

Flash Microcontroller Programming Specification

1.0 DEVICE OVERVIEW

This document includes the programming specifications for the following devices:

- PIC18F1220
- PIC18F1320
- PIC18F2220
- PIC18F2320
- PIC18F4220
- PIC18F4320

2.0 PROGRAMMING OVERVIEW

These devices can be programmed using the high-voltage In-Circuit Serial Programming™ (ICSP™) method, or the low-voltage ICSP method, both while in the user's 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 these devices in all package types.

2.1 Hardware Requirements

In High-Voltage ICSP mode, these devices require 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 “AC/DC Characteristics”** for additional hardware parameters.

2.1.1 LOW-VOLTAGE ICSP PROGRAMMING

In Low-Voltage ICSP mode, these devices 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 “AC/DC Characteristics”** for additional hardware parameters.

2.1.2 VDD POWER SUPPLY

It is recommended that the power supply decoupling capacitance be added at the programmer socket. Capacitance in the range of 0.1 µF to 10 µF should be connected from VDD to VSS, and located as close to the programming socket as possible.

2.2 Pin Diagrams

The programming pin descriptions for these devices are shown in Table 2-1 and the pin diagrams are shown in Figure 2-1 through Figure 2-6. The pin descriptions of these diagrams do not represent the complete functionality of the device types. Refer to the appropriate device data sheet for complete pin descriptions.

TABLE 2-1: PIN DESCRIPTIONS (DURING PROGRAMMING)

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

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

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

2: RA5 is only available on the PIC18F1X20.

PIC18FX220/X320

FIGURE 2-1: PIC18F1X20 18-PIN PDIP, SOIC

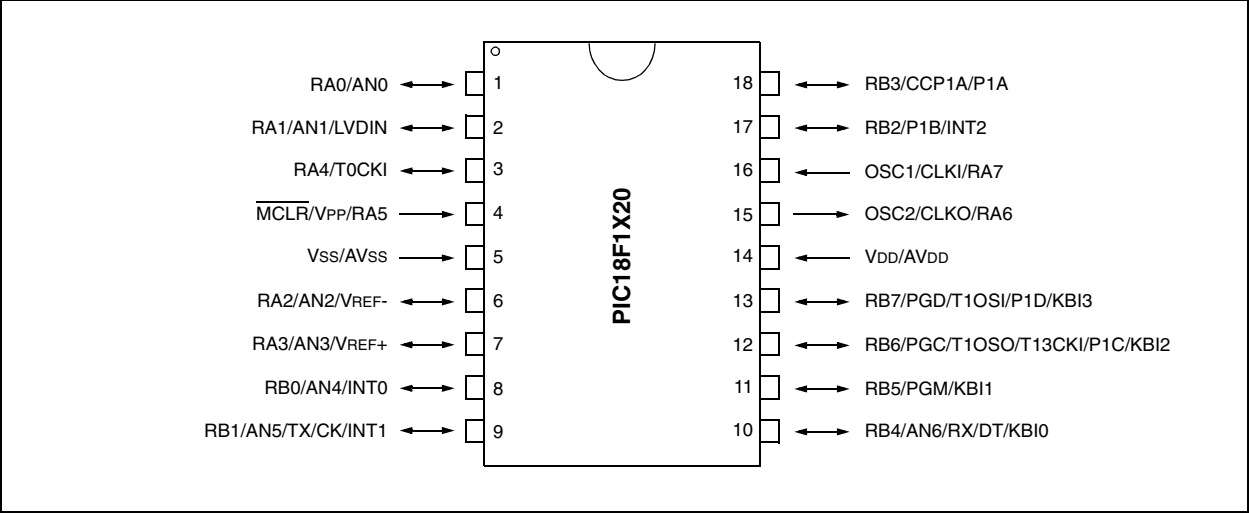


FIGURE 2-2: PIC18F1X20 20-PIN SSOP

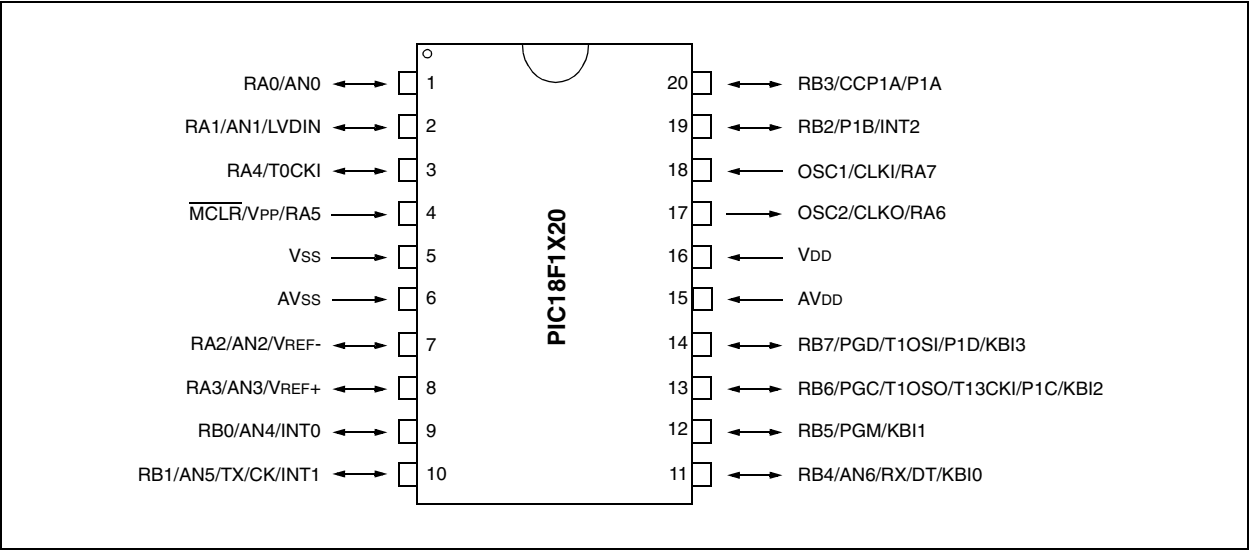


FIGURE 2-3: PIC18F1X20 28-PIN QFN

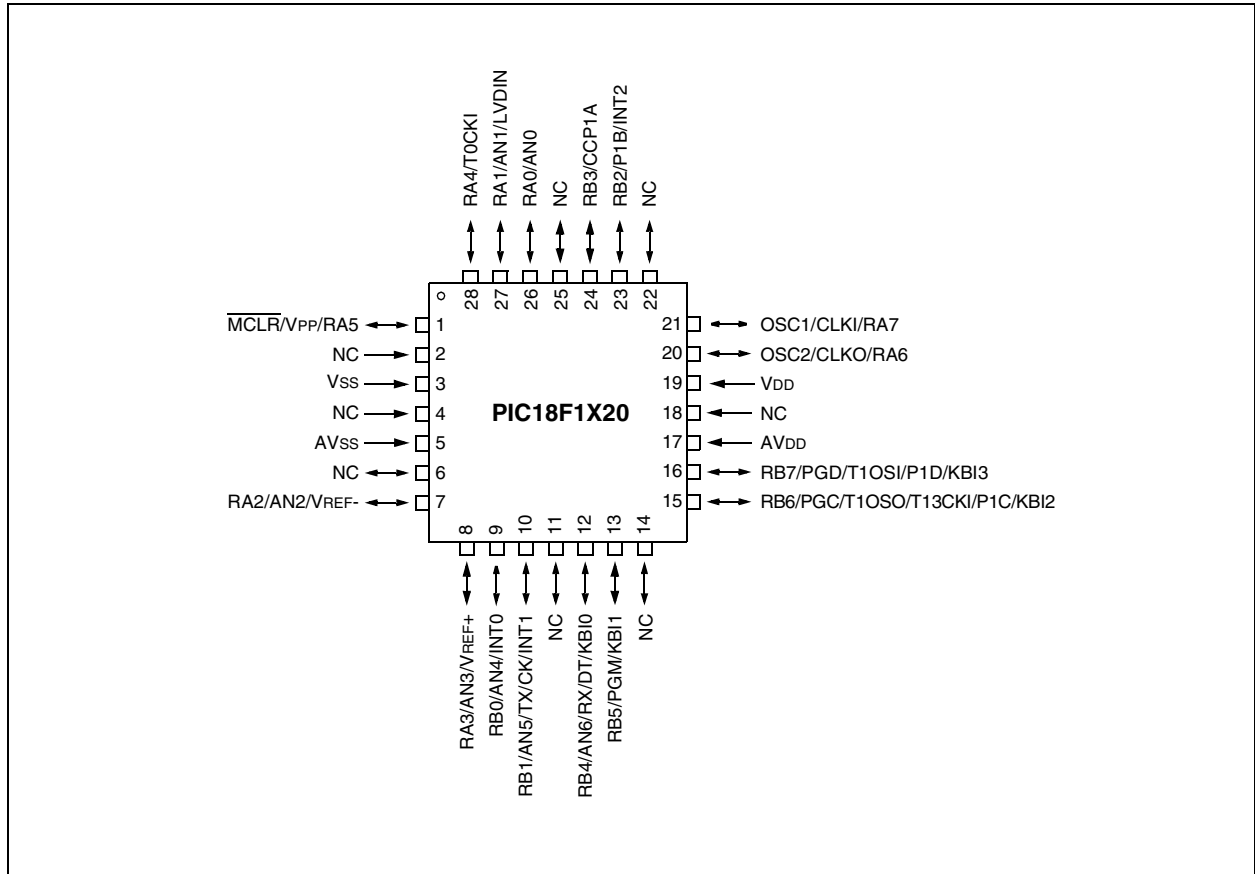
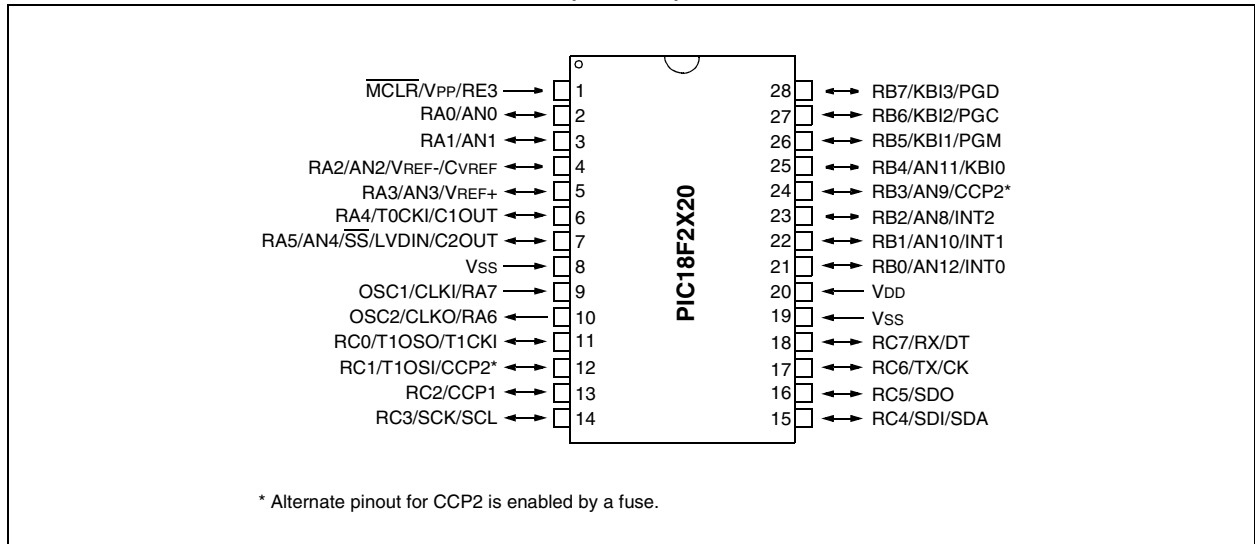


FIGURE 2-4: PIC18F2X20 28-PIN SDIP (300 MIL), SOIC



PIC18FX220/X320

FIGURE 2-5: PIC18F4X20 40-PIN PDIP (600 MIL)

