



Welcome to [E-XFL.COM](#)

What is "[Embedded - Microcontrollers](#)"?

"[Embedded - Microcontrollers](#)" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "[Embedded - Microcontrollers](#)"

Details

Product Status	Active
Core Processor	PIC
Core Size	16-Bit
Speed	40 MIPS
Connectivity	CANbus, I ² C, IrDA, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	53
Program Memory Size	64KB (22K x 24)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	8K x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 3.6V
Data Converters	A/D 18x10b/12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-TQFP
Supplier Device Package	64-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic24hj64gp506-i-pt

3.0 CPU

Note: This data sheet summarizes the features of the PIC24HJXXXGPX06/X08/X10 family of devices. However, 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”, **Section 2. “CPU”** (DS70245), which is available from the Microchip website (www.microchip.com).

The PIC24HJXXXGPX06/X08/X10 CPU module has a 16-bit (data) modified Harvard architecture with an enhanced instruction set and addressing modes. The CPU has a 24-bit instruction word with a variable length opcode field. The Program Counter (PC) is 23 bits wide and addresses up to 4M x 24 bits of user program memory space. The actual amount of program memory implemented varies by device. A single-cycle instruction prefetch mechanism is used to help maintain throughput and provides predictable execution. All instructions execute in a single cycle, with the exception of instructions that change the program flow, the double word move (MOV.D) instruction and the table instructions. Overhead-free, single-cycle program loop constructs are supported using the REPEAT instruction, which is interruptible at any point.

The PIC24HJXXXGPX06/X08/X10 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 PIC24HJXXXGPX06/X08/X10 instruction set includes many addressing modes and is designed for optimum C compiler efficiency. For most instructions, the PIC24HJXXXGPX06/X08/X10 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 3-1, and the programmer's model for the PIC24HJXXXGPX06/X08/X10 is shown in Figure 3-2.

3.1 Data Addressing Overview

The data space can be linearly addressed as 32K words or 64 Kbytes using an Address Generation Unit (AGU). The upper 32 Kbytes of the data space memory map can optionally be mapped into program space at any 16K program word boundary defined by the 8-bit Program Space Visibility Page (PSVPAG) register. The program to data space mapping feature lets any instruction access program space as if it were data space.

The data space also includes 2 Kbytes of DMA RAM, which is primarily used for DMA data transfers, but may be used as general purpose RAM.

3.2 Special MCU Features

The PIC24HJXXXGPX06/X08/X10 features a 17-bit by 17-bit, single-cycle multiplier. The multiplier can perform signed, unsigned and mixed-sign multiplication. Using a 17-bit by 17-bit multiplier for 16-bit by 16-bit multiplication makes mixed-sign multiplication possible.

The PIC24HJXXXGPX06/X08/X10 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.

PIC24HJXXXGPX06/X08/X10

4.1.1 PROGRAM MEMORY ORGANIZATION

The program memory space is organized in word-addressable blocks. Although it is treated as 24 bits wide, it is more appropriate to think of each address of the program memory as a lower and upper word, with the upper byte of the upper word being unimplemented. The lower word always has an even address, while the upper word has an odd address (Figure 4-2).

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

4.1.2 INTERRUPT AND TRAP VECTORS

All PIC24HJXXXGPX06/X08/X10 devices reserve the addresses between 0x00000 and 0x000200 for hard-coded program execution vectors. A hardware Reset vector is provided to redirect code execution from the default value of the PC on device Reset to the actual start of code. A GOTO instruction is programmed by the user at 0x000000, with the actual address for the start of code at 0x000002.

PIC24HJXXXGPX06/X08/X10 devices also have two interrupt vector tables, located from 0x000004 to 0x0000FF and 0x000100 to 0x0001FF. These vector tables allow each of the many device interrupt sources to be handled by separate Interrupt Service Routines (ISRs). A more detailed discussion of the interrupt vector tables is provided in **Section 7.1 “Interrupt Vector Table”**.

