Familiarisation Quartus design environment

Objectives:

The aim of this module is to introduce you to the Intel Quartus Prime Development Suite (Quartus Prime 21.1.1 Lite Edition)(or any other version of Quartus) <mark>and implement using Intel DE1 SOC FPGA board</mark>

Introduction

This module introduces the Intel Quartus Prime 21.1.1 Lite Development tools to create a simple digital circuit using Verilog. The design flow consists of creating the Verilog model, functional simulation using Questasim Intel FPGA Starter Edition, implementation, and testing of the model using DE1 SOC FPGA board with Intel Cyclone V FPGA. We will be using the **full adder** circuit as an illustrative example. You are encouraged to make full use of the online help and tutorials, **Help → PDF Tutorials,** provided by the software for detailed information on various aspects of the tools. A typical design flow is given below. The circled number indicates the corresponding step in this lab.
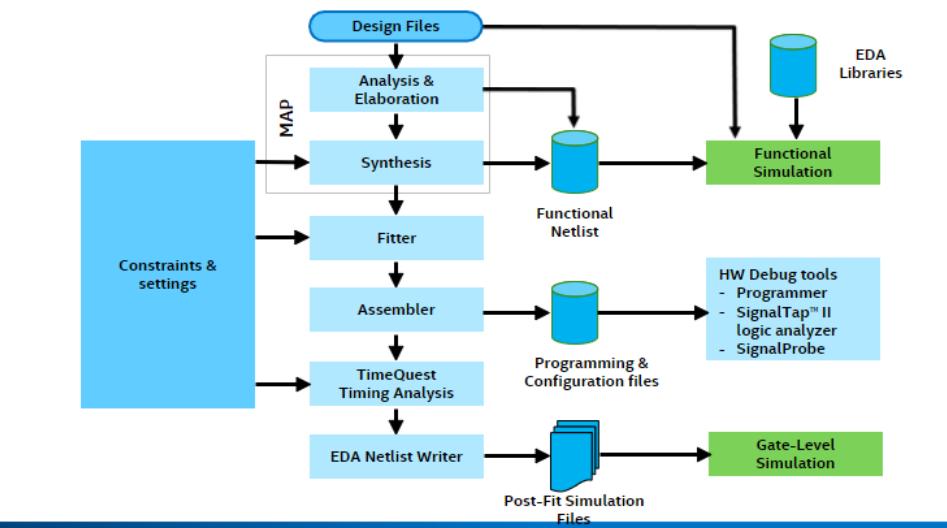

Fig.1.1 A typical design flow

1. Create a Project in Quartus

- Launch **Quartus 21.1.1 Lite**
- Click **Create New Project Wizard** to start the wizard.
- **Pane 1: Introduction** - There is nothing to configure on this pane. Click **Next** to continue.
- **Pane 2: Directory, Name, Top-Level Entity**
    - o Fill in with a directory of your choice. It is recommended to be a personal directory, and not a directory under Quartus installation which is the default.
    - o Call the project **Lab0** and the top level entity **fulladder**
- **Pane 3: Project Type**. Select **Empty project**. Click **Next**.
- **Pane 4: Add Files**. Click **Next**. We will add project source files later.
- **Pane 5: Family, Device, and Board Settings**.
    - o Select **DE1-SOC Board** in **Board – Development Kit.** Click **Next**
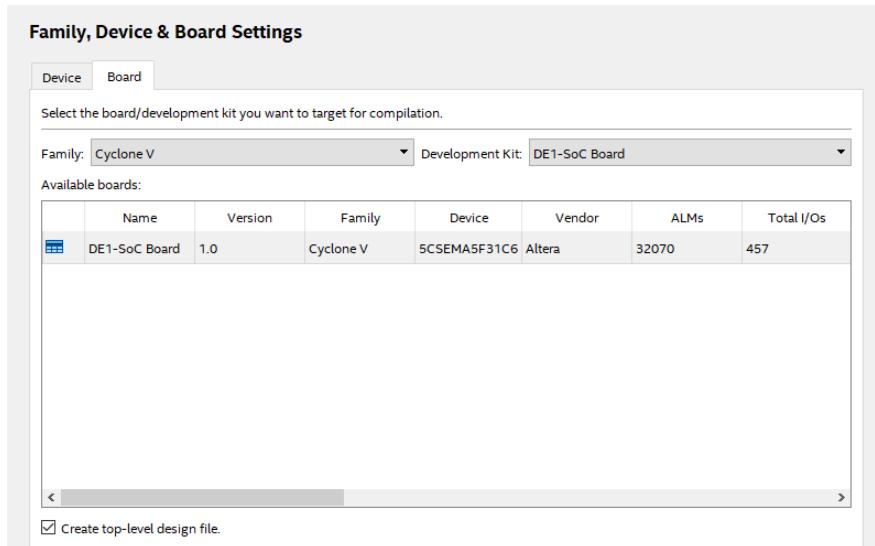
Fig.1.2 Board Selection

