CS M152A - Lab 1 Report

Team 2:
Edgar Ayala (UID: 604619231)
Nathaniel Thomas (UID: 506164479)

An Easier Way to Load Sequencer Program

1. **Identify the part of the tb.v where the instructions are sent to the UUT.**

The instructions are sent to the UUT in the following code segment (lines 32-40):

```
tskRunPUSH(0,4);
tskRunPUSH(0,0);
tskRunPUSH(1,3);
tskRunMULT(0,1,2);
tskRunADD(2,0,3);
tskRunSEND(0);
tskRunSEND(1);
tskRunSEND(2);
tskRunSEND(3);
```

2. **Which user tasks are called in this process?**

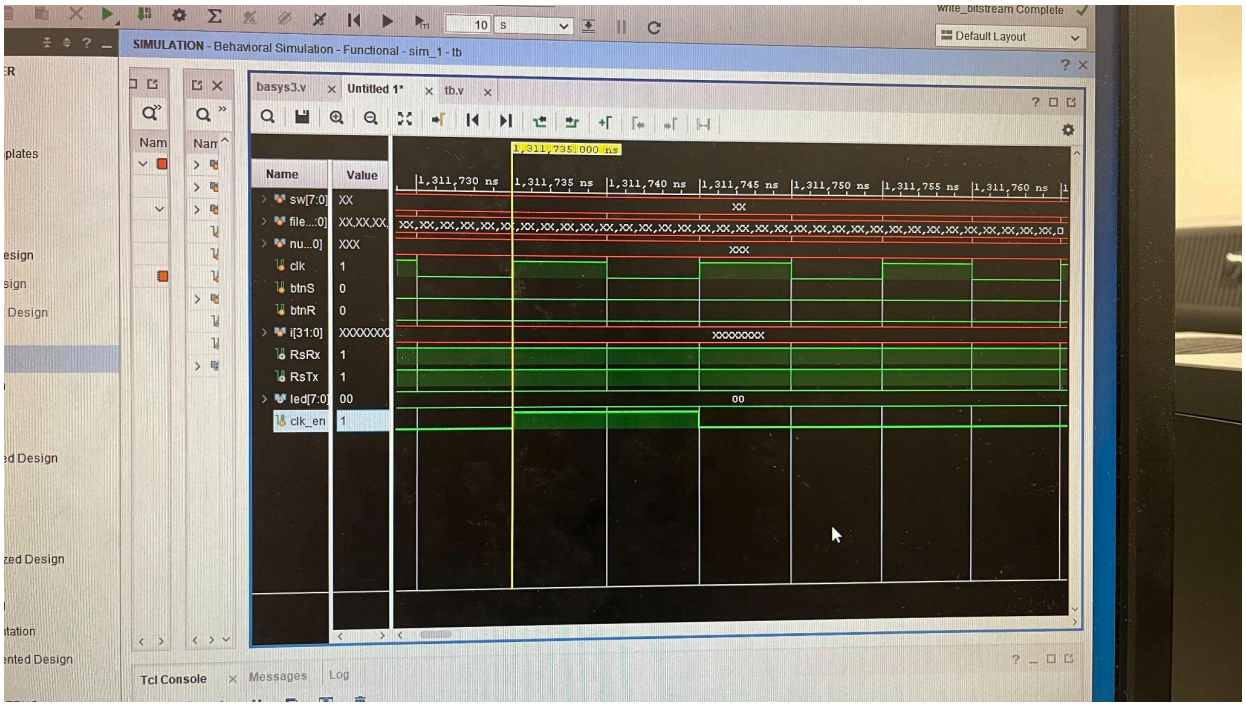
The following user tasks are called:
tskRunPUSH
tskRunMULT
tskRunADD
tskRunSEND

Clock Dividers

In the basys3.v file, there is a signal/reg named clk_en. The clk_en represents a clock that is much slower than the master clock clk. Read the section of code that's relevant to clk_en and try to understand how the clock divider is implemented.

1. **Add clk_en to the simulation's waveform tab and then run the simulation again. Use the cursor to find the periodicity of this signal (you can select the signal and use arrow keys to reach the exact edges). Capture a waveform picture that shows two occurrences of clk_en, and include it in the lab report. Indicate the exact period of the signal in the report.**

The first positive edge occurs at 1311735 ns and the second occurs at 2622455 ns so the period is the difference between these two which is 1310720 ns (2^17).



