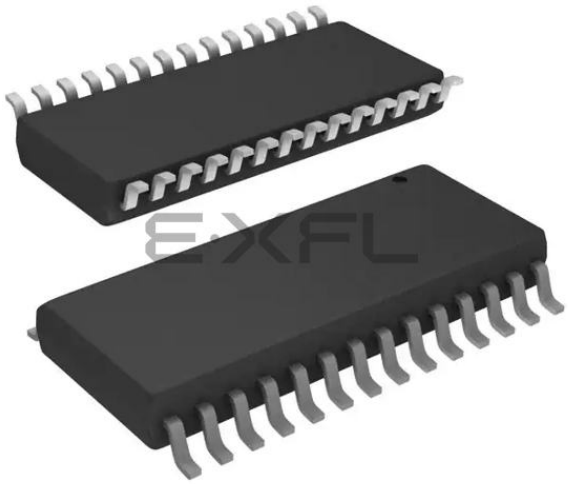


**Welcome to E-XFL.COM**

## 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.

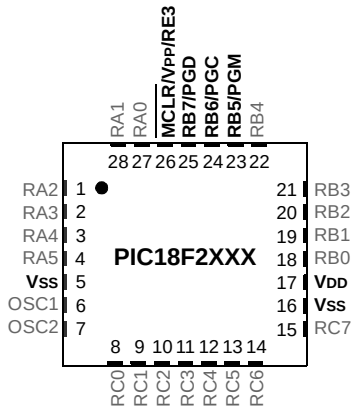
## Applications of "Embedded - Microcontrollers"

| Details | |
|---|---|
| Product Status | 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 | 48KB (24K x 16) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 3.8K x 8 |
| Voltage - Supply (Vcc/Vdd) | 4.2V ~ 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/pic18f2515t-i-so |

The following devices are included in 28-pin QFN parts:

- PIC18F2221
- PIC18F2321
- PIC18F2410
- PIC18F2420
- PIC18F2423
- PIC18F2450
- PIC18F2480
- PIC18F2510
- PIC18F2520
- PIC18F2523
- PIC18F2580
- PIC18F2682
- PIC18F2685

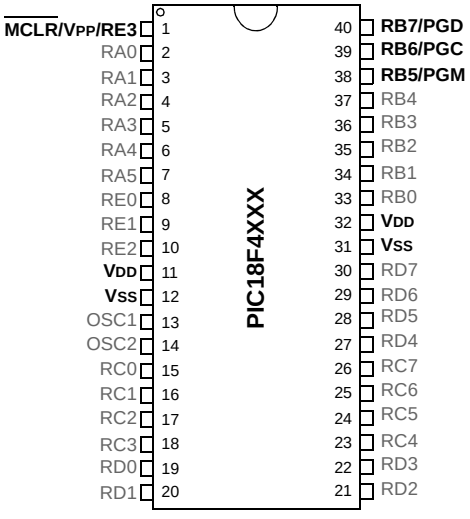
**FIGURE 2-2: 28-Pin QFN**




The following devices are included in 40-pin PDIP parts:

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

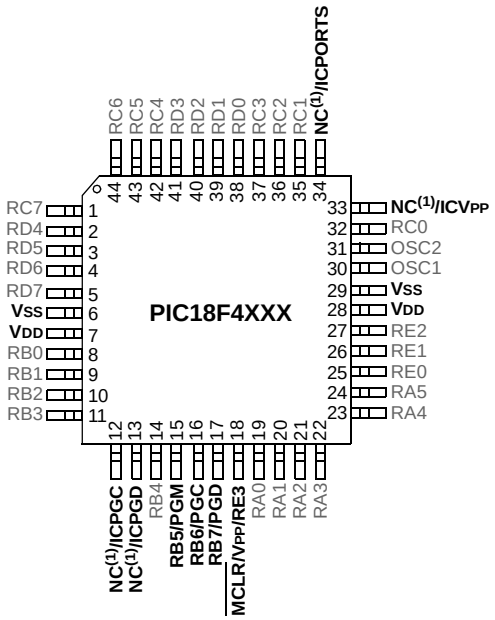
**FIGURE 2-3: 40-Pin PDIP**



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).

For PIC18F2685/4685 devices, the code memory space extends from 0000h to 017FFFh (96 Kbytes) in five 16-Kbyte blocks. For PIC18F2682/4682 devices, the code memory space extends from 0000h to 0013FFFh (80 Kbytes) in four 16-Kbyte blocks. Addresses, 0000h through 0FFFh, 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.

The size of the Boot Block in PIC18F2685/4685 and PIC18F2682/4682 devices can be configured as 1, 2 or 4K words (see Figure 2-7). This is done through the BBSIZ<2:1> bits in the Configuration register, CONFIG4L. It is important to note that increasing the size of the Boot Block decreases the size of Block 0.

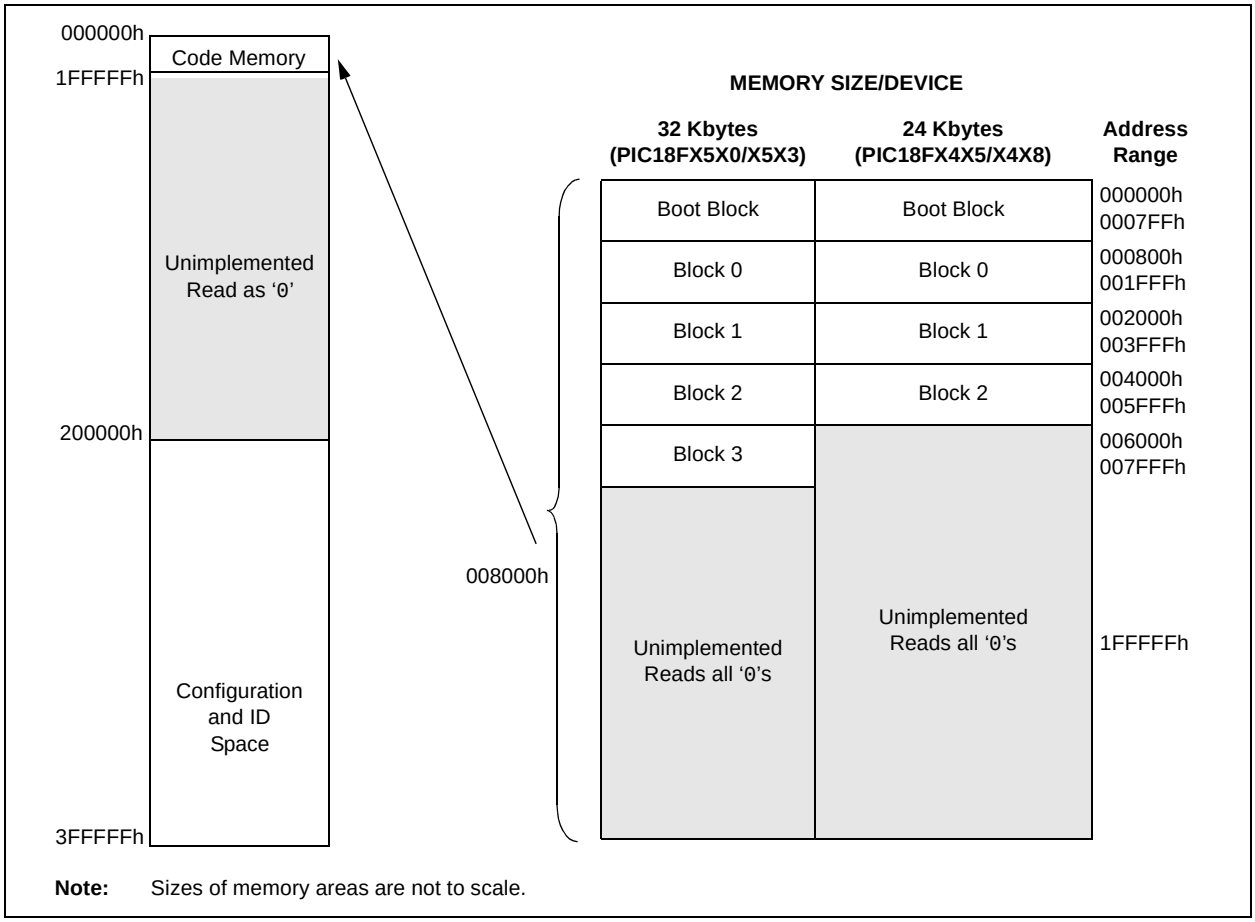
**TABLE 2-3: IMPLEMENTATION OF CODE MEMORY**

