PIC24FXXKA1XX/FVXXKA3XX Flash Programming Specifications

1.0 DEVICE OVERVIEW

This document defines the programming specifications for the PIC24FXXKA1XX/FVXXKA3XX family of 16-bit microcontroller devices. This is required only for developing programming support for the PIC24FXXKA1XX/FVXXKA3XX family. Users of any one of these devices should use the development tools that are already supporting the device programming.

The programming specifications are specific to the following devices:

- PIC24F08KA101
- PIC24F16KA101
- PIC24F08KA102
- PIC24F16KA102
- PIC24FV16KA301(1)
- PIC24FV32KA301(1)
- PIC24FV16KA302(1)
- PIC24FV32KA302(1)
- PIC24FV16KA304(1)
- PIC24FV32KA304(1)

Note 1: Includes corresponding PIC24FXXKA30X devices.

2.0 PROGRAMMING OVERVIEW OF THE PIC24FXXKA1XX/FVXXKA3XX FAMILY

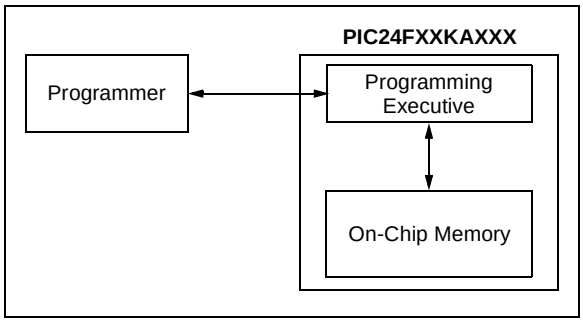
This section describes the two methods of programming the PIC24FXXKA1XX/FVXXKA3XX family of devices:

- In-Circuit Serial Programming™ (ICSP™)
- Enhanced In-Circuit Serial Programming (Enhanced ICSP)

The ICSP programming method is the most direct method for programming the device. However, it is also the slower of the two methods. It provides a native, low-level programming capability to erase, program, and verify the device. **Section 3.0 "Device Programming – ICSP"** describes the ICSP method.

The Enhanced ICSP method is a faster method, which takes advantage of the programming executive, as illustrated in Figure 2-1. The programming executive provides the necessary functionality to erase, program and verify the device through a command set. The command set allows the programmer to program the PIC24FXXKA1XX/FVXXKA3XX devices without having to deal with the low-level programming protocols of the device. **Section 4.0 "Device Programming – Enhanced ICSP"** describes the ICSP method using the programming executive.

FIGURE 2-1: PROGRAMMING SYSTEM OVERVIEW FOR ENHANCED ICSP™ METHOD



2.1 Power Requirements

Devices in the PIC24FXXKA1XX and PIC24FXXKA3XX families are 3.3V supply designs. The core, the peripherals and the I/O pins operate at 3.3V. The device can operate from 1.8V to 3.6V.

Devices in the PIC24FVXXKA3XX families are 5.0V supply designs. The core, the peripherals and the I/O pins operate at 3.3V. The device can operate from 2.0V to 5.5V; an internal regulator operates the core logic at 3.25V.

Table 2-1 provides the pins that are required for programming, which are indicated in Figure 2-3. Refer to the appropriate device data sheet for complete pin descriptions. Note that all power supply and ground pins must be connected appropriately for programming.

TABLE 2-1: PIN DESCRIPTIONS (DURING PROGRAMMING)

Pin Name	During Programming		
	Pin Name	Pin Type	Pin Description
MCLR/VPP	MCLR/VPP	P	Programming Enable
VDD	VDD	P	Power Supply
VSS	VSS	P	Ground
VCAP	VCAP	P	Stabilizing Capacitor for Voltage Regulator(1)
PGECx	PGC	I	Programming Pin Pair: Serial Clock
PGEDx	PGD	I/O	Programming Pin Pair: Serial Data

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

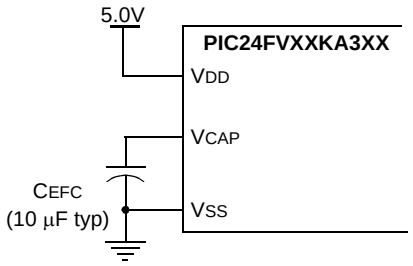
Note 1: PIC24FVXXKA3XX devices only.

2.1.1 ON-CHIP VOLTAGE REGULATOR CONNECTIONS

For PIC24FVXXKA3XX devices, an on-chip regulator provides power to the core from the other VDD pins. A low ESR capacitor (such as high-quality ceramic or tantalum) must be connected to the VDDCORE pin (Figure 2-2). This helps to maintain the stability of the regulator. The specifications for core voltage and capacitance are listed in **Section 7.0 "AC/DC Characteristics and Timing Requirements"**.

PIC24FXXKA1XX and PIC24FXXKA3XX devices do not use an on-chip regulator.

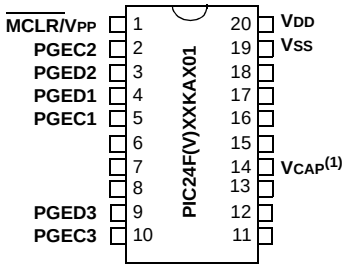
FIGURE 2-2: CONNECTIONS FOR THE ON-CHIP REGULATOR



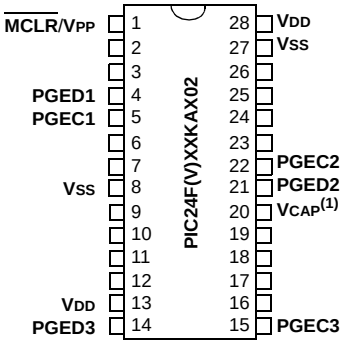
Note 1: These are typical operating voltages. Refer to **Section 7.0 "AC/DC Characteristics and Timing Requirements"** for the full operating ranges of VDD and VDDCORE.

FIGURE 2-3: PIN DIAGRAMS

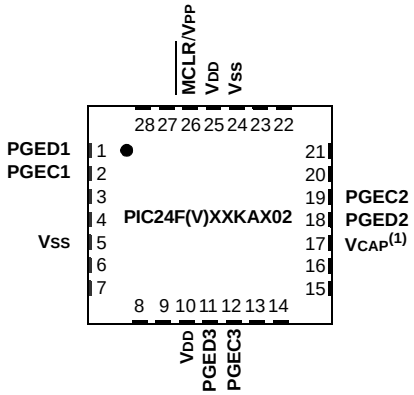
20-Pin SPDIP, SOIC



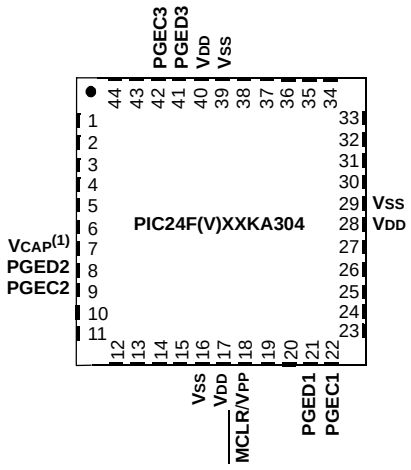
28-Pin PDIP, SOIC



28-Pin QFN



44-Pin QFN



Note 1: PIC24FVXXKA3XX devices only. PIC24FXXKA3XX devices do not use VCAP.

2.2 Memory Map

The program memory map for PIC24FXXKA1XX/FVXXKA3XX devices extends from 000000h to FFFFFEh. Code storage is located at the base of the memory map and supports up to 11K instruction words (about 32 Kbytes).

Additionally, PIC24FXXKA1XX/FVXXKA3XX family devices have an on-chip data EEPROM. This data EEPROM is mapped to the program memory area from location 7FFE00h to 7FFFFEh.

Table 2-2 provides the program memory and data EEPROM size, and the number of memory rows present in each device variant.

The erase operation can be done on one word, half of a row or one row at a time. The program operation can be done only one word at a time.

TABLE 2-2: MEMORY SIZES FOR PIC24FXXKA1XX/FVXXKA3XX DEVICES

Device	Program Memory Upper Address (Instruction Words)	Rows	Data EEPROM Size (Words)	Rows
PIC24F08KA10X	15FEh (2.75K)	88	256	32
PIC24F16KA10X	2BFEh (5.5K)	176	256	32
PIC24FV08KA30X(1)	15FEh (2.75K)	88	256	32
PIC24FV16KA30X(1)	2BFEh (5.5K)	176	256	32
PIC24FV32KA30X(1)	57FEh (11K)	352	256	32

Note 1: Includes corresponding PIC24FXXKA3XX devices.

Locations 800000h through 8007FEh are reserved for executive code memory. This region stores the programming executive, the debugging executive and the Diagnostic Words. The programming executive is used for device programming and the debug executive is used for in-circuit debugging. This region of memory cannot be used to store user code.

The device Configuration registers are implemented from location F80000h to F80010h, and can be erased or programmed one register at a time. Table 2-3 provides the implemented Configuration registers and their locations.

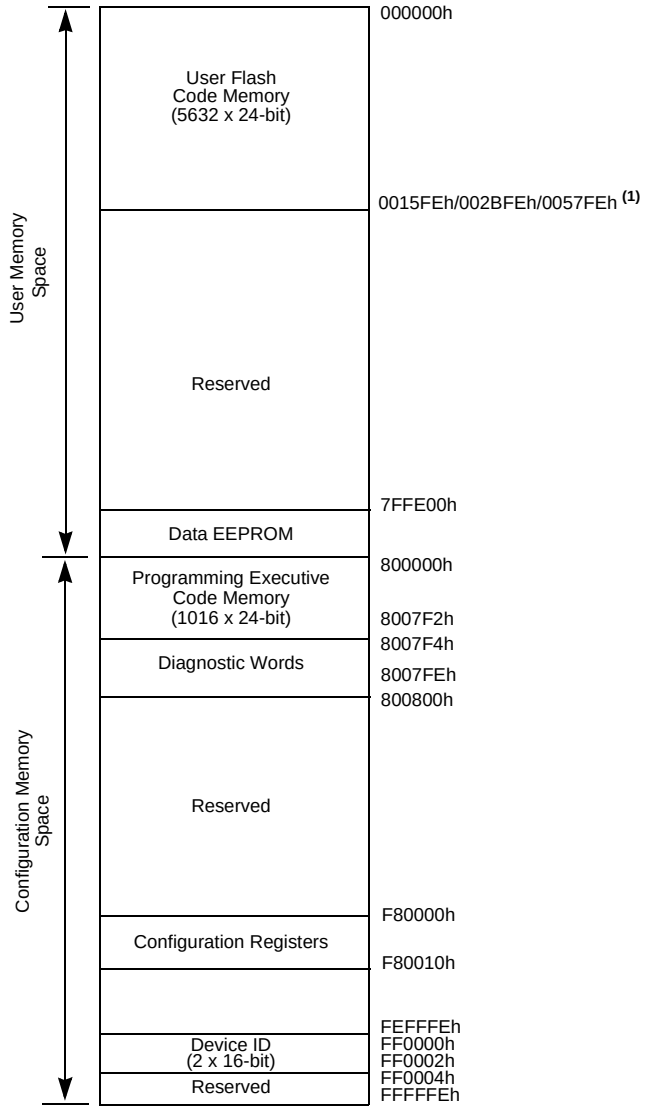
Locations, FF0000h and FF0002h, are reserved for the Device ID registers. These bits can be used by the programmer to identify the device type that is being programmed. See **Section 6.0 “Device ID”** for more information. The Device ID registers read out normally even after code protection is applied.

Figure 2-4 depicts the memory map for the PIC24FXXKA1XX/FVXXKA3XX family variants.

TABLE 2-3: CONFIGURATION REGISTER LOCATIONS

Configuration Register	Address
FBS	F80000
FGS	F80004
FOSCSEL	F80006
FOSC	F80008
FWDT	F8000A
FPOR	F8000C
FICD	F8000E
FDS	F80010

FIGURE 2-4: PROGRAM MEMORY MAP



Note 1: The upper address boundaries for user program memory are device-specific. See Table 2-2 for details.

3.0 DEVICE PROGRAMMING – ICSP

The ICSP method is a special programming protocol that allows reading and writing to the PIC24FXXKA1XX/FVXXKA3XX device family memory. ICSP is the most direct method used to program a device; however, Enhanced ICSP is faster. The ICSP mode also reads the contents of the executive memory to determine if the programming executive is present. This is accomplished by applying control codes and instructions, serially to the device, using PGCx and PGDx pins.

In ICSP mode, the system clock is taken from the PGCx pin regardless of the device's oscillator Configuration bits. All of the instructions are shifted serially to an internal buffer, loaded into the Instruction Register (IR), and then executed. No program is fetched from the internal memory. Instructions are fed in 24 bits at a time. PGDx is used to shift data in, and PGCx is used as both the serial shift clock and the CPU execution clock.

Note:	During ICSP operation, the operating frequency of PGCx should not exceed 8 MHz.

3.1 Overview of the Programming Process

Figure 3-1 illustrates the high-level overview of the programming process.

After entering the ICSP mode, perform the following:

1. Bulk Erase the device.
2. Program and verify the code memory.
3. Program and verify the data EEPROM memory.
4. Program and verify the device configuration.
5. Program the code-protect Configuration bits, if required.

3.2 ICSP Operation

Upon entry into ICSP mode, the CPU is Idle. An internal state machine governs the execution of the CPU. A 4-bit control code is clocked in, using PGCx and PGDx, and this control code is used to command the CPU (see Table 3-1).

The SIX control code is used to send instructions to the CPU for execution, and the REGOUT control code is used to read data out of the device via the VISI register.

FIGURE 3-1: HIGH–LEVEL ICSP™ PROGRAMMING FLOW

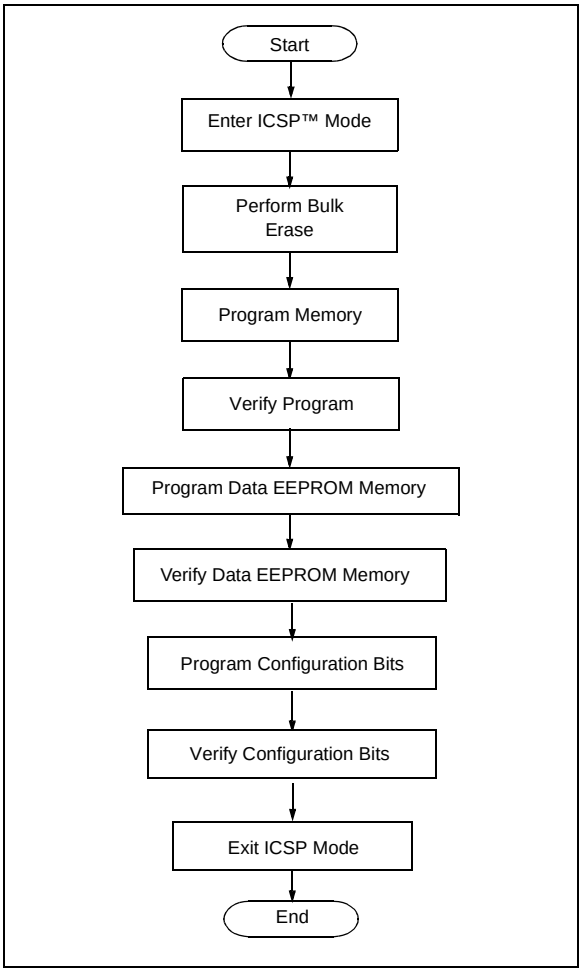


TABLE 3-1: CPU CONTROL CODES IN ICSP™ MODE

4-Bit Control Code	Mnemonic	Description
0000b	SIX	Shift in 24-bit instruction and execute.
0001b	REGOUT	Shift out the VISI (0784h) register.
0010b-1111b	N/A	This is reserved.

3.2.1 SIX SERIAL INSTRUCTION EXECUTION

The SIX control code allows execution of PIC24FXXKA1XX/FVXXKA3XX family assembly instructions. When the SIX code is received, the CPU is suspended for 24 clock cycles as the instruction is then clocked into the internal buffer. Once the instruction is shifted in, the state machine allows it to be executed over the next four PGC clock cycles. While the received instruction is executed, the state machine simultaneously shifts in the next 4-bit command (see Figure 3-2).

Coming out of Reset, the first 4-bit control code is always forced to SIX, and a forced NOP instruction is executed by the CPU. Five additional PGCx clocks are needed on start-up; thereby resulting in a 9-bit SIX command, instead of the normal 4-bit SIX command.

After the forced SIX is clocked in, the ICSP operation resumes to normal. That is, the next 24 clock cycles load the first instruction word to the CPU.

Note:	To account for this forced NOP, all example codes in this specification begin with a NOP to ensure that no data is lost.