2. **A duty cycle is the percentage of one period in which a signal or system is active: $D = \frac{T}{P} \times 100\%$, where D is the duty cycle, T is the interval where the signal is high, and p is the period. What is the exact duty cycle of clk_en signal?**

T = 10 ns

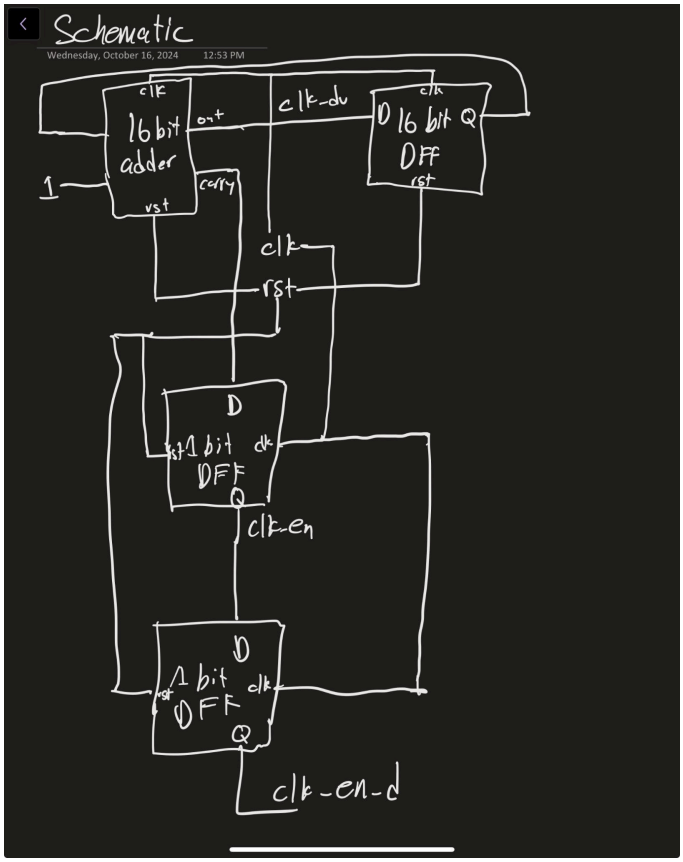
P = 1310720 ns

D = T/P * 100% = 10/1310720 * 100% = 0.00000762939 * 100% = 0.000762939%

3. **What is the value of clk_dv signal during the clock cycle that clk_en is high?**

clk_dv = 0000000000000000

4. **Draw a simple schematic/diagram of signals clk_dv, clk_en, and clk_en_d signals. It should be a translation of the corresponding Verilog code.**



Debouncing

Now move on to the signal inst_vld. Read the relevant code and use the simulation as your aid, answer the following questions in your lab report.

