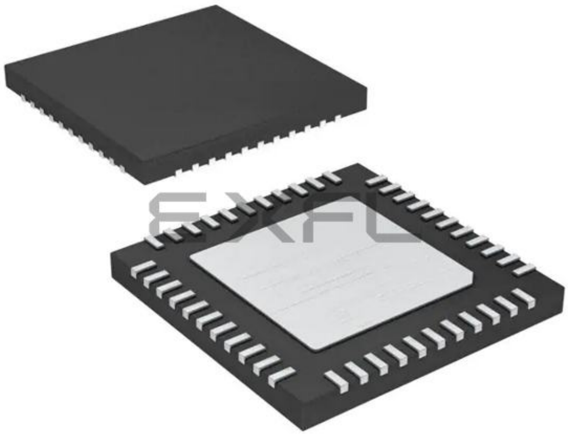


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	dsPIC
Core Size	16-Bit
Speed	20 MIPS
Connectivity	CANbus, I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, Motor Control PWM, QEI, POR, PWM, WDT
Number of I/O	20
Program Memory Size	48KB (16K x 24)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	2.5V ~ 5.5V
Data Converters	A/D 6x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-VQFN Exposed Pad
Supplier Device Package	44-QFN (8x8)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/dspic30f4012t-20i-ml

3.0 PROGRAMMING EXECUTIVE APPLICATION

3.1 Programming Executive Overview

The programming executive resides in executive 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 dsPIC30F, using a simple command set and communication protocol.

The following capabilities are provided by the programming executive:

- Read memory
  - Code memory and data EEPROM
  - Configuration registers
  - Device ID
- Erase memory
  - Bulk Erase by segment
  - Code memory (by row)
  - Data EEPROM (by row)
- Program memory
  - Code memory
  - Data EEPROM
  - Configuration registers
- Query
  - Blank Device
  - Programming executive software version

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

The programming procedure is outlined in **Section 5.0 “Device Programming”**.

3.2 Programming Executive Code Memory

The programming executive is stored in executive code memory and executes from this reserved region of memory. It requires no resources from user code memory or data EEPROM.

3.3 Programming Executive Data RAM

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

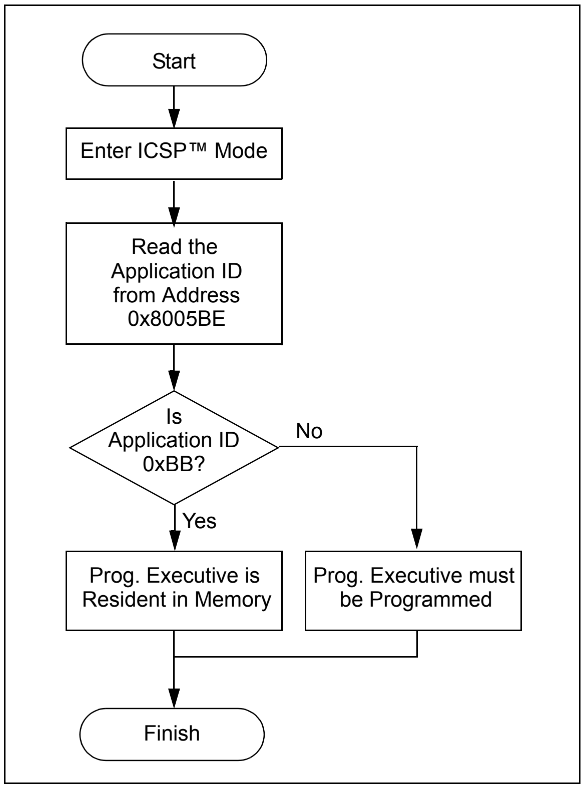
4.0 CONFIRMING THE CONTENTS OF EXECUTIVE MEMORY

Before programming can begin, the programmer must confirm that the programming executive is stored in executive memory. The procedure for this task is illustrated in Figure 4-1.

First, ICSP mode is entered. The unique application ID word stored in executive memory is then read. If the programming executive is resident, the application ID word is 0xBB, which means programming can resume as normal. However, if the application ID word is not 0xBB, the programming executive must be programmed to Executive Code memory using the method described in **Section 12.0 “Programming the Programming Executive to Memory”**.

Section 11.0 “ICSP™ Mode” describes the process for the ICSP programming method. **Section 11.13 “Reading the Application ID Word”** describes the procedure for reading the application ID word in ICSP mode.

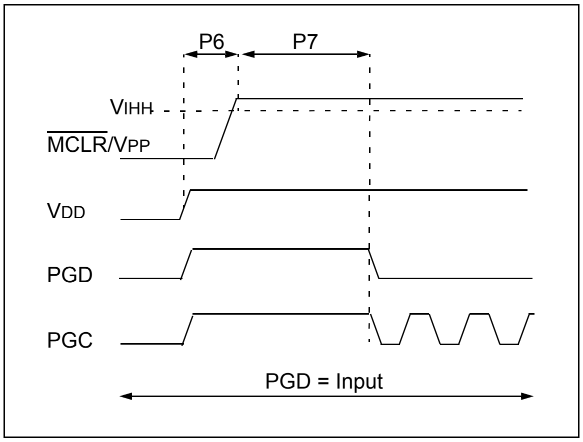
FIGURE 4-1: CONFIRMING PRESENCE OF THE PROGRAMMING EXECUTIVE



5.2 Entering Enhanced ICSP Mode

The Enhanced ICSP mode is entered by holding PGC and PGD high, and then raising MCLR/VPP to VIHH (high voltage), as illustrated in Figure 5-2. In this mode, the code memory, data EEPROM and Configuration bits can be efficiently programmed using the programming executive commands that are serially transferred using PGC and PGD.

FIGURE 5-2: ENTERING ENHANCED ICSP™ MODE



> **Note 1:** The sequence that places the device into Enhanced ICSP mode places all unused I/Os in the high-impedance state.
>
> **2:** Before entering Enhanced ICSP mode, clock switching must be disabled using ICSP, by programming the FCKSM<1:0> bits in the FOSC Configuration register to '11' or '10'.
>
> **3:** When in Enhanced ICSP mode, the SPI output pin (SDO1) will toggle while the device is being programmed.

