PIC18FX5X5/X6X0

Flash Microcontroller Programming Specification

1.0 DEVICE OVERVIEW

This document includes the programming specifications for the following devices:

- PIC18F2515
- PIC18F2525
- PIC18F2585
- PIC18F2610
- PIC18F2620
- PIC18F2680
- PIC18F4515
- PIC18F4525
- PIC18F4585
- PIC18F4610
- PIC18F4620
- PIC18F4680

2.0 PROGRAMMING OVERVIEW OF THE PIC18FX5X5/X6X0

PIC18FX5X5/X6X0 devices can be programmed using either the high-voltage In-Circuit Serial Programming™ (ICSP™) method, or the low-voltage ICSP method. Both methods can be done with the device in the users' system. The low-voltage ICSP method is slightly different than the high-voltage method and these differences are noted where applicable. This programming specification applies to PIC18FX5X5/X6X0 devices in all package types.

2.1 Hardware Requirements

In High-Voltage ICSP mode, PIC18FX5X5/X6X0 devices require two programmable power supplies: one for VDD and one for $\overline{\text{MCLR}}$/VPP. Both supplies should have a minimum resolution of 0.25V. Refer to **Section 6.0 "AC/DC Characteristics Timing Requirements for Program/Verify Test Mode"** for additional hardware parameters.

2.1.1 LOW-VOLTAGE ICSP PROGRAMMING

In Low-Voltage ICSP mode, PIC18FX5X5/X6X0 devices can be programmed using a VDD source in the operating range. The $\overline{\text{MCLR}}$/VPP does not have to be brought to a different voltage, but can instead be left at the normal operating voltage. Refer to **Section 6.0 "AC/DC Characteristics Timing Requirements for Program/Verify Test Mode"** for additional hardware parameters.

2.2 Pin Diagrams

The pin diagrams for the PIC18FX5X5/X6X0 family are shown in Figure 2-1 and Figure 2-2.

TABLE 2-1: PIN DESCRIPTIONS (DURING PROGRAMMING): PIC18FX5X5/X6X0

Pin Name	During Programming		
	Pin Name	**Pin Type**	**Pin Description**
$\overline{\text{MCLR}}$/VPP/RE3	VPP	P	Programming Enable
VDD[2]	VDD	P	Power Supply
VSS[2]	VSS	P	Ground
RB5	PGM	I	Low-Voltage ICSP Input when LVP Configuration bit equals '1'[1]
RB6	PGC	I	Serial Clock
RB7	PGD	I/O	Serial Data

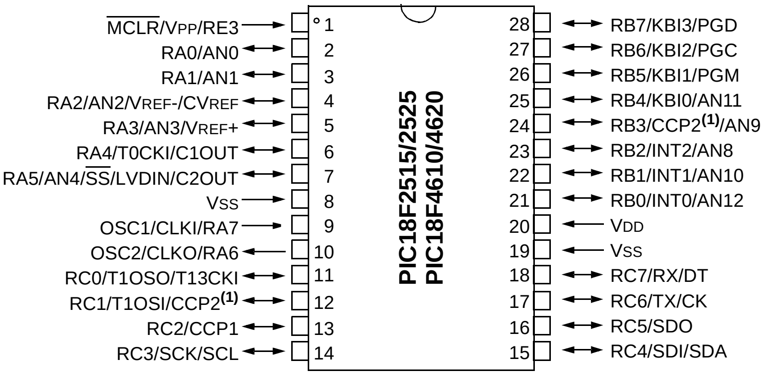
Legend: I = Input, O = Output, P = Power

Note 1: See **Section 5.3 "Single-Supply ICSP Programming"** for more detail.

2: All power supply (VDD) and ground (VSS) must be connected.

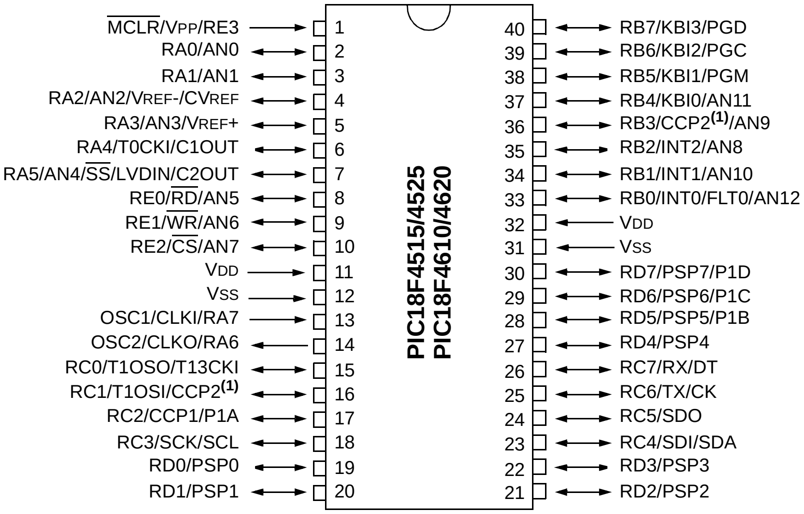
FIGURE 2-1: PIC18FX5X5/X6X0 FAMILY PIN DIAGRAMS

28-Pin SDIP, SOIC (300 MIL)



Note 1: Pin feature is dependent on device configuration.

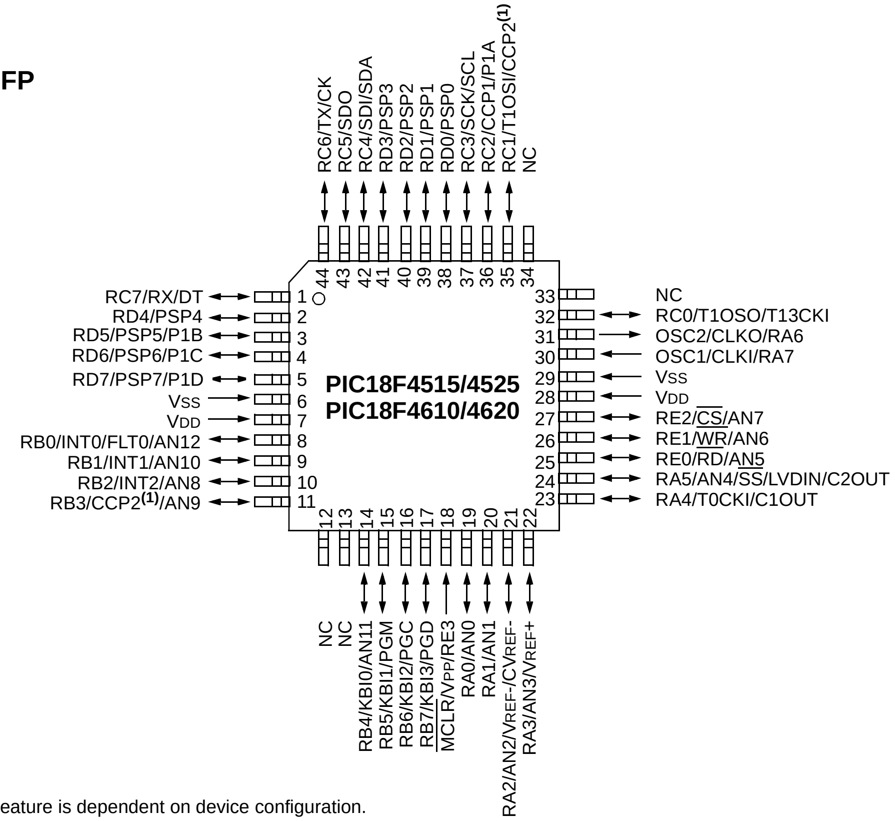
40-Pin PDIP (600 MIL)



Note 1: Pin feature is dependent on device configuration.

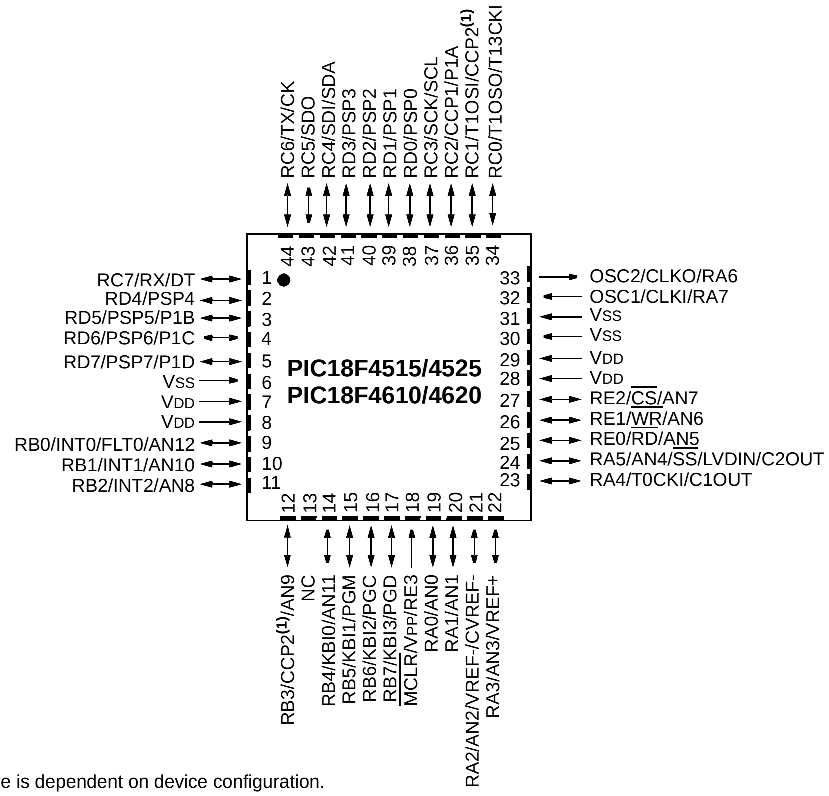
FIGURE 2-2: PIC18FX5X5/X6X0 FAMILY PIN DIAGRAMS

44-Pin TQFP



Note 1: Pin feature is dependent on device configuration.

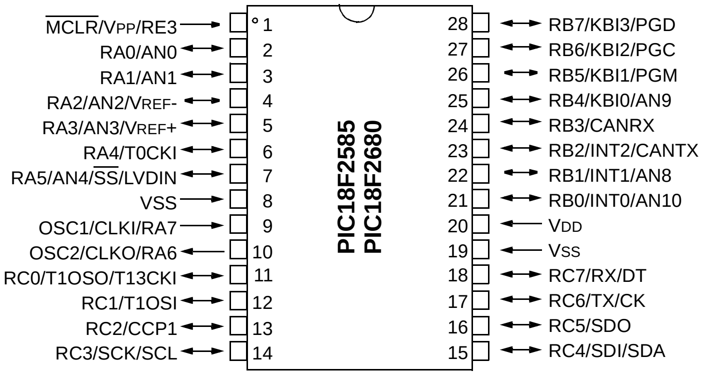
44-Pin QFN



Note 1: Pin feature is dependent on device configuration.