| Device | Code Memory Size (Bytes) |
|---|---|
| PIC18F2682 | 000000h-013FFFh (80K) |
| PIC18F4682 | |
| PIC18F2685 | 000000h-017FFFh (96K) |
| PIC18F4685 | |

**TABLE 2-4: IMPLEMENTATION OF CODE MEMORY**

| Device | Code Memory Size (Bytes) |
|---|---|
| PIC18F2455 | 000000h-005FFFh (24K) |
| PIC18F2458 | |
| PIC18F4455 | |
| PIC18F4458 | |
| PIC18F2510 | 000000h-007FFFh (32K) |
| PIC18F2520 | |
| PIC18F2523 | |
| PIC18F2550 | |
| PIC18F2553 | |
| PIC18F4510 | |
| PIC18F4520 | |
| PIC18F4523 | |
| PIC18F4550 | |
| PIC18F4553 | |

**FIGURE 2-8: MEMORY MAP AND THE CODE MEMORY SPACE FOR PIC18FX4X5/X4X8/X5X0/X5X3 DEVICES**




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

For PIC18FX4X0/X4X3 devices, the code memory space extends from 000000h to 003FFFh (16 Kbytes) in two 8-Kbyte blocks. Addresses, 000000h through 0003FFh, 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.

In addition to the code memory space, there are three blocks that are accessible to the user through Table Reads and Table Writes. Their locations in the memory map are shown in Figure 2-12.

Users may store identification information (ID) in eight ID registers. These ID registers are mapped in addresses, 200000h through 200007h. The ID locations read out normally, even after code protection is applied.

Locations, 300000h through 30000Dh, are reserved for the Configuration bits. These bits select various device options and are described in **Section 5.0 "Configuration Word"**. These Configuration bits read out normally, even after code protection.

Locations, 3FFFFEh and 3FFFFFh, are reserved for the Device ID bits. These bits may be used by the programmer to identify what device type is being programmed and are described in **Section 5.0 "Configuration Word"**. These Device ID bits read out normally, even after code protection.

### 2.3.1 MEMORY ADDRESS POINTER

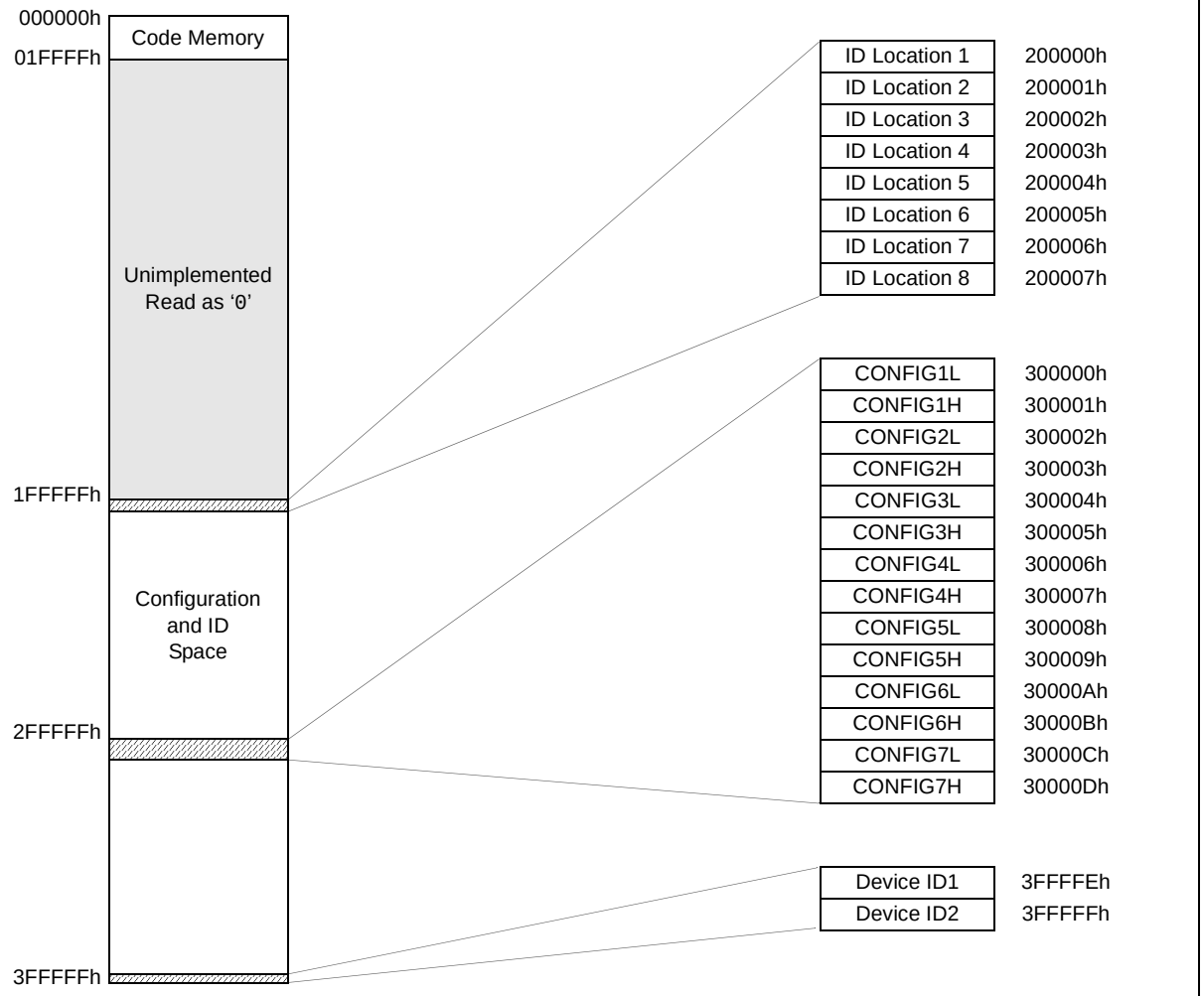
Memory in the address space, 0000000h to 3FFFFFh, is addressed via the Table Pointer register, which is comprised of three pointer registers:

