EARTH PEOPLE TECHNOLOGY, Inc

BEEPROLOGIC DEVELOPMENT SYSTEM
User Manual

The BeeProLogic is a CPLD development board that is designed to be user friendly and a great introduction into digital design for Electrical Engineering students and hobbyists. This board provides a simplified method for debugging programmable logic code. It has been designed with plenty of LEDs and Pushbuttons to allow a large amount of interaction between user and hardware operations. The board provides a convenient, user-friendly work flow by connecting seamlessly with Intel's Quartus Prime software. The user will develop the code in the Quartus environment on a Windows Personal Computer. The programmable logic code is loaded into the CPLD using only the Quartus Programmer tool.

http://www.earthpeopletechnology.com/

BeeProLogic Development System User Manual

Contents

BeeProLogic Development System User Manual

1 Introduction

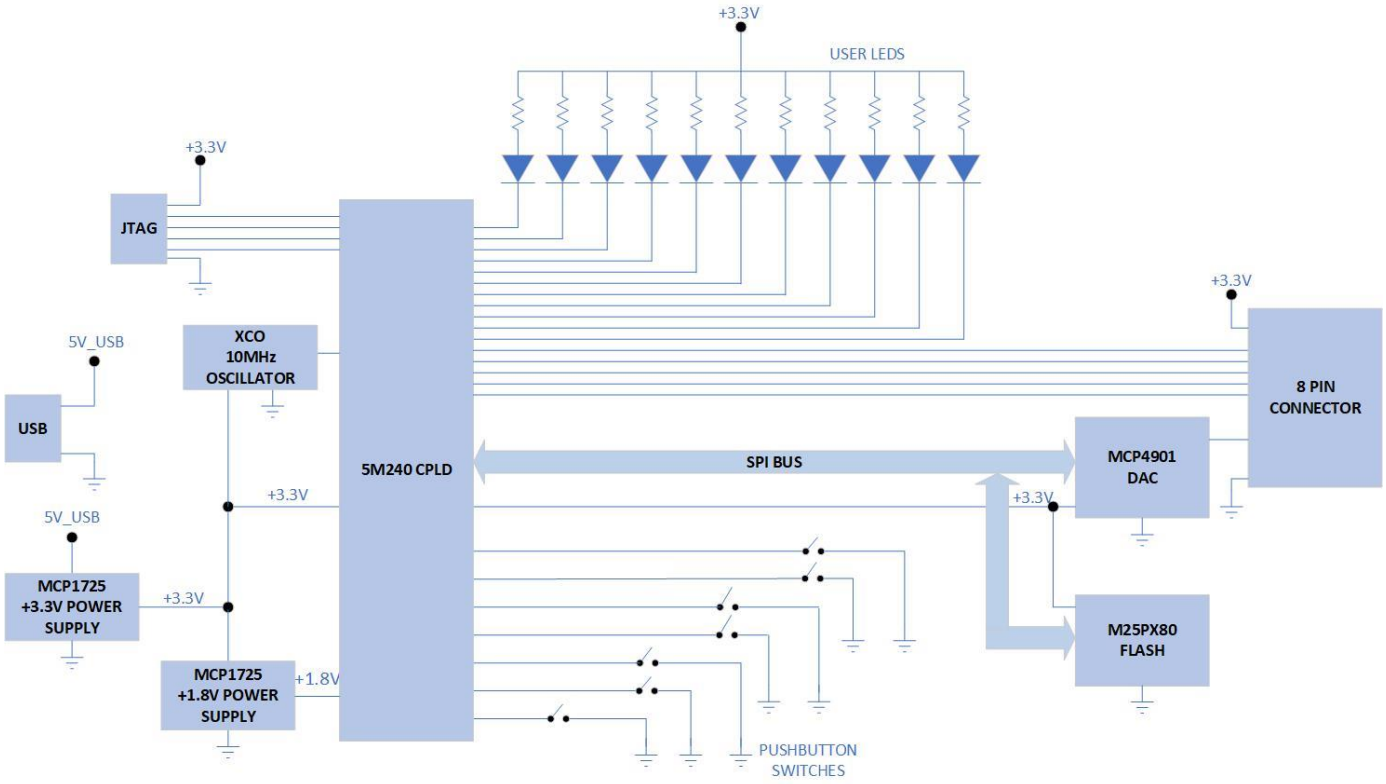
The Earth People Technology BeeProLogic CPLD development system hardware consists of a a CPLD, 8Mb Flash and a single channel DAC. The board requires an external JTAG Programmer compatible with the JTAG Blaster. The Core of the board is the MAX V chip. The board can be powered by standard USB Micro B connector. There is a 10MHz oscillator attached to the MAX V chip for clocking user code. The board also includes 10 Green User LEDS, Seven User Pushbuttons and One external connector with six pins for GPIO.

2 User Setup

The BeeProLogic is ready to go straight out of the box. Just connect power from a PC/Laptop. Next, connect a JTAG Blaster or an Intel USB Blaster to J7 for programming the MAX V chip. Please note, the BeeProLogic comes with an adapter to convert the six pin JST connector at J7 to the ten pin 0.1 Inch 5x2 header with the JTAG Blaster standard pinout. Then, write some code, synthesize, and program the chip. See section x for details about programming the BeeProLogic board.

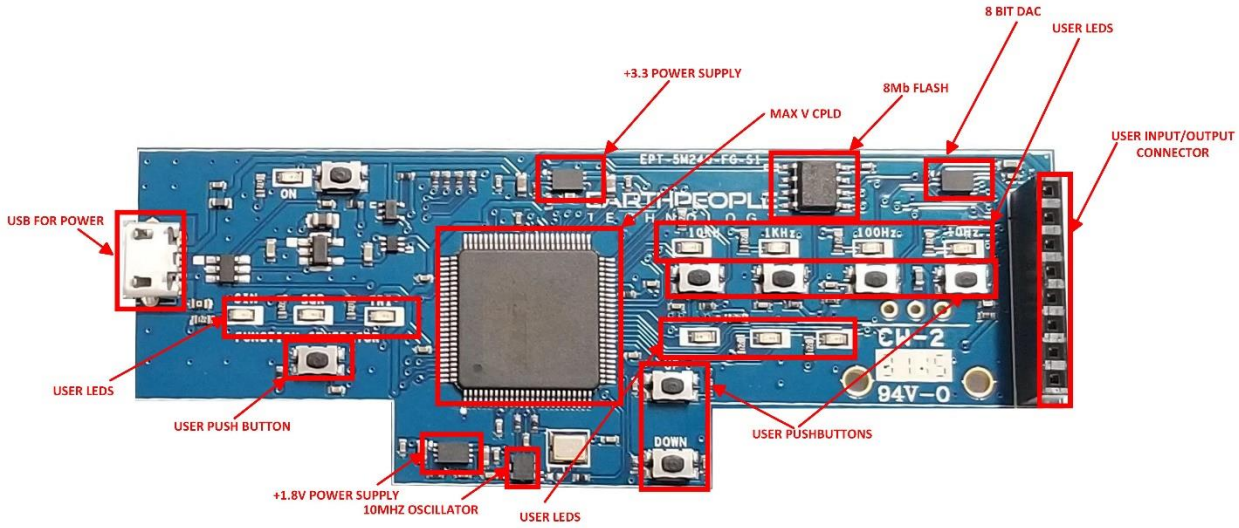
3 BeeProLogic Description

3.1 Functional Block Diagram



3.2 BeeProLogic Specifications

- MAX V 5M240ZT100 CPLD From Intel/ 240 Logic Elements;
- 8 Mbit Flash Chip
- 8 bit Single Channel DAC
- 5 Inputs/Outputs available at 8x1 connector on board
- 11 Green User configurable LEDs
- 8 User Configurable Pushbutton Switch
- Power: Standard USB (+5V @ 2Amp) Using Micro-B connector
- 10MHz Oscillator
- Standard Programming Connector fits any JTAG Blaster



3.3 CPLD

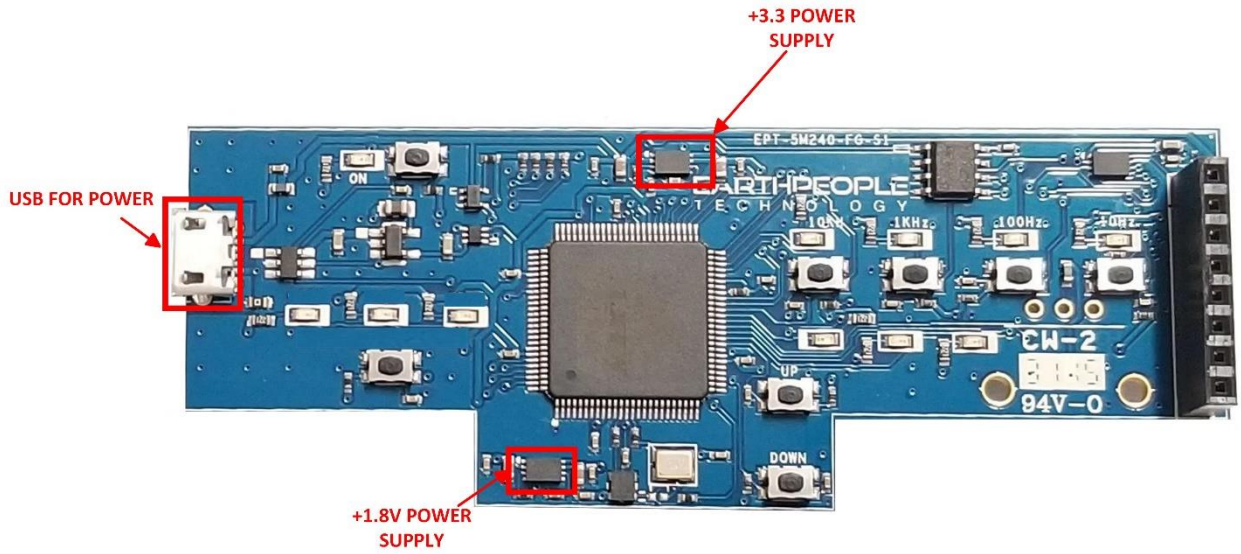
The BeeProLogic includes the Intel 5M240ZT100 CPLD. It is an PQFP 100 pin package. This CPLD incorporates 240 Logic Cells in the chip. The chip is programmed via JTAG and is compatible with the standard JTAG Blaster. Connector J7 on the rear of the board is the JTAG Connector. It is a JST 6x1 connector and the BeeProLogic includes an adapter to convert the 6x1 pins into the standard 5x2 JTAG Blaster connector.

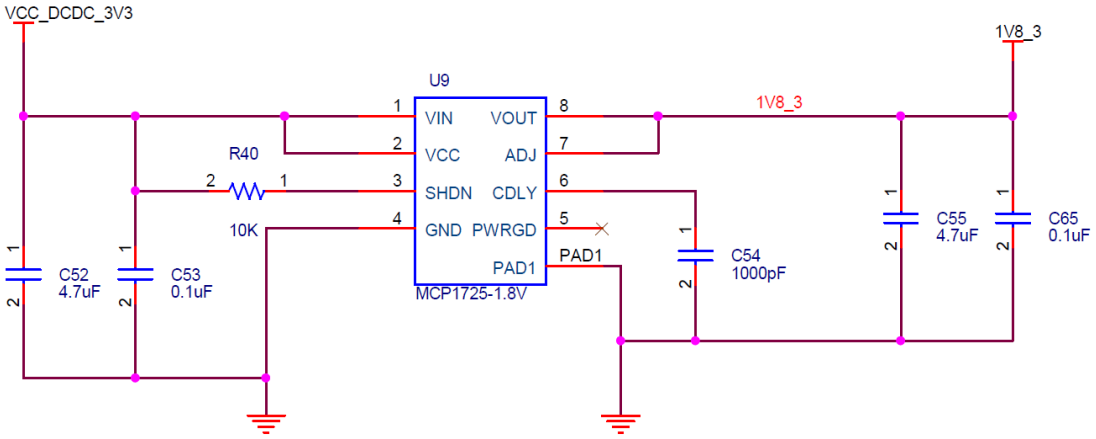
3.4 Power Supply

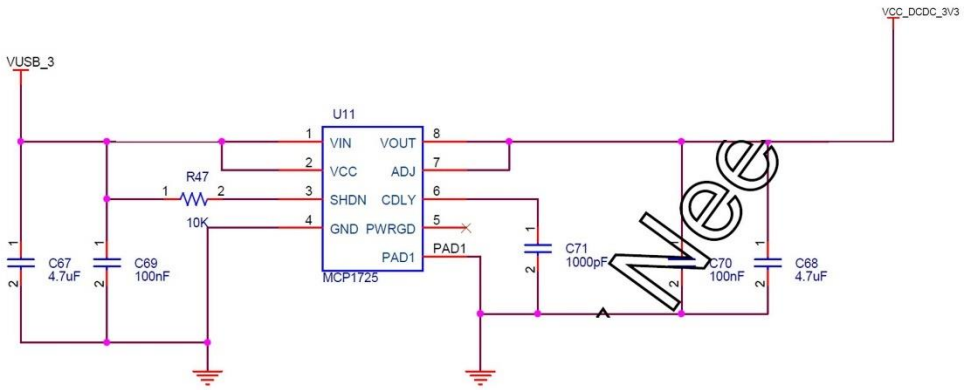
The BeeProLogic is designed to be operated from the following power source:

- Standard USB Micro-B cable

The board has a Molex style USB Micro-B connector on board. Connect the USB cable either to a USB Port on a PC or a standard charger (+5V @ 2A). This provides power for two high-efficiency serial regulators (LDO) on-board. A +3.3V LDO provides power for the CPLD and other devices on the board. The +1.8V LDO provides power for the core of the CPLD.

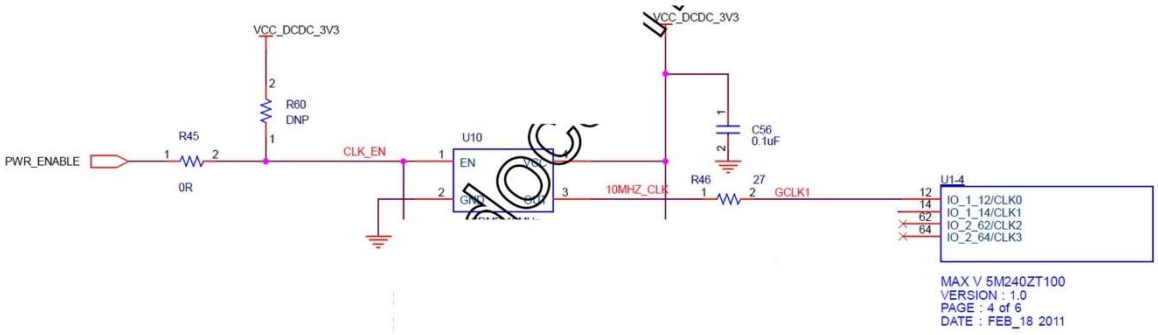

+3.3 POWER SUPPLY
USB FOR POWER
+1.8V POWER SUPPLY
ON
UP
DOWN
EARTHPEOPLE
TECHNOLOGY
94V-0


VCC_DCDC_3V3
1V8_3
U9
VIN
VOUT
VCC
ADJ
SHDN
CDLY
GND
PWRGD
PAD1
MCP1725-1.8V
R40
10K
C52
4.7uF
C53
0.1uF
C54
1000pF
C55
4.7uF
C65
0.1uF



3.5 Clock Domains

There is one clock domain external to the MAX V CPLD, 10 MHz. The 10 MHz oscillator is a +3.3VDC device that provides a high speed clock to the CPLD. It is a CMOS device that provides a stable 10 MHz at ±50 ppm. This clock can be used directly in the user code internal to the CPLD. It is intended that this clock will drive the logic of the user code.



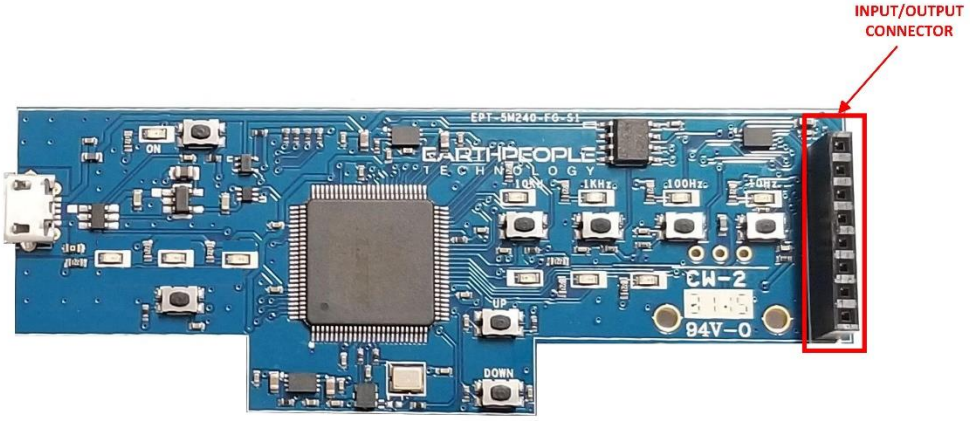
The CPLD is responsible for asserting the PWR_ENABLE signal to the oscillator. The signal must assert high for the oscillator to operate. The PWR_ENABLE signal is available on the following CPLD pin:

Component	Net Name	Pin on CPLD	Signal in EPT Project Pinout
U10	PWR_ENABLE	99	10MHz_ENABLE

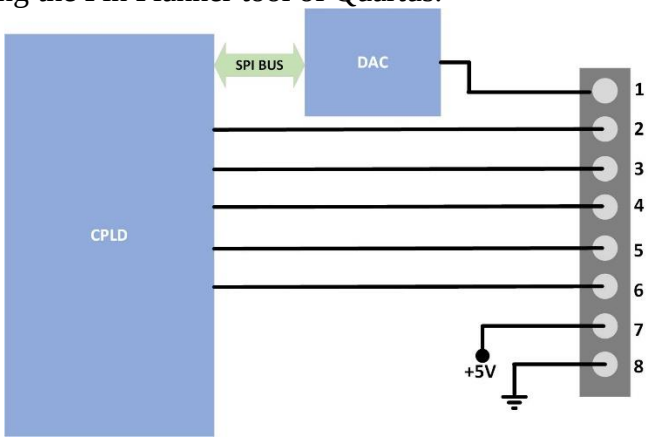
If the oscillator is not needed, de-assert the PWR_ENABLE signal.

3.6 Digital I/Os

The BeeProLogic has one 8 pin header that provide 5 digital Inputs and Outputs. All of the I/O's are +3.3 VDC only. All I/O's connect directly from the CPLD.



The digital pins of the I/O Connector are directly accessible from pins on the CPLD. The user can access these pins by defining them in user HDL code. Then ensure that the pins are connected properly using the Pin Planner tool of Quartus.

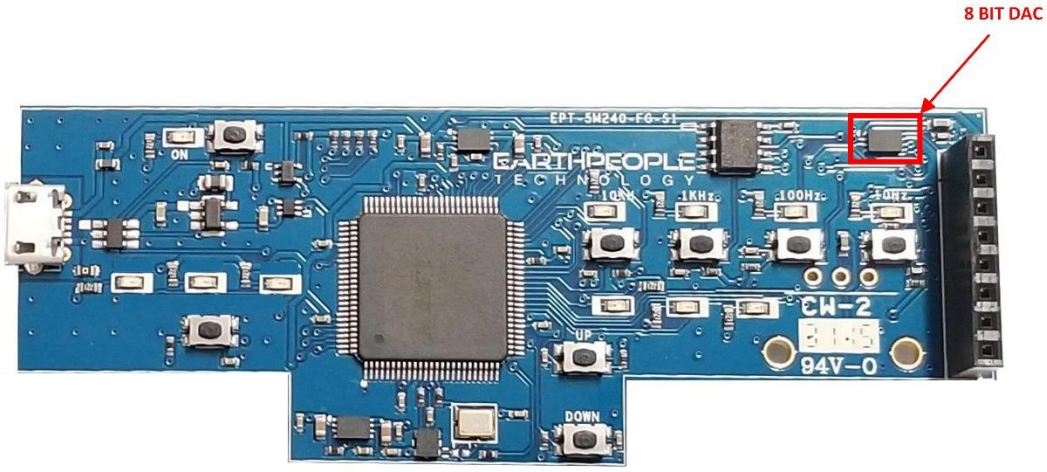


Component	Net Name	Pin on CPLD	Signal in EPT Project Pinout
J7	FG_GPIO_1	41	USER_GP_1
J7	FG_GPIO_2	42	USER_GP_2
J7	FG_GPIO_3	47	USER_GP_3
J7	FG_GPIO_4	37	USER_GP_4
J7	FG_GPIO_5	38	USER_GP_5

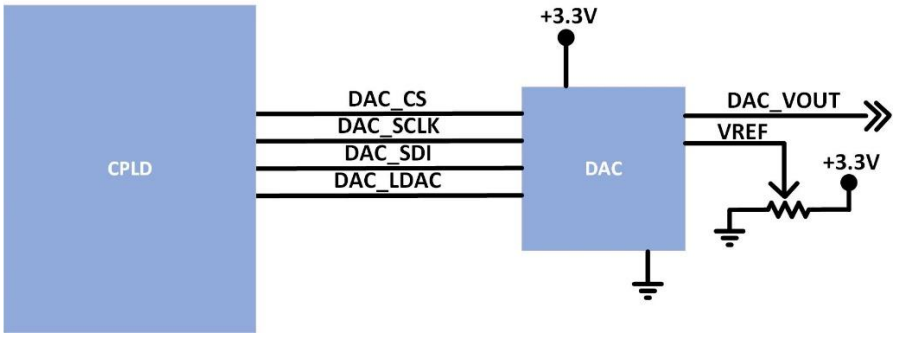
All CPLD I/O are intrinsically bidirectional. Based on the I/O direction specified in the configuration file, each I/O can be configured as input-only, output-only, or bidirectional. While the output buffer of a bidirectional I/O has always included a dynamic output enable signal, MAX V CPLDs also include an input buffer enable/disable capability. When an input buffer is dynamically disabled, the buffer presents its last known state to the CPLD core fabric, so that no inputs inside the CPLD are left floating.

3.7 Analog I/O's

The BeeProLogic includes a single channel 8 bit DAC. The user can write any 8 bit digital value to the DAC and the associated analog voltage will appear on the output.



The DAC is directly connected to the CPLD. The communication path between CPLD and the 8 bit DAC is via SPI bus.



Component	Net Name	Pin on CPLD	Signal in EPT Project Pinout
U22	DAC_CS	61	DAC_CS
U22	DAC_SCLK	58	DAC_SCLK
U22	DAC_SDI	57	DAC_SDI
U22	DAC_LDAC	56	DAC_LDAC

A 20KΩ potentiometer is connected to the VREF pin. This varies the voltage applied to the VREF pin and provides amplitude control for the analog output voltage. The MCP4901 DAC

chip will convert any eight bit digital word into an analog voltage. The settling time for the chip is 4.5µseconds. The output analog voltage can reproduce a sine wave of approximately 800Hz.