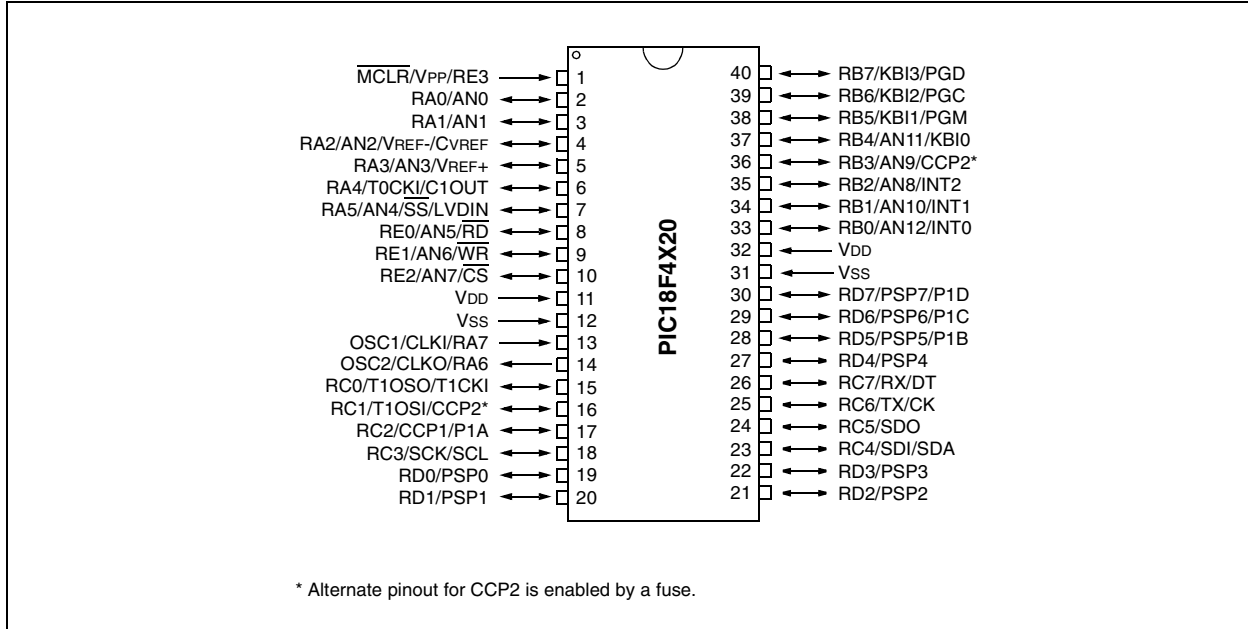
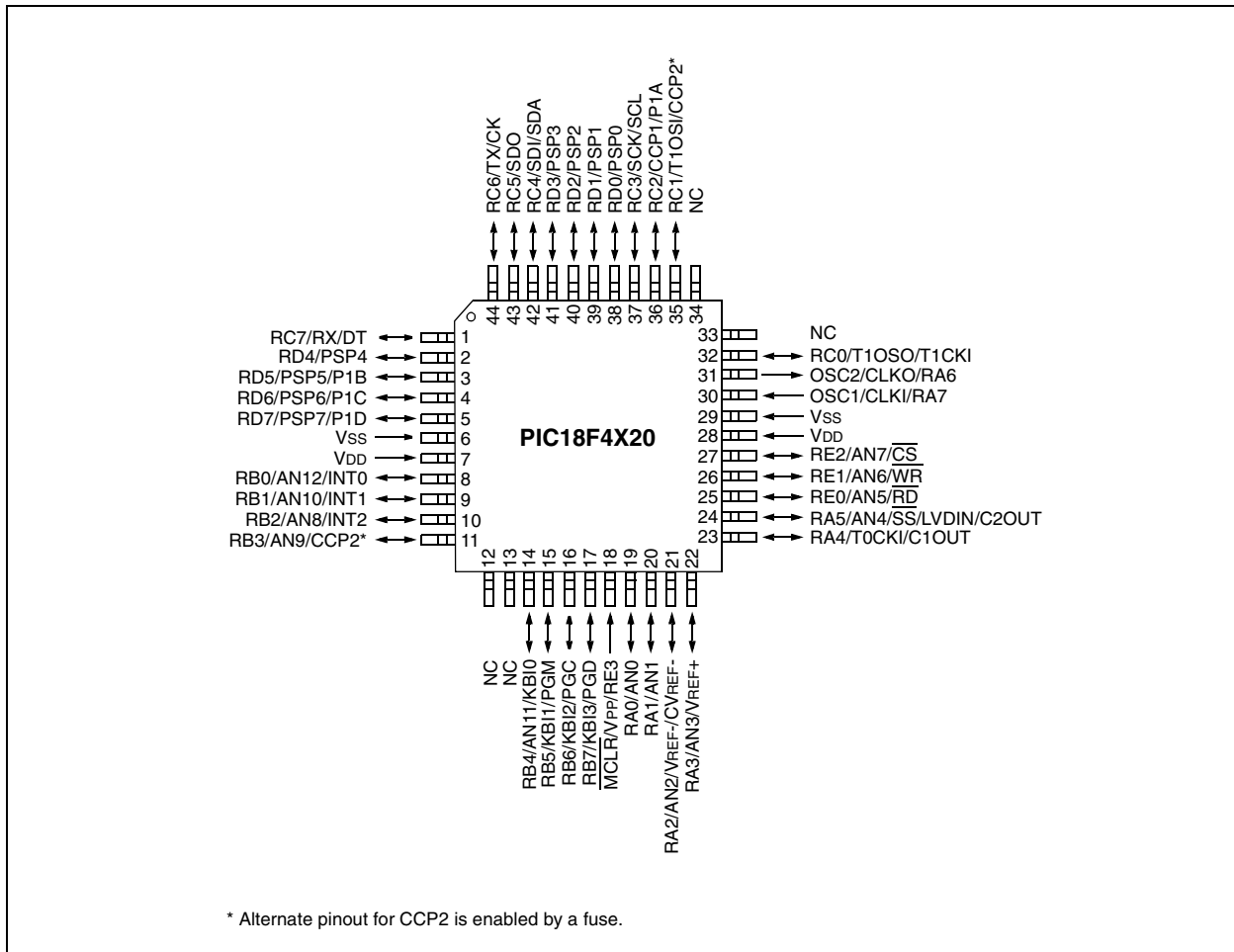


FIGURE 2-6: PIC18F4X20 44-PIN TQFP



2.3 Memory Map

The code memory space extends from 0000h to 1FFFh (8 Kbytes) in a single 8-Kbyte panel. Addresses 0000h through 01FFh, however, define a “Boot Block” region that is treated separately from Panel 1. All code memory is on-chip.

A user 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.

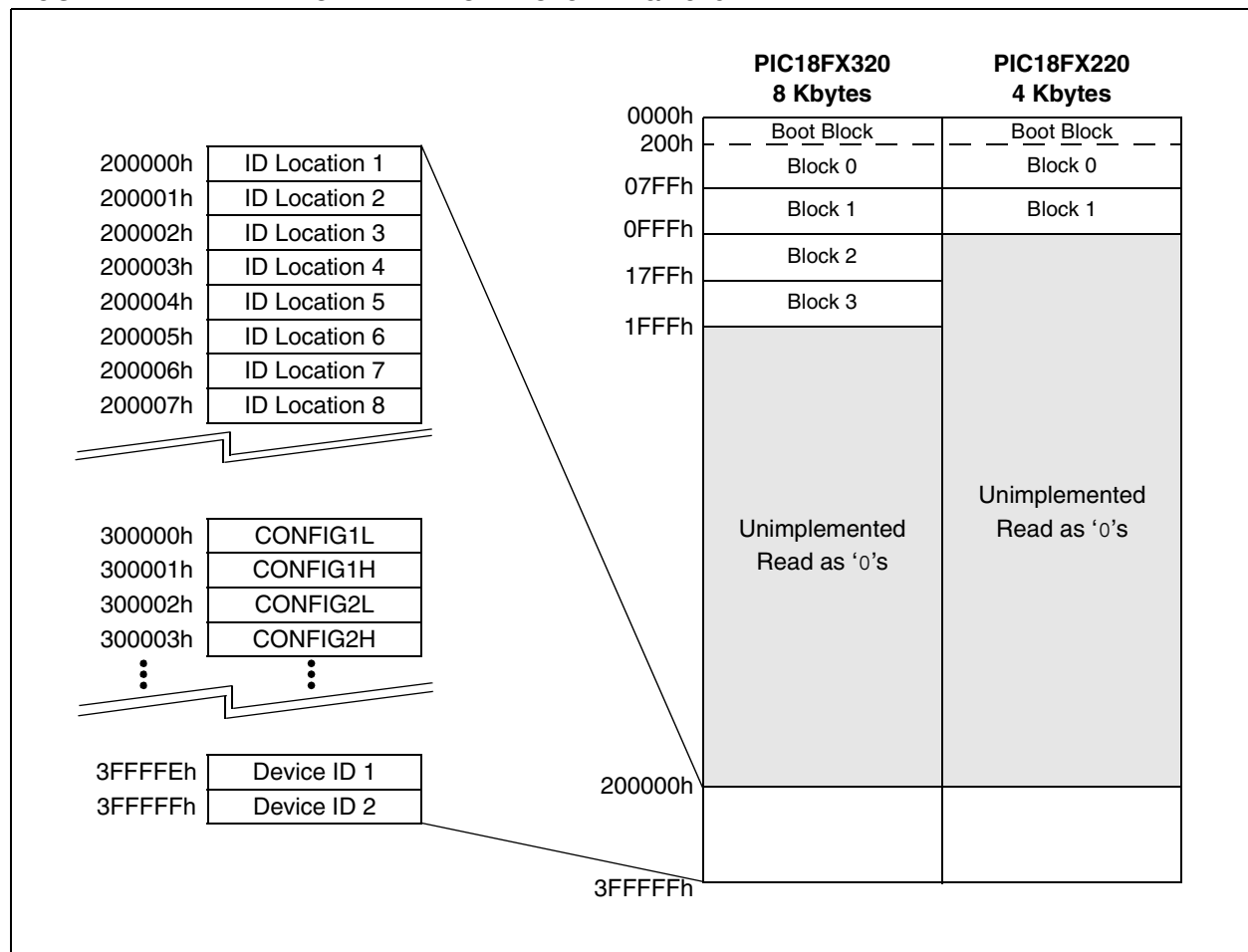
Locations 300001h through 30000Dh are reserved for the Configuration Words. These Words may be set to select various device options and are described in **Section 5.0 “Configuration Word”**. These Configuration Words read out normally, even after code protection.

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

TABLE 2-2: IMPLEMENTATION OF CODE MEMORY

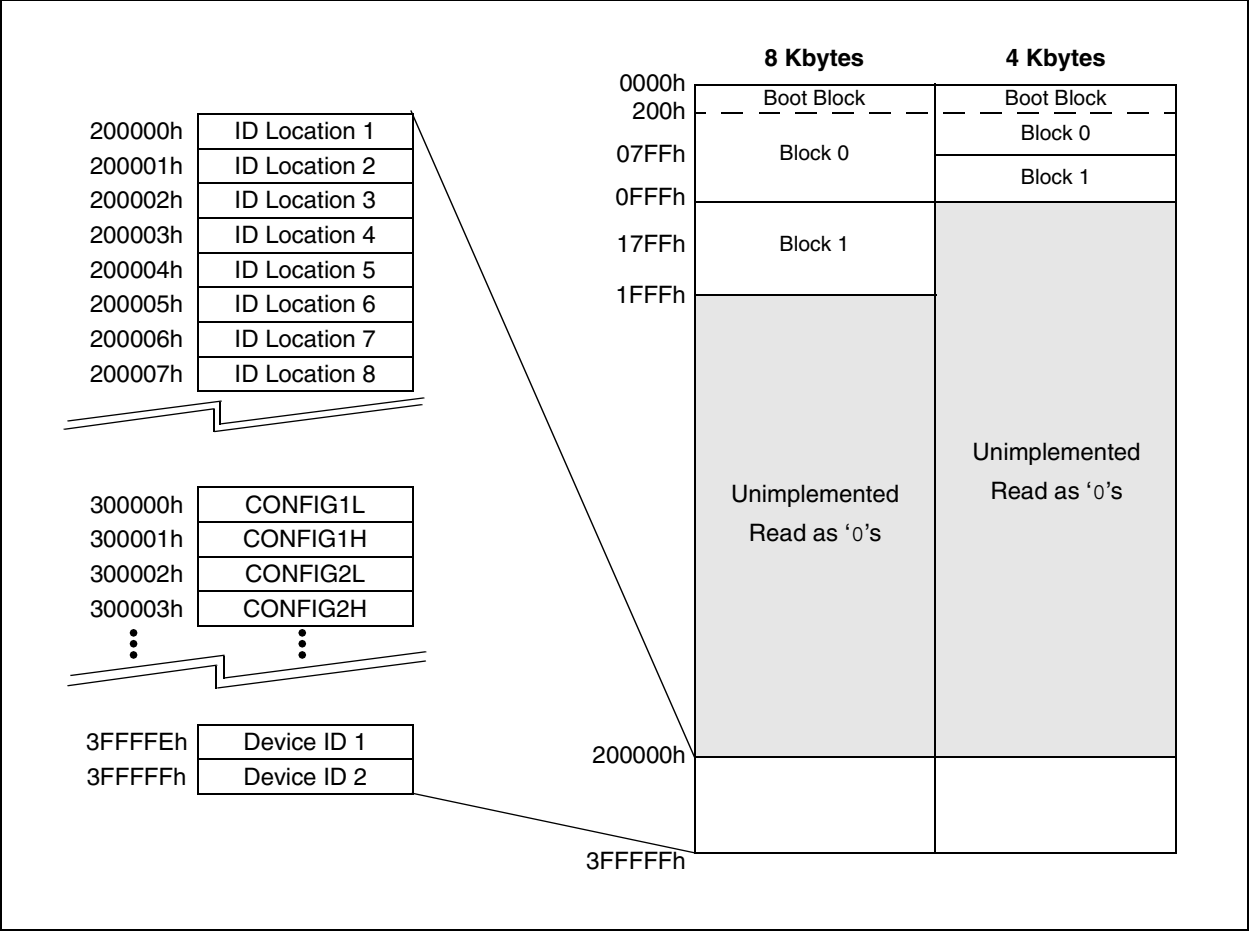
Device	Code Memory Size (Bytes)	Data EEPROM Size (Bytes)
PIC18F1220	0000h-0FFFh (4K)	000-0FFh (256)
PIC18F2220	0000h-0FFFh (4K)	000-0FFh (256)
PIC18F4220	0000h-0FFFh (4K)	000-0FFh (256)
PIC18F1320	0000h-1FFFh (8K)	000-0FFh (256)
PIC18F2320	0000h-1FFFh (8K)	000-0FFh (256)
PIC18F4320	0000h-1FFFh (8K)	000-0FFh (256)

