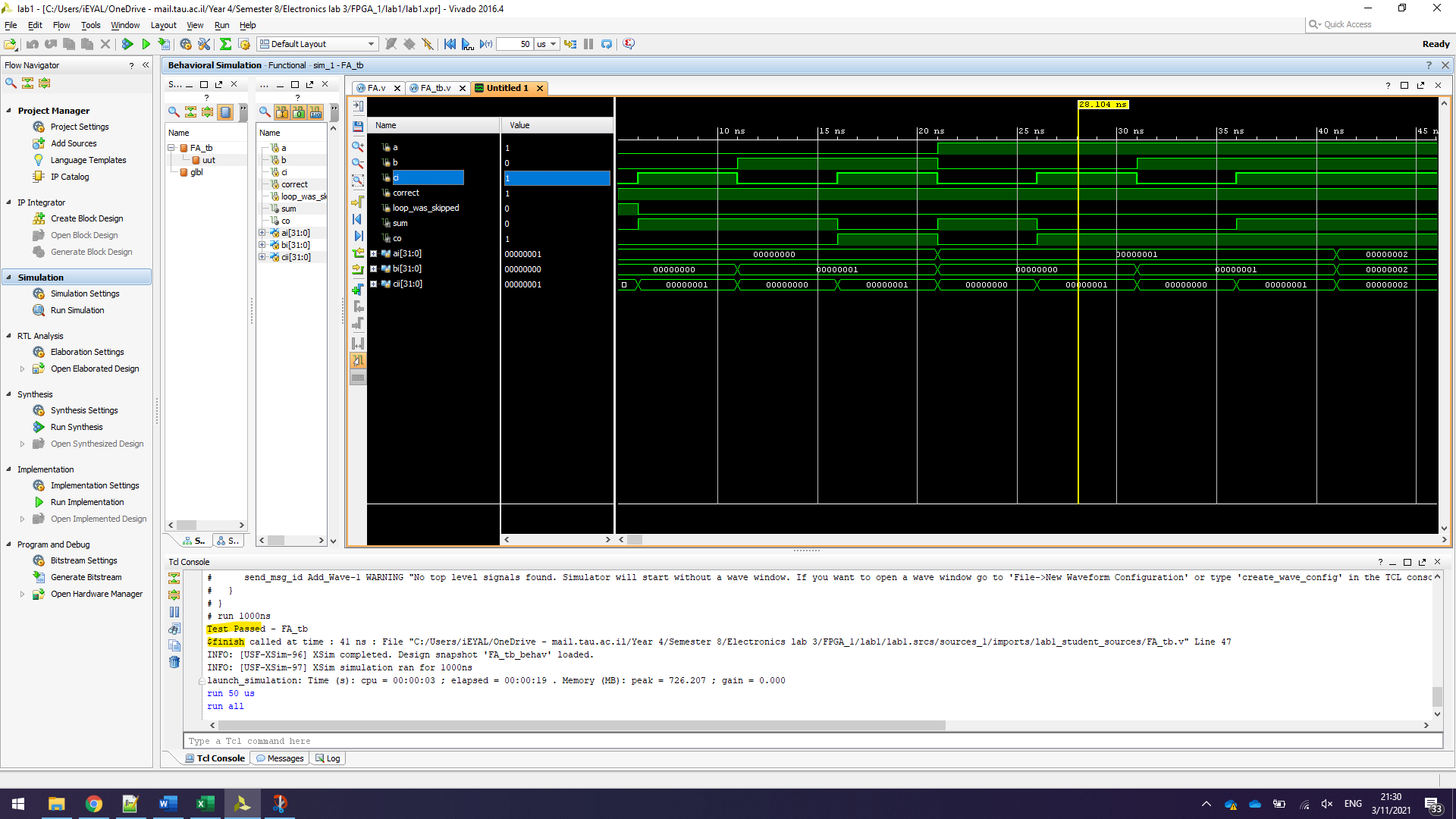
