# CSE 352 Tutorial #4

## Synthesizing onto an FPGA

### Objectives

This tutorial will walk you through the steps of implementing a design made in Active-HDL onto the Altera Cyclone II FPGA

**NOTE:** This tutorial contains many large illustrations. Page breaks have been added to keep images on the same page as the step that they represent.

#### Before you start:

Important: You need to make sure that the USB-Blaster Drivers are up to date. This can be done in less than 30 seconds by following <u>those steps</u>.

#### Steps

Open up Active-HDL in administrator mode if you're working on your home machine. Otherwise, open up Active-HDL normally. Create a new workspace.

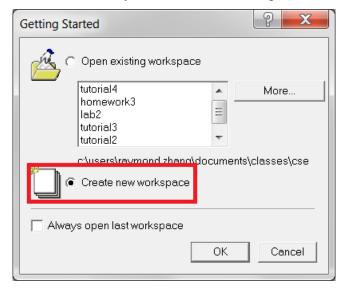


Figure 1 – Create a new workspace.

When selecting the location for your new workspace, <u>make sure that the path does not contain</u> <u>spaces</u>. Name your workspace "tutorial4".

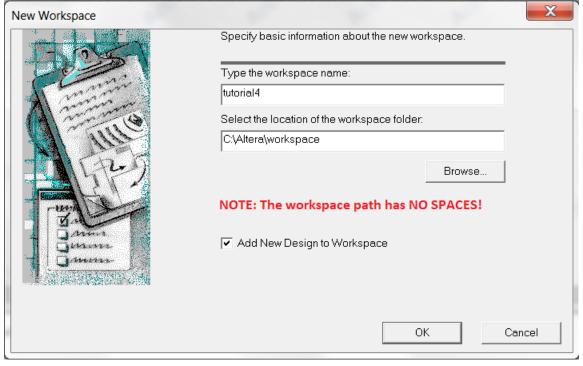


Figure 2 – Name the workspace.

In the new design wizard, select "Create an Empty Design with Design Flow". Click Next.

New [	Design Wizard	J
	How would you like to create Design Resources?	
	C Create an Empty Design	
	Create an Empty Design with Design Flow	
	C Add existing Resource Files	
	C Import a Design from Active-CAD	
	This option creates an empty design and enables Design Flow Manager. You can select a vendor of your synthesis or implementation tool, technology, libraries, and specify the default HDL language of your new design entry sources.	
	C Create New Workspace	
	Add Design to Current Workspace	
	< Back Next > Cancel	

Figure 3 - Create an empty design with design flow.

Next, you should see a window as depicted in the screenshot. Make sure that the [Block Diagram Configuration] is set to use (Default HDL Language), and the [Default HDL Language] is set to (VERILOG). Click on Flow Settings.

lew Design	Wizard
Specify	additional information about the new design.
	C-Synthesis tool: <none></none>
	Synthesis tool: Altera QuartusII Synthesis Implementation 12.0
	Physical Synthesis tool: <none></none>
	Implementation tool: <none></none>
	Default Family: Quartus Cyclonell Flow Settings
	Block Diagram Configuration: Default HDL Language  Default HDL Language: VERILOG
	< Back Next > Cancel

Figure 4 - Set the HDL language, then click on Flow Settings

Under the [HDL Synthesis:] heading you should set the Tool to be (Altera Quartus II Synthesis & Implementation 12.0), or (Altera Quartus II Synthesis & Implementation 12.1) if you are working on your own computer. Also you need to set the [Family:] to be (Quartus Cyclone II). When all of this is done, click [OK] to close the Flow Settings window, and then click [Next >] on the New Design Wizard to move on.