- **Pane 6: EDA Tool Settings**
    - o In **Simulation** select **Questa-Intel FPGA** and **Verilog HDL.** Click **Next**
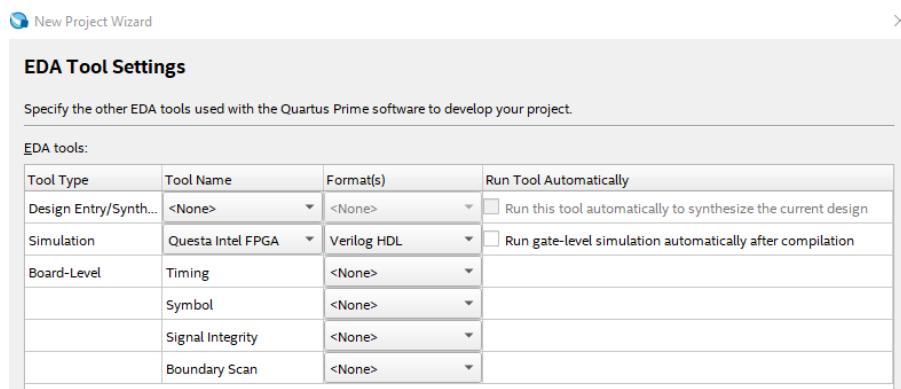
Fig.1.3 EDA Tool Selection

- **Pane 7: Summary**. Click **Finish**
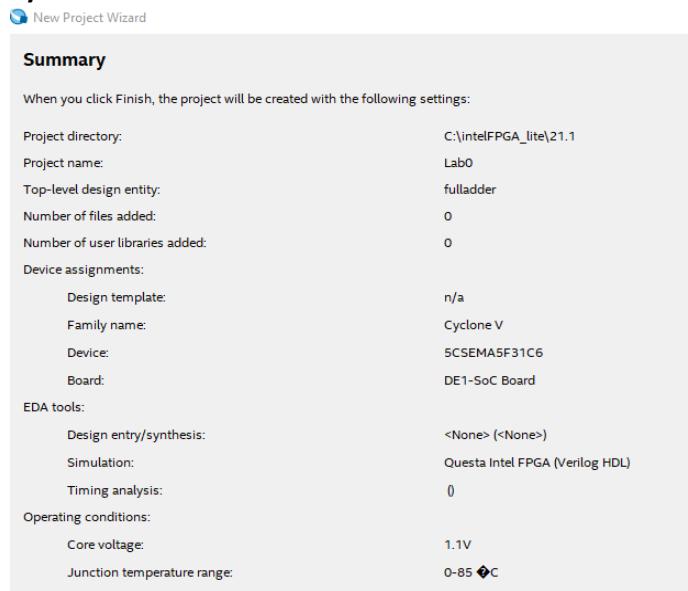
Fig. 1.4 Summary

- You should now see something similar to Fig. 1.5. (The Tool View Window may just have a gray Quartus Prime screen). Some windows may not be shown by default. To customize what windows are shown, click on the **View** tab and look under the **Utlity Windows** drop down
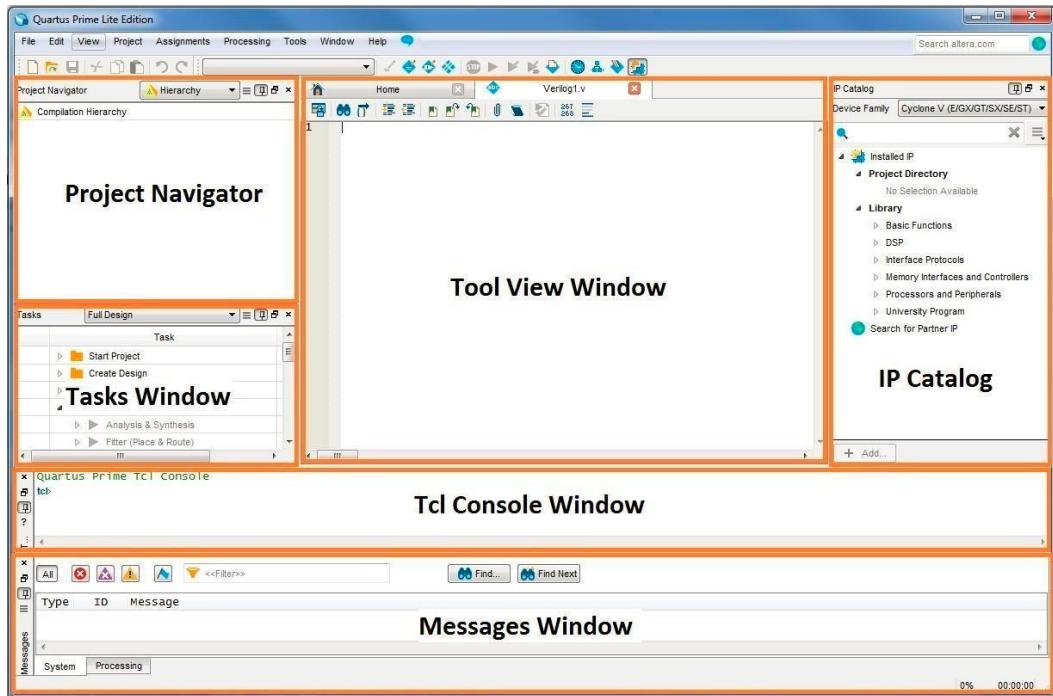

