ECE2504: Introduction to Computer Engineering

ECE3544: Digital Design I

Laboratory Manual

Contents

Suggestions on Project Management

Version: 17 August 2015 (NJD, KLC)

The following items are suggestions to make the completion of your project assignments easier. Many should be "common sense." However, the instructors, lab engineers, and GTAs that work with ECE 2504 have found that repeating them here helps! Good luck. Allow plenty of time to complete your project. Plan for the unexpected….

IDENTIFY AND USE THE RESOURCES THAT ARE AVAILABLE TO YOU

- The project assignment itself.
- Classroom discussions where your instructor may overview the project and answer questions.
- Textbooks and other reference materials.
- CEL and other recommended web pages.
- Instructor's office hours.
- Help from CEL TAs.
- Study groups.

READ! Most of your questions can be answered by reading the materials that are available to you.

- Read the entire assignment before beginning your design.
- Read the entire assignment – again!
- Read the project validation page.
- Read your class notes, your textbooks, the 2504 CEL web pages.
- Read the supplementary documents that are referenced in the assignment or mentioned in class.

REQUIREMENTS: What do you need to obtain in order to do the project?

- Datasheets?
- Schematic diagrams? Printouts?
- Software? Project files?
- Parts?
- Wire? Chips?
- Reports? Style?

TASK SEQUENCING: How should you sequence the completion of the project?

- Have you split the project into clear objectives?
- Work in stages. Decompose the problem into several tasks, and make sure one thing works before starting another.
- Complete your design. Simulate your design. Implement it only after verifying that it works in simulation.
- Test and debug your project.
- Write the report. Address validation difficulties (if any) in the report! (If you cannot get the project to work properly, explain why and what does work.
- Follow the submission instructions carefully. The work you submit is the work that will be graded.

TIMELINE: Failure to manage your available time is the single biggest problem in completing your projects

- How many days or hours until the due date?
- Start your projects early.
- Expect problems to occur and allow time to resolve them.
- Do you have other concurrent projects or tests that impact the completion of this project?

- Budget your time so as to get the project done at least 2 days before the due date – to allow for unexpected events.
- Many projects take 4 times the amount of time that you have budgeted.
- In half the budgeted time, have you finished more than half the project?

CRITICAL ITEMS

- Do you understand the assignment?
- If you are stuck on a question or concept, ask someone NOW!
- The CEL supports multiple sections and hundreds of students. Plan for crowds.
- What if one of your chips is bad?
- Before turning in the project deliverables, make sure you have included everything.
- All items you turn in must be well documented and referenced in the body of your report.
- Follow the submission instructions! Did you submit all the files required?

LAST MINUTE THINKING

- What works? What doesn't work?
- Am I using the same old technique that has failed? What about a fresh approach?
- Can I get partial credit on the validation?
- How can I improve this process so I don't end up here again?
- When should I cut my losses and stop wasting time and CEL resources?

Project Grading Guide

Version: 12 January 2017 (NJD, KLC)

Check your instructor's syllabus, lab assignments, or other handouts for specific report requirements. The following comments provide some general guidelines. Project grades depend not only on what you say, but also how you say it. In general, the graders will check how each project report meets the following criteria:

FORMAT

- The report must be neat. It must be prepared using a word processor program.
- The report should contain a cover sheet, or at the least a title indicating the contents/purpose of the document.
- Most reports will include supporting documentation, such as tables, schematics, block diagrams, or timing diagrams. All of these must be neat and legible.
- All figures and tables must be prepared using the word processor and numbered, labeled, and referenced in the report.
- All circuit and wiring diagrams must be created with Quartus. Note that Quartus circuits can be "cut and pasted" into MS Word documents, as needed.

CONTENT

- Check the rubric to be sure you have included all the requirements in your report. Read all report and submission instructions in the project specification and on the course website.
- The body of the report should contain clear section headings. Usually one section at the beginning will describe the project design requirements and objectives, and a section at the end will summarize the report. The sections in between these two should be meaningful and be used to organize your presentation.
- All descriptions must be logically consistent and easy to understand.
- Assume that the lab report reader/grader is generally knowledgeable about the material taught in the course. Thus, you do not need to define or explain common terminology and procedures.
- Your report should thoroughly discuss the design process that you used to complete the lab project. You should fully document design choices, alternatives, and tradeoffs that you evaluated and why design decisions were made. Documentation such as circuit diagrams, truth tables, Karnaugh maps, and timing diagrams that aid in the understanding of your design should be included in your report and discussed in the text/body of the report. Simulation results should be analyzed.

  A large portion of your project grades will be based on how well you can write a technical report. Expectations include:
  - → Use of complete English sentences though out the report.
  - → A very common error made by students is to mix verb tense usage in the report (or even within individual sentences). This is very poor! In general, use the past tense to describe work that you have already completed. Use present tense to refer to the report contents, the final design, and general methodologies.
  - → Avoid overuse of the "first person" writing style. In technical writing, "I," "me," "we," etc. are generally not considered to be "good style." In particular, the use of "we" is nonsensical for individual projects.
  - → Proofread your report for spelling, grammar, and coherent structure.

VIRGINIA TECH WRITING CENTER

For additional help with writing, you can make an appointment at the Writing Center (http://www.composition.english.vt.edu/writing-center/). Communication is an essential skill, regardless of discipline.

Lab Materials

Version: 12 January 2017 (KLC)

ECE2504

DE0 Nano FPGA board (provided by the ECE department and can be picked up in the CEL.) Please note that your final grade will not be entered until you have returned the Nano board and all borrowed accessories.

ECE 2504 Parts Kit (provided by the ECE department and can be picked up in the CEL.), includes two 7-segment displays, a tiny breadboard, and a wire jumper kit.

Intel FPGA design suite software (freely downloadable, see Installing Software Tools for ECE 2504 and ECE 3544)

ECE3544

DE1 SOC board (provided by the ECE department and can be picked up in the CEL.) Please note that your final grade will not be entered until you have returned the DE1 board and all borrowed accessories.

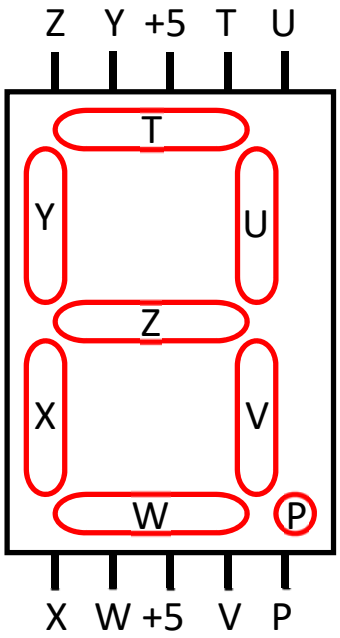
Intel FPGA design suite software (freely downloadable, see Installing Software Tools for ECE 2504 and ECE 3544)

IC Descriptions

Version: 18 August 2015 (KLC, JST)

7-SEGMENT DISPLAY

The TIL-321A display contains eight light emitting diodes (LEDs). (Seven segments plus decimal point.)



There are two types of LED 7-segment displays: common cathode and common anode (CA). In a common anode display, the anodes of all the LEDs are connected together. Power (+5V) should be connected to one of the CA pins.

To light each LED diode, current must be passed in the forward direction through the diode (anode more positive than cathode pin). This takes about a 2.0 volt “drop” across the diode. If the cathode is grounded, there is a sufficient forward bias (voltage drop from anode to cathode) across the diode for current to flow, so the diode lights up. If the cathode is connected to +5V, then there is not sufficient forward bias across the diode for current to flow, so it doesn't light up. Typically, an additional resistor must be added to the circuit to limit the amount of current flowing thru each LED segment.

What does this mean? A given segment of the display is **lit when a logic 0** is applied to its input, and **unlit when a logic 1** is applied to its input.

For more information, read the datasheet on the CEL website.
Note: The MAN6760 datasheet on the CEL website includes three different packages for the 6700 family of 7-seg displays. (It is very common to have a single datasheet for multiple pin outs.) The one you have is shown in drawing C.

Installing Software Tools for ECE 2504 and ECE 3544

Version: 21 August 2018 (KLC)
Installation version: 16.1.0.196 (original install), 16.1.2.203 (after update)

OBJECTIVE

This document will walk through the process of installing the necessary software for ECE 2504 and ECE 3544. The Intel FPGA (formerly Altera) Tools include the following main components, among many other useful tools:

- Quartus 16.1 Prime Lite Edition
- ModelSim Altera Starter Edition 10.5b (used in ECE 3544)

REQUIREMENTS

You will need a computer that runs Windows. The version should not be a problem, and one of the authors of this document uses Windows 10.

For different versions of Quartus, a Linux bundle might be available for download. This document does NOT provide support for installing the Linux version of Quartus. If you choose to install Quartus under Linux, you will have to explore the installation details on your own.

Note: Quartus Version 16 was released in November 2016; the 2504 project files have been tested on this version. v17 has become available more recently, but the 2504 files have not been tested using v17. If you choose to install v17, please be aware that we are not currently supporting it. Most important, do not install multiple versions of Quartus.

SUMMARY OF INSTALLATION STEPS

Step 1: Install Quartus v16.1.0.196

Step 2: Install the USB Blaster II Driver.

Step 3: Update Quartus v16.1.0.196. to v16.1.2.203.

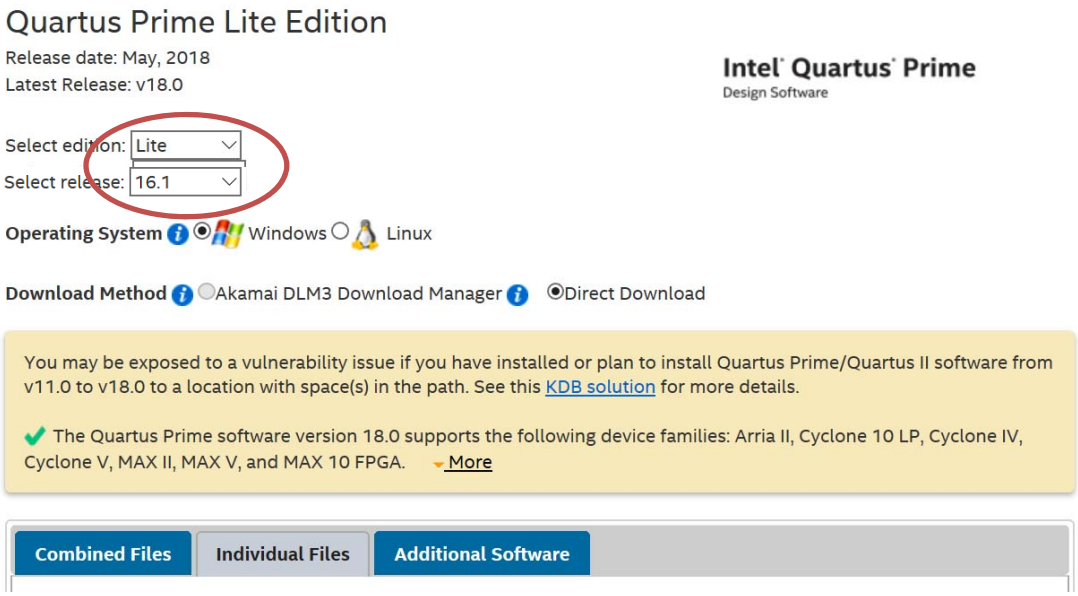
Step 4: Update USB-Blaster Driver.

DETAILED INSTALLATION STEPS

Step 1. Install Quartus Prime Lite Edition v16.1

Go to https://fpgasoftware.intel.com/ and follow the instructions on the following pages. At some point, you will have to create an account, so create something that is memorable to you but not easy to guess. If you have trouble with the link, go to the Intel home page (www.intel.com), click *Support*, then *Intel FPGA Support*, then *Downloads*.

Select the Quartus **Lite** Edition then select Release **16.1** as shown below.



Choose **Windows** as the operating system. You will need to download the files for

- **Quartus Prime (includes Nios EDS)**
- **ModelSim-Intel FPGA Edition**
- **Cyclone IV Device Support**
- **Cyclone V Device Support**

You can do this either using the **Individual Files** tab or the **Combined Files** tab. (Note the Combined Files download gives you more than you need but does it in a single download.)

When you start the download, you will have to log in to the Intel site. Follow the instructions that you find to create an account, if you have not already done so.

Note: If you are having trouble with the download, you may be trying to download the wrong version. You need Quartus 16.1 Prime **Lite** Edition (not the Pro Edition). How to select Lite vs Pro on the Altera website changes somewhat regularly. Look for the phrase "Lite Edition".

