MICROCHIP

PIC18F1XK22/LF1XK22

Flash Memory Programming Specification

1.0 DEVICE OVERVIEW

This document includes the programming specifications for the following devices:

- PIC18F13K22
- PIC18LF13K22
- PIC18F14K22
- PIC18LF14K22

2.0 PROGRAMMING OVERVIEW

The PIC18F1XK22/LF1XK22 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. The PIC18F1XK22 devices operate from 1.8 to 5.5 volts and the PIC18LF1XK22 devices operate from 1.8 to 3.6 volts. All other aspects of the PIC18F1XK22 with regards to the PIC18LF1XK22 devices are identical.

2.1 Hardware Requirements

In High-Voltage ICSP mode, the PIC18F1XK22/LF1XK22 devices require two programmable power supplies: one for VDD and one for $\overline{MCLR}$/VPP/RA3. Both supplies should have a minimum resolution of 0.25V. Refer to **Section 8.0 "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, the PIC18F1XK22/LF1XK22 devices can be programmed using a single VDD source in the operating range. The $\overline{MCLR}$/VPP/RA3 does not have to be brought to a different voltage, but can instead be left at the normal operating voltage. Refer to **Section 8.0 "AC/DC Characteristics Timing Requirements for Program/Verify Test Mode"** for additional hardware parameters.

2.1.1.1 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' (enabled) from the factory.

If Single-Supply Programming mode is not used, the LVP bit can be programmed to a '0' and RC3/PGM becomes a digital I/O pin. However, the LVP bit may only be programmed by entering the High-Voltage ICSP mode, where $\overline{MCLR}$/VPP/RA3 is raised to VIHH. 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 High-Voltage ICSP mode is always available, regardless of the state of the LVP bit, by applying VIHH to the $\overline{MCLR}$/VPP/RA3 pin.
2:	While in Low-Voltage ICSP mode, the RC3 pin can no longer be used as a general purpose I/O.

2.2 Pin Diagrams

The pin diagrams for the PIC18F1XK22/LF1XK22 family are shown in Figure 2-1.

TABLE 2-1: PIN DESCRIPTIONS (DURING PROGRAMMING): PIC18F1XK22/LF1XK22

Pin Name	During Programming		
	Pin Name	**Pin Type**	**Pin Description**
$\overline{MCLR}$/VPP/RA3	VPP	P	Programming Enable
VDD(2)	VDD	P	Power Supply
VSS(2)	VSS	P	Ground
RC3	PGM	I	Low-Voltage ICSP™ input when LVP Configuration bit equals '1'(1)
RA1	PGC	I	Serial Clock
RA0	PGD	I/O	Serial Data

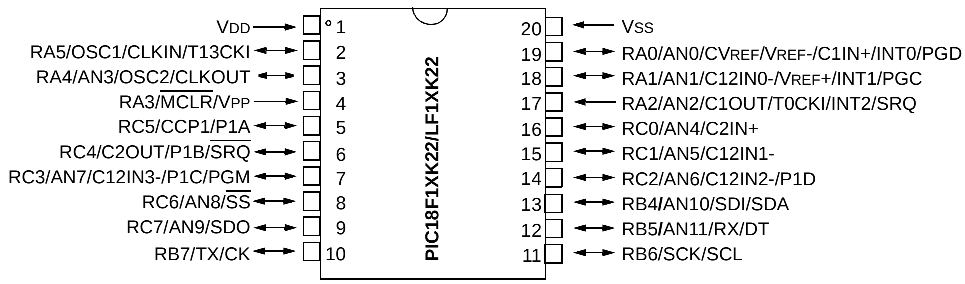
Legend: I = Input, O = Output, P = Power

Note 1: See Figure 6-1 for more information.

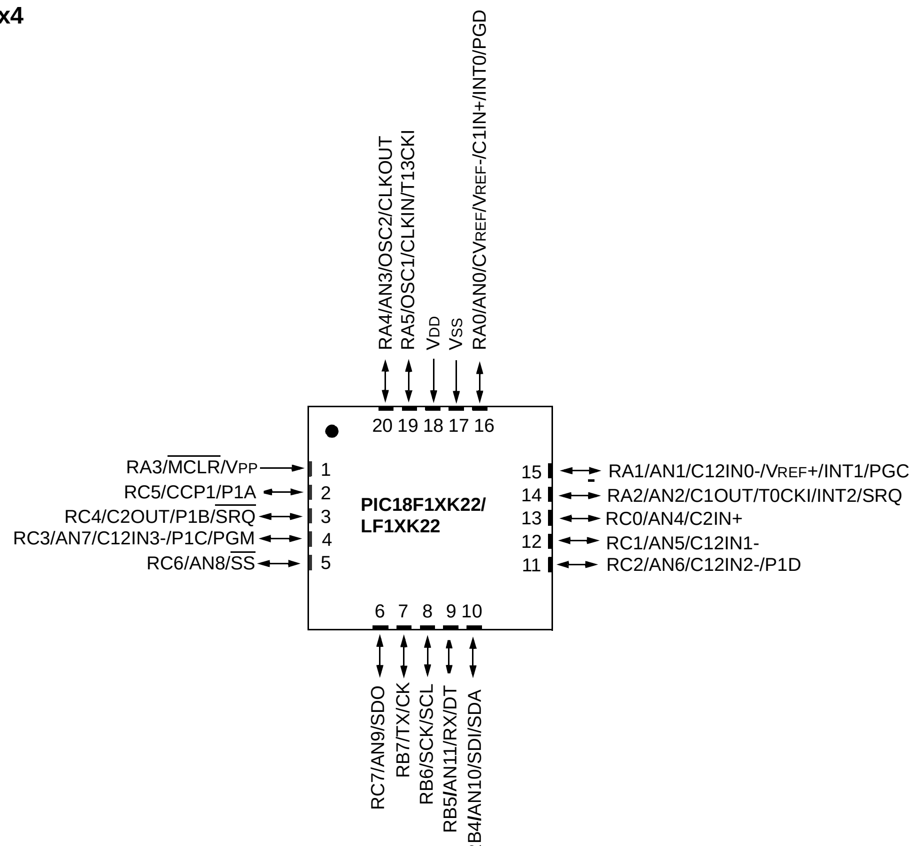
2: All power supply (VDD) and ground (VSS) pins must be connected.

FIGURE 2-1: 20-PIN PDIP, SSOP AND SOIC PIN DIAGRAM FOR PIC18F1XK22/LF1XK22

20-pin PDIP, SSOP, SOIC (300 MIL)



20-Pin QFN 4x4



3.0 MEMORY MAPS

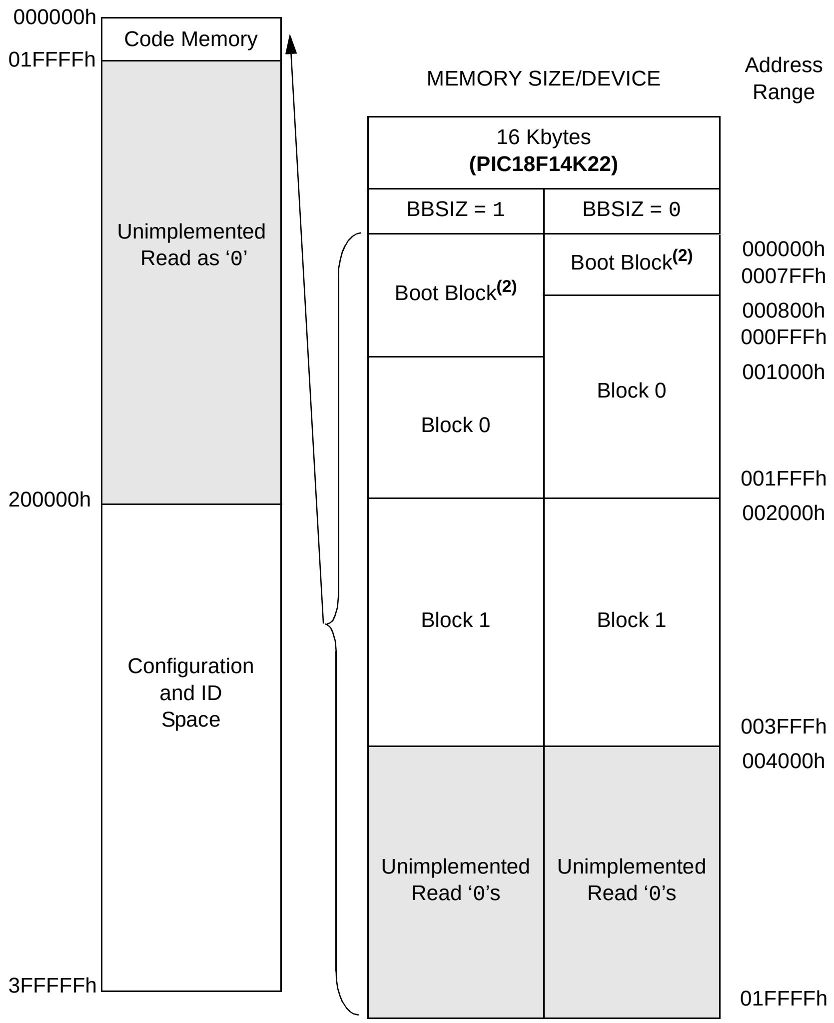
For the PIC18F14K22/LF14K22 device, the code memory space extends from 0000h to 03FFFh (16 Kbytes) in two 8-Kbyte blocks. For the PIC18F13K22/LF13K22 device, the code memory space extends from 0000h to 01FFFh (8 Kbytes) in two 4-Kbyte blocks.

For the PIC18F14K22/LF14K22 addresses 0000h through 0FFFh, however, define a "Boot Block" region that is treated separately from Block 0. For the PIC18F13K22/LF13K22 addresses 0000h through 07FFh, define the "Boot Block" region. All of these blocks define code protection boundaries within the code memory space. The size of the Boot Block in the PIC18F14K22/LF14K22 devices can be configured as 2K, or 4 Kbyte (see Figure 3-1). The size of the Boot Block in the PIC18F13K22/LF13K22 devices can be configured as 1K, or 2 Kbytes (see Figure 3-1). This is done through the BBSIZ bit in the Configuration register, CONFIG4L. It is important to note that increasing the size of the Boot Block decreases the size of the Block 0.

