PIC18F6X2X/8X2X

Flash Microcontroller Programming Specification

1.0 DEVICE OVERVIEW

This document includes the programming specifications for the following devices:

- PIC18F6525
- PIC18F6621
- PIC18F8525
- PIC18F8621

2.0 PROGRAMMING OVERVIEW OF THE PIC18F6X2X/8X2X

PIC18F6X2X/8X2X devices can be programmed using either the high voltage In-Circuit Serial Programming™ (ICSP™) method, or the low voltage ICSP method. Both of these 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 PIC18F6X2X/8X2X devices in all package types.

2.1 Hardware Requirements

In High Voltage ICSP mode, PIC18F6X2X/8X2X 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, PIC18F6X2X/8X2X devices can be programmed using a VDD source in the operating range. This only means that $\overline{\text{MCLR}}$/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 PIC18F6X2X/8X2X family are shown in Figure 2-1 and Figure 2-2. The pin descriptions of these diagrams do not represent the complete functionality of the device types. Users should refer to the appropriate device data sheet for complete pin descriptions.

TABLE 2-1: PIN DESCRIPTIONS (DURING PROGRAMMING): PIC18F6X2X/8X2X

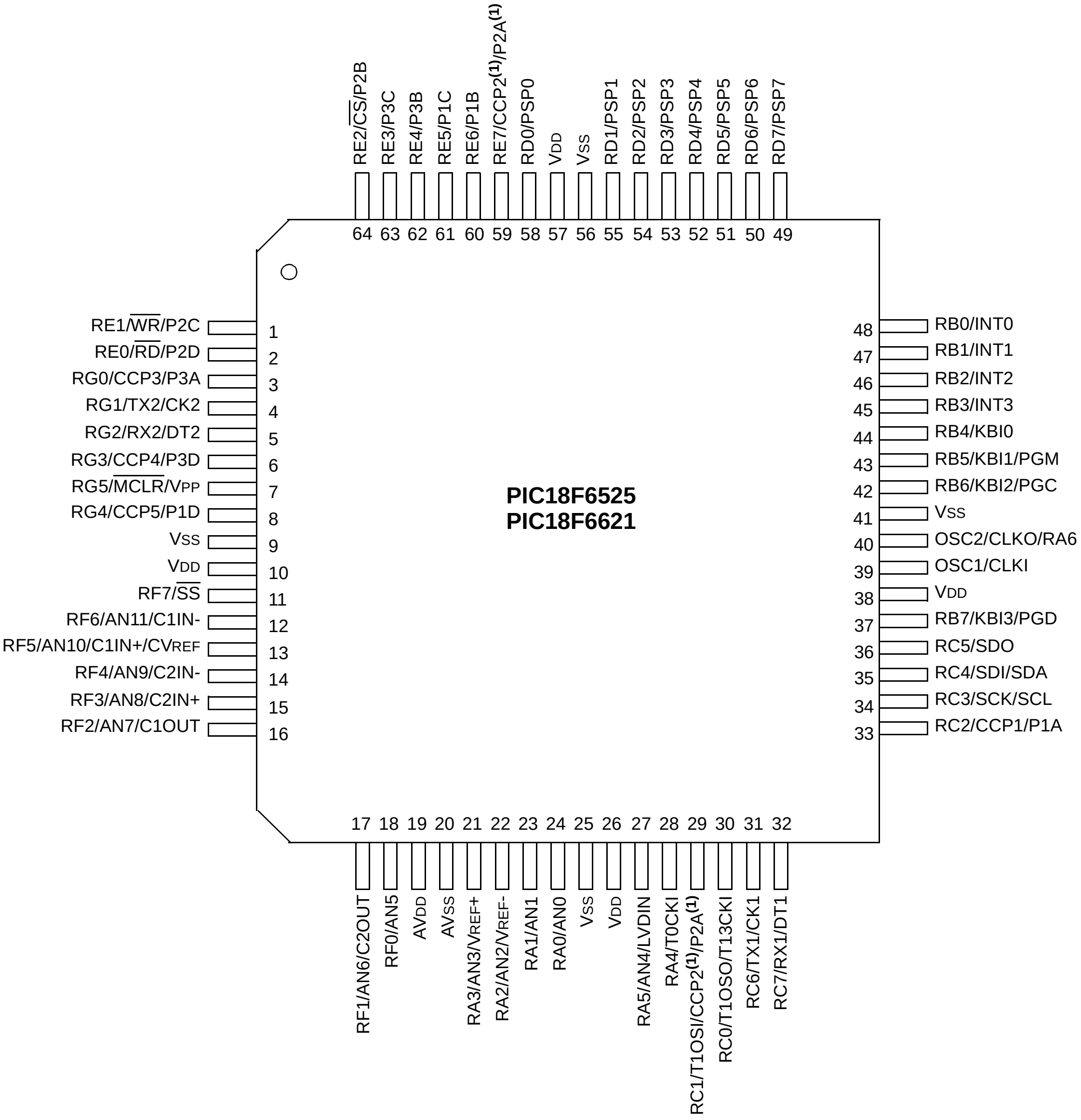
Pin Name	During Programming		
	Pin Name	Pin Type	Pin Description
$\overline{\text{MCLR}}$/VPP/RG5	VPP	P	Programming Enable
VDD(2)	VDD	P	Power Supply
VSS(2)	VSS	P	Ground
AVDD(2)	AVDD	P	Analog Power Supply
AVSS(2)	AVSS	P	Analog Ground
RB5	PGM	I	Low Voltage ICSP™ Input when LVP Configuration bit equals '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 "Low Voltage Programming (LVP) Bit"** for more detail.

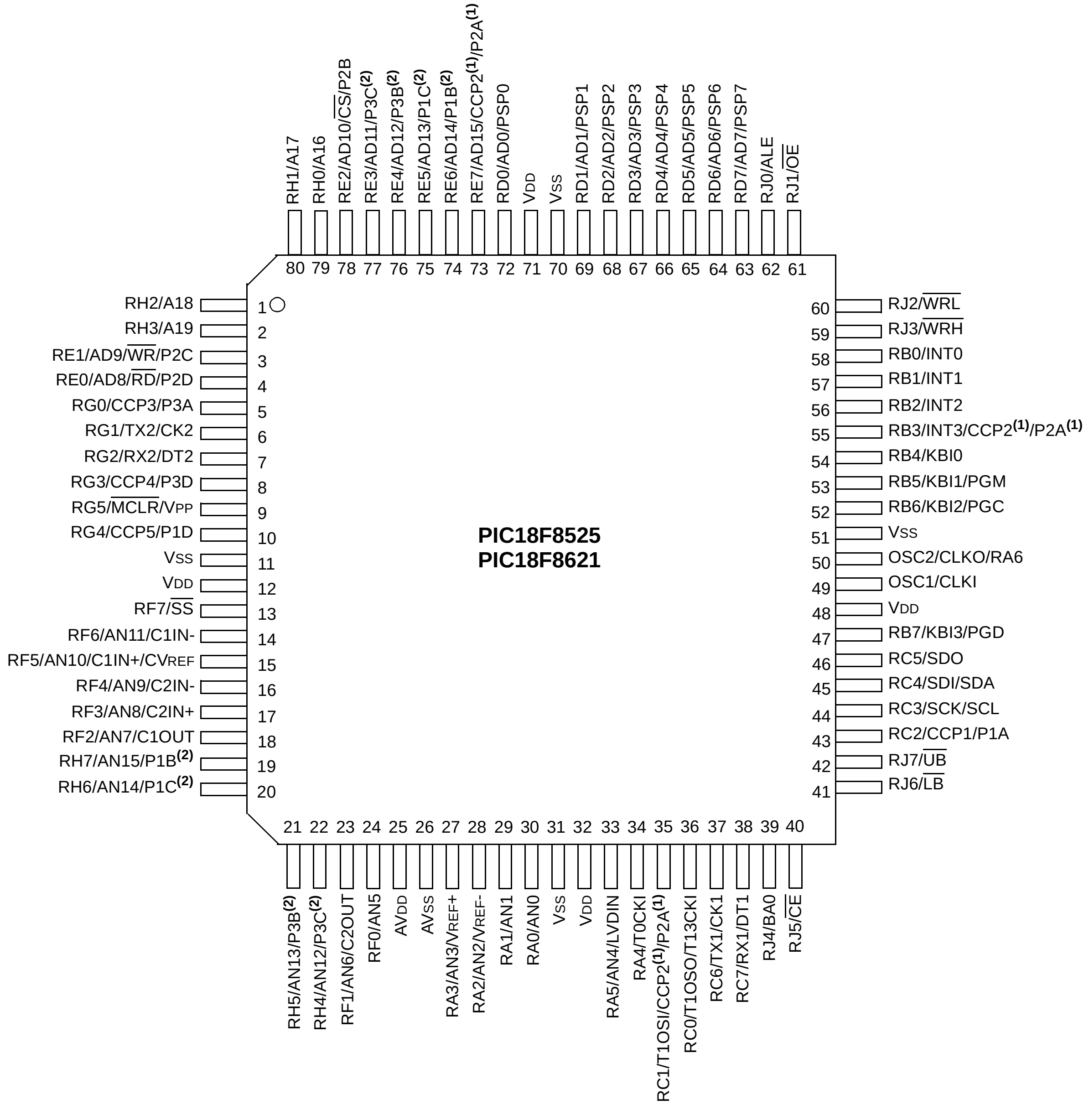
2: All power supply (VDD/AVDD) and ground (VSS/AVSS) must be connected.

FIGURE 2-1: 64-PIN TQFP PIN DIAGRAM (PIC18F6X2X)



Note 1: The CCP2 pin placement depends on the CCP2MX fuse setting.

FIGURE 2-2: 80-PIN TQFP PIN DIAGRAM (PIC18F8X2X)



Note 1: The CCP2 pin placement depends on the CCP2MX fuse and Processor mode settings.

2: P1B, P1C, P3B, and P3C pin placement depends on the ECCPMUX fuse setting.

2.3 Memory Map

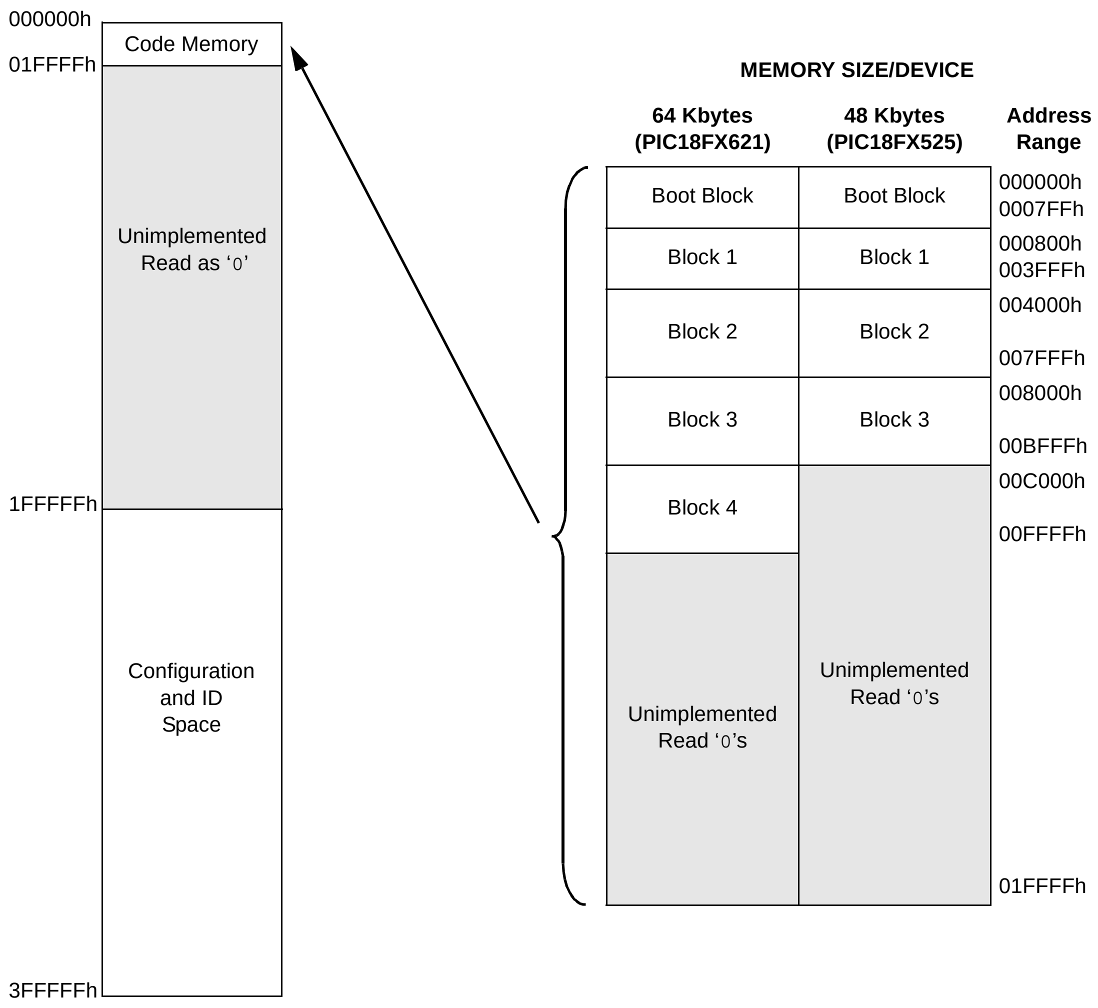
The code memory space extends from 0000h to 0FFFFh (64 Kbytes) in four 16-Kbyte blocks. Addresses 0000h through 07FFh, however, define a "Boot Block" region that is treated separately from Block 1. All of these blocks define code protection boundaries within the code memory space.

