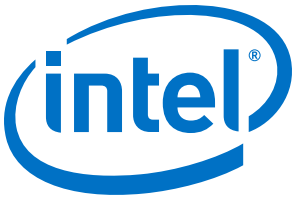


AN 741: Remote System Upgrade for MAX 10 FPGA Devices over UART with the Nios II Processor

AN-741
2017.02.21


Subscribe


Send Feedback

Contents

1 Remote System Upgrade for MAX 10 FPGA Devices over UART with the Nios II Processor

The reference design provides a simple application that implements basic remote configuration features in Nios II-based systems for MAX 10 FPGA devices. The UART interface included in the MAX 10 FPGA Development Kit is used together with Altera UART IP core to provide the remote configuration functionality.

Related Links

Reference Design Files

1.1 Remote System Upgrade with MAX 10 FPGA Overview

With the remote system upgrade feature, enhancements and bug fixes for FPGA devices can be done remotely. In an embedded system environment, firmware needs to be updated frequently over the various type of protocol, such as UART, Ethernet, and I2C. When the embedded system includes an FPGA, firmware updates can include updates of the hardware image on the FPGA.

MAX10 FPGA devices provide the capability to store up to two configuration images which further enhance the remote system upgrade feature. One of the images will be the back up image that is loaded if an error occurs in the current image.

1.2 Abbreviations

Table 1. List of Abbreviations

Abbreviation	Description
Avalon-MM	Avalon Memory-Mapped
CFM	Configuration flash memory
GUI	Graphical user interface
ICB	Initialization Configuration Bit
MAP/`.map`	Memory Map File
Nios II EDS	Nios II Embedded Design Suite Support
PFL	Parallel Flash Loader IP core
POF/`.pof`	Programmer Object File
QSPI	Quad serial peripheral interface
RPD/`.rpd`	Raw programming data
SBT	Software Build Tools

continued...

Abbreviation	Description
SOF/.sof	SRAM Object File
UART	Universal asynchronous receiver/transmitter
UFM	User flash memory

1.3 Prerequisite

The application of this reference design requires you to have the indicated level of knowledge or experience in the following areas:

- Working knowledge of Nios II systems and the tools to build them. These systems and tools include the Quartus® II software, Qsys, and the Nios II EDS.
- Knowledge of Intel FPGA configuration methodologies and tools, such as the MAX 10 FPGA internal configuration, remote system upgrade feature and PFL.

1.4 Requirements

The following are the hardware and software requirements for the reference design:

- MAX 10 FPGA development kit
- Quartus II version 15.0 with Nios II EDS
- A computer with a working UART driver and interface
- Any binary/hexadecimal file editor

1.5 Reference Design Files

Table 2. Design Files Included in the Reference Design

File Name	Description
Factory_image	• Quartus II hardware design file to be stored in CFM0. • The fallback image/factory image to be used when error occurs in the application image download.
app_image_1	• Quartus II hardware design file to be stored in CFM1 and CFM2.[1] • The initial application image loaded in the device.
app_image_2	Quartus II hardware design file that replaces app_image_2 during remote system upgrade.
Remote_system_upgrade.c	Nios II software application code acting as the controller for the remote upgrade system design.
Remote Terminal.exe	• Executable file with a GUI. • Functions as the terminal for host to interact with MAX 10 FPGA development kit. • Sends programming data through UART. • Source code for this terminal is included.

