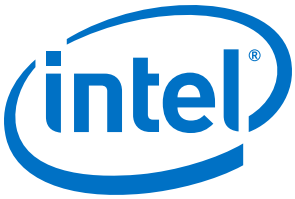


Intel® MAX® 10 User Flash Memory User Guide

Updated for Intel® Quartus® Prime Design Suite: **18.0**

Subscribe
Send Feedback

UG-M10UFM | 2019.07.02
Latest document on the web: PDF | HTML

Contents

1. Intel® MAX® 10 User Flash Memory Overview

Intel® Intel MAX® 10 FPGAs offer a user flash memory (UFM) block that stores non-volatile information.

The UFM provides an ideal storage solution that you can access using the Avalon Memory Mapped (Avalon-MM) slave interface to UFM.

The UFM block also offers the following features.

Features	Capacity
Endurance	Counts to at least 10,000 program/erase cycles
Data retention (after 10,000 program/ erase cycles)	• 20 years at 85 ºC • 10 years at 100 ºC
Maximum operating frequency	• Serial interface — 10M02,10M04, 10M08, 10M16, 10M25: 7.25 MHz — 10M40, 10M50: 4.81 MHz • Parallel interface — 10M02: 7.25 MHz — 10M04, 10M08, 10M16, 10M25, 10M40, 10M50: 116 MHz
Data length	Stores data of up to 32 bits length in parallel

Related Information

- Utilizing the User Flash Memory (UFM) on Intel MAX 10 Devices with a Nios II Processor
- Putting MAX Series FPGAs in Hibernation Mode Using User Flash Memory
- Intel MAX 10 User Flash Memory User Guide Archive on page 26
  Provides a list of user guides for previous versions of the On-Chip Flash Intel FPGA IP core.

2. Intel MAX 10 UFM Architecture and Features

The UFM architecture of Intel MAX 10 devices is a combination of soft and hard IPs. You can only access the UFM using the On-Chip Flash Intel FPGA IP core in the Intel Quartus® Prime software.

2.1. UFM and CFM Array Size

Each array is organized as various sectors.

A page is the smallest amount of flash memory that you can erase at one time. A sector contains a number of pages. You can erase each page or sector independently.

The On-Chip Flash Intel FPGA IP core also gives you access to configuration flash memory (CFM) based on your specification in the parameter editor.

Table 1. UFM and CFM Array Size

This table lists the dimensions of the UFM and CFM arrays.

Device	Pages per Sector					Page Size (Kb)	Total User Flash Memory Size (Kb) (1)	Total Configuration Memory Size (Kb) (1)
	UFM1	UFM0	CFM2 (Image 2)	CFM1 (Image 2)	CFM0 (Image 1)			
10M02	3	3	0	0	34	16	96	544
10M04	0	8	41	29	70	16	1,248	2,240
10M08	8	8	41	29	70	16	1,376	2,240
10M16	4	4	38	28	66	32	2,368	4,224
10M25	4	4	52	40	92	32	3,200	5,888
10M40	4	4	48	36	84	64	5,888	10,752
10M50	4	4	48	36	84	64	5,888	10,752

2.2. UFM Memory Organization Map

The address scheme changes based on the configuration mode you specify in the On-Chip Flash Intel FPGA IP core parameter editor.

The following tables show the dynamic UFM support based on different configuration mode and Intel MAX 10 variant.

(1) The maximum possible value, which is dependent on the mode you select.

Table 2. Dynamic Flash Size Support: Flash and Analog Variants

Configuration	UFM1	UFM0	CFM2 (Image 2)	CFM1 (Image 2)	CFM0 (Image 1)
Dual compressed images	UFM space	UFM space	—	—	—
Single uncompressed image	UFM space	UFM space	UFM space	—	—
Single compressed image	UFM space	UFM space	UFM space	UFM space	—
Single uncompressed image with memory initialization	UFM space	UFM space	—	—	—
Single compressed image with memory initialization	UFM space	UFM space	—	—	—

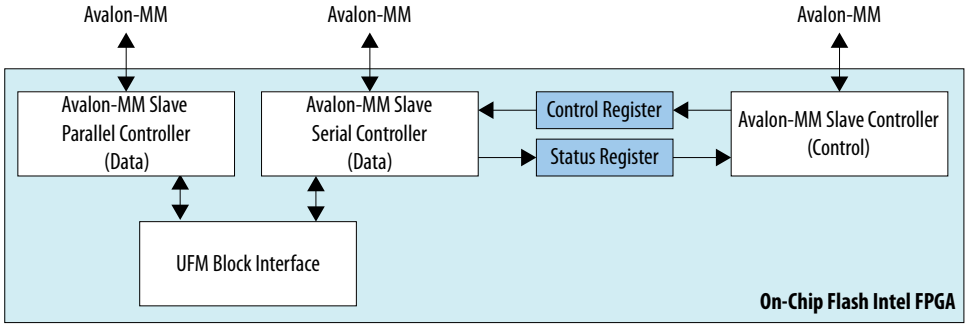
Table 3. Dynamic Flash Size Support: Compact Variant

Configuration	UFM1	UFM0	CFM2 (Image 2)	CFM1 (Image 2)	CFM0 (Image 1)
Dual compressed images	Not available				
Single uncompressed image	UFM space	UFM space	—	—	—
Single compressed image	UFM space	UFM space	—	—	—
Single uncompressed image with memory initialization	Not available				
Single compressed image with memory initialization	Not available				

2.3. UFM Block Diagrams

This figure shows the top level view of the On-Chip Flash Intel FPGA IP core block diagram. The On-Chip Flash Intel FPGA IP core supports both parallel and serial interfaces for Intel MAX 10 FPGAs.

Figure 1. On-Chip Flash Intel FPGA IP Core Block Diagram



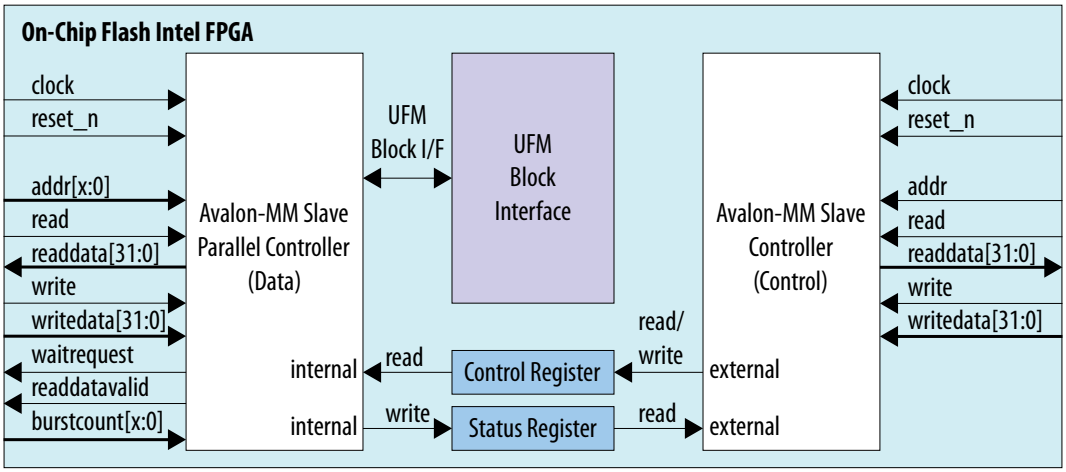
This IP block has two Avalon-MM slave controllers:

- Data—a wrapper of the UFM block that provides read and program accesses to the flash.
- Control—the CSR and status register for the flash, which is required only for program and erase operations.

