**ECEN 340**

**Lab #4**

Verilog Simulation, 7-Segment Display

**Purpose:**

1. To learn how to interface with a 7-segment display

2. To learn how to simulate by “forcing” signals

3. To learn how to simulate with a test bench

Procedure:

Part 1a. Refer to the [Basys 3 FPGA Board Reference Manual](https://content.byui.edu/file/f5da977b-df50-45e8-9961-123e53b81103/1/TechnicalResources/Basys%203%20Reference%20Manual.pdf), pp. 15-16 to learn how to interface to a single 7-segment digit. You will need to combine the information found in the above reference manual with the constraint file to determine the true order of the segments “abcd…” and how they relate to the 7-bit “seg” bus. Remember that with the declaration

output reg [6:0] seg,

seg[6] is the MSB,

seg[0] is the LSB.

If the constraint file assigns seg[0] to pin W7 of the FPGA, how do you combine the images from the reference manual (replicated below) with the constraint file to determine the correct order of the LEDs (“abcdefg” VS “gfedcba”)?

A diagram of a circuit

Description automatically generated A diagram of an electronic device

Description automatically generated

Modified from Figures 16 and 18 of the Reference Manual

Part 1b. Write a Verilog module to interface to one digit of the 7-seg display. You will use four of the switches to control the displayed value. You will use the “if, else if, else” structure or the “case” structure to interface to display. From the constraints file, you will be using “an”, “dp”, “seg”, and “sw”. Make sure you have matching ports in your module port list! Double check the size of each port!

Example of “if statement” method:

assign an = 4’b1110; //select digit on right with low assertion

assign dp = 1’b1; //force the decimal point off

always @(sw) // sw is in the sensitivity list

begin

if (sw == 4’h0) seg = 7’b1000000;

else if (sw == …

…

else seg = … // This should be the default state of seg

end //always

Example of “case statement” method (similar to Fig. 4.34 in textbook):

assign an = 4’b1110; //select digit on right with low assertion

assign dp = 1’b1; //force the decimal point off

always @(sw) // sw is in the sensitivity list

case (sw)

4’h0: seg = 7’b1000000; // This could also be 7’h40

4’h1: seg = …

…

default: …// This should be the default state of seg

endcase

Part 1c. Follow the following simulation steps:

* Synthesize the design and fix all errors.
* Run simulation (behavioral). Observe the value of sw[3:0] (Figure 1).
* Right click on “value” and select “Force Constant” to for sw[3:0] to 0 (Figure 2).
* Above the waveform viewer, you will see two blue run arrows. To run the simulation for 1us, enter 1us in the box to the right of the blue arrows, select the blue arrow with the “t”, and observe the simulated contents of sseg (Figure 3).
* Change the forced value of “sw” to simulate each of the possibilities for to verify functionality. After each simulation, you will need to “right click” on the simulation output window to resize the simulation viewing area.
* Generate Bitstream and program device to verify the actual functionality of your design.

A screenshot of a computer

Description automatically generated

Figure 1 – Simulation results with no test bench. Note that sw[3:0] is “Z”, resulting in “XX” for sseg[7:0] (note that you may only be using sseg[6:0]).