TABLE 3-1: IMPLEMENTATION OF CODE MEMORY

Device	Code Memory Size (Bytes)
PIC18F13K22/ LF13K22	000000h-001FFFh (8K)
PIC18F14K22/ LF14K22	000000h-003FFFh (16K)

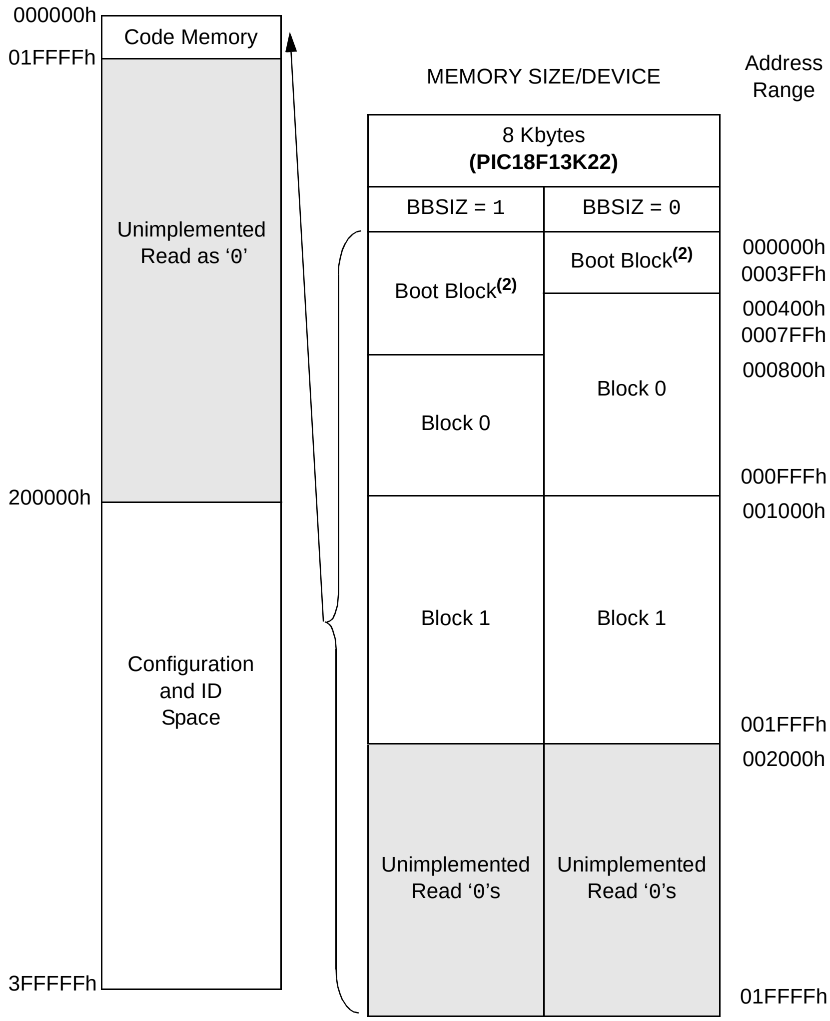
FIGURE 3-1: MEMORY MAP AND THE CODE MEMORY SPACE FOR PIC18F14K22/LF14K22 DEVICES(1)



Note 1: Sizes of memory areas are not to scale.

2: Boot Block size is determined by the BBSIZ bit in the CONFIG4L register.

FIGURE 3-2: MEMORY MAP AND THE CODE MEMORY SPACE FOR PIC18F13K22/LF13K22 DEVICES(1)



Note 1: Sizes of memory areas are not to scale.

2: Boot Block size is determined by the BBSIZ bit in the CONFIG4L register.

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 3-3.

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 300001h through 30000Dh are reserved for the Configuration bits. These bits select various device options and are described in **Section 6.0 "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 6.0 "Configuration Word"**. These device ID bits read out normally, even after code protection.

3.0.1 MEMORY ADDRESS POINTER

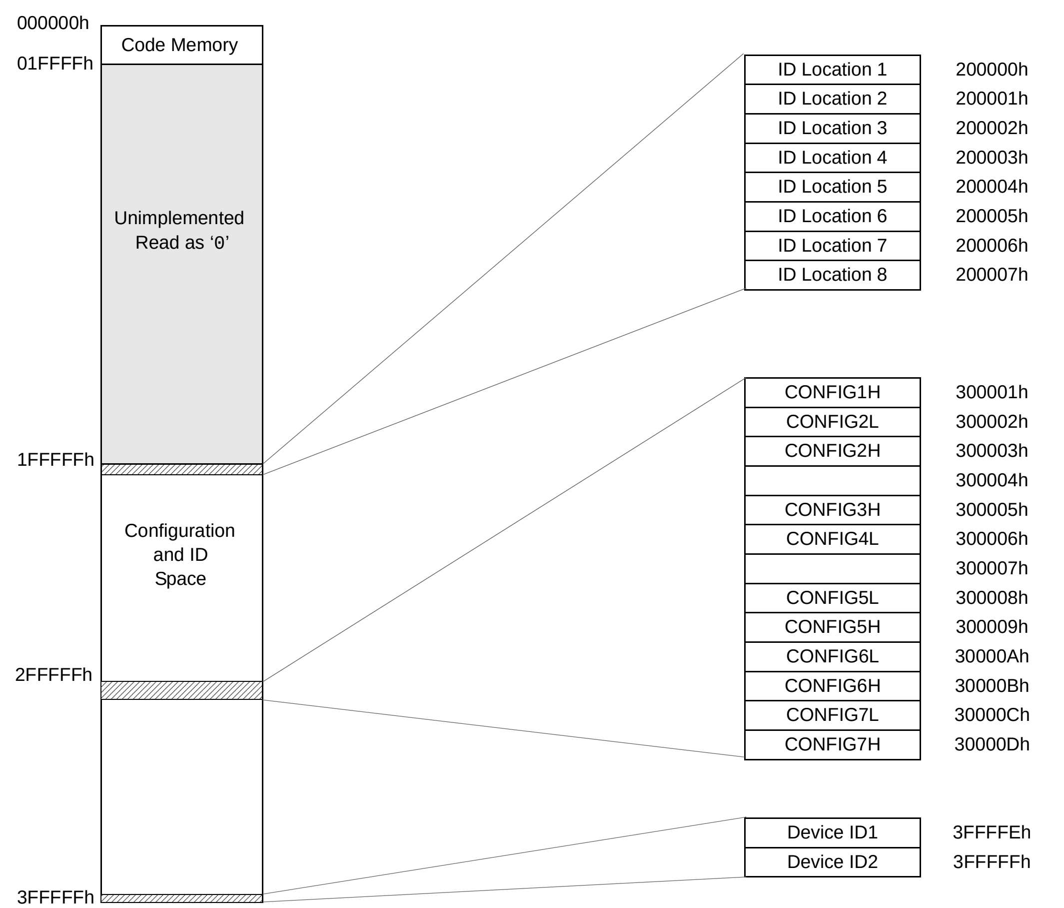
Memory in the address space, 0000000h to 3FFFFFh, is addressed via the Table Pointer register, which is comprised of three Pointer registers:

- 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 any read or write operations.

FIGURE 3-3: CONFIGURATION AND ID LOCATIONS FOR PIC18F1XK22/LF1XK22 DEVICES

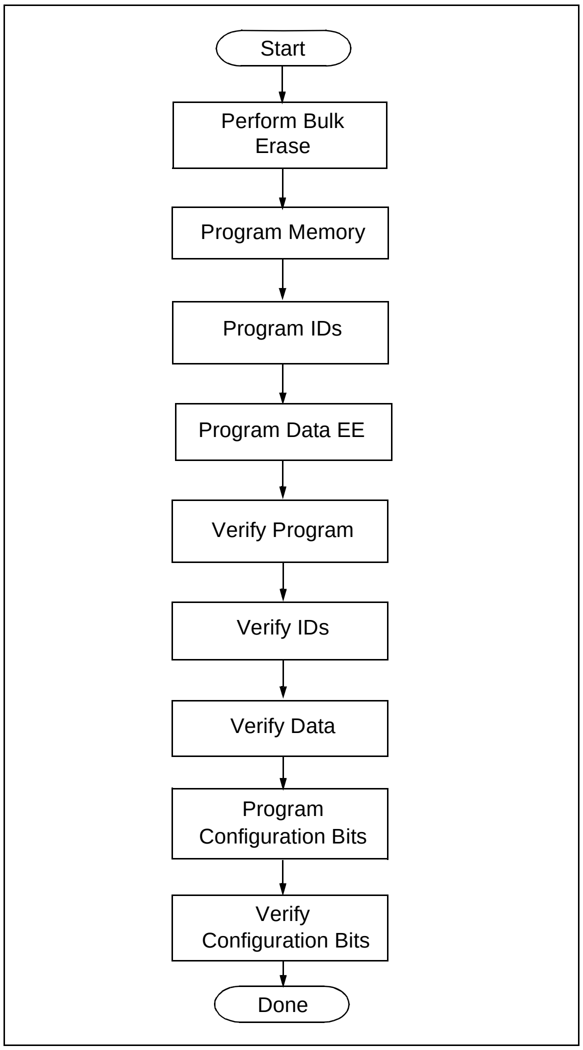


Note: Sizes of memory areas are not to scale.

3.1 High-Level Overview of the Programming Process

Figure 3-4 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.

FIGURE 3-4: HIGH-LEVEL PROGRAMMING FLOW

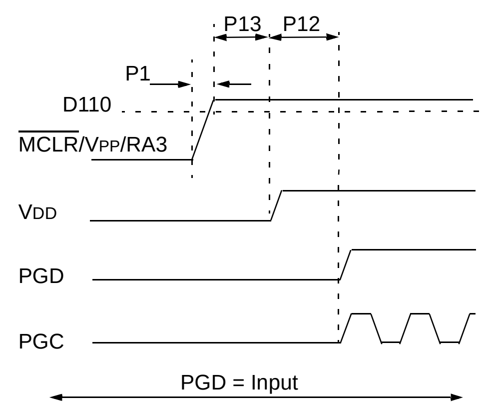


3.2 Entering and Exiting High-Voltage ICSP Program/Verify Mode

As shown in Figure 3-6, the High-Voltage ICSP Program/Verify mode is entered by holding PGC and PGD low and then raising $\overline{MCLR}$/VPP/RA3 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. Figure 3-7 shows the exit sequence.

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

FIGURE 3-5: VPP-FIRST PROGRAM/VERIFY MODE ENTRY



Note: This method of entry is valid, regardless of Configuration Word selected.

