Running Verilog in ECS server titan.ecs.csus.edu

First you need to have an ecs account

- Logon to Titan from an ecs computer or using a terminal emulator in your computer.

Note: The department recommends **MobaXTerm** or **Putty** which are built for Windows Machines. If you are Mac user, you can use the "**terminal app**" located in the applications folder.

Logon using ssh and enter your password.

> ssh <ECS-Username>@titan.ecs.csus.edu

A prompt will appear. Something like:

[ECS-Username@titan:1]>

- Logon to Titan from off campus:

From off campus use ***titan.ecs.csus.edu***. You can install a VNC Viewer such as Ultra VNC if you wish. If you use a VNC Viewer, use port 50 or 52 (enter machine name as titan:50 from on campus or titan.ecs.csus.edu:50 from off campus.) For secure off campus connection run VPN before you run VNC.

General procedure on how to run Verilog simulator on Titan (legacy instructions):

1. Logon to Titan. From off campus use tital.ecs.csus.edu. You can install a VNC Viewer such as Ultra VNC if you wish. If you use a VNC Viewer, use port 50 or 52 (enter machine name as titan:50 from on campus or titan.ecs.csus.edu:50 from off campus.) For secure off campus connection run VPN before you run VNC.
2. Enter vcs +v2k filename.v at the Unix prompt to compile your verilog code on Titan. The compiled file is saved as simv and to simulate the design enter simv at the Unix/Linux prompt. Use `include to include the lower level .v files in each of the higher level .v files; note the tick (`) before include. Save the simulation output into a file as simv > outfile if you wish.

Create a project directory and create Verilog files.

- From the Unix prompt line create a directory for the class (ecs137) and a project subdirectory called full_adder

```
[ECS-Username@titan:1]> mkdir ecs137          <= creates directory named ecs137
[ECS-Username@titan:2]> cd ecs137             <= change location to new directory
[ECS-Username@titan:3]> mkdir full_adder      <= make subdirectory called full_adder
[ECS-Username@titan:4]> cd full_adder         <= change location to new subdirectory full_adder
[ECS-Username@titan:4]> pwd                   <= confirm your directory using pwd (present working dir)
/gaia/class/student/<username>/ecs137/full_adder
```

Use a text editor and create the following verilog files

Create the Verilog HDL circuit***: full_adder.v***

```
module full_adder
(
  input a,b,ci,
  output s, co
);


wire w1, w2, w3;


xor   x1(w1, a, b);
xor   x2(s, w1, ci);
nand  n1(w2, w1, ci);
nand  n2(w3, a, b);
nand  n3(co, w2, w3);


endmodule
```

Create the testbench ***full_adder_tb.v***

```
`timescale 1ns/100ps
`include "full_adder.v"

module full_adder_tb();

reg a, b, ci;
wire s, co;

full_adder fa1(a, b, ci, s, co);

initial begin
   $display("-----------------------");
   $display("|t(ns)| a b ci | co s | ");
   $display("-----------------------");
   a = 0; b = 0; ci = 0;
   #1 $display("|%4d | %b %b %b  |  %b %b |", $time, a, b, ci, co, s);
   a = 0; b = 0; ci = 1;
   #1 $display("|%4d | %b %b %b  |  %b %b |", $time, a, b, ci, co, s);
   a = 0; b = 1; ci = 0;
   #1 $display("|%4d | %b %b %b  |  %b %b |", $time, a, b, ci, co, s);
   a = 0; b = 1; ci = 1;
   #1 $display("|%4d | %b %b %b  |  %b %b |", $time, a, b, ci, co, s);
   a = 1; b = 0; ci = 0;
   #1 $display("|%4d | %b %b %b  |  %b %b |", $time, a, b, ci, co, s);
   a = 1; b = 0; ci = 1;
   #1 $display("|%4d | %b %b %b  |  %b %b |", $time, a, b, ci, co, s);
   a = 1; b = 1; ci = 0;
   #1 $display("|%4d | %b %b %b  |  %b %b |", $time, a, b, ci, co, s);
   a = 1; b = 1; ci = 1;
   #1 $display("|%4d | %b %b %b  |  %b %b |", $time, a, b, ci, co, s);
  $display("-----------------------");
end
endmodule
```

