



HOMEWORK – USING ASSERTS -- Prof. Don Bouldin

Purpose: Perform pre-synthesis simulation with *ModelSim* using Asserts.

Part B

Download: <http://www.ece.utk.edu/~bouldin/protected/551-hw6b.tar.gz>

Or copy the files on `ada3.eecs.utk.edu`:

1. **`cp ~bouldin/webhome/protected/551-hw6b.tar.gz .`**
2. **`gunzip 551-hw6b.tar.gz`**
3. **`tar -xvf 551-hw6b.tar`**

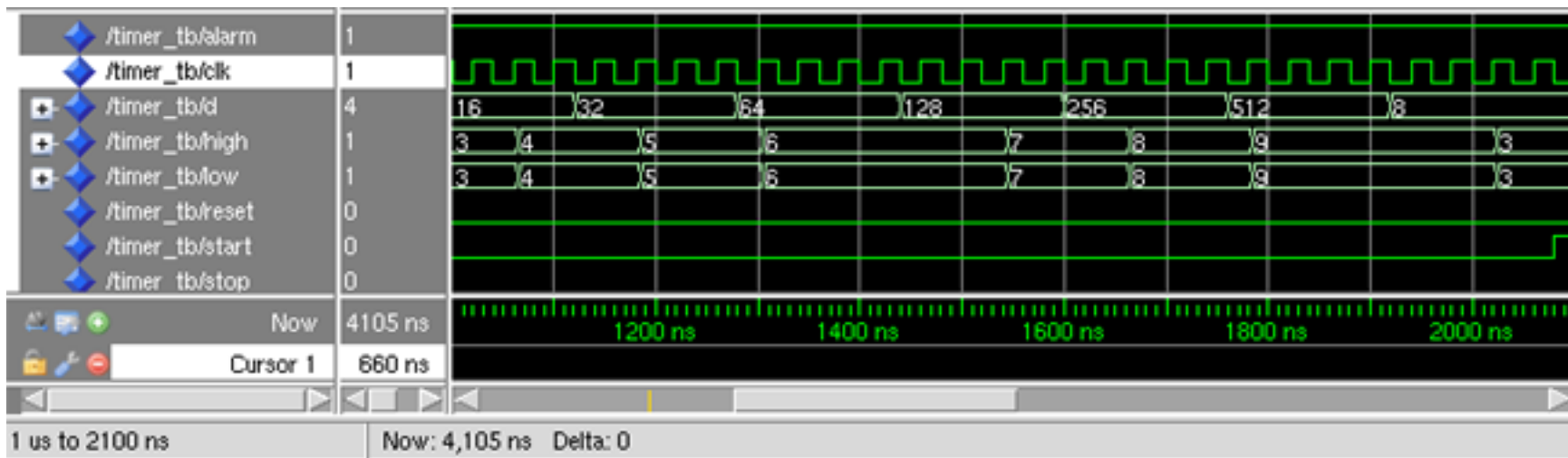
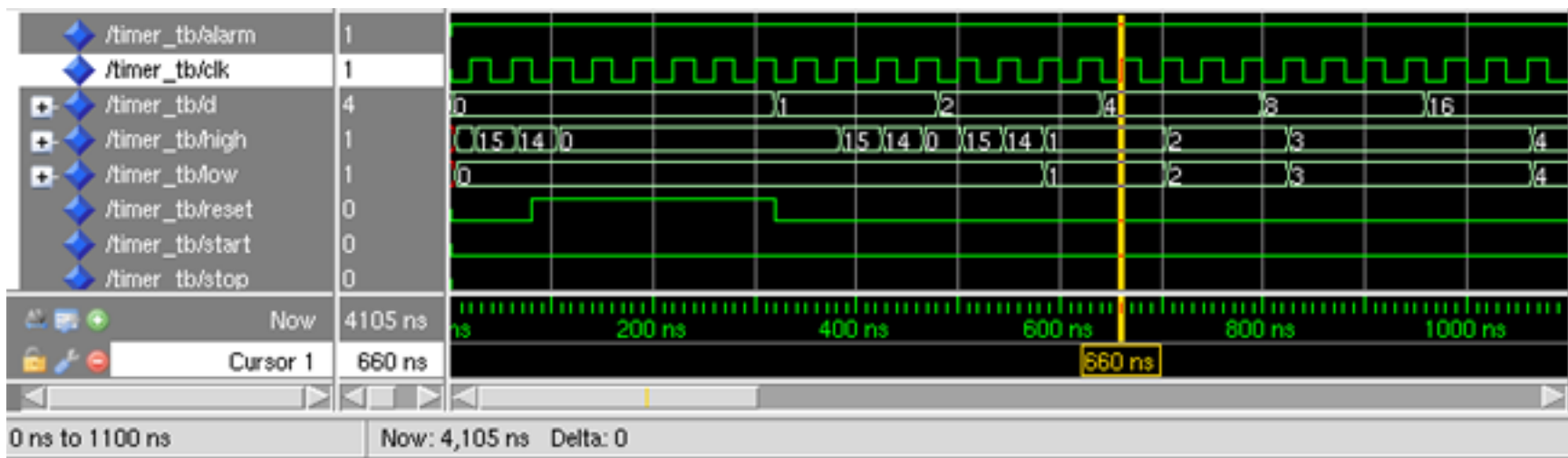
Now, move down to the subdirectory:

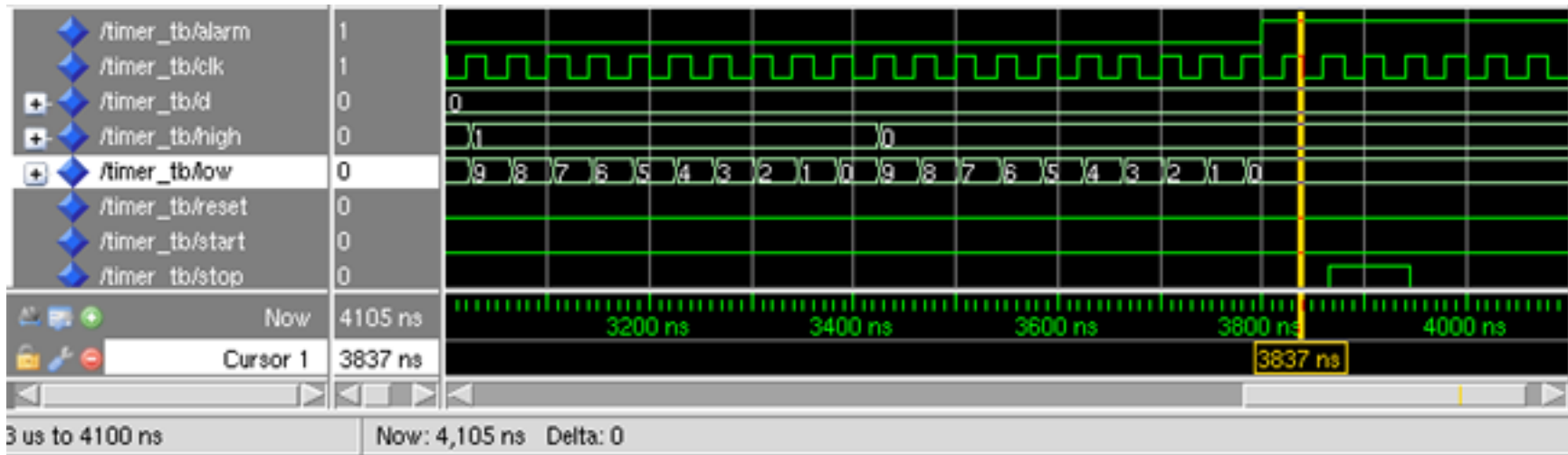
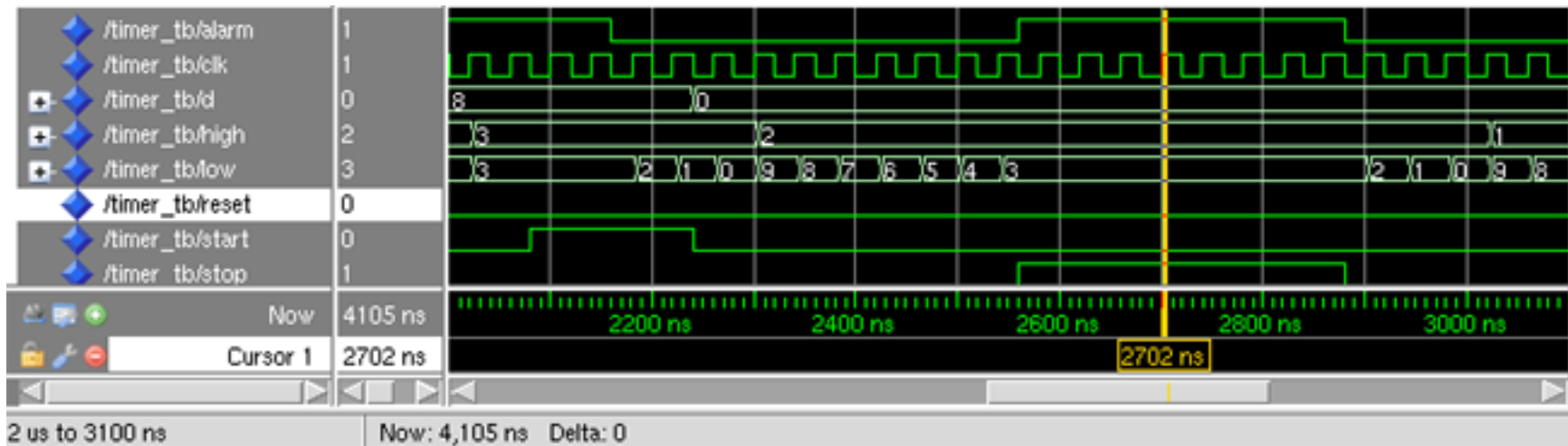
4. **`cd 551-hw6b`**

