Getting Started with Intel® Quartus® Prime v21.0 and the Intel® MAX® 10-10M08 Evaluation Kit

By Sean D. Liming and John R. Malin
Annabooks – www.annabooks.com

October 2022

The Intel Max 10 is the lower cost and lower end of the Intel FPGA family. The Max 10 FPGAs are a popular solution for many applications. The Max 10 can be found on development boards featuring the higher-end Intel FPGAs. The MAX 10 -10M08 Evaluation Kit developed by Terasic Inc. is a very simple development board to help one get started developing with the Quartus Prime software. As a getting started to our hands-on articles on Intel FPGAs with the latest software, this article walks through a couple of basic designs to demonstrate the latest Quartus Prime V21.0 software targeting the MAX 10 -10M08 Evaluation Kit.

The Project Requirements:

- Intel Quartus Prime Lite Edition V21.0 – 7zip is required to extract the tar file.
- Intel® MAX® 10 - 10M08 Evaluation Kit and the schematic for the evaluation board are required. The schematic PDF file can be downloaded from the Intel FPGA website.
- Intel FPGA Programming cable – USB Blaster II or EthernetBlaster II. Unlike other FPGA boards, the MAX 10 -10M08 Evaluation Kit doesn't have a built-in USB Blaster solution, so a separate programming cable is required.

Note: There are equivalent MAX 10 development and evaluation boards available. These boards can also be used as the target, but you will have to adjust to the available features on the board. Please make sure that you have the board's schematic files as these will be needed to identify pins.

1.1 Download and Install the Quartus Prime Lite Edition Software

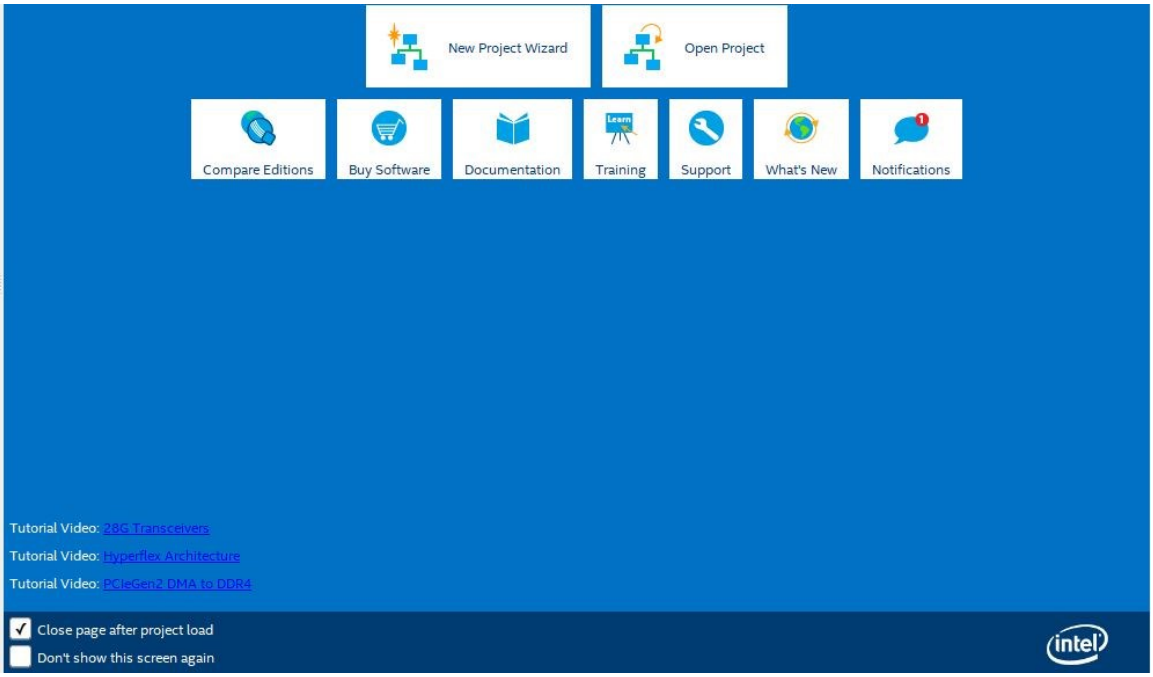
The Quartus can run on Windows and different Linux distributions. The installation of the software has many steps. Please see the article *Intel® Quartus® Prime Lite and NIOS® II SBT for Eclipse Installation Instructions* on Annabooks.com to install the software needed for this hands-on exercise. All you need for this hands-on paper is the installation of the Quartus Prime Lite software with Max 10 support.

1.2 LED Toggle Project

As hello world is the first programming project, toggling an LED is the first electronic circuit project. The MAX 10 -10M08 Evaluation Kit has a set of 5 LEDs on board available for user control. One of the LEDs will be toggled on and off using the 50Mhz clock on the board.

1.2.1 Create the Project

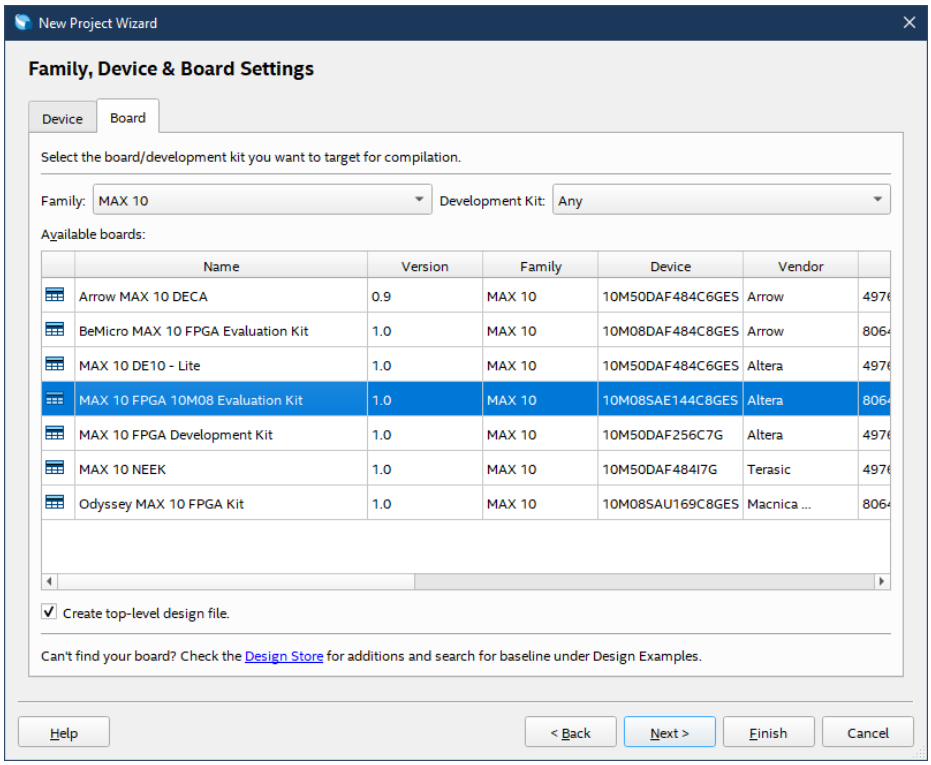
1. Open Quartus.
2. Click on the New Project Wizard.



3. Click Next to the Introduction dialog.
4. Select or create a new project directory (other than the Quartus installation directory) and name the project “LedToggle”. Click Next.

Note: By default, the root directory is the Quartus installation directory. Make sure the root project directory is a separate path from the Quartus installation files. i.e. c:\MAX10DesignExamples

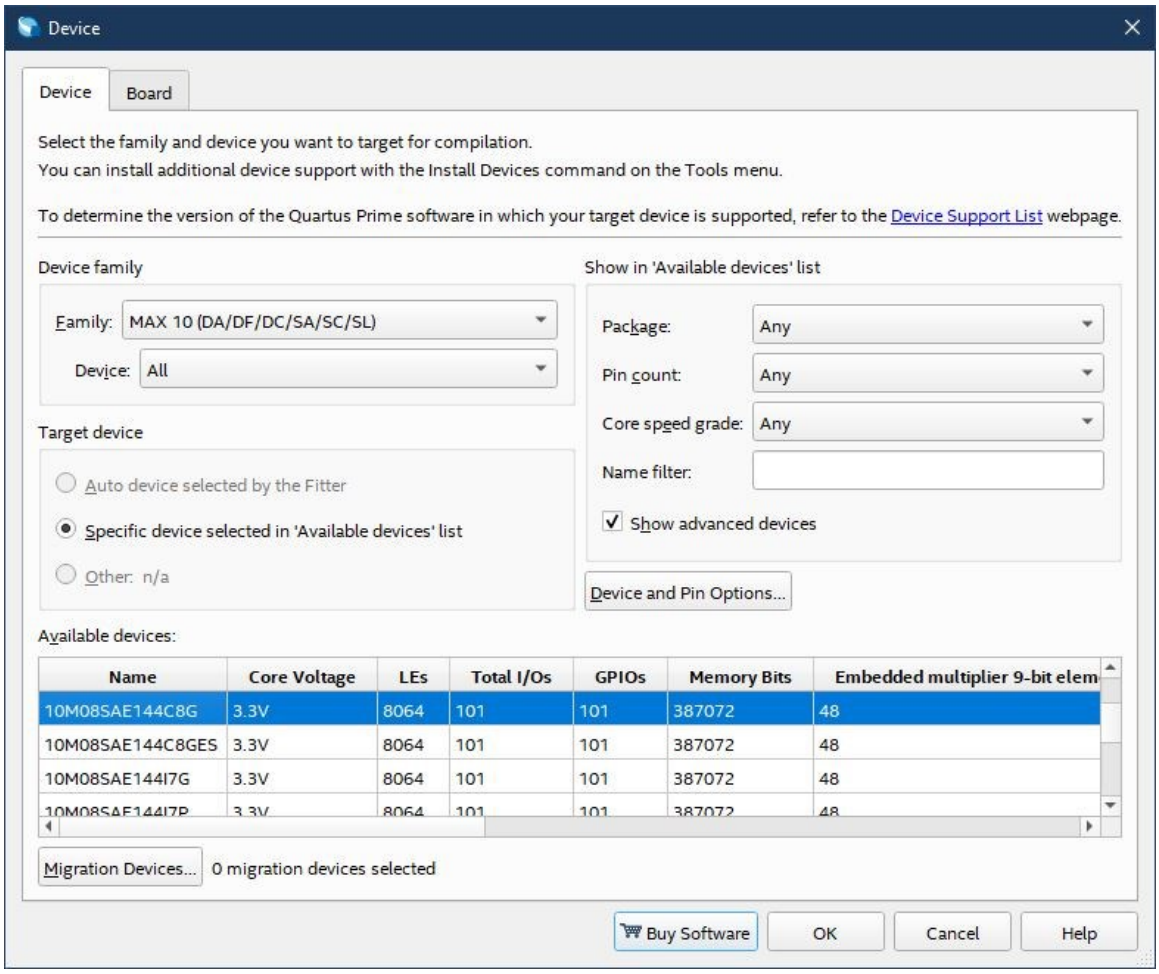
5. Project Type: Empty project, click Next.
6. Add File: no files to add, click Next.
7. Family, Device & Board Settings: click the Board tab and select: MAX 10 FPGA 10M08 Evaluation Kit and click Next.