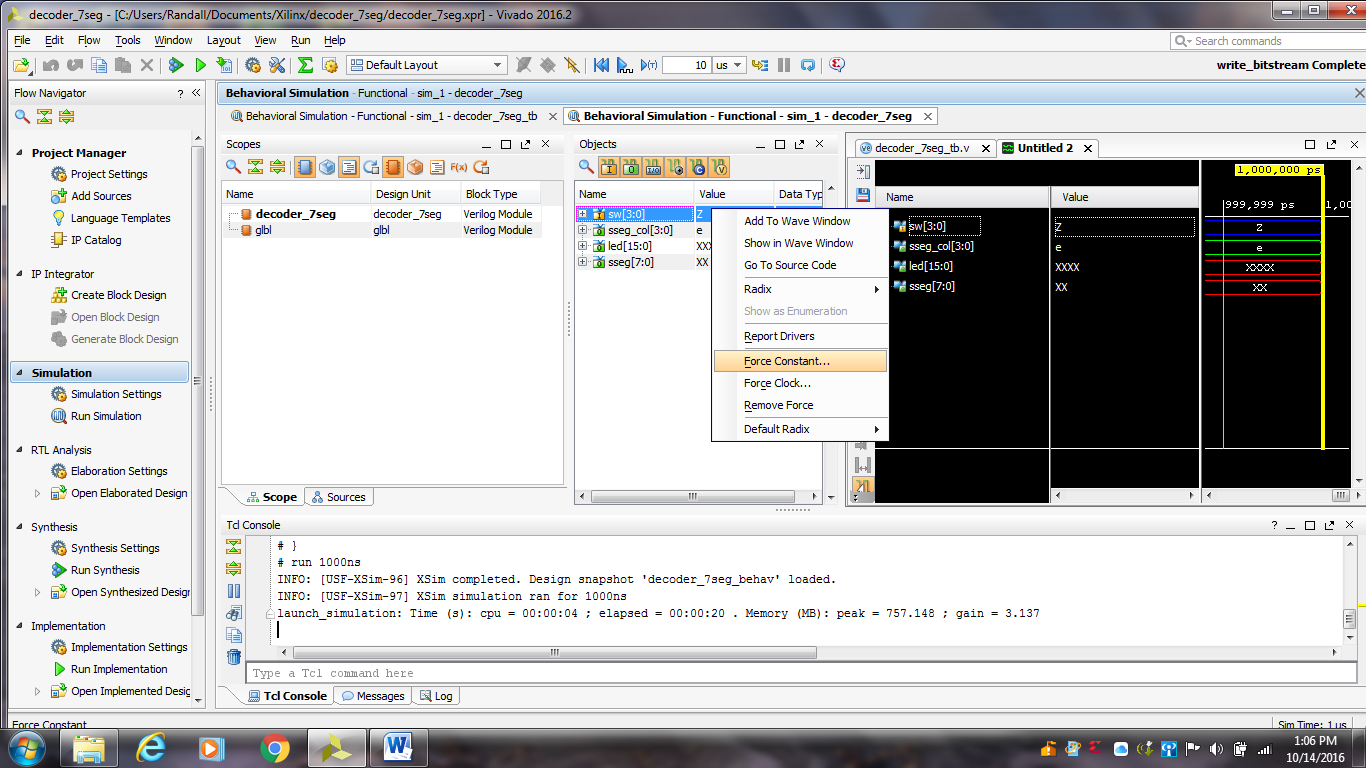


Figure 2 – Forcing sw[3:0] to zero

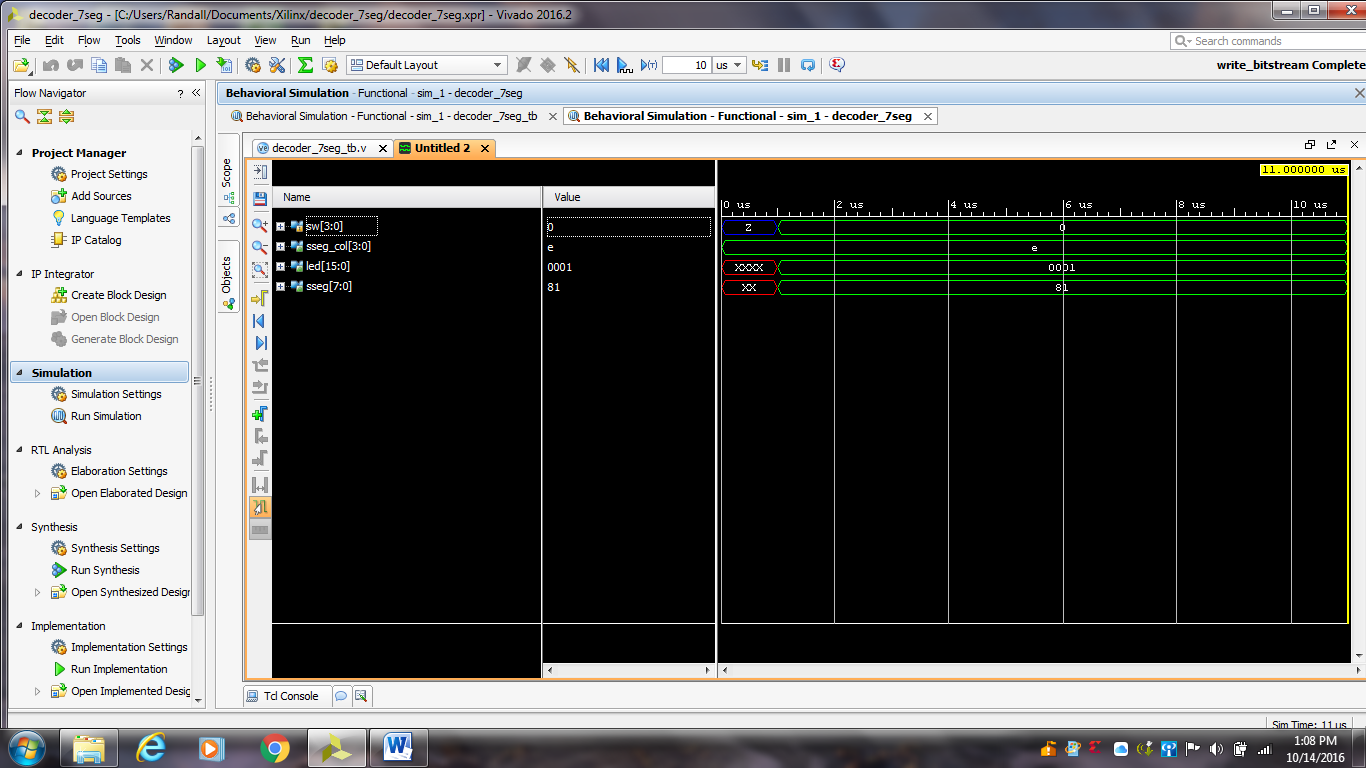


Figure 3 – Simulation result after forcing sw[3:0] to zero

Part 2. Write a test bench to simulate all of the possible switch settings. Here are some important tips about test benches:

1) The test bench module is stored in a separate Verilog file. It is not combined in the same file as the modules used to implement the design. Remember, the test bench is not synthesized, so it is not part of the design.

2) In Vivado, there are provisions to add the test bench Verilog to the project. To do this go to the “Add Sources” menu under the project manager and add the test bench as a “simulation source” rather than a “design source” (Figure 4).   
  
Select the green “+” to add the new file. If you already have a test bench file, you will select “Add Files”. Otherwise, you will select “Create File” (Figure 5). You will enter the test bench name and press “Finish”.   
  
The name of the test bench should be based on the name of the module under test. For example: “decoder\_7seg\_tb” is a good name for the “decoder\_7seg” test bench.

3) A test bench has no ports. If you use the GUI to create the test bench, ignore the GUI prompt for port names and select “OK”.

4) Within the test bench, the UUT (Unit Under Test) module will be instantiated. In the circuit design world, the word “instantiate” means that “an instance” of the module is placed within another module. In more familiar programming terms, it is like calling a procedure from another procedure.

5) Make sure the new simulation module is selected as the top-level for simulation purposes. This is done by selecting “Simulation Settings” from the simulation menu (Figure 6).

6) Start out simple with the test bench (Figure 7), and make sure it simulates error free (Figure 8). Then add the necessary complexity.

