dsPIC33EVXXXGM00X/10X
Flash Programming Specification

1.0 DEVICE OVERVIEW

This document defines the programming specification for the dsPIC33EVXXXGM00X/10X 16-bit Digital Signal Controller (DSC) families. This programming specification is required only for those developing programming support for the following devices:

- dsPIC33EV64GM002
- dsPIC33EV64GM004
- dsPIC33EV64GM006
- dsPIC33EV64GM102
- dsPIC33EV64GM104
- dsPIC33EV64GM106
- dsPIC33EV256GM002
- dsPIC33EV256GM004
- dsPIC33EV256GM006
- dsPIC33EV256GM102
- dsPIC33EV256GM104
- dsPIC33EV256GM106
- dsPIC33EV128GM002
- dsPIC33EV128GM004
- dsPIC33EV128GM006
- dsPIC33EV128GM102
- dsPIC33EV128GM104
- dsPIC33EV128GM106

Customers only using these devices for application development should use development tools that already provide support for device programming.

Topics covered include:

- **Section 1.0 “Device Overview”**
- **Section 2.0 “Programming Overview”**
- **Section 3.0 “Device Programming – ICSP”**
- **Section 4.0 “Device Programming – Enhanced ICSP”**
- **Section 5.0 “Programming the Programming Executive to Memory”**
- **Section 6.0 “The Programming Executive”**
- **Section 7.0 “Device ID”**
- **Section 8.0 “Checksum Computation”**
- **Section 9.0 “AC/DC Characteristics and Timing Requirements”**

2.0 PROGRAMMING OVERVIEW

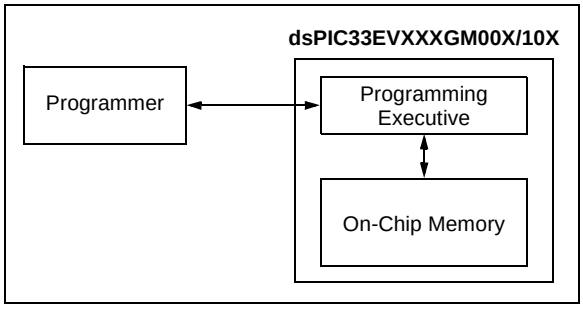
There are two methods of programming that are discussed in this programming specification:

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

The ICSP programming method is the most direct method to program the device; however, it is also the slower of the two methods. It provides native, low-level programming capability to erase, program and verify the device.

The Enhanced ICSP protocol uses a faster method that takes advantage of the Programming Executive (PE), as illustrated in Figure 2-1. The PE provides all the necessary functionality to erase, program and verify the chip through a small command set. The command set allows the programmer to program a dsPIC33EVXXXGM00X/10X device without dealing with the low-level programming protocols.

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



This programming specification is divided into two major sections that describe the programming methods independently. **Section 3.0 “Device Programming – ICSP”** describes the ICSP method. **Section 4.0 “Device Programming – Enhanced ICSP”** describes the Enhanced ICSP method.

2.1 Required Connections

These devices require specific connections for programming to take place. These connections include power, VCAP, MCLR and one programming pin pair (PGEDx/PGECx). Table 2-1 describes these connections (refer to the specific device data sheet for pin descriptions and power connection requirements).

2.2 Power Requirements

All dsPIC33EVXXXGM00X/10X devices, power their core digital logic at a nominal 1.8V. All devices in the dsPIC33EVXXXGM00X/10X families incorporate an on-chip regulator that allows the device to run its core logic from VDD.

The regulator provides power to the core from the VDD pins. A low-ESR capacitor (such as ceramic or tantalum) must be connected to the VCAP pin (see Figure 2-2 and Table 2-1). This maintains the stability of the regulator. The specifications for core voltage and capacitance are listed in **Section 9.0 "AC/DC Characteristics and Timing Requirements"**.

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

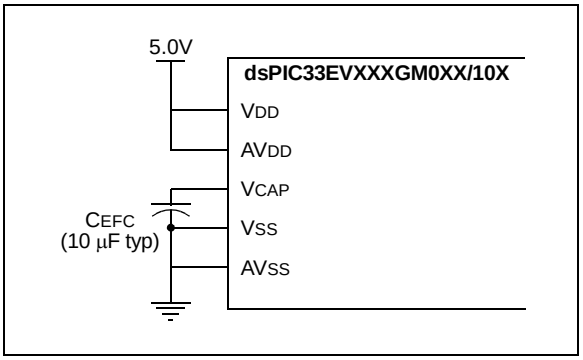


TABLE 2-1: PINS USED DURING PROGRAMMING

Pin Name	Pin Type	Pin Description
MCLR	I	Programming Enable
VDD and AVDD(1)	P	Power Supply(1)
VSS and AVSS(1)	P	Ground(1)
VCAP	P	CPU Logic Filter Capacitor Connection
PGECx	I	Programming Pin Pair: Serial Clock
PGEDx	I/O	Programming Pin Pair: Serial Data

Legend: I = Input O = Output P = Power

Note 1: All power supply and ground pins must be connected, including AVDD and AVSS.

2.3 Pin Diagrams

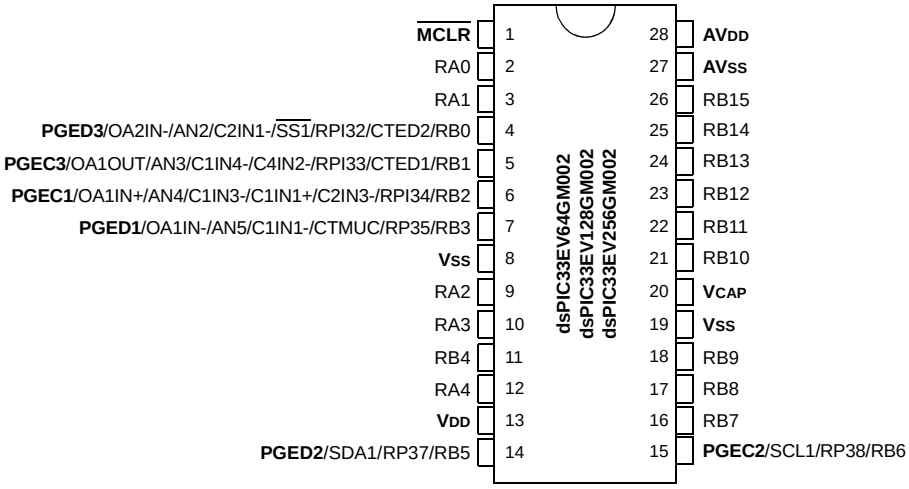
Figure 2-3 through Figure 2-5 provide the pin diagrams for the dsPIC33EVXXXGM00X/10X families. The pins that are required for programming are listed in Table 2-1 and are indicated in bold text in the figures. Refer to the appropriate device data sheet for complete pin descriptions.

2.3.1 PGECx AND PGEDx PIN PAIRS

All devices in the dsPIC33EVXXXGM00X/10X families have three separate pairs of programming pins, labeled as PGEC1/PGED1, PGEC2/PGED2 and PGEC3/PGED3. Any one of these pin pairs may be used for device programming by either ICSP or Enhanced ICSP. Unlike voltage supply and ground pins, it is not necessary to connect all three pin pairs to program the device. However, the programming method must use both pins of the same pair.

FIGURE 2-3: PIN DIAGRAMS

28-Pin SPDIP/SOIC/SSOP



28-Pin QFN-S

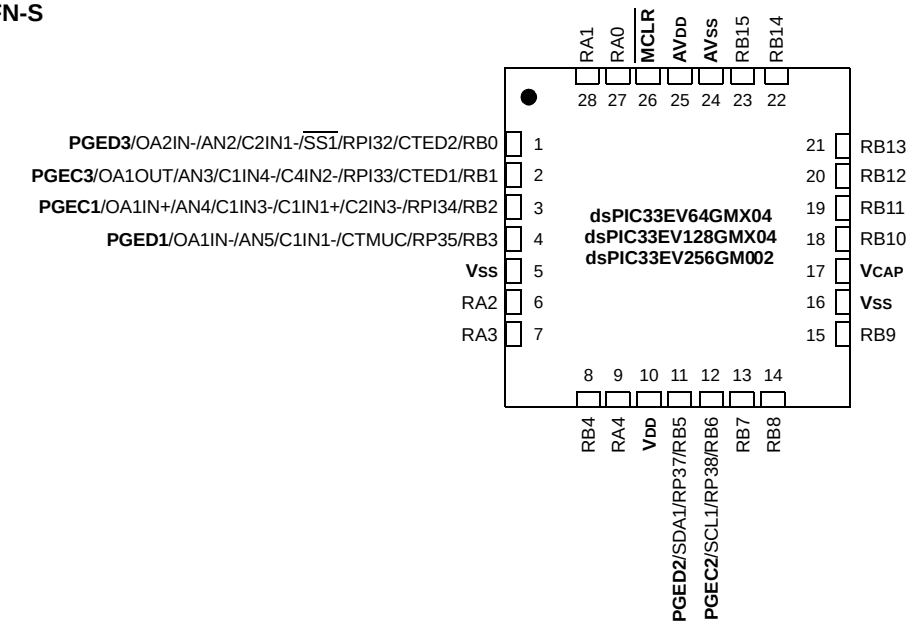


FIGURE 2-4: PIN DIAGRAMS (CONTINUED)

44-Pin TQFP/TLA/QFN

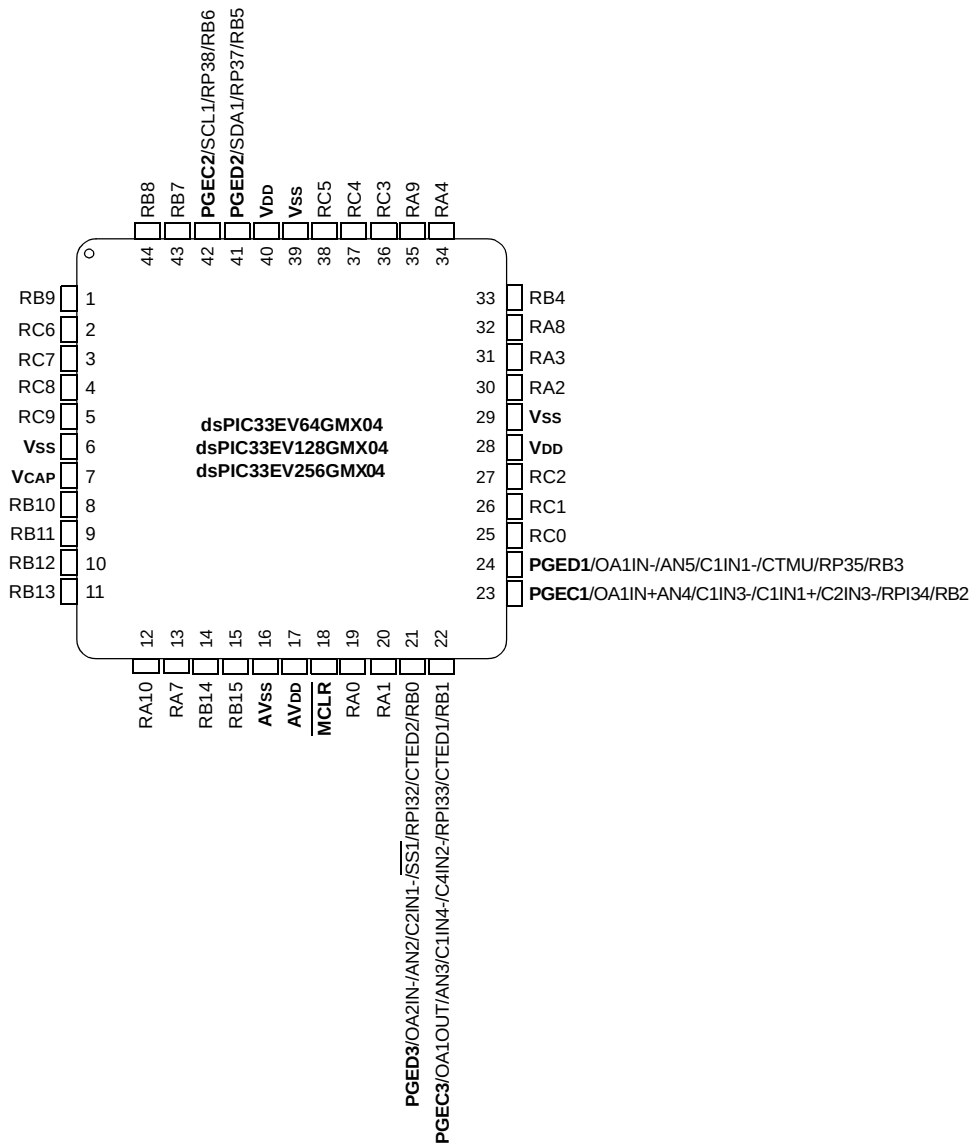
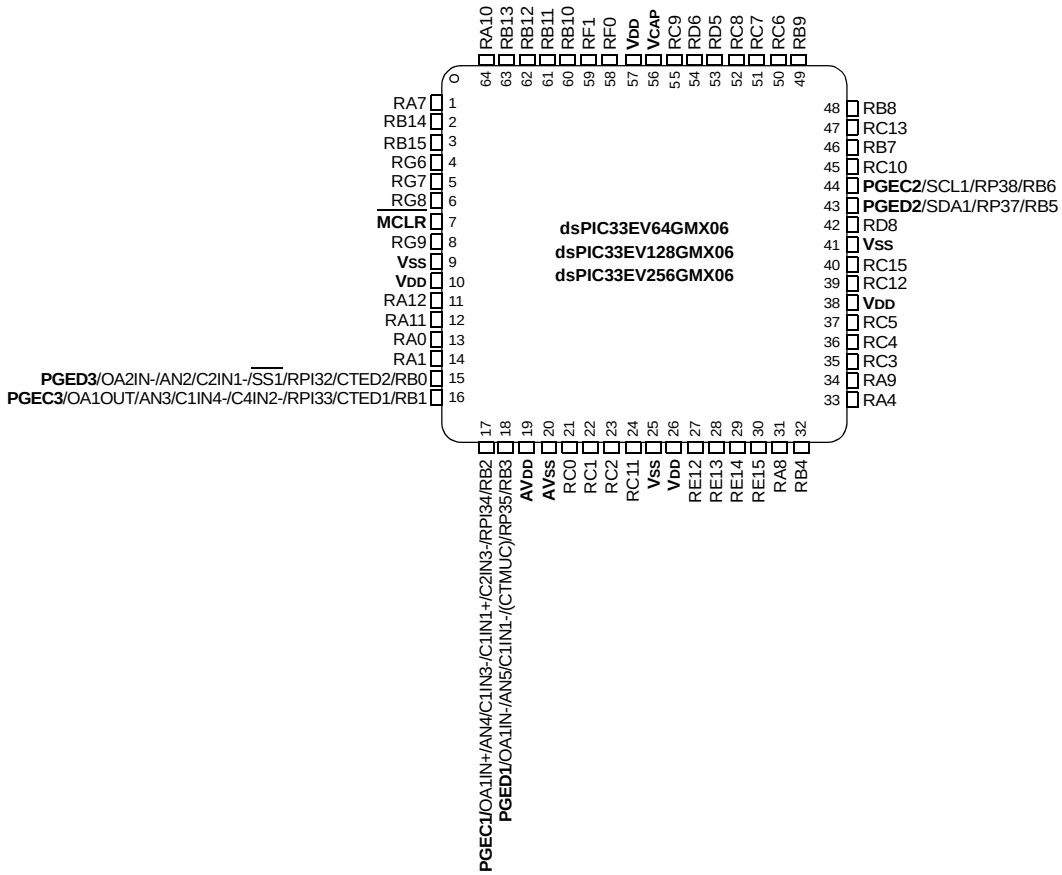


FIGURE 2-5: PIN DIAGRAMS (CONTINUED)

64-Pin TQFP/QFN



2.4 Program Memory Write/Erase Requirements

The program Flash memory has a specific write/erase requirement that must be adhered to for proper device operation. The rule is that any given word in memory must not be written without first erasing the page in which it is located. Thus, the easiest way to conform to this rule is to write all the data in a programming block within one write cycle. The programming methods specified in this document comply with this requirement.

Note: A program memory bit can be programmed from '1' to '0' only.

2.5 Memory Map

The program memory map extends from 000000h to FFFFFEh. Code memory is located at the start of the memory map. The last locations of implemented code memory are reserved for the device Configuration bits.

Table 2-2 lists the code memory size, the size of the erase blocks and the number of erase blocks present in each device variant.

Locations, 800200h through 800BFEh, are reserved for executive code memory. This region stores the PE and the debugging executive, which is used for device programming. This region of memory cannot be used to store user code. See **Section 6.0 "The Programming Executive"** for more information. The special latches used for device programming are located at addresses, FA0000h and FA0002h.

