Lab Report #5

Hanz Chua U26738740

Viraj Amarasinghe U14806439

Jian Gong U42910460

Introduction

In this lab, we assembled 4-to-1 multiplexers using inverters and AND gates. The purpose of this lab was to help gain a deeper understanding of higher-level devices like decoders and multiplexers and the benefits of hierarchical design.

Methods and Materials

2 74HC11 3-Input AND Gate

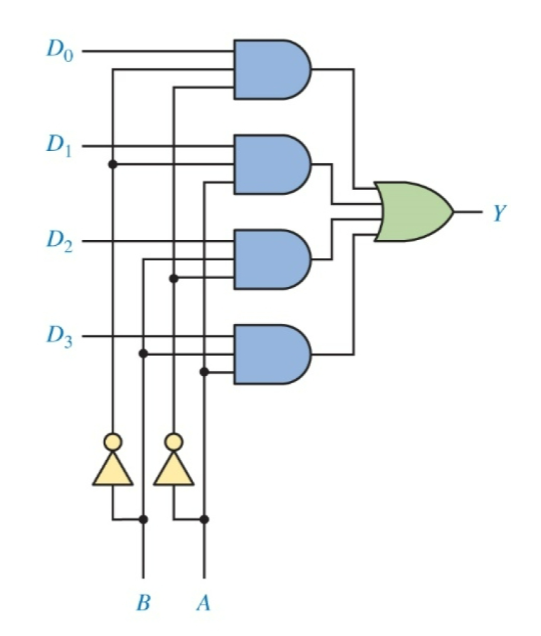
1 74LS32N 2-Input OR Gate

1 74LS04N Hex Inverter

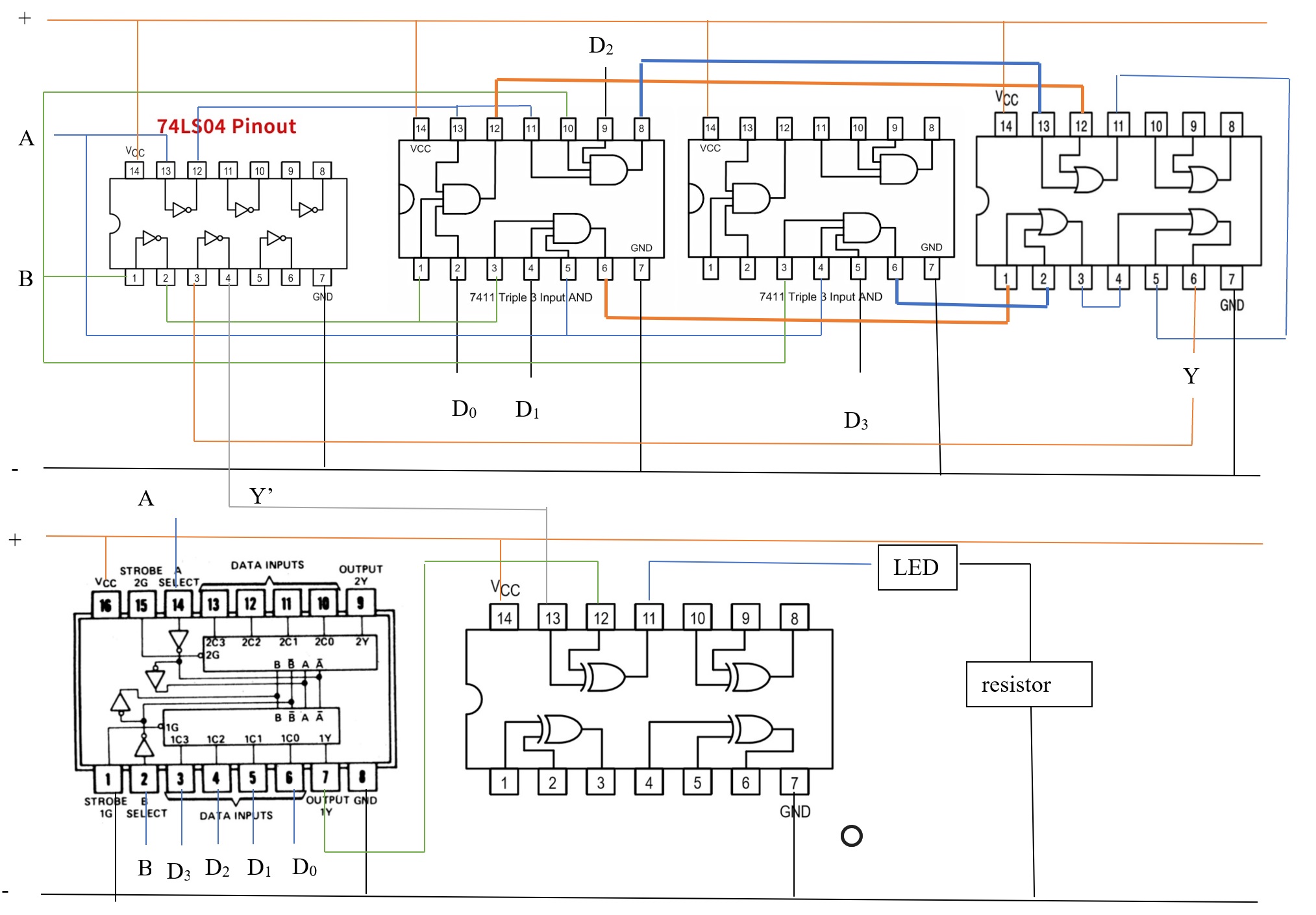
1 74LS153N 4 to 1 Multiplexer

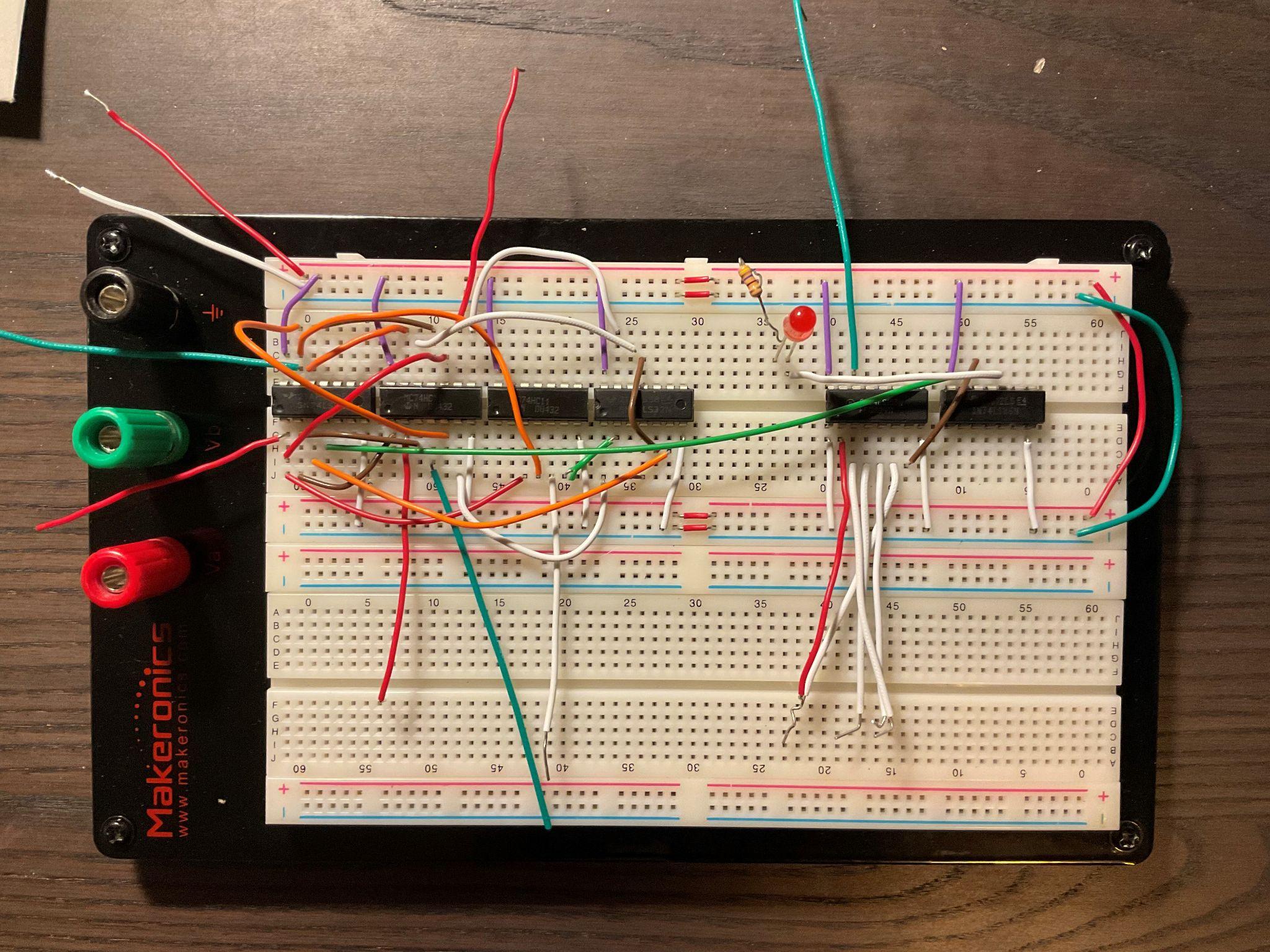
1 74LS86N Quad 2-Input XOR Gate

1. First, we set up the verilog code on the EDA Playground. Go to the test bench window and copy the code at the bottom of this document and under the label Test Bench. Do the same for the module window with the code under the Module