1 In dual configuration images configuration mode, CFM1 and CFM2 are combined to a single CFM storage.

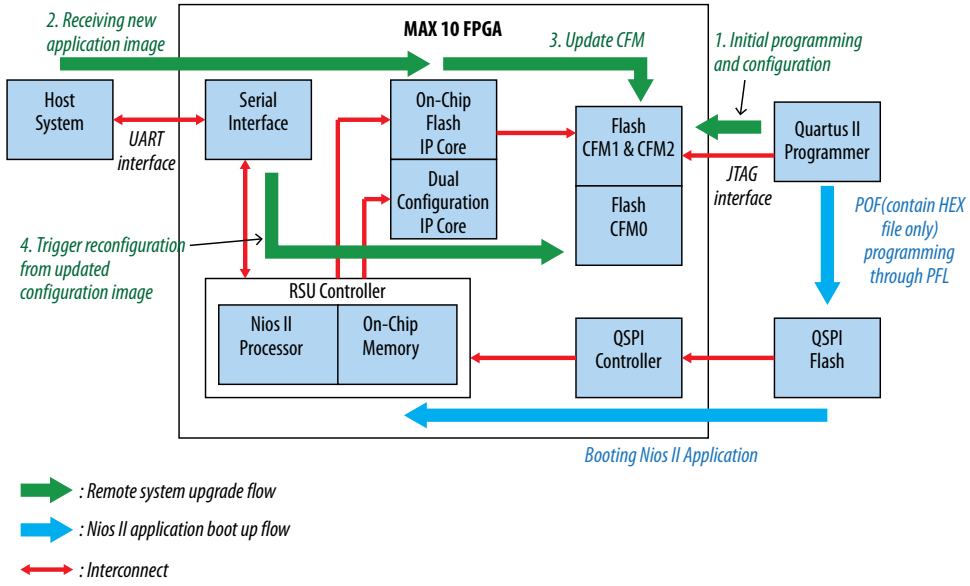
Table 3. Master Files Included in the Reference Design

You can use these master files for the reference design without compiling the design files.

File Name	Description
`factory_application1.pof` `factory_application1.rpd`	Quartus II programming file that consists of factory image and application image 1, to be programmed into CFM0 and CFM1 & CFM2 respectively at initial stage.
`factory_application2.pof` `factory_application2.rpd`	• Quartus II programming file that consists of factory image and application image 2. • Application image 2 will be extracted later to replace application image 1 during remote system upgrade, named `application_image_2.rpd` below.
`application_image_1.rpd`	Quartus II raw programming data file that contain application image 1 only.
`application_image_2.rpd`	Quartus II raw programming data file that contains application image 2 only.
`Nios_application.pof`	• Programming file that consists Nios II processor software application `.hex` file only. • To be programmed into external QSPI flash.
`pfl.sof`	• Quartus II **.sof** containing PFL. • Programmed into QSPI flash on MAX 10 FPGA Development kit.

1.6 Reference Design Functional Description

Figure 1. Reference Design Block Diagram



1.6.1 Reference Design IP Core Components

1.6.1.1 Nios II Gen2 Processor

The Nios II Gen2 Processor in the reference design has the following functions:

- A bus master which handles all interface operations with the Altera On-Chip Flash IP core including read, write, and erase.
- Provides an algorithm in software to receive the programming bit stream from a host computer and trigger reconfiguration through the Dual Configuration IP core.

You need to set the reset vector of the processor accordingly. This is to ensure the processor boots the correct application code from either UFM or external QSPI flash.

Note: If the Nios II application code is large, Intel recommends that you store the application code in the external QSPI flash. In this reference design, the reset vector is pointing to the external QSPI flash where the Nios II application code is stored.

Related Links

Nios II Gen2 Hardware Development Tutorial
Provides more information about developing Nios II Gen2 Processor.

1.6.1.2 Altera On-Chip Flash IP Core

The Altera On-Chip Flash IP core functions as an interface for the Nios II processor to do a read, write or erase operation to the CFM and UFM. The Altera On-Chip Flash IP core provides allows you to access, erase and update the CFM with a new configuration bit stream. The Altera On-Chip Flash IP parameter editor shows a predetermined address range for each memory sector.

Related Links

Altera On-Chip Flash IP Core
Provides more information about Altera On-Chip Flash IP Core.

1.6.1.3 Altera Dual Configuration IP Core

You can use the Altera Dual Configuration IP core to access the remote system upgrade block in MAX 10 FPGA devices. The Altera Dual Configuration IP core allows you to trigger reconfiguration once the new image has been downloaded.

Related Links

Altera Dual Configuration IP Core
Provides more information about Altera Dual Configuration IP Core

1.6.1.4 Altera UART IP Core

The UART IP core allows the communication of serial character streams between an embedded system in MAX 10 FPGA and an external device. As an Avalon-MM master, the Nios II processor communicates with the UART IP core, which is an Avalon-MM slave. This communication is done by reading and writing control and data registers.