FIGURE 2-7: MEMORY MAP FOR PIC18FX220/X320



PIC18FX220/X320

FIGURE 2-8: MEMORY MAP FOR PIC18F1X20



2.3.1 MEMORY ADDRESS POINTER

Memory in the address space, 000000h to 3FFFFFh, is addressed via the Table Pointer, which is comprised of three pointer registers:

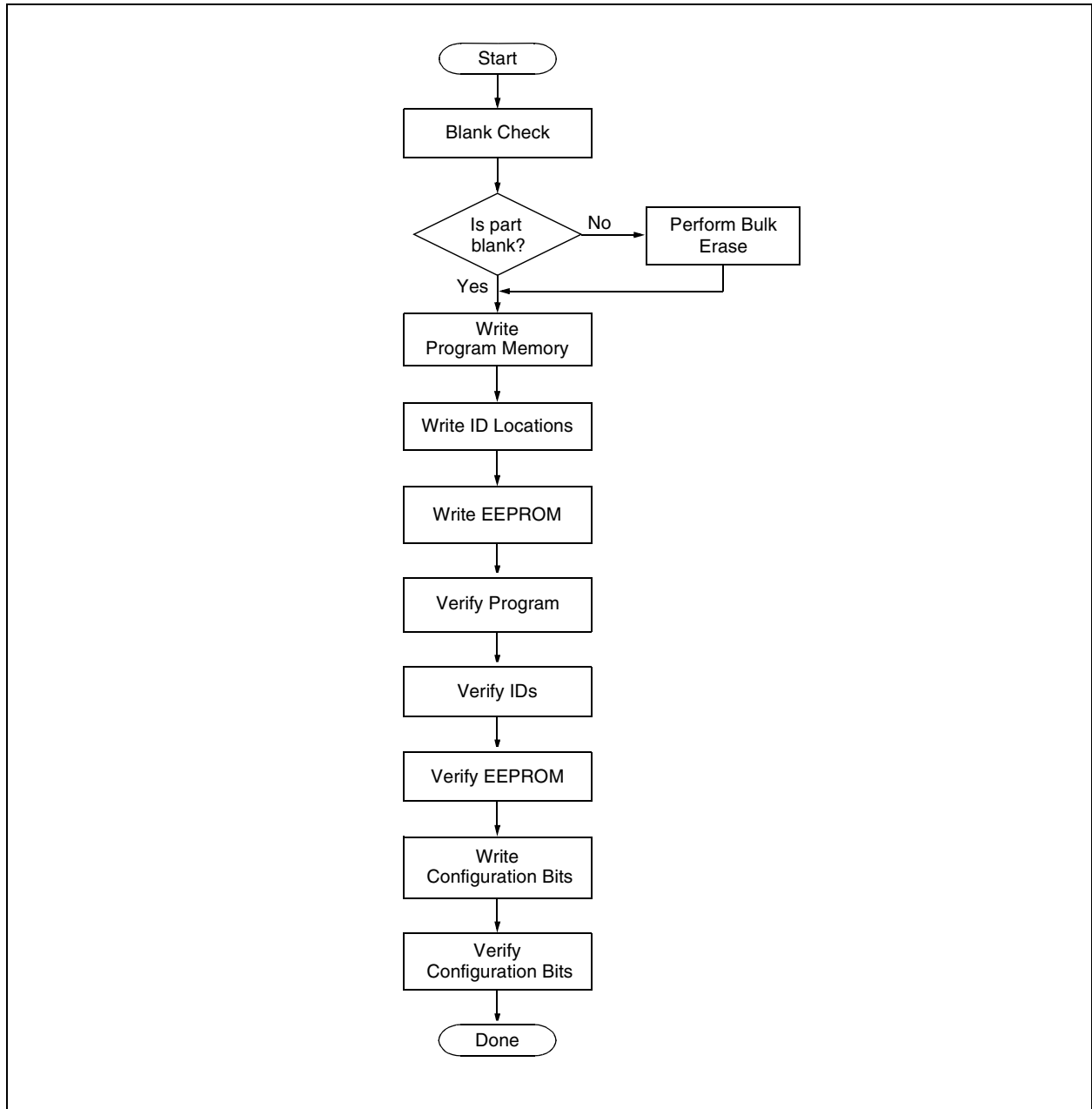
- TBLPTRU at address 0FF8h
- TBLPTRH at address 0FF7h
- TBLPTRL at address 0FF6h

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

2.4 High-Level Overview of the Programming Process

Figure 2-9 shows the high-level overview of the programming process. The device is first checked to see if it is blank; if it is not, a Bulk Erase is performed. Next, the program memory, ID locations and data EEPROM are written. These memories are then verified to ensure that programming was successful. If no errors are detected, the Configuration bits are then written and verified.

FIGURE 2-9: HIGH-LEVEL PROGRAMMING FLOW



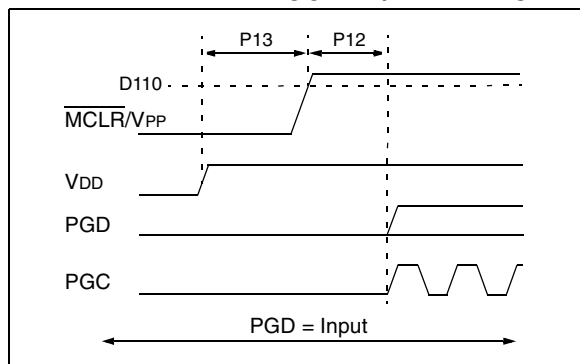
PIC18FX220/X320

2.5 Entering High-Voltage ICSP Program/Verify Mode

The High-Voltage ICSP Program/Verify mode is entered by holding PGC and PGD low, and then raising MCLR/VPP to VIH (high voltage). Once in this mode, the code memory, data EEPROM, ID locations and Configuration bits can be accessed and written in serial fashion.

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

FIGURE 2-10: ENTERING HIGH-VOLTAGE PROGRAM/VERIFY MODE

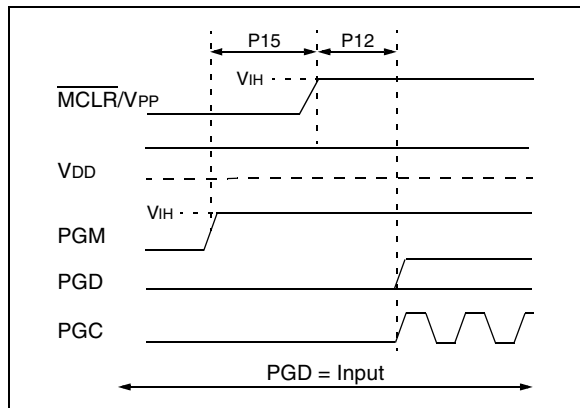


2.6 Entering Low-Voltage ICSP Program/Verify Mode

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

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

FIGURE 2-11: ENTERING LOW-VOLTAGE PROGRAM/VERIFY MODE



2.7 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 sent Least Significant bit (LSb) first.

2.7.1 4-BIT COMMANDS

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

TABLE 2-3: COMMANDS FOR PROGRAMMING

Description	4-Bit Command
Core Instruction (Shift in 16-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

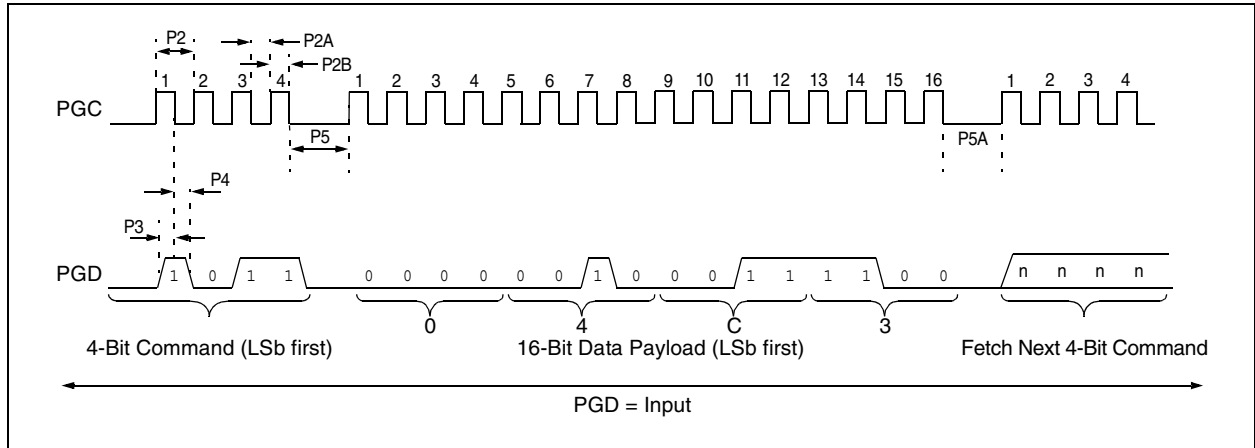
Depending on the 4-bit command, the 16-bit operand represents 16 bits or 8 bits of data.

Throughout this specification, commands and data are presented as illustrated in Table 2-4. The 4-bit command is shown MSb first. The command operand or "Data Payload" is shown <MSB:LSB>. Figure 2-12 demonstrates how to serially present a 20-bit command/operand to the device.

TABLE 2-4: SAMPLE COMMAND SEQUENCE

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

FIGURE 2-12: TABLE WRITE, POST-INCREMENT TIMING (1101)



2.7.2 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.

If the instruction is a 1-word, 1-cycle instruction, it will be executed while the next command is clocked in.

If the instruction is a 2-word, 2-cycle instruction, another core instruction command is required with the second word of the instruction. The instruction will complete when a third 4-bit command has been loaded.

PIC18FX220/X320

3.0 DEVICE PROGRAMMING

3.1 Blank Check

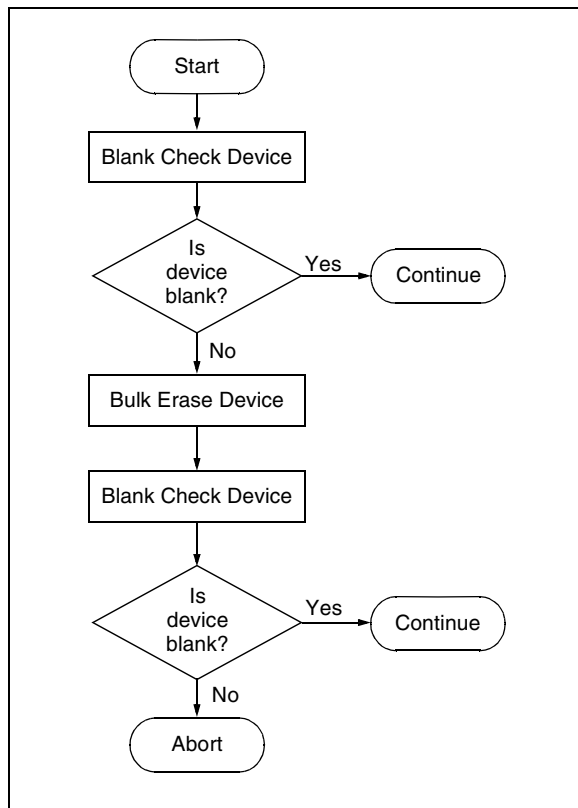
The term “Blank Check” means to verify that the device has no programmed memory cells. All memories must be verified: code memory, data EEPROM, ID locations and Configuration bits. The Device ID registers (3FFFFEh:3FFFFFh) should be ignored.

A “blank” or “erased” memory cell will read as ‘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 as ‘0’ (programmed). Refer to Table 5-2 for blank configuration expected data for the various devices.

If it is determined that the device is not blank, then the device should be Bulk Erased (see **Section 3.2 “High-Voltage ICSP Bulk Erase”**) before any attempt to program is made.

Given that “Blank Checking” is merely code and data EEPROM verification, with FFh as the expected data, refer to **Section 4.1 “Read Data EEPROM Memory”** and **Section 4.3 “Verify Code Memory and ID Locations”** for implementation details.

FIGURE 3-1: BLANK CHECK FLOW



3.2 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 0	88h
Erase Block 1	89h
Erase Block 2	8Ah
Erase Block 3	8Bh

The actual Bulk Erase function is a self-timed operation. Once the erase has started (falling edge of the 4th PGC after the write command), serial execution will cease until the erase completes (parameter P11). During this time, PGC may continue to toggle, but PGD must be held low.

