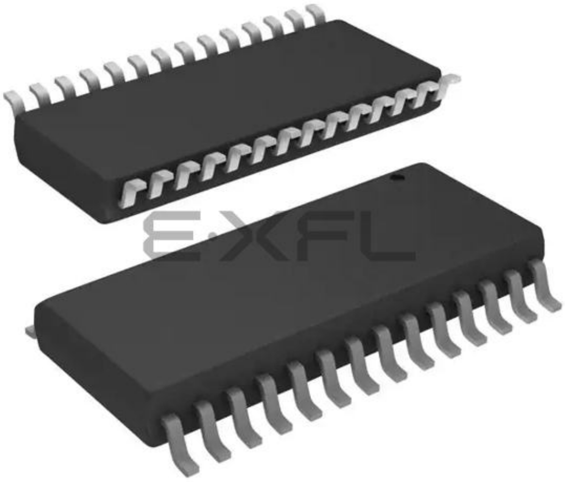


What is "Embedded - Microcontrollers"?

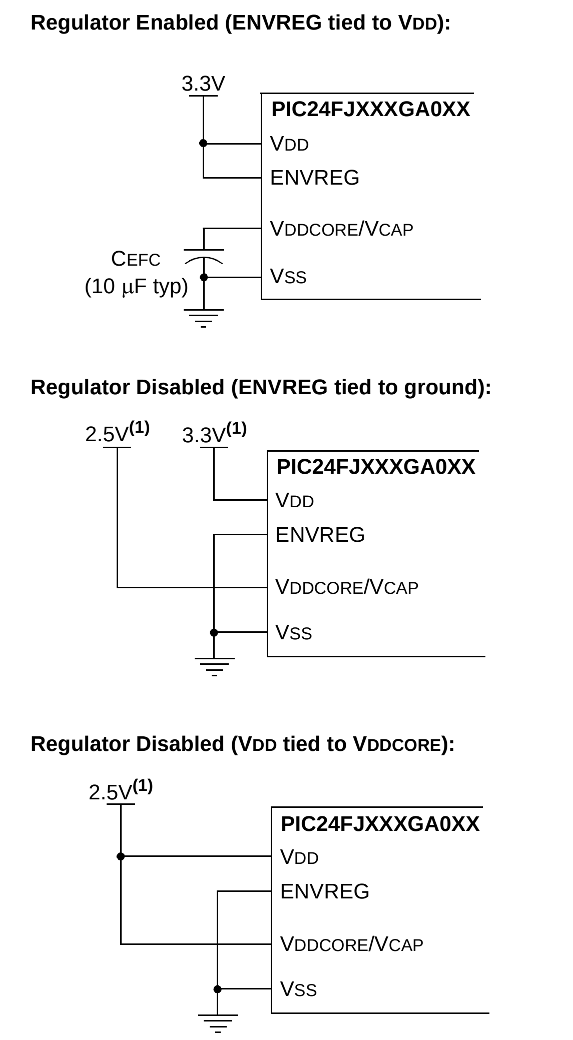
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" 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	32MHz
Connectivity	I²C, PMP, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	21
Program Memory Size	32KB (11K x 24)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	8K x 8
Voltage - Supply (Vcc/Vdd)	2V ~ 3.6V
Data Converters	A/D 10x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	28-SOIC (0.295", 7.50mm Width)
Supplier Device Package	28-SOIC
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic24fj32ga002-e-so

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)

