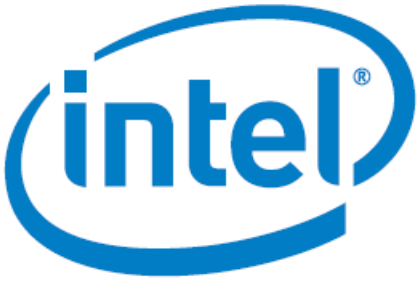

# Nios II Processor Booting from CFI Flash

# (Cyclone V GX FPGA Development Kit)

Date: 31 July 2017

Revision: 1.0

# Table of Contents

# 1. Introduction

This design example demonstrates step-by-step instructions on how to boot a simple Hello World Nios II application from CFI flash using Cyclone V GX FPGA Development Kit. The boot option used in this design example is "boot copier" method, however, you may change the BSP editor settings to enable "execute-in-place" method.

# 2. Prerequisite

You are required to have knowledge in instantiating and developing a system with a Nios II processor. Intel recommends that you go through the online tutorials and training materials provided in Intel FPGA Training before using this design example.

Related information:

- Generic Nios II Booting Methods User Guide
- Embedded Design Handbook
- Avalon Tri-State Conduit Components User Guide
- Parallel Flash Loader IP Core User Guide

# 3. Requirements

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

- Cyclone V GX FPGA Development Kit & Kit installation
- Quartus Prime version 16.1
- USB cable for USB-Blaster II JTAG

# 4. Design Example Files

The master files are provided with this design example. You may proceed to reading Programming the CFI flash section and Run the Design section to program the CFI flash and run the design without any recompilation or programing file generation.

| File Name | Description |
|---|---|
| master_image\output_file.pof | Quartus Prime programming file for CFI flash |
| master_image\ext_flash.hex<br>master_image\cvgx_cfi_booting.sof | Hardware image file (.sof) and Nios II application image file (.hex) that are used to generate CFI flash programming file |
| software\cvgx_cfi_hello_world.zip<br>software\cvgx_cfi_hello_world_bsp.zip | Hello world software code and BSP for the Nios II processor |

# 5. Block Diagram

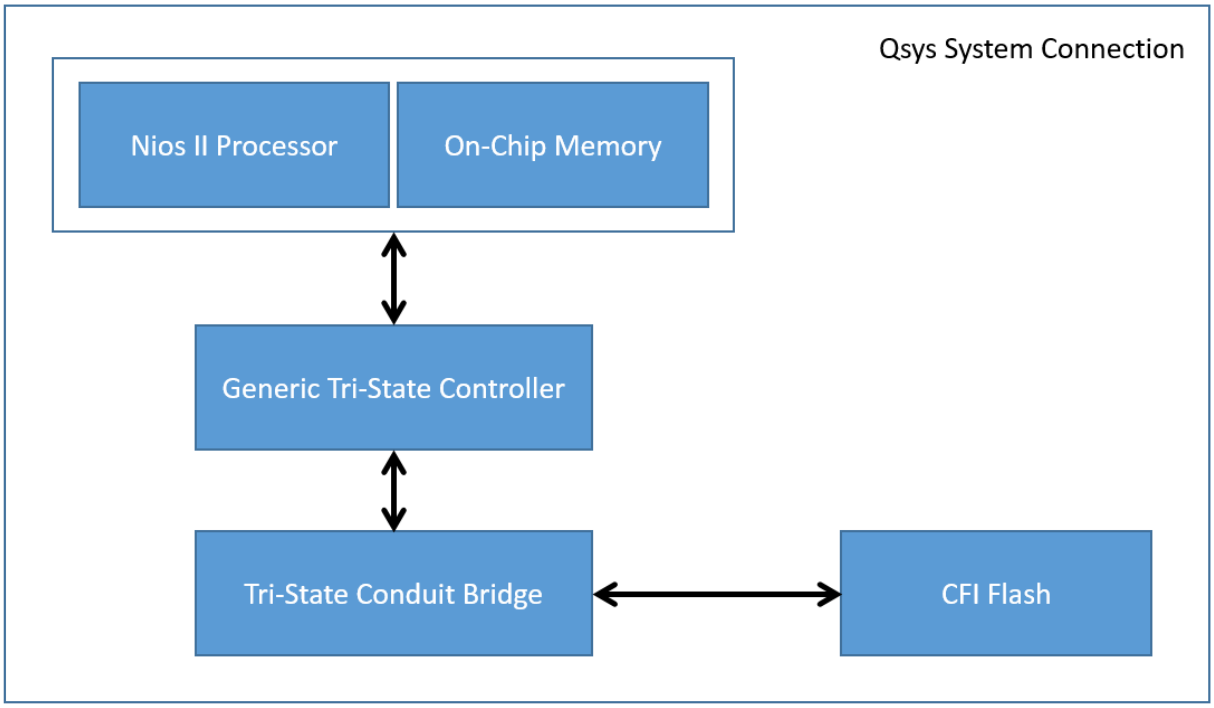


*Figure 1: Qsys System Connection*

Figure 1 shows the main IP components required for building a Nios II system booting from CFI flash. Details of the connection between each IP blocks can be found in the included Qsys file for this design example.

# 6. Steps to Build a Bootable System

The following sections describe the steps to build a bootable system for a:

- Nios II processor application executing in place from CFI flash
- Nios II processor application copied from CFI flash to RAM using a boot copier

## 6.1 Quartus Setting

The MSEL pins in Cyclone V GX development kit are defaulted to FPPx16 configuration scheme. In Quartus Prime software, go to **Assignments → Device → Device and Pin Options → Configuration**, ensure the *Configuration scheme* is set to **Passive Parallel x16**.

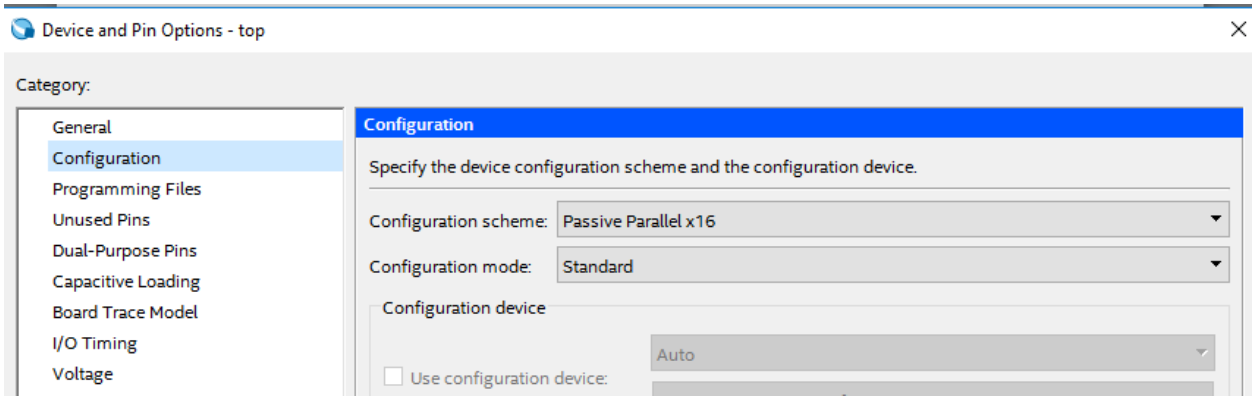


*Figure 2: Configuration Scheme Setting*

## 6.2 Design

The Qsys file in this design example includes all main components that are required for a Nios II system booting from CFI flash, essentially the *Generic Tri-State Controller* and the *Tri-State Conduit Bridge* that connects the CFI flash to the Nios II processor.