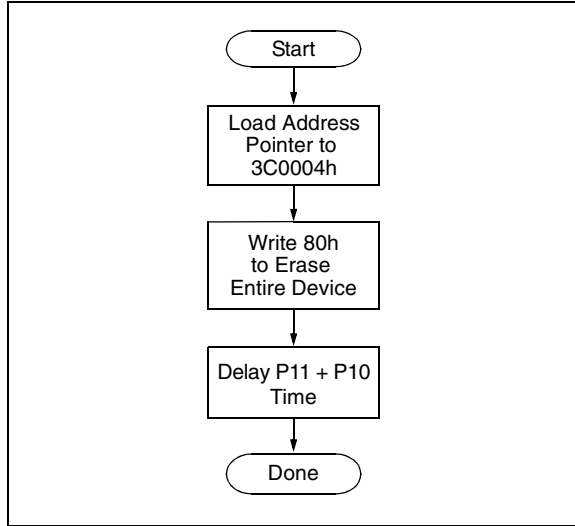
The code sequence to erase the entire device is shown in Table 3-2 and the flowchart is shown in Figure 3-2.

Note: A Bulk Erase is the only way to reprogram code-protect bits from an ON state to an OFF state.

TABLE 3-2: BULK ERASE COMMAND SEQUENCE

4-Bit Command	Data Payload	Core Instruction
0000	0E 3C	MOVLW 3Ch
0000	6E F8	MOVWF TBLPTRU
0000	0E 00	MOVLW 00h
0000	6E F7	MOVWF TBLPTRH
0000	0E 04	MOVLW 04h
0000	6E F6	MOVWF TBLPTRL
1100	00 80	Write 80h TO 3C0004h to erase entire device.
0000	00 00	NOP
0000	00 00	Hold PGD low until erase completes.

FIGURE 3-2: BULK ERASE FLOW



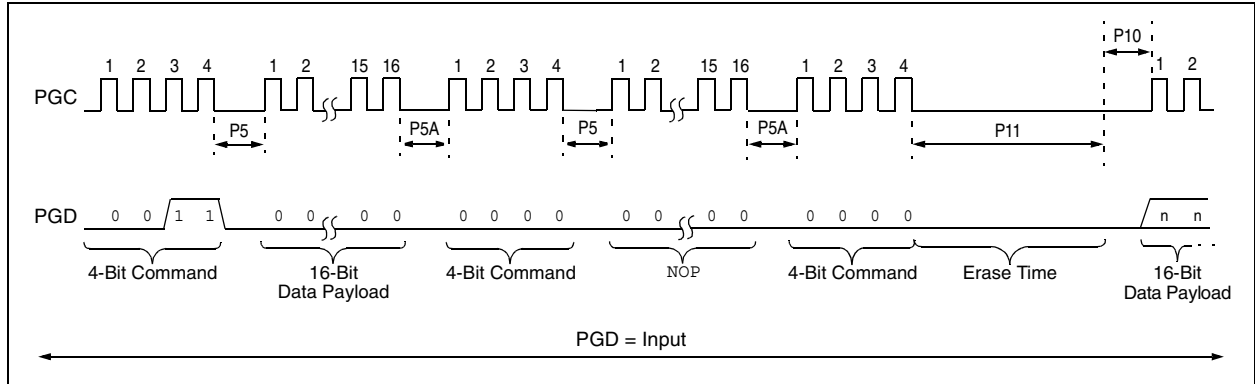
3.2.1 LOW-VOLTAGE ICSP BULK ERASE

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 **Section 3.3.1 “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 3.4 “Data EEPROM Programming”** and write zeros to the array.

FIGURE 3-3: BULK ERASE TIMING



3.3 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 2-7 and Figure 2-8) 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.

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 (see Figure 3-6).

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 device is shown in Table 3-3. The flowchart shown in Figure 3-5 depicts the logic necessary to completely write a device.

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

PIC18FX220/X320

FIGURE 3-4: ERASE AND WRITE BOUNDARIES

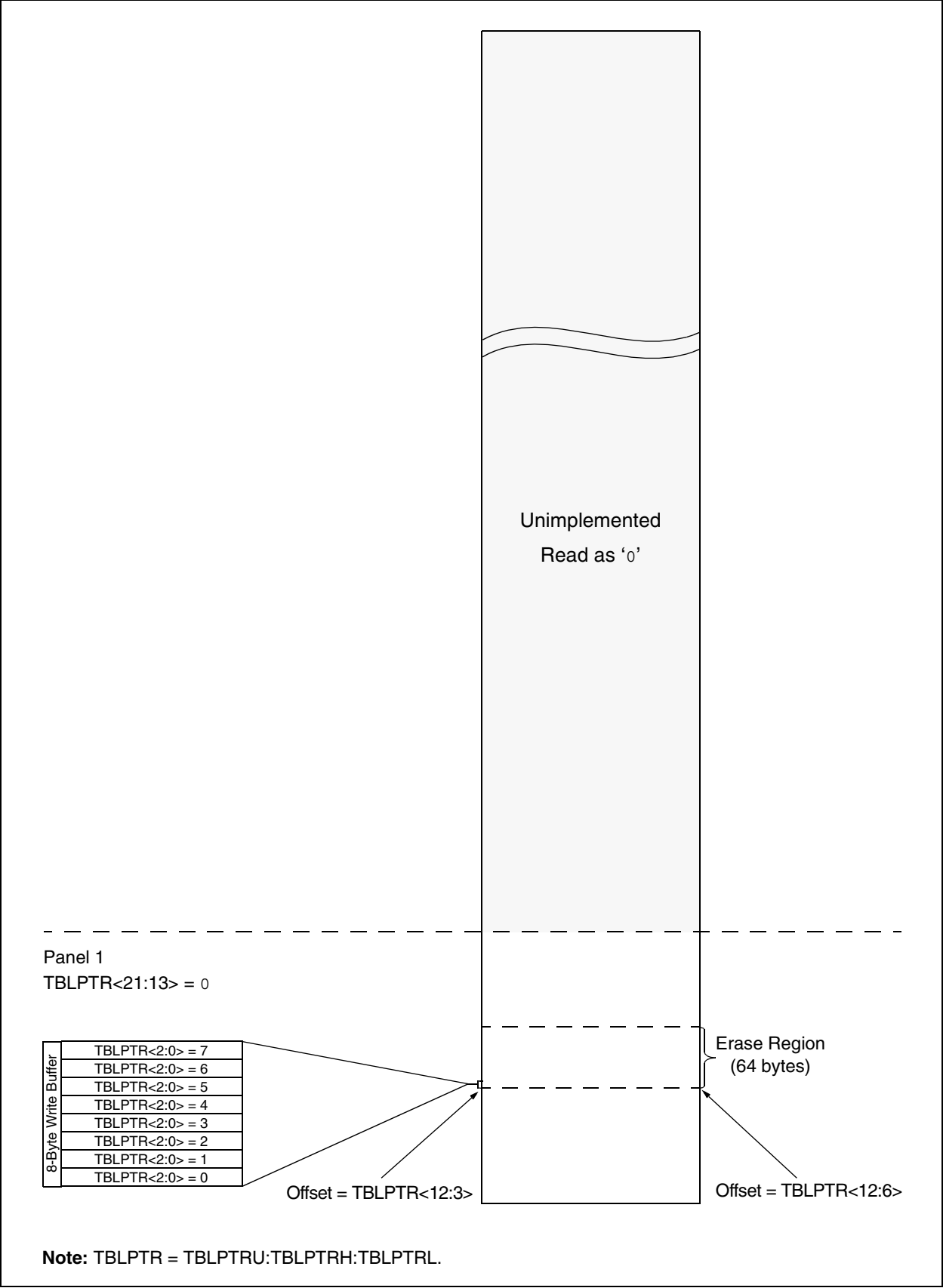
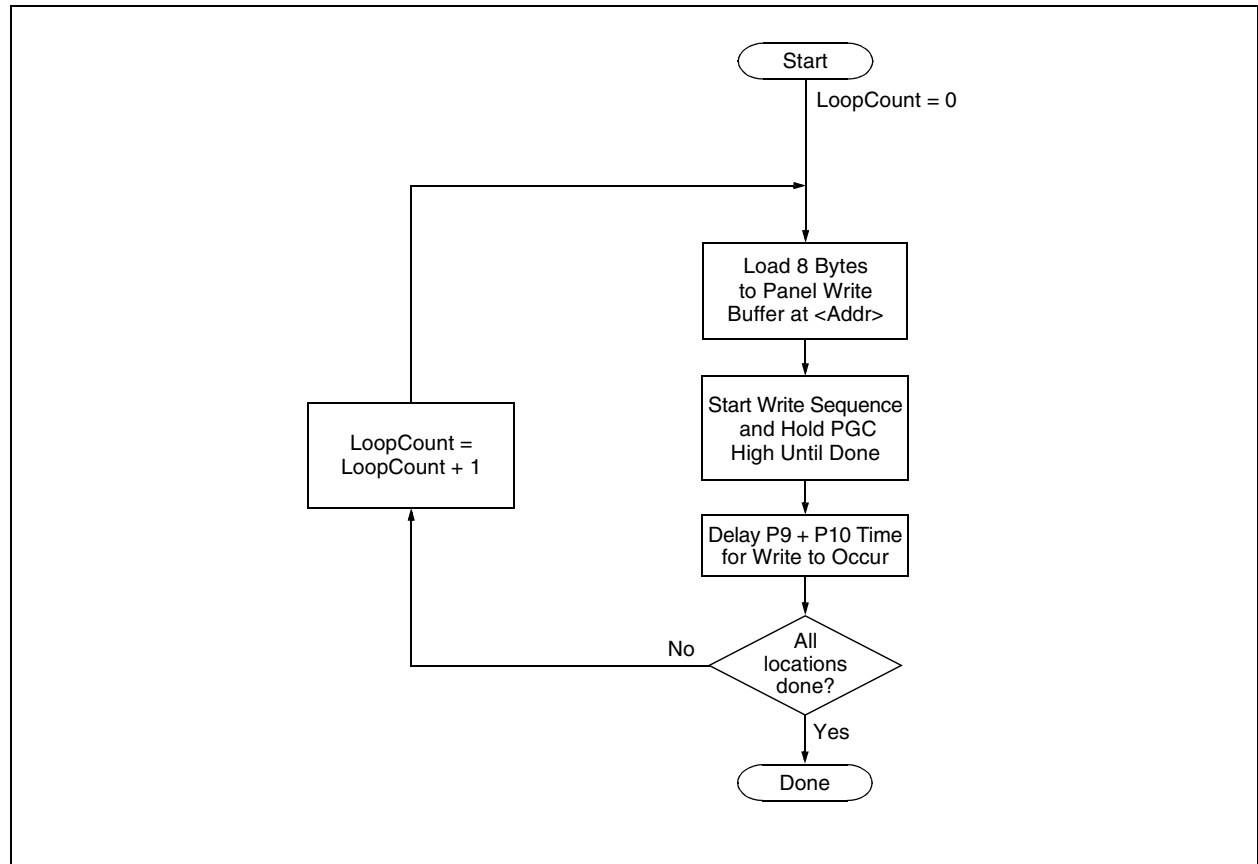


TABLE 3-3: WRITE CODE MEMORY CODE SEQUENCE

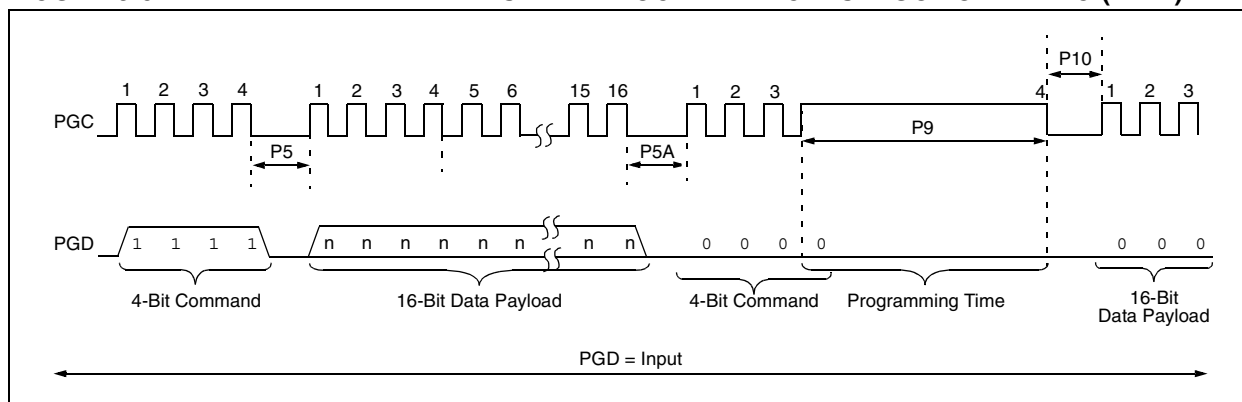
4-Bit Command	Data Payload	Core Instruction
Step 1: Direct access to code memory.		
0000	8E A6	BSF EECON1, EEPGD
0000	9C A6	BCF EECON1, CFGS
Step 2: Load write buffer for Panel 1.		
0000	0E <Addr[21:16]>	MOVLW <Addr[21:16]>
0000	6E F8	MOVWF TBLPTRU
0000	0E <Addr[15:8]>	MOVLW <Addr[15:8]>
0000	6E F7	MOVWF 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
1111	<LSB><MSB>	Write 2 bytes and start programming
0000	00 00	NOP - hold PGC high for time P9
To continue writing data, repeat step 2, where the Address Pointer is incremented by 8 at each iteration of the loop.		

