1.0 PROGRAMMING THE PIC12F6XX/16F6XX DEVICES

The PIC12F6XX/16F6XX devices are programmed using a serial method. The Serial mode will allow the PIC12F6XX/16F6XX devices to be programmed while in the user's system. This programming specification applies to the PIC12F6XX/16F6XX devices in all packages.

1.1 Hardware Requirements

PIC12F6XX/16F6XX devices require one power supply for Vdd (5.0V) and one for Vpp (12.0V).

1.2 Program/Verify Mode

The Program/Verify mode for the PIC12F6XX/16F6XX devices allow programming of user program memory, data memory, user ID locations and the Configuration Word.

Programming and verification can take place on any memory region, independent of the remaining regions. This allows independent programming of program and data memory regions. Therefore, unprotected data memory can be reprogrammed and protected without losing the content in the program memory.

---

### TABLE 1-1: PIN DESCRIPTIONS IN PROGRAM/VERIFY MODE

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Function</th>
<th>During Programming</th>
<th>Pin Type</th>
<th>Pin Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GP1/RA1</td>
<td>ICSPCLK</td>
<td>I</td>
<td>Clock input – Schmitt Trigger input</td>
<td></td>
</tr>
<tr>
<td>GP0/RA0</td>
<td>ICSPDAT</td>
<td>I/O</td>
<td>Data input/output – Schmitt Trigger input</td>
<td></td>
</tr>
<tr>
<td>MCLR</td>
<td>Program/Verify mode</td>
<td>p(1)</td>
<td>Program Mode Select</td>
<td></td>
</tr>
<tr>
<td>Vdd</td>
<td>Vdd</td>
<td>P</td>
<td>Power Supply</td>
<td></td>
</tr>
<tr>
<td>Vss</td>
<td>Vss</td>
<td>P</td>
<td>Ground</td>
<td></td>
</tr>
</tbody>
</table>

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

**Note 1:** In the PIC12F6XX/16F6XX, the programming high voltage is internally generated. To activate the Program/Verify mode, high voltage needs to be applied to MCLR input. Since the MCLR is used for a level source, MCLR does not draw any significant current.
FIGURE 1-1: 8-PIN DIAGRAM FOR PIC12F635/683

PDIP, SOIC, DFN

Vdd 1 8 Vss
GP5/OSC1/CLKIN 2 7 GP0/ICSPDAT
GP4/OSC2/CLKOUT 3 6 GP1/ICSPCLK
GP3/MCLR/Vpp 4 5 GP2

FIGURE 1-2: 14-PIN DIAGRAM FOR PIC16F636/684/688

PDIP, SOIC, TSSOP

Vdd 1 14 Vss
RA5/OSC1/CLKIN 2 13 RA0/ICSPDAT
RA4/OSC2/CLKOUT 3 12 RA1/ICSPCLK
RA3/MCLR/VPP 4 11 RC2
RC5 5 10 RC0
RC4 6 9 RC1
RC3 7 8 RC2

FIGURE 1-3: 16-PIN DIAGRAMS FOR PIC16F636/684/688

QFN

RA5/OSC1/CLKIN 1 12 RA0/ICSPDAT
RA4/OSC2/CLKOUT 2 11 RA1/ICSPCLK
RA3/MCLR/VPP 3 10 RA2
RC5 4 9 RC0
RC4 5 8 RC1
RC3 6 7 RC2
RC2 7 6 RC3
RC1 8 5 RC4
RC0 9 4 RC5
Vdd 10 15 NC
NC 11 14 NC
Vss 13 16
FIGURE 1-4: 20-PIN DIAGRAMS

PDIP, SOIC\(^{(1)}\), TSSOP

<table>
<thead>
<tr>
<th>Pin</th>
<th>Description</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Vdd</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>RA5/OSC1/CLKIN</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>RA4/OSC2/CLKOUT</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>RA3/MCLR/VPP</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>RC5</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>RC4</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>RC3</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>VddT</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>RA0/ICSPDAT</td>
<td>9</td>
</tr>
<tr>
<td>10</td>
<td>RA1/ICSPCLK</td>
<td>10</td>
</tr>
<tr>
<td>11</td>
<td>RA2</td>
<td>11</td>
</tr>
<tr>
<td>12</td>
<td>RA3/MCLR/VPP</td>
<td>12</td>
</tr>
<tr>
<td>13</td>
<td>RC6</td>
<td>13</td>
</tr>
<tr>
<td>14</td>
<td>RC7</td>
<td>14</td>
</tr>
<tr>
<td>15</td>
<td>PIC16F639</td>
<td>15</td>
</tr>
<tr>
<td>16</td>
<td>RC2</td>
<td>16</td>
</tr>
<tr>
<td>17</td>
<td>RC1</td>
<td>17</td>
</tr>
<tr>
<td>18</td>
<td>RC0</td>
<td>18</td>
</tr>
<tr>
<td>19</td>
<td>RA0/ICSPDAT</td>
<td>19</td>
</tr>
<tr>
<td>20</td>
<td>Vss</td>
<td>20</td>
</tr>
</tbody>
</table>


FIGURE 1-5: 20-PIN DIAGRAM FOR PIC16F631/677/685/687/689/690

QFN
2.0 MEMORY DESCRIPTION

2.1 Program Memory Map

The user memory space extends from 0x0000 to 0x1FFF. In Program/Verify mode, the program memory space extends from 0x0000 to 0x3FFF, with the first half (0x0000-0x1FFF) being user program memory and the second half (0x2000-0x3FFF) being configuration memory. The PC will increment from 0x0000 to 0x1FFF and wrap to 0x000, 0x2000 to 0x3FFF and wrap around to 0x2000 (not to 0x0000). Once in configuration memory, the highest bit of the PC stays a '1', thus always pointing to the configuration memory. The only way to point to user program memory is to reset the part and re-enter Program/Verify mode as described in Section 3.0 “Program/Verify Mode”.

For the PIC12F6XX/16F6XX (not including PIC12F635/636/639) devices, the configuration memory space, 0x2000 to 0x2008 are physically implemented. However, only locations 0x2000 to 0x2003, 0x2007 and 0x2008 are available. Other locations are reserved.

For the PIC12F635/636/639 devices, the configuration memory space (0x2000-0x2009) are physically implemented. However, only locations 0x2000 to 0x2003 and locations 0x2006 to 0x2009 are available. Other locations are reserved.

2.2 User ID Locations

A user may store identification information (user ID) in four designated locations. The user ID locations are mapped in 0x0200 to 0x0203. It is recommended that the user use only the seven Least Significant bits (LSB) of each user ID location. The user ID locations read out normally, even after code protection is enabled. It is recommended that ID locations are written as 'xx xxxx xbbb bbbb' where 'bbb bbbb' is user ID information.

The 14 bits may be programmed, but only the seven LSB's are displayed by MPLAB® IDE. The xxxx’s are “don't care” bits and are not read by MPLAB® IDE.

2.3 Calibration Word

For the PIC16F631/677/685/687/689/690 (not including PIC12F635/636/639) devices, the 8 MHz Internal Oscillator (INTOSC), the Power-on Reset (POR) and the Brown-out Reset (BOR) modules are factory calibrated. These values are stored in the Calibration Word (0x2008). See the applicable device data sheet for more information.

For the PIC12F635/636/639 devices, the 8 MHz Internal Oscillator (INTOSC), the Power-on Reset and the Brown-out Reset modules are factory calibrated and stored in the Calibration Word (0x2008). The Wake-up Reset (WUR) and Low-Voltage Detect (LVD) modules are factory calibrated and stored in the Calibration Word (0x2009). See the applicable device data sheet for more information.