label. Make sure to select Synopsys VCS as the code’s simulator. Make sure “Open EPWave after run” is checked off, so that we can observe the test case in timing diagram form. Then press run.
2. Below, we have the multiplexer circuit put together with only the gates.



Below is the schematic of the circuit we built on the breadboard. In order to assemble we must connect the wires in the following order, as shown in the pictures below.





Result

We observed that the 4 to 1 mux and the 74LS153N mux had the same functionality; the same inputs yield the same output for both multiplexers. We encountered little issues in making the circuit, but once they were resolved, we concluded that the results came out as expected and contributed to the goal of the experiment; to help us understand the higher-level devices.

Conclusion

In conclusion, we assembled a 4 to 1 multiplexer with gates and compared it to the multiplexer built into the IC chip. We had small issues, such as connecting wires to the wrong holes in the breadboard, but other than that we had no other issues.

Questions

1.

2-to-1 Multiplexer

truth table:

S D0 D1 Y

0 0 X 0

0 1 X 1

1 X 0 0

1 X 1 1

SOP equation: Y = S’D0 + SD1

The output is D0 when Select value is S = 0 and the output is D1 when Select value is S = 1

4-to-1 Multiplexer

truth table:

S0 S1 D0 D1 D2 D3 Y

0 0 0 X X X 0

0 0 1 X X X 1

0 1 X 0 X X 0

0 1 X 1 X X 1

1 0 X X 0 X 0

1 0 X X 1 X 1

1 1 X X X 0 0

1 1 X X X 1 1

Y = S0’S1’ D0 + S0’ S1 D1 + S0 S1’ D2 + S0 S1 D3

2.

4 IC’s used for the 4 to 1 multiplexer.

2 74HC11 3-Input AND Gate,

1 74LS32N 2-Input OR Gate,

1 74LS04N Hex Inverter.

1 74LS153N 4 to 1 Multiplexer and 1 74LS86N Quad 2-Input XOR Gate

3.