FIGURE 2-3: PIC18FX585/X680 FAMILY PIN DIAGRAMS

28-Pin SDIP, SOIC (300 MIL)



40-Pin PDIP (600 MIL)

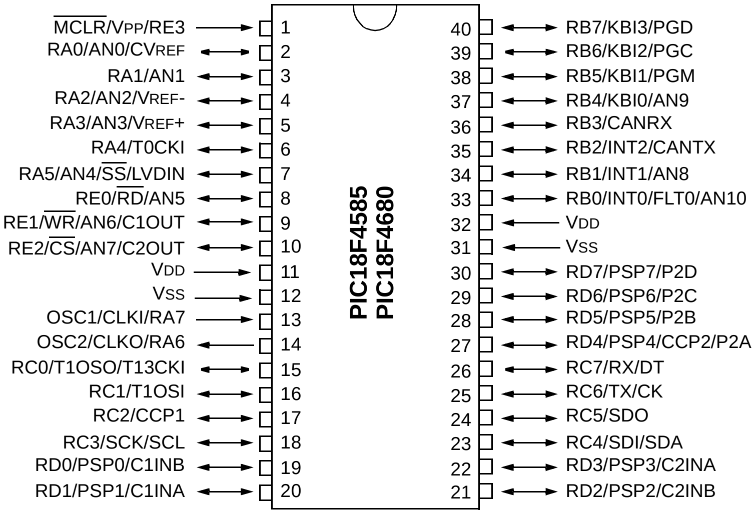
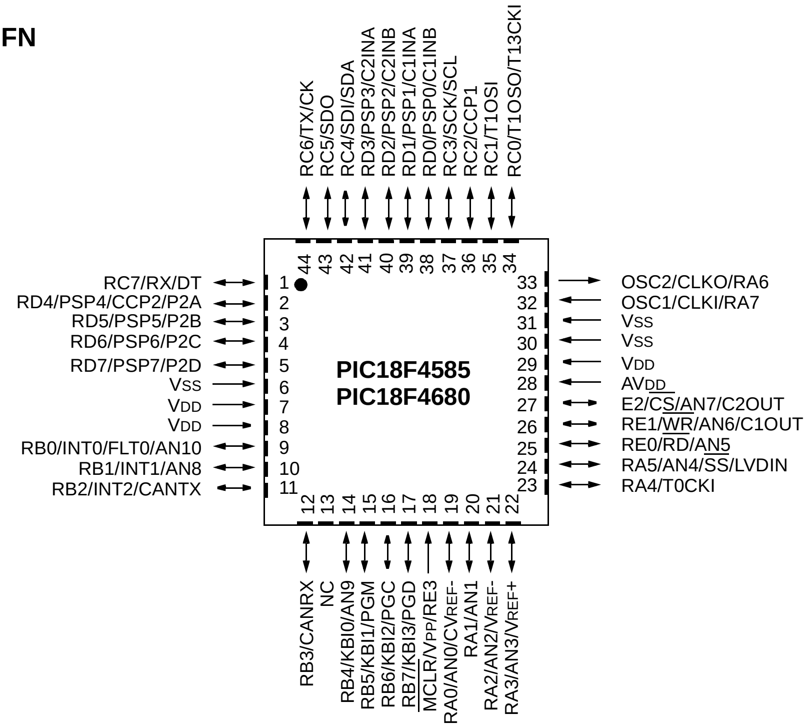
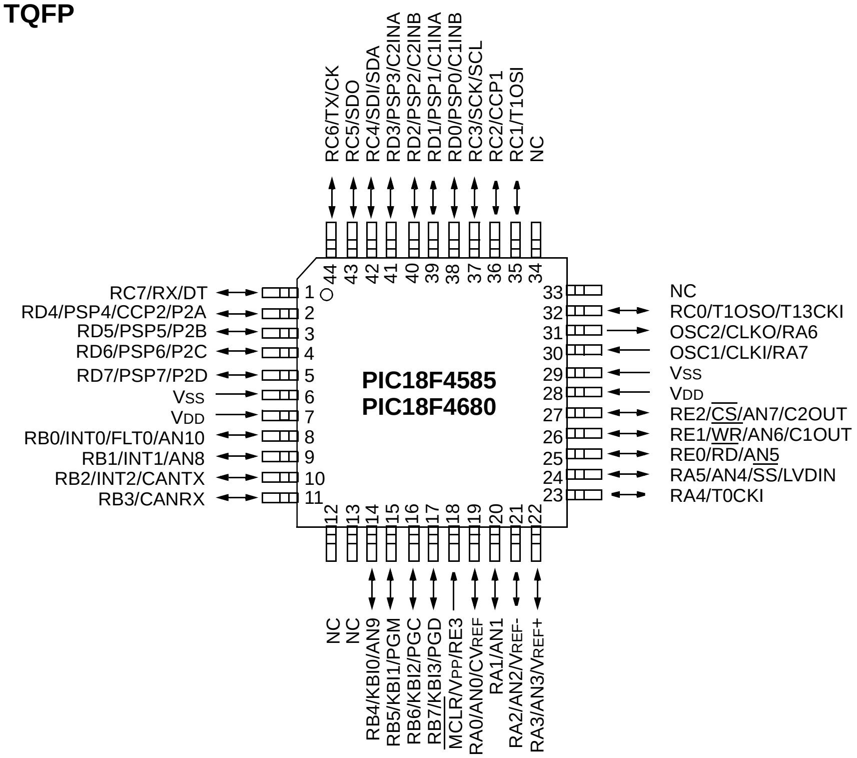


FIGURE 2-4: PIC18FX585/X680 FAMILY PIN DIAGRAMS

44-Pin QFN



44-Pin TQFP



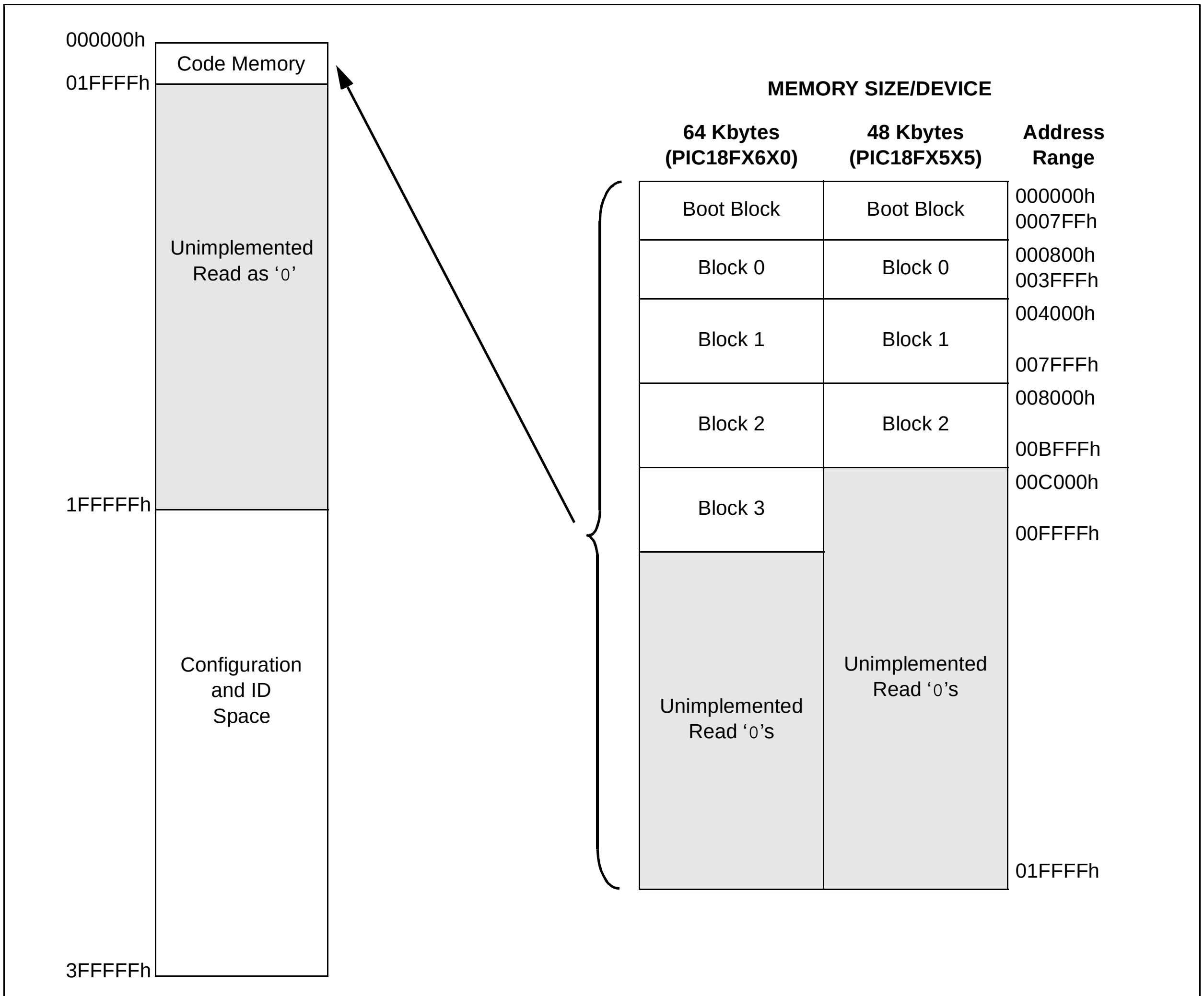
2.3 Memory Map

The code memory space extends from 0000h to 0FFFFh (64 Kbytes) in four 16-Kbyte blocks. Addresses 0000h through 07FFh, 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 2-2: IMPLEMENTATION OF CODE MEMORY

Device	Code Memory Size (Bytes)
PIC18F2515	000000h-00BFFFh (48K)
PIC18F2525	
PIC18F2585	
PIC18F4515	
PIC18F4525	
PIC18F4585	
PIC18F2610	000000h-00FFFFh (64K)
PIC18F2620	
PIC18F2680	
PIC18F4610	
PIC18F4620	
PIC18F4680	

FIGURE 2-5: MEMORY MAP AND THE CODE MEMORY SPACE FOR PIC18FX5X5/X6X0 DEVICES



Note: Sizes of memory areas are not to scale.

In addition to the code memory space, there are three blocks in the configuration and ID space that are accessible to the user through table reads and table writes. Their locations in the memory map are shown in Figure 2-6.

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

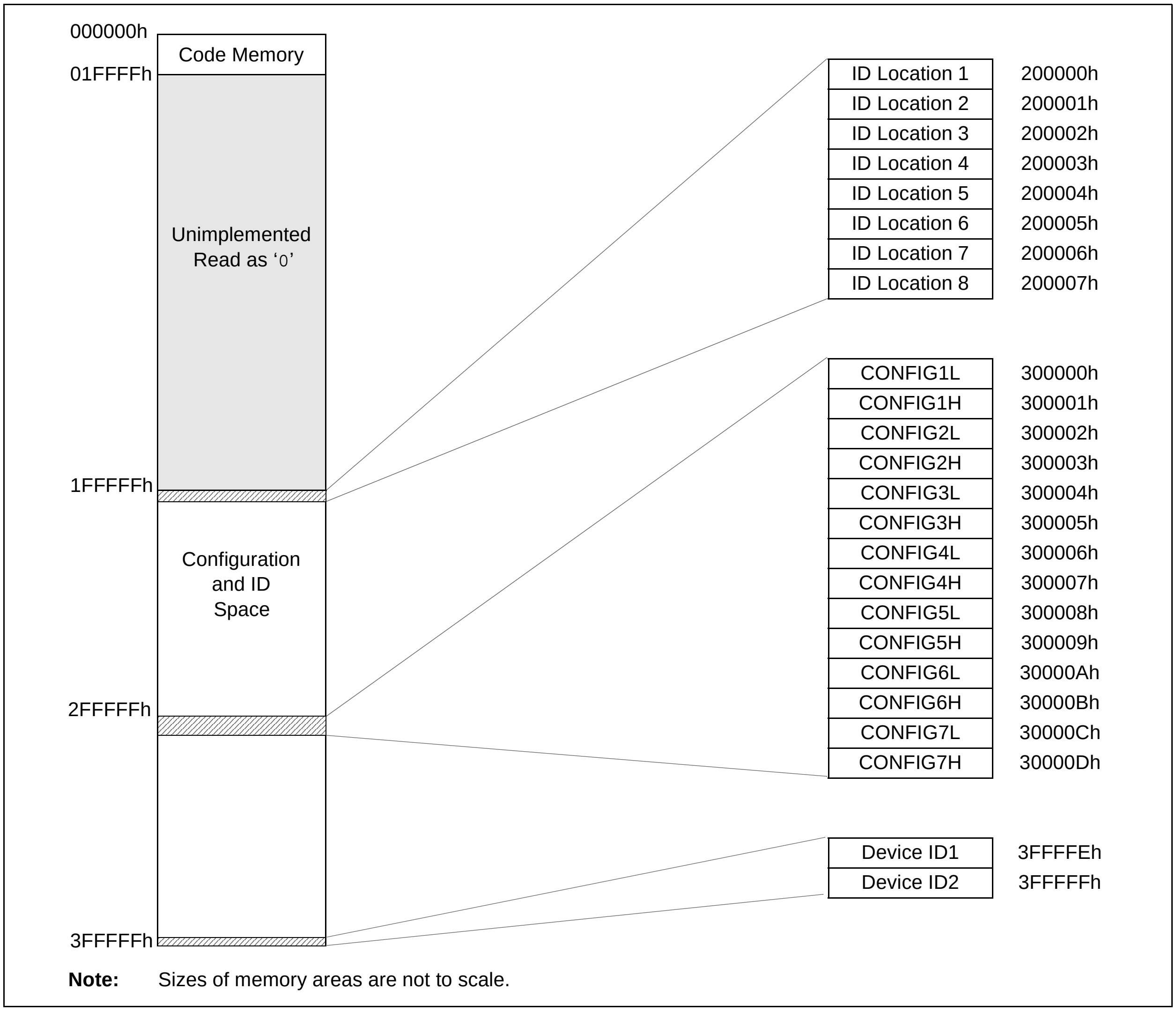
Memory in the address space, 0000000h to 3FFFFFh, is addressed via the Table Pointer register, which is comprised of three Pointer registers:

- 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 2-6: CONFIGURATION AND ID LOCATIONS FOR PIC18FX5X5/X6X0 DEVICES



Note: Sizes of memory areas are not to scale.

