



Welcome to [E-XFL.COM](https://www.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	Active
Core Processor	PIC
Core Size	16-Bit
Speed	16MHz
Connectivity	I ² C, PMP, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	53
Program Memory Size	128KB (43K x 24)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	8K x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 3.6V
Data Converters	A/D 16x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-VFQFN Exposed Pad
Supplier Device Package	64-VQFN (9x9)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic24fj128ga006-i-mr

PIC24FJXXXGA0XX

The regulator provides power to the core from the other VDD pins. A low-ESR capacitor (such as tantalum) must be connected to the VDDCORE pin (Figure 2-2 and Figure 2-3). This helps to maintain the stability of the regulator. The specifications for core voltage and capacitance are listed in **Section 7.0 “AC/DC Characteristics and Timing Requirements”**.

FIGURE 2-2: CONNECTIONS FOR THE ON-CHIP REGULATOR (64/80/100-PIN DEVICES)

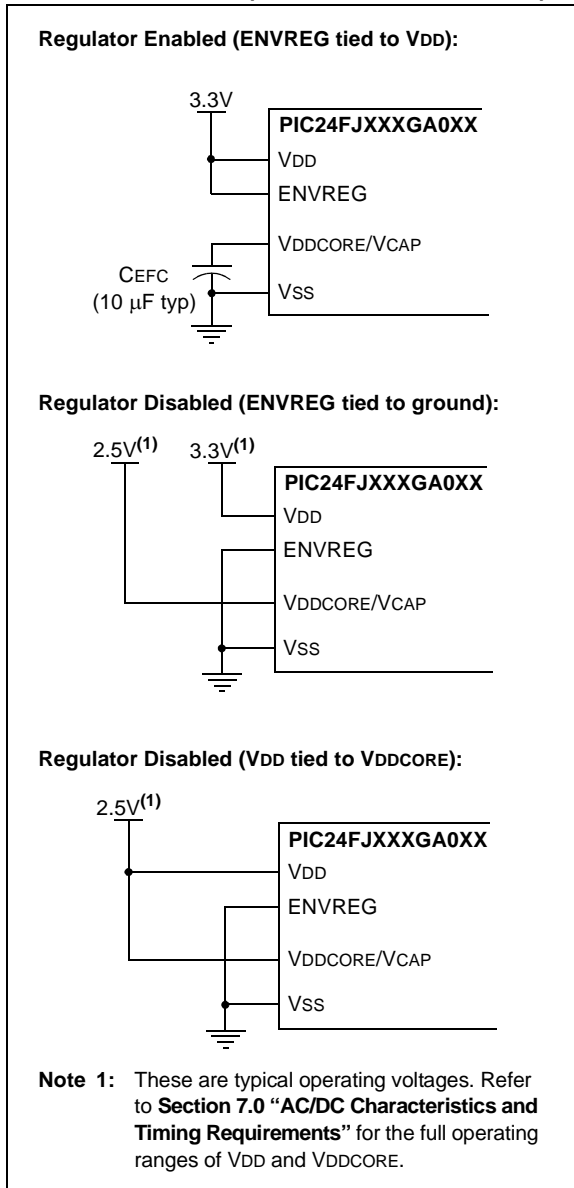
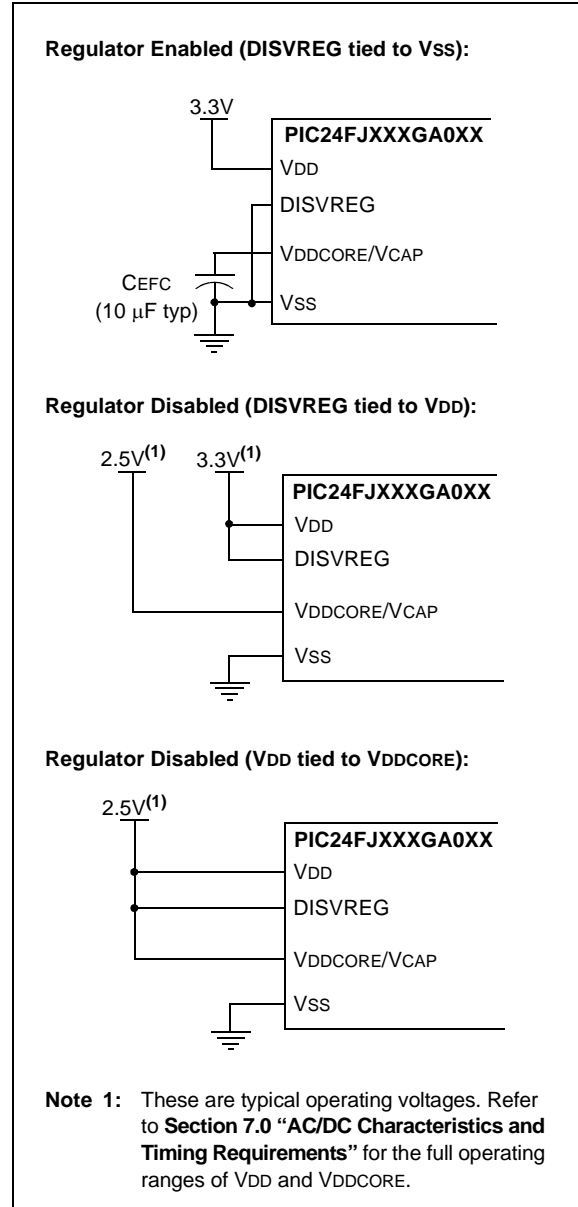