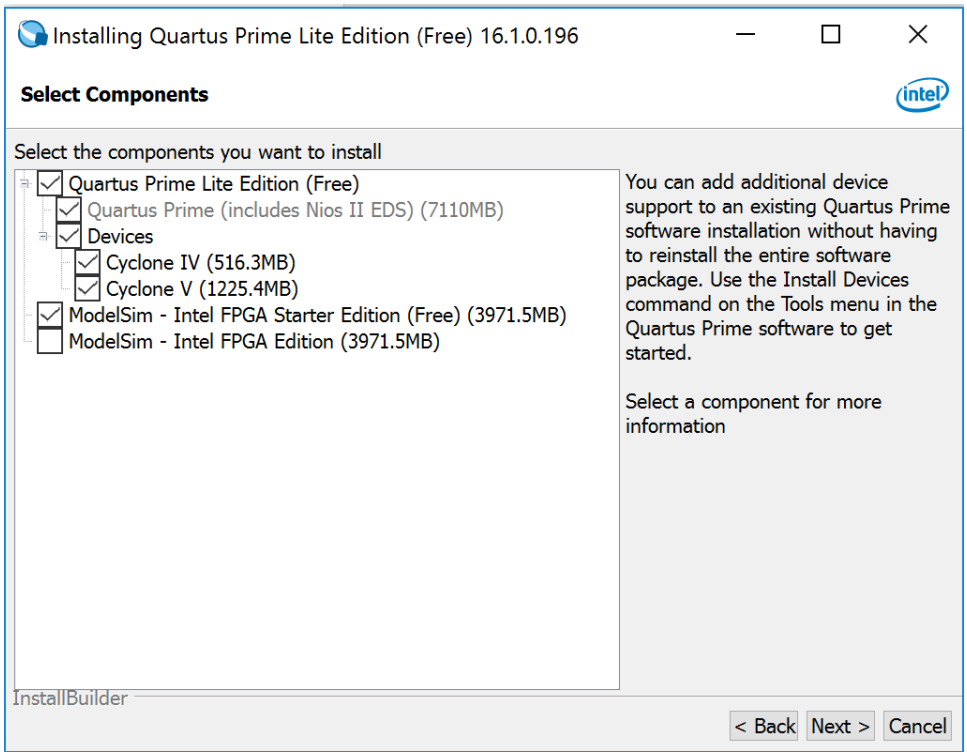
Once the download completes, run QuartusLiteSetup.
Allow the installer to make changes if it asks.
Accept the license agreement.
Keep the default installation folder (C:\intelFPGA_lite\16.1).

Note: You cannot have spaces anywhere in the pathname.

You should have the following items checked for installation (see below):

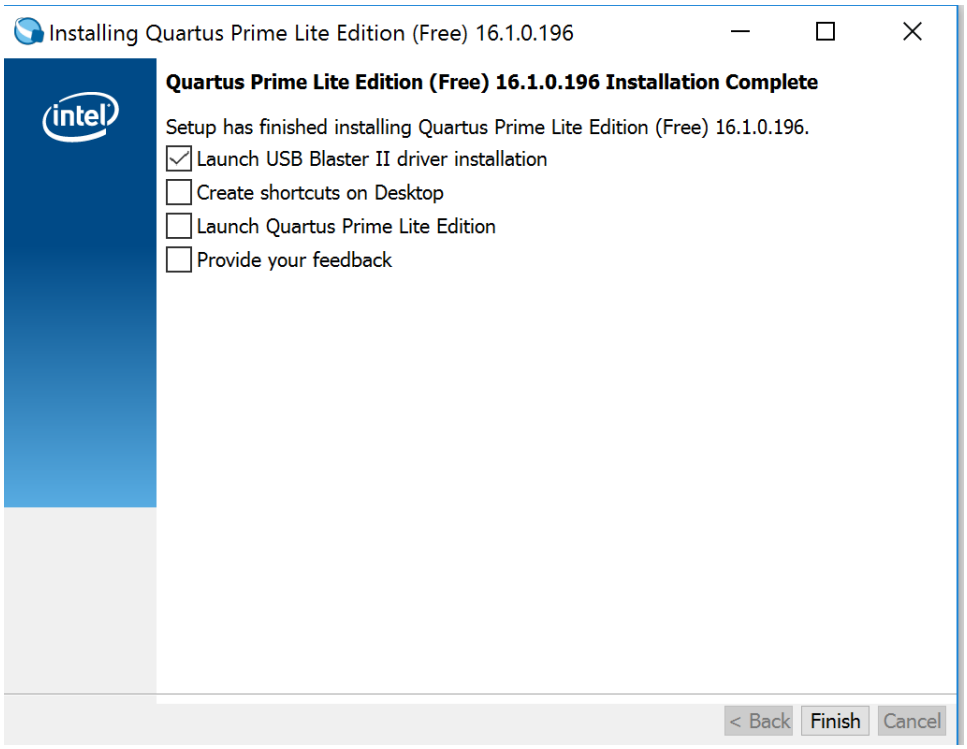
- Quartus Prime (includes Nios II EDS)
- Cyclone IV and Cyclone V under Devices
- ModelSim - Intel FPGA Starter Edition (Free)

Do not check the bottom box to install “ModelSim - Intel FPGA Edition”.



(Some of the options are gray because I previously had the software installed on my computer. Options that are available for installation should appear in black.)

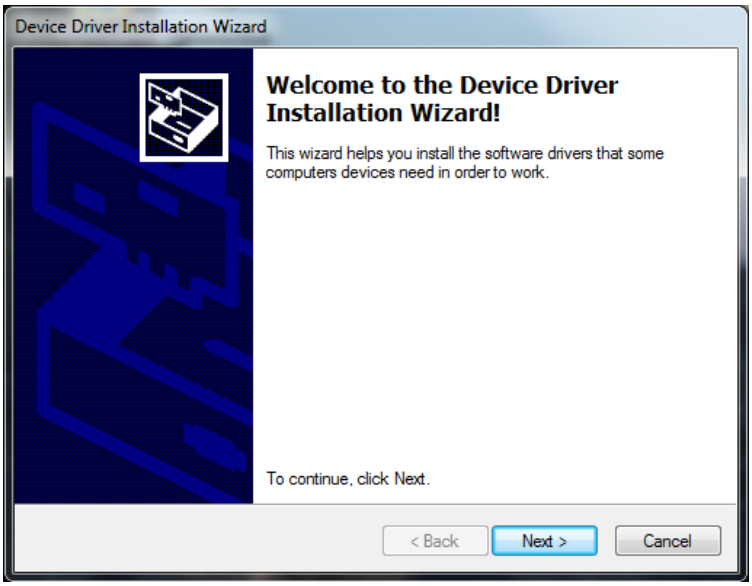
Click **Next**. The installer should begin. If something needs permission to make changes during installation, allow it temporarily.



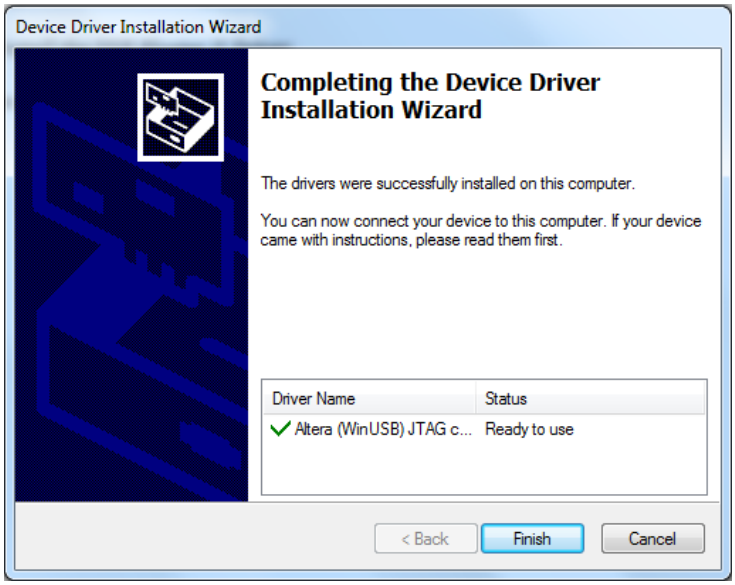
Whether or not you create Desktop shortcuts or provide feedback is up to you. I chose not to launch Quartus as part of the installation of my USB Blaster II drivers, but you might find that you will need to launch Quartus to install your drivers. Keep the **Launch USB Blaster II driver installation** box checked. Click **Finish**.

Step 2: Install the USB Blaster II Driver.

From the previous step, press **Next** to continue with the driver installation.



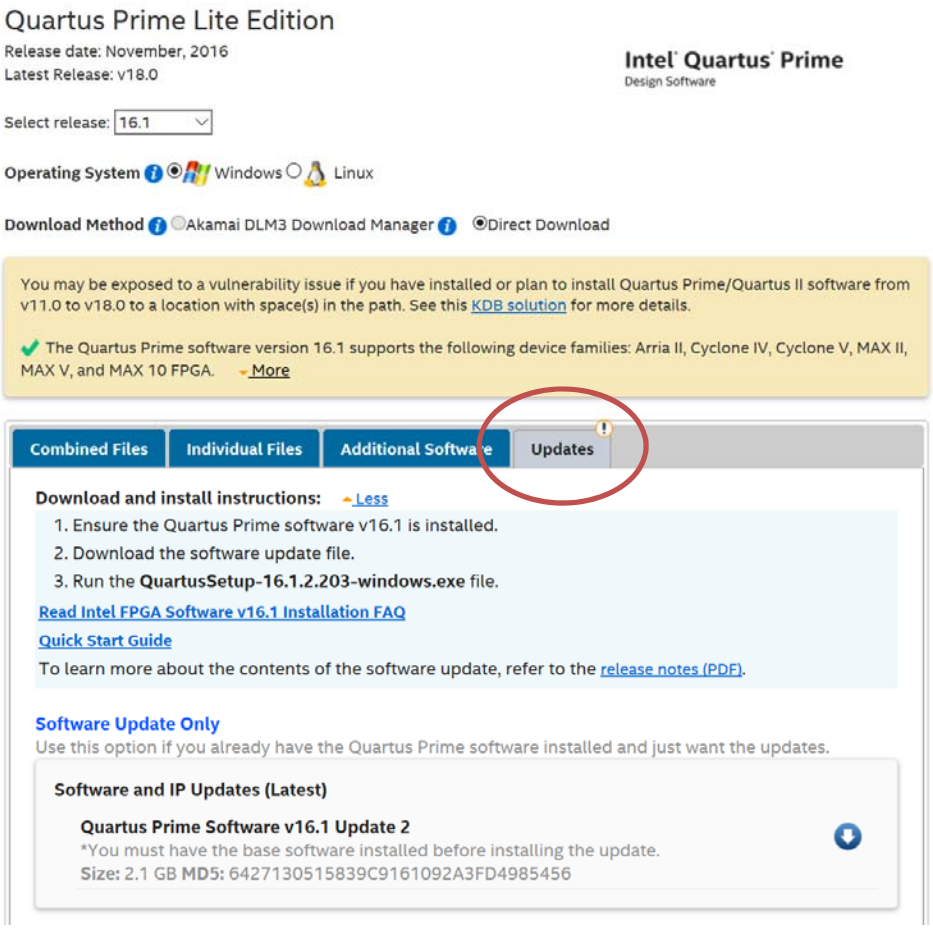
Wait for the drivers to install. If the drivers install correctly, press “Finish” on the window to complete the driver installation.



If you are unable to complete this step correctly, see your instructor.

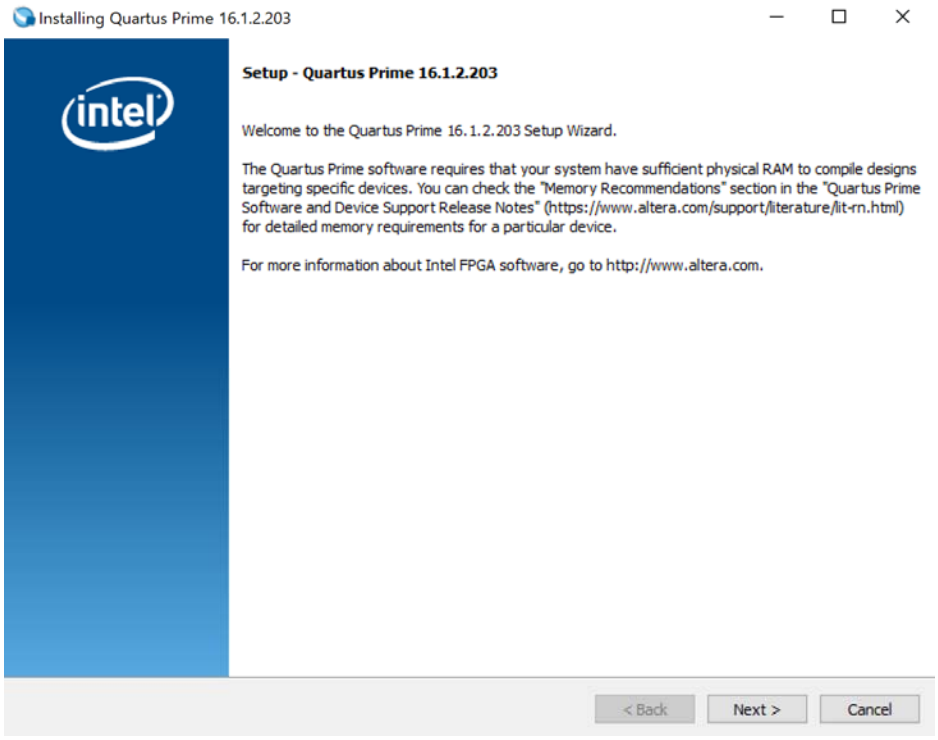
Step 3: Update Quartus 16.1.

