Lab 1

Fnu Shiva Sandesh, Junyoung Kim, and Brandon Tai

March 25, 2017

M152 A - Miodrag Potkonjak

TA: Brinda Alluri

Workshop 1
Clock Dividers

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.

Answer : The exact period of the signal was visible from one of the graph is 1,310,710 ns.

Period = Time between the two rising edges = 58,983,415 ns - 57672,705 ns = 1,310,710 ns

Since we know that the frequency of the signal is T^{-1} therefore the frequency of the clk_en signal is equal to $1310710^{-1}\ ns = 762.95\ Hz$.

Since the clk_en signal is only high for 10 ns is hard to capture the version where the rising and falling edge of the wave can be seen properly. However, the following two pictures represent the signals for better clarity.

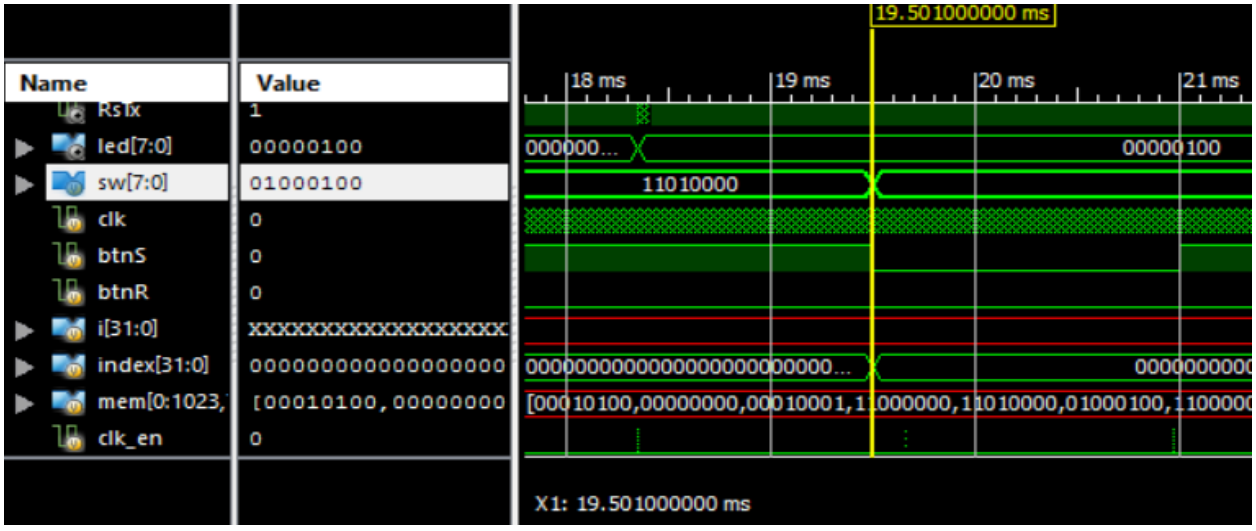

Figure 1: The dotted peaks represents the two occurrences of the clk_en signal.

