**Theoretical Part**

**1. Introduction to lab.**

The purpose of the lab1 is to let us get more familiar with Quartus software and review it. This is crucial because we need the basic knowledge for operating the software so that we can operate more complex tasks in the future. Also we can apply the knowledge from the course to the technical operations.

**2. Discussion of problem (diagrams, flowcharts, requirements)**

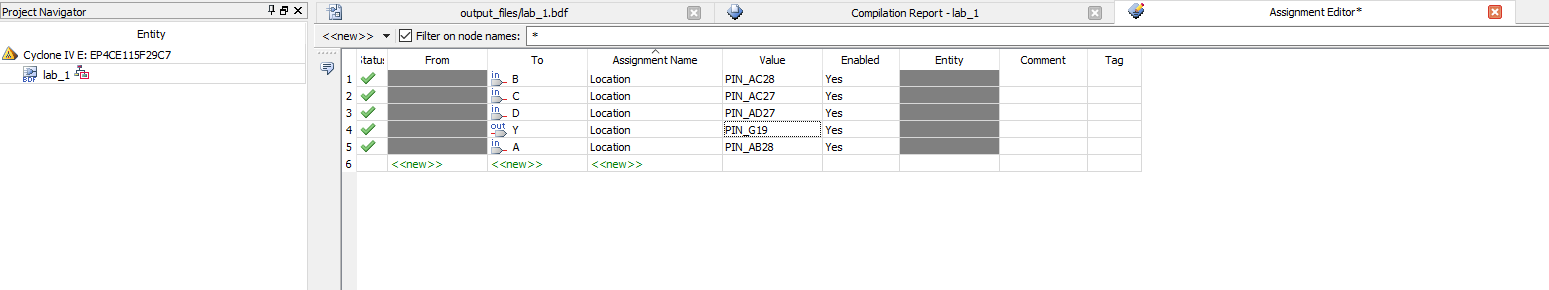
We came cross some errors while compiling the project. After rechecking the block diagram, it was found that the line was not well connected at the port of input (a small cross upon the connecting point).

**3. Discussion of algorithmic solution**

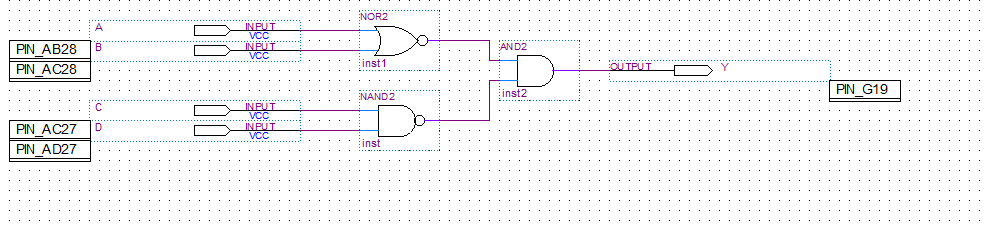
The disconnecting line was deleted and reconnected properly. The project was compiled again, and zero error occurred.

**Design Part**

1. **Presentation of the design**



*Figure1: complete pin assignment editor*



*Figure2: Block diagram with pin assignments*

**Logic equation: (A+B)’(CD)’= Y**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| AB/CD | 00 | 01 | 11 | 10 |
| 00 | 1 | 1 | 1 | 0 |
| 01 | 0 | 0 | 0 | 0 |
| 11 | 0 | 0 | 0 | 0 |
| 10 | 0 | 0 | 0 | 0 |

*Table 1: K map for the diagram*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Truth table** | | | | |
| Input | Input | Input | Input | Output |
| A | B | C | D | Y |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 |

*Table 2: Truth table for the diagram*

1. **Discussion of used components:**

The inclusive NOR (Not – OR) gate has an output that is normally at logic level “1” and only goes “LOW” to logic level “0” when ANY of its inputs are at logic level “1”. The Logic NOR Gate is the reverse or “*Complementary*” form of the inclusive OR gate. The output state of a “Logic AND Gate” only returns “LOW” again when ANY of its inputs are at a logic level “0”. In other words, for a logic AND gate, any LOW input will give a LOW output. The NAND (Not – AND) gate has an output that is normally at logic level “1” and only goes “LOW” to logic level “0” when ALL of its inputs are at logic level “1”. The Logic NAND Gate is the reverse or “*Complementary*” form of the AND gate discussed previously. Logic diagrams are shown below.

**3. Discussion of actual solution:**

Lab instruction was read firstly. A complete truth table was built according to the given circuit diagram. The circuit was then implemented by using block diagram in Quartus II software. Each part of the circuit was named and connected. It was then compiled successfully with zero errors. The time diagram was tested after and the designed circuit was finally connected and simulated on Altera DE2-115 board. First four of the eighteen switches on the board counting from right were standing for A, B, C and D inputs of the circuit. The LED red light on each switch stand for 0 value of the output when it was off and 1 while it was on. After starting the circuit simulation, the circuit was then proved designing correctly by observing the outputs while changing the values of the four inputs. The whole procedure was then showed to TA for rechecking.

