Lab 3: Arithmetic Logic Unit (ALU)

**CEG 2136 B - Computer Architecture**

**Fall 2017**

**School of Electrical Engineering and Computer Science University of Ottawa**

Course Coordinator: Dr. Sawsan Abdul-majid

Group 18

Felix Singerman 7970742

Nasser Haidar  8543506

Submission Date: November 6, 2017

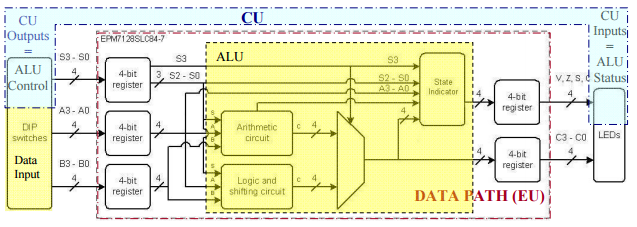
# Theoretical Part

1. **Introduction of problem**

In this lab we designed, simulated, and built an Arithmetic Logic Unit (ALU). The ALU has to execute 16 different operations on two operands of 4 bits and will provide a 4-bit result as well as 4 status bits (Overflow, Sign, Zero, Carry).

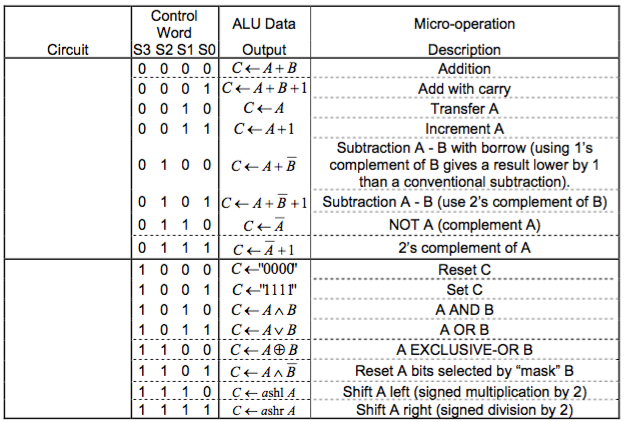
2. **Discussion of problem**

The problem for this lab consisted of designing, simulating, building and testing an Arithmetic Logic Unit.



*Figure 1: ALU datapath*

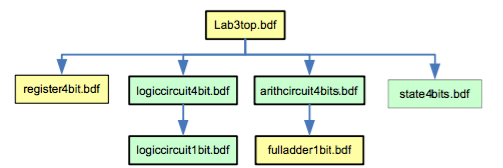
The ALU has to perform the following arithmetic and logic micro-operations:



*Table 1: Arithmetic and logic micro-operations*

**3. Discussion of algorithmic solution**

In order to build the ALU, we used a hierarchical design. The lowest level consists of the 1-bit full adder and 1-bit logic and shift circuit (LSC). The intermediate files consists of a 4-bit register, a 4-bit arithmetic circuit (AC), a 4-bit logic and shift circuit (LSC) and a state circuit. Finally, the highest level will consist of the complete circuit, which is what is loaded to the circuit board.

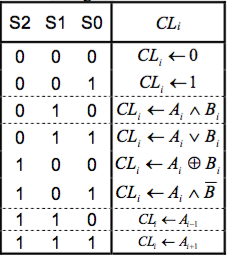


*Figure 2: Hierarchy of the files*

# Design Part

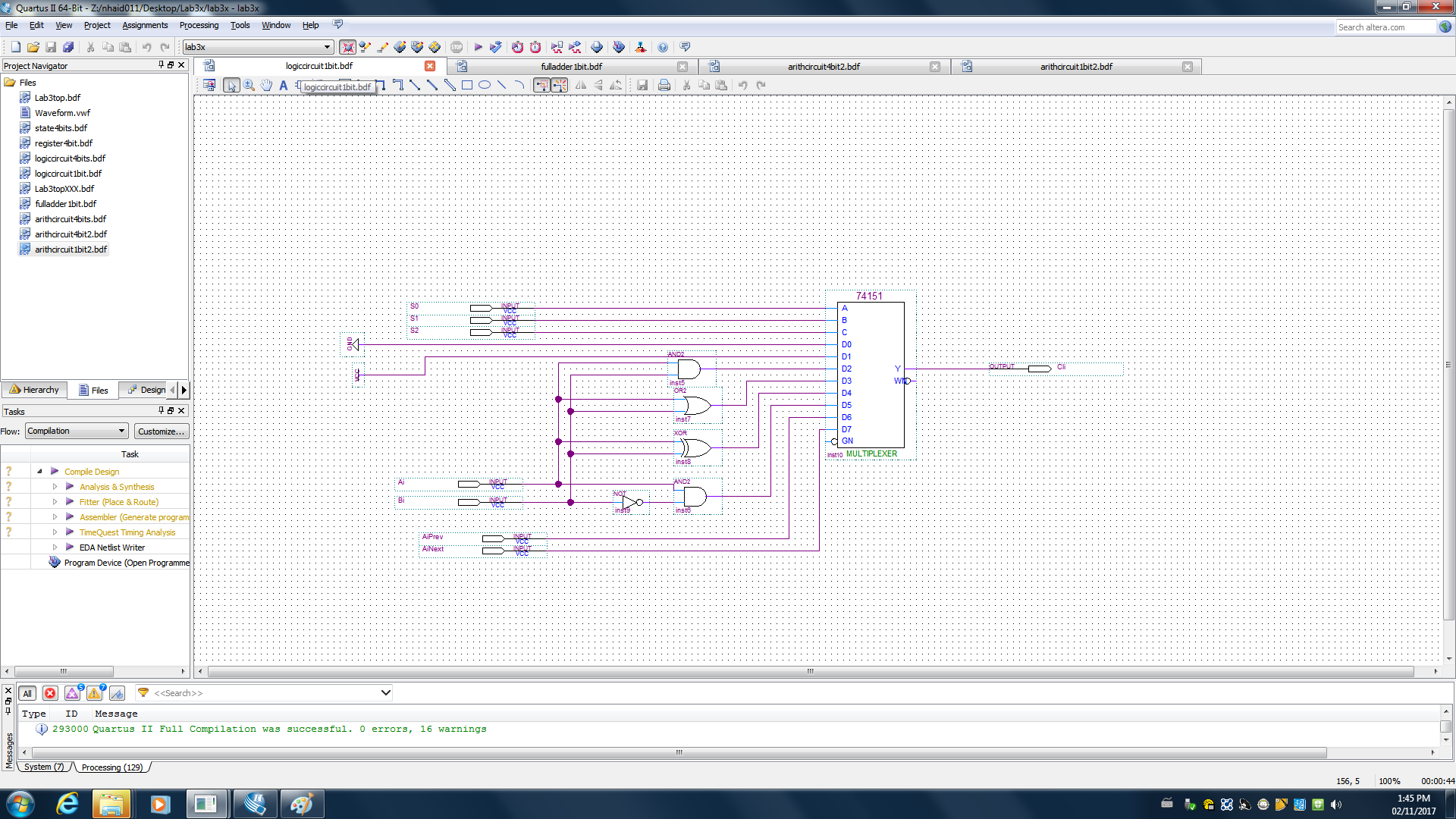
1. **Presentation of the design methodology applied to solving the lab problems**