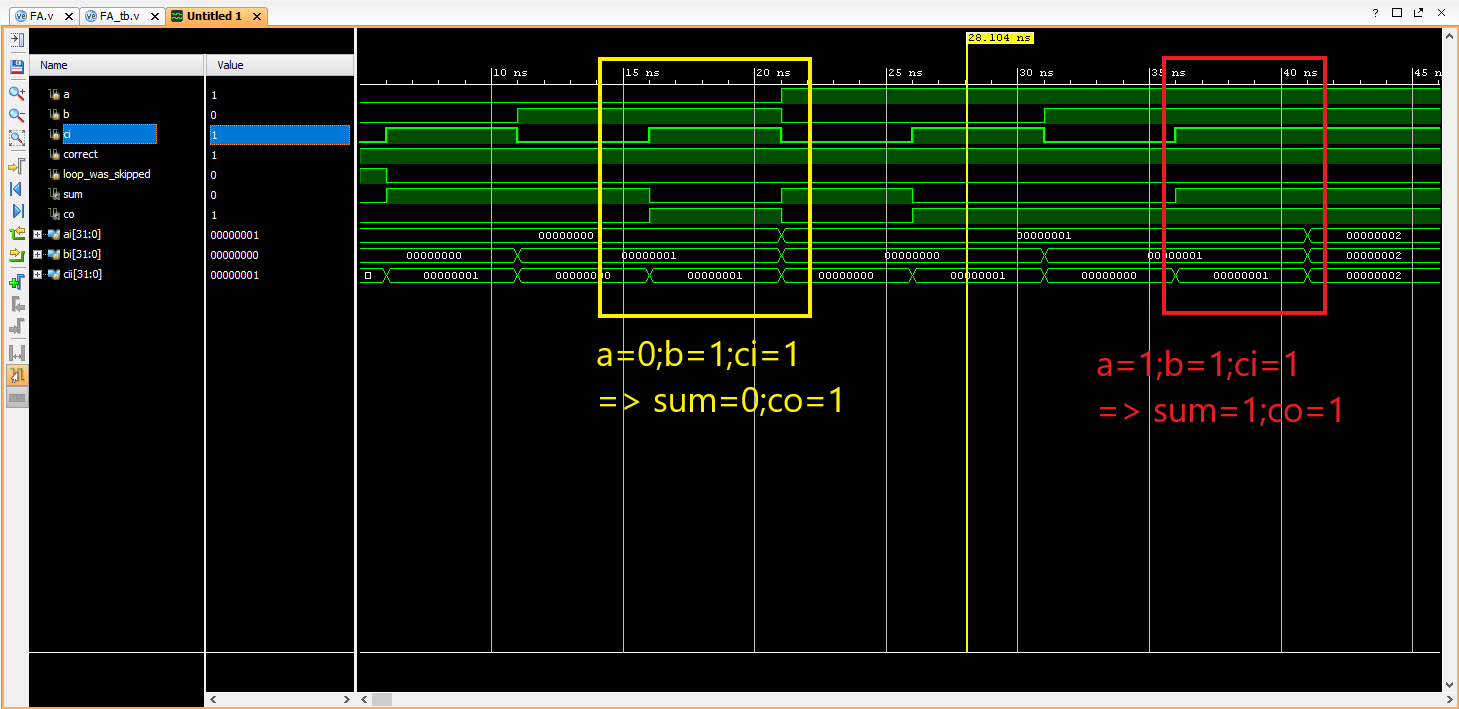
**FPGA- PART 1 PRELIMINARY REPORT**

