



# **User Flash Memory (ALTUFM) Megafunction**

---

## **User Guide**



101 Innovation Drive  
San Jose, CA 95134  
[www.altera.com](http://www.altera.com)

UG-040105-3.1



Feedback



Subscribe

© 2012 Altera Corporation. All rights reserved. ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS and STRATIX words and logos are trademarks of Altera Corporation and registered in the U.S. Patent and Trademark Office and in other countries. All other words and logos identified as trademarks or service marks are the property of their respective holders as described at [www.altera.com/common/legal.html](http://www.altera.com/common/legal.html). Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Altera assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.



## Chapter 1. About this Megafunction

Features .....	1-1
Device Support .....	1-1
Resource Utilization and Performance .....	1-1

## Chapter 2. Parameter Settings

MegaWizard Parameter Settings .....	2-1
Command Line Interface Parameters .....	2-10

## Chapter 3. Functional Description

User Flash Memory .....	3-1
Memory Organization Map .....	3-1
Using & Accessing UFM Storage .....	3-2
UFM Operating Modes .....	3-3
Serial Peripheral Interface .....	3-4
Parallel Interface .....	3-4
None (Altera Serial Interface) .....	3-5
Inter-Integrated Circuit Interface .....	3-5
Common Applications .....	3-6
Design Example: User Flash Memory with SPI Interface .....	3-6
Generate the User Flash Memory .....	3-6
Implement the User Flash Memory .....	3-7
Functional Results—Simulate the User Flash Memory in the Quartus II Software .....	3-8
Functional Results—Simulate the User Flash Memory in ModelSim-Altera Software .....	3-9
ALTUFM_PARALLEL Megafunction Ports .....	3-10
Input Ports .....	3-10
Output Ports .....	3-10
ALTUFM_SPI Megafunction Ports .....	3-11
Input Ports .....	3-11
Output Ports .....	3-11
ALTUFM_I2C Megafunction Ports .....	3-11
Input Ports .....	3-11
Output Ports .....	3-12
Bidirectional Ports .....	3-12
ALTUFM_NONE Megafunction Ports .....	3-12
Input Ports .....	3-12
Output Ports .....	3-13

## Additional Information

Document Revision History .....	Info-1
How to Contact Altera .....	Info-1
Typographic Conventions .....	Info-1



The ALTUFM megafunction provides interface logic for a subset of parallel interface , serial peripheral interface (SPI), inter-integrated circuit (I<sup>2</sup>C,) and the built-in dedicated user flash memory (UFM) serial interface.

## Features

The ALTUFM megafunction provides the following additional features:

- Up to 8K bits for non-volatile storage
- Two sectors for partitioned sector erase
- Interface protocols: parallel, SPI, I2C, and none (use dedicated UFM)
- Memory initialization using Memory Initialization File or HEX File
- Built-in oscillator that provides oscillator frequency for the user flash memory
- Program, erase, and busy signals
- Easy Instantiation with the MegaWizard™ Plug-In Manager

## Device Support

The ALTUFM megafunction supports the MAX® II and Max V devices.

## Resource Utilization and Performance

The ALTUFM megafunction is only available for MAX II and MAX V devices. Resource usage is reported with different interface options. Configuration mode settings described in the following tables are available on page 3 of the MegaWizard Plug-In Manager.

Table 1–1 lists the resource usage by the ALTUFM \_SPI megafunction.

**Table 1–1. ALTUFM \_SPI Resource Usage**

Access Mode <sup>(1)</sup>	Optimization <sup>(2)</sup>	Configuration Mode <sup>(3)</sup>	Logic Element	UFM Blocks
Read/write	Balanced	Base mode	147	1
Read only	Balanced	Base mode	72	1
Read/write	Balanced	Extended mode	134	1
Read only	Balanced	Extended mode	40	1

**Notes to Table 1–1:**

- (1) Choose the access mode option to configure the UFM to the required mode of operation. This option is available on page 3 of the MegaWizard Plug-In Manager.
- (2) Choose a design implementation that balances high performance with minimal logic usage. The balanced optimization logic option is set in Analysis and Synthesis settings (Assignments menu).
- (3) Choose the configure mode to set the mode of access with SPI interface to the UFM block. Set this option to **Base Mode** or **Extended Mode** depending on the widths of the address and data buses.

Table 1–2 lists the resource usage by the ALTUFM\_PARALLEL megafunction.

**Table 1–2. ALTUFM\_PARALLEL Resource Usage**

Access Mode (1)	Optimization (2)	Address Width	Data Width	Logic Element	UFM Blocks
Read/write	Balanced	9	16	88	1
Read only	Balanced	9	16	69	1

**Notes to Table 1–2:**

- (1) Choose the access mode option to configure the UFM to the required mode of operation.
- (2) Choose a design implementation that balances high performance with minimal logic usage. The balanced optimization logic option is set in Analysis and Synthesis settings (Assignments menu).

Table 1–3 lists the resource usage by the ALTUFM\_NONE megafunction.

**Table 1–3. ALTUFM\_NONE Resource Usage**

Logic Elements	Optimization (1)	UFM Blocks
3	Balanced	1

**Note to Table 1–3:**

- (1) Choose a design implementation that balances high performance with minimal logic usage. The balanced optimization logic option is set in Analysis and Synthesis settings (Assignments menu).

Table 1–4 through Table 1–8 lists the resource usage by the ALTUFM\_I2C megafunction in different access modes. Choose the Access Mode option to configure the UFM to the required mode of operation.



Memory size is available only for the I2C interface where the size of the memory to be protected is specified. This option is valid only when the access mode is set to **Read and Write**. Set this option on Page 4 of the MegaWizard Plug-In Manager.

Table 1–4 lists the resource usage by the ALTUFM\_I2C megafunction in **Read only** access mode. (Optimization effort for the I<sup>2</sup>C interface is set to balanced for all listed combinations).

**Table 1–4. ALTUFM\_I2C Resource Usage**

Access Mode	Memory Size	Logic Element	UFM Blocks
Read only	1K	117	1
Read only	2K	117	1

Table 1–5 lists the resource usage by the ALTUFM\_I2C megafunction in **Read and write** access mode with the **Memory address erase** option selected in the MegaWizard Plug-In Manager.

**Table 1–5. ALTUFM\_I2C Resource Usage**

Access Mode	Memory Size	Write Protection			UFM Blocks
		No Write Protection	Full Memory Write Protected	Half Memory Write Protected	
Read/write	1K	154 LE	154 LE	156 LE	1

**Table 1–5. ALTUFM\_I2C Resource Usage**

Access Mode	Memory Size	Write Protection			UFM Blocks
		No Write Protection	Full Memory Write Protected	Half Memory Write Protected	
Read/write	2K	155 LE	155 LE	157 LE	1
Read/write	4K	119 LE	119 LE	120 LE	1

Table 1–6 lists the resource usage by the ALTUFM\_I2C megafunction for **Read and write** access mode with the **Device address erase** option selected in the MegaWizard Plug-In Manager.

**Table 1–6. ALTUFM\_I2C Resource Usage**

Access Mode	Memory Size	Write Protection			UFM Blocks
		No Write Protection	Full Memory Write Protected	Half Memory Write Protected	
Read/write	1K	141 LE	141 LE	142 LE	1
Read/write	2K	164 LE	164 LE	166 LE	1

Table 1–7 lists the resource usage by the ALTUFM\_I2C megafunction for **Read and write** access mode with the **A2 erase** option selected in the MegaWizard Plug-In Manager.

**Table 1–7. ALTUFM\_I2C Resource Usage**

Access Mode	Memory Size	Write Protection			UFM Blocks
		No Write Protection	Full Memory Write Protected	Half Memory Write Protected	
Read/write	1K	151 LE	151 LE	153 LE	1
Read/write	2K	152 LE	152 LE	153 LE	1

Table 1–8 lists the resource usage by the ALTUFM\_I2C megafunction for **Read and write** access mode with the **No erase** option selected in the MegaWizard Plug-In Manager.

