
dsPIC33EPXXXGM3XX/6XX/7XX Flash Programming Specification

1.0 DEVICE OVERVIEW

This document defines the programming specification for the dsPIC33EPXXXGM3XX/6XX/7XX 16-bit Digital Signal Controller (DSC) family. This programming specification is required only for those developing programming support for the dsPIC33EPXXXGM3XX/6XX/7XX family. All other customers should use development tools that already provide support for device programming.

Topics covered include:

- [Section 1.0 “Device Overview”](#)
- [Section 2.0 “Programming Overview of the dsPIC33EPXXXGM3XX/6XX/7XX”](#)
- [Section 3.0 “Device Programming – Enhanced ICSP”](#)
- [Section 4.0 “Checksum Computation”](#)
- [Section 5.0 “The Programming Executive”](#)
- [Section 6.0 “Device Programming – ICSP”](#)
- [Section 7.0 “Programming the Programming Executive to Memory”](#)
- [Section 8.0 “Device ID”](#)
- [Section 9.0 “AC/DC Characteristics and Timing Requirements”](#)
- [Appendix A: “Hex File Format”](#)
- [Appendix B: “Revision History”](#)

2.0 PROGRAMMING OVERVIEW OF THE dsPIC33EPXXXGM3XX/6XX/7XX

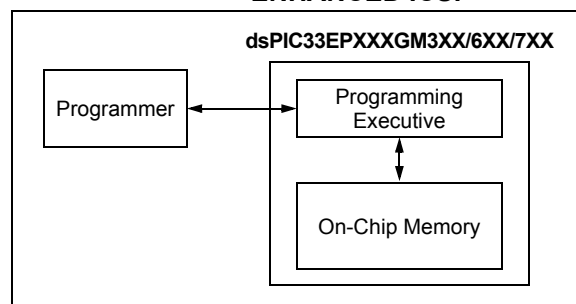
There are two methods of programming the dsPIC33EPXXXGM3XX/6XX/7XX family of devices discussed in this programming specification. They are:

- 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 chip.

The Enhanced ICSP protocol uses a faster method that takes advantage of the Programming Executive, as illustrated in [Figure 2-1](#). The Programming Executive 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 the dsPIC33EPXXXGM3XX/6XX/7XX devices without having to deal with the low-level programming protocols of the chip.

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



This specification is divided into major sections that describe the programming methods independently. [Section 3.0 “Device Programming – Enhanced ICSP”](#) describes the Enhanced ICSP method. [Section 6.0 “Device Programming – ICSP”](#) describes the ICSP method.

dsPIC33EPXXXGM3XX/6XX/7XX

2.1 Required Connections

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

Note: Refer to the specific device data sheet for complete pin diagrams of the dsPIC33EPXXXGM3XX/6XX/7XX family devices.

TABLE 2-1: PINS USED DURING PROGRAMMING

During Programming		
Pin Name	Pin Type	Pin Description
MCLR	I	Programming Enable
VDD and AVDD	P	Power Supply ⁽¹⁾
VSS and AVSS	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.

2.2 Program Memory Write/Erase Requirements

The program Flash memory on the dsPIC33EPXXXGM3XX/6XX/7XX devices 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 word can be programmed twice before an erase, but only if (a) the same data is used in both program operations or (b) bits containing '1' are set to '0' but no '0' is set to '1'.

2.3 Memory Map

The program memory map extends from 0x0 to 0xFFFFFE. Code storage is located at the base of the memory map and supports up to 175,104 instructions (about 512 Kbytes). [Table 2-2](#) shows the program memory size and number of erase and program blocks present in each device variant. Each erase block or page contains 512 instructions and each program block or row contains 64 instructions.

Locations, 0x800000 through 0x800FFE, are reserved for executive code memory. This region stores the Programming Executive and the debugging executive. The Programming Executive is used for device programming and the debug executive is used for in-circuit debugging. This region of memory cannot be used to store user code.

The last four locations of the executive code memory consist of four User ID registers (FUID0-FUID3) that are used for storing product information, such as serial numbers, system manufacturing dates, manufacturing lot numbers and other application-specific information.

Locations, 0xFF0000 and 0xFF0002, are reserved for the Device ID Word registers (DEVID). These bits can be used by the programmer to identify which device type is being programmed. They are described in [Section 8.0 "Device ID"](#). The Device ID registers read out normally, even after code protection is applied.

[Figure 2-2](#) shows the memory map for the dsPIC33EPXXXGM3XX/6XX/7XX family variants.

dsPIC33EPXXXGM3XX/6XX/7XX

TABLE 2-2: CODE MEMORY SIZE

dsPIC33EPXXXGM3XX/6XX/ 7XX Device	User Memory Address Limit (Instruction Words)	Write Blocks	Erase Blocks	Executive Memory Address Limit (Instruction Words)
dsPIC33EP128GM304	0x157FE (44032)	688	86	0x800000-0x800FFE (2K)
dsPIC33EP128GM604	0x157FE (44032)	688	86	0x800000-0x800FFE (2K)
dsPIC33EP128GM306	0x157FE (44032)	688	86	0x800000-0x800FFE (2K)
dsPIC33EP128GM706	0x157FE (44032)	688	86	0x800000-0x800FFE (2K)
dsPIC33EP128GM310	0x157FE (44032)	688	86	0x800000-0x800FFE (2K)
dsPIC33EP128GM710	0x157FE (44032)	688	86	0x800000-0x800FFE (2K)
dsPIC33EP256GM304	0x2AFFE (87552)	1368	171	0x800000-0x800FFE (2K)
dsPIC33EP256GM604	0x2AFFE (87552)	1368	171	0x800000-0x800FFE (2K)
dsPIC33EP256GM306	0x2AFFE (87552)	1368	171	0x800000-0x800FFE (2K)
dsPIC33EP256GM706	0x2AFFE (87552)	1368	171	0x800000-0x800FFE (2K)
dsPIC33EP256GM310	0x2AFFE (87552)	1368	171	0x800000-0x800FFE (2K)
dsPIC33EP256GM710	0x2AFFE (87552)	1368	171	0x800000-0x800FFE (2K)
dsPIC33EP512GM304	0x0557FE (175104)	2736	342	0x800000-0x800FFE (2K)
dsPIC33EP512GM604	0x0557FE (175104)	2736	342	0x800000-0x800FFE (2K)
dsPIC33EP512GM306	0x0557FE (175104)	2736	342	0x800000-0x800FFE (2K)
dsPIC33EP512GM706	0x0557FE (175104)	2736	342	0x800000-0x800FFE (2K)
dsPIC33EP512GM310	0x0557FE (175104)	2736	342	0x800000-0x800FFE (2K)
dsPIC33EP512GM710	0x0557FE (175104)	2736	342	0x800000-0x800FFE (2K)

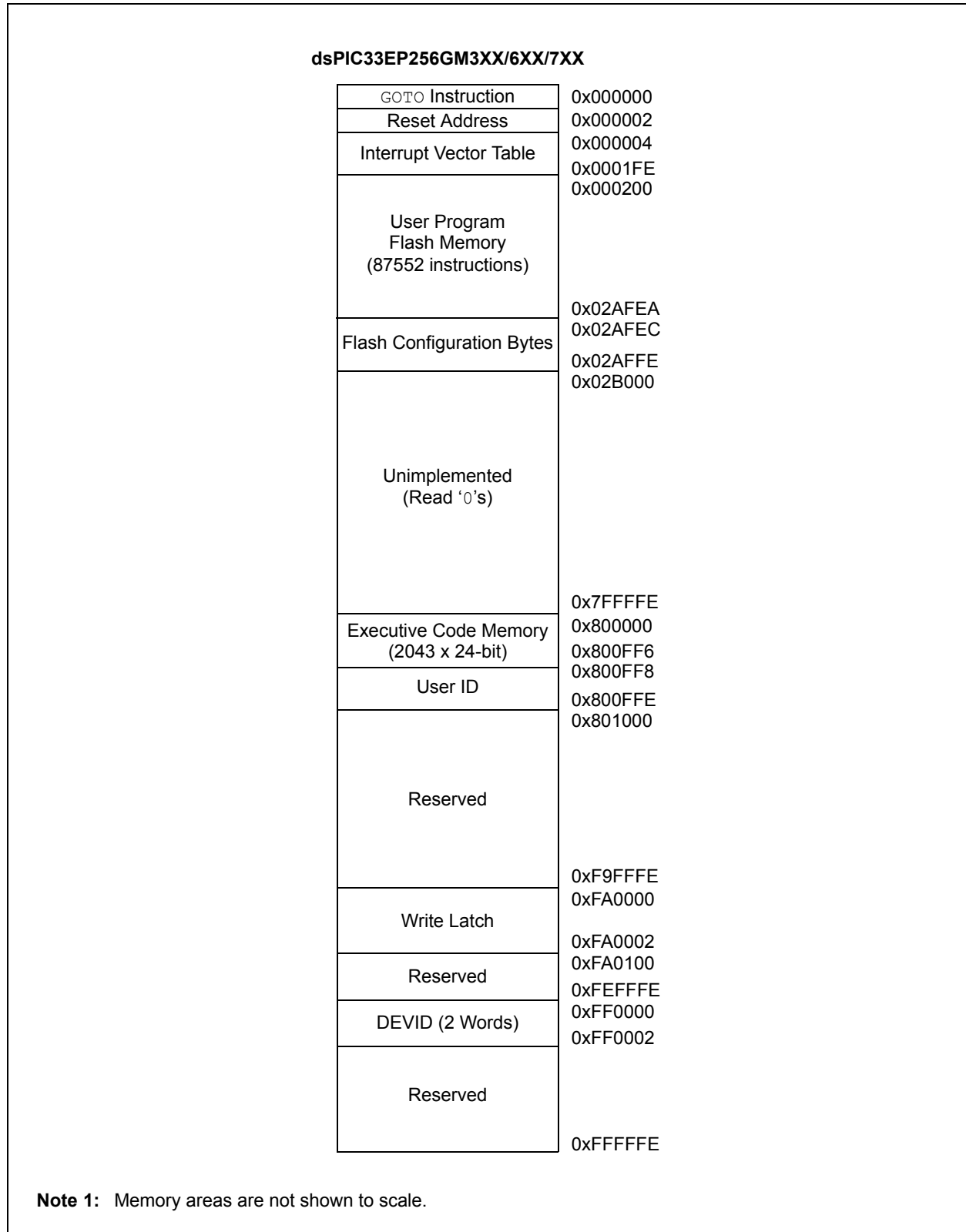
dsPIC33EPXXXGM3XX/6XX/7XX

FIGURE 2-2: PROGRAM MEMORY MAP FOR dsPIC33EP128GM3XX/6XX/7XX DEVICES

dsPIC33EP128GM3XX/6XX/7XX	
GOTO Instruction	0x000000
Reset Address	0x000002
Interrupt Vector Table	0x000004 0x0001FE 0x000200
User Program Flash Memory (44032 instructions)	0x0157EA 0x0157EC
Flash Configuration Bytes	0x0157FE 0x015800
Unimplemented (Read '0's)	0x7FFFFE
Executive Code Memory (2043 x 24-bit)	0x800000 0x800FF6 0x800FF8
User ID	0x800FFE 0x801000
Reserved	0xF9FFFF 0xFA0000
Write Latch	0xFA0002 0xFA0100
Reserved	0xFEFFFF 0xFF0000 0xFF0002
DEVID (2 Words)	
Reserved	0xFFFFFE

