Section 5. Flash Programming

This section of the manual contains the following major topics:

Note: This family reference manual section is meant to serve as a complement to device data sheets. Depending on the device variant, this manual section may not apply to all dsPIC33E/PIC24E devices.

Please consult the note at the beginning of the **"Flash Program Memory"** chapter in the current device data sheet to check whether this document supports the device you are using.

Device data sheets and family reference manual sections are available for download from the Microchip Worldwide Web site at: http://www.microchip.com

5.1 INTRODUCTION

This section describes the technique for programming Flash program memory. The dsPIC33E/PIC24E family of devices have an internal programmable Flash program memory for execution of user code. There are two methods to program this memory:

- Run-Time Self-Programming (RTSP)
- In-Circuit Serial Programming™ (ICSP™)

This section describes RTSP programming, which is performed by the user's software.

ICSP is performed using a serial data connection to the device and allows for faster programming than RTSP. The ICSP protocol is defined in the *"dsPIC33E/PIC24E Flash Programming Specification"* (DS70619), which can be downloaded from the Microchip web site.

© 2011 Microchip Technology Inc.

5.2 TABLE INSTRUCTION OPERATION

The table instructions provide one method of transferring data between the Flash program memory space and the data memory space of dsPIC33E/PIC24E devices. This section provides a summary of the table instructions used during programming of the Flash program memory. There are four basic table instructions:

- `TBLRDL`: Table Read Low
- `TBLRDH`: Table Read High
- `TBLWTL`: Table Write Low
- `TBLWTH`: Table Write High

The `TBLRDL` instruction is used to read from bits <15:0> of program memory space. The `TBLWTL` instruction is used to write to bits <15:0> of Flash program memory space. `TBLRDL` and `TBLWTL` can access Flash program memory in Word or Byte mode.

The `TBLRDH` and `TBLWTH` instructions are used to read or write to bits <23:16> of program memory space. `TBLRDH` and `TBLWTH` can access Flash program memory in Word or Byte mode. Because the Flash program memory is only 24 bits wide, the `TBLRDH` and `TBLWTH` instructions can address an upper byte of Flash program memory that does not exist. This byte is called the “phantom byte”. Any read of the phantom byte will return 0x00. A write to the phantom byte has no effect.

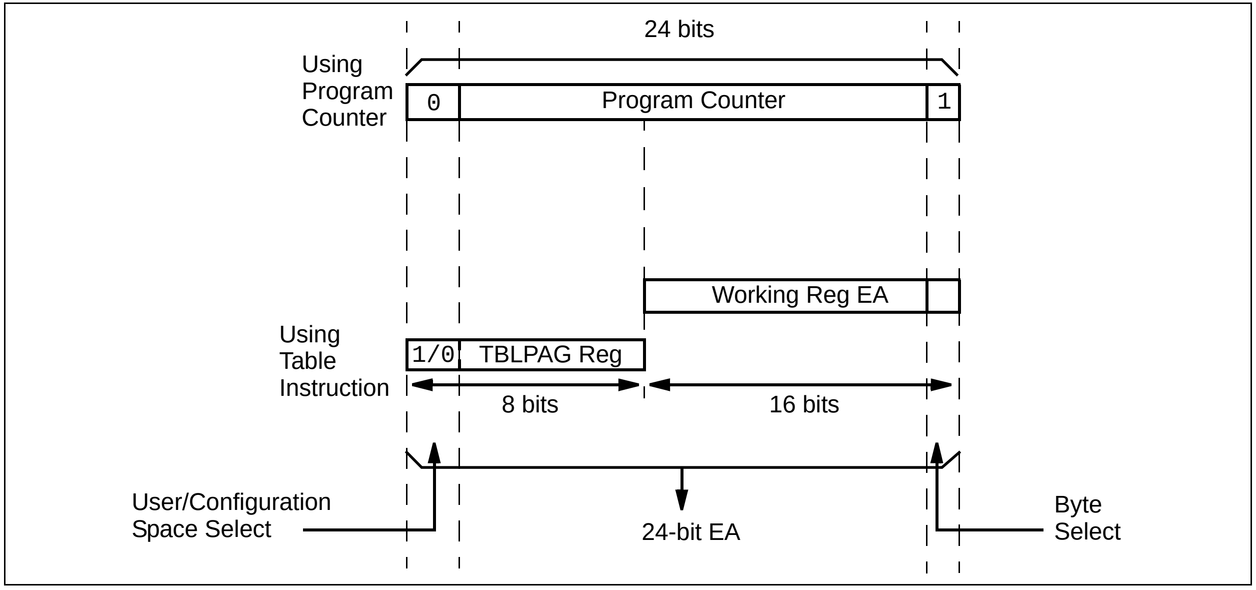
The 24-bit Flash program memory can be regarded as two side-by-side 16-bit spaces, with each space sharing the same address range. Therefore, the `TBLRDL` and `TBLWTL` instructions access the “low” program memory space (PM<15:0>). The `TBLRDH` and `TBLWTH` instructions access the “high” program memory space (PM<31:16>). Any reads or writes to PM<31:24> will access the phantom (unimplemented) byte. When any of the table instructions are used in Byte mode, the Least Significant bit (LSb) of the table address will be used as the byte select bit. The LSb determines which byte in the high or low program memory space is accessed.