**Logic Circuit:**

****

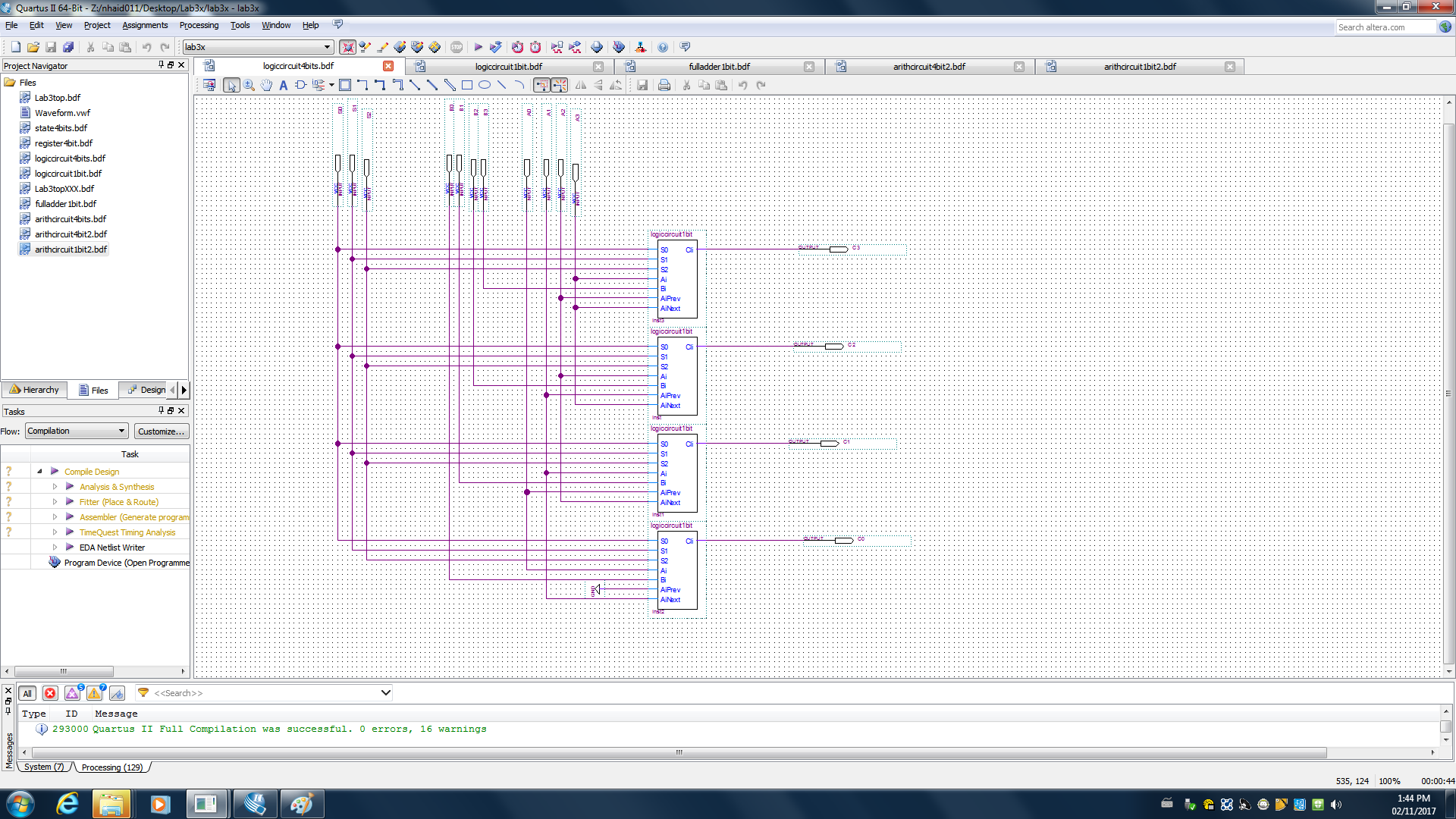
*Table 2: The LSC logic table*

Using Table 1 and a 8-1 Multiplexer we were able to design the LSC 1-bit



*Figure 3: 1-bit LSC*

We combined 4 1-bit LSCs to form a 4-bit LSC.