These figures show the detailed overview of the Avalon-MM interface during read and program (write) operation.

Figure 2. On-Chip Flash Intel FPGA IP Core Avalon-MM Slave Read and Program (Write) Operation in Parallel Mode

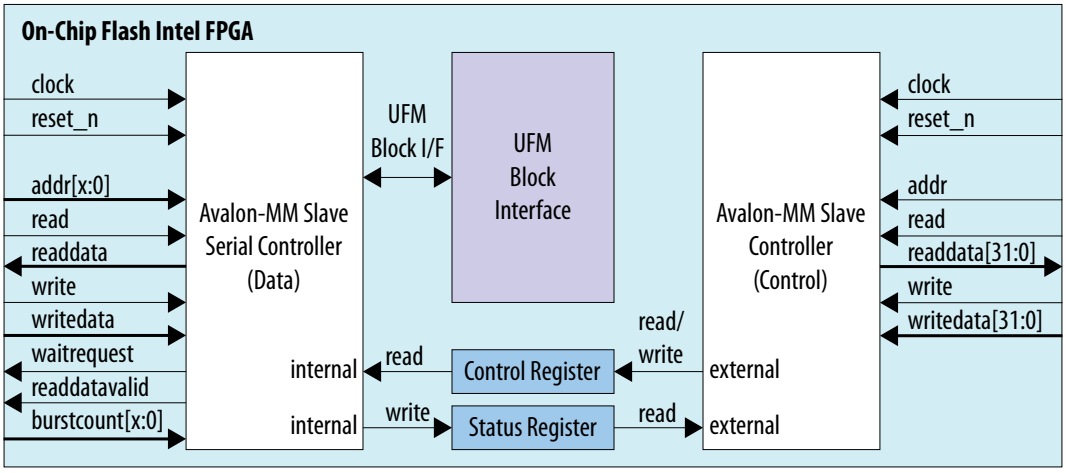
This figure shows the standard interface for Intel MAX 10 devices in parallel mode.



Note: The maximum frequency for all devices in parallel mode, except for 10M02, is 116 MHz. The maximum frequency for 10M02 devices is 7.25 MHz.

Figure 3. On-Chip Flash Intel FPGA IP Core Avalon-MM Slave Read and Program (Write) Operation in Serial Mode

This figure shows the standard interface for Intel MAX 10 devices in serial mode.



These figures show the detailed overview of the Avalon-MM interface during read only operation.

Figure 4. On-Chip Flash Intel FPGA IP Core Avalon-MM Slave Read Only Operation in Parallel Mode

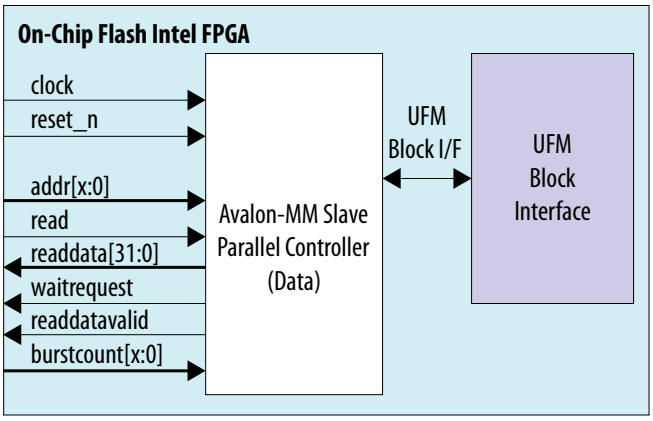
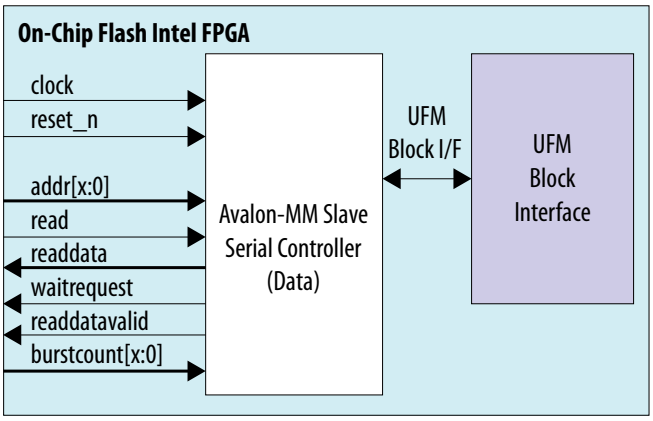


Figure 5. On-Chip Flash Intel FPGA IP Core Avalon-MM Slave Read Only Operation in Serial Mode



2.4. UFM Operating Modes

The UFM block offers the following operating modes:

- Read
- Burst read
- Program (Write)
- Sector erase
- Page erase
- Sector write protection

You can choose one of the following access modes in the On-Chip Flash Intel FPGA IP core parameter editor to read and control the operations.

- Read and program mode—this mode allows both data and control slave interface. This mode is applicable for both UFM and CFM sectors.
- Read only mode—this mode allows only data slave interface, and restricted to only read operations. This mode is applicable for both UFM and CFM sectors.
- Hidden—this mode does not allow any read or program (write) operations. This mode is applicable only for CFM sectors.

The following table shows the comparison between parallel and serial modes.