2.4 High-Level Overview of the Programming Process

Figure 2-8 shows the high-level overview of the programming process. First, a bulk erase is performed. Next, the code memory, ID locations and data EEPROM (PIC18FXX2X and PIC18FXX8X only) are programmed. These memories are then verified to ensure that programming was successful. If no errors are detected, the configuration bits are then programmed and verified.

2.5 Entering High-Voltage ICSP Program/Verify Mode

The High-Voltage ICSP Program/Verify mode is entered by holding PGC and PGD low and then raising $\overline{\text{MCLR}}$/VPP to VIHH (high voltage). Once in this mode, the code memory, data EEPROM (PIC18FXX2X and PIC18FXX8X only), ID locations and configuration bits can be accessed and programmed in serial fashion.

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

2.5.1 ENTERING LOW-VOLTAGE ICSP PROGRAM/VERIFY MODE

When the LVP configuration bit is '1' (see **Section 5.3 "Single-Supply ICSP Programming"**), the Low-Voltage ICSP mode is enabled. Low-Voltage ICSP Program/Verify mode is entered by holding PGC and PGD low, placing a logic high on PGM and then raising $\overline{\text{MCLR}}$/VPP to VIH. In this mode, the RB5/PGM pin is dedicated to the programming function and ceases to be a general purpose I/O pin.

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

FIGURE 2-7: ENTERING HIGH-VOLTAGE PROGRAM/VERIFY MODE

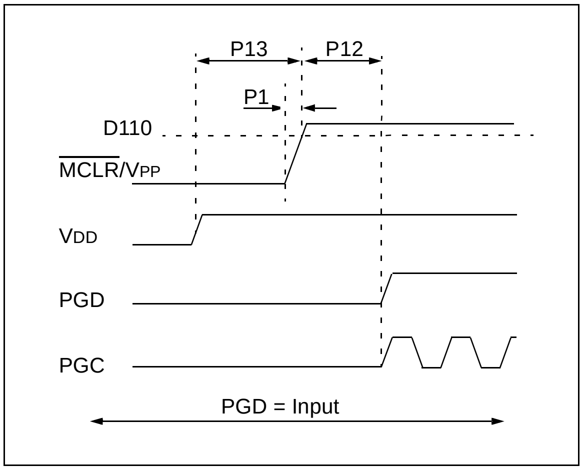


FIGURE 2-8: HIGH-LEVEL PROGRAMMING FLOW

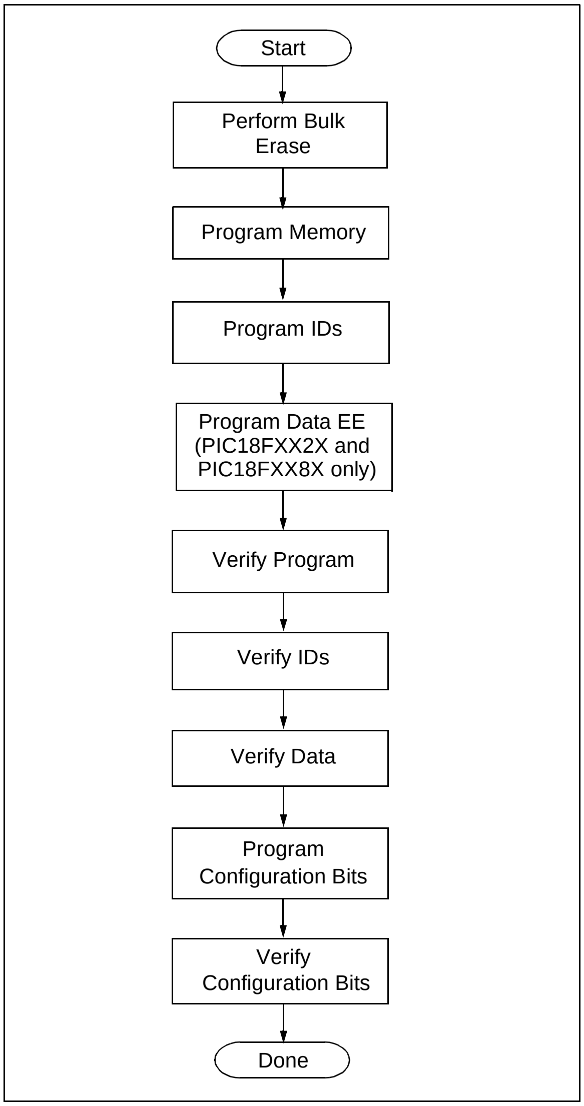
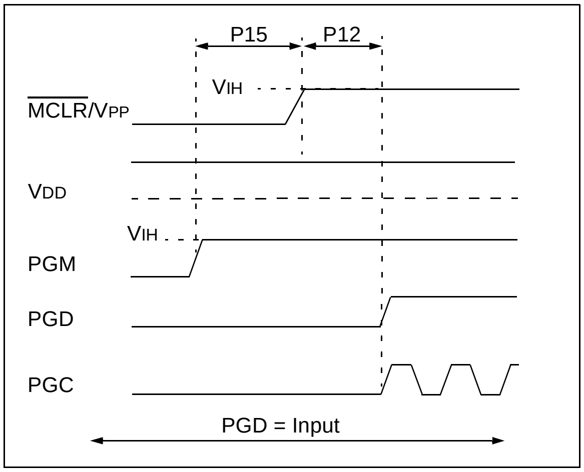

Start
Perform Bulk Erase
Program Memory
Program IDs
Program Data EE (PIC18FXX2X and PIC18FXX8X only)
Verify Program
Verify IDs
Verify Data
Program Configuration Bits
Verify Configuration Bits
Done

FIGURE 2-9: ENTERING LOW-VOLTAGE PROGRAM/VERIFY MODE



2.6 Serial Program/Verify Operation

The PGC pin is used as a clock input pin and the PGD pin is used for entering command bits and data input/output during serial operation. Commands and data are transmitted on the rising edge of PGC, latched on the falling edge of PGC and are Least Significant bit (LSb) first.

2.6.1 4-BIT COMMANDS

All instructions are 20 bits, consisting of a leading 4-bit command followed by a 16-bit operand, which depends on the type of command being executed. To input a command, PGC is cycled four times. The commands needed for programming and verification are shown in Table 2-3.

Depending on the 4-bit command, the 16-bit operand represents 16 bits of input data or 8 bits of input data and 8 bits of output data.

Throughout this specification, commands and data are presented as illustrated in Table 2-4. The 4-bit command is shown MSb first. The command operand, or “Data Payload”, is shown <MSB><LSB>. Figure 2-10 demonstrates how to serially present a 20-bit command/operand to the device.

2.6.2 CORE INSTRUCTION

The core instruction passes a 16-bit instruction to the CPU core for execution. This is needed to setup registers as appropriate for use with other commands.

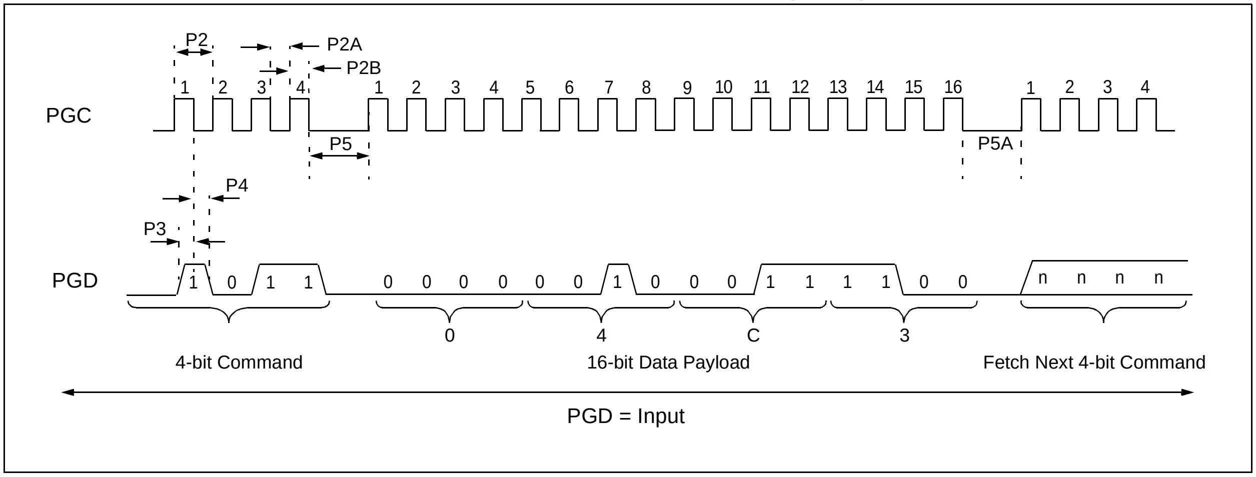
TABLE 2-3: COMMANDS FOR PROGRAMMING

Description	4-Bit Command
Core Instruction (Shift in16-bit instruction)	`0000`
Shift out TABLAT register	`0010`
Table Read	`1000`
Table Read, post-increment	`1001`
Table Read, post-decrement	`1010`
Table Read, pre-increment	`1011`
Table Write	`1100`
Table Write, post-increment by 2	`1101`
Table Write, start programming, post-increment by 2	`1110`
Table Write, start programming	`1111`

TABLE 2-4: SAMPLE COMMAND SEQUENCE

4-Bit Command	Data Payload	Core Instruction
`1101`	`3C 40`	`Table Write, post-increment by 2`

FIGURE 2-10: TABLE WRITE, POST-INCREMENT TIMING (`1101`)



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 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., 0X84h as shown in Table 3-1, where X defines the block to be erased).

TABLE 3-1: BULK ERASE OPTIONS

Description	Data
Chip Erase	0F87h
Erase Data EEPROM[1]	0084h
Erase Boot Block	0081h
Erase Config Bits	0082h
Erase Block 0	0180h
Erase Block 1	0280h
Erase Block 2	0480h
Erase Block 3	0880h

Note 1: PIC18FXX2X and PIC18FXX8X only.

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 3-2 and the flow chart 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	0F 0F	Write 0Fh 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	87 87	Write 8787h TO 3C0004h to erase entire device.
0000	00 00	NOP
0000	00 00	Hold PGD low until erase completes.

FIGURE 3-1: BULK ERASE FLOW

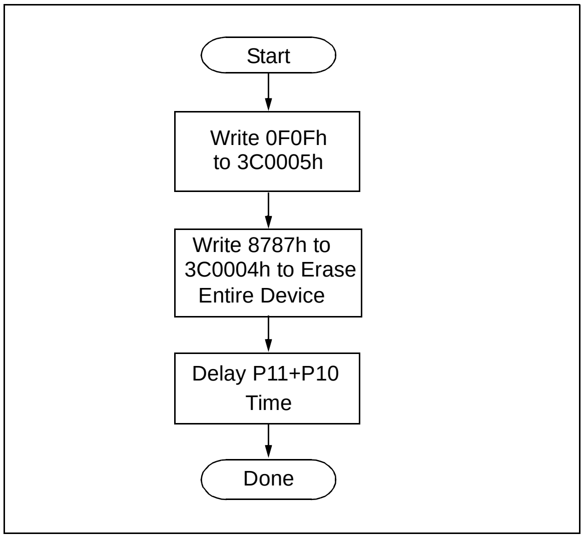
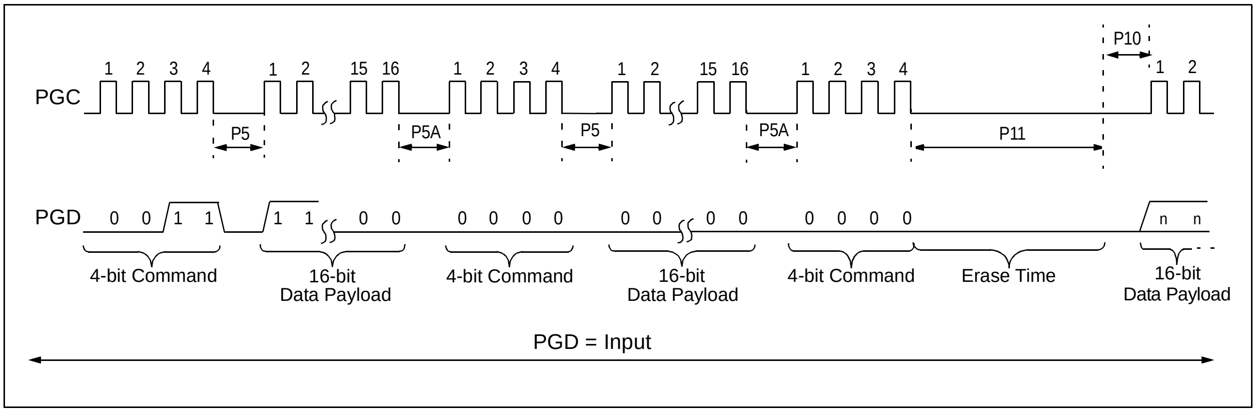


FIGURE 3-2: BULK ERASE TIMING



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 Sections 3.1.3 and 3.2.1.