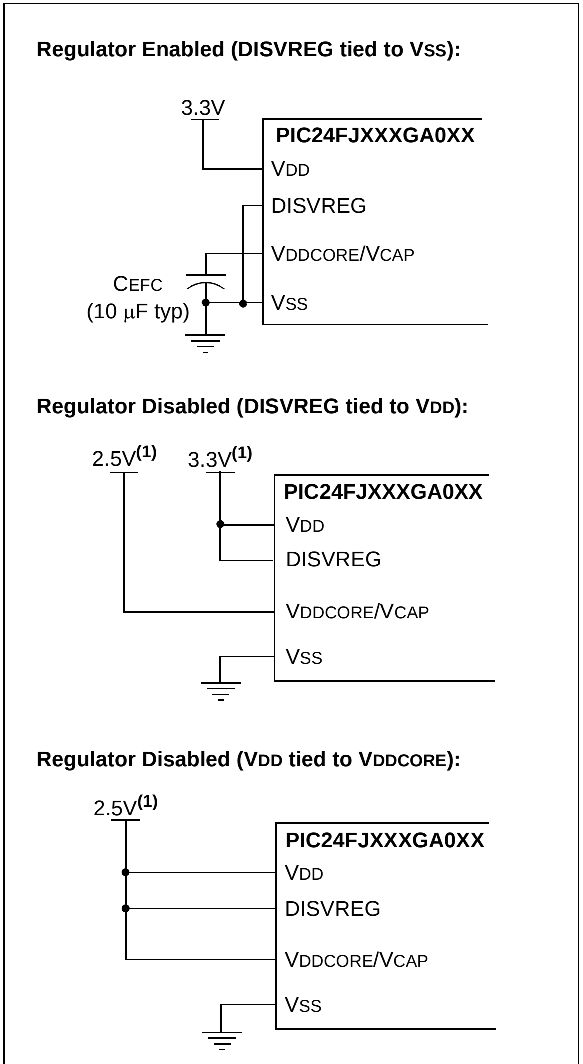

Regulator Enabled (ENVREG tied to VDD):

Regulator Disabled (ENVREG tied to ground):

Regulator Disabled (VDD tied to VDDCORE):

Note 1: These are typical operating voltages. Refer to **Section 7.0 "AC/DC Characteristics and Timing Requirements"** for the full operating ranges of VDD and VDDCORE.

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


Regulator Enabled (DISVREG tied to VSS):

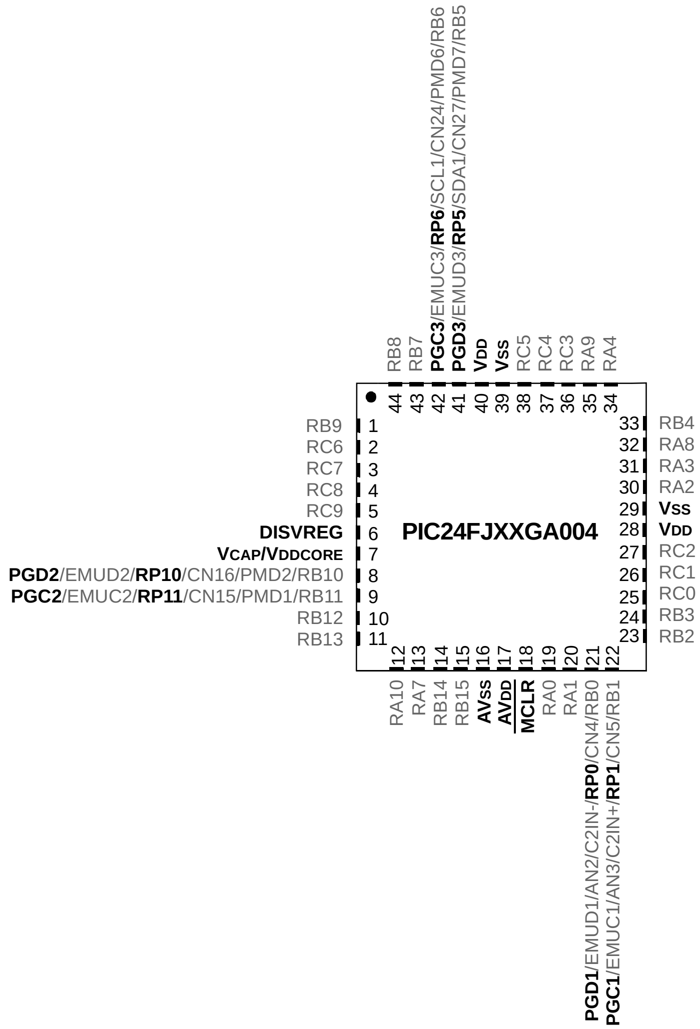
Regulator Disabled (DISVREG tied to VDD):

Regulator Disabled (VDD tied to VDDCORE):

Note 1: These are typical operating voltages. Refer to **Section 7.0 "AC/DC Characteristics and Timing Requirements"** for the full operating ranges of VDD and VDDCORE.

Pin Diagrams (Continued)

44-Pin QFN(1)