Table 4. Comparison between Parallel Mode and Serial Mode

Feature	Parallel Mode	Serial Mode
Avalon-MM Data Interface	Parallel mode with 32-bit data bus	Serial mode with 32 bits based burst count
Access Mode	• Read and program • Read only • Hidden	• Read and program • Read only • Hidden
Read Mode	• Incrementing burst read • Wrapping burst read	Incrementing burst read only
Program (Write) Operation	Single 32-bit parallel program operation	Single 32-bit serial program operation

3. Intel MAX 10 UFM Design Considerations

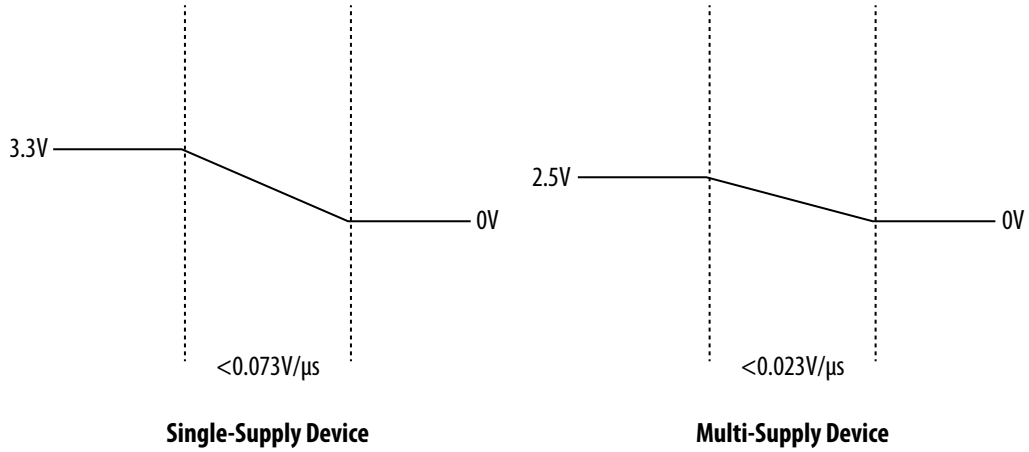
There are several considerations that require your attention to ensure the success of your designs. Unless noted otherwise, these design guidelines apply to all variants of this device family.

3.1. Guideline: UFM Power Supply Requirement

During UFM and CFM operations, make sure to follow the maximum slew rate requirement for power supply ramp down. This setting prevents device damage in case of power loss.

Table 5. Maximum Slew Rate Requirement

Device	Maximum Slew Rate
Single-supply device	0.073V/µs
Multi-supply device	0.023V/µs



3.2. Guideline: Program and Read UFM with JTAG

You can program UFM using JTAG interface version IEEE Standard 1149.1.

The JTAG interface supports Jam™ Standard Test and Programming Language (STAPL) Format File (`.jam`), Programmer Object File (`.pof`), and JAM Byte Code File (`.jbc`).

You can use the Intel Quartus Prime Programmer to program `.pof` through the JTAG interface. To program `.pof`, into the flash, follow these steps:

1. In the **Programmer** window, click **Hardware Setup**, and select **USB Blaster**.
2. In the **Mode** list, select **JTAG**.
3. Click **Auto Detect** on the left pane.
4. Select the device to be programmed, and click **Add File**.
5. Select the `.pof` to be programmed to the selected device.
6. Select the **UFM** in the **Program/Configure** column.
7. Click **Start** to start programming.

To program through `.jam` or `.jbc` files, refer to the *Using the Command-Line Jam STAPL Solution for Device Programming* application note.

Related Information

AN 425: Using the Command-Line Jam STAPL Solution for Device Programming

3.3. Guideline: UFM Content Initialization

You can initialize the UFM content using software.

The initial memory content supports Memory Initialization File (`.mif`), and Hexadecimal (Intel-Format) File (`.hex`).

You can initialize the UFM content using either one of the following ways:

- Set the initial memory content through the On-Chip Flash Intel FPGA IP core.
- Set the initial memory content through the **Convert Programming File** tool in the Intel Quartus Prime software when you convert `.sof` to `.pof`.

For more information about the initialization flash content, refer to the *Programming Files Generation* section in the *Embedded Design Handbook*.

Related Information

Programming Files Generation
Provides more information about the initialization flash content.

3.4. Guideline: Erase Before Program

Make sure to erase the flash location before you perform a program (write) operation.

4. Intel MAX 10 UFM Implementation Guides

Related Information

- Utilizing the User Flash Memory (UFM) on Intel MAX 10 Devices with a Nios II Processor
- Putting MAX Series FPGAs in Hibernation Mode Using User Flash Memory

4.1. On-Chip Flash Intel FPGA IP Core

The IP core design flow helps you get started with any IP core.

The On-Chip Flash Intel FPGA IP core is installed as part of the Intel Quartus Prime installation process. You can select and parameterize any IP core from the Intel FPGA IP library. Intel provides an integrated parameter editor that allows you to customize the On-Chip Flash Intel FPGA IP core to support a wide variety of applications. The parameter editor guides you through the setting of parameter values and selection of optional ports.

Related Information

Introduction to Intel FPGA IP Cores
Provides more information about Intel FPGA IP cores.

4.2. UFM Avalon-MM Operating Modes

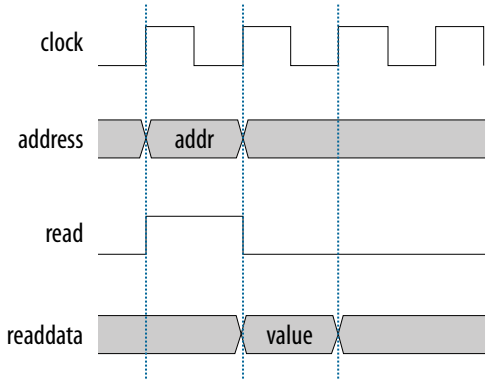
The UFM operating modes use Avalon-MM interface.

4.2.1. UFM Read Status and Control Register

You can access the control register value through the Avalon-MM control slave interface.

Figure 6. Read Status and Control Register

The figure below shows the timing diagram for the read status and control register.



To use the control register, assert the `read` signal and send the control register address to the control slave address.

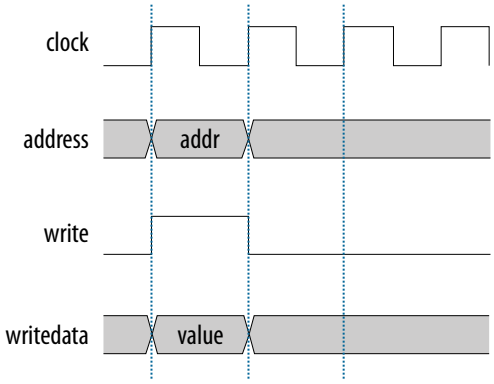
The flash IP core then sends the register value through the `readdata` bus.

4.2.2. UFM Write Control Register

You can program (write) the control register value through Avalon-MM control slave interface.

Figure 7. Program (Write) Control Register

The figure below shows the timing diagram for the program control register.



To program the control register, assert the `write` signal.

The flash IP core then sends address `0×01` (control register) and `writedata` (register value) to control the slave interface.

4.2.3. UFM Program (Write) Operation

The UFM offers a single 32-bit program (write) operation.