FIGURE 3-6: VDD-FIRST PROGRAM/VERIFY MODE ENTRY

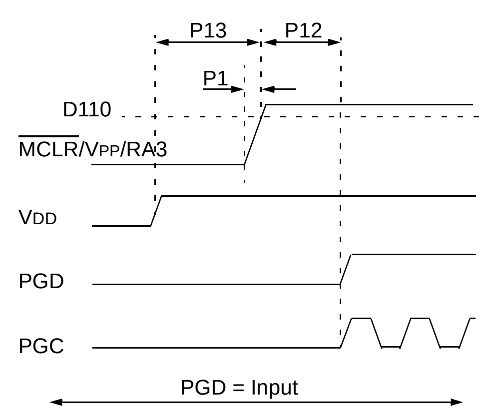


FIGURE 3-7: EXITING HIGH-VOLTAGE PROGRAM/VERIFY MODE

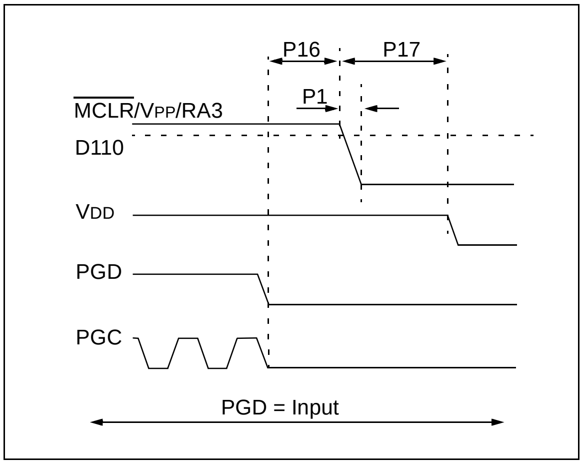
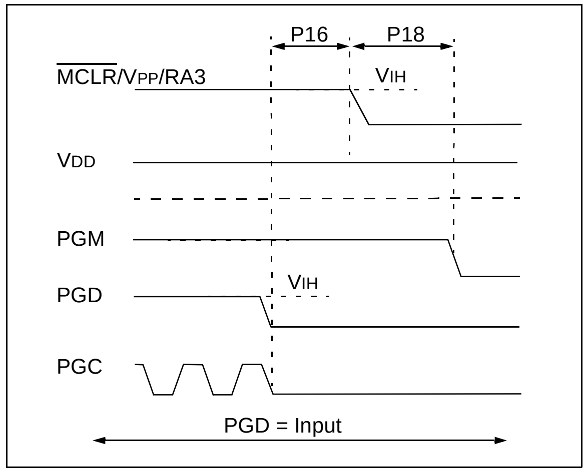


FIGURE 3-9: EXITING LOW-VOLTAGE PROGRAM/VERIFY MODE

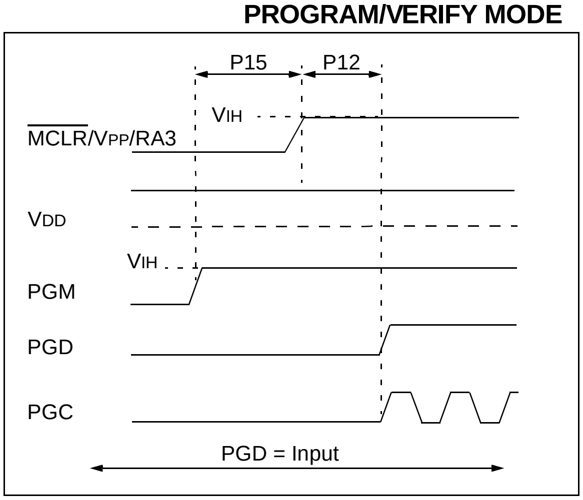


3.3 Entering and Exiting Low-Voltage ICSP Program/Verify Mode

When the LVP Configuration bit is '1' (see **Section 2.1.1.1 "Single-Supply ICSP Programming"**), the Low-Voltage ICSP mode is enabled. As shown in Figure 3-8, Low-Voltage ICSP Program/Verify mode is entered by holding PGC and PGD low, placing a logic high on PGM and then raising MCLR/VPP/RA3 to VIH. In this mode, the RC3/PGM pin is dedicated to the programming function and ceases to be a general purpose I/O pin. Figure 3-9 shows the exit sequence.

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

FIGURE 3-8: ENTERING LOW-VOLTAGE PROGRAM/VERIFY MODE



3.4 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.

3.4.1 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 3-2.

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 3-3. The 4-bit command, Most Significant bit (MSb), is shown first. The command operand, or “Data Payload”, is shown <MSB><LSB>. Figure 3-10 demonstrates how to serially present a 20-bit command/operand to the device.

3.4.2 CORE INSTRUCTION

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

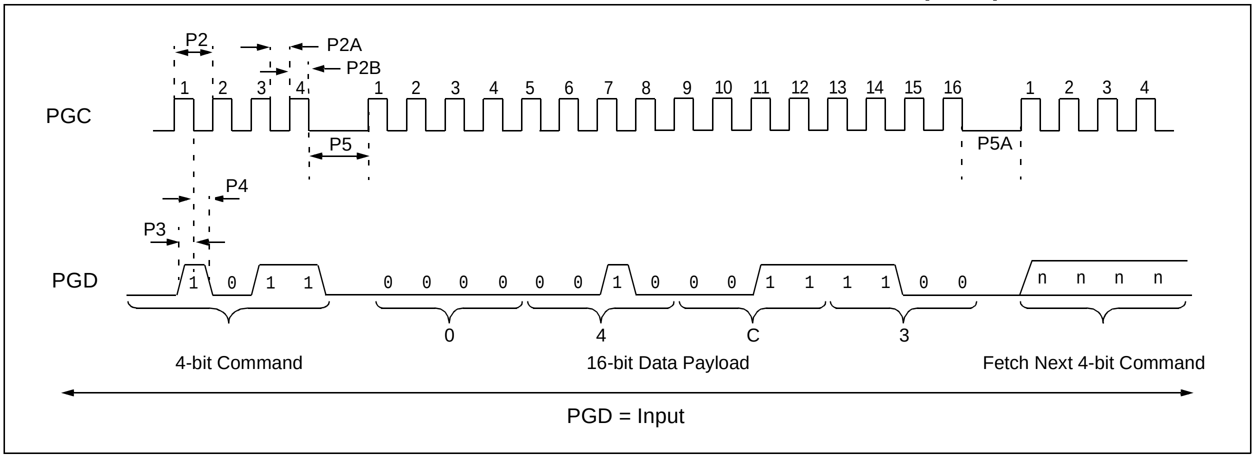
TABLE 3-2: COMMANDS FOR PROGRAMMING

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 3-3: SAMPLE COMMAND SEQUENCE

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

FIGURE 3-10: TABLE WRITE, POST-INCREMENT TIMING DIAGRAM (1101)



4.0 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 or erase.

4.1 ICSP Erase

4.1.1 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 4-1. If data EEPROM is code-protected (CPD = 0), the user must request an erase of data EEPROM (e.g., 00008484h as shown in Table 4-1).

TABLE 4-1: BULK ERASE OPTIONS

Description	Data (3C0005h:3C0004h)
Chip Erase	0F0F8787h
Erase Data EEPROM	00008484h
Erase Boot Block	00008181h
Erase Config Bits	00008282h
Erase Code EEPROM Block 0	01018080h
Erase Code EEPROM Block 1	02028080h
Erase Code EEPROM Block 2	04048080h
Erase Code EEPROM Block 3	08088080h

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 4-2 and the flowchart is shown in Figure 4-1.

> **Note:** A Bulk Erase is the only way to reprogram code-protect bits from an "on" state to an "off" state.

TABLE 4-2: 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 4-1: BULK ERASE FLOW

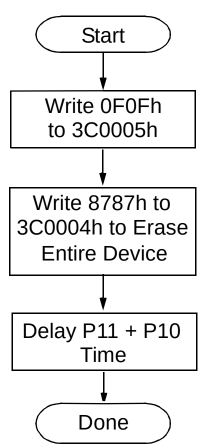
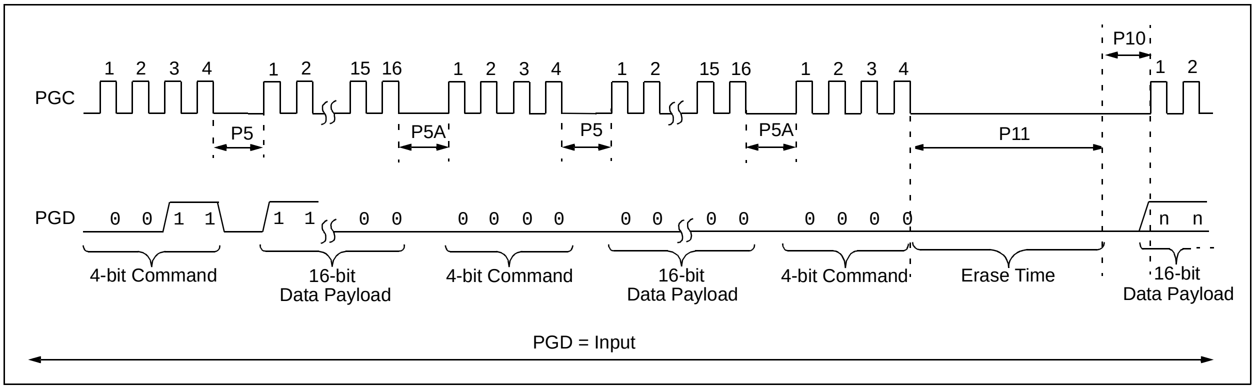


FIGURE 4-2: BULK ERASE TIMING DIAGRAM



4.1.2 LOW-VOLTAGE ICSP BULK ERASE

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

If it is determined that a program memory erase must be performed at a supply voltage below the Bulk Erase limit, refer to the erase methodology described in **Section 4.1.3 "ICSP Row Erase"** and **Section 4.2.1 "Modifying Code Memory"**.

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 4.3 "Data EEPROM Programming"** and write '1's to the array.

4.1.3 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 3.0 "Memory Maps"**).

The Row Erase duration is self-timed. After the WR bit in EECON1 is set, two NOPs are issued. Erase starts upon the 4th PGC of the second NOP. It ends when the WR bit is cleared by hardware.

The code sequence to Row Erase a PIC18F1XK22/LF1XK22 device is shown in Table 4-3. The flowchart shown in Figure 4-3 depicts the logic necessary to completely erase the PIC18F1XK22/LF1XK22 devices. The timing diagram for Row Erase is identical to the data EEPROM write timing shown in Figure 4-7.

