

## PIC16F/LF720/721 Flash Memory Programming Specification

This document includes the programming specifications for the following devices:

- PIC16F720
- PIC16LF720
- PIC16F721
- PIC16LF721

### 1.0 OVERVIEW

The PIC16F/LF720 and PIC16F/LF721 devices are programmed using In-Circuit Serial Programming™ (ICSP™). This programming specification applies to the PIC16F/LF720 and PIC16F/LF721 devices in all packages.

With the exception of memory size and the voltage regulator, all other aspects of the PIC16F/LF720 and PIC16F/LF721 devices are identical.

### 1.1 Hardware Requirements

PIC16F/LF720 and PIC16F/LF721 devices require one power supply for VDD and one for  $\overline{\text{MCLR}}/\text{VPP}$ . (See Section 8.0 “Electrical Specifications” for more details.)

### 1.2 Pin Utilization

Five pins are needed for ICSP™ programming. The pins are listed in Table 1-1.

**TABLE 1-1: PIN DESCRIPTIONS DURING PROGRAMMING**

| Pin Name                            | During Programming  |                  |  |
|-------------------------------------|---------------------|------------------|--|
|                                     | Function            | Pin Type         | Pin Description                              |
| RA1                                 | ICSPCLK             | I                | Clock Input – Schmitt Trigger Input          |
| RA0                                 | ICSPDAT             | I/O              | Data Input/Output – Schmitt Trigger Input    |
| $\overline{\text{MCLR}}/\text{VPP}$ | Program/Verify mode | P <sup>(1)</sup> | Program Mode Select/Programming power supply |
| VDD                                 | VDD                 | P                | Power Supply                                 |
| VSS                                 | VSS                 | P                | Ground                                       |

**Legend:** I = Input, O = Output, P = Power

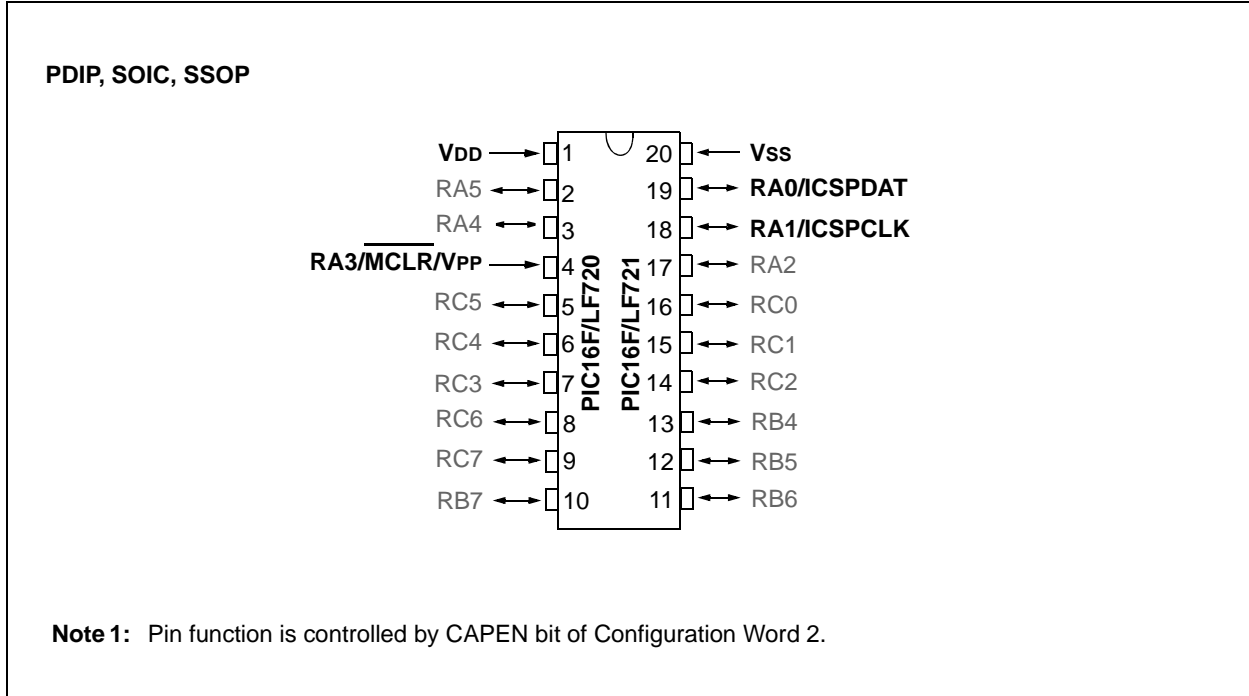
**Note 1:** To activate the Program/Verify mode, high voltage needs to be applied to  $\overline{\text{MCLR}}/\text{VPP}$  input. Since the  $\overline{\text{MCLR}}/\text{VPP}$  is used for a level source,  $\overline{\text{MCLR}}/\text{VPP}$  does not draw any significant current.

# PIC16F/LF720/721

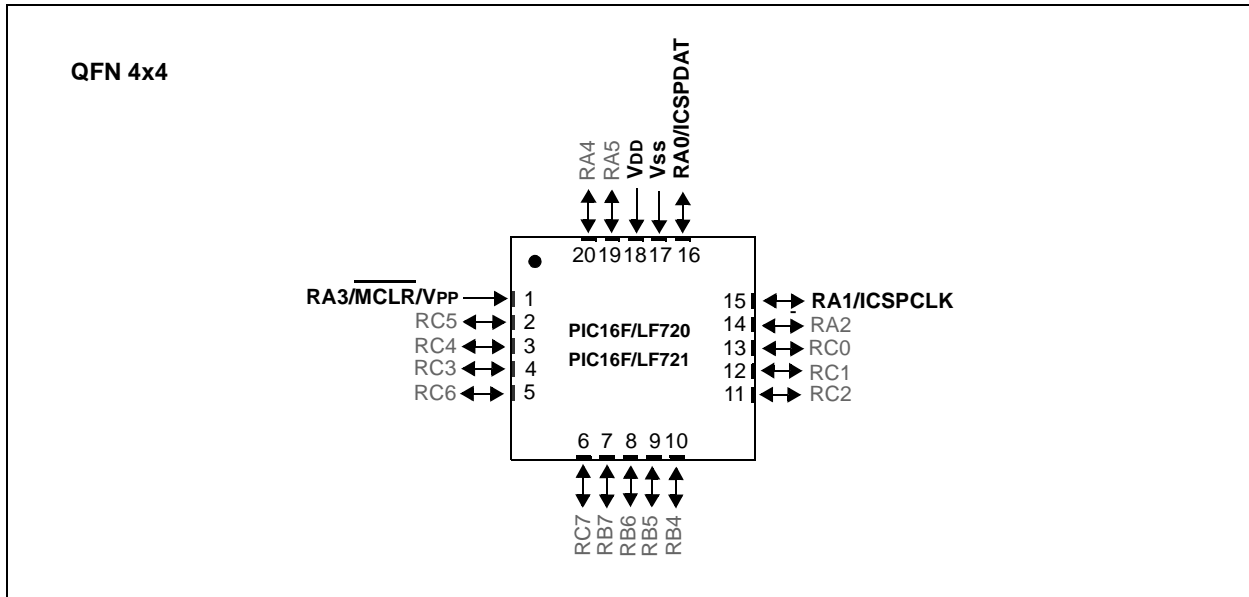
## 2.0 DEVICE PINOUTS

The pin diagrams for the PIC16F/LF720 and PIC16F/LF721 family are shown in Figure 2-1 and Figure 2-2. The pins that are required for programming are listed in Table 1-1 and shown in bold lettering in the pin diagrams.

**FIGURE 2-1: 20-PIN DIAGRAM FOR PIC16F/LF720 AND PIC16F/LF721**



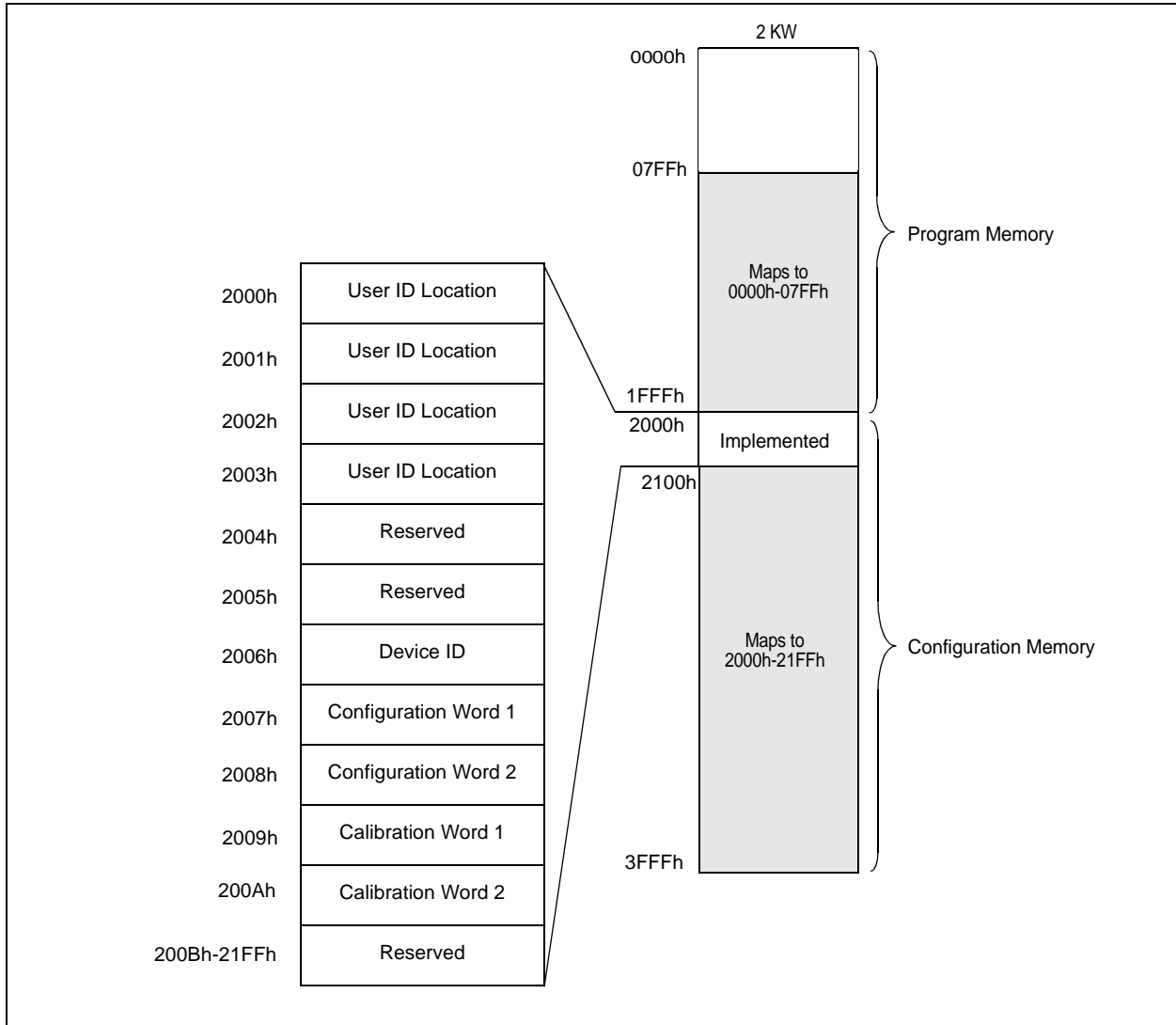
**FIGURE 2-2: 20-PIN DIAGRAM FOR PIC16F/LF720 AND PIC16F/LF721**



