Heaven's Light is Our Guide

Rajshahi University of Engineering & Technology

Department of Electrical & Computer Engineering

LAB REPORT

- *Course title(Sessional) : Digital Techniques Sessional*
- *Course Code : ECE-2112*
- *Date of submission : 10/02/2025*

Submitted By :	Submitted To :
Name : Sanumong Marma	***Md. Omaer Faruq Goni***
Roll : 2210061	***Assistant Professor***
Reg no. : 1115	***Department of ECE,***
Series : 22	***RUET***
Department of ECE, RUET	

Lab no. : 03

Experiment Name : Study of 3 to 8 Decoder and 4 to 1 Mux using verilog and DEEDS simulator.

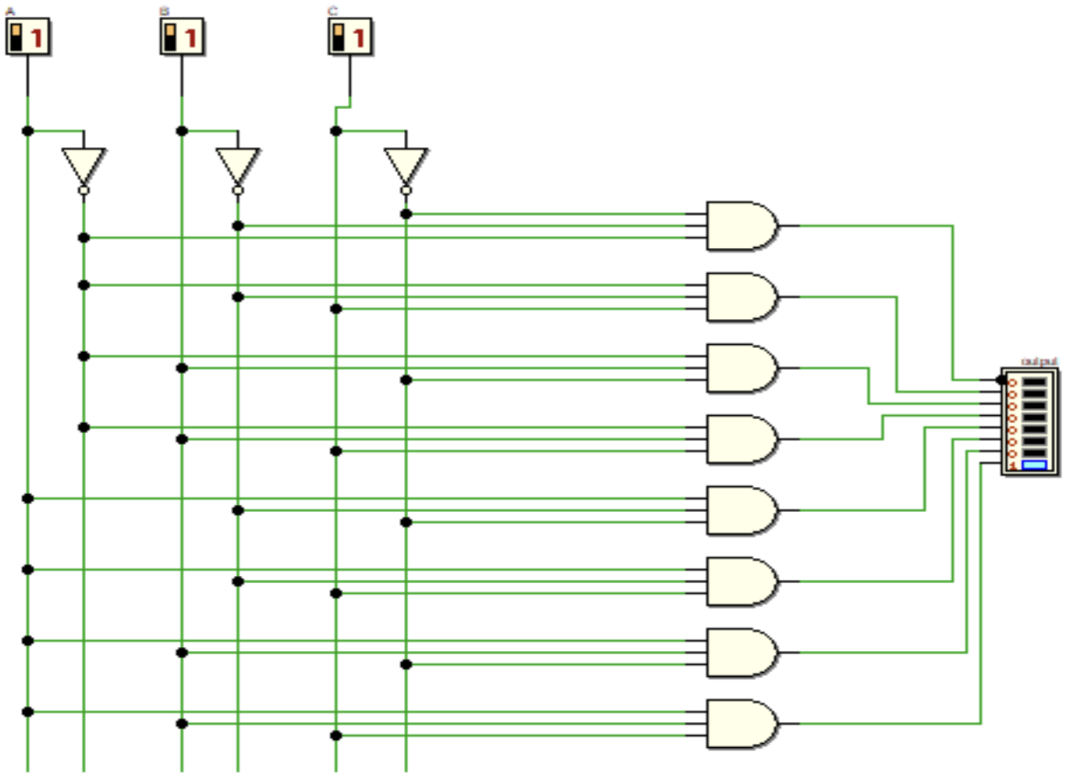
Theory: Decoder and Multiplexer (Mux) are fundamental combinational circuits in digital electronics, each serving a distinct purpose in data processing and control systems.

Decoder : A binary decoder is a digital circuit that converts a binary code into a set of outputs. The binary code represents the position of the desired output and is used to select the specific output that is active. Binary decoders are the inverse of encoders and are commonly used in digital systems to convert a serial code into a parallel set of outputs. A 2-to-4 decoder has 2 input lines and 4 output lines. Depending on the binary input (00, 01, 10, or 11), a specific output line is activated.

Multiplexer : A multiplexer is a combinational circuit that has many data inputs and a single output, depending on control or select inputs. For N input lines, log2(N) selection lines are required, or equivalently, for 2^n input lines, n selection lines are needed. Multiplexers are also known as “N-to-1 selectors,” parallel-to-serial converters, many-to-one circuits, and universal logic circuits. They are mainly used to increase the amount of data that can be sent over a network within a certain amount of time and bandwidth . A 4-to-1 Mux has 4 input lines, 1 output line, and 2 control/select lines. The select lines determine which input is passed to the output.

Circuit:

Decoder



.

