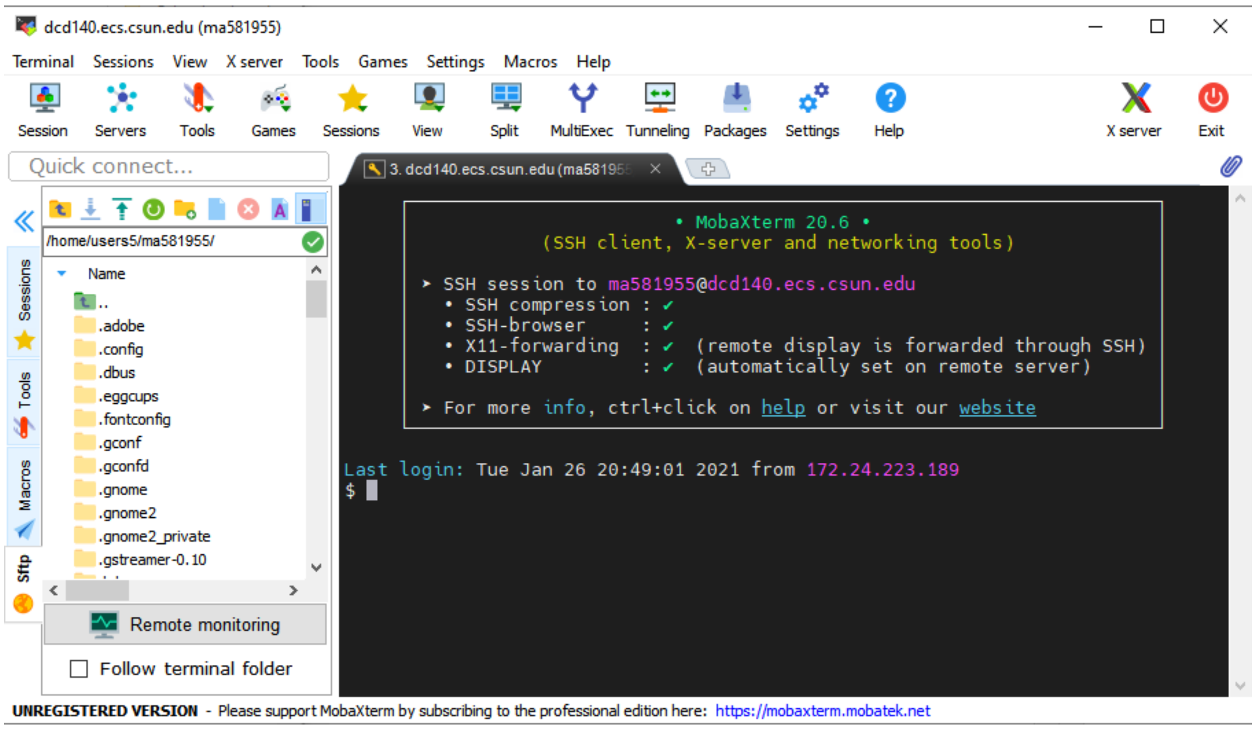
ECE 526L Spring 2021
Lab1
Familiarization with Linux and Synopsys VCS

In this lab you will simulate a simple design using Synopsys VCS in Linux OS environment. Keep in mind that there are multiple ways to achieve the same outcome and if you are more familiar with Linux you can certainly do things your way.

Connect to one of remote computers (dcd130-dcd159) with MobaXterm (there is a module on Canvas with guidelines on how to connect and there is a Lab0 which you should do before this lab).

After you connected you should see a window like this:


dcd140.ecs.csun.edu (ma581955)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help X server Exit
Quick connect...
3. dcd140.ecs.csun.edu (ma58195
/home/users5/ma581955/
Name
..
.adobe
.config
.dbus
.eggcups
.fontconfig
.gconf
.gconfd
.gnome
.gnome2
.gnome2_private
.gstreamer-0.10
Sessions
Tools
Macros
Sftp
Remote monitoring
Follow terminal folder

```
                • MobaXterm 20.6 •
       (SSH client, X-server and networking tools)

 ➤ SSH session to ma581955@dcd140.ecs.csun.edu
   • SSH compression : ✔
   • SSH-browser     : ✔
   • X11-forwarding  : ✔  (remote display is forwarded through SSH)
   • DISPLAY         : ✔  (automatically set on remote server)

 ➤ For more info, ctrl+click on help or visit our website

Last login: Tue Jan 26 20:49:01 2021 from 172.24.223.189
$
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

For this semester create a directory called 526LSP21 (this can be any name you choose, just avoid space in any of the folders or files in linux envinroment)

