Tutorials for Intel/Altera Quartus Prime 18.0

Krista Hill (kmhill@hartford.edu)

Copyright June. 22, 2018, V.09 Updated Sept. 17, 2019

Contents

1 Introduction

The Intel/Altera Quartus Prime package allows you to use schematics, hardware description language (HDL) files, and modules to design logic based systems. The purpose of these tutorials is to get students quickly using the software and associated hardware so the presentation is intentionally brief and to the point. We have found that the sooner students become familiar with CAD tools, the sooner they can start mastering the underlying concepts, that is they can learn to design and use logic circuits. Besides the Quartus Prime installation package, the following is also important.

- A Windows 7, 8 or 10 compatible PC, more details are in section 2. The Linux installation is not considered in this document.
- Programmable logic device module such as the CPLD module described in section 15 or a development board. Devices such as the CPLD module will need a JTAG programming cable such as a USB-Blaster, to configure the device. Development boards come with such logic already built in.
- If you are using a CPLD module then hardware such as a logic trainer or breadboard will be needed. The XMOD logic trainer documentation includes schematics so you can breadboard the trainer as well.

The tutorials and additional content are also available from our WEBPAGES

2 Installing Quartus Prime

You are welcome to install Intel/Altera Quartus Prime on your own computer. The installation material for the Lite Edition for Intel/Altera Quartus Prime is available at the following URL. For the Lite Edition there is no charge for the download and there is no license required. There are resources available to help you install Quartus.

- Download Intel/Altera Quartus Prime

  https://www.intel.com/content/www/us/en/software/programmable/quartus-prime/download.html
- Install tutorials for Windows

  https://www.youtube.com/watch?v=ROtJSPu8x50

  https://www.intel.com/content/dam/altera-www/global/en_US/pdfs/literature/manual/quartus_install.pdf

3 Quartus Project Directory, Moving Projects, and Such

In using Quartus it is important to keep in mind where your project is located. If you are using your own computer then the C: drive is a convenient location to store your Quartus projects. We will refer to this location as QDIR. Consider the following path for QDIR, where UNAME is your username.

C:\users\UNAME\Quartus

If you are using a University lab PC note that while you can use the location shown above, the location will be lost when the lab PC is rebooted. MORE TO COME...

4 Make New Project

This section outlines how to create a new Quartus project. Other parts discuss how to enter a schematic or VHDL description, perform a functional simulation and how to configure a CPLD. If you only need to configure a CPLD with some basic gates and are not yet interested in schematic capture then see section 16.

Start Quartus Prime and Make a New Project

- To start ISE, on the desktop click left to select the following:

  ```
  Start >> Intel FPGA 18.0...  >> Quartus (Quartus Prime 18.0)
  ```

- In the Quartus Prime project window, select the following to open the new project wizard, then click the Next button.

  ```
  File >> New Project Wizard...
  ```

- In the New Project Wizard pop-up window, select a working directory for the new project. Section 3 provides advice. Figure 1 shows an example where the working directory is on a USB memory stick. It is strongly suggested to not use the installation folder.

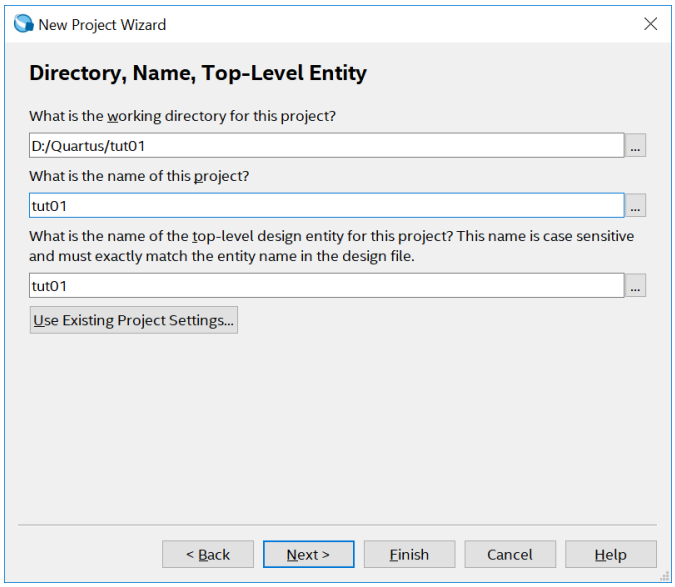


Figure 1: Assigning a working directory

- To create a new sub-directory, in the navigation window click the 'New folder' button. To enter a folder double click the file name. Select the desired folder name and click the 'Select folder' button. Note that Quartus uses the forward-slash (/) character is shown as the delimiter rather than backslash (\). Here the following path to the working directory is:

  ```
  D:/Quartus/tut01
  ```

- When you type in the project name field, text also appears in the top entity name field. The name of the top-level design entity is case sensitive and must exactly match the entity name in the corresponding design file. Click the 'Next' button. Here the project name is:

  ```
  tut01
  ```

- In the 'Project Type' pop-up window, here we select 'Empty project' and then click 'Next'. The use of a project template, not discussed here, simplifies how one makes a new project.

- The 'Add Files' pop-up window provides a simple way to include existing files into the project file list. For now click the 'Next' button.

- Figure 2 shows an example device selection window.

  - Family: MAX V
  - Package: EQFP
  - Pin count
  - Core speed grade: 4
  - Target device: Specific device selected in 'Available devices' list
  - 5M40ZE64C4 device is selected

- Click the 'Next' button

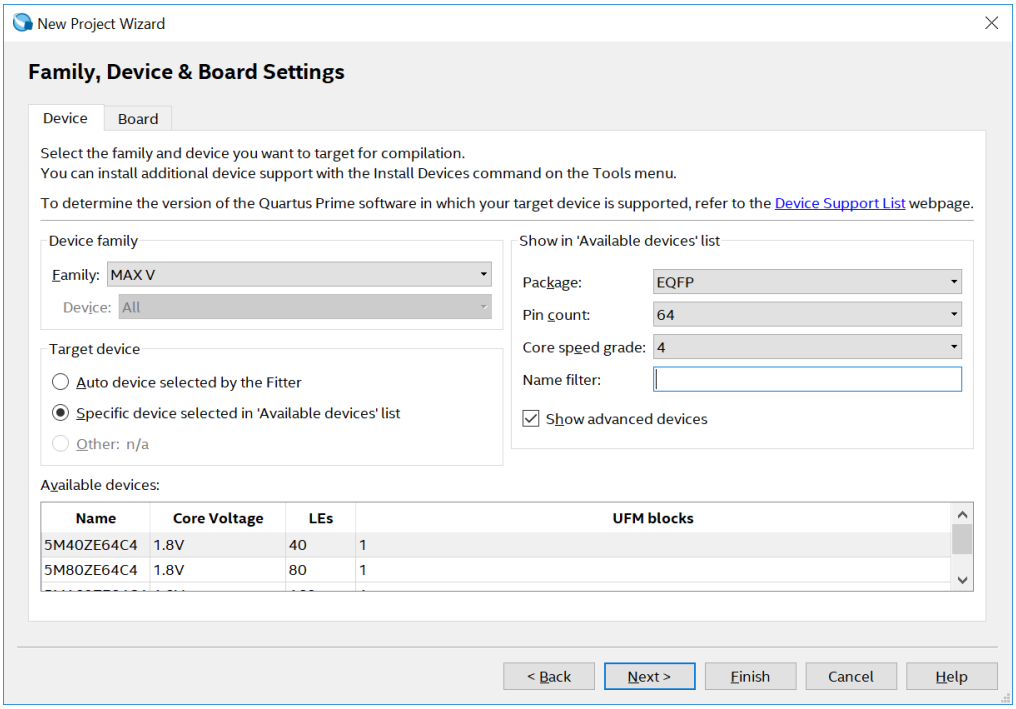


Figure 2: Assigning a programmable device

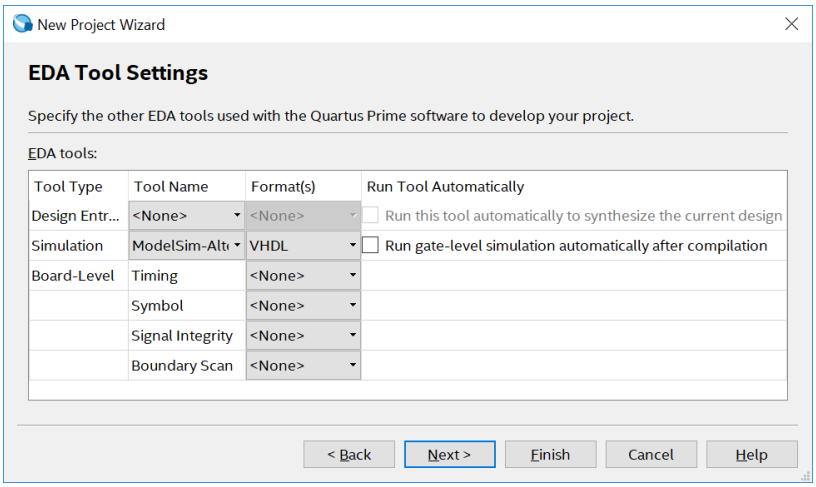


Figure 3: Assigning a tool program

- The EDA Tool Setting window is used to assign external tools. As shown in Figure 3, for simulation the following choice is shown. Click the 'Next' button.
  - Simulation Tool Name: ModelSim-Altera
  - Simulation Format(s): VHDL

- Review the summary list and then click the 'Finish' button the complete the new project.

5 Schematic Capture

This section introduces the block diagram and schematic capture tool. In following the quick-start theme, this document is intentionally not comprehensive. There will be much for you to discover as you become more familiar with the software.

- In the Quartus window, with the tut01 project open, select to make a new schematic.

  ```
  File => New
  ```

- In the New choice pop-up window, select the following, then click Okay.

  ```
  Design Files => Block Diagram/Schematic File
  ```

- At this point a new schematic page appears in the Quartus project window. To save the schematic as a new file select the following and then in the pop-up window check that the file name tut01 is entered, then click Save.

  ```
  File => Save As
  ```

- To make it easier to draw a schematic, detatch the block editor. Just above the schematic, in the tab tut01.bdf, right click and in the pop-up choice box pick Detach Window.

- Later, to reattach the block editor, either click the Reattach Window icon (far left) or select

  ```
  Window => Attach Window
  ```

  If the Window icon is not visible, either stretch the size of the block editor or use the extra icons button >> to access the Window icon.

- Figure 4 shows the top part of the schematic editor, which has three ways to enter commands: (1) Pull-down menus are accessed by a key-word such as File, Edit, View, and so on. (2) Icons are short-cuts to commonly used commands. Hovering the cursor over an icons cause a descriptive bubble to appear. (3) Hot-keys are control-key combinations that also serve as short-cuts to commonly used commands. Hot-keys are listed in the pull-down menus. You can make use of hot-keys, as you become more familiar with Quartus.

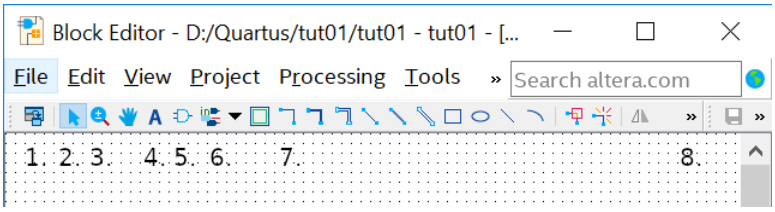


Figure 4: Top part of schematic editor window

Some of the icons shown in Figure 4 are numbered. The following outlines each such icon. As you read note that the editor has what's called a mode, which starts in the selection mode. Some commands cause what's called the mode to change, so that the shape and behavior of the cursor changes.

