



Faculty	: Industrial Technology	Meeting To	: 1
Department / Study	: Electrical Engineering	Module To	: 1
Program Course Code	: STE422	Number of pages	: 23
Course Name	: Digital System Practice	Be in effect	: 2019

UNIT 1

INTRODUCTION TO SIMULATION SOFTWARE AND PROGRAMMING LANGUAGES

VHDL

This unit introduces the basic features of Quartus II software which is used to design, simulate and implement digital circuit using VHSIC Hardware Description Language (VHDL); VHSIC = Very High Speed Integrated Circuit. Design steps digital circuit includes:

1. Create a project.
2. Coding the circuit using VHDL.
3. Synthesize the circuit that has been created
4. Perform the fitting process on the Altera FPGA
5. Connect the input and output ports of the circuit to the FPGA pins.
6. Simulate the circuit that has been created/designed.
7. Program the FPGA chip on the board

In this unit 1, Quartus II software will be introduced up to the simulation stage. with VHDL language.

I. Quarter II

Any digital circuit designed using software Quartus II is called a project. Quartus II only works on one project at a time. the same and save all project information in one directory (folder). The first step in designing a circuit is to create a directory for save all design files.

Run the Quartus II software, then the window display will be as shown in the following figure. image 1 will appear.

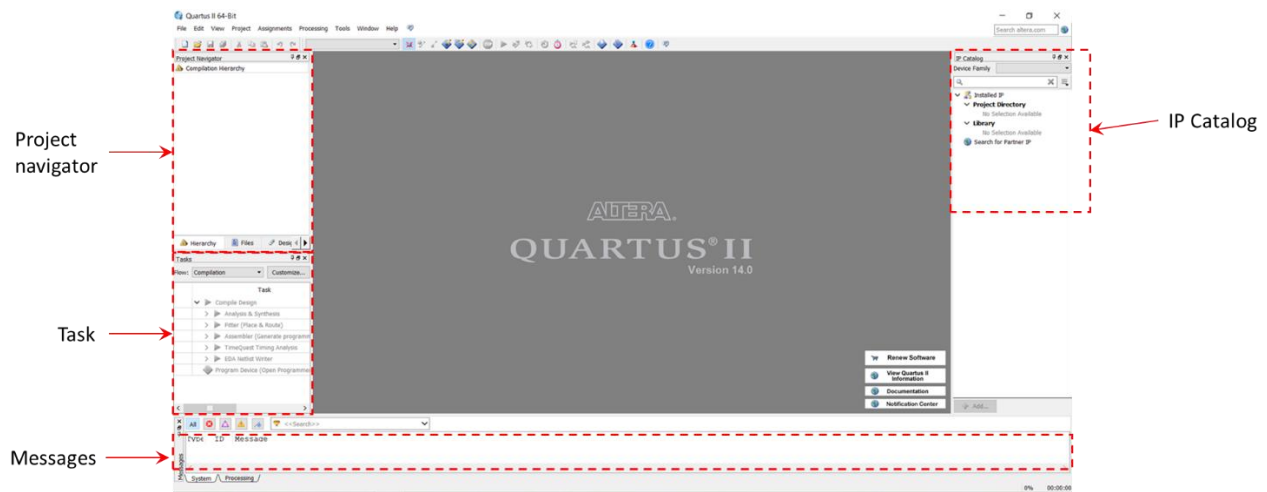


Figure 1. Quartus II version 14.0 main view

This window contains several other windows that are used to access all the features of Quartus II software, users can choose them by using the mouse. Some parts of the main display as shown in Figure 1 consists of project navigator, tasks, messages and IP catalog. Commands- Quartus II software commands can be accessed using the menu bar below the title bar. In general, the left mouse button is used to select something. To start a new project, the steps to take are:

1. Select File > New Project Wizard, a window like the one in Figure 2 appears. asks for the name and directory of the project.

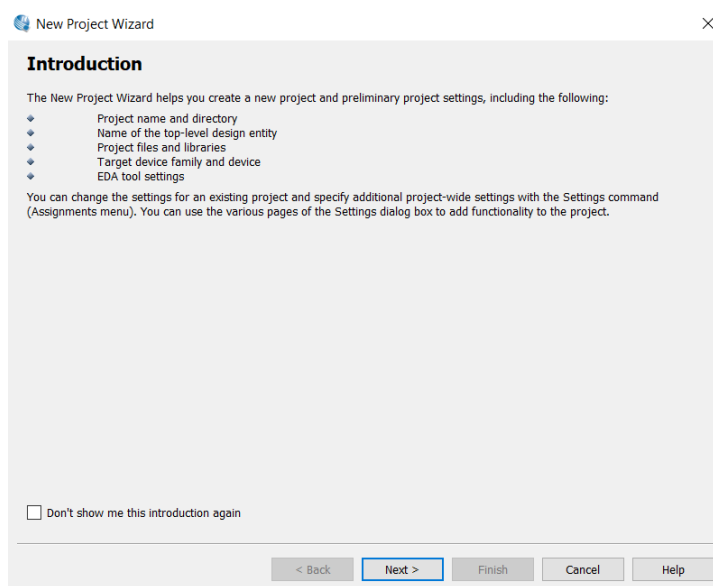


Figure 2. Initial view of the new project wizard

2. Click next, so that a display appears as in image 3.

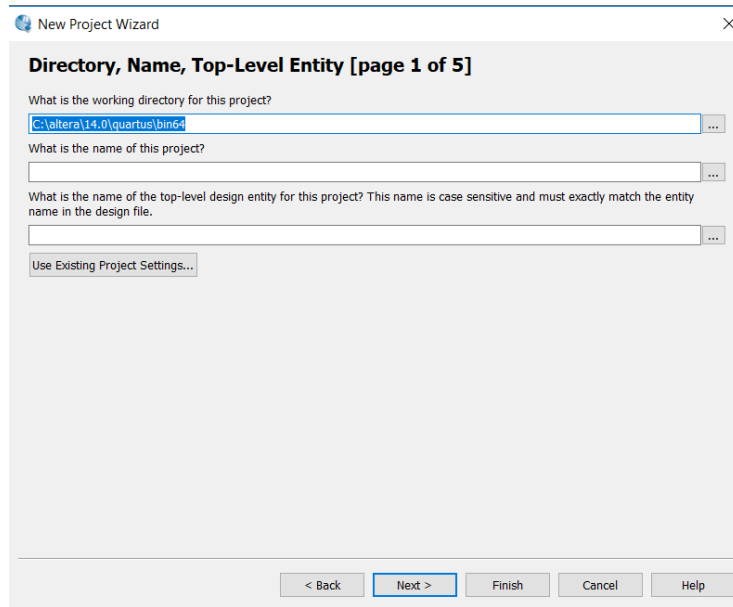



Figure 2. The display asks for the directory and project name.

3. Fill in the project directory location in the desired location. You can also do this

Clicking knob  next to his right. Example in **D:\data_altera\latihan1**. Fill in the project name with the name **lat1** and name The top-level design is the same as the project name, namely **lat1**. Appearance after filling as shown in figure 3.