```
mkdir 526LSP21
```

For each lab you should make a separate directory

```
cd 526LSP21
mkdir Lab1
cd Lab1
```

For future labs you might be in a different directory make sure you navigate to correct directory first, you can find out where you currently are with “pwd” command.

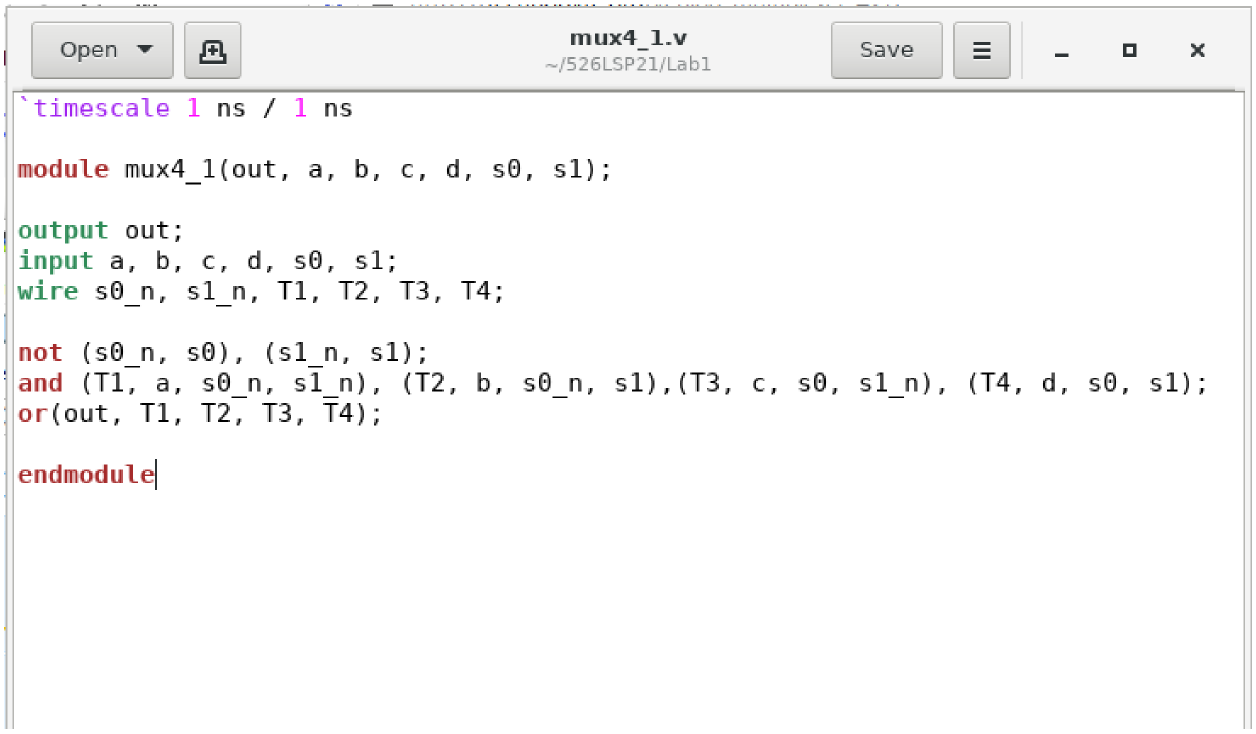
You need to use a text editor for writing Verilog code, if you prefer command line base editor you can use vi and Appendix B (a separate module) has common commands. If you prefer a graphical text editor, you can use gedit.

To invoke gedit but keep the current terminal window active use below command
