Spring 2021

California State University, Northridge

Department of Electrical & Computer Engineering

Experiment 6

Scalable Multiplexor

March 24, 2021

ECE 526L

Written By: Jose Luis Martinez

Introduction

In this lab we will design a multiplexor with variable inputs and outputs. We can do this by creating a parameter in the module where your multiplexor is. You would make the default parameter value a 1 and then when you are making instances of that module you can specify the parameter value if you want. For this lab we will test three different ways of redefining the parameter value and then testing out different inputs for the Scalable Multiplexor including 3 test cases when SEL is 1'bx.

Methodology

I followed the following steps in order to complete the lab:

1. First I created a scalable multiplexor module in the sca.mux.v file shown in **Fig. 1.**
    a. A parameter called SIZE is made at the beginning of the module in order to change the size of its inputs/outputs.
    b. The inputs A & B and output OUT are sized accordingly to the SIZE parameter using [SIZE-1:0].
    c. Then for the logic this statement was made: assign OUT = SEL ? B : A
2. Then a test bench verilog file was created in order to test the module shown in **Fig. 2.**
    a. Reg's were made for the inputs and wires for the outputs.
    b. Four instances of the sca_mux module were made. The modules were instantiated as follows:
        i. m1 was instantiated without redefining the parameter.
        ii. m4 was instantiated with #(4) for the parameter.
        iii. m5 was instantiated with #(.SIZE(5)) for the parameter.
        iv. m6 was instantiated normally but right after the parameter is changed with defparam m6.SIZE = 6;
    c. Then the inputs A, B, and SEL were changed with delays in between in order to see the output.
3. Then the verilog files were compiled and then simulated with the results shown in **Fig. 4.**
4. Waveforms were then simulated as shown in **Fig. 5.**

Results/Verilog Files

sca_mux.v | Lab6_tb.v | compileV.f

```
/*****************************************************************************
***                                                                        ***
*** ECE 526 L Experiment #6            Jose Luis Martinez, Spring, 2021    ***
***                                                                        ***
*** Experiment 6 - Scalable Mux                                           ***
***                                                                        ***
*****************************************************************************
*** Filename: sca_mux.v       Created by Jose Luis Martinez, March 24, 2021 ***
***                                                                        ***
*****************************************************************************/
`timescale 1ns/100ps

module sca_mux(A, B, SEL, OUT);
    parameter SIZE = 1;
    input [SIZE-1:0]A;
    input [SIZE-1:0]B;
    input SEL;
    output [SIZE-1:0]OUT;

    assign OUT = SEL ? B : A;
endmodule
```