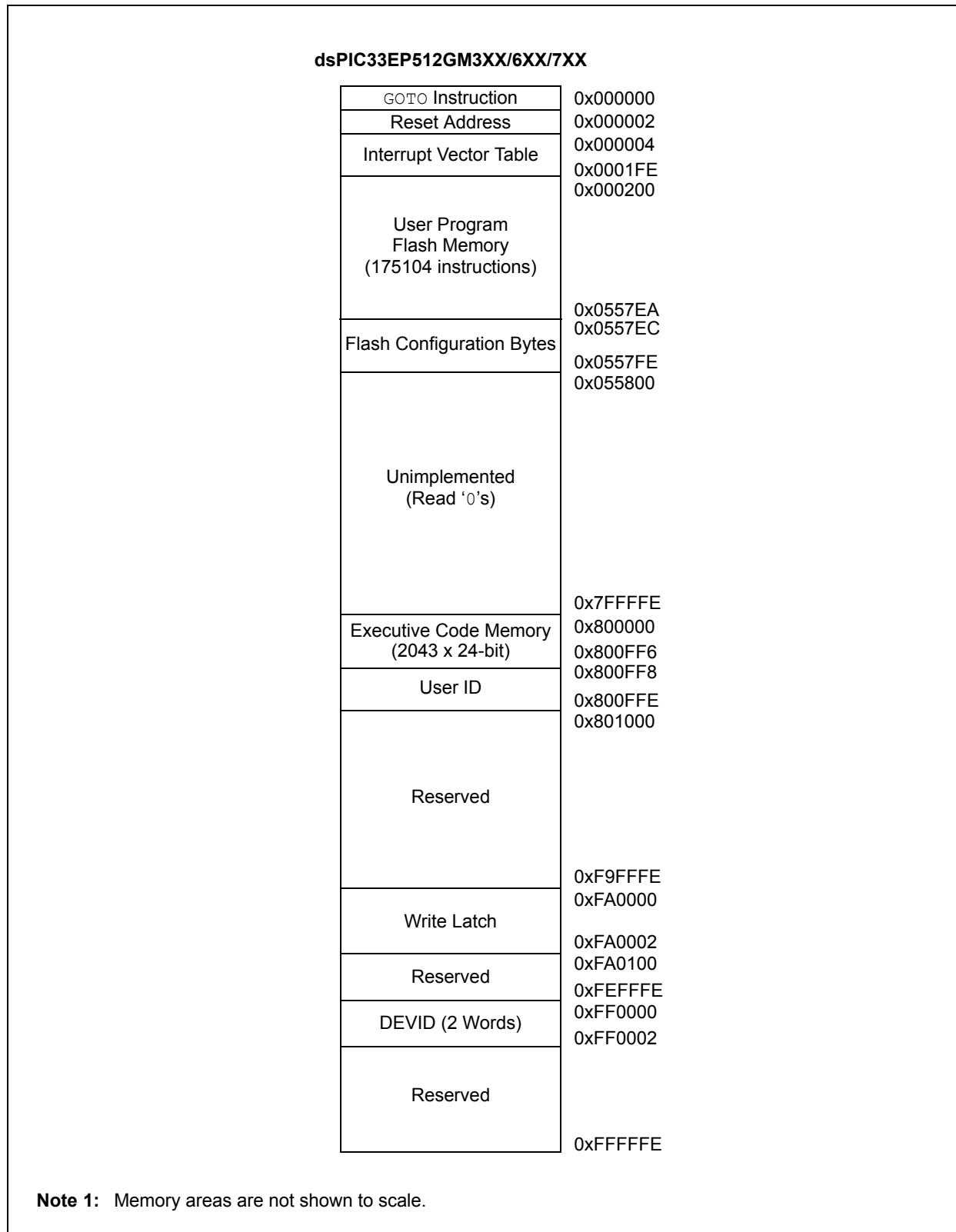
Note 1: Memory areas are not shown to scale.

FIGURE 2-3: PROGRAM MEMORY MAP FOR dsPIC33EP256GM3XX/6XX/7XX DEVICES



dsPIC33EPXXXGM3XX/6XX/7XX

FIGURE 2-4: PROGRAM MEMORY MAP FOR dsPIC33EP512GM3XX/6XX/7XX DEVICES



3.0 DEVICE PROGRAMMING – ENHANCED ICSP

This section discusses programming the device through Enhanced ICSP and the Programming Executive. The Programming Executive resides in executive memory (separate from code memory) and is executed when Enhanced ICSP Programming mode is entered. The Programming Executive provides the mechanism for the programmer (host device) to program and verify the dsPIC33EPXXXGM3XX/6XX/7XX family devices using a simple command set and communication protocol. There are several basic functions provided by the Programming Executive:

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

The Programming Executive performs the low-level tasks required for erasing, programming and verifying a device. This allows the programmer to program the device by issuing the appropriate commands and data. [Table 3-1](#) summarizes the commands. A detailed description for each command is provided in [Section 5.2 “Programming Executive Commands”](#).

TABLE 3-1: COMMAND SET SUMMARY

Command	Description
SCHECK	Sanity check.
READC	Read Configuration registers or Device ID registers.
READP	Read primary Flash memory.
PROGP	Program one row of code memory and verify.
PROG2W	Program a double instruction word of code memory and verify.
ERASEB	Bulk Erase the device.
ERASEP	Erase Page command.
CRCP	Performs CRC on memory.
QBLANK	Query to check whether code memory is blank.
QVER	Query the software version.

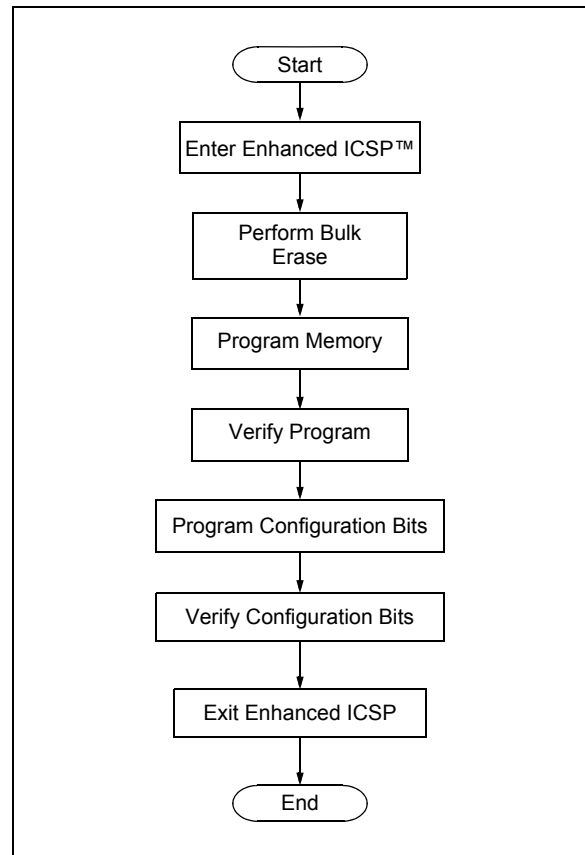
The Programming Executive uses the device's data RAM for variable storage and program execution. After the Programming Executive is run, no assumptions should be made about the contents of data RAM.

3.1 Overview of the Programming Process

[Figure 3-1](#) shows the high-level overview of the programming process. After entering Enhanced ICSP mode, the Programming Executive is verified. Next, the device is erased and then, the program Flash memory (primary Flash memory) is programmed, followed by the nonvolatile device Configuration registers. Code memory (including the Configuration registers) is then verified to ensure that programming was successful.

After the Programming Executive has been verified in memory (or loaded if not present), the dsPIC33EPXXXGM3XX/6XX/7XX devices can be programmed using the command set shown in [Table 3-1](#).

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



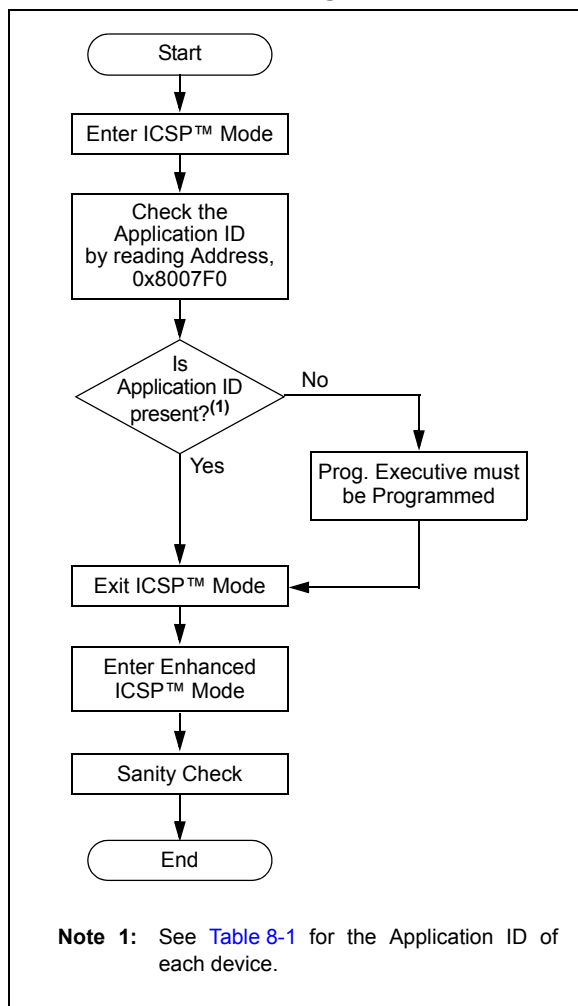
3.2 Confirming the Presence of the Programming Executive

Before programming, the programmer must confirm that the Programming Executive is stored in executive memory. The procedure for this task is shown in Figure 3-2.

First, ICSP mode is entered. Then, the unique Application ID Word stored in executive memory is read. If the Programming Executive is resident, the correct Application ID Word is read and programming can resume as normal. However, if the Application ID Word is not present, the Programming Executive must be programmed to executive code memory using the method described in Section 7.0 “Programming the Programming Executive to Memory”. See Table 8-1 for the Application ID of each device.

Section 6.0 “Device Programming – ICSP” describes the ICSP programming method. Section 6.10 “Reading the Application ID Word” describes the procedure for reading the Application ID Word in ICSP mode.

FIGURE 3-2: CONFIRMING PRESENCE OF PROGRAMMING EXECUTIVE



3.3 Entering Enhanced ICSP Mode

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

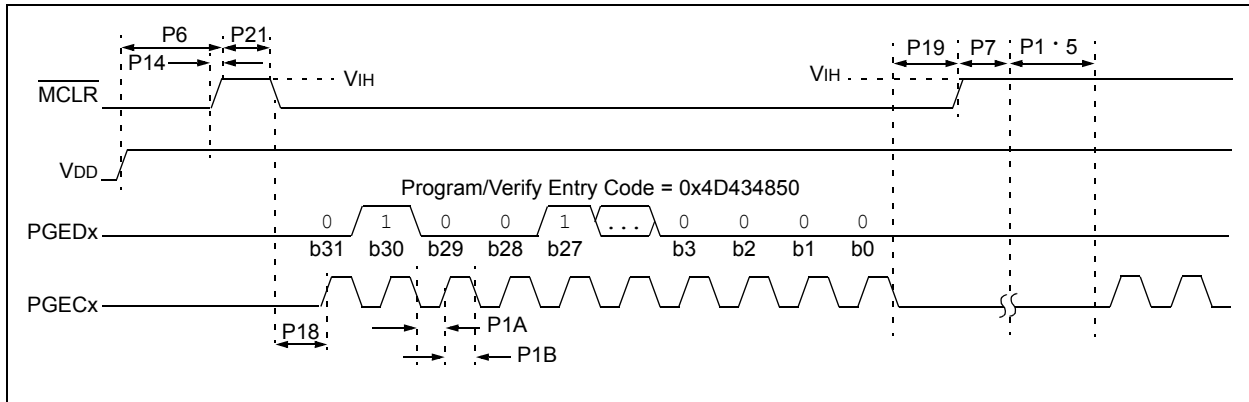
1. The $\overline{\text{MCLR}}$ pin is briefly driven high, then low.
2. A 32-bit key sequence is clocked into PGD.
3. $\overline{\text{MCLR}}$ is then driven high within a specified period of time and held.

The programming voltage applied to $\overline{\text{MCLR}}$ is V_{IH} , which is essentially V_{DD} in the case of dsPIC33EPXXXGM3XX/6XX/7XX devices. There is no minimum time requirement for holding at V_{IH} . After V_{IH} is removed, an interval of at least P18 must elapse before presenting the key sequence on PGD.

The key sequence is a specific 32-bit pattern, ‘0100 1101 0100 0011 0100 1000 0101 0000’ (more easily remembered as 0x4D434850 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, V_{IH} must be applied to $\overline{\text{MCLR}}$ 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 PGD. Signals appearing on PGD before this time has elapsed will not be interpreted as valid.

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

FIGURE 3-3: ENTERING ENHANCED ICSP™ MODE

3.4 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 (0xFF0000:0xFF0002) can be ignored by the Blank Check since this region stores device information that cannot be erased. The device Configuration registers are also ignored by the Blank Check. Additionally, all unimplemented memory space should be ignored from 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.