gedit &
& after any commands keeps the same terminal window active for using with other commands instead of it being just used for the previous command.

Using Windows based text editors such as Microsoft word might add hidden characters to the file that might cause issues later, always use linux based text editors.

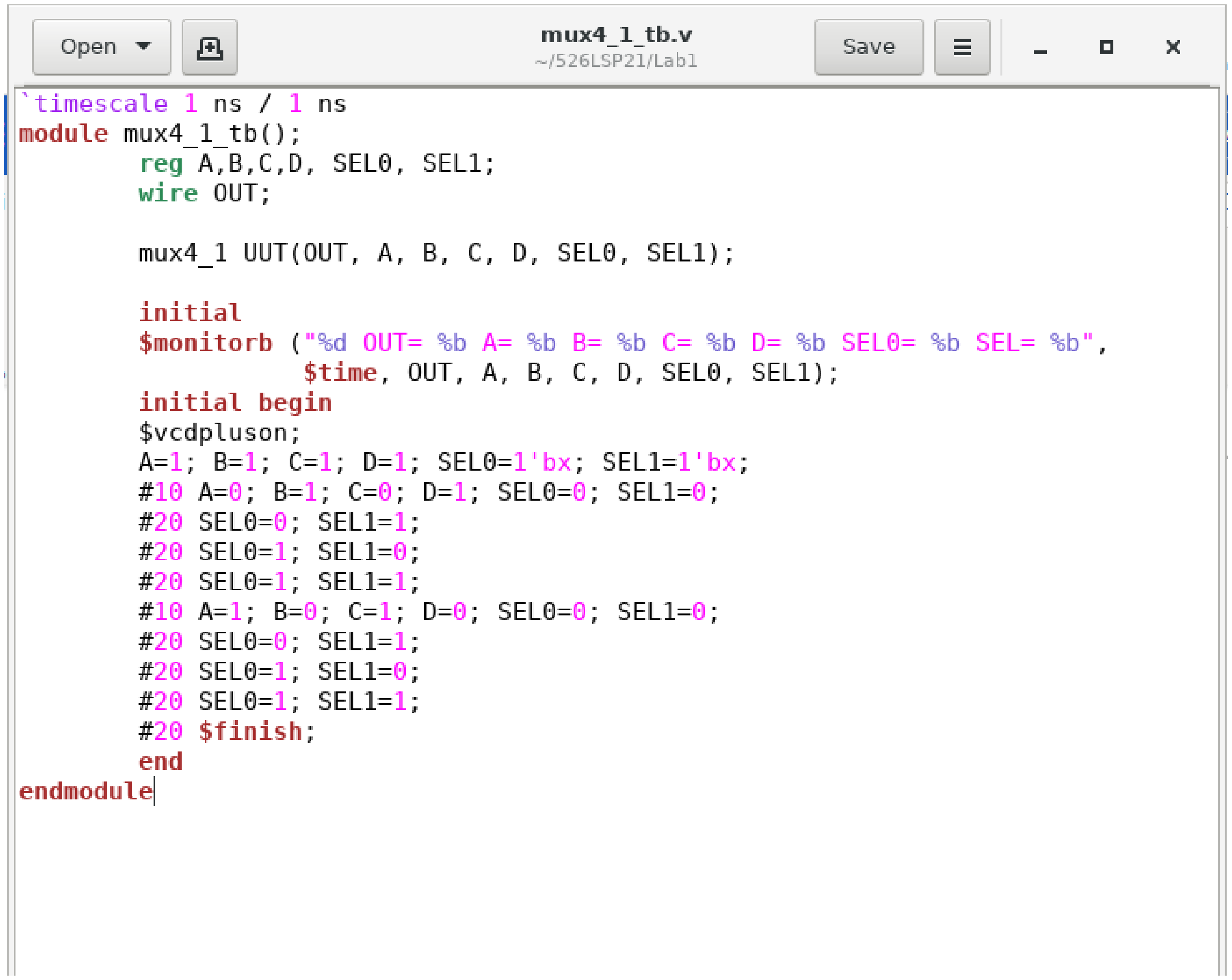
The file should have ".v" extension, if you are writing code in SystemVerilog (future labs) use ".sv" extension.

Write below code in your text editor, this code models a 4 input Mux. Save as mux4_1.v



