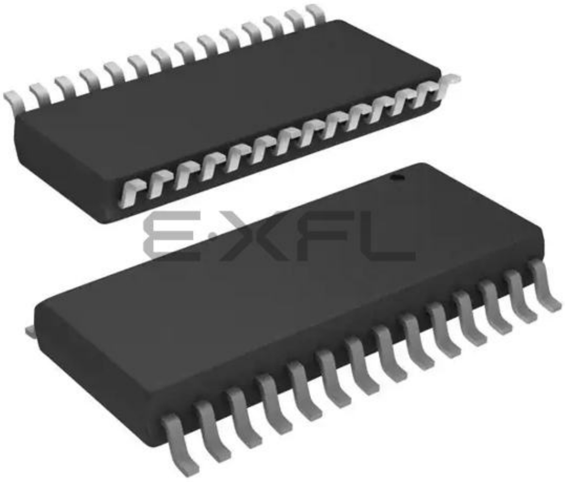
# Microchip Technology - PIC18LF2610T-I/SO Datasheet



## 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.

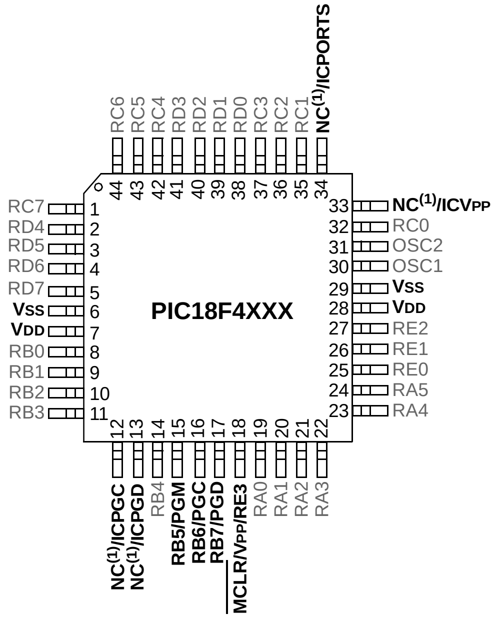
## Applications of "Embedded - Microcontrollers"

| **Details** | |
|---|---|
| Product Status | Obsolete |
| Core Processor | PIC |
| Core Size | 8-Bit |
| Speed | 40MHz |
| Connectivity | I²C, SPI, UART/USART |
| Peripherals | Brown-out Detect/Reset, HLVD, POR, PWM, WDT |
| Number of I/O | 25 |
| Program Memory Size | 64KB (32K x 16) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 3.8K x 8 |
| Voltage - Supply (Vcc/Vdd) | 2V ~ 5.5V |
| Data Converters | A/D 10x10b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 28-SOIC (0.295", 7.50mm Width) |
| Supplier Device Package | 28-SOIC |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/pic18lf2610t-i-so |

The following devices are included in 44-pin TQFP parts:

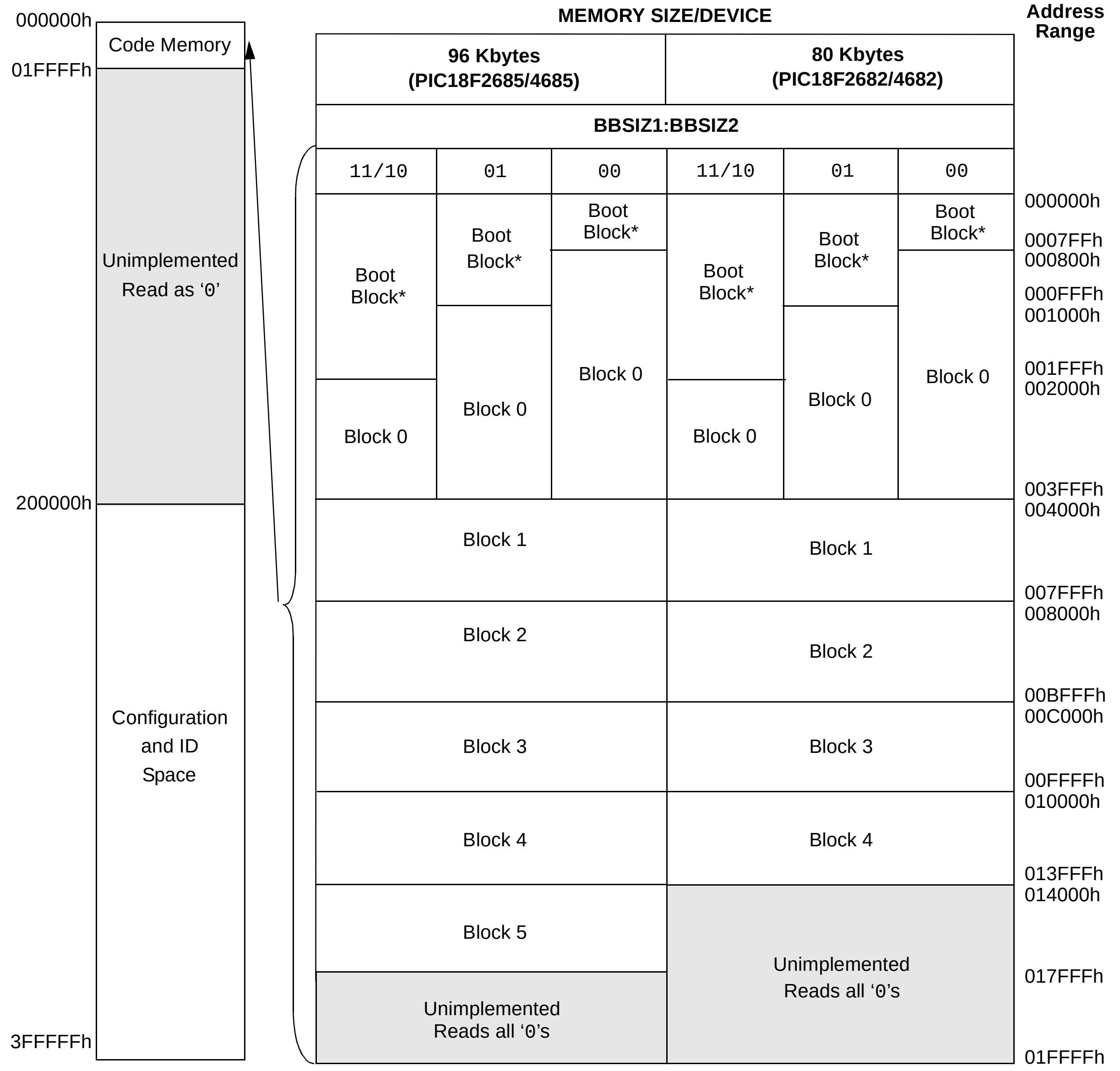
- PIC18F4221
- PIC18F4321
- PIC18F4410
- PIC18F4420
- PIC18F4423
- PIC18F4450
- PIC18F4455
- PIC18F4458
- PIC18F4480
- PIC18F4510
- PIC18F4520
- PIC18F4515
- PIC18F4523
- PIC18F4525
- PIC18F4550
- PIC18F4553
- PIC18F4580
- PIC18F4585
- PIC18F4610
- PIC18F4620
- PIC18F4680
- PIC18F4682
- PIC18F4685

**FIGURE 2-4: 44-PIN TQFP**



**Note 1:** These pins are NC (No Connect) for all devices listed above with the exception of the PIC18F4450, PIC18F4455, PIC18F4458 and the PIC18F4553 devices (see **Section 2.8 “Dedicated ICSP/ICD Port (44-Pin TQFP Only)”** for more information on programming these pins in these devices).