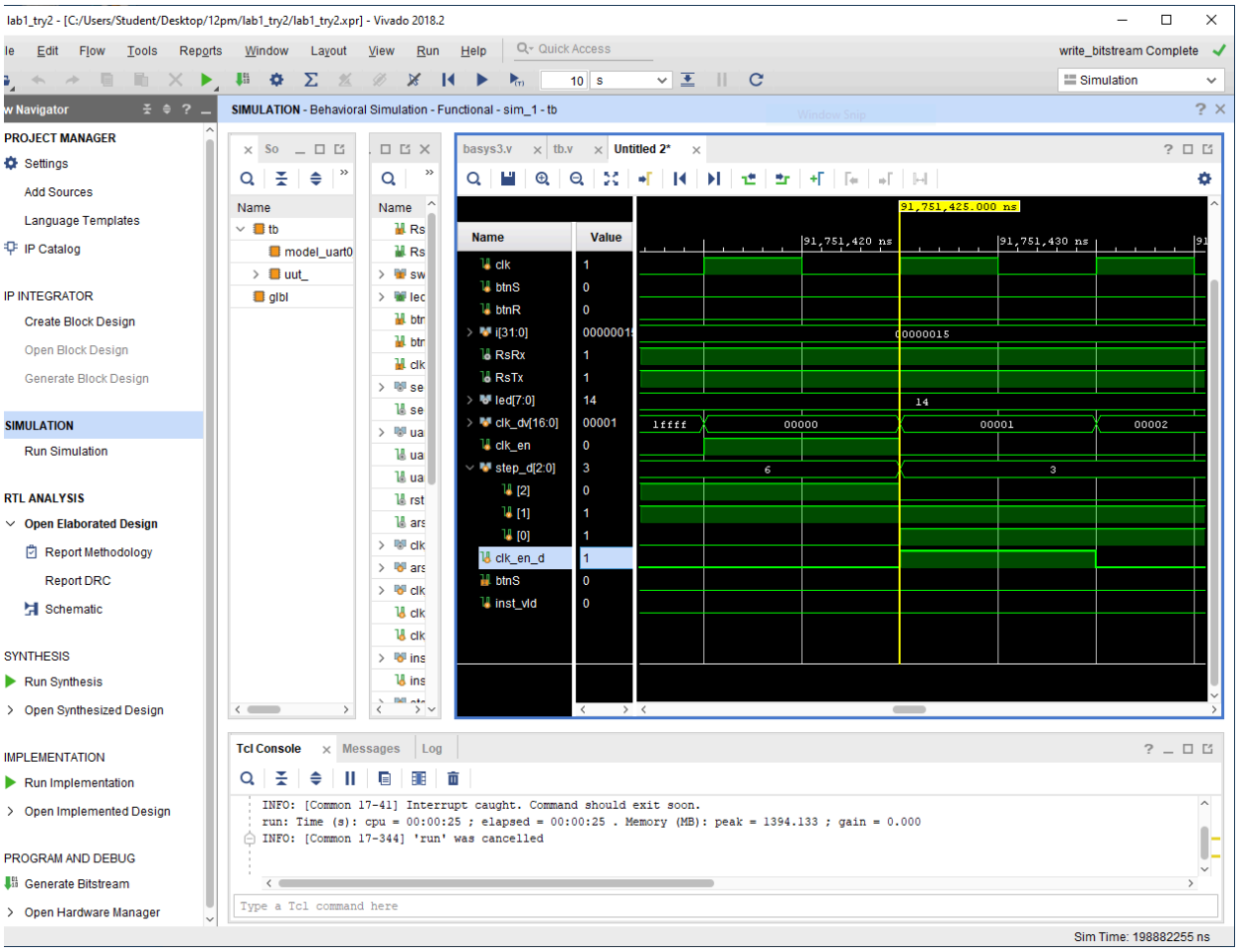
1. **What is the purpose of clk_en_d signal when used in expression ~step_d[0] & step_d[1] & clk_en_d? Why don't we use clk_en?**

We intentionally delay the clock cycle to allow the previous value of clk_en_d to propagate correctly before using it in the expression. If we were to use clk_en, then the value may be incorrect if the clock hasn't finished its cycle.

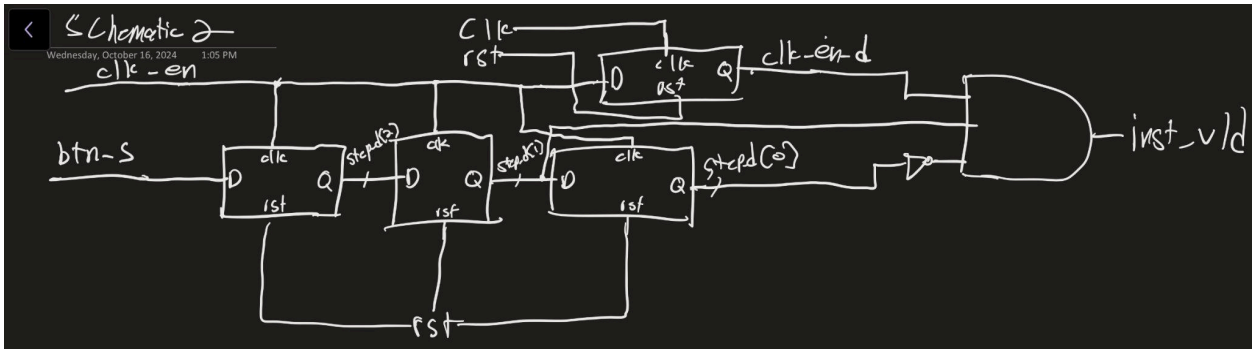
2. **Instead of clk_en <= clk_dv_inc[17], can we do clk_en <= clk_dv[16], making the duty cycle of clk_en 50%? Why?**

It is correct that it would make the clock cycle 50% but that would break the rest of the code that depends on the clock cycle being high once every 2^17 clock cycles and only high for one cycle.