1. Reattach/Deattach Window – The reattaches the schematic editor window to the Quartus project window. This icon causes the same effect as selecting Window => Reattach Window or Window => Detach Window, respectively.
2. Selection Tool – Changes back to the selection mode. Pressing the escape key (Esc) has the same effect
3. Zoom Tool – Change briefly to the Zoom mode. Use the cursor to draw a box that the display will zoom to.
4. Text Tool – Change to the text entry mode. Use the cursor to indicate where new text will be inserted. More discussion on how to use this icon is presented soon.
5. Symbol Tool – Change to the symbol entry mode. More discussion follows.
6. Pin Tool – Change to the pin entry mode, for either an input, output, or bidirectional pin. More discussion follows.
7. Orthogonal Node Tool – Changes to the wiring mode, which allows the wires to be horizontal or vertical.
8. Display More Icon – Given the width of the window, this icon is used to represent icons that cannot be fit. To increase the size of a window, point the cursor at a window edge, press mouse button, move the cursor to resize, then release the mouse button.

If you haven't already tried the Reattach Window icon yet, give it a try and then detach the editor window a second time. Details of the rest of icons follows below.

Placing Components

The next step is to insert component symbols to look like what's shown in Figure 5. The inputs to the left are `Ai`, `Bi`, and `Ci`. The outputs to the right are `Co` and `So`. There are three AND gates, an OR gate, and two XOR gates.

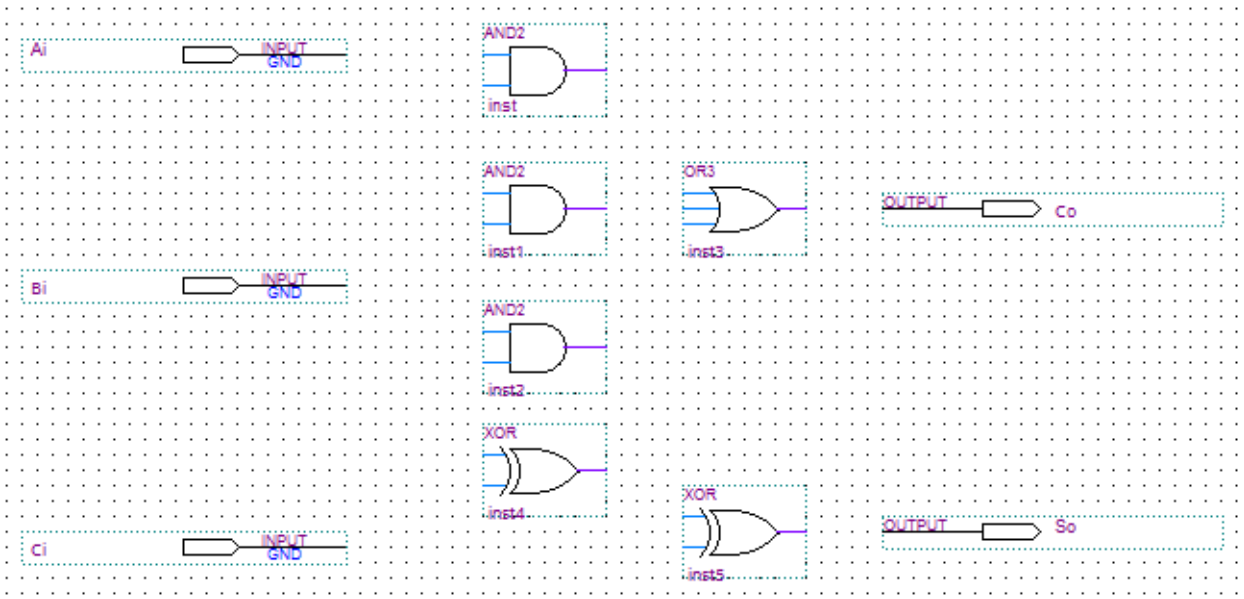


Figure 5: Components inserted into schematic

- Click the Symbol Tools icon (5.). In the Symbol window that appears, in the Libraries pane, click the right-pointing arrow to expand the Quartus library.
- Next, click the right-pointing arrow to expand the 'primitives' category. If necessary, scroll and click to expand the 'logic' subcategory.
- Scroll through the list of component, find AND2 and click on it. The corresponding symbol appears in the display window. Click OK, move the cursor to the schematic editor, and click to place the upper AND gate. Move the cursor and click to place the second AND gate, and then the third AND gate. Press the escape key to exit the Symbol Tool.
- Repeat the prior step to place an OR3 gate and then two XOR gate symbols. The 'logic' subcategory should still be open.
- Repeat the prior steps, but this time to open the 'pin' subcategory and place three instances of the 'input pin' symbol and two instances of the 'output pin' symbol. The Pin Tool icon (6.) down arrow provides a short-cut to placing pin symbols that also can be used.
- Double-click the upper-left pin to open the Pin Properties window. Under the General tab change the Pin name to `Ai` and change the Default value to `GND`.

- Repeat the prior step changing the other two pins to have the Pin names `Bi`, and `Ci`, respectively. Also change then pin names of the output pins to `Co` and `So`, respectively. Note that the output pins don't have default values.

- To arrange the components to look more like those in Figure 5, first press the escape key to make sure the editor is in the select mode. To move a component first place the cursor on a component, press the mouse button, and drag the mouse to move the given component. Next, release the mouse button to leave the component at the new position.

Next assign a label to each gate. Double click on one gate at a time, starting with the upper AND gate, and in the pop-up widow under the General tab enter text for the designator, U1. Repeat for the remaining gates, the result will look similar to Figure 6. As shown, the labels U1 to U6 are used.

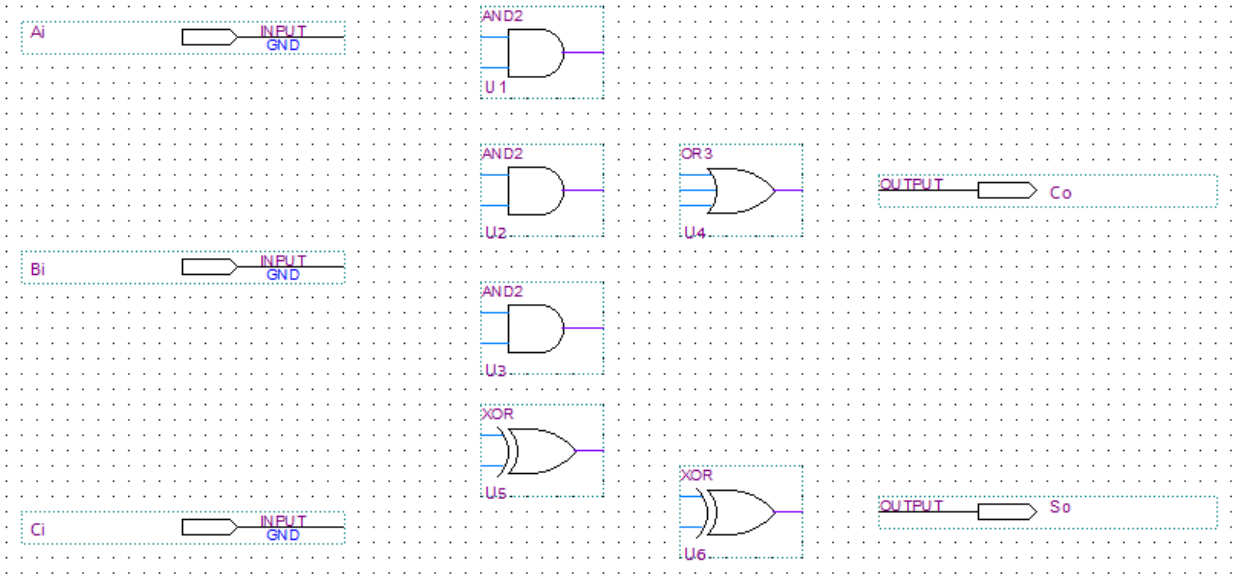


Figure 6: Components named in schematic

Wiring the Components

At this point you have a schematic editor open with components arranged in a manner similar to that shown in Figure 5. After wires are inserted the schematic will look similar to that shown in Figure 7. Each narrow line is a wire that connects between pins. Note that the wire connecting to the output So is highlighted as it was just inserted into the schematic. Your final schematic should not have any wires highlighted in this way. The dots are junctions which indicate that the corresponding horizontal and vertical wires are connected.

- Click the Orthogonal Node Tool icon to enter the wiring mode. Note the with the cursor in the schematic window, in the wiring mode the cursor will have a cross-hairs and bent wire as a symbol.

- To start a wire connection, point the cursor cross hairs at a connection point and press and drag the mouse. Start by pointing at the right end of the Ai signal pin and press the mouse left button. Next, drag the mouse to

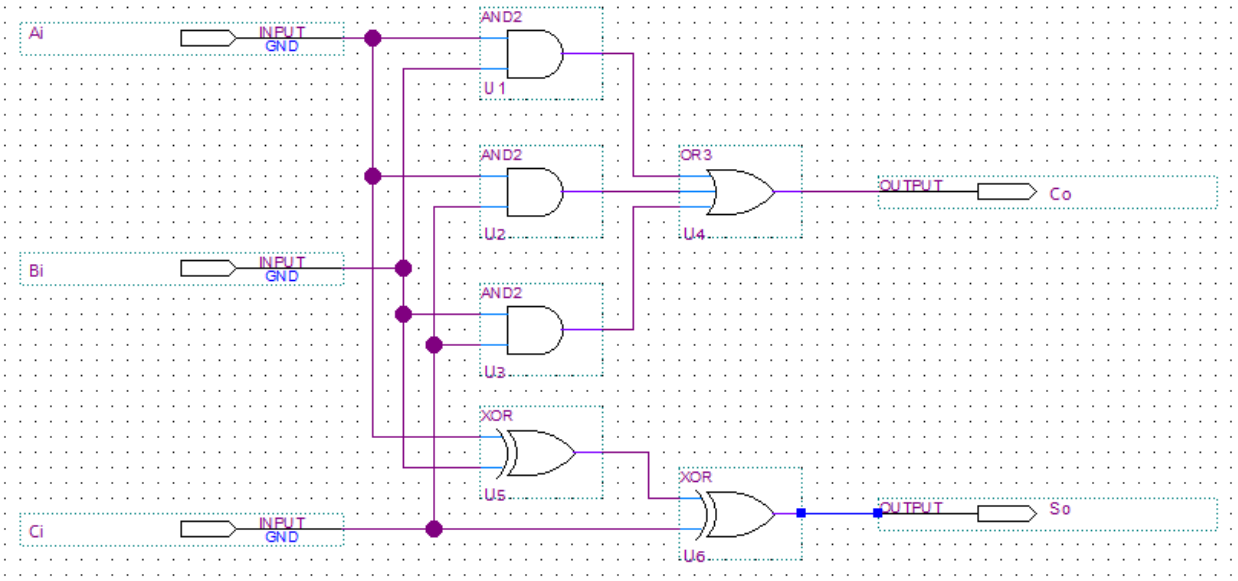


Figure 7: Schematic with components and wires

the right so the cursor moves to the upper input pin of the upper AND gate. When the cursor reaches possible connection point a small box appears. Release the mouse button to complete the connection. A newly placed wire will be highlighted in blue for a time.

