



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.

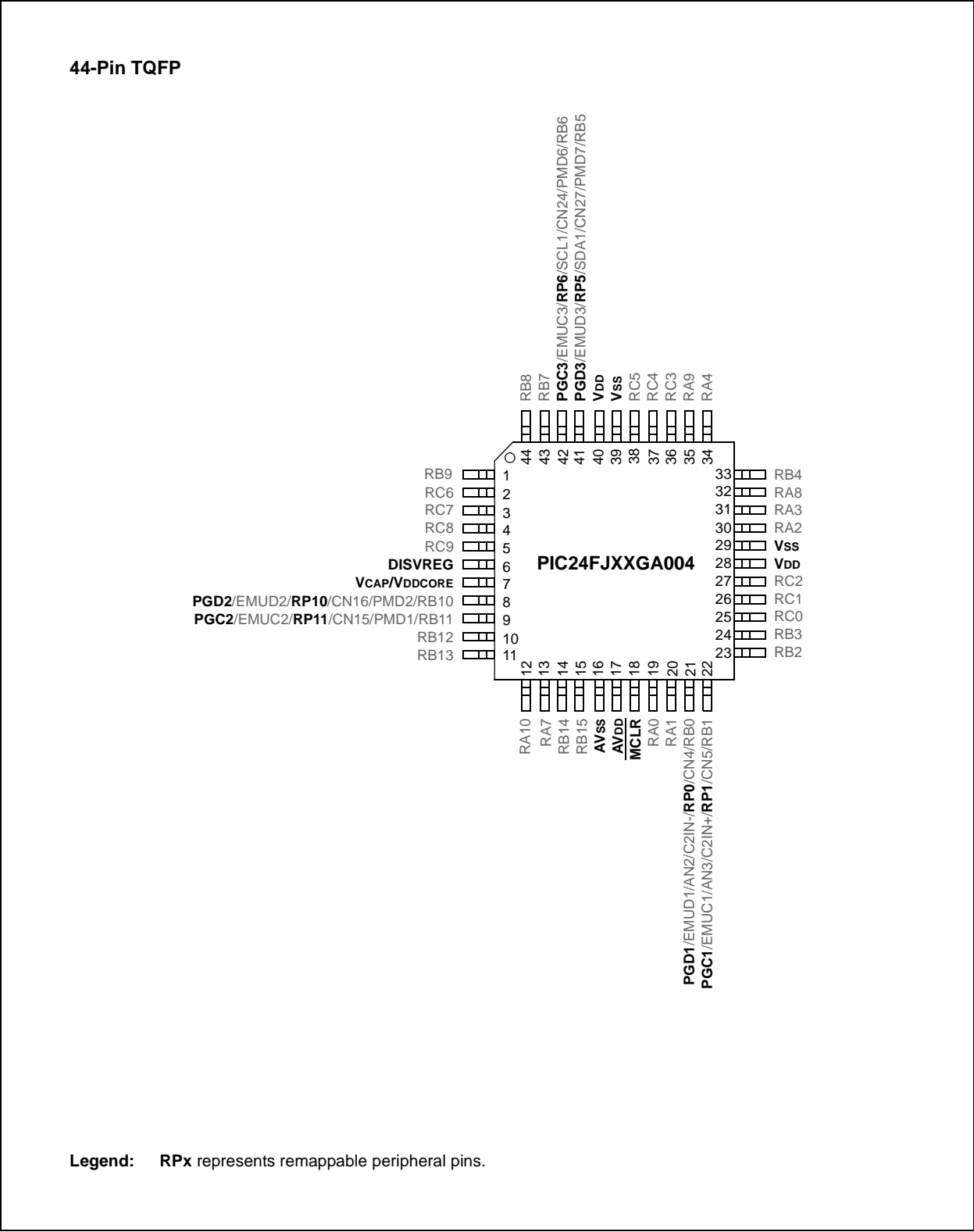
Applications of "[Embedded - Microcontrollers](#)"

Details

| | |
|----------------------------|---|
| Product Status | Active |
| Core Processor | PIC |
| Core Size | 16-Bit |
| Speed | 16MHz |
| Connectivity | I ² C, SPI, UART/USART |
| Peripherals | Brown-out Detect/Reset, POR, PWM, WDT |
| Number of I/O | 84 |
| 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 | 100-TQFP |
| Supplier Device Package | 100-TQFP (12x12) |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/pic24fj128ga010t-i-pt |