FIGURE 4-2: PROGRAM MEMORY ORGANIZATION

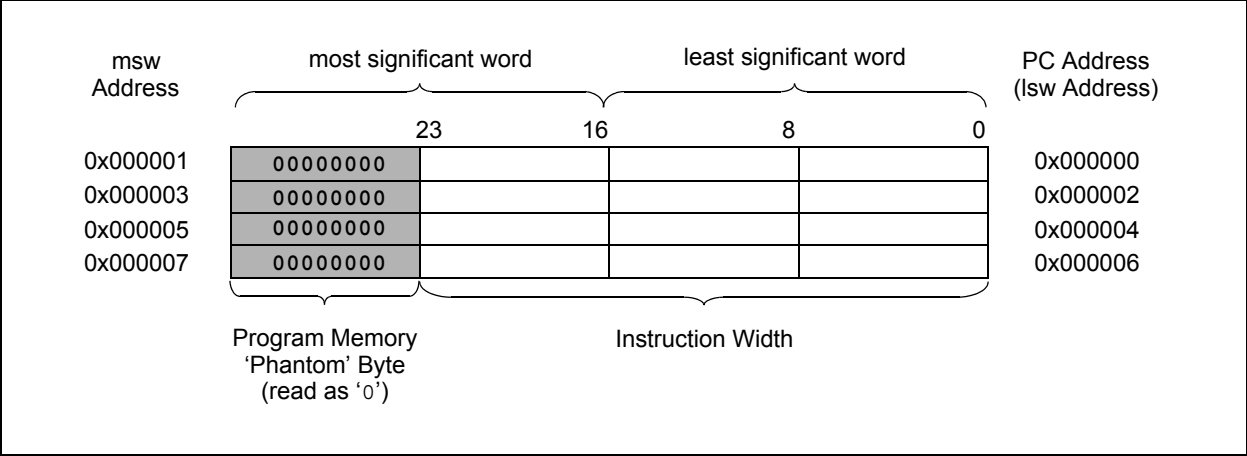


TABLE 4-21: ECAN2 REGISTER MAP WHEN C2CTRL1.WIN = 0 OR 1 FOR PIC24HJ256GP610 DEVICES ONLY

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
C2CTRL1	0500	—	—	CSIDL	ABAT	—	REQOP<2:0>			OPMODE<2:0>			—	CANCAP	—	—	WIN	0480
C2CTRL2	0502	—	—	—	—	—	—	—	—	—	—	—	DNCNT<4:0>					0000
C2VEC	0504	—	—	—	FILHIT<4:0>					—	ICODE<6:0>							0000
C2FCTRL	0506	DMABS<2:0>			—	—	—	—	—	—	—	—	FSA<4:0>					0000
C2FIFO	0508	—	—	FBP<5:0>					—	—	FNRB<5:0>							0000
C2INTF	050A	—	—	TXBO	TXBP	RXBP	TXWAR	RXWAR	EWARN	IVRIF	WAKIF	ERRIF	—	FIFOIF	RBOVIF	RBIF	TBIF	0000
C2INTE	050C	—	—	—	—	—	—	—	—	IVRIE	WAKIE	ERRIE	—	FIFOIE	RBOVIE	RBIE	TBIE	0000
C2EC	050E	TERRCNT<7:0>								RERRCNT<7:0>								0000
C2CFG1	0510	—	—	—	—	—	—	—	—	SJW<1:0>		BRP<5:0>						0000
C2CFG2	0512	—	WAKFIL	—	—	—	SEG2PH<2:0>			SEG2PHTS	SAM	SEG1PH<2:0>			PRSEG<2:0>			0000
C2FEN1	0514	FLTEN15	FLTEN14	FLTEN13	FLTEN12	FLTEN11	FLTEN10	FLTEN9	FLTEN8	FLTEN7	FLTEN6	FLTEN5	FLTEN4	FLTEN3	FLTEN2	FLTEN1	FLTEN0	FFFF
C2FMSKSEL1	0518	F7MSK<1:0>		F6MSK<1:0>		F5MSK<1:0>		F4MSK<1:0>		F3MSK<1:0>		F2MSK<1:0>		F1MSK<1:0>		F0MSK<1:0>		0000
C2FMSKSEL2	051A	F15MSK<1:0>		F14MSK<1:0>		F13MSK<1:0>		F12MSK<1:0>		F11MSK<1:0>		F10MSK<1:0>		F9MSK<1:0>		F8MSK<1:0>		0000

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

TABLE 4-22: ECAN2 REGISTER MAP WHEN C2CTRL1.WIN = 0 FOR PIC24HJ256GP610 DEVICES ONLY

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
	0500-051E	See definition when WIN = x																
C2RXFUL1	0520	RXFUL15	RXFUL14	RXFUL13	RXFUL12	RXFUL11	RXFUL10	RXFUL9	RXFUL8	RXFUL7	RXFUL6	RXFUL5	RXFUL4	RXFUL3	RXFUL2	RXFUL1	RXFUL0	0000
C2RXFUL2	0522	RXFUL31	RXFUL30	RXFUL29	RXFUL28	RXFUL27	RXFUL26	RXFUL25	RXFUL24	RXFUL23	RXFUL22	RXFUL21	RXFUL20	RXFUL19	RXFUL18	RXFUL17	RXFUL16	0000
C2RXOVF1	0528	RXOVF15	RXOVF14	RXOVF13	RXOVF12	RXOVF11	RXOVF10	RXOVF09	RXOVF08	RXOVF7	RXOVF6	RXOVF5	RXOVF4	RXOVF3	RXOVF2	RXOVF1	RXOVF0	0000
C2RXOVF2	052A	RXOVF31	RXOVF30	RXOVF29	RXOVF28	RXOVF27	RXOVF26	RXOVF25	RXOVF24	RXOVF23	RXOVF22	RXOVF21	RXOVF20	RXOVF19	RXOVF18	RXOVF17	RXOVF16	0000
C2TR01CON	0530	TXEN1	TX ABAT1	TX LARB1	TX ERR1	TX REQ1	RTREN1	TX1PRI<1:0>		TXEN0	TX ABAT0	TX LARB0	TX ERR0	TX REQ0	RTREN0	TX0PRI<1:0>		0000
C2TR23CON	0532	TXEN3	TX ABAT3	TX LARB3	TX ERR3	TX REQ3	RTREN3	TX3PRI<1:0>		TXEN2	TX ABAT2	TX LARB2	TX ERR2	TX REQ2	RTREN2	TX2PRI<1:0>		0000
C2TR45CON	0534	TXEN5	TX ABAT5	TX LARB5	TX ERR5	TX REQ5	RTREN5	TX5PRI<1:0>		TXEN4	TX ABAT4	TX LARB4	TX ERR4	TX REQ4	RTREN4	TX4PRI<1:0>		0000
C2TR67CON	0536	TXEN7	TX ABAT7	TX LARB7	TX ERR7	TX REQ7	RTREN7	TX7PRI<1:0>		TXEN6	TX ABAT6	TX LARB6	TX ERR6	TX REQ6	RTREN6	TX6PRI<1:0>		xxxx
C2RXD	0540	Recieved Data Word																xxxx
C2TXD	0542	Transmit Data Word																xxxx