- You form a wire junction by pointing at a wire and placing the wire as you did before. This time start a wire just to the right of where the Ai pin connects to the wire. Have the wire connect to the upper pin of the second AND gate. Note that this tool inserts wire segments that are either horizontal or vertical.
- If necessary, you can briefly leave a wire dangling by releasing the mouse button, then press and hold the mouse button to continue wiring.
- To select a wire first press the escape key or click the Select icon (2.) to return to the select mode. Point the cursor at a wire and click the mouse once, so that only the one wire segment is selected. Try this again but double-click instead. Note that the entire wire segment is selected.
- To delete a wire segment, first return to the select mode, point and click to select a wire, then press the delete key on the keyboard.
- Insert the remaining wires so your schematic looks similar to that shown in Figure 7.

Save and Check Your Work

- Save your schematic File => Save
- Attach the schematic editor back to the project window
- In the Tasks pane, click to expand the Compile Design then double click on Analysis & Synthesis
- After a few moments some results will appear below in the output pane. If you see any errors then go back and fix them. For example if a wire were missing for the U1 AND gate, a message like the following should appear

12009: Node "U1" is missing source

Quartus Prime Analysis & Synthesis was unsuccessful. 1 error, 1 warning

Insert Title Information

It is important that you insert title information into your schematics. To help others it is important to know what the schematic is, who made it, and when the schematic was made. For larger schematics you can use a title component, but for a modest schematic like the full adder, insert at least the following information as shown in Figure 8.

- A short description of what the schematic describes
- The file name
- Your name and the date

To insert text into the schematic click on the Text Tool icon (4.) then position the cursor in your schematic and click the mouse to place the text cursor, so you can type text. Once you have finished typing your text, click the Select Tool icon to place the text. With the text selected in a blue box, the box and text can be moved as you like.

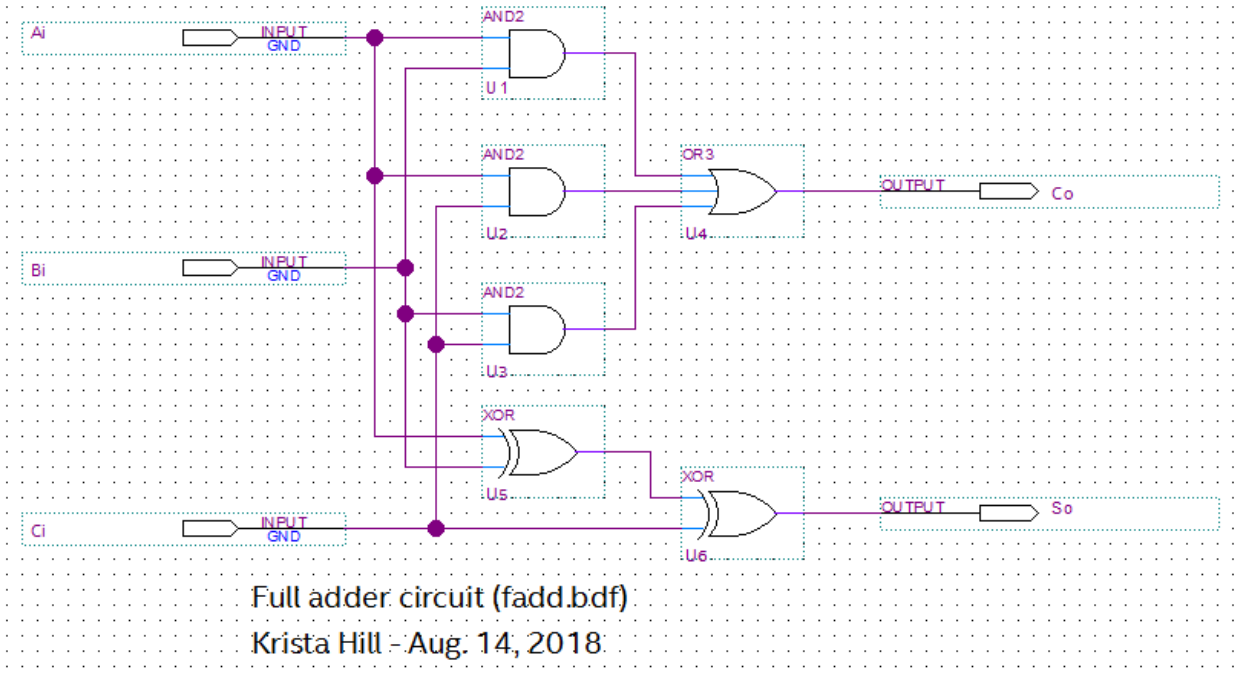


Figure 8: Schematic with title information

Save and produce output

To save your work, under the title bar for the schematic editor select File => Save

To produce a schematic that you can use in a word processed document consider producing a screen capture.

- To improve the resolution of the screen capture, it is important to make the image large. Resize the schematic editor window to make it large in the screen.

- Zoom out slightly View => Fit in Window
- Click the Zoom Tool icon (3.). Next position the cursor at a corner of what you want to zoom into, press and hold the left mouse button. Then drag the cursor to the opposite corner and release the mouse button. The screen will zoom to show your selection.
- You can use the snipping tool to capture the image you created in the schematic editor. (Start => Accessories => Snipping tool)
- Otherwise, you can click to select the schematic editor window, press the Alt and PrtScn keys together to capture the window.
  - Open Paint (Start => Accessories => Paint).
  - Within Paint click the Paste icon to paste in the window image, then click the Select icon and draw a box around your schematic, then click the crop button to keep only the schematic
- At this point you can save the image or just copy and paste into a word processed document.

6 Making a VHDL Design File

VHDL is a hardware description language that uses text files to describe logic systems. If you just entered a schematic and want to perform a simulation, then proceed to section 7. This section considers an alternative to schematic capture. From within a new project, such as the tut01 project outlined in section 4, do the following:

- Select: File => New... Then in the New pop-up window select Design Files => VHDL File and click OK.

After a few moments a new text editor window opens. Save the new file with a name:

- Select: File => Save As... Then in the Save As pop-up window enter tut01.vhd in the File name field and click Save
- At this moment you have a blank file. For your convenience you can detact the text editor from the Quartus application window and later reattach. Figure 4 from the Quartus application window, icon 1. is the Detach Window icon, but in the detached window, icon .1 causes the window to be reattached to the Quartus application window.
- Enter the following code into your text editor, then be sure to save your work File => Save

```
----------------------------------------------------------
-- tut01.vhd - Krista Hill - 9/9/2018
-- Full adder
----------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity tut01 is
  Port (ain,bin,cin : in STD_LOGIC;
        cout,sum : out STD_LOGIC);
end tut01;
architecture arch of tut01 is
begin
  cout <= (ain and bin) or (ain and cin) or (bin and cin);
  sum <= (ain xor bin) xor cin;
end arch;
```

- At this point, Quartus application window task pane, the task view in the center should be set to Compilation. Under Compile Design, double click on Analysis & Synthesis. After a few moments having a green check-mark to the left that task indicates success. Otherwise, if there are any errors, then review the messag pane at the bottom and go back and correct the errors.

- For relatively simple VHDL modules, the simulation described in section 7 suffices. Otherwise, section XX outlines how to use ModelSim in a direct way.

7 Performing a Simulation

Implementing a design involves assigning pins, wiring up a breadboard, and then configuring a device. Before putting in that effort, it's prudent to perform a simulation to perform a simple sanity check. The idea is to focus your attention where it's needed. In addition, performing a simulation can give you a better feel for what you're expecting the behavior of your circuit to be.

To simulate a logic circuit it's necessary to create a list of input values, called test vectors, that will be displayed in a waveform. To assign the input values we will use the waveform editor. You will be assigning eight different sets of input values (test vectors) and hold each set for 100ns, as shown in Figure 9

Do the following:

- Open the tutorial from sections 4 and 5. If you have not done so already, perform the Analysis & Synthesis step so that a green check-mark appears in the task status column.

- In the main Quartus project window select: File => New... Then in the pop-up window select under the heading Verification/Debugging Files click to select University Program VWF, then click OK.

- The Simulation Waveform Editor window will appear. To give the file a name select File => Save as... Then in the pop-up window, the text in in the File name field will be the file name. Click OK to accept the default value.

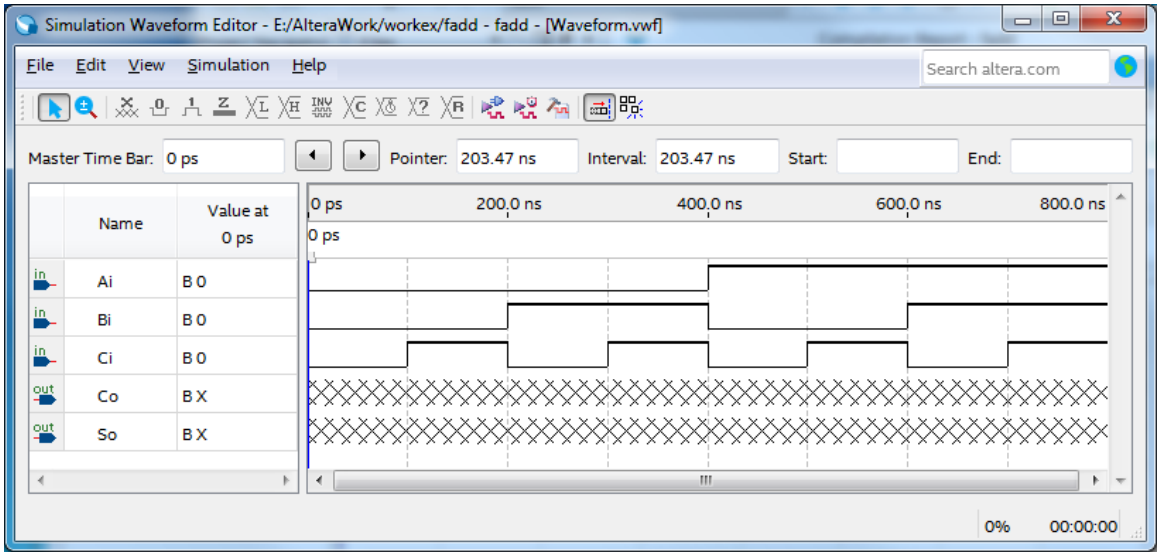


Figure 9: Simulation test vector waveforms

- Select Edit => Grid Size... In the pop-up window Grid Size, set the value to 100.0 and set the unit pull-down to ns for nano-seconds. Then click OK.
- Select Edit => End Time... In the pop-up window End Time, set the value to 800.0 and set the unit pull-down to ns for nano-seconds. Then click OK.
- Select Edit => Insert => Insert Node or Bus... In the pop-up window, click the Node Finder button. In the Node Finder pop-up window click the List button. In the Nodes Found pane click on the top-most signal then hold down the shift key, point at the bottom signal, and left click the mouse. All the signals will be selected. Click the > button to move the signals to the Selected Nodes pane. Click OK. In the Insert Node or Bus window click OK.