8. EDA Tools: click Next.
9. Summary: click Finish.

Note: The actual MAX 10 on our board is the 10M08SAE144C8G, thus it is not an Engineering Sample (ES). The next two steps change the device to the production device. Your experience with your hardware might be different.

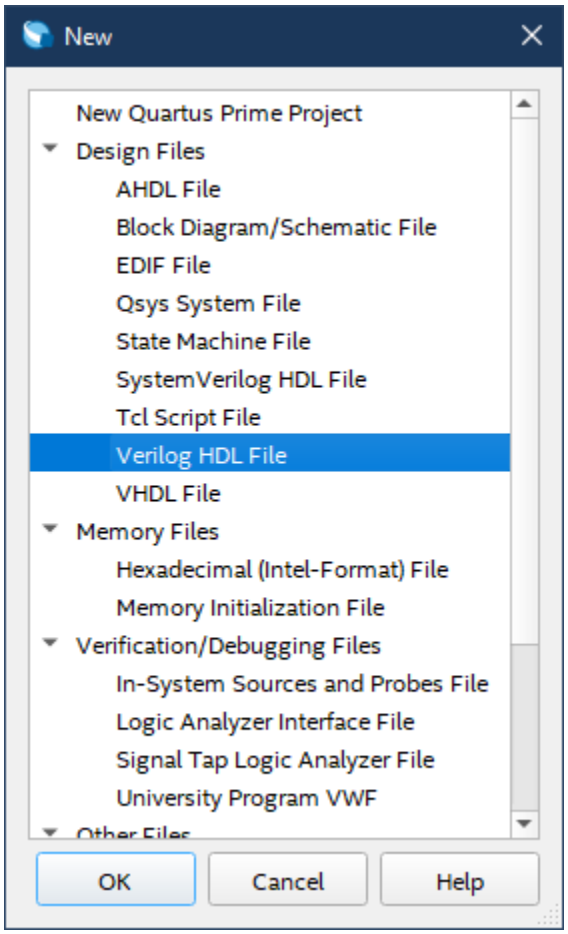
10. In the project navigation pane on the left, right-click on 10: 10M08SAE144C8GE, and select Device from the context menu.
11. In the Available devices, scroll down and select the 10M08SAE144C8G, click OK.



1.2.2 Create the Verilog file

Quartus supports many design types to create an FPGA design. Verilog is a popular Hardware Definition Language (HDL) that is like the C programming language. We will create a Verilog file to have the 50Mhz clock toggle the LED.

1. From the menu, select New, and from the New dialog, select Design Files--> Verilog HDL File.



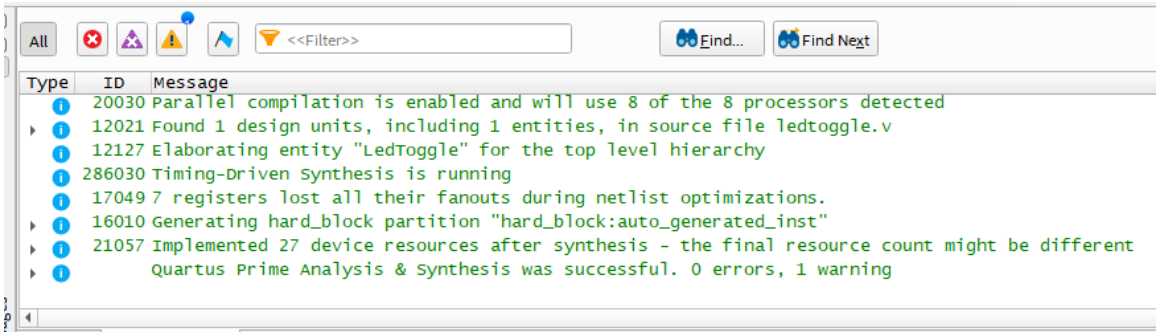
2. In the text area, write the following code:

```
//LED Toggle
//New Module
module LedToggle (

        input wire clk_50Mhz,
        output wire LEDD1
);

//Create a register
reg [31:0] counter;

//Assign the LED to one of the bits in the counter
assign LEDD1 = counter[25];

initial begin

    counter <= 32'h0;

end

always @(posedge clk_50Mhz) begin

    counter <= counter + 1;
```

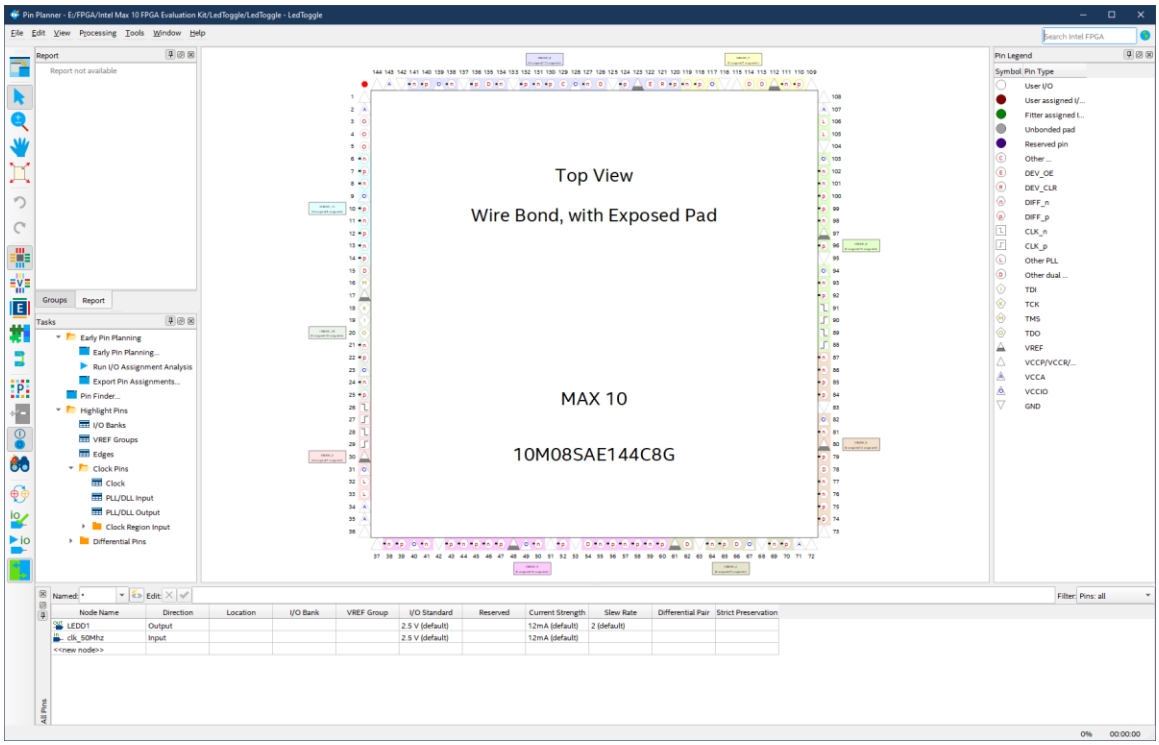
```
end

endmodule
```

The Verilog file is considered a module. Within the module creation, the input and output pins are defined for the clock and the LED. A 32-bit register is then created, and the LED is assigned to bit 26 of the register. The register is set to 0h on startup. The always statement says that on every positive edge of the clock, the register will be incremented by one. The counter will roll over once the maximum count is reached. The LED will turn on and off as the 26th bit is set to 1 and 0.

3. Save the file as LedToggle.v to the project folder.
4. Now let's test compile the code to make sure there is nothing mistyped. Under Task, right-click on Analysis & Synthesis, and select Start from the context menu. The output should show that the compilation was successful. If there are any errors, fix them and recompile the project.



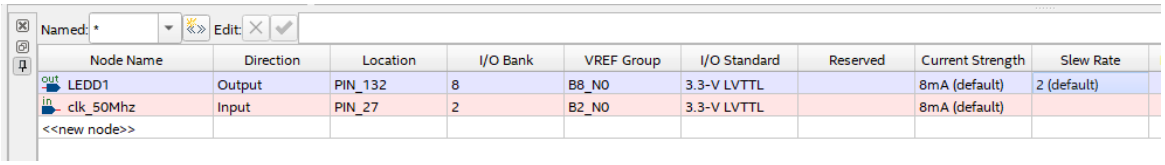
5. The pin assignments need to be set. From the menu, select Assignments->Pin Planner. The wire keyword populated the input and output in the Node Name list at the bottom of the Pin Planner dialog.