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

TABLE 4-24: PORTA REGISTER MAP⁽¹⁾

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
TRISA	02C0	TRISA15	TRISA14	TRISA13	TRISA12	—	TRISA10	TRISA9	—	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	F6FF
PORTA	02C2	RA15	RA14	RA13	RA12	—	RA10	RA9	—	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0	xxxx
LATA	02C4	LATA15	LATA14	LATA13	LATA12	—	LATA10	LATA9	—	LATA7	LATA6	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	xxxx
ODCA	06C0	ODCA15	ODCA14	—	—	—	—	—	—	—	—	ODCA5	ODCA4	ODCA3	ODCA2	ODCA1	ODCA0	0000

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

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

TABLE 4-25: PORTB REGISTER MAP⁽¹⁾

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

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

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

TABLE 4-26: PORTC REGISTER MAP⁽¹⁾

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
TRISC	02CC	TRISC15	TRISC14	TRISC13	TRISC12	—	—	—	—	—	—	—	TRISC4	TRISC3	TRISC2	TRISC1	—	F01E
PORTC	02CE	RC15	RC14	RC13	RC12	—	—	—	—	—	—	—	RC4	RC3	RC2	RC1	—	xxxx
LATC	02D0	LATC15	LATC14	LATC13	LATC12	—	—	—	—	—	—	—	LATC4	LATC3	LATC2	LATC1	—	xxxx

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

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

TABLE 4-27: PORTD REGISTER MAP⁽¹⁾

File Name	Addr	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
TRISD	02D2	TRISD15	TRISD14	TRISD13	TRISD12	TRISD11	TRISD10	TRISD9	TRISD8	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	FFFF
PORTD	02D4	RD15	RD14	RD13	RD12	RD11	RD10	RD9	RD8	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	xxxx
LATD	02D6	LATD15	LATD14	LATD13	LATD12	LATD11	LATD10	LATD9	LATD8	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0	xxxx
ODCD	06D2	ODCD15	ODCD14	ODCD13	ODCD12	ODCD11	ODCD10	ODCD9	ODCD8	ODCD7	ODCD6	ODCD5	ODCD4	ODCD3	ODCD2	ODCD1	ODCD0	0000

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

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

5.0 FLASH PROGRAM MEMORY

Note: This data sheet summarizes the features of the PIC24HJXXXGPX06/X08/X10 family of devices. However, 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”, **Section 5. “Flash Programming”** (DS70228), which is available from the Microchip website (www.microchip.com).

The PIC24HJXXXGPX06/X08/X10 devices contain internal Flash program memory for storing and executing application code. The memory is readable, writable and erasable during normal operation over the entire VDD range.

Flash memory can be programmed in two ways:

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

ICSP programming capability allows a PIC24HJXXXGPX06/X08/X10 device to be serially programmed while in the end application circuit. This is simply done with two lines for programming clock and programming data (one of the alternate programming pin pairs: PGECx/PGEDx, and three other lines for power (VDD), ground (VSS) and Master Clear (MCLR). This allows customers to manufacture boards with unprogrammed devices and then program the digital

signal controller just before shipping the product. This also allows the most recent firmware or a custom firmware to be programmed.