From the download page for Quartus Prime Lite (See Step 1), choose the **Updates** tab.



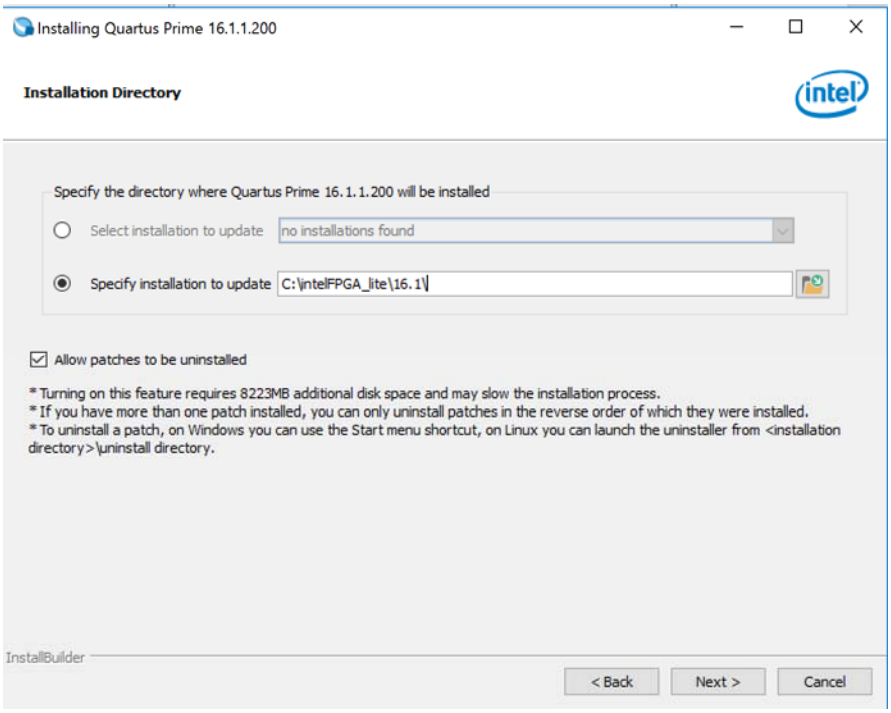
Download and run the **Quartus Prime Software v16.1 Update 2**. Allow the installer to make changes if it asks. Note: This is a very large download.

You should see something similar to this window as part of the setup for the Quartus update.



Click **Next**. Accept the license agreement. Click **Next**.

In the window that appears, you should be able to choose the default installation to update; it should correspond to the version of Quartus that you installed in Step 1. If no installations are found, use the Specify option and navigate to the path selected in Step 1. Click **Next**. The update installation will begin.

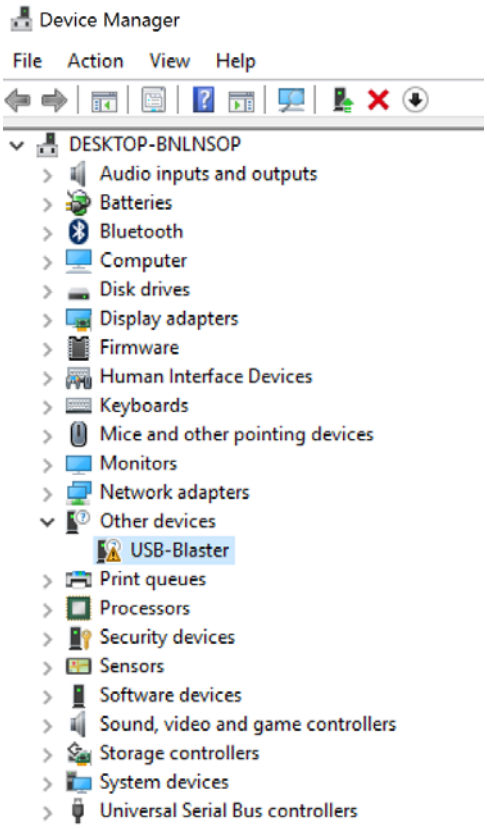


When installation finishes, press **Finish**.

Step 4: Update USB-Blaster driver.

If you have trouble getting the DE-0 board to be recognized

1. Go to the Device Manager from your computer's Start Menu.
2. Select "USB Blaster" (under Other Devices).



3. Right click and select "Update Driver".
4. Give the path C:\intelFPGA_lite\16.1\quartus\drivers.

Creating and Simulating Schematic Designs in Quartus

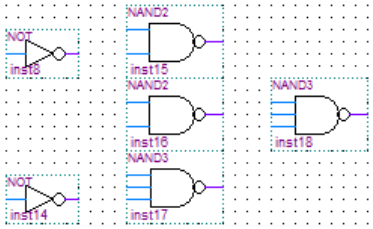
Version: 16 August 2017 (JST, KLC)

1. Create a working directory for the files you will create in this assignment. (For example, **c:\ece2504\hw3**)

2. Choose File → New, then choose Block Design/Schematic File under Design Files.

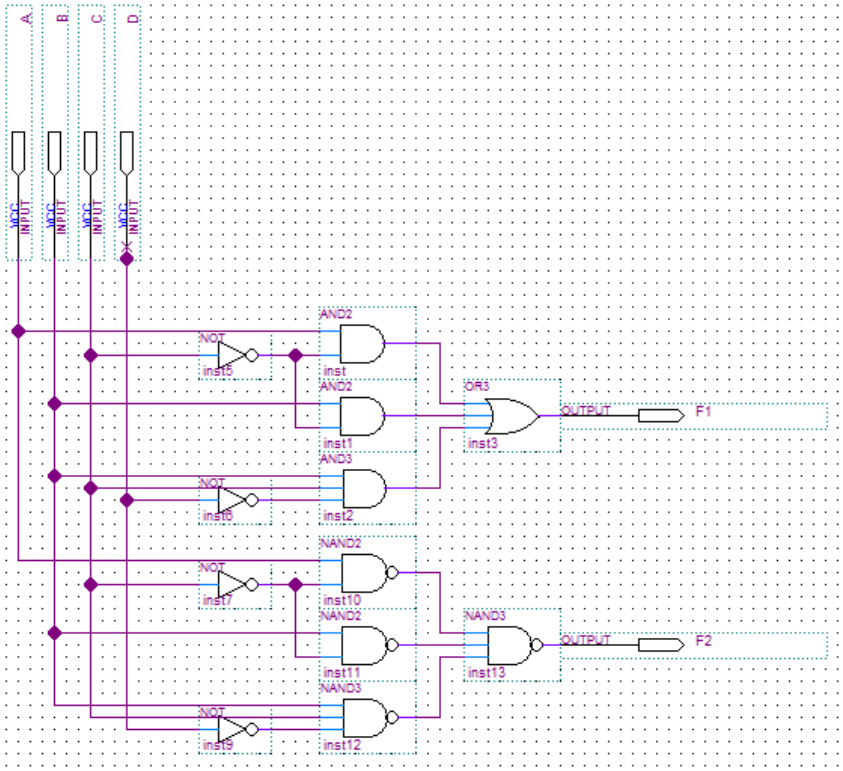
3. On the toolbar, choose the **Symbol Tool**. In the **Libraries** window, expand the top-level folder, then the **primitives** folder, and then the **logic** folder. Here you will find logic gates of various types having different numbers of inputs.

4. Choose a gate by double-clicking it, then click a spot in the schematic window to place the gate. You should lay out your gates carefully so that you can keep your wiring neat and organized.



5. To insert wires, choose the **Orthogonal Node Tool**, or hold the cursor at a gate input or output until you see the symbol for the Orthogonal Node Tool appear. To place the wires, click and drag between the pins you want to connect. *Be careful about creating wire nodes, as they can create connections between wires that you don't want to connect.* You may have to create pin-to-pin wires in part by drawing the first part of the wire from the starting pin to a point near the pin you want to connect, and then finishing the wire by connecting the endpoint of the wire to the pin, or the pin to the endpoint of the wire. You will know that you have connected a wire to a pin when a square appears on the pin you are trying to connect.

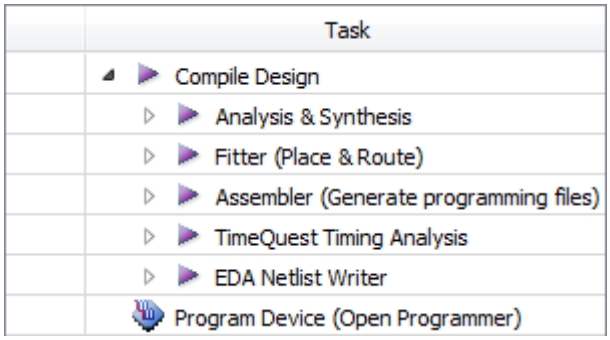
6. To insert inputs and outputs, choose the **Pin Tool**. Use the drag-down menu to choose the kind of pin type you want (input or output), then place the pin in a convenient location that will make it easy to connect to the circuit. One strategy is to place an input pin and then rotate it 270 degrees left (right-click on the pin to find this option) so that you can create a master set of wires that run down the right side of the page. In this manner, you can wire more than one circuit to the same set of inputs.



Rename a pin by double-clicking on the existing name and typing in a name of your choosing. You will need one input pin per input variable, and one output pin for each circuit output.

7. Once you have finished your schematic, save your file. When prompted to create a new project with the file, say Yes:
   → In Step 1, the working directory should be the same one that you created for your files. The project name and top-level entity should match the name that you used to save the file.
   → In Step 2, you don't need to add any files.
   → In Step 3, choose **Cyclone IV E** as the family, and **EP4CE22F17C6** as the device.
   → In Step 4, make sure that "Verilog HDL" appears as the **Format** for **Simulation**.
   → In Step 5, choose **Finish**.

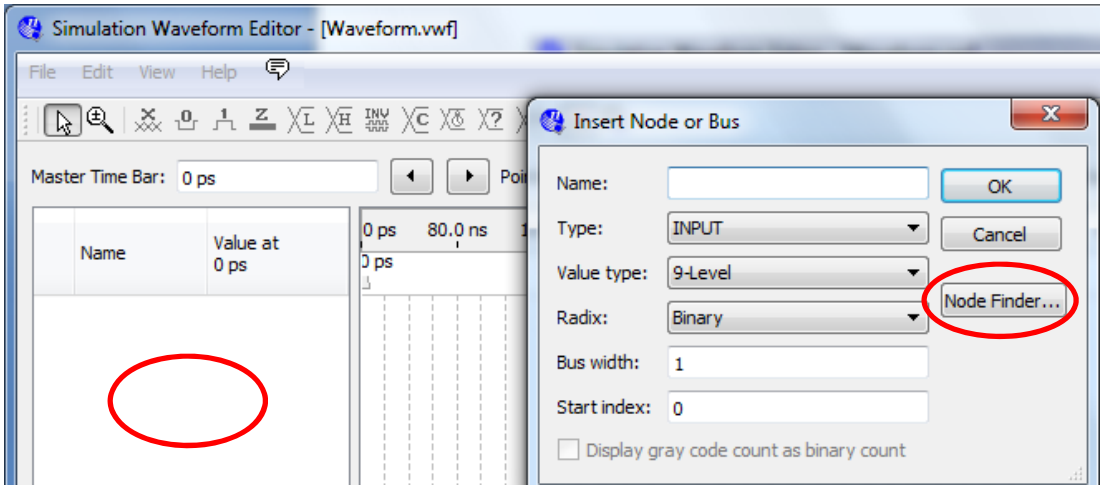
8. After creating a project for the file, you should be able to compile the design.