FIGURE 3-5: PROGRAM CODE MEMORY FLOW



PIC18FX220/X320

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



3.3.1 MODIFYING CODE MEMORY

All of the programming examples, up to this point, have assumed that the device is blank prior to programming. In fact, if the device is not blank, the direction has been to completely erase the device via a Bulk Erase operation (see **Section 3.2 “High-Voltage ICSP Bulk Erase”**).

It may be the case, however, that the user wishes to modify only a section of an already programmed device. In such a situation, erasing the entire device is not a realistic option.

The minimum amount of data that can be written to the device is 8 bytes. This is accomplished by 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 one row of 64 bytes and it is selected using the TBLPTR registers. The sixth LSb of the TBLPTR address is ignored. The EECON1 register must then be used to erase the 64-byte target space prior to writing the data. This is known as a “Row Erase”.

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 0AAh, immediately prior to asserting the WR bit in order for the write to occur.

The erase will begin on the falling edge of the 4th PGC after the WR bit is set.

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

TABLE 3-4: 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: 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[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 3: Enable memory writes and set up an erase.		
0000	84 A6	BSF EECON1, WREN
0000	88 A6	BSF EECON1, FREE
Step 4: Perform Flash unlock sequence.		
0000	0E 55	MOVLW 0X55
0000	6E A7	MOVWF EECON2
0000	0E AA	MOVLW 0XAA
0000	6E A7	MOVWF EECON2
Step 5: Initiate erase.		
0000	82 A6	BSF EECON1, WR
0000	00 00	NOP
Step 6: Wait for P11 + P10 and then disable writes.		
0000	94 A6	BCF EECON1, WREN
Step 7: Load write buffer for panel.		
1101	<LSB><MSB>	Write 2 bytes and post-increment address by 2
1101	<LSB><MSB>	Write 2 bytes and post-increment address by 2
1101	<LSB><MSB>	Write 2 bytes and post-increment address by 2
1111	<LSB><MSB>	Write 2 bytes and start programming
0000	00 00	NOP - hold PGC high for time P9 at the end of 4-bit command
To continue writing data, repeat step 7, where the Address Pointer is incremented by 8 at each iteration of the loop.		

PIC18FX220/X320

3.4 Data EEPROM Programming

Data EEPROM is accessed one byte at a time via an Address Pointer, EEADR, and a Data Latch, EEDATA. Data EEPROM is written by loading EEADR with the desired memory location, loading 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, the EEPGD bit must be cleared ($\text{EECON1}\langle 7 \rangle = 0$) and the CFGS bit must be cleared ($\text{EECON1}\langle 6 \rangle = 0$). The WREN bit must be set ($\text{EECON1}\langle 2 \rangle = 1$) to enable writes of any sort and this must be done prior to initiating a write sequence.

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 0AAh, immediately prior to asserting the WR bit in order for the write to occur. The write sequence is initiated by setting the WR bit ($\text{EECON1}\langle 1 \rangle = 1$). It is strongly recommended that the WREN bit be set only when absolutely necessary.

The write will begin on the falling edge of the 4th PGC after the WR bit is set.

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

FIGURE 3-7: PROGRAM DATA FLOW

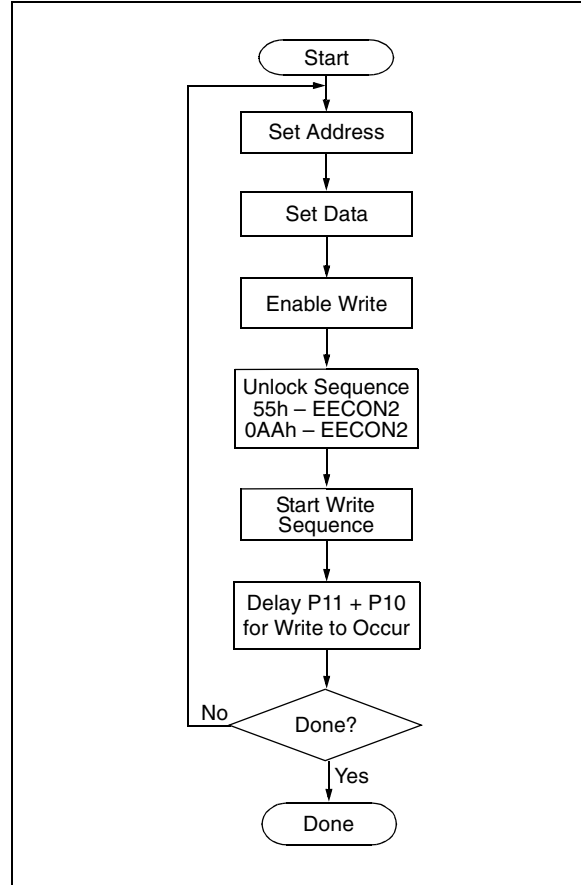


FIGURE 3-8: DATA EEPROM WRITE TIMING

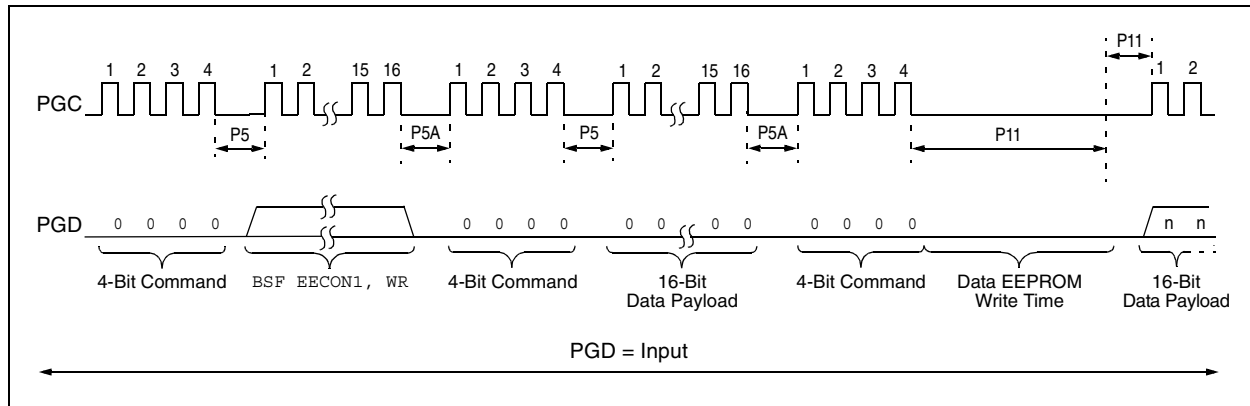


TABLE 3-5: PROGRAMMING DATA MEMORY

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
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 data EEPROM unlock 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
0000	00 00	NOP
0000	00 00	NOP
Step 7: Wait for P11 and then disable writes.		
0000	94 A6	BCF EECON1, WREN
Repeat steps 2 through 7 to write more data.		

PIC18FX220/X320

3.5 ID Location Programming

The ID locations are programmed much like the code memory. 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: The user must fill the 8-byte data buffer for the panel.

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

The Table Pointer must be manually set to 200000h (base address of the ID locations). The post-increment feature of the table read 4-bit command may not be used to increment the Table Pointer to 200000h. The post-increment feature may then be used to increment to 200001h, 200002h, etc.

TABLE 3-6: 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
Step 2: Load write buffer.		
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 PGC high for time P9

3.6 Boot Block Programming

The Boot Block segment is programmed in exactly the same manner as the ID locations (see **Section 3.5 “ID Location Programming”**).

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

3.7 Configuration Bits Programming

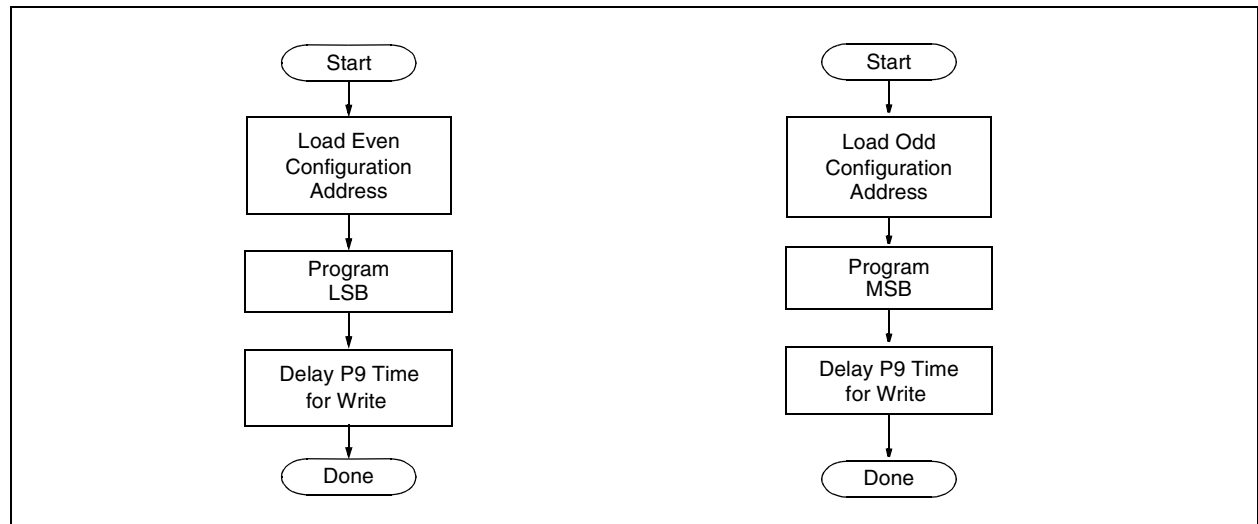
Unlike code memory, the Configuration bits are programmed a byte at a time. The “Table Write, Begin Programming” 4-bit command (1111) is used, but only 8 bits of the following 16-bit payload will be written. The LSB of the payload will be written to even addresses and the MSB will be written to odd addresses. The code sequence to program two consecutive configuration locations is shown in Table 3-7.

TABLE 3-7: 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
Step 2: Position the program counter. ⁽¹⁾		
0000	EF 00	GOTO 0x100000
0000	F8 00	
Step 3: Set Table Pointer for Configuration Word 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 PGC high for time P9
0000	2A F6	INCF TBLPTRL
1111	<LSB ignored><MSB>	Load 2 bytes and start programming
0000	00 00	NOP - hold PGC 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 writes. To avoid this situation, move the program counter outside the code protection area (e.g., GOTO 0x100000).
- 2:** Enabling the write protection of Configuration bits (WRTC = 0 in CONFIG6H) will prevent further writing of Configuration bits. Always write all the Configuration bits before enabling the write protection for Configuration bits.

FIGURE 3-9: CONFIGURATION PROGRAMMING FLOW



PIC18FX220/X320

4.0 READING THE DEVICE

4.1 Read Data EEPROM Memory

Data EEPROM is accessed one byte at a time via an Address Pointer, 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 4-2).

The command sequence to read a single byte of data is shown in Table 4-1.

FIGURE 4-1: READ DATA EEPROM FLOW

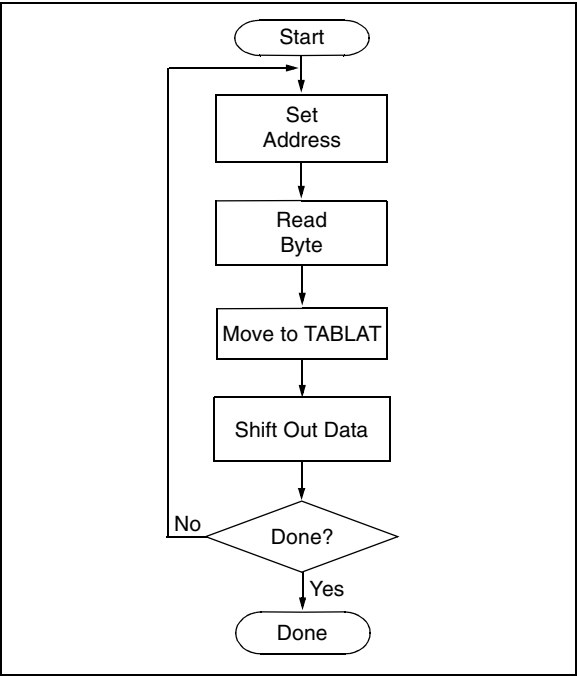
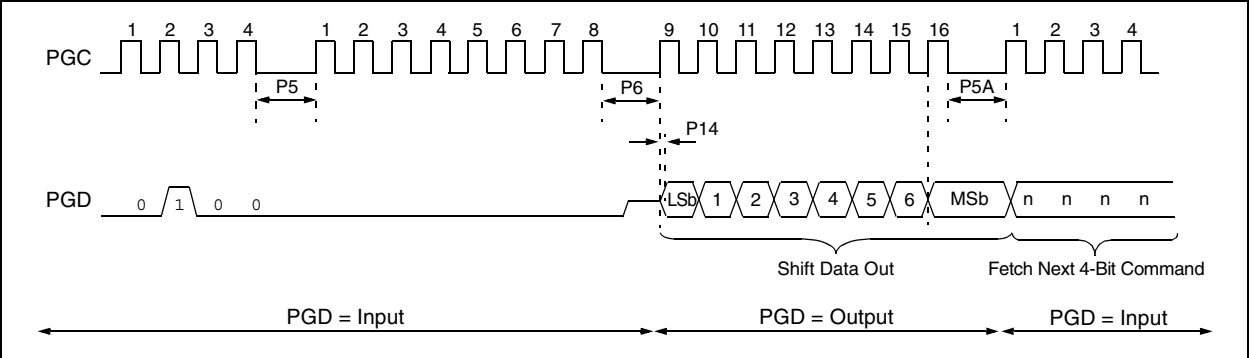