- TBLPTRU at RAM address 0FF8h
- TBLPTRH at RAM address 0FF7h
- TBLPTRL at RAM address 0FF6h

| TBLPTRU | TBLPTRH | TBLPTRL |
|---|---|---|
| Addr[21:16] | Addr[15:8] | Addr[7:0] |

The 4-bit command, '0000' (core instruction), is used to load the Table Pointer prior to using many read or write operations.

**FIGURE 2-12: CONFIGURATION AND ID LOCATIONS FOR PIC18F2XXX/4XXX FAMILY DEVICES**




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

## 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.

**TABLE 3-2: BULK ERASE COMMAND SEQUENCE**

| 4-Bit Command | Data Payload | Core Instruction |
|---|---|---|
| 0000 | 0E 3C | MOVLW 3Ch |
| 0000 | 6E F8 | MOVWF TBLPTRU |
| 0000 | 0E 00 | MOVLW 00h |
| 0000 | 6E F7 | MOVWF TBLPTRH |
| 0000 | 0E 05 | MOVLW 05h |
| 0000 | 6E F6 | MOVWF TBLPTRL |
| 1100 | 3F 3F | Write 3F3Fh to 3C0005h |
| 0000 | 0E 3C | MOVLW 3Ch |
| 0000 | 6E F8 | MOVWF TBLPTRU |
| 0000 | 0E 00 | MOVLW 00h |
| 0000 | 6E F7 | MOVWF TBLPTRH |
| 0000 | 0E 04 | MOVLW 04h |
| 0000 | 6E F6 | MOVWF TBLPTRL |
| 1100 | 8F 8F | Write 8F8Fh TO 3C0004h to erase entire device. |
| | | NOP |
| | | Hold PGD low until erase completes. |
| 0000 | 00 00 | |
| 0000 | 00 00 | |