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

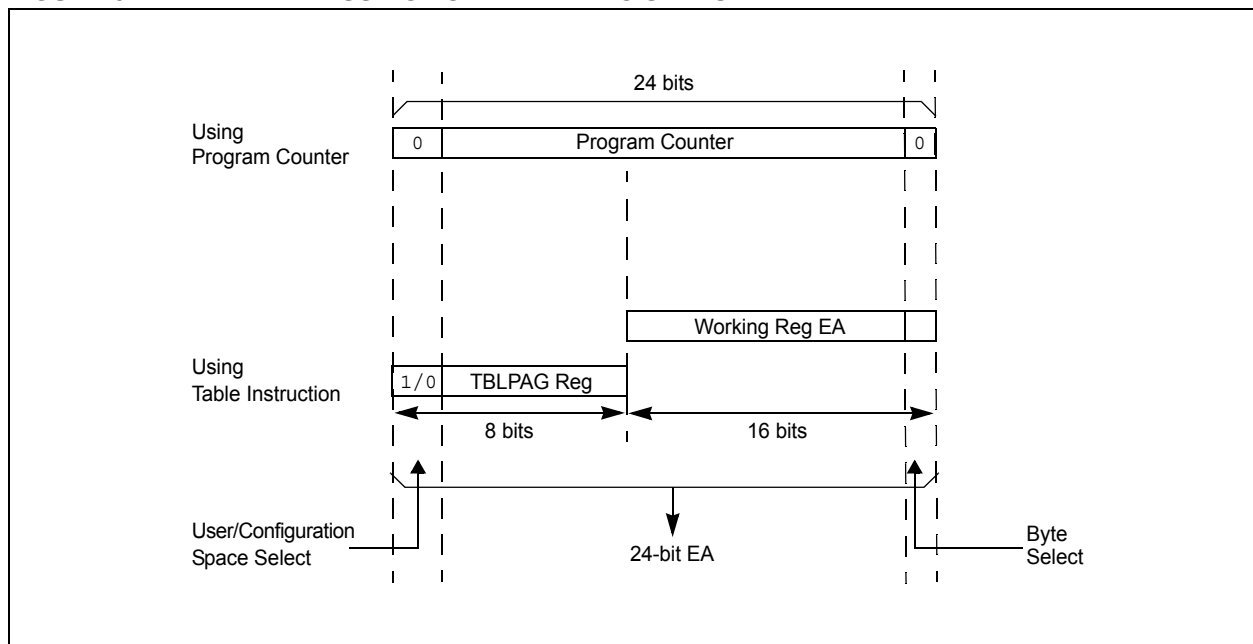
5.1 Table Instructions and Flash Programming

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 5-1.

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

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

FIGURE 5-1: ADDRESSING FOR TABLE REGISTERS



5.2 RTSP Operation

The PIC24HJXXXGPX06/X08/X10 Flash program memory array is organized into rows of 64 instructions or 192 bytes. RTSP allows the user to erase a page of memory, which consists of eight rows (512 instructions) at a time, and to program one row or one word at a time. Table 24-12 displays typical erase and programming times. The 8-row erase pages and single row write rows are edge-aligned, from the beginning of program memory, on boundaries of 1536 bytes and 192 bytes, respectively.

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

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

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

5.3 Programming Operations

A complete programming sequence is necessary for programming or erasing the internal Flash in RTSP mode. The processor stalls (waits) until the programming operation is finished.

The programming time depends on the FRC accuracy (see Table 24-19) and the value of the FRC Oscillator Tuning register (see Register 9-4). Use the following formula to calculate the minimum and maximum values for the Row Write Time, Page Erase Time and Word Write Cycle Time parameters (see Table 24-12).

EQUATION 5-1: PROGRAMMING TIME

$$\frac{T}{7.37 \text{ MHz} \times (\text{FRC Accuracy})\% \times (\text{FRC Tuning})\%}$$

For example, if the device is operating at +85°C, the FRC accuracy will be ±2%. If the TUN<5:0> bits (see Register 9-4) are set to 'b111111, the Minimum Row Write Time is:

$$T_{RW} = \frac{11064 \text{ Cycles}}{7.37 \text{ MHz} \times (1 + 0.02) \times (1 - 0.00375)} = 1.48 \text{ ms}$$

and, the Maximum Row Write Time is:

$$T_{RW} = \frac{11064 \text{ Cycles}}{7.37 \text{ MHz} \times (1 - 0.02) \times (1 - 0.00375)} = 1.54 \text{ ms}$$

Setting the WR bit (NVMCON<15>) starts the operation, and the WR bit is automatically cleared when the operation is finished.

5.4 Control Registers

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

The NVMCON register (Register 5-1) controls which blocks are to be erased, which memory type is to be programmed and the start of the programming cycle.

NVMKEY is a write-only register that is used for write protection. To start a programming or erase sequence, the user must consecutively write 0x55 and 0xAA to the NVMKEY register. Refer to **Section 5.3 “Programming Operations”** for further details.