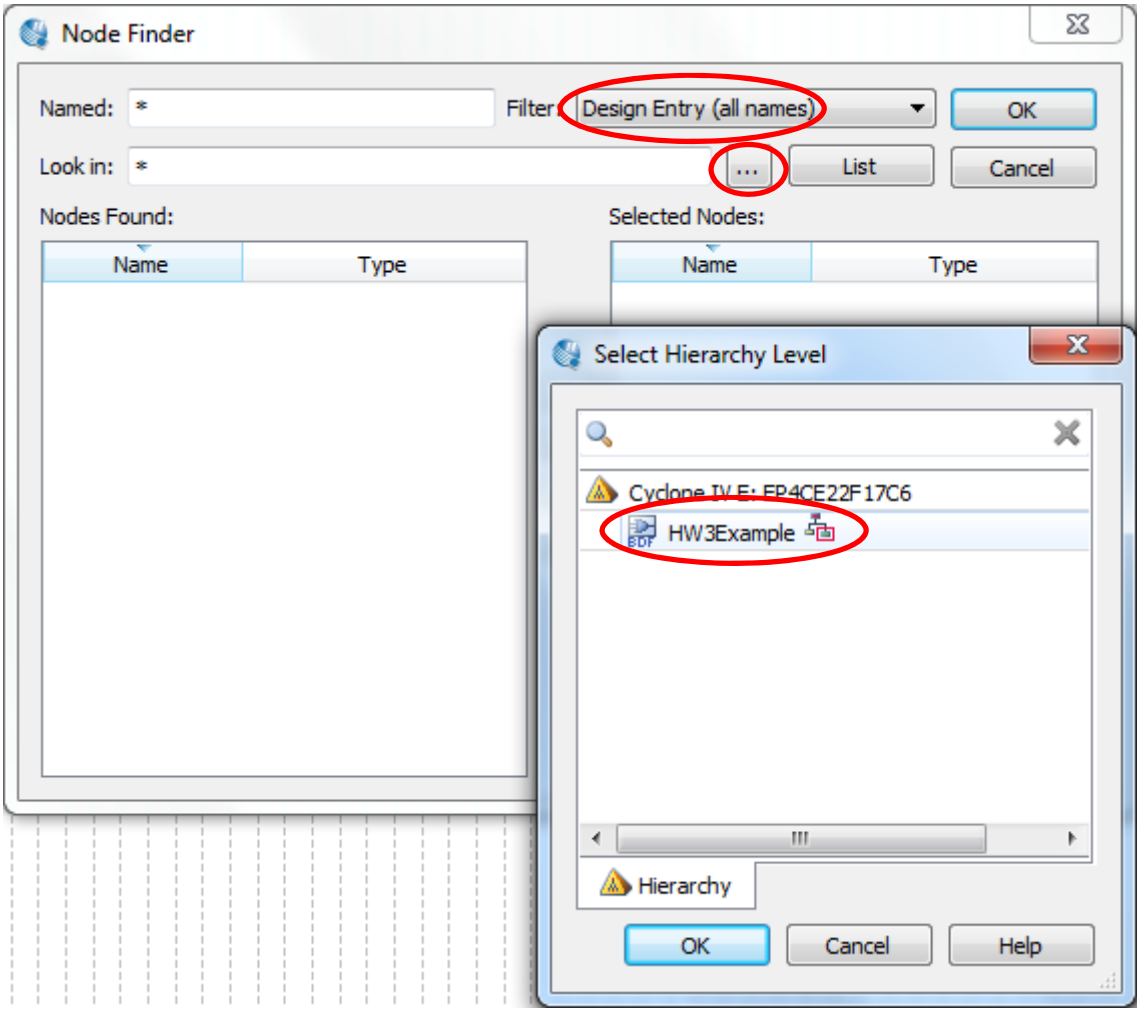
For a design that you don't plan to implement on your DE0 Nano Board, you only need to perform the **Analysis & Synthesis** by double-clicking that line in the Task window. If your design is free of physical defects, you should see the Analysis & Synthesis progress bar run from 0% to 100%.

9. Once you have compiled the design, you should be able to simulate it. To begin simulation, choose **File > New**, then choose **University Program VWF** under **Verification/Debugging Files**. This will invoke the Simulation Waveform Editor.

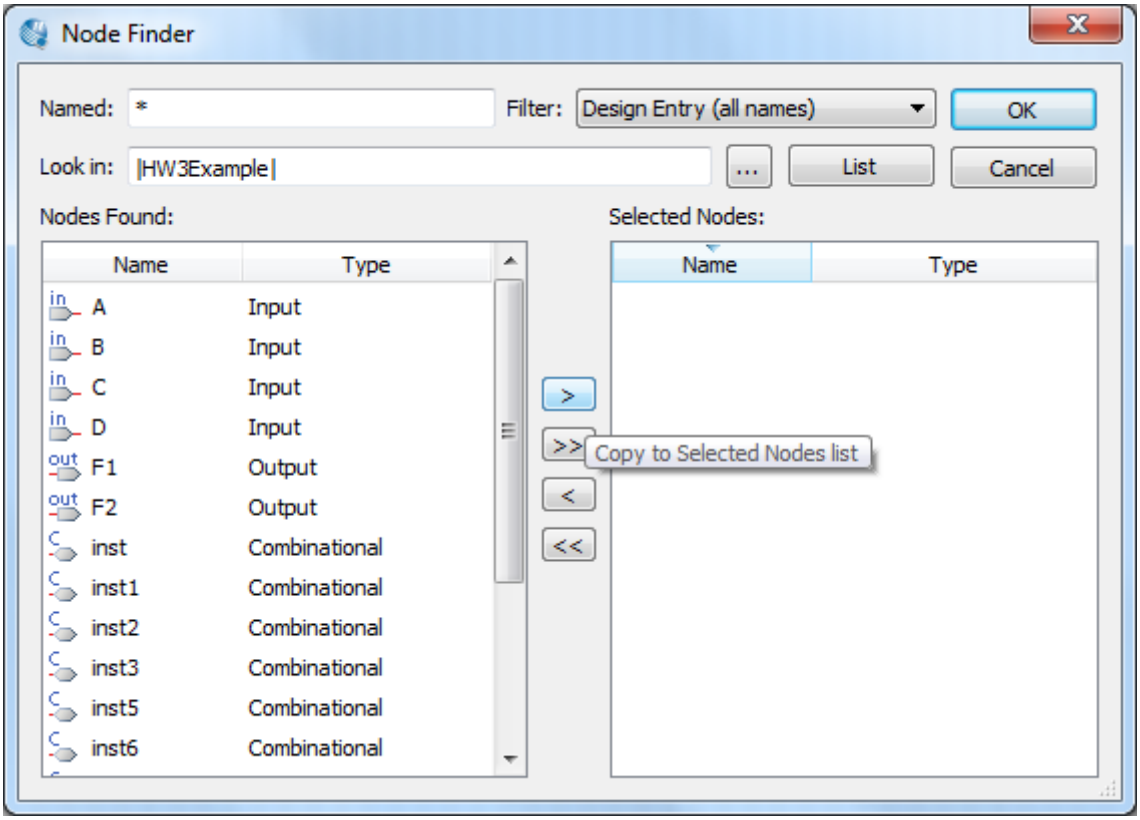
   Double-click the blank area as below to open the **Insert Node or Bus** dialog box, then click **Node Finder…** in the **Insert Node or Bus** window.



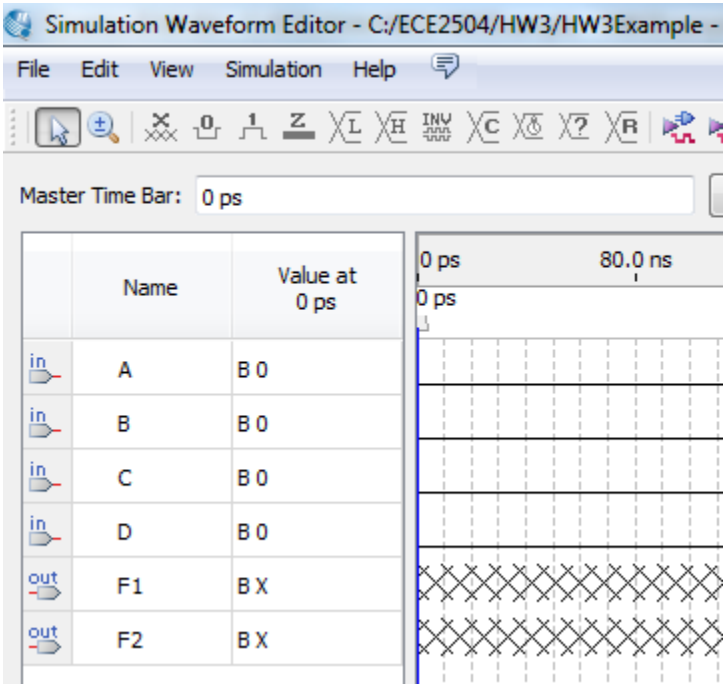
Select **Design Entry (all names)** from the Filter drop-down menu. Click **…** to select the hierarchy. Select **basicmoduleP0** in the Select Hierarchy pop-up window and click **OK**.



Click the **List** button to display the list of nodes. If no nodes appear, you should re-compile the design and repeat the procedures Step 9 from the beginning. Choose the names that you gave to your inputs and outputs, then click > to move them to the **Selected Nodes** list. Click **OK** after you have chosen all of the nodes you want to appear in the simulation.

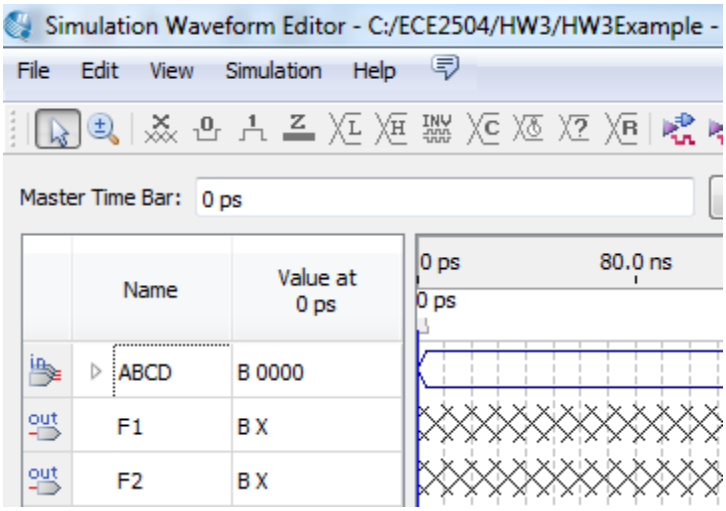


Click **OK** on the **Insert Node or Bus** window. You should see something like this for the nodes you actually used:



Group inputs by selecting the inputs you want to put into a group (use Ctrl to select multiple objects), right-clicking, and choosing **Grouping** > **Group**. You can work with the inputs to a particular circuit easily when you group all of that circuit's inputs together.

Be careful – the signals will be grouped in the order they are listed. For example, if the signals A, B, C, and D are listed as shown in the figure above, assigning a value of 1100 to the group will make A and B equal 1, and C and D equal 0. However, if the signals were listed in the order D-C-B-A before being grouped, assigning a value of 1100 to the group will make A and B equal **0**, and C and D equal **1**, which is quite different.

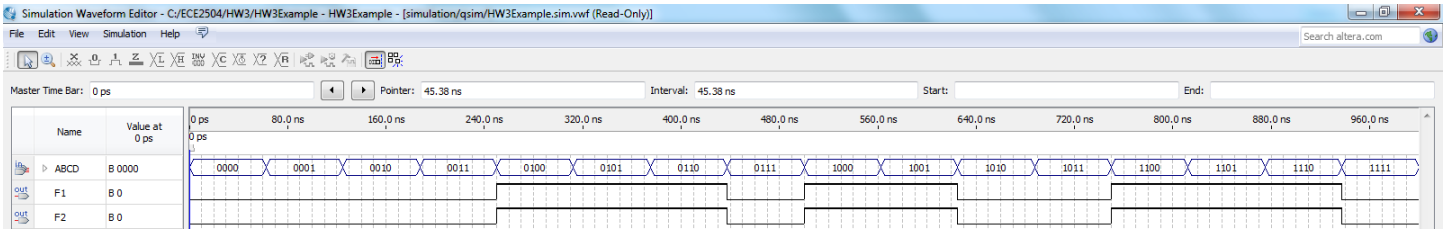


Use the **Count Value** button to apply input combinations in numeric order. 1000 ns of simulation time are visible in the window, so set a counting period that will allow you to see all sixteen combinations of the four input values.

Note: The Count Value tool is only appropriate when you want to see all the input combinations in numeric order. Otherwise, highlight the group for the time period you want to assign, right click, and assign the value you want.

In the Simulation Wave Editor window, choose **File** > **Save As**, give your file a name and click **Save**. Then choose **Simulation** > **Functional Simulation**. Let the simulation complete. At the end, a window will open containing your simulation result.

10. You can screen shot your simulation result and schematic for inclusion in your homework document. If you are worried about the resolution, paste the screenshot into MS Paint and crop any unneeded white space.