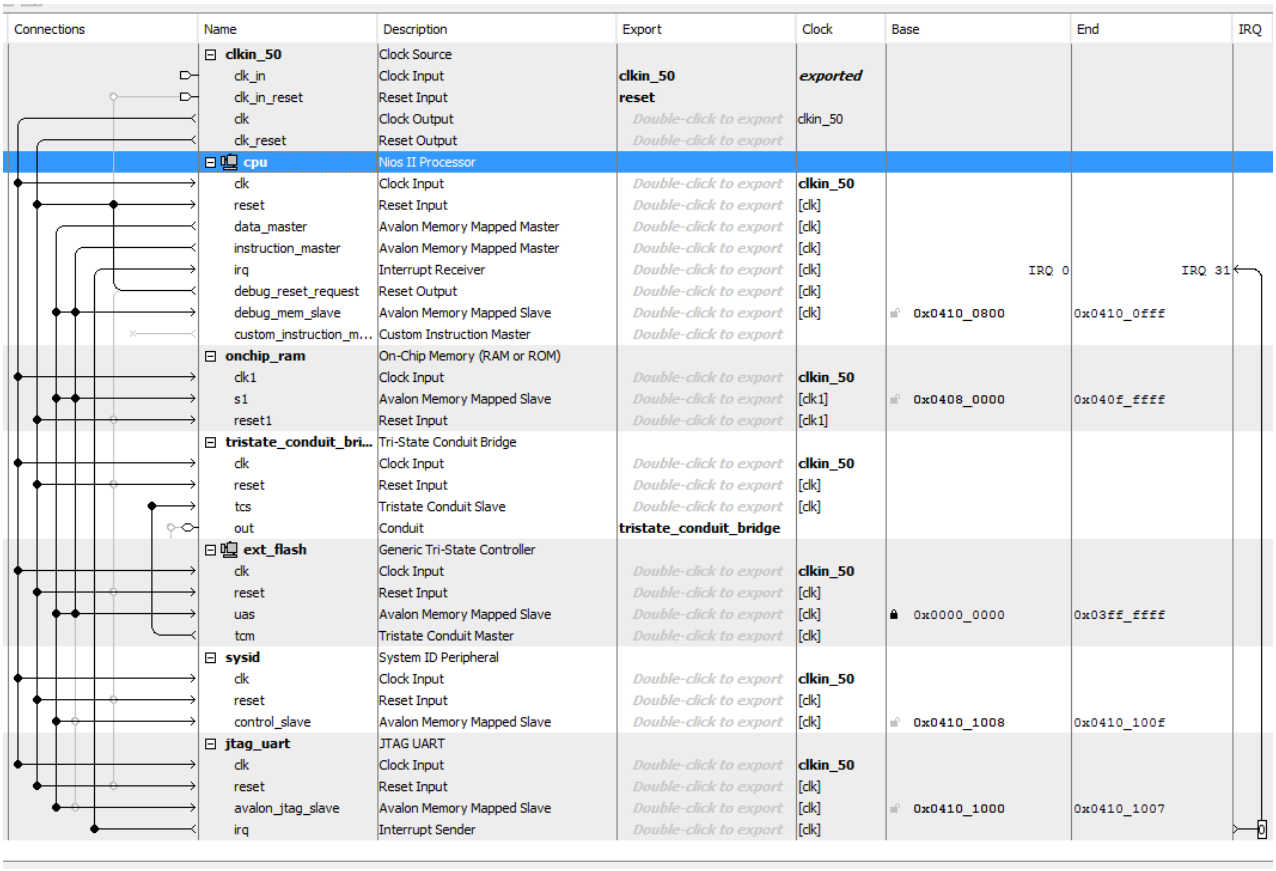


*Figure 3: Nios II Qsys System Connections*

### 6.2.1 Nios II Processor Qsys Setting

1. In the Nios II Processor parameter editor, set the *Reset vector memory* to CFI flash (ext_flash.uas) and *Exception vector memory* to On-Chip Memory (onchip_ram.s1).
2. Set the *Reset vector offset* to 0x01bc0000. The reset vector must be the base address of your application. In this example, it is 0x01bc0000.

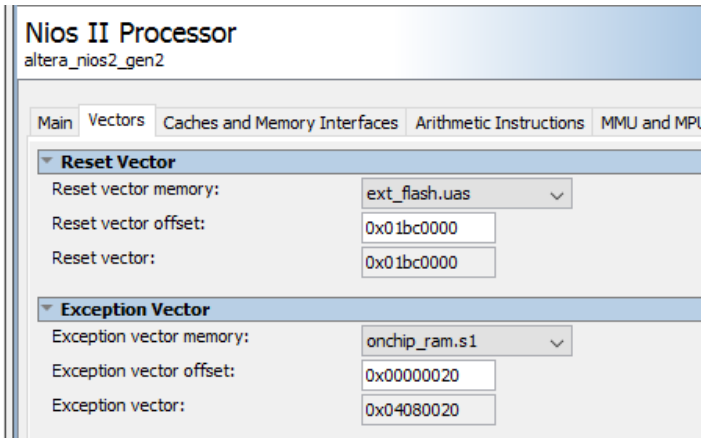


*Figure 4: Nios II Processor Vectors Configuration*

*Note: Reset vector memory determines where the reset code (boot loader) resides, and Reset vector offset determines the location of the reset vector (reset address).*

### 6.2.2 Generic Tri-State Controller Qsys Setting

The Generic Tri-State Controller provides preset configurations for many commonly used external devices. If you select one of the preset configurations, all the parameters are automatically assigned to the correct values. In case your CFI flash is not in the preset library, you can use the parameter editor to specify the required settings accordingly.