To perform a UFM program operation, follow these steps:

1. Disable the write protection mode. Write 0 into the write protection register for the sector of the given data through the Avalon-MM control interface.
2. Program the following data into flash through the Avalon-MM data interface.
   - Address: legal address (from Avalon-MM address map)
   - Data: user data

   Set burst count to 1 (parallel mode) or 32 (serial mode).
3. The flash IP core sets the `busy` field in the status register to `2'b10` when the program operation is in progress.
4. If the operation goes well, the flash IP core sets the write successful field in the status register to `1'b1` or write successful. The flash IP core sets the write successful field in the status register to `1'b0` (failed) if one of the following conditions takes place:
   - The burst count is not equal to 1 (parallel mode) or 32 (serial mode).
   - The given address is out of range.
   - The sector protection mode or write protection mode of the corresponding sector is not clear (the value is not `1'b0`).
5. Repeat the earlier steps if you want to perform another program operation.
6. You have to enable back the write protection mode when the program operation completes. Write 1 into the write protection register for the corresponding sector through the Avalon-MM control interface.

   Note: Check the status register after each write to make sure the program operation is successful (write successful).

Figure 8. Program Operation in Parallel Mode

The figure below shows the write data timing diagram in parallel mode.

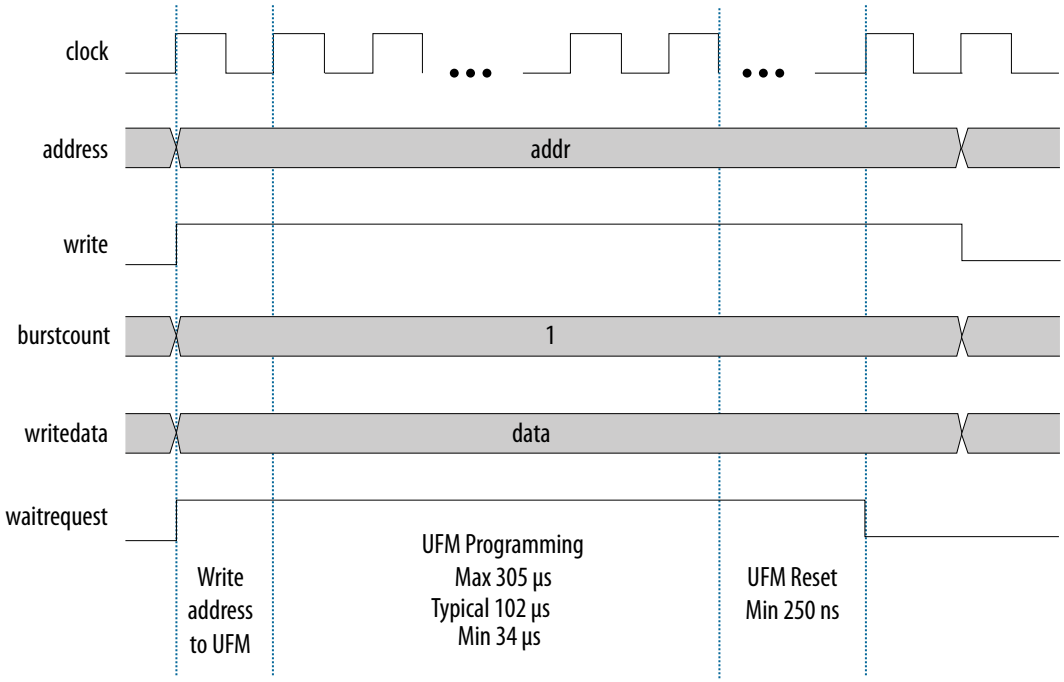
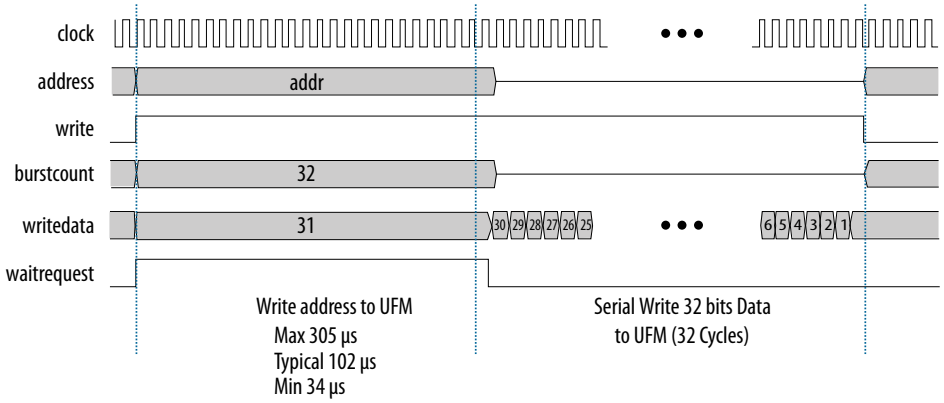


Figure 9. Program Operation in Serial Mode

The figure below shows the write data timing diagram in serial mode.



4.2.4. UFM Sector Erase Operation

The sector erase operation allows the UFM to erase by sectors.

To perform a UFM sector erase operation, follow these steps:

1. Disable the write protection mode. Write 0 into the write protection register for the sector through the Avalon-MM control interface.
2. Write the appropriate bits into the control register to select the sector erase location. The flash IP core stores the sector erase address and initiates the sector erase operation.

Note: The IP core only accepts the sector erase address when it is in IDLE state; `busy` field at status register is `2'b00`. If the IP core is busy, it will ignore the sector erase address.

3. The flash IP core sets the `busy` field in the status register to `2'b01` when the erase operation is in progress.
4. The flash IP core then asserts the `waitrequest` signal if there are any new incoming read or write commands from the data interface.
5. The flash IP core erases the sector. It stores the physical flash erase result in the erase successful field in the status register when the sector erase operation completes.

   Note: The maximum erase time is 350 ms.
6. The flash IP core sets the erase successful field in the status register to `1'b0` (failed) if one of the following conditions takes place:
   - You send an illegal sector number.
   - The sector protection mode or write protection mode of the corresponding sector is not clear (the value is not `1'b0`).
7. Repeat the earlier steps if you want to perform another sector erase operation.
8. You have to enable back the write protection mode when the sector erase operation completes. Write 1 into the write protection register for the corresponding sector through the Avalon-MM control interface.

   Note: Check the status register after each erase to make sure the erase operation is successful (erase successful).

4.2.5. UFM Page Erase Operation

The page erase operation allows the UFM to erase by pages.

To perform a UFM page erase operation, follow these steps:

1. Disable the write protection mode. Write 0 into the write protection register for the sector through the Avalon-MM control interface.
2. Write the appropriate bits into the control register to select the page erase location. The flash IP core stores the page erase address and initiates the page erase operation.

   Note: The IP core only accepts the page erase address when the IP is in IDLE state; `busy` field at status register is `2'b00`. If the IP core is busy, it will ignore the page erase address.
3. The flash IP core sets the `busy` field in the status register to `2'b01` when the erase operation is in progress.
4. The flash IP core then asserts the `waitrequest` signal if there are any new incoming read or write commands from the data interface.
5. The flash IP core erases the page. It stores the physical flash erase result in the erase successful field in the status register when the page erase operation completes.

   Note: The maximum erase time is 350 ms.

