Lab 1

Introduction to Quartus Schematic
Binary Adders Using Simulation & the DE1-SoC Board

1. Introduction

The DE1-SoC is an embedded computing device built around the Altera System-on-Chip (SoC) FPGA and an ARM processor running a Linux operating system. You will be using this platform throughout this course to get hands-on experience working with designing digital systems. The platform is a very rich environment with abundant hardware features, though you will primarily focus on the Cyclone V FPGA. The DE1-SoC also contains a wide set of input/output (I/O) devices, among which you will use LED lights, switches, buttons, and 7-segment displays.

In this lab you will design a circuit capable of adding two binary numbers, and you will use the Quartus Prime Schematic to simulate and validate its behavior. This tool allows you to construct complex circuits with a hierarchical schematic design, which you can then test with different artificial inputs. You will also upload your designs into the FPGA on the DE1-SoC Board and verify the designs using Switches and LEDs on the board. You are **not** allowed to use any of the Quartus software library modules unless for the modules that are explicitly allowed in the lab instructions.

Check the Lab Resources on Canvas for the document "***Installing the Quartus Prime Lite Software***" in case you want to practice with the software on your Windows or Linux computer. Installing the software on your own computer is not required though, as the software is installed on our lab computers.

1.1 Prelab Assignment

A half adder is a circuit capable of adding two 1-bit numbers. The circuit has two inputs and two outputs. The circuit can add two 1-bit numbers (inputs), for example, $1_2+1_2 = 10_2$. Here 0 in the result represents the Sum (***S***) and 1 represents the Carry out (***C***$_{out}$).

A	*B*	*S*	C_{out}
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Complete the shown truth table for a half adder circuit where ***AB*** represents the 2-bit input.

Have the requirements of the prelab assignment ready to be checked at the beginning of the lab session.

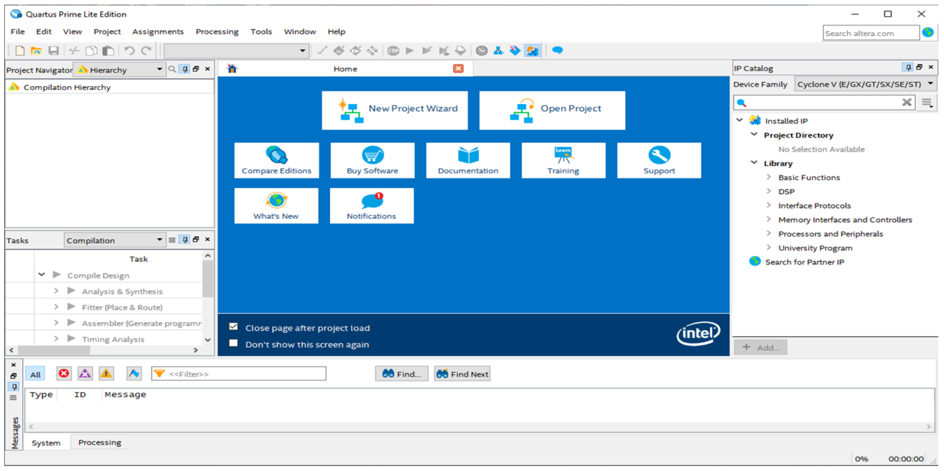
2. The Half Adder Simulation

1. Use your Northeastern credentials to log in to the lab desktop computer.
2. Get a DE1-SoC board from the cabinets in the lab and power it with the given power adapter. Connect the board to your computer using the given USB-A/USB-B cable. The USB-A goes to the lab computer and the USB-B goes to the DE1-SoC board.
3. Building your first project:

a) Create a **half_adder** folder on your computer.

b) Run the Quartus Prime Lite software.

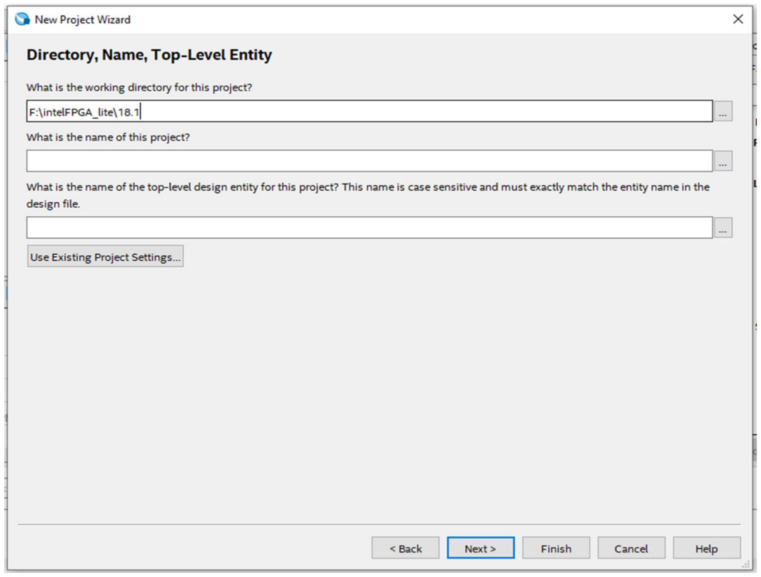
Note: to open an existing project, make sure to select File → **Open Project** (not just Open)



c) Select **File →New Project Wizard** and click **Next**.

d) Set the Working Directory: browse to the **half_adder** folder you created in step **b** above.

- Name of this project: **half_adder**
- Top-level entity: **half_adder**
- Then click **Next**.



New Project Wizard

Directory, Name, Top-Level Entity

What is the working directory for this project?

F:\intelFPGA_lite\18.1

What is the name of this project?

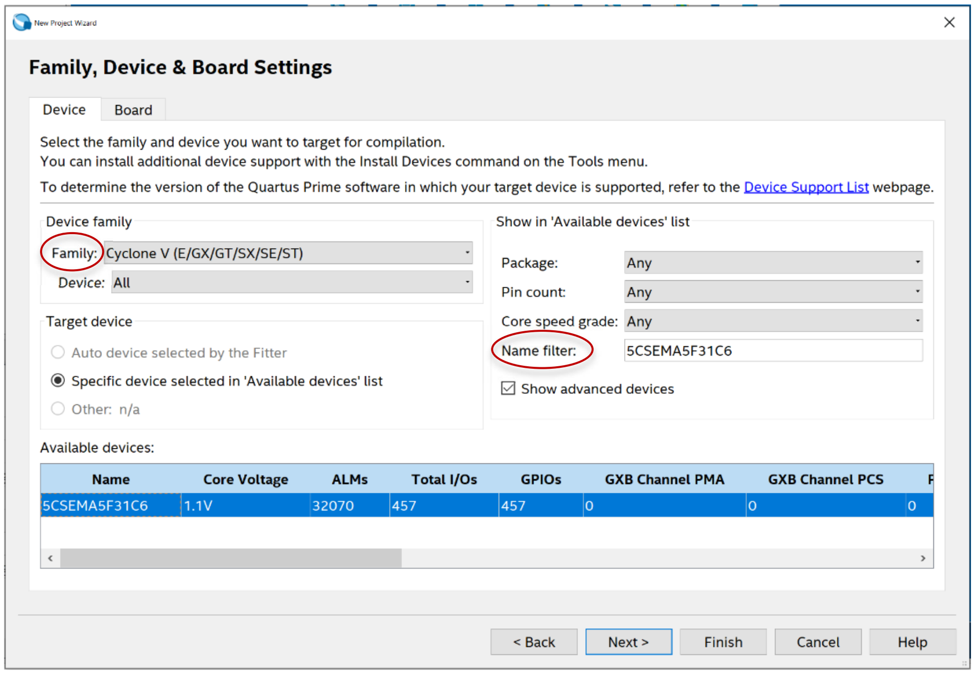
What is the name of the top-level design entity for this project? This name is case sensitive and must exactly match the entity name in the design file.

Note: Make sure that naming your projects and any of its interfaces follow this rule: The name is composed of a space-free sequence of letters from the alphabet, the digits (0, 1, ... , 9), the underscore(_), and the $ symbol. The name may not begin with a digit or $.

e) Choose the **Empty project**, then click **Next**.

f) Click **Next** to skip the **Add Files** step.

g) In the **Family, Device & Board Settings** window, type in **5CSEMA5F31C6** in the **Name filter** and this device will appear on the *Available devices* window at the bottom. Click on the device to highlight it and then click **Next**.



New Project Wizard

Family, Device & Board Settings

Device | Board

Select the family and device you want to target for compilation.
You can install additional device support with the Install Devices command on the Tools menu.
To determine the version of the Quartus Prime software in which your target device is supported, refer to the Device Support List webpage.