3.5 Code Memory Programming

3.5.1 PROGRAMMING METHODOLOGY

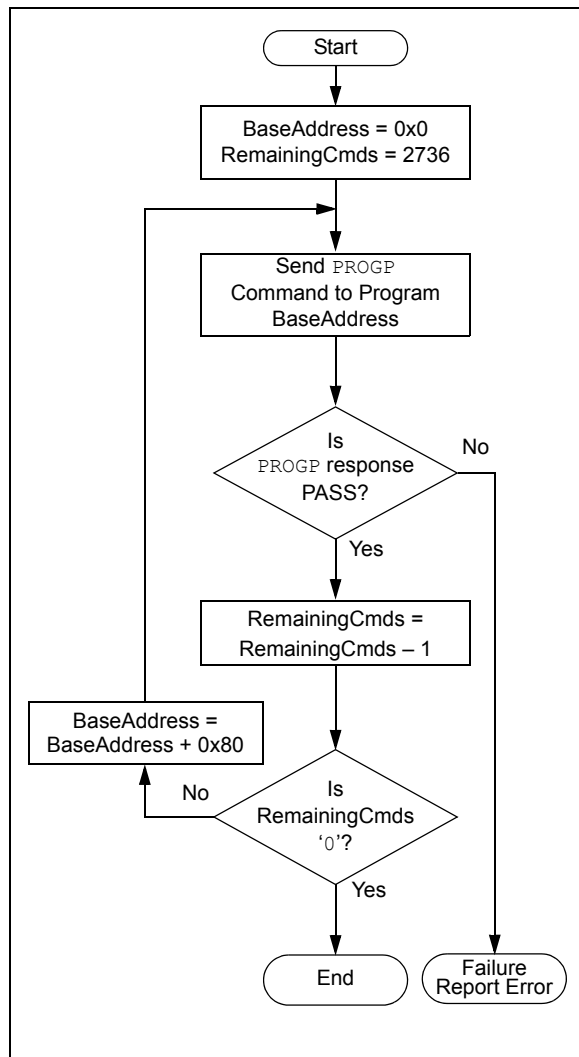
Code memory (primary Flash) is programmed with the `PROGP` command. `PROGP` programs one row of code memory, starting from the memory address specified in the command. The number of `PROGP` commands required to program a device depends on the number of write blocks that must be programmed in the device.

A flowchart for programming code memory is shown in [Figure 3-4](#). In this example, all 175,104 instruction words of a dsPIC33EPXXXGM3XX/6XX/7XX device are programmed. First, the number of commands to send (called ‘RemainingCmds’ in the flowchart) is set to 2736 and the destination address (called ‘BaseAddress’) is set to ‘0’. Next, one write block in the device is programmed with a `PROGP` command. Each `PROGP` command contains data for one row of code memory of the dsPIC33EPXXXGM3XX/6XX/7XX. After the first command is processed successfully, ‘RemainingCmds’ is decremented by ‘1’ and compared with ‘0’. Since there are more `PROGP` commands to send, ‘BaseAddress’ is incremented by 0x80 to point to the next row of memory.

On the second `PROGP` command, the second row is programmed. This process is repeated until the entire device is programmed.

Note: If a bootloader needs to be programmed, the bootloader code must not be programmed into the first page of code memory. For example, if a bootloader located at address, 0x200, attempts to erase the first page, it would inadvertently erase itself. Instead, program the bootloader into the second page (e.g., 0x400).

FIGURE 3-4: FLOWCHART FOR PROGRAMMING CODE MEMORY



3.5.2 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.

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

4.0 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 bytes

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

Table 4-1 is an example of the checksum calculation for the dsPIC33EP512GM710 device.

Table 4-2 describes the Configuration bit masks for each device.

TABLE 4-1: CHECKSUM COMPUTATION EXAMPLE

Device	Read Code Protection	Checksum Computation	Erased Value	Value with 0xAAAAAA at 0x0 and Last Code Address
dsPIC33EP512GM710	Disabled	CFGB + SUM(0:0x0557EA) ⁽¹⁾	0xF7F1	0xF3F5
	Enabled	Reads of program memory return 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 ((FGS & 0x03) + (FOSCSEL & 0xC7) + (FOSC & 0xE7) + (FWDT & 0xFF) + (FPOR & 0xF8) + (FICD & 0x67))

Note 1: For the checksum computation example, the Configuration bits are set to the recommended default value as shown in Table 4-2.

TABLE 4-2: CONFIGURATION BIT MASKS

Device	Configuration Bit Masks					
	FGS	FOSCSEL	FOSC	FWDT	FPOR	FICD
dsPIC33EP128GM304	0x03	0xC7	0xE7	0xFF	0xF8	0x67
dsPIC33EP128GM604	0x03	0xC7	0xE7	0xFF	0xF8	0x67
dsPIC33EP128GM306	0x03	0xC7	0xE7	0xFF	0xF8	0x67
dsPIC33EP128GM706	0x03	0xC7	0xE7	0xFF	0xF8	0x67
dsPIC33EP128GM310	0x03	0xC7	0xE7	0xFF	0xF8	0x67
dsPIC33EP128GM710	0x03	0xC7	0xE7	0xFF	0xF8	0x67
dsPIC33EP256GM304	0x03	0xC7	0xE7	0xFF	0xF8	0x67
dsPIC33EP256GM604	0x03	0xC7	0xE7	0xFF	0xF8	0x67
dsPIC33EP256GM306	0x03	0xC7	0xE7	0xFF	0xF8	0x67
dsPIC33EP256GM706	0x03	0xC7	0xE7	0xFF	0xF8	0x67
dsPIC33EP256GM310	0x03	0xC7	0xE7	0xFF	0xF8	0x67
dsPIC33EP256GM710	0x03	0xC7	0xE7	0xFF	0xF8	0x67
dsPIC33EP512GM304	0x03	0xC7	0xE7	0xFF	0xF8	0x67
dsPIC33EP512GM604	0x03	0xC7	0xE7	0xFF	0xF8	0x67
dsPIC33EP512GM306	0x03	0xC7	0xE7	0xFF	0xF8	0x67
dsPIC33EP512GM706	0x03	0xC7	0xE7	0xFF	0xF8	0x67
dsPIC33EP512GM310	0x03	0xC7	0xE7	0xFF	0xF8	0x67
dsPIC33EP512GM710	0x03	0xC7	0xE7	0xFF	0xF8	0x67

dsPIC33EPXXXGM3XX/6XX/7XX

4.1 Configuration Bits Programming

4.1.1 OVERVIEW

The dsPIC33EPXXXGM3XX/6XX/7XX devices have Configuration bits stored in eight 8-bit Configuration registers, aligned on even configuration memory address boundaries. These bits can be set or cleared to select various device configurations. There are three types of Configuration bits: system operation bits, code-protect bits and unit ID bits. The system operation bits determine the power-on settings for system level components, such as oscillator and Watchdog Timer. The code-protect bits prevent program memory from being read and written.

The register descriptions for the FGS, FOSCSEL, FOSC, FWDT, FPOR and FICD Configuration registers are shown in [Table 4-3](#).

The Configuration register map is shown in [Table 4-4](#).

TABLE 4-3: dsPIC33EPXXXGM3XX/6XX/7XX CONFIGURATION BITS DESCRIPTION

Bit Field	Register	Description
GCP	FGS	General Segment Code-Protect bit 1 = User program memory is not code-protected 0 = User program memory is code-protected
GWRP	FGS	General Segment Write-Protect bit 1 = User program memory is not write-protected 0 = User program memory is write-protected
IESO	FOSCSEL	Two-Speed Oscillator Start-up Enable bit 1 = Start-up device with FRC, then automatically switch to the user-selected oscillator source when ready 0 = Start-up device with user-selected oscillator source
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
PWMLOCK	FOSCSEL	PWM Lock Enable bit 1 = Certain PWM registers may only be written after the key sequence 0 = PWM registers may be written without the key sequence
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
IOL1WAY	FOSC	Peripheral Pin Select Configuration bit 1 = Allow only one reconfiguration 0 = Allow multiple reconfigurations
OSCIOFNC	FOSC	OSC2 Pin Function bit (except in XT and HS modes) 1 = OSC2 is clock output 0 = OSC2 is general purpose digital I/O pin
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

dsPIC33EPXXXGM3XX/6XX/7XX

TABLE 4-3: dsPIC33EPXXXGM3XX/6XX/7XX CONFIGURATION BITS DESCRIPTION (CONTINUED)

Bit Field	Register	Description
FWDTEN	FWDT	Watchdog Timer Enable bit 1 = Watchdog Timer is always enabled (LPRC oscillator cannot be disabled. Clearing the SWDTEN bit in the RCON register has no effect.) 0 = Watchdog Timer is enabled/disabled by user software (LPRC can be disabled by clearing the SWDTEN bit in the RCON register)
WINDIS	FWDT	Watchdog Timer Window Enable bit 1 = Watchdog Timer in Non-Window mode 0 = Watchdog Timer in Window mode
PLLKEN	FWDT	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 PLL lock
WDTPRE	FWDT	Watchdog Timer Prescaler bit 1 = 1:128 0 = 1:32
WDTPOST<3:0>	FWDT	Watchdog Timer Postscaler bits 1111 = 1:32,768 1110 = 1:16,384 • • • 0001 = 1:2 0000 = 1:1
WDTWIN<1:0>	FPOR	Watchdog Window Select bits 11 = WDT Window is 25% of WDT period 10 = WDT Window is 37.5% of WDT period 01 = WDT Window is 50% of WDT period 00 = WDT Window is 75% of WDT period
BOREN	FPOR	Brown-out Reset (BOR) Detection Enable bit 1 = BOR is enabled 0 = BOR is disabled
ALT2C2	FPOR	Alternate I ² C™ pins for I2C2 bit 1 = I2C2 is mapped to the SDA2/SCL2 pins 0 = I2C2 is mapped to the ASDA2/ASCL2 pins
ALT2C1	FPOR	Alternate I ² C pins for I2C1 bit 1 = I2C1 is mapped to the SDA1/SCL1 pins 0 = I2C1 is mapped to the ASDA1/ASCL1 pins
JTAGEN	FICD	JTAG Enable bit 1 = JTAG is enabled 0 = JTAG is disabled
ICS<1:0>	FICD	ICD Communication Channel Select bits 11 = Communicate on PGEC1 and PGED1 10 = Communicate on PGEC2 and PGED2 01 = Communicate on PGEC3 and PGED3 00 = Reserved, do not use

dsPIC33EPXXXGM3XX/6XX/7XX

TABLE 4-4: dsPIC33EPXXXGM3XX/6XX/7XX DEVICE CONFIGURATION REGISTER MAP

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0557EC	Reserved	—	—	—	—	—	—	—	—
0x0557EE	Reserved	—	—	—	—	—	—	—	—
0x0557F0	FICD	Reserved ⁽²⁾	—	JTAGEN	Reserved ⁽¹⁾	Reserved ⁽²⁾	—	ICS<1:0>	
0x0557F2	FPOR	WDTWIN<1:0>		ALT12C2	ALT12C1	BOREN	—	—	—
0x0557F4	FWDT	FWDTEN	WINDIS	PLLKEN	WDTPRE	WDTPOST<3:0>			
0x0557F6	FOSC	FCKSM<1:0>		IOL1WAY	—	—	OSCIOFNC	POSCMD<1:0>	
0x0557F8	FOSCSEL	IESO	PWMLOCK	—	—	—	FNOSC<2:0>		
0x0557FA	FGS	—	—	—	—	—	—	GCP	GWRP
0x0557FC	Reserved	—	—	—	—	—	—	—	—
0x0557FE	Reserved	—	—	—	—	—	—	—	—

Legend: — = unimplemented bit, read as '0'

Note 1: This bit is reserved and must be programmed as '0'.

Note 2: This bit is reserved and must be programmed as '1'.

4.1.2 BENEFIT OF USER UNIT ID

The dsPIC33EPXXXGM3XX/6XX/7XX devices contain four User ID Words, located at addresses, 0x800FF8 through 0x800FFE. The User ID 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 at program time and not at run time; they can be read at run time.

The User ID Words are part of the last page of executive memory, as a result performing a Page Erase of the last page of executive memory will also erase the User ID Words. The Executive Memory Bulk Erase command (NVMCON = 0x400F) will also erase the User ID Words.