Now, perform pre-synthesis simulation by typing:

5. **`./presynth_sim`**

This will bring up the following waveform window (with the radix of “d”, “high” and “low” manually changed to “unsigned”):





Highlight “control(fsm)” to see the coverage:

Instance	Design unit	Stmt %	Stmt graph	Branch %	Branch graph
standard	standard				
std_logic_1164	std_logic_1164				
std_logic_arith	std_logic_arith				
timer_tb	timer_tb(struct)				
i0	timer(struct)	100%		100%	
i1	counter(struct)	100%			
i0	bcdcounter(spec)	100%		100%	
i1	bcdcounter(spec)	88.9%		87.5%	
line__76	counter(struct)				
line__79	counter(struct)				
line__112	counter(struct)				
i0	control(fsm)	87.5%		85.4%	

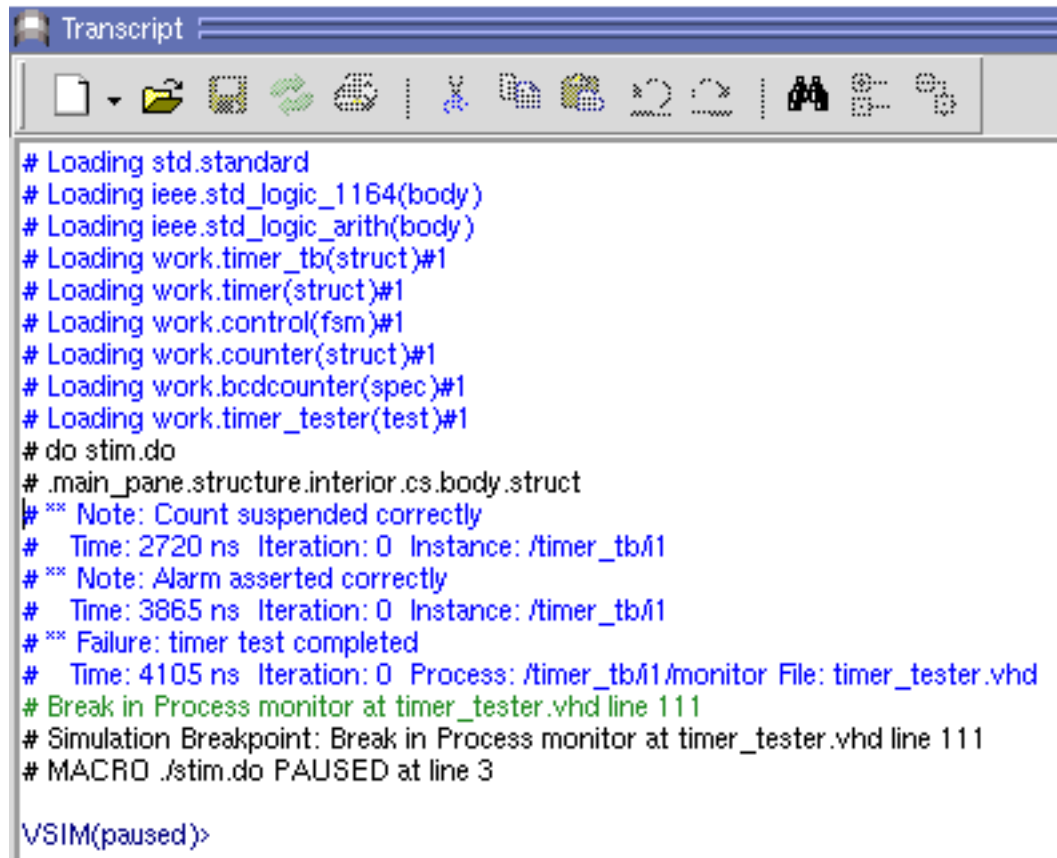
Highlight “control(fsm)” to see the statements missed:

```
▼ Missed Statements
[-] [H] control.vhd
  -X   114      next_state <= flush;
  -X   128      next_state <= load_t;
  -X   134      next_state <= alarm;
  -X   138      next_state <= standby;
  -X   164      next_state <= end_count;
  -X   167      next_state <= flush;
```

Highlight “bcdcounter.” to see the statements missed:

```
▼ Missed Statements
[-] [H] bcdcounter.vhd
  -X   45 ) THEN int_count := "1001" ;
```

The transcript window shows which “asserts” were printed:

A screenshot of a VSIM transcript window. The window title is "Transcript". It features a toolbar with icons for file operations (new, open, save, print), editing (undo, redo, copy, paste), and simulation control (run, stop, pause, refresh). The transcript text shows the loading of various standard and user-defined modules, followed by a stimulus file execution. Key simulation events include a note about suspended counts, an alarm assertion, a failure in the timer test, and a break in the process monitor at line 111 of timer_tester.vhd. The simulation is currently paused.

