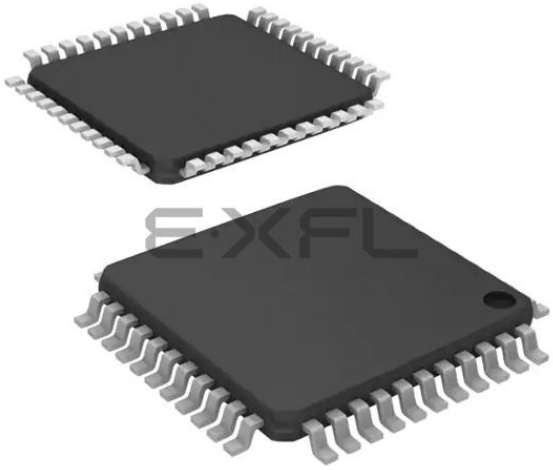


Welcome to E-XFL.COM

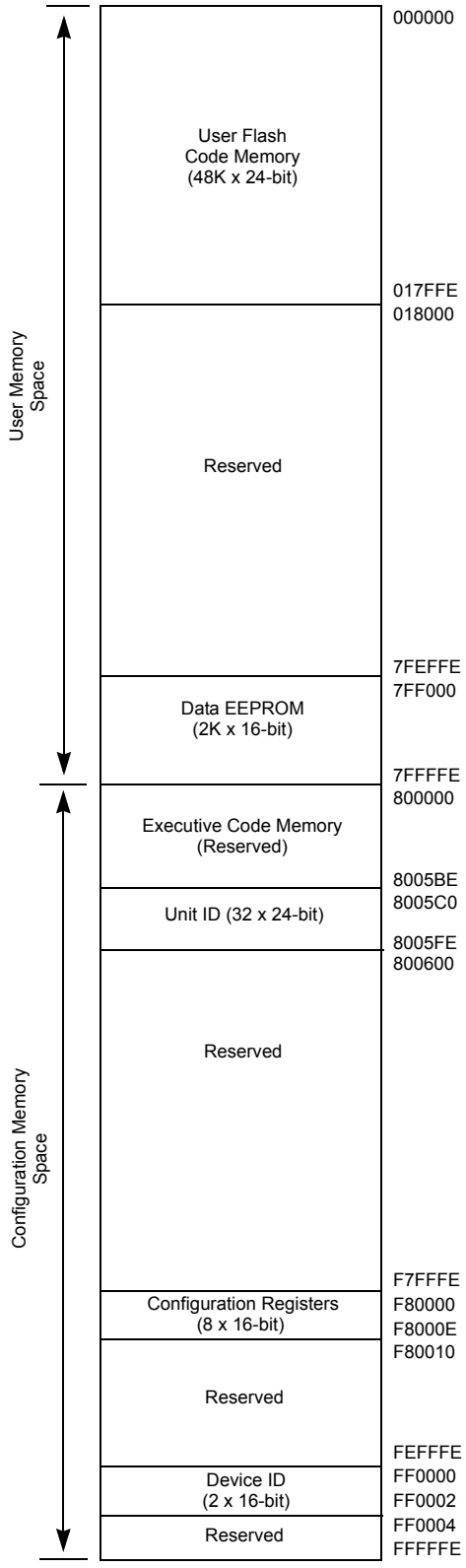
What is "Embedded - Microcontrollers"?

"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	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, Motor Control PWM, QEI, POR, PWM, WDT
Number of I/O	30
Program Memory Size	24KB (8K x 24)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	2.5V ~ 5.5V
Data Converters	A/D 9x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-TQFP
Supplier Device Package	44-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/dspic30f3011t-20i-pt

FIGURE 2-2: PROGRAM MEMORY MAP



Note: The address boundaries for user Flash code memory and data EEPROM are device-dependent.

5.0 DEVICE PROGRAMMING

5.1 Overview of the Programming Process

Once the programming executive has been verified in memory (or loaded if not present), the dsPIC30F can be programmed using the command set shown in Table 5-1. A detailed description for each command is provided in **Section 8.0 "Programming Executive Commands"**.