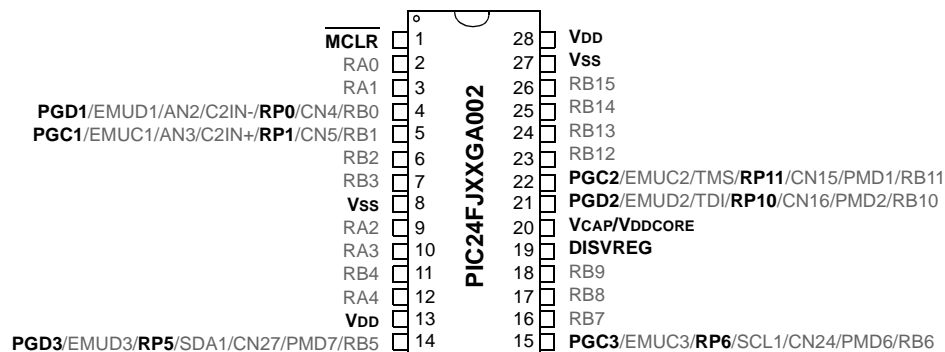
FIGURE 2-3: CONNECTIONS FOR THE ON-CHIP REGULATOR (28/44-PIN DEVICES)



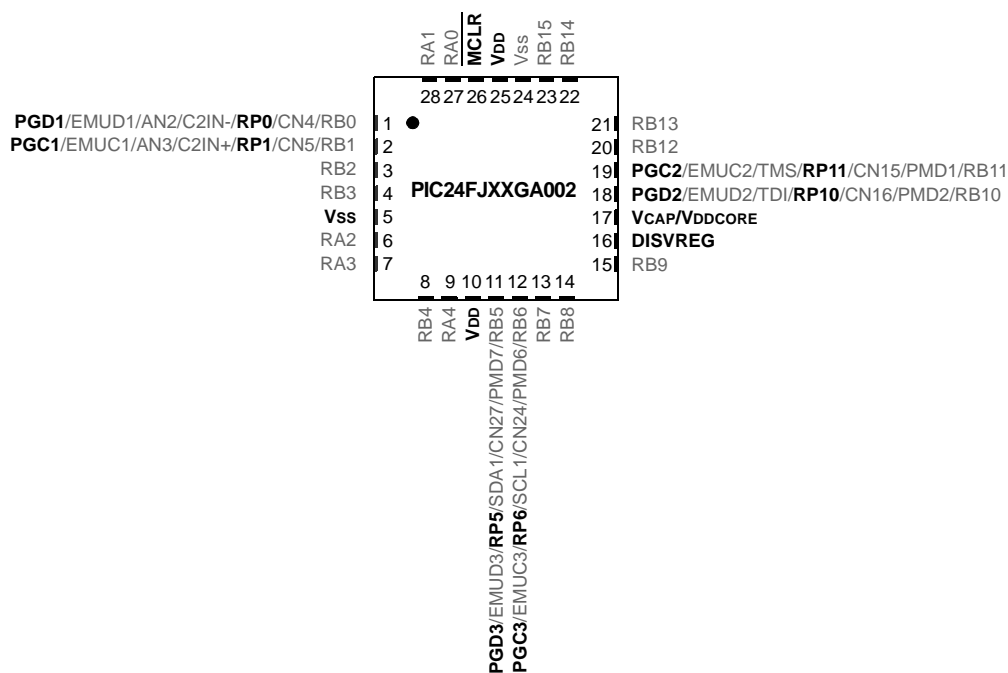
PIC24FJXXGA0XX

Pin Diagrams

28-Pin PDIP, SSOP, SOIC



28-Pin QFN⁽¹⁾

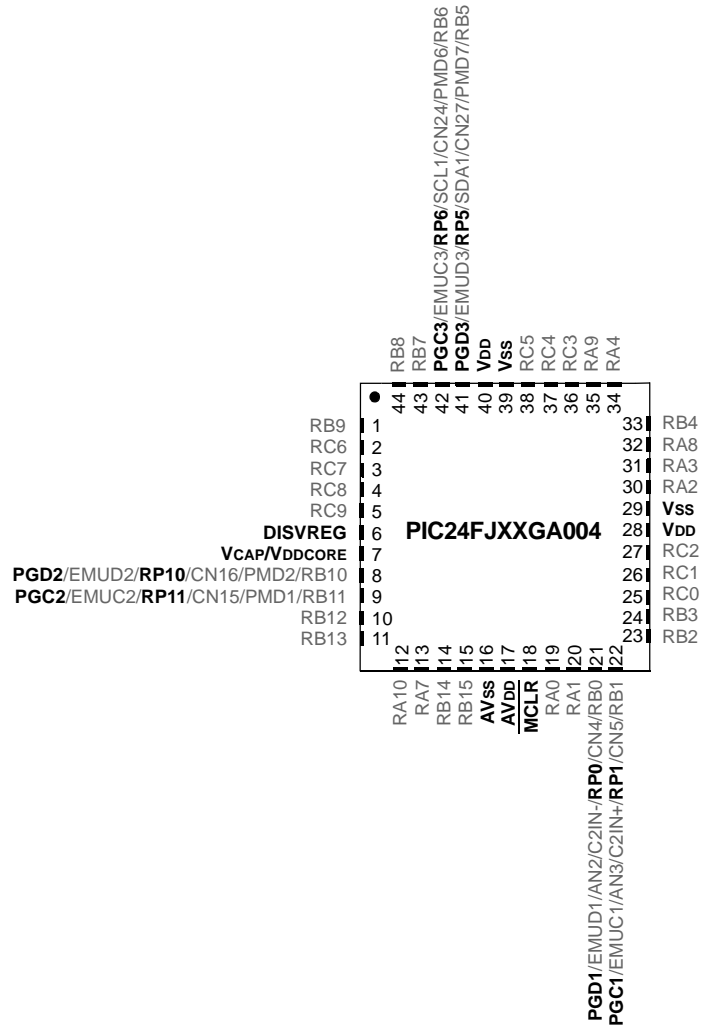


Legend: RP_x represents remappable peripheral pins.