**Table 1–8. ALTUFM\_I2C Resource Usage**

Access Mode	Memory Size	Write Protection			UFM Blocks
		No Write Protection	Full Memory Write Protected	Half Memory Write Protected	
Read/write	1K	144 LE	144 LE	146 LE	1
Read/write	2K	145 LE	145 LE	147 LE	1

The MegaWizard Plug-In Manager reports approximate resource utilization based on user specification and parameters, available in the lower left corner of the MegaWizard Plug-In Manager screen.



This section describes the parameter settings for the ALTUFM megafunction. You can parameterize the megafunction using the MegaWizard™ Plug-In Manager or the command-line interface (CLI). Altera recommends that you configure the megafunctions using the MegaWizard Plug-In Manager.



This user guide assumes that you are familiar with megafunctions and how to create them. If you are unfamiliar with Altera megafunctions, refer to the [Introduction to Megafunctions User Guide](#).

### MegaWizard Parameter Settings

Table 2–3 lists the parameter settings for the ALTUFM\_PARALLEL megafunction.

**Table 2–1. ALTUFM\_PARALLEL Parameter Settings (Part 1 of 3)**

MegaWizard Plug-in Manager Page	Configuration Setting	Description
1	Which action do you want to perform?	You can select from the following options: <b>Create a new custom megafunction variation</b> , <b>Edit an existing custom megafunction variation</b> , or <b>Copy an existing custom megafunction variation</b> .
2a	Select a megafunction from the list below	Select <b>ALTUFM_PARALLEL</b> from the <b>Memory Compiler</b> category.
	Which device family will you be using?	Specify the device family that you want to use.
	Which type of output file do you want to create?	You can choose AHDL(. <b>tdf</b> ), VHDL(. <b>vhd</b> ), or Verilog HDL (. <b>v</b> ) as the output file type.
	What name do you want for the output file?	Specify the name of the output file.
	Return to this page for another create operation	Turn on this option if you want to return to this page to create multiple megafunctions.

Table 2–1. ALTUFM\_PARALLEL Parameter Settings (Part 2 of 3)

MegaWizard Plug-in Manager Page	Configuration Setting	Description
3	Currently selected device family	Specifies the device family you chose on page 2a.
	Match project/default	Turn on this option to ensure that the device selected matches the device family that is chosen in the previous page.
	Read and write	Turn on this option if you want to enable the read and write access mode.
	Read only	Turn on this option if you want to enable the read only mode.
	'addr' width	Select the width for the address bus. The maximum size of the address bus can be 9.
	'datain' width	Select the width for the data bus. The maximum size of the data bus can be 16.
	Use 'osc' output port	Turn on this option to route the oscillator frequency to an external oscillator port.
	Use 'oscena' input port	Turn on this option to enable the oscillator enable port.
45	Memory content initialization	<ul style="list-style-type: none"> <li>■ Select <b>Initialize blank memory</b> if you do not want to specify any initialization file. Select <b>Initialize from hex or mif file</b> to specify the initialization file. Type the file name or browse for the required file.</li> <li>■ In the Quartus II software, the memory content values from your <b>.hex</b> or <b>.mif</b> are hard-coded into your ALTUFM megafunction variation file when you run the ALTUFM_I2C, ALTUFM_NONE, ALTUFM_PARALLEL, or ALTUFM_SPI MegaWizard™ Plug-In Managers. If you change the contents of your <b>.hex</b> or <b>.mif</b> after running the MegaWizard Plug-In Manager, these updates will not be reflected in simulation. This may cause a mismatch between simulation and device behavior because compilation and program file generation in the Quartus II software use the current <b>.hex</b> or <b>.mif</b> contents instead of the hard-coded values. To avoid this mismatch, re-run the MegaWizard Plug-In Manager for your ALTUFM megafunction whenever you update your <b>.hex</b> or <b>.mif</b>.</li> </ul>
	Oscillator frequency	Specify the oscillator frequency for the user flash memory. This parameter is used for simulation purposes only. The values are <b>5.56MHz</b> and <b>3.33MHz</b> . If omitted, the default is <b>5.56MHz</b> .
	Erase time	Specify the erase time in unit of ns. Simulation erase time for the UFM block can only be from <b>1600 ns</b> to <b>999999 ns</b> .
	Program time	Specify the program time in unit ns. Simulation erase time for the UFM block can only be from <b>1600 ns</b> to <b>100000 ns</b> .

**Table 2–1. ALTUFM\_PARALLEL Parameter Settings (Part 3 of 3)**

MegaWizard Plug-in Manager Page	Configuration Setting	Description
5	Generate netlist	Turn on this option if you want to generate a netlist for your third-party EDA synthesis tool to estimate the timing and resource usage of the megafunction. If you turn on this option, a netlist file ( <b>_syn.v</b> ) is generated. This file is a representation of the customized logic used in the Quartus II software and provides the connectivity of the architectural elements in the megafunction but may not represent true functionality.
6	Summary Page	Specify the types of files to be generated. The Variation file ( <i>&lt;function name&gt;.v</i> ) contains wrapper code in the language you specified on page 2a and is automatically generated. Choose from the following types of files: <ul style="list-style-type: none"> <li>■ AHDL Include file (<i>&lt;function name&gt;.inc</i>)</li> <li>■ VHDL component declaration file (<i>&lt;function name&gt;.cmp</i>)</li> <li>■ Quartus II symbol file (<i>&lt;function name&gt;.bsf</i>)</li> <li>■ Instantiation template file (<i>&lt;function name&gt;_inst.v</i>)</li> <li>■ Verilog HDL black box file (<i>&lt;function name&gt;_bb.v</i>)</li> </ul> For more information about the wizard-generated files, refer to the <a href="#">Introduction to Megafunctions User Guide</a> .

Table 2–3 lists the parameter settings for the ALTUFM\_SPI megafunction.

**Table 2–2. ALTUFM\_SPI Parameter Settings (Part 1 of 3)**

MegaWizard Plug-in Manager Page	Configuration Setting	Description
1	Which action do you want to perform?	You can select from the following options: <b>Create a new custom megafunction variation</b> , <b>Edit an existing custom megafunction variation</b> , or <b>Copy an existing custom megafunction variation</b> .
2a	Select a megafunction from the list below	Select <b>ALTUFM_SPI</b> from the <b>Memory Compiler</b> category.
	Which device family will you be using?	Specify the device family that you want to use.
	Which type of output file do you want to create?	You can choose AHDL( <b>.tdf</b> ), VHDL( <b>.vhd</b> ), or Verilog HDL ( <b>.v</b> ) as the output file type.
	What name do you want for the output file?	Specify the name of the output file.
	Return to this page for another create operation	Turn on this option if you want to return to this page to create multiple megafunctions.

Table 2-2. ALTUFM\_SPI Parameter Settings (Part 2 of 3)

MegaWizard Plug-in Manager Page	Configuration Setting	Description
3	Currently selected device family	Specifies the device family you chose on page 2a.
	Match project/default	Turn on this option to ensure that the device selected matches the device family that is chosen in the previous page.
	Read and write	Turn on this option if you want to enable the read and write access mode.
	Read only	Turn on this option if you want to enable the read only mode.
	Configuration mode	Select <b>Base mode</b> to use 8-bit address and data. Select <b>Extended mode</b> to use 16-bit address and data.
	Use 'osc' output port	Turn on this option to route the oscillator frequency to an external oscillator port.
	Use 'oscena' input port	Turn on this option to enable the oscillator enable port.
45	Memory content initialization	<ul style="list-style-type: none"> <li>■ Select <b>Initialize blank memory</b> if you do not want to specify any initialization file. Select <b>Initialize from hex or mif file</b> to specify the initialization file. Type the file name or browse for the required file.</li> <li>■ In the Quartus II software, the memory content values from your <b>.hex</b> or <b>.mif</b> are hard-coded into your ALTUFM megafunction variation file when you run the ALTUFM_I2C, ALTUFM_NONE, ALTUFM_PARALLEL, or ALTUFM_SPI MegaWizard™ Plug-In Managers. If you change the contents of your <b>.hex</b> or <b>.mif</b> after running the MegaWizard Plug-In Manager, these updates will not be reflected in simulation. This may cause a mismatch between simulation and device behavior because compilation and program file generation in the Quartus II software use the current <b>.hex</b> or <b>.mif</b> contents instead of the hard-coded values. To avoid this mismatch, re-run the MegaWizard Plug-In Manager for your ALTUFM megafunction whenever you update your <b>.hex</b> or <b>.mif</b>.</li> </ul>
	Oscillator frequency	Specify the oscillator frequency for the user flash memory. This parameter is used for simulation purposes only. The values are <b>5.56MHz</b> and <b>3.33MHz</b> . If omitted, the default is <b>5.56MHz</b> .
	Erase time	Specify the erase time.
	Program time	Specify the program time.