Figure 5-1 shows how the Flash program memory is addressed using the table instructions. A 24-bit program memory address is formed using bits <7:0> of the TBLPAG register and the effective address (EA) from a W register specified in the table instruction. The 24-bit program counter (PC) is shown in Figure 5-1 for reference. The upper 23 bits of the EA are used to select the Flash program memory location.

For the Byte mode table instructions, the LSb of the W register EA is used to select which byte of the 16-bit Flash program memory word is addressed. A ‘`1`’ selects bits <15:8>. A ‘`0`’ selects bits <7:0>. The LSb of the W register EA is ignored for a table instruction in Word mode.

In addition to the Flash program memory address, the table instruction also specifies a W register (or a W Register Pointer to a memory location), that is the source of the Flash program memory data to be written or the destination for a Flash program memory read. For a table write operation in Byte mode, bits <15:8> of the source working register are ignored.

Figure 5-1: Addressing for Table Instructions



5.2.1 Using Table Read Instructions

Table reads require two steps:

- The address pointer is set up using the TBLPAG register and one of the W registers
- The Flash program memory contents at the address location may be read

5.2.1.1 READ WORD MODE

The code shown in Example 5-1, shows how to read a word of Flash program memory using the table instructions in Word mode.

Example 5-1: Read Word Mode

```
; Set up the address pointer to program space
    MOV         #tblpage(PROG_ADDR),W0        ; get table page value
    MOV         W0,TBLPAG                     ; load TBLPAG register
    MOV         #tbloffset(PROG_ADDR),W0      ; load address LS word
; Read the program memory location
    TBLRDH      [W0],W3                       ; Read high byte to W3
    TBLRDL      [W0],W4                       ; Read low word to W4
```

5.2.1.2 READ BYTE MODE

The code shown in Example 5-2, shows the post-increment operator on the read of the low byte, which causes the address in the working register to increment by one. This sets EA<0> to a ‘1’ for access to the middle byte in the third write instruction. The last post-increment sets W0 back to an even address, pointing to the next Flash program memory location.

Example 5-2: Read Byte Mode

```
; Set up the address pointer to program space
    MOV         #tblpage(PROG_ADDR),W0        ; get table page value
    MOV         W0,TBLPAG                     ; load TBLPAG register
    MOV         #tbloffset(PROG_ADDR),W0      ; load address LS word
; Read the program memory location
    TBLRDH.B    [W0],W3                       ; Read high byte to W3
    TBLRDL.B    [W0++],W4                     ; Read low byte to W4
    TBLRDL.B    [W0++],W5                     ; Read middle byte to W5
```

© 2011 Microchip Technology Inc.

5.2.1.3 TABLE WRITE HOLDING LATCHES

Table write instructions do not write directly to the nonvolatile program. Instead, the table write instructions load holding latches that store the write data. The NVM address registers must be loaded with the page address where latched data should be written. When all of the holding latches have been loaded, the actual memory programming operation is started by executing a special sequence of instructions.

The dsPIC33E/PIC24E Flash program memory is segmented into pages (1024 instruction words) and rows (128 instruction words).