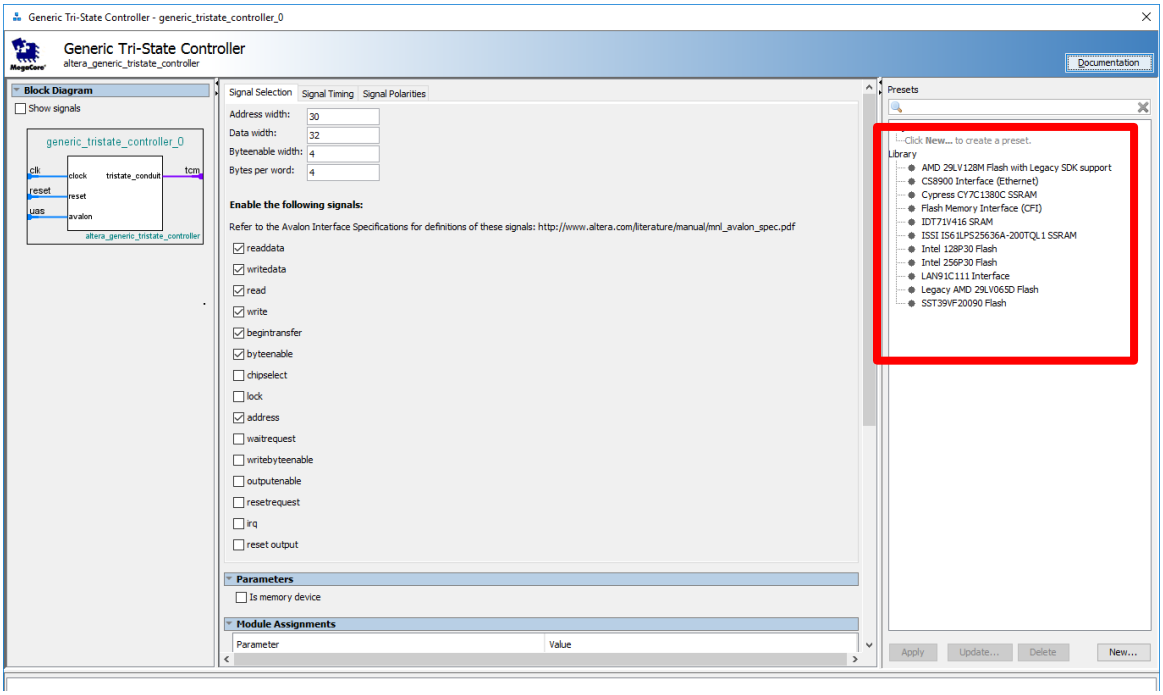


*Figure 5: Generic Tri-State Controller Configuration Window*

The following snapshots show the configuration setting for the CFI flash used in this development kit.

