Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

PIC18FXX20

Flash Microcontroller Programming Specification

1.0 DEVICE OVERVIEW

This document includes the programming specifications for the following devices:

- PIC18F6520
- PIC18F6620
- PIC18F6720
- PIC18F8520
- PIC18F8620
- PIC18F8720

2.0 PROGRAMMING OVERVIEW OF THE PIC18FXX20

PIC18FXX20 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 PIC18FXX20 devices in all package types.

2.1 Hardware Requirements

In high voltage ICSP mode, the PIC18FXX20 requires two programmable power supplies: one for VDD and one for MCLR/VPP. Both supplies should have a minimum resolution of 0.25V. Refer to Section 6.0 for additional hardware parameters.

2.1.1 LOW VOLTAGE ICSP PROGRAMMING

In low voltage ICSP mode, the PIC18FXX20 can be programmed using a VDD source in the operating range. This only means that 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 for additional hardware parameters.

2.2 Pin Diagrams

The pin diagrams for the PIC18FXX20 family are shown in Figure 2-1. 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): PIC18FXX20

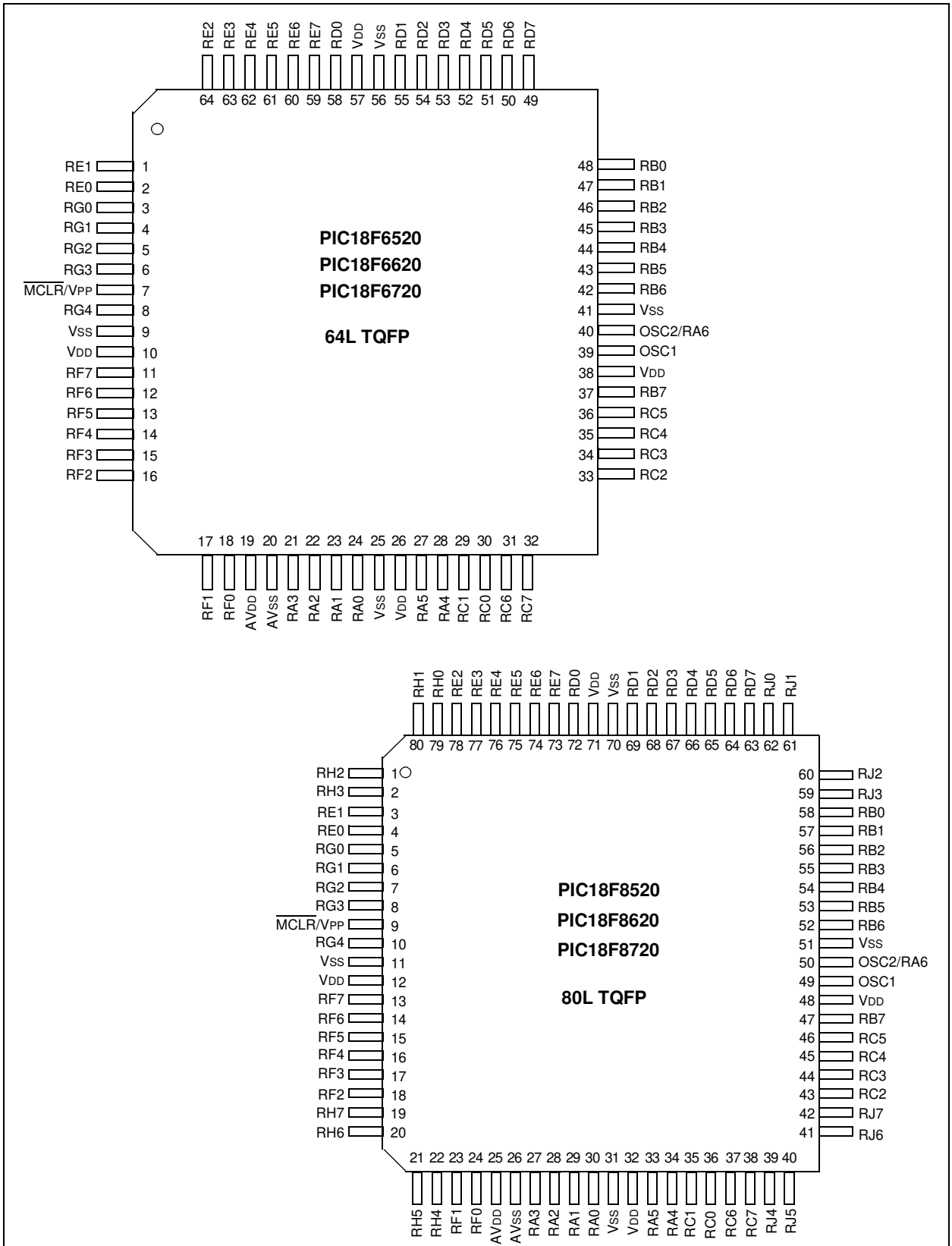
Pin Name	During Programming		
	Pin Name	Pin Type	Pin Description
MCLR/VPP/RA5	VPP	P	Programming Enable
VDD(2)	VDD	P	Power Supply
VSS(2)	VSS	P	Ground
AVDD	AVDD	P	Analog Power Supply
AVSS	AVSS	P	Analog Ground
RB5	PGM	I	Low Voltage ICSP™ Input when LVP Configuration bit equals '1' (1)
RB6	SCLK	I	Serial Clock
RB7	SDATA	I/O	Serial Data

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

Note 1: See Section 5.3 for more detail.

2: All power supply and ground must be connected.

FIGURE 2-1: PIC18FXX20 FAMILY PIN DIAGRAMS



Note: Not all multiplexed pin definitions are shown. Refer to the appropriate data sheet for complete pin descriptions.

2.3 Memory Map

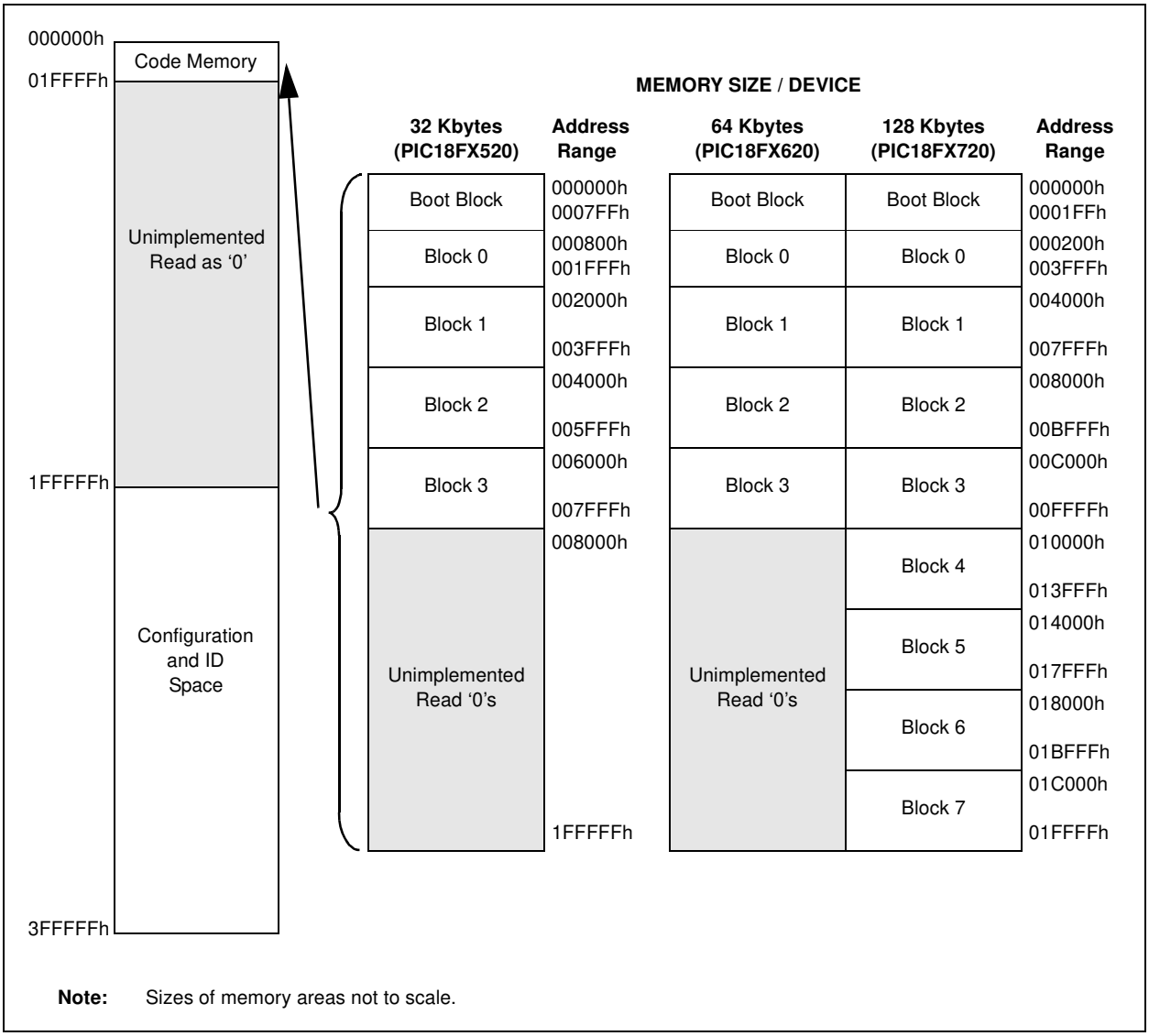
The code memory space extends from 0000h to 1FFFFh (128 Kbytes) in eight 16-Kbyte blocks. Addresses 0000h through 01FFh, however, define a “Boot Block” region that is treated separately from Block 1. All of these blocks define code protection boundaries within the code memory space.