The dsPIC33E/PIC24E family of devices support 128 holding registers to program one row of memory at a time. The memory logic automatically decides which set of write latches to load based on the address value of the NVM address registers.

Refer to the specific device data sheet for more information.

5.3 CONTROL REGISTERS

Several Special Function Registers (SFRs) are used to program the Flash program memory erase and write operations: NVMCON, NVMKEY, and the NVM Address registers, NVMADR and NVMADRU.

5.3.1 NVMCON Register

The NVMCON register is the primary control register for Flash and program/erase operations. This register selects whether an erase or program operation will be performed and can start the program or erase cycle.

The NVMCON register is shown in Register 5-1. The lower byte of NVMCON configures the type of NVM operation that will be performed.

5.3.2 NVMKEY Register

The NVMKEY register (Register 5-4) is a write-only register used to prevent accidental writes or erasures of Flash memory. To start a programming or erase sequence, the following steps must be considered:

1. Write 0x55 to NVMKEY.
2. Write 0xAA to NVMKEY.
3. Execute two `NOP` instructions.

After this sequence, a write will be allowed to the NVMCON register for one instruction cycle. In most cases, the user application needs to set the WR bit in the NVMCON register to start the program or erase cycle. Interrupts should be disabled during the unlock sequence. Example 5-3 shows how the unlock sequence is performed.

Example 5-3: NVMKEY Unlock Sequence

```
;   PUSH    SR              ; Disable interrupts, if enabled
    MOV     #0x00E0,W0
    IOR     SR

    MOV     #0x55,W0
    MOV     W0, NVMKEY
    MOV     #0xAA,W0
    MOV     W0, NVMKEY      ; NOP not required
    BSET    NVMCON,#15      ; Start the program/erase cycle
    NOP
    NOP
    POP     SR              ; Re-enable interrupts
```

Refer to **5.4.2 “Flash Programming Operations”** for more programming examples.

5.3.3 NVM Address Registers

The two NVM Address registers, NVMADRU and NVMADR, when concatenated, form the 24-bit EA of the selected row or word for programming operations. The NVMADRU register is used to hold the upper eight bits of the EA, and the NVMADR register is used to hold the lower 16 bits of the EA.

© 2011 Microchip Technology Inc.

Register 5-1: NVMCON: Flash Memory Control Register

R/SO-0[1]	R/W-0[1]	R/W-0[1]	R/W-0	U-0	U-0	U-0	U-0
WR	WREN	WRERR	NVMSIDL[2]	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	U-0	R/W-0[1]	R/W-0[1]	R/W-0[1]	R/W-0[1]
—	—	—	—	NVMOP<3:0>[3,5]			
bit 7							bit 0

Legend:	SO = Settable-only bit		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 15 **WR:** Write Control bit

`1` = Initiates a Flash program memory or erase operation. The operation is self-timed and the bit is cleared by hardware once operation is complete
`0` = Program or erase operation is complete and inactive

bit 14 **WREN:** Write Enable bit

`1` = Enable Flash program memory/erase operations
`0` = Inhibit Flash program memory/erase operations

bit 13 **WRERR:** Write Sequence Error Flag bit

`1` = An improper program or erase sequence attempt or termination has occurred (bit is set automatically on any set attempt of the WR bit)
`0` = The program or erase operation completed normally

bit 12 **NVMSIDL:** Stop in Idle Mode bit[2]

`1` = Discontinue Primary and Auxiliary Flash operation when the device enters Idle mode
`0` = Continue Primary and Auxiliary Flash operation when the device enters Idle mode

bit 11-4 **Unimplemented:** Read as ‘`0`’

bit 3-0 **NVMOP<3:0>:** NVM Operation Select bits[3,5]

`1111` = Reserved
`1110` = Reserved
`1101` = Bulk erase primary Flash program memory
`1100` = Reserved[4]
`1011` = Reserved[4]
`1010` = Bulk erase auxiliary Flash program memory
`0011` = Memory page erase operation
`0010` = Memory row program operation
`0001` = Memory word program operation[6]
`0000` = Program a single Configuration register byte

Note 1: These bits can only be reset on POR.
2: When exiting Idle mode, there is a power-up delay (TNPD) before Flash program memory becomes operational. Refer to the specific device data sheet for more information.
3: All other combinations of NVMOP<3:0> are unimplemented.
4: Entire segment is erased with the exception of IVT.
5: Execution of `PWRSAV` instruction is ignored while any of the NVM operations are in progress.
6: The Word Programming RTSP operation causes two adjacent words (an even/odd instruction pair) to get reprogrammed.