If it is determined that a data EEPROM (PIC18FXX2X and PIC18FXX8X only) erase must be performed at a supply voltage below the Bulk Erase limit, follow the methodology described in **Section 3.3 "Data EEPROM Programming (PIC18FXX2X and PIC18FXX8X only)"** and write '1's to the array.

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 Map"**).

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.

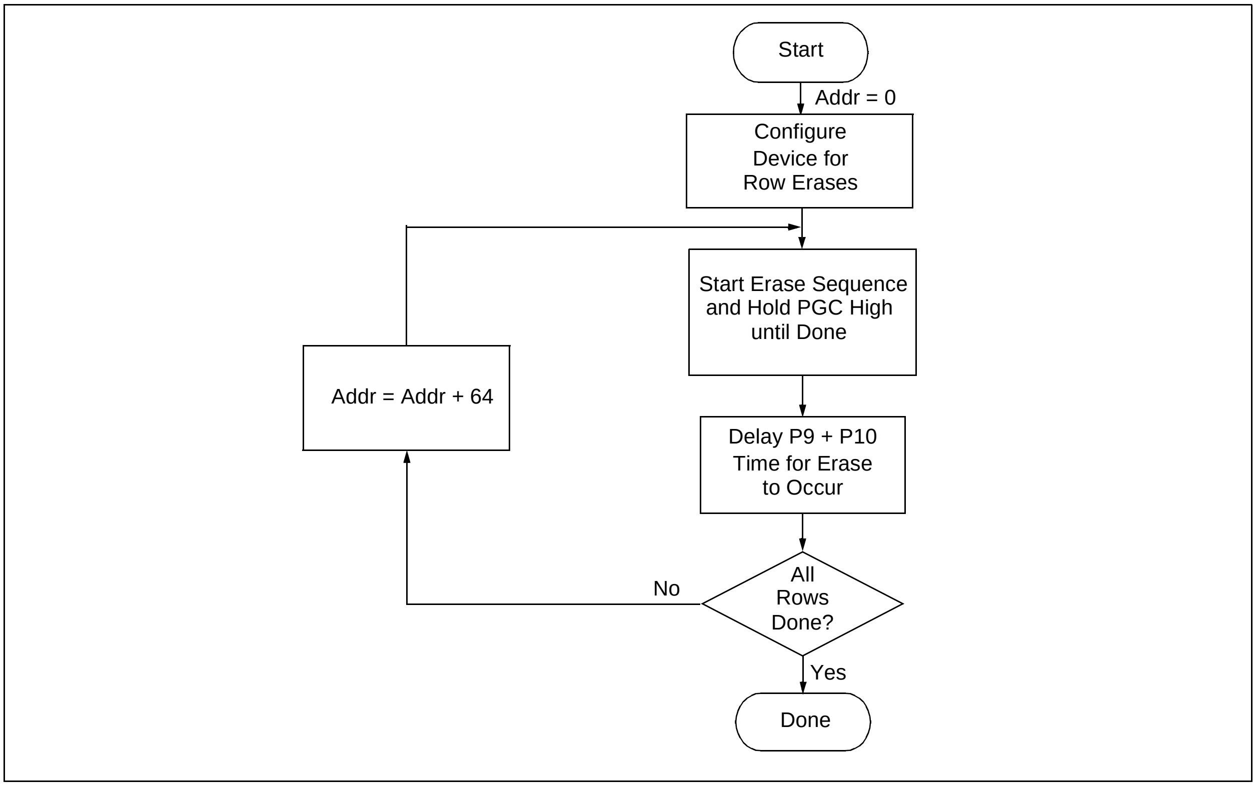
The code sequence to Row Erase a PIC18FX5X5/X6X0 device is shown in Table 3-3. The flow chart shown in Figure 3-3 depicts the logic necessary to completely erase a PIC18FX5X5/X6X0 device. The timing diagram that details the "Start Programming" command and parameters P9 and P10 is shown in Figure 3-5.

Note:	The TBLPTR register can point at any byte within the row(s) 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 0000 0000	8E A6 9C A6 84 A6	BSF EECON1, EEPGD BCF EECON1, CFGS BSF EECON1, WREN
Step 2: Point to first row in code memory.		
0000 0000 0000	6A F8 6A F7 6A F6	CLRF TBLPTRU CLRF TBLPTRH CLRF TBLPTRL
Step 3: Enable erase and erase single row.		
0000 0000 0000	88 A6 82 A6 00 00	BSF EECON1, FREE BSF EECON1, WR NOP - hold PGC high for time P9.
Step 4: Repeat step 3, with address pointer incremented by 64 until all rows are erased.		

FIGURE 3-3: SINGLE ROW ERASE CODE MEMORY FLOW


Start
Addr = 0
Configure Device for Row Erases
Start Erase Sequence and Hold PGC High until Done
Addr = Addr + 64
Delay P9 + P10 Time for Erase to Occur
No
All Rows Done?
Yes
Done

3.2 Code Memory Programming

Programming code memory is accomplished by first loading data into the write buffer and then initiating a programming sequence. The write buffer is 64 bytes in size and can be mapped to any 64-byte area in code memory beginning at location 000000h. The actual memory write sequence takes the contents of this buffer and programs the 64-byte code memory region that contains the Table Pointer.

The programming duration is externally timed and is controlled by PGC. After a “Start Programming” command is issued (4-bit command, ‘1111’), 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 program a PIC18FX5X5/X6X0 device is shown in Table 3-4. The flow chart shown in Figure 3-4 depicts the logic necessary to completely write a PIC18FX5X5/X6X0 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 must point to the same 64-byte region when initiating the programming sequence as it did when the write buffers were loaded.

TABLE 3-4: WRITE CODE MEMORY CODE SEQUENCE

4-bit Command	Data Payload	Core Instruction
Step 1: Direct access to code memory and enable writes.		
0000 0000	8E A6 9C A6	BSF EECON1, EEPGD BCF EECON1, CFGS
Step 2: Load write buffer.		
0000 0000 0000 0000 0000 0000	0E <Addr[21:16]> 6E F8 0E <Addr[15:8]> 6E F7 0E <Addr[7:0]> 6E F6	MOVLW <Addr[21:16]> MOVWF TBLPTRU MOVLW <Addr[15:8]> MOVWF TBLPTRH MOVLW <Addr[7:0]> 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 0000	<MSB><LSB> 00 00	Write 2 bytes and start programming NOP - hold PGC high for time P9
To continue writing data, repeat steps 2 through 4, where the address pointer is incremented by 2 at each iteration of the loop.		

FIGURE 3-4: PROGRAM CODE MEMORY FLOW

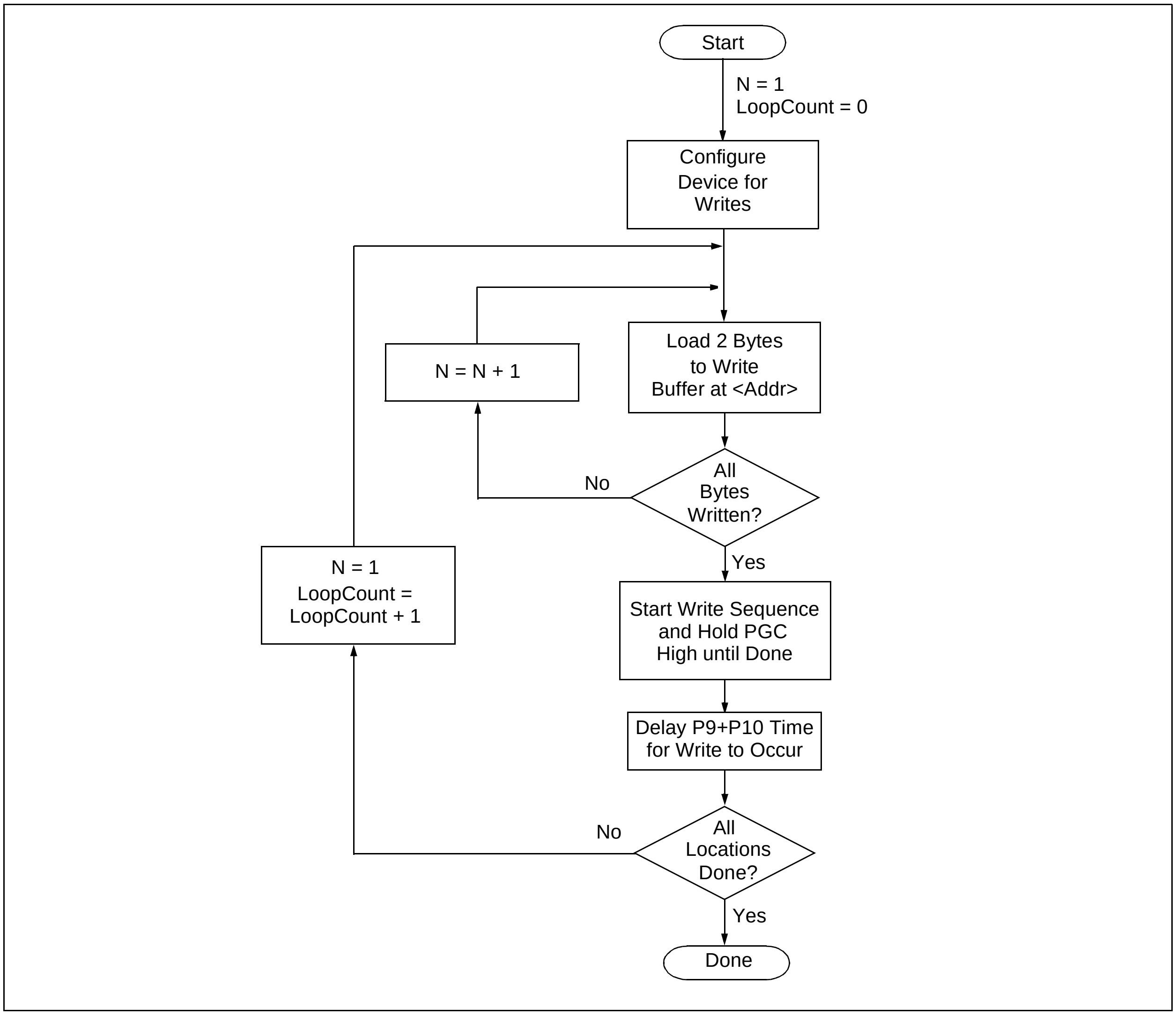
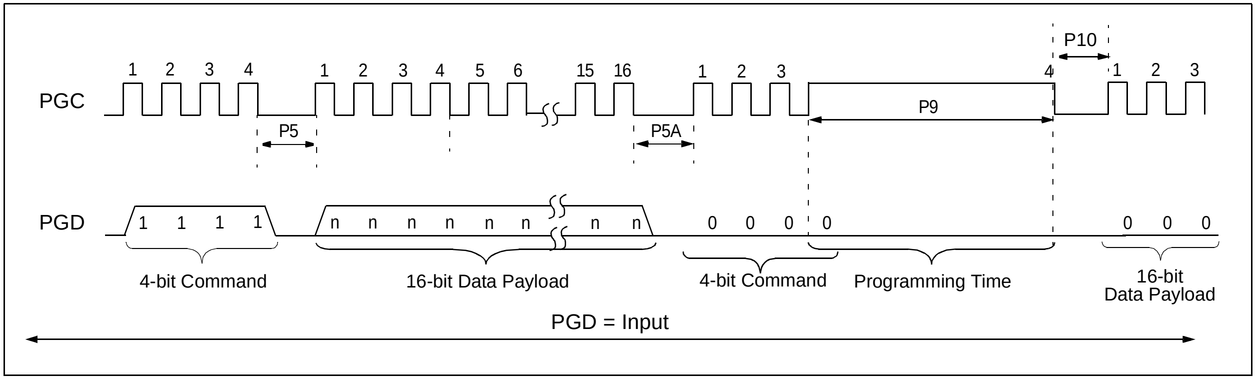

Start
N = 1
LoopCount = 0
Configure Device for Writes
Load 2 Bytes to Write Buffer at <Addr>
N = N + 1
All Bytes Written?
No
Yes
Start Write Sequence and Hold PGC High until Done
Delay P9+P10 Time for Write to Occur
All Locations Done?
No
N = 1
LoopCount = LoopCount + 1
Yes
Done

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



3.2.1 MODIFYING CODE MEMORY

The previous programming example assumed that the device has 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.