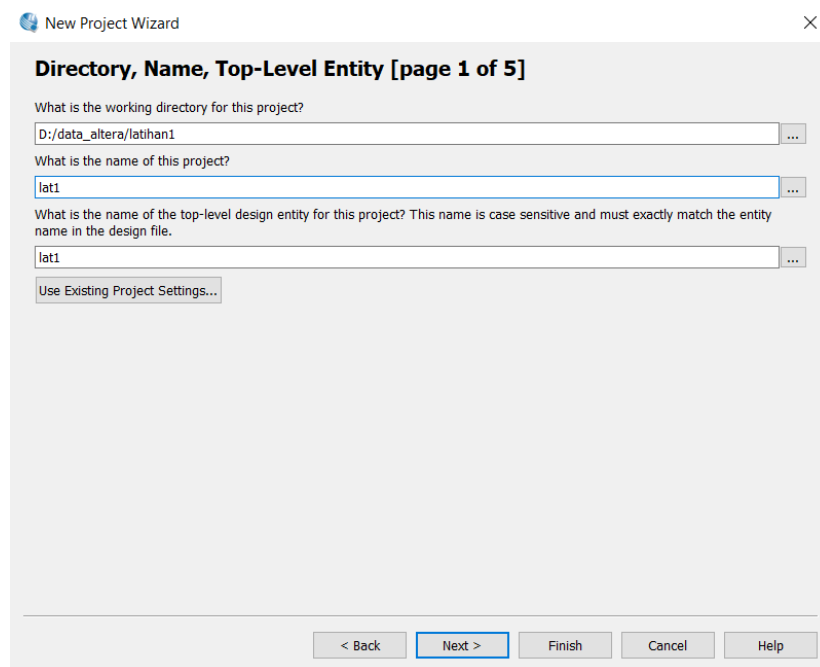


Figure 3. Display of directory contents and project name

4. Click the next button so that the display appears in image 4. If we already have a file that will be involved in this project, then add the file to the box. However, because we don't have any files yet, has been created then click next so that the family & device settings appear. Select and fill in according to the device to be used. In this activity the device to be used is Cyclone IVseri 4 EP4CE6E22C8N. So fill it in as in Figure 5.

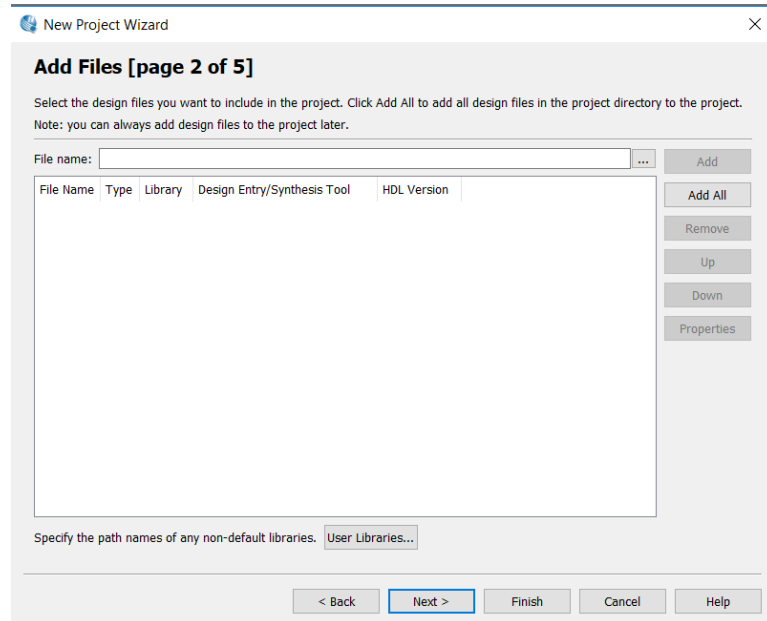


Figure 4. Display after filling in the directory and project name.

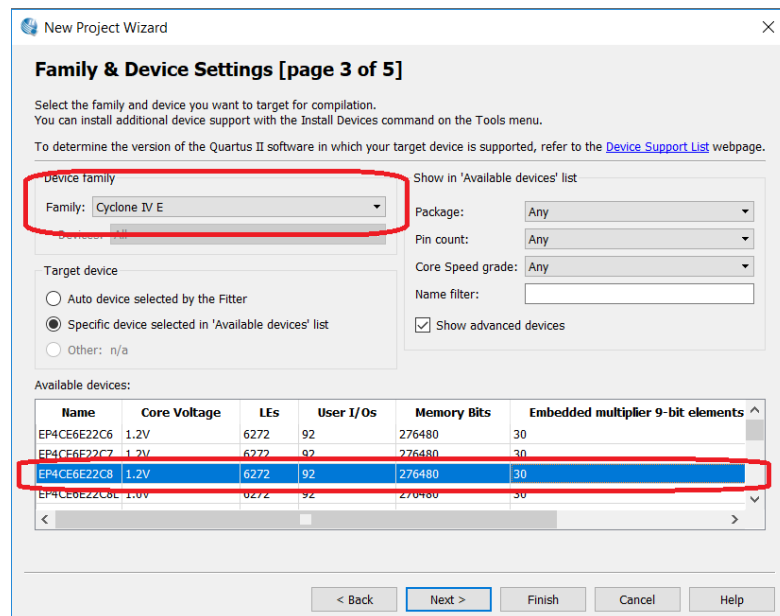


Figure 5. Fill in the family and device settings section

5. Click next so that the EDA tool setting display appears as in Figure 6. If it is in accordance with Figure 6, click next so that the summary display is obtained as in Figure 7. Then click finish.

Tool Type	Tool Name	Format(s)	Run Tool Automatically
Design Entry/Synt...	<None>	<None>	<input type="checkbox"/> Run this tool automatically to synthesize the current design
Simulation	ModelSim-Altera	VHDL	<input type="checkbox"/> Run gate-level simulation automatically after compilation
Formal Verification	<None>		
Board-Level	Timing	<None>	
	Symbol	<None>	
	Signal Integrity	<None>	
	Boundary Scan	<None>	

Figure 6. EDA tool setting

When you click Finish, the project will be created with the following settings:

Project directory:	D:/data_altera/latihan1
Project name:	lat1
Top-level design entity:	lat1
Number of files added:	0
Number of user libraries added:	0
Device assignments:	
Family name:	Cyclone IV E
Device:	EP4CE6E22C8
EDA tools:	
Design entry/synthesis:	<None> (<None>)
Simulation:	ModelSim-Altera (VHDL)
Timing analysis:	()
Operating conditions:	
VCCINT voltage:	1.2V
Junction temperature range:	0-85 °C

Figure 7. The final /summary section of the New project wizard.

6. After the new project creation process, the display on the main screen will change slightly as shown in Figure 8.

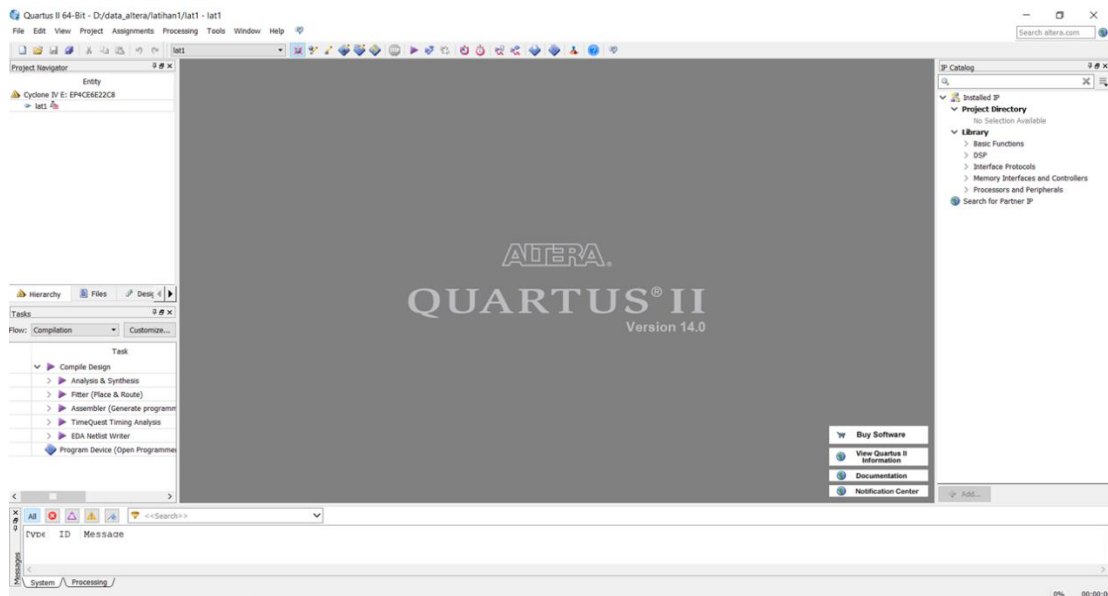


Figure 8. Main layer view after the new project creation process.

