Lab 2 Report

Name: Hyokwon Chung UT EID: hc27426 Section: 17540

Name: Rishi Jarral UT EID: rsj647 Section: 17508

Note: Only 1 student per group needs to submit a report.

Checklist:

Part 1 –

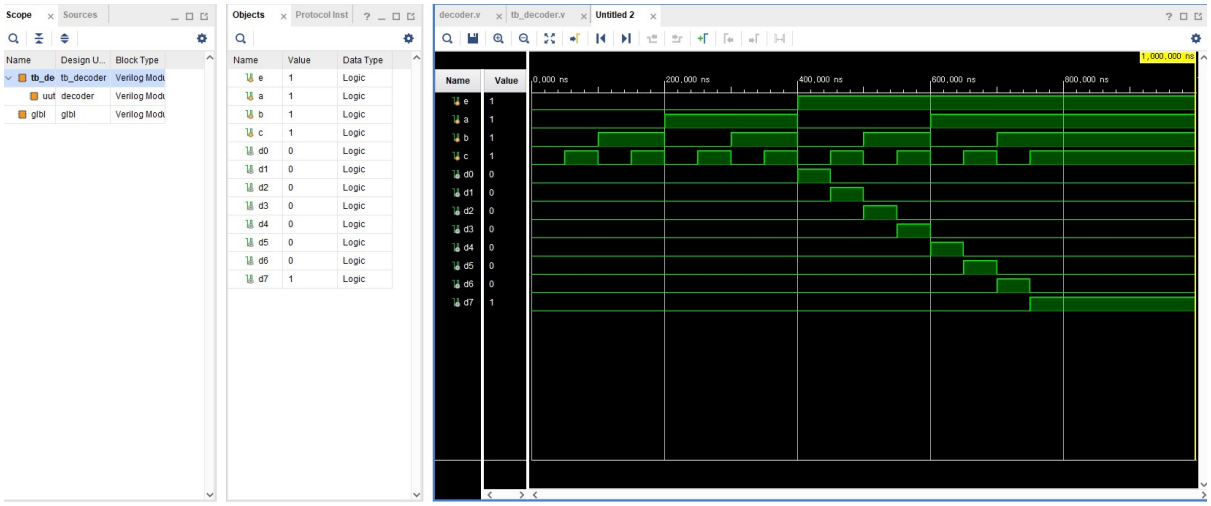
i. Simulation waveforms for Part 1 for Structural as well as Behavioral modelling (Screenshots)

Part 2 –

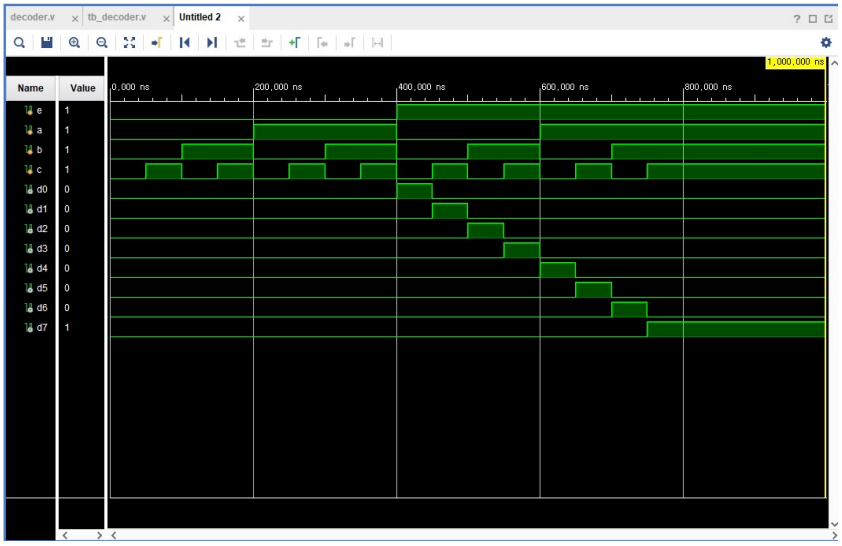
ii. Truth table of the function
iii. Algebraic expression of the logic function
iv. Logic circuit schematic
v. Verilog codes for module and testbench for structural modelling
vi. Simulation waveform for structural modelling (Screenshot)
vii. Verilog codes for module and testbench for behavioral modelling
viii. Simulation waveform for behavioral modelling (Screenshot)