Compile the testbench with the synopsis Verilog compiler: `vcs +v2k full_adder_tb.v`

```
[<userdesign>@titan:66]> vcs +v2k full_adder_tb.v
                         Chronologic VCS (TM)
           Version I-2014.03-2 -- Fri Mar  8 17:54:02 2019
               Copyright (c) 1991-2014 by Synopsys Inc.
                         ALL RIGHTS RESERVED


This program is proprietary and confidential information of Synopsys Inc.
and may be used and disclosed only as authorized in a license agreement
controlling such use and disclosure.


Parsing design file 'full_adder_tb.v'
Parsing included file 'full_adder.v'.
Back to file 'full_adder_tb.v'.
Top Level Modules:
       full_adder_tb
TimeScale is 1 ns / 100 ps
Starting vcs inline pass...
1 module and 0 UDP read.
recompiling module full_adder_tb
rm -f  _csrc*.so linux_scvhdl_*.so pre_vcsobj_*.so share_vcsobj_*.so
ld -m elf_i386 -shared -o ./7../simv.daidir/7_csrc1.so --whole-archive _vcsobj_1_1.a  --no-whole-archive
ld -m elf_i386 -shared -o .//../simv.daidir//_csrc0.so 5NrI_d.o 5NrIB_d.o SIM_I.o
if [ -x ../simv ]; then chmod -x ../simv; fi
g++  -o ../simv  -m32   -Wl,-rpath-link=./ -Wl,-rpath='$ORIGIN'/simv.daidir/ -Wl,-
rpath='$ORIGIN'/simv.daidir//scsim.db.dir   _csrc1.so _csrc0.so     rmapats_mop.o rmapats.o
rmar.o           /titan/software/synopsys147vcs/linux7lib/libzerosoft_rt_stubs.so
/titan/software/synopsys14/vcs/linux/lib/libvirsim.so /titan/software/synopsys14/vcs/linux/lib/librterrorinf.so
/titan/software/synopsys14/vcs/linux/lib/libsnpsmalloc.so    /titan/software/synopsys14/vcs/linux/lib/libvcsnew.so
/titan/software/synopsys14/vcs/linux/lib/libuclinative.so   -Wl,-whole-archive
/titan/software/synopsys14/vcs/linux/lib/libvcsucli.so -Wl,-no-whole-
archive          /titan/software/synopsys14/vcs/linux/lib/vcs_save_restore_new.o
/titan/software/synopsys14/vcs/linux/lib/ctype-stubs_32.a -ldl  -lc -lm -lpthread -ldl
../simv up to date
CPU time: .087 seconds to compile + .065 seconds to elab + .156 seconds to link
```

Run the *simv* executable that was created during compilation and save the results in an “outfile” as shown below.
ACTION Submit the ”outfile” to canvas. It should show your username in the command promt)

```
[<username>@titan:24]> simv > outfile

Chronologic VCS simulator copyright 1991-2014
Contains Synopsys proprietary information.
Compiler version I-2014.03-2; Runtime version I-2014.03-2;  Mar  8 13:08 2019
----------------------
|t(ns)| a b ci | co s |
----------------------
|   1 | 0 0 0  |  0 0 |
|   2 | 0 0 1  |  0 1 |
|   3 | 0 1 0  |  0 1 |
|   4 | 0 1 1  |  1 0 |
|   5 | 1 0 0  |  0 1 |
|   6 | 1 0 1  |  1 0 |
|   7 | 1 1 0  |  1 0 |
|   8 | 1 1 1  |  1 1 |
----------------------
           V C S   S i m u l a t i o n   R e p o r t
Time: 8000 ps
CPU Time:      0.200 seconds;       Data structure size:   0.0Mb
Fri Mar  8 13:08:58 2019
```