The core implements the RS-232 protocol timing and provides the following features:

- adjustable baud rate, parity, stop, and data bits
- optional RTS/CTS flow control signals

Related Links

UART Core

Provides more information about UART Core.

1.6.1.5 Generic Quad SPI Controller IP Core

The Generic Quad SPI Controller IP core functions as an interface between MAX 10 FPGA, the external flash and the on-board QSPI flash. The core provides access to the QSPI flash through read, write and erase operations.

When the Nios II application expands with more instructions, the file size of the hex file generated from Nios II application will be larger. Beyond a certain size limit, the UFM will not have a sufficient space to store the application hex file. To solve this, you can use the external QSPI flash available on the MAX 10 FPGA Development kit to store the application hex file.

1.6.2 The Nios II EDS Software Application Design

The reference design includes Nios II software application code that controls the remote upgrade system design. The Nios II software application code responses to the host terminal through UART by executing specific instructions.

1.6.2.1 Updating Application Images Remotely

After you have transmitted a programming bit stream file using the Remote Terminal, the Nios II software application is designed do the following:

1. Set the Altera On-Chip Flash IP core Control Register to un-protect the CFM1 & 2 sector.
2. Perform sector erase operation on CFM1 and CFM2. The software polls the status register of the Altera On-Chip Flash IP core to ensure successful erase has been completed.
3. Receive 4 bytes of bit stream at a time from `stdin`. Standard input and output can be used to receive data directly from the host terminal and print output onto it. Types of standard input and output option can be set through the **BSP Editor** in Nios II Eclipse Build tool.
4. Reverses the bit order for each byte.

   Note: Due to the configuration of Altera On-Chip Flash IP Core, every byte of data needs to be reversed before writing it into CFM.
5. Start to write 4 bytes of data at a one time into CFM1 and CFM2. This process continues until the end of programming bit stream.
6. Polls the status register of Altera On-Chip Flash IP to ensure successful write operation. Prompts a message to indicate the transmission is complete.

   Note: If the write operation fails, the terminal will halt the bit stream sending process and generate an error message.
7. Sets the Control Register to re-protect CFM1 and CFM2 to prevent any unwanted write operation.

Related Links

.pof Generation through Convert Programming Files on page 11
Provides information about creating rpd files during convert programming files.

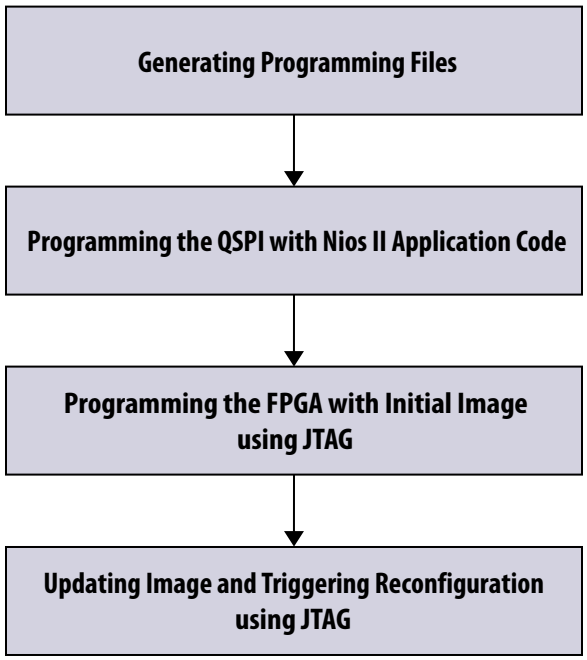
1.6.2.2 Triggering Reconfiguration Remotely

After you select trigger reconfiguration operation in the host Remote Terminal, the Nios II software application will do the following:

1. Receive the command from standard input.
2. Start the reconfiguration with the following two write operations:
   - Write `0x03` to the offset address of `0x01` in the Dual Configuration IP core. This operation overwrites the physical `CONFIG_SEL` pin and sets Image 1 as the next boot configuration image.
   - Write `0x01` to the offset address of `0x00` in the Dual Configuration IP core. This operation triggers reconfiguration to application image in CFM1 and CFM2

1.7 Reference Design Walkthrough

Figure 2. Remote System Upgrade over UART with the Nios II Processor for MAX 10 FPGA Devices Flow