**Table 2–2. ALTUFM\_SPI Parameter Settings (Part 3 of 3)**

MegaWizard Plug-in Manager Page	Configuration Setting	Description
5	Generate netlist	Turn on this option if you want to generate a netlist for your third-party EDA synthesis tool to estimate the timing and resource usage of the megafunction. If you turn on this option, a netlist file ( <b>_syn.v</b> ) is generated. This file is a representation of the customized logic used in the Quartus II software and provides the connectivity of the architectural elements in the megafunction but may not represent true functionality.
6	Summary Page	Specify the types of files to be generated. The Variation file ( <i>&lt;function name&gt;.v</i> ) contains wrapper code in the language you specified on page 2a and is automatically generated. Choose from the following types of files: <ul style="list-style-type: none"> <li>■ AHDL Include file (<i>&lt;function name&gt;.inc</i>)</li> <li>■ VHDL component declaration file (<i>&lt;function name&gt;.cmp</i>)</li> <li>■ Quartus II symbol file (<i>&lt;function name&gt;.bsf</i>)</li> <li>■ Instantiation template file (<i>&lt;function name&gt;_inst.v</i>)</li> <li>■ Verilog HDL black box file (<i>&lt;function name&gt;_bb.v</i>)</li> </ul> For more information about the wizard-generated files, refer to the <a href="#">Introduction to Megafunctions User Guide</a> .

Table 2–3 lists the parameter settings for the ALTUFM\_I2C megafunction.

**Table 2–3. ALTUFM\_I2C Parameter Settings (Part 1 of 3)**

MegaWizard Plug-in Manager Page	Configuration Setting	Description
1	Which action do you want to perform?	You can select from the following options: <b>Create a new custom megafunction variation</b> , <b>Edit an existing custom megafunction variation</b> , or <b>Copy an existing custom megafunction variation</b> .
2a	Select a megafunction from the list below	Select <b>ALTUFM_I2C</b> from the <b>Memory Compiler</b> category.
	Which device family will you be using?	Specify the device family that you want to use.
	Which type of output file do you want to create?	You can choose AHDL( <b>.tdf</b> ), VHDL( <b>.vhd</b> ), or Verilog HDL ( <b>.v</b> ) as the output file type.
	What name do you want for the output file?	Specify the name of the output file.
	Return to this page for another create operation	Turn on this option if you want to return to this page to create multiple megafunctions.

Table 2-3. ALTUFM\_I2C Parameter Settings (Part 2 of 3)

MegaWizard Plug-in Manager Page	Configuration Setting	Description
3	Currently selected device family	Specifies the device family you chose on page 2a.
	Match project/default	Turn on this option to ensure that the device selected matches the device family that is chosen in the previous page.
	Read and write	Turn on this option if you want to enable the read and write access mode.
	Read only	Turn on this option if you want to enable the read only mode.
	Device address MSB (binary)	Select 4-bit address that specifies the 4 MSBs of the device address.
	Memory size	Select the memory size. Values are <b>1K</b> , <b>2K</b> , <b>4K</b> , and <b>8K</b> . If omitted, the default is <b>4K</b> .
	Use 'global_reset' input port	Turn on this option to enable global reset input port.
	Use 'osc' output port	Turn on this option to route the oscillator frequency to an external oscillator port.
	Use 'oscena' input port	Turn on this option to enable the oscillator enable port.
4	Write Configuration	Specify the write configuration for the I2C protocol. Select <b>Single byte write</b> or <b>Page write</b> . If you select <b>Page write</b> , specify <b>8 bytes</b> , <b>16 bytes</b> , or <b>32 bytes</b> .
	Use 'wp' write protect input port	Select this option to use the write protect input port, and define how the write protect is applied; write protect the full memory or only the upper half of the memory.
	Erase method	Select the erase method. Options are full erase, sector erase, and no erase. If you select the <b>Sector Erase Triggered by Byte Address</b> option, enter the binary address for sectors 0 and 1.  Note that the <b>Sector Erase Triggered by Byte Address</b> option is only available if <b>Single byte write</b> is selected as the write configuration.

Table 2-3. ALTUFM\_I2C Parameter Settings (Part 3 of 3)

MegaWizard Plug-in Manager Page	Configuration Setting	Description
5	Memory content initialization	<ul style="list-style-type: none"> <li>■ Select <b>Initialize blank memory</b> if you do not want to specify any initialization file. Select <b>Initialize from hex or mif file</b> to specify the initialization file. Type the file name or browse for the required file.</li> <li>■ In the Quartus II software, the memory content values from your <b>.hex</b> or <b>.mif</b> are hard-coded into your ALTUFM_I2C, ALTUFM_NONE, ALTUFM_PARALLEL, or ALTUFM_SPI MegaWizard™ Plug-In Managers. If you change the contents of your <b>.hex</b> or <b>.mif</b> after running the MegaWizard Plug-In Manager, these updates will not be reflected in simulation. This may cause a mismatch between simulation and device behavior because compilation and program file generation in the Quartus II software use the current <b>.hex</b> or <b>.mif</b> contents instead of the hard-coded values. To avoid this mismatch, re-run the MegaWizard Plug-In Manager for your ALTUFM megafunction whenever you update your <b>.hex</b> or <b>.mif</b>.</li> </ul>
	Oscillator frequency	Specify the oscillator frequency for the user flash memory. This parameter is used for simulation purposes only. The values are <b>5.56MHz</b> and <b>3.33MHz</b> . If omitted, the default is <b>5.56MHz</b> .
	Erase time	Specify the erase time.
	Program time	Specify the program time.
6	Generate netlist	Turn on this option if you want to generate a netlist for your third-party EDA synthesis tool to estimate the timing and resource usage of the megafunction. If you turn on this option, a netlist file ( <b>_syn.v</b> ) is generated. This file is a representation of the customized logic used in the Quartus II software and provides the connectivity of the architectural elements in the megafunction but may not represent true functionality.
7	Summary Page	<p>Specify the types of files to be generated. The Variation file (<b>&lt;function name&gt;.v</b>) contains wrapper code in the language you specified on page 2a and is automatically generated. Choose from the following types of files:</p> <ul style="list-style-type: none"> <li>■ AHDL Include file (<b>&lt;function name&gt;.inc</b>)</li> <li>■ VHDL component declaration file (<b>&lt;function name&gt;.cmp</b>)</li> <li>■ Quartus II symbol file (<b>&lt;function name&gt;.bsf</b>)</li> <li>■ Instantiation template file (<b>&lt;function name&gt;_inst.v</b>)</li> <li>■ Verilog HDL black box file (<b>&lt;function name&gt;_bb.v</b>)</li> </ul> <p>For more information about the wizard-generated files, refer to the <a href="#">Introduction to Megafunctions User Guide</a>.</p>

Table 2–3 lists the parameter settings for the ALTUFM\_NONE megafunction.