The next step is to create a circuit design with using VHDL language. The steps to make it are explained in the next chapter.

II. VHDL

VHDL (*VHSIC Hardware Description Language*); VHSIC (*Very High Speed Integrated Circuit*) is a type of HDL language (*hardware description language*) which is used to describe various circuit functions. digital such as logic gates, flip-flops, and so on. VHDL can also used as a programming language for circuit simulation of components digital components. HDL is used by hardware designers (*hardware*) For write the properties, signals and functionality of a description based on *hardware* from a series.

Digital circuit design with VHDL is done at a higher level of abstraction. higher. This methodology is combined with synthesis tools to translate and optimize the description of a design. The synthesis engine is used to map the design to physical hardware such as *application specific integrated circuit* (ASIC) or *field programmable gate array* (FPGA).

The basic structure of a VHDL program is LIBRARY, ENTITY and ARCHITECTURE with the condition that it does not take into account upper or lower case letters (*not case sensitive*). LIBRARY is a container, which holds files.

which defines an entity, architecture, or package. The format for writing LIBRARY is:

```
LIBRARYlibrary_name;  
USElibrary_name.package_name.package_parts;
```

Example:

```
IEEE LIBRARY;  
USE IEEE.STD_LOGIC_1164.ALL; USE  
ieee.std_logic_arith.all;
```

ENTITY is a declaration of the entity name and external interface to the entity.

design made. Writing format:

```
ENTITYentity_nameIS  
PORT (  
    port_name :signal_modesignal_type;  
    port_name :signal_modesignal_type;  
    ... );  
ENDentity_name;
```

Example:

```
ENTITY example1 IS  
PORT (  
    A:in std_logic;  
    B: in std_logic;  
    Z1: out std_logic;  
    Z2: out std_logic  
);  
END example1;
```

The ARCHITECTURE section contains an internal description of the design entity which can be *behavior*, structure, or a mixture of both. The writing format:

```
ARCHITECTURE entity_bag_name OF entity_name IS BEGIN
```

```
    Expression of behavior/structure;
```

```
    Expression of behavior/structure; ...
```

```
END entity_bag_name;
```

Example:

```
circuit ARCHITECTURE1 OF example1 IS BEGIN
```

```
    Z1 <= A;
```

```
    Z2 <= B;
```

```
END circuit1;
```

Let's practice creating designs with VHDL in Quartus II.

1. Click File -> New, then a display will appear as in figure 9. Select or click VHDL file then press OK, so the main layer will appear turns white. This is the screen where we will write the VHDL code.

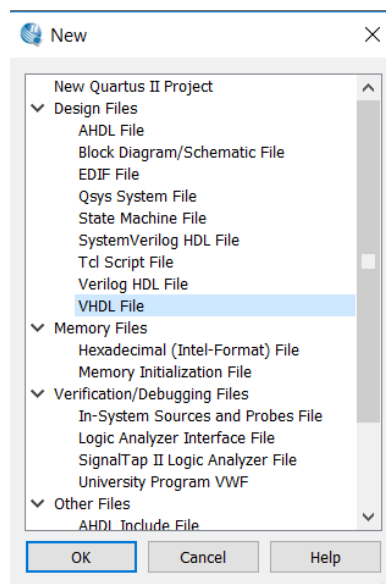


Figure 9. Window view before creating a VHDL file.

2. Write/type the program as follows:

```
IEEE LIBRARY;  
USE IEEE.STD_LOGIC_1164.ALL;  
ENTITY lat1 IS  
PORT (  
    A: in std_logic;  
    B: in std_logic;  
    Z1: out std_logic;  
    Z2: out std_logic  
);  
END lat1;  
ARCHITECTURE by_pass OF lat1 IS  
BEGIN  
    Z1 <= B;  
    Z2 <= A;  
END by_pass;
```

The purpose of this coding is that we create an entity that named lat1. Within this entity there are input ports A and B, as well as output ports Z1 and Z2. Inside the lat1 entity there is a design named by_pass. The circuit in by_pass is just a direct connection between the ports input B to the output port Z1, and input port A to the output port Z2. Picture 10 is an illustration of the lat1 entity.

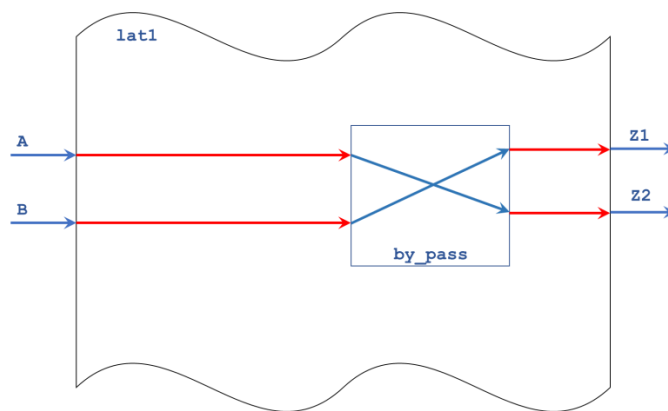

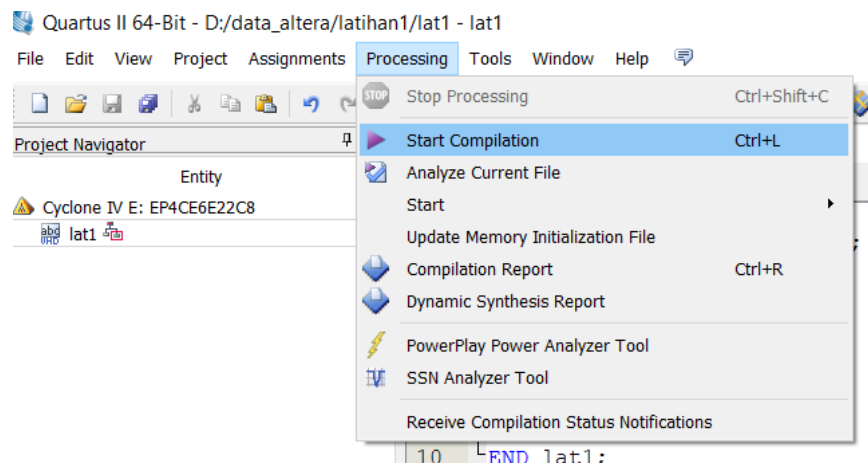
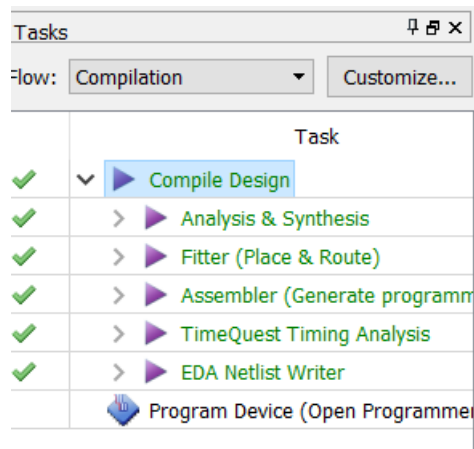


Figure 10. Illustration of the layout design lat1

3. Save the VHDL file by clicking file -> save. Save the file name with the name lat1.vhd. The storage directory is the same as the directory that was created previously, namely latihan1.
4. The next process is to compile by clicking Processing -> start compilation as in figure 11(a), or by double clicking compile design in the tasks section as shown in Figure 11(b), or you can also click on the symbol .



(a) Start compilation from processing ->start compilation



(b) Start the compilation by double clicking on tasks.

Figure 11. Command to start compiling a project

5. After the compilation process is complete without any problems, a compilation results report will appear as shown in Figure 12. From this report we can also see how many resources are required. to implement the design we made. By clicking other parts in the table of contents, we can also see reports on other aspects.