Fig. 1.5 Quartus Prime Project Window

- If you navigate to your project directory, you will see some files and directories created by the New Project Wizard as part of the setup process.

2. Creating the design
- Create a Verilog HDL file. Go the **File** dropdown menu and select **New.**
- A window, should pop up**.** Click on **Verilog HDL File** and then **Ok.**
- Create a simple module in your Verilog HDL file by typing in the following code.

```
module fulladder(
    input a,
    input b,
    input cin,
    output sum,
    output cout
    );
  assign sum =  a ^ b ^ cin;
  assign cout = (a & b) | (a & cin) | (b & cin);
  endmodule
```

- In **File**, Click on **Save As**, name the file as **fulladder** (ensuring case-sensitivity), and save your Verilog file
- Next you will run Analysis and Elaboration. Analysis and Elaboration checks the syntax of your Verilog code, resolves references to other modules and maps to FPGA logic. If you see any errors during the Analysis and Elaboration step, carefully review your Verilog code for syntax errors and re-run this step.
- To run Analysis and Elaboration, click the **Play** button with a green check mark.
- If there are errors in the code, they are shown in the **Messages** window
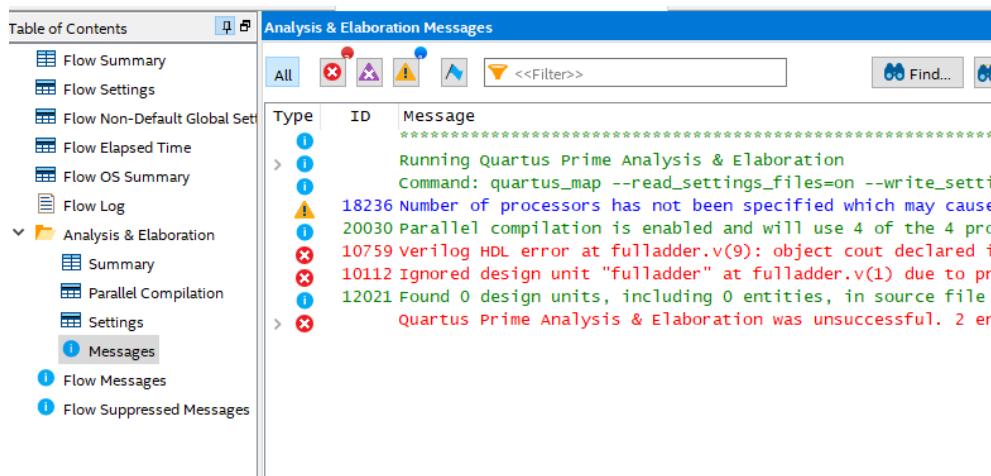
Fig. 1.6 Error messages