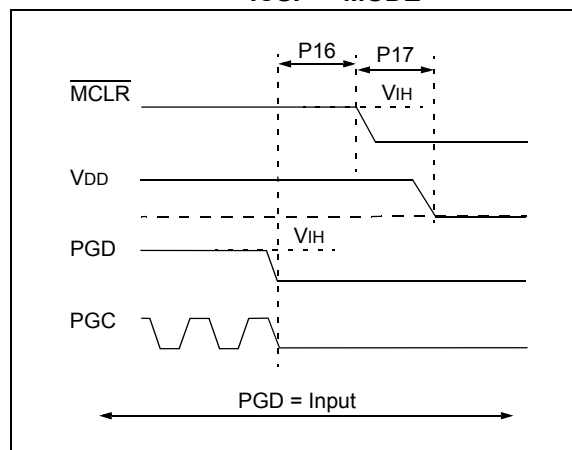
TABLE 4-5: USER ID WORDS REGISTER MAP

Name	Address	Bit 23-16	Bits 15-0
FUID0	0x800FF8	—	UID0
FUID1	0x800FFA	—	UID1
FUID2	0x800FFC	—	UID2
FUID3	0x800FFE	—	UID3

4.2 Exiting Enhanced ICSP Mode

Exiting Program/Verify mode is done by removing V_{IH} from MCLR, as shown in Figure 4-1. The only requirement for exit is that an interval, P16, should elapse between the last clock and program signals on PGC and PGD before removing V_{IH} .

FIGURE 4-1: EXITING ENHANCED ICSP™ MODE



5.0 THE PROGRAMMING EXECUTIVE

Note: The Programming Executive (PE) can be located within the following folder within your installation of MPLAB® IDE:
 ...\\Microchip\\MPLAB IDE\\REAL ICE,
 and then selecting the Hex PE file,
 RIPE_10_XXXXXX.hex (where XXXXXX
 is the version number).

5.1 Programming Executive Communication

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

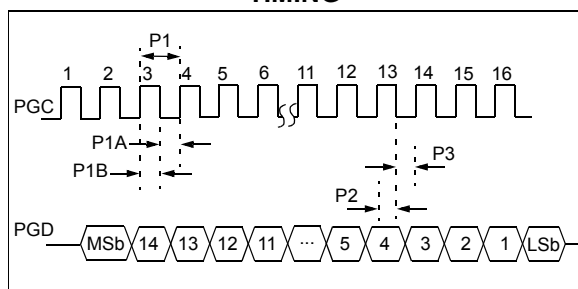
All communication is initiated by the programmer in the form of a command. Only one command at a time can be sent to the Programming Executive. In turn, the Programming Executive only sends one response to the programmer after receiving and processing a command. The Programming Executive command set is described in [Section 5.2 “Programming Executive Commands”](#). The response set is described in [Section 5.3 “Programming Executive Responses”](#).

5.1.1 COMMUNICATION INTERFACE AND PROTOCOL

The 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 Programming Executive.

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 Most Significant bit (MSb) first using 16-bit mode (see [Figure 5-1](#)).

FIGURE 5-1: 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 Programming Executive to drive this line high. The Programming Executive keeps the PGD line high to indicate that it is processing the command.

After the Programming Executive 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 the maximum wait (P9b) and it must provide the necessary amount of clock pulses to receive the entire response from the Programming Executive.

After the entire response is clocked out, the programmer should terminate the clock on PGC until it is time to send another command to the Programming Executive. This protocol is shown in [Figure 5-2](#).

5.1.2 SPI RATE

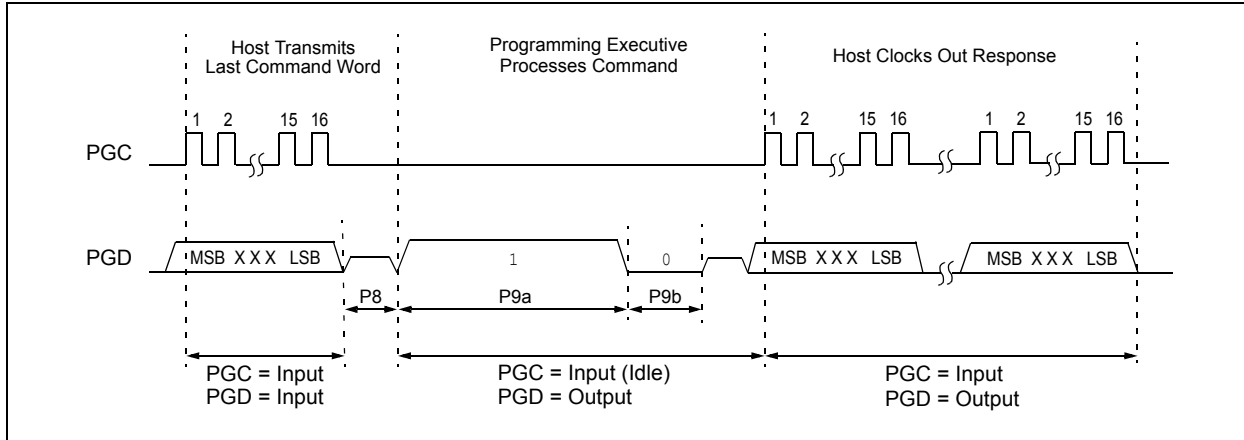
In Enhanced ICSP mode, the dsPIC33EPXXXGM3XX/6XX/7XX family devices operate from the Fast Internal RC oscillator, which has a nominal frequency of 7.3728 MHz. This oscillator frequency yields an effective system clock frequency of 3.6864 MHz. To ensure that the programmer does not clock too fast, it is recommended that a 1.8432 MHz clock be provided by the programmer.

5.1.3 TIME-OUTS

The Programming Executive uses no Watchdog Timer or time-out for transmitting responses to the programmer. If the programmer does not follow the flow control mechanism using PGC as described in [Section 5.1.1 “Communication Interface and Protocol”](#), it is possible that the Programming Executive will behave unexpectedly while trying to send a response to the programmer. Since the Programming Executive 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 5-1](#). If the command time-out expires, the programmer should reset the Programming Executive and start programming the device again.

FIGURE 5-2: PROGRAMMING EXECUTIVE – PROGRAMMER COMMUNICATION PROTOCOL



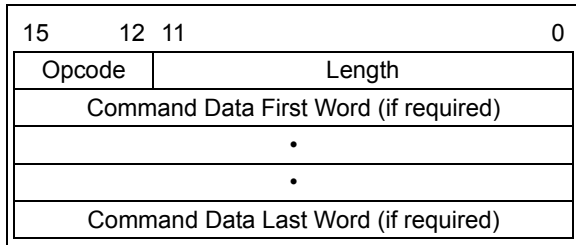
5.2 Programming Executive Commands

The Programming Executive command set is shown in Table 5-1. 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 (Section 5.2.4 “Command Descriptions”).

5.2.1 COMMAND FORMAT

All Programming Executive commands have a general format, consisting of a 16-bit header and any required data for the command (see Figure 5-3). 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 5-3: COMMAND FORMAT



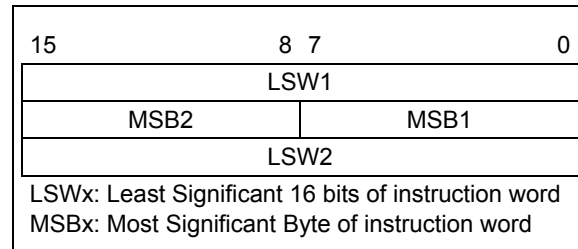
The command opcode must match one of those in the command set. Any command that is received, which does not match the list in Table 5-1, will return a “NACK” response (see Section 5.3.1.1 “Opcode Field”).

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

5.2.2 PACKED DATA FORMAT

When 24-bit instruction words are transferred across the 16-bit SPI interface, they are packed to conserve space using the format shown in Figure 5-4. This format minimizes traffic over the SPI and provides the Programming Executive with data that is properly aligned for performing table write operations.

FIGURE 5-4: PACKED INSTRUCTION WORD FORMAT



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

5.2.3 PROGRAMMING EXECUTIVE ERROR HANDLING

The Programming Executive will “NACK” all unsupported commands. Additionally, due to the memory constraints of the Programming Executive, no checking is performed on the data contained in the programmer command. It is the responsibility of the programmer to command the Programming Executive with valid command arguments or the programming operation may fail. Additional information on error handling is provided in Section 5.3.1.3 “QE_Code Field”.

TABLE 5-1: PROGRAMMING EXECUTIVE COMMAND SET

Opcode	Mnemonic	Length (16-bit words)	Time-out	Description
0x0	SCHECK	1	1 ms	Sanity check.
0x1	READC	3	1 ms	Read an 8-bit word from the specified Configuration register or Device ID register.
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	195	5 ms	Program one row of primary Flash memory at the specified 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 the device.
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 Programming Executive software version.
0xC	CRCP	5	1s	Performs 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.

Note: One row of code memory consists of (64) 24-bit words. Refer to [Table 2-2](#) for device-specific information.

5.2.4 COMMAND DESCRIPTIONS

All commands supported by the Programming Executive are described in [Section 5.2.5 “SCHECK Command”](#) through [Section 5.2.14 “QVER Command”](#).

5.2.5 SCHECK COMMAND

15	12	11	0
Opcode	Length		
Field	Description		
Opcode	0x0		
Length	0x1		

The **SCHECK** command instructs the Programming Executive to do nothing but generate a response. This command is used as a “Sanity Check” to verify that the Programming Executive is operational.

Expected Response (2 words):

0x1000
0x0002

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

dsPIC33EPXXXGM3XX/6XX/7XX

5.2.6 READC COMMAND

15	12	11	8	7	0
Opcode		Length			
N			Addr_MSB		
Addr_LS					

Field	Description
Opcode	0x1
Length	0x3
N	Number of 8-bit Configuration registers or Device ID registers to read (maximum of 256)
Addr_MSB	MSB of 24-bit source address
Addr_LS	Least Significant 16 bits of 24-bit source address

The **READC** command instructs the Programming Executive to read N Configuration registers 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 8-bit or 16-bit data.

When this command is used to read Configuration registers, the upper byte in every data word returned by the Programming Executive is 0x00 and the lower byte contains the Configuration register value.

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

0x1100

2 + N

Configuration register or Device ID Register 1

...

Configuration register or Device ID Register N

Note: Reading unimplemented memory will cause the Programming Executive to reset. Please ensure that only memory locations present on a particular device are accessed.

5.2.7 READP COMMAND

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

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 Programming Executive to read N 24-bit words of code memory, 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 5.2.2 "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: Reading unimplemented memory will cause the Programming Executive to reset. Please ensure that only memory locations present on a particular device are accessed.

5.2.8 PROG2W COMMAND

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

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 Programming Executive 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 Programming Executive verifies the programmed data against the data in the command.

Expected Response (2 words):

0x1300
0x0002

5.2.9 PROGP COMMAND

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

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 191
D_192	16-Bit Data Word 192

The **PROGP** command instructs the Programming Executive to program one row of code memory (128 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 0x100.

The data to program the memory, located in command words, D_1 through D_192, must be arranged using the packed instruction word format shown in [Figure 5-4](#).

After all data has been programmed to code memory, the Programming Executive 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.

dsPIC33EPXXXGM3XX/6XX/7XX

5.2.10 ERASEB COMMAND

15	12	11	0
Opcode		Length	

Field	Description
Opcode	0x7
Length	0x1

The **ERASEB** command instructs the Programming Executive to perform a Bulk Erase (i.e., erase all of the primary Flash memory, executive memory and code-protect bits).

Expected Response (2 words):

0x1700

0x0002

5.2.11 ERASEP COMMAND

15	12	11	8	7	0
Opcode		Length			
NUM_PAGES			Addr_MSB		
Addr_LS					

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 Programming Executive to Page Erase [NUM_PAGES] of code memory. The code memory must be erased at an “even” 1024 instruction word address boundary.

Expected Response (2 words):

0x1900

0x0002

5.2.12 CRCP COMMAND

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

Field	Description
Opcode	0xC
Length	0x5
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 5-4](#), byte-wise, Least Significant Byte (LSB) first.

Example:

CRC-CCITT-16 with test data of “123456789” becomes 0x29B1

Expected Response (3 words):

QE_Code: 0x1C00

Length: 0x0003

CRC Value: 0XXXXX

5.2.13 QBLANK COMMAND

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

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 Programming Executive 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 0xF0 if the specified code memory is blank; otherwise, **QBLANK** returns a QE_Code of 0x0F.

Expected Response (2 words for blank device):

0x1DF0
0x0002

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

0x1D0F
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.

5.2.14 QVER COMMAND

15	12	11	0
Opcode	Length		

Field	Description
Opcode	0xB
Length	0x1

The **QVER** command queries the version of the Programming Executive 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 Programming Executive software).