TABLE 4-1: READ DATA EEPROM MEMORY

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
Step 3: Initiate a memory read.		
0000	80 A6	BSF EECON1, RD
Step 4: Load data into the Serial Data Holding register.		
0000	50 A8	MOVF EEDATA, W, 0
0000	6E F5	MOVWF TABLAT
0010	<LSB><MSB>	Shift Out Data ⁽¹⁾

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

FIGURE 4-2: SHIFT OUT DATA HOLDING REGISTER TIMING (0010)



4.2 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 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

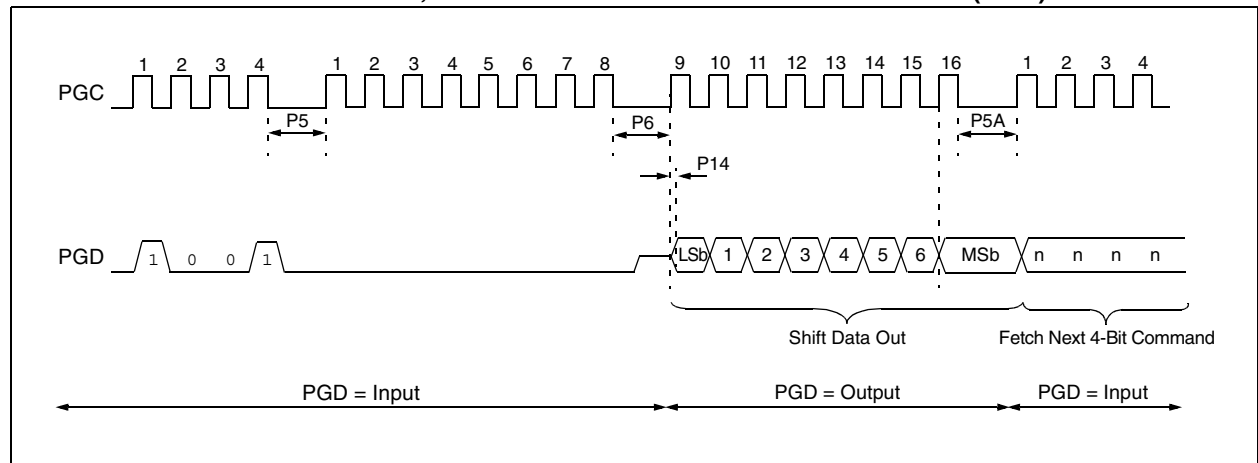
P6 must be introduced after the falling edge of the 8th PGC of the operand to allow PGD to transition from an input to an output. During this time, PGC must be held low (see Table 4-2). This operation also increments the Table Pointer by one, pointing to the next byte in code memory for the next read.

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

TABLE 4-2: 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 PGD, LSb to MSb.		
1001	00 00	TBLRD *+

FIGURE 4-3: TABLE READ, POST-INCREMENT INSTRUCTION TIMING (1001)



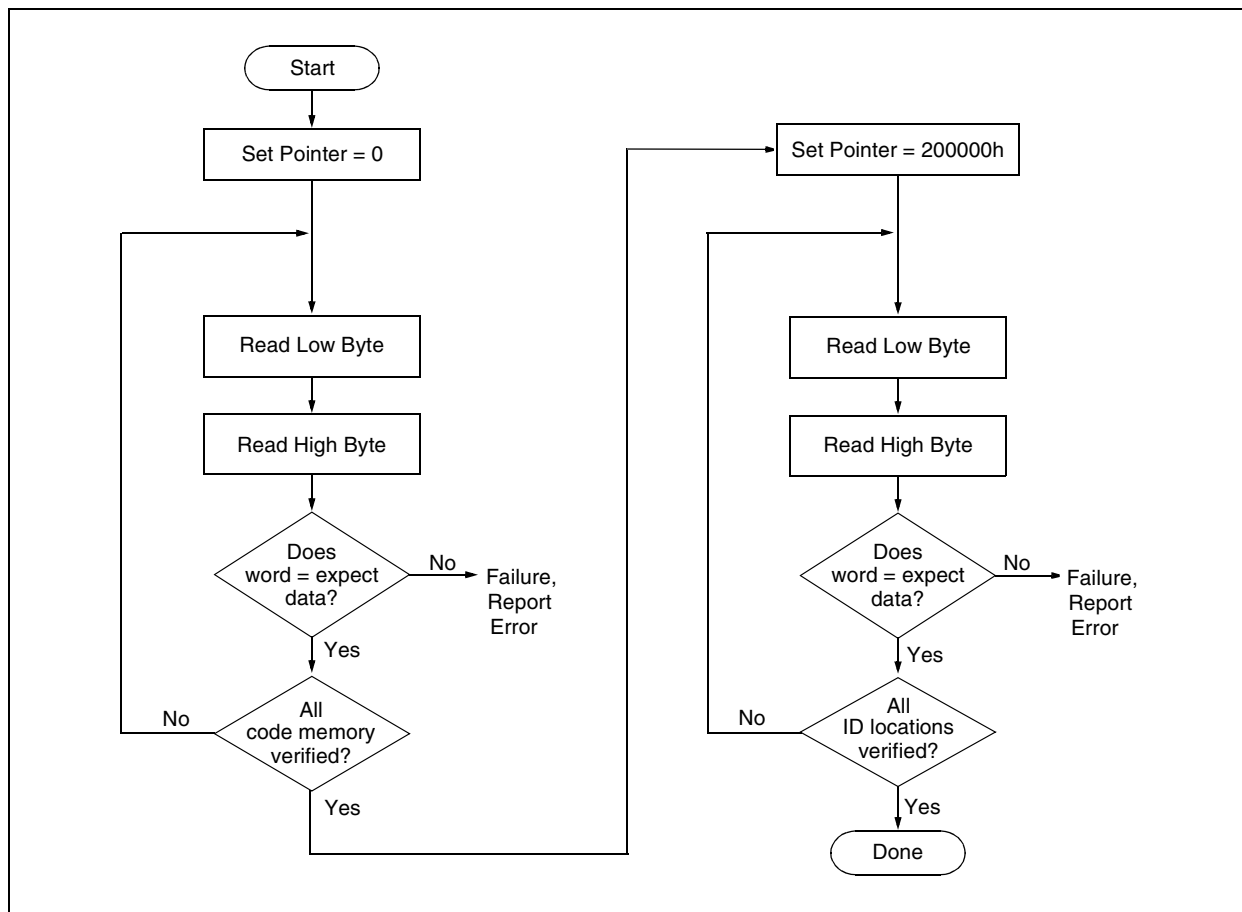
PIC18FX220/X320

4.3 Verify Code Memory and ID Locations

The verify step involves reading back the code memory space and comparing it against the copy held in the programmer's buffer. Memory reads occur a single byte at a time, so two bytes must be read to compare against the word in the programmer's buffer. Refer to **Section 4.2 "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). The post-increment feature of the table read 4-bit command may not be used to increment the Table Pointer to 200000h. The post-increment feature may then be used to increment to 200001h, 200002h, etc.

FIGURE 4-4: VERIFY CODE MEMORY FLOW



4.4 Verify Configuration Bits

A configuration address may be read and output on PGD via the 4-bit command, '1001'. Configuration data is read and written in a byte-wise fashion, so it is not necessary to merge two bytes into a word prior to a compare. The result may then be immediately compared to the appropriate configuration data in the programmer's memory for verification. Refer to **Section 4.2 "Read Code Memory, ID Locations and Configuration Bits"** for implementation details of reading configuration data.

4.5 Verify Data EEPROM

A data EEPROM address may be read via a sequence of core instructions (4-bit command, '0000') and then output on PGD via the 4-bit command, '0010' (Shift Out Data Holding register). The result may then be immediately compared to the appropriate data in the programmer's memory for verification. Refer to **Section 4.1 "Read Data EEPROM Memory"** for implementation details of reading data EEPROM.

5.0 CONFIGURATION WORD

The devices have several Configuration Words. Bits in these registers can be set or cleared to select various device configurations. All other memory areas should be programmed and verified prior to setting Configuration Words. These bits may be read out normally, even after read or code-protected. Tables 5-2, 5-3 and 5-4 provide information on various Configuration bits.

5.1 ID Locations

A user may store identification information (ID) in eight ID locations mapped in 200000h:200007h. It is recommended that the most significant nibble of each ID be 0Fh. In doing so, if the user code inadvertently tries to execute from the ID space, the ID data will execute as a NOP.

5.2 Device ID Word

The Device ID Word for the 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.

5.3 Single-Supply ICSP Programming

The LVP bit in Configuration register, CONFIG4L, enables Single-Supply (Low-Voltage) ICSP Programming mode. The LVP bit defaults to a '1' (enabled) following an erase.

If Single-Supply Programming mode is not used, the LVP bit can be programmed to a '0' and RB5/PGM becomes a digital I/O pin. However, the LVP bit may only be programmed by entering the High-Voltage ICSP mode, where MCLR/VPP is raised to VIH. 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 High-Voltage ICSP mode is always available, regardless of the state of the LVP bit, by applying VIH to the MCLR/VPP pin.

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

TABLE 5-1: DEVICE ID VALUE

Device	Device ID Value	
	DEVID2	DEVID1
PIC18F1220	07	111x xxxx
PIC18F2220	05	100x xxxx
PIC18F4220	05	101x xxxx
PIC18F1320	07	110x xxxx
PIC18F2320	05	000x xxxx
PIC18F4320	05	001x xxxx

PIC18FX220/X320

TABLE 5-2: PIC18F2X20/4X20 CONFIGURATION BITS AND DEVICE IDs

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Erased or "Blank" Value
300000h CONFIG1L	—	—	—	—	—	—	—	—	-----
300001h CONFIG1H	IESO	FSCM	—	—	FOSC3	FOSC2	FOSC1	FOSC0	11-- 1111
300002h CONFIG2L	—	—	—	—	BORV1	BORV0	BOR	PWRT	---- 1111
300003h CONFIG2H	—	—	—	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDT	---1 1111
300004h CONFIG3L	—	—	—	—	—	—	—	—	-----
300005h CONFIG3H	MCLRE	—	—	—	—	—	PBAD	CCP2MX	1--- --11
300006h CONFIG4L	DEBUG	—	—	—	—	LVP	—	STVR	1--- -1-1
300007h CONFIG4H	—	—	—	—	—	—	—	—	-----
300008h CONFIG5L	—	—	—	—	CP3 ⁽¹⁾	CP2 ⁽¹⁾	CP1	CP0	---- 1111
300009h CONFIG5H	CPD	CPB	—	—	—	—	—	—	11-- ----
30000Ah CONFIG6L	—	—	—	—	WRT3 ⁽¹⁾	WRT2 ⁽¹⁾	WRT1	WRT0	---- 1111
30000Bh CONFIG6H	WRTD	WRTB	WRTC	—	—	—	—	—	111- ----
30000Ch CONFIG7L	—	—	—	—	EBTR3 ⁽¹⁾	EBTR2 ⁽¹⁾	EBTR1	EBTR0	---- 1111
30000Dh CONFIG7H	—	EBTRB	—	—	—	—	—	—	-1-- ----
3FFFFEh DEVID1	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	Table 5-1
3FFFFFh DEVID2	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	Table 5-1

Legend: - = unimplemented. Shaded cells are unimplemented, read as '0'.

Note 1: Unimplemented in PIC18F2220 and PIC18F4220 devices, read as '0'.

TABLE 5-3: PIC18F1X20 CONFIGURATION BITS AND DEVICE IDs

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Erased or "Blank" Value
300000h CONFIG1L	—	—	—	—	—	—	—	—	-----
300001h CONFIG1H	IESO	FSCM	—	—	FOSC3	FOSC2	FOSC1	FOSC0	11-- 1111
300002h CONFIG2L	—	—	—	—	BORV1	BORV0	BOR	PWRTEN	---- 1111
300003h CONFIG2H	—	—	—	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDT	---1 1111
300004h CONFIG3L	—	—	—	—	—	—	—	—	-----
300005h CONFIG3H	MCLRE	—	—	—	—	—	—	—	1--- ----
300006h CONFIG4L	DEBUG	—	—	—	—	LVP	—	STVR	1--- -1-1
300007h CONFIG4H	—	—	—	—	—	—	—	—	-----
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	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	Table 5-1
3FFFFFh DEVID2	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	Table 5-1

Legend: - = Unimplemented. Shaded cells are unimplemented, read as '0'.