Assign Input Values

At this point the Simulation Waveform Editor should display the inputs (Ai, Bi, Ci) and output signals (Co, So). No input values have been assigned yet and the output values have not been produced yet. Figure 10 points out some simulation buttons. At this point the Snap to Grid mode button (6.) should appear with darkened outline and the Snap to Transition mode button (7.) should have a lighter shade outline, as shown. The following is a summary of the icons.

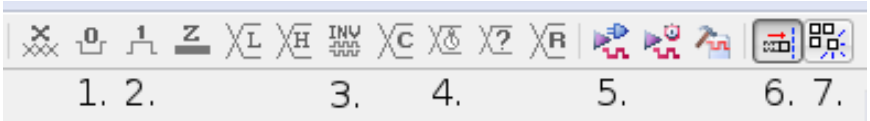


Figure 10: Some simulation buttons

1. Assign low '0' value to selection

2. Assign high '1' value to selection
3. Invert bit value to selection
4. Overwrite with Clock signal values
5. Run functional simulation
6. Snap to Grid mode
7. Snap to Transition mode

In the following you will consider how the Assign high, Invert, Assign low, and Overwrite with Clock icon each behave.

- To the right of the signal Ai in the waveform area, press and drag the mouse from 400ns to 800ns and release. Doing to will select that time range. Next, click the Assign high icon (2.), to assign high to the selected region.
- To the right of the signal Bi select the waveform from 200ns to 800ns as you did before. This time click the Invert icon (3.). Next, click in a non-selected area and then select the same waveform from 400ns to 600ns and then click the Assign low icon.
- Finally, click on the signal name Ci so that it's selected, then click the Overwrite with Clock icon (4.). In the pop-up clock window change the period value to 200.0 ns, then click OK.
- At this point the Simulation Waveform Editor should appear similar to that in Figure 9. Save your work with File => Save

Simulation and Results

A so-called Functional Simulation essentially checks the logic of a circuit, without other considerations such as propagation delay.

- To perform the simulation either select Simulation => Run Functional Simulation, or click the Run Functional Simulation icon (5.). After a few moments a new Simulation Waveform Editor appears. Examine the waveforms to verify that they are correct.
- To produce output for a word processor first pull the bottom of the window down slightly to have some space at the bottom for text.
- With the display window in focus, press the Alt-PrtScn key combination to capture the window image.
- Open the Paint program (Accessories => Paint), click the Paste icon to paste in the display image. In the tools area, click the A icon to change to the Text mode. Move the cursor below the waveforms and click left to open a New Text field. The in the following text in one or more lines.
  - Your name
  - Brief description of the simulation
  - The date when you produced the simulation

- Click in a nearby non-selected area to exit the Text mode. In the Image area click on the Select icon so the cursor changes to cross-hairs use the cursor to draw a box on the image you want to save, then in the Image area click the Crop icon. The result will look like that shown in Figure 11.
- At this point you can either save the image to a file or click the Select button, draw a box, and click the Copy button. With the image copied to the clipboard you can paste the image into a word processed document.

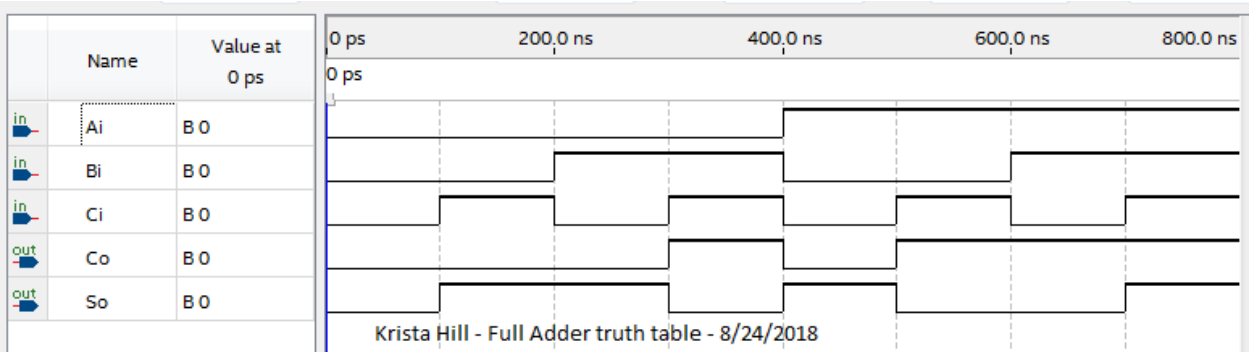


Figure 11: Resulting simulation

Later, to reopen your simulation, in the Quartus Project Navigator pane in the center click on the view tab, to change from Hierarchy to Files View. At this point you can double-click to open your simulation .vwf file. To change back to the Hierarchy view, change the view tab.

8 Implementing the Circuit

There are module pins (Mod.) which are plugged into a breadboard as well as device pins (Dev.) for the CPLD. In using Quartus, it's the device pins that are assigned. To tell the types apart, module pin designators each start with the letter P. Device pin designators for our CPLD are each just a number. The assignments to use for the full adder circuit are in Table 1. The complete CPLD module pin list is in section 15.

Table 1: Pin assignments for full-adder example

Signal	Mod.	Dev.	Signal	Mod.	Dev.
Ai	P1	19	Co	P56	18
Bi	P2	20	So	P55	13
Ci	P3	21			

- In the Quartus project window, in the Tasks pane expand the Compile Design step and double-click the Analysis & Synthesis step. After a few moments a green check-mark should appear to the left of the step.
- In the Quartus project window select Assignments => Pin Planner. In the Pin Planner window the Node Name column lists the signal names. You will put the device pin numbers in Location column, in the corresponding row, to look like that shown in Figure 12. Start with Ai, type 19 into the corresponding Location row, then press Enter. Note that the entry is automatically changed to PIN_19. Likewise, type in the device pin number for the remaining signals.

Node Name	Direction	Location	I/O Bank	Fitter Location	I/O Standard	Reserved
Ai	Input	PIN_19	1	PIN_19	3.3-V LVTTL	
Bi	Input	PIN_20	1	PIN_20	3.3-V LVTTL	
Ci	Input	PIN_21	1	PIN_21	3.3-V LVTTL	
Co	Output	PIN_18	1	PIN_18	3.3-V LVTTL	
So	Output	PIN_13	1	PIN_13	3.3-V LVTTL	
<<new node>>						

Figure 12: Pin Planner pin assignment pane for full adder

- Under the Pin Planner title bar, click File to see the choices. If there is a 'Save' choice then select that. Otherwise, you can make a record of your pin assignments. Select File => Export... and then in the pop-up Export window, click Export to accept the default file name fadd.csv.
- Close the Pin Planner window File => Close

Compiling the Design

Back in the Quartus project window, in the Tasks pane title, the task view tab in the center should be set to Compilation.

- Double-click on Compile Design to completely implement the logic circuit.
- You can view the progress of the various compilation steps as it proceeds. After some time a green check-mark will appear next to the Compile Design step.

Wiring the CPLD Module

Figure 13 shows the wiring diagram used to connect the CPLD module to the logic trainer. The SW items are data source switches, the LED items are logic indicators, VCC is 3.3Volt power and GND is ground. Once the CPLD module is wired into the trainer, it's necessary to connect the USB-Blaster module used to configure the CPLD, as shown in Figure 14. The connector to the USB-Blaster is keyed to ensure that it connects correctly. At this point the drivers should already be installed in the PC for the USB-Blaster. Otherwise, update the USB-Blaster driver from the directory:

```
<QuartusInstallDir>\drivers\usb-blaster
```

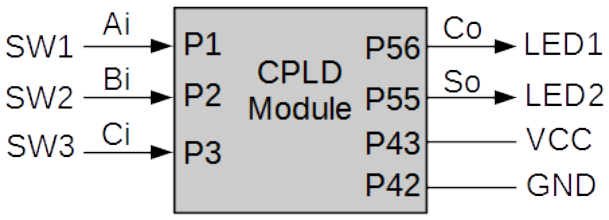


Figure 13: Wiring diagram to test full-adder circuit

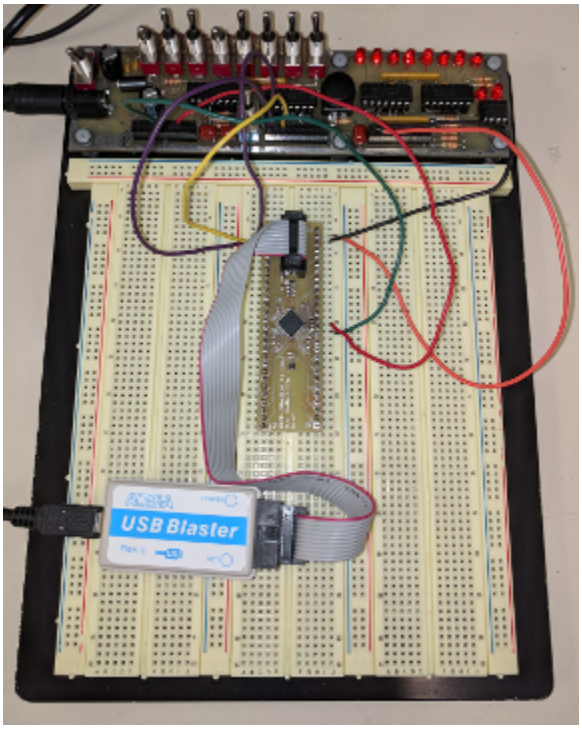


Figure 14: CPLD module wired as full-adder into logic trainer and USB-Blaster

With the CPLD module wired to trainer, the USB Blaster connected between the PC and CPLD module, and the power adapter in place, turn on the power switch (upper left of trainer).

- In the Quartus project window Task pane scroll and double click on 'Program Device (Open Programmer). After a moment the Programmer window will appear.
- Under the Programmer title bar click on 'Hardware Setup...' In the Hardware Setup window, in the Available hardware items double-click to select 'USB-Blaster'. The Currently selected hardware field should now indicate USB-Blaster. Click Close.
- The Mode choice should be set to JTAG.
- In the pane below the Mode choice, in the output_files/fadd.pof click on the Program/Configure button.
- In the pane to the left, click the Start button. In a few moments the CPLD should be configured. Close the Programmer window.
- Manually test the board, assign each input combination and verify the output.
- With the CPLD configured it will remain configured after power is removed. Disconnect the USB-Blaster, reapply power, and manually test the circuit.

9 Hierarchical Design

The use of hierarchy in design is a method of using simpler circuit descriptions to make successively more complicated ones. A block diagram is a form of technical communication that can convey a sense of hierarchy. Here we consider hierarchy instance diagrams as well as hierarchy dependencies diagrams. To illustrate, consider how six soda cans are gathered into a six pack. Figure 15 shows six *instances* of soda cans. Note the use of levels. At level 1 we can see that there are six cans involved.

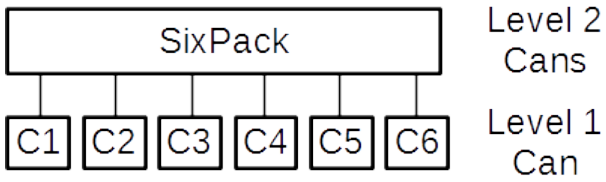


Figure 15: Hierarchy instances diagram for six pack of soda

By constrast, Figure 16 shows a dependency. Here, generic block (Cx) describes a signle can. At level 1 it's not shown how many cans there are. It's Level 2 that determines how many cans are involved. So, once we've described what a can is, we can collect cans to make a six pack.