6. The flash IP core sets the erase successful field in the status register to `1b'0` (failed) if you send an illegal address.
7. Repeat the earlier steps if you want to perform another page erase operation.
8. You have to enable back the write protection mode when the page erase operation completes. Write 1 into the write protection register for the corresponding page through the Avalon-MM control interface.

   Note: Check the status register after each erase to make sure the erase operation is successful (erase successful).

4.2.6. UFM Read Operation

The UFM offers a single 32-bit read operation.

To perform a read operation, the address register must be loaded with the reference address where the data is or is going to be located in the UFM.

To perform a UFM read operation, follow these steps:

1. Assert the `read` signal to send the legal data address to the data slave interface.
2. Set the burst count to 1 (parallel mode) or 32 (serial mode).
3. The flash IP core asserts the `waitrequest` signal when it is busy.
4. The flash IP core asserts the `readdatavalid` signal and sends the data through the `readdata` bus.
5. The flash IP core sets the `busy` field in the status register to `2'b11` when the read operation is in progress.
6. If the operation goes well, the flash IP core sets the read successful field in the status register to `1'b1` or read successful. It sets the read successful field in the status register to `1'b0` (failed) and returns empty flash if you try to read from an illegal address or protected sector.

The following figures show the timing diagrams for the read operations for the different Intel MAX 10 devices in parallel and serial modes.

Figure 10. Read Operation for 10M04, 10M08, 10M16 and 10M25 Devices in Parallel Mode

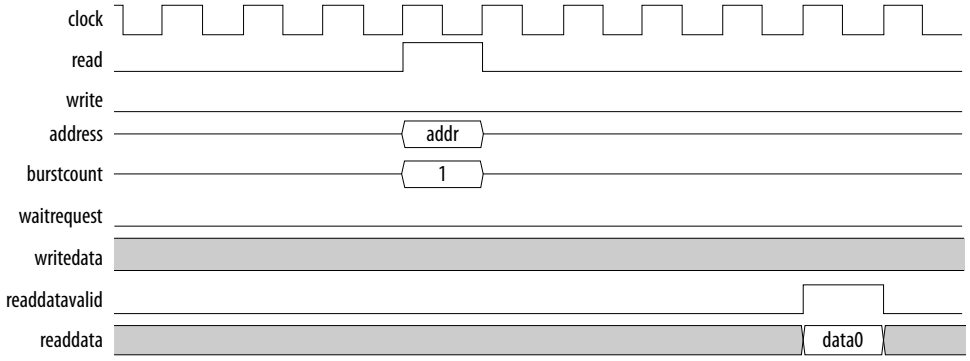


Figure 11. Read Operation for 10M40 and 10M50 Devices in Parallel Mode

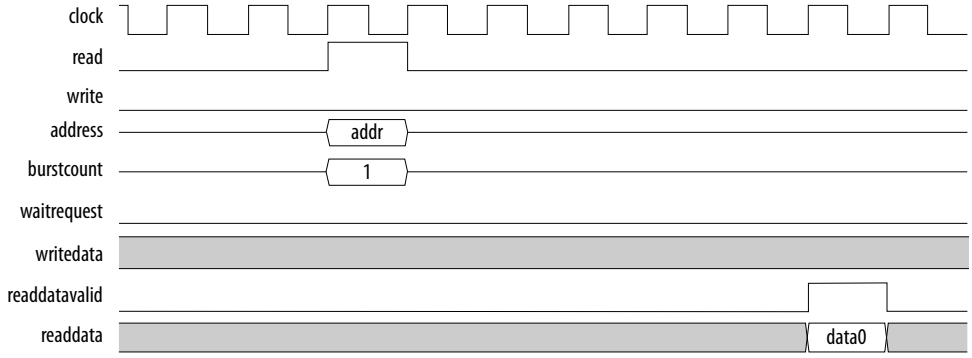
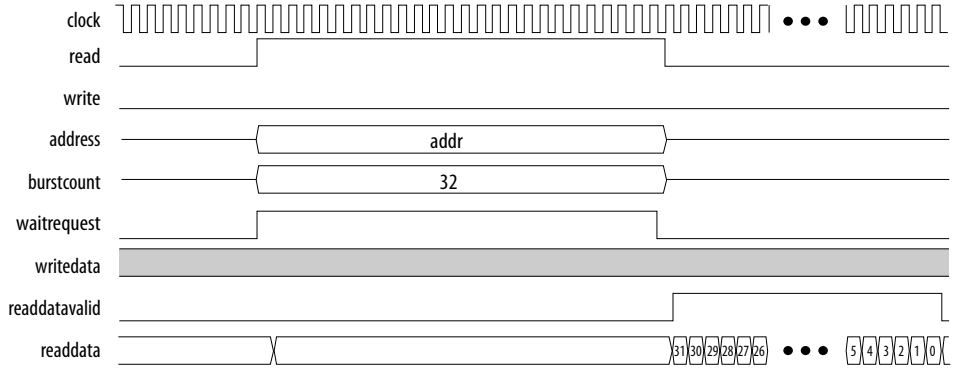


Figure 12. Read Operation for Intel MAX 10 Devices in Serial Mode



4.2.7. UFM Burst Read Operation

The burst read operation is a streaming 32-bit read operation.

The burst read operation offers the following modes:

- Data incrementing burst read—allows a maximum of 128 burst counts.
- Data wrapping burst read—has fixed burst counts of 2 (10M04/08) and 4 (10M16/25/40/50)

To perform a UFM burst read operation, follow these steps:

1. Assert the `read` signal and send the legal burst count and legal data addresses to the data interface.
2. The flash IP core asserts the `waitrequest` signal when it is busy.
3. The flash IP core then asserts the `readdatavalid` signal and sends the data through the `readdata` bus.

Note: For data wrapping burst read operation, if the address reaches the end of the flash, it wraps back to the beginning of the flash and continues reading.

4. The flash IP core sets the `busy` field in the status register to `2'b11` or `busy_read` when the read operation is in progress.
5. If the operation goes well, the flash IP core sets the read successful field in the status register to `1'b1` or read successful. It sets the read successful field in the status register to `1'b0` (failed) and changes empty flash to 1 if you try to read from an illegal address or protected sector.

4.2.7.1. UFM Data Incrementing Burst Read

The following figures show the timing diagrams for the data incrementing burst read operations for the different Intel MAX 10 devices.