Register 5-2: NVMADRU: Nonvolatile Memory Upper Address Register

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
NVMADRU<7:0>							
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 15-8	**Unimplemented:** Read as ‘0’
bit 7-0	**NVMADRU<7:0>:** Nonvolatile Memory Upper Write Address bits Selects the upper eight bits of the location to program or erase in Flash program memory. This register may be read or written by the user application.

Register 5-3: NVMADR: Nonvolatile Memory Address Register

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
NVMADR<15:8>							
bit 15							bit 8

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
NVMADR<7:0>							
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 15-0	**NVMADR<15:0>:** Nonvolatile Memory Write Address bits Selects the upper eight bits of the location to program or erase in Flash program memory. This register may be read or written by the user application.

Register 5-4: NVMKEY: Nonvolatile Memory Key Register

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
NVMKEY<7:0>							
bit 7							bit 0

Legend:	SO = Settable-only bit		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 15-8	**Unimplemented:** Read as ‘0’
bit 7-0	**NVMKEY<7:0>:** Key Register (write-only) bits

5.4 RUN-TIME SELF-PROGRAMMING (RTSP)

RTSP allows the user application to modify Flash program memory contents. RTSP is accomplished using `TBLRD` (table read) and `TBLWT` (table write) instructions, and the NVM Control registers. With RTSP, the user application can erase Flash program memory eight rows (128 * 8 = 1024 instructions) at a time and can write the Flash program memory data a single row (128 instructions) at a time.

5.4.1 RTSP Operation

The dsPIC33E/PIC24E Flash program memory array is organized into rows of 128 instructions or 384 bytes. RTSP allows the user application to erase blocks of eight rows (1024 instructions) at a time and to program 128 instructions at a time. The 8-row erase blocks and single-row write blocks are edge-aligned, from the beginning of Flash program memory, on boundaries of 3072 bytes and 384 bytes, respectively.

The Flash program memory implements holding buffers that can contain 128 instructions of programming data. Prior to the actual programming operation, the write data must be loaded into the buffers in sequential order. The instruction words loaded must always be from a group of 128 boundaries.

The basic sequence for RTSP is to set up a Table Pointer, and then perform a series of `TBLWT` instructions to load the buffers. Programming is performed by setting the control bits in the NVMCON register. A total of 128 `TBLWTL` and `TBLWTH` instructions are required to load the instructions.

All of the table write operations are single-word writes (two instruction cycles), because only the buffers are written. A programming cycle is required for programming each row.

5.4.2 Flash Programming Operations

A program or erase operation is necessary for programming or erasing the internal Flash program memory in RTSP mode. The program or erase operation is automatically timed by the device (refer to the specific device data sheet for timing information). Setting the WR bit (NVMCON<15>) starts the operation. The WR bit is automatically cleared when the operation is finished. In addition, the dsPIC33E/PIC24E device provides an NVM interrupt, which indicates when programming operation has completed.

The CPU stalls until the programming operation is finished. The CPU will not execute any instructions or respond to interrupts during this time. If any interrupts occur during the programming cycle, they will remain pending until the cycle completes.

Some dsPIC33E/PIC24E devices may provide auxiliary Flash program memory (refer to the specific device data sheet for details), which allows instruction execution without CPU stalls while user Flash program memory is being erased and/or programmed. Conversely, auxiliary Flash program memory can be programmed without CPU stall, as long as code is executed from the user Flash program memory.

Note 1: If a POR or BOR event occurs while an RTSP erase or programming operation is in progress, the RTSP operation is aborted immediately. The user should execute the RTSP operation again after the device comes out of Reset.

2: If an EXTR, SWR, WDTO, TRAPR, CM, or IOPUWR reset event occurs while an RTSP erase or programming operation is in progress, the device will be Reset only after the RTSP operation is complete.

© 2011 Microchip Technology Inc.

5.4.2.1 FLASH ROW PROGRAMMING ALGORITHM

The user application can program one row of Flash program memory (128 instruction words). To perform this, it is necessary to erase a page (1024 instruction words) containing the desired row.