3. **Include waveform captures that clearly show the timing relationship between clk_en, step_d[1], step_d[0], btnS, clk_en_d, and inst_vld.**



4. **Draw a simple schematic/diagram of the signals above. It should be a translation of the corresponding Verilog code.**



Register File

The sequencer's register file is located in a file called seq_rf.v. It stores the values of the four registers. Take a look at the source code and see if you can understand how it is implemented. Answer the following questions in the lab report.

1. **Find the line of code where a register is written a non-zero value. Is this sequential logic or combinatorial logic?**

The register is written a non-zero value in the line: rf[i_wsel] <= i_wdata;

This uses sequential logic as it is located within the always block, as nonblocking assignments (<=) cannot be combined with blocking assignments (=) within the same block.

2. **Find the lines of code where the register values are read out from the register file. Is this sequential or combinatorial logic? If you were to manually implement the readout logic, what kind of logic elements would you use?**

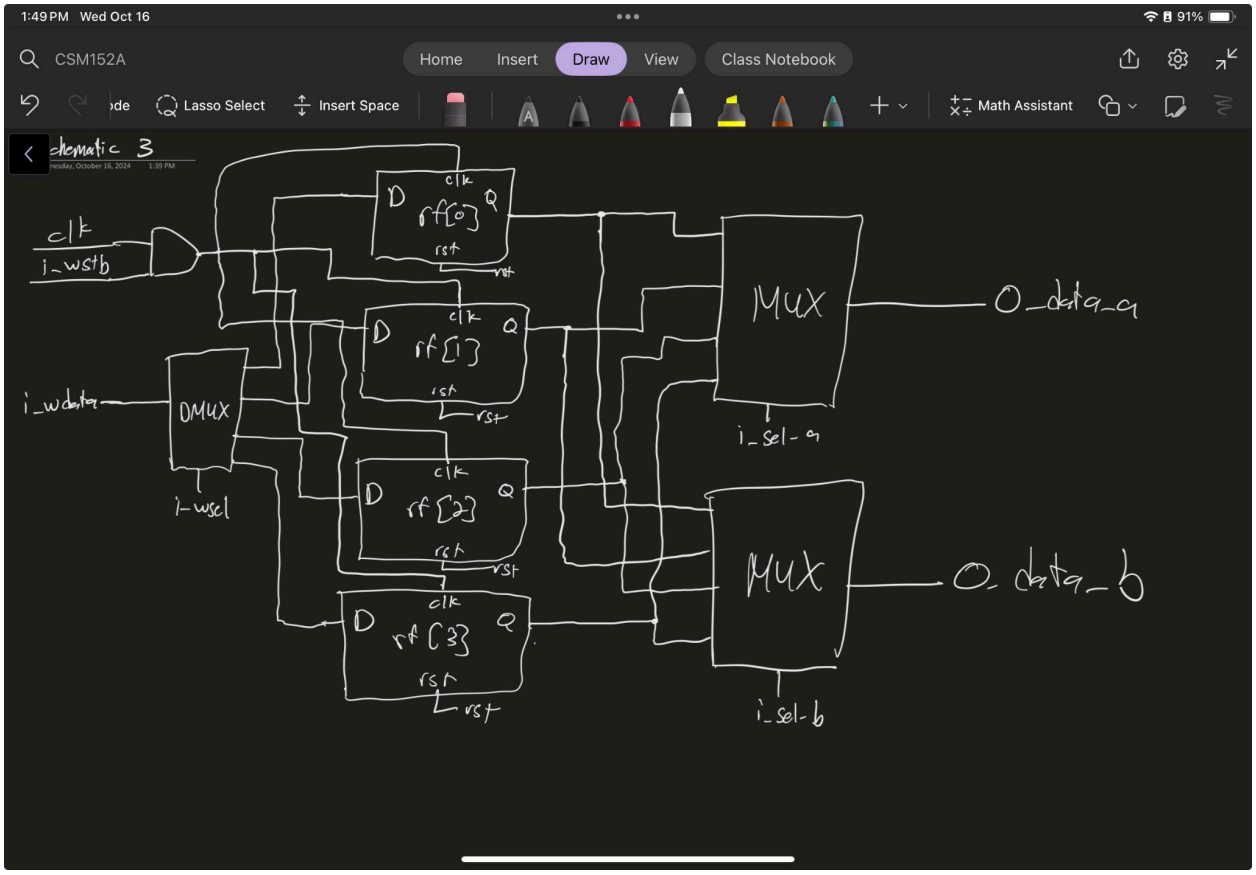
The lines of code where the register values are read out from the register file are lines 35 and 36:

assign o_data_a = rf[i_sel_a];

assign o_data_b = rf[i_sel_b];