Note *–> The Verilog codes should be* ***copied in your lab report****, and the actual Verilog (.v) files need to be* ***zipped and submitted*** *as well on Canvas. You are not allowed to change your Verilog codes after final submission as the TAs may download the submitted codes from Canvas during checkouts. For the truth table, algebraic expression and circuit schematic, you are free to draw it on paper and then put the pictures in your lab report, but please make sure it is legible for the TAs to grade it properly.*

I

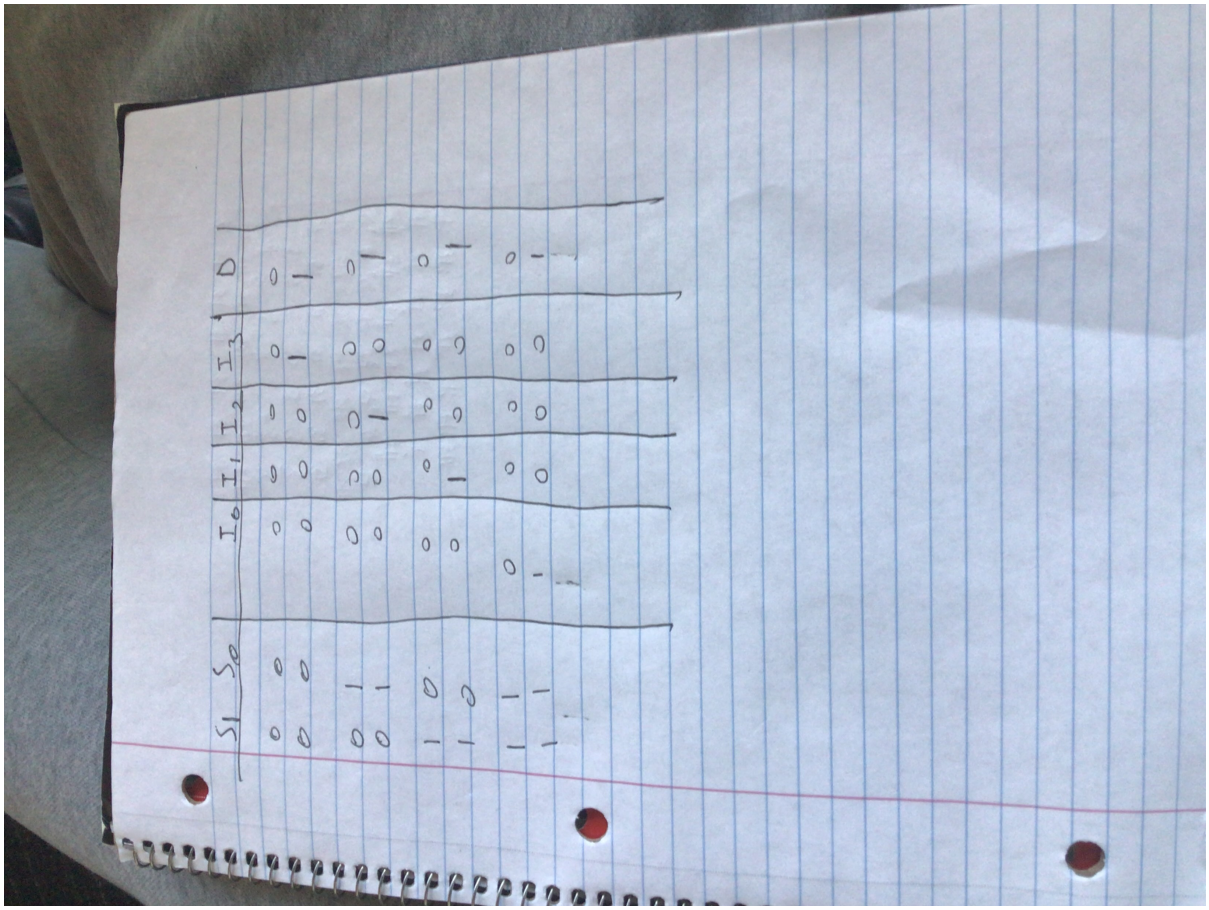


-part 1 Behaviour



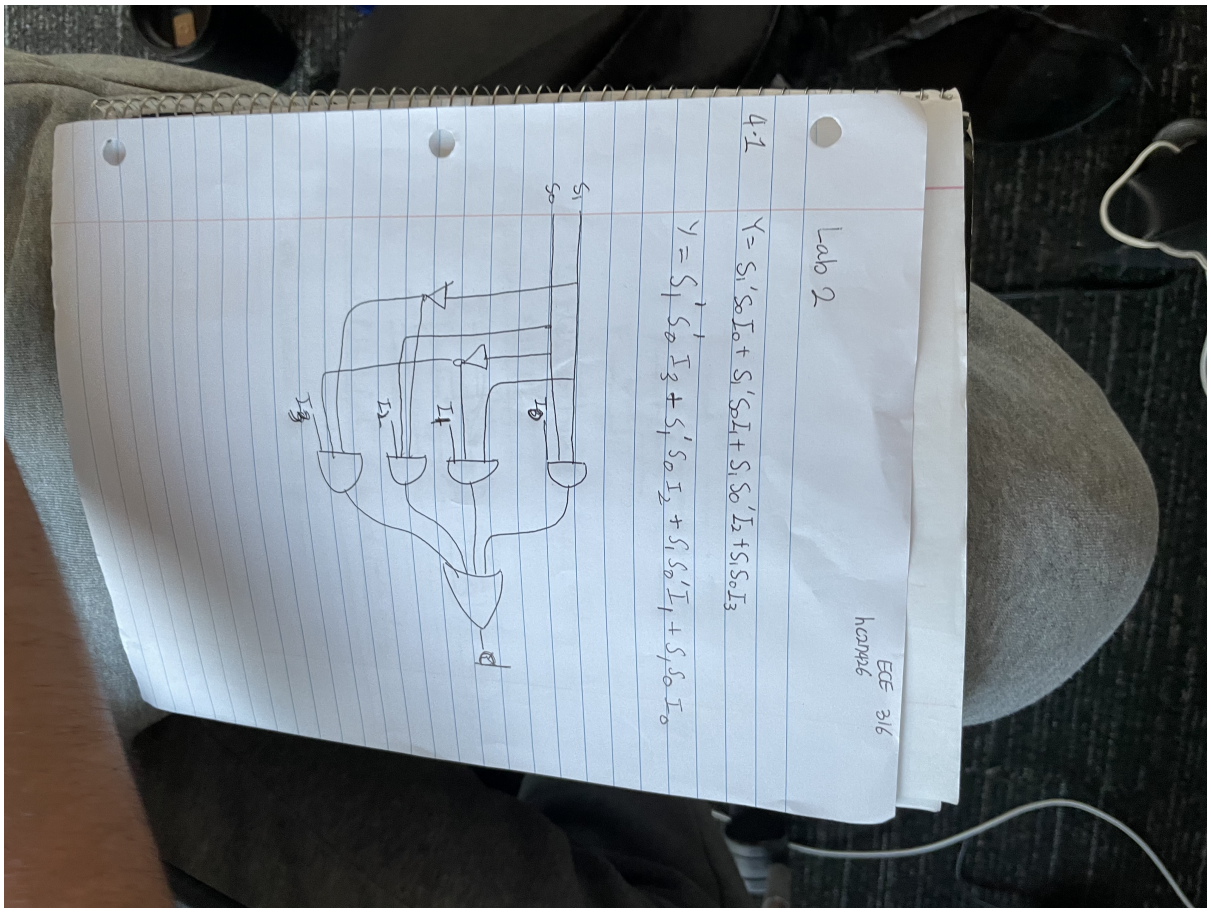
-part 1 Constructial

II – Truth table



III – Algebraic equation

IV – Logic circuit



V - MUX

```
module MUX(
    input i0,
    input i1,
    input i2,
    input i3,
    input s0,
    input s1,
    output d0 );



    wire s0_n;
    wire s1_n;
    not not0( s0_n, s0 );
    not not1( s1_n, s1 );

    wire as,bs,cs,ds;
    and and0(as,i0,s1,s0);
    and and1(bs,i1,s1,s0_n);