Expected Response (2 words):

0x1BMN (where "MN" stands for version M.N)
0x0002

5.3 Programming Executive Responses

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

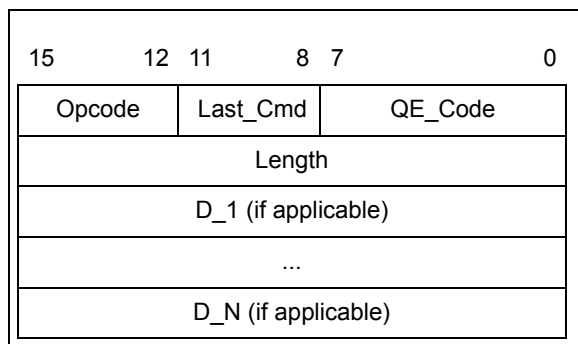
The Programming Executive response set is shown in [Table 5-2](#). This table contains the opcode, mnemonic and description for each response. The response format is described in [Section 5.3.1 "Response Format"](#).

TABLE 5-2: PROGRAMMING EXECUTIVE RESPONSE OPCODES

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

5.3.1 RESPONSE FORMAT

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



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

5.3.1.1 Opcode Field

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

5.3.1.2 Last_Cmd Field

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

5.3.1.3 QE_Code Field

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

When the Programming Executive processes one of the two query commands (QBLANK or QVER), the returned opcode is always PASS and the QE_Code holds the query response data. The format of the QE_Code for both queries is shown in [Table 5-3](#).

TABLE 5-3: QE_Code FOR QUERIES

Query	QE_Code
QBLANK	0x0F = Code memory is NOT blank 0xF0 = Code memory is blank
QVER	0xMN, where Programming Executive software version = M.N (i.e., 0x32 means software version 3.2)

When the Programming Executive processes any command other than a Query, the QE_Code represents an error code. Supported error codes are shown in [Table 5-4](#). 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 PROG command fails, the QE_Code is set to 0x1. For all other Programming Executive errors, the QE_Code is 0x2.

TABLE 5-4: QE_Code FOR NON-QUERY COMMANDS

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

5.3.1.4 Response Length

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

With the exception of the response for the READP command, the length of each response is only 2 words.

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

6.0 DEVICE PROGRAMMING – ICSP

ICSP mode is a special programming protocol that allows you to read and write to the dsPIC33EPXXXGM3XX/6XX/7XX device family memory. The ICSP mode is the most direct method used to program the device; however, note that Enhanced ICSP is faster. ICSP mode also has the ability to read the contents of executive memory to determine if the Programming Executive is present. This capability is accomplished by applying control codes and instructions serially to the device using pins, PGC and PGD.

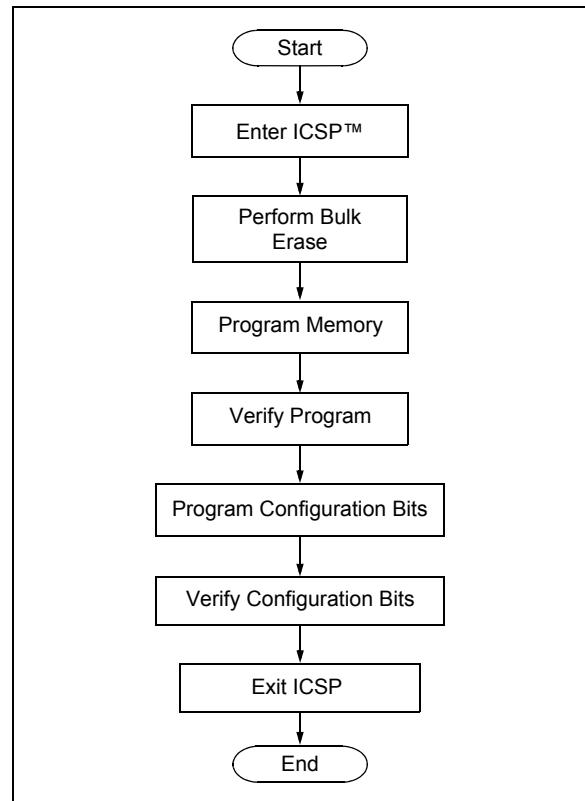
In ICSP mode, the system clock is taken from the PGC pin, regardless of the device's oscillator Configuration bits. All instructions are shifted serially into an internal buffer, then loaded into the Instruction Register (IR) and executed. No program fetching occurs from internal memory. Instructions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

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

6.1 Overview of the Programming Process

Figure 6-1 shows the high-level overview of the programming process. After entering ICSP mode, the first action is to Bulk Erase the device. Next, the code memory is programmed, followed by the device Configuration registers. Code memory (including the Configuration registers) is then verified to ensure that programming was successful. Then, program the code-protect Configuration bits, if required.

FIGURE 6-1: HIGH-LEVEL ICSP™ PROGRAMMING FLOW



6.2 Entering ICSP Mode

As shown in [Figure 6-5](#), entering ICSP Program/Verify mode requires three steps:

1. $\overline{\text{MCLR}}$ is briefly driven high, then low (P21).⁽¹⁾
2. A 32-bit key sequence is clocked into PGD.
3. $\overline{\text{MCLR}}$ is then driven high within a specified period of time and held.

Note 1: The $\overline{\text{MCLR}}$ capacitor value can vary the high time required for entering ICSP mode.

The programming voltage applied to $\overline{\text{MCLR}}$ is V_{IH} , which is essentially V_{DD} in the case of dsPIC33EPXXXGM3XX/6XX/7XX devices. There is no minimum time requirement for holding at V_{IH} . After V_{IH} is removed, an interval of at least P18 must elapse before presenting the key sequence on PGD.

The key sequence is a specific 32-bit pattern, '0100 1101 0100 0011 0100 1000 0101 0001' (more easily remembered as 0x4D434851 in hexadecimal). The device will enter Program/Verify mode only if the sequence is valid. The Most Significant bit of the most significant nibble must be shifted in first.

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

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

6.3 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 PGC and PGD, and this control code is used to command the CPU (see [Table 6-1](#)).

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

TABLE 6-1: CPU CONTROL CODES IN ICSP™ MODE

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

6.3.1 SIX SERIAL INSTRUCTION EXECUTION

The SIX control code allows execution of the dsPIC33EPXXXGM3XX/6XX/7XX device 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 clock cycles. While the received instruction is executed, the state machine simultaneously shifts in the next 4-bit command (see [Figure 6-3](#)).

Note 1: Coming out of the ICSP entry sequence, the first 4-bit control code is always forced to SIX and a forced NOP instruction is executed by the CPU. Five additional PGC clocks are needed on start-up, thereby resulting in a 9-bit SIX command instead of the normal 4-bit SIX command. After the forced SIX is clocked in, ICSP operation resumes as normal (the next 24 clock cycles load the first instruction word to the CPU). For more information, see [Figure 6-2](#).

- 2:** TBLRDH and TBLRDL instructions must be followed by five NOP instructions. TBLWTH and TBLWTL instructions must be followed by two NOP instructions.

6.3.2 REGOUT SERIAL INSTRUCTION EXECUTION

The REGOUT control code allows for 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, over the PGD pin. After the REGOUT control code is received, the CPU is held Idle for eight cycles. After these eight cycles, an additional 16 cycles are required to clock the data out (see [Figure 6-4](#)).

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

Note: The device will latch input PGD data on the rising edge of PGC and will output data on the PGD line on the rising edge of PGC. For all data transmissions, the Least Significant bit (LSb) is transmitted first.

FIGURE 6-2: PROGRAM ENTRY AFTER RESET

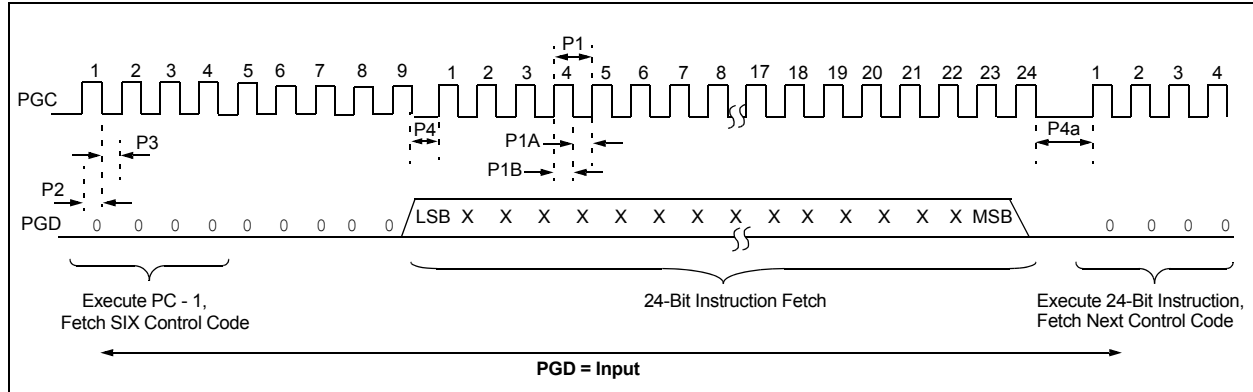


FIGURE 6-3: SIX SERIAL EXECUTION

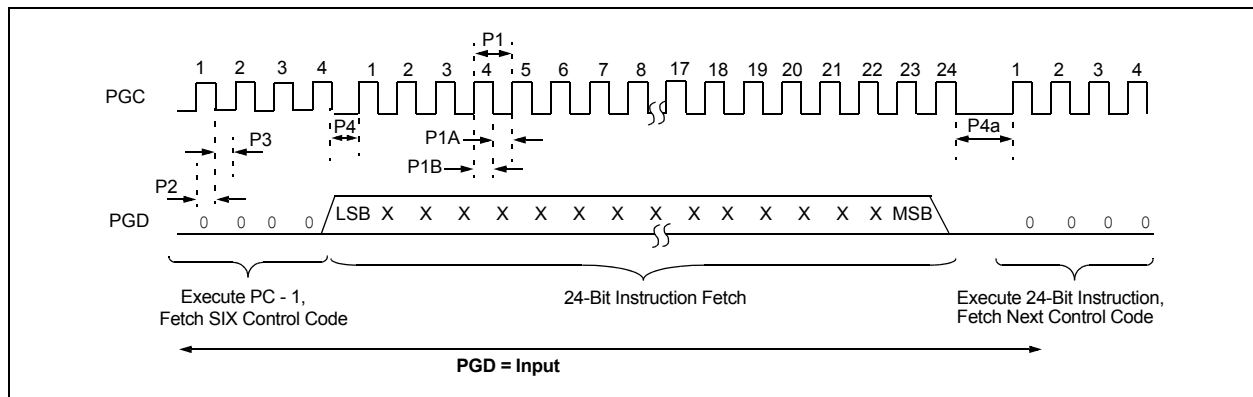
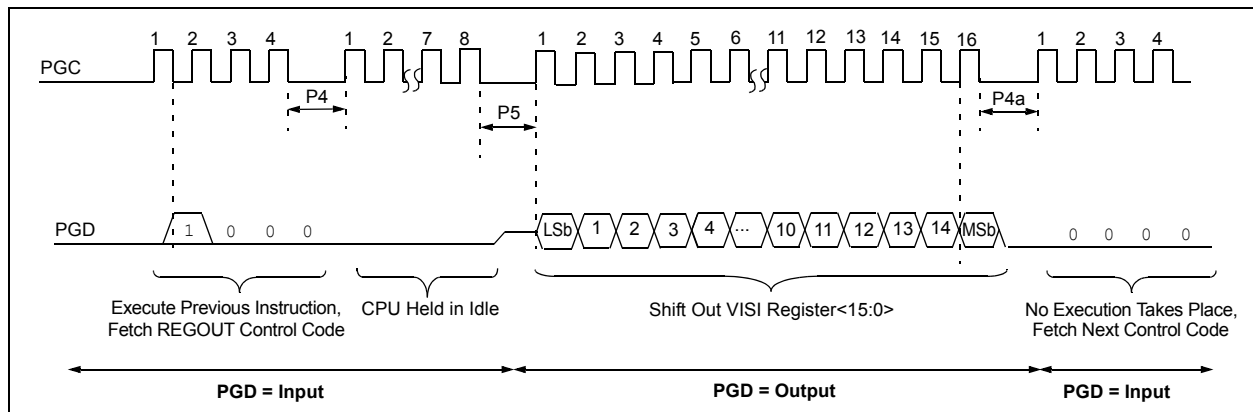


FIGURE 6-4: REGOUT SERIAL EXECUTION



6.4 Flash Memory Programming in ICSP Mode

6.4.1 PROGRAMMING OPERATIONS

Flash memory write and erase operations are controlled by the NVMCON register. Programming is performed by setting NVMCON to select the type of erase operation (Table 6-2) or write operation (Table 6-3) and initiating the programming by setting the WR control bit (NVMCON<15>).