**Table 2–4. ALTUFM\_NONE Parameter Settings (Part 1 of 2)**

MegaWizard Plug-in Manager Page	Configuration Setting	Description
1	Which action do you want to perform?	You can select from the following options: <b>Create a new custom megafunction variation</b> , <b>Edit an existing custom megafunction variation</b> , or <b>Copy an existing custom megafunction variation</b> .
2a	Select a megafunction from the list below	Select <b>ALTUFM_NONE</b> from the <b>Memory Compiler</b> category.
	Which device family will you be using?	Specify the device family that you want to use.
	Which type of output file do you want to create?	You can choose AHDL(. <b>tdf</b> ), VHDL(. <b>vhd</b> ), or Verilog HDL(. <b>v</b> ) as the output file type.
	What name do you want for the output file?	Specify the name of the output file.
	Return to this page for another create operation	Turn on this option if you want to return to this page to create multiple megafunctions.
3	Currently selected device family	Specifies the device family you chose on page 2a.
	Match project/default	Turn on this option to ensure that the device selected matches the device family that is chosen in the previous page.
	Use 'arclkena' output port	Turn on this option to enable the <b>arclkena</b> port.
	Use 'drclkena' input port	Turn on this option to enable <b>drclkena</b> port.
4	Memory content initialization	<ul style="list-style-type: none"> <li>■ Select <b>Initialize blank memory</b> if you do not want to specify any initialization file. Select <b>Initialize from hex or mif file</b> to specify the initialization file. Type the file name or browse for the required file.</li> <li>■ In the Quartus II software, the memory content values from your <b>.hex</b> or <b>.mif</b> are hard-coded into your ALTUFM megafunction variation file when you run the ALTUFM_I2C, ALTUFM_NONE, ALTUFM_PARALLEL, or ALTUFM_SPI MegaWizard™ Plug-In Managers. If you change the contents of your <b>.hex</b> or <b>.mif</b> after running the MegaWizard Plug-In Manager, these updates will not be reflected in simulation. This may cause a mismatch between simulation and device behavior because compilation and program file generation in the Quartus II software use the current <b>.hex</b> or <b>.mif</b> contents instead of the hard-coded values. To avoid this mismatch, re-run the MegaWizard Plug-In Manager for your ALTUFM megafunction whenever you update your <b>.hex</b> or <b>.mif</b>.</li> </ul>
	Oscillator frequency	Specify the oscillator frequency for the user flash memory. This parameter is used for simulation purposes only. The values are <b>5.56MHz</b> and <b>3.33MHz</b> . If omitted, the default is <b>5.56MHz</b> .
	Erase time	Specify the erase time.
	Program time	Specify the program time.

**Table 2-4. ALTUFM\_NONE Parameter Settings (Part 2 of 2)**

MegaWizard Plug-in Manager Page	Configuration Setting	Description
5	Generate netlist	Turn on this option if you want to generate a netlist for your third-party EDA synthesis tool to estimate the timing and resource usage of the megafunction. If you turn on this option, a netlist file ( <b>_syn.v</b> ) is generated. This file is a representation of the customized logic used in the Quartus II software and provides the connectivity of the architectural elements in the megafunction but may not represent true functionality.
6	Summary Page	Specify the types of files to be generated. The Variation file ( <i>&lt;function name&gt;.v</i> ) contains wrapper code in the language you specified on page 2a and is automatically generated. Choose from the following types of files: <ul style="list-style-type: none"> <li>■ AHDL Include file (<i>&lt;function name&gt;.inc</i>)</li> <li>■ VHDL component declaration file (<i>&lt;function name&gt;.cmp</i>)</li> <li>■ Quartus II symbol file (<i>&lt;function name&gt;.bsf</i>)</li> <li>■ Instantiation template file (<i>&lt;function name&gt;_inst.v</i>)</li> <li>■ Verilog HDL black box file (<i>&lt;function name&gt;_bb.v</i>)</li> </ul> For more information about the wizard-generated files, refer to the <a href="#">Introduction to Megafunctions User Guide</a> .

## Command Line Interface Parameters

Expert users can choose to instantiate and parameterize the megafunction through the command-line interface using the clear box generator command. This method requires you to have command-line scripting knowledge.



For more information about using the clear box generator, refer to the *Introduction to Megafunctions User Guide*.

Table 2–5 lists the parameters for the ALTUFM\_PARALLEL megafunction.

**Table 2–5. ALTUFM\_PARALLEL Megafunction Parameters**

Parameter Name	Type	Required	Description
ACCESS_MODE	String	No	Specifies the access mode for the user flash memory. Values are READ_WRITE and READ_ONLY. <sup>(1)</sup>
LPM_FILE	String	No	Name of the Memory Initialization File (.mif) or Hexadecimal (Intel-Format) Files (.hex) containing RAM initialization data (<file name>), or UNUSED. If you use a HEX file to initialize with the memory, you must use a word size of 16 bits. Data bits need to be located in the MSBs of the data word and pad the LSBs with 1.
OSC_FREQUENCY	Integer	No	Specifies the oscillator frequency for the user flash memory. Values are 180000 ps (5.56 MHz) or 300000 ps (3.33 MHz). If omitted, the default is 180000 ps. The OSC_FREQUENCY setting controls the oscillator frequency during simulations and does not affect the device's oscillator frequency, which can range from 3.33 MHz (300000 ps) to 5.56 MHz (180000 ps). <sup>(2)</sup>
WIDTH_ADDRESS	Integer	Yes	Specifies the width for the addr bus. <sup>(3)</sup>
WIDTH_DATA	Integer	Yes	Specifies the width for the data bus. <sup>(3)</sup>
WIDTH_UFM_ADDRESS	Integer	Yes	Specifies the internal address width for the user flash memory.
LPM_TYPE	String	No	Identifies the library of parameterized modules (LPM) entity name in VHDL Design Files (.vhd).

**Notes to Table 2–5:**

- (1) This parameter is used with the parallel or SPI interface protocol only.
- (2) This parameter is used for simulation purposes only.
- (3) This parameter is used with the parallel interface protocol only.

Table 2–5 lists the parameters for the ALTUFM\_SPI megafunction.

**Table 2-6. ALTUFM\_SPI Megafunction Parameters**

Parameter Name	Type	Required	Description
ACCESS_MODE	String	No	Specifies the access mode for the user flash memory. Values are <code>READ_WRITE</code> and <code>READ_ONLY</code> . This parameter is used with the parallel or SPI interface protocol only.
CONFIG_MODE	String	Yes	<p>Specifies the configuration mode for the SPI interface protocol. Values are <code>BASE</code> and <code>EXTENDED</code>. The default is <code>EXTENDED</code>. When the <code>CONFIG_MODE</code> parameter is set to <code>EXTENDED</code>, the flash memory uses 16-bit word size for the address and data word. When the <code>CONFIG_MODE</code> parameter is set to <code>BASE</code>, the flash memory uses 8-bit word size for the address and data word and uses only sector 0 of the UFM.</p> <p>When the <code>CONFIG_MODE</code> parameter is specified to <code>BASE</code> and you specify a MIF or HEX file, you must specify the initial contents of sector 0 and 1 of the UFM. Addresses 0 through 255 of the MIF or HEX file must contain user data while addresses 256 through 511 should be set to 1. If you use a HEX file, you must specify all 16-bits of data. The user data (8-bits) should be placed in the MSBs of the data word and the LSBs should be padded with 1.</p>
LPM_FILE	String	No	Name of the Memory Initialization File ( <code>.mif</code> ) or Hexadecimal (Intel-Format) Files ( <code>.hex</code> ) containing RAM initialization data ( <code>&lt;file name&gt;</code> ), or <code>UNUSED</code> .
OSC_FREQUENCY	Integer	No	Specifies the oscillator frequency for the user flash memory. This parameter is used for simulation purposes only. Values are 180000 ps (5.56 MHz) or 300000 ps (3.33 MHz). If omitted, the default is 180000 ps. The <code>OSC_FREQUENCY</code> setting controls the oscillator frequency during simulations and does not affect the device's oscillator frequency, which can range from 3.33 MHz (300000 ps) to 5.56 MHz (180000 ps).
WIDTH_UFM_ADDRESS	Integer	Yes	Specifies the internal address width for the user flash memory.
LPM_TYPE	String	No	Identifies the library of parameterized modules (LPM) entity name in VHDL Design Files ( <code>.vhd</code> ).

Table 2–7 lists the parameters for the ALTUFM\_I2C megafunction.