Table of Contents	Flow Summary																																
<ul style="list-style-type: none"> Flow Summary Flow Settings Flow Non-Default Global Settings Flow Elapsed Time Flow OS Summary Flow Log > Analysis & Synthesis > Fitter > Assembler > TimeQuest Timing Analyzer > EDA Netlist Writer Flow Messages Flow Suppressed Messages 	<table> <tr> <td>Flow Status</td><td>Successful - Mon Jan 28 14:39:48 2019</td></tr> <tr> <td>Quartus II 64-Bit Version</td><td>14.0.0 Build 200 06/17/2014 SJ Web Edition</td></tr> <tr> <td>Revision Name</td><td>lat1</td></tr> <tr> <td>Top-level Entity Name</td><td>lat1</td></tr> <tr> <td>Family</td><td>Cyclone IV E</td></tr> <tr> <td>Device</td><td>EP4CE6E22C8</td></tr> <tr> <td>Timing Models</td><td>Final</td></tr> <tr> <td>Total logic elements</td><td>0 / 6,272 (0 %)</td></tr> <tr> <td> Total combinational functions</td><td>0 / 6,272 (0 %)</td></tr> <tr> <td> Dedicated logic registers</td><td>0 / 6,272 (0 %)</td></tr> <tr> <td>Total registers</td><td>0</td></tr> <tr> <td>Total pins</td><td>4 / 92 (4 %)</td></tr> <tr> <td>Total virtual pins</td><td>0</td></tr> <tr> <td>Total memory bits</td><td>0 / 276,480 (0 %)</td></tr> <tr> <td>Embedded Multiplier 9-bit elements</td><td>0 / 30 (0 %)</td></tr> <tr> <td>Total PLLs</td><td>0 / 2 (0 %)</td></tr> </table>	Flow Status	Successful - Mon Jan 28 14:39:48 2019	Quartus II 64-Bit Version	14.0.0 Build 200 06/17/2014 SJ Web Edition	Revision Name	lat1	Top-level Entity Name	lat1	Family	Cyclone IV E	Device	EP4CE6E22C8	Timing Models	Final	Total logic elements	0 / 6,272 (0 %)	Total combinational functions	0 / 6,272 (0 %)	Dedicated logic registers	0 / 6,272 (0 %)	Total registers	0	Total pins	4 / 92 (4 %)	Total virtual pins	0	Total memory bits	0 / 276,480 (0 %)	Embedded Multiplier 9-bit elements	0 / 30 (0 %)	Total PLLs	0 / 2 (0 %)
Flow Status	Successful - Mon Jan 28 14:39:48 2019																																
Quartus II 64-Bit Version	14.0.0 Build 200 06/17/2014 SJ Web Edition																																
Revision Name	lat1																																
Top-level Entity Name	lat1																																
Family	Cyclone IV E																																
Device	EP4CE6E22C8																																
Timing Models	Final																																
Total logic elements	0 / 6,272 (0 %)																																
Total combinational functions	0 / 6,272 (0 %)																																
Dedicated logic registers	0 / 6,272 (0 %)																																
Total registers	0																																
Total pins	4 / 92 (4 %)																																
Total virtual pins	0																																
Total memory bits	0 / 276,480 (0 %)																																
Embedded Multiplier 9-bit elements	0 / 30 (0 %)																																
Total PLLs	0 / 2 (0 %)																																

Figure 12. Display of the compilation results after completion.

III. Functional Simulation

Functional simulations are carried out to determine the functionality of the design in general. ideal. To perform this simulation you must have installed ModelSIM minimum starter pack (no license required). Steps taken to perform functional simulation is as follows:

1. Click tools -> run simulation tool -> RTL simulation as shown in Figure 13. After clicking, Quartus II will call ModelSIM so that The screen display appears as in Figure 14.

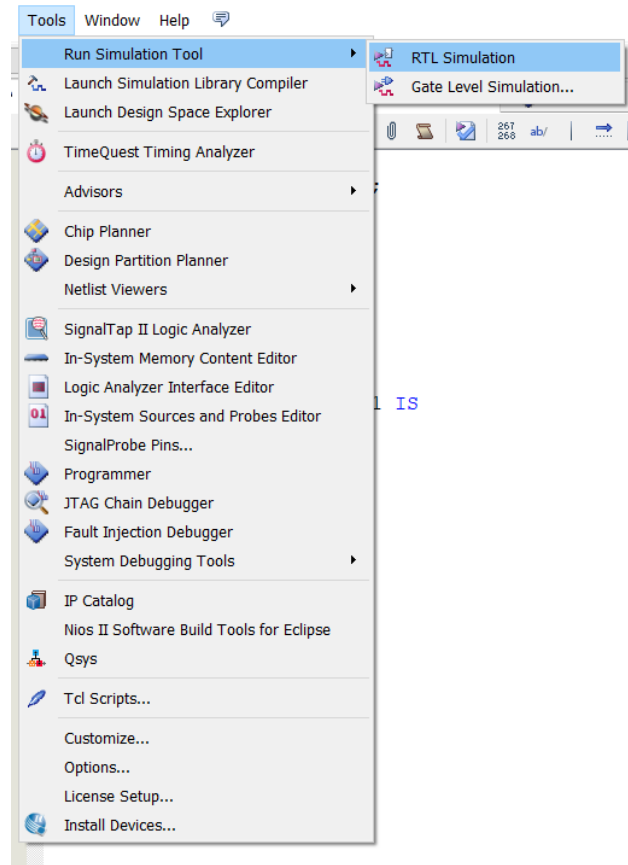


Figure 13. Starting a functional simulation

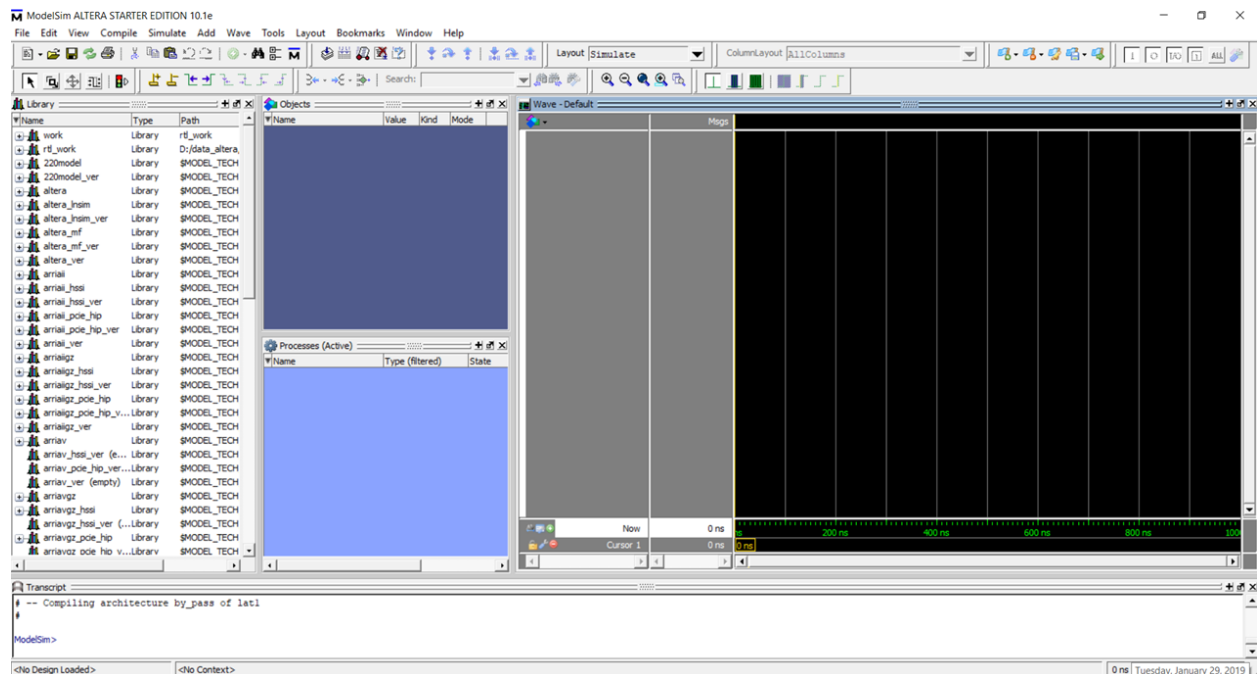


Figure 14. Initial view of the simulation in ModelSIM

2. Click the (+) sign in the Library so that lat1 appears as shown in Figure 15.

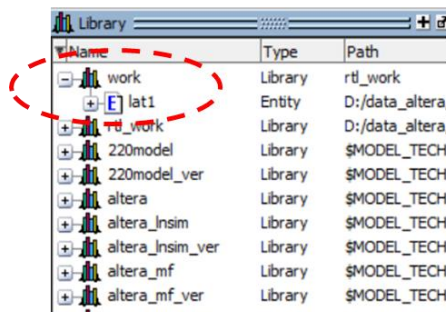


Figure 15. Displaying the design name in the work project

3. Right click on lat1 then click simulate (see figure 16), so

The screen display changes as shown in Figure 17.