5.3 Chip Erase

Before a chip can be programmed, it must be erased. The Bulk Erase command (ERASEB) is used to perform this task. Executing this command with the MS command field set to 0x3 erases all code memory, data EEPROM and code-protect Configuration bits. The Chip Erase process sets all bits in these three memory regions to '1'.

Since non-code-protect Configuration bits cannot be erased, they must be manually set to '1' using multiple PROGC commands. One PROGC command must be sent for each Configuration register (see **Section 5.7 "Configuration Bits Programming"**).

If Advanced Security features are enabled, then individual Segment Erase operations would need to be performed, depending on which segment needs to be programmed at a given stage of system programming. The user should have the flexibility to select specific segments for programming.

> **Note:** The Device ID registers cannot be erased. These registers remain intact after a Chip Erase is performed.

5.4 Blank Check

The term "Blank Check" means to verify that the device has been successfully erased and has no programmed memory cells. A blank or erased memory cell reads as '1'. The following memories must be blank checked:

- All implemented code memory
- All implemented data EEPROM
- All Configuration bits (for their default value)

The Device ID registers (0xFF0000:0xFF0002) can be ignored by the Blank Check since this region stores device information that cannot be erased. 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 and data EEPROM are erased by testing these memory regions. A 'BLANK' or 'NOT BLANK' response is returned. The READD command is used to read the Configuration registers. If it is determined that the device is not blank, it must be erased (see **Section 5.3 "Chip Erase"**) before attempting to program the chip.

5.5 Code Memory Programming

5.5.1 OVERVIEW

The Flash code memory array consists of 512 rows of thirty-two, 24-bit instructions. Each panel stores 16K instruction words, and each dsPIC30F device has either 1, 2 or 3 memory panels (see Table 5-2).

TABLE 5-2: DEVICE CODE MEMORY SIZE

Device	Code Size (24-bit Words)	Number of Rows	Number of Panels
dsPIC30F2010	4K	128	1
dsPIC30F2011	4K	128	1
dsPIC30F2012	4K	128	1
dsPIC30F3010	8K	256	1
dsPIC30F3011	8K	256	1
dsPIC30F3012	8K	256	1
dsPIC30F3013	8K	256	1
dsPIC30F3014	8K	256	1
dsPIC30F4011	16K	512	1
dsPIC30F4012	16K	512	1
dsPIC30F4013	16K	512	1
dsPIC30F5011	22K	704	2
dsPIC30F5013	22K	704	2
dsPIC30F5015	22K	704	2
dsPIC30F5016	22K	704	2
dsPIC30F6010	48K	1536	3
dsPIC30F6010A	48K	1536	3
dsPIC30F6011	44K	1408	3
dsPIC30F6011A	44K	1408	3
dsPIC30F6012	48K	1536	3
dsPIC30F6012A	48K	1536	3
dsPIC30F6013	44K	1408	3
dsPIC30F6013A	44K	1408	3
dsPIC30F6014	48K	1536	3
dsPIC30F6014A	48K	1536	3
dsPIC30F6015	48K	1536	3

5.5.2 PROGRAMMING METHODOLOGY

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

A flowchart for programming of code memory is illustrated in Figure 5-3. In this example, all 48K instruction words of a dsPIC30F6014A device are programmed. First, the number of commands to send (called 'RemainingCmds' in the flowchart) is set to 1536 and the destination address (called 'BaseAddress') is set to '0'.

Next, one row in the device is programmed with a PROGP command. Each PROGP command contains data for one row of code memory of the dsPIC30F6014A. After the first command is processed successfully, 'RemainingCmds' is decremented by 1 and compared to 0. Since there are more PROGP commands to send, 'BaseAddress' is incremented by 0x40 to point to the next row of memory.

On the second PROGP command, the second row of each memory panel is programmed. This process is repeated until the entire device is programmed. No special handling must be performed when a panel boundary is crossed.