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

TABLE 2-2: IMPLEMENTATION OF CODE MEMORY

Device	Code Memory Size (Bytes)
PIC18F6520	000000h - 007FFFh (32K)
PIC18F8520	
PIC18F6620	000000h - 00FFFFh (64K)
PIC18F8620	
PIC18F6720	000000h - 01FFFFh (128K)
PIC18F8720	

FIGURE 2-2: MEMORY MAP AND THE CODE MEMORY SPACE FOR PIC18FXX20 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 2-3.

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

2.3.1 MEMORY ADDRESS POINTER

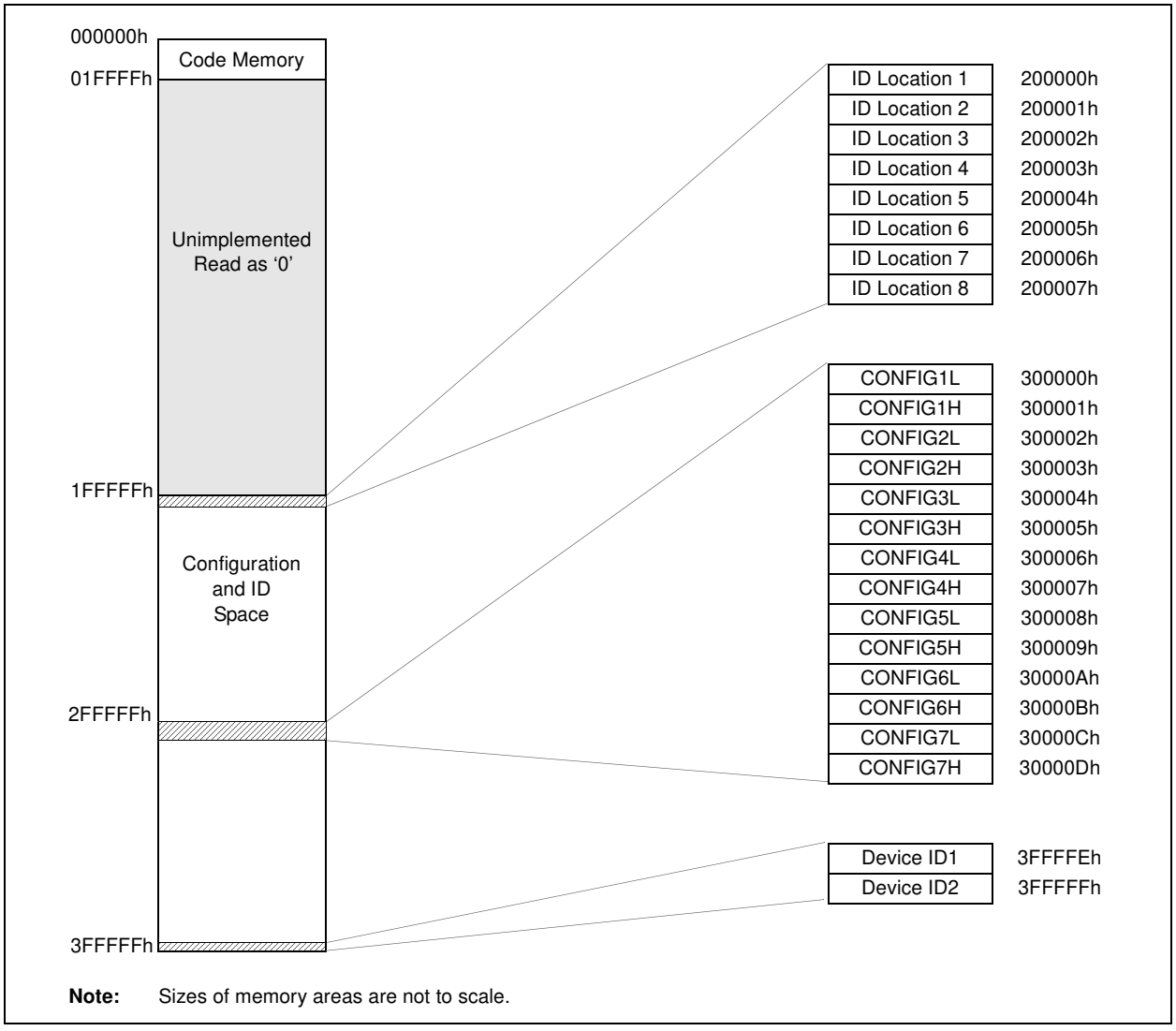
Memory in the address space 0000000h to 3FFFFFh is addressed via the Table Pointer, which is comprised of three pointer registers:

- 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 2-3: CONFIGURATION AND ID LOCATIONS FOR PIC18FXX20 DEVICES



Note: Sizes of memory areas are not to scale.

2.4 High Level Overview of the Programming Process

Figure 2-4 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.

