Lab 02: Introduction to the Software Tool Logisim, and Implementation of Arithmetic Logic Unit (ALU) in Logisim

Objectives:

After completing this experiment, students will be able to:

- Understand the essential software tools including LogiSim.
- Explore LogiSim as a logic simulator for designing and simulating digital circuits.
- Understand the basic functions of an ALU.
- Use different components like MUX, ADDER, SUBTRACTOR and basic Logical Operations in LogiSim.
- Design and Implement 1-, 8-, 16- and 32-bit ALU in LogiSim.

LogiSim

Logisim is an educational tool for designing and simulating digital logic circuits. With its simple toolbar interface and simulation of circuits as they are built, it is simple enough to facilitate learning the most basic concepts related to logic circuits. With the capacity to build larger circuits from smaller subcircuits, and to draw bundles of wires with a single mouse drag, Logisim can be used (and is used) to design and simulate entire CPUs for educational purposes.

Students at colleges and universities around the world use Logisim for a variety of purposes, including:

1. A module in general-education computer science surveys
2. A unit in sophomore-level computer organization courses
3. Over a full semester in upper-division computer architecture courses

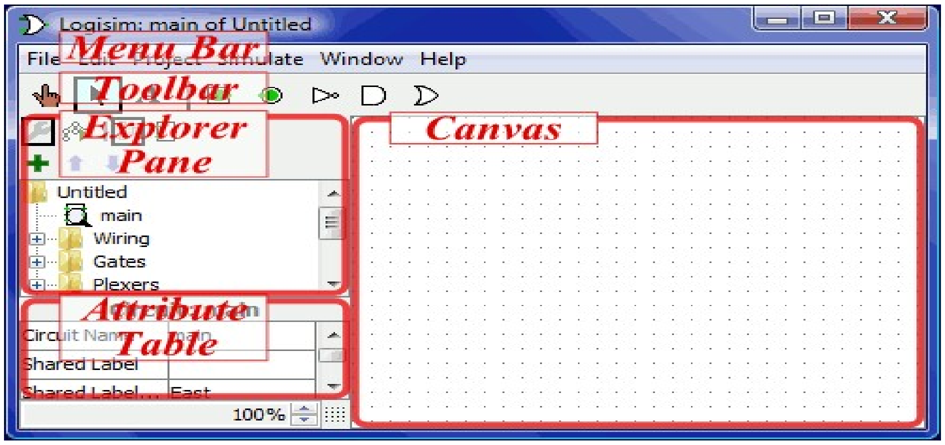


Figure 1.2: Logisim Interface

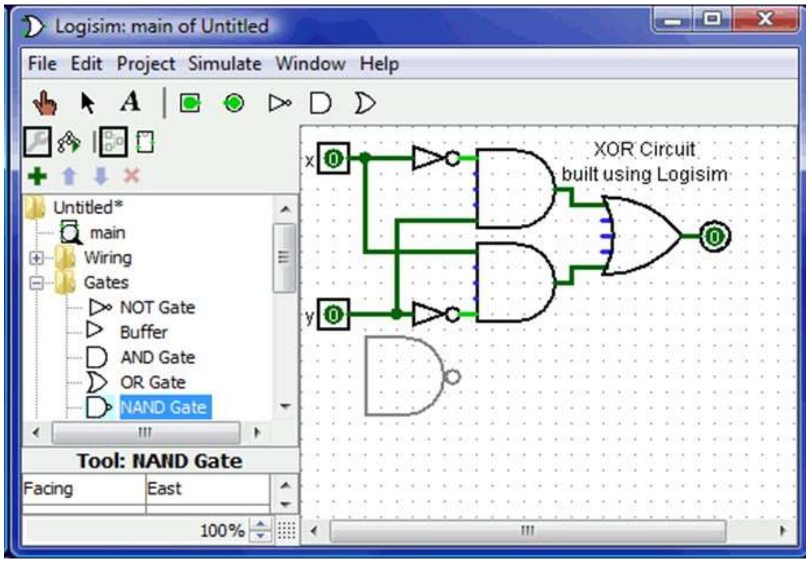
The explorer pane

Logisim organizes tools into libraries. They are displayed as folders in the explorer pane; to access a library's components, you have only to double-click the corresponding folder. Below, I have opened the Gates library and selected the NAND tool from it. You can see that Logisim now stands ready to add NAND gates into the circuit.

If you look through the choices in the Gates library, you'll notice that there was no need for us to develop a XOR circuit earlier: It's built into Logisim.