6. Using the board schematic, locate the pins for the LEDD1 and the 50Mhz clock. Set the Location values for both node names. For the MAX 10 – 10M08 Evaluation Board, these values are as follows:

   LEDD1: PIN_132
   Clk_50MHz: PIN_27

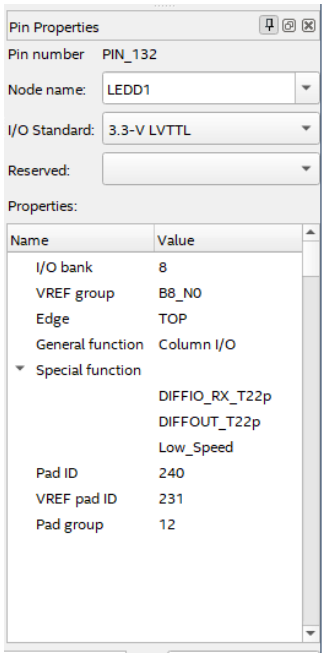
7. Set the I/O Standard to 3.3V-LVTTL. You can see from the schematic that the I/O are all tied to 3.3V.



Named: * Edit:

Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard	Reserved	Current Strength	Slew Rate
LEDD1	Output	PIN_132	8	B8_N0	3.3-V LVTTL		8mA (default)	2 (default)
clk_50Mhz	Input	PIN_27	2	B2_N0	3.3-V LVTTL		8mA (default)	
<<new node>>								

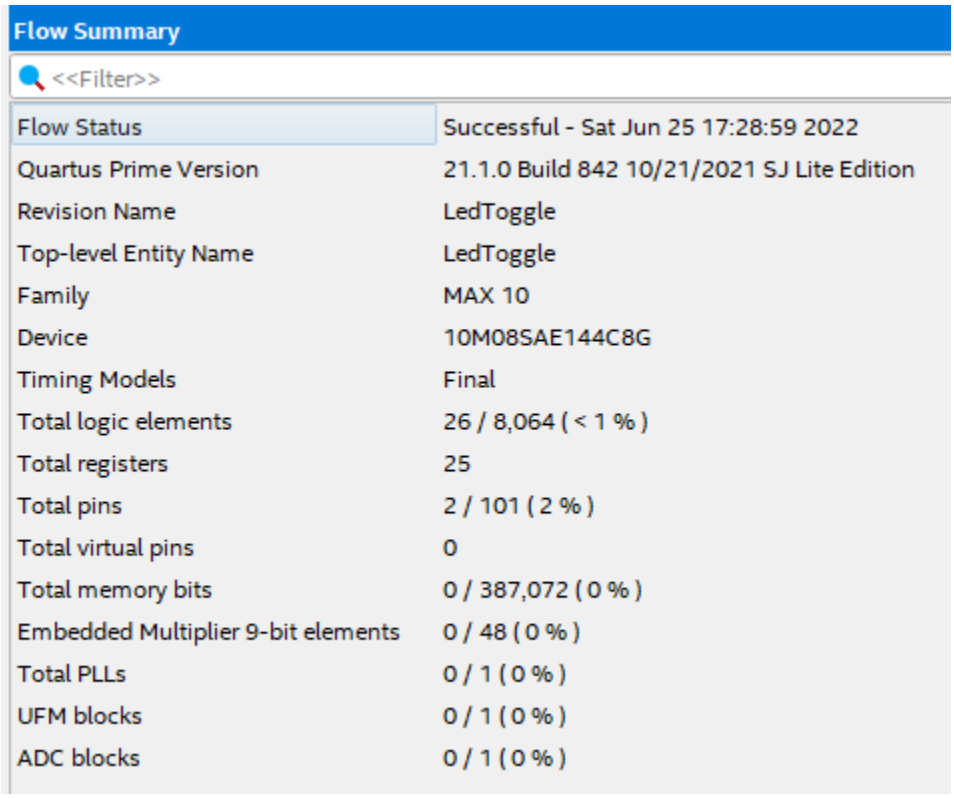
The chip pin diagram will show that both pins are assigned. You can double-click on a pin and the properties pane will appear with the pin information.



8. Close the Pin Planner when finished.
9. Finally, create a timing constraints file. From the menu, select New->Text File.
10. Enter the following:

```
#Set timing constraint of clk_50Mhz is 50Mhz

create_clock -name clk_50Mhz -period "50Mhz" [get_ports clk_50Mhz]

#LEDD1 has no timing constraints

set_false_path -from * -to [get_ports LEDD1]
```

11. Save the file as LedToggle.sdc to the project folder.
12. Finally, compile the design. In the Task pane, right-click on Compile and Design and select Start from the context menu, or you can click on the Play symbol in the toolbar. Make sure the project compiles successfully.



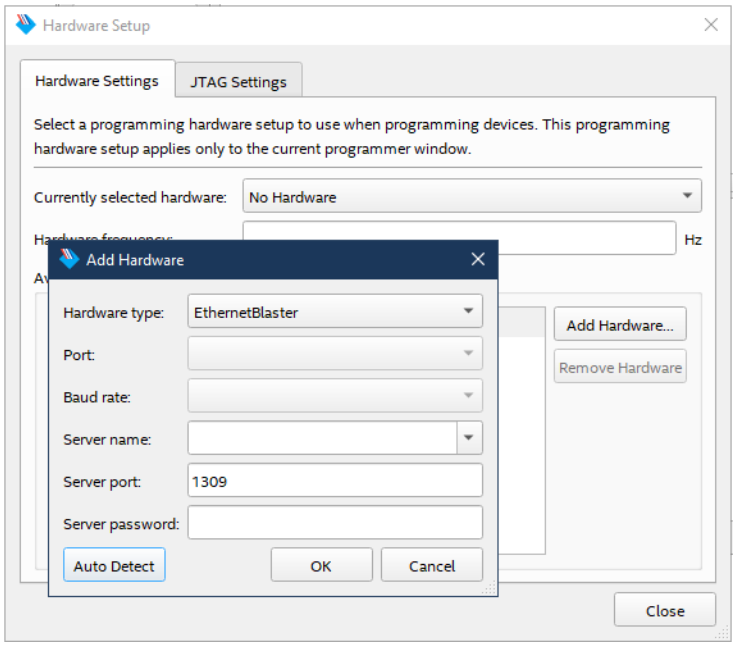
1.2.3 Program the Board

With the design compiled, we can now test the design on the board.

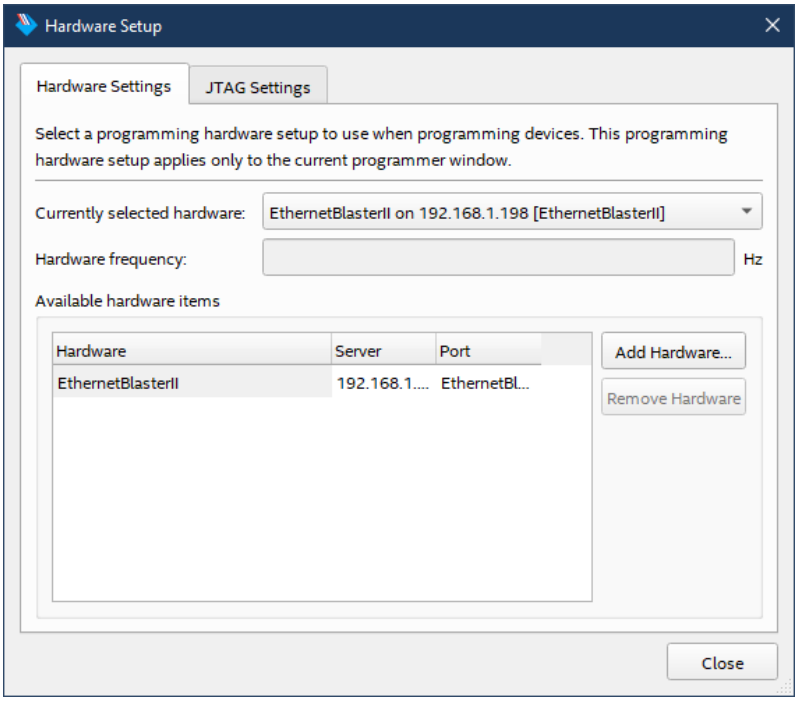
1. Connect the board and the programming cable together per the cable instructions.

Note: The MAX 10 – 10M08 Evaluation Kit doesn't come with a programming cable or built-in JTAG USB Blaster II. You will have to use either the USB Blaster II or EthernetBlaster II external cables. The EthernetBlaster II was used in this example. DHCP setup was not working, so a direct Ethernet cable connection between a PC and the EthernetBlaster II had to be made. Set the static IP for the PC network card to 198.162.0.1. Access the EthernetBlaster II via a browser and then change the IP to a static IP that matches the network. The new IP address was used as the Server name in this example. Your experience might be different.