1.7.1 Generating Programming Files

You have to generate the following programming files before being able to use the remote system upgrade on the MAX 10 FPGA Development kit:

- For QSPI Programming:
  - `.sof`—use the `pfl.sof` included in the reference design or you can choose to create a different `.sof` containing your own PFL design
  - `.pof`—configuration file generated from a `.hex` and programmed into the QSPI flash.
- For remote System Upgrade:
  - `.pof`—configuration file generated from a `.sof` and programmed into the internal flash.
  - `.rpd`—contains the data for internal flash which includes ICB settings, CFM0, CFM1 and UFM.
  - `.map`—holds the address for each memory sector of ICB settings, CFM0, CFM1 and UFM.

1.7.1.1 Generating files for QSPI Programming

To generate the `.pof` file for QSPI programming, perform the following steps:

1. Build Nios II Project and generate HEX file.

   Note: Refer to AN730: Nios II Processor Booting Methods In MAX 10 Devices for information about building Nios II project and generating HEX file.
2. On the **File** menu, click **Convert Programming Files**.
3. Under **Output programming file**, select **Programmer Object File (.pof)** in the **Programming file type** list.
4. In the **Mode** list, select **1-bit Passive Serial**.
5. In the **Configuration device** list, select **CFI_512Mb**.
6. In the **File name** box, specify the file name for the programming file you want to create.
7. In the **Input files to convert** list, remove the **Options and SOF data** row. Click **Add Hex Data** and a **Add Hex Data** dialog box appear. In the **Add Hex Data** box, select **Absolute addressing** and insert the `.hex` file generated from Nios II EDS Build Tools.
8. After all settings are set, click **Generate** to generate related programming file.

Related Links

AN730: Nios II Processor Booting Methods In MAX 10 FPGA Devices

1.7.1.2 Generating files for Remote System Upgrade

To generate the `.pof`, `.map` and `.rpd` files for remote system upgrade, perform the following steps:

1. Restore the `Factory_image`, `application_image_1` and `application_image_2`, and compile all three designs.
2. Generate two `.pof` files described in the following table:

   Note: Refer *.pof Generation through Convert Programming Files* for steps on generating `.pof` files.

Generated .pof[2]	.sof to be Included
app1.pof	• Factory_Image.sof • Application_Image_1.sof
app2.pof	• Factory_Image.sof • Application_Image_2.sof *Note:* You have to generate the .rpd and .map files when generating the .pof.

3. Open the app2.rpd using any hex editor.
4. In the hex editor, select the binary data block based on the start and end offset by referring to the .map file. The start and end offset for the 10M50 device is 0x12000 and 0xB9FFF respectively. Copy this block to a new file and save it in a different .rpd file. This new .rpd file contains application image 2 only.

Figure 3. Example of .map file

The start and end offsets shown are applicable for 10M50 devices.

```
BLOCK        START ADDRESS        END ADDRESS

ICB          0x00000000           0x00001FFF
UFM          0x00002000           0x00011FFF
CFM0         0x000BA000           0x00161FFF (0x00108F8B)
CFM1         0x00012000           0x000B9FFF (0x000615DF)


Max 10 Setting:

    EPOF: OFF
    Verify protect: OFF
    Watchdog value: Not activated
    Auto-reconfigure when initial image fails: ON
    POR: Instant ON
    IO Pullup: ON
    SPI IO Pullup: ON



Notes:

- Data checksum for this conversion is 0x0E103627

- All the addresses in this file are byte addresses
```

2 File name for the generated .pof is an example. You are free to rename the files differently.

1.7.1.2.1 .pof Generation through Convert Programming Files

To convert `.sof` files to `.pof` files, follow these steps:

1. On the **File** menu, click **Convert Programming Files**.
2. Under **Output programming file**, select **Programmer Object File** (`.pof`) in the **Programming file type** list.
3. In the **Mode** list, select **Internal Configuration**.
4. In the **File name** box, specify the file name for the programming file you want to create.
5. To generate a Memory Map File (`.map`), turn on **Create Memory Map File** (Auto generate output_file.map). The **.map** contains the address of the CFM and UFM with the ICB setting that you set through the **Option/Boot Info** option.
6. To generate a Raw Programming Data (`.rpd`), turn on **Create config data RPD** (Generate output_file_auto.rpd).