```
`timescale 1 ns / 1 ns

module mux4_1(out, a, b, c, d, s0, s1);

output out;
input a, b, c, d, s0, s1;
wire s0_n, s1_n, T1, T2, T3, T4;

not (s0_n, s0), (s1_n, s1);
and (T1, a, s0_n, s1_n), (T2, b, s0_n, s1),(T3, c, s0, s1_n), (T4, d, s0, s1);
or(out, T1, T2, T3, T4);

endmodule
```

We need a testbench to verify our design, write below code in your text editor. Save as mux4_1_tb.v



```
`timescale 1 ns / 1 ns
module mux4_1_tb();
        reg A,B,C,D, SEL0, SEL1;
        wire OUT;

        mux4_1 UUT(OUT, A, B, C, D, SEL0, SEL1);

        initial
        $monitorb ("%d OUT= %b A= %b B= %b C= %b D= %b SEL0= %b SEL= %b",
                   $time, OUT, A, B, C, D, SEL0, SEL1);
        initial begin
        $vcdpluson;
        A=1; B=1; C=1; D=1; SEL0=1'bx; SEL1=1'bx;
        #10 A=0; B=1; C=0; D=1; SEL0=0; SEL1=0;
        #20 SEL0=0; SEL1=1;
        #20 SEL0=1; SEL1=0;
        #20 SEL0=1; SEL1=1;
