Nios II Processor Booting Methods In MAX 10 FPGA Devices

2015.06.15

AN-730 Subscribe Send Feedback

Overview

MAX® 10 FPGA devices are the first MAX device series that can support Nios® II processor.

MAX 10 FPGA devices contain on-chip flash that is segmented into two types:

- Configuration Flash Memory (CFM)—to store hardware configuration data for MAX 10 FPGA.
- User Flash Memory (UFM)—to store user data or software applications.

You can configure the Nios II processor to boot and execute software from different memory locations, including the MAX10 FPGA on chip RAM and UFM.

This document gives an overview of the Altera On-chip Flash IP core and describes the various boot or software execution options available with the Nios II processor and MAX 10 FPGAs.

Abbreviations

Table 1: List of Abbreviations

Abbreviation	Description
CFM	Configuration Flash Memory
ERAM	Embedded Random Access Memory
HEX	Hexadecimal File(1)
memcpy	Memory copy
OCRAM	On-Chip RAM
POF	Programmer Object File
QSPI	Quad Serial Parallel Interface
RAM	Random Access Memory
SBT	Software Build Tools
SOF	SRAM Object File

(1) This is an ASCII text file with the extension of .hex which stores the initial memory values for a memory block.

Abbreviation	Description
SPI	Serial Parallel Interface
UART	Universal asynchronous receiver/transmitter
UFM	User Flash Memory
XIP	Execute In Place

Prerequisite

You are required to have the knowledge of instantiating and developing a Nios II processor based system. Altera recommends you to go through the online tutorials and training materials provided at **http://www.altera.com/education/edu-index.html** before using this application note.

Related Information

- **Nios II Gen2 Hardware Development Tutorial.**
  A step by step procedure to build a Nios II Gen2 soft core processor system.
- **Getting Started with the Graphical User Interface.**
  This document provides the details of Nios II Software Build Tools using graphical user interface.

The MAX 10 FPGA On-chip Flash Overview

The MAX 10 FPGA On-chip Flash consists of two flash regions with the functionality shown in the following table.

Table 2: On-chip Flash Regions in MAX 10 FPGA Devices

Flash Regions	Functionality
Configuration Flash Memory (sectors CFM0-2)	FPGA configuration file storage
User Flash Memory (sectors UFM0-1)	Nios II processor application and/or user data

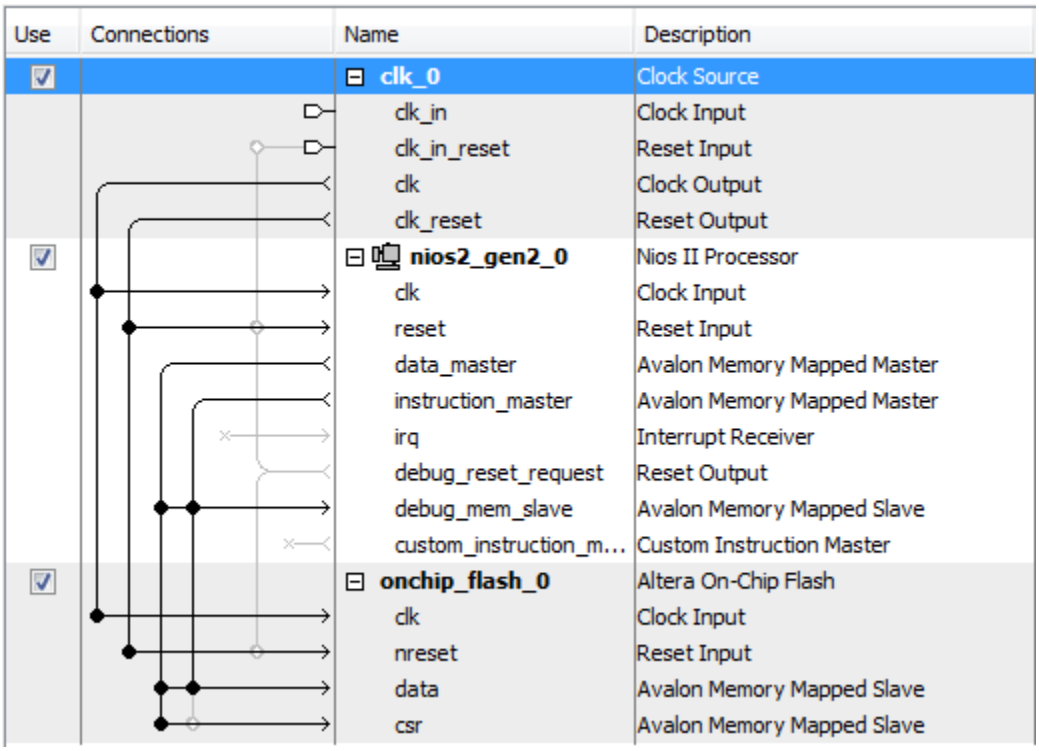
MAX 10 FPGA devices support several different configuration modes and some of these modes allow CFM1 and CFM2 to be used as an additional UFM region. The following table shows the storage location of the FPGA configuration images based on the MAX 10 FPGA's internal configuration modes. You can refer to the MAX 10 FPGA Configuration User Guide for more information on the configuration mode supported in MAX 10 FPGA.

Table 3: Storage Location of FPGA Configuration Images with Different Configuration Modes

Internal Configuration Mode	CFM2[2]	CFM1[2]	CFM0
Dual images	Compressed Image 2		Compressed Image 1
Single uncompressed image	UFM[3]	Uncompressed image	
Single uncompressed image with Memory Initialization	Uncompressed image (with pre-initialized on-chip memory content)		
Single compressed image with Memory Initialization	Compressed image (with pre-initialized on-chip memory content)		
Single compressed image	UFM[3]		Compressed Image

You must use the Altera On-chip Flash IP core to access to the flash memory in MAX 10 FPGAs. You can instantiate and connect the Altera On-chip Flash IP to the Nios II processor using the Qsys system design tool in Quartus II software. The Nios II soft core processor uses the Avalon® Memory-Mapped (Avalon-MM) interface to communicate with the Altera On-chip Flash IP.

Figure 1: Example of Connections for The Altera On-chip Flash IP and the Nios II Gen2 Soft Core Processor



(2) This sector is NOT supported in 10M02 device.
(3) CFM sector is configured as virtual UFM.

Related Information

- **MAX 10 FPGA Configuration User Guide**
- **MAX 10 User Flash Memory User Guide**

Altera On-chip Flash IP Architecture and Features

The Altera On-chip Flash IP core can provide access to five flash sectors:

- UFM0
- UFM1
- CFM0
- CFM1
- CFM2

Important facts about UFM and CFM sectors:

- CFM sectors are intended for configuration (bitstream) data (*.pof) storage.
- You can store user data in the UFM sectors.
- Certain devices do not have a UFM1 sector. You can refer to **Table 4** for available sectors in each individual MAX 10 FPGA device.
- You can configure CFM2 as a virtual UFM by selecting "Single Uncompressed Image" configuration mode.
- You can configure CFM2 and CFM1 as a virtual UFM by selecting "Single Compressed Image" configuration mode.
- The size of each sector varies with the selected MAX 10 FPGA devices.

Table 4: UFM and CFM Sector Size

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

Device	Pages per Sector					Page Size (Kbit)	Maximum User Flash Memory Size (Kbit)(4)	Total Configuration Memory Size (Kbit)	OCRAM Size (Kbit)
	UFM1	UFM0	CFM2	CFM1	CFM0				
10M02	3	3	0	0	34	16	96	544	108
10M04	0	8	41	29	70	16	1248	2240	189
10M08	8	8	41	29	70	16	1376	2240	378
10M16	4	4	38	28	66	32	2368	4224	549
10M25	4	4	52	40	92	32	3200	5888	675
10M40	4	4	48	36	84	64	5888	10752	1260
10M50	4	4	48	36	84	64	5888	10752	1638

The Altera On-chip Flash IP core supports the following features:

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

- Read or write accesses to UFM and CFM sectors using the Avalon MM data and control slave interface.
- Supports page erase, sector erase and sector write.
- Simulation model for UFM read / write accesses using various EDA simulation tool.

ERAM Preload Option

The FPGA configuration data may contain MAX 10 FPGA On-chip RAM or ERAM initialization data. The ERAM preload occurs during FPGA configuration before the device enters user mode. The ERAM preload option allows initialization data for the On-chip RAM to be stored in the CFM sectors, this could be any type of application data, including Nios II soft core processor software.

All MAX 10 FPGA devices except for the MAX 10 10M02 device can support dual FPGA configuration images. This however requires the ERAM preload to be set to OFF, in order to reduce FPGA configuration image size and ensure that the images will fit into the CFM.

When the ERAM preload feature is set to OFF, features that require initialization of on-chip RAM will not work. The ERAM preload option is set to OFF by default.

Selecting Single Compressed Image with Memory Initialization or Single Uncompressed Image with Memory Initialization configuration mode will enable the ERAM Preload option but will reduce the size of UFM available.

Nios II Processor Boot Options

Table 5: Summary of Nios II Processor Boot Options

Boot Option	Application Code Stored Location	Application Runtime Location	Boot Method
Option 1: Nios II processor application executes in-place from Altera On-chip Flash (UFM)	UFM	UFM (XIP) + OCRAM/ External RAM (for data)	Using the alt_load () function
Option 2: Nios II processor application copied from UFM to RAM using boot copier	UFM	OCRAM/ External RAM	Using memcpy-based boot copier
Option 3: Nios II processor application executes in-place from Altera On-chip Memory (OCRAM)	OCRAM	OCRAM	No boot copier is required

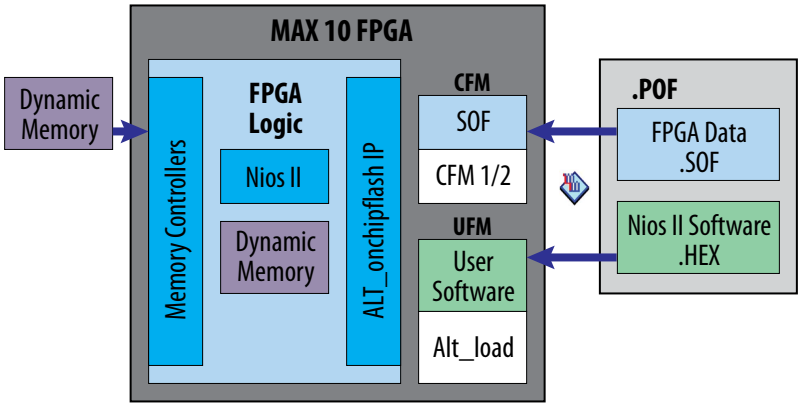
Option 1: Nios II Processor Application Executes In-place From Altera On-chip Flash (UFM)

This solution is suitable for Nios II processor applications which require limited on-chip memory usage. The alt_load() function operates as a mini boot copier which copies the data sections (.rodata, .rwdata, or .exceptions) from boot memory to RAM based on the BSP settings. The code section (.text), which is a read only section, remains in the Altera On-chip Flash memory region. This helps to minimize the RAM usage but may limit the code execution performance as access to the flash memory is slower than the on-chip RAM.

The Nios II processor application is programmed into the UFM sector. The Nios II processor reset vector points to the UFM sector in order to execute code from the UFM after the system resets.

If you are debugging the application using the source-level debugger, you must use a hardware breakpoint to debug because the UFM does not support random memory access; which is required for soft breakpoint debug.

Figure 2: Boot Option 1 Block Diagram



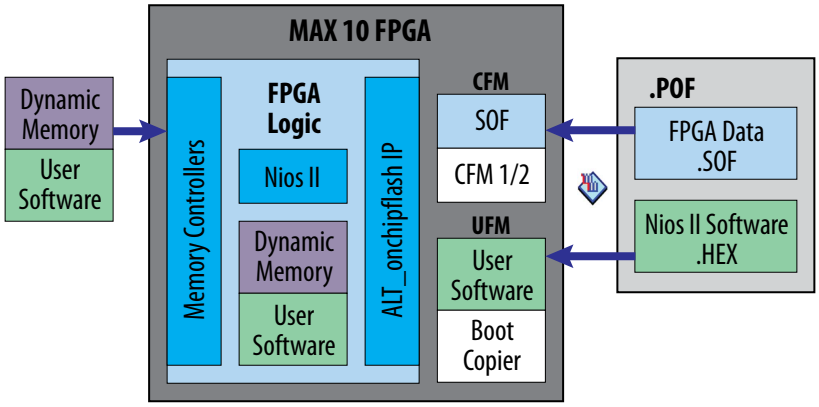
Option 2: Nios II Processor Application Copied From UFM To RAM Using Boot Copier

Altera recommends this solution for MAX 10 FPGA Nios II processor system designs where there may be multiple iterations of application software development and when high system performance is required. The boot copier is located within the UFM at an offset which is the same address with the reset vector. The Nios II application is located after the boot copier.

For this boot option, Nios II processor starts executing the boot copier software upon system reset to copy the application from the UFM sector to the OCRAM/external RAM. Once this is complete, the Nios II processor transfers the program control over to the application.

Note: The OCRAM size is limited, you have to ensure that the size is sufficient for application execution. Refer to **Table 7** for more information.

Figure 3: Boot Option 2 Block Diagram



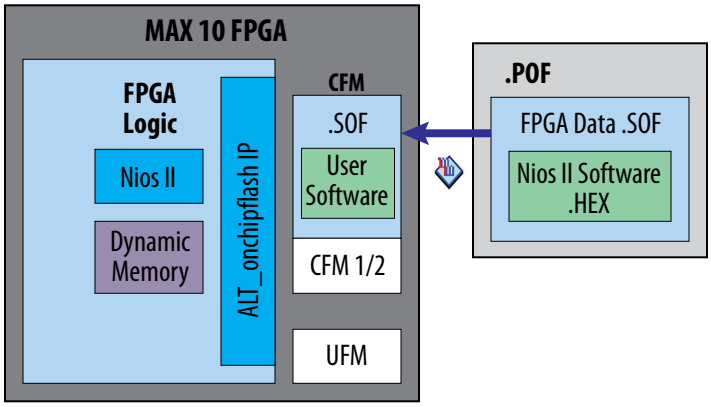
Option 3: Nios II Processor Application Executes In-place from Altera On-chip Memory (OCRAM)

The On-chip Memory is initialized during FPGA configuration with data from a Nios II application image. This data is built into the FPGA configuration bitstream, the programmer object file (POF). This eliminates the need for a boot copier, as the Nios II application is already in place at system reset.

However, this option will not work in any of the following situations:

- When configuration mode with non memory initialization is selected.
- After a soft reset where the memory contents have been modified by the application and the application code has been corrupted.

Figure 4: Boot Option 3 Block Diagram



Nios II Soft Core Processor Configuration and Boot Flow

There are three guidelines to build a bootable system based on the MAX 10 FPGA configuration mode. You can choose any of the booting options supported. Refer to the correct guideline in the following table based on the boot option and the configuration mode.

Table 6: Nios II Processor Boot Options and MAX 10 FPGA Configuration Modes

Boot Option	Supported Configuration Mode	Guideline to Use
• Option 1: Nios II processor application executes in-place from Altera On-chip Flash (UFM). • Option 2: Nios II processor application copied from UFM to RAM using boot copier.	• Single uncompressed image • Single compressed image	Single Uncompressed/Compressed Image Bootable System Guideline
	• Dual Compressed Images	Dual Compressed Images Bootable System Guideline
• Option 3: Nios II processor application executes in-place from Altera On-chip Memory (OCRAM).	• Single uncompressed image with Memory Initialization • Single compressed image with Memory Initialization	Single Uncompressed/Compressed Image with Memory Initialization Bootable System Guideline

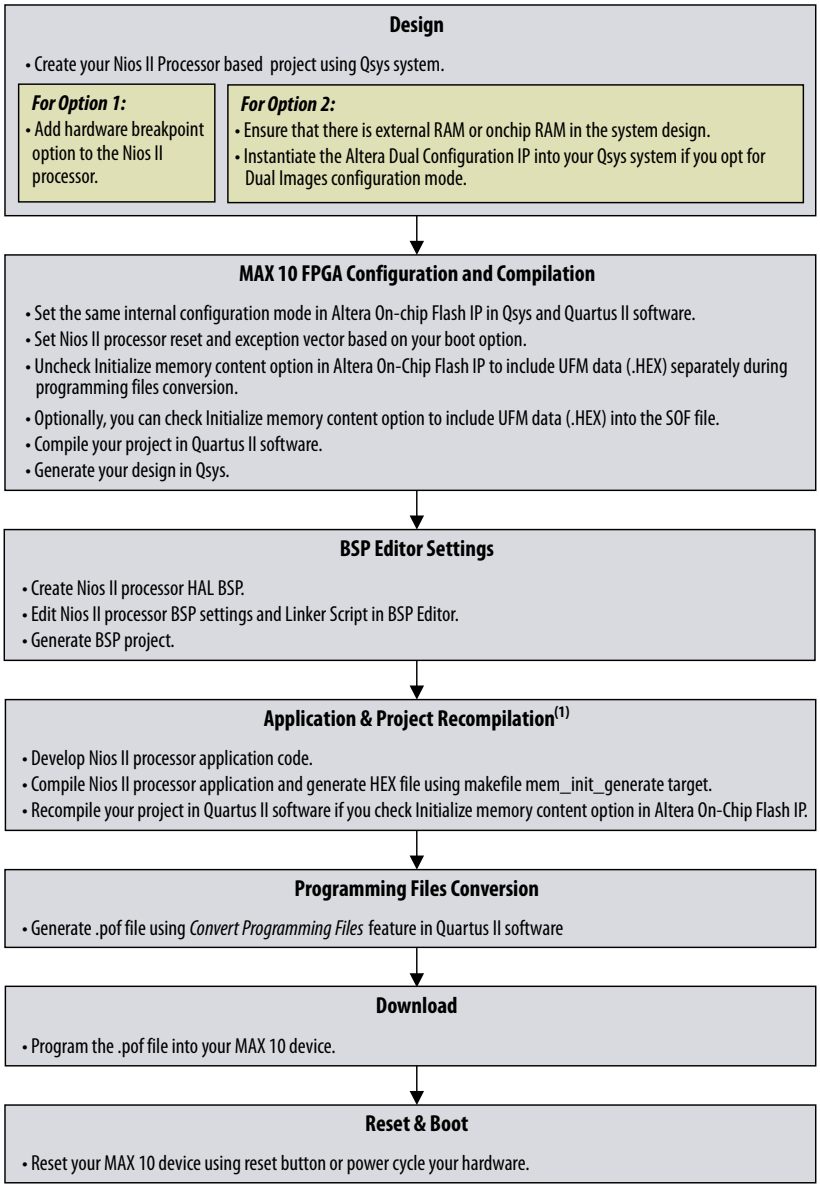
Table 7: RAM and ROM Size Requirement For Each Boot Option

Note: You can manually determine whether your code fits into the Altera On-Chip Flash by referring to the initial part of the **.objdump** file, created when you build your application.

Boot Option	RAM Size Requirement	ROM Size Requirement
Option 1: Nios II processor application execute in-place from Altera On-chip Flash (UFM).	Equivalent to the dynamic memory space usage during run time which is the sum of the maximum heap and stack size.	Executable code must not exceed the size of the UFM.
Option 2: Nios II processor application copied from UFM to RAM using boot copier.	Equivalent to the executable code and dynamic memory size required by user program.	Executable code and boot copier must not exceed the size of the UFM.
Option 3: Nios II processor application execute in-place from Altera On-chip Memory (OCRAM).	Equivalent to the executable code and dynamic memory size required by user program.	Not applicable for this boot option.

Boot System Guidelines for Option 1 and Option 2

Figure 5: Configuration and Booting Flow for Option 1 and Option 2



(1) Project re-compilation is needed if Initialize Flash Content option was checked in Altera On-Chip Flash IP. You can ignore this step if Intialize Flash Content option was unchecked in Altera On-Chip Flash IP.

Single Uncompressed/Compressed Image Bootable System Guideline

Qsys Settings

1. In Nios II Gen2 Processor parameter editor, set the reset vector memory and exception vector memory based on the boot options below:

Boot Option	Reset vector memory:	Exception vector memory:
Option 1a(5)(6)	Altera On-chip Flash	OCRAM/ External RAM

Boot Option	Reset vector memory:	Exception vector memory:
Option 1b[5]	Altera On-chip Flash	Altera On-chip Flash
Option 2	Altera On-chip Flash	OCRAM/ External RAM

Figure 6: Nios II Gen2 Parameter Editor Settings Boot Option 1a and 2

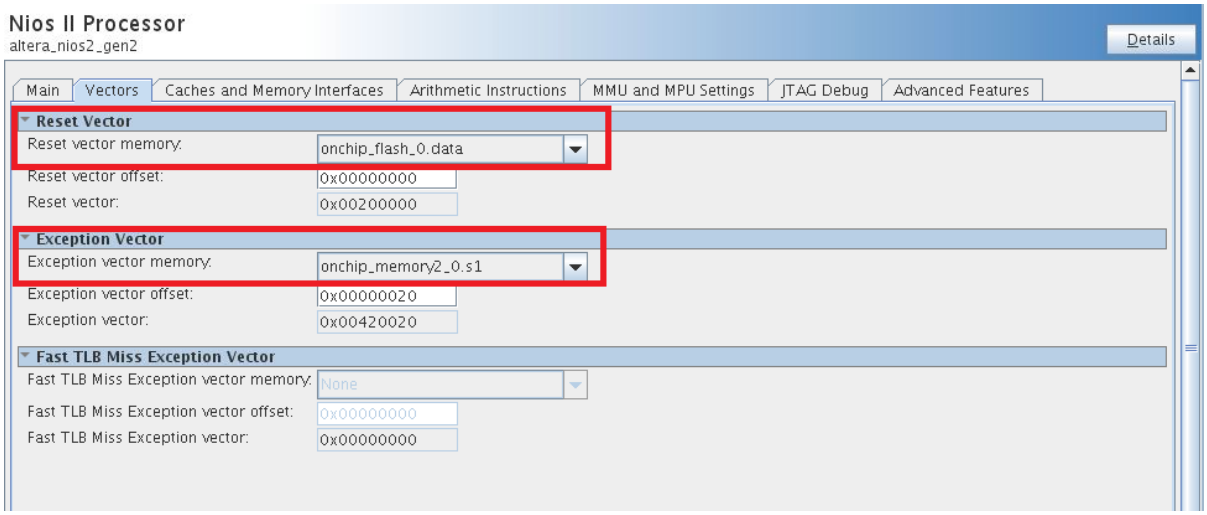
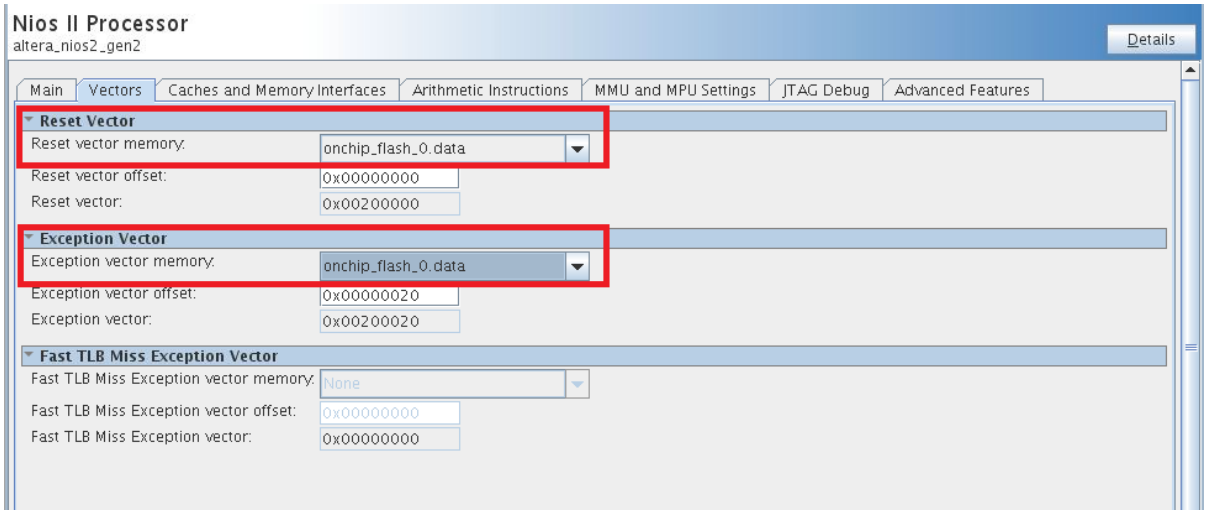