## 3.0 MEMORY MAP

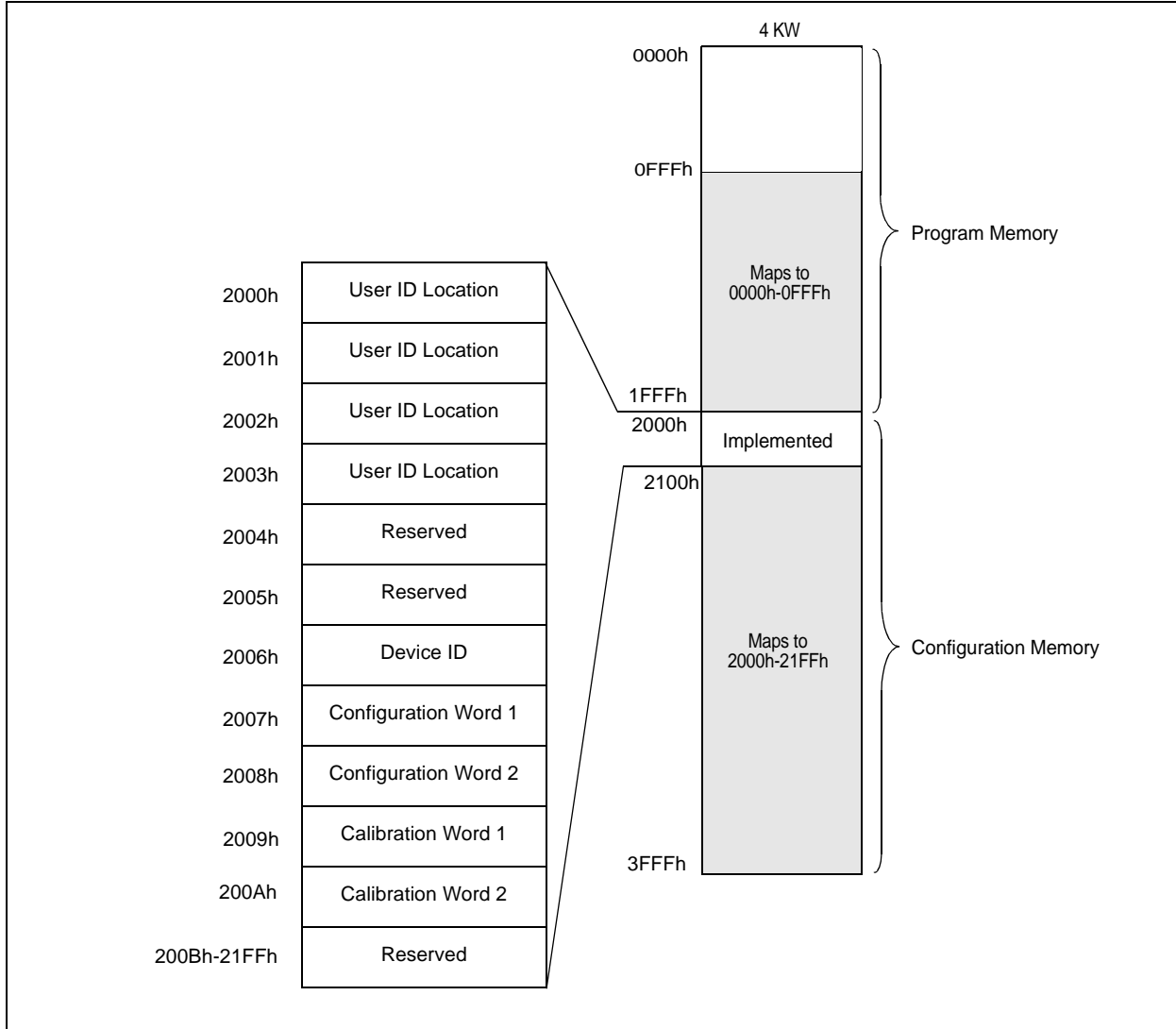
The memory for the PIC16F/LF720 and PIC16F/LF721 devices is broken into two sections: program memory and configuration memory. The size of the program memory and the configuration memory is different between devices.

**FIGURE 3-1: PIC16F720 AND PIC16LF720 PROGRAM MEMORY MAPPING**



# PIC16F/LF720/721

**FIGURE 3-2: PIC16F721 AND PIC16LF721 PROGRAM MEMORY MAPPING**



## 3.1 User ID Location

A user may store identification information (user ID) in four designated locations. The user ID locations are mapped to 2000h-2003h. Each location is 14 bits in length. Code protection has no effect on these memory locations. Each location may be read with code protection enabled or disabled.

**Note:** MPLAB® IDE only displays the 7 Least Significant bits (LSb) of each user ID location, the upper bits are not read. It is recommended that only the 7 LSbs be used if MPLAB IDE is the primary tool used to read these addresses.

## 3.2 Device ID

The device ID word for the PIC16F/LF720 and the PIC16F/LF721 is located at 2006h. This location cannot be erased or modified.

**REGISTER 3-1: DEVICEID: DEVICE ID REGISTER<sup>(1)</sup>**

|        |      |      |      |      |      |       |
|--------|------|------|------|------|------|-------|
| R-q    | R-q  | R-q  | R-q  | R-q  | R-q  | R-q   |
| DEV8   | DEV7 | DEV6 | DEV5 | DEV4 | DEV3 | DEV2  |
| bit 13 |      |      |      |      |      | bit 7 |
|        |      |      |      |      |      |       |
| R-q    | R-q  | R-q  | R-q  | R-q  | R-q  | R-q   |
| DEV1   | DEV0 | REV4 | REV3 | REV2 | REV1 | REV0  |
| bit 6  |      |      |      |      |      | bit 0 |

|                   |                      |                                    |
|-------------------|----------------------|------------------------------------|
| <b>Legend:</b>    | P = Programmable bit | U = Unimplemented bit, read as '0' |
| R = Readable bit  | W = Writable bit     | '0' = Bit is cleared               |
| -n = Value at POR | '1' = Bit is set     | x = Bit is unknown                 |

bit 13-5      **DEV<8:0>**: Device ID bits  
 These bits are used to identify the part number.

bit 4-0      **REV<4:0>**: Revision ID bits  
 These bits are used to identify the revision.

**Note 1:** This location cannot be written.

**TABLE 3-1: DEVICE ID VALUES**

| DEVICE     | DEVICE ID VALUES |         |
|------------|------------------|---------|
|            | DEV              | REV     |
| PIC16F720  | 01 1100 0000     | x xxxxx |
| PIC16F721  | 01 1100 0010     | x xxxxx |
| PIC16LF720 | 01 1100 0100     | x xxxxx |
| PIC16LF721 | 01 1100 0110     | x xxxxx |

# PIC16F/LF720/721

---

## 3.3 Configuration Words

The PIC16F/LF720 and PIC16F/LF721 have two Configuration Words, Configuration Word 1 (2007h) and Configuration Word 2 (2008h). The individual bits within these Configuration Words are used to enable or disable device functions such as the Brown-out Reset, code protection and Power-up Timer.

## 3.4 Calibration Words

For the PIC16F/LF720 and PIC16F/LF721 devices, the 16 MHz internal oscillator (INTOSC) and the Brown-out Reset (BOR) are factory calibrated and stored in Calibration Words 1 and 2 (2009h and 200Ah).

The Calibration Words do not participate in erase operations. The device can be erased without affecting the Calibration Words.

## REGISTER 3-2: CONFIGURATION WORD 1

| R/P-1                           | R/P-1 | U-1 | U-1 | R/P-0  | R/P-0  | U-1   |
|---------------------------------|-------|-----|-----|--------|--------|-------|
| $\overline{\text{DEBUG}}^{(1)}$ | PLLEN | —   | —   | BOREN1 | BOREN0 | —     |
| bit 13                          |       |     |     |        |        | bit 7 |

| R/P-1                  | R/P-1 | R/P-1                                       | R/P-1 | U-1 | R/P-1 | R/P-1 |
|------------------------|-------|---|-------|-----|-------|-------|
| $\overline{\text{CP}}$ | MCLRE | $\overline{\text{PWRT}}\overline{\text{E}}$ | WDTEN | —   | FOSC1 | FOSC0 |
| bit 6                  |       |   |       |     |       | bit 0 |

|                             |                                    |                    |
|-----------------------------|------------------------------------|--------------------|
| <b>Legend:</b>              | W = Writable bit                   | 0 = Bit is cleared |
| R = Readable bit            | 1 = Bit is set                     | x = Bit is unknown |
| -n = Value for blank device | U = Unimplemented bit, read as '1' |                    |

- bit 13  **$\overline{\text{DEBUG}}^{(1)}$** : Debugger Mode bit  
 0 = Background debugger is enabled  
 1 = Background debugger is disabled
- bit 12 **PLLEN**: INTOSC PLL Enable bit  
 0 = INTOSC Frequency is 500 kHz  
 1 = INTOSC Frequency is 16 MHz (32x)
- bit 11-10 **Unimplemented**: Read as '1'
- bit 9-8 **BOREN<1:0>**: Brown-out Reset Enable bits <sup>(2)</sup>  
 0x = Brown-out Reset disabled (Preconditioned State)  
 10 = Brown-out Reset enabled during operation and disabled in Sleep  
 11 = Brown-out Reset enabled
- bit 7 **Unimplemented**: Read as '1'
- bit 6 **CP**: Flash Program Memory Code Protection bit  
 PIC16F720/721  
 0 = 0000h to 07FFh/0FFFh code protection on  
 1 = Code protection off
- bit 5 **MCLRE**: RA3/MCLR/VPP Pin Function Select bit  
 1 = RA3/MCLR/VPP pin function is MCLR; Weak pull-up enabled.  
 0 = RA3/MCLR/VPP pin function is digital input; MCLR internally disabled; Weak pull-up disabled
- bit 4 **PWRT**: Power-up Timer Enable bit  
 0 = PWRT enabled  
 1 = PWRT disabled
- bit 3 **WDTEN**: Watchdog Timer Enable bit  
 0 = WDT disabled  
 1 = WDT enabled
- bit 2 **Unimplemented**: Read as '1'
- bit 1-0 **FOSC<1:0>**: Oscillator Selection bits  
 11 = EC oscillator: CLKO function on RA4/CLKO pin, CLKI on RA5/CLKI  
 10 = EC oscillator: I/O function on RA4/CLKO pin, CLKI on RA5/CLKI  
 01 = INTOSC oscillator: CLKO function on RA4/CLKO pin, I/O function on RA5/CLKI  
 00 = INTOSCIO oscillator: I/O function on RA4/CLKO pin, I/O function on RA5/CLKI

**Note 1:** Debug bit is ignored when code-protect is enabled ( $\overline{\text{CP}}= 0$ ).

**2:** Fixed Voltage Reference is automatically enabled whenever the BOR is enabled.

# PIC16F/LF720/721

## REGISTER 3-3: CONFIGURATION WORD 2

|        |     |     |     |     |     |       |
|--------|-----|-----|-----|-----|-----|-------|
| U-1    | U-1 | U-1 | U-1 | U-1 | U-1 | U-1   |
| —      | —   | —   | —   | —   | —   | —     |
| bit 13 |     |     |     |     |     | bit 7 |

|       |     |        |     |     |       |       |
|-------|-----|--------|-----|-----|-------|-------|
| U-1   | U-1 | R/P-1  | U-1 | U-1 | R/P-1 | R/P-1 |
| —     | —   | VCAPEN | —   | —   | WRT1  | WRT0  |
| bit 6 |     |        |     |     |       | bit 0 |

|                             |                                    |                    |
|-----------------------------|------------------------------------|--------------------|
| <b>Legend:</b>              | W = Writable bit                   | 0 = Bit is cleared |
| R = Readable bit            | 1 = Bit is set                     | x = Bit is unknown |
| -n = Value for blank device | U = Unimplemented bit, read as '1' |                    |

bit 13-5 **Unimplemented:** Read as '1'

bit 4 **VCAPEN<sup>(1)</sup>:** Voltage Regulator Capacitor Enable bits  
 0 = VCAP functionality is enabled on RA2. (VDDCORE is connected to the pad)  
 1 = All VCAP pin functions are disabled (Erased or Preconditioned State)

bit 3-2 **Unimplemented:** Read as '1'

bit 1-0 **WRT<1:0>:** Flash Memory Self-Write Protection bits

2 kW Flash memory: PIC16F/LF720 :

- 11 = Write protection off
- 10 = 000h to 1FFh write protected, 200h to 7FFh may be modified by PMCON control
- 01 = 000h to 3FFh write protected, 400h to 7FFh may be modified by PMCON control
- 00 = 000h to 7FFh write protected, no addresses may be modified by PMCON control

4 kW Flash memory: PIC16F/LF721:

- 11 = Write protection off
- 10 = 000h to 1FFh write protected, 200h to FFFh may be modified by PMCON control
- 01 = 000h to 7FFh write protected, 800h to FFFh may be modified by PMCON control
- 00 = 000h to FFFh write protected, no addresses may be modified by PMCON control

**Note 1:** For the PIC16F720/721 only.



## 4.0 PROGRAM/VERIFY MODE

In Program/Verify mode, the program memory and the configuration memory can be accessed and programmed in serial fashion. ICSPDAT and ICSPCLK are used for the data and the clock, respectively. All commands and data words are transmitted LSb first. Data changes on the rising edge of the ICSPCLK and latched on the falling edge. In Program/Verify mode, both the ICSPDAT and ICSPCLK are Schmitt Trigger inputs. The sequence that enters the device into Program/Verify mode places all other logic into the Reset state. Upon entering Program/Verify mode, all I/Os are automatically configured as high-impedance inputs and the address is cleared.