- **Assigning Pins -** By default, Quartus Prime does not know how the FPGA pins on the DE1-SOC development board are connected to the switches and LEDs. Because our FPGA is already on a PCB, we need to tell Quartus what pins to use. Although Quartus allows you to select a development board with a predefined pinout, this lab shows you how to define your own pinout as an exercise.
- The next steps will assign the switches and LED signals in your code to the appropriate pins.
- Using the main toolbar at the top of the Quartus window, navigate to the **Assignments** dropdown menu. Click on **Pin Planner** and a window similar to the image in Fig 1.7 should open.
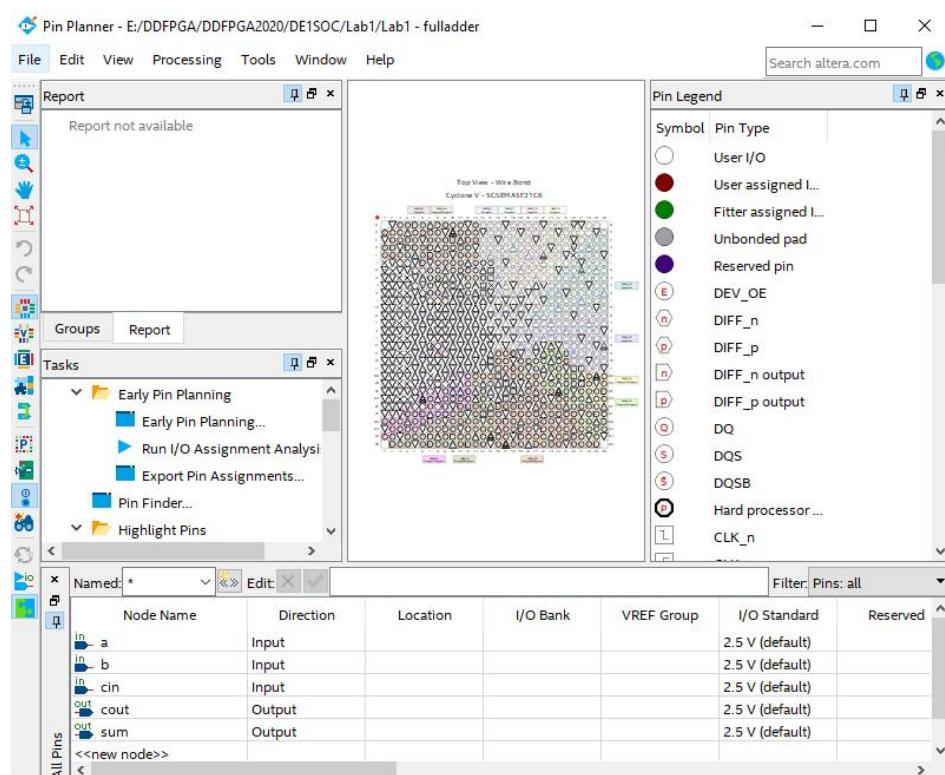

Fig. 1.7 Pin Planner

- We can see the I/O pins have not been assigned to any locations yet. To make the right pin assignments for this project, we have to look it up in development board manual. https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&No=836&PartNo=4

Table 1.1.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 1.2.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

- Click on Location and select the PINs

Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard	Reserved
a	Input	PIN_AB12	3A	B3A_N0	2.5 V (default)	
b	Input	PIN_AC12	3A	B3A_N0	2.5 V (default)	
cin	Input	PIN_AF9	3A	B3A_N0	2.5 V (default)	
cout	Output	PIN_V16	4A	B4A_N0	2.5 V (default)	
sum	Output	PIN_W16	4A	B4A_N0	2.5 V (default)	
<<new node>>						
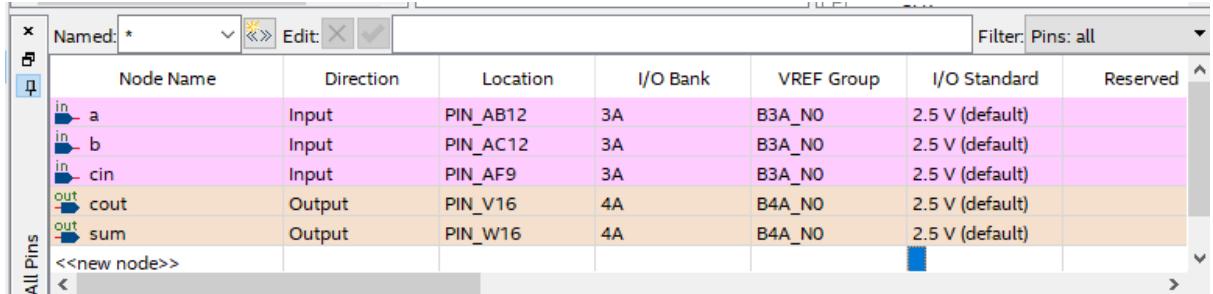
Fig. 1.8 Pin Assignments

- Pin assignment can also be done using the Assignment Editor Window (**Assignments → Assignment Editor**)
- **Compiling the Code** – Click the **Blue Arrow** , located at the top of the main Quartus window, to start the full compilation of your code. You can also go to: **Processing→Start Compilation**.
- Once the steps are complete **Flow Summary** is displayed

Flow Summary

Flow Status	Successful - Wed Dec 02 15:50:11 2020
Quartus Prime Version	18.1.1 Build 646 04/11/2019 SJ Lite Edition
Revision Name	fulladder
Top-level Entity Name	fulladder
Family	Cyclone V
Device	5CSEMA5F31C6
Timing Models	Final
Logic utilization (in ALMs)	2 / 32,070 (< 1 %)
Total registers	0
Total pins	5 / 457 (1 %)
Total virtual pins	0
Total block memory bits	0 / 4,065,280 (0 %)
Total DSP Blocks	0 / 87 (0 %)
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0