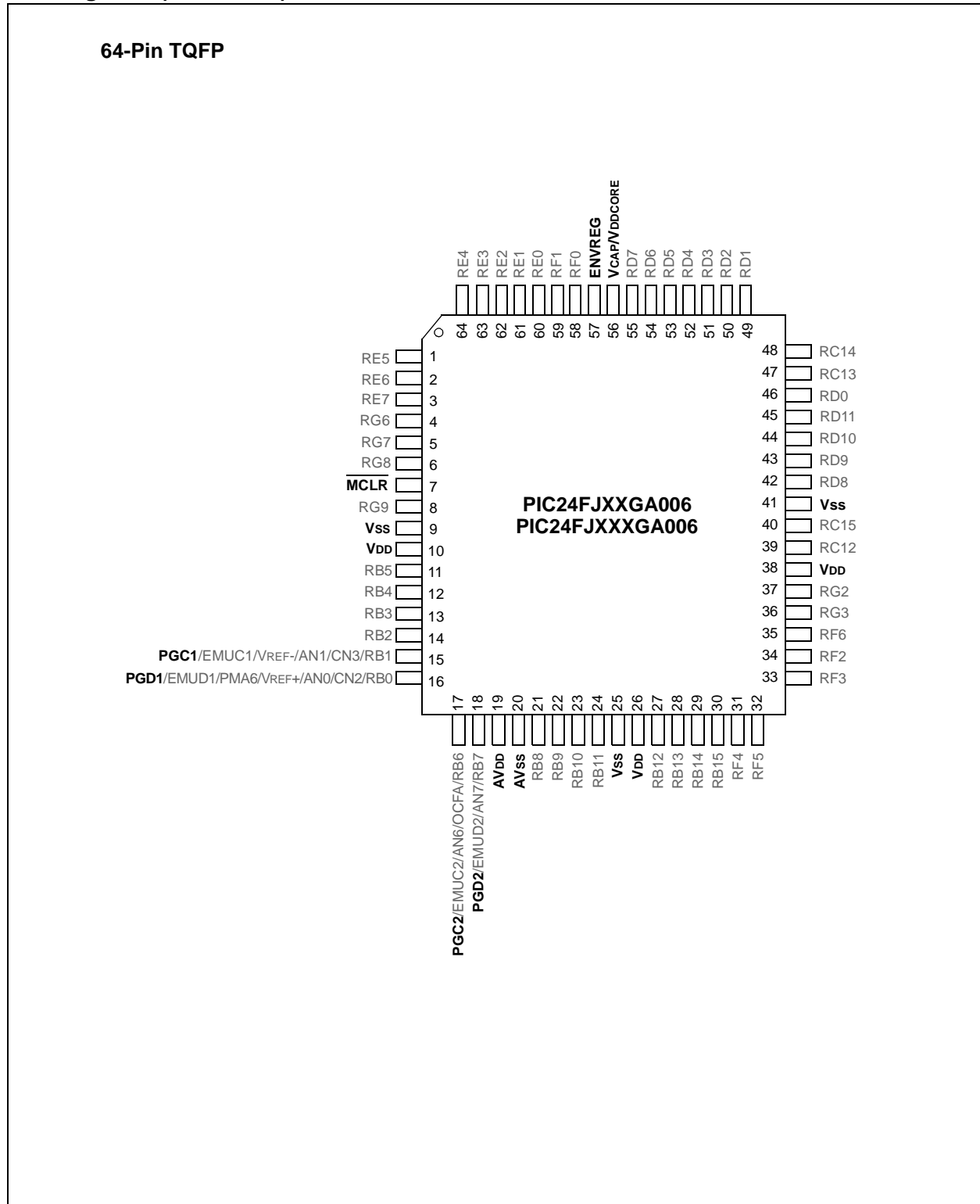
PIC24FJXXXGA0XX

Pin Diagrams (Continued)



PIC24FJXXXGA0XX

Pin Diagrams (Continued)



PIC24FJXXXGA0XX

Pin Diagrams (Continued)

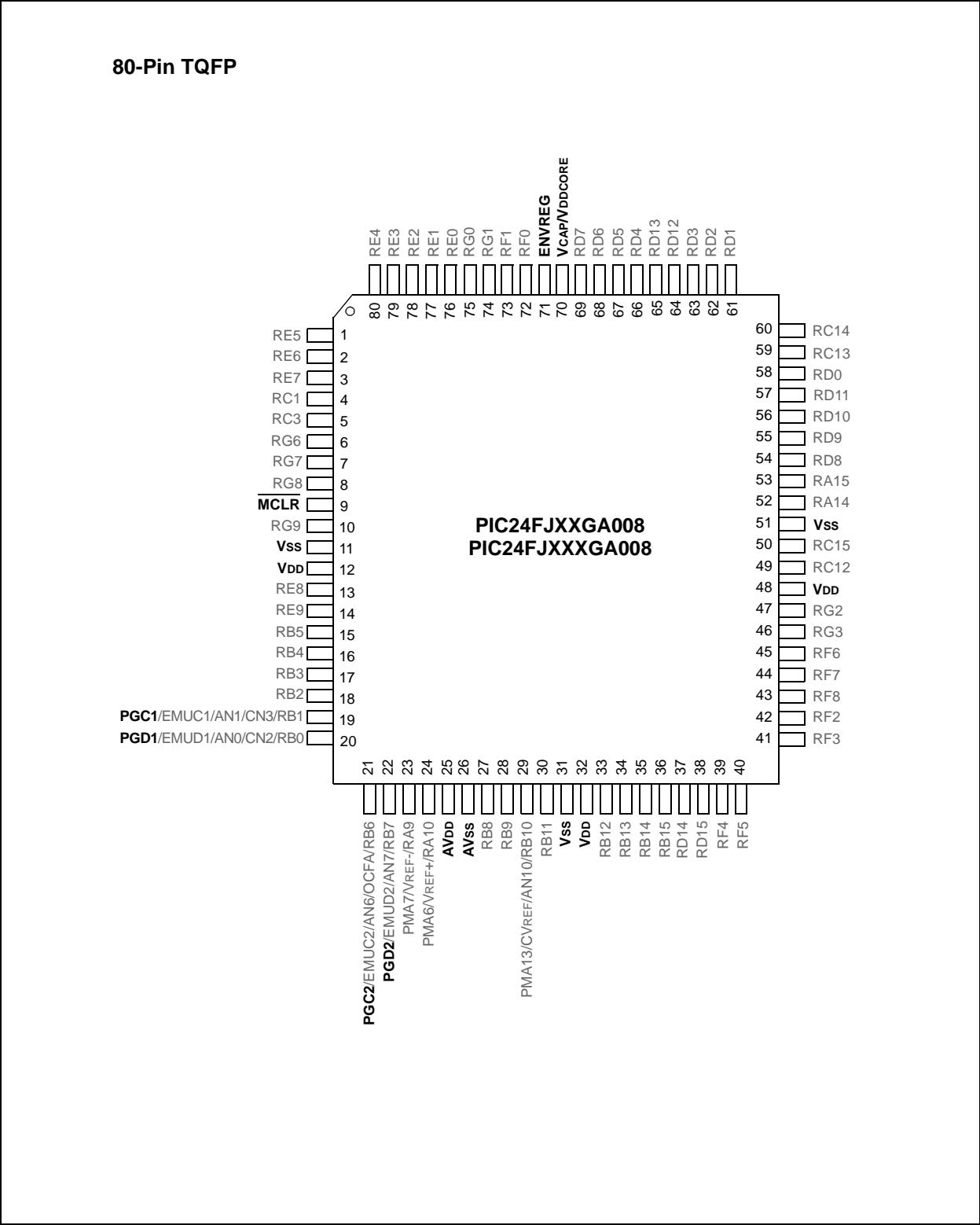
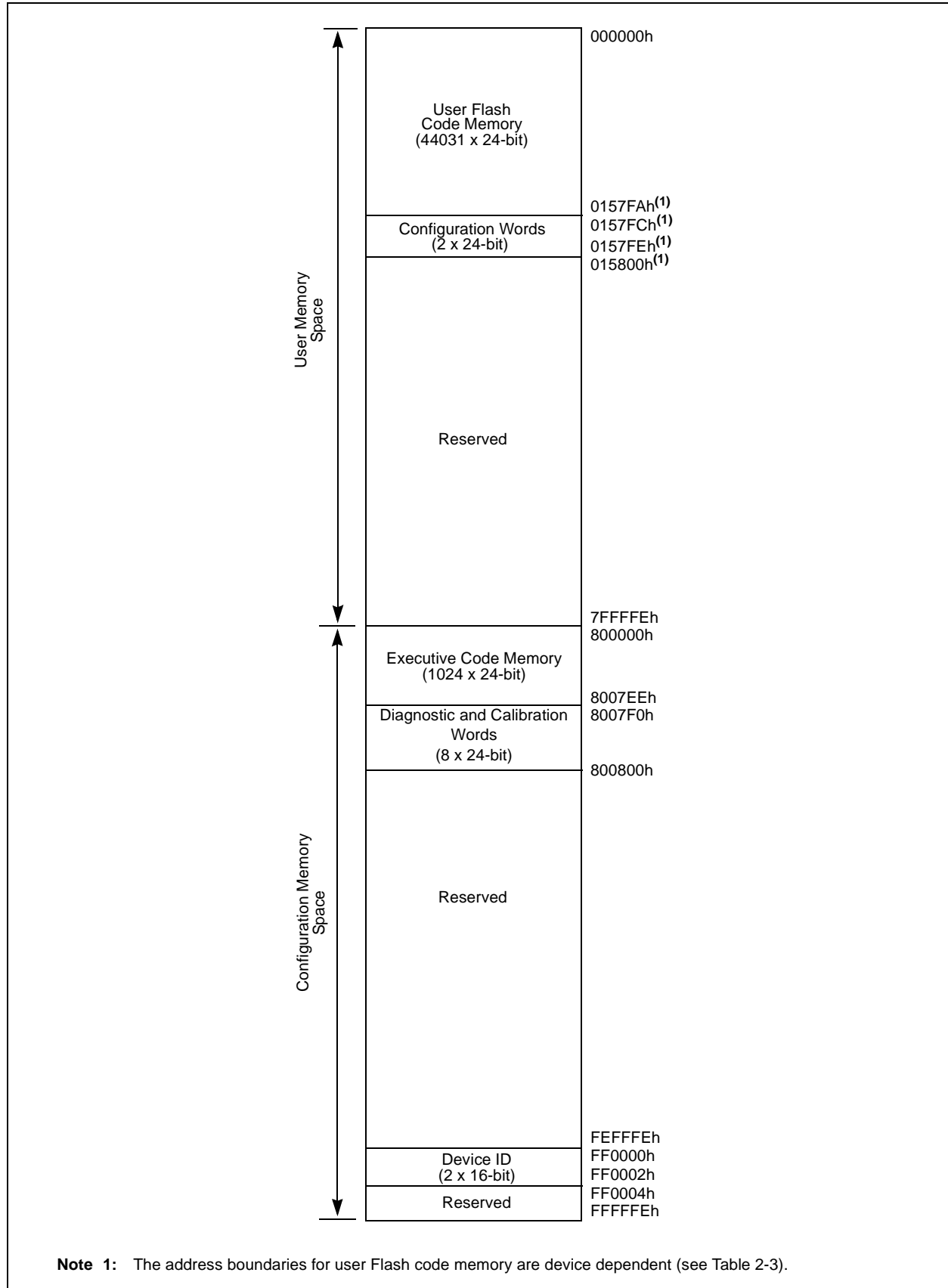