The Calibration Word locations are written at the time of manufacturing and are not erased when a Bulk Erase is performed. See Section 3.1.5.10 “Bulk Erase Program Memory” for more information on the various erase sequences. However, it is possible to inadvertently write to these locations. The device may not function properly or may operate outside of specifications if the Calibration Word locations do not contain the correct value. Therefore, it is recommended that the Calibration Words be read prior to any programming procedure and verified after programming is complete. See Figure 3-21 for a flowchart of the recommended verification procedure.

The device should not be used if the verification of the Calibration Word values fail after the device is programmed. The 0x3FFF value is a special case, it is a valid calibration value but, it is also the erased state of the register.

<table>
<thead>
<tr>
<th>Device</th>
<th>EEDATA</th>
<th>Program Flash</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC12F635</td>
<td>128 x 8</td>
<td>1k x 14</td>
</tr>
<tr>
<td>PIC12F683</td>
<td>256 x 8</td>
<td>2k x 14</td>
</tr>
<tr>
<td>PIC16F631</td>
<td>128 x 8</td>
<td>1k x 14</td>
</tr>
<tr>
<td>PIC16F636</td>
<td>256 x 8</td>
<td>2k x 14</td>
</tr>
<tr>
<td>PIC16F639</td>
<td>256 x 8</td>
<td>2k x 14</td>
</tr>
<tr>
<td>PIC16F677</td>
<td>256 x 8</td>
<td>2k x 14</td>
</tr>
<tr>
<td>PIC16F684</td>
<td>256 x 8</td>
<td>2k x 14</td>
</tr>
<tr>
<td>PIC16F685</td>
<td>256 x 8</td>
<td>4k x 14</td>
</tr>
<tr>
<td>PIC16F687</td>
<td>256 x 8</td>
<td>2k x 14</td>
</tr>
<tr>
<td>PIC16F688</td>
<td>256 x 8</td>
<td>4k x 14</td>
</tr>
<tr>
<td>PIC16F689</td>
<td>256 x 8</td>
<td>4k x 14</td>
</tr>
<tr>
<td>PIC16F690</td>
<td>256 x 8</td>
<td>4k x 14</td>
</tr>
</tbody>
</table>
FIGURE 2-1: PIC16F631 PROGRAM MEMORY MAPPING

- **2000**: User ID Location
- **2001**: User ID Location
- **2002**: User ID Location
- **2003**: User ID Location
- **2004**: Reserved
- **2005**: Reserved
- **2006**: Device ID
- **2007**: Configuration Word
- **2008**: Calibration Word
- **2009-207F**: Reserved
- **1FFF**: Implemented
- **03FF**: Implemented
- **1FFF**: Maps to 0-3FF
- **2000**: Implemented
- **2080**: Maps to 2000-207F
- **3FFF**: Reserved
- **1 KW**

**Program Memory**

**Configuration Memory**
FIGURE 2-2: PIC12F635 PROGRAM MEMORY MAPPING

2000
User ID Location

2001
User ID Location

2002
User ID Location

2003
User ID Location

2004
Reserved

2005
Reserved

2006
Device ID

2007
Configuration Word

2008
Calibration Word 1

2009
Calibration Word 2

200A-203F
Reserved

1FFF
Maps to 0-3FF

03FF
Implemented

2000
Implemented

2040
Maps to 2000-203F

3FFF
Reserved
FIGURE 2-3: PIC16F636/639 PROGRAM MEMORY MAPPING

- User ID Location
- User ID Location
- User ID Location
- User ID Location
- Reserved
- Reserved
- Device ID
- Configuration Word
- Calibration Word 1
- Calibration Word 2
- Reserved

- 2 KW

Program Memory

- Maps to 0-7FF

- 07FF

Configuration Memory

- Maps to 2000-203F

- 2000

- 1FFF

- 2040

- 2009

- 3FFF

- 200A-203F
FIGURE 2-4: PIC12F683/684 PROGRAM MEMORY MAPPING

- 2000: User ID Location
- 2001: User ID Location
- 2002: User ID Location
- 2003: User ID Location
- 2004: Reserved
- 2005: Reserved
- 2006: Device ID
- 2007: Configuration Word
- 2008: Calibration Word
- 2009-203F: Reserved

2000-203F maps to 2000-203F

07FF implemented

Program Memory

1FFF implemented

Maps to 0-7FF

2040 implemented

Maps to 2000-203F

3FFF implemented

Configuration Memory
FIGURE 2-5: PIC16F677/687 PROGRAM MEMORY MAPPING

- **07FF** Implemented
- **Maps to 0-7FF**
- **2000** Implemented
- **Maps to 2000-207F**
- **2FFF**
- **2 KW**

- User ID Location
- User ID Location
- User ID Location
- User ID Location
- Reserved
- Reserved
- Device ID
- Configuration Word
- Calibration Word
- Reserved

Program Memory

Configuration Memory
FIGURE 2-6: PIC16F688 PROGRAM MEMORY MAPPING

<table>
<thead>
<tr>
<th>Address Range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000-0FFF</td>
<td>Implemented</td>
</tr>
<tr>
<td>1000-1FFF</td>
<td>Implemented</td>
</tr>
<tr>
<td>2000-203F</td>
<td>Maps to 2000-203F</td>
</tr>
<tr>
<td>3000-3FFF</td>
<td>Maps to 0-FFF</td>
</tr>
</tbody>
</table>

- User ID Location
- Reserved
- Device ID
- Configuration Word
- Calibration Word
- Reserved
FIGURE 2-7: PIC16F685/689/690 PROGRAM MEMORY MAPPING

- Implemented
- Maps to 0-FFF
- Maps to 2000-207F
- Program Memory
- Configuration Memory

- User ID Location
- User ID Location
- User ID Location
- User ID Location
- Reserved
- Reserved
- Device ID
- Configuration Word
- Calibration Word
- Reserved

2000-207F
3.0 PROGRAM/VERIFY MODE

Two methods are available to enter Program/Verify mode. The "VPP-first" is entered by holding ICSPDAT and ICSPCLK low while raising MCLR pin from VIL to VIHH (high voltage), then applying VDD and data. This method can be used for any Configuration Word selection and must be used if the INTOSC and internal MCLR options are selected (FOSC<2:0> = 100 or 101 and MCLRE = 0). The VPP-first entry prevents the device from executing code prior to entering Program/Verify mode. See the timing diagram in Figure 3-1.

The second entry method, "VDD-first", is entered by applying VDD, holding ICSPDAT and ICSPCLK low, then raising MCLR pin from VIL to VIHH (high voltage), followed by data. This method can be used for any Configuration Word selection except when INTOSC and internal MCLR options are selected (FOSC<2:0> = 100 or 101 and MCLRE = 0). This technique 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 3-2.

Once in this mode, the program memory, data memory and configuration memory can be accessed and programmed in serial fashion. ICSPDAT and ICSPCLK are Schmitt Trigger inputs in this mode. RA4 is tri-state regardless of fuse setting.

The sequence that enters the device into the Program/Verify mode places all other logic into the Reset state (the MCLR pin was initially at VIL). Therefore, all I/O’s are in the Reset state (high-impedance inputs) and the Program Counter (PC) is cleared.

To prevent a device configured with INTOSC and internal MCLR from executing after exiting Program/Verify mode, VDD needs to power-down before VPP. See Figure 3-3 for the timing.

FIGURE 3-1: VPP-FIRST PROGRAM/VERIFY MODE ENTRY

FIGURE 3-2: VDD-FIRST PROGRAM/VERIFY MODE ENTRY

FIGURE 3-3: PROGRAM/VERIFY MODE EXIT

3.1 Program/Erase Algorithms

The PIC12F6XX/16F6XX program memory may be written in two ways. The fastest method writes four words at a time. However, one-word writes are also supported for backward compatibility with previous 8-pin and 14-pin Flash devices. The four-word algorithm is used to program the program memory only. The one-word algorithm can write any available memory location (i.e., program memory, configuration memory and data memory).

