Lab 3 – Arithmetic Logic Unit

CEG 2136 – Computer Architecture I

Fall 2022

University of Ottawa

Lab TAs: Reza & Yang

Jaiden Clee 300174453

Jasmin Cartier 300160723

Experiment Dates: 10/13/2022, 10/20/2022, 11/03/2022

Table of Contents

Introduction

The purpose of this lab was to design and simulate an Arithmetic Logic Unit (ALU) using Quartus II and the Altera DE2-115 board. The ALU was a combinational circuit which included a Logic and Shift Circuit (LSC) and an Arithmetic Circuit (AC) which worked together to execute operations. The ALU had to execute 16 different micro-operations on the 4 bit operands, A and B. As outputs, the ALU had to produce a 4-bit result of the micro-operation along with 4 status bits. These status bits represented the overflow of the ALU, the sign of the micro-operation output, the carry value, and the zero value of the circuit. Once the ALU was designed, it was tested on the Altera board.

In preparation for this lab, the LSC and AC units of the ALU were designed. Starting with the logic and shift circuit, the micro-operations for this unit were studied and then a 1 bit module was designed to execute these operations. The 4 bit model of the LSC was designed by building on the 1 bit module design. The AC unit was designed in a similar manner. The LSC and AC units were then combined to form the ALU design. After completing this design, the equations were derived for the status bits and ALU output values. Finally, a table of simulation sequences was derived, which included inputs for A and B with the expected ALU outputs. This table was used later on during the simulation and testing of the ALU on the Altera board.

Upon completing the lab preparation, each circuit design was built and compiled in Quartus. After the final design of the ALU was completed and compiled, it was simulated and tested on the Altera board. These results were verified with the simulation sequences derived earlier.

Design

For this design five 4-bit registers have been used, three for the inputs of the ALU and two for the outputs. Figure 7 shows the simple implementation of the registers with D flip-flop. The preset and the clear are set to one because they are active low. The first register is the selection register, where the three least significant bits tell the AC or the LSC what micro-operation to do. The most significant bit tells the 74257 multiplexer, present on the figure 18, which one between the AC and the LSC to choose as output of the ALU. The ALU outputs 4 status bits, along with the output from the micro-operation executed in either the AC or the LSC. The ALU accepts two 4-bit inputs, A and B, which are also stored in two 4-bit registers. These registers both feed into the AC and the LSC. The AC and the LSC are 4-bit circuits built using 1-bit modules.

Figure 8 shows the implementation of the 1-bit Logic and Shift Circuit. S0, S1, S2 are used to select between the different inputs of the 3-8 multiplexer. Figure 1 shows each input and for the two first operations the input is equal to S0. Although VCC and GND could have been used we were not aware how to implement those in Quartus at this point, so this implementation has been used. For input 2 to 5 of the multiplexer we can see respectively (A and B), (A or B), (A or exclusive B) and (A and B'). The two last inputs are used for the left shift and the right shift. You can see in Figure 10 how they are used to shift the values to the left or the right. This isn't a circular shift so S0 has been used to input zero for the most significant bit or the last significant bit.

Figure 12 shows the implementation of the 1-bit Arithmetic Circuit. S1 and S2 choose the inputs of the full adder component (Figure 6) from the two 2-4 multiplexers. Four basic case are possible; (S1 =

0, S2 = 0) inputs: A and B, (S1 = 1, S2 = 0) inputs: A and 0, (S1 = 0, S2 = 1) inputs: A and B', (S1 = 1, S2 = 1) inputs: A' and 0. As you can see in Figure 3 on the 5.4.1 Table, each time S0 = 1 the result from the line above that has the same S1 and S2 is just incremented by one. This explains why the Carry_in input of the full adder is equal to S0.

Finally, Figure 16 shows the 4-bit State Register circuit implementation. This circuit is complex compared to other circuit implementations in this lab since each output has its own unique logic equation. Cy is set to one if the most significant value of the AC is one, and the output is from the AC (Cy = S3' and C03). S represents the sign of the output so it is simply equal to the most significant bit of the ALU output. Z is set to zero if all the ALU output bits are zero (Z = (ALU0')(ALU1')(ALU2')(ALU3')). V is the overflow detection and an overflow can occur in two scenarios. The first scenario is in the AC when adding to a signed number. Meaning an overflow occurs when S3 equals one (selecting the AC) and the two most significant carry values are different. The second scenario is in the LSC when a left shift is executed. An overflow occurs when S0 = 0, S1= 1, S2 = 1, S3 = 1 and the two most significant bits of A are different. The equation of V is (S0' S1 S2 S3)(A3 or exclusive A2) or (CO3 or exclusive CO2)(S3).

Simulation and Verification of Implemented Design

Table 1 shows 16 operations with different A and B input values and their expected results. Figure 20 represents the waveform simulating the final design. In the waveform, "Selection" represents the four selection bits in order. A and B are the inputs and for the "don't care" inputs, random inputs have been assigned. C is the ALU output and V, Z, S, Cy are the outputs of the 4-bit State Register.

The waveform simulation outputs were verified with Table 1 and then the ALU was uploaded to the Altera board. At first sight the data on the board wasn't corresponding. After double checking that all the PINs were correctly assigned and testing again we noticed it was only a testing error. The bits were read left to right in Table 1 but the switch corresponding to the inputs was flipped right to left. After compensation for this error, the Altera board outputs were correctly verified with the expected outputs.

Discussion

In this lab, Quartus II was used to design and simulate an Arithmetic Logic Unit(ALU) that executed 16 different micro-operations on 4-bit operands. This unit was implemented by a Logic and Shift Circuit(LSC), and an Arithmetic Circuit(AC). The ALU produced a 4-bit micro-operation output along with 4 status bits. Before completing any new designs, circuit diagrams were built and compiled in Quartus for a 1-bit full adder, and a 4-bit register. These diagrams can be found in Figures 6 and 7 respectively.