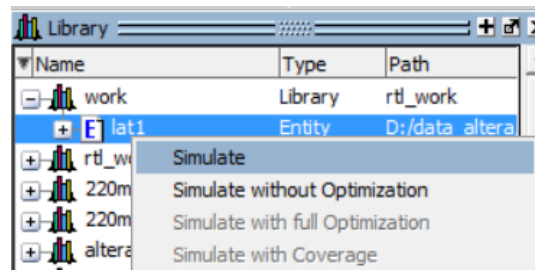


Figure 16. Command to perform simulation

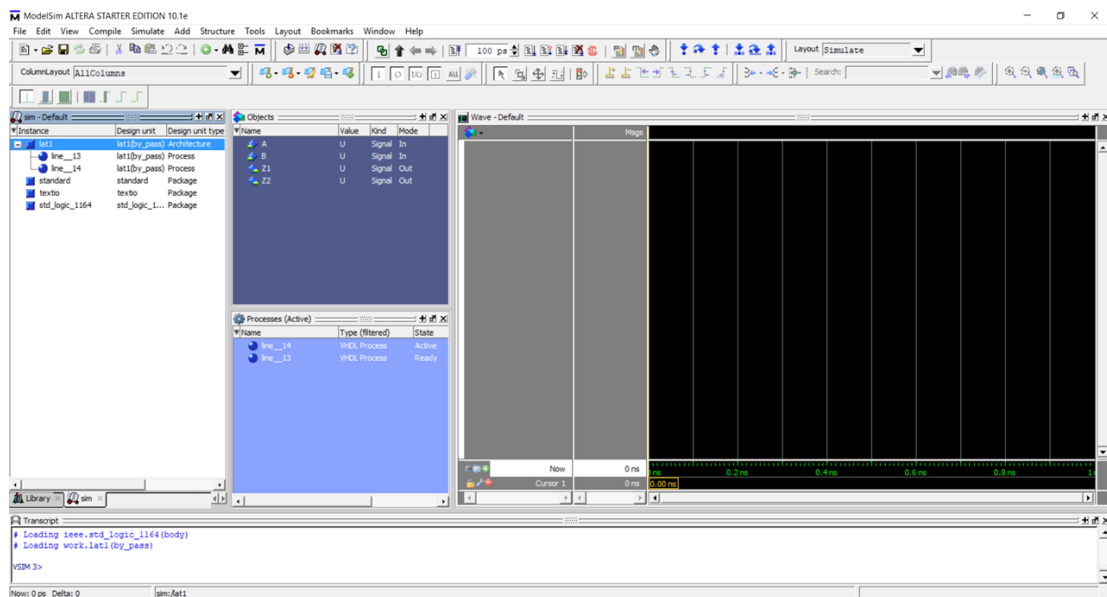


Figure 17. Screen view after simulation command

4. On the object the name of the port design created in Quartus II will appear. namely A, B, Z1 and Z2. To display the ports to the simulation viewer (wave)

Right click each port then click add wave (see figure 18). Do this for all ports in the objects. Once completed, all ports will appear in the simulation viewer (wave) as shown in figure 19.

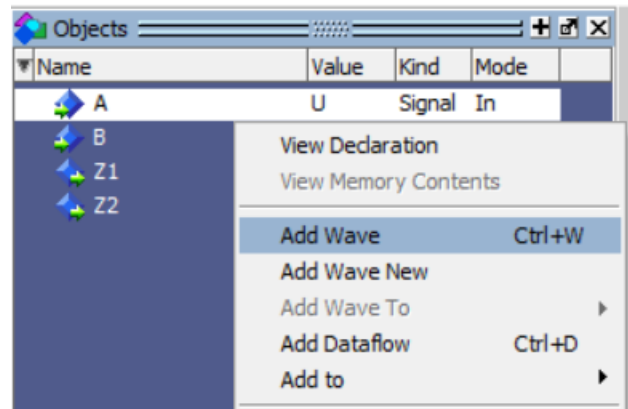


Figure 18. Inserting a port into the simulation viewer

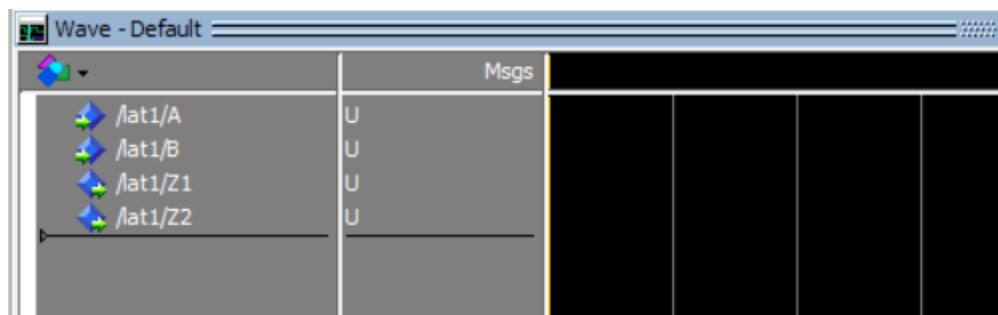






Figure 19. Display in Wave

5. The next step is to run the simulation with F9 or press run

which is symbolized . By clicking on the symbol it means we are running simulation during the period listed next to it. For  so

The simulation will run for 100ps. To continue please click again

symbol . After you click the run symbol or  then a signal appears waves on the screen as shown in figure 20. However, the displayed is a red line because we have not set the port input value A and B.

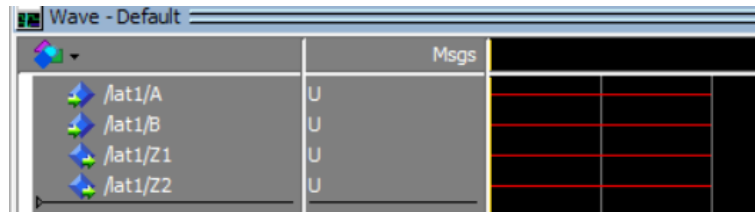


Figure 20. Simulation signal display

6. To assign values to ports A and B, right click on port A then click Force (see image 21), then the Force Selected Signal will appear as shown in image 22. Replace the value 'U' with the value '1', then click OK.