Device family
Family: Cyclone V (E/GX/GT/SX/SE/ST)
Device: All

Target device
Auto device selected by the Fitter
[✓ Specific device selected in 'Available devices' list]
Other: n/a

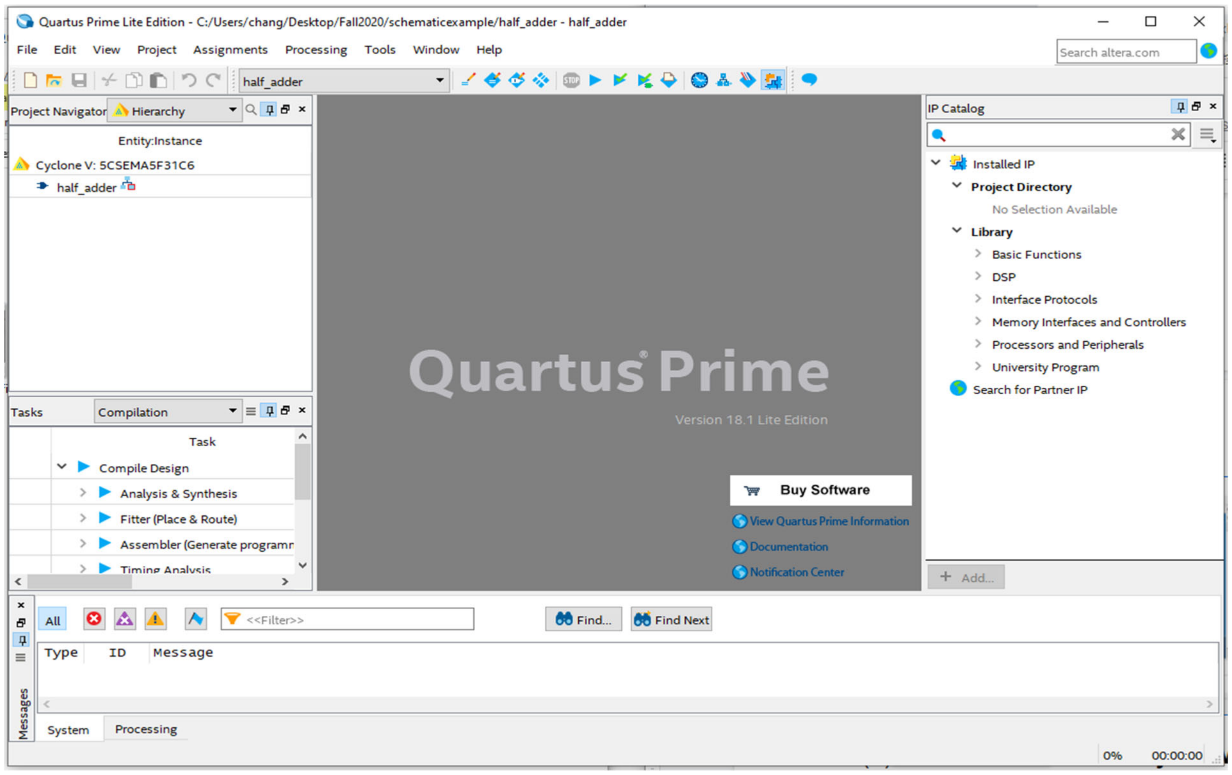
Show in 'Available devices' list
Package: Any
Pin count: Any
Core speed grade: Any
Name filter: 5CSEMA5F31C6
[✓ Show advanced devices]

Available devices:

Name	Core Voltage	ALMs	Total I/Os	GPIOs	GXB Channel PMA	GXB Channel PCS	F
5CSEMA5F31C6	1.1V	32070	457	457	0	0	0

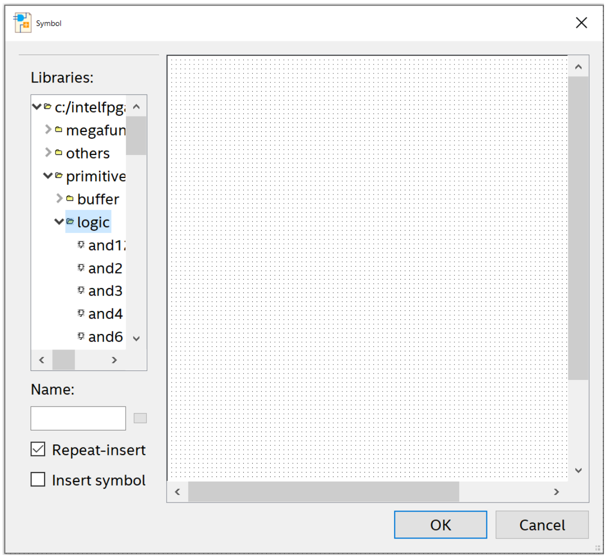
h) Click **Next** for the **EDA Tool Settings** leaving everything as default.

i) Click **Finish** to complete the setting of the project and get the following screen.



Quartus Prime Lite Edition - C:/Users/chang/Desktop/Fall2020/schematicexample/half_adder - half_adder

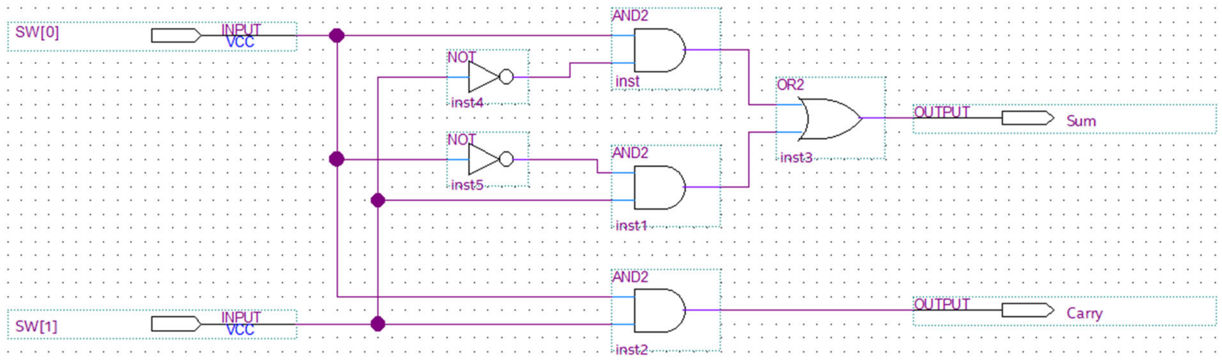
4. Creating the project schematic:
   a) Click on **File → New**. Select **Block Diagram/ Schematic File** and click **Ok**.
   b) Click on **File → Save As** and save the schematic as **half_adder.bdf**
   c) Click on **Project → Add Current File to Project**
   d) Now you can see a dotted screen. Access the **Symbol** library by clicking on the Symbol Tool on the top of the dotted screen.
   Hint: to allow for a bigger space to work on, detach the schematic screen by clicking **Window → Detach Window**.
   e) Some basic logic gates such as **and2**, **or2** and **not** can be found under **Symbol → primitives → logic** as shown.
   Double click on the gate to insert it in the schematic.



☑ Repeat-insert
☐ Insert symbol

Hint: you can also search for an arbitrary symbol by typing its full name in the **Name** box.

   f) To add inputs and outputs, use menu (next to the Symbol Tool)
   g) To connect the logic gates, use **Orthogonal** or **Diagonal Node Tool**, in the tools group: or use the **Selection and Smart Drawing Tool** button to make the connections.
   h) In your schematic screen build the half-adder circuit following the schematic shown below.



Note: Only on your personal Windows computers, if the text in the schematic is not scaling properly perform the following:

- On the desktop screen, right click the Quartus icon → select Properties → Select the Compatibility tab → click on Change high DPI settings → check the box that says Override high DPI scaling behavior → In the drop-down menu below this item, select System → Click Ok twice.