TABLE 5-1: COMMAND SET SUMMARY

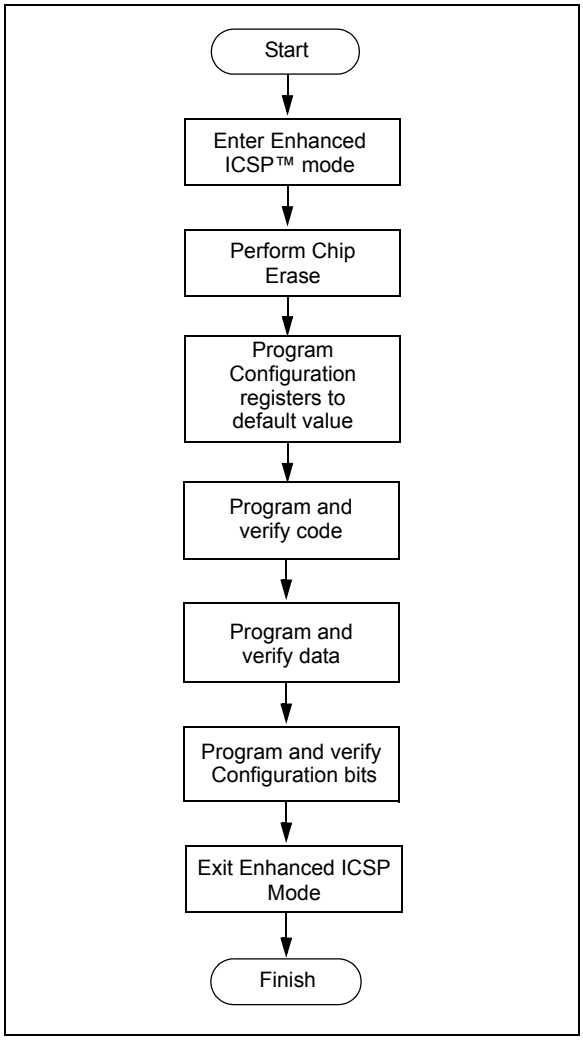
Command	Description
SCHECK	Sanity check
READD	Read data EEPROM, Configuration registers and device ID
READP	Read code memory
PROGD	Program one row of data EEPROM and verify
PROGP	Program one row of code memory and verify
PROGC	Program Configuration bits and verify
ERASEB	Bulk Erase, or erase by segment
ERASED	Erase data EEPROM
ERASEP	Erase code memory
QBLANK	Query if the code memory and data EEPROM are blank
QVER	Query the software version

A high-level overview of the programming process is illustrated in Figure 5-1. The process begins by entering Enhanced ICSP mode. The chip is then bulk erased, which clears all memory to '1' and allows the device to be programmed. The Chip Erase is verified before programming begins. Next, the code memory, data Flash and Configuration bits are programmed. As these memories are programmed, they are each verified to ensure that programming was successful. If no errors are detected, the programming is complete and Enhanced ICSP mode is exited. If any of the verifications fail, the procedure should be repeated, starting from the Chip Erase.

If Advanced Security features are enabled, then individual Segment Erase operations need to be performed, based on user selections (i.e., based on the specific needs of the user application). The specific operations that are used typically depend on the order in which various segments need to be programmed for a given application or system.

Section 5.2 "Entering Enhanced ICSP Mode" through **Section 5.8 "Exiting Enhanced ICSP Mode"** describe the programming process in detail.

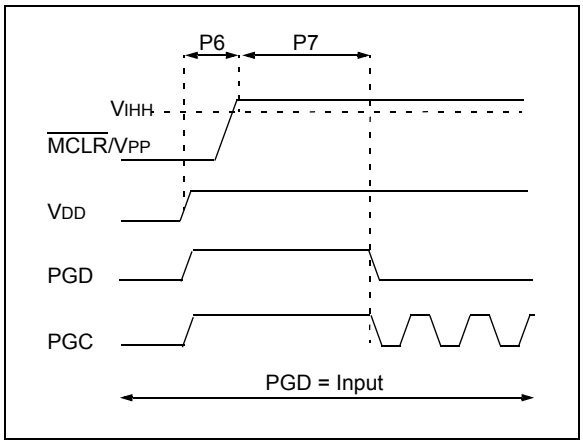
FIGURE 5-1: PROGRAMMING FLOW



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.

5.5.3 PROGRAMMING VERIFICATION

Once 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 once the entire device is programmed using a checksum computation, as described in **Section 6.8 "Checksum Computation"**.

5.6 Data EEPROM Programming

5.6.1 OVERVIEW

The panel architecture for the data EEPROM memory array consists of 128 rows of sixteen 16-bit data words. Each panel stores 2K words. All devices have either one or no memory panels. Devices with data EEPROM provide either 512 words, 1024 words or 2048 words of memory on the one panel (see Table 5-3).