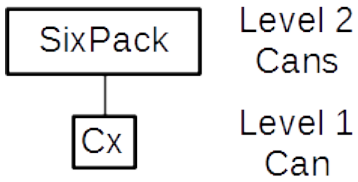


Figure 16: Hierarchy dependencies diagram

Symbols are the mechanism to use hierarchy in a schematic. To start, Quartus has a number of pre-made symbols that you can use. In the Symbols choices window, of particular in the primitives category is the storage subcategory which contains various flip flops; see section 11 for more details. Also, in the others category, the maxplus2 subcategory contains some symbols equivalent to 74-type devices as well as some generic components. Table 2 lists several medium scale integration (MSI) type device symbols.

Table 2: Some decoders and multiplexers

Decoders		Multiplexers	
Part number	Description	Part number	Description
7442	four to ten	161mux	sixteen to one
74138	three to eight	74151	eight to one
74139	dual two to four	74153	four to one
74154	four to sixteen	74157	two to one
74155	dual two to four		

10 Making and Using a New Symbol

This section outlines how to make a full-adder symbol that represents the tut01 schematic given in section 5. The following sections outline how to use the symbol in hierarchy either with or without a bus signal. For this exercise we'll start with a blank project.

- Make a new project named Add3nob in a folder with that name, by following the steps in section 4.
- If you have the tut01 full-adder schematic in another project, then:
  - In the project window, select File => Open. In the pop-up window navigate to the tut01 schematic, then select it, click open.
  - Select File => Save As. In the pop-up window navigate back to the Add3nob project folder. In the File name field change the entry to fadd, click Save.
  - Select Project => Add Current File to Project.
- If you don't have the tut01 full-adder schematic in another project, then follow the steps in section 5 to make a new schematic in this folder, but name the schematic fadd instead.
- In the Project Navigator pane change the view from Hierchy to Files. Right-click on fadd.bdf and select 'Set as Top Level Entity'.
- In the tasks pane, perform the Analysis & Synthesis step to verify the correctness of the fadd schematic
- In the Quartus window select File => Create/Update => Create Symbol Files for Current File. In the pop-up window click Save then Okay. This step could also be used to make a new symbol for a VHDL design unit or that for Verilog.

The placement of the input and output pins can make it easier to use a symbol. At this point it's wise to use the symbol editor to review the symbol and if necessary make some adustments.

- In the Quartus window select File => Open. In the pop-up window, to the right of the File name field, change the filter type to All Files (*.*). Select fadd.bsf and click Open. The symbol editor opens with the fadd symbol. Right click on the fadd.bsf file tab and select Detach Window. Figure 17 shows the top of the editor window.

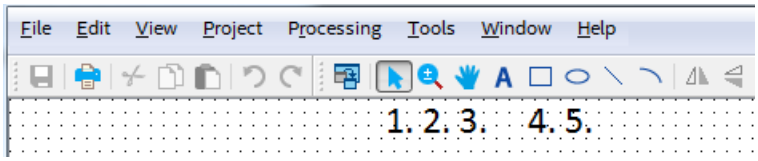


Figure 17: Top of symbol editor window

- Click the Zoom tool (2.) then in the editor window move the cursor to a location just above and to the left of the fadd symbol. Press the mouse button and drag to a location just below and to the right of the symbol. When you release the mouse button the display will zoom to acccomdate your choice. Click the Select icon (1.) to exit the zoom-in mode.

- The input signals are shown to the left of the symbol and the outputs are to the right.
- The hand tool (3.) is used to pan the editor view. The rectangle tool (4.) and oval tool (5.) are each used to draw each respective shape.
- The placement of each signal name is relative to it's corresponding pin. Once a signal name is positioned the way you like it relative to its pin, to move the pin click and drag its X symbol. The resulting symbol is shown in Figure 18

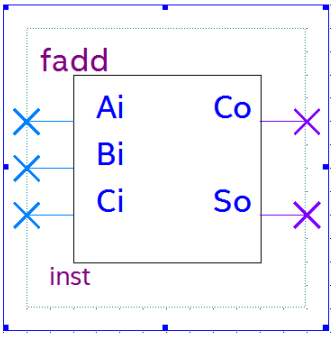


Figure 18: New fadd full adder symbol

Save File => Save and then close File => Close the symbol editor. Note that once you start actually using a symbol in a schematic, you can open the symbol editor from within the schematic.

Using a Symbol in a schematic without busses

This section outlines how to draw the adder circuit for a pair of three bit values, shown in Figure 19. This schematic does not make use of signal busses. The next subsection introduces the use symbols and busses.

- In the Quartus window open a new schematic; select File => New and in the pop-up select Design Files => Block Diagram/Schematic File, click OK. Select File => Save As, then in the pop-up window the File name field should have the value Add3nob. Click Save.
- At this point be sure to assign Add3nob as the Top-Level Entity. In the Project Navigator pane, change the view to Files. Click to select Add3nob.bdf then right click and pick Set as Top-Level Entity. At this point it helps to detach the scehamtic capture window and resize/stretch the borders as you did before and if necessary you can zoom-out with File => Zoom Out.

In the next few steps you will insert fadd symbols and pins, to look like those shown in Figure 19.

- To insert the full adder fadd symbol, look into the Project library. Insert three instances of the full adder fadd symbol and arrange the symbols in a column pattern near the center of the page. The ground symbol GND is in the Primitives => Other category.
- For each fadd symbol first click to select it, then right click and select Properties. In the general tab enter and instance its Instance name then click OK. From the top-most to the bottom-most fadd symbol the names are F3, F2, and F1, repsectively.

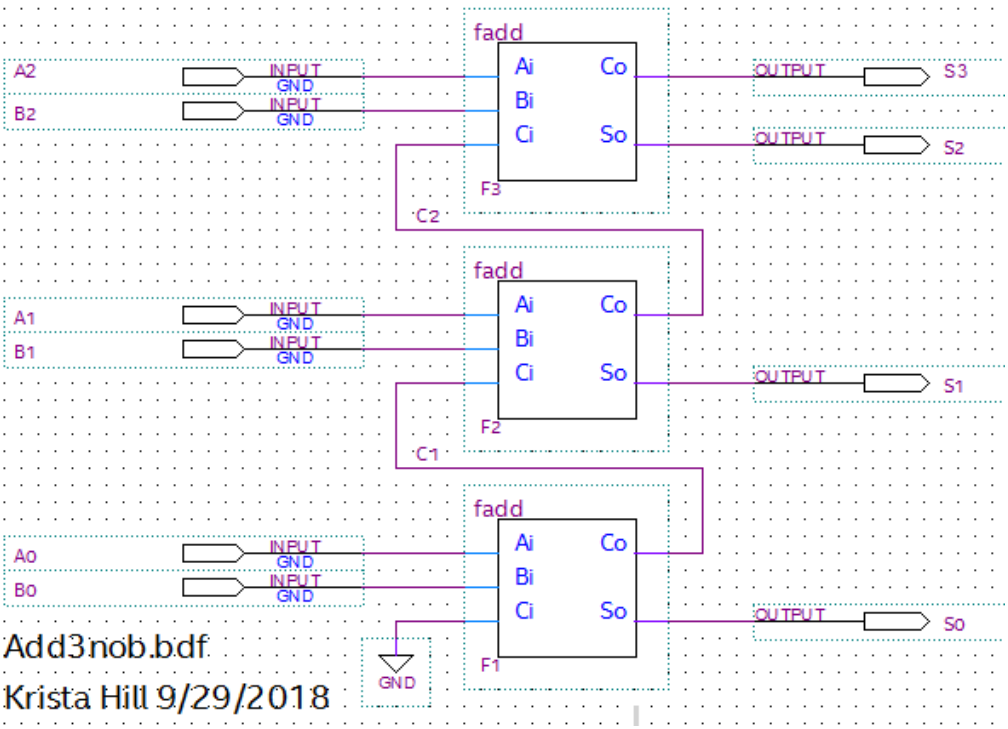


Figure 19: Adder without busses for pair of three bit values

- Insert input pins for the inputs A2, A1, A0 as well as B2, B1, and B0, as you did in section 5. Thet default value for each input pin is set to GND. Likewise, insert output pins S3, S2, S1, and S0.

In the following you will finish the schematic.

- Insert wires for the input and output pins as well as the ground symbol, as you did with the schematic in section 5. Between pairs of fadd symbols connect a wire from the lower Co pin to the next symbol Ci pin. Click to select the horizontal wire segment between the pairs of adjacent fadd sybols, then right click and pick Properties. In the Node Properties General Tab, in the Name field enter C1, then click OK. Repeat for the C2 wire as well.

- Insert text for the file name, your name, and the date. Save your work.

Using a Symbol in a schematic with busses

The use of busses is very helpful for modules having many related input and output signals.This section outlines how to draw the adder circuit for a pair of three bit values.

- In the Quartus window open a new schematic; select File => New and in the pop-up select Design Files => Block Diagram/Schematic File, click OK. Select File => Save As, then in the pop-up window the File name field should have the value Add3nob. Click Save.

- At this point be sure to assign Add3nob as the Top-Level Entity. In the Project Navigator pane, change the view to Files. Click to select Add3nob.bdf then right click and pick Set as Top-Level Entity. At this point it

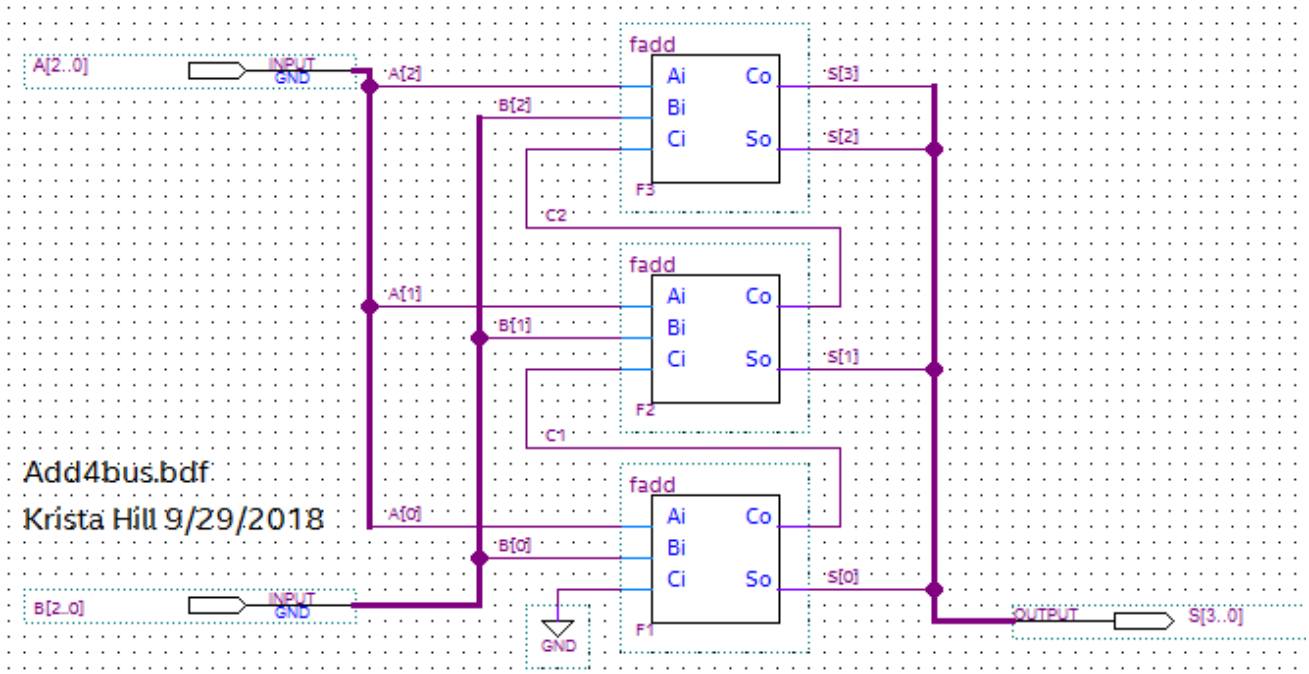


Figure 20: Adder with busses for pair of three bit values

helps to detach the scehamtic capture window and resize/stretch the borders as you did before and if necessary you can zoom-out with File => Zoom Out.