Figure 7: Nios II Gen2 Parameter Editor Settings Boot Option 1b

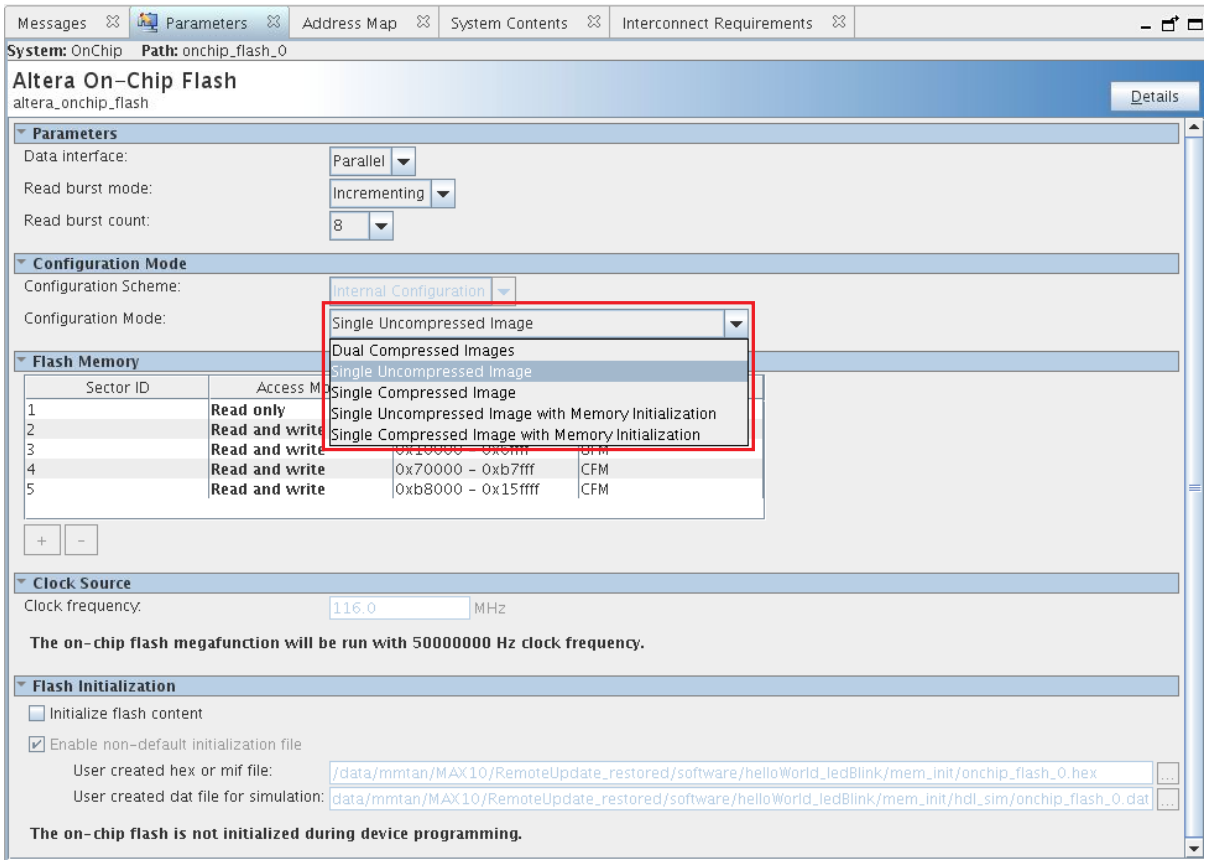


2. In Altera On-chip Flash IP parameter editor, set the **Configuration Mode:** to **Single Uncompressed Image** or **Single Compressed Image**.

(5) You can set the exception vector for Boot Option 1 to OCRAM/ External RAM (Option 1a) or Altera On-chip Flash (option 1b) according to your design preference.

(6) Boot option 1a which sets exception vector memory to OCRAM/External RAM is recommended to make the interrupt processing faster.

Figure 8: Configuration Mode Selection in Altera On-Chip Flash Parameter Editor



3. Refer to the following table for the options to program UFM data (HEX file) and settings required in Altera On-chip Flash IP.

Options to program UFM data	Method	Settings in Altera On-Chip Flash IP
Option 1: Initialize the UFM data in the SOF	Quartus II includes the UFM initialization data in the SOF during compilation. SOF recompilation is needed if there are changes in the UFM data.	1. Check **Initialize flash content** 2. If default path is used, add **meminit.qip** generated during "make mem_init_generate" into Quartus II project. Refer to **Figure 10**. Make sure the generated HEX naming matches the default naming. 3. If non-default path is selected, enable the **Enable non-default initialization file** and specify the path of the HEX file. **Note:** For more information about Steps 2 and 3, refer to HEX File Generation section.
Option 2: Combine UFM data with a compiled SOF during programming files (POF) conversion (7)	UFM data is combined with the compiled SOF during the programming files conversion. SOF recompilation is NOT needed even if there are changes in the UFM data.	Uncheck **Initialize flash content**

(7) This is the recommended method for application developer. You are not required to recompile SOF file for application changes during development.

Figure 9: Initialize Flash Contents with Default Initialization File

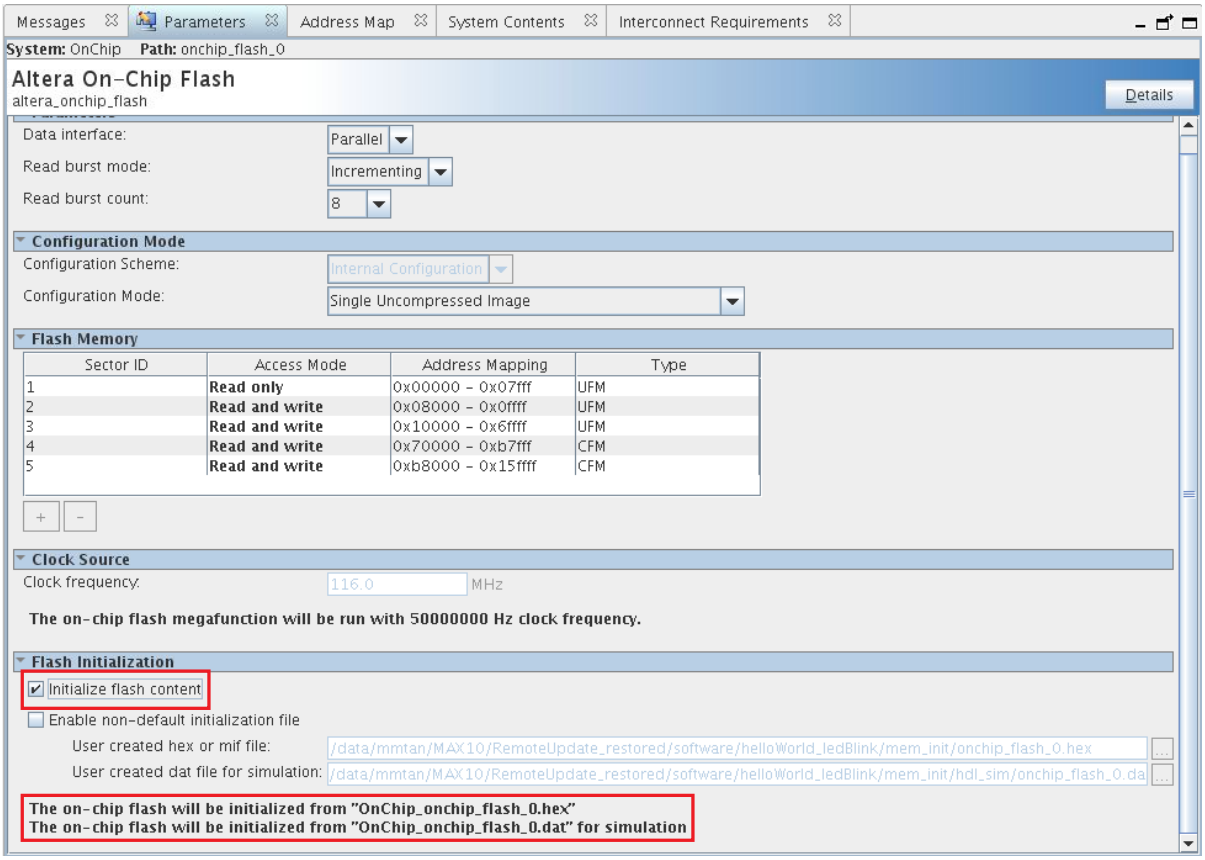


Figure 10: Adding meminit.qip File in Quartus II

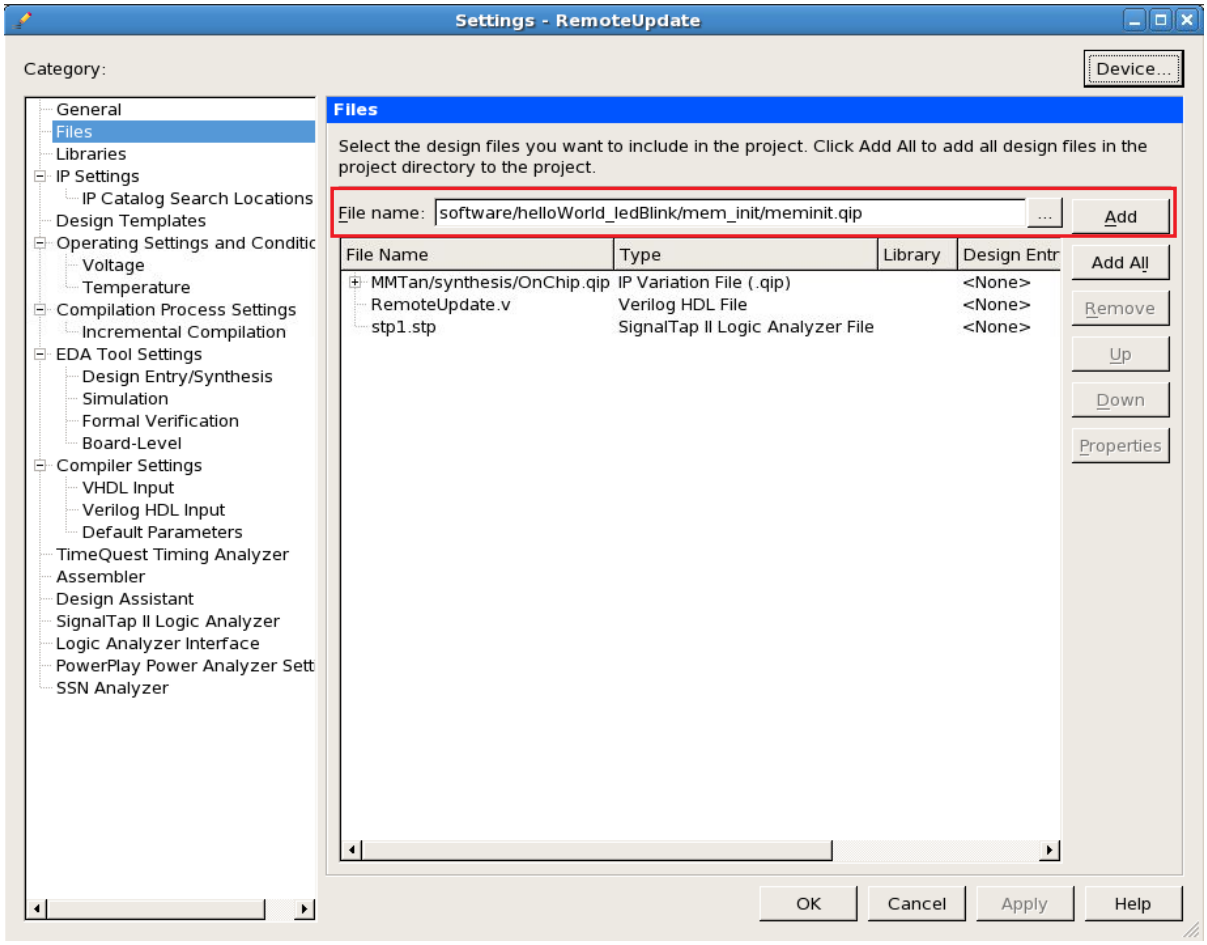
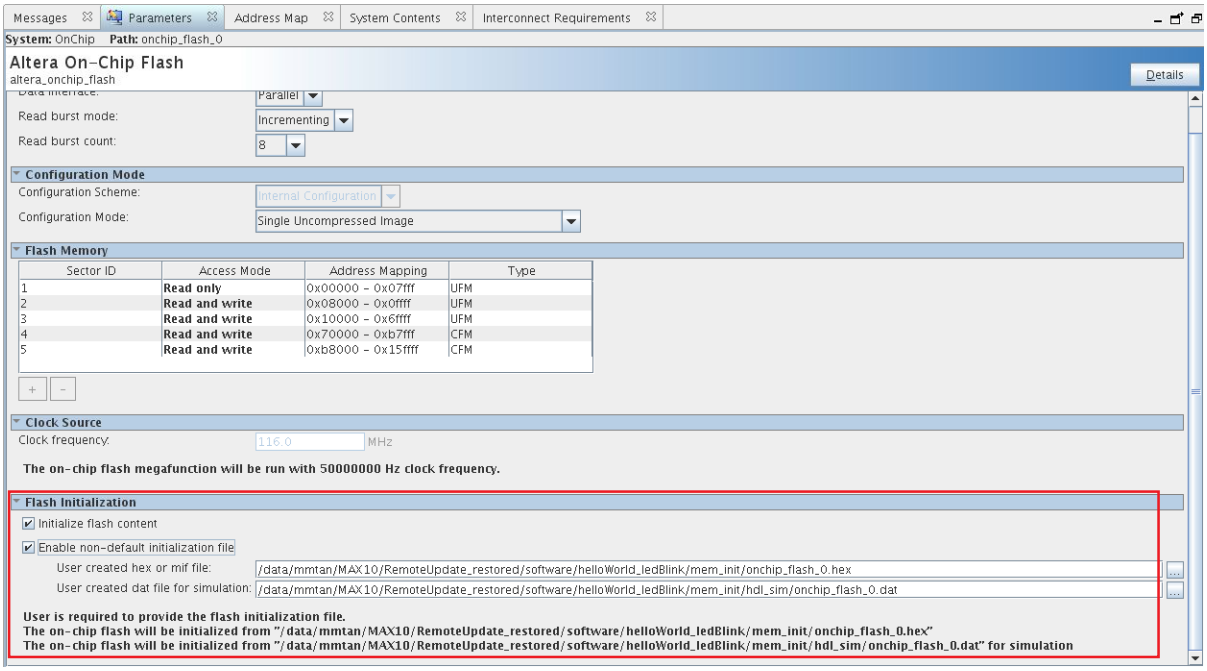
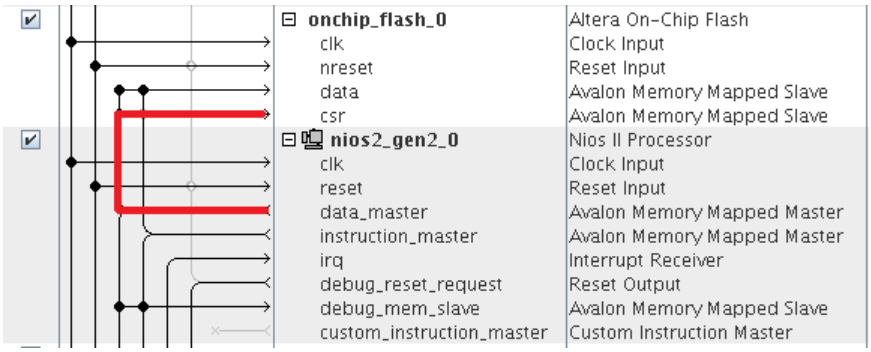


Figure 11: Initialize Flash Content with Non-default Initialization File



4. Ensure that the Altera On-chip Flash CSR port is connected to the Nios II Gen2 processor data master to enable write and erase operations.

Figure 12: CSR Connection to Nios II Gen2 `data_master`



5. Click **Generate HDL**, the **Generation** dialog box appears.
6. Specify output file generation options, and then click **Generate**.

Related Information

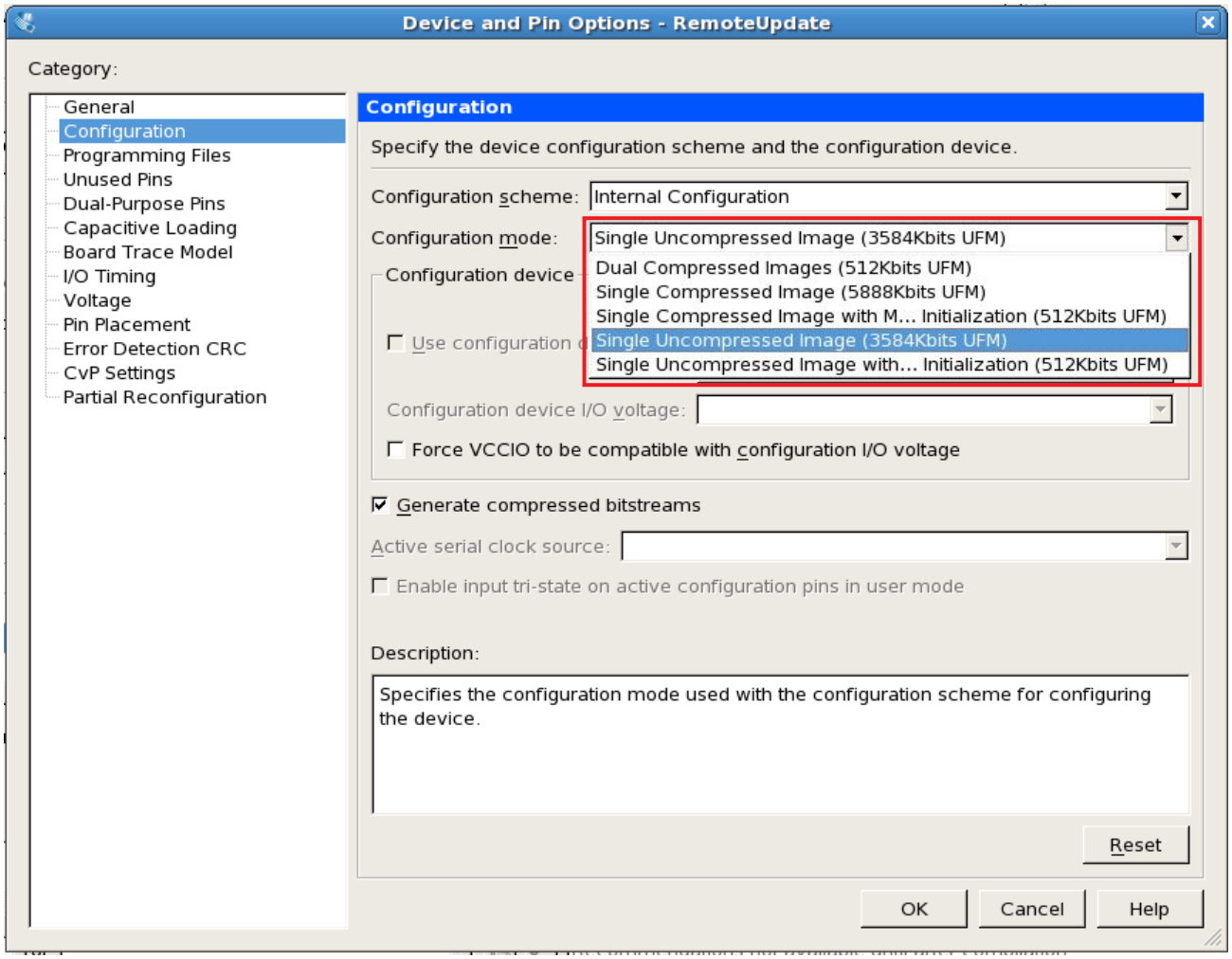
HEX File Generation on page 20

Quartus II Software Settings

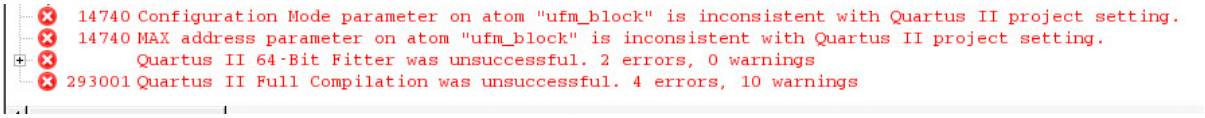
1. In Quartus II software, click on **Assignment** -> **Device** -> **Device and Pin Options** -> **Configuration**. Set **Configuration mode:** to **Single Uncompressed Image** or **Single Compressed Image**.[8]

[8] The size of UFM shown will vary according to your device selection.

Figure 13: Configuration Mode Selection in Quartus II Software



Note: If the configuration mode setting in Quartus II software and Qsys parameter editor is different, the Quartus II project compilation will fail with the following error message.



2. Click **OK** to exit the **Device and Pin Options window**.
3. Click **OK** to exit the **Device** window.
4. Click **Start Compilation** to compile your project and generate the **.sof** file.