   With the help of Memory Map File, you can easily identify the data for each functional block in the `.rpd` file. You can also extract the flash data for third party programming tools or update the configuration or user data through the Altera On-Chip Flash IP.
7. The `.sof` can be added through **Input files to convert** list and you can add up to two `.sof` files.

   For remote system upgrade purposes, you can retain the original page 0 data in the `.pof`, and replace page 1 data with new `.sof` file. To perform this, you need to add the `.pof` file in page 0, then add `.sof` page, then add the new `.sof` file to page 1.
8. After all settings are set, click **Generate** to generate related programming file.

1.7.2 Programming the QSPI

To program the Nios II application code into the QSPI flash, perform the following steps:

1. On the MAX 10 FPGA Development Kit, switch the MAX10_BYPASSn to 0 to bypass on-board VTAP (MAX II) device.
2. Connect the Intel FPGA Download Cable (formerly USB Blaster) to the JTAG header.
3. In the **Programmer** window, click **Hardware Setup** and select **USB Blaster**.
4. In the **Mode** list, select **JTAG**.
5. Click **Auto Detect** button on the left pane.
6. Select the device to be programmed, and click **Add File**.
7. Select the `pfl.sof`.
8. Click **Start** to start programming.
9. After programming is successful, without turning-off the board, click **Auto Detect** button on the left pane again. You will see a **QSPI_512Mb** flash appear in the programmer window.

10. Select the QSPI device, and click **Add File**.
11. Select the `.pof` file generated previously from `.hex` file.
12. Click **Start** to start programming the QSPI flash.

1.7.3 Programming the FPGA with Initial Image using JTAG

You have to program the `app1.pof` into the FPGA as the device initial image. To program the `app1.pof` into the FPGA, perform the following steps:

1. In the **Programmer** window, click **Hardware Setup** and select **USB Blaster**.
2. In the **Mode** list, select **JTAG**.
3. Click **Auto Detect** button on the left pane.
4. Select the device to be programmed, and click **Add File**.
5. Select the `app1.pof`.
6. Click **Start** to start programming.

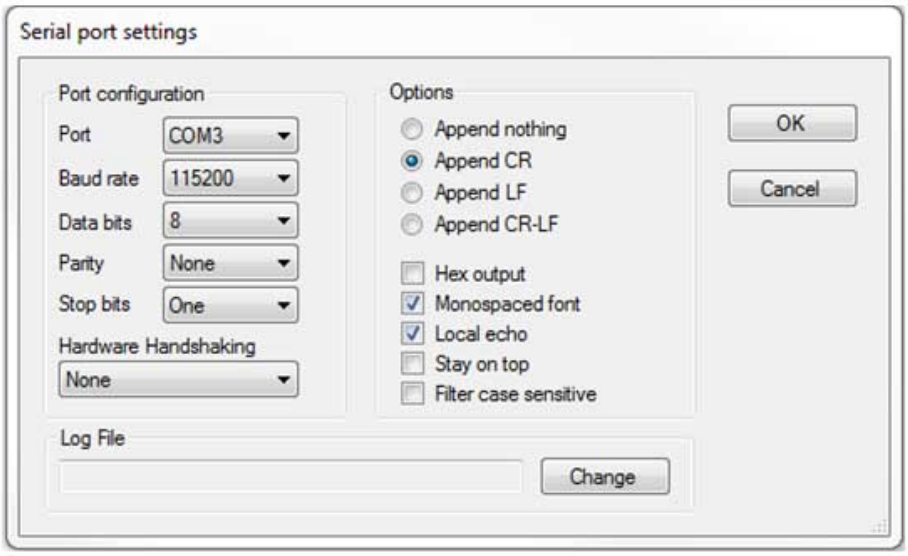
1.7.4 Updating Image and Triggering Reconfiguration using UART

To remotely configure your MAX10 FPGA development kit, perform the following steps:

1. *Notes:* Before you start, ensure the following:
   - the CONFIG_SEL pin on the board is set to 0
   - your board's UART port is connected to your computer

   Open `Remote Terminal.exe` and the **Remote Terminal** interface opens.
2. Click **Settings** and **Serial port settings** window will appear.
3. Set the parameters of remote terminal to match the UART settings selected in Quartus II UART IP core. After setting is complete, click **OK**.