Using Quartus and Qsim to Simulate Verilog Models

Version: 12 January 2017 (TLM, JST, KLC)

This tutorial assumes that you are already familiar with using Quartus and Qsim to simulate schematics. The tutorial will walk you through the steps of creating a new project and adding Verilog files to it. You will then simulate the Verilog models with Qsim, in the same way that you simulated your schematics for projects 0 and

CREATING A NEW QUARTUS PROJECT

1. Start Quartus.
2. Select **File** > **New Project Wizard**.
3. The New Project Wizard window will open. Select **Next**.
4. Choose a working directory. The default will be in the Intel FPGA installation directory, but you should choose a folder where you are storing your ECE 2504 materials.
5. Choose a name for the project. For this tutorial, we will use *verilog_tutorial*.
6. Choose a name for the top-level entity for the project. This name must match exactly the name of the top-level module in your Verilog model. For this tutorial, we will use the default, which is the same as the name for the project, *verilog_tutorial*.
7. Select **Next**.
8. On the **Project Type** page, select **Empty project**.
9. On the **Add Files** page, select **Next** without adding any files. We will create a Verilog file later. If you already had a Verilog file created, you would add it here by navigating to the directory where the file is located and selecting it.
10. On the **Family & Device Settings** page, choose **Cyclone IV E** under the **Family** pull-down menu. Under **Available devices**, select **EP4CE22F17C6**. You can click and drag the **Name** column in the **Available devices** menu to widen the column. This will help you to see the entire device name. The device does not matter for this homework, but this device setting will be important for using the DE0 Nano board to implement designs, as you did in Project 0 and Project 1.
11. Select **Next**.
12. On the **EDA Tool Settings** page, choose **Verilog** on the pull-down menu in the **Format** column of the **Simulation** tool type. Your installation of Quartus may default to this.
13. Select **Next**.
14. On the **Summary** page, select **Finish**.

You now have created a new Quartus project. The next set of steps will create a Verilog model in the project.

ADDING A VERILOG MODEL TO THE PROJECT

1. Select **File** > **New**. In the window that opens, select **Verilog HDL File** under **Design Files.** Select **OK**.
2. A new Verilog file will be created with a default name. The file will be open in the right Quartus panel. Select **File** > **Save As**, and then change the file name to *verilog_tutorial*. Make sure the **Add file to current project** option is checked. Select **Save**.
3. Type the following model into the verilog_tutorial.v file, then save it. Observe how the module in the tutorial uses indentation to structure the Verilog code.

```
// My first Verilog model
module verilog_tutorial(a, b, c, d, f);
    input  a, b, c, d;
    output f;
    wire   g1, g2;

// For the built-in gate primitives, the output is the first port.
// The remaining ports are inputs.

    and a1(g1, a, b);
    and a2(g2, c, d);
    or  o1(f, g1, g2);

endmodule
```

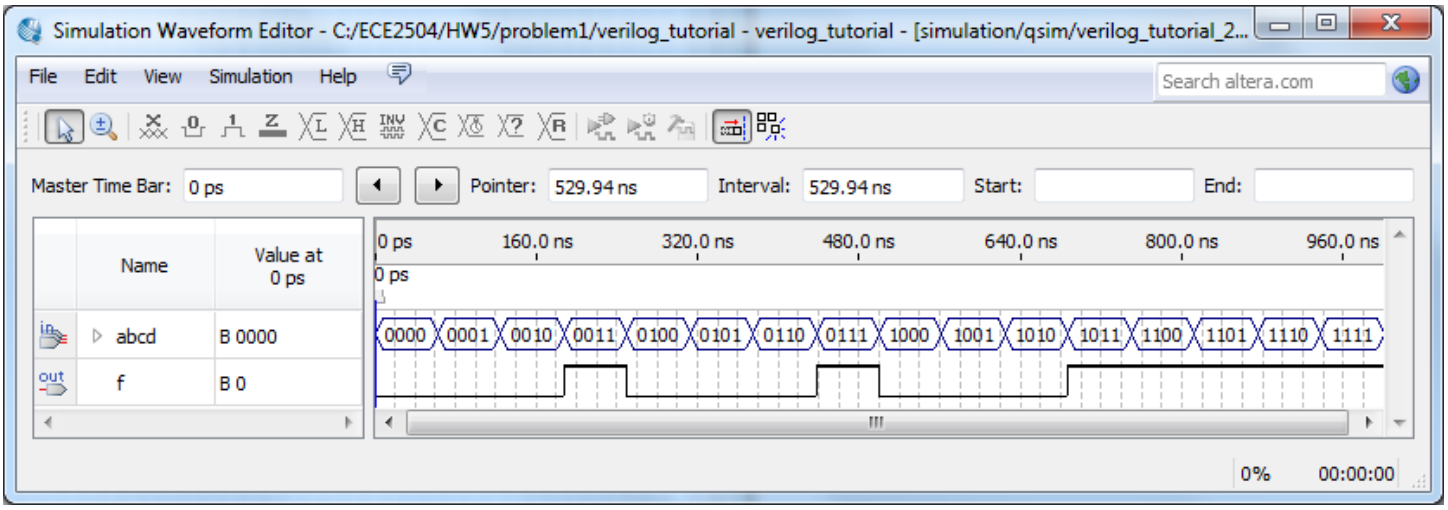
4. Once you have saved the file, compile it by double-clicking on the **Analysis and Synthesis** button under **Compile Design** in the **Tasks** window to the left. You can also navigate to this option using the menus: **Processing** > **Start** > **Start Analysis and Synthesis**. You did this in Project 0 and Project 1 for your schematics.

If you have any compilation errors, you should correct them before continuing to the remaining steps.

SIMULATING THE PROJECT IN QSIM

You can now simulate your module using ModelSim using the steps that you used with simulating schematics in Project 0 and Project 1. If you have difficulty with the following steps, refer to the simulation instructions from Project 0 and Project 1.

1. Open the *verilog_tutorial* project in Quartus.
2. Select **File** > **New**. In the window that opens, select **University Program VWF** under **Verification/Debugging Files.** Select **OK**.
3. Use the **Node Finder** to generate a simulation input file having the inputs and outputs indicated in the *verilog_tutorial* module. Set up waveforms for inputs a, b, c, and d that use all possible combinations of values for them. Try grouping the inputs and then using the Count Value function to assign values.
4. Under the **Simulation** menu, choose **Run Functional Simulation**.
5. Verify that the output of the simulation behaves as you would expect from the netlist in the Verilog model. Your simulation results should be similar to the results shown below.



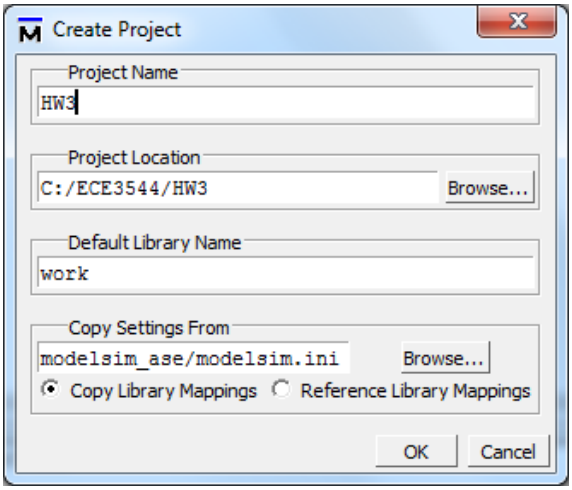
Creating and Simulating Verilog Source Files in ModelSim

Version: 09 September 2015 (JST, TLM)

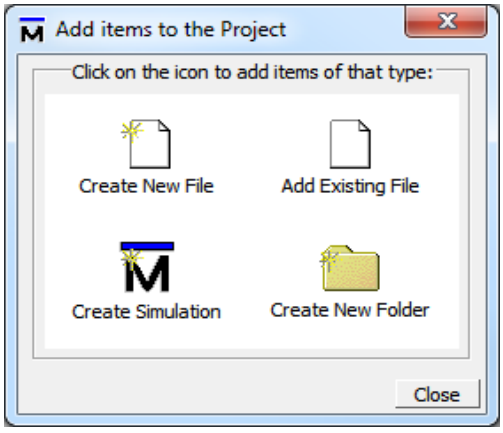
Step 1: Make a project.

You will need a folder in which to hold the source code and test bench files that you generate. Create a folder for your assignments to have a clean hierarchy of projects on your computer. Using the "Downloads" folder or the Desktop is **not recommended**! Use a path name like "C:/ECE3544/HW3" or "C:/ECE3544/Project1" for the folder containing your work. You should make a habit creating a new folder for each new assignment.

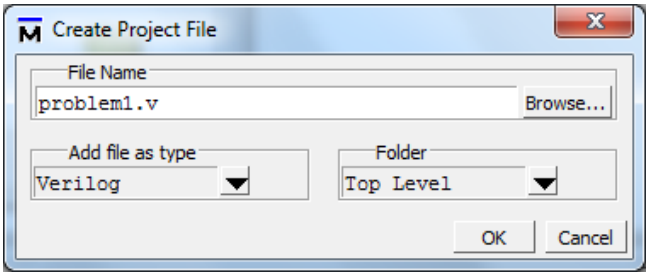
Run ModelSim and create a new project. Choose **File > New > Project** on the menu bar**.** Specify your project location and assign a name to your project, then click **OK**. Let ModelSim create the directory if it does not already exist.



Now, you should see the window shown below. Click the icon that is appropriate for what you want to do.



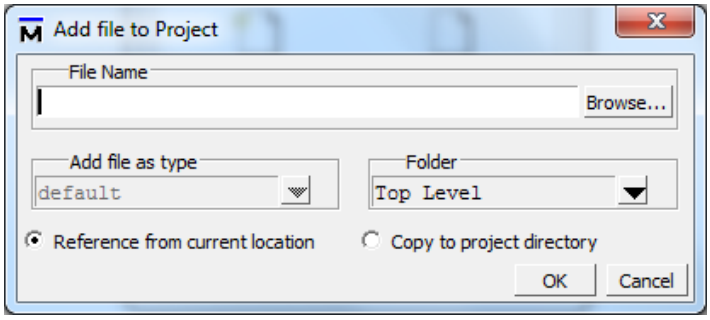
Sometimes you will need to create new files for a project. When you click the **Create New File** icon, you will see a window similar to the one shown below.



When you enter a file name, remember to put the .v extension after the file name you provide. Choose **Verilog** as the file type and **Top Level** as the Folder. You can repeat these steps – choosing **Create New File** on the **Add items to the Project** window, then setting the attributes for each file in the **Create Project File** window – for each file that you need to add to the project.

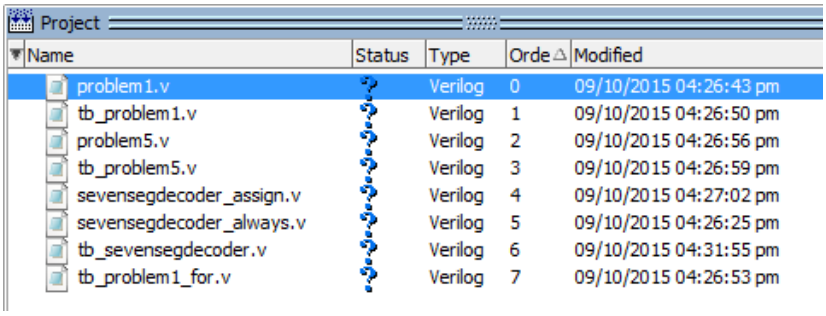
Test benches files are themselves Verilog modules. Instead of using the **Create Simulation** icon to create simulation files, you should create new Verilog files to serve as your test benches.

Other times you will work from existing files that you want to add to a project. In the "Add items to the Project" window, click **Add Existing File.** You should see a window similar to the one shown below.



In the new window, click **Browse** to locate and select the files you want to add to the project. You can repeat these steps to add all of the files that you want to the current Project.

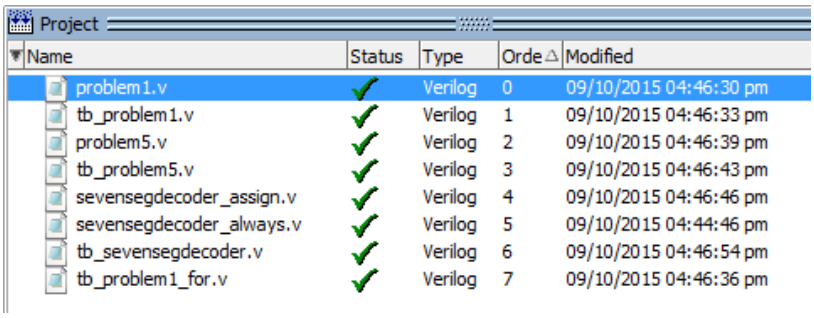
Once you have added all of the files that you want, close the **Add items to the Project** window. You should be able to see added files in the **Project** panel of ModelSim:



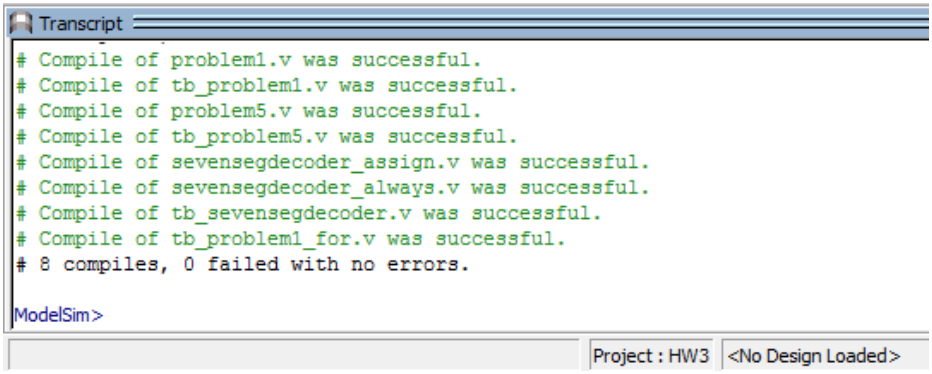
Double-click on files to open them and see their contents. (Sometimes this doesn't work. You can also choose **File > Open** on the Toolbar.) Newly-created files will be empty. You might be asked to associate a program with the files if you haven't already done so. Use a useful program such as Notepad++ to view Verilog files. You can also use ModelSim directly to view and edit your files. (I actually recommend this.)