2.5 Entering High Voltage ICSP Program/Verify Mode

The high voltage ICSP Program/Verify mode is entered by holding SCLK and SDATA low and then raising $\overline{MCLR}/V_{PP}$ to V_{IHH} (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), the low voltage ICSP mode is enabled. Low voltage ICSP Program/Verify mode is entered by holding SCLK and SDATA low, placing a logic high on PGM, and then raising $\overline{MCLR}/V_{PP}$ to V_{IH}. 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 2-4: HIGH LEVEL PROGRAMMING FLOW

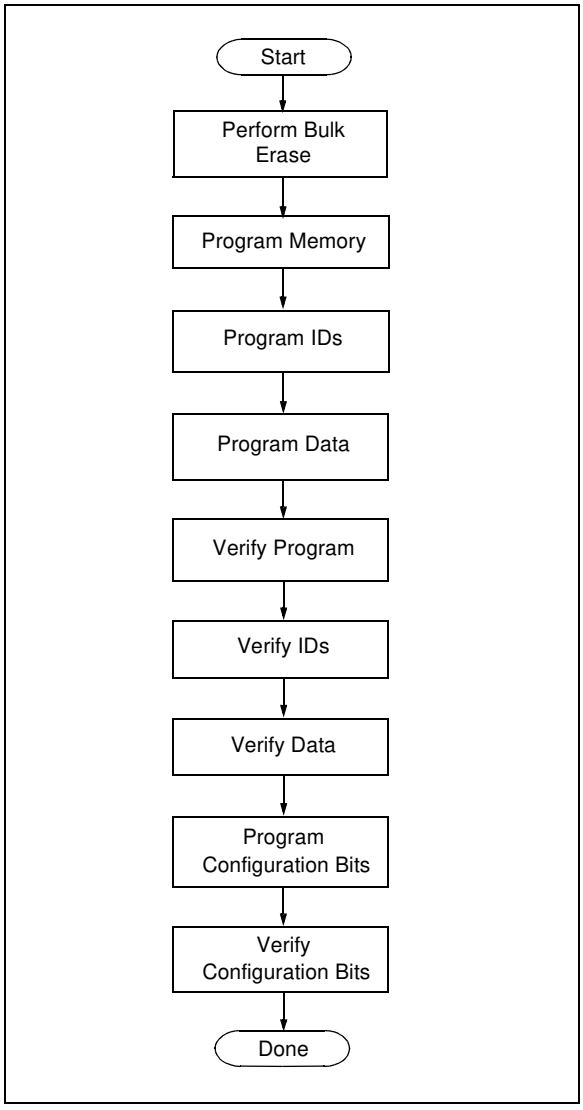


FIGURE 2-5: ENTERING HIGH VOLTAGE PROGRAM/ VERIFY MODE

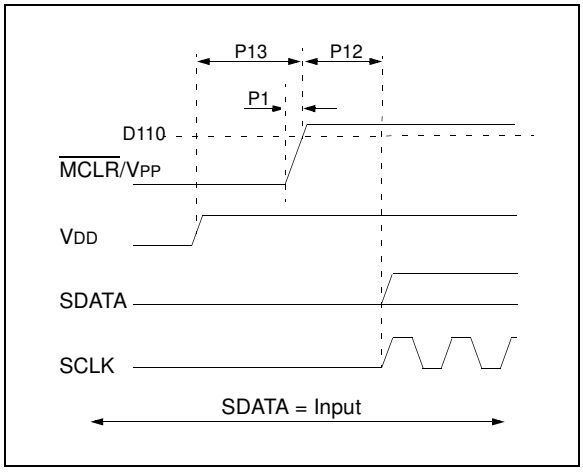
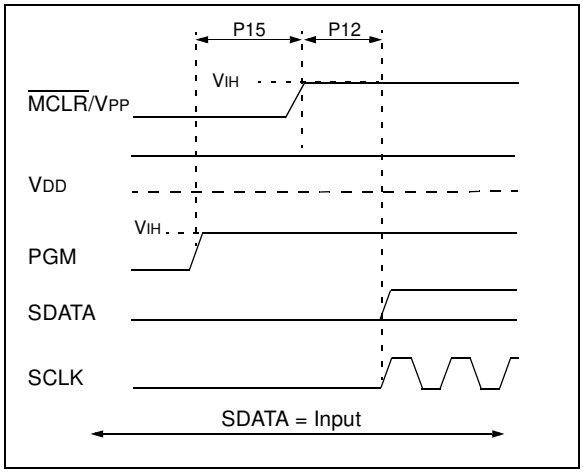


FIGURE 2-6: ENTERING LOW VOLTAGE PROGRAM/ VERIFY MODE



2.6 Serial Program/Verify Operation

The SCLK pin is used as a clock input pin and the SDATA pin is used for entering command bits and data input/output during serial operation. Commands and data are transmitted on the rising edge of SCLK, latched on the falling edge of SCLK, 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, SCLK 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 Figure 2-4. The 4-bit command is shown MSb first. The command operand, or “Data Payload”, is shown <MSB><LSB>. Figure 2-7 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.

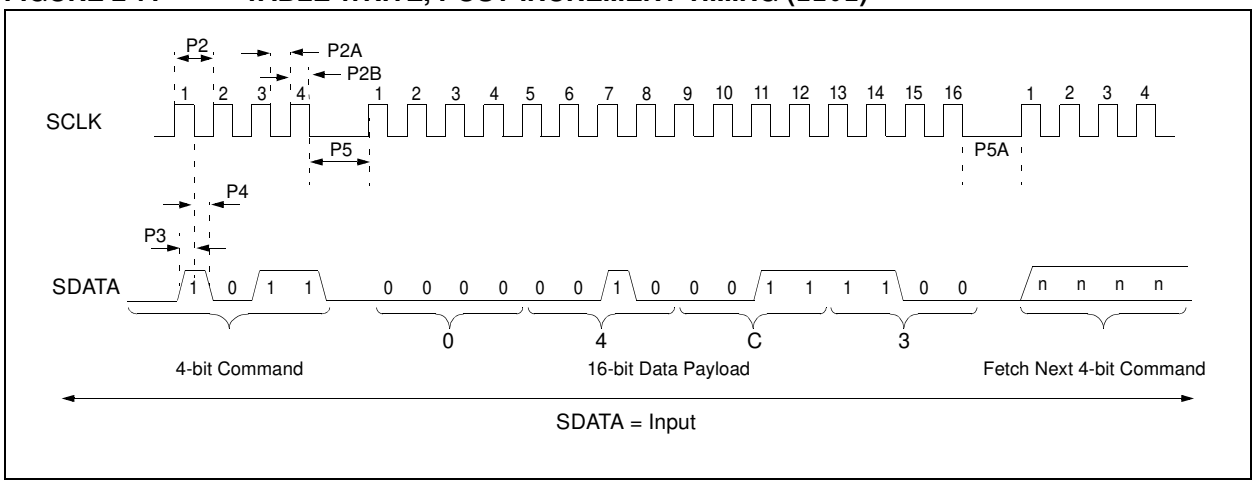
TABLE 2-3: COMMANDS FOR PROGRAMMING

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 2-7: 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 SCLK after the NOP command), serial execution will cease until the erase completes (parameter P11). During this time, SCLK may continue to toggle, but SDATA must be held low.

The code sequence to erase the entire device is shown in Figure 3-1 and the flowchart is shown in Figure 3-2.

Note: A bulk erase is the only way to reprogram code protect bits from an on-state to an off-state.
Non-code protect bits are not returned to default settings by a bulk erase. These bits should be programmed to ones, as outlined in Section 3.6, "Configuration Bits Programming".

FIGURE 3-1: 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 SDATA low until erase completes.