Figure 13. Incrementing Burst Read Operation for 10M02, 10M04, and 10M08 Devices in Parallel Mode

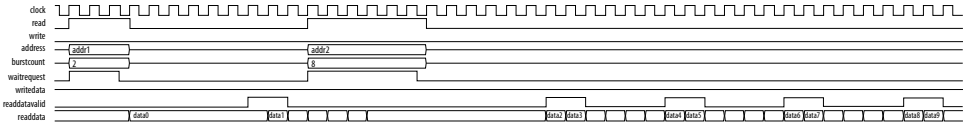


Figure 14. Incrementing Burst Read Operation for 10M16 and 10M25 Devices in Parallel Mode

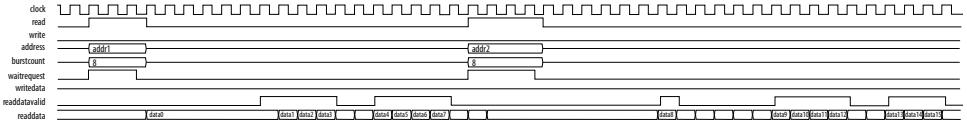


Figure 15. Incrementing Burst Read Operation for 10M50 Devices in Parallel Mode

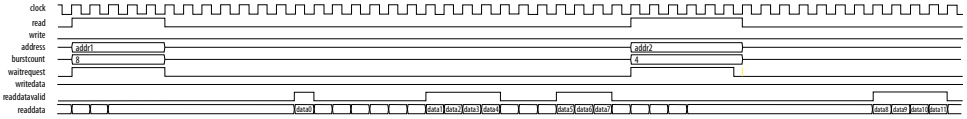


Figure 16. Unaligned Address Incrementing Burst Read Operation for 10M50 Devices in Parallel Mode

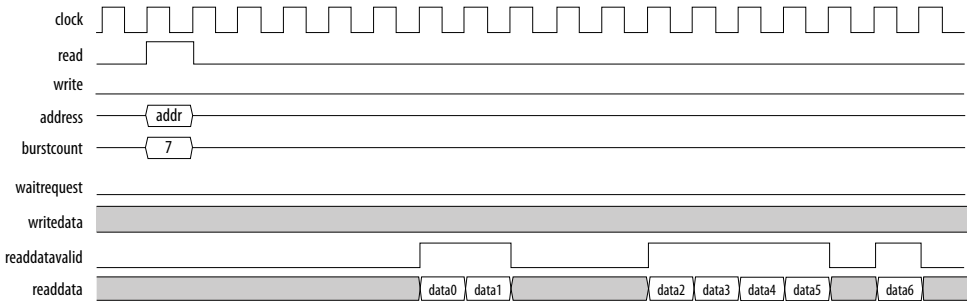
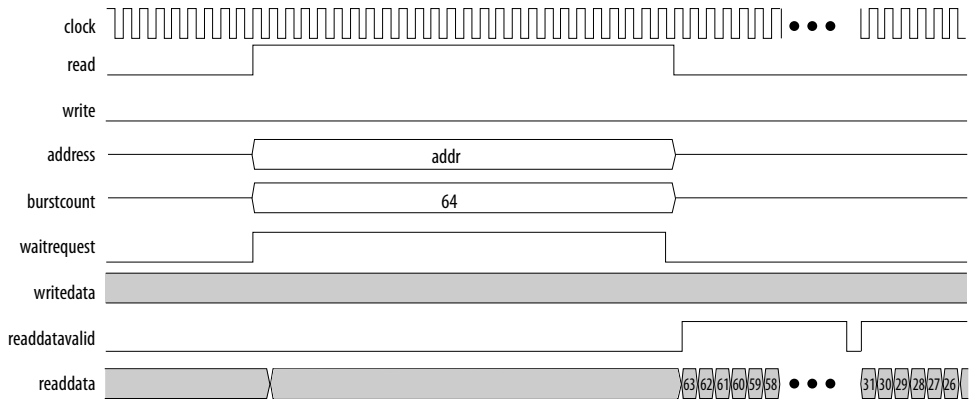


Figure 17. Incrementing Burst Read Operation for Intel MAX 10 Devices in Serial Mode



4.2.7.2. UFM Data Wrapping Burst Read

The UFM supports data wrapping when it receives an unaligned address.

Note: Wrapping burst read is available only for parallel interface.

Table 6. Data Wrapping Support for Intel MAX 10 Devices

Device	Data Register Length	Flash IP Data Bus Width	Fixed Supported Burst Count	Data Wrapping
10M04, or 10M08	32	64	2	The address wraps back to the previous boundary after 64 bits or 2 cycles. For example, for a wrapping in a 32-bit data interface: 1. Start address is 0×01 2. Address sequence will be 0×01, then back to address 0×00
10M16, 10M25, 10M40, or 10M50	32	128	4	The address wraps back to the previous boundary after 128 bits or 4 cycles. For example, for a wrapping in a 32-bit data interface: 1. Start address is 0×02 2. Address sequence will be 0×02 and 0×03, then back to address 0×00 and 0×01

The following figures show the timing diagrams for the data wrapping burst read operations for the different Intel MAX 10 devices.

Figure 18. Wrapping Burst Read Operation for 10M04 and 10M08 Devices

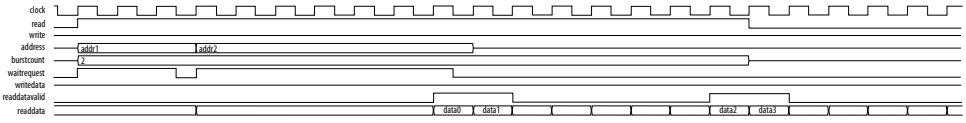


Figure 19. Wrapping Burst Read Operation for 10M16 and 10M25 Devices

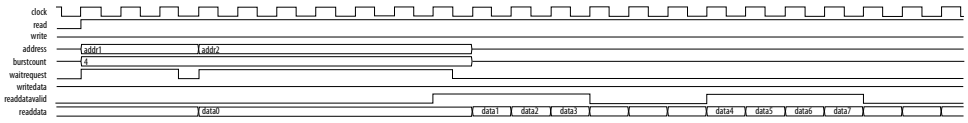
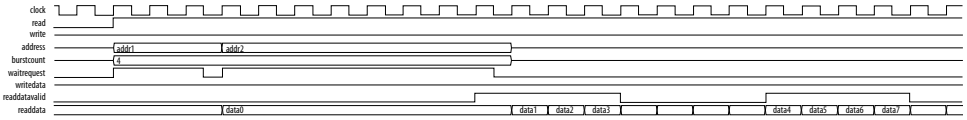


Figure 20. Wrapping Burst Read Operation for 10M40 and 10M50 Devices



4.3. Flash Initialization Files

The On-Chip Flash Intel FPGA IP core supports the `.hex`, `.mif`, and `.dat` files.

If the total data size in the initialization file is less the maximum UFM size, the IP core retains blank data (all 1's).

If the total data size in the initialization file is larger than the maximum UFM size, the IP core ignores the extra data.

Table 7. Types of Flash Initialization File Supported

File Type	Format	Notes
`.hex`	Standard Intel hexadecimal file—uses byte addressing.	For flash initialization in actual hardware.
`.mif`	Standard Intel FPGA memory initialization file—uses word addressing.	For flash initialization in actual hardware.
`.dat`	32-bit data width file—uses word addressing.	For flash initialization in simulation model.