**4. Discussion of tool (Optional):**

Used tools and related equipment’s list:

1. Quartus II (web edition):

Quartus II is programmable logic device design software produced by Altera. Digital circuits can be built and functioned on its simulator for computer architecture designing purpose. The software can also be connected to Altera DE2-115 board (introduction #2) which produced by same company to simulate and test circuits.

1. Altera DE2-115 board:

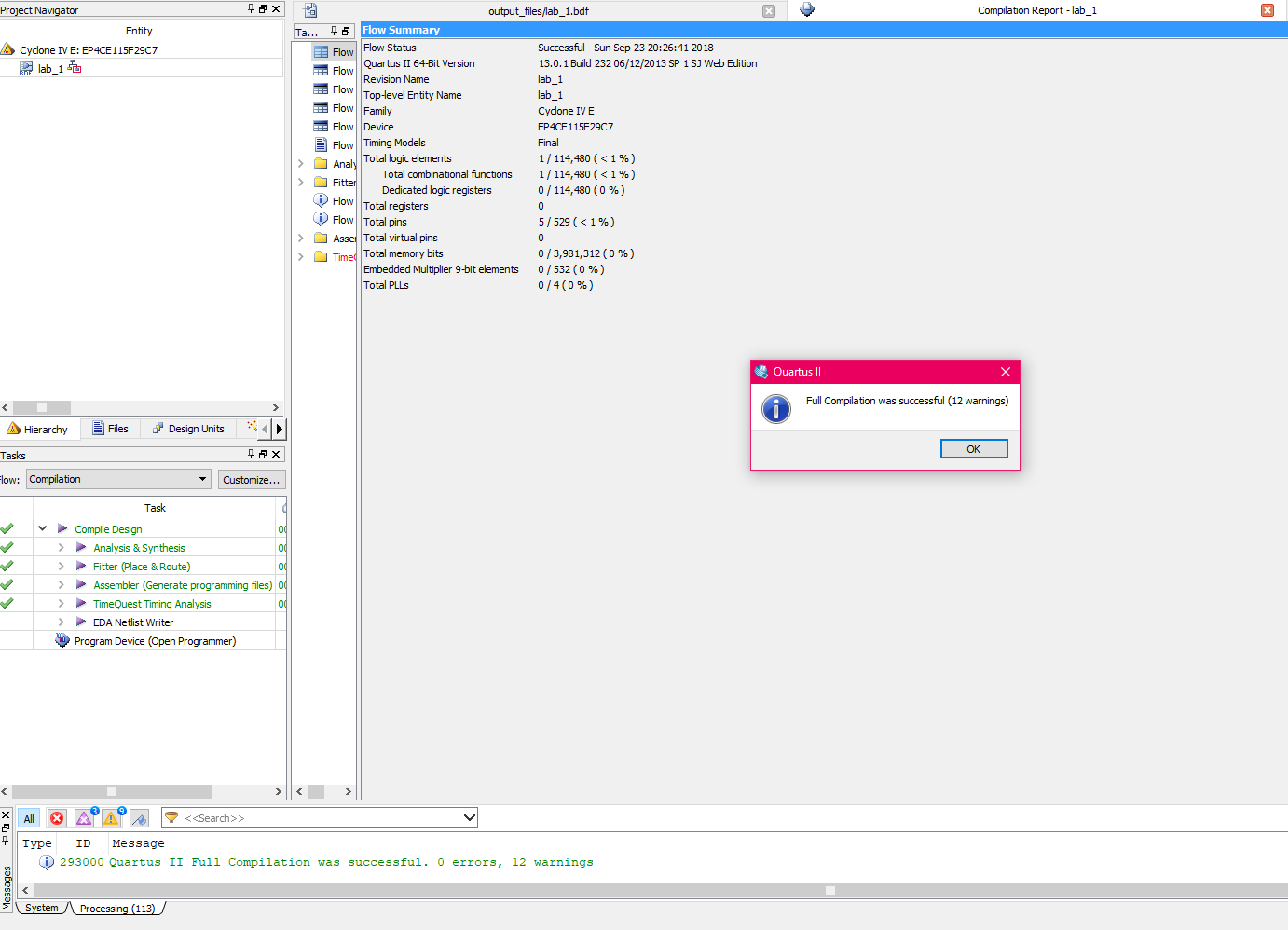
Altera DE2-115 board has USB-Blaster cable and Power supplied by 12V/2A power source. 18 switches and LED lights can operate multiple-input digital circuits and test the outputting results.