Firstly, the Logic and Shift Circuit(LSC) was designed. The LSC had to perform operations such as left and right logic shifts, boolean logic functions, and clearing the ALU outputs to zero or one. A 1-bit LSC module was designed to facilitate the final 4-bit design. In the 1-bit LSC module, an 8-1 multiplexer was used to choose between the 8 micro-operations this unit had to execute. For this multiplexer, S0, S1, and S2 were used as selection bits. As shown in Figure 8, for this module we used an LMP multiplexer which was imported from the Quartus library. After the 1-bit module was compiled, a symbol was created

and used to create the final 4-bit LSC. In the 4-bit LSC, the 1-bit module was repeated four times to accommodate for each bit of the 4-bit operands. This circuit design can be seen in Figure 10. The 4-bit circuit was then compiled and the fitter summary was analyzed. This summary, which can be seen in Figure 11, shows that the 4-bit LSC used about 3% of the chip resources. No difficulties were encountered during this part of the design process.

Secondly, the Arithmetic Circuit(AC) was designed. The AC had to perform operations such as addition, subtraction, 2's complement, incrementing the operands, and transferring operands into the output. In a similar manner to the LSC, a 1-bit module of the AC was created first. This module accepted two 1 bit operands (A and B) and the selection bits as inputs. Two 4-1 multiplexers along with a 1-bit full adder were used as the main components. Unlike the 1-bit LSC module, the 1-bit AC used S1, and S2 as the selection bits for both multiplexers while S0 was used as the carry in value for the full-adder. In this design, two 4-1 LMP multiplexers were used instead of one 8-1 multiplexer to have a cost-effective solution. As shown in Figure 12, each multiplexer dealt with one operand whose outputs were fed into the full-adder to produce the final circuit output. During the design and testing of the 1-bit AC module, we noticed that the circuit was not producing the correct outputs for every micro-operation. We discovered that the multiplexer was receiving the selection bits in the wrong order which meant the micro-operations were not being chosen correctly. After fixing this mistake, the 1-bit module was working correctly and a symbol of this module was created and used to design the final 4-bit AC. As shown in Figure 14, the 1-bit module was repeated four times, and the final 4-bit circuit produced the micro-operations outputs along with the two most significant carry values. The 4-bit AC was compiled and the fitter summary was analyzed. As shown in Figure 15, the final AC used about 3% of the chip resources which is almost identical to the 4-bit LSC.

Thirdly, the ALU status circuit was designed. This circuit was responsible for producing the 4 bits (Cy, S, Z, V) that corresponded to the ALU status. Cy was set to true if an arithmetic operation produced a carry of 1. S was set to the sign of the most significant bit of the ALU result. Z was set to true if the ALU output contained all zeros. Lastly, V was set to true when an overflow occurred during operations on signed numbers in 2's complement representation. The derived equations for these status values are shown in Figure 5. After the status circuit was designed and compiled, a symbol was created to use in the final ALU circuit design. There were no difficulties in this part of the design process.

Finally, the Arithmetic Logic Unit(ALU) was designed. Symbols of the 4-bit register, 4-bit LSC, 4-bit AC, and the state circuit were created and combined to create the final ALU circuit. To keep the design easier to follow, we learnt how to use virtual pin assignments. This allowed us to not have wired connections all over the design, and is shown in Figure 18. In the beginning, there were difficulties in using this technique but once we learnt how to properly name the connections, this design method was favored. After completing the ALU design, it was compiled and the fitter summary was analyzed. As shown in Figure 19, the complete ALU used about 4% of the chip resources. Then the ALU was simulated using the test sequences we derived in lab preparation, which are shown in Table 1. First, we simulated the ALU using a waveform, shown in Figure 20, and the output values were verified with our expected output values. Note that in the waveform simulation, there is a slight time delay between a change of input and output due to the input and output registers. We have decreased our clock period to

minimize this time delay in our waveform simulation. After verifying the waveform outputs, pin assignments were made and the ALU was tested on the Altera board.

In conclusion, this lab combined many skills learnt in previous labs and lectures. The design process was extremely helpful in learning how the different components of an ALU work together to produce an output. Additionally, the virtual pin assignment technique was helpful, and will most likely be used in future labs to help keep our designs clear.

Appendix

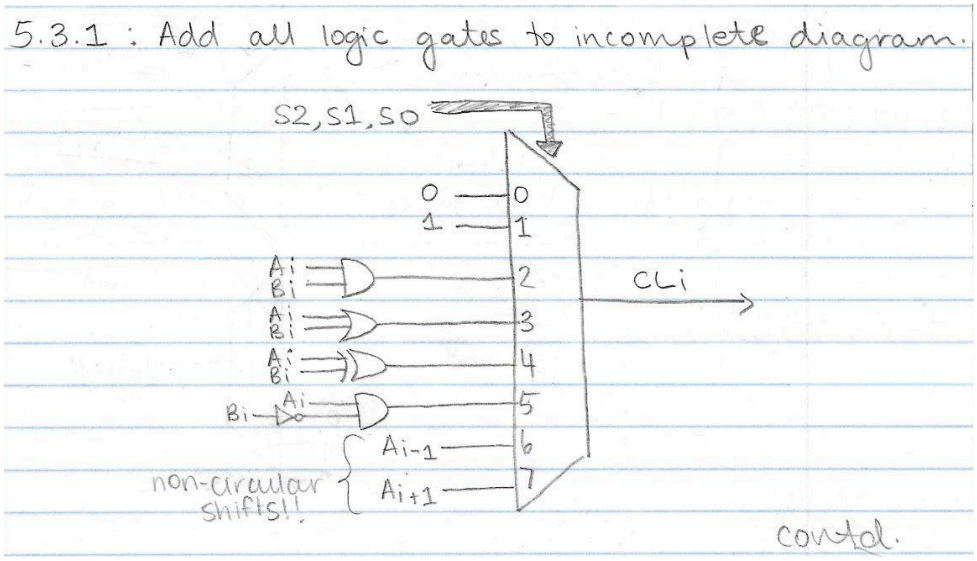

Figure 1: 1-bit module design of the Logic and Shift Circuit derived in lab preparation