TABLE 5-4: PIC18FX220/X320 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
FSCM	CONFIG1H	Fail-Safe Clock Monitor Enable bit 1 = Fail-Safe Clock Monitor enabled 0 = Fail-Safe Clock Monitor disabled
FOSC3:FOSC0	CONFIG1H	Oscillator Selection bits 11xx = External RC oscillator, CLKO function on RA6 101x = External RC oscillator, CLKO function on RA6 1001 = Internal RC oscillator, CLKO function on RA6, port function on RA7 1000 = Internal RC oscillator, port function on RA6, port function on RA7 0111 = External RC oscillator, port function on RA6 0110 = HS oscillator, PLL enabled (Clock Frequency = 4 x FOSC1) 0101 = EC oscillator, port function on RA6 0100 = EC oscillator, CLKO function on RA6 0011 = External RC oscillator, CLKO function on RA6 0010 = HS oscillator 0001 = XT oscillator 0000 = LP oscillator
BORV1:BORV0	CONFIG2L	Brown-out Reset Voltage bits 11 = Reserved 10 = VBOR set to 2.7V 01 = VBOR set to 4.2V 00 = VBOR set to 4.5V
BOR	CONFIG2L	Brown-out Reset Enable bit 1 = Brown-out Reset enabled 0 = Brown-out Reset disabled
PWRT	CONFIG2L	Power-up Timer Enable bit for PIC18F2X20/4X20 1 = PWRT disabled 0 = PWRT enabled
PWRTEN	CONFIG2L	Power-up Timer Enable bit for PIC18F1X20 1 = PWRT disabled 0 = PWRT enabled
WDTPS3:WDTPS0	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
WDT	CONFIG2H	Watchdog Timer Enable bit 1 = WDT enabled 0 = WDT disabled (control is placed on SWDTEN bit)

PIC18FX220/X320

TABLE 5-4: PIC18FX220/X320 BIT DESCRIPTIONS (CONTINUED)

Bit Name	Configuration Words	Description
MCLRE	CONFIG3H	MCLR Pin Enable bit for PIC18F2X20/4X20 1 = MCLR pin enabled, RE3 input pin disabled 0 = RE3 input pin enabled, MCLR pin disabled
MCLRE	CONFIG3H	MCLR Pin Enable bit for PIC18F1X20 1 = MCLR pin enabled, RA5 input pin disabled 0 = RA5 input pin enabled, MCLR pin disabled
PBAD	CONFIG3H	PORTB A/D Enable bit for PIC18F2X20/4X20 1 = PORTB A/D<4:0> pins are configured as analog input channels on Reset 0 = PORTB A/D<4:0> pins are configured as digital I/O on Reset
CCP2MX	CONFIG3H	CCP2 MUX bit for PIC18F2X20/4X20 1 = CCP2 input/output is multiplexed with RC1 0 = CCP2 input/output is multiplexed with RB3
DEBUG	CONFIG4L	In-Circuit Debugger Enable bit 1 = In-Circuit Debugger disabled (RB6, RB7 have I/O port function) 0 = In-Circuit Debugger enabled (RB6, RB7 have ICSP™ serial communication function)
LVP	CONFIG4L	Low-Voltage Programming Enable bit 1 = Low-Voltage Programming enabled, RB5 is the PGM pin 0 = Low-Voltage Programming disabled, RB5 is an I/O pin
STVR	CONFIG4L	Stack Overflow/Underflow Reset Enable bit 1 = Reset on stack overflow/underflow enabled 0 = Reset on stack overflow/underflow disabled
CP3	CONFIG5L	Code Protection bit for PIC18F2320/4320 (Block 3 code memory area: 001800h-001FFFh, unimplemented in PIC18F2220/4220) 1 = Block 3 is not code-protected 0 = Block 3 is code-protected
CP2	CONFIG5L	Code Protection bit for PIC18F2320/4320 (Block 2 code memory area: 001000h-0017FFh, unimplemented in PIC18F2220/4220) 1 = Block 2 is not code-protected 0 = Block 2 is code-protected
CP1	CONFIG5L	Code Protection bit for PIC18F1220 (Block 1 code memory area: 000800h-000FFFh) 1 = Block 1 is not code-protected 0 = Block 1 is code-protected
CP1	CONFIG5L	Code Protection bit for PIC18F1320 (Block 1 code memory area: 001000h-001FFFh) 1 = Block 1 is not code-protected 0 = Block 1 is code-protected
CP1	CONFIG5L	Code Protection bit for PIC18F2X20/4X20 (Block 1 code memory area: 000800h-000FFFh) 1 = Block 1 is not code-protected 0 = Block 1 is code-protected
CP0	CONFIG5L	Code Protection bit for PIC18F1220 (Block 0 code memory area: 000200h-0007FFh) 1 = Block 0 is not code-protected 0 = Block 0 is code-protected
CP0	CONFIG5L	Code Protection bit for PIC18F1320 (Block 0 code memory area: 000200h-0007FFh) 1 = Block 0 is not code-protected 0 = Block 0 is code-protected
CP0	CONFIG5L	Code Protection bit for PIC18F2X20/4X20 (Block 0 code memory area: 000200h-0007FFh) 1 = Block 0 is not code-protected 0 = Block 0 is code-protected

TABLE 5-4: PIC18FX220/X320 BIT DESCRIPTIONS (CONTINUED)

Bit Name	Configuration Words	Description
CPD	CONFIG5H	Code Protection bit (Data EEPROM) 1 = Data EEPROM is not code-protected 0 = Data EEPROM is code-protected
CPB	CONFIG5H	Code Protection bit (Boot Block memory area: 000000h-0001FFh) 1 = Boot Block is not code-protected 0 = Boot Block is code-protected
WRT3	CONFIG6L	Write Protection bit for PIC18F2320/4320 (Block 3 code memory area: 001800h-001FFFh, unimplemented in PIC18F2220/4220) 1 = Block 3 is not write-protected 0 = Block 3 is write-protected
WRT2	CONFIG6L	Write Protection bit for PIC18F2320/4320 (Block 2 code memory area: 001000h-0017FFh, unimplemented in PIC18F2220/4220) 1 = Block 2 is not write-protected 0 = Block 2 is write-protected
WRT1	CONFIG6L	Write Protection bit for PIC18F1220 (Block 1 code memory area: 000800h-000FFFh) 1 = Block 1 is not write-protected 0 = Block 1 is write-protected
WRT1	CONFIG6L	Write Protection bit for PIC18F1320 (Block 1 code memory area: 001000h-001FFFh) 1 = Block 1 is not write-protected 0 = Block 1 is write-protected
WRT1	CONFIG6L	Write Protection bit for PIC18F2X20/4X20 (Block 1 code memory area: 000800h-000FFFh) 1 = Block 1 is not write-protected 0 = Block 1 is write-protected
WRT0	CONFIG6L	Write Protection bit for PIC18F1220 (Block 0 code memory area: 000200h-0007FFh) 1 = Block 0 is not write-protected 0 = Block 0 is write-protected
WRT0	CONFIG6L	Write Protection bit for PIC18F1320 (Block 0 code memory area: 000200h-000FFFh) 1 = Block 0 is not write-protected 0 = Block 0 is write-protected
WRT0	CONFIG6L	Write Protection bit for PIC18F2X20/4X20 (Block 0 code memory area: 000200h-0007FFh) 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: 000000h-0001FFh) 1 = Boot Block is not write-protected 0 = Boot Block is write-protected
WRTC	CONFIG6H	Write Protection bit (Configuration registers: 300000h-3000FFh) 1 = Configuration registers are not write-protected 0 = Configuration registers are write-protected
EBTR3	CONFIG7L	Table Read Protection bit for PIC18F2320/4320 (Block 3 code memory area: 001800h-001FFFh, unimplemented in PIC18F2220/4220) 1 = Block 3 is not protected from table reads executed in other blocks 0 = Block 3 is protected from table reads executed in other blocks
EBTR2	CONFIG7L	Table Read Protection bit for PIC18F2320/4320 (Block 2 code memory area: 001000h-0017FFh, unimplemented in PIC18F2220/4220) 1 = Block 2 is not protected from table reads executed in other blocks 0 = Block 2 is protected from table reads executed in other blocks

PIC18FX220/X320

TABLE 5-4: PIC18FX220/X320 BIT DESCRIPTIONS (CONTINUED)

Bit Name	Configuration Words	Description
EBTR1	CONFIG7L	Table Read Protection bit for PIC18F1220 (Block 1 code memory area: 000800h-000FFFh) 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
EBTR1	CONFIG7L	Table Read Protection bit for PIC18F1320 (Block 1 code memory area: 001000h-001FFFh) 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
EBTR1	CONFIG7L	Table Read Protection bit for PIC18F2X20/4X20 (Block 1 code memory area: 000800h-000FFFh) 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 for PIC18F1220 (Block 0 code memory area: 000200h-0007FFh) 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
EBTR0	CONFIG7L	Table Read Protection bit for PIC18F1320 (Block 0 code memory area: 000200h-000FFFh) 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
EBTR0	CONFIG7L	Table Read Protection bit for PIC18F2X20/4X20 (Block 0 code memory area: 000200h-0007FFh) 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: 000000h-0001FFh) 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
DEV10:DEV3	DEVID2	Device ID bits These bits are used with the DEV2:DEV0 bits in the DEVID1 register to identify the part number.
DEV2:DEV0	DEVID1	Device ID bits These bits are used with the DEV10:DEV3 bits in the DEVID2 register to identify the part number.
REV4:REV0	DEVID1	Revision ID bits These bits are used to indicate the revision of the device.

5.4 Embedding Configuration Word Information in the HEX File

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

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

5.5 Embedding Data EEPROM Information in the HEX File

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

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

5.6 Checksum Computation

The checksum is calculated by summing the following:

- The contents of all code memory locations
- The Configuration Word, appropriately masked
- ID locations

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

Table 5-5 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.
--------------	---

PIC18FX220/X320

TABLE 5-5: CHECKSUM COMPUTATION – PIC18FX220/X320

Device	Code-Protect	Checksum	Blank Value	0xAA at 0 and Max Address
PIC18F1220	None	SUM (0000:01FFh) + SUM (0200:0FFFh) + SUM (1000:1FFFh) + (CONFIG1H & 0CFh) + (CONFIG2L & 0Fh) + (CONFIG2H & 1Fh) + (CONFIG3H & 080h) + (CONFIG4L & 085h) + (CONFIG5L & 03h) + (CONFIG5H & 0C0h) + (CONFIG6L & 03h) + (CONFIG6H & 0E0h) + (CONFIG7L & 03h) + (CONFIG7H & 040h)	F3EB	F341
	Boot Block	SUM (0200:0FFFh) + SUM (1000:1FFFh) + (CONFIG1H & 0CFh) + (CONFIG2L & 0Fh) + (CONFIG2H & 1Fh) + (CONFIG3H & 080h) + (CONFIG4L & 085h) + (CONFIG5L & 03h) + (CONFIG5H & 0C0h) + (CONFIG6L & 03h) + (CONFIG6H & 0E0h) + (CONFIG7L & 03h) + (CONFIG7H & 040h) + SUM (IDs)	F5D6	F56D
	Boot/ Panel 1/ Panel 2	(CONFIG1H & 0CFh) + (CONFIG2L & 0Fh) + (CONFIG2H & 1Fh) + (CONFIG3H & 080h) + (CONFIG4L & 085h) + (CONFIG5L & 03h) + (CONFIG5H & 0C0h) + (CONFIG6L & 03h) + (CONFIG6H & 0E0h) + (CONFIG7L & 03h) + (CONFIG7H & 040h) + SUM (IDs)	03D3	03BF
	All	(CONFIG1H & 0CFh) + (CONFIG2L & 0Fh) + (CONFIG2H & 1Fh) + (CONFIG3H & 080h) + (CONFIG4L & 085h) + (CONFIG5L & 03h) + (CONFIG5H & 0C0h) + (CONFIG6L & 03h) + (CONFIG6H & 0E0h) + (CONFIG7L & 03h) + (CONFIG7H & 040h) + SUM (IDs)	03D3	03BF
PIC18F1320	None	SUM (0000:01FFh) + SUM (0200:07FFh) + SUM (0800:0FFFh) + (CONFIG1H & 0CFh) + (CONFIG2L & 0Fh) + (CONFIG2H & 1Fh) + (CONFIG3H & 080h) + (CONFIG4L & 085h) + (CONFIG5L & 03h) + (CONFIG5H & 0C0h) + (CONFIG6L & 03h) + (CONFIG6H & 0E0h) + (CONFIG7L & 03h) + (CONFIG7H & 040h)	E3EB	E341
	Boot Block	SUM (0200:07FFh) + SUM (0800:0FFFh) + (CONFIG1H & 0CFh) + (CONFIG2L & 0Fh) + (CONFIG2H & 1Fh) + (CONFIG3H & 080h) + (CONFIG4L & 085h) + (CONFIG5L & 03h) + (CONFIG5H & 0C0h) + (CONFIG6L & 03h) + (CONFIG6H & 0E0h) + (CONFIG7L & 03h) + (CONFIG7H & 040h) + SUM (IDs)	E5D5	E56C
	Boot/ Panel 1/ Panel 2	(CONFIG1H & 0CFh) + (CONFIG2L & 0Fh) + (CONFIG2H & 1Fh) + (CONFIG3H & 080h) + (CONFIG4L & 085h) + (CONFIG5L & 03h) + (CONFIG5H & 0C0h) + (CONFIG6L & 03h) + (CONFIG6H & 0E0h) + (CONFIG7L & 03h) + (CONFIG7H & 040h) + SUM (IDs)	03D2	03BE
	All	(CONFIG1H & 0CFh) + (CONFIG2L & 0Fh) + (CONFIG2H & 1Fh) + (CONFIG3H & 083h) + (CONFIG4L & 085h) + (CONFIG5L & 0Fh) + (CONFIG5H & 0C0h) + (CONFIG6L & 0Fh) + (CONFIG6H & 0E0h) + (CONFIG7L & 0Fh) + (CONFIG7H & 040h) + SUM (IDs)	03D2	03BE