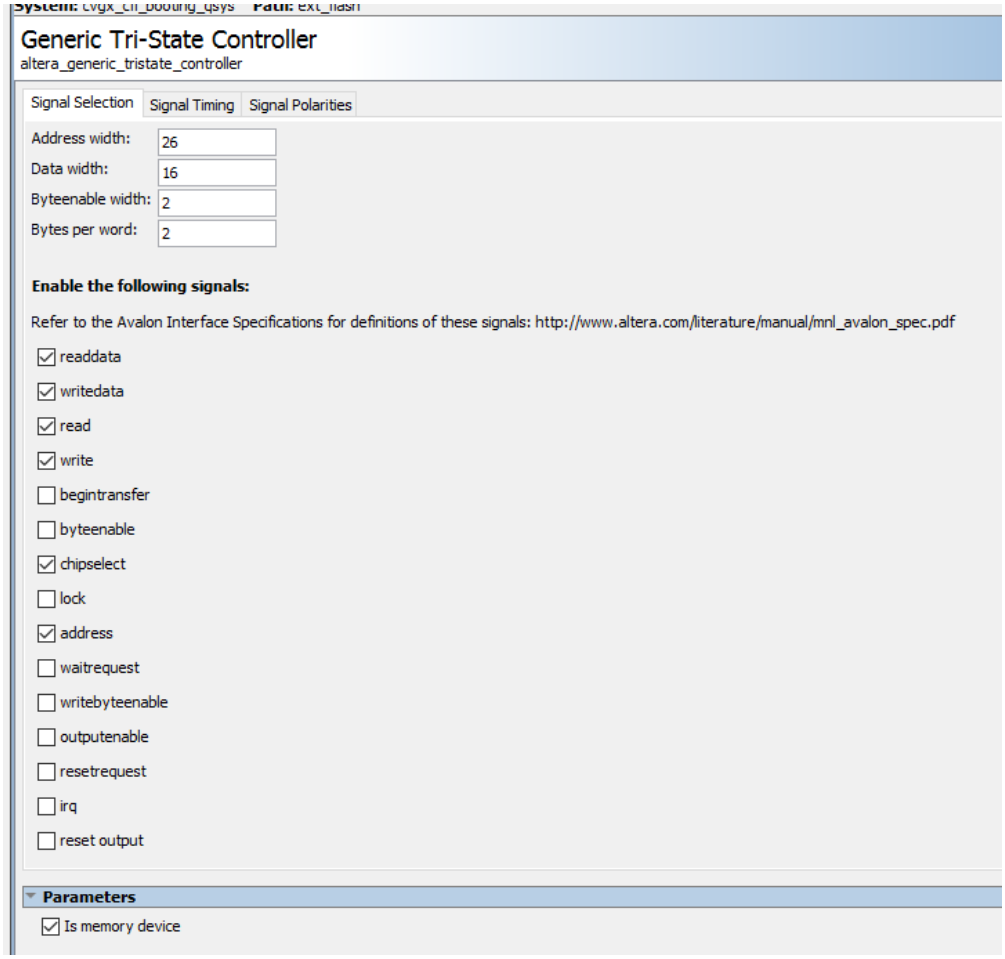


*Figure 6: Signal Selection Part 1*

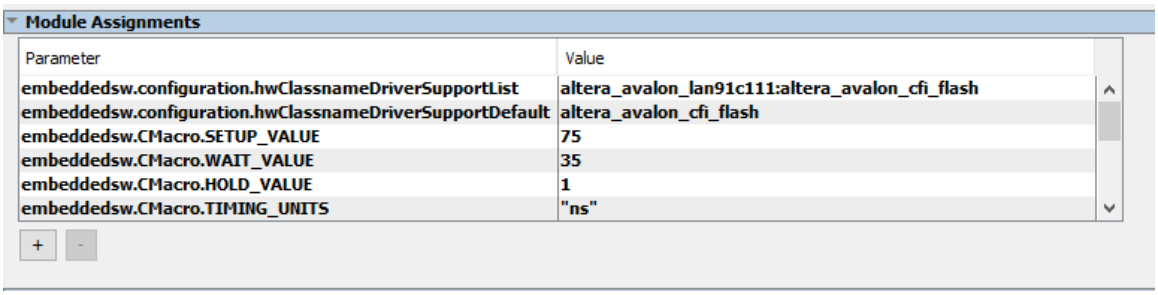


*Figure 7: Signal Selection Part 2*

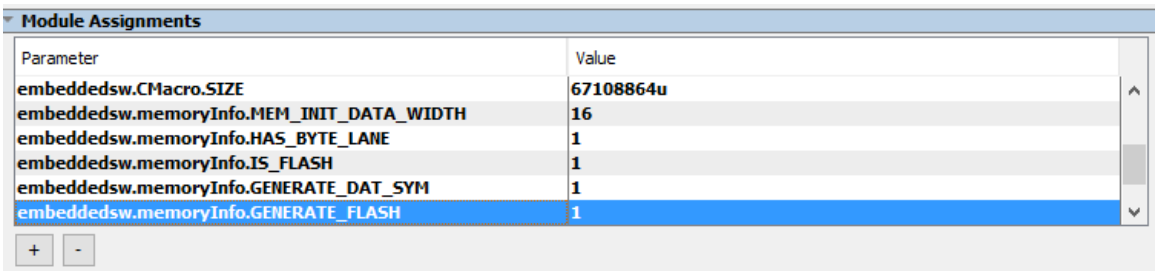


*Figure 8: Signal Selection Part 3*

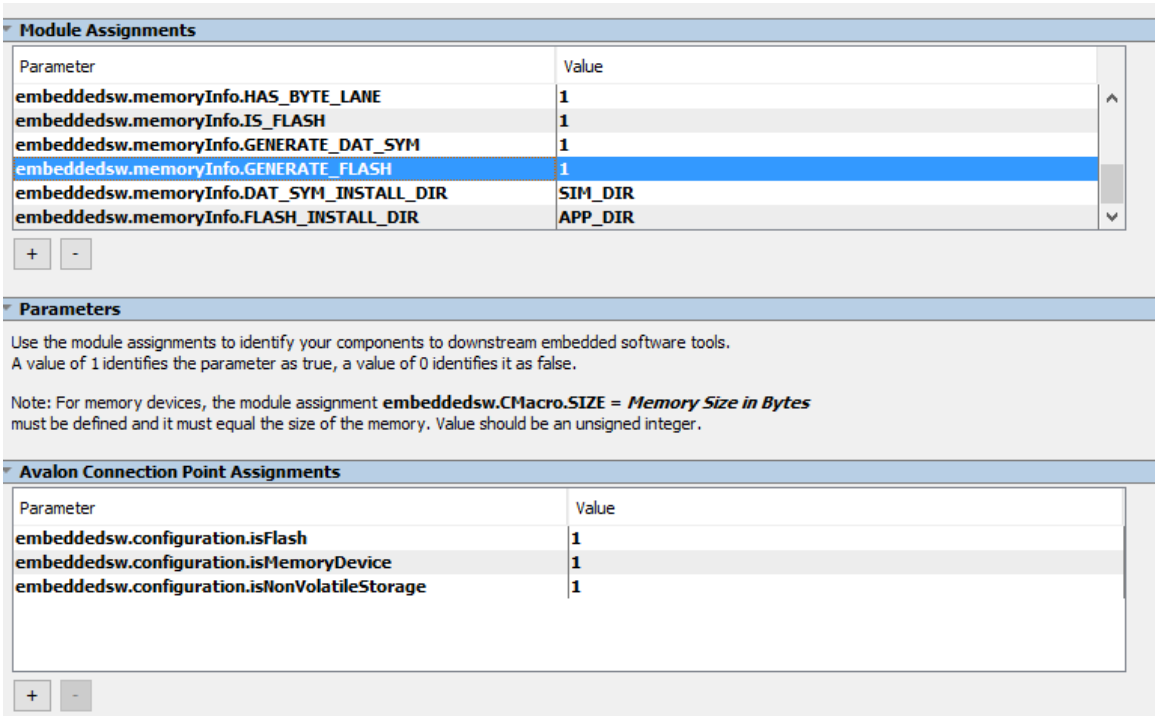


*Figure 9: Signal Selection Part 4*

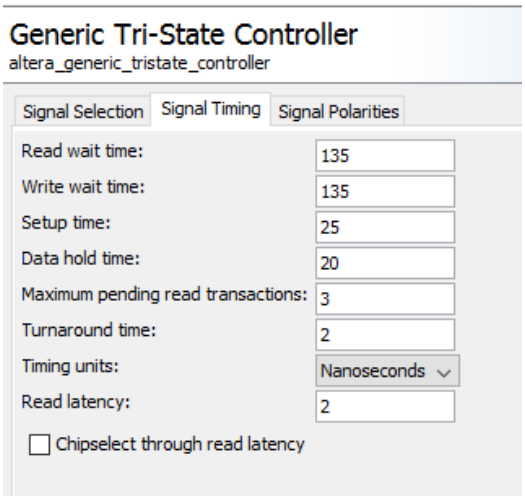


*Figure 10: Signal Timing*

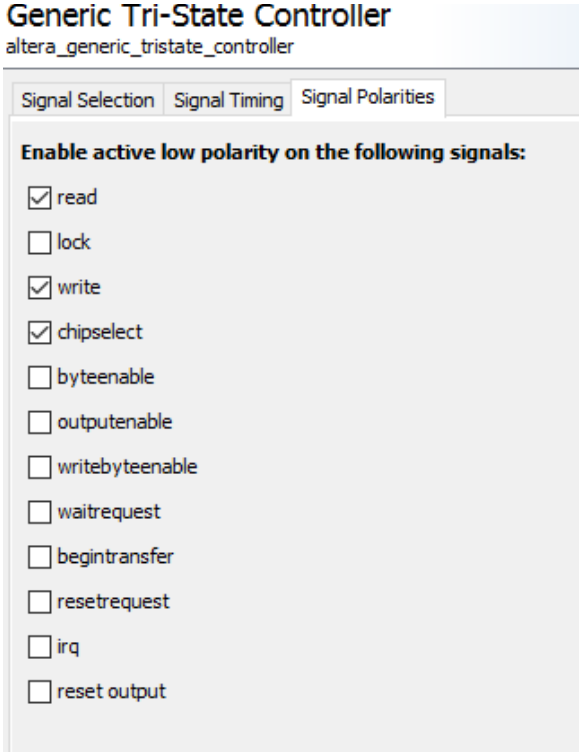


*Figure 11: Signal Polarities*

## 6.3 BSP Editor Settings

There are two boot options for Nios II processor booting from CFI flash - Nios II application execute-in-place from CFI flash and Nios II application copied from CFI flash to RAM using boot copier. Refer to table below for BSP editor settings in Nios II SBT for each boot option:

| Boot Option | BSP Editor Setting: Linker Script | BSP Editor Setting: Settings.Advanced.hal.linker |
|---|---|---|
| **Nios II processor application execute-in-place from CFI flash** | • Set **.text** Linker Section to CFI flash<br>• Set other Linker Sections (**.heap**, **.rwdata**, **.rodata**, **.bss**, **.stack**) to OCRAM/ External RAM | If the exception vector memory is set to OCRAM/ External RAM, enable the following settings:<br>• allow_code_at_reset<br>• enable_alt_load<br>• enable_alt_load_copy_rodata<br>• enable_alt_load_copy_rwdata<br>• enable_alt_load_copy_exceptions<br><br>If the exception vector memory is set to CFI flash, enable the following settings:<br>• allow_code_at_reset<br>• enable_alt_load<br>• enable_alt_load_copy_rodata<br>• enable_alt_load_copy_rwdata |
| **Nios II processor application copied from CFI flash to RAM using boot copier** | All Linker Sections are set to OCRAM/ External RAM | All settings are left unchecked |