The user code will include the library that provides the SPI communications. The MCP4901 device is designed to interface directly with the Serial Peripheral Interface (SPI) port, which is available on many microcontrollers and supports Mode 0,0 and Mode 1,1. Commands and data are sent to the device via the SDI pin, with data being clocked-in on the rising edge of SCK. The CS pin must be held low for the duration of a write command. The write command consists of 16 bits and is used to configure the DAC's control and data latches.

There are no registers to set and no registers to read from in the MCP4901. The chip has a maximum SPI clock frequency of:

- 5MHz

It requires to eight bit bytes to be written to the chip in order to perform a Digital to Analog conversion.

3.8 Register Set

There is only one register accessible on the MCP4901, Write Command Register.

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	BUF	GA	SHDN	D7	D6	D5	D4	D3	D2	D1	D0	X	X	X	X

Where:

bit 15
- 0 = Write to DAC register
- 1 = Ignore this command

bit 14 **BUF**: VREF Input Buffer Control bit
- 1 = Buffered
- 0 = Unbuffered

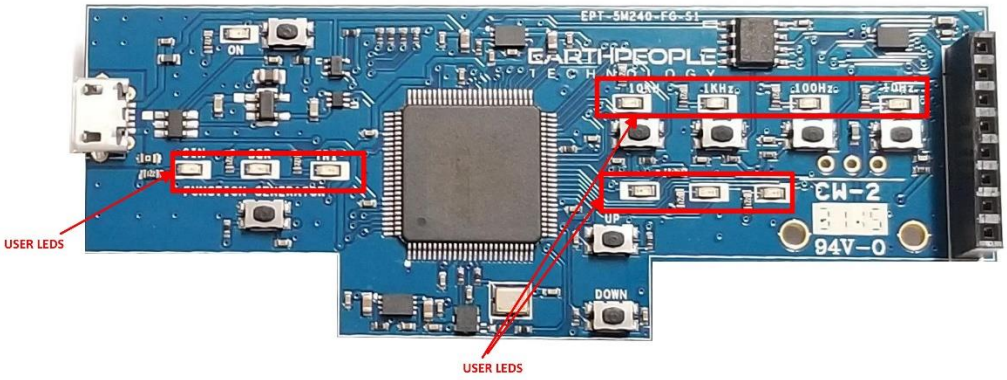
bit 13 **GA**: Output Gain Selection bit
- 1 = 1x (VOUT = VREF * D/256)
- 0 = 2x (VOUT = 2 * VREF * D/256)

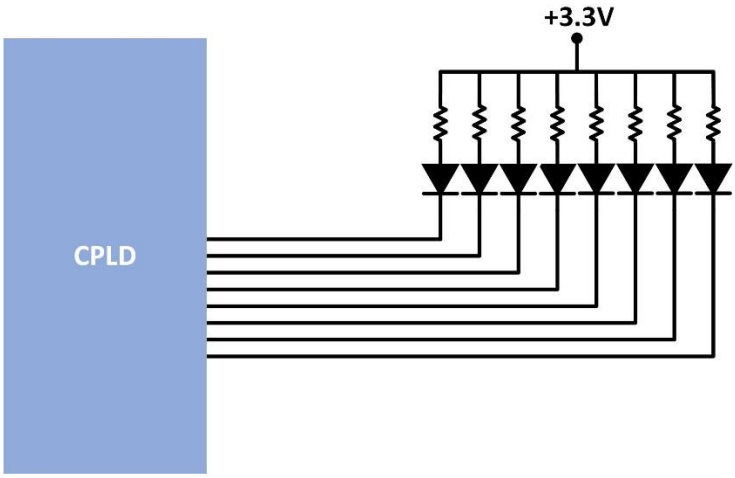
bit 12 **SHDN**: Output Shutdown Control bit
- 1 = Active mode operation. VOUT is available.
- 0 = Shutdown the device. Analog output is not available. VOUT pin is connected to 500 kΩ (typical).

bit 11-0 D11:D0: DAC Input Data bits. Bit x is ignored.

3.9 LEDs

The BeeProLogic includes eleven user LEDs. The LEDs are directly sinked from the CPLD with the cathode connected to +3.3V.





Component	Net Name	Pin on CPLD	Signal in EPT Project Pinout
D5	USER_LED_1	48	LED[0]
D6	USER_LED_2	40	LED[1]
D7	USER_LED_3	53	LED[2]
D8	USER_LED_4	54	LED[3]
D9	USER_LED_5	39	LED[4]

D10	USER_LED_6	36	LED[5]
D11	USER_LED_7	35	LED[6]
D12	USER_LED_8	69	LED[7]
D13	USER_LED_9	95	LED[8]
D14	USER_LED_10	96	LED[9]
D15	USER_LED_11	97	LED[10]

They use the +3.3V I/O's along with a 220 Ohm series resistor for each LED. This provides the following current through the LEDS.

$$I_{LED} = \frac{V_O - V_F}{R}$$

$$I_{LED} = \frac{3.3V\ - 2.0V}{220}$$

$$I_{LED} = 5.9mA$$

The code to drive the LEDs is either zero (1'b0) or floating (1'bz). First, declare the LED as an output. In the example below, the vector LED is set to 'reg' because it is driven in an always block.

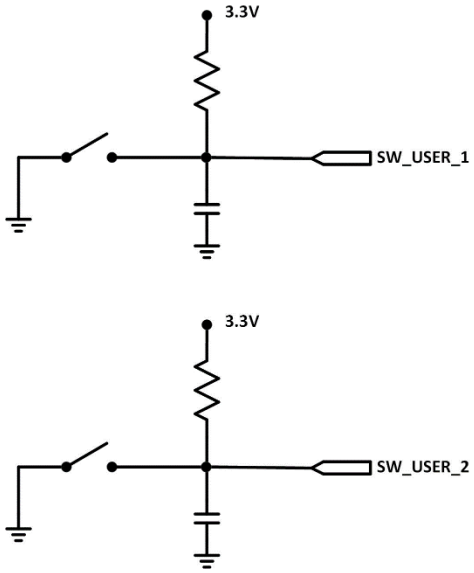
```
//*****************************************************************************
//* Module Declaration
//*****************************************************************************

module EPT_5M240_AP_S2_Top (

    input wire                  CLK_10MHZ,
    input wire                  RST,

    input wire                  SW_USER_1,
    input wire                  SW_USER_2,
    input wire                  SW_USER_3,
    input wire                  SW_USER_4,
    input wire                  SW_USER_5,
    input wire                  SW_USER_6,
    input wire                  SW_USER_9,

    input wire                  PWR_ENABLE,
    output wire                 PWR_KILL,

    output reg  [10:0]          LED
    );
```

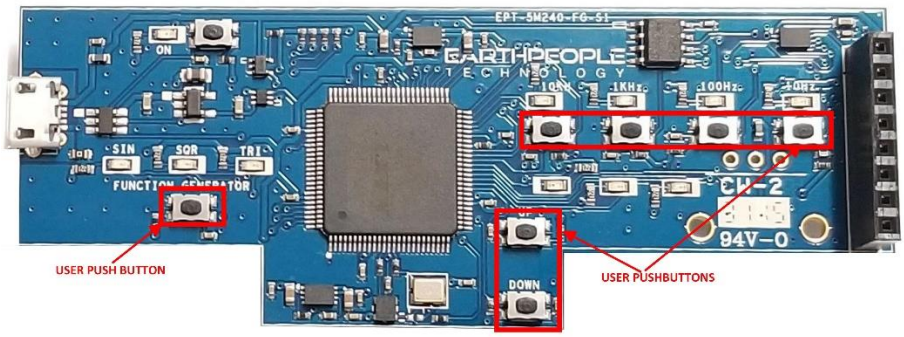
To turn the selected LED on, set the signal equal to 1'b0. This will apply a ground to the cathode side of the LED and allow current to flow through the circuit turning the LED on. To turn the selected LED off, set the signal equal to 1'bz. This will float the cathode side of the LED and no current will flow through the LED.

```
//-------------------------------------------------
// Set the LED outputs
//-------------------------------------------------
always @(posedge CLK_10MHZ or negedge RST)
begin
  if(!RST)
      LED <= 8'hz;
  else
  begin
    if(state[LOAD_LEDS])
    begin
      if ( led_reg[0] )
          LED[0] = 1'b0;
      else
          LED[0] = 1'bz;

      if ( led_reg[1] )
          LED[1] = 1'b0;
      else
          LED[1] = 1'bz;

      if ( led_reg[2] )
          LED[2] = 1'b0;
      else
          LED[2] = 1'bz;
```

3.10 Pushbutton Switches

The BeeProLogic includes eight push button switches. All are momentary contact switches. They include a 1uF cap to ground to debounce both switches. The code to use these switches is simple, just declare an input on the selected switch pin. Then use code to activate the a function with the input signal goes to low.



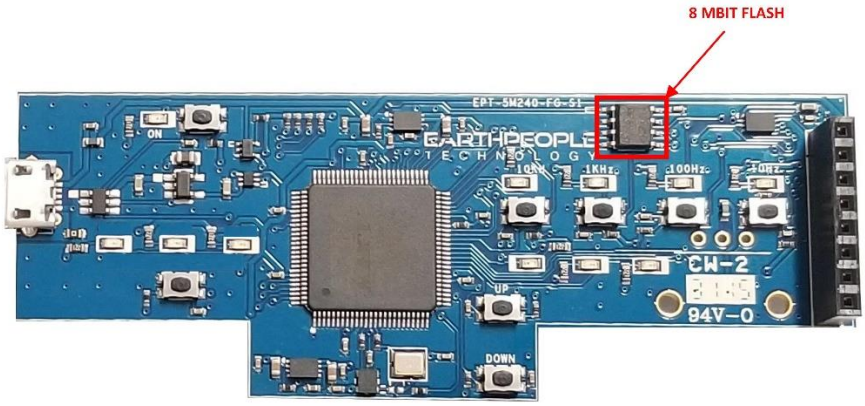


Component	Net Name	Pin on CPLD	Signal in EPT Project Pinout
SW1	SW_USER_1	53	SW_USER_1
SW2	SW_USER_2	51	SW_USER_2
SW3	SW_USER_3	50	SW_USER_3
SW4	SW_USER_4	49	SW_USER_4
SW5	SW_USER_5	34	SW_USER_5
SW6	SW_USER_6	33	SW_USER_6

SW9	SW_USER_7	98	SW_USER_7

3.11 8 MBit Flash

The BeeProLogic includes an 8 Mbit Flash.

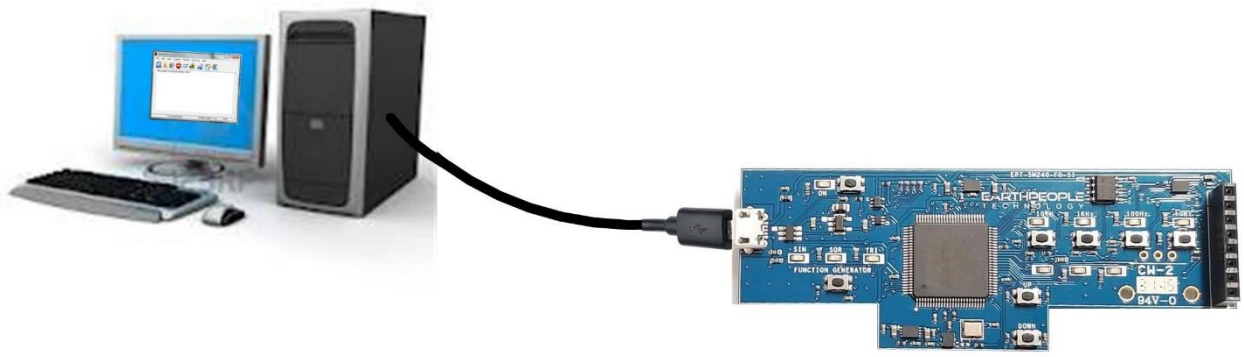


3.12 Power Input

The BeeProLogic is designed to be operated from the following power source:

- Standard USB Micro-B cable

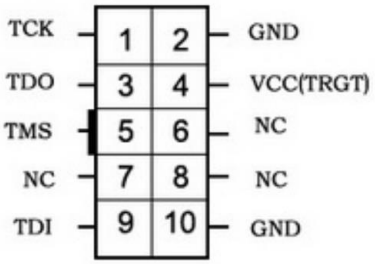
The board has a Molex style USB Micro-B connector on board. Connect the USB cable either to a USB Port on a PC or a standard charger (+5V @ 2A). This provides power for a high-efficiency switching regulator on-board to provide +3.3 VDC. The power supply provides reliable power under dynamic loads and high frequency switching internal to the CPLD.



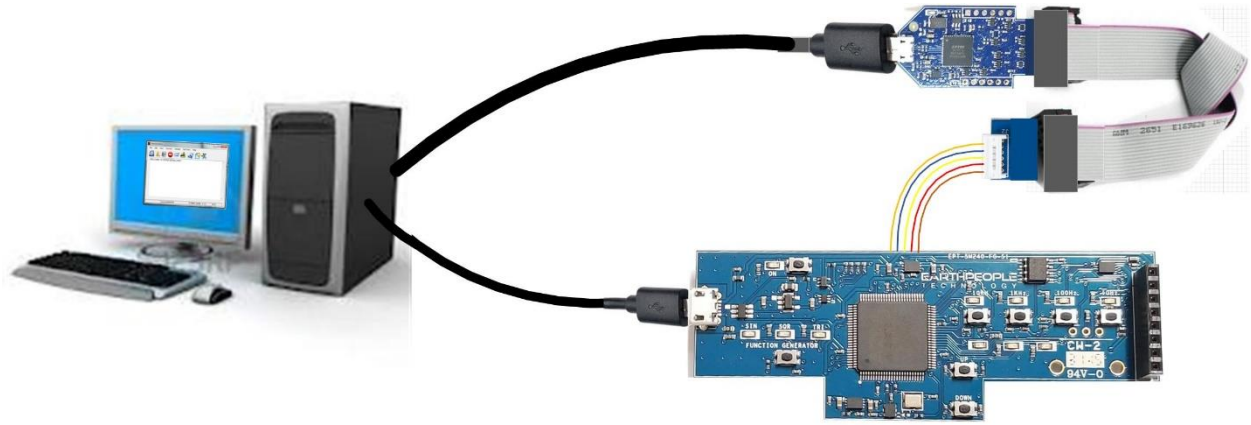
3.13 JTAG Interface

Connector J7 on the rear of the board is the JTAG Connector. It is a JST 6x1 connector and the BeeProLogic includes an adapter to convert the 6x1 pins into the standard 5x2 JTAG Blaster connector.

This connector uses the standard Altera Blaster connector pinout.



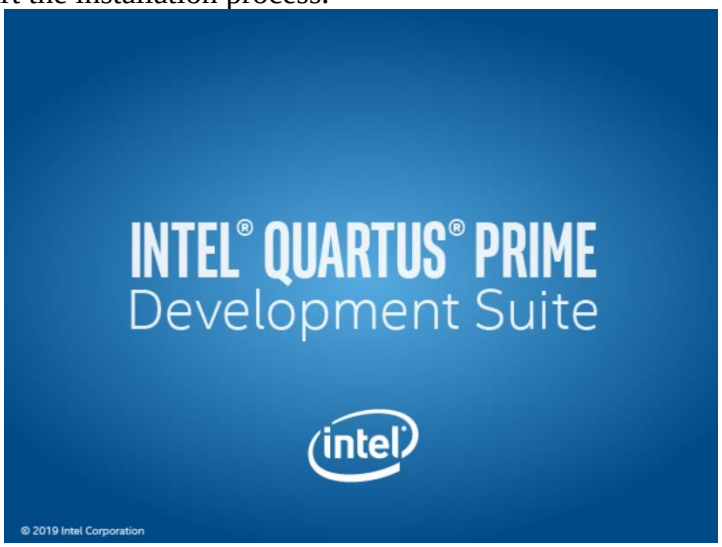
The VCC(TRGT) is set to +3.3V on the BeeProLogic. There are no jumper settings to make in order to program the MAX V CPLD. Just connect a compatible Blaster to the connector and the PC, then use the Quartus software to program the CPLD.

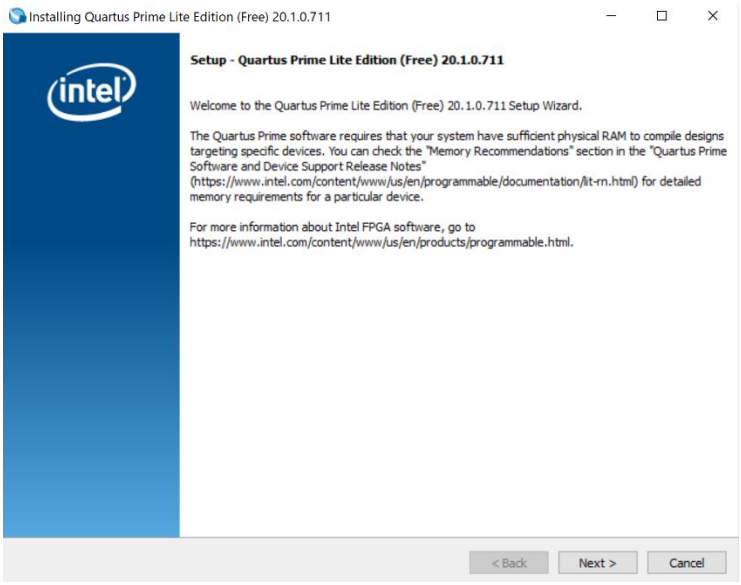


4 Installing Quartus

You can download the Quartus Prime Lite by following the directions in the Section Downloading Quartus.

If you don't need to download Quartus, double click on the QuartusLiteSetup-xxx.xxx.xxx-windows .exe (the xxx is the build number of the file, it is subject to change). The Quartus Prime Web Edition will start the installation process.





When the install shield window pops up click “Yes” or if needed, enter the administrator password for the users PC. Click “Ok”

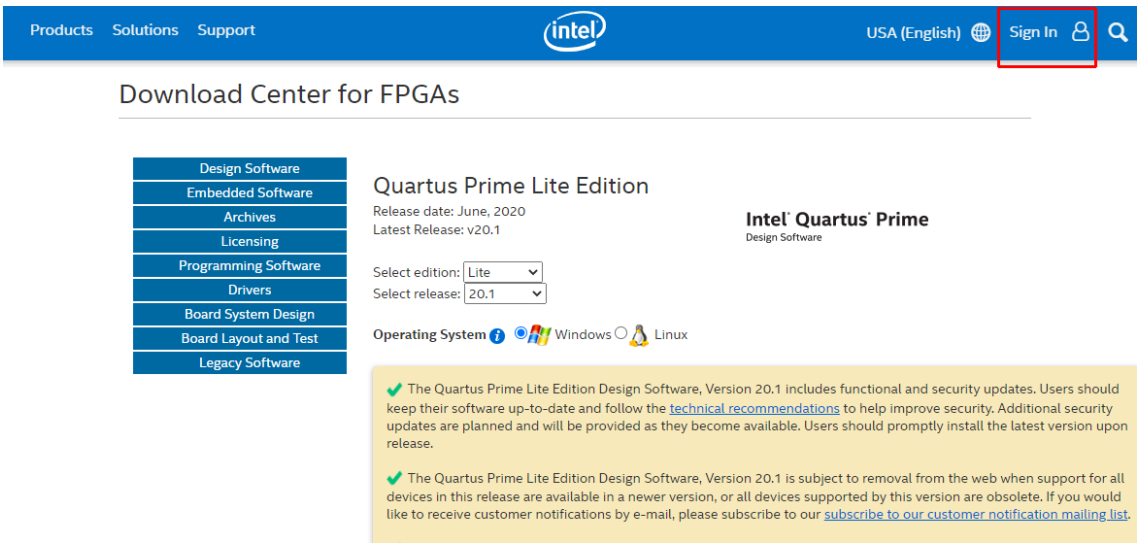
Next, skip the “Download Quartus” section. Go down to the “Quartus Installer” section to complete the Quartus installation.

4.1.1 Downloading Quartus

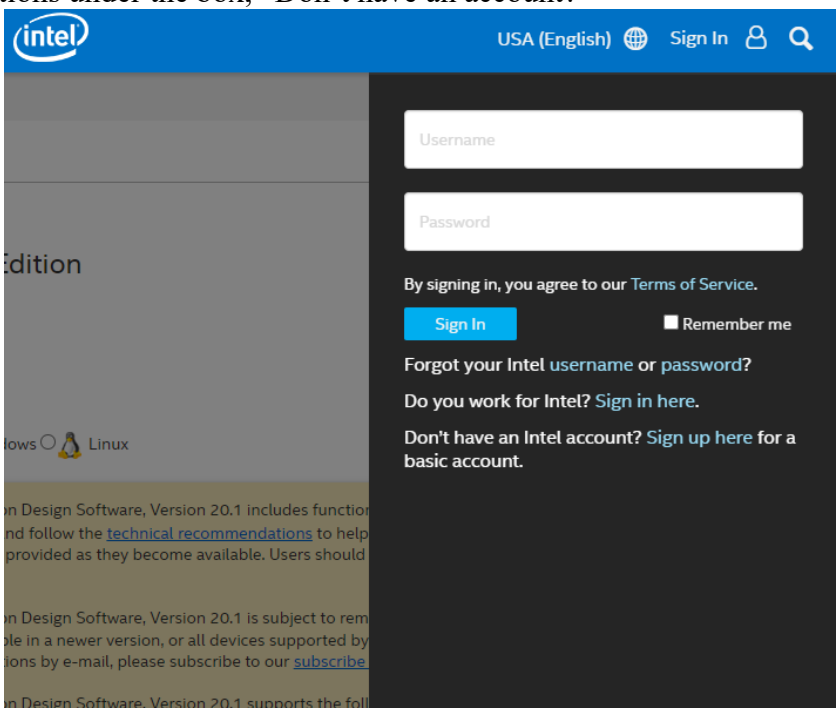
The first thing to do in order build a project in Quartus is to download and install the application. You can find the latest version of Quartus at:

Intel FPGA Quartus Prime Lite

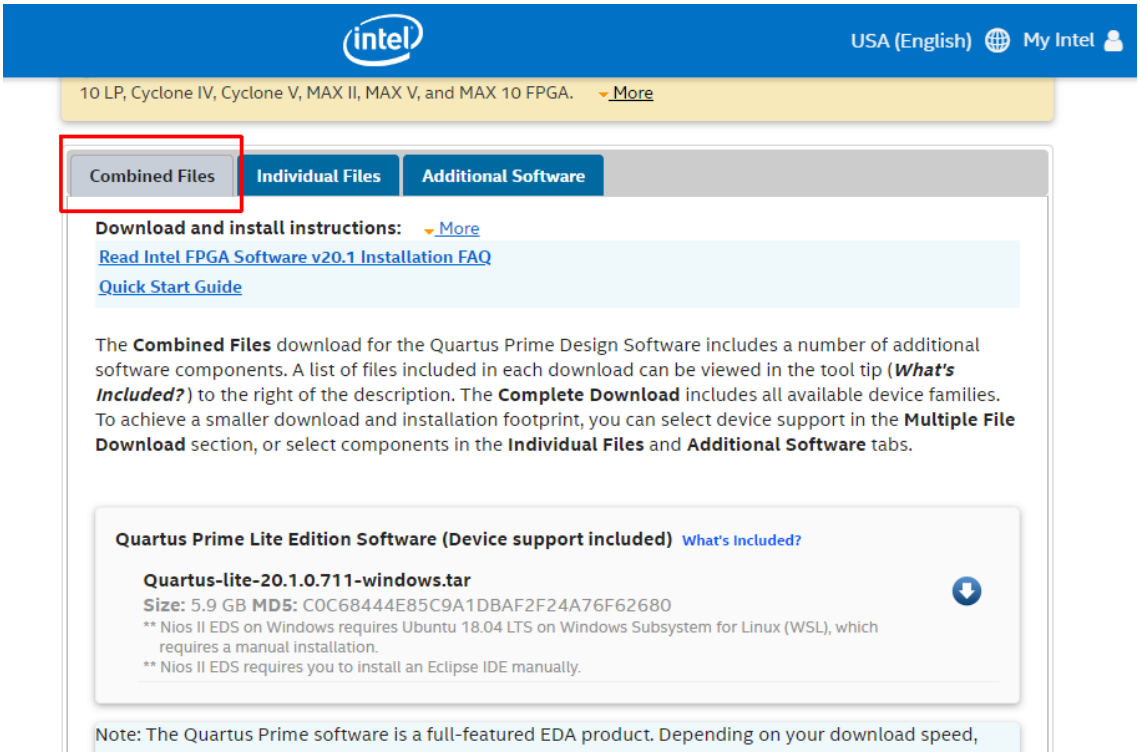
You will first need to apply for an account with Intel. Then use your login and password to access the download site. Click on the Download Windows Version.