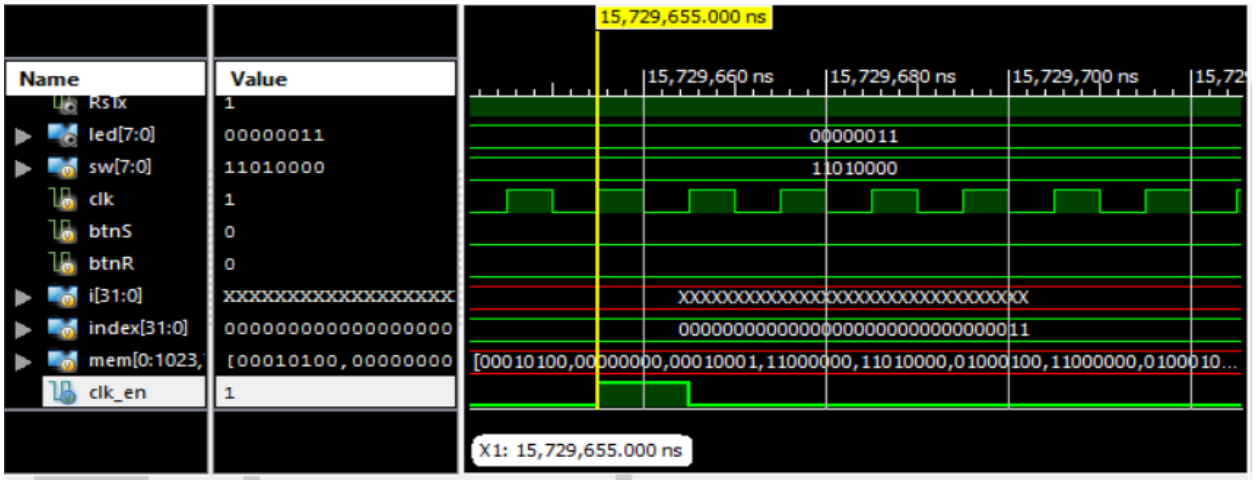

Figure 2 : Depicting the period when clk_en is high.

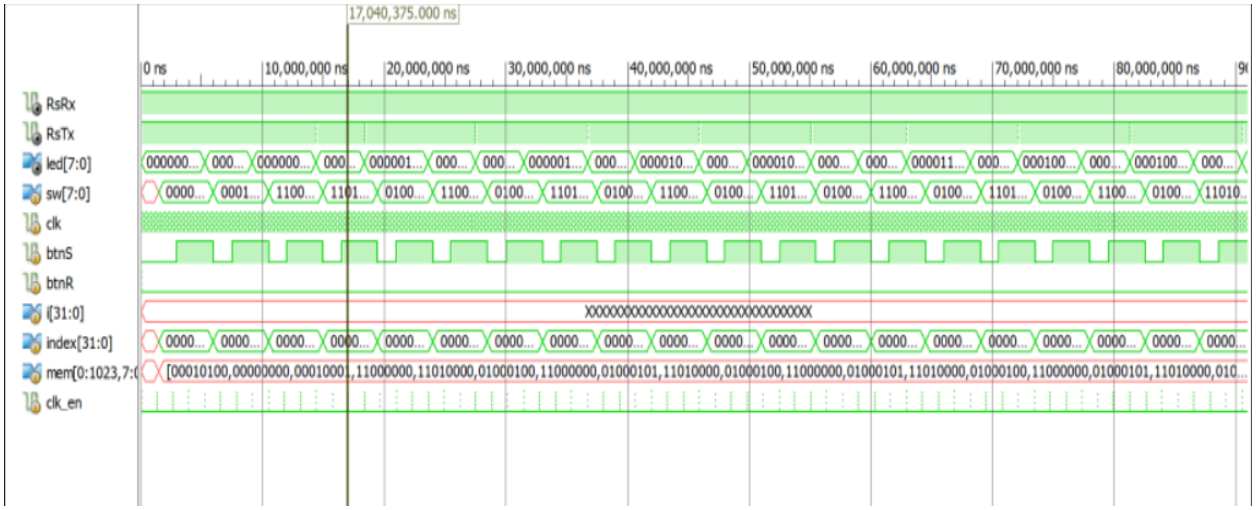

Figure 3: Depicting multiple occurrences of clk_en high.

2. A duty cycle is the percentage of one period in which a signal or system is active: $D = T\ P \times 100\%$, 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?

Answer : Period = Time between the two rising edges

= 58,983,415 ns - 57,672,705 ns = 1,310,710 ns

Time for which signal was high = 58,983,425ns - 58,983,415ns = 10 ns

$D = \frac{10}{1310710} \times 100\% = 7.62 \times 10^{-4}\ \%$.

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

Answer : The clk_dv signal have the value 0 i.e. it has a set of 17 zeros as the value when the clk_en signal was high.