3.2.1.1 Differences Between SIX Instruction Execution and Normal Instruction Execution

There are some differences between executing instructions using the SIX ICSP command and normal device instruction execution. As a result, the code examples in this specification might not match those required to perform the same operations during normal device operation.

The differences are:

- Two-word instructions require 2 SIX operations to clock in all the necessary data.

  Examples of two-word instructions are GOTO and CALL.

- Two-cycle instructions require 2 SIX operations to complete. The first SIX operation shifts in the instruction and begins to execute it. A second SIX operation, which should shift in a NOP to avoid losing data, allows the CPU clocks required to finish executing the instruction.

  Examples of two-cycle instructions are table read and table write instructions.

- The CPU does not automatically stall to account for pipeline changes. A CPU stall occurs when an instruction modifies a register, which is used by the instruction immediately following the CPU stall for Indirect Addressing. During normal operation, the CPU forces a NOP while the new data is read. To account for this, while using ICSP, any indirect references to a recently modified register should be proceeded with a NOP.

  For example, `MOV #0x0,W0` followed by `MOV [W0],W1` must have a NOP inserted in between.

  If a two-cycle instruction modifies a register, which is used indirectly, it requires two following NOPs; one to execute the second half of the instruction and the other to stall the CPU to correct the pipeline.

  For example, `TBLWTL [W0++],[W1]` should be followed by 2 NOPs.

- The device Program Counter (PC) continues to automatically increment during the ICSP instruction execution, even though the Flash memory is not being used. As a result, it is possible for the PC to be incremented so that it points to invalid memory locations.

  Examples of invalid memory spaces are unimplemented Flash addresses or the vector space (location 0x0 to 0x1FF).

  If the PC ever points to these locations, it causes the device to reset, possibly interrupting the ICSP operation. To prevent this, instructions should be periodically executed to reset the PC to a safe space. The optimal method of achieving this is to perform a "`GOTO 0x200`".

3.2.2 REGOUT SERIAL INSTRUCTION EXECUTION

The REGOUT control code allows for the data to be extracted from the device in the ICSP mode. It is used to clock the contents of the VISI register out of the device over the PGDx pin. After the REGOUT control code is received, the CPU is held Idle for 8 cycles. After this, an additional 16 cycles are required to clock the data out (see Figure 3-3).

The REGOUT code is unique, as the PGDx pin is an input when the control code is transmitted to the device. However, after the control code is processed, the PGDx pin becomes an output as the VISI register is shifted out.

Note 1:	After the contents of VISI are shifted out, the PIC24FXXKA1XX/FVXXKA3XX devices maintain PGDx as an output until the first rising edge of the next clock is received.
2:	Data changes on the falling edge and latches on the rising edge of PGCx. For all data transmissions, the Least Significant bit (LSb) is transmitted first.

FIGURE 3-2: SIX SERIAL EXECUTION

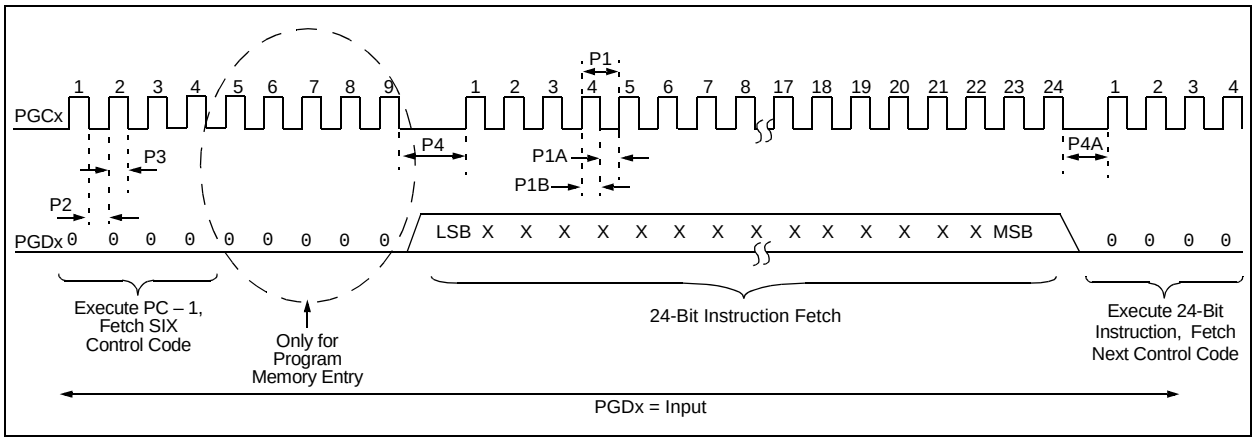
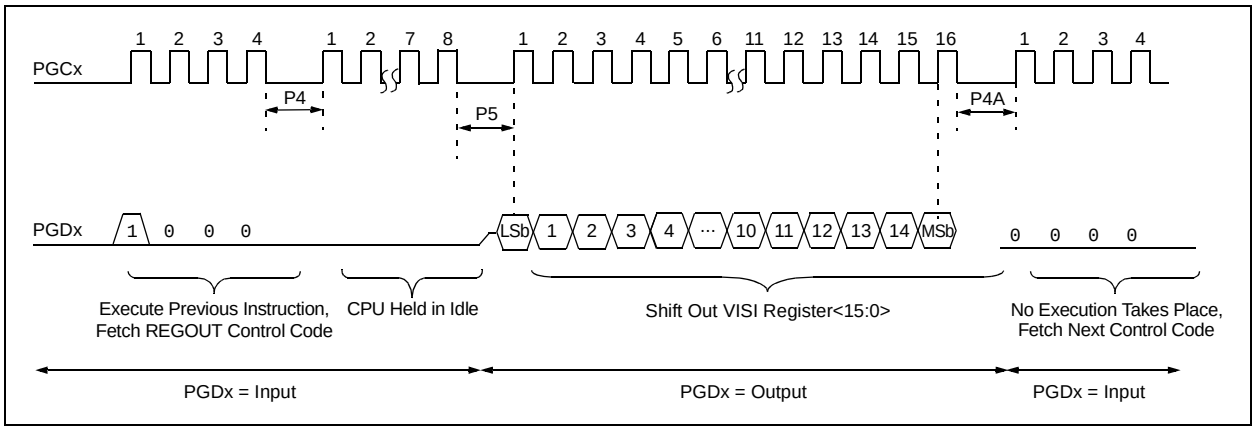


FIGURE 3-3: REGOUT SERIAL EXECUTION



3.3 Entering ICSP Mode

3.3.1 LOW-VOLTAGE ICSP ENTRY

As illustrated in Figure 3-4, the following processes are involved in entering ICSP Program/Verify mode using MCLR:

1. MCLR is briefly driven high, then low.
2. A 32-bit key sequence is clocked into PGDx.
3. MCLR is then driven high within a specified period of time and held.

The programming voltage VIH IS applied to MCLR; this is VDD in the case of PIC24FXXKA1XX/FVXXKA3XX devices. There is no minimum time requirement for holding at VIH. After VIH is removed, an interval of at least P18 must elapse before presenting the key sequence on PGDx.

The key sequence is a specific 32-bit pattern: '0100 1101 0100 0011 0100 1000 0101 0001' (more easily remembered as 4D434851h in hexadecimal). The device will enter Program/Verify mode only if the sequence is valid. The Most Significant bit (MSb) of the most significant nibble must be shifted in first.

Once the key sequence is complete, VIH must be applied to MCLR and held at that level for as long as the Program/Verify mode is to be maintained. An interval of at least P19 and P7 must elapse before presenting data on PGDx. Signals appearing on PGCx before P7 has elapsed would not be interpreted as valid.

3.3.2 HIGH-VOLTAGE ICSP ENTRY

Entering the ICSP Program/Verify mode, using the VPP pin, is the same as entering the mode using MCLR. The only difference is the programming voltage applied to VPP is VIHH, and before presenting the key sequence on PGDx, an interval of at least P18 should elapse (see Figure 3-5).

Once the key sequence is complete, an interval of at least P7 should elapse, and the voltage should remain at VIHH. The voltage, VIHH, must be held at that level for as long as the Program/Verify mode is to be maintained. An interval of at least P7 must elapse before presenting the data on PGDx.

Signals appearing on PGDx before P7 has elapsed will not be interpreted as valid.

Upon a successful entry, the program memory can be accessed and programmed in serial fashion. While in ICSP mode, all unused I/Os are placed in a high-impedance state.

3.3.3 CODE-PROTECT ICSP ENTRY

When code protection is employed on the PIC24FXXKA3XX devices (BWRP, GSS0 or GWRP = 0), then the voltage on VDD must be above VBULK in order to erase, and then program the device. Care must be taken in the design and layout of a board so that any parts connected to VDD can withstand what may be an increase in voltage if the device is running below VBULK.

FIGURE 3-4: ENTERING ICSP™ MODE USING LOW-VOLTAGE ENTRY

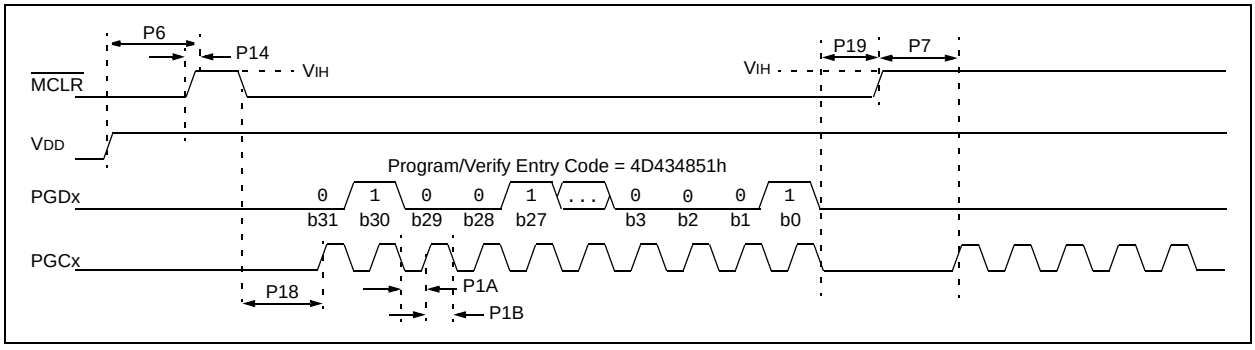
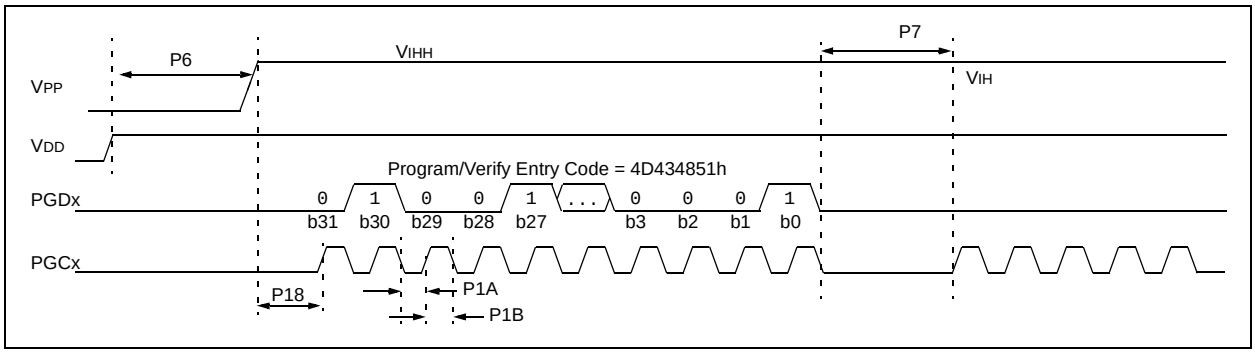


FIGURE 3-5: ENTERING ICSP™ MODE USING HIGH-VOLTAGE ENTRY



3.4 Flash Memory Programming in ICSP Mode

3.4.1 PROGRAMMING OPERATIONS

The NVMCON register controls the Flash memory write and erase operations. To program the device, set the NVMCON register to select the type of erase operation (see Table 3-2) or write operation (see Table 3-3). Set the WR control bit (NVMCON<15>) to initiate the program.

In ICSP mode, all programming operations are self-timed. There is an internal delay between setting and automatic clearing of the WR control bit when the programming operation is complete. Refer to **Section 7.0 "AC/DC Characteristics and Timing Requirements"** for information on the delays associated with various programming operations.

3.4.2 STARTING AND STOPPING A PROGRAMMING CYCLE

The WR bit (NVMCON<15>) is used to start an erase or write cycle. Initiate the programming cycle by setting the WR bit.

All erase and write cycles are self-timed. The WR bit should be polled to determine if the erase or write cycle is complete. Start a programming cycle as follows:

```
BSET    NVMCON, #WR
```

TABLE 3-2: NVMCON VALUES FOR ERASE OPERATIONS

NVMCON Value	Erase Operation
4064h	Erase the code memory and Configuration registers (does not erase programming executive code and Device ID registers).
404Ch	Erase the general segment and Configuration bits associated with it.
4068h	Erase the boot segment and Configuration bits associated with it.
405Ah(1)	Erase four rows of code memory.
4059h(1)	Erase two rows of code memory.
4058h(1)	Erase a row of code memory.
4050h	Erase the entire data EEPROM memory and Configuration bits associated with it.
405Ah(1)	Erase eight words of data EEPROM memory.
4059h(1)	Erase four words of data EEPROM memory.
4058h(1)	Erase one word of data EEPROM memory.
4054h	Erase all the Configuration registers (except the code-protect fuses).
4058h(1)	Erase Configuration registers except FBS and FGS.

Note 1: The destination address decides the region (code memory, data EEPROM memory or Configuration register) of the erased rows/words.

TABLE 3-3: NVMCON VALUES FOR WRITE OPERATIONS

NVMCON Value	Write Operation
4004h(1)	Write one Configuration register.
4004h(1)	Program one row (32 instruction words) of code memory or executive memory.
4004h(1)	Program one word of data EEPROM memory.

Note 1: The destination address decides the region (code memory, data EEPROM memory or Configuration register) of the erased rows/words.

3.5 Erasing Program Memory

To erase the program memory (all of code memory, data memory and Configuration bits, including the code-protect bits), set the NVMCON to 4064h and then execute the programming cycle.

Figure 3-6 illustrates the ICSP programming process for Bulk Erase. This process includes the ICSP command code, which must be transmitted (for each instruction), LSB first, using the PGCx and PGDx pins (see Figure 3-2).

Table 3-4 provides the steps for executing serial instruction for the Bulk Erase mode.

Note:	Program memory must be erased before writing any data to program memory.

FIGURE 3-6: BULK ERASE FLOW

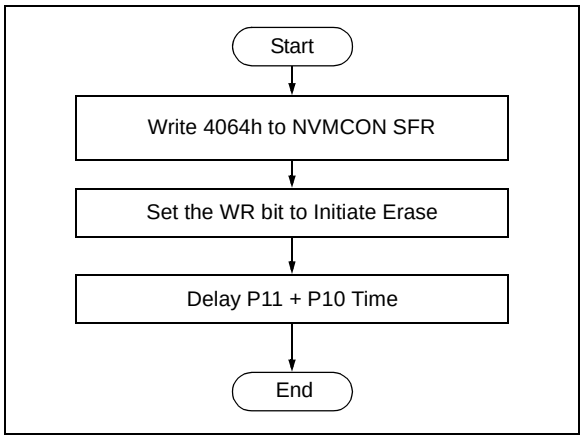


TABLE 3-4: SERIAL INSTRUCTION EXECUTION FOR CHIP ERASE

Command (Binary)	Data (Hex)	Description
Step 1: Exit the Reset vector.		
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
Step 2: Set the NVMCON to erase the entire program memory.		
0000	24064A	MOV #0x4064, W10
0000	883B0A	MOV W10, NVMCON
Step 3: Set the TBLPAG and perform dummy table write to select the erased memory.		
0000	200000	MOV #<PAGEVAL>, W0
0000	880190	MOV W0, TBLPAG
0000	200000	MOV #0x0000, W0
0000	BB0800	TBLWTL W0, [W0]
0000	000000	NOP
0000	000000	NOP
Step 4: Initiate the erase cycle.		
0000	A8E761	BSET NVMCON, #WR
0000	000000	NOP
0000	000000	NOP
Step 5: Repeat this step to poll the WR bit (bit 15 of NVMCON) until it is cleared by the hardware.		
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
0000	803B02	MOV NVMCON, W2
0000	883C22	MOV W2, VISI
0000	000000	NOP
0001	<VISI>	Clock out the contents of the VISI register.
0000	000000	NOP

3.6 Writing Code Memory

The procedure for writing code memory is the same as writing the Configuration registers. The difference is that the 32 instruction words are programmed one at a time. To facilitate this operation, working registers, W0:W5, are used as temporary holding registers for the data to be programmed. Figure 3-8 illustrates the code memory writing flow.

