This document includes the programming specifications for the following devices:

- MCP19214
- MCP19215

1.0 PROGRAMMING THE MCP19214 AND MCP19215 DEVICES

The MCP19214/5 devices are programmed using a serial method. The Serial mode will allow these devices to be programmed while in the user's system. These programming specifications apply to all of the above devices in all packages.

1.1 Hardware Requirements

This family of devices requires one power supply for $V_{IN}$ (see Table 6-1). The $V_{DD}$ that is used to bias all internal analog and digital circuitry is internally generated and regulated to 5V.

1.2 Program/Verify Mode

The Program/Verify mode for this family of devices allows programming the user program memory, the user ID locations, the Calibration Word and the Configuration Word.

FIGURE 1-1: PIN DIAGRAM – 28-PIN QFN (MCP19214)
### TABLE 1-1: PIN DESCRIPTIONS IN PROGRAM/VERIFY MODE: MCP19214

<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>GPB5</td>
<td>ICSPCLK</td>
<td>I</td>
<td></td>
<td>Clock Input – Schmitt Trigger Input</td>
</tr>
<tr>
<td>GPB4</td>
<td>ICSPDAT</td>
<td>I/O</td>
<td></td>
<td>Data Input/Output – Schmitt Trigger Input</td>
</tr>
<tr>
<td>MCLR</td>
<td>Program/Verify mode</td>
<td>P(1)</td>
<td></td>
<td>Program Mode Select</td>
</tr>
<tr>
<td>VIN</td>
<td>VIN</td>
<td>P</td>
<td></td>
<td>Device Power Supply Input</td>
</tr>
<tr>
<td>VDD</td>
<td>VDD</td>
<td>P</td>
<td></td>
<td>Power Supply Output</td>
</tr>
<tr>
<td>EP</td>
<td>VSS</td>
<td>P</td>
<td></td>
<td>Ground</td>
</tr>
</tbody>
</table>

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

**Note 1:** In the MCP19214, the programming high voltage is internally generated. To activate the Program/Verify mode, a voltage of $V_{IH}$ and a current of $I_{IH}$ (see Table 6-1) need to be applied to the MCLR input.
FIGURE 1-2: PIN DIAGRAM – 32-PIN QFN (MCP19215)

TABLE 1-2: PIN DESCRIPTIONS IN PROGRAM/VERIFY MODE: MCP19215

<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>GPB5</td>
<td>ICSPCLK</td>
<td></td>
<td>I</td>
<td>Clock Input – Schmitt Trigger Input</td>
</tr>
<tr>
<td>GPB4</td>
<td>ICSPDAT</td>
<td></td>
<td>I/O</td>
<td>Data Input/Output – Schmitt Trigger Input</td>
</tr>
<tr>
<td>MCLR</td>
<td>Program/Verify mode</td>
<td>p(1)</td>
<td>P</td>
<td>Program Mode Select</td>
</tr>
<tr>
<td>V_IN</td>
<td>V_IN</td>
<td></td>
<td>P</td>
<td>Device Power Supply Input</td>
</tr>
<tr>
<td>V_DD</td>
<td>V_DD</td>
<td></td>
<td>P</td>
<td>Power Supply Output</td>
</tr>
<tr>
<td>EP</td>
<td>V_SS</td>
<td></td>
<td>P</td>
<td>Ground</td>
</tr>
</tbody>
</table>

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

Note 1: In the MCP19215, the programming high voltage is internally generated. To activate the Program/Verify mode, a voltage of $V_{IH}$ and a current of $I_{IH}$ (see Table 6-1) need to be applied to the MCLR input.

© 2016 Microchip Technology Inc.
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 Program Counter (PC) will increment from 0x0000 to 0x1FFF and wrap around to 0x0000. If the PC is between 0x2000 and 0x3FFF, it will 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 to reenter the Program/Verify mode, as described in Section 3.0 “Program/Verify Mode”.

For all of the devices covered in this document, the configuration memory space, 0x2000 to 0x208F, is physically implemented. However, only locations, 0x2000 to 0x2003, 0x2007 and 0x2080 to 0x208C, are available. Other locations are reserved. The program memory mapping is depicted in Figure 2-1.