**FIGURE 2-7: MEMORY MAP AND THE CODE MEMORY SPACE FOR PIC18F2685/4685 AND PIC18F2682/4682 DEVICES**



**Note:** Sizes of memory areas are not to scale.

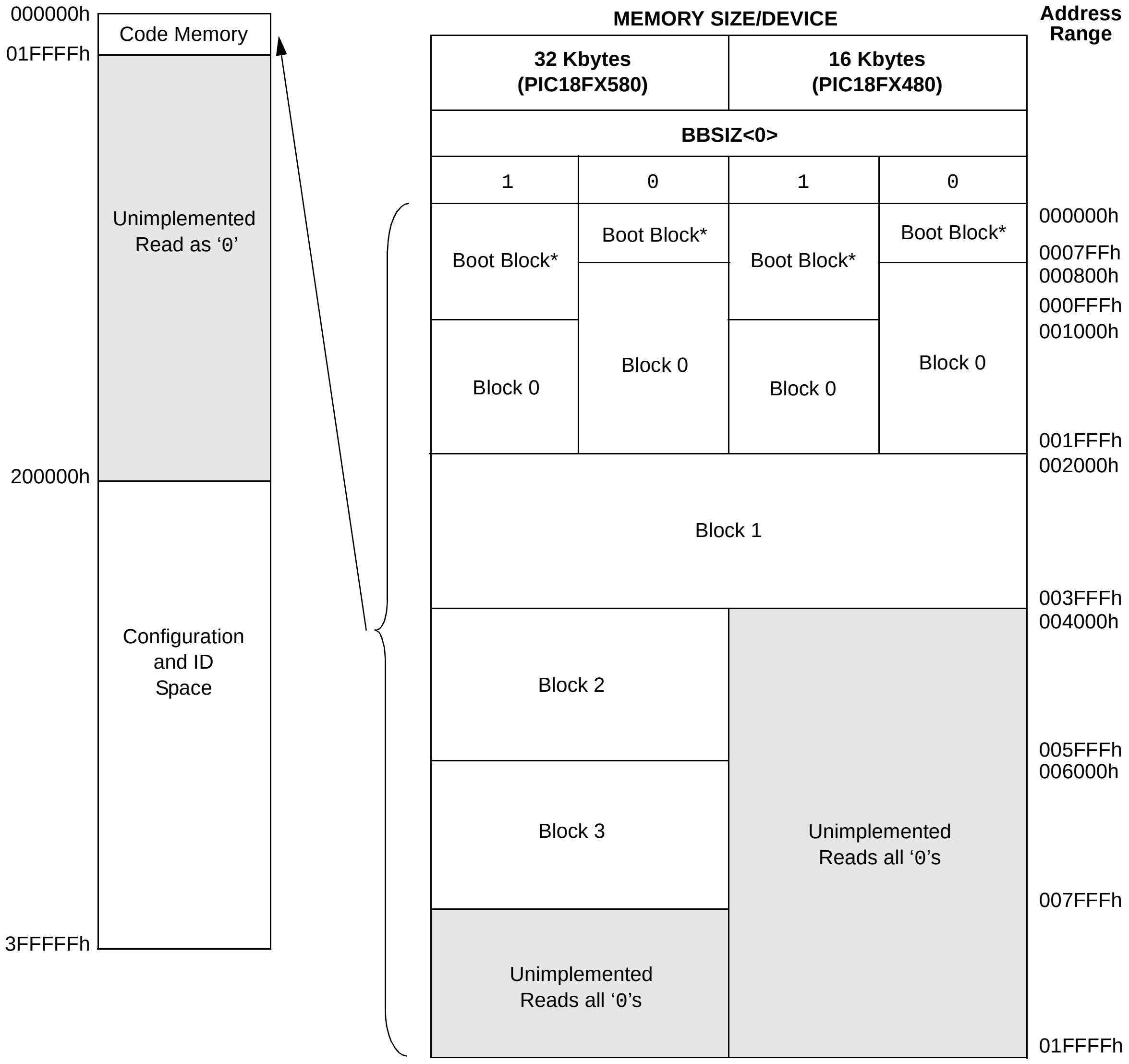
* Boot Block size is determined by the BBSIZ<1:2> bits in the CONFIG4L register.

For PIC18FX5X0/X5X3 devices, the code memory space extends from 000000h to 007FFFh (32 Kbytes) in four 8-Kbyte blocks. For PIC18FX4X5/X4X8 devices, the code memory space extends from 000000h to 005FFFh (24 Kbytes) in three 8-Kbyte blocks. Addresses, 000000h through 0007FFh, however, define a "Boot Block" region that is treated separately from Block 0. All of these blocks define code protection boundaries within the code memory space.

**TABLE 2-6: IMPLEMENTATION OF CODE MEMORY**

| Device | Code Memory Size (Bytes) |
|---|---|
| PIC18F2480 | 000000h-003FFFh (16K) |
| PIC18F4480 | |
| PIC18F2580 | 000000h-007FFFh (32K) |
| PIC18F4580 | |

**FIGURE 2-10: MEMORY MAP AND THE CODE MEMORY SPACE FOR PIC18F2480/2580/4480/4580 DEVICES**



**Note:** Sizes of memory areas are not to scale.

* Boot Block size is determined by the BBSIZ<0> bit in the CONFIG4L register.

For PIC18F2221/4221 devices, the code memory space extends from 0000h to 00FFFh (4 Kbytes) in one 4-Kbyte block. For PIC18F2321/4321 devices, the code memory space extends from 0000h to 01FFFh (8 Kbytes) in two 4-Kbyte blocks. Addresses, 0000h through 07FFh, however, define a variable "Boot Block" region that is treated separately from Block 0. All of these blocks define code protection boundaries within the code memory space.

## 2.5 Entering and Exiting High-Voltage ICSP Program/Verify Mode

As shown in Figure 2-14, the High-Voltage ICSP Program/Verify mode is entered by holding PGC and PGD low and then raising $\overline{\text{MCLR}}$/VPP/RE3 to VIHH (high voltage). Once in this mode, the code memory, data EEPROM (selected devices only, see **Section 3.3 "Data EEPROM Programming"**), ID locations and Configuration bits can be accessed and programmed in serial fashion. Figure 2-15 shows the exit sequence.

The sequence that enters the device into the Program/Verify mode places all unused I/Os in the high-impedance state.