Locations, FF0000h and FF0002h, are reserved for the Device ID Word registers. These bits can be used by the programmer to identify which device type is being programmed. They are described in **Section 7.0 "Device ID"**. The Device ID registers read out normally, even after code protection is applied.

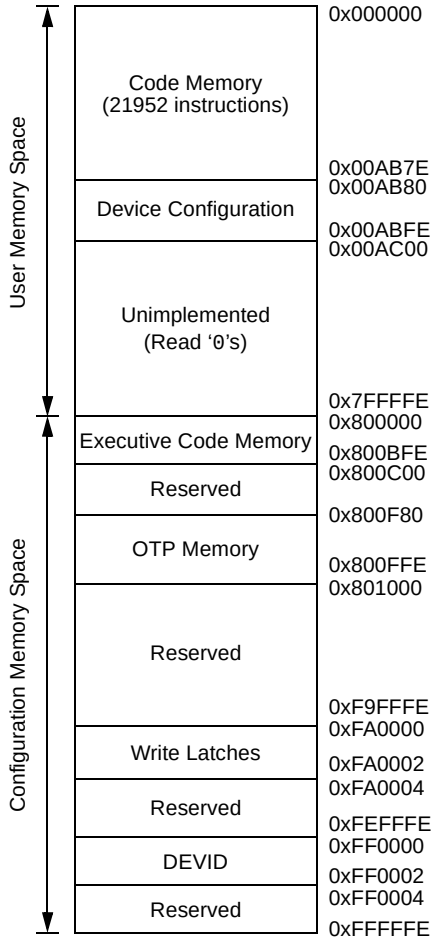
The locations, 800F80h-800FFEh, are the One-Time-Programmable (OTP) memory area. The user OTP Words can be used for storing product information, such as serial numbers, system manufacturing dates, manufacturing lot numbers and other application-specific information. They are described in **Section 2.6.3 "OTP (One-Time-Programmable) Memory"**.

Figure 2-6 through Figure 2-8 show a generic memory map for all devices. See the **"Memory Organization"** chapter in the specific device data sheet for more information.

TABLE 2-2: CODE MEMORY SIZE

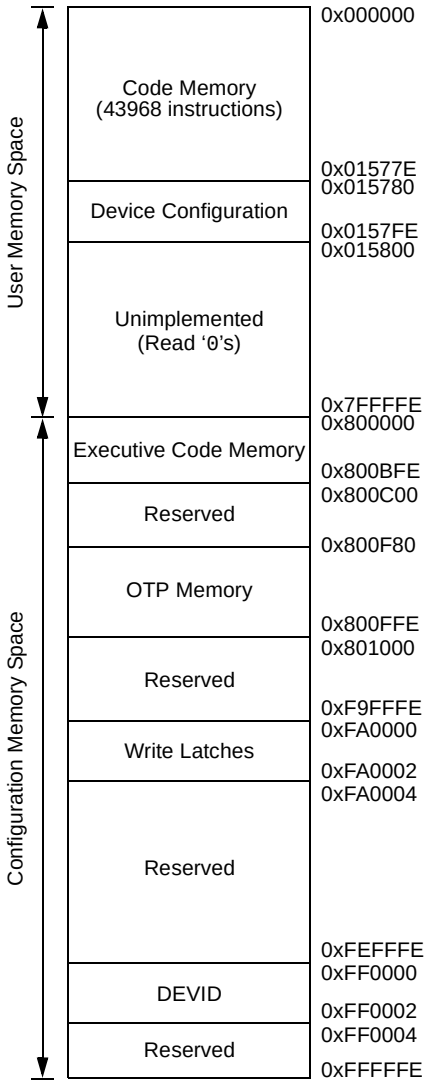
Device Family	User Memory Limit (Instruction Words)	Write Blocks/ No. of Rows	Erase Blocks/ No. of Pages
dsPIC33EV64GMX02	AB7E (21,952)	343	43
dsPIC33EV64GMX04			
dsPIC33EV64GMX06			
dsPIC33EV128GMX02	1577E (43,968)	687	86
dsPIC33EV128GMX04			
dsPIC33EV128GMX06			
dsPIC33EV256GMX02	2AB7E (87,488)	1367	171
dsPIC33EV256GMX04			
dsPIC33EV256GMX06			

FIGURE 2-6: PROGRAM MEMORY MAP FOR dsPIC33EV64GM00X/10X DEVICES(1)



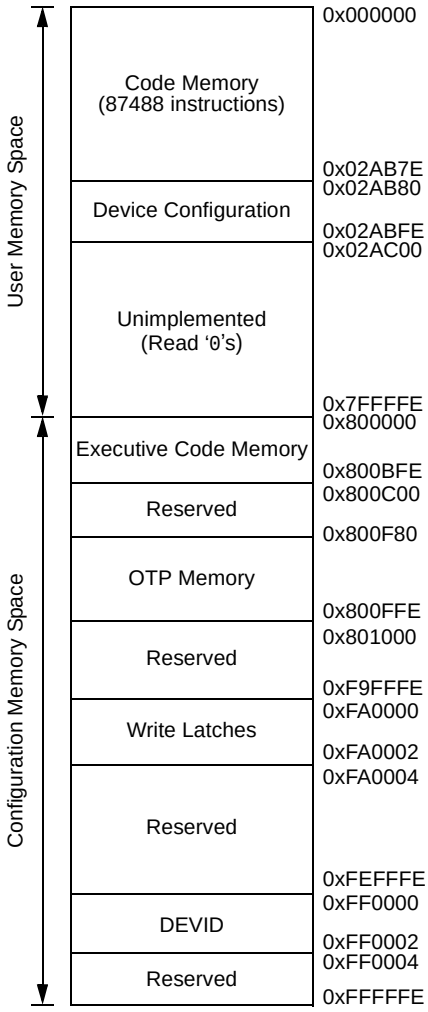
Note 1: Memory areas are not shown to scale.

FIGURE 2-7: PROGRAM MEMORY MAP FOR dsPIC33EV128GM00X/10X DEVICES(1)



Note 1: Memory areas are not shown to scale.

FIGURE 2-8: PROGRAM MEMORY MAP FOR dsPIC33EV256GM00X/10X DEVICES(1)



Note 1: Memory areas are not shown to scale.

2.6 Configuration Bits

2.6.1 OVERVIEW

The Configuration bits are stored in the last page location of implemented program memory. These bits can be set or cleared to select various device configurations. There are two types of Configuration bits: system operation bits and code-protect bits. The system operation bits determine the power-on settings for system level components, such as the oscillator and the Watchdog Timer. The code-protect bits prevent program memory from being read and written.

Table 2-3 and Table 2-4 lists the Configuration register address range for each device.

Table 2-5 shows the Configuration register map. Refer to the **"Special Features"** chapter in the specific device data sheet for more information.

2.6.2 CODE-PROTECT CONFIGURATION BITS

The device implements an intermediate security feature defined by the FSEC register. The Boot Segment (BS) is the highest privileged segment and the General Segment (GS) is the lowest privileged segment. The total code memory can be split into BS or GS. The size of the segments is determined by the BSLIM<12:0> bits. The relative location of the segments within user space does not change, such that the BS (if present) occupies the memory area just after the Interrupt Vector Table (IVT) and the GS occupies the space just after BS (or if the Alternate Vector Table (AIVT) is enabled, just after the AIVT VS). The Configuration Segment (or CS) is a small segment (less than a page, typically just one row) within the code memory address space that contains all user configuration data.

TABLE 2-3: CONFIGURATION WORD ADDRESSES (1 OF 2)

Device	FSEC	FBSLIM	FSIGN	FOSCSEL	FOSC	FWDT	FPOR
dsPIC33EV64GMX02	0xAB80	0xAB90	0xAB94	0xAB98	0xAB9C	0xABA0	0xABA4
dsPIC33EV64GMX04							
dsPIC33EV64GMX06							
dsPIC33EV128GMX02	0x15780	0x15790	0x15794	0x15798	0x1579C	0x157A0	0x157A4
dsPIC33EV128GMX04							
dsPIC33EV128GMX06							
dsPIC33EV256GMX02	0x2AB80	0x2AB90	0x2AB94	0x2AB98	0x2AB9C	0x2ABA0	0x2ABA4
dsPIC33EV256GMX04							
dsPIC33EV256GMX06							

TABLE 2-4: CONFIGURATION WORD ADDRESSES (2 OF 2)

Device	FICD	FDMTINTVL	FDMTINTVH	FDMTCNTL	FDMTCNTH	FDMT	FDEVOPT	FALTREG
dsPIC33EV64GMX02	0xABA8	0xABAC	0xABB0	0xABB4	0xABB8	0xABBC	0xABC0	0xABC4
dsPIC33EV64GMX04								
dsPIC33EV64GMX06								
dsPIC33EV128GMX02	0x157A8	0x157AC	0x157B0	0x157B4	0x157B8	0x157BC	0x157C0	0x157C4
dsPIC33EV128GMX04								
dsPIC33EV128GMX06								
dsPIC33EV256GMX02	0x2ABA8	0x2ABAC	0x2ABB0	0x2ABB4	0x2ABB8	0x2ABBC	0x2ABC0	0x2ABC4
dsPIC33EV256GMX04								
dsPIC33EV256GMX06								

TABLE 2-5: CONFIGURATION REGISTER MAP

Register Name	Bits 23-16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	
FSEC	—	AIVTDIS	—	—	—	CSS2	CSS1	CSS0	CWRP	GSS1	GSS0	GWRP	—	BSS2	B
FBSLIM	—	—	—	—	BSLIM<12:0>										
Reserved	—	r(2)	—	—	—	—	—	—	—	—	—	—	—	—	
FOSCSEL	—	—	—	—	—	—	—	—		IESO	—	—	—	—	FN
FOSC	—	—	—	—	—	—	—	—	PLLKEN	FCKSM1	FCKSM0	IOL1WAY	—	—	OS
FWDT	—	—	—	—	—	—	—	WDTWIN1	WDTWIN0	WINDIS	FWDTEN1	FWDTEN0	WDTPRE	WDTPS3	WI
FPOR	—	—	—	—	—	—	—	—	—	r(1)	—	—	—	—	
FICD	—	—	—	—	—	—	—	—	—	r(1)	—	—	—	—	
FDMTINTVL	—	DMTINTV<15:0>													
FDMTINTVH	—	DMTINTV<31:16>													
FDMTCNTL	—	DMTCNT<15:0>													
FDMCNTH	—	DMTCNT<31:16>													
FDMT	—	—	—	—	—	—	—	—	—	—	—	—	—	—	
FDEVOPT	—	—	—	—	—	—	—	—	—	—	—	—	—	ALTI2C1	
FALTREG	—	—	—	—	—		—	—	—	—	CTXT2<2:0>			—	

Legend: — = unimplemented, read as '1'; r = Reserved bit.

Note 1: This bit is reserved and must be programmed as '1'

2: This bit is reserved and must be programmed as '0'

TABLE 2-6: dsPIC33EVXXXGM00X/10X CONFIGURATION BITS DESCRIPTION

Bit Field	Register	Description
BWRP	FSEC	Boot Segment Write-Protect bit 1 = User program memory is not write-protected 0 = User program memory is write-protected
BSS<2:0>	FSEC	Boot Segment Code Flash Protection Level bits 11 = No protection (other than BWRP write protection) 10 = Standard security 0x = High security
GWRP	FSEC	General Segment Write-Protect bit 1 = User program memory is not write-protected 0 = User program memory is write-protected
GSS<1:0>	FSEC	General Segment Code Flash Protection Level bits 11 = No protection (other than GWRP write protection) 10 = Standard security 0x = High security
CWRP	FSEC	Configuration Segment Write-Protect bit 1 = Configuration Segment is not write-protected 0 = Configuration Segment is write-protected
CSS<2:0>	FSEC	Configuration Segment Code Flash Protection Level bits 111 = No protection (other than CWRP write protection) 110 = Standard security 10x = Enhanced security 0xx = High security
AIVTDIS	FSEC	Alternate Interrupt Vector Table Disable bit 1 = Disables AIVT 0 = Enables AIVT
BSLIM<12:0>	FBSLIM	Boot Segment Code Flash Page Address Limit bits Contains the page address of the first active General Segment page. The value to be programmed is the inverted page address, such that programming additional '0's can only increase the Boot Segment size (i.e., 0x1FFD = 2 Pages or 1024 instruction words).
FNOSC<2:0>	FOSCSEL	Initial Oscillator Source Selection bits 111 = Internal Fast RC (FRC) Oscillator with Postscaler 110 = Internal Fast RC (FRC) Oscillator with Divide-by-16 101 = LPRC Oscillator 100 = Secondary (LP) Oscillator 011 = Primary (XT, HS, EC) Oscillator with PLL 010 = Primary (XT, HS, EC) Oscillator 001 = Internal Fast RC (FRC) Oscillator with PLL 000 = FRC Oscillator
IESO	FOSCSEL	Two-Speed Oscillator Start-up Enable bit 1 = Starts up device with FRC, then automatically switches to the user-selected oscillator source when ready 0 = Starts up device with the user-selected oscillator source
POSCMD<1:0>	FOSC	Primary Oscillator Mode Select bits 11 = Primary Oscillator is disabled 10 = HS Crystal Oscillator mode 01 = XT Crystal Oscillator mode 00 = EC (External Clock) mode

TABLE 2-6: dsPIC33EVXXXGM00X/10X CONFIGURATION BITS DESCRIPTION (CONTINUED)

Bit Field	Register	Description
OSCIOFNC	FOSC	OSC2 Pin Function bit (except in XT and HS modes) 1 = OSC2 is the clock output 0 = OSC2 is the general purpose digital I/O pin
IOL1WAY	FOSC	Peripheral Pin Select (PPS) Configuration bit 1 = Allows only one reconfiguration 0 = Allows multiple reconfigurations
FCKSM<1:0>	FOSC	Clock Switching Mode 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
PLLKEN	FOSC	PLL Lock Wait Enable bit 1 = Clock switches to the PLL source; will wait until the PLL lock signal is valid 0 = Clock switch will not wait for the PLL lock
WDTPS<3:0>	FWDT	Watchdog Timer Postscaler bits 1111 = 1:32,768 1110 = 1:16,384 • • • 0001 = 1:2 0000 = 1:1
WDTPRE	FWDT	Watchdog Timer Prescaler bit 1 = 1:128 0 = 1:32
FWDTEN<1:0>	FWDT	Watchdog Timer Enable bits 11 = WDT is enabled in hardware 10 = WDT is controlled via the SWDTEN bit 01 = WDT is enabled only while device is active and is disabled in Sleep; SWDTEN bit is disabled 00 = WDT and SWDTEN are disabled
WINDIS	FWDT	Watchdog Timer Window Enable bit 1 = Watchdog Timer is in Non-Window mode 0 = Watchdog Timer is in Window mode
WDTWIN<1:0>	FWDT	Watchdog Timer Window Select bits 11 = WDT window is 25% of the WDT period 10 = WDT window is 37.5% of the WDT period 01 = WDT window is 50% of the WDT period 00 = WDT Window is 75% of the WDT period
BOREN	FPOR	Brown-out Reset (BOR) Detection Enable bit 1 = BOR is enabled 0 = BOR is disabled
ICS<1:0>	FICD	ICD Communication Channel Select bits 11 = Communicates on PGEC1 and PGED1 10 = Communicates on PGEC2 and PGED2 01 = Communicates on PGEC3 and PGED3 00 = Reserved, do not use
DMTINTV<15:0>	FDMTINTVL	Lower 16 bits of the 32-bit field that configures the DMT window interval bit.
DMTINTV<31:16>	FDMTINTVH	Upper 16 bits of the 32-bit field that configures the DMT window interval bit.
DMTCNT	FDMTCNTL	Lower 16 bits of the 32-bit field that configures the DMT instruction count time-out value bit.
DMTCNT	FDMCNTH	Upper 16 bits of the 32-bit field that configures the DMT instruction count time-out value bit.