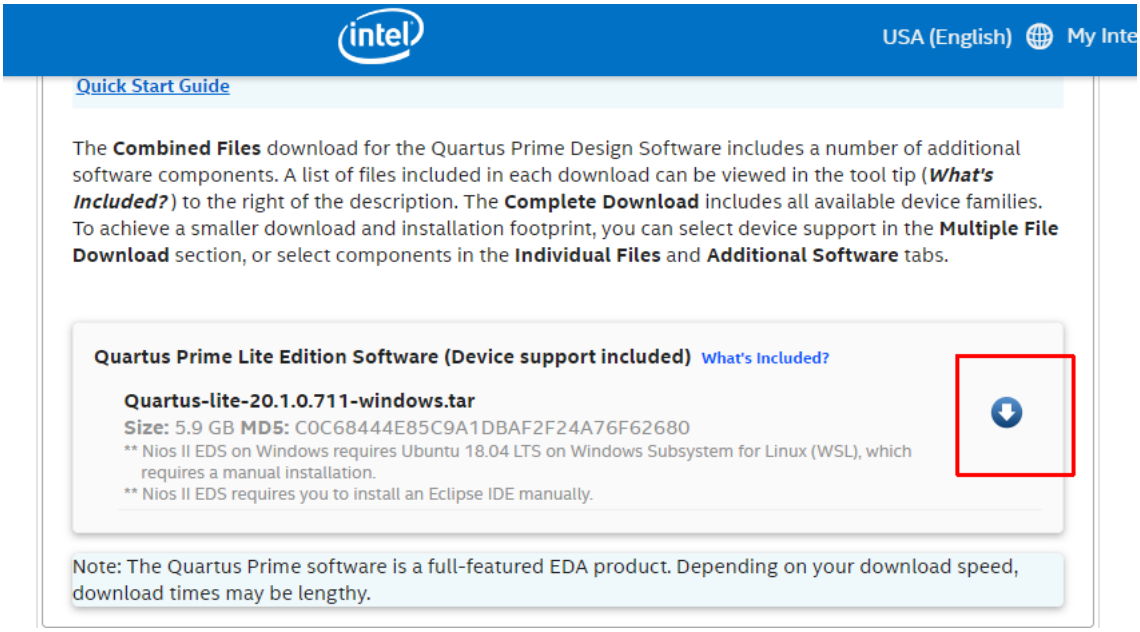
The next page will require you to sign into your "myAltera" account. If you do not have one, follow the directions under the box, "Don't have an account?"



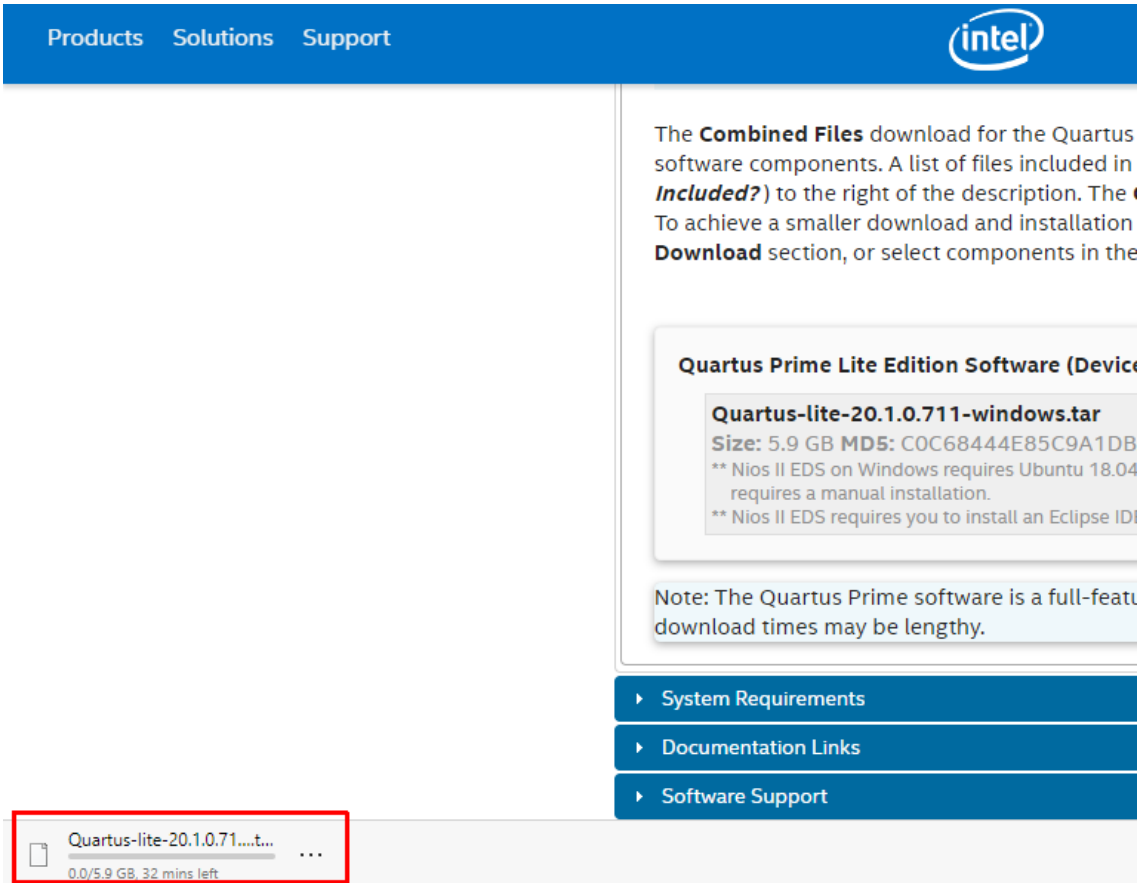
Once you have created your myAltera account, enter the User Name and Password. The next window will ask you to allow pop ups so that the file download can proceed.



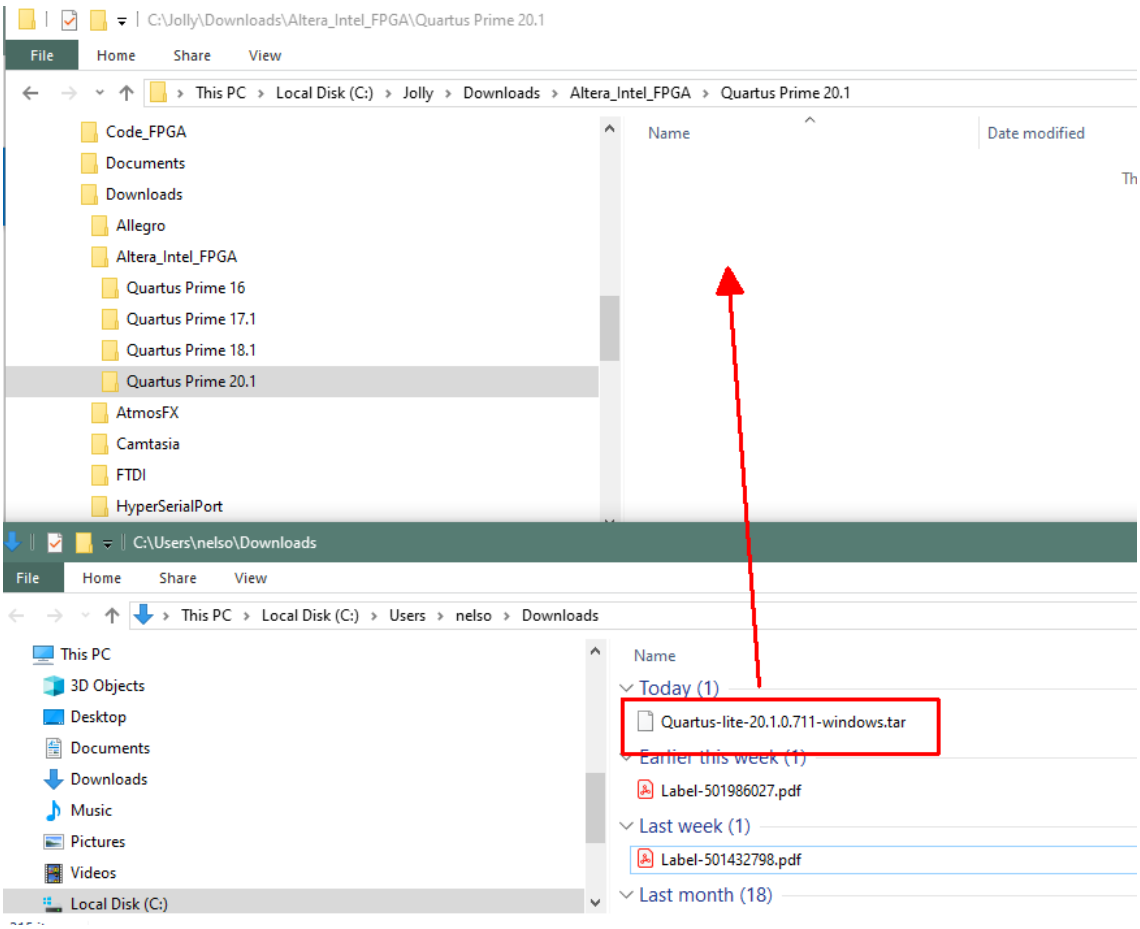
Click on the download icon.



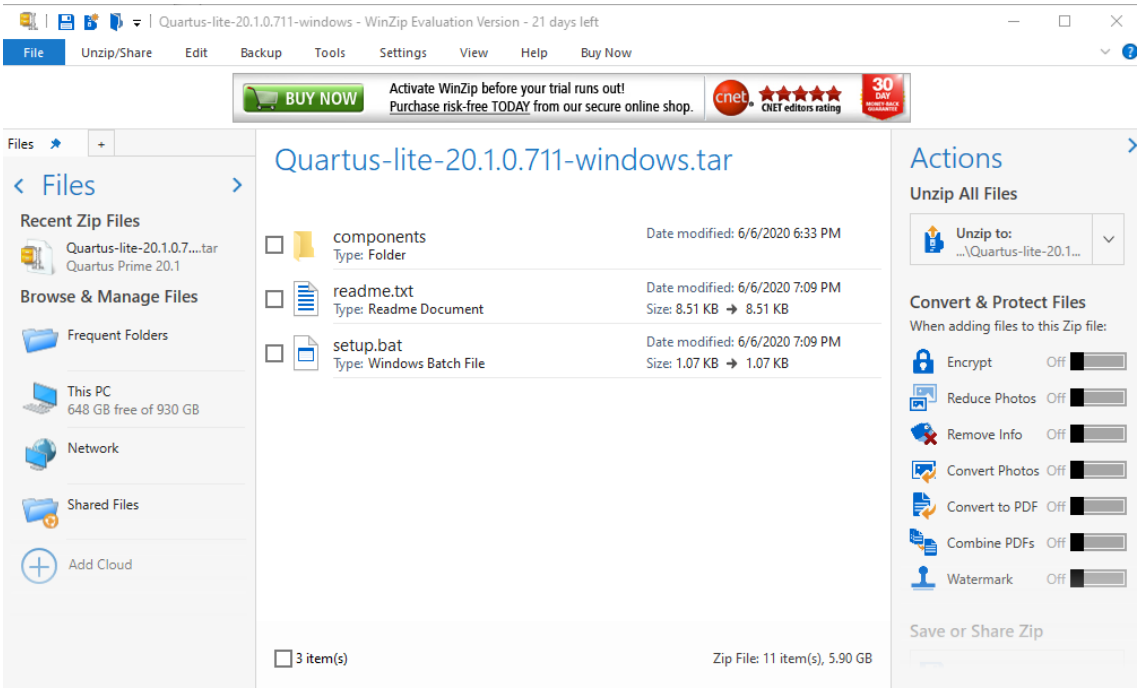
This will start the download.



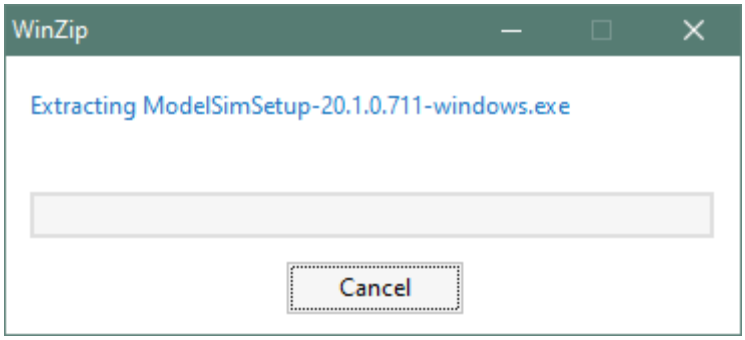
The file is 5.9 GB, so this could take a couple of hours depending on your internet connection. When download is complete, store the *.tar file in a directory on your PC.



Use a tool such as WinZip to Extract the *.tar file.

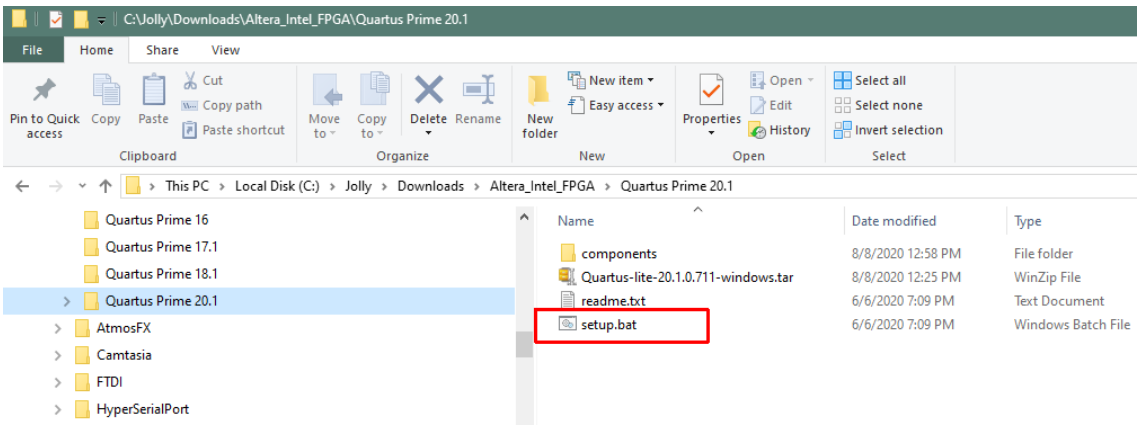


The tool will unpack all files.

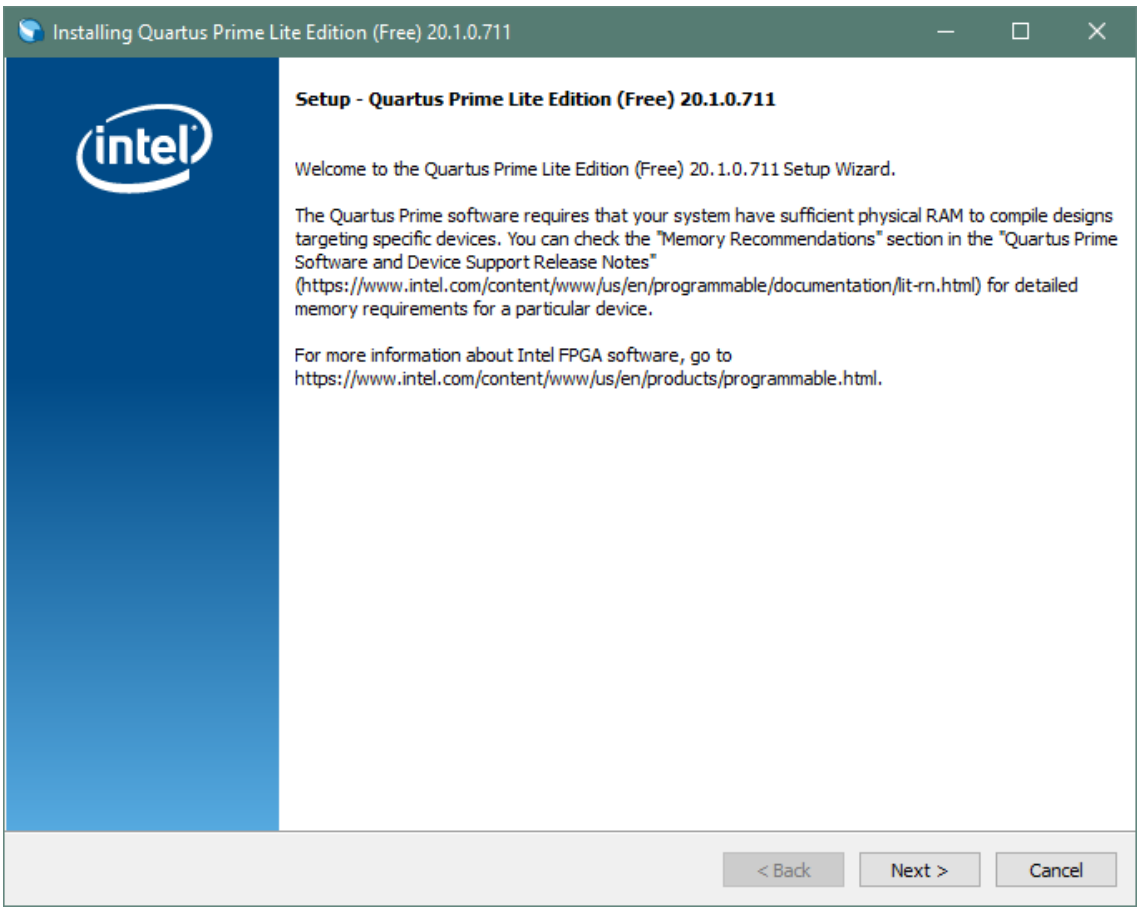


4.1.2 Quartus Installer

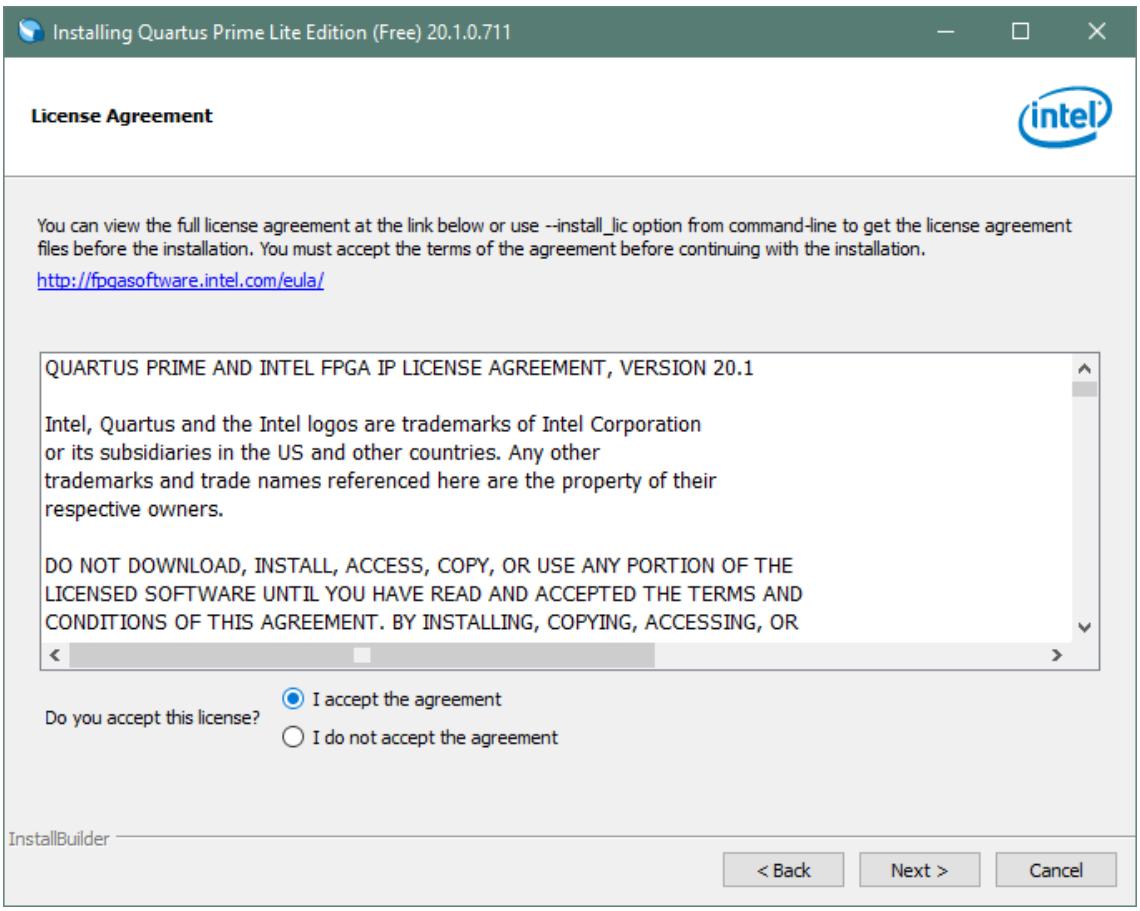
When the unpacking finishes from the previous section, double click the setup.bat file in the download folder.



Click “Next” on the Introduction Window.