Michal Keren (204783161) & Itamar Eyal (302309539)

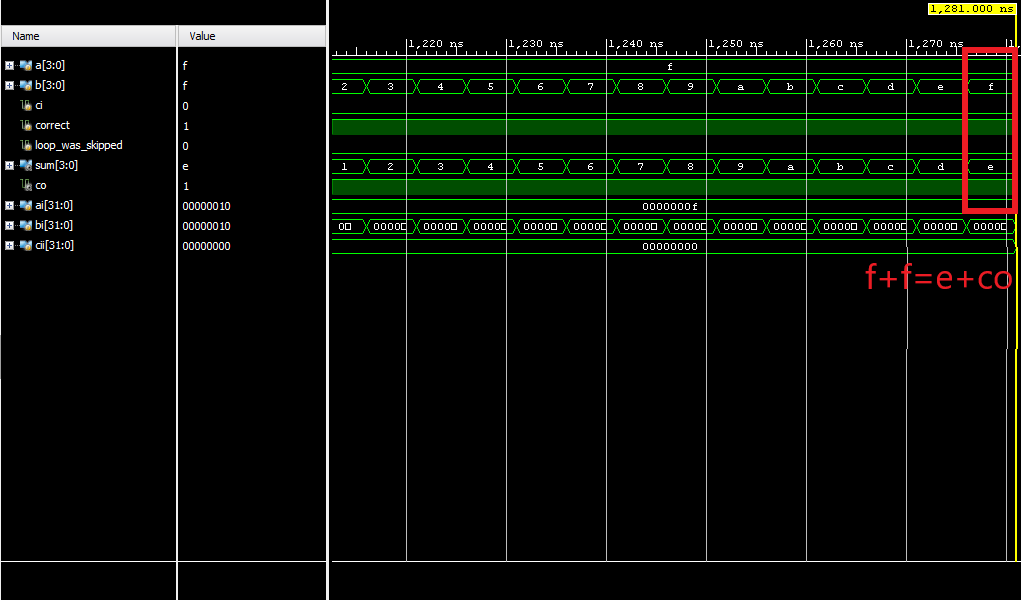
**Q2**

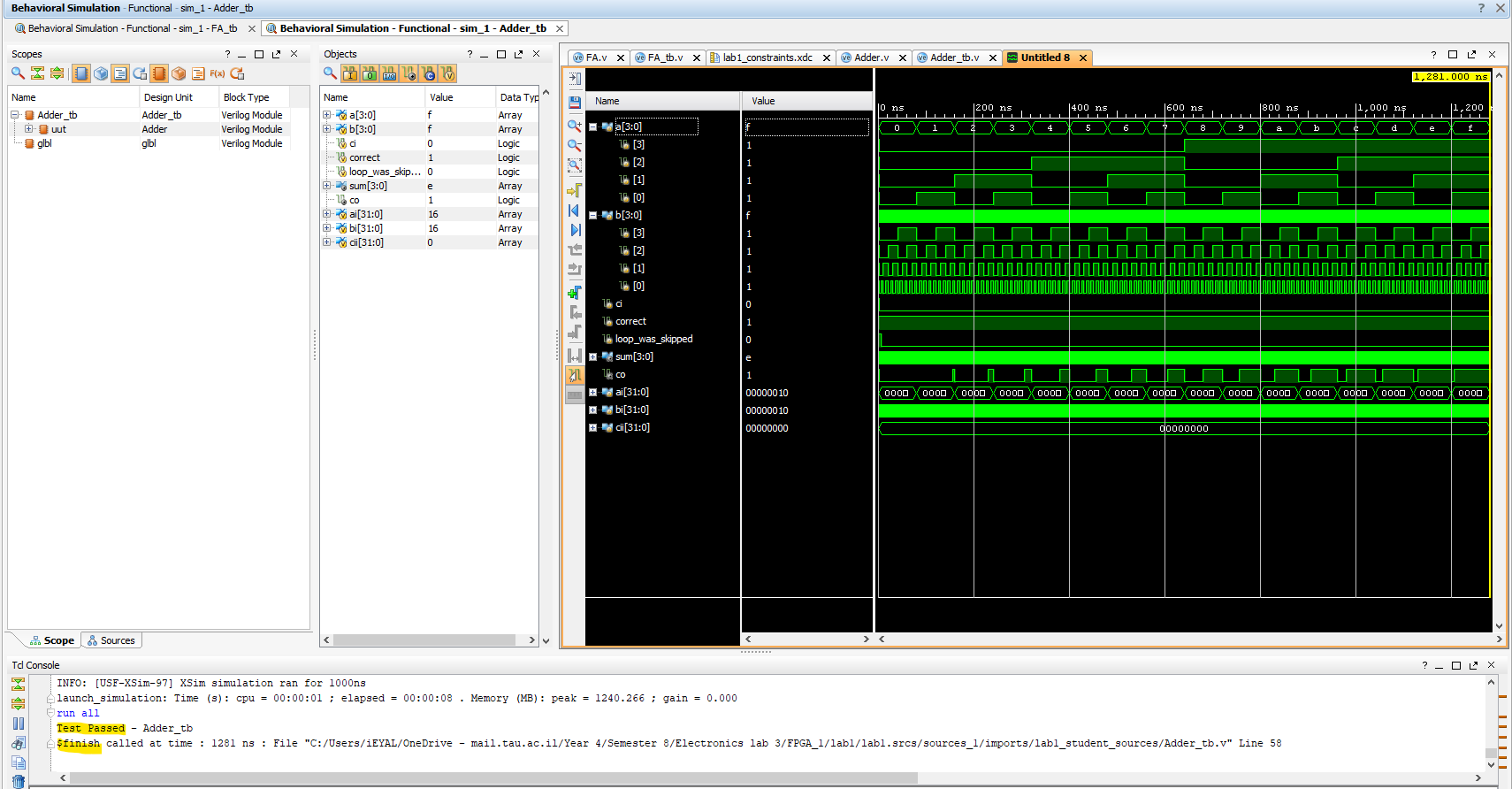
2.d.