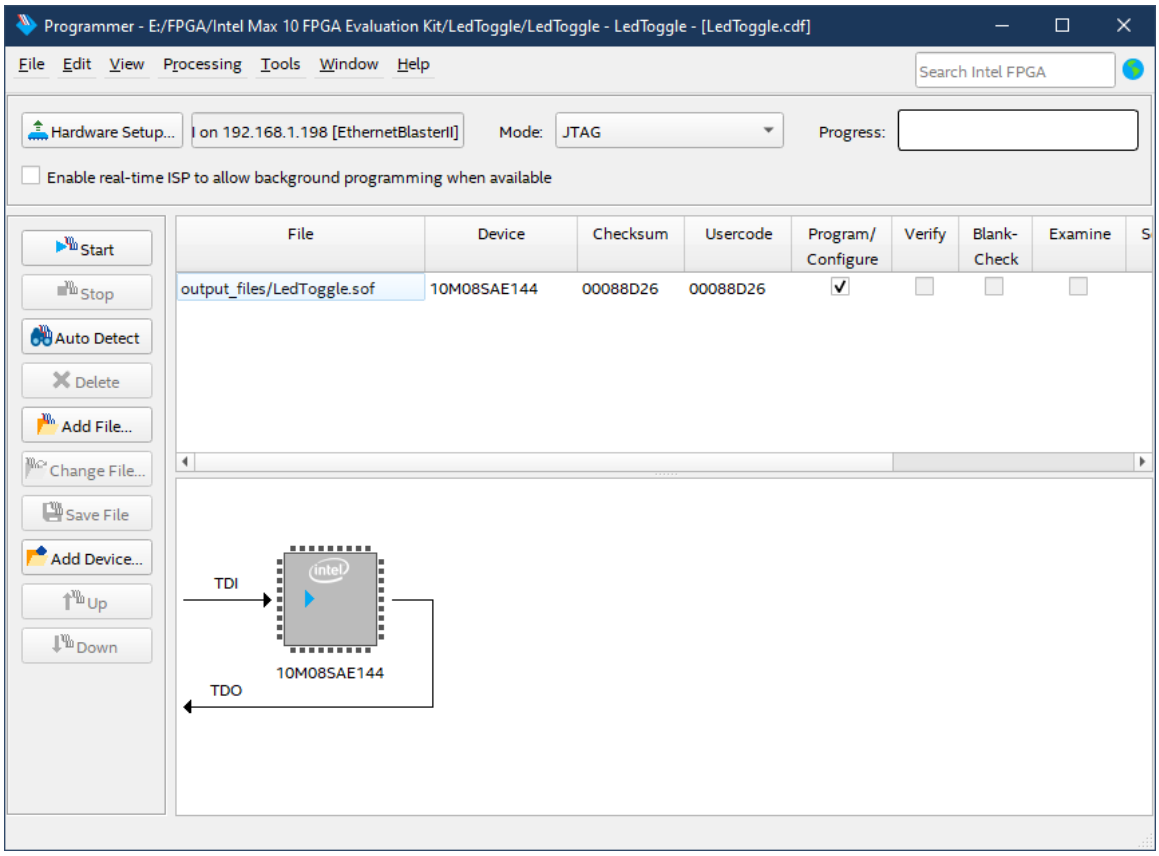
2. Power on the board and the programming cable box.
3. In Quartus Prime, from the Task pane, right-click on Program Device (Open Programmer) and select Open from the context menu.
4. The Programmer dialog appears. Click on the "Hardware Setup" button.
5. Click the Add hardware button, select the Hardware type and file and any remaining information, and click OK.



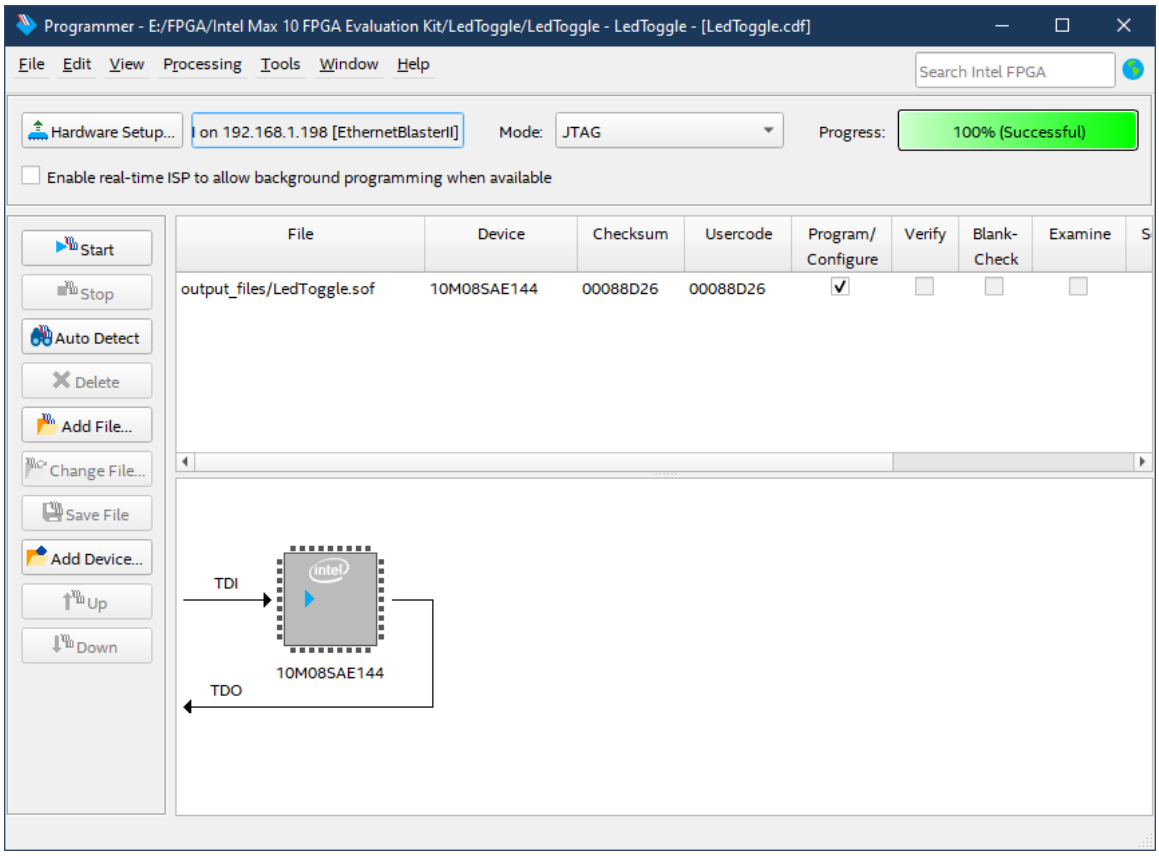
6. The tool allows you to connect to a number of programming cables. We need to select the one for our board. In the “Currently selected hardware”, click the drop-down, select the hardware cable for the board, and click Close when finished.



7. A LedToggle.sof file gets created during the Compile Design flow. The file is automatically filled in. There is only one FPGA on the board and in the JTAG chain, so the file already has the Program/Configure checkbox checked. Click the Start button.



The process takes a few seconds and shows that the task was completed successfully.



If you look at the board, you will see that the LED D1 is blinking.

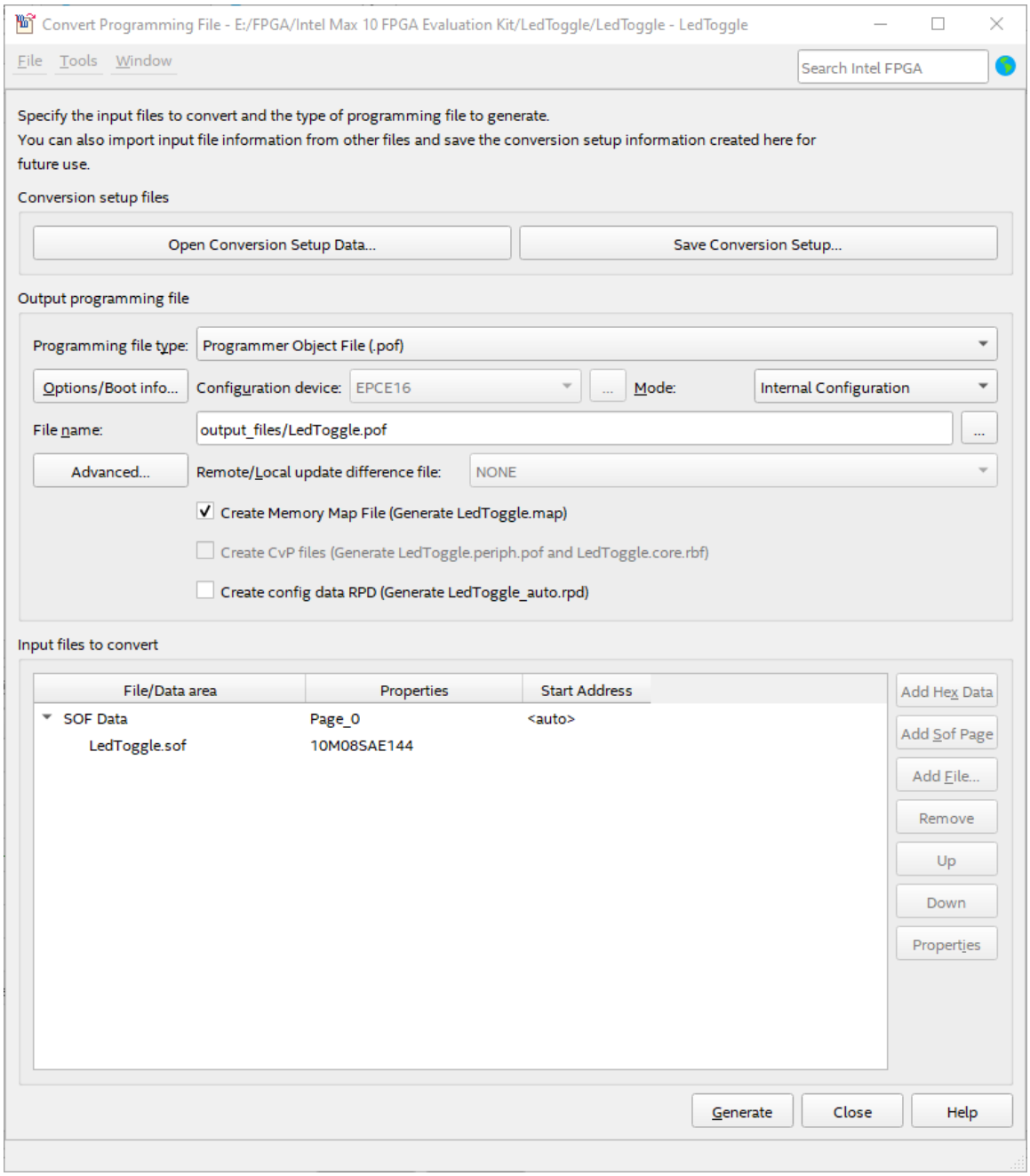
1.2.4 Program the Internal Flash with POF file.

The downloading of the .SOF file was to the SRAM of the MAX 10. Once the system has been power cycled the design is lost. In this section, we will convert the LedToggle.sof to LedToggle.pof that will be programmed into the internal flash and stay resident.