In contrast, code memory panels are defined in 8-Kbyte boundaries. Panels are discussed in greater detail in **Section 3.2 "Code Memory Programming"**.

TABLE 2-2: IMPLEMENTATION OF CODE MEMORY

Device	Code Memory Size (Bytes)
PIC18F6525	000000h - 00BFFFh (48K)
PIC18F8525	
PIC18F6621	000000h - 00FFFFh (64K)
PIC18F8621	

FIGURE 2-3: MEMORY MAP AND THE CODE MEMORY SPACE FOR PIC18F6X2X/8X2X DEVICES



Note: Sizes of memory areas 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 2-4.

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

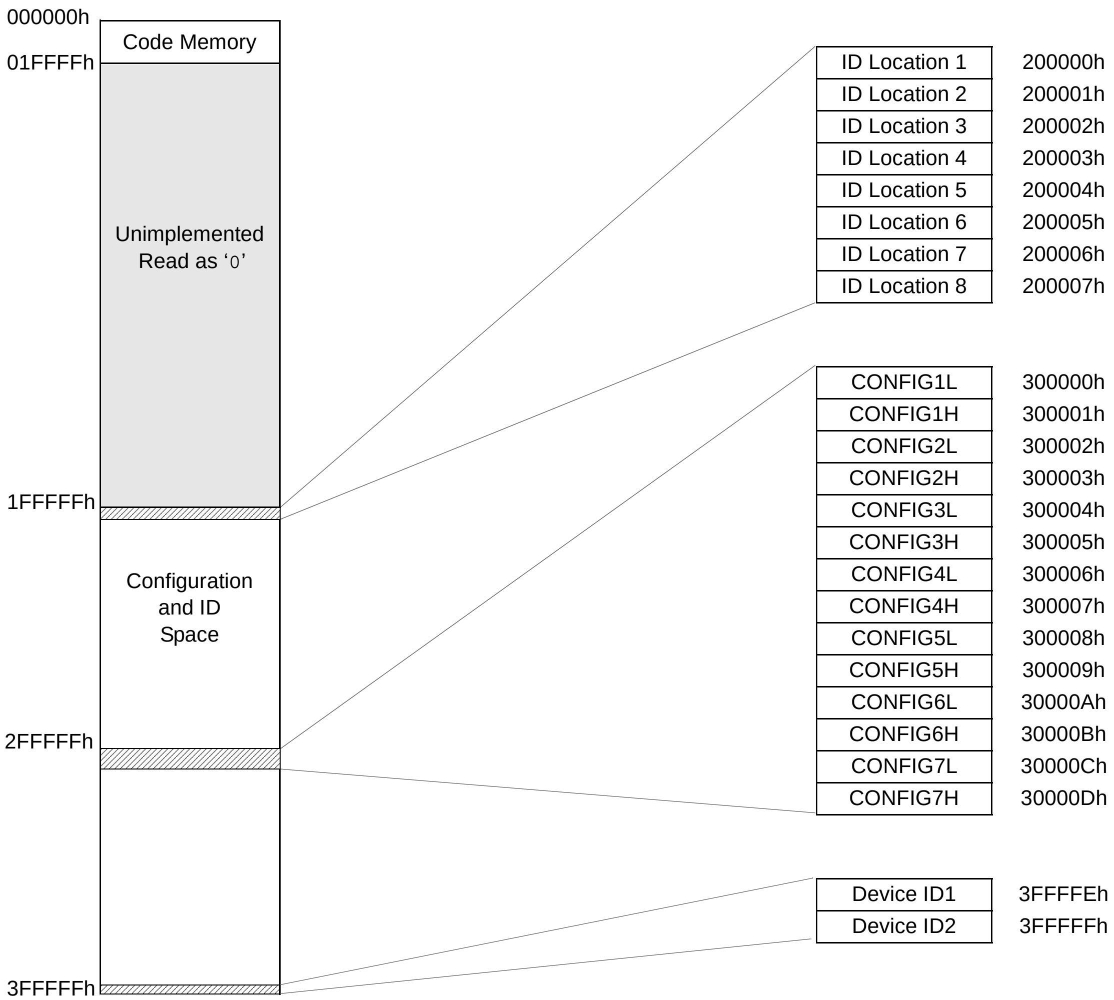
Memory in the address space 0000000h to 3FFFFFh is addressed via the Table Pointer register, which is comprised of three Pointer registers:

- 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 2-4: CONFIGURATION AND ID LOCATIONS FOR PIC18F6X2X/8X2X DEVICES



Note: Sizes of memory areas are not to scale.

2.4 High Level Overview of the Programming Process

Figure 2-6 shows the high level overview of the programming process. First, a bulk erase is performed. Next, the code memory, ID locations, and data EEPROM 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, 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 "Low Voltage Programming (LVP) Bit"**), 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 2-5: ENTERING HIGH VOLTAGE PROGRAM/VERIFY MODE

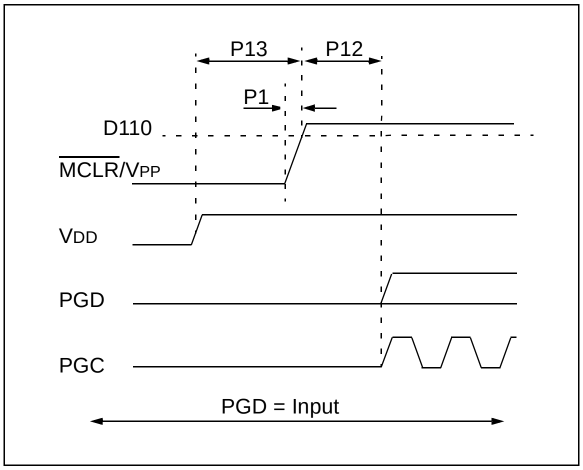


FIGURE 2-6: HIGH LEVEL PROGRAMMING FLOW

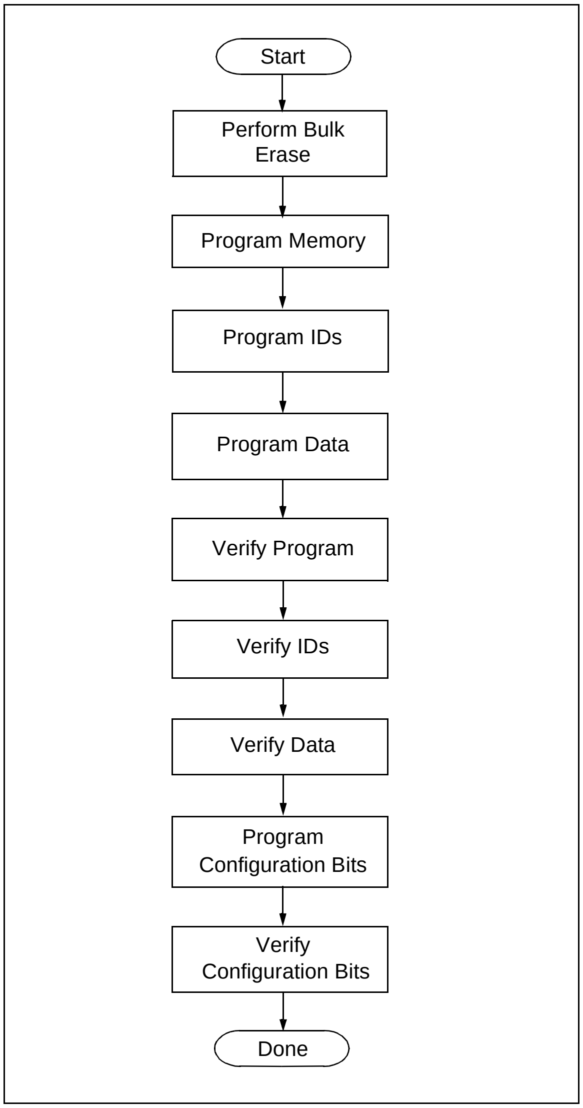
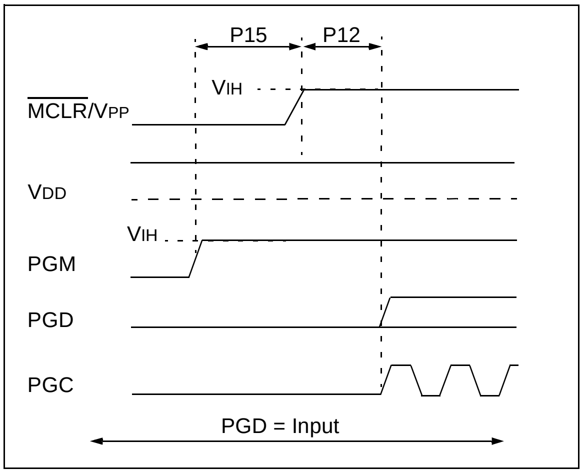


FIGURE 2-7: ENTERING LOW VOLTAGE PROGRAM/ VERIFY MODE



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 2-8 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.

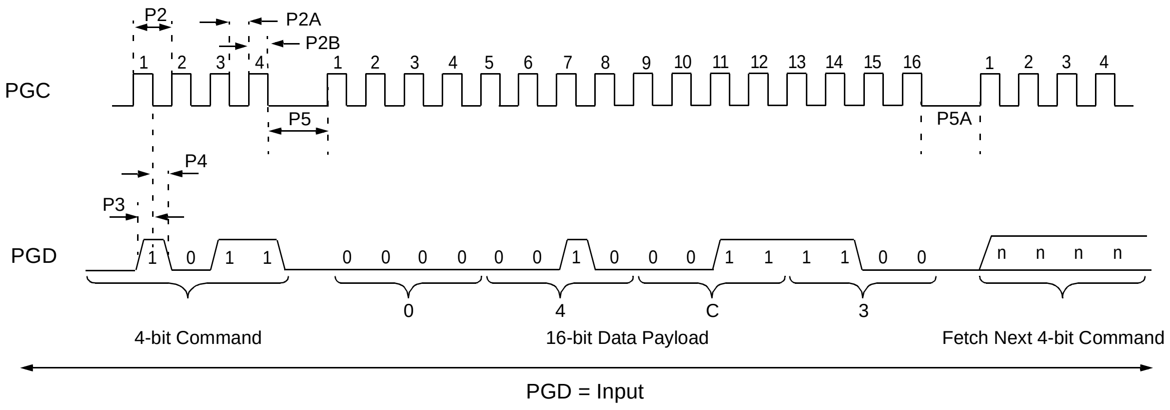
TABLE 2-3: COMMANDS FOR PROGRAMMING

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, post-decrement 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 2-8: TABLE WRITE, POST-INCREMENT TIMING (1101)



3.0 DEVICE PROGRAMMING

3.1 High Voltage ICSP Bulk Erase

Erasing code or data EEPROM is accomplished by writing an “erase option” to address 3C0004h. 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.

