

Spring 2021
California State University, Northridge
Department of Electrical & Computer Engineering



Experiment 1
Familiarization with Linux and Synopsys VCS
January 28, 2021
ECE 526L

Written By: Jose Luis Martinez

Introduction

In experiment 1 our goal was to familiarize ourselves with using Linux and Synopsys by testing out sample code that was given to us.

Methodology

Being the first experiment of the semester there was nothing to design as we were more or less just following the laboratory manual.

The following are the steps I took from using the manual.

1. Connect to CSUN via the VPN program Global Protect.
2. Open MobaXterm and connect to one of the remote CSUN computers.
3. Created a directory called ECE526Spring21 in the documents directory.
4. Created another directory called Lab1.
5. I then created “mux4_1.v” and “mux4_1_tb.v” on my own computer.
6. I uploaded these files from my computer into the directory I created in the remote computer.
7. I then compiled the files by typing “vcs -debug -full64 mux4_1.v mux4_1_tb.v”.
8. I then verified my code was working by typing simv.
9. To save the log file I typed “simv -l Lab1.log”.
10. To then see the waveform I typed “dve -full64 &”.
11. Then I opened the file “vcdplus.vpd” and selected the variables and right clicked “Add to Wave -> New Wave View”.
12. A new window opened up with the waveforms and I screenshot the results.
13. I exited DVE and downloaded all of the files in the directory Lab1 to my computer.

Results

mux4_1.v

```
1  /*****
2  ***
3  *** ECE 526 L Experiment #1           Jose Luis Martinez, Spring, 2021 ***
4  ***
5  *** Familiarization with Linux and Synopsys VCS
6  ***
7  ****
8  *** Filename: mux4_1.v   Created by Jose Luis Martinez, January 29, 2021 ***
9  ***
10 ****/
11
12 `timescale 1 ns / 1 ns
13
```

```

14 module mux4_1(out, a, b, c, d, s0, s1);
15
16 output out;
17 input a, b, c, d, s0, s1;
18 wire s0_n, s1_n, T1, T2, T3, T4;
19
20 not (s0_n, s0), (s1_n, s1);
21 and (T1, a, s0_n, s1_n), (T2, b, s0_n, s1), (T3, c, s0, s1_n), (T4, d, s0, s1);
22 or(out, T1, T2, T3, T4);
23
24 endmodule

```

mux4_1_tb.v

```

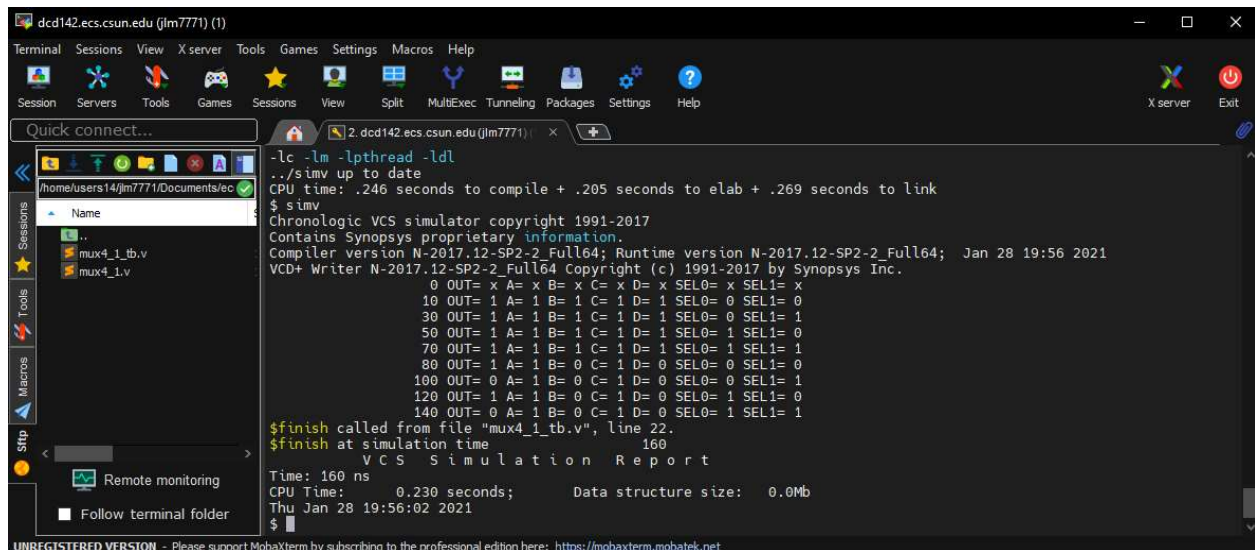
1  /*****
2  ***
3  *** ECE 526 L Experiment #1          Jose Luis Martinez, Spring, 2021 ***
4  ***
5  *** Familiarization with Linux and Synopsys VCS          ***
6  ***
7  *****/
8  *** Filename: mux4_1_tb.v    Created by Jose Luis Martinez, January 29, 2021*
9  ***
10 *****/
11
12 `timescale 1 ns / 1 ns
13
14 module mux4_1_tb();
15     reg A,B,C,D, SEL0, SEL1;
16     wire OUT;
17
18     mux4_1 uut(OUT, A, B, C, D, SEL0, SEL1);
19
20     initial
21     $monitorb ("%d OUT= %d A= %d B= %d C= %d D= %d SEL0= %d SEL1= %d",
22         $time, OUT, A, B, C, D, SEL0, SEL1);
23     initial begin
24         $vcdpluson;
25         #10 A= 1; B= 1; C= 1; D= 1; SEL0=0; SEL1= 0;
26         #20 SEL0=0; SEL1=1;
27         #20 SEL0=1; SEL1=0;
28         #20 SEL0=1; SEL1=1;
29         #10 A= 1; B= 0; C= 1; D= 0; SEL0=0; SEL1= 0;
30         #20 SEL0=0; SEL1=1;
31         #20 SEL0=1; SEL1=0;
32         #20 SEL0=1; SEL1=1;
33         #20 $finish;
34     end
35 endmodule