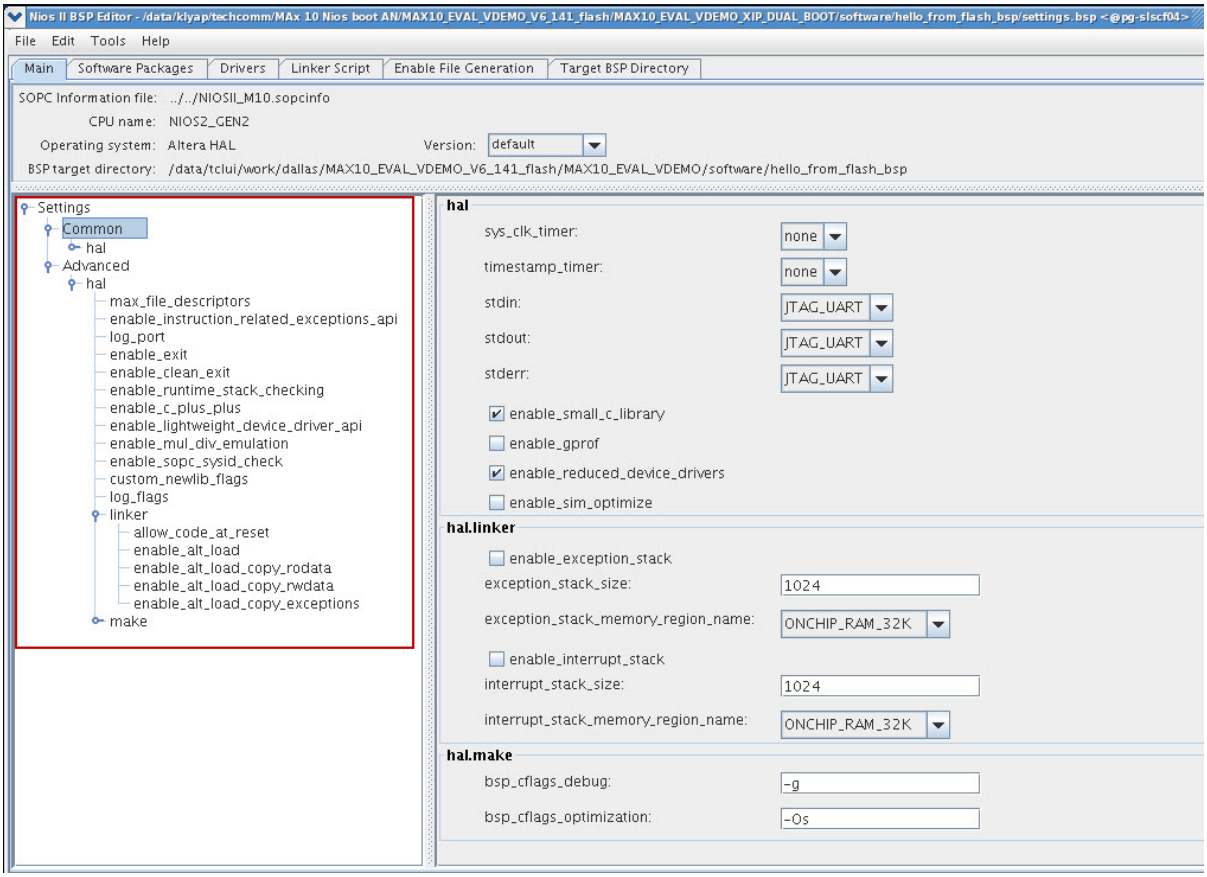
Related Information

HEX File Generation on page 20

BSP Editor Settings

You must edit the BSP editor settings according to the selected Nios II processor boot options.

1. In the Nios II SBT tool, right click on your BSP project in the **Project Explorer** window. Select **Nios II > BSP Editor...** to open the **Nios II BSP Editor**.
2. In Nios II BSP Editor, click on **Advanced** tab under **Settings**.
3. Click on **hal** to expand the list.
4. Click on **linker** to expand the list.



5. Based on the boot option used, do one of the following:

- For boot option 1a, if exception vector memory is set to OCRAM/ External RAM, enable the following:
  - **allow_code_at_reset**
  - **enable_alt_load**
  - **enable_alt_load_copy_rodata**
  - **enable_alt_load_copy_rwdata**
  - **enable_alt_load_copy_exceptions**
- For boot option 1b, if exception vector memory is set to Altera On-chip Flash, enable the following:
  - **allow_code_at_reset**
  - **enable_alt_load**
  - **enable_alt_load_copy_rodata**
  - **enable_alt_load_copy_rwdata**
- For boot option 2, leave all the hal.linker settings unchecked.

Figure 14: Advanced.hal.linker Settings for Boot Option 1a

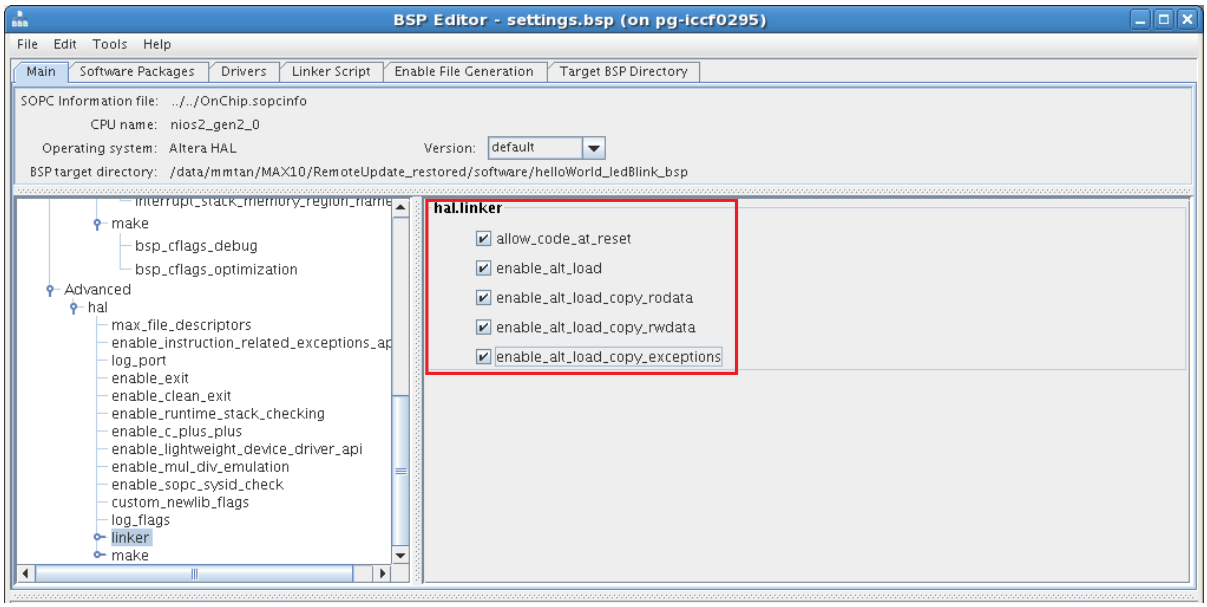


Figure 15: Advanced.hal.linker Settings for Boot Option 1b

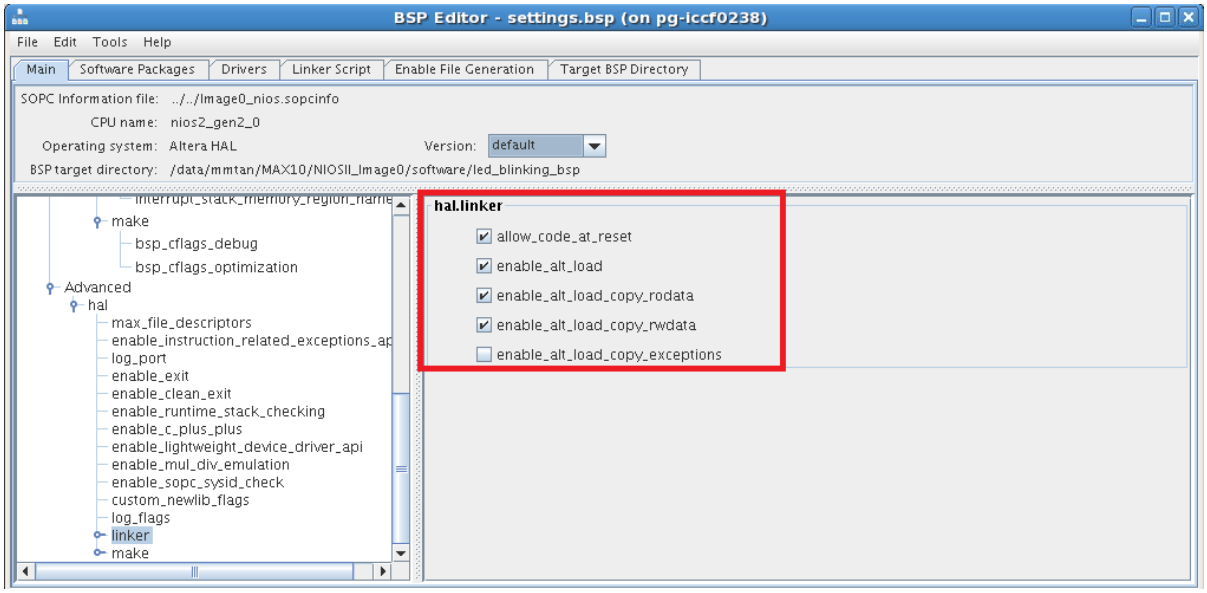
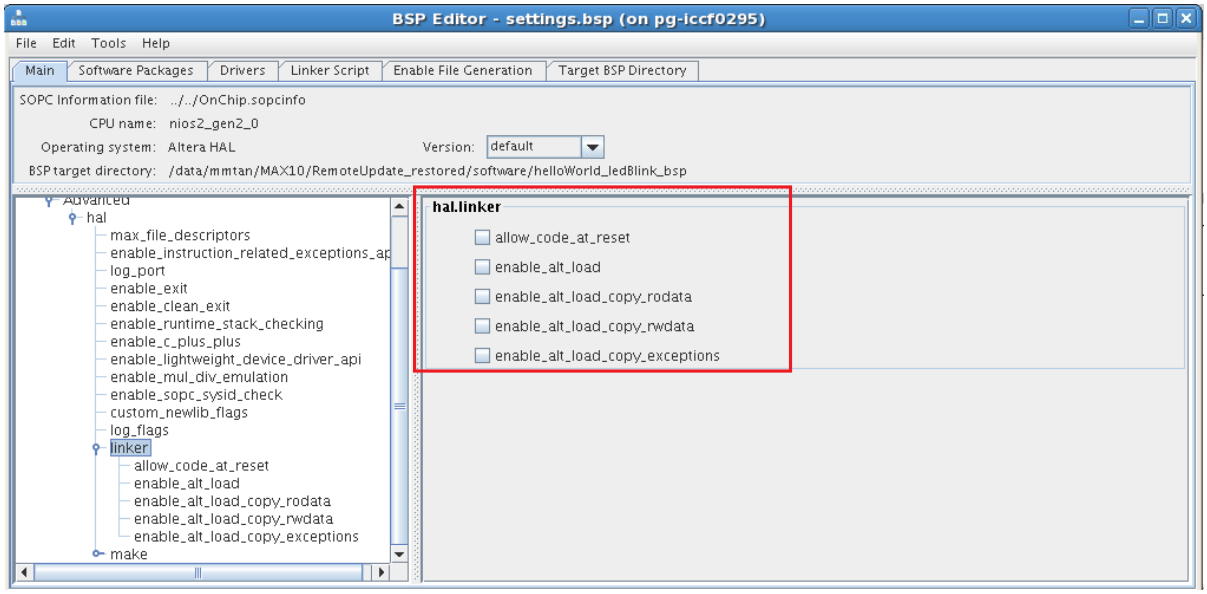


Figure 16: Advanced.hal.linker Settings for Boot Option 2



6. Click on **Linker Script** tab in the Nios II BSP Editor.
7. Based on the boot option used, do one of the following:
   - For boot option 1a and 1b, set the **.text** item in the **Linker Section Name** to the Altera On-chip Flash in the **Linker Region Name**. Set the rest of the items in the **Linker Section Name** list to the Altera On-chip Memory (OCRAM) or external RAM.
   - For boot option 2, set all of the items in the **Linker Section Name** list to Altera On-chip Memory (OCRAM) or external RAM.

Figure 17: Linker Region Settings for Boot Option 1a and 1b

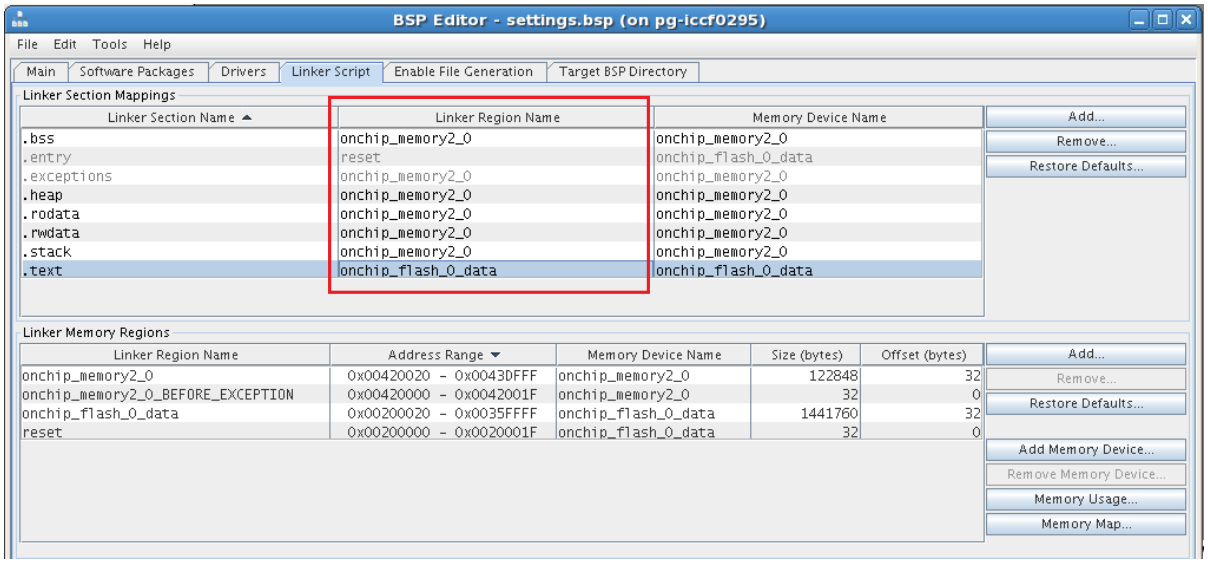
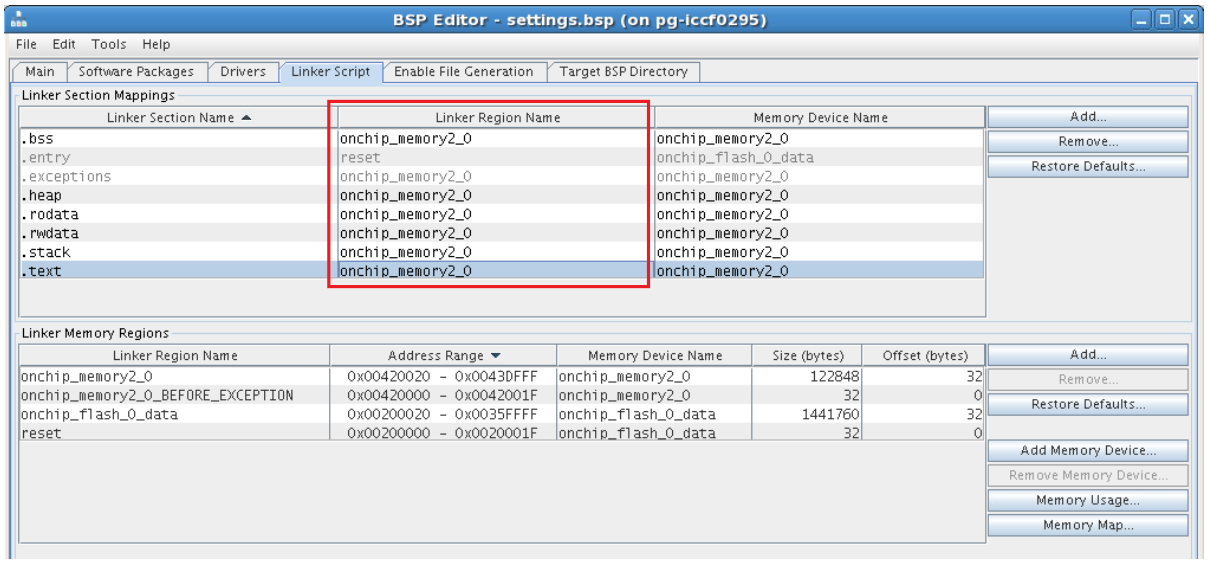


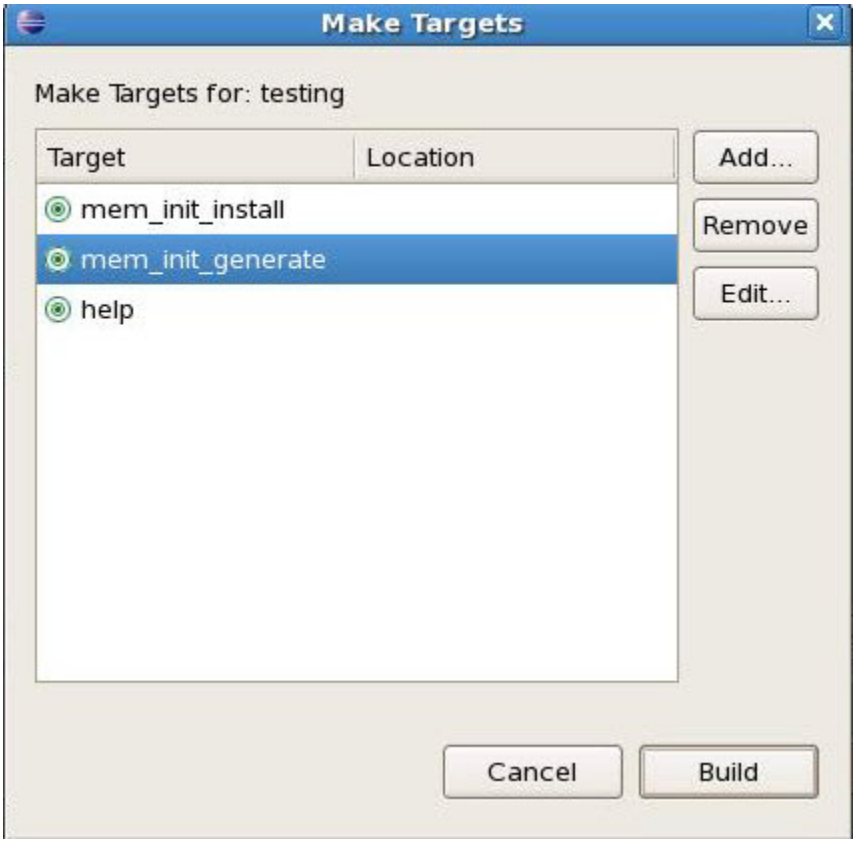
Figure 18: Linker Region Settings for Boot Option 2



HEX File Generation

1. In the Nios II SBT tool, right click on your project in the Project Explorer window.
2. Click **Make Targets** -> **Build…**, the Make Targets dialog box appears. You can also press shift + F9 to trigger the Make Target dialog box.
3. Select **mem_init_generate**.
4. Click **Build** to generate the HEX file.

Figure 19: Selecting mem_init_generate in Make Targets



5. The "mem_init_generate" macro will create two HEX files; **on_chip_ram.hex** and **on_chip_flash.hex**. The **on_chip_ram.hex** will be used for boot option 3 and **on_chip_flash.hex** is used for boot option 1 and 2.

   Note:
   - The **mem_init_generate** target also generates a Quartus II IP file (**meminit.qip**). Quartus II software will refer to the **meminit.qip** for the location of the initialization files.
   - All these files can be found under "<project_folder>/software/<application_name>/mem_init" folder.

6. Recompile your project in Quartus II software if you check **Initialize memory content** option in Altera On-Chip Flash IP. This is to include the software data (.HEX) into the SOF file.

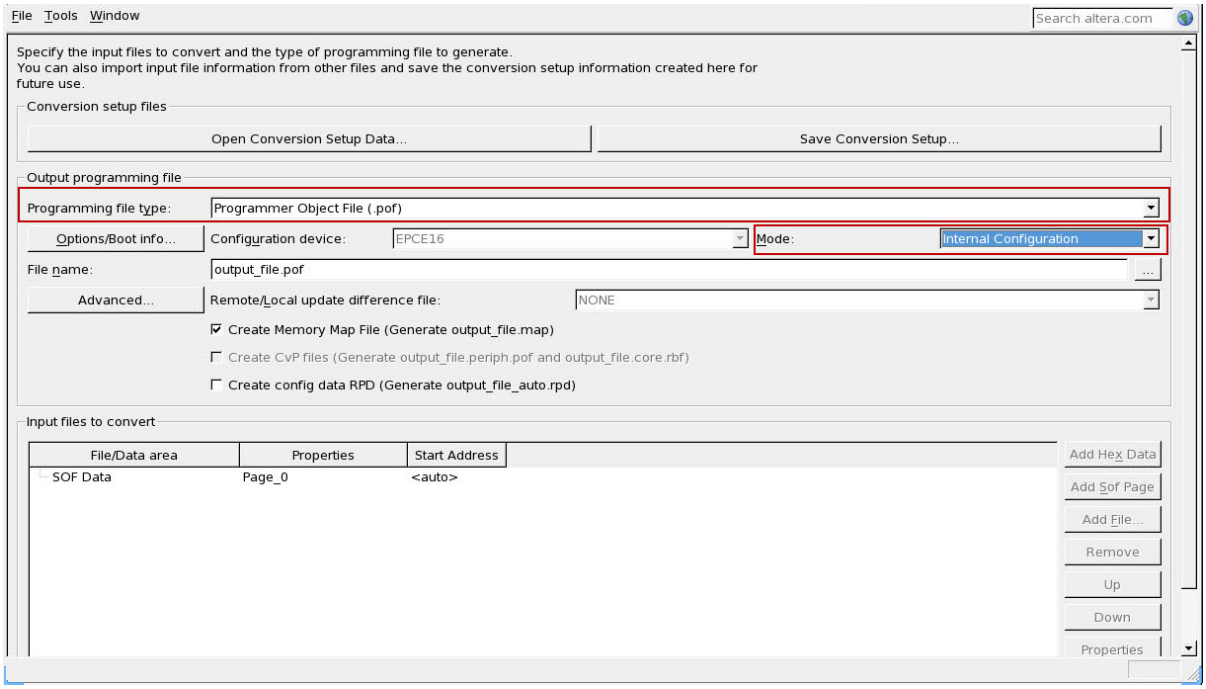
Related Information

- Qsys Settings on page 9
- Quartus II Software Settings on page 15

Programmer Object File (.pof) Generation

1. In Quartus II, click on **Convert Programming Files (.pof)** from the **File** tab.
2. Choose **Programmer Object File** as **Programming file type:**.
3. Set **Mode** to **Internal Configuration**.

Figure 20: Convert Programming File Settings



4. Click on **Options/Boot info...**, the MAX 10 Device Options dialog box appears.
5. Based on the **Initialize flash content** settings in the Altera On-chip Flash IP, do one of the following:
   - If **Initialize flash content** is checked, the UFM initialization data was included in the SOF during Quartus II compilation. Select **Page_0** for **UFM source:** option. Click **OK** and proceed to next step.
   - If **Initialize flash content** is not checked, choose **Load memory file** for **UFM source:** option. Browse to the generated Altera On-chip Flash HEX file (**on_chip_flash.hex**) in the **File path:** and click **OK**. This will add UFM data separately to the SOF file during the programming file conversion.

