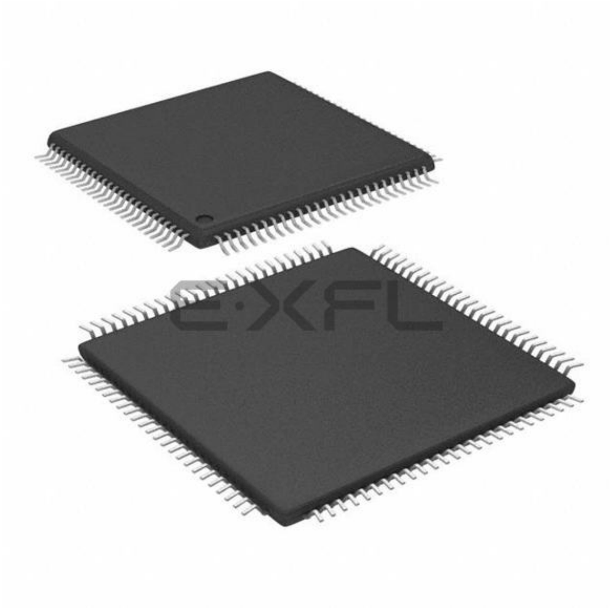


Welcome to E-XFL.COM

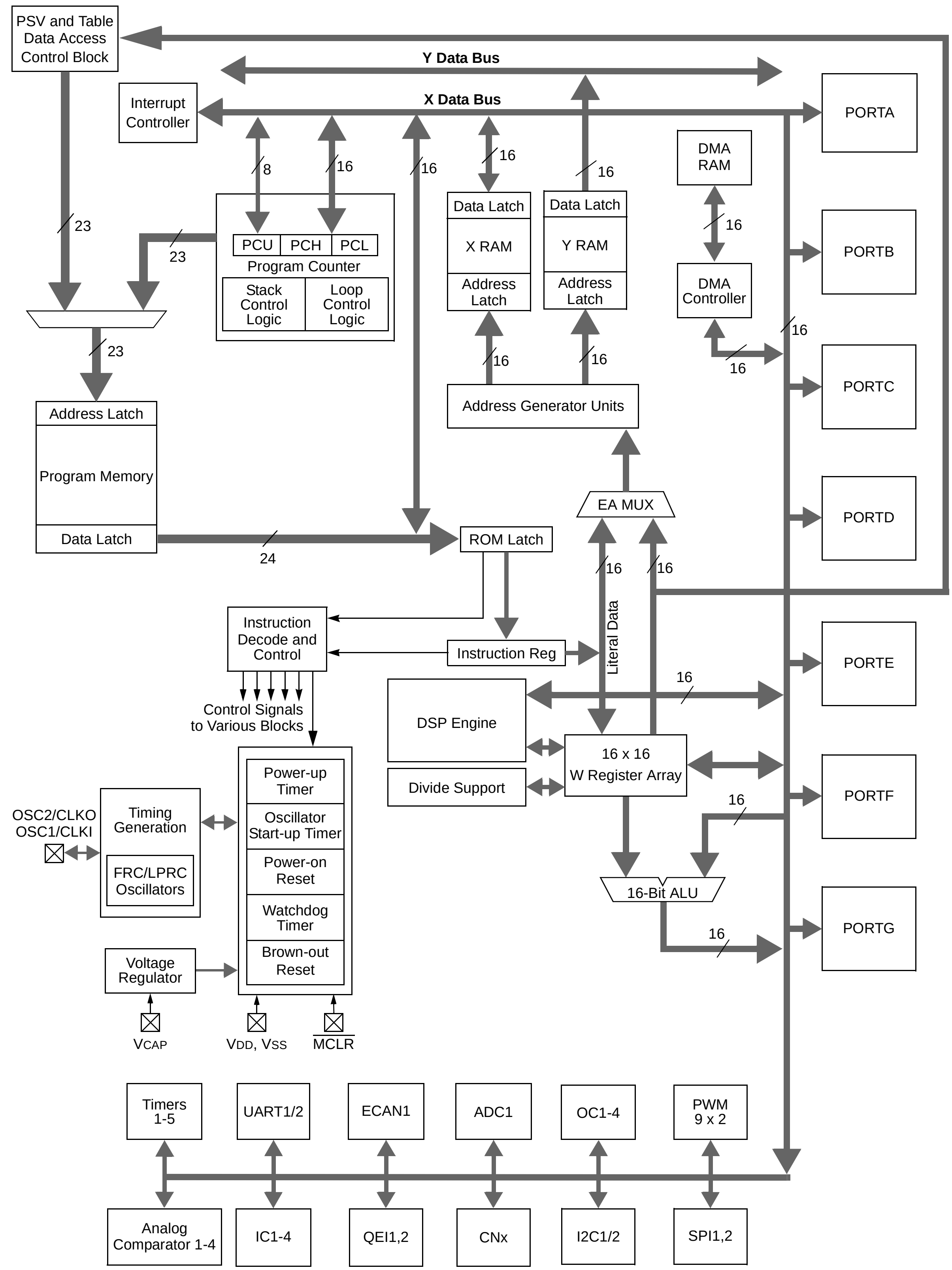
### What is "Embedded - Microcontrollers"?