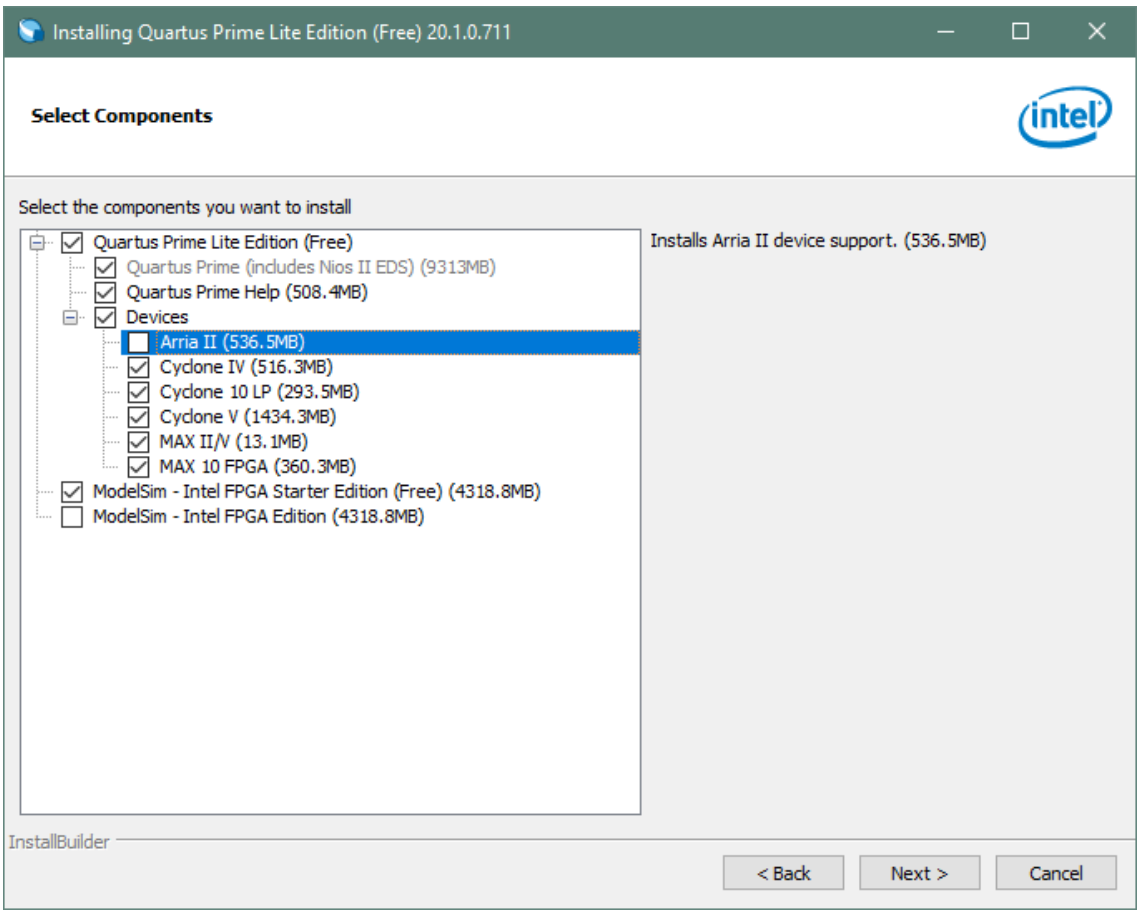


Click the checkbox to agree to the license terms. Then click “Next”.

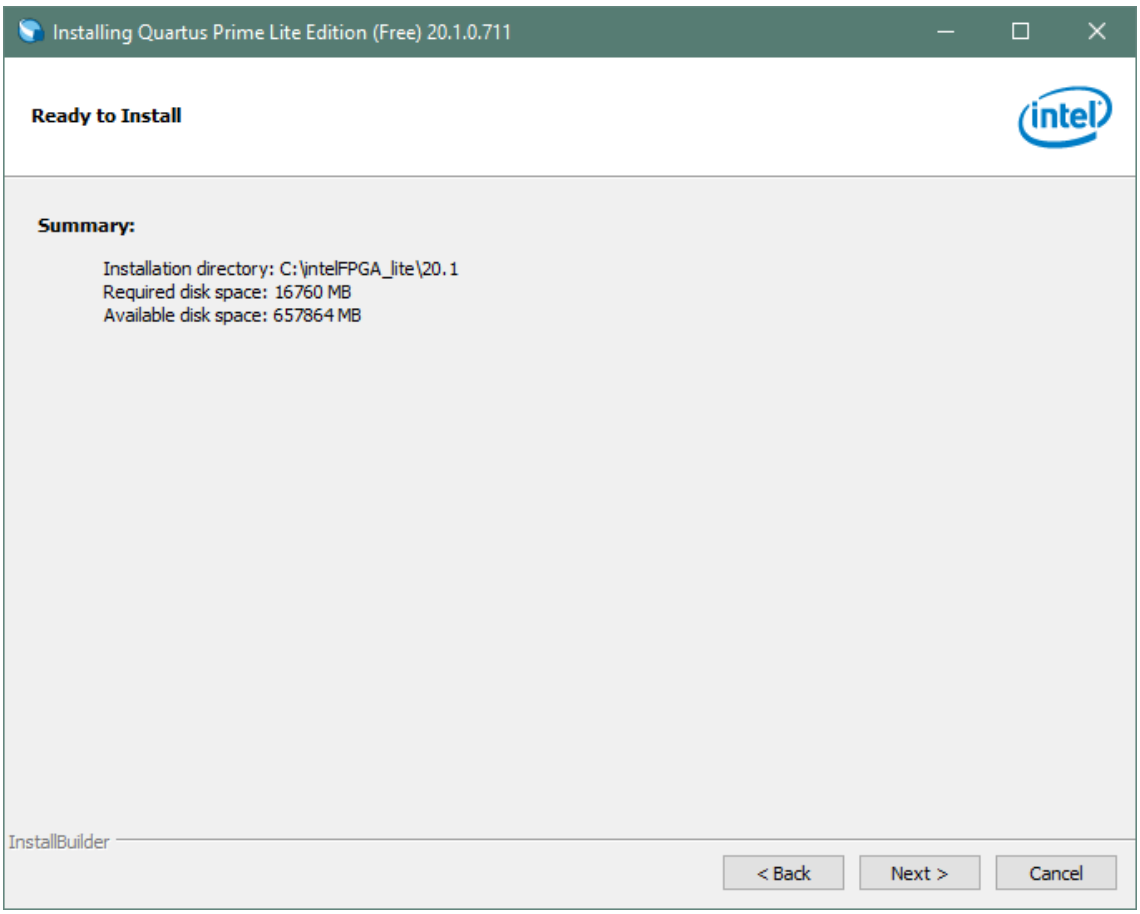


Click “Next” and accept the defaults.

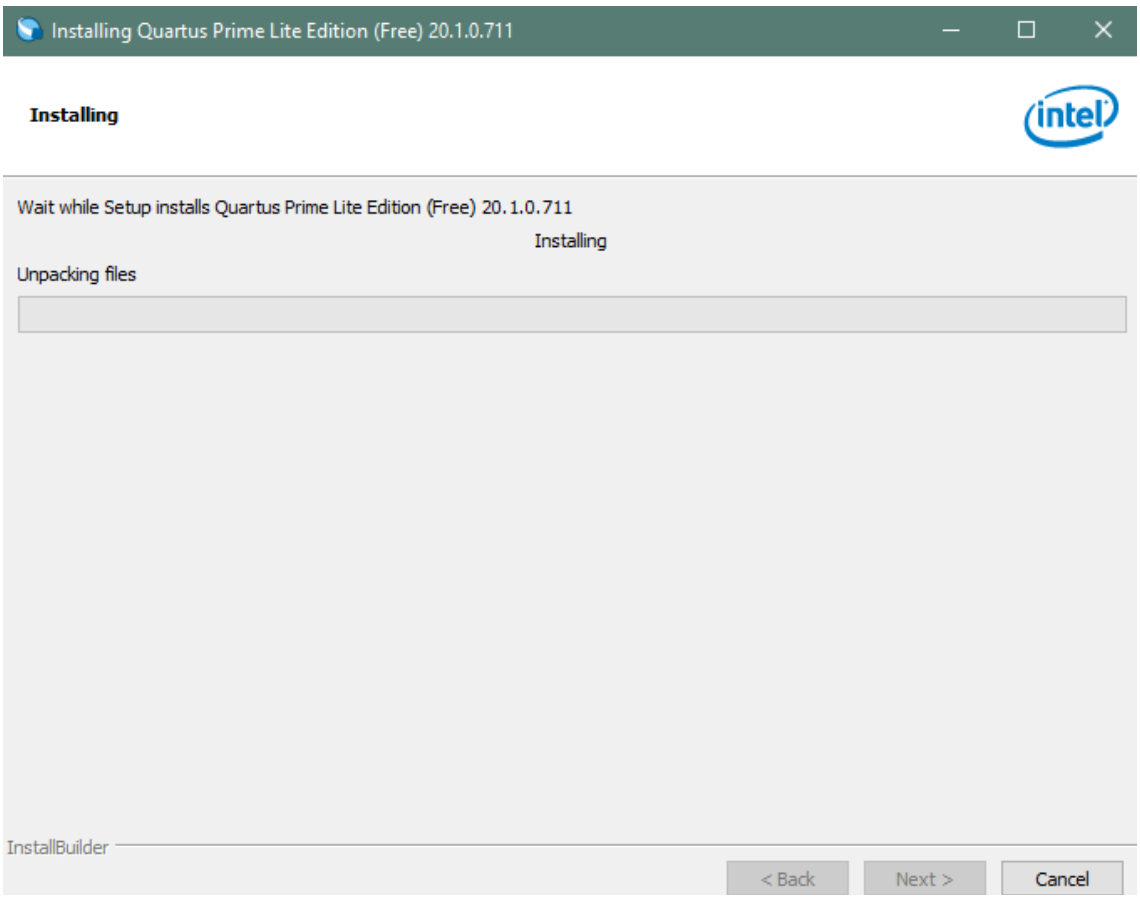
At the Select Products Window, de-select the Quartus Prime Supbscription Edition by clicking on its check box so that the box is not checked. Then click on the check box by the Quartus Prime Web Edition (Free).



Click “Next” to accept the defaults

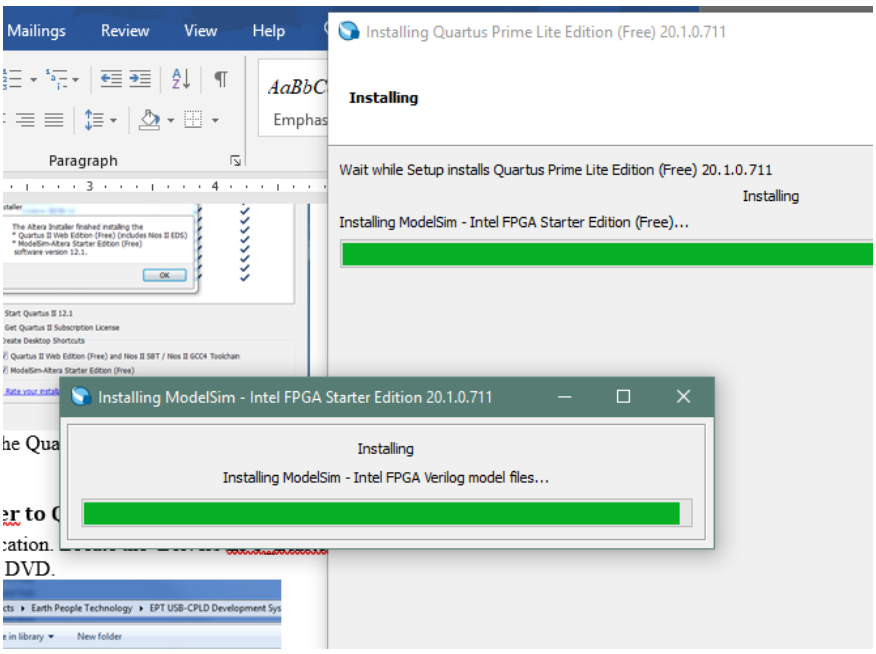


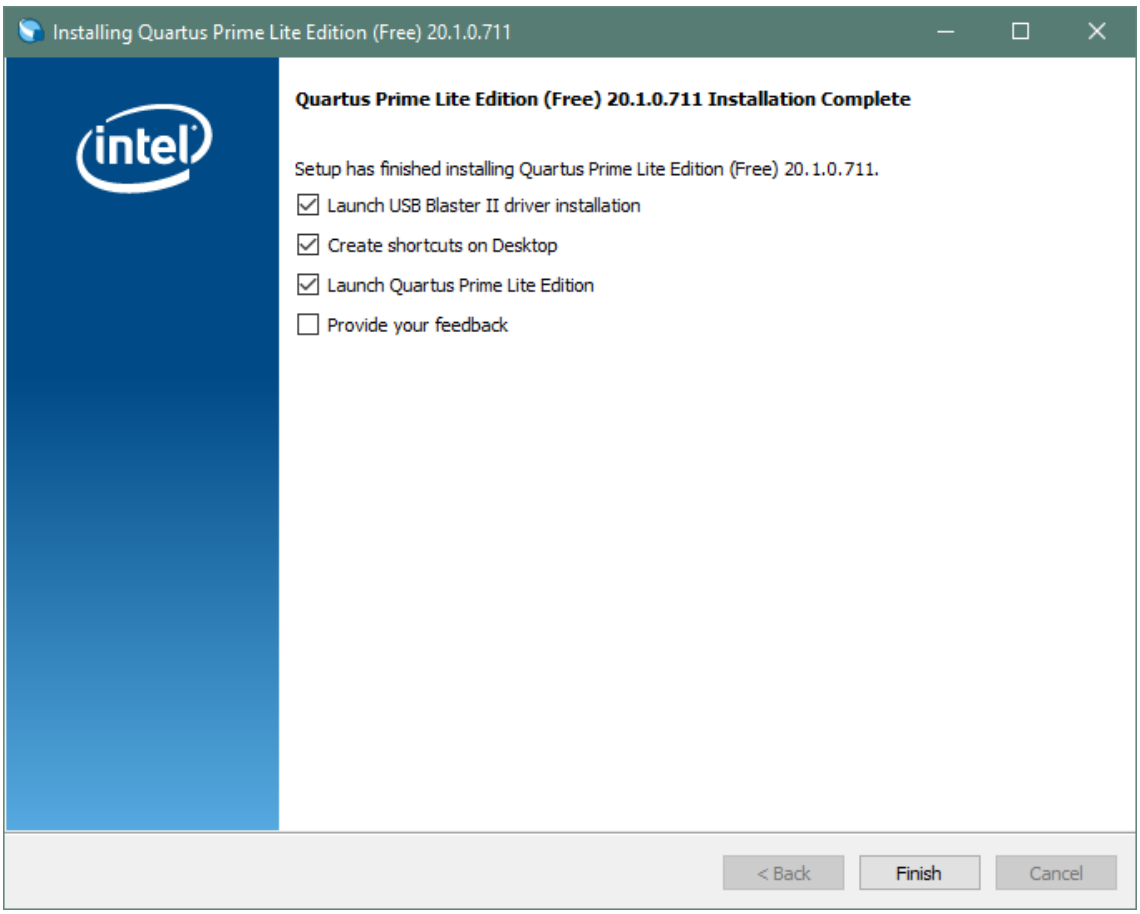
Click “Next” to accept the defaults



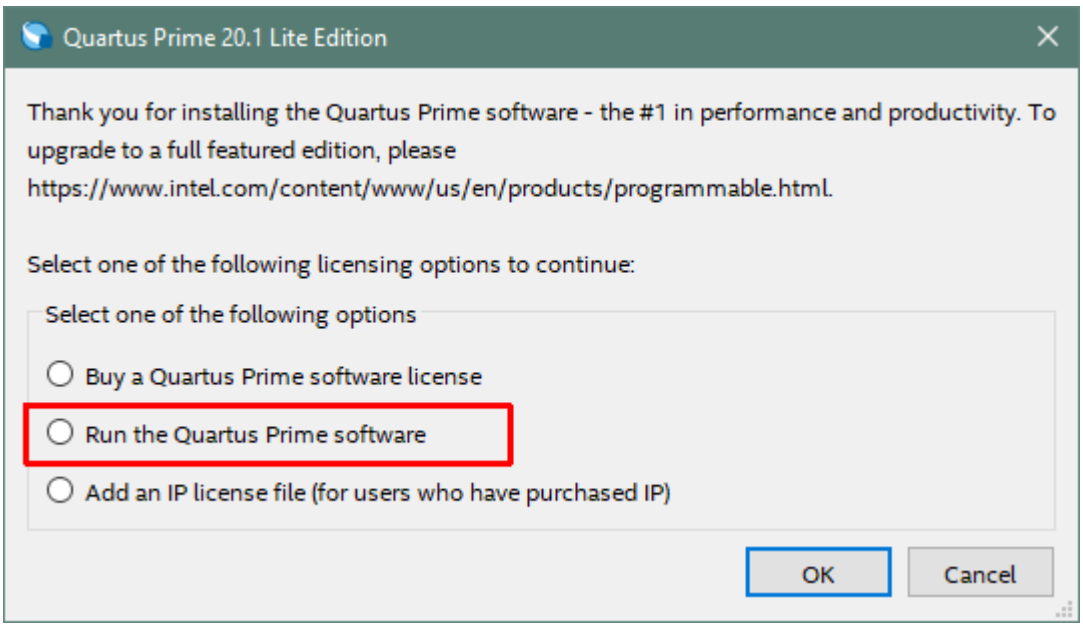
Wait for the installation to complete.

BeeProLogic Development System User Manual



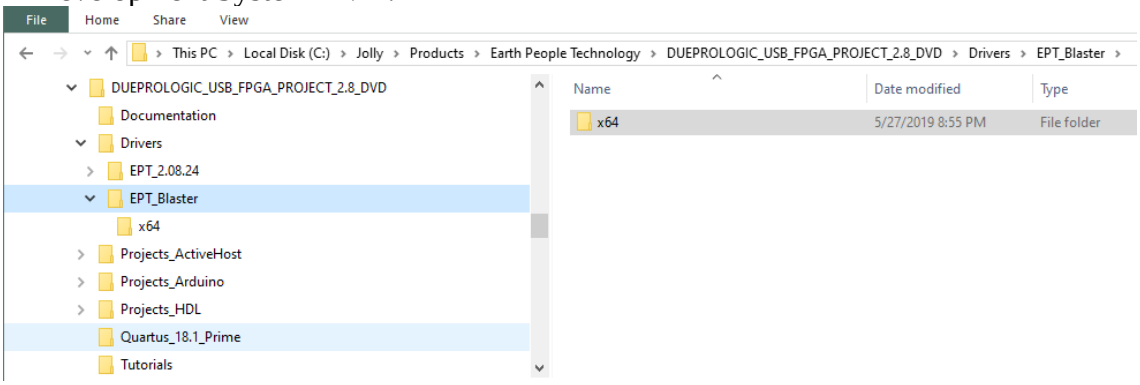


Click “Ok”, then click “Finish”. The Quartus Prime is now installed and ready to be used.



4.1.3 Adding the EPT_Blaster to Quartus Prime

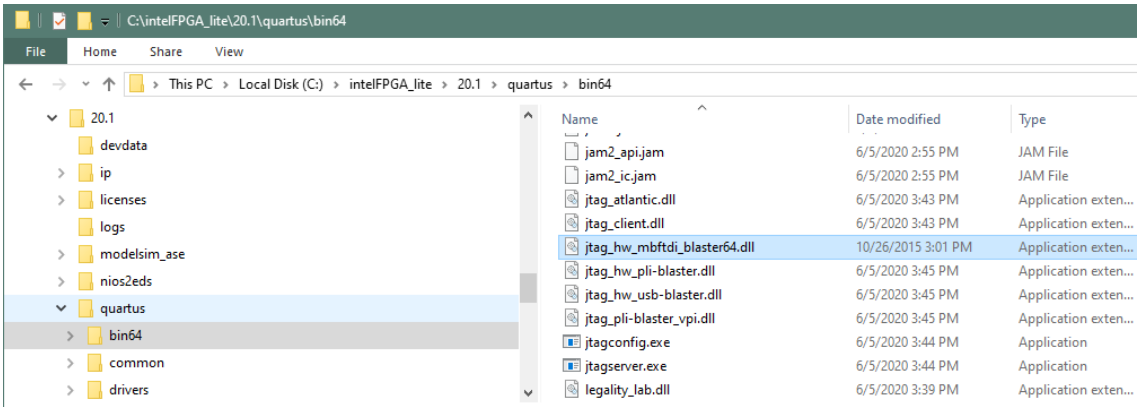
Close out the Quartus Prime application. Locate the \Drivers\EPT_Blaster folder on the EPT FPGA Development System DVD.



Follow these directions:

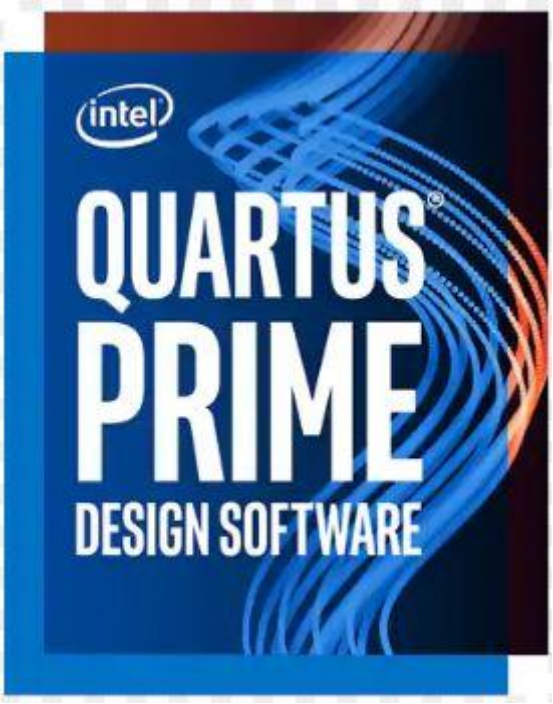
1. Open the C:\ :\..\BEEPROLOGIC_USB_CPLD_PROJECT_x.x_DVD \Drivers\EPT_Blaster\x64 folder.
2. Select the file “jtag_hw_mbftdi_blaster.dll” and copy it.
3. Browse over to C:\intelFPGA_lite\xx.x\quartus\bin64.
4. Right click in the folder and select Paste

5. Click Ok.
6. Open the Quartus Prime application.



The DLL is installed and the JTAG server should recognize it. Go to the section “Programming the FPGA” of this manual for testing of the programming. If the driver is not found in the Programmer Tool->Hardware Setup box, see the JTAG DLL Insert to Quartus Prime Troubleshooting Guide.

