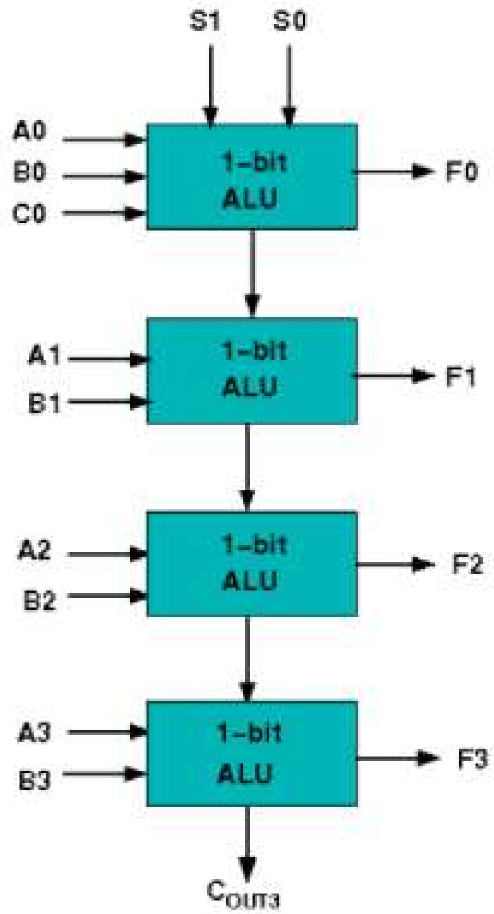
Experiment No.10
Implement ALU design.
Name: Singh Rahul
Roll Number: 56
Date of Performance:
Date of Submission:

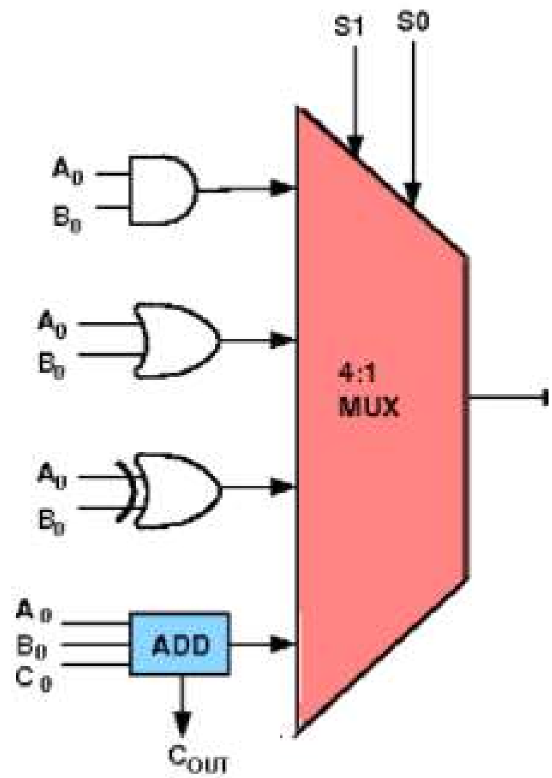
Objective : Objective of 4 bit arithmetic logic unit (with AND, OR, XOR, ADD operation):

- To understand behaviour of arithmetic logic unit from working module.
- To Design an arithmetic logic unit for given parameter.

Theory:

ALU or Arithmetic Logical Unit is a digital circuit to do arithmetic operations like addition, subtraction,division, multiplication and logical oparations like and, or, xor, nand, nor etc. A simple block diagram of a 4 bit ALU for operations and,or,xor and Add is shown here :





The 4-bit ALU block is combined using 4 1-bit ALU block

Design Issues :

The circuit functionality of a 1 bit ALU is shown here, depending upon the control signal S1 and S0 the circuit operates as follows:

for Control signal S1 = 0 , S0 = 0, the output is A And B,
for Control signal S1 = 0 , S0 = 1, the output is A Or B,
for Control signal S1 = 1 , S0 = 0, the output is A Xor B,
for Control signal S1 = 1 , S0 = 1, the output is A Add B.