**FIGURE 2-14: ENTERING HIGH-VOLTAGE PROGRAM/VERIFY MODE**

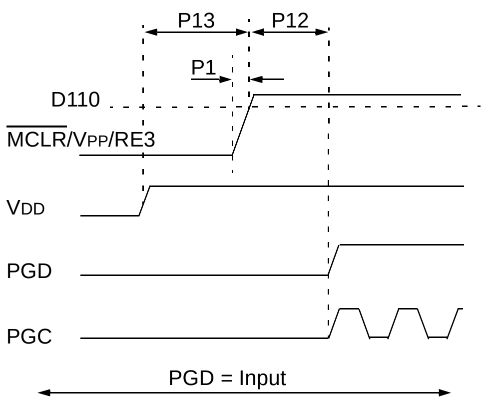
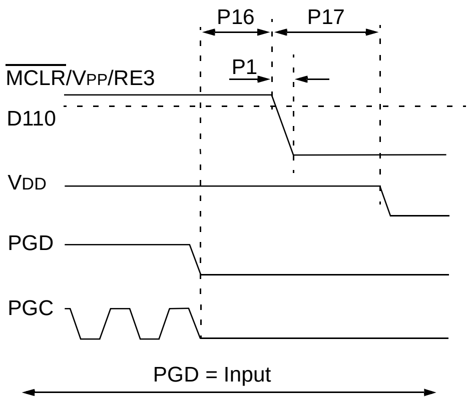


**FIGURE 2-15: EXITING HIGH-VOLTAGE PROGRAM/VERIFY MODE**



## 3.0 DEVICE PROGRAMMING

Programming includes the ability to erase or write the various memory regions within the device.

In all cases, except high-voltage ICSP Bulk Erase, the EECON1 register must be configured in order to operate on a particular memory region.

When using the EECON1 register to act on code memory, the EEPGD bit must be set (EECON1<7> = `1`) and the CFGS bit must be cleared (EECON1<6> = `0`). The WREN bit must be set (EECON1<2> = `1`) to enable writes of any sort (e.g., erases) and this must be done prior to initiating a write sequence. The FREE bit must be set (EECON1<4> = `1`) in order to erase the program space being pointed to by the Table Pointer. The erase or write sequence is initiated by setting the WR bit (EECON1<1> = `1`). It is strongly recommended that the WREN bit only be set immediately prior to a program erase.

## 3.1 ICSP Erase

### 3.1.1 HIGH-VOLTAGE ICSP BULK ERASE

Erasing code or data EEPROM is accomplished by configuring two Bulk Erase Control registers located at 3C0004h and 3C0005h. Code memory may be erased, portions at a time, or the user may erase the entire device in one action. Bulk Erase operations will also clear any code-protect settings associated with the memory block being erased. Erase options are detailed in Table 3-1. If data EEPROM is code-protected (CPD = `0`), the user must request an erase of data EEPROM (e.g., 0084h as shown in Table 3-1).

**TABLE 3-1: BULK ERASE OPTIONS**

| Description | Data (3C0005h:3C0004h) |
|---|---|
| Chip Erase | 3F8Fh |
| Erase Data EEPROM[1] | 0084h |
| Erase Boot Block | 0081h |
| Erase Configuration Bits | 0082h |
| Erase Code EEPROM Block 0 | 0180h |
| Erase Code EEPROM Block 1 | 0280h |
| Erase Code EEPROM Block 2 | 0480h |
| Erase Code EEPROM Block 3 | 0880h |
| Erase Code EEPROM Block 4 | 1080h |
| Erase Code EEPROM Block 5 | 2080h |

**Note 1:** Selected devices only, see **Section 3.3 "Data EEPROM Programming"**.

The actual Bulk Erase function is a self-timed operation. Once the erase has started (falling edge of the 4th PGC after the `NOP` command), serial execution will cease until the erase completes (Parameter P11). During this time, PGC may continue to toggle but PGD must be held low.

The code sequence to erase the entire device is shown in Table  and the flowchart is shown in Figure 3-1.

**Note:** A Bulk Erase is the only way to reprogram code-protect bits from an ON state to an OFF state.

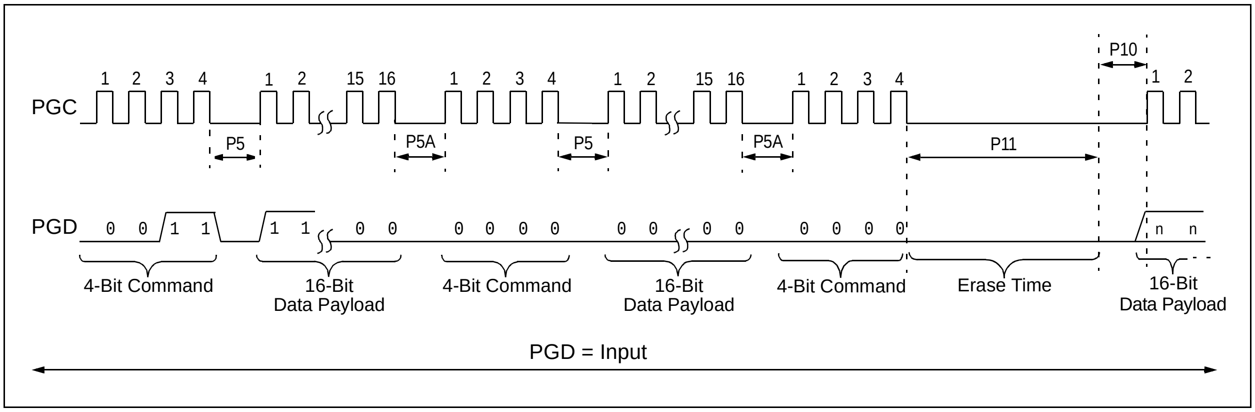
### 3.1.2 LOW-VOLTAGE ICSP BULK ERASE

When using low-voltage ICSP, the part must be supplied by the voltage specified in Parameter D111 if a Bulk Erase is to be executed. All other Bulk Erase details, as described above, apply.

If it is determined that a program memory erase must be performed at a supply voltage below the Bulk Erase limit, refer to the erase methodology described in **Section 3.1.3 "ICSP Row Erase"** and **Section 3.2.1 "Modifying Code Memory"**.

If it is determined that a data EEPROM erase (selected devices only, see **Section 3.3 "Data EEPROM Programming"**) must be performed at a supply voltage below the Bulk Erase limit, follow the methodology described in **Section 3.3 "Data EEPROM Programming"** and write '1's to the array.

**FIGURE 3-2: BULK ERASE TIMING**



### 3.1.3 ICSP ROW ERASE

Regardless of whether high or low-voltage ICSP is used, it is possible to erase one row (64 bytes of data), provided the block is not code or write-protected. Rows are located at static boundaries, beginning at program memory address, 000000h, extending to the internal program memory limit (see **Section 2.3 "Memory Maps"**).

The Row Erase duration is externally timed and is controlled by PGC. After the WR bit in EECON1 is set, a `NOP` is issued, where the 4th PGC is held high for the duration of the programming time, P9.

After PGC is brought low, the programming sequence is terminated. PGC must be held low for the time specified by Parameter P10 to allow high-voltage discharge of the memory array.

The code sequence to Row Erase a PIC18F2XXX/4XXX Family device is shown in Table 3-3. The flowchart, shown in Figure 3-3, depicts the logic necessary to completely erase a PIC18F2XXX/4XXX Family device. The timing diagram that details the Start Programming command and Parameters P9 and P10 is shown in Figure 3-5.