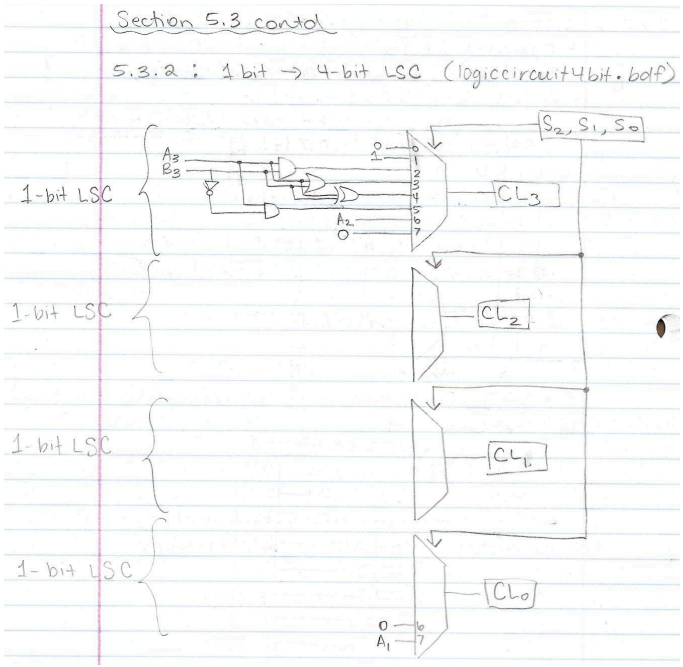

Figure 2: 4-bit module design of the Logic and Shift Circuit derived in lab preparation

5.4.1: Complete truth table.

S3	S2	S1	S0	op1	op2	Cy-in	CA
0	*0	0	0	A	B	0	A+B
0	0	0	1	A	B	1	A+B+1
0	*0	1	0	A	0	0	A
0	0	1	1	A	0	1	A+1
0	*1	0	0	A	$\bar{B}$	0	A+$\bar{B}$
0	1	0	1	A	$\bar{B}$	1	A+$\bar{B}$+1
0	*1	1	0	$\bar{A}$	0	0	$\bar{A}$
0	1	1	1	$\bar{A}$	0	1	$\bar{A}$+1

5.4.2: Design Op1 and Op2 multiplexers.

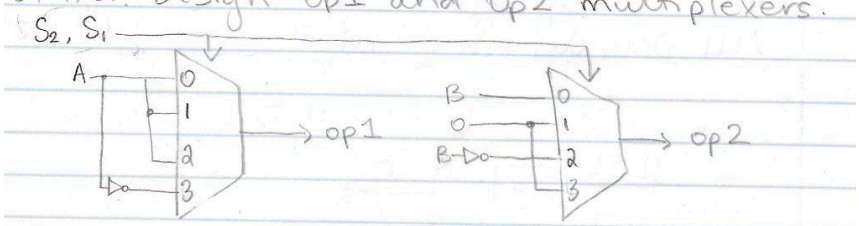

Figure 3: Design of operation 1 and 2 used in the Arithmetic Circuit derived in lab preparation

5.4.4: Complete the AC Design.

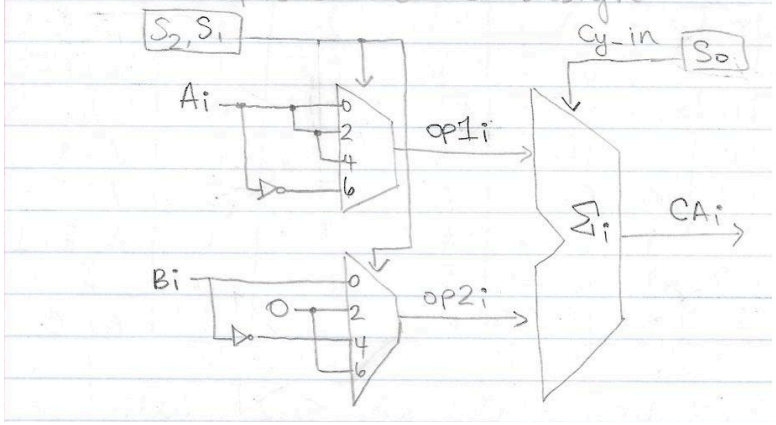

Figure 4: 1-bit module design of the Arithmetic Circuit derived in lab preparation

Status bit equations

$V = [S3(\bar{S_0}S_1S_2)(A3 \oplus A2)] + [(C02 \oplus C03)\bar{S3}]$

$Cy = \bar{S3}C03$

S = ALU3 aka most significant ALU output

$Z = (\overline{ALU0} \cdot \overline{ALU1} \cdot \overline{ALU2} \cdot \overline{ALU3})$

Figure 5: Equations of the status bits derived in lab preparation

Table 1: Sequences of micro-operations to simulate

Cycle	RTL Micro-operation	S3	S2	S1	S0	A->op1	B->op2	C	V,Z,S,Cy	S_{16}	A_{16}	B_{16}	C_{16}	St_{16}
1	C ← A ∧ B’	1	1	0	1	1010	0011	1000	0010	D	A	3	8	2
2	C ← ashl A	1	1	1	0	0110	XXXX	1100	1010	E	6	X	C	A
3	C ← A + B	0	0	0	0	0011	0101	1000	1010	0	3	5	8	A
4	C ← A + 1	0	0	1	1	1100	XXXX	1101	0010	3	C	X	D	2
5	C ← A ⊕ B	1	1	0	0	0011	0101	0110	0000	C	3	5	6	0
6	C ← A’	0	1	1	0	1010	XXXX	0101	0000	6	A	X	5	0
7	C ← 0000	1	0	0	0	XXXX	XXXX	0000	0100	8	X	X	0	4
8	C ← A + B’ + 1	0	1	0	1	0101	0011	0010	0001	5	5	3	2	1
9	C ← A	0	0	1	0	1110	XXXX	1110	0010	2	E	X	E	2
10	C ← A’ + 1	0	1	1	1	0110	XXXX	1010	0010	7	6	X	A	2
11	C ← A ∧ B	1	0	1	0	0101	0011	0001	0000	A	5	3	1	0
12	C ← A + B + 1	0	0	0	1	0001	0010	0100	0000	1	1	2	4	0
13	C ← ashr A	1	1	1	1	1101	XXXX	0110	0000	F	D	X	6	0
14	C ← A + B’	0	1	0	0	0110	0101	0000	0101	4	6	5	0	5
15	C ← 1111	1	0	0	1	XXXX	XXXX	1111	0010	9	X	X	F	2
16	C ← A ∨ B	1	0	1	1	1100	1010	1110	0010	B	C	A	E	2

