

CALIFORNIA STATE UNIVERSITY, NORTHRIDGE

Department of Electrical and Computer Engineering

ECE 526L

LAB – 7: Simulation of scalable multiplexer using Synopsys VCS in Linux OS environment.

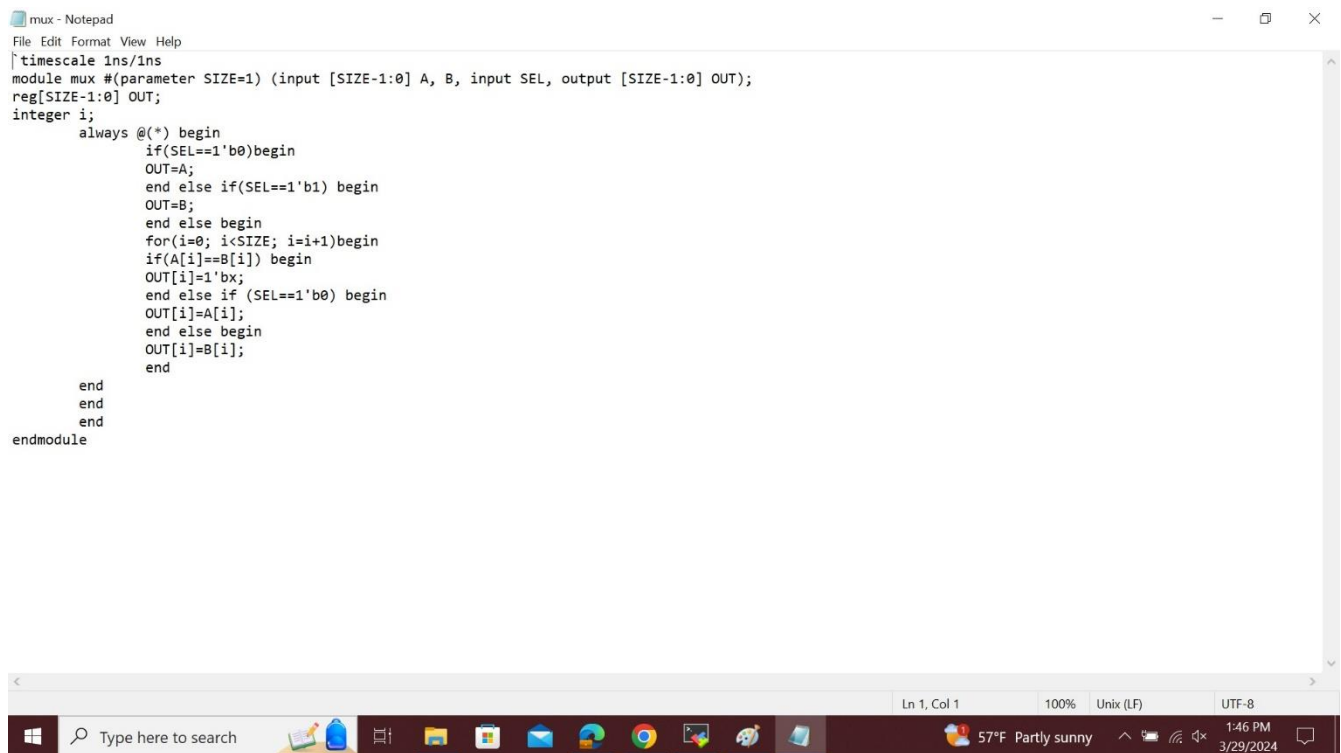
Written By: Raj Kumar

Introduction:

In this lab, we will create a Verilog model of a scalable multiplexer using **Synopsys VCS** in **Linux OS** environment and different terminal commands.