Note 1: The bottom pad of QFN packages should be connected to Vss.

Pin Diagrams (Continued)

44-Pin QFN⁽¹⁾

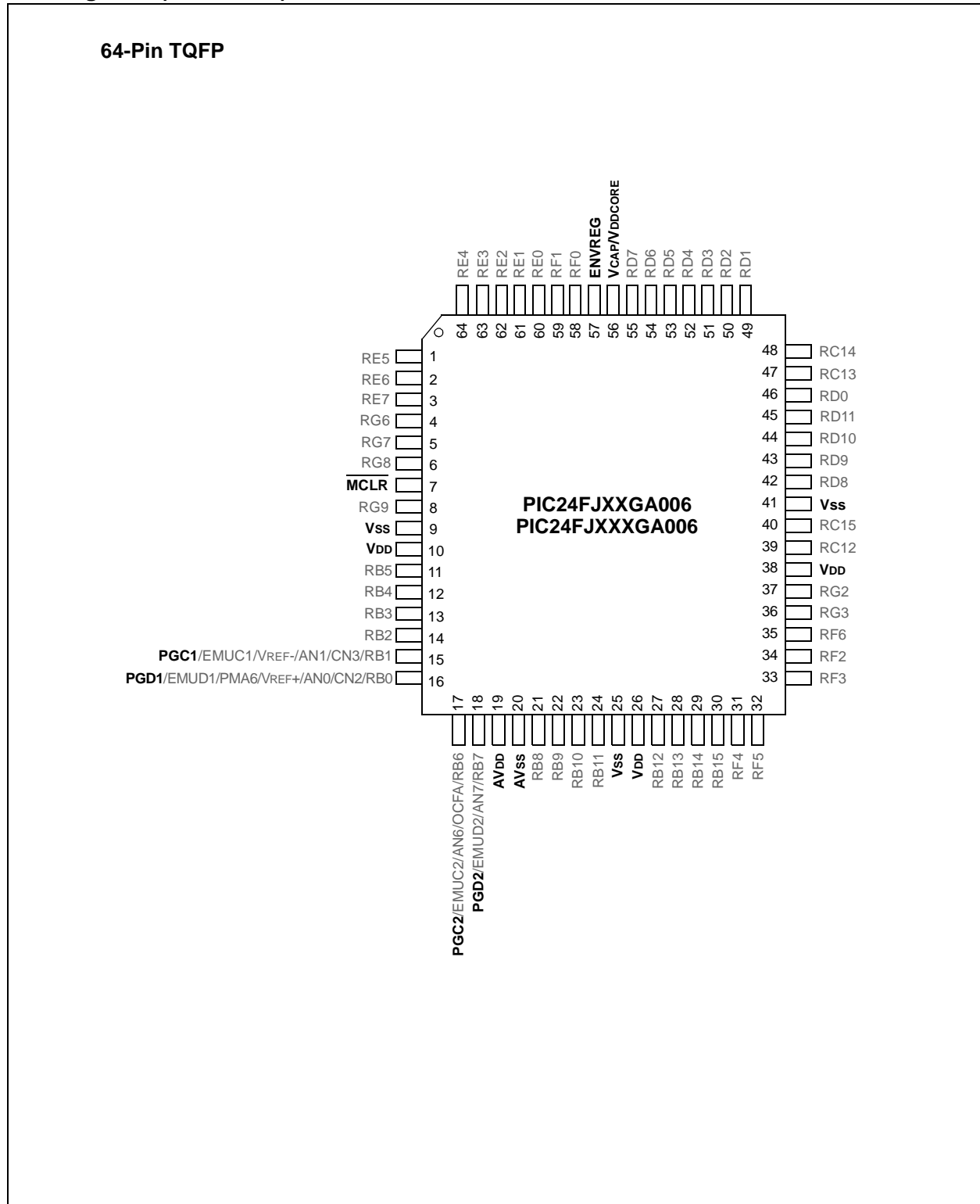


Legend: RPx represents remappable peripheral pins.

Note 1: The bottom pad of QFN packages should be connected to Vss.

PIC24FJXXXGA0XX

Pin Diagrams (Continued)



3.3 Entering ICSP Mode

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

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

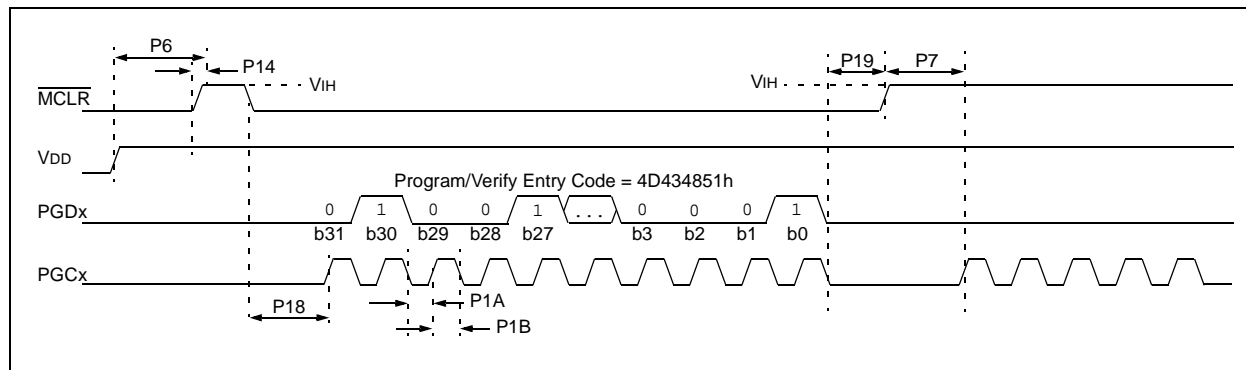
The programming voltage applied to $\overline{\text{MCLR}}$ is V_{IH} , which is essentially V_{DD} in the case of PIC24FJXXXGA0XX devices. There is no minimum time requirement for holding at V_{IH} . After V_{IH} is removed, an interval of at least P18 must elapse before presenting the key sequence on PGDx.