Legend:

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

TABLE 5-5: CHECKSUM COMPUTATION – PIC18FX220/X320 (CONTINUED)

Device	Code-Protect	Checksum	Blank Value	0xAA at 0 and Max Address
PIC18F2220/ PIC18F4220	None	SUM (0000:01FFh) + SUM (0200:07FFh) + SUM (0800:0FFFh) + (CONFIG1H & 0CFh) + (CONFIG2L & 0Fh) + (CONFIG2H & 1Fh) + (CONFIG3H & 083h) + (CONFIG4L & 085h) + (CONFIG5L & 0Fh) + (CONFIG5H & 0C0h) + (CONFIG6L & 0Fh) + (CONFIG6H & 0E0h) + (CONFIG7L & 0Fh) + (CONFIG7H & 040h)	0F412h	0F368h
	Boot Block	SUM (0200:07FFh) + SUM (0800:0FFFh) + (CONFIG1H & 0CFh) + (CONFIG2L & 0Fh) + (CONFIG2H & 1Fh) + (CONFIG3H & 083h) + (CONFIG4L & 085h) + (CONFIG5L & 0Fh) + (CONFIG5H & 0C0h) + (CONFIG6L & 0Fh) + (CONFIG6H & 0E0h) + (CONFIG7L & 0Fh) + (CONFIG7H & 040h) + SUM (IDs)	0F5E8h	0F59Dh
	Boot Block/ Block 0	SUM (0800:0FFFh) + (CONFIG1H & 0CFh) + (CONFIG2L & 0Fh) + (CONFIG2H & 1Fh) + (CONFIG3H & 083h) + (CONFIG4L & 085h) + (CONFIG5L & 0Fh) + (CONFIG5H & 0C0h) + (CONFIG6L & 0Fh) + (CONFIG6H & 0E0h) + (CONFIG7L & 0Fh) + (CONFIG7H & 040h) + SUM (IDs)	0FBE7h	0FB9Ch
	Boot Block/ Block 0/ Block 1	(CONFIG1H & 0CFh) + (CONFIG2L & 0Fh) + (CONFIG2H & 1Fh) + (CONFIG3H & 083h) + (CONFIG4L & 085h) + (CONFIG5L & 0Fh) + (CONFIG5H & 0C0h) + (CONFIG6L & 0Fh) + (CONFIG6H & 0E0h) + (CONFIG7L & 0Fh) + (CONFIG7H & 040h) + SUM (IDs)	03E5h	03EFh
	All	(CONFIG1H & 0CFh) + (CONFIG2L & 0Fh) + (CONFIG2H & 1Fh) + (CONFIG3H & 083h) + (CONFIG4L & 085h) + (CONFIG5L & 0Fh) + (CONFIG5H & 0C0h) + (CONFIG6L & 0Fh) + (CONFIG6H & 0E0h) + (CONFIG7L & 0Fh) + (CONFIG7H & 040h) + SUM (IDs)	03E5h	03EFh

Legend:

<u>Item</u>	<u>Description</u>
CONFIG	= Configuration Word
SUM[a:b]	= Sum of locations, a to b inclusive
SUM (IDs)	= Byte-wise sum of lower four bits of all ID locations
+	= Addition
&	= Bit-wise AND

PIC18FX220/X320

TABLE 5-5: CHECKSUM COMPUTATION – PIC18FX220/X320 (CONTINUED)

Device	Code-Protect	Checksum	Blank Value	0xAA at 0 and Max Address
PIC18F2320/ PIC18F4320	None	SUM (0000:01FFh) + SUM (0200:07FFh) + SUM (0800:0FFFh) + SUM (1000:17FFh) + SUM (1800:1FFFh) + (CONFIG1H & 0CFh) + (CONFIG2L & 0Fh) + (CONFIG2H & 1Fh) + (CONFIG3H & 083h) + (CONFIG4L & 085h) + (CONFIG5L & 0Fh) + (CONFIG5H & 0C0h) + (CONFIG6L & 0Fh) + (CONFIG6H & 0E0h) + (CONFIG7L & 0Fh) + (CONFIG7H & 040h)	0E412h	0E368h
	Boot Block	SUM (0200:07FFh) + SUM (0800:0FFFh) + SUM (1000:17FFh) + SUM (1800:1FFFh) + (CONFIG1H & 0CFh) + (CONFIG2L & 0Fh) + (CONFIG2H & 1Fh) + (CONFIG3H & 083h) + (CONFIG4L & 085h) + (CONFIG5L & 0Fh) + (CONFIG5H & 0C0h) + (CONFIG6L & 0Fh) + (CONFIG6H & 0E0h) + (CONFIG7L & 0Fh) + (CONFIG7H & 040h) + SUM (IDs)	0E5E7h	0E59Ch
	Boot Block/ Block 0	SUM (0800:0FFFh) + SUM (1000:17FFh) + SUM (1800:1FFFh) + (CONFIG1H & 0CFh) + (CONFIG2L & 0Fh) + (CONFIG2H & 1Fh) + (CONFIG3H & 083h) + (CONFIG4L & 085h) + (CONFIG5L & 0Fh) + (CONFIG5H & 0C0h) + (CONFIG6L & 0Fh) + (CONFIG6H & 0E0h) + (CONFIG7L & 0Fh) + (CONFIG7H & 040h) + SUM (IDs)	0EBE6h	0EB9Bh
	Boot Block/ Block 0/ Block 1	SUM (1000:17FFh) + SUM (1800:1FFFh) + (CONFIG1H & 0CFh) + (CONFIG2L & 0Fh) + (CONFIG2H & 1Fh) + (CONFIG3H & 083h) + (CONFIG4L & 085h) + (CONFIG5L & 0Fh) + (CONFIG5H & 0C0h) + (CONFIG6L & 0Fh) + (CONFIG6H & 0E0h) + (CONFIG7L & 0Fh) + (CONFIG7H & 040h) + SUM (IDs)	0F3E4h	0F399h
	Boot Block/ Block 0/ Block 1/ Block 2	SUM (1800:1FFFh) + (CONFIG1H & 0CFh) + (CONFIG2L & 0Fh) + (CONFIG2H & 1Fh) + (CONFIG3H & 083h) + (CONFIG4L & 085h) + (CONFIG5L & 0Fh) + (CONFIG5H & 0C0h) + (CONFIG6L & 0Fh) + (CONFIG6H & 0E0h) + (CONFIG7L & 0Fh) + (CONFIG7H & 040h) + SUM (IDs)	0FBE0h	0FB95h
	Boot Block/ Block 0/ Block 1/ Block 2/ Block 3	(CONFIG1H & 0CFh) + (CONFIG2L & 0Fh) + (CONFIG2H & 1Fh) + (CONFIG3H & 083h) + (CONFIG4L & 085h) + (CONFIG5L & 0Fh) + (CONFIG5H & 0C0h) + (CONFIG6L & 0Fh) + (CONFIG6H & 0E0h) + (CONFIG7L & 0Fh) + (CONFIG7H & 040h) + SUM (IDs)	03D8h	03E2h
	All	(CONFIG1H & 0CFh) + (CONFIG2L & 0Fh) + (CONFIG2H & 1Fh) + (CONFIG3H & 083h) + (CONFIG4L & 085h) + (CONFIG5L & 0Fh) + (CONFIG5H & 0C0h) + (CONFIG6L & 0Fh) + (CONFIG6H & 0E0h) + (CONFIG7L & 0Fh) + (CONFIG7H & 040h) + SUM (IDs)	03D8h	03E2h

Legend:

<u>Item</u>	<u>Description</u>
CONFIG	= Configuration Word
SUM[a:b]	= Sum of locations, a to b inclusive
SUM (IDs)	= Byte-wise sum of lower four bits of all ID locations
+	= Addition
&	= Bit-wise AND

6.0 AC/DC CHARACTERISTICS

TABLE 6-1: TIMING REQUIREMENTS FOR PROGRAM/VERIFY TEST MODE

Standard Operating Conditions Operating Temperature: 25°C is recommended						
Param No.	Sym	Characteristic	Min	Max	Units	Conditions
D110	VIHH	High-Voltage Programming Voltage on $\overline{\text{MCLR}}/\text{VPP}$	9.00	13.25	V	
D110A	VIHL	Low-Voltage Programming Voltage on $\overline{\text{MCLR}}/\text{VPP}$	2.00	5.50	V	
D111	VDD	Supply Voltage during Programming	2.00	5.50	V	Externally timed; row erase and all writes
			4.50	5.50	V	Self-timed; all bulk erases
D112	IPP	Programming Current on $\overline{\text{MCLR}}/\text{VPP}$	—	300	μA	
D113	IDDP	Supply Current during Programming	—	1	mA	
D031	VIL	Input Low Voltage	VSS	0.2 VSS	V	
D041	VIH	Input High Voltage	0.8 VDD	VDD	V	
D080	VOL	Output Low Voltage	—	0.6	V	IOL = 8.5 mA
D090	VOH	Output High Voltage	VDD – 0.7	—	V	IOH = -3.0 mA
D012	CIO	Capacitive Loading on I/O pin (PGD)	—	50	pF	To meet AC specifications
P2	TscLk	Serial Clock (Program Clock, PGC) Period	100	—	ns	VDD = 5.0V
			1	—	μs	VDD = 2.0V
P2A	TscLkL	Serial Clock (Program Clock, PGC) Low Time	40	—	ns	VDD = 5.0V
			400	—	ns	VDD = 2.0V
P2B	TscLkH	Serial Clock (Program Clock, PGC) High Time	40	—	ns	VDD = 5.0V
			400	—	ns	VDD = 2.0V
P3	Tset1	Input Data Setup Time to Serial Clock ↓	15	—	ns	
P4	Thld1	Input Data Hold Time from SCK ↓	15	—	ns	
P5	Tdly1	Delay between 4-bit Command and Command Operand	20	—	ns	
P5A	Tdly1a	Delay between 4-bit Command Operand and Next 4-bit Command	20	—	ns	
P6	Tdly2	Delay between Last SCK ↓ of Command Byte to First SCK ↑ of Read of Data Word	20	—	ns	
P9	Tdly5	SCK High Time (minimum programming time)	1	—	ms	
P10	Tdly6	SCK Low Time after Programming (high-voltage discharge time)	5	—	μs	
P11	Tdly7	Delay to allow Self-Timed Data Write or Bulk Erase to Occur	5	—	ms	
P12	Thld2	Input Data Hold Time from $\overline{\text{MCLR}}/\text{VPP}$ ↑	2	—	μs	
P13	Tset2	VDD ↑ Setup Time to $\overline{\text{MCLR}}/\text{VPP}$ ↑	100	—	ns	
P14	Tvalid	Data Out Valid from SCK ↑	10	—	ns	
P15	Tset3	PGM ↑ Setup Time to $\overline{\text{MCLR}}/\text{VPP}$ ↑	2	—	μs	

PIC18FX220/X320

NOTES:

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

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

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

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

Trademarks

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


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

Analog-for-the-Digital Age, Application Maestro, CodeGuard, dsPICDEM, dsPICDEM.net, dsPICworks, ECAN, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, Linear Active Thermistor, Mindi, MiWi, MPASM, MPLIB, MPLINK, PICkit, PICDEM, PICDEM.net, PICLAB, PICtail, PowerCal, PowerInfo, PowerMate, PowerTool, REAL ICE, rfLAB, rfPICDEM, Select Mode, Smart Serial, SmartTel, Total Endurance, UNI/O, WiperLock and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

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

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

© 2006, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

 Printed on recycled paper.

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 Mountain View, California. The Company's quality system processes and procedures are for its PIC® 8-bit MCUs, KEELoQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.



WORLDWIDE SALES AND SERVICE

AMERICAS

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

Atlanta

Alpharetta, GA
Tel: 770-640-0034
Fax: 770-640-0307

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 - Fuzhou
Tel: 86-591-8750-3506
Fax: 86-591-8750-3521

China - Hong Kong SAR
Tel: 852-2401-1200
Fax: 852-2401-3431

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 - Shunde
Tel: 86-757-2839-5507
Fax: 86-757-2839-5571

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

China - Xian
Tel: 86-29-8833-7250
Fax: 86-29-8833-7256

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 - Gumi
Tel: 82-54-473-4301
Fax: 82-54-473-4302

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

Malaysia - Penang
Tel: 60-4-646-8870
Fax: 60-4-646-5086

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: 43-7242-2244-3910
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