Methodology:

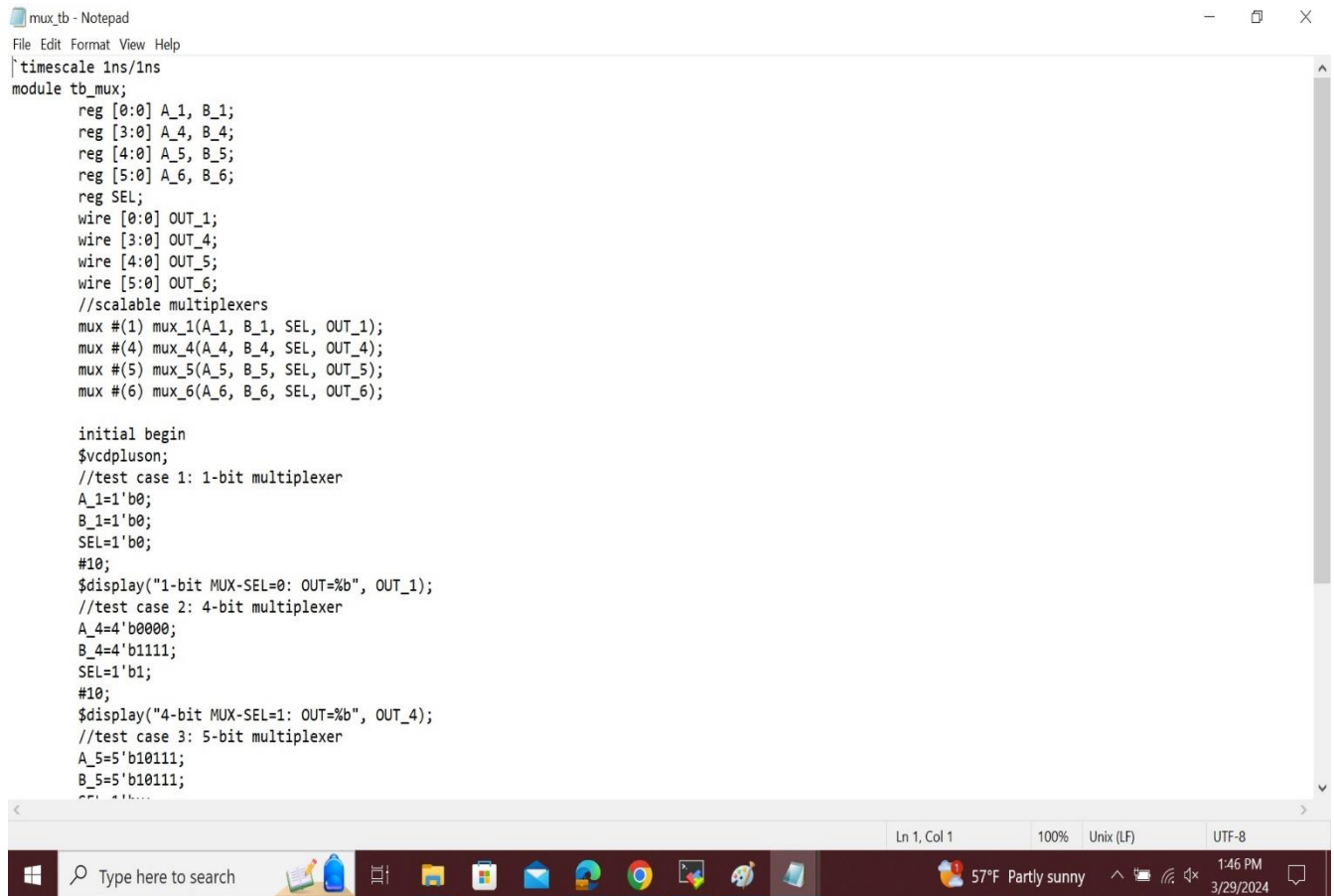
In this lab, we will create a Verilog model of a scalable multiplexer using the given module and steps in assignment we should write a design code for the scalable multiplexer and save it as “**mux.v**” in a new folder named “**Lab7**”. The module name and file name should be consistent.



```
File Edit Format View Help
|timescale 1ns/1ns
module mux #(parameter SIZE=1) (input [SIZE-1:0] A, B, input SEL, output [SIZE-1:0] OUT);
reg[SIZE-1:0] OUT;
integer i;
    always @(*) begin
        if(SEL==1'b0)begin
            OUT=A;
        end else if(SEL==1'b1) begin
            OUT=B;
        end else begin
            for(i=0; i<SIZE; i=i+1)begin
                if(A[i]==B[i]) begin
                    OUT[i]=1'bx;
                end else if (SEL==1'b0) begin
                    OUT[i]=A[i];
                end else begin
                    OUT[i]=B[i];
                end
            end
        end
    end
endmodule
```

This is the figure of **Design code “mux.v”**.