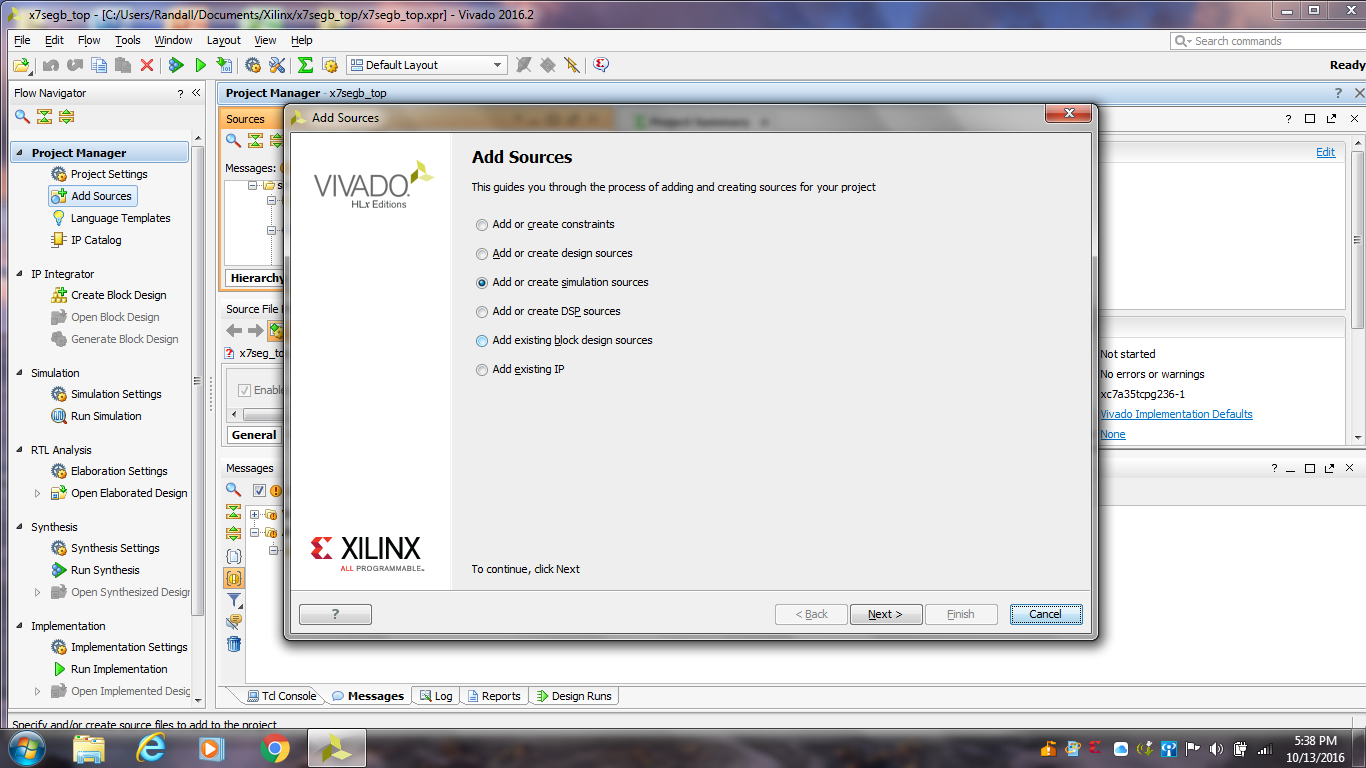


Figure 4 – Adding a test bench file to the project

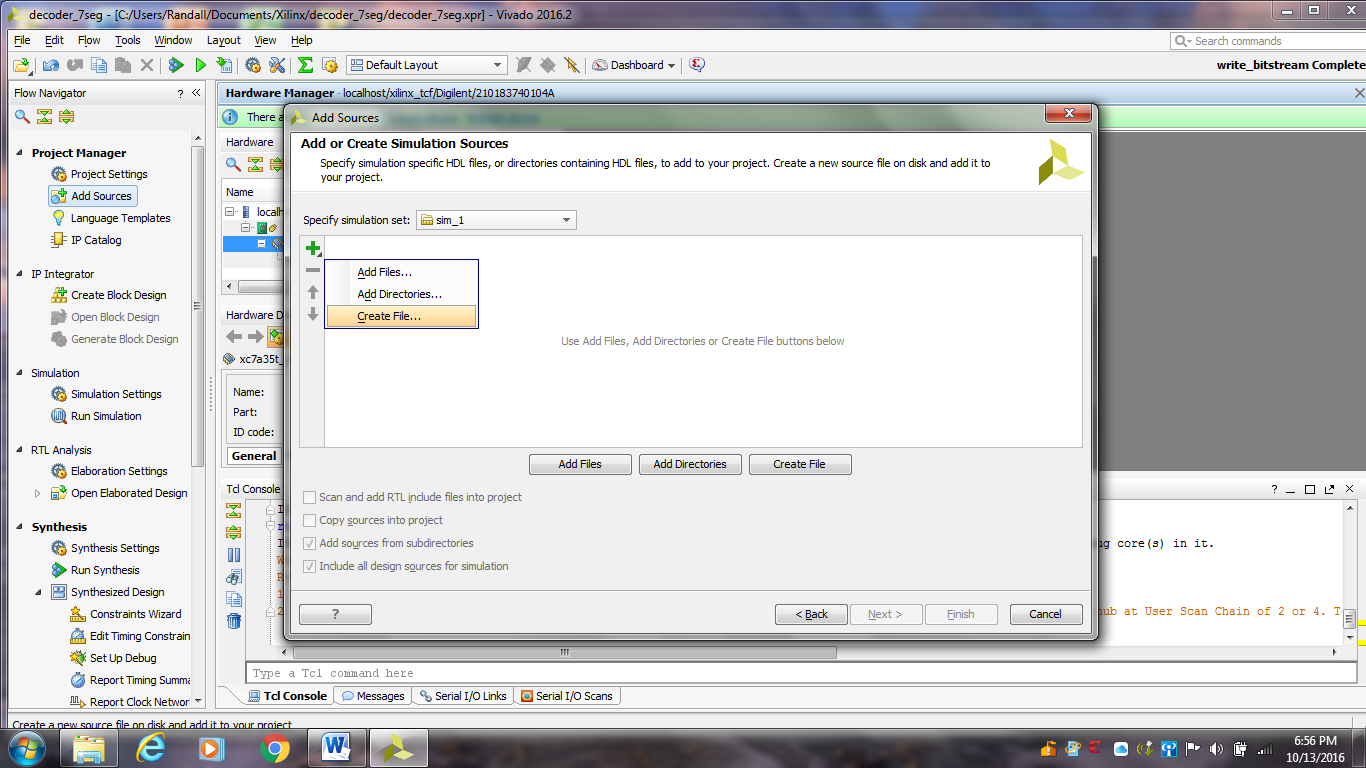


Figure 5 – Selecting an existing file or creating a new file

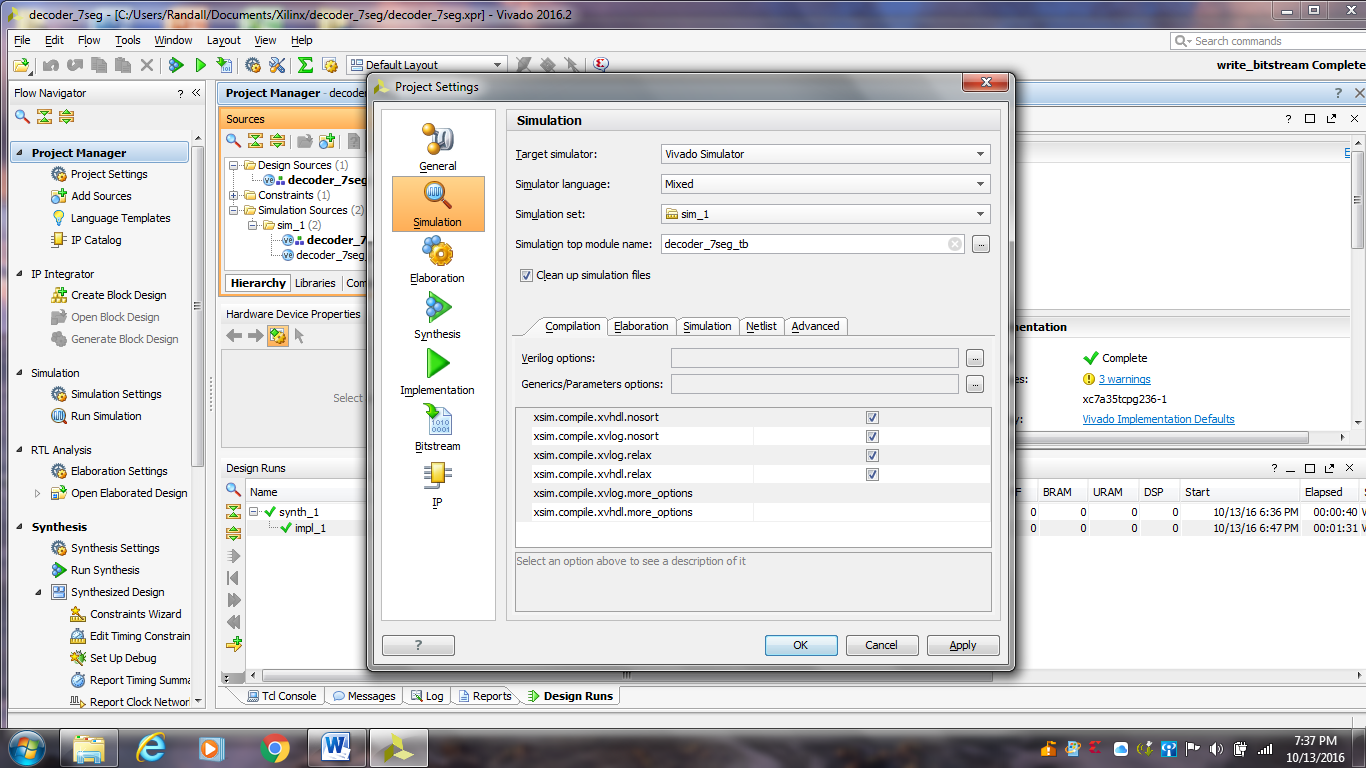


Figure 6 – Selecting the simulation top module name