"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 | dsPIC |
| Core Size | 16-Bit |
| Speed | 50 MIPs |
| Connectivity | I²C, IrDA, LINbus, SPI, UART/USART |
| Peripherals | Brown-out Detect/Reset, QEI, POR, PWM, WDT |
| Number of I/O | 85 |
| Program Memory Size | 32KB (32K x 8) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 4K x 8 |
| Voltage - Supply (Vcc/Vdd) | 3V ~ 3.6V |
| Data Converters | A/D 24x10b; D/A 1x10b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 100-TQFP |
| Supplier Device Package | 100-TQFP (12x12) |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/dspic33fj32gs610-50i-pt |

**FIGURE 1-1: DEVICE 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.

## 2.5 ICSP Pins

The PGECx and PGEDx pins are used for In-Circuit Serial Programming™ (ICSP™) and debugging purposes. It is recommended to keep the trace length between the ICSP connector and the ICSP pins on the device as short as possible. If the ICSP connector is expected to experience an ESD event, a series resistor is recommended, with the value in the range of a few tens of Ohms, not to exceed 100 Ohms.

Pull-up resistors, series diodes, and capacitors on the PGECx and PGEDx pins are not recommended as they will interfere with the programmer/debugger communications to the device. If such discrete components are an application requirement, they should be removed from the circuit during programming and debugging. Alternatively, refer to the AC/DC characteristics and timing requirements information in the respective device Flash programming specification for information on capacitive loading limits and pin input voltage high ($V_{IH}$) and input low ($V_{IL}$) requirements.

Ensure that the “Communication Channel Select” (i.e., PGECx/PGEDx pins) programmed into the device matches the physical connections for the ICSP to MPLAB® ICD 3 or MPLAB REAL ICE™.

For more information on ICD 3 and REAL ICE connection requirements, refer to the following documents that are available on the Microchip web site.

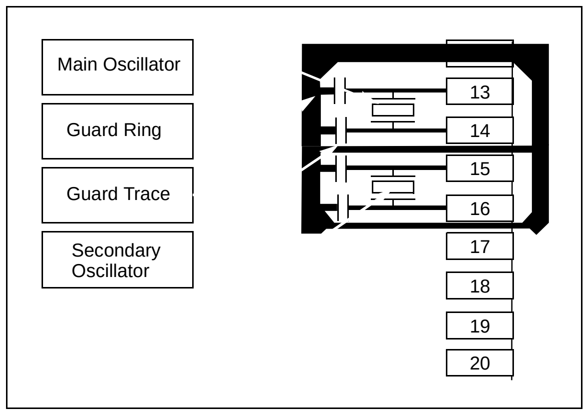
- *“Using MPLAB® ICD 3”* (poster) (DS51765)
- *“MPLAB® ICD 3 Design Advisory”* (DS51764)
- *“MPLAB® REAL ICE™ In-Circuit Debugger User’s Guide” (*DS51616)
- *“Using MPLAB® REAL ICE™”* (poster) (DS51749)

## 2.6 External Oscillator Pins

Many DSCs have options for at least two oscillators: a high-frequency primary oscillator and a low-frequency secondary oscillator (refer to **Section 9.0 “Oscillator Configuration”** for details).

The oscillator circuit should be placed on the same side of the board as the device. Also, place the oscillator circuit close to the respective oscillator pins, not exceeding one-half inch (12 mm) distance between them. The load capacitors should be placed next to the oscillator itself, on the same side of the board. Use a grounded copper pour around the oscillator circuit to isolate them from surrounding circuits. The grounded copper pour should be routed directly to the MCU ground. Do not run any signal traces or power traces inside the ground pour. Also, if using a two-sided board, avoid any traces on the other side of the board where the crystal is placed. A suggested layout is shown in Figure 2-3.

**FIGURE 2-3: SUGGESTED PLACEMENT OF THE OSCILLATOR CIRCUIT**



### 4.2.5 X AND Y DATA SPACES

The core has two data spaces, X and Y. These data spaces can be considered either separate (for some DSP instructions), or as one unified linear address range (for MCU instructions). The data spaces are accessed using two Address Generation Units (AGUs) and separate data paths. This feature allows certain instructions to concurrently fetch two words from RAM, thereby enabling efficient execution of DSP algorithms such as Finite Impulse Response (FIR) filtering and Fast Fourier Transform (FFT).

The X data space is used by all instructions and supports all addressing modes. X data space has separate read and write data buses. The X read data bus is the read data path for all instructions that view data space as combined X and Y address space. It is also the X data prefetch path for the dual operand DSP instructions (`MAC` class).

The Y data space is used in concert with the X data space by the `MAC` class of instructions (`CLR`, `ED`, `EDAC`, `MAC`, `MOVSAC`, `MPY`, `MPY.N` and `MSC`) to provide two concurrent data read paths.

Both the X and Y data spaces support Modulo Addressing mode for all instructions, subject to addressing mode restrictions. Bit-Reversed Addressing mode is only supported for writes to X data space.