Flow Configuration Set	tings	X
C Synthesis:	<none></none>	Select
HDL Synthesis:		
Tool name:	Altera QuartusII Synthesis & Implementation 12.0	Select
- ⊇-f	Use Server Farm	
Physical Synthesis:		
Tool name:	<none></none>	Select
Implementation:		
Tool name:	<none></none>	Select
	Use Server Farm	
Simulation:		
	🔲 Use Server Farm	
Defaults: Family:	Quartus Cyclonell	1
	OK Cancel	Help

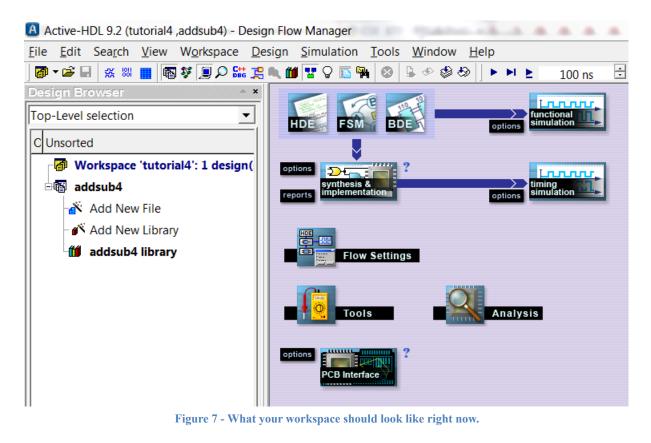
Figure 5 - Set the HDL Synthesis tool name and the Family.

On this screen, choose an appropriate name for your design (e.g. addsub4). Remember that you should not begin your names with numbers, or other odd characters. Also your names cannot contain spaces to avoid problems with software in the future. Click through the rest of the dialogue boxes until you see that your design has been added to your workspace.

lew Design Wizard			×
Specify basic information about the new	design.		
Type the design name:			
addsub4			
Select the location of the design folder:			
C:\Altera\workspace\tutorial4			
			Browse
The name of the default working library of	of the design:		
addsub4			
The name specified here will be used a library files and as the logical name of th the logical name later on.			
	< Back	Next >	Cancel

Figure 6 - Set the name of the new design.

Your workspace should look like figure 7. The various icons and buttons on the right-side of the window is the Design Flow. Active HDL uses it to guide you through the process of synthesizing and implementing your design onto the FPGA.



Now that we have created our design we are ready to create the schematic for the Four Bit Adder/Subtractor that you will synthesize onto the FPGA. This is the same Four-Bit Adder/Subtractor that you made in Tutorials 1 & 2, starting with the Full Adder and building on that. You need to recreate those designs using the yellow "Built-In" gates so that you will be able to synthesize the design on the hardware. Once you have the Four Bit Adder/Subtractor schematic re-created, continue below.

NOTE: revisit <u>Tutorial 1</u> and/or <u>Tutorial 2</u> if you need help designing the circuits.

Check your Full Adder and Four Bit Adder/Subtractor circuits; do they look like the ones to the right? It is important that you are using the yellow symbols and not the pink ones because the pink ones will not synthesize correctly onto the FPGA.

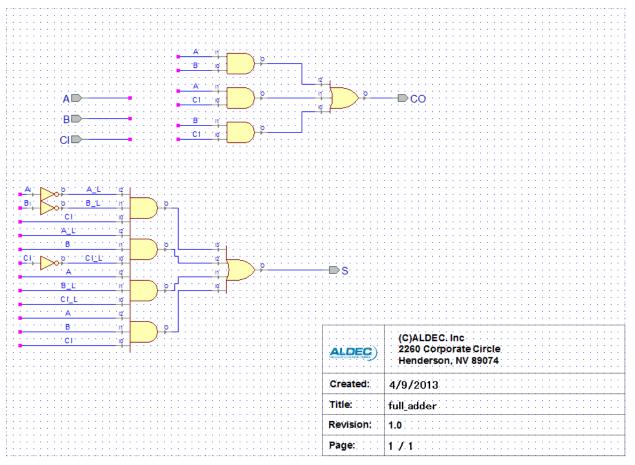


Figure 8 - The full adder circuit. Note the usage of the yellow (built-in) symbols.