    and and2(cs,i2,s1_n,s0);
    and and3(ds,i3,s1_n,s0_n);

    or or3(d0,as,bs,cs,ds);
endmodule
```

V – Testbench

```
`timescale 1ns / 1ps

module tb_mux;

    // Inputs, defined as registers
    reg s0;
    reg s1;
    reg i0;
    reg i1;
    reg i2;
    reg i3;

    // Outputs, defined as wires
    wire d0;

    // Instantiate the UUT (unit under test)
    MUX uut(
        .s0(s0),
        .s1(s1),
        .i0(i0),
        .i1(i1),
        .i2(i2),
        .i3(i3),
        .d0(d0)
    );

    initial begin
```

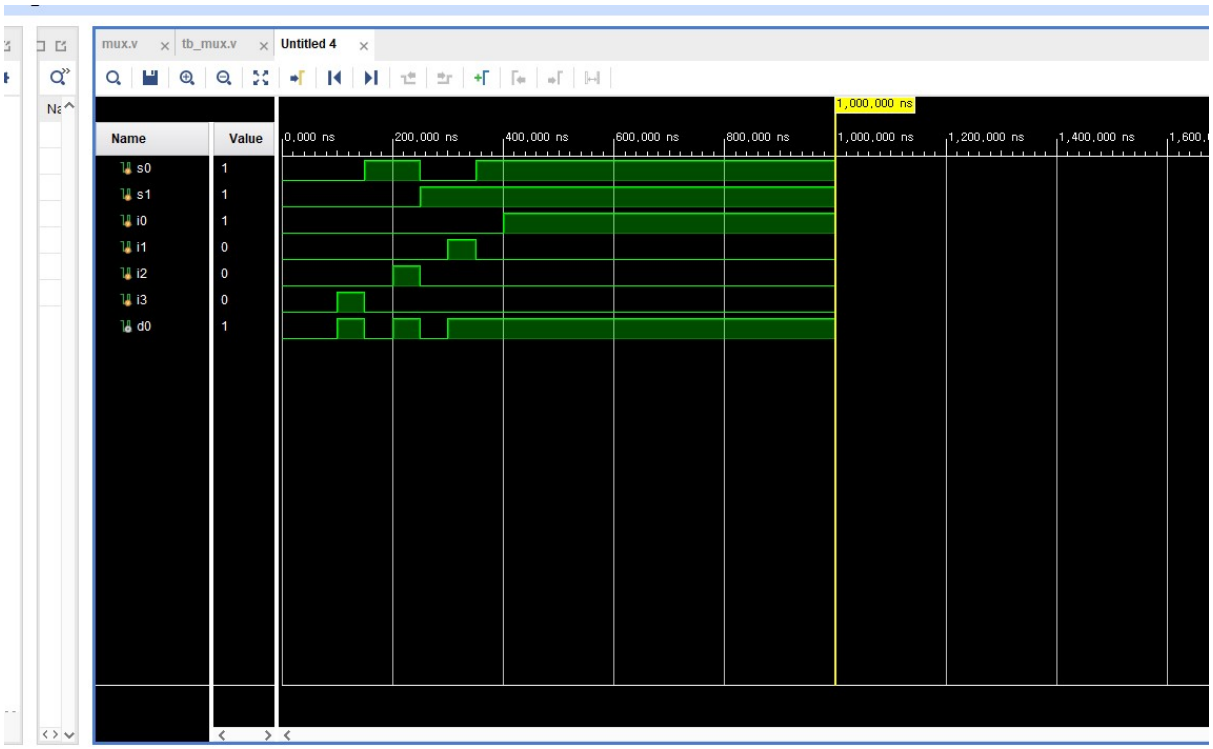
```
        // Initialize Inputs
        s0 = 0;
        s1 = 0;
        i0 = 0;
        i1 = 0;
        i2 = 0;
        i3 = 0;

        // Wait 50 ns for global reset to finish
        #50;

        // Stimulus -- all input combinations followed by some wait time to observe the o/p
        s0 = 0;
        s1 = 0;
        i0 = 0;
        i1 = 0;
        i2 = 0;
        i3 = 0;
        #50;
        $display("TC01");