All data memory writes, including in DSP instructions, view data space as combined X and Y address space. The boundary between the X and Y data spaces is device-dependent and is not user-programmable.

All Effective Addresses (EAs) are 16 bits wide and point to bytes within the data space. Therefore, the data space address range is 64 Kbytes, or 32K words, though the implemented memory locations vary by device.

### 4.2.6 DMA RAM

Some devices contain 1 Kbyte of dual ported DMA RAM, which is located at the end of Y data space. Memory locations that are part of Y data RAM and are in the DMA RAM space are accessible simultaneously by the CPU and the DMA Controller module. DMA RAM is utilized by the DMA Controller to store data to be transferred to various peripherals using DMA, as well as data transferred from various peripherals using DMA. The DMA RAM can be accessed by the DMA Controller without having to steal cycles from the CPU.

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

**TABLE 4-1: CPU CORE REGISTER MAP**

| File 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 |
| ACCAL | 0022 | ACCAL | | | | | | | | | | | | | | | | xxxx |
| ACCAH | 0024 | ACCAH | | | | | | | | | | | | | | | | xxxx |
| ACCAU | 0026 | ACCA<39> | ACCA<39> | ACCA<39> | ACCA<39> | ACCA<39> | ACCA<39> | ACCA<39> | ACCA<39> | ACCAU | | | | | | | | xxxx |
| ACCBL | 0028 | ACCBL | | | | | | | | | | | | | | | | xxxx |
| ACCBH | 002A | ACCBH | | | | | | | | | | | | | | | | xxxx |
| ACCBU | 002C | ACCB<39> | ACCB<39> | ACCB<39> | ACCB<39> | ACCB<39> | ACCB<39> | ACCB<39> | ACCB<39> | ACCBU | | | | | | | | xxxx |
| PCL | 002E | Program Counter Low Byte 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 |
| DCOUNT | 0038 | DCOUNT<15:0> | | | | | | | | | | | | | | | | xxxx |
| DOSTARTL | 003A | DOSTARTL<15:1> | | | | | | | | | | | | | | | 0 | xxxx |
| DOSTARTH | 003C | — | — | — | — | — | — | — | — | — | — | DOSTARTH<5:0> | | | | | | 00xx |
| DOENDL | 003E | DOENDL<15:1> | | | | | | | | | | | | | | | 0 | xxxx |
| DOENDH | 0040 | — | — | — | — | — | — | — | — | — | — | DOENDH | | | | | | 00xx |
| SR | 0042 | OA | OB | SA | SB | OAB | SAB | DA | DC | IPL2 | IPL1 | IPL0 | RA | N | OV | Z | C | 0000 |

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

### TABLE 4-28: UART1 REGISTER MAP

| File 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 | PDSEL1 | PDSEL0 | STSEL | 0000 |
| U1STA | 0222 | UTXISEL1 | UTXINV | UTXISEL0 | — | UTXBRK | UTXEN | UTXBF | TRMT | URXISEL1 | URXISEL0 | ADDEN | RIDLE | PERR | FERR | OERR | URXDA | 0110 |
| U1TXREG | 0224 | — | — | — | — | — | — | — | UART1 Transmit Register | | | | | | | | | xxxx |
| U1RXREG | 0226 | — | — | — | — | — | — | — | UART1 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 4-29: UART2 REGISTER MAP

| File Name | SFR Addr | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | All Resets |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| U2MODE | 0230 | UARTEN | — | USIDL | IREN | RTSMD | — | UEN1 | UEN0 | WAKE | LPBACK | ABAUD | URXINV | BRGH | PDSEL1 | PDSEL0 | STSEL | 0000 |
| U2STA | 0232 | UTXISEL1 | UTXINV | UTXISEL0 | — | UTXBRK | UTXEN | UTXBF | TRMT | URXISEL1 | URXISEL0 | ADDEN | RIDLE | PERR | FERR | OERR | URXDA | 0110 |
| U2TXREG | 0234 | — | — | — | — | — | — | — | UART2 Transmit Register | | | | | | | | | xxxx |
| U2RXREG | 0236 | — | — | — | — | — | — | — | UART2 Receive Register | | | | | | | | | 0000 |
| U2BRG | 0238 | Baud Rate Generator Prescaler | | | | | | | | | | | | | | | | 0000 |

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

## TABLE 4-60: PMD REGISTER MAP FOR dsPIC33FJ32GS610 DEVICES

| File 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PMD1 | 0770 | T5MD | T4MD | T3MD | T2MD | T1MD | QEI1MD | PWMMD | — | I2C1MD | U2MD | U1MD | SPI2MD | SPI1MD | — | — | ADCMD | 0000 |
| PMD2 | 0772 | — | — | — | — | IC4MD | IC3MD | IC2MD | IC1MD | — | — | — | — | OC4MD | OC3MD | OC2MD | OC1MD | 0000 |
| PMD3 | 0774 | — | — | — | — | — | CMPMD | — | — | — | — | QEI2MD | — | — | — | I2C2MD | — | 0000 |
| PMD4 | 0776 | — | — | — | — | — | — | — | — | — | — | — | — | REFOMD | — | — | — | 0000 |
| PMD6 | 077A | PWM8MD | PWM7MD | PWM6MD | PWM5MD | PWM4MD | PWM3MD | PWM2MD | PWM1MD | — | — | — | — | — | — | — | — | 0000 |
| PMD7 | 077C | — | — | — | — | CMP4MD | CMP3MD | CMP2MD | CMP1MD | — | — | — | — | — | — | — | PWM9MD | 0000 |