TABLE 5-3: DATA EEPROM SIZE

Device	Data EEPROM Size (Words)	Number of Rows
dsPIC30F2010	512	32
dsPIC30F2011	0	0
dsPIC30F2012	0	0
dsPIC30F3010	512	32
dsPIC30F3011	512	32
dsPIC30F3012	512	32
dsPIC30F3013	512	32
dsPIC30F3014	512	32
dsPIC30F4011	512	32
dsPIC30F4012	512	32
dsPIC30F4013	512	32
dsPIC30F5011	512	32
dsPIC30F5013	512	32
dsPIC30F5015	512	32
dsPIC30F5016	512	32
dsPIC30F6010	2048	128
dsPIC30F6010A	2048	128
dsPIC30F6011	1024	64
dsPIC30F6011A	1024	64
dsPIC30F6012	2048	128
dsPIC30F6012A	2048	128
dsPIC30F6013	1024	64
dsPIC30F6013A	1024	64
dsPIC30F6014	2048	128
dsPIC30F6014A	2048	128
dsPIC30F6015	2048	128

5.6.2 PROGRAMMING METHODOLOGY

The programming executive uses the `PROGD` command to program the data EEPROM. Figure 5-4 illustrates the flowchart of the process. Firstly, the number of rows to program (RemainingRows) is based on the device size, and the destination address (DestAddress) is set to '0'. In this example, 128 rows (2048 words) of data EEPROM will be programmed.

The first `PROGD` command programs the first row of data EEPROM. Once the command completes successfully, 'RemainingRows' is decremented by 1 and compared with 0. Since there are 127 more rows to program, 'BaseAddress' is incremented by 0x20 to point to the next row of data EEPROM. This process is then repeated until all 128 rows of data EEPROM are programmed.

FIGURE 5-4: FLOWCHART FOR PROGRAMMING dsPIC30F6014A DATA EEPROM

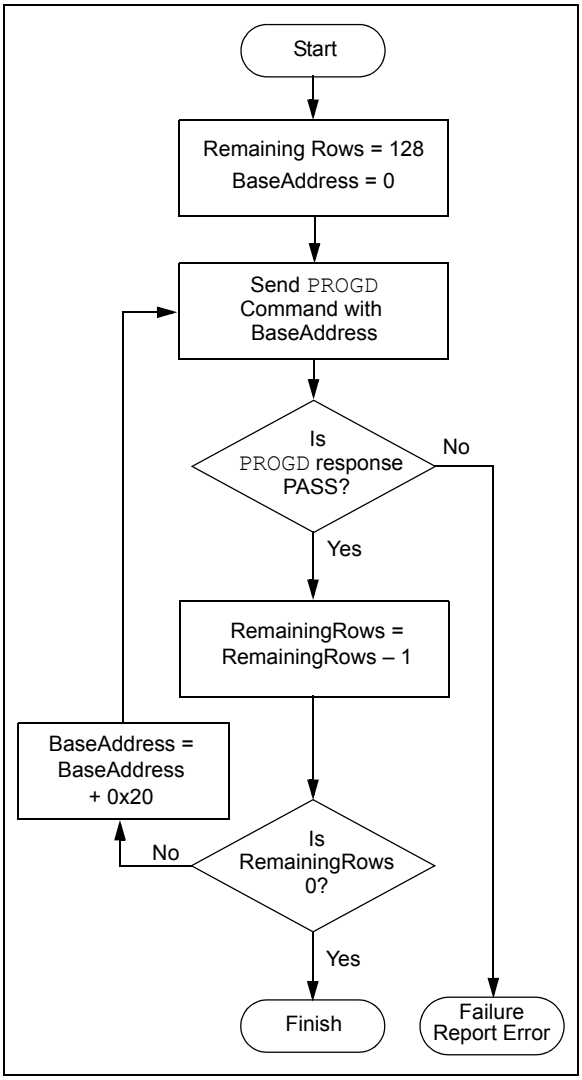


TABLE 5-10: dsPIC30F CONFIGURATION REGISTERS (FOR dsPIC30F2011/2012, dsPIC30F3010/3011/3012/3013/3014, dsPIC30F4013 AND dsPIC30F5015/5016)

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').

6.0 OTHER PROGRAMMING FEATURES

6.1 Erasing Memory

Memory is erased by using an ERASEB, ERASED or ERASEP command, as detailed in **Section 8.5 "Command Descriptions"**. Code memory can be erased by row using ERASEP. Data EEPROM can be erased by row using ERASED. When memory is erased, the affected memory locations are set to '1's.