In the next few steps you will insert fadd symbols and pins, to look like those shown in Figure 20.

- To insert the full adder fadd symbol, look into the Project library. Insert three instances of the full adder fadd symbol and arrange the symbols in a column pattern near the center of the page. The ground symbol GND is in the Primitives => Other category.
- For each fadd symbol first click to select it, then right click and select Properties. In the general tab enter and instance its Instance name then click OK. From the top-most to the bottom-most fadd symbol the names are F3, F2, and F1, repsectively.
- Insert input pins for the bus inputs A[2..0], as well as B[2..0], as you did in section 5. Thet default value for each input pin is set to GND. Likewise, insert output pins S[3..0].

In the following you will finish the schematic.

- This time rather than using wires to connect pins, use the orthogonal bus tool to insert busses for the input and output pins in a manner similar to that in section 5. It is important to name each bus tap, which is an attached bus or wire to make it clear how it connects into the bus. Click to select a given wire then right click and pick Properties. In the General tab name field enter the wire name. For example the upper fadd input Ai connects to the tap named A[2].
- Between pairs of fadd symbols connect a wire from the lower Co pin to the next symbol Ci pin. Click to select

the horizontal wire segment between the pairs of adjacent fadd sybols, then right click and pick Properties. In the Node Properties General Tab, in the Name field enter C1, then click OK. Repeat for the C2 wire as well.

- Insert text for the file name, your name, and the date. Save your work.

11 Flip Flops and State machines

The Quartus library has premade symbols for common flip-flops including D, T, and JK type. The following some common flip-flops in the Primitives => Storage library. For our purpose here, we consider D-type flip-flops without enable or synchronous clear.

- `dff` – D type, rising clock edge sensitive, asynchronous negative logic preset (PRN) and clear (CLRN)
- `jkff` – JK type, rising clock edge sensitive, asynchronous negative logic preset (PRN) and clear (CLRN)
- `tff` – T type, rising clock edge sensitive, asynchronous negative logic preset (PRN) and clear (CLRN)

An alternative is to use a text description of a flip-flop like the file DFlipFlop.vhd given below. Start with File => New => VHDL FIle. Copy in the given code and save (File => Save As...) that as DFlipFlop.vhd, then follow the new symbol notes in section 10 to make a new symbol.

- This flip-flop is sensitive to a rising clock (clk) and has asynchronous positive logic clear (C) input.
- At a rising clock edge the data input (D) is assigned to the state (Q) signal.

```
-------------------------------------------
-- DFlipFlop.vhd - 10/25/2018
-- D-type FlipFlop, async. pos. logic clear
-------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity DFlipFlop is
   Port ( D,C,clk : in  STD_LOGIC;
          Q : out  STD_LOGIC);
end DFlipFlop;
architecture Behavior of DFlipFlop is
begin
   process(C,clk)
   begin
      if C = '1' then
         Q <= '0';
      elsif clk'event and clk = '1' then
         Q <= D;
      end if;
   end process;
end Behavior;
```

The schematic for a simple state machine is shown in Figure 21. Apart from clock and clear, the input is Xi and the output is Yo. The schematic is drawn using the techiques described in section 5. Any symbols you made will be in the project library. The flip-flop has been assigned the instance name '`ff1`'. The next-state eqation (1) and output equation (2) are produced by inspecting the schematic. The corresponding state diagram is shown in Figure 22.

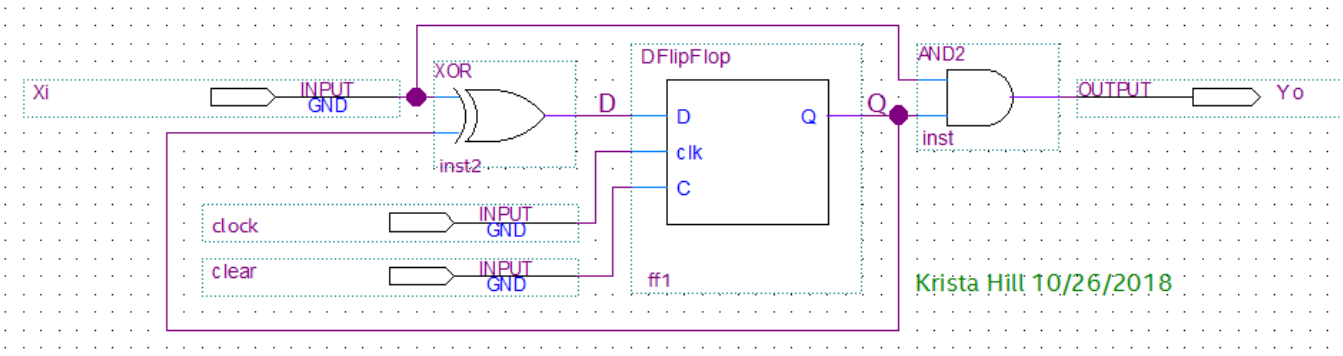


Figure 21: Example state machine

$$Q^{+} = D = Xi \oplus Q \tag{1}$$

$$Yo = Xi \cdot Q \tag{2}$$

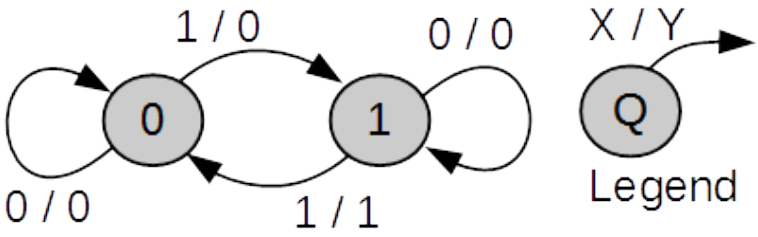


Figure 22: State diagram for example state machine

The following are some key points for performing a simulation like that shown in Figure 23.

- The grid size and clock parameters are below. Note that the input Xi cannot change value at the same instant as a rising clock edge. Also, in a clock cycle there are two moments when the input Xi can change value which helps to make the behavior of a Mealy type state machine clearly visible. Use the techniques outlined in section 7 to assign the clock as well as the waveforms for the clear and Xi input signals.
  - Grid size here is 50 ns
  - Clock period is 100 ns
  - Clock offset is 25 ns so that at 0.0 ns the clock phase is 0.25T.
  - The clear signal becomes low when the clock phase is 0.75T.

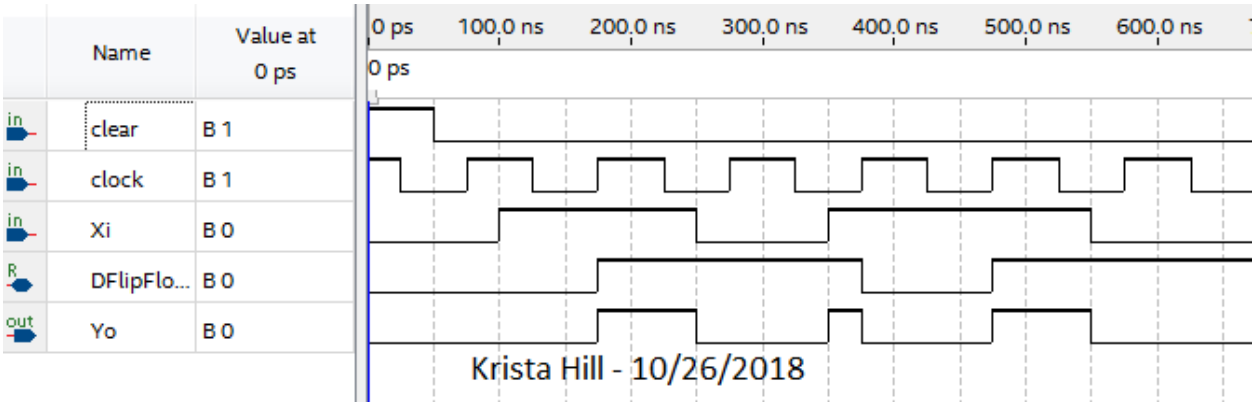


Figure 23: Timing diagram for example state machine

- The state value Q is not assigned to a pin. So to insert Q into the Simulation Waveform Editor, open the 'Insert Node or Bus' window (Edit => Insert => Insert node or bus) as usual. Click the 'Node Finder' button and in the 'Node Finder' window change the Filter field choice to 'Design Entry (all names) and click 'List'. The flip-flop state is named DFlipFlop:ff1|Q, add this signal name to the waveform display.
- To capture the image press the key combintion Alt-PrtScn. Open a drawing program such as paint, paste in the image, insert your name and date. Click the Select icon and draw a box about the are you want to use. Click the copy icon then in your word processor paste in the image..

12 Counter Register and Hierarchy

A counter register is a state machine, but while the register can be described in terms of a transition table, transition diagram, and such, the excitation table best describes its behavior. To produce our excitation table we will first consider the following components.

- DFlipFlop - D type flip-flop, described in section 11.
- HalfAdd - Half adder, adds a pair of bits
- Mux2 - Two data input multiplexer

The excitation table in Table 5 neatly sumamrizes the behavior of a counter and is more understandable than a transition table. The variable '`Q+`' refers to the next value, following an active clock edge. The symbol '`d`' refers to the 'dont-care' condition.

Half Adder (HalfAdd)

A half adder adds a pair of bits (Ai,Bi) producing a two bit sum (Co,So). The truth table for the half adder is in Table 3. By inspection you should be able to see that the circuit is just two gates, an AND as well as an XOR gate. The corresponding schematic is shown in Figure 24.

