Introduction to FPGA Simple PWM Tutorial

For the BeMicro MAX 10 FPGA Evaluation Kit

Version 14.0.2	10/17/2014	Tutorial

Table of Contents

MODULE 1. GETTING STARTED AND INSTALLATION OF TOOLS

BeMicro Max 10 is a FPGA evaluation kit that is designed to get you started with using an FPGA. BeMicro Max 10 adopts Altera's non-volatile **MAX® 10 FPGA** built on 55-nm flash process.

MAX 10 FPGAs revolutionize non-volatile integration by delivering advanced processing capabilities in a low-cost, instant-on, small form factor programmable logic device. The devices also include full-featured FPGA capabilities such as digital signal processing, analog functionality, Nios II embedded processor support and memory controllers.

The BeMicro Max 10 includes a variety of peripherals connected to the FPGA device, such as 8MB SDRAM, accelerometer, digital-to-analog converter (DAC), temperature sensor, thermal resistor, photo resistor, LEDs, pushbuttons and several different options for expansion connectivity.

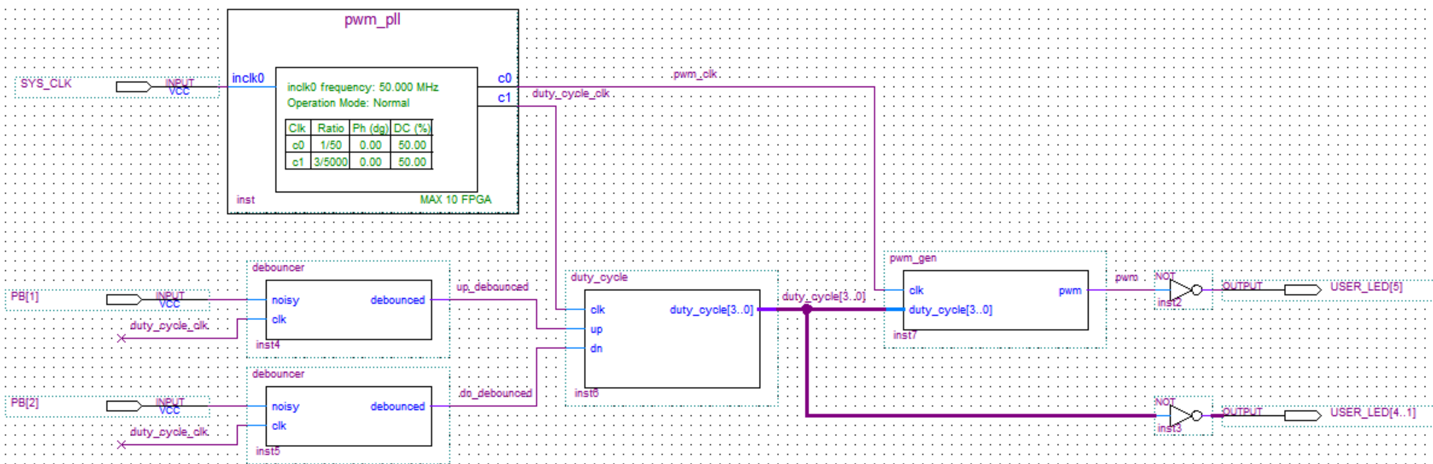
Before continuing with this Tutorial, ensure that the Altera tools and drivers have been installed. Please refer to the *BeMicro MAX 10 Getting Started User Guide* for instructions.

This tutorial will show the basic flow of how to create an FPGA design using the Altera Quartus II design software. A simple FPGA design will create a PWM component that and interact with the LEDs on the kit.

MODULE 2. YOUR FIRST FPGA DESIGN

Overview: During this exercise, you will follow a step-by-step guide to create a simple design. To do this you will configure a PLL block, connect together a few simple blocks created with Verilog code, assign pins, and download to a target board, either a BeMicro MAX 10, a BeMicro CV or a BeMicroSDK.

The final design will be structured as follows:



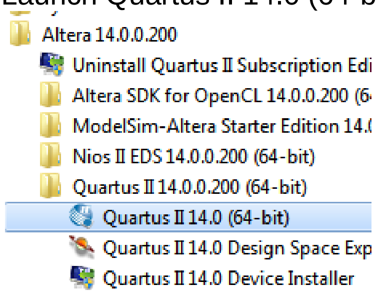
The design above is a circuit that will be used to generate a PWM output which will drive one of the LEDs on the kit and vary the LED intensity. Two pushbuttons on the kit will be used to increase or decrease the PWM duty cycle value between 16 steps, 6.25% each. The **`pwm_pll`** block is a PLL within the MAX 10 FPGA device which will be used to take the incoming 50 MHz clock input and generate lower clock frequencies used to clock the PWM logic. The **`debouncer`** blocks are created from Verilog code to sample the incoming pushbuttons, debounce them and generate a single pulse output when the button is pressed. The **`duty_cycle`** block is implemented with Verilog code which increments or decrements a duty cycle value based on the button presses. This 4-bit duty cycle value will also be displayed on LEDs on the kit. The **`pwm_gen`** block is created from Verilog code and generated the PWM'ed output which will drive the LEDs.

2.1 Extract the Tutorial Files

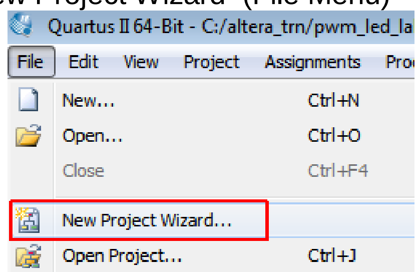
The lab files are included in a ZIP file called **`pwm_led_lab_files.zip`** which you should have downloaded with this Tutorial document. Extract these lab files to a location on your computer, such as **`c:\altera_trn\`**. The lab files provide you with a few very simple Verilog files that will be used to create simple PWM logic in our FPGA project and a partially completed top-level schematic which you will complete as part of the tutorial.

2.2 Part A: Create the Quartus II Project

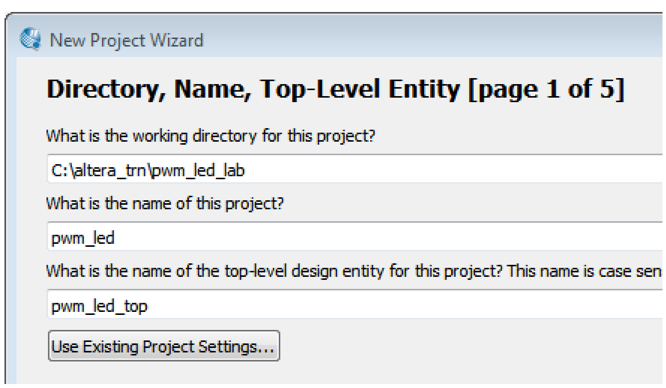
1) Launch Quartus II 14.0 (64-bit)



2) Create a new project using the New Project Wizard (File Menu)

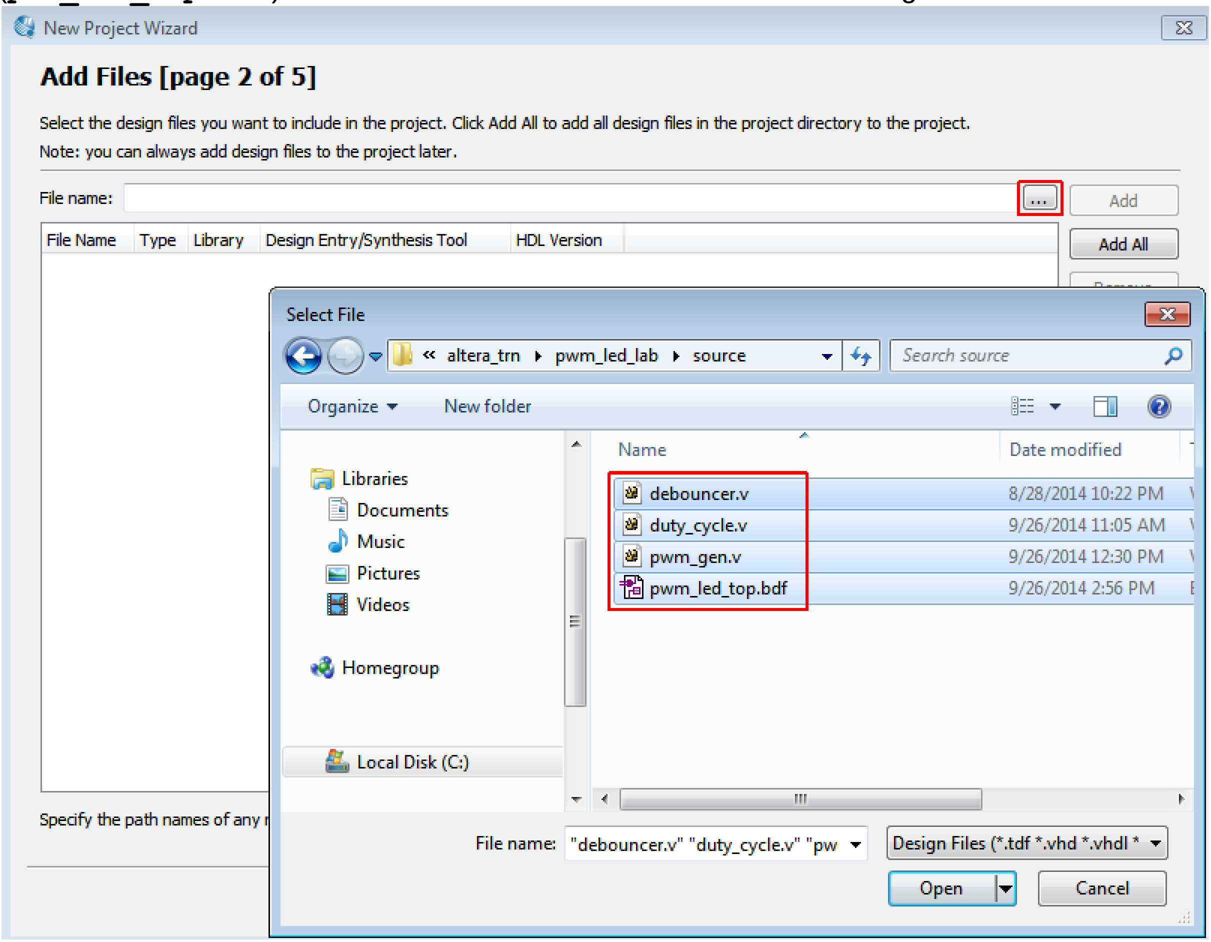


3) Browse to the path where you extracted the lab files: **`c:\altera_trn\pwm_led_lab\`**
Specify the name of the project: **`pwm_led`**
Specify the name of the top-level design entity: **`pwm_led_top`**
(It is a common naming convention to include the word "top" in the top-level design entity to make it clear and obvious which entity is at the top of the hierarchy.)

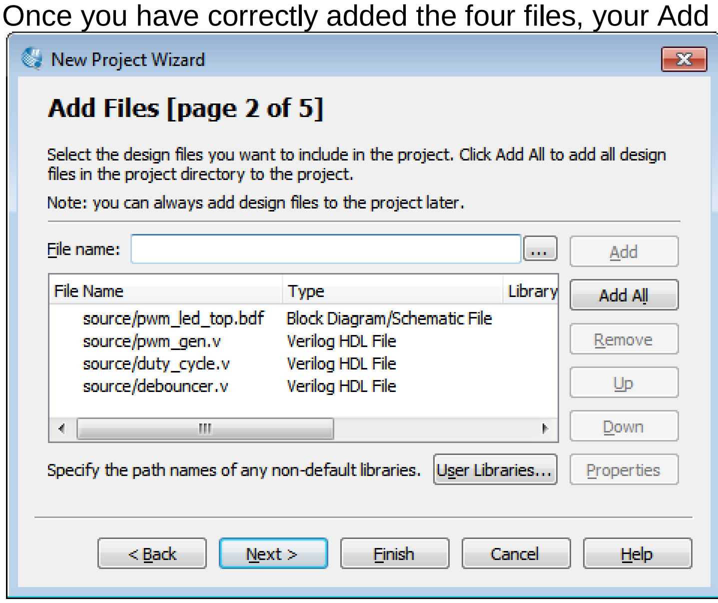


4) Click [Next >] to reach the New Project Wizard: Add Files [page 2 of 5] and click on the [...] button and browse into the **`source`** directory where you will locate the three provided Verilog files (**`debouncer.v`**, **`duty_cycle.v`** and **`pwm_gen.v`**) and the provided schematic file

(**`pwm_led_top.bdf`**). Select all 4 files and Add them to the files listing.