**FIGURE 3-1: BULK ERASE FLOW**

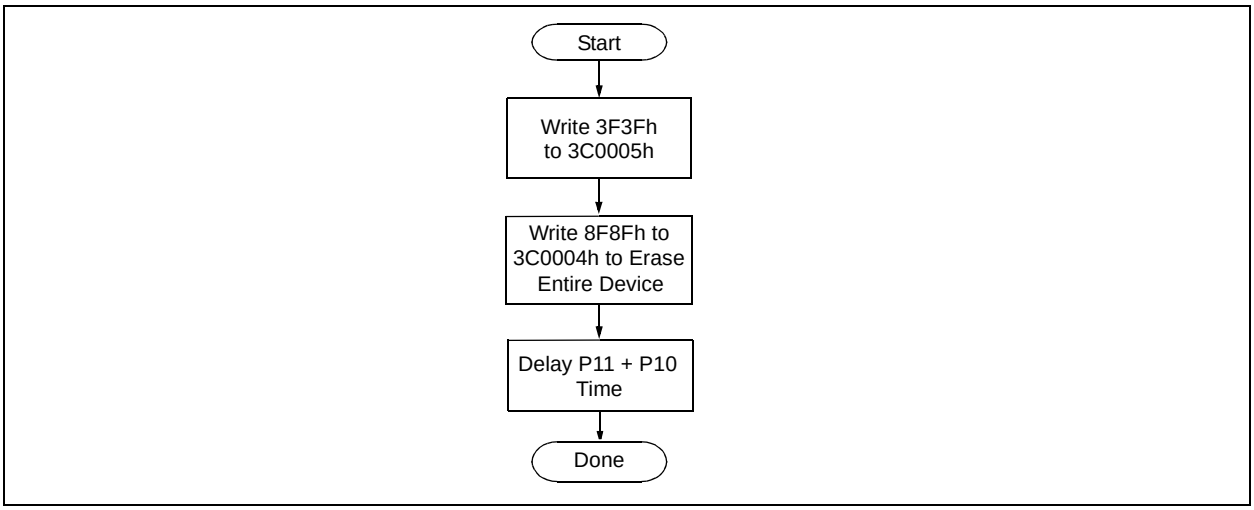


**FIGURE 3-4: PROGRAM CODE MEMORY FLOW**

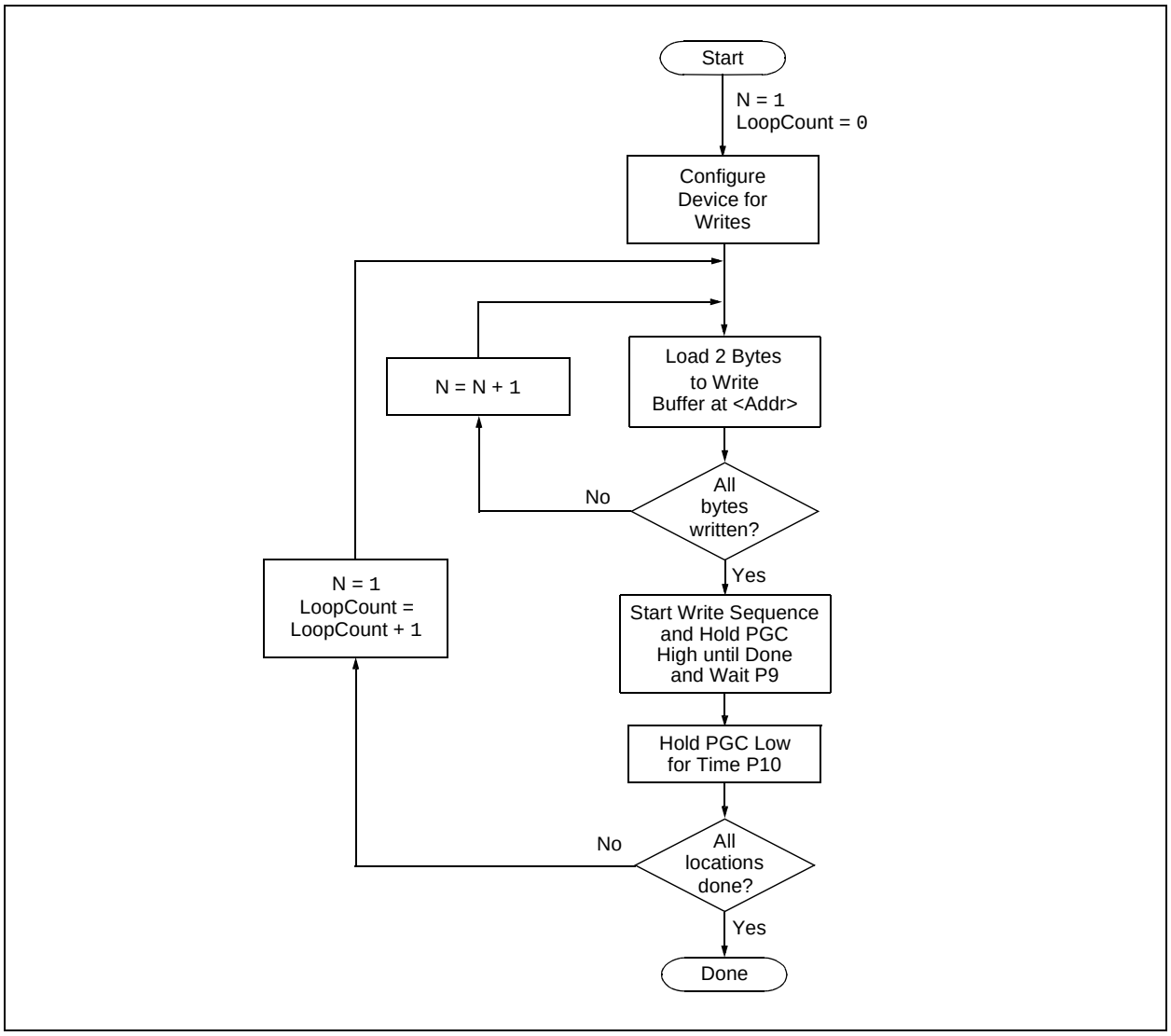



**FIGURE 3-5: TABLE WRITE AND START PROGRAMMING INSTRUCTION TIMING (1111)**

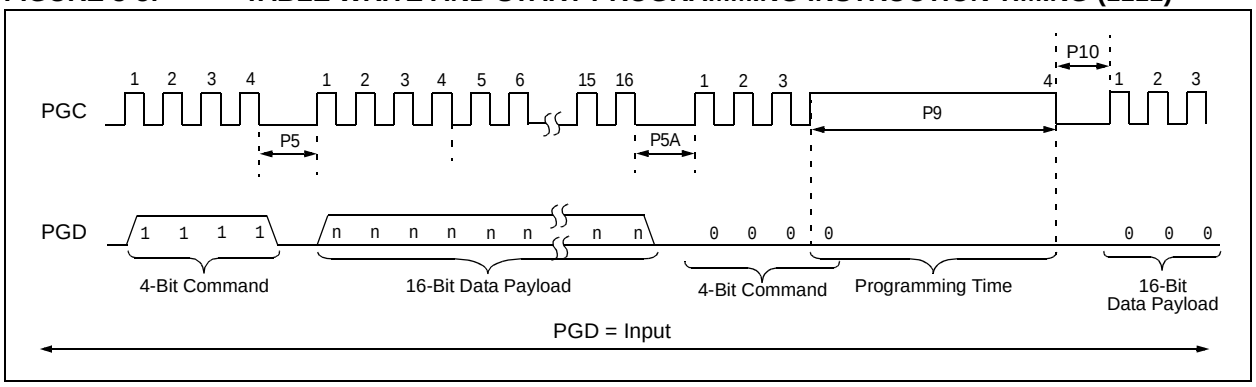
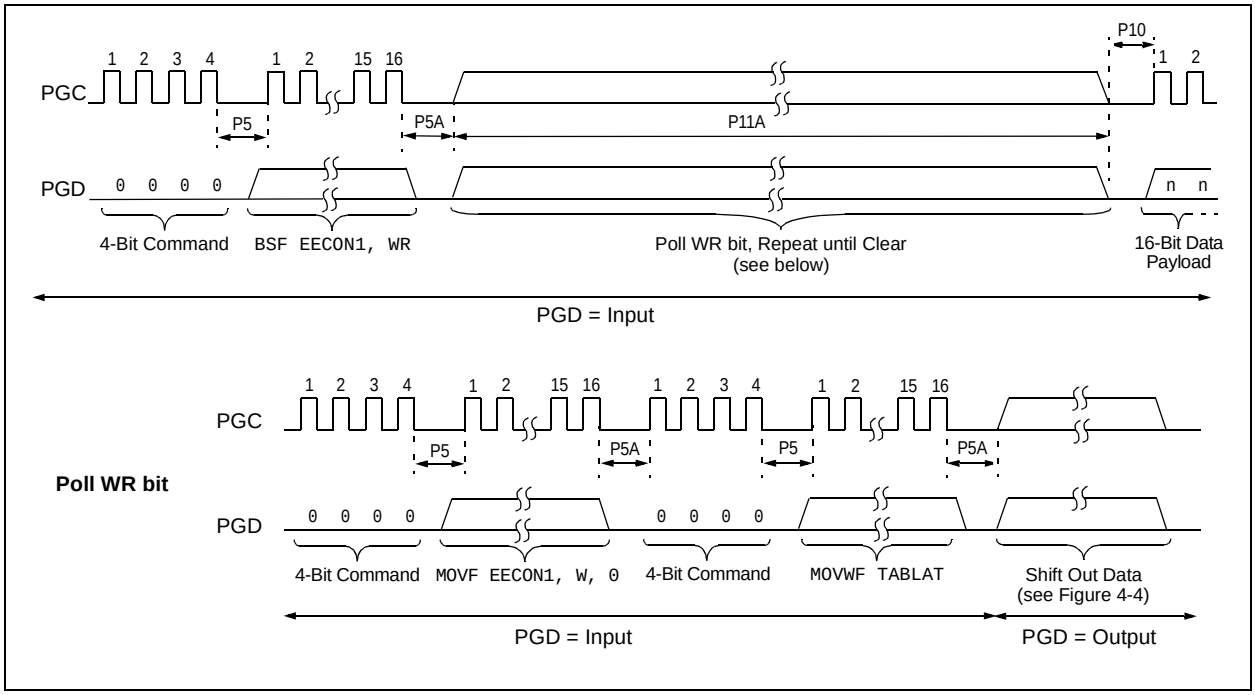


**FIGURE 3-7: DATA EEPROM WRITE TIMING**



# 4.0 READING THE DEVICE

## 4.1 Read Code Memory, ID Locations and Configuration Bits

Code memory is accessed, one byte at a time, via the 4-bit command, '1001' (Table Read, post-increment). The contents of memory pointed to by the Table Pointer (TBLPTRU:TBLPTRH:TBLPTRL) are serially output on PGD.