Table 3-5 provides the ICSP programming details, including the serial pattern with the ICSP command code, which must be transmitted LSB first, using the PGCx and PGDx pins (see Figure 3-2).

In Step 1 of Table 3-5, the Reset vector is exited; in Step 2, the NVMCON register is initialized for programming a full row of code memory, and in Step 3, the 24-bit starting destination address for programming is loaded into the TBLPAG register and W7 register. The upper byte of the starting destination address is stored in TBLPAG and the lower 16 bits of the destination address are stored in W7.

To minimize the programming time, a packed instruction format is used (see Figure 3-7).

In Step 4 of Table 3-5, four packed instruction words are stored in working registers, W0:W5, using the MOV instruction. The Read Pointer, W6, is initialized. Figure 3-7 illustrates the contents of W0:W5 holding the packed instruction word data. In Step 5, eight TBLWT instructions are used to copy the data from W0:W5 to the write latches of the code memory. Since code memory is programmed 32 instruction words at a time, Steps 3 to 5 are repeated eight times to load all the write latches (see Step 6).

After the write latches are loaded, initiate programming by writing to the NVMCON register in Steps 7 and 8. In Step 9, the internal PC is reset to 200h. This is a precautionary measure to prevent the PC from incrementing to unimplemented memory when large devices are being programmed. Finally, in Step 10, repeat Steps 3 through 9 until all of the code memory is programmed.

FIGURE 3-7: PACKED INSTRUCTION WORDS IN W0:W5

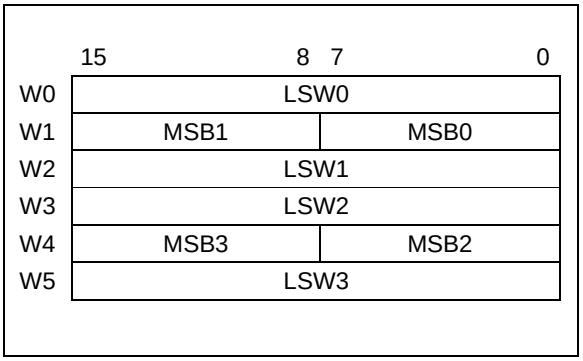


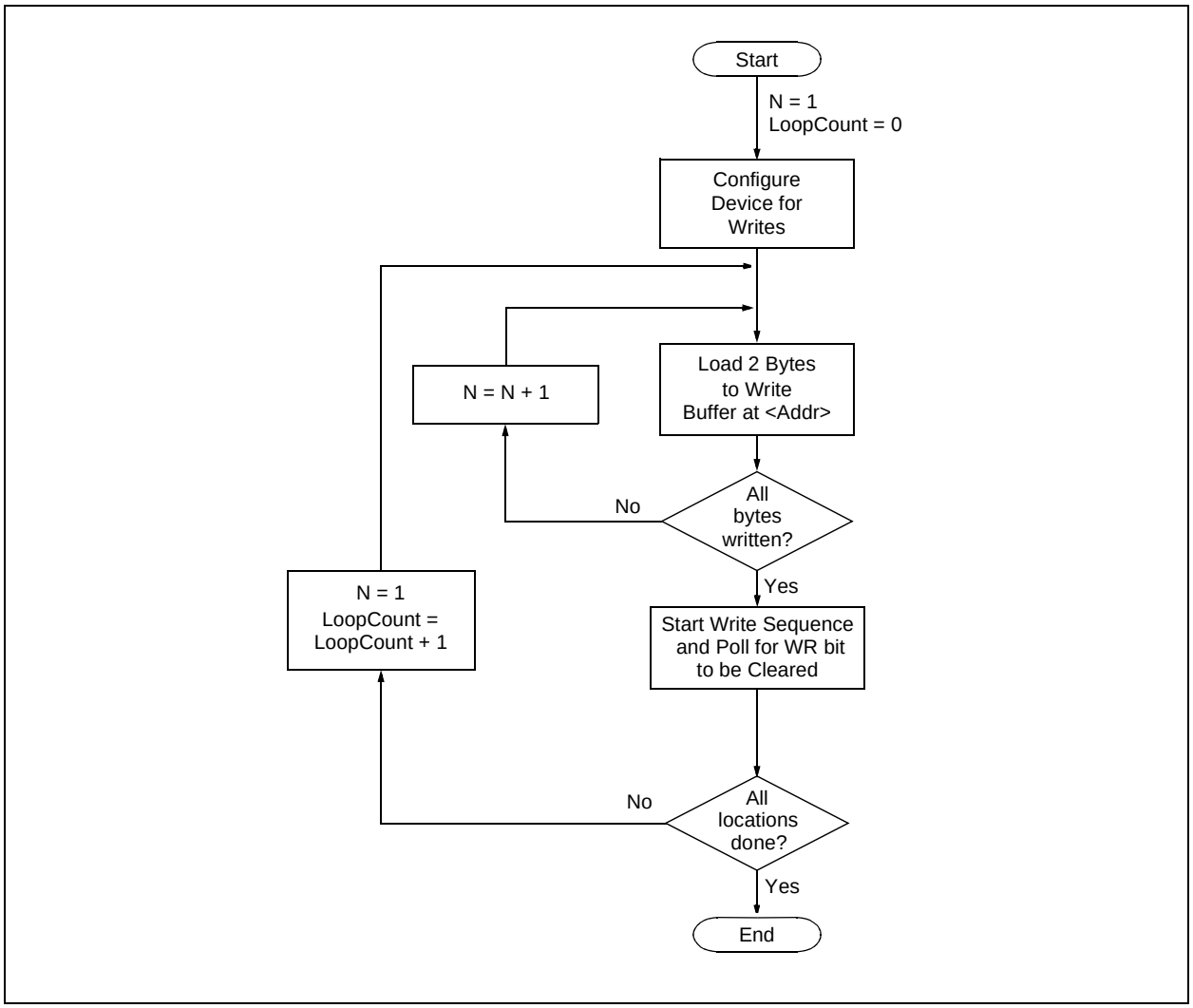
TABLE 3-5: SERIAL INSTRUCTION EXECUTION FOR WRITING CODE MEMORY

Command (Binary)	Data (Hex)	Description
Step 1: Exit the Reset vector.		
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
Step 2: Set the NVMCON to program 32 instruction words.		
0000	24004A	MOV #0x4004, W10
0000	883B0A	MOV W10, NVMCON
Step 3: Initialize the Write Pointer (W7) for TBLWT instruction.		
0000	200xx0	MOV #<DestinationAddress23:16>, W0
0000	880190	MOV W0, TBLPAG
0000	2xxxx7	MOV #<DestinationAddress15:0>, W7
Step 4: Load W0:W5 with the next 4 instruction words to program.		
0000	2xxxx0	MOV #<LSW0>, W0
0000	2xxxx1	MOV #<MSB1:MSB0>, W1
0000	2xxxx2	MOV #<LSW1>, W2
0000	2xxxx3	MOV #<LSW2>, W3
0000	2xxxx4	MOV #<MSB3:MSB2>, W4
0000	2xxxx5	MOV #<LSW3>, W5

TABLE 3-5: SERIAL INSTRUCTION EXECUTION FOR WRITING CODE MEMORY (CONTINUED)

Command (Binary)	Data (Hex)	Description
Step 5: Set the Read Pointer (W6) and load the (next set of) write latches.		
0000	EB0300	CLR W6
0000	000000	NOP
0000	BB0BB6	TBLWTL [W6++], [W7]
0000	000000	NOP
0000	000000	NOP
0000	BBDBB6	TBLWTH.B [W6++], [W7++]
0000	000000	NOP
0000	000000	NOP
0000	BBEBB6	TBLWTH.B [W6++], [++W7]
0000	000000	NOP
0000	000000	NOP
0000	BB1BB6	TBLWTL [W6++], [W7++]
0000	000000	NOP
0000	000000	NOP
0000	BB0BB6	TBLWTL [W6++], [W7]
0000	000000	NOP
0000	000000	NOP
0000	BBDBB6	TBLWTH.B [W6++], [W7++]
0000	000000	NOP
0000	000000	NOP
0000	BBEBB6	TBLWTH.B [W6++], [++W7]
0000	000000	NOP
0000	000000	NOP
0000	BB1BB6	TBLWTL [W6++], [W7++]
0000	000000	NOP
0000	000000	NOP
Step 6: Repeat Steps 3 though 5, eight times to load the write latches for 32 instructions.		
Step 7: Initiate the write cycle.		
0000	A8E761	BSET NVMCON, #WR
0000	000000	NOP
0000	000000	NOP
Step 8: Repeat this step to poll the WR bit (bit 15 of NVMCON) until it is cleared by the hardware.		
0000	040200	GOTO 0x200
0000	000000	NOP
0000	803B02	MOV NVMCON, W2
0000	883C22	MOV W2, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of the VISI register.
0000	000000	NOP
Step 9: Reset device internal PC.		
0000	040200	GOTO 0x200
0000	000000	NOP
Step 10: Repeat Steps 3 through 9 until the entire code memory is programmed.		

FIGURE 3-8: PROGRAM CODE MEMORY FLOW



3.7 Writing Data EEPROM

Figure 3-9 illustrates the flow of programming the data EEPROM memory. The procedure is the same as writing code memory. The only difference is that only one word is programmed in each operation. When writing data EEPROM, one word is programmed during each operation. Working register, W0, is used as a temporary holding register for the data to be programmed.

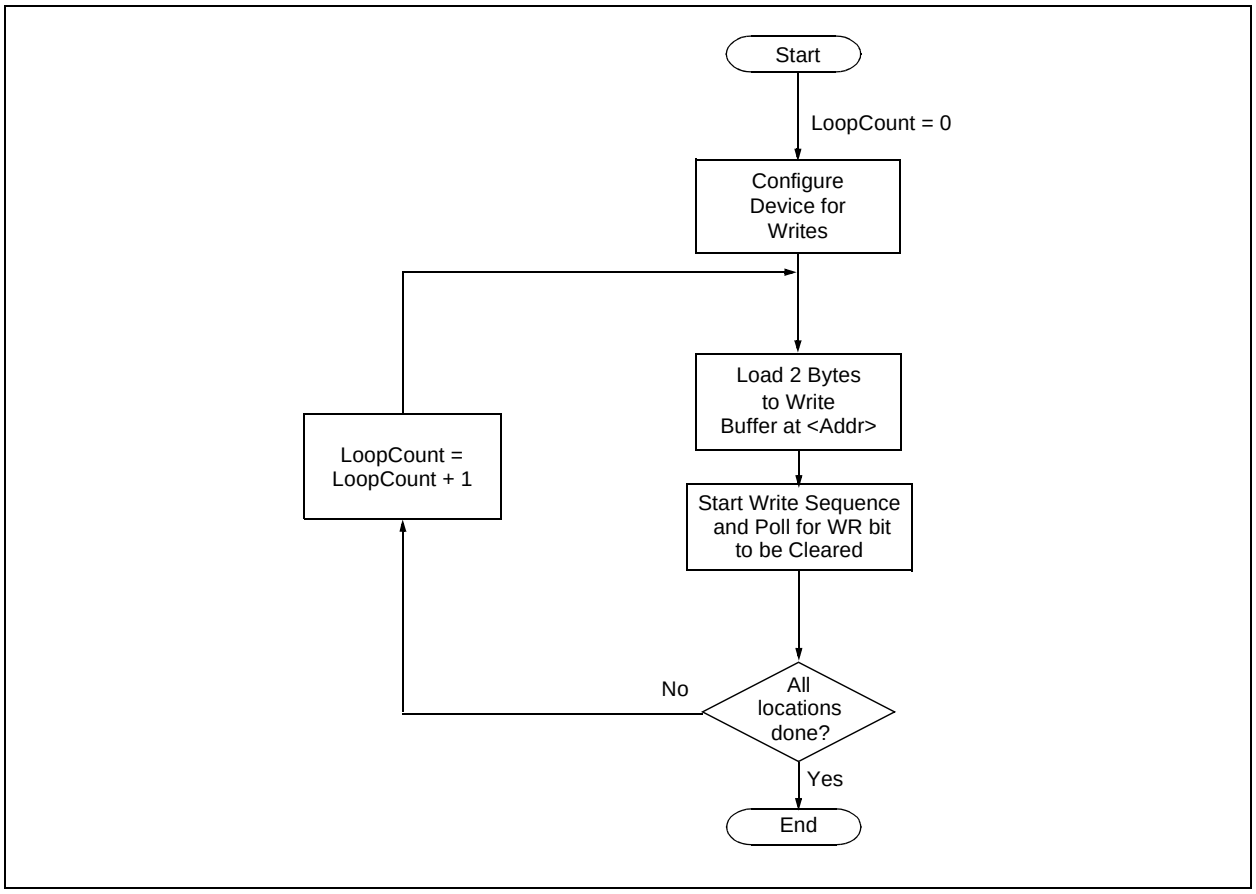
Table 3-6 provides the ICSP programming details for writing data EEPROM.

Note:	When writing to EEPROM, always set the TBLPAG register to 7Fh. This is the upper byte address of all locations of data EEPROM.

TABLE 3-6: INSTRUCTION EXECUTION FOR WRITING DATA EEPROM

Command (Binary)	Data (Hex)	Description
Step 1: Exit the Reset vector.		
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
Step 2: Set the NVMCON to program 1 data word.		
0000	24004A	MOV #0x4004, W10
0000	883B0A	MOV W10, NVMCON
Step 3: Initialize the Write Pointer (W7) for TBLWT instruction.		
0000	2007F0	MOV #0x7F, W0
0000	880190	MOV W0, TBLPAG
0000	2FExx7	MOV #<DestinationAddress15:0>, W7
Step 4: Load W0 with the data word to program and load the write latch.		
0000	2xxxx0	MOV #<Data_Word_Value>, W0
0000	BB1B80	TBLWTL W0, [W7++]
0000	000000	NOP
0000	000000	NOP
Step 5: Initiate the write cycle.		
0000	A8E761	BSET NVMCON, #WR
0000	000000	NOP
0000	000000	NOP
Step 6: Repeat this step to poll the WR bit (bit 15 of NVMCON) until it is cleared by the hardware.		
0000	040200	GOTO 0x200
0000	000000	NOP
0000	803B02	MOV NVMCON, W2
0000	883C22	MOV W2, VISI
0000	000000	NOP
00001	<VISI>	Clock out contents of the VISI register.
0000	000000	NOP
Step 7: Reset device internal PC.		
0000	040200	GOTO 0x200
0000	000000	NOP
Step 8: Repeat Steps 4 through 7 until the entire data EEPROM memory is programmed.		

FIGURE 3-9: PROGRAM DATA EEPROM MEMORY FLOW



3.8 Writing Configuration Registers

The procedure for writing the Configuration registers is the same as for writing code memory. The only difference is that only one word is programmed in each operation. When writing Configuration registers, one word is programmed during each operation; only working register, W0, is used as a temporary holding register for the data to be programmed.

Table 3-7 provides the default values of the Configuration registers.

Note:	When writing to the Configuration registers, always set the TBLPAG register to F8h. This is the upper byte address of all locations of the Configuration registers.

Table 3-7 provides the ICSP programming details for programming the Configuration registers, including the serial pattern with the ICSP command code, which must be transmitted LSB first using the PGCx and PGDx pins (see Figure 3-2). In Step 1 of Table 3-8, the Reset vector is exited. In Step 2, the NVMCON register is initialized for programming code memory. In Step 3, the 24-bit starting destination address for programming is loaded into the TBLPAG register and W7 register.

TABLE 3-7: DEFAULT VALUES FOR CONFIGURATION REGISTER SERIAL INSTRUCTION

Configuration Registers	Value
FBS	0Fh
FGS	03h
FOSCSEL	87h
FOSC	FFh
FWDT	DFh
FPOR(1)	FBh
FICD(2)	C3h
FDS	FFh

Note 1: The I2C2SEL bit (FPOR<4>) is not implemented on PIC24FXXKAX01 devices and should be programmed as ‘1’.

2: The FICD<6> bit is a reserved bit and should be programmed as ‘1’.