Now, we should write the test bench code for the design code. We will create four instances of scalable multiplexer by using 1,4,5 and 6 for different widths and consider the points given in assignment and that should be saved as “mux_tb.v” shown in below figure.



```
mux_tb - Notepad
File Edit Format View Help
timescale 1ns/1ns
module tb_mux;
    reg [0:0] A_1, B_1;
    reg [3:0] A_4, B_4;
    reg [4:0] A_5, B_5;
    reg [5:0] A_6, B_6;
    reg SEL;
    wire [0:0] OUT_1;
    wire [3:0] OUT_4;
    wire [4:0] OUT_5;
    wire [5:0] OUT_6;
    //scalable multiplexers
    mux #(1) mux_1(A_1, B_1, SEL, OUT_1);
    mux #(4) mux_4(A_4, B_4, SEL, OUT_4);
    mux #(5) mux_5(A_5, B_5, SEL, OUT_5);
    mux #(6) mux_6(A_6, B_6, SEL, OUT_6);

    initial begin
        $vcdpluson;
        //test case 1: 1-bit multiplexer
        A_1=1'b0;
        B_1=1'b0;
        SEL=1'b0;
        #10;
        $display("1-bit MUX-SEL=0: OUT=%b", OUT_1);
        //test case 2: 4-bit multiplexer
        A_4=4'b0000;
        B_4=4'b1111;
        SEL=1'b1;
        #10;
        $display("4-bit MUX-SEL=1: OUT=%b", OUT_4);
        //test case 3: 5-bit multiplexer
        A_5=5'b10111;
        B_5=5'b10111;
        SEL=1'b0;
        #10;
        $display("5-bit MUX-SEL=0: OUT=%b", OUT_5);
    end
endmodule
```

Now compile the two files by using the command line as follows,

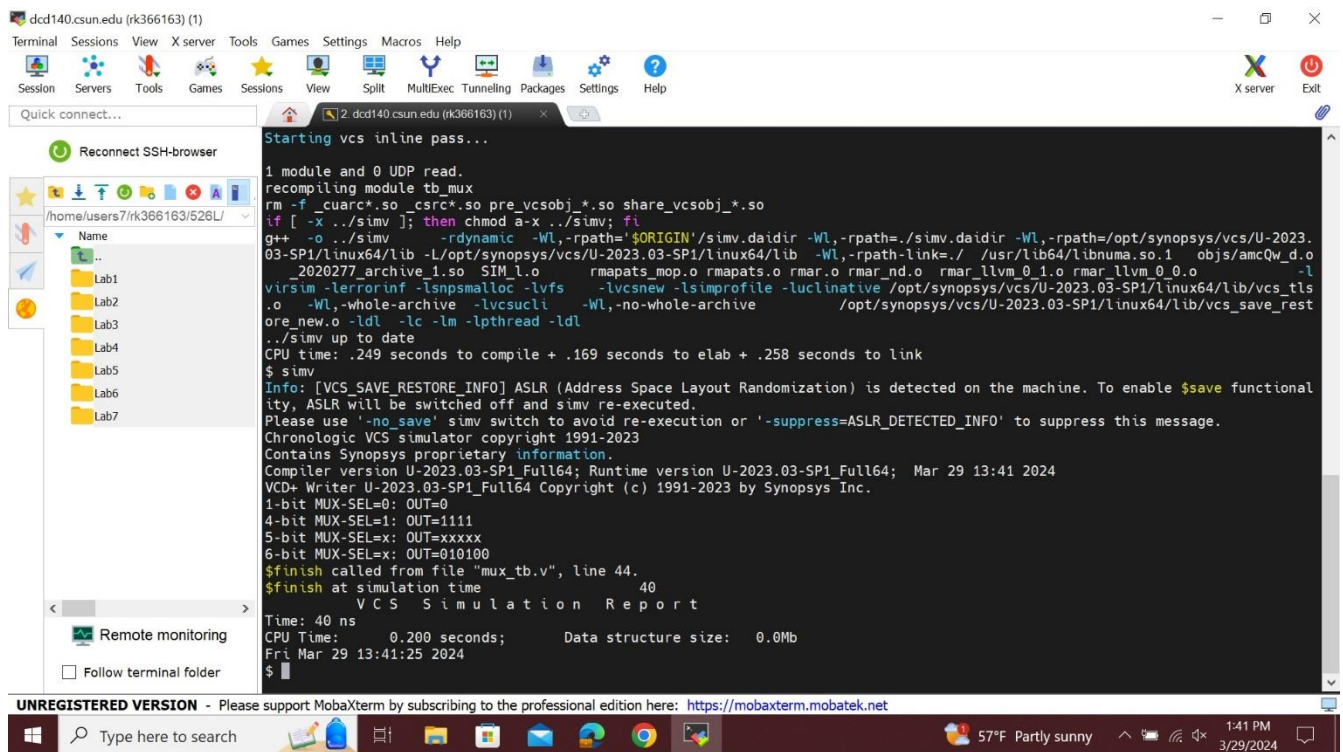
“vcs -debug_access+all mux.v mux_tb.v”