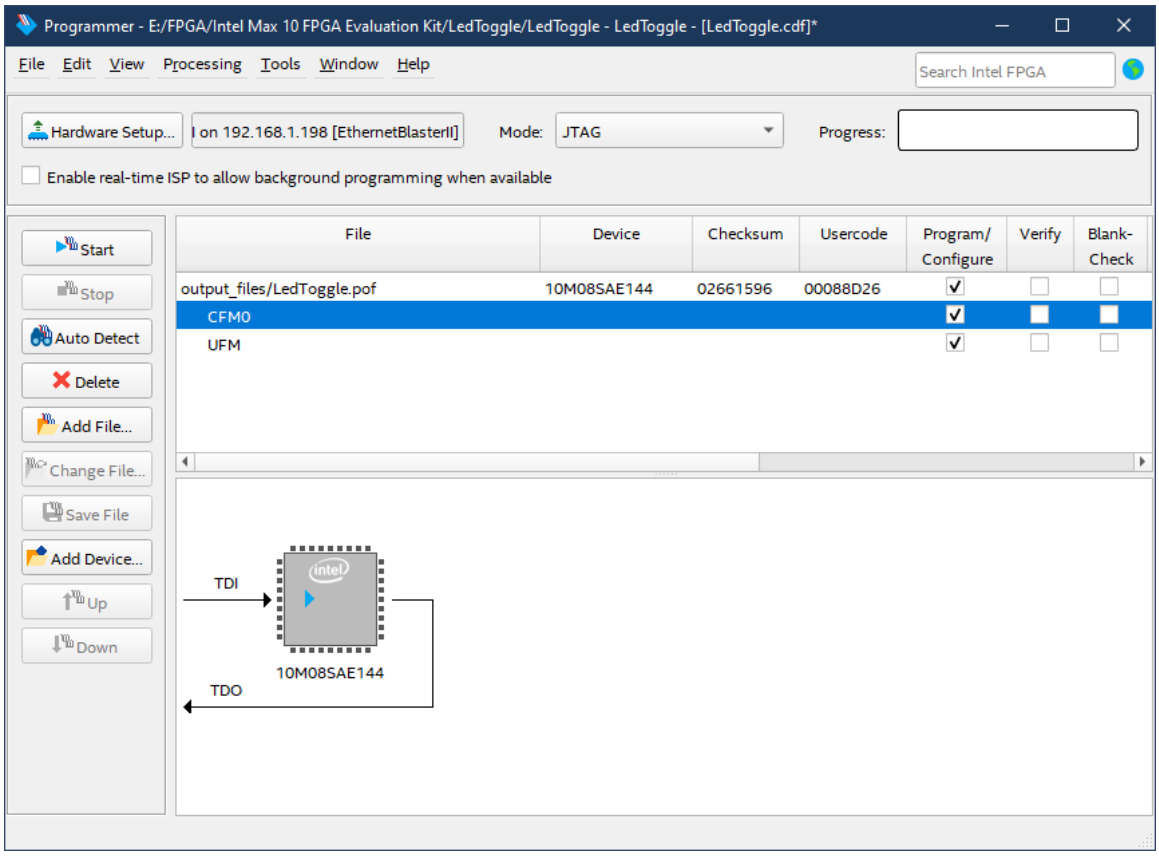
1. Unplug the power for the Evaluation Board, wait a few seconds, and then plug the power back in. You will notice that the design doesn't come back up and flash the LED. The .SOF file is downloaded to the internal SRAM and lost on a power cycle.
2. The MAX 10 has internal flash that provides non-volatile storage for the program allowing it to persist through a power-down and launch on power on. We need to convert the .SOF file to a .POF file to program the internal flash. In Quartus, from the menu, select File->Convert Programming Files…
3. In the Convert Programming Files dialog set the following:

   Programming file time: Programmer Object File (.pof).
   Mode: Internal Configuration.
   File name: output_files/LedToggle.pof.

4. Under the Input files to convert, click Add File and browse to the output folder location to open the LedToggle.sof file.



5. Click the Generate button. A dialog will appear stating generation was success full. Click OK.
6. Click Close to close the Convert Program File dialog.
7. In Quartus Prime, from the Task pane, right-click on Program Device (Open Programmer) and select Open from the context menu.
8. The programming cable should already be selected. If not, select your programming cable.
9. In the file list, right-click on the LEDToggle.sof file name and select Change File.
10. Open the LedToggle.pof file.
11. Check the Program/Configure checkbox. This will automatically check the CFMo and UFM.



12. Click the Start button. The process will take a little longer since the internal flash is being programmed.
13. LED D1 should be blinking. Power cycle the board and the program should persist through the power cycling and start up and run this time.

1.3 Logic Switches Project

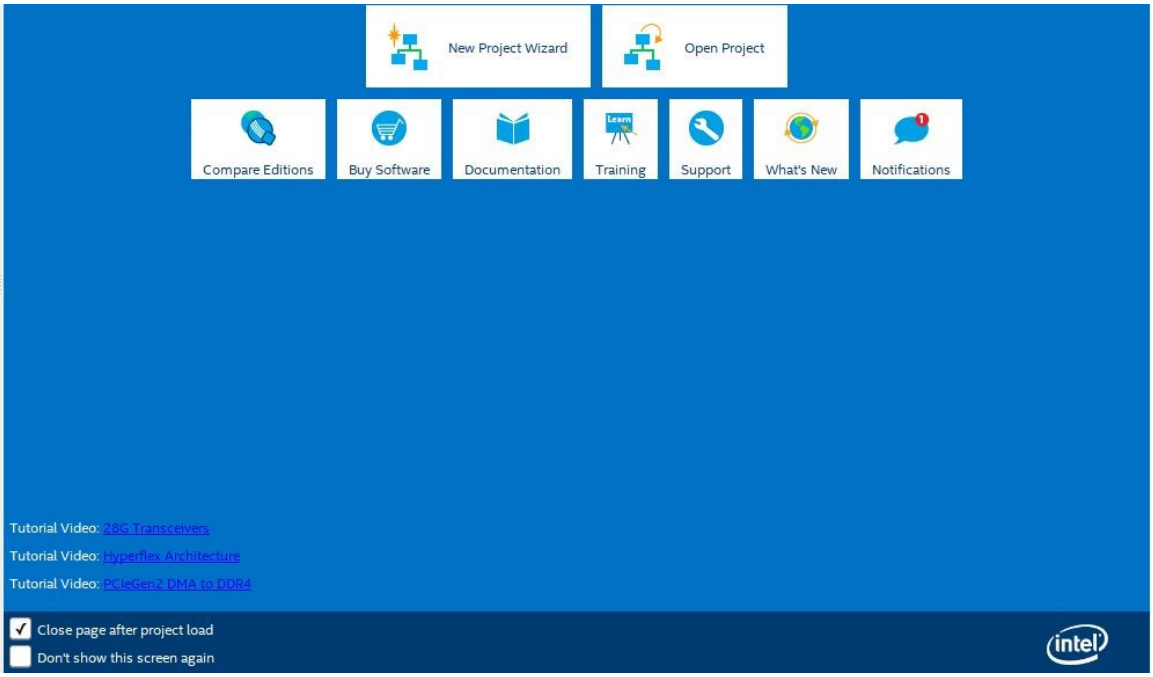
The previous project used Verilog to create the design. In the project, we will create a design using a schematic. The MAX® 10 - 10M08 Evaluation Kit has a 5-switch pack on board. Three switches will be used to light up LED D2 and LED D3 as follows:

Input			**Output**	
Switch1	Switch2	Switch3	LEDD2	LEDD3
0	0	0	0	0
0	0	1	0	0
0	1	0	1	0
0	1	1	1	1
1	0	0	1	0
1	0	1	1	1
1	1	0	1	0
1	1	1	1	1

Looking at the schematic, the LEDs are tied to 3.3 V. The LEDs turn on when the output pin is low. When the switches slide to the ON position, they are in the logic 0 state, and when the switches slide to the OFF position, they are in the logic 1 state.

1.3.1 Create the Project

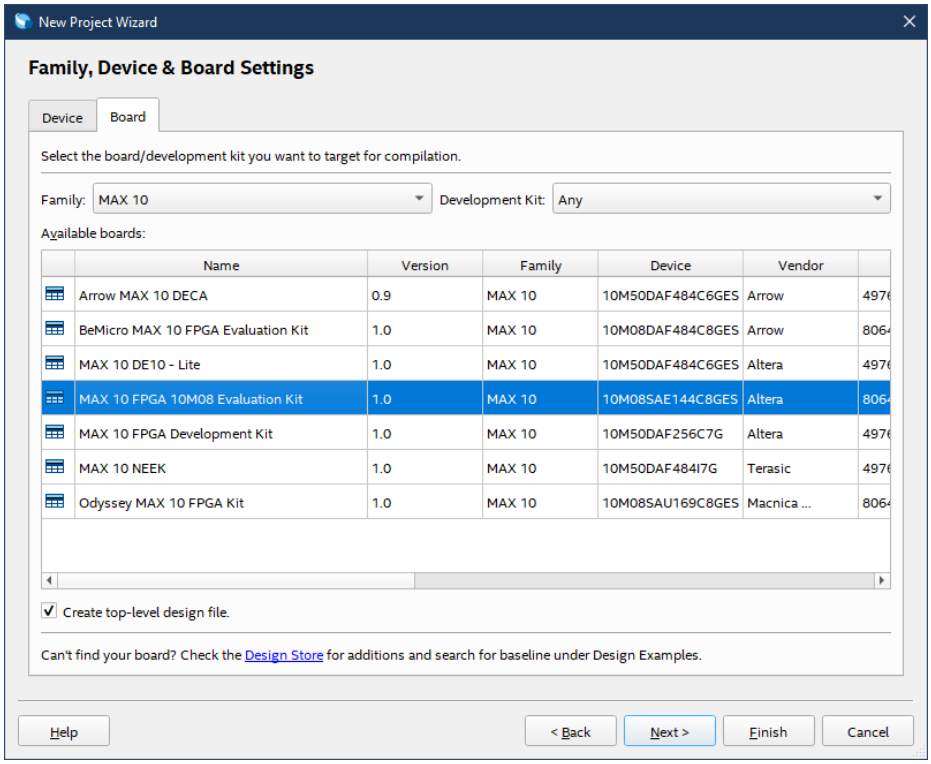
1. Open Quartus.
2. Click on the New Project Wizard.