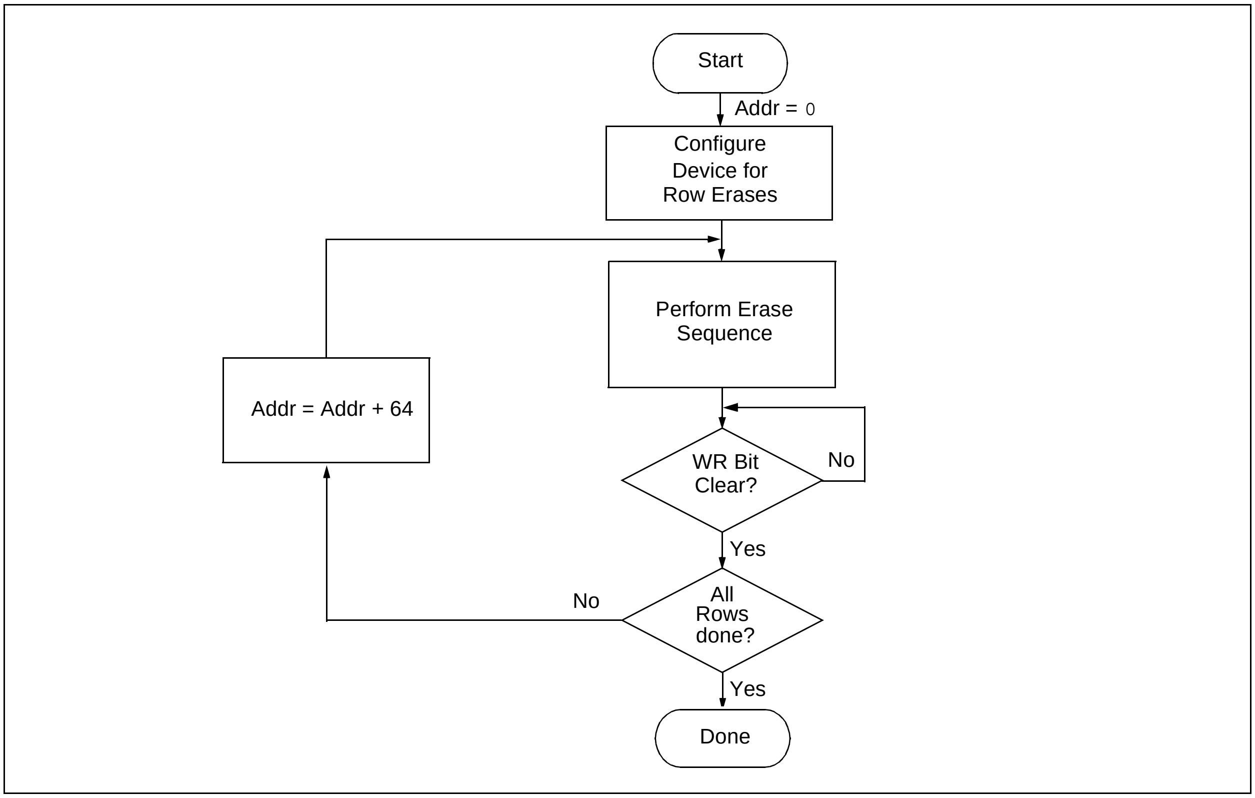
Note:	The TBLPTR register can point at any byte within the row intended for erase.

TABLE 4-3: 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 0000	88 A6 82 A6 00 00 00 00	BSF EECON1, FREE BSF EECON1, WR NOP NOP Erase starts on the 4th clock of this instruction
Step 4: Poll WR bit. Repeat until bit is clear.		
0000 0000 0000 0010	50 A6 6E F5 00 00 <MSB><LSB>	MOVF EECON1, W, 0 MOVWF TABLAT NOP Shift out data(1)
Step 5: Hold PGC low for time P10.		
Step 6: Repeat step 3 with Address Pointer incremented by 64 until all rows are erased.		
Step 7: Disable writes.		
0000	94 A6	BCF EECON1, WREN

Note 1: See Figure 5-4 for details on shift out data timing.

FIGURE 4-3: SINGLE ROW ERASE CODE MEMORY FLOW



 Advance Information

4.2 Code Memory Programming

Programming code memory is accomplished by first loading data into the write buffer and then initiating a programming sequence. The write and erase buffer sizes shown in Table 4-4 can be mapped to any location of the same size beginning at 000000h. The actual memory write sequence takes the contents of this buffer and programs the proper amount of code memory 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 PIC18F1XK22/LF1XK22 device is shown in Table 4-5. The flowchart shown in Figure 4-4 depicts the logic necessary to completely write a PIC18F1XK22/LF1XK22 device. The timing diagram that details the Start Programming command and parameters P9 and P10 is shown in Figure 4-5.

Note:	The TBLPTR register must point to the same region when initiating the programming sequence as it did when the write buffers were loaded.

TABLE 4-4: WRITE AND ERASE BUFFER SIZES

Devices	Write Buffer Size (bytes)	Erase Size (bytes)
PIC18F14K22	16	64
PIC18F13K22	8	64

TABLE 4-5: WRITE CODE MEMORY CODE SEQUENCE

4-bit Command	Data Payload	Core Instruction
Step 1: Direct access to code memory.		
0000 0000 0000	8E A6 9C A6 84 A6	BSF EECON1, EEPGD BCF EECON1, CFGS BSF EECON1, WREN
Step 2: Point to row to write.		
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: Load write buffer. 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 and start programming.		
1111 0000	<MSB><LSB> 00 00	Write 2 bytes and start programming. NOP - hold PGC high for time P9 and low for time P10.
To continue writing data, repeat steps 2 through 4, where the Address Pointer is incremented by 2 at each iteration of the loop.		

FIGURE 4-4: PROGRAM CODE MEMORY FLOW

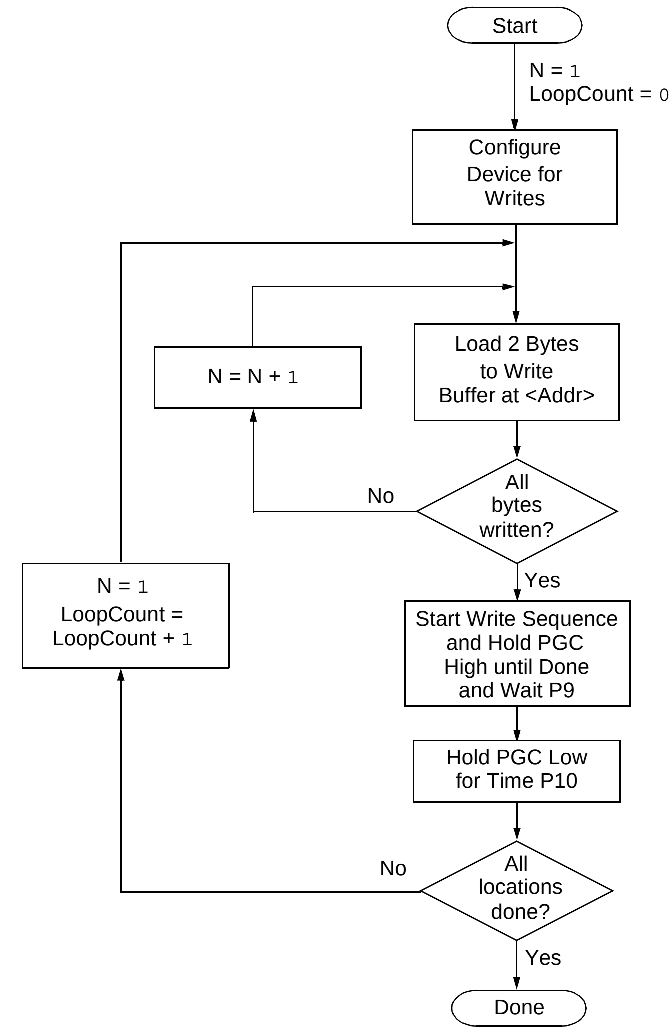
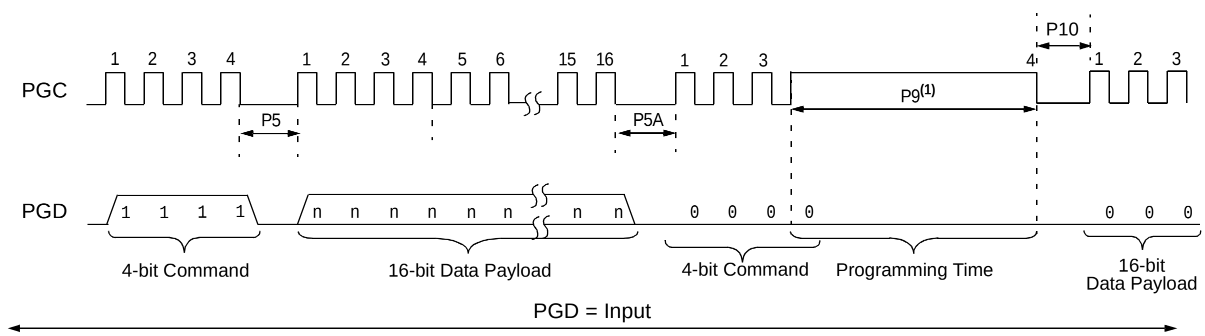


FIGURE 4-5: TABLE WRITE AND START PROGRAMMING INSTRUCTION TIMING DIAGRAM (1111)



Note 1: Use P9A for Configuration Word programming.

 Advance Information

4.2.1 MODIFYING CODE MEMORY

The previous programming example assumed that the device has been Bulk Erased prior to programming (see **Section 4.1.1 “High-Voltage ICSP Bulk Erase”**). It may be the case, however, that the user wishes to modify only a section of an already programmed device.

The appropriate number of bytes required for the erase buffer must be read out of code memory (as described in **Section 5.2 “Verify Code Memory and ID Locations”**) and buffered. Modifications can be made on this buffer. Then, the block of code memory that was read out must be erased and rewritten with the modified data.