To start the simulator type **“simv -l Lab7.log”**, by using this command we will get logfile and **simv** outcome.

The **Log File** which I created is as shown in the figure in the next page.

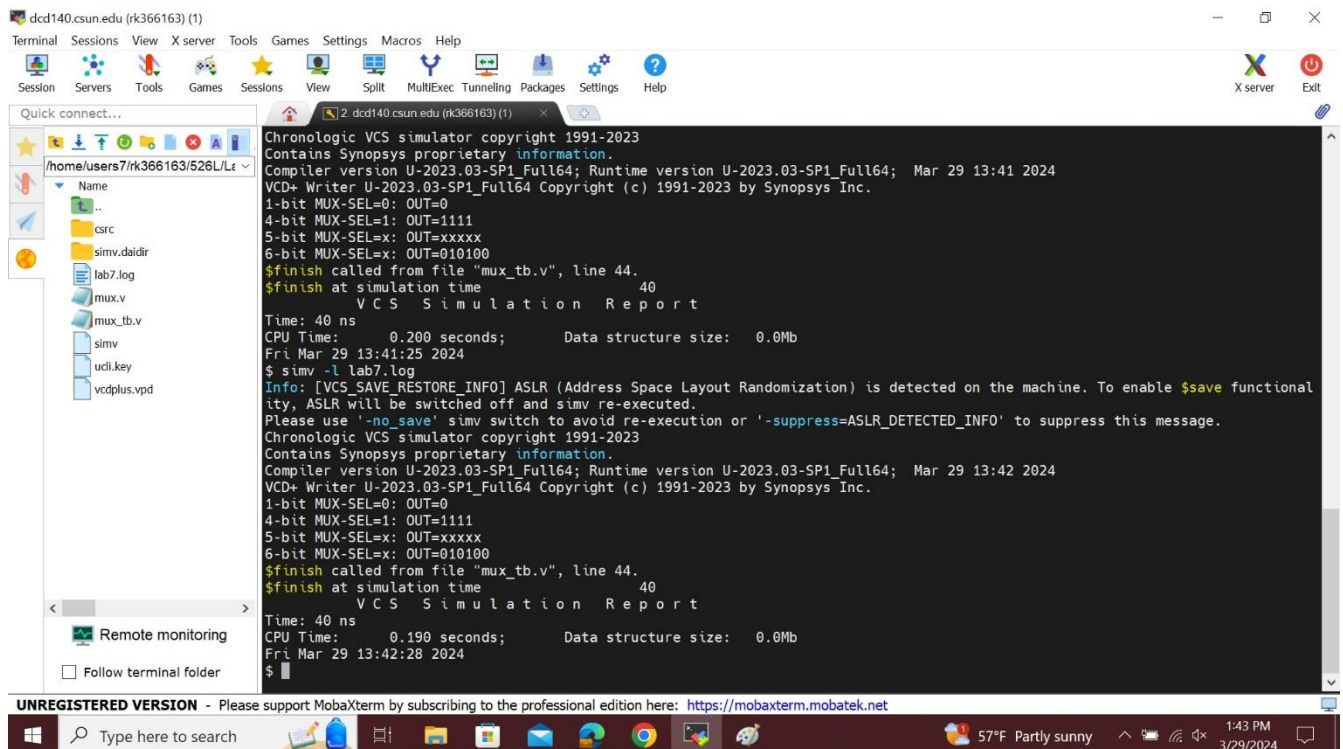
If we don't see any error messages if everything is typed correctly then output should look like this as shown in below figure.