When you create a project, it automatically includes several libraries:

1. Wiring: Components that interact directly with wires.
2. Gates: Components that perform simple logic functions.
3. Plexers: More complex combinational components, like multiplexers and decoders.
4. Arithmetic: Components that perform arithmetic.
5. Memory: Components that remember data, like flip-flops, registers, and RAM.
6. I/O: Components that exist for the purpose of interacting with the user.
7. Base: Tools that are integral to using Logisim, though you probably won't need to dig into this library very often.



Tool: NAND Gate

Facing	East

Figure 1.3: Explorer Pane

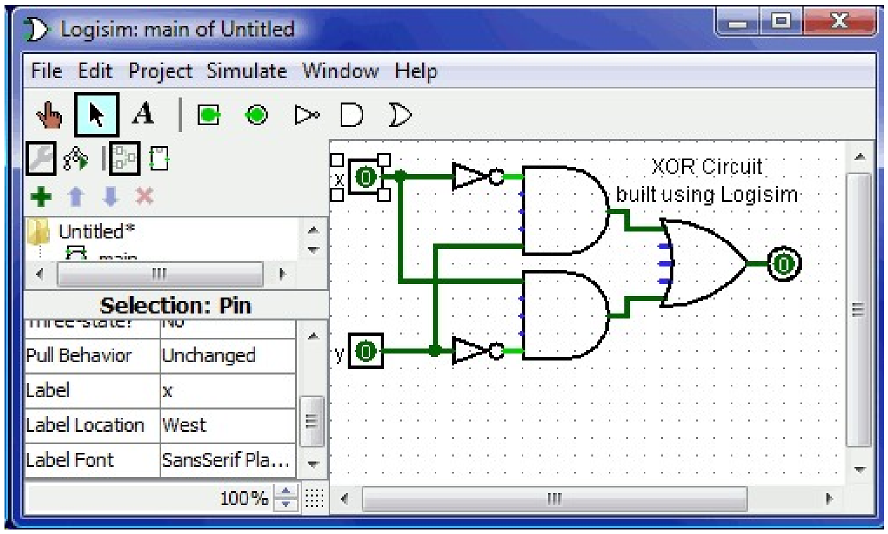
The attribute table

Many components have attributes, which are properties for configuring how the component behaves or appears. The attribute table is for viewing and displaying a component's attribute values.

To select which component's attributes you wish to view, click the component using the Edit tool ( ). (You can also right-click (or control-click) the component and choose Show Attributes from the popup menu. Also, manipulating a component via the Poke tool ( ) or the Text tool (*A*) will display that component's attributes.)

The screen shot below demonstrates what things look like after selecting the upper input of our XOR circuit and scrolling down to view the Label Font attribute.

To modify an attribute value, click on the value. The interface for modifying the attribute will depend on which attribute you are changing; in the case of the Label Font attribute, a dialog box will appear for selecting the new font; but some attributes (like Label) will allow you to edit the value as a text field, while others (like Label Location) will display a drop-down menu from which to select the value.


Logisim: main of Untitled
File Edit Project Simulate Window Help
Untitled*
main
XOR Circuit
built using Logisim
x
y
100%

Selection: Pin

Three-state?	No
Pull Behavior	Unchanged
Label	x
Label Location	West
Label Font	SansSerif Pla...

Figure 1.4: The Attribute Table

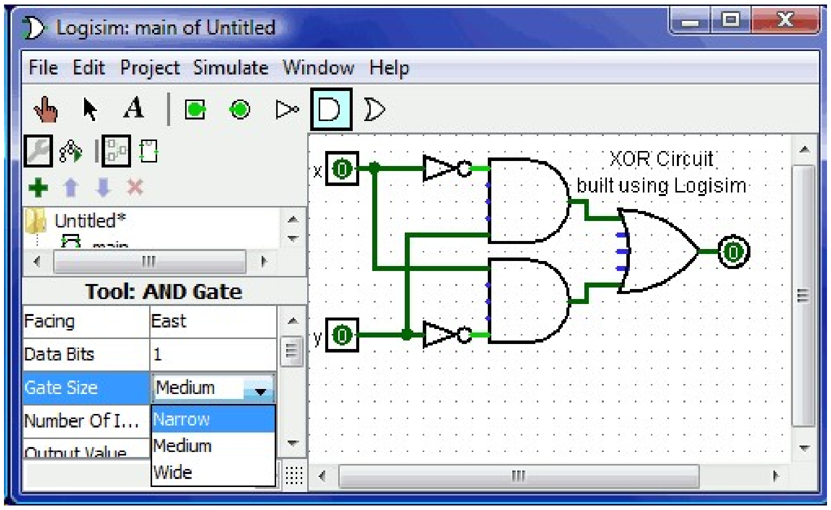
Tool attributes

Every tool for adding components to a circuit also has a set of attributes, which are imparted to the components created by the tool, although the components' attributes may be changed later without affecting the tool's attributes. When you select a tool, Logisim will change the attribute table to display that tool's attributes.