        #10 A=1; B=0; C=1; D=0; SEL0=0; SEL1=0;
        #20 SEL0=0; SEL1=1;
        #20 SEL0=1; SEL1=0;
        #20 SEL0=1; SEL1=1;
        #20 $finish;
        end
endmodule
```

Compile the two files:

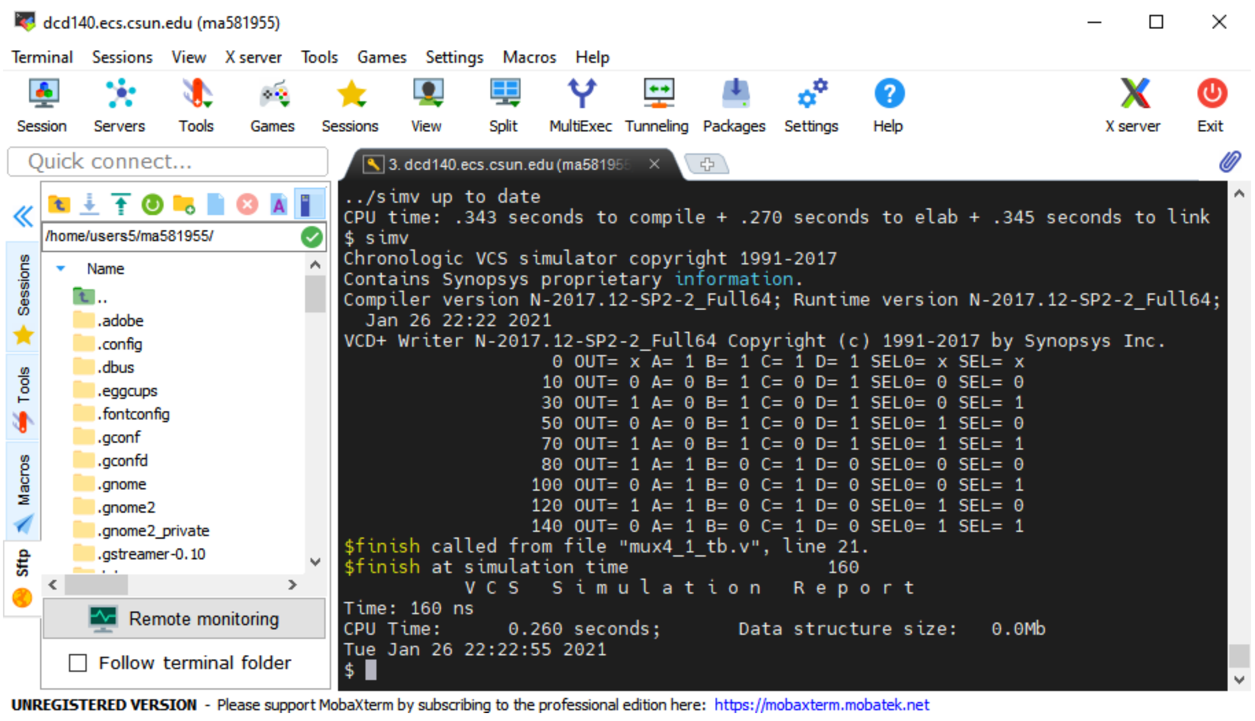
vcs –debug –full64 mux4_1.v mux4_1_tb.v

If you have a systemverilog file add –sverilog switch to the command

Now invoke the simulator:

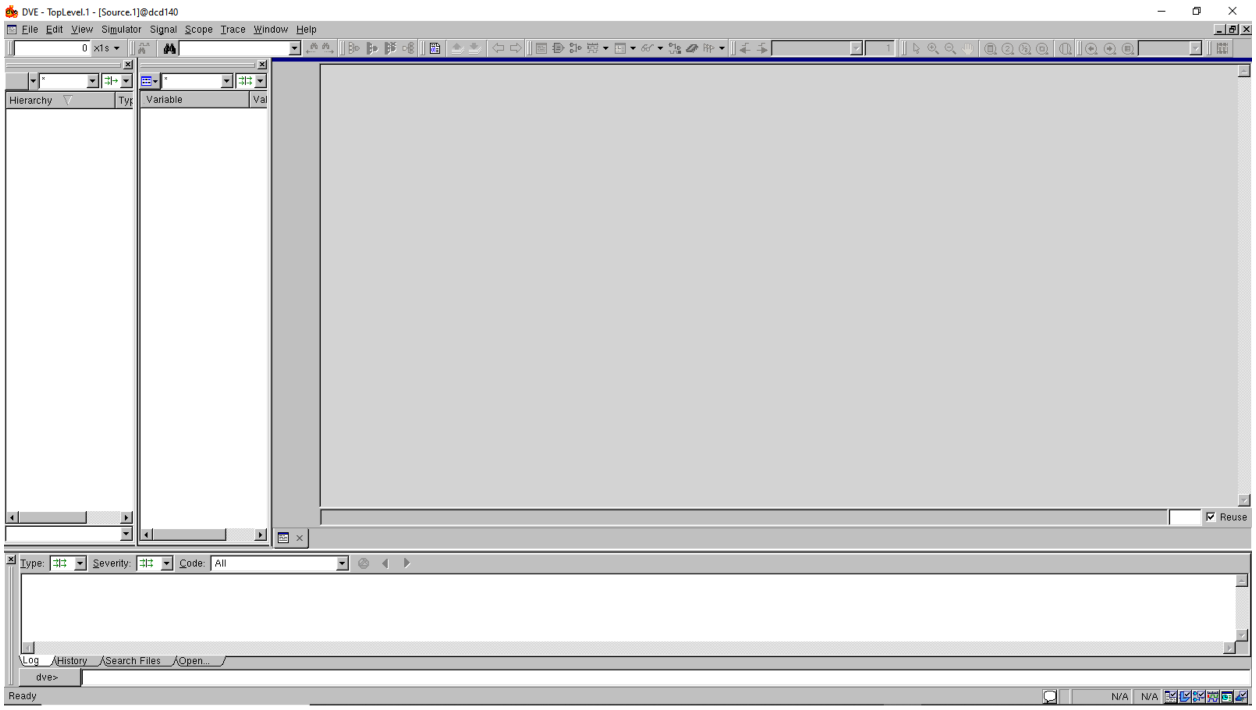
simv

You should not see any error messages if everything is typed correctly. Output should look like this:

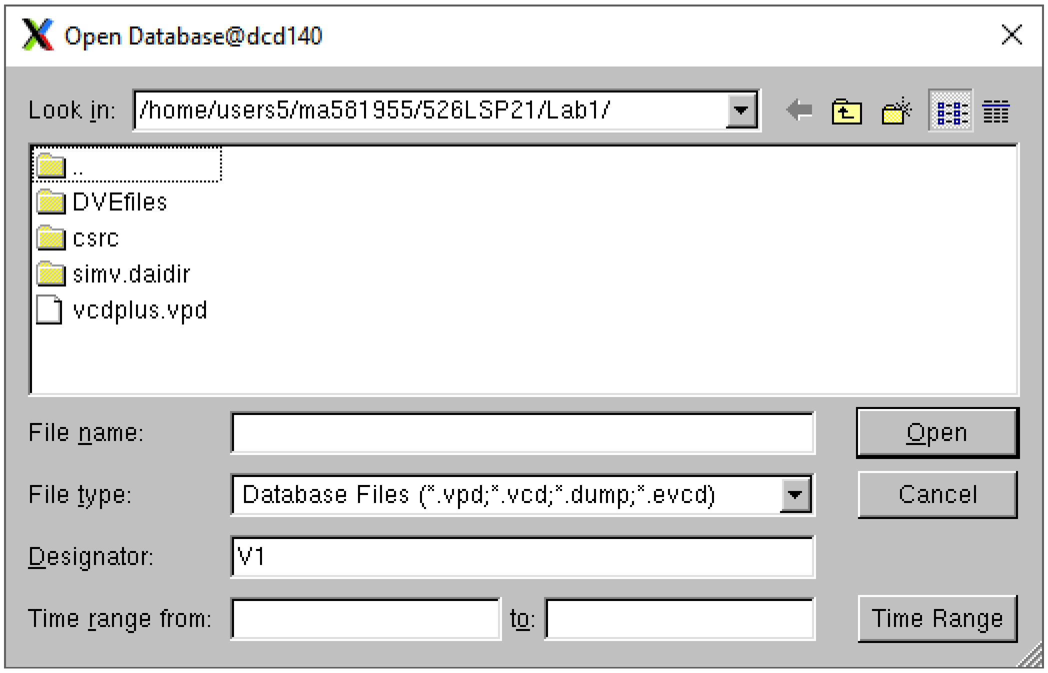

dcd140.ecs.csun.edu (ma581955)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help X server Exit
Quick connect...
3. dcd140.ecs.csun.edu (ma58195
/home/users5/ma581955/
Name
..
.adobe
.config
.dbus
.eggcups
.fontconfig
.gconf
.gconfd
.gnome
.gnome2
.gnome2_private
.gstreamer-0.10
Remote monitoring
Follow terminal folder
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