3. Click Next to the Introduction dialog.
4. Select the directory (other than the Quartus directory) and name the project: SwitchLogic. Click Next.

Note: By default, the root directory is the Quartus installation directory. Make sure the root project directory is a separate path from the Quartus installation files. i.e. c:\MAX10DesignExamples.

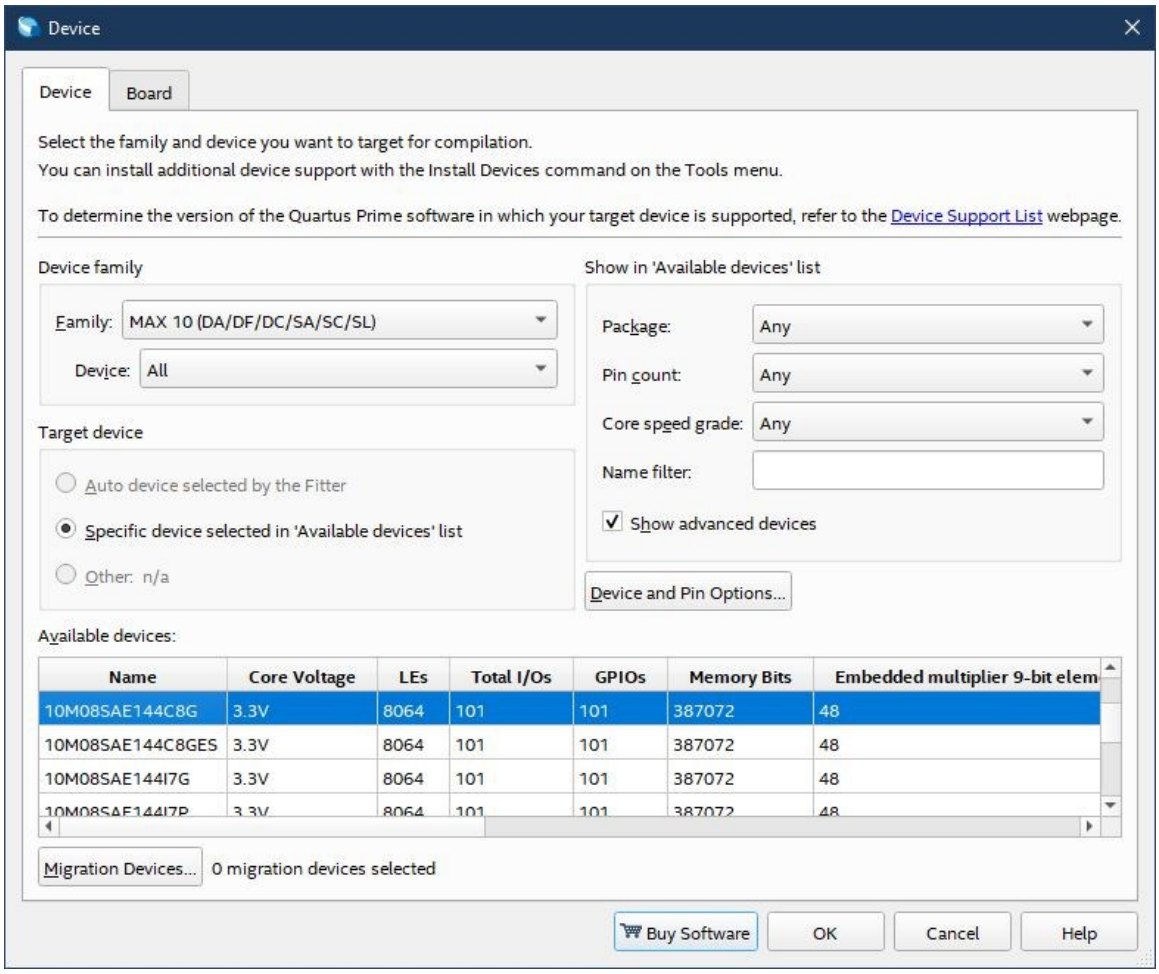
5. Project Type: Empty project, click Next.
6. Add File: There are no files to add, click Next.
7. Family, Device & Board Settings: click the Board tab and select: MAX 10 FPGA 10M08 Evaluation Kit. Click Next.



8. EDA Tools: click Next.
9. Summary: click Finish.

Note: The actual MAX 10 on our board is the 10M08SAE144C8G, thus it is not an Engineering Sample (ES). The next two steps change the device to the production device. Your experience might be different depending on your hardware.

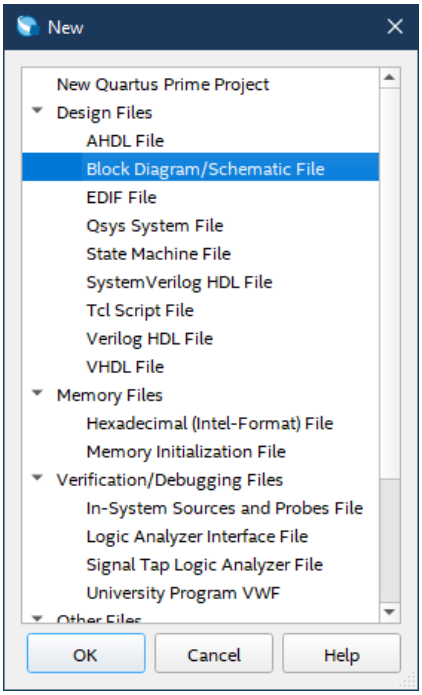
10. In the project navigation pane on the left, right-click on 10: 10M08SAE144C8GE and select Device from the context menu.
11. In the Available devices, scroll down and select the 10M08SAE144C8G. Click OK.



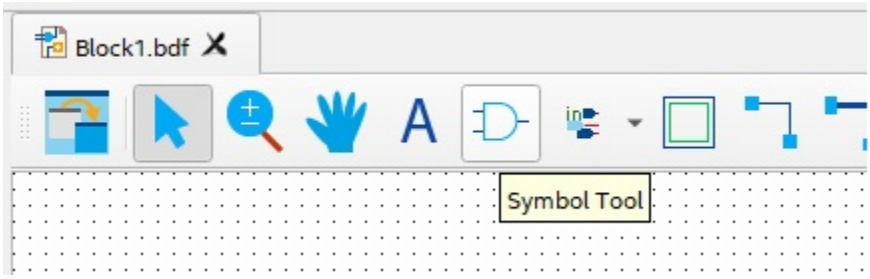
1.3.2 Create the Schematic File

Now we will create the schematic file.

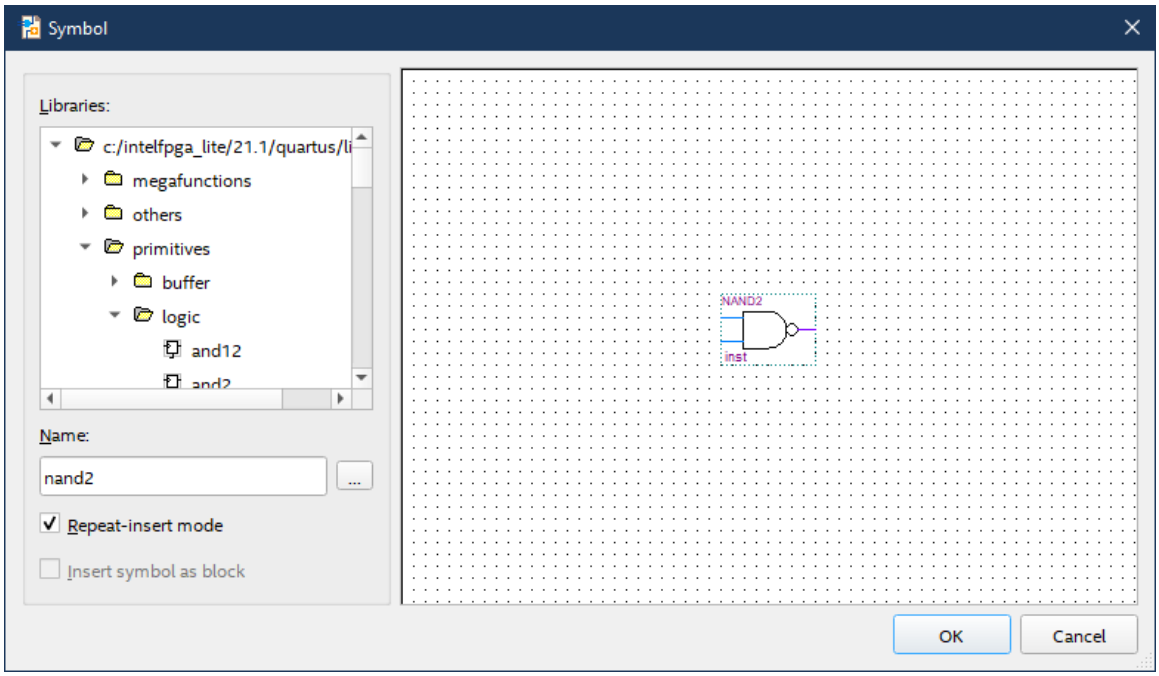
1. Open Quartus.
2. From the menu, select New; and from the New dialog, select Design Files--> Block Diagram/Schematic File.