**Table 2–7. ALTUFM\_I2C Megafunction Parameters**

Parameter Name	Type	Required	Description
ACCESS_MODE	String	Yes	Specifies the access mode. Values are READ_WRITE and READ_ONLY. If omitted, the default is READ_WRITE.
INTENDED_DEVICE_FAMILY	String	No	Create the ALTMULT_ADD megafunction with the MegaWizard Plug-in Manager to calculate the value for this parameter. <sup>(1)</sup>
ERASE_METHOD	String	No	Specifies the erase method. Values are MEM_ADD, A2_ERASE, DEV_ADD_111, and NO_ERASE. If omitted, the default is MEM_ADD.
FIXED_DEVICE_ADD	String	Yes	4-bit address that specifies the 4 MSBs of the device address.
LPM_FILE	String	No	Name of the Memory Initialization File (.mif) or Hexadecimal (Intel-Format) Files (.hex) containing RAM initialization data (<file name>), or UNUSED. If you use a HEX file to initialize with the memory, you must use a word size of 16 bits. Data bits need to be located in the MSBs of the data word and pad the LSBs with 1.
MEM_ADD_ERASE0	String	No	If the ERASE_METHOD parameter has a value of MEM_ADD, the MEM_ADD_ERASE0 parameter must specify the 8-bit memory address that erases sector 0 of the UFM block.
MEM_ADD_ERASE1	String	No	If the ERASE_METHOD parameter has a value of MEM_ADD, the MEM_ADD_ERASE1 parameter must specify the 8-bit memory address that erases sector 1 of the UFM block.
MEM_PROTECT	String	No	Specifies whether the write-protect port protects only the upper half of the UFM block or the entire UFM block from writes/erases. Values are FULL and UPPER_HALF. If omitted, the default is FULL.
MEMORY_SIZE	String	Yes	Specifies the memory size. Values are 1K, 2K, and 4K. If omitted, the default is 4K.
OSC_FREQUENCY	Integer	No	Specifies the oscillator frequency for the user flash memory. Values are 180000 ps (5.56 MHz) or 300000 ps (3.33 MHz). If omitted, the default is 180000 ps. The OSC_FREQUENCY setting controls the oscillator frequency during simulations and does not affect the device's oscillator frequency, which can range from 3.33 MHz (300000 ps) to 5.56 MHz (180000 ps). <sup>(2)</sup>
WRITE_MODE	String	No	Specifies the write mode. If omitted, the default is SINGLE_BYTE.
LPM_HINT	String	No	Allows you to assign Altera-specific parameters in VHDL Design Files (.vhd). The default is UNUSED.
LPM_TYPE	String	No	Identifies the library of parameterized modules (LPM) entity name in VHDL Design Files.

**Notes to Table 2–7:**

- (1) This parameter is used for modeling and behavioral simulation purposes
- (2) This parameter is used for simulation purposes only.

Table 2–8 lists the parameters for the ALTUFM\_NONE megafunction.

**Table 2–8. ALTUFM\_NONE Megafunction Parameters**

Parameter Name	Type	Required	Description
busy	Yes	Busy signal that indicates when memory is busy.	(1)
drdout	Yes	Data register output.	(1)
osc	No	Oscillator output.	If the osc port is specified, the oscena port is required. (1)
rtpbusy	Yes	Busy signal that indicates when in system configuration is using flash memory.	When the rtpbusy is high, it cannot be used. (1)

**Note to Table 2–8:**

(1) This port is used without an interface protocol only.



This chapter describes the functional description and the design examples of the ALTUFM megafunction. This section also includes the ports descriptions of the ALTUFM megafunction. You can use the ports to customize the ALTUFM megafunction according to your application.

### User Flash Memory

User flash memory (UFM) provides access to the serial flash memory blocks in MAX II and MAX V devices. The UFM can be used like a serial EEPROM for storing up to 8,192 bits of non-volatile information. The hardware interface is a simple 12-pin protocol, providing interfaces similar to the industry standards for serial EEPROMs. The UFM connects to the logic array through the MultiTrack™ interconnect, allowing any logic cell to interface with the UFM. The UFM features two sectors for partitioned sector erase, a built-in internal oscillator that drives internal device logic; program, erase, and busy signals, auto-increment addressing, and a serial interface to the internal device logic with a programmable interface. The ALTUFM megafunction, available in Altera's Quartus® II software, provides interface logic for a subset of these interfaces (parallel and SPI).



Any interfaces not provided by the megafunction or design examples require you to create user logic to bridge the UFM block to your desired interface protocol.

Each UFM array is organized as two separate sectors, with 4,096 bits per sector. Each sector can be erased independently.

Table 3–1 lists the capacity for the UFM block for all MAX II devices.

**Table 3–1. MAX II UFM Array Size**

Device	Total Bits	Sectors	Address Bits	Data Width
EPM240	8,192	2 (4096 bits per sector)	9	16
EPM570	8,192	2 (4096 bits per sector)	9	16
EPM1270	8,192	2 (4096 bits per sector)	9	16
EPM2210	8,192	2 (4096 bits per sector)	9	16

### Memory Organization Map

The memory organization map includes 512 locations with 9 bits, addressing a range of 000h to 1FFh. Each location stores 16-bit wide data. The most significant bit (MSB) of the address register indicates the sector in operation.

Table 3–2 lists the memory organization for the MAX II UFM block.

**Table 3–2. Memory Organization**

Sector	Address Range	
1	100h	1FFh
0	000h	0FFh

## Using & Accessing UFM Storage

Use the UFM to store data of different memory sizes and data widths. The UFM storage width is 16 bits, however, you can implement different data widths or a serial interface using the ALTUFM megafunction.

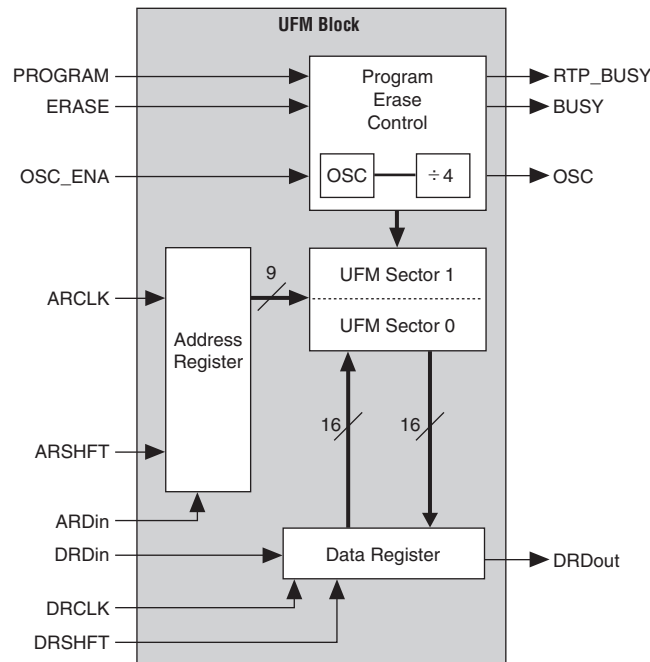
The different data widths available for the three types of interfaces supported in the Quartus II software are shown in Table 3–3.

**Table 3–3. Data Widths for Logic Array Interfaces**

Logic Array Interface	Data Width (Bits)	Interface Type
SPI	8 or 16	Serial
Parallel	Options of 3 to 16	Parallel
I2C	8	Serial
None	16	Serial

The MAX II UFM block diagram is shown in [Figure 3-1](#).

**Figure 3-1. MAX II UFM Block Diagram and Interface Signals Block Diagram**




## UFM Operating Modes

There are three UFM block modes:

- Read/stream read
- Program (write)
- Erase

The UFM block supports byte write, but does not support byte erase, requiring a sector-based erase sequence prior to any programming or writing. If the data content of a specific byte location needs to be overwritten in the UFM, the entire sector that byte resides in must be erased unless that byte location was already erased (all 1s).

 For more information about programming and erasing the UFM block and the ALTUFM megafunction, refer to the [Using User Flash Memory in MAX II Devices](#) chapter in the *MAX II Device Handbook* and the [User Flash Memory in MAX V Devices](#) chapter in the *MAX V Device Handbook*.

If your design allows you to access the MAX II and MAX V UFM (write or erase), you must ensure that all the erase or write operations of the UFM are completed before starting any ISP session (including stand-alone verify, examine, setting security bit, and reading the contents of the UFM). Never start an ISP session when any erase or write operation of the UFM is in progress, as this may put the device in an unrecoverable state. This restriction does not apply to the read operation of the UFM.