```
../simv up to date
CPU time: .343 seconds to compile + .270 seconds to elab + .345 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 26 22:22 2021
VCD+ Writer N-2017.12-SP2-2_Full64 Copyright (c) 1991-2017 by Synopsys Inc.
                   0 OUT= x A= 1 B= 1 C= 1 D= 1 SEL0= x SEL= x
                  10 OUT= 0 A= 0 B= 1 C= 0 D= 1 SEL0= 0 SEL= 0
                  30 OUT= 1 A= 0 B= 1 C= 0 D= 1 SEL0= 0 SEL= 1
                  50 OUT= 0 A= 0 B= 1 C= 0 D= 1 SEL0= 1 SEL= 0
                  70 OUT= 1 A= 0 B= 1 C= 0 D= 1 SEL0= 1 SEL= 1
                  80 OUT= 1 A= 1 B= 0 C= 1 D= 0 SEL0= 0 SEL= 0
                 100 OUT= 0 A= 1 B= 0 C= 1 D= 0 SEL0= 0 SEL= 1
                 120 OUT= 1 A= 1 B= 0 C= 1 D= 0 SEL0= 1 SEL= 0
                 140 OUT= 0 A= 1 B= 0 C= 1 D= 0 SEL0= 1 SEL= 1
$finish called from file "mux4_1_tb.v", line 21.
$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:      0.260 seconds;       Data structure size:   0.0Mb
Tue Jan 26 22:22:55 2021
$
```

To create a log file :
simv –l Lab1.log
To see the waveform (invoked by using “$vcdpluson” in testbench file and using “–debug” in vcs command) first type:
dve –full64 &
You should see the blank simulation window like this:



Go to File -> Open Database



Open Database@dcd140

Look in: /home/users5/ma581955/526LSP21/Lab1/