For example, suppose we want to create smaller AND gates. Right now, each time we select the AND tool, it creates a large AND gate. But if we edit the Gate Size attribute just after selecting the tool (before placing its AND gate into the circuit), we'll be changing the attributes for the tool, so that future AND gates added using the tool would be narrow instead.

With some tools, the tool's icon reflects some of the attributes' values. One example of this is the Pin tool, whose icon faces the same way as its Facing attribute says.

The tools in the toolbar each have a separate attribute set from the corresponding tools in the explorer pane. Thus, even though we changed the toolbars AND tool to create narrow AND gates, the AND tool in the Gates library will still create wide AND gates unless you change its attributes too.


Logisim: main of Untitled
File Edit Project Simulate Window Help
Untitled*
main
XOR Circuit
built using Logisim
x
y
Narrow
Medium
Wide

Tool: AND Gate

Facing	East
Data Bits	1
Gate Size	Medium
Number Of I...	
Output Value	

Figure 1.5: The Attribute Table

How to Get Started:

Download and Install:

- Visit the Logisim website and download the latest version suitable for your operating system. Install it following the provided instructions.

Create a New Circuit:

- Open Logisim and create a new project. Start designing your circuit by adding components from the toolbar.

Wire Components:

- Connect components by dragging wires between them. Establish logical connections according to your design.

Simulation:

- Use the simulation tool to test your circuit. Provide input values and observe the output behavior.

Save and Export:

- Save your project for future editing. You can also export your circuit as a file or an image.

Arithmetic Logic Unit (ALU)

Background Theory

An Arithmetic Logic Unit (ALU) is a fundamental component in digital systems, playing a crucial role in performing arithmetic and logical operations. It serves as the computational heart of a microprocessor or a digital circuit, executing mathematical calculations and logical comparisons that are essential for processing data.

Functions of an ALU

Addition

The primary arithmetic function of an ALU is addition. It can add binary numbers, which is a fundamental operation in any digital computing system. The ALU takes two binary inputs and produces their sum as the output.

Subtraction

Subtraction is closely tied to addition in digital systems. An ALU can perform subtraction by employing a mechanism known as two's complement arithmetic. This allows for efficient addition of negative numbers, and the ALU can effectively subtract one binary number from another.

Logical AND Operations

ALUs also perform logical operations, including the AND operation. In the AND operation, each bit of the output is the logical AND of the corresponding bits of the input. It results in a '1' only if both input bits are '1'; otherwise, the output is '0'.

Logical OR Operations

Another essential logical operation is the OR operation. The OR operation produces a '1' if at least one of the corresponding input bits is '1'. It results in '0' only when both input bits are '0'.

Logical XOR Operations

The XOR operation is a logical operation where the output is '1' if the input bits are different and '0' if they are the same. XOR is frequently used in various applications, including data comparison and error detection.

Importance in Digital Systems

Central Processing Unit (CPU)

ALUs are a critical component of a CPU, where they perform the arithmetic and logical operations necessary for executing program instructions. The speed and efficiency of an ALU directly impact on the overall performance of the CPU.

Data Processing

In digital systems, ALUs are responsible for manipulating data, making them indispensable in applications ranging from basic calculators to complex data processing units in computers and embedded systems.

Control Flow

ALUs play a key role in controlling the flow of operations within a digital system. They enable decision-making processes, comparisons, and conditional branching based on the results of arithmetic and logical operations.