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

Once the key sequence is complete, 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 and P7, must elapse before presenting data on PGDx. Signals appearing on PGCx before P7 has elapsed will not be interpreted as valid.

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

FIGURE 3-4: ENTERING ICSP™ MODE



PIC24FJXXXGA0XX

3.6 Writing Code Memory

The procedure for writing code memory is the same as 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 3-5 shows the ICSP programming details, including the serial pattern with the ICSP command code which must be transmitted, Least Significant bit first, using the PGCx and PGDx pins (see Figure 3-2).

In Step 1, the Reset vector is exited. In Step 2, the NVMCON register is initialized for programming a full row 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 in TBLPAG and the lower 16 bits of the destination address are stored in W7.)

To minimize the programming time, A packed instruction format is used (Figure 3-6).

In Step 4, four packed instruction words are stored in 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) are shown in Figure 3-6.

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 64 instruction words at a time, Steps 4 and 5 are repeated 16 times to load all the write latches (Step 6).

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

FIGURE 3-6: PACKED INSTRUCTION WORDS IN W<0:5>

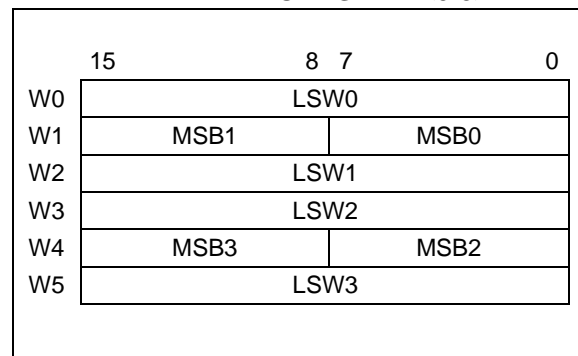
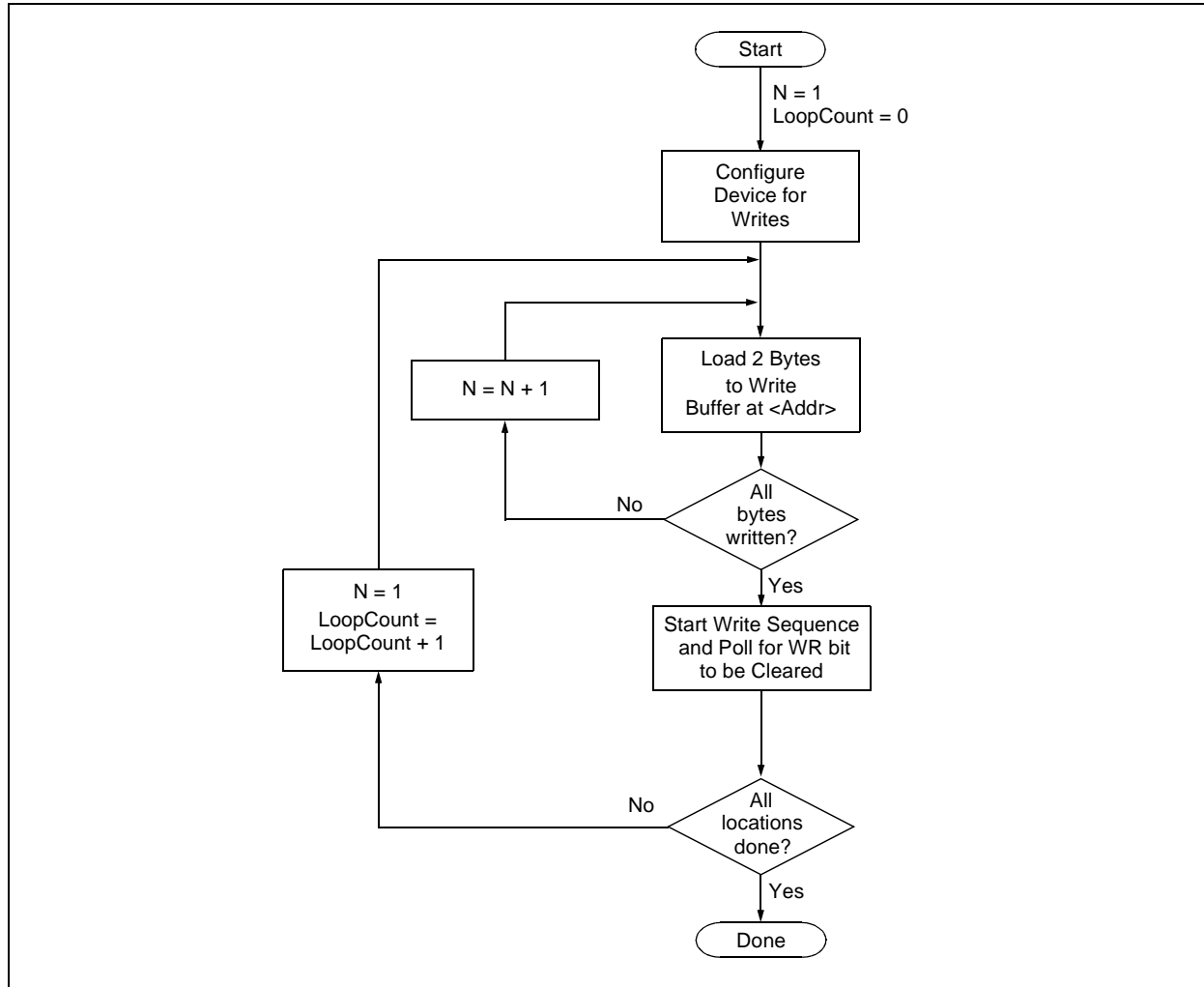