Fig. 1 sca_mux.v

sca_mux.v | Lab6_tb.v | compileV.f

```
*****************************************************************************
*** Filename: Lab6_tb.v       Created by Jose Luis Martinez, March 24, 2021 ***
***                                                                        ***
*****************************************************************************/
`timescale 1ns/100ps

`define DISPLAY_STRING "%d, A = %b, B = %b, SEL = %b, | OUT1 = %b, OUT4 = %b, OUT5

module Lab6_tb();
    reg [5:0]A;
    reg [5:0]B;
    reg SEL;
    wire OUT1;
    wire [3:0]OUT4;
    wire [4:0]OUT5;
    wire [5:0]OUT6;

    sca_mux m1(.A(A), .B(B), .SEL(SEL), .OUT(OUT1));
    sca_mux #(4) m4(.A(A), .B(B), .SEL(SEL), .OUT(OUT4));
    sca_mux #(.SIZE(5)) m5(.A(A), .B(B), .SEL(SEL), .OUT(OUT5));
    sca_mux m6(.A(A), .B(B), .SEL(SEL), .OUT(OUT6));
        defparam m6.SIZE = 6;

    initial
    begin
        $vcdpluson;
        A <= 6'b10_0100;
        B <= 6'b11_0011;
```

```
35          SEL <= 1'b0;
36          #(20) $display(`DISPLAY_STRING, $time, A, B, SEL, OUT1, OUT4, OUT5, OUT6);
37          A <= 6'b10_1100;
38          B <= 6'b11_0111;
39          SEL <= 1'b0;
40          #(20) $display(`DISPLAY_STRING, $time, A, B, SEL, OUT1, OUT4, OUT5, OUT6);
41          A <= 6'b10_0100;
42          B <= 6'b11_0011;
43          SEL <= 1'b1;
44          #(20) $display(`DISPLAY_STRING, $time, A, B, SEL, OUT1, OUT4, OUT5, OUT6);
45          A <= 6'b10_1100;
46          B <= 6'b11_0111;
47          SEL <= 1'b1;
48          #(20) $display(`DISPLAY_STRING, $time, A, B, SEL, OUT1, OUT4, OUT5, OUT6);
49
50          A <= 6'b01_0111;
51          B <= 6'b10_1001;
```

```
52          #(20) $display(`DISPLAY_STRING, $time, A, B, SEL, OUT1, OUT4, OUT5, OUT6);
53          A <= 6'b01_1001;
54          B <= 6'b01_1001;
55          SEL <= 1'b0;
56          #(20) $display(`DISPLAY_STRING, $time, A, B, SEL, OUT1, OUT4, OUT5, OUT6);
57          A <= 6'b01_0111;
58          B <= 6'b10_1001;
59          SEL <= 1'b1;
60          #(20) $display(`DISPLAY_STRING, $time, A, B, SEL, OUT1, OUT4, OUT5, OUT6);
61          A <= 6'b01_1001;
62          B <= 6'b01_1001;
63          SEL <= 1'b1;
64          #(20) $display(`DISPLAY_STRING, $time, A, B, SEL, OUT1, OUT4, OUT5, OUT6);
65
66          A <= 6'b01_1001;
67          B <= 6'b01_1001;
68          SEL <= 1'bx;
69          #(20) $display(`DISPLAY_STRING, $time, A, B, SEL, OUT1, OUT4, OUT5, OUT6);
70          A <= 6'b01_1001;
71          B <= 6'b10_0110;
72          SEL <= 1'bx;
73          #(20) $display(`DISPLAY_STRING, $time, A, B, SEL, OUT1, OUT4, OUT5, OUT6);
74          A <= 6'b01_1001;
75          B <= 6'b11_1000;
76          SEL <= 1'bx;
77          #(20) $display(`DISPLAY_STRING, $time, A, B, SEL, OUT1, OUT4, OUT5, OUT6);
78          $finish;
79      end
80
81  endmodule
```

Fig. 2 Lab6_tb.v

```
sca_mux m1(.A(A), .B(B), .SEL(SEL), .OUT(OUT1));
sca_mux #(4) m4(.A(A), .B(B), .SEL(SEL), .OUT(OUT4));
sca_mux #(.SIZE(5)) m5(.A(A), .B(B), .SEL(SEL), .OUT(OUT5));
sca_mux m6(.A(A), .B(B), .SEL(SEL), .OUT(OUT6));
    defparam m6.SIZE = 6;
```