2.2 User ID Locations

A user may store identification information (user ID) in four designated locations. The user ID locations are mapped in 0x2000 to 0x2003. It is recommended that the user uses only the seven Least Significant bits (LSBs) 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 the user ID information.

The 14 bits may be programmed, but only the seven LSbs are read and displayed by the MPLAB® Integrated Development Environment (IDE).

2.3 Calibration Word

For all of the devices covered in this document, Calibration Words are included to allow storing the trim values for various analog peripherals (i.e., INTOSC module) at final test. These values are stored in Calibration Words, 0x2080, 0x2081, 0x2082, 0x2083, 0x2084 and 0x2085. See the applicable device data sheet for more information.

The Calibration Words do not necessarily participate in the erase operation unless a specific procedure is executed. Therefore, the device can be erased without affecting the Calibration Words. This simplifies the erase procedure, since these values do not need to be read and restored after the device is erased.
FIGURE 2-1: MCP19214 AND MCP19215 PROGRAM MEMORY MAPPING

<table>
<thead>
<tr>
<th>2000</th>
<th>User ID Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>2001</td>
<td>User ID Location</td>
</tr>
<tr>
<td>2002</td>
<td>User ID Location</td>
</tr>
<tr>
<td>2003</td>
<td>User ID Location</td>
</tr>
<tr>
<td>2004</td>
<td>ICD Instruction</td>
</tr>
<tr>
<td>2005</td>
<td>Revision ID</td>
</tr>
<tr>
<td>2006</td>
<td>Device ID</td>
</tr>
<tr>
<td>2007</td>
<td>Configuration Words</td>
</tr>
<tr>
<td>2008-207F</td>
<td>Reserved</td>
</tr>
<tr>
<td>2080-208F</td>
<td>Calibration Words</td>
</tr>
</tbody>
</table>

- Implemented
- Unimplemented
- Maps to 2000-20FF

Program Memory

- 8 kW Implemented

Configuration Memory

- 1FF
- 2000
- 208F
- 2090
- 2100
- 3FF
- Unimplemented
3.0 PROGRAM/VERIFY MODE

Two methods are available to enter Program/Verify mode. "TEST_EN-first" is entered by holding ICSPDAT and ICSPCLK low while raising the MCLR pin from $V_{IL}$ (high voltage), then applying $V_{DD}$ and data. This method can be used for any Configuration Word selection and must be used if the internal MCLR option is selected ($MCLRE = 0$). The TEST_EN-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, "$V_{DD}$-first", is entered by applying $V_{DD}$, holding ICSPDAT and ICSPCLK low, then raising the MCLR pin from $V_{IL}$ to $V_{IHH}$ (high voltage), followed by data. This method can be used for any Configuration Word selection, except when the internal MCLR option is selected ($MCLRE = 0$). This technique is useful when programming the device with $V_{DD}$ already applied, for it is not necessary to disconnect the $V_{DD}$ to enter the Program/Verify mode. See the timing diagram in Figure 3-2.

Once in Program/Verify mode, the program memory and configuration memory can be accessed, and programmed in serial fashion. ICSPDAT and ICSPCLK are Schmitt Trigger inputs in this mode.

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 $V_{IL}$). Therefore, all I/Os are in the Reset state (high-impedance inputs) and the PC is cleared.

To prevent a device configured with internal MCLR from executing after exiting Program/Verify mode, $V_{DD}$ needs to power-down before TEST_EN. See Figure 3-3 for the timing diagram.

The MCP19214/5 devices’ $V_{DD}$ is internally generated by applying voltage to the $V_{IN}$ pin. See Table 6-1 for the appropriate range for $V_{IN}$. To remove $V_{DD}$, $V_{IN}$ must be removed.