TABLE 3-5: SERIAL INSTRUCTION EXECUTION FOR WRITING CODE MEMORY

Command (Binary)	Data (Hex)	Description
Step 1: Exit the Reset vector.		
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
Step 2: Set the NVMCON to program 64 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: 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

FIGURE 3-7: PROGRAM CODE MEMORY FLOW



3.7 Writing Configuration Words

The PIC24FJXXXGA0XX family configuration is stored in Flash Configuration Words at the end of the user space program memory and in multiple register Configuration Words located in the test space.

These registers reflect values read at any Reset from program memory locations. The values can be changed only by programming the content of the corresponding Flash Configuration Word and resetting the device. The Reset forces an automatic reload of the Flash stored configuration values by sequencing through the dedicated Flash Configuration Words and transferring the data into the Configuration registers. To change the values of the Flash Configuration Word once it has been programmed, the device must be Chip Erased, as described in **Section 3.5 “Erasing Program Memory”**, and reprogrammed to the desired value. It is not possible to program a '0' to '1', but they may be programmed from a '1' to '0' to enable code protection.

Table 3-7 shows the ICSP programming details for programming the Configuration Word locations, including the serial pattern with the ICSP command code which must be transmitted, Least Significant bit first, using the PGCx and PGDx pins (see Figure 3-2).

In Step 1, the Reset vector is exited. In Step 2, the NVMCON register is initialized for 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 TBLPAG register must be loaded with the following:

- 96 and 64 Kbyte devices – 00h
- 128 Kbyte devices – 01h

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

TABLE 3-6: DEFAULT CONFIGURATION REGISTER VALUES

Address	Name	Default Value
Last Word	CW1	7FFFh ⁽¹⁾
Last Word – 2	CW2	FFFFh

Note 1: CW1<15> is reserved and must be programmed to '0'.

PIC24FJXXXGA0XX

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

Command (Binary)	Data (Hex)	Description
Step 1: Exit the Reset vector.		
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
Step 2: Initialize the Write Pointer (W7) for the TBLWT instruction.		
0000	2xxxx7	MOV <CW2Address15:0>, W7
Step 3: Set the NVMCON register to program CW2.		
0000	24003A	MOV #0x4003, W10
0000	883B0A	MOV W10, NVMCON
Step 4: Initialize the TBLPAG register.		
0000	200xx0	MOV <CW2Address23:16>, W0
0000	880190	MOV W0, TBLPAG
Step 5: Load the Configuration register data to W6.		
0000	2xxxx6	MOV #<CW2_VALUE>, W6
Step 6: Write the Configuration register data to the write latch and increment the Write Pointer.		
0000	000000	NOP
0000	BB1B86	TBLWTL W6, [W7++]
0000	000000	NOP
0000	000000	NOP
Step 7: Initiate the write cycle.		
0000	A8E761	BSET NVMCON, #WR
0000	000000	NOP
0000	000000	NOP
Step 8: Repeat this step to poll the WR bit (bit 15 of NVMCON) until it is cleared by the hardware.		
0000	040200	GOTO 0x200
0000	000000	NOP
0000	803B02	MOV NVMCON, W2
0000	883C22	MOV W2, VISI
0000	000000	NOP
0001	<VISI>	Clock out contents of the VISI register.
0000	000000	NOP
Step 9: Reset device internal PC.		
0000	040200	GOTO 0x200
0000	000000	NOP
Step 10: Repeat Steps 5-9 to write CW1.		

4.4 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 (FF0002h:FF0000h) 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 by the Blank Check.

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

4.5 Code Memory Programming

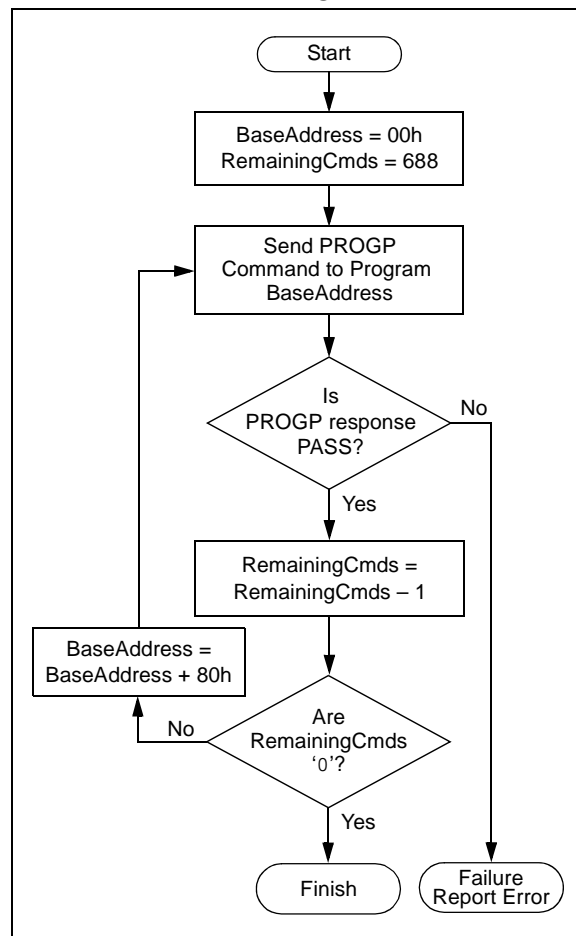
4.5.1 PROGRAMMING METHODOLOGY

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

A flowchart for programming code memory is shown in Figure 4-4. In this example, all 44K instruction words of a PIC24FJ128GA device are programmed. First, the number of commands to send (called ‘RemainingCmds’ in the flowchart) is set to 688 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 PIC24FJXXXGA0XX device. 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 80h 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. No special handling must be performed when a panel boundary is crossed.