TABLE 2-6: dsPIC33EVXXXGM00X/10X CONFIGURATION BITS DESCRIPTION (CONTINUED)

Bit Field	Register	Description
DMTEN	FDMT	Deadman Timer Enable bit 1 = Deadman timer is enabled and cannot be disabled by software 0 = Deadman timer is disabled and can be enabled by software
PWMLOCK	FDEVOPT	PWM Lock Enable bit 1 = Certain PWM registers may only be written after a key sequence 0 = PWM registers may be written without a key sequence
ALTI2C1	FDEVOPT	Alternate I^2C™ Pins for I2C1 bit 1 = I2C1 is mapped to the SDA1/SCL1 pins 0 = I2C1 is mapped to the ASDA1/ASCL1 pins
CTXT1<2:0>	FALTREG	Specifies the Alternate Working Register Set #1 Association with Interrupt Priority Level (IPL) bits 111 = Not assigned 110 = Alternate Register Set #1 assigned to IPL Level 7 101 = Alternate Register Set #1 assigned to IPL Level 6 100 = Alternate Register Set #1 assigned to IPL Level 5 011 = Alternate Register Set #1 assigned to IPL Level 4 010 = Alternate Register Set #1 assigned to IPL Level 3 001 = Alternate Register Set #1 assigned to IPL Level 2 000 = Alternate Register Set #1 assigned to IPL Level 1
CTXT2<2:0>	FALTREG	Specifies the Alternate Working Register Set #2 Association with Interrupt Priority Level (IPL) bits 111 = Not assigned 110 = Alternate Register Set #2 assigned to IPL Level 7 101 = Alternate Register Set #2 assigned to IPL Level 6 100 = Alternate Register Set #2 assigned to IPL Level 5 011 = Alternate Register Set #2 assigned to IPL Level 4 010 = Alternate Register Set #2 assigned to IPL Level 3 001 = Alternate Register Set #2 assigned to IPL Level 2 000 = Alternate Register Set #2 assigned to IPL Level 1

2.6.3 OTP (ONE-TIME-PROGRAMMABLE) MEMORY

dsPIC33EVXXXGM00X/10X devices contain 64 OTP Words, located at addresses, 800F80h through 800FFEh. The OTP Words can be utilized by the user for storing checksum, code revisions, product information, such as serial numbers, system manufacturing dates, manufacturing lot numbers and other application-specific information. These words can only be written once.

Note:	The OTP area is not cleared by only the erase command. This memory can be written only once.

3.0 DEVICE PROGRAMMING – ICSP

ICSP mode is a special programming protocol that allows you to read and write to device memory. The ICSP mode is the most direct method used to program the device, which is accomplished by applying control codes and instructions, serially to the device, using the PGECx and PGEDx pins. ICSP mode also has the ability to read the executive memory to determine if the Programming Executive (PE) is present. This mode is also able to write the PE to executive memory if it is missing and if the Enhanced ICSP mode is to be used.

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

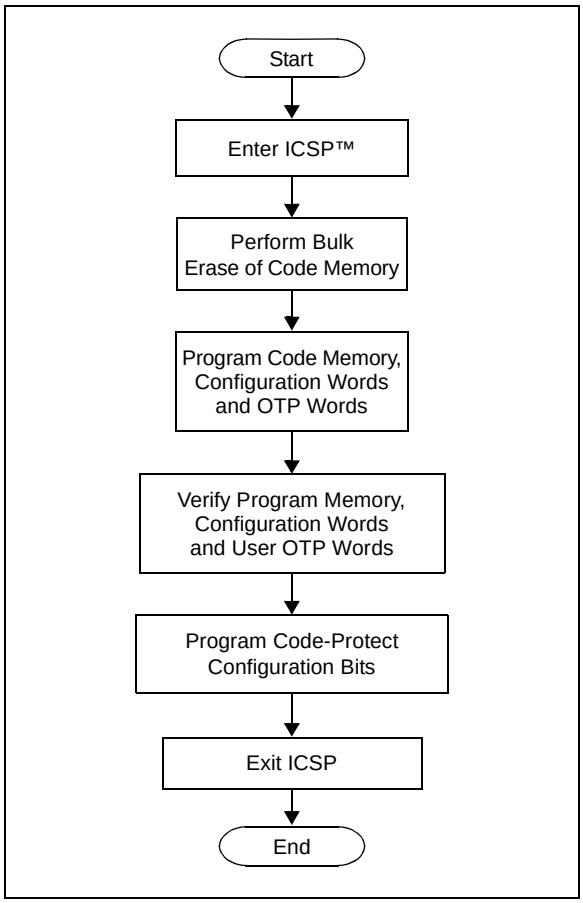
Note 1: During ICSP operation, the operating frequency of PGECx must not exceed 5 MHz.

2: ICSP mode is slower than Enhanced ICSP mode for programming.

3.1 Overview of the Programming Process

Figure 3-1 illustrates the high-level overview of the programming process. After entering ICSP mode, the first action is to Bulk Erase the code memory. Next, the code memory is programmed, followed by the device Configuration bits. Code memory (including the Configuration bits) is then verified to ensure that programming was successful. Then, the code-protect Configuration bits are programmed if required.

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



3.2 Entering ICSP Mode

As illustrated in Figure 3-2, entering ICSP Program/ Verify mode requires four steps:

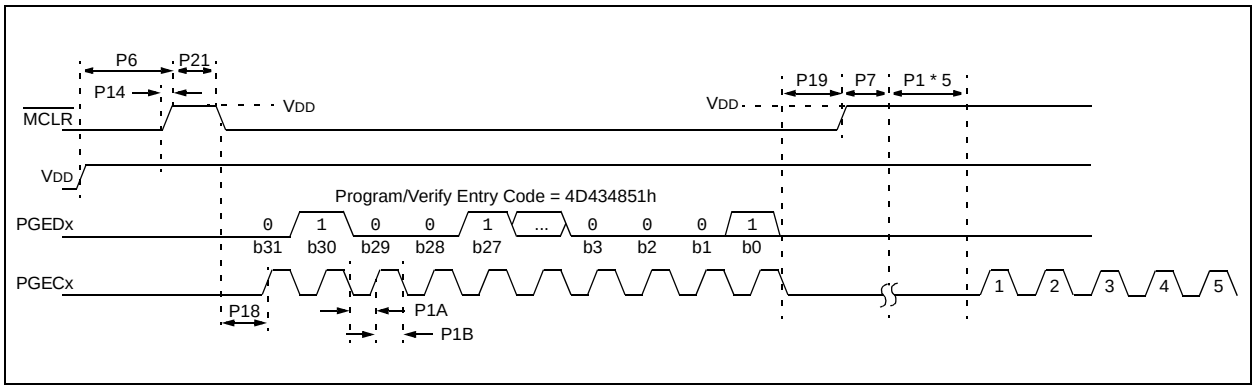
1. $\overline{\text{MCLR}}$ is briefly driven high and then low (P21).
2. A 32-bit key sequence is clocked into PGEDx. An interval of at least P18 must elapse before presenting the key sequence on PGEDx.
3. $\overline{\text{MCLR}}$ is held low during a specified period, P19, and then driven high.
4. After a P7 + 5 * P1 delay, five clock pulses must be generated on the PGECx pin.

Note: If a capacitor is present on the $\overline{\text{MCLR}}$ pin, the high time for entering ICSP mode can vary.

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 ICSP mode only if the sequence is valid. The Most Significant bit (MSb) of the most significant nibble must be shifted in first.

On successful ICSP mode entry, the program memory can be accessed and programmed in serial fashion.

FIGURE 3-2: ENTERING ICSP™ MODE



3.3 ICSP Operation

After entering into ICSP mode, the CPU is Idle. Execution of the CPU is governed by an internal state machine. A 4-bit control code is clocked in using PGECx and PGEDx, 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 through the VISI register.

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

4-Bit Control Code	Mnemonic	Description
0000	SIX	Shift in 24-bit instruction and execute.
0001	REGOUT	Shift out the VISI register.
0010-1111	N/A	Reserved.

3.3.1 SIX SERIAL INSTRUCTION EXECUTION

The SIX control code allows execution of the 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 3-3).

Note:	Data bits on PGEDx are latched on the rising edge of the PGECx clock pulses.

3.3.1.1 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 of 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 required CPU clocks to finish executing the instruction.

  Examples of two-cycle instructions are Table Read (TBLRD) and Table Write (TBLWT) 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 NOP executes the second half of the instruction and the other NOP stalls 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" instruction.

3.3.2 REGOUT SERIAL INSTRUCTION EXECUTION

The REGOUT control code allows the data to be extracted from the device in ICSP mode. It is used to clock the contents of the VISI register out of the device and over the PGEDx 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 3-4).

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

Note 1:	After the contents of VISI are shifted out, the devices maintain PGEDx 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 PGECx. For all data transmissions, the Least Significant bit (LSb) is transmitted first.

FIGURE 3-3: SIX SERIAL EXECUTION

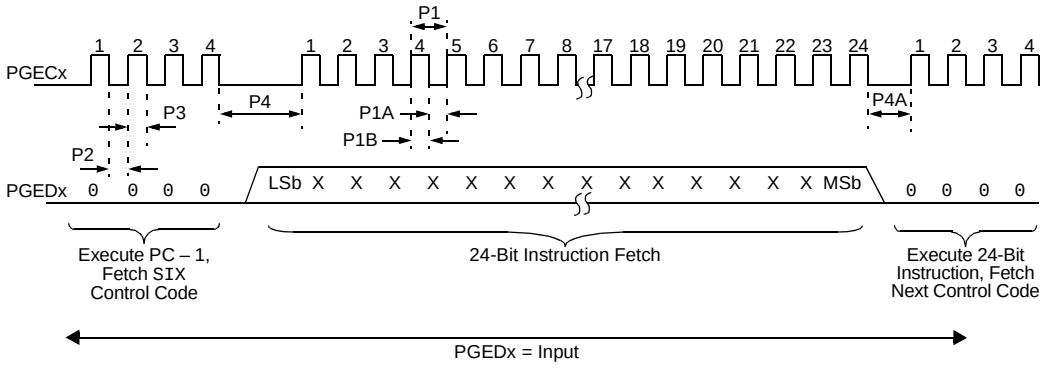
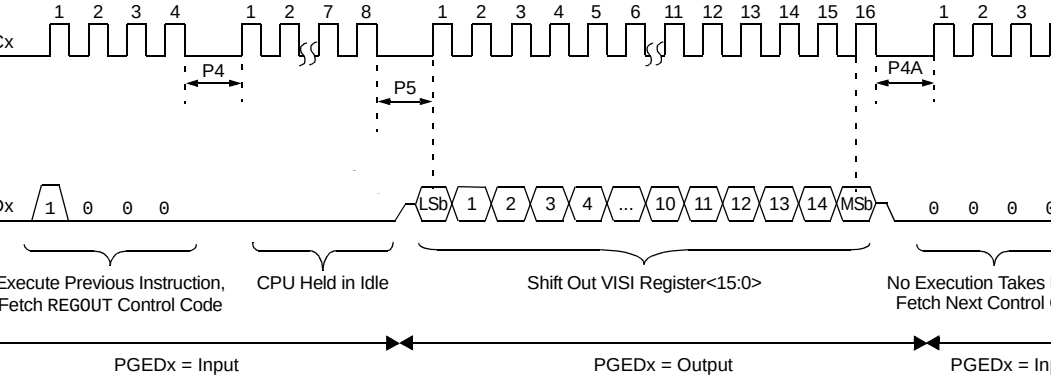


FIGURE 3-4: REGOUT SERIAL EXECUTION



3.4 Flash Memory Programming in ICSP Mode

3.4.1 PROGRAMMING OPERATIONS

Flash memory write/erase operations are controlled by the NVMCON register. Programming is performed by setting NVMCON to select the type of erase operation (Table 3-2) or write operation (Table 3-3) and initiating the programming by setting the WR control bit (NVMCON<15>).

The PGECx clock is required to complete the programming operation. The WR control bit is cleared by hardware when the operation is finished. Refer to **Section 9.0 "AC/DC Characteristics and Timing Requirements"** for information about the maximum time required for various programming operations.

TABLE 3-2: NVMCON ERASE OPERATIONS

NVMCON Value	Erase Operation
400Eh	Bulk Erase of user memory only (does not erase Device ID, Programming Executive memory and OTP Words).
4003h	Erases a page of program or Programming Executive memory.

TABLE 3-3: NVMCON WRITE OPERATIONS

NVMCON Value	Write Operation
4001h	Double-word programming operation.

3.4.2 STARTING AND STOPPING A PROGRAMMING CYCLE

For protection against accidental operations, the erase/write initiation sequence must be written to the NVMKEY register to allow any erase or program operation to proceed. The two instructions following the start of the programming sequence should be NOPs. To start an erase or write sequence, the following steps must be completed:

1. Write 55h to the NVMKEY register.
2. Write AAh to the NVMKEY register.
3. Set the WR bit in the NVMCON register.
4. Execute three NOP instructions.

The WR bit should be polled to generate enough clock cycles for the programming operation and to determine if the erase or write cycle has been completed.

REGISTER 3-1: NVMCON: NONVOLATILE MEMORY CONTROL REGISTER (REFERENCE ONLY)

R/SO-0(1)	R/W-0(1)	R/W-0(1)	R/W-0	U-0	U-0	R/W-0	R/W-0
WR	WREN	WRERR	NVMSIDL(2)	—	—	RPDF(6)	URERR(6)
bit 15							bit 8

U-0	U-0	U-0	U-0	R/W-0(1)	R/W-0(1)	R/W-0(1)	R/W-0(1)
—	—	—	—	NVMOP3(4)	NVMOP2(3,4)	NVMOP1(3,4)	NVMOP0(3,4)
bit 7							bit 0

Legend:	SO = Settable Only bit		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 15 **WR:** Write Control Bit(1)
- 1 = Initiates a Flash memory program or erase operation; the operation is self-timed and the bit is cleared by hardware once operation is complete
- 0 = Program or erase operation is complete and inactive

bit 14 **WREN:** Write Enable bit(1)
- 1 = Enables Flash program/erase operations
- 0 = Inhibits Flash program/erase operations

bit 13 **WRERR:** Write Sequence Error Flag bit(1)
- 1 = An improper program/erase sequence attempt or termination has occurred (bit is set automatically on any set attempt of the WR bit)
- 0 = The program/erase operation completed normally

bit 12 **NVMSIDL:** NVM Stop in Idle Control bit(2)
- 1 = Discontinues primary Flash operation when the device enters Idle mode
- 0 = Continues primary Flash operation when the device enters Idle mode

bit 11-10 **Unimplemented:** Read as '0'

bit 9 **RPDF:** Row Programming Data Format Control bit(6)
- 1 = Row data to be stored in RAM is in compressed format
- 0 = Row data to be stored in RAM is in uncompressed format

bit 8 **URERR:** Row Programming Data Underrun Error Flag bit(6)
- 1 = Row programming operation has been terminated due to a data underrun error
- 0 = No data underrun has occurred

bit 7-4 **Unimplemented:** Read as '0'

Note 1: These bits can only be reset on a POR.

2: If this bit is set, there will be minimal power savings (IIDLE), and upon exiting Idle mode, there is a delay (TVREG) before Flash memory becomes operational.

3: All other combinations of NVMOP<3:0> are unimplemented.

4: Execution of the PWRSAV instruction is ignored while any of the NVM operations are in progress.

5: Two adjacent words on a 4-word boundary are programmed during execution of this operation.

6: Not used in ICSP™ mode.

REGISTER 3-1: NVMCON: NONVOLATILE MEMORY CONTROL REGISTER (REFERENCE ONLY) (CONTINUED)

bit 3-0 **NVMOP<3:0>:** NVM Operation Select bits(1,3,4)

1111 = Reserved
1110 = User memory Bulk Erase operation
1101 = Reserved
1100 = Reserved
1011 = Reserved
1010 = Reserved
1001 = Reserved
1000 = Reserved
0111 = Reserved
0101 = Reserved
0100 = Reserved
0011 = Memory Page Erase operation
0010 = Memory row program operation(6)
0001 = Memory double-word operation(5)
0000 = Reserved

Note 1: These bits can only be reset on a POR.

2: If this bit is set, there will be minimal power savings (I_{IDLE}), and upon exiting Idle mode, there is a delay (T_{VREG}) before Flash memory becomes operational.

3: All other combinations of NVMOP<3:0> are unimplemented.

4: Execution of the PWRSAV instruction is ignored while any of the NVM operations are in progress.

5: Two adjacent words on a 4-word boundary are programmed during execution of this operation.

6: Not used in ICSP™ mode.

3.5 Erasing Program Memory

The procedure for erasing the entire code memory using Bulk Erase is shown in Figure 3-5.

Figure 3-6 shows the procedure for erasing a page of code memory.

Table 3-4 and Table 3-5 illustrate the ICSP programming process for Bulk Erase and Page Erase, respectively.

Note 1: Program memory must be erased before writing any data to program memory.

2: For Page Erase operations, the NVMCON value should be modified suitably according to Table 3-2. The NVMADR/U registers should be pointing to any of the locations of the page to be erased.