FIGURE 5-3: FLOWCHART FOR PROGRAMMING dsPIC30F6014A CODE MEMORY

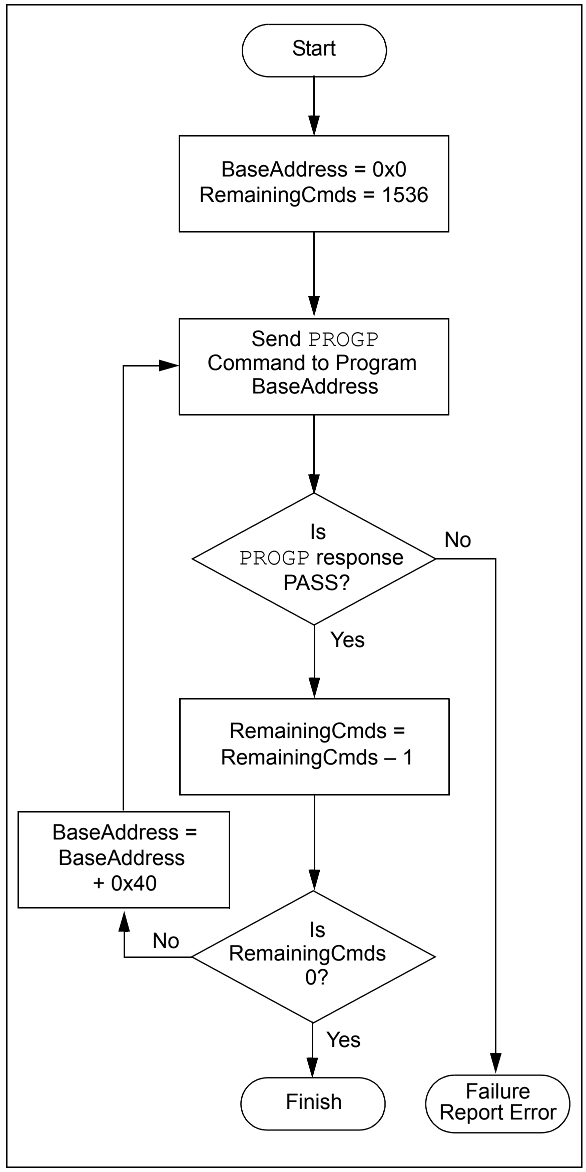


TABLE 5-5: FOSC CONFIGURATION BITS DESCRIPTION FOR dsPIC30F4011/4012 AND dsPIC30F5011/5013

Bit Field	Register	Description
FCKSM<1:0>	FOSC	**Clock Switching Mode** 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
FOS<1:0>	FOSC	**Oscillator Source Selection on POR** 11 = Primary Oscillator 10 = Internal Low-Power RC Oscillator 01 = Internal Fast RC Oscillator 00 = Low-Power 32 kHz Oscillator (Timer1 Oscillator)
FPR<3:0>	FOSC	**Primary Oscillator Mode** 1111 = ECIO w/PLL 16X – External Clock mode with 16X PLL. OSC2 pin is I/O 1110 = ECIO w/PLL 8X – External Clock mode with 8X PLL. OSC2 pin is I/O 1101 = ECIO w/PLL 4X – External Clock mode with 4X PLL. OSC2 pin is I/O 1100 = ECIO – External Clock mode. OSC2 pin is I/O 1011 = EC – External Clock mode. OSC2 pin is system clock output (FOSC/4) 1010 = FRC w/PLL 8x – Internal fast RC oscillator with 8x PLL. OSC2 pin is I/O 1001 = ERC – External RC Oscillator mode. OSC2 pin is system clock output (FOSC/4) 1000 = ERCIO – External RC Oscillator mode. OSC2 pin is I/O 0111 = XT w/PLL 16X – XT Crystal Oscillator mode with 16X PLL 0110 = XT w/PLL 8X – XT Crystal Oscillator mode with 8X PLL 0101 = XT w/PLL 4X – XT Crystal Oscillator mode with 4X PLL 0100 = XT – XT Crystal Oscillator mode (4 MHz-10 MHz crystal) 0011 = FRC w/PLL 16x – Internal fast RC oscillator with 16x PLL. OSC2 pin is I/O 0010 = HS – HS Crystal Oscillator mode (10 MHz-25 MHz crystal) 0001 = FRC w/PLL 4x – Internal fast RC oscillator with 4x PLL. OSC2 pin is I/O 0000 = XTL – XTL Crystal Oscillator mode (200 kHz-4 MHz crystal)

TABLE 5-7: CONFIGURATION BITS DESCRIPTION (CONTINUED)

Bit Field	Register	Description
SSS<2:0>	FSS	**Secure Segment Program Memory Code Protection (only present in dsPIC30F5011/5013/6010A/6011A/6012A/6013A/6014A/6015)** `111` = No Secure Segment `110` = Standard security; Small-sized Secure Program Flash [Secure Segment starts after BS and ends at 0x001FFF] `101` = Standard security; Medium-sized Secure Program Flash [Secure Segment starts after BS and ends at 0x003FFF] `100` = Standard security; Large-sized Secure Program Flash [Secure Segment starts after BS and ends at 0x007FFF] `011` = No Secure Segment `010` = High security; Small-sized Secure Program Flash [Secure Segment starts after BS and ends at 0x001FFF] `001` = High security; Medium-sized Secure Program Flash [Secure Segment starts after BS and ends at 0x003FFF] `000` = High security; Large-sized Secure Program Flash [Secure Segment starts after BS and ends at 0x007FFF]
SWRP	FSS	**Secure Segment Program Memory Write Protection (only present in dsPIC30F5011/5013/6010A/6011A/6012A/6013A/6014A/6015)** `1` = Secure Segment program memory is not write-protected `0` = Secure program memory is write-protected
GSS<1:0>	FGS	**General Segment Program Memory Code Protection (only present in dsPIC30F5011/5013/6010A/6011A/6012A/6013A/6014A/6015)** `11` = Code protection is disabled `10` = Standard security code protection is enabled `0x` = High security code protection is enabled
GCP	FGS	**General Segment Program Memory Code Protection (present in all devices except dsPIC30F5011/5013/6010A/6011A/6012A/6013A/6014A/6015)** `1` = General Segment program memory is not code-protected `0` = General Segment program memory is code-protected
GWRP	FGS	**General Segment Program Memory Write Protection** `1` = General Segment program memory is not write-protected `0` = General Segment program memory is write-protected
BKBUG	FICD	**Debugger/Emulator Enable** `1` = Device will reset into Operational mode `0` = Device will reset into Debug/Emulation mode
COE	FICD	**Debugger/Emulator Enable** `1` = Device will reset into Operational mode `0` = Device will reset into Clip-on Emulation mode
ICS<1:0>	FICD	**ICD Communication Channel Select** `11` = Communicate on PGC/EMUC and PGD/EMUD `10` = Communicate on EMUC1 and EMUD1 `01` = Communicate on EMUC2 and EMUD2 `00` = Communicate on EMUC3 and EMUD3
RESERVED	FBS, FSS, FGS	**Reserved (read as ‘`1`’, write as ‘`1`’)**
—	All	**Unimplemented (read as ‘`0`’, write as ‘`0`’)**

TABLE 5-10: dsPIC30F CONFIGURATION REGISTERS (FOR dsPIC30F2011/2012, dsPIC30F3010/3011/3012/3013/3014, dsPIC30F4013 AND dsPIC30F5015/5016)

Address	Name	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0xF80000	FOSC	FCKSM<1:0>		—	—	—	FOS<2:0>			—	—	—	FPR<4:0>				
0xF80002	FWDT	FWDTEN	—	—	—	—	—	—	—	—	—	FWPSA<1:0>		FWPSB<3:0>			
0xF80004	FBORPOR	MCLREN	—	—	—	—	PWMPIN[1]	HPOL[1]	LPOL[1]	BOREN	—	BORV<1:0>		—	—	FPWRT<1:0>	
0xF80006	FBS	—	—	Reserved[2]		—	—	—	Reserved[2]	—	—	—	—	Reserved[2]			
0xF80008	FSS	—	—	Reserved[2]		—	—	Reserved[2]		—	—	—	—	Reserved[2]			
0xF8000A	FGS	—	—	—	—	—	—	—	—	—	—	—	—	—	Reserved[3]	GCP	GWRP
0xF8000C	FICD	BKBUG	COE	—	—	—	—	—	—	—	—	—	—	—	—	ICS<1:0>	

Note 1: On the 2011, 2012, 3012, 3013, 3014 and 4013, these bits are reserved (read as '1' and must be programmed as '1').
2: Reserved bits read as '1' and must be programmed as '1'.
3: The FGS<2> bit is a read-only copy of the GCP bit (FGS<1>).