ERASEB provides several Bulk Erase options. Performing a Chip Erase with the ERASEB command clears all code memory, data EEPROM and code protection registers. Alternatively, ERASEB can be used to selectively erase either all code memory or data EEPROM. Erase options are summarized in Table 6-1.

TABLE 6-1: ERASE OPTIONS

Command	Affected Region
ERASEB	Entire chip(1) or all code memory or all data EEPROM, or erase by segment
ERASED	Specified rows of data EEPROM
ERASEP(2)	Specified rows of code memory

Note 1: The system operation Configuration registers and device ID registers are not erasable.

2: ERASEP cannot be used to erase code-protect Configuration bits. These bits must be erased using ERASEB.

6.2 Modifying Memory

Instead of bulk-erasing the device before programming, it is possible that you may want to modify only a section of an already programmed device. In this situation, Chip Erase is not a realistic option.

Instead, you can erase selective rows of code memory and data EEPROM using ERASEP and ERASED, respectively. You can then reprogram the modified rows with the PROGP and PROGD command pairs. In these cases, when code memory is programmed, single-panel programming must be specified in the PROGP command.

For modification of Advanced Code Protection bits for a particular segment, the entire chip must first be erased with the ERASEB command. Alternatively, on devices that support Advanced Security, individual segments (code and/or data EEPROM) may be erased, by suitably changing the MS (Memory Select) field in the ERASEB command. The code-protect Configuration bits can then be reprogrammed using the PROGC command.

Note: If read or write code protection is enabled for a segment, no modifications can be made to that segment until code protection is disabled. Code protection can only be disabled by performing a Chip Erase or by performing a Segment Erase operation for the required segment.

6.3 Reading Memory

The READD command reads the data EEPROM, Configuration bits and device ID of the device. This command only returns 16-bit data and operates on 16-bit registers. READD can be used to return the entire contents of data EEPROM.

The READP command reads the code memory of the device. This command only returns 24-bit data packed as described in **Section 8.3 "Packed Data Format"**. READP can be used to read up to 32K instruction words of code memory.

Note: Reading an unimplemented memory location causes the programming executive to reset. All READD and READP commands **must** specify only valid memory locations.

6.4 Programming Executive Software Version

At times, it may be necessary to determine the version of programming executive stored in executive memory. The QVER command performs this function. See **Section 8.5.11 "QVER Command"** for more details about this command.

6.5 Data EEPROM Information in the Hexadecimal File

To allow portability of code, the programmer must read the data EEPROM information from the hexadecimal file. If data EEPROM information is not present, a simple warning message should be issued by the programmer. Similarly, when saving a hexadecimal file, all data EEPROM information must be included. An option to not include the data EEPROM information can be provided.

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

TABLE 8-1: PROGRAMMING EXECUTIVE COMMAND SET

Opcode	Mnemonic	Length (16-bit words)	Time Out	Description
0x0	SCHECK	1	1 ms	Sanity check.
0x1	READD	4	1 ms/row	Read N 16-bit words of data EEPROM, Configuration registers or device ID starting from specified address.
0x2	READP	4	1 ms/row	Read N 24-bit instruction words of code memory starting from specified address.
0x3	Reserved	N/A	N/A	This command is reserved. It will return a NACK.
0x4	PROGD(2)	19	5 ms	Program one row of data EEPROM at the specified address, then verify.
0x5	PROGP(1)	51	5 ms	Program one row of code memory at the specified address, then verify.
0x6	PROGC	4	5 ms	Write byte or 16-bit word to specified Configuration register.
0x7	ERASEB	2	5 ms	Bulk Erase (entire code memory or data EEPROM), or erase by segment.
0x8	ERASED(2)	3	5 ms/row	Erase rows of data EEPROM from specified address.
0x9	ERASEP(1)	3	5 ms/row	Erase rows of code memory from specified address.
0xA	QBLANK	3	300 ms	Query if the code memory and data EEPROM are blank.
0xB	QVER	1	1 ms	Query the programming executive software version.

Note 1: One row of code memory consists of (32) 24-bit words. Refer to Table 5-2 for device-specific information.
2: One row of data EEPROM consists of (16) 16-bit words. Refer to Table 5-3 for device-specific information.

8.5.5 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	0x33
Reserved	0x0
Addr_MSB	MSB of 24-bit destination address
Addr_LS	LS 16 bits of 24-bit destination address
D_1	16-bit data word 1
D_2	16-bit data word 2
...	16-bit data word 3 through 47
D_48	16-bit data word 48

The PROGP command instructs the programming executive to program one row of code memory (32 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 0x40.