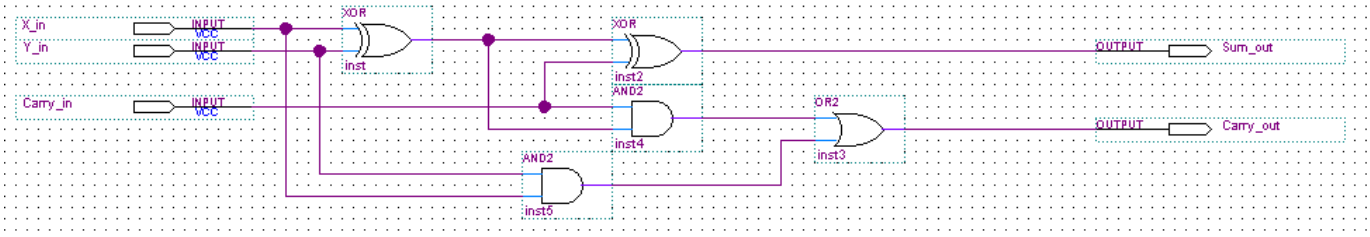

Figure 6: 1-bit Full-Adder circuit design

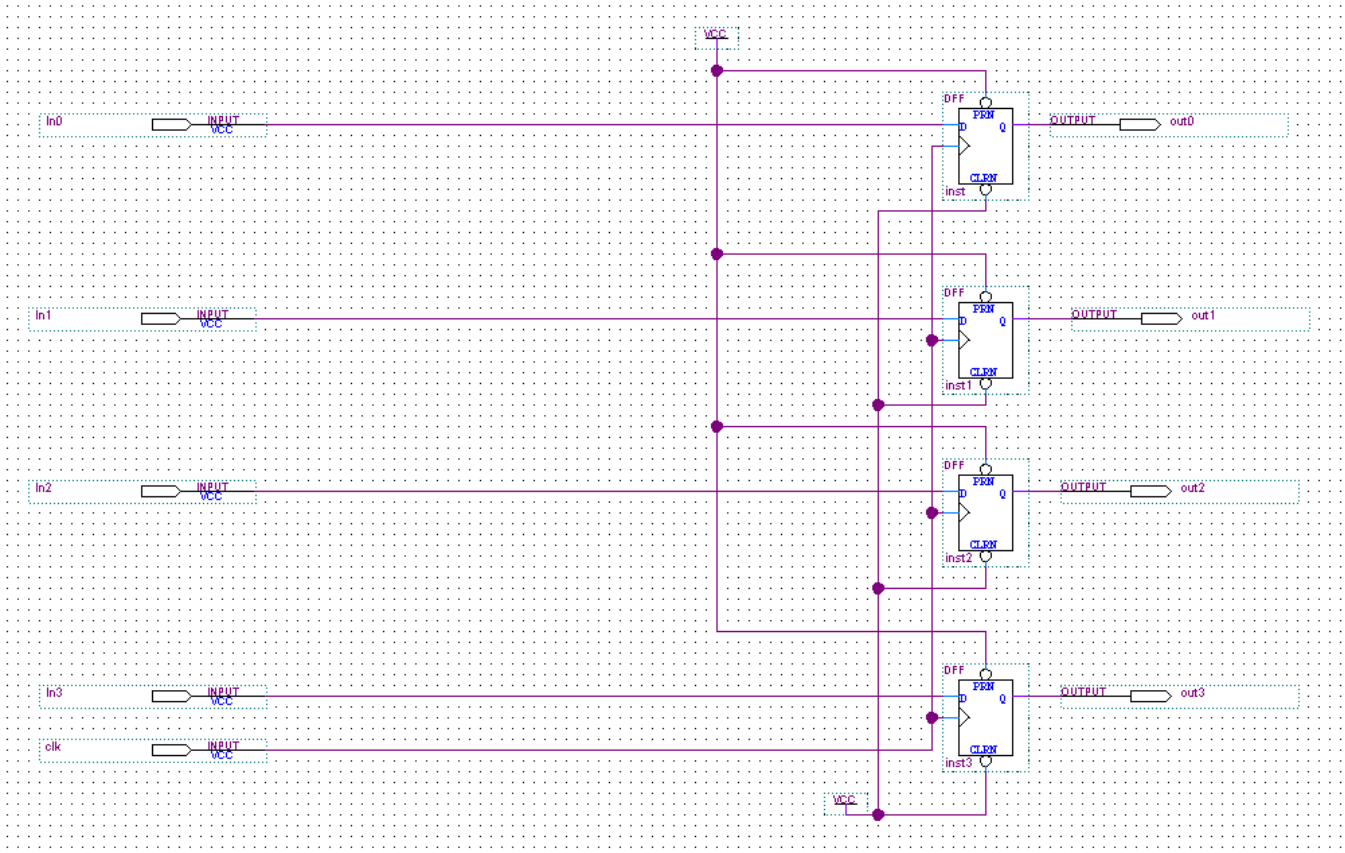

Figure 7: 4-bit Register circuit design

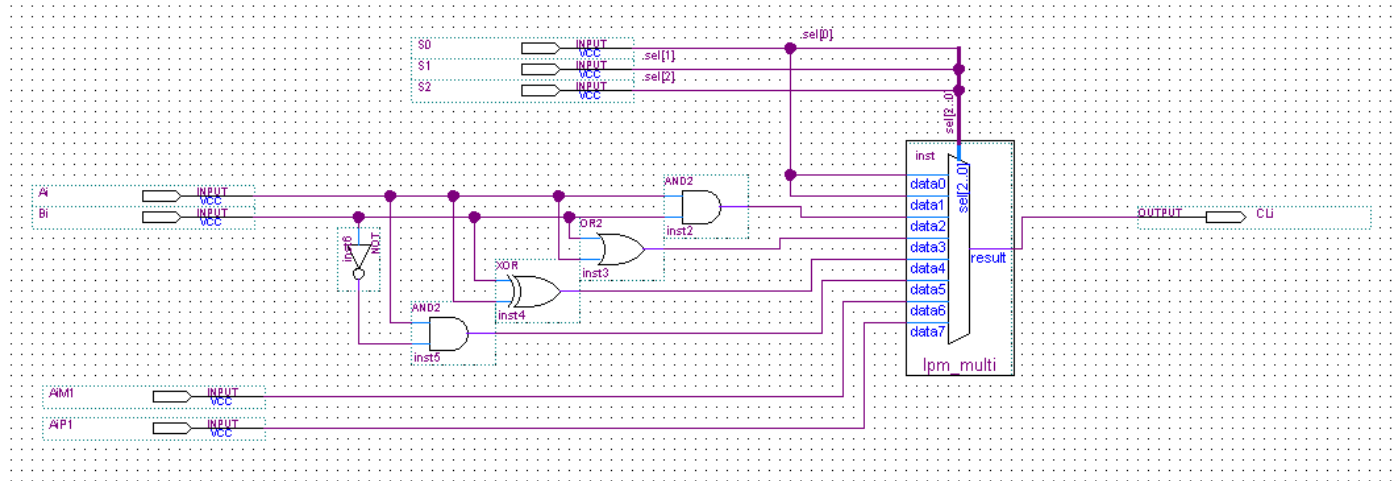

Figure 8: 1-bit Logic and Shift Circuit design

Fitter Summary	
Fitter Status	Successful - Thu Oct 13 13:28:33 2022
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	Lab3
Top-level Entity Name	logiccircuit1bitsLmp
Family	Cyclone IV E
Device	EP4CE115F29C7
Timing Models	Final
Total logic elements	5 / 114,480 (< 1 %)
Total combinational functions	5 / 114,480 (< 1 %)
Dedicated logic registers	0 / 114,480 (0 %)
Total registers	0
Total pins	8 / 529 (2 %)
Total virtual pins	0
Total memory bits	0 / 3,981,312 (0 %)