**

*Figure 4: 4-bit LSC*

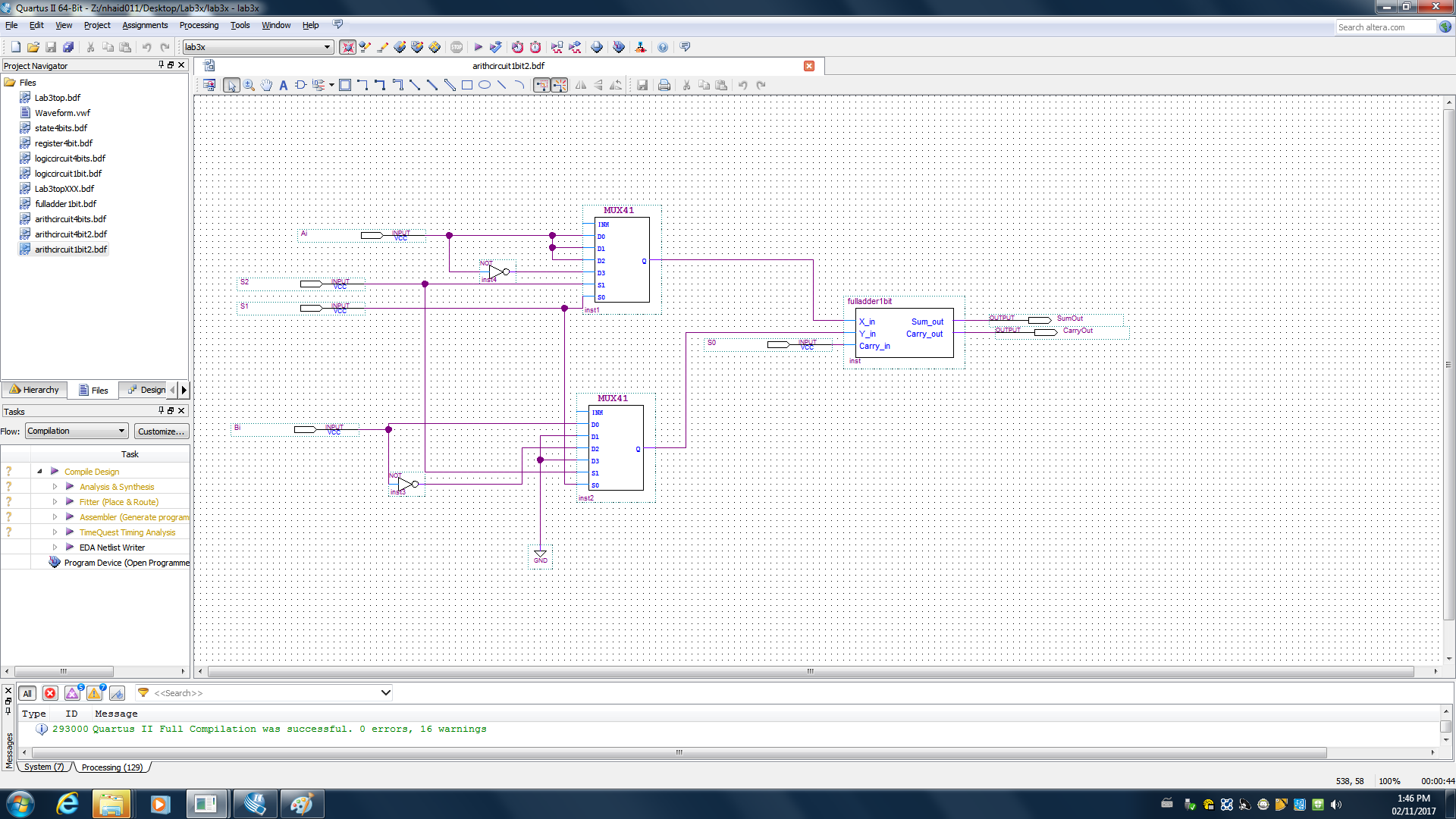
**Arithmetic Circuit:**

We used the CA Outputs to determine the inputs of A and B.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| S2 | S1 | S0 | Op1 | Op2 | Cy\_in | CA Out |
| 1 | 0 | 0 | A | B | 0 | CA🡨A+B |
| 0 | 0 | 1 | A | B | 1 | CA🡨A+B+1 |
| 0 | 1 | 0 | A | 0 | 0 | CA🡨A |
| 0 | 1 | 1 | A | 0 | 1 | CA🡨A+1 |
| 1 | 0 | 0 | A | B’ | 0 | CA🡨A+B’ |
| 1 | 0 | 1 | A | B’ | 1 | CA🡨A+B’+1 |
| 1 | 1 | 0 | A’ | 0 | 0 | CA🡨A’ |
| 1 | 1 | 1 | A’ | 0 | 1 | CA🡨A’+1 |