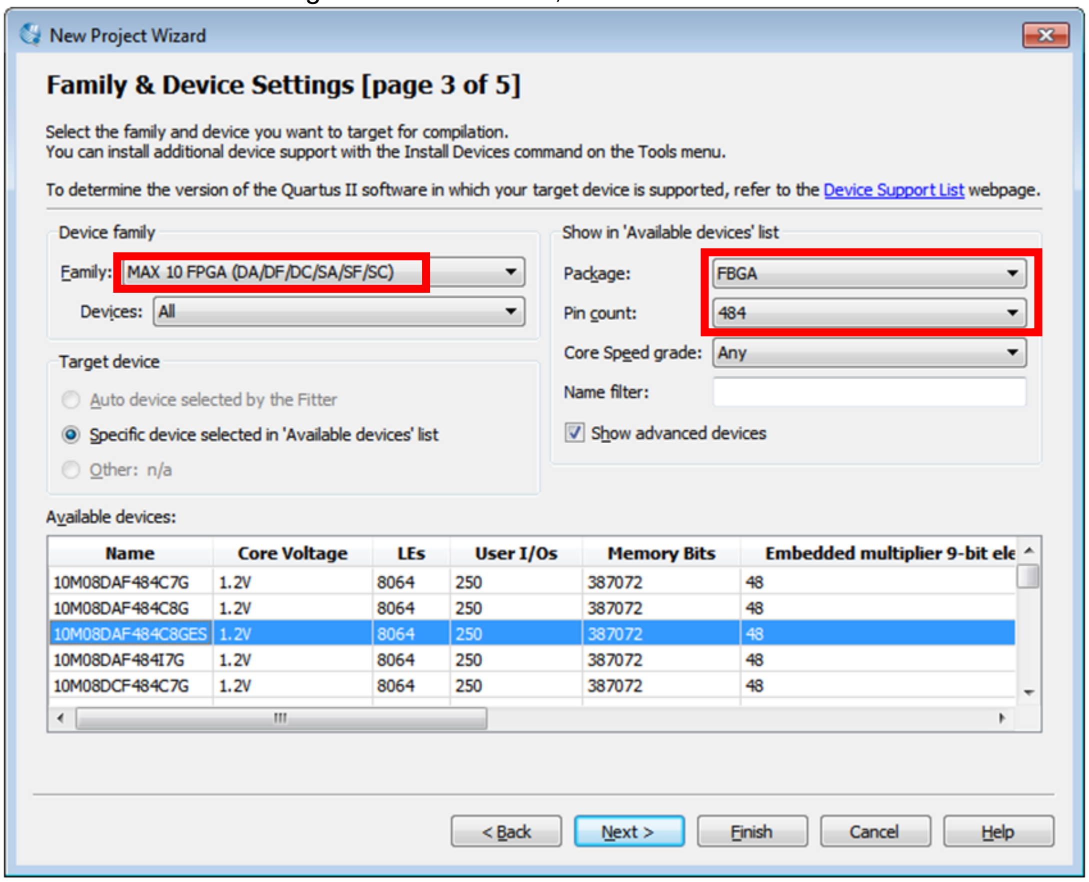


5) Once you have correctly added the four files, your Add Files dialogue box should look as follows:

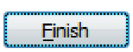


6) Click Next > to reach: New Project Wizard: Family & Device Setting [page 3 of 5], select the device on your kit by using the filters as shown below. Start with **Family**, then **Package**, then **Pin count** to filter the selection of available devices so that you can select the device on your kit. If using the

BeMicro MAX 10, select the **10M08DAF484C8GES**. If using the BeMicro CV, select the **EP5CEA2F23I7**. If using the BeMicro SDK, select the **EP4CE22F17C7**.

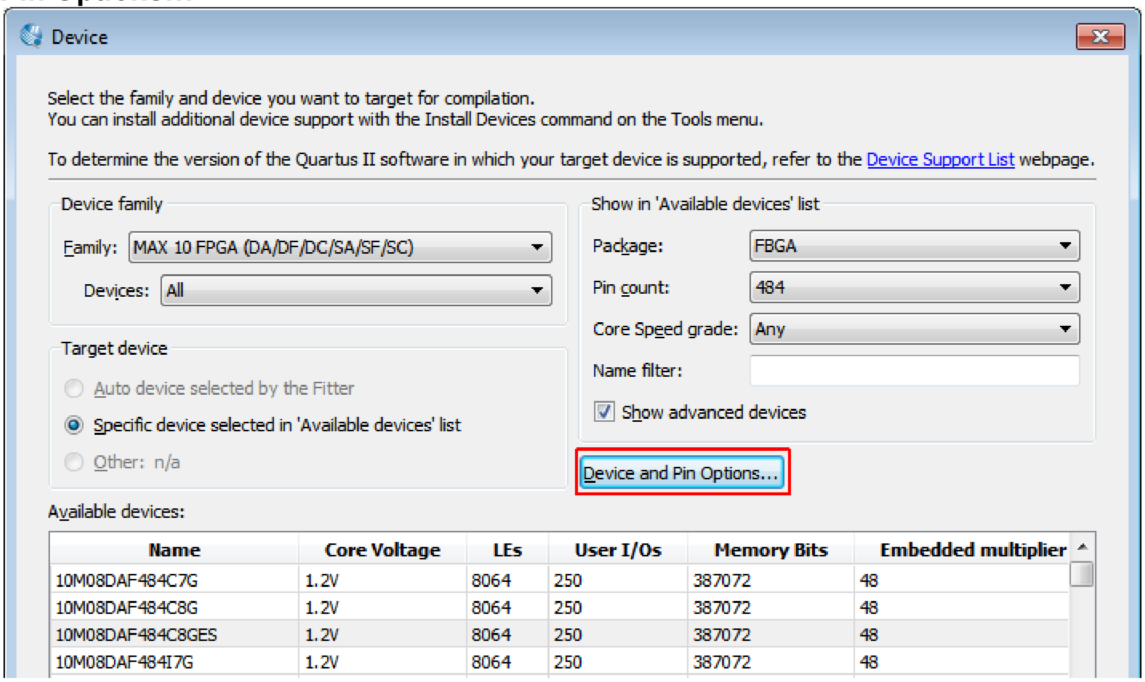


After making your selection, look at your kit and confirm that the part number marked on your device matches your selection.

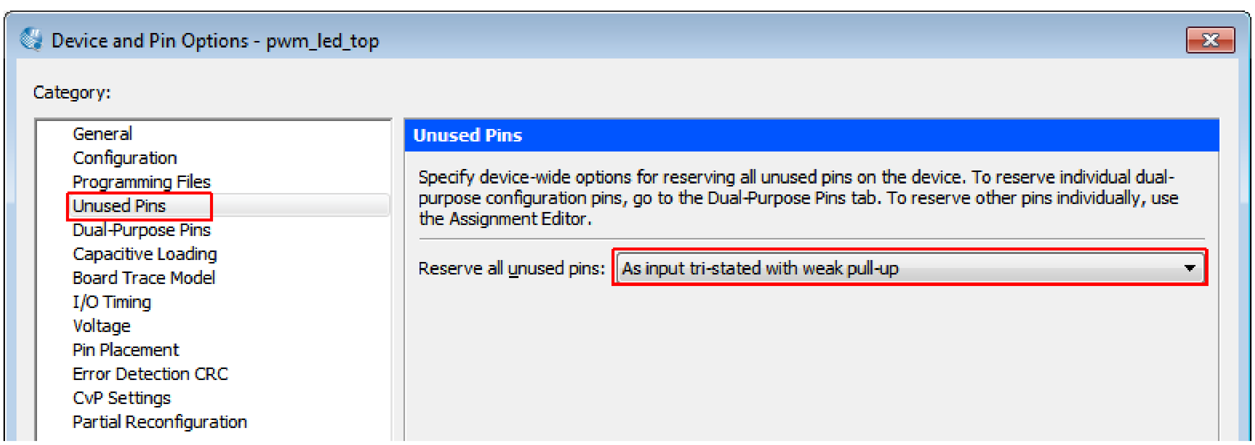
7) Click 

An option in the Quartus project is regarding how to deal with any user I/O pins that are unused by our design.

1) Go to the **Assignments** Menu and choose **Device**, then in the Device dialog box, select **Device and Pin Options…**



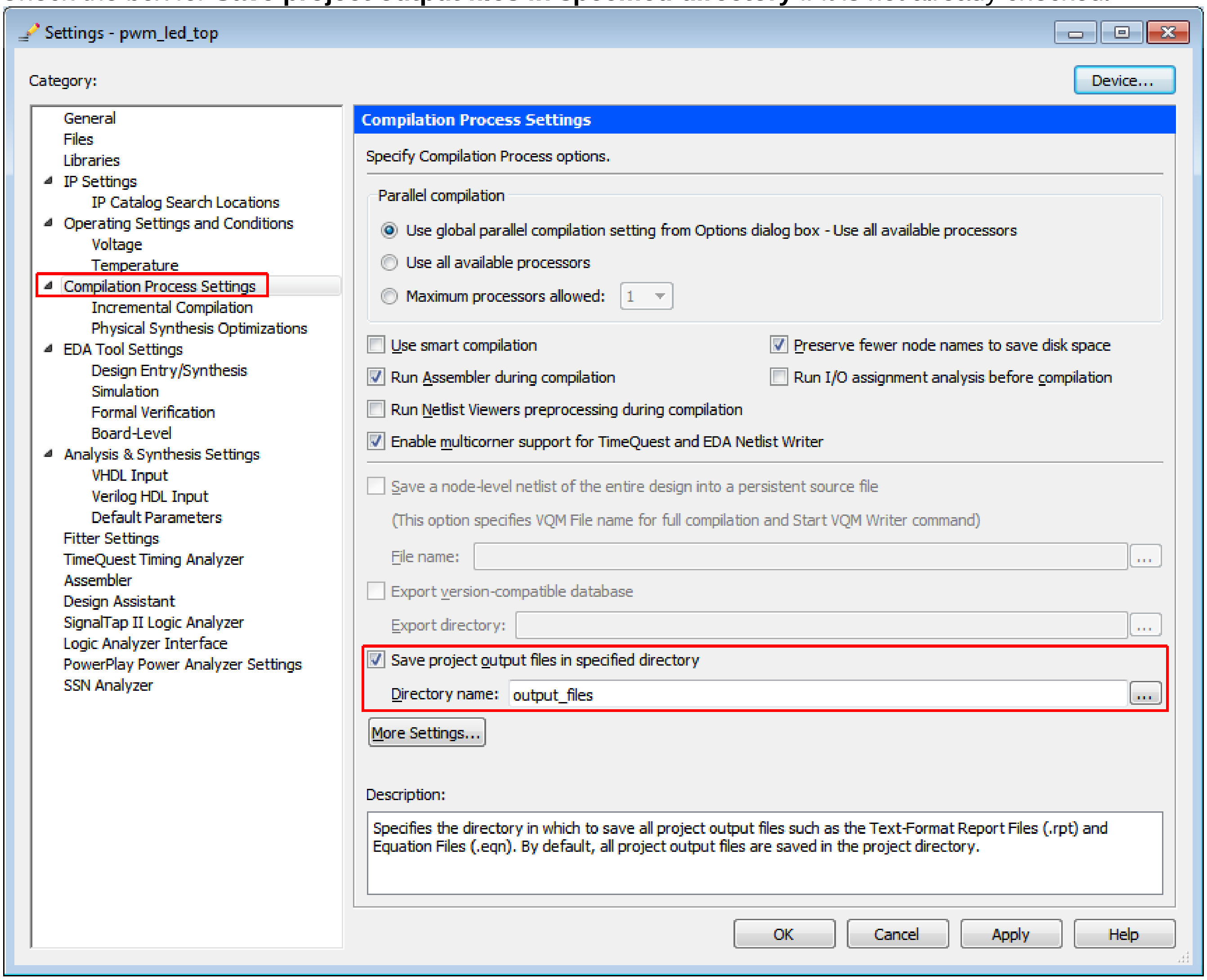
2) This brings up the Device and Pin Options dialog box. Select **Unused Pins** and use the pulldown menu to change the option to Reserve all unused pins **As input tri-stated** and then click **OK**:



Another useful option in the Quartus project is regarding how to deal with the many output files that a project compilation will generate. It can be useful to have Quartus

1. Go to the **Assignments** Menu and choose **Settings**, then in the Settings dialog box, select **Compilation Process Settings**

2. Check the box for **Save project output files in specified directory** if it is not already checked.

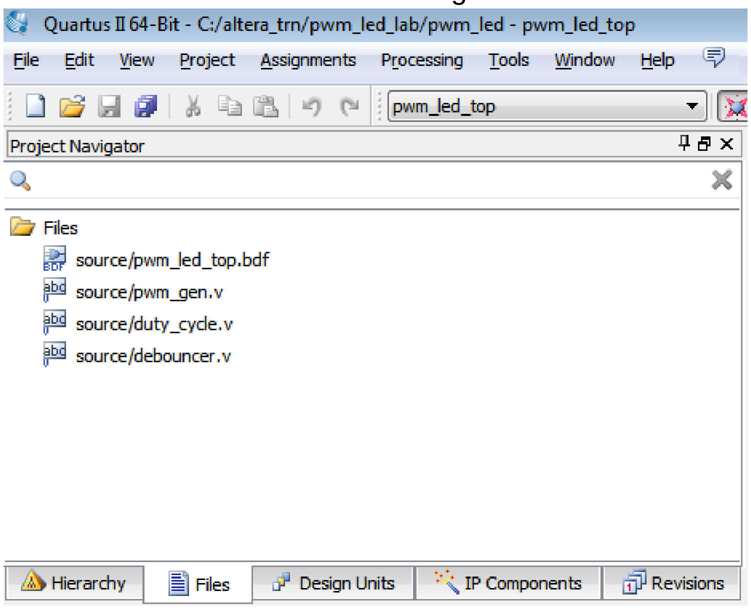


Congratulations, you've just completed setup of the Quartus II Project.

2.3 Examine the Design Files

In the top left corner of Quartus, there is a **Project Navigator** pane, which has several tabs, including Hierarchy, Files, Design Units, IP Components and Revisions.

1. Click on the Files tab to see a listing of the files that we have added to our project.



2. Double-click on each of the files to view the contents.

   - **source/pwm_gen.v** – This Verilog file takes a 4-bit duty cycle input and generates a PWM output with 16 intensity levels (6.25% each).
   - **source/duty_cycle.v** – This Verilog file takes up and down inputs and uses them to increment or decrement a 4-bit counter.
   - **source/debouncer.v** – This Verilog file debounces button inputs to product a single output pulse when a button on the kit is pressed.
   - **source/pwm_led_top.bdf** – This is a partially completed schematic will be used to connect the various blocks of the design.

2.4 Create a PLL

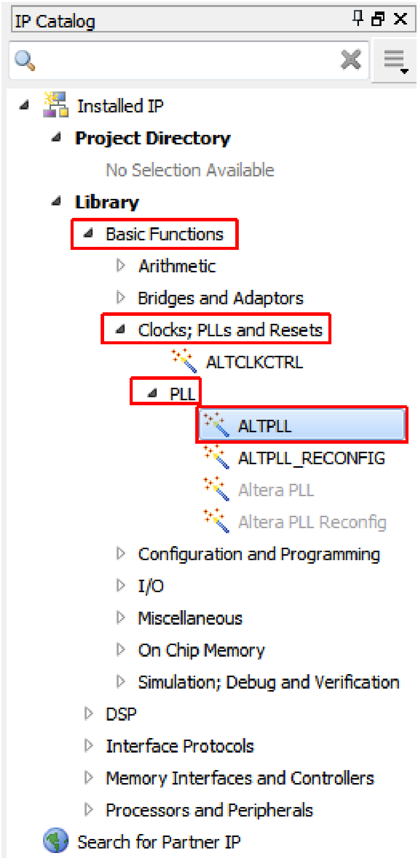
In addition to the above files, one additional item we will use for our design is a PLL. The BeMicro MAX 10 kit has an onboard 50MHz crystal oscillator which comes into the FPGA on of the the MAX 10 FPGA's clock input pins.

MAX 10 FPGAs include PLLs within the device that can be used to generate other clock frequencies. For our design, we will use a PLL to take the 50MHz SYS_CLK input and generate 1MHz and 30kHz clock outputs.

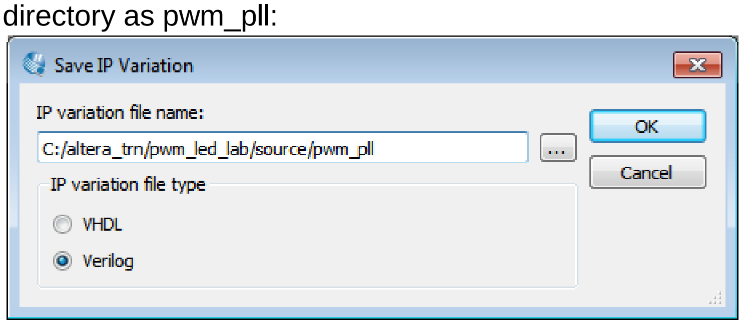
1. From the **Tools** menu, select **IP Catalog**

2. An IP Catalog pane should now be present on the right side of your Quartus window.

3. Locate the ALTPLL IP Component from the IP Catalog. It is found under **Basic Functions** -> **Clocks: PLLs and Resets** -> **PLL** -> **ALTPLL** and double click to launch the IP Component's wizard.

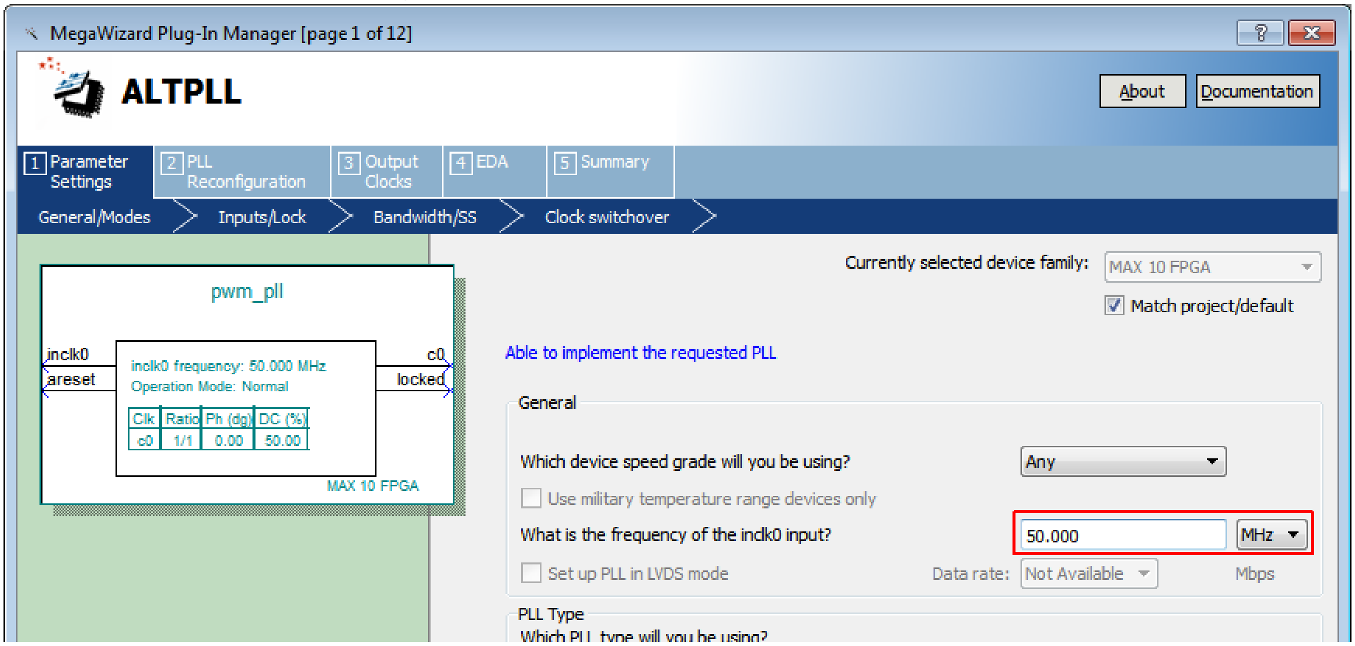


4. A dialog box will appear asking where you wish to save the IP Variation. Save it within the source sub-directory as pwm_pll:

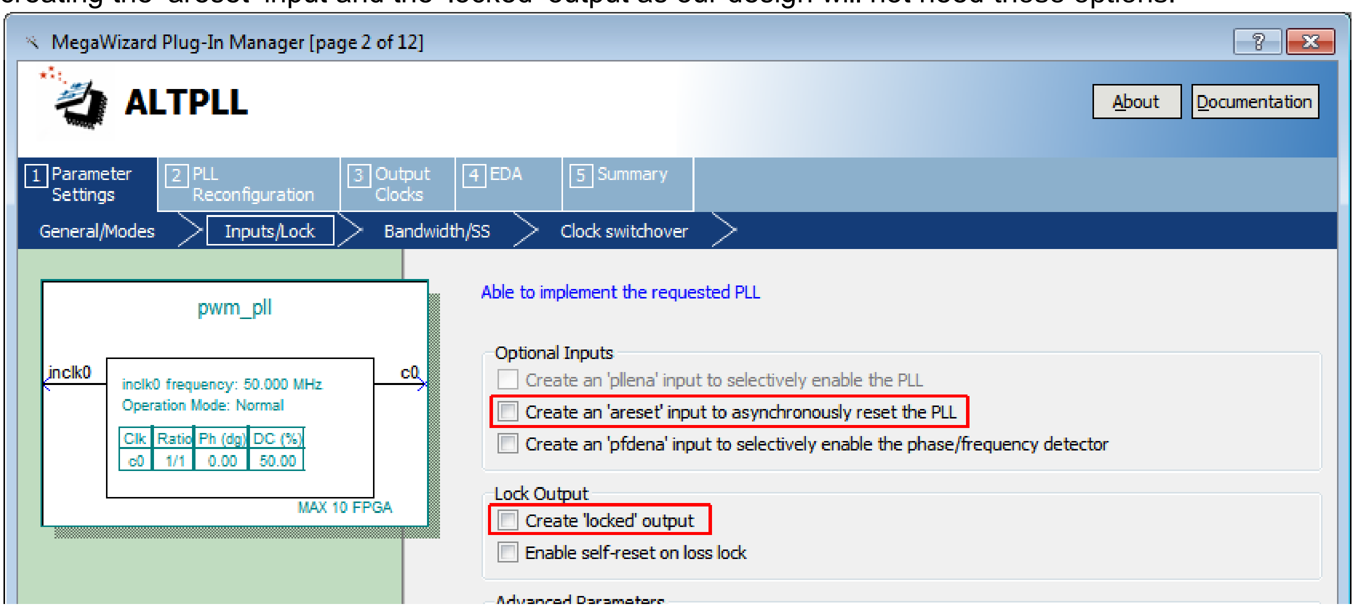


5. The ALPLL MegaWizard will launch. On the first screen of the wizard, change the frequency of the inclk0 input to be 50.000 MHz to match the 50MHz clock input on our kit. Leave the other options on