Step 2: Compile your Project.

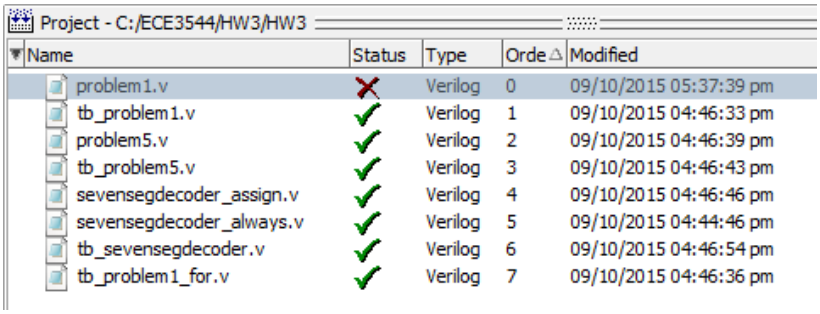
Compiling the program makes sure the code is error-free and generates the model for future steps. On the Menu bar, click **Compile > Compile All**. Green checkmarks in front of the file names indicates that they compiled successfully.

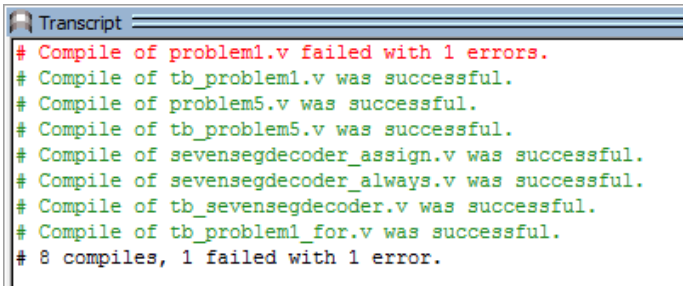


You can also see this at the bottom in the **Transcript** panel.

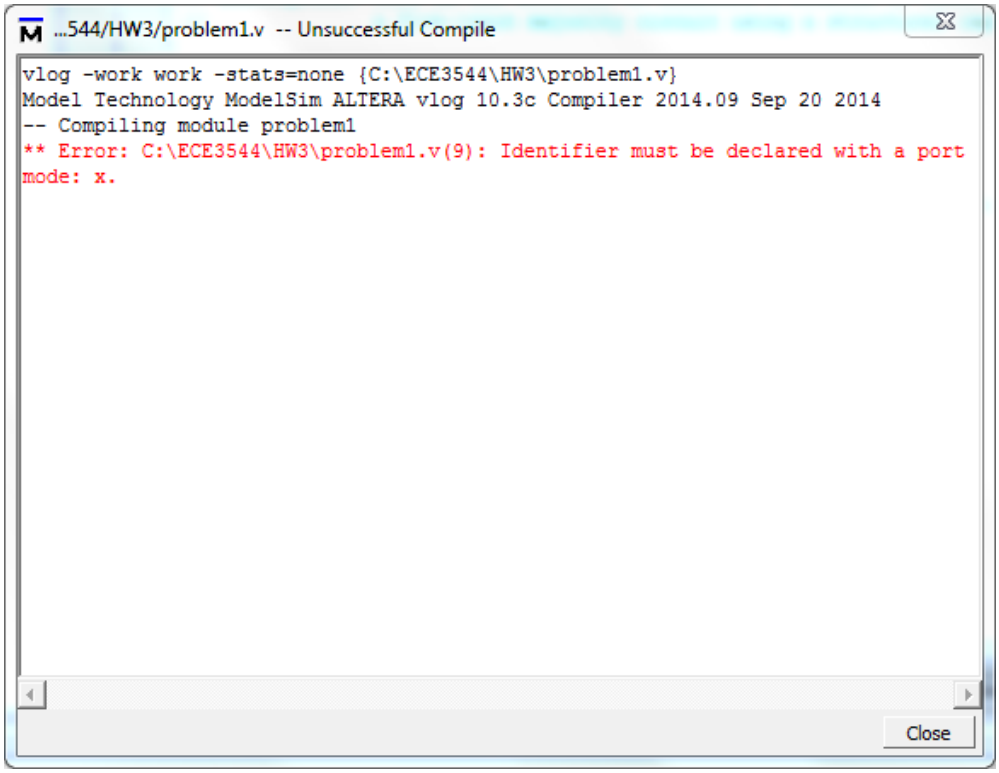


The same panels will provide you with evidence that your files did not compile error-free:





For each compile that fails, double-click on the line in the **Transcript** panel to bring up a window showing the reasons for the unsuccessful compile:

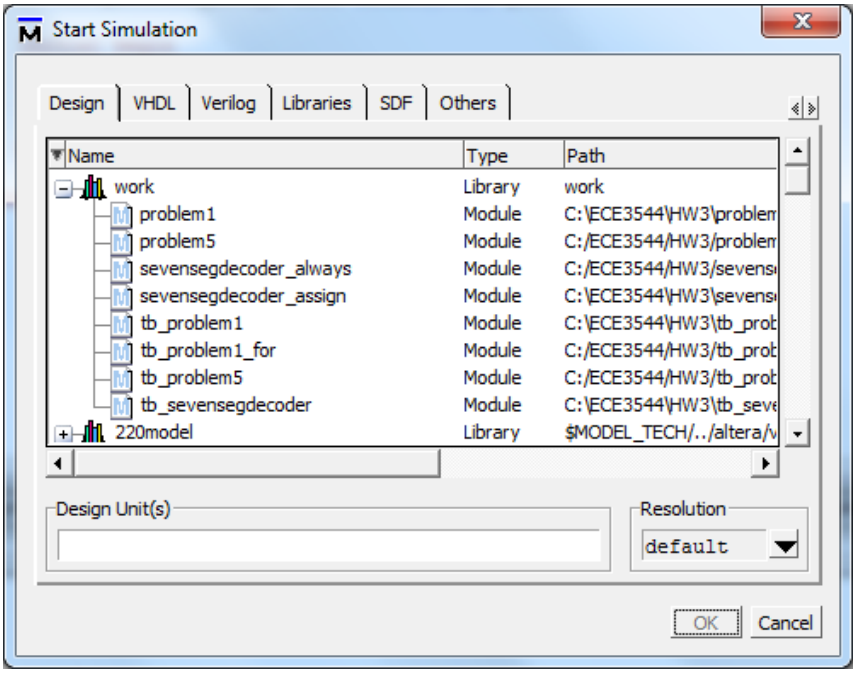


Double-click on each error to highlight the line in your source file where the complier "believes" the error has occurred. Those of you who have experience using compilers probably know that the line indicated by a compiler as containing an error is not always the line that actually contains the error. For example, you might have forgotten a semicolon in a line that precedes the one marked by the compiler. You may have to hunt for errors throughout your code to account for the errors marked by the compiler.

Debug the errors that the compiler indicates until your project compiles without error.

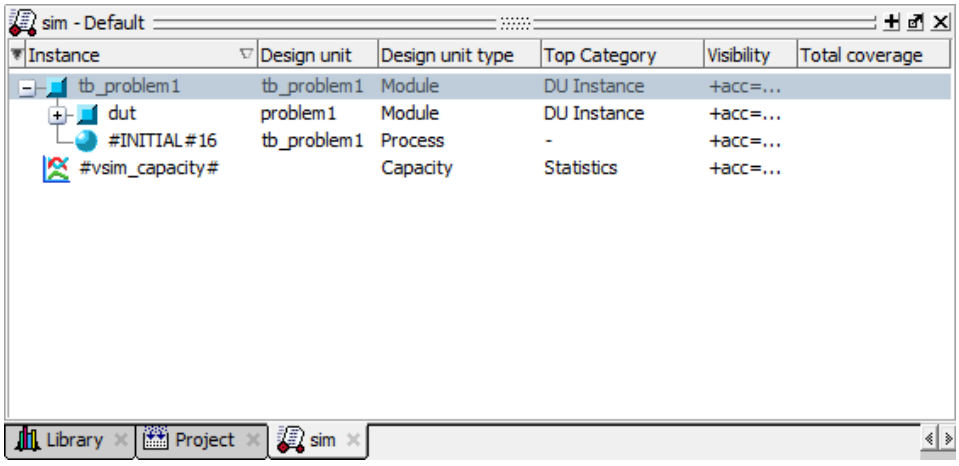
Step 3: Run a simulation.

On the menu bar, click **Simulate > Start Simulation…** In the window that pops up, under the **Design Tab**, expand the **work** library. It looks like this:



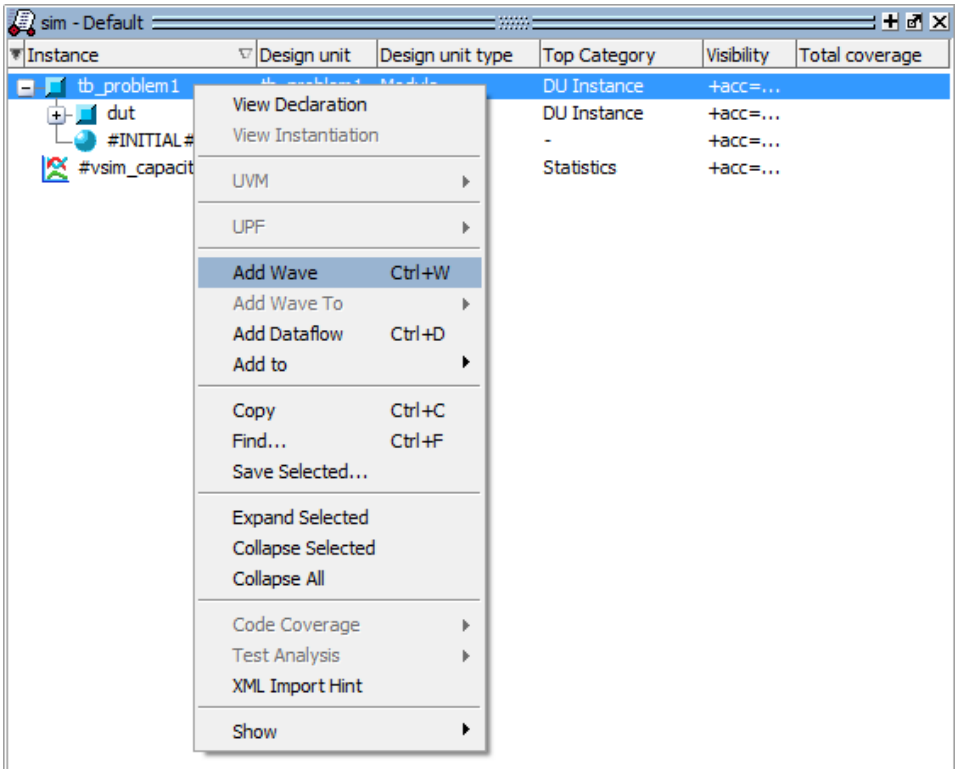
We want to simulate the *test bench* for the design of Problem 1, which was contained in a file and module called tb_problem1.v. Select **tb_problem1** and click **OK**. *Do not choose **problem1*** – this is the module containing your design. You cannot directly simulate the behavior of the design using your source file, as the source code for Problem 1 does not include any input stimulus. This is point of having written a test bench.

Wait a few seconds for ModelSim to load the model for simulation. You will know that it is ready when you can see the **Sim** panel shown below. It will be on the left side of the ModelSim screen.