Part 2 Create 2 new files: *full_adder_wires_tb.v* and *full_adder_wires.v*

```
module full_adder_wires(
  input a,b,ci,
  output s, co, w1, w2, w3);

// wire w1, w2, w3;
//
// The "wire" declaration above was commented out because
// these wires were moved to the "output" portlist. This
// technique is useful to make w1, w2, w3 visible for
// debuggin at the testbench level
//

xor   x1(w1, a, b);
xor   x2(s, w1, ci);
nand  n1(w2, w1, ci);
nand  n2(w3, a, b);
nand  n3(co, w2, w3);

endmodule
```

```
`timescale 1ns/100ps
`include "full_adder_wires.v"


module full_adder_wires_tb();


reg a, b, ci;
wire s, co, w1, w2, w3;


full_adder_wires fa1(a, b, ci, s, co, w1, w2, w3);


initial begin


 $display("-------------------------------");
 $display("t(ns)| a b ci | co s | w1 w2 w3");
 $display("-------------------------------");
 a = 0; b = 0; ci = 0; #1 $display("%4d | %b %b %b  |  %b %b |  %b  %b  %b", $time, a, b, ci, co, s, w1, w2, w3);
 a = 0; b = 0; ci = 1; #1 $display("%4d | %b %b %b  |  %b %b |  %b  %b  %b", $time, a, b, ci, co, s, w1, w2, w3);
 a = 0; b = 1; ci = 0; #1 $display("%4d | %b %b %b  |  %b %b |  %b  %b  %b", $time, a, b, ci, co, s, w1, w2, w3);
 a = 0; b = 1; ci = 1; #1 $display("%4d | %b %b %b  |  %b %b |  %b  %b  %b", $time, a, b, ci, co, s, w1, w2, w3);
 a = 1; b = 0; ci = 0; #1 $display("%4d | %b %b %b  |  %b %b |  %b  %b  %b", $time, a, b, ci, co, s, w1, w2, w3);
 a = 1; b = 0; ci = 1; #1 $display("%4d | %b %b %b  |  %b %b |  %b  %b  %b", $time, a, b, ci, co, s, w1, w2, w3);
 a = 1; b = 1; ci = 0; #1 $display("%4d | %b %b %b  |  %b %b |  %b  %b  %b", $time, a, b, ci, co, s, w1, w2, w3);
 a = 1; b = 1; ci = 1; #1 $display("%4d | %b %b %b  |  %b %b |  %b  %b  %b", $time, a, b, ci, co, s, w1, w2, w3);
 $display("-------------------------------");
 $display("---------- Above we used $display statements for each input combination ----------");
 $display();


 $display("--- Below we used a single $monitor statement to report every time the inputs change---");
 $display("-----------------------------------------------------");
 $display("T(ns)| A B Ci | Co S | W1 W2 W3 inputs go 7 -> 0 now");
 $display("-----------------------------------------------------");
 $monitor("%4d | %b %b %b  |  %b %b |  %b  %b  %b", $time, a, b, ci, co, s, w1, w2, w3);
 #1 a = 1; b = 1; ci = 1;
 #1 a = 1; b = 1; ci = 0;
 #1 a = 1; b = 0; ci = 1;
 #1 a = 1; b = 0; ci = 0;
 #1 a = 0; b = 1; ci = 1;
 #1 a = 0; b = 1; ci = 0;
 #1 a = 0; b = 0; ci = 1;
 #1 a = 0; b = 0; ci = 0;
 $display("-----------------------------------------------------");

end


endmodule
```

Compile the testbench: vcs +v2k full_adder_wires_tb.v and run your **simv** executable

ACTION: submit your "simv" executable file and briefly explain how for this case we can see the internal wires w1, w2, w3 of the full adder.

```
[<username>@titan:26]> simv
Chronologic VCS simulator copyright 1991-2014
Contains Synopsys proprietary information.
Compiler version I-2014.03-2; Runtime version I-2014.03-2;  Mar  8 13:12 2019
------------------------------
t(ns)| a b ci | co s | w1 w2 w3
------------------------------
   1 | 0 0 0  |  0 0 |  0  1  1
   2 | 0 0 1  |  0 1 |  0  1  1
   3 | 0 1 0  |  0 1 |  1  1  1
   4 | 0 1 1  |  1 0 |  1  0  1
   5 | 1 0 0  |  0 1 |  1  1  1
   6 | 1 0 1  |  1 0 |  1  0  1
   7 | 1 1 0  |  1 0 |  0  1  0
   8 | 1 1 1  |  1 1 |  0  1  0
------------------------------
---------- Above we used $display statements for each input combination -----------

--- Below we used a single $monitor statement to report every time the inputs change---
-----------------------------------------------
T(ns)| A B Ci | Co S | W1 W2 W3 inputs go 7 -> 0 now
-----------------------------------------------
   8 | 1 1 1  |  1 1 |  0  1  0
  10 | 1 1 0  |  1 0 |  0  1  0
  11 | 1 0 1  |  1 0 |  1  0  1
  12 | 1 0 0  |  0 1 |  1  1  1
  13 | 0 1 1  |  1 0 |  1  0  1
  14 | 0 1 0  |  0 1 |  1  1  1
  15 | 0 0 1  |  0 1 |  0  1  1
-----------------------------------------------
  16 | 0 0 0  |  0 0 |  0  1  1
           V C S   S i m u l a t i o n   R e p o r t
Time: 16000 ps
CPU Time:      0.180 seconds;       Data structure size:   0.0Mb
Fri Mar  8 13:12:41 2019
```