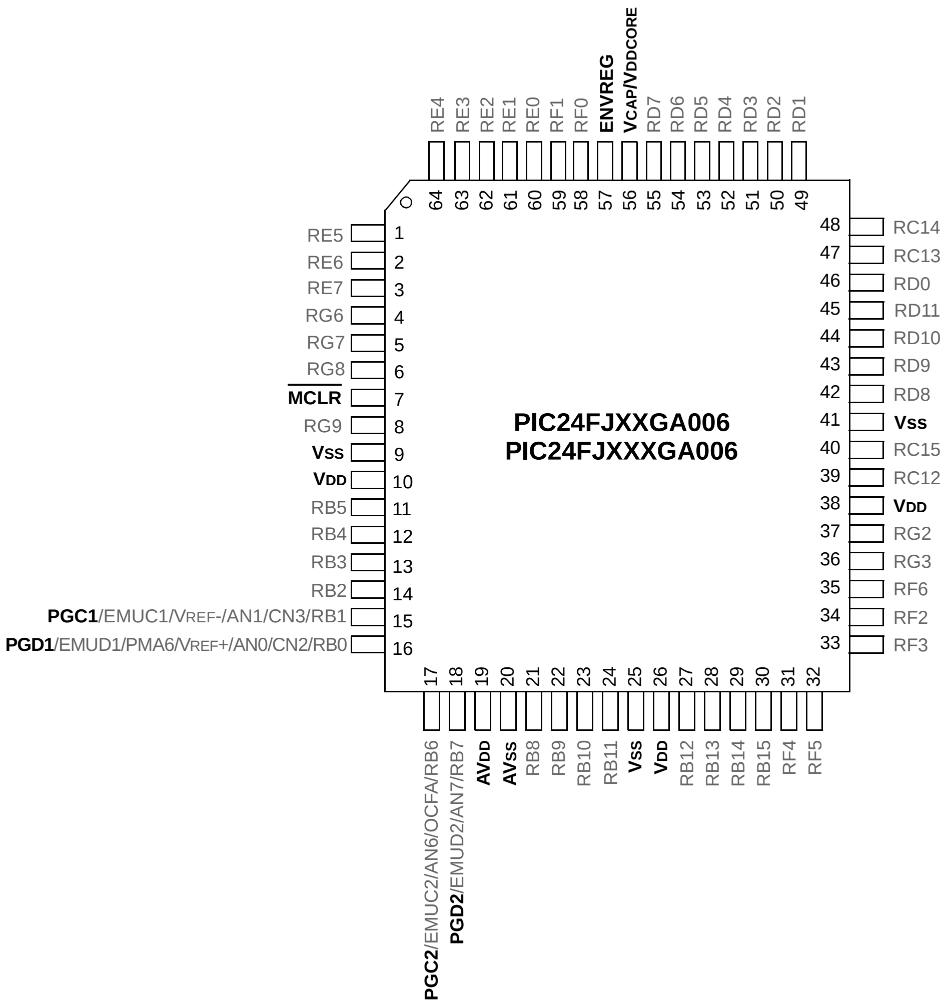


Legend: **RPx** represents remappable peripheral pins.

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

Pin Diagrams (Continued)

64-Pin TQFP



3.2.2 REGOUT SERIAL INSTRUCTION EXECUTION

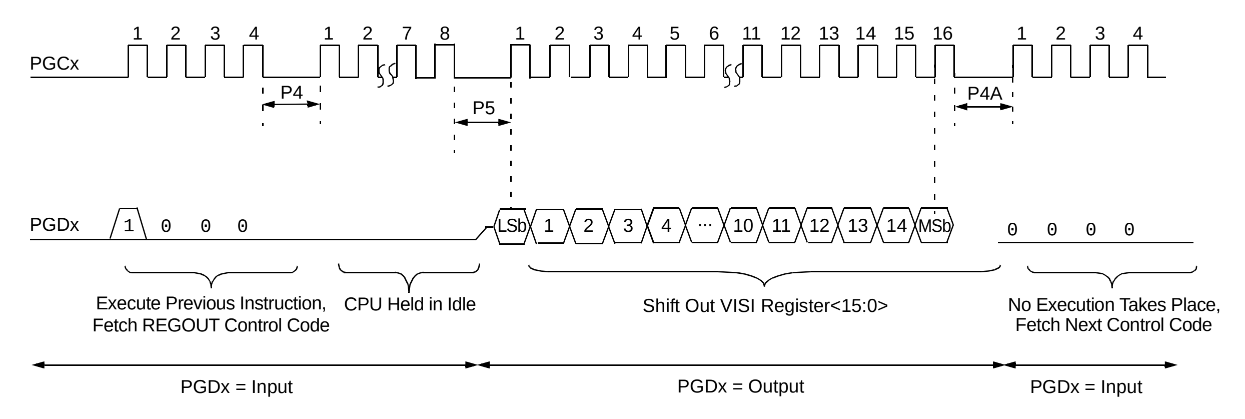
The REGOUT control code allows for data to be extracted from the device in ICSP mode. It is used to clock the contents of the VISI register, out of the device, over the PGDx pin. After the REGOUT control code is received, the CPU is held Idle for 8 cycles. After these 8 cycles, an additional 16 cycles are required to clock the data out (see Figure 3-3).

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

Note 1: After the contents of VISI are shifted out, the PIC24FJXXXGA0XX device maintains PGDx as an output until the first rising edge of the next clock is received.

2: Data changes on the falling edge and latches on the rising edge of PGCx. For all data transmissions, the Least Significant bit (LSb) is transmitted first.