After writing the array, the PC may be reset and read back to verify the write. It is not possible to verify immediately following the write because the PC can only increment, not decrement.

A device Reset will clear the PC and set the address to ‘0’. The Increment Address command will increment the PC. The Load Configuration command will set the PC to 0x2000. The available commands are shown in Table 3-1.
## 3.1.1 FOUR-WORD PROGRAMMING

Only the program memory can be written using this algorithm. Data and configuration memory (>0x2000) must use the one-word programming algorithm (Section 3.1.2 “One-Word Programming”).

This algorithm writes four sequential addresses in program memory. The four addresses must point to a four-word block with addresses modulo 4 of 0, 1, 2 and 3. For example, programming address 4 through 7 can be programmed together. Programming addresses 2 through 5 will create an unexpected result.

The sequence for programming four words of program memory at a time is as follows:

1. Load a word at the current program memory address using Load Data for Program Memory command.
2. Issue a Increment Address command.
3. Load a word at the current program memory address using Load Data for Program Memory command.
4. Repeat Step 2 and Step 3 two times.
5. Issue a Begin Programming command either internally or externally timed.
6. Wait TPROG1 (internally timed) or TPROG2 (externally timed).
7. Issue a End Programming command if externally timed.
8. Issue a Increment Address command.
9. Repeat this sequence as required to write program memory.

See Figure 3-17 for more information.

## 3.1.2 ONE-WORD PROGRAMMING

The program memory may also be written one word at a time to allow compatibility with other 8-pin and 14-pin Flash PIC® devices. Configuration memory (>0x2000) and data memory must be written one word (or byte) at a time.

**Note:** The four write latches must be reset after programming the user ID (0x2000-0x2003) or Configuration Word (0x2007). See Section 3.1.3 “Resetting Write Latches”.

The sequence for programming one word of program memory at a time is as follows:

1. Load a word at the current program memory address using Load Data for Program Memory command.
2. Issue a Begin Programming command either internally or externally timed.
3. Wait TPROG1 (internally timed) or TPROG2 (externally timed).
4. Issue a End Programming command if externally timed.
5. Issue a Increment Address command.
6. Repeat this sequence as required to write program, data or configuration memory.

See Figure 3-16 for more information.

## 3.1.3 RESETTING WRITE LATCHES

The user ID (0x2000-0x2003) and Configuration Word (0x2007) are mapped into the configuration memory, but do not physically reside in it. As a result, the write latches are not reset when programming these locations and must be reset by the programmer. This can be done in two ways, either loading all four latches with ‘1’s or by exiting Program/Verify mode.

The sequence for manually resetting the write latches is as follows:

1. Load a word using Load Data for Program Memory or Load Data for Configuration Memory command with a data word of all ‘1’s.
2. Issue a Increment Address command.
3. Repeat this sequence three times to reset all four write latches.
3.1.4 ERASE ALGORITHMS
The PIC12F6XX/16F6XX will erase different memory locations depending on the Program Counter (PC), CP and CPD values and which erase command executed. The following sequences can be used to erase noted memory locations. In each sequence, the data memory will be erased if the CPD bit in the Configuration Word is programmed (clear).

To erase the program memory and Configuration Word (0x2007), the following sequence must be performed. Note the Calibration Words (0x2008-0x2009) and user ID (0x2000-0x2003) will not be erased.
1. Do a Bulk Erase Program Memory command.
2. Wait TERA to complete erase.

To erase the user ID (0x2000-0x2003), Configuration Word (0x2007) and program memory, use the following sequence. Note that the Calibration Words (0x2008-0x2009) will not be erased.
1. Perform Load Configuration with dummy data to point the Program Counter (PC) to 0x2000.
2. Perform a Bulk Erase Program Memory command.
3. Wait TERA to complete erase.

To erase the data memory, use the following sequence:
1. Perform a Bulk Erase Data Memory command.
2. Wait TERA to complete erase.

3.1.5 SERIAL PROGRAM/VERIFY OPERATION
The ICSPCLK pin is used as a clock input and the ICSPDAT pin is used for entering command bits and data input/output during serial operation. To input a command, ICSPCLK is cycled six times. Each command bit is latched on the falling edge of the clock with the LSb of the command being input first. The data input onto the ICSPDAT pin is required to have a minimum setup and hold time (see Table 6-1), with respect to the falling edge of the clock. Commands that have data associated with them (Read and Load) are specified to have a minimum delay of 1 \(\mu\)s between the command and the data. After this delay, the clock pin is cycled 16 times with the first cycle being a Start bit and the last cycle being a Stop bit.

During a read operation, the LSb will be transmitted onto the ICSPDAT pin on the rising edge of the second cycle. For a load operation, the LSb will be latched on the falling edge of the second cycle. A minimum 1 \(\mu\)s delay is also specified between consecutive commands, except for the End Programming command, which requires a 100 \(\mu\)s TDis.

All commands and data words are transmitted LSb first. Data is transmitted on the rising edge and latched on the falling edge of the ICSPCLK. To allow for decoding of commands and reversal of data pin configuration, a time separation of at least 1 \(\mu\)s is required between a command and a data word.

The commands that are available are described in Table 3-1.

<table>
<thead>
<tr>
<th>Command</th>
<th>Mapping (MSb … LSb)</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load Configuration</td>
<td>(x) (x) 0 0 0 0 0</td>
<td>0, data (14), 0</td>
</tr>
<tr>
<td>Load Data for Program Memory</td>
<td>(x) (x) 0 0 1 0 1</td>
<td>0, data (14), 0</td>
</tr>
<tr>
<td>Load Data for Data Memory</td>
<td>(x) (x) 0 0 1 1 1</td>
<td>0, data (8), zero (6), 0</td>
</tr>
<tr>
<td>Read Data from Program Memory</td>
<td>(x) (x) 0 1 0 0 0</td>
<td>0, data (14), 0</td>
</tr>
<tr>
<td>Read Data from Data Memory</td>
<td>(x) (x) 0 1 0 1 1</td>
<td>0, data (8), zero (6), 0</td>
</tr>
<tr>
<td>Increment Address</td>
<td>(x) (x) 0 1 1 1 0</td>
<td></td>
</tr>
<tr>
<td>Begin Programming</td>
<td>(x) 0 1 0 0 0 0 0</td>
<td>Internally Timed</td>
</tr>
<tr>
<td>Begin Programming</td>
<td>(x) 1 1 0 0 0 0 0</td>
<td>Externally Timed</td>
</tr>
<tr>
<td>End Programming</td>
<td>(x) 0 1 0 1 0 1 0</td>
<td></td>
</tr>
<tr>
<td>Bulk Erase Program Memory</td>
<td>(x) (x) 1 0 0 0 1</td>
<td>Internally Timed</td>
</tr>
<tr>
<td>Bulk Erase Data Memory</td>
<td>(x) (x) 1 0 1 1 1</td>
<td>Internally Timed</td>
</tr>
<tr>
<td>Row Erase Program Memory</td>
<td>(x) 1 0 0 0 0 1 1</td>
<td>Internally Timed</td>
</tr>
</tbody>
</table>

TABLE 3-1: COMMAND MAPPING FOR PIC12F6XX/16F6XX
3.1.5.1 Load Configuration

The Load Configuration command is used to access the Configuration Word (0x2007) and user ID (0x2000-0x2003). This command sets the Program Counter (PC) to address 0x2000 and loads the data latches with one word of data.

To access the configuration memory, send the Load Configuration command. Individual words within the configuration memory can be accessed by sending Increment Address commands and issuing load or read data for program memory.

After the 6-bit command is input, the ICSPCLK pin is cycled an additional 16 times for the Start bit, 14 bits of data and a Start bit (see Figure 3-4).

After the configuration memory is entered, the only way to get back to the program memory is to exit the Program/Verify mode by taking MCLR low (VIL).