### 4.1 Program/Verify Mode Entry and Exit

There are two different methods of entering Program/Verify mode:

- VPP – First entry mode
- VDD – First entry mode

#### 4.1.1 VPP – FIRST ENTRY MODE

To enter Program/Verify mode via the VPP-first method the following sequence must be followed:

1. Hold ICSPCLK and ICSPDAT low. All other pins should be unpowered.
2. Raise the voltage on  $\overline{\text{MCLR}}$  from 0V to  $V_{IH}$ .
3. Raise the voltage on VDD from 0V to the desired operating voltage.

The VPP-first entry prevents the device from executing code prior to entering Program/Verify mode. For example, when the Configuration Word has  $\overline{\text{MCLR}}$

disabled ( $\text{MCLRE} = 0$ ), the power-up time is disabled ( $\text{PWRTE} = 0$ ), the internal oscillator is selected ( $\text{FOSC} = 10x$ ), and RA0 and RA1 are driven by the user application, the device will execute code. Since this may prevent entry, VPP-First Entry mode is strongly recommended. See the timing diagram in Figure 8-2.

#### 4.1.2 VDD – FIRST ENTRY MODE

To enter Program/Verify mode via the VDD-first method, the following sequence must be followed:

1. Hold ICSPCLK and ICSPDAT low.
2. Raise the voltage on VDD from 0V to the desired operating voltage.
3. Raise the voltage on  $\overline{\text{MCLR}}$  from VDD or below to  $V_{IH}$ .

The VDD-first method is useful when programming the device, when VDD is already applied, for it is not necessary to disconnect VDD to enter Program/Verify mode. See the timing diagram in Figure 8-1.

#### 4.1.3 PROGRAM/VERIFY MODE EXIT

To exit Program/Verify mode take  $\overline{\text{MCLR}}$  to VDD or lower ( $V_{IL}$ ). See Figures 8-3 and 8-4.

## 4.2 Program/Verify Commands

The PIC16F/LF720 and PIC16F/LF721 implement 10 programming commands, each six bits in length. The commands are summarized in Table 4-1.

Commands that have data associated with them are specified to have a minimum delay of TDLY between the command and the data. After this delay 16 clocks are required to either clock in or clock out the 14-bit data word. The first clock is for the Start bit and the last clock is for the Stop bit.

**TABLE 4-1: COMMAND MAPPING FOR PIC16F/LF720 AND PIC16F/LF721**

| Command                            | Mapping              |   |   |   |   |   | Data/Note |                  |
|------------------------------------|----------------------|---|---|---|---|---|-----------|------------------|
|                                    | Binary (MSb ... LSb) |   |   |   |   |   |           | Hex              |
| Load Configuration                 | x                    | 0 | 0 | 0 | 0 | 0 | 00h       | 0, data (14), 0  |
| Load Data For Program Memory       | x                    | 0 | 0 | 0 | 1 | 0 | 02h       | 0, data (14), 0  |
| Read Data From Program Memory      | x                    | 0 | 0 | 1 | 0 | 0 | 04h       | 0, data (14), 0  |
| Increment Address                  | x                    | 0 | 0 | 1 | 1 | 0 | 06h       |                  |
| Reset Address                      | x                    | 1 | 0 | 1 | 1 | 0 | 16h       |                  |
| Begin Internally Timed Programming | x                    | 0 | 1 | 0 | 0 | 0 | 08h       |                  |
| Begin Externally Timed Programming | x                    | 1 | 1 | 0 | 0 | 0 | 18h       |                  |
| End Externally Timed Programming   | x                    | 0 | 1 | 0 | 1 | 0 | 0Ah       |                  |
| Bulk Erase Program Memory          | x                    | 0 | 1 | 0 | 0 | 1 | 09h       | Internally Timed |
| Row Erase Program Memory           | x                    | 1 | 0 | 0 | 0 | 1 | 11h       | Internally Timed |

# PIC16F/LF720/721

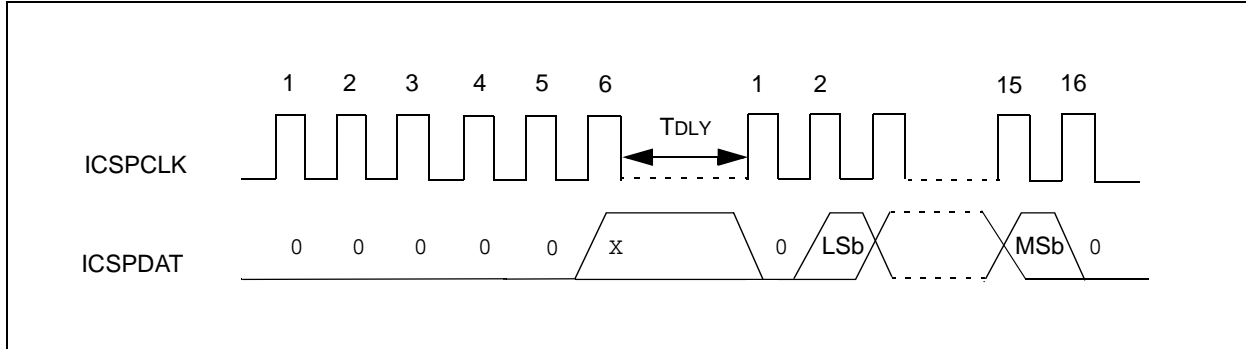
## 4.2.1 LOAD CONFIGURATION

The Load Configuration command is used to access the configuration memory (User ID Locations, Configuration Words, Calibration Words). The Load Configuration command sets the address to 2000h and loads the data latches with one word of data (see Figure 4-1).

After issuing the Load Configuration command, use the Increment Address command until the proper address to be programmed is reached. The address is then programmed by issuing either the Begin Internally Timed Programming or Begin Externally Timed Programming command.

The only way to get back to the program memory (address 0) is to exit Program/Verify mode or issue the Reset Address command after the configuration memory has been accessed by the Load Configuration command.

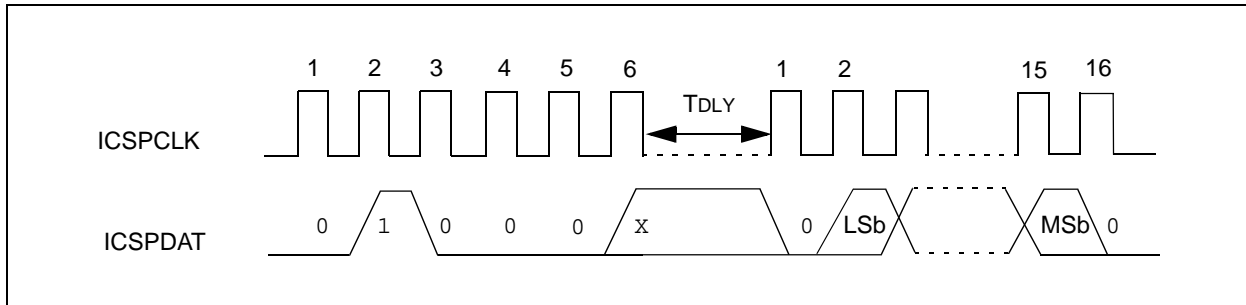
**FIGURE 4-1: LOAD CONFIGURATION**



## 4.2.2 LOAD DATA FOR PROGRAM MEMORY

The Load Data for Program Memory command is used to load one 14-bit word into the data latches. The word programs into program memory after the Begin Internally Timed Programming or Begin Externally Timed Programming command is issued (see Figure 4-2).

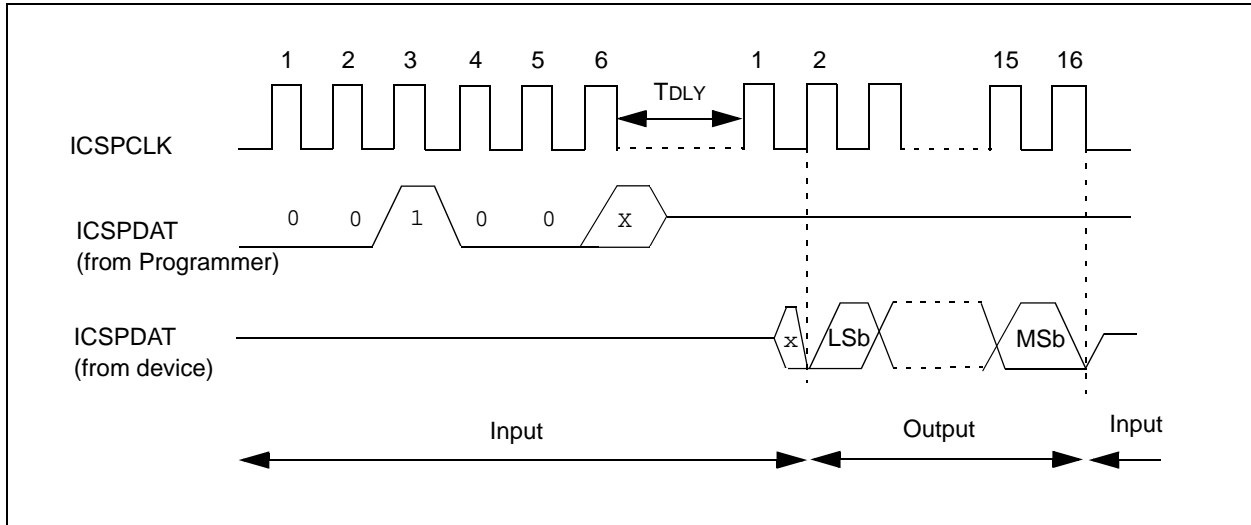
**FIGURE 4-2: LOAD DATA FOR PROGRAM MEMORY**



## 4.2.3 READ DATA FROM PROGRAM MEMORY

The Read Data from Program Memory command will transmit data bits out of the program memory map currently accessed, starting with the second rising edge of the clock input. The ICSPDAT pin will go into Output mode on the first falling clock edge, and it will revert to Input mode (high-impedance) after the 16th falling edge of the clock. If the program memory is code-protected (CP), the data will be read as zeros (see Figure 4-3).

**FIGURE 4-3: READ DATA FROM PROGRAM MEMORY**

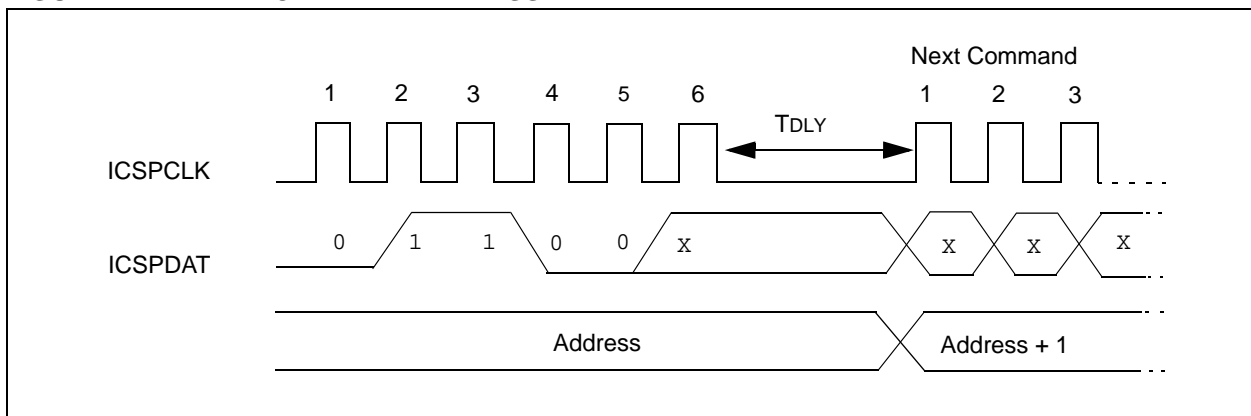


## 4.2.4 INCREMENT ADDRESS

The address is incremented when this command is received. It is not possible to decrement the address. To reset this counter, the user must use the Reset Address command or exit Program/Verify mode and re-enter it.

If the address is incremented from address 1FFFh, it will wrap-around to location 0000h. If the address is incremented from 3FFFh, it will wrap-around to location 2000h.