Fig. 3 Parameter Redefinition Methods

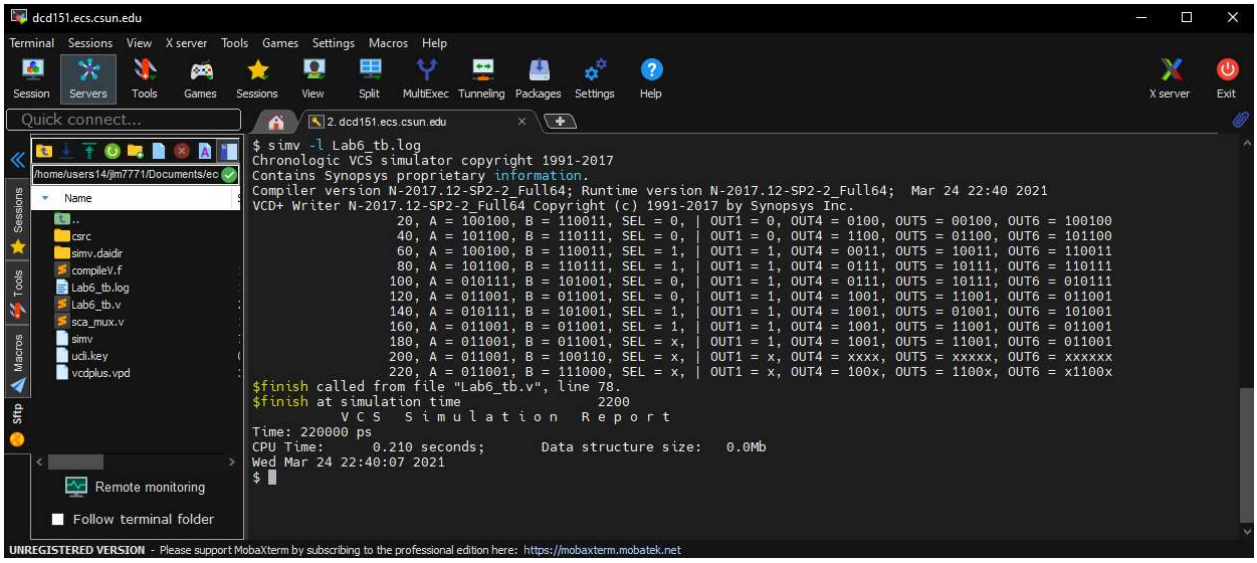

Fig. 4 simv Output

Fig. 5 Waveforms

Analysis

From our simv output in **Fig. 4** and waveform output in **Fig. 5** we can see that our results match the logic of what a multiplexor should behave like.

If we take a look at the first inputs we can see that our A = 6'b10_0100, B=6'11_0011, and SEL = 0. According to this our output should be whatever A is and as we can see in **Fig. 4** our OUT6 = 10_0100, OUT5 = 0_0100, OUT4 = 0100, and OUT = 0. As you can see the output is truncated as we reduce the size of the muxes.

If we take a look at the first inputs we can see that our A = 6'b10_0100, B=6'11_0011, and SEL = 1. According to this our output should be whatever A is and as we can see in **Fig. 4** our OUT6 = 11_0011, OUT5 = 1_0011, OUT4 = 0011, and OUT = 1. As you can see the output is truncated as we reduce the size of the muxes.

If we take a look at the first inputs we can see that our A = 6'b01_1001, B=6'01_1001, and SEL = X. According to this our output should be either the value of a bit it is the same as A and B or x when otherwise. As we can see in **Fig. 4** our OUT6 = 01_1001, OUT5 = 1_1001, OUT4 = 1001, and OUT = 1. As you can see the output is truncated as we reduce the size of the muxes.

If we take a look at the first inputs we can see that our A = 6'b01_1001, B=6'10_0110, and SEL = X. According to this our output should be either the value of a bit it is the same as A and B or x when otherwise. As we can see in **Fig. 4** our OUT6 = XX_XXXX, OUT5 = X_XXXX, OUT4 = XXXX, and OUT = X. As you can see the output is truncated as we reduce the size of the muxes.

If we take a look at the first inputs we can see that our A = 6'b01_1001, B=6'11_1000, and SEL = X. According to this our output should be either the value of a bit it is the same as A and B or x when otherwise. As we can see in **Fig. 4** our OUT6 = x1_100x, OUT5 = 1_100x, OUT4 = 100x, and OUT = x. As you can see the output is truncated as we reduce the size of the muxes.

Conclusion

In conclusion we learned how to create a scalable multiplexor with varying sizes for the inputs. From our results we can see that we were able to successfully implement the scalable multiplexor.

Academic Dishonesty

Submitting any report that is not entirely your own work is a form of academic dishonesty and will not be tolerated. Each and every lab report must include the following statement, signed and dated by the student. Lab reports without the statement will be summarily rejected.

I hereby attest that this lab report is entirely my own work. I have not copied either code or text from anyone, nor have I allowed or will I allow anyone to copy my work.

Name (printed) _____

Name (signed) _____ Date _____