Fig. 3 to 8 decoder

Code:

```
module Decoderr(input [2:0] inp, output reg [7:0] y);
always @(*) begin
y = 8'b00000000; // Default all outputs to 0
y[inp] = 1'b1; // Activate only one output
end
endmodule
`timescale 1ns/1ns
`include "Decoderr.v"
module Decoderr_tb;
reg [2:0] inp;
wire [7:0] y;
// Instantiate the Decoder module
Decoderr dec (inp, y);
initial begin
$monitor("Time = %0t | inp = %b -> y = %b", $time, inp, y);
// Apply test cases
inp = 3'b000; #10;
inp = 3'b001; #10;
inp = 3'b010; #10;
inp = 3'b011; #10;
inp = 3'b100; #10;
inp = 3'b101; #10;
inp = 3'b110; #10;inp = 3'b111; #10;
// Terminate simulation
```

```
#10 $finish;
end
Endmodule
```

Output:

```
[Running] Decoderr_tb.v
Time = 0 | inp = 000 -> y = 00000001
Time = 10 | inp = 001 -> y = 00000010
Time = 20 | inp = 010 -> y = 00000100
Time = 30 | inp = 011 -> y = 00001000
Time = 40 | inp = 100 -> y = 00010000
Time = 50 | inp = 101 -> y = 00100000
Time = 60 | inp = 110 -> y = 01000000
Time = 70 | inp = 111 -> y = 10000000
Decoderr_tb.v:25: $finish called at 90 (1ns)
[Done] exit with code=0 in 0.648 seconds
```

Circuit :

Multiplexer

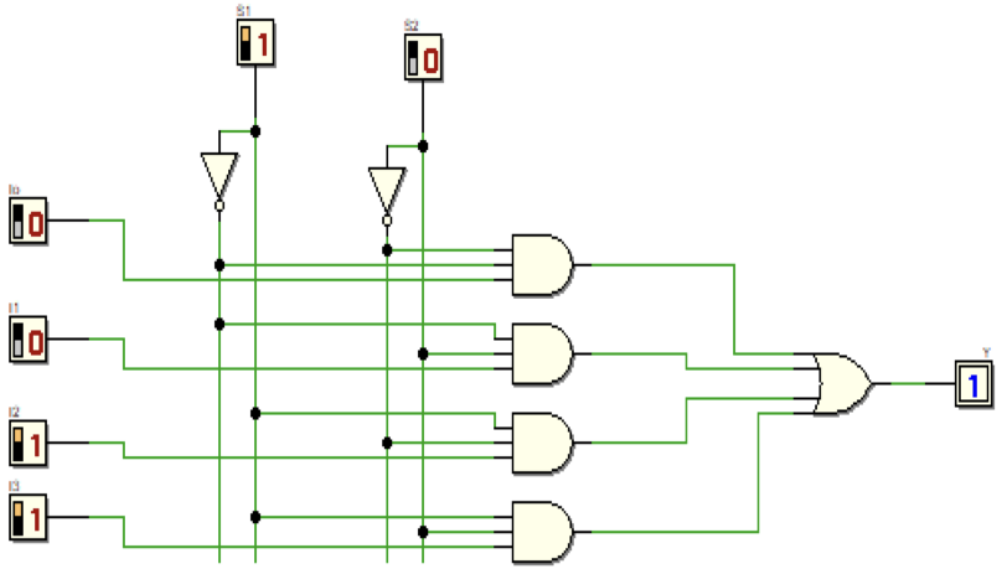


Fig. 4 to 1 mux

Code:

```
module Mux4to1(input [1:0] sel, input d0, d1, d2, d3, output reg y);
always @(*) begin
case (sel)
2'b00: y = d0;
2'b01: y = d1;
2'b10: y = d2;
2'b11: y = d3;
default: y = 0; // Default case (shouldn't happen in a 2-bit selector)
endcase
end
endmodule`timescale 1ns/1ns
`include "Mux4to1.v"
module Mux4to1_tb;
reg [1:0] sel;
```

```
reg d0, d1, d2, d3;
wire y;
// Instantiate the 4-to-1 Multiplexer
Mux4to1 mux (sel, d0, d1, d2, d3, y);
initial begin
$monitor("Time = %0t | sel = %b | d0 = %b, d1 = %b, d2 = %b, d3 = %b | y = %b",
$time, sel, d0, d1, d2, d3, y);
// Test different input cases
d0 = 1; d1 = 0; d2 = 1; d3 = 0; // Assign test values
sel = 2'b00; #10; // Expect y = d0 (1)
sel = 2'b01; #10; // Expect y = d1 (0)
sel = 2'b10; #10; // Expect y = d2 (1)
sel = 2'b11; #10; // Expect y = d3 (0)
// End simulation
#10 $finish;
end
Endmodule
```

Output:

```
[Running] Mux4to1_tb.v
Time = 0 | sel = 00 | d0 = 1, d1 = 0, d2 = 1, d3 = 0 | y = 1
Time = 10 | sel = 01 | d0 = 1, d1 = 0, d2 = 1, d3 = 0 | y = 0
Time = 20 | sel = 10 | d0 = 1, d1 = 0, d2 = 1, d3 = 0 | y = 1
Time = 30 | sel = 11 | d0 = 1, d1 = 0, d2 = 1, d3 = 0 | y = 0
Mux4to1_tb.v:25: $finish called at 50 (1ns)
[Done] exit with code=0 in 1.421 seconds
```

Discussion: In decoder only one output is active at a time. It is commonly used in memory addressing,instruction decoding in CPUs, and data routing. In this experiment, we implemented a 3-to-8 decoder,where a 3-bit input determines which of the 8 outputs is activated. The decoder logic ensures that onlyone output is high (1), while the others remain low (0).A multiplexer (Mux) is a combinational circuit that selects one of several inputs and forwards it to asingle output based on a selector input. In this lab, we implemented a 4-to-1 Mux, where a 2-bit selectsignal determines which one of the four inputs is passed to the output. Mux circuits are widely used in data selection, signal routing, and arithmetic logic units (ALUs).