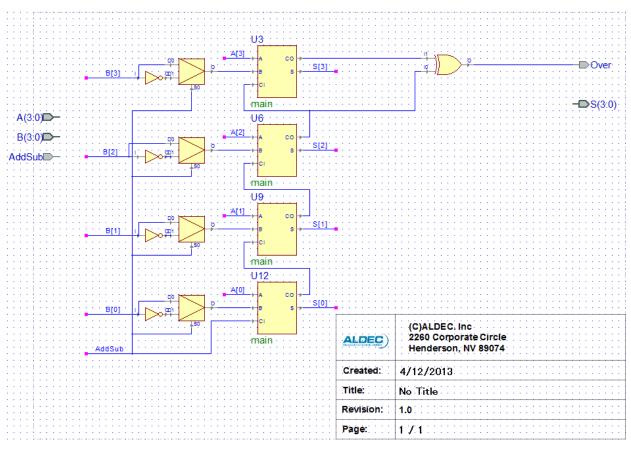


Figure 9 - The four-bit addition/subtraction circuit. Note the usage of the yellow (built-in) symbols.

There is one last piece that we need to create before synthesizing onto the FPGA. Since the FPGA doesn't know what the various Inputs and Outputs in your design are connected to, we have to create a (.qsf) file for the FPGA to read and use. If we don't create one the FPGA will randomly assign pins to be the outputs and inputs and that could be very bad considering the FPGA has hundreds of pins. You can think of the (.qsf) like a setting file which you use to inform the FPGA what pins you want connected to what.

For this lab we want to use the switches, buttons and LEDs for our inputs and outputs. The reason why we need to assign switches, buttons, and LEDs during the synthesis is because the old connections you were using is actually a program that was preloaded into the FPGA redirecting the outputs in and inputs of the switches, LEDs and buttons from their standard inputs and outputs to the connectors on the board. Of course now that everything is on the FPGA, you can't actually wire a connection from those pinouts into the FPGA, so we have to program in our connections. Luckily for us, the FPGA has pins which are directly connected to the switches, buttons and LEDs. Creating a (.qsf) file is pretty easy; the FPGA follows a very straightforward format. For example, the pin connected to Switch 0 is called [PIN\_L22], so to assign one of our inputs to PIN\_L22 we use the following:

set\_location\_assignment PIN\_L22 -to A[0]

By using that line of text we have assigned Switch 0 to be the first input of A. If you look at our design above, you will see that there is A[0] through A[3], so each of those will need a pin assigned to it. It's easy because we can directly reference the names we used in our design. Go ahead and create the (.qsf) file with notepad and save it in the [src] folder of your design. Remember to use good naming conventions for your file by avoiding numbers and spaces. You can refer to this file <u>pinouts.html</u> to see what pins are connected to what inputs/outputs on the FPGA. Make sure you don't miss any of your inputs and outputs; you should have assignments for:

INPUTS: A[0], A[1], A[2], A[3], B[0], B[1], B[2], B[3], AddSub OUTPUTS: Over, S[0], S[1], S[2], S[3]

You can use this (.qsf) file as a <u>template</u>. It is important however that you understand the assignments.

Now that we have created the final piece needed to synthesize onto the FPGA we can actually begin the actual process of putting the design onto the FPGA. Go back to the Design Flow that we saw earlier. There is a quick button at the top of the screen that you can click that will bring you to the Design Flow as well.



Figure 10 - The highlighted button will show the Design Flow, if it's been closed.

From here you should click on the [options] button next to [synthesis & implementation] to gain access to the window where we will set up our configuration for the FPGA.

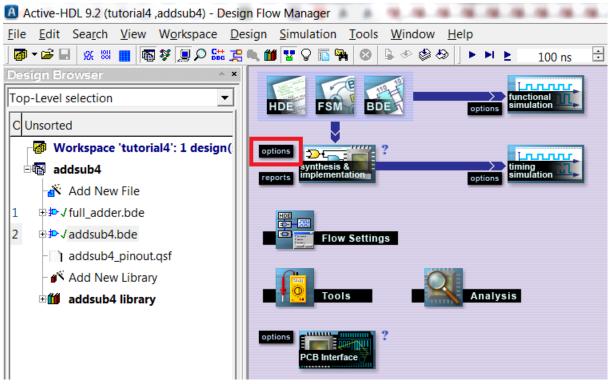


Figure 11 - The location of the options button.