**FIGURE 4-4: INCREMENT ADDRESS**

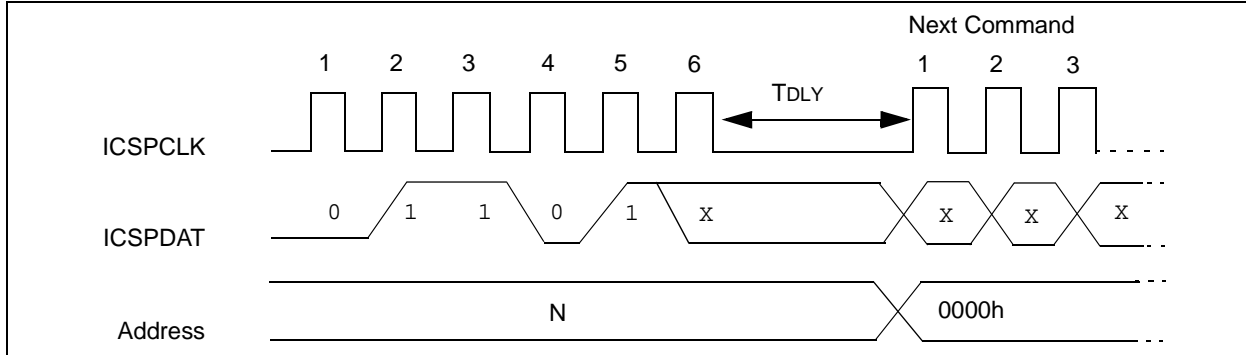


# PIC16F/LF720/721

## 4.2.5 RESET ADDRESS

The Reset Address command will reset the address to 0000h, regardless of the current value. The address is used in program memory or the configuration memory.

**FIGURE 4-5: RESET ADDRESS**



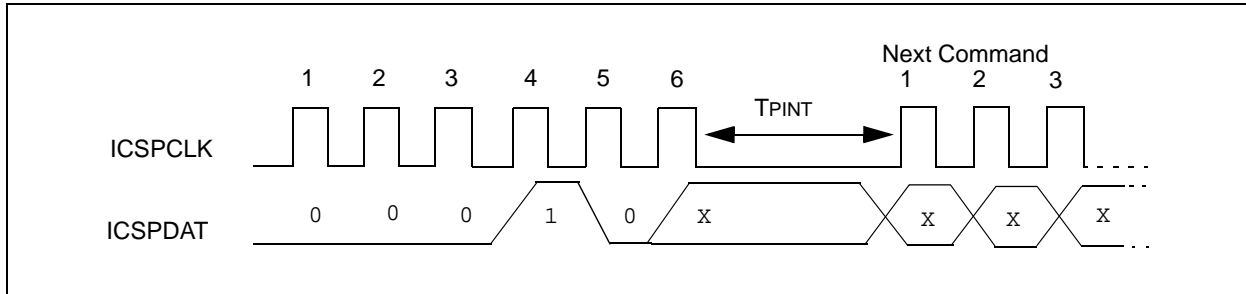
## 4.2.6 BEGIN INTERNALLY TIMED PROGRAMMING

A Load Configuration or Load Data for Program Memory command must be given before every Begin Programming command. Programming of the addressed memory will begin after this command is received. An internal timing mechanism executes the write. The user must allow for the program cycle time, TPINT, for the programming to complete.

The End Externally Timed Programming command is not needed when the Begin Internally Timed Programming is used to start the programming.

The program memory address that is being programmed is not erased prior to being programmed.

**FIGURE 4-6: BEGIN INTERNALLY TIMED PROGRAMMING**

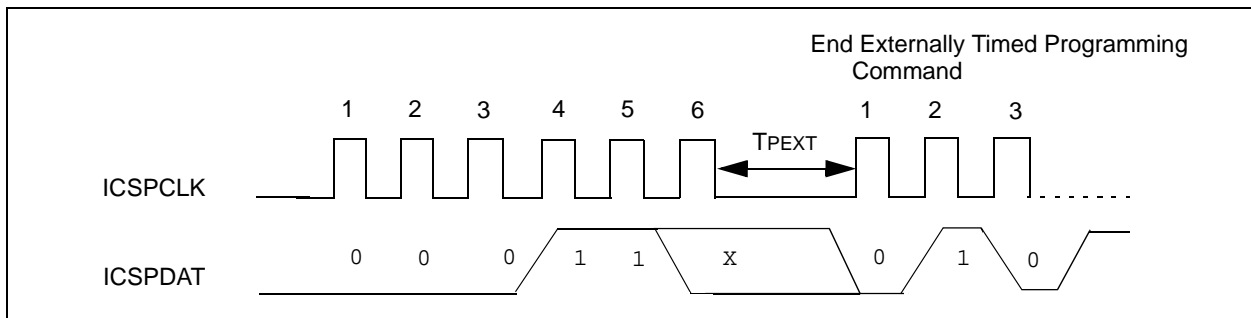


## 4.2.7 BEGIN EXTERNALLY TIMED PROGRAMMING

A Load Configuration or Load Data for Program Memory command must be given before every Begin Programming command. Programming of the addressed memory will begin after this command is received. To complete the programming, the End Externally Timed Programming command must be sent in the specified time window defined by  $T_{PEXT}$ . The program memory address that is being programmed is not erased prior to being programmed.

The Begin Externally Timed Programming command cannot be used for programming the Configuration Words (see Figure 4-7).

**FIGURE 4-7: BEGIN EXTERNALLY TIMED PROGRAMMING**

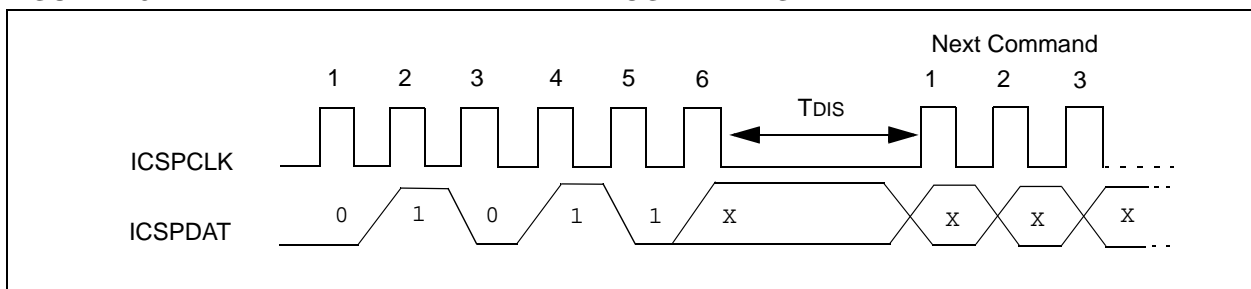


## 4.2.8 END EXTERNALLY TIMED PROGRAMMING

This command is required after a Begin Externally Timed Programming command is given. This command must be sent within the time window specified by  $T_{PEXT}$  after the Begin Externally Timed Programming command is sent.

After sending the End Externally Timed Programming command, an additional delay ( $T_{DIS}$ ) is required before sending the next command. This delay is longer than the delay ordinarily required between other commands (see Figure 4-8).

**FIGURE 4-8: END EXTERNALLY TIMED PROGRAMMING**



# PIC16F/LF720/721

## 4.2.9 BULK ERASE PROGRAM MEMORY

The Bulk Erase Program Memory command performs two different functions dependent on the current state of the address.

Address 0000h-1FFFh:

- Program Memory is erased
- Configuration words are erased

Address 2000h-2008h:

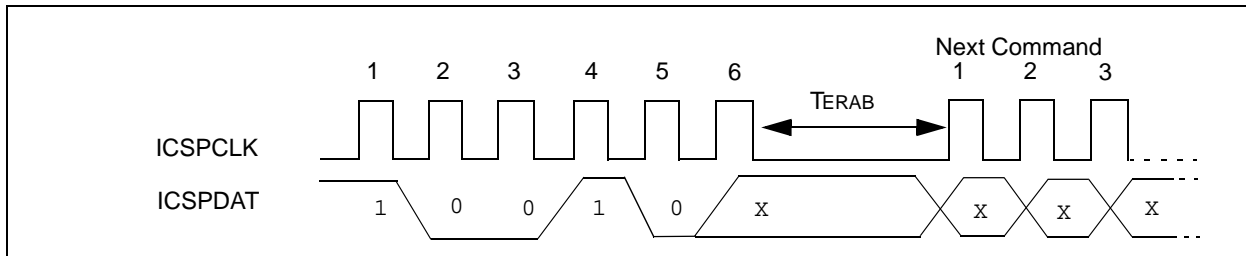
- Program Memory is erased
- Configuration Words are erased
- User ID Locations are erased

A Bulk Erase Program Memory command should not be issued when the address is greater than 2008h.

After receiving the Bulk Erase Program Memory command, the erase will not complete until the time interval, TERAB, has expired.

**Note:** The code protection Configuration bit ( $\overline{CP}$ ) has no effect on the Bulk Erase Program Memory command.

**FIGURE 4-9: BULK ERASE PROGRAM MEMORY**

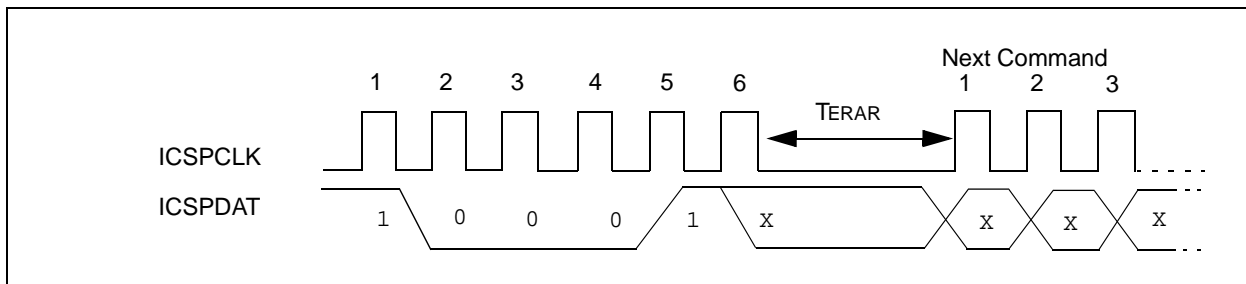


## 4.2.10 ROW ERASE PROGRAM MEMORY

This command erases the 32-word row of program memory pointed to by  $PC<13:5>$ . If the program memory array is protected ( $\overline{CP} = 0$ ) or the PC points to the configuration memory ( $> 0x2000$ ), the command is ignored. When the address is 2000h-2008h, the Row Erase Program Memory command will only erase the user ID locations, regardless of the configuration bit CP setting.

After receiving the Row Erase Program Memory command, the erase will not be complete until the time interval, TERAR, has expired.

**FIGURE 4-10: ROW ERASE PROGRAM MEMORY**



## 5.0 PROGRAMMING ALGORITHMS

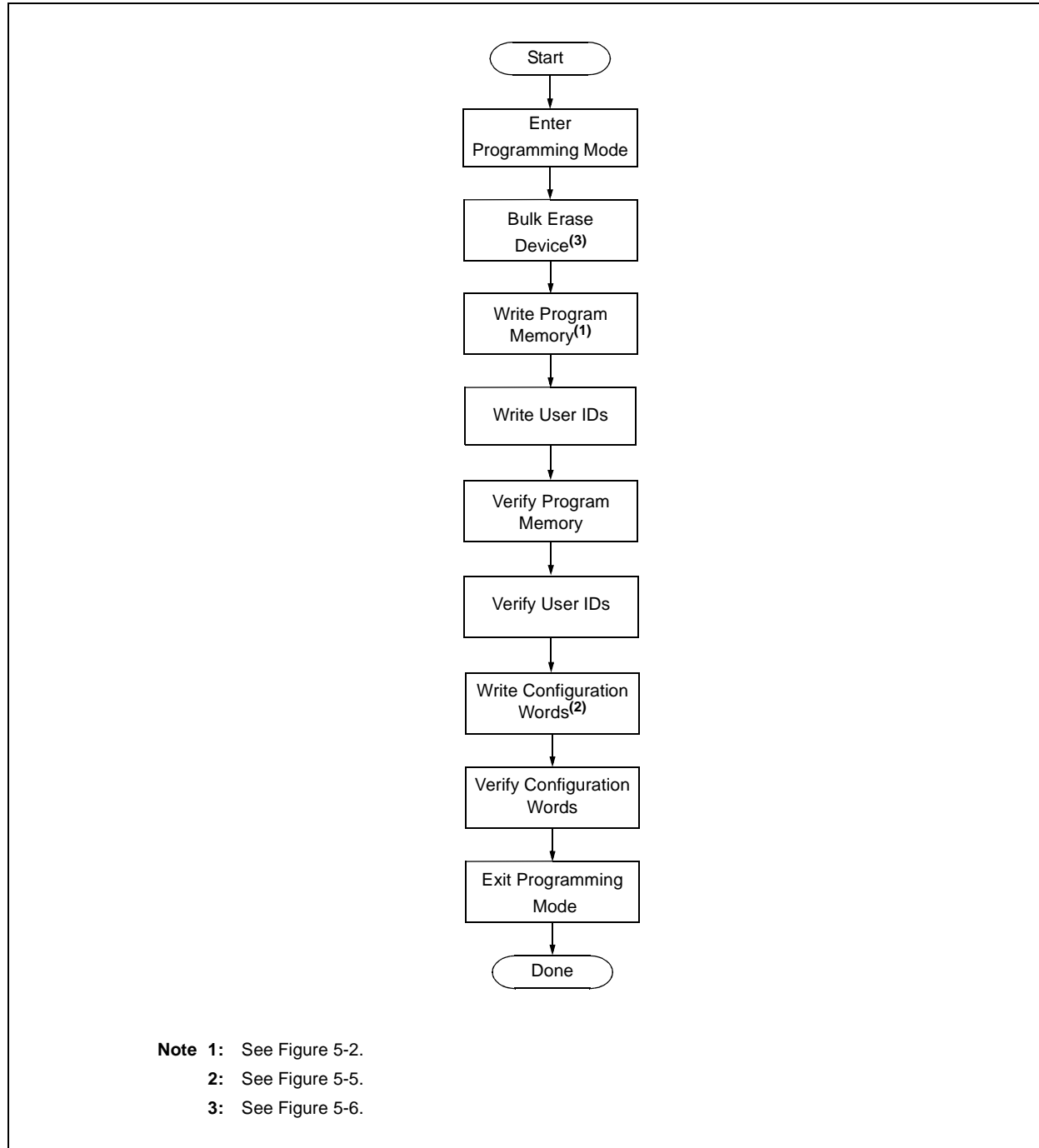
The PIC16F/LF720 and PIC16F/LF721 devices have the capability of storing 32 14-bit words in its data latches. The data latches are internal to the PIC16F/LF720 and PIC16F/LF721 devices and are only used for programming. The data latches allow the user to program up to 32 program words with a single Begin Externally Timed Programming or Begin Internally Timed Programming command. The Load Program Data or the Load Configuration command is used to load a single data latch. The data latch will hold the data until the Begin Externally Timed Programming or Begin Internally Timed Programming command is given.