FIGURE 4-4: FLOWCHART FOR PROGRAMMING CODE MEMORY



PIC24FJXXXGA0XX

4.5.2 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 of the programmed code memory.

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

4.6 Configuration Bits Programming

4.6.1 OVERVIEW

The PIC24FJXXXGA0XX family has Configuration bits stored in the last two locations of implemented program memory (see Table 2-2 for locations). 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 CW1 and CW2 Configuration registers are shown in Table 4-2.

TABLE 4-2: PIC24FJXXXGA0XX FAMILY CONFIGURATION BITS DESCRIPTION

Bit Field	Register	Description
I2C1SEL ⁽¹⁾	CW2<2>	I2C1 Pin Mapping bit 1 = Default location for SCL1/SDA1 pins 0 = Alternate location for SCL1/SDA1 pins
DEBUG	CW1<11>	Background Debug Enable bit 1 = Device will reset in User mode 0 = Device will reset in Debug mode
FCKSM1:FCKSM0	CW2<7:6>	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
FNOSC2:FNOSC0	CW2<10:8>	Initial Oscillator Source Selection bits 111 = Internal Fast RC (FRCDIV) oscillator with postscaler 110 = Reserved 101 = Low-Power RC (LPRC) oscillator 100 = Secondary (SOSC) oscillator 011 = Primary (XTPLL, HSPLL, ECPLL) oscillator with PLL 010 = Primary (XT, HS, EC) oscillator 001 = Internal Fast RC (FRCPLL) oscillator with postscaler and PLL 000 = Fast RC (FRC) oscillator
FWDTEN	CW1<7>	Watchdog Timer Enable bit 1 = Watchdog Timer always enabled (LPRC oscillator cannot be disabled; clearing the SWDTEN bit in the RCON register will have no effect) 0 = Watchdog Timer enabled/disabled by user software (LPRC can be disabled by clearing the SWDTEN bit in the RCON register)
GCP	CW1<13>	General Segment Code-Protect bit 1 = User program memory is not code-protected 0 = User program memory is code-protected
GWRP	CW1<12>	General Segment Write-Protect bit 1 = User program memory is not write-protected 0 = User program memory is write-protected
ICS	CW1<8>	ICD Communication Channel Select bit 1 = Communicate on PGC2/EMUC2 and PGD2/EMUD2 0 = Communicate on PGC1/EMUC1 and PGD1/EMUD1

Note 1: Available on 28 and 44-pin packages only.

2: Available only on 28 and 44-pin devices with a silicon revision of 3042h or higher.

5.0 THE PROGRAMMING EXECUTIVE

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

Data transmits to the device must change on the rising edge and hold on the falling edge. Data receives from the device must change on the falling edge and hold on the rising edge.

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 FOR DATA RECEIVED FROM DEVICE

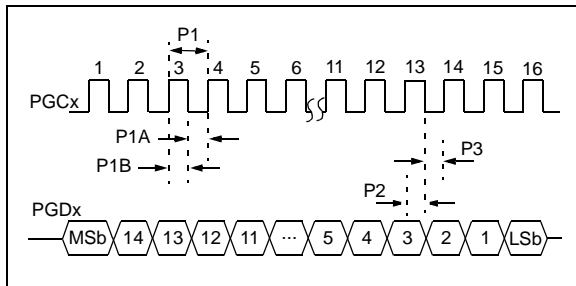
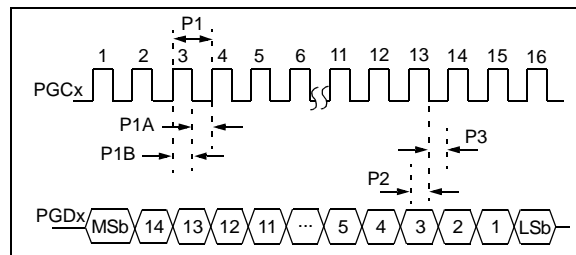


FIGURE 5-2: PROGRAMMING EXECUTIVE SERIAL TIMING FOR DATA TRANSMITTED TO DEVICE



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

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

After the entire response is clocked out, the programmer should terminate the clock on PGCx until it is time to send another command to the programming executive. This protocol is shown in Figure 5-3.