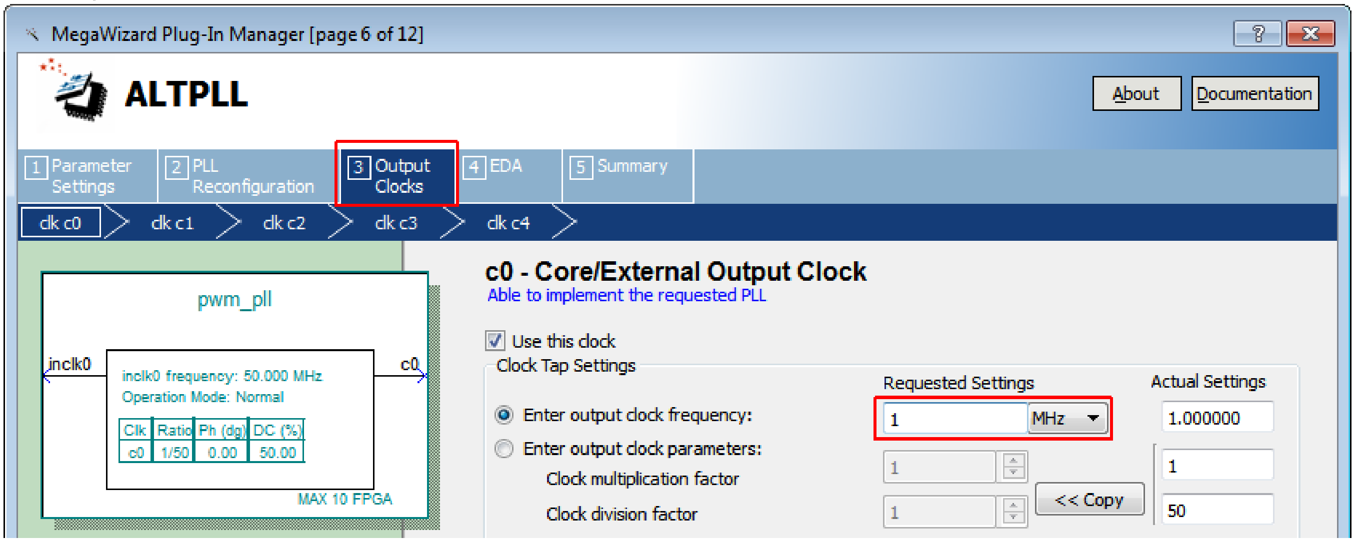
this screen at the default setting.



6. Click **Next** to move to the next page of the MegaWizard "**Inputs/Lock**." Uncheck the options for creating the 'areset' input and the 'locked' output as our design will not need those options.

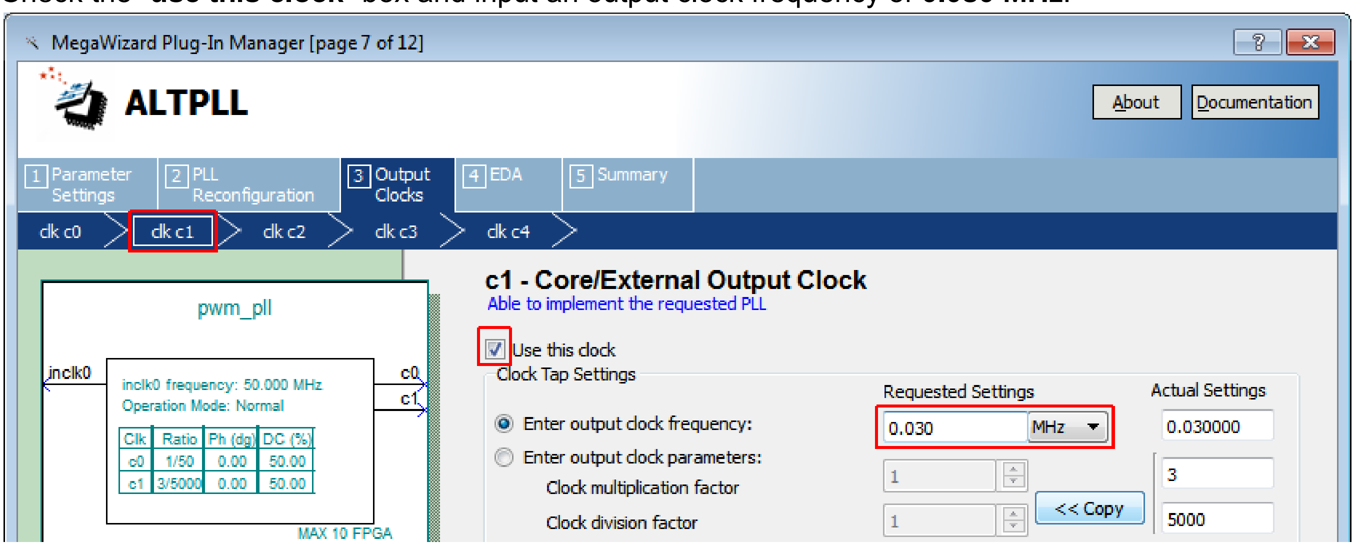


7. Click on the "**[3] Output Clocks**" tab to jump ahead to the MegaWizard page [6 of 12]where we will set our output clock.



8. Click on the "**clk c1**" tab so that we can enable a 2nd output of the PLL to generate a slower 30kHz clock output.

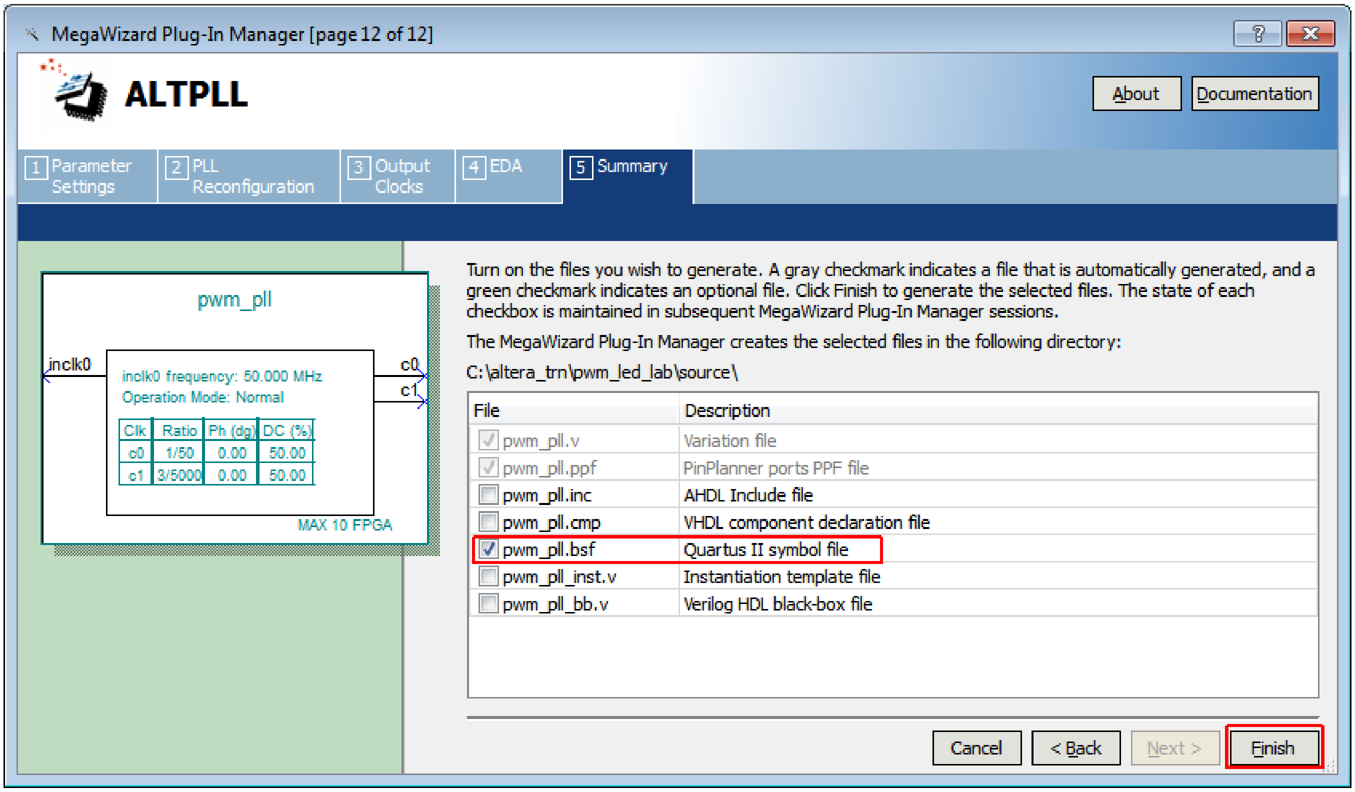
9. Check the "**use this clock**" box and input an output clock frequency of **0.030 MHz**.



10. Click **Finish** which will take you to the **Summary** tab.

11. On the **Summary** tab, select the **pwm_pll.bsf** to have the Wizard generate a **Quartus II symbol file** so that we can place the PLL within a schematic sheet and click **Finish** to complete the PLL

MegaWizard.



12. The PLL MegaWizard generated an IP variation file which is now located in your source subdirectory. Also, a Quartus IP file with file extension of *.qip has also been automatically added to your Quartus project. You should now see the **source/pwm_pll.qip** file in your Files listing.

The same process of using the IP Catalog could be used in your own FPGA design to create IP blocks such as counters, numerically controlled oscillators, FIR filters, FFTs, and DDR3 controllers, to point out a few examples. Explore the IP Catalog to see other items that can be created.

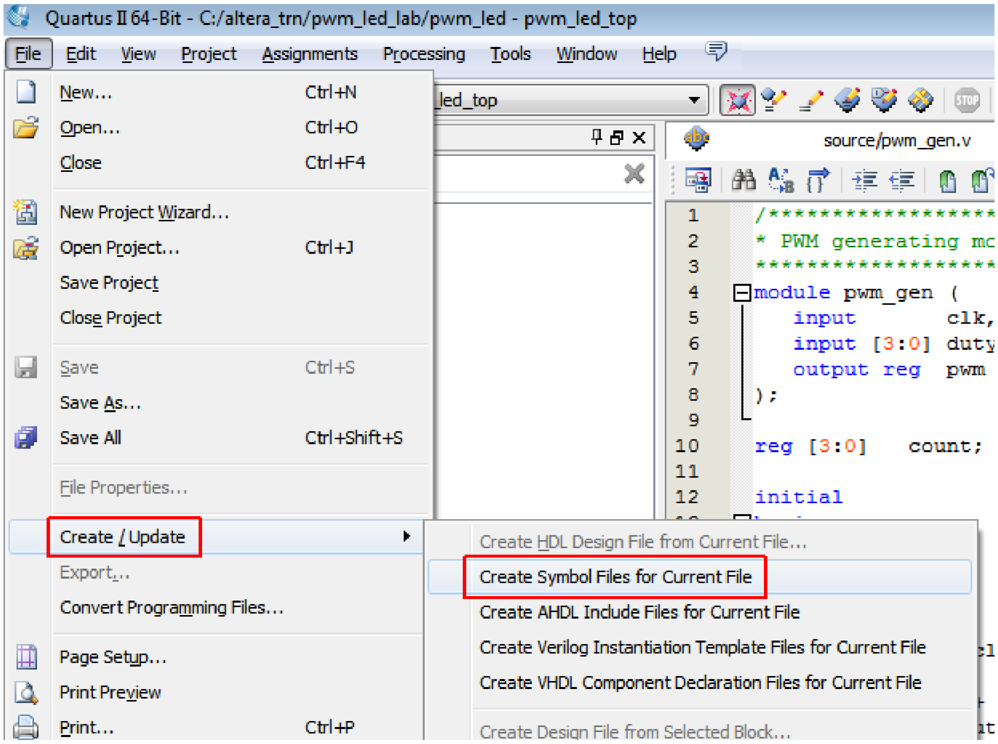
2.5 Create Symbols for the Verilog source files.

The design uses several Verilog files which each define a design entity. The different design entities will need to be connected together to create our FPGA design. While the Verilog design entities can be connected together with Verilog code, another option exists in the Quartus II software. It is possible to create symbols for Verilog files and then the design entities can be placed onto a Quartus II block diagram file and can be connected which need to be connected together using the Quartus II schematic editor.