The data to program to memory, located in command words D_1 through D_48, must be arranged using the packed instruction word format shown in Figure 8-2.

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 5-2 for code memory size information.

8.5.6 PROGC COMMAND

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

Field	Description
Opcode	0x6
Length	0x4
Reserved	0x0
Addr_MSB	MSB of 24-bit destination address
Addr_LS	LS 16 bits of 24-bit destination address
Data	Data to program

The PROGC command programs data to the specified Configuration register and verifies the programming. Configuration registers are 16 bits wide, and this command allows one Configuration register to be programmed.

Expected Response (2 words):
0x1600
0x0002

Note: This command can only be used for programming Configuration registers.

9.2.3 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 9-3.

TABLE 9-3: QE_Code FOR QUERIES

Query	QE_Code
QBLANK	0x0F = Code memory and data EEPROM are NOT blank 0xF0 = Code memory and data EEPROM are 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 9-4. If a command is successfully processed, the returned QE_Code is set to 0x0, which indicates that there was no error in the command processing. If the verify of the programming for the PROGD, PROGP or PROGC command fails, the QE_Code is set to 0x1. For all other programming executive errors, the QE_Code is 0x2.

TABLE 9-4: QE_Code FOR NON-QUERY COMMANDS

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

9.2.4 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 READD and READP commands, the length of each response is only 2 words.

The response to the READD command is N + 2 words, where N is the number of words specified in the READD command.

The response to the READP command uses the packed instruction word format described in **Section 8.3 "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.

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"**.

Table 11-4 shows the ICSP programming process for bulk-erasing program memory. This process includes the ICSP command code, which must be transmitted (for each instruction) to the Least Significant bit first using the PGC and PGD pins (see Figure 11-2).

If an individual Segment Erase operation is required, the NVMCON value must be replaced by the value for the corresponding Segment Erase operation.

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

TABLE 11-4: SERIAL INSTRUCTION EXECUTION FOR BULK ERASING PROGRAM MEMORY (ONLY IN NORMAL-VOLTAGE SYSTEMS)

Command (Binary)	Data (Hexadecimal)	Description
Step 1: Exit the Reset vector.		
0000 0000 0000	040100 040100 000000	GOTO 0x100 GOTO 0x100 NOP
Step 2: Set NVMCON to program the FBS Configuration register.(1)		
0000 0000	24008A 883B0A	MOV #0x4008, W10 MOV W10, NVMCON
Step 3: Initialize the TBLPAG and write pointer (W7) for TBLWT instruction for Configuration register.(1)		

0000
0000
0000
	200F80
	880190
	200067
		MOV #0xF8, W0
		MOV W0, TBLPAG
		MOV #0x6, W7

Step 4: Load the Configuration Register data to W6.(1)		
0000 0000	EB0300 000000	CLR W6 NOP
Step 5: Load the Configuration Register write latch. Advance W7 to point to next Configuration register.(1)		
0000	BB1B86	TBLWTL W6, [W7++]
Step 6: Unlock the NVMCON for programming the Configuration register.(1)		

0000
0000
0000
0000
	200558
	200AA9
	883B38
	883B39
		MOV #0x55, W8
		MOV #0xAA, W9
		MOV W8, NVMKEY
		MOV W9, NVMKEY

Step 7: Initiate the programming cycle.(1)		
0000 0000 0000 – 0000 0000 0000 0000 0000	A8E761 000000 000000 – 000000 000000 A9E761 000000 000000	BSET NVMCON, #WR NOP NOP Externally time 2 ms NOP NOP BCLR NVMCON, #WR NOP NOP
Step 8: Repeat steps 5-7 one time to program 0x0000 to RESERVED2 Configuration register.(1)		
Step 9: Set the NVMCON to erase all Program Memory.		
00000 0000	2407FA 883B0A	MOV #0x407F, W10 MOV W10, NVMCON
Step 10: Unlock the NVMCON for programming.		

Note 1: Steps 2-8 are only required for the dsPIC30F5011/5013 devices. These steps may be skipped for all other devices in the dsPIC30F family.