TABLE 3-8: SERIAL INSTRUCTION EXECUTION FOR WRITING CONFIGURATION REGISTERS

Command (Binary)	Data (Hex)	Command (Binary)
Step 1: Exit the Reset vector.		
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
Step 2: Initialize the Write Pointer (W7) for the TBLWT instruction.		
0000	200007	MOV #0x0000, W7
Step 3: Set the NVMCON register to program Configuration registers.		
0000	24004A	MOV #0x4004, W10
0000	883B0A	MOV W10, NVMCON
Step 4: Initialize the TBLPAG register.		
0000	200F80	MOV #0xF8, W6
0000	880190	MOV W0, TBLPAG
Step 5: Load the Configuration register data to W6.		
0000	2xxxx6	MOV #<FBS_VALUE>, W6
Step 6: Write the Configuration register data to the write latch and increment the Write Pointer.		
0000	000000	NOP
0000	BB1B86	TBLWTL W6, [W7++]
0000	000000	NOP
0000	000000	NOP
Step 7: Initiate the write cycle.		
0000	A8E761	BSET NVMCON, #WR
0000	000000	NOP
0000	000000	NOP
Step 8: Repeat this step to poll the WR bit (bit 15 of NVMCON) until it is cleared by the hardware.		
0000	040200	GOTO 0x200
0000	000000	NOP
0000	803B02	MOV NVMCON, W2
0000	883C22	MOV W2, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of the VISI register.
0000	000000	NOP
Step 9: Reset device internal PC.		
0000	040200	GOTO 0x200
0000	000000	NOP
Step 10: Repeat Steps 5 through 9 to write other fuses, Load W6 with their respective values and W7 with their respective addresses.		

3.9 Reading Code Memory

To read the code memory, execute a series of TBLRD instructions and clock out the data using the REGOUT command.

Table 3-9 provides the ICSP programming details for reading code memory. In Step 1, the Reset vector is exited. In Step 2, the 24-bit starting source address for reading is loaded into the TBLPAG register and the W6 register. The upper byte of the starting source address is stored in TBLPAG, and the lower 16 bits of the source address are stored in W6.

To minimize the reading time, the packed instruction word format, which was used for writing, is also used for reading (see Figure 3-7). In Step 3, the Write Pointer, W7, is initialized. In Step 4, two instruction words are read from code memory, and clocked out of the device through the VISI register, using the REGOUT command. Step 4 is repeated until the required amount of code memory is read.

TABLE 3-9: SERIAL INSTRUCTION EXECUTION FOR READING CODE MEMORY

Command (Binary)	Data (Hex)	Description
Step 1: Exit Reset vector.		
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
Step 2: Initialize TBLPAG and the Read Pointer (W6) for TBLRD instruction.		
0000	200xx0	MOV #<SourceAddress23:16>, W0
0000	880190	MOV W0, TBLPAG
0000	2xxxx6	MOV #<SourceAddress15:0>, W6
Step 3: Initialize the Write Pointer (W7) to point to the VISI register.		
0000	207847	MOV #VISI, W7
0000	000000	NOP
Step 4: Read and clock out the contents of the next two locations of code memory through the VISI register using the REGOUT command.		
0000	BA1B96	TBLRDL [W6], [W7]
0000	000000	NOP
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
0000	BADBB6	TBLRDH [W6++], [W7]
0000	000000	NOP
0000	000000	NOP
0000	BAD3D6	TBLRDH.B [++W6], [W7--]
0000	000000	NOP
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
0000	BA0BB6	TBLRDL [W6++], [W7]
0000	000000	NOP
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
Step 5: Reset device internal PC.		
0000	040200	GOTO 0x200
0000	000000	NOP
Step 6: Repeat Steps 4 and 5 until the required code memory is read.		

3.10 Reading Data EEPROM Memory

The procedure for reading data EEPROM memory is the same as reading the code memory. The only difference is that the 16-bit data words are read instead of the 24-bit words.

Table 3-10 provides the ICSP programming details for reading data memory.

Note: When reading from EEPROM, always set the TBLPAG register to 7Fh. This is the upper byte address of all locations of data EEPROM.

TABLE 3-10: SERIAL INSTRUCTION EXECUTION FOR READING DATA EEPROM MEMORY

Command (Binary)	Data (Hex)	Description
Step 1: Exit Reset vector.		
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
Step 2: Initialize TBLPAG and the Read Pointer (W6) for TBLRD instruction.		
0000	2007F0	MOV #0x7F, W0
0000	880190	MOV W0, TBLPAG
0000	2FExx6h	MOV #<SourceAddress15:0>, W6;(FExx)
Step 3: Initialize the Write Pointer (W7) to point to the VISI register.		
0000	207847	MOV #VISI, W7
0000	000000	NOP
Step 4: Read and clock out the contents of the next location of data EEPROM memory through the VISI register using the REGOUT command.		
0000	BA1B96	TBLRDL [W6++], [W7]
0000	000000	NOP
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
Step 5: Repeat Step 4 until the required data EEPROM memory is read.		
Step 6: Reset device internal PC.		
0000	040200	GOTO 0x200
0000	000000	NOP

3.11 Reading Configuration Memory

The procedure for reading a Configuration register is the same as reading the code memory. The only difference is that the 16-bit data words are read (with the upper byte read being all '0's) instead of the 24-bit words. There are eight Configuration registers and they are read one register at a time.

Table 3-11 provides the ICSP programming details for reading all of the Configuration registers.

Note:	When reading from the Configuration registers, always set the TBLPAG register to F8h. This is the upper byte address of all locations of the Configuration registers. The Read Pointer, W6, should be initialized to 00h.

TABLE 3-11: SERIAL INSTRUCTION EXECUTION FOR READING ALL THE CONFIGURATION REGISTERS

Command (Binary)	Data (Hex)	Description
Step 1: Exit Reset vector.		
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
Step 2: Initialize TBLPAG, the Read Pointer (W6) and the Write Pointer (W7) for TBLRD instruction.		
0000	200F80	MOV #0xF8, W0
0000	880190	MOV W0, TBLPAG
0000	200007	MOV #0x0000,W6
0000	207847	MOV #VISI, W7
0000	000000	NOP
Step 3: Read the Configuration register and write it to the VISI register (located at 784h), and clock out the VISI register using the REGOUT command.		
0000	BA0BB6	TBLRDL [W6++], [W7]
0000	000000	NOP
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
Step 4: Repeat Step 3 to read other fuses. Load W6 with their respective address.		
Step 5: Reset device internal PC.		
0000	040200	GOTO 0x200
0000	000000	NOP

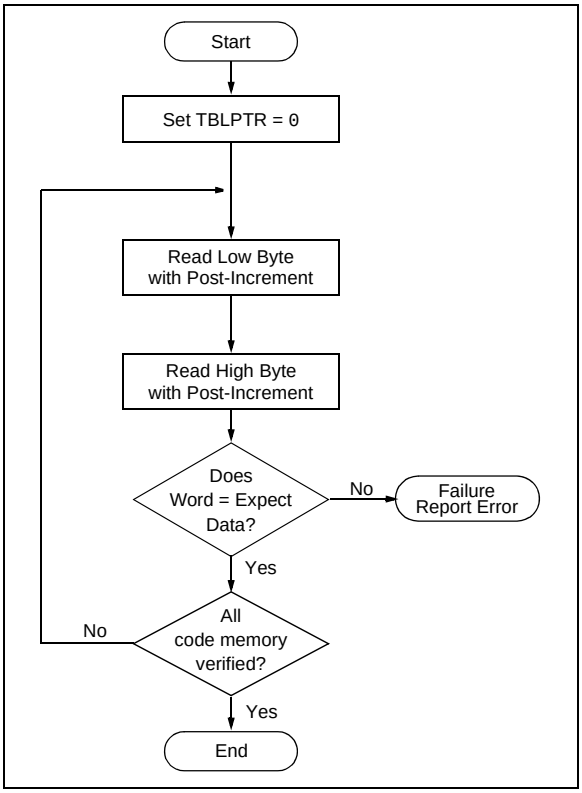
3.12 Verifying Code Memory, Data EEPROM Memory and Configuration Registers

To verify the code memory, read the code memory space and compare it with the copy held in the programmer's buffer. To verify the data EEPROM and Configuration registers, follow the similar procedure.

Figure 3-10 illustrates the verify process flowchart. Memory reads occur 1 byte at a time, hence 2 bytes must be read to compare with the word in the programmer's buffer. Refer to **Section 3.9 "Reading Code Memory"** for implementation details of reading code memory. On the same lines, the data EEPROM and Configuration registers can be verified.

Note:	Code memory should be verified immediately after writing if code protection is enabled. Since Configuration registers include the device code protection bit, the device will not be readable or verifiable if a device Reset occurs after the code-protect bits are set (value = 0).

FIGURE 3-10: VERIFY CODE MEMORY FLOW



3.13 Reading the Application ID Word

The Application ID Word is stored in address 8007F0h in the executive code memory. To read this memory location, use the SIX control code to move this program memory location to the VISI register. Then, the REGOUT control code must be used to clock the contents of the VISI register out of the device. Table 3-12 provides the corresponding control and instruction codes that must be serially transmitted to the device to perform this operation.

After the programmer has clocked out the Application ID Word, it must be inspected. If the Application ID has the value, CBh, the programming executive resides in the memory and the device can be programmed using the mechanism described in **Section 4.0 "Device Programming – Enhanced ICSP"**. However, if the Application ID has any other value, the programming executive does not reside in the memory; it must be loaded to memory before the device can be programmed. **Section 5.4 "Programming the Programming Executive to Memory"** describes the procedure for loading the programming executive to memory.

3.14 Exiting ICSP Mode

Exit the Program/Verify mode by removing V_{IH} from $\overline{MCLR}/V_{PP}$ as illustrated in Figure 3-11. The only requirement to exit is that an interval of P16 should elapse between the last clock and program signals on PGCx and PGDx before removing V_{IH}.

FIGURE 3-11: EXITING ICSP™ MODE

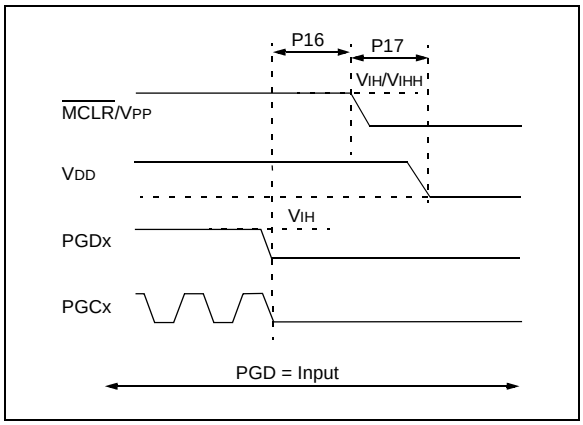


TABLE 3-12: SERIAL INSTRUCTION EXECUTION FOR READING THE APPLICATION ID WORD

Command (Binary)	Data (Hex)	Description
Step 1: Exit Reset vector.		
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
Step 2: Initialize TBLPAG and the Read Pointer (W0) for TBLRD instruction.		
0000	200800	MOV #0x80, W0
0000	880190	MOV W0, TBLPAG
0000	2057F0	MOV #0x7F0, W0
0000	207841	MOV #VISI, W1
0000	000000	NOP
0000	BA0890	TBLRDL [W0], [W1]
0000	000000	NOP
0000	000000	NOP
Step 3: Output the VISI register using the REGOUT command.		
0001	<VISI>	Clock out contents of the VISI register.
0000	000000	NOP

4.0 DEVICE PROGRAMMING – ENHANCED ICSP

Note: The Programming Executive (PE) can be located in the following folder within your installation of MPLAB® IDE: `...\Microchip\MPLAB IDE\REAL ICE` and then selecting the hex PE file, `RIPE_01b_xxxxxx.hex`.

This section describes the programming of the device through Enhanced ICSP and the programming executive. The programming executive resides in the executive memory (separate from user memory space) and is executed when Enhanced ICSP Programming mode is entered. The programming executive provides the mechanism for the programmer (host device) to program and verify the PIC24FXXKA1XX/FVXXKA3XX devices using a simple command set and communication protocol. The basic functions provided by the programming executive are:

- Read Memory
- Program Memory
- Blank Check
- Read Executive Firmware Revision

The programming executive performs the low-level tasks required for erasing, programming and verifying a device. This allows the programmer to program the device by issuing the appropriate commands and data.

Table 4-1 provides these commands. For detailed descriptions of each command, see **Section 5.2 "Programming Executive Commands"**.

TABLE 4-1: COMMAND SET SUMMARY

Command	Description
SCHECK	Sanity check.
READC	Read Device ID registers.
READD	Read data EEPROM memory.
READP	Read Code register.
PROGC	Write User ID.
PROGD	Program and verify one word of data EEPROM memory.
PROGP	Program and verify one row of code memory or one Configuration register.
QBLANK	Query if the code memory is blank.
QVER	Query the software version.

The programming executive uses the device's data RAM for variable storage and program execution. After the programming executive is run, no assumptions should be made about the contents of the data RAM.

4.1 Overview of the Programming Process

Figure 4-1 illustrates the high-level overview of the programming process.

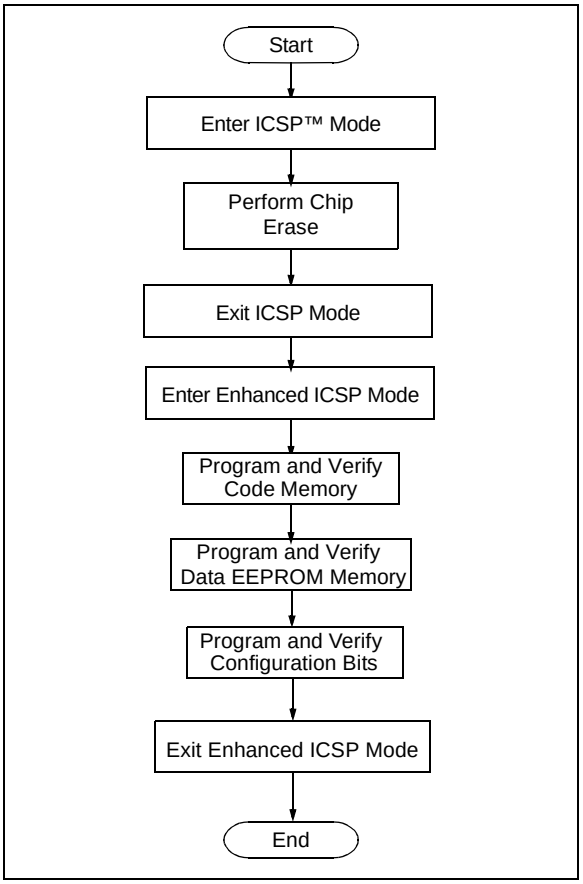
Perform the following steps:

1. Enter ICSP mode.
2. Erase the device.
3. Verify the programming executive.
4. Exit ICSP mode.
5. Enter Enhanced ICSP mode.
6. Program the code memory.
7. Verify the code memory.
8. Program the Configuration registers.
9. Verify the Configuration registers.

Steps 7 and 9 ensure that the programming was successful.

After the programming executive is verified in memory (or loaded if not present), the PIC24FXXKA1XX/FVXXKA3XX devices can be programmed using the command set provided in Table 4-1.

FIGURE 4-1: HIGH-LEVEL ENHANCED ICSP™ PROGRAMMING FLOW



4.2 Confirming the Presence of the Programming Executive

Before beginning programming, confirm if the programming executive is stored in the executive memory and perform the following:

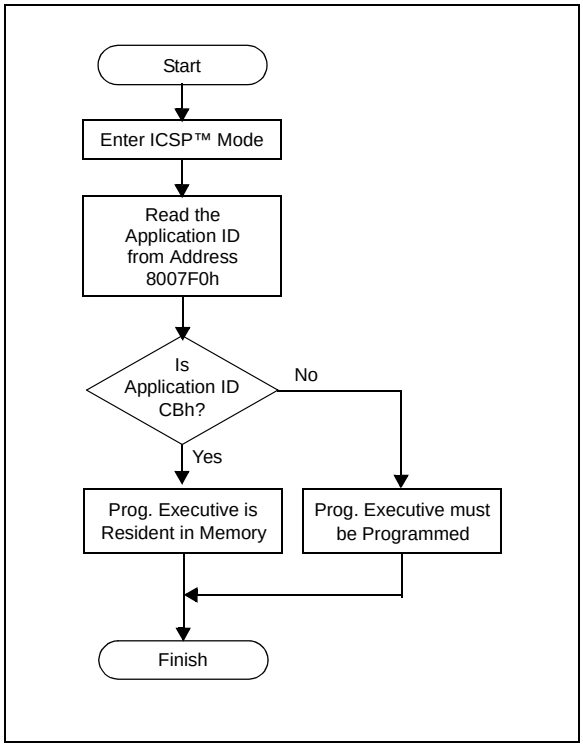
1. Enter In-Circuit Serial Programming mode (ICSP).
2. Read the unique Application ID Word stored in the executive memory.

Figure 4-2 illustrates this procedure.

If the programming executive is resident, the Application ID Word is CBh, which means programming can resume as normal. However, if the Application ID Word is not CBh, the programming executive must be programmed to executive code memory using the method described in **Section 5.4 "Programming the Programming Executive to Memory"**.