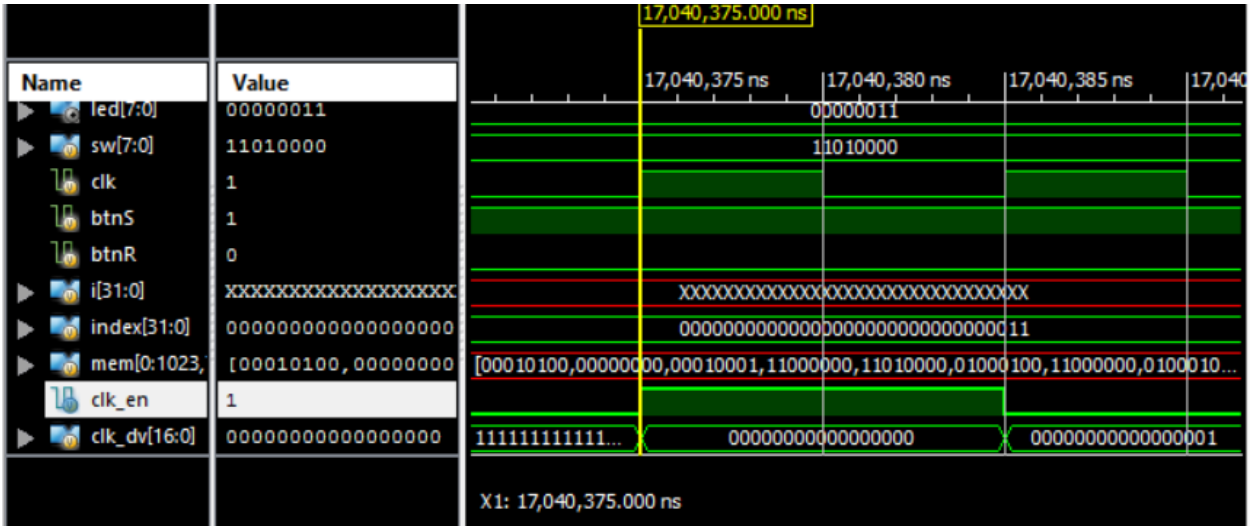

Figure 4 : Representing the value of the clk_dv whe cllk_en is high.

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.

Answer :

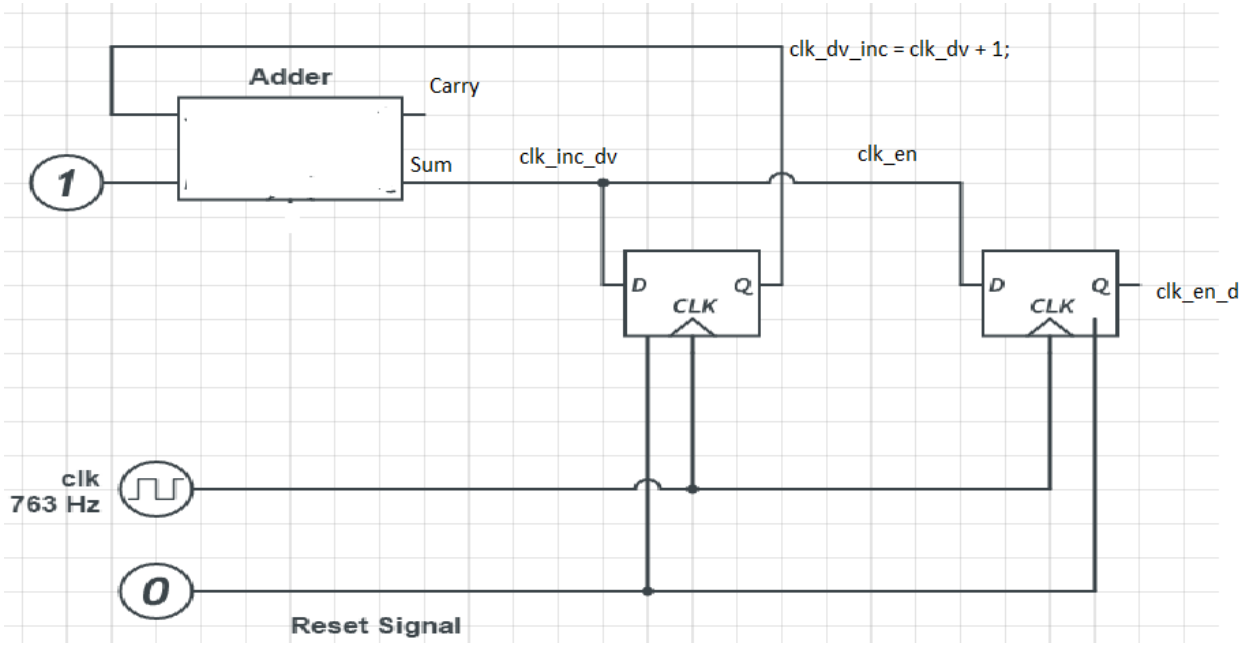

Figure 5 : Representing the signals clk_dv, clk_en, and clk_en_d

Debouncing

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?

Answer : That expression is being used to determine whether or not the given instruction is valid. The program needs to know if the instruction is valid to increment the instruction counter. The clk_en_d signal is a special clock used to safeguard against bouncing where an instruction is toggled in rapid succession because of imperfect connections.

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?