In this case, 64 bytes 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 64-byte 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 3-5: 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`
`0000`
	`8E A6`
	`9C A6`
		`BSF     EECON1, EEPGD`
		`BCF     EECON1, CFGS`

Step 3: Set the Table Pointer for the block to be erased.		

`0000`
`0000`
`0000`
`0000`
`0000`
`0000`
	`0E <Addr[21:16]>`
	`6E F8`
	`0E <Addr[8:15]>`
	`6E F7`
	`0E <Addr[7:0]>`
	`6E F6`
		`MOVLW   <Addr[21:16]>`
		`MOVWF   TBLPTRU`
		`MOVLW   <Addr[8:15]>`
		`MOVWF   TBLPTRH`
		`MOVLW   <Addr[7:0]>`
		`MOVWF   TBLPTRL`

Step 4: Enable memory writes and setup an erase.		

`0000`
`0000`
	`84 A6`
	`88 A6`
		`BSF     EECON1, WREN`
		`BSF     EECON1, FREE`

Step 5: Initiate erase.		

`0000`
`0000`
	`82 A6`
	`00 00`
		`BSF     EECON1, WR`
		`NOP - hold PGC high for time P9`

Step 6: Wait for P10.		
Step 7: Load write buffer. The correct bytes will be selected based on the Table Pointer.		

`0000`
`0000`
`0000`
`0000`
`0000`
`0000`
`1101`
.
.
.
`1111`
`0000`
	`0E <Addr[21:16]>`
	`6E F8`
	`0E <Addr[8:15]>`
	`6E F7`
	`0E <Addr[7:0]>`
	`6E F6`
	`<MSB><LSB>`
	.
	.
	.
	`<MSB><LSB>`
	`00 00`
		`MOVLW   <Addr[21:16]>`
		`MOVWF   TBLPTRU`
		`MOVLW   <Addr[8:15]>`
		`MOVWF   TBLPTRH`
		`MOVLW   <Addr[7:0]>`
		`MOVWF   TBLPTRL`
		`Write 2 bytes and post-increment address by 2`
		
		`Repeat 30 times`
		
		`Write 2 bytes and start programming`
		`NOP - hold PGC high for time P9`

To continue modifying data, repeat Steps 2 through 7, where the address pointer is incremented by 64 at each iteration of the loop.		
Step 8: Disable writes.		
`0000`	`94 A6`	`BCF     EECON1, WREN`

3.3 Data EEPROM Programming (PIC18FXX2X and PIC18FXX8X only)

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

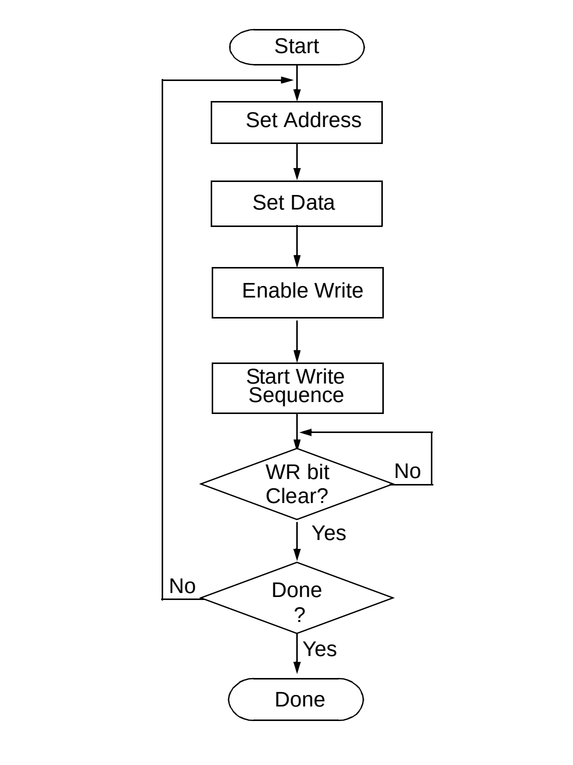


FIGURE 3-7: DATA EEPROM WRITE TIMING

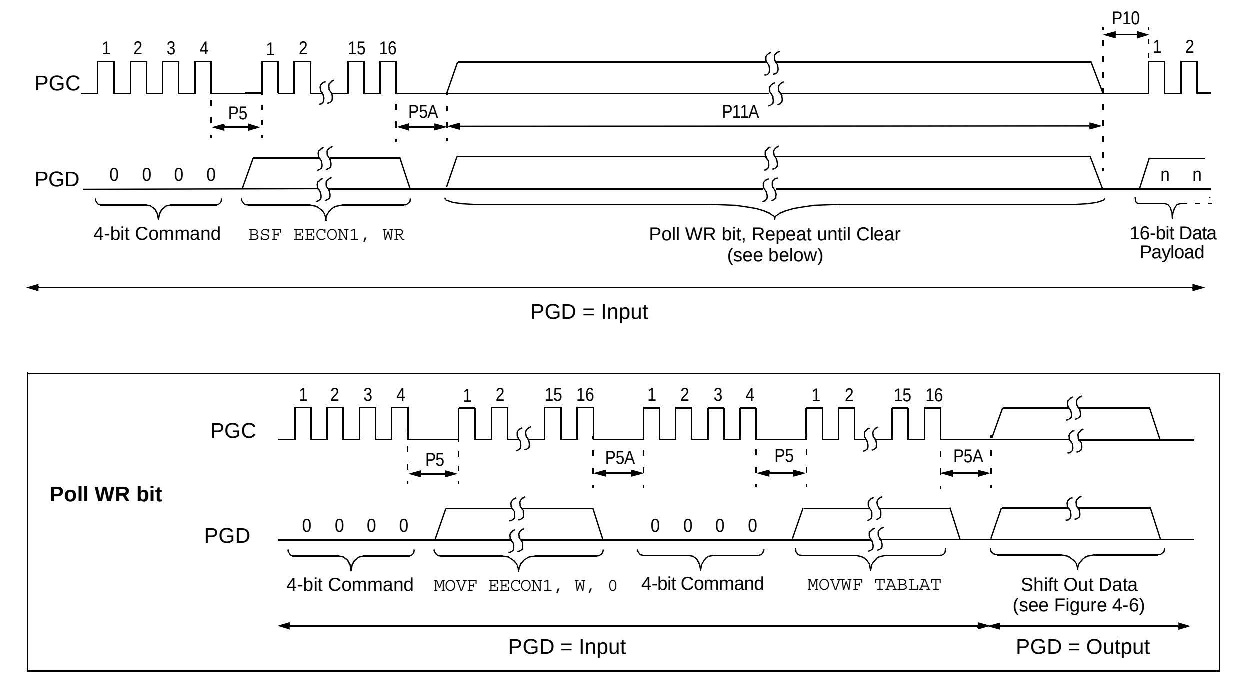


TABLE 3-6: PROGRAMMING DATA MEMORY

4-bit Command	Data Payload	Core Instruction
Step 1: Direct access to data EEPROM.		
`0000`	`9E A6`	`BCF     EECON1, EEPGD`
`0000`	`9C A6`	`BCF     EECON1, CFGS`
Step 2: Set the data EEPROM address pointer.		
`0000`	`0E <Addr>`	`MOVLW   <Addr>`
`0000`	`6E A9`	`MOVWF   EEADR`
`0000`	`OE <AddrH>`	`MOVLW   <AddrH>`
`0000`	`6E AA`	`MOVWF   EEADRH`
Step 3: Load the data to be written.		
`0000`	`0E <Data>`	`MOVLW   <Data>`
`0000`	`6E A8`	`MOVWF   EEDATA`
Step 4: Enable memory writes.		
`0000`	`84 A6`	`BSF     EECON1, WREN`
Step 5: Initiate write.		
`0000`	`82 A6`	`BSF     EECON1, WR`
Step 6: Poll WR bit, repeat until the bit is clear.		
`0000`	`50 A6`	`MOVF    EECON1, W, 0`
`0000`	`6E F5`	`MOVWF   TABLAT`
`0000`	`00 00`	`NOP`
`0010`	`<MSB><LSB>`	`Shift out data`[1]
Step 7: Disable writes.		
`0000`	`94 A6`	`BCF     EECON1, WREN`
Repeat steps 2 through 7 to write more data.		

Note 1: See Figure 4-4 for details on shift out data timing.

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 3-7 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 modified.

TABLE 3-7: WRITE ID SEQUENCE

4-bit Command	Data Payload	Core Instruction
Step 1: Direct access to code memory and enable writes.		

`0000`
`0000`
	`8E A6`
	`9C A6`
		`BSF   EECON1, EEPGD`
		`BCF   EECON1, CFGS`

Step 2: Load write buffer with 8 bytes and write.		

`0000`
`0000`
`0000`
`0000`
`0000`
`0000`
`1101`
`1101`
`1101`
`1111`
`0000`
	`0E 20`
	`6E F8`
	`0E 00`
	`6E F7`
	`0E 00`
	`6E F6`
	`<MSB><LSB>`
	`<MSB><LSB>`
	`<MSB><LSB>`
	`<MSB><LSB>`
	`00 00`
		`MOVLW 20h`
		`MOVWF TBLPTRU`
		`MOVLW 00h`
		`MOVWF TBLPTRH`
		`MOVLW 00h`
		`MOVWF TBLPTRL`
		`Write 2 bytes and post-increment address by 2`
		`Write 2 bytes and post-increment address by 2`
		`Write 2 bytes and post-increment address by 2`
		`Write 2 bytes and start programming`
		`NOP - hold PGC high for time P9`

3.5 Boot Block Programming

The code sequence detailed in Table 3-4 should be used, except that the address data 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 8 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 3-8.

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

TABLE 3-8: SET ADDRESS POINTER TO CONFIGURATION LOCATION

4-bit Command	Data Payload	Core Instruction
Step 1: Enable writes and direct access to config memory.		

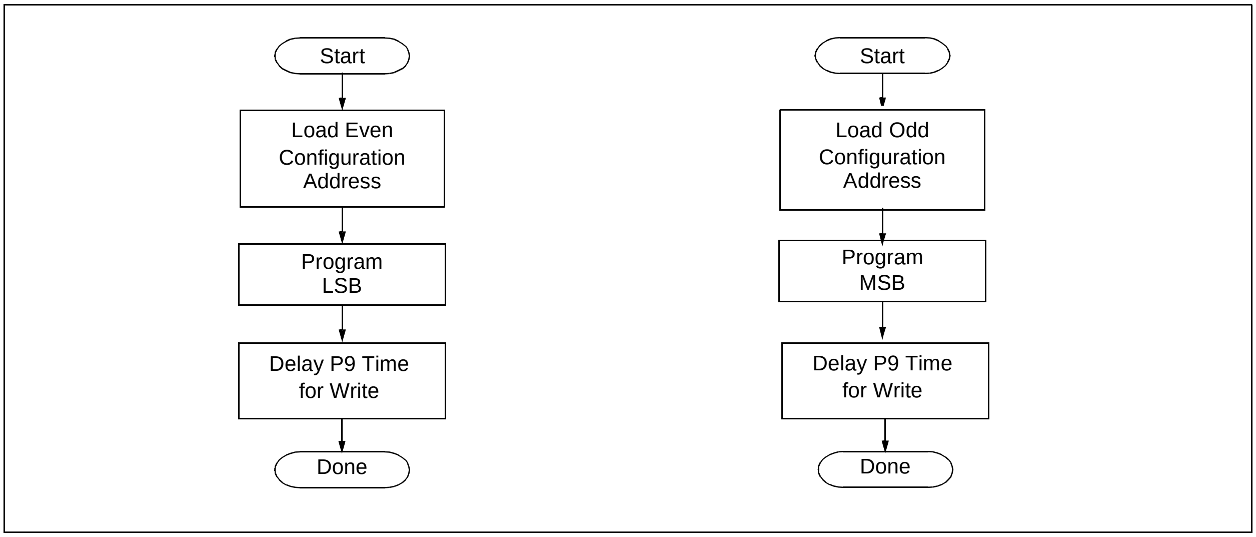
`0000`
`0000`
	`8E A6`
	`8C A6`
		`BSF EECON1, EEPGD`
		`BSF EECON1, CFGS`

Step 2[1]: Set Table Pointer for config byte to be written. Write even/odd addresses.		