```
..
DVEfiles
csrc
simv.daidir
vcdplus.vpd
```

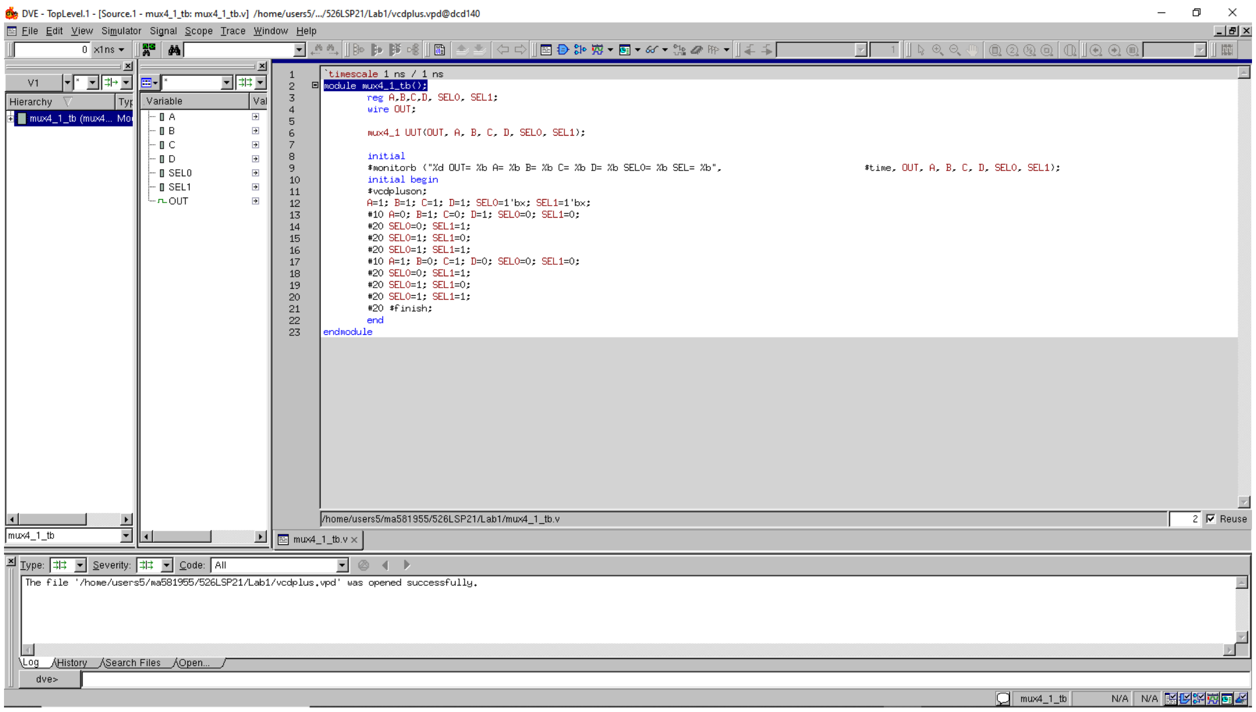
File name:

File type: Database Files (*.vpd;*.vcd;*.dump;*.evcd)

Designator: V1

Time range from: to:

Choose vcdplus.vpd and open.



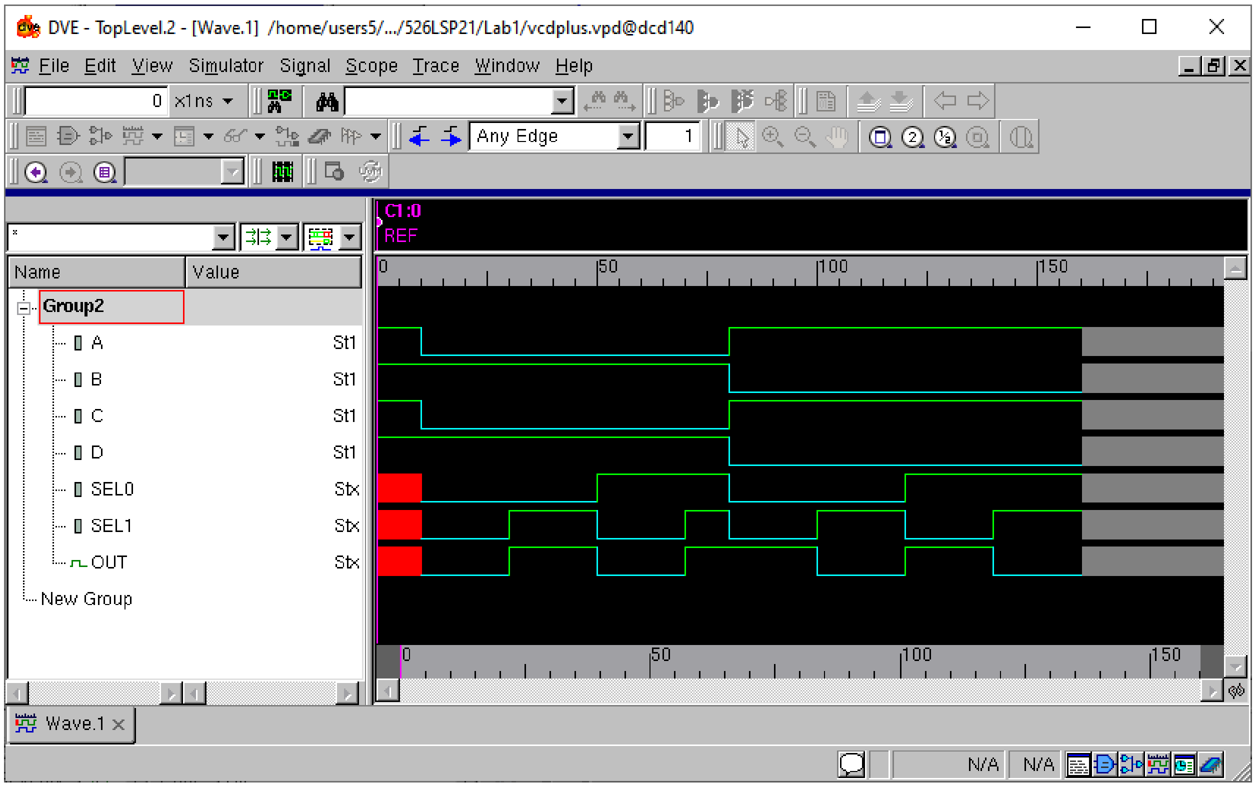
DVE - TopLevel.1 - [Source.1 - mux4_1_tb: mux4_1_tb.v] /home/users5/.../526LSP21/Lab1/vcdplus.vpd@dcd140