FIGURE 3-3: REGOUT SERIAL EXECUTION


1 2 3 4 1 2 7 8 1 2 3 4 5 6 11 12 13 14 15 16 1 2 3 4
PGCx
P4
P5
P4A
PGDx
1 0 0 0
LSb 1 2 3 4 ··· 10 11 12 13 14 MSb
0 0 0 0
Execute Previous Instruction, Fetch REGOUT Control Code
CPU Held in Idle
Shift Out VISI Register<15:0>
No Execution Takes Place, Fetch Next Control Code
PGDx = Input
PGDx = Output
PGDx = Input

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:

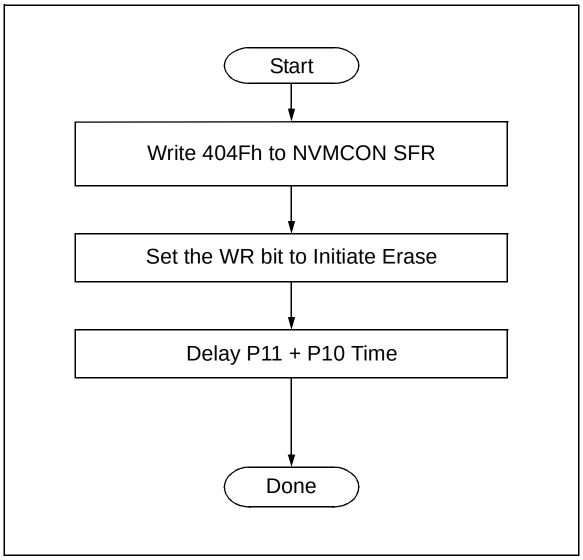
- If the TBLPAG register points to user space (is less than 0x80), the Chip Erase will erase only user memory.
- If TBLPAG points to configuration space (is greater than or equal to 0x80), the Chip Erase will erase both user and configuration memory.

  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



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'.

3.9 Reading Configuration Words

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

Table 3-9 shows the ICSP programming details for reading the Configuration Words. Note that the TBLPAG register must be loaded with 00h for 96 Kbyte and below devices and 01h for 128 Kbyte devices (the upper byte address of configuration memory), and the Read Pointer, W6, is initialized to the lower 16 bits of the Configuration Word location.

TABLE 3-9: SERIAL INSTRUCTION EXECUTION FOR READING ALL CONFIGURATION MEMORY

Command (Binary)	Data (Hex)	Description
Step 1: Exit Reset vector.		

`0000`
`0000`
`0000`
	`000000`
	`040200`
	`000000`
		`NOP`
		`GOTO 0x200`
		`NOP`

Step 2: Initialize TBLPAG, the Read Pointer (W6) and the Write Pointer (W7) for `TBLRD` instruction.		

`0000`
`0000`
`0000`
`0000`
`0000`
	`200xx0`
	`880190`
	`2xxxx7`
	`207847`
	`000000`
		`MOV <CW2Address23:16>, W0`
		`MOV W0, TBLPAG`
		`MOV <CW2Address15:0>, W6`
		`MOV #VISI, W7`
		`NOP`

Step 3: Read the Configuration register and write it to the VISI register (located at 784h), and clock out the VISI register using the REGOUT command.		

`0000`
`0000`
`0000`
`0001`
`0000`
	`BA0BB6`
	`000000`
	`000000`
	`<VISI>`
	`000000`
		`TBLRDL [W6++], [W7]`
		`NOP`
		`NOP`
		`Clock out contents of VISI register`
		`NOP`

Step 4: Repeat Step 3 again to read Configuration Word 1.		
Step 5: Reset device internal PC.		

`0000`
`0000`
	`040200`
	`000000`
		`GOTO 0x200`
		`NOP`

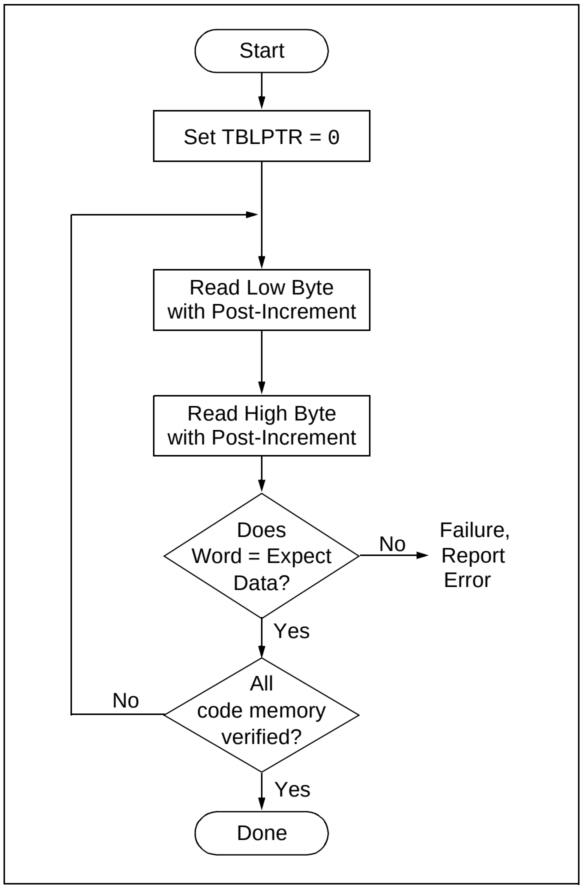
3.10 Verify Code Memory and Configuration Word