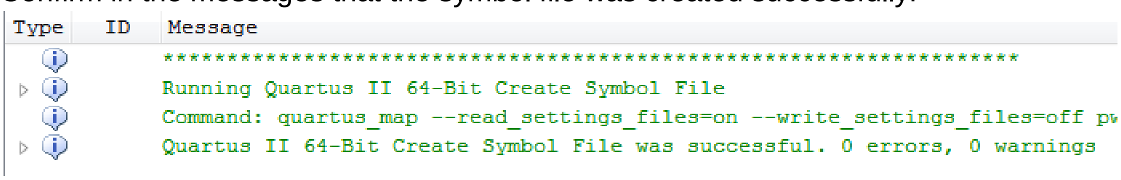
1. For each of the 3 Verilog files in the project, perform the following:

    a. Open the Verilog source file.

b. From the **File** menu, select **Create / Update** -> **Create Symbol Files for Current File**



c. Confirm in the messages that the symbol file was created successfully:



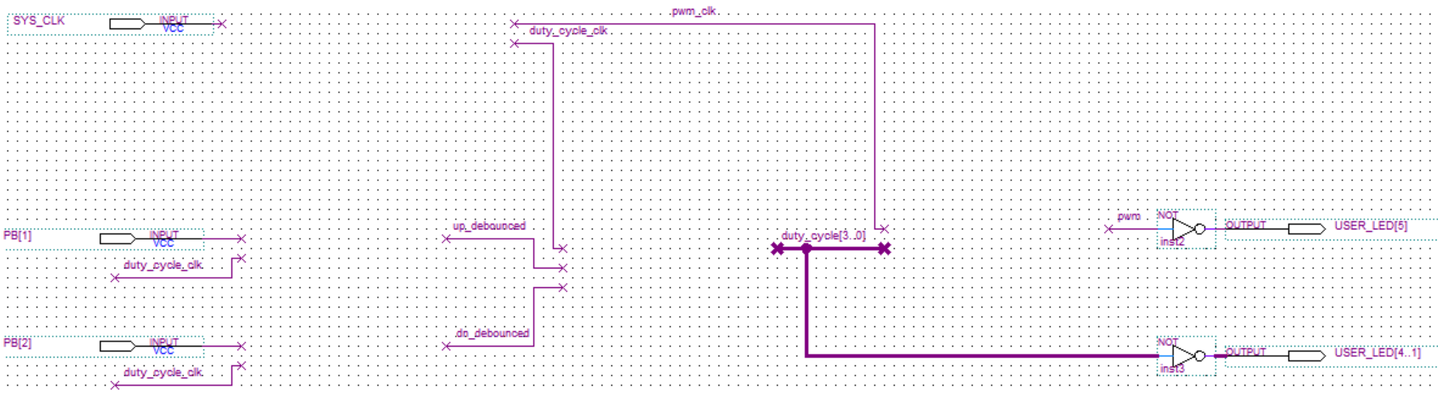
Then repeat these steps to create the symbol for the other Verilog files.

2.6 Complete the Schematic

1. Open the **source/pwm_led_top.bdf** file and the partially completed schematic should appear in the Quartus window.

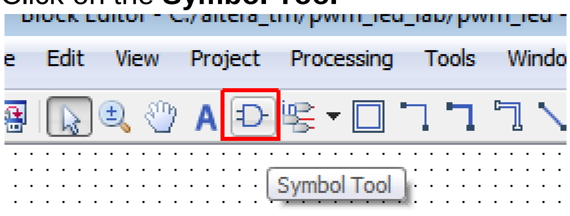
2. You may find it useful to right click on the schematic tab and select "**Detach Window**" to allow the schematic to open in a separate window that can be maximized to see the entire schematic.

3. The partially-completed schematic should appear as follows:



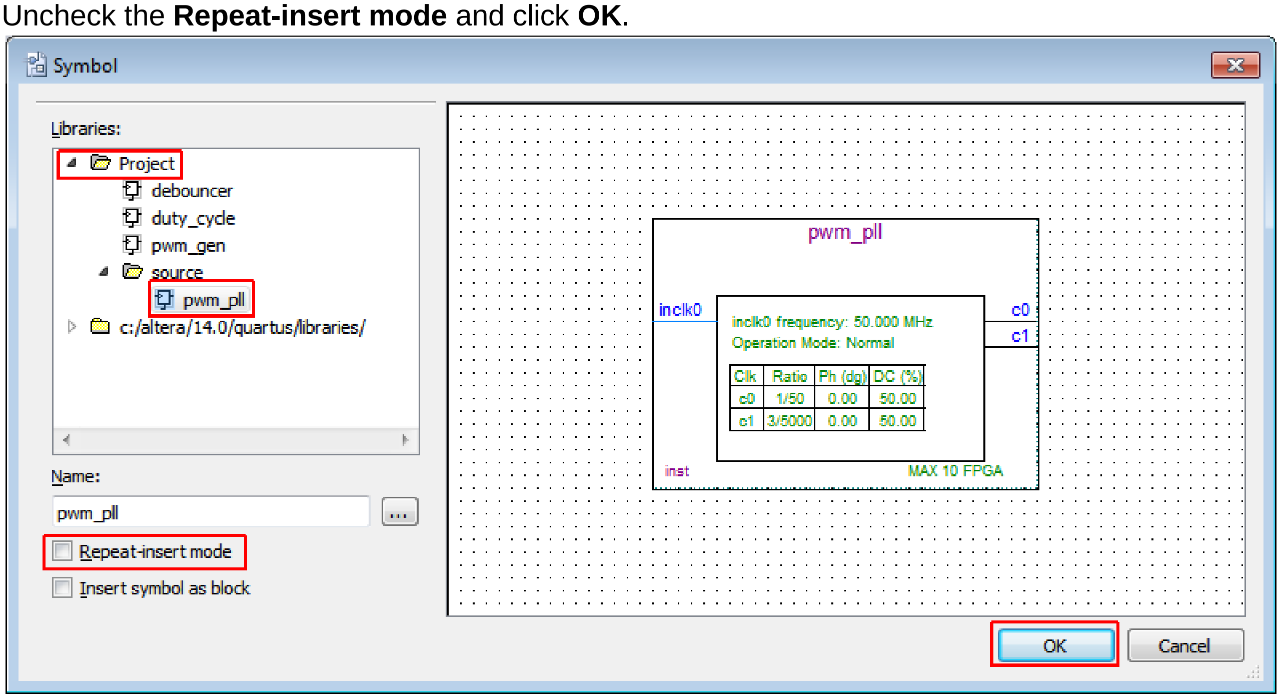
4. Our goal is to insert the symbols for the various blocks in the design to create a completed schematic which will look as follows:

5. Click on the **Symbol Tool**

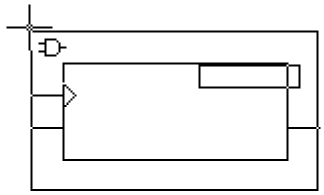


6. Under the **Project** library, you will find the symbols that we have generated previously in this tutorial.. Select the **pwm_pll** symbol.

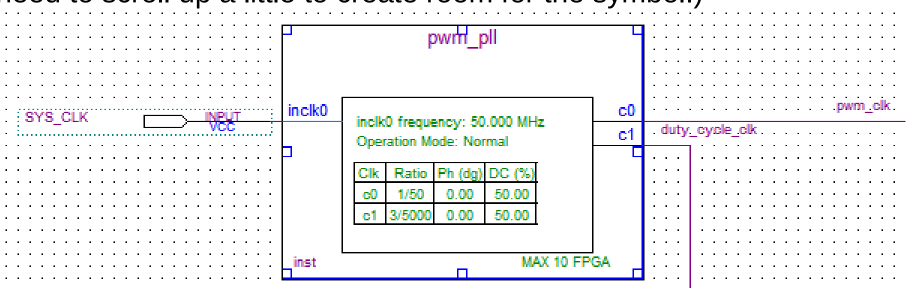
7. Uncheck the **Repeat-insert mode** and click **OK**.



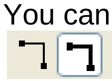
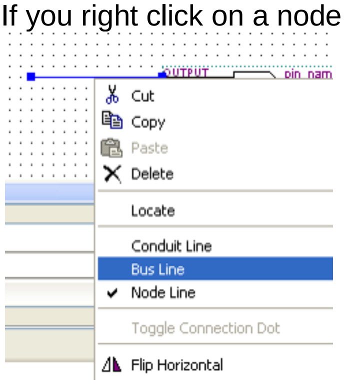
8. You will now have a floating symbol that can be placed onto the schematic sheet.



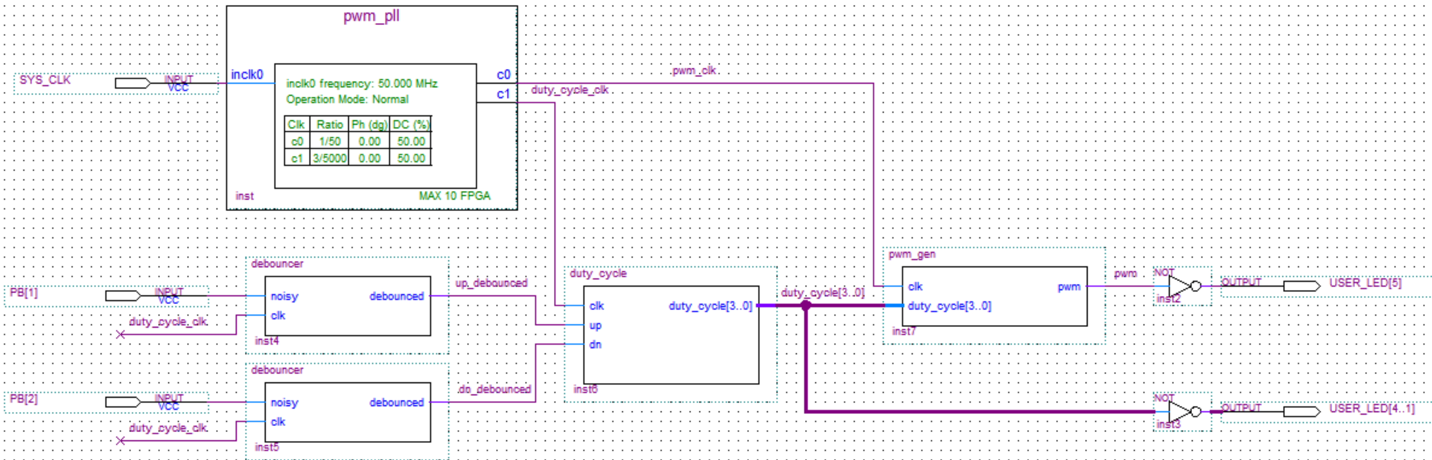
9. Place it in the correct location at the top left such that the **SYS_CLK** input pin lines up with the **inclk0** port and the **c0** and **c1** ports line up with the **pwm_clk** and **duty_cycle_clk** wires as follows. (You may need to scroll up a little to create room for the symbol.)



Below are a few tips to related to schematic entry:

- Note that Quartus schematic sheets use square brackets `[]` and two periods `..` to denote bus notation. E.g. `duty_cycle[3..0]`
- In the symbol selection dialog box, if you know the name of the symbol that you need (e.g. “input”, “not”, “and2”, “and3”, etc.) you can type it into the “Name” field, rather than to navigate the library tree to find it.
- You can find the orthogonal node line and orthogonal bus line icons on the toolbar:
  
- Alternatively, when you hover over a node on a symbol, the cursor will change to the node line drawing icon and you can directly begin drawing even if you do not have the node line tool selected.
- If you right click on a node line, you can change it to a bus line from the context menu:
  

10. Repeat in the same manner to add 2 debouncer symbols, the duty_cycle symbol and the pwm_gen symbol to the schematic sheet. It should now appear as follows:



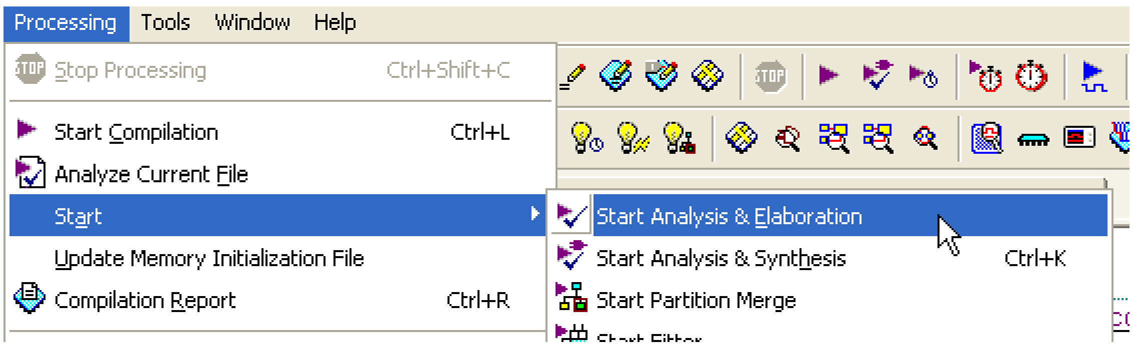
11. Go to the **File** menu and select **Save** to save the changes you have made to the schematic block diagram file.