FIGURE 3-2: BULK ERASE FLOW

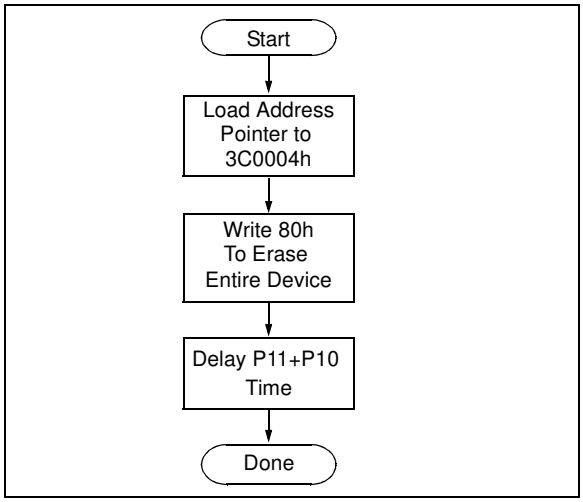
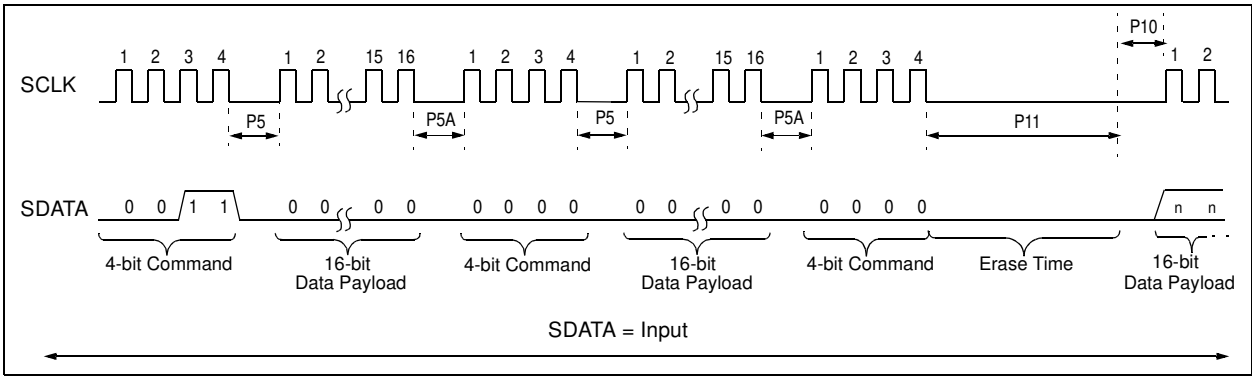


FIGURE 3-3: BULK ERASE TIMING



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 and write ones to the array.

3.1.2 ICSP MULTI-PANEL SINGLE ROW ERASE

Irrespective of whether high or low voltage ICSP is used, it is possible to erase 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 3-6). 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 SCLK. After a “Start Programming” command is issued (4-bit, ‘1111’), a NOP is issued, where the 4th SCLK is held high for the duration of the programming time, P9.

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

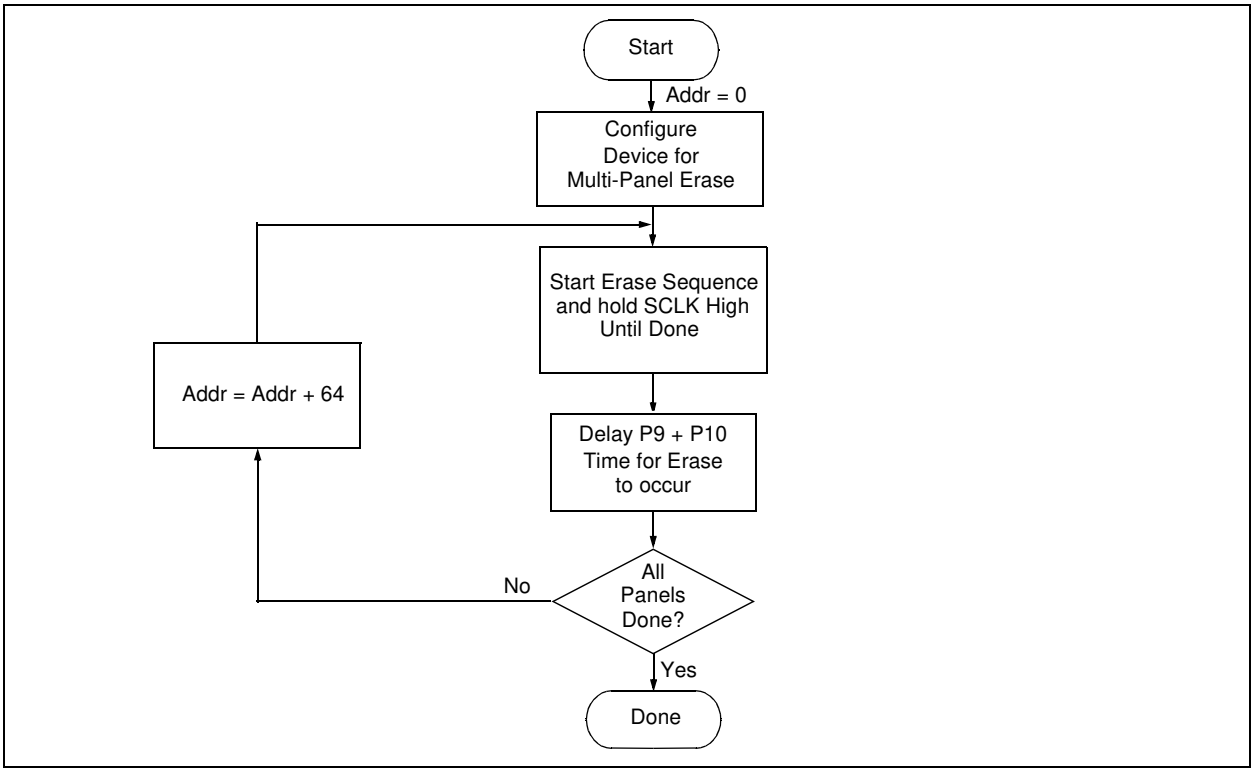
The code sequence to program a PIC18FXX20 device is shown in Table 3-2. The flowchart shown in Figure 3-4 depicts the logic necessary to completely erase a PIC18FXX20 device. The timing diagram that details the “Start Programming” command, and parameters P9 and P10 is shown in Figure 3-7.

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 3-2: 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	86 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 and enable erase.		
0000	8E A6	BSF EECON1, EEPGD
0000	9C A6	BCF EECON1, CFGS
0000	88 A6	BSF EECON1, FREE
0000	6A F8	CLRF TBLPTRU
0000	6A F7	CLRF TBLPTRH
0000	6A F6	CLRF TBLPTRL
Step 4: Erase single row of all panels at an offset.		
1111	<DummyLSB> <DummyMSB>	Write 2 dummy bytes and start programming.
0000	00 00	NOP - hold SCLK high for time P9.
Step 5: Repeat step 4, with Address Pointer incremented by 64 until all panels are erased.		

FIGURE 3-4: 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 2-2) 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 128-Kbyte device (16 panels with an 8-byte buffer per panel), 128 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 3-5). Multi-Panel Write mode is enabled by appropriately configuring the Programming Control register located at 3C0006h.