In ICSP mode, all programming operations are self-timed. There is an internal delay between the user setting the WR control bit and the automatic clearing of the WR control bit when the programming operation is complete. For more information about the delays associated with various programming operations, see Section 9.0 “AC/DC Characteristics and Timing Requirements”.

TABLE 6-2: NVMCON ERASE OPERATIONS

NVMCON Value	Erase Operation
0x400F	Bulk Erase user memory, executive memory, and User ID Words (does not erase Device ID registers)
0x400E	Bulk Erase user address space
0x400D	Bulk Erase General Segment
0x4003	Erase a page of code memory or executive memory

TABLE 6-3: NVMCON WRITE OPERATIONS

NVMCON Value	Write Operation
0x4002	Program 1 row (64 instruction words) of primary Flash memory or executive memory
0x4001	Program an even-odd pair of words in primary Flash memory

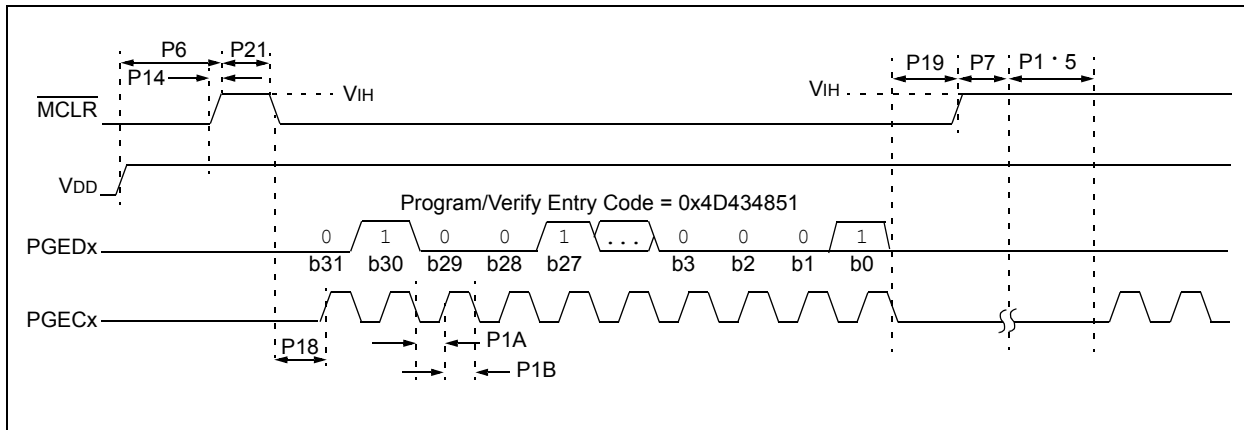
6.4.2 STARTING AND STOPPING A PROGRAMMING CYCLE

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

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

```
BSET    NVMCON, #WR
```

FIGURE 6-5: ENTERING ICSP™ MODE



REGISTER 6-1: NVMCON: NON-VOLATILE MEMORY (NVM) CONTROL REGISTER

R/SO-0 ⁽¹⁾	R/W-0 ⁽¹⁾	R/W-0 ⁽¹⁾	R/W-0	U-0	U-0	R/W-0	U-0
WR	WREN	WRERR	NVMSIDL ⁽²⁾	—	—	RPDF	URERR
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⁾
—	—	—	—	NVMOP<3:0> ^(3,5)			
bit 7				bit 0			

Legend:	SO = Satiabie Only bit
R = Readable bit	W = Writable bit
-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 = Enable Flash program/erase operations
0 = Inhibit Flash program/erase operations
- bit 13 **WRERR:** Write Sequence Error Flag bit⁽¹⁾
1 = An improper program or erase sequence attempt, or termination has occurred (bit is set automatically on any set attempt of the WR bit)
0 = The program or erase operation completed normally
- bit 12 **NVMSIDL:** NVM Stop-in-Idle Control bit⁽²⁾
1 = Discontinue primary Flash operation when the device enters Idle mode
0 = Continue primary Flash operation when the device enters Idle mode
- bit 11-10 **Unimplemented:** Read as '0'
- bit 9 **RPDF:** Bus Mastered Row Programming Data Format Control Bit
1 = Row data to be stored in RAM in compressed format
0 = Row data to be stored in RAM in uncompressed format
- bit 8 **URERR:** Row Programming Data Underrun Error Flag Bit
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 POR.
- 2:** When exiting Idle mode, there is a delay (TNPD) before Flash memory becomes operational.
- 3:** All other combinations of NVMOP<3:0> are unimplemented.
- 4:** The entire segment is erased with the exception of IVT.
- 5:** Execution of the PWRSAV instruction is ignored while any of the NVM operations are in progress.

REGISTER 6-1: NVMCON: NON-VOLATILE MEMORY (NVM) CONTROL REGISTER (CONTINUED)

bit 3-0	NVMOP<3:0> : NVM Operation Select bits ^(1,3,5)
	1111 = User memory and executive memory Bulk Erase operation ⁽⁴⁾
	1110 = User memory Bulk Erase operation
	1101 = Bulk Erase primary program Flash memory
	1100 = Reserved
	1011 = Reserved
	1010 = Reserved
	0011 = Memory Page Erase operation
	0010 = Memory Row Program operation
	0001 = Memory double-word (even-odd pair of words aligned at an address that is a multiple of 0x4) program operation
	0000 = Reserved

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

2: When exiting Idle mode, there is a delay (TNPD) before Flash memory becomes operational.

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

4: The entire segment is erased with the exception of IVT.

5: Execution of the `PWRSV` instruction is ignored while any of the NVM operations are in progress.

6.5 Erasing Program Memory

The procedure for erasing program memory (all of primary Flash memory, executive memory and code-protect Configuration bits) consists of setting NVMCON to 0x400F and then executing the programming cycle. For segment erase operations, the NVMCON value should be modified suitably, according to [Table 6-2](#).

[Figure 6-6](#) shows the ICSP programming process for Bulk Erasing program memory. This process includes the ICSP command code, which must be transmitted (for each instruction), Least Significant bit first, using the PGC and PGD pins (see [Figure 6-2](#)).

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

FIGURE 6-6: BULK ERASE FLOW

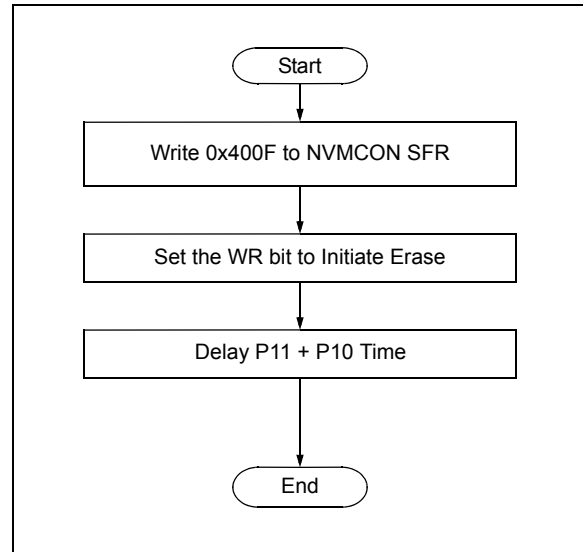


TABLE 6-4: SERIAL INSTRUCTION EXECUTION FOR BULK ERASING CODE MEMORY

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 to erase all program memory.		
0000	2400EA	MOV #0x400F, 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: Wait for Bulk Erase operation to complete and make sure WR bit is clear.		
—	—	Externally time, 'P11' msec (see Section 9.0 “AC/DC Characteristics and Timing Requirements”), to allow sufficient time for the Bulk Erase operation to complete.

6.6 Writing Code Memory

The procedure for writing code memory (primary Flash) is similar to the procedure for writing the Configuration registers, except that 64 instruction words are programmed at a time. To facilitate this operation, working registers, W0:W5, are used as temporary holding registers for the data to be programmed.

Table 6-5 shows the ICSP programming details, including the serial pattern with the ICSP command code, which must be transmitted, Least Significant bit first, using the PGC and PGD pins (see Figure 6-2). In Step 1, the Reset vector is exited.

To minimize the programming time, the same packed instruction format that the Programming Executive uses is utilized (Figure 5-4). In Step 2, point to the start of the code in the data memory. In Step 3, set the NVMADRU/NVMADR register pair to point to the correct row. In Step 4, set the NVMCON to program 64 instruction words. In Step 5, initiate the write cycle. In Step 6, wait for Row Program operation to complete and make sure the WR bit is clear. In Step 7, repeat Steps 2-6 until all code memory is programmed.

TABLE 6-5: SERIAL INSTRUCTION EXECUTION FOR WRITING CODE MEMORY (PRIMARY FLASH)

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: Point the NVMSRCADRH and NVMSRCADRL registers to the start of code in RAM.		
0000	2xxxx2	MOV #Destination Address<15:0>, W2
0000	2xxxx3	MOV #Destination Address<23:16>, W3
0000	8xxxx2	MOV W2, NVMSRCADRL
0000	8xxxx3	MOV W3, NVMSRCADRH
Step 3: Set the NVMADRU/NVMADR register pair to point to the correct row.		
0000	2xxxx2	MOV #DestinationAddress<15:0>, W2
0000	2xxxx3	MOV #DestinationAddress<23:16>, W3
0000	883963	MOV W3, NVMADRU
0000	883952	MOV W2, NVMADR
Step 4: Set the NVMCON to program 64 instruction words.		
0000	24002A	MOV #0x4002, W10
0000	88394A	MOV W10, NVMCON
0000	000000	NOP
0000	000000	NOP
Step 5: 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 ⁽¹⁾
0000	000000	NOP ⁽¹⁾
0000	000000	NOP ⁽¹⁾
0000	000000	NOP ⁽¹⁾
0000	000000	NOP ⁽¹⁾
0000	000000	NOP ⁽¹⁾

Note 1: These three NOP instructions must be transmitted with a clock frequency greater than 2 MHz.

TABLE 6-5: SERIAL INSTRUCTION EXECUTION FOR WRITING CODE MEMORY (PRIMARY FLASH) (CONTINUED)

Command (Binary)	Data (Hex)	Description
Step 6: Wait for Row Program operation to complete and make sure WR bit is clear.		
—	—	Externally time, 'P13' msec (see Section 9.0 “AC/DC Characteristics and Timing Requirements”), to allow sufficient time for the Row Program operation to complete.
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 7: Repeat Steps 2-6 until all code memory is programmed.		

Note 1: These three NOP instructions must be transmitted with a clock frequency greater than 2 MHz.