The data latches are aligned with the 5 LSb of the address. The address at the time the Begin Externally Timed Programming or Begin Internally Timed Programming command is given will determine which location(s) in memory are written. Writes cannot cross a physical 32-word boundary. For example, attempting to write from address 0002h-0021h will result in data being written to 0020h-003Fh.

If more than 32 data latches are written without a Begin Externally Timed Programming or Begin Internally Timed Programming command, the data in the data latches will be overwritten. The following figures show the recommended flowcharts for programming.

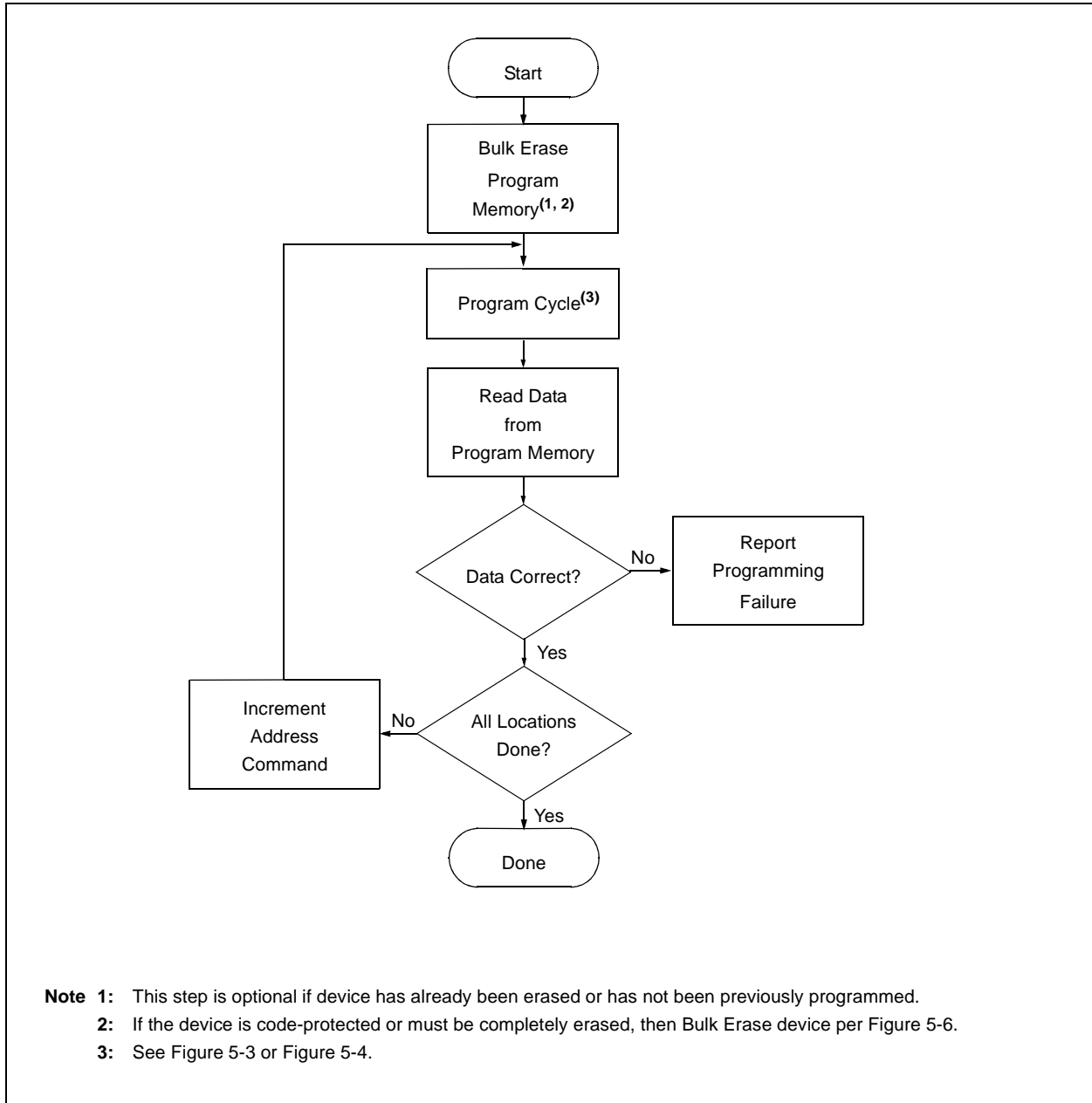
# PIC16F/LF720/721

FIGURE 5-1: DEVICE PROGRAM/VERIFY FLOWCHART



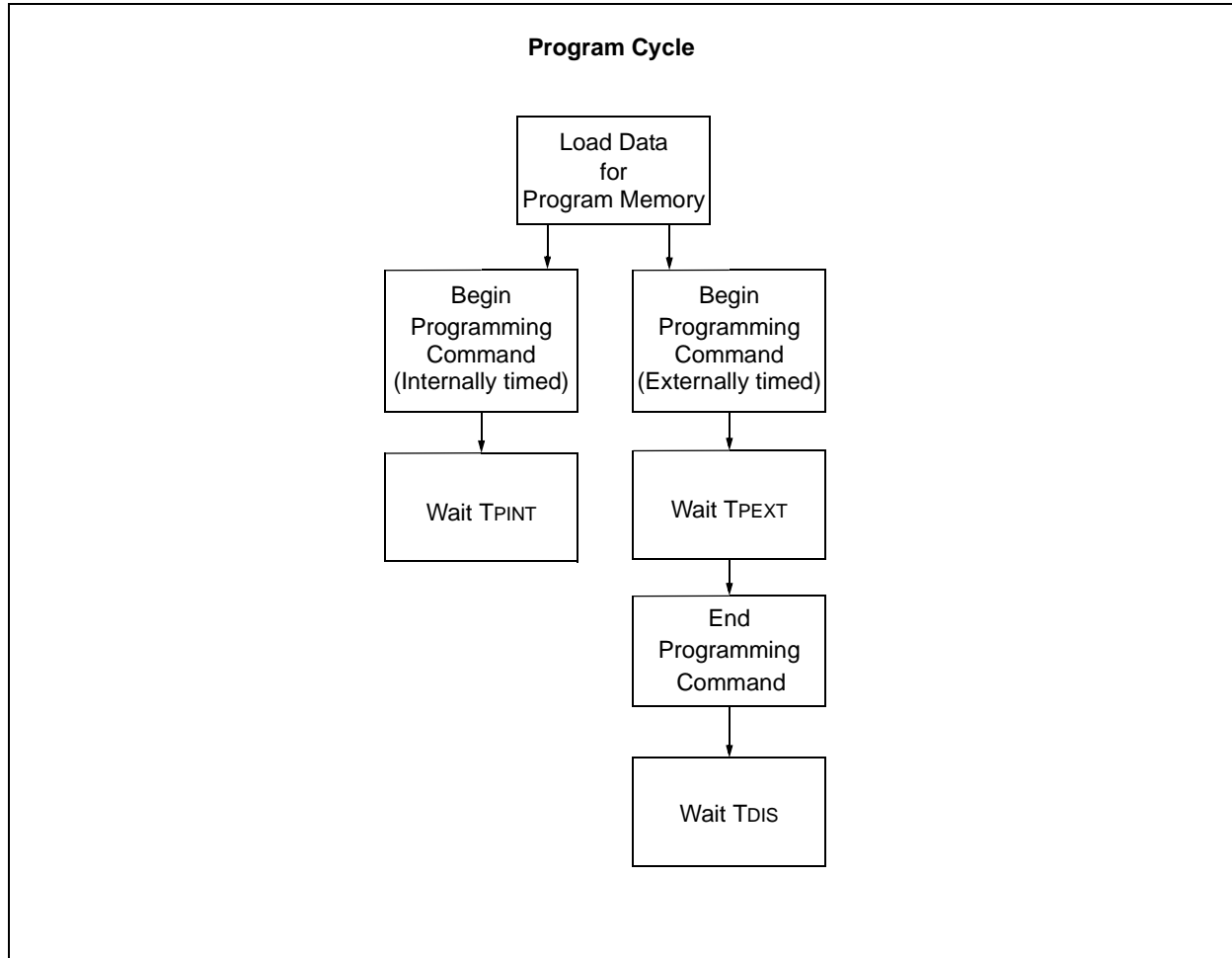


**FIGURE 5-2: PROGRAM MEMORY FLOWCHART**

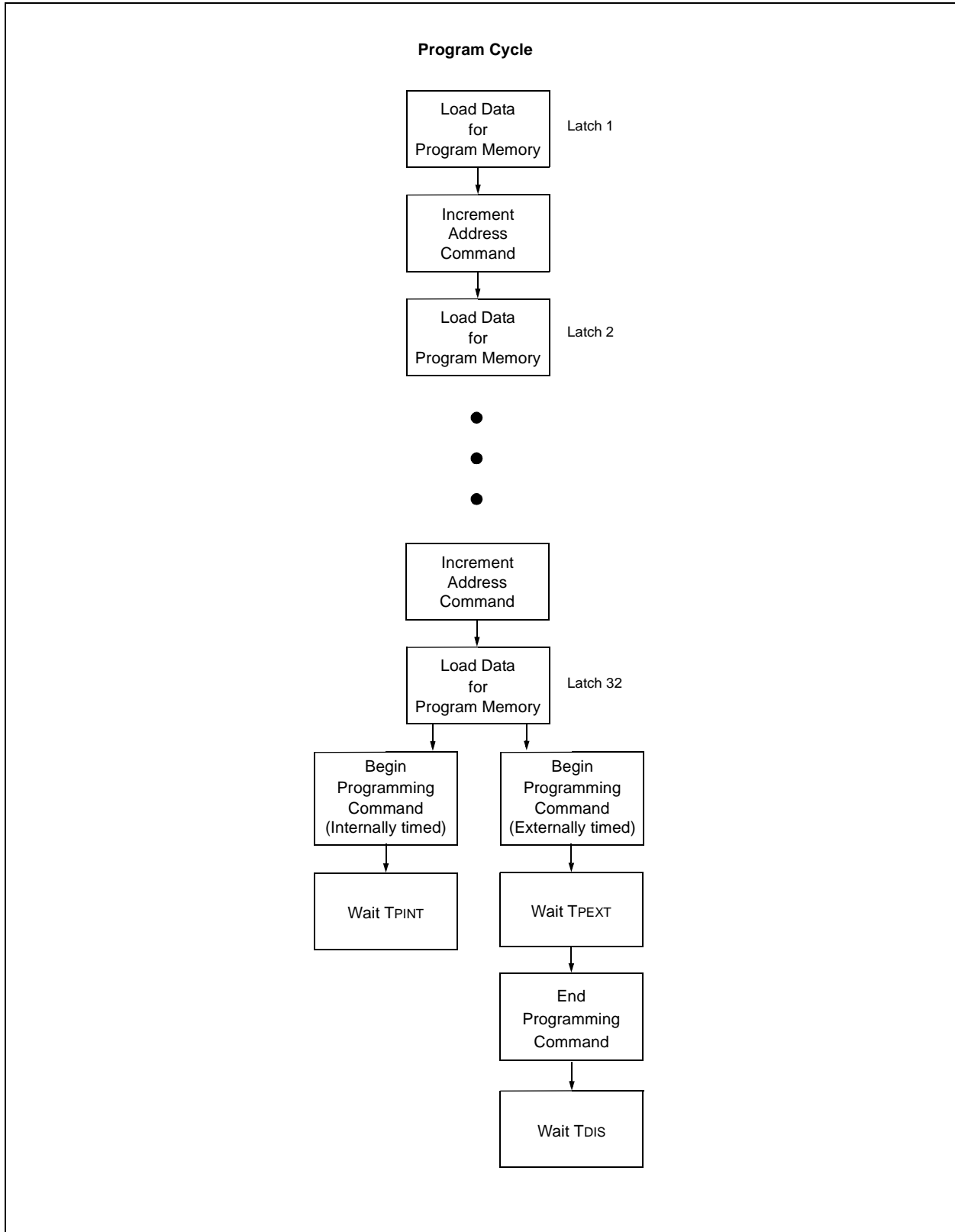


# PIC16F/LF720/721

FIGURE 5-3: ONE-WORD PROGRAM CYCLE

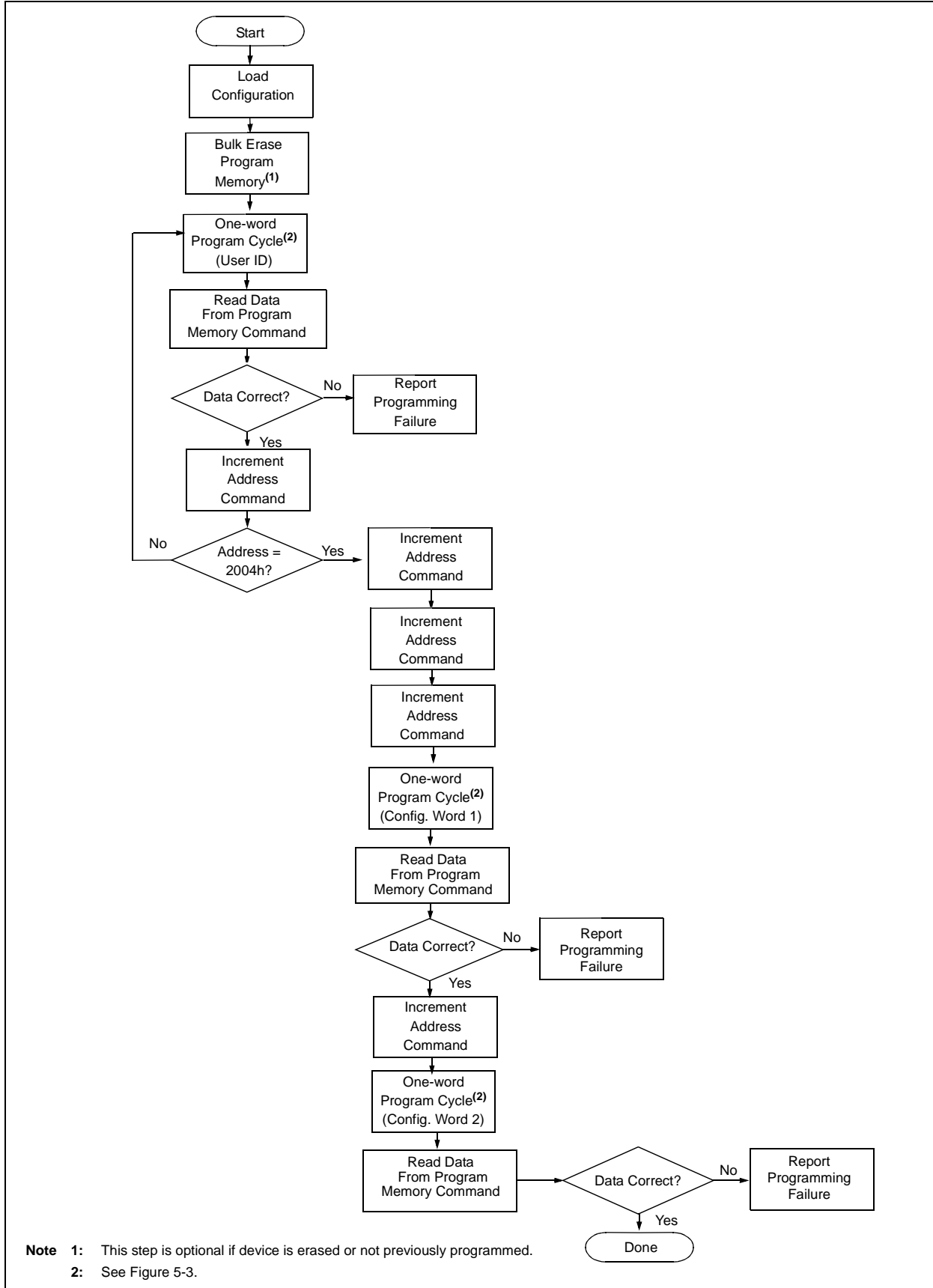


**FIGURE 5-4: MULTIPLE-WORD PROGRAM CYCLE**

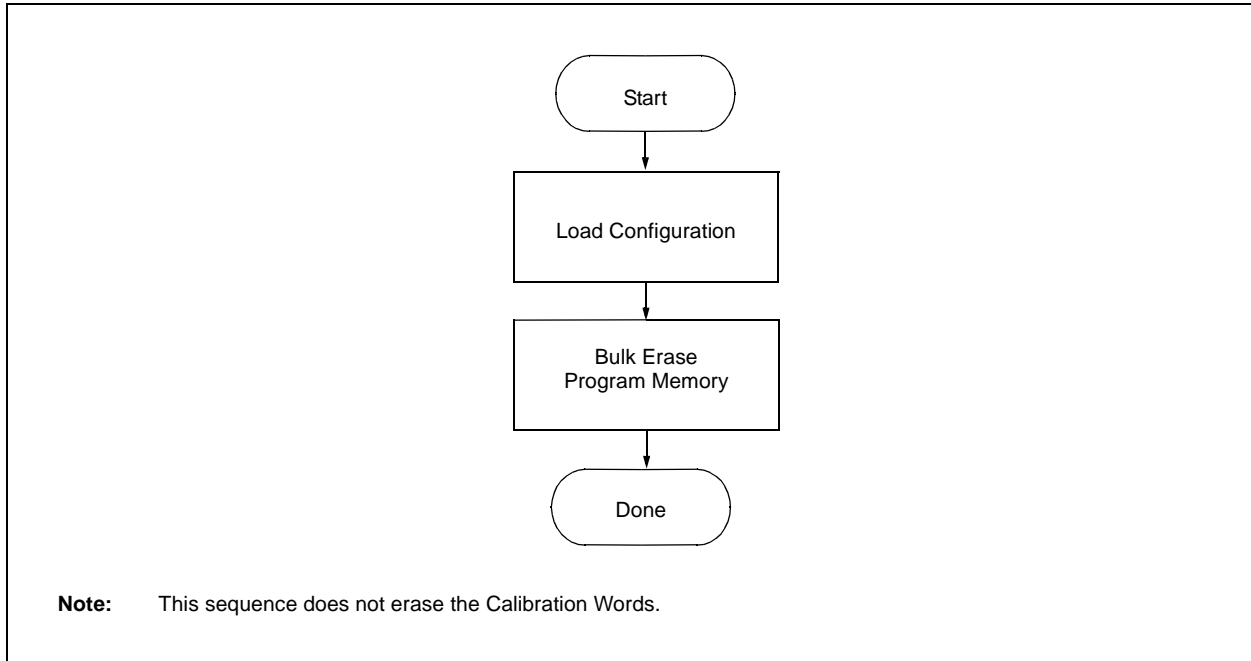


# PIC16F/LF720/721

**FIGURE 5-5: CONFIGURATION MEMORY PROGRAM FLOWCHART**



**FIGURE 5-6: ERASE FLOWCHART**



# PIC16F/LF720/721

---

## 6.0 CODE PROTECTION

Code protection is controlled using the  $\overline{CP}$  bit in Configuration Word 1. When code protection is enabled, all program memory locations 0000h-0FFFh for the PIC16F/LF720 and 0000h-07FFh for the PIC16F/LF721 will read as '0' and further programming of the program memory is disabled. Program memory can still be read by user code during program execution.

The user ID locations and Configuration Words can be programmed and read out regardless of the code protection settings.

### 6.1 Enabling Code Protection

Code protection is enabled by programming the  $\overline{CP}$  bit in Configuration Word 1 to '0'.

### 6.2 Disabling Code Protection

The only way to disable code protection is to use the Bulk Erase Program Memory command.

## 7.0 HEX FILE USAGE

In the hex file there are two bytes per program word stored in the Intel® INH8M hex format. Data is stored LSB first, MSB second. Because there are two bytes per word, the addresses in the hex file are 2x the address in program memory. (Example: The Configuration Word 1 is stored at 2007h on the PIC16F/LF720 and PIC16F/LF721. In the hex file this will be at location 400Eh-400Fh).

### 7.1 Configuration Word

To allow portability of code, it is strongly recommended that the programmer is able to read the Configuration Words and user ID locations from the hex file. If the Configuration Words information was not present in the hex file, a simple warning message may be issued. Similarly, while saving a hex file, Configuration Words and user ID information should be included.

### 7.2 Device ID and Revision

If a device ID is present in the hex file at 400Ch-400Dh (2006h on the part), the programmer should verify the device ID (excluding the revision) against the value read from the part. On a mismatch condition, the programmer should generate a warning message.

## 7.3 Checksum Computation

The checksum is calculated by two different methods, dependent on the setting of the  $\overline{CP}$  Configuration bit.