Mathematical Computations

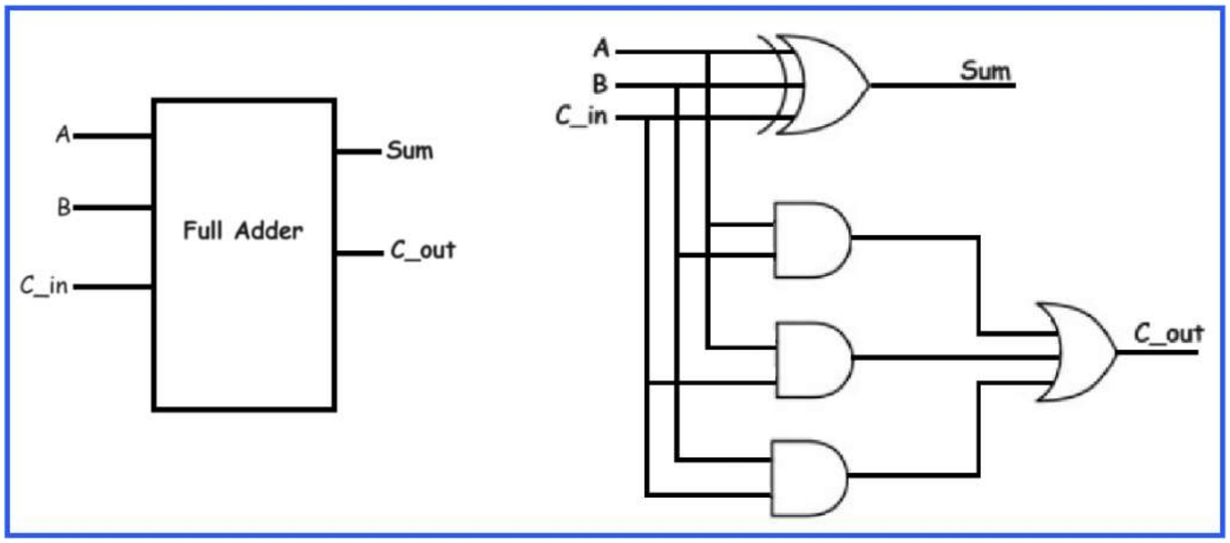
ALUs are crucial for performing mathematical computations, making them integral to applications such as scientific calculations, simulations, and numerical analysis.

So, the Arithmetic Logic Unit is a fundamental building block of digital systems, providing the computational capabilities necessary for a wide range of applications in the field of electronics and computing. Its ability to perform both arithmetic and logical operations makes it a versatile and indispensable component in the world of digital design and information processing.

Lab Tasks:

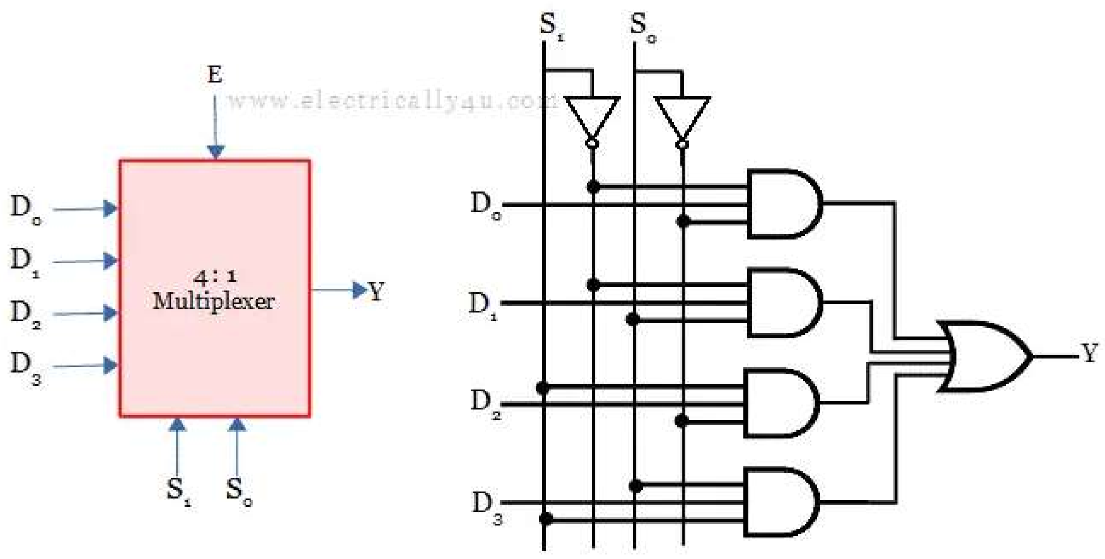
Lab Task-1: **Implement and Verify All the basic Logic Gates in Logisim.**