Here you can see the **tb_problem1** module and its hierarchy of blocks

Right click on **tb_problem1**, then click **Add Wave.**

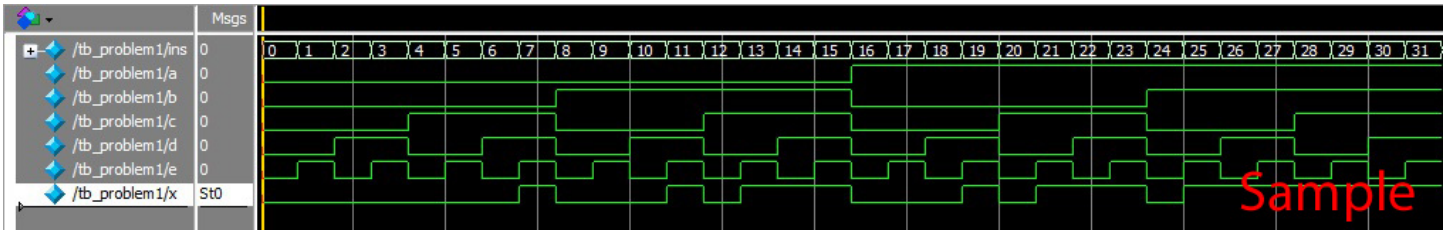


This opens a waveform window with the signals of your module added to it. In this window, you can see some new bars for simulation actions. One of them looks like this:

Click the **Run-All** button. This runs the simulation for as long as events occur in your test-bench. Be careful: if your test bench contains a process that does not terminate, your simulation will not terminate. Click the **Stop** button to terminate a simulation. The Stop button will be greyed-out when the simulation is not running.

You should now be able to see the output waveform in the Wave window. You will probably have to use the zoom functions to get the best view of the results. You can also undock the Wave window from the set of simulation windows.

Your waveform should look something like this:



Viewing waveforms in binary is generally acceptable. However, in some situations, you might prefer to view the waveforms in another base such as decimal or hexadecimal. To change the base, right-click on the waveform name to the left of the waveform (the gray area in the figure above), select **Radix**, and then choose your preferred radix.

You can go back to the design environment by clicking **Simulate > End Simulation.**

Simulating Quartus Modules Using ModelSim

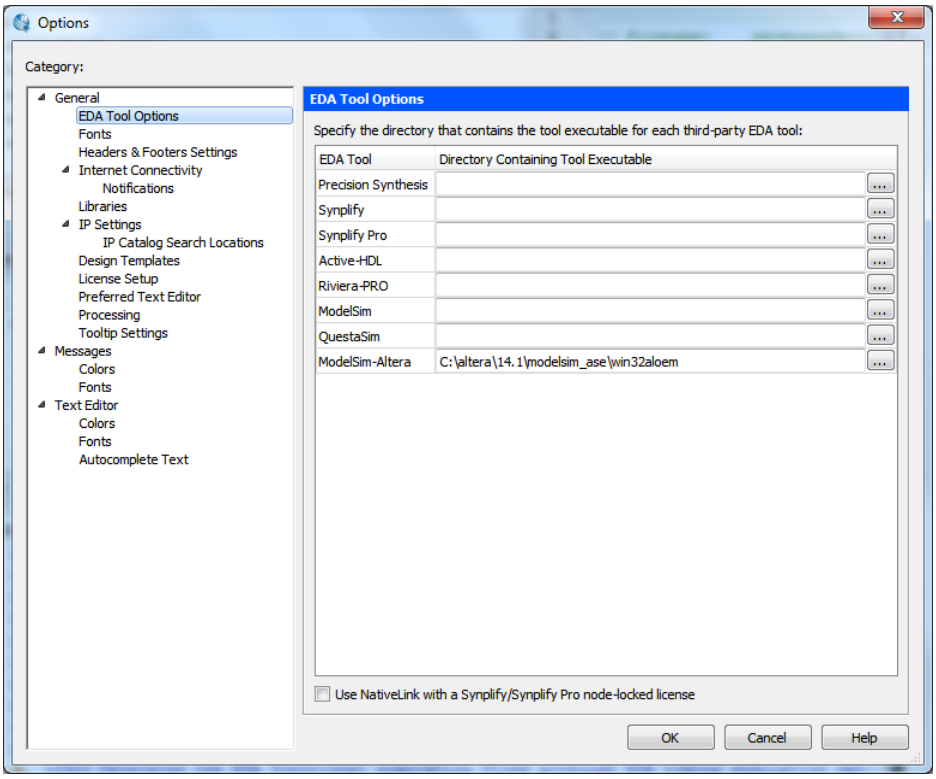
Version: 08 October 2015 (JST, TLM)

Configuring Quartus to Invoke ModelSim

Before running the ModelSim simulation of your module, you must make sure that Quartus has the absolute path to ModelSim.

1. On the menu bar, select **Tools** → **Options**.

2. Select **EDA Tool Options**.



3. If the ModelSim-Altera path does not appear in the list "Location of Executable," select the "…" at the end of the ModelSim-Altera row and browse to where ModelSim-Altera is installed on your computer. The default is C:\altera\14.1\modelsim_ase\win32aloem.

Be careful when you follow Step 3:

- *There is a ModelSim row and a ModelSim-Altera row. Make sure that you select the ModelSim-Altera row.*
- *You may need to add a backslash to the end of the default path, but you might not. (Different versions of Quartus have been fine without the backslash while others have failed to simulate without it.)*

You should only have to follow the above steps once, not each time you run the simulation.

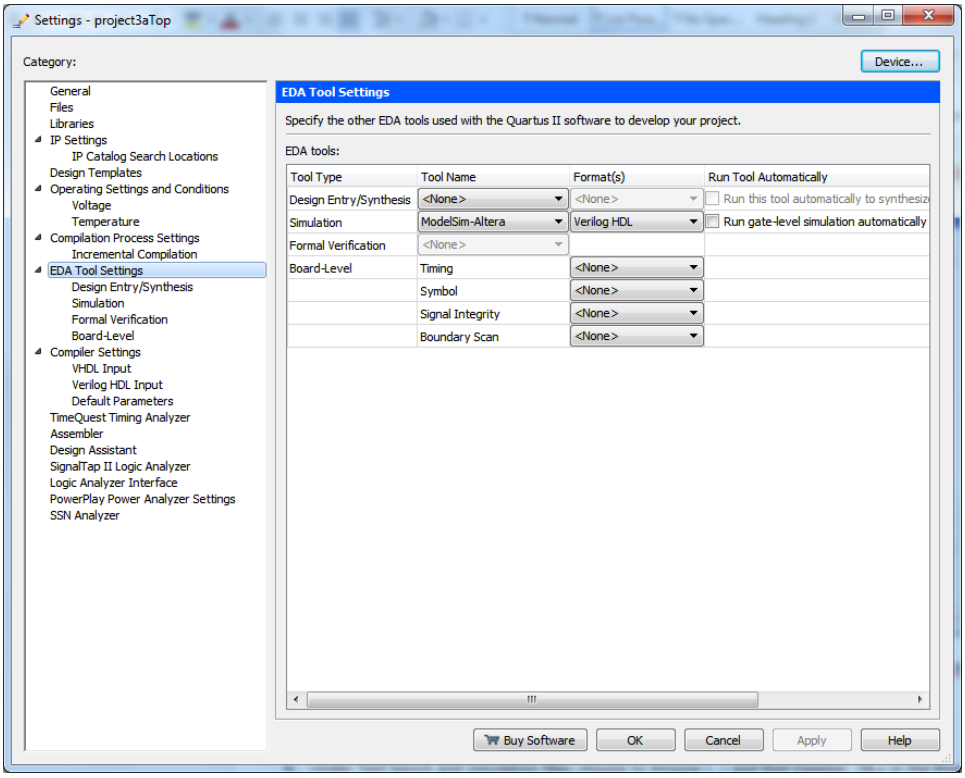
Running ModelSim from Quartus

When you wish to run a simulation in ModelSim from Quartus, you will need to make a few changes in the project settings.

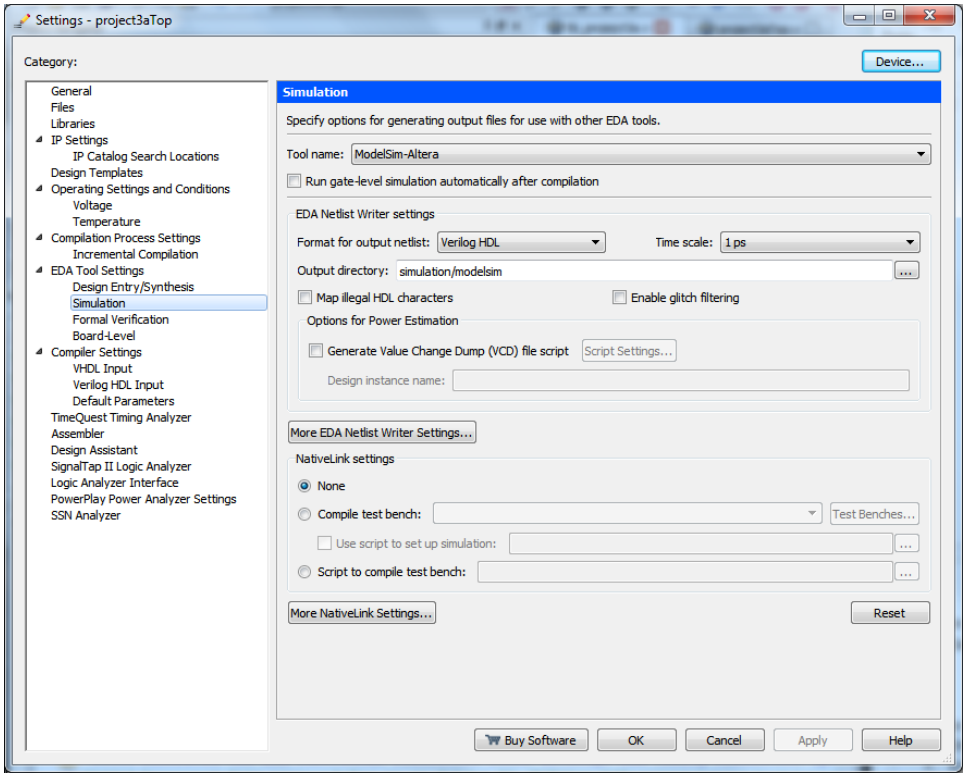
1. On the menu bar, click **Assignments** → **Settings**.

2. In the category list on the left side, click on **EDA Tool Settings**.

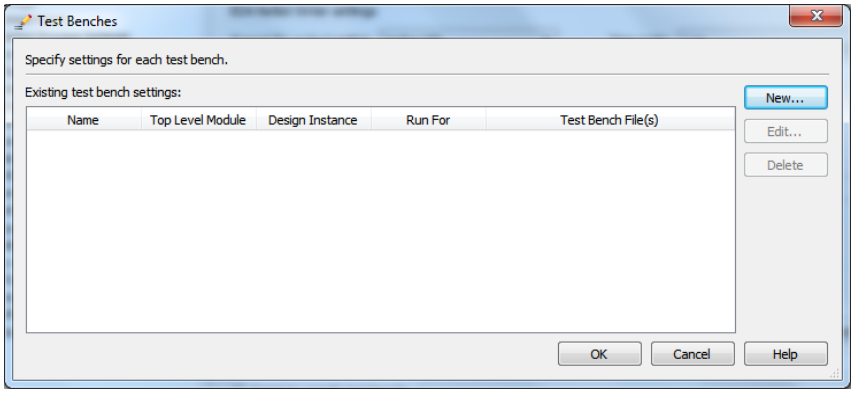
3. Make sure in the **Simulation** row, the column **Tool Name** is set to **ModelSim-Altera** and Format(s) is set to **Verilog HDL**. This should be set by default, but change it if that is not so.



4. In the category list on the left side, click on **Simulation** below **EDA Tool Settings**. Under **NativeLink settings**, select **Compile test bench** and click **Test Benches…**



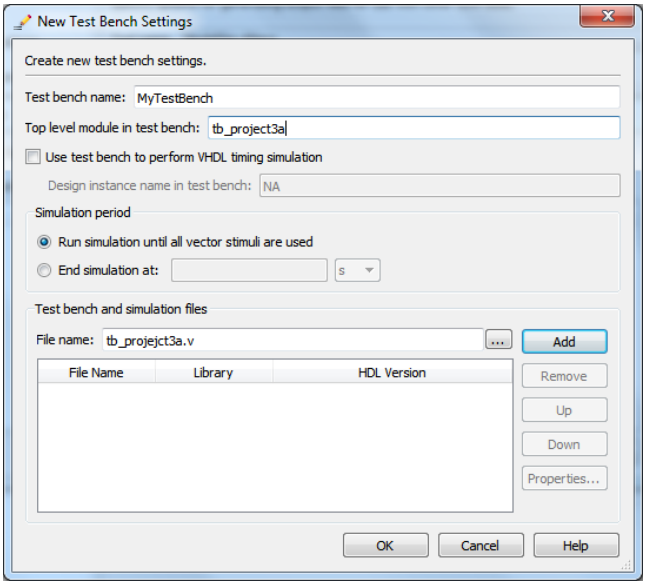
5. In the Test Benches window, click **New…**



6. Enter a name for the test in the **Test Bench Name** field. Note that this is only a name, not an actual file.

7. Enter the name of the test bench module in the **Top Level Module in Test Bench** field. In general, this should correspond to the name of the file.

8. Under **Test bench and simulation files**, choose to browse (…) and find the file containing your test bench in the Project folder. Click **Add** to add this file to the list.