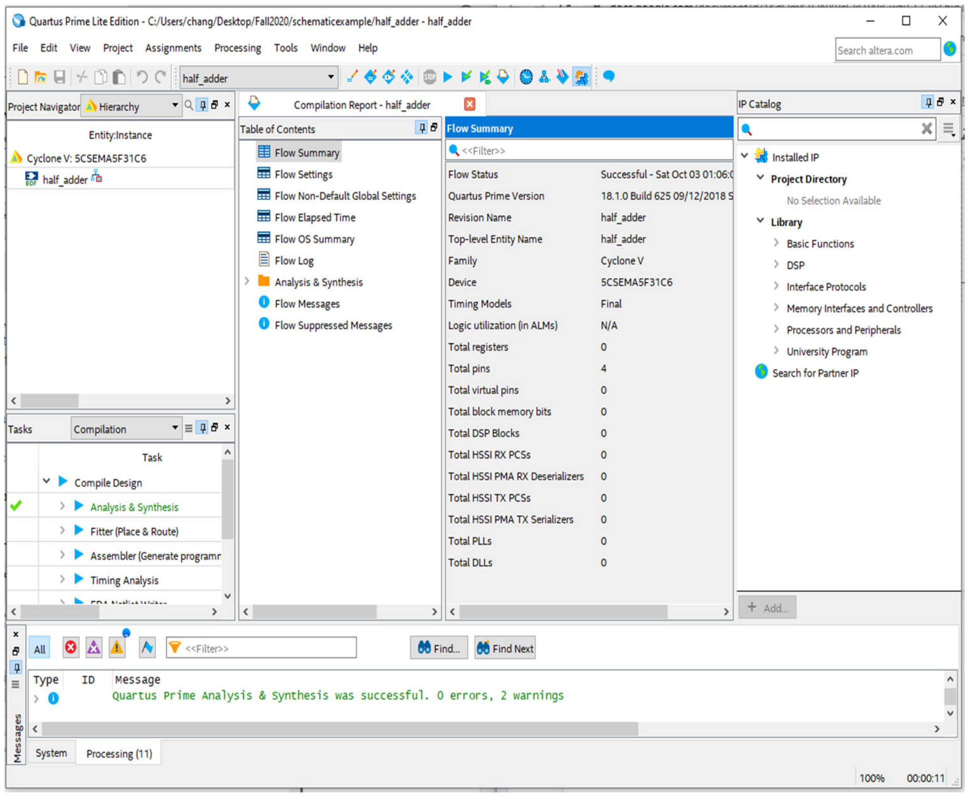
b) Right click on the **Input pins** and select **Properties** then rename them as SW[0] and SW[1]. Also rename the output pins as Sum and Carry as shown in the figure above.

c) Save the design by pressing Ctrl + S.

<u>*Note*</u>: As you are carrying out step 5 below and while trying to compile the simulation waveform, if you get an error that includes: "*… Waveform.vwf.vt used with --testbench_file option contains a non-existent directory path …*" you can fix this error by going to the directory, using Windows File Explore, where the `Waveform.vwf` file is previously saved → delete the `Waveform.vwf` file → close the Quartus software and open it again. Then you can make a brand-new simulation file through the normal procedure (New → University Program VWF) and use that to run a new simulation. Make sure you name the new *University Waveform VWF* as '`Waveform`' and that no other files in the directory have the same name.

5. Running the project simulation with Waveform:

a) Click on the third icon ▶ ✓ ✓ to **Start Analysis & Synthesis**. Wait until you see a message at the bottom of the window (as shown below) that indicates *successful* analysis and synthesis (ignore the warnings). If there are any errors, fix your schematic until you receive a successful analysis and synthesis.

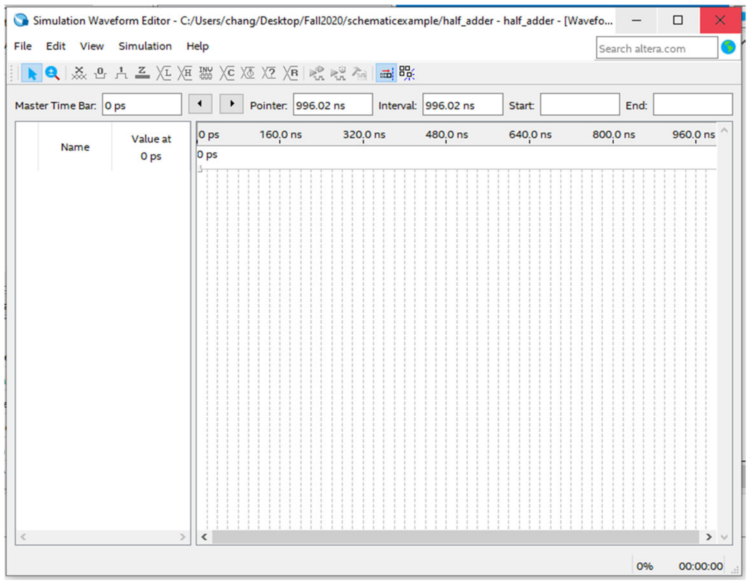


Flow Summary

Flow Status	Successful - Sat Oct 03 01:06:0
Quartus Prime Version	18.1.0 Build 625 09/12/2018 S
Revision Name	half_adder
Top-level Entity Name	half_adder
Family	Cyclone V
Device	5CSEMA5F31C6
Timing Models	Final
Logic utilization (in ALMs)	N/A
Total registers	0
Total pins	4
Total virtual pins	0
Total block memory bits	0
Total DSP Blocks	0
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI PMA TX Serializers	0
Total PLLs	0
Total DLLs	0

```
Type  ID  Message
          Quartus Prime Analysis & Synthesis was successful. 0 errors, 2 warnings
```

b) Click on **File → New**. From the **Verification/Debugging Files** list, Select **University Program VWF** and click **Ok** to get the following **Simulation Waveform Editor** screen.

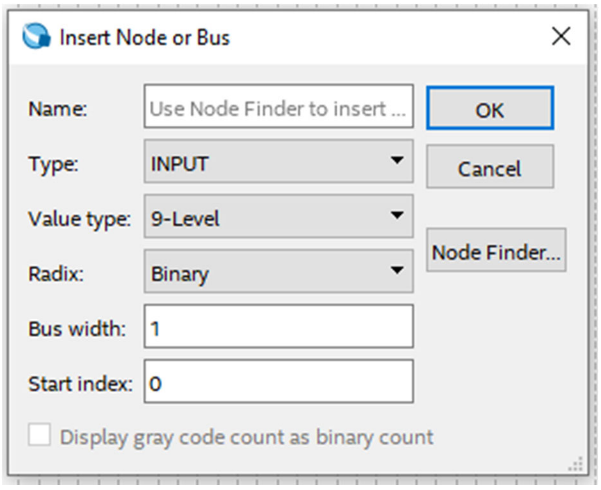


Simulation Waveform Editor - C:/Users/chang/Desktop/Fall2020/schematicexample/half_adder - half_adder - [Wavefo...

Master Time Bar: 0 ps
Pointer: 996.02 ns
Interval: 996.02 ns
Start:
End:

c) In that Editor screen, set the desired simulation to run from 0 to 200ns by selecting **Edit → Set End Time** and enter 200 ns in the window that pops up. Select **View → Fit in Window** to fit the simulation time range in the window.

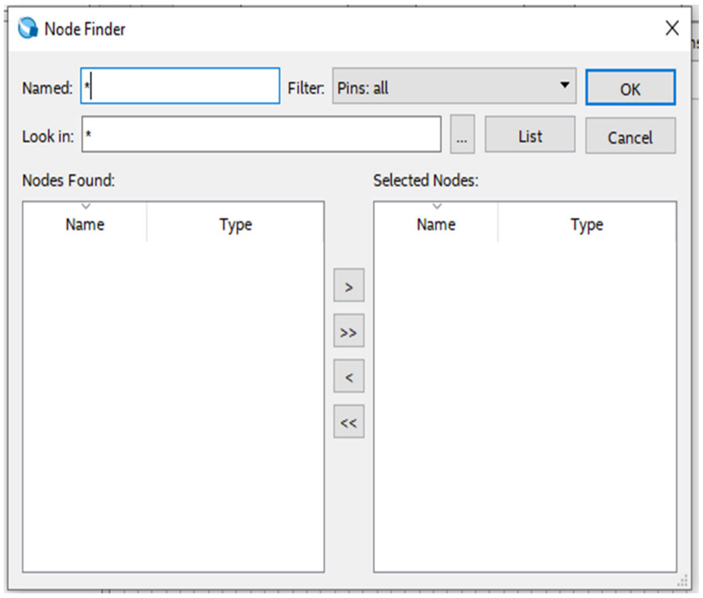
d) Right click on the blank space of the left window and select **Insert Node or Bus**.



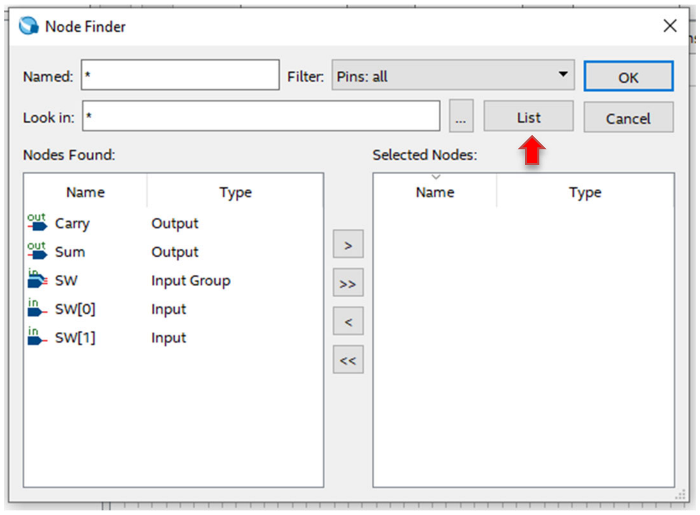
Insert Node or Bus

Name: Use Node Finder to insert ...
Type: INPUT
Value type: 9-Level
Radix: Binary
Bus width: 1
Start index: 0
☐ Display gray code count as binary count

e) Select **Node Finder**.