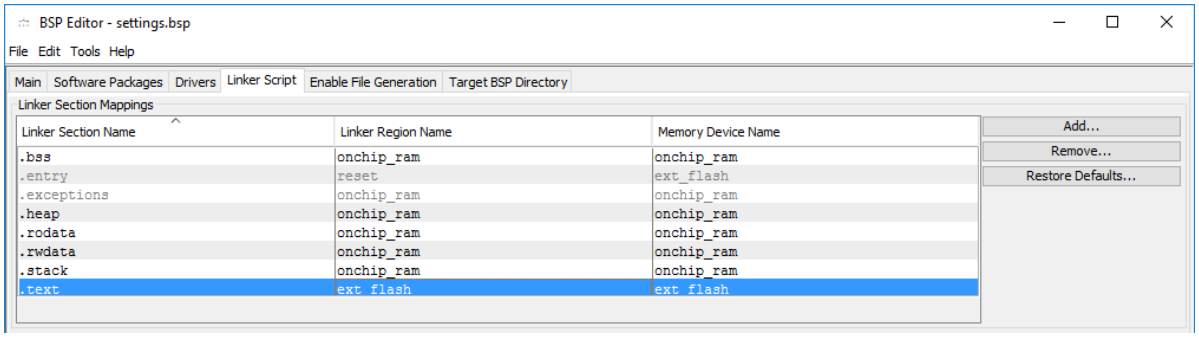


*Figure 12: Linker Script for Execute-In-Place Method*

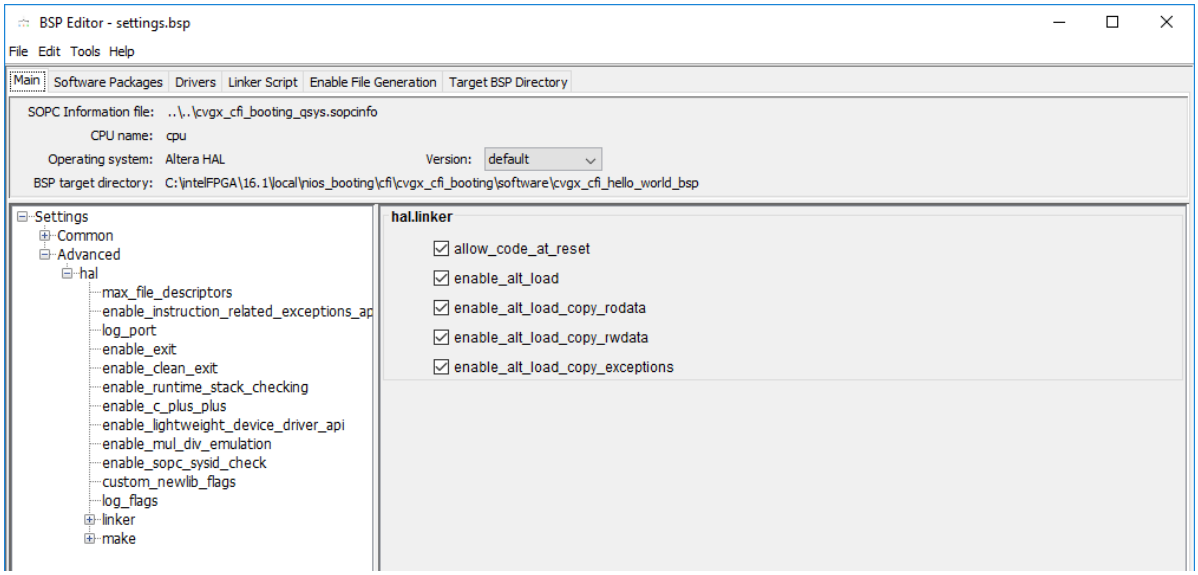


*Figure 13: Settings.Advanced.hal.linker for Execute-In-Place Method if Exception vector is set to OCRAM/ External RAM*

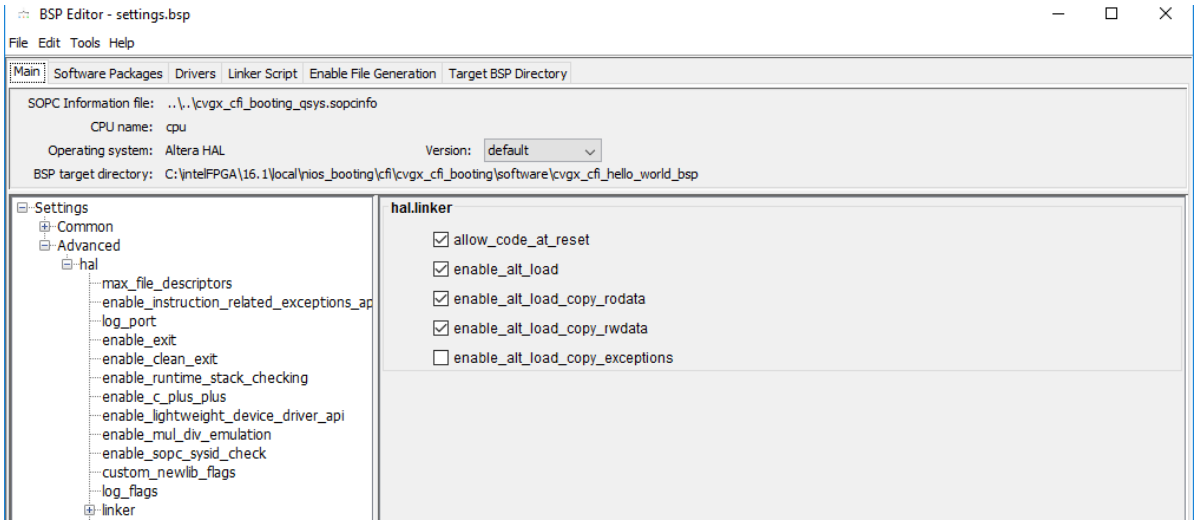


*Figure 14: Settings.Advanced.hal.linker for Execute-In-Place Method if Exception vector is set to CFI flash*

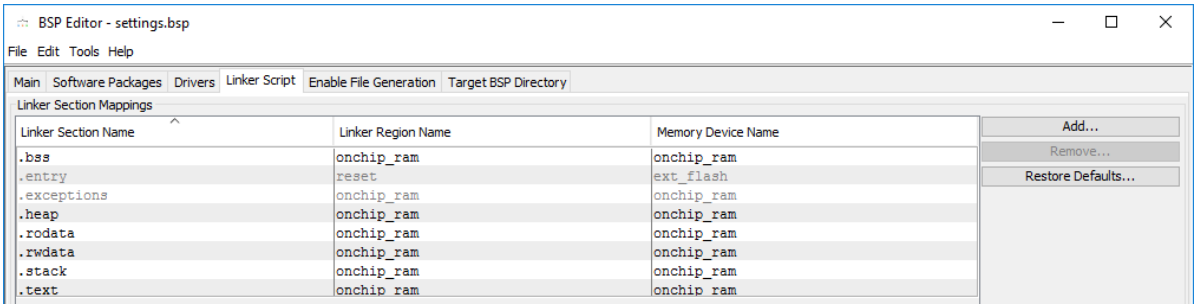


*Figure 15: Linker Script for Boot Copier Method*

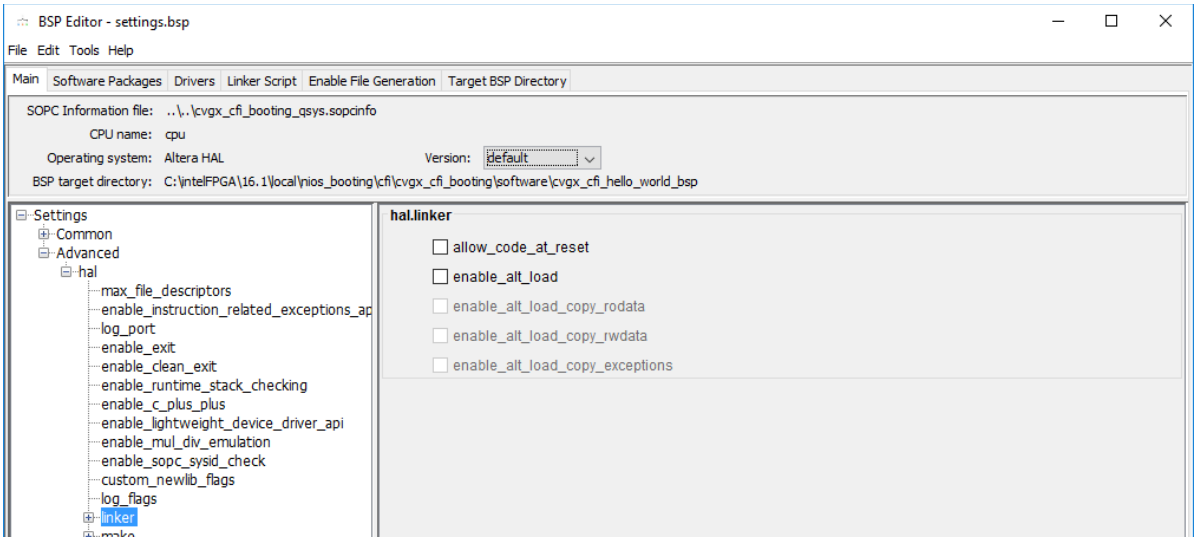


*Figure 16: Settings.Advanced.hal.linker for Boot Copier Method*

## 6.4 Application

This design example includes the Hello World software code and the BSP. You may also create your own Nios II application using the Nios II SBT.