3.1.5.2 Load Data for Program Memory

After receiving this command, the chip will load in a 14-bit “data word” when 16 cycles are applied, as described previously. A timing diagram for the Load Data For Program Memory command is shown in Figure 3-5.
3.1.5.3 Load Data for Data Memory

After receiving this command, the chip will load in a 14-bit "data word" when 16 cycles are applied. However, the data memory is only eight bits wide and thus, only the first eight bits of data after the Start bit will be programmed into the data memory. It is still necessary to cycle the clock the full 16 cycles in order to allow the internal circuitry to reset properly. The data memory contains 256 bytes.

FIGURE 3-6: LOAD DATA FOR DATA MEMORY COMMAND

3.1.5.4 Read Data from Program Memory

After receiving this command, the chip will transmit data bits out of the program memory (user or configuration) currently accessed, starting with the second rising edge of the clock input. The data pin will go into Output mode on the second rising clock edge, and it will revert to Input mode (high-impedance) after the 16th rising edge.

If the program memory is code-protected (CP = 0), the data is read as zeros.

FIGURE 3-7: READ DATA FROM PROGRAM MEMORY COMMAND
3.1.5.5 Read Data from Data Memory

After receiving this command, the chip will transmit data bits out of the data memory, starting with the second rising edge of the clock input. The ICSPDAT pin will go into Output mode on the second rising edge and it will revert to Input mode (high-impedance) after the 16th rising edge. As previously stated, the data memory is eight bits wide and, therefore, only the first eight bits that are output are actual data. If the data memory is code-protected, the data is read as all zeros. A timing diagram of this command is shown in Figure 3-8.

FIGURE 3-8: READ DATA FROM DATA MEMORY COMMAND

3.1.5.6 Increment Address

The PC is incremented when this command is received. A timing diagram of this command is shown in Figure 3-9.

It is not possible to decrement the address counter. To reset this counter, the user should exit and re-enter Program/Verify mode.

FIGURE 3-9: INCREMENT ADDRESS COMMAND (PROGRAM/VERIFY)
3.1.5.7 Begin Programming (Internally Timed)

A Load command must be given before every Begin Programming command. Programming of the appropriate memory (user program memory, configuration memory or data memory) will begin after this command is received and decoded. An internal timing mechanism executes a write. The user must allow for program cycle time for programming to complete. No End Programming command is required. The addressed location is not erased before programming.

FIGURE 3-10: BEGIN PROGRAMMING COMMAND (INTERNALLY TIMED)

3.1.5.8 Begin Programming (Externally Timed)

A Load command must be given before every Begin Programming command. Programming of the appropriate memory (program memory, configuration or data memory) will begin after this command is received and decoded. Programming requires (TPROG2) time and is terminated using an End Programming command. The addressed location is not erased before programming.

FIGURE 3-11: BEGIN PROGRAMMING (EXTERNALLY TIMED)
3.1.5.9  End Programming

**FIGURE 3-12: END PROGRAMMING (SERIAL PROGRAM/VERIFY)**

3.1.5.10  Bulk Erase Program Memory

After this command is performed, the entire program memory and Configuration Word (0x2007) is erased. Data memory will also be erased if the CPD bit in the Configuration Word is programmed (clear). See Section 3.1.4 “Erase Algorithms” for erase sequences.

**FIGURE 3-13: BULK ERASE PROGRAM MEMORY COMMAND**
3.1.5.11  Bulk Erase Data Memory

To perform an erase of the data memory, the following sequence must be performed.

1. Perform a Bulk Erase Data Memory command.
2. Wait TERA to complete Bulk Erase.

Data memory won't erase if code-protected (CPD = 0).

**Note:** All Bulk Erase operations must take place between 4.5V and 5.5V VDD for PIC12F6XX/16F6XX and 2.0V to 5.5V VDD for PIC12F6XX/16F6XX-ICD.

FIGURE 3-14:  BULK ERASE DATA MEMORY COMMAND

3.1.5.12  Row Erase Program Memory

This command erases the 16-word row of program memory pointed to by PC<11:4>. If the program memory array is protected (CP = 0) or the PC points to configuration memory (>0x2000), the command is ignored.

To perform a Row Erase Program Memory, the following sequence must be performed.

1. Execute a Row Erase Program Memory command.
2. Wait TERA to complete a row erase.

FIGURE 3-15:  ROW ERASE PROGRAM MEMORY COMMAND
FIGURE 3-16: ONE-WORD PROGRAMMING FLOWCHART

Note 1: This step is optional if the device has already been erased or has not been previously programmed.

Note 2: This step is optional if the data memory does not require updates.

Note 3: If the device is code-protected or must be completely erased, then Bulk Erase the device per Figure 3-20.
FIGURE 3-17: FOUR-WORD PROGRAMMING FLOWCHART

Note 1: This step is optional if the device is erased or not previously programmed.
2: Verification in Four-Word mode is accomplished after programming by reading back the entire memory.
3: This step is optional if the data memory does not require updates.
4: If the device is code-protected or must be completely erased, then Bulk Erase the device per Figure 3-20.
FIGURE 3-18: PROGRAM FLOWCHART – PIC12F6XX/16F6XX CONFIGURATION MEMORY

Start

Load Configuration

One-word Program Cycle (User ID)

Read Data From Program Memory Command

Data Correct?

Yes

Increment Address Command

No

Address = 0x2004?

Yes

Increment Address Command

No

One-word Program Cycle (Config. bits)

Read Data From Program Memory Command

Data Correct?

Yes

Done

No

Report Programming Failure

Program Cycle

Load Data for Program Memory

Begin Programming Command (Internally timed)

Wait TPROG1

End Programming

Wait TDIS

Begin Programming Command (Externally timed)

Wait TPROG2

Note: Ensure that a device Bulk Erase has been performed or that the device is blank prior to programming the configuration memory.
FIGURE 3-19: PROGRAM FLOWCHART – PIC12F6XX/16F6XX DATA MEMORY

Start

Program Cycle

Bulk Erase Data Memory

Read Data From Data Memory Command

Data Correct?

No

Increment Address Command

Yes

Report Programming Failure

Programmed

All Locations Done?

No

Yes

Done

PROGRAM CYCLE

Load Data for Data Memory

Begin Programming Command (Internally timed)

Wait TPROG1

End Programming

Wait TDIS

Begin Programming Command (Externally timed)

Wait TPROG2
FIGURE 3-20: PROGRAM FLOWCHART – ERASE FLASH DEVICE(1)

Start

Read and Store Calibration Memory Values (Figure 3-21)

Bulk Erase Program Memory

Load Configuration

Bulk Erase Program Memory

Bulk Erase Data Memory

Read and Verify Calibration Memory Values (Figure 3-21)

Done
FIGURE 3-21: CALIBRATION WORD VERIFICATION FLOWCHART

Note 1: This step is not required for the Read and Store Calibration Memory Values procedure.

2: The device should not be used if verification of the Calibration Word locations fails. This information should be reported to the user through the user interface of the device programmer.

3: Several devices within this family do not possess Calibration Word 2. The remainder of this procedure is unnecessary for those devices without Calibration Word 2.
### 4.0 CONFIGURATION WORD

The PIC12F6XX/16F6XX has several Configuration bits. These bits can be programmed (reads '0') or left unchanged (reads '1'), to select various device configurations.