**Note:** The TBLPTR register can point to any byte within the row intended for erase.

### TABLE 3-3: ERASE CODE MEMORY CODE SEQUENCE

| 4-Bit Command | Data Payload | Core Instruction |
|---|---|---|
| Step 1: Direct access to code memory and enable writes. | | |
| `0000`<br>`0000`<br>`0000` | `8E A6`<br>`9C A6`<br>`84 A6` | `BSF EECON1, EEPGD`<br>`BCF EECON1, CFGS`<br>`BSF EECON1, WREN` |
| Step 2: Point to first row in code memory. | | |
| `0000`<br>`0000`<br>`0000` | `6A F8`<br>`6A F7`<br>`6A F6` | `CLRF TBLPTRU`<br>`CLRF TBLPTRH`<br>`CLRF TBLPTRL` |
| Step 3: Enable erase and erase single row. | | |
| `0000`<br>`0000`<br>`0000` | `88 A6`<br>`82 A6`<br>`00 00` | `BSF EECON1, FREE`<br>`BSF EECON1, WR`<br>`NOP – hold PGC high for time P9 and low for time P10.` |
| Step 4: Repeat Step 3, with the Address Pointer incremented by 64 until all rows are erased. | | |

### FIGURE 3-3: SINGLE ROW ERASE CODE MEMORY FLOW

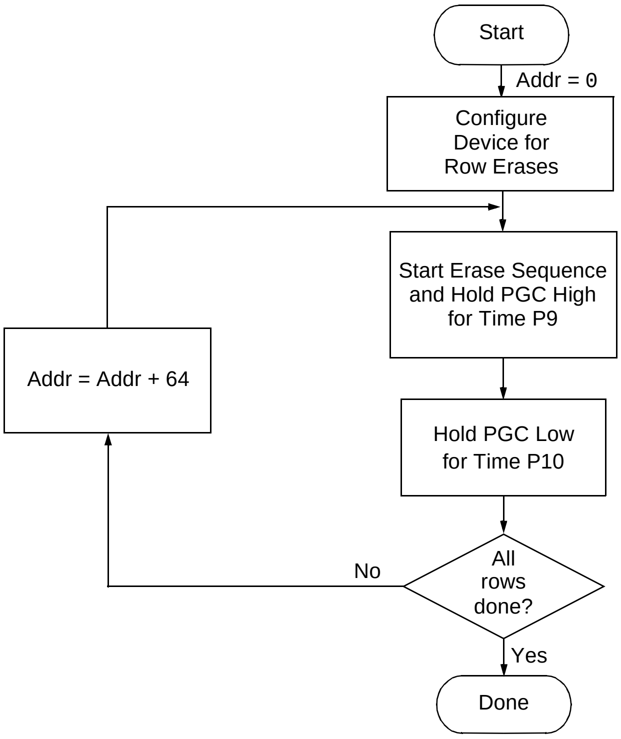


**TABLE 3-5: WRITE CODE MEMORY CODE SEQUENCE**

| 4-Bit Command | Data Payload | Core Instruction |
|---|---|---|
| Step 1: Direct access to code memory and enable writes. | | |
| 0000<br>0000 | 8E A6<br>9C A6 | BSF   EECON1, EEPGD<br>BCF   EECON1, CFGS |
| Step 2: Load write buffer. | | |
| 0000<br>0000<br>0000<br>0000<br>0000<br>0000 | 0E <Addr[21:16]><br>6E F8<br>0E <Addr[15:8]><br>6E F7<br>0E <Addr[7:0]><br>6E F6 | MOVLW <Addr[21:16]><br>MOVWF TBLPTRU<br>MOVLW <Addr[15:8]><br>MOVWF TBLPTRH<br>MOVLW <Addr[7:0]><br>MOVWF TBLPTRL |
| Step 3: Repeat for all but the last two bytes. | | |
| 1101 | <MSB><LSB> | Write 2 bytes and post-increment address by 2. |
| Step 4: Load write buffer for last two bytes. | | |
| 1111<br>0000 | <MSB><LSB><br>00 00 | Write 2 bytes and start programming.<br>NOP - hold PGC high for time P9 and low for time P10. |
| To continue writing data, repeat Steps 2 through 4, where the Address Pointer is incremented by 2 at each iteration of the loop. | | |

### 3.2.1 MODIFYING CODE MEMORY

The previous programming example assumed that the device had been Bulk Erased prior to programming (see **Section 3.1.1 “High-Voltage ICSP Bulk Erase”**). It may be the case, however, that the user wishes to modify only a section of an already programmed device.

The appropriate number of bytes required for the erase buffer must be read out of code memory (as described in **Section 4.2 “Verify Code Memory and ID Locations”**) and buffered. Modifications can be made on this buffer. Then, the block of code memory that was read out must be erased and rewritten with the modified data.

The WREN bit must be set if the WR bit in EECON1 is used to initiate a write sequence.

**TABLE 3-6: MODIFYING CODE MEMORY**

| 4-Bit Command | Data Payload | Core Instruction |
|---|---|---|
| Step 1: Direct access to code memory. | | |
| Step 2: Read and modify code memory (see **Section 4.1 “Read Code Memory, ID Locations and Configuration Bits”**). | | |
| 0000 | 8E A6 | BSF EECON1, EEPGD |
| 0000 | 9C A6 | BCF EECON1, CFGS |
| Step 3: Set the Table Pointer for the block to be erased. | | |
| 0000 | 0E <Addr[21:16]> | MOVLW <Addr[21:16]> |
| 0000 | 6E F8 | MOVWF TBLPTRU |
| 0000 | 0E <Addr[8:15]> | MOVLW <Addr[8:15]> |
| 0000 | 6E F7 | MOVWF TBLPTRH |
| 0000 | 0E <Addr[7:0]> | MOVLW <Addr[7:0]> |
| 0000 | 6E F6 | MOVWF TBLPTRL |
| Step 4: Enable memory writes and set up an erase. | | |
| 0000 | 84 A6 | BSF EECON1, WREN |
| 0000 | 88 A6 | BSF EECON1, FREE |
| Step 5: Initiate erase. | | |
| 0000 | 82 A6 | BSF EECON1, WR |
| 0000 | 00 00 | NOP - hold PGC high for time P9 and low for time P10. |
| Step 6: Load write buffer. The correct bytes will be selected based on the Table Pointer. | | |
| 0000 | 0E <Addr[21:16]> | MOVLW <Addr[21:16]> |
| 0000 | 6E F8 | MOVWF TBLPTRU |
| 0000 | 0E <Addr[8:15]> | MOVLW <Addr[8:15]> |
| 0000 | 6E F7 | MOVWF TBLPTRH |
| 0000 | 0E <Addr[7:0]> | MOVLW <Addr[7:0]> |
| 0000 | 6E F6 | MOVWF TBLPTRL |
| 1101 | <MSB><LSB> | Write 2 bytes and post-increment address by 2. |
| . | . | |
| . | . | Repeat as many times as necessary to fill the write buffer |
| . | . | |
| 1111 | <MSB><LSB> | Write 2 bytes and start programming. |
| 0000 | 00 00 | NOP - hold PGC high for time P9 and low for time P10. |
| To continue modifying data, repeat Steps 2 through 6, where the Address Pointer is incremented by the appropriate number of bytes (see Table 3-4) at each iteration of the loop. The write cycle must be repeated enough times to completely rewrite the contents of the erase buffer. | | |
| Step 7: Disable writes. | | |
| 0000 | 94 A6 | BCF EECON1, WREN |

### 3.3 Data EEPROM Programming

| **Note:** Data EEPROM programming is **not** available on the following devices: | |
|---|---|
| PIC18F2410 | PIC18F4410 |
| PIC18F2450 | PIC18F4450 |
| PIC18F2510 | PIC18F4510 |
| PIC18F2515 | PIC18F4515 |
| PIC18F2610 | PIC18F4610 |