Fig. 1.9 Flow summary

- Click on Technology Map Viewer under Fitter (Place and Route) to see the schematic diagram
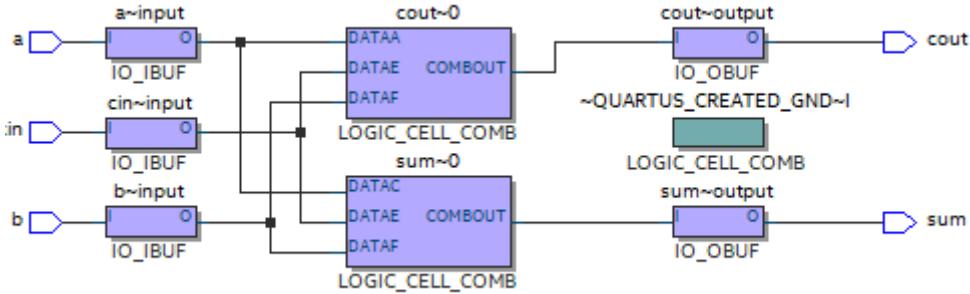

Fig. 1.10 Schematic of the circuit

- Click on Chip Planner to get the complete view of the FPGA
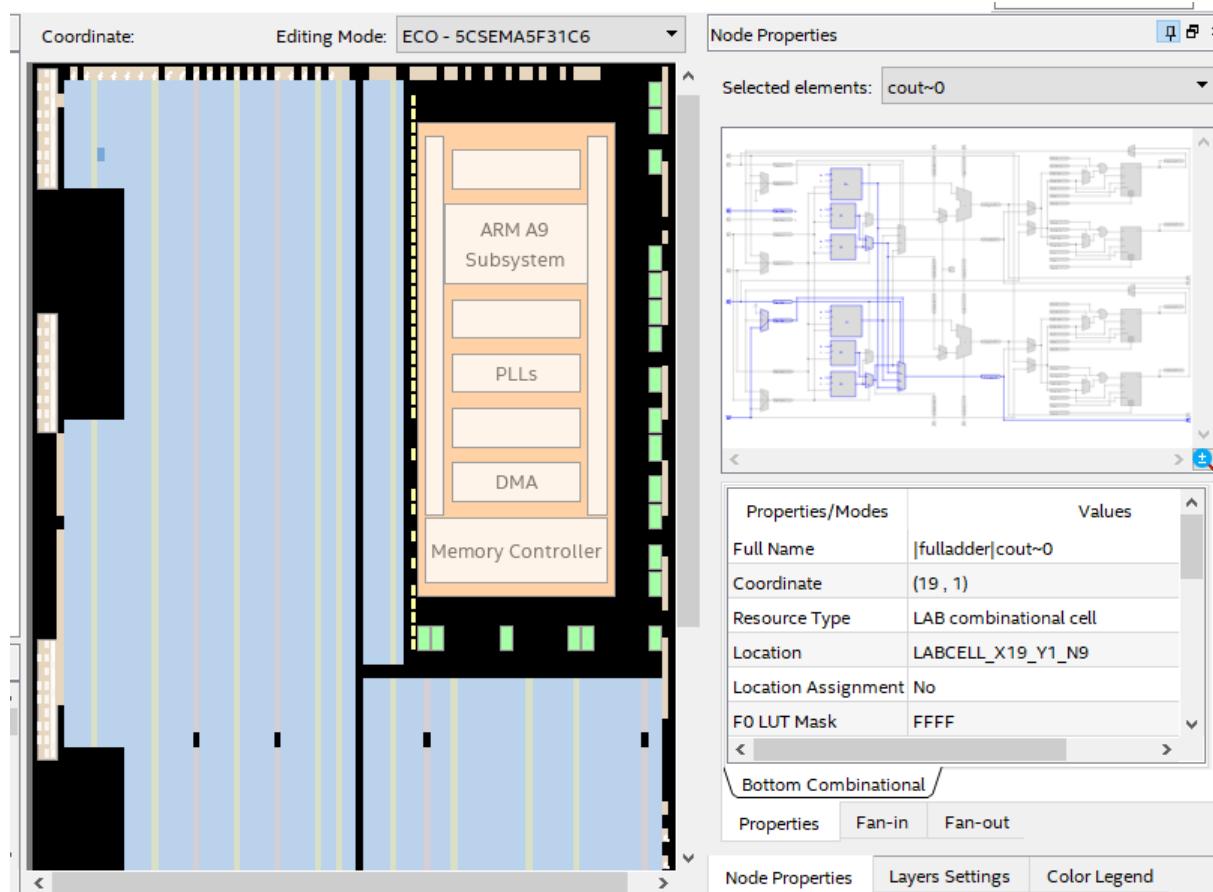
Fig. 1.11 Chip Planner - FPGA layout