Figure 21: Setting Page_0 for UFM Source if Initialize flash content is Checked

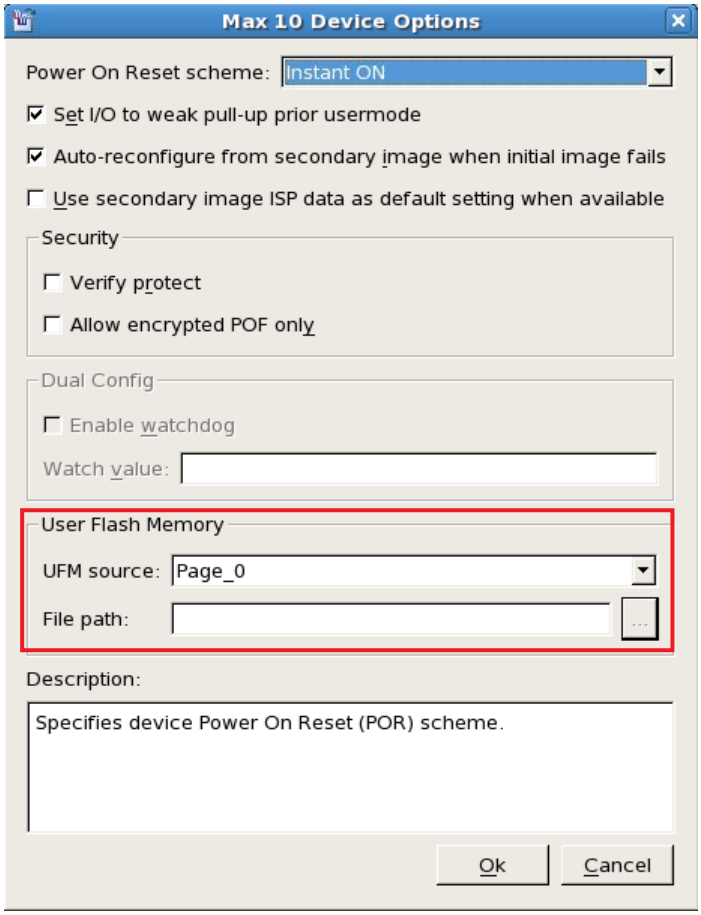
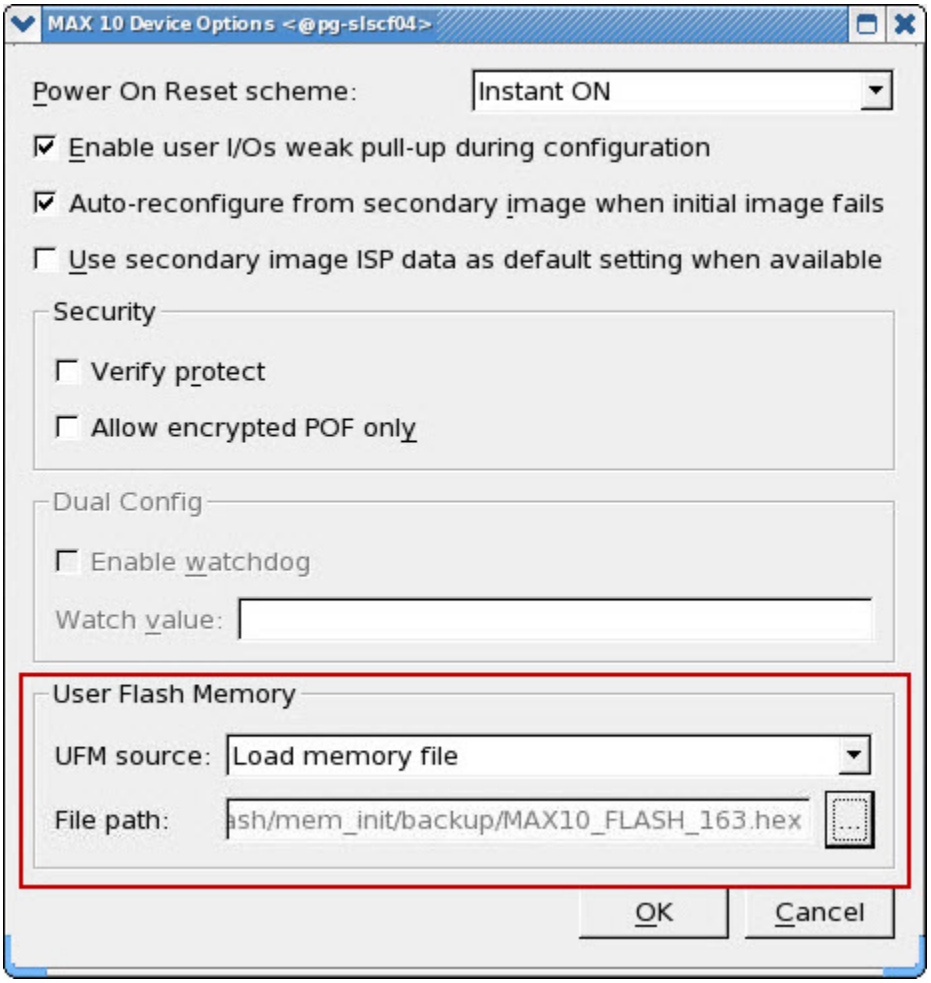
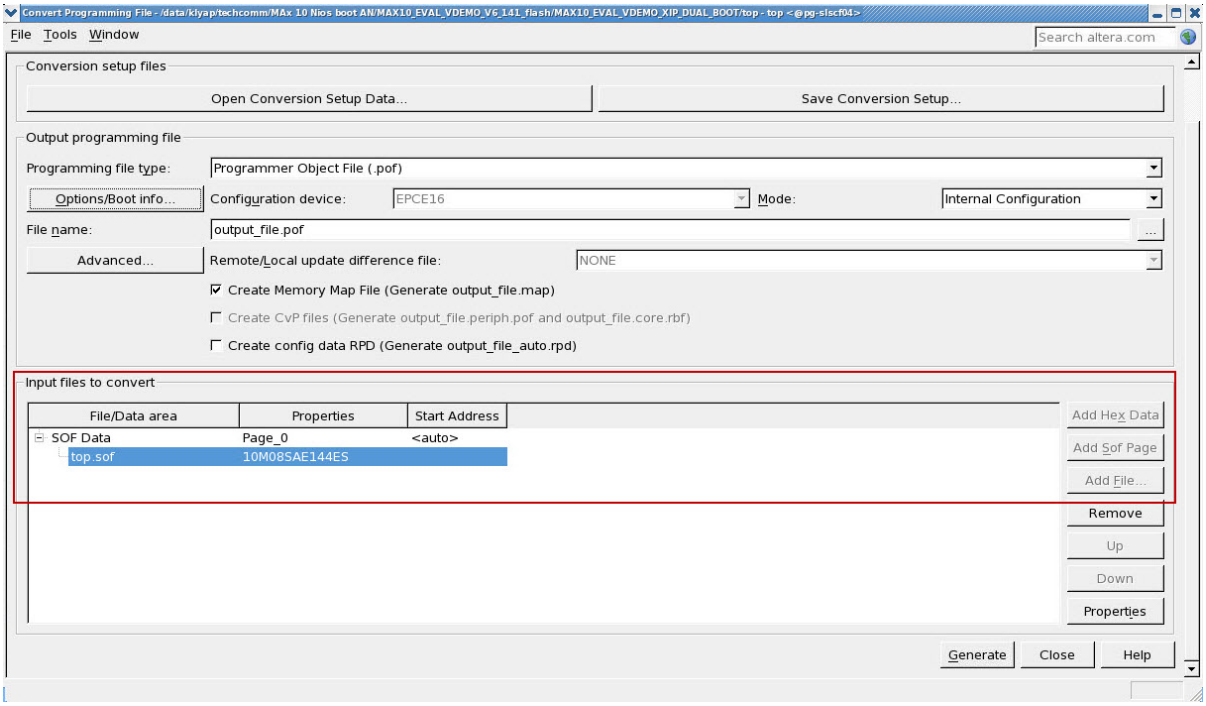


Figure 22: Setting Load Memory File for UFM Source if Initialize flash content is not Checked



6. In the Convert Programming File dialog box, at the **Input files to convert** section, click **Add File...** and point to the generated Quartus II .sof file.

Figure 23: Input Files to Convert in Convert Programming Files



7. Click **Generate** to create the .pof file.
8. Program the .pof file into your MAX 10 device.

Dual Compressed Images Bootable System Guideline

Related Information

MAX 10 FPGA Configuration User Guide, section Remote System Upgrade in Dual Compressed Images.

Qsys Settings

1. In Nios II Gen2 Processor parameter editor, set the reset vector memory and exception vector memory based on the boot options below:

Boot Option	Reset vector memory:	Exception vector memory:
Option 1a[9][10]	Altera On-chip Flash	OCRAM/ External RAM
Option 1b[9]	Altera On-chip Flash	Altera On-chip Flash
Option 2	Altera On-chip Flash	OCRAM/ External RAM

(9) You can set the exception vector for Boot Option 1 to OCRAM/ External RAM (Option 1a) or Altera On-chip Flash (option 1b) according to your design preference.

(10) Boot option 1a which sets exception vector memory to OCRAM/External RAM is recommended to make the interrupt processing faster.

Figure 24: Nios II Gen2 Parameter Editor Settings Boot Option 1a and 2

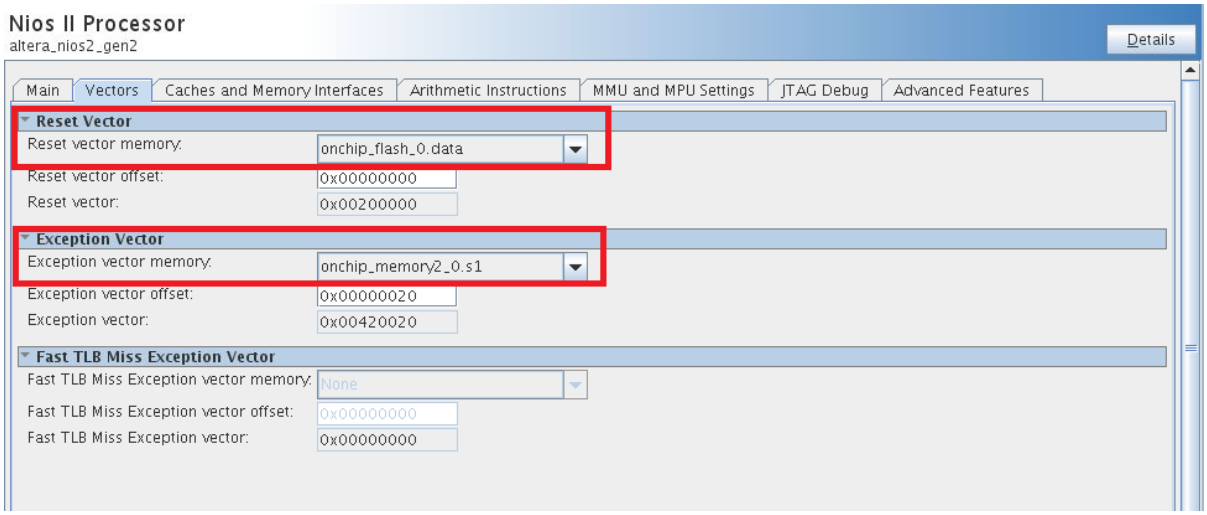
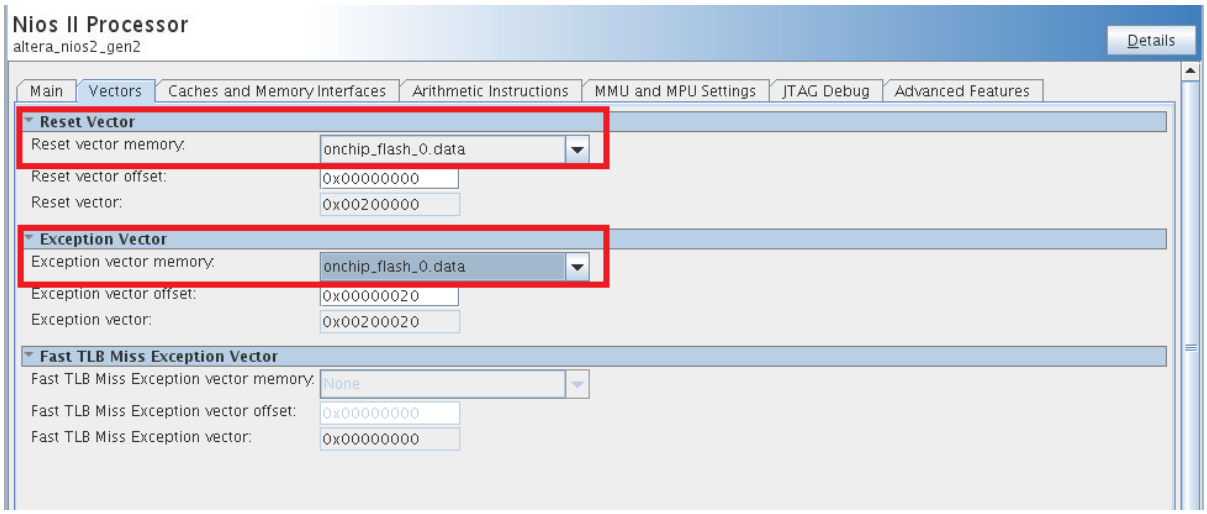


Figure 25: Nios II Gen2 Parameter Editor Settings Boot Option 1b



2. In Altera On-chip Flash IP parameter editor, set the **Configuration Mode:** to **Dual Compressed Images**.
3. Refer to the following table for the options to program UFM data (HEX file) and settings required in Altera On-chip Flash IP.

Options to program UFM data	Method	Settings in Altera On-Chip Flash IP
Option 1: Initialize the UFM data in the SOF	Quartus II includes the UFM initialization data in the SOF during compilation. SOF recompilation is needed if there are changes in the UFM data.	1. Check **Initialize flash content** 2. If default path is used, add **meminit.qip** generated during "make mem_init_generate" into Quartus II project. Refer to Figure 27. Make sure the generated HEX naming matches the default naming. 3. If non-default path is selected, enable the **Enable non-default initialization file** and specify the path of the HEX file. **Note:** For more information about Steps 2 and 3, refer to HEX File Generation section.
Option 2: Combine UFM data with a compiled SOF during programming files (POF) conversion (11)	UFM data is combined with the compiled SOF during the programming files conversion. SOF recompilation is NOT needed even if there are changes in the UFM data.	Uncheck **Initialize flash content**

(11) This is the recommended method for application developer. You are not required to recompile SOF file for application changes during development.

Figure 26: Dual Compressed Images Mode with initialize flash content Turned-on

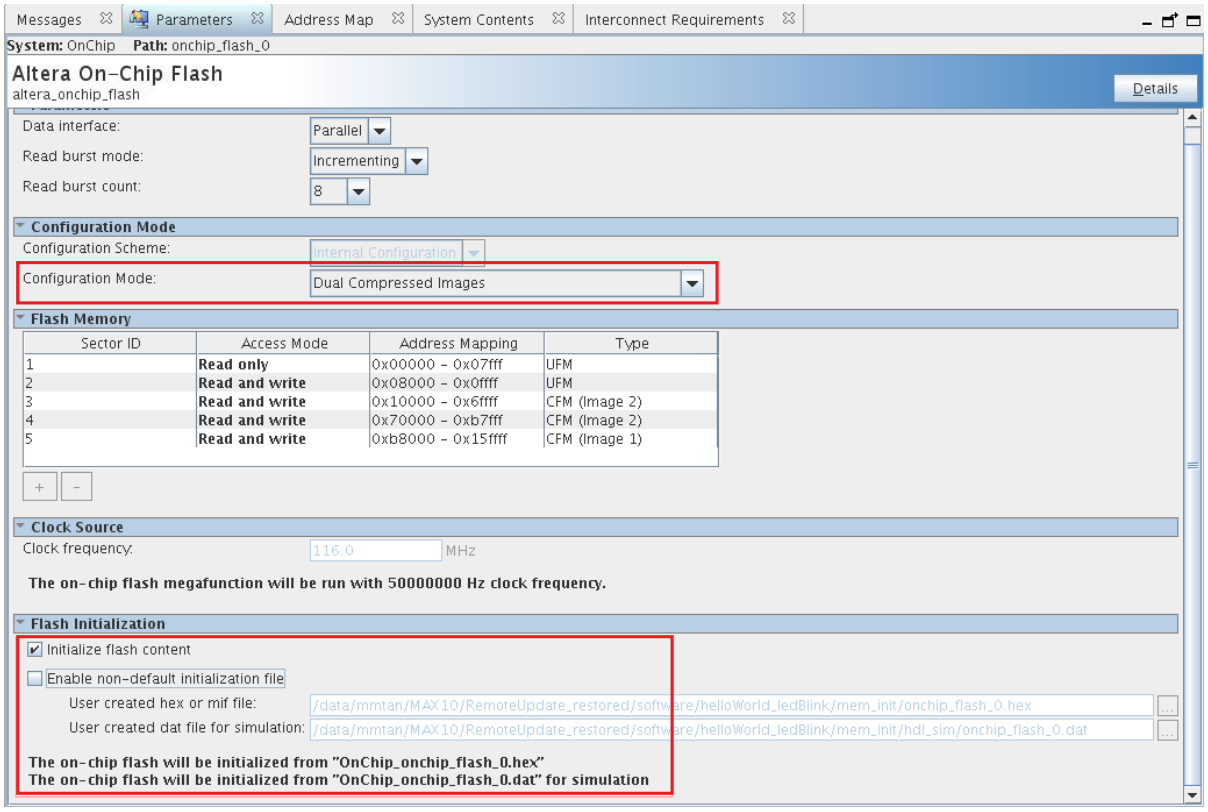


Figure 27: Adding meminit.qip File in Quartus II

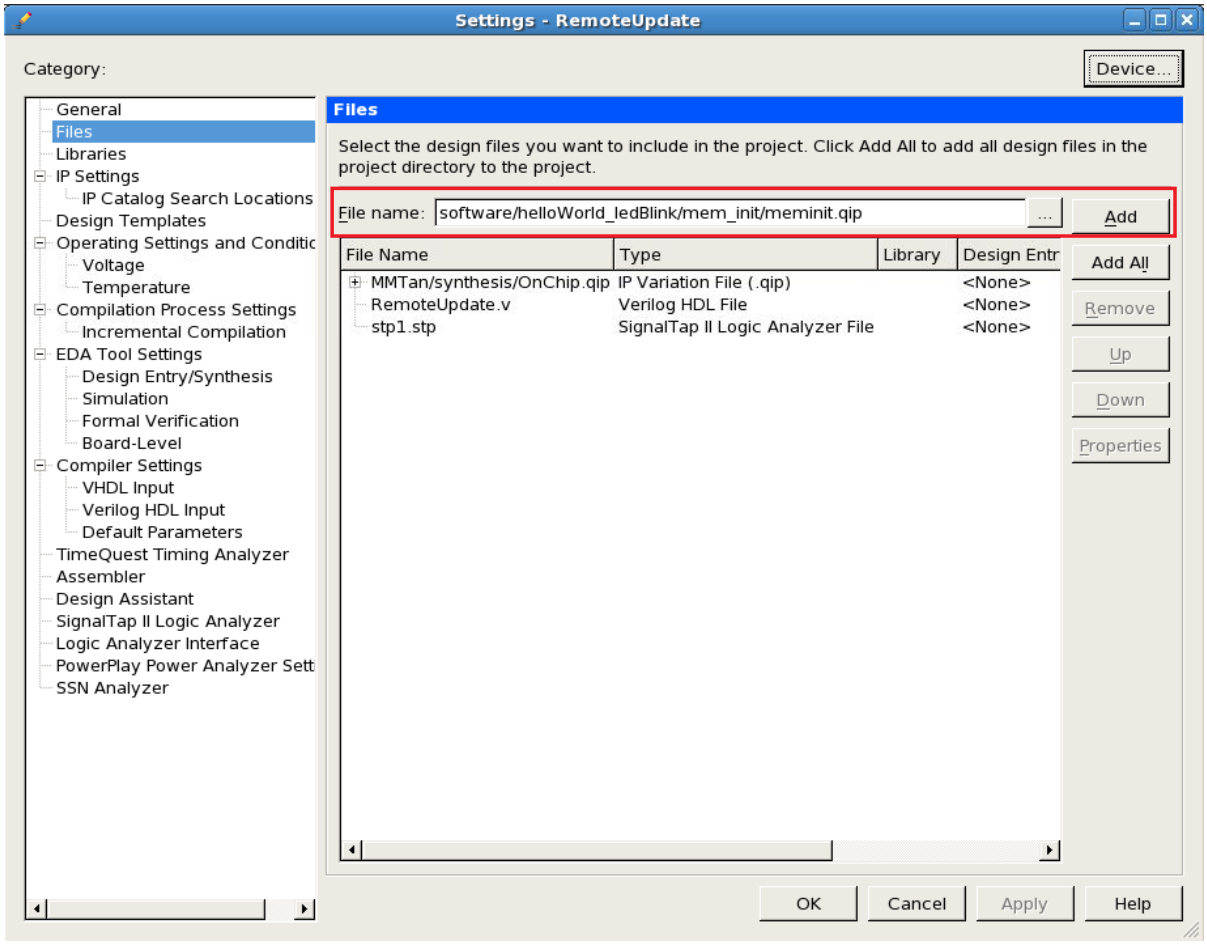
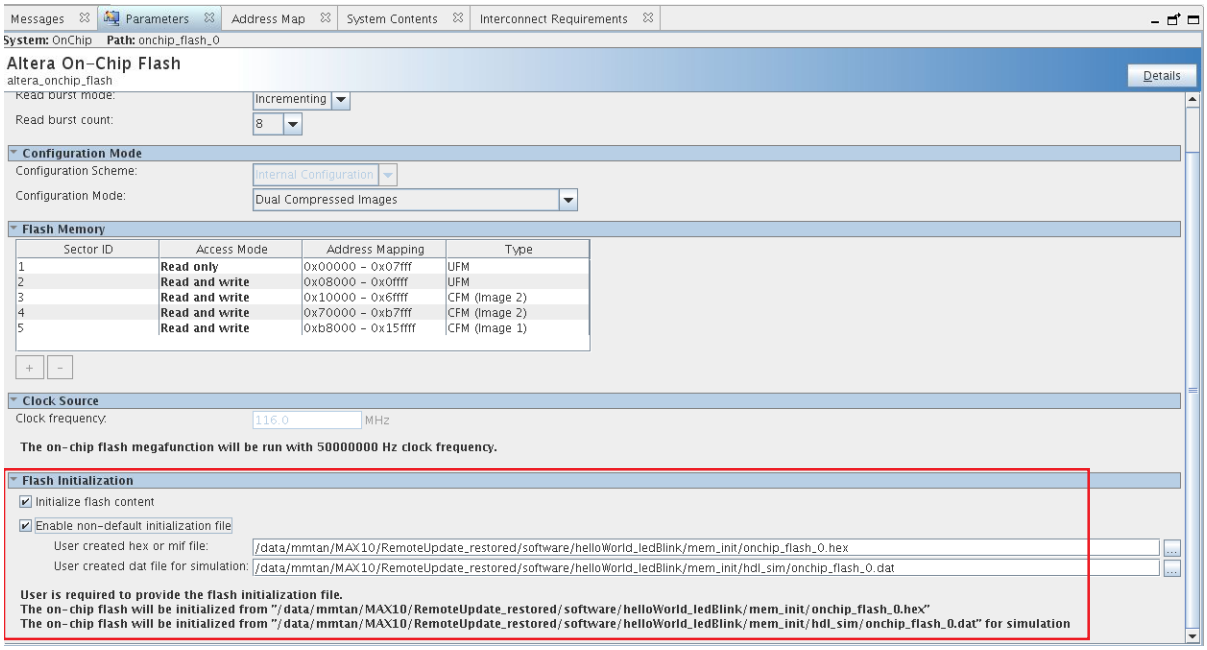
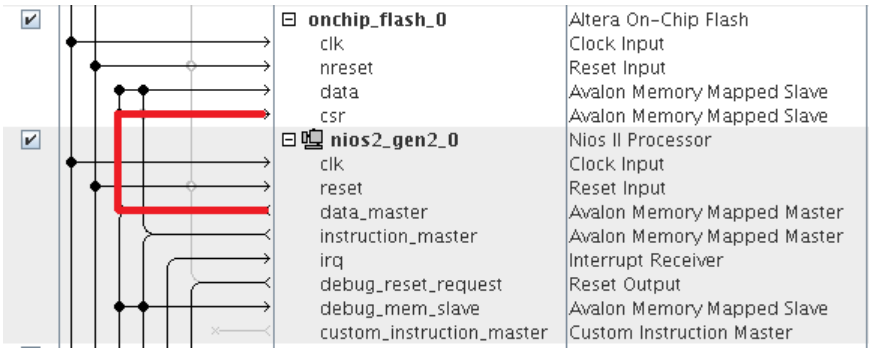


Figure 28: Dual compressed Images with Non-default Initialization File Enabled



4. Ensure that the Altera On-chip Flash CSR port is connected to the Nios II Gen2 processor data master to enable write and erase operations.