TABLE 3-1: BULK ERASE OPTIONS

Description	Data
Chip Erase	80h
Erase Data EEPROM	81h
Erase Boot Block	83h
Erase Block 1	88h
Erase Block 2	89h
Erase Block 3	8Ah
Erase Block 4	8Bh
Erase Block 5	8Ch
Erase Block 6	8Dh
Erase Block 7	8Eh
Erase Block 8	8Fh

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 Figure 3-2 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 04	MOVLW 04h
0000	6E F6	MOVWF TBLPTRL
1100	00 80	Write 80h 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

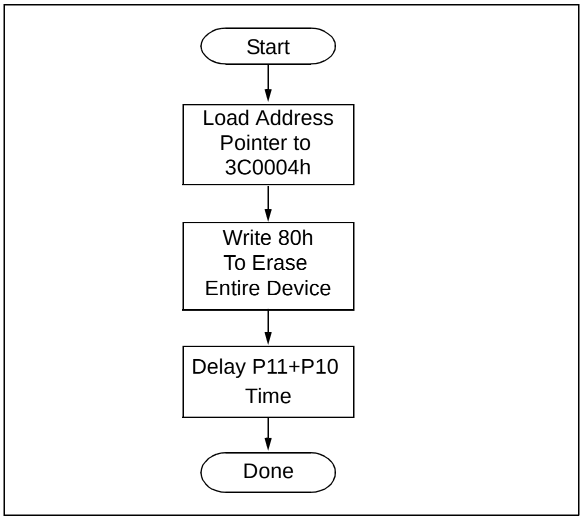
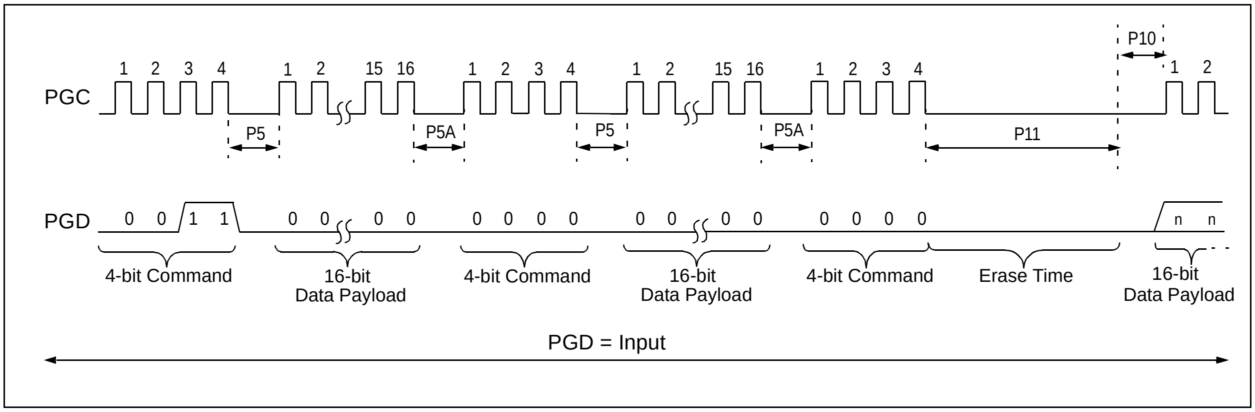


FIGURE 3-2: BULK ERASE TIMING



3.1.1 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 3.1.2 and 3.2.2.

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

3.1.2 ICSP MULTI-PANEL SINGLE ROW ERASE

Regardless of whether high or low voltage ICSP is used, it is possible to erase a single row (64 bytes of data) in all panels at once. For example, in the case of a 64-Kbyte device (8 panels), 512 bytes through 64 bytes in each panel can be erased simultaneously during each erase sequence. In this case, the offset of the erase within each panel is the same (see Figure 3-5). Multi-panel single row erase is enabled by appropriately configuring the Programming Control register located at 3C0006h.

The multi-panel single row erase 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.

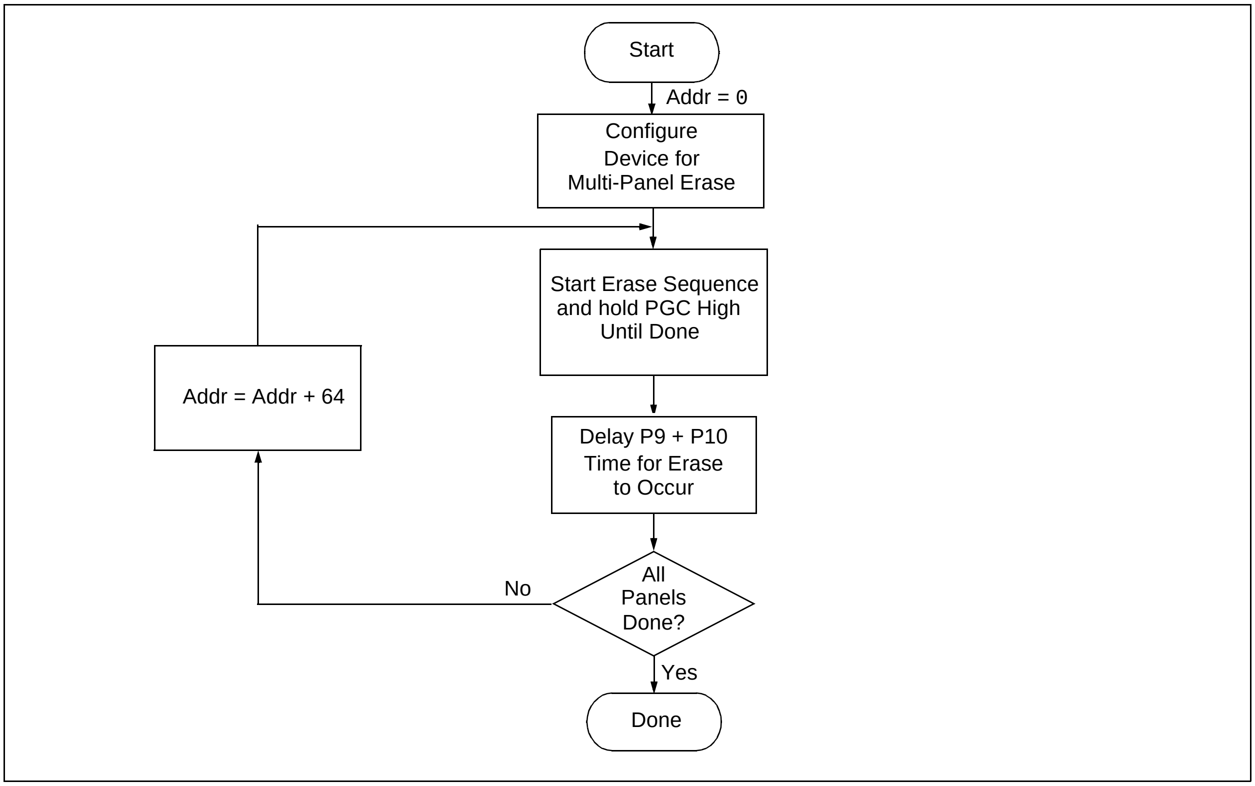
The code sequence to erase a PIC18F6X2X/8X2X device is shown in Table 3-3. The flowchart shown in Figure 3-3 depicts the logic necessary to completely erase a PIC18F6X2X/8X2X device. The timing diagram that details the "Start Programming" command, and parameters P9 and P10 is shown in Figure 3-6.

Note:	The TBLPTR register must contain the same offset value when initiating the programming sequence as it did when the write buffers were loaded.

TABLE 3-3: ERASE CODE MEMORY CODE SEQUENCE

4-bit Command	Data Payload	Core Instruction
Step 1: Direct access to config memory.		
0000	8E A6	BSF EECON1, EEPGD
0000	8C A6	BSF EECON1, CFGS
0000	84 A6	BSF EECON1, WREN
Step 2: Configure device for multi-panel writes.		
0000	0E 3C	MOVLW 3Ch
0000	6E F8	MOVWF TBLPTRU
0000	0E 00	MOVLW 00h
0000	6E F7	MOVWF TBLPTRH
0000	0E 06	MOVLW 06h
0000	6E F6	MOVWF TBLPTRL
1100	00 40	Write 40h to 3C0006h to enable multi-panel erase.
Step 3: Direct access to code memory.		
0000	8E A6	BSF EECON1, EEPGD
0000	9C A6	BCF EECON1, CFGS
0000	6A F8	CLRF TBLPTRU
0000	6A F7	CLRF TBLPTRH
0000	6A F6	CLRF TBLPTRL
Step 4: Enable erase and erase single row of all panels at an offset.		
0000	88 A6	BSF EECON1, FREE
1111	<DummyLSB> <DummyMSB>	Write 2 dummy bytes and start programming.
0000	00 00	NOP - hold PGC high for time P9.
Step 5: Repeat step 4, with Address Pointer incremented by 64 until all panels are erased.		

FIGURE 3-3: MULTI-PANEL SINGLE ROW ERASE CODE MEMORY FLOW



3.2 Code Memory Programming

Programming code memory is accomplished by first loading data into the appropriate write buffers and then initiating a programming sequence. Each panel in the code memory space (see Figure 2-3) has an 8-byte deep write buffer that must be loaded prior to initiating a write sequence. The actual memory write sequence takes the contents of these buffers and programs the associated EEPROM code memory.

Typically, all of the program buffers are written in parallel (Multi-Panel Write mode). In other words, in the case of a 64-Kbyte device (8 panels with an 8-byte buffer per panel), 64 bytes will be simultaneously programmed during each programming sequence. In this case, the offset of the write within each panel is the same (see Figure 3-4). Multi-Panel Write mode is enabled by appropriately configuring the Programming Control register located at 3C0006h.

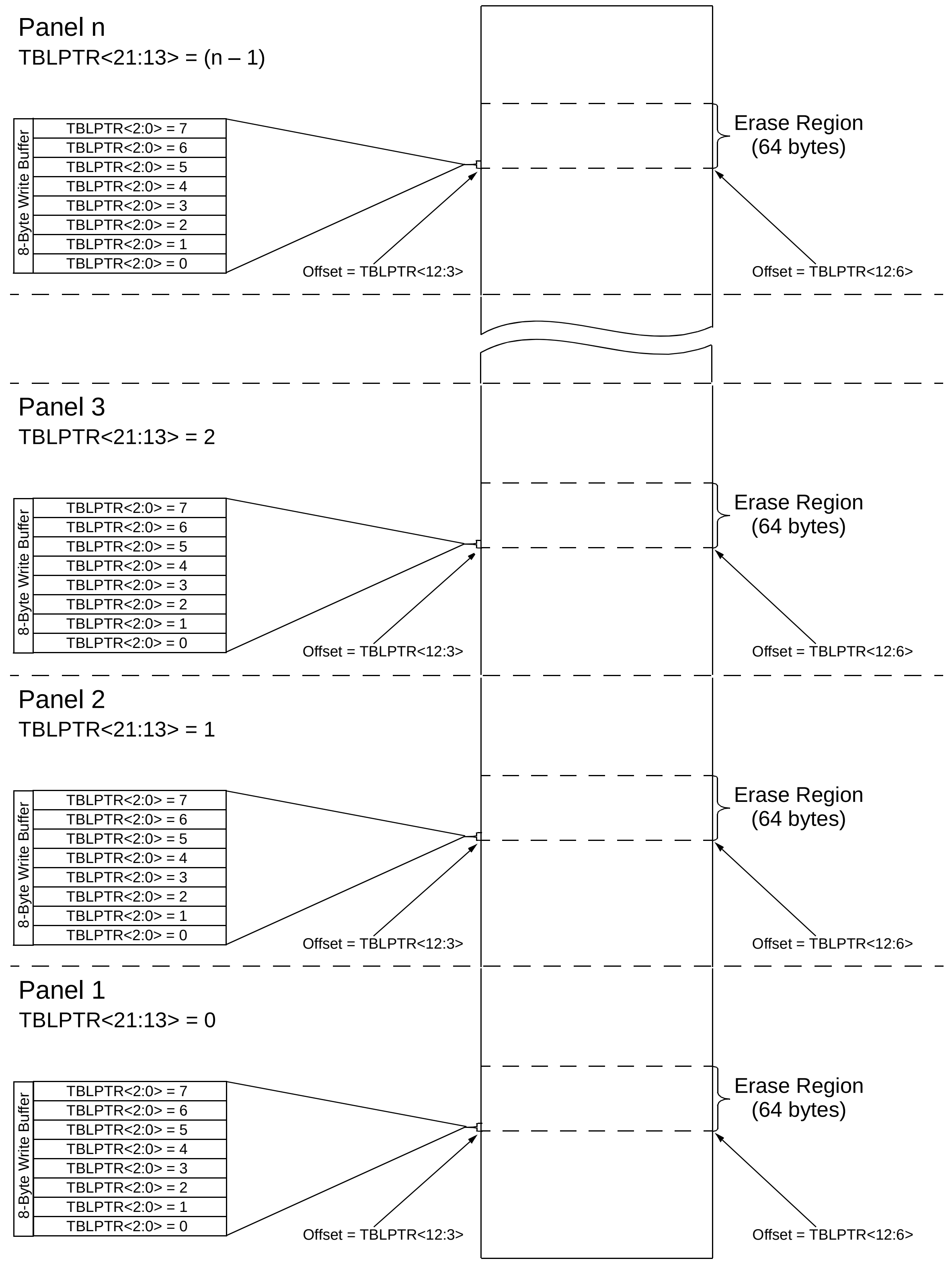
The programming duration is externally timed and is controlled by PGC. After a "Start Programming" command is issued (4-bit command, '`1111`'), a `NOP` is issued, where the 4th PGC is held high for the duration of the programming time, P9.