Once you click on the options button a (Synthesis Options) window will come up. The first thing to check is to make sure all the important files are selected in the (Design Files) browser on the left side of the window. For this design we need to make sure the Four Bit Adder/Subtractor will be included as well as the Full Adder design. You know it's included when there is a little red arrow next to the file.

Synthesis and Implementation Options		X
Design Files: ⊡ <b>® addsub4.adf</b> 1 ⊕≇≎ <b>←</b> full_adder.bde	$/{\rm Main} \\ \\ \sqrt{\rm Compilation} \\ \sqrt{\rm Synthesis} \\ \sqrt{\rm Fitter} \\ \sqrt{\rm Tim. \ Analyzer} \\ \sqrt{\rm Device} \\ \sqrt{\rm Programming} \\ \sqrt{\rm Pins} \\ \sqrt{\rm Local Compilation} \\ \sqrt{\rm Synthesis} \\ \sqrt{\rm Synthesis} \\ \sqrt{\rm Synthesis} \\ \sqrt{\rm Tim. \ Analyzer} \\ \sqrt{\rm Device} \\ \sqrt{\rm Programming} \\ \sqrt{\rm Pins} \\ \sqrt{\rm Local Compilation} \\ \sqrt{\rm Synthesis} \\ \rm Synth$	og ()S
2	Top-level Unit	•
	Simulation output format:	Verilog 🗾
	Time scale:	1 ps 🔻
	Run Mode: GBatch C GUI	F Auto close
		Generate Impl. Script
	Device Assignment	
	Family Quartus Stratix	AUTO
	Speed Grade ANY	
	HardCopy: <pre></pre>	P & RAM to HardCopy device resources.
	✓ Create HardCopy Companion Revision	
	Quartus 12.0 Project Options	
	☐ Import pin assignments from Quartus Settings File:	Browse
	File name:	
	Generate FPGA Exchange file:	Browse
	Directory name: C:/Altera/workspace/tutorial4/addsub4/symbols/fpgaxchange	
	Voverwrite existing project	
		Use Default Settings
	Quartus Settings File	
▲		
Update synthesis order	ОК	Cancel Help

Figure 12 - Include the block diagrams for the full adder and the four-bit adder/subtractor.

Next we need to select the (Top-level Unit), this is the primary design that will be put on the FPGA. Since we are trying to put the Four Bit Adder/Subtractor on the FPGA we will choose that as our (Top-level Unit). If we were only synthesizing the Full Adder, we could choose the Full Adder's design instead from the drop down menu.

Synthesis and Implementation Options	×
Design Files: ■	/Main     Compilation     /Synthesis     /Fitter     /Tim. Analyzer     /Programming     /Pins     /Log     Image: Synthesis       Main     Top-level Unit     addsub4     Image: Synthesis     Image: Synthesis
	Simulation output format:
	Time scale:
	Run Mode: ⓒ Batch ⓒ GUI
	Generate Impl. Script
	Device Assignment
	Family Quartus Stratix   Device AUTO
	Speed Grade ANY
	HardCopy: <pre></pre> Imit DSP & RAM to HardCopy device resources.
	Create HardCopy Companion Revision
	Quartus 12.0 Project Options
	Import pin assignments from Quartus Settings File:     Browse
	File name:
	Generate FPGA Exchange file:     Browse  Directory name: C:/Altera/workspace/tutorial4/addsub4/symbols/fpgaxchange
	✓ Overwrite existing project
	Use Default Settings
	Quartus Settings File
< III >	
Update synthesis order	OK Cancel Help

Figure 13 - Set the four-bit adder/subtracter as the top-level unit.

The (Simulation output format) should be None and the (Time Scale) should be 1 ps. Make sure the (Run Mode) is set to Batch.