Section 3.0 "Device Programming – ICSP" describes the ICSP programming method. **Section 3.13 "Reading the Application ID Word"** describes the procedure for reading the Application ID Word in ICSP mode.

FIGURE 4-2: CONFIRMING PRESENCE OF PROGRAMMING EXECUTIVE



4.3 Entering Enhanced ICSP Mode

4.3.1 LOW-VOLTAGE ENTRY

Perform the following steps to enter Enhanced ICSP Program/Verify mode using MCLR:

1. Briefly drive the MCLR pin high and then low.
2. Clock a 32-bit key sequence into PGDx.
3. Drive the MCLR high within a specified period and continue to hold high.

Figure 4-3 illustrates this procedure.

The programming voltage applied to MCLR is VIH, which is essentially VDD in the case of PIC24FXXKA1XX/FVXXKA3XX devices. There is no minimum time requirement for holding at VIH. After VIH is removed, an interval of at least P18 must elapse before presenting the key sequence on PGDx.

The key sequence is a specific 32-bit pattern: '0100 1101 0100 0011 0100 1000 0101 0000'(more easily remembered as 4D434850h in hexadecimal format). The device will enter Program/Verify mode only if the key sequence is valid. The MSB of the most significant nibble must be shifted in first.

Once the key sequence is complete, VIH must be applied to MCLR and held at that level for as long as the Program/Verify mode is to be maintained. An interval of at least P19 and P7 must elapse before presenting data on PGDx. Signals appearing on PGDx before P7 has elapsed will not be interpreted as valid.

4.3.2 HIGH-VOLTAGE ENTRY

The procedure for entering Enhanced ICSP Program/Verify mode using the VPP pin is the same as entering the mode using MCLR. The only differences are that the programming voltage applied to VPP is VIHH, and before presenting the key sequence on PGDx, an interval of at least P18 should elapse. Figure 4-4 illustrates this procedure.

Once the key sequence is complete, VIHH must be applied to VPP and held at that level for as long as the Program/Verify mode is to be maintained. An interval of at least P19 and P7 should elapse before presenting data on PGDx.

Signals appearing on PGDx before P7 has elapsed will not be interpreted as valid.

On successful entry, the program memory can be accessed and programmed in serial fashion. While in the Program/Verify mode, all unused I/Os are placed in the high-impedance state.

4.3.3 CODE-PROTECT ICSP ENTRY

When code protection is employed on the PIC24FXXKA3XX devices (BWRP, GSS0 or GWRP = 0), then the voltage on VDD must be above VBULK in order to erase, and then program the device. Care must be taken in the design and layout of a board so that any parts connected to VDD can withstand what may be an increase in voltage if the device is running below VBULK.

FIGURE 4-3: ENTERING ENHANCED ICSP™ MODE USING LOW-VOLTAGE ENTRY

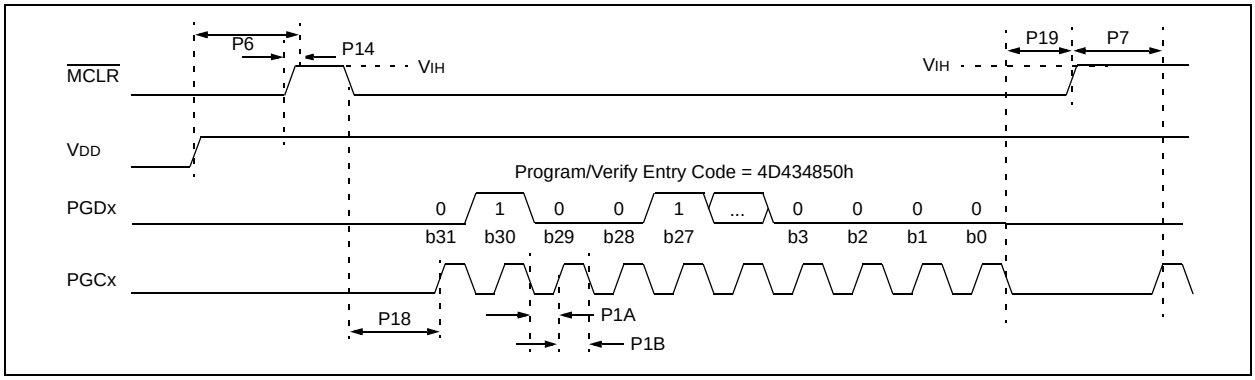
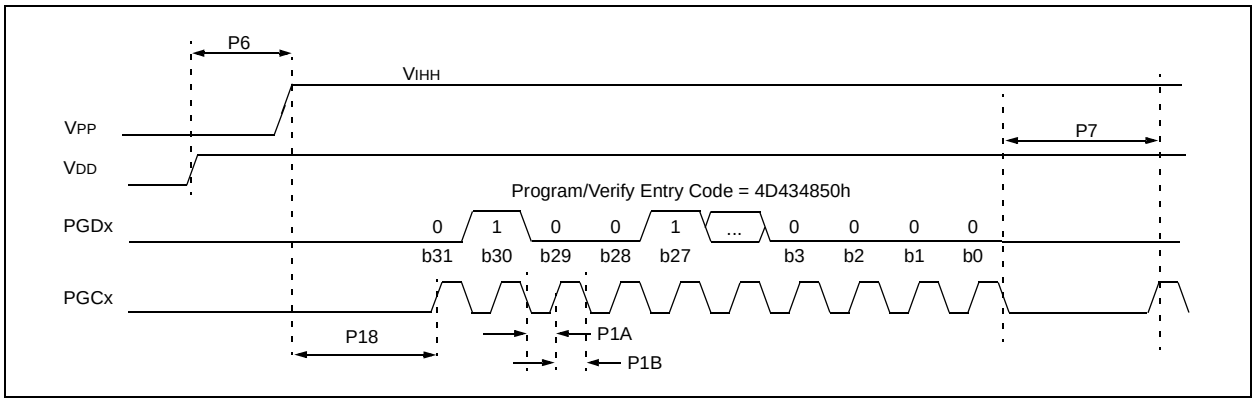


FIGURE 4-4: ENTERING ENHANCED ICSP™ MODE USING HIGH-VOLTAGE ENTRY



4.4 Blank Check

The term “Blank Check” implies verifying if the device has been successfully erased and has no programmed memory locations. A blank or erased memory location is always read as ‘1’.

The Device ID registers (FF0002h:FF0000h) can be ignored by the Blank Check as this region stores device information that cannot be erased. The device Configuration registers are also ignored by the Blank Check. Additionally, all unimplemented memory space should be ignored by the Blank Check.

The QBLANK command is used for the Blank Check. It determines if the code memory is erased by testing these memory regions. A ‘BLANK’ or ‘NOT BLANK’ response is returned. If it is determined that the device is not blank, it must be erased before attempting to program the device.

4.5 Code Memory Programming

4.5.1 PROGRAMMING METHODOLOGY

Code memory is programmed with the PROGP command. PROGP programs one row of code memory, starting from the memory address specified in the command. The number of PROGP commands required to program a device depends on the number of write blocks that must be programmed in the device.

Figure 4-5 illustrates an example flowchart for programming code memory. In this example, all 5.5K instruction words of a PIC24FXXKA1XX/FVXXKA3XX device are programmed.

- First, the number of commands to send (titled ‘RemainingCmds’ in the flowchart) is set to 176, and the destination address (called ‘BaseAddress’) is set to ‘0’.
- Next, one write block in the device is programmed with a PROGP command. Each PROGP command contains data for one row of code memory of the PIC24FXXKA1XX/FVXXKA3XX device.
- After the first command is processed successfully, ‘RemainingCmds’ is decremented by 1 and compared with 0.

Since there are more PROGP commands to be sent, ‘BaseAddress’ is incremented by 40h to point to the next row of memory. On the second PROGP command, the second row is programmed. This process is repeated until the entire device is programmed. Special handling should not be performed when a panel boundary is crossed.

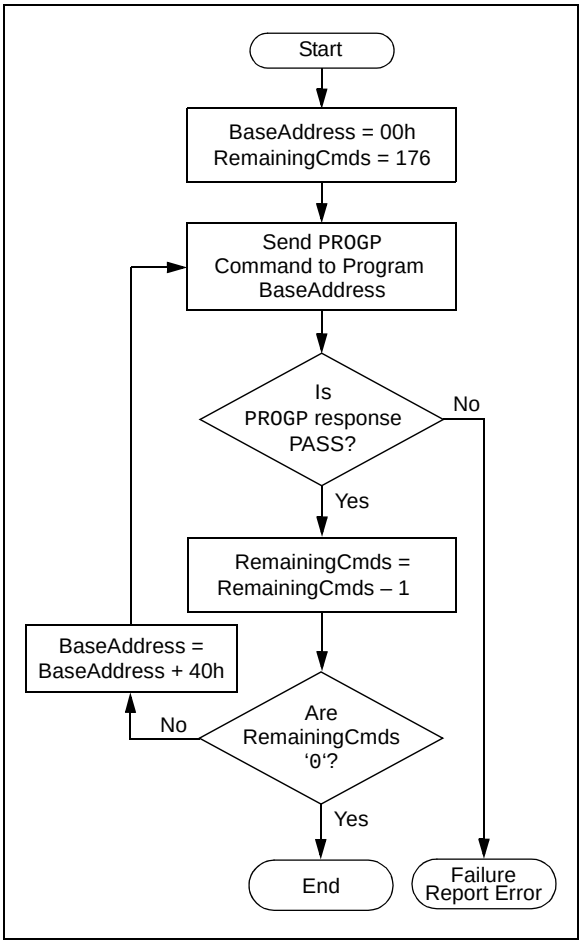
4.5.2 PROGRAMMING VERIFICATION

After the code memory is programmed, the contents of the memory can be verified to ensure that the programming is successful. Verification requires the code memory to be read back and compared with the copy held in the programmer‘s buffer.

The READP command can be used to read back all of the programmed code memory.

Alternatively, the programmer can perform the verification after the entire device is programmed using a checksum computation.

FIGURE 4-5: FLOWCHART FOR PROGRAMMING CODE MEMORY



4.6 Data EEPROM Programming

4.6.1 PROGRAMMING METHODOLOGY

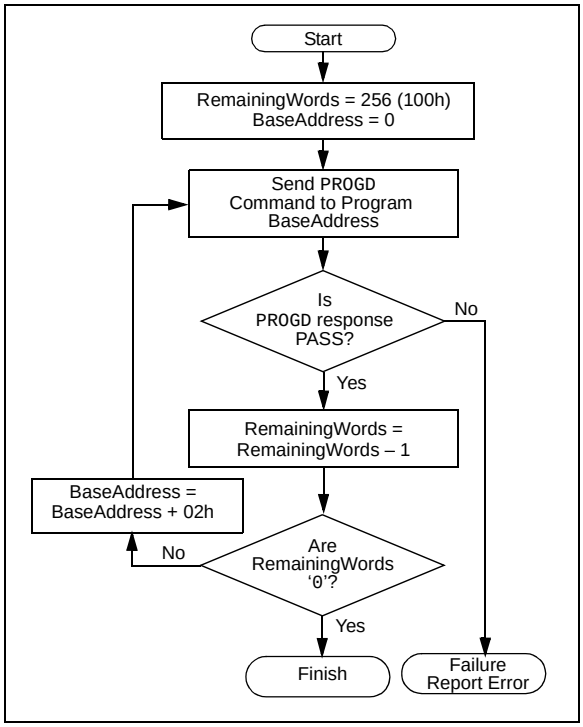
The programming executive uses the PROGD command to program the data EEPROM. Figure 4.7 illustrates this process.

- First, the number of words to program (RemainingWords) is based on the device size and the destination address (DestAddress) is set to 0. In this example, 256 words of data EEPROM will be programmed.
- The first PROGD command programs the first word of data EEPROM.
- Once the command completes successfully, 'RemainingWords' is decremented by 1 and compared with 0.
- Since there are 255 more words to program, 'BaseAddress' is incremented by 02h to point to the next word of data EEPROM.
- This process is then repeated until all 256 words of data EEPROM are programmed.

4.6.2 PROGRAMMING VERIFICATION

Once the data EEPROM is programmed, the contents of the memory can be verified to ensure the programming was successful. Verification requires the data EEPROM to be read back and compared with the copy held in the programmer's buffer. The READD command reads back the programmed data EEPROM. Alternatively, the programmer can perform the verification once the entire device is programmed using a checksum computation, as described in **Section 6.1.1 "Checksum Computation"**.

FIGURE 4-6: FLOWCHART FOR PROGRAMMING DATA EEPROM



4.7 Configuration Bits Programming

The PIC24FXXKA1XX/FVXXKA3XX family has eight Configuration registers. The bits of these registers can be set or cleared to select various device configurations. There are three types of Configuration bits:

- System Operation Bits

  These bits determine the power-on settings for system-level components, such as the oscillator and Watchdog Timer.

- Code-Protect Bits

  These bits prevent program memory from being read and written.

- Device ID Bits

  These are read-only bits, which are located from FF0000 to FF0003, and are unique to every device.

Table 4-2 provides the Configuration registers.

4.7.1 PROGRAMMING METHODOLOGY

Configuration bits may be programmed, a single byte at a time, using the PROGP command. This command specifies the configuration data and Configuration register address. When Configuration bits are programmed, any unimplemented or reserved bits must be programmed with a '1'.

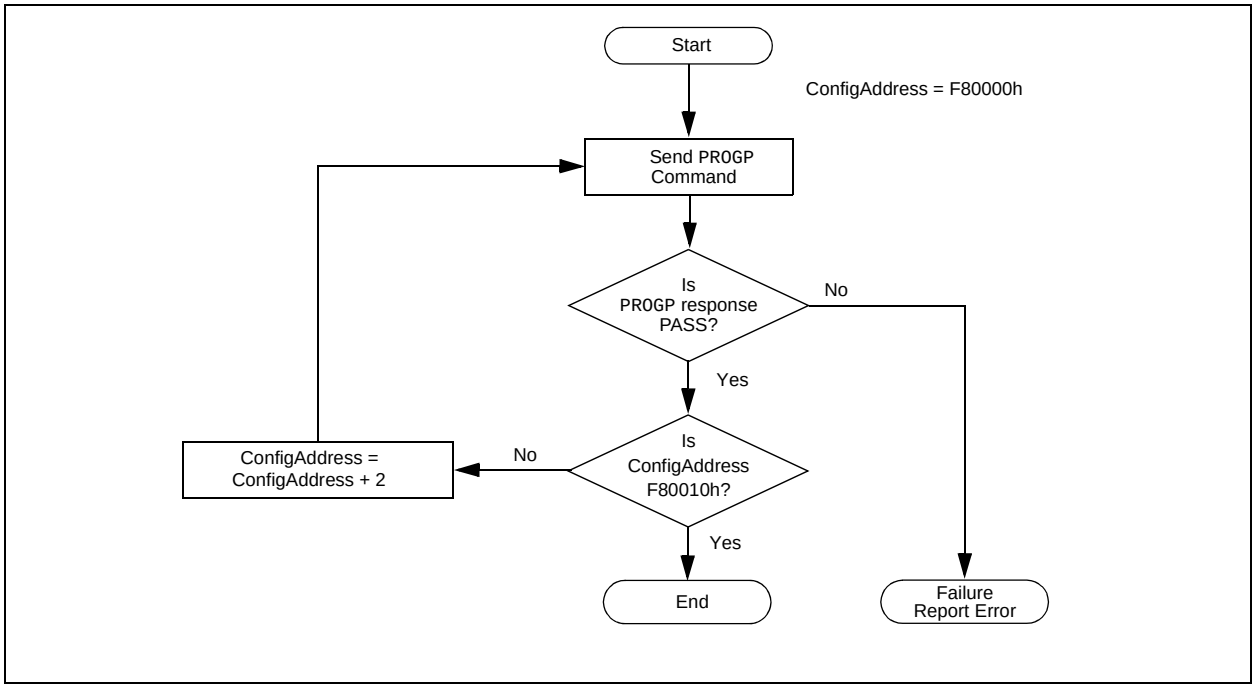
Eight PROGP commands are required to program the Configuration bits. Figure 4-7 illustrates the flowchart for Configuration bit programming.

Note:	If the General Segment Code-Protect bit (GCP) is programmed to '0', code memory is code-protected and cannot be read. Code memory must be verified before enabling code protection. See **Section 4.7.3 "Code-Protect Configuration Bits"** for more information on code-protect Configuration bits.

4.7.2 PROGRAMMING VERIFICATION

After the Configuration bits are programmed, the contents of memory should be verified to ensure that the programming was successful. Verification requires the Configuration bits to be read back and compared against the copy held in the programmer's buffer. The READP command reads back the programmed Configuration bits and verifies that the programming was successful.