PIC24HJXXXGPX06/X08/X10

5.4.1 PROGRAMMING ALGORITHM FOR FLASH PROGRAM MEMORY

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

1. Read eight rows of program memory (512 instructions) and store in data RAM.
2. Update the program data in RAM with the desired new data.
3. Erase the page (see Example 5-1):
 - a) Set the NVMOP bits (NVMCON<3:0>) to '0010' to configure for block erase. Set the ERASE (NVMCON<6>) and WREN (NVMCON<14>) bits.
 - b) Write the starting address of the page to be erased into the TBLPAG and W registers.
 - c) Perform a dummy table write operation (TBLWTL) to any address within the page that needs to be erased.
 - d) Write 0x55 to NVMKEY.
 - e) Write 0xAA to NVMKEY.
 - f) Set the WR bit (NVMCON<15>). The erase cycle begins and the CPU stalls for the duration of the erase cycle. When the erase is done, the WR bit is cleared automatically.
4. Write the first 64 instructions from data RAM into the program memory buffers (see Example 5-2).
5. Write the program block to Flash memory:
 - a) Set the NVMOP bits to '0001' to configure for row programming. Clear the ERASE bit and set the WREN bit.
 - b) Write 0x55 to NVMKEY.
 - c) Write 0xAA to NVMKEY.
 - d) Set the WR bit. The programming cycle begins and the CPU stalls for the duration of the write cycle. When the write to Flash memory is done, the WR bit is cleared automatically.
6. Repeat steps 4 and 5, using the next available 64 instructions from the block in data RAM by incrementing the value in TBLPAG, until all 512 instructions are written back to Flash memory.

For protection against accidental operations, the write initiate sequence for NVMKEY must be used to allow any erase or program operation to proceed. After the programming command has been executed, the user must wait for the programming time until programming is complete. The two instructions following the start of the programming sequence should be NOPs, as shown in Example 5-3.