Synthesis and Implementation Options		X
Design Files: ■	/Main\/Compilation\/Synthesis\/Fitter\/Tim. Analyzer\/Device\/Programming\/Pins\/ Main	
2 ⊕ ‡⊅ ← addsub4.bde	Top-level Unit addsub4	<u> </u>
	Simulation output format:	<none></none>
	Time scale:	1 ps 💌
	Run Mode: 🕫 Batch C GUI	☐ Auto close
•		Generate Impl. Script
	Device Assignment	
	Family Quartus Stratix   Device	e AUTO 💌
1	Speed Grade ANY	
	HardCopy: Sones	SP & RAM to HardCopy device resources.
	✓ Create HardCopy Companion Revision	
	Quartus 12.0 Project Options	
	☐ Import pin assignments from Quartus Settings File:	Browse
	File name:	
	Generate FPGA Exchange file:	Browse
	Directory name: C:/Altera/workspace/tutorial4/addsub4/symbols/fpgaxchange	
	✓ Overwrite existing project	
1		Use Default Settings
1	Quartus Settings File	
4 III +		
Update synthesis order	OK	Cancel Help

Figure 14 - Set the simulation output format and the run mode.

Next under Device Assignment we need to tell the program what kind of FPGA we have. The (Family) should be Quartus CycloneII, and the (Device) should be EP2C20F484C, this is the same serial that is written directly on the FPGA chip.

When you change the (Device) the (Speed Grade) should automatically be updated to 6.

Synthesis and Implementation Options	×
Design Files: ⊡ <b>ആ addsub4.adf</b> 1 ⊕⊉⊷¢full_adder.bde	/Main√Compilation√Synthesis√Fitter√Tim. Analyzer√Device√Programming√Pins√Log         Main
2 ⊕ ‡⊅ ← addsub4.bde	Top-level Unit addsub4
	Simulation output format:
	Time scale:
	Run Mode: C Batch C GUI
	Generate Impl. Script
	Device Assignment
	Family Quartus CycloneII   Device EP2C20F484C
	Speed Grade 6
	HardCopy: <none> 💌 🔽 Limit DSP &amp; RAM to HardCopy device resources.</none>
	✓ Create HardCopy Companion Revision
	Quartus 12.0 Project Options
	☐ Import pin assignments from Quartus Settings File: Browse
	File name:
	Generate FPGA Exchange file:     Browse Directory name: C:/Altera/workspace/tutorial4/addsub4/symbols/fpgaxchange
	Orrectory name: IC:/Attend/workspace/tutonam/add/subm/symbols/rpgaxchange     Overwrite existing project
	Use Default Settings
	Quartus Settings File
III ►	
Update synthesis order	OK Cancel Help

Figure 15 - Set the device family, the device itself, and the speed grade.

Finally the last setting that you need to add in is to set up the pin assignment. Check the little box that says (Input pin assignment from Quartus Settings File) and Browse to the (.qsf) file that you made and select it. This way the program will use the pinouts that you assigned in the (.qsf) file.

Synthesis and Implementation Options		×
Design Files: ■	$/{\rm Main} \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\$	Log
2	Top-level Unit addsub4	•
	Simulation output format:	<none></none>
1	Time scale:	1 ps 💌
	Run Mode: GBatch C GUI	T Auto close
•		Generate Impl. Script
	Device Assignment	
	Family Quartus CycloneII   Devic	e EP2C20F484C
•	Speed Grade 6	
	HardCopy: <none></none>	SP & RAM to HardCopy device resources.
	Create HardCopy Companion Revision	
	Quartus 12.0 Project Options	
	Import pin assignments from Quartus Settings File:	Browse
	File name: C:/Altera/workspace/tutorial4/addsub4/src/addsub4_pinout.qsf	
	Generate FPGA Exchange file:	Browse
	Directory name: C:/Altera/workspace/tutorial4/addsub4/symbols/fpgaxchange	
	Overwrite existing project	
		Use Default Settings
	Quartus Settings File	
۰ III •		
Update synthesis order	OK	Cancel Help

Figure 16 - Set the Quartus Settings File to the file containing your pin assignments.