FIGURE 2-4: PROGRAM MEMORY MAP



3.2.1 SIX SERIAL INSTRUCTION EXECUTION

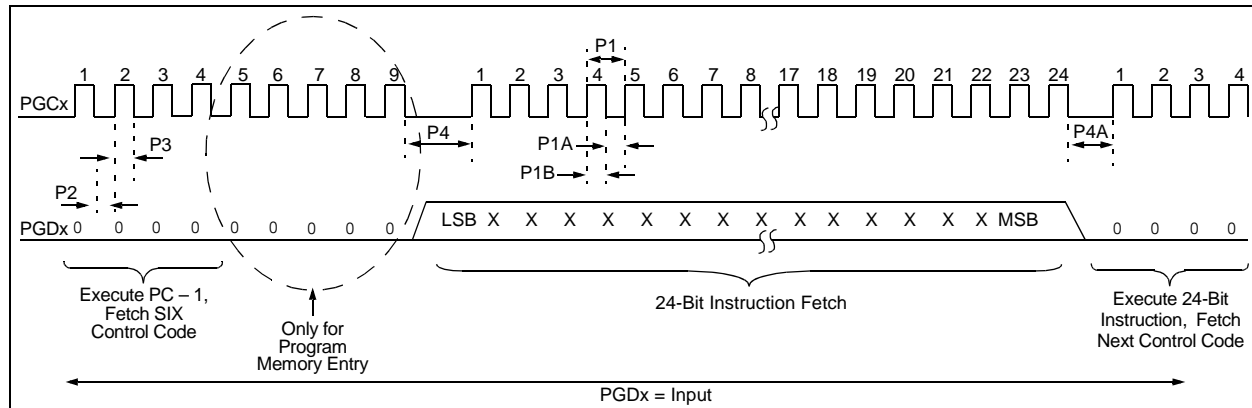
The SIX control code allows execution of the PIC24FJXXXGA0XX family assembly instructions. When the SIX code is received, the CPU is suspended for 24 clock cycles, as the instruction is then clocked into the internal buffer. Once the instruction is shifted in, the state machine allows it to be executed over the next four PGC clock cycles. While the received instruction is executed, the state machine simultaneously shifts in the next 4-bit command (see Figure 3-2).

Coming out of Reset, the first 4-bit control code is always forced to SIX and a forced NOP instruction is executed by the CPU. Five additional PGCx clocks are needed on start-up, resulting in a 9-bit SIX command instead of the normal 4-bit SIX command.

After the forced SIX is clocked in, ICSP operation resumes as normal. That is, the next 24 clock cycles load the first instruction word to the CPU.

Note: To account for this forced NOP, all example code in this specification begin with a NOP to ensure that no data is lost.