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

TABLE 4-6: MODIFYING CODE MEMORY

4-bit Command	Data Payload	Core Instruction
Step 1: Direct access to code memory.		
0000	8E A6	`BSF EECON1, EEPGD`
0000	9C A6	`BCF EECON1, CFGS`
Step 2: Read code memory into buffer (**Section 5.1 “Read Code Memory, ID Locations and Configuration Bits”**).		
Step 3: Set the Table Pointer for the block to be erased.		
0000	0E <Addr[21:16]>	`MOVLW <Addr[21:16]>`
0000	6E F8	`MOVWF TBLPTRU`
0000	0E <Addr[8:15]>	`MOVLW <Addr[8:15]>`
0000	6E F7	`MOVWF TBLPTRH`
0000	0E <Addr[7:0]>	`MOVLW <Addr[7:0]>`
0000	6E F6	`MOVWF TBLPTRL`
Step 4: Enable memory writes and setup an erase.		
0000	84 A6	`BSF EECON1, WREN`
0000	88 A6	`BSF EECON1, FREE`
Step 5: Initiate erase.		
0000	88 A6	`BSF EECON1, FREE`
0000	82 A6	`BSF EECON1, WR`
0000	00 00	`NOP`
0000	00 00	`NOP Erase starts on the 4th clock of this instruction`
Step 6: Poll WR bit. Repeat until bit is clear.		
0000	50 A6	`MOVF EECON1, W, 0`
0000	6E F5	`MOVWF TABLAT`
0000	00 00	`NOP`
0000	<MSB><LSB>	`Shift out data`[1]
Step 7: Load write buffer. The correct bytes will be selected based on the Table Pointer.		
0000	0E <Addr[21:16]>	`MOVLW <Addr[21:16]>`
0000	6E F8	`MOVWF TBLPTRU`
0000	0E <Addr[8:15]>	`MOVLW <Addr[8:15]>`
0000	6E F7	`MOVWF TBLPTRH`
0000	0E <Addr[7:0]>	`MOVLW <Addr[7:0]>`
0000	6E F6	`MOVWF TBLPTRL`
1101	<MSB><LSB>	`Write 2 bytes and post-increment address by 2.`
•	•	
•	•	`Repeat as many times as necessary to fill the write buffer`
•	•	`Write 2 bytes and start programming.`
1111	<MSB><LSB>	`NOP - hold PGC high for time P9 and low for time P10.`
0000	00 00	
To continue modifying data, repeat Steps 2 through 6, where the Address Pointer is incremented by the appropriate number of bytes (see Table 4-4) at each iteration of the loop. The write cycle must be repeated enough times to completely rewrite the contents of the erase buffer.		
Step 8: Disable writes.		
0000	94 A6	`BCF EECON1, WREN`

4.3 Data EEPROM Programming

Data EEPROM is accessed one byte at a time via an Address Pointer (register EEADR) and a data latch (EEDATA). Data EEPROM is written by loading 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 24th PGC after the WR bit is set. It ends when the WR bit is cleared by hardware.

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

FIGURE 4-6: PROGRAM DATA FLOW

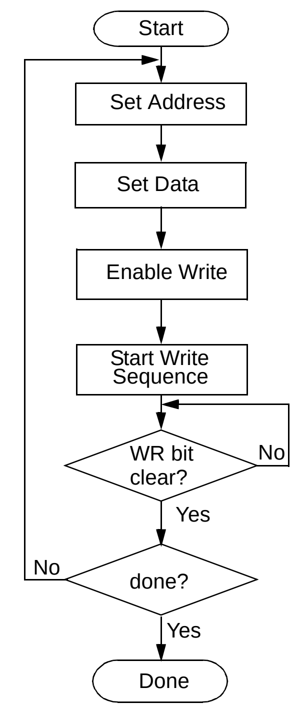
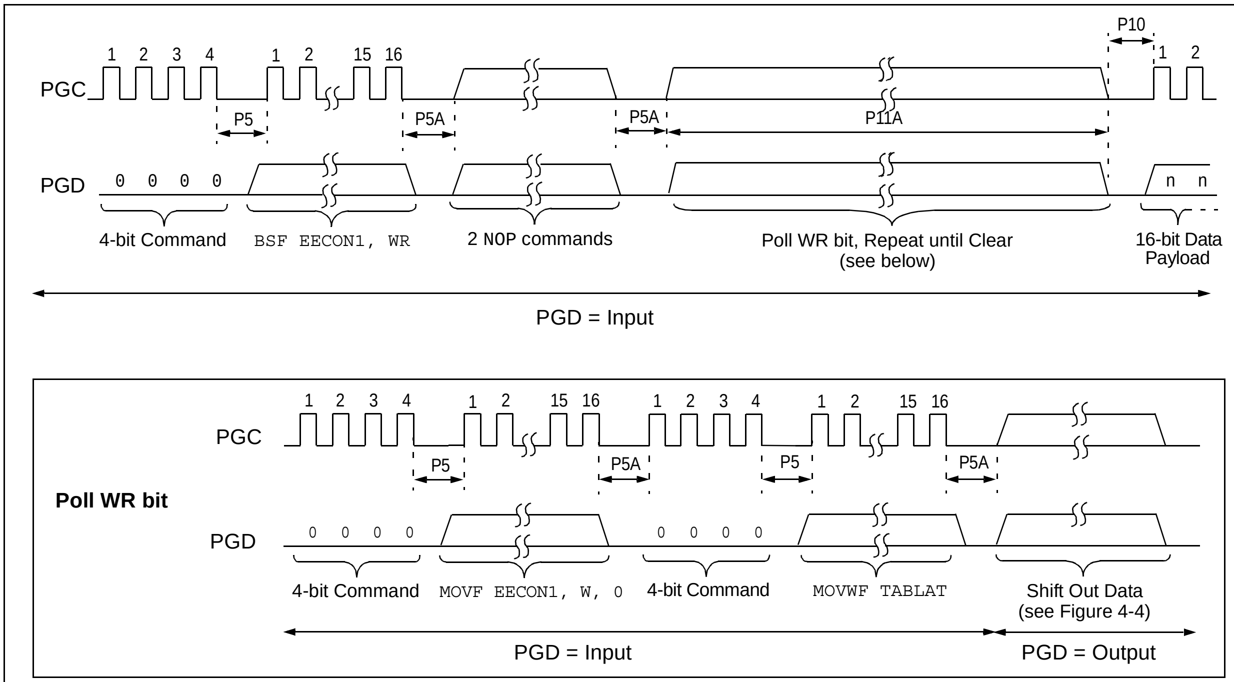


FIGURE 4-7: DATA EEPROM WRITE TIMING DIAGRAM



 Advance Information

TABLE 4-7: 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	0E <Addr> 6E A9	MOVLW <Addr> MOVWF EEADR
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: Initiate write.		
0000 0000 0000	82 A6 00 00 00 00	BSF EECON1, WR NOP NOP ;write starts on 4th clock of this instruction
Step 6: Poll WR bit, repeat until the bit is clear.		
0000 0000 0000 0010	50 A6 6E F5 00 00 <MSB><LSB>	MOVF EECON1, W, 0 MOVWF TABLAT NOP Shift out data(1)
Step 7: Hold PGC low for time P10.		
Step 8: Disable writes.		
0000	94 A6	BCF EECON1, WREN
Repeat steps 2 through 8 to write more data.		

Note 1: See Figure 5-4 for details on shift out data timing.

4.4 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 4-8 demonstrates the code sequence required to write the ID locations.

In order to modify the ID locations, refer to the methodology described in **Section 4.2.1 "Modifying Code Memory"**. As with code memory, the ID locations must be erased before being modified.

TABLE 4-8: WRITE ID SEQUENCE

4-bit Command	Data Payload	Core Instruction
Step 1: Direct access to code memory.		
0000	8E A6	`BSF EECON1, EEPGD`
0000	9C A6	`BCF EECON1, CFGS`
0000	84 A6	`BSF EECON1, WREN`
Step 2: Set Table Pointer to ID. Load write buffer with 8 bytes and write.		
0000	0E 20	`MOVLW 20h`
0000	6E F8	`MOVWF TBLPTRU`
0000	0E 00	`MOVLW 00h`
0000	6E F7	`MOVWF TBLPTRH`
0000	0E 00	`MOVLW 00h`
0000	6E F6	`MOVWF TBLPTRL`
1101	<MSB><LSB>	`Write 2 bytes and post-increment address by 2.`
1101	<MSB><LSB>	`Write 2 bytes and post-increment address by 2.`
1101	<MSB><LSB>	`Write 2 bytes and post-increment address by 2.`
1111	<MSB><LSB>	`Write 2 bytes and start programming.`
0000	00 00	`NOP - hold PGC high for time P9 and low for time P10.`

4.5 Boot Block Programming

The code sequence detailed in Table 4-5 should be used, except that the address used in “Step 2” will be in the following ranges:

If BBSIZ = 0:

000000h-0003FFh for PIC18F13K22/LF13K22

000000h-0007FFh for PIC18F14K22/LF14K22

If BBSIZ = 1:

000000h-0007FFh for PIC18F13K22/LF13K22

000000h-000FFFh for PIC18F14K22/LF14K22

4.6 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 4-9. See Figure 4-5 for the timing diagram.

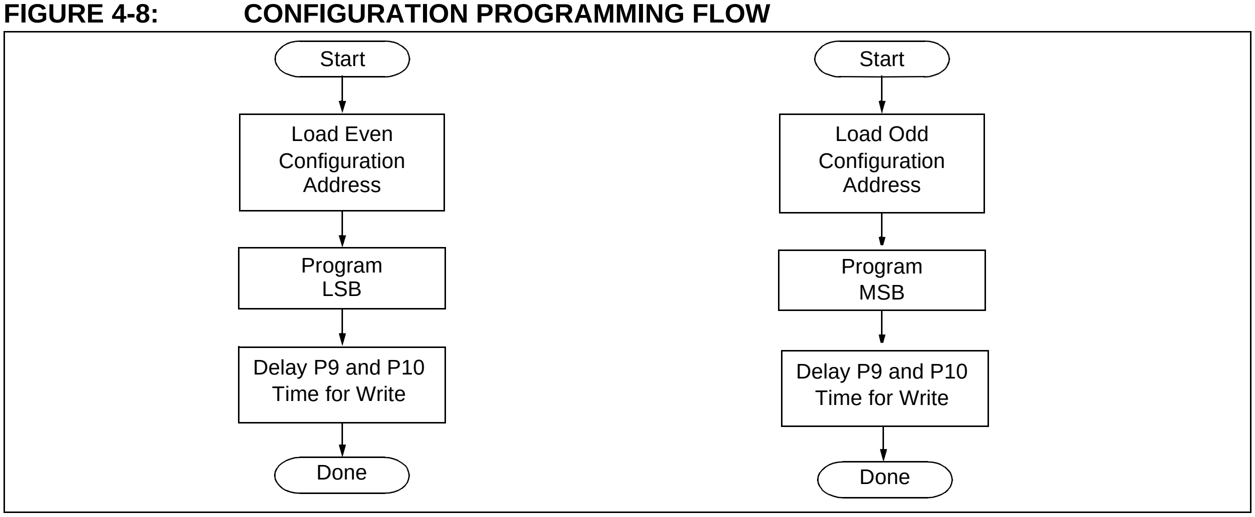
Note:	The address must be explicitly written for each byte programmed. The addresses can not be incremented in this mode.