FIGURE 4-7: CONFIGURATION BIT PROGRAMMING FLOW



4.7.3 CODE-PROTECT CONFIGURATION BITS

The FBS and FGS Configuration registers are special Configuration registers which control the code protection for the boot segment and general segment, respectively. For each segment, two forms of code protection are provided. One form prevents code memory from being written (write protection), while the other prevents code memory from being read (read protection).

The BWRP and GWRP bits control write protection and the BSS0 and GSS0 bits control read protection.

When write protection is enabled, any programming operation to code memory will fail. When read protection is enabled, any read from code memory will cause a 00h to be read, regardless of the actual contents of code memory. Since the programming executive always verifies what it programs, attempting to program code memory with read protection enabled also results in failure.

It is imperative that all code protection bits should be '1' while the device is being programmed and verified. Only after the device is programmed and verified should any of the above bits be programmed to '0' (see **Section 4.7 "Configuration Bits Programming"**).

Note:	All bits in the FBS and FGS Configuration registers can only be programmed to a value of '0'. Bulk Erasing the chip is the only way to reprogram code-protect bits from on ('0') to off ('1').

TABLE 4-2: PIC24F(V)XXKAXXX FAMILY CONFIGURATION BITS DESCRIPTION

Bit Field	Register	Description
BOREN<1:0>	FPOR<1:0>	Brown-out Reset Enable bits 11 = Brown-out Reset enabled in hardware; SBOREN bit disabled 10 = Brown-out Reset enabled only while device is active and disabled in Sleep; SBOREN bit disabled 01 = Brown-out Reset controlled with the SBOREN bit setting 00 = Brown-out Reset disabled in hardware; SBOREN bit disabled
BORV<1:0>	FPOR<6:5>	Brown-out Reset Voltage bits For PIC24FXXKA1XX and PIC24FXXKA3XX devices: 11 = VBOR set to 1.8V min 10 = VBOR set to 2.7V min 01 = VBOR set to 3.0V min For PIC24FVXXKA3XX devices: 11 = VBOR set to 2.0V min 10 = VBOR set to 2.7V min 01 = VBOR set to 3.0V min For all devices: 00 = Downside protection on POR enabled – "Zero-power" selected
BSS0	FBS<3>	Boot Segment Program Flash Code Protection bit 1 = Standard security enabled 0 = High security enabled
BSZ<1:0>	FBS<2:1>	Boot Segment Program Flash Size Selection bits(1) 11 = No boot program Flash segment 10 = Boot program Flash segment starts at 200h, ends at 000AFEh 01 = Boot program Flash segment starts at 200h, ends at 0015FEh For PIC24FV32KA3XX devices: 00 = Reserved For all other devices: 00 = Reserved

Note 1: Applies only to 28-pin devices.

2: The MCLRE fuse can only be changed when using the VPP-Based Test mode entry. This prevents a user from accidentally locking out the device from low-voltage test entry.

TABLE 4-2: PIC24F(V)XXKAXXX FAMILY CONFIGURATION BITS DESCRIPTION (CONTINUED)

Bit Field	Register	Description
BWRP	FBS<0>	Boot Segment Program Flash Write Protection bit 1 = Boot segment may be written 0 = Boot segment is write-protected
$\overline{\text{DEBUG}}$	FICD<7>	Background Debugger Enable bit 1 = Background debugger disabled 0 = Background debugger functions enabled
DSWDTEN	FDS<7>	Deep Sleep Watchdog Timer Enable bit 1 = DSWDT enabled 0 = DSWDT disabled
DSWCKSEL	FDS<4>	DSWDT Reference Clock Select bit 1 = DSWDT uses LPRC as reference clock 0 = DSWDT uses SOSC as reference clock
DSWDTPS<3:0>	FDS<3:0>	Deep Sleep Watchdog Timer Postscale Select bits The DSWDT prescaler is 32; this creates an approximate base time unit of 1 ms. 1111 = 1:2,147,483,648 (25.7 days) 1110 = 1:536,870,912 (6.4 days) 1101 = 1:134,217,728 (38.5 hours) 1100 = 1:33,554,432 (9.6 hours) 1011 = 1:8,388,608 (2.4 hours) 1010 = 1:2,097,152 (36 minutes) 1001 = 1:524,288 (9 minutes) 1000 = 1:131,072 (135 seconds) 0111 = 1:32,768 (34 seconds) 0110 = 1:8,192 (8.5 seconds) 0101 = 1:2,048 (2.1 seconds) 0100 = 1:512 (528 ms) 0011 = 1:128 (132 ms) 0010 = 1:32 (33 ms) 0001 = 1:8 (8.3 ms) 0000 = 1:2 (2.1 ms)
DSBOREN	FDS<6>	Deep Sleep Zero-Power BOR Enable bit 1 = Zero-Power BOR enabled in Deep Sleep 0 = Zero-Power BOR disabled in Deep Sleep (does not affect operation in non Deep Sleep modes)
FCKSM<1:0>	FOSC<7:6>	Clock Switching and Monitor Selection Configuration bits 1x = Clock switching is disabled, Fail-Safe Clock Monitor is disabled 01 = Clock switching is enabled, Fail-Safe Clock Monitor is disabled 00 = Clock switching is enabled, Fail-Safe Clock Monitor is enabled
FNOSC<2:0>	FOSCSEL<2:0>	Oscillator Selection bits 000 = Fast RC Oscillator (FRC) 001 = Fast RC Oscillator with divide-by-N with PLL module (FRCDIV+PLL) 010 = Primary Oscillator (XT, HS, EC) 011 = Primary Oscillator with PLL module (HS+PLL, EC+PLL) 100 = Secondary Oscillator (SOSC) 101 = Low-Power RC Oscillator (LPRC) 110 = Reserved; do not use 111 = Fast RC Oscillator with divide-by-N (FRCDIV)

Note 1: Applies only to 28-pin devices.

2: The MCLRE fuse can only be changed when using the VPP-Based Test mode entry. This prevents a user from accidentally locking out the device from low-voltage test entry.

TABLE 4-2: PIC24F(V)XXKAXXX FAMILY CONFIGURATION BITS DESCRIPTION (CONTINUED)

Bit Field	Register	Description
FWDTEN	FWDT<7>	Watchdog Timer Enable bit (PIC24FXXKA1XX devices only) 1 = WDT enabled 0 = WDT disabled (control is placed on the SWDTEN bit)
FWDTEN<1:0>	FWDT<7,5>	Watchdog Timer Enable bit (all PIC24FVXXKA3XX devices) 11 = WDT enabled in hardware 10 = WDT controlled with the SWDTEN bit 01 = WDT enabled only while device active and disabled in Sleep; SWDTEN bit disabled 00 = WDT disabled in hardware; SWDTEN bit disabled
GSS0	FGS<1>	General Segment Code Flash Code Protection bit 1 = No protection 0 = Standard security enabled
GWRP	FGS<0>	General Segment Code Flash Write Protection bit 1 = General segment may be written 0 = General segment is write-protected
ICS<1:0>	FICD<1:0>	ICD Pin Placement Select bit 11 = ICD EMUC/EMUD pins are shared with PGEC1/PGED1 10 = ICD EMUC/EMUD pins are shared with PGEC2/PGED2 01 = ICD EMUC/EMUD pins are shared with PGEC3/PGED3 00 = Reserved; do not use
IESO	FOSCSEL<7>	Internal External Switchover bit 1 = Internal External Switchover mode enabled (Two-Speed Start-up enabled) 0 = Internal External Switchover mode disabled (Two-Speed Start-up disabled)
I2C1SEL(1)	FPOR<4>	Alternate I2C1 Pin Mapping bit(1) 0 = Alternate location for SCL1/SDA1 pins 1 = Default location for SCL1/SDA1 pins
MCLRE(2)	FPOR<7>	$\overline{\text{MCLR}}$ Pin Enable bit(2) 1 = $\overline{\text{MCLR}}$ pin enabled; RA5 input pin disabled 0 = RA5 input pin enabled; $\overline{\text{MCLR}}$ disabled
OSCIOFNC	FOSC<2>	CLKO Enable Configuration bit 1 = CLKO output signal active on the OSCO pin; primary oscillator must be disabled or configured for the External Clock mode (EC) for the CLKO to be active (POSCMD<1:0> = 11 or 00) 0 = CLKO output disabled
POSCMD<1:0>	FOSC<1:0>	Primary Oscillator Configuration bits 11 = Primary oscillator disabled 10 = HS Oscillator mode selected (4 MHz-25 MHz) 01 = XT Oscillator mode selected (100 kHz-4 MHz) 00 = External Clock mode selected
POSCFREQ<1:0>	FOSC<4:3>	Primary Oscillator Frequency Range Configuration bits 11 = Primary oscillator/external clock input frequency greater than 8 MHz 10 = Primary oscillator/external clock input frequency between 100 kHz and 8 MHz 01 = Primary oscillator/external clock input frequency less than 100 kHz 00 = Reserved, do not use
PWRTEN	FPOR<3>	Power-up Timer Enable bit 0 = PWRT disabled 1 = PWRT enabled

Note 1: Applies only to 28-pin devices.

2: The MCLRE fuse can only be changed when using the VPP-Based Test mode entry. This prevents a user from accidentally locking out the device from low-voltage test entry.

TABLE 4-2: PIC24F(V)XXKAXXX FAMILY CONFIGURATION BITS DESCRIPTION (CONTINUED)

Bit Field	Register	Description
RTCCKSEL	FDS<5>	RTCC Reference Clock Select bit (PIC24FXXKA1XX devices only) 1 = RTCC uses SOSC as reference clock 0 = RTCC uses LPRC as reference clock
SOSCSEL	FOSC<5>	Secondary Oscillator Select bit 1 = Secondary oscillator configured for high-power operation 0 = Secondary oscillator configured for low-power operation
FWPSA	FWDT<4>	WDT Prescaler 1 = WDT prescaler ratio of 1:128 0 = WDT prescaler ratio of 1:32
WDTPS<3:0>	FWDT<3:0>	Watchdog Timer Postscale Select bits 1111 = 1:32,768 1110 = 1:16,384 • • • 0001 = 1:2 0000 = 1:1
WINDIS	FWDT<6>	Windowed Watchdog Timer Disable bit 1 = Standard WDT selected; windowed WDT disabled 0 = Windowed WDT enabled

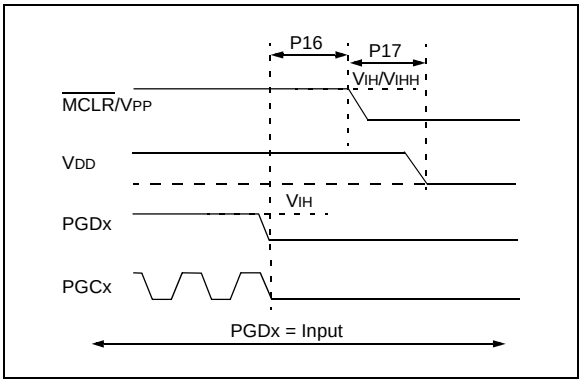
Note 1: Applies only to 28-pin devices.

2: The MCLRE fuse can only be changed when using the VPP-Based Test mode entry. This prevents a user from accidentally locking out the device from low-voltage test entry.

4.8 Exiting the Enhanced ICSP Mode

To exit the Program/Verify mode, remove VIH from MCLR/VPP, as illustrated in Figure 4-8. For exiting, an interval, P16, should elapse between the last clock and program signals on PGCx and PGDx before removing VIH.

FIGURE 4-8: EXITING ENHANCED ICSP™ MODE



5.0 THE PROGRAMMING EXECUTIVE

This section describes the programming executive communication, programming executive commands, programming responses, programming the programming executive to memory and programming verification.

5.1 Programming Executive Communication

The programmer and the programming executive have a master-slave relationship, where the programmer is the master programming device and the programming executive is the slave.

Communication is initiated by the programmer in the form of a command. Only one command at a time can be sent to the programming executive. The programming executive, in turn, only sends one response to the programmer after receiving and processing a command. The programming executive command set is described in **Section 5.2 "Programming Executive Commands"**. The response set is described in **Section 5.3 "Programming Executive Responses"**.

5.1.1 COMMUNICATION INTERFACE AND PROTOCOL

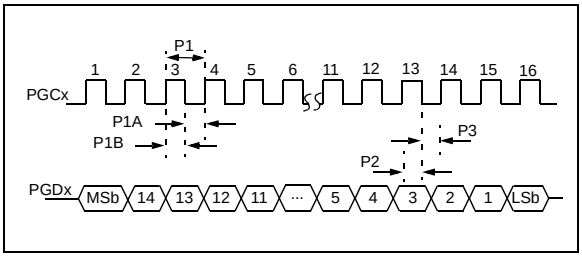
The Enhanced ICSP interface is a two-wire SPI, implemented using the PGCx and PGDx pins. The PGCx pin is used as a clock input pin; the programmer should provide the clock source. The PGDx pin is used to send the command data to, and receive response data from, the programming executive.

Data transmits to the device should change on the rising edge and hold on the falling edge of PGCx.

Data receives from the device change on the falling edge and holds on the rising edge of PGCx.

The data transmissions are sent to the MSB first using 16-bit mode (see Figure 5-1 and Figure 5-2).

FIGURE 5-1: PROGRAMMING EXECUTIVE SERIAL TIMING FOR DATA RECEIVED FROM DEVICE



As a 2-wire SPI is used and data transmissions are half-duplex, a simple protocol is used to control the direction of PGDx. When the programmer completes a command transmission, it releases the PGDx line and allows the programming executive to drive this line high. The programming executive keeps the PGDx line high to indicate that it is processing the command.

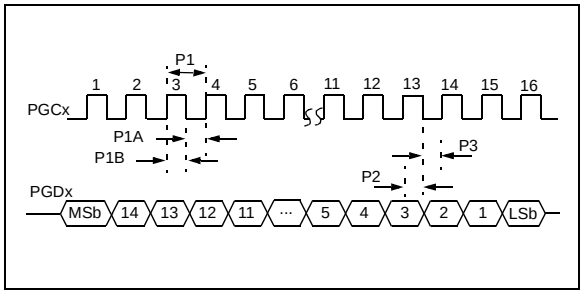
After the programming executive has processed the command, it brings PGDx low for 15 μsec to indicate to the programmer that the response is available to be clocked out. The programmer can begin to clock out the response, 23 μsec after PGDx is brought low, and it must provide the necessary amount of clock pulses to receive the entire response from the programming executive.

After the entire response is clocked out, the programmer should terminate the clock on PGCx until it is time to send another command to the programming executive; Figure 5.2 displays this protocol.

5.1.2 SPI RATE

In Enhanced ICSP mode, the PIC24FXXKA1XX/FVXXKA3XX family devices operate from the internal Fast RC Oscillator, which has a nominal frequency of 8 MHz. This oscillator frequency yields an effective system clock frequency of 4 MHz. To ensure that the programmer does not clock too fast, it is recommended that a 4 MHz clock be provided by the programmer.

FIGURE 5-2: PROGRAMMING EXECUTIVE SERIAL TIMING FOR DATA TRANSMITTED TO DEVICE

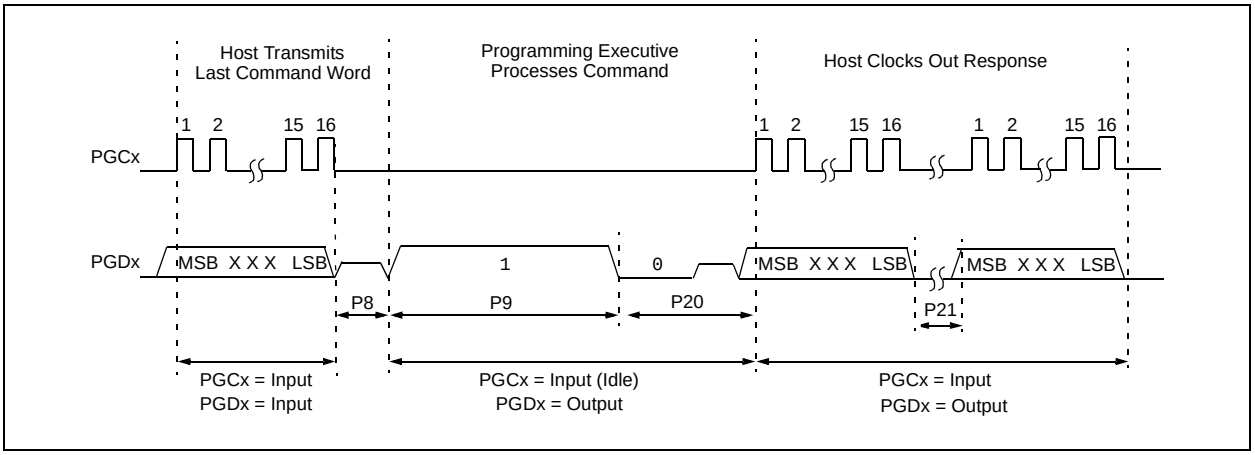


5.1.3 TIME-OUTS

The programming executive does not use the Watchdog Timer or time-out for transmitting responses to the programmer. If the programmer does not follow the flow control mechanism using PGCx, as described in **Section 5.1.1 "Communication Interface and Protocol"**, it is possible that the programming executive will behave unexpectedly while trying to send a response to the programmer. Since the programming executive does not have a time-out, it is imperative that the programmer correctly follow the described communication protocol.