FIGURE 3-2: SIX SERIAL EXECUTION



3.2.1.1 Differences Between Execution of SIX and Normal Instructions

There are some differences between executing instructions normally and using the SIX ICSP command. As a result, the code examples in this specification may not match those for performing the same functions during normal device operation.

The important differences are:

- Two-word instructions require two SIX operations to clock in all the necessary data.
Examples of two-word instructions are `GOTO` and `CALL`.
- Two-cycle instructions require two SIX operations.
The first SIX operation shifts in the instruction and begins to execute it. A second SIX operation – which should shift in a NOP to avoid losing data – provides the CPU clocks required to finish executing the instruction.
Examples of two-cycle instructions are table read and table write instructions.
- The CPU does not automatically stall to account for pipeline changes.
A CPU stall occurs when an instruction modifies a register that is used for Indirect Addressing by the following instruction.

During normal operation, the CPU automatically will force a NOP while the new data is read. When using ICSP, there is no automatic stall, so any indirect references to a recently modified register should be preceded by a NOP.

For example, the instructions, `mov #0x0, W0` and `mov [W0], W1`, must have a NOP inserted between them.

If a two-cycle instruction modifies a register that is used indirectly, it will require two following NOPs: one to execute the second half of the instruction and a second to stall the CPU to correct the pipeline.

Instructions such as `tblwtl [W0++], [W1]` should be followed by two NOPs.

- The device Program Counter (PC) continues to automatically increment during ICSP instruction execution, even though the Flash memory is not being used.

As a result, the PC may be incremented to point to invalid memory locations. Invalid memory spaces include unimplemented Flash addresses and the vector space (locations 0x0 to 0x1FF).

If the PC points to these locations, the device will reset, possibly interrupting the ICSP operation. To prevent this, instructions should be periodically executed to reset the PC to a safe space. The optimal method to accomplish this is to perform a `GOTO 0x200`.

PIC24FJXXXGA0XX

3.4 Flash Memory Programming in ICSP Mode

3.4.1 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 3-2) or write operation (Table 3-3) and initiating the programming by setting the WR control bit (NVMCON<15>).

In ICSP mode, all programming operations are self-timed. There is an internal delay between the user setting the WR control bit and the automatic clearing of the WR control bit when the programming operation is complete. Please refer to **Section 7.0 “AC/DC Characteristics and Timing Requirements”** for information about the delays associated with various programming operations.

TABLE 3-2: NVMCON ERASE OPERATIONS

| NVMCON Value | Erase Operation |
|--------------|--|
| 404Fh | Erase all code memory, executive memory and Configuration registers (does not erase Unit ID or Device ID registers). |
| 4042h | Erase a page of code memory or executive memory. |

TABLE 3-3: NVMCON WRITE OPERATIONS

| NVMCON Value | Write Operation |
|--------------|--|
| 4003h | Write a Configuration Word register. |
| 4001h | Program 1 row (64 instruction words) of code memory or executive memory. |

3.4.2 STARTING AND STOPPING A PROGRAMMING CYCLE

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

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

```
BSET    NVMCON, #WR
```

3.5 Erasing Program Memory

The procedure for erasing program memory (all of code memory, data memory, executive memory and code-protect bits) consists of setting NVMCON to 404Fh and executing the programming cycle.

A Chip Erase can erase all of user memory or all of both the user and configuration memory. A table write instruction should be executed prior to performing the Chip Erase to select which sections are erased.

When this table write instruction is executed:

- If the TBLPAG register points to user space (is less than 0x80), the Chip Erase will erase only user memory.
- If TBLPAG points to configuration space (is greater than or equal to 0x80), the Chip Erase will erase both user and configuration memory.

If configuration memory is erased, the internal oscillator Calibration Word, located at 0x807FE, will be erased. This location should be stored prior to performing a whole Chip Erase and restored afterward to prevent internal oscillators from becoming uncalibrated.

Figure 3-5 shows the ICSP programming process for performing a Chip Erase. This process includes the ICSP command code, which must be transmitted (for each instruction), Least Significant bit first, using the PGCx and PGDx pins (see Figure 3-2).

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

FIGURE 3-5: CHIP ERASE FLOW

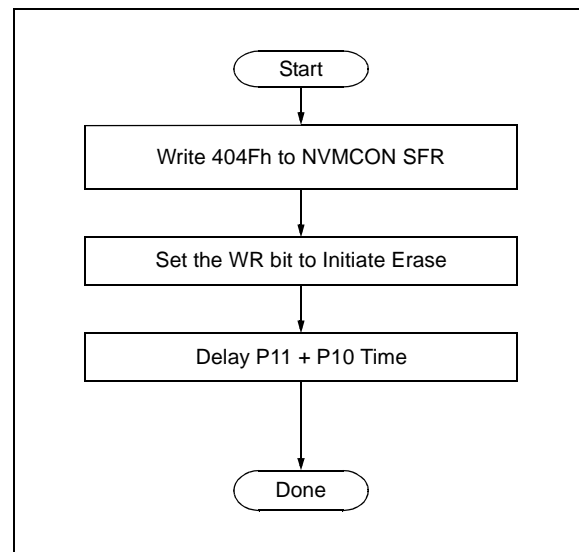


TABLE 3-4: SERIAL INSTRUCTION EXECUTION FOR CHIP ERASE

| 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 erase all program memory. | | |
| 0000 | 2404FA | MOV #0x404F, W10 |
| 0000 | 883B0A | MOV W10, NVMCON |
| Step 3: Set TBLPAG and perform dummy table write to select what portions of memory are erased. | | |
| 0000 | 200000 | MOV #<PAGEVAL>, W0 |
| 0000 | 880190 | MOV W0, TBLPAG |
| 0000 | 200000 | MOV #0x0000, W0 |
| 0000 | BB0800 | TBLWTL W0, [W0] |
| 0000 | 000000 | NOP |
| 0000 | 000000 | NOP |
| Step 4: Initiate the erase cycle. | | |
| 0000 | A8E761 | BSET NVMCON, #WR |
| 0000 | 000000 | NOP |
| 0000 | 000000 | NOP |
| Step 5: 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 |