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 PIC18F6X2X/8X2X device is shown in Table 3-4. The flowchart shown in Figure 3-5 depicts the logic necessary to completely write a PIC18F6X2X/8X2X device. The timing diagram that details the "Start Programming" command, and parameters P9 and P10, is shown in Figure 3-6.

Note: The TBLPTR register must contain the same offset value when initiating the programming sequence as it did when the write buffers were loaded.

FIGURE 3-4: ERASE AND WRITE BOUNDARIES



Note: TBLPTR = TBLPTRU:TBLPTRH:TBLPTRL.

TABLE 3-4: WRITE CODE MEMORY CODE SEQUENCE

4-bit Command	Data Payload	Core Instruction
Step 1: Direct access to config memory.		
0000	8E A6	BSF EECON1, EEPGD
0000	8C A6	BSF EECON1, CFGS
0000	84 A6	BSF EECON1, WREN
Step 2: Configure device for multi-panel writes.		
0000	0E 3C	MOVLW 3Ch
0000	6E F8	MOVWF TBLPTRU
0000	0E 00	MOVLW 00h
0000	6E F7	MOVWF TBLPTRH
0000	0E 06	MOVLW 06h
0000	6E F6	MOVWF TBLPTRL
1100	00 40	Write 40h to 3C0006h to enable multi-panel writes.
Step 3: Direct access to code memory.		
0000	8E A6	BSF EECON1, EEPGD
0000	9C A6	BCF EECON1, CFGS
Step 4: Load write buffer for Panel 1.		
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
1101	<LSB><MSB>	Write 2 bytes and post-increment address by 2
1101	<LSB><MSB>	Write 2 bytes and post-increment address by 2
1101	<LSB><MSB>	Write 2 bytes and post-increment address by 2
1100	<LSB><MSB>	Write 2 bytes
Step 5: Repeat for Panel 2.		
Step 6: Repeat for all but the last panel (N – 1).		
Step 7: Load write buffer for last panel.		
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
1101	<LSB><MSB>	Write 2 bytes and post-increment address by 2
1101	<LSB><MSB>	Write 2 bytes and post-increment address by 2
1101	<LSB><MSB>	Write 2 bytes and post-increment address by 2
1111	<LSB><MSB>	Write 2 bytes and start programming
0000	00 00	NOP - hold PGC high for time P9
To continue writing data, repeat steps 2 through 5, where the Address Pointer is incremented by 8 in each panel at each iteration of the loop.		

FIGURE 3-5: PROGRAM CODE MEMORY FLOW

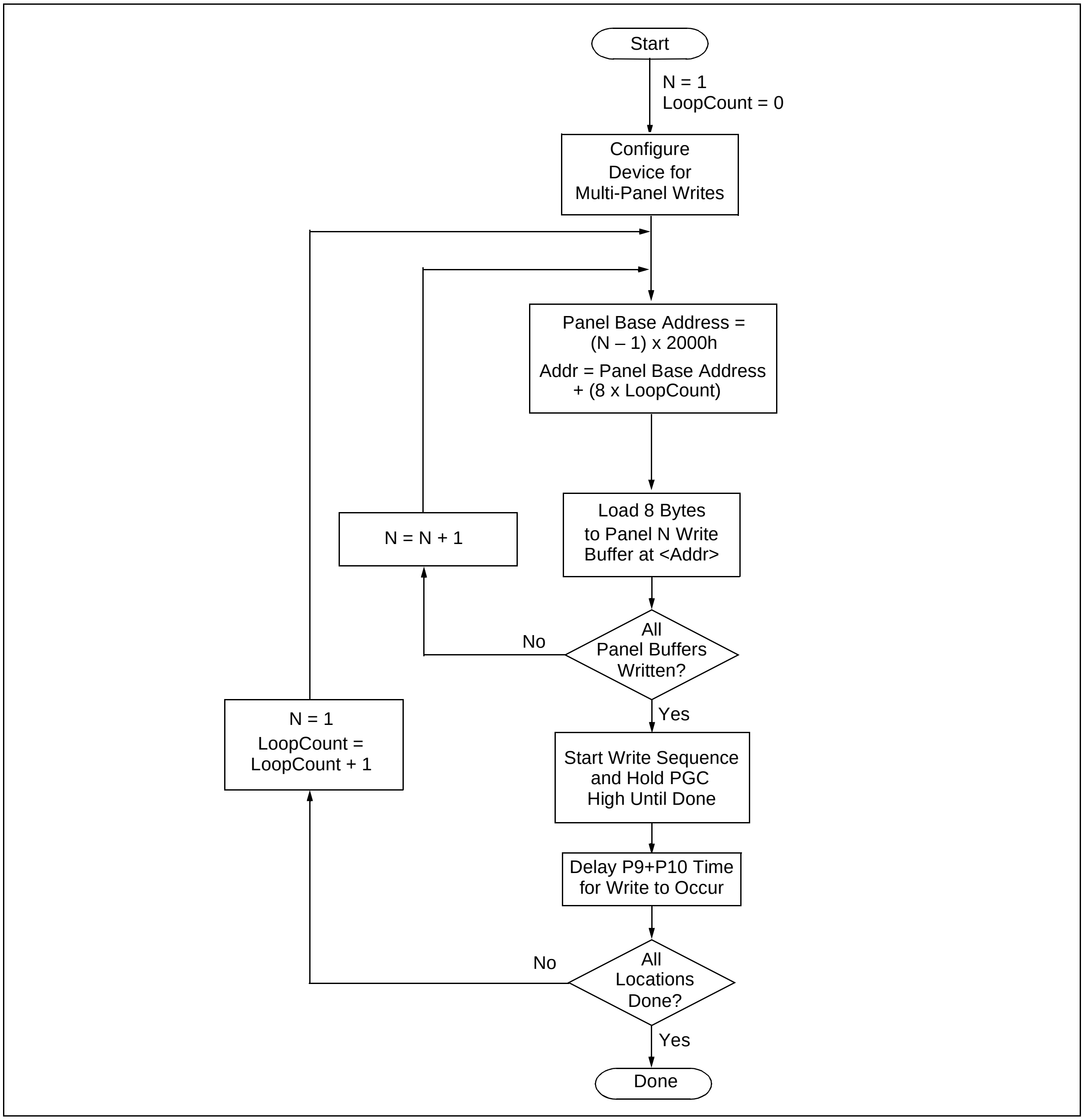
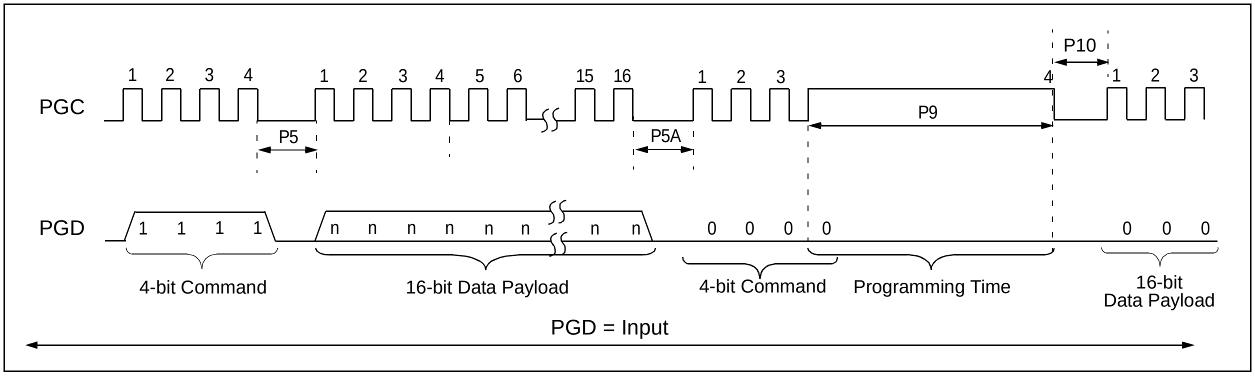


FIGURE 3-6: TABLE WRITE AND START PROGRAMMING INSTRUCTION TIMING (`1111`)



3.2.1 SINGLE PANEL PROGRAMMING

The programming example presented in **Section 3.2 "Code Memory Programming"** utilizes multi-panel programming. This technique greatly decreases the total amount of time necessary to completely program a device and is the recommended method of completely programming a device.

There may be situations, however, where it is advantageous to limit writes to a single panel. In such cases, the user only needs to disable the multi-panel write feature of the device by appropriately configuring the Programming Control register located at 3C0006h.

The single panel that will be written will automatically be enabled based on the value of the Table Pointer.

Note:	Even though multi-panel writes are disabled, the user must still fill the 8-byte write buffer for the given panel.

3.2.2 MODIFYING CODE MEMORY

All of the programming examples up to this point have assumed that the device has been bulk erased prior to programming (see **Section 3.1 "High Voltage ICSP Bulk Erase"**). It may be the case, however, that the user wishes to modify only a section of an already programmed device.

The minimum amount of data that can be written to the device is 8 bytes. This is accomplished by placing the device in Single Panel Write mode (see **Section 3.2.1 "Single Panel Programming"**), loading the 8-byte write buffer for the panel, and then initiating a write sequence. In this case, however, it is assumed that the address space to be written already has data in it (i.e., it is not blank).

The minimum amount of code memory that may be erased at a given time is 64 bytes. Again, the device must be placed in Single Panel Write mode. The EECON1 register must then be used to erase the 64-byte target space prior to writing the data.

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 sequence is initiated by the setting the WR bit (EECON1<1> = `1`). It is strongly recommended that the WREN bit be set only when absolutely necessary.

To help prevent inadvertent writes when using the EECON1 register, EECON2 is used to "enable" the WR bit. This register must be sequentially loaded with 55h and then AAh, immediately prior to asserting the WR bit in order for the write to occur.

The erase will begin on the falling edge of the 4th PGC after the WR bit is set. After the erase sequence terminates, PGC must still be held low for the time specified by parameter #P10 to allow high voltage discharge of the memory array.

TABLE 3-5: MODIFYING CODE MEMORY

4-bit Command	Data Payload	Core Instruction
Step 1: Direct access to config memory.		
0000	8E A6	BSF EECON1, EEPGD
0000	8C A6	BSF EECON1, CFGS
Step 2: Configure device for single panel writes.		
0000	0E 3C	MOVLW 3Ch
0000	6E F8	MOVWF TBLPTRU
0000	0E 00	MOVLW 00h
0000	6E F7	MOVWF TBLPTRH
0000	0E 06	MOVLW 06h
0000	6E F6	MOVWF TBLPTRL
1100	00 00	Write 00h to 3C0006h to enable single-panel writes.
Step 3: Direct access to code memory.		
0000	8E A6	BSF EECON1, EEPGD
0000	9C A6	BCF EECON1, CFGS
Step 4: Set the Table Pointer for the block to be erased.		
0000	0E <Addr[21:16]>	MOVLW <Addr[21:16]>
0000	6E F8	MOVWF TBLPTRU
0000	0E <Addr[8:15]>	MOVLW <Addr[8:15]>
0000	6E F7	MOVWF TBLPTRH
0000	0E <Addr[7:0]>	MOVLW <Addr[7:0]>
0000	6E F6	MOVWF TBLPTRL
Step 5: Enable memory writes and setup an erase.		
0000	84 A6	BSF EECON1, WREN
0000	88 A6	BSF EECON1, FREE
Step 6: Perform required sequence.		
0000	0E 55	MOVLW 55h
0000	6E A7	MOVWF EECON2
0000	0E AA	MOVLW 0AAh
0000	6E A7	MOVWF EECON2
Step 7: Initiate erase.		
0000	82 A6	BSF EECON1, WR
0000	00 00	NOP
Step 8: Wait for P11+P10.		
Step 9: Load write buffer for panel. The correct panel will be selected based on the Table Pointer.		
0000	0E <Addr[21:16]>	MOVLW <Addr[21:16]>
0000	6E F8	MOVWF TBLPTRU
0000	0E <Addr[8:15]>	MOVLW <Addr[8:15]>
0000	6E F7	MOVWF TBLPTRH
0000	0E <Addr[7:0]>	MOVLW <Addr[7:0]>
0000	6E F6	MOVWF TBLPTRL
1101	<LSB><MSB>	Write 2 bytes and post-increment address by 2
1101	<LSB><MSB>	Write 2 bytes and post-increment address by 2
1101	<LSB><MSB>	Write 2 bytes and post-increment address by 2
1111	<LSB><MSB>	Write 2 bytes and start programming
0000	00 00	NOP - hold PGC high for time P9
To continue writing data, repeat Step 9, where the Address Pointer is incremented by 8 at each iteration of the loop.		
Step 10: Disable writes.		
0000	94 A6	BCF EECON1, WREN

3.3 Data EEPROM Programming

Data EEPROM is accessed one byte at a time via an Address Pointer (register pair EEADR:EEADRH) and a data latch (EEDATA). Data EEPROM is written by loading EEADR:EEADRH with the desired memory location, EEDATA with the data to be written, and initiating a memory write by appropriately configuring the EECON1 and EECON2 registers. 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`). It is strongly recommended that the WREN bit be set only when absolutely necessary.