**TABLE 7-1: CONFIGURATION WORD MASK VALUES**

| Device     | Config. Word 1 Mask | Config. Word 2 Mask |
|------------|---------------------|---------------------|
| PIC16F720  | 337Bh               | 0013h               |
| PIC16LF720 | 337Bh               | 0003h               |
| PIC16F721  | 337Bh               | 0013h               |
| PIC16LF721 | 337Bh               | 0003h               |

### 7.3.1 CODE PROTECTION DISABLED

With the code protection disabled, the checksum is computed by reading the contents of the PIC16F/LF720 and PIC16F/LF721 program memory locations and adding up the program memory data, starting at address 0000h, up to the maximum user addressable location, 0FFFh for the PIC16F/LF720 and 07FFh for the PIC16F/LF721. Any Carry bit exceeding 16 bits are neglected. Additionally, the relevant bits of the Configuration Words are added to the checksum. All unused Configuration bits are masked to '0'. See Table 7-1 for Configuration Word Mask Values.

Example 7-1 through Example 7-4 shown below are for a blank device and for a device with 00AAh at the first and last program memory locations.

**EXAMPLE 7-1: CHECKSUM COMPUTED WITH CODE PROTECTION DISABLED (PIC16F720), BLANK DEVICE**

|                  |   |       |
|------------------|---|-------|
| <b>PIC16F720</b> | Sum of Memory addresses 0000h-07FFh <sup>(1)</sup>  | F800h |
|                  | Configuration Word 1 <sup>(2)</sup>   | 3FFFh |
|                  | Configuration Word 1 mask <sup>(3)</sup>  | 337Bh |
|                  | Configuration Word 2 <sup>(2)</sup>   | 3FFFh |
|                  | Configuration Word 2 mask <sup>(3)</sup>  | 0013h |
|                  | Checksum = F800h + (3FFFh and 337Bh) + (3FFFh and 0013h)  |       |
|                  | = F800h + 337Bh + 0013h   |       |
|                  | = 2B8Eh   |       |
| <br>             |   |       |
| <b>Note 1:</b>   | Sum of Memory addresses = (Total number of program memory address locations) x (3FFFh) = F800h, truncated to 16 bits. |       |
| <b>2:</b>        | Configuration Word 1 and 2 = all bits are '1'; thus, code-protect is disabled.  |       |
| <b>3:</b>        | Configuration Word 1 and 2 Mask = all bits are set to '1', except for unimplemented bits that are '0'.                |       |

# PIC16F/LF720/721

## EXAMPLE 7-2: CHECKSUM COMPUTED WITH CODE PROTECTION DISABLED (PIC16LF720), 00AAh AT FIRST AND LAST ADDRESS

|                   |  |   |
|-------------------|--|---|
| <b>PIC16LF720</b> | Sum of Memory addresses 0000h-07FFh <sup>(1)</sup> | 7956h   |
|                   | Configuration Word 1 <sup>(2)</sup>                | 3FFFh   |
|                   | Configuration Word 1 mask <sup>(3)</sup>           | 337Bh   |
|                   | Configuration Word 2 <sup>(2)</sup>                | 3FFFh   |
|                   | Configuration Word 2 mask <sup>(4)</sup>           | 0003h   |
|                   | Checksum   | = 7956h + (3FFFh and 337Bh) + (3FFFh and 0003h) |
|                   |  | = 7956h + 337Bh + 0003h                         |
|                   |  | = ACD4h   |

**Note 1:** Total number of program memory address locations:  $07FFh + 1 = 0800h$ . Then,  $0800h - 2 = 07FEh$ . Thus,  $[(07FEh \times 3FFFh) + (2 \times 00AAh)] = 7956h$ , truncated to 16 bits.