FIGURE 3-5: BULK ERASE FLOW

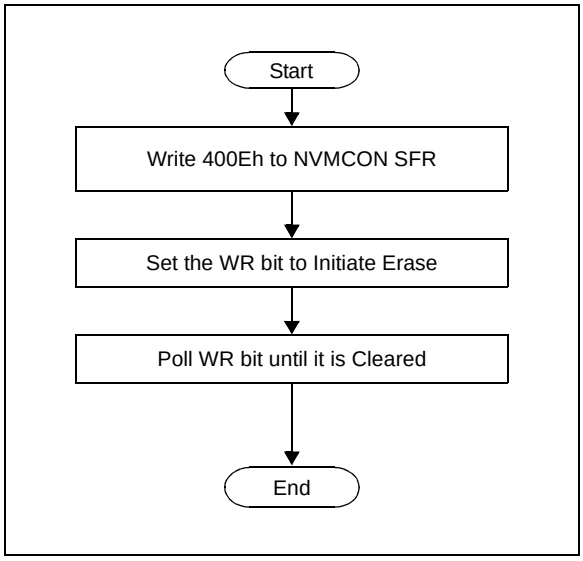


FIGURE 3-6: PAGE ERASE FLOW

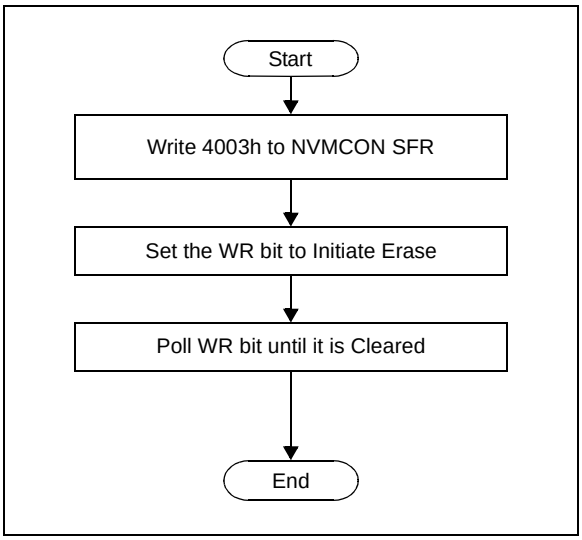


TABLE 3-4: SERIAL INSTRUCTION EXECUTION FOR BULK ERASE OF CODE MEMORY

Command (Binary)	Data (Hex)	Description
Step 1: Exit the Reset vector.		
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
Step 2: Set the NVMCON register to erase all user program memory.		
0000	2400EA	MOV #0x400E, W10
0000	88394A	MOV W10, NVMCON
0000	000000	NOP
0000	000000	NOP
Step 3: Initiate the erase cycle.		
0000	200551	MOV #0x55, W1
0000	883971	MOV W1, NVMKEY
0000	200AA1	MOV #0xAA, W1
0000	883971	MOV W1, NVMKEY
0000	A8E729	BSET NVMCON, #WR
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
Step 4: Generate clock pulses for the user memory Bulk Erase operation to complete until the WR bit is clear.		
0000	000000	NOP
0000	803940	MOV NVMCON, W0
0000	000000	NOP
0000	887C40	MOV W0, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
—	—	Repeat until the WR bit is clear.

TABLE 3-5: SERIAL INSTRUCTION EXECUTION FOR ERASING A PAGE OF CODE MEMORY

Command (Binary)	Data (Hex)	Description
Step 1: Exit the Reset vector.		
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
Step 2: Set the NVMADRU/NVMADR register pair to point to the correct page to be erased.		
0000	2xxxx3	MOV #DestinationAddress<15:0>, W3
0000	2xxxx4	MOV #DestinationAddress<23:16>, W4
0000	883953	MOV W3, NVMADR
0000	883964	MOV W4, NVMADRU
Step 3: Set the NVMCON register to erase the first page of executive memory.		
0000	24003A	MOV #0x4003, W10
0000	88394A	MOV W10, NVMCON
0000	000000	NOP
0000	000000	NOP
Step 4: Initiate the erase cycle.		
0000	200551	MOV #0x55, W1
0000	883971	MOV W1, NVMKEY
0000	200AA1	MOV #0xAA, W1
0000	883971	MOV W1, NVMKEY
0000	A8E729	BSET NVMCON, #WR
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
Step 5: Generate clock pulses for Page Erase operation to complete until the WR bit is clear.		
0000	000000	NOP
0000	803940	MOV NVMCON, W0
0000	000000	NOP
0000	887C40	MOV W0, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
—	—	Repeat until the WR bit is clear.

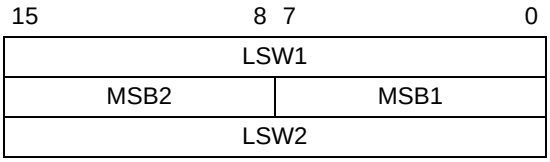
3.6 Writing Code Memory

Figure 3-8 provides a high-level description of the method for writing to code memory.

Two-word writes program the code memory with two instruction words at a time. Two words are loaded into the write latches located at address, FA0000h, and the destination address must be loaded to the NVMADRU/NVMADR register pair. Next, the WR bit is set to initiate the write sequence. Then, the WR bit must be checked for the sequence to be complete. This process continues for all the data to be programmed. Table 3-6 shows the ICSP programming details.

The data loaded into the programming latches must be in the packed format, as shown in Figure 3-8.

FIGURE 3-7: PACKED INSTRUCTION WORD FORMAT



LSWx: Least Significant 16 bits of instruction word
MSBx: Most Significant Byte of instruction word

Note: When the number of instruction words transferred is odd, MSB2 is zero and LSW2 cannot be transmitted.

FIGURE 3-8: PROGRAM CODE MEMORY FLOW

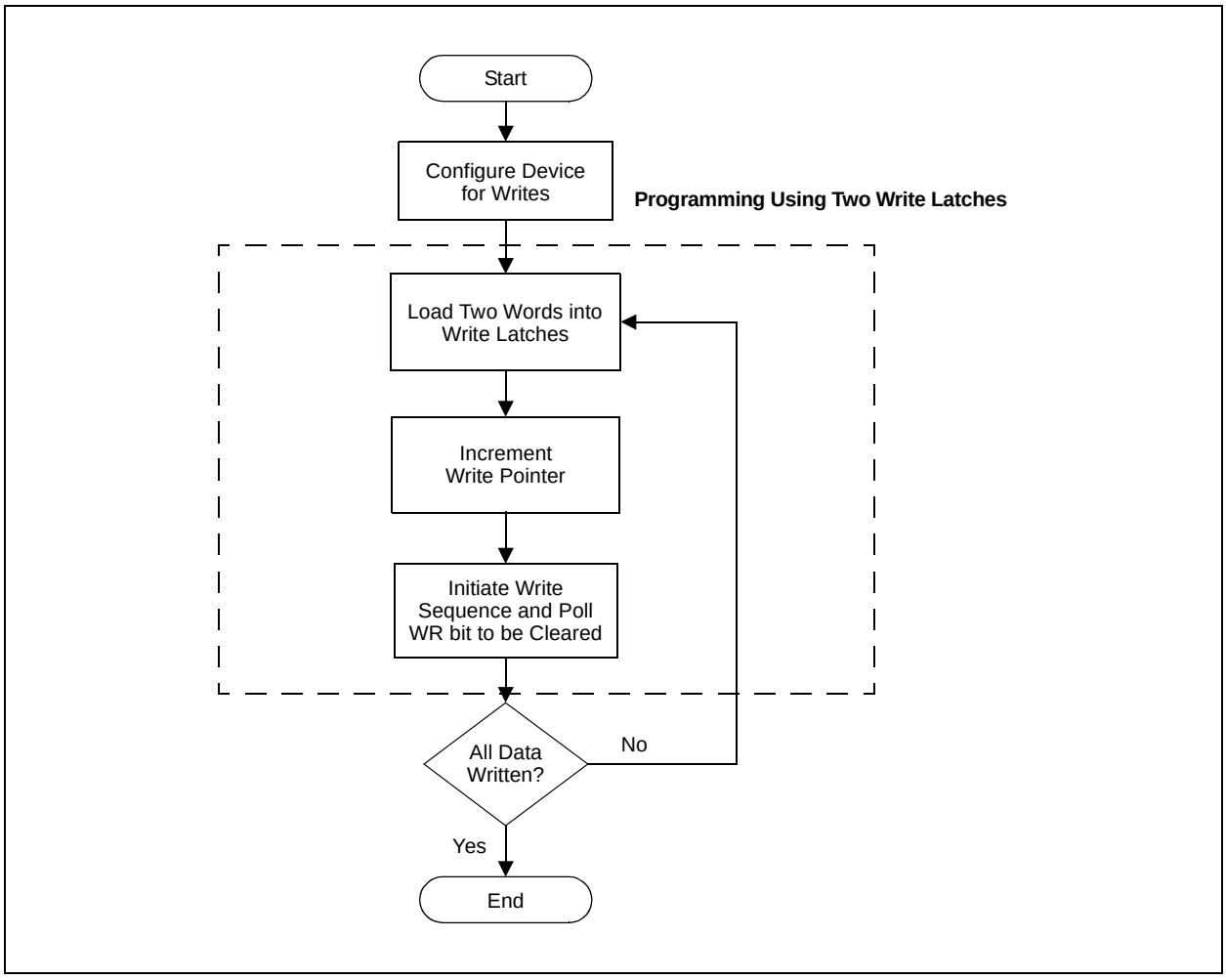


TABLE 3-6: SERIAL INSTRUCTION EXECUTION FOR PROGRAMMING CODE MEMORY: TWO-WORD LATCH WRITES

Command (Binary)	Data (Hex)	Description
Step 1: Exit the Reset vector.		
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
Step 2: Initialize the TBLPAG register for writing to the latches.		
0000	200FAC	MOV #0xFA, W12
0000	8802AC	MOV W12, TBLPAG
Step 3: Load W0:W2 with the next two packed instruction words to program.		
0000	2xxxx0	MOV #<LSW0>, W0
0000	2xxxx1	MOV #<MSB1:MSB0>, W1
0000	2xxxx2	MOV #<LSW1>, W2
Step 4: Set the Read Pointer (W6) and Write Pointer (W7), and load the (next set of) write latches.		
0000	EB0300	CLR W6
0000	000000	NOP
0000	EB0380	CLR W7
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	BB0B96	TBLWTL.W [W6], [W7]
0000	000000	NOP
0000	000000	NOP
Step 5: Set the NVMADRU/NVMADR register pair to point to the correct address.		
0000	2xxxx3	MOV #DestinationAddress<15:0>, W3
0000	2xxxx4	MOV #DestinationAddress<23:16>, W4
0000	883953	MOV W3, NVMADR
0000	883964	MOV W4, NVMADRU
Step 6: Set the NVMCON register to program two instruction words.		
0000	24001A	MOV #0x4001, W10
0000	000000	NOP
0000	88394A	MOV W10, NVMCON
0000	000000	NOP
0000	000000	NOP

TABLE 3-6: SERIAL INSTRUCTION EXECUTION FOR PROGRAMMING CODE MEMORY: TWO-WORD LATCH WRITES (CONTINUED)

Command (Binary)	Data (Hex)	Description
Step 7: Initiate the write cycle.		
0000	200551	MOV #0x55, W1
0000	883971	MOV W1, NVMKEY
0000	200AA1	MOV #0xAA, W1
0000	883971	MOV W1, NVMKEY
0000	A8E729	BSET NVMCON, #WR
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
Step 8: Generate clock pulses for program operation to complete until the WR bit is clear.		
0000	000000	NOP
0000	803940	MOV NVMCON, W0
0000	000000	NOP
0000	887C40	MOV W0, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
—	—	Repeat until the WR bit is clear.
Step 9: Repeat Steps 3-8 until all code memory is programmed.		

3.7 Writing Configuration Bits

The procedure for writing Configuration bits is similar to the procedure for writing code memory, except that only two 24-bit words can be programmed at a time.

To change the values of the Configuration bits once they have been programmed, the device must be erased, as described in **Section 3.5 "Erasing Program Memory"**, and reprogrammed to the desired value. Code protection can be enabled by programming '0' in the code protection Configuration bits.

Table 3-7 shows the ICSP programming details for writing the Configuration bits.

In order to verify the data by reading the Configuration bits after performing the write, the code protection bits should initially be programmed to a '1' to ensure that the verification can be performed properly. After verification is finished, the code protection bits can be programmed to a '0' by using a word write to the appropriate Configuration register.

TABLE 3-7: SERIAL INSTRUCTION EXECUTION FOR WRITING CONFIGURATION WORDS

Command (Binary)	Data (Hex)	Description
Step 1: Exit the Reset vector.		
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
Step 2: Initialize the TBLPAG register for writing to the latches.		
0000	200FAC	MOV #0xFA, W12
0000	8802AC	MOV W12, TBLPAG
Step 3: Load W0:W1 with the next two Configuration Words to program.		
0000	2xxxx0	MOV #<Config1 lower word data>, W0
0000	2xxxx1	MOV #<Config1 upper word data>, W1
0000	2xxxx2	MOV #<Config2 lower word data>, W2
0000	2xxxx3	MOV #<Config2 upper word data>, W3
Step 4: Set the Write Pointer (W3) and load the write latches.		
0000	EB0300	CLR W6
0000	000000	NOP
0000	BB0B00	TBLWTL W0, [W6]
0000	000000	NOP
0000	000000	NOP
0000	BB9B01	TBLWTH W1, [W6++]
0000	000000	NOP
0000	000000	NOP
0000	BB0B02	TBLWTL W2, [W6]
0000	000000	NOP
0000	000000	NOP
0000	BB9B03	TBLWTH W3, [W6++]
0000	000000	NOP
0000	000000	NOP
Step 5: Set the NVMADRU/NVMADR register pair to point to the correct Configuration Word address.		
0000	2xxxx4	MOV #DestinationAddress<15:0>, W4
0000	2xxxx5	MOV #DestinationAddress<23:16>, W5
0000	883954	MOV W4, NVMADR
0000	883965	MOV W5, NVMADRU

TABLE 3-7: SERIAL INSTRUCTION EXECUTION FOR WRITING CONFIGURATION WORDS (CONTINUED)

Command (Binary)	Data (Hex)	Description
Step 6: Set the NVMCON register to program two instruction words.		
0000	24001A	MOV #0x4001, W10
0000	000000	NOP
0000	88394A	MOV W10, NVMCON
0000	000000	NOP
0000	000000	NOP
Step 7: Initiate the write cycle.		
0000	200551	MOV #0x55, W1
0000	883971	MOV W1, NVMKEY
0000	200AA1	MOV #0xAA, W1
0000	883971	MOV W1, NVMKEY
0000	A8E729	BSET NVMCON, #WR
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
Step 8: Generate clock pulses for program operation to complete until the WR bit is clear.		
0000	000000	NOP
0000	803940	MOV NVMCON, W0
0000	000000	NOP
0000	887C40	MOV W0, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
—	—	Repeat until the WR bit is clear.
Step 9: Repeat Steps 3-8 until all Configuration registers are programmed.		

3.8 Writing OTP Words

The procedure for writing to the OTP Words is similar to the procedure for writing code, except that the OTP Words can only be written once. It is not possible to program a '0' to a '1', but the OTP Words may be programmed from a '1' to a '0'. Refer to Figure 2-6, through Figure 2-8 for the locations of the User OTP Words.

OTP memory must be written, one double-word at a time. Before writing to any double-word in OTP memory, the corresponding memory locations must be read first. The double-word read value must be compared with the double-word value to be written and the write may be performed only if one of the following conditions is true:

1. The read value is (0xFFFFFF:0xFFFFFF).
2. The read value is identical to the value to be written (all 6 bytes must be identical).
3. If the value to be written is (0xFFFFFF:0xFFFFFF), the read value must be written back to memory. In this case, the original value to be written is ignored.

3.9 Reading OTP Words

The procedure for reading OTP Words is similar to the procedure for reading code memory. Since there are multiple OTP Words, they are read one at a time.

3.10 Reading Code Memory

Reading from code memory is performed by executing a series of TBLRD instructions and clocking out the data using the REGOUT command.

Table 3-8 shows the ICSP sequence for reading code memory.

To minimize reading time, the same packed data format that the write procedure uses is utilized. See **Section 3.6 "Writing Code Memory"** for more details on the packed data format.

TABLE 3-8: SERIAL INSTRUCTION EXECUTION FOR READING CODE MEMORY

Command (Binary)	Data (Hex)	Description
Step 1: Exit the Reset vector.		
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
Step 2: Initialize the TBLPAG register and the Read Pointer (W6) for the TBLRD instruction.		
0000	200xx0	MOV #<SourceAddress23:16>, W0
0000	8802A0	MOV W0, TBLPAG
0000	2xxxx6	MOV #<SourceAddress15:0>, W6

TABLE 3-8: SERIAL INSTRUCTION EXECUTION FOR READING CODE MEMORY (CONTINUED)

Command (Binary)	Data (Hex)	Description
Step 3: Initialize the Write Pointer (W7) and store the next four locations of code memory to W0:W5.		
0000	EB0380	CLR W7
0000	000000	NOP
0000	BA1B96	TBLRDL [W6], [W7++]
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	BADBB6	TBLRDH.B [W6++], [W7++]
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	BADBD6	TBLRDH.B [++W6], [W7++]
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	BA1BB6	TBLRDL [W6++], [W7++]
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	BA1B96	TBLRDL [W6], [W7++]
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	BADBB6	TBLRDH.B [W6++], [W7++]
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	BADBD6	TBLRDH.B [++W6], [W7++]
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	BA0BB6	TBLRDL [W6++], [W7]
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP

TABLE 3-8: SERIAL INSTRUCTION EXECUTION FOR READING CODE MEMORY (CONTINUED)

Command (Binary)	Data (Hex)	Description
Step 4: Output W0:W5 using the VISI register and the REGOUT command.		
0000	887C40	MOV W0, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
0000	887C41	MOV W1, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
0000	887C42	MOV W2, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
0000	887C43	MOV W3, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
0000	887C44	MOV W4, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
0000	887C45	MOV W5, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
Step 5: Reset the device internal PC.		
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
Step 6: Repeat Steps 3-5 until all desired code memory is read.		