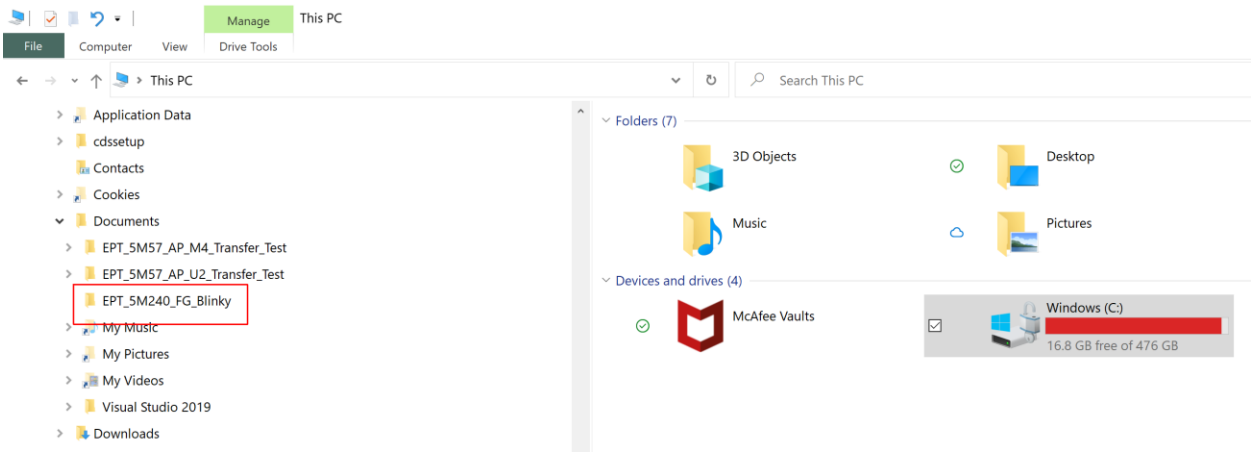
5 Compiling, Synthesizing, and Programming CPLD



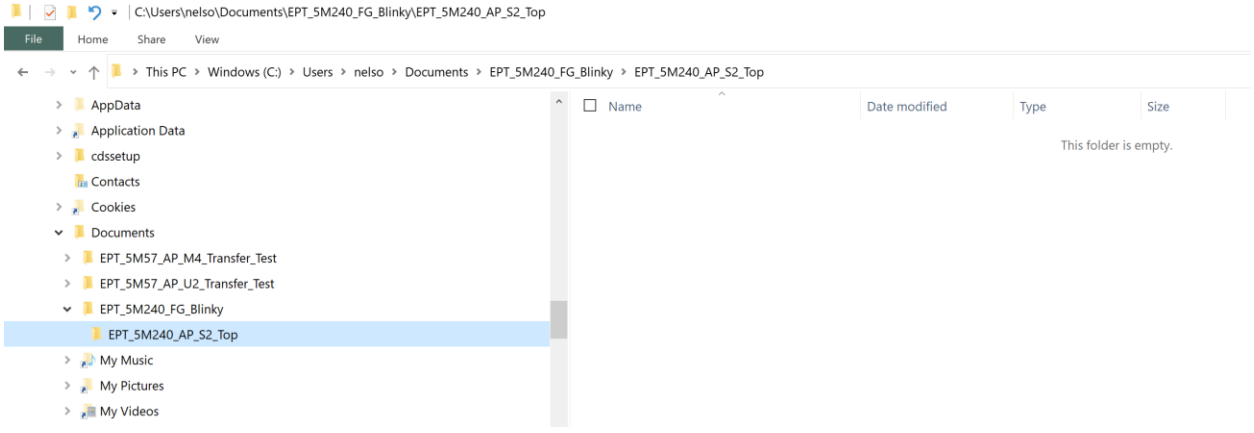
The CPLD on the BeeProLogic can be programmed with custom HDL code created by the user. Programming the CPLD requires the use of the Quartus Prime software and a standard JTAG Blaster. Once the user HDL code is written according to the syntax rules of the language (Verilog and VHDL) it can be compiled and synthesized using the Quartus Prime software. This manual will not focus on HDL coding or proper coding techniques, instead it will use the example code to compile, synthesize and program the CPLD.

5.1 Setting up the Project and Compiling

Once the HDL code (Verilog or VHDL) is written and verified using a simulator, a project can be created using Quartus Prime Lite. Writing the HDL code and simulating it will be covered in later sections. Bring up Quartus Prime Lite, then use Windows Explorer to browse to C:\Users\nelso\Documents to create a new directory called: “EPT_5M240_FG_Blinky”.

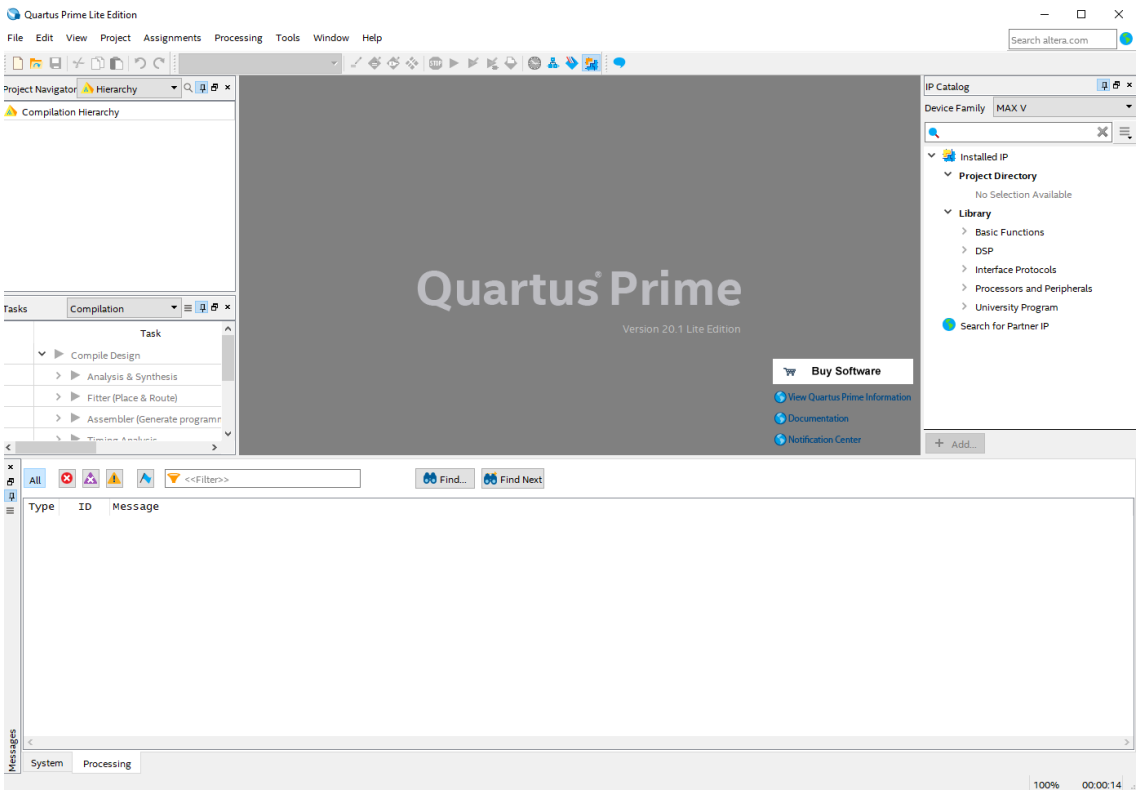


Create a folder called: EPT_5M240_FG_Blinky\EPT_5M240_AP_S2_Top directory to store your project. Type in a name for your project “EPT_5M240_AP_S2_Top”.

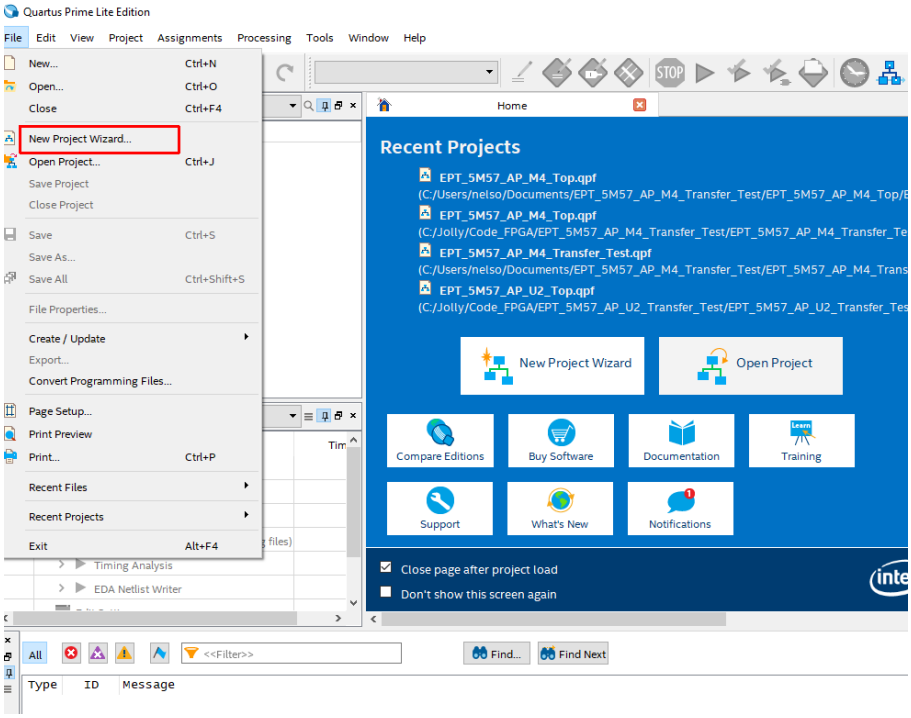


Open Quartus Prime by clicking on the icon 

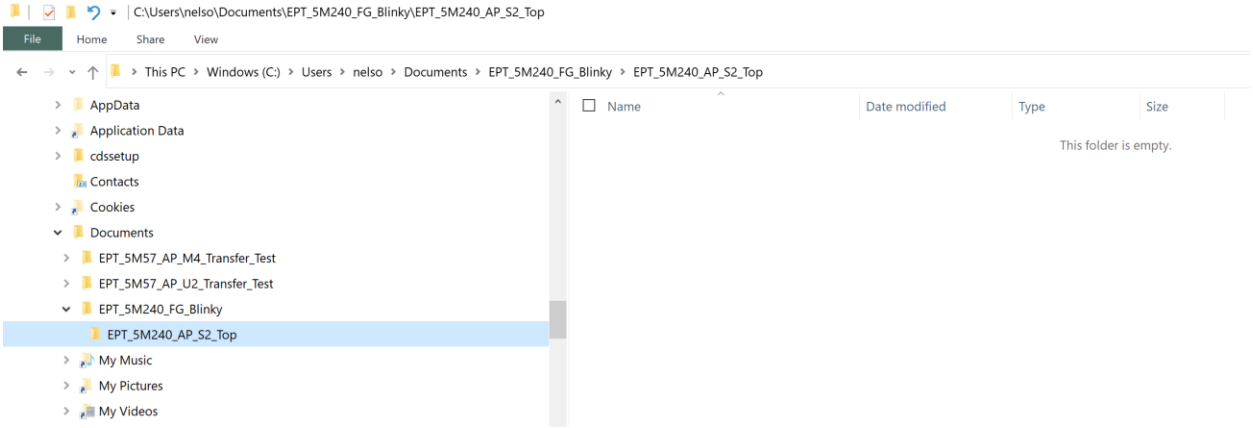
.

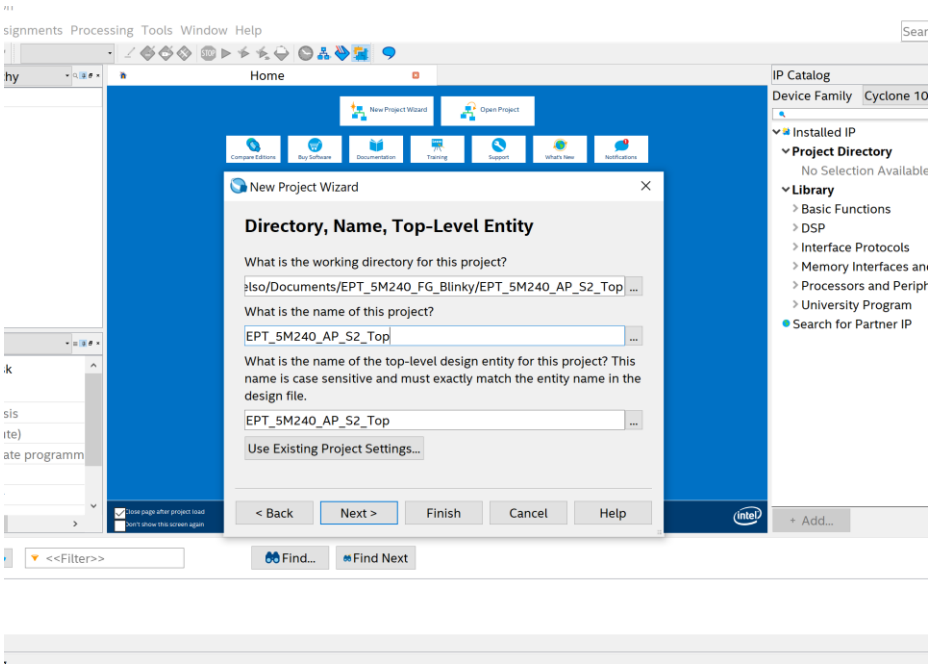


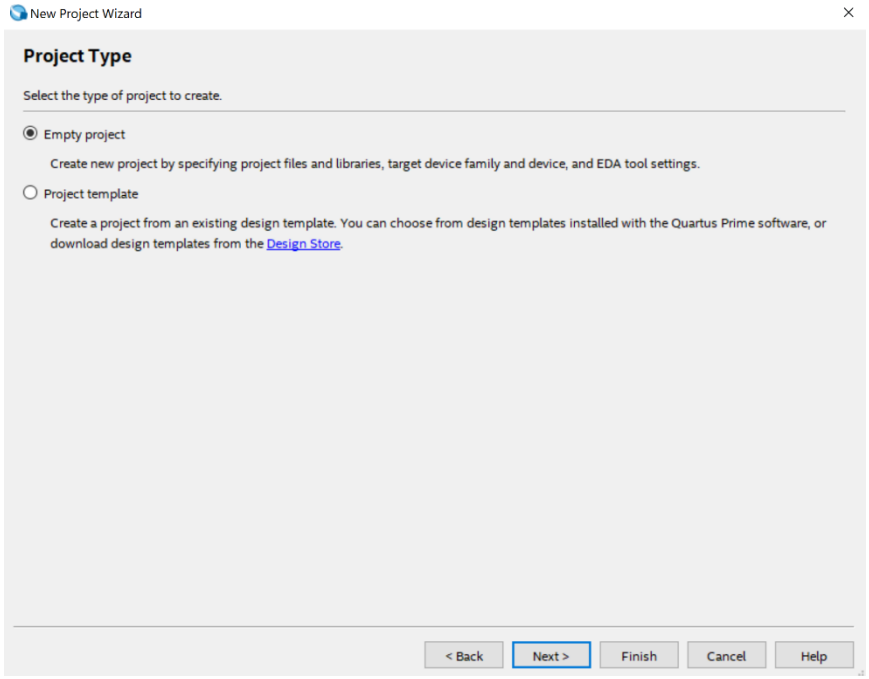
Under Quartus, Select File->New Project Wizard. The Wizard will walk you through setting up files and directories for your project.



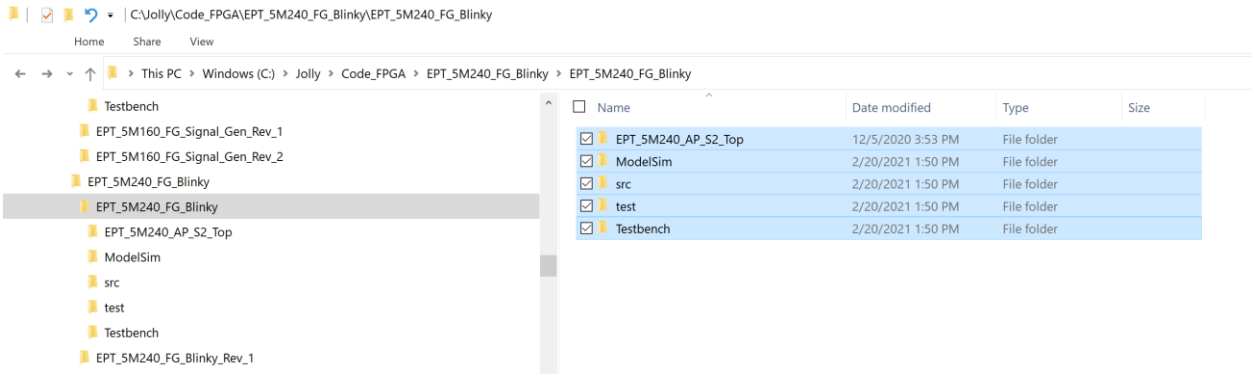
At the Top-Level Entity page, browse to the C:\Users\<your name>\Documents\EPT_5M240_FG_Blinky\EPT_5M240_AP_S2_Top directory to store your project. Type in a name for your project “EPT_5M240_AP_S2_Top”.



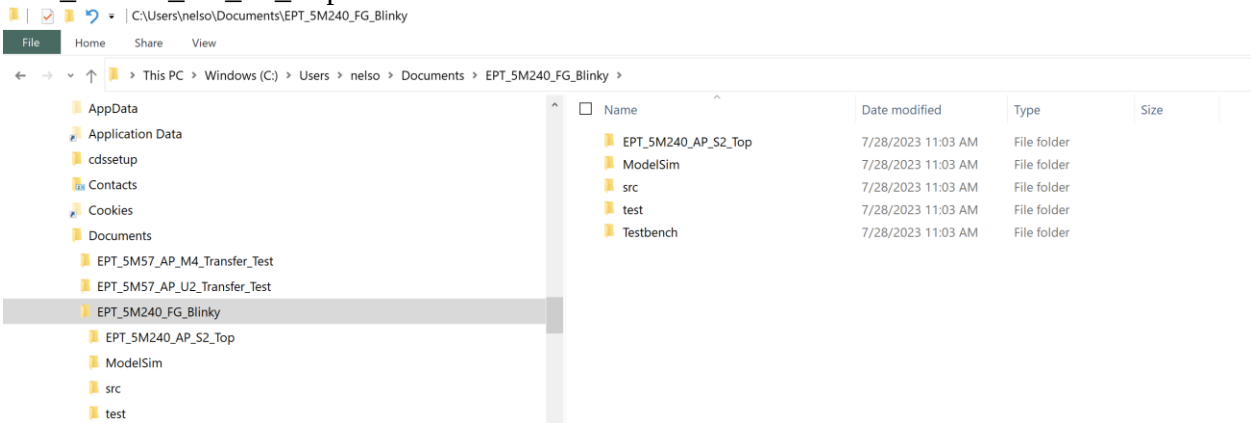

Home
New Project Wizard
Directory, Name, Top-Level Entity
What is the working directory for this project?
What is the name of this project?
EPT_5M240_AP_S2_Top
What is the name of the top-level design entity for this project? This name is case sensitive and must exactly match the entity name in the design file.
EPT_5M240_AP_S2_Top
Use Existing Project Settings...
< Back
Next >
Finish
Cancel
Help
IP Catalog
Device Family
Installed IP
Project Directory
Library
Basic Functions
DSP
Interface Protocols
Memory Interfaces
Processors and Periph
University Program
Search for Partner IP
Find...
Find Next


New Project Wizard
Project Type
Select the type of project to create.
Empty project
Create new project by specifying project files and libraries, target device family and device, and EDA tool settings.
Project template
Create a project from an existing design template. You can choose from design templates installed with the Quartus Prime software, or download design templates from the Design Store.
< Back
Next >
Finish
Cancel
Help

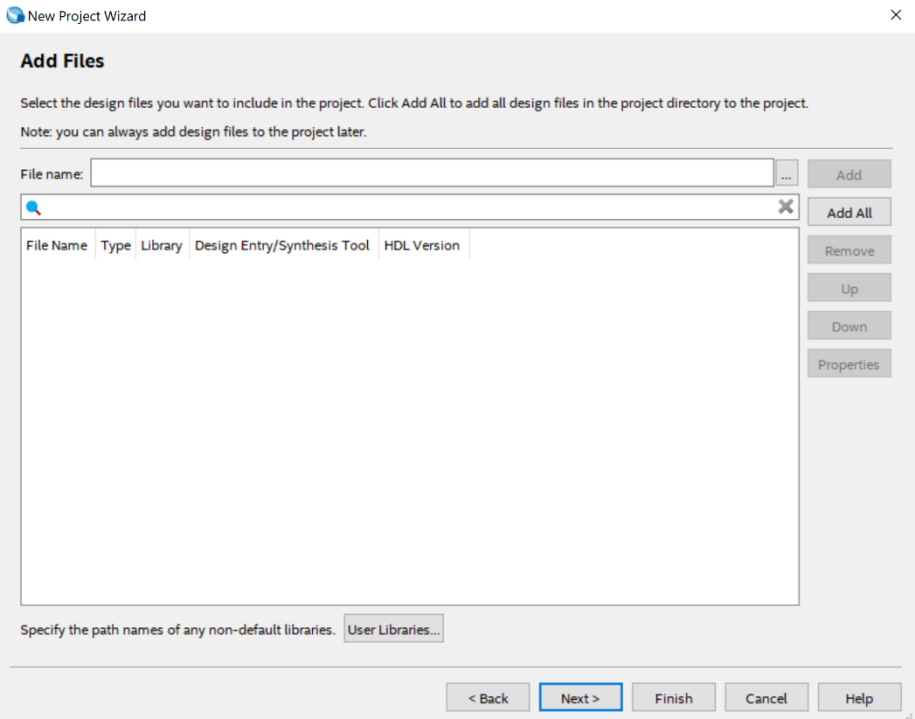
Select Next. At the Add Files window: Browse to the \Projects_HDL\ EPT_5M240_FG_Blinky folder of the BeeProLogicDevelopment System DVD. Copy the files from the following directory.