3. **Downloading the design to FPGA board and testing (Proceed to next section – Simulation as we will be using the kit only in next lab)**

   The FPGA device must be programmed and configured to implement the designed circuit. The required configuration file is generated (**fulladder.sof**) by the Quartus Prime Compiler's Assembler module. Intel's DE-series board allows the configuration to be done in two different ways, known as JTAG and AS modes. In the JTAG mode, the configuration data is loaded directly into the FPGA device. If the FPGA is configured in this manner, it will retain its configuration as long as the power

remains turned on. The configuration information is lost when the power is turned off. The second possibility is to use the Active Serial (AS) mode. In this case, a configuration device that includes some flash memory is used to store the configuration data. We will be using the JTAG mode for this lab.

- Connect DE1-SOC kit to the USB port of the PC and then switch it on
- Select **Tools** - **Programmer** to reach the window in Fig.1.12
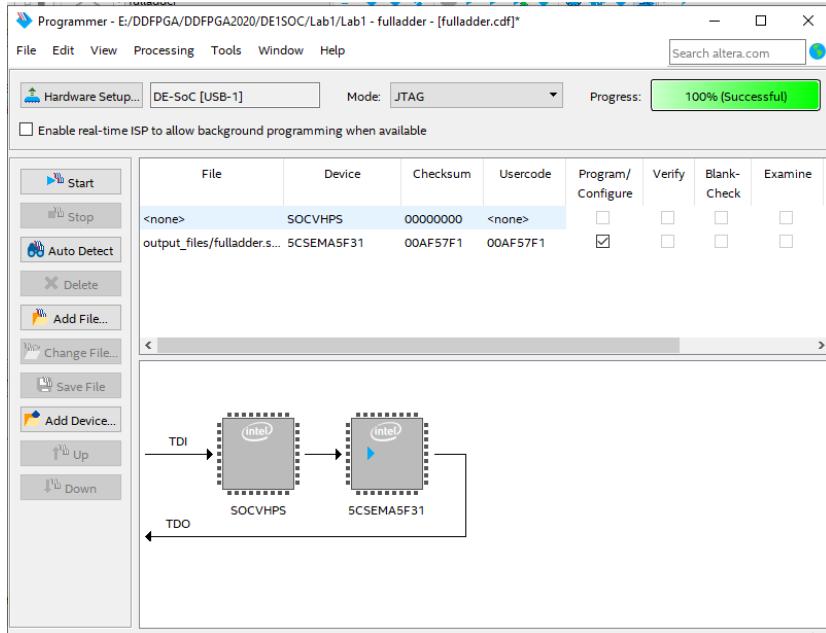

Fig. 1.12 Programmer window

Here it is necessary to specify the programming hardware and the mode that should be used. If not already chosen by default, select **JTAG** in the **Mode** box. Also, if **DE-SoC** is not chosen by default as the programming hardware, then press the **Hardware Setup**... button and select the DE-SoC in the window that pops up.

Observe that the configuration file **fulladder.sof** in directory output files is listed in the window. If the file is not already listed, then click **Add File** and select it. This is a binary file produced by the Compiler's Assembler module, which contains the data needed to configure the FPGA device. The extension .sof stands for SRAM Object File. Ensure the Program/Configure box is checked. This setting is used to select the FPGA in the Cyclone V SoC chip for programming. If the SOCVHPS device is not shown as in Fig. 1.12, click **Add Device - SoC Series V - SOCVHPS** then click **OK**. Ensure that your device order is consistent with Fig. 1.12 by clicking on a device and then clicking **Up** or **Down** .
- Now, press **Start** in the Programmer. An LED on the board will light up while the FPGA device is being programmed. If you see an error reported by Quartus Prime software indicating that programming failed, then check to ensure that the board is properly powered on.
- Verify the functionality by flipping switches and observing the output on the LEDs
- When satisfied, power OFF the board.
- Close the Programmer