Answer : From the waveform of the signals it is visible that the clk_dv_inc is behaving like a 18-bit counter (this is visible from the indices of the register which go from 0 to 17). The signal clk_en only goes high when the count sequence transitions from a series of all 1's to all 0's. Now we know that each of the slices are 10 ns long. The clk_en transitions to high for 10 ns in every 2^{18} counts of the clock pulse. This corresponds to a duty cycle of 3.81×10^{-4} %. Now if we assign clk_en <= clk_dv[16], the clk_en transitions to high for 10 ns in every 2^{17} counts of the clock pulse. This corresponds to a duty cycle of 7.62×10^{-4} %. If we take the ratio of the duty cycle from clk_dv_inc[17] to clk_dv[16] it is 50%.

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.

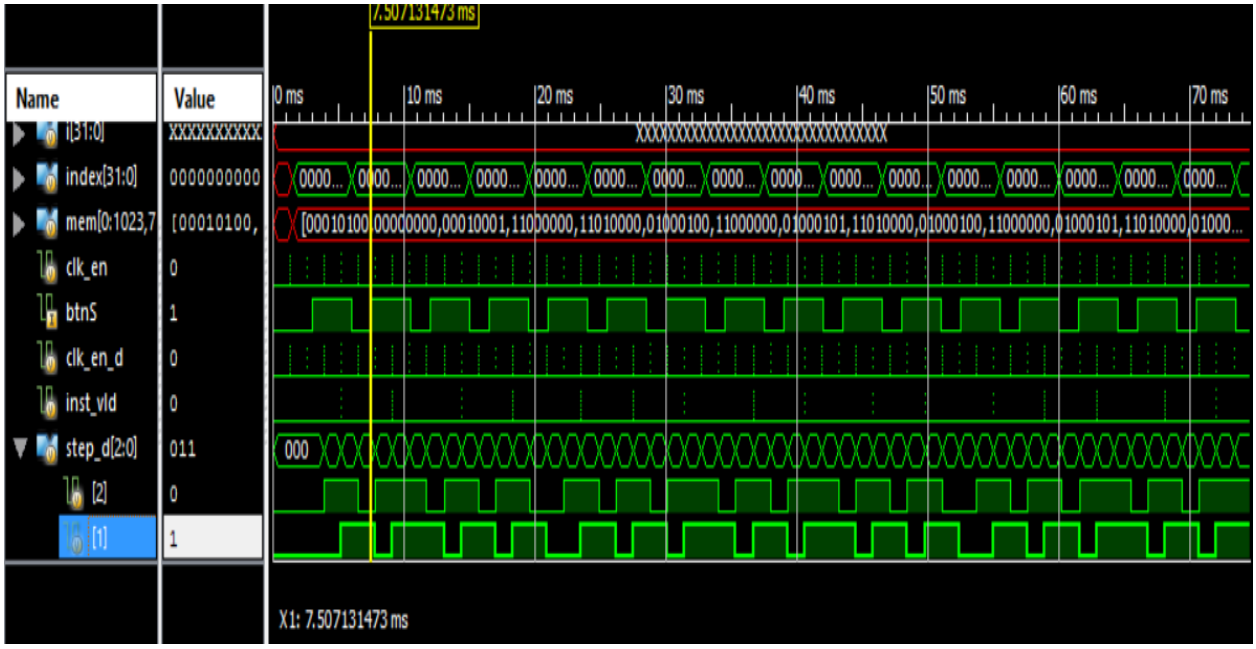

Figure 6 : Waveform capturing clk_en, step_d[1], step_d[0], btnS, clk_en_d, and inst_vld.