EXAMPLE 5-1: 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
```

Note: A program memory page erase operation is set up by performing a dummy table write (TBLWTL) operation to any address within the page. This methodology is different from the page erase operation on dsPIC30F/33F devices in which the erase page was selected using a dedicated pair of registers (NVMADRU and NVMADR).

7.3 Interrupt Control and Status Registers

PIC24HJXXXGPX06/X08/X10 devices implement a total of 30 registers for the interrupt controller:

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

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

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

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

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

The INTTREG register contains the associated interrupt vector number and the new CPU interrupt priority level, which are latched into vector number (VECTNUM<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 7-1. 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>).

Although they are not specifically part of the interrupt control hardware, two of the CPU Control registers contain bits that control interrupt functionality. The CPU STATUS register, SR, contains the IPL<2:0> bits (SR<7:5>). These bits indicate the current CPU interrupt priority level. The user can change the current CPU priority level by writing to the IPL bits.

The CORCON register contains the IPL3 bit which, together with IPL<2:0>, also indicates the current CPU priority level. IPL3 is a read-only bit so that trap events cannot be masked by the user software.

All Interrupt registers are described in **Register 7-1, SR: CPU Status Register⁽¹⁾** through **Register 7-32, IPC17: Interrupt Priority Control Register 17**, in the following pages.

PIC24HJXXXGPX06/X08/X10

REGISTER 7-21: IPC6: INTERRUPT PRIORITY CONTROL REGISTER 6

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

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

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

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

bit 14-12 **T4IP<2:0>:** Timer4 Interrupt Priority bits
 111 = Interrupt is priority 7 (highest priority interrupt)
 •
 •
 •
 001 = Interrupt is priority 1
 000 = Interrupt source is disabled

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

bit 10-8 **OC4IP<2:0>:** Output Compare Channel 4 Interrupt Priority bits
 111 = Interrupt is priority 7 (highest priority interrupt)
 •
 •
 •
 001 = Interrupt is priority 1
 000 = Interrupt source is disabled

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

bit 6-4 **OC3IP<2:0>:** Output Compare Channel 3 Interrupt Priority bits
 111 = Interrupt is priority 7 (highest priority interrupt)
 •
 •
 •
 001 = Interrupt is priority 1
 000 = Interrupt source is disabled

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

bit 2-0 **DMA2IP<2:0>:** DMA Channel 2 Data Transfer Complete Interrupt Priority bits
 111 = Interrupt is priority 7 (highest priority interrupt)
 •
 •
 •
 001 = Interrupt is priority 1
 000 = Interrupt source is disabled

PIC24HJXXXGPX06/X08/X10

REGISTER 8-2: DMAxREQ: DMA CHANNEL x IRQ SELECT REGISTER

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

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

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15 **FORCE:** Force DMA Transfer bit⁽¹⁾

1 = Force a single DMA transfer (Manual mode)

0 = Automatic DMA transfer initiation by DMA request

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

bit 6-0 **IRQSEL<6:0>:** DMA Peripheral IRQ Number Select bits⁽²⁾

0000000-1111111 = DMAIRQ0-DMAIRQ127 selected to be Channel DMAREQ

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

2: Please see Table 8-1 for a complete listing of IRQ numbers for all interrupt sources.

PIC24HJXXXGPX06/X08/X10

NOTES:

10.2.2 IDLE MODE

Idle mode has these features:

- The CPU stops executing instructions.
- The WDT is automatically cleared.
- The system clock source remains active. By default, all peripheral modules continue to operate normally from the system clock source, but can also be selectively disabled (see **Section 10.4 “Peripheral Module Disable”**).
- If the WDT or FSCM is enabled, the LPRC also remains active.

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

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

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

10.2.3 INTERRUPTS COINCIDENT WITH POWER SAVE INSTRUCTIONS

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

10.3 Doze Mode

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

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

Doze mode is enabled by setting the DOZEN bit (`CLKDIV<11>`). The ratio between peripheral and core clock speed is determined by the `DOZE<2:0>` bits (`CLKDIV<14:12>`). There are eight possible configurations, from 1:1 to 1:128, with 1:1 being the default setting.

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

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

10.4 Peripheral Module Disable

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

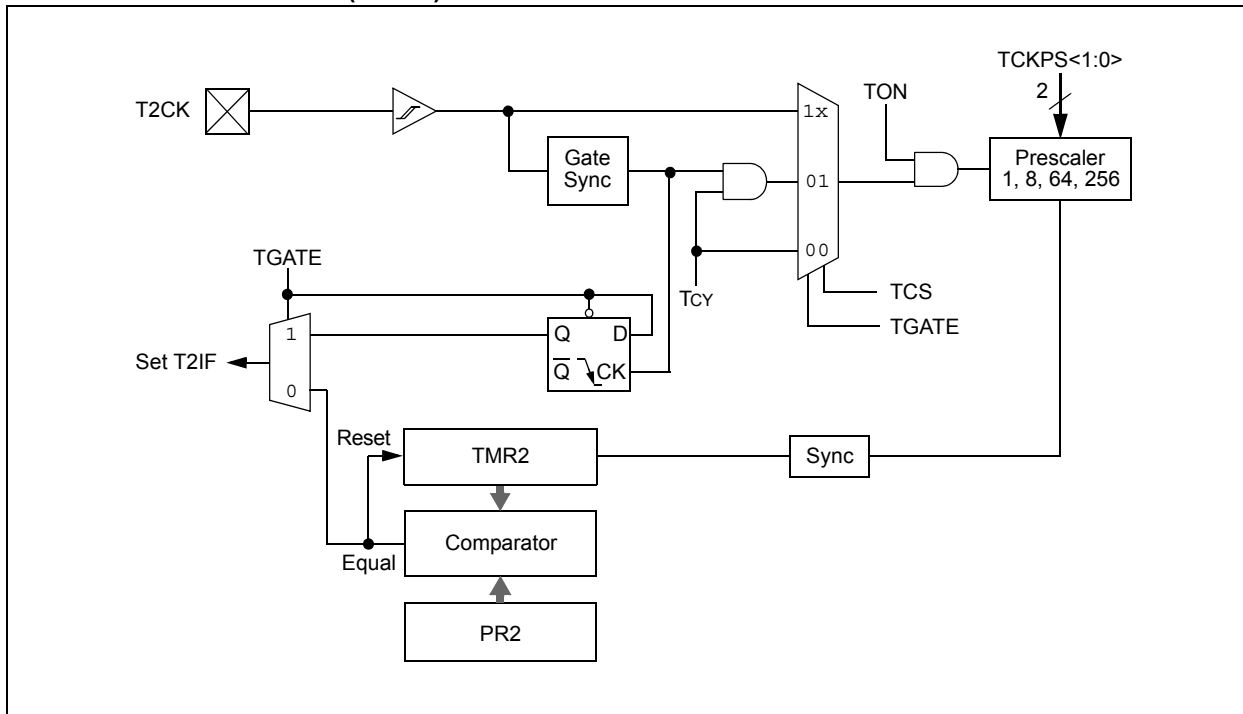