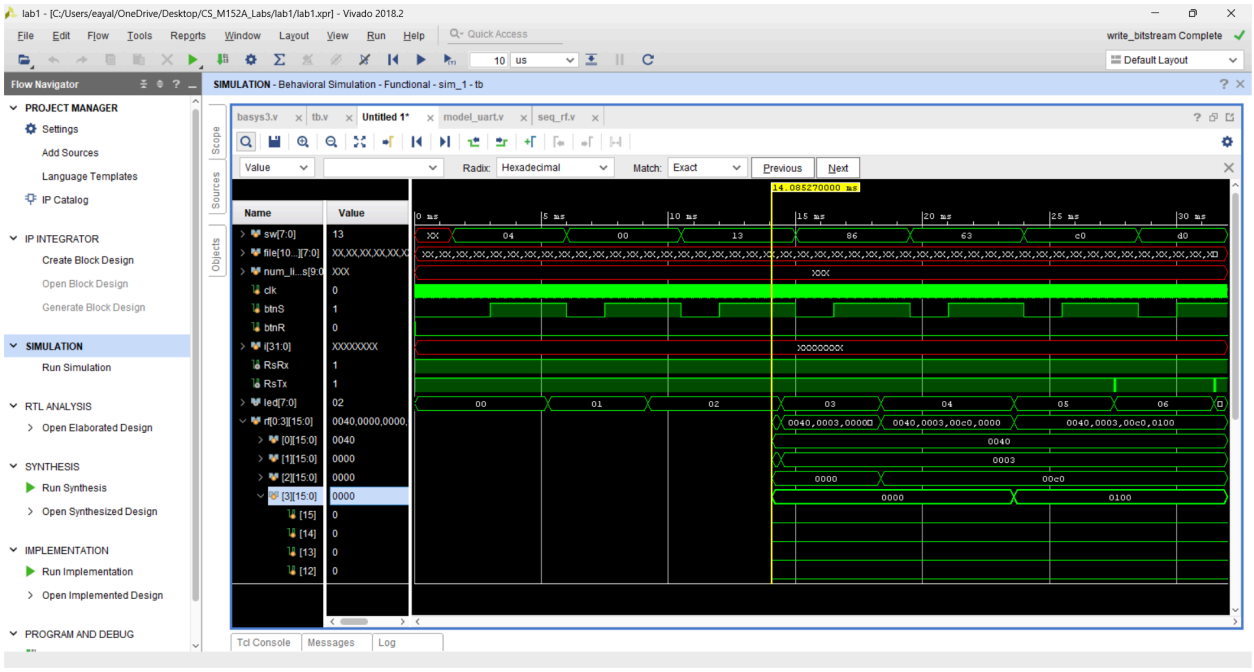
This is combinational logic.