**2:** Configuration Word 1 and 2 = all bits are '1'; thus, code-protect is disabled.

**3:** Configuration Word 1 Mask = all Configuration Word bits are set to '1', except for unimplemented bits that are '0'.

**4:** On the PIC16LF720 device, the VCAPEN bit is not implemented in Configuration Word 2; thus, all unimplemented bits are '0'.

## EXAMPLE 7-3: CHECKSUM COMPUTED WITH CODE PROTECTION DISABLED (PIC16F721), BLANK DEVICE

|                  |  |   |
|------------------|--|---|
| <b>PIC16F721</b> | Sum of Memory addresses 0000h-0FFFh <sup>(1)</sup> | F000h   |
|                  | Configuration Word 1 <sup>(2)</sup>                | 3FFFh   |
|                  | Configuration Word 1 mask <sup>(3)</sup>           | 337Bh   |
|                  | Configuration Word 2 <sup>(2)</sup>                | 3FFFh   |
|                  | Configuration Word 2 mask <sup>(3)</sup>           | 0013h   |
|                  | Checksum   | = F000h + (3FFFh and 337Bh) + (3FFFh and 0013h) |
|                  |  | = F000h + 337Bh + 0013h                         |
|                  |  | = 56F6h   |

**Note 1:** Sum of Memory addresses = (Total number of program memory address locations)  $\times$  (3FFFh) = F800h, truncated to 16 bits.

**2:** Configuration Word 1 and 2 = all bits are '1'; thus, code-protect is disabled.

**3:** Configuration Word 1 and 2 Mask = all bits are set to '1', except for unimplemented bits that are '0'.



**EXAMPLE 7-4: CHECKSUM COMPUTED WITH CODE PROTECTION DISABLED (PIC16LF721), 00AAh AT FIRST AND LAST ADDRESS**

|                   |  |  |
|-------------------|--|--|
| <b>PIC16LF721</b> | Sum of Memory addresses 0000h-0FFFh <sup>(1)</sup> | 7156h  |
|                   | Configuration Word 1 <sup>(2)</sup>                | 3FFFh  |
|                   | Configuration Word 1 mask <sup>(3)</sup>           | 337Bh  |
|                   | Configuration Word 2 <sup>(2)</sup>                | 3FFFh  |
|                   | Configuration Word 2 mask <sup>(4)</sup>           | 0003h  |
|                   | Checksum   | =7156h + (3FFFh and 337Bh) + (3FFFh and 0003h) |
|                   |  | = 7156h + 337Bh + 0003h                        |
|                   |  | = A4D4h  |

**Note 1:** Total number of program memory address locations:  $0FFFh + 1 = 1000h$ . Then,  $1000h - 2 = 0FFEh$ . Thus,  $[(0FFEh \times 3FFFh) + (2 \times 00AAh)] = 7156h$ , truncated to 16 bits.

**2:** Configuration Word 1 and 2 = all bits are '1'; thus, code-protect is disabled.

**3:** Configuration Word 1 Mask = all Configuration Word bits are set to '1', except for unimplemented bits that are '0'.

**4:** On the PIC16LF721 device, the VCAPEN bit is not implemented in Configuration Word 2; thus, all unimplemented bits are '0'.

# PIC16F/LF720/721

## 7.3.2 CODE PROTECTION ENABLED

With the program code protection enabled, the checksum is computed in the following manner: the Least Significant nibble of each user ID is used to create a 16-bit value. The masked value of user ID location 2000h is the Most Significant nibble. This Sum of user IDs is summed with the Configuration Words (all unimplemented Configuration bits are masked to '0').

Example 7-5 through Example 7-8 shown below are for a blank device and for a device with 00AAh at the first and last program memory locations. Also, see Table 7-1 for Configuration Word mask values with code protection enabled.

### EXAMPLE 7-5: CHECKSUM COMPUTED WITH CODE PROTECTION ENABLED (PIC16F720), BLANK DEVICE

|                  |  |  |
|------------------|--|--|
| <b>PIC16F720</b> | Configuration Word 1 <sup>(2)</sup>      | 3FBFh  |
|                  | Configuration Word 1 mask <sup>(3)</sup> | 337Bh  |
|                  | Configuration Word 2 <sup>(2)</sup>      | 3FFFh  |
|                  | Configuration Word 2 mask <sup>(3)</sup> | 0013h  |
|                  | User ID (2000h) <sup>(1)</sup>           | 0001h  |
|                  | User ID (2001h) <sup>(1)</sup>           | 0007h  |
|                  | User ID (2002h) <sup>(1)</sup>           | 000Ah  |
|                  | User ID (2003h) <sup>(1)</sup>           | 000Fh  |
|                  | Sum of User IDs <sup>(4)</sup>           | $= (0001h \text{ and } 000Fh) \ll 12 + (0007h \text{ and } 000Fh) \ll 8 +$ $(000Ah \text{ and } 000Fh) \ll 4 + (000Fh \text{ and } 000Fh)$ $= 1000h + 0700h + 00A0h + 000Fh$ $= 17AFh$ |
|                  | Checksum                                 | $= (3FBFh \text{ and } 337Bh) + (3FFFh \text{ and } 0013h) + \text{Sum of User IDs}$ $= 333Bh + 0013h + 17AFh$ $= 4AFDh$   |

**Note 1:** User ID values in this example are random values.

**2:** Configuration Word 1 and 2 = all bits are '1', except the code-protect enable bit.

**3:** Configuration Word 1 and 2 Mask = all Configuration Word bits are set to '1', except for unimplemented bits which read '0'.

**4:**  $\ll$  = shift left, thus the LSb of the first user ID value is the MSb of the sum of user IDs and so on until the LSb of the last user ID value becomes the LSb of the sum of user IDs.

## EXAMPLE 7-6: CHECKSUM COMPUTED WITH CODE PROTECTION ENABLED (PIC16F721), BLANK DEVICE

|                  |  |   |
|------------------|--|---|
| <b>PIC16F721</b> | Configuration Word 1 <sup>(2)</sup>      | 3FBFh   |
|                  | Configuration Word 1 mask <sup>(3)</sup> | 337Bh   |
|                  | Configuration Word 2 <sup>(2)</sup>      | 3FFFh   |
|                  | Configuration Word 2 mask <sup>(3)</sup> | 0013h   |
|                  | User ID (2000h) <sup>(1)</sup>           | 0001h   |
|                  | User ID (2001h) <sup>(1)</sup>           | 0007h   |
|                  | User ID (2002h) <sup>(1)</sup>           | 000Ah   |
|                  | User ID (2003h) <sup>(1)</sup>           | 000Fh   |
|                  | Sum of User IDs <sup>(4)</sup>           | $= (0001h \text{ and } 000Fh) \ll 12 + (0007h \text{ and } 000Fh) \ll 8 +$<br>$(000Ah \text{ and } 000Fh) \ll 4 + (000Fh \text{ and } 000Fh)$<br>$= 1000h + 0700h + 00A0h + 000Fh$<br>$= 17AFh$ |
|                  | Checksum                                 | $= (3FBFh \text{ and } 337Bh) + (3FFFh \text{ and } 0013h) + \text{Sum of User IDs}$<br>$= 333Bh + 0013h + 17AFh$<br>$= 4AFDh$  |

**Note 1:** User ID values in this example are random values.

**2:** Configuration Word 1 and 2 = all bits are '1', except the code-protect enable bit.

**3:** Configuration Word 1 and 2 Mask = all Configuration Word bits are set to '1', except for unimplemented bits which read '0'.

**4:**  $\ll$  = shift left, thus the LSb of the first user ID value is the MSb of the sum of user IDs and so on until the LSb of the last user ID value becomes the LSb of the sum of user IDs.

# PIC16F/LF720/721

## EXAMPLE 7-7: CHECKSUM COMPUTED WITH CODE PROTECTION ENABLED (PIC16LF720), 00AAh AT FIRST AND LAST ADDRESS

|                   |   |  |
|-------------------|---|--|
| <b>PIC16LF720</b> | Configuration Word 1 <sup>(2)</sup>           | 3FBFh  |
|                   | Configuration Word 1 mask <sup>(3)</sup>      | 337Bh  |
|                   | Configuration Word 2 <sup>(2)</sup>           | 3FFFh  |
|                   | Configuration Word 2 mask <sup>(3), (5)</sup> | 0003h  |
|                   | User ID (2000h) <sup>(1)</sup>                | 0009h  |
|                   | User ID (2001h) <sup>(1)</sup>                | 0008h  |
|                   | User ID (2002h) <sup>(1)</sup>                | 000Dh  |
|                   | User ID (2003h) <sup>(1)</sup>                | 0005h  |
|                   | Sum of User IDs <sup>(4)</sup>                | = (0009h and 000Fh) << 12 + (0008h and 000Fh) << 8 +<br>(000Dh and 000Fh) << 4 + (0005h and 000Fh)<br>= 9000h + 0800h + 00D0h + 0005h<br>= 98D5h |
|                   | Checksum                                      | = (3FBFh and 337Bh) + (3FFFh and 0003h) + Sum of User IDs<br>= 333Bh + 0003h + 98D5h<br>= CC13h  |

**Note 1:** User ID values in this example are random values.

**2:** Configuration Word 1 and 2 = all bits are '1', except the code-protect enable bit.

**3:** Configuration Word 1 and 2 Mask = all Configuration Word bits are set to '1', except for unimplemented bits which read '0'.

**4:** << = shift left, thus the LSb of the first user ID value is the MSb of the sum of user IDs and so on until the LSb of the last user ID value becomes the LSb of the sum of user IDs.

**5:** On the PIC16LF720 device, the VCAPEN bit is not implemented in Configuration Word 2; thus, all unimplemented bits are '0'.

## EXAMPLE 7-8: CHECKSUM COMPUTED WITH CODE PROTECTION ENABLED (PIC16LF721), 00AAh AT FIRST AND LAST ADDRESS

|                   |   |  |
|-------------------|---|--|
| <b>PIC16LF721</b> | Configuration Word 1 <sup>(2)</sup>           | 3FBFh  |
|                   | Configuration Word 1 mask <sup>(3)</sup>      | 337Bh  |
|                   | Configuration Word 2 <sup>(2)</sup>           | 3FFFh  |
|                   | Configuration Word 2 mask <sup>(3), (5)</sup> | 0003h  |
|                   | User ID (2000h) <sup>(1)</sup>                | 0009h  |
|                   | User ID (2001h) <sup>(1)</sup>                | 0008h  |
|                   | User ID (2002h) <sup>(1)</sup>                | 000Dh  |
|                   | User ID (2003h) <sup>(1)</sup>                | 0005h  |
|                   | Sum of User IDs <sup>(4)</sup>                | = (0009h and 000Fh) << 12 + (0008h and 000Fh) << 8 +<br>(000Dh and 000Fh) << 4 + (0005h and 000Fh)<br>= 9000h + 0800h + 00D0h + 0005h<br>= 98D5h |
|                   | Checksum                                      | = (3FBFh and 337Bh) + (3FFFh and 0003h) + Sum of User IDs<br>= 333Bh + 0003h + 98D5h<br>= CC13h  |

**Note 1:** User ID values in this example are random values.

**2:** Configuration Word 1 and 2 = all bits are '1', except the code-protect enable bit.

**3:** Configuration Word 1 and 2 Mask = all Configuration Word bits are set to '1', except for unimplemented bits which read '0'.

**4:** << = shift left, thus the LSb of the first user ID value is the MSb of the sum of user IDs and so on until the LSb of the last user ID value becomes the LSb of the sum of user IDs.

**5:** On the PIC16LF721 device, the VCAPEN bit is not implemented in Configuration Word 2; thus, all unimplemented bits are '0'.

# PIC16F/LF720/721

## 8.0 ELECTRICAL SPECIFICATIONS

Refer to device specific data sheet for absolute maximum ratings.