5.1.2 SPI RATE

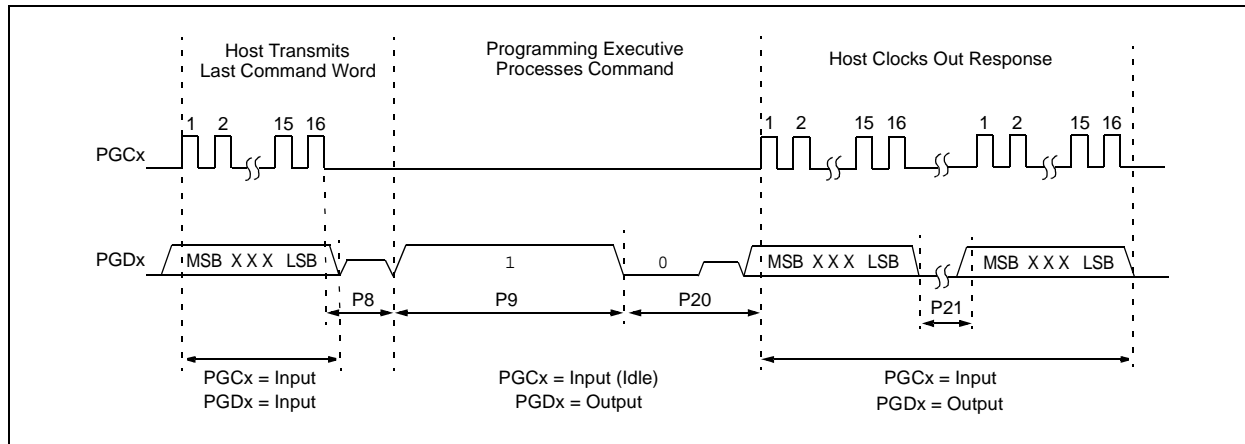
In Enhanced ICSP mode, the PIC24FJXXXGA0XX family devices operate from the internal Fast RC oscillator (FRCDIV), which has a nominal frequency of 8 MHz. This oscillator frequency yields an effective system clock frequency of 4 MHz. To ensure that the programmer does not clock too fast, it is recommended that a 4 MHz clock be provided by the programmer.

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 PGCx, as described in **Section 5.1.1 “Communication Interface and Protocol”**, it is possible that the programming executive will behave unexpectedly while trying to send a response to the programmer. Since the programming executive 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 5-3: PROGRAMMING EXECUTIVE – PROGRAMMER COMMUNICATION PROTOCOL



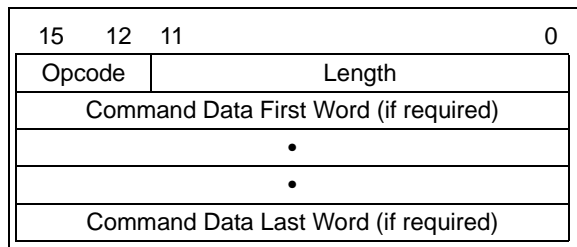
5.2 Programming Executive Commands

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 **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 5-4). The 16-bit header consists of a 4-bit opcode field, which is used to identify the command, followed by a 12-bit command length field.

FIGURE 5-4: COMMAND FORMAT



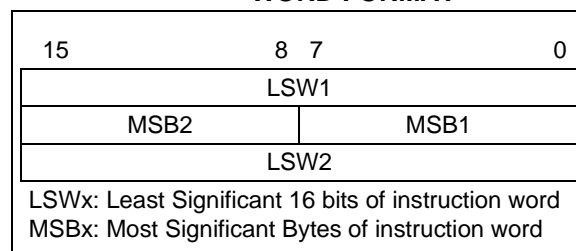
The command opcode must match one of those in the command set. Any command that is received which does not match the list in Table 5-1 will return a “NACK” response (see **Section 5.3.1.1 “Opcode Field”**).

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 5-5. This format minimizes traffic over the SPI and provides the programming executive with data that is properly aligned for performing table write operations.

FIGURE 5-5: 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”**.

PIC24FJXXXGA0XX

TABLE 5-1: PROGRAMMING EXECUTIVE COMMAND SET

Opcode	Mnemonic	Length (16-bit words)	Time-out	Description
0h	SCHECK	1	1 ms	Sanity check.
1h	READC	3	1 ms	Read an 8-bit word from the specified Device ID register.
2h	READP	4	1 ms/row	Read N 24-bit instruction words of code memory starting from the specified address.
3h	RESERVED	N/A	N/A	This command is reserved. It will return a NACK.
4h	PROGC	4	5 ms	Write an 8-bit word to the specified Device ID registers.
5h	PROGP	99	5 ms	Program one row of code memory at the specified address, then verify. ⁽¹⁾
7h	RESERVED	N/A	N/A	This command is reserved. It will return a NACK.
8h	RESERVED	N/A	N/A	This command is reserved. It will return a NACK.
9h	RESERVED	N/A	N/A	This command is reserved. It will return a NACK.
Ah	QBLANK	3	TBD	Query if the code memory is blank.
Bh	QVER	1	1 ms	Query the programming executive software version.
Dh	PROGW	4	5 ms	Program one instruction word of code memory at the specified address, then verify.

Legend: TBD = To Be Determined

Note 1: One row of code memory consists of (64) 24-bit words. Refer to Table 2-3 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 5.2.12 “QVER Command”**.

5.2.5 SCHECK COMMAND

15	12	11	0
Opcode		Length	

Field	Description
Opcode	0h
Length	1h

The SCHECK command instructs the programming executive to 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):

1000h
0002h

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

5.2.10 PROGW COMMAND

15	12	11	8	7	0
Opcode		Length			
Data_MSB			Addr_MSB		
Addr_LS					
Data_LS					

Field	Description
Opcode	Dh
Length	4h
Reserved	0h
Addr_MSB	MSB of 24-bit destination address
Addr_LS	Least Significant 16 bits of 24-bit destination address
Data_MSB	MSB of 24-bit data
Data_LS	Least Significant 16 bits of 24-bit data

The PROGW command instructs the programming executive to program one word of code memory (3 bytes) to the specific memory address.

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

Expected Response (2 words):

1600h
0002h

5.2.11 QBLANK COMMAND

15	12	11	0
Opcode	Length		
PSize_MSW			
PSize_LSW			

Field	Description
Opcode	Ah
Length	3h
PSize	Length of program memory to check in 24-bit words plus one (max. of 49152)

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

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

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

Expected Response (2 words for blank device):

1AF0h
0002h

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

1A0Fh
0002h

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

PIC24FJXXXGA0XX

5.2.12 QVER COMMAND

15	12	11	0
Opcode	Length		

Field	Description
Opcode	Bh
Length	1h

The QVER command queries the version of the programming executive software stored in 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., 23h means version 2.3 of programming executive software).