To build the Nios II application image file:

1. After creating the Nios II application, right click on the project in **Project Explorer** and select **Build Project**. A ***.elf** file is created under the project folder.
2. Right click on the project in Project Explorer and select **Make Targets → Build…**. Select **mem_init_generate** and click **Build**. A ***.flash** file is created under the project folder.

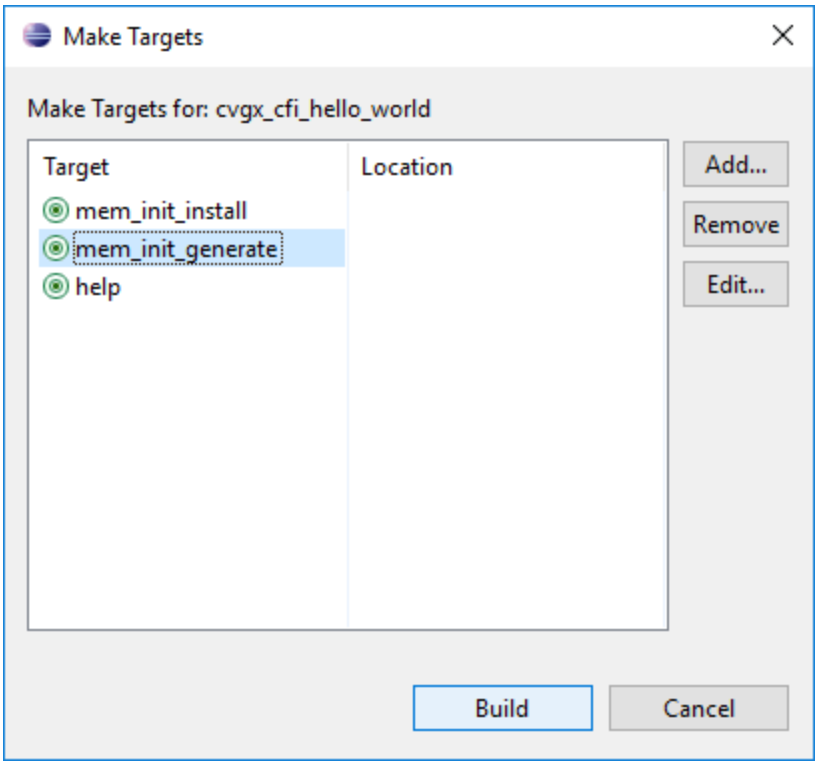


*Figure 17: Make Targets Window*

3. Launch **Nios II Command Shell**. Use *nios2-elf-objcopy* utility to convert **.flash** file to **HEX** file.

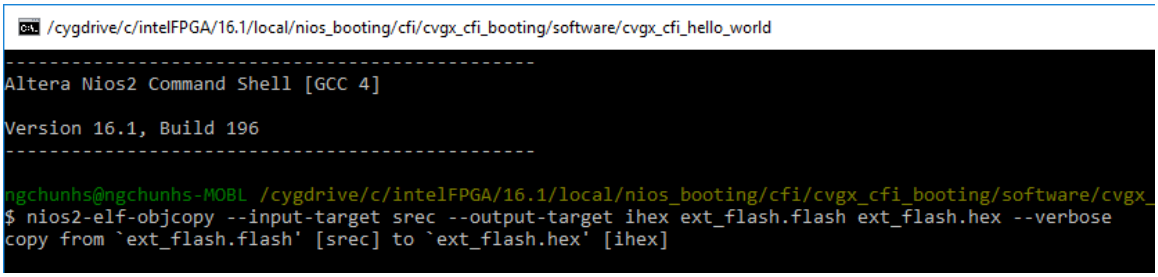


*Figure 18: Nios II Command Shell Window*

## 6.5 Generate programming files

Fast Passive Parallel (FPP) is the configuration scheme used in this design example. In Cyclone V GX development kit, the on board CPLD (MAX V) is used to control the FPGA configuration process. You may refer to the factory MAX V design in *<kit installation directory>/ cycloneVGX_5cgxfc7df31es_fpga/ examples/ max5*, after completing the kit installation on your PC. This design example uses the factory MAX V design from development kit, thus the CFI flash

memory map must not be altered. You need to just focus on the factory software, factory hardware and PFL option bits.

| Block Description | KB Size | Address Range |
|---|---|---|
| Unused | 128 | 0x03FE.0000 - 03FF.FFFF |
| User software | 28,800 | 0x023C.0000 - 03FD.FFFF |
| Factory software | 8192 | 0x01BC.0000 - 023B.FFFF |
| zipfs (html, web content) | 4096 | 0x017C.0000 - 01BB.FFFF |
| User hardware 2 | 8064 | 0x00FE.0000 - 017B.FFFF |
| User hardware 1 | 8064 | 0x0080.0000 - 00FD.FFFF |
| Factory hardware | 8064 | 0x0002.0000 - 007F.FFFF |
| PFL option bits | 32 | 0x0001.8000 - 0001.FFFF |
| Board information | 32 | 0x0001.0000 - 0001.7FFF |
| Ethernet option bits | 32 | 0x0000.8000 - 000.FFFF |
| User design reset vector | 32 | 0x0000.0000 - 000.7FFF |

*Figure 19: Default CFI Flash Memory Map*

To generate the programming file:

1. Open the **Convert Programming Files…** from **File** menu in Quartus Prime software.
2. Under **Output programming file** section, set the following items:
    a. **Programming file type:** Programmer Object File (.pof)
    b. **Configuration device:** CFI_512Mb
    c. **Mode:** Passive Parallel x 16
    d. **File name:** You may select your preferred path and file name for the output file (.pof). By default, this is generated under your project root directory.
3. Under **Input files to convert** section, follow these steps:
    a. For hardware image file:
        i. Select **Page_0** of **SOF Data** and click **Properties**.
        ii. Set the **Address mode for selected pages** to **Start**.
        iii. Set the **Start address** to **0x020000** (this is to replace Factory hardware) and click **OK**.
        iv. Click **Add File…** and open **master_image\cvgx_cfi_booting.sof**.
        v. Select the added SOF file and click **Properties**.
        vi. Check the **Compression** checkbox and click **OK**.
    b. For Nios II application image file:
        i. Click **Add Hex Data**.
        ii. Select **Absolute addressing**.
        iii. Open the HEX file **master_image\ext_flash.hex** and click **OK**.

4. Click **Options/Boot info…**, set the **Option bit address** to **0x018000** and click **OK**.
5. Click **Generate** to generate the POF programming file.