Lab Task-2: **Implement and Verify the Following Full Adder in Logisim**

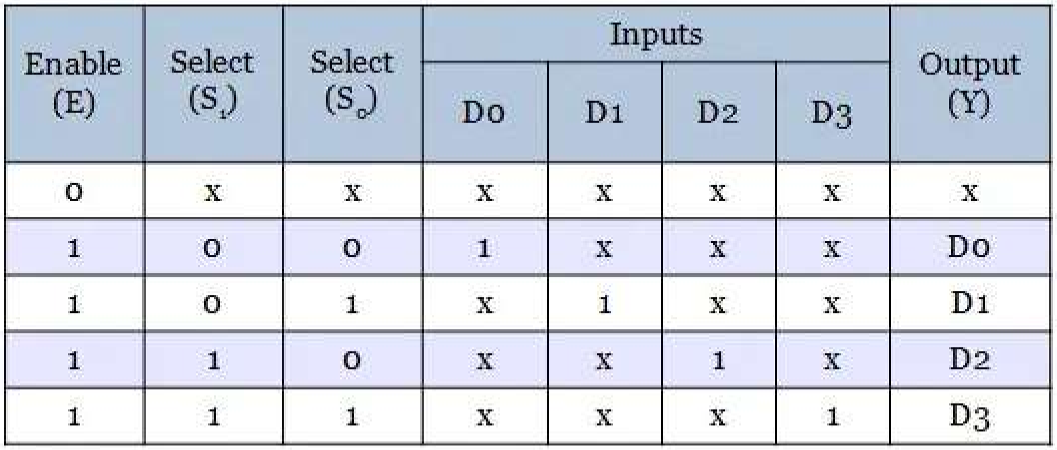


Lab Task-3: **Implement and Verify the Following 4 to 1 mux in Logisim**

For 4 to 1 multiplexer, 4 data inputs, 2 selection lines and 1 output is needed. The block diagram and circuit diagram is shown below.



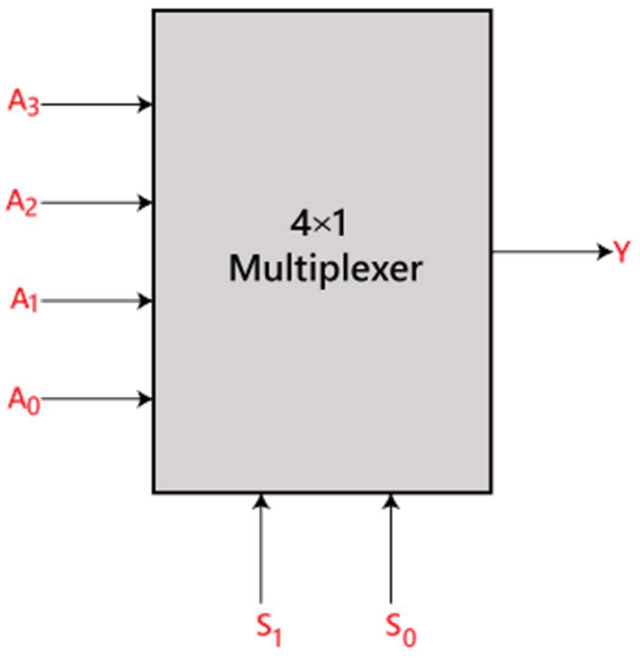
Block diagram and logic circuit of 4 to 1 mux