9. Click OK on all open windows to go back to the main Quartus Environment. Now you have defined your test bench for Quartus when running simulations.

10. Select **Tools** → **Run Simulation Tool** → **RTL Simulation**.

    ModelSim should start up, compile your modules. If you have compilation errors, correct them and either close the ModelSim window and restart the simulation from Quartus, or select the compile menu in ModelSim and navigate to the file for your module.

    If there are no compilation errors, ModelSim will generate the test waveforms for all inputs, outputs, and auxiliary variables that you defined in your test bench. The simulation should automatically run for as long as the test bench is meant to run. If the waveforms do not appear, then stop the simulation in ModelSim – you likely have an always block running forever or perhaps some condition of a loop is not being met.

Pin Assignment for the DE1-SoC Board

Version 10 October 2015 (JST, TLM)

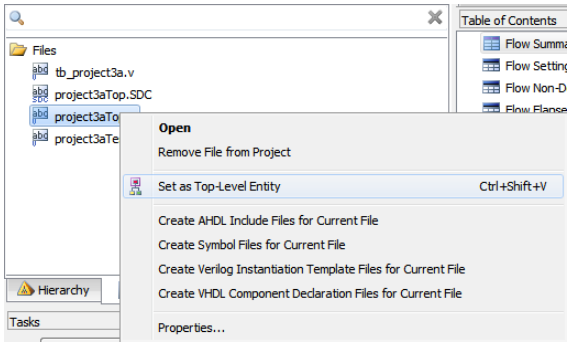
Pin Information

Take note of the following pin assignment information, which will be useful as you move from project to project.

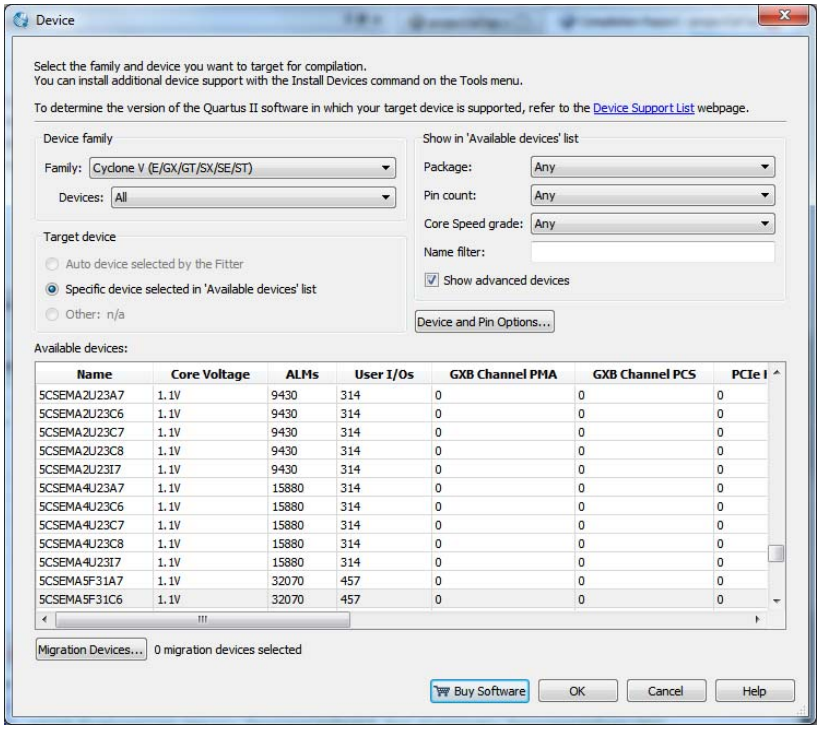
- Table 3-5 on Page 22 of the DE1-SoC User Manual shows the pin assignment of the Clock inputs.
- Table 3-6 on Page 25 shows the pin assignment of the Slide Switches
- Table 3-7 on Page 25 shows the pin assignment of the Push-buttons.
- Table 3-8 on Page 26 shows the pin assignment for the LEDs.
- Table 3-9 on Page 27 shows the pin assignment of the seven-segment displays.

Assigning Pins in Quartus

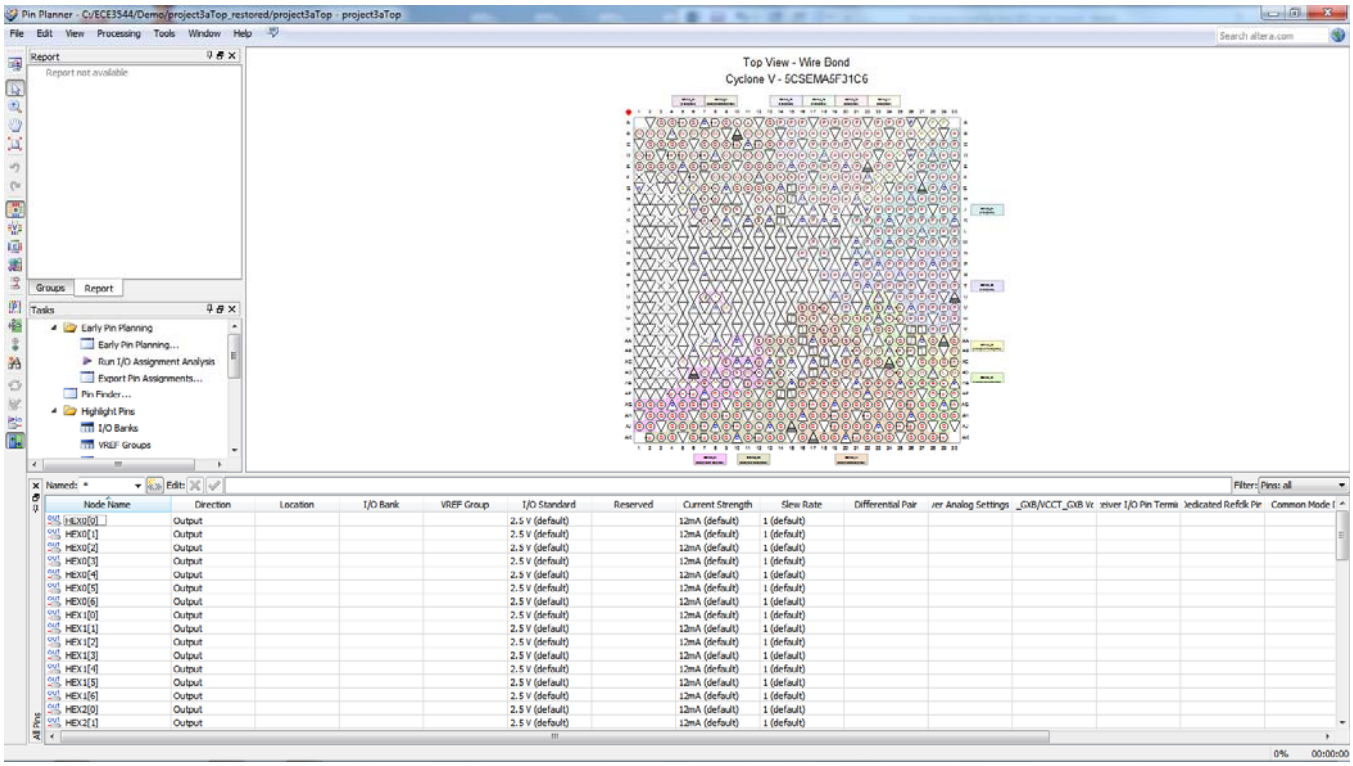
1. Open the top-level module in Quartus. Review the module to locate all references to the pin references or peripheral names that are described above.
2. If the top-level module is not already set as the top-level entity, right-click on it in the **Files** tab and select **Set as Top-Level Entity**.



3. In the menu bar, click on **Assignments** → **Device**. The device family should already be set to Cyclone V, and the target should be set to **5CSEMA5F31C6**. Choose these options if they are not already set.



4. In the menu bar, click on **Processing** → **Start** → **Start Analysis & Elaboration**. You can also double click **Analysis & Design** under **Compile Design** in the **Tasks** window. This step will make the synthesis tool aware of the input and output ports on the top level module and generate a list of the ports for the Pin Planner in Step 6 below.
5. In the menu bar click on **Assignments** → **Pin Planner**.



6. At the bottom of the Pin Planner window, you should see a table. You should see your top-level entity ports in this window. Fill in the **Location** column according to the pin information that you find in the DE1-SoC manual. A

quick way to set the Location values is to type the pin identifier (*e.g.*, AA14) into each box. The Pin Planner will match the pin name. For each pin, set the value in the **I/O Standard** column to 3.3-V LVTTL. A quick way to set the I/O Standard values is to select all of the rows in that column – click the one in the first row, then Shift-click the one in the last row. Then type the value into the **Edit** dialog box and click the box with the check.

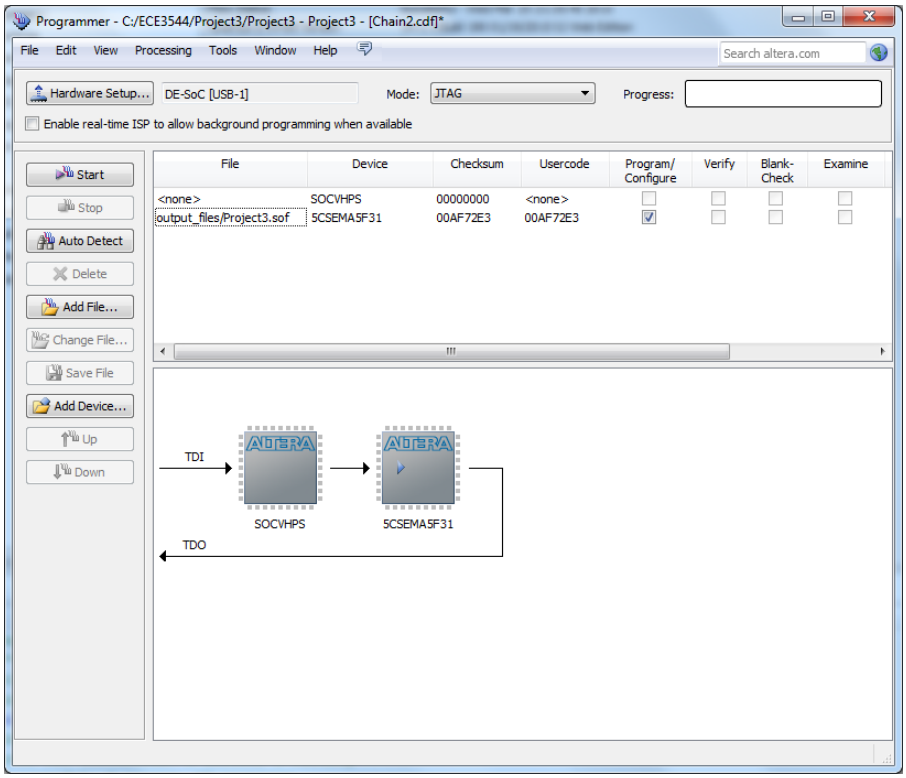
7. Double-check your pin assignments to make sure they are correct.
8. Close the Pin Planner.
9. Compile the project by selecting **Processing** $\rightarrow$ **Start Compilation** on the menu bar, or by double-clicking the **Compile Design** in the **Tasks** window. You should always re-compile a project after setting or making changes to the pin assignment.

Programming your DE1-SoC Board

Version 10 October 2015 (JST, TLM)

Do not perform these steps until after you have correctly assigned FPGA pins to your design! Follow the instructions in the document "Pin Assignment for the DE1-SoC Board." Programming your board with an incomplete or incorrect pin assignment could damage your board!

1. Connect your DE1-SoC board to a USB port on your machine. Turn on the board by pressing the power button – it's the red pushbutton below the AC adapter port.
2. In the menu bar, click on **Tools** → **Programmer**. This window should appear:



If you can see DE-SoC in front of the Hardware Setup button, you may continue. If you don't see it (you might still see No Hardware), then click on the Hardware Setup button. In the new window, select DE-SoC from the drop-down menu and close the window. Now you should be able to see DE-SoC in front of the Hardware Setup button.

If there is no DE-SoC option in the drop-down menu, it means the USB-Blaster driver is not installed on your machine. Review the instructions on the last page of the original installation instructions.

3. If you do not see the two devices shown in the diagram, click **Auto Detect** and select **5CSEMA5F31** when given the choice.
4. If it does not already have a checkbox, click the radio check box in the **Program/Configure** column for the .sof file that corresponds to your project's top-level module. If no file appears, double-click **<none>** under the **File**

column in the row for the 5CSEMA5F32 device. Your file is most likely to be in a subfolder of your work folder called **output_files**.

5. Click Start. Wait a few seconds while the bitstream gets downloaded to your board. All of the LEDs on the board should go dim while the programming is in process.
6. Now you can change the switch values and press the buttons to see the result of the counter on the LEDs. Verify that your counter is working.