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

## TABLE 4-61: PMD REGISTER MAP FOR dsPIC33FJ64GS608 DEVICES

| File 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PMD1 | 0770 | T5MD | T4MD | T3MD | T2MD | T1MD | QEI1MD | PWMMD | — | I2C1MD | U2MD | U1MD | SPI2MD | SPI1MD | — | C1MD | ADCMD | 0000 |
| PMD2 | 0772 | — | — | — | — | IC4MD | IC3MD | IC2MD | IC1MD | — | — | — | — | OC4MD | OC3MD | OC2MD | OC1MD | 0000 |
| PMD3 | 0774 | — | — | — | — | — | CMPMD | — | — | — | — | QEI2MD | — | — | — | I2C2MD | — | 0000 |
| PMD4 | 0776 | — | — | — | — | — | — | — | — | — | — | — | — | REFOMD | — | — | — | 0000 |
| PMD6 | 077A | PWM8MD | PWM7MD | PWM6MD | PWM5MD | PWM4MD | PWM3MD | PWM2MD | PWM1MD | — | — | — | — | — | — | — | — | 0000 |
| PMD7 | 077C | — | — | — | — | CMP4MD | CMP3MD | CMP2MD | CMP1MD | — | — | — | — | — | — | — | — | 0000 |

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

## TABLE 4-62: PMD REGISTER MAP FOR dsPIC33FJ32GS608 DEVICES

| File 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PMD1 | 0770 | T5MD | T4MD | T3MD | T2MD | T1MD | QEI1MD | PWMMD | — | I2C1MD | U2MD | U1MD | SPI2MD | SPI1MD | — | — | ADCMD | 0000 |
| PMD2 | 0772 | — | — | — | — | IC4MD | IC3MD | IC2MD | IC1MD | — | — | — | — | OC4MD | OC3MD | OC2MD | OC1MD | 0000 |
| PMD3 | 0774 | — | — | — | — | — | CMPMD | — | — | — | — | QEI2MD | — | — | — | I2C2MD | — | 0000 |
| PMD4 | 0776 | — | — | — | — | — | — | — | — | — | — | — | — | REFOMD | — | — | — | 0000 |
| PMD6 | 077A | PWM8MD | PWM7MD | PWM6MD | PWM5MD | PWM4MD | PWM3MD | PWM2MD | PWM1MD | — | — | — | — | — | — | — | — | 0000 |
| PMD7 | 077C | — | — | — | — | CMP4MD | CMP3MD | CMP2MD | CMP1MD | — | — | — | — | — | — | — | — | 0000 |