        if( d0 != 0 ) $display("Result is wrong");

        s0 = 0;
        s1 = 0;
        i0 = 0;
        i1 = 0;
        i2 = 0;
```

```
i3 = 1;
#50;
$display("TC02");
if( d0 != i3  ) $display("Result is wrong");

s0 = 1;
s1 = 0;
i0 = 0;
i1 = 0;
i2 = 0;
i3 = 0;
#50;
$display("TC03");
if( d0 != 0  ) $display("Result is wrong");

s0 = 1;
s1 = 0;
i0 = 0;
i1 = 0;
i2 = 1;
i3 = 0;
#50;
$display("TC04");
if( d0 != i2 ) $display("Result is wrong");

s0 = 0;
s1 = 1;
i0 = 0;
```

```
i1 = 0;
i2 = 0;
i3 = 0;
#50;
$display("TC05");
if( d0 != 0 ) $display("Result is wrong");

s0 = 0;
s1 = 1;
i0 = 0;
i1 = 1;
i2 = 0;
i3 = 0;
#50;
$display("TC06");
if( d0 != i1 ) $display("Result is wrong");

s0 = 1;
s1 = 1;
i0 = 0;
i1 = 0;
i2 = 0;
i3 = 0;
#50;
$display("TC07");
if( d0 != 0 ) $display("Result is wrong");

s0 = 1;
```

```
        s1 = 1;
        i0 = 1;
        i1 = 0;
        i2 = 0;
        i3 = 0;
        #50;
        $display("TC08");
        if( d0 != i0 ) $display("Result is wrong");


    end


endmodule
```