Table 3: Half Adder Function Table

Ai	Bi	Co	So
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

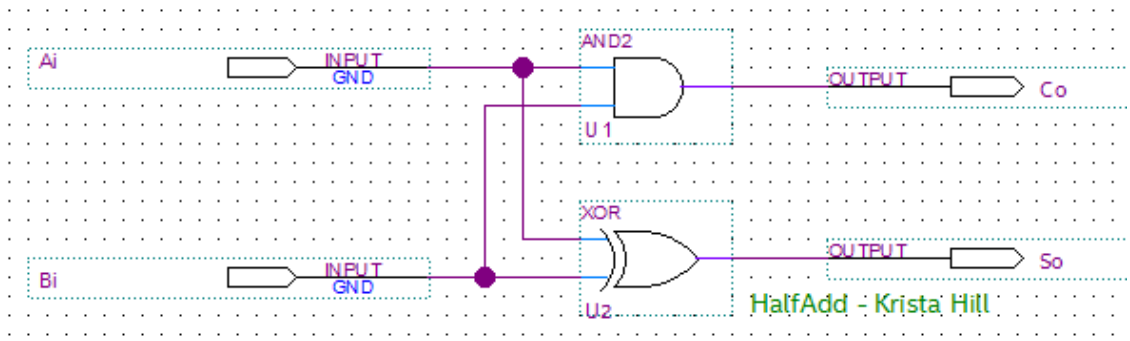


Figure 24: Half adder (HalfAdd) schematic

Two Input Multiplexer (Mux2)

A multiplexer is a switch that selects its output (Fo) from one of several data inputs (A0,A1). The function table for a two data-input multiplexer is in Table 4. Unlike a truth-table a function table can contain variables which makes it easier to see the behavior that's being described. The corresponding schematic is shown in Figure 25.

Table 4: Multiplexer (Mux2) function table

S	Fo
0	A0
1	A1

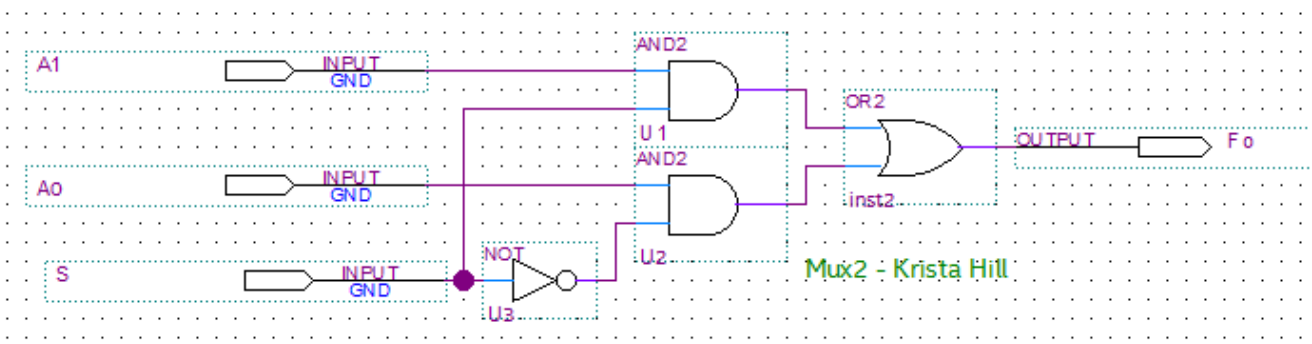


Figure 25: Multiplexer (Mux2) schematic

Counter Register

An excitation table describes the synchronous behavior of a sequential circuit. To produce the excitation table shown in Table 5, consider the circuit shown in Figure .26. The multiplexers produce the flip-flop D inputs.

- When S is low, the half adder outputs (S2,S1,S0) are selected
- When S is high, the P inputs (P2,P1,P0) are selected

The half-adders are connected in cascade to form an incrementer. When the input run is high, the input (Q2,Q1,Q0) is incrmented by one to produce (CF,S2,S1,S0), where CF indicates a carry-out.

Table 5: Excitation table for counter register (Counter3bit)

load	run	Description	CF	Q+[2..0]
0	0	Store	0	Q[2..0]
0	1	Increment	Q2·Q1·Q0	Q[2..0]+"001"
1	0	Load	0	P[2..0]
1	1	Load	Q2·Q1·Q0	P[2..0]

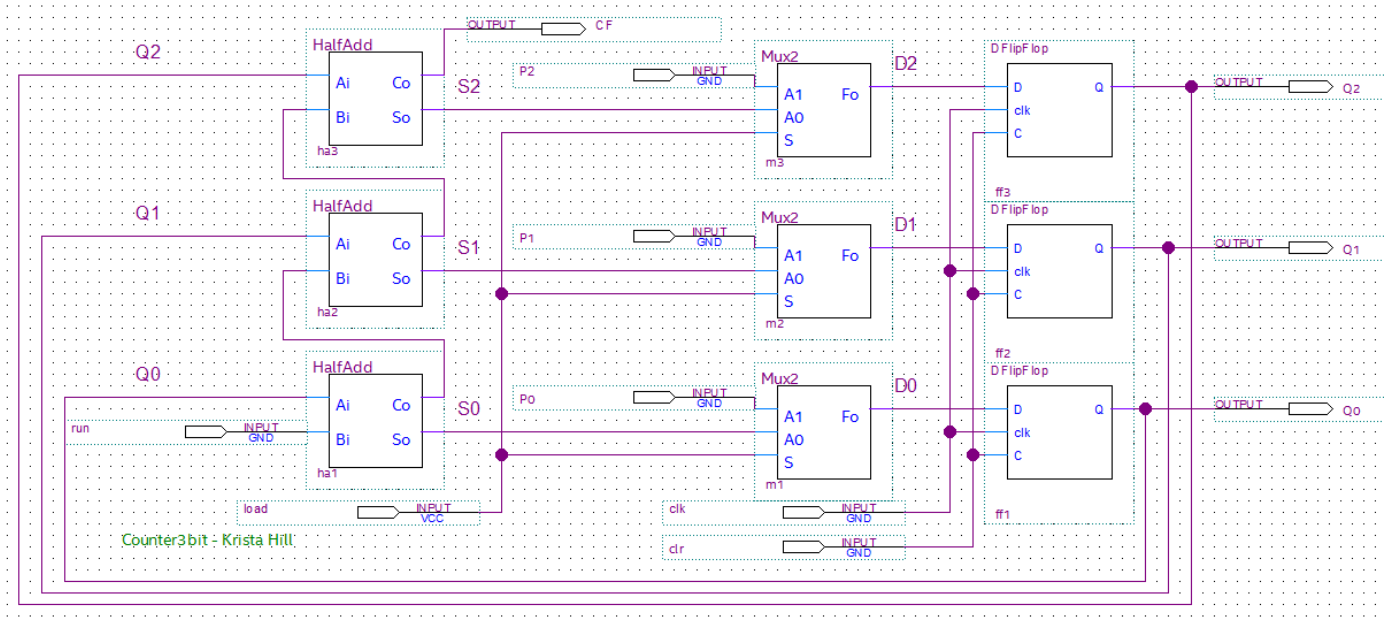


Figure 26: Counter register (Counter3bit)

The compressed file Counter3.zip contains VHDL files that describe the counter. You can use the parts in CounterParts.vhd to make symbols for a schematic of the counter, or use both files. A symbol generated for the counter is shown in Figure 27. The file Counter3bit.vhd is a text description of the schematic. The simulation produced by count3bit_tb.vhd is shown in Figure 28.

- CounterParts.vhd - contains HalfAdd, Mux2, DFlipFlop
- Counter3bit.vhd - uses the parts in CounterParts.vhd to implement the counter

- count3bit_tb.vhd - test bench file

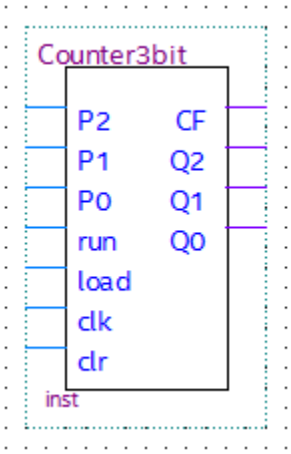


Figure 27: Symbol for Counter3bit

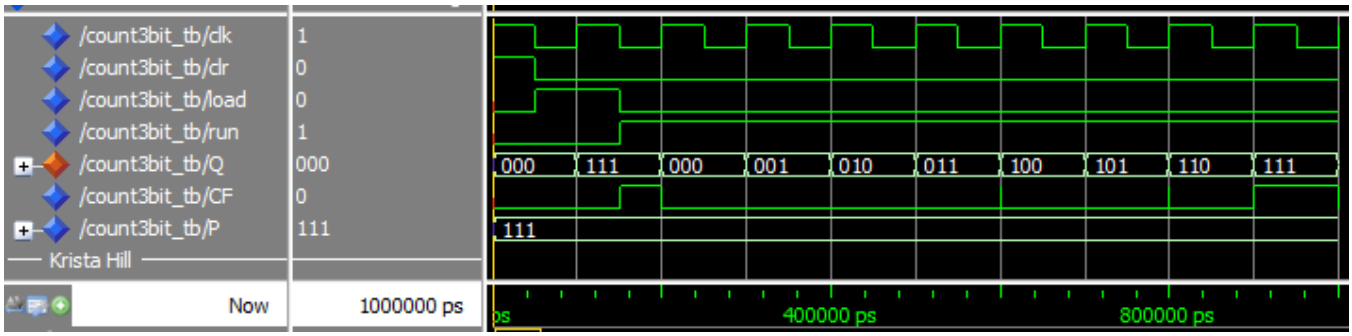


Figure 28: Simulation of Counter3bit

13 Preparing a Schematic to Run ModelSim Stand Alone with VHDL

ModelSim is a separate packate provided with Quartus. ModelSim can be started from inside Quartus or as a separate standalone program. In the following we consider the later. As outlined in section 7, for simpler work, the graphical waveform editor provides is more convenient than using ModelSim more directly. Section 11 outlines how to include the internal signal from the top level of your design. For more complicated work, you may find that using ModelSim directly is more convenient.

Here, we use ModelSim to simulate the counter circuit schematic shown in Figure 29. The Counter3bit symbol is outlined in section 12. Here a symbol was made for Counter3bit, using VHDL code from the compressed file Counter3.zip, as described in that section. The Counter3bit symbol is used in a circuit that counts starting at zero (000) and then overflows after four (100). As always, before attempting to simulate a circuit, make sure that it compiles as debugging later will be more difficult.

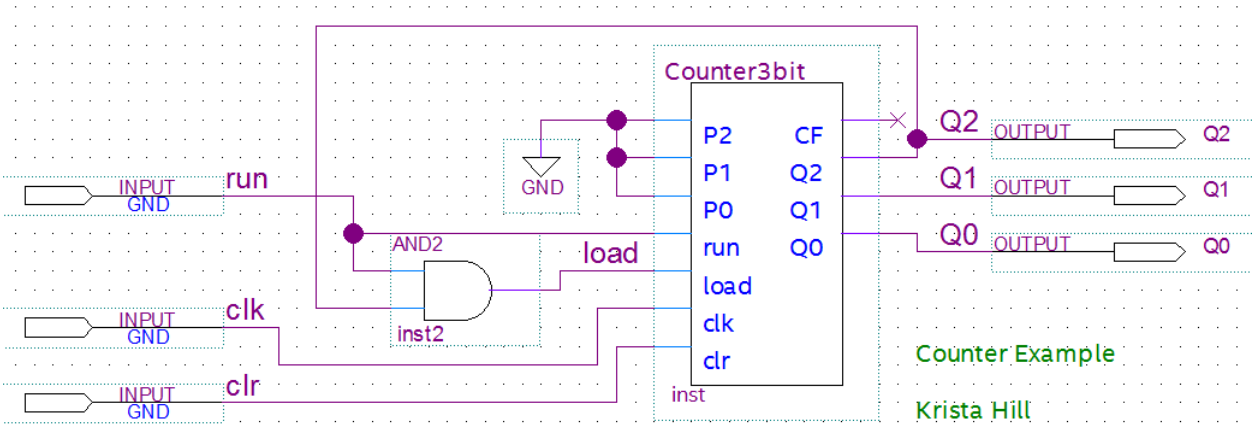


Figure 29: Example module based counter Counter3ex

The first step is to produce a VHDL file from your .bdf schematic. From the main window open the schematic for Counter3ex. Next, select the following. If you've detached the schematic editor pane, you can use the File menu here as well.

```
File => Create/Update => Create HDL Design File From Current File...
```

Next, in the pop-up window, for File type click to select VHDL, then click OK. After a few moments a message should appear that the VHDL file was successfully created. Section 14 describes how to run Modelsim stand alone along with a VHDL test bench file.