Figure 29: CSR Connection to Nios II Gen2 `data_master`



5. Ensure the Altera Dual Configuration IP is instantiated in Qsys to enable dual images configuration.
6. Click **Generate HDL**, the **Generation** dialog box appears.
7. Specify output file generation options, and then click **Generate**.

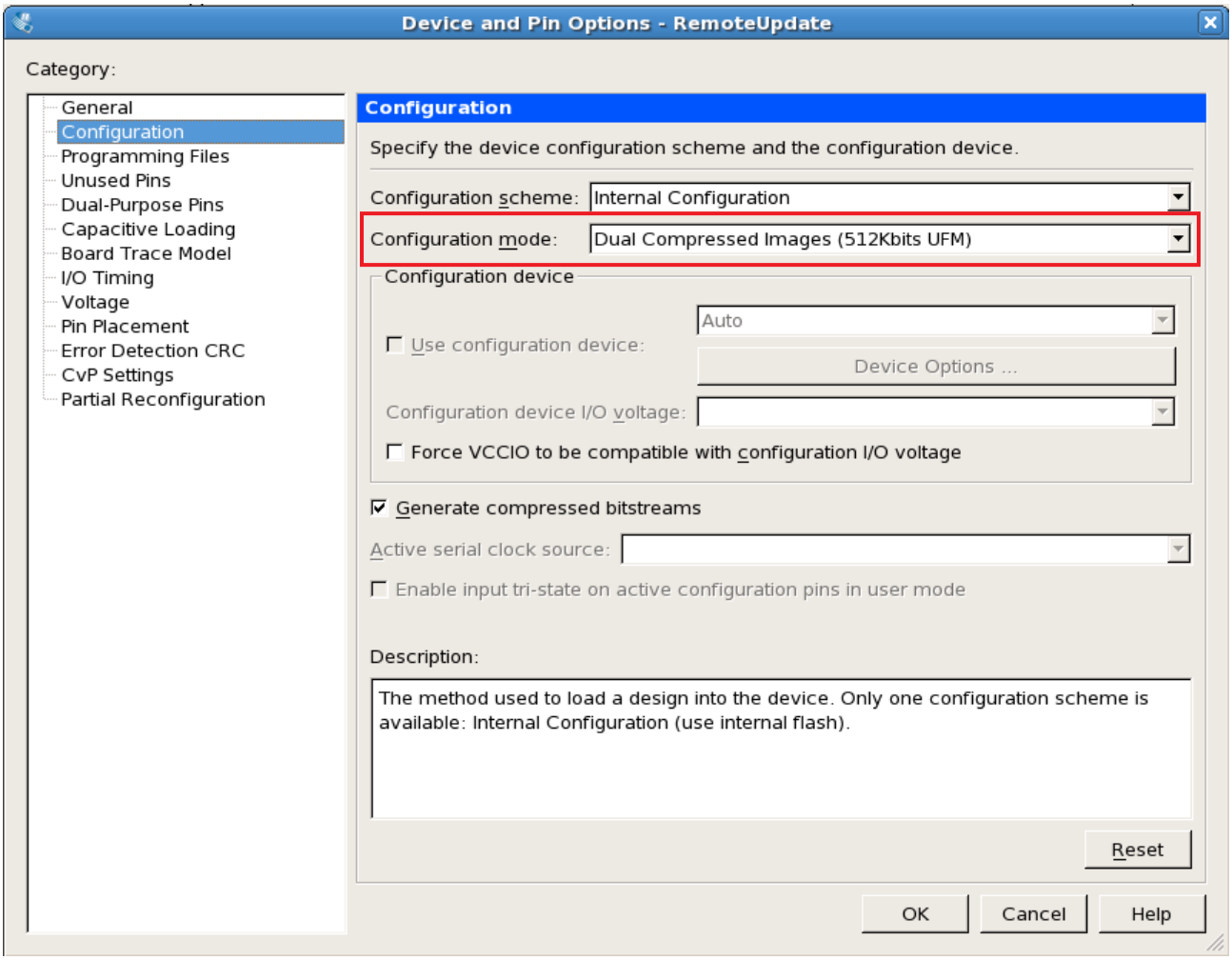
Related Information

HEX File Generation on page 35

Quartus II Software Settings

1. In Quartus II software, click on **Assignment** -> **Device** -> **Device and Pin Options** -> **Configuration**. Set **Configuration mode:** to **Dual Compressed Images**.(12)

Figure 30: Dual Compressed Images Configuration Mode in Quartus II



Note: If the configuration mode setting in Quartus II software and Qsys parameter editor is different, the Quartus II project compilation will fail with the following error message.

```
14740 Configuration Mode parameter on atom "ufm_block" is inconsistent with Quartus II project setting.
14740 MAX address parameter on atom "ufm_block" is inconsistent with Quartus II project setting.
      Quartus II 64-Bit Fitter was unsuccessful. 2 errors, 0 warnings
293001 Quartus II Full Compilation was unsuccessful. 4 errors, 10 warnings
```

2. Click **OK** to exit the **Device and Pin Options window**.
3. Click **OK** to exit the **Device** window.
4. Click **Start Compilation** to compile your project and generate the .sof file.

Related Information

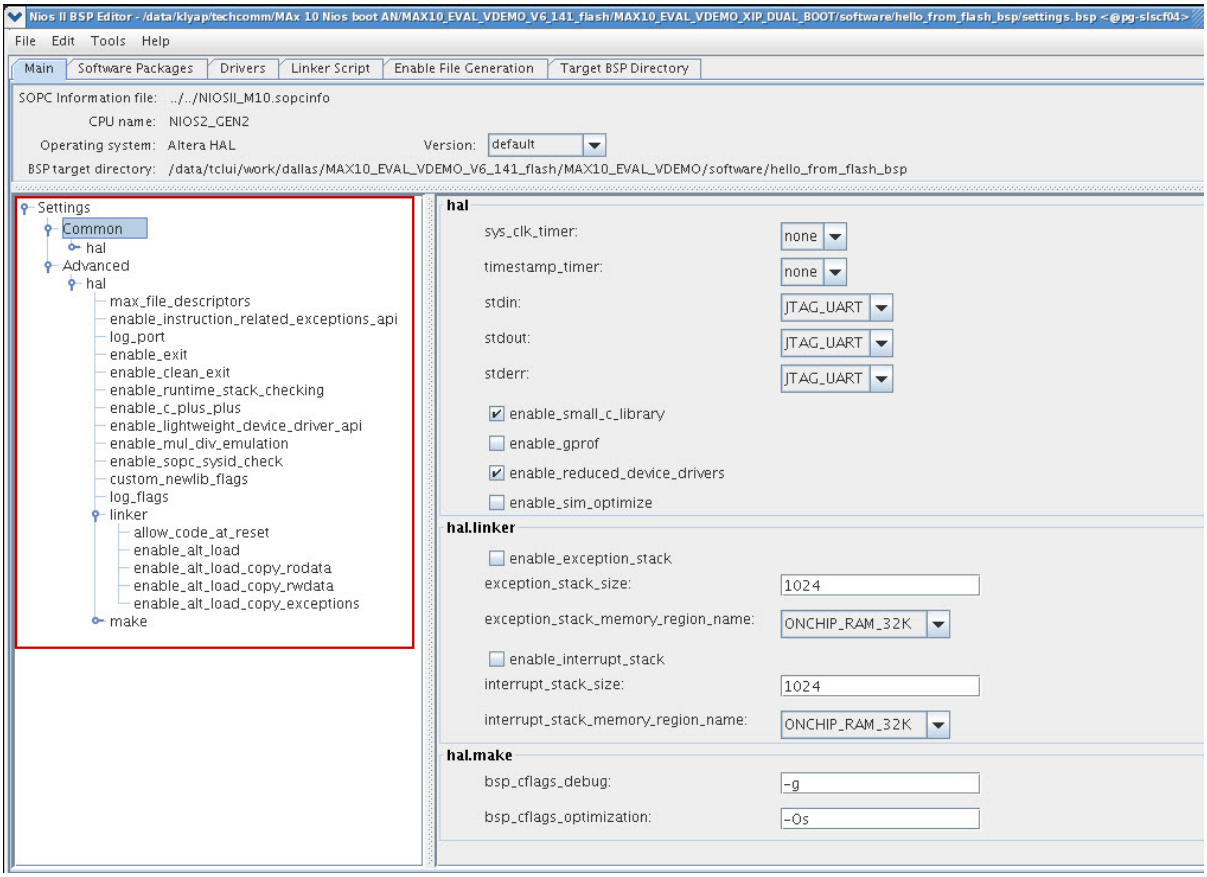
HEX File Generation on page 35

(12) The size of UFM sector will vary according to your device selection.

BSP Editor Settings

You must edit the BSP editor settings according to the selected Nios II processor boot options.

1. In the Nios II SBT tool, right click on your BSP project in the **Project Explorer** window. Select **Nios II > BSP Editor...** to open the **Nios II BSP Editor**.
2. In Nios II BSP Editor, click on **Advanced** tab under **Settings**.
3. Click on **hal** to expand the list.
4. Click on **linker** to expand the list.



5. Based on the boot option used, do one of the following:

- For boot option 1a, if exception vector memory is set to OCRAM/ External RAM, enable the following:
  - **allow_code_at_reset**
  - **enable_alt_load**
  - **enable_alt_load_copy_rodata**
  - **enable_alt_load_copy_rwdata**
  - **enable_alt_load_copy_exceptions**
- For boot option 1b, if exception vector memory is set to Altera On-chip Flash, enable the following:
  - **allow_code_at_reset**
  - **enable_alt_load**
  - **enable_alt_load_copy_rodata**
  - **enable_alt_load_copy_rwdata**
- For boot option 2, leave all the hal.linker settings unchecked.

Figure 31: Advanced.hal.linker Settings for Boot Option 1a

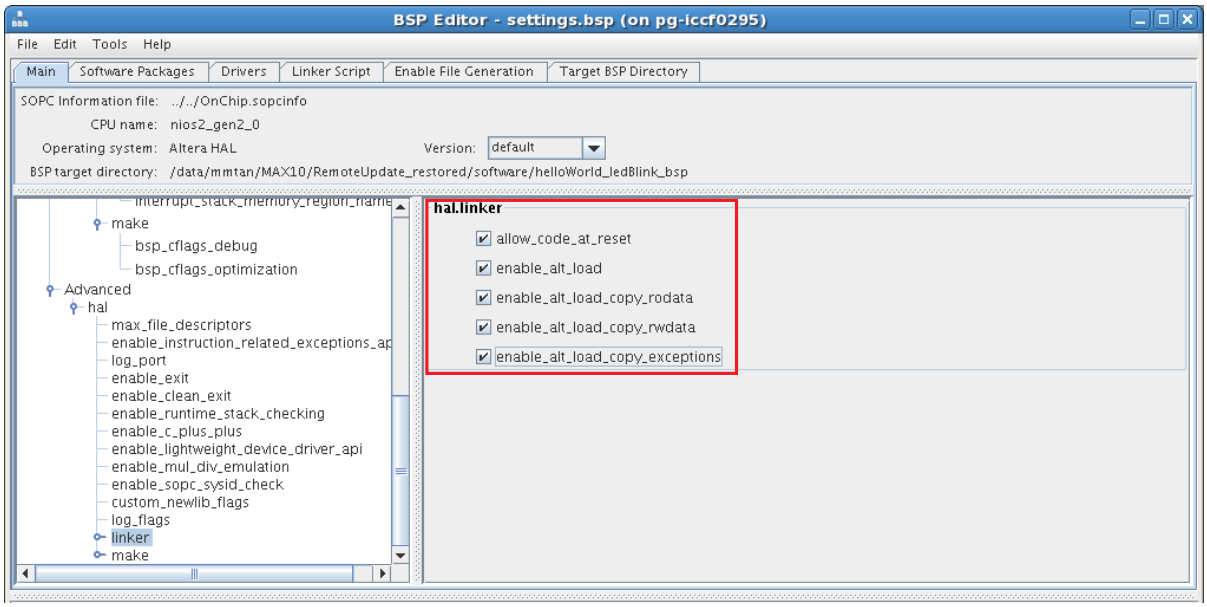


Figure 32: Advanced.hal.linker Settings for Boot Option 1b

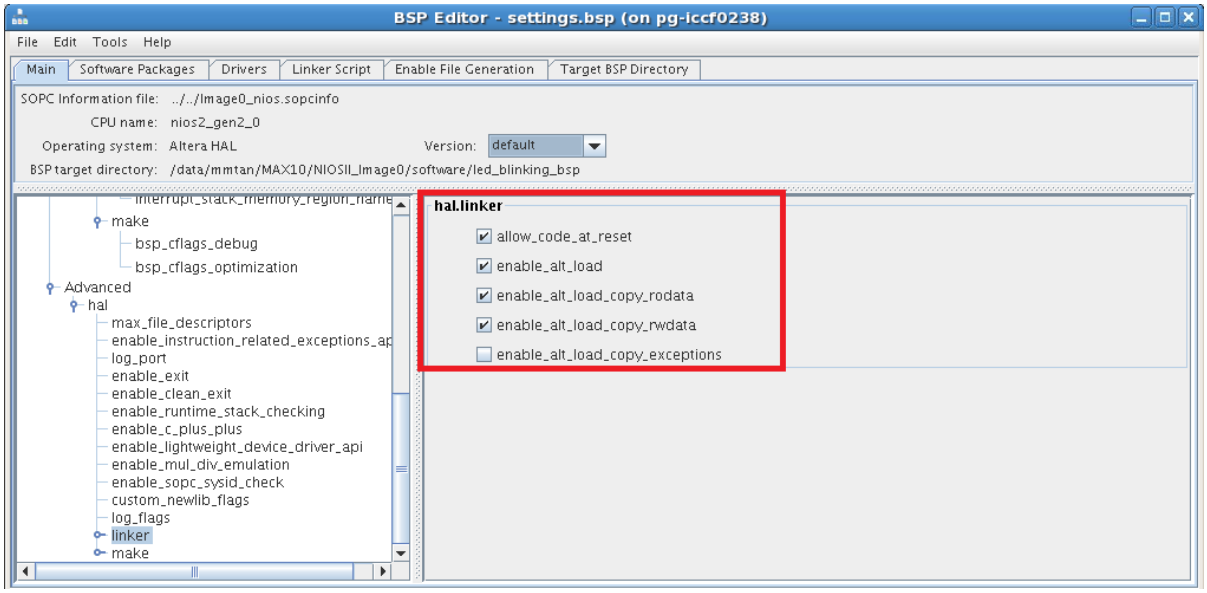
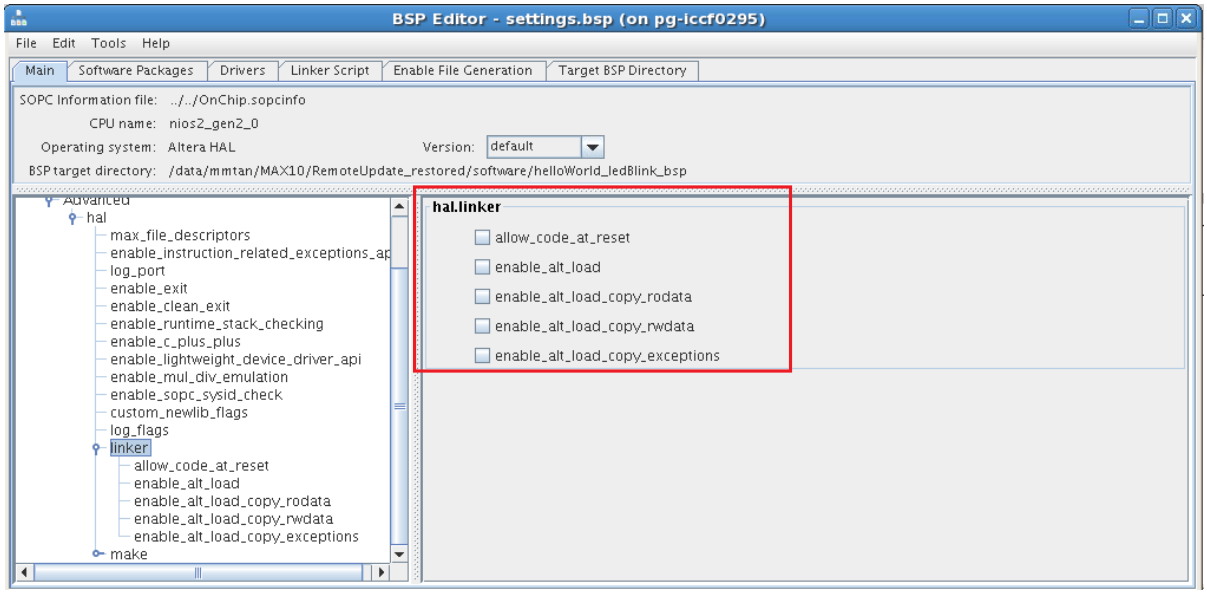


Figure 33: Advanced.hal.linker Settings for Boot Option 2



6. Click on **Linker Script** tab in the Nios II BSP Editor.
7. Based on the boot option used, do one of the following:
   - For boot option 1a and 1b, set the **.text** item in the **Linker Section Name** to the Altera On-chip Flash in the **Linker Region Name**. Set the rest of the items in the **Linker Section Name** list to the Altera On-chip Memory (OCRAM) or external RAM.
   - For boot option 2, set all of the items in the **Linker Section Name** list to Altera On-chip Memory (OCRAM) or external RAM.

Figure 34: Linker Region Settings for Boot Option 1a and 1b

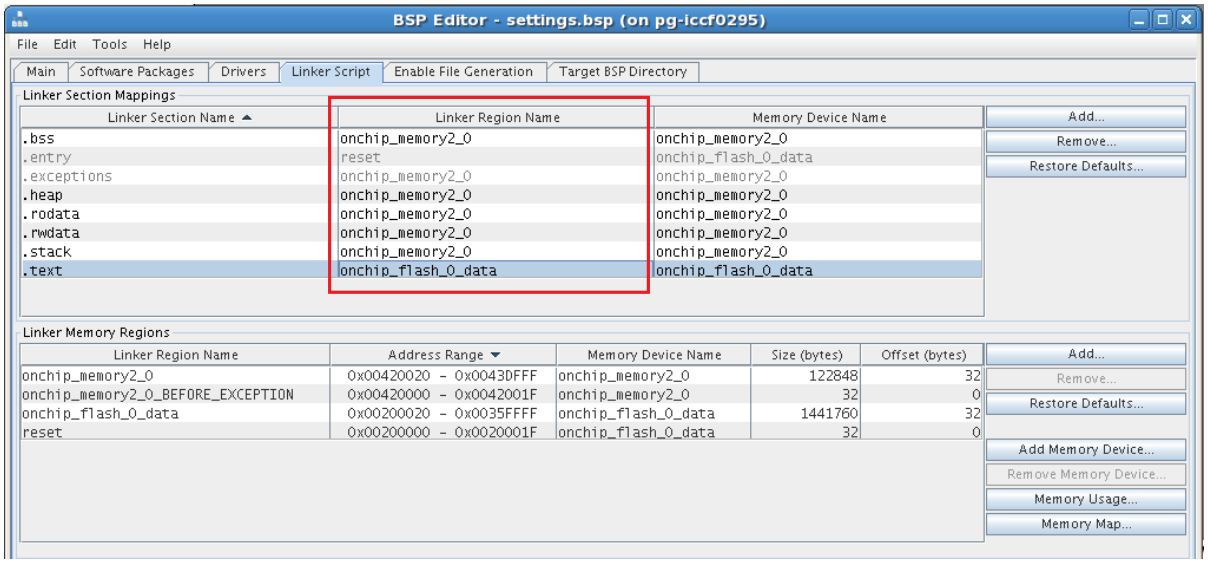
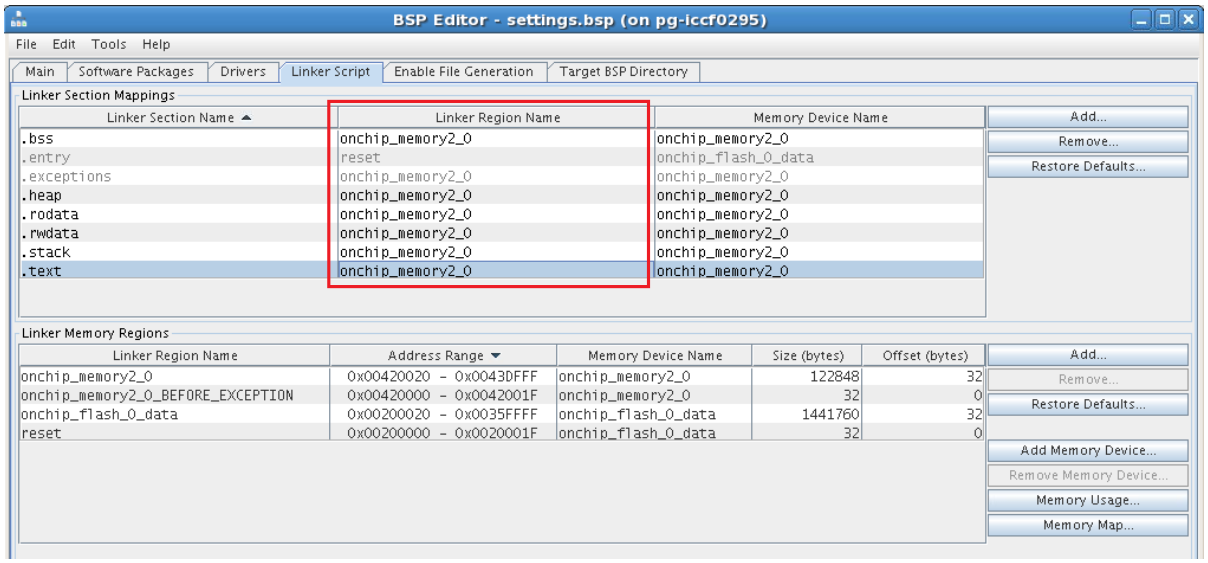


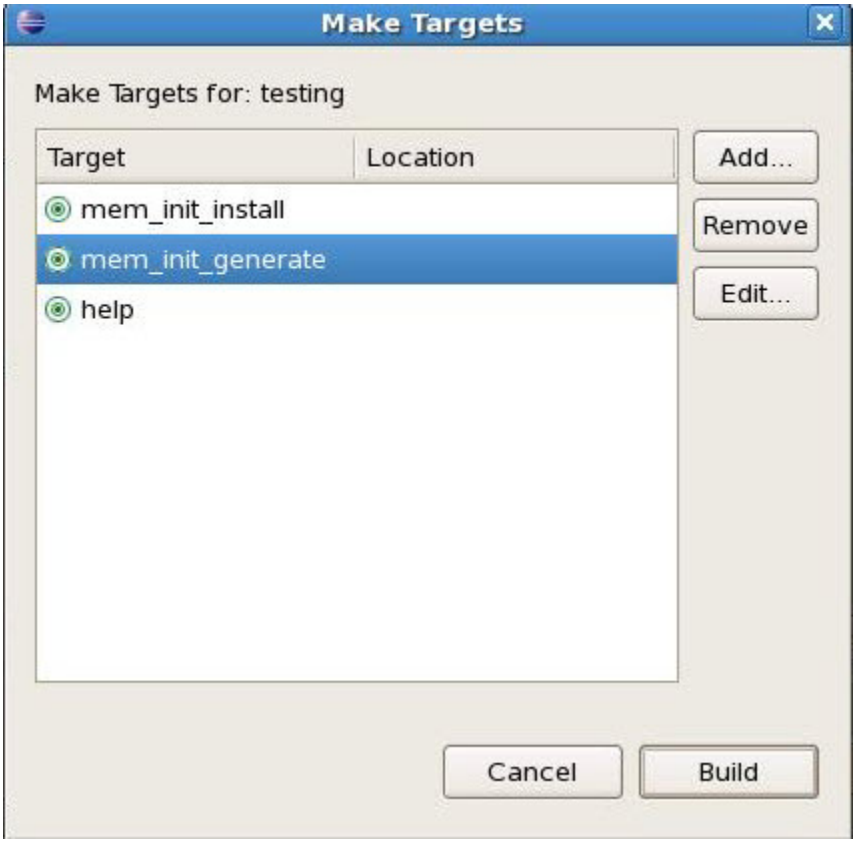
Figure 35: Linker Region Settings for Boot Option 2



HEX File Generation

1. In the Nios II SBT tool, right click on your project in the Project Explorer window.
2. Click **Make Targets** -> **Build…**, the Make Targets dialog box appears. You can also press shift + F9 to trigger the Make Target dialog box.
3. Select **mem_init_generate**.
4. Click **Build** to generate the HEX file.

