ECE3300L

Summer 2025 Semester

Lab 1

Kevin Yu

Noah Bocanegra

6/18/25

For the second lab, we are tasked with designing the code for a 4 to 16 decoder with an enable input. When the enable is HIGH, the 4-bit input will determine exactly which single output will be set to HIGH, while all the other outputs will be set to LOW. We will write this decoder in two methods, gate level (structural) and behavioral coding. We will also write a testbench code to simulate and verify all the inputs correspond with their expected output.

The youtube link of our Demo: https://youtube.com/shorts/SuwfFToTyY4?feature=share

First we coded the behavioral and structural code and modified the XDC file to fit our variable names and enable pin.

Structural:

```
module demux4to16(
    input wire [3:0] A,
    input wire E,
    output wire [15:0] Y
    );
    assign Y[0]  = E & ~A[3] & ~A[2] & ~A[1] & ~A[0];
    assign Y[1]  = E & ~A[3] & ~A[2] & ~A[1] &  A[0];
    assign Y[2]  = E & ~A[3] & ~A[2] &  A[1] & ~A[0];
    assign Y[3]  = E & ~A[3] & ~A[2] &  A[1] &  A[0];
    assign Y[4]  = E & ~A[3] &  A[2] & ~A[1] & ~A[0];
    assign Y[5]  = E & ~A[3] &  A[2] & ~A[1] &  A[0];
    assign Y[6]  = E & ~A[3] &  A[2] &  A[1] & ~A[0];
    assign Y[7]  = E & ~A[3] &  A[2] &  A[1] &  A[0];
    assign Y[8]  = E &  A[3] & ~A[2] & ~A[1] & ~A[0];
    assign Y[9]  = E &  A[3] & ~A[2] & ~A[1] &  A[0];
    assign Y[10] = E &  A[3] & ~A[2] &  A[1] & ~A[0];
    assign Y[11] = E &  A[3] & ~A[2] &  A[1] &  A[0];
    assign Y[12] = E &  A[3] &  A[2] & ~A[1] & ~A[0];
    assign Y[13] = E &  A[3] &  A[2] & ~A[1] &  A[0];
    assign Y[14] = E &  A[3] &  A[2] &  A[1] & ~A[0];
    assign Y[15] = E &  A[3] &  A[2] &  A[1] &  A[0];
endmodule
```

Behavioral:

```
module demux4to16(   //Initialize all wires for this module
    input wire [3:0] A, //Initialize 4 input wires
    input wire E, //Initialize 1 enable wire
    output reg [15:0] Y //Initialize 16 output wires
    );

    always @(*) begin
    Y = 16'b0;  //Reset output wires
    if (E) begin // ▸ only decode when enabled
```

```
      case (A)  //Check input state
       4'b0000: Y = 16'b0000_0000_0000_0001; // ▸ output 0
       4'b0001: Y = 16'b0000_0000_0000_0010; // ▸ output 1
       4'b0010: Y = 16'b0000_0000_0000_0100; // ▸ output 2
       4'b0011: Y = 16'b0000_0000_0000_1000; // ▸ output 3
       4'b0100: Y = 16'b0000_0000_0001_0000; // ▸ output 4
       4'b0101: Y = 16'b0000_0000_0010_0000; // ▸ output 5
       4'b0110: Y = 16'b0000_0000_0100_0000; // ▸ output 6
       4'b0111: Y = 16'b0000_0000_1000_0000; // ▸ output 7
       4'b1000: Y = 16'b0000_0001_0000_0000; // ▸ output 8
       4'b1001: Y = 16'b0000_0010_0000_0000; // ▸ output 9
       4'b1010: Y = 16'b0000_0100_0000_0000; // ▸ output 10
       4'b1011: Y = 16'b0000_1000_0000_0000; // ▸ output 11
       4'b1100: Y = 16'b0001_0000_0000_0000; // ▸ output 12
       4'b1101: Y = 16'b0010_0000_0000_0000; // ▸ output 13
       4'b1110: Y = 16'b0100_0000_0000_0000; // ▸ output 14
       4'b1111: Y = 16'b1000_0000_0000_0000; // ▸ output 15
      endcase
     end
   end
endmodule
```