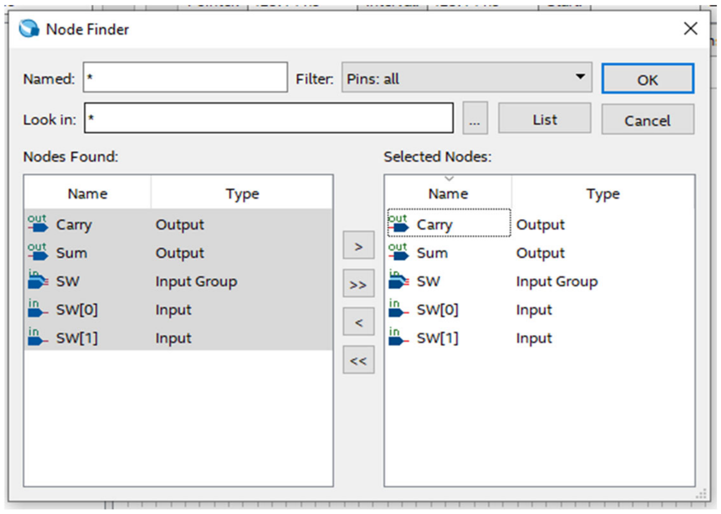


f) Click on **List**.



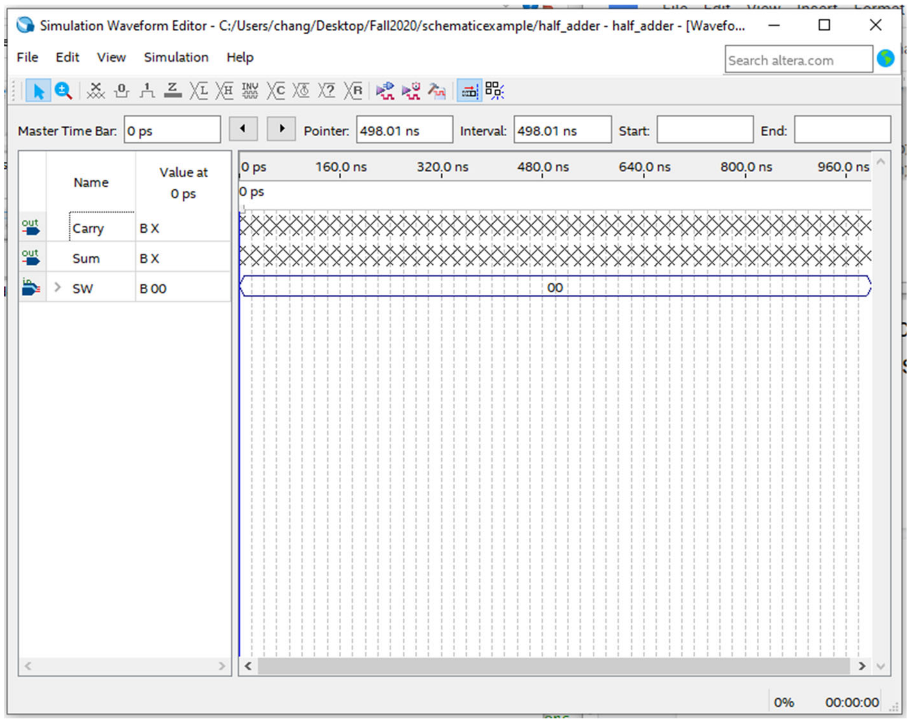
Name	Type
Carry	Output
Sum	Output
SW	Input Group
SW[0]	Input
SW[1]	Input

g) Click >> to copy all to **Selected Nodes** on the right-hand side.



Name	Type
Carry	Output
Sum	Output
SW	Input Group
SW[0]	Input
SW[1]	Input

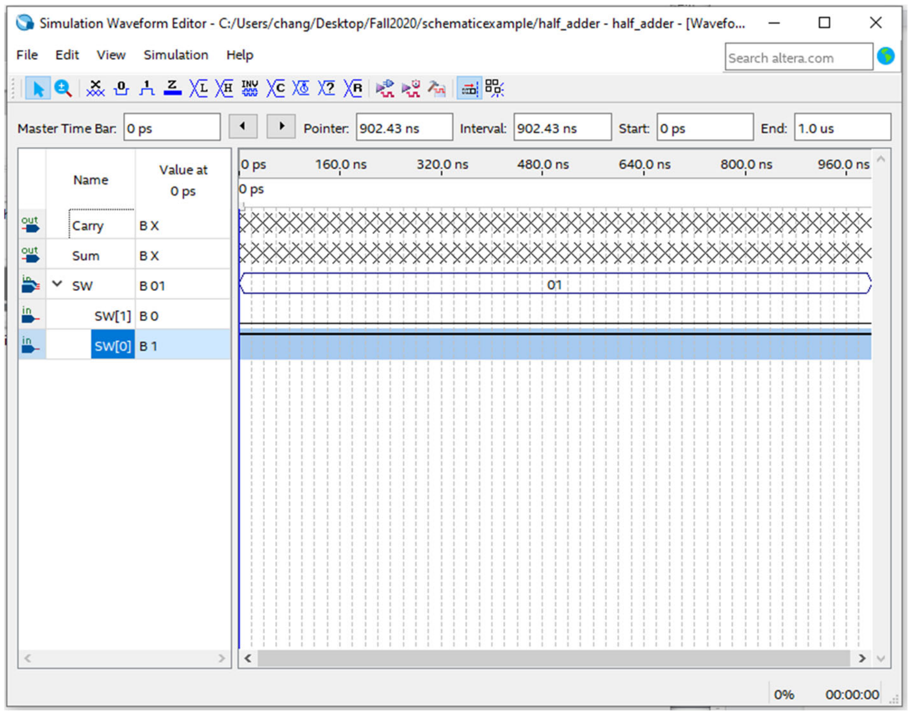
h) Click **OK**, then click another **OK** on the **Insert Node or Bus** window.


Simulation Waveform Editor - C:/Users/chang/Desktop/Fall2020/schematicexample/half_adder - half_adder - [Wavefo...
File Edit View Simulation Help
Search altera.com
Name
Value at 0 ps
0 ps
160.0 ns
320.0 ns
480.0 ns
640.0 ns
800.0 ns
960.0 ns
0%
00:00:00

Master Time Bar: 0 ps Pointer: 498.01 ns Interval: 498.01 ns Start: End:

Name	Value at 0 ps
Carry	B X
Sum	B X
SW	B 00

i) You can click on the arrow SW to expand the inputs and select different inputs as shown below. Then you can use on the top to change the input values. For example, change SW[0] to be 1.

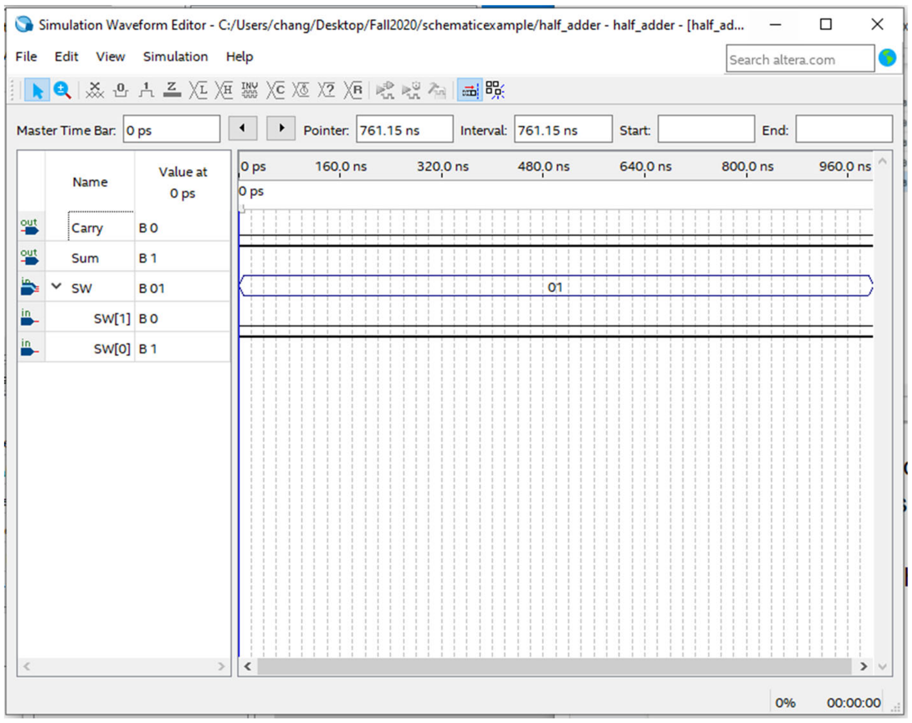

Simulation Waveform Editor - C:/Users/chang/Desktop/Fall2020/schematicexample/half_adder - half_adder - [Wavefo...
File Edit View Simulation Help
Search altera.com
Name
Value at 0 ps
0 ps
160.0 ns
320.0 ns
480.0 ns
640.0 ns
800.0 ns
960.0 ns
0%
00:00:00