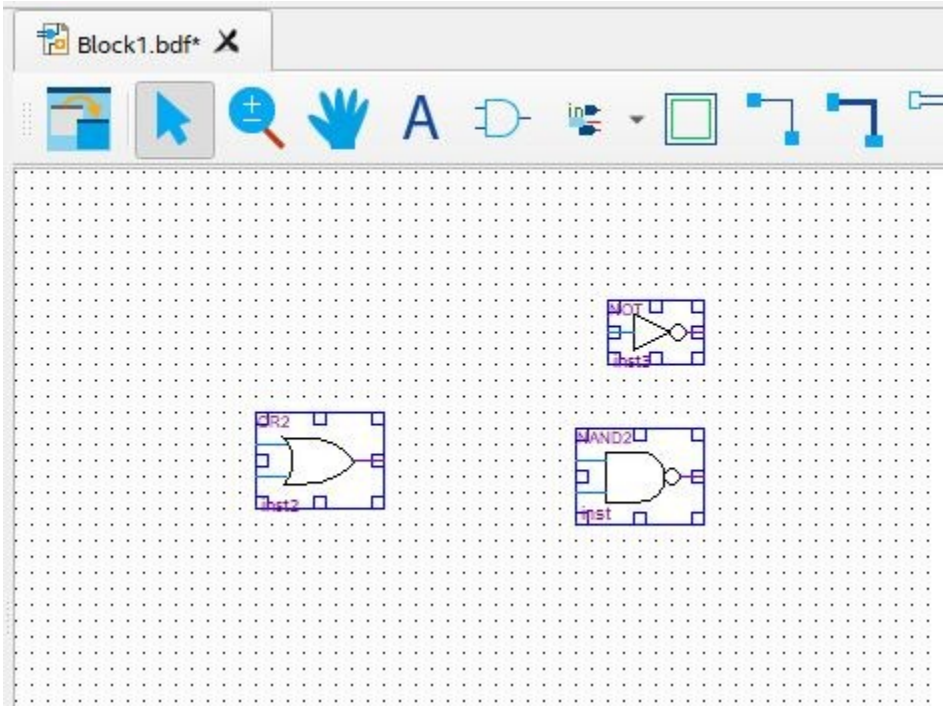
3. A blank schematic editor appears. In the tools bar, click on the AND symbol to open the Symbol Tool, or double-click anywhere in the empty space of the design.



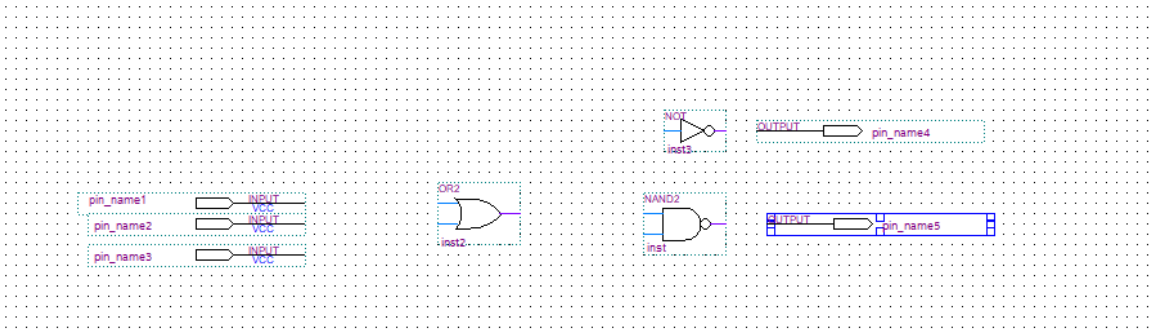
4. The Symbol dialog appears, which contains the libraries of available symbols from which to choose. Expand the hierarchy to get to the logic symbols: primitives->logic.



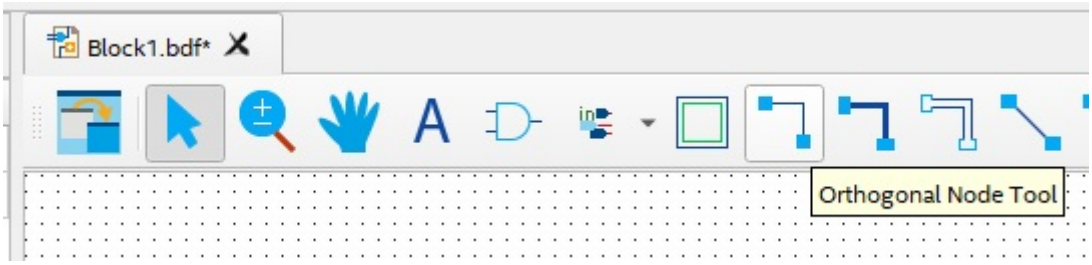
5. Scroll down the list and click on nand2. This is a 2-input NAND gate.
6. Click Ok. The dialog will disappear and the mouse will have the symbol attached.
7. Drag and place (right-click) the NAND gate on the schematic diagram.
8. The tool allows you to add more NAND gates, and we only need one. Hit the ESC key when finished.
9. Repeat the steps to add an OR gate (or2) and a NOT gate (inverter).



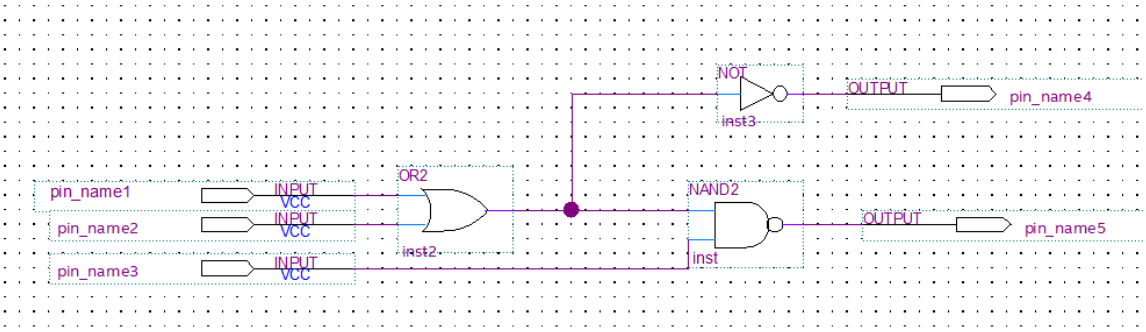
10. Open the Symbol Tool, and expand the hierarchy to primitives->pin.
11. Select input and click OK.
12. Drop the input symbol three times and then hit ESC.
13. Repeat steps 10-12 to add 2 output pins.



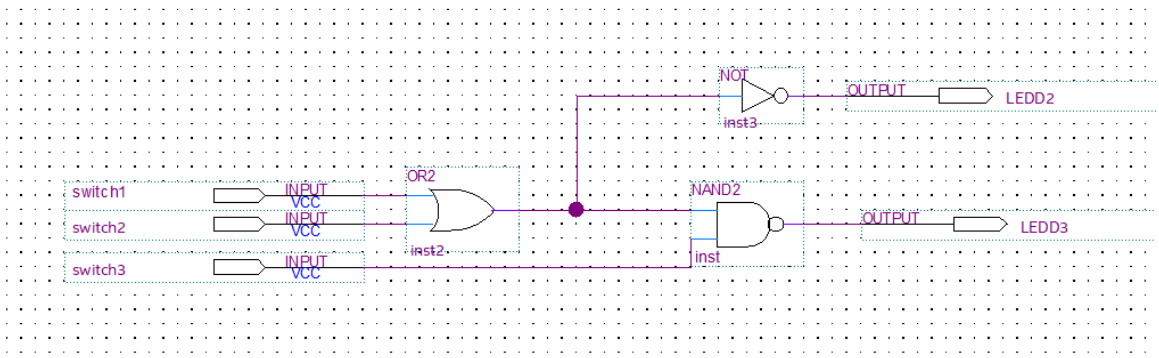
14. From the toolbar, click on the Orthogonal Node Tool. The tool is used to connect the wires between the gates and pins.



15. Using the Orthogonal Node tool, connect the gates and pins as follows:

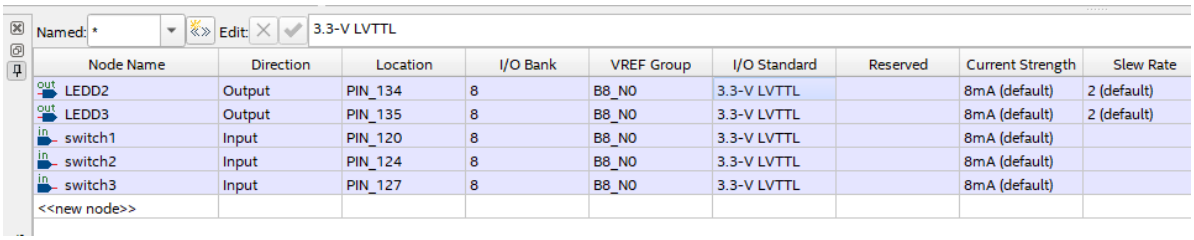


16. The final step is to assign the proper names to the pins. Double-click on the name pin_name1 to edit the name. Rename to switch1.
17. Rename pin_name2 to switch2.
18. Rename pin_name3 to switch3.
19. Rename pin_name4 to LEDD2.
20. Rename pin_name5 to LEDD3.



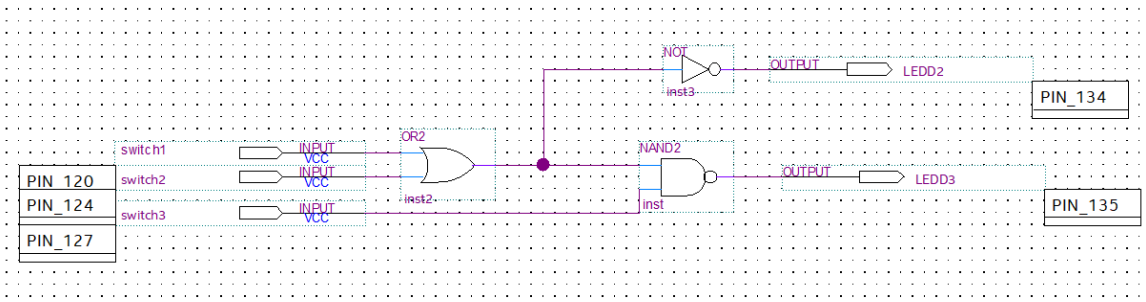
21. Save the file as SwitchLogic.bdf.
22. Now, we need to assign the pins to the input and output pins in the schematic. In the Task pane, right-click on Analysis & Synthesis. This will check the design and generate the pin hierarchy.
23. Correct any errors if they appear, but the process should return successfully.
24. From the menu, select Assignments->Pin Planner.
25. All the pins will be listed. Using the board schematic, set all the node names with the proper pin locations and change I/O Standard for all pins to 3.3V-LVTTL.