XDC code:

```
set_property -dict { PACKAGE_PIN J15   IOSTANDARD LVCMOS33 } [get_ports { A[0] }];
#IO_L24N_T3_RS0_15 Sch=sw[0]
set_property -dict { PACKAGE_PIN L16   IOSTANDARD LVCMOS33 } [get_ports { A[1] }];
#IO_L3N_T0_DQS_EMCCLK_14 Sch=sw[1]
set_property -dict { PACKAGE_PIN M13   IOSTANDARD LVCMOS33 } [get_ports { A[2] }];
#IO_L6N_T0_D08_VREF_14 Sch=sw[2]
set_property -dict { PACKAGE_PIN R15   IOSTANDARD LVCMOS33 } [get_ports { A[3] }];
#IO_L13N_T2_MRCC_14 Sch=sw[3]
set_property -dict { PACKAGE_PIN R17   IOSTANDARD LVCMOS33 } [get_ports { E }];
#IO_L12N_T1_MRCC_14 Sch=sw[4]

## LEDs
set_property -dict { PACKAGE_PIN H17   IOSTANDARD LVCMOS33 } [get_ports { Y[0] }];
#IO_L18P_T2_A24_15 Sch=led[0]
set_property -dict { PACKAGE_PIN K15   IOSTANDARD LVCMOS33 } [get_ports { Y[1] }];
#IO_L24P_T3_RS1_15 Sch=led[1]
set_property -dict { PACKAGE_PIN J13   IOSTANDARD LVCMOS33 } [get_ports { Y[2] }];
#IO_L17N_T2_A25_15 Sch=led[2]
set_property -dict { PACKAGE_PIN N14   IOSTANDARD LVCMOS33 } [get_ports { Y[3] }];
#IO_L8P_T1_D11_14 Sch=led[3]
```

```
set_property -dict { PACKAGE_PIN R18   IOSTANDARD LVCMOS33 } [get_ports { Y[4] }];
#IO_L7P_T1_D09_14 Sch=led[4]
set_property -dict { PACKAGE_PIN V17   IOSTANDARD LVCMOS33 } [get_ports { Y[5] }];
#IO_L18N_T2_A11_D27_14 Sch=led[5]
set_property -dict { PACKAGE_PIN U17   IOSTANDARD LVCMOS33 } [get_ports { Y[6] }];
#IO_L17P_T2_A14_D30_14 Sch=led[6]
set_property -dict { PACKAGE_PIN U16   IOSTANDARD LVCMOS33 } [get_ports { Y[7] }];
#IO_L18P_T2_A12_D28_14 Sch=led[7]
set_property -dict { PACKAGE_PIN V16   IOSTANDARD LVCMOS33 } [get_ports { Y[8] }];
#IO_L16N_T2_A15_D31_14 Sch=led[8]
set_property -dict { PACKAGE_PIN T15   IOSTANDARD LVCMOS33 } [get_ports { Y[9] }];
#IO_L14N_T2_SRCC_14 Sch=led[9]
set_property -dict { PACKAGE_PIN U14   IOSTANDARD LVCMOS33 } [get_ports { Y[10] }];
#IO_L22P_T3_A05_D21_14 Sch=led[10]
set_property -dict { PACKAGE_PIN T16   IOSTANDARD LVCMOS33 } [get_ports { Y[11] }];
#IO_L15N_T2_DQS_DOUT_CSO_B_14 Sch=led[11]
set_property -dict { PACKAGE_PIN V15   IOSTANDARD LVCMOS33 } [get_ports { Y[12] }];
#IO_L16P_T2_CSI_B_14 Sch=led[12]
set_property -dict { PACKAGE_PIN V14   IOSTANDARD LVCMOS33 } [get_ports { Y[13] }];
#IO_L22N_T3_A04_D20_14 Sch=led[13]
set_property -dict { PACKAGE_PIN V12   IOSTANDARD LVCMOS33 } [get_ports { Y[14] }];
#IO_L20N_T3_A07_D23_14 Sch=led[14]
set_property -dict { PACKAGE_PIN V11   IOSTANDARD LVCMOS33 } [get_ports { Y[15] }];
#IO_L21N_T3_DQS_A06_D22_14 Sch=led[15]
```

Testbench:

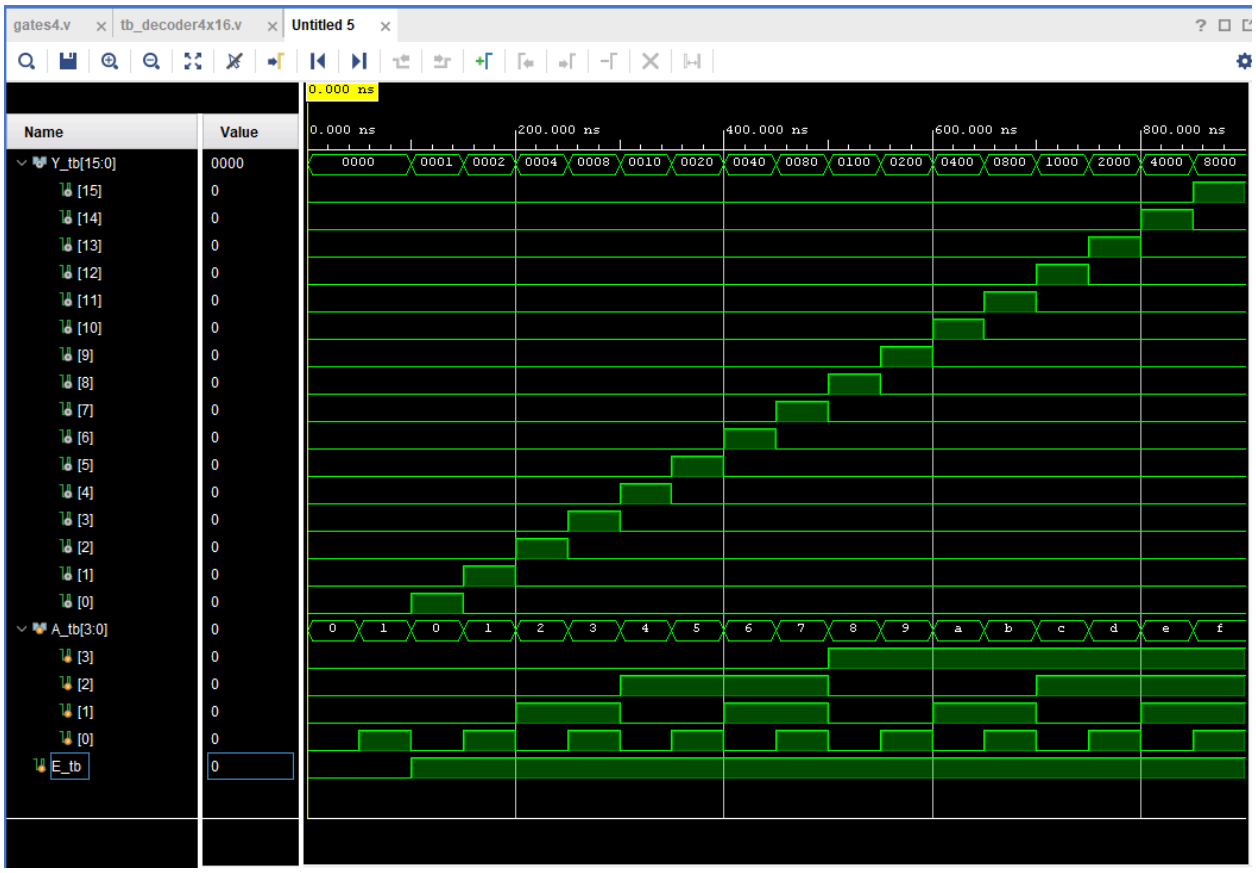
```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 06/25/2025 10:59:28 PM
// Design Name:
// Module Name: tb_decoder4x16
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
```

```
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module tb_decoder4x16(
    );
    wire [15:0] Y_tb;
    reg [3:0] A_tb;
    reg E_tb;

    demux4to16 X(.A(A_tb), .E(E_tb), .Y(Y_tb));

    initial
        begin
            A_tb = 4'b0000;
            E_tb = 1'b0;
        #50
            A_tb = 4'b0001;
            E_tb = 1'b0;
        #50
            A_tb = 4'b0000;
            E_tb = 1'b1;
        #50
            A_tb = 4'b0001;
            E_tb = 1'b1;
        #50
            A_tb = 4'b0010;
            E_tb = 1'b1;
        #50
            A_tb = 4'b0011;
            E_tb = 1'b1;
        #50
           A_tb = 4'b0100;
            E_tb = 1'b1;
        #50
            A_tb = 4'b0101;
            E_tb = 1'b1;
        #50
            A_tb = 4'b0110;
            E_tb = 1'b1;
        #50
            A_tb = 4'b0111;
            E_tb = 1'b1;
```