TABLE 4-9: SET ADDRESS POINTER TO CONFIGURATION LOCATION

4-bit Command	Data Payload	Core Instruction
Step 1: Direct access to configuration memory.		
0000	8E A6	BSF EECON1, EEPGD
0000	8C A6	BSF EECON1, CFGS
0000	84 A6	BSF EECON1, WREN
Step 2[(1)]: Set Table Pointer for configuration byte to be written. Write even/odd addresses.		
0000	0E 30	MOVLW 30h
0000	6E F8	MOVWF TBLPTRU
0000	0E 00	MOVLW 00h
0000	6E F7	MOVWF TBLPRTH
0000	0E 00	MOVLW 00h
0000	6E F6	MOVWF TBLPTRL
1111	<MSB ignored><LSB>	Load 2 bytes and start programming.
0000	00 00	NOP - hold PGC high for time P9 and low for time P10.
0000	0E 01	MOVLW 01h
0000	6E F6	MOVWF TBLPTRL
1111	<MSB><LSB ignored>	Load 2 bytes and start programming.
0000	00 00	NOP - hold PGC high for time P9A and low for time P10.

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 4-8: CONFIGURATION PROGRAMMING FLOW



5.0 READING THE DEVICE

5.1 Read Code Memory, ID Locations and Configuration Bits

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

The 4-bit command is shifted in LSb first. The read is executed during the next 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 5-1). This operation also increments the Table Pointer by one, pointing to the next byte in code memory for the next read.

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

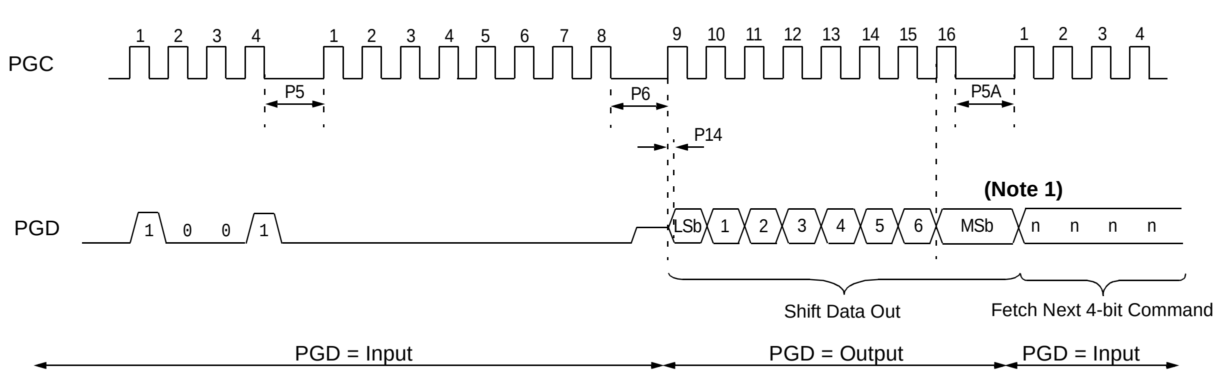
TABLE 5-1: READ CODE MEMORY SEQUENCE

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 5-1: TABLE READ POST-INCREMENT INSTRUCTION TIMING DIAGRAM (`1001`)



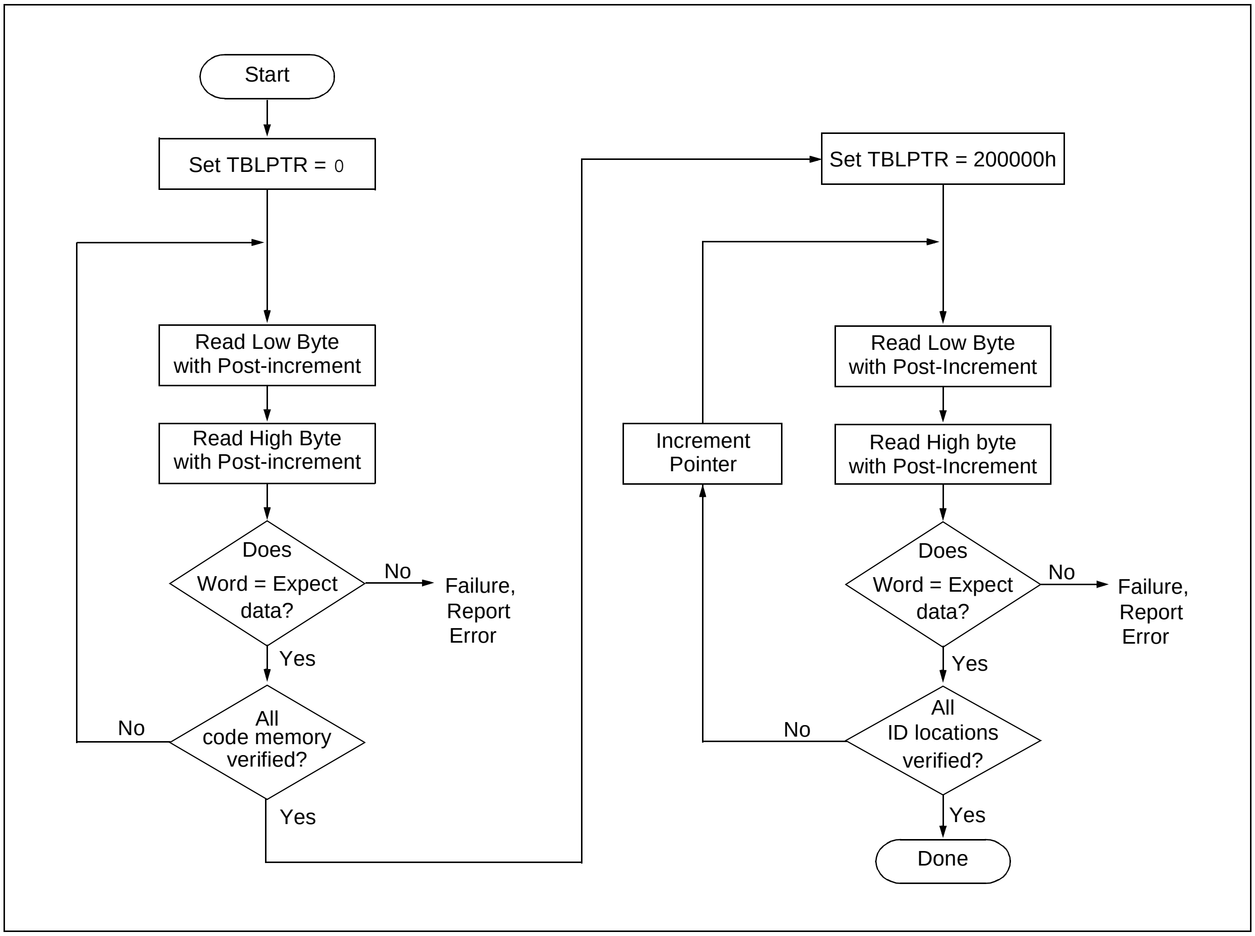
Note 1: Magnification of the high-impedance delay between PGC and PGD is shown in Figure 5-6.

5.2 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 5.1 "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 can 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 5-2: VERIFY CODE MEMORY FLOW



5.3 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 5.1 “Read Code Memory, ID Locations and Configuration Bits”** for implementation details of reading configuration data.

5.4 Read Data EEPROM Memory

Data EEPROM is accessed one byte at a time via an Address Pointer (register EEADR) and a data latch (EEDATA). Data EEPROM is read by loading 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 5-4).

The command sequence to read a single byte of data is shown in Table 5-2.

FIGURE 5-3: READ DATA EEPROM FLOW

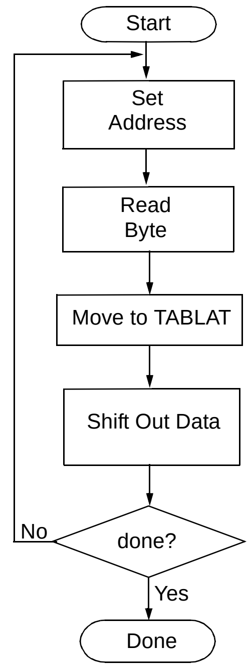
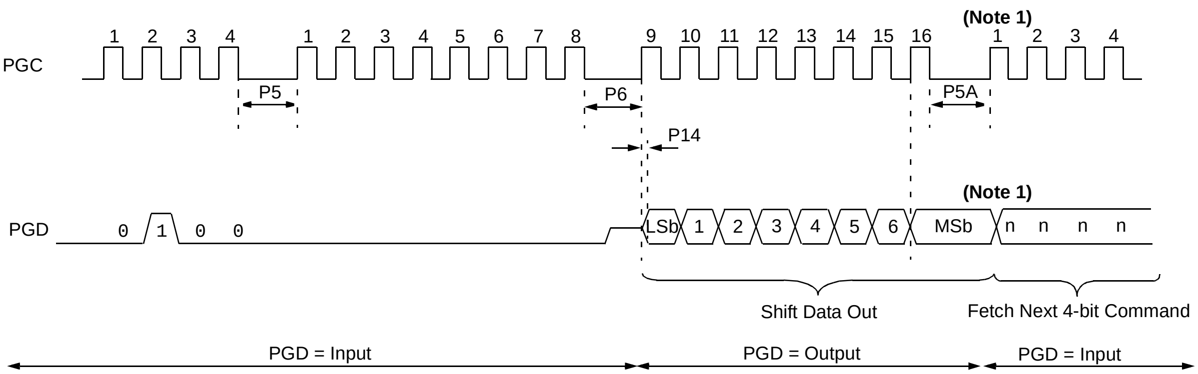


TABLE 5-2: READ DATA EEPROM MEMORY

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	0E <Addr> 6E A9	MOVLW <Addr> MOVWF EEADR
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)

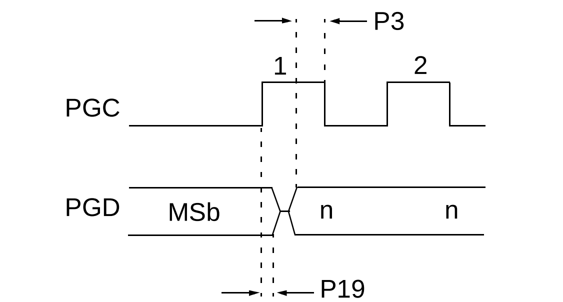
Note 1: The <LSB> is undefined. The <MSB> is the data.

FIGURE 5-4: SHIFT OUT DATA HOLDING REGISTER TIMING DIAGRAM (`0010`)



Note 1: Magnification of the high-impedance delay between PGC and PGD is shown in Figure 5-5.

FIGURE 5-5: HIGH-IMPEDANCE DELAY



5.5 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`’ (TABLAT register). The result may then be immediately compared to the appropriate data in the programmer’s memory for verification. Refer to **Section 5.4 “Read Data EEPROM Memory”** for implementation details of reading data EEPROM.