The general process is as follows:

1. Read one page (1024 instruction words) of Flash program memory and store it into data RAM as a data “image”. The RAM image must be read from an even 1024-word program memory address boundary.
2. Update the RAM data image with the new Flash program memory data.
3. Erase the Flash program memory page.
    a) Set up the NVMCON register to erase one page of Flash program memory.
    b) Disable interrupts.
    c) Write the address of the row to be erased into the NVMADRU and NMVADR registers (can be any address within the row).
    d) Write the key sequence to the NVMKEY register to enable the erase.
    e) Set the WR bit (NVMCON<15>). This will start the erase cycle.
    f) The WR bit is cleared when the erase cycle ends.
    g) Re-enable interrupts.
4. Load the row (128 instruction words) of instruction words from RAM into the write latches using a table write operation.
5. Program the row (128 instruction words) in Flash program memory.
    a) Set up the NVMCON register to program one row of Flash program memory.
    b) Disable interrupts.
    c) Write the address of the row to be programmed into the NVMADRU and NVMADR registers (can be any address within the row).
    d) Write the key sequence to the NVMKEY register to enable the program cycle.
    e) Set the WR bit. This will start the program cycle.
    f) The WR bit is cleared by the hardware when the program cycle ends.
    g) Re-enable interrupts.
6. Repeat steps 4 through 6 to program all eight rows in the program memory page.
7. Repeat steps 1 through 7, as needed, to program the desired amount of Flash program memory.

Note 1: The user should remember that the minimum amount of Flash program memory that can be erased using RTSP is 1024 instruction word locations. Therefore, it is important that an image of these locations be stored in general purpose RAM before an erase cycle is initiated.

2: A row or word in Flash program memory should not be programmed more than twice before being erased.

5.4.2.2 ERASING ONE PAGE OF FLASH

The code sequence shown in Example 5-4 can be used to erase a page (1024 instructions) of Flash program memory. The NVMCON register is configured to erase one page of program memory. The NVMADR and NMVADRU registers are loaded with the address of the page to be erased. The program memory must be erased at an “even” 1024 instruction word address boundary. Therefore, the 11 LSbs of the table write program memory address have no effect when a page is erased.

The erase operation is initiated by writing a special unlock, or key, sequence to the NVMKEY register before setting the WR control bit (NVMCON<15>). The unlock sequence needs to be executed in the exact order, as shown in Example 5-4, without interruption. Therefore, interrupts should be disabled prior to writing the sequence.

Two `NOP` instructions should be inserted in the code after the erase cycle. Finally, interrupts can be enabled (if required).

Example 5-4: Erasing a Page of Flash Program Memory

```
; Define the start address from where the erase has to start
       .equ PROG_ADDR, 0x022222
; Perform dummy table write to the Page to be erased.
       MOV    #tblpage(PROG_ADDR),W0
       MOV    W0,NVMADRU
       MOV    #tbloffset(PROG_ADDR),W0
       MOV    W0,NVMADR
       TBLWTL w0,[w0]
; Setup NVMCON to erase one row of Program Memory
       MOV    #0x4042,W0
       MOV    W0,NVMCON
; Disable interrupts while the KEY sequence is written
       PUSH   SR
       MOV    #0x00E0,W0
       IOR    SR
; Write the KEY Sequence
       MOV    #0x55,W0
       MOV    W0,NVMKEY
       MOV    #0xAA,W0
       MOV    W0,NVMKEY
; Start the erase operation
       BSET   NVMCON,#15
; Insert two NOPs after the erase cycle (required)
       NOP
       NOP
;Re-enable interrupts, if needed
       POP    SR
```

© 2011 Microchip Technology Inc.

5.4.2.3 LOADING WRITE LATCHES

The write latches are used as a storage mechanism between the user application table writes and the actual row programming sequence. Example 5-5 shows the sequence of instructions that can be used to load 128 write latches (128 instruction words). 128 `TBLWTL` and 128 `TBLWTH` instructions are needed to load the write latches for programming a row of Flash program memory.

The row of 128 instruction words does not necessarily have to be written in sequential order. The 8 LSbs of the table write address determine which of the latches will be written. However, all 128 instruction words should be written for each programming cycle in order to overwrite old data.