FIGURE 6-7: PROGRAM CODE MEMORY FLOW

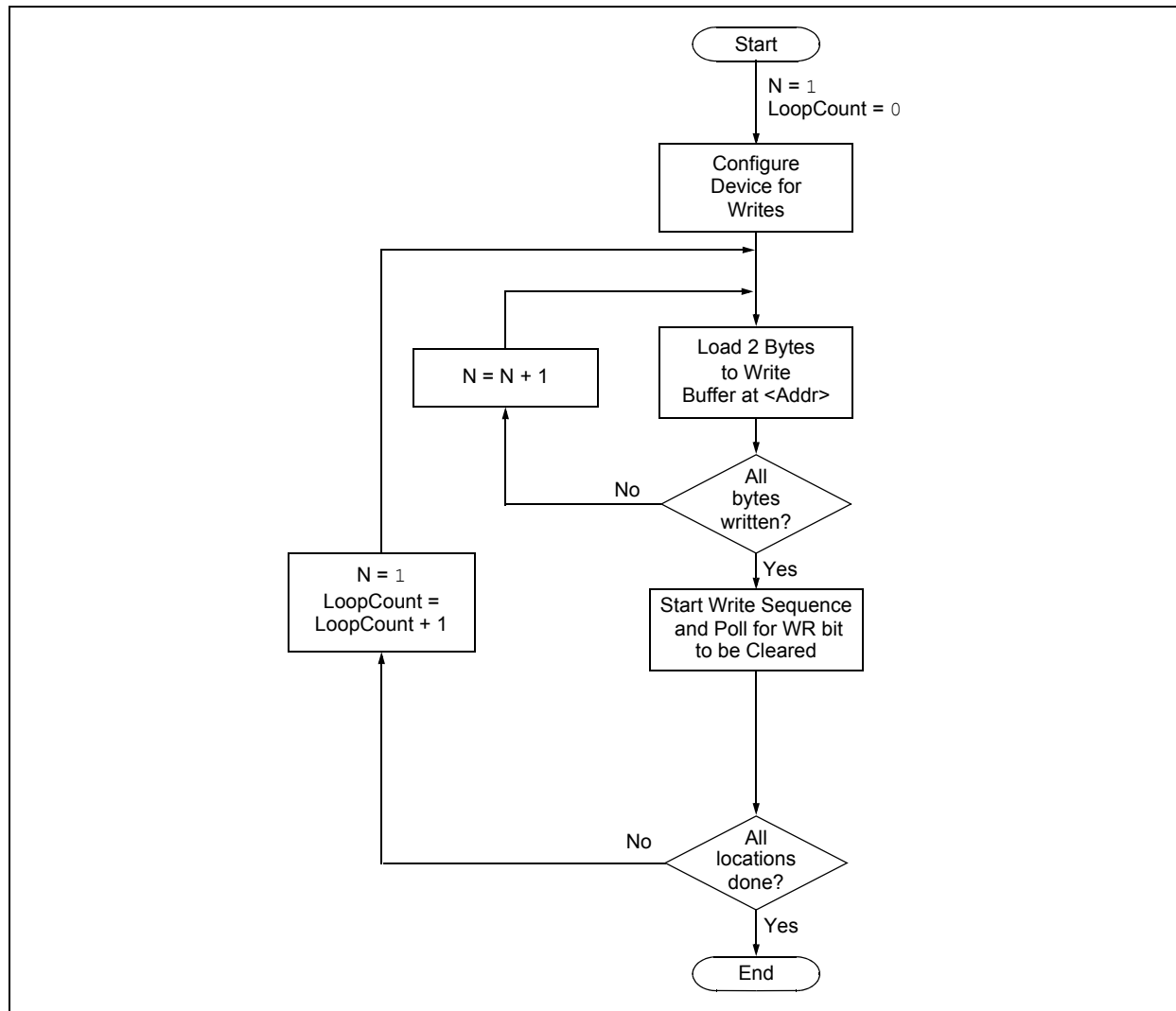


TABLE 6-6: DEFAULT CONFIGURATION REGISTER VALUES

Address	Name	Default Value
0x0557F0	FICD	0x67
0x0557F2	FPOR	0xF8
0x0557F4	FWDT	0xFF
0x0557F6	FOSC	0xE7
0x0557F8	FOSCSEL	0xC7
0x0557FA	FGS	0x03

6.7 Reading Code Memory

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

Table 6-7 shows the ICSP programming details for reading code memory. In Step 1, the Reset vector is exited. In Step 2, the 24-bit starting source address for reading is loaded into the `TBLPAG` register and `W6` register. The upper byte of the starting source address is stored in `TBLPAG` and the lower 16 bits of the source address are stored in `W6`.

To minimize the reading time, the packed instruction word format that was utilized for writing is also used for reading (see Figure 6-8). In Step 3, the Write Pointer, `W7`, is initialized. In Step 4, two instruction words are read from code memory and clocked out of the device, through the `VISI` register, using the `REGOUT` command. Step 4 is repeated until the desired amount of code memory is read.

FIGURE 6-8: PACKED INSTRUCTION WORDS IN W0:W5

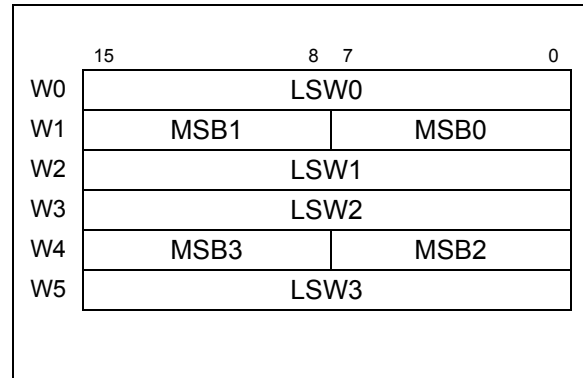


TABLE 6-7: SERIAL INSTRUCTION EXECUTION FOR READING CODE MEMORY (PRIMARY)

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 <code>TBLPAG</code> and the Read Pointer (<code>W6</code>) for <code>TBLRD</code> instruction.		
0000	200xx0	MOV #<SourceAddress23:16>, W0
0000	8802A0	MOV W0, TBLPAG
0000	2xxxx6	MOV #<SourceAddress15:0>, W6

dsPIC33EPXXXGM3XX/6XX/7XX

TABLE 6-7: SERIAL INSTRUCTION EXECUTION FOR READING CODE MEMORY (PRIMARY) (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
0000	000000	NOP

TABLE 6-7: SERIAL INSTRUCTION EXECUTION FOR READING CODE MEMORY (PRIMARY) (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: Repeat Step 4 until all desired code memory is read.		
Step 6: Reset 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

dsPIC33EPXXXGM3XX/6XX/7XX

6.8 Reading Configuration Memory

The procedure for reading configuration memory is similar to the procedure for reading code memory, except that 16-bit data words are read (with the upper byte read being all '0's) instead of 24-bit words. Since there are eight Configuration registers, they are read one register at a time.

Table 6-8 shows the ICSP programming details for reading all of the configuration memory. Note that the TBLPAG register is loaded 0x05 (the upper byte address of configuration memory) and the Read Pointer, W6, is initialized to 57EC.

TABLE 6-8: SERIAL INSTRUCTION EXECUTION FOR READING ALL CONFIGURATION 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 TBLPAG, the Read Pointer (W6) and the Write Pointer (W7) for TBLRD instruction.		
0000	200F80	MOV #05, W0
0000	8802A0	MOV W0, TBLPAG
0000	200046	MOV #0x57EC, W6
0000	20F887	MOV #VISI, W7
0000	000000	NOP
Step 3: Read the Configuration register and write it to the VISI register (located at 0x784) and clock out the VISI register using the REGOUT command.		
0000	BA0BB6	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 Step 3 eight times to read all the Configuration registers.		
Step 5: Reset 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

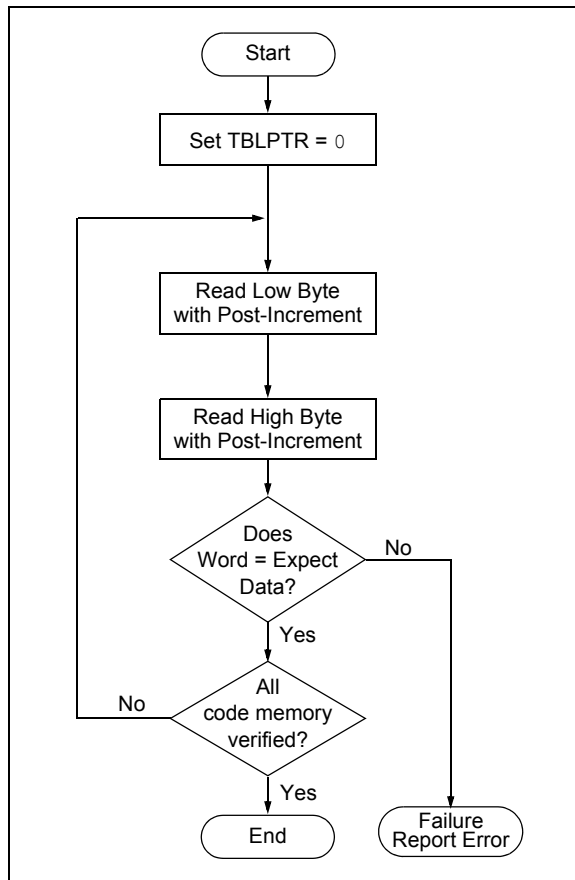
6.9 Verify Code Memory and Configuration Word

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

The verify process is shown in the flowchart in Figure 6-9. Memory reads occur, a single byte at a time, so two bytes must be read to compare against the word in the programmer's buffer. Refer to Section 6.7 "Reading Code Memory" for implementation details of reading code memory.

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

FIGURE 6-9: VERIFY CODE MEMORY FLOW



6.10 Reading the Application ID Word

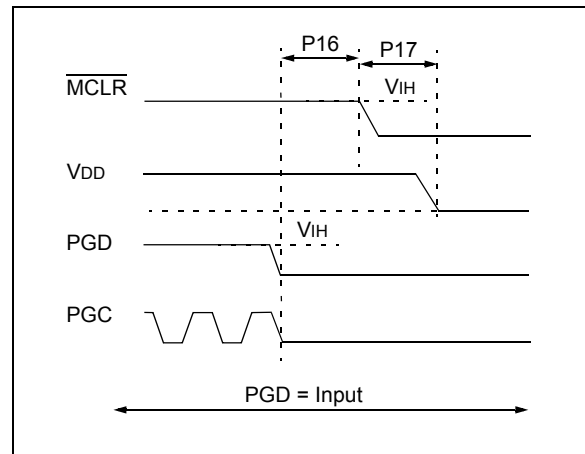
The Application ID Word is stored at address, 0x800FF0, in executive code memory. To read this memory location, you must use the SIX control code to move this program memory location to the VISI register. Then, the REGOUT control code must be used to clock the contents of the VISI register out of the device. The corresponding control and instruction codes, that must be serially transmitted to the device to perform this operation, are shown in Table 6-9.

After the programmer has clocked out the Application ID Word, it must be inspected. If the Application ID has the value, 0xDD, the Programming Executive is resident in memory and the device can be programmed using the mechanism described in Section 3.0 "Device Programming – Enhanced ICSP". However, if the application ID has any other value, the Programming Executive is not resident in memory; it must be loaded to memory before the device can be programmed. The procedure for loading the Programming Executive to memory is described in Section 7.0 "Programming the Programming Executive to Memory".

6.11 Exiting ICSP Mode

Exiting Program/Verify mode is done by removing V_{IH} from MCLR, as shown in Figure 6-10. The only requirement for exit is that an interval, P16, should elapse between the last clock and program signals on PGC and PGD before removing V_{IH} .

FIGURE 6-10: EXITING ICSP™ MODE



dsPIC33EPXXXGM3XX/6XX/7XX

TABLE 6-9: 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 TBLPAG and the Read Pointer (W0) for TBLRD instruction.		
0000	200800	MOV #0x80, W0
0000	8802A0	MOV W0, TBLPAG
0000	207F00	MOV #0x7F0, 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
0000	000000	NOP
Step 3: Output the VISI register using the REGOUT command.		
0001	<VISI>	Clock out contents of the VISI register.

7.0 PROGRAMMING THE PROGRAMMING EXECUTIVE TO MEMORY

Storing the Programming Executive to executive memory is similar to normal programming of code memory. Namely, the executive memory must first be erased, and then the Programming Executive must be programmed 64 words at a time. This control flow is summarized in [Table 7-1](#).

7.1 Overview

If it is determined that the Programming Executive is not present in executive memory (as described in [Section 3.2 “Confirming the Presence of the Programming Executive”](#)), it must be programmed into executive memory using ICSP, as described in [Section 6.0 “Device Programming – ICSP”](#).

TABLE 7-1: PROGRAMMING THE PROGRAMMING EXECUTIVE

Command (Binary)	Data (Hex)	Description
Step 1: Exit the Reset vector and erase executive memory.		
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 NVMCON to erase a page of executive memory.		
0000	24003A	MOV #0x4003, W10
0000	88394A	MOV W10, NVMCON
0000	000000	NOP
0000	000000	NOP

Note 1: These three NOP instructions must be transmitted with a clock frequency greater than 2 MHz.

dsPIC33EPXXXGM3XX/6XX/7XX

TABLE 7-1: PROGRAMMING THE PROGRAMMING EXECUTIVE (CONTINUED)

Command (Binary)	Data (Hex)	Description
Step 3: Initiate the erase cycle, wait for erase to complete and make sure WR bit is clear.		
0000	200083	MOV #0x80, W3
0000	883963	MOV W3, NVMADRU
0000	200002	MOV #0x00, W2
0000	883952	MOV W2, NVMADR
0000	000000	NOP
0000	000000	NOP
0000	200551	MOV #0x55, W1
0000	883971	MOV W1, NVMKEY
0000	200AA1	MOV #0xAA, W1
0000	883971	MOV W1, NVMKEY
0000	A8E729	BSET NVMCON, #15
0000	000000	NOP
0000	000000	NOP
0000	000000	NOP
—	—	Externally time 'P12' msec (see Section 9.0 “AC/DC Characteristics and Timing Requirements”) to allow sufficient time for the Page Erase operation to complete.
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. Repeat until the WR bit is clear.
Step 4: Repeat Step 3 to erase the second page of executive memory (modifying NVMADR suitably).		
Step 5: Initialize the NVMCON to program 64 instruction words.		
0000	24002A	MOV #0x4002, W10
0000	88394A	MOV W10, NVMCON
Step 6: Initialize the NVMSRCADRH and NVMSRCADRL registers to point to the source of data in RAM and also the NVMADRU and NVMADR for each row in Flash.		
0000	200803	MOV #0x80, W3
0000	8802A0	MOV W3, NVMADRU
0000	2xxxx2	MOV #DestinationAddress<15:0>, W2
0000	883952	MOV W2, NVMADR
0000	2xxxx4	MOV #SourceAddress<15:0>, W4
0000	8xxxx4	MOV W4, NVMSRCADRL
0000	2xxxx5	MOV #SourceAddress<23:16>, W5
0000	8xxxx5	MOV W5, NVMSRCADRH
0000	000000	NOP
Step 7: Repeat Step 6 to suitably modify the NVMADR, NVMADRU, NVMSRCADR and NVMSRCADRH registers to point to the next row of data to be written.		