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

**NOTES:**

### 5.4.1 PROGRAMMING ALGORITHM FOR FLASH PROGRAM MEMORY

One row of program Flash memory can be programmed at a time. To achieve 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 5-1):
   a) Set the NVMOPx 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) Write 0x55 to NVMKEY.
   d) Write 0xAA 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 5-2).
5. Write the program block to Flash memory:
   a) Set the NVMOPx 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 application 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
```

**REGISTER 7-3: INTCON1: INTERRUPT CONTROL REGISTER 1**

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

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

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

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

bit 14 **OVAERR:** Accumulator A Overflow Trap Flag bit
1 = Trap was caused by an overflow of Accumulator A
0 = Trap was not caused by an overflow of Accumulator A

bit 13 **OVBERR:** Accumulator B Overflow Trap Flag bit
1 = Trap was caused by an overflow of Accumulator B
0 = Trap was not caused by an overflow of Accumulator B

bit 12 **COVAERR:** Accumulator A Catastrophic Overflow Trap Flag bit
1 = Trap was caused by a catastrophic overflow of Accumulator A
0 = Trap was not caused by a catastrophic overflow of Accumulator A

bit 11 **COVBERR:** Accumulator B Catastrophic Overflow Trap Flag bit
1 = Trap was caused by a catastrophic overflow of Accumulator B
0 = Trap was not caused by a catastrophic overflow of Accumulator B

bit 10 **OVATE:** Accumulator A Overflow Trap Enable bit
1 = Trap overflow of Accumulator A
0 = Trap is disabled

bit 9 **OVBTE:** Accumulator B Overflow Trap Enable bit
1 = Trap overflow of Accumulator B
0 = Trap is disabled

bit 8 **COVTE:** Catastrophic Overflow Trap Enable bit
1 = Trap on a catastrophic overflow of Accumulator A or B is enabled
0 = Trap is disabled

bit 7 **SFTACERR:** Shift Accumulator Error Status bit
1 = Math error trap was caused by an invalid accumulator shift
0 = Math error trap was not caused by an invalid accumulator shift

bit 6 **DIV0ERR:** Arithmetic Error Status bit
1 = Math error trap was caused by a divide-by-zero
0 = Math error trap was not caused by a divide-by-zero

bit 5 **DMACERR:** DMA Controller Error Status bit
1 = DMA Controller error trap has occurred
0 = DMA Controller error trap has not occurred

bit 4 **MATHERR:** Arithmetic Error Status bit
1 = Math error trap has occurred
0 = Math error trap has not occurred

### REGISTER 7-10: IFS5: INTERRUPT FLAG STATUS REGISTER 5

| R/W-0 | R/W-0 | R/W-0 | U-0 | U-0 | U-0 | U-0 | U-0 |
|---|---|---|---|---|---|---|---|
| PWM2IF | PWM1IF | ADCP12IF | — | — | — | — | — |
| bit 15 | | | | | | | bit 8 |

| U-0 | U-0 | U-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | U-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | ADCP11IF | ADCP10IF | ADCP9IF | ADCP8IF | — |
| 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 **PWM2IF:** PWM2 Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 14 **PWM1IF:** PWM1 Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 13 **ADCP12IF:** ADC Pair 12 Conversion Done Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 12-5 **Unimplemented:** Read as ‘0’

bit 4 **ADCP11IF:** ADC Pair 11 Conversion Done Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 3 **ADCP10IF:** ADC Pair 10 Conversion Done Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 2 **ADCP9IF:** ADC Pair 9 Conversion Done Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 1 **ADCP8IF:** ADC Pair 8 Conversion Done Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

bit 0 **Unimplemented:** Read as ‘0’

### REGISTER 7-44: IPC28: INTERRUPT PRIORITY CONTROL REGISTER 28

| U-0 | R/W-1 | R/W-0 | R/W-0 | U-0 | R/W-1 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | ADCP5IP2 | ADCP5IP1 | ADCP5IP0 | — | ADCP4IP2 | ADCP4IP1 | ADCP4IP0 |
| 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 |
|---|---|---|---|---|---|---|---|
| — | ADCP3IP2 | ADCP3IP1 | ADCP3IP0 | — | ADCP2IP2 | ADCP2IP1 | ADCP2IP0 |