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

The verify process is shown in the flowchart in Figure 3-8. Memory reads occur a single byte at a time, so two bytes must be read to compare against the word in the programmer's buffer. Refer to **Section 3.8 "Reading Code Memory"** for implementation details of reading code memory.

> **Note:** Because the Configuration registers include the device code protection bit, code memory should be verified immediately after writing if code protection is enabled. This is because the device will not be readable or verifiable if a device Reset occurs after the code-protect bit in CW1 has been cleared.

FIGURE 3-8: VERIFY CODE MEMORY FLOW


Start
Set TBLPTR = 0
Read Low Byte with Post-Increment
Read High Byte with Post-Increment
Does Word = Expect Data?
No
Failure, Report Error
Yes
All code memory verified?
No
Yes
Done

3.11 Reading the Application ID Word

The Application ID Word is stored at address 8005BEh in executive code memory. To read this memory location, you must use the SIX control code to move this program memory location to the VISI register. Then, the REGOUT control code must be used to clock the contents of the VISI register out of the device. The corresponding control and instruction codes that must be serially transmitted to the device to perform this operation are shown in Table 3-10.

After the programmer has clocked out the Application ID Word, it must be inspected. If the Application ID has the value, BBh, the programming executive is resident in memory and the device can be programmed using the mechanism described in **Section 4.0 "Device Programming – Enhanced ICSP"**. However, if the Application ID has any other value, the programming executive is not resident in memory; it must be loaded to memory before the device can be programmed. The procedure for loading the programming executive to memory is described in **Section 5.4 "Programming the Programming Executive to Memory"**.

3.12 Exiting ICSP Mode

Exiting Program/Verify mode is done by removing VIH from $\overline{\text{MCLR}}$, as shown in Figure 3-9. The only requirement for exit is that an interval, P16, should elapse between the last clock and program signals on PGCx and PGDx before removing VIH.

FIGURE 3-9: EXITING ICSP™ MODE

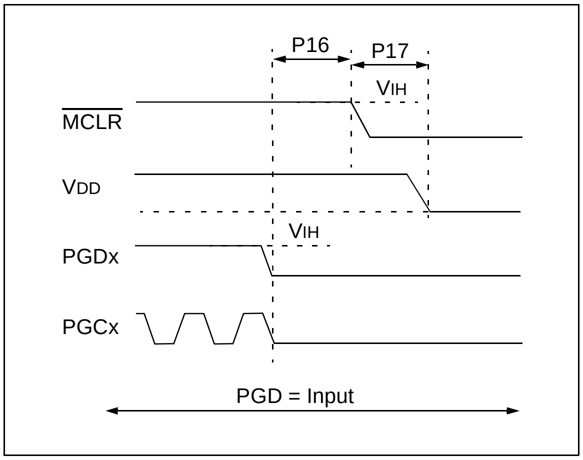


TABLE 3-10: SERIAL INSTRUCTION EXECUTION FOR READING THE APPLICATION ID WORD

Command (Binary)	Data (Hex)	Description
Step 1: Exit Reset vector.		