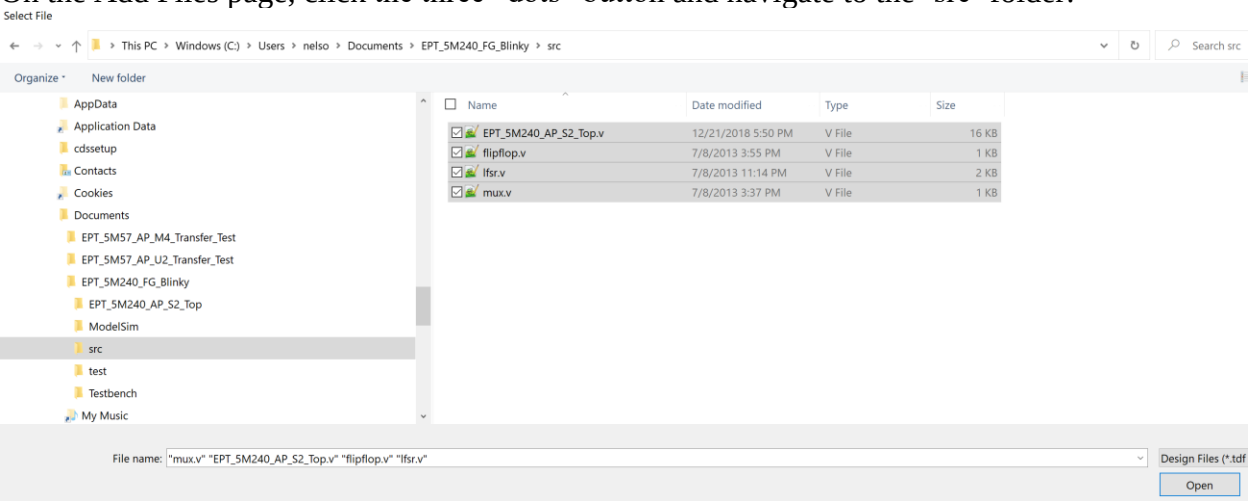
Then paste them in the C:\users\<your name>\Documents\EPT_5M240_FG_Blinky\ EPT_5M240_AP_S2_Top folder:



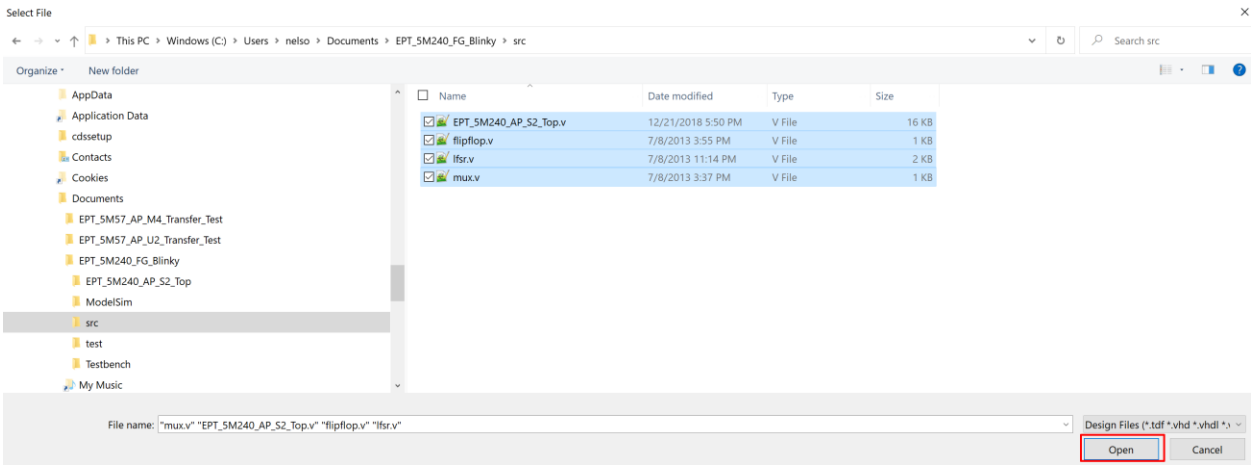
Then go to the “Add Files” page:

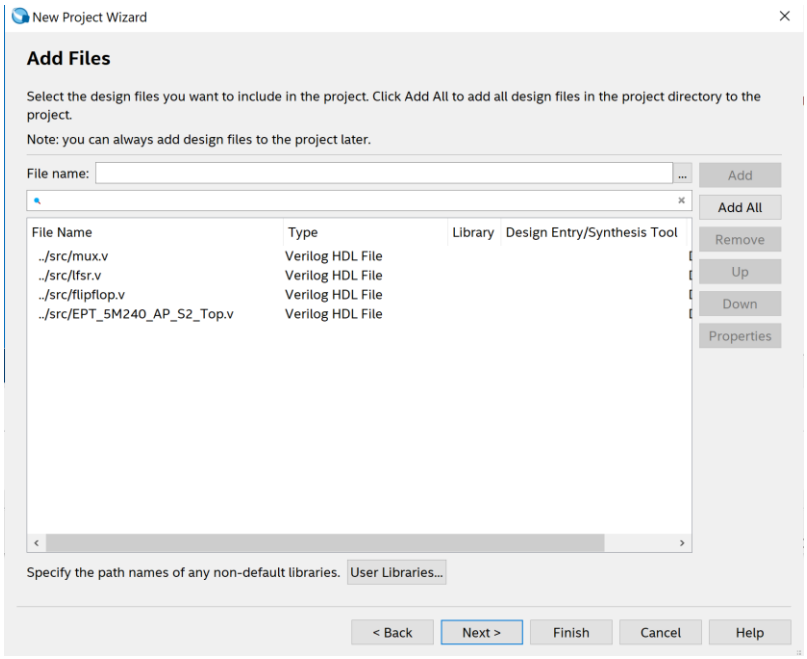


On the Add Files page, click the three “dots” button and navigate to the ‘src’ folder:

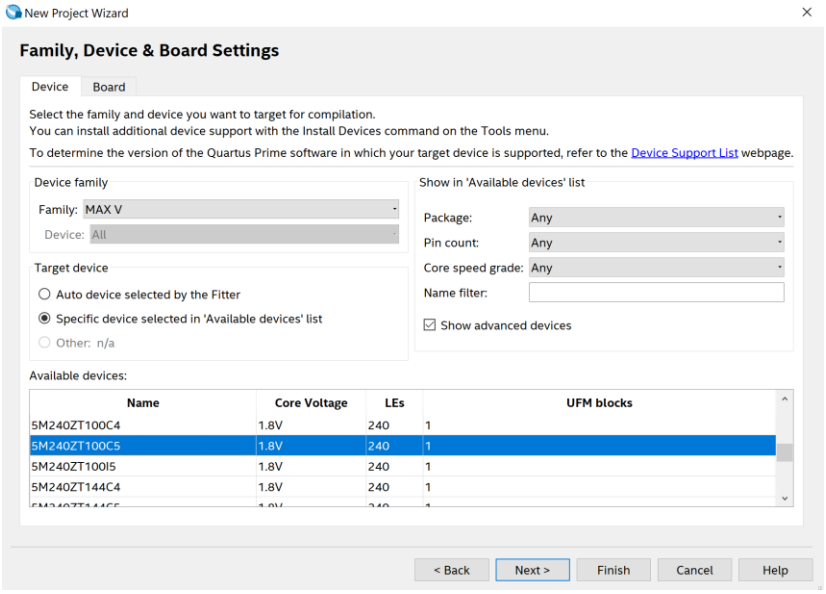


- EPT_5M240_AP_S2_Top.v
- Flipflop.v
- Lfsr.v
- Mux.v

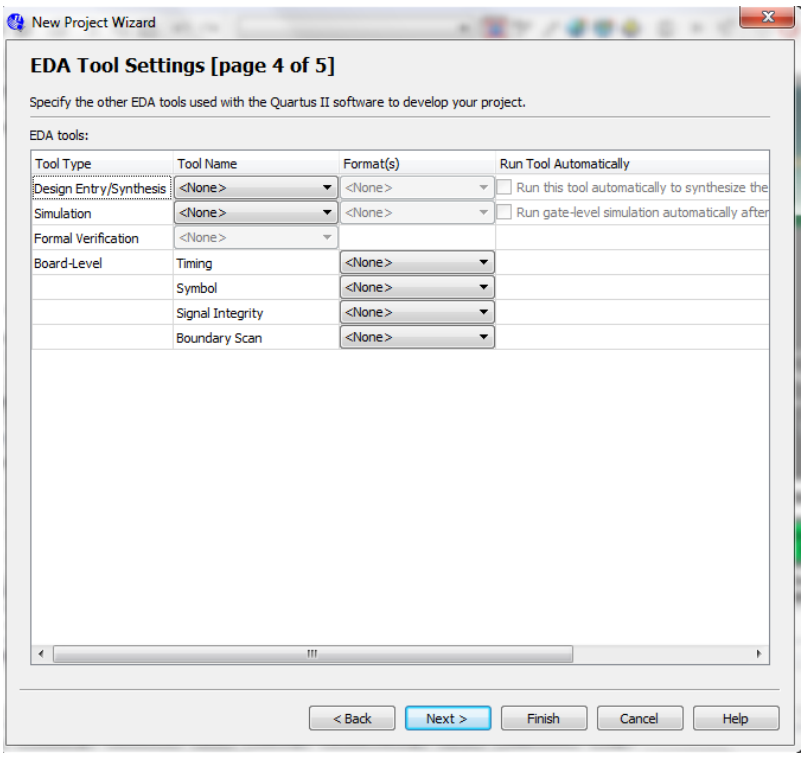




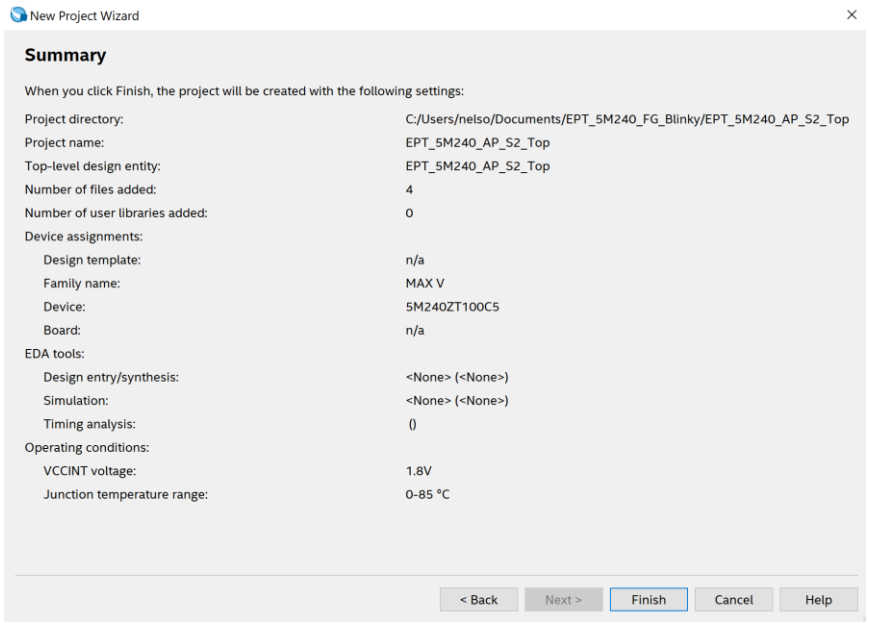
Select Next, at the Device Family group, select MAX V for Family. In the Available Devices group, browse down to 5M240ZT100C5 for Name.



Select Next, leave defaults for the EDA Tool Settings.



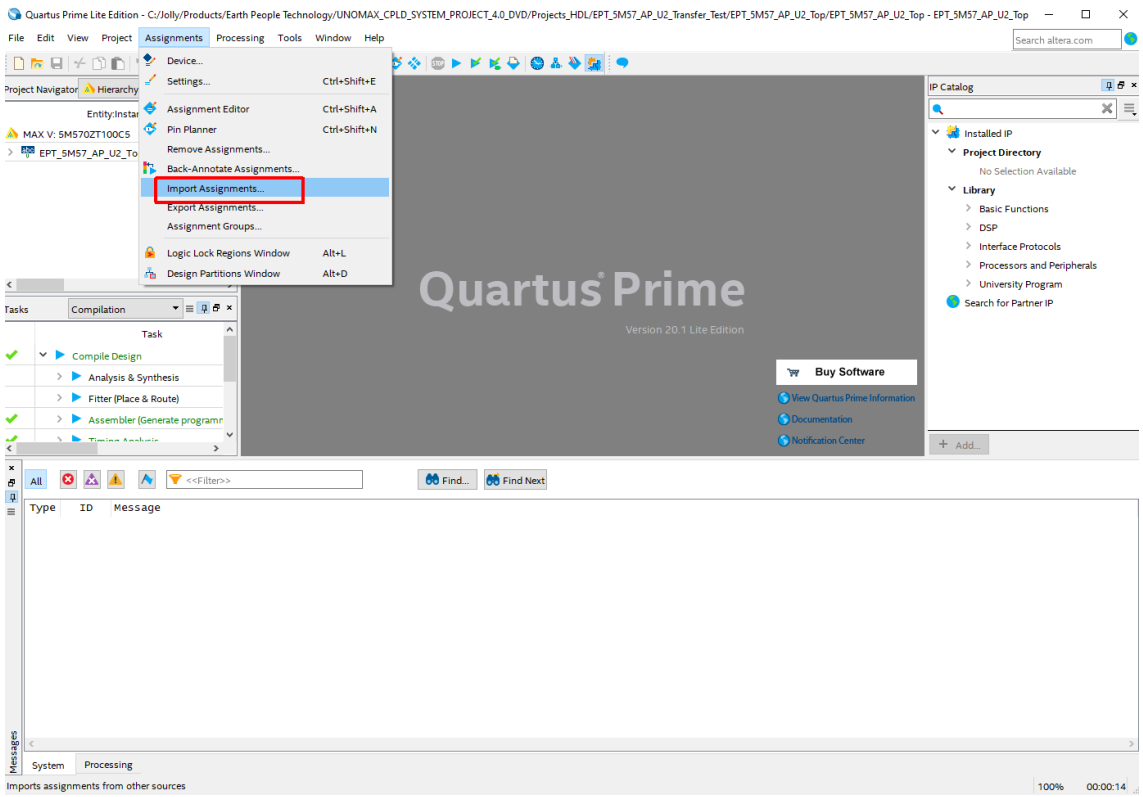
Select Next, then select Finish. You are done with the project level selections.



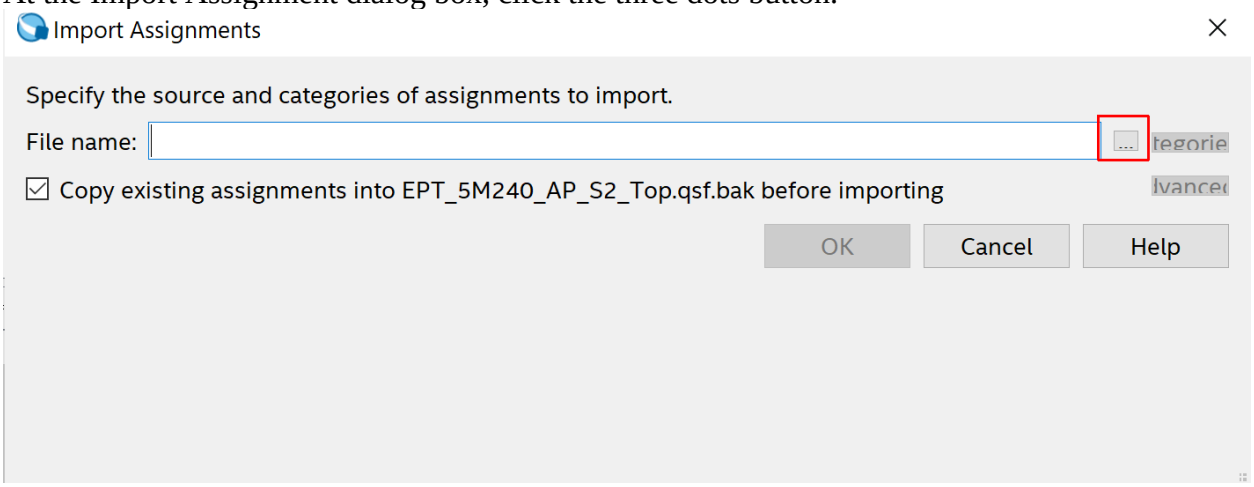
Next, we will select the pins and synthesize the project.

5.1.1 Selecting Pins and Synthesizing

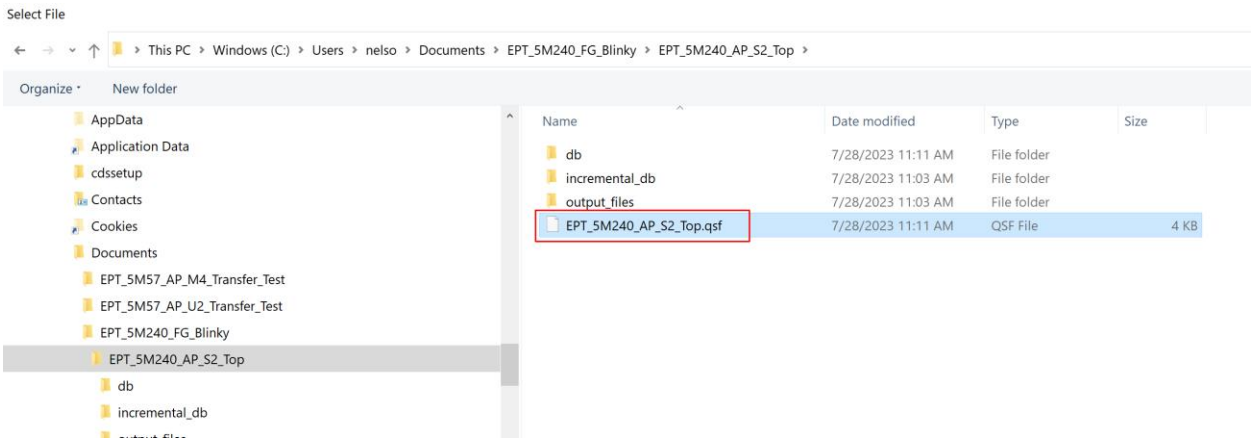
With the project created, we need to assign pins to the project. The signals defined in the top level file (in this case: EPT_5M240_AP_S2_Top.v) will connect directly to pins on the CPLD. The Pin Planner Tool from Quartus Prime will add the pins and check to verify that our pin selections do not violate any restrictions of the device. In the case of this example we will import pin assignments that created at an earlier time. Under Assignments, Select Import Assignments.

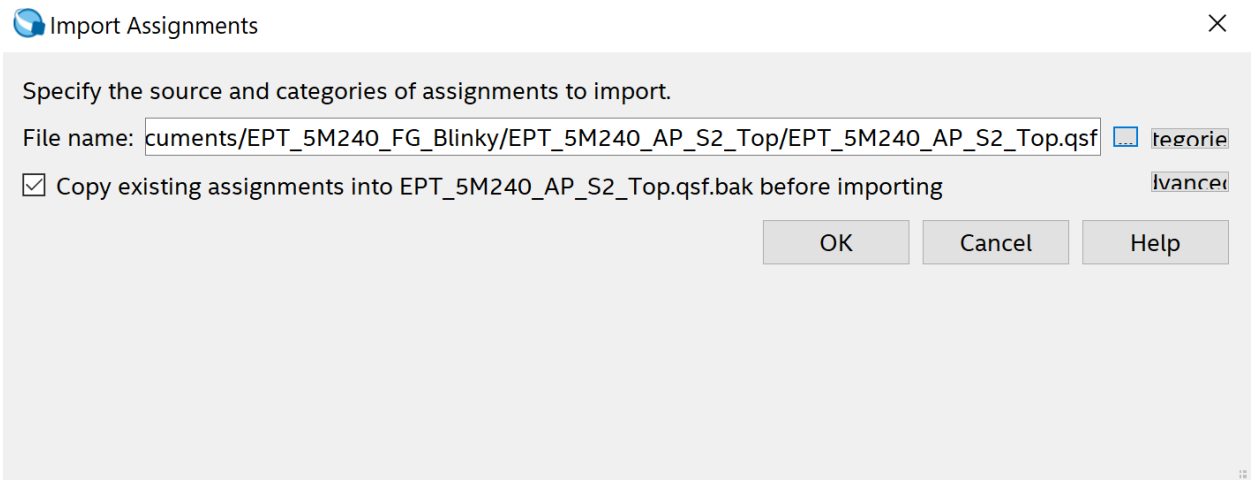


At the Import Assignment dialog box, click the three dots button.

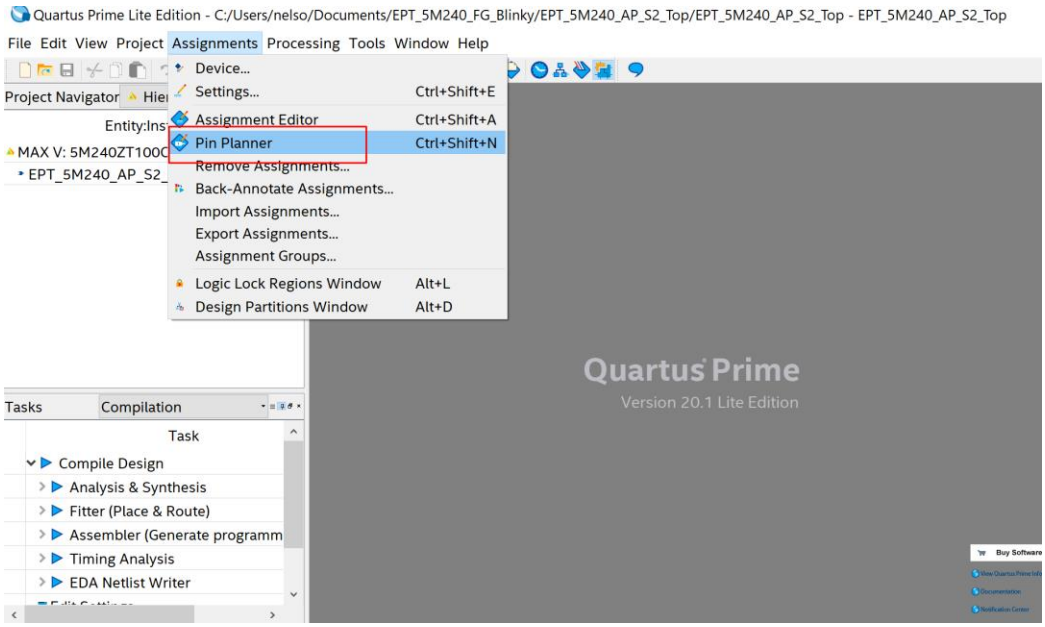


Browse to the \Projects_HDL\ EPT_5M240_FG_Blinky\EPT_5M240_AP_S2_Top folder of the BeeProLogicDevelopment System DVD. Select the “EPT_5M240_AP_S2_Top.qsf” file.

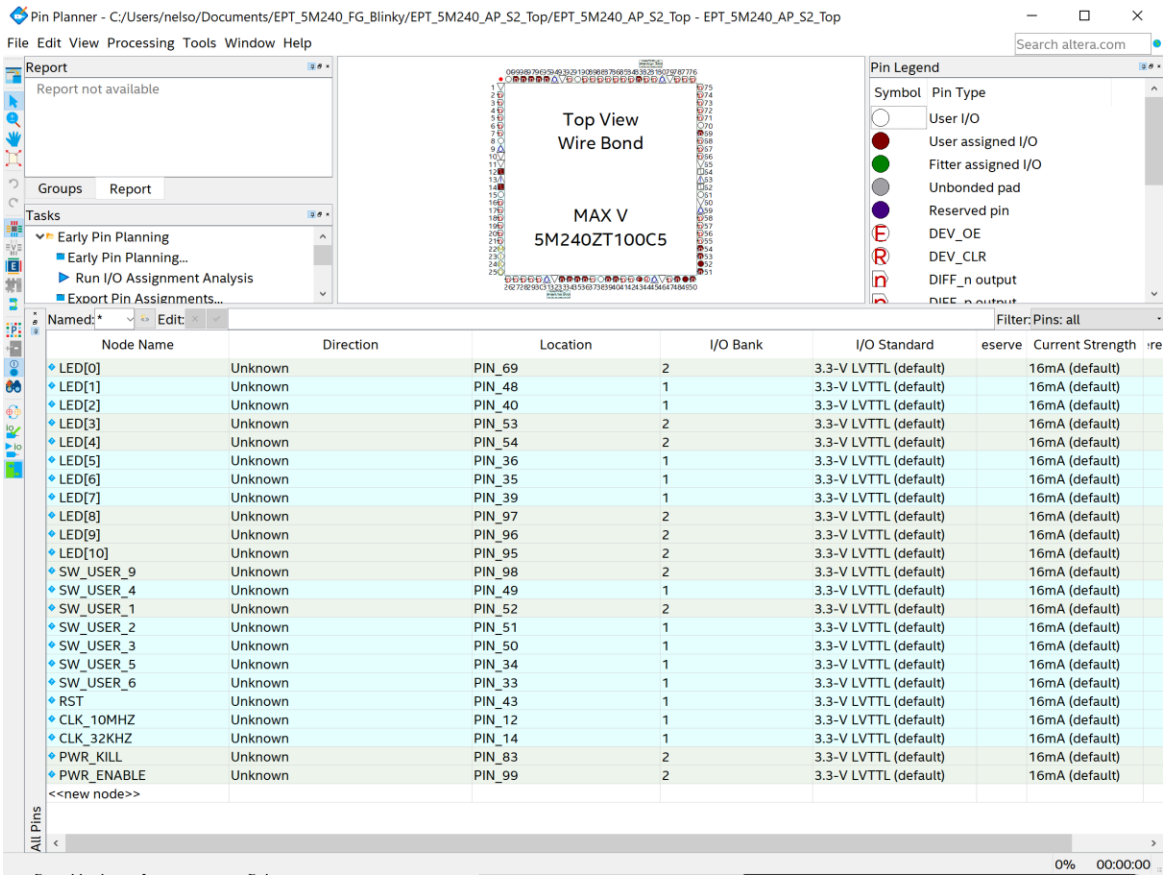




Click Ok. Under Assignments, Select Pin Planner. Verify the pins have been imported correctly.