3.11 Reading Configuration Registers

The procedure for reading Configuration bits is similar to the procedure for reading code memory. Since there are multiple Configuration Words, they are read one at a time.

Table 3-9 shows the ICSP programming details for reading the Configuration bits.

TABLE 3-9: SERIAL INSTRUCTION EXECUTION FOR READING CONFIGURATION WORDS

Command (Binary)	Data (Hex)	Description
Step 1: Exit the Reset vector.		
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
Step 2: Initialize TBLPAG, the Write Pointer (W7) and the Read Pointer (W6) for the TBLRD instruction.		
0000	200xx0	MOV #<Address23:16>, W0
0000	20F887	MOV #<VISI>, W7
0000	8802A0	MOV W0, TBLPAG
0000	2xxxx6	MOV #<Address15:0>, W6
Step 3: Store the Configuration register and send the contents of the VISI register.		
0000	000000	NOP
0000	BA8B96	TBLRDH [W6], [W7]
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	BA0B96	TBLRDL [W6], [W7]
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
Step 4: Repeat Steps 1-3 until all Configuration registers are read.		

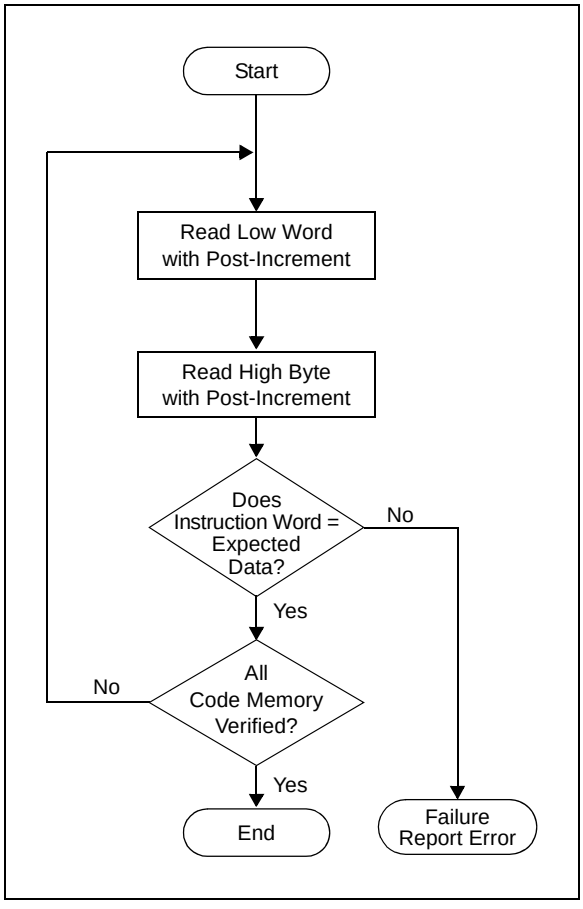
3.12 Verify Code Memory and Configuration Bits

The verify step involves reading back the code memory space and comparing it against the copy held in the programmer's buffer. The Configuration Words are verified with the rest of the code.

The verify process is illustrated in Figure 3-9. The lower word of the instruction is read, and then the lower byte of the upper word is read and compared against the instruction stored in the programmer's buffer. Refer to **Section 3.10 "Reading Code Memory"** for implementation details of reading code memory.

Note:	Because the Configuration Words include the device code protection bit, code memory should be verified immediately after writing if code protection is to be enabled. This is because the device will not be readable or verifiable if a device Reset occurs after the code-protect bit has been cleared.

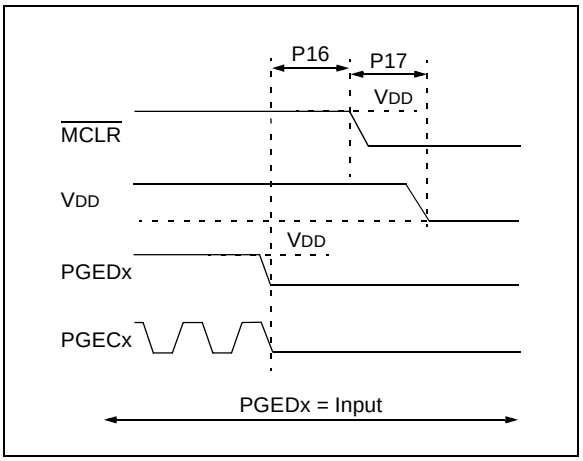
FIGURE 3-9: VERIFY CODE MEMORY FLOW



3.13 Exiting ICSP Mode

Exiting Program/Verify mode is done by removing VDD from MCLR, as illustrated in Figure 3-10. The only requirement for exit is that an interval, P16, should elapse between the last clock, and program signals on PGECx and PGEDx, before removing VDD.

FIGURE 3-10: EXITING ICSP™ MODE



4.0 DEVICE PROGRAMMING – ENHANCED ICSP

This section discusses programming the device through Enhanced ICSP and the Programming Executive (PE). The PE resides in executive memory (separate from code memory) and is executed when Enhanced ICSP Programming mode is entered. The PE provides the mechanism for the programmer (host device) to program and verify the dsPIC33EVXXXGM00X/10X devices using a simple command set and communication protocol. There are several basic functions provided by the PE:

- Read Memory
- Erase Memory
- Program Memory
- Blank Check

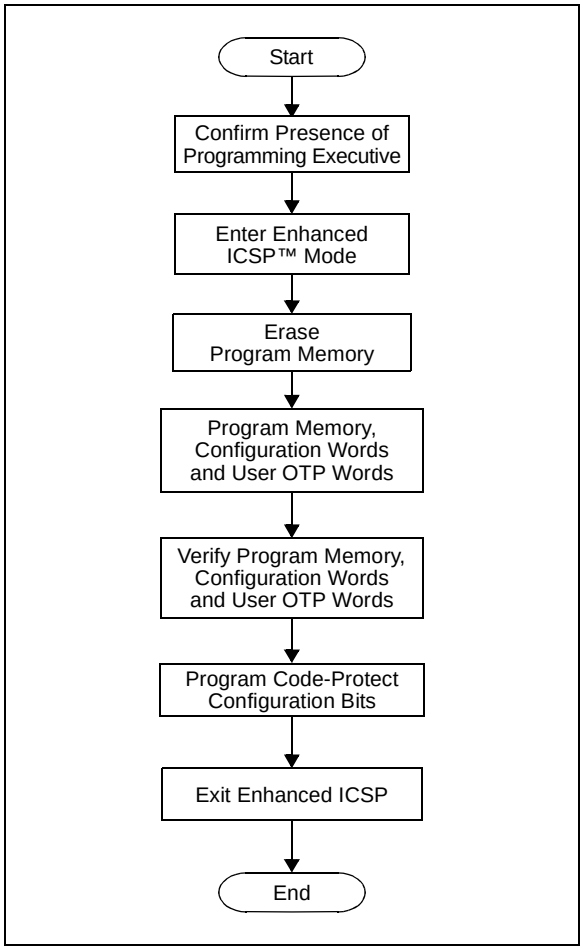
The PE performs the low-level tasks required for erasing, programming and verifying a device. This allows the programmer to program the device by issuing the appropriate commands and data. A detailed description for each command is provided in **Section 6.2 "Programming Executive Commands"**.

Note:	The PE uses the device's data RAM for variable storage and program execution. After running the PE, no assumptions should be made about the contents of data RAM.

4.1 Overview of the Programming Process

Figure 4-1 shows the high-level overview of the programming process. First, it must be determined if the PE is present in executive memory, and then, Enhanced ICSP mode is entered. The program memory is then erased, and the program memory and Configuration Words are programmed and verified. Last, the code-protect Configuration bits are programmed (if required) and Enhanced ICSP mode is exited.

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



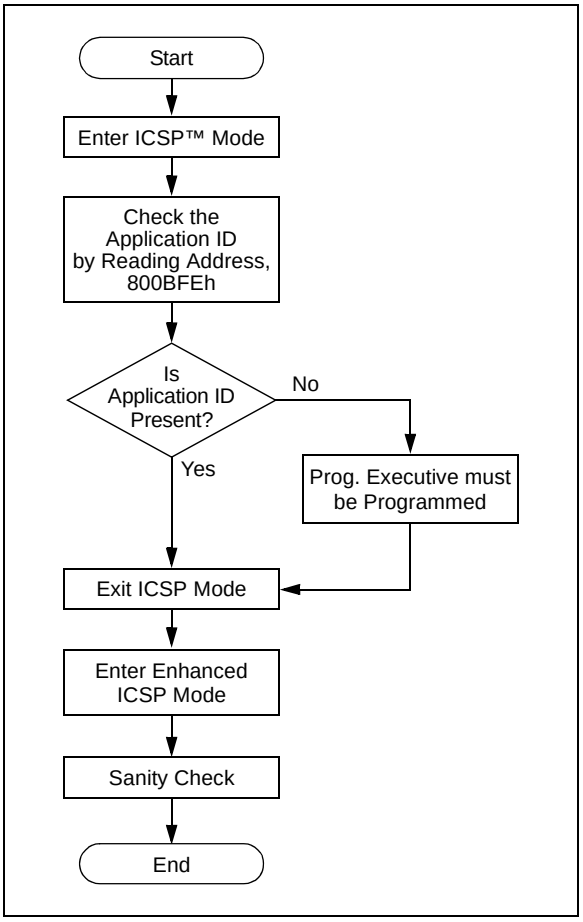
4.2 Confirming the Presence of the Programming Executive

Before programming, the programmer must confirm that the PE is stored in executive memory. The procedure for this task is illustrated in Figure 4-2.

First, the ICSP mode is entered. Then, the unique Application ID Word, stored in the executive memory, is read. If the PE is resident, the correct Application ID Word, 0xDF, is read and programming can resume as normal. However, if the Application ID Word is not present, the PE must be programmed to executive code memory using the method described in the **Section 5.0 "Programming the Programming Executive to Memory"**.

Section 3.0 "Device Programming – ICSP" describes the ICSP programming method. **Section 4.3 "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 Reading the Application ID Word

The Application ID Word is stored at address, 800BFEh, in executive code memory. To read this memory location, you must 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. The corresponding control and instruction codes that must be serially transmitted to the device to perform this operation are shown in Table 4-1.

After the programmer has clocked out the Application ID Word, it must be inspected. If the Application ID has the value, 0xDF, the PE is resident in 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 PE is not resident in memory; it must be loaded to memory before the device can be programmed. The procedure for loading the PE to memory is described in **Section 5.0 "Programming the Programming Executive to Memory"**.

TABLE 4-1: SERIAL INSTRUCTION EXECUTION FOR READING THE APPLICATION ID WORD

Command (Binary)	Data (Hex)	Description
Step 1: Exit the Reset vector.		
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
Step 2: Initialize the TBLPAG register and the Read Pointer (W0) for the TBLRD instruction.		
0000	200800	MOV #0x80, W0
0000	8802A0	MOV W0, TBLPAG
0000	20BFE0	MOV #0xBFE, W0
0000	20F881	MOV #VISI, W1
0000	000000	NOP
0000	BA0890	TBLRDL [W0], [W1]
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
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.

4.4 Entering Enhanced ICSP Mode

As illustrated in Figure 4-3, entering Enhanced ICSP Program/Verify mode requires three steps:

1. The MCLR pin is briefly driven high and then low.
2. A 32-bit key sequence is clocked into PGEDx. An interval of at least P18 must elapse before presenting the key sequence on PGEDx.
3. MCLR is held low for a specified period of time and then driven high.

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 Most Significant bit (MSb) of the most significant nibble must be shifted in first.

Once the key sequence is complete, VDD must be applied to MCLR and held at that level for as long as Program/Verify mode is to be maintained. An interval time of at least P19, P7 and P1 * 5 must elapse before presenting data on PGEDx. Signals appearing on PGEDx before P7 has elapsed will not be interpreted as valid.

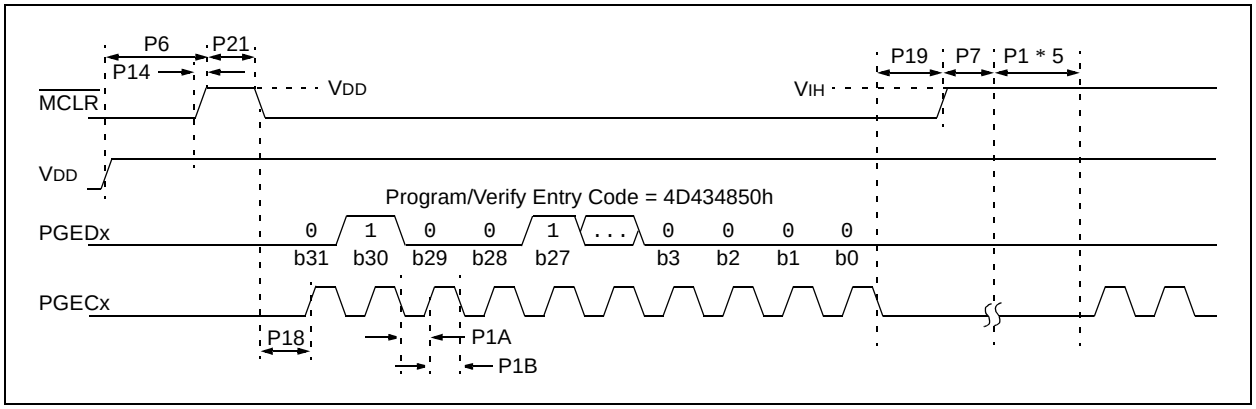
4.5 Blank Check

The term, "Blank Check", implies verifying that 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 (FF0000h:FF0002h) can be ignored by the Blank Check, since this region stores device information that cannot be erased. Additionally, all unimplemented memory space and Calibration registers 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 chip.

FIGURE 4-3: ENTERING ENHANCED ICSP™ MODE



4.6 Code Memory Programming

4.6.1 PROGRAMMING METHODOLOGY

There are two commands that can be used for programming code memory when utilizing the PE. The PROG2W command programs and verifies two 24-bit instruction words into the program memory, starting at the address specified. The second and faster command, PROGP, allows up to sixty-four 24-bit instruction words to be programmed and verified into program memory, starting at the address specified. See **Section 6.0 "The Programming Executive"** for a full description of each of these commands.

Figure 4-4 and Figure 4-5 show the programming methodology for the PROG2W and PROGP commands. In both instances, 87552K instruction words of the dsPIC33EVXXXGM00X/10X devices are programmed.