VI


mux.v
tb_mux.v
Untitled 4
Name
Value
s0
1
s1
1
i0
1
i1
0
i2
0
i3
0
d0
1
0,000 ns
200,000 ns
400,000 ns
600,000 ns
800,000 ns
1,000,000 ns
1,200,000 ns
1,400,000 ns
1,000,000 ns

VII - MUX

```
module MUX( input i0, i1, i2, i3,
            input s0, s1,
            output d0    );



    reg out = 0;
    assign d0 = out ;
    always @ (*)
    begin
        if (s0 == 0 && s1 ==0)  out =i3;
        else if (s0 == 1 && s1 ==0)  out =i2;
        else if (s0 == 0 && s1 ==1)  out =i1;
       // else if (s0 == 1 && s1 ==1)  out =i0;


   end


endmodule
```

VII – testbench

```
`timescale 1ns / 1ps


module tb_mux;

    // Inputs, defined as registers
    reg s0;
    reg s1;
    reg i0;
    reg i1;
    reg i2;
    reg i3;

    // Outputs, defined as wires
    wire d0;

    // Instantiate the UUT (unit under test)
    MUX uut(
        .s0(s0),
        .s1(s1),
        .i0(i0),
        .i1(i1),
        .i2(i2),
        .i3(i3),
        .d0(d0)
    );

    initial begin
```