`0000`
`0000`
`0000`
	`000000`
	`040200`
	`000000`
		`NOP`
		`GOTO    0x200`
		`NOP`

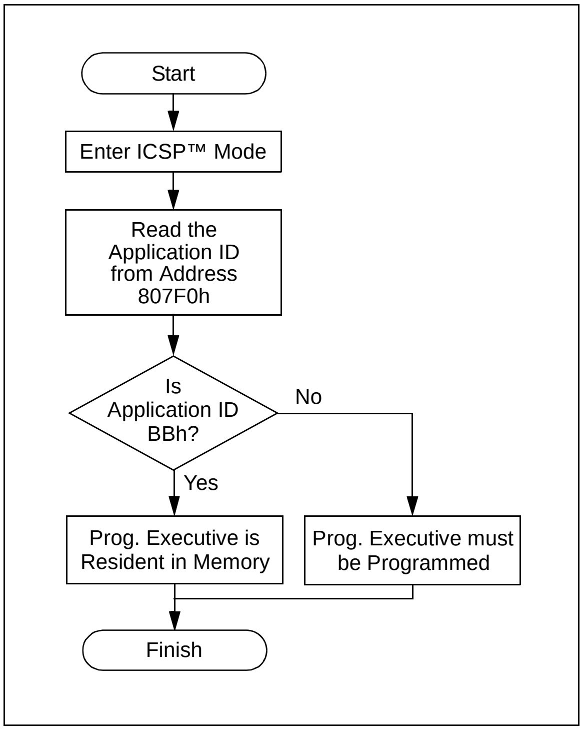
Step 2: Initialize TBLPAG and the Read Pointer (W0) for `TBLRD` instruction.		

`0000`
`0000`
`0000`
`0000`
`0000`
`0000`
`0000`
`0000`
	`200800`
	`880190`
	`205BE0`
	`207841`
	`000000`
	`BA0890`
	`000000`
	`000000`
		`MOV     #0x80, W0`
		`MOV     W0, TBLPAG`
		`MOV     #0x5BE, W0`
		`MOV     #VISI, W1`
		`NOP`
		`TBLRDL  [W0], [W1]`
		`NOP`
		`NOP`

Step 3: Output the VISI register using the REGOUT command.		

`0001`
`0000`
	`<VISI>`
	`000000`
		`Clock out contents of the VISI register`
		`NOP`

FIGURE 4-2: CONFIRMING PRESENCE OF PROGRAMMING EXECUTIVE


Start
Enter ICSP™ Mode
Read the Application ID from Address 807F0h
Is Application ID BBh?
No
Yes
Prog. Executive is Resident in Memory
Prog. Executive must be Programmed
Finish

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.

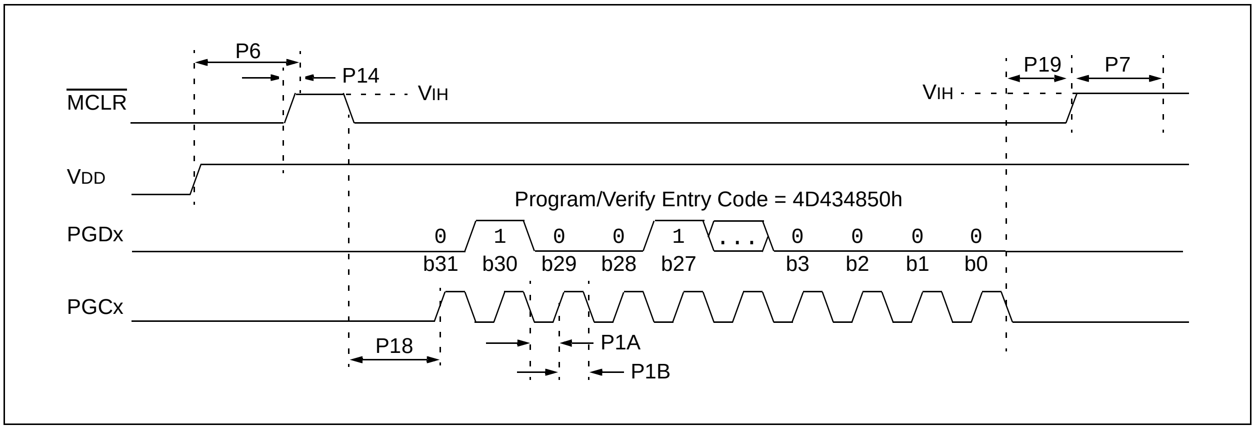
The programming voltage applied to $\overline{\text{MCLR}}$ is VIH, which is essentially VDD in the case of PIC24FJXXXGA0XX devices. There is no minimum time requirement for holding at VIH. After VIH is removed, an interval of at least P18 must elapse before presenting the key sequence on PGDx.

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, VIH 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


P6
P14
VIH
MCLR
VIH
P19
P7
VDD
Program/Verify Entry Code = 4D434850h
PGDx
0 1 0 0 1 ... 0 0 0 0
b31 b30 b29 b28 b27 b3 b2 b1 b0
PGCx
P18
P1A
P1B

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
$\overline{\text{DEBUG}}$	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.