TABLE 5-11: dsPIC30F CONFIGURATION REGISTERS (FOR dsPIC30F6010A/6011A/6012A/6013A/6014A AND dsPIC30F6015)

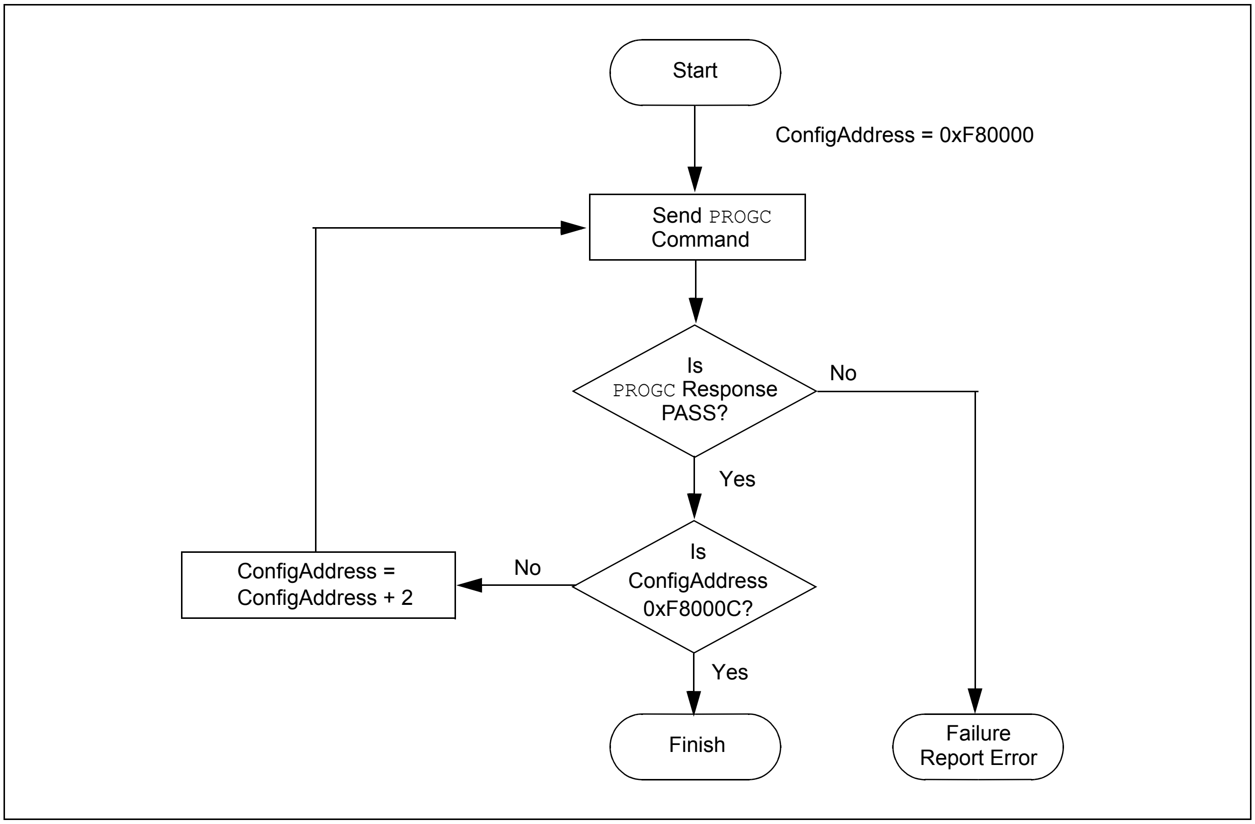
Address	Name	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0xF80000	FOSC	FCKSM<1:0>		—	—	—	FOS<2:0>			—	—	—	FPR<4:0>				
0xF80002	FWDT	FWDTEN	—	—	—	—	—	—	—	—	—	FWPSA<1:0>		FWPSB<3:0>			
0xF80004	FBORPOR	MCLREN	—	—	—	—	PWMPIN[1]	HPOL[1]	LPOL[1]	BOREN	—	BORV<1:0>		—	—	FPWRT<1:0>	
0xF80006	FBS	—	—	RBS<1:0>		—	—	—	EBS	—	—	—	—	BSS<2:0>			BWRP
0xF80008	FSS	—	—	RSS<1:0>		—	—	ESS<1:0>		—	—	—	—	SSS<2:0>			SWRP
0xF8000A	FGS	—	—	—	—	—	—	—	—	—	—	—	—	—	GSS<1:0>		GWRP
0xF8000C	FICD	BKBUG	COE	—	—	—	—	—	—	—	—	—	—	—	—	ICS<1:0>	

Note 1: On the 6011A, 6012A, 6013A and 6014A, these bits are reserved (read as '1' and must be programmed as '1').

5.8 Exiting Enhanced ICSP Mode

The Enhanced ICSP mode is exited by removing power from the device or bringing $\overline{MCLR}$ to V_{IL}. When normal user mode is next entered, the program that was stored using Enhanced ICSP will execute.

FIGURE 5-5: CONFIGURATION BIT PROGRAMMING FLOW



8.0 PROGRAMMING EXECUTIVE COMMANDS

8.1 Command Set

The programming executive command set is shown in Table 8-1. This table contains the opcode, mnemonic, length, time out and description for each command. Functional details on each command are provided in the command descriptions (see **Section 8.5 "Command Descriptions"**).

8.2 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 8-1). 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 8-1: COMMAND FORMAT

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

The command opcode must match one of those in the command set. Any command that is received which does not match the list in Table 8-1 will return a "NACK" response (see **Section 9.2.1 "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.

8.3 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 8-2. This format minimizes traffic over the SPI and provides the programming executive with data that is properly aligned for performing table write operations.