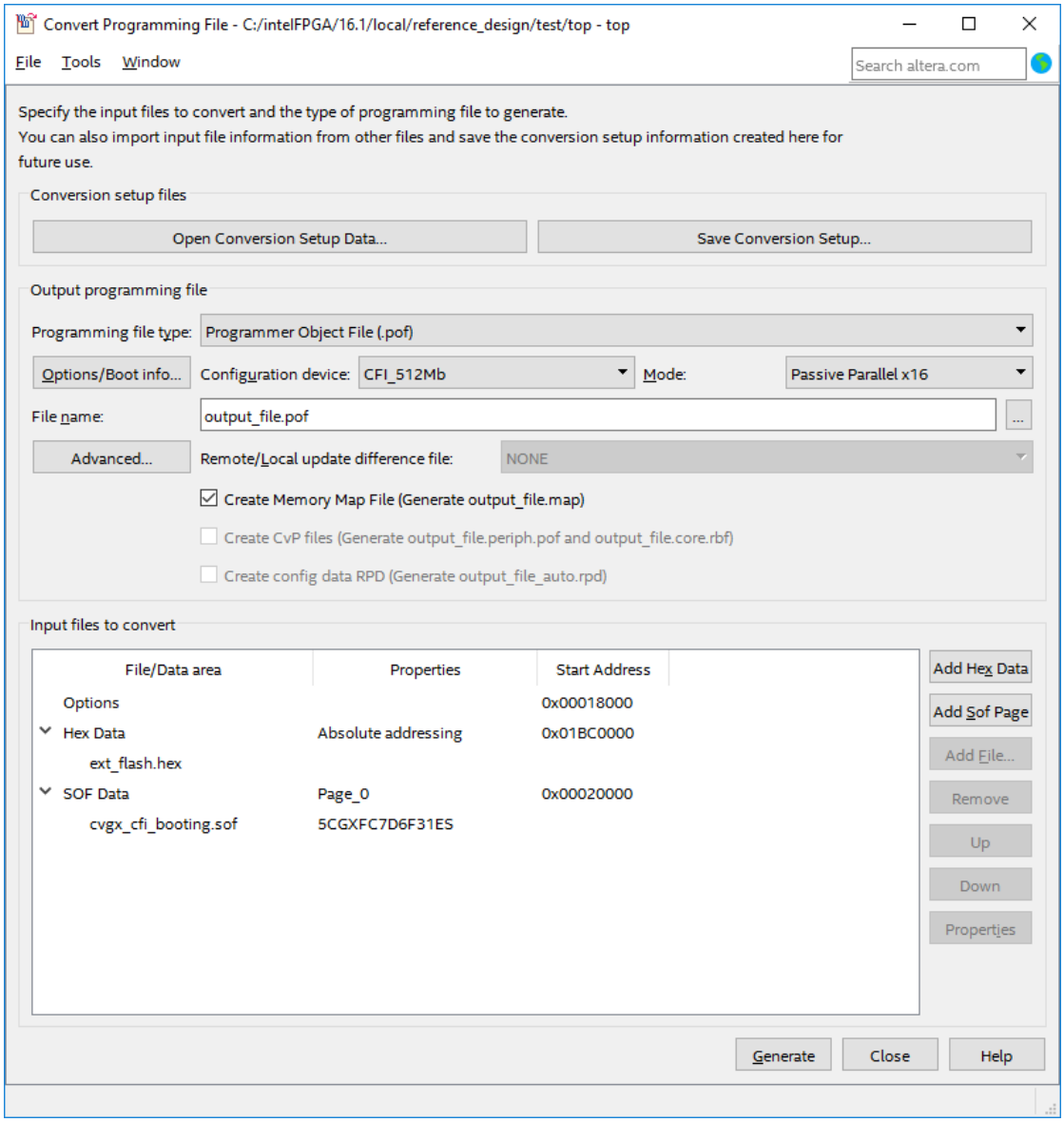


*Figure 20: Convert Programming File Window*

## 6.6 Programming the CFI flash

Intel provides Parallel Flash Loader (PFL) IP core that can program and control the FPGA configuration with data from external flash. A simple MAX V design that instantiates the PFL IP core to allow CFI flash programming is available here.

To program the CFI flash:

1. Connect the USB Blaster cable from PC to the on-board USB Blaster II connector and power up the board.
2. Open the **Programmer** from **Tools** menu in Quartus Prime software, then click **Hardware Setup…** and select **USB-BlasterII**.
3. Click **Auto Detect**, then select **5CGXFC7D6ES** and click **OK**.
4. Right click on the MAX V device **5M2210Z** and click **Change File**.
5. Open the simple MAX V PFL design output file named **maxV_pfl.pof** from here.
6. Tick the **Program/Configure** checkbox for MAX V **5M2210ZF256** device, including **CFM** and **UFM**.
7. Click **Start** to start programming the MAX V device.

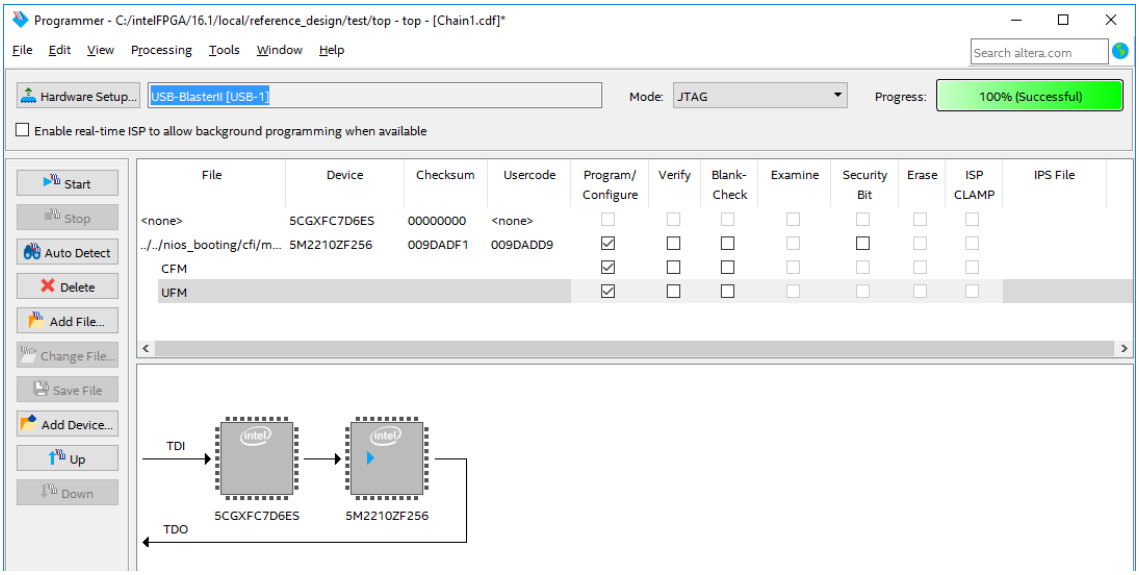


*Figure 21: Programming MAX V Window*

8. After the programming successful, click **Auto Detect**. If a warning dialog box pops up as below, click **Yes**.

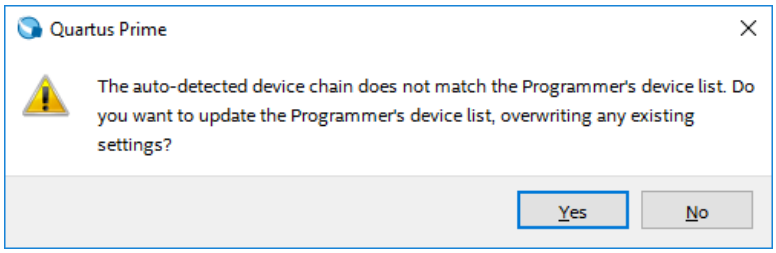


*Figure 22: Warning Pop Up Box*

9. You should see **CFI_512Mb** attached to the MAX V device.
10. Right click on the **CFI_512Mb** device and click **Change File**.