The Flash program memory must be programmed at an “even” 128 instruction word address boundary. In effect, the 8 LSbs of the table write operation select the write latches to program the instruction word within the row. They have no effect when a row is programmed.

Note: The code shown in Example 5-5 is the `Load_Write_Latch_Row` and the code shown in Example 5-6 is the `Load_Write_Latch_Word`. The code shown in both of these examples is referred to in subsequent examples.

Example 5-5: Loading Write Latches for Row Programming

```
Load_Write_Latch_Row:
; Set up a pointer to the first latch location to be written.
       MOV     #0xFA,W0
       MOV     W0,TBLPAG
       MOV     #0,W1

; Perform the TBLWT instructions to write the latches
; W2 is incremented in the TBLWTH instruction to point to the
; next instruction location.
       MOV     #128,W3

   loop:
   TBLWTL.b    [W2++], [W1++]
   TBLWTL.b    [W2++], [W1--]
   TBLWTH.b    [W2++], [W1]
   INC2        W1, W1
   DEC         W3, W3
   BRA         NZ, loop
```

Example 5-6: Loading Write Latches for Word Programming

```
Load_Write_Latch_Word:
; Define the start address from where the programming has to start
    .equ PROG_ADDR, 0x022222
; Set up a pointer to the first latch location to be written.
    MOV     #0xFA,W0
    MOV     W0,TBLPAG
    MOV     #0,W1

; Perform the TBLWT instructions to write the latches
    TBLWTL  [W2++],[W1]
    TBLWTH  [W2++],[W1++]
    TBLWTL  [W2++],[W1]
    TBLWTH  [W2++],[W1++]
```

5.4.2.4 SINGLE ROW PROGRAMMING EXAMPLE

The NVMCON register is configured to program one row of Flash program memory. The program operation is initiated by writing a special unlock, or key, sequence to the NVMKEY register before setting the WR control bit (NVMCON<15>). The unlock sequence needs to be executed in the exact order as shown in Example 5-7, without interruption. Therefore, interrupts should be disabled prior to writing the sequence.

Two `NOP` instructions should be inserted in the code after the programming cycle. Finally, interrupts can be enabled (if required).

Example 5-7: Single Row Programming

```
; Define the start address from where the programming has to start
       .equ PROG_ADDR, 0x022222
; Load the destination address to be written
       MOV     #tblpage(PROG_ADDR),W9
       MOV     #tbloffset(PROG_ADDR),W8
       MOV     W9,NVMADRU
       MOV     W8,NVMADR
; Setup NVMCON to write 1 row of program memory
       MOV     #0x4002,W0
       MOV     W0,NVMCON
; Load the 64 program memory write latches
       CALL    Load_Write_Latch_Row
; Disable interrupts, if enabled
       PUSH    SR
       MOV     #0x00E0,W0
       IOR     SR
; Write the KEY sequence
       MOV     #0x55,W0
       MOV     W0,NVMKEY
       MOV     #0xAA,W0
       MOV     W0,NVMKEY
; Start the programming sequence
       BSET    NVMCON,#15
; Insert two NOPs after programming
       NOP
       NOP
; Re-enable interrupts, if required
       POP     SR
```

Note: Refer to **5.4.2.3 "Loading Write Latches"** for additional information.

© 2011 Microchip Technology Inc.

5.4.2.5 WORD PROGRAMMING

Assuming the user application has erased the Flash location to be programmed, use table write instructions to write a pair of words into the first two word memory locations in the write latch memory area. This is slightly different from the existing Word programming for the dsPIC33F devices where only one word memory location is used to write into the write latch memory area. Load the address of the word of Flash memory to be programmed into the NVM address registers.

The NVMCON register is configured to program a pair of words of Flash program memory. The program operation is initiated by writing a special unlock, or key, sequence to the NVMKEY register before setting the WR control bit (NVMCON<15>). The unlock sequence needs to be executed in the exact order shown in Example 5-8, without interruption. Therefore, interrupts should be disabled prior to writing the sequence.

Two `NOP` instructions should be inserted in the code after the programming cycle. Finally, interrupts can be enabled (if required).