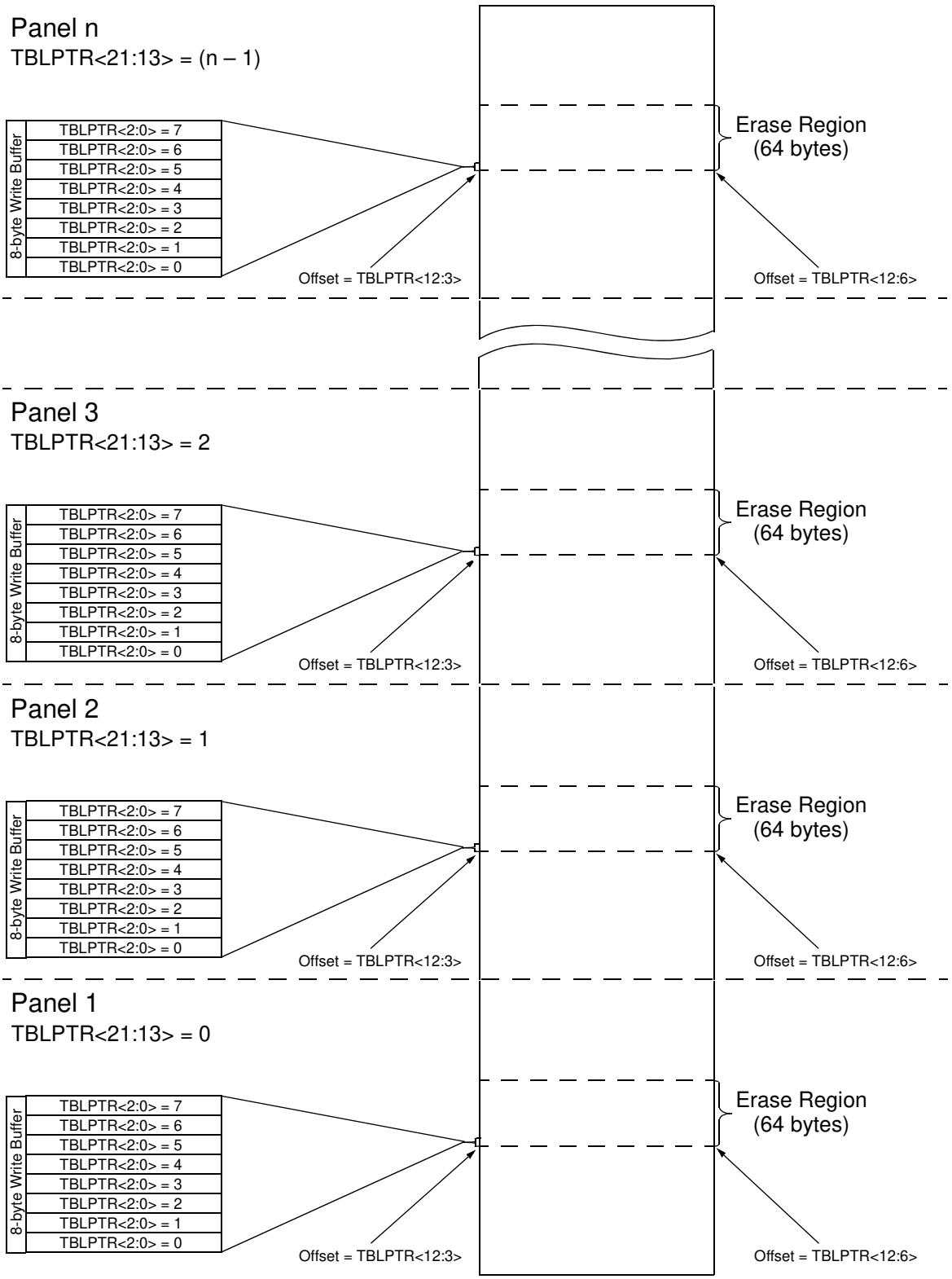
The programming duration is externally timed and is controlled by SCLK. After a "Start Programming" command is issued (4-bit command, '1111'), a NOP is issued, where the 4th SCLK is held high for the duration of the programming time, P9.

After SCLK is brought low, the programming sequence is terminated. SCLK 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 PIC18FXX20 device is shown in Figure 3-3. The flowchart shown in Figure 3-6 depicts the logic necessary to completely write a PIC18FXX20 device. The timing diagram that details the "Start Programming" command, and parameters P9 and P10, is shown in Figure 3-7.

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 3-5: ERASE AND WRITE BOUNDARIES



Note: TBLPTR = TBLPTRU:TBLPTRH:TBLPTRL.

TABLE 3-3: 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	86 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 SCLK 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 3-6: PROGRAM CODE MEMORY FLOW

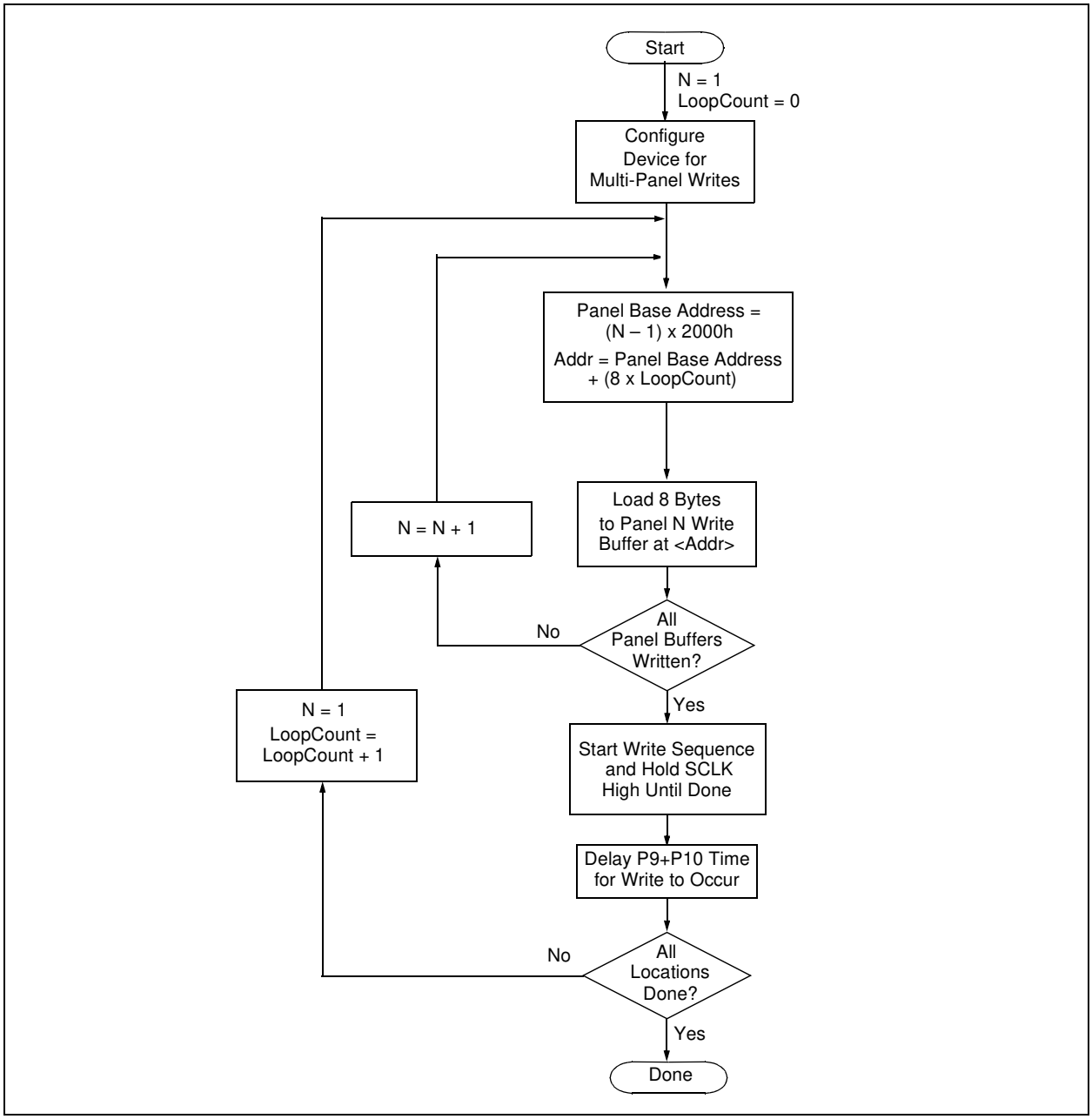
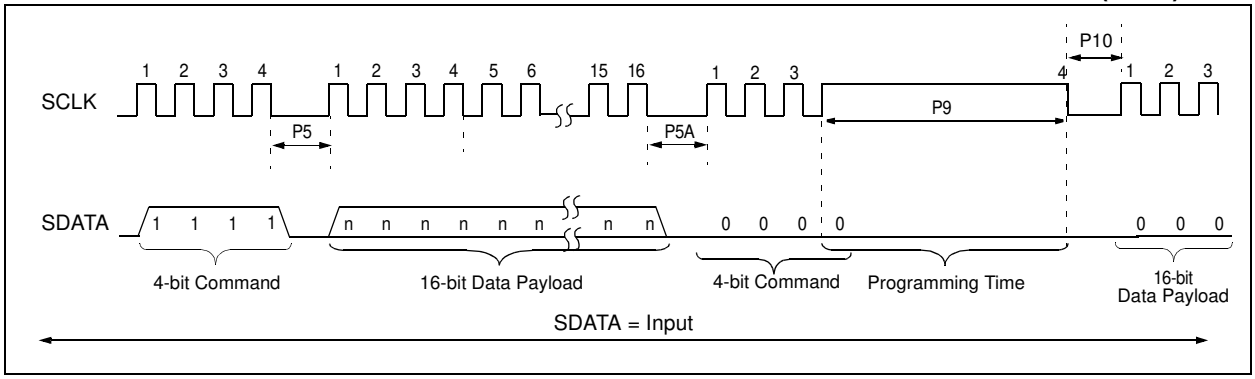