Master Time Bar: 0 ps Pointer: 902.43 ns Interval: 902.43 ns Start: 0 ps End: 1.0 us

Name	Value at 0 ps
Carry	B X
Sum	B X
SW	B 01
SW[1]	B 0
SW[0]	B 1

j) Click on to run **Functional Simulation**. Click **Yes** to save the changes when the new window pops up. Click **Save** if prompted to save the waveform (`.vwf`) file. *Note:* you can later open that file from the Quartus project window to re-run the same simulation.

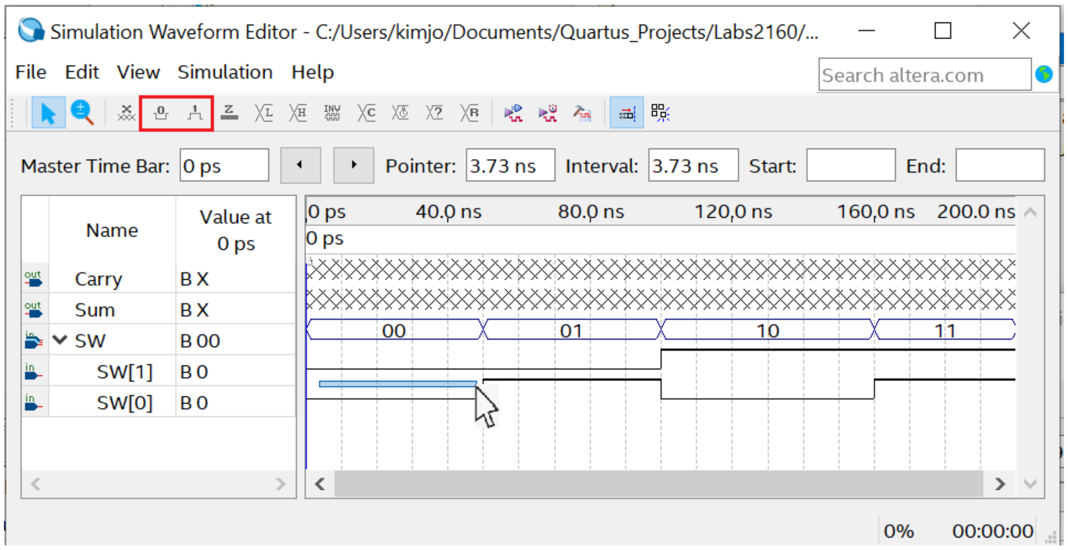
k) A second Waveform Editor window will open with the results of the simulation. You should see the **Sum** goes to 1 in the simulation result.


Simulation Waveform Editor - C:/Users/chang/Desktop/Fall2020/schematicexample/half_adder - half_adder - [half_ad...

Master Time Bar: 0 ps Pointer: 761.15 ns Interval: 761.15 ns Start: End:

Name	Value at 0 ps
Carry	B 0
Sum	B 1
SW	B 01
SW[1]	B 0
SW[0]	B 1

On the original Waveform Editor window, set SW[1] to 0 in the time interval 0 to 100 ns, which is probably already set by default. Next, SW[1] to 1 in the time interval 100 to 200 ns. Do this by pressing the mouse at the start of the interval and dragging it to its end, which highlights the selected interval, and choosing the logic value 1 or 0 in the toolbar . Make SW[0] = 1 from 50 to 100 ns and also from 150 to 200 ns, and 0 otherwise, using the same method. Your simulation pane should look similar to the following:


Simulation Waveform Editor - C:/Users/kimjo/Documents/Quartus_Projects/Labs2160/...
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 3.73 ns
Interval: 3.73 ns
Start:
End:
0 ps 40.0 ns 80.0 ns 120.0 ns 160.0 ns 200.0 ns
0%
00:00:00

Name	Value at 0 ps
Carry	B X
Sum	B X
SW	B 00
SW[1]	B 0
SW[0]	B 0

6. Take a screenshot of the Simulation Waveform Editor window with the **expected output displayed** and include this **simulation result** in your report

Notes:

- Any saved files on the lab PC are deleted regularly. So, before signing out from the PC and if you plan to continue with your project at a later time, make sure to back up your project folder on an external storage media (e.g., on a flash or OneDrive). To continue with the project, restore its folder back to the PC.
- If you ever close your simulation file in Quartus, you should re-open the existing Waveform (*.wvf) file rather than making a new one. To do this, first open your current project (if not already open), select the 'files' option in the project navigator and you should be able to select your already created waveform file by double clicking.

3. The Half Adder on the DE1-SoC Board

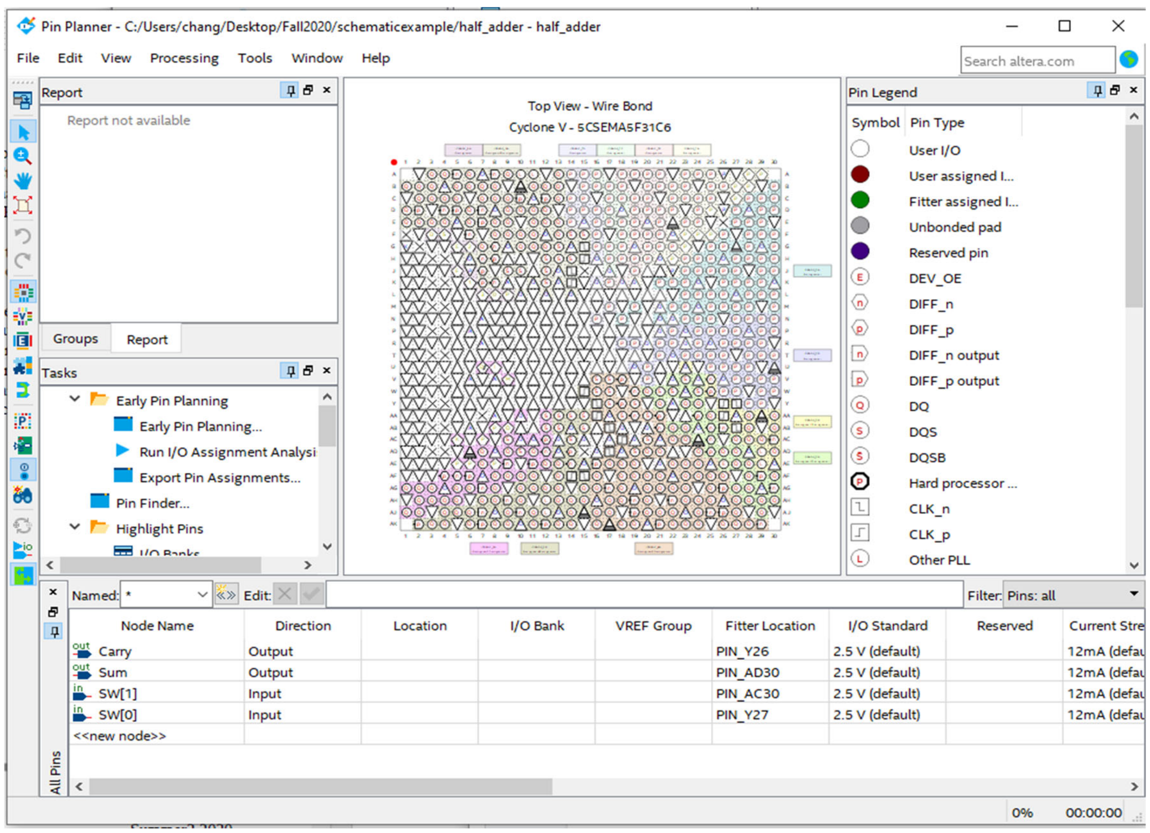
Follow the following steps to implement and test the half adder schematic on the DE1-SoC board.

a) Close the Simulation Waveform Editor if it is still open.

b) **Compile** the schematic by clicking the first button or using Ctrl + L. This step might take minutes to finish. Watch its progress in the Tasks window on the left.

c) After compilation is successful, you will see green check marks on the left-hand side **Task** window. Fix any reported errors and ignore the warnings.

d) Go to **Assignments → Pin Planner** where the following window will open.