To help prevent inadvertent writes when using the EECON1 register, EECON2 is used to “enable” the WR bit. This register must be sequentially loaded with 55h and then AAh immediately prior to asserting the WR bit in order for the write to occur.

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 3-7: PROGRAM DATA FLOW

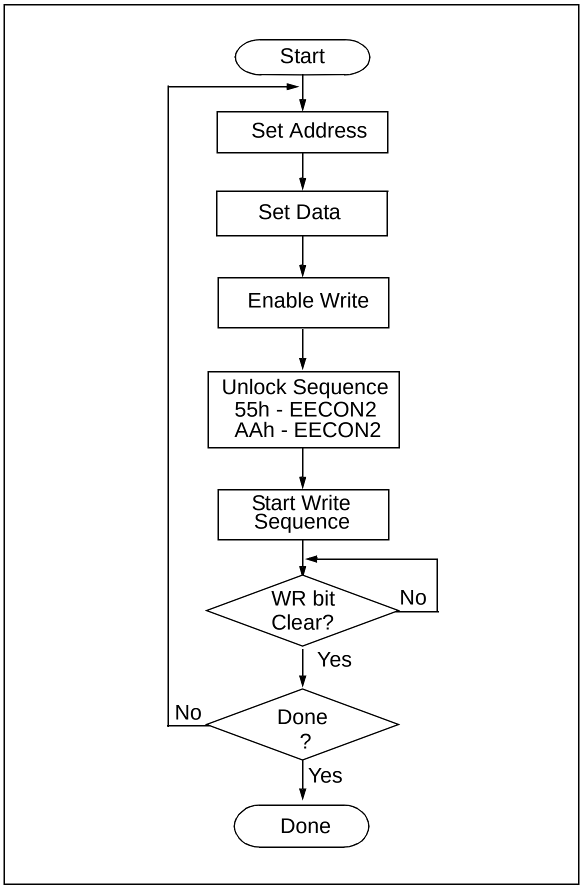


FIGURE 3-8: DATA EEPROM WRITE TIMING

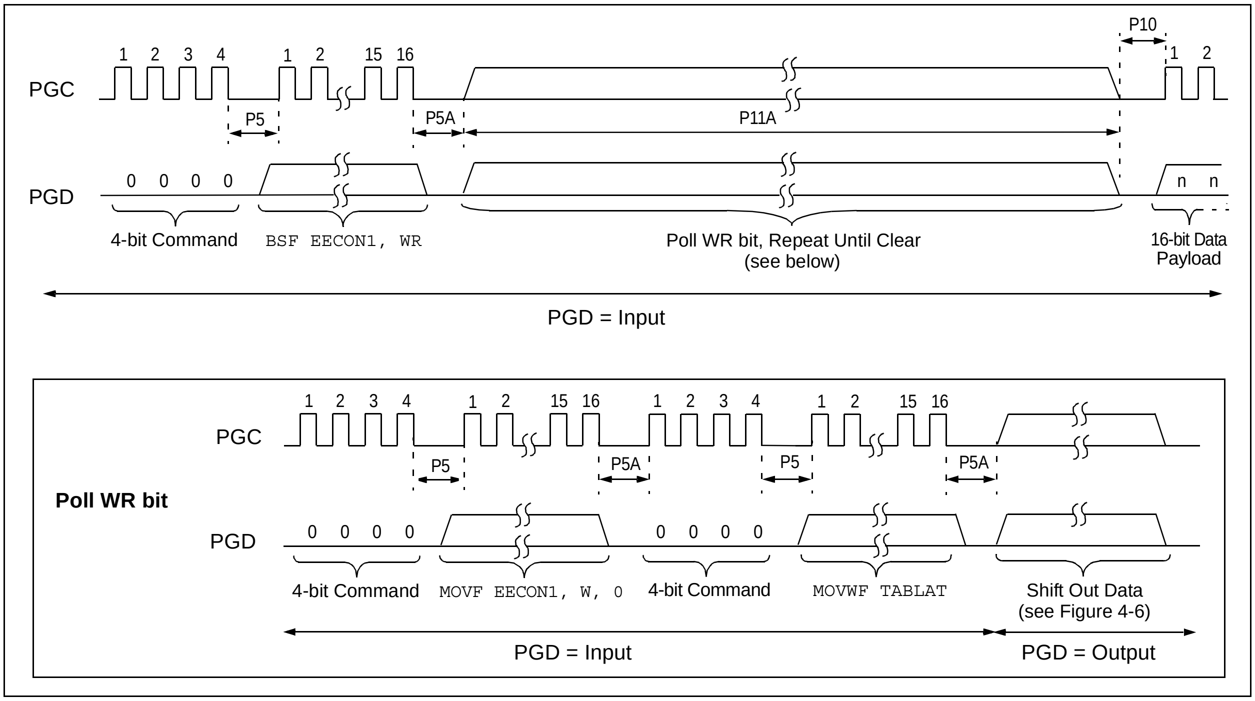


TABLE 3-6: PROGRAMMING DATA MEMORY

4-bit Command	Data Payload	Core Instruction
Step 1: Direct access to data EEPROM.		
`0000` `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: Load the data to be written.		
`0000` `0000`	`0E <Data>` `6E A8`	`MOVLW   <Data>` `MOVWF   EEDATA`
Step 4: Enable memory writes.		
`0000`	`84 A6`	`BSF     EECON1, WREN`
Step 5: Perform required sequence.		

`0000`
`0000`
`0000`
`0000`
	`0E 55`
	`6E A7`
	`0E AA`
	`6E A7`
		`MOVLW   0X55`
		`MOVWF   EECON2`
		`MOVLW   0XAA`
		`MOVWF   EECON2`

Step 6: Initiate write.		
`0000`	`82 A6`	`BSF     EECON1, WR`
Step 7: Poll WR bit, repeat until the bit is clear.		

`0000`
`0000`
`0010`
	`50 A6`
	`6E F5`
	`<LSB><MSB>`
		`MOVF    EECON1, W, 0`
		`MOVWF   TABLAT`
		`Shift out data`(1)

Step 8: Disable writes.		
`0000`	`94 A6`	`BCF     EECON1, WREN`
Repeat steps 2 through 8 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, except that multi-panel writes must be disabled. The single panel that will be written will automatically be enabled, based on the value of the Table Pointer. The ID registers are mapped in addresses 200000h through 200007h. These locations read out normally even after code protection.

Note: Even though multi-panel writes are disabled, the user must still fill the 8-byte data buffer for the panel.

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 3.2.2 "Modifying Code Memory"**, "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 config memory.		
`0000` `0000`	`8E A6` `8C A6`	`BSF   EECON1, EEPGD` `BSF   EECON1, CFGS`
Step 2: Configure device for single panel writes.		

`0000`
`0000`
`0000`
`0000`
`0000`
`0000`
`1100`
	`0E 3C`
	`6E F8`
	`0E 00`
	`6E F7`
	`0E 06`
	`6E F6`
	`00 00`
		`MOVLW 3Ch`
		`MOVWF TBLPTRU`
		`MOVLW 00h`
		`MOVWF TBLPTRH`
		`MOVLW 06h`
		`MOVWF TBLPTRL`
		`Write 00h to 3C0006h to enable single panel writes.`

Step 3: Direct access to code memory.		
`0000` `0000`	`8E A6` `9C A6`	`BSF   EECON1, EEPGD` `BCF   EECON1, CFGS`
Step 4: Load write buffer. Panel will be automatically determined by address.		

`0000`
`0000`
`0000`
`0000`
`0000`
`0000`
`1101`
`1101`
`1101`
`1111`
`0000`
	`0E 20`
	`6E F8`
	`0E 00`
	`6E F7`
	`0E 00`
	`6E F6`
	`<LSB><MSB>`
	`<LSB><MSB>`
	`<LSB><MSB>`
	`<LSB><MSB>`
	`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 Boot Block segment is programmed in exactly the same manner as the ID locations (see **Section 3.4 “ID Location Programming”**). Multi-panel writes must be disabled so that only addresses in the range 0000h to 07FFh will be written.

The code sequence detailed in Table 3-7 should be used, except that the address data used in “Step 2” will be in the range 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: Execute four `NOPS` between every configuration byte programming.

TABLE 3-8: SET ADDRESS POINTER TO CONFIGURATION LOCATION

4-bit Command	Data Payload	Core Instruction
Step 1: Direct access to config memory.		
`0000` `0000`	`8E A6` `8C A6`	`BSF EECON1, EEPGD` `BSF EECON1, CFGS`
Step 2: Position the program counter[(1)].		
`0000` `0000`	`EF 00` `F8 00`	`GOTO 100000h`
Step 3[(2)]: Set Table Pointer for config byte to be written. Write even/odd addresses.		

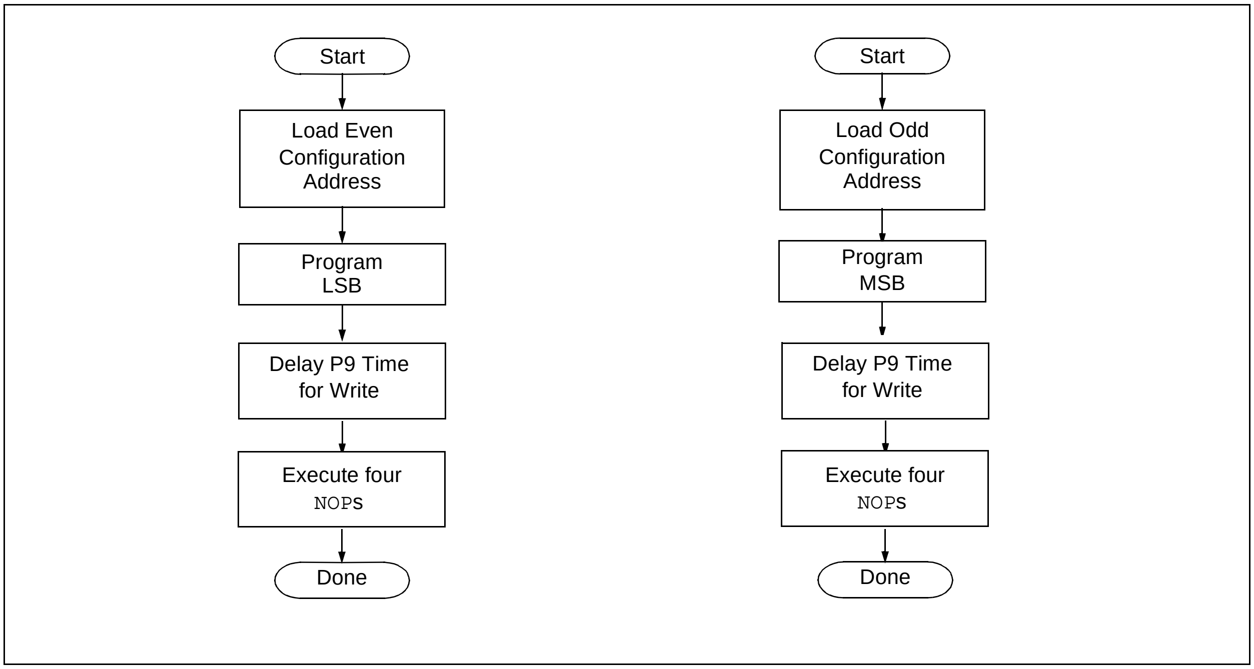
`0000`
`0000`
`0000`
`0000`
`0000`
`0000`
`1111`
`0000`
`0000`
`1111`
`0000`
	`0E 30`
	`6E F8`
	`0E 00`
	`6E F7`
	`0E 00`
	`6E F6`
	`<LSB><MSB ignored>`
	`00 00`
	`2A F6`
	`<LSB ignored><MSB>`
	`00 00`
		`MOVLW 30h`
		`MOVWF TBLPTRU`
		`MOVLW 00h`
		`MOVWF TBLPRTH`
		`MOVLW 00h`
		`MOVWF TBLPTRL`
		`Load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

Step 4: Execute four `NOP`s.		
0000 0000 0000 0000	`00 00` `00 00` `00 00` `00 00`	

Note 1: If the code protection bits are programmed while the program counter resides in the same block, then the interaction of code protection logic may prevent further table write. To avoid this situation, move the program counter outside the code protection area (e.g., `GOTO 100000h`).

2: 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 3-9: 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 loaded into the Table Latch and then serially output on PGD.