Congratulations, you've completed your first design file.

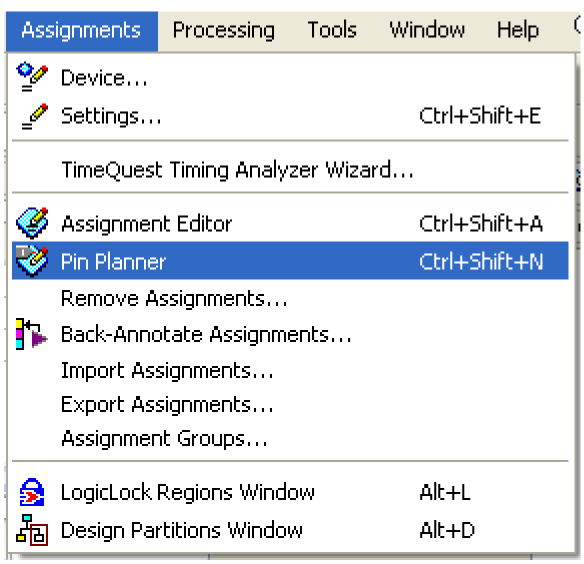
Part D: Analyze the Design and Assign Pins

Now that the design is created, we should verify our syntax to ensure that the ports were connected properly. We also want to build a database of our net, node, and pin names so that we refer to these in other tools with Quartus (ie Pin Planner, Assignment Editor, etc).

1) Run an Analysis and Elaboration (or Analysis & Synthesis)



2) Now we're going to assign pins. There are multiple ways to accomplish the same task. You can either right-click one of your I/O pins and Locate in Pin Planner, or use the menu (**Assignments -> Pin Planner**)



3) Specify the I/O pins in your design to match the development kit
Here is a table of pin locations for the BeMicro MAX 10. Refer to this table to make these assignments:

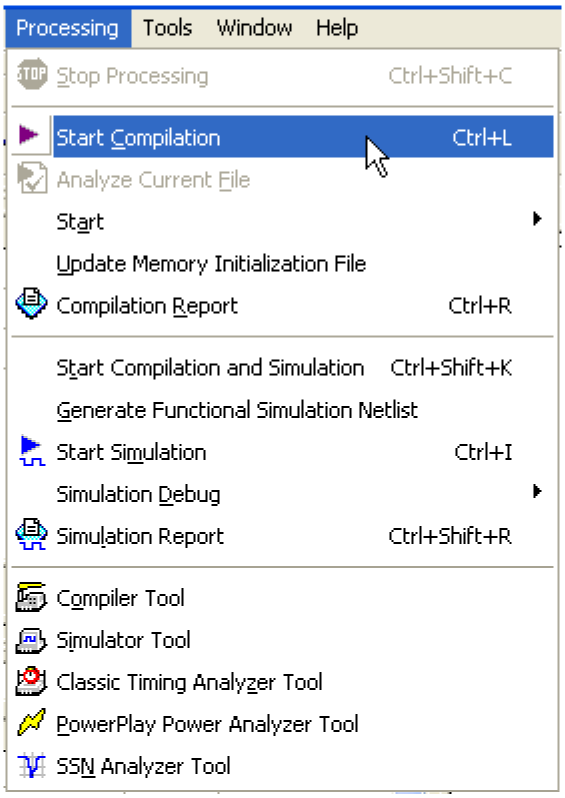
BeMicro MAX 10

Signal Name	***Location***
PB[2]	PIN_R1
PB[1]	PIN_M1
SYS_CLK	PIN_N14
USER_LED[5]	PIN_V4
USER_LED[4]	PIN_T1
USER_LED[3]	PIN_R2
USER_LED[2]	PIN_N1
USER_LED[1]	PIN_M2

Part E: Compile the Design

We are now ready to compile the design.

1) Start the compilation by choosing this icon: , or by selecting **Start Compilation** in the **Processing Menu**:

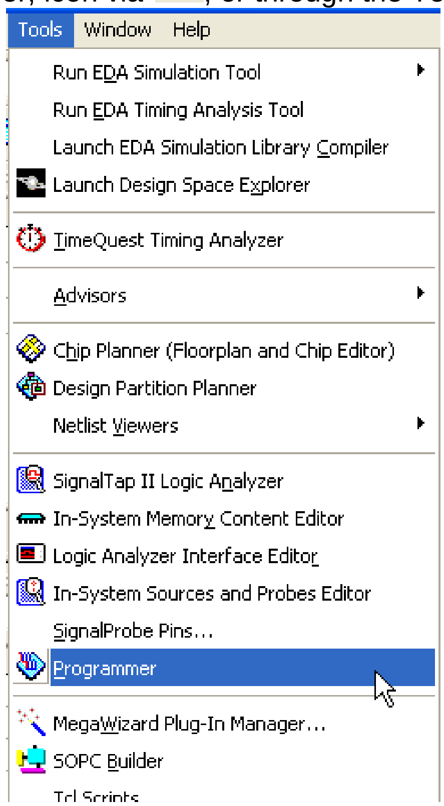


When compilation complete, you can take a look at the Critical Warnings and Warnings. Note that there are tabs at the bottom of the compilation messages pane that allow you to filter by message type.

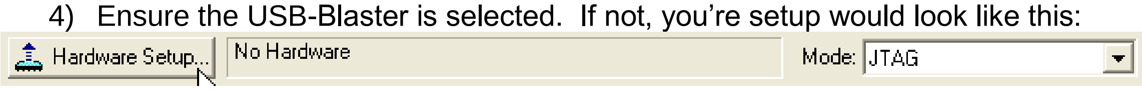
At this point, you will have some Critical Warnings regarding timing analysis and you may have a single Warning regarding skipping of power analysis. If you have any other warnings or errors, please ask for help.

After Compilation, you are able to download to the board.

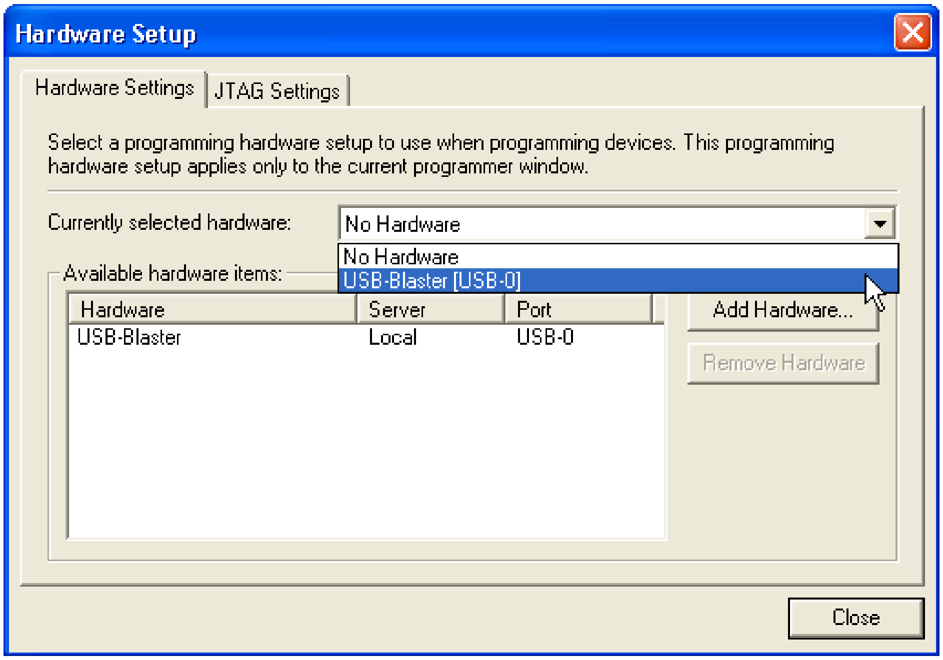
1) Connect the Mini USB cable to your BeMicro MAX 10 kit.
2) Plug the other end of the USB cable to a USB port of your computer.
3) Launch the Quartus II Programmer, icon via , or through the Tools menu (**Tools -> Programmer**)



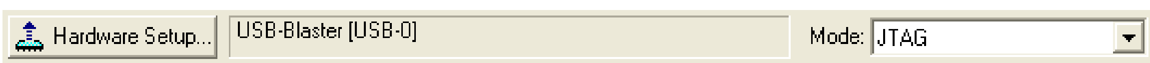
4) Ensure the USB-Blaster is selected. If not, you're setup would look like this:



In this case, Click the Hardware Setup button and select USB-Blaster from the pull-down.



Close the Dialog Box and your setup should appear as this:



Ensure the .sof (programming file) is already listed, and make sure the Program/Configure box is checked:

4) Click 

to begin the download.
5) Play with the board to see if the design behaves as expected.

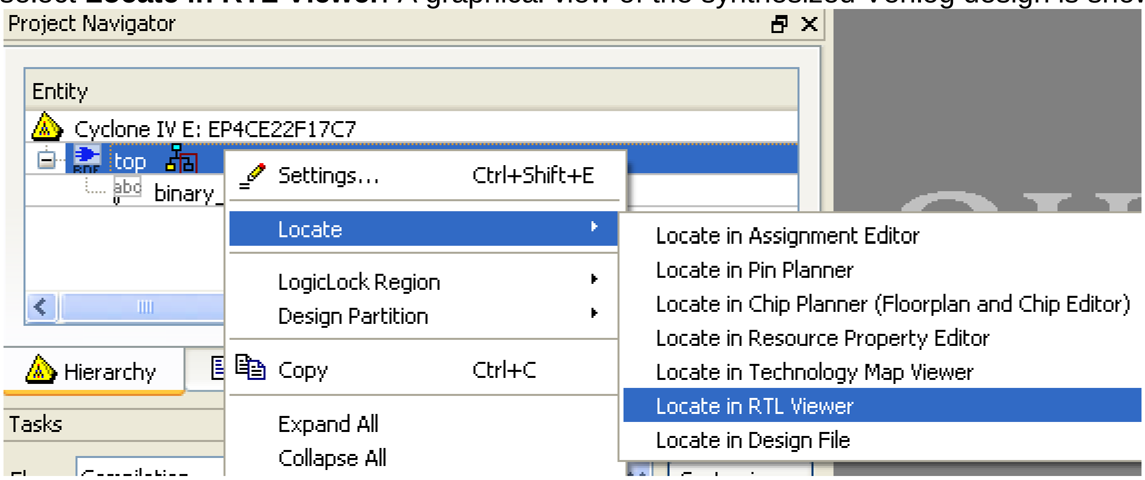
Congratulations, you've completed your first FPGA design

MODULE 3. EXPLORING THE DETAILS OF YOUR DESIGN

Overview: During this exercise, you will learn how to use the Quartus tools to explore the results of your compiled design.

Part A: Explore design with the RTL Viewer

1. In the **Hierarchy** tab of the **Project Navigator** select and right click on **top**, next click on **Locate** and then select **Locate in RTL Viewer**. A graphical view of the synthesized Verilog design is shown.