FIGURE 8-2: PACKED INSTRUCTION WORD FORMAT

15 — 8	7 — 0
lsw1	
MSB2	MSB1
lsw2	

lswx: Least significant 16 bits of instruction word
MSBx: Most Significant Byte of instruction word

> **Note:** When the number of instruction words transferred is odd, MSB2 is zero and lsw2 cannot be transmitted.

8.4 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 9.2.3 "QE_Code Field"**.

8.5.7 ERASEB COMMAND

15–12	11–2	0
Opcode	Length	
Reserved		MS

Field	Description
Opcode	0x7
Length	0x2
Reserved	0x0
MS	Select memory to erase: 0x0 = All Code in General Segment 0x1 = All Data EEPROM in General Segment 0x2 = All Code and Data EEPROM in General Segment, interrupt vectors and FGS Configuration register 0x3 = Full Chip Erase 0x4 = All Code and Data EEPROM in Boot, Secure and General Segments, and FBS, FSS and FGS Configuration registers 0x5 = All Code and Data EEPROM in Secure and General Segments, and FSS and FGS Configuration registers 0x6 = All Data EEPROM in Boot Segment 0x7 = All Data EEPROM in Secure Segment

The ERASEB command performs a Bulk Erase. The MS field selects the memory to be bulk erased, with options for erasing Code and/or Data EEPROM in individual memory segments.

When Full Chip Erase is selected, the following memory regions are erased:

- All code memory (even if code-protected)
- All data EEPROM
- All code-protect Configuration registers

Only the executive code memory, Unit ID, device ID and Configuration registers that are not code-protected remain intact after a Chip Erase.

Expected Response (2 words):
0x1700
0x0002

> **Note:** A Full Chip Erase cannot be performed in low-voltage programming systems (VDD less than 4.5 volts). ERASED and ERASEP must be used to erase code memory, executive memory and data memory. Alternatively, individual Segment Erase operations may be performed.

8.5.8 ERASED COMMAND

15–12	11–8	7–0
Opcode	Length	
Num_Rows		Addr_MSB
Addr_LS		

Field	Description
Opcode	0x8
Length	0x3
Num_Rows	Number of rows to erase (max of 128)
Addr_MSB	MSB of 24-bit base address
Addr_LS	LS 16 bits of 24-bit base address

The ERASED command erases the specified number of rows of data EEPROM from the specified base address. The specified base address must be a multiple of 0x20. Since the data EEPROM is mapped to program space, a 24-bit base address must be specified.

After the erase is performed, all targeted bytes of data EEPROM will contain 0xFF.

Expected Response (2 words):
0x1800
0x0002

> **Note:** The ERASED command cannot be used to erase the Configuration registers or device ID. Code-protect Configuration registers can only be erased with the ERASEB command, while the device ID is read-only.

10.0 DEVICE ID

The device ID region is 2 x 16 bits and can be read using the READD command. This region of memory is read-only and can also be read when code protection is enabled.

Table 10-1 shows the device ID for each device, Table 10-2 shows the device ID registers and Table 10-3 describes the bit field of each register.

TABLE 10-1: DEVICE IDS

Device	DEVID	Silicon Revision							
		A0	A1	A2	A3	A4	B0	B1	B2
dsPIC30F2010	0x0040	0x1000	0x1001	0x1002	0x1003	0x1004	—	—	—
dsPIC30F2011	0x0240	—	0x1001	—	—	—	—	—	—
dsPIC30F2012	0x0241	—	0x1001	—	—	—	—	—	—
dsPIC30F3010	0x01C0	0x1000	0x1001	0x1002	—	—	—	—	—
dsPIC30F3011	0x01C1	0x1000	0x1001	0x1002	—	—	—	—	—
dsPIC30F3012	0x00C1	—	—	—	—	—	0x1040	0x1041	—
dsPIC30F3013	0x00C3	—	—	—	—	—	0x1040	0x1041	—
dsPIC30F3014	0x0160	—	0x1001	0x1002	—	—	—	—	—
dsPIC30F4011	0x0101	—	0x1001	0x1002	0x1003	0x1003	—	—	—
dsPIC30F4012	0x0100	—	0x1001	0x1002	0x1003	0x1003	—	—	—
dsPIC30F4013	0x0141	—	0x1001	0x1002	—	—	—	—	—
dsPIC30F5011	0x0080	—	0x1001	0x1002	0x1003	0x1003	—	—	—
dsPIC30F5013	0x0081	—	0x1001	0x1002	0x1003	0x1003	—	—	—
dsPIC30F5015	0x0200	0x1000	—	—	—	—	—	—	—
dsPIC30F5016	0x0201	0x1000	—	—	—	—	—	—	—
dsPIC30F6010	0x0188	—	—	—	—	—	—	0x1040	0x1042
dsPIC30F6010A	0x0281	—	—	0x1002	0x1003	0x1004	—	—	—
dsPIC30F6011	0x0192	—	—	—	0x1003	—	—	0x1040	0x1042
dsPIC30F6011A	0x02C0	—	—	0x1002	—	—	0x1040	0x1041	—
dsPIC30F6012	0x0193	—	—	—	0x1003	—	—	0x1040	0x1042
dsPIC30F6012A	0x02C2	—	—	0x1002	—	—	0x1040	0x1041	—
dsPIC30F6013	0x0197	—	—	—	0x1003	—	—	0x1040	0x1042
dsPIC30F6013A	0x02C1	—	—	0x1002	—	—	0x1040	0x1041	—
dsPIC30F6014	0x0198	—	—	—	0x1003	—	—	0x1040	0x1042
dsPIC30F6014A	0x02C3	—	—	0x1002	—	—	0x1040	0x1041	—
dsPIC30F6015	0x0280	—	—	0x1002	0x1003	0x1004	—	—	—

TABLE 10-2: dsPIC30F 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<15:0>															
0xFF0002	DEVREV	PROC<3:0>				REV<5:0>						DOT<5:0>					

11.4 Flash Memory Programming in ICSP Mode

Programming in ICSP mode is described in **Section 11.4.1 “Programming Operations”** through **Section 11.4.3 “Starting and Stopping a Programming Cycle”**. Step-by-step procedures are described in **Section 11.5 “Erasing Program Memory in Normal-Voltage Systems”** through **Section 11.13 “Reading the Application ID Word”**. All programming operations must use serial execution, as described in **Section 11.2 “ICSP Operation”**.