FIGURE 3-7: TABLE WRITE AND START PROGRAMMING INSTRUCTION TIMING (1111)



3.2.1 SINGLE PANEL PROGRAMMING

The programming example presented in Section 3.2 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). 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), 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 SCLK, after the WR bit is set. After the erase sequence terminates, SCLK must still be held low for the time specified by parameter #P10 to allow high voltage discharge of the memory array.

TABLE 3-4: 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 set up 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 and then disable writes.		
0000	94 A6	BCF EECON1, WREN
Step 9: Load write buffer for panel. The correct panel will be selected based on the Table Pointer.		
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 SCLK high for time P9
To continue writing data, repeat step 8, where the Address Pointer is incremented by 8 at each iteration of the loop.		

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 SCLK after the WR bit is set. It ends when the WR bit is cleared by hardware.

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

FIGURE 3-8: PROGRAM DATA FLOW

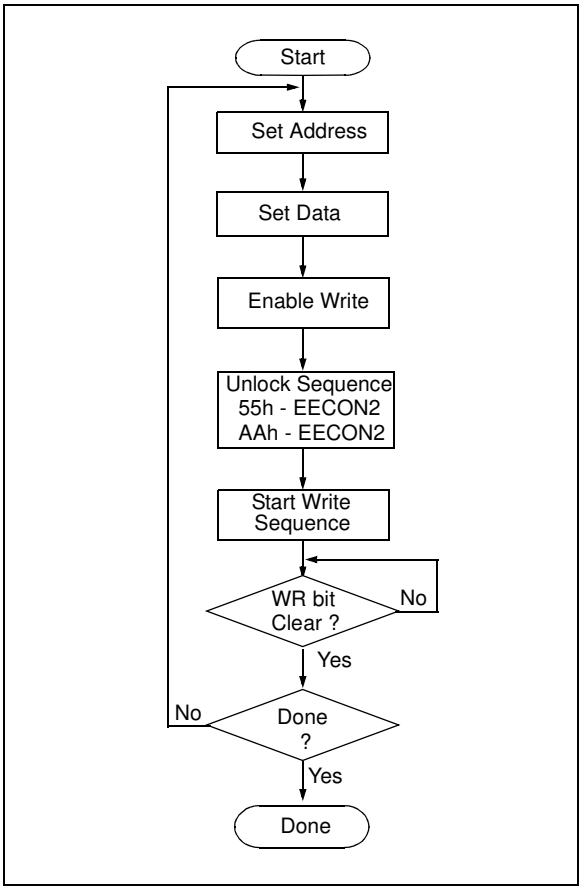


FIGURE 3-9: DATA EEPROM WRITE TIMING

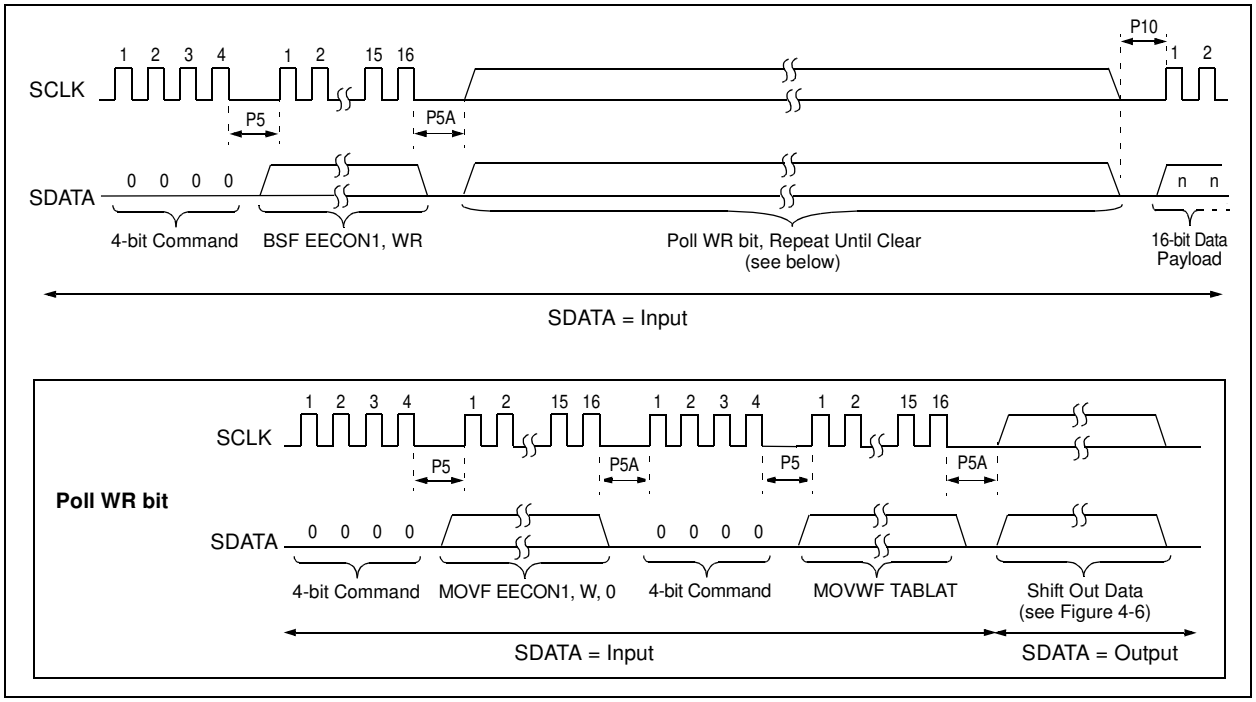


TABLE 3-5: PROGRAMMING DATA MEMORY

4-Bit Command	Data Payload	Core Instruction
Step 1: Direct access to data EEPROM.		
0000	9E A6	BCF EECON1, EEPGD
0000	9C A6	BCF EECON1, CFGS
Step 2: Set the data EEPROM Address Pointer.		
0000	0E <Addr>	MOVLW <Addr>
0000	6E A9	MOVWF EEADR
0000	OE <AddrH>	MOVLW <AddrH>
0000	6E AA	MOVWF EEADRH
Step 3: Load the data to be written.		
0000	0E <Data>	MOVLW <Data>
0000	6E A8	MOVWF EEDATA
Step 4: Enable memory writes.		
0000	84 A6	BSF EECON1, WREN
Step 5: Perform required sequence.		
0000	0E 55	MOVLW 0X55
0000	6E A7	MOVWF EECON2
0000	0E AA	MOVLW 0XAA
0000	6E A7	MOVWF EECON2
Step 6: Initiate write.		
0000	82 A6	BSF EECON1, WR
Step 7: Poll WR bit, repeat until the bit is clear.		
0000	50 A6	MOVF EECON1, W, 0
0000	6E F5	MOVWF TABLAT
0010	<LSB><MSB>	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.

Figure 3-6 demonstrates the code sequence required to write the ID locations.