Now you are all set. Check to make sure the settings are all right and click [OK].

Synthesis and Implementation Options		X
Design Files: ■ <b>® addsub4.adf</b> 1 ⊕ <b>₽ ←</b> full_adder.bde	$\label{eq:main} $$ $$ Main $$ Original Compilation $$ Synthesis $$ Fitter $$ Tim. Analyzer $$ Device $$ Programming $$ Pins $$ Main $$ Main $$ $$ $$ $$ $$ $$ $$ $$ $$ $$ $$ $$ $$$	
2 ⊕ #⊇ ← addsub4.bde	Top-level Unit addsub4	•
	Simulation output format:	<none></none>
	Time scale:	1 ps 💌
	Run Mode: G Batch C GUI	T Auto close
		Generate Impl. Script
	Device Assignment	
	Family Quartus CycloneII	ce EP2C20F484C
	Speed Grade 6	
	HardCopy: <none></none>	DSP & RAM to HardCopy device resources.
	Create HardCopy Companion Revision	
	Quartus 12.0 Project Options	
	✓ Import pin assignments from Quartus Settings File:	Browse
	File name: C:/Altera/workspace/tutorial4/addsub4/src/addsub4_pinout.qsf	
	Generate FPGA Exchange file:	Browse
	Directory name: C:/Altera/workspace/tutorial4/addsub4/symbols/fpgaxchange	
	Verwrite existing project	
		Use Default Settings
	Quartus Settings File	
۰ III ا		
Update synthesis order	OK	Cancel Help

Figure 17 - What your options settings should look like.

Now that everything is all set. Click on the [Synthesis & Implementation] button and let the program do its work. If everything worked well, the Quartus 12.0 window will pop open and do all of the necessary work to create a file that you can program onto the FPGA.

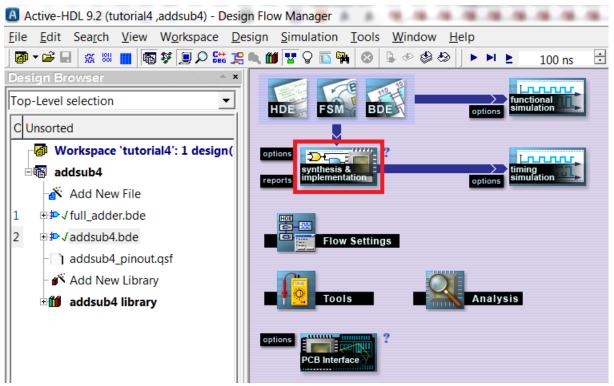


Figure 18 - The location of the Synthesis & Implementation button.

Quartus 12.0 should complete successfully with no errors, and only minimal warnings. You will likely see at least two warnings regarding "Classic Timing Analyzer, it is safe to ignore those. You should make sure to check any other warnings you receive, and make sure you understand why they occur. Ask your T.A. for help if you have any questions regarding specific warnings or errors.

**Note:** There used to be an odd problem with Active-HDL, where it would throw an error even when there are no errors. If you see red text in the Active-HDL console stating that Quartus 12.0 finished with errors, but Quartus itself says it did not have any errors, you can safely ignore the Active-HDL message.

```
% Executing Quartus 12.0
Info (293000): Quartus II Full Compilation was successful. 0 e
rrors, 9 warnings
Info (23030): Evaluation of Tcl script c:/altera/12.1/quartus/
common/tcl/internal/qsh_flow.tcl was successful
Info: Quartus II 64-Bit Shell was successful. 0 errors, 9 warn
ings
Info: Peak virtual memory: 200 megabytes
Info: Processing ended: Fri May 03 15:43:17 2013
Info: Elapsed time: 00:00:14
Info: Total CPU time (on all processors): 00:00:01
Quartus Processes have finished.
Finished with 9 warning(s).
Close
```

Figure 19 - The output of Quartus. Note that there are a few warnings but no errors.