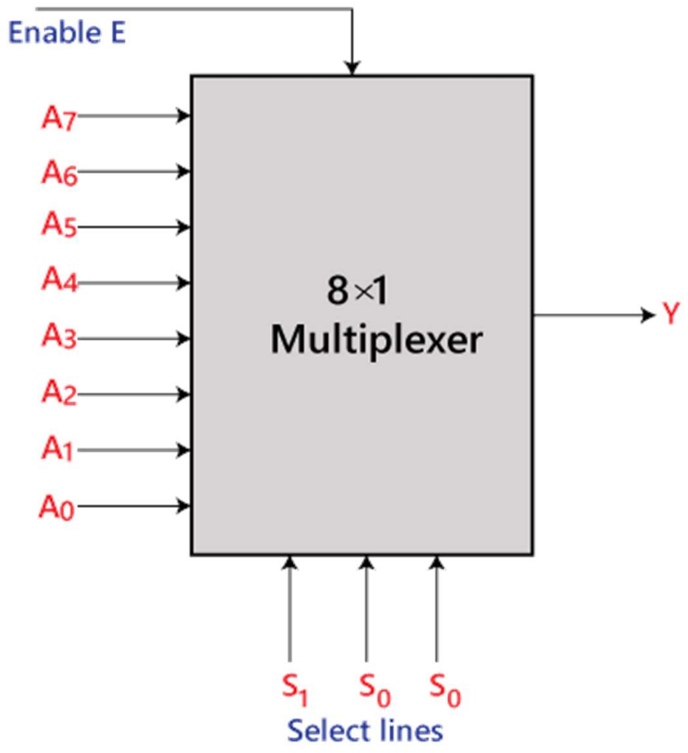


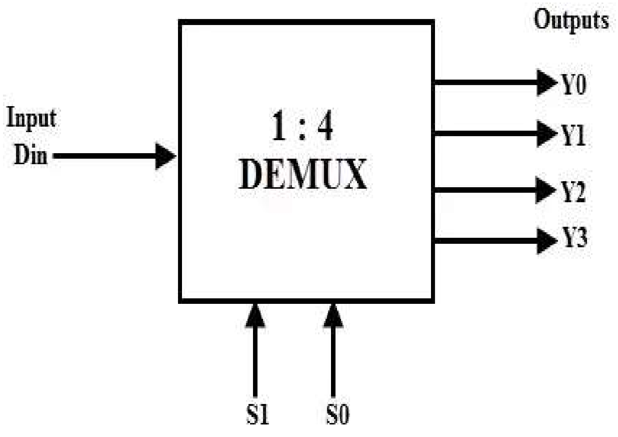
Enable (E)	Select (S_1)	Select (S_0)	Inputs				Output (Y)
			D0	D1	D2	D3	
0	x	x	x	x	x	x	x
1	0	0	1	x	x	x	D0
1	0	1	x	1	x	x	D1
1	1	0	x	x	1	x	D2
1	1	1	x	x	x	1	D3

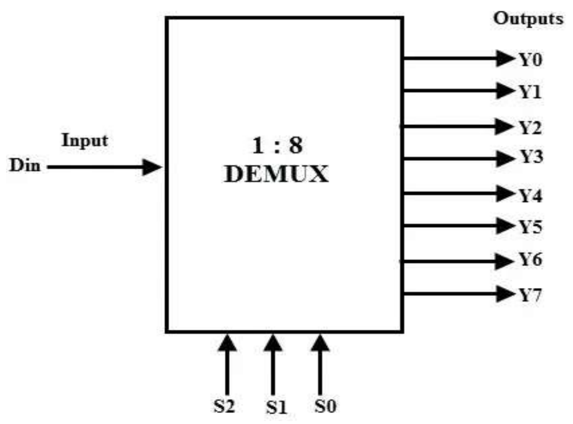
Function table of 4 to 1 mux

Lab Task-4: Implement and Verify the Multiplexers and Demultiplexers using the built -in ICs in Logisim.









Lab Task-5:**Implement a Memory-Based System Using RAM and ROM ICs in Logisim**

Components to Implement:

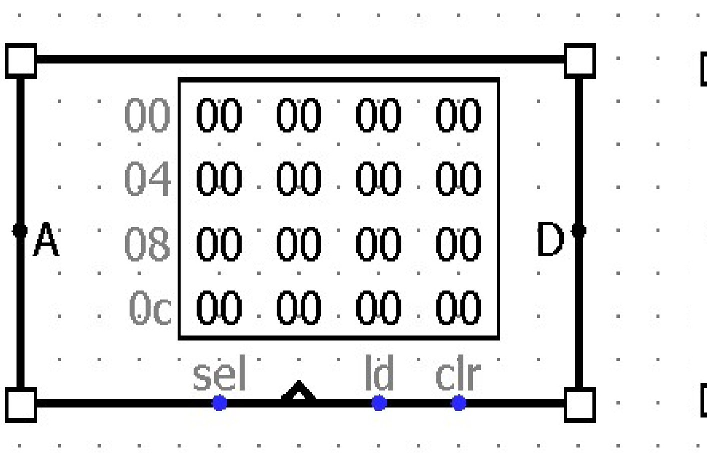
- **RAM (Random Access Memory):**
  - Implement reading and writing operations based on control signals.