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>

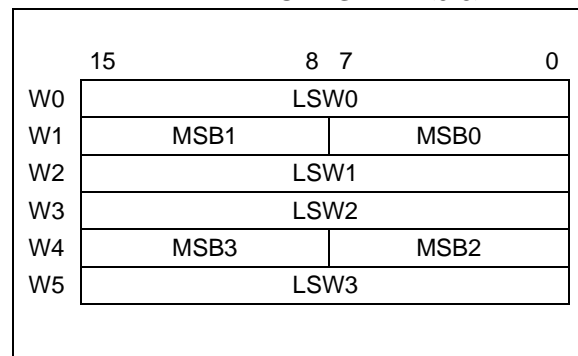
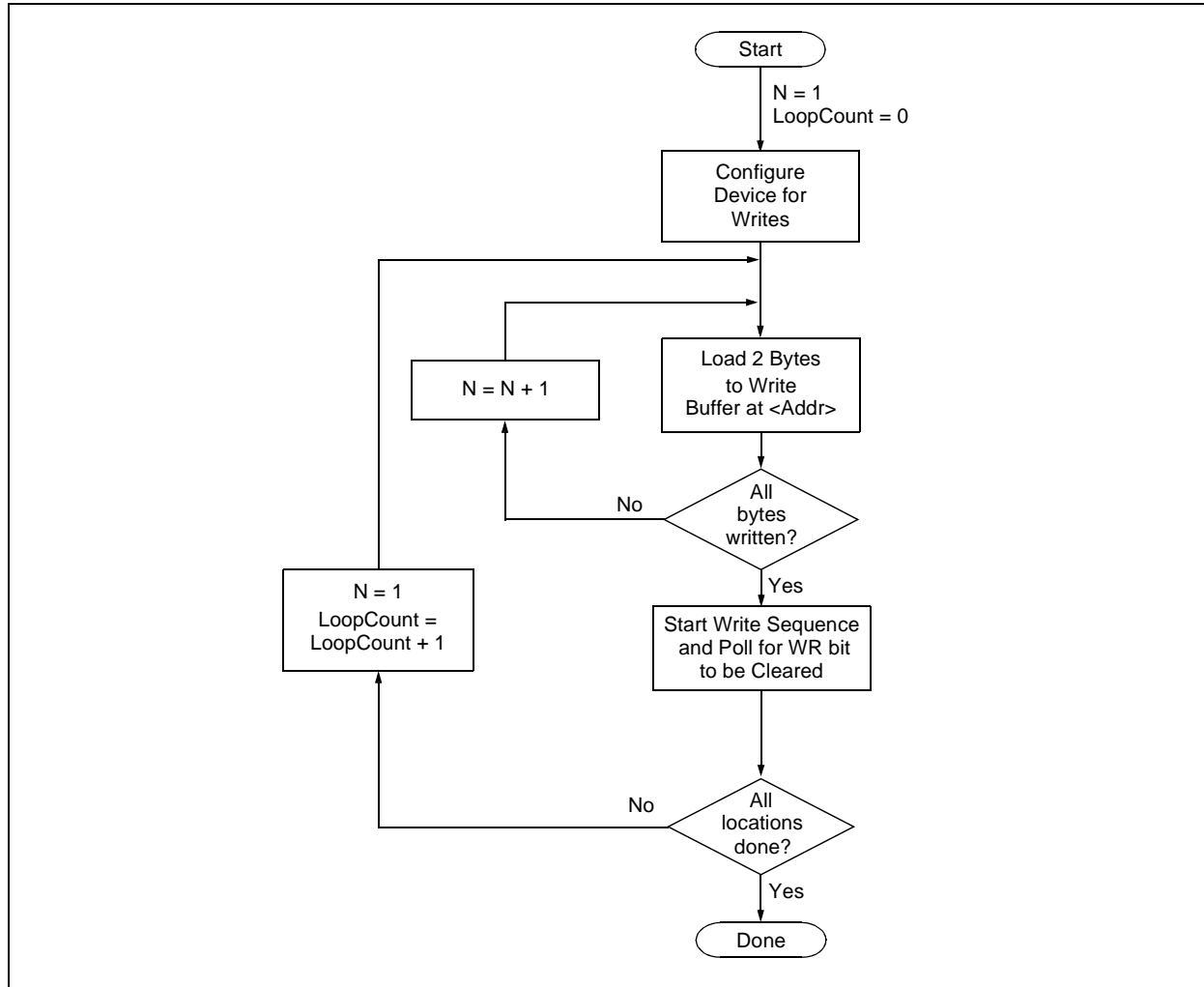


TABLE 3-5: SERIAL INSTRUCTION EXECUTION FOR WRITING CODE MEMORY

| 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



4.0 DEVICE PROGRAMMING – ENHANCED ICSP

This section discusses programming the device through Enhanced ICSP and the programming executive. The programming executive resides in executive memory (separate from code memory) and is executed when Enhanced ICSP Programming mode is entered. The programming executive provides the mechanism for the programmer (host device) to program and verify the PIC24FJXXXGA0XX devices using a simple command set and communication protocol. There are several basic functions provided by the programming executive:

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

The programming executive performs the low-level tasks required for erasing, programming and verifying a device. This allows the programmer to program the device by issuing the appropriate commands and data. Table 4-1 summarizes the commands. A detailed description for each command is provided in **Section 5.2 “Programming Executive Commands”**.

TABLE 4-1: COMMAND SET SUMMARY

| Command | Description |
|---------|--|
| SCHECK | Sanity Check |
| READC | Read Device ID Registers |
| READP | Read Code Memory |
| PROGP | Program One Row of Code Memory and Verify |
| PROGW | Program One Word of Code Memory and Verify |
| QBLANK | Query if the Code Memory is Blank |
| QVER | Query the Software Version |