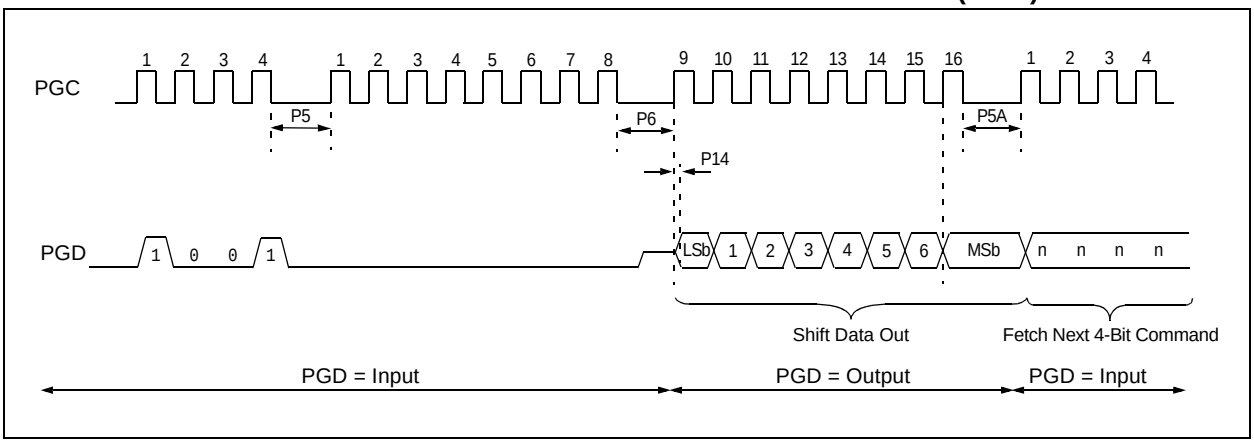
The 4-bit command is shifted in, LSb first. The read is executed during the next eight clocks, then shifted out on PGD during the last eight clocks, LSb to MSb. A delay of P6 must be introduced after the falling edge of the 8th PGC of the operand to allow PGD to transition from an input to an output. During this time, PGC must be held low (see Figure 4-1). This operation also increments the Table Pointer by one, pointing to the next byte in code memory for the next read.

This technique will work to read any memory in the 000000h to 3FFFFFh address space, so it also applies to the reading of the ID and Configuration registers.

**TABLE 4-1: READ CODE MEMORY SEQUENCE**

| 4-Bit Command | Data Payload | Core Instruction |
|---|---|---|
| Step 1: Set Table Pointer. | | |
| 0000 | 0E <Addr[21:16]> | MOVLW Addr[21:16] |
| 0000 | 6E F8 | MOVWF TBLPTRU |
| 0000 | 0E <Addr[15:8]> | MOVLW <Addr[15:8]> |
| 0000 | 6E F7 | MOVWF TBLPTRH |
| 0000 | 0E <Addr[7:0]> | MOVLW <Addr[7:0]> |
| 0000 | 6E F6 | MOVWF TBLPTRL |
| Step 2: Read memory and then shift out on PGD, LSb to MSb. | | |
| 1001 | 00 00 | TBLRD *+ |

**FIGURE 4-1: TABLE READ POST-INCREMENT INSTRUCTION TIMING (1001)**



## 4.4 Read Data EEPROM Memory

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 read by loading EEADRH:EEADR with the desired memory location and initiating a memory read by appropriately configuring the EECON1 register. The data will be loaded into EEDATA, where it may be serially output on PGD via the 4-bit command, '0010' (Shift Out Data Holding register). A delay of P6 must be introduced after the falling edge of the 8th PGC of the operand to allow PGD to transition from an input to an output. During this time, PGC must be held low (see Figure 4-4).

The command sequence to read a single byte of data is shown in Table 4-2.