Embedded Multiplier 9-bit elements	0 / 532 (0 %)
Total PLLs	0 / 4 (0 %)

Figure 9: 1-bit Logic and Shift Circuit fitter summary

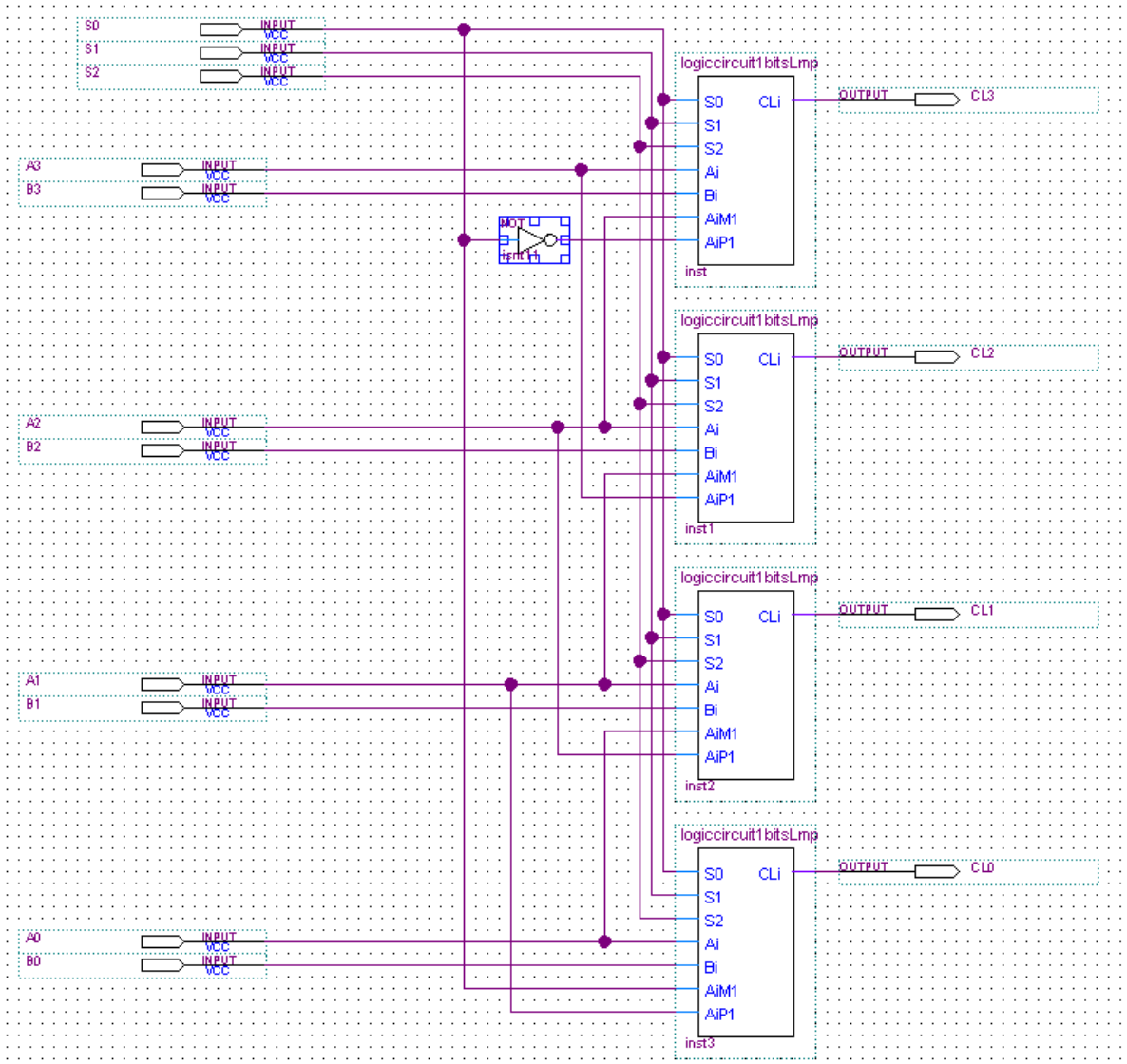

Figure 10: 4-bit Logic and Shift Circuit diagram

Fitter Summary	
Fitter Status	Successful - Thu Oct 13 12:37:10 2022
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	Lab3
Top-level Entity Name	logiccircuit4bits
Family	Cyclone IV E
Device	EP4CE115F29C7
Timing Models	Final
Total logic elements	26 / 114,480 (< 1 %)
Total combinational functions	26 / 114,480 (< 1 %)
Dedicated logic registers	0 / 114,480 (0 %)
Total registers	0
Total pins	15 / 529 (3 %)
Total virtual pins	0
Total memory bits	0 / 3,981,312 (0 %)
Embedded Multiplier 9-bit elements	0 / 532 (0 %)
Total PLLs	0 / 4 (0 %)