The MAX II and MAX V UFM can be programmed, erased, and verified through the Joint Test Action Group (JTAG) port, or through connections to or from the logic array in accordance with IEEE Std. 1532-2002. There are 13 interface signals (Figure 3-1) to and from the UFM block and logic array, which allow the logic array to read or write to the UFM during device user mode. A reference design or user logic can be used to interface the UFM to many standard interface protocols such as Serial Communication Interface (SCI), Serial Peripheral Interface (SPI), Inter-Integrated Circuit (I2C), Microwire, or other proprietary protocols.

## Serial Peripheral Interface

Serial peripheral interface (SPI) is a four-pin serial communication subsystem included on the Motorola 6805 and 68HC11 series microcontrollers. SPI allows the microcontroller unit to communicate with peripheral devices, and is capable of inter-processor communications in a multiple-master system.

The SPI bus consists of masters and slaves. The master device initiates and controls the data transfers, and provides the clock signal for synchronization. The slave device responds to the data transfer request from the master device. The master device in an SPI bus initiates a service request and the slave devices respond to the service request. The UFM is configured as the slave device for the SPI bus.

There are only four pins in SPI: *SI*, *SO*, *SCK*, and *nCS*. Data transmitted to the *SI* port of the slave device is sampled by the slave device at the positive *SCK* clock. Data transmits from the slave device through *SO* at the negative *SCK* clock edge. When *nCS* is asserted, it means the current device is being selected by the master device from the other end of the SPI bus for service. When *nCS* is not asserted, the *SI* and *SCK* ports should be blocked from receiving signals from the master device, and *SO* should be in high impedance state to avoid causing contention on the shared SPI bus. All instructions, addresses, and data are transferred with the MSB first, and start with high-to-low *nCS* transition.

The *nCS* signal cannot be toggled simultaneously with the clock edge of *SCK*. During read/write mode, a low-to-high transition of *nCS* requires a minimum of 420 ns of hold time before it can be asserted low again. A high-to-low transition of *nCS* requires a minimum wait of 420 ns before the first *SCK* clock edge.



For more information on the SPI bus to the UFM, refer to the *Using User Flash Memory in MAX II Devices* chapter in the *MAX II Device Handbook* and the *User Flash Memory in MAX V Devices* chapter in the *MAX V Device Handbook*.

## Parallel Interface


This interface allows for parallel communication between the UFM block and outside logic. Once the *READ* request, *WRITE* request, or *ERASE* request is asserted (active low assertion), the outside logic or device (such as a microcontroller) are free to continue their operation while the data in the UFM is retrieved, written, or erased. During this time, the *nBUSY* signal is driven “low” to indicate that it is not available to respond to any further request. After the operation is complete, the *nBUSY* signal is brought back to “high” to indicate that it is now available to service a new request. If it was the *Read* request, the *DATA\_VALID* is driven “high” to indicate that the data at the *DO* port is the valid data from the last read address.

Asserting READ, WRITE, and ERASE at the same time is not allowed. Multiple requests are ignored and nothing is read from, written to, or erased in the UFM block. There is no support for sequential read and page write in the parallel interface.

Even though the altufm megafunction allows you to select the address widths range from 3 bits to 9 bits, the UFM block always expects full 9 bits width for the address register. Therefore, the ALTUFM megafunction always pads the remaining LSB of the address register with '0's if the register width selected is less than 9 bits. The address register will point to sector 0 if the address received at the address register starts with a '0'. On the other hand, the address register will point to sector 1 if the address received starts with a '1'.


Even though you can select an optional data register width of 3 to 16 bits using the altufm megafunction, the UFM block always expects full 16 bits width for the data register. Reading from the data register will always proceed from MSB to LSB. The altufm megafunction will always pad the remaining LSB of the data register with 1s if the user selects a data width of less than 16-bits.

During the read/write mode, a high-to-low transition of a mode signal (nREAD, nWRITE, or nERASE) requires a minimum of 420 ns of hold time before the instruction signal can be pulled high again. The address register and data input must be held for at least 420 ns once the mode signal is asserted low. The high-to-low transition of nBUSY requires a maximum wait of 210 ns once nRead, nWrite, or nErase is asserted low.

 For more information about the parallel interface to the UFM, refer to the *Using User Flash Memory in MAX II Devices* chapter in the *MAX II Device Handbook* and the *User Flash Memory in MAX V Devices* chapter in the *MAX V Device Handbook*.

## None (Altera Serial Interface)

None means using the dedicated UFM serial interface. The built-in UFM interface uses 13 pins for the communication. You can produce your own interface design to communicate to/from the dedicated UFM interface and implement it in the logic array.

 For more information about the Altera interface to the UFM, refer to the *Using User Flash Memory in MAX II Devices* chapter in the *MAX II Device Handbook* and the *User Flash Memory in MAX V Devices* chapter in the *MAX V Device Handbook*.

## Inter-Integrated Circuit Interface

The inter-integrated circuit (I<sup>2</sup>C) is a bidirectional two-wire interface protocol. Choose this interface to configure the UFM block and logic as a slave device for the I<sup>2</sup>C bus. The size of UFM memory, the access mode, the erase method, and the protection required for the UFM block all dictate the resources required on a particular device for this interface implementation.

 For more information about using the ALTUFM megafunction with the I<sup>2</sup>C interface, refer to *AN489: Using the UFM in MAX II Devices*.

## Common Applications

The MAX II and MAX V UFM block is the best choice for storing manufacturing data, helping to improve board space efficiency, and minimizing system cost by integrating board-level flash memory, EEPROM capabilities, and system logic into one device. You can customize the UFM communication system to comply with different manufacturers' standard interface protocols to access manufacturing product data.

The UFM block is used to replace on-board flash and EEPROM memory devices which are used to store ASSP or processor configuration bits, or electronic ID information for a board during manufacturing. Since you can program the UFM block to suit your needs, MAX II and MAX V devices offer more interface flexibility than an off-the-shelf EEPROM device.

## Design Example: User Flash Memory with SPI Interface

This design example uses the ALTUFM megafunction to implement user flash memory with the SPI Interface using the MegaWizard Plug-In Manager in the Quartus II software.

In this example, you perform the following activities:

- Create user flash memory with an SPI interface using the ALTUFM megafunction and the MegaWizard Plug-in Manager
- Implement the design and assign the EPM2210F256C3 device to the project
- Compile and simulate the design

The design examples are available for download from the following locations:

- On the [Documentation: Quartus II Development Software](#) page, expand the **Using Megafunctions** section and then expand the **I/O** section.
- On the [Documentation: User Guides](#) section of the Altera website.

## Generate the User Flash Memory

Perform the following steps to generate the user flash memory:

1. In the Quartus II software, open **alt\_ufm\_DesignExample.qar** and restore the archive file into your working directory.
2. On the Tools menu, select **MegaWizard Plug-In Manager**. The **MegaWizard Plug-In Manager** page displays
3. Select **Create a new custom megafunction variation**.
4. Click **Next**. Page 2a of the MegaWizard Plug-In Manager appears.

- In the MegaWizard Plug-In Manager pages, select or verify the configuration settings shown in Table 3-4. Click **Next** to advance from one page to the next.

**Table 3-4. Configuration Settings for ALTUFM Design Example**

MegaWizard Plug-in Manager Page	Configuration Setting	Value
2a	Select a megafunction	<b>ALTUFM_SPI</b>
	Which device family will you be using?	<b>MAX II</b>
	Which type of output file do you want to create?	<b>Verilog HDL</b>
	What name do you want for the output file?	ufm_ex
3	Currently selected device family	<b>MAX II</b>
	Match project/default	Turned on
	Access mode	<b>Read and write</b>
	Configuration mode	<b>Extended mode (16 bit address and data)</b>
	Use 'osc' output port	Turned on
	Use 'oscena' input port	Turned off
4	Memory content initialization	Initialize blank memory
	Oscillator frequency	<b>3.33</b>
	Erase time	—
	Program time	—
5	Generate netlist	Turned off
6	Variation file	Turned on
	Quartus II symbol file	Turned off
	Instantiation template file	Turned off
	Verilog HDL black-box file	Turned on
	AHDL Include file	Turned off
	VHDL component declaration file	Turned off

- Click **Finish**.

The ALTUFM variation is now built.