**FIGURE 4-3: READ DATA EEPROM FLOW**

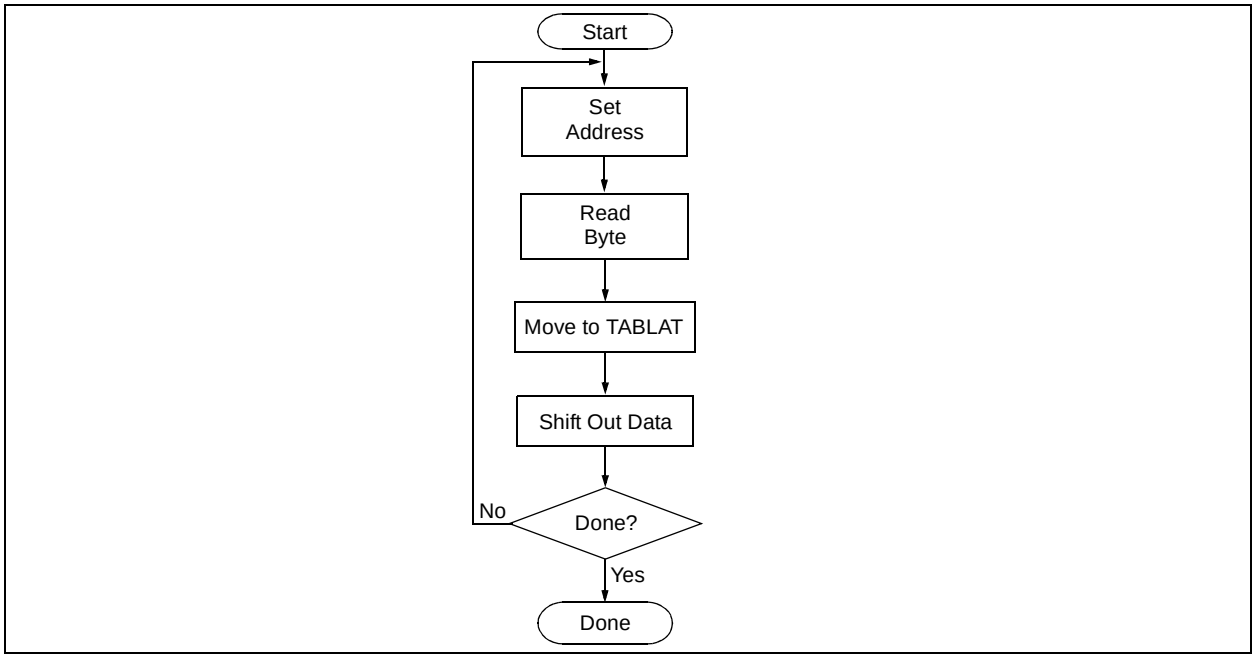



**TABLE 4-2: READ DATA EEPROM MEMORY**

| 4-Bit Command | Data Payload | Core Instruction |
|---|---|---|
| Step 1: Direct access to data EEPROM. | | |
| 0000<br>0000 | 9E A6<br>9C A6 | BCF EECON1, EEPGD<br>BCF EECON1, CFGS |
| Step 2: Set the data EEPROM Address Pointer. | | |
| 0000<br>0000<br>0000<br>0000 | 0E <Addr><br>6E A9<br>OE <AddrH><br>6E AA | MOVLW <Addr><br>MOVWF EEADR<br>MOVLW <AddrH><br>MOVWF EEADRH |
| Step 3: Initiate a memory read. | | |
| 0000 | 80 A6 | BSF EECON1, RD |
| Step 4: Load data into the Serial Data Holding register. | | |
| 0000<br>0000<br>0000<br>0010 | 50 A8<br>6E F5<br>00 00<br><MSB><LSB> | MOVF EEDATA, W, 0<br>MOVWF TABLAT<br>NOP<br>Shift Out Data(1) |

**Note 1:** The <LSB> is undefined. The <MSB> is the data.

## 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**

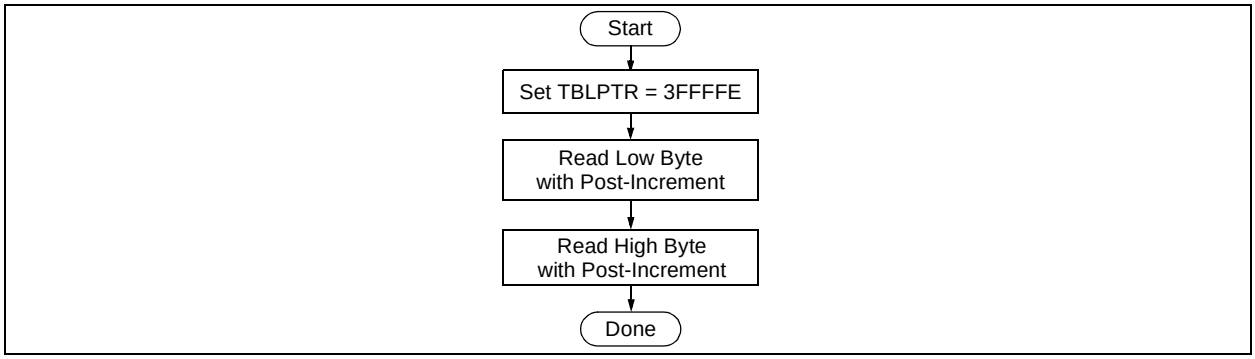


**TABLE 5-2: DEVICE ID VALUES**

| 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 |
|---|---|---|
| WDTEN | CONFIG2H | Watchdog Timer Enable bit<br>1 = WDT is enabled<br>0 = WDT is disabled (control is placed on the SWDTEN bit) |
| MCLRE | CONFIG3H | $\overline{\text{MCLR}}$ Pin Enable bit<br>1 = $\overline{\text{MCLR}}$ pin is enabled, RE3 input pin is disabled<br>0 = RE3 input pin is enabled, $\overline{\text{MCLR}}$ pin is disabled |
| LPT1OSC | CONFIG3H | Low-Power Timer1 Oscillator Enable bit<br>1 = Timer1 is configured for low-power operation<br>0 = Timer1 is configured for high-power operation |
| PBADEN | CONFIG3H | PORTB A/D Enable bit<br>1 = PORTB A/D<4:0> pins are configured as analog input channels on Reset<br>0 = PORTB A/D<4:0> pins are configured as digital I/O on Reset |
| PBADEN | CONFIG3H | PORTB A/D Enable bit **(PIC18FXX8X devices only)**<br>1 = PORTB A/D<4:0> and PORTB A/D<1:0> pins are configured as analog input channels on Reset<br>0 = PORTB A/D<4:0> pins are configured as digital I/O on Reset |
| CCP2MX | CONFIG3H | CCP2 MUX bit<br>1 = CCP2 input/output is multiplexed with RC1**(2)**<br>0 = CCP2 input/output is multiplexed with RB3 |
| $\overline{\text{DEBUG}}$ | CONFIG4L | Background Debugger Enable bit<br>1 = Background debugger is disabled, RB6 and RB7 are configured as general purpose I/O pins<br>0 = Background debugger is enabled, RB6 and RB7 are dedicated to In-Circuit Debug |
| XINST | CONFIG4L | Extended Instruction Set Enable bit<br>1 = Instruction set extension and Indexed Addressing mode are enabled<br>0 = Instruction set extension and Indexed Addressing mode are disabled (Legacy mode) |
| ICPRT | CONFIG4L | Dedicated In-Circuit (ICD/ICSP™) Port Enable bit **(PIC18F2455/2550/4455/4550, PIC18F2458/2553/4458/4553 and PIC18F2450/4450 devices only)**<br>1 = ICPORT is enabled<br>0 = ICPORT is disabled |
| BBSIZ<1:0>**(1)** | CONFIG4L | Boot Block Size Select bits **(PIC18F2585/2680/4585/4680 devices only)**<br>11 = 4K words (8 Kbytes) Boot Block<br>10 = 4K words (8 Kbytes) Boot Block<br>01 = 2K words (4 Kbytes) Boot Block<br>00 = 1K word (2 Kbytes) Boot Block |
| BBSIZ<2:1>**(1)** | CONFIG4L | Boot Block Size Select bits **(PIC18F2682/2685/4582/4685 devices only)**<br>11 = 4K words (8 Kbytes) Boot Block<br>10 = 4K words (8 Kbytes) Boot Block<br>01 = 2K words (4 Kbytes) Boot Block<br>00 = 1K word (2 Kbytes) Boot Block |