Figure 11: 4-bit Logic and Shift Circuit fitter summary

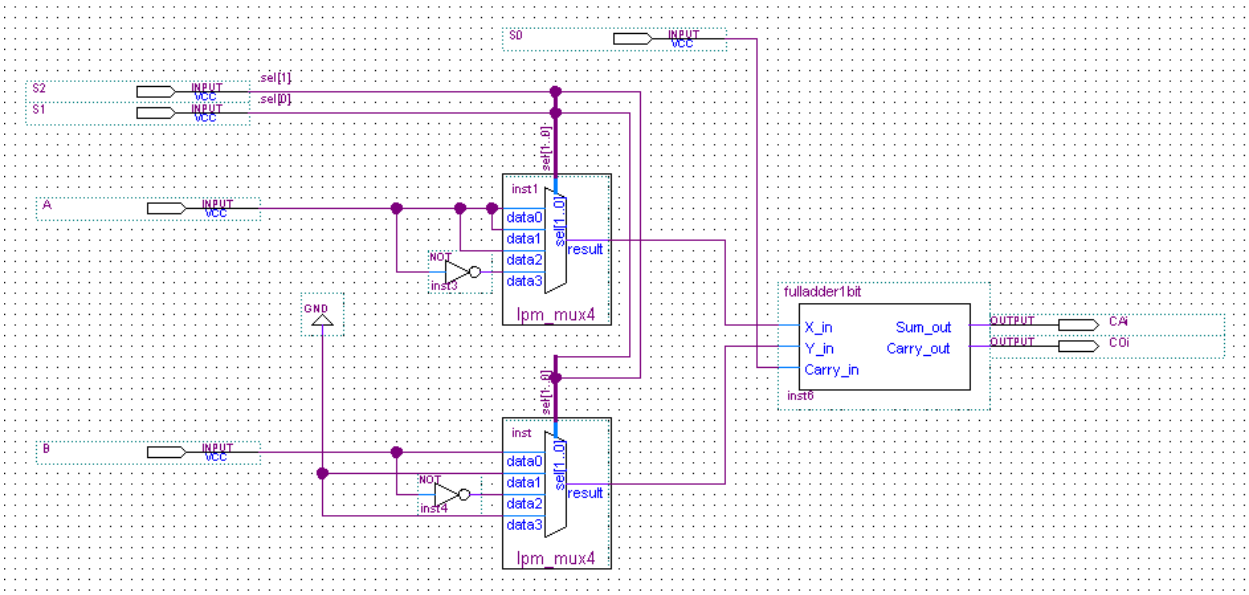

Figure 12: 1-bit Arithmetic Circuit design

Fitter Summary	
Fitter Status	Successful - Thu Nov 03 13:16:09 2022
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	Lab3
Top-level Entity Name	arithcircuit1bit
Family	Cyclone IV E
Device	EP4CE115F29C7
Timing Models	Final
Total logic elements	4 / 114,480 (< 1 %)
Total combinational functions	4 / 114,480 (< 1 %)
Dedicated logic registers	0 / 114,480 (0 %)
Total registers	0
Total pins	7 / 529 (1 %)
Total virtual pins	0
Total memory bits	0 / 3,981,312 (0 %)
Embedded Multiplier 9-bit elements	0 / 532 (0 %)
Total PLLs	0 / 4 (0 %)