Example 5-8: Programming Two Words of Flash Memory

```
; Load the two words into the latches
       CALL Load_Write_Latch_Word
; Setup NVMCON for word programming
       MOV    #0x4001,W0
       MOV    W0,NVMCON
; Disable interrupts while the KEY sequence is written
       PUSH   SR
       MOV    #0x00E0,W0
       IOR    SR
; Write the key sequence
       MOV    #0x55,W0
       MOV    W0,NVMKEY
       MOV    #0xAA,W0
       MOV    W0,NVMKEY
; Start the write cycle
       BSET   NVMCON,#15
;Re-enable interrupts, if needed
       POP    SR
```

5.4.3 Writing to Device Configuration Registers

The RTSP can be used to write to the device Configuration registers. RTSP allows each Configuration register to be individually rewritten without first performing an erase cycle. Caution must be exercised when writing the Configuration registers since they control critical device operating parameters, such as the system clock source, PLL, and WDT enable.

The procedure for programming a device Configuration register is similar to the procedure for programming Flash program memory, except that only `TBLWTL` instructions are required. This is because the upper eight bits in each device Configuration register are unused. Furthermore, bit 23 of the table write address must be set to access the Configuration registers. Refer to **Section 30. "Device Configuration"** (DS70618) in the "*dsPIC33E/PIC24E Family Reference Manual*" and the specific device data sheet for a full description of the device Configuration registers.

Note 1: Writing to device Configuration registers is not available in all devices. Refer to the specific device data sheet to determine the modes that are available according to the device-specific NVMOP<3:0> bits definition.

2: While performing RTSP on device Configuration registers, the device must be operating using the Internal FRC Oscillator (without PLL). If the device is operating from a different clock source, a clock switch to the Internal FRC Oscillator (NOSC<2:0> = `000`) must be performed prior to performing RTSP operation in Device Configuration registers.

3: If the Primary Oscillator Mode Select bits (POSCMD<1:0>) in the Oscillator Configuration register (FOSC) are being reprogrammed to a new value, the user must ensure that the Clock Switching Mode bits (FCKSM<1:0>) in the FOSC register have an initial programmed value of '`00`', prior to performing this RTSP operation.

5.4.3.1 CONFIGURATION REGISTER WRITE ALGORITHM

The general procedure is as follows:

1. Store the value to be programmed into data RAM as data "image".
2. Write the new configuration value to the table write latch using a `TBLWTL` instruction.
3. Configure NVMCON for a Configuration register write (NVMCON = 0x4000).
4. Disable interrupts, if enabled.
5. Write the address of the Configuration register to be programmed into the NVMADRU and NVMADR registers.
6. Write the key sequence to NVMKEY.
7. Start the write sequence by setting WR (NVMCON<15>).
8. Re-enable interrupts, if needed.

Example 5-9 shows the code sequence that can be used to modify a device Configuration register.

© 2011 Microchip Technology Inc.

Example 5-9: Configuration Register Write Code Example

```
; Define the address to be written
       .equ DestinationAddress, 0x022222
; Initialize the write pointer for writing to the latches
       MOV     #0x0000, W7
; Initialize TBLPAG register for writing to the latches
       MOV     #0xFA, W12
       MOV     W12, TBLPAG
; Get the new data to write to the configuration register
       MOV     #ConfigValue,W1
; Perform the table write to load the write latch
       TBLWTL  W1,[W7]
; Load the address which is to be programmed
       MOV     #DestinationAddress<15:0>,W2
       MOV     #DestinationAddress<23:16>,W3
       MOV     W3,NVMADRU
       MOV     W2,NVMADR
; Configure NVMCON for a configuration register write
       MOV     #0x4000,W0
       MOV     W0,NVMCON
; Disable interrupts, if enabled
       PUSH    SR
       MOV     #0x00E0,W0
       IOR     SR
; Write the KEY sequence
       MOV     #0x55,W0
       MOV     W0,NVMKEY
       MOV     #0xAA,W0
       MOV     W0,NVMKEY
; Start the programming sequence
       BSET    NVMCON,#15
; Insert two NOPs after programming
       NOP
       NOP
; Re-enable interrupts, if required
       POP     SR
```

5.5 REGISTER MAP

A summary of the registers associated with Flash Programming is provided in Table 5-1.