ROM (Read-Only Memory):

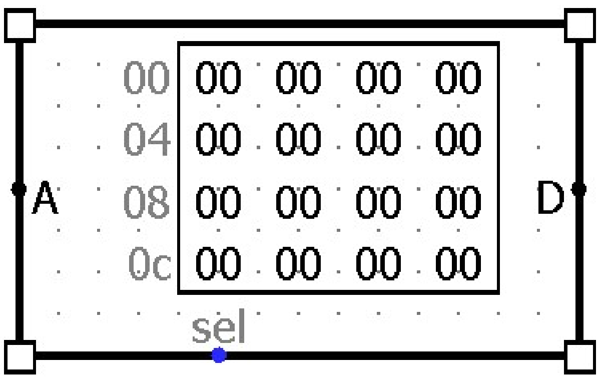
- Use the ROM to read data based on address inputs

Memory-Based System:

  - Combine the ROM and RAM into a single system.
  - Use a multiplexer to switch between ROM and RAM for data access.

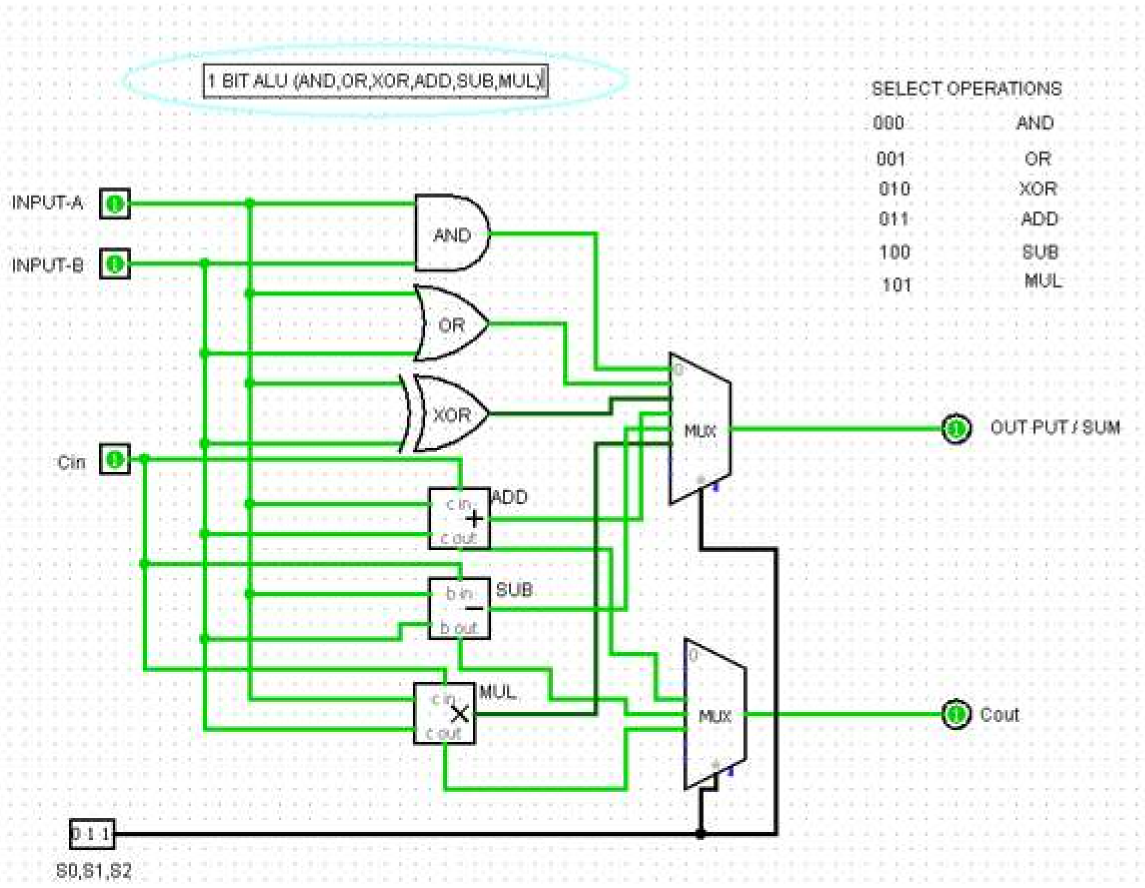


```
00 00 00 00 00
04 00 00 00 00
08 00 00 00 00
0c 00 00 00 00
```



```
00 00 00 00 00
04 00 00 00 00
08 00 00 00 00
0c 00 00 00 00
```

Lab Task-5: **Design 1 Bits ALU in LogiSim having (AND, OR, XOR, ADD , SUB and mul) functions.**