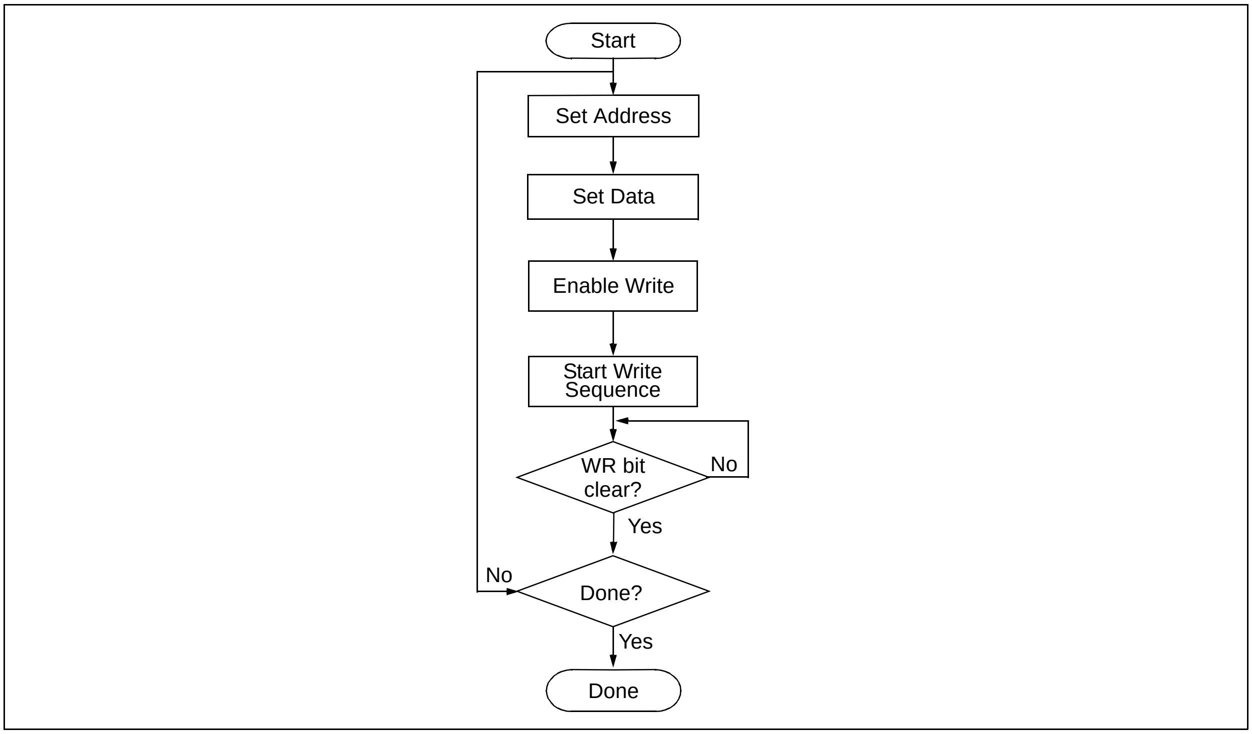
Data EEPROM is accessed one byte at a time via an Address Pointer (register pair: EEADRH:EEADR) and a data latch (EEDATA). Data EEPROM is written by loading EEADRH:EEADR with the desired memory location, EEDATA, with the data to be written and initiating a memory write by appropriately configuring the EECON1 register. A byte write automatically erases the location and writes the new data (erase-before-write).

When using the EECON1 register to perform a data EEPROM write, both the EEPGD and CFGS bits must be cleared (EECON1<7:6> = `00`). The WREN bit must be set (EECON1<2> = `1`) to enable writes of any sort and this must be done prior to initiating a write sequence. The write sequence is initiated by setting the WR bit (EECON1<1> = `1`).

The write begins on the falling edge of the 4th PGC after the WR bit is set. It ends when the WR bit is cleared by hardware.

After the programming sequence terminates, PGC must still be held low for the time specified by Parameter P10 to allow high-voltage discharge of the memory array.

**FIGURE 3-6: PROGRAM DATA FLOW**



## 3.4 ID Location Programming

The ID locations are programmed much like the code memory. The ID registers are mapped in addresses, 200000h through 200007h. These locations read out normally even after code protection.

**Note:** The user only needs to fill the first 8 bytes of the write buffer in order to write the ID locations.

Table 3-8 demonstrates the code sequence required to write the ID locations.

In order to modify the ID locations, refer to the methodology described in **Section 3.2.1 “Modifying Code Memory”**. As with code memory, the ID locations must be erased before being modified.

**TABLE 3-8: WRITE ID SEQUENCE**

| 4-Bit Command | Data Payload | Core Instruction |
|---|---|---|
| Step 1: Direct access to code memory and enable writes. | | |
| `0000` | `8E A6` | `BSF   EECON1, EEPGD` |
| `0000` | `9C A6` | `BCF   EECON1, CFGS` |
| Step 2: Load write buffer with 8 bytes and write. | | |
| `0000` | `0E 20` | `MOVLW 20h` |
| `0000` | `6E F8` | `MOVWF TBLPTRU` |
| `0000` | `0E 00` | `MOVLW 00h` |
| `0000` | `6E F7` | `MOVWF TBLPTRH` |
| `0000` | `0E 00` | `MOVLW 00h` |
| `0000` | `6E F6` | `MOVWF TBLPTRL` |
| `1101` | `<MSB><LSB>` | `Write 2 bytes and post-increment address by 2.` |
| `1101` | `<MSB><LSB>` | `Write 2 bytes and post-increment address by 2.` |
| `1101` | `<MSB><LSB>` | `Write 2 bytes and post-increment address by 2.` |
| `1111` | `<MSB><LSB>` | `Write 2 bytes and start programming.` |
| `0000` | `00 00` | `NOP - hold PGC high for time P9 and low for time P10.` |

## 3.5 Boot Block Programming

The code sequence detailed in Table 3-5 should be used, except that the address used in “Step 2” will be in the range of 000000h to 0007FFh.

## 3.6 Configuration Bits Programming

Unlike code memory, the Configuration bits are programmed a byte at a time. The Table Write, Begin Programming 4-bit command (‘`1111`’) is used, but only eight bits of the following 16-bit payload will be written. The LSB of the payload will be written to even addresses and the MSB will be written to odd addresses. The code sequence to program two consecutive configuration locations is shown in Table 3-9.