Table 5-1: Flash Programming Registers

File Name	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
NVMCON	WR	WREN	WRERR	NVMSIDL	—	—	—	—	—	—	—	—	NVMOP<3:0>				0000
NVMADRU	—	—	—	—	—	—	—	—	NVMADRU<7:0>								0000
NVMADR	NVMADR<15:0>																0000
NVMKEY	—	—	—	—	—	—	—	—	NVMKEY<7:0>								0000

Legend: x = unknown value on Reset, — = unimplemented, read as ‘0’. Reset values are shown in hexadecimal.
Note 1: Not all bits are available for all devices. Please refer to the specific device data sheet for details.

© 2011 Microchip Technology Inc.

5.6 RELATED APPLICATION NOTES

This section lists application notes that are related to this section of the manual. These application notes may not be written specifically for the dsPIC33E/PIC24E product families, but the concepts are pertinent and could be used with modification and possible limitations. The current application notes related to the Flash Programming module are:

Title	**Application Note #**
No related application notes at this time	

Note:	Please visit the Microchip web site (www.microchip.com) for additional Application Notes and code examples for the dsPIC33E/PIC24E family of devices.

5.7 REVISION HISTORY

Revision A (August 2009)

This is the initial released version of this document.

Revision B (February 2011)

This revision includes the following updates:

- Examples:
  - Removed Example 5-3 and Example 5-4
  - Updated Example 5-4, Example 5-7 and Example 5-9
  - Any references to `#WR` were updated to `#15` in Example 5-4, Example 5-7 and Example 5-8
  - Updated the following in Example 5-5:
    - Updated the title “Word Programming” to “Loading Write Latches for Row Programming”
    - Any reference to `#ram_image` was updated to `#0xFA`
  - Added Example 5-6
  - Updated the title in Example 5-8
- Notes:
  - Added two notes in **5.4.2 “Flash Programming Operations”**
  - Updated the note in **5.4.2.3 “Loading Write Latches”**
  - Added three notes in **5.4.3 “Writing to Device Configuration Registers”**
  - Added Note 1 in Table 5-1
- Registers:
  - Updated the bit values for NVMOP<3:0>: NVM Operation Select bits in the Flash Memory Control (NVMCON) register (see Register 5-1)
- Sections:
  - Removed sections **5.2.1.4 “Write Word Mode”** and **5.2.1.5 “Write Byte Mode”**
  - Updated **5.3 “Control Registers”**
  - Updated the following in **5.4.2.5 “Word Programming”**:
    - Changed the section title “Programming One Word of Flash Memory“ to “Word Programming”
    - Updated the first paragraph
    - Changed the terms “one word“ to “a pair of words” in the second paragraph
  - Added a new Step 1 to **5.4.3.1 “Configuration Register Write Algorithm”**
- Tables:
  - Updated Table 5-1
- A few references to program memory were updated to Flash program memory
- Other minor updates such as language and formatting updates were incorporated throughout the document

© 2011 Microchip Technology Inc.

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, dsPIC, KEELOQ, KEELOQ logo, MPLAB, PIC, PICmicro, PICSTART, PIC^{32} logo, rfPIC and UNI/O are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

FilterLab, Hampshire, HI-TECH C, Linear Active Thermistor, MXDEV, MXLAB, SEEVAL 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, HI-TIDE, In-Circuit Serial Programming, ICSP, Mindi, MiWi, MPASM, MPLAB Certified logo, MPLIB, MPLINK, mTouch, Omniscient Code Generation, PICC, PICC-18, PICDEM, PICDEM.net, PICkit, PICtail, REAL ICE, rfLAB, Select Mode, Total Endurance, TSHARC, UniWinDriver, 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.

© 2011, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

Printed on recycled paper.

ISBN: 978-1-60932-917-4

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

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

MICROCHIP

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://www.microchip.com/support
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Cleveland
Independence, OH
Tel: 216-447-0464
Fax: 216-447-0643

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Farmington Hills, MI
Tel: 248-538-2250
Fax: 248-538-2260

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453

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

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

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

ASIA/PACIFIC

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

ASIA/PACIFIC

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

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

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

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

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

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

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

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

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

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

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

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

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

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

EUROPE

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

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

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

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

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

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

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

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

02/18/11

© 2011 Microchip Technology Inc.