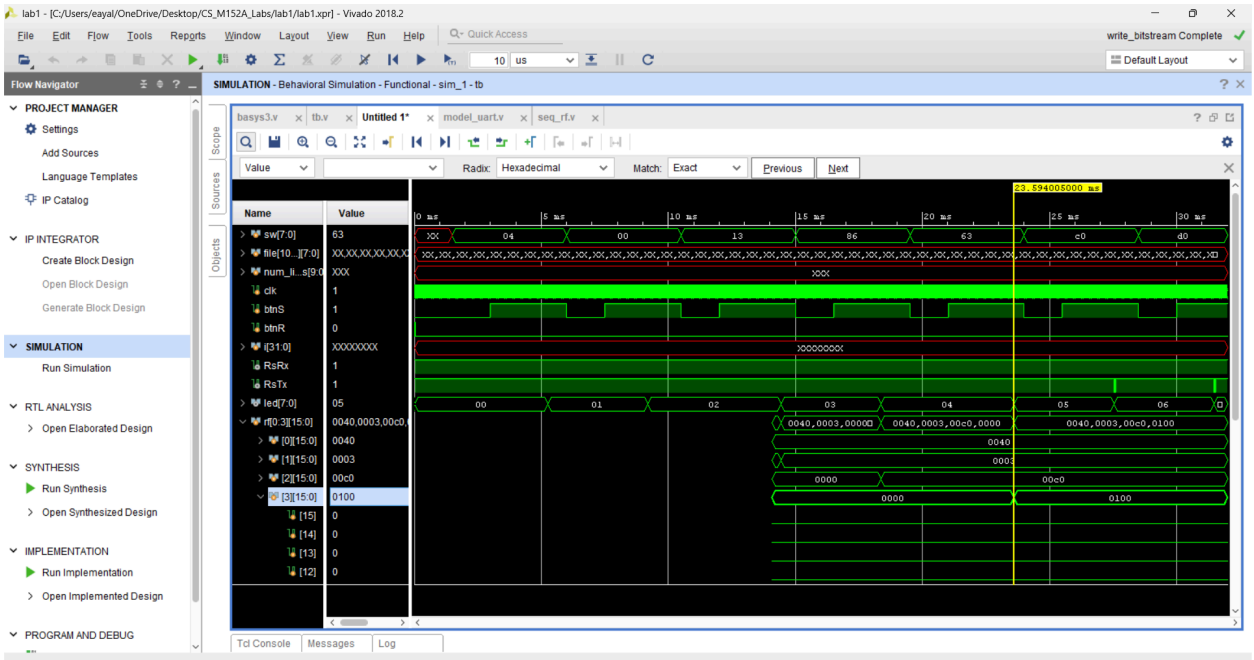
3. **Draw a circuit diagram of the register file block. It should be a translation of the corresponding Verilog code.**



4. **Capture a waveform that shows the first time register 3 is written with a non-zero value.**

Register 3 is propagated a non-zero value at time 14.085270000 ms. You can see the written value at time 23.594005000 ms.


lab1 - [C:/Users/eayal/OneDrive/Desktop/CS_M152A_Labs/lab1/lab1.xpr] - Vivado 2018.2
File Edit Flow Tools Reports Window Layout View Run Help
write_bitstream Complete
Default Layout
Flow Navigator
PROJECT MANAGER
Settings
Add Sources
Language Templates
IP Catalog
IP INTEGRATOR
Create Block Design
Open Block Design
Generate Block Design
SIMULATION
Run Simulation
RTL ANALYSIS
Open Elaborated Design
SYNTHESIS
Run Synthesis
Open Synthesized Design
IMPLEMENTATION
Run Implementation
Open Implemented Design
PROGRAM AND DEBUG
SIMULATION - Behavioral Simulation - Functional - sim_1 - tb
basys3.v
tb.v
Untitled 1*
model_uart.v
seq_rf.v
Value
Radix: Hexadecimal
Match: Exact
Previous
Next
Name
Value
sw[7:0]
file[10...][7:0]
num_li...s[9:0]
clk
btnS
btnR
i[31:0]
RsRx
RsTx
led[7:0]
rf[0:3][15:0]
[0][15:0]
[1][15:0]
[2][15:0]
[3][15:0]
[15]
[14]
[13]
[12]
14.085270000 ns
Tcl Console
Messages
Log


lab1 - [C:/Users/eayal/OneDrive/Desktop/CS_M152A_Labs/lab1/lab1.xpr] - Vivado 2018.2
File Edit Flow Tools Reports Window Layout View Run Help
write_bitstream Complete
Default Layout
Flow Navigator
PROJECT MANAGER
Settings
Add Sources
Language Templates
IP Catalog
IP INTEGRATOR
Create Block Design
Open Block Design
Generate Block Design
SIMULATION
Run Simulation
RTL ANALYSIS
Open Elaborated Design
SYNTHESIS
Run Synthesis
Open Synthesized Design
IMPLEMENTATION
Run Implementation
Open Implemented Design
PROGRAM AND DEBUG
SIMULATION - Behavioral Simulation - Functional - sim_1 - tb
basys3.v
tb.v
Untitled 1*
model_uart.v
seq_rf.v
Value
Radix: Hexadecimal
Match: Exact
Previous
Next
Name
Value
sw[7:0]
file[10...][7:0]
num_li...s[9:0]
clk
btnS
btnR
i[31:0]
RsRx
RsTx
led[7:0]
rf[0:3][15:0]
[0][15:0]
[1][15:0]
[2][15:0]
[3][15:0]
[15]
[14]
[13]
[12]
23.894005000 ns
Tcl Console
Messages
Log