Node Name	Location
LEDD2	PIN_134
LEDD3	PIN_135
Switch1	PIN_120
Switch2	PIN_124
Switch3	PIN_127



Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard	Reserved	Current Strength	Slew Rate
LEDD2	Output	PIN_134	8	B8_N0	3.3-V LVTTL		8mA (default)	2 (default)
LEDD3	Output	PIN_135	8	B8_N0	3.3-V LVTTL		8mA (default)	2 (default)
switch1	Input	PIN_120	8	B8_N0	3.3-V LVTTL		8mA (default)	
switch2	Input	PIN_124	8	B8_N0	3.3-V LVTTL		8mA (default)	
switch3	Input	PIN_127	8	B8_N0	3.3-V LVTTL		8mA (default)	
<<new node>>								

26. Close the Pin Planner when finished. The schematic will now show the pin assignment names.

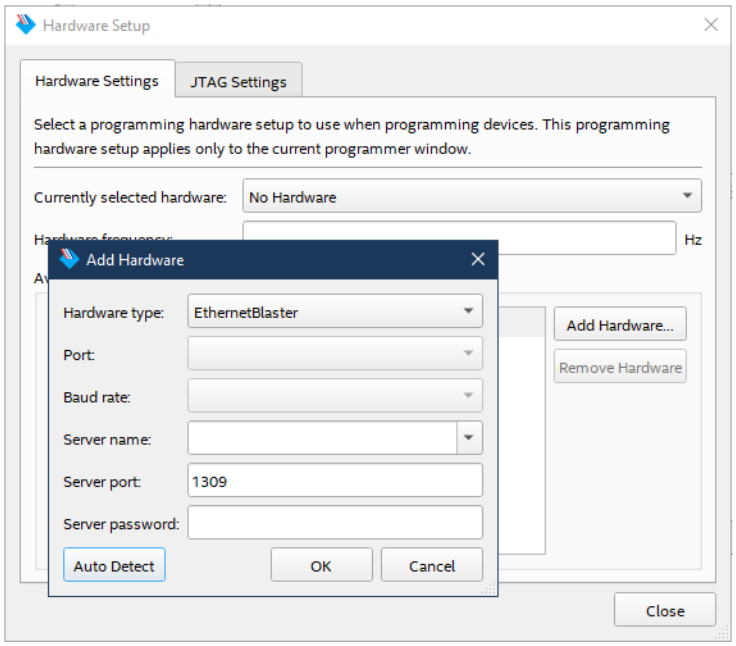


27. Save the project.
28. There is no clock, so a timing constraint file is not required. In the Task pane, right-click on Compile Design and select start from the context menu; or you can click on the Play symbol in the toolbar. The process should be successful.

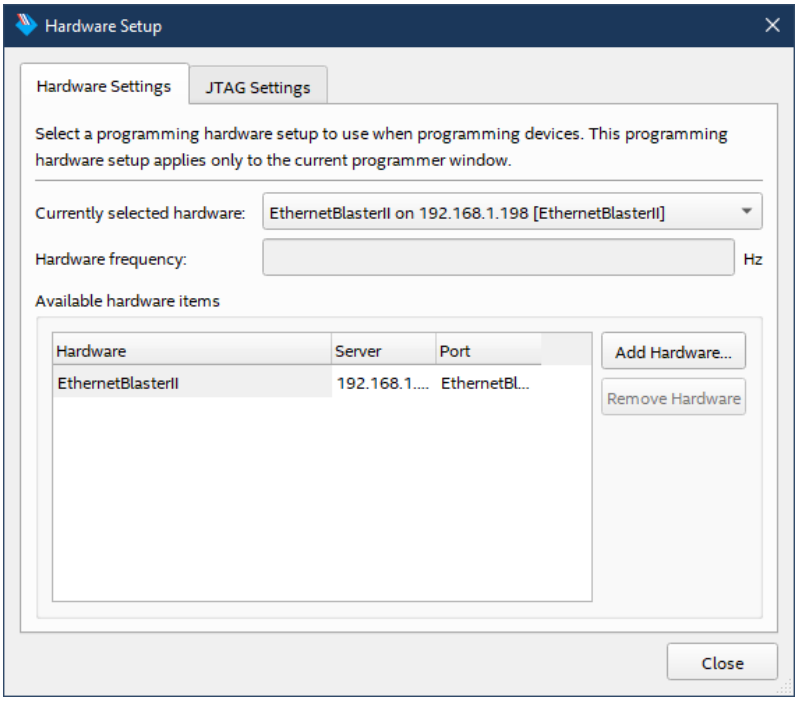
1.3.3 Program the Board

With the design compiled, we can now test the design on the board.

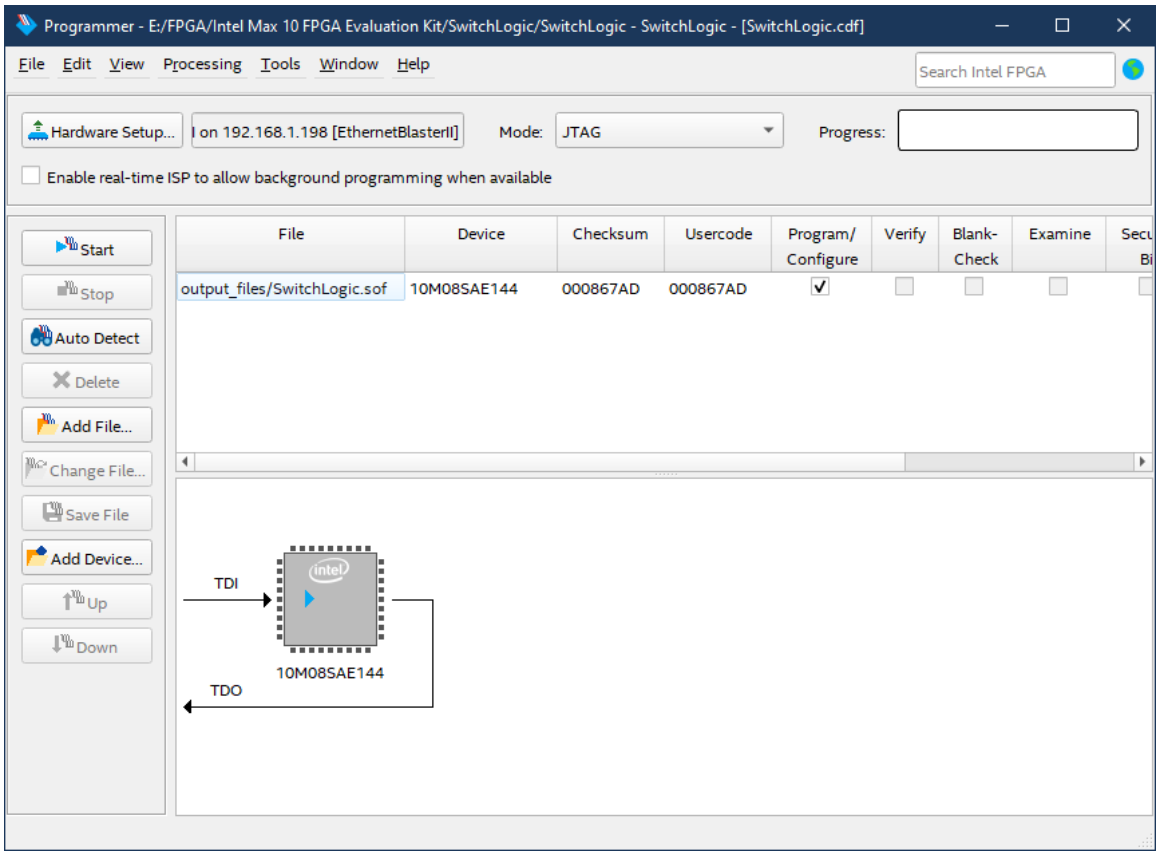
1. Connect the board to the programming cable per the cable instructions.
2. Power on the board and the programming cable box.
3. In Quartus Prime, from the Task pane, right-click on Program Device (Open Programmer) and select Open from the context menu.
4. The Programmer dialog appears. Click on the “Hardware Setup” button.
5. Click the Add hardware button, select the Hardware type and file in any remaining information, and click OK.



6. The tool allows you to connect to a number of programming cables. We need to select the one for our board. In the “Currently selected hardware”, click the drop-down and select the hardware cable for the board, and click Close when finished.



7. A SwitchLogic.sof file gets created during the Compile Design flow. The file is automatically filled in. There is only one FPGA on the board and in the JTAG chain so the file already has the Program/Configure checkbox checked. Click the Start button.



The process takes a few seconds and shows that the task completed successfully. The board will be running the design. Flip the switches on and off to make sure the LEDs light up per the truth table.

1.4 Summary: Getting Started

Quartus Prime is a powerful FPGA development tool. The deeper details of the tools' features and compilation steps can be found online at Intel.com The examples in this article provide an update to the Intel Quartus Prime Pro Edition User Guide: Getting Started for the latest Quartus Prime release.

1.5 References

The following references were used for this article:

- Electronics - Verilog - Blinking a LED - https://www.badprog.com/electronics-verilog-blinking-a-led
- How to Begin a Simple FPGA Design – intel.com Intel FPGA training site Intel® FPGA Technical Training / https://www.youtube.com/watch?v=bwoyQ_RnaiA
- *MAX 10 FPGA (10M08S, 144-EQFP) Evaluation Kit User Guide*, Intel (Altera), 2015