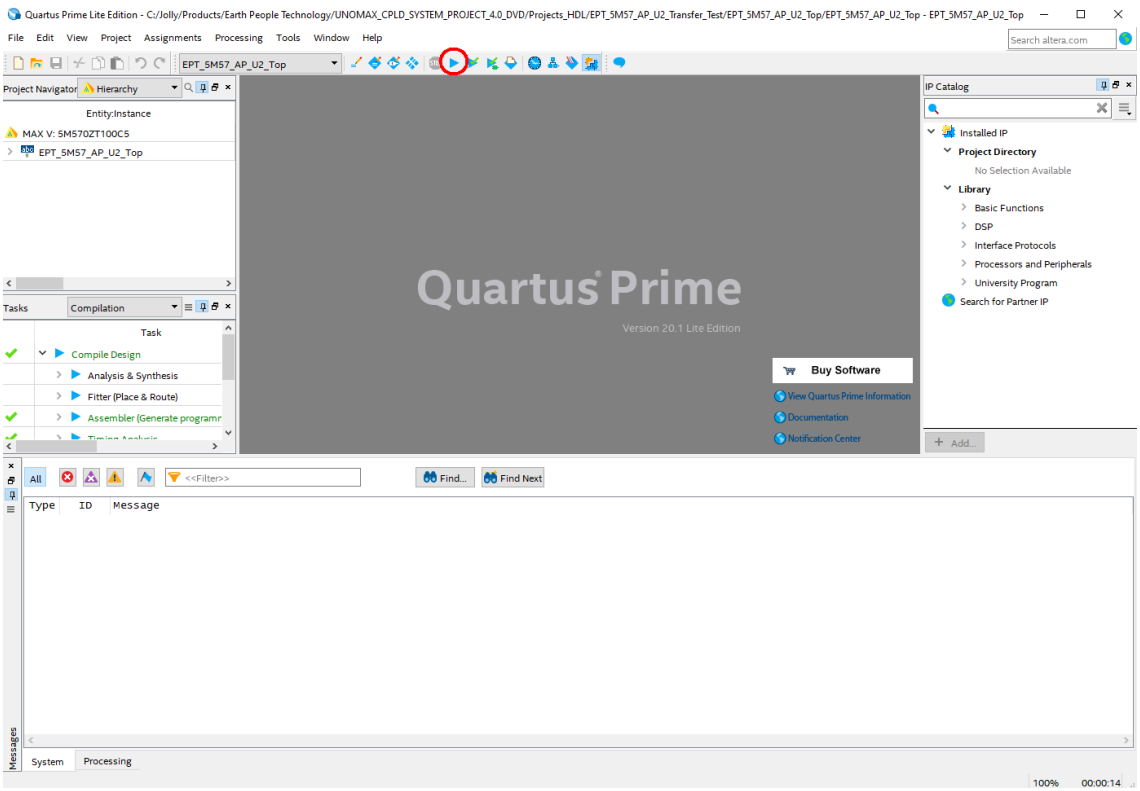


The pin locations should not need to be changed for BeeProLogicDevelopment System. However, if you need to change any pin location, just click on the “location” column for the particular node you wish to change. Then, select the new pin location from the drop down box.

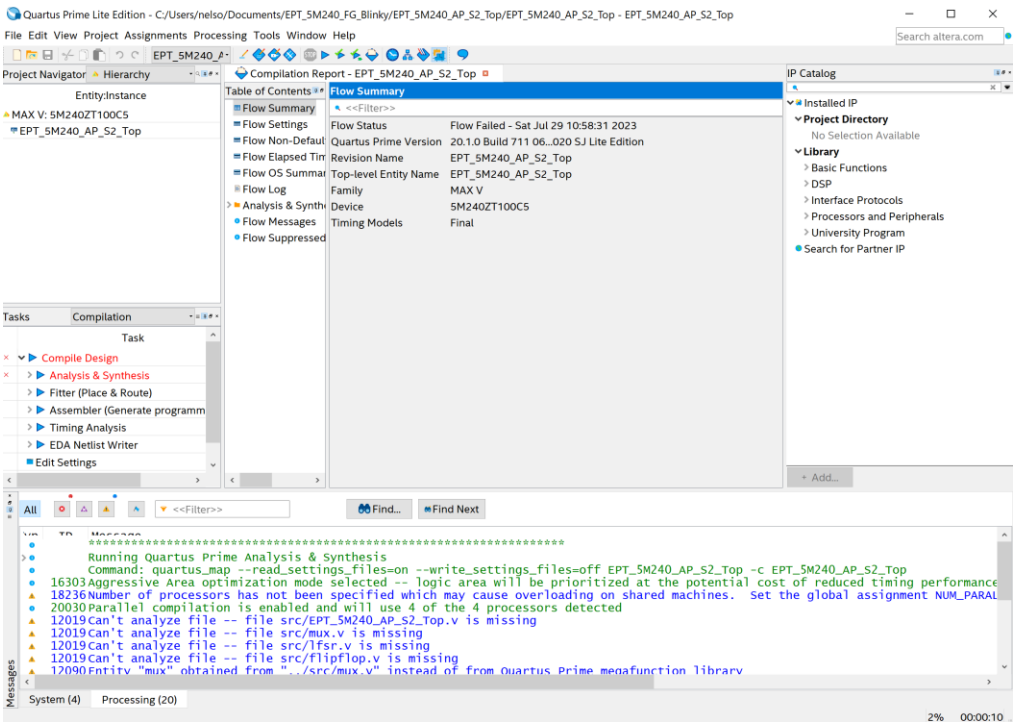


Node Name	Direction	Location	I/O Bank	I/O Standard	eserve	Current Strength	re
LED[0]	Unknown	PIN_69	2	3.3-V LVTTL (default)		16mA (default)	
LED[1]	Unknown	PIN_48	1	3.3-V LVTTL (default)		16mA (default)	
LED[2]	Unknown	PIN_40	1	3.3-V LVTTL (default)		16mA (default)	
LED[3]	Unknown	PIN_53	2	3.3-V LVTTL (default)		16mA (default)	
LED[4]	Unknown	PIN_54	2	3.3-V LVTTL (default)		16mA (default)	
LED[5]	Unknown	PIN_36	1	3.3-V LVTTL (default)		16mA (default)	
LED[6]	Unknown	PIN_35	1	3.3-V LVTTL (default)		16mA (default)	
LED[7]	Unknown	PIN_39	1	3.3-V LVTTL (default)		16mA (default)	
LED[8]	Unknown	PIN_97	2	3.3-V LVTTL (default)		16mA (default)	
LED[9]	Unknown	PIN_96	2	3.3-V LVTTL (default)		16mA (default)	
LED[10]	Unknown	PIN_95	2	3.3-V LVTTL (default)		16mA (default)	
SW_USER_9	Unknown	PIN_98	2	3.3-V LVTTL (default)		16mA (default)	
SW_USER_4	Unknown	PIN_49	1	3.3-V LVTTL (default)		16mA (default)	
SW_USER_1	Unknown	PIN_52	2	3.3-V LVTTL (default)		16mA (default)	
SW_USER_2	Unknown	PIN_51	1	3.3-V LVTTL (default)		16mA (default)	
SW_USER_3	Unknown	PIN_50	1	3.3-V LVTTL (default)		16mA (default)	
SW_USER_5	Unknown	PIN_34	1	3.3-V LVTTL (default)		16mA (default)	
SW_USER_6	Unknown	PIN_33	1	3.3-V LVTTL (default)		16mA (default)	
RST	Unknown	PIN_43	1	3.3-V LVTTL (default)		16mA (default)	
CLK_10MHZ	Unknown	PIN_12	1	3.3-V LVTTL (default)		16mA (default)	
CLK_32KHZ	Unknown	PIN_14	1	3.3-V LVTTL (default)		16mA (default)	
PWR_KILL	Unknown	PIN_83	2	3.3-V LVTTL (default)		16mA (default)	
PWR_ENABLE	Unknown	PIN_99	2	3.3-V LVTTL (default)		16mA (default)	
<<new node>>							

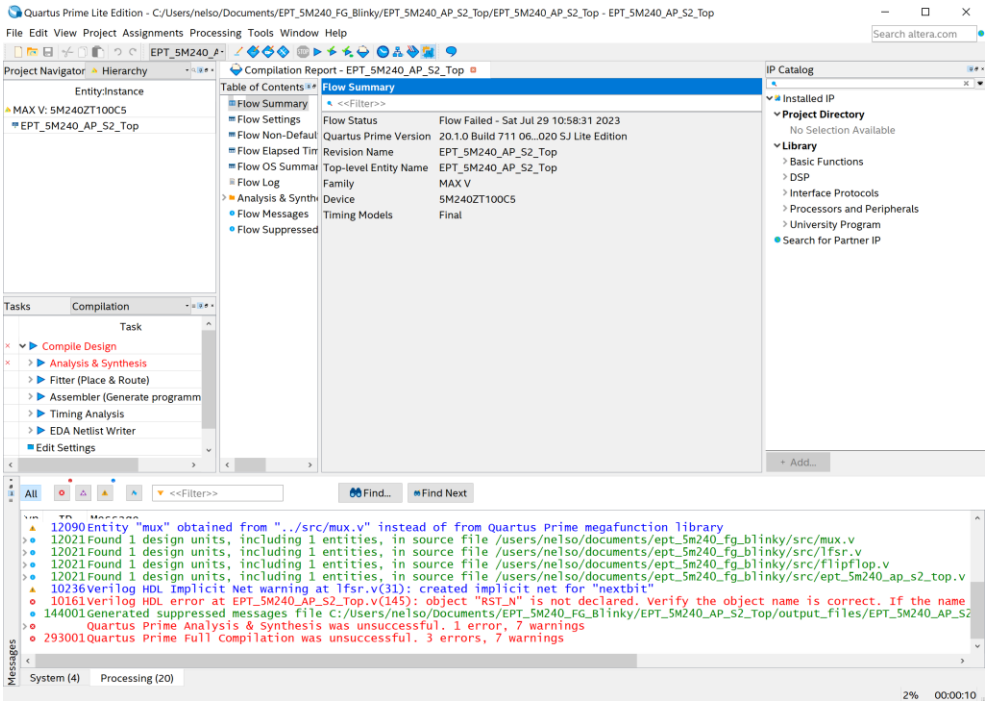
Exit the Pin Planner.
Select the Start Compilation button.



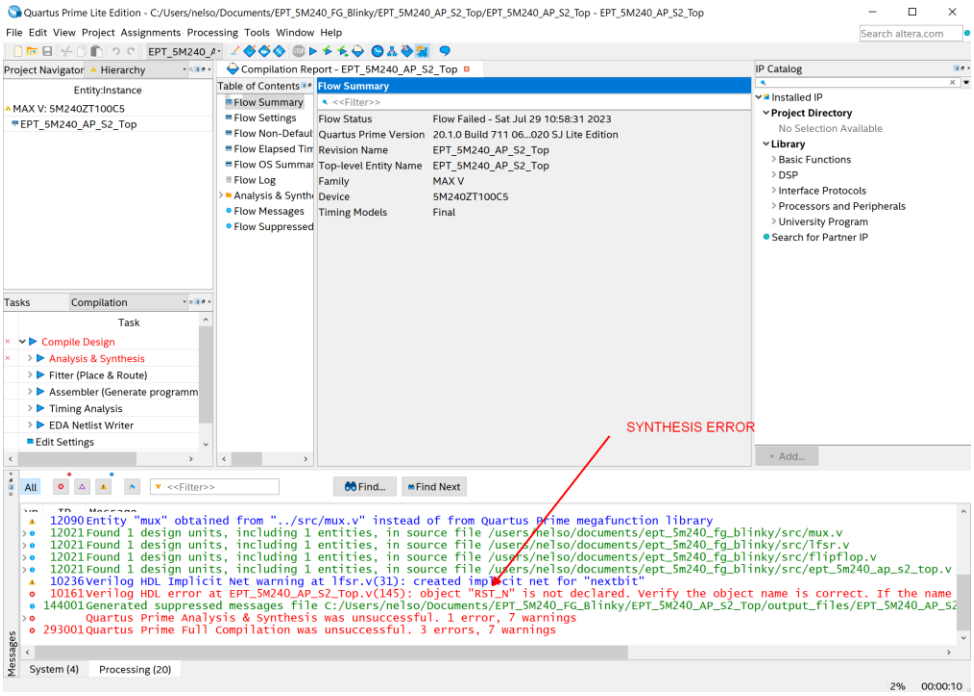
Then compilation will start.



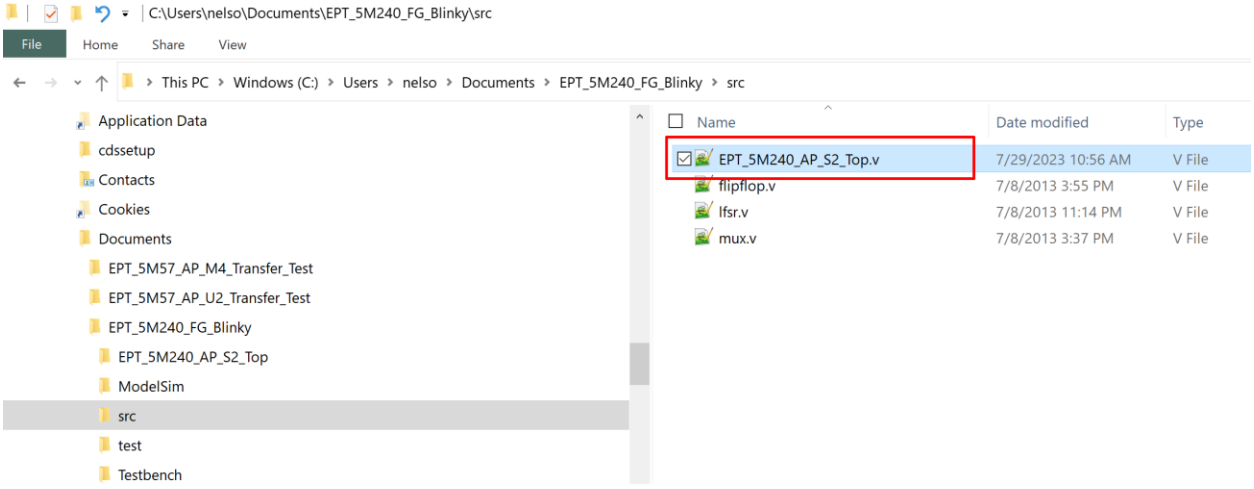
If you forget to include a file or some other error you should expect to see a screen similar to this:



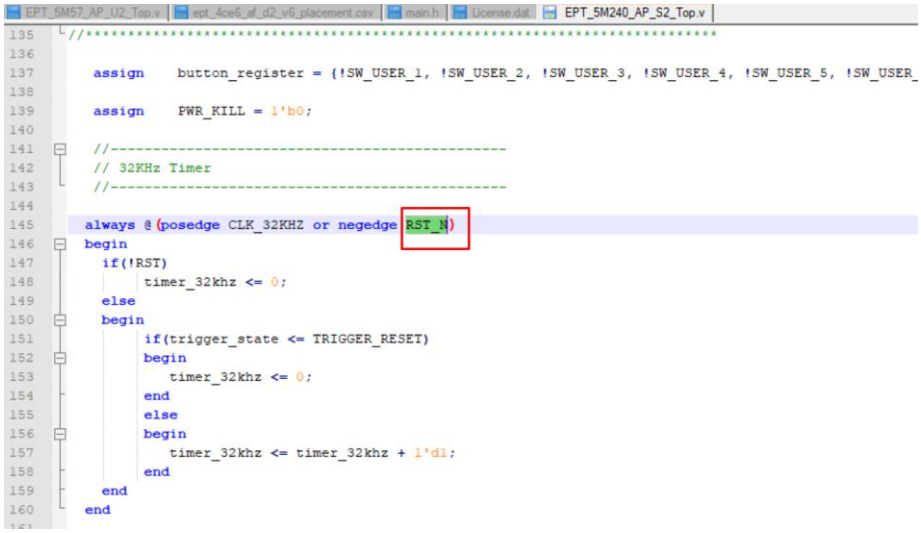
Click on the “Processing” tab at bottom to see the error.



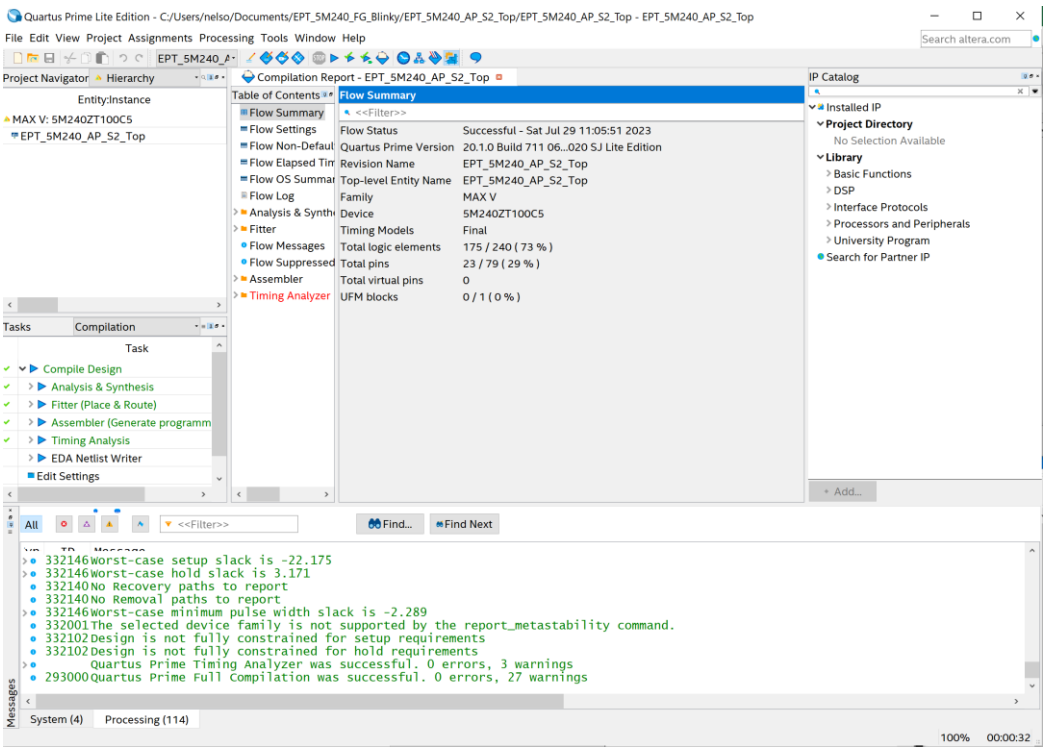
The error in this case is the missing object “RST_N”. Go to the source file “EPT_5M240_AP_S2_Top.v”, and open it in an editor.



Scroll down to the line, 145 and locate the object “RST_N”..



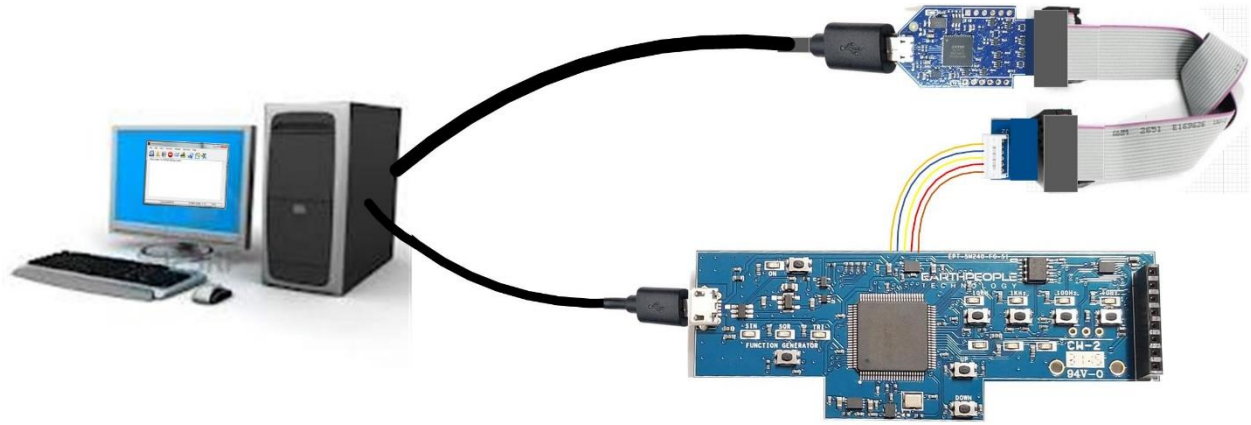
Chage the object to “RST”. Click save then re-run the Compile process. After successful completion, the screen should look like the following:



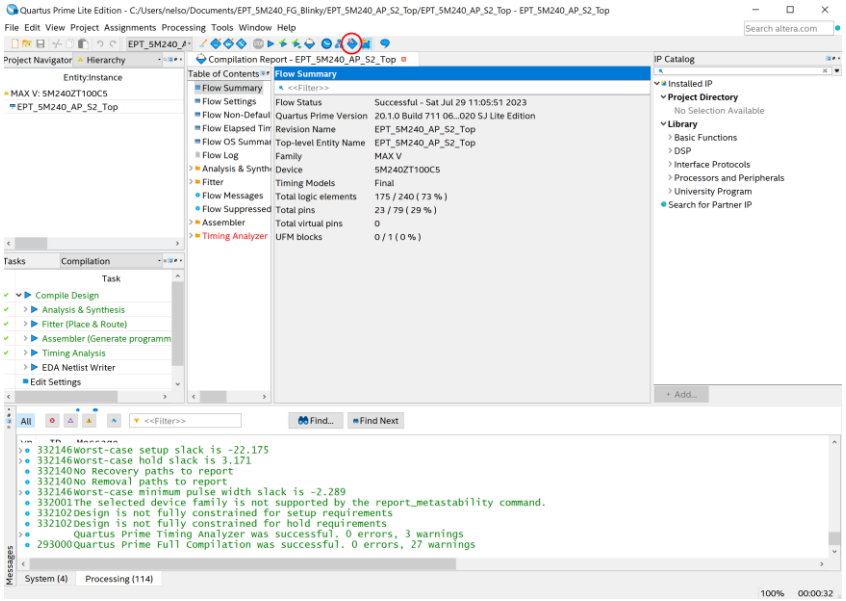
At this point the project has been successfully compiled, synthesized and a programming file has been produce. See the next section on how to program the CPLD.