Figure 36: Selecting mem_init_generate in Make Targets



5. The "mem_init_generate" macro will create two HEX files; **on_chip_ram.hex** and **on_chip_flash.hex**. The **on_chip_ram.hex** will be used for boot option 3 and **on_chip_flash.hex** is used for boot option 1 and 2.

   Note:
   - The **mem_init_generate** target also generates a Quartus II IP file (**meminit.qip**). Quartus II software will refer to the **meminit.qip** for the location of the initialization files.
   - All these files can be found under "<project_folder>/software/<application_name>/mem_init" folder.

6. Recompile your project in Quartus II software if you check **Initialize memory content** option in Altera On-Chip Flash IP. This is to include the software data (.HEX) into the SOF file.

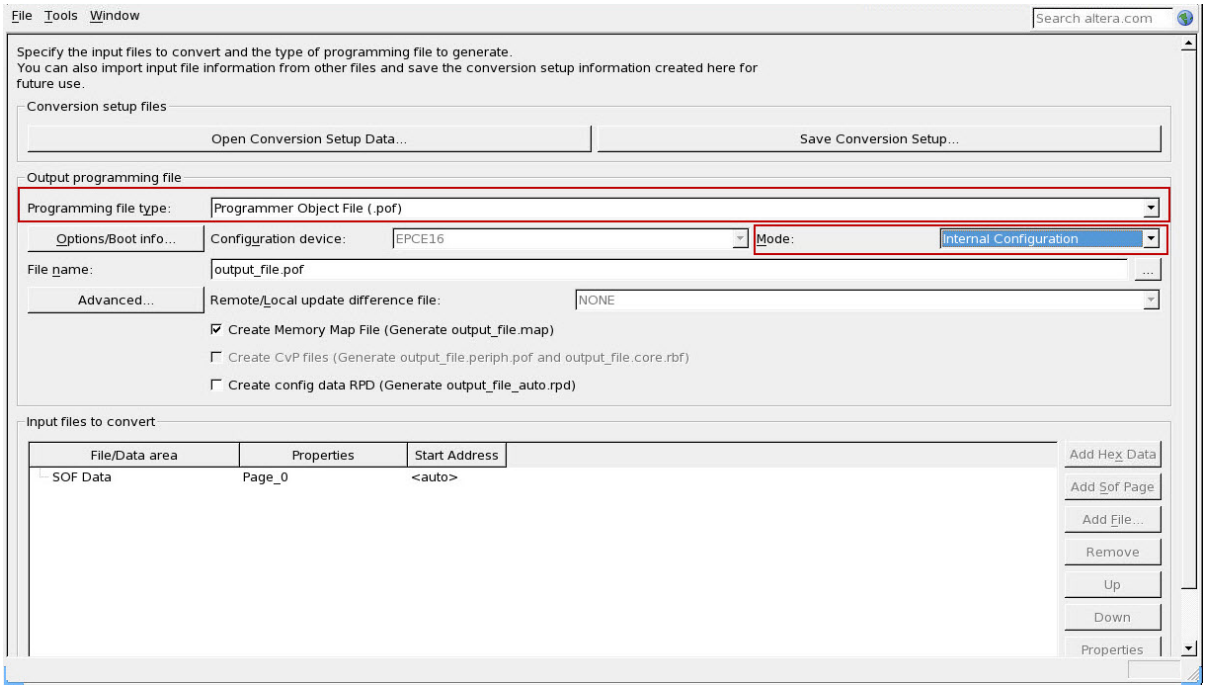
Related Information

- Qsys Settings on page 25
- Quartus II Software Settings on page 30

Programmer Object File (.pof) Generation

1. In Quartus II, click on **Convert Programming Files** from the **File** tab.
2. Choose **Programmer Object File** as **Programming file type:**.
3. Set **Mode** to **Internal Configuration**.

Figure 37: Convert Programming File Settings



4. Click on **Options/Boot info...**, the MAX 10 Device Options dialog box appears.
5. Based on the **Initialize flash content** settings, do one of the following:
   - If **Initialize flash content** was checked, make sure **Page_0** or **Page_1** is selected for **UFM source:** option. Click OK.

     Note: UFM data (.HEX file) can be included in either Page_0 or Page_1 only. The Altera On-chip flash does not support two .HEX files for Dual Compressed images configuration mode.
   - If **Initialize flash content** was not checked, choose **Load memory file** for **UFM source:** option. Browse to the generated Altera On-chip Flash HEX file (**on_chip_flash.hex**) in the **File path:** and click **OK**.

Figure 38: Setting Page_0 or Page_1 for UFM Source If Initialize flash content is Checked

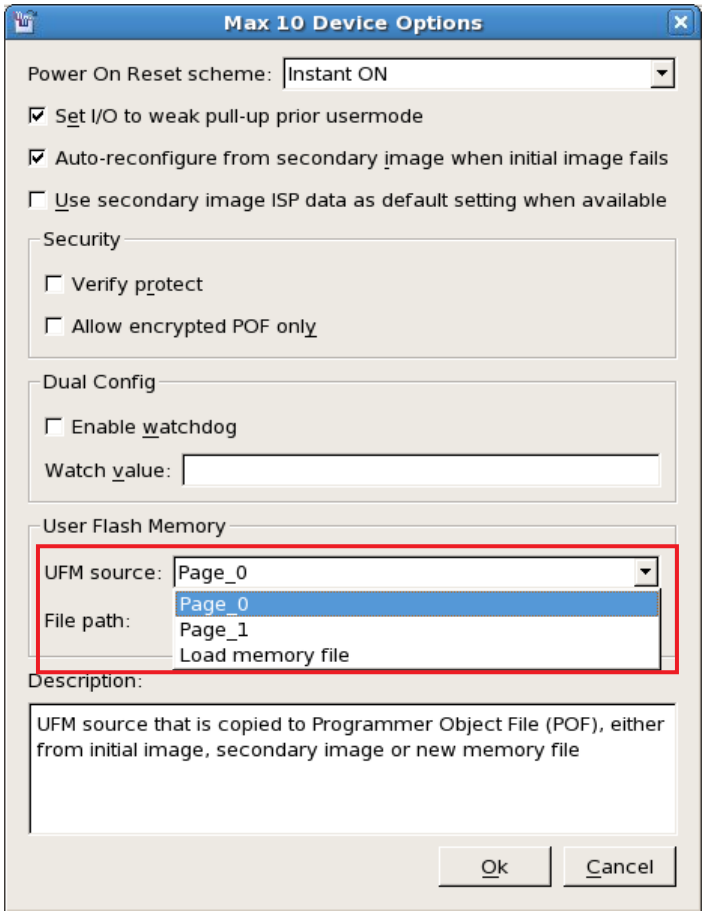
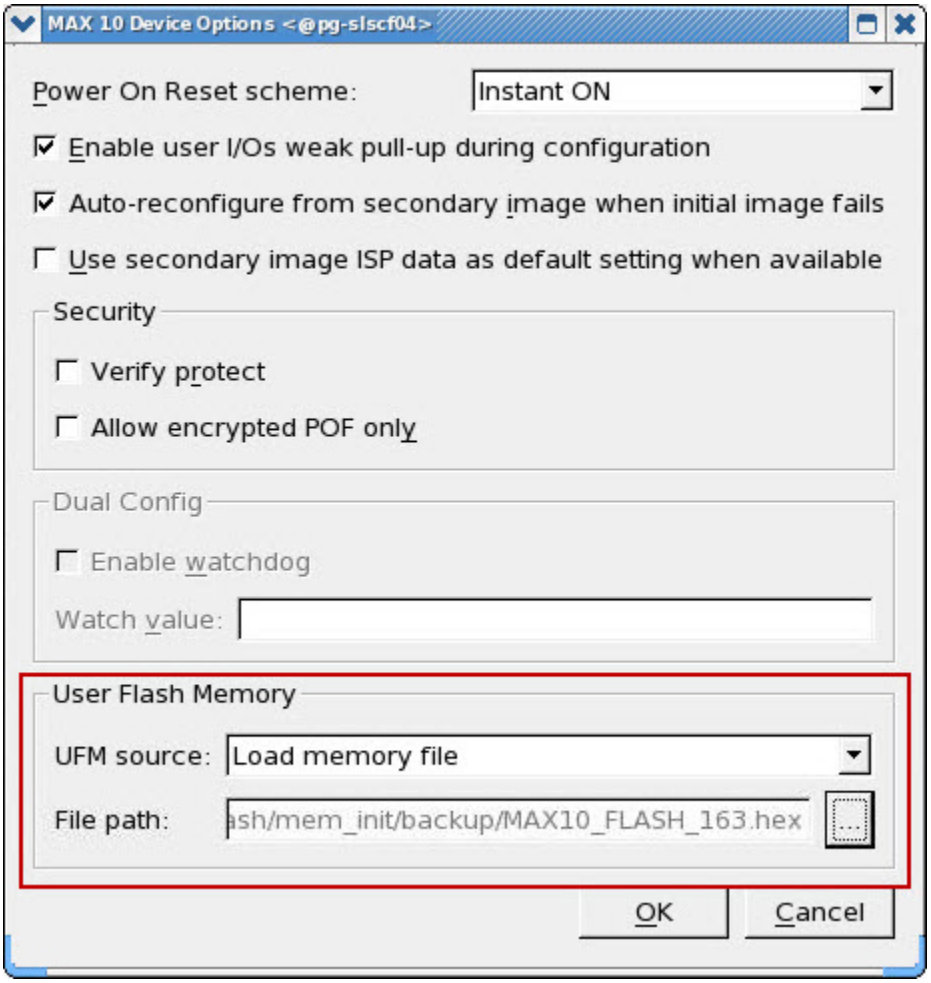
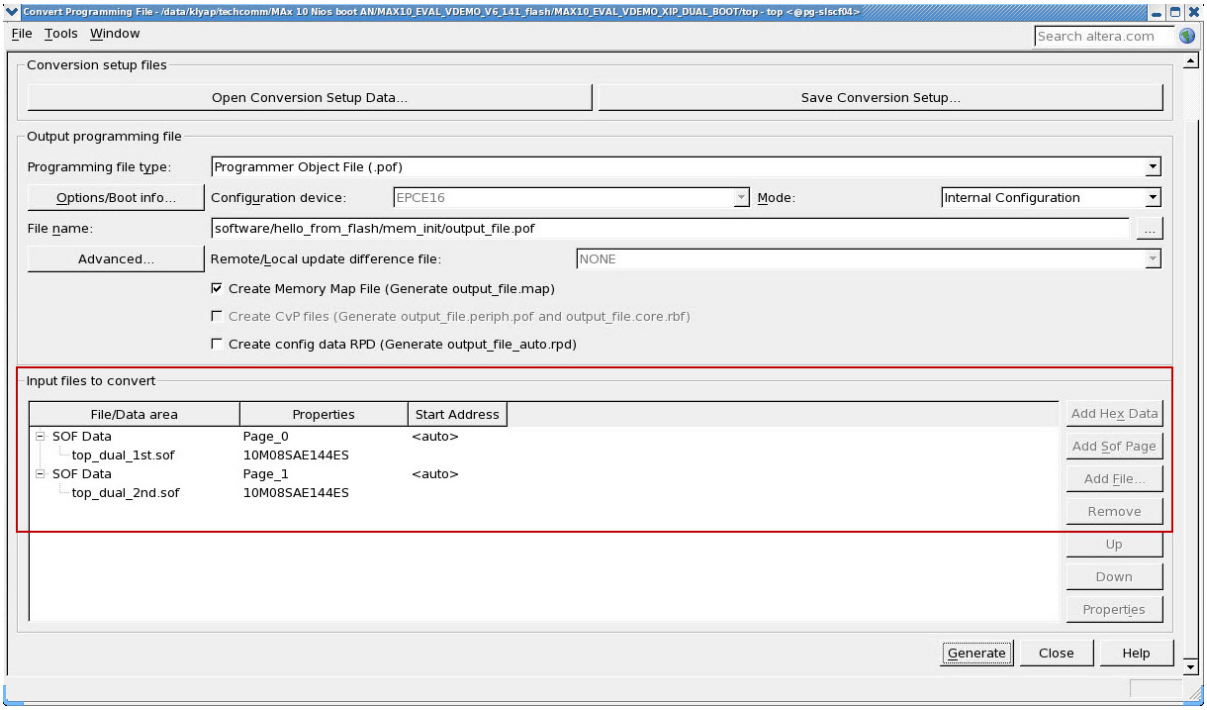


Figure 39: Setting Load Memory File for UFM Source if Initialize flash content is not Checked



6. In the Convert Programming File dialog box, at the **Input files to convert** section, click **Add File...** and point to the first generated Quartus II .sof file to add the **.sof** file at page_0.
7. Click on **Add Sof Page** to create additional page for .sof file. This creates SOF data Page_1 automatically. Click **Add File...** and point to the second generated Quartus II **.sof** file to add the **.sof** file at page_1.

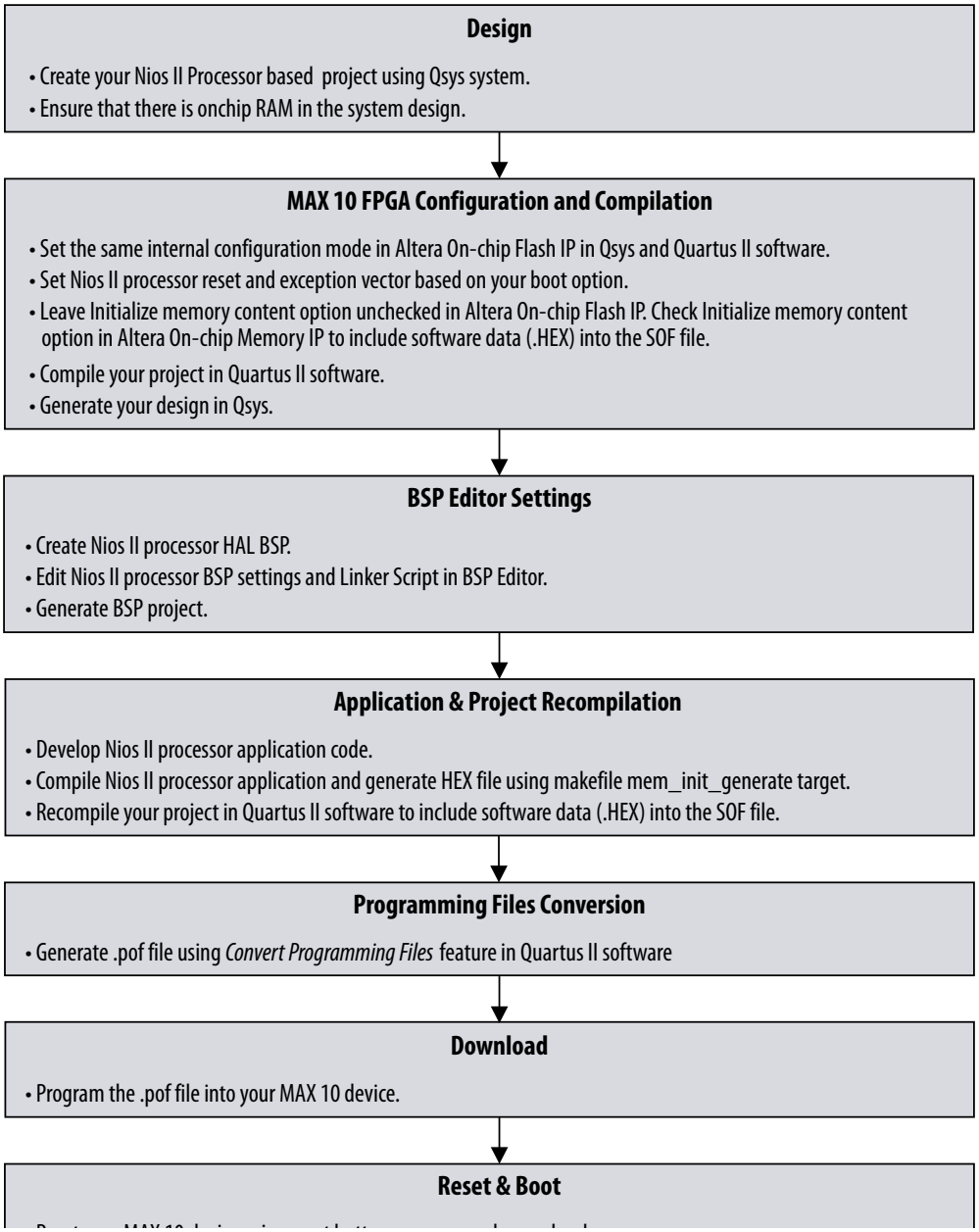
Figure 40: Input Files to Convert in Convert Programming Files for Dual Images Mode



8. Click **Generate** to create the **.pof** file.
9. Program the .pof file into your MAX 10 device.

Boot System Guidelines for Option 3

Figure 41: Configuration and Booting Flow for Option 3

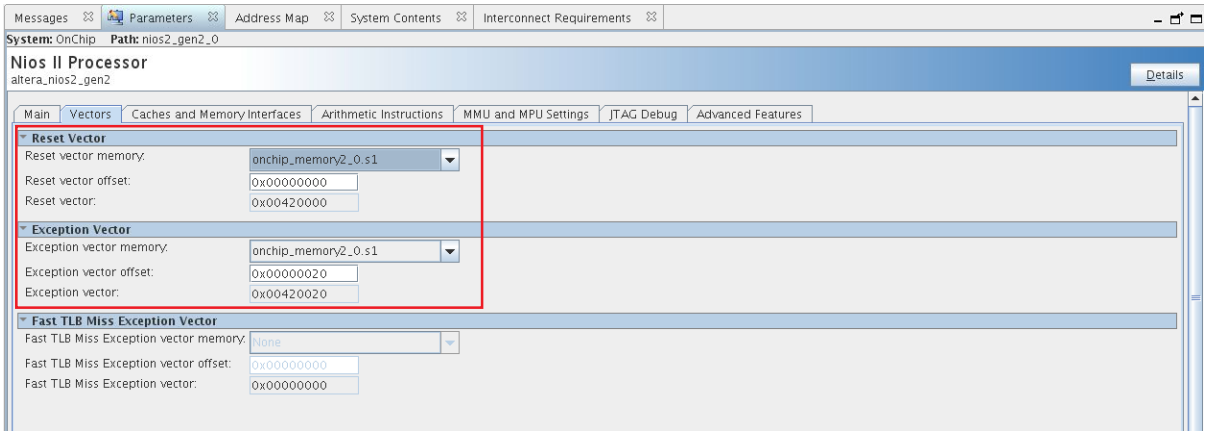


Single Uncompressed/Compressed Image with Memory Initialization Bootable System Guideline

Qsys Settings

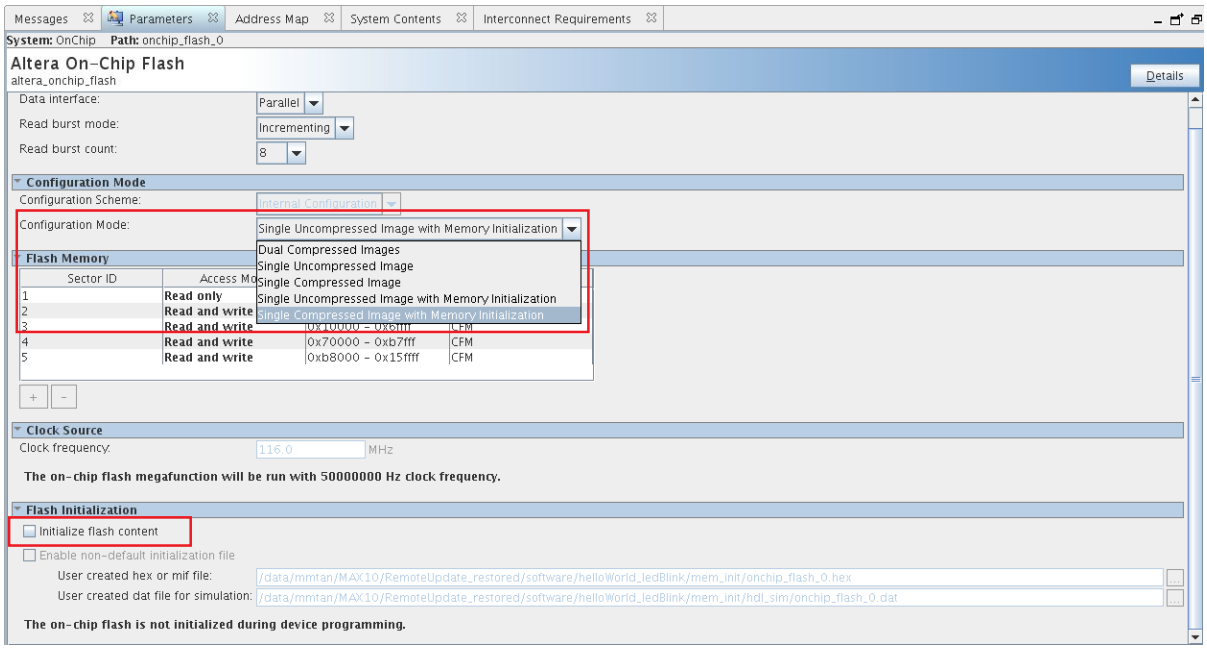
1. In Nios II Gen2 Processor parameter editor, set both the **Reset vector memory:** and **Exception vector memory:** to Altera On-Chip Memory (OCRAM).