**REGISTER 4-1: CONFIG(1): CONFIGURATION WORD (ADDRESS:2007h) – PIC12F635/PIC16F636/PIC16F639**

<table>
<thead>
<tr>
<th>bit 13</th>
<th>WURE</th>
<th>FCMEN</th>
<th>IESO</th>
<th>BOREN1</th>
<th>BOREN0</th>
<th>CPD</th>
</tr>
</thead>
<tbody>
<tr>
<td>U/P-1</td>
<td>R/P-1</td>
<td>R/P-1</td>
<td>R/P-1</td>
<td>R/P-1</td>
<td>R/P-1</td>
<td>R/P-1</td>
</tr>
</tbody>
</table>

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

#### bit 13 - Unimplemented: Read as ‘1’

#### bit 12

- **WURE**: Wake-up Reset Enable bit
  - 1 = Standard wake-up and continue enabled
  - 0 = Wake-up and Reset enabled

#### bit 11

- **FCMEN**: Fail-Safe Clock Monitor Enable bit
  - 1 = Fail-Safe Clock Monitor enabled
  - 0 = Fail-Safe Clock Monitor disabled

#### bit 10

- **IESO**: Internal-External Switch Over bit
  - 1 = Internal External Switchover mode enabled
  - 0 = Internal External Switchover mode disabled

#### bit 8-9

- **BOREN<1:0>**: Brown-out Reset Enable bits
  - 11 = BOR enabled and SBOREN bit disabled
  - 10 = BOR enabled while running and disabled in Sleep. SBOREN bit disabled.
  - 01 = SBOREN in the PCON register controls BOR function
  - 00 = BOR and SBOREN disabled

#### bit 7

- **CPD**: Data Code Protection bit(2)
  - 1 = Data memory is not protected
  - 0 = Data memory is external read-protected

#### bit 6

- **CP**: Code Protection bit(3)
  - 1 = Program memory is not code-protected
  - 0 = Program memory is external read and write-protected

#### bit 5

- **MCLRE**: MCLR Pin Function Select bit(5)
  - 1 = MCLR pin is MCLR function and weak internal pull-up is enabled
  - 0 = MCLR pin is alternate function, MCLR function is internally disabled

#### bit 4

- **PWRT**: Power-up Timer Enable bit(4)
  - 1 = PWRT disabled
  - 0 = PWRT enabled

#### bit 3

- **WDTE**: Watchdog Timer Enable bit
  - 1 = WDT enabled
  - 0 = WDT disabled and can be enabled using SWDTEN in the WDTCON register

#### bit 2-0

- **FOSC<2:0>**: Oscillator Selection bits
  - 000 = LP oscillator: Low-power crystal on RA5(GP5)/OSC1/CLKIN and RA4(GP4)/OSC2/CLKOUT
  - 001 = XT oscillator: Crystal/resonator on RA5(GP5)/OSC1/CLKIN and RA4(GP4)/OSC2/CLKOUT
  - 010 = HS oscillator: High-speed crystal/resonator on RA5(GP5)/OSC1/CLKIN and RA4(GP4)/OSC2/CLKOUT
  - 011 = EC: I/O function on RA4(GP4)/OSC2/CLKOUT, CLKin on RA5(GP5)/OSC1/CLKIN
  - 100 = INTOSCIO oscillator: I/O function on RA4(GP4)/OSC2/CLKOUT, I/O function on RA5(GP5)/OSC1/CLKIN
  - 101 = INTOSC oscillator: CLKOUT function on RA4(GP4)/OSC2/CLKOUT, I/O function on RA5(GP5)/OSC1/CLKIN
  - 110 = EXTRCIO oscillator: I/O function on RA4(GP4)/OSC2/CLKOUT, RC on RA5(GP5)/OSC1/CLKIN
  - 111 = EXTRC oscillator: CLKOUT function on RA4(GP4)/OSC2/CLKOUT, RC on RA5(GP5)/OSC1/CLKIN

**Note**

1. This Configuration Word register applies to PIC12F635/PIC16F636/PIC16F639 devices only.
2. The entire data memory will be erased when the code protection is turned off.
3. The entire program memory will be erased when the code protection is turned off.
4. Enabling Brown-out Detect does not automatically enable Power-up Timer.
5. When MCLRC is asserted in INTOSC or RC mode, the internal clock oscillator is disabled.

<table>
<thead>
<tr>
<th>U-1</th>
<th>U-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>bit 13</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bit 12</th>
<th>U-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>bit 7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>bit 13-12</th>
<th>Unimplemented: Read as ‘1’.</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit 11</td>
<td>FCMEN: Fail-Safe Clock Monitor Enable bit</td>
</tr>
<tr>
<td></td>
<td>1 = Fail-Safe Clock Monitor enabled</td>
</tr>
<tr>
<td></td>
<td>0 = Fail-Safe Clock Monitor disabled</td>
</tr>
<tr>
<td>bit 10</td>
<td>IESO: Internal-External Switch Over bit</td>
</tr>
<tr>
<td></td>
<td>1 = Internal External Switchover mode enabled</td>
</tr>
<tr>
<td></td>
<td>0 = Internal External Switchover mode disabled</td>
</tr>
<tr>
<td>bit 9-8</td>
<td>BOREN&lt;1:0&gt;: Brown-out Reset Enable bits(4)</td>
</tr>
<tr>
<td></td>
<td>11 = BOR enabled and SBOREN bit disabled</td>
</tr>
<tr>
<td></td>
<td>10 = BOR enabled while running and disabled in Sleep. SBOREN bit disabled.</td>
</tr>
<tr>
<td></td>
<td>01 = SBOREN in the PCON register controls BOR function</td>
</tr>
<tr>
<td></td>
<td>00 = BOR and SBOREN disabled</td>
</tr>
<tr>
<td>bit 7</td>
<td>CPD: Code Protection Data bit(2)</td>
</tr>
<tr>
<td></td>
<td>1 = Data memory is not protected</td>
</tr>
<tr>
<td></td>
<td>0 = Data memory is external read-protected</td>
</tr>
<tr>
<td>bit 6</td>
<td>CP: Code Protection bit(3)</td>
</tr>
<tr>
<td></td>
<td>1 = Program memory is not code-protected</td>
</tr>
<tr>
<td></td>
<td>0 = Program memory is external read and write-protected</td>
</tr>
<tr>
<td>bit 5</td>
<td>MCLRE: MCLR Pin Function Select(5) bit</td>
</tr>
<tr>
<td></td>
<td>1 = MCLR pin is MCLR function and weak internal pull-up is enabled</td>
</tr>
<tr>
<td></td>
<td>0 = MCLR pin is alternate function, MCLR function is internally disabled</td>
</tr>
<tr>
<td>bit 4</td>
<td>PWRT: Power-up Timer Enable bit(4)</td>
</tr>
<tr>
<td></td>
<td>1 = PWRT disabled</td>
</tr>
<tr>
<td></td>
<td>0 = PWRT enabled</td>
</tr>
<tr>
<td>bit 3</td>
<td>WDTE: Watchdog Timer Enable bit</td>
</tr>
<tr>
<td></td>
<td>1 = WDT enabled</td>
</tr>
<tr>
<td></td>
<td>0 = WDT disabled and can be enabled using SWDTEN in the WDTCNN register</td>
</tr>
<tr>
<td>bit 2-0</td>
<td>FOSC&lt;2:0&gt;: Oscillator Selection bits</td>
</tr>
<tr>
<td></td>
<td>000 = LP oscillator: Low-power crystal on RA5(GP5)/OSC1/CLKIN and RA4(GP4)/OSC2/CLKOUT</td>
</tr>
<tr>
<td></td>
<td>001 = XT oscillator: Crystal/resonator on RA5(GP5)/OSC1/CLKIN and RA4(GP4)/OSC2/CLKOUT</td>
</tr>
<tr>
<td></td>
<td>010 = HS oscillator: High-speed crystal/resonator on RA5(GP5)/OSC1/CLKIN and RA4(GP4)/OSC2/CLKOUT</td>
</tr>
<tr>
<td></td>
<td>011 = EC: I/O function on RA4(GP4)/OSC2/CLKOUT, CLKIN on RA5(GP5)/OSC1/CLKIN</td>
</tr>
<tr>
<td></td>
<td>100 = INTOSC8 oscillator: I/O function on RA4(GP4)/OSC2/CLKOUT, /O function on RA5(GP5)/OSC1/CLKIN</td>
</tr>
<tr>
<td></td>
<td>101 = INTOSC oscillator: CLKOUT function on RA4(GP4)/OSC2/CLKOUT, I/O function on RA5(GP5)/OSC1/CLKIN</td>
</tr>
<tr>
<td></td>
<td>110 = EXTRC oscillator: I/O function on RA4(GP4)/OSC2/CLKOUT, RC on RA5(GP5)/OSC1/CLKIN</td>
</tr>
<tr>
<td></td>
<td>111 = EXTRC oscillator: CLKOUT function on RA4(GP4)/OSC2/CLKOUT, RC on RA5(GP5)/OSC1/CLKIN</td>
</tr>
</tbody>
</table>