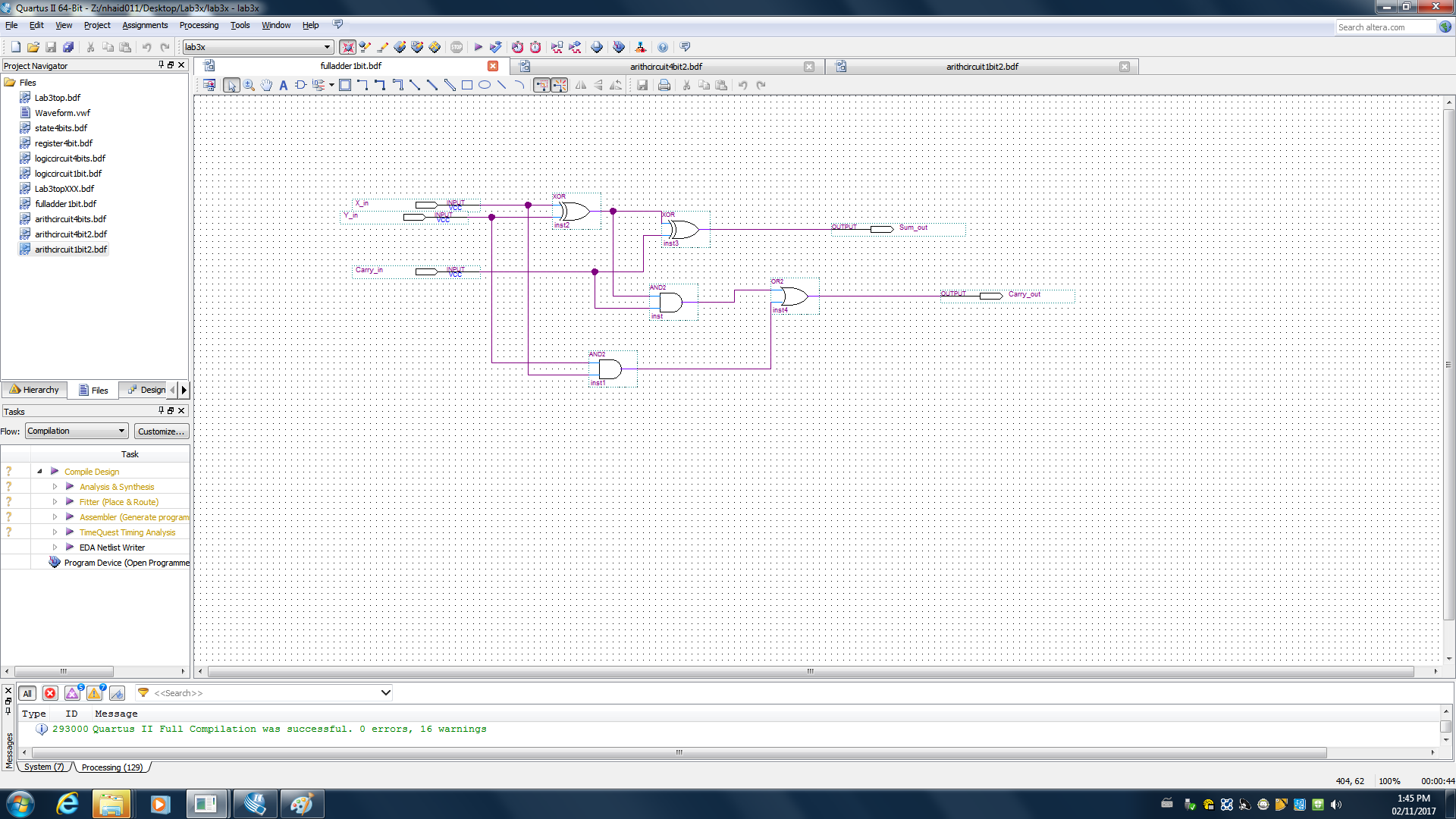
*Figure 5: Operations of AC*

We used Table 2 to determine the inputs for the multiplexers along with a full adder to build the 1-bit Arithmetic circuit.