TABLE 4-2: PIC24FJXXXGA0XX FAMILY CONFIGURATION BITS DESCRIPTION (CONTINUED)

Bit Field	Register	Description
ICS[1]	CW1<8>	ICD Pin Placement Select bit 11 = ICD EMUC/EMUD pins are shared with PGC1/PGD1 10 = ICD EMUC/EMUD pins are shared with PGC2/PGD2 01 = ICD EMUC/EMUD pins are shared with PGC3/PGD3 00 = Reserved; do not use
IESO	CW2<15>	Internal External Switchover bit 1 = Two-Speed Start-up enabled 0 = Two-Speed Start-up disabled
IOL1WAY[1]	CW2<4>	IOLOCK Bit One-Way Set Enable bit 0 = The OSCCON<IOLOCK> bit can be set and cleared as needed (provided an unlocking sequence is executed) 1 = The OSCCON<IOLOCK> bit can only be set once (provided an unlocking sequence is executed). Once IOLOCK is set, this prevents any possible future RP register changes
JTAGEN	CW1<14>	JTAG Enable bit 1 = JTAG enabled 0 = JTAG disabled
OSCIOFNC	CW2<5>	OSC2 Pin Function bit (except in XT and HS modes) 1 = OSC2 is clock output 0 = OSC2 is general purpose digital I/O pin
SOSCSEL1: SOSCSEL0[2]	CW2<12:11>	Secondary Oscillator Power Mode Select bits 11 = Default (high drive strength) mode 01 = Low-Power (low drive strength) mode x0 = Reserved; do not use
POSCMD1: POSCMD0	CW2<1:0>	Primary Oscillator Mode Select bits 11 = Primary oscillator disabled 10 = HS Crystal Oscillator mode 01 = XT Crystal Oscillator mode 00 = EC (External Clock) mode
WDTPOST3: WDTPOST0	CW1<3:0>	Watchdog Timer Prescaler bit 1111 = 1:32,768 1110 = 1:16,384 . . . 0001 = 1:2 0000 = 1:1
WDTPRE	CW1<4>	Watchdog Timer Postscaler bit 1 = 1:128 0 = 1:32
WINDIS	CW1<6>	Windowed WDT bit 1 = Watchdog Timer in Non-Window mode 0 = Watchdog Timer in Window mode; FWDTEN must be ‘1’
WUTSEL1: WUTSEL0[2]	CW2<14:13>	Voltage Regulator Standby Mode Wake-up Time Select bits 11 = Default regulator wake time used 01 = Fast regulator wake time used x0 = Reserved; do not use

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.

5.2.6 READC COMMAND

15 12	11 8	7 0
Opcode	Length	
N		Addr_MSB
Addr_LS		

Field	Description
Opcode	1h
Length	3h
N	Number of 8-bit Device ID registers to read (max. of 256)
Addr_MSB	MSB of 24-bit source address
Addr_LS	Least Significant 16 bits of 24-bit source address

The READC command instructs the programming executive to read N or Device ID registers, starting from the 24-bit address specified by Addr_MSB and Addr_LS. This command can only be used to read 8-bit or 16-bit data.

When this command is used to read Device ID registers, the upper byte in every data word returned by the programming executive is 00h and the lower byte contains the Device ID register value.

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

1100h
2 + N
Device ID Register 1
...
Device ID Register N

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

5.2.7 READP COMMAND

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

Field	Description
Opcode	2h
Length	4h
N	Number of 24-bit instructions to read (max. of 32768)
Reserved	0h
Addr_MSB	MSB of 24-bit source address
Addr_LS	Least Significant 16 bits of 24-bit source address

The READP command instructs the programming executive to read N 24-bit words of code memory, including Configuration Words, starting from the 24-bit address specified by Addr_MSB and Addr_LS. This command can only be used to read 24-bit data. All data returned in response to this command uses the packed data format described in **Section 5.2.2 “Packed Data Format”**.

Expected Response (2 + 3 * N/2 words for N even):

1200h
2 + 3 * N/2
Least significant program memory word 1
...
Least significant data word N

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

1200h
4 + 3 * (N – 1)/2
Least significant program memory word 1
...
MSB of program memory word N (zero padded)

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

5.2.8 PROGC COMMAND

<table>
<tr><th>15 12</th><th>11 8</th><th>7 0</th></tr>
<tr><td>Opcode</td><td colspan="2">Length</td></tr>
<tr><td colspan="2">Reserved</td><td>Addr_MSB</td></tr>
<tr><td colspan="3">Addr_LS</td></tr>
<tr><td colspan="3">Data</td></tr>
</table>

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

<table>
<tr><th>15 12</th><th>11 8</th><th>7 0</th></tr>
<tr><td>Opcode</td><td colspan="2">Length</td></tr>
<tr><td colspan="2">Reserved</td><td>Addr_MSB</td></tr>
<tr><td colspan="3">Addr_LS</td></tr>
<tr><td colspan="3">D_1</td></tr>
<tr><td colspan="3">D_2</td></tr>
<tr><td colspan="3">...</td></tr>
<tr><td colspan="3">D_96</td></tr>
</table>

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.

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.