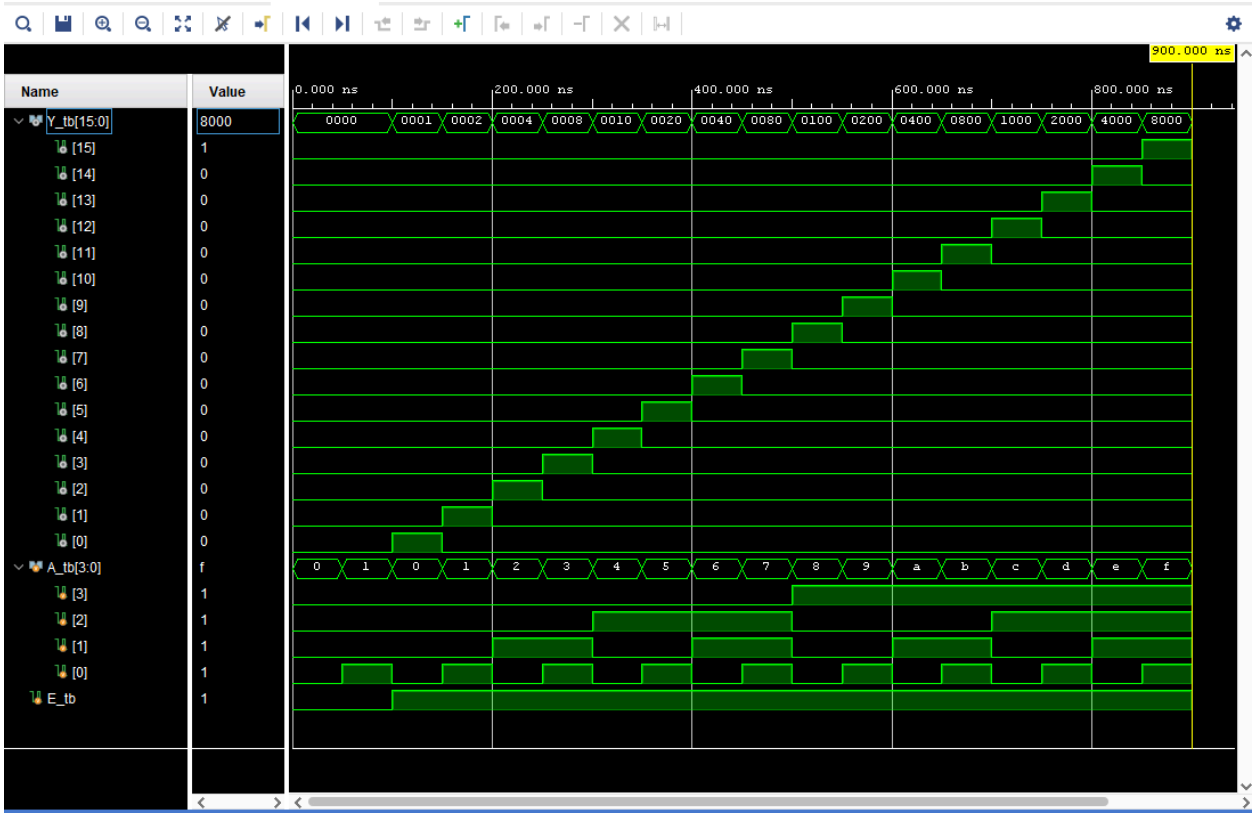
```
        #50
            A_tb = 4'b1000;
            E_tb = 1'b1;
        #50
            A_tb = 4'b1001;
            E_tb = 1'b1;
        #50
            A_tb = 4'b1010;
            E_tb = 1'b1;
        #50
            A_tb = 4'b1011;
            E_tb = 1'b1;
        #50
           A_tb = 4'b1100;
            E_tb = 1'b1;
        #50
            A_tb = 4'b1101;
            E_tb = 1'b1;
        #50
            A_tb = 4'b1110;
            E_tb = 1'b1;
        #50
            A_tb = 4'b1111;
            E_tb = 1'b1;
        #50
            $finish;
        end
endmodule
```

Our Resulting Structural and behavioral Test Bench results were below where we can see the different codes were correct and we were able to correctly get the proper LED to match the proper input switches.

Structural:



Behavioral:



In conclusion, we were correctly able to write both structural and behavioral code, and again properly upload it to the board and get the board to correctly show the proper led and switch combinations. We were also able to create a testbench and simulate our codes to ensure they worked. This lab was our first time using the simulation but we were able to work through and figure out how to use it and see its importance for future projects in saving time by not having to test by sending code to the board everytime.