## Implement the User Flash Memory

This section describes how to assign the EPM2210F256C3 device to the project and compile the project.

- In the Quartus II software, on the Assignments menu, click **Settings**.
- The **Device Settings** window displays.
- In the **Category** list, select **Device**.
- In the **Family** list, select **MAX II**.
- In the **Target device** list, click **Specific device selected in 'Available devices' list**.

6. In the **Available devices** list, select **EPM2210F256C3**.
7. Leave the other options in the default state and click **OK**.
8. On the Processing menu, select **Start Compilation** to compile the design.
9. The **Full compilation was successful** box displays. Click **OK**.

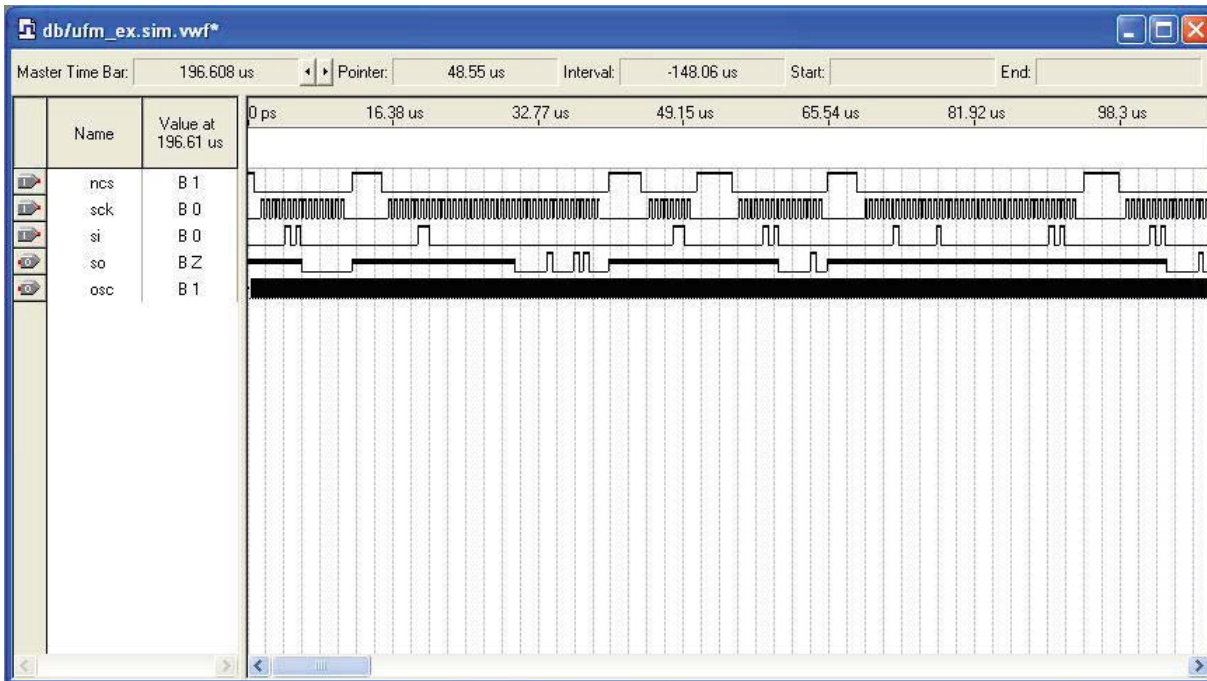
## Functional Results—Simulate the User Flash Memory in the Quartus II Software

This section describes how to verify the design example you just created by simulating the design using the Quartus II Simulator. To set up the Quartus II Simulator, follow these steps:

1. On the Processing menu, click the **Generate Functional Simulation Netlist** option.
2. When the **Functional Simulation Netlist Generation was successful** message appears, click **OK**.
3. On the Assignments menu, click **Settings** and then select **Simulator Settings** from the Category list.
4. In the **Category** list, select **Simulator**.
5. In the **Simulation mode** list, select **Functional**.
6. Type **ufm\_ex\_ip.vwf** in the **Simulation input** box, or click **Browse (...)** to select the file in the project folder.
7. Turn on the **End simulation at:** option and type **50.0** and select **ms** from the list.
8. Turn on **Automatically add pins to simulation output waveforms** and **Simulation coverage reporting** options.
9. Turn off **Check outputs** option.
10. Turn off **Overwrite simulation input file with simulation results**.
11. Turn off **Generate Signal Activity File** option.
12. Click **OK**.
13. On the Processing menu, click **Start Simulation** to run a simulation.

14. The Simulation Report window appears. Verify the results (Figure 3-2).

Figure 3-2. Simulation Waveform



## Functional Results—Simulate the User Flash Memory in ModelSim-Altera Software

Simulate the design in the ModelSim-Altera software to compare the results of both simulators.

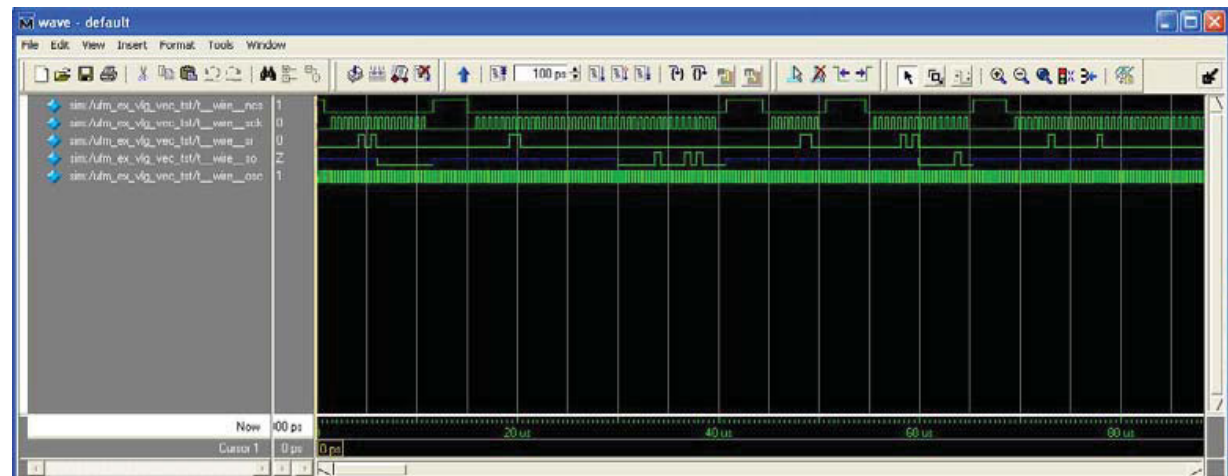
To set up the ModelSim-Altera software, follow these steps:

1. Unzip the **alt\_ufm\_msim.zip** file to any working directory on your PC.
2. Start the ModelSim-Altera software.
3. On the File menu, click **Change Directory**.
4. Select the folder in which you unzipped the files. Click **OK**.
5. On the Tools menu, click **Execute Macro**.
6. Select the **ufm\_ex.do** file and click **Open**. This is a script file for ModelSim-Altera software that automated all necessary settings for the simulation.
7. Verify the results shown in the Waveform Viewer window.

You can rearrange signals, remove signals, add signals, and change the radix by modifying the script in **ufm\_ex.do** accordingly.

Figure 3-1 shows the expected simulation results in the ModelSim-Altera software.

**Figure 3-3. ModelSim-Altera Software Simulation Waveforms**



## ALTUFM\_PARALLEL Megafunction Ports

Table 3-5 and Table 3-6 list the input and output ports for the ALTUFM\_PARALLEL megafunction.

### Input Ports

**Table 3-5. ALTUFM\_PARALLEL Megafunction Input Ports**

Port Name	Required	Description	Comments
addr[]	Yes	Address bus.	Input port [WIDTH_ADDRESS-1..0] wide. The add port is used with the parallel interface protocol only. <sup>(1)</sup>
di[]	Yes	Data bus input.	Input port [WIDTH_DATA-1..0] wide. <sup>(1)</sup>
nerase	No	Erase input port.	When the ACCESS_MODE parameter is set to READ_ONLY, the nerase port cannot be used. <sup>(1)</sup>
nread	Yes	Read input port.	<sup>(1)</sup>
nwrite	No	Write input port.	<sup>(1)</sup>

**Note to Table 3-5:**

(1) This port is used with the parallel interface protocol only.