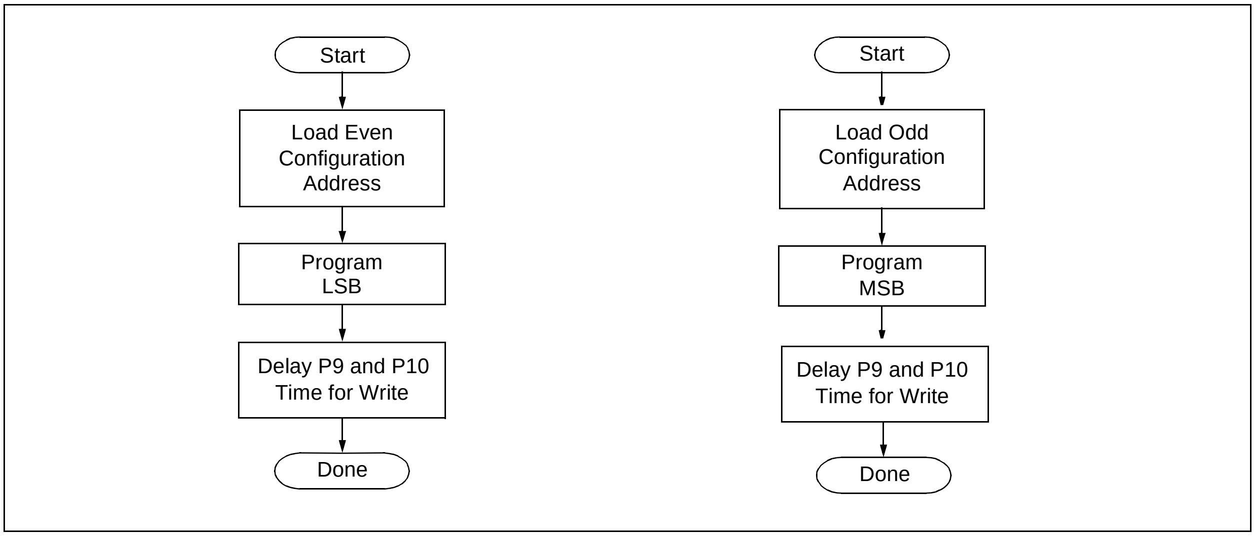
**Note:** The address must be explicitly written for each byte programmed. The addresses can not be incremented in this mode.

### TABLE 3-9: SET ADDRESS POINTER TO CONFIGURATION LOCATION

| 4-Bit Command | Data Payload | Core Instruction |
|---|---|---|
| Step 1: Enable writes and direct access to configuration memory. | | |
| `0000`<br>`0000` | `8E A6`<br>`8C A6` | `BSF   EECON1, EEPGD`<br>`BSF   EECON1, CFGS` |
| Step 2: Set Table Pointer for configuration byte to be written. Write even/odd addresses.[1] | | |
| `0000`<br>`0000`<br>`0000`<br>`0000`<br>`0000`<br>`0000`<br>`1111`<br>`0000`<br>`0000`<br>`0000`<br>`1111`<br>`0000` | `0E 30`<br>`6E F8`<br>`0E 00`<br>`6E F7`<br>`0E 00`<br>`6E F6`<br>`<MSB ignored><LSB>`<br>`00 00`<br>`0E 01`<br>`6E F6`<br>`<MSB><LSB ignored>`<br>`00 00` | `MOVLW 30h`<br>`MOVWF TBLPTRU`<br>`MOVLW 00h`<br>`MOVWF TBLPRTH`<br>`MOVLW 00h`<br>`MOVWF TBLPTRL`<br>`Load 2 bytes and start programming.`<br>`NOP - hold PGC high for time P9 and low for time P10.`<br>`MOVLW 01h`<br>`MOVWF TBLPTRL`<br>`Load 2 bytes and start programming.`<br>`NOP - hold PGC high for time P9 and low for time P10.` |

**Note 1:** Enabling the write protection of Configuration bits (WRTC = `0` in CONFIG6H) will prevent further writing of the Configuration bits. Always write all the Configuration bits before enabling the write protection for Configuration bits.

### FIGURE 3-8: CONFIGURATION PROGRAMMING FLOW



# 5.0 CONFIGURATION WORD

The PIC18F2XXX/4XXX Family devices have several Configuration Words. These bits can be set or cleared to select various device configurations. All other memory areas should be programmed and verified prior to setting the Configuration Words. These bits may be read out normally, even after read or code protection. See Table 5-1 for a list of Configuration bits and Device IDs, and Table 5-3 for the Configuration bit descriptions.

## 5.1 ID Locations

A user may store identification information (ID) in eight ID locations, mapped in 200000h:200007h. It is recommended that the Most Significant nibble of each ID be Fh. In doing so, if the user code inadvertently tries to execute from the ID space, the ID data will execute as a `NOP`.

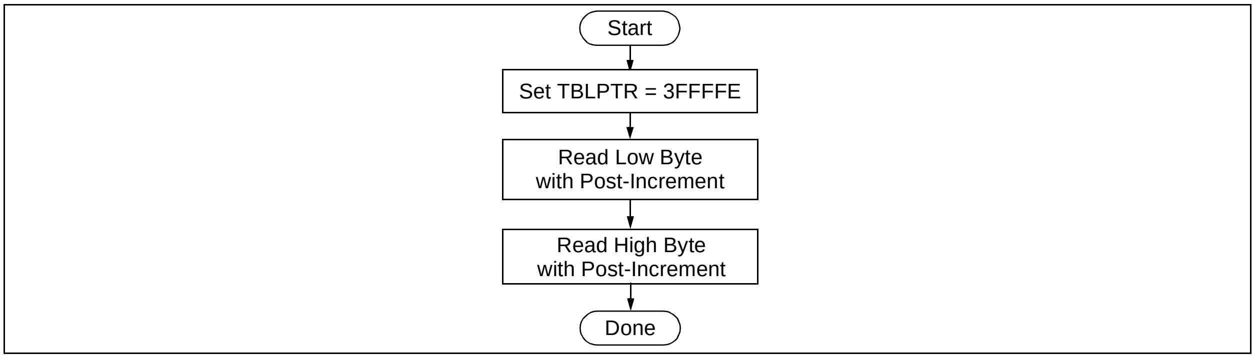
## 5.2 Device ID Word

The Device ID Word for the PIC18F2XXX/4XXX Family devices is located at 3FFFFEh:3FFFFFh. These bits may be used by the programmer to identify what device type is being programmed and read out normally, even after code or read protection.

In some cases, devices may share the same DEVID values. In such cases, the Most Significant bit of the device revision, REV4 (DEVID1<4>), will need to be examined to completely determine the device being accessed.

See Table 5-2 for a complete list of Device ID values.