FIGURE 4-4: FLOWCHART FOR DOUBLE-WORD PROGRAMMING

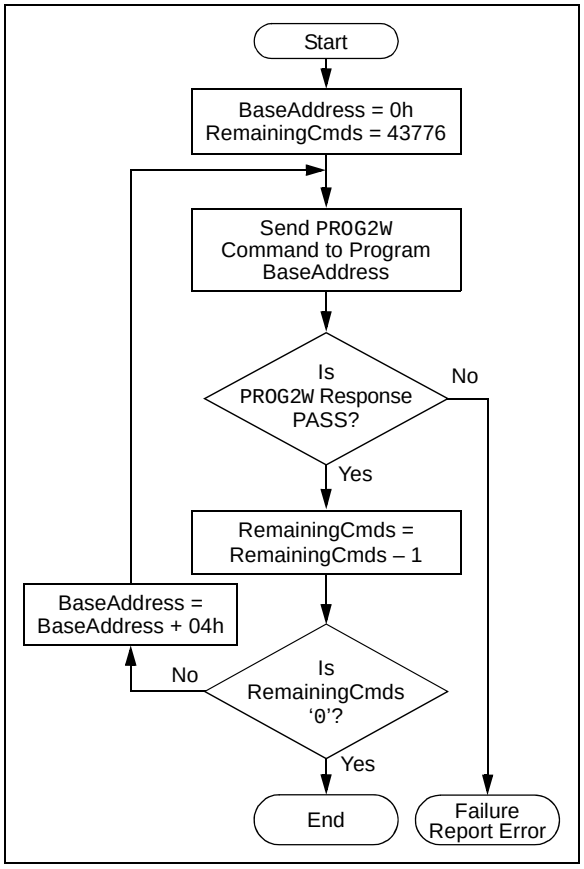
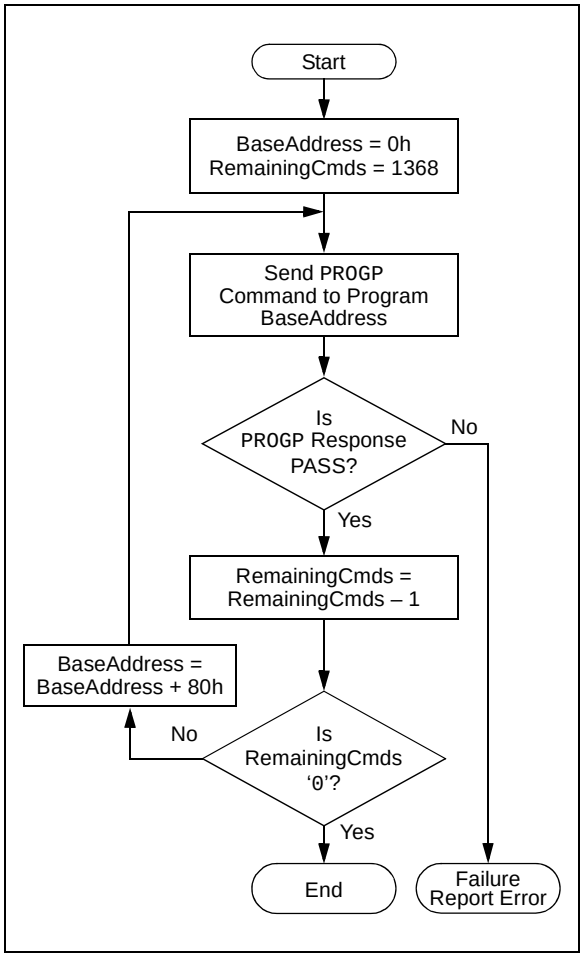


FIGURE 4-5: FLOWCHART FOR MULTIPLE WORD PROGRAMMING

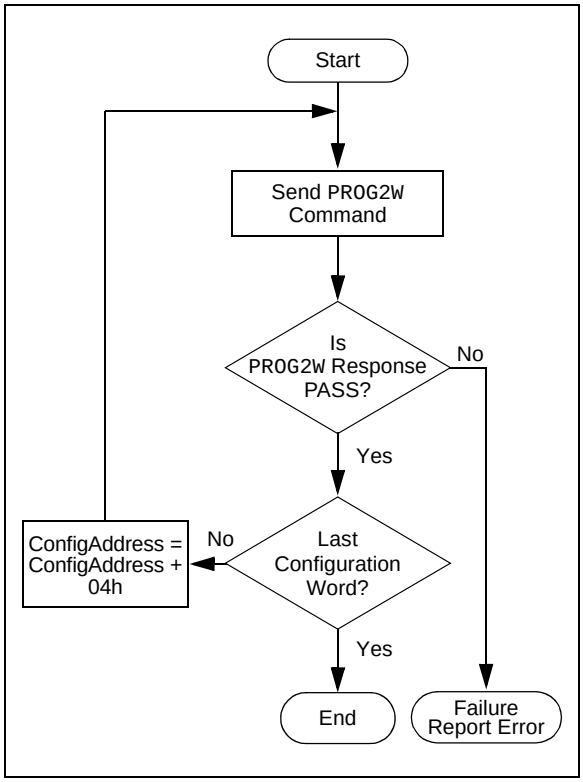


4.7 Configuration Bit Programming

Configuration bits are programmed one at a time using the PROG2W command. This command specifies the configuration data and address. When Configuration bits are programmed, any unimplemented bits must be programmed with a '1'.

Multiple PROG2W commands are required to program all Configuration bits. A flowchart for Configuration bit programming is shown in Figure 4-6.

FIGURE 4-6: CONFIGURATION BIT PROGRAMMING FLOW



4.8 Programming Verification

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

The READP command can be used to read back all the programmed code memory and Configuration Words.

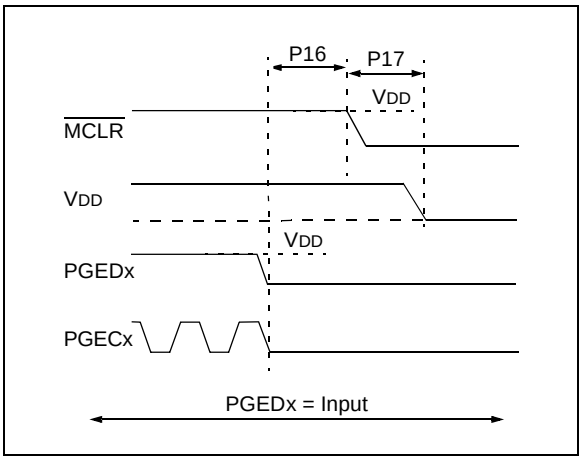
Alternatively, you can have the programmer perform the verification after the entire device is programmed using a checksum computation.

See **Section 8.0 "Checksum Computation"** for more information on calculating the checksum.

4.9 Exiting Enhanced ICSP Mode

Exiting Program/Verify mode is done by removing VDD from MCLR, as illustrated in Figure 4-7. The only requirement for exit is that an interval, P16, should elapse between the last clock, and program signals on PGECx and PGEDx, before removing VDD.

FIGURE 4-7: EXITING ENHANCED ICSP™ MODE



5.0 PROGRAMMING THE PROGRAMMING EXECUTIVE TO MEMORY

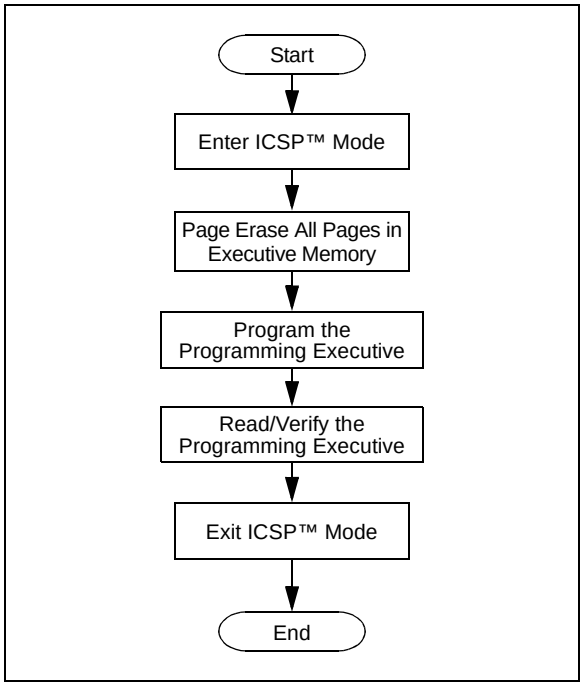
Note: The Programming Executive (PE) can be obtained from each device page on the Microchip web site: www.microchip.com.

5.1 Overview

If it is determined that the PE is not present in executive memory (as described in **Section 4.2 "Confirming the Presence of the Programming Executive"**), the PE must be programmed to executive memory.

Figure 5-1 shows the high-level process of programming the PE into executive memory. First, ICSP mode must be entered and executive memory must be erased. Then, the PE is programmed and verified. Finally, ICSP mode is exited.

FIGURE 5-1: HIGH-LEVEL PROGRAMMING EXECUTIVE PROGRAM FLOW



5.2 Erasing Executive Memory

The procedure for erasing each page of executive memory is similar to that of erasing program memory and is shown in Figure 5-2. It consists of setting NVMCON to 4003h and then executing the programming cycle.

Table 5-1 illustrates the ICSP programming process for erasing executive code memory.

Note: The PE memory must always be erased before it is programmed, as described in Figure 5-1.

FIGURE 5-2: PAGE ERASE FLOW

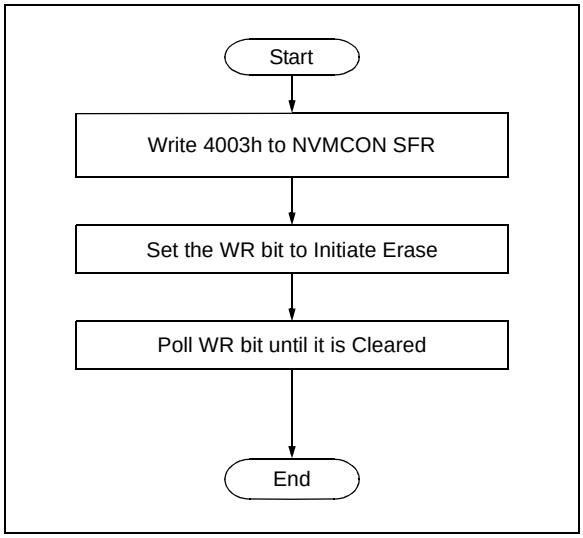


TABLE 5-1: SERIAL INSTRUCTION EXECUTION FOR ERASING ALL PAGES OF EXECUTIVE MEMORY

Command (Binary)	Data (Hex)	Description
Step 1: Exit the Reset vector.		
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
Step 2: Set the NVMADRU/NVMADR register pair to point to the correct page of executive memory to be erased.		
0000	2xxxx3	MOV #DestinationAddress<15:0>, W3
0000	2xxxx4	MOV #DestinationAddress<23:16>, W4
0000	883953	MOV W3, NVMADR
0000	883964	MOV W4, NVMADRU
Step 3: Set the NVMCON register to erase the first page of executive memory.		
0000	24003A	MOV #0x4003, W10
0000	88394A	MOV W10, NVMCON
0000	000000	NOP
0000	000000	NOP
Step 4: Initiate the erase cycle.		
0000	200551	MOV #0x55, W1
0000	883971	MOV W1, NVMKEY
0000	200AA1	MOV #0xAA, W1
0000	883971	MOV W1, NVMKEY
0000	A8E729	BSET NVMCON, #WR
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
Step 5: Generate clock pulses for the Page Erase operation to complete until the WR bit is clear.		
0000	000000	NOP
0000	803940	MOV NVMCON, W0
0000	000000	NOP
0000	887C40	MOV W0, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
—	—	Repeat until the WR bit is clear.
Step 6: Repeat Steps 2-5 for all pages of executive memory.		

5.3 Program the Programming Executive

Storing the PE to executive memory is similar to normal programming of code memory. The executive memory must first be erased and then programmed using two-word writes (two instruction words). The control flow for this method is summarized in Figure 5-3.

Table 5-2 illustrates the ICSP programming processes for PE memory. To minimize programming time, the same packed data format that the PE uses is utilized. See **Section 6.2 "Programming Executive Commands"** for more details on the packed data format.

FIGURE 5-3: PROGRAMMING EXECUTIVE PROGRAM FLOW

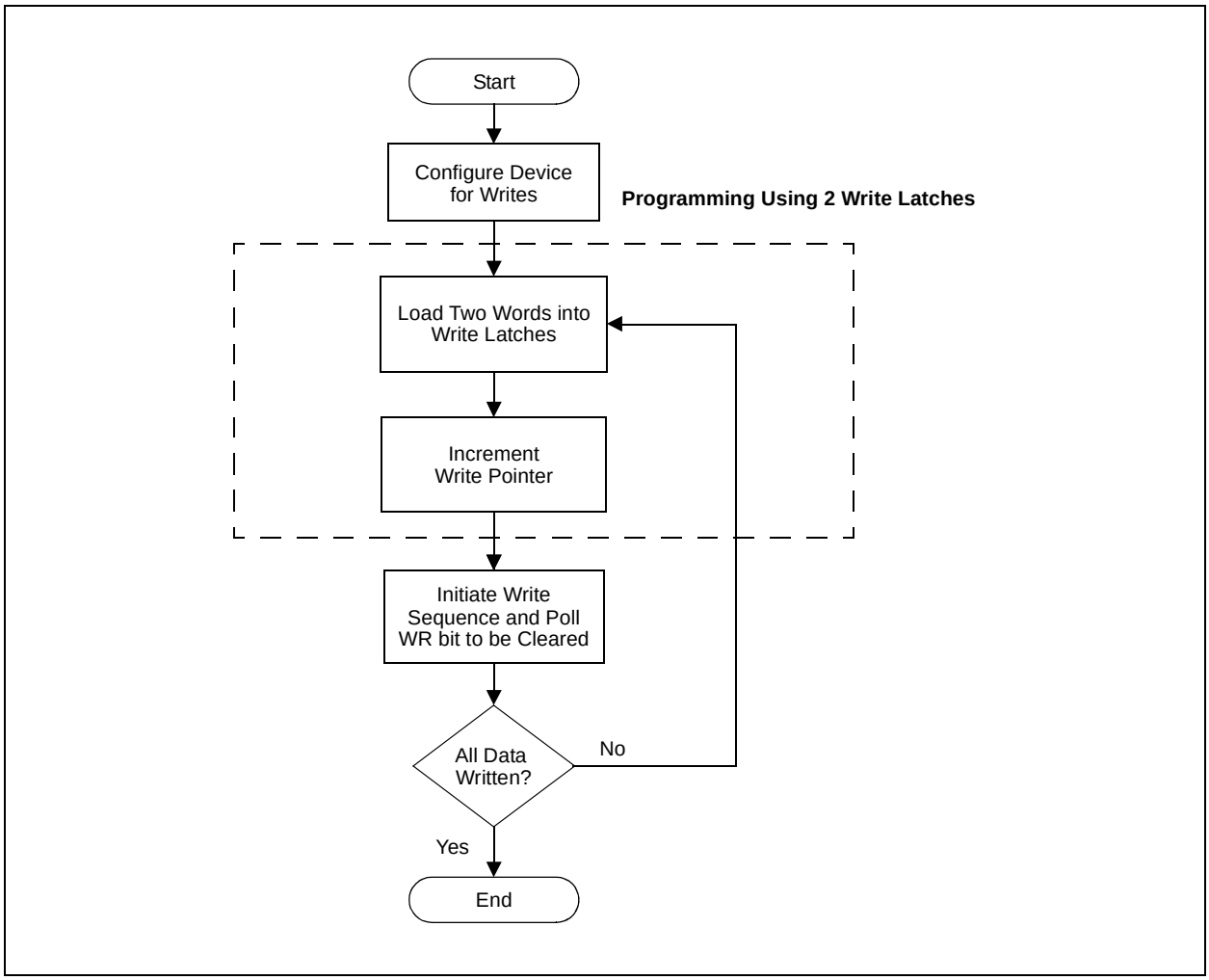


TABLE 5-2: PROGRAMMING THE PROGRAMMING EXECUTIVE (TWO-WORD LATCH WRITES)

Command (Binary)	Data (Hex)	Description
Step 1: Exit the Reset vector.		
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
Step 2: Initialize the TBLPAG register for writing to the latches.		
0000	200FAC	MOV #0xFA, W12
0000	8802AC	MOV W12, TBLPAG
Step 3: Load W0:W2 with the next two packed instruction words to program.		
0000	2xxxx0	MOV #<LSW0>, W0
0000	2xxxx1	MOV #<MSB1:MSB0>, W1
0000	2xxxx2	MOV #<LSW1>, W2
Step 4: Set the Read Pointer (W6) and the Write Pointer (W7), and load the write latches.		
0000	EB0300	CLR W6
0000	000000	NOP
0000	EB0380	CLR W7
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	BB0B96	TBLWTL.W [W6], [W7]
0000	000000	NOP
0000	000000	NOP
Step 5: Set the NVMADRU/NVMADR register pair to point to the correct row.		
0000	2xxxx3	MOV #DestinationAddress<15:0>, W3
0000	2xxxx4	MOV #DestinationAddress<23:16>, W4
0000	883953	MOV W3, NVMADR
0000	883964	MOV W4, NVMADRU
Step 6: Set the NVMCON register to program two instruction words.		
0000	24001A	MOV #0x4001, W10
0000	000000	NOP
0000	88394A	MOV W10, NVMCON
0000	000000	NOP
0000	000000	NOP

TABLE 5-2: PROGRAMMING THE PROGRAMMING EXECUTIVE (TWO-WORD LATCH WRITES) (CONTINUED)

Command (Binary)	Data (Hex)	Description
Step 7: Initiate the write cycle.		
0000	200551	MOV #0x55, W1
0000	883971	MOV W1, NVMKEY
0000	200AA1	MOV #0xAA, W1
0000	883971	MOV W1, NVMKEY
0000	A8E729	BSET NVMCON, #WR
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
Step 8: Generate clock pulses for program operation to complete until the WR bit is clear.		
0000	000000	NOP
0000	803940	MOV NVMCON, W0
0000	000000	NOP
0000	887C40	MOV W0, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
—	—	Repeat until the WR bit is clear.
Step 9: Repeat Steps 3-8 until all code memory is programmed.		