Testbench

module my\_function\_tb();

reg D0, D1, D2, D3, S0, S1;

wire Y;

four\_to\_one\_mux M3(Y, D0, D1, D2, D3, S0, S1);

initial begin

$dumpfile("test.vcd");

$dumpvars;

$display("Starting simulation...\n");

$display("Time\ta\tb\tcin\ts\tcout\n");

$monitor("%2d\t%d\t%d\t%d\t%d\t%d",$time,D0, D1, D2, D3, S0, S1);

D0 = 0;

D1 = 0;

D2 = 0;

D3 = 0;

S0 = 0;

S1 = 0;

#10 D0 = 0; D1 = 0; D2 = 0; D3 = 0; S0 = 0; S1 = 0;

#10 D0 = 0; D1 = 0; D2 = 0; D3 = 1; S0 = 0; S1 = 1;

#10 D0 = 0; D1 = 0; D2 = 1; D3 = 0; S0 = 1; S1 = 0;

#10 D0 = 0; D1 = 0; D2 = 1; D3 = 1; S0 = 1; S1 = 1;

#10 D0 = 0; D1 = 1; D2 = 0; D3 = 0; S0 = 0; S1 = 0;

#10 D0 = 0; D1 = 1; D2 = 0; D3 = 1; S0 = 0; S1 = 1;

#10 D0 = 0; D1 = 1; D2 = 1; D3 = 0; S0 = 1; S1 = 0;

#10 D0 = 0; D1 = 1; D2 = 1; D3 = 1; S0 = 1; S1 = 1;

#10 D0 = 1; D1 = 0; D2 = 0; D3 = 0; S0 = 0; S1 = 0;

#10 D0 = 1; D1 = 0; D2 = 0; D3 = 1; S0 = 0; S1 = 1;

#10 D0 = 1; D1 = 0; D2 = 1; D3 = 0; S0 = 1; S1 = 0;

#10 D0 = 1; D1 = 0; D2 = 1; D3 = 1; S0 = 1; S1 = 1;

#10 D0 = 1; D1 = 1; D2 = 0; D3 = 0; S0 = 0; S1 = 0;

#10 D0 = 1; D1 = 1; D2 = 0; D3 = 1; S0 = 0; S1 = 1;

#10 D0 = 1; D1 = 1; D2 = 1; D3 = 0; S0 = 1; S1 = 0;

#10 D0 = 1; D1 = 1; D2 = 1; D3 = 1; S0 = 1; S1 = 1;

#10 D0 = 1; D1 = 1; D2 = 1; D3 = 1; S0 = 1; S1 = 1;

#10 $finish;

end

endmodule

Module

module two\_to\_one\_mux(Y, D0, D1, S);

input D0, D1, S;

output Y;

wire W1, W2, S1;

not(S1 ,S);

and(W1, D1, S);

and(W2, D0, S1);

or(Y, W1, W2);

endmodule

module four\_to\_one\_mux(Y, D0, D1, D2, D3, S0, S1);

input D0, D1, D2, D3, S0, S1;

output Y;

wire W1, W2;

two\_to\_one\_mux M0(W1, D0, D1, S0);

two\_to\_one\_mux M1(W2, D1, D2, S0);

two\_to\_one\_mux M2(Y, W1, W2, S1);

endmodule

4.

1. Hierarchical design defines common functionality in HDL code and physical circuits. This makes the circuit more modular and compact; repetitive code can easily be re-called. Because of this however, a possible drawback is that it is harder to examine and understand the logic behind the circuit.
2. A decoder with an active high/low input can realize and SOP or multiple SOP functions as u can associate any combination to a desired output. The limitations are determined by the amount of inputs and outputs we have in our decoder.
3. To construct an 8 to 1 mux, we combine two 4 to 1 muxes. This forms an 8 to 2 mux, which we can connect to a 2 to 1 mux to finally get an 8 to 1 mux.