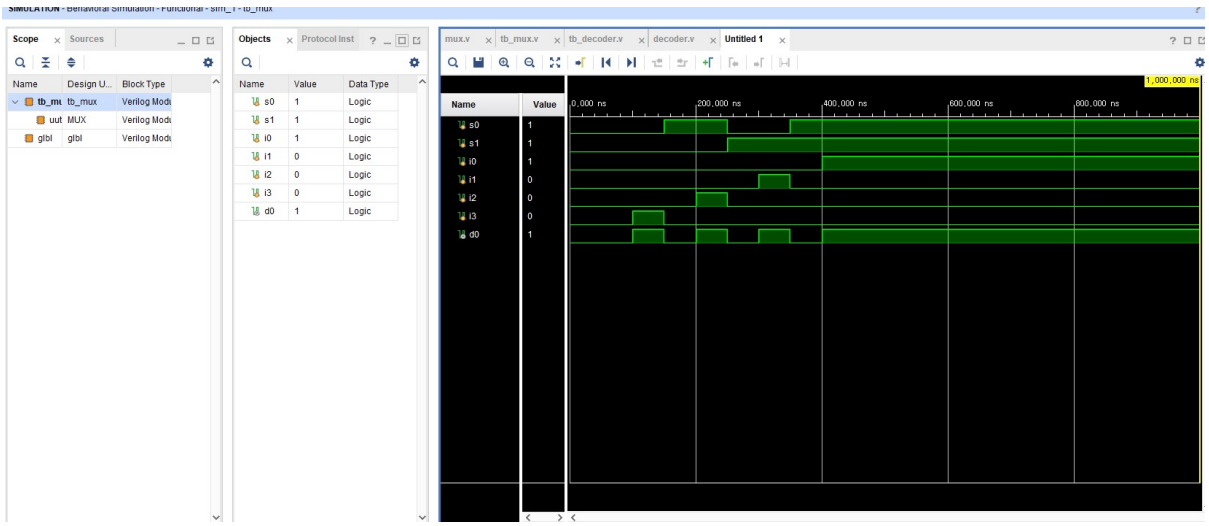
```
        // Initialize Inputs
        s0 = 0;
        s1 = 0;
        i0 = 0;
        i1 = 0;
        i2 = 0;
        i3 = 0;

        // Wait 50 ns for global reset to finish
        #50;

        // Stimulus -- all input combinations followed by some wait time to observe the o/p
        s0 = 0;
        s1 = 0;
        i0 = 0;
        i1 = 0;
        i2 = 0;
        i3 = 0;
        #50;
        $display("TC01");
        if( d0 != 0 ) $display("Result is wrong");

        s0 = 0;
        s1 = 0;
        i0 = 0;
        i1 = 0;
        i2 = 0;
```

```
i3 = 1;
#50;
$display("TC02");
if( d0 != i3  ) $display("Result is wrong");

s0 = 1;
s1 = 0;
i0 = 0;
i1 = 0;
i2 = 0;
i3 = 0;
#50;
$display("TC03");
if( d0 != 0  ) $display("Result is wrong");

s0 = 1;
s1 = 0;
i0 = 0;
i1 = 0;
i2 = 1;
i3 = 0;
#50;
$display("TC04");
if( d0 != i2 ) $display("Result is wrong");

s0 = 0;
s1 = 1;
i0 = 0;
```

```
i1 = 0;
i2 = 0;
i3 = 0;
#50;
$display("TC05");
if( d0 != 0 ) $display("Result is wrong");

s0 = 0;
s1 = 1;
i0 = 0;
i1 = 1;
i2 = 0;
i3 = 0;
#50;
$display("TC06");
if( d0 != i1 ) $display("Result is wrong");

s0 = 1;
s1 = 1;
i0 = 0;
i1 = 0;
i2 = 0;
i3 = 0;
#50;
$display("TC07");
if( d0 != 0 ) $display("Result is wrong");

s0 = 1;
```

```
        s1 = 1;
        i0 = 1;
        i1 = 0;
        i2 = 0;
        i3 = 0;
        #50;
        $display("TC08");
        if( d0 != i0 ) $display("Result is wrong");


    end


endmodule
```

viii



Part2 – Behaviioral