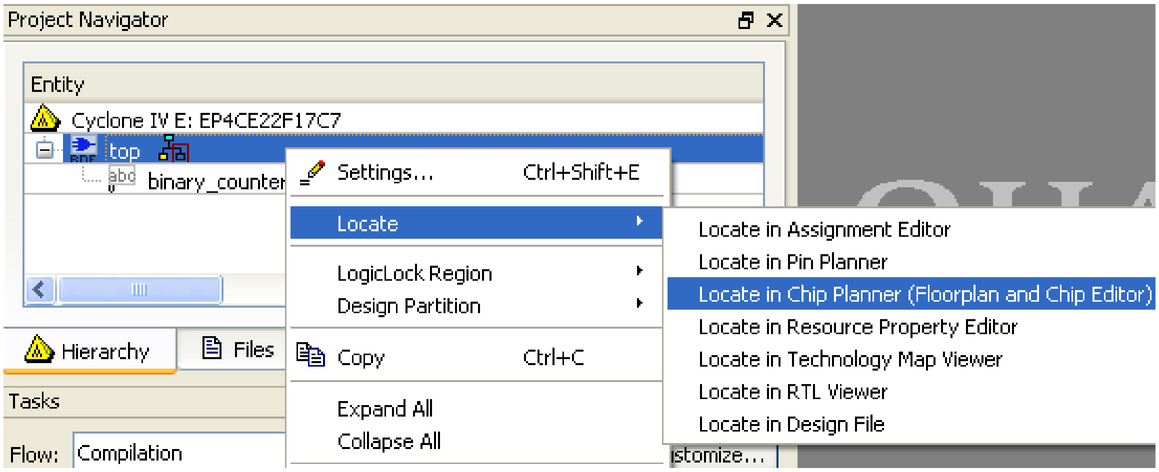


The RTL Viewer is a viewer that shows you a schematic representation of the last successful compliation. The design is shown at the post-Synthesis stage prior to any mapping to physical locations in the chip.

2. Navigate through the hierarchy of the design by double-clicking into some of the blocks.

Part B: Explore design in device with the Chip Planner

1. In the **Hierarchy** tab of the **Project Navigator** select and right click on **top**, next click on **Locate** and then select **Locate in Chip Planner (Floorplan and Chip Editor)**. A graphic view of the device is shown.



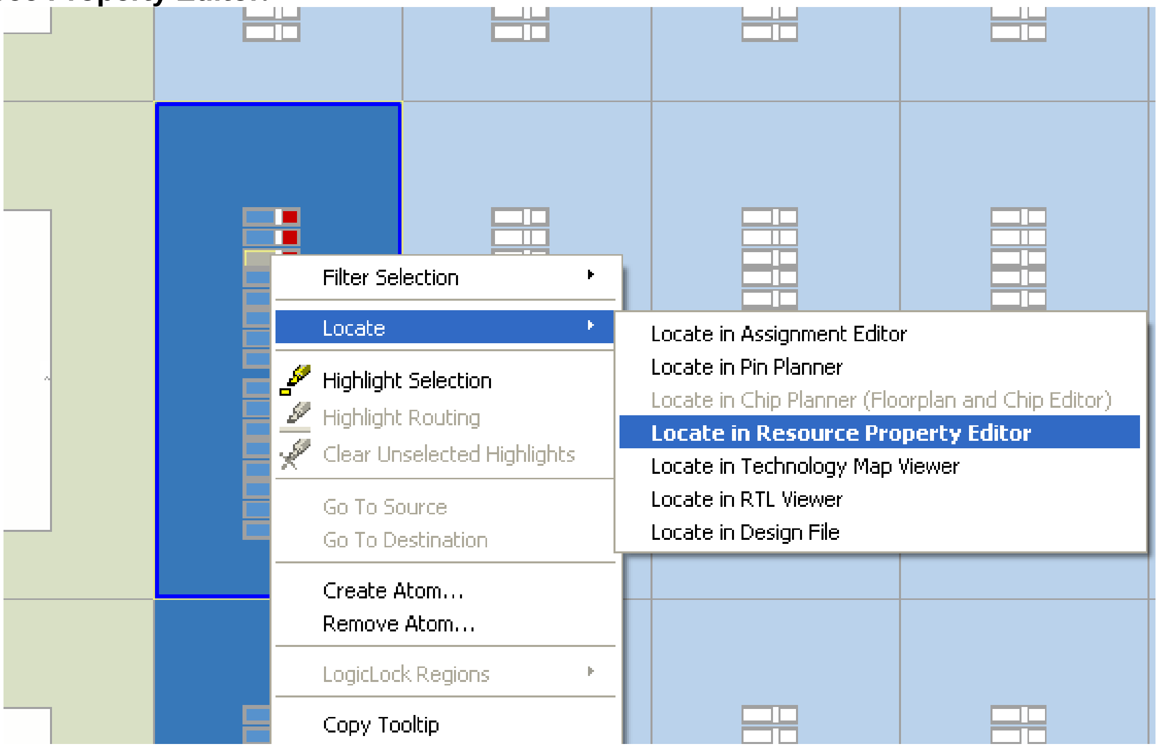
The Chip Planner is a viewer that shows you the results of the last compile along with any user placement assignments you might have made

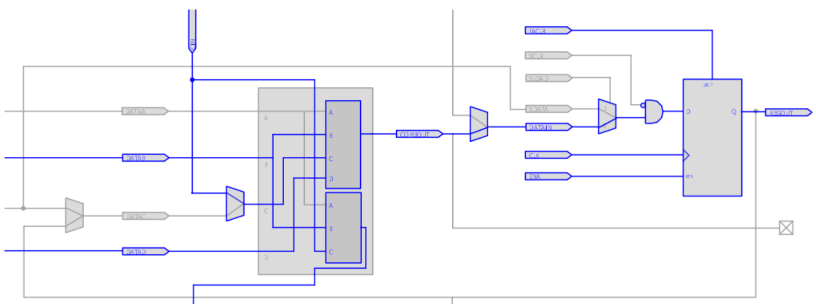
*Quartus II will now highlight a couple cells of the floorplan in a darker blue color. These are the cells in which **top** has been placed.*

2. Can you locate and identify the following in the ChipPlanner View?
   - LABs (the 2 dark blue highlighted squares)
   - Logic Elements (click on the dark blue squares and then press Ctrl-Space several times to zoom in)
   - Memory
   - I/O Pins

Part C: Explore design in device with the Resource Property Editor

1. Select <u>just one</u> of the Logic Elements used in the binary counter. Right-click and select **Locate in Resource Property Editor**.



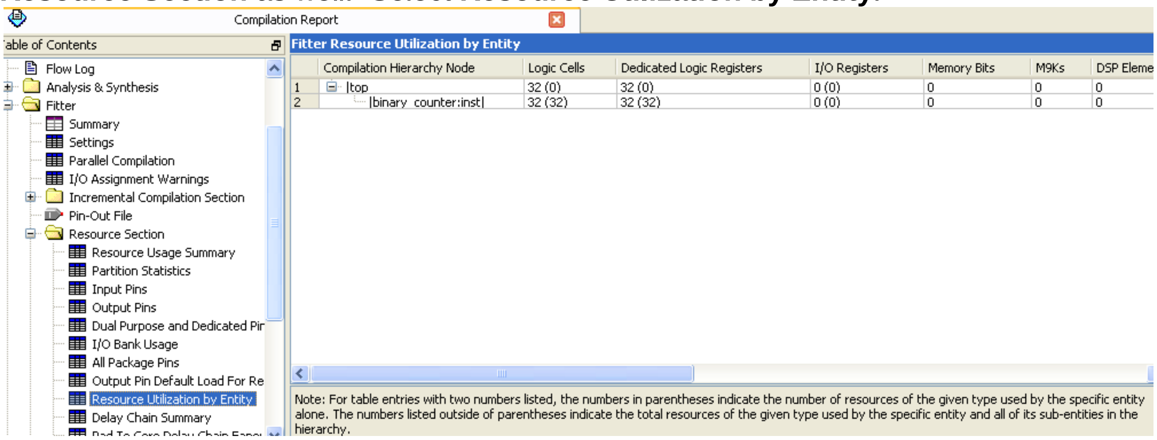


1. Verify that the signal driving the register's ENA enable input is |top|disable~input.
2. Double-click on the "Inverted" field for the ENA input and notice that you could change the value from False to True.
3. Close the Resource Property Editor.

3.1 Part D: Gather information from the Compilation Report

1. From the Compilation Report, determine the number of Logic Elements being used by the design: ____

2. In the Compilation Report, click on the **+** sign next to expand the **Fitter** folder and then expand out the **Resource Section** as well. Select **Resource Utilization by Entity**.



Our design only contains a single entity within our top level, but if we had multiple entities, this is where we could see the number of Logic Cells, Registers, Memory Bits and memory blocks (M9Ks) that each entities are utilizing.

END OF MODULE 3

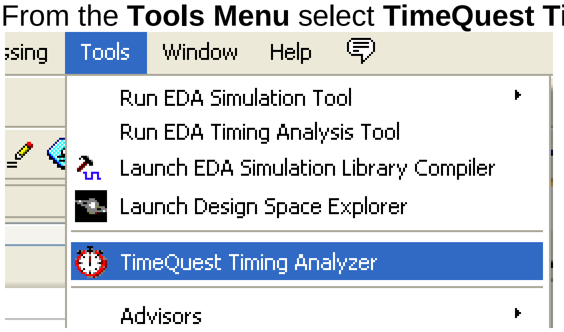
MODULE 4. SETTING UP TIMING CONSTRAINTS

Overview: During this exercise, you will learn how to use TimeQuest to enter basic timing constraints into an SDC file so that Quartus can analyze the timing of your design.

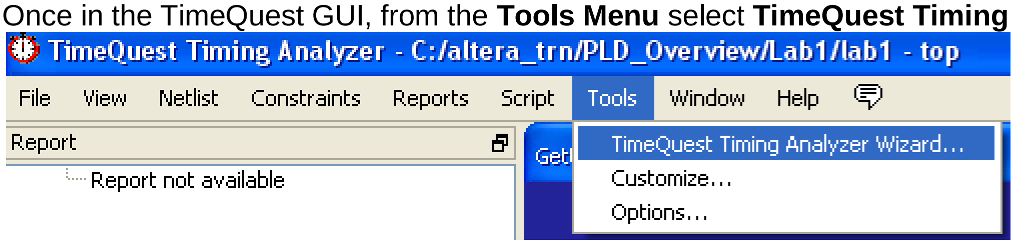
Part A: Launch the TimeQuest Timing Analyzer

We are now ready to compile the design.

1) From the **Tools Menu** select **TimeQuest Timing Analyzer**:



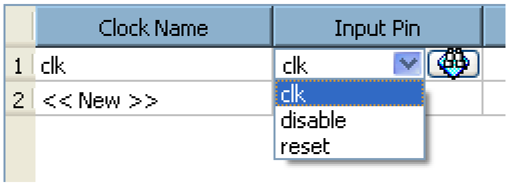
2) Once in the TimeQuest GUI, from the **Tools Menu** select **TimeQuest Timing Analyzer Wizard:**



This wizard will help us create some basic constraints and will place these constraints into a template SDC file that we can use for our project.

3) Click **Next** to go to the **Clock** tab. On line 1 of the table, type “clk” into the Clock Name field.

4) Double click into the Input Pin field to bring up a pulldown menu of input pins from our design. Select the **clk** pin.



5) Enter the clock period into the appropriate field. For the MAX V kit, the clock is 10MHz, so enter **100ns**. For the BeMicroSDK, the clock is 50MHz, so enter **20ns**.