TABLE 3-6: WRITE ID 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
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: Load write buffer. Panel will be automatically determined by address.		
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	<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 SCLK high for time P9

In order to modify the ID locations, refer to the methodology described in Section 3.2.2, “Modifying Code Memory”. As with code memory, the ID locations must be erased before modified.

3.5 Boot Block Programming

The Boot Block segment is programmed in exactly the same manner as the ID locations (see Section 3.4). Multi-panel writes must be disabled so that only addresses in the range 0000h to 01FFh will be written.

The code sequence detailed in Figure 3-6 should be used, except that the address data used in “Step 2” will be in the range 000000h to 0001FFh.

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 Figure 3-7.

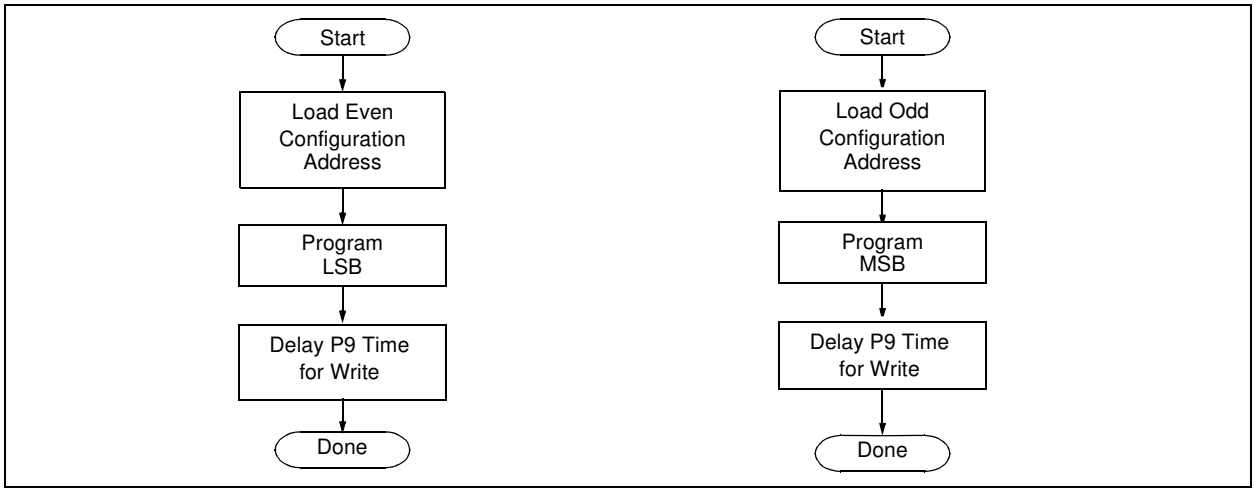
TABLE 3-7: SET ADDRESS POINTER TO CONFIGURATION LOCATION

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: Position the program counter(1).		
0000	EF 00	GOTO 100000h
0000	F8 00	
Step 3(2): Set Table Pointer for config 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	<LSB><MSB ignored>	Load 2 bytes and start programming
0000	00 00	NOP - hold SCLK high for time P9
0000	2A F6	INCF TBLPTRL
1111	<LSB ignored><MSB>	Load 2 bytes and start programming
0000	00 00	NOP - hold SCLK high for time P9

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 3-10: 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) are loaded into the Table Latch and then serially output on SDATA.

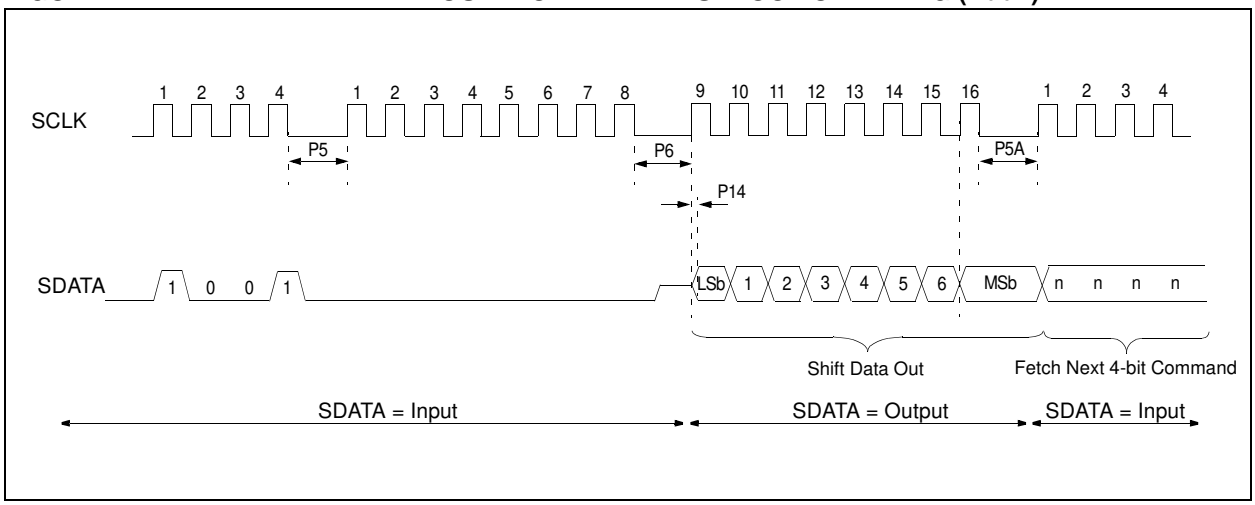
The 4-bit command is shifted in LSb first. The Table Read is executed during the next 8 clocks, then shifted out on SDATA during the last 8 clocks, LSb to MSb. A delay of P6 must be introduced after the falling edge of the 8th SCLK of the operand to allow SDATA to transition from an input to an output.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