A peripheral module is only enabled if both the associated bit in the PMD register is cleared and the peripheral is supported by the specific dsPIC® DSC variant. If the peripheral is present in the device, it is enabled in the PMD register by default.

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

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

[illegible]

FIGURE 13-2: TIMER2 (16-BIT) BLOCK DIAGRAM



PIC24HJXXXGPX06/X08/X10

14.1 Input Capture Registers

REGISTER 14-1: 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 ⁽¹⁾	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⁾

1 = TMR2 contents are captured on capture event

0 = TMR3 contents are captured on capture event

bit 6-5 **ICI<1:0>:** Select Number of Captures per Interrupt bits

11 = Interrupt on every fourth capture event

10 = Interrupt on every third capture event

01 = Interrupt on every second capture event

00 = Interrupt on every capture event

bit 4 **ICOV:** Input Capture Overflow Status Flag bit (read-only)

1 = Input capture overflow occurred

0 = No input capture overflow occurred

bit 3 **ICBNE:** Input Capture Buffer Empty Status bit (read-only)

1 = Input capture buffer is not empty, at least one more capture value can be read

0 = Input capture buffer is empty

bit 2-0 **ICM<2:0>:** Input Capture Mode Select bits

111 = Input capture functions as interrupt pin only when device is in Sleep or Idle mode
(Rising edge detect only, all other control bits are not applicable.)

110 = Unused (module disabled)

101 = Capture mode, every 16th rising edge

100 = Capture mode, every 4th rising edge

011 = Capture mode, every rising edge

010 = Capture mode, every falling edge

001 = Capture mode, every edge (rising and falling)

(ICI<1:0> bits do not control interrupt generation for this mode.)

000 = Input capture module turned off

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

17.0 INTER-INTEGRATED CIRCUIT™ (I²C™)

Note: This data sheet summarizes the features of the PIC24HJXXXGPX06/X08/X10 family of devices. However, 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*”, **Section 19. “Inter-Integrated Circuit™ (I²C™)”** (DS70235), which is available from the Microchip web-site (www.microchip.com).

The Inter-Integrated Circuit (I²C) module provides complete hardware support for both Slave and Multi-Master modes of the I²C serial communication standard, with a 16-bit interface.

The PIC24HJXXXGPX06/X08/X10 devices have up to two I²C interface modules, denoted as I2C1 and I2C2. Each I²C module has a 2-pin interface: the SCLx pin is clock and the SDAx pin is data.

Each I²C module ‘x’ (x = 1 or 2) offers the following key features:

- I²C interface supporting both master and slave operation.
- I²C Slave mode supports 7 and 10-bit address.
- I²C Master mode supports 7 and 10-bit address.
- I²C Port allows bidirectional transfers between master and slaves.
- Serial clock synchronization for I²C port can be used as a handshake mechanism to suspend and resume serial transfer (SCLREL control).
- I²C supports multi-master operation; detects bus collision and will arbitrate accordingly.

17.1 Operating Modes

The hardware fully implements all the master and slave functions of the I²C Standard and Fast mode specifications, as well as 7 and 10-bit addressing.

The I²C module can operate either as a slave or a master on an I²C bus.

The following types of I²C operation are supported:

- I²C slave operation with 7-bit address
- I²C slave operation with 10-bit address
- I²C master operation with 7 or 10-bit address

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

17.2 I²C Registers

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

I2CxRSR is the shift register used for shifting data, whereas I2CxRCV is the buffer register to which data bytes are written, or from which data bytes are read. I2CxRCV is the receive buffer. 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. The I2CxBRG acts as the Baud Rate Generator (BRG) reload value.

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

19.3 Modes of Operation

The CAN module can operate in one of several operation modes selected by the user. These modes include:

- Initialization Mode
- Disable Mode
- Normal Operation Mode
- Listen Only Mode
- Listen All Messages Mode
- Loopback Mode

Modes are requested by setting the REQOP<2:0> bits (CiCTRL1<10:8>). Entry into a mode is Acknowledged by monitoring the OPMODE<2:0> bits (CiCTRL1<7:5>). The module will not change the mode and the OPMODE bits until a change in mode is acceptable, generally during bus Idle time, which is defined as at least 11 consecutive recessive bits.

19.3.1 INITIALIZATION MODE

In the Initialization mode, the module will not transmit or receive. The error counters are cleared and the interrupt flags remain unchanged. The programmer will have access to Configuration registers that are access restricted in other modes. The module will protect the user from accidentally violating the CAN protocol through programming errors. All registers which control the configuration of the module can not be modified while the module is on-line. The CAN module will not be allowed to enter the Configuration mode while a transmission is taking place. The Configuration mode serves as a lock to protect the following registers.