**FIGURE 5-1: READ DEVICE ID WORD FLOW**



### TABLE 5-1: CONFIGURATION BITS AND DEVICE IDS

| File Name | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Default/ Unprogrammed Value |
|---|---|---|---|---|---|---|---|---|---|---|
| 300000h(1,8) | CONFIG1L | — | — | USBDIV | CPUDIV1 | CPUDIV0 | PLLDIV2 | PLLDIV1 | PLLDIV0 | `--00 0000` |
| 300001h | CONFIG1H | IESO | FCMEN | — | — | FOSC3 | FOSC2 | FOSC1 | FOSC0 | `00-- 0111`<br>`00-- 0101`(1,8) |
| 300002h | CONFIG2L | — | — | —<br>VREGEN(1,8) | BORV1 | BORV0 | BOREN1 | BOREN0 | $\overline{\text{PWRTEN}}$ | `---1 1111`<br>`--01 1111`(1,8) |
| 300003h | CONFIG2H | — | — | — | WDTPS3 | WDTPS2 | WDTPS1 | WDTPS0 | WDTEN | `---1 1111` |
| 300005h | CONFIG3H | MCLRE | — | — | — | — | LPT1OSC | PBADEN | CCP2MX(7)<br>— | `1--- -011`(7)<br>`1--- -01-` |
| 300006h | CONFIG4L | $\overline{\text{DEBUG}}$ | XINST | ICPRT(1)<br>BBSIZ1<br>—<br>ICPRT(8)<br>BBSIZ1(2) | —<br>BBSIZ0<br>BBSIZ(3)<br>—<br>BBSIZ2(2) | —<br>—<br>—<br>BBSIZ(8)<br>— | LVP | — | STVREN | `100- -1-1`(1)<br>`1000 -1-1`<br>`10-0 -1-1`(3)<br>`100- 01-1`(8)<br>`1000 -1-1`(2) |
| 300008h | CONFIG5L | — | — | CP5(10) | CP4(9) | CP3(4) | CP2(4) | CP1 | CP0 | `--11 1111` |
| 300009h | CONFIG5H | CPD | CPB | — | — | — | — | — | — | `11-- ----` |
| 30000Ah | CONFIG6L | — | — | WRT5(10) | WRT4(9) | WRT3(4) | WRT2(4) | WRT1 | WRT0 | `--11 1111` |
| 30000Bh | CONFIG6H | WRTD | WRTB | WRTC(5) | — | — | — | — | — | `111- ----` |
| 30000Ch | CONFIG7L | — | — | EBTR5(10) | EBTR4(9) | EBTR3(4) | EBTR2(4) | EBTR1 | EBTR0 | `--11 1111` |
| 30000Dh | CONFIG7H | — | EBTRB | — | — | — | — | — | — | `-1-- ----` |
| 3FFFFEh | DEVID1(6) | DEV2 | DEV1 | DEV0 | REV4 | REV3 | REV2 | REV1 | REV0 | See Table 5-2 |
| 3FFFFFh | DEVID2(6) | DEV10 | DEV9 | DEV8 | DEV7 | DEV6 | DEV5 | DEV4 | DEV3 | See Table 5-2 |

**Legend:** - = unimplemented. Shaded cells are unimplemented, read as ‘`0`’.

**Note 1:** Implemented only on PIC18F2455/2550/4455/4550 and PIC18F2458/2553/4458/4553 devices.

**2:** Implemented on PIC18F2585/2680/4585/4680, PIC18F2682/2685 and PIC18F4682/4685 devices only.

**3:** Implemented on PIC18F2480/2580/4480/4580 devices only.

**4:** These bits are only implemented on specific devices based on available memory. Refer to **Section 2.3 “Memory Maps”**.

**5:** In PIC18F2480/2580/4480/4580 devices, this bit is read-only in Normal Execution mode; it can be written only in Program mode.

**6:** DEVID registers are read-only and cannot be programmed by the user.

**7:** Implemented on all devices with the exception of the PIC18FXX8X and PIC18F2450/4450 devices.

**8:** Implemented on PIC18F2450/4450 devices only.

**9:** Implemented on PIC18F2682/2685 and PIC18F4682/4685 devices only.

**10:** Implemented on PIC18F2685/4685 devices only.

### TABLE 5-2: DEVICE ID VALUES

| Device | Device ID Value | |
|---|---|---|
| | DEVID2 | DEVID1 |
| PIC18F2221 | 21h | 011x xxxx |
| PIC18F2321 | 21h | 001x xxxx |
| PIC18F2410 | 11h | 011x xxxx |
| PIC18F2420 | 11h | 010x xxxx(1) |
| PIC18F2423 | 11h | 010x xxxx(2) |
| PIC18F2450 | 24h | 001x xxxx |
| PIC18F2455 | 12h | 011x xxxx |
| PIC18F2458 | 2Ah | 011x xxxx |
| PIC18F2480 | 1Ah | 111x xxxx |
| PIC18F2510 | 11h | 001x xxxx |
| PIC18F2515 | 0Ch | 111x xxxx |
| PIC18F2520 | 11h | 000x xxxx(1) |
| PIC18F2523 | 11h | 000x xxxx(2) |
| PIC18F2525 | 0Ch | 110x xxxx |
| PIC18F2550 | 12h | 010x xxxx |
| PIC18F2553 | 2Ah | 010x xxxx |
| PIC18F2580 | 1Ah | 110x xxxx |
| PIC18F2585 | 0Eh | 111x xxxx |
| PIC18F2610 | 0Ch | 101x xxxx |
| PIC18F2620 | 0Ch | 100x xxxx |
| PIC18F2680 | 0Eh | 110x xxxx |
| PIC18F2682 | 27h | 000x xxxx |
| PIC18F2685 | 27h | 001x xxxx |
| PIC18F4221 | 21h | 010x xxxx |
| PIC18F4321 | 21h | 000x xxxx |
| PIC18F4410 | 10h | 111x xxxx |
| PIC18F4420 | 10h | 110x xxxx(1) |
| PIC18F4423 | 10h | 110x xxxx(2) |
| PIC18F4450 | 24h | 000x xxxx |
| PIC18F4455 | 12h | 001x xxxx |
| PIC18F4458 | 2Ah | 001x xxxx |
| PIC18F4480 | 1Ah | 101x xxxx |
| PIC18F4510 | 10h | 101x xxxx |
| PIC18F4515 | 0Ch | 011x xxxx |
| PIC18F4520 | 10h | 100x xxxx(1) |
| PIC18F4523 | 10h | 100x xxxx(2) |
| PIC18F4525 | 0Ch | 010x xxxx |
| PIC18F4550 | 12h | 000x xxxx |
| PIC18F4553 | 2Ah | 000x xxxx |
| PIC18F4580 | 1Ah | 100x xxxx |

**Legend:** The ‘x’s in DEVID1 contain the device revision code.

**Note 1:** DEVID1 bit 4 is used to determine the device type (REV4 = `0`).

**2:** DEVID1 bit 4 is used to determine the device type (REV4 = `1`).

**TABLE 5-3: PIC18F2XXX/4XXX FAMILY BIT DESCRIPTIONS (CONTINUED)**

| Bit Name | Configuration Words | Description |
|---|---|---|
| PLLDIV<2:0> | CONFIG1L | Oscillator Selection bits<br>**(PIC18F2455/2550/4455/4550, PIC18F2458/2553/4458/4553 and PIC18F2450/4450 devices only)**<br>Divider must be selected to provide a 4 MHz input into the 96 MHz PLL:<br>111 = Oscillator divided by 12 (48 MHz input)<br>110 = Oscillator divided by 10 (40 MHz input)<br>101 = Oscillator divided by 6 (24 MHz input)<br>100 = Oscillator divided by 5 (20 MHz input)<br>011 = Oscillator divided by 4 (16 MHz input)<br>010 = Oscillator divided by 3 (12 MHz input)<br>001 = Oscillator divided by 2 (8 MHz input)<br>000 = No divide – oscillator used directly (4 MHz input) |
| VREGEN | CONFIG2L | USB Voltage Regulator Enable bit<br>**(PIC18F2455/2550/4455/4550, PIC18F2458/2553/4458/4553 and PIC18F2450/4450 devices only)**<br>1 = USB voltage regulator is enabled<br>0 = USB voltage regulator is disabled |
| BORV<1:0> | CONFIG2L | Brown-out Reset Voltage bits<br>11 = $V_{BOR}$ is set to 2.0V<br>10 = $V_{BOR}$ is set to 2.7V<br>01 = $V_{BOR}$ is set to 4.2V<br>00 = $V_{BOR}$ is set to 4.5V |
| BOREN<1:0> | CONFIG2L | Brown-out Reset Enable bits<br>11 = Brown-out Reset is enabled in hardware only (SBOREN is disabled)<br>10 = Brown-out Reset is enabled in hardware only and disabled in Sleep mode SBOREN is disabled)<br>01 = Brown-out Reset is enabled and controlled by software (SBOREN is enabled)<br>00 = Brown-out Reset is disabled in hardware and software |
| PWRTEN | CONFIG2L | Power-up Timer Enable bit<br>1 = PWRT is disabled<br>0 = PWRT is enabled |
| WDPS<3:0> | CONFIG2H | Watchdog Timer Postscaler Select bits<br>1111 = 1:32,768<br>1110 = 1:16,384<br>1101 = 1:8,192<br>1100 = 1:4,096<br>1011 = 1:2,048<br>1010 = 1:1,024<br>1001 = 1:512<br>1000 = 1:256<br>0111 = 1:128<br>0110 = 1:64<br>0101 = 1:32<br>0100 = 1:16<br>0011 = 1:8<br>0010 = 1:4<br>0001 = 1:2<br>0000 = 1:1 |