Node Name	Direction	Location	I/O Bank	VREF Group	Fitter Location	I/O Standard	Reserved	Current Stre
Carry	Output				PIN_Y26	2.5 V (default)		12mA (defau
Sum	Output				PIN_AD30	2.5 V (default)		12mA (defau
SW[1]	Input				PIN_AC30	2.5 V (default)		12mA (defau
SW[0]	Input				PIN_Y27	2.5 V (default)		12mA (defau
<<new node>>								

To test a digital circuit on the DE1-SoC board, you will need to assign the board's FPGA pins to the circuit input/output interfaces. You will need to refer to the following tables to complete this task. The tables are copied from the DE1-SoC User Manual.

Table 3-5 Pin Assignment of Clock Inputs

Signal Name	*FPGA Pin No.*	*Description*	*I/O Standard*
CLOCK_50	PIN_AF14	50 MHz clock input	3.3V
CLOCK2_50	PIN_AA16	50 MHz clock input	3.3V
CLOCK3_50	PIN_Y26	50 MHz clock input	3.3V
CLOCK4_50	PIN_K14	50 MHz clock input	3.3V
HPS_CLOCK1_25	PIN_D25	25 MHz clock input	3.3V
HPS_CLOCK2_25	PIN_F25	25 MHz clock input	3.3V

Table 3-6 Pin Assignment of Slide Switches

Signal Name	*FPGA Pin No.*	*Description*	*I/O Standard*
SW[0]	PIN_AB12	Slide Switch[0]	3.3V
SW[1]	PIN_AC12	Slide Switch[1]	3.3V
SW[2]	PIN_AF9	Slide Switch[2]	3.3V
SW[3]	PIN_AF10	Slide Switch[3]	3.3V
SW[4]	PIN_AD11	Slide Switch[4]	3.3V
SW[5]	PIN_AD12	Slide Switch[5]	3.3V
SW[6]	PIN_AE11	Slide Switch[6]	3.3V
SW[7]	PIN_AC9	Slide Switch[7]	3.3V
SW[8]	PIN_AD10	Slide Switch[8]	3.3V
SW[9]	PIN_AE12	Slide Switch[9]	3.3V

Table 3-7 Pin Assignment of Push-buttons

Signal Name	*FPGA Pin No.*	*Description*	*I/O Standard*
KEY[0]	PIN_AA14	Push-button[0]	3.3V
KEY[1]	PIN_AA15	Push-button[1]	3.3V
KEY[2]	PIN_W15	Push-button[2]	3.3V
KEY[3]	PIN_Y16	Push-button[3]	3.3V

Table 3-8 Pin Assignment of LEDs

Signal Name	*FPGA Pin No.*	*Description*	*I/O Standard*
LEDR[0]	PIN_V16	LED [0]	3.3V
LEDR[1]	PIN_W16	LED [1]	3.3V
LEDR[2]	PIN_V17	LED [2]	3.3V
LEDR[3]	PIN_V18	LED [3]	3.3V
LEDR[4]	PIN_W17	LED [4]	3.3V
LEDR[5]	PIN_W19	LED [5]	3.3V
LEDR[6]	PIN_Y19	LED [6]	3.3V
LEDR[7]	PIN_W20	LED [7]	3.3V
LEDR[8]	PIN_W21	LED [8]	3.3V
LEDR[9]	PIN_Y21	LED [9]	3.3V

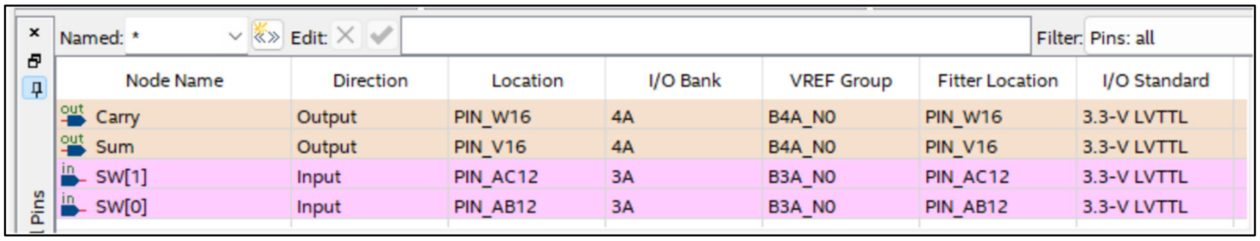
Table 3-9 Pin Assignment of 7-segment Displays

Signal Name	*FPGA Pin No.*	*Description*	*I/O Standard*
HEX0[0]	PIN_AE26	Seven Segment Digit 0[0]	3.3V
HEX0[1]	PIN_AE27	Seven Segment Digit 0[1]	3.3V
HEX0[2]	PIN_AE28	Seven Segment Digit 0[2]	3.3V
HEX0[3]	PIN_AG27	Seven Segment Digit 0[3]	3.3V
HEX0[4]	PIN_AF28	Seven Segment Digit 0[4]	3.3V
HEX0[5]	PIN_AG28	Seven Segment Digit 0[5]	3.3V
HEX0[6]	PIN_AH28	Seven Segment Digit 0[6]	3.3V
HEX1[0]	PIN_AJ29	Seven Segment Digit 1[0]	3.3V
HEX1[1]	PIN_AH29	Seven Segment Digit 1[1]	3.3V
HEX1[2]	PIN_AH30	Seven Segment Digit 1[2]	3.3V
HEX1[3]	PIN_AG30	Seven Segment Digit 1[3]	3.3V
HEX1[4]	PIN_AF29	Seven Segment Digit 1[4]	3.3V
HEX1[5]	PIN_AF30	Seven Segment Digit 1[5]	3.3V
HEX1[6]	PIN_AD27	Seven Segment Digit 1[6]	3.3V
HEX2[0]	PIN_AB23	Seven Segment Digit 2[0]	3.3V
HEX2[1]	PIN_AE29	Seven Segment Digit 2[1]	3.3V
HEX2[2]	PIN_AD29	Seven Segment Digit 2[2]	3.3V
HEX2[3]	PIN_AC28	Seven Segment Digit 2[3]	3.3V
HEX2[4]	PIN_AD30	Seven Segment Digit 2[4]	3.3V
HEX2[5]	PIN_AC29	Seven Segment Digit 2[5]	3.3V
HEX2[6]	PIN_AC30	Seven Segment Digit 2[6]	3.3V
HEX3[0]	PIN_AD26	Seven Segment Digit 3[0]	3.3V
HEX3[1]	PIN_AC27	Seven Segment Digit 3[1]	3.3V
HEX3[2]	PIN_AD25	Seven Segment Digit 3[2]	3.3V
HEX3[3]	PIN_AC25	Seven Segment Digit 3[3]	3.3V
HEX3[4]	PIN_AB28	Seven Segment Digit 3[4]	3.3V
HEX3[5]	PIN_AB25	Seven Segment Digit 3[5]	3.3V
HEX3[6]	PIN_AB22	Seven Segment Digit 3[6]	3.3V
HEX4[0]	PIN_AA24	Seven Segment Digit 4[0]	3.3V
HEX4[1]	PIN_Y23	Seven Segment Digit 4[1]	3.3V
HEX4[2]	PIN_Y24	Seven Segment Digit 4[2]	3.3V
HEX4[3]	PIN_W22	Seven Segment Digit 4[3]	3.3V
HEX4[4]	PIN_W24	Seven Segment Digit 4[4]	3.3V
HEX4[5]	PIN_V23	Seven Segment Digit 4[5]	3.3V
HEX4[6]	PIN_W25	Seven Segment Digit 4[6]	3.3V
HEX5[0]	PIN_V25	Seven Segment Digit 5[0]	3.3V
HEX5[1]	PIN_AA28	Seven Segment Digit 5[1]	3.3V
HEX5[2]	PIN_Y27	Seven Segment Digit 5[2]	3.3V
HEX5[3]	PIN_AB27	Seven Segment Digit 5[3]	3.3V
HEX5[4]	PIN_AB26	Seven Segment Digit 5[4]	3.3V
HEX5[5]	PIN_AA26	Seven Segment Digit 5[5]	3.3V
HEX5[6]	PIN_AA25	Seven Segment Digit 5[6]	3.3V

e) For your half-adder, go to the **Location** column in the Pin Planner and assign LEDR[1] **PIN_W16** to the **Carry** output, LEDR[0] **PIN_V16** to the **Sum** output, **PIN_AC12** to **SW[1]**, **PIN_AB12** to **SW[0]**. Make sure to type the pin names correctly.

f) Change all I/O Standards to **3.3-V LVTTL** by double clicking and using the drop-down menu on each input or output's I/O Standard entry. Close the **Pin Planner.**

Hint: to change all the I/O Standard entries simultaneously, change only one entry to 3.3-V LVTTL and copy it (Ctrl+C). Then select all remaining entries in the I/O Standard column by clicking and dragging the cursor and then click Ctrl+V to paste. You can also copy the PIN No value (e.g., **PIN_W16**) from the above tables and paste it into the **Location** entry as shown below.