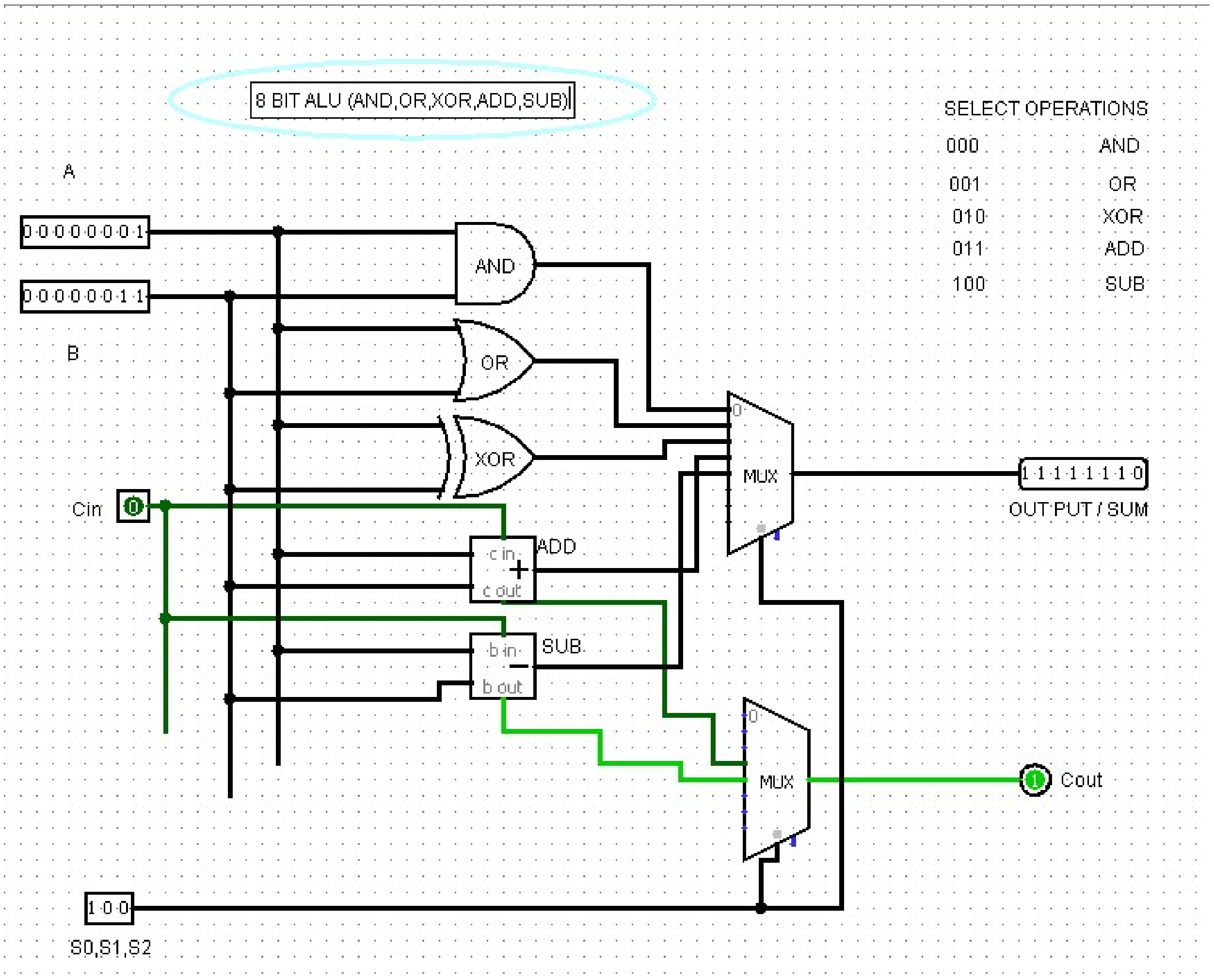


SELECT OPERATIONS

000	AND
001	OR
010	XOR
011	ADD
100	SUB
101	MUL

Lab Task-6:**Design 8 Bits ALU in LogiSim having (AND, OR, XOR, ADD , SUB and mul) functions.**

Just change the input and output bit size to 8 bits and the rest of the circuit will be same as shown in figure below.


8 BIT ALU (AND,OR,XOR,ADD,SUB)
A
0 0 0 0 0 0 0 1
0 0 0 0 0 0 1 1
B
AND
OR
XOR
Cin
0
c in
c out
ADD
b in
b out
SUB
MUX
MUX
1 1 1 1 1 1 1 0
OUT PUT / SUM
1
Cout
1 0 0
S0,S1,S2
SELECT OPERATIONS
000 AND
001 OR
010 XOR
011 ADD
100 SUB