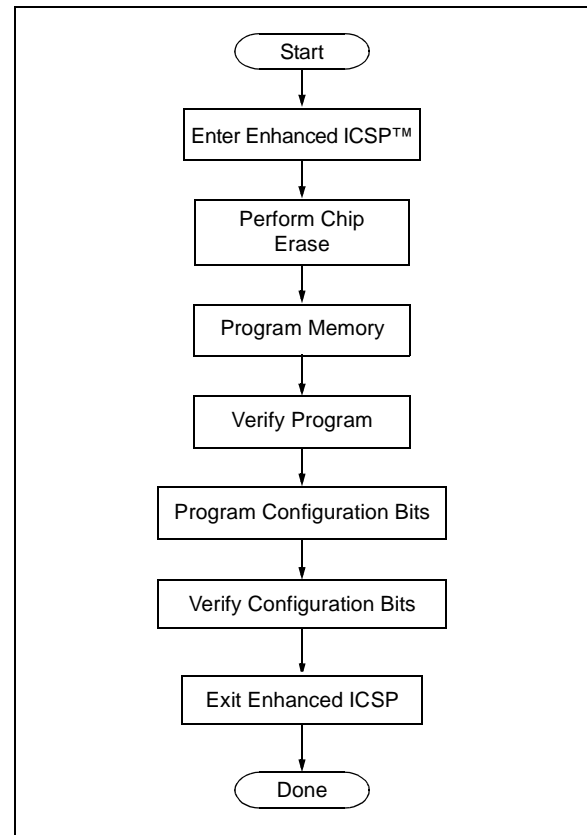
The programming executive uses the device's data RAM for variable storage and program execution. After the programming executive has run, no assumptions should be made about the contents of data RAM.

4.1 Overview of the Programming Process

Figure 4-1 shows the high-level overview of the programming process. After entering Enhanced ICSP mode, the programming executive is verified. Next, the device is erased. Then, the code memory is programmed, followed by the configuration locations. Code memory (including the Configuration registers) is then verified to ensure that programming was successful.

After the programming executive has been verified in memory (or loaded if not present), the PIC24FJXXXGA0XX family can be programmed using the command set shown in Table 4-1.

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



4.2 Confirming the Presence of the Programming Executive

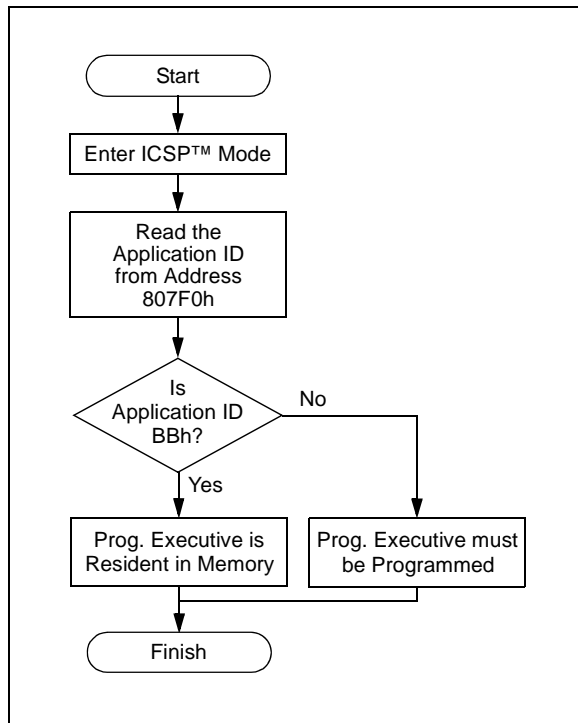
Before programming can begin, the programmer must confirm that the programming executive is stored in executive memory. The procedure for this task is shown in Figure 4-2.

First, In-Circuit Serial Programming mode (ICSP) is entered. Then, the unique Application ID Word stored in executive memory is read. If the programming executive is resident, the Application ID Word is BBh, which means programming can resume as normal. However, if the Application ID Word is not BBh, the programming executive must be programmed to executive code memory using the method described in **Section 5.4 “Programming the Programming Executive to Memory”**.

Section 3.0 “Device Programming – ICSP” describes the ICSP programming method. **Section 3.11 “Reading the Application ID Word”** describes the procedure for reading the Application ID Word in ICSP mode.

PIC24FJXXXGA0XX

FIGURE 4-2: CONFIRMING PRESENCE OF PROGRAMMING EXECUTIVE



4.3 Entering Enhanced ICSP Mode

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

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

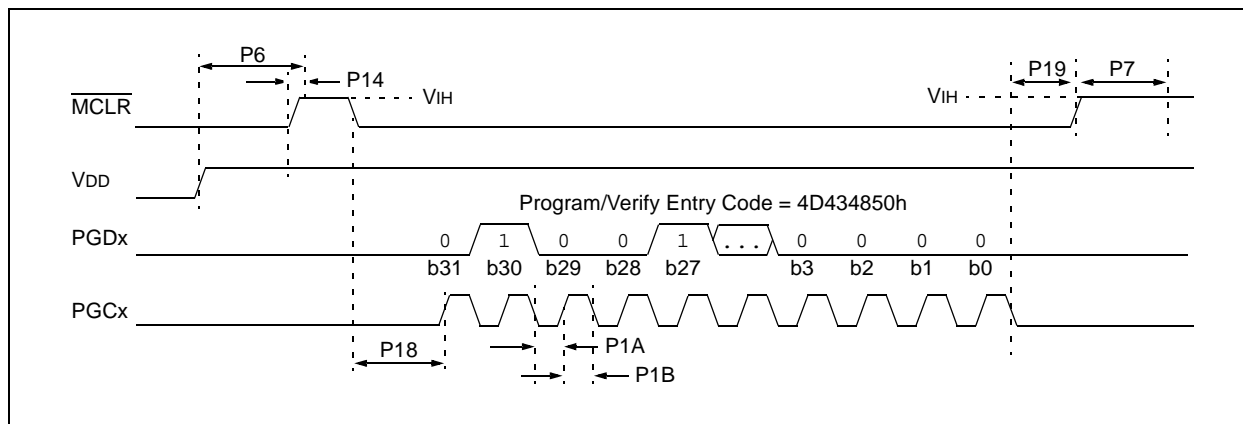
The programming voltage applied to $\overline{\text{MCLR}}$ is V_{IH} , which is essentially V_{DD} in the case of PIC24FJXXXGA0XX devices. There is no minimum time requirement for holding at V_{IH} . After V_{IH} is removed, an interval of at least P18 must elapse before presenting the key sequence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

The key sequence is a specific 32-bit pattern: '0100 1101 0100 0011 0100 1000 0101 0000' (more easily remembered as 4D434850h in hexadecimal format). The device will enter Program/Verify mode only if the key sequence is valid. The Most Significant bit (MSb) of the most significant nibble must be shifted in first.

Once the key sequence is complete, 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 PGDx 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 the Program/Verify mode, all unused I/Os are placed in the high-impedance state.