```

Lab1.log

```
1 Command: /home/users14/jlm7771/Documents/ece526LabSpring21/Lab1/./simv -l Lab1.log
2 Chronologic VCS simulator copyright 1991-2017
3 Contains Synopsys proprietary information.
4 Compiler version N-2017.12-SP2-2_Full64; Runtime version N-2017.12-SP2-2_Full64; Jan
  28 20:06 2021
5 VCD+ Writer N-2017.12-SP2-2_Full64 Copyright (c) 1991-2017 by Synopsys Inc.
6      0 OUT= x A= x B= x C= x D= x SEL0= x SEL1= x
7      10 OUT= 1 A= 1 B= 1 C= 1 D= 1 SEL0= 0 SEL1= 0
8      30 OUT= 1 A= 1 B= 1 C= 1 D= 1 SEL0= 0 SEL1= 1
9      50 OUT= 1 A= 1 B= 1 C= 1 D= 1 SEL0= 1 SEL1= 0
10     70 OUT= 1 A= 1 B= 1 C= 1 D= 1 SEL0= 1 SEL1= 1
11     80 OUT= 1 A= 1 B= 0 C= 1 D= 0 SEL0= 0 SEL1= 0
12    100 OUT= 0 A= 1 B= 0 C= 1 D= 0 SEL0= 0 SEL1= 1
13    120 OUT= 1 A= 1 B= 0 C= 1 D= 0 SEL0= 1 SEL1= 0
14    140 OUT= 0 A= 1 B= 0 C= 1 D= 0 SEL0= 1 SEL1= 1
15 $finish called from file "mux4_1_tb.v", line 22.
16 $finish at simulation time 160
17     V C S   S i m u l a t i o n   R e p o r t
18 Time: 160 ns
19 CPU Time:      0.220 seconds;      Data structure size:  0.0Mb
20 Thu Jan 28 20:06:55 2021
21
```