simv:



```
dcd140.csun.edu (rk366163) (1)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
Quick connect...
Reconnect SSH-browser
/home/users7/rk366163/526L/
Name
Lab1
Lab2
Lab3
Lab4
Lab5
Lab6
Lab7
Remote monitoring
Follow terminal folder
Starting vcs inline pass...
1 module and 0 UDP read.
recompiling module tb_mux
rm -f _cuarc*.so _csrc*.so pre_vcsobj*.so share_vcsobj*.so
if [ -x ../simv ]; then chmod a-x ../simv; fi
g++ -o ../simv -rdynamic -Wl,-rpath='$_ORIGIN'/simv.daidir -Wl,-rpath=../simv.daidir -Wl,-rpath=/opt/synopsys/vcs/U-2023.03-SP1/linux64/lib -L/opt/synopsys/vcs/U-2023.03-SP1/linux64/lib -Wl,-rpath-link=/usr/lib64/libnuma.so.1 objs/amcQw_d.o
2020277_archive_1.so SIM_lo rmapats_mop.o rmapats.o rmar.o rmar_nd.o rmar_llvm_0_1.o rmar_llvm_0_0.o -l
virsim -lerrorinf -lsnpsmalloc -lvfs -lvcsnew -lsimprofile -luclnativ /opt/synopsys/vcs/U-2023.03-SP1/linux64/lib/vcs_tls
.o -Wl,-whole-archive -lvcsucli -Wl,-no-whole-archive /opt/synopsys/vcs/U-2023.03-SP1/linux64/lib/vcs_save_rest
ore_new.o -ldl -lc -lm -lpthread -ldl
../simv up to date
CPU time: .249 seconds to compile + .169 seconds to elab + .258 seconds to link
$ simv
Info: [VCS_SAVE_RESTORE_INFO] ASLR (Address Space Layout Randomization) is detected on the machine. To enable $save functional
ity, ASLR will be switched off and simv re-executed.
Please use '-no_save' simv switch to avoid re-execution or '-suppress=ASLR_DETECTED_INFO' to suppress this message.
Chronologic VCS simulator copyright 1991-2023
Contains Synopsys proprietary information.
Compiler version U-2023.03-SP1_Full64; Runtime version U-2023.03-SP1_Full64; Mar 29 13:41 2024
VCD+ Writer U-2023.03-SP1_Full64 Copyright (c) 1991-2023 by Synopsys Inc.
1-bit MUX-SEL=0: OUT=0
4-bit MUX-SEL=1: OUT=1111
5-bit MUX-SEL=x: OUT=xxxxx
6-bit MUX-SEL=x: OUT=010100
$finish called from file "mux_tb.v", line 44.
$finish at simulation time 40
VCS Simulation Report
Time: 40 ns
CPU Time: 0.200 seconds; Data structure size: 0.0Mb
Fri Mar 29 13:41:25 2024
$
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net
Type here to search 57°F Partly sunny 1:41 PM 3/29/2024
```