5. On-Chip Flash Intel FPGA IP Core References

This section provides information about the On-Chip Flash Intel FPGA IP core parameters, signals, and registers.

5.1. On-Chip Flash Intel FPGA IP Core Parameters

The following table lists the parameters for the On-Chip Flash Intel FPGA IP core.

Table 8. On-Chip Flash Intel FPGA IP Core Parameters

Parameters	Default Value	Description
Data interface	Parallel	Allows you to select the type of interface. You can choose parallel or serial.
Read burst mode	Incrementing	Allows you to select the type of read burst mode. You can choose incrementing or wrapping. Incrementing mode: Read burst count is 2, 4, 8, ... 128 Wrapping mode: Burst count fixed to 2 or 4 *Note:* Serial interface supports only incrementing mode. Parallel interface does not support wrapping mode for 10M02 devices.
Read burst count	2	Allows you the flexibility to adjust the maximum burst count bus width. • Parallel mode: This setting represents the maximum burst count number. • Serial mode: This setting supports stream read and represents the words to be read for each read operation. The Avalon-MM interface burst count bus width is equal to 32*read burst count.
Configuration mode	Single uncompressed image	Allows you to select the configuration mode. You can choose one of these options: • Dual compressed images • Single uncompressed image: Accesses CFM2 sector as UFM • Single compressed image: Accesses CFM2 and CFM1 sectors as UFM • Single uncompressed image with memory initialization • Single compressed image with memory initialization
Flash Memory	—	The sector ID, address range value, and flash type are generated dynamically by hardware `.tcl` based on the device and configuration mode you select. Indicates the address mapping for each sector and adjusts the **Access Mode** for each sector individually. *Note:* Only CFM sectors support **Hidden** access mode.

continued...

Parameters	Default Value	Description
Clock frequency	116.0 MHz	Key in the appropriate clock frequency in MHz. The maximum frequency is 116.0 MHz for parallel interface and 7.25 MHz for serial interface. *Note:* If you use 10M02 devices, the maximum frequency for parallel interface is 7.25 MHz.
Initialize flash content	Off	Turn on this option to initialize the flash content.
Enable non-default initialization file	Off	Turn on this option to enable your preferred initialization file. If you choose to have a non-default file, type the filename or select the `.hex` or `.mif` file using the browse button.
User created hex or mif file	—	This option is only available if you turn on **Enable non-default initialization file**. Assign your own `.hex` or `.mif` filename.
User created dat file for simulation	—	This option is only available if you turn on **Enable non-default initialization file**. Assign your own simulation filename.

5.2. On-Chip Flash Intel FPGA IP Core Signals

The following table lists the signals for the On-Chip Flash Intel FPGA IP core.

Table 9. Avalon-MM Slave Input and Output Signals for Parallel and Serial Modes.

Signal	Width	Direction	Description
Clock and Reset			
`clock`	1	Input	System clock signal that clocks the entire peripheral.
`reset_n`	1	Input	System synchronous reset signal that resets the entire peripheral. The IP core asserts this signal asynchronously. This signal becomes synchronous in the IP core after the rising edge of the clock.
Control			
`avmm_csr_addr`	1	Input	Avalon-MM address bus that decodes registers.
`avmm_csr_read`	1	Input	Avalon-MM read control signal. The IP core asserts this signal to indicate a read transfer. If present, the `readdata` signal is required.
`avmm_csr_readdata`	32	Output	Avalon-MM read back data signal. The IP core asserts this signal during read cycles.
`avmm_csr_write`	1	Input	Avalon-MM write control signal. The IP core asserts this signal to indicate a write transfer. If present, the `writedata` signal is required.
`avmm_csr_writedata`	32	Input	Avalon-MM write data bus. The bus master asserts this bus during write cycles.
Data			
`avmm_data_addr`	User-defined	Input	Avalon-MM address bus that indicates the flash data address. The width of this address depends on your selection of device and configuration mode.
`avmm_data_read`	1	Input	Avalon-MM read control signal. The IP core asserts this signal to indicate a read transfer. If present, the `readdata` signal is required.

continued...

Signal	Width	Direction	Description
avmm_data_readdata	• Parallel mode: 32 • Serial mode: 1	Output	Avalon-MM read back data signal. The IP core asserts this signal during read cycles.
avmm_data_write	1	Input	Avalon-MM write control signal. The IP core asserts this signal to indicate a write transfer. If present, the writedata signal is required.
avmm_data_writedata	• Parallel mode: 32 • Serial mode: 1	Input	Avalon-MM write data bus. The bus master asserts this bus during write cycles.
avmm_data_waitrequest	1	Output	The IP core asserts this bus to pause the master when the IP core is busy during read or write operations.
avmm_data_readdatavalid	1	Output	The IP core asserts this signal when the readdata signal is valid during read cycles.
avmm_data_burstcount	User-defined	Input	The bus master asserts this signal to initiate a burst read operation. • In write operations, the burst count is always fixed to 1 for parallel mode and 32 for serial mode. • In incrementing burst read mode, the supported read burst count range: Parallel mode: $1\text{-}2^{(\text{burstcount width-1})}$ Serial mode: 1-128*32 • In wrapping burst read mode (parallel mode only), the supported read burst count is fixed to 2 and 4. 10M04, and 10M08: 1–2 10M16, 10M25, 10M40 and 10M50: 1–4

5.3. On-Chip Flash Intel FPGA IP Core Registers

The following table lists the address mapping and registers for the On-Chip Flash Intel FPGA IP core.

Table 10. On-Chip Flash Intel FPGA IP Core Control Address Mapping

Register	Address	Access	Description
Status Register	0×00	Read only	Stores the status and result of recent operations and sector protection mode.
Control Register	0×01	Read/Program	Stores the following information: • Page erase address • Sector erase address • Sector write protection mode

Table 11. On-Chip Flash Intel FPGA IP Core Status Register

Bit Offset	Field	Default Value	Description
1–0	busy	2'b00	2'b00 IDLE 2'b01 BUSY_ERASE 2'b10 BUSY_WRITE

continued...

Bit Offset	Field	Default Value	Description
			2'b11 BUSY_READ
2	rs (read successful)	1'b0	1'b0 Read failed 1'b1 Read successful
3	ws (write successful)	1'b0	1'b0 Write failed 1'b1 Write successful
4	es (erase successful)	1'b0	1'b0 Erase failed 1'b1 Erase successful
5	sp (Sector ID 1 protection bit)	—	The IP core sets these bits based on the device, and configuration and access mode settings you specify during instantiation. These settings are fixed. If the IP core sets one of these bits, you cannot read or program on the specified sector.
6	sp (Sector ID 2 protection bit)	—	
7	sp (Sector ID 3 protection bit)	—	
8	sp (Sector ID 4 protection bit)	—	
9	sp (Sector ID 5 protection bit)	—	
31–10	dummy (padding)	—	All of these bits are set to 1.