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

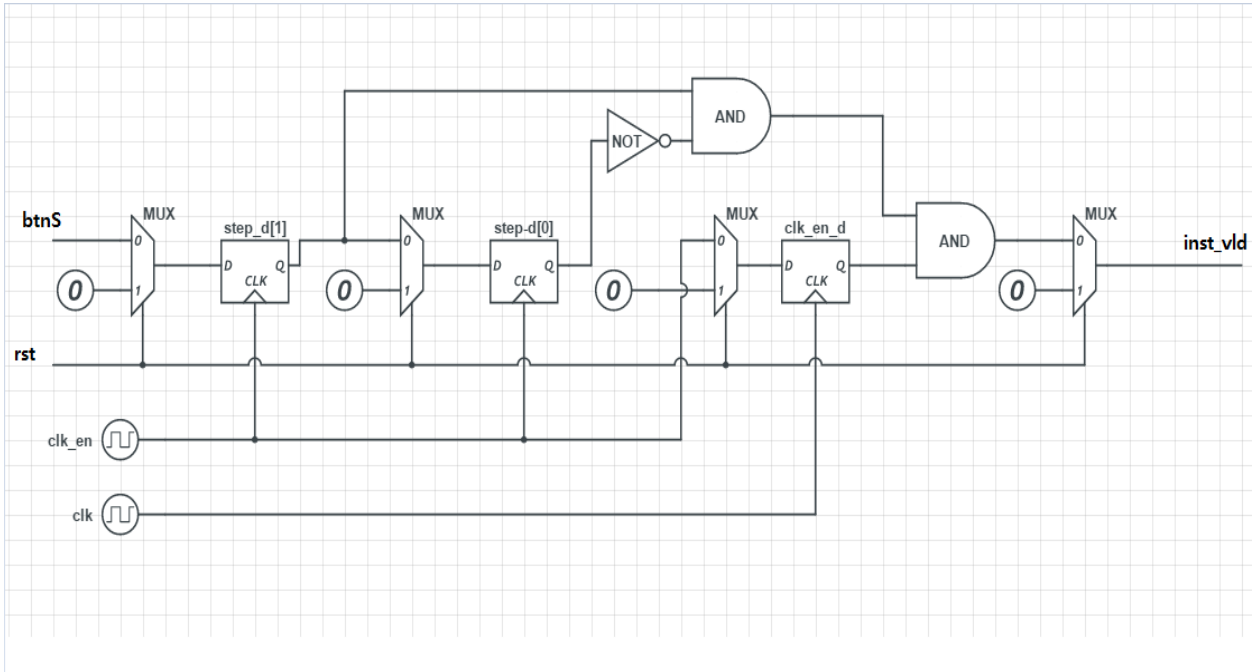

Figure:7 schematic/diagram of clk_en, step_d[1], step_d[0], btnS, clk_en_d, and inst_vld

Register File

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

Answer:

```
else if (i_wstb)
   rf[i_wsel] <= i_wdata;
```

- This is a sequential logic because the input of following registers are depend on the output data of previous registers.

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?

Answer:

```
assign o_data_a = rf[i_sel_a];
assign o_data_b = rf[i_sel_b];
```

- This is a combinatorial logic because the data assigned(read) from the register is independent of other registers.

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

Answer:

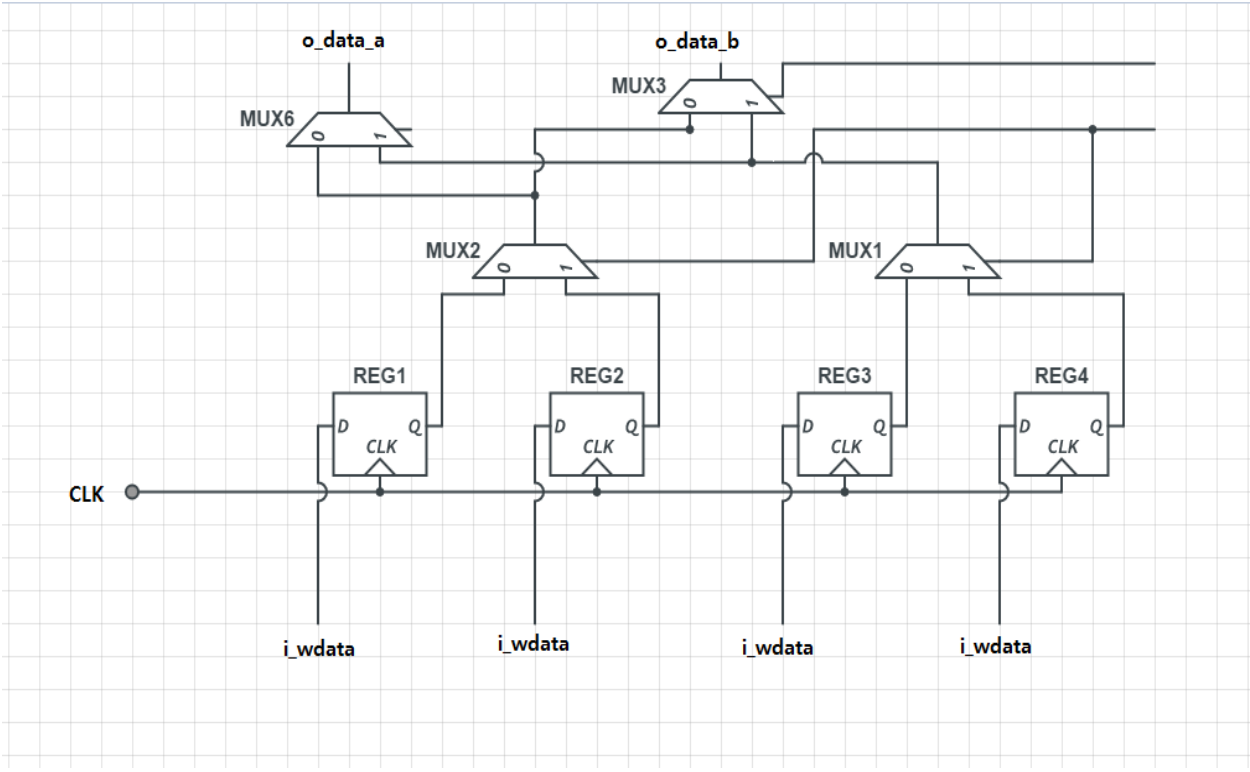

Figure 8: Circuit diagram of the register file block

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

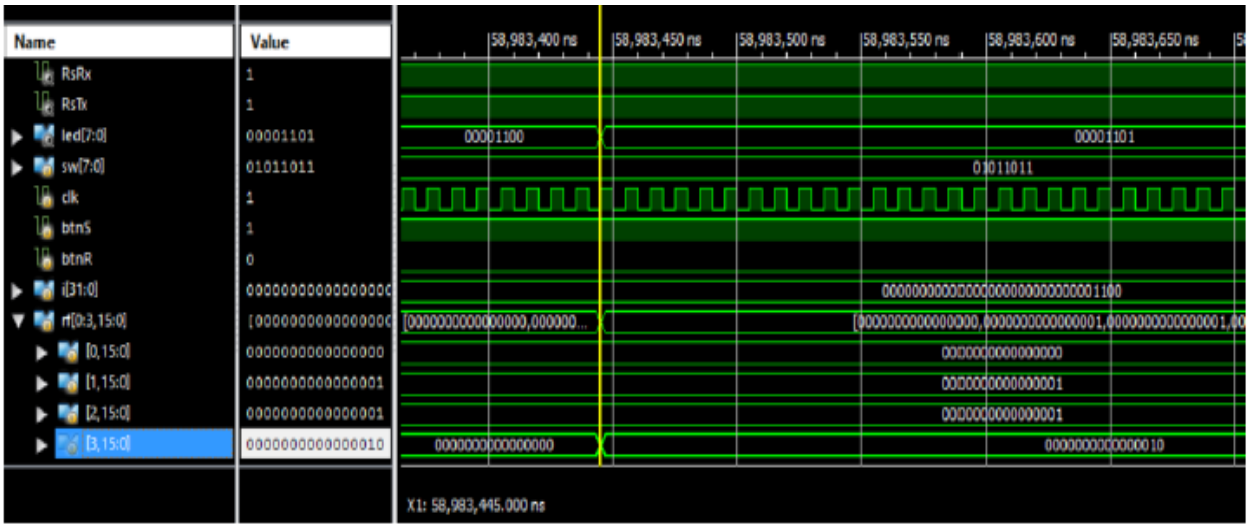

Figure 9: waveform that shows the first time register 3 is written with a non-zero value

Workshop 2

An Easier Way to Load Sequencer Program

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

Answer: The instructions are performed with hlp of taskRunInst and following code is for the same.

```
task tskRunInst;
    input [7:0] inst;
    begin
      $display ("%d ... Running instruction %08b", $stime, inst);
      sw = inst;
      #1500000 btnS = 1;
      #3000000 btnS = 0;
    end
  endtask //
```

The instructions are sent to the Unit Under Test (UUT) are the eight instructions which are hardcoded in the tb.v these instructions are as follows.

```
      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)
```

After that the display statements are used to represent the completion of the use task.

```
$display("%d ... instruction %08b executed", $stime, uut_.inst_wd);
$display("%d ... led output changed to %08b", $stime, led);
```

2. Which user tasks are called in this process?

Answer: The user task that are called during the process depends on the instruction being executed. However, all the task invoke the tasks of Push, Add, Mult and Send. Now we are also reading the instructions from the file(for fibonacci) so it is also invoking other user processes for reading the file into the array.