14 Running ModelSim Stand Alone with VHDL

Before starting ModelSim, use Quartus to make a test bench file. Even though you won't have your test bench file in your design hierarchy, the next time you build the project Quartus will perform error checking on the test bench for you, which is a huge benefit. Section 6 outlines how to make a VHDL file, the file used here is named Counter3ex_tb.vhd, given below. To see if your test bench file is in the project, in the Project Navigator pane change the view from 'Hierarchy' to 'Files'.

To start ModelSim, go back to the Windows start menu. Using ModelSim is a two step process, first make a new project, add and compile files, then perform a simulation. Close the welcome window.

```
Start => Programs => Intel FPGA => ModelSim - Intel => ModelSim
```

By default ModelSim will reopen the most recent project. To start new, if a project appears to be open, then close it and in the pop-up window click Yes to confirm your choice.

```
File => Close Project
```

Next, select the following and navigate to your project folder, but don't click OK yet, rather click 'Make New Folder'. type 'modelsim' click on the new folder, then click OK.

```
File => Change Directory
```

```
--------------------------------------------------------
-- Counter3bit_tb.vhd - YourName - TheDate
-- Testbench file for use with Counter3ex example
--------------------------------------------------------
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
ENTITY Counter3ex_tb IS
END Counter3ex_tb;
ARCHITECTURE behavioral OF Counter3ex_tb IS
   COMPONENT Counter3ex
   PORT( run,clk,clr  : IN  STD_LOGIC;
         Q2,Q1,Q0 : OUT STD_LOGIC);
   END COMPONENT;
   SIGNAL clk,clr  : STD_LOGIC := '1';
   SIGNAL run      : STD_LOGIC := '0';
   SIGNAL Q2,Q1,Q0 : STD_LOGIC;
BEGIN
   UUT: Counter3ex PORT MAP(
        clk => clk, clr => clr, run => run,
        Q0 => Q0, Q1 => Q1, Q2 => Q2);
   clk <= not clk after 50 ns;
   clr <= '1', '0' after 50 ns;
   tb : PROCESS
   BEGIN
      run <= '0'; wait for 50 ns;
      run <= '1'; wait for 600 ns;
      run <= '0'; wait for 100 ns;
      run <= '1'; wait for 100 ns;
      run <= '0'; WAIT; -- will wait forever
   END PROCESS;
END;
```

Make a new project

```
File => New => Project
```

In the pop-up window, for the Project Name field enter a name such as Sim01, check the following items and then click OK.

- The Project Location should be for the folder you just made
- The Default Library Name must be work and don't change any other settings.

At this point the Add items to the Project window will appear. Click 'Add Existing File'. In the pop-up window

click 'Browse' to navigate up one level to your project folder. Press and hold the control key, so when you click to select a file, a prior choice will remain selected. Select the following files, then click Open. Back in the 'Add file to Project' window click OK. In teh 'Add items to the Project click Close.

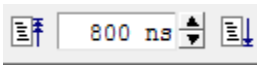
- Counter3bit.vhd
- Counter3ex.vhd
- Counter3ex_tb.vhd
- CounterParts.vhd

Given that you haven't compiled the files yet, then click Compile to open the Compile Order window. In that window click 'Auto Generate', then OK to close the notification window, then OK to close the Compile Order window. The next time you open this simulation project it won't be neccessary to set the order. Rather, use 'Compile All' or 'Compile Selected' instead.

```
Compile => Compile Order
```

To the lower left click on the 'Library' tab and then at the top click the plus-box to exapand the work library. Right click on 'counter3ex_tb' and select 'Simulate'. This is a moment of truth, as if there are any issues between files, this step will likely fail. If the simulation view appears, continue on. The next step is to select signals to insert into the simulation waveform view window.

- The sim pane shows the design hierarchy. Have topmost entry 'counter3ex_tb' selected/highlighted.
- The Objects pane shows signals visible in the given level of hierarchy. .Click to select the topmost signal, then press and hold the shift key while you click the last signal. All the signals should now be highlighted. Right-click and select 'Add Wave new'.
- After a moment a waveform display window will appear. To get a better view of the display you can undock the viewer by pressing the dock/undock button second to the upper right. Next, near the top of the pane change the simulation run time to 800 ns, then just to the right click the Run button which has a down arrow pointing at a line. The button just ot the left with an up arrow is used to restart the simulation.

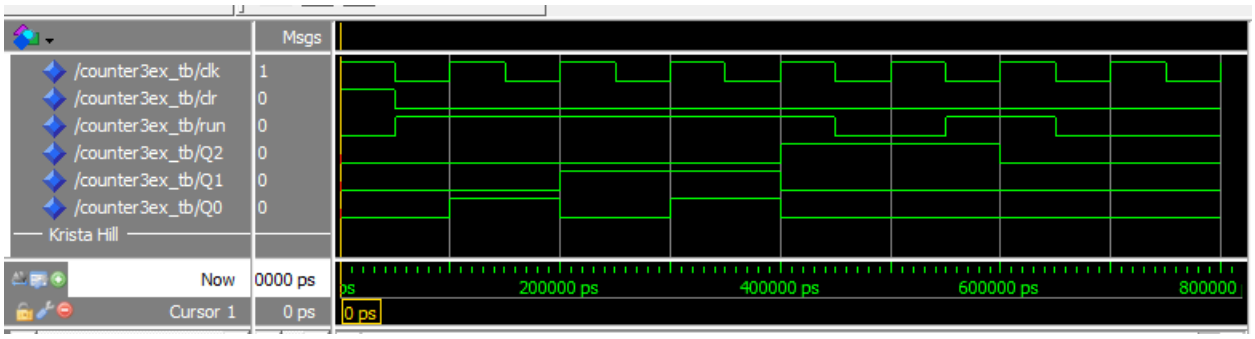


- Next, click the zoom-full button which is in the center of the zoom group. The icon looks like a blue magnifying glass.

- At this point you can move the signals up and down in the display. Insert a divider with your name, click the following, then in the pop-up window, in the Divider Name field, enter your name

```
Add => Divider...
```

- At this point the waveform viewer should look simiar to the following. To save and reuse the structure of waveform viewer configuration, under the File pull-down. you can use 'Save Format...' and then later use'Load => Macro File...'. At the end of the pathname use a filename like wave01.do.



If you make a change to a source file, end the simulation using Simulate => End, click on the Project tab (lower left), then recompile and start the simulation as you did before. Be sure to look about the various menus that we've visited to get a better sense of what ModelSim can do.

15 CPLD Module

Figure 30 shows a breadboard adapter module for an Altera/Intel device in an E64 type package. In this case a 5M40ZE64C5 device is mounted on the adapter module. The following are some key features of the adapter module.

- The CPLD device is in the center of the adapter module
- Just to the left of the CPLD is a regulator that produces 1.8Volt power for internal CPLD use
- A keyed ten pin connector, to the right is used to configure the adapter module. We use a USB-Blaster device to configure the CPLD.
- Breadboard pin, which are pointing downward, are used to attach the adapter module to a breadboard. Pins1 through 28 are along the far edge and pins 29 through 56 are along the near edge. Pins 1, 28, 29, and 56 are each identified by its corresponding pin number.
- Pins 42 and 43, which are in the middle of the near edge, both connect directly to a larger chip capacitor on the board. Pins 42 and 43 convey ground (GND) as well as 3.3Volt power (VCC), respectively.

Table 6 summarizes all the pins used by the CPLD adapter module. Each pin has a module P pin number, a CPLD device pin number, the corresponding device bank number, and a summary of some alternate uses, though the use of pins for differential signaling is not considered here. The following are some alternate uses.

- The four CLKn pins have special capability, each for conveying a clock signal

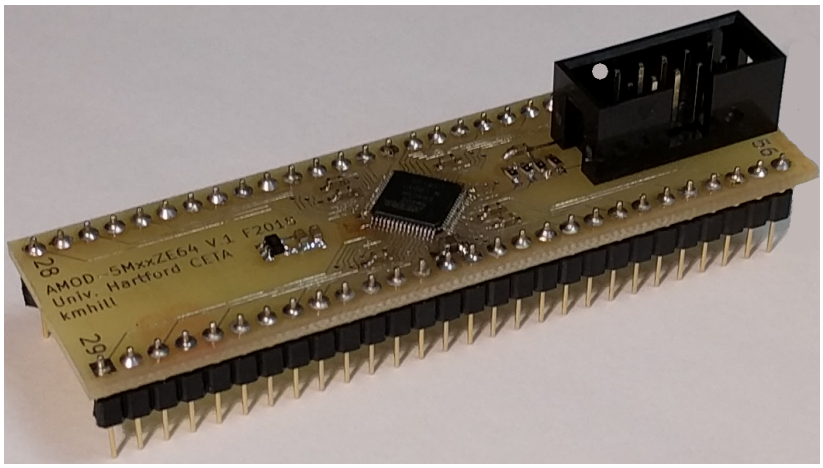


Figure 30: CPLD module for MAX V device

- The OE pin can be used to enable CPLD pins as outputs
- The CLRn pin is for conveying a device clear signal

Given the potential importance of the alternate uses of these pins, if you can, avoid using these pins to covey ordinary signals. If it turns out that you need the alternate pin behavior, it will be available to you.

Table 6: CPLD Pin Assignment

Mod.	Dev.	Bank	AltUse	Mod.	Dev.	Bank	AltUse
P1	19	1		P56	18	1	
P2	20	1		P55	13	1	
P3	21	1		P54	12	1	
P4	22	1		P53	11	1	
P5	24	1		P52	10	1	
P6	25	1		P51	9	1	CLK1
P7	26	1		P50	7	1	CLK0
P8	27	1		P49	5	1	
P9	28	1	OE	P48	4	1	
P10	29	1	CLRn	P47	3	1	
P11	30	1		P46	2	1	
P12	31	1		P45	1	1	
P13	32	1		P44	64	1	
P14	33	1		P43	VCC	–	
P15	34	2		P42	GND	–	
P16	35	2		P41	63	2	
P17	36	2		P40	62	2	
P18	37	2		P39	61	2	
P19	38	2		P38	60	2	
P20	40	2	CLK2	P37	59	2	
P21	42	2	CLK3	P36	58	2	
P22	43	2		P35	56	2	
P23	44	2		P34	55	2	
P24	45	2		P33	54	2	
P25	46	2		P32	53	2	
P26	47	2		P31	52	2	
P27	48	2		P30	51	2	
P28	49	2		P29	50	2	

16 Configure a CPLD With Basic Gates

Section not written yet