Table 12. On-Chip Flash Intel FPGA IP Core Control Register

<table>
<tr><th>Bit Offset</th><th>Field</th><th>Default Value</th><th>Description</th></tr>
<tr><td>19–0</td><td>pe (page erase address)</td><td>All 1's</td><td>Sets the page erase address to initiate a page erase operation. The IP core only accepts the page erase address when it is in IDLE state. Otherwise, the page address will be ignored.
The legal value is any available address. The IP core erases the corresponding page of the given address.</td></tr>
<tr><td>22–20</td><td>se (sector erase address)</td><td>3'b111</td><td>Sets the sector erase address to initiate a sector erase operation. The IP core only accepts the sector erase address when it is in IDLE state. Otherwise, the page address will be ignored.
<table>
<tr><td>3'b001</td><td>Sector ID 1</td></tr>
<tr><td>3'b010</td><td>Sector ID 2</td></tr>
<tr><td>3'b011</td><td>Sector ID 3</td></tr>
<tr><td>3'b100</td><td>Sector ID 4</td></tr>
<tr><td>3'b101</td><td>Sector ID 5</td></tr>
<tr><td>Other values</td><td>Illegal address</td></tr>
</table>
If the device you selected has only 3 sectors, the value mapped to sectors ID 4 and 5 will become illegal address.
Note: If you set both sector address and page address at the same time, the sector erase address gets the priority. The IP core accepts and executes the sector erase address and ignores the page erase address.
For more detailed description, refer to Sector Address on page 25.</td></tr>
</table>

continued...

Bit Offset	Field	Default Value	Description
23	wp (Sector ID 1 write protection)	1	The IP core uses these bits to protect the sector from write and erase operation. You must clear the corresponding sector write protection bit before your program or erase the sector. 1'b0: Disable write protected mode 1'b1: Enable write protected mode
24	wp (Sector ID 2 write protection)	1	
25	wp (Sector ID 3 write protection)	1	
26	wp (Sector ID 4 write protection)	1	
27	wp (Sector ID 5 write protection)	1	
31–28	dummy (padding)	—	All of these bits are set to 1.

5.3.1. Sector Address

You need to convert the sector address in the parameter editor to 32-bit address.

The address mapping in the parameter editor uses byte address. The Avalon-MM interface in the On-Chip Flash Intel FPGA IP core uses 32-bit address.

Table 13. Address Mapping Example

Sector	Parameter Editor Address	Avalon-MM Address
Sector ID 1	0x0000–0x17ff	0x000–0x5ff
Sector ID 2	0x1800–0x2fff	0x600–0xBff
Sector ID 3	0x3000–0x13fff	0x0C00–0x4fff

6. Intel MAX 10 User Flash Memory User Guide Archive

If an IP core version is not listed, the user guide for the previous IP core version applies.

IP Core Version	User Guide
16.0	MAX 10 User Flash Memory User Guide
15.1	MAX 10 User Flash Memory User Guide
15.0	MAX 10 User Flash Memory User Guide
14.1	MAX 10 User Flash Memory User Guide

7. Document Revision History for the Intel MAX 10 User Flash Memory User Guide

Document Version	Intel Quartus Prime Version	Changes
2019.07.02	18.0	Updated the *Program Operation in Serial Mode* figure to provide the minimum, typical, and maximum values for the write address to UFM.
2018.12.28	18.0	Added 10M02 device support to the *Incrementing Burst Read Operation for 10M02, 10M04, and 10M08 Devices in Parallel Mode* figure.
2018.06.29	18.0	• Beginning from the Intel Quartus Prime software version 18.0, the name of this IP core has been changed from Altera On-Chip Flash IP core to On-Chip Flash Intel FPGA IP core. • Added a reference to the *Programming Files Generation* section of the *Embedded Design Handbook* for the initialization flash content. • Updated the *Incrementing Burst Read Operation for 10M04 and 10M08 Devices in Parallel Mode* figure. • Updated the *Incrementing Burst Read Operation for 10M16 and 10M25 Devices in Parallel Mode* figure. • Updated the *Incrementing Burst Read Operation for 10M50 Devices in Parallel Mode* figure. • Updated the *Wrapping Burst Read Operation for 10M04 and 10M08 Devices* figure. • Updated the *Wrapping Burst Read Operation for 10M16 and 10M25 Devices* figure. • Updated the *Wrapping Burst Read Operation for 10M40 and 10M50 Devices* figure.

Date	Version	Changes
February 2017	2017.02.21	Rebranded as Intel.
December 2016	2016.12.20	• Updated the description for Altera On-Chip Flash bit offsets 5–9 that the IP core sets these bits based on the device, and configuration and access mode settings you specify during instantiation. These settings are fixed. • Updated the description for Altera On-Chip Flash bit offsets 22–27 to include clearer information about sector address. • Added *Sector Address* topic that provides details about converting sector address from byte addressing to bit addressing.
May 2016	2016.05.02	• Added the typical and minimum UFM programming time in parallel mode. • Corrected the minimum UFM reset time in parallel mode to 250 ns. • Added links to archived versions of the *MAX 10 User Flash Memory User Guide*.
November 2015	2015.11.02	• Added information about the supported flash initialization files. • Added serial interface support for 10M40 and 10M50 devices. The maximum frequency for MAX 10 devices is 7.25 MHz, except for 10M40 and 10M50 devices, which is 4.81 MHz. • Added parallel interface support for 10M02 devices. The maximum frequency for MAX 10 devices is 116 MHz, except for 10M02 devices, which is 7.25 MHz. • Changed instances of *Quartus II* to *Intel Quartus Prime*.

continued...

Date	Version	Changes
May 2015	2015.05.04	• Changed *write* to industry-standard term *program*. • Added a note to the *UFM and CFM Array Size* section that the total UFM size is the maximum possible value, which is dependent on the selected mode. • Added design consideration information about the maximum slew rate requirement for power supply ramp down. • Added design consideration information about erasing the flash location before performing a program operation.
December 2014	2014.12.15	• Added support for serial interface. • Added maximum operating frequency of 7.25 MHz for serial interface. • Updated the UFM block diagram to include serial interface. • Added design consideration information about creating initial memory content using the IP core, and programming UFM using JTAG interface version IEEE Standard 1149.1. • Added new timing diagrams for read and write operations in serial mode. • Added information for the new serial interface related GUI parameters, signals, and registers. • Added information for the following new Avalon-MM slave interface signals for serial mode: `addr`, `read`, `readdata`, `write`, `writedata`, `waitrequest`, `readdatavalid`, and `burstcount`. • Added information for the following new parameters: — **Data Interface** that allows you to choose between **Parallel** and **Serial** interface. — **Configuration Scheme** and **Configuration Mode** that replace **Dual Images**. The new parameters include all supported configuration modes. — **Read Burst Count** that allows the burst count width to be auto-adjusted.
September 2014	2014.09.22	Initial release.