- All Module Control Registers
- Baud Rate and Interrupt Configuration Registers
- Bus Timing Registers
- Identifier Acceptance Filter Registers
- Identifier Acceptance Mask Registers

19.3.2 DISABLE MODE

In Disable mode, the module will not transmit or receive. The module has the ability to set the WAKIF bit due to bus activity, however, any pending interrupts will remain and the error counters will retain their value.

If the REQOP<2:0> bits (CiCTRL1<10:8>) = 001, the module will enter the Module Disable mode. If the module is active, the module will wait for 11 recessive bits on the CAN bus, detect that condition as an Idle bus, then accept the module disable command. When the OPMODE<2:0> bits (CiCTRL1<7:5>) = 001, that indicates whether the module successfully went into Module Disable mode. The I/O pins will revert to normal I/O function when the module is in the Module Disable mode.

The module can be programmed to apply a low-pass filter function to the CiRX input line while the module or the CPU is in Sleep mode. The WAKFIL bit (CiCFG2<14>) enables or disables the filter.

Note: Typically, if the CAN module is allowed to transmit in a particular mode of operation and a transmission is requested immediately after the CAN module has been placed in that mode of operation, the module waits for 11 consecutive recessive bits on the bus before starting transmission. If the user switches to Disable mode within this 11-bit period, then this transmission is aborted and the corresponding TXABT bit is set and TXREQ bit is cleared.

19.3.3 NORMAL OPERATION MODE

Normal Operation mode is selected when REQOP<2:0> = 000. In this mode, the module is activated and the I/O pins will assume the CAN bus functions. The module will transmit and receive CAN bus messages via the CiTX and CiRX pins.

19.3.4 LISTEN ONLY MODE

If the Listen Only mode is activated, the module on the CAN bus is passive. The transmitter buffers revert to the port I/O function. The receive pins remain inputs. For the receiver, no error flags or Acknowledge signals are sent. The error counters are deactivated in this state. The Listen Only mode can be used for detecting the baud rate on the CAN bus. To use this, it is necessary that there are at least two further nodes that communicate with each other.

19.3.5 LISTEN ALL MESSAGES MODE

The module can be set to ignore all errors and receive any message. The Listen All Messages mode is activated by setting REQOP<2:0> = '111'. In this mode, the data which is in the message assembly buffer, until the time an error occurred, is copied in the receive buffer and can be read via the CPU interface.

19.3.6 LOOPBACK MODE

If the Loopback mode is activated, the module will connect the internal transmit signal to the internal receive signal at the module boundary. The transmit and receive pins revert to their port I/O function.

PIC24HJXXXGPX06/X08/X10

TABLE 24-4: DC TEMPERATURE AND VOLTAGE SPECIFICATIONS

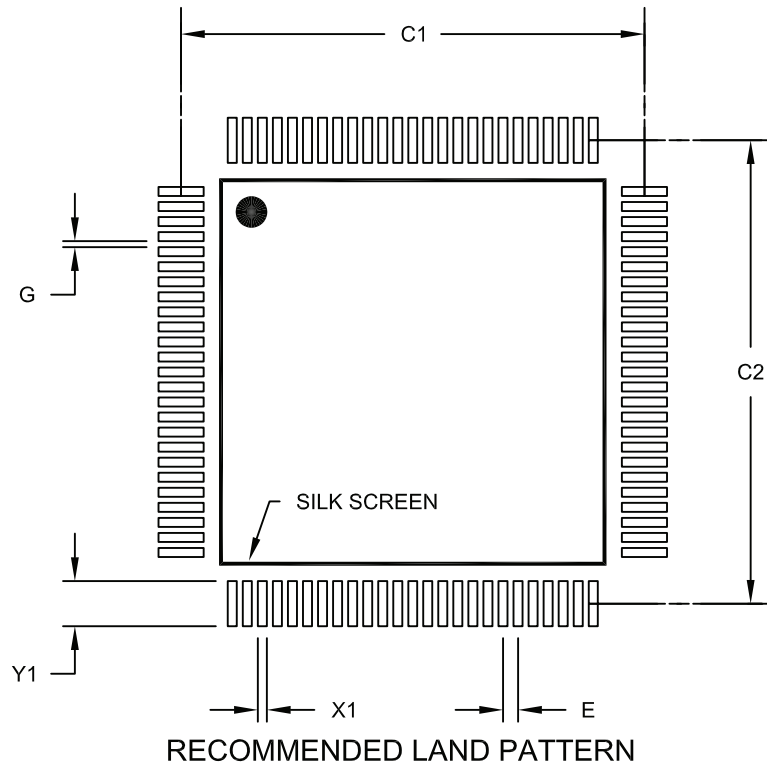
DC CHARACTERISTICS			Standard Operating Conditions: 3.0V to 3.6V (unless otherwise stated) Operating temperature $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ for Industrial				
Param No.	Symbol	Characteristic	Min	Typ ⁽¹⁾	Max	Units	Conditions
Operating Voltage							
DC10	Supply Voltage						
	VDD		3.0	—	3.6	V	—
DC12	VDR	RAM Data Retention Voltage⁽²⁾	1.8	—	—	V	—
DC16	VPOR	VDD Start Voltage⁽⁴⁾ 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.
4: VDD voltage must remain at VSS for a minimum of 200 μs to ensure POR.

PIC24HJXXXGPX06/X08/X10

100-Lead Plastic Thin Quad Flatpack (PF) – 14x14x1 mm Body, 2.00 mm [TQFP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E	0.50 BSC		
Contact Pad Spacing	C1		15.40	
Contact Pad Spacing	C2		15.40	
Contact Pad Width (X100)	X1			0.30
Contact Pad Length (X100)	Y1			1.50
Distance Between Pads	G	0.20		

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2110A

PIC24HJXXXGPX06/X08/X10

PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.

	PIC 24 HJ 256 GP6 10 T I/PT - XXX									
Microchip Trademark										
Architecture										
Flash Memory Family										
Program Memory Size (KB)										
Product Group										
Pin Count										
Tape and Reel Flag (if applicable)										
Temperature Range										
Package										
Pattern										

Architecture:	24	=	16-bit Microcontroller
Flash Memory Family:	HJ	=	Flash program memory, 3.3V, High-speed
Product Group:	GP2	=	General purpose family
	GP3	=	General purpose family
	GP5	=	General purpose family
	GP6	=	General purpose family
Pin Count:	06	=	64-pin
	10	=	100-pin
Temperature Range:	I	=	-40°C to +85°C (Industrial)
Package:	PT	=	10x10 or 12x12 mm TQFP (Thin Quad Flat-pack)
	PF	=	14x14 mm TQFP (Thin Quad Flatpack)
Pattern:	Three-digit QTP, SQTP, Code or Special Requirements (blank otherwise)		
	ES	=	Engineering Sample

Examples:

- PIC24HJ256GP210I/PT:
General-purpose PIC24H, 256 KB program memory, 100-pin, Industrial temp., TQFP package.
- PIC24HJ64GP506I/PT-ES:
General-purpose PIC24H, 64 KB program memory, 64-pin, Industrial temp., TQFP package, Engineering Sample.