| 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 **ADCP5IP<2:0>:** ADC Pair 5 Conversion Done 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 **ADCP4IP<2:0>:** ADC Pair 4 Conversion Done 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 **ADCP3IP<2:0>:** ADC Pair 3 Conversion Done 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 **ADCP2IP<2:0>:** ADC Pair 2 Conversion Done Interrupt Priority bits
111 = Interrupt is Priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is Priority 1
000 = Interrupt source is disabled

### REGISTER 16-23: LEBCONx: LEADING-EDGE BLANKING CONTROL x REGISTER (CONTINUED)

| | |
|---|---|
| bit 1 | **BPLH:** Blanking in PWMxL High Enable bit<br>`1` = State blanking (of current-limit and/or Fault input signals) when PWMxL output is high<br>`0` = No blanking when PWMxL output is high |
| bit 0 | **BPLL:** Blanking in PWMxL Low Enable bit<br>`1` = State blanking (of current-limit and/or Fault input signals) when PWMxL output is low<br>`0` = No blanking when PWMxL output is low |

**Note 1:** The blanking signal is selected via the BLANKSELx bits in the AUXCONx register.

### REGISTER 19-1: 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:** | HC = Hardware 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 **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 $I^2C$ pins are controlled by port functions

bit 14 **Unimplemented:** Read as ‘0’

bit 13 **I2CSIDL:** I2Cx Stop in Idle Mode bit
1 = Discontinues module operation when device enters Idle mode
0 = Continues module operation in Idle mode

bit 12 **SCLREL:** SCLx Release Control bit (when operating as $I^2C$ slave)
1 = Releases SCLx clock
0 = Holds 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 is clear at beginning of slave transmission. Hardware is clear at end of slave reception.
If STREN = 0:
Bit is R/S (i.e., software can only write ‘1’ to release clock). Hardware is clear at beginning of slave transmission.

bit 11 **IPMIEN:** Intelligent Peripheral Management Interface (IPMI) Enable bit
1 = IPMI mode is enabled; all addresses are Acknowledged
0 = IPMI mode is 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 is disabled
0 = Slew rate control is enabled

bit 8 **SMEN:** SMBus Input Levels bit
1 = Enables I/O pin thresholds compliant with SMBus specification
0 = Disables SMBus input thresholds

bit 7 **GCEN:** General Call Enable bit (when operating as $I^2C$™ slave)
1 = Enables interrupt when a general call address is received in the I2CxRSR (module is enabled for reception)
0 = General call address is disabled

bit 6 **STREN:** SCLx Clock Stretch Enable bit (when operating as $I^2C$ slave)
Used in conjunction with the SCLREL bit.
1 = Enables software or receives clock stretching
0 = Disables software or receives clock stretching

## REGISTER 20-1: 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[1] | — | USIDL | IREN[2] | RTSMD | — | UEN1 | UEN0 |
| 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 | PDSEL1 | PDSEL0 | STSEL |
| bit 7 | | | | | | | bit 0 |

| **Legend:** | HC = Hardware 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 **UARTEN:** UARTx Enable bit[1]
- `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 is minimal

bit 14 **Unimplemented:** Read as ‘0’

bit 13 **USIDL:** UARTx Stop in Idle Mode bit
- `1` = Discontinues module operation when device enters Idle mode
- `0` = Continues module operation in Idle mode

bit 12 **IREN:** IrDA® Encoder and Decoder Enable bit[2]
- `1` = IrDA encoder and decoder are enabled
- `0` = IrDA encoder and decoder are disabled

bit 11 **RTSMD:** Mode Selection for $\overline{\text{UxRTS}}$ Pin bit
- `1` = $\overline{\text{UxRTS}}$ pin is in Simplex mode
- `0` = $\overline{\text{UxRTS}}$ pin is in Flow Control mode

bit 10 **Unimplemented:** Read as ‘0’

bit 9-8 **UEN<1:0>:** UARTx Pin Enable bits
- `11` = UxTX, UxRX and BCLK pins are enabled and used; $\overline{\text{UxCTS}}$ pin is 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 is controlled by port latches
- `00` = UxTX and UxRX pins are enabled and used; $\overline{\text{UxCTS}}$ and $\overline{\text{UxRTS}}$/BCLK pins are 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 is generated on falling edge, bit is cleared in hardware on following rising edge