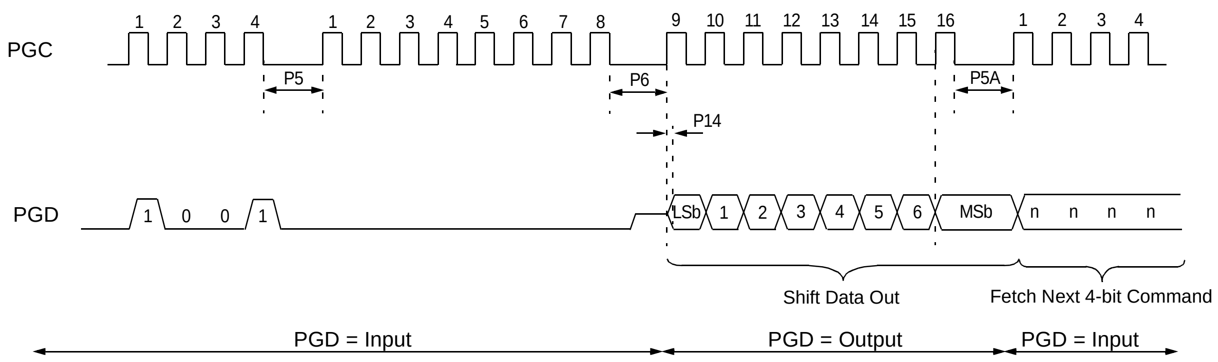
The 4-bit command is shifted in LSb first. The read is executed during the next 8 clocks, then shifted out on PGD during the last 8 clocks, LSb to MSb. A delay of P6 must be introduced after the falling edge of the 8th PGC of the operand to allow PGD to transition from an input to an output. During this 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 into Table Latch and then shift out on PGD, LSb to MSb.		
1001	00 00	TBLRD *+

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

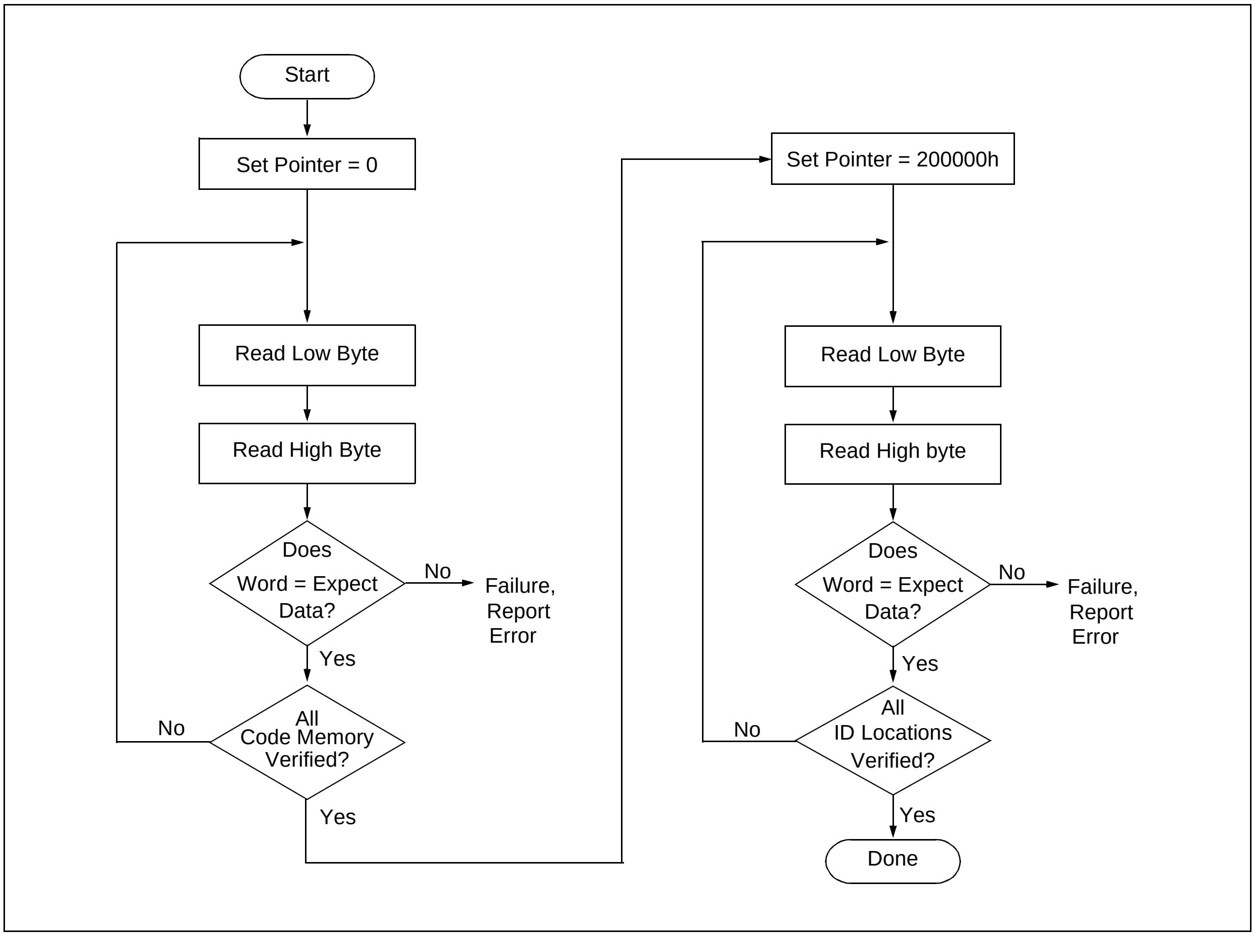


4.2 Verify Code Memory and ID locations

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 0000h, rather than point to unimplemented address 10000h.

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 EEADR:EEADRH) and a data latch (EEDATA). Data EEPROM is read by loading EEADR:EEADRH 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

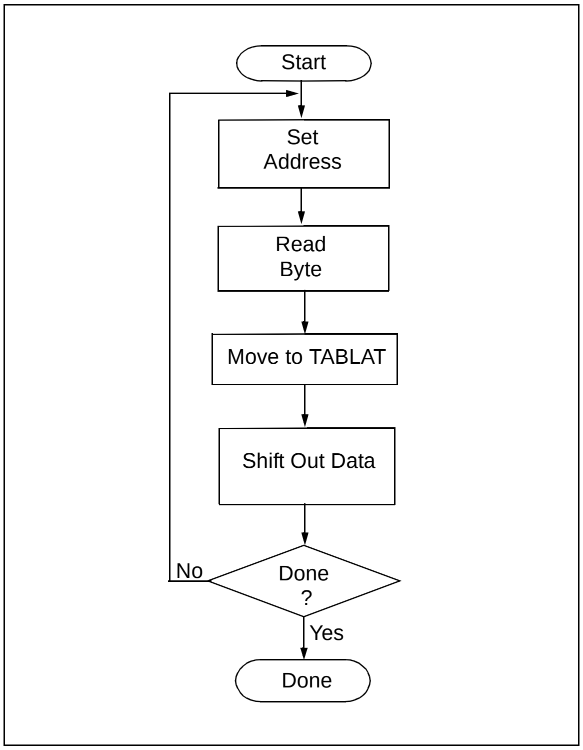


TABLE 4-2: READ DATA EEPROM MEMORY

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.		

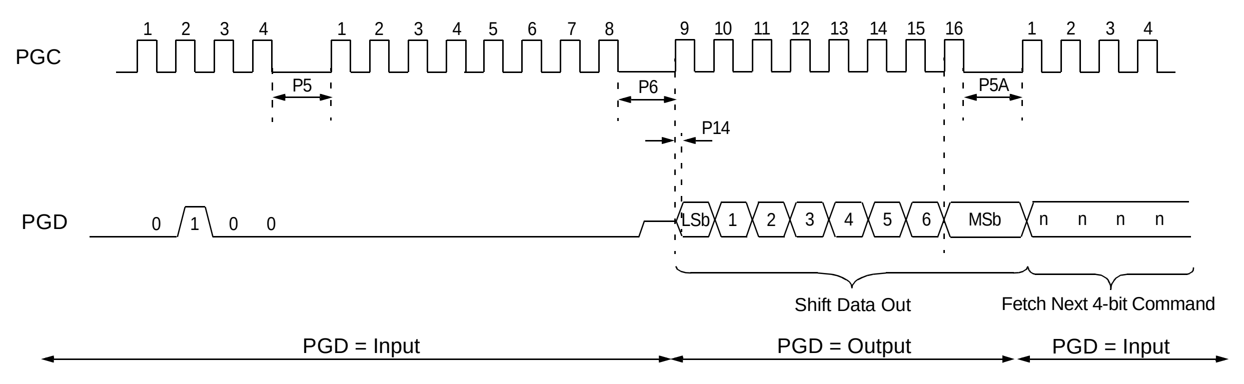
`0000`
`0000`
`0000`
`0000`
	`0E <Addr>`
	`6E A9`
	`OE <AddrH>`
	`6E AA`
		`MOVLW <Addr>`
		`MOVWF EEADR`
		`MOVLW <AddrH>`
		`MOVWF EEADRH`

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

`0000`
`0000`
`0010`
	`50 A8`
	`6E F5`
	`<LSB><MSB>`
		`MOVF EEDATA, W, 0`
		`MOVWF TABLAT`
		`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.

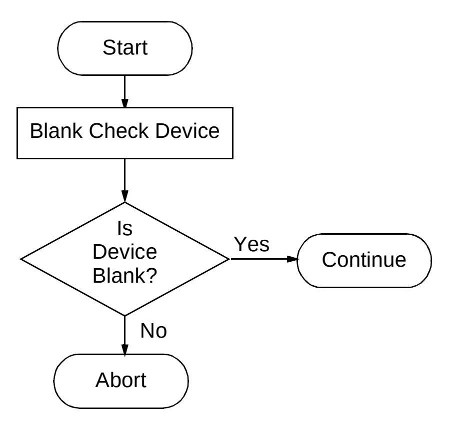
4.6 Blank Check

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 PIC18F6X2X/8X2X 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 PIC18F6X2X/8X2X 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 PIC18F6X2X/8X2X 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 Low Voltage Programming (LVP) Bit

The LVP bit in Configuration register, CONFIG4L, enables low voltage ICSP programming. The LVP bit defaults to a '`1`' from the factory.

If Low Voltage 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.

3: If the device Master Clear is disabled, verify that either of the following is done to ensure proper entry into ICSP mode:
a) disable Low Voltage Programming (CONFIG4L<2> = `0`); or
b) make certain that RB5/PGM is held low during entry into ICSP.

TABLE 5-1: DEVICE ID VALUE

Device	Device ID Value	
	DEVID2	DEVID1
18F6525	0Ah	`111x xxxx`
18F6621	0Ah	`101x xxxx`
18F8525	0Ah	`110x xxxx`
18F8621	0Ah	`100x xxxx`

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

TABLE 5-2: PIC18F6X2X/8X2X 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	—	—	$\overline{\text{OSCSEN}}$	—	FOSC3	FOSC2	FOSC1	FOSC0	0010 1111
300002h	CONFIG2L	—	—	—	—	BORV1	BORV0	BODEN	$\overline{\text{PWRTEN}}$	0000 1111
300003h	CONFIG2H	—	—	—	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDTEN	0001 1111
300004h(1)	CONFIG3L	WAIT	—	—	—	—	—	PM1	PM0	1000 0011
300005h	CONFIG3H	MCLRE(3)	—	—	—	—	—	ECCPMUX(1)	CCP2MX	1000 0011
300006h	CONFIG4L	$\overline{\text{DEBUG}}$	—	—	—	—	LVP	—	STVREN	1000 0101
300008h	CONFIG5L	—	—	—	—	CP3(2)	CP2	CP1	CP0	0000 1111
300009h	CONFIG5H	CPD	CPB	—	—	—	—	—	—	1100 0000
30000Ah	CONFIG6L	—	—	—	—	WRT3(2)	WRT2	WRT1	WRT0	0000 1111
30000Bh	CONFIG6H	WRTD	WRTB	WRTC	—	—	—	—	—	1110 0000
30000Ch	CONFIG7L	—	—	—	—	EBTR3(2)	EBTR2	EBTR1	EBTR0	0000 1111
30000Dh	CONFIG7H	—	EBTRB	—	—	—	—	—	—	0100 0000
3FFFFEh	DEVID1	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	Table 5-1
3FFFFFh	DEVID2	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	Table 5-1

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

Note 1: Unimplemented in PIC18F6X2X devices; maintain this bit set.

2: Unimplemented in PIC18FX525 devices; maintain this bit set.

3: If $\overline{\text{MCLR}}$ is disabled, either disable low voltage ICSP or hold RB5/PGM low to ensure proper entry into ICSP mode.