**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 |
|---|---|---|
| WRT5 | CONFIG6L | Write Protection bit (Block 5 code memory area)<br>**(PIC18F2685 and PIC18F4685 devices only)**<br>1 = Block 5 is not write-protected<br>0 = Block 5 is write-protected |
| WRT4 | CONFIG6L | Write Protection bit (Block 4 code memory area)<br>**(PIC18F2682/2685 and PIC18F4682/4685 devices only)**<br>1 = Block 4 is not write-protected<br>0 = Block 4 is write-protected |
| WRT3 | CONFIG6L | Write Protection bit (Block 3 code memory area)<br>1 = Block 3 is not write-protected<br>0 = Block 3 is write-protected |
| WRT2 | CONFIG6L | Write Protection bit (Block 2 code memory area)<br>1 = Block 2 is not write-protected<br>0 = Block 2 is write-protected |
| WRT1 | CONFIG6L | Write Protection bit (Block 1 code memory area)<br>1 = Block 1 is not write-protected<br>0 = Block 1 is write-protected |
| WRT0 | CONFIG6L | Write Protection bit (Block 0 code memory area)<br>1 = Block 0 is not write-protected<br>0 = Block 0 is write-protected |
| WRTD | CONFIG6H | Write Protection bit (Data EEPROM)<br>1 = Data EEPROM is not write-protected<br>0 = Data EEPROM is write-protected |
| WRTB | CONFIG6H | Write Protection bit (Boot Block memory area)<br>1 = Boot Block is not write-protected<br>0 = Boot Block is write-protected |
| WRTC | CONFIG6H | Write Protection bit (Configuration registers)<br>1 = Configuration registers are not write-protected<br>0 = Configuration registers are write-protected |
| EBTR5 | CONFIG7L | Table Read Protection bit (Block 5 code memory area)<br>**(PIC18F2685 and PIC18F4685 devices only)**<br>1 = Block 5 is not protected from Table Reads executed in other blocks<br>0 = Block 5 is protected from Table Reads executed in other blocks |
| EBTR4 | CONFIG7L | Table Read Protection bit (Block 4 code memory area)<br>**(PIC18F2682/2685 and PIC18F4682/4685 devices only)**<br>1 = Block 4 is not protected from Table Reads executed in other blocks<br>0 = Block 4 is protected from Table Reads executed in other blocks |
| EBTR3 | CONFIG7L | Table Read Protection bit (Block 3 code memory area)<br>1 = Block 3 is not protected from Table Reads executed in other blocks<br>0 = Block 3 is protected from Table Reads executed in other blocks |
| EBTR2 | CONFIG7L | Table Read Protection bit (Block 2 code memory area)<br>1 = Block 2 is not protected from Table Reads executed in other blocks<br>0 = Block 2 is protected from Table Reads executed in other blocks |
| EBTR1 | CONFIG7L | Table Read Protection bit (Block 1 code memory area)<br>1 = Block 1 is not protected from Table Reads executed in other blocks<br>0 = Block 1 is protected from Table Reads executed in other blocks |

**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 5-4: DEVICE BLOCK LOCATIONS AND SIZES**

| Device | Memory Size (Bytes) | Pins | Ending Address | | | | | | | Size (Bytes) | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Boot Block | Block 0 | Block 1 | Block 2 | Block 3 | Block 4 | Block 5 | Boot Block | Block 0 | Remaining Blocks | Device Total |
| PIC18F2221 | 4K | 28 | 0001FF | 0007FF | 000FFF | — | — | — | — | 512 | 1536 | 2048 | 4096 |
| | | | 0003FF | | | | | | | 1024 | 1024 | | |
| PIC18F2321 | 8K | 28 | 0001FF | 000FFF | 001FFF | — | — | — | — | 512 | 3584 | 4096 | 8192 |
| | | | 0003FF | | | | | | | 1024 | 3072 | | |
| | | | 0007FF | | | | | | | 2048 | 2048 | | |
| PIC18F2410 | 16K | 28 | 0007FF | 001FFF | 003FFF | — | — | — | — | 2048 | 6144 | 8192 | 16384 |
| PIC18F2420 | 16K | 28 | 0007FF | 001FFF | 003FFF | — | — | — | — | 2048 | 6144 | 8192 | 16384 |
| PIC18F2423 | 16K | 28 | 0007FF | 001FFF | 003FFF | — | — | — | — | 2048 | 6144 | 8192 | 16384 |
| PIC18F2450 | 16K | 28 | 0007FF | 001FFF | 003FFF | — | — | — | — | 2048 | 6144 | 8192 | 16384 |
| | | | 000FFF | | | | | | | 4096 | 4096 | | |
| PIC18F2455 | 24K | 28 | 0007FF | 001FFF | 003FFF | 005FFF | — | — | — | 2048 | 6144 | 16384 | 24576 |
| PIC18F2458 | 24K | 28 | 0007FF | 001FFF | 003FFF | 005FFF | — | — | — | 2048 | 6144 | 16384 | 24576 |
| PIC18F2480 | 16K | 28 | 0007FF | 001FFF | 003FFF | — | — | — | — | 2048 | 6144 | 8192 | 16384 |
| | | | 000FFF | | | | | | | 4096 | 4096 | | |
| PIC18F2510 | 32K | 28 | 0007FF | 001FFF | 003FFF | 005FFF | 007FFF | — | — | 2048 | 6144 | 24576 | 32768 |
| PIC18F2515 | 48K | 28 | 0007FF | 003FFF | 007FFF | 00BFFF | — | — | — | 2048 | 14336 | 32768 | 49152 |
| PIC18F2520 | 32K | 28 | 0007FF | 001FFF | 003FFF | 005FFF | 007FFF | — | — | 2048 | 14336 | 16384 | 32768 |
| PIC18F2523 | 32K | 28 | 0007FF | 001FFF | 003FFF | 005FFF | 007FFF | — | — | 2048 | 14336 | 16384 | 32768 |
| PIC18F2525 | 48K | 28 | 0007FF | 003FFF | 007FFF | 00BFFF | — | — | — | 2048 | 14336 | 32768 | 49152 |
| PIC18F2550 | 32K | 28 | 0007FF | 001FFF | 003FFF | 005FFF | 007FFF | — | — | 2048 | 6144 | 24576 | 32768 |
| PIC18F2553 | 32K | 28 | 0007FF | 001FFF | 003FFF | 005FFF | 007FFF | — | — | 2048 | 6144 | 24576 | 32768 |
| PIC18F2580 | 32K | 28 | 0007FF | 001FFF | 003FFF | 005FFF | 007FFF | — | — | 2048 | 6144 | 24576 | 32768 |
| | | | 000FFF | | | | | | | 4096 | 4096 | | |
| PIC18F2585 | 48K | 28 | 0007FF | 003FFF | 007FFF | 00BFFF | — | — | — | 2048 | 14336 | 32768 | 49152 |
| | | | 000FFF | | | | | | | 4096 | 12288 | | |
| | | | 001FFF | | | | | | | 8192 | 8192 | | |
| PIC18F2610 | 64K | 28 | 0007FF | 003FFF | 007FFF | 00BFFF | 00FFFF | — | — | 2048 | 14336 | 49152 | 65536 |
| PIC18F2620 | 64K | 28 | 0007FF | 003FFF | 007FFF | 00BFFF | 00FFFF | — | — | 2048 | 14336 | 49152 | 65536 |
| PIC18F2680 | 64K | 28 | 0007FF | 003FFF | 007FFF | 00BFFF | 00FFFF | — | — | 2048 | 14336 | 49152 | 65536 |
| | | | 000FFF | | | | | | | 4096 | 12288 | | |
| | | | 001FFF | | | | | | | 8192 | 8192 | | |
| PIC18F2682 | 80K | 28 | 0007FF | 003FFF | 007FFF | 00BFFF | 00FFFF | 013FFF | — | 2048 | 14336 | 65536 | 81920 |
| | | | 000FFF | | | | | | | 4096 | 12288 | | |
| | | | 001FFF | | | | | | | 8192 | 8192 | | |
| PIC18F2685 | 96K | 28 | 0007FF | 003FFF | 007FFF | 00BFFF | 00FFFF | 013FFF | 017FFF | 2048 | 14336 | 81920 | 98304 |
| | | | 000FFF | | | | | | | 4096 | 12288 | | |
| | | | 001FFF | | | | | | | 8192 | 8192 | | |
| PIC18F4221 | 4K | 40 | 0001FF | 0007FF | 000FFF | — | — | — | — | 512 | 1536 | 2048 | 4096 |
| | | | 0003FF | | | | | | | 1024 | 1024 | | |
| PIC18F4321 | 8K | 40 | 0001FF | 000FFF | 001FFF | — | — | — | — | 512 | 3584 | 4096 | 8192 |
| | | | 0003FF | | | | | | | 1024 | 3072 | | |
| | | | 0007FF | | | | | | | 2048 | 2048 | | |
| PIC18F4410 | 16K | 40 | 0007FF | 001FFF | 003FFF | — | — | — | — | 2048 | 6144 | 8192 | 16384 |
| PIC18F4420 | 16K | 40 | 0007FF | 001FFF | 003FFF | — | — | — | — | 2048 | 6144 | 8192 | 16384 |
| PIC18F4423 | 16K | 40 | 0007FF | 001FFF | 003FFF | — | — | — | — | 2048 | 6144 | 8192 | 16384 |
| PIC18F4450 | 16K | 40 | 0007FF | 001FFF | 003FFF | — | — | — | — | 2048 | 6144 | 8192 | 16384 |
| | | | 000FFF | | | | | | | 4096 | 4096 | | |