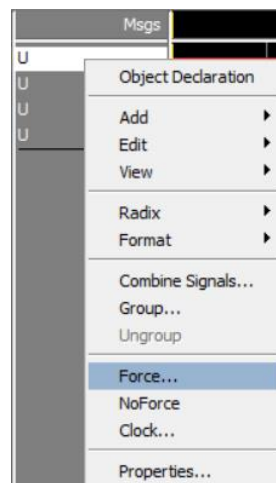


Figure 21. Command to change the value on the input port

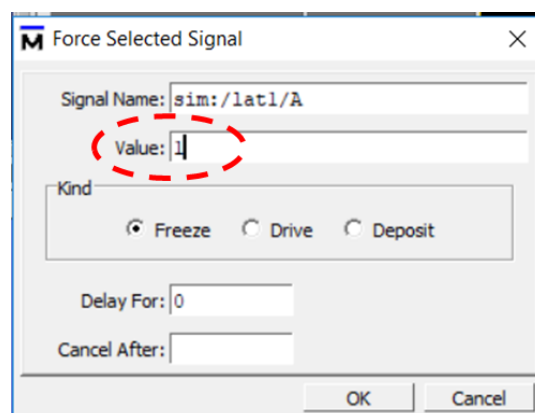


Figure 22. Changing the port value

7. Next, run the simulation again in step no. 5, then the resulting signal
The simulation will be different from the previous one where input port A will be

changed to '1' and the output port Z2 also follows to '1'. Repeat step no.6 for input port B changed to '0', then the simulation results are obtained as in figure 23.

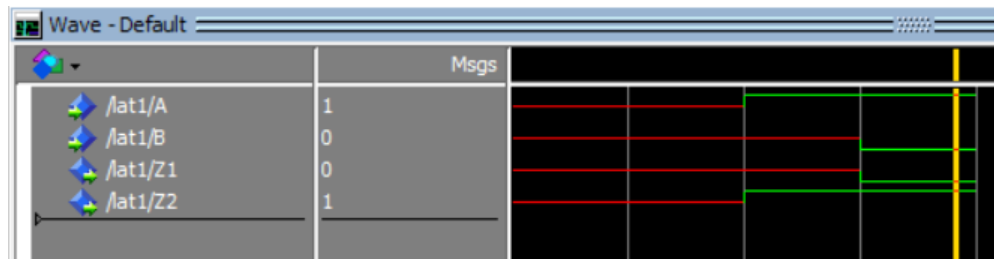


Figure 23. Simulation result signal

IV. Download Program to FPGA

After you are sure that all the coding and compilation results have no problems and the results are... simulation gives results as desired, the next process is to download the program to the FPGA. The steps are as follows are as follows:

1. Do pin assignment

Click Assignments -> Pin Planner (see image 24) so that it appears

The pin planner display looks like in figure 25. Fill in the location as shown in figure 26.

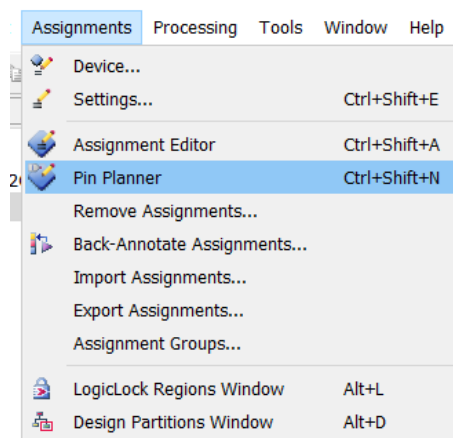


Figure 24. The process will perform pin assignment.

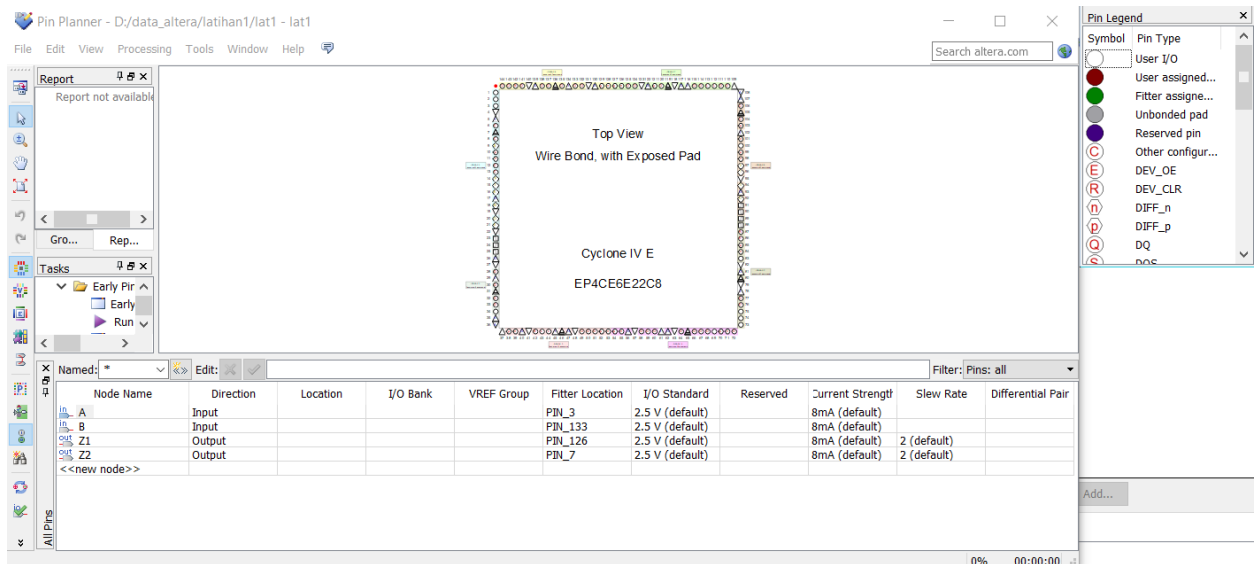


Figure 25. Pin planner view

Named: *		Edit:		
Node Name	Direction	Location	I/O Bank	
in A	Input	PIN_91	6	E
in B	Input	PIN_90	6	E
out Z1	Output	PIN_135	8	E
out Z2	Output	PIN_136	8	E
<<new node>>				

Figure 26. Filling in the pin location

2. Next, click processing -> start I/O assignment analysis

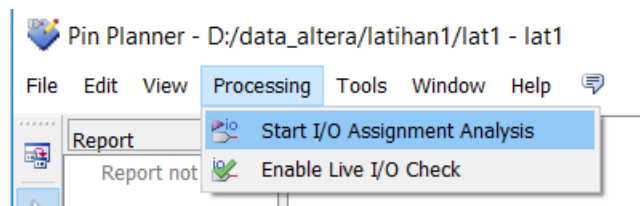


Figure 27. Assignment Analysis

3. Once you are sure there are no problems, close the pin planner display window. Process

Next is the download stage to FPGA. The first step is to click Tools

-> Programmer (see figure 28), so that the window display appears.

Programmer as in figure 29. Click hardware setup and on

hardware setting select USB-blaster on currently selected hardware (see image 30), then click close. If the USB-blaster option is not available

means you need to install the USB-blaster driver available in the ...
\\quartus\drivers folder.