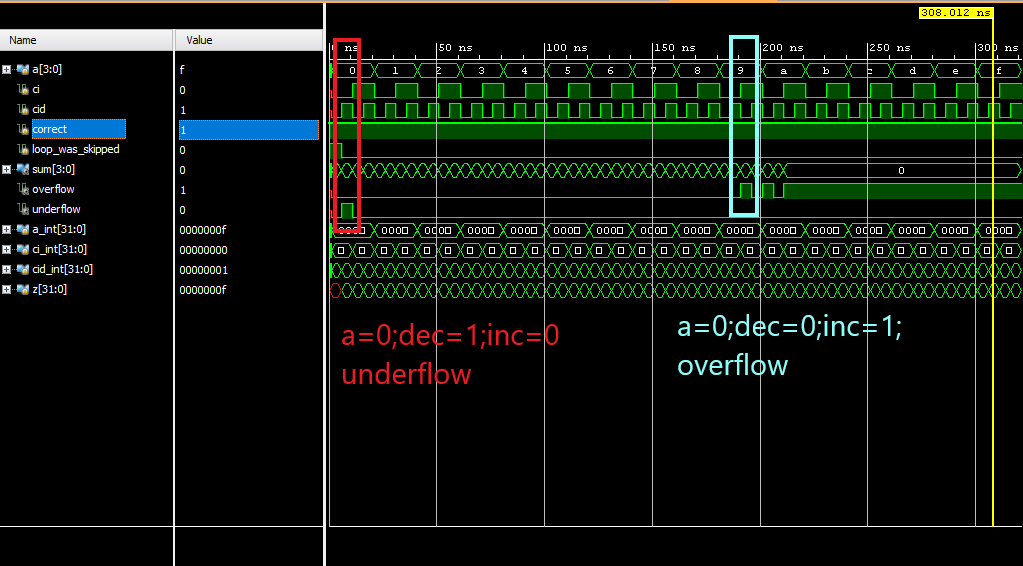


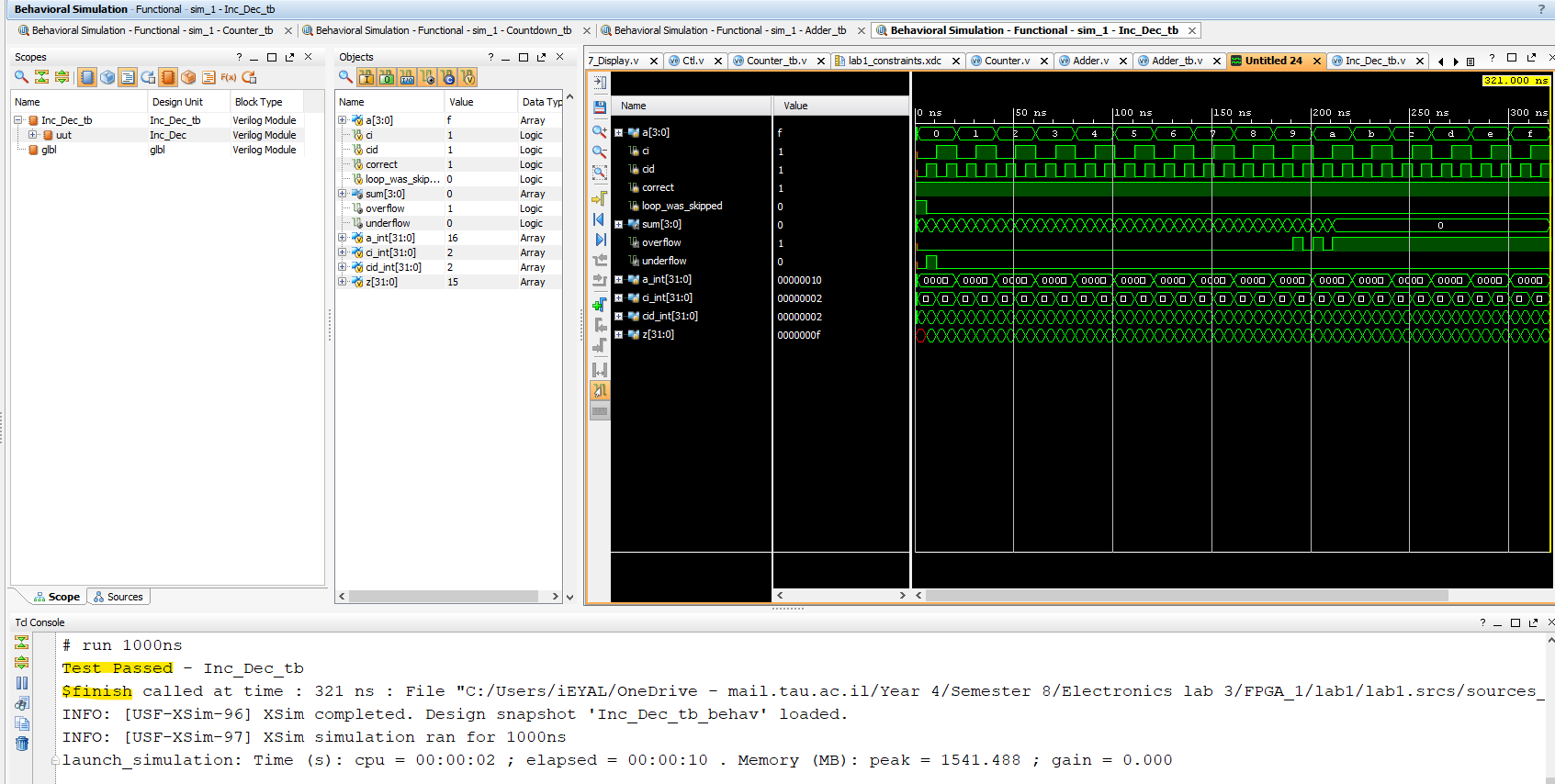
**Q3**

3.d. 



**Q4**

4.d.



**Q5**

5.a.

First, we shall implement an inc-dec(100MHz) in order to update the clock count (clk\_cnt).