As a safety measure, the programmer should use the command time-outs identified and listed in Table 5-1. If the command time-out expires, the programmer should reset the programming executive and start programming the device again.

FIGURE 5-3: PROGRAMMING EXECUTIVE – PROGRAMMER COMMUNICATION PROTOCOL



5.2 Programming Executive Commands

The programming executive command set is listed in Table 5-1. This table contains the opcode, mnemonic, length, time-out and description for each command. **Section 5.2.4 "Command Descriptions"** provides functional details on each command.

5.2.1 COMMAND FORMAT

The programming executive commands have a general format consisting of a 16-bit header and any required data for the command (see Figure 5-4). The 16-bit header consists of a 4-bit opcode field, which is used to identify the command, followed by a 12-bit command length field.

FIGURE 5-4: COMMAND FORMAT

15 12	11 0
Opcode	Length
Command Data First Word (if required)	
•	
•	
Command Data Last Word (if required)	

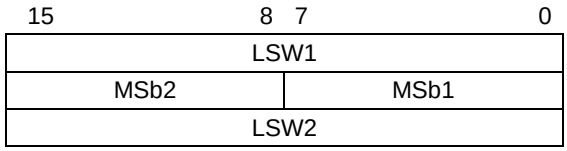
The command opcode must match one of those in the command set. Any command that is received that does not match the list in Table 5-1 returns a "NACK" response (see **Section 5.3.1.1 "Opcode Field"**).

The command length is represented in 16-bit words as the SPI operates in 16-bit mode. The programming executive uses the command length field to determine the number of words to read from the SPI port. If the value of this field is incorrect, the command will not be properly received by the programming executive.

5.2.2 PACKED DATA FORMAT

When 24-bit instruction words are transferred across the 16-bit SPI interface, they are packed to conserve space using the format illustrated in Figure 5-5. This format minimizes the traffic over the SPI and provides the programming executive the data that is properly aligned for performing table write operations.

FIGURE 5-5: PACKED INSTRUCTION WORD FORMAT



LSWx: Least Significant 16 bits of instruction word
MSbx: Most Significant bits of instruction word

Note: When the number of instruction words transferred is odd, MSb2 is zero and LSW2 cannot be transmitted.

5.2.3 PROGRAMMING EXECUTIVE ERROR HANDLING

The programming executive will return a "NACK" response for all unsupported commands. Additionally, due to the memory constraints of the programming executive, no checking is performed on the data contained in the programmer command. It is the responsibility of the programmer to command the programming executive with valid command arguments; otherwise, the programming operation might fail. **Section 5.3.1.3 "QE_Code Field"** provides additional information on error handling.

5.2.4 COMMAND DESCRIPTIONS

The commands supported by the programming executive are described in **Section 5.2.5 "SCHECK Command"** through **Section 5.2.13 "QVER Command"**.

TABLE 5-1: PROGRAMMING EXECUTIVE COMMAND SET

Opcode	Mnemonic	Length (16-bit words)	Time-out	Description
0h	SCHECK	1	1 ms	Sanity check.
1h	READC	3	1 ms	Read up to (256) 8-bit words starting from the specified Device ID register.
2h	READP	4	1 ms/row	Read up to 32K instruction words of code memory starting from the specified address.(1)
3h	RESERVED	N/A	N/A	This command is reserved; it returns a NACK.
4h	PROGC	4	5 ms	Write an 8-bit word to the specified Device ID registers.
5h	PROGP	99	5 ms	Program up to 32 instructions (one row) of code memory at the specified address and then verify.(1)
6h	RESERVED	N/A	N/A	This command is reserved; it returns a NACK.
7h	RESERVED	N/A	N/A	This command is reserved; it returns a NACK.
8h	RESERVED	N/A	N/A	This command is reserved; it returns a NACK.
9h	RESERVED	N/A	N/A	This command is reserved; it returns a NACK.
Ah	QBLANK	3	TBD	Query if the code memory is blank.(1)
Bh	QVER	1	1 ms	Query the programming executive software version.
Ch	RESERVED	N/A	N/A	This command is reserved; it returns a NACK.
Dh	RESERVED	N/A	N/A	This command is reserved. it returns a NACK.
Eh	READD	4	1 ms/word	Read up to (256) 16-bit words starting from the specified address.
Fh	PROGD	19	5 ms	Program one word of data EEPROM memory at the specified address and then verify.

Note 1: One row of code memory consists of (32) 24-bit words. Refer to Table 2-2 for device-specific information.

5.2.5 SCHECK COMMAND

15	12	11	0
Opcode		Length	

Field	Description
Opcode	0h
Length	1h

The SCHECK command instructs the programming executive to merely generate a response. This command is used as a “Sanity Check” to verify if the programming executive is operational.

Expected Response (2 words):

1000h

0002h

Note:	This instruction is provided for development purposes only; this is not required for programming.

5.2.6 READC COMMAND

15	12	11	8	7	0
Opcode		Length			
N				Addr_MSB	
Addr_LS					

Field	Description
Opcode	1h
Length	3h
N	Number of 8-bit Device ID registers to read (max. of 256).
Addr_MSB	MSb of 24-bit source address.
Addr_LS	Least Significant 16 bits of 24-bit source address.

The READC command instructs the programming executive to read N or Device ID registers, starting from the 24-bit address specified by Addr_MSB and Addr_LS.

Note:	The READC command can only be used to read 8-bit or 16-bit data.

When this command is used to read the Device ID registers, the upper byte in every data word returned by the programming executive is 00h and the lower byte contains the Device ID register value.

Expected Response (4 + 3 * (N – 1)/2 words for N odd):

1100h

2 + N

Device ID Register 1

...

Device Register N

Note:	Reading unimplemented memory will cause the programming executive to reset. Ensure that only memory locations present on a particular device are accessed.

5.2.7 READD COMMAND

15 12	11 8	7 0
Opcode	Length	
Reserved0		N
Reserved1		Addr_MSB
Addr_LS		

Field	Description
Opcode	Eh
Length	4h
Reserved0	0h
N	Number of 16-bit words to read (max of 256).
Reserved1	0h
Addr_MSB	MSb of 24-bit source address.
Addr_LS	Least Significant 16 bits of 24-bit source address.

The READD command instructs the programming executive to read N 16-bit words from data EEPROM memory, starting from the 24-bit address specified by Addr_MSB and Addr_LS.

Note: This command can only be used to read 16-bit data.

Expected Response (N + 2 words):

1E00h

2 + N

Data word 1

...

Data word N

5.2.8 READP COMMAND

15 12	11 8	7 0
Opcode	Length	
N		
Reserved		Addr_MSB
Addr_LS		

Field	Description
Opcode	2h
Length	4h
N	Number of 24-bit instructions to read (max. of 32768).
Reserved	0h
Addr_MSB	MSb of 24-bit source address.
Addr_LS	Least Significant 16 bits of 24-bit source address.

The READP command instructs the programming executive to read N 24-bit words of code memory, including Configuration registers, starting from the 24-bit address specified by Addr_MSB and Addr_LS.

Note: This command can only be used to read 24-bit data.

The entire data returned in response to this command uses the packed data format described in **Section 5.2.2 "Packed Data Format"**.

Expected Response (2 + 3 * N/2 words for N even):

1200h

2 + 3 * N/2

LSB program memory word 1

...

LSB data word N

Expected Response (4 + 3 * (N – 1)/2 words for N odd):

1200h

4 + 3 * (N – 1)/2

LSB program memory word 1

...

MSB of program memory word N (zero-padded)

Note: Reading unimplemented memory will cause the programming executive to reset. Ensure that only memory locations present on a particular device are accessed.

5.2.9 PROGC COMMAND

15 12	11 8	7 0
Opcode	Length	
Reserved	Addr_MSB	
Addr_LS		
Data		

Field	Description
Opcode	4h
Length	4h
Reserved	0h
Addr_MSB	MSb of 24-bit destination address.
Addr_LS	Least Significant 16 bits of 24-bit destination address.
Data	8-bit data word.

The PROGC command instructs the programming executive to program a single User ID register located at the specified memory address.

After the specified data word has been programmed to code memory, the programming executive verifies the programmed data against the data in the command.

Expected Response (2 words):

1400h
0002h

5.2.10 PROGD COMMAND

15 12	11 8	7 0
Opcode	Length	
Reserved	Addr_MSB	
Addr_LS		
D_1		

Field	Description
Opcode	Fh
Length	4h
Reserved	0h
Addr_MSB	MSB of 24-bit source address.
Addr_LS	Least Significant 16 bits of 24-bit source address.
D_1	16-bit data word.

The PROGD command instructs the programming executive to program one word (16-bit) of data EEPROM memory, starting from the 24-bit address specified by Addr_MSB and Addr_LS.

Once one word of data EEPROM has been programmed, the programming executive verifies the programmed data against the data in the command.

Expected Response (2 words):

1F00h
0002h

Note: Refer to Table 2-2 for data EEPROM memory size information.

5.2.11 PROGP COMMAND

15 12	11 8	7 0
Opcode	Length	
Reserved	Addr_MSB	
Addr_LS		
D_1		
D_2		
...		
D_48		

Field	Description
Opcode	5h
Length	33h
Reserved	0h
Addr_MSB	MSb of 24-bit destination address.
Addr_LS	Least Significant 16 bits of 24-bit destination address.
D_1	16-bit data word 1.
D_2	16-bit data word 2.
...	16-bit data word 3 through 47.
D_48	16-bit data word 48.

The PROGP command instructs the programming executive to program one row of code memory to the specified memory address. Programming begins with the row address specified in the command. The destination address should be a multiple of 40h.

The data to program to memory, located in command words, D_1 through D_48, should be arranged using the packed instruction word format depicted in Figure 5-5.

After the entire data is programmed to code memory, the programming executive verifies the programmed data against the data in the command.

The PROGP command is also used to program Configuration Words. While PROGP is used to program Configuration Words, the length in the command should be four. Only one Configuration Word at a time can be programed. The unimplemented bits of the Configuration Word should be stuffed with '1's.

Expected Response (2 words):

1500h

0002h

Note:	Refer to Table 2-2 for program memory size information.

5.2.12 QBLANK COMMAND

15 12	11 0
Opcode	Length
PSize	
Reserved	DSize

Field	Description
Opcode	Ah
Length	3h
PSize	Length of program memory to check in 24-bit words (max. of 49152).
Reserved	0h
DSize	Length of data memory to check in 16-bit words (max. of 2048).

The QBLANK command queries the programming executive to determine if the contents of code memory and code-protect Configuration bits (GCP and GWRP) are blank (contain all '1's). The size of the code memory to check should be specified in the command.

The Blank Check for code memory begins at 0h and advances toward larger addresses for the specified number of instruction words.

QBLANK returns a QE_Code of F0h if the specified code memory and code-protect bits are blank; otherwise, it returns a QE_Code of 0Fh.

Expected Response (2 words for blank device):

1AF0h

0002h

Expected Response (2 words for non-blank device):

1A0Fh

0002h

Note:	QBLANK does not check the system operation Configuration bits as these bits are not set to '1' when a Chip Erase is performed.

5.2.13 QVER COMMAND

15 12	11 0
Opcode	Length

Field	Description
Opcode	Bh
Length	1h

The QVER command queries the version of the programming executive software stored in the test memory. The “version.revision” information is returned in the response‘s QE_Code using a single byte in the following format:

Main version is in the upper nibble and the revision is in the lower nibble (i.e., 23h stands for version 2.3 of the programming executive software).

Expected Response (2 words):

1BMNh (“MN” stands for version M.N)

0002h

5.3 Programming Executive Responses

The programming executive sends a response to the programmer for each command that it receives. The response indicates if the command was processed correctly. It includes any required response data or error data.

The programming executive response set is provided in Table 5-2. This table contains the opcode, mnemonic and description for each response. The response format is described in **Section 5.3.1 “Response Format”**.

TABLE 5-2: PROGRAMMING EXECUTIVE RESPONSE OPCODES

Opcode	Mnemonic	Description
1h	PASS	Command successfully processed.
2h	FAIL	Command unsuccessfully processed.
3h	NACK	Command not known.

5.3.1 RESPONSE FORMAT

All programming executive responses have a general format, consisting of a two-word header and any required data for the command.

15 12	11 8	7 0
Opcode	Last_Cmd	QE_Code
Length		
D_1 (if applicable)		
...		
D_N (if applicable)		

Field	Description
Opcode	Response opcode.
Last_Cmd	Programmer command that generated the response.
QE_Code	Query code or error code.
Length	Response length in 16-bit words (includes 2 header words).
D_1	First 16-bit data word (if applicable).
D_N	Last 16-bit data word (if applicable).

5.3.1.1 Opcode Field

The opcode is a 4-bit field in the first word of the response. The opcode indicates how the command was processed (see Table 5-2). If the command was processed successfully, the response opcode is PASS. If there was an error in processing the command, the response opcode is FAIL and the QE_Code indicates the reason for the failure. If the command sent to the programming executive is not identified, the programming executive returns a NACK response.

5.3.1.2 Last_Cmd Field

The Last_Cmd is a 4-bit field in the first word of the response and it indicates the command that the programming executive processed. As the programming executive can process only one command at a time, this field is technically not required. However, it can be used to verify the programming executive correctly received the command that the programmer transmitted.

5.3.1.3 QE_Code Field

The QE_Code is a byte in the first word of the response. This byte is used to return data for query commands and error codes for all of the other commands.

When the programming executive processes one of the two query commands (QBLANK or QVER), the returned opcode is always PASS and the QE_Code holds the query response data.

Table 5-3 provides the format of the QE_Code for both queries.

TABLE 5-3: QE_Code FOR QUERIES

Query	QE_Code
QBLANK	0Fh = Code memory is NOT blank F0h = Code memory is blank
QVER	0xMN, where programming executive software version = M.N (i.e., 32h stands for version 3.2 of the programming executive software)

When the programming executive processes any command other than a query, the QE_Code represents an error code. Table 5-4 provides the supported error codes.

If a command is successfully processed, the returned QE_Code is set to 0h, which indicates that there was no error in the command processing. If the verify of the programming for the PROGP or PROGC command fails, the QE_Code is set to 1h. For all other programming executive errors, the QE_Code is 2h.

TABLE 5-4: QE_Code FOR NON-QUERY COMMANDS

QE_Code	Description
0h	No error.
1h	Verify failed.
2h	Other error.

5.3.1.4 Response Length

The response length indicates the length of the programming executive's response in 16-bit words. This field includes the two words of the response header.

With the exception of the response for the READP command, the length of each response is only two words.

The response to the READP command uses the packed instruction word format described in **Section 5.2.2 "Packed Data Format"**. When reading an odd number of program memory words (N odd), the response to the READP command is (3 * (N + 1)/2 + 2) words. When reading an even number of program memory words (N even), the response to the READP command is (3 * N/2 + 2) words.

5.4 Programming the Programming Executive to Memory

This section describes the programming of the programming executive to memory and also provides the procedure to perform this.

5.4.1 OVERVIEW

If it is determined that the programming executive is not present in the executive memory (as described in **Section 4.2 "Confirming the Presence of the Programming Executive"**), it must be programmed into the executive memory using ICSP, as described in **Section 3.0 "Device Programming – ICSP"**.

Storing the programming executive to executive memory is the same as normal programming of code memory. The executive memory should be erased and then the programming executive must be programmed 32 words at a time.

Erasing the last page of executive memory causes the FRC oscillator calibration settings and device diagnostic data in the Diagnostic and Calibration Words, at addresses, 8007F4h to 8007FEh, to be erased. To retain this data, the memory locations should be read and stored prior to erasing the executive memory, and then be reprogrammed in the last words of program memory. Table 5-5 provides this control flow.