Expected Response (2 words):

1BMNh (where “MN” stands for version M.N)
0002h

5.3 Programming Executive Responses

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

The programming executive response set is 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 OP CODES

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

5.3.1 RESPONSE FORMAT

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

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

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

5.3.1.1 Opcode Field

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

5.3.1.2 Last_Cmd Field

The Last_Cmd is a 4-bit field in the first word of the response and 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	0Fh = Code memory is NOT blank F0h = Code memory is blank
QVER	0xMN, where programming executive software version = M.N (i.e., 32h 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 0h, which indicates that there was no error in the command processing. If the verify of the programming for the PROGP or PROGK command fails, the QE_Code is set to 1h. For all other programming executive errors, the QE_Code is 2h.

TABLE 5-4: QE_Code FOR NON-QUERY COMMANDS

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

5.3.1.4 Response Length

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

PIC24FJXXXGA0XX

5.4 Programming the Programming Executive to Memory

5.4.1 OVERVIEW

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

Storing the programming executive to executive memory is similar to normal programming of code memory. Namely, the executive memory must be erased, and then the programming executive must be programmed 64 words at a time. Erasing the last page of executive memory will cause the FRC oscillator calibration settings and device diagnostic data in the Diagnostic and Calibration Words, at addresses 8007F0h to 8007FEh, to be erased. In order to retain this calibration, these memory locations should be read and stored prior to erasing executive memory. They should then be reprogrammed in the last words of program memory. This control flow is summarized in Table 5-5.

TABLE 5-5: PROGRAMMING THE PROGRAMMING EXECUTIVE

Command (Binary)	Data (Hex)	Description
Step 1: Exit Reset vector and erase executive memory.		
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
Step 2: Initialize pointers to read Diagnostic and Calibration Words for storage in W6-W13.		
0000	200800	MOV #0x80, W0
0000	880190	MOV W0, TBLPAG
0000	207F00	MOV #0x07F0, W1
0000	2000C2	MOV #0xC, W2
0000	000000	NOP
Step 3: Repeat this step 8 times to read Diagnostic and Calibration Words, storing them in W registers, W6-W13.		
0000	BA1931	TBLRDL [W1++].[W2++]
0000	000000	NOP
0000	000000	NOP
Step 4: Initialize the NVMCON to erase executive memory.		
0000	240420	MOV #0x4042, W0
0000	883B00	MOV W0, NVMCON
Step 5: Initialize Erase Pointers to first page of executive and then initiate the erase cycle.		
0000	200800	MOV #0x80, W0
0000	880190	MOV W0, TBLPAG
0000	200001	MOV #0x0, W1
0000	000000	NOP
0000	BB0881	TBLWTL W1, [W1]
0000	000000	NOP
0000	000000	NOP
0000	A8E761	BSET NVMCON, #15
00000	000000	NOP
0000	000000	NOP
Step 6: Repeat this step to poll the WR bit (bit 15 of NVMCON) until it is cleared by the hardware.		
0000	040200	GOTO 0x200
0000	000000	NOP
0000	803B02	MOV NVMCON, W2
0000	883C22	MOV W2, VISI
0001	000000	NOP
	<VISI>	Clock out contents of the VISI register.
0000	000000	NOP

5.4.2 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 3.8 “Reading Code Memory”**. A procedure for reading executive memory is shown in Table 5-6. Note that in Step 2, the TBLPAG register is set to 80h, such that executive memory may be read. The last eight words of executive memory should be verified with stored values of the Diagnostic and Calibration Words to ensure accuracy.

TABLE 5-6: READING EXECUTIVE MEMORY

Command (Binary)	Data (Hex)	Description
Step 1: Exit the Reset vector.		
0000	000000	NOP
0000	040200	GOTO 0x200
0000	000000	NOP
Step 2: Initialize TBLPAG and the Read Pointer (W6) for TBLRD instruction.		
0000	200800	MOV #0x80, W0
0000	880190	MOV W0, TBLPAG
0000	EB0300	CLR W6
Step 3: Initialize the Write Pointer (W7) to point to the VISI register.		
0000	207847	MOV #VISI, W7
0000	000000	NOP
Step 4: Read and clock out the contents of the next two locations of executive memory through the VISI register using the REGOUT command.		
0000	BA0B96	TBLRDL [W6], [W7]
0000	000000	NOP
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register
0000	000000	NOP
0000	BADBB6	TBLRDH.B [W6++], [W7++]
0000	000000	NOP
0000	000000	NOP
0000	BAD3D6	TBLRDH.B [W6++], [W7--]
0000	000000	NOP
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register
0000	000000	NOP
0000	BA0BB6	TBLRDL [W6++], [W7]
0000	000000	NOP
0000	000000	NOP
0001	<VISI>	Clock out contents of VISI register
0000	000000	NOP
Step 5: Reset the device internal PC.		
0000	040200	GOTO 0x200
0000	000000	NOP
Step 6: Repeat Steps 4 and 5 until all desired code memory is 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.

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


FilterLab, Linear Active Thermistor, MXDEV, MXLAB, SEEVAL, SmartSensor 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, CodeGuard, dsPICDEM, dsPICDEM.net, dsPICworks, dsSPEAK, ECAN, ECONOMONITOR, FanSense, In-Circuit Serial Programming, ICSP, ICEPIC, Mindi, MiWi, MPASM, MPLAB Certified logo, MPLIB, MPLINK, mTouch, PICkit, PICDEM, PICDEM.net, PICtail, PIC³² logo, PowerCal, PowerInfo, PowerMate, PowerTool, REAL ICE, rfLAB, Select Mode, Total Endurance, 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.

© 2008, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

 Printed on recycled paper.

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.