**Legend:** — = unimplemented.

**TABLE 5-4: DEVICE BLOCK LOCATIONS AND SIZES (CONTINUED)**

| Device | Memory Size (Bytes) | Pins | Ending Address | | | | | | | Size (Bytes) | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Boot Block | Block 0 | Block 1 | Block 2 | Block 3 | Block 4 | Block 5 | Boot Block | Block 0 | Remaining Blocks | Device Total |
| PIC18F4455 | 24K | 40 | 0007FF | 001FFF | 003FFF | 005FFF | — | — | — | 2048 | 6144 | 16384 | 24576 |
| PIC18F4458 | 24K | 40 | 0007FF | 001FFF | 003FFF | 005FFF | — | — | — | 2048 | 6144 | 16384 | 24576 |
| PIC18F4480 | 16K | 40 | 0007FF | 001FFF | 003FFF | — | — | — | — | 2048 | 6144 | 8192 | 16384 |
| | | | 000FFF | | | | | | | 4096 | 4096 | | |
| PIC18F4510 | 32K | 40 | 0007FF | 001FFF | 003FFF | 005FFF | 007FFF | — | — | 2048 | 6144 | 24576 | 32768 |
| PIC18F4515 | 48K | 40 | 0007FF | 003FFF | 007FFF | 00BFFF | — | — | — | 2048 | 14336 | 32768 | 49152 |
| PIC18F4520 | 32K | 40 | 0007FF | 001FFF | 003FFF | 005FFF | 007FFF | — | — | 2048 | 14336 | 16384 | 32768 |
| PIC18F4523 | 32K | 40 | 0007FF | 001FFF | 003FFF | 005FFF | 007FFF | — | — | 2048 | 14336 | 16384 | 32768 |
| PIC18F4525 | 48K | 40 | 0007FF | 003FFF | 007FFF | 00BFFF | — | — | — | 2048 | 14336 | 32768 | 49152 |
| PIC18F4550 | 32K | 40 | 0007FF | 001FFF | 003FFF | 005FFF | 007FFF | — | — | 2048 | 6144 | 24576 | 32768 |
| PIC18F4553 | 32K | 40 | 0007FF | 001FFF | 003FFF | 005FFF | 007FFF | — | — | 2048 | 6144 | 24576 | 32768 |
| PIC18F4580 | 32K | 40 | 0007FF | 001FFF | 003FFF | 005FFF | 007FFF | — | — | 2048 | 6144 | 24576 | 32768 |
| | | | 000FFF | | | | | | | 4096 | 4096 | | |
| PIC18F4585 | 48K | 40 | 0007FF | 003FFF | 007FFF | 00BFFF | — | — | — | 2048 | 14336 | 32768 | 49152 |
| | | | 000FFF | | | | | | | 4096 | 12288 | | |
| | | | 001FFF | | | | | | | 8192 | 8192 | | |
| PIC18F4610 | 64K | 40 | 0007FF | 003FFF | 007FFF | 00BFFF | 00FFFF | — | — | 2048 | 14336 | 49152 | 65536 |
| PIC18F4620 | 64K | 40 | 0007FF | 003FFF | 007FFF | 00BFFF | 00FFFF | — | — | 2048 | 14336 | 49152 | 65536 |
| PIC18F4680 | 64K | 40 | 0007FF | 003FFF | 007FFF | 00BFFF | 00FFFF | — | — | 2048 | 14336 | 49152 | 65536 |
| | | | 000FFF | | | | | | | 4096 | 12288 | | |
| | | | 001FFF | | | | | | | 8192 | 8192 | | |
| PIC18F4682 | 80K | 40 | 0007FF | 003FFF | 007FFF | 00BFFF | 00FFFF | 013FFF | — | 2048 | 14336 | 65536 | 81920 |
| | | | 000FFF | | | | | | | 4096 | 12288 | | |
| | | | 001FFF | | | | | | | 8192 | 8192 | | |
| PIC18F4685 | 96K | 44 | 0007FF | 003FFF | 007FFF | 00BFFF | 00FFFF | 013FFF | 017FFF | 2048 | 14336 | 81920 | 98304 |
| | | | 000FFF | | | | | | | 4096 | 12288 | | |
| | | | 001FFF | | | | | | | 8192 | 8192 | | |

**Legend:** — = unimplemented.

**Note the following details of the code protection feature on Microchip devices:**

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.



**Trademarks**



ISBN: 978-1-63277-856-7

**QUALITY MANAGEMENT SYSTEM**
**CERTIFIED BY DNV**
**═ ISO/TS 16949 ═**

*Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.*