TABLE 5-3: PIC18F6X2X/8X2X CONFIGURATION BIT DESCRIPTIONS

Bit Name	Configuration Words	Description
$\overline{\text{OSCEN}}$	CONFIG1H	Low Power System Clock Option (Timer1) Enable bit 1 = Disabled 0 = Timer1 oscillator system clock option enabled
FOSC3:FOSC0	CONFIG1H	Oscillator Selection bits 1111 = RC oscillator w/ OSC2 configured as RA6 1110 = HS oscillator w/ s/w enabled PLL 1101 = EC w/ OSC2 configured as RA6 and s/w enabled PLL 1100 = EC w/ OSC2 configured as RA6 and PLL enabled 1011 = Reserved; do not use 1010 = Reserved; do not use 1001 = Reserved; do not use 1000 = Reserved; do not use 0111 = RC oscillator w/ OSC2 configured as RA6 0110 = HS oscillator w/ PLL enabled 0101 = EC w/ OSC2 configured as RA6 0100 = EC w/ OSC2 configured as 'divide-by-4 clock output' 0011 = RC oscillator w/ OSC2 configured as 'divide-by-4 clock output' 0010 = HS oscillator 0001 = XT oscillator 0000 = LP oscillator
BORV1:BORV0	CONFIG2L	Brown-out Reset Voltage bits 11 = VBOR set to 2.0V 10 = VBOR set to 2.7V 01 = VBOR set to 4.2V 00 = VBOR set to 4.5V
BOREN	CONFIG2L	Brown-out Reset Enable bit 1 = Brown-out Reset enabled 0 = Brown-out Reset disabled
$\overline{\text{PWRTEN}}$	CONFIG2L	Power-up Timer Enable bit 1 = PWRT disabled 0 = PWRT enabled

Note 1: Unimplemented in PIC18F6X2X devices; maintain this bit set.

2: Unimplemented in PIC18FX525 devices; maintain this bit set.

3: If $\overline{\text{MCLR}}$ is disabled, either disable low voltage ICSP or hold RB5/PGM low to ensure proper entry into ICSP mode.

TABLE 5-3: PIC18F6X2X/8X2X CONFIGURATION BIT DESCRIPTIONS (CONTINUED)

Bit Name	Configuration Words	Description
WDTPS3:WDTPS0	CONFIG2H	Watchdog Timer Postscaler Select bits 1111 = 1:32768 1110 = 1:16384 1101 = 1:8192 1100 = 1:4,096 1011 = 1:2048 1010 = 1:1024 1001 = 1:512 1000 = 1:256 0111 = 1:128 0110 = 1:64 0101 = 1:32 0100 = 1:16 0011 = 1:8 0010 = 1:4 0001 = 1:2 0000 = 1:1
WDTEN	CONFIG2H	Watchdog Timer Enable bit 1 = WDT enabled 0 = WDT disabled (control is placed on SWDTEN bit)
WAIT(1)	CONFIG3L	External Bus Data Wait Enable bit 1 = Wait selections unavailable 0 = Wait selections determined by WAIT1:WAIT0 bits of MEMCON register
PM1:PM0(1)	CONFIG3L	Processor Mode Select bits 11 = Microcontroller mode 10 = Microprocessor mode 01 = Microprocessor with Boot Block mode 00 = Extended Microcontroller mode
MCLRE(3)	CONFIG3H	$\overline{\text{MCLR}}$ Enable bit 1 = $\overline{\text{MCLR}}$ pin enabled, RG5 disabled 0 = RG5 input pin enabled, $\overline{\text{MCLR}}$ disabled
ECCPMUX(1)	CONFIG3H	ECCP Mux bit 1 = Enhanced CCP1 (P1B/P1C) and enhanced CCP3 (P3B/P3C) PWM outputs are multiplexed onto RE6 through RE3 0 = Enhanced CCP1 (P1B/P1C) and enhanced CCP3 (P3B/P3C) PWM outputs are multiplexed onto RH7 through RH4
CCP2MX	CONFIG3H	CCP2 MUX bit 1 = CCP2 input/output is multiplexed with RC1 0 = CCP2 input/output is multiplexed with RE7 in Microcontroller mode; CCP2 input/output is multiplexed with RB3 in Extended Microcontroller mode, Microprocessor or Microprocessor w/ Boot Block mode

Note 1: Unimplemented in PIC18F6X2X devices; maintain this bit set.

2: Unimplemented in PIC18FX525 devices; maintain this bit set.

3: If $\overline{\text{MCLR}}$ is disabled, either disable low voltage ICSP or hold RB5/PGM low to ensure proper entry into ICSP mode.

TABLE 5-3: PIC18F6X2X/8X2X CONFIGURATION BIT DESCRIPTIONS (CONTINUED)

Bit Name	Configuration Words	Description
$\overline{\text{DEBUG}}$	CONFIG4L	Background Debugger Enable bit `1` = Background debugger disabled `0` = Background debugger enabled
LVP	CONFIG4L	Low Voltage Programming Enable bit `1` = Low voltage programming enabled `0` = Low voltage programming disabled
STVREN	CONFIG4L	Stack Overflow/Underflow Reset Enable bit `1` = Stack overflow/underflow will cause RESET `0` = Stack overflow/underflow will not cause RESET
CP0	CONFIG5L	Code Protection bits (code memory area 0800h - 3FFFh) `1` = Code memory not code protected `0` = Code memory code protected
CP1	CONFIG5L	Code Protection bits (code memory area 4000h - 7FFFh) `1` = Code memory not code protected `0` = Code memory code protected
CP2	CONFIG5L	Code Protection bits (code memory area 8000h - BFFFh) `1` = Code memory not code protected `0` = Code memory code protected
CP3[(2)]	CONFIG5L	Code Protection bits (code memory area C000h - FFFFh) `1` = Code memory not code protected `0` = Code memory code protected
CPD	CONFIG5H	Code Protection bits (data EEPROM) `1` = Data EEPROM not code protected `0` = Data EEPROM code protected
CPB	CONFIG5H	Code Protection bits (boot block, memory area 0000h - 07FFh) `1` = Boot block not code protected `0` = Boot block code protected
WRT0	CONFIG6L	Table Write Protection bit (code memory area 0800h - 3FFFh) `1` = Code memory not write protected `0` = Code memory write protected
WRT1	CONFIG6L	Table Write Protection bit (code memory area 4000h - 7FFFh) `1` = Code memory not write protected `0` = Code memory write protected
WRT2	CONFIG6L	Table Write Protection bit (code memory area 8000h - BFFFh) `1` = Code memory not write protected `0` = Code memory write protected
WRT3[(2)]	CONFIG6L	Table Write Protection bit (code memory area C000h - FFFFh) `1` = Code memory not write protected `0` = Code memory write protected

Note 1: Unimplemented in PIC18F6X2X devices; maintain this bit set.

2: Unimplemented in PIC18FX525 devices; maintain this bit set.

3: If $\overline{\text{MCLR}}$ is disabled, either disable low voltage ICSP or hold RB5/PGM low to ensure proper entry into ICSP mode.

TABLE 5-3: PIC18F6X2X/8X2X CONFIGURATION BIT DESCRIPTIONS (CONTINUED)

Bit Name	Configuration Words	Description
WRTD	CONFIG6H	Table Write Protection bit (data EEPROM) `1` = Data EEPROM not write protected `0` = Data EEPROM write protected
WRTB	CONFIG6H	Table Write Protection bit (boot block, memory area 0000h - 07FFh) `1` = Boot block not write protected `0` = Boot block write protected
WRTC	CONFIG6H	Table Write Protection bit (Configuration registers) `1` = Configuration registers not write protected `0` = Configuration registers write protected
EBTR0	CONFIG7L	Table Read Protection bit (code memory area 0800h - 3FFFh) `1` = Code memory not protected from table reads executed in other blocks `0` = Code memory protected from table reads executed in other blocks
EBTR1	CONFIG7L	Table Read Protection bit (code memory area 4000h - 7FFFh) `1` = Code memory not protected from table reads executed in other blocks `0` = Code memory protected from table reads executed in other blocks
EBTR2	CONFIG7L	Table Read Protection bit (code memory area 8000h - BFFFh) `1` = Code memory not protected from table reads executed in other blocks `0` = Code memory protected from table reads executed in other blocks
EBTR3(2)	CONFIG7L	Table Read Protection bit (code memory area C000h - FFFFh) `1` = Code memory not protected from table reads executed in other blocks `0` = Code memory protected from table reads executed in other blocks
EBTRB	CONFIG7H	Table Read Protection bit (boot block, memory area 0000h - 07FFh) `1` = Boot block not protected from table reads executed in other blocks `0` = Boot block protected from table reads executed in other blocks
DEV10:DEV3	DEVID2	Device ID bits These bits are used with the DEV2:DEV0 bits in the DEVID1 register to identify part number.
DEV2:DEV0	DEVID1	Device ID bits These bits are used with the DEV10:DEV3 bits in the DEVID2 register to identify part number.
REV4:REV0	DEVID1	These bits are used to indicate the revision of the device.

Note 1: Unimplemented in PIC18F6X2X devices; maintain this bit set.

2: Unimplemented in PIC18FX525 devices; maintain this bit set.

3: If $\overline{\text{MCLR}}$ is disabled, either disable low voltage ICSP or hold RB5/PGM low to ensure proper entry into ICSP mode.

5.4 Embedding Configuration Word Information in the HEX File

To allow portability of code, a PIC18F6X2X/8X2X 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 PIC18F6X2X/8X2X 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 5-4 (pages 33 through 34) 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 5-4: CHECKSUM COMPUTATION

Device	Code Protect	Checksum	Blank Value	0xAA at 0 and Max. Address
PIC18F6525	None	SUM(0000:07FF)+SUM(0800:3FFF)+SUM(4000:7FFF)+ SUM(8000:BFFF)+(CONFIG1L & 0000)+(CONFIG1H & 002F)+ (CONFIG2L & 000F)+(CONFIG2H & 001F)+(CONFIG3L & 0000)+ (CONFIG3H & 0081)+(CONFIG4L & 0085)+(CONFIG4H & 0000)+ (CONFIG5L & 0007)+(CONFIG5H & 00C0)+(CONFIG6L & 0007)+ (CONFIG6H & 00E0)+(CONFIG7L & 0007)+(CONFIG7H & 0040)	4358	42AE
	Boot Block	SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+ (CONFIG1L & 0000)+(CONFIG1H & 002F)+(CONFIG2L & 000F)+ (CONFIG2H & 001F)+(CONFIG3L & 0000)+(CONFIG3H & 0081)+ (CONFIG4L & 0085)+(CONFIG4H & 0000)+(CONFIG5L & 0007)+ (CONFIG5H & 00C0)+(CONFIG6L & 0007)+(CONFIG6H & 00E0)+ (CONFIG7L & 0007)+(CONFIG7H & 0040)+SUM(IDs)	4B2C	4A8C
	Boot/ Block1/ Block2	SUM(8000:BFFF)+(CONFIG1L & 0000)+(CONFIG1H & 002F)+ (CONFIG2L & 000F)+(CONFIG2H & 001F)+(CONFIG3L & 0000)+ (CONFIG3H & 0081)+(CONFIG4L & 0085)+(CONFIG4H & 0000)+ (CONFIG5L & 0007)+(CONFIG5H & 00C0)+(CONFIG6L & 0007)+ (CONFIG6H & 00E0)+(CONFIG7L & 0007)+(CONFIG7H & 0040)+ SUM(IDs)	C329	C289
	All	(CONFIG1L & 0000)+(CONFIG1H & 002F)+(CONFIG2L & 000F)+ (CONFIG2H & 001F)+(CONFIG3L & 0000)+(CONFIG3H & 0083)+ (CONFIG4L & 0085)+(CONFIG4H & 0000)+(CONFIG5L & 00FF)+ (CONFIG5H & 00C0)+(CONFIG6L & 00FF)+(CONFIG6H & 00E0)+ (CONFIG7L & 00FF)+(CONFIG7H & 0040)+SUM(IDs)	0325	0285
PIC18F6621	None	SUM(0000:07FF)+SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+ SUM(C000:FFFF)+(CONFIG1L & 0000)+(CONFIG1H & 002F)+ (CONFIG2L & 000F)+(CONFIG2H & 001F)+(CONFIG3L & 0000)+ (CONFIG3H & 0081)+(CONFIG4L & 0085)+(CONFIG4H & 0000)+ (CONFIG5L & 000F)+(CONFIG5H & 00C0)+(CONFIG6L & 000F)+ (CONFIG6H & 00E0)+(CONFIG7L & 000F)+(CONFIG7H & 0040)	0370	02C6
	Boot Block	SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+ (CONFIG1L & 0000)+(CONFIG1H & 002F)+(CONFIG2L & 000F)+ (CONFIG2H & 001F)+(CONFIG3L & 0000)+(CONFIG3H & 0081)+ (CONFIG4L & 0085)+(CONFIG4H & 0000)+(CONFIG5L & 000F)+ (CONFIG5H & 00C0)+(CONFIG6L & 000F)+(CONFIG6H & 00E0)+ (CONFIG7L & 000F)+(CONFIG7H & 0040)+SUM(IDs)	0B3A	0AEF
	Boot/ Block1/ Block2	SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG1L & 0000)+ (CONFIG1H & 002F)+(CONFIG2L & 000F)+(CONFIG2H & 001F)+ (CONFIG3L & 0000)+(CONFIG3H & 0081)+(CONFIG4L & 0085)+ (CONFIG4H & 0000)+(CONFIG5L & 000F)+(CONFIG5H & 00C0)+ (CONFIG6L & 000F)+(CONFIG6H & 00E0)+(CONFIG7L & 000F)+ (CONFIG7H & 0040)+SUM(IDs)	8337	82EC
	All	(CONFIG1L & 0000)+(CONFIG1H & 002F)+(CONFIG2L & 000F)+ (CONFIG2H & 001F)+(CONFIG3L & 0000)+(CONFIG3H & 0081)+ (CONFIG4L & 0085)+(CONFIG4H & 0000)+(CONFIG5L & 000F)+ (CONFIG5H & 00C0)+(CONFIG6L & 000F)+(CONFIG6H & 00E0)+ (CONFIG7L & 000F)+(CONFIG7H & 0040)+SUM(IDs)	032B	0335