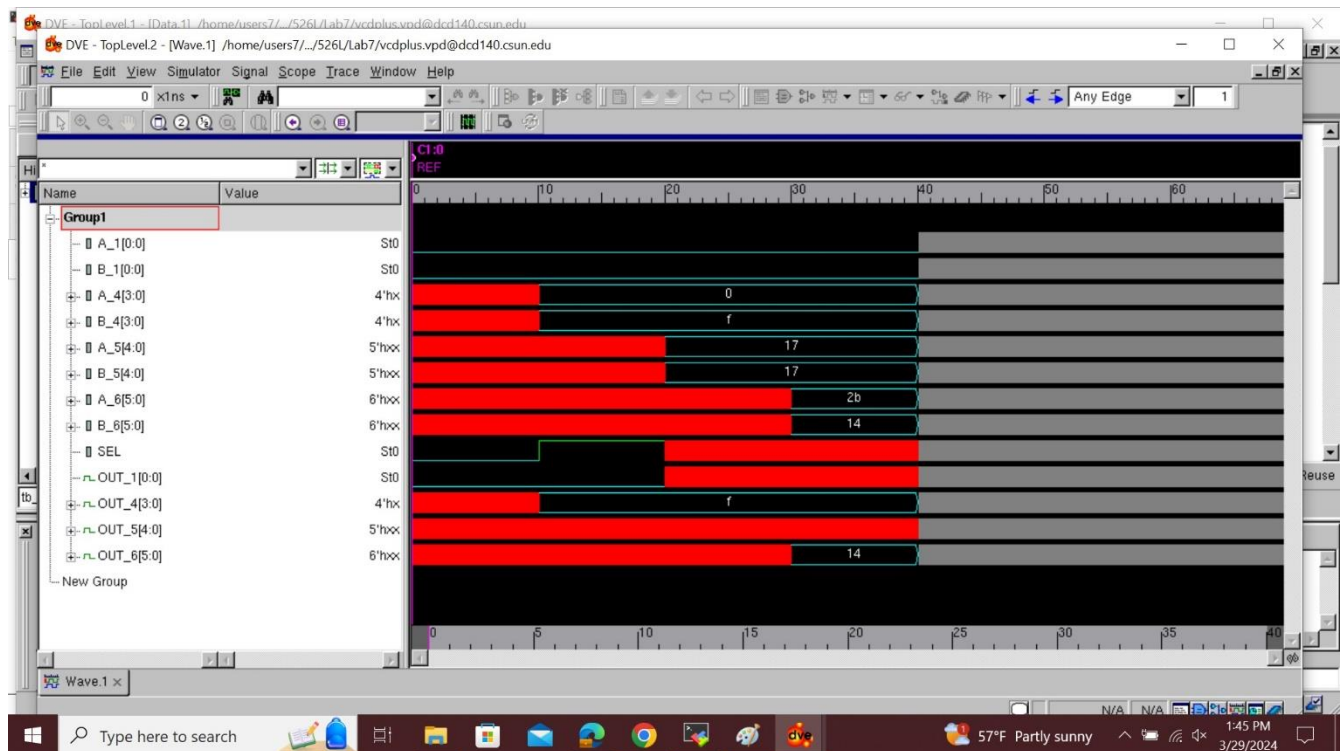
Log File:



```
dcd140.csun.edu (rk366163) (1)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
Quick connect...
Reconnect SSH-browser
/home/users7/rk366163/526L/Lt
Name
csrc
simv.daidir
lab7.log
mux.v
mux_tb.v
simv
ucln.key
vcdplus.vpd
Remote monitoring
Follow terminal folder
Chronologic VCS simulator copyright 1991-2023
Contains Synopsys proprietary information.
Compiler version U-2023.03-SP1_Full64; Runtime version U-2023.03-SP1_Full64; Mar 29 13:41 2024
VCD+ Writer U-2023.03-SP1_Full64 Copyright (c) 1991-2023 by Synopsys Inc.
1-bit MUX-SEL=0: OUT=0
4-bit MUX-SEL=1: OUT=1111
5-bit MUX-SEL=x: OUT=xxxxx
6-bit MUX-SEL=x: OUT=010100
$finish called from file "mux_tb.v", line 44.
$finish at simulation time 40
VCS Simulation Report
Time: 40 ns
CPU Time: 0.190 seconds; Data structure size: 0.0Mb
Fri Mar 29 13:42:28 2024
$
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net
Type here to search 57°F Partly sunny 1:43 PM 3/29/2024
```

To see the wave form first type “**dve**” in command line then we will get the blank **simulation window** => **then go to file => open database => choose vcdplus.vpd => open the file.**

Then finally select all the signals with the mouse, then right click and select to “**add to wave => new wave view**”. Then we will get the waveform as shown in below figure.



Analysis of Result:

Scalable multiplexer refers to a digital circuit that can select one of multiple input signals and route it to the output based on a control signal. The result for a scalable multiplexer in Verilog is typically defined using a case statement or an if-else construct in given lab we have taken 4 instances in different cases depending upon the input values we provide and the sel value, the output value varies. By increasing the number of input signals and utilizing the proper control signal width, we may scale this idea to construct larger multiplexers, such as 4:1, 8:1, or even larger multiplexers.

Conclusion:

In this lab, we simulated scalable multiplexer using Synopsys VCS in Linux OS environment and got the required wave-form.

I hereby attest that this lab report is entirely my own work. I have not copied either code or text from anyone, neither have I allowed nor I will let anyone to copy my work.

Name(printed) Raj Kumar

Name(signed) Raj Kumar

Date 04/05/2024