#### Note:
1. This Configuration Word register applies to PIC16F631/677/685/687/689/690 (not including PIC12F635/PIC16F636/PIC16F639) only.
2. The entire data memory will be erased when the code protection is turned off.
3. The entire program memory will be erased when the code protection is turned off.
4. Enabling Brown-out Detect does not automatically enable Power-up Timer.
5. When MCLR is asserted in INTOSC or RC mode, the internal clock oscillator is disabled.
6. For PIC16F685/PIC16F687/PIC16F689/PIC16F690, the pin is RA4/AN3/T1G/OSC2/CLKOUT.
### REGISTER 4-3: CALIB (1): CALIBRATION WORD (ADDRESS: 2008h) – PIC12F683/684/688 (2), (3)

<table>
<thead>
<tr>
<th>R/P-1</th>
<th>U-1</th>
<th>P/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>—</td>
<td></td>
<td>FCAL6</td>
<td>FCAL5</td>
<td>FCAL4</td>
<td>FCAL3</td>
<td>FCAL2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bit 13</th>
<th>bit 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>—</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>R/P-1</th>
<th>U-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>FCAL0</td>
<td></td>
<td>POR1</td>
<td>POR0</td>
<td>BOR2</td>
<td>BOR1</td>
<td>BOR0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bit 6</th>
<th>bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>—</td>
<td></td>
</tr>
</tbody>
</table>

#### Legend:

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

- **bit 13** Unimplemented: Read as ‘0’
- **bit 12-6** FCAL<6:0>: Internal Oscillator Calibration bits
  - 0111111 = Maximum frequency
  - 0000001 = Center frequency
  - 1111111 = Minimum frequency
- **bit 5** Unimplemented: Read as ‘0’
- **bit 4-3** POR<1:0>: POR Calibration bits
  - 00 = Lowest POR voltage
  - 11 = Highest POR voltage
- **bit 2-0** BOR<2:0>: BOR Calibration bits
  - 000 = Reserved
  - 001 = Lowest BOR voltage
  - 111 = Highest BOR voltage

**Note 1:** This Calibration Word register applies to PIC12F683/PIC16F684/PIC16F688 devices only.
**Note 2:** This location does not participate in Bulk Erase operations if the procedure in Figure 3-20 is used.
**Note 3:** Calibration bits are reserved for factory calibration. These values can and will change across the entire range, therefore, specific values and available adjustment range cannot be specified.
**REGISTER 4-4: CALIB**

CALIB is the Calibration Word register (ADDRESS: 2008h) available in PIC16F631/677/685/687/689/690 devices. It is used for adjusting the device's clock frequency and POR/BOR voltages.

<table>
<thead>
<tr>
<th>Bit 13</th>
<th>Bit 12-6</th>
<th>Bit 5-3</th>
<th>Bit 2-0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unimplemented</td>
<td>FCAL&lt;6:0&gt;: Internal Oscillator Calibration bits</td>
<td>POR&lt;2:0&gt;: POR Calibration bits</td>
<td>BOR&lt;2:0&gt;: BOR Calibration bits</td>
</tr>
</tbody>
</table>

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

**Note:**
1. This Calibration Word register applies to PIC16F631/677/685/687/689/690 devices only.
2. This location does not participate in Bulk Erase operations if the procedure in Figure 3-20 is used.
3. Calibration bits are reserved for factory calibration. These values can and will change across the entire range, therefore, specific values and available adjustment range cannot be specified.
4. The calibration bits must be read, preserved, then replaced by the user during Program Memory Bulk Erase operation with PC = 2008h.
**REGISTER 4-5: CALIB1: CALIBRATION WORD 1 (ADDRESS: 2008H) – PIC12F635/636/639\(^{(1)}\)**

<table>
<thead>
<tr>
<th>U-1</th>
<th>P/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>FCAL6</td>
<td>FCAL5</td>
<td>FCAL4</td>
<td>FCAL3</td>
<td>FCAL2</td>
<td>FCAL1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bit 13</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>bit 7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>R/P-1</th>
<th>P/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>FCAL0</td>
<td>POR2</td>
<td>POR1</td>
<td>POR0</td>
<td>BOR2</td>
<td>BOR1</td>
<td>BOR0</td>
<td></td>
</tr>
<tr>
<td>bit 6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>bit 0</td>
</tr>
</tbody>
</table>

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

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

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

<table>
<thead>
<tr>
<th>bit 13</th>
<th>Unimplemented: Read as ‘1’</th>
</tr>
</thead>
<tbody>
<tr>
<td>bit 12-6</td>
<td><strong>FCAL&lt;6:0&gt;: Internal Oscillator Calibration bits</strong></td>
</tr>
<tr>
<td></td>
<td>0111111 = Maximum frequency</td>
</tr>
<tr>
<td></td>
<td>•</td>
</tr>
<tr>
<td></td>
<td>•</td>
</tr>
<tr>
<td></td>
<td>0000001</td>
</tr>
<tr>
<td></td>
<td>0000000 = Center frequency. Oscillator is running at the calibrated frequency</td>
</tr>
<tr>
<td></td>
<td>1111111</td>
</tr>
<tr>
<td></td>
<td>•</td>
</tr>
<tr>
<td></td>
<td>•</td>
</tr>
<tr>
<td></td>
<td>1000000 = Minimum frequency</td>
</tr>
<tr>
<td>bit 5-3</td>
<td><strong>POR&lt;2:0&gt;: POR Calibration bits</strong></td>
</tr>
<tr>
<td></td>
<td>111 = Maximum POR voltage</td>
</tr>
<tr>
<td></td>
<td>110</td>
</tr>
<tr>
<td></td>
<td>101</td>
</tr>
<tr>
<td></td>
<td>100 = Center POR voltage</td>
</tr>
<tr>
<td></td>
<td>000 = Center POR voltage</td>
</tr>
<tr>
<td></td>
<td>001</td>
</tr>
<tr>
<td></td>
<td>010</td>
</tr>
<tr>
<td></td>
<td>011 = Minimum POR voltage</td>
</tr>
<tr>
<td>bit 2-0</td>
<td><strong>BOR&lt;2:0&gt;: BOR Calibration bits</strong></td>
</tr>
<tr>
<td></td>
<td>111 = Maximum BOR voltage</td>
</tr>
<tr>
<td></td>
<td>110</td>
</tr>
<tr>
<td></td>
<td>101</td>
</tr>
<tr>
<td></td>
<td>100 = Center BOR voltage</td>
</tr>
<tr>
<td></td>
<td>000 = Center BOR voltage</td>
</tr>
<tr>
<td></td>
<td>001</td>
</tr>
<tr>
<td></td>
<td>010</td>
</tr>
<tr>
<td></td>
<td>011 = Minimum BOR voltage</td>
</tr>
</tbody>
</table>

**Note 1:** This location does not participate in Bulk Erase operation, unless PC = 2008h.
4.1 Device ID Word

The device ID word for the PIC12F6XX/16F6XX is located at 2006h. This location cannot be erased.