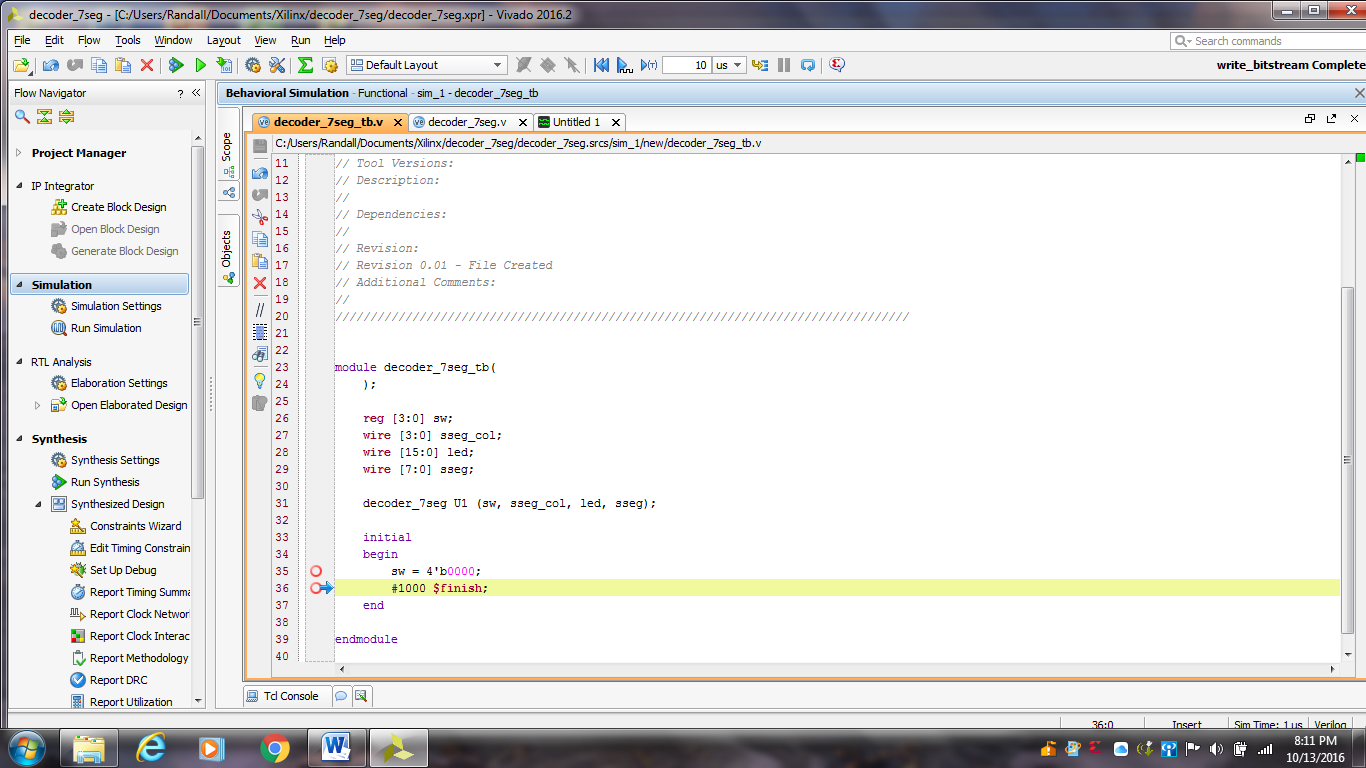


Figure 7 – A very basic starting point for the test bench

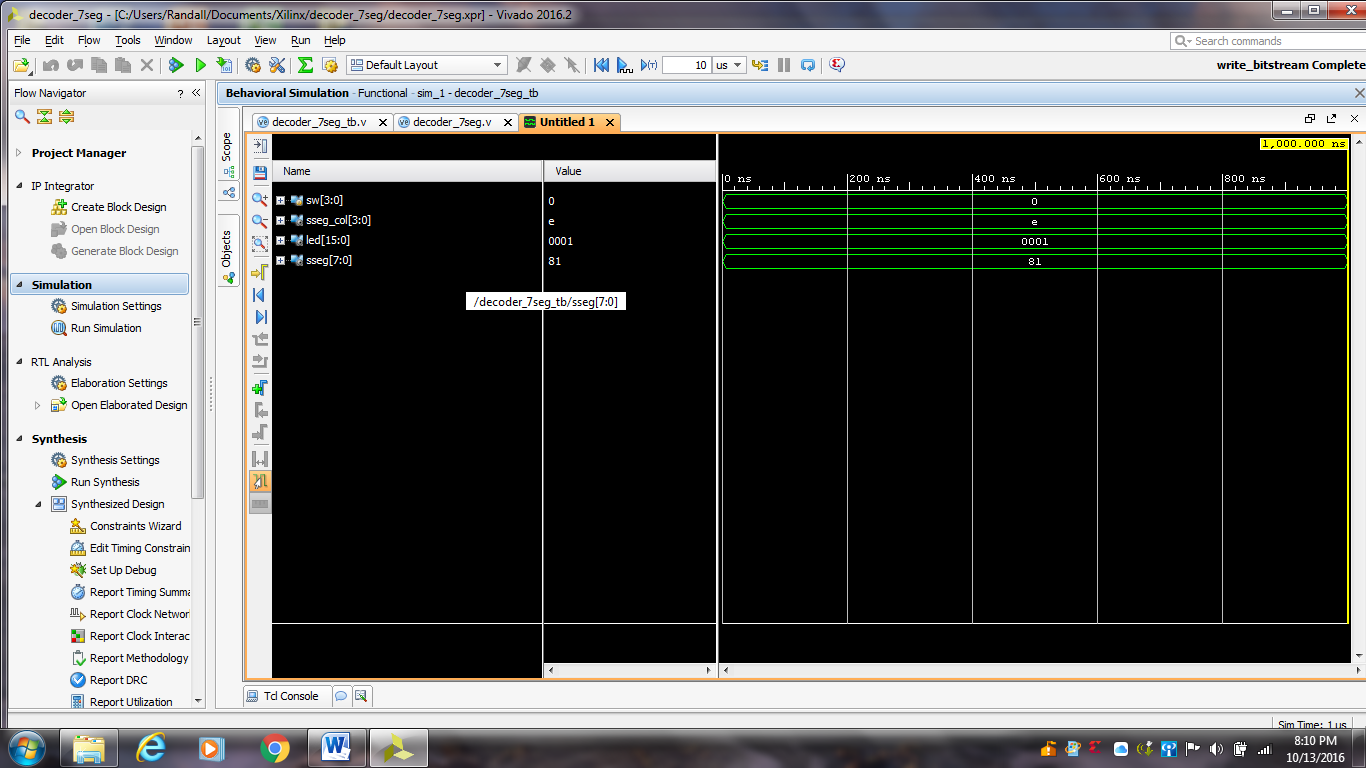


Figure 8 – Simulation results of basic starting point for the test bench

**Submitting your report:**

You will submit a combined report for both parts of this lab which should include the following:

1) A copy of the Verilog code used to implement the design(s). The Verilog code should be complete with useful comments (5 pts).

2) A brief description of the unique (new) tools, technologies, or methods used to implement this lab (5 pts).

3) The report should be professional quality—meaning it will be neat and use proper English. You may include Vivado screen shots to make it easier to document your work (5 pts).

4) In your conclusion statement, you will discuss your results, the method of testing, and include the level of functionality of the lab (10 pts).