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`

6.2 Checksum Computation

Checksums for the PIC24FJXXXGA0XX family are 16 bits in size. The checksum is calculated by summing the following:

- Contents of code memory locations
- Contents of Configuration registers

Table 6-4 describes how to calculate the checksum for each device. All memory locations are summed, one byte at a time, using only their native data size. More specifically, Configuration registers are summed by adding the lower two bytes of these locations (the upper byte is ignored), while code memory is summed by adding all three bytes of code memory.

TABLE 6-4: CHECKSUM COMPUTATION

Device	Read Code Protection	Checksum Computation	Erased Checksum Value	Checksum with 0xAAAAAA at 0x0 and Last Code Address
PIC24FJ16GA002	Disabled	CFGB + SUM(0:02BFB)	0xBB5A	0xB95C
	Enabled	0	0x0000	0x0000
PIC24FJ16GA004	Disabled	CFGB + SUM(0:02BFB)	0xBB5A	0xB95C
	Enabled	0	0x0000	0x0000
PIC24FJ32GA002	Disabled	CFGB + SUM(0:057FB)	0x795A	0x775C
	Enabled	0	0x0000	0x0000
PIC24FJ32GA004	Disabled	CFGB + SUM(0:057FB)	0x795A	0x775C
	Enabled	0	0x0000	0x0000
PIC24FJ48GA002	Disabled	CFGB + SUM(0:083FB)	0x375A	0x355C
	Enabled	0	0x0000	0x0000
PIC24FJ48GA004	Disabled	CFGB + SUM(0:083FB)	0x375A	0x355C
	Enabled	0	0x0000	0x0000
PIC24FJ64GA002	Disabled	CFGB + SUM(0:0ABFB)	0xFB5A	0xF95C
	Enabled	0	0x0000	0x0000
PIC24FJ64GA004	Disabled	CFGB + SUM(0:0ABFB)	0xFB5A	0xF95C
	Enabled	0	0x0000	0x0000
PIC24FJ64GA006	Disabled	CFGB + SUM(0:0ABFB)	0xFACC	0xF8CE
	Enabled	0	0x0000	0x0000
PIC24FJ64GA008	Disabled	CFGB + SUM(0:0ABFB)	0xFACC	0xF8CE
	Enabled	0	0x0000	0x0000
PIC24FJ64GA010	Disabled	CFGB + SUM(0:0ABFB)	0xFACC	0xF8CE
	Enabled	0	0x0000	0x0000
PIC24FJ96GA006	Disabled	CFGB + SUM(0:0FFFB)	0x7CCC	0x7ACE
	Enabled	0	0x0000	0x0000
PIC24FJ96GA008	Disabled	CFGB + SUM(0:0FFFB)	0x7CCC	0x7ACE
	Enabled	0	0x0000	0x0000
PIC24FJ96GA010	Disabled	CFGB + SUM(0:0FFFB)	0x7CCC	0x7ACE
	Enabled	0	0x0000	0x0000

Legend:

Item		Description
SUM[a:b]	=	Byte sum of locations, a to b inclusive (all 3 bytes of code memory)
CFGB	=	Configuration Block (masked), 64/80/100-Pin Devices = Byte sum of (CW1 & 0x7DDF + CW2 & 0x87E3) 28/44-Pin Devices = Byte sum of (CW1 & 0x7FDF + CW2 & 0xFFF7)

Note: CW1 address is last location of implemented program memory; CW2 is (last location – 2).

TABLE 6-4: CHECKSUM COMPUTATION (CONTINUED)

Device	Read Code Protection	Checksum Computation	Erased Checksum Value	Checksum with 0xAAAAAA at 0x0 and Last Code Address
PIC24FJ128GAGA006	Disabled	CFGB + SUM(0:0157FB)	0xF8CC	0xF6CE
	Enabled	0	0x0000	0x0000
PIC24FJ128GAGA008	Disabled	CFGB + SUM(0:0157FB)	0xF8CC	0xF6CE
	Enabled	0	0x0000	0x0000
PIC24FJ128GAGA010	Disabled	CFGB + SUM(0:0157FB)	0xF8CC	0xF6CE
	Enabled	0	0x0000	0x0000

Legend: Item Description
SUM[a:b] = Byte sum of locations, a to b inclusive (all 3 bytes of code memory)
CFGB = Configuration Block (masked),
64/80/100-Pin Devices = Byte sum of (CW1 & 0x7DDF + CW2 & 0x87E3)
28/44-Pin Devices = Byte sum of (CW1 & 0x7FDF + CW2 & 0xFFF7)

Note: CW1 address is last location of implemented program memory; CW2 is (last location – 2).

NOTES:

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, PIC^{32} 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.

♻ 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.