### TABLE 4-1: DEVICE ID VALUES

<table>
<thead>
<tr>
<th>Device</th>
<th>Device ID Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dev</td>
<td>Rev</td>
</tr>
<tr>
<td>PIC12F635</td>
<td>00 1111 101</td>
</tr>
<tr>
<td>PIC12F683</td>
<td>00 0100 011</td>
</tr>
<tr>
<td>PIC16F631</td>
<td>01 0100 001</td>
</tr>
<tr>
<td>PIC16F636</td>
<td>01 0000 101</td>
</tr>
<tr>
<td>PIC16F639</td>
<td>01 0000 101</td>
</tr>
<tr>
<td>PIC16F677</td>
<td>01 0100 010</td>
</tr>
<tr>
<td>PIC16F684</td>
<td>01 0000 100</td>
</tr>
<tr>
<td>PIC16F685</td>
<td>00 0100 101</td>
</tr>
<tr>
<td>PIC16F687</td>
<td>01 0011 001</td>
</tr>
<tr>
<td>PIC16F688</td>
<td>01 0001 100</td>
</tr>
<tr>
<td>PIC16F689</td>
<td>01 0011 010</td>
</tr>
<tr>
<td>PIC16F690</td>
<td>01 0100 000</td>
</tr>
</tbody>
</table>
5.0 CODE PROTECTION

For PIC12F6XX/16F6XX, once the CP bit is programmed to ‘0’, all program memory locations read all ‘0’s. The user ID locations and the Configuration Word read out in an unprotected fashion. Further programming is disabled for the entire program memory.

Data memory is protected with its own code-protect bit (CPD). When enabled, the data memory can still be programmed and read using the EECON1 register (see the applicable data sheet for more information).

The user ID locations and the Configuration Word can be programmed regardless of the state of the CP and CPD bits.

5.1 Disabling Code Protection

It is recommended to use the procedure in Figure 3-20 to disable code protection of the device. This sequence will erase the program memory, data memory, Configuration Word (0x2007) and user ID locations (0x2000-0x2003). The Calibration Words (0x2008-0x2009) will not be erased.

Note: To ensure system security, if CPD bit = 0, Bulk Erase Program Memory command will also erase data memory.

5.2 Embedding Configuration Word and User ID Information in the Hex File

To allow portability of code, the programmer is required to read the Configuration Word and user ID locations from the hex file when loading the hex file. If Configuration Word information was not present in the hex file, a simple warning message may be issued. Similarly, while saving a hex file, Configuration Word and user ID information must be included. An option to not include this information may be provided.

Specifically for the PIC12F6XX/16F6XX, the data memory should also be embedded in the hex file (see Section 5.3.2 "Embedding Data Memory Contents in Hex File").

Microchip Technology Incorporated feels strongly that this feature is important for the benefit of the end customer.

5.3 Checksum Computation

5.3.1 CHECKSUM

Checksum is calculated by reading the contents of the PIC12F6XX/16F6XX memory locations and adding up the opcodes up to the maximum user addressable location (e.g., 0x7FF for the PIC16F684). Any Carry bits exceeding 16 bits are neglected. Finally, the Configuration Word (appropriately masked) is added to the checksum. Checksum computation for the PIC12F6XX/16F6XX devices is shown in Table 5-1.

The checksum is calculated by summing the following:

- The contents of all program memory locations
- The Configuration Word, appropriately masked
- Masked user ID locations (when applicable)

The Least Significant 16 bits of this sum is the checksum.

The following table describes how to calculate the checksum for each device. Note that the checksum calculation differs depending on the code-protect setting. Since the program memory locations read out zeroes when code-protected, the table describes how to manipulate the actual program memory values to simulate values that would be read from a protected device. When calculating a checksum by reading a device, the entire program memory can simply be read and summed. The Configuration Word and user ID locations can always be read regardless of the code-protect setting.

Note: Some older devices have an additional value added in the checksum. This is to maintain compatibility with older device programmer checksums.
### TABLE 5-1: CHECKSUM COMPUTATIONS

<table>
<thead>
<tr>
<th>Device</th>
<th>Code Protect</th>
<th>Checksum*</th>
<th>Blank Value</th>
<th>0x25E6 at 0 and Max. Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC12F635</td>
<td>CP = 1, CPD = 1</td>
<td>SUM[0x000:0x03FF] + (CFGW &amp; 1FFF) (CFGW &amp; 1FFF) + SUM_ID</td>
<td>0x1BFF</td>
<td>0x7EDC</td>
</tr>
<tr>
<td></td>
<td>CP = 0, CPD = 1</td>
<td>(CFGW &amp; 0FFF) + SUM_ID</td>
<td>0x3BBE</td>
<td>0x78C</td>
</tr>
<tr>
<td>PIC12F683</td>
<td>CP = 1, CPD = 1</td>
<td>SUM[0x000:0x07FF] + (CFGW &amp; 0FFF)</td>
<td>0x07FF</td>
<td>0x3DEC</td>
</tr>
<tr>
<td></td>
<td>CP = 0, CPD = 1</td>
<td>(CFGW &amp; 0x0FFF) + SUM_ID</td>
<td>0x17BE</td>
<td>0xE8C</td>
</tr>
<tr>
<td>PIC16F631</td>
<td>CP = 1, CPD = 1</td>
<td>SUM[0x000:0x03FF] + (CFGW &amp; 0FFF)</td>
<td>0x0BFF</td>
<td>0x7DEC</td>
</tr>
<tr>
<td></td>
<td>CP = 0, CPD = 1</td>
<td>(CFGW &amp; 0x0FFF) + SUM_ID</td>
<td>0x1BBE</td>
<td>0xE8C</td>
</tr>
<tr>
<td>PIC16F636</td>
<td>CP = 1, CPD = 1</td>
<td>SUM[0x000:0x07FF] + (CFGW &amp; 1FFF)</td>
<td>0x17FF</td>
<td>0xE3EC</td>
</tr>
<tr>
<td></td>
<td>CP = 0, CPD = 1</td>
<td>(CFGW &amp; 0x1FFF) + SUM_ID</td>
<td>0x37BE</td>
<td>0x038C</td>
</tr>
<tr>
<td>PIC16F639</td>
<td>CP = 1, CPD = 1</td>
<td>SUM[0x000:0x07FF] + (CFGW &amp; 1FFF)</td>
<td>0x17FF</td>
<td>0xE3EC</td>
</tr>
<tr>
<td></td>
<td>CP = 0, CPD = 1</td>
<td>(CFGW &amp; 0x1FFF) + SUM_ID</td>
<td>0x37BE</td>
<td>0x038C</td>
</tr>
<tr>
<td>PIC16F677</td>
<td>CP = 1, CPD = 1</td>
<td>SUM[0x000:0x07FF] + (CFGW &amp; 0FFF)</td>
<td>0x07FF</td>
<td>0x3DEC</td>
</tr>
<tr>
<td></td>
<td>CP = 0, CPD = 1</td>
<td>(CFGW &amp; 0x0FFF) + SUM_ID</td>
<td>0x17BE</td>
<td>0xE8C</td>
</tr>
<tr>
<td>PIC16F684</td>
<td>CP = 1, CPD = 1</td>
<td>SUM[0x000:0x07FF] + (CFGW &amp; 0FFF)</td>
<td>0x07FF</td>
<td>0x3DEC</td>
</tr>
<tr>
<td></td>
<td>CP = 0, CPD = 1</td>
<td>(CFGW &amp; 0x0FFF) + SUM_ID</td>
<td>0x17BE</td>
<td>0xE8C</td>
</tr>
<tr>
<td>PIC16F685</td>
<td>CP = 1, CPD = 1</td>
<td>SUM[0x000:0x0FFF] + (CFGW &amp; 0FFF)</td>
<td>0xFFF</td>
<td>0xCB8C</td>
</tr>
<tr>
<td></td>
<td>CP = 0, CPD = 1</td>
<td>(CFGW &amp; 0x0FFF) + SUM_ID</td>
<td>0xFBFE</td>
<td>0xDB8C</td>
</tr>
<tr>
<td>PIC16F687</td>
<td>CP = 1, CPD = 1</td>
<td>SUM[0x000:0x07FF] + (CFGW &amp; 0FFF)</td>
<td>0x07FF</td>
<td>0x3DEC</td>
</tr>
<tr>
<td></td>
<td>CP = 0, CPD = 1</td>
<td>(CFGW &amp; 0x0FFF) + SUM_ID</td>
<td>0x17BE</td>
<td>0xE8C</td>
</tr>
<tr>
<td>PIC16F688</td>
<td>CP = 1, CPD = 1</td>
<td>SUM[0x000:0x0FFF] + (CFGW &amp; 0FFF)</td>
<td>0xFFF</td>
<td>0xCB8C</td>
</tr>
<tr>
<td></td>
<td>CP = 0, CPD = 1</td>
<td>(CFGW &amp; 0x0FFF) + SUM_ID</td>
<td>0xFBFE</td>
<td>0xDB8C</td>
</tr>
<tr>
<td>PIC16F689</td>
<td>CP = 1, CPD = 1</td>
<td>SUM[0x000:0x0FFF] + (CFGW &amp; 0FFF)</td>
<td>0xFFF</td>
<td>0xCB8C</td>
</tr>
<tr>
<td></td>
<td>CP = 0, CPD = 1</td>
<td>(CFGW &amp; 0x0FFF) + SUM_ID</td>
<td>0xFBFE</td>
<td>0xDB8C</td>
</tr>
<tr>
<td>PIC16F690</td>
<td>CP = 1, CPD = 1</td>
<td>SUM[0x000:0x0FFF] + (CFGW &amp; 0FFF)</td>
<td>0xFFF</td>
<td>0xCB8C</td>
</tr>
<tr>
<td></td>
<td>CP = 0, CPD = 1</td>
<td>(CFGW &amp; 0x0FFF) + SUM_ID</td>
<td>0xFBFE</td>
<td>0xDB8C</td>
</tr>
</tbody>
</table>