The truth table for 16-bit ALU with capabilities similar to 74181 is shown here:
Required functionality of ALU (inputs and outputs are active high)

MODE SELECT	F_N FOR ACTIVE HIGH OPERANDS	
INPUTS	LOGIC	ARITHMETIC (NOTE 2)

S3	S2	S1	S0	(M = H)	(M = L) (Cn=L)
L	L	L	L	A'	A
L	L	L	H	A'+B'	A+B
L	L	H	L	A'B	A+B'
L	L	H	H	Logic 0	minus 1
L	H	L	L	(AB)'	A plus AB'
L	H	L	H	B'	(A + B) plus AB'
L	H	H	L	$A \oplus B$	A minus B minus 1
L	H	H	H	AB'	AB minus 1
H	L	L	L	A'+B	A plus AB
H	L	L	H	$(A \oplus B)'$	A plus B
H	L	H	L	B	(A + B') plus AB
H	L	H	H	AB	AB minus 1
H	H	L	L	Logic 1	A plus A (Note 1)
H	H	L	H	A+B'	(A + B) plus A
H	H	H	L	A+B	(A + B') plus A
H	H	H	H	A	A minus 1

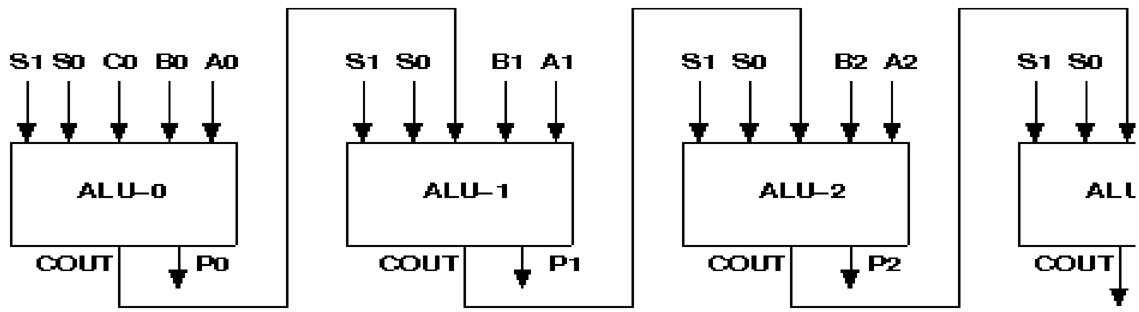
Procedure

- Start the simulator as directed.This simulator supports 5-valued logic.
- To design the circuit we need 4 1-bit ALU, 11 Bit switch (to give input,which will toggle its value with a double click), 5 Bit displays (for seeing output), wires.
- The pin configuration of a component is shown whenever the mouse is hovered on any canned component of the palette. Pin numbering starts from 1 and from the bottom left corner (indicating with the circle) and increases anticlockwise.
- For 1-bit ALU input A0 is in pin-9,B0 is in pin-10, C0 is in pin-11 (this is input carry), for selection of operation, S0 is in pin-12, S1 is in pin-13, output F is in pin-8 and output carry is pin-7
- Click on the 1-bit ALU component (in the Other Component drawer in the pallet) and then click on the position of the editor window where you want to add the component (no drag and drop, simple click will serve the purpose), likewise add 3 more 1-bit ALU (from the Other Component drawer in the pallet), 11 Bit switches and 5 Bit Displays (from Display and Input drawer of the pallet,if it is not seen scroll down in the drawer), 3 digital display and 1 bit Displays (from Display and Input drawer of the pallet,if it is not seen scroll down in the drawer)
- To connect any two components select the Connection menu of Palette, and then click on the Source terminal and click on the target terminal. According to the

circuit diagram connect all the components. Connect the Bit switches with the inputs and Bit displays component with the outputs. After the connection is over click the selection tool in the pallete.