11. Open the **master_image\output_file.pof**.
12. Tick the **Program/Configure** checkbox for the **CFI_512Mb** device including **Page_0**, **ext_flash.hex** and **OPTION_BITS**.
13. Click **Start** to start programming the CFI flash.

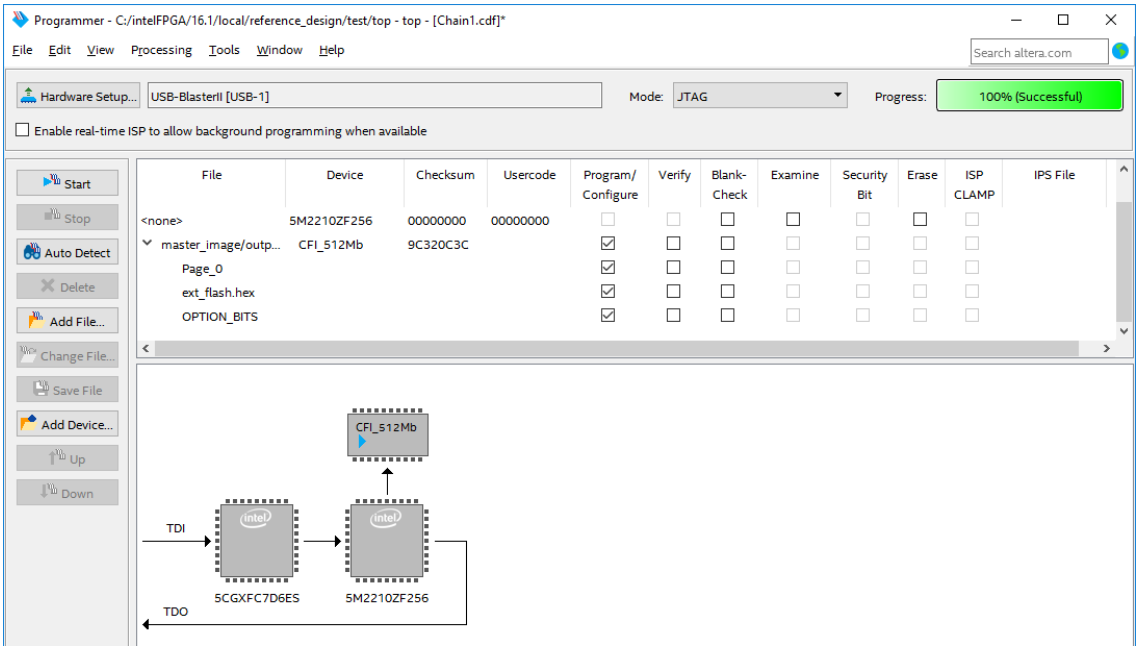


*Figure 23: Programming CFI Flash Window*

14. After the programming successful, right click on the MAX V device **5M2210ZF256** and click **Change File**.
15. Open the **max5.pof** from *<kit installation directory>/ cycloneVGX_5cgxfc7df31es_fpga/ examples/ max5*.
16. Tick the **Program/Configure** checkbox for MAX V **5M2210ZF256** device, including **CFM** and **UFM**.
17. Click **Start** to start programming the MAX V device.

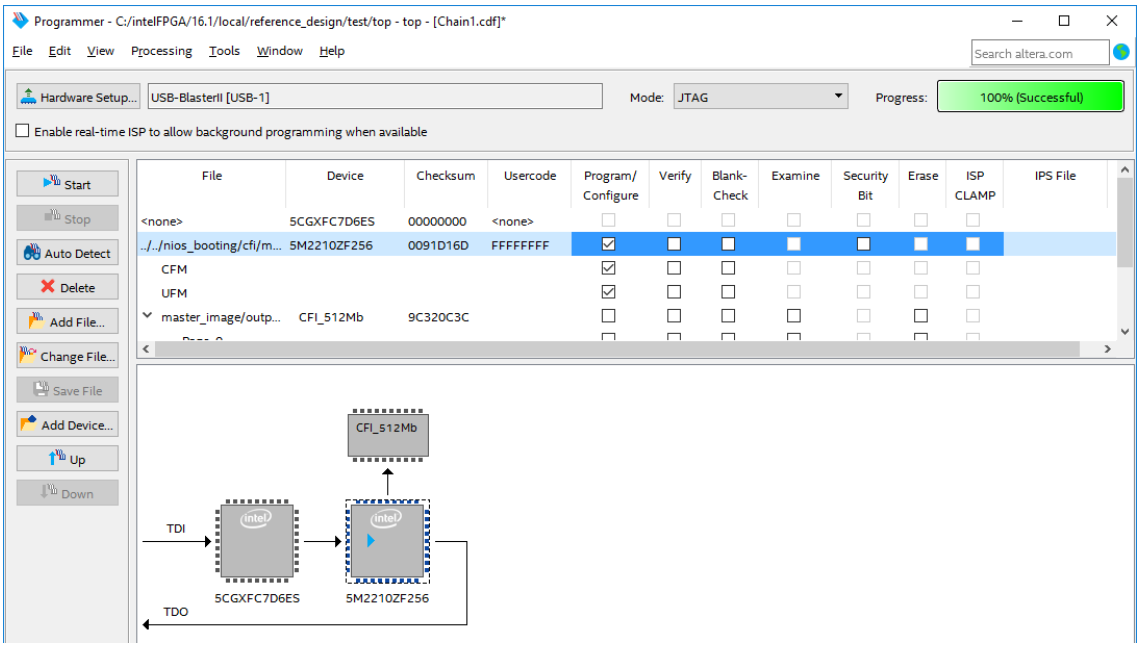


*Figure 24: Programming MAX V Window*

18. Power cycle the board.

## 6.7 Run the Design

Once you have programmed the CFI flash, the design is ready to be executed.

1. Launch **Nios II Command Shell**.
2. Run nios2-terminal and you should see **Hello from Cyclone V GX Nios II!**.

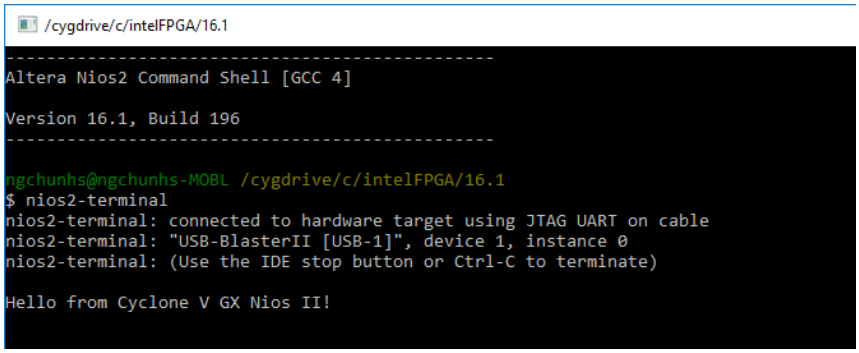


*Figure 25: Nios II Command Shell*

# 7. Revision History

| Revision Date | Comments | Author |
|---|---|---|
| 31 July 2017 | First release. | Ng, Chun Hsien |
| | | |