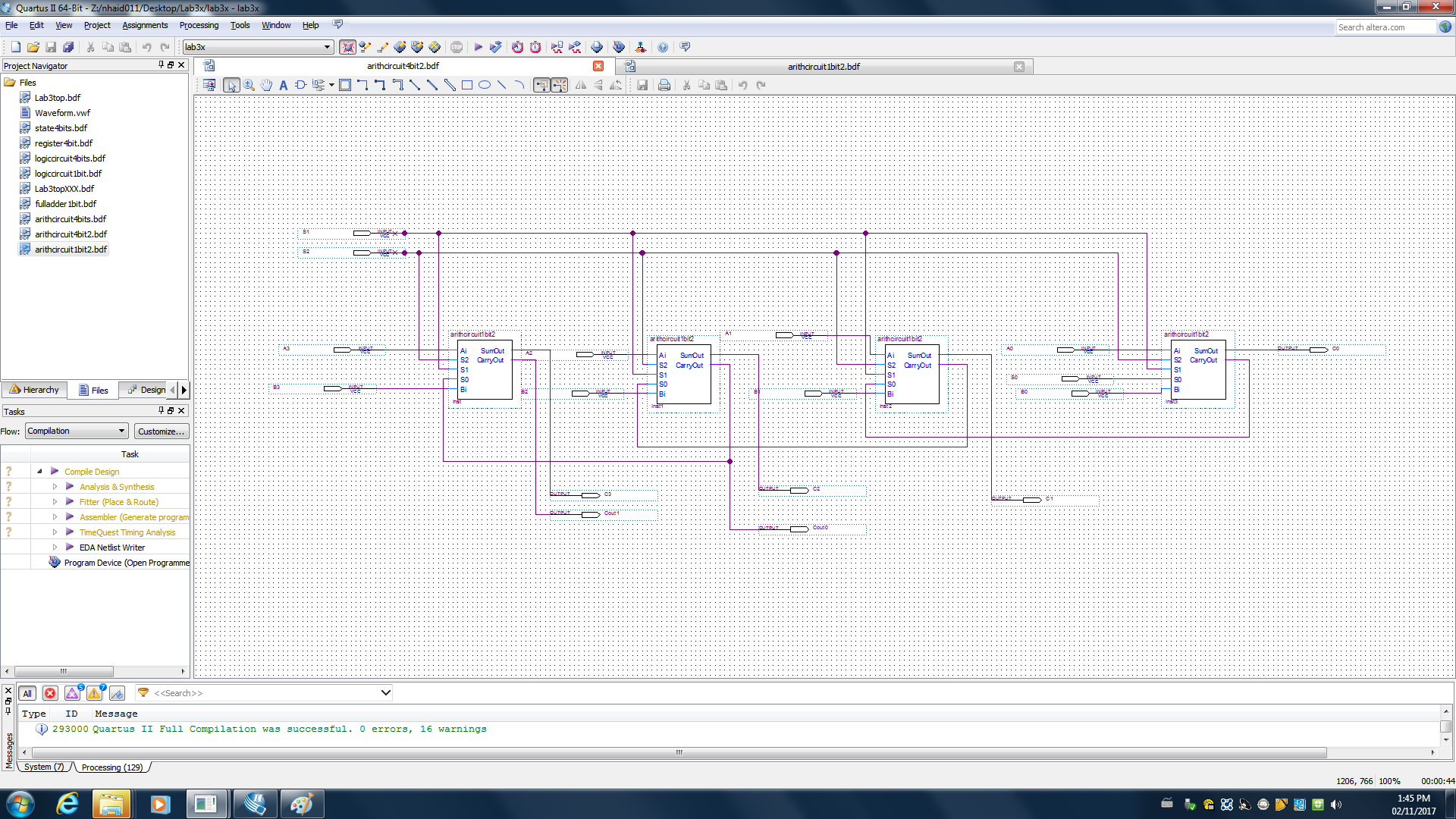
**

*Figure 6: 1-bit Arithmetic Circuit*

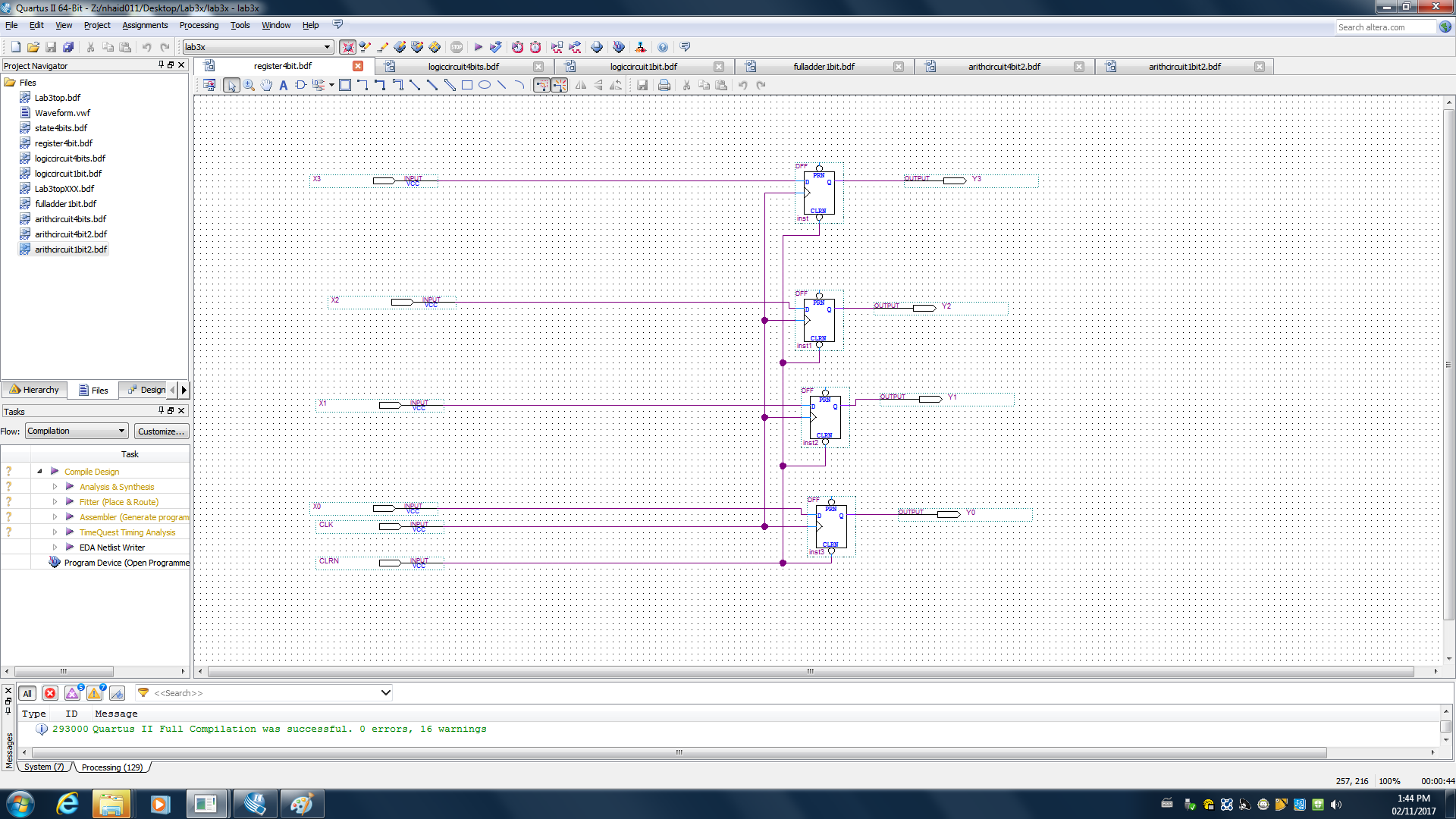
We combined 4 1-bit ACs to form the 4-bit AC.