```
`timescale 1 ns / 1 ns
module mux4_1_tb();
        reg A,B,C,D, SEL0, SEL1;
        wire OUT;

        mux4_1 UUT(OUT, A, B, C, D, SEL0, SEL1);

        initial
        $monitorb ("%d OUT= %b A= %b B= %b C= %b D= %b SEL0= %b SEL= %b",                    $time, OUT, A, B, C, D, SEL0, SEL1);
        initial begin
        $vcdpluson;
        A=1; B=1; C=1; D=1; SEL0=1'bx; SEL1=1'bx;
        #10 A=0; B=1; C=0; D=1; SEL0=0; SEL1=0;
        #20 SEL0=0; SEL1=1;
        #20 SEL0=1; SEL1=0;
        #20 SEL0=1; SEL1=1;
        #10 A=1; B=0; C=1; D=0; SEL0=0; SEL1=0;
        #20 SEL0=0; SEL1=1;
        #20 SEL0=1; SEL1=0;
        #20 SEL0=1; SEL1=1;
        #20 $finish;
        end
endmodule
```

/home/users5/ma581955/526LSP21/Lab1/mux4_1_tb.v

```
The file '/home/users5/ma581955/526LSP21/Lab1/vcdplus.vpd' was opened successfully.
```

Select all the signals with the mouse, then right click and select “Add to Wave -> New Wave View.” You should see the below window:


DVE - TopLevel.2 - [Wave.1] /home/users5/.../526LSP21/Lab1/vcdplus.vpd@dcd140
File Edit View Simulator Signal Scope Trace Window Help
0 x1ns
Any Edge
1
C1:0
REF
Name
Value
Group2
A St1
B St1
C St1
D St1
SEL0 StX
SEL1 StX
OUT StX
New Group
0
50
100
150
Wave.1
N/A
N/A

Returning to the Hierarchy window on the DVE screen, enable scoping down into the design by clicking on the + sign next to the test fixture's name. Click on the lower-level design and its signals too will appear in the Variable window. These internal signals can then be added to the waveform.

You can navigate around the waveform, add and remove cursors, zoom in and out, etc. via the mouse and by selecting menu items from the top of the waveform window. You can rearrange the order of signals displayed by dragging and dropping them in the leftmost waveform window. Changing the radix displayed, which can be done from the leftmost window in DVE, will prove useful in later labs but in this lab, all signals are single bit, so simple binary is the only option.

To exit from DVE, select File -> Exit from the DVE window. Click OK to confirm the exit.

For your lab report submit the verilog codes, log file, screenshot of your simv output that shows your username on top of the window and screenshot of DVE waveform. These should be supporting files of your lab report not the report itself.

Extra Credit

If we wanted to have an exhaustive test that covers every possible input, how many test cases we should write?
If just for this lab we consider X and Z as possible inputs, then how many test cases we would have?