```
# Loading std.standard
# Loading ieee.std_logic_1164(body)
# Loading ieee.std_logic_arith(body)
# Loading work.timer_tb(struct)#1
# Loading work.timer(struct)#1
# Loading work.control(fsm)#1
# Loading work.counter(struct)#1
# Loading work.bcdcounter(spec)#1
# Loading work.timer_tester(test)#1
# do stim.do
# .main_pane.structure.interior.cs.body.struct
#** Note: Count suspended correctly
# Time: 2720 ns Iteration: 0 Instance: /timer_tb/1
#** Note: Alarm asserted correctly
# Time: 3865 ns Iteration: 0 Instance: /timer_tb/1
#** Failure: timer test completed
# Time: 4105 ns Iteration: 0 Process: /timer_tb/1/monitor File: timer_tester.vhd
# Break in Process monitor at timer_tester.vhd line 111
# Simulation Breakpoint: Break in Process monitor at timer_tester.vhd line 111
# MACRO ./stim.do PAUSED at line 3

VSIM(paused)>
```

Modify the testbench and the assert statements to achieve near 100% coverage with no missed statements and capture the new figures.

presynth_sim

```
#!/bin/csh -f
vlib work
vcom -cover bcescf -work work bcdcounter.vhd
vcom -cover bcescf -work work counter.vhd
vcom -cover bcescf -work work control.vhd
vcom -cover bcescf -work work timer.vhd
vcom -cover bcescf -work work timer_tester.vhd
vcom -cover bcescf -work work timer_tb.vhd
vsim -coverage -do stim.do timer_tb
```

stim.do

```
view structure
add wave *
run 5000
```