`0000`
`0000`
`0000`
`0000`
`0000`
`0000`
`1111`
`0000`
`0000`
`0000`
`1111`
`0000`
	`0E 30`
	`6E F8`
	`0E 00`
	`6E F7`
	`0E 00`
	`6E F6`
	`<MSB ignored><LSB>`
	`00 00`
	`0E 01`
	`6E F6`
	`<MSB><LSB ignored>`
	`00 00`
		`MOVLW 30h`
		`MOVWF TBLPTRU`
		`MOVLW 00h`
		`MOVWF TBLPRTH`
		`MOVLW 00h`
		`MOVWF TBLPTRL`
		`Load 2 bytes and start programming`
		`NOP - hold PGC high for time P9`
		`MOVLW 01h`
		`MOVWF TBLPTRL`
		`Load 2 bytes and start programming`
		`NOP - hold PGC high for time P9`

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

FIGURE 3-8: CONFIGURATION PROGRAMMING FLOW



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) is serially output on PGD.

The 4-bit command is shifted in LSb first. The read is executed during the next 8 clocks, then shifted out on PGD during the last 8 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 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.

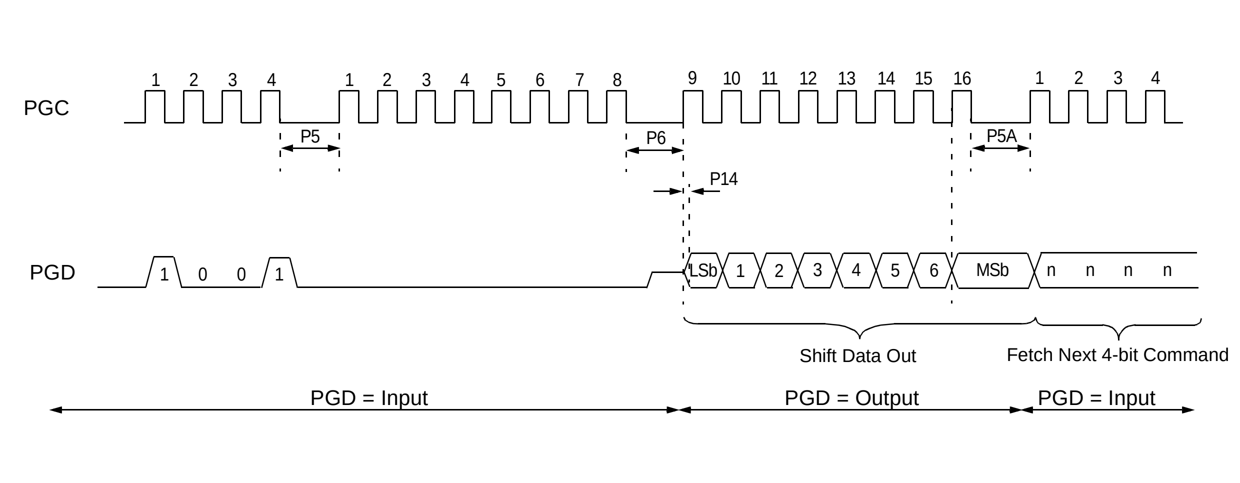
TABLE 4-1: READ CODE MEMORY SEQUENCE

4-bit Command	Data Payload	Core Instruction
Step 1: Set Table Pointer.		

`0000`
`0000`
`0000`
`0000`
`0000`
`0000`
	`0E <Addr[21:16]>`
	`6E F8`
	`0E <Addr[15:8]>`
	`6E F7`
	`0E <Addr[7:0]>`
	`6E F6`
		`MOVLW Addr[21:16]`
		`MOVWF TBLPTRU`
		`MOVLW <Addr[15:8]>`
		`MOVWF TBLPTRH`
		`MOVLW <Addr[7:0]>`
		`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`)

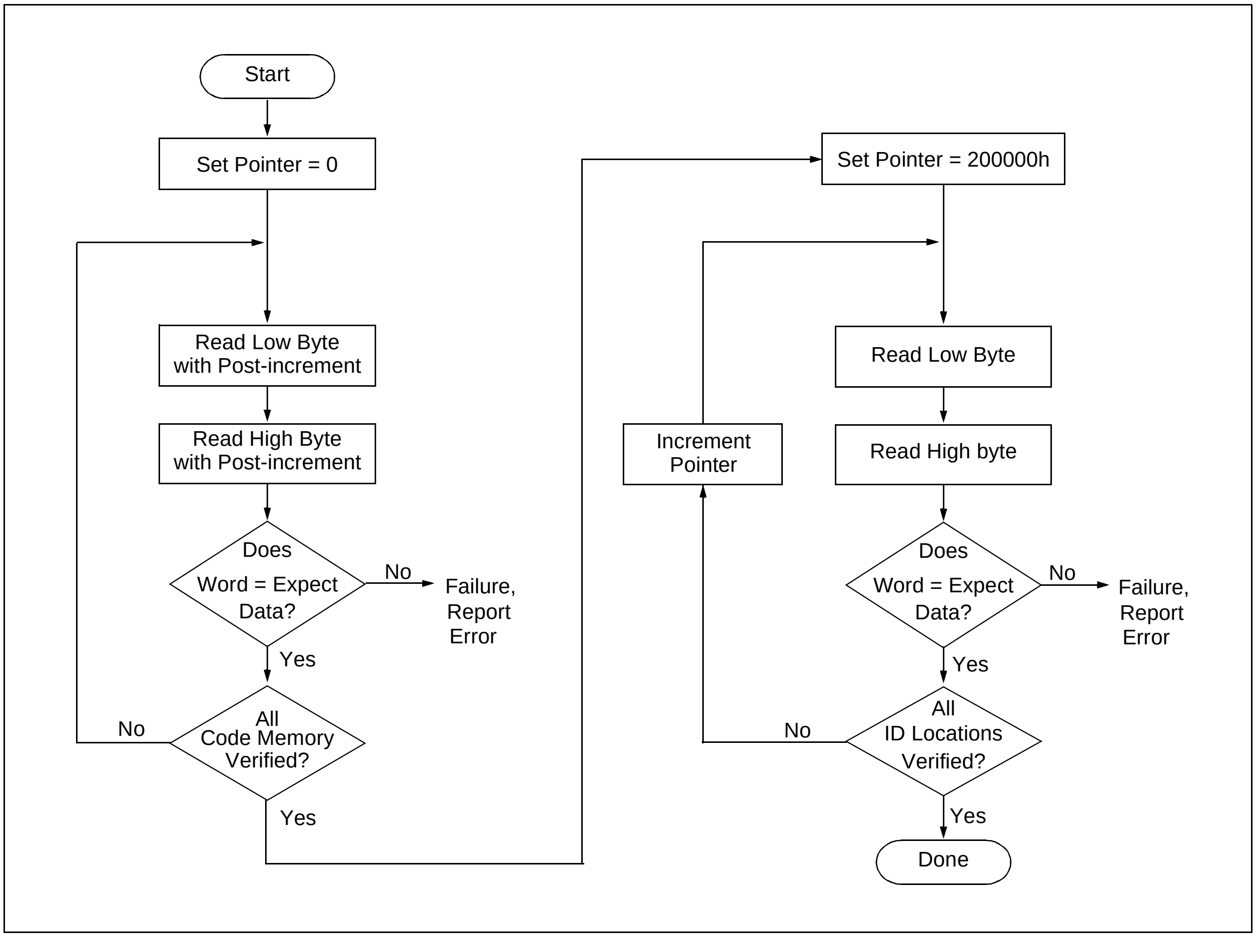


4.2 Verify Code Memory and ID Locations

The verify step involves reading back the code memory space and comparing it against the copy held in the programmer's buffer. Memory reads occur a single byte at a time, so two bytes must be read to compare against the word in the programmer's buffer. Refer to **Section 4.1 "Read Code Memory, ID Locations and Configuration Bits"** for implementation details of reading code memory.

The Table Pointer must be manually set to 200000h (base address of the ID locations) once the code memory has been verified. The post-increment feature of the table read 4-bit command may not be used to increment the Table Pointer beyond the code memory space. In a 64-Kbyte device, for example, a post-increment read of address FFFFh will wrap the Table Pointer back to 000000h, rather than point to unimplemented address 010000h.

FIGURE 4-2: VERIFY CODE MEMORY FLOW



4.3 Verify Configuration Bits

A configuration address may be read and output on PGD via the 4-bit command, ‘`1001`’. Configuration data is read and written in a byte-wise fashion, so it is not necessary to merge two bytes into a word prior to a compare. The result may then be immediately compared to the appropriate configuration data in the programmer’s memory for verification. Refer to **Section 4.1 “Read Code Memory, ID Locations and Configuration Bits”** for implementation details of reading configuration data.

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

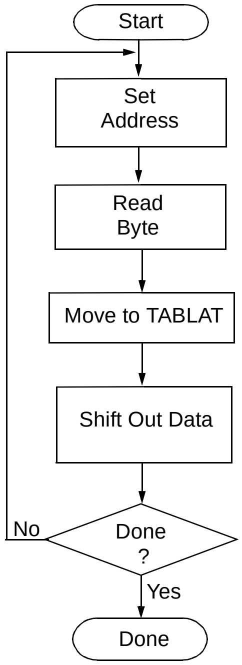


TABLE 4-2: READ DATA EEPROM MEMORY

4-bit Command	Data Payload	Core Instruction
Step 1: Direct access to data EEPROM.		

`0000`
`0000`
	`9E A6`
	`9C A6`
		`BCF     EECON1, EEPGD`
		`BCF     EECON1, CFGS`

Step 2: Set the data EEPROM address pointer.		

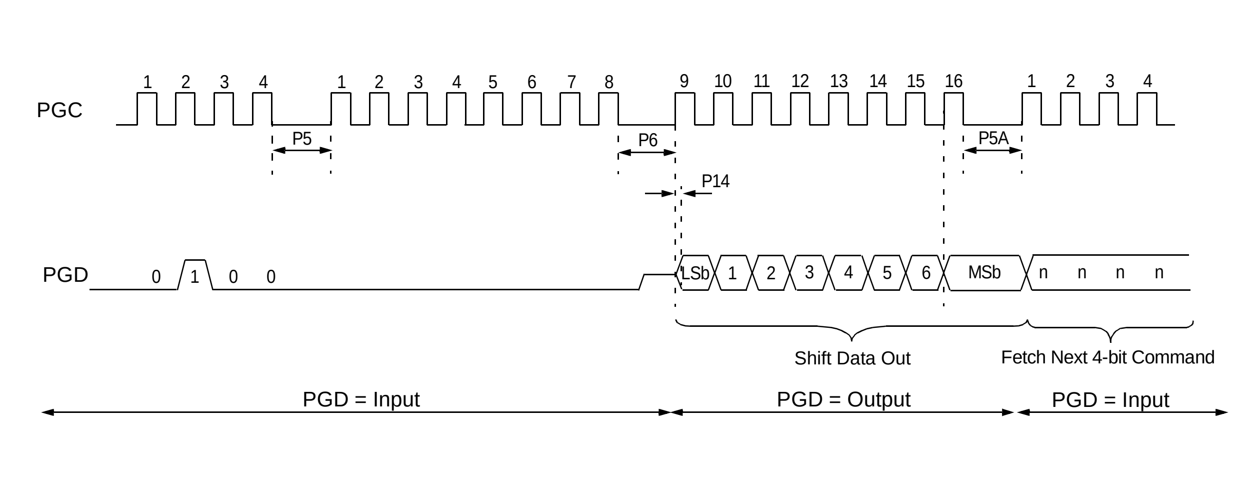
`0000`
`0000`
`0000`
`0000`
	`0E <Addr>`
	`6E A9`
	`OE <AddrH>`
	`6E AA`
		`MOVLW   <Addr>`
		`MOVWF   EEADR`
		`MOVLW   <AddrH>`
		`MOVWF   EEADRH`

Step 3: Initiate a memory read.		
`0000`	`80 A6`	`BSF     EECON1, RD`
Step 4: Load data into the Serial Data Holding register.		