FIGURE 4-3: ENTERING ENHANCED ICSP™ MODE



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

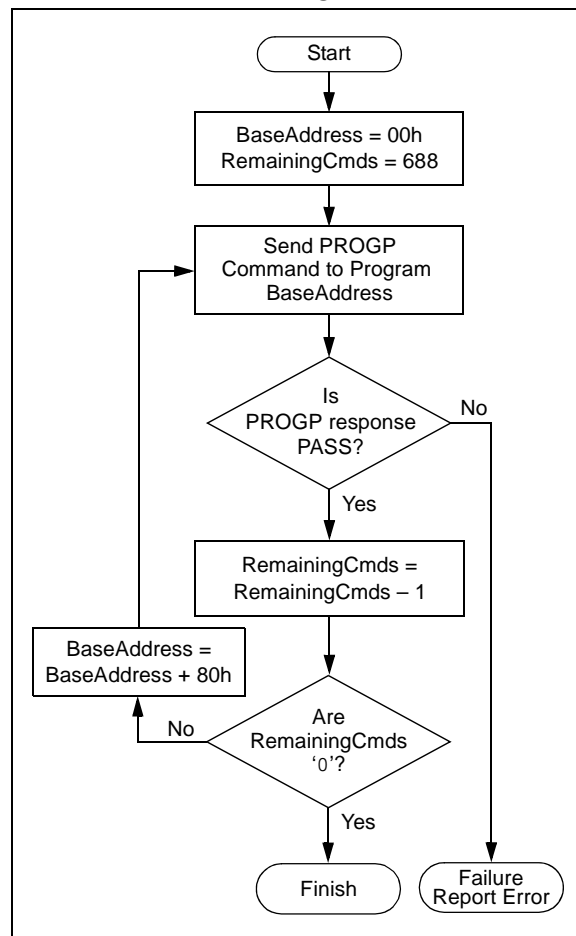
4.5.1 PROGRAMMING METHODOLOGY

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

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.

PIC24FJXXXGA0XX

5.2.8 PROGC COMMAND

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

| Field | Description |
|----------|---|
| Opcode | 4h |
| Length | 4h |
| Reserved | 0h |
| Addr_MSB | MSB of 24-bit destination address |
| Addr_LS | Least Significant 16 bits of 24-bit destination address |
| Data | 8-bit data word |

The PROGC command instructs the programming executive to program a single Device ID register located at the specified memory address.

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

Expected Response (2 words):

1400h
0002h

5.2.9 PROGP COMMAND

| | | | | | |
|----------|----|--------|----------|---|---|
| 15 | 12 | 11 | 8 | 7 | 0 |
| Opcode | | Length | | | |
| Reserved | | | Addr_MSB | | |
| Addr_LS | | | | | |
| D_1 | | | | | |
| D_2 | | | | | |
| ... | | | | | |
| D_96 | | | | | |

| Field | Description |
|----------|---|
| Opcode | 5h |
| Length | 63h |
| Reserved | 0h |
| Addr_MSB | MSB of 24-bit destination address |
| Addr_LS | Least Significant 16 bits of 24-bit destination address |
| D_1 | 16-bit data word 1 |
| D_2 | 16-bit data word 2 |
| ... | 16-bit data word 3 through 95 |
| D_96 | 16-bit data word 96 |

The PROGP command instructs the programming executive to program one row of code memory, including Configuration Words (64 instruction words), to the specified memory address. Programming begins with the row address specified in the command. The destination address should be a multiple of 80h.

The data to program to memory, located in command words, D_1 through D_96, must be arranged using the packed instruction word format shown in Figure 5-5.

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):

1500h
0002h

Note: Refer to Table 2-3 for code memory size information.

TABLE 5-5: PROGRAMMING THE PROGRAMMING EXECUTIVE (CONTINUED)

| Command (Binary) | Data (Hex) | Description |
|--|---------------|--------------------------------------|
| Step 7: Repeat Steps 5 and 6 to erase the second page of executive memory. The W1 Pointer should be incremented by 400h to point to the second page. | | |
| Step 8: Initialize TBLPAG and NVMCON to write stored diagnostic and calibration as single words. Initialize W1 and W2 as Write and Read Pointers to rewrite stored Diagnostic and Calibration Words. | | |
| 0000 | 200800 | MOV #0x80, W0 |
| 0000 | 880190 | MOV W0, TBLPAG |
| 0000 | 240031 | MOV #0x4003, W1 |
| 0000 | 883B01 | MOV W1, NVMCON |
| 0000 | 207F00 | MOV #0x07F0, W1 |
| 0000 | 2000C2 | MOV #0xC, W2 |
| 0000 | 000000 | NOP |
| Step 9: Perform write of a single word of calibration data and initiate single-word write cycle. | | |
| 0000 | BB18B2 | TBLWTL [W2++], [W1++] |
| 0000 | 000000 | NOP |
| 0000 | 000000 | NOP |
| 0000 | A8E761 | BSET NVMCON, #15 |
| 0000 | 000000 | NOP |
| 0000 | 000000 | NOP |
| Step 10: 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 | 803B00 | MOV NVMCON, W0 |
| 0000 | 883C20 | MOV W0, VISI |
| 0000 | 000000 | NOP |
| 0001 | <VISI> | Clock out contents of VISI register. |
| 0000 | 000000 | NOP |
| Step 11: Repeat steps 9-10 seven more times to program the remainder of the Diagnostic and Calibration Words back into program memory. | | |
| Step 12: Initialize the NVMCON to program 64 instruction words. | | |
| 0000 | 240010 | MOV #0x4001, W0 |
| 0000 | 883B00 | MOV W0, NVMCON |
| Step 13: Initialize TBLPAG and the Write Pointer (W7). | | |
| 0000 | 200800 | MOV #0x80, W0 |
| 0000 | 880190 | MOV W0, TBLPAG |
| 0000 | EB0380 | CLR W7 |
| 0000 | 000000 | NOP |
| Step 14: Load W0:W5 with the next four words of packed programming executive code and initialize W6 for programming. Programming starts from the base of executive memory (800000h) 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 |