Great, we now have a file that we can program onto the FPGA board.

Check to make sure the USB cord is fastened securely and plugged into the board. Also, check to make sure the little switch below the power button is on the RUN side and not the PROG side. Finally, power on your FPGA board, make sure it runs and is flashing the usual CSE 352 across the HEX display.



Figure 20 - Make sure this switch is set to RUN.

Click on the [Analysis] button on the Design Flow, and then click on the [Programmer] button.

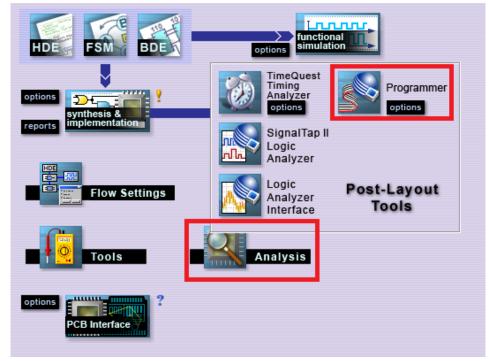


Figure 21 - The location of the Programmer button. Click Analysis to bring up the menu, then click Programmer.

The Quartus II Programmer will pop up. This is the interface that connects your computer to the FPGA via the USB. Check to make sure that the Hardware is set up correctly, you should see that the connection is via (USB- Blaster).

ų	🖗 Quartus II 64-Bit Programmer - [Chain1.cdf]								
F	File Edit View	Processing Tools Wi	ndow Help 🐬				Sear	ch altera.c	com 🔇
	🔔 Hardware Set	up USB-Blaster [USB-	0] Mode:	JTAG		<ul> <li>Progre</li> </ul>	ess:		
	Enable real-tim	ne ISP to allow background	d programming (fo	r MAX II and MA	X V devices)				
	Start	File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank- Check	Examine
	🔎 Stop								
	Auto Detect								
	💢 Delete								
	붤 Add File	•		111					•
	🕍 Change File								
	🚰 Save File								
	<sub>)</sub> Add Device								
	ி <sup>ற</sup> ் Up								
	Ju Down								
		🚺 🔬 🔻 < <se< td=""><td>arch&gt;&gt;</td><td></td><td>~</td><td></td><td></td><td></td><td></td></se<>	arch>>		~				
ļ	Type II	) Message							
	ides								
	System / P	rocessing /							Þ
	∑\_System_/\_P								h.

#### Figure 22 - Check that the hardware setup says USB-Blaster.

If you don't see this, click on the [Hardware Setup] button and select the connection to be via the USB-Blaster.

**NOTE:** If you don't see the USB-Blaster option, you may need to click the [Add Hardware...] button and add the USB-Blaster setup. Make sure that USB cord plugged into the board is plugged into your computer at the other end.

	y to the current prog e: USB-Blaster [US	rammer window	g devices. This programming
Hardware USB-Blaster	Server Local	Port USB-0	Add Hardware Remove Hardware
			Close

Figure 23 - Set the Currently Selected Hardware to USB-Blaster.

Make sure that the (Mode) is set to (JTAG).

We Quartus II 64-Bit Programmer - [Chain1.cdf]										
	File Edit View	Processing Tools Wir	ndow Help 🐬				Searc	ch altera.c	com 🍕	
	🔔 Hardware Setu	up USB-Blaster [USB-	0] Mode:	JTAG		- Progre	ess:			
Enable real-time ISP to allow background programming (for MAX II and MAX V devices)										
	Start	File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank- Check	Examine	
	🛯 🖉 Stop									
	Auto Detect									
	🔀 Delete									
	Add File	•		111					4	
	🔓 Change File									
	Save File									
	🖄 Add Device									
	¶ <sup>™</sup> Up									
	Down									
	× All 🙆 🛆	🔟 🔬 쿠 < <se< td=""><td>arch&gt;&gt;</td><td></td><td>~</td><td></td><td></td><td></td><td></td></se<>	arch>>		~					
	Type ID Message									
	des									
	System / Pr	ocessing /							Þ	

Figure 24 - Set the Mode to JTAG.