5.4 Reading Executive Memory

Reading from executive memory is performed by executing a series of TBLRD instructions and clocking out the data using the REGOUT command.

Table 5-3 shows the ICSP programming details for reading executive memory.

To minimize reading time, the same packed data format that the PE uses is utilized. See **Section 6.2 "Programming Executive Commands"** for more details on the packed data format.

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

Command (Binary)	Data (Hex)	Description
Step 1: Exit the Reset vector.		
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
Step 2: Initialize the TBLPAG register and the Read Pointer (W6) for the TBLRD instruction.		
0000	200xx0	MOV #<SourceAddress23:16>, W0
0000	8802A0	MOV W0, TBLPAG
0000	2xxxx6	MOV #<SourceAddress15:0>, W6

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

Command (Binary)	Data (Hex)	Description
Step 3: Initialize the Write Pointer (W7) and store the next four locations of code memory to W0:W5.		
0000	EB0380	CLR W7
0000	000000	NOP
0000	BA1B96	TBLRDL [W6], [W7++]
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	BADBB6	TBLRDH.B [W6++], [W7++]
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	BADBD6	TBLRDH.B [++W6], [W7++]
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	BA1BB6	TBLRDL [W6++], [W7++]
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	BA1B96	TBLRDL [W6], [W7++]
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	BADBB6	TBLRDH.B [W6++], [W7++]
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	BADBD6	TBLRDH.B [++W6], [W7++]
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	BA0BB6	TBLRDL [W6++], [W7]
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP

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

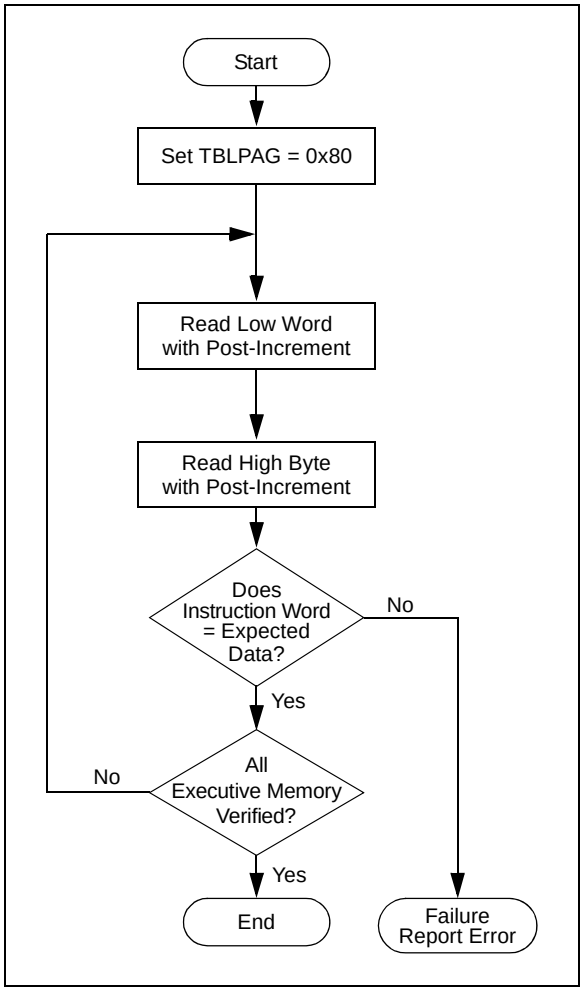
Command (Binary)	Data (Hex)	Description
Step 4: Output W0:W5 using the VISI register and REGOUT command.		
0000	887C40	MOV W0, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
0000	887C41	MOV W1, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
0000	887C42	MOV W2, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
0000	887C43	MOV W3, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
0000	887C44	MOV W4, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
0000	887C45	MOV W5, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register.
0000	000000	NOP
Step 5: Reset the device internal PC.		
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
Step 6: Repeat Steps 3-5 until all desired code memory is read.		

5.5 Verify Programming Executive

The verify step involves reading back the executive memory space and comparing it against the copy held in the programmer's buffer.

The verify process is illustrated in Figure 5-4. The lower word of the instruction is read, and then the lower byte of the upper word is read and compared against the instruction stored in the programmer's buffer. Refer to **Section 5.4 "Reading Executive Memory"** for implementation details of reading executive memory.

FIGURE 5-4: VERIFY PROGRAMMING EXECUTIVE MEMORY FLOW



6.0 THE PROGRAMMING EXECUTIVE

Note: The Programming Executive (PE) can be obtained from each device page on the Microchip web site: www.microchip.com.

6.1 Programming Executive Communication

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

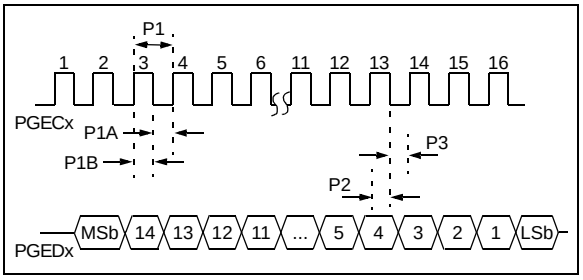
All communication is initiated by the programmer in the form of a command. Only one command at a time can be sent to the PE. In turn, the PE only sends one response to the programmer after receiving and processing a command. The PE command set is described in **Section 6.2 "Programming Executive Commands"**. The response set is described in **Section 6.3 "Programming Executive Responses"**.

6.1.1 COMMUNICATION INTERFACE AND PROTOCOL

The ICSP/Enhanced ICSP interface is a 2-wire SPI, implemented using the PGC and PGD pins. The PGC pin is used as a clock input pin and the clock source must be provided by the programmer. The PGD pin is used for sending command data to, and receiving response data from, the PE.

Note: For Enhanced ICSP, all serial data is transmitted on the falling edge of PGC and latched on the rising edge of PGC. All data transmissions are sent to the MSb first, using 16-bit mode (see Figure 6-1).

FIGURE 6-1: PROGRAMMING EXECUTIVE SERIAL TIMING



Since a 2-wire SPI is used, and data transmissions are bidirectional, a simple protocol is used to control the direction of PGD. When the programmer completes a command transmission, it releases the PGD line and allows the PE to drive this line high. The PE keeps the PGEDx line high to indicate that it is processing the command.

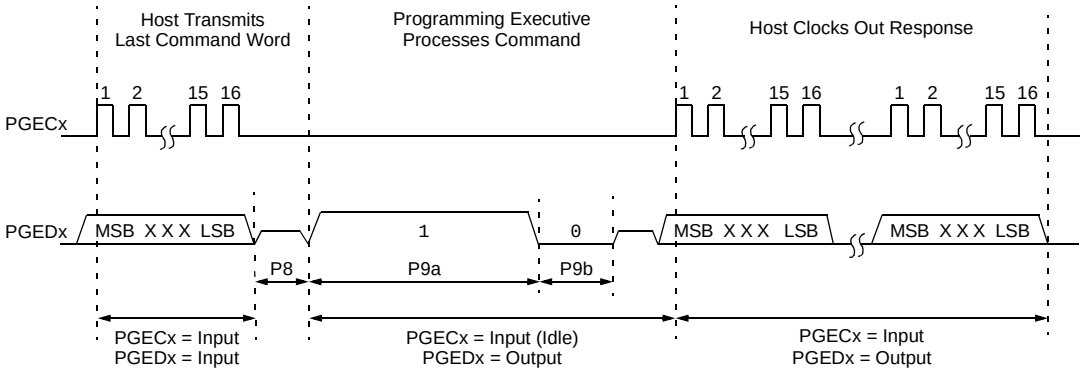
After the PE has processed the command, it brings PGD low (P9b) to indicate to the programmer that the response is available to be clocked out. The programmer can begin to clock out the response after a maximum wait (P9b) and the programmer must provide the necessary amount of clock pulses to receive the entire response from the PE.

After the entire response is clocked out, the programmer should terminate the clock on PGC until it is time to send another command to the PE. This protocol is illustrated in Figure 6-2.

6.1.2 SPI RATE

In Enhanced ICSP mode, the dsPIC33EVXXXGM00X/10X devices operate from the Fast Internal RC Oscillator (FRC), which has a nominal frequency of 7.3728 MHz. This oscillator frequency yields an effective system clock frequency of 3.6864 MHz. To ensure that the programmer does not clock too fast, it is recommended that a 1.8432 MHz clock be provided by the programmer.

FIGURE 6-2: PROGRAMMING EXECUTIVE – PROGRAMMER COMMUNICATION PROTOCOL



Note 1: A delay of 25 ms is required between commands.

6.1.3 TIME-OUTS

The PE uses no Watchdog Timer or time-out for transmitting responses to the programmer. If the programmer does not follow the flow control mechanism using PGECx, as described in **Section 6.1.1 "Communication Interface and Protocol"**, it is possible that the PE will behave unexpectedly while trying to send a response to the programmer. Since the PE has no 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 in Table 6-1. If the command time-out expires, the programmer should reset the PE and start programming the device again.

TABLE 6-1: PROGRAMMING EXECUTIVE COMMAND SET

Opcode	Mnemonic	Length (16-bit words)	Time-out	Description
0x0	SCHECK	1	1 ms	Sanity check.
0x1	Reserved	N/A	N/A	This command is reserved; it will return a NACK.
0x2	READP	4	1 ms/row	Read 'N' 24-bit instruction words of primary Flash memory, starting from the specified address.
0x3	PROG2W	6	5 ms	Program a double instruction word of code memory at the specified address and verify.
0x4	Reserved	N/A	N/A	This command is reserved; it will return a NACK.
0x5	PROGP	99	5 ms	Program 64 words of program memory at the specified starting address, then verify.
0x6	Reserved	N/A	N/A	This command is reserved; it will return a NACK.
0x7	ERASEB	1	125 ms	Bulk Erase user memory.
0x8	Reserved	N/A	N/A	This command is reserved; it will return a NACK.
0x9	ERASEP	3	25 ms	Command to erase a page.
0xA	Reserved	N/A	N/A	This command is reserved; it will return a NACK.
0xB	QVER	1	1 ms	Query the PE software version.
0xC	CRCP	5	1s	Perform a CRC-16 on the specified range of memory.
0xD	Reserved	N/A	N/A	This command is reserved; it will return a NACK.
0xE	QBLANK	5	700 ms	Query to check whether the code memory is blank.

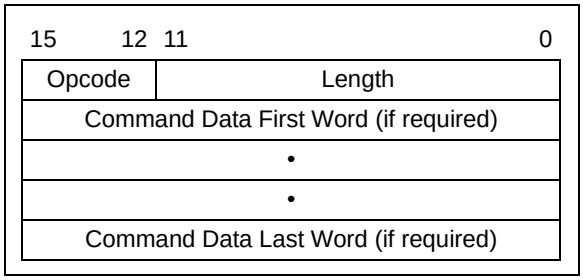
6.2 Programming Executive Commands

The PE command set is shown in Table 6-1. This table contains the opcode, mnemonic, length, time-out and description for each command. Functional details on each command are provided in the command descriptions (see **Section 6.2.4 "Command Descriptions"**).

6.2.1 COMMAND FORMAT

All PE commands have a general format, consisting of a 16-bit header and any required data for the command (see Figure 6-3). 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 6-3: COMMAND FORMAT



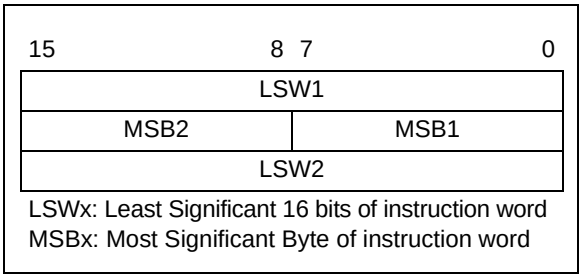
The command opcode must match one of those in the command set. Any command that is received, which does not match the list in Table 6-1, will return a "NACK" response (see **Section 6.3.1.1 "Opcode Field"**).

The command length is represented in 16-bit words since the SPI operates in 16-bit mode. The PE 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 PE.

6.2.2 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 6-4. This format minimizes traffic over the SPI and provides the PE with data that is properly aligned for performing Table Write operations.

FIGURE 6-4: PACKED INSTRUCTION WORD FORMAT



LSWx: Least Significant 16 bits of instruction word
MSBx: Most Significant Byte of instruction word

Note:	When the number of instruction words transferred is odd, MSB2 is zero and LSW2 cannot be transmitted.

6.2.3 PROGRAMMING EXECUTIVE ERROR HANDLING

The PE will "NACK" all unsupported commands. Additionally, due to the memory constraints of the PE, no checking is performed on the data contained in the programmer command. It is the responsibility of the programmer to command the PE with valid command arguments or the programming operation may fail. Additional information on error handling is provided in **Section 6.3.1.3 "QE_Code Field"**.

6.2.4 COMMAND DESCRIPTIONS

All commands supported by the PE are described in **Section 6.2.4.1 "SCHECK Command"** through **Section 6.2.4.9 "QBLANK Command"**.

6.2.4.1 SCHECK Command

15 12	11 0
Opcode	Length

Table 6-2 shows the description for the SCHECK command.

TABLE 6-2: COMMAND DESCRIPTION

Field	Description
Opcode	0x0
Length	0x1

The SCHECK command instructs the PE to do nothing but generate a response. This command is used as a "Sanity Check" to verify that the PE is operational.

Expected Response (2 words):

0x1000

0x0002

Note:	This instruction is not required for programming, but is provided for development purposes only.

6.2.4.2 READP Command

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

Table 6-3 shows the description for READP command.

TABLE 6-3: COMMAND DESCRIPTION

Field	Description
Opcode	0x2
Length	0x4
N	Number of 24-bit instructions to read (maximum of 32768)
Reserved	0x0
Addr_MSB	MSB of 24-bit source address
Addr_LS	Least Significant 16 bits of 24-bit source address

The READP command instructs the PE to read N 24-bit words of code memory, Flash Configuration Words or Device ID registers, starting from the 24-bit address specified by Addr_MSB and Addr_LS. This command can only be used to read 24-bit data. All data returned in the response to this command uses the packed data format described in **Section 6.2.2 "Packed Data Format"**.

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

0x1200

2 + 3 * N/2

Least Significant Program Memory Word 1

...

Least Significant Data Word N

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

0x1200

4 + 3 * (N – 1)/2

Least Significant Program Memory Word 1

...

MSB of Program Memory Word N (zero-padded)

Note 1:	Reading unimplemented memory will cause the PE to reset. To prevent this from occurring, ensure that only memory locations present on a particular device are accessed.
2:	When this command is used to read Device ID registers, the upper byte (bits<23:16>) of each word returned by the PE must be ignored.

6.2.4.3 PROG2W Command

15 12	11 8	7 0
Opcode	Length	
Reserved	Addr_MSB	
Addr_LS		
DataL_LS		
DataH_MSB	DataL_MSB	
DataH_LS		

Table 6-4 shows the description for the PROG2W command.

TABLE 6-4: COMMAND DESCRIPTION

Field	Description
Opcode	0x3
Length	0x6
DataL_MSB	MSB of 24-bit data for low instruction word
DataH_MSB	MSB of 24-bit data for high instruction word
Addr_MSB	MSB of 24-bit destination address
Addr_LS	Least Significant 16 bits of 24-bit destination address
DataL_LS	Least Significant 16 bits of 24-bit data for low instruction word
DataH_LS	Least Significant 16 bits of 24-bit data for high instruction word

The PROG2W command instructs the PE to program two instruction words of code memory (6 bytes) to the specified memory address.

After the words have been programmed to code memory, the PE verifies the programmed data against the data in the command.

Expected Response (2 words):

0x1300
0x0002

6.2.4.4 PROGP Command

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

Table 6-5 shows the description for the PROGP command.

TABLE 6-5: COMMAND DESCRIPTION

Field	Description
Opcode	0x5
Length	0x63
Reserved	0x0
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 95
D_96	16-bit Data Word 96

The PROGP command instructs the PE to program one row of code memory (64 instruction words) to the specified memory address. Programming begins with the row address specified in the command. The destination address should be a multiple of 0x80.

The data to program the memory, located in command words, D_1 through D_96, must be arranged using the packed instruction word format illustrated in Figure 6-4.

After all data has been programmed to code memory, the PE verifies the programmed data against the data in the command.

Expected Response (2 words):

0x1500
0x0002

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

6.2.4.5 ERASEB Command

15 12	11 8	7 0
Opcode		Length

Table 6-6 shows the description for the ERASEB command.

TABLE 6-6: COMMAND DESCRIPTION

Field	Description
Opcode	0x7
Length	0x1

The ERASEB command instructs the PE to perform a Bulk Erase of the user Flash memory.