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

TABLE 4-1: READ CODE MEMORY SEQUENCE

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

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

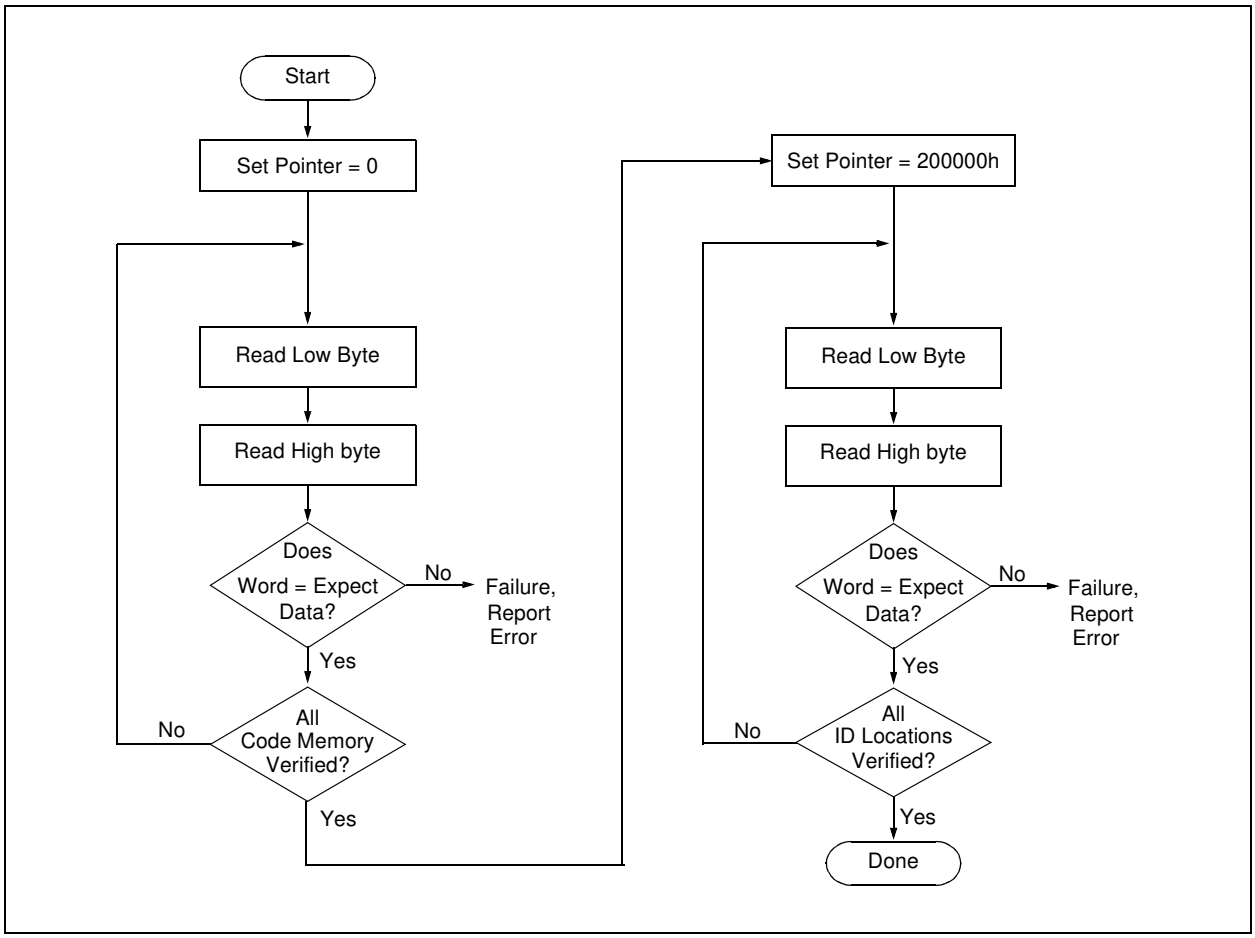


4.2 Verify Code Memory and ID locations

The verify step involves reading back the code memory space and comparing 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 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 32-Kbyte device, for example, a post-increment read of address 7FFFh will wrap the Table Pointer back to 0000h, rather than point to unimplemented address 8000h.

FIGURE 4-2: VERIFY CODE MEMORY FLOW



4.3 Verify Configuration Bits

A configuration address may be read and output on SDATA 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 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 SDATA 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 SCLK of the operand to allow SDATA to transition from an input to an output. During this time, SCLK must be held low (see Figure 4-4).

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

FIGURE 4-3: READ DATA EEPROM FLOW

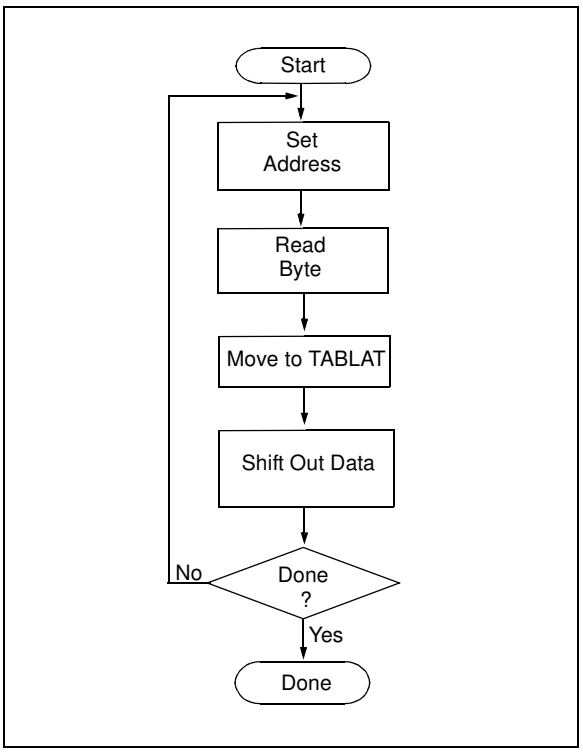
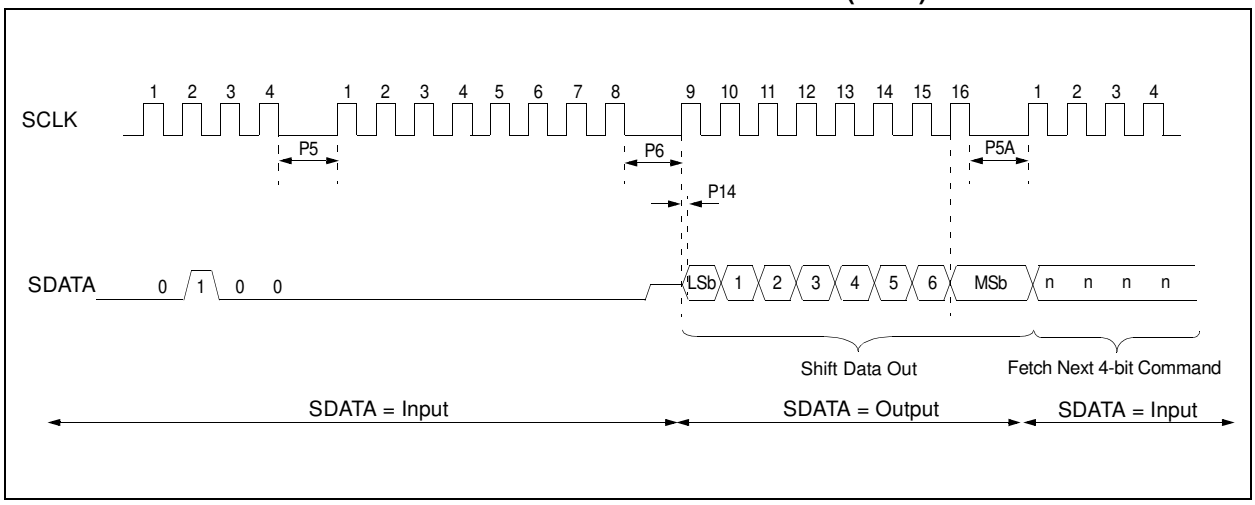


TABLE 4-2: READ DATA EEPROM MEMORY

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 SDATA 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 for implementation details of reading data EEPROM.

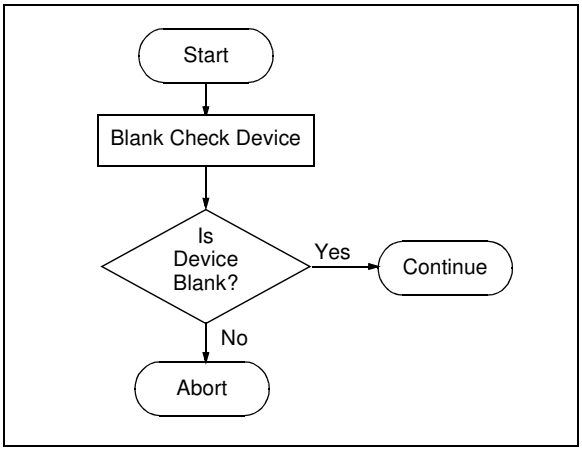
4.6 Blank Check

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 PIC18FXX20 devices.

Given that "Blank Checking" is merely code and data EEPROM verification with FFh expect data, refer to Section 4.4 and Section 4.2 for implementation details.

FIGURE 4-5: BLANK CHECK FLOW



5.0 CONFIGURATION WORD

The PIC18FXX20 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.

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 NOP.

5.2 Device ID Word

The device ID word for the PIC18FXX20 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.

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 $\overline{MCLR}/V_{PP}$ 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 $\overline{MCLR}/V_{PP}$ pin.

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

TABLE 5-1: DEVICE ID VALUES

Device	Device ID Value	
	DEVID2	DEVID1
PIC18F6520	0Bh	001x xxxx
PIC18F6620	06h	011x xxxx
PIC18F6720	06h	001x xxxx
PIC18F8520	0Bh	000x xxxx
PIC18F8620	06h	010x xxxx
PIC18F8720	06h	000x xxxx

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