4. Simulation

In this section, we will introduce the concept of test bench and show how to verify the function of our full adder using Questa Intel FPGA Starter Edition for Functional simulation.
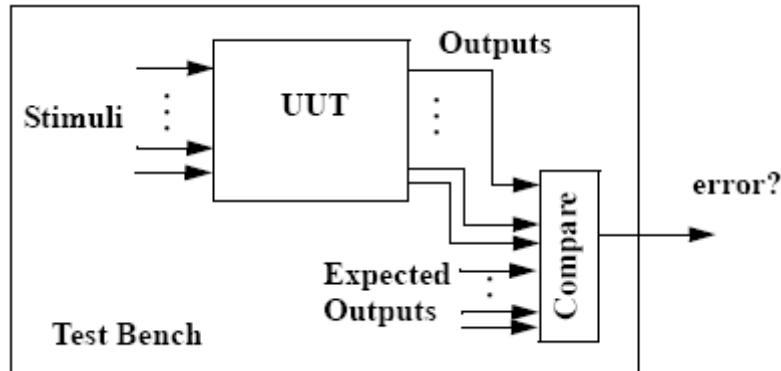

Fig. 1.13 Conceptual diagram of test bench

A test bench is an entity (usually a HDL program) which is used to verify the correctness of a design. The design to be verified is called Unit Under Test (UUT). The test bench supplies stimuli to the design, observes the outputs of the design, and compares the observed outputs with the expected values. If any mismatch happens, the test bench issues certain messages signifying that there are errors in the design. Fig. 1.13 shows the concept of test bench.

Create Testbench
- Go the File dropdown menu and select New.
- Click on Verilog HDL File and then Ok.
- Create a simple module in your Verilog HDL file by typing in the following code.

```
`timescale 1ns / 1ps
module fulladder_tb;
 reg pa, pb, pcin;
 wire psum, pcout;
// instantiate the fulladder module
 fulladder uut(pa, pb, pcin, psum, pcout); // positional association
 initial
   begin: blk_only_once
    reg [3:0] pa1;

    for (pa1=0; pa1 < 8; pa1 = pa1+1)
     begin
     {pa, pb, pcin} <= pa1;
     #50;
     end
   end
endmodule
```

- In **File**, Click on **Save As**, name the file as **fulladder_tb** (ensuring case-sensitivity), and save your Verilog file
- In **Assignments – Settings – EDA Tool Settings – Simulation- Nativelink Setting**s – Select **Compile Testbench** and Click on **Testbench**. Select **fulladder_tb.v** as the testbench as shown in Fig. 1.14
- Now the settings will look as in Fig. 1.15. Click **OK**
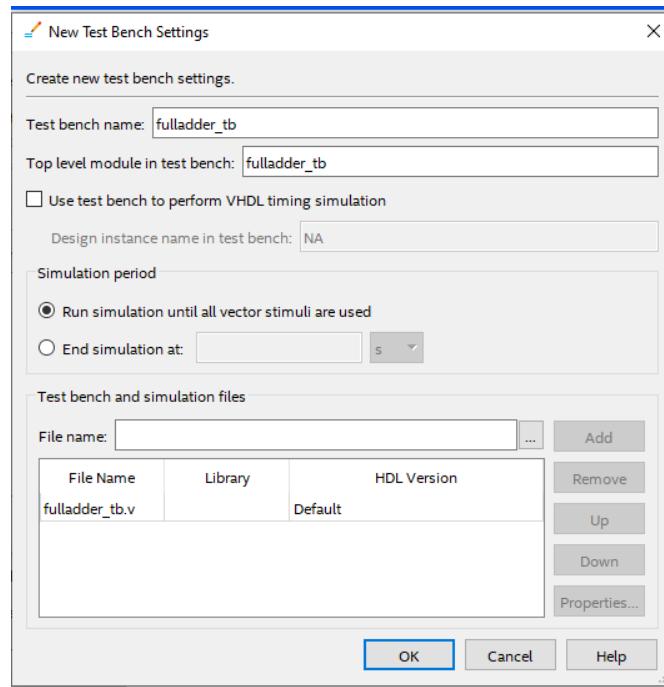
Fig. 1.14 Select Testbench file
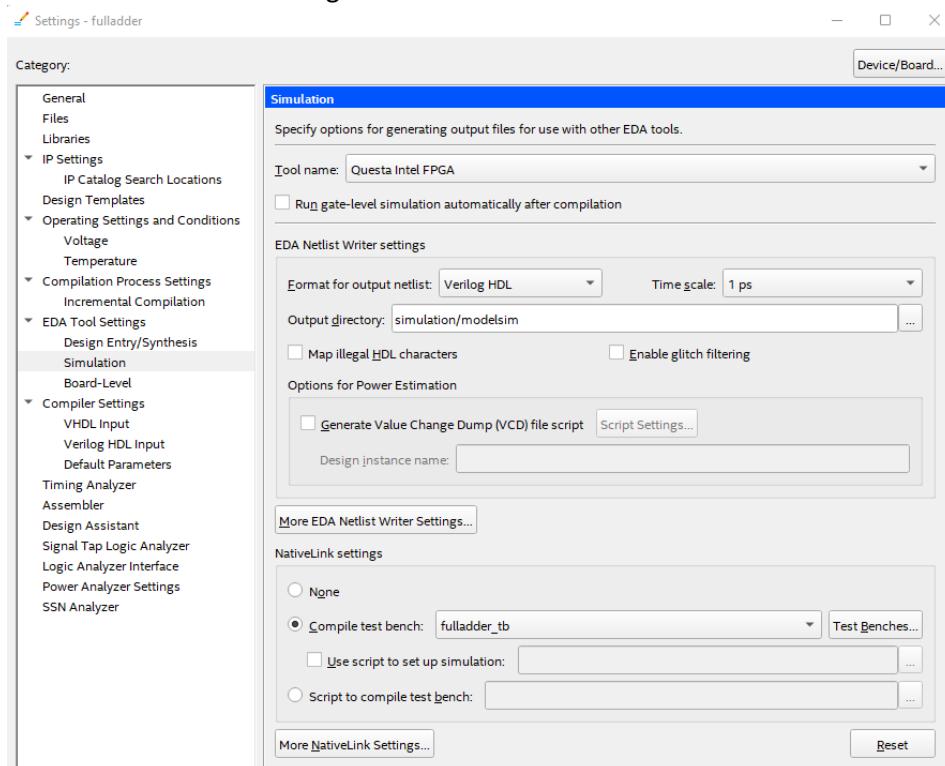
Fig. 1.15 Settings – simulation