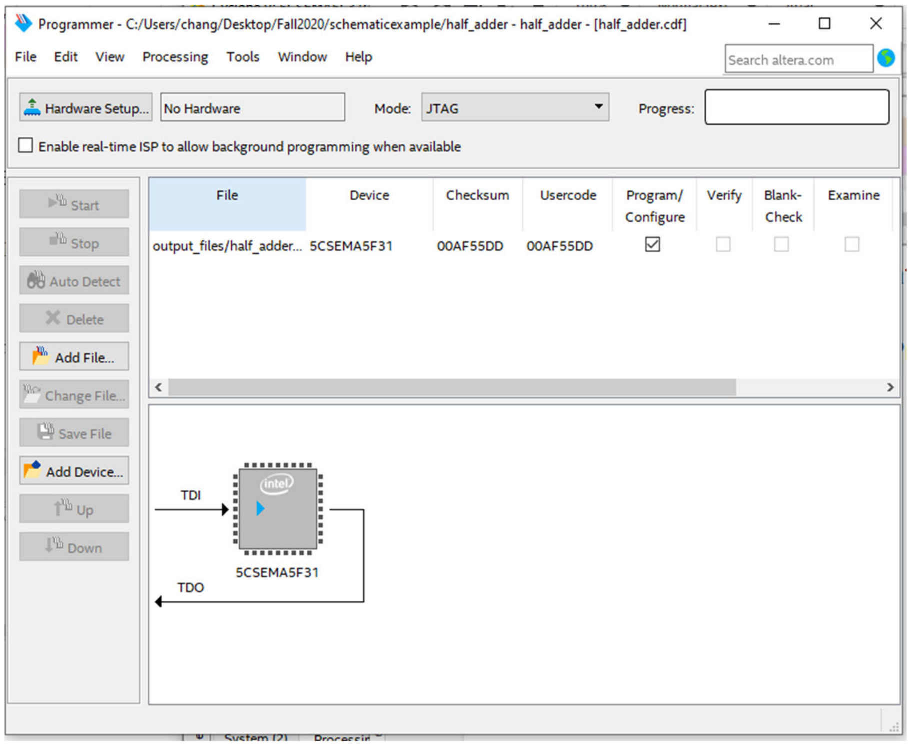


Node Name	Direction	Location	I/O Bank	VREF Group	Fitter Location	I/O Standard
Carry	Output	PIN_W16	4A	B4A_N0	PIN_W16	3.3-V LVTTL
Sum	Output	PIN_V16	4A	B4A_N0	PIN_V16	3.3-V LVTTL
SW[1]	Input	PIN_AC12	3A	B3A_N0	PIN_AC12	3.3-V LVTTL
SW[0]	Input	PIN_AB12	3A	B3A_N0	PIN_AB12	3.3-V LVTTL

g) On the main Quartus window **Compile** the schematic again and wait for the green check-offs in the **Tasks** window (it is on the left).

h) Make sure your DE1-SoC board is powered using its power adapter and connected using the *USB-B to USB-A cable*. To turn on the board, press the big red button near the power adapter connection.

i) Go to the **Tasks** window and double click on the **Program Device** to open the following window.



No Hardware

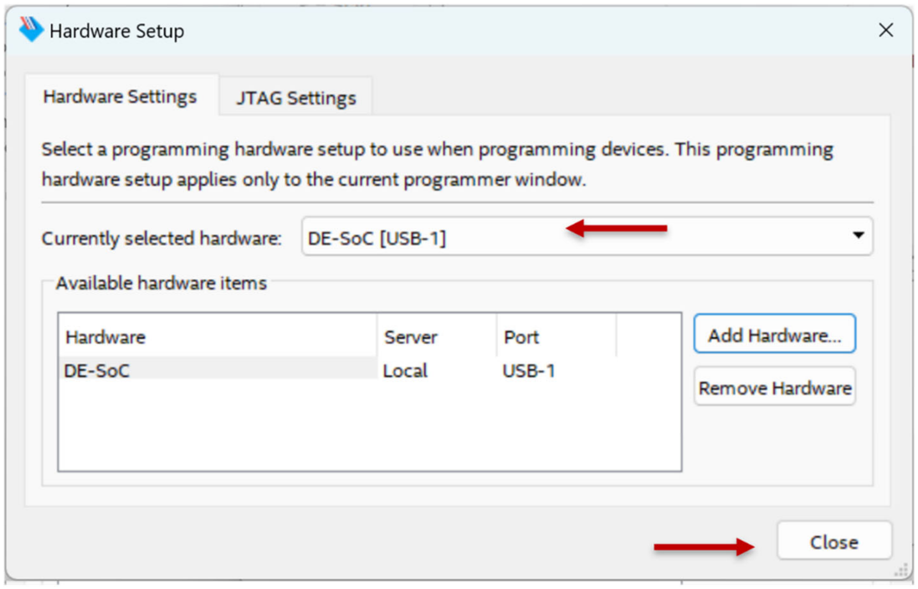
Mode: JTAG

Progress:

☐ Enable real-time ISP to allow background programming when available

File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank-Check	Examine
output_files/half_adder...	5CSEMA5F31	00AF55DD	00AF55DD	☑	☐	☐	☐

j) Click on **Hardware Setup** and make sure it is assigned to DE1-SOC from the drop-down and hit **Close**. (if no hardware is detected, make sure the USB Blaster II driver is installed properly by going to the Device manager on Windows and making sure it is listed under JTAG cables).



Hardware Setup

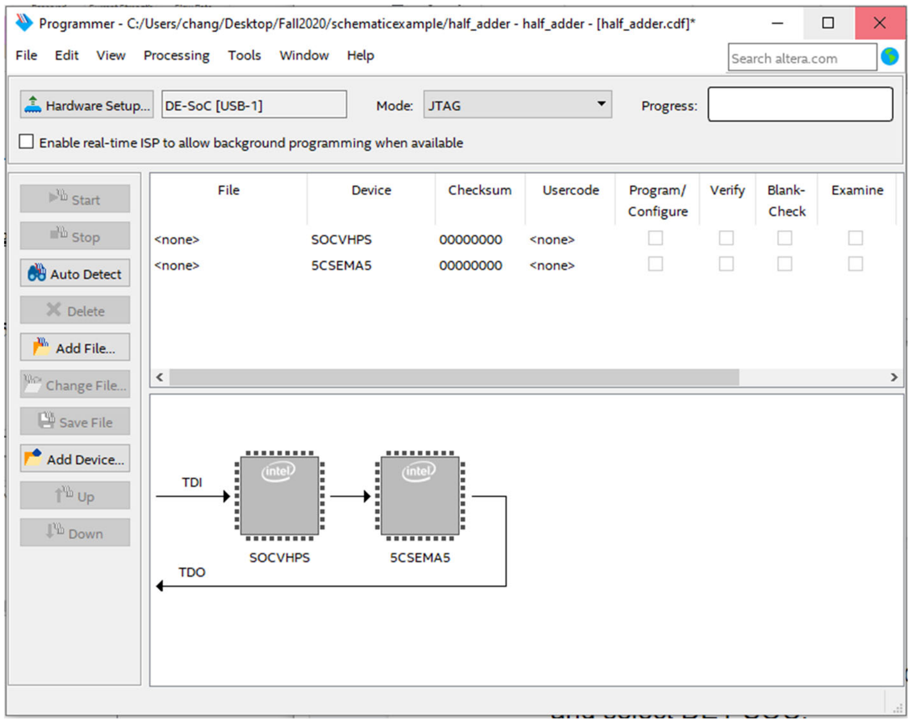
Select a programming hardware setup to use when programming devices. This programming hardware setup applies only to the current programmer window.

Currently selected hardware: DE-SoC [USB-1]

Available hardware items

Hardware	Server	Port
DE-SoC	Local	USB-1

k) Click **Auto-Detect** on the left side and select **5CSEMA5**. Click **OK** then **Yes** when a new window shows up.



