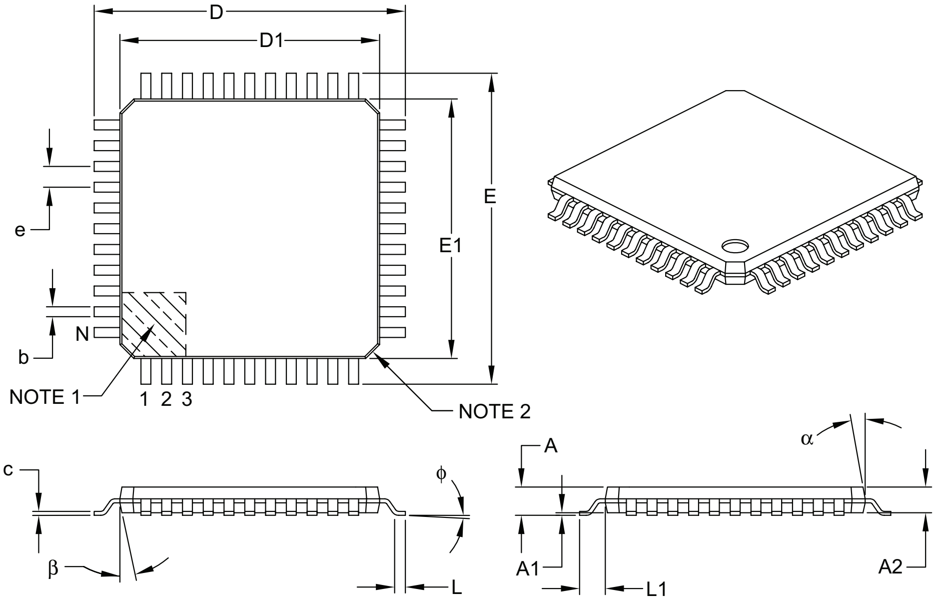
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated.
3. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-103B

44-Lead Plastic Thin Quad Flatpack (PT) – 10x10x1 mm Body, 2.00 mm Footprint [TQFP]

Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Leads	N	44		
Lead Pitch	e	0.80 BSC		
Overall Height	A	–	–	1.20
Molded Package Thickness	A2	0.95	1.00	1.05
Standoff	A1	0.05	–	0.15
Foot Length	L	0.45	0.60	0.75
Footprint	L1	1.00 REF		
Foot Angle	ϕ	0°	3.5°	7°
Overall Width	E	12.00 BSC		
Overall Length	D	12.00 BSC		
Molded Package Width	E1	10.00 BSC		
Molded Package Length	D1	10.00 BSC		
Lead Thickness	c	0.09	–	0.20
Lead Width	b	0.30	0.37	0.45
Mold Draft Angle Top	α	11°	12°	13°
Mold Draft Angle Bottom	β	11°	12°	13°

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Chamfers at corners are optional; size may vary.
3. Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-076B

APPENDIX A: REVISION HISTORY

Revision A (August 2007)

Initial release of this document

NOTES:

INDEX

NOTES:

THE MICROCHIP WEB SITE

Microchip provides online support via our WWW site at www.microchip.com. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at www.microchip.com, 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

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: PIC24HJ32GP202/204 and PIC24HJ16GP304 Literature Number: DS70289A

Questions:

1. What are the best features of this document?

____

____

2. How does this document meet your hardware and software development needs?

____

____

3. Do you find the organization of this document easy to follow? If not, why?

____

____

4. What additions to the document do you think would enhance the structure and subject?

____

____

5. What deletions from the document could be made without affecting the overall usefulness?

____

____

6. Is there any incorrect or misleading information (what and where)?

____

____

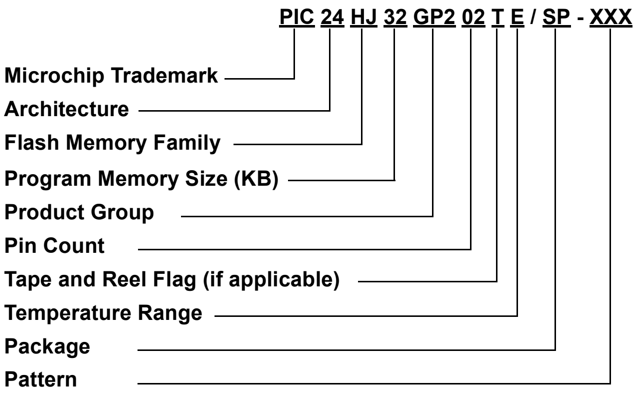
7. How would you improve this document?

____

____

PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.



Architecture: 24 = 16-bit Microcontroller

Flash Memory Family: HJ = Flash program memory, 3.3V

Product Group:
GP2 = General purpose family
GP3 = General purpose family

Pin Count:
02 = 28-pin
03 = 44-pin

Temperature Range:
I = -40°C to +85°C (Industrial)
E = -40°C to +125°C (Extended)

Package:
SP = Skinny Plastic Dual In-Line - 300 mil body (SPDIP)
SO = Plastic Small Outline - Wide - 7.5 mm body (SOIC)
MM = Plastic Quad, No Lead Package - 6x6 mm body (QFN-S)
PT = Plastic Thin Quad Flatpack - 10x10x1 mm body (TQFP
ML = Plastic Quad, No Lead Package - 8x8 mm body (QFN)

Examples:

a) PIC24HJ32GP202-E/SP:
General-purpose PIC24H, 32 KB program memory, 28-pin, Extended temp., SPDIP package.

WORLDWIDE SALES AND SERVICE

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
http://support.microchip.com
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Farmington Hills, MI
Tel: 248-538-2250
Fax: 248-538-2260

Kokomo
Kokomo, IN
Tel: 765-864-8360
Fax: 765-864-8387

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

Santa Clara
Santa Clara, CA
Tel: 408-961-6444
Fax: 408-961-6445

Toronto
Mississauga, Ontario,
Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8528-2100
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

China - 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: 86-29-8833-7252
Fax: 86-29-8833-7256

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-4182-8400
Fax: 91-80-4182-8422

India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

India - Pune
Tel: 91-20-2566-1512
Fax: 91-20-2566-1513

Japan - Yokohama
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Malaysia - 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: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

UK - Wokingham
Tel: 44-118-921-5869
Fax: 44-118-921-5820

06/25/07