11.4.1 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 11-2) or write operation (Table 11-3), writing a key sequence to enable the programming and initiating the programming by setting the WR control bit, NVMCON<15>.

In ICSP mode, all programming operations are externally timed. An external 2 ms delay must be used between setting the WR control bit and clearing the WR control bit to complete the programming operation.

TABLE 11-2: NVMCON ERASE OPERATIONS

NVMCON Value	Erase Operation
0x407F	Erase all code memory, data memory (does not erase UNIT ID).
0x4075	Erase 1 row (16 words) of data EEPROM.
0x4074	Erase 1 word of data EEPROM.
0x4072	Erase all executive memory.
0x4071	Erase 1 row (32 instruction words) from 1 panel of code memory.
0x406E	Erase Boot Secure and General Segments, then erase FBS, FSS and FGS configuration registers.
0x4066	Erase all Data EEPROM allocated to Boot Segment.
0x405E	Erase Secure and General Segments, then erase FSS and FGS configuration registers.
0x4056	Erase all Data EEPROM allocated to Secure Segment.
0x404E	Erase General Segment, then erase FGS configuration register.
0x4046	Erase all Data EEPROM allocated to General Segment.

TABLE 11-3: NVMCON WRITE OPERATIONS

NVMCON Value	Write Operation
0x4008	Write 1 word to configuration memory.
0x4005	Write 1 row (16 words) to data memory.
0x4004	Write 1 word to data memory.
0x4001	Write 1 row (32 instruction words) into 1 panel of program memory.

11.4.2 UNLOCKING NVMCON FOR PROGRAMMING

Writes to the WR bit (NVMCON<15>) are locked to prevent accidental programming from taking place. Writing a key sequence to the NVMKEY register unlocks the WR bit and allows it to be written to. The unlock sequence is performed as follows:

```
MOV    #0x55, W8
MOV    W8, NVMKEY
MOV    #0xAA, W9
MOV    W9, NVMKEY
```

Note:	Any working register, or working register pair, can be used to write the unlock sequence.

11.4.3 STARTING AND STOPPING A PROGRAMMING CYCLE

Once the unlock key sequence has been written to the NVMKEY register, the WR bit (NVMCON<15>) is used to start and stop an erase or write cycle. Setting the WR bit initiates the programming cycle. Clearing the WR bit terminates the programming cycle.

All erase and write cycles must be externally timed. An external delay must be used between setting and clearing the WR bit. Starting and stopping a programming cycle is performed as follows:

```
BSET   NVMCON, #WR
<Wait 2 ms>
BCLR   NVMCON, #WR
```

11.5 Erasing Program Memory in Normal-Voltage Systems

The procedure for erasing program memory (all code memory, data memory, executive memory and code-protect bits) consists of setting NVMCON to 0x407F, unlocking NVMCON for erasing and then executing the programming cycle. This method of bulk erasing program memory only works for systems where VDD is between 4.5 volts and 5.5 volts. The method for erasing program memory for systems with a lower VDD (3.0 volts-4.5 volts) is described in **Section 6.1 “Erasing Memory”**.

11.8 Writing Code Memory

The procedure for writing code memory is similar to the procedure for clearing the Configuration registers, except that 32 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 11-8 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 11-2). In Step 1, the Reset vector is exited. In Step 2, the NVMCON register is initialized for single-panel programming of code memory. In Step 3, the 24-bit starting destination address for programming is loaded into the TBLPAG register and W7 register. The upper byte of the starting destination address is stored to TBLPAG, while the lower 16 bits of the destination address are stored to W7.

To minimize the programming time, the same packed instruction format that the programming executive uses is utilized (Figure 8-2). In Step 4, four packed instruction words are stored to working registers W0:W5 using the `MOV` instruction and the read pointer W6 is initialized. The contents of W0:W5 holding the packed instruction word data is shown in Figure 11-4.

In Step 5, eight `TBLWT` instructions are used to copy the data from W0:W5 to the write latches of code memory. Since code memory is programmed 32 instruction words at a time, Steps 4 and 5 are repeated eight times to load all the write latches (Step 6).

After the write latches are loaded, programming is initiated by writing to the NVMKEY and NVMCON registers in Steps 7 and 8. In Step 9, the internal PC is reset to 0x100. This is a precautionary measure to prevent the PC from incrementing into unimplemented memory when large devices are being programmed. Lastly, in Step 10, Steps 2-9 are repeated until all of code memory is programmed.

FIGURE 11-5: PACKED INSTRUCTION WORDS IN W0:W5

	15 … 8	7 … 0
W0	lsw0	
W1	MSB1	MSB0
W2	lsw1	
W3	lsw2	
W4	MSB3	MSB2
W5	lsw3	

TABLE 11-8: SERIAL INSTRUCTION EXECUTION FOR WRITING CODE MEMORY

Command (Binary)	Data (Hexadecimal)	Description
Step 1: Exit the Reset vector.		
0000	040100	`GOTO 0x100`
0000	040100	`GOTO 0x100`
0000	000000	`NOP`
Step 2: Set the NVMCON to program 32 instruction words.		
0000	24001A	`MOV #0x4001, W10`
0000	883B0A	`MOV W10, NVMCON`
Step 3: Initialize the write pointer (W7) for `TBLWT` instruction.		
0000	200xx0	`MOV #<DestinationAddress23:16>, W0`
0000	880190	`MOV W0, TBLPAG`
0000	2xxxx7	`MOV #<DestinationAddress15:0>, W7`
Step 4: Initialize the read pointer (W6) and load W0:W5 with the next 4 instruction words to program.		
0000	2xxxx0	`MOV #<LSW0>, W0`
0000	2xxxx1	`MOV #<MSB1:MSB0>, W1`
0000	2xxxx2	`MOV #<LSW1>, W2`
0000	2xxxx3	`MOV #<LSW2>, W3`
0000	2xxxx4	`MOV #<MSB3:MSB2>, W4`
0000	2xxxx5	`MOV #<LSW3>, W5`

TABLE 11-8: SERIAL INSTRUCTION EXECUTION FOR WRITING CODE MEMORY (CONTINUED)