Figure 13: 1-bit Arithmetic Circuit fitter summary

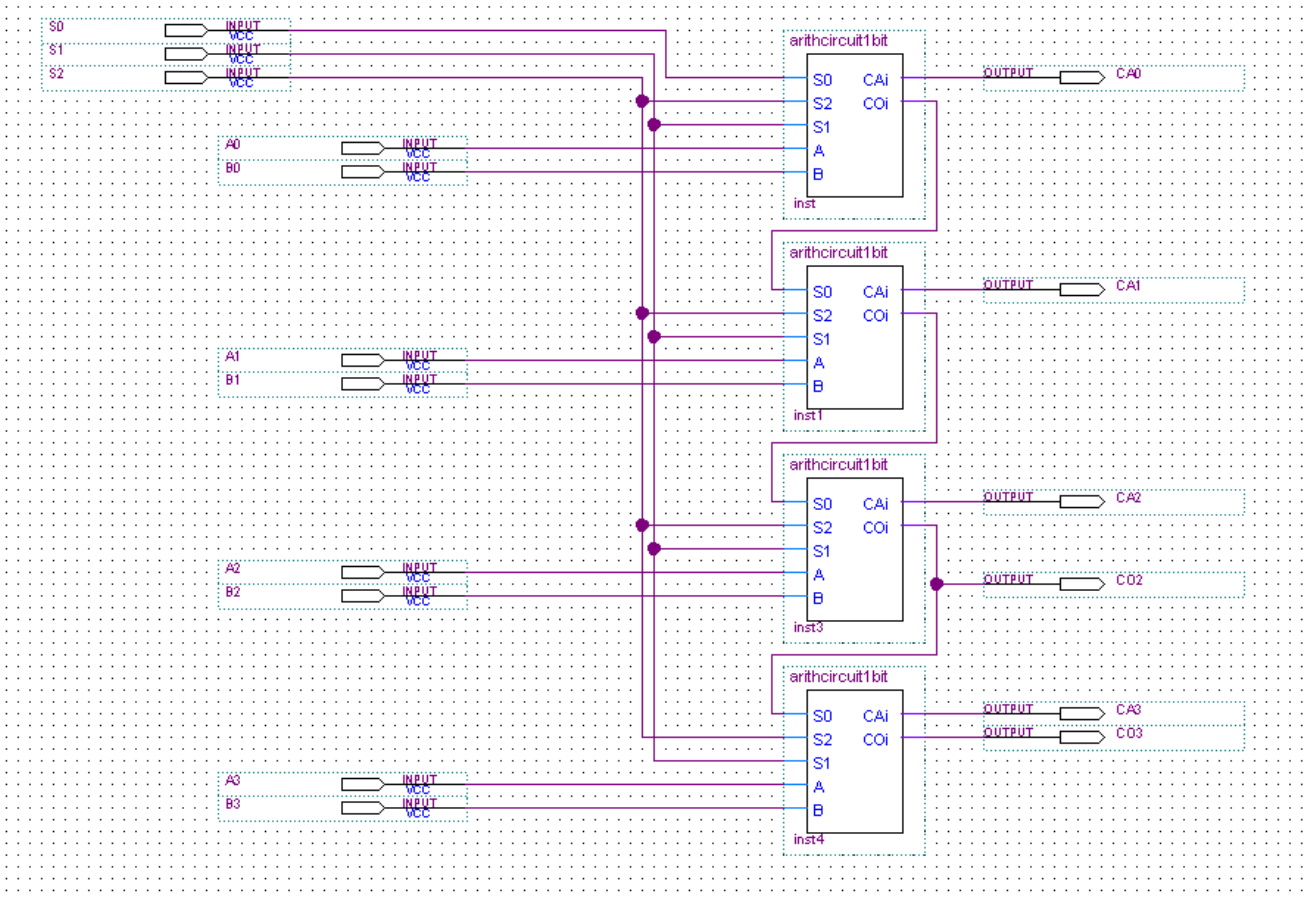

Figure 14: 4-bit Arithmetic Circuit design

Fitter Summary	
Fitter Status	Successful - Thu Nov 03 13:32:02 2022
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	Lab3
Top-level Entity Name	arithcircuit4bits
Family	Cyclone IV E
Device	EP4CE115F29C7
Timing Models	Final
Total logic elements	16 / 114,480 (< 1 %)
Total combinational functions	16 / 114,480 (< 1 %)
Dedicated logic registers	0 / 114,480 (0 %)
Total registers	0
Total pins	17 / 529 (3 %)
Total virtual pins	0
Total memory bits	0 / 3,981,312 (0 %)
Embedded Multiplier 9-bit elements	0 / 532 (0 %)
Total PLLs	0 / 4 (0 %)

Figure 15: 4-bit Arithmetic Circuit fitter summary

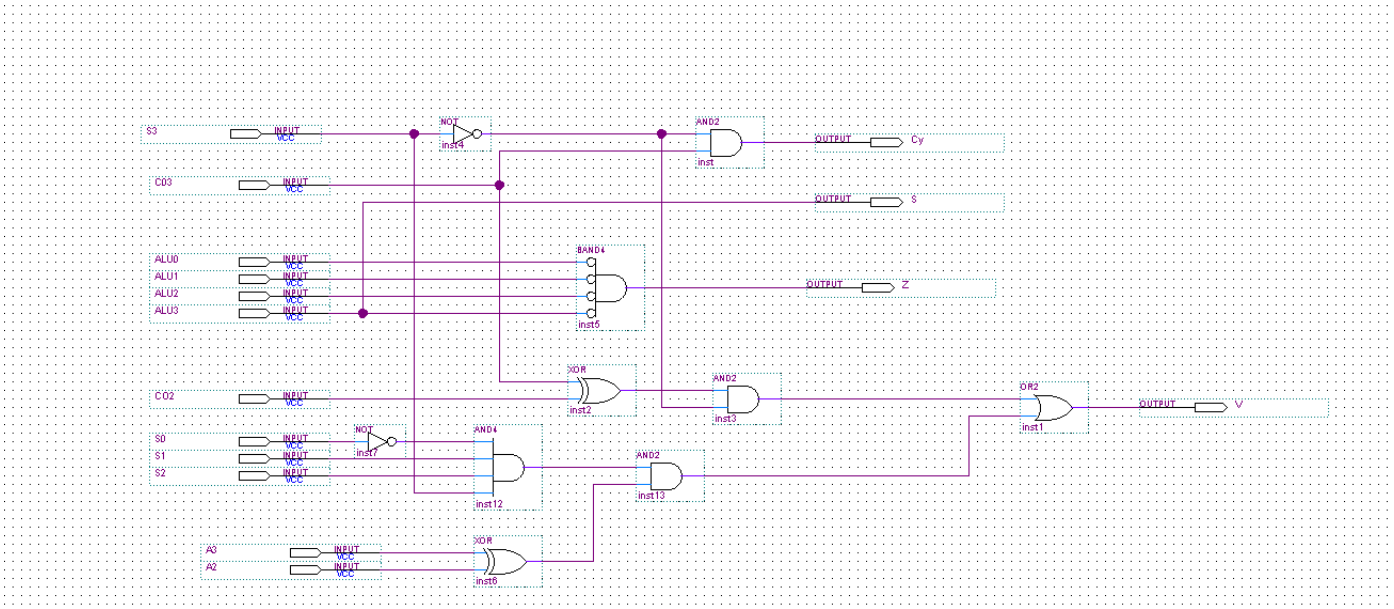

Figure 16: 4-bit State Register circuit diagram

Fitter Summary	
Fitter Status	Successful - Thu Nov 03 12:03:10 2022
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	Lab3
Top-level Entity Name	state4bits
Family	Cyclone IV E
Device	EP4CE115F29C7
Timing Models	Final
Total logic elements	5 / 114,480 (< 1 %)
Total combinational functions	5 / 114,480 (< 1 %)
Dedicated logic registers	0 / 114,480 (0 %)
Total registers	0
Total pins	16 / 529 (3 %)
Total virtual pins	0
Total memory bits	0 / 3,981,312 (0 %)
Embedded Multiplier 9-bit elements	0 / 532 (0 %)
Total PLLs	0 / 4 (0 %)