**5. Discussion of challenging problems (Bonus):**

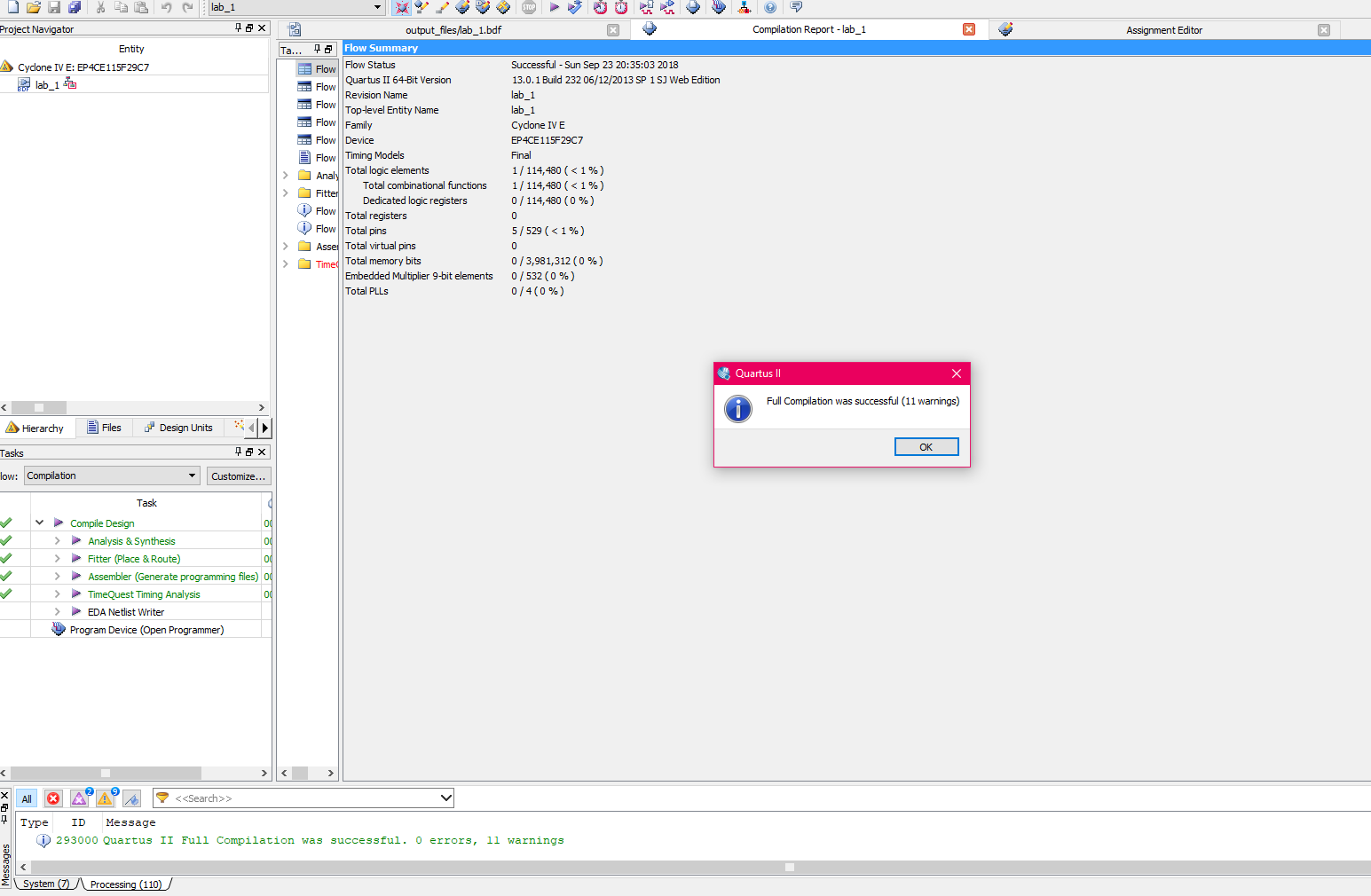
In all, theoretical data matches experimental data and laws had been verified. Everything worked as expected. However, we managed to finish the lab before the lab section, but Quartus II has a massive of data that would take a huge amount of time to download under university internet area. Therefore, we searched around the libraries and labs in university campus; And finally finished the lab at Colonel By hall B302 before the lab section.