simv output

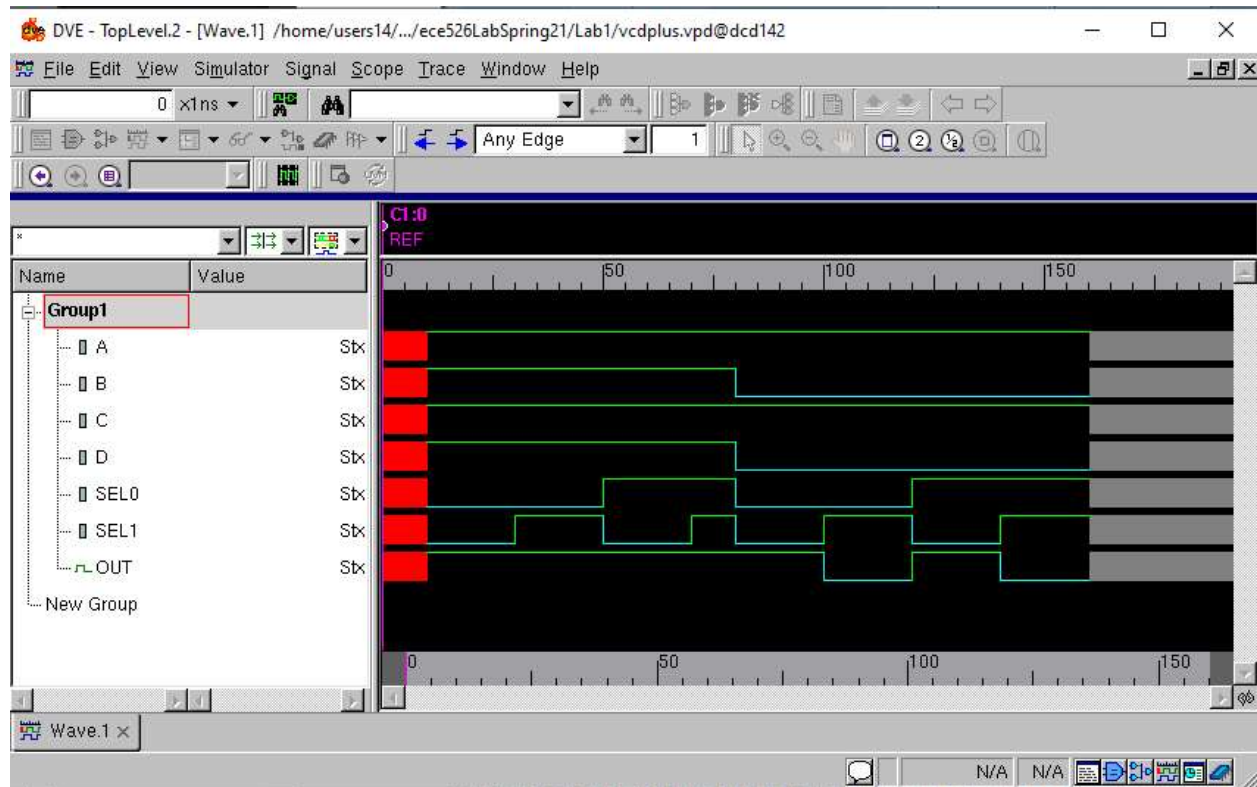


The screenshot shows a MobaXterm window titled "dcd142.ecs.csun.edu (jlm7771) (1)". The terminal displays the following output:

```
-lc -lm -lpthread -ldl
../simv up to date
CPU time: .246 seconds to compile + .205 seconds to elab + .269 seconds to link
$ simv
Chronologic VCS simulator copyright 1991-2017
Contains Synopsys proprietary information.
Compiler version N-2017.12-SP2-2_Full64; Runtime version N-2017.12-SP2-2_Full64; Jan 28 19:56 2021
VCD+ Writer N-2017.12-SP2-2_Full64 Copyright (c) 1991-2017 by Synopsys Inc.
      0 OUT= x A= x B= x C= x D= x SEL0= x SEL1= x
      10 OUT= 1 A= 1 B= 1 C= 1 D= 1 SEL0= 0 SEL1= 0
      30 OUT= 1 A= 1 B= 1 C= 1 D= 1 SEL0= 0 SEL1= 1
      50 OUT= 1 A= 1 B= 1 C= 1 D= 1 SEL0= 1 SEL1= 0
      70 OUT= 1 A= 1 B= 1 C= 1 D= 1 SEL0= 1 SEL1= 1
      80 OUT= 1 A= 1 B= 0 C= 1 D= 0 SEL0= 0 SEL1= 0
     100 OUT= 0 A= 1 B= 0 C= 1 D= 0 SEL0= 0 SEL1= 1
     120 OUT= 1 A= 1 B= 0 C= 1 D= 0 SEL0= 1 SEL1= 0
     140 OUT= 0 A= 1 B= 0 C= 1 D= 0 SEL0= 1 SEL1= 1
$finish called from file "mux4_1_tb.v", line 22.
$finish at simulation time 160
     V C S   S i m u l a t i o n   R e p o r t
Time: 160 ns
CPU Time:      0.230 seconds;      Data structure size:  0.0Mb
Thu Jan 28 19:56:02 2021
$
```

The MobaXterm interface includes a sidebar with "Sessions", "Servers", "Tools", "Games", "Sessions", "View", "Split", "MultiExec", "Tunneling", "Packages", "Settings", and "Help". The "Sessions" list shows a session named "mux4_1_tb.v" with a status of "Running". The terminal output is displayed in the main pane, and the status bar at the bottom indicates "UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>".

Waveforms



Analysis

A, B, C, and D are our inputs into the mux. SEL0 and SEL1 are our selection bits. OUT is the mux output. According to the data we got from simv we can see that the logic matches that of a multiplexer. In our waveforms we can see that OUT goes to logic level LOW when our SEL bits select a multiplexer input that is also logic LOW.

Conclusion

In conclusion we learned how to work with linux and simulate our verilog files using Synopsys. We learned in particular how to compile and generate an output log using simv. We also learned how to view our output waveforms to verify our results.

Extra Credit

1. If we wanted to have an exhaustive test that covers every possible input, how many test cases should we write.
 - a. We would want to test 2^n times with n being the number of inputs.
 - b. For this lab with 2 selection bits and 4 inputs feeding into the mux we would have to test $2^6 = 64$ different inputs.
2. If just for this lab we consider X and Z as possible inputs, then how many test cases we would have?
 - a. If X and Z are also now possible inputs we would test 4^n times with n being the number of inputs.
 - b. For this lab with 2 selection bits and 4 inputs feeding into the mux and considering we have 4 possibilities of input values we would have to test $4^6 = 4096$ different inputs.