TABLE 5-5: PROGRAMMING THE PROGRAMMING EXECUTIVE

Command (Binary)	Data (Hex)	Description
Step 1: Exit Reset vector and erase the executive memory.		
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
Step 2: Initialize pointers to read Diagnostic Words for storage in data RAM.		
0000	200800	MOV #0x80, W0
0000	880190	MOV W0, TBLPAG
0000	207F01	MOV #0x07F4, W1
0000	208002	MOV #0x0800, W2
0000	000000	NOP
Step 3: Repeat this step six times to read Diagnostic Words, storing them in data RAM starting at 0x800.		
0000	BA0191	TBLRDL [W1], W3
0000	000000	NOP
0000	000000	NOP
	BA8231	TBLRDH [W1++], W4
	000000	NOP
	000000	NOP
	781903	MOV W3, [W2++]
	000000	NOP
	781904	MOV W4, [W2++]
	000000	NOP
Step 4: Initialize the NVMCON to erase the executive memory.		
0000	2405A0	MOV #0x405A, W0
0000	883B00	MOV W0, NVMCON
Step 5: Initiate the erase cycle.		
0000	200800	MOV #0x80, W0
0000	880190	MOV W0, TBLPAG
0000	200001	MOV #0x00, W1
0000	000000	NOP
0000	BB0881	TBLWTL W1, [W1]
0000	000000	NOP
0000	000000	NOP
0000	A8E761	BSET NVMCON, #15
0000	000000	NOP
0000	000000	NOP

TABLE 5-5: PROGRAMMING THE PROGRAMMING EXECUTIVE (CONTINUED)

Command (Binary)	Data (Hex)	Description
Step 6: Repeat this step to poll the WR bit (bit 15 of NVMCON) until it is cleared by the hardware.		
0000	040200	GOTO 0x200
0000	000000	NOP
0000	803B02	MOV NVMCON, W2
0000	883C22	MOV W2, VISI
0000	000000	NOP
0000	000000	NOP
0001	<VISI>	Clock out contents of the VISI register
0000	000000	NOP
Step 7: Repeat Steps 5 and 6 to erase the rest of the executive memory. W1 should be incremented by 100h each time to point to the next four rows.		
Step 8: Initialize the NVMCON to program 32 instruction words.		
0000	240041	MOV #0x4004, W1
0000	883B01	MOV W1, NVMCON
Step 9: Initialize TBLPAG and the Write Pointer (W2).		
0000	200800	MOV #0x80, W0
0000	880190	MOV W0, TBLPAG
0000	EB0280	CLR W5
0000	000000	NOP
Step 10: Load W0:W2 with the next two words of packed programming executive code.		
0000	2<LSW0>0	MOV #<LSW0>, W0
0000	2<MSB1:MSB0>1	MOV #<MSB1:MSB0>, W1
0000	2<LSW1>2	MOV #<LSW1>, W2
Step 11: Set the Read Pointer (W6) and load the (next four write) latches.		
0000	EB0300	CLR W4
0000	000000	NOP
0000	BB0BB6	TBLWTL [W4++], [W5]
0000	000000	NOP
0000	000000	NOP
0000	BBDBB6	TBLWTH.B [W4++], [W5++]
0000	000000	NOP
0000	000000	NOP
0000	BBEBB6	TBLWTH.B [W4++], [++W5]
0000	000000	NOP
0000	000000	NOP
0000	BB1BB6	TBLWTL [W4++], [W5++]
0000	000000	NOP
0000	000000	NOP
Step 12: Repeat Steps 10 and 11, sixteen times, to load the write latches for the 32 instructions.		
Step 13: Initiate the programming cycle.		
0000	A8E761	BSET NVMCON, #15
0000	000000	NOP
0000	000000	NOP

TABLE 5-5: PROGRAMMING THE PROGRAMMING EXECUTIVE (CONTINUED)

Command (Binary)	Data (Hex)	Description
Step 14: Repeatedly read the NVMCON register and poll for WR bit to get cleared.		
0000	040200	GOTO 0x200
0000	000000	NOP
0000	803B02	MOV NVMCON, W2
0000	883C22	MOV W2, VISI
0000	000000	NOP
0000	000000	NOP
0001	<VISI>	Clock out contents of the VISI register.
0000	000000	NOP
Step 15: Reset the device internal PC.		
0000	040200	GOTO 0x200
0000	000000	NOP
Step 16: Repeat Steps 8 through 15 until all the last, but one (31) row of executive memory, has been programmed.		
Step 17: Repeat Steps 10 and 11, 12 times, to load the first 24 write latches.		
Step 18: Load the data RAM address into W4.		
0000	208004	MOV #0x0800, W4
0000	000000	NOP
Step 19: Load the saved Diagnostic Words in the write latch.		
0000	780134	MOV [W4++], W2
0000	000000	NOP
0000	7801B4	MOV [W4++], W3
0000	000000	NOP
0000	BB0A82	TBLWTL W2, [W5]
0000	000000	NOP
0000	000000	NOP
0000	BB9A83	TBLWTH W3, [W5++]
0000	000000	NOP
0000	000000	NOP
Step 20: Repeat Step 19 for each of the saved Diagnostic Words.		
Step 21: Repeat Steps 13 through 15.		

5.5 Programming Verification

After the programming executive is programmed to the executive memory using ICSP, it must be verified. Verify by reading out the contents of the executive memory and comparing it with the image of the programming executive stored in the programmer.

Read the contents of the executive memory using the same method described in **Section 3.9 "Reading Code Memory"**.

Table 5-6 provides the procedure for reading the executive memory.

Note:	In Step 2 of Table 5-6, the TBLPAG register must be set to 80h in order to read the executive memory.

TABLE 5-6: READING EXECUTIVE MEMORY

Command (Binary)	Data (Hex)	Description
Step 1: Exit the Reset vector.		
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
Step 2: Initialize TBLPAG and the Read Pointer (W6) for TBLRD instruction.		
0000	200800	MOV #0x80, W0
0000	880190	MOV W0, TBLPAG
0000	EB0300	CLR W6
Step 3: Initialize the Write Pointer (W7) to point to the VISI register.		
0000	207847	MOV #VISI, W7
0000	000000	NOP
Step 4: Read and clock out the contents of the next two locations of the executive memory through the VISI register using the REGOUT command.		
0000	BA1B96	TBLRDL [W6], [W7]
0000	000000	NOP
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
0000	BADBB6	TBLRDH [W6++], [W7]
0000	000000	NOP
0000	000000	NOP
0000	BAD3D6	TBLRDH.B [++W6], [W7--]
0000	000000	NOP
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
0000	BA0BB6	TBLRDL [W6++], [W7]
0000	000000	NOP
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
Step 5: Reset the device internal PC.		
0000	040200	GOTO 0x200
0000	000000	NOP
Step 6: Repeat Steps 4 and 5 until the entire executive memory is read.		

6.0 DEVICE ID

The Device ID region of memory can be used to determine the mask, variant and manufacturing information about the device. The Device ID region is 2 x 16 bits and it can be read using the READC command. This region of memory is read-only and can also be read when code protection is enabled.

Table 6-1 provides the Device ID for each device and Table 6-2 provides the Device ID registers. Table 6-3 describes the bit field of each register.

TABLE 6-1: DEVICE IDs

Device ID	DEVID
PIC24F08KA101	0D08h
PIC24F16KA101	0D01h
PIC24F08KA102	0D0Ah
PIC24F16KA102	0D03h
PIC24FV16KA301	4509h
PIC24F16KA301	4508h
PIC24FV16KA302	4503h
PIC24F16KA302	4502h
PIC24FV16KA304	4507h
PIC24F16KA304	4506h
PIC24FV32KA301	4519h
PIC24F32KA301	4518h
PIC24FV32KA302	4513h
PIC24F32KA302	4512h
PIC24FV32KA304	4517h
PIC24F32KA304	4516h

TABLE 6-2: PIC24FXXKA1XX/FVXXKA3XX DEVICE ID REGISTERS

Address	Name	Bit															
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FF0000h	DEVID	FAMID<7:0>								DEV<7:0>							
FF0002h	DEVREV	—												REV<3:0>			

TABLE 6-3: DEVICE ID BITS DESCRIPTION

Bit Field	Register	Description
FAMID<7:0>	DEVID	Encodes the family ID of the device.
DEV<7:0>	DEVID	Encodes the individual ID of the device.
REV<3:0>	DEVREV	Encodes the revision number of the device.

6.1 Checksums

6.1.1 CHECKSUM COMPUTATION

Checksums for the PIC24FXXKA1XX/FVXXKA3XX family are 16 bits. The checksum is calculated by summing the following:

- Contents of the code memory locations
- Contents of the Configuration registers

Table 6-4 describes how to calculate the checksum for each device.

All memory locations are summed, one byte at a time, using only their native data size. More specifically, Configuration registers are summed by adding the lower two bytes of these locations (the upper byte is ignored) while the code memory is summed by adding all three bytes of the code memory.

TABLE 6-4: CHECKSUM COMPUTATION

Device	Read Code Protection	Checksum Computation	Erased Checksum Value	Chip Checksum with 0xAAAAAA at 0x00 Location and at Last Location
PIC24FV32KA30X(1)	Disabled	CFGB + SUM (0:0057FE)	0x8158	0x7F5A
	Enabled	0	0x0000	0x0000
PIC24FV16KA30X(1)	Disabled	CFGB + SUM (0:002BFE)	0xC358	0xC15A
	Enabled	0	0x0000	0x0000
PIC24F16KA1XX	Disabled	CFGB + SUM (0:002BFE)	0xC334	0xC136
	Enabled	0	0x0000	0x0000
PIC24F08KA1XX	Disabled	CFGB + SUM (0:0015FE)	0xE434	0xE236
	Enabled	0	0x0000	0x0000

Legend: Item Description
SUM[a:b] = Byte sum of locations, a to b inclusive (all 3 bytes of code memory)
CFGB = Configuration Block (masked):
For PIC24FXXKA1 devices: Byte sum of (FBS & 0x000F + FGS & 0x0003 + FOSCSEL & 0x0087 + FOSC & 0x00FF + FWDT & 0x00DF + FPOR & 0x00FB + FICD & 0x00C3 + FDS & 0x00FF)
For PIC24FVXXKA3 devices: Byte sum of (FBS & 0x000F + FGS & 0x0003 + FOSCSEL & 0x00E7 + FOSC & 0x00FF + FWDT & 0x00FF + FPOR & 0x00FF + FICD & 0x0083 + FDS & 0x00DF)

Note 1: Includes PIC24FVXXKA3XX devices.

7.0 AC/DC CHARACTERISTICS AND TIMING REQUIREMENTS

TABLE 7-1: STANDARD OPERATING CONDITIONS

Standard Operating Conditions Operating Temperature: 0°C to +70°C and programming: +25°C is recommended.						
Param No.	**Symbol**	**Characteristic**	**Min**	**Max**	**Units**	**Conditions**
D111	VDD	Supply Voltage During Programming	2.0	3.60	V	Normal programming, PIC24FXXKA1XX and PIC24FXXKA3XX
			VDDCORE	5.00	V	Normal programming, PIC24FVXXKA3XX
D112	IPP	Programming Current on $\overline{MCLR}$	—	50	μA	
D113	IDDP	Supply Current During Programming	—	2	mA	
D031	VIL	Input Low Voltage	VSS	0.2 VDD	V	
D041	VIH	Input High Voltage	0.8 VDD	VDD	V	
D042	VIHH	Programing Voltage on VPP	VDD + 1.5	9	V	
D080	VOL	Output Low Voltage	—	0.4	V	IOL = 8.5 mA @ 3.6V
D090	VOH	Output High Voltage	1.4	—	V	IOH = -3.0 mA @ 3.6V
D012	CIO	Capacitive Loading on I/O pin (PGDx)	—	50	pF	To meet AC specifications
D100	VBULK	Bulk Erase Voltage	2.7	—	V	PIC24F(V)XXKA3XX
			2.0	—	V	PIC24FXXKA1XX
P1	TPGC	Serial Clock (PGCx) Period	125	—	ns	ICSP™ mode
			250	—	ns	Enhanced ICSP mode
P1A	TPGCL	Serial Clock (PGCx) Low Time	50	—	ns	ICSP mode
			100	—	ns	Enhanced ICSP mode
P1B	TPGCH	Serial Clock (PGCx) High Time	50	—	ns	ICSP mode
			100	—	ns	Enhanced ICSP mode
P2	TSET1	Input Data Setup Time to Serial Clock ↑	15	—	ns	
P3	THLD1	Input Data Hold Time from PGCx ↑	15	—	ns	
P4	TDLY1	Delay Between 4-Bit Command and Command Operand	40	—	ns	
P4A	TDLY1A	Delay Between 4-Bit Command Operand and the Next 4-Bit Command	40	—	ns	
P5	TDLY2	Delay Between Last PGCx ↓ of Command Byte and First PGCx ↑ of Read of Data Word	20	—	ns	
P6	TSET2	VDD ↑ Setup Time to $\overline{MCLR}$ ↑	100	—	ns	
P7	THLD2	Input Data Hold Time from $\overline{MCLR}$ ↑ VPP↓ (from VIHH to VIH)	25	—	ms	
P8	TDLY3	Delay Between Last PGCx ↓ of Command Byte and PGDx ↑ by Programming Executive	12	—	μs	
P9	TDLY4	Programming Executive Command Processing Time	40	—	μs	
P10	TDLY6	PGCx Low Time After Programming	400	—	ns	
P11	TDLY7	Chip Erase Time	2.5	—	ms	
P12	TDLY10	Page (4 rows) Erase Time	2.5	—	ms	
P13	TDLY9	Row Programming Time	1.25	—	ms	
P14	TR	$\overline{MCLR}$ Rise Time to Enter ICSP™ mode	—	1.0	μs	
P15	TVALID	Data Out Valid from PGCx ↑	10	—	ns	
P16	TDLY10	Delay Between Last PGCx ↓ and $\overline{MCLR}$ ↓	0	—	s	—

TABLE 7-1: STANDARD OPERATING CONDITIONS (CONTINUED)

Standard Operating Conditions Operating Temperature: 0°C to +70°C and programming: +25°C is recommended.						
Param No.	**Symbol**	**Characteristic**	**Min**	**Max**	**Units**	**Conditions**
P17	THLD3	$\overline{MCLR}$ ↓ to VDD ↓	—	100	ns	—
P18	TKEY1	Delay Between First $\overline{MCLR}$ ↓ and First PGCx ↑ for Key Sequence on PGDx	1	—	ms	—
P19	TKEY2	Delay Between Last PGCx ↓ for Key Sequence on PGDx and Second $\overline{MCLR}$ ↑	1	—	ms	—
P20	TDLY11	Delay Between PGDx ↓ by Programming Executive and First PGCx↑ of Reception of Response	23	—	μs	—
P21	TDLY12	Delay Between Programming Executive Command Response Words	8	—	ns	—

APPENDIX A: REVISION HISTORY

Rev A Document (6/2008)

Original version of this document, covering PIC24FXXKA1XX and PIC24XXKA20X family devices.

Rev B Document (4/2011)

Adds PIC24FVXXKA3XX family devices; removes PIC24FXXKA20X devices for inclusion in their own programming specification (DS399XX). These changes occur throughout the entire document, except as otherwise noted; programming algorithms remain unchanged.

Updated Figure 2-3 to reflect the deletion of 14-pin packaging specific to PIC24FXXKA20X devices and the addition of 44-pin packaging for PIC24FVXXKA3XX devices.

Revised **Section 2.0 "Programming Overview of the PIC24FXXKA1XX/FVXXKA3XX Family"** to include new information on voltage regulators for PIC24FVXXKA3XX devices. Reorganized Table 2-3 to combine information.

Corrects **Section 4.2 "Confirming the Presence of the Programming Executive"** and **Section 5.4 "Programming the Programming Executive to Memory"** regarding verifying and programming the Programming Executive code.

Revised Table 4-2 with new Configuration bit descriptions specific to PIC24FVXXKA3XX devices and removing descriptions specific to PIC24FXXKA20X devices.

Updated **Section 6.0 "Device ID"** with new device ID and checksum information. Corrected existing checksum information for PIC24FXXKA1XX devices.

Updated **Section 7.0 "AC/DC Characteristics and Timing Requirements"** for new specifications related to programming voltage for PIC24FVXXKA3XX devices (D111) and Enhanced ICSP timing specification (P1, P1A and P1B). Modified erase and programming times (P11, P12 and P13), and initial key delay for entering programming modes (P18).

Other minor typographic corrections throughout.

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

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

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

Trademarks

Printed on recycled paper.

ISBN: 978-1-61341-074-5

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

Microchip received ISO/TS-16949:2002 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

Worldwide Sales and Service

AMERICAS

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

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

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

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

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

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

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

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

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

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

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

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

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

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

China - Shenzhen
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

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

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

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

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

ASIA/PACIFIC

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

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

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

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

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

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

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

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

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

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

Taiwan - Hsin Chu
Tel: 886-3-6578-300
Fax: 886-3-6578-370

Taiwan - Kaohsiung
Tel: 886-7-213-7830
Fax: 886-7-330-9305

Taiwan - Taipei
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

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

EUROPE

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

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

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

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

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

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

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

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

02/18/11