- See the output, in the screenshot diagram we have given the value of S1 S0=11 which will perform add operation and two number input as A0 A1 A2 A3=0010 and B0 B1 B2 B3=0100 so get output F0 F1 F2 F3=0110 as sum and 0 as carry which is indeed an add operation.you can also use many other combination of different values and check the result. The operations are implemented using the truth table for 4 bit ALU given in the theory.

Circuit diagram of 4 bit ALU:

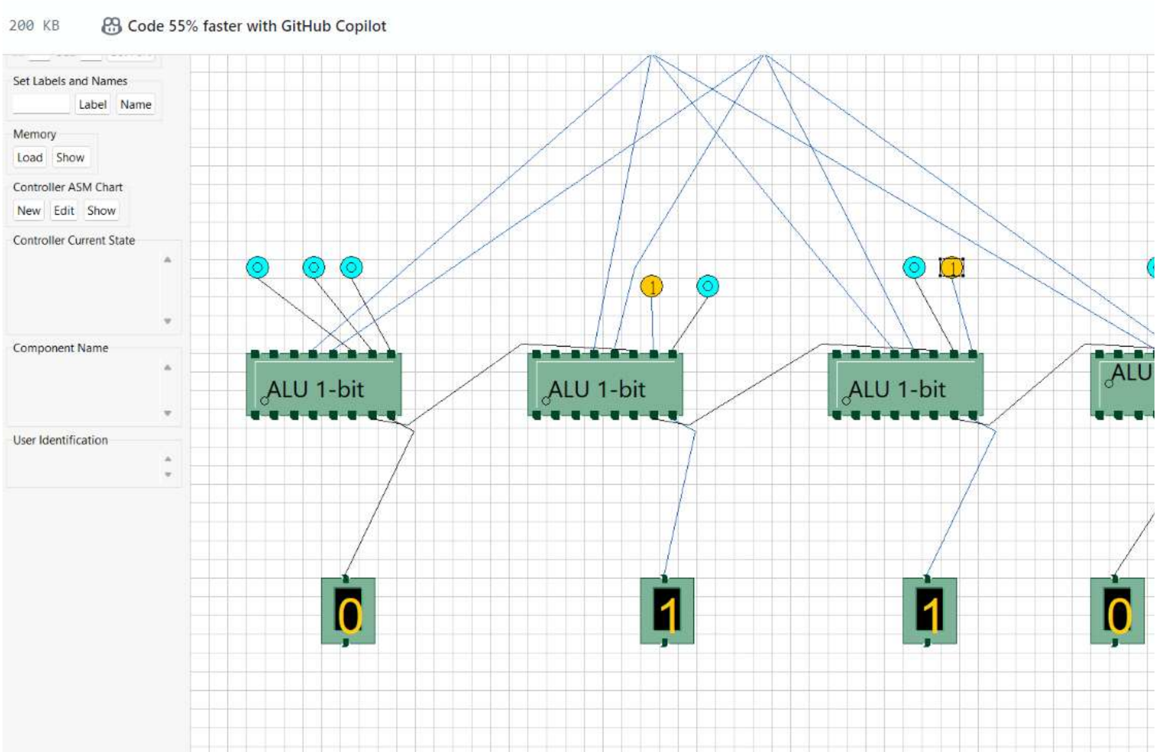


Components required :

To build any 4 bit ALU, we need :

- AND gate, OR gate, XOR gate
- Full Adder,
- 4-to-1 MUX
- Wires to connect.

Screenshots of ALU design:



Conclusion:

In conclusion, the experiment conducted on ALU design in Logisim has provided valuable insights into the fundamental aspects of arithmetic and logic unit functionality. Through rigorous testing and analysis, we have successfully designed and implemented an ALU that demonstrates efficient operation and accuracy in performing various arithmetic and logical operations. This experiment has not only enhanced our understanding of digital logic design but also highlighted the importance of meticulous planning and testing in creating reliable computing components. These findings underscore the significance of ALUs in modern computer architecture and their pivotal role in processing and executing instructions.