- `0` = No wake-up is enabled

bit 6 **LPBACK:** UARTx Loopback Mode Select bit
- `1` = Enables Loopback mode
- `0` = Loopback mode is disabled

bit 5 **ABAUD:** Auto-Baud Enable bit
- `1` = Enables 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 is disabled or completed

**Note 1:** Refer to **“UART”** (DS70188) in the *“dsPIC33/PIC24 Family Reference Manual”* for information on enabling the UART module for receive or transmit operation. That section of the manual is available on the Microchip web site: www.microchip.com.

**2:** This feature is only available for the 16x BRG mode (BRGH = `0`).

### REGISTER 21-6: CxINTF: ECANx INTERRUPT FLAG REGISTER

| U-0 | U-0 | R-0 | R-0 | R-0 | R-0 | R-0 | R-0 |
|---|---|---|---|---|---|---|---|
| — | — | TXBO | TXBP | RXBP | TXWAR | RXWAR | EWARN |
| bit 15 | | | | | | | bit 8 |

| R/C-0 | R/C-0 | R/C-0 | U-0 | R/C-0 | R/C-0 | R/C-0 | R/C-0 |
|---|---|---|---|---|---|---|---|
| IVRIF | WAKIF | ERRIF | — | FIFOIF | RBOVIF | RBIF | TBIF |
| bit 7 | | | | | | | bit 0 |

| Legend: | C = Writable, but only ‘0’ can be written to clear the 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 15-14 | **Unimplemented:** Read as ‘0’ |
| bit 13 | **TXBO:** Transmitter in Error State Bus Off bit<br>1 = Transmitter is in Bus Off state<br>0 = Transmitter is not in Bus Off state |
| bit 12 | **TXBP:** Transmitter in Error State Bus Passive bit<br>1 = Transmitter is in Bus Passive state<br>0 = Transmitter is not in Bus Passive state |
| bit 11 | **RXBP:** Receiver in Error State Bus Passive bit<br>1 = Receiver is in Bus Passive state<br>0 = Receiver is not in Bus Passive state |
| bit 10 | **TXWAR:** Transmitter in Error State Warning bit<br>1 = Transmitter is in Error Warning state<br>0 = Transmitter is not in Error Warning state |
| bit 9 | **RXWAR:** Receiver in Error State Warning bit<br>1 = Receiver is in Error Warning state<br>0 = Receiver is not in Error Warning state |
| bit 8 | **EWARN:** Transmitter or Receiver in Error State Warning bit<br>1 = Transmitter or receiver is in Error Warning state<br>0 = Transmitter or receiver is not in Error Warning state |
| bit 7 | **IVRIF:** Invalid Message Received Interrupt Flag bit<br>1 = Interrupt request has occurred<br>0 = Interrupt request has not occurred |
| bit 6 | **WAKIF:** Bus Wake-up Activity Interrupt Flag bit<br>1 = Interrupt request has occurred<br>0 = Interrupt request has not occurred |
| bit 5 | **ERRIF:** Error Interrupt Flag bit (multiple sources in CxINTF<13:8> register bits)<br>1 = Interrupt request has occurred<br>0 = Interrupt request has not occurred |
| bit 4 | **Unimplemented:** Read as ‘0’ |
| bit 3 | **FIFOIF:** FIFO Almost Full Interrupt Flag bit<br>1 = Interrupt request has occurred<br>0 = Interrupt request has not occurred |
| bit 2 | **RBOVIF:** RX Buffer Overflow Interrupt Flag bit<br>1 = Interrupt request has occurred<br>0 = Interrupt request has not occurred |

### REGISTER 22-7: ADCPC1: ADC CONVERT PAIR CONTROL REGISTER 1 (CONTINUED)

bit 12-8 **TRGSRC3<4:0>:** Trigger 3 Source Selection bits

Selects trigger source for conversion of analog channels AN7 and AN6.
11111 = Timer2 period match
11110 = PWM Generator 8 current-limit ADC trigger
11101 = PWM Generator 7 current-limit ADC trigger
11100 = PWM Generator 6 current-limit ADC trigger
11011 = PWM Generator 5 current-limit ADC trigger
11010 = PWM Generator 4 current-limit ADC trigger
11001 = PWM Generator 3 current-limit ADC trigger
11000 = PWM Generator 2 current-limit ADC trigger
10111 = PWM Generator 1 current-limit ADC trigger
10110 = PWM Generator 9 secondary trigger is selected
10101 = PWM Generator 8 secondary trigger is selected
10100 = PWM Generator 7 secondary trigger is selected
10011 = PWM Generator 6 secondary trigger is selected
10010 = PWM Generator 5 secondary trigger is selected
10001 = PWM Generator 4 secondary trigger is selected
10000 = PWM Generator 3 secondary trigger is selected
01111 = PWM Generator 2 secondary trigger is selected
01110 = PWM Generator 1 secondary trigger is selected
01101 = PWM secondary Special Event Trigger is selected