**TABLE 8-1: AC/DC CHARACTERISTICS TIMING REQUIREMENTS FOR PROGRAM/VERIFY MODE**

| AC/DC CHARACTERISTICS                  |   | Standard Operating Conditions (unless otherwise stated)<br>Operating Temperature +10°C ≤ TA ≤ +40°C |       |                               |       |  |  |
|--|---|---|-------|-------------------------------|-------|--|--|
| Sym.                                   | Characteristics   | Min.  | Type. | Max.                          | Units | Conditions/<br>Comments  |  |
| <b>Supply Voltages and currents</b>    |   |   |       |                               |       |  |  |
| VDD                                    | Read/Write and Row Erase operations   | PIC16F720<br>PIC16F721  | 2.1   | —                             | 5.5   | V  |  |
|  |   | PIC16LF720<br>PIC16LF721  | 2.1   | —                             | 3.6   | V  |  |
|  | Bulk Erase operations   | PIC16F720<br>PIC16F721  | 2.7   | —                             | 5.5   | V  |  |
|  |   | PIC16LF720<br>PIC16LF721  | 2.7   | —                             | 3.6   | V  |  |
| IDDI                                   | Current on VDD, Idle  | —   | —     | 1.0                           | mA    |  |  |
| IDDA                                   | Current on VDD, program cycle or Bulk Erase in progress   | —   | —     | 5.0                           | mA    |  |  |
| <b>VPP</b>                             |   |   |       |                               |       |  |  |
| VIHH                                   | High voltage on $\overline{\text{MCLR}}$ /VPP for Program/Verify mode entry                                   | 8.0   | —     | 9.0                           | V     |  |  |
| TVHHR                                  | $\overline{\text{MCLR}}$ rise time (VDD to VIHH) for Program/Verify mode entry                                | —   | —     | 1.0                           | μs    |  |  |
| IPP                                    | Current on $\overline{\text{MCLR}}$ /VPP  |   |       | 600                           | μA    |  |  |
| <b>I/O pins</b>                        |   |   |       |                               |       |  |  |
| VIH                                    | (ICSPCLK, ICSPDAT) input high level   | 0.8 VDD   | —     | —                             | V     |  |  |
| VIL                                    | (ICSPCLK, ICSPDAT) input low level  | —   | —     | 0.2 VDD                       | V     |  |  |
| VOH                                    | ICSPDAT output high level   | VDD-0.7<br>VDD-0.7<br>VDD-0.7   | —     | VDD                           | V     | IOH = 3.5 mA, VDD = 5V<br>IOH = 3 mA, VDD = 3.3V<br>IOH = 2 mA, VDD = 1.8V |  |
| VOL                                    | ICSPDAT output low level  | VSS   | —     | VSS+0.6<br>VSS+0.6<br>VSS+0.6 | V     | IOH = 8 mA, VDD = 5V<br>IOH = 6 mA, VDD = 3.3V<br>IOH = 3 mA, VDD = 1.8V   |  |
| <b>Programming mode entry and exit</b> |   |   |       |                               |       |  |  |
| TENTS                                  | Programming mode entry setup time: ICSPCLK, ICSPDAT setup time before VDD or $\overline{\text{MCLR}}\uparrow$ | 100   | —     | —                             | ns    |  |  |
| TENTH                                  | Programming mode entry hold time: ICSPCLK, ICSPDAT hold time after VDD or $\overline{\text{MCLR}}\uparrow$    | 250   | —     | —                             | μs    |  |  |
| <b>Serial Program/Verify</b>           |   |   |       |                               |       |  |  |
| TCKL                                   | Clock Low Pulse Width   | 100   | —     | —                             | ns    |  |  |
| TCKH                                   | Clock High Pulse Width  | 100   | —     | —                             | ns    |  |  |
| Tds                                    | Data in setup time before clock↓  | 100   | —     | —                             | ns    |  |  |
| TDH                                    | Data in hold time after clock↓  | 100   | —     | —                             | ns    |  |  |
| TCO                                    | Clock↑ to data out valid (during a Read Data command)   | 0   | —     | 80                            | ns    |  |  |
| TLZD                                   | Clock↓ to data low-impedance (during a Read Data command)   | 0   | —     | 80                            | ns    |  |  |
| THZD                                   | Clock↓ to data high-impedance (during a Read Data command)  | 0   | —     | 80                            | ns    |  |  |
| TDLY                                   | Data input not driven to next clock input (delay required between command/data or command/command)            | 1.0   | —     | —                             | μs    |  |  |

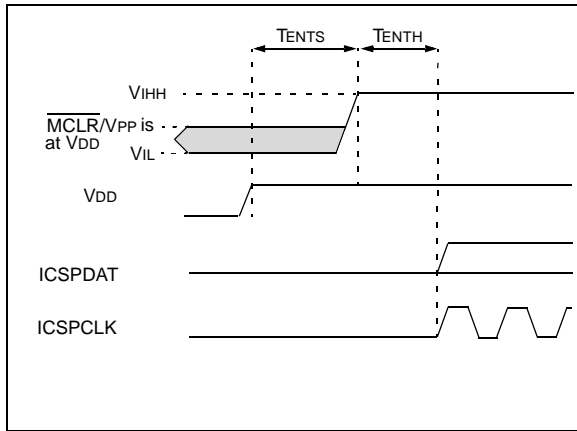
**TABLE 8-1: AC/DC CHARACTERISTICS TIMING REQUIREMENTS FOR PROGRAM/VERIFY MODE**

| AC/DC CHARACTERISTICS |  | Standard Operating Conditions (unless otherwise stated)<br>Operating Temperature +10°C ≤ TA ≤ +40°C |       |      |       |                                       |
|-----------------------|--|---|-------|------|-------|---------------------------------------|
| Sym.                  | Characteristics  | Min.  | Type. | Max. | Units | Conditions/<br>Comments               |
| TERAB                 | Bulk Erase cycle time                                  | —   | —     | 5    | ms    |                                       |
| TERAR                 | Row Erase cycle time                                   | —   | —     | 2.5  | ms    |                                       |
| TPINT                 | Internally timed programming operation time            | —   | —     | 2.5  | ms    | Program memory<br>Configuration fuses |
|                       |  | —   | —     | 5    | ms    |                                       |
| TPEXT                 | Externally timed programming pulse                     | 1.0   | —     | 2.1  | ms    | 10°C ≤ TA ≤ +40°C<br>Program memory   |
| TDIS                  | Time delay from program to compare (HV discharge time) | 100   | —     | —    | μs    |                                       |
| TEXIT                 | Time delay when exiting Program/Verify mode            | 1   | —     | —    | μs    |                                       |

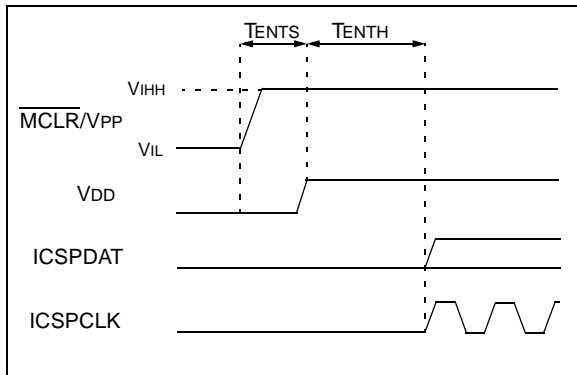
# PIC16F/LF720/721

## 8.1 AC Timing Diagrams

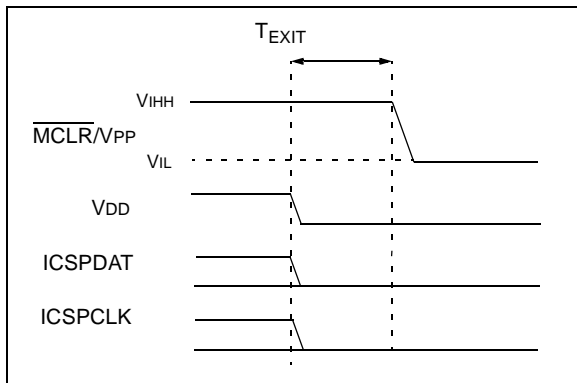
**FIGURE 8-1: PROGRAMMING MODE ENTRY – V<sub>DD</sub> FIRST**



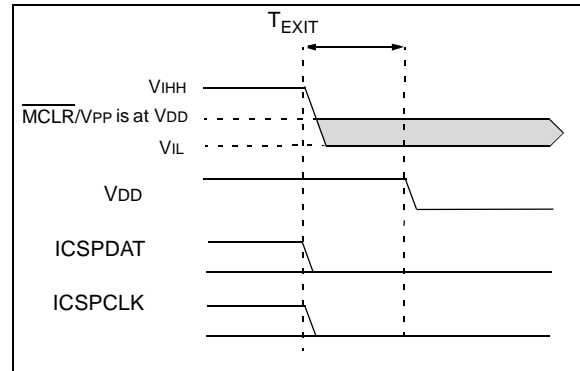
**FIGURE 8-2: PROGRAMMING MODE ENTRY – MCLR/VPP FIRST**



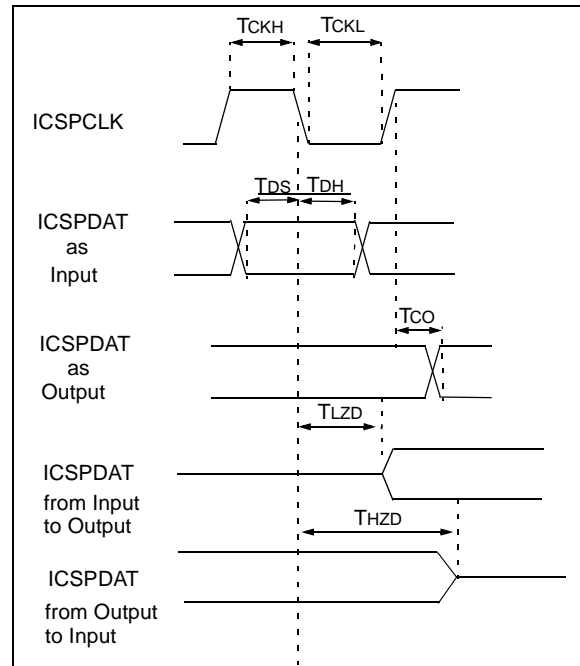
**FIGURE 8-3: PROGRAMMING MODE EXIT – MCLR/VPP LAST**



**FIGURE 8-4: PROGRAMMING MODE EXIT – V<sub>DD</sub> LAST**

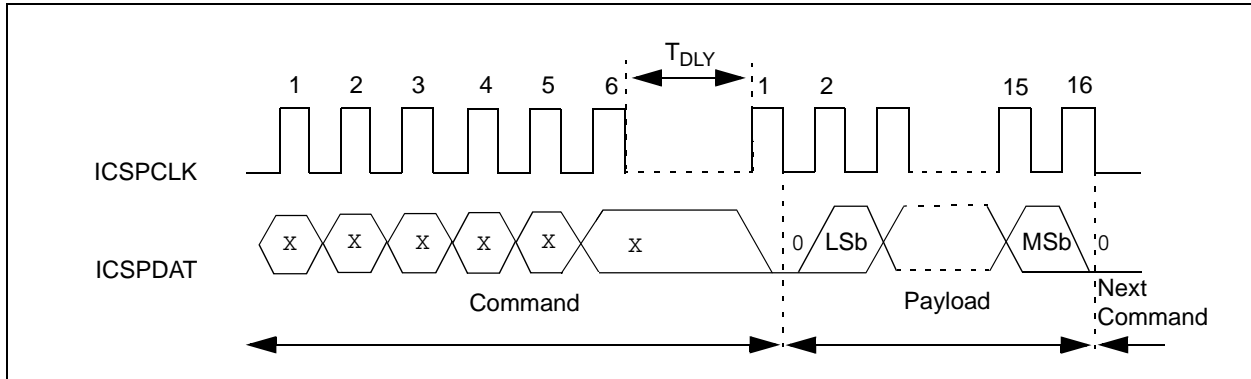


**FIGURE 8-5: CLOCK AND DATA TIMING**





**FIGURE 8-6: COMMAND-PAYLOAD TIMING**



# PIC16F/LF720/721

---

## APPENDIX A: REVISION HISTORY

### Revision A (12/2009)

Initial release of this document.

---

**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, rPIC 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, Octopus, Omniscient Code Generation, PICC, PICC-18, PICDEM, PICDEM.net, PICkit, PICtail, PIC<sup>32</sup> logo, REAL ICE, rLAB, 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.

© 2009, 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](http://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

**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: 91-80-3090-4444  
Fax: 91-80-3090-4080

**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: 886-7-536-4818  
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

03/26/09