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 5-4: CHECKSUM COMPUTATION (CONTINUED)

Device	Code Protect	Checksum	Blank Value	0xAA at 0 and Max. Address
PIC18F8525	None	SUM(0000:07FF)+SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+(CONFIG1L & 0000)+(CONFIG1H & 002F)+(CONFIG2L & 000F)+(CONFIG2H & 001F)+(CONFIG3L & 0083)+(CONFIG3H & 0083)+(CONFIG4L & 0085)+(CONFIG4H & 0000)+(CONFIG5L & 0007)+(CONFIG5H & 00C0)+(CONFIG6L & 0007)+(CONFIG6H & 00E0)+(CONFIG7L & 0007)+(CONFIG7H & 0040)	43DD	4333
	Boot Block	SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+(CONFIG1L & 0000)+(CONFIG1H & 002F)+(CONFIG2L & 000F)+(CONFIG2H & 001F)+(CONFIG3L & 0083)+(CONFIG3H & 0083)+(CONFIG4L & 0085)+(CONFIG4H & 0000)+(CONFIG5L & 0007)+(CONFIG5H & 00C0)+(CONFIG6L & 0007)+(CONFIG6H & 00E0)+(CONFIG7L & 0007)+(CONFIG7H & 0040)+SUM(IDs)	4BBE	4B00
	Boot/ Block1/ Block2	SUM(8000:BFFF)+(CONFIG1L & 0000)+(CONFIG1H & 002F)+(CONFIG2L & 000F)+(CONFIG2H & 001F)+(CONFIG3L & 0083)+(CONFIG3H & 0083)+(CONFIG4L & 0085)+(CONFIG4H & 0000)+(CONFIG5L & 0007)+(CONFIG5H & 00C0)+(CONFIG6L & 0007)+(CONFIG6H & 00E0)+(CONFIG7L & 0007)+(CONFIG7H & 0040)+SUM(IDs)	C3BB	C2FD
	All	(CONFIG1L & 0000)+(CONFIG1H & 002F)+(CONFIG2L & 000F)+(CONFIG2H & 001F)+(CONFIG3L & 0083)+(CONFIG3H & 0083)+(CONFIG4L & 0085)+(CONFIG4H & 0000)+(CONFIG5L & 0007)+(CONFIG5H & 00C0)+(CONFIG6L & 0007)+(CONFIG6H & 00E0)+(CONFIG7L & 0007)+(CONFIG7H & 0040)+SUM(IDs)	03B7	02F9
PIC18F8621	None	SUM(0000:07FF)+SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG1L & 0000)+(CONFIG1H & 002F)+(CONFIG2L & 000F)+(CONFIG2H & 001F)+(CONFIG3L & 0083)+(CONFIG3H & 0083)+(CONFIG4L & 0085)+(CONFIG4H & 0000)+(CONFIG5L & 000F)+(CONFIG5H & 00C0)+(CONFIG6L & 000F)+(CONFIG6H & 00E0)+(CONFIG7L & 000F)+(CONFIG7H & 0040)	03F5	034B
	Boot Block	SUM(0800:3FFF)+SUM(4000:7FFF)+SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG1L & 0000)+(CONFIG1H & 002F)+(CONFIG2L & 000F)+(CONFIG2H & 001F)+(CONFIG3L & 0083)+(CONFIG3H & 0083)+(CONFIG4L & 0085)+(CONFIG4H & 0000)+(CONFIG5L & 000F)+(CONFIG5H & 00C0)+(CONFIG6L & 000F)+(CONFIG6H & 00E0)+(CONFIG7L & 000F)+(CONFIG7H & 0040)+SUM(IDs)	0BCC	0B1D
	Boot/ Block1/ Block2	SUM(8000:BFFF)+SUM(C000:FFFF)+(CONFIG1L & 0000)+(CONFIG1H & 002F)+(CONFIG2L & 000F)+(CONFIG2H & 001F)+(CONFIG3L & 0083)+(CONFIG3H & 0083)+(CONFIG4L & 0085)+(CONFIG4H & 0000)+(CONFIG5L & 000F)+(CONFIG5H & 00C0)+(CONFIG6L & 000F)+(CONFIG6H & 00E0)+(CONFIG7L & 000F)+(CONFIG7H & 0040)+SUM(IDs)	83C9	831A
	All	(CONFIG1L & 0000)+(CONFIG1H & 002F)+(CONFIG2L & 000F)+(CONFIG2H & 001F)+(CONFIG3L & 0083)+(CONFIG3H & 0083)+(CONFIG4L & 0085)+(CONFIG4H & 0000)+(CONFIG5L & 000F)+(CONFIG5H & 00C0)+(CONFIG6L & 000F)+(CONFIG6H & 00E0)+(CONFIG7L & 000F)+(CONFIG7H & 0040)+SUM(IDs)	03BD	030E

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	VIHH	High Voltage Programming Voltage on $\overline{\text{MCLR}}$/VPP	9.00	13.25	V	
D110A	VIHL	Low Voltage Programming Voltage on $\overline{\text{MCLR}}$/VPP	2.00	5.50	V	
D111	VDD	Supply Voltage During Programming	2.00	5.50	V	Normal programming
			4.50	5.50	V	Bulk erase operations
D112	IPP	Programming Current on $\overline{\text{MCLR}}$/VPP	—	300	μA	
D113	IDDP	Supply Current During Programming	—	10	mA	
D031	VIL	Input Low Voltage	VSS	0.2 VDD	V	
D041	VIH	Input High Voltage	0.8 VDD	VDD	V	
D080	VOL	Output Low Voltage	—	0.6	V	IOL = 8.5 mA @ 4.5V
D090	VOH	Output High Voltage	VDD – 0.7	—	V	IOH = -3.0 mA @ 4.5V
D012	CIO	Capacitive Loading on I/O pin (PGD)	—	50	pF	To meet AC specifications
P1	TR	$\overline{\text{MCLR}}$/VPP Rise Time to enter Program/Verify mode	—	1.0	μs	**(Note 1)**
P2	TPGC	Serial Clock (PGC) Period	100	—	ns	VDD = 5.0V
			1	—	μs	VDD = 2.0V
P2A	TPGCL	Serial Clock (PGC) Low Time	40	—	ns	VDD = 5.0V
			400	—	ns	VDD = 2.0V
P2B	TPGCH	Serial Clock (PGC) High Time	40	—	ns	VDD = 5.0V
			400	—	ns	VDD = 2.0V
P3	Tset1	Input Data Setup Time to Serial Clock ↓	15	—	ns	
P4	Thld1	Input Data Hold Time from PGC ↓	15	—	ns	
P5	Tdly1	Delay between 4-bit Command and Command Operand	40	—	ns	
P5A	Tdly1a	Delay between 4-bit Command Operand and next 4-bit Command	40	—	ns	
P6	Tdly2	Delay between Last PGC ↓ of Command Byte to First PGC ↑ of Read of Data Word	20	—	ns	
P9	Tdly5	PGC High Time (minimum programming time)	1	—	ms	
P10	Tdly6	PGC Low Time after Programming (high voltage discharge time)	5	—	μs	
P11	Tdly7	Delay to allow Self-Timed Data Write or Bulk Erase to occur	10	—	ms	
P11A	Tdrwt	Data Write Polling Time	4	—	ms	
P12	Thld2	Input Data Hold Time from $\overline{\text{MCLR}}$/VPP ↑	2	—	μs	
P13	Tset2	VDD ↑ Setup Time to $\overline{\text{MCLR}}$/VPP ↑	100	—	ns	
P14	Tvalid	Data Out Valid from PGC ↑	10	—	ns	
P15	Tset3	PGM ↑ Setup Time to $\overline{\text{MCLR}}$/VPP ↑	2	—	μs	

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

APPENDIX A: REVISION HISTORY

Revision A

Initial Release

Revision B (07/2003)

Revision C (01/2010)

Revision D (10/2015)

Added Revision History; Revised Table 5-4.

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 provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE**.** Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Trademarks

The Microchip name and logo, the Microchip logo, dsPIC, FlashFlex, flexPWR, JukeBlox, KEELOQ, KEELOQ logo, Kleer, LANCheck, MediaLB, MOST, MOST logo, MPLAB, OptoLyzer, PIC, PICSTART, PIC^{32} logo, RightTouch, SpyNIC, SST, SST Logo, SuperFlash and UNI/O are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

The Embedded Control Solutions Company and mTouch are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Analog-for-the-Digital Age, BodyCom, chipKIT, chipKIT logo, CodeGuard, dsPICDEM, dsPICDEM.net, ECAN, In-Circuit Serial Programming, ICSP, Inter-Chip Connectivity, KleerNet, KleerNet logo, MiWi, motorBench, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, RightTouch logo, REAL ICE, SQI, Serial Quad I/O, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

Silicon Storage Technology is a registered trademark of Microchip Technology Inc. in other countries.

GestIC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

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

© 2003-2015, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

ISBN: 978-1-63277-909-0

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.

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:
http://www.microchip.com/support
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Austin, TX
Tel: 512-257-3370

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Cleveland
Independence, OH
Tel: 216-447-0464
Fax: 216-447-0643

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Novi, MI
Tel: 248-848-4000

Houston, TX
Tel: 281-894-5983

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

New York, NY
Tel: 631-435-6000

San Jose, CA
Tel: 408-735-9110

Canada - Toronto
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon

Hong Kong
Tel: 852-2943-5100
Fax: 852-2401-3431

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8569-7000
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

China - Chongqing
Tel: 86-23-8980-9588
Fax: 86-23-8980-9500

China - Dongguan
Tel: 86-769-8702-9880

China - Hangzhou
Tel: 86-571-8792-8115
Fax: 86-571-8792-8116

China - Hong Kong SAR
Tel: 852-2943-5100
Fax: 852-2401-3431

China - Nanjing
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

China - Qingdao
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

China - Shanghai
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

China - Shenyang
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

China - Shenzhen
Tel: 86-755-8864-2200
Fax: 86-755-8203-1760

China - Wuhan
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

China - Xian
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

ASIA/PACIFIC

China - Xiamen
Tel: 86-592-2388138
Fax: 86-592-2388130

China - Zhuhai
Tel: 86-756-3210040
Fax: 86-756-3210049

India - Bangalore
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

India - Pune
Tel: 91-20-3019-1500

Japan - Osaka
Tel: 81-6-6152-7160
Fax: 81-6-6152-9310

Japan - Tokyo
Tel: 81-3-6880- 3770
Fax: 81-3-6880-3771

Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Malaysia - Kuala Lumpur
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

Malaysia - Penang
Tel: 60-4-227-8870
Fax: 60-4-227-4068

Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Hsin Chu
Tel: 886-3-5778-366
Fax: 886-3-5770-955

Taiwan - Kaohsiung
Tel: 886-7-213-7828

Taiwan - Taipei
Tel: 886-2-2508-8600
Fax: 886-2-2508-0102

Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Dusseldorf
Tel: 49-2129-3766400

Germany - Karlsruhe
Tel: 49-721-625370

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Italy - Venice
Tel: 39-049-7625286

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Poland - Warsaw
Tel: 48-22-3325737

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

Sweden - Stockholm
Tel: 46-8-5090-4654

UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820

07/14/15