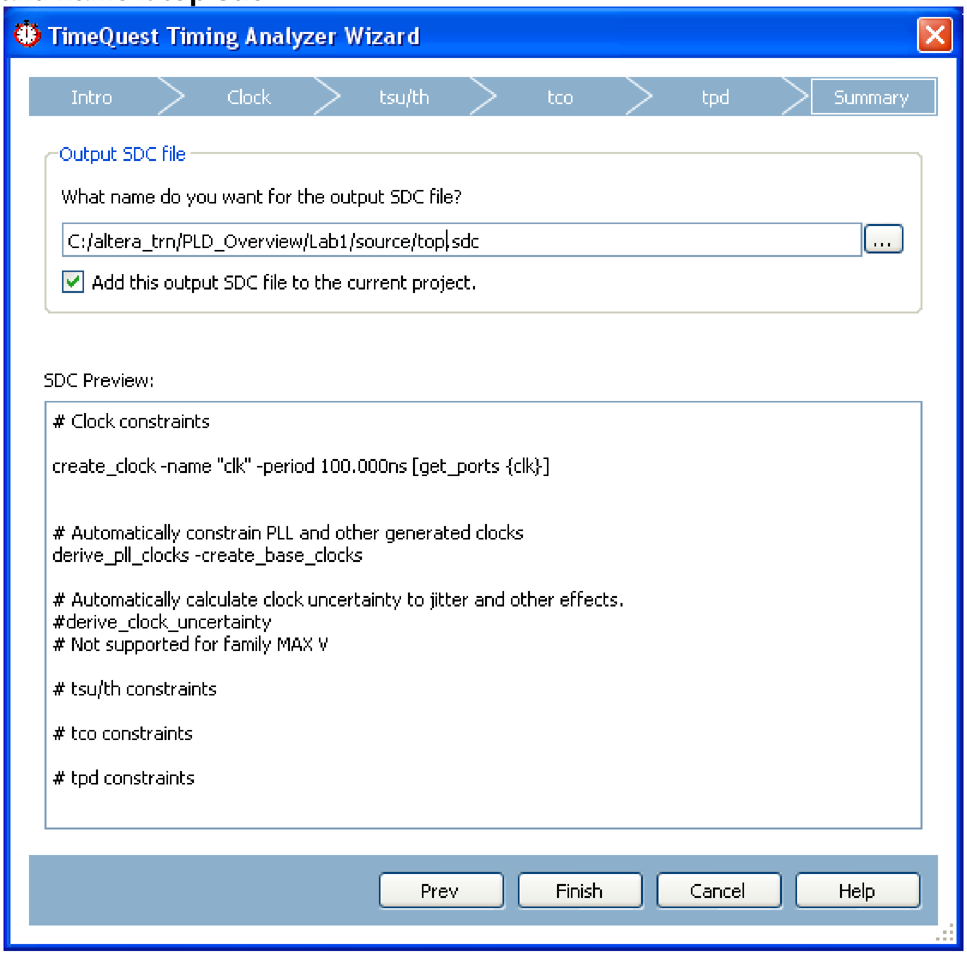
6) The Rising and Falling are used to specify the clock edges for any non-standard duty cycle clocks. For standard 50% duty cycle clocks, these do not need to be specified. For the clock in our design, **leave the Rising and Falling fields blank**.

7) Click **Next** to go to the tsu/th tab. This tab can be used to enter setup and hold constraints on the input pins in the design. In our design, these come from a push-buttons, so setup and hold constraints do not apply, so **leave the tsu/th tab blank**.

8) Click **Next** to go to the tco tab. This tab can be used to enter clock-to-out constraints on the output pins in the design. In our design, the outputs go to LEDs, so tco constraints are not needed, so **leave the tco tab blank**.

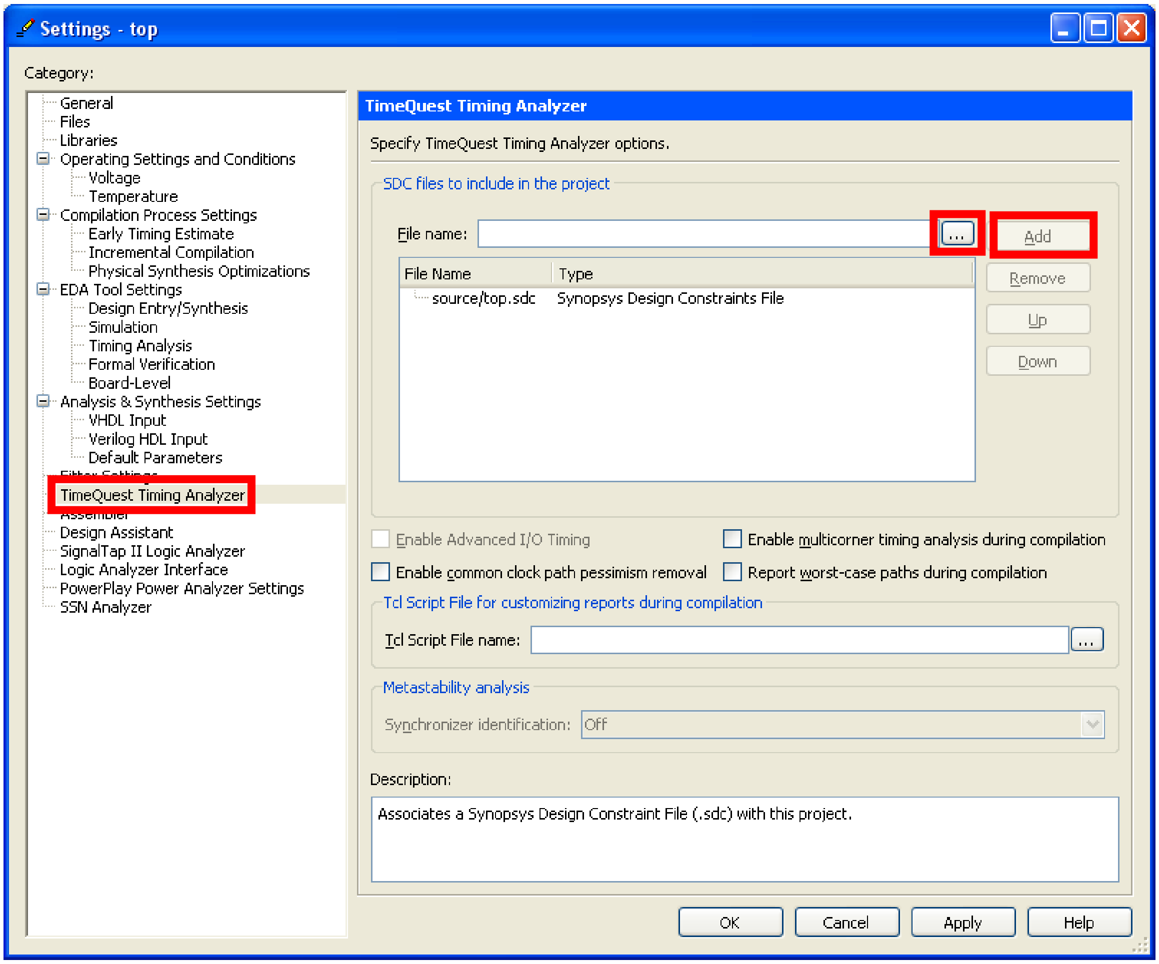
9) Click **Next** and **leave the tpd tab blank** as we do not have any combinatorial logic for which we need to specify constraints.

10) Click **Next** and on the **Summary** tab, change the location of the output file to be in our **source** directory and name it **top.sdc**.



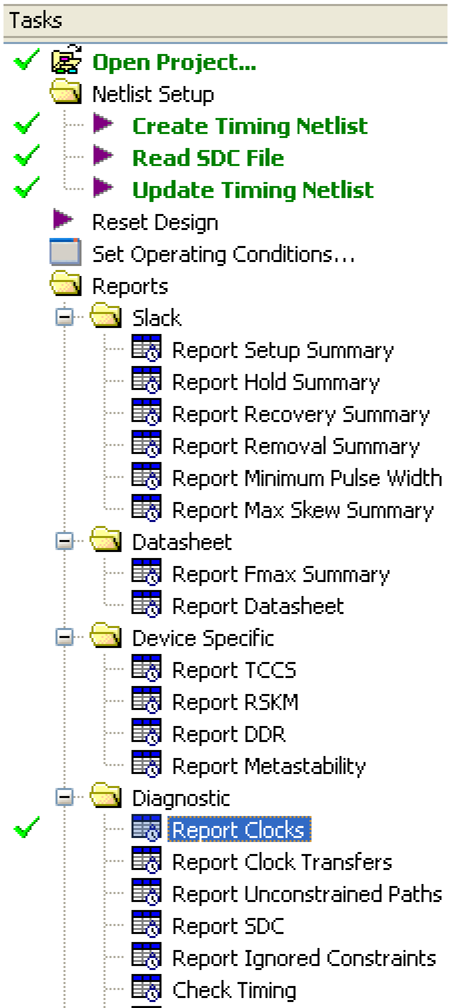
11) From the **Window** menu and choose **Quartus II** to return to the main Quartus window.

12) From the Quartus window, go to the **Assignments** menu and select **Settings** and then click on the **TimeQuest Timing Analyzer** category. Then use the "**…**" and **Add** buttons to add the **source/top.sdc** file to your Quartus project and click **OK**.



13) From the **Window** menu, choose **TimeQuest Timing Analyzer…** to return to the TimeQuest Window.

14) In TimeQuest, go to the **Tasks** pane and double click on **Report Clocks** under the Reports/Diagnostic section.

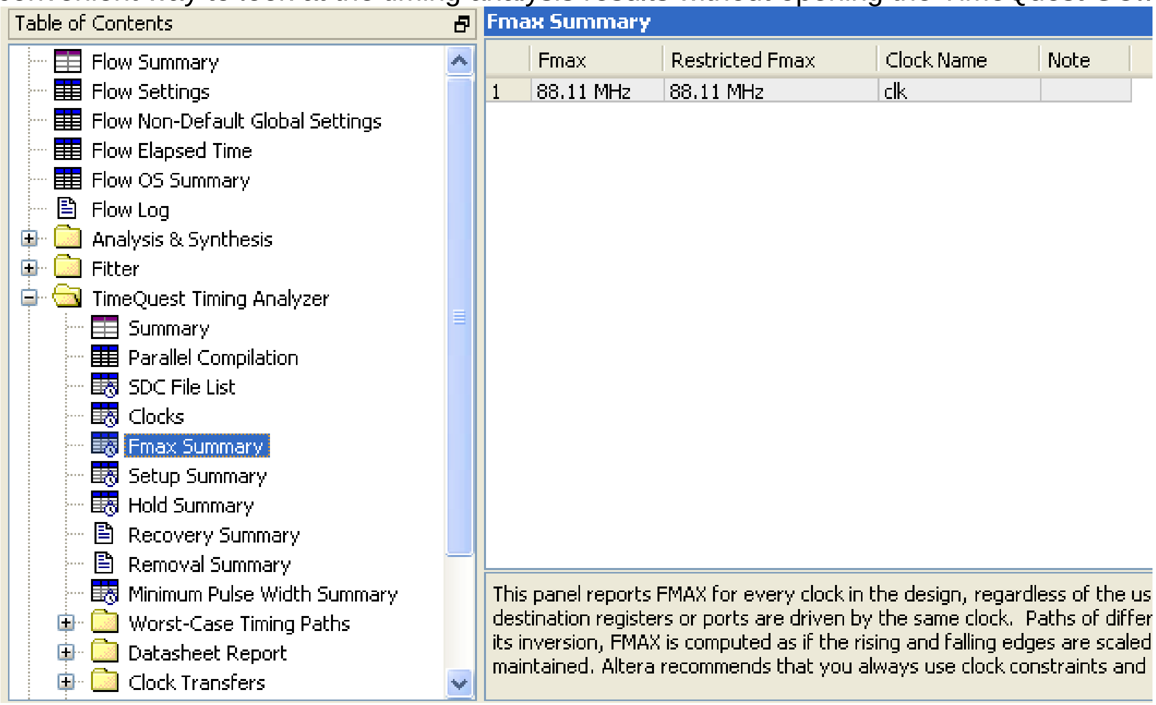


Confirm that your clock is shown with the appropriate clock frequency constraint.

15) Similarly, double click on the **Report Fmax Summary** from the Reports/Datasheet section. This report analyzes the constraints against the last successful compile and indicates the maximum frequency that the compiled design can safely operate at under worst-case conditions.

16) Close the TimeQuest Window to return to the main Quartus II window and perform another full compilation by going to **Processing … Start** Compilation. Now that we have an SDC file we should see that our design compiles without any Critial Warnings. (Note: you may still have 1 warning regarding power analysis.)

17) Note that in the **Compilation Report**, you can view several reports directly, which provides a convenient way to look at the timing analysis results without opening the TimeQuest GUI.



END OF MODULE 4