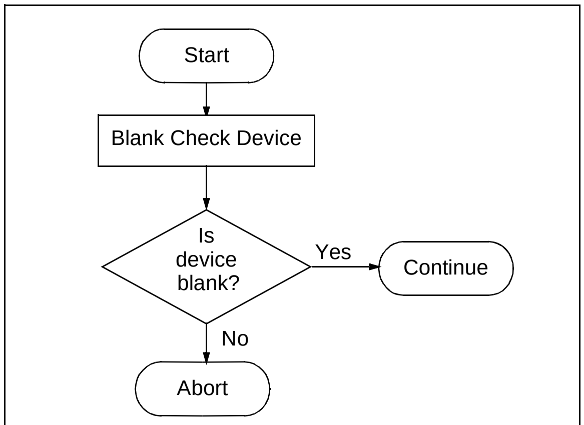
5.6 Blank Check

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`’. Therefore, 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 6-1 for blank configuration expect data for the various PIC18F1XK22/LF1XK22 devices.

Given that Blank Checking is merely code and data EEPROM verification with FFh expect data, refer to **Section 5.4 “Read Data EEPROM Memory”** and **Section 5.2 “Verify Code Memory and ID Locations”** for implementation details.

FIGURE 5-6: BLANK CHECK FLOW



6.0 CONFIGURATION WORD

The PIC18F1XK22/LF1XK22 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 protection. See Table 6-1 for a list of Configuration bits and device IDs and Table 6-3 for the Configuration bit descriptions.

6.1 ID Locations

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

6.2 Device ID Word

The device ID word for the PIC18F1XK22/LF1XK22 devices is located at 3FFFFEh:3FFFFFh. These bits may be used by the programmer to identify what device type is being programmed and read out normally, even after code or read protection. See Table 6-2 for a complete list of device ID values.

FIGURE 6-1: READ DEVICE ID WORD FLOW

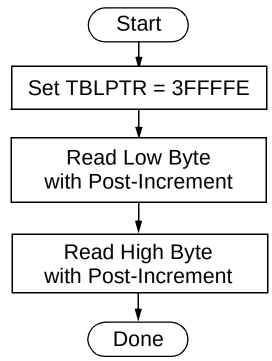


TABLE 6-1: CONFIGURATION BITS AND DEVICE IDs

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default/ Unprogrammed Value
300001h	CONFIG1H	IESO	FCMEN	PRI_CLK_EN	PLL_EN	FOSC3	FOSC2	FOSC1	FOSC0	0010 0111
300002h	CONFIG2L	—	—	—	BORV1	BORV0	BOREN1	BOREN0	$\overline{\text{PWRTEN}}$	---1 1111
300003h	CONFIG2H	—	—	—	WDPS3	WDPS2	WDPS1	WDPS0	WDTEN	---1 1111
300005h	CONFIG3H	MCLRE	—	—	—	HFOFST	—	—	—	1--- 1---
300006h	CONFIG4L	$\overline{\text{BKBUG}}$	ENHCPU	—	—	BBSIZ	LVP	—	STVREN	10-- 01-1
300008h	CONFIG5L	—	—	—	—	—	—	CP1	CP0	---- --11
300009h	CONFIG5H	CPD	CPB	—	—	—	—	—	—	11-- ----
30000Ah	CONFIG6L	—	—	—	—	—	—	WRT1	WRT0	---- --11
30000Bh	CONFIG6H	WRTD	WRTB	WRTC	—	—	—	—	—	111- ----
30000Ch	CONFIG7L	—	—	—	—	—	—	EBTR1	EBTR0	---- --11
30000Dh	CONFIG7H	—	EBTRB	—	—	—	—	—	—	-1-- ----
3FFFFEh	DEVID1(2)	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	See Table 6-2
3FFFFFh	DEVID2(2)	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	See Table 6-2

Legend: x = unknown, u = unchanged, – = unimplemented. Shaded cells are unimplemented, read as '0'.
Note 1: These bits are only implemented on specific devices. Refer to **Section 3.0 "Memory Maps"** to determine which bits apply based on available memory.
2: DEVID registers are read-only and cannot be programmed by the user.

TABLE 6-2: DEVICE ID VALUE

Device	Device ID Value	
	DEVID2	DEVID1
PIC18LF13K22	4Fh	100x xxxx
PIC18LF14K22	4Fh	011x xxxx
PIC18F13K22	4Fh	010x xxxx
PIC18F14K22	4Fh	001x xxxx

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

TABLE 6-3: PIC18F1XK22/LF1XK22 BIT DESCRIPTIONS

Bit Name	Configuration Words	Description
IESO	CONFIG1H	Internal External Switchover bit 1 = Internal External Switchover mode enabled 0 = Internal External Switchover mode disabled
FCMEN	CONFIG1H	Fail-Safe Clock Monitor Enable bit 1 = Fail-Safe Clock Monitor enabled 0 = Fail-Safe Clock Monitor disabled
PRI_CLK_EN	CONFIG1H	Primary Clock Enable bit 1 = Primary Clock enabled 0 = Primary Clock disabled
PLL_EN	CONFIG1H	4 X PLL Enable bit 1 = Oscillator multiplied by 4 0 = Oscillator used directly
FOSC<3:0>	CONFIG1H	Oscillator Selection bits 1111 = External RC oscillator, CLKOUT function on OSC2 1110 = External RC oscillator, CLKOUT function on OSC2 1101 = EC oscillator (low) 1100 = EC oscillator, CLKOUT function on OSC2 (low) 1011 = EC oscillator (medium) 1010 = EC oscillator, CLKOUT function on OSC2 (medium) 1001 = Internal RC oscillator, CLKOUT function on OSC2 1000 = Internal RC oscillator 0111 = External RC oscillator 0110 = External RC oscillator, CLKOUT function on OSC2 0101 = EC oscillator (high) 0100 = EC oscillator, CLKOUT function on OSC2 (high) 0011 = External RC oscillator, CLKOUT function on OSC2 0010 = HS oscillator 0001 = XT oscillator 0000 = LP oscillator
BORV<1:0>	CONFIG2L	Brown-out Reset Voltage bits 11 = V_{BOR} set to 1.9V 10 = V_{BOR} set to 2.2V 01 = V_{BOR} set to 2.7V 00 = V_{BOR} set to 3.0V
BOREN<1:0>	CONFIG2L	Brown-out Reset Enable bits 11 = Brown-out Reset enabled in hardware only (SBOREN is disabled) 10 = Brown-out Reset enabled in hardware only and disabled in Sleep mode SBOREN is disabled) 01 = Brown-out Reset enabled and controlled by software (SBOREN is enabled) 00 = Brown-out Reset disabled in hardware and software
$\overline{\text{PWRTEN}}$	CONFIG2L	Power-up Timer Enable bit 1 = PWRT disabled 0 = PWRT enabled

.

TABLE 6-3: PIC18F1XK22/LF1XK22 BIT DESCRIPTIONS (CONTINUED)

Bit Name	Configuration Words	Description
WDPS<3:0>	CONFIG2H	Watchdog Timer Postscaler Select bits 1111 = 1:32,768 1110 = 1:16,384 1101 = 1:8,192 1100 = 1:4,096 1011 = 1:2,048 1010 = 1:1,024 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)
MCLRE	CONFIG3H	$\overline{MCLR}$ Pin Enable bit 1 = $\overline{MCLR}$ pin enabled, RA3 input pin disabled 0 = RA3 input pin enabled, $\overline{MCLR}$ pin disabled
HFOFST	CONFIG3H	HFINTOSC Fast Start 1 = HFINTOSC output is not delayed 0 = HFINTOSC output is delayed until oscillator is stable (IOFS = 1)
ENHCPU	CONFIG4L	Enhanced CPU Enable bit 1 = Enhanced CPU enabled 0 = Enhanced CPU disabled
BBSIZ	CONFIG4L	Boot Block Size Select bit 1 = 2 kW Boot Block size for PIC18F14K22 (1 kW Boot Block size for PIC18F13K22) 0 = 1 kW Boot Block size for PIC18F14K22 (512 W Boot Block size for PIC18F13K22)
LVP	CONFIG4L	Low-Voltage Programming Enable bit 1 = Low-Voltage Programming enabled, RC3 is the PGM pin 0 = Low-Voltage Programming disabled, RC3 is an I/O pin
STVREN	CONFIG4L	Stack Overflow/Underflow Reset Enable bit 1 = Reset on stack overflow/underflow enabled 0 = Reset on stack overflow/underflow disabled
CP1	CONFIG5L	Code Protection bits (Block 1 code memory area) 1 = Block 1 is not code-protected 0 = Block 1 is code-protected
CP0	CONFIG5L	Code Protection bits (Block 0 code memory area) 1 = Block 0 is not code-protected 0 = Block 0 is code-protected
CPD	CONFIG5H	Code Protection bits (Data EEPROM) 1 = Data EEPROM is not code-protected 0 = Data EEPROM is code-protected

.

TABLE 6-3: PIC18F1XK22/LF1XK22 BIT DESCRIPTIONS (CONTINUED)

Bit Name	Configuration Words	Description
CPB	CONFIG5H	Code Protection bits (Boot Block memory area) 1 = Boot Block is not code-protected 0 = Boot Block is code-protected
WRT1	CONFIG6L	Write Protection bits (Block 1 code memory area) 1 = Block 1 is not write-protected 0 = Block 1 is write-protected
WRT0	CONFIG6L	Write Protection bits (Block 0 code memory area) 1 = Block 0 is not write-protected 0 = Block 0 is write-protected
WRTD	CONFIG6H	Write Protection bit (Data EEPROM) 1 = Data EEPROM is not write-protected 0 = Data EEPROM is write-protected
WRTB	CONFIG6H	Write Protection bit (Boot Block memory area) 1 = Boot Block is not write-protected 0 = Boot Block is write-protected
WRTC	CONFIG6H	Write Protection bit (Configuration registers) 1 = Configuration registers are not write-protected 0 = Configuration registers are write-protected
EBTR1	CONFIG7L	Table Read Protection bit (Block 1 code memory area) 1 = Block 1 is not protected from table reads executed in other blocks 0 = Block 1 is protected from table reads executed in other blocks
EBTR0	CONFIG7L	Table Read Protection bit (Block 0 code memory area) 1 = Block 0 is not protected from table reads executed in other blocks 0 = Block 0 is protected from table reads executed in other blocks
EBTRB	CONFIG7H	Table Read Protection bit (Boot Block memory area) 1 = Boot Block is not protected from table reads executed in other blocks 0 = Boot Block is protected from table reads executed in other blocks
DEV<10:3>	DEVID2	Device ID bits These bits are used with the DEV<2:0> bits in the DEVID1 register to identify part number.
DEV<2:0>	DEVID1	Device ID bits These bits are used with the DEV<10:3> bits in the DEVID2 register to identify part number.
REV<4:0>	DEVID1	Revision ID bits These bits are used to indicate the revision of the device.