Figure 17: 4-bit State Register fitter summary

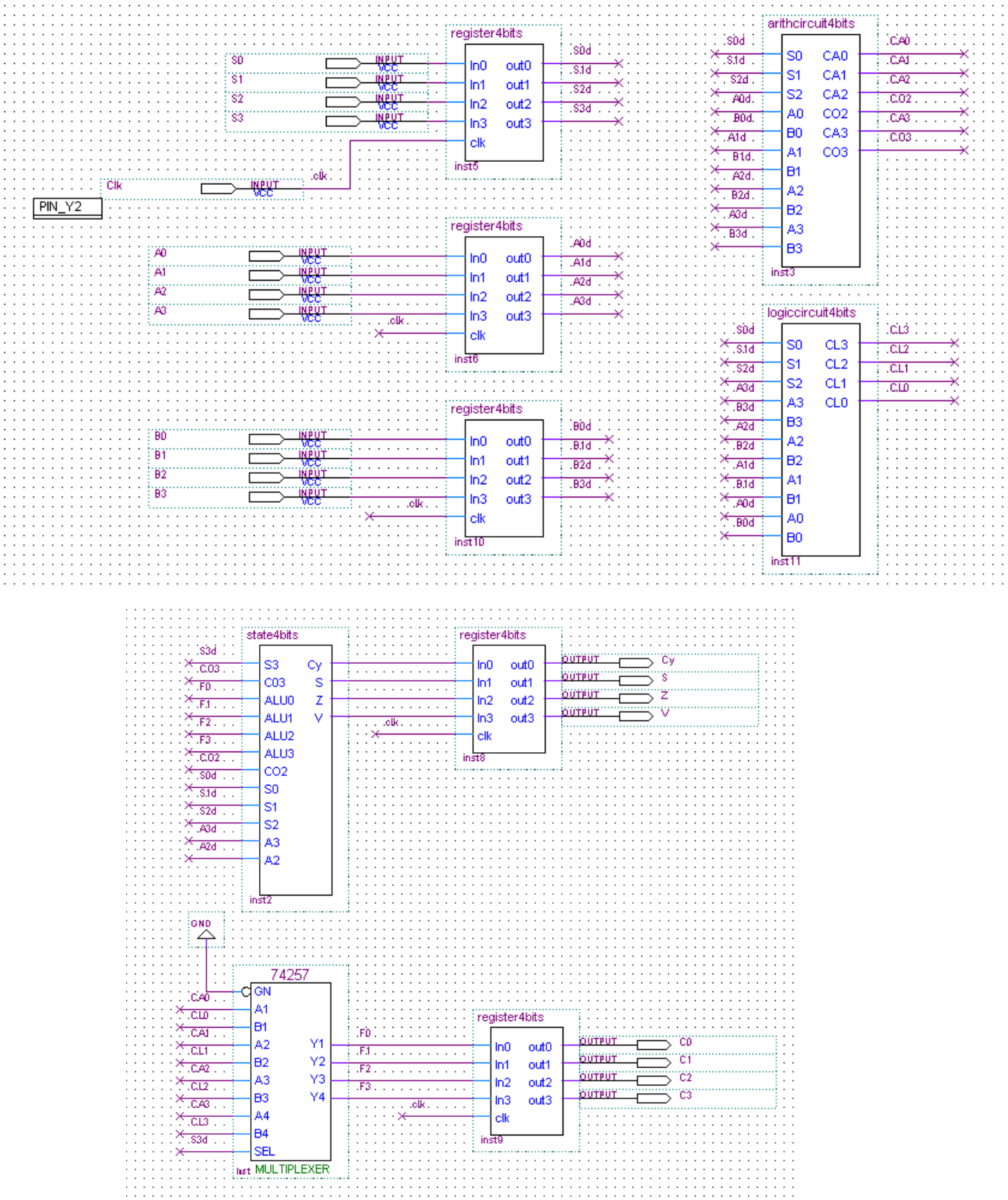

Figure 18: ALU circuit design

Flow Summary	
Flow Status	Successful - Thu Nov 03 13:34:51 2022
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	Lab3
Top-level Entity Name	Lab3top
Family	Cyclone IV E
Device	EP4CE115F29C7
Timing Models	Final
Total logic elements	43 / 114,480 (< 1 %)
Total combinational functions	41 / 114,480 (< 1 %)
Dedicated logic registers	20 / 114,480 (< 1 %)
Total registers	20
Total pins	21 / 529 (4 %)
Total virtual pins	0
Total memory bits	0 / 3,981,312 (0 %)
Embedded Multiplier 9-bit elements	0 / 532 (0 %)
Total PLLs	0 / 4 (0 %)

Figure 19: ALU fitter summary

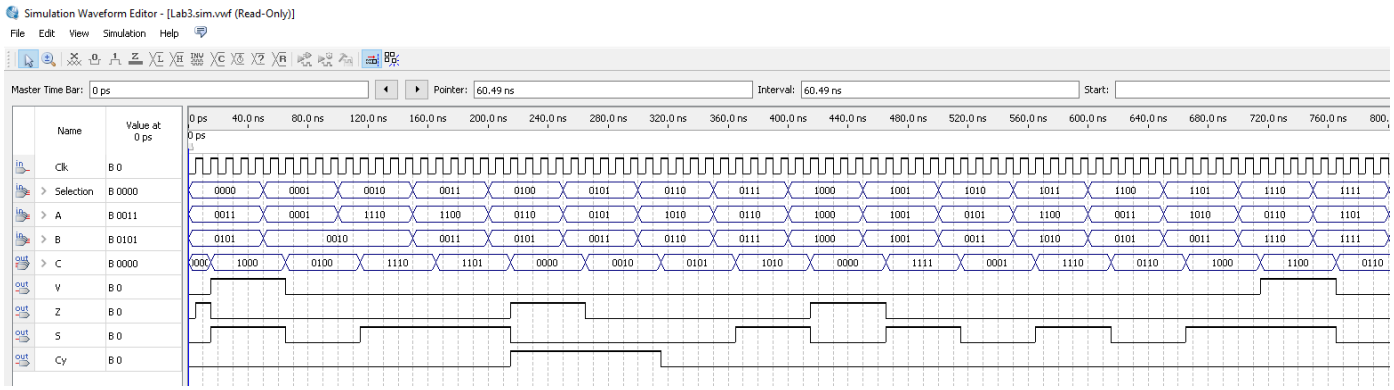

Figure 20: ALU waveform simulation using the sequences from Table 1