**Simulation and Verification Part**

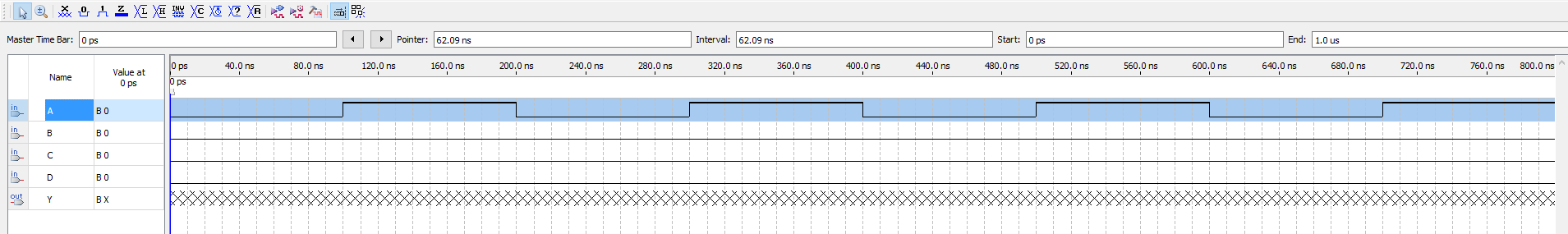
**1. Shown simulation/synthesis results**



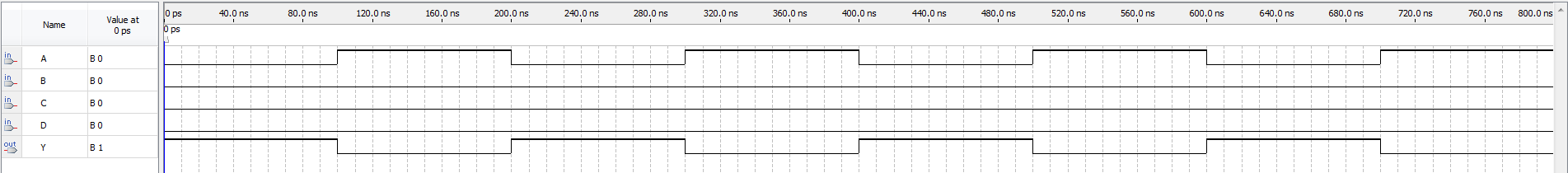
*Figure 3: Screenshots of the compilation results 1*



*Figure 4: Screenshots of the compilation results 2*

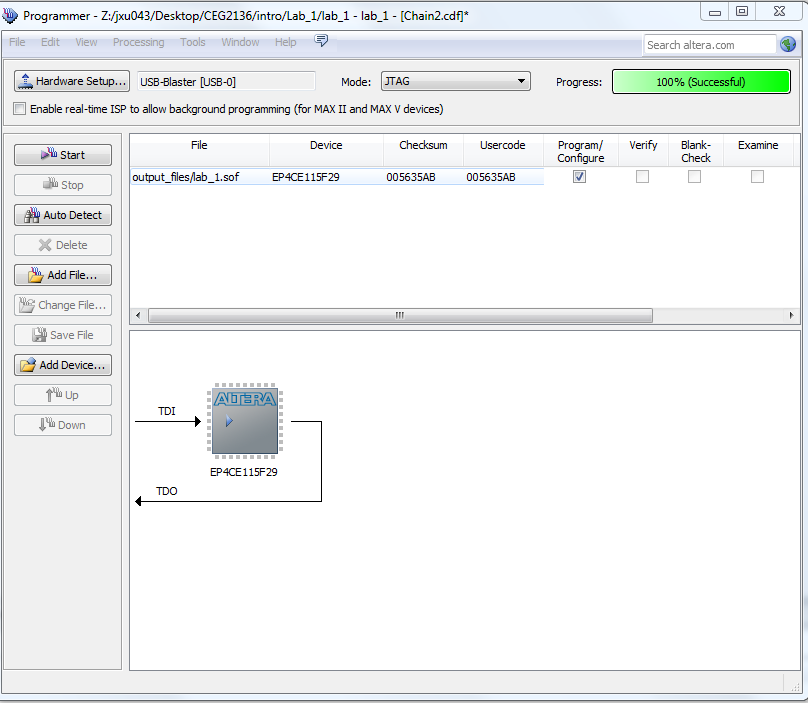


*Figure 5: Screenshots of the simulation*



*Figure 6: Waveform diagram complete*

1. **Verification (live demonstration)**



*Figure 7: screenshot of successful*

After it shows successful, the output displays 1. Because the initial input is 0000, compared with the output, it matches.

Applied all 16 possible logic combinations to the circuit’s inputs by modifying the switches (A is SW0, B is SW1, C is SW2, D is SW3), observed the output Y(LEDR0), compared with truth table of the output, all of them are matched.

**Discussion and conclusions**

Circuit on the task was implemented on Quartus II software and tested on Altera DE2-115 board. The results of outputs were the same as the ones on truth table derived in prelab. Therefore, the lab was successfully operated. Giving a briefly review of the Quartus II software and practising the knowledge learnt in class were the main points of this lab. During the lab, there are a few problems occurred while compiling the project on Quartus II and they were managed to be solved by asking for help from teaching assistance and following the instructions carefully.