TABLE 11-5: SERIAL INSTRUCTION EXECUTION FOR ERASING PROGRAM MEMORY (EITHER IN LOW-VOLTAGE OR NORMAL-VOLTAGE SYSTEMS) (CONTINUED)

Command (Binary)	Data (Hexadecimal)	Description
Step 18: Unlock the NVMCON to erase 1 row of data memory.		
0000	200558	MOV #0x55, W8
0000	883B38	MOV W8, NVMKEY
0000	200AA9	MOV #0xAA, W9
0000	883B39	MOV W9, NVMKEY
Step 19: Initiate the erase cycle.		
0000	A8E761	BSET NVMCON, #WR
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, #WR
0000	000000	NOP
0000	000000	NOP
Step 20: Update the row address stored in NVMADR.		
0000	430307	ADD W6, W7, W6
0000	883B16	MOV W6, NVMADR
Step 21: Reset device internal PC.		
0000	040100	GOTO 0x100
0000	000000	NOP
Step 22: Repeat Steps 17-21 until all rows of data memory are erased.		

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

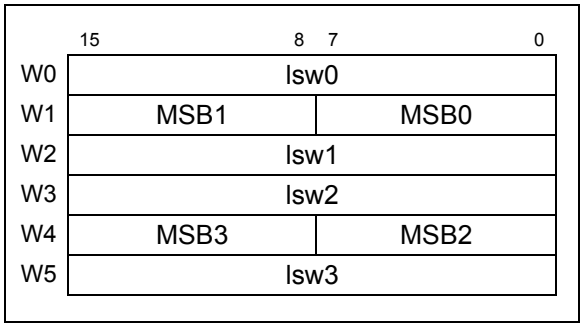


TABLE 11-8: SERIAL INSTRUCTION EXECUTION FOR WRITING CODE MEMORY

Command (Binary)	Data (Hexadecimal)	Description
Step 1: Exit the Reset vector.		
0000 0000 0000	040100 040100 000000	GOTO 0x100 GOTO 0x100 NOP
Step 2: Set the NVMCON to program 32 instruction words.		
0000 0000	24001A 883B0A	MOV #0x4001, W10 MOV W10, NVMCON
Step 3: Initialize the write pointer (W7) for `TBLWT` instruction.		

0000
0000
0000
	200xx0
	880190
	2xxxx7
		MOV #<DestinationAddress23:16>, W0
		MOV W0, TBLPAG
		MOV #<DestinationAddress15:0>, W7

Step 4: Initialize the read pointer (W6) and load W0:W5 with the next 4 instruction words to program.		

0000
0000
0000
0000
0000
0000
	2xxxx0
	2xxxx1
	2xxxx2
	2xxxx3
	2xxxx4
	2xxxx5
		MOV #<LSW0>, W0
		MOV #<MSB1:MSB0>, W1
		MOV #<LSW1>, W2
		MOV #<LSW2>, W3
		MOV #<MSB3:MSB2>, W4
		MOV #<LSW3>, W5

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

12.2 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 11.10 “Reading Code Memory” . A procedure for reading executive memory is shown in Table 12-2. Note that in Step 2, the TBLPAG register is set to 0x80 such that executive memory may be read.

TABLE 12-2: READING EXECUTIVE 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: Initialize TBLPAGand the read pointer (W6) for TBLRDinstruction.			
0000	200800	MOV	#0x80, W0
0000	880190	MOV	W0, TBLPAG
0000	EB0300	CLR	W6
Step 3: Initialize the write pointer (W7), and store the next four locations of executive memory to W0:W5.			
0000	EB0380		
0000	000000		
0000	BA1B96		
0000	000000		
0000	000000		
0000	BADBB6		
0000			
0000			
0000			
0000			
0000			
0000			
0000			
0000			
0000			
0000			
0000			
0000			
0000			
0000			
0000			
0000			
0000			
0000			
0000			
0000			

TABLE 12-2: READING EXECUTIVE 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	—	Clock out contents of VISI register
0000	883C21	MOV W1, VISI
0000	000000	NOP
0001	—	Clock out contents of VISI register
0000	883C22	MOV W2, VISI
0000	000000	NOP
0001	—	Clock out contents of VISI register
0000	883C23	MOV W3, VISI
0000	000000	NOP
0001	—	Clock out contents of VISI register
0000	883C24	MOV W4, VISI
0000	000000	NOP
0001	—	Clock out contents of VISI register
0000	883C25	MOV W5, VISI
0000	000000	NOP
0001	—	Clock out contents of VISI register
Step 5: Reset the device internal PC.		
0000	040100	GOTO 0x100
0000	000000	NOP
Step 6: Repeat Steps 3-5 until all 736 instruction words of executive memory are read.		

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

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

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

Trademarks

Printed on recycled paper.

ISBN: 978-1-60932-636-4

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

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