`0000`
`0000`
`0000`
`0010`
	`50 A8`
	`6E F5`
	`00 00`
	`<MSB><LSB>`
		`MOVF    EEDATA, W, 0`
		`MOVWF   TABLAT`
		`NOP`
		`Shift Out Data`[1]

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

FIGURE 4-4: SHIFT OUT DATA HOLDING REGISTER TIMING (`0010`)



4.5 Verify Data EEPROM

A data EEPROM address may be read via a sequence of core instructions (4-bit command, '`0000`') and then output on PGD via the 4-bit command, '`0010`' (Shift Out Data Holding register). The result may then be immediately compared to the appropriate data in the programmer's memory for verification. Refer to **Section 4.4 "Read Data EEPROM Memory"** for implementation details of reading data EEPROM.

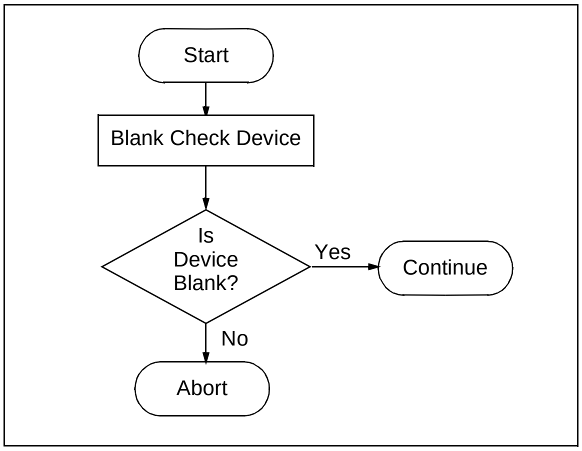
4.6 Blank Check

The term "Blank Check" means to verify that the device has no programmed memory cells. All memories must be verified: code memory, data EEPROM, ID locations and configuration bits. The Device ID registers (3FFFFEh:3FFFFFh) should be ignored.

A "blank" or "erased" memory cell will read as a '`1`'. So, "Blank Checking" a device merely means to verify that all bytes read as FFh except the configuration bits. Unused (reserved) configuration bits will read '`0`' (programmed). Refer to Table 5-2 for blank configuration expect data for the various PIC18FX5X5/X6X0 devices.

Given that "Blank Checking" is merely code and data EEPROM verification with FFh expect data, refer to **Section 4.4 "Read Data EEPROM Memory"** and **Section 4.2 "Verify Code Memory and ID Locations"** for implementation details.

FIGURE 4-5: BLANK CHECK FLOW



5.0 CONFIGURATION WORD

The PIC18FX5X5/X6X0 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 configuration words. These bits may be read out normally, even after read or code-protected.

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 0Fh. 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 PIC18FX5X5/X6X0 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-protected.

5.3 Single-Supply ICSP Programming

The LVP bit in Configuration register, CONFIG4L, enables Single-Supply (Low-Voltage) ICSP Programming. The LVP bit defaults to a '1' from the factory.

If Single-Supply Programming mode is not used, the LVP bit can be programmed to a '0' and RB5/PGM becomes a digital I/O pin. However, the LVP bit may only be programmed by entering the High-Voltage ICSP mode, where $\overline{\text{MCLR}}/V_{PP}$ is raised to V_{IHH}. Once the LVP bit is programmed to a '0', only the High-Voltage ICSP mode is available and only the High-Voltage ICSP mode can be used to program the device.

Note 1: The normal ICSP mode is always available, regardless of the state of the LVP bit, by applying V_{IHH} to the $\overline{\text{MCLR}}/V_{PP}$ pin.

2: While in Low-Voltage ICSP mode, the RB5 pin can no longer be used as a general purpose I/O.

TABLE 5-1: DEVICE ID VALUE

Device	Device ID Value	
	DEVID2	DEVID1
PIC18F2515	0Ch	111x xxxx
PIC18F2525	0Ch	110x xxxx
PIC18F2585	0Eh	111x xxxx
PIC18F4515	0Ch	011x xxxx
PIC18F4525	0Ch	010x xxxx
PIC18F4585	0Eh	101x xxxx
PIC18F2610	0Ch	101x xxxx
PIC18F2620	0Ch	100x xxxx
PIC18F2680	0Eh	110x xxxx
PIC18F4610	0Ch	001x xxxx
PIC18F4620	0Ch	000x xxxx
PIC18F4680	0Eh	100x xxxx

Note: The 'x's in DEVID1 contain the device revision code.

TABLE 5-2: 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
300001h	CONFIG1H	IESO	FCMEN	—	—	FOSC3	FOSC2	FOSC1	FOSC0	`00-- 0111`
300002h	CONFIG2L	—	—	—	BORV1	BORV0	BOREN1	BOREN0	$\overline{\text{PWRTEN}}$	`---1 1111`
300003h	CONFIG2H	—	—	—	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDTEN	`---1 1111`
300005h	CONFIG3H	MCLRE	—	—	—	—	LPT1OSC	PBADEN	CCP2MX —[2]	`1--- -011` `1--- -01-`[2]
300006h	CONFIG4L	$\overline{\text{DEBUG}}$	XINST	—	—	—	LVP	—	STVREN	`10-- -1-1`
300008h	CONFIG5L	—	—	—	—	CP3	CP2	CP1	CP0	`---- 1111`
300009h	CONFIG5H	CPD	CPB	—	—	—	—	—	—	`11-- ----`
30000Ah	CONFIG6L	—	—	—	—	WRT3	WRT2	WRT1	WRT0	`---- 1111`
30000Bh	CONFIG6H	WRTD	WRTB	WRTC	—	—	—	—	—	`111- ----`
30000Ch	CONFIG7L	—	—	—	—	EBTR3	EBTR2	EBTR1	EBTR0	`---- 1111`
30000Dh	CONFIG7H	—	EBTRB	—	—	—	—	—	—	`-1-- ----`
3FFFFEh	DEVID1[1]	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	`xxxx xxxx`[1]
3FFFFFh	DEVID2[1]	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	`0000 1100`[3] `0000 1110`[2]

Legend: `x` = unknown, `u` = unchanged, - = unimplemented, `q` = value depends on condition.
Shaded cells are unimplemented, read as ‘`0`’.

Note 1: DEVID registers are read-only and cannot be programmed by the user.

2: PIC18FXX8X devices only.

3: PIC18FXX1X and PIC18FXX2X devices only.

5.4 Embedding Configuration Word Information in the HEX File

To allow portability of code, a PIC18FX5X5/X6X0 programmer is required to read the configuration word locations from the hex file. If configuration word information is not present in the hex file, then a simple warning message should be issued. Similarly, while saving a hex file, all configuration word information must be included. An option to not include the configuration word information may be provided. When embedding configuration word information in the hex file, it should start at address 300000h.

Microchip Technology Inc. feels strongly that this feature is important for the benefit of the end customer.

5.5 Embedding Data EEPROM Information In the HEX File

To allow portability of code, a PIC18FX5X5/X6X0 programmer is required to read the data EEPROM information from the hex file. If data EEPROM information is not present, a simple warning message should be issued. Similarly, when saving a hex file, all data EEPROM information must be included. An option to not include the data EEPROM information may be provided. When embedding data EEPROM information in the hex file, it should start at address F00000h.

Microchip Technology Inc. believes that this feature is important for the benefit of the end customer.

5.6 Checksum Computation

The checksum is calculated by summing the following:

- The contents of all code memory locations
- The configuration word, appropriately masked
- ID locations

The Least Significant 16 bits of this sum are the checksum.

Table 5-3 (pages 27 through 32) describes how to calculate the checksum for each device.

Note: The checksum calculation differs depending on the code-protect setting. Since the code memory locations read out differently depending on the code-protect setting, the table describes how to manipulate the actual code memory values to simulate the values that would be read from a protected device. When calculating a checksum by reading a device, the entire code memory can simply be read and summed. The configuration word and ID locations can always be read.

TABLE 5-3: CHECKSUM COMPUTATION

Device	Code-Protect	Checksum	Blank Value	0xAA at 0 and Max Address
PIC18F2515	None	SUM(0000:07FF)+SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)	0466	03BC
	Boot Block	SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0C36	0BEB
	Boot/Block1/Block2	SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	8433	83E8
	All	(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0427	0431
PIC18F2525	None	SUM(0000:07FF)+SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)	0466	03BC
	Boot Block	SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0C36	0BEB
	Boot/Block1/Block2	SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	8433	83E8
	All	(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0427	0431

Legend:

Item		Description
CFGW	=	Configuration Word
SUM[a:b]	=	Sum of locations, a to b inclusive
SUM_ID	=	Byte-wise sum of lower four bits of all customer ID locations
+	=	Addition
&	=	Bit-wise AND

TABLE 5-3: CHECKSUM COMPUTATION (CONTINUED)

Device	Code-Protect	Checksum	Blank Value	0xAA at 0 and Max Address
PIC18F2585	None	SUM(0000:07FF)+SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0086)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)	0465	03BB
	Boot Block	SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0086)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0C34	0BE9
	Boot/Block1/Block2	SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0086)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	8431	83E6
	All	(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0086)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0425	042F
PIC18F2610	None	SUM(0000:07FF)+SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)	0466	03BC
	Boot Block	SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0C36	0BEB
	Boot/Block1/Block2	SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	8433	83E8
	All	(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0427	0431

Legend:

Item		Description
CFGW	=	Configuration Word
SUM[a:b]	=	Sum of locations, a to b inclusive
SUM_ID	=	Byte-wise sum of lower four bits of all customer ID locations
+	=	Addition
&	=	Bit-wise AND

TABLE 5-3: CHECKSUM COMPUTATION (CONTINUED)

Device	Code-Protect	Checksum	Blank Value	0xAA at 0 and Max Address
PIC18F2620	None	SUM(0000:07FF)+SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)	0466	03BC
	Boot Block	SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0C36	0BEB
	Boot/ Block1/ Block2	SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	8433	83E8
	All	(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0427	0431
PIC18F2680	None	SUM(0000:07FF)+SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0086)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)	0465	03BB
	Boot Block	SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0086)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0C34	0BE9
	Boot/ Block1/ Block2	SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0086)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	8431	83E6
	All	(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0086)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0425	042F

Legend:

Item		Description
CFGW	=	Configuration Word
SUM[a:b]	=	Sum of locations, a to b inclusive
SUM_ID	=	Byte-wise sum of lower four bits of all customer ID locations
+	=	Addition
&	=	Bit-wise AND

TABLE 5-3: CHECKSUM COMPUTATION (CONTINUED)

Device	Code-Protect	Checksum	Blank Value	0xAA at 0 and Max Address
PIC18F4515	None	SUM(0000:07FF)+SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+ SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+ (CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+ (CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+ (CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+ (CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)	0466	03BC
	Boot Block	SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+ (CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+ (CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+ (CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+ (CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+ (CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0C36	0BEB
	Boot/ Block1/ Block2	SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+ (CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+ (CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+ (CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+ (CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	8433	83E8
	All	(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+ (CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+ (CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+ (CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+ (CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0427	0431
PIC18F4525	None	SUM(0000:07FF)+SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+ SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+ (CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+ (CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+ (CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+ (CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)	0466	03BC
	Boot Block	SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+ (CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+ (CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+ (CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+ (CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+ (CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0C36	0BEB
	Boot/ Block1/ Block2	SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+ (CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+ (CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+ (CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+ (CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	8433	83E8
	All	(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+ (CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+ (CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+ (CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+ (CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0427	0431

Legend:

Item		Description
CFGW	=	Configuration Word
SUM[a:b]	=	Sum of locations, a to b inclusive
SUM_ID	=	Byte-wise sum of lower four bits of all customer ID locations
+	=	Addition
&	=	Bit-wise AND