3.1 Program/Erase Algorithms

The MCP19214/5 program memory may be written in two ways. The fastest method writes eight words at a time. However, one-word writes are also supported. The eight-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 calibration 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 EIGHT-WORD PROGRAMMING
The MCP19214/5 program memory can be written four words at a time using the four-word algorithm. Configuration memory (addresses > 0x2000) and non-aligned (Addresses Modulo 8 not equal to zero) starting addresses must use the one-word programming algorithm.

This algorithm writes eight sequential addresses in program memory. The eight addresses must point to an eight-word block with an Address Modulo 8 of 0, 1, 2, 3, 4, 5, 6 and 7. For example, Programming Addresses 4 through 7 can be programmed together. Programming Addresses 2 through 9 will create an unexpected result.

The sequence for programming eight words of program memory at a time is:

1. Load a word at the current program memory address using the Load Data For Program Memory command. This location must be Address Modulo 8 equal to 0.
2. Issue an Increment Address command to point to the next address in the block.
3. Load a word at the current program memory address using the Load Data for Program Memory command.
4. Repeat Steps 2 and 3 seven times to load all 8 words at consecutive addresses.
5. Issue a Begin Programming command externally timed.
8. Wait T DIS.
9. Issue an Increment Address command to point to the start of the next block of addresses.
10. Repeat Steps 1 through 9 as required to write the desired range of program memory.

See Figure 3-13 for more information.

3.1.2 ERASE ALGORITHMS
The MCP19214/5 devices will erase different memory locations depending on the PC and CP. The following sequences can be used to erase noted memory locations. To erase the program memory and Configuration Word (0x2007), the following sequence must be performed. Note the Calibration Words (0x2080 to 0x208F) will not be erased.

1. Do a Bulk Erase Program Memory command.
2. Wait T ERA 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 (0x2080 to 0x208F) will not be erased.

1. Perform Load Configuration with dummy data to point the PC to 0x2000.
2. Perform a Bulk Erase Program Memory command.
3. Wait T ERA to complete erase.

3.1.3 SERIAL PROGRAM/VERIFY OPERATION
The ICSPCLK pin is used as a clock input and the ICSPDAT pin is used for entering command bits, and for 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 µs 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 µs delay is also specified between consecutive commands, except for the End Programming command, which requires a 100 µs (T DIS).

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 decoding of commands and reversal of data pin configuration, a time separation of at least 1 µs (T DL1) is required between a command and a data word.

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

The Load Configuration command is used to access the Configuration Word (0x2007), User ID (0x2000-0x2003) and Calibration Words (0x2080 to 0x208F). This command sets the 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 using 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 the Stop 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).

### TABLE 3-1: COMMAND MAPPING FOR MCP19214/5

<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</td>
<td>0, data (14), 0</td>
</tr>
<tr>
<td>Load Data for Program Memory</td>
<td>x x 0 0 1 0</td>
<td>0, data (14), 0</td>
</tr>
<tr>
<td>Read Data from Program Memory</td>
<td>x x 0 1 0 0</td>
<td>0, data (14), 0</td>
</tr>
<tr>
<td>Increment Address</td>
<td>x x 0 1 1 0</td>
<td></td>
</tr>
<tr>
<td>Begin Programming</td>
<td>x 1 1 0 0 0</td>
<td>Externally Timed</td>
</tr>
<tr>
<td>End Programming</td>
<td>x 0 1 0 1 0</td>
<td></td>
</tr>
<tr>
<td>Bulk Erase Program Memory</td>
<td>x x 1 0 0 1</td>
<td>Internally Timed</td>
</tr>
<tr>
<td>Row Erase Program Memory</td>
<td>x 1 0 0 0 1</td>
<td>Internally Timed</td>
</tr>
</tbody>
</table>

### FIGURE 3-4: Load Configuration COMMAND

[Diagram showing the timing of the Load Configuration command, including ICSPCLK and ICSPDAT signals with timing labels such as TSET1, THLD1, TDLV1, TDLV3, etc.]
3.1.3.2 Load Data for Program Memory

After receiving the Load Data for Program Memory command, the chip will load in a 14-bit "data word" when 16 cycles are applied, as described in Section 3.1.3.1 "Load Configuration". A timing diagram of this command is shown in Figure 3-5.