.

7.0 EMBEDDING CONFIGURATION WORD INFORMATION IN THE HEX FILE

To allow portability of code, a PIC18F1XK22/LF1XK22 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.

7.1 Embedding Data EEPROM Information In the HEX File

To allow portability of code, a PIC18F1XK22/LF1XK22 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.

7.2 Checksum Computation

The checksum is calculated by summing the following:

- The contents of all code memory locations
- The Configuration Word, appropriately masked
- ID locations (Only if any portion of program memory is code-protected)

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

Code protection limits access to program memory by both external programmer (code-protect) and code execution (table read protect). The ID locations, when included in a code protected checksum, contain the checksum of an unprotected part. The unprotected checksum is distributed: one nibble per ID location. Each nibble is right justified.

Table 7-1 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 7-1: CHECKSUM COMPUTATION

Device	Code-Protect BBSIZ = 0	Checksum	Blank Value	0xAA at 0 and Max Address
PIC18F14K22/ PIC18LF14K22	None	SUM[0000:01FFF]+SUM[2000:3FFF]+ (CONFIG1L & 00h)+(CONFIG1H & FFh)+(CONFIG2L & 1Fh)+ (CONFIG2H & 1F)+(CONFIG3L & 00h)+(CONFIG3H & 88h)+ (CONFIG4L & CDh)+(CONFIG4H & 00h)+(CONFIG5L & 03h)+ (CONFIG5H & C0h)+(CONFIG6L & 03h)+(CONFIG6H & E0h)+ (CONFIG7L & 03h)+(CONFIG7H & 40h)	C35Bh	C2B1h
	Boot Block	SUM[0800:1FFF]+SUM[2000:3FFF]+ (CONFIG1L & 00h)+(CONFIG1H & FFh)+(CONFIG2L & 1Fh)+ (CONFIG2H & 1F)+(CONFIG3L & 00h)+ (CONFIG3H & 88h)+ (CONFIG4L & CDh)+(CONFIG4H & 00h)+(CONFIG5L & 03h)+ (CONFIG5H & C0h)+(CONFIG6L & 03h)+(CONFIG6H & E0h)+ (CONFIG7L & 03h)+(CONFIG7H & 40h)+SUM_ID	CB3Ah	CAE0h
	Boot/ Block 0	SUM[2000:3FFF]+ (CONFIG1L & 00h)+(CONFIG1H & FFh)+(CONFIG2L & 1Fh)+ (CONFIG2H & 1F)+(CONFIG3L & 00h)+(CONFIG3H & 88h)+ (CONFIG4L & CDh)+(CONFIG4H & 00h)+(CONFIG5L & 03h)+ (CONFIG5H & C0h)+(CONFIG6L & 03h)+(CONFIG6H & E0h)+ (CONFIG7L & 03h)+(CONFIG7H & 40h)+SUM_ID	E537h	E2DFh
	All	(CONFIG1L & 00h)+(CONFIG1H & FFh)+(CONFIG2L & 1Fh)+ (CONFIG2H & 1F)+(CONFIG3L & 00h)+(CONFIG3H & 88h)+ (CONFIG4L & CDh)+(CONFIG4H & 00h)+(CONFIG5L & 03h)+ (CONFIG5H & C0h)+(CONFIG6L & 03h)+(CONFIG6H & E0h)+ (CONFIG7L & 03h)+(CONFIG7H & 40h)+SUM_ID	0337h	0332h
PIC18F13K22/ PIC18LF13K22	None	SUM[0000:0FFF]+SUM[1000:1FFF]+ (CONFIG1L & 00h)+(CONFIG1H & FFh)+(CONFIG2L & 1Fh)+ (CONFIG2H & 1F)+(CONFIG3L & 00h)+(CONFIG3H & 88h)+ (CONFIG4L & CDh)+(CONFIG4H & 00h)+(CONFIG5L & 03h)+ (CONFIG5H & C0h)+(CONFIG6L & 03h)+(CONFIG6H & E0h)+ (CONFIG7L & 03h)+(CONFIG7H & 40h)	E35Bh	E2B1h
	Boot Block	SUM[0400:0FFF]+SUM[1000:1FFF]+ (CONFIG1L & 00h)+(CONFIG1H & FFh)+(CONFIG2L & 1Fh)+ (CONFIG2H & 1F)+(CONFIG3L & 00h)+ (CONFIG3H & 88h)+ (CONFIG4L & CDh)+(CONFIG4H & 00h)+(CONFIG5L & 03h)+ (CONFIG5H & C0h)+(CONFIG6L & 03h)+(CONFIG6H & E0h)+ (CONFIG7L & 03h)+(CONFIG7H & 40h)+SUM_ID	E73Ch	E6E2h
	Boot/ Block 0	SUM[1000:1FFF]+ (CONFIG1L & 00h)+(CONFIG1H & FFh)+(CONFIG2L & 1Fh)+ (CONFIG2H & 1F)+(CONFIG3L & 00h)+(CONFIG3H & 88h)+ (CONFIG4L & CDh)+(CONFIG4H & 00h)+(CONFIG5L & 03h)+ (CONFIG5H & C0h)+(CONFIG6L & 03h)+(CONFIG6H & E0h)+ (CONFIG7L & 03h)+(CONFIG7H & 40h)+SUM_ID	F539h	F2E1h
	All	(CONFIG1L & 00h)+(CONFIG1H & FFh)+(CONFIG2L & 1Fh)+ (CONFIG2H & 1F)+(CONFIG3L & 00h)+(CONFIG3H & 88h)+ (CONFIG4L & CDh)+(CONFIG4H & 00h)+(CONFIG5L & 03h)+ (CONFIG5H & C0h)+(CONFIG6L & 03h)+(CONFIG6H & E0h)+ (CONFIG7L & 03h)+(CONFIG7H & 40h)+SUM_ID	0339h	0334h

Legend:

Item		Description
CONFIGx	=	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

8.0 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}/RA3	8	9	V	
D110A	V_{IHL}	Low-Voltage Programming Voltage on $\overline{MCLR}$/V_{PP}/RA3	1.80	V_{DD}	V	
D111	V_{DD}	PIC18F1XK22 (includes Bulk Erase)	2.70	5.50	V	
		PIC18LF1XK22 (includes Bulk Erase)	2.70	3.60	V	
D112	I_{PP}	Programming Current on $\overline{MCLR}$/V_{PP}/RA3	—	5	mA	
D113	I_{DDP}	Supply Current During Programming	—	5	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} = 3.0 mA @ 2.7V
D090	V_{OH}	Output High Voltage	V_{DD} – 0.7	—	V	I_{OH} = -2.0 mA @ 2.7V
D012	C_{IO}	Capacitive Loading on I/O pin (PGD)	—	50	pF	To meet AC specifications
P1	T_R	$\overline{MCLR}$/V_{PP}/RA3 Rise Time to enter Program/Verify mode	—	1.0	μs	**(Note 1)**
P2	T_{PGC}	Serial Clock (PGC) Period	100	—	ns	V_{DD} = 3.6V
			1	—	μs	V_{DD} = 1.8V
P2A	T_{PGCL}	Serial Clock (PGC) Low Time	40	—	ns	V_{DD} = 3.6V
			400	—	ns	V_{DD} = 1.8V
P2B	T_{PGCH}	Serial Clock (PGC) High Time	40	—	ns	V_{DD} = 3.6V
			400	—	ns	V_{DD} = 1.8V
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	Externally Timed
P9A	T_{DLY5A}	PGC High Time	5		ms	Configuration Word programming time
P10	T_{DLY6}	PGC Low Time after Programming (high-voltage discharge time)	100	—	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	

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.

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

Standard Operating Conditions
Operating Temperature: 25°C is recommended

Param No.	Sym.	Characteristic	Min.	Max.	Units	Conditions
P12	THLD2	Input Data Hold Time from $\overline{MCLR}$/VPP/RA3 ↑	2	—	μs	
P12A	THLD2A	Input Data Hold Time from $\overline{MCLR}$/VPP/RA3 ↑	70	—	μs	PIC18F1XK22 Only. Refer to Figure 2.1.1.
P13	TSET2	VDD ↑ Setup Time to $\overline{MCLR}$/VPP/RA3 ↑	100	—	ns	
P13A	TSET2A	VDD ↑ Setup Time to $\overline{MCLR}$/VPP/RA3 ↑	70	—	μs	PIC18F1XK22 Only. Refer to Figure 2.1.1.
P14	TVALID	Data Out Valid from PGC ↑	10	—	ns	
P15	TSET3	PGM ↑ Setup Time to $\overline{MCLR}$/VPP/RA3 ↑	2	—	μs	
P16	TDLY8	Delay between Last PGC ↓ and $\overline{MCLR}$/VPP/RA3 ↓	0	—	s	
P17	THLD3	$\overline{MCLR}$/VPP/RA3 ↓ to VDD ↓	—	100	ns	
P18	THLD4	$\overline{MCLR}$/VPP/RA3 ↓ to PGM ↓	0	—	s	
P19	THIZ	Delay from PGC ↑ to PGD High-Z	3	10	nS	
P20	TPPDP	Hold time after VPP changes	5	—	μs	

Note 1: Do not allow excess time when transitioning $\overline{MCLR}$ 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.

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

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

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

Trademarks

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

Microchip received ISO/TS-16949:2002 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.

MICROCHIP

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

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

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

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

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

Detroit
Farmington Hills, MI
Tel: 248-538-2250
Fax: 248-538-2260

Kokomo
Kokomo, IN
Tel: 765-864-8360
Fax: 765-864-8387

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

Santa Clara
Santa Clara, CA
Tel: 408-961-6444
Fax: 408-961-6445

Toronto
Mississauga, Ontario,
Canada
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: 852-2401-1200
Fax: 852-2401-3431

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

China - Beijing
Tel: 86-10-8528-2100
Fax: 86-10-8528-2104

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

China - Hong Kong SAR
Tel: 852-2401-1200
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: 86-755-8203-2660
Fax: 86-755-8203-1760

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

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

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

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

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-4182-8400
Fax: 91-80-4182-8422

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

India - Pune
Tel: 91-20-2566-1512
Fax: 91-20-2566-1513

Japan - Yokohama
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

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: 886-3-572-9526
Fax: 886-3-572-6459

Taiwan - Kaohsiung
Tel: 886-7-536-4818
Fax: 886-7-536-4803

Taiwan - Taipei
Tel: 886-2-2500-6610
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 - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

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

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

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

UK - Wokingham
Tel: 44-118-921-5869
Fax: 44-118-921-5820

01/02/08