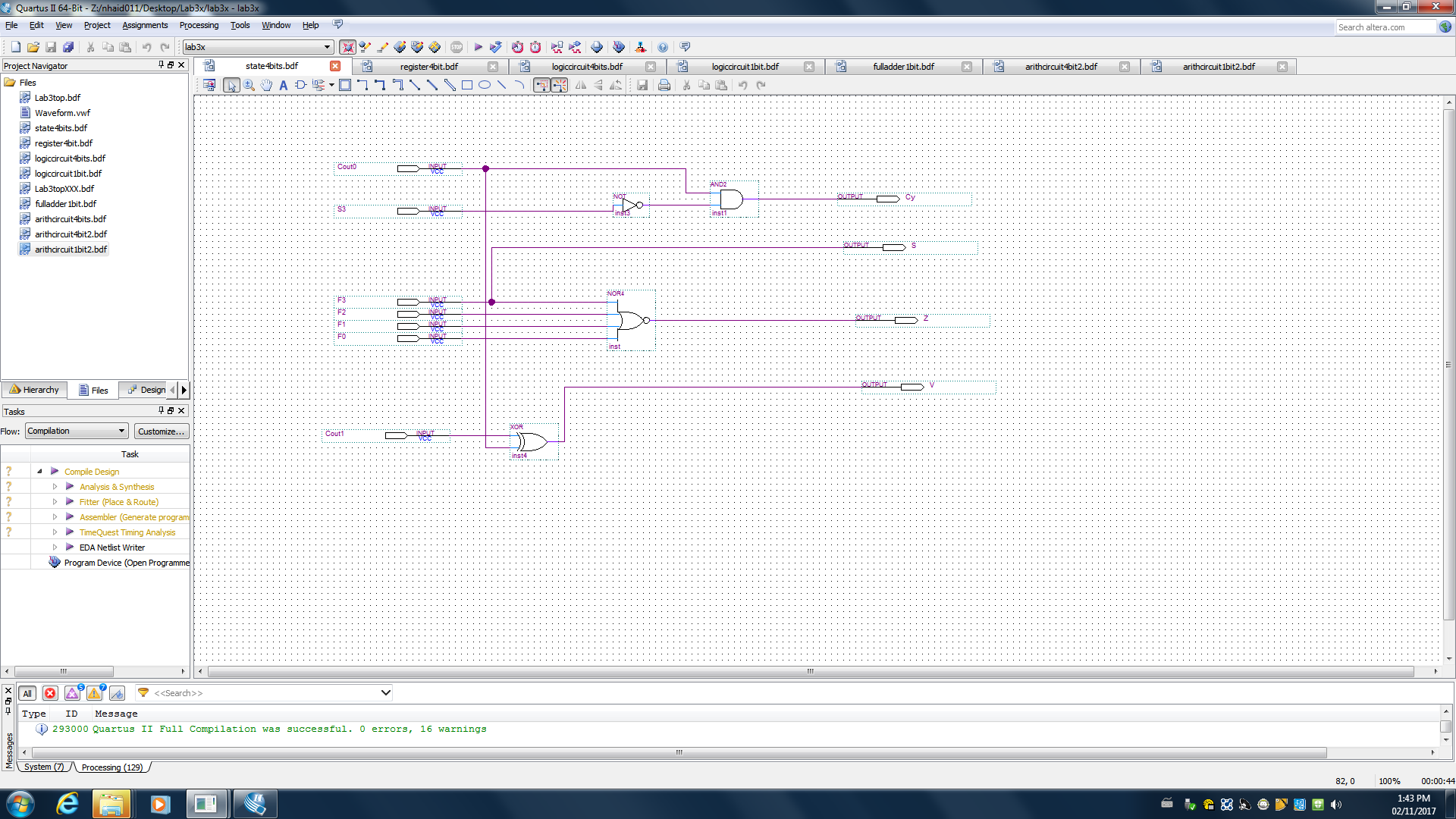
**

*Figure 7: 1-bit Full Adder*

**

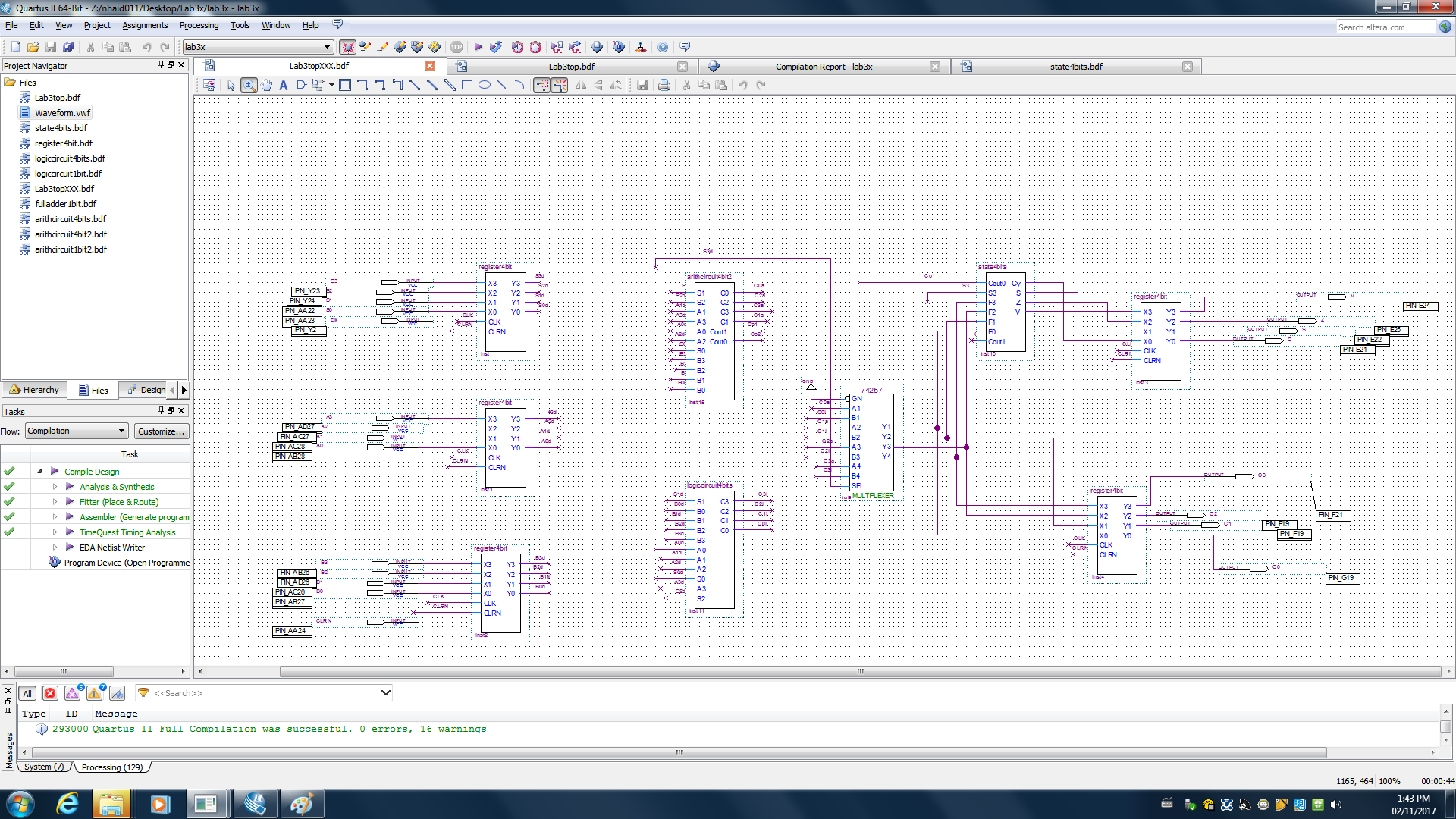
*Figure 8: 4-bit Arithmetic Circuit*

*Figure 9: 4-bit Register*

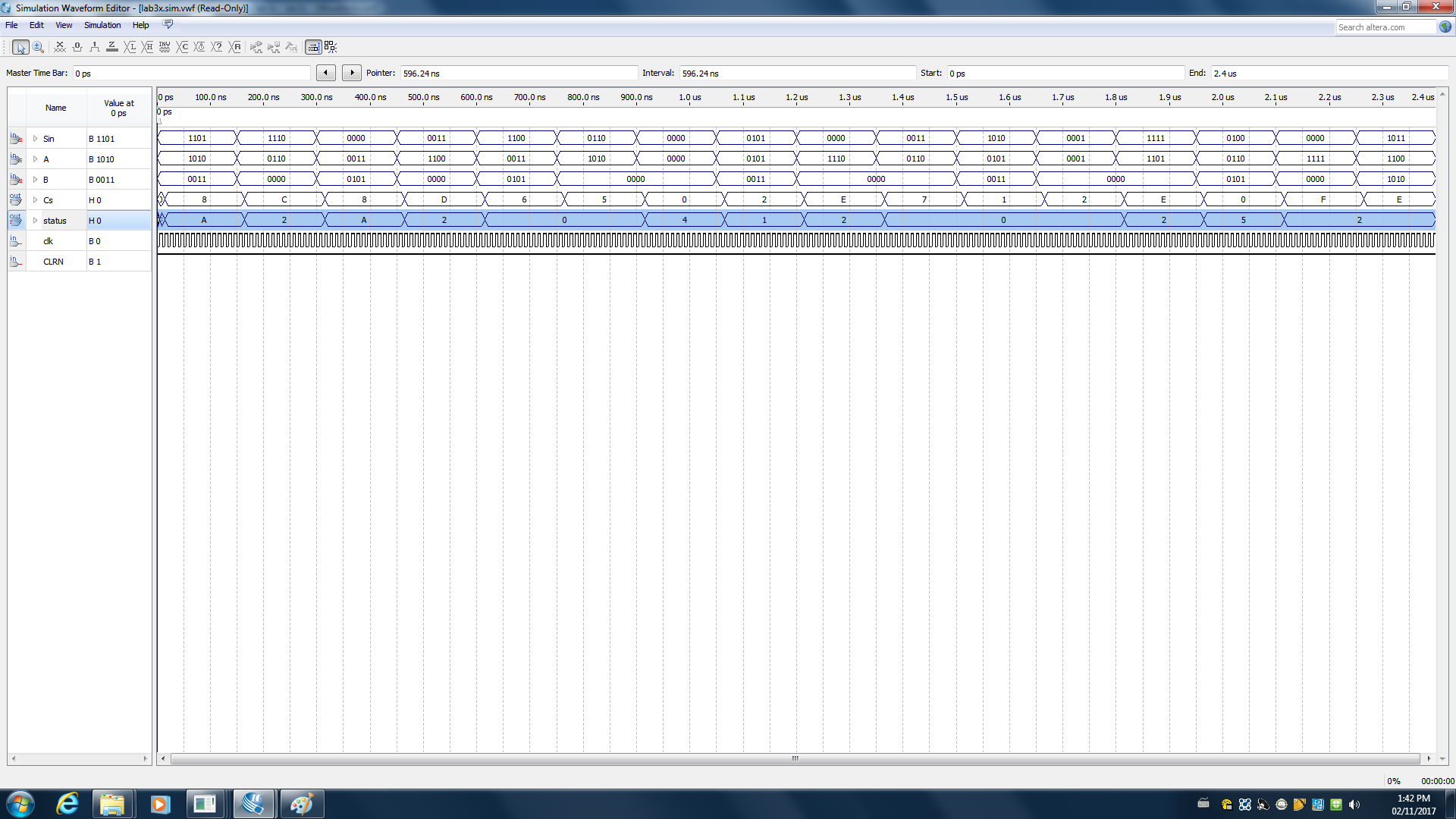
**

*Figure 10: State 4-bit Register*

Finally*,* using specifications provided in the lab manual we created this ALU.

**

*Figure 11: Complete ALU Design*



*Figure 12: Waveform Diagram of the complete ALU*

**2. Discussion of used components**

**AND Gate**

A gate where the output is 1 if and only if both inputs have a value of 1, otherwise, its output is 0.

**NOT Gate**

A gate where the output is 1 if the input is 0, and vice versa.

**XOR Gate**

A gate where the output is 1 if only one of the inputs is 1.

**NOR Gate**

A gate where the output is 1 if and only if both inputs are 0, otherwise, its output is 0.