- Click on **Tools – Run Simulation Tool – RTL simulation**
- The testbench and source files will be compiled and the Questasim simulator will run (assuming no errors). You will see a simulator output similar to Fig. 1.16.
- Click on the zoom-full button to see the entire waveform. Check the outputs.
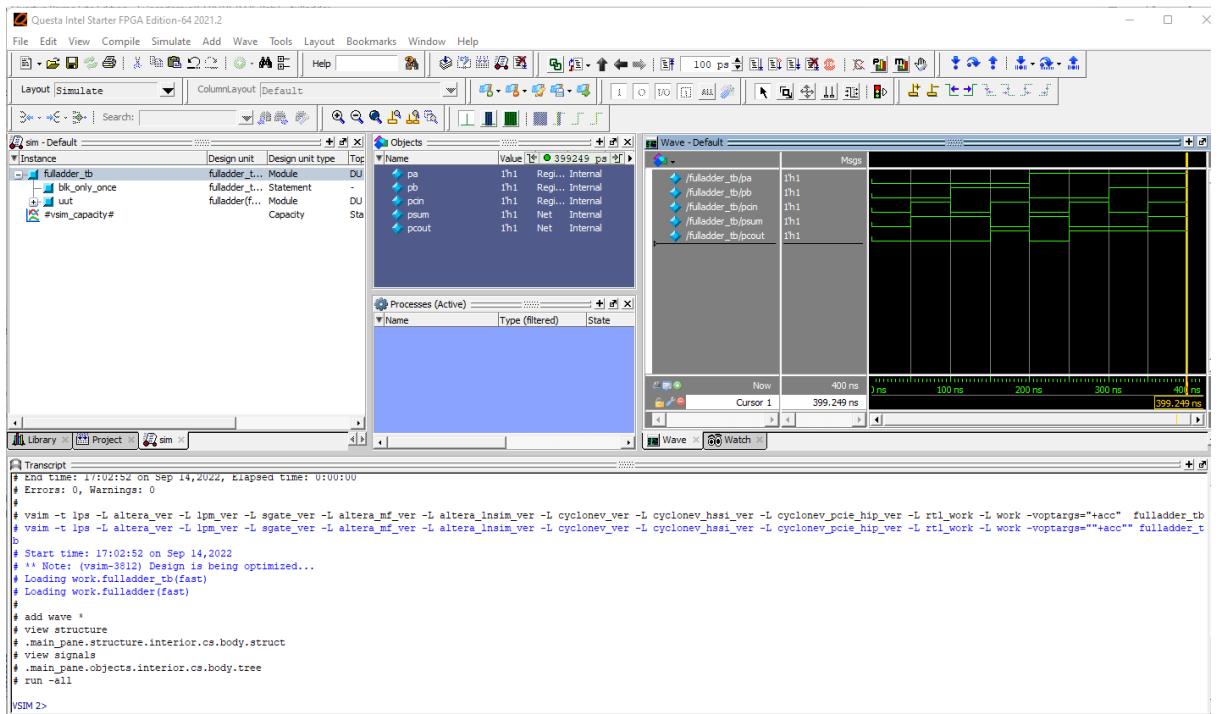
Fig. 1.16 Simulator output

Exercise

1. Learn the features of the simulator
2. Design a 4 bit adder as a structural model by instantiating the full adder module 4 times. Simulate and test the code.