Command (Binary)	Data (Hexadecimal)	Description
Step 5: Set the read pointer (W6) and load the (next set of) write latches.		
0000	EB0300	CLR W6
0000	000000	NOP
0000	BB0BB6	TBLWTL [W6++], [W7]
0000	000000	NOP
0000	000000	NOP
0000	BBDBB6	TBLWTH.B [W6++], [W7++]
0000	000000	NOP
0000	000000	NOP
0000	BBEBB6	TBLWTH.B [W6++], [++W7]
0000	000000	NOP
0000	000000	NOP
0000	BB1BB6	TBLWTL [W6++], [W7++]
0000	000000	NOP
0000	000000	NOP
0000	BB0BB6	TBLWTL [W6++], [W7]
0000	000000	NOP
0000	000000	NOP
0000	BBDBB6	TBLWTH.B [W6++], [W7++]
0000	000000	NOP
0000	000000	NOP
0000	BBEBB6	TBLWTH.B [W6++], [++W7]
0000	000000	NOP
0000	000000	NOP
0000	BB1BB6	TBLWTL [W6++], [W7++]
0000	000000	NOP
0000	000000	NOP
Step 6: Repeat steps 4-5 eight times to load the write latches for 32 instructions.		
Step 7: Unlock the NVMCON for writing.		
0000	200558	MOV #0x55, W8
0000	883B38	MOV W8, NVMKEY
0000	200AA9	MOV #0xAA, W9
0000	883B39	MOV W9, NVMKEY
Step 8: Initiate the write cycle.		
0000	A8E761	BSET NVMCON, #WR
0000	000000	NOP
0000	000000	NOP
—	—	Externally time 'P12a' ms (see **Section 13.0 "AC/DC Characteristics and Timing Requirements"**)
0000	000000	NOP
0000	000000	NOP
0000	A9E761	BCLR NVMCON, #WR
0000	000000	NOP
0000	000000	NOP
Step 9: Reset device internal PC.		
0000	040100	GOTO 0x100
0000	000000	NOP
Step 10: Repeat steps 2-9 until all code memory is programmed.		

TABLE 11-10: SERIAL INSTRUCTION EXECUTION FOR READING CODE MEMORY (CONTINUED)

Command (Binary)	Data (Hexadecimal)	Description
Step 4: Output W0:W5 using the VISI register and REGOUT command.		
0000	883C20	MOV W0, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register
0000	000000	NOP
0000	883C21	MOV W1, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register
0000	000000	NOP
0000	883C22	MOV W2, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register
0000	000000	NOP
0000	883C23	MOV W3, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register
0000	000000	NOP
0000	883C24	MOV W4, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register
0000	000000	NOP
0000	883C25	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	040100	GOTO 0x100
0000	000000	NOP
Step 6: Repeat steps 3-5 until all desired code memory is read.		

12.0 PROGRAMMING THE PROGRAMMING EXECUTIVE TO MEMORY

12.1 Overview

If it is determined that the programming executive does not reside in executive memory (as described in **Section 4.0 "Confirming the Contents of Executive Memory"**), it must be programmed into executive memory using ICSP and the techniques described in **Section 11.0 "ICSP™ Mode"**.

Storing the programming executive to executive memory is similar to normal programming of code memory. The executive memory must first be erased, and then the programming executive must be programmed 32 words at a time. This control flow is summarized in Table 12-1.

TABLE 12-1: PROGRAMMING THE PROGRAMMING EXECUTIVE

Command (Binary)	Data (Hexadecimal)	Description
Step 1: Exit the Reset vector and erase executive memory.		
0000	040100	GOTO 0x100
0000	040100	GOTO 0x100
0000	000000	NOP
Step 2: Initialize the NVMCON to erase executive memory.		
0000	24072A	MOV #0x4072, W10
0000	883B0A	MOV W10, NVMCON
Step 3: Unlock the NVMCON for programming.		
0000	200558	MOV #0x55, W8
0000	883B38	MOV W8, NVMKEY
0000	200AA9	MOV #0xAA, W9
0000	883B39	MOV W9, NVMKEY
Step 4: Initiate the erase cycle.		
0000	A8E761	BSET NVMCON, #15
0000	000000	NOP
0000	000000	NOP
—	—	Externally time 'P13a' ms (see **Section 13.0 "AC/DC Characteristics and Timing Requirements"**)
0000	000000	NOP
0000	000000	NOP
0000	A9E761	BCLR NVMCON, #15
0000	000000	NOP
0000	000000	NOP
Step 5: Initialize the TBLPAG and the write pointer (W7).		
0000	200800	MOV #0x80, W0
0000	880190	MOV W0, TBLPAG
0000	EB0380	CLR W7
0000	000000	NOP
0000	000000	NOP
Step 6: Initialize the NVMCON to program 32 instruction words.		
0000	24001A	MOV #0x4001, W10
0000	883B0A	MOV W10, NVMCON
Step 7: Load W0:W5 with the next 4 words of packed programming executive code and initialize W6 for programming. Programming starts from the base of executive memory (0x800000) using W6 as a read pointer and W7 as a write pointer.		
0000	2<LSW0>0	MOV #<LSW0>, W0
0000	2<MSB1:MSB0>1	MOV #<MSB1:MSB0>, W1
0000	2<LSW1>2	MOV #<LSW1>, W2
0000	2<LSW2>3	MOV #<LSW2>, W3
0000	2<MSB3:MSB2>4	MOV #<MSB3:MSB2>, W4
0000	2<LSW3>5	MOV #<LSW3>, W5

APPENDIX A: DEVICE-SPECIFIC INFORMATION

A.1 Checksum Computation

The checksum computation is described in **Section 6.8 "Checksum Computation"**. Table A-1 shows how this 16-bit computation can be made for each dsPIC30F device. Computations for read code protection are shown both enabled and disabled. The checksum values assume that the Configuration registers are also erased. However, when code protection is enabled, the value of the FGS register is assumed to be 0x5.

A.2 dsPIC30F5011 and dsPIC30F5013

A.2.1 ICSP PROGRAMMING

The dsPIC30F5011 and dsPIC30F5013 processors require that the FBS and FSS registers be programmed with 0x0000 before the device is chip erased. The steps to perform this action are shown in Table 11-4.