5.1.2 Programming the CPLD

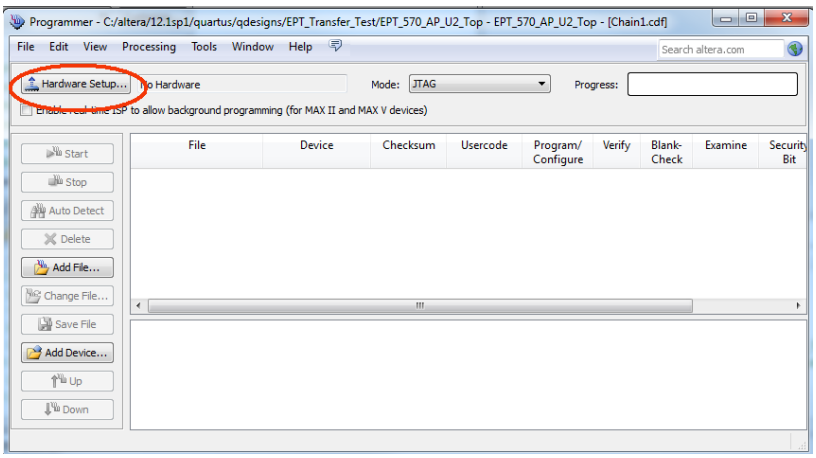
Programming the CPLD is quick and easy. All that is required is a standard USB Micro-B cable, JST Adapter board and the EPT_Blaster Driver DLL. Connect the BeeProLogic to the PC, open up Quartus Prime, open the programmer tool, and click the Start button. To program the CPLD, follow the steps to install the USB Driver and the JTAG Driver Insert for Quartus Prime.



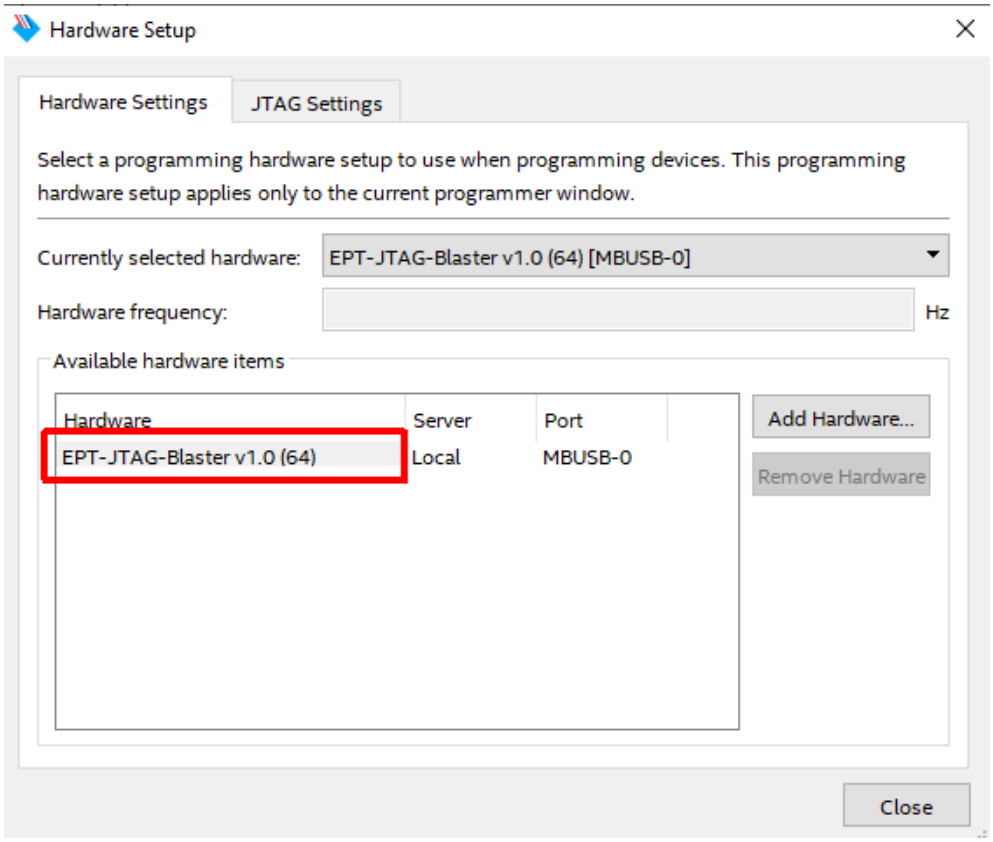
If the project created in the previous sections is not open, open it. Click on the Programmer button.



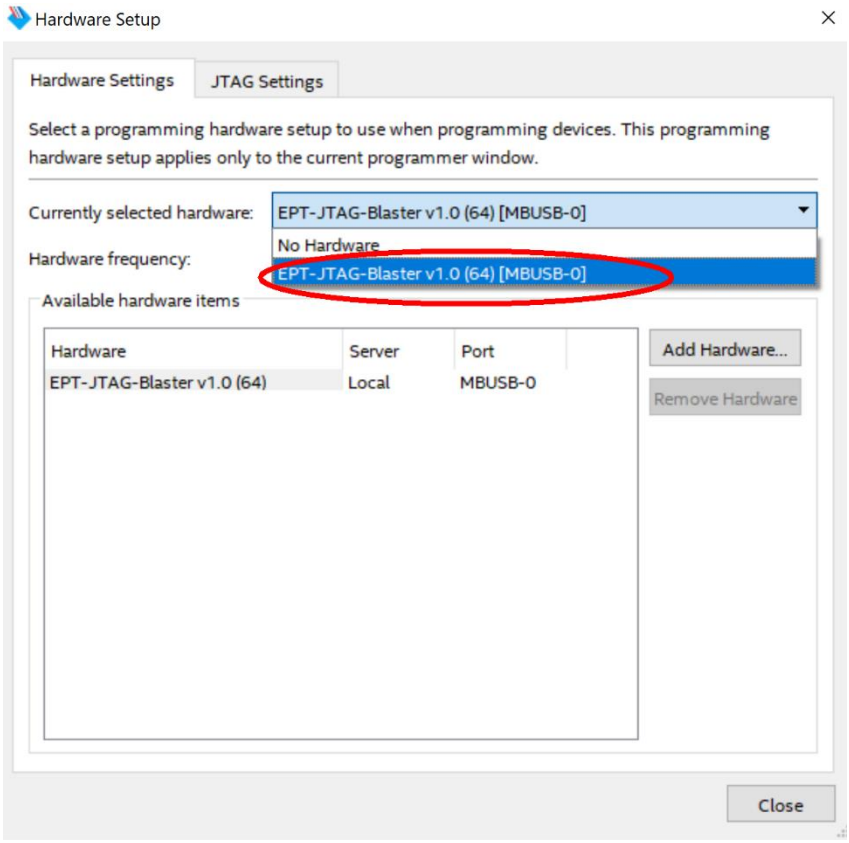
The Programmer Window will open up with the programming file selected. Click on the Hardware Setup button in the upper left corner.



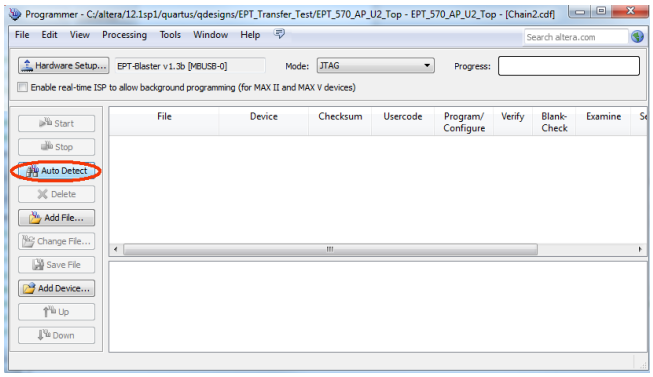
The Hardware Setup Window will open. In the “Available hardware items”, double click on “EPT-Blaster v1.0”.



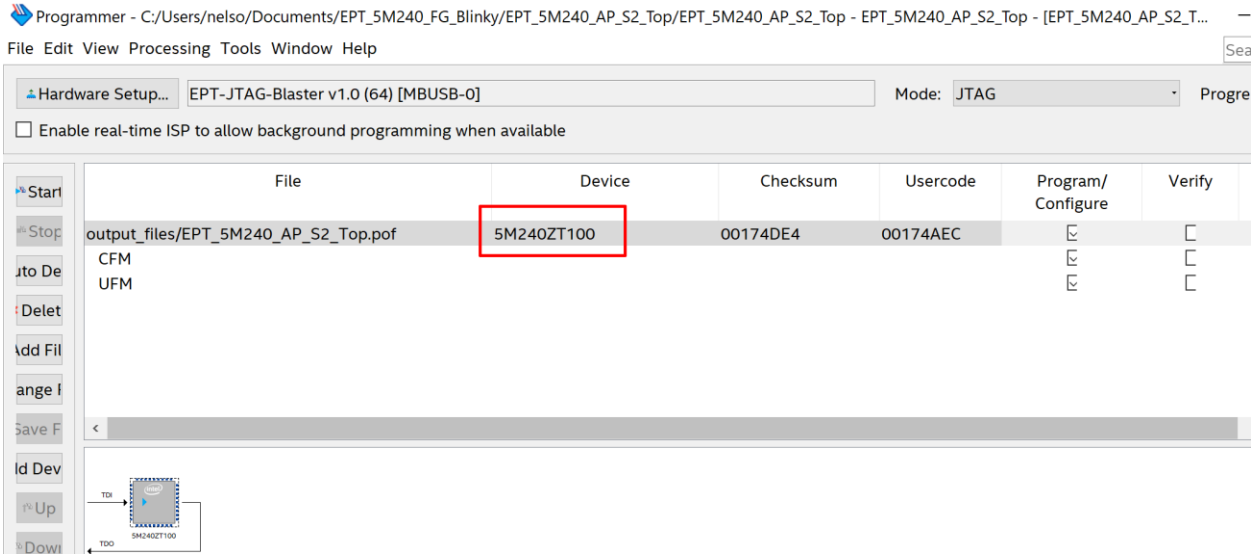
If you successfully double clicked, the “Currently selected hardware:” dropdown box will show the “EPT-Blaster v1.0 (64) [MBUSB-0]”.



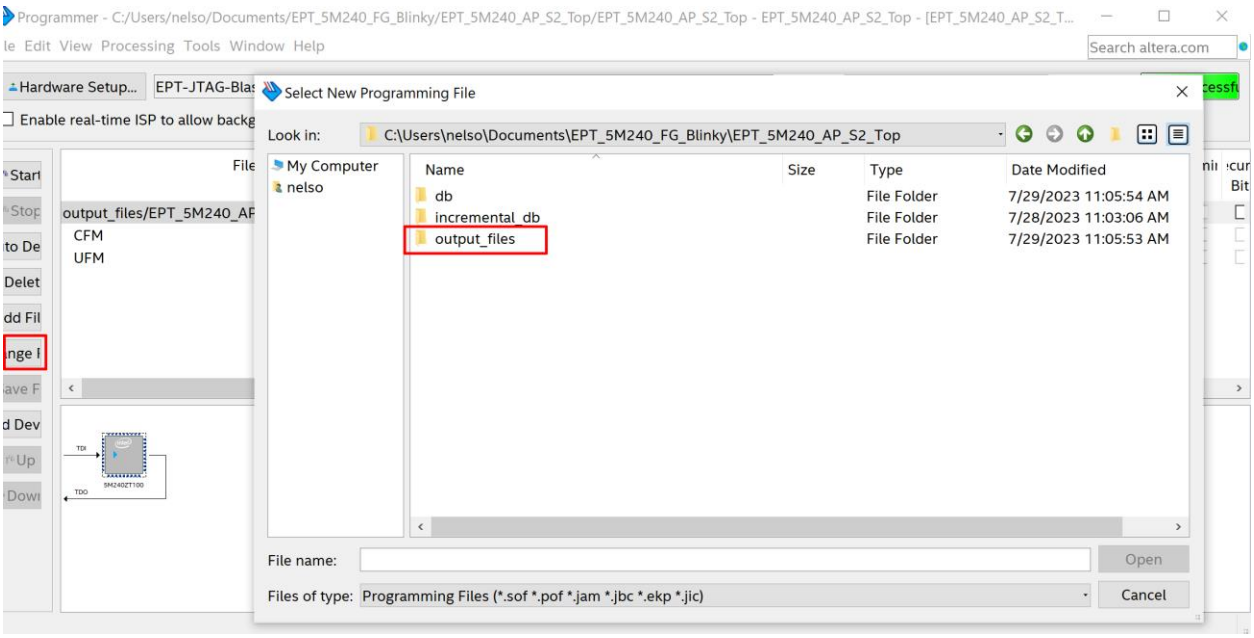
Click on the Auto-Detect button. This will verify that the EPT-Blaster driver can connect with the BeeProLogic device.



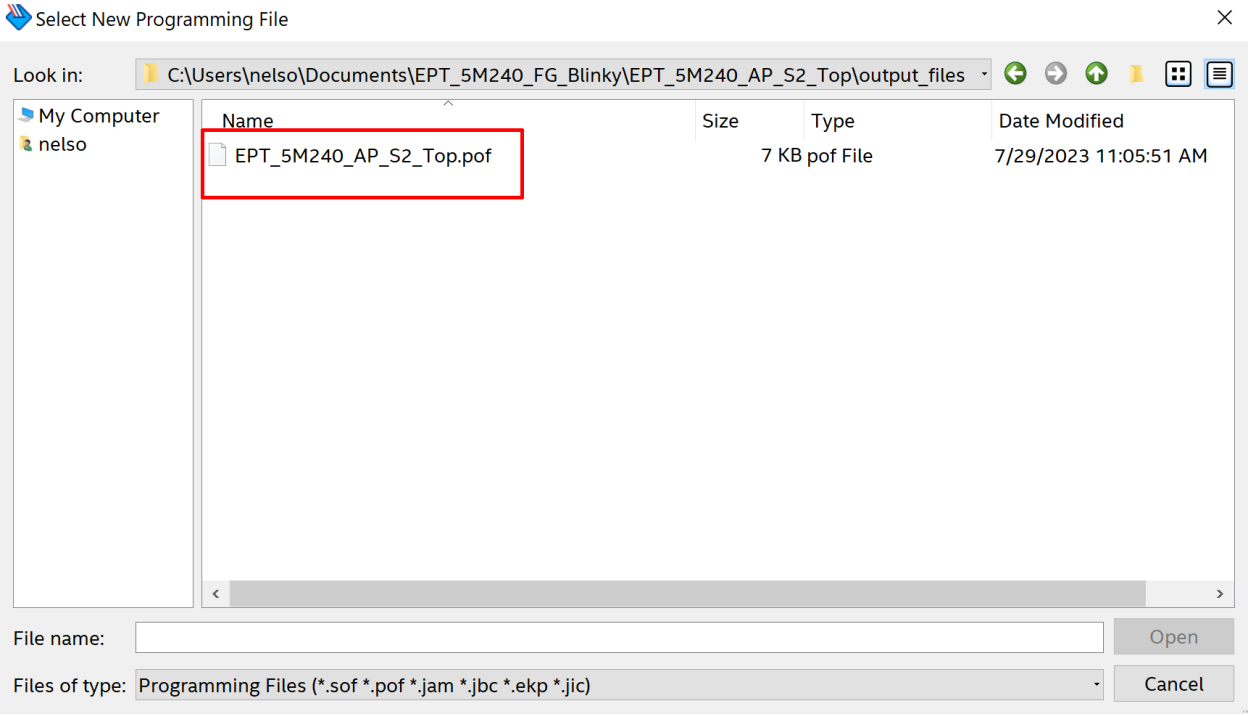
Select the 5M570 under “Device”.



Click on the “Change File” button and browse to the output_files folder.

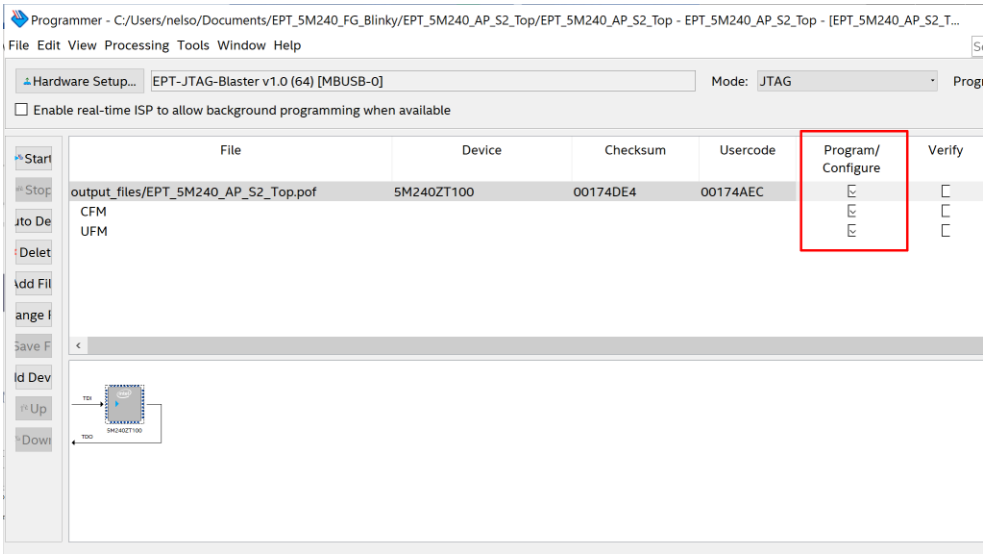


Click on the EPT_5M240_AP_S2_Top.pof file to select it.

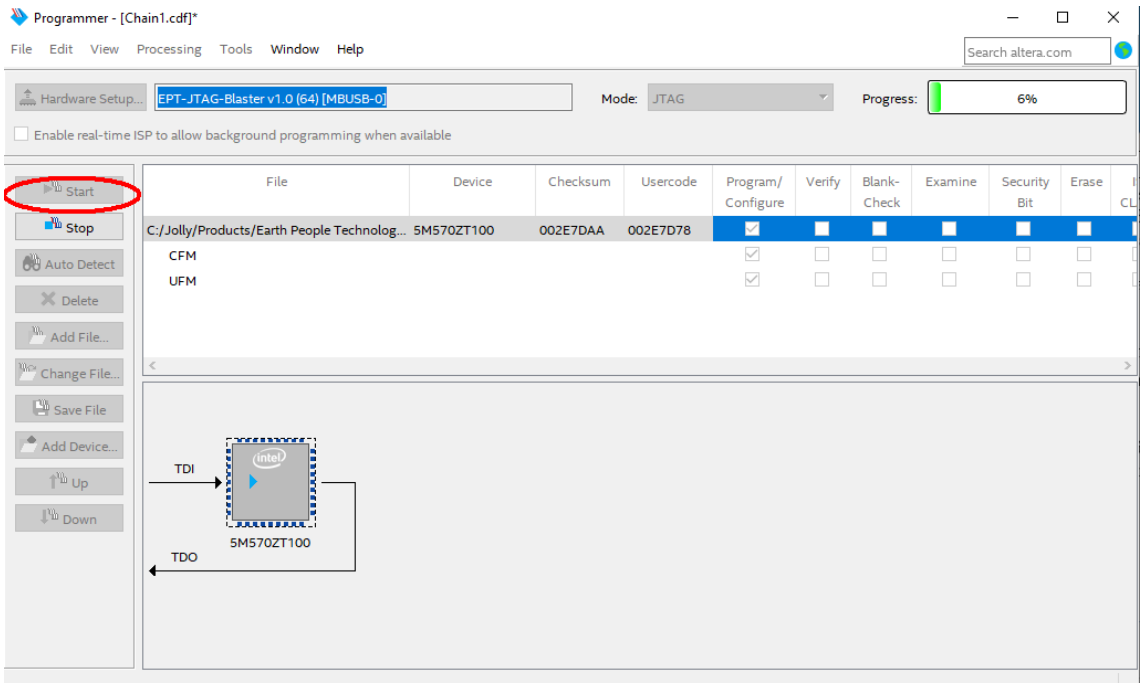


Click the Open button in the lower right corner.

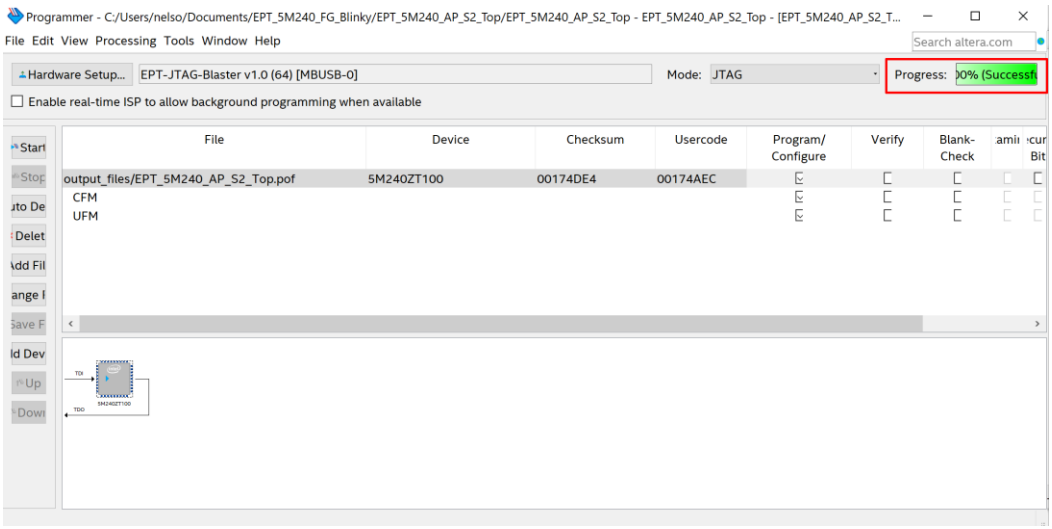
Next, selet the checkbox under the “Program/Configure” of the Programmer Tool. The checkboxes for the CFM and UFM will be selected automatically.



Click on the Start button to to start programming the CPLD. The Progress bar will indicate the progress of programming.



When the programming is complete, the Progress bar will indicate success.



At this point, the BeeProLogic is programmed and ready for use. To test that the CPLD is properly programmed, follow the next section to see the LEDs blink in a heart beat pattern.

5.1.3 Sample Code Operation

Once the BeeProLogic is programmed, operate the sample code by pressing the SW2. The LEDs will blink in a heart beat pattern. Press SW3 to see a different pattern.