PIC24FJXXXGA0XX

TABLE 5-5: PROGRAMMING THE PROGRAMMING EXECUTIVE (CONTINUED)

| Command (Binary) | Data (Hex) | Description |
|--|---------------|--|
| Step 15: Set the Read Pointer (W6) and load the (next four 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 | BEBB6 | 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 | BEBB6 | TBLWTH.B [W6++], [++W7] |
| 0000 | 000000 | NOP |
| 0000 | 000000 | NOP |
| 0000 | BB1BB6 | TBLWTL [W6++], [W7++] |
| 0000 | 000000 | NOP |
| 0000 | 000000 | NOP |
| Step 16: Repeat Steps 14-15, sixteen times, to load the write latches for the 64 instructions. | | |
| Step 17: Initiate the programming cycle. | | |
| 0000 | A8E761 | BSET NVMCON, #15 |
| 0000 | 000000 | NOP |
| 0000 | 000000 | NOP |
| Step 18: 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 19: Reset the device internal PC. | | |
| 0000 | 040200 | GOTO 0x200 |
| 0000 | 000000 | NOP |
| Step 20: Repeat Steps 14-19 until all 16 rows of executive memory have been programmed. On the final row, make sure to initialize the write latches at the Diagnostic and Calibration Words locations with 0xFFFFF to ensure that the calibration is not overwritten. | | |

7.0 AC/DC CHARACTERISTICS AND TIMING REQUIREMENTS

| Standard Operating Conditions | | | | | | |
|---|--------------------|--|---------------------|---------------------|-------|-------------------------------------|
| Operating Temperature: 0°C to +70°C. Programming at +25°C is recommended. | | | | | | |
| Param No. | Symbol | Characteristic | Min | Max | Units | Conditions |
| D111 | VDD | Supply Voltage During Programming | VDDCORE + 0.1 | 3.60 | V | Normal programming ^(1,2) |
| D112 | I _{PP} | Programming Current on $\overline{\text{MCLR}}$ | — | 5 | μA | |
| D113 | I _{DDP} | Supply Current During Programming | — | 2 | mA | |
| D031 | V _{IL} | Input Low Voltage | V _{SS} | 0.2 V _{DD} | V | |
| D041 | V _{IH} | Input High Voltage | 0.8 V _{DD} | V _{DD} | V | |
| D080 | V _{OL} | Output Low Voltage | — | 0.4 | V | I _{OL} = 8.5 mA @ 3.6V |
| D090 | V _{OH} | Output High Voltage | 3.0 | — | V | I _{OH} = -3.0 mA @ 3.6V |
| D012 | C _{IO} | Capacitive Loading on I/O pin (PGDx) | — | 50 | pF | To meet AC specifications |
| D013 | C _F | Filter Capacitor Value on VCAP | 4.7 | 10 | μF | Required for controller core |
| P1 | T _{PGC} | Serial Clock (PGCx) Period | 100 | — | ns | |
| P1A | T _{PGCL} | Serial Clock (PGCx) Low Time | 40 | — | ns | |
| P1B | T _{PGCH} | Serial Clock (PGCx) High Time | 40 | — | ns | |
| P2 | T _{SET1} | Input Data Setup Time to Serial Clock ↑ | 15 | — | ns | |
| P3 | T _{HLD1} | Input Data Hold Time from PGCx ↑ | 15 | — | ns | |
| P4 | T _{DLY1} | Delay Between 4-Bit Command and Command Operand | 40 | — | ns | |
| P4A | T _{DLY1A} | Delay Between 4-Bit Command Operand and Next 4-Bit Command | 40 | — | ns | |
| P5 | T _{DLY2} | Delay Between Last PGCx ↓ of Command Byte to First PGCx ↑ of Read of Data Word | 20 | — | ns | |
| P6 | T _{SET2} | V _{DD} ↑ Setup Time to $\overline{\text{MCLR}}$ ↑ | 100 | — | ns | |
| P7 | T _{HLD2} | Input Data Hold Time from $\overline{\text{MCLR}}$ ↑ | 25 | — | ms | |
| P8 | T _{DLY3} | Delay Between Last PGCx ↓ of Command Byte to PGDx ↑ by Programming Executive | 12 | — | μs | |
| P9 | T _{DLY4} | Programming Executive Command Processing Time | 40 | — | μs | |
| P10 | T _{DLY6} | PGCx Low Time After Programming | 400 | — | ns | |
| P11 | T _{DLY7} | Chip Erase Time | 400 | — | ms | |
| P12 | T _{DLY8} | Page Erase Time | 40 | — | ms | |
| P13 | T _{DLY9} | Row Programming Time | 2 | — | ms | |
| P14 | T _R | $\overline{\text{MCLR}}$ Rise Time to Enter ICSP™ mode | — | 1.0 | μs | |
| P15 | T _{VALID} | Data Out Valid from PGCx ↑ | 10 | — | ns | |
| P16 | T _{DLY10} | Delay Between Last PGCx ↓ and $\overline{\text{MCLR}}$ ↓ | 0 | — | s | |
| P17 | T _{HLD3} | $\overline{\text{MCLR}}$ ↓ to V _{DD} ↓ | 100 | — | ns | |
| P18 | T _{KEY1} | Delay from First $\overline{\text{MCLR}}$ ↓ to First PGCx ↑ for Key Sequence on PGDx | 40 | — | ns | |
| P19 | T _{KEY2} | Delay from Last PGCx ↓ for Key Sequence on PGDx to Second $\overline{\text{MCLR}}$ ↑ | 1 | — | ms | |
| P20 | T _{DLY11} | Delay Between PGDx ↓ by Programming Executive to PGDx Driven by Host | 23 | — | μs | |
| P21 | T _{DLY12} | Delay Between Programming Executive Command Response Words | 8 | — | ns | |

Note 1: VDDCORE must be supplied to the VDDCORE/VCAP pin if the on-chip voltage regulator is disabled. See **Section 2.1 “Power Requirements”** for more information. (Minimum VDDCORE allowing Flash programming is 2.25V.)

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

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.



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

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 - 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 - Xiamen

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

China - Xian

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

China - Zhuhai

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

ASIA/PACIFIC

India - Bangalore

Tel: 91-80-4182-8400
Fax: 91-80-4182-8422

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: 886-3-572-9526
Fax: 886-3-572-6459

Taiwan - Kaohsiung

Tel: 886-7-536-4818
Fax: 886-7-536-4803

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