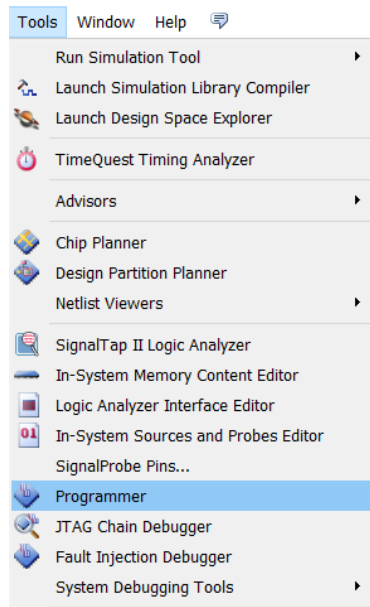


Figure 28. Steps for downloading to FPGA

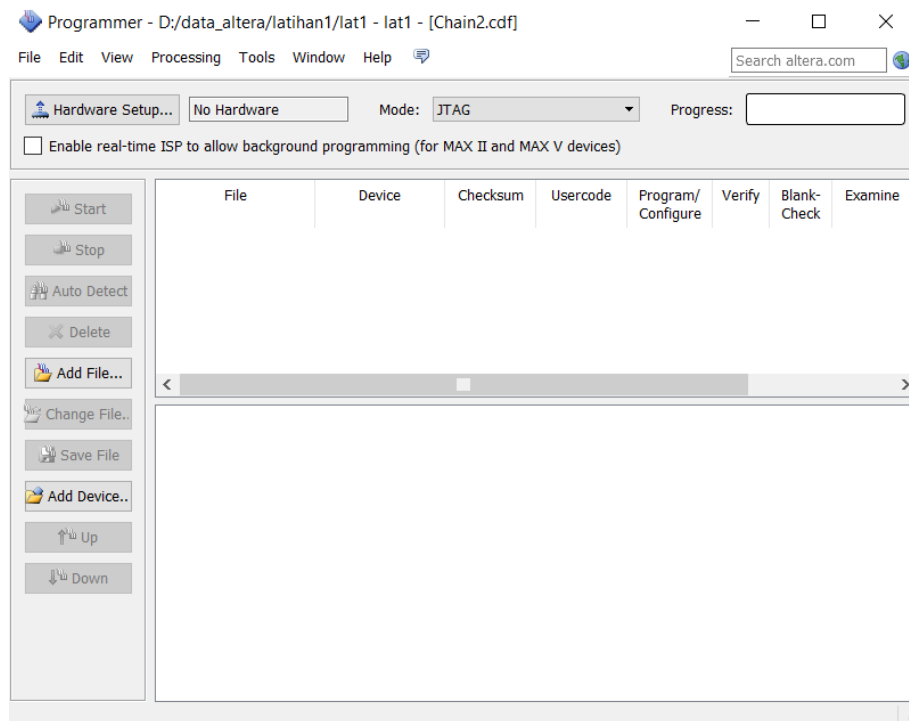


Figure 29. Programmer window display

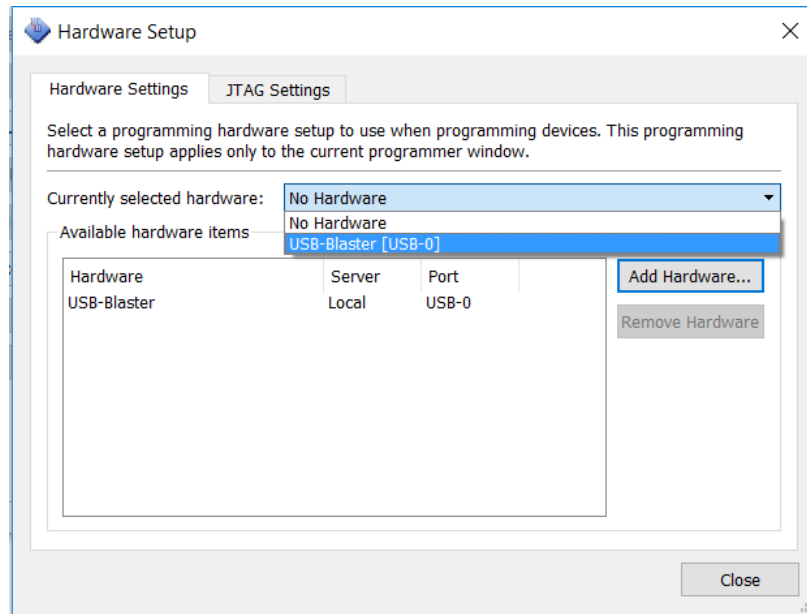


Figure 30. USB-blaster selection on currently selected hardware

4. On the programmer window screen, click Add file, navigate to the "output file" folder. which is in the project for example in ...\\latihan1\\ output_files. In the folder There is a file with the extension ".sof", select the file then click open. The display in the programmer window becomes as shown in the image 31.

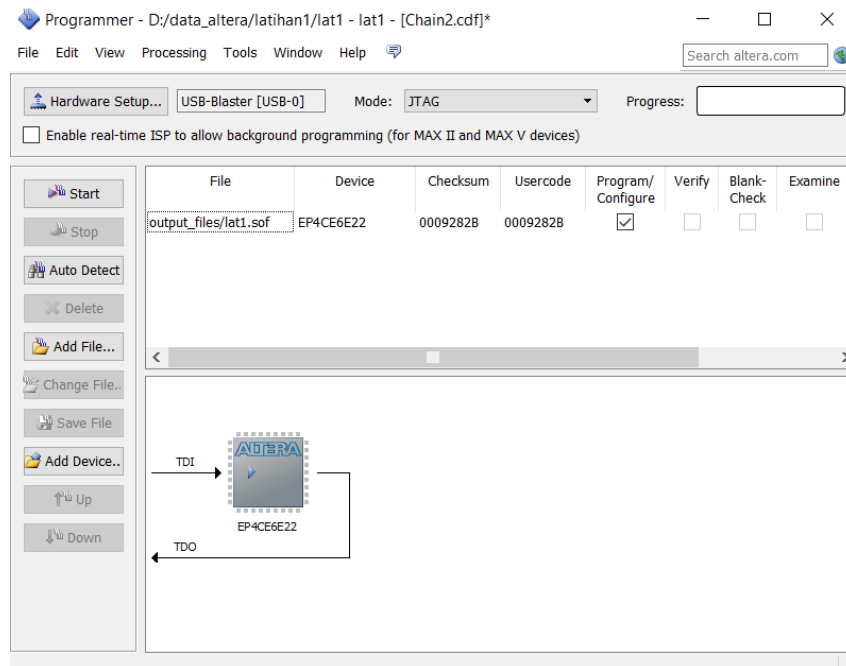


Figure 31. Programmer window display after adding file

5. The next step is to start the process of downloading the “.sof” file to the FPGA by clicking the start button. Wait until the download process is complete which is marked with the success notation in the progress (see figure 32).

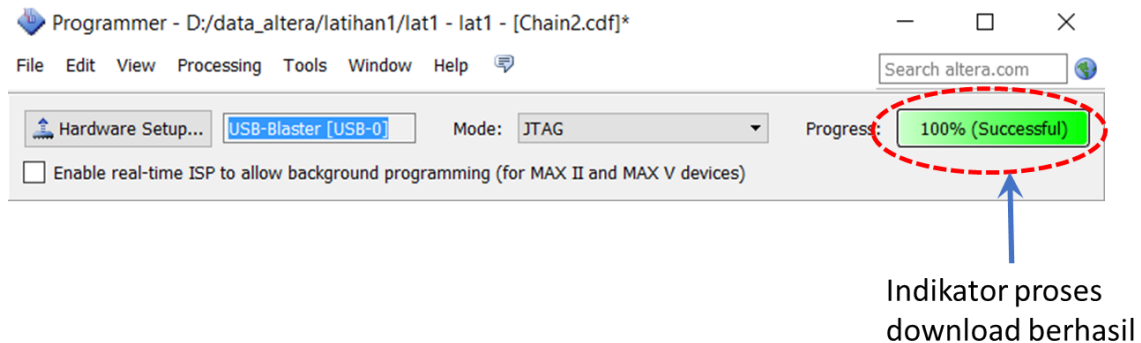


Figure 32. Indication that the download process is complete

Next, you can do hardware verification with provide input to the FPGA, namely at KEY1 and KEY2, where KEY1 connected to port A and KEY2 is connected to port B. While port The output of Z1 is connected to D1 and Z2 is connected to Z1 like so shown in figure 33. The series of push buttons KEY1 to KEY5 and LED are shown in figure 34. From the circuit It is known that when the push button is pressed, the voltage across the push button will be connected to ground or logic '0', while the LED will lights up when the LED tip is connected to ground or given a logic '0'. Because KEY1 is connected to D2 and KEY2 is connected to D1 then when KEY1 is pressed then D2 will light up while when KEY2 is pressed then D1 will light up. Board details can be seen in figure 35. Meanwhile, the complete table of names and PIN numbers is in the attachment.