Programmer - C:/Users/chang/Desktop/Fall2020/schematicexample/half_adder - half_adder - [half_adder.cdf]*

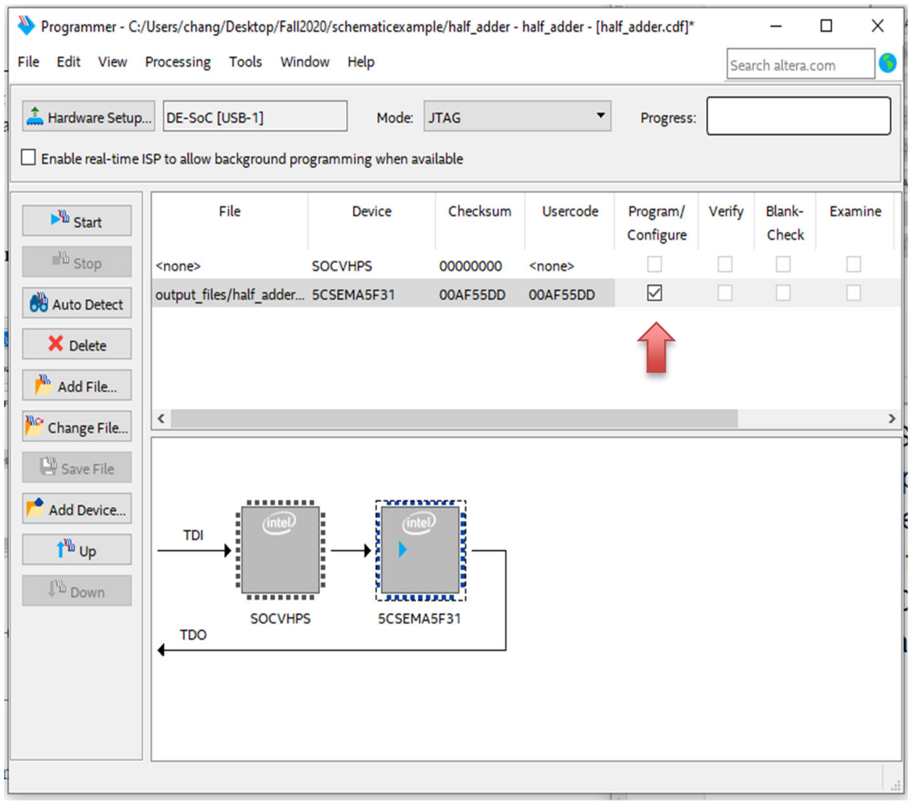
Hardware Setup... DE-SoC [USB-1] Mode: JTAG Progress:

Enable real-time ISP to allow background programming when available

File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank-Check	Examine
<none>	SOCVHPS	00000000	<none>	☐	☐	☐	☐
<none>	5CSEMA5	00000000	<none>	☐	☐	☐	☐

l) Select and right click on the **5CSEMA5** chip → select **Edit → Change File** →Browse inside folder **output_files** and select **half_adder.sof** → click **Open**.

m) In the middle screen shown on top of the chips diagram, check on **Program/Configure** check box and then press **Start** on the left-hand side.


Programmer - C:/Users/chang/Desktop/Fall2020/schematicexample/half_adder - half_adder - [half_adder.cdf]*

Hardware Setup... DE-SoC [USB-1] Mode: JTAG Progress:

☐ Enable real-time ISP to allow background programming when available

File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank-Check	Examine
<none>	SOCVHPS	00000000	<none>	☐	☐	☐	☐
output_files/half_adder...	5CSEMA5F31	00AF55DD	00AF55DD	☑	☐	☐	☐

n) After the top-right Progress message shows 100%, you can play with the switches SW[1], SW[0], and check the outputs on the LEDs.

o) Pack the half adder schematic as a logic block named **half_adder.bsf** for later use. To do this, **go back to your half adder schematic**, then on the **File** main menu, select **Create/Update → Create Symbol Files for Current File** and save the schematic as **half_adder.bsf** file.

p) Close the project by selecting **Close Project** under the **File** main menu.

Assignment 1
Include a screenshot of your **half adder schematic**, and **simulation results**, and add them to your report. Test your circuit by applying different inputs and observing Sum and Carry. **Set both inputs to be 1 and take a picture of the Board**.

4. The Full Adder Circuit

The full adder is a circuit capable of adding three 1-bit numbers. The circuit takes three inputs and generates two outputs, for example, $1_2+1_2+1_2= 11_2$. Like in the half adder, the least significant bit in the result is the Sum (***S***), and the most significant bit is the Carry (***C***).

a) Complete the shown truth table for the full adder circuit.

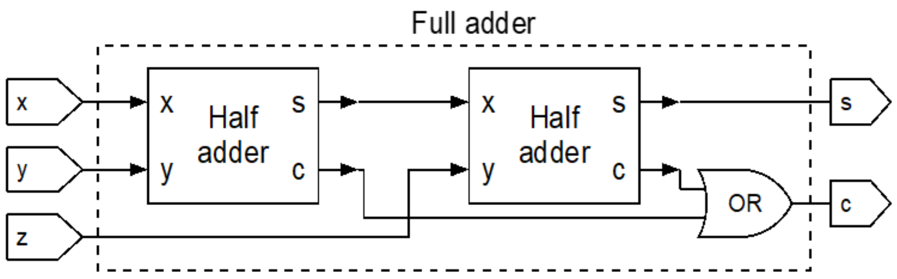
b) Create a new folder for a new project named **full_adder**.

c) Copy the **half_adder.bsf** and the **half_adder.bdf** files from your previous project **half_adder** folder to this new **full_adder** project folder.

d) Create the new project following the same steps outlined on the half adder project. Set the project folder to the **full_adder** folder and project name to **full_adder**.

e) Build a full adder schematic with **two half adders** and an **OR** gate as shown below.

X	Y	Z	S	C
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



- When adding the components in your schematic, you will notice a new library in the **Symbol Tool library** (on the top of the dotted screen) with your **half_adder module**. Use it as you would with any other component.
- Connect the 3 inputs **x**, **y**, and **z** to switches **SW[2]**, **SW[1]**, and **SW[0]** respectively, and the outputs **s** (for sum) and **c** (for carry out) to **LEDR[0]** and **LEDR[1]** respectively for testing on the DE1-SoC board. Check tables 3-6 and 3-8 on page 12 for FPGA Pin No.

f) Test the schematic on the DE1-SoC board following the same steps you had with the half_adder project.

g) Pack the full adder schematic as a logic block named **full_adder.bsf** for later use.

h) Close the project by selecting **Close Project** under the **File** main menu.

Assignment 2
Design and implement the **full adder** in Quartus schematic. **Include a screenshot** of your **full adder schematic**. Test your circuit on the Board using at least 4 combinations of the inputs that result in four different sum and carry outputs. Include **pictures** of the board for these tests.

5. Lab Submission Instructions

- The lab is intended to be implemented by each **individual** student. It is also allowed for a **maximum of two students** to work in the lab together. If you decide to work in a group of two students, **both students** must **check off** the working lab assignments with a lab instructor and only **one student** must complete the lab submission as explained in the lab assignment on Canvas.
- Write your lab report following the **report template** provided on Canvas. If you are working with a partner, write both names/IDs on the cover page. It is **prohibited** to include a partner's name on a lab report cover page if that partner does not actively participate in implementing the lab.
- The pre-lab questions prepare you for the challenges you will be presented with in the actual lab. Have their answers ready before the lab session and include these answers in the lab report together with the lab assignments.
- For each lab, submit the following on Canvas before the announced due date/time (only submissions on Canvas are accepted):
    - The lab report as a single Word or PDF document
    - Any files the assignments might ask for (e.g., a demonstration video file)
    - Submit each of the above files separately (do not upload compressed files).
- You can submit multiple attempts for this lab; however, only what you submit in the last attempt will be graded (i.e., all required reports and files must be included in this last attempt).
- **Late submission** of any lab assignment after the given due date and time will be penalized at 1 point per every late hour. No assignment submission will be accepted 48 hours after its due date/time.
- **Successful submission**: it is your responsibility to verify that you submitted successfully all the required files for the lab. First, to verify that your files were submitted, go to the assignment details page, and make sure that it says "Successfully submitted" under grade. Second, you need to validate your submission details by clicking the Submission and Rubric link on that page.
- **Deliberate plagiarism** in any assignment is an extremely serious offense that may result in failing the course. Plagiarism includes submitting answers that are not yours (e.g., copied from someone else or from the Internet). Copying materials from the course slides posted on Canvas is not considered plagiarism.