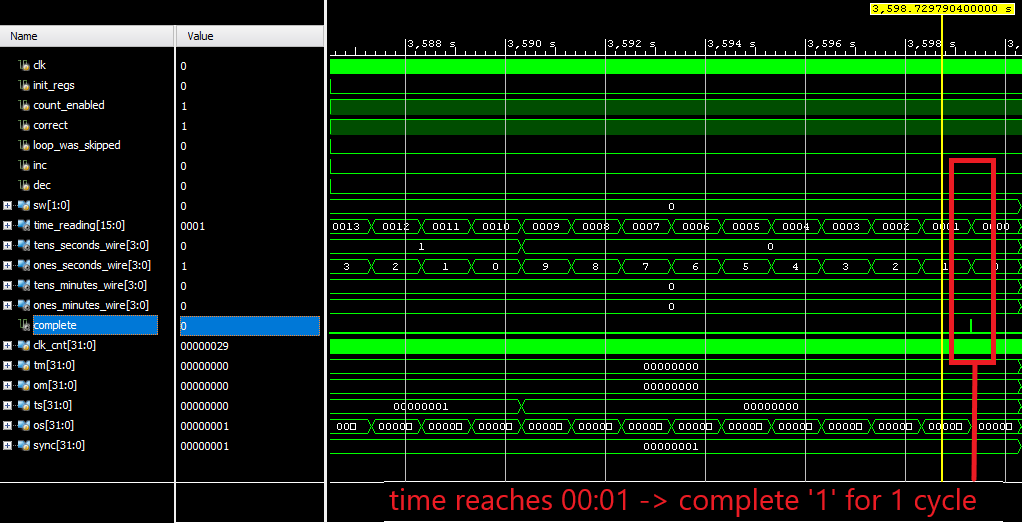
The ‘inc’ input to this instance will be high from the moment that count\_enabled was switched to high, and the clock count will be reset to zero at that moment as well. (‘dec’ input is set to zero)