01100 = Timer1 period match
01011 = PWM Generator 8 primary trigger is selected
01010 = PWM Generator 7 primary trigger is selected
01001 = PWM Generator 6 primary trigger is selected
01000 = PWM Generator 5 primary trigger is selected
00111 = PWM Generator 4 primary trigger is selected
00110 = PWM Generator 3 primary trigger is selected
00101 = PWM Generator 2 primary trigger is selected
00100 = PWM Generator 1 primary trigger is selected
00011 = PWM Special Event Trigger is selected
00010 = Global software trigger is selected
00001 = Individual software trigger is selected
00000 = No conversion is enabled

bit 7 **IRQEN2:** Interrupt Request Enable 2 bit

1 = Enables IRQ generation when requested conversion of Channels AN5 and AN4 is completed
0 = IRQ is not generated

bit 6 **PEND2:** Pending Conversion Status 2 bit

1 = Conversion of Channels AN5 and AN4 is pending; set when selected trigger is asserted
0 = Conversion is complete

bit 5 **SWTRG2:** Software Trigger 2 bit

1 = Starts conversion of AN5 and AN4 (if selected by the TRGSRCx<4:0> bits)[(1)]
This bit is automatically cleared by hardware when the PEND2 bit is set.
0 = Conversion has not started

**Note 1:** The trigger source must be set as an individual software trigger prior to setting this bit to '1'. If other conversions are in progress, the conversion is performed when the conversion resources are available.

**REGISTER 22-12: ADCPC6: ADC CONVERT PAIR CONTROL REGISTER 6(2) (CONTINUED)**

bit 4-0 **TRGSRC12<4:0>:** Trigger 12 Source Selection bits

Selects trigger source for conversion of analog channels AN25 and AN24.
11111 = Timer2 period match
11110 = PWM Generator 8 current-limit ADC trigger
11101 = PWM Generator 7 current-limit ADC trigger
11100 = PWM Generator 6 current-limit ADC trigger
11011 = PWM Generator 5 current-limit ADC trigger
11010 = PWM Generator 4 current-limit ADC trigger
11001 = PWM Generator 3 current-limit ADC trigger
11000 = PWM Generator 2 current-limit ADC trigger
10111 = PWM Generator 1 current-limit ADC trigger
10110 = PWM Generator 9 secondary trigger selected
10101 = PWM Generator 8 secondary trigger selected
10100 = PWM Generator 7 secondary trigger selected
10011 = PWM Generator 6 secondary trigger selected
10010 = PWM Generator 5 secondary trigger selected
10001 = PWM Generator 4 secondary trigger selected
10000 = PWM Generator 3 secondary trigger selected
01111 = PWM Generator 2 secondary trigger selected
01110 = PWM Generator 1 secondary trigger selected
01101 = PWM secondary Special Event Trigger selected
01100 = Timer1 period match
01011 = PWM Generator 8 primary trigger selected
01010 = PWM Generator 7 primary trigger selected
01001 = PWM Generator 6 primary trigger selected
01000 = PWM Generator 5 primary trigger selected
00111 = PWM Generator 4 primary trigger selected
00110 = PWM Generator 3 primary trigger selected
00101 = PWM Generator 2 primary trigger selected
00100 = PWM Generator 1 primary trigger selected
00011 = PWM Special Event Trigger selected
00010 = Global software trigger selected
00001 = Individual software trigger selected
00000 = No conversion is enabled

**Note 1:** The trigger source must be set as an individual software trigger prior to setting this bit to '1'. If other conversions are in progress, the conversion is performed when the conversion resources are available.

**2:** This register is not available on dsPIC33FJ32GS406 and dsPIC33FJ64GS406 devices.

**FIGURE 27-21: I2Cx BUS START/STOP BITS TIMING CHARACTERISTICS (SLAVE MODE)**

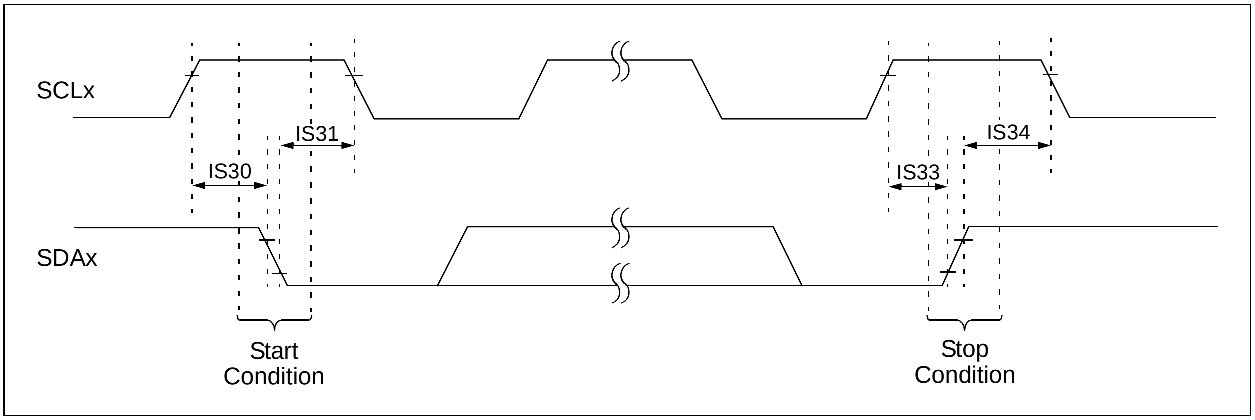
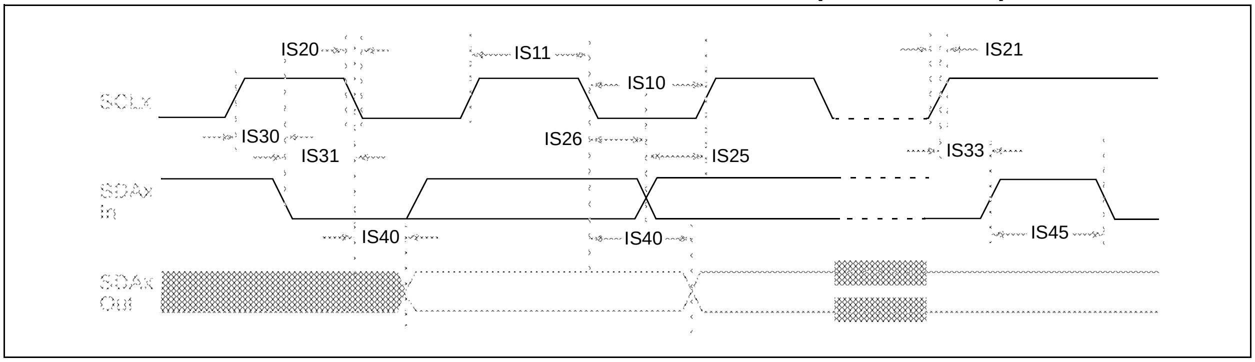


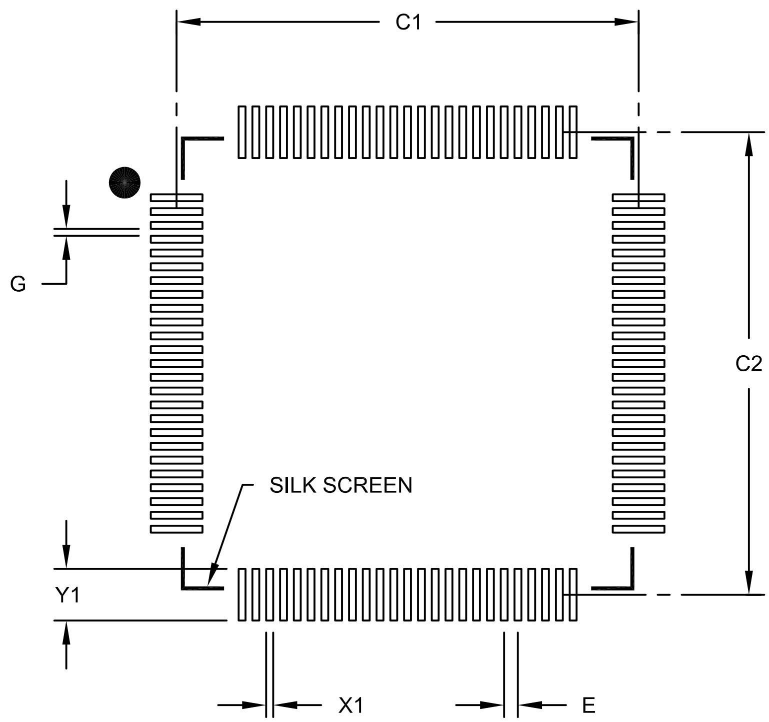

**FIGURE 27-22: I2Cx BUS DATA TIMING CHARACTERISTICS (SLAVE MODE)**



100-Lead Plastic Thin Quad Flatpack (PT) - 12x12x1mm 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




RECOMMENDED LAND PATTERN

| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Contact Pitch | E | 0.40 BSC | | |
| Contact Pad Spacing | C1 | | 13.40 | |
| Contact Pad Spacing | C2 | | 13.40 | |
| Contact Pad Width (X100) | X1 | | | 0.20 |
| 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. C04-2100B

## U



## V



## W