Click on the [Add File] button and navigate to your design folder. Under your design folder you should see a folder called: synthesis.

V Quartus II 64-Bit Programmer - [Chain1.cdf]											
	File Edit View	Processing Tools Wi	ndow Help 💎				Searc	ch altera.c	com 🔇		
	Hardware Setu	USB-Blaster [USB- e ISP to allow background		JTAG r MAX II and MA		Progre	ess:				
	Start	File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank- Check	Examine		
	Here Stop										
	X Delete	•							4		
	Change File								F		
	Add Device										
	J <sup>™</sup> Down										
	All     Image: All image: All										
	System / Pr								4		
	∑ System / Pr	ocessing /									

Figure 25 - Click the Add File... button, and navigate to the synthesis folder.

Select the (.sof) file, this is the programming file that you will load onto the FPGA. Click [Open].

**Note:** You have another option which isn't shown in the picture. When you click on the [Programmer] button under [Analysis] you can click on the little {Options} tag next to the [Programmer] button to select the programming file ahead of time.

**Note 2:** Make sure to select the (.sof) file and not the (.pof) file. The latter is used when programming the flash memory on the DE-1.

Select Programming File										
Look in: C:\Altera\workspace\tutorial4\addsub4\synthesis	) 📑 📰 🔳									
My Computer Raymond Zhang addsub4.sof										
File name: addsub4.sof	Open									
Files of type:       Programming Files (*.sof *.pof *.jam *.jbc *.ekp *.jic)	Cancel									

Figure 26 - Make sure you select the .sof file!

Now all that is left is to program the file onto your FPGA. Check the little box under [Program/ Configure] and click the [Start] button to program your design onto the FPGA.

**Note:** You should see the chip ID listed under the chip graphic. This is the same ID as you selected in the Synthesis Options steps above.

W Quartus II 64-Bit Programmer - [Chain1.cdf]*												
File         Edit         View         Processing         Tools         Window         Help         Search altera.com         Searc												
🔔 Hardware Set	Lardware Setup USB-Blaster [USB-0] Mode: JTAG   Progress:											
Enable real-time ISP to allow background programming (for MAX II and MAX V devices)												
Start	File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank- Check	Examine				
🕮 Stop	C:/Altera/workspace	EP2C20F484	001B1C4B	FFFFFFF	✓							
Auto Detect												
🗶 Delete												
Add File	•		111					4				
🕍 Change File	Image: Save File       Ima											
Save File												
Add Device												
1 <sup>11</sup> Up												
J <sup>™</sup> Down	EP2C20F48	34						-				
П. П												
Ges												
System / P	rocessing /							Þ				
	Jucessing /											

Figure 27 - Click the Start button to program the FPGA.

If the programming went as expected, you should see the green [Progress] bar at 100% at the upper right and the green "success" message in the bottom portion of the Quartus II Programmer window. If not, verify your settings and try again. If you continue to have issues, consult your T.A.

-	Bit Programmer - [Chai Processing Tools Wir		)			Sear	ch altera.	com 🤇
🔔 Hardware Set	up USB-Blaster [USB- ne ISP to allow background	-	JTAG for MAX II and M		▼ Progre	ess:	100% (Su	ccessful)
🏴 Start	File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank- Check	Examine
Stop Auto Detect Comparison Delete	C:/Altera/workspace	EP2C20F484	001B1C4B	FFFFFFF				
Add File	•		III					
Add Device								
United States and Stat								
All 😢 🛆		arch>>		~				
Type II 2090 2090	Message 07 Configuratio 011 Successfully 061 Ended Progra	performed	l operatio	n(s)	_		13	
System (6)	Processing /							P

Figure 28 - Success!

If your programming was successful, your DE-1 board should now be running your program with the Switches, Keys, and/or LEDs that you selected in the (.qsf) file.

Try it out!