**Note 1:** The BBSIZ bits, BBSIZ<1:0> and BBSIZ<2:1> bits, cannot be changed once any of the following code-protect bits are enabled: CPB or CP0, WRTB or WRT0, EBTRB or EBTR0.

**2:** Not available in PIC18FXX8X and PIC18F2450/4450 devices.

**TABLE 5-3: PIC18F2XXX/4XXX FAMILY BIT DESCRIPTIONS (CONTINUED)**

| Bit Name | Configuration Words | Description |
|---|---|---|
| EBTR0 | CONFIG7L | Table Read Protection bit (Block 0 code memory area)<br>`1` = Block 0 is not protected from Table Reads executed in other blocks<br>`0` = Block 0 is protected from Table Reads executed in other blocks |
| EBTRB | CONFIG7H | Table Read Protection bit (Boot Block memory area)<br>`1` = Boot Block is not protected from Table Reads executed in other blocks<br>`0` = Boot Block is protected from Table Reads executed in other blocks |
| DEV<10:3> | DEVID2 | Device ID bits<br>These bits are used with the DEV<2:0> bits in the DEVID1 register to identify part number. |
| DEV<2:0> | DEVID1 | Device ID bits<br>These bits are used with the DEV<10:3> bits in the DEVID2 register to identify part number. |
| REV<4:0> | DEVID1 | Revision ID bits<br>These bits are used to indicate the revision of the device. The REV4 bit is sometimes used to fully specify the device type. |

**Note 1:** The BBSIZ bits, BBSIZ<1:0> and BBSIZ<2:1> bits, cannot be changed once any of the following code-protect bits are enabled: CPB or CP0, WRTB or WRT0, EBTRB or EBTR0.

**2:** Not available in PIC18FXX8X and PIC18F2450/4450 devices.

### 5.6.3 ID LOCATIONS

Normally, the contents of these locations are defined by the user, but MPLAB® IDE provides the option of writing the device's unprotected 16-bit checksum in the 16 Most Significant bits of the ID locations (see MPLAB IDE Configure/ID Memory" menu). The lower 16 bits are not used and remain clear. This is the sum of all program memory contents and Configuration Words (appropriately masked) before any code protection is enabled.

If the user elects to define the contents of the ID locations, nothing about protected blocks can be known. If the user uses the preprotected checksum, provided by MPLAB IDE, an indirect characteristic of the programmed code is provided.

### 5.6.4 CODE PROTECTION

Blocks that are code-protected read back as all '`0`'s and have no effect on checksum calculations. If any block is code-protected, then the contents of the ID locations are included in the checksum calculation.

All Configuration Words and the ID locations can always be read out normally, even when the device is fully code-protected. Checking the code protection settings in Configuration Words can direct which, if any, of the program memory blocks can be read, and if the ID locations should be used for checksum calculations.

**TABLE 5-5: CONFIGURATION WORD MASKS FOR COMPUTING CHECKSUMS (CONTINUED)**

| Device | Configuration Word (CONFIGxx) | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 1L | 1H | 2L | 2H | 3L | 3H | 4L | 4H | 5L | 5H | 6L | 6H | 7L | 7H |
| | Address (30000xh) | | | | | | | | | | | | | |
| | 0h | 1h | 2h | 3h | 4h | 5h | 6h | 7h | 8h | 9h | Ah | Bh | Ch | Dh |
| PIC18F4620 | 00 | CF | 1F | 1F | 00 | 87 | C5 | 00 | 0F | C0 | 0F | E0 | 0F | 40 |
| PIC18F4680 | 00 | CF | 1F | 1F | 00 | 86 | C5 | 00 | 0F | C0 | 0F | E0 | 0F | 40 |
| PIC18F4682 | 00 | CF | 1F | 1F | 00 | 86 | C5 | 00 | 3F | C0 | 3F | E0 | 3F | 40 |
| PIC18F4685 | 00 | CF | 1F | 1F | 00 | 86 | C5 | 00 | 3F | C0 | 3F | E0 | 3F | 40 |

**Legend:** Shaded cells are unimplemented.

## 6.0 AC/DC CHARACTERISTICS TIMING REQUIREMENTS FOR PROGRAM/VERIFY TEST MODE (CONTINUED)

**Standard Operating Conditions**
Operating Temperature: 25°C is recommended

| Param No. | Sym | Characteristic | Min | Max | Units | Conditions |
|---|---|---|---|---|---|---|
| P11A | TDRWT | Data Write Polling Time | 4 | — | ms | |
| P12 | THLD2 | Input Data Hold Time from $\overline{\text{MCLR}}$/VPP/RE3 ↑ | 2 | — | µs | |
| P13 | TSET2 | VDD ↑ Setup Time to $\overline{\text{MCLR}}$/VPP/RE3 ↑ | 100 | — | ns | **(Note 2)** |
| P14 | TVALID | Data Out Valid from PGC ↑ | 10 | — | ns | |
| P15 | TSET3 | PGM ↑ Setup Time to $\overline{\text{MCLR}}$/VPP/RE3 ↑ | 2 | — | µs | **(Note 2)** |
| P16 | TDLY8 | Delay Between Last PGC ↓ and $\overline{\text{MCLR}}$/VPP/RE3 ↓ | 0 | — | s | |
| P17 | THLD3 | $\overline{\text{MCLR}}$/VPP/RE3 ↓ to VDD ↓ | — | 100 | ns | |
| P18 | THLD4 | $\overline{\text{MCLR}}$/VPP/RE3 ↓ to PGM ↓ | 0 | — | s | |

**Note 1:** Do not allow excess time when transitioning $\overline{\text{MCLR}}$ between VIL and VIHH. This can cause spurious program executions to occur. The maximum transition time is:

1 TCY + TPWRT (if enabled) + 1024 TOSC (for LP, HS, HS/PLL and XT modes only) +
2 ms (for HS/PLL mode only) + 1.5 µs (for EC mode only)

where TCY is the instruction cycle time, TPWRT is the Power-up Timer period and TOSC is the oscillator period. For specific values, refer to the Electrical Characteristics section of the device data sheet for the particular device.

**2:** When ICPRT = 1, this specification also applies to ICVPP.

**3:** At 0°C-50°C.