Figure 4. UART Serial Port Settings Screen Capture



4. Press the nCONFIG button on the development kit or key-in 1 in the **Send** text box, and then hit **Enter**.

   A list of operation choice will appear on the terminal, as shown below:

```
Hello from Nios II!
Enter 1,2,3 or 4 to select the operation:
1: Write Image to CFM0
2: Write Image to CFM1 and CFM2
3: Trigger reconfiguration to CFM0 (Factory Image)
4: Trigger reconfiguration to CFM1 and CFM2 (App Image)
```

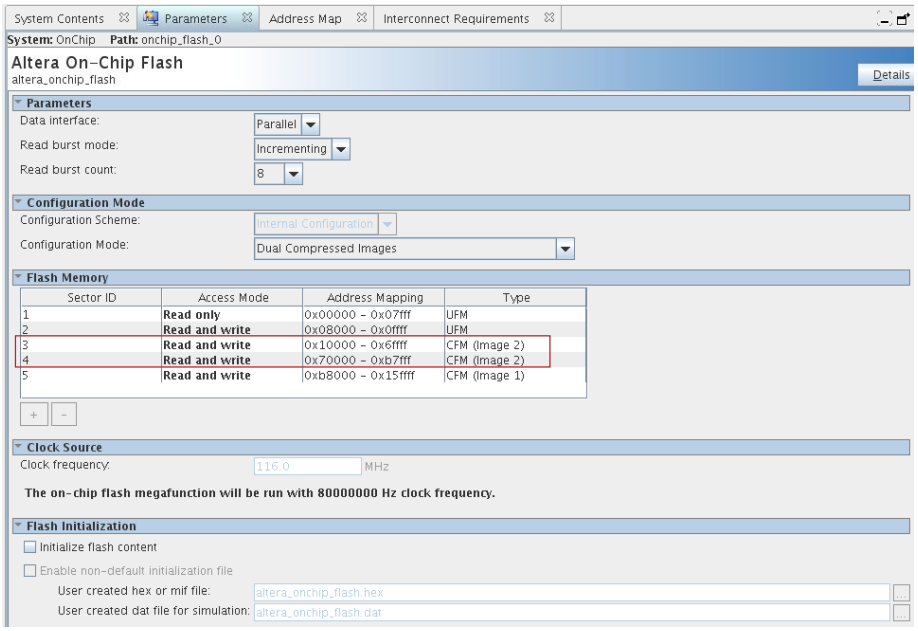
   Note: To select an operation, key in the number in the **Send** text box, and then hit **Enter**.

5. To update application image 1 with application image 2, select operation 2. You will be prompted to insert start and end address of CFM1 and CFM2.

   Note: The address shown in the map file includes ICB settings, CFM and UFM but the Altera On-Chip Flash IP can access CFM and UFM only. Hence, there is an address offset between the address shown in map file and Altera On-Chip Flash IP parameter window.

6. Key in the address based on the address specified by the Altera On-Chip Flash IP parameter window.

Figure 5. Start and End Address in Altera On-Chip Flash IP Core




```
Please key in start address:
10000
Please key in end address:
b7fff
```

Erase will automatically start after you enter the end address.

```
CFM2 Erased
CFM1 Erased
Enter Programming file.
```

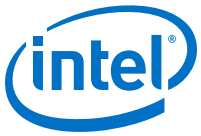
7. After erase successful, you will be prompted to enter programming `.rpd` file for application image 2. To upload image, click **SendFile** button, and then select the `.rpd` containing application image 2 only and click **Open**.

   Note: Other than application image 2, you can use any new image that you wish to update into the device.

   The update process will start directly and you can monitor the progress through the terminal. The operation menu will prompt Done and you can now choose the next operation.

8. To trigger reconfiguration, select operation 4. You can observe the LED behavior indicating the different image loaded into the device.

Image	LED Status (Active Low)
Factory Image	01010
Application Image 1	10101
Application Image 2	01110



1.8 Document Revision History

Date	Version	Changes
February 2017	2017.02.21	Rebranded as Intel.
June 2015	2015.06.15	Initial release.