Note 1: These three NOP instructions must be transmitted with a clock frequency greater than 2 MHz.

TABLE 7-1: PROGRAMMING THE PROGRAMMING EXECUTIVE (CONTINUED)

Command (Binary)	Data (Hex)	Description
Step 8: Initiate the programming 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, #15
0000	000000	NOP ⁽¹⁾
0000	000000	NOP ⁽¹⁾
0000	000000	NOP ⁽¹⁾
0000	000000	NOP ⁽¹⁾
0000	000000	NOP ⁽¹⁾
Step 9: Wait for the Row Program operation to complete.		
—	—	Externally time 'P13' msec (see Section 9.0 "AC/DC Characteristics and Timing Requirements") to allow sufficient time for the Row Program operation to complete.
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 10: Repeat Steps 6-9 until all 32 rows of executive memory have been programmed.		

Note 1: These three NOP instructions must be transmitted with a clock frequency greater than 2 MHz.

7.2 Programming Verification

After the Programming Executive has been programmed to executive memory using ICSP, it must be verified. Verification is performed by reading out the contents of executive memory and comparing it with the image of the Programming Executive stored in the programmer.

Reading the contents of executive memory can be performed using the same technique described in [Section 6.7 “Reading Code Memory”](#). A procedure for reading executive memory is shown in [Table 7-2](#). Note that in Step 2, the TBLPAG register is set to 0x80, such that executive memory may be read.

TABLE 7-2: READING 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: Initialize TBLPAG and the Read Pointer (W6) for TBLRD instruction.		
0000	200800	MOV #0x80, W0
0000	8802A0	MOV W0, TBLPAG
0000	EB0300	CLR W6

TABLE 7-2: READING EXECUTIVE 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	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

dsPIC33EPXXXGM3XX/6XX/7XX

TABLE 7-2: READING EXECUTIVE 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 4-5 until all 2048 instruction words of executive memory are read.		

8.0 DEVICE ID

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

Table 8-1 lists the identification information for each device. Table 8-2 shows the Device ID registers. Register 8-1 provides the JTAG ID register format.

TABLE 8-1: DEVICE IDS AND REVISION

Device	DEVID Register Value	DEVREV Register Value	Silicon Revision
dsPIC33EP128GM304	0x1940	0x4000	A0
dsPIC33EP128GM604	0x1948		
dsPIC33EP128GM306	0x1943		
dsPIC33EP128GM706	0x194B		
dsPIC33EP128GM310	0x1947		
dsPIC33EP128GM710	0x194F		
dsPIC33EP256GM304	0x1A80	0x4000	A0
dsPIC33EP256GM604	0X1A88		
dsPIC33EP256GM306	0X1A83		
dsPIC33EP256GM706	0X1A8B		
dsPIC33EP256GM310	0X1A87		
dsPIC33EP256GM710	0X1A8F		
dsPIC33EP512GM304	0x1BC0	0x4000	A0
dsPIC33EP512GM604	0x1BC8		
dsPIC33EP512GM306	0x1BC3		
dsPIC33EP512GM706	0x1BCB		
dsPIC33EP512GM310	0x1BC7		
dsPIC33EP512GM710	0x1BCF		

TABLE 8-2: dsPIC33EPXXXGM3XX/6XX/7XX DEVICE ID REGISTERS

Address	Name	Bit															
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xFF0000	DEVID	DEVID Value															
0xFF0002	DEVREV	DEVREV Value															

REGISTER 8-1: JTAG ID REGISTER

31	28	27										12	11				0
DEVREV<3:0>				DEVID<15:0>										Manufacturer ID (0x053)			
4 bits				16 bits										12 bits			

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-85°C. Programming at 25°C is recommended.						
Param. No.	Symbol	Characteristic	Min.	Max.	Units	Conditions
D111	VDD	Supply Voltage During Programming	3.00	3.60	V	Normal programming ⁽¹⁾
D112	I _{PP}	Programming Current on MCLR	—	5	μA	
D113	I _{DDP}	Supply Current During Programming	—	10	mA	
D114	I _{PP}	Instantaneous Peak Current During Start-up	—	200	mA	
D031	V _{IL}	Input Low Voltage	V _{SS}	0.2 V _{DD}	V	
D041	V _{IH}	Input High Voltage	0.8 V _{DD}	V _{DD}	V	
D080	V _{OL}	Output Low Voltage	—	0.4	V	I _{OL} = 8 mA @ 3.3V
D090	V _{OH}	Output High Voltage	2.4	—	V	I _{OH} = 8 mA @ 3.3V
D012	C _{IO}	Capacitive Loading on I/O pin (PGD)	—	50	pF	To meet AC specifications
P1	T _{PGC}	Serial Clock (PGC) Period (ICSP™)	200	—	ns	
P1	T _{PGC}	Serial Clock (PGC) Period (Enhanced ICSP)	500	—	ns	
P1A	T _{PGCL}	Serial Clock (PGC) Low Time (ICSP)	80	—	ns	
P1A	T _{PGCL}	Serial Clock (PGC) Low Time (Enhanced ICSP)	200	—	ns	
P1B	T _{PGCH}	Serial Clock (PGC) High Time (ICSP)	80	—	ns	
P1B	T _{PGCH}	Serial Clock (PGC) High Time (Enhanced ICSP)	200	—	ns	
P2	T _{SET1}	Input Data Setup Time to PGC ↓	15	—	ns	
P3	T _{HLD1}	Input Data Hold Time from PGC ↓	15	—	ns	
P4	T _{DLY1}	Delay between 4-bit Command and Command Operand	40	—	ns	
P4A	T _{DLY1A}	Delay between Command Operand and Next 4-bit Command	40	—	ns	
P5	T _{DLY2}	Delay between Last PGC ↓ of Command to First PGC ↑ of Read of Data Word	20	—	ns	
P6	T _{SET2}	V _{DD} ↑ Setup Time to MCLR ↑	100	—	ns	
P7	T _{HLD2}	Input Data Hold Time from MCLR ↑	25	—	ms	
P8	T _{DLY3}	Delay between Last PGC ↓ of Command Byte to PGD ↑ by Programming Executive	12	—	μs	
P9a	T _{DLY4}	Programming Executive Command Processing Time	10	—	μs	
P9b	T _{DLY5}	Delay between PGD ↓ by Programming Executive to PGD Released by Programming Executive	15	23	μs	

Note 1: VDD must also be supplied to the AVDD pin 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.

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

Standard Operating Conditions Operating Temperature: -40°C-85°C. Programming at 25°C is recommended.						
Param. No.	Symbol	Characteristic	Min.	Max.	Units	Conditions
P10	TDLY6	PGC Low Time After Programming	400	—	ns	
P11a	TDLY7a	Bulk Erase Time (primary Flash memory)	50	70	ms	See Note 2
P12	TDLY8	Page Erase Time	17	23	ms	See Note 2
P13	TDLY9	Row Programming Time	1.2	1.6	ms	See Note 2
P14	TR	MCLR Rise Time to Enter ICSP™ mode	—	1.0	μs	
P15	TVALID	Data Out Valid from PGC ↑	10	—	ns	
P16	TDLY10	Delay between Last PGC ↓ and MCLR ↓	0	—	s	
P17	THLD3	MCLR ↓ to VDD ↓	100	—	ns	
P18	TKEY1	Delay from First MCLR ↓ to First PGC ↑ for Key Sequence on PGD	1	—	ms	
P19	TKEY2	Delay from Last PGC ↓ for Key Sequence on PGD to Second MCLR ↑	25	—	ns	
P21	TMCLRH	MCLR High Time	—	500	μs	

Note 1: VDD must also be supplied to the AVDD pin 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.

NOTES:

APPENDIX A: HEX FILE FORMAT

Flash programmers process the standard Hex format used by the Microchip development tools. The format supported is the Intel® HEX32 Format (INHX32). For more information about Hex file formats, refer to Appendix A in the “MPASM™ Assembler, MPLINK™ Object Linker, MPLIB™ Object Librarian Users Guide” (DS33014).

The basic format of the Hex file is:

```
:BBAAAATTHHHH...HHHHCC
```

Each data record begins with a 9-character prefix and always ends with a 2-character checksum. All records begin with ‘:’ regardless of the format. The individual elements are described below.

- **BB** – is a two-digit hexadecimal byte count representing the number of data bytes that appear on the line. Divide this number by two to get the number of words per line.
- **AAAA** – is a four-digit hexadecimal address representing the starting address of the data record. Format is high byte first, followed by low byte. The address is doubled because this format only supports 8 bits. Divide the value by two to find the real device address.
- **TT** – is a two-digit record type that will be ‘00’ for data records, ‘01’ for End-of-File (EOF) records and ‘04’ for extended address records.
- **HHHH** – is a four-digit hexadecimal data word. Format is low byte followed by high byte. There will be **BB/2** data words following **TT**.
- **CC** – is a two-digit hexadecimal checksum that is the two’s complement of the sum of all the preceding bytes in the line record.

Because the Intel Hex file format is byte-oriented, and the 16-bit program counter is not, program memory sections require special treatment. Each 24-bit program word is extended to 32 bits by inserting a so-called “phantom byte”. Each program memory address is multiplied by 2 to yield a byte address.

As an example, a section that is located at 0x100 in program memory will be represented in the Hex file as 0x200.

The Hex file will be produced with the following contents:

```
:020000040000fa
:040200003322110096
:00000001FF
```

Notice that the data record (line 2) has a load address of 0200, while the source code specifies address, 0x100. Also, note that the data is represented in “little-endian” format, meaning the Least Significant Byte (LSB) appears first. The phantom byte appears last, just before the checksum.

APPENDIX B: REVISION HISTORY

Revision A (August 2012)

This is the initial released version of 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.

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

Trademarks

The Microchip name and logo, the Microchip logo, dsPIC, KEELOQ, KEELOQ logo, MPLAB, PIC, PICmicro, PICSTART, PIC³² logo, rfPIC and UNI/O are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.


FilterLab, Hampshire, HI-TECH C, Linear Active Thermistor, MXDEV, MXLAB, SEEVAL and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Analog-for-the-Digital Age, Application Maestro, BodyCom, chipKIT, chipKIT logo, CodeGuard, dsPICDEM, dsPICDEM.net, dsPICworks, dsSPEAK, ECAN, ECONOMONITOR, FanSense, HI-TIDE, In-Circuit Serial Programming, ICSP, Mindi, MiWi, MPASM, MPLAB Certified logo, MPLIB, MPLINK, mTouch, Omniscent Code Generation, PICC, PICC-18, PICDEM, PICDEM.net, PICKIT, PICtail, REAL ICE, rLAB, Select Mode, Total Endurance, TSHARC, UniWinDriver, WiperLock and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

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

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

© 2012, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

 Printed on recycled paper.

ISBN:978-1-62076-499-2

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

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

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

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

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

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

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

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

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

Toronto
Mississauga, Ontario,
Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

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

China - Beijing
Tel: 86-10-8569-7000
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: 86-571-2819-3187
Fax: 86-571-2819-3189

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

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

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

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

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

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

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

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

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

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

ASIA/PACIFIC

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

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

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

Japan - Osaka
Tel: 81-66-152-7160
Fax: 81-66-152-9310

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

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

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

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

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

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

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

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

Taiwan - Kaohsiung
Tel: 886-7-536-4818
Fax: 886-7-330-9305

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

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

EUROPE

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

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

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

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

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

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

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

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