**Circuit Diagrams**

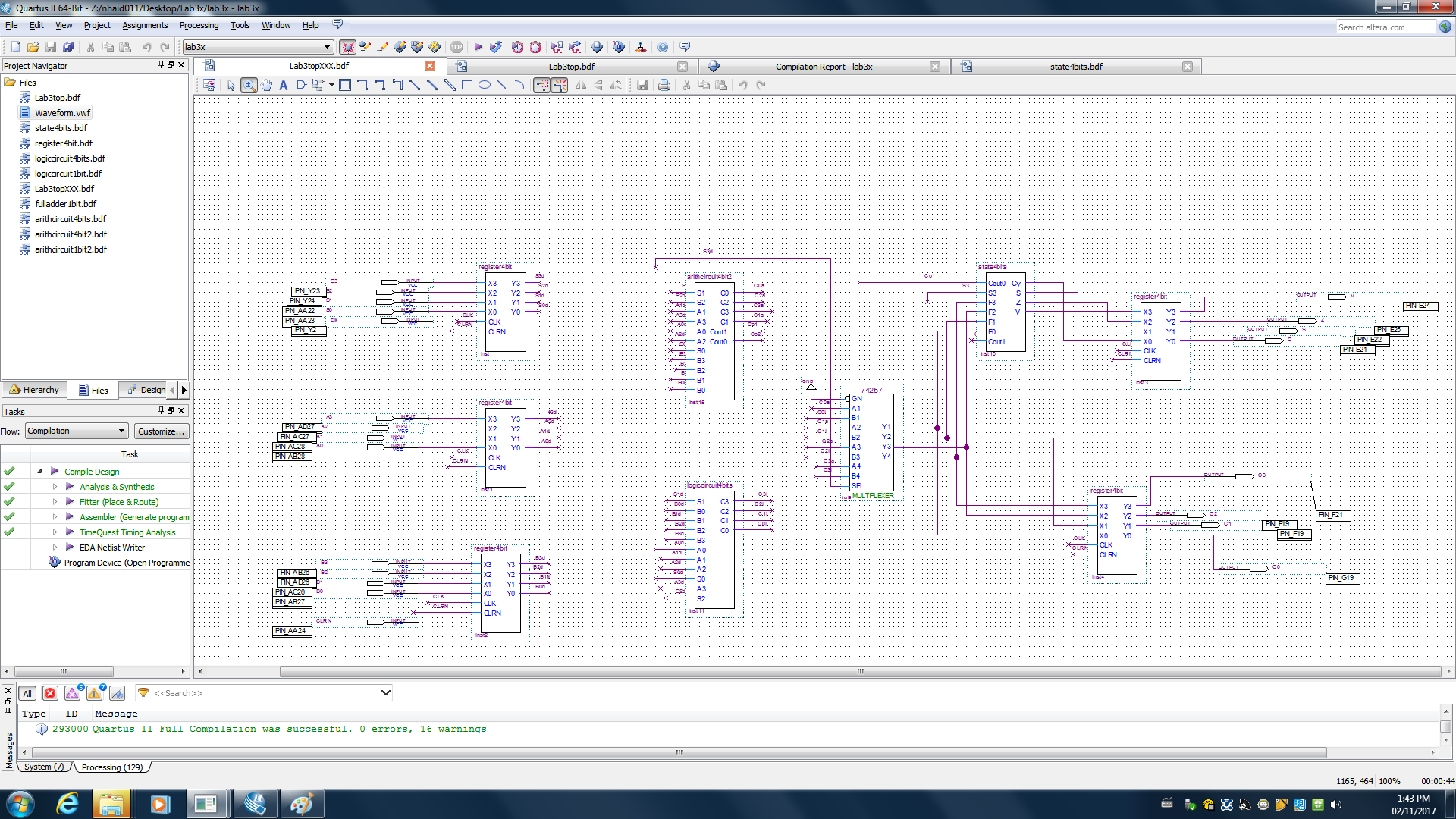
*Refer to “Design Part”*

**3. Discussion of actual solution**

**Circuit Diagrams**

*Refer to “Design Part”*

After creating each circuit separately, we created symbols for each one to make it easier to connect everything together. This can be seen in the figure below. Since an ALU is a combinational circuit, a control signal - S3 - is used to determine whether the Arithmetic Circuit or the Logic and Shifting Circuit will be used. Furthermore, the control signals S2-S0 is used to determine which operations will be performed by each circuit. There are 8 data inputs: A3 - A0 and B3 - B0, 4 data outputs (C3 - C0) and the most significant carry bits are used for the status circuits.

**

*Figure 12: Complete ALU Design*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Cycle | S3 | S2 | S1 | S0 | A←op1 | B←op2 | C | V, Z, N, Cy | S16 | A16 | B16 | C16 | St16 |
| 1 | 1 | 1 | 0 | 1 | 1010 | 0011 | 1000 | 1010 | D | A | 3 | 8 | A |
| 2 | 1 | 1 | 1 | 0 | 0110 | X | 1100 | 0010 | E | 6 | X | C | 2 |
| 3 | 0 | 0 | 0 | 0 | 0011 | 0101 | 1000 | 1010 | 0 | 3 | 5 | 8 | A |
| 4 | 0 | 0 | 1 | 1 | 1100 | X | 1101 | 0010 | 3 | C | X | D | 2 |
| 5 | 1 | 1 | 0 | 0 | 0011 | 0101 | 0110 | 0000 | C | 3 | 5 | 6 | 0 |
| 6 | 0 | 1 | 1 | 0 | 1010 | X | 0101 | 0000 | 6 | A | X | 5 | 0 |
| 7 | 1 | 0 | 0 | 0 | X | X | 0000 | 0100 | 8 | X | X | 0 | 4 |
| 8 | 0 | 1 | 0 | 1 | 0101 | 0011 | 0010 | 0001 | 5 | 5 | 3 | 2 | 1 |
| 9 | 0 | 0 | 1 | 0 | 1110 | X | 1110 | 0010 | 2 | E | X | E | 2 |
| 10 | 0 | 1 | 1 | 1 | 0110 | X | 1010 | 0010 | 7 | 6 | X | A | 2 |
| 11 | 1 | 0 | 1 | 0 | 0101 | 0011 | 0001 | 0000 | A | 5 | 3 | 1 | 0 |
| 12 | 0 | 0 | 0 | 1 | 0001 | 0010 | 0100 | 0000 | 1 | 1 | 2 | 4 | 0 |
| 13 | 1 | 1 | 1 | 1 | 1101 | X | 1110 | 0010 | F | D | X | E | 2 |
| 14 | 0 | 1 | 0 | 0 | 0110 | 0101 | 0000 | 0101 | 4 | 6 | 5 | 0 | 5 |
| 15 | 1 | 0 | 0 | 1 | X | X | 1111 | 0010 | 9 | X | X | F | 2 |
| 16 | 1 | 0 | 1 | 1 | 1100 | 1010 | 1110 | 0010 | B | C | A | E | 2 |

*Figure 13:* Sequence of micro-operations to execute

**4. Discussion of tool**

We first needed to design the circuit in our prelab using the given operations and tables. The arithmetic logic unit included many smaller circuits that all contributed to the final schematic diagram. First, we needed to design and implement a 1-bit arithmetic circuit to handle the arithmetic side of the unit. Since we were creating a four-bit circuit we then made a 4-bit arithmetic circuit using four symbols we created of the 1-bit circuit. The next part that was designed was the logic and shift side of the unit. We approached this the same way we did with the arithmetic circuit, starting with 1-bit logic circuits and then creating a 4-bit one from that. Finally, we needed to create the 4-bit status register. During the implementation, we also had to create schematics for a 4-bit register and a full adder that were provided in the instructions. Using all of these circuits we implemented the arithmetic logic unit that is seen in the above diagrams.

**Altera DE2-115 Board**

This is a circuit board which consists of multiple pins, buttons, and LEDs. It allows us to visualize and test our designed circuits from the Quartus software.

**5. Discussion of challenging problems**

We experienced several problems during this lab, but with the guidance of the TA’s we were able to solve them. When simulating in our waveform we had our MSB and LSB swapped so we were not receiving the results that we were supposed to. We also weren’t connecting the diagrams together properly as there were many lines. We were getting the first Arithmetic computation correct so we narrowed down the issue to be part of the Logic. We started from the 1-bit and worked our way up. We now have gained a full understanding of the ALU and all of the components needed to create one.