Whenever clk\_cnt equals the clock frequency minus 1 (minus 1 because we count from zero), we know that a full clock cycle is complete and therefore a decrease of the seconds should accrue.

So, by implementing an inc-dec(10) for the single seconds (ones\_sec) , fed by ‘dec’ input that equals the condition : (clk\_cnt = CLK\_FREQ-1), the single second will be decreased every 100MHz clock cycle as desired.

In order to update the rest of the time reading, we will implement three additional inc-dec(L) instances. (L=6 for the tens\_sec & tens\_min ; L=10 for the ones\_min)

We will use the ‘underflow’ output of the previous instance to feed the ‘dec’ input of the next. (ones\_sec -> tens\_sec -> ones\_min -> tens\_min ).

5.e.i.

5.e.ii.



5.e.ii.

Theoretically, the test bench will still test the counting from 59:59 to 00:00, but in practice, the number of iterations required to complete this counting is much larger than the iterations limit.

let’s denote the num of tb iteration,when CLK\_FREQ= 100Hz, by itr.

if we increase CLK\_FREQ to 100MHz, the number of tb iteration will be ( iter\*M) which is obviously a very large number.

this explains the error received when trying to run the tb after the changes:



note:

if we change the timescale as well as the CLK\_FREQ by the same factor (CLK\_FREQ multiplied by 1M and time res divided by 1M), the delay times (#) will behave exactly like before (since the CLK\_PERIOD\_IN\_TIMEUNITS is divided by 1Msec and the time scale is also decreased by 1Msec ).

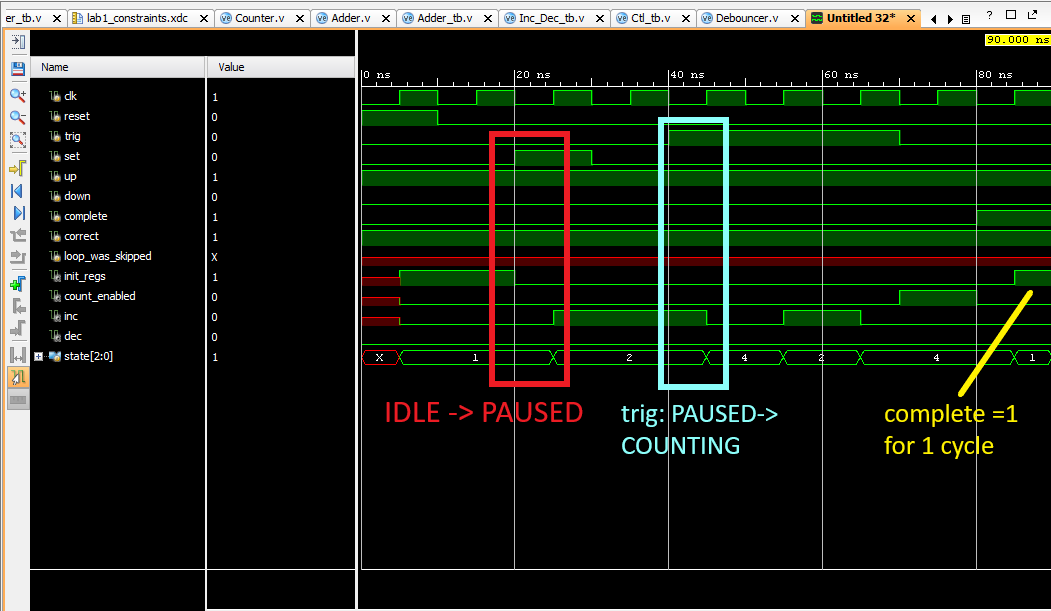
**Q6**