**Legend:**
- CFGW = Configuration Word. Example calculations assume Configuration Word is erased (all ‘1’s).
- SUM[a:b] = [Sum of locations a to b inclusive]
- SUM_ID = User ID locations masked by 0xF then made into a 16-bit value with ID0 as the Most Significant nibble.
  - For example, ID0 = 0x1, ID1 = 0x2, ID3 = 0x3, ID4 = 0x4, then SUM_ID = 0x1234.
  - The 4 LSb’s of the unprotected checksum is used for the example calculations.
- *Checksum = [Sum of all the individual expressions] MODULO [0xFFFF]*
- + = Addition
- & = Bitwise AND

### 5.3.2 EMBEDDING DATA MEMORY CONTENTS IN HEX FILE

The programmer should be able to read data memory information from a hex file and conversely (as an option), write data memory contents to a hex file along with program memory information and Configuration Word (0x0007) and user ID (0x2000-0x2003) information.

The 256 data memory locations are logically mapped starting at address 0x2100. The format for data memory storage is one data byte per address location, LSb aligned.
### 6.0 PROGRAM/VERIFY MODE ELECTRICAL CHARACTERISTICS

#### TABLE 6-1: AC/DC CHARACTERISTICS TIMING REQUIREMENTS FOR PROGRAM/VERIFY MODE

<table>
<thead>
<tr>
<th>AC/DC CHARACTERISTICS</th>
<th>Standard Operating Conditions (unless otherwise stated)</th>
<th>Operating Temperature -40°C ≤ TA ≤ +85°C</th>
<th>Operating Voltage 4.5V ≤ VDD ≤ 5.5V</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Sym.</strong></td>
<td><strong>Characteristics</strong></td>
<td><strong>Min.</strong></td>
<td><strong>Typ.</strong></td>
</tr>
<tr>
<td><strong>General</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VDD</td>
<td>VDD level for read/write operations, program and data memory</td>
<td>2.0</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>VDD level for Bulk Erase operations, program and data memory</td>
<td>2.0</td>
<td>4.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>High voltage on MCLR for Program/Verify mode entry</td>
<td>10</td>
<td>—</td>
</tr>
<tr>
<td>VIHH</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TVHHR</td>
<td>MCLR rise time (Vss to VHH) for Program/Verify mode entry</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>TPPDP</td>
<td>Hold time after VPP changes</td>
<td>5</td>
<td>—</td>
</tr>
<tr>
<td>VH1</td>
<td>(ICSPCLK, ICSPDAT) input high level</td>
<td>0.8 VDD</td>
<td>—</td>
</tr>
<tr>
<td>VL1</td>
<td>(ICSPCLK, ICSPDAT) input low level</td>
<td>0.2 VDD</td>
<td>—</td>
</tr>
<tr>
<td>TSET0</td>
<td>ICSPCLK, ICSPDAT setup time before MCLR↑ (Program/Verify mode selection pattern setup time)</td>
<td>100</td>
<td>—</td>
</tr>
<tr>
<td>THLD0</td>
<td>Hold time after VDD changes</td>
<td>0</td>
<td>—</td>
</tr>
<tr>
<td><strong>Serial Program/Verify</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSET1</td>
<td>Data in setup time before clock↓</td>
<td>100</td>
<td>—</td>
</tr>
<tr>
<td>THLD1</td>
<td>Data in hold time after clock↓</td>
<td>100</td>
<td>—</td>
</tr>
<tr>
<td>TDLY1</td>
<td>Data input not driven to next clock input (delay required between command/data or command/command)</td>
<td>1.0</td>
<td>—</td>
</tr>
<tr>
<td>TDLY2</td>
<td>Delay between clock↓ to clock↑ of next command or data</td>
<td>1.0</td>
<td>—</td>
</tr>
<tr>
<td>TDLY3</td>
<td>Clock↑ to data out valid (during a Read Data command)</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>TERA</td>
<td>Erase cycle time</td>
<td>—</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TPROG1</td>
<td>Programming cycle time (internally timed)</td>
<td>3</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>—</td>
</tr>
<tr>
<td>TPROG2</td>
<td>Programming cycle time (externally timed)</td>
<td>3</td>
<td>—</td>
</tr>
<tr>
<td>TDIS</td>
<td>Time delay from program to compare (HV discharge time)</td>
<td>100</td>
<td>—</td>
</tr>
</tbody>
</table>
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, FlashFlex, flexPWR, JukeBlox, KEELOG, KEELOG logo, Kleer, LANCheck, MediaLB, MOST, MOST logo, MPLAB, OptoLyzer, PIC, PICSTART, PIC\(^2\) logo, RightTouch, SpyNIC, SST, SST Logo, SuperFlash and UNI/O are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

The Embedded Control Solutions Company and mTouch are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Analog-for-the-Digital Age, BodyCom, chipKIT, chipKIT logo, CodeGuard, dsPICDEM, dsPICDEM.net, ECAN, In-Circuit Serial Programming, ICSP, Inter-Chip Connectivity, KleerNet, KleerNet logo, MiWi, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, RightTouch logo, REAL ICE, SQI, Serial Quad I/O, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, 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.

Silicon Storage Technology is a registered trademark of Microchip Technology Inc. in other countries.

GestIC is a registered trademarks of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

All other trademarks mentioned herein are property of their respective companies.

© 2009-2015, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

ISBN: 978-1-63276-942-8

Microchip received ISO/TS-16949:2009 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\(^\circ\) MCUs and dsPIC\(^\circ\) DSCs, Keeloc\(^\circ\) 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.