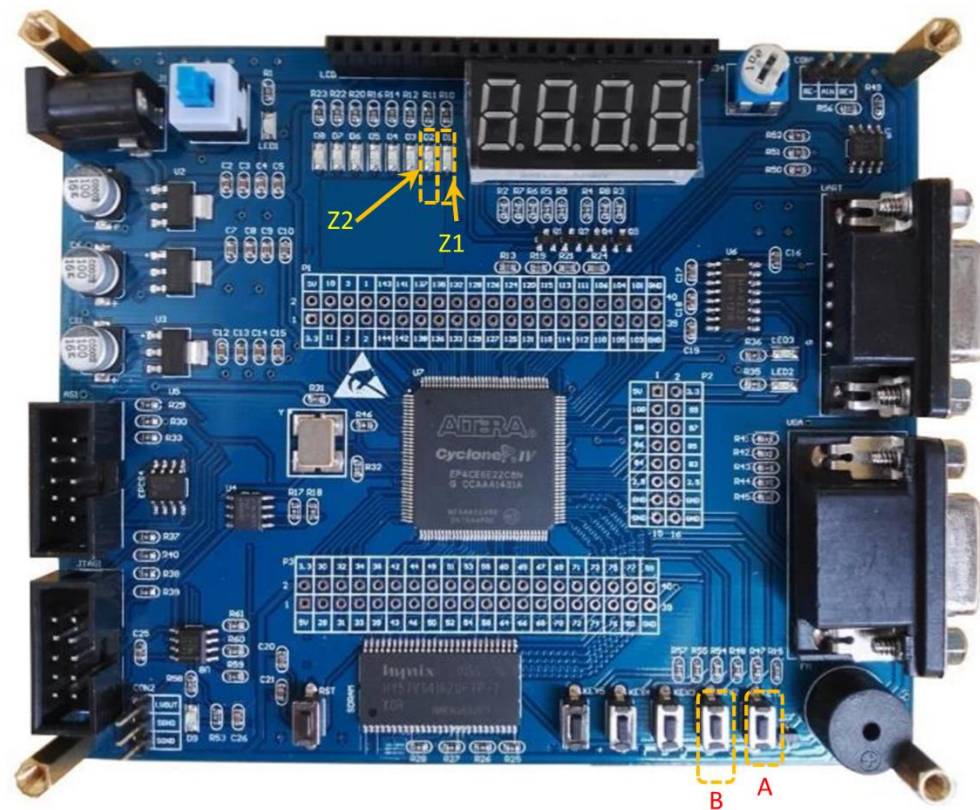


Figure 33. I/O involved in lat1 design

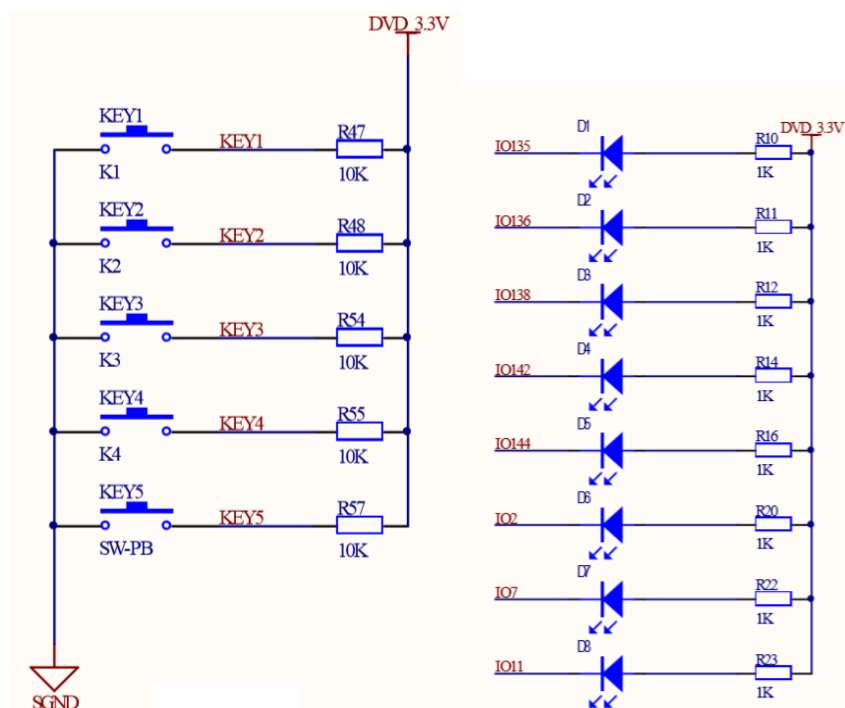


Figure 34. Push button and LED circuit

LED

D8	D7	D6	D5	D4	D3	D2	D1
PIN_11	PIN_7	PIN_2	PIN_144	PIN_142	PIN_138	PIN_136	PIN_135

Key

KEY1	PIN_91
KEY2	PIN_90
KEY3	PIN_89
KEY4	PIN_88
KEY5	PIN_25

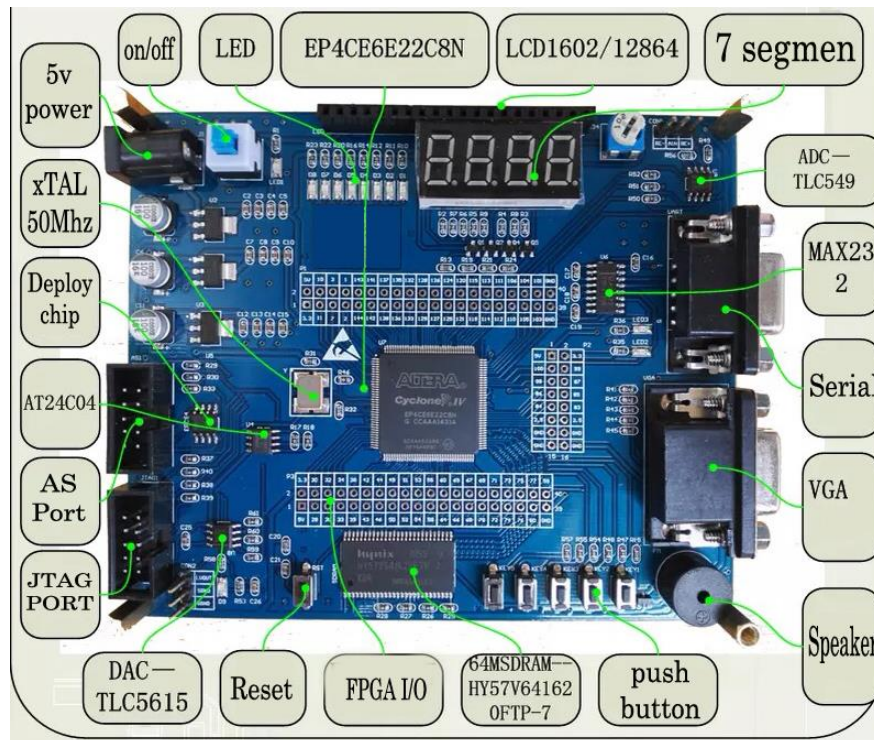


Figure 35. Board display and facilities

V. Practical Assignment

1. Create a design with one input and 8 outputs, where input is connected to KEY1 and output to all LEDs. When KEY1 is pressed all LEDs are on, while when it is released then all LEDs turn off.

2. Create a design with 5 inputs and 5 outputs, where the input is connected to KEY1 and the output is connected to all LED1, KEY2 and the output is connected to all LED 2 and so on. When KEY1 pressed LED 1 lights up, while when released all LEDs light up dead.
3. Create a plan to light one or more sides of the a 7 segment display available on the board. Number of ports input and output are up to you.

7 Segmen

Column							
Seven Segment 1		Seven Segment 2		Seven Segment 3		Seven Segment 4	
PIN_114		PIN_115		PIN_119		PIN_120	
Row							
Bit 7	Bit 6	Bit 5	Bit4	Bit3	Bit2	Bit 1	Bit 0
PIN_126	PIN_124	PIN_129	PIN_128	PIN_127	PIN_125	PIN_121	PIN_132

SEG