Expected Response (2 words):

0x1700

0x0002

6.2.4.6 ERASEP Command

15 12	11 8	7 0
Opcode	Length	
NUM_PAGES	Addr_MSB	
Addr_LS		

Table 6-5 shows the description for the ERASEP command.

TABLE 6-7: COMMAND DESCRIPTION

Field	Description
Opcode	0x9
Length	0x3
NUM_PAGES	Up to 255
Addr_MSB	Most Significant Byte of the 24-bit address
Addr_LS	Least Significant 16 bits of the 24-bit address

The ERASEP command instructs the PE to Page Erase [NUM_PAGES] of code memory. The code memory must be erased at an “even” 512 instruction words address boundary.

Expected Response (2 words):

0x1900

0x0002

6.2.4.7 QVER Command

15 12	11 0
Opcode	Length

Table 6-8 shows the description for the QVER command.

TABLE 6-8: COMMAND DESCRIPTION

Field	Description
Opcode	0xB
Length	0x1

The QVER command queries the version of the PE software stored in test memory. The “version.revision” information is returned in the response’s QE_Code, using a single byte with the following format: main version in upper nibble and revision in the lower nibble (i.e., 0x23 means Version 2.3 of PE software).

Expected Response (2 words):

0x1BMN (where “MN” stands for version M.N)

0x0002

6.2.4.8 CRCP Command

15 12	11 8	7 0
Opcode	Length	
Reserved	Addr_MSB	
Addr_LSW		
Reserved	Size_MSB	
Size_LSW		

Table 6-10 shows the description for the CRCP command.

TABLE 6-9: COMMAND DESCRIPTION

Field	Description
Opcode	Ch
Length	5h
Addr_MSB	Most Significant Byte of 24-bit address
Addr_LSW	Least Significant 16 bits of 24-bit address
Size	Number of 24-bit locations (address range divided by 2)

The CRCP command performs a CRC-16 on the range of memory specified. This command can substitute for a full chip verify. Data is shifted in a packed method, as demonstrated in Figure 6-4, byte-wise, Least Significant Byte (LSB) first.

Example:

CRC-CCITT-16 with test data of “123456789” becomes 29B1h

Expected Response (3 words):

QE_Code: 0x1C00

Length: 0x0003

CRC Value: 0xXXXX

6.2.4.9 QBLANK Command

15 12	11 0
Opcode	Length
Reserved	Size_MSB
Size_LSW	
Reserved	Addr_MSB
Addr_LSW	

Table 6-10 shows the description for the QBLANK command.

TABLE 6-10: COMMAND DESCRIPTION

Field	Description
Opcode	0xE
Length	0x5
Size	Length of program memory to check (in 24-bit words) + Addr_MS
Addr_MSB	Most Significant Byte of the 24-bit address
Addr_LSW	Least Significant 16 bits of the 24-bit address

The QBLANK command queries the PE to determine if the contents of code memory are blank (contains all '1's). The size of code memory to check must be specified in the command.

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

QBLANK returns a QE_Code of F0h if the specified code memory is blank; otherwise, QBLANK returns a QE_Code of 0Fh.

Expected Response (2 words for blank device):

0x1EF0

0x0002

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

0x1E0F

0x0002

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

6.3 Programming Executive Responses

The PE 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 PE response set is shown in Table 6-11. This table contains the opcode, mnemonic and description for each response. The response format is described in **Section 6.3.1 "Response Format"**.

TABLE 6-11: PROGRAMMING EXECUTIVE RESPONSE OPCODES

Opcode	Mnemonic	Description
0x1	PASS	Command successfully processed
0x2	FAIL	Command unsuccessfully processed
0x3	NACK	Command not known

6.3.1 RESPONSE FORMAT

All PE 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)		

Table 6-12 shows the description of the response format.

TABLE 6-12: RESPONSE FORMAT DESCRIPTION

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)

6.3.1.1 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 6-11). 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 PE is not identified, the PE returns a NACK response.

6.3.1.2 Last_Cmd Field

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

6.3.1.3 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 other commands.

When the PE 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. The format of the QE_Code for both queries is shown in Table 6-13.

TABLE 6-13: QE_Code FOR QUERIES

Query	QE_Code
QBLANK	0x0F = Code memory is NOT blank 0xF0 = Code memory is blank
QVER	0xMN, where PE Software Version = M.N (i.e., 0x32 means Software Version 3.2)

When the PE processes any command other than a query, the QE_Code represents an error code. Supported error codes are shown in Table 6-14. If a command is successfully processed, the returned QE_Code is set to 0x0, which indicates that there is no error in the command processing. If the verify of the programming for the PROGW command fails, the QE_Code is set to 0x1. For all other PE errors, the QE_Code is 0x02.

TABLE 6-14: QE_Code FOR NON-QUERY COMMANDS

QE_Code	Description
0x0	No error
0x1	Verify failed
0x2	Other error

6.3.1.4 Response Length

The response length indicates the length of the PE's response in 16-bit words. This field includes the 2 words of the response header.

With the exception of the response for the read commands, the length of each response is only 2 words.

The response to the READP commands uses the packed instruction word format, described in **Section 6.2.2 "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.

7.0 DEVICE ID

The Device ID region of memory can be used to determine variant and manufacturing information about the chip. This region of memory is read-only and can be read when code protection is enabled.

Table 7-1 lists the identification information for each device. Table 7-2 shows the Device ID registers.

TABLE 7-1: DEVICE IDs

Device	DEVID	DEVREV	Silicon Revision
dsPIC33EV64GM002	0x5D11	0x4004	A4
dsPIC33EV64GM004	0x5D10		
dsPIC33EV64GM006	0x5D13		
dsPIC33EV64GM102	0x5D19		
dsPIC33EV64GM104	0x5D18		
dsPIC33EV64GM106	0x5D1B		
dsPIC33EV128GM002	0x5D21		
dsPIC33EV128GM004	0x5D20		
dsPIC33EV128GM006	0x5D23		
dsPIC33EV128GM102	0x5D29		
dsPIC33EV128GM104	0x5D28		
dsPIC33EV128GM106	0x5D2B		
dsPIC33EV256GM002	0x5D31		
dsPIC33EV256GM004	0x5D30		
dsPIC33EV256GM006	0x5D33		
dsPIC33EV256GM102	0x5D39		
dsPIC33EV256GM104	0x5D38		
dsPIC33EV256GM106	0x5D3B		

TABLE 7-2: DEVICE ID REGISTERS

Address	Name	Bit															
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FF0000h	DEVID	DEVID Value															
FF0002h	DEVREV	DEVREV Value															

8.0 CHECKSUM COMPUTATION

Checksums for devices are 16 bits in size. The checksum is calculated by summing the following:

- Contents of code memory locations
- Contents of Configuration Words

All memory locations, including Configuration Words, are summed by adding all three bytes of each memory address. For certain Configuration Words, a read mask is used to ignore the bits which are reserved.

Table 8-1 is an example of the checksum calculation for the dsPIC33EV256GMX06 devices.

TABLE 8-1: CHECKSUM COMPUTATION EXAMPLE

Device	Read Code Protection	Checksum Computation	Erased Value	Value with 0xAAAAAA at 0x0 and Last Code Address
dsPIC33EV256GMX06	Disabled	CFGB + SUM (0x02AB7E) Reads of program memory return	0x4CCE(1,2)	0x4AD0(1,2)
	Enabled	0x00	0x0000	0x0000

Item Description:

SUM(a:b) = Byte sum of locations, a to b inclusive (all 3 bytes of code memory)

CFGB = Configuration Block (masked) = Byte sum of ((FSEC & 0x008FEF) + (FBSLIM & 0x001FFF) + (Reserved & 0x008000) + (FOSCSEL & 0x000087) + (FOSC & 0x0001E7) + (FWDT & 0x0003FF) + (FPOR & 0x000001) + (FICD & 0x000083) + (FDMTINTVL & 0x00FFFF) + (FDMTINTVH & 0x00FFFF) + (FDMTCNTL & 0x00FFFF) + (FDMTCNTH & 0x00FFFF) + (FDMT & 0x000001) + (FDEVOPT & 0x00000D) + (FALTREG & 0x000077))

Note 1: For the checksum computation example, the Configuration bits are set to the recommended default value, which is 0xFFFFFF for most Configuration registers.

2: For the Reserved register, a default value of 0xFF7FFF is used.

9.0 AC/DC CHARACTERISTICS AND TIMING REQUIREMENTS

Table 9-1 lists the AC/DC characteristics and timing requirements.

TABLE 9-1: AC/DC CHARACTERISTICS AND TIMING REQUIREMENTS

Standard Operating Conditions Operating Temperature: -40°C to +85°C. Programming at +25°C is recommended.						
Param. No.	**Symbol**	**Characteristic**	**Min.**	**Max.**	**Units**	**Conditions**
D111	VDD	Supply Voltage During Programming	4.7	5.5	V	See **Note 1**
P1	TPGC	Serial Clock (PGECx) Period (ICSP™)	200	—	ns	
P1	TPGC	Serial Clock (PGECx) Period (Enhanced ICSP)	500	—	ns	
P1A	TPGCL	Serial Clock (PGECx) Low Time (ICSP)	80	—	ns	
P1A	TPGCL	Serial Clock (PGECx) Low Time (Enhanced ICSP)	200	—	ns	
P1B	TPGCH	Serial Clock (PGECx) High Time (ICSP)	80	—	ns	
P1B	TPGCH	Serial Clock (PGECx) High Time (Enhanced ICSP)	200	—	ns	
P2	TSET1	Input Data Setup Time to Serial Clock ↓	15	—	ns	
P3	THLD1	Input Data Hold Time from PGECx ↓	15	—	ns	
P4	TDLY1	Delay Between 4-Bit Command and Command Operand	40	—	ns	
P4A	TDLY1A	Delay Between Command Operand and Next 4-Bit Command	40	—	ns	
P5	TDLY2	Delay Between Last PGECx ↓ of Command to First PGECx ↑ of Read of Data Word	20	—	ns	
P6	TSET2	VDD ↑ Setup Time to $\overline{\text{MCLR}}$ ↑	100	—	ns	
P7	THLD2	Input Data Hold Time from $\overline{\text{MCLR}}$ ↑	50	—	ms	
P8	TDLY3	Delay Between Last PGECx ↓ of Command Byte to PGEDx ↑ by PE	12	—	µs	
P9A	TDLY4	PE Command Processing Time	10	—	µs	
P9B	TDLY5	Delay Between PGEDx ↓ by PE to PGEDx Released by PE	15	23	µs	
P10	TDLY6	PGECx Low Time After Programming	400	—	ns	
P11	TDLY7	Bulk Erase Time	16	24	ms	
P12	TDLY8	Page Erase Time	16	24	ms	See **Note 2**
P13	TDLY9	Double-Word Programming Time	—	—	µs	See **Note 2** and **Note 3**
P14	TR	$\overline{\text{MCLR}}$ Rise Time to Enter ICSP mode	—	1.0	µs	
P15	TVALID	Data Out Valid from PGECx ↑	10	—	ns	
P16	TDLY10	Delay Between Last PGECx ↓ and $\overline{\text{MCLR}}$ ↓	0	—	s	
P17	THLD3	$\overline{\text{MCLR}}$ ↓ to VDD ↓	100	—	ns	
P18	TKEY1	Delay from First $\overline{\text{MCLR}}$ ↓ to First PGECx ↑ for Key Sequence on PGEDx	1	—	ms	

Note 1: VDD must also be supplied to the AVDD pins during programming. AVDD and AVSS should always be within ±0.3V of VDD and VSS, respectively.

2: Time depends on the FRC accuracy and the value of the FRC Oscillator Tuning register. Refer to the **"Electrical Characteristics"** chapter in the specific device data sheet.

3: This time applies to Program Memory Words, Configuration Words and User ID Words.

TABLE 9-1: AC/DC CHARACTERISTICS AND TIMING REQUIREMENTS (CONTINUED)

Standard Operating Conditions Operating Temperature: -40°C to +85°C. Programming at +25°C is recommended.						
Param. No.	**Symbol**	**Characteristic**	**Min.**	**Max.**	**Units**	**Conditions**
P19	TKEY2	Delay from Last PGECx ↓ for Key Sequence on PGEDx to Second $\overline{\text{MCLR}}$ ↑	25	—	ns	
P21	TMCLRH	$\overline{\text{MCLR}}$ High Time	—	500	µs	

Note 1: VDD must also be supplied to the AVDD pins during programming. AVDD and AVSS should always be within ±0.3V of VDD and VSS, respectively.

2: Time depends on the FRC accuracy and the value of the FRC Oscillator Tuning register. Refer to the **"Electrical Characteristics"** chapter in the specific device data sheet.

3: This time applies to Program Memory Words, Configuration Words and User ID Words.

APPENDIX A: REVISION HISTORY

Revision A (September 2013)

Original version of this programming specification created for dsPIC33EVXXXGM00X/10X device families.

Revision B (February 2014)

This revision includes the following updates:

- Sections
  - Updated **Section 2.5 “Memory Map”**, **Section 2.6.1 “Overview”**, **Section 2.6.3 “OTP (One-Time-Programmable) Memory”**, **Section 3.1 “Overview of the Programming Process”**, **Section 3.5 “Erasing Program Memory”**, **Section 3.7 “Writing Configuration Bits”**, **Section 3.8 “Writing OTP Words”**, **Section 3.11 “Reading Configuration Registers”**, **Section 3.9 “Reading OTP Words”**, **Section 3.12 “Verify Code Memory and Configuration Bits”**, **Section 4.2 “Confirming the Presence of the Programming Executive”**, **Section 4.3 “Reading the Application ID Word”**, **Section 4.5 “Blank Check”**, **Section 5.2 “Erasing Executive Memory”**, **Section 6.0 “The Programming Executive”**, **Section 6.2.4.1 “SCHECK Command”**, **Section 6.2.4.4 “PROGP Command”**, **Section 8.0 “Checksum Computation”**
- Tables
  - Updated Table 2-2, Table 2-3, Table 2-4, Table 2-6, Table 3-2, Table , Table 3-6, Table 3-7, Table 3-9, Table 4-1, Table 5-1, Table 5-2, Table 5-3, Table 6-1, Table 6-6, Table 8-1.
- Figures
  - Updated Figure 2-6, Figure 2-7, Figure 2-8, Figure 3-1, Figure 3-8, Figure 3-9, Figure 4-1, Figure 4-2, Figure 4-4, Figure 4-5, Figure 4-6, Figure 5-1, Figure 5-2, Figure 5-3
- Register
  - Updated Register 3-1
- Minor updates to text and formatting were incorporated throughout the document

Revision C (June 2014)

This revision includes the following updates:

- Sections
  - Updated **Section 3.7 “Writing Configuration Bits”**, **Section 3.11 “Reading Configuration Registers”**, **Section 3.9 “Reading OTP Words”**, **Section 3.12 “Verify Code Memory and Configuration Bits”**, **Section 4.2 “Confirming the Presence of the Programming Executive”**, **Section 4.3 “Reading the Application ID Word”**, **Section 6.0 “The Programming Executive”**, **Section 6.2.4.4 “PROGP Command”**, **Section 8.0 “Checksum Computation”**
- Tables
  - Updated Table 2-5, Table 2-6, Table 3-4, Table 3-6, Table 3-7, Table 3-8, Table 3-9, Table 5-1, Table 5-2, Table 5-3
- Figures
  - Updated Figure 2-6, Figure 2-7, Figure 2-8, Figure 4-1, Figure 4-2, Figure 4-7
- Minor updates to text and formatting were incorporated throughout the document

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

ISBN: 978-1-63276-392-1

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
═ ISO/TS 16949 ═

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

Worldwide Sales and Service

AMERICAS

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

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

Austin, TX
Tel: 512-257-3370

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

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

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

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

Detroit
Novi, MI
Tel: 248-848-4000

Houston, TX
Tel: 281-894-5983

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

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

New York, NY
Tel: 631-435-6000

San Jose, CA
Tel: 408-735-9110

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

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2943-5100
Fax: 852-2401-3431

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

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

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

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

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

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

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

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

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

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

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

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

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

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: 91-20-3019-1500

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

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

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

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

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

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

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

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

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

Taiwan - Kaohsiung
Tel: 886-7-213-7830

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

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

EUROPE

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

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

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

Germany - Dusseldorf
Tel: 49-2129-3766400

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

Germany - Pforzheim
Tel: 49-7231-424750

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

Italy - Venice
Tel: 39-049-7625286

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

Poland - Warsaw
Tel: 48-22-3325737

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

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

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

03/25/14

Looking for pricing, stock, or lifecycle information?

Click below to explore more details on WIN SOURCE:

- View DSPIC33EV64GM002-I/MM on WIN SOURCE
- Microchip Technology Information

Optimize Your Supply Chain with WIN SOURCE Solutions

- Global Sourcing Solution
- Obsolete Management
- Cost Control Management
- Shortage Management
- Alternative Solution
- Excess Inventory Management