TABLE 5-3: CHECKSUM COMPUTATION (CONTINUED)

Device	Code-Protect	Checksum	Blank Value	0xAA at 0 and Max Address
PIC18F4585	None	SUM(0000:07FF)+SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0086)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)	0465	03BB
	Boot Block	SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0086)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0C34	0BE9
	Boot/ Block1/ Block2	SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0086)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	8431	83E6
	All	(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0086)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0425	042F
PIC18F4610	None	SUM(0000:07FF)+SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)	0466	03BC
	Boot Block	SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0C36	0BEB
	Boot/ Block1/ Block2	SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	8433	83E8
	All	(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0427	0431

Legend:	Item		Description
	CFGW	=	Configuration Word
	SUM[a:b]	=	Sum of locations, a to b inclusive
	SUM_ID	=	Byte-wise sum of lower four bits of all customer ID locations
	+	=	Addition
	&	=	Bit-wise AND

TABLE 5-3: CHECKSUM COMPUTATION (CONTINUED)

Device	Code-Protect	Checksum	Blank Value	0xAA at 0 and Max Address
PIC18F4620	None	SUM(0000:07FF)+SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)	0466	03BC
	Boot Block	SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0C36	0BEB
	Boot/ Block1/ Block2	SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	8433	83E8
	All	(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0087)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0427	0431
PIC18F4680	None	SUM(0000:07FF)+SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0086)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)	0465	03BB
	Boot Block	SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0086)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0C34	0BE9
	Boot/ Block1/ Block2	SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0086)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	8431	83E6
	All	(CONFIG0 & 0000)+(CONFIG1 & 00CF)+(CONFIG2 & 001F)+(CONFIG3 & 001F)+(CONFIG4 & 0000)+(CONFIG5 & 0086)+(CONFIG6 & 00C5)+(CONFIG7 & 0000)+(CONFIG8 & 000F)+(CONFIG9 & 00C0)+(CONFIG10 & 000F)+(CONFIG11 & 00E0)+(CONFIG12 & 000F)+(CONFIG13 & 0040)+SUM(IDs)	0425	042F

Legend:

Item		Description
CFGW	=	Configuration Word
SUM[a:b]	=	Sum of locations, a to b inclusive
SUM_ID	=	Byte-wise sum of lower four bits of all customer ID locations
+	=	Addition
&	=	Bit-wise AND

6.0 AC/DC CHARACTERISTICS TIMING REQUIREMENTS FOR PROGRAM/VERIFY TEST MODE

Standard Operating Conditions
Operating Temperature: 25°C is recommended

Param No.	Sym	Characteristic	Min	Max	Units	Conditions
D110	V_{IHH}	High-Voltage Programming Voltage on $\overline{MCLR}/V_{PP}$	9.00	13.25	V	
D110A	V_{IHL}	Low-Voltage Programming Voltage on $\overline{MCLR}/V_{PP}$	2.00	5.50	V	
D111	V_{DD}	Supply Voltage During Programming	2.00	5.50	V	Normal programming
			4.50	5.50	V	Bulk erase operations
D112	I_{PP}	Programming Current on $\overline{MCLR}/V_{PP}$	—	300	μA	
D113	I_{DDP}	Supply Current During Programming	—	10	mA	
D031	V_{IL}	Input Low Voltage	V_{SS}	0.2 V_{DD}	V	
D041	V_{IH}	Input High Voltage	0.8 V_{DD}	V_{DD}	V	
D080	V_{OL}	Output Low Voltage	—	0.6	V	I_{OL} = 8.5 mA @ 4.5V
D090	V_{OH}	Output High Voltage	V_{DD} – 0.7	—	V	I_{OH} = -3.0 mA @ 4.5V
D012	C_{IO}	Capacitive Loading on I/O pin (PGD)	—	50	pF	To meet AC specifications
P1	T_R	$\overline{MCLR}/V_{PP}$ Rise Time to enter Program/Verify mode	—	1.0	μs	**(Note 1)**
P2	T_{PGC}	Serial Clock (PGC) Period	100	—	ns	V_{DD} = 5.0V
			1	—	μs	V_{DD} = 2.0V
P2A	T_{PGCL}	Serial Clock (PGC) Low Time	40	—	ns	V_{DD} = 5.0V
			400	—	ns	V_{DD} = 2.0V
P2B	T_{PGCH}	Serial Clock (PGC) High Time	40	—	ns	V_{DD} = 5.0V
			400	—	ns	V_{DD} = 2.0V
P3	T_{SET1}	Input Data Setup Time to Serial Clock ↓	15	—	ns	
P4	T_{HLD1}	Input Data Hold Time from PGC ↓	15	—	ns	
P5	T_{DLY1}	Delay between 4-bit Command and Command Operand	40	—	ns	
P5A	T_{DLY1A}	Delay between 4-bit Command Operand and next 4-bit Command	40	—	ns	
P6	T_{DLY2}	Delay between Last PGC ↓ of Command Byte to First PGC ↑ of Read of Data Word	20	—	ns	
P9	T_{DLY5}	PGC High Time (minimum programming time)	1	—	ms	
P10	T_{DLY6}	PGC Low Time after Programming (high-voltage discharge time)	40	—	μs	
P11	T_{DLY7}	Delay to allow Self-Timed Data Write or Bulk Erase to occur	5	—	ms	
P11A	T_{DRWT}	Data Write Polling Time	4	—	ms	
P12	T_{HLD2}	Input Data Hold Time from $\overline{MCLR}/V_{PP}$ ↑	2	—	μs	
P13	T_{SET2}	V_{DD} ↑ Setup Time to $\overline{MCLR}/V_{PP}$ ↑	100	—	ns	
P14	T_{VALID}	Data Out Valid from PGC ↑	10	—	ns	
P15	T_{SET3}	PGM ↑ Setup Time to $\overline{MCLR}/V_{PP}$ ↑	2	—	μs	

Note 1: Do not allow excess time when transitioning $\overline{MCLR}$ between V_{IL} and V_{IHH}; this can cause spurious program executions to occur. The maximum transition time is:
1 T_{CY} + T_{PWRT} (if enabled) + 1024 T_{OSC} (for LP, HS, HS/PLL and XT modes only) +
2 ms (for HS/PLL mode only) + 1.5 μs (for EC mode only)
where T_{CY} is the instruction cycle time, T_{PWRT} is the Power-up Timer period, and T_{OSC} is the oscillator period. For specific values, refer to the Electrical Characteristics section of the device data sheet for the particular device.

NOTES:

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

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

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

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip’s products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights.

Trademarks

The Microchip name and logo, the Microchip logo, Accuron, dsPIC, KEELOQ, MPLAB, PIC, PICmicro, PICSTART, PRO MATE and PowerSmart are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

AmpLab, FilterLab, microID, MXDEV, MXLAB, PICMASTER, SEEVAL, SmartShunt and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Application Maestro, dsPICDEM, dsPICDEM.net, dsPICworks, ECAN, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, Migratable Memory, MPASM, MPLIB, MPLINK, MPSIM, PICkit, PICDEM, PICDEM.net, PICtail, PowerCal, PowerInfo, PowerMate, PowerTool, rfLAB, rfPIC, Select Mode, SmartSensor, SmartTel and Total Endurance are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

Serialized Quick Turn Programming (SQTP) is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

♻ Printed on recycled paper.

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
ISO/TS 16949:2002

Microchip received ISO/TS-16949:2002 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona and Mountain View, California in October 2003. The Company’s quality system processes and procedures are for its PICmicro® 8-bit MCUs, 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.

WORLDWIDE SALES AND SERVICE

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support: 480-792-7627
Web Address: http://www.microchip.com

Atlanta
3780 Mansell Road, Suite 130
Alpharetta, GA 30022
Tel: 770-640-0034
Fax: 770-640-0307

Boston
2 Lan Drive, Suite 120
Westford, MA 01886
Tel: 978-692-3848
Fax: 978-692-3821

Chicago
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
4570 Westgrove Drive, Suite 160
Addison, TX 75001
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Tri-Atria Office Building
32255 Northwestern Highway, Suite 190
Farmington Hills, MI 48334
Tel: 248-538-2250
Fax: 248-538-2260

Kokomo
2767 S. Albright Road
Kokomo, IN 46902
Tel: 765-864-8360
Fax: 765-864-8387

Los Angeles
18201 Von Karman, Suite 1090
Irvine, CA 92612
Tel: 949-263-1888
Fax: 949-263-1338

San Jose
1300 Terra Bella Avenue
Mountain View, CA 94043
Tel: 650-215-1444
Fax: 650-961-0286

Toronto
6285 Northam Drive, Suite 108
Mississauga, Ontario L4V 1X5, Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Australia
Suite 22, 41 Rawson Street
Epping 2121, NSW
Australia
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Unit 706B
Wan Tai Bei Hai Bldg.
No. 6 Chaoyangmen Bei Str.
Beijing, 100027, China
Tel: 86-10-85282100
Fax: 86-10-85282104

China - Chengdu
Rm. 2401-2402, 24th Floor,
Ming Xing Financial Tower
No. 88 TIDU Street
Chengdu 610016, China
Tel: 86-28-86766200
Fax: 86-28-86766599

China - Fuzhou
Unit 28F, World Trade Plaza
No. 71 Wusi Road
Fuzhou 350001, China
Tel: 86-591-7503506
Fax: 86-591-7503521

China - Hong Kong SAR
Unit 901-6, Tower 2, Metroplaza
223 Hing Fong Road
Kwai Fong, N.T., Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

China - Shanghai
Room 701, Bldg. B
Far East International Plaza
No. 317 Xian Xia Road
Shanghai, 200051
Tel: 86-21-6275-5700
Fax: 86-21-6275-5060

China - Shenzhen
Rm. 1812, 18/F, Building A, United Plaza
No. 5022 Binhe Road, Futian District
Shenzhen 518033, China
Tel: 86-755-82901380
Fax: 86-755-8295-1393

China - Shunde
Room 401, Hongjian Building, No. 2
Fengxiangnan Road, Ronggui Town, Shunde
District, Foshan City, Guangdong 528303, China
Tel: 86-757-28395507 Fax: 86-757-28395571

China - Qingdao
Rm. B505A, Fullhope Plaza,
No. 12 Hong Kong Central Rd.
Qingdao 266071, China
Tel: 86-532-5027355 Fax: 86-532-5027205

India
Divyasree Chambers
1 Floor, Wing A (A3/A4)
No. 11, O'Shaugnessey Road
Bangalore, 560 025, India
Tel: 91-80-2290061 Fax: 91-80-2290062

Japan
Benex S-1 6F
3-18-20, Shinyokohama
Kohoku-Ku, Yokohama-shi
Kanagawa, 222-0033, Japan
Tel: 81-45-471- 6166 Fax: 81-45-471-6122

Korea
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku
Seoul, Korea 135-882
Tel: 82-2-554-7200 Fax: 82-2-558-5932 or
82-2-558-5934

Singapore
200 Middle Road
#07-02 Prime Centre
Singapore, 188980
Tel: 65-6334-8870 Fax: 65-6334-8850

Taiwan
Kaohsiung Branch
30F - 1 No. 8
Min Chuan 2nd Road
Kaohsiung 806, Taiwan
Tel: 886-7-536-4818
Fax: 886-7-536-4803

Taiwan
Taiwan Branch
11F-3, No. 207
Tung Hua North Road
Taipei, 105, Taiwan
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139

EUROPE

Austria
Durisolstrasse 2
A-4600 Wels
Austria
Tel: 43-7242-2244-399
Fax: 43-7242-2244-393

Denmark
Regus Business Centre
Lautrup hoj 1-3
Ballerup DK-2750 Denmark
Tel: 45-4420-9895 Fax: 45-4420-9910

France
Parc d'Activite du Moulin de Massy
43 Rue du Saule Trapu
Batiment A - ler Etage
91300 Massy, France
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany
Steinheilstrasse 10
D-85737 Ismaning, Germany
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy
Via Quasimodo, 12
20025 Legnano (MI)
Milan, Italy
Tel: 39-0331-742611
Fax: 39-0331-466781

Netherlands
P. A. De Biesbosch 14
NL-5152 SC Drunen, Netherlands
Tel: 31-416-690399
Fax: 31-416-690340

United Kingdom
505 Eskdale Road
Winnersh Triangle
Wokingham
Berkshire, England RG41 5TU
Tel: 44-118-921-5869
Fax: 44-118-921-5820

01/26/04