Figure 42: Nios II Gen2 Parameter Editor Settings for Boot Option 3



2. In Altera On-chip Flash IP parameter editor, set the **Configuration Mode:** to **Single Uncompressed Image with Memory Initialization** or **Single Compressed Image with Memory Initialization**. Leave the **Initialize flash content** unchecked. This is because the Altera On-chip flash initialization data will not be enabled.

Figure 43: Configuration Mode with Memory Initialization Selection and Initialize Flash Content Setting



3. In the Altera On-chip Memory (RAM or ROM) IP parameter editor, check **Initialize flash content**. If default path is used, add **meminit.qip** generated during "make mem_init_generate" into Quartus II project. Refer to **Figure 45**. Make sure the generated HEX naming matches the default naming. If non-default path is selected, check **Enable non-default initialization file** and specify the path of the HEX file (**onchip_memory2_0.hex**).

   Note: The **meminit.qip** stores the location information of the initialization files.

Figure 44: Enable Initialize Memory Content with Default Initialization File in On-Chip Memory Parameter Editor Settings

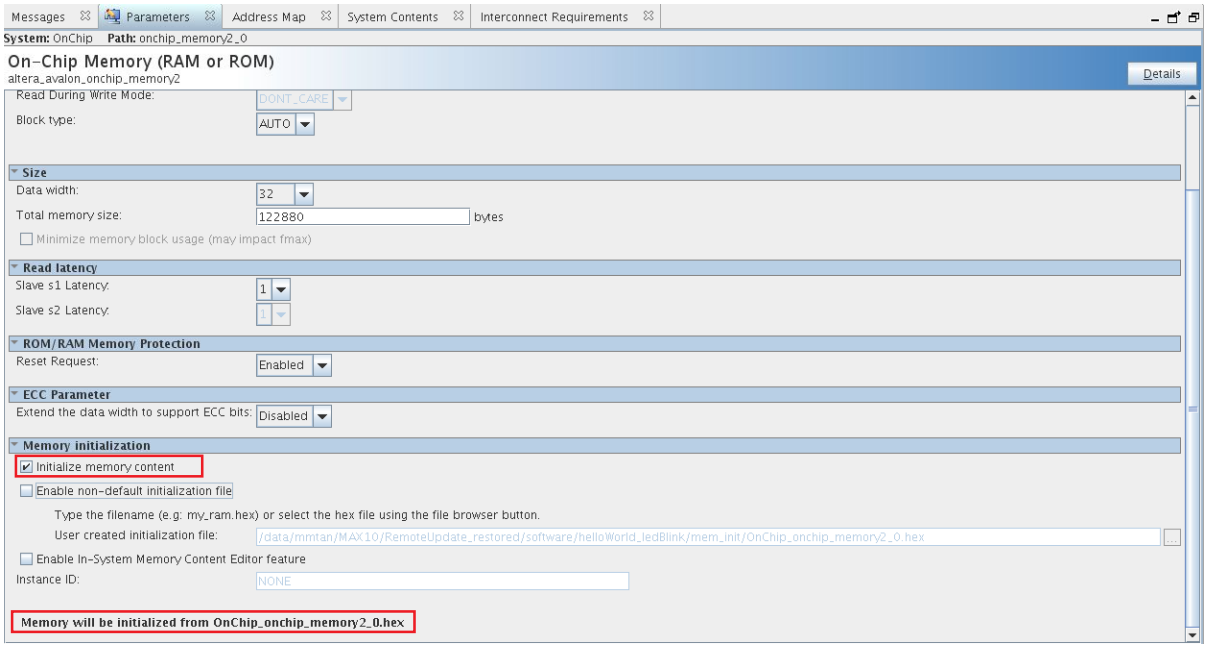


Figure 45: Adding meminit.qip File into Quartus II

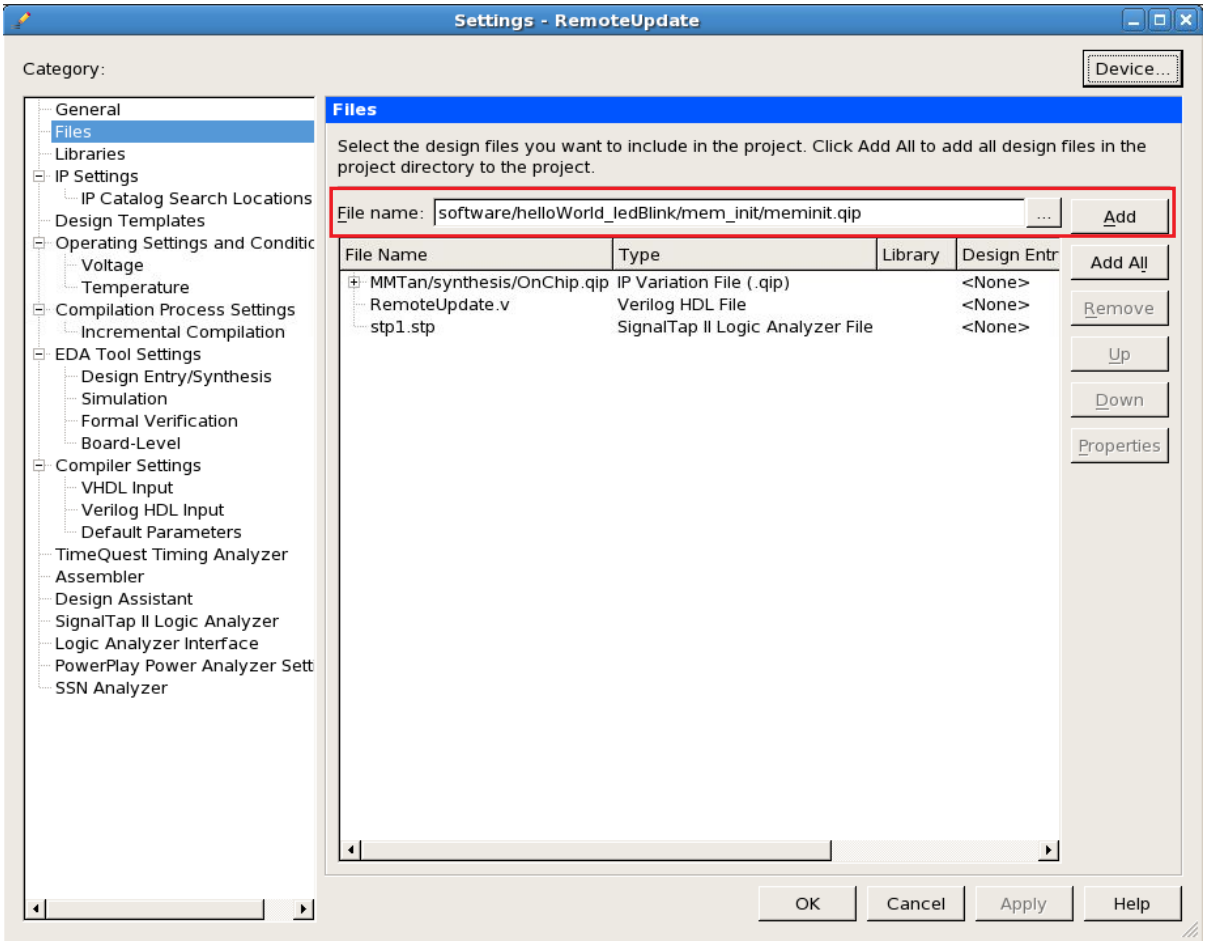
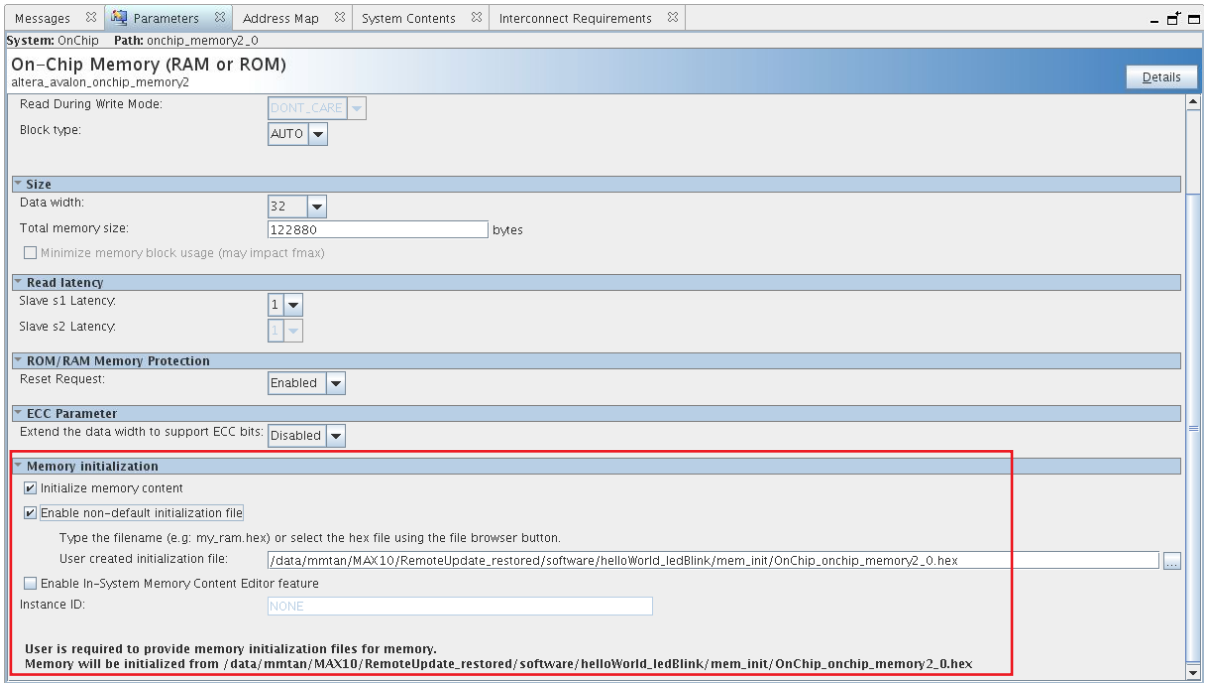
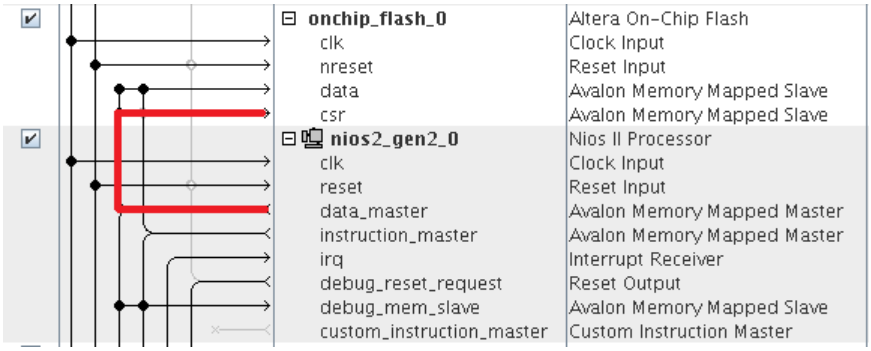


Figure 46: Enable Initialize Memory Content with Non-default Initialization File in On-Chip Memory Parameter Editor Settings



4. Ensure that Altera On-chip Flash CSR port is connected to the Nios II Gen2 processor data master to enable write and erase operations.

Figure 47: CSR Connection to Nios II Gen2 `data_master`



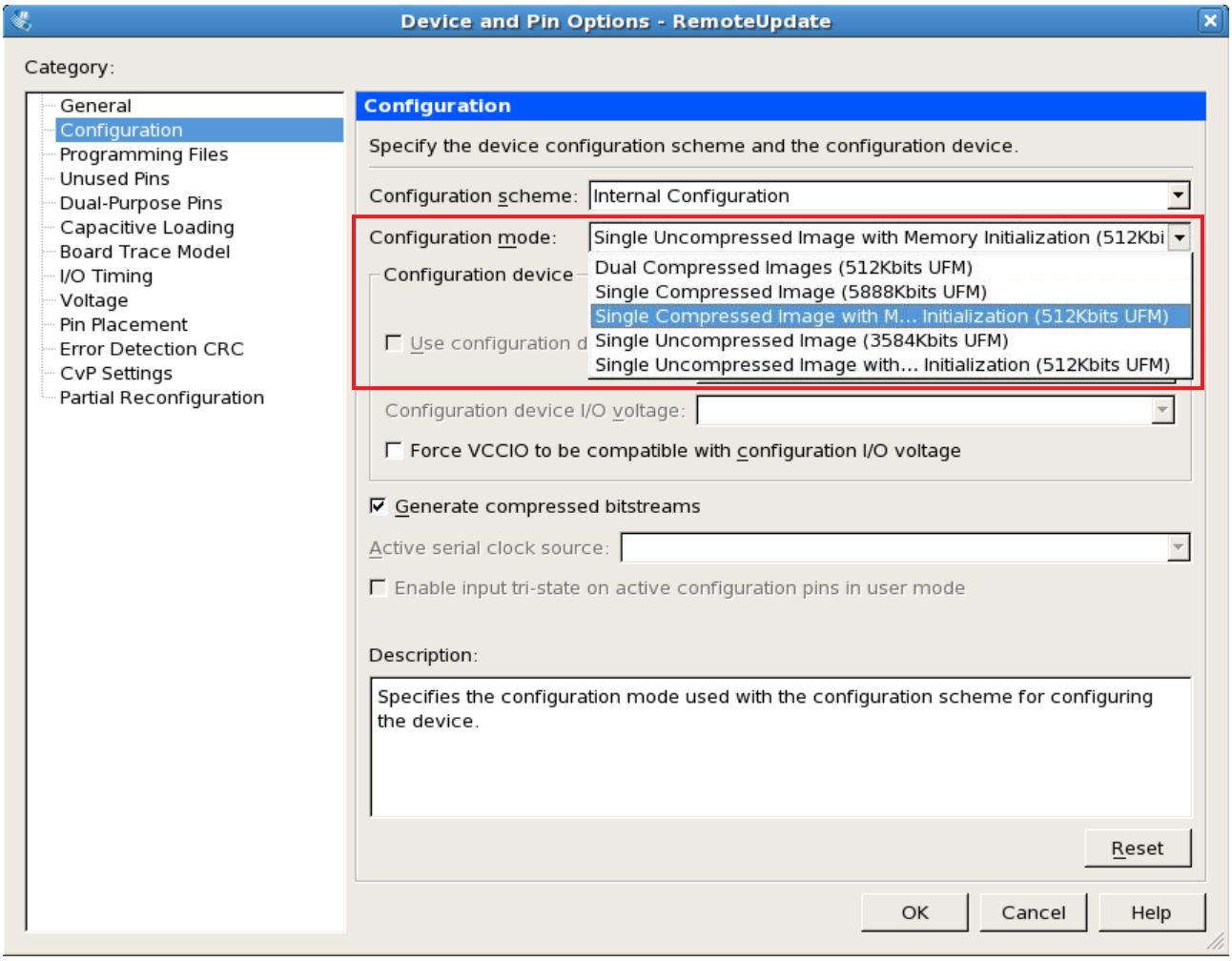
5. Click **Generate HDL**, the **Generation** dialog box appears.
6. Specify output file generation options, and then click **Generate**.

Quartus II Software Settings

1. In Quartus II software, click on **Assignment** -> **Device** -> **Device and Pin Options** -> **Configuration**. Set **Configuration mode:** to **Single Uncompressed Image with Memory Initialization** or **Single Compressed Image with Memory Initialization**.(13)

(13) The size of UFM sector will vary according to your device selection.

Figure 48: Configuration Modes with Memory Initialization Selection in Quartus II



Note:

- If the configuration mode setting in Quartus II software and Qsys parameter editor is different, the Quartus II project compilation will fail with the following error message.

```
14740 Configuration Mode parameter on atom "ufm_block" is inconsistent with Quartus II project setting.
14740 MAX address parameter on atom "ufm_block" is inconsistent with Quartus II project setting.
      Quartus II 64-Bit Fitter was unsuccessful. 2 errors, 0 warnings
293001 Quartus II Full Compilation was unsuccessful. 4 errors, 10 warnings
```

- If internal configuration mode without memory initialization is selected with **Initialize flash content** checked in Altera On-chip Memory IP, the Quartus II project compilation will fail with the following error message:

```
16031 Current Internal Configuration mode does not support memory initialization or ROM. Select Internal Configuration mode with ERAM.
16031 Current Internal Configuration mode does not support memory initialization or ROM. Select Internal Configuration mode with ERAM.
16031 Current Internal Configuration mode does not support memory initialization or ROM. Select Internal Configuration mode with ERAM.
16031 Current Internal Configuration mode does not support memory initialization or ROM. Select Internal Configuration mode with ERAM.
21057 Implemented 7144 device resources after synthesis - the final resource count might be different
      Quartus II 64-Bit Analysis & Synthesis was unsuccessful. 32 errors, 17 warnings
293001 Quartus II Full Compilation was unsuccessful. 34 errors, 17 warnings
```

2. Click **OK** to exit the **Device and Pin Options window**.
3. Click **OK** to exit the **Device** window.
4. Click **Start Compilation** to compile your project and generate the **.sof** file.

BSP Editor Settings

For boot option 3, you must leave both the BSP Editor settings to default; **Advanced.hal.linker** unchecked and all **Linker Section Regions** set to Altera On-Chip Memory (OCRAM) because no boot copier is needed for this boot option.

Figure 49: Advanced.hal.linker Default Settings

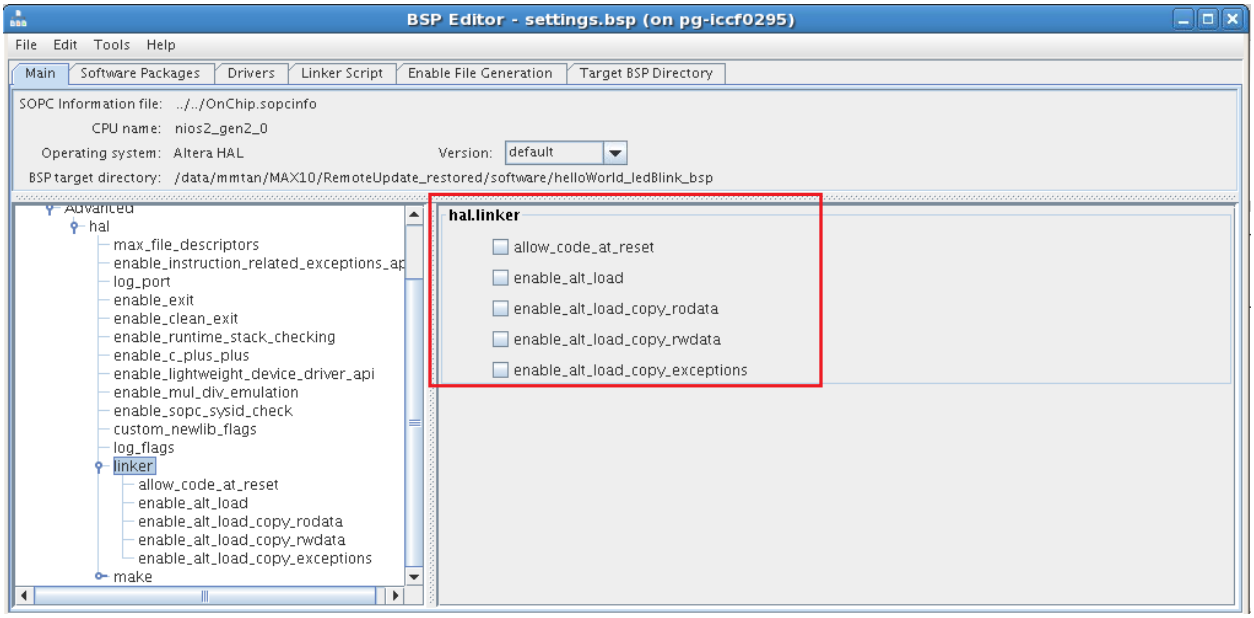
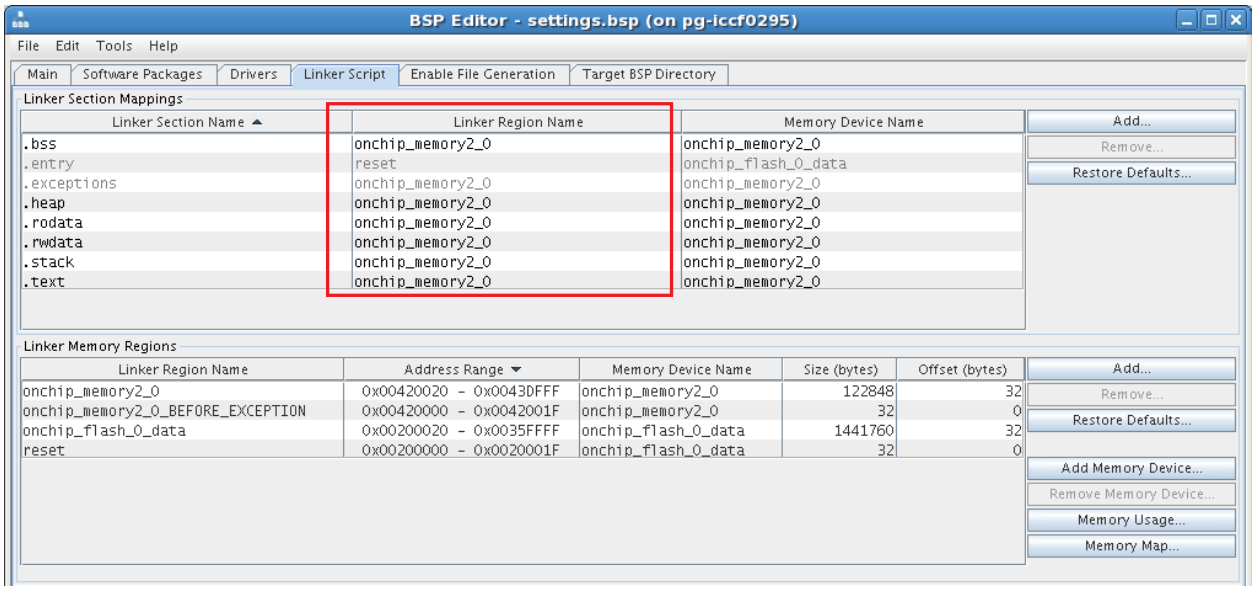


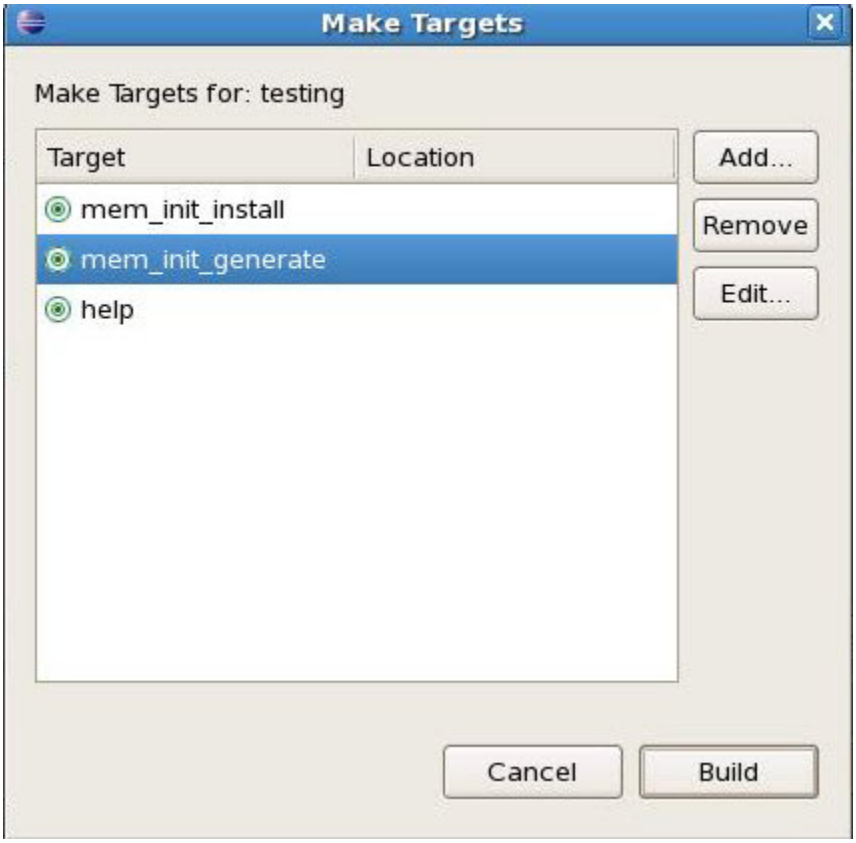
Figure 50: Linker Region Default Settings



HEX File Generation

1. In the Nios II SBT tool, right click on your project in the Project Explorer window.
2. Click **Make Targets** -> **Build…**, the Make Targets dialog box appears. You can also press shift + F9 to trigger the Make Target dialog box.
3. Select **mem_init_generate**.
4. Click **Build** to generate the HEX file.