**FIGURE 3-5: Load Data for Program Memory COMMAND**

3.1.3.3 Read Data From Program Memory

After receiving the Read Data from Program Memory 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.

A timing diagram of this command is shown in Figure 3-6.

**FIGURE 3-6: Read Data from Program Memory COMMAND**
3.1.3.4 Increment Address

The PC is incremented when the **Increment Address** command is received. A timing diagram of this command is shown in **Figure 3-7**. Incrementing past 0x07FF in program memory rolls the Program Counter to '0'. Incrementing past 203Fh in test memory returns the Program Counter to 2000h.

**FIGURE 3-7:** Increment Address COMMAND (PROGRAM/VERIFY)

![](image)

3.1.3.5 Begin Programming (Externally Timed)

A **Load** command must be given before every **Begin Programming** command. Programming of the appropriate memory (program memory, configuration or calibration memory) will begin after this command is received and decoded. Programming requires (TPROG) time and is terminated using an **End Programming** command. A timing diagram for this command is shown in **Figure 3-8**.

The addressed locations are not erased before programming.

**FIGURE 3-8:** Begin Programming (EXTERNALLY TIMED)

![](image)
3.1.3.6 End Programming

After this command is performed, the write procedure will stop. A timing diagram of this command is shown in Figure 3-9.

FIGURE 3-9: End Programming (SERIAL PROGRAM/VERIFY)

3.1.3.7 Bulk Erase Program Memory

After this command is performed, the entire program memory and Configuration Word (0x2007) are erased. The user ID and calibration memory may also be erased, depending on the value of the PC. See Section 3.1.2 “Erase Algorithms” for erase sequences. A timing diagram for this command is shown in Figure 3-10.
3.1.3.8 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 the configuration memory (> 0x2000), the command is ignored.

To perform a Row Erase Program Memory command, the following sequence must be performed:
1. Execute a Row Erase Program Memory command.
2. Wait TERA to complete a row erase.

**FIGURE 3-11: Row Erase Program Memory COMMAND**

**FIGURE 3-12: 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:** If the device is code-protected or must be completely erased, then bulk erase the device per Figure 3-15.
FIGURE 3-13: EIGHT-WORD PROGRAMMING FLOWCHART

Start

Bulk Erase Program Memory (1,2)

Eight-Word Program Cycle

Increment Address Command

No

All Locations Done?

Yes

Program User ID/Config. Bits

Done

Program Cycle

Load Data for Program Memory

Increment Address Command

Load Data for Program Memory

Increment Address Command

Load Data for Program Memory

Increment Address Command

Load Data for Program Memory

Increment Address Command

Load Data for Program Memory

Increment Address Command

Begin Programming Command (externally timed)

Wait TPROG

End Programming

Wait TDIS

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

2: If the device is code-protected or must be completely erased, then bulk erase the device per Figure 3-15.
FIGURE 3-14: PROGRAM FLOWCHART – 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

Program Cycle

Load Data for Program Memory

Begin Programming Command (externally timed)

Wait T_PROG

End Programming

Wait T_DIS

No

Increment Address Command

Increment Address Command

Increment Address Command

One-Word Program Cycle (Config. Bits)

Read Data from Program Memory Command

Data Correct?

Yes

Done

No

Report Programming Failure

Report Programming Failure
FIGURE 3-15: PROGRAM FLOWCHART – ERASE FLASH DEVICES

Start

Load Configuration

Bulk Erase⁽¹⁾ Program Memory

Done

Note 1: See Section 3.1.3.7 “Bulk Erase Program Memory” for more information on the bulk erase procedure.
4.0 CONFIGURATION WORD

The MCP19214/5 devices have several Configuration bits. These bits can be programmed (read ‘0’) or left unchanged (read ‘1’) to select various device configurations.

REGISTER 4-1: CONFIG: CONFIGURATION WORD (ADDRESS: 2007h)

<table>
<thead>
<tr>
<th>R/P-1</th>
<th>U-1</th>
<th>R/P-1</th>
<th>R/P-1</th>
<th>U-1</th>
<th>R/P-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBGEN</td>
<td></td>
<td>WRT1</td>
<td>WRT0</td>
<td></td>
<td>BOREN</td>
</tr>
</tbody>
</table>

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

bit 13 DBGEN: ICD Debug bit
 1 = ICD Debug mode is disabled
 0 = ICD Debug mode is enabled

bit 12 Unimplemented: Read as ‘1’

bit 11-10 WRT<1:0>: Flash Program Memory Self-Write Enable bits
 11 = Write protection off
 10 = 000h to 3FFh are write-protected, 400h to FFFh may be modified by PMCON1 control
 01 = 000h to 7FFh are write-protected, 800h to FFFh may be modified by PMCON1 control
 00 = 000h to FFFh are write-protected, entire program memory is write-protected

bit 9 Unimplemented: Read as ‘1’

bit 8 BOREN: Brown-out Reset Enable bits
 1 = BOR is disabled during Sleep and enabled during operation
 0 = BOR is disabled

bit 7 Unimplemented: Read as ‘1’

bit 6 CP: Code Protection bit
 1 = Program memory is not code-protected
 0 = Program memory is external read and write-protected

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

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

bit 3 WDTE: Watchdog Timer Enable bit
 1 = WDT is enabled
 0 = WDT is disabled

bit 2-0 Unimplemented: Read as ‘1’

Note 1: Bit is reserved and not controlled by user.
4.1 Device ID Word

The Device ID Word for the MCP19214/5 is loaded at 2006h. The Device ID Word is a 14-bit word and is stored separately from the Revision ID Word. This location cannot be erased.

<table>
<thead>
<tr>
<th>Device</th>
<th>Device ID Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCP19214</td>
<td>11 0000 0000 1000</td>
</tr>
<tr>
<td>MCP19215</td>
<td>11 0000 0000 1001</td>
</tr>
</tbody>
</table>

4.2 Revision ID Word

The Revision ID Word for the MCP19214/5 is loaded at 2005h. This location cannot be erased.

<table>
<thead>
<tr>
<th>Device</th>
<th>Revision ID Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCP19214</td>
<td>----</td>
</tr>
<tr>
<td>MCP19215</td>
<td>----</td>
</tr>
</tbody>
</table>
5.0 CODE PROTECTION

For MCP19214/5, 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.

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

5.1 Disabling Code Protection

It is recommended to use the procedure in Figure 3-15 to disable the code protection of the device. This sequence will erase the program memory, the Configuration Word (0x2007) and the user ID locations (0x2000-0x2003). The Calibration Words (0x2080 to 0x2083) will not be erased.

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 it. If Configuration Word information is 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.

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

**EXAMPLE 5-1:** CHECKSUM COMPUTED WITH PROGRAM CODE PROTECTION DISABLED (CP = 1), MCP19214 AND MCP19215 BLANK DEVICES

<table>
<thead>
<tr>
<th>Sum of Memory addresses 000h-1FFH</th>
<th>E000h(1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Word</td>
<td>3FFFh(2)</td>
</tr>
<tr>
<td>Configuration Word mask</td>
<td>2D78h(3)</td>
</tr>
<tr>
<td>Checksum</td>
<td>E000h + (3FFFh and 2D78h)(4)</td>
</tr>
<tr>
<td></td>
<td>E000h + 2D78h</td>
</tr>
<tr>
<td></td>
<td>D78h</td>
</tr>
</tbody>
</table>

**Note 1:** This value is obtained by taking the total number of program memory locations (0x0000h to 0x0FFFh, which is 0x2000h) and multiplying it by the blank memory value of 0x3FFF to get the sum of 7FF E000h. Then, truncate to 16 bits, thus having a final value of E000h.

**Note 2:** This value is obtained by making all bits of the Configuration Word a ‘1’, then converting it to hex, thus having a value of 3FFFh.

**Note 3:** This value is obtained by making all used bits of the Configuration Word a ‘1’, then converting it to hex, thus having a value of 2D78h.

**Note 4:** This value is obtained by ANDing the Configuration Word value with the Configuration Word mask value and adding it to the sum of memory addresses (3FFFh and 2D78) + E000h = 10D78h. Then, truncate to 16 bits, thus having a final value of D78h.

5.3 Checksum Computation

The checksum is calculated by two different methods dependent on the setting of the CP Configuration bit.

5.3.1 PROGRAM CODE PROTECTION DISABLED

With the program code protection disabled, the checksum is computed by reading the contents of the program memory locations and adding up the program memory data, starting at address 0x0000h, up to the maximum user-addressable location. Any Carry bit exceeding 16 bits is ignored. Additionally, the relevant bits of the Configuration Words are added to the checksum. All unimplemented Configuration bits are masked to ‘0’.
5.3.2 PROGRAM 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 the user IDs is summed with the Configuration Word (all unimplemented Configuration bits are masked to ‘0’).

EXAMPLE 5-2: CHECKSUM COMPUTED WITH PROGRAM CODE PROTECTION ENABLED (CP = 0), MCP19214 AND MCP19215 BLANK DEVICES

<table>
<thead>
<tr>
<th>Configuration Word</th>
<th>3FBFh(1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Word mask</td>
<td>2D38h(2)</td>
</tr>
<tr>
<td>User ID (2000h)</td>
<td>0006h(3)</td>
</tr>
<tr>
<td>User ID (2001h)</td>
<td>0007h(3)</td>
</tr>
<tr>
<td>User ID (2002h)</td>
<td>0001h(3)</td>
</tr>
<tr>
<td>User ID (2003h)</td>
<td>0002h(3)</td>
</tr>
</tbody>
</table>

Sum of User IDs = (0006h and 000Fh) << 12 + (0007h and 000Fh) << 8 + (0001h and 000Fh) << 4 + (0002h and 000Fh)(4)
= 6000h + 0700h + 0010h + 0002h
= 6712h

Checksum = (3FBFh and 2D38h) + Sum of User IDs(5)
= 2D38h + 6712h
= 944Ah

Note 1: This value is obtained by making all bits of the Configuration Word a ‘1’, but the code protection bit is ‘0’ (thus, enabled), then converting it to a hex, thus having a value of 3FBFh.

2: This value is obtained by making all used bits of the Configuration Word a ‘1’, but the code protection bit is ‘0’ (thus, enabled), then converting it to a hex, thus having a value of 2D38h.

3: These values are picked at random for this example; they could be any 16-bit value.

4: In order to calculate the sum of user IDs, take the 16-bit value of the first user ID location (0006h), AND the address to (000Fh), thus masking the MSB. This gives you the value of 0006h, then shift left 12 bits, giving you 6000h. Do the same procedure for the 16-bit value of the second user ID location (0007h), except shift left eight bits. Also, do the same for the third user ID location (0001h), except shift left four bits. For the fourth user ID location, do not shift. Finally, add up all four user ID values to get the final sum of user IDs of 6712h.

5: This value is obtained by ANDing the Configuration Word value with the Configuration Word mask value and adding it to the sum of user IDs: (3FBFh and 2D38h) + (6712h) = 944Ah.
### 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>Conditions/Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Operating Temperature -40°C ≤ TA ≤ +85°C</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Operating Voltage 4.5V ≤ VDD ≤ 5.5V</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sym.</th>
<th>Characteristics</th>
<th>Min.</th>
<th>Typ.</th>
<th>Max.</th>
<th>Units</th>
<th>Conditions/Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>General</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VIN</td>
<td>V_IN Level for Read/Write Operations, Program and Data Memory</td>
<td>4.5</td>
<td>—</td>
<td>42</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>VITH</td>
<td>V_IN Level for Bulk Erase Operations, Program and Data Memory</td>
<td>4.5</td>
<td>—</td>
<td>42</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>VIHH</td>
<td>High Voltage on MCLR for Program/Verify Mode Entry</td>
<td>VDD + 3.5</td>
<td>—</td>
<td>13</td>
<td>V</td>
<td>VDD regulated internally to 5V</td>
</tr>
<tr>
<td>IHH</td>
<td>MCLR Current During Programming</td>
<td>—</td>
<td>300</td>
<td>1000</td>
<td>µA</td>
<td></td>
</tr>
<tr>
<td>TVHR</td>
<td>MCLR Rise Time (VSS to VHH) for Program/Verify Mode Entry</td>
<td>—</td>
<td>—</td>
<td>1.0</td>
<td>µs</td>
<td></td>
</tr>
<tr>
<td>TPPDP</td>
<td>Hold Time after TEST_EN Changes</td>
<td>5</td>
<td>—</td>
<td>—</td>
<td>µs</td>
<td></td>
</tr>
<tr>
<td>VIL1</td>
<td>(ICSPCLK, ICSPDAT) Input Low Level</td>
<td>0.2 VDD</td>
<td>—</td>
<td>—</td>
<td>V</td>
<td>Schmitt Trigger input</td>
</tr>
<tr>
<td>VIL1</td>
<td>(ICSPCLK, ICSPDAT) Setup Time before MCLR (Program/Verify mode selection pattern setup time)</td>
<td>100</td>
<td>—</td>
<td>—</td>
<td>ns</td>
<td></td>
</tr>
<tr>
<td>THLD</td>
<td>Hold Time after VDD Changes</td>
<td>5</td>
<td>—</td>
<td>—</td>
<td>µs</td>
<td></td>
</tr>
<tr>
<td><strong>Serial Program/Verify</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TSET</td>
<td>Data in Setup Time before Clock↓</td>
<td>100</td>
<td>—</td>
<td>—</td>
<td>ns</td>
<td></td>
</tr>
<tr>
<td>T召HLD</td>
<td>Data in Hold Time after Clock↓</td>
<td>100</td>
<td>—</td>
<td>—</td>
<td>ns</td>
<td></td>
</tr>
<tr>
<td>TDL1</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>
<td>—</td>
<td>µs</td>
<td></td>
</tr>
<tr>
<td>TDL2</td>
<td>Delay between Clock↓ to Clock↑ of Next Command or Data</td>
<td>1.0</td>
<td>—</td>
<td>—</td>
<td>µs</td>
<td></td>
</tr>
<tr>
<td>TDL3</td>
<td>Clock↓ to Data Out Valid (during a Read Data command)</td>
<td>—</td>
<td>—</td>
<td>80</td>
<td>ns</td>
<td></td>
</tr>
<tr>
<td>TERA</td>
<td>Erase Cycle Time</td>
<td>—</td>
<td>5</td>
<td>6</td>
<td>ms</td>
<td></td>
</tr>
<tr>
<td>TRPG</td>
<td>Programming Cycle Time</td>
<td>3</td>
<td>—</td>
<td>—</td>
<td>ms</td>
<td>+10°C ≤ TA ≤ +40°C</td>
</tr>
<tr>
<td>TDIS</td>
<td>Time Delay from Program to Compare (HV discharge time)</td>
<td>100</td>
<td>—</td>
<td>—</td>
<td>µs</td>
<td></td>
</tr>
</tbody>
</table>
APPENDIX A: REVISION HISTORY

Revision A (December 2016)

• Original 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 unless otherwise stated.

Trademarks

The Microchip name and logo, the Microchip logo, AnyRate, AVR, AVR logo, AVR Freaks, BeaconThings, BitCloud, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, Heldo, JukeBlox, KEELOO logo, KEELOO logo, Kleer, LANCheck, LINK MD, maXSylus, maXTouch, MediaLB, megaAVR, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, Prochip Designer, QTouch, RightTouch, SAM-BA, SpyNIC, SST, SST Logo, SuperFlash, tinyAVR, UNI/O, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

ClockWorks, The Embedded Control Solutions Company, EtherSynch, Hyper Speed Control, HyperLight Load, IntelliMOS, mTouch, Precision Edge, and Quiet-Wire are registered trademarks of Microchip Technology Incorporated in the U.S.A.


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 trademark 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.

© 2016, Microchip Technology Incorporated, All Rights Reserved.