### Output Ports

**Table 3-6. ALTUFM\_PARALLEL Megafunction Output Ports**

Port Name	Required	Description	Comments
data_valid	Yes	Data output.	<sup>(1)</sup>
nbusy	Yes	Busy signal.	<sup>(1)</sup>

**Table 3-6. ALTUFM\_PARALLEL Megafunction Output Ports**

Port Name	Required	Description	Comments
do[]	Yes	Data bus output.	Output port [WIDTH_DATA-1..0] wide. (1)

**Note to Table 3-6:**

(1) This port is used with the parallel interface protocol only.

## ALTUFM\_SPI Megafunction Ports

Table 3-7 and Table 3-8 list the input and output ports for the ALTUFM\_SPI megafunction.

### Input Ports

**Table 3-7. ALTUFM\_SPI Megafunction Input Ports**

Port Name	Required	Description	Comments
ncs	Yes	Device select input.	(1)
sck	Yes	Serial data clock.	(1)
si	Yes	Serial data input.	(1)

**Note to Table 3-7:**

(1) This port is used with the SPI interface protocol only.

### Output Ports

**Table 3-8. ALTUFM\_SPI Megafunction Output Ports**

Port Name	Required	Description	Comments
so	Yes	Serial bus output.	The si port is used with the SPI interface protocol only.

## ALTUFM\_I2C Megafunction Ports

Table 3-9, Table 3-10, and Table 3-11 list the input, output, and bidirectional ports for the ALTUFM\_I2C megafunction.

### Input Ports

**Table 3-9. ALTUFM\_I2C Megafunction Input Ports**

Port Name	Required	Description	Comments
a0	Yes	Input port that specifies the LSB (bit 0) of the 7-bit device address.	(1)
a1	Yes	Input port that specifies the first bit of the 7-bit device address.	(1)

**Table 3–9. ALTUFM\_I2C Megafunction Input Ports**

Port Name	Required	Description	Comments
a2	Yes	Input port that specifies the second bit of the 7-bit device address.	(1)
wp	No	Write protect input port.	If wp port is set to 1, the memory is write protected and erase and write are disabled.

**Note to Table 3–9:**

(1) This port can be used to vary the device address allocated to the ALTUFM\_I2C megafunction.

## Output Ports

**Table 3–10. ALTUFM\_I2C Megafunction Output Ports**

Port Name	Required	Description	Comments
osc	No	Oscillator output port.	The osc port can be used to access the UFM internal oscillator. The oscillator can be used as a general-purpose clock for other logic circuitry.

## Bidirectional Ports

**Table 3–11. ALTUFM\_I2C Megafunction Bidirectional Ports**

Port Name	Required	Description	Comments
scl	Yes	Bidirectional clock port.	Clock from master to slave.
sda	Yes	Bidirectional clock port.	Data input from master and data output from slave.

## ALTUFM\_NONE Megafunction Ports

Table 3–12 and Table 3–13 list the input and output ports for the ALTUFM\_NONE megafunction.

## Input Ports

**Table 3–12. ALTUFM\_NONE Megafunction Input Ports**

Port Name	Required	Description	Comments
arclk	Yes	Clock for the address register	(1)
ardin	Yes	Input for the address register.	(1)
arshft	Yes	Shift signal for the address register.	(1)
drclk	Yes	Clock for the data register.	(1)
drdin	Yes	Input for the data register.	(1)
drshft	Yes	Shift signal for the data register.	(1)

**Table 3-12. ALTUFM\_NONE Megafunction Input Ports**

Port Name	Required	Description	Comments
erase	Yes	Signal that controls the erase sequence.	(1)
oscena	No	Signal that enables the internal oscillator.	If the <code>osc</code> port is specified, the <code>oscena</code> port is required. (1)
program	No	Signal that initiates a program sequence.	(1)

**Note to Table 3-12:**

(1) This port is used without an interface protocol only.

## Output Ports

**Table 3-13. ALTUFM\_NONE Megafunction Output Ports**

Port Name	Required	Description	Comments
busy	Yes	Busy signal that indicates when memory is busy.	(1)
drdout	Yes	Data register output.	(1)
osc	No	Oscillator output.	If the <code>osc</code> port is specified, the <code>oscena</code> port is required. (1)
rtpbust	Yes	Busy signal that indicates when in system configuration is using flash memory.	When the <code>rtpbust</code> is high, it cannot be used. (1)

**Note to Table 3-13:**

(1) This port is used without an interface protocol only.



This chapter provides additional information about the document and Altera.

## Document Revision History

The following table lists the revision history for this document.

Date	Version	Changes
May 2012	3.1	Updated "Parameter Settings" on page 2–1.
March 2012	3.0	Added MAX V support.
August 2006	2.0	Updated for Quartus II 6.0 software.
July 2005	1.1	Updated for Quartus II 4.2 software.
May 2005	1.0	Initial release.

## How to Contact Altera

To locate the most up-to-date information about Altera products, refer to the following table.

Contact <sup>(1)</sup>	Contact Method	Address
Technical support	Website	<a href="http://www.altera.com/support">www.altera.com/support</a>
Technical training	Website	<a href="http://www.altera.com/training">www.altera.com/training</a>
	Email	<a href="mailto:custrain@altera.com">custrain@altera.com</a>
Product literature	Website	<a href="http://www.altera.com/literature">www.altera.com/literature</a>
Nontechnical support (general) (software licensing)	Email	<a href="mailto:nacomp@altera.com">nacomp@altera.com</a>
	Email	<a href="mailto:authorization@altera.com">authorization@altera.com</a>









### Note to Table:

(1) You can also contact your local Altera sales office or sales representative.

## Typographic Conventions

The following table shows the typographic conventions this document uses.

Visual Cue	Meaning
<b>Bold Type with Initial Capital Letters</b>	Indicate command names, dialog box titles, dialog box options, and other GUI labels. For example, <b>Save As</b> dialog box. For GUI elements, capitalization matches the GUI.
<b>bold type</b>	Indicates directory names, project names, disk drive names, file names, file name extensions, software utility names, and GUI labels. For example, <b>\qdesigns</b> directory, <b>D:</b> drive, and <b>chiptrip.gdf</b> file.
<i>Italic Type with Initial Capital Letters</i>	Indicate document titles. For example, <i>Stratix IV Design Guidelines</i> .

Visual Cue	Meaning
<i>italic type</i>	Indicates variables. For example, $n + 1$ . Variable names are enclosed in angle brackets (< >). For example, <file name> and <project name>.pof file.
Initial Capital Letters	Indicate keyboard keys and menu names. For example, the Delete key and the Options menu.
“Subheading Title”	Quotation marks indicate references to sections in a document and titles of Quartus II Help topics. For example, “Typographic Conventions.”
Courier type	Indicates signal, port, register, bit, block, and primitive names. For example, data1, tdi, and input. The suffix n denotes an active-low signal. For example, resetn. Indicates command line commands and anything that must be typed exactly as it appears. For example, c:\qdesigns\tutorial\chiptrip.gdf. Also indicates sections of an actual file, such as a Report File, references to parts of files (for example, the AHDL keyword SUBDESIGN), and logic function names (for example, TRI).
↵	An angled arrow instructs you to press the Enter key.
1., 2., 3., and a., b., c., and so on	Numbered steps indicate a list of items when the sequence of the items is important, such as the steps listed in a procedure.
■ ■ ■	Bullets indicate a list of items when the sequence of the items is not important.
	The hand points to information that requires special attention.
	The question mark directs you to a software help system with related information.
	The feet direct you to another document or website with related information.
	The multimedia icon directs you to a related multimedia presentation.
	A caution calls attention to a condition or possible situation that can damage or destroy the product or your work.
	A warning calls attention to a condition or possible situation that can cause you injury.
	The envelope links to the <a href="#">Email Subscription Management Center</a> page of the Altera website, where you can sign up to receive update notifications for Altera documents.
	The feedback icon allows you to submit feedback to Altera about the document. Methods for collecting feedback vary as appropriate for each document.