Figure 51: Selecting mem_init_generate in Make Targets



5. The "mem_init_generate" macro will create two HEX files; **on_chip_ram.hex** and **on_chip_flash.hex**. The **on_chip_ram.hex** will be used for boot option 3 and **on_chip_flash.hex** is used for boot option 1 and 2.

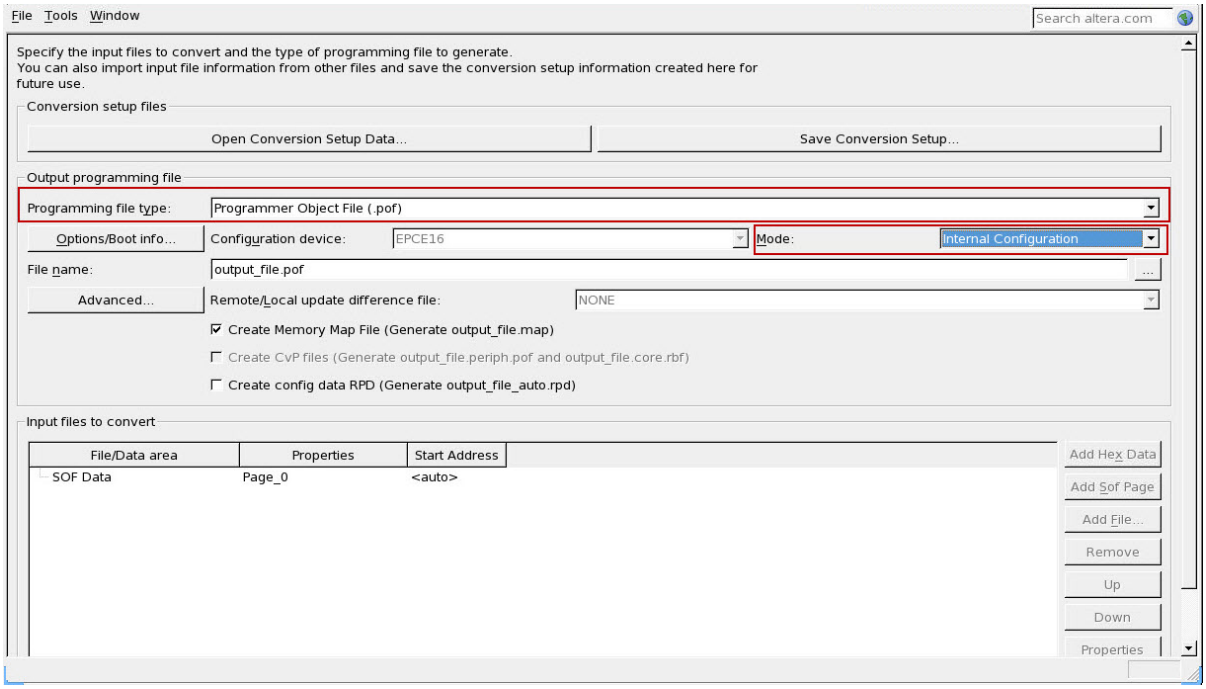
   Note:
   - The **mem_init_generate** target also generates a Quartus II IP file (**meminit.qip**). Quartus II software will refer to the **meminit.qip** for the location of the initialization files.
   - All these files can be found under "<project_folder>/software/<application_name>/mem_init" folder.

6. Recompile your project in Quartus II software to include the software data (.HEX) into the SOF file.

Programmer Object File (.pof) Generation

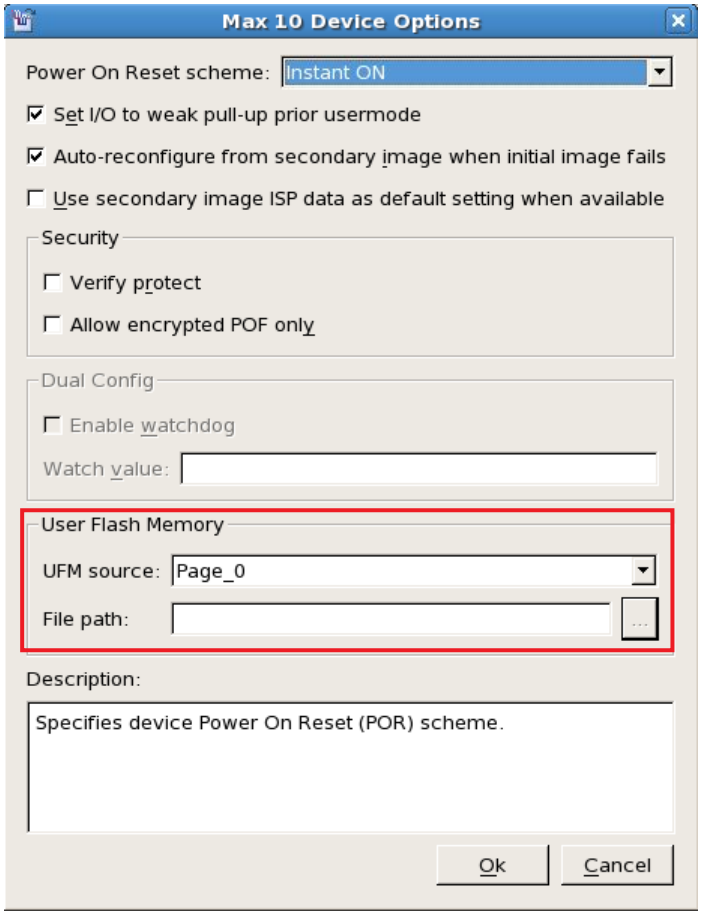
1. In Quartus II, click on **Convert Programming Files** from the **File** tab.
2. Choose **Programmer Object File** as **Programming file type:**.
3. Set **Mode** to **Internal Configuration**.

Figure 52: Convert Programming File Settings



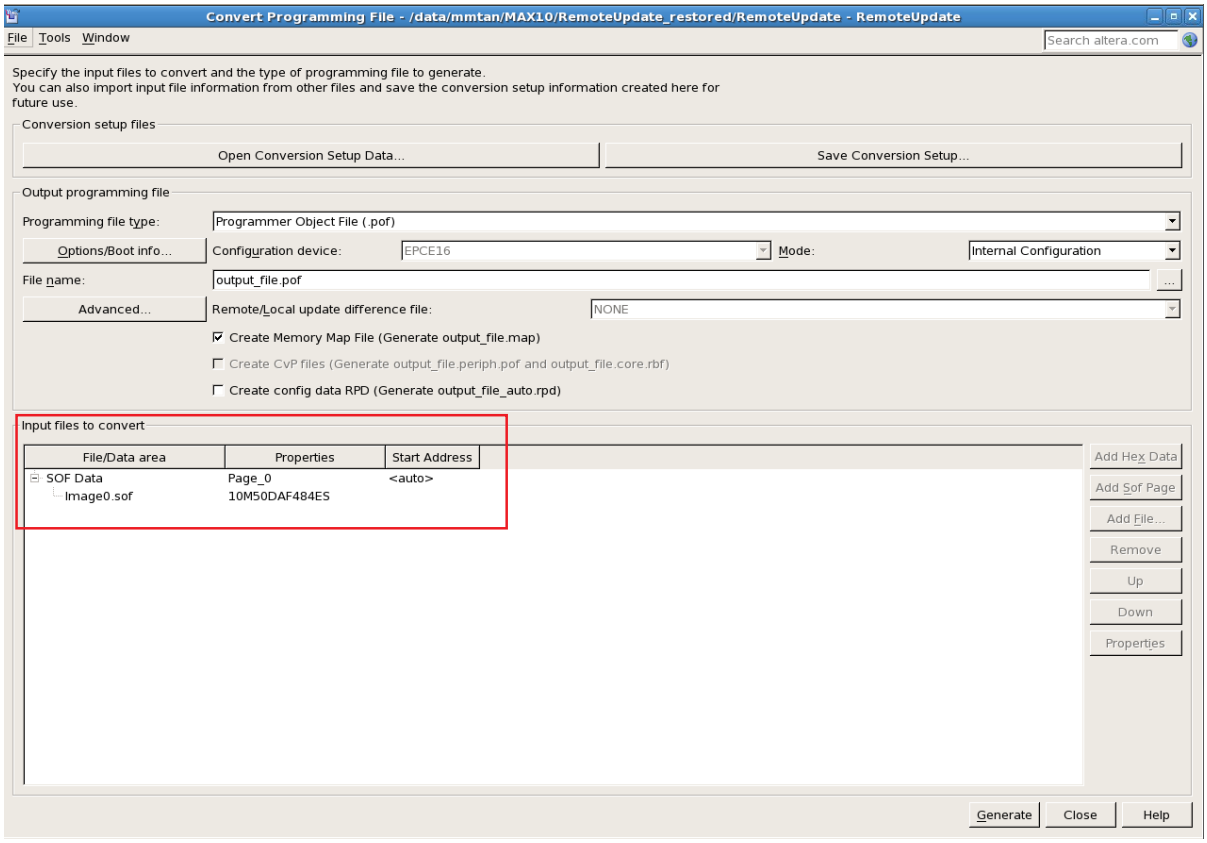
4. Click on **Options/Boot info...**, the MAX 10 Device Options dialog box appears.
5. Make sure Page_0 is set as the **UFM source:** option. Click **OK**.

Figure 53: Setting Page_0 for UFM Source



6. In the Convert Programming File dialog box, at the **Input files to convert** section, click **Add File...** and point to the generated Quartus II **.sof** file to add the **.sof** file at **page_0**.

Figure 54: Input Files to Convert in Convert Programming Files



7. Click **Generate** to create the **.pof** file.
8. Program the **.pof** file into your MAX 10 FPGA device.

Summary of Nios II Processor Vector Configurations and BSP Settings

The following table shows a summary of Nios II processor reset and exception vector configurations, and BSP settings.

Table 8: Summary of Nios II Processor Vector Configurations and BSP Settings

Boot Option	Reset Vector Configuration	Exception Vector Configuration	BSP Editor Setting: Settings.Advanced.hal.linker	BSP Editor Setting: Linker Script
Option 1: Nios II processor application execute-in-place from Altera On-chip Flash (UFM)	Altera On-chip Flash	1. OCRAM/ External RAM, **OR** 2. Altera On-chip Flash	If the exception vector memory is set to OCRAM/ External RAM, enable the following settings in **Settings.Advanced.hal.linker**: • allow_code_at_reset • enable_alt_load • enable_alt_load_copy_rodata • enable_alt_load_copy_rwdata • enable_alt_load_copy_exceptions If the exception vector memory is set to Altera On-chip Flash, enable the following settings in **Settings.Advanced.hal.linker**: • allow_code_at_reset • enable_alt_load • enable_alt_load_copy_rodata • enable_alt_load_copy_rwdata	• Set **.text** Linker Section to Altera On-chip Flash • Set other Linker Sections (**.heap, .rwdata, .rodata, .bss, .stack**) to OCRAM/ External RAM
Option 2: Nios II processor application copied from UFM to RAM using boot copier	Altera On-chip Flash	OCRAM/ External RAM	Make sure all settings in **Settings.Advanced.hal.linker** are left unchecked	Make sure all Linker Sections are set to OCRAM/ External RAM
Option 3: Nios II processor application execute-in-place from Altera On-chip Memory (OCRAM)	OCRAM	OCRAM	Make sure all settings in **Settings.Advanced.hal.linker** are left unchecked	Make sure all Linker Sections are set to OCRAM

Booting Elements

All Nios II processor boot options introduced in this application note uses the following booting elements to create the necessary boot files:

- The memcpy-based boot copier
- The alt_load function
- The Nios II SBT "make mem_init_generate" target
- The Convert Programming Files feature

Nios II Processor Memcpy-based Boot Copier

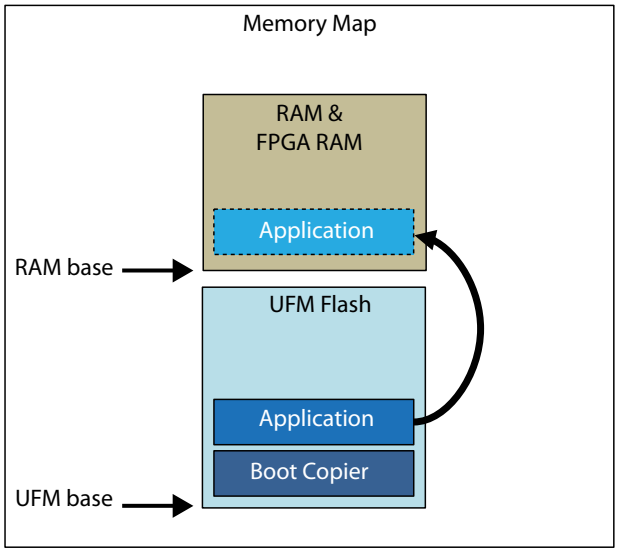
The Nios II processor memcpy-based boot copier has the following features:

- Supports EPCQ, CFI and Altera On-chip Flash (UFM) flash memories
- Locates software application in the memory
- Unpacks and copies software application image to RAM
- Automatically switches to application code in RAM after copy completes

The memcpy-based boot copier is used to support boot option 2 (Nios II soft processor application copied from UFM to RAM using boot copier). The memcpy-based boot copier is automatically appended into the HEX file during memory initialization file generation ("mem_init_generate" target). When you download your **.pof** file into the FPGA, the boot copier will be placed at the beginning of the UFM sector and the software application will be placed at the end of the boot copier.

The function of the memcpy-based boot copier is to copy the software application image to the RAM which is the entry point indicated by the software application (ELF file). Depending on the system design, RAM can be either OCRAM or external RAM . Once the copying is done, the boot copier will pass the system control to the application in RAM.

Figure 55: Memory Map of An Example System Using Default Boot Copier



The alt_load() function

The alt_load() function is a mini-bootcopier included in the HAL code and provides the following capabilities:

- Optionally copies sections from the boot memory to RAM based on BSP settings.
- Able to copy data sections (**.rodata**, **.rwdata**, **.exceptions**) to RAM but not the code sections (**.text**).

The alt_load() function can be enabled in the BSP Settings as shown in the following table:

Table 9: BSP Settings and the alt_load() Functions

BSP Settings	Functions
hal.linker.enable_alt_load	enable alt_load() function
hal.linker.enable_alt_load_copy_rodata	alt_load() copies .rodata section to RAM
hal.linker.enable_alt_load_copy_rwdata	alt_load() copies .rwdata section to RAM
hal.linker.enable_alt_load_copy_exceptions	alt_load() copies .exceptions section to RAM

Nios II SBT Makefile "mem_init_generate" Target

The Nios II SBT application Makefile "mem_init_generate" target is responsible for generating memory initialization files using various file conversion tools. This includes a HEX file for the UFM data, a HEX file for initialization of the on chip RAM in the SOF and a DAT file for initializing the on chip flash model for simulation.

When required, the Nios II SBT tool automatically adds the Nios II processor memcpy-based boot copier to the system when the executable file (**.elf**) is converted to memory initialization file (**.hex**). This operation take place whenever the **.text** section is located in a different memory that the reset vector points to, which indicates a code copy is required. The file conversion happens during execution of "make mem_init_generate" target.

The "make mem_init_generate" target generates different HEX file content based on the specified boot options:

- For boot option 1 & 3, the generated HEX file contains ELF loadable section.
- For boot option 2, the generated HEX file contains the boot copier and the ELF payload.

The **mem_init_generate** target also generates a Quartus II IP file (**meminit.qip**). Quartus II software will refer to the **meminit.qip** for the location of the initialization files.

The Convert Programing Files Option

You can use the Convert Programming Files option in Quartus II software to convert programming files from one file format to another. This tool is used for combining a **.sof** and a HEX file into a single **.pof** file for programming into the Altera On-chip Flash.

MAX 10 FPGA Nios II Design Boot Time Estimation and Guidance

Altera has performed MAX 10 FPGA boot time performance analysis based on a few scenarios. You may refer to the information in following tables as guidance when designing your customized design.

Table 10: Boot Time Analysis Scenario: Execute-in-place

Test Case	Boot From	Boot Time Counter (Clock cycle)
Design without External Memory[14]	On Chip Flash[15]	41,000
	On Chip Memory[16]	18,000
Design with External Memory	On Chip Flash	2,000,000
	On Chip Memory	2,000,000

Note: The huge clock cycle difference between Design with External Memory and Design without External Memory is caused by the external memory calibration.

Table 11: Boot Time Analysis Scenario: Using Boot Copier

Test Case	Boot from	Run From	Boot Time Counter (Clock cycle)
Design without External Memory	On Chip Flash	On Chip Memory	2,800,000
Design with External Memory	On Chip Flash	External Memory	4,700,000
	On Chip Flash	On Chip Memory	2,800,000

Note: The clock cycle difference between Design with External Memory and Design without External Memory is insignificant, this is because the time needed for boot copier outweigh the time needed for external memory calibration.

Table 12: Boot Time Analysis Scenario: Effects of CPU Caches

Test Case	Boot From	Boot Time Counter
Execute-in-place	On Chip Flash	0-1% reduction
	On Chip Memory	0-1% reduction
Boot Copier	On Chip Flash	29-42% reduction

Table 13: Boot Time Analysis Scenario: Design with Different System Clock Speed

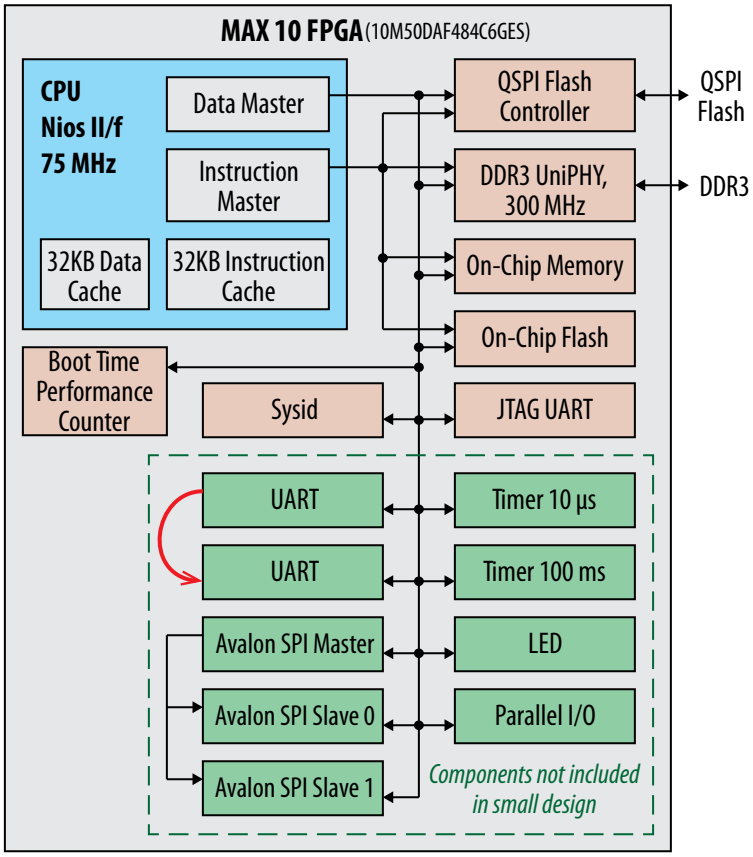
Boot From	Boot Time Counter
On Chip Flash	Directly proportional to system frequency

The following diagram shows the large design used for boot time performance analysis. For small design, components in green in the diagram are not included.

[14] Also known as External RAM
[15] Also known as UFM
[16] Also known as OCRAM

Figure 56: Boot Time Performance Analysis Design Block Diagram

Note:
1. The Boot Time Performance Analysis is done using MAX10 10M50 Development Kit (10M50DAF484C6GES).
2. For External Memory calibration to boot time counter analysis, the hardware design without cache is used:
   - The large design with exception vector set to External Memory
   - The small design with exception vector set to External Memory
   - The large design with exception vector set to On Chip Memory
   - The small design without External Memory
3. For CPU caches analysis, the hardware design uses large design with 32kB data cache and 32kB instruction cache and also another large design without the CPU caches.
4. Design with different system clock speed analysis sets the Nios II frequency to 75 MHz and 150 MHz for Slow and Fast design respectively. This analysis was executed using small design.
5. The software application uses "hello world small software example" that had been modified to have an elf size of around 32kB for all the test cases above.



Document Revision History

This table lists the revision history for this application note.

Table 14: Document Revision History

Date	Version	Changes
June 2015	2015.06.15	• Added *MAX 10 FPGA Nios II Design Boot Time Estimation and Guidance* section containing boot time performance analysis. • Added *The alt_load() function* table. • Added ROM Size Requirement to *RAM and ROM Size Requirement For Each Boot Option* table. • Added block diagrams for all boot options in *Nios II Processor Booting Options Using On-chip Flash*. • Added OCRAM size in *UFM and CFM Array Size* table. • Added Boot Option: 3 Nios II processor application execute in-place from Altera On-chip Memory. • Updated 'Configuration and Booting Flow' to include Boot Option: 3. • Updated Steps to Build a Bootable System to Guidelines to Build a Bootable System. • Updated Single Uncompressed/Compressed Image Bootable System Guideline to support Single Compressed Image Mode. • Added the third guideline; 'Single Uncompressed/Compressed Image with Memory Initialization Bootable System Guideline'. • Editorial changes.
January 2015	2015.01.23	Initial release.