A.2.2 ENHANCED ICSP PROGRAMMING

The dsPIC30F5011 and dsPIC30F5013 processors require that the FBS and FSS registers be programmed with 0x0000 using the `PROGC` command before the `ERASEB` command is used to erase the chip.

TABLE A-1: CHECKSUM COMPUTATION

Device	Read Code Protection	Checksum Computation	Erased Value	Value with 0xAAAAAA at 0x0 and Last Code Address
dsPIC30F2010	Disabled	CFGB+SUM(0:001FFF)	0xD406	0xD208
	Enabled	CFGB	0x0404	0x0404
dsPIC30F2011	Disabled	CFGB+SUM(0:001FFF)	0xD406	0xD208
	Enabled	CFGB	0x0404	0x0404
dsPIC30F2012	Disabled	CFGB+SUM(0:001FFF)	0xD406	0xD208
	Enabled	CFGB	0x0404	0x0404
dsPIC30F3010	Disabled	CFGB+SUM(0:003FFF)	0xA406	0xA208
	Enabled	CFGB	0x0404	0x0404
dsPIC30F3011	Disabled	CFGB+SUM(0:003FFF)	0xA406	0xA208
	Enabled	CFGB	0x0404	0x0404
dsPIC30F3012	Disabled	CFGB+SUM(0:003FFF)	0xA406	0xA208
	Enabled	CFGB	0x0404	0x0404
dsPIC30F3013	Disabled	CFGB+SUM(0:003FFF)	0xA406	0xA208
	Enabled	CFGB	0x0404	0x0404
dsPIC30F3014	Disabled	CFGB+SUM(0:003FFF)	0xA406	0xA208
	Enabled	CFGB	0x0404	0x0404
dsPIC30F4011	Disabled	CFGB+SUM(0:007FFF)	0x4406	0x4208
	Enabled	CFGB	0x0404	0x0404
dsPIC30F4012	Disabled	CFGB+SUM(0:007FFF)	0x4406	0x4208
	Enabled	CFGB	0x0404	0x0404
dsPIC30F4013	Disabled	CFGB+SUM(0:007FFF)	0x4406	0x4208
	Enabled	CFGB	0x0404	0x0404
dsPIC30F5011	Disabled	CFGB+SUM(0:00AFFF)	0xFC06	0xFA08
	Enabled	CFGB	0x0404	0x0404
dsPIC30F5013	Disabled	CFGB+SUM(0:00AFFF)	0xFC06	0xFA08
	Enabled	CFGB	0x0404	0x0404
dsPIC30F5015	Disabled	CFGB+SUM(0:00AFFF)	0xFC06	0xFA08
	Enabled	CFGB	0x0404	0x0404

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 ((FOSC&0xC10F) + (FWDT&0x803F) + (FBORPOR&0x87B3) + (FBS&0x310F) + (FSS&0x330F) + (FGS&0x0007) + (FICD&0xC003))

TABLE A-1: CHECKSUM COMPUTATION (CONTINUED)

Device	Read Code Protection	Checksum Computation	Erased Value	Value with 0xAAAAAA at 0x0 and Last Code Address
dsPIC30F5016	Disabled	CFGB+SUM(0:00AFFF)	0xFC06	0xFA08
	Enabled	CFGB	0x0404	0x0404
dsPIC30F6010	Disabled	CFGB+SUM(0:017FFF)	0xC406	0xC208
	Enabled	CFGB	0x0404	0x0404
dsPIC30F6010A	Disabled	CFGB+SUM(0:017FFF)	0xC406	0xC208
	Enabled	CFGB	0x0404	0x0404
dsPIC30F6011	Disabled	CFGB+SUM(0:015FFF)	0xF406	0xF208
	Enabled	CFGB	0x0404	0x0404
dsPIC30F6011A	Disabled	CFGB+SUM(0:015FFF)	0xF406	0xF208
	Enabled	CFGB	0x0404	0x0404
dsPIC30F6012	Disabled	CFGB+SUM(0:017FFF)	0xC406	0xC208
	Enabled	CFGB	0x0404	0x0404
dsPIC30F6012A	Disabled	CFGB+SUM(0:017FFF)	0xC406	0xC208
	Enabled	CFGB	0x0404	0x0404
dsPIC30F6013	Disabled	CFGB+SUM(0:015FFF)	0xF406	0xF208
	Enabled	CFGB	0x0404	0x0404
dsPIC30F6013A	Disabled	CFGB+SUM(0:015FFF)	0xF406	0xF208
	Enabled	CFGB	0x0404	0x0404
dsPIC30F6014	Disabled	CFGB+SUM(0:017FFF)	0xC406	0xC208
	Enabled	CFGB	0x0404	0x0404
dsPIC30F6014A	Disabled	CFGB+SUM(0:017FFF)	0xC406	0xC208
	Enabled	CFGB	0x0404	0x0404
dsPIC30F6015	Disabled	CFGB+SUM(0:017FFF)	0xC406	0xC208
	Enabled	CFGB	0x0404	0x0404

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 ((FOSC&0xC10F) + (FWDT&0x803F) + (FBORPOR&0x87B3) + (FBS&0x310F) + (FSS&0x330F) + (FGS&0x0007) + (FICD&0xC003))

APPENDIX B: HEX FILE FORMAT

Flash programmers process the standard HEX format used by the Microchip development tools. The format supported is the Intel® HEX 32 Format (INHX32). Please refer to Appendix A in the "*MPASM User's Guide*" (DS33014) for more information about hex file formats.

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 records and '04' for extended-address record.
- `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 specified address 0x100. Note also 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.

NOTES:

Worldwide Sales and Service

AMERICAS

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

Atlanta
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

Kokomo
Kokomo, IN
Tel: 765-864-8360
Fax: 765-864-8387

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

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

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

ASIA/PACIFIC

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

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

China - Beijing
Tel: 86-10-8528-2100
Fax: 86-10-8528-2104

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

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

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

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

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

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

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

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

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

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

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

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

ASIA/PACIFIC

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

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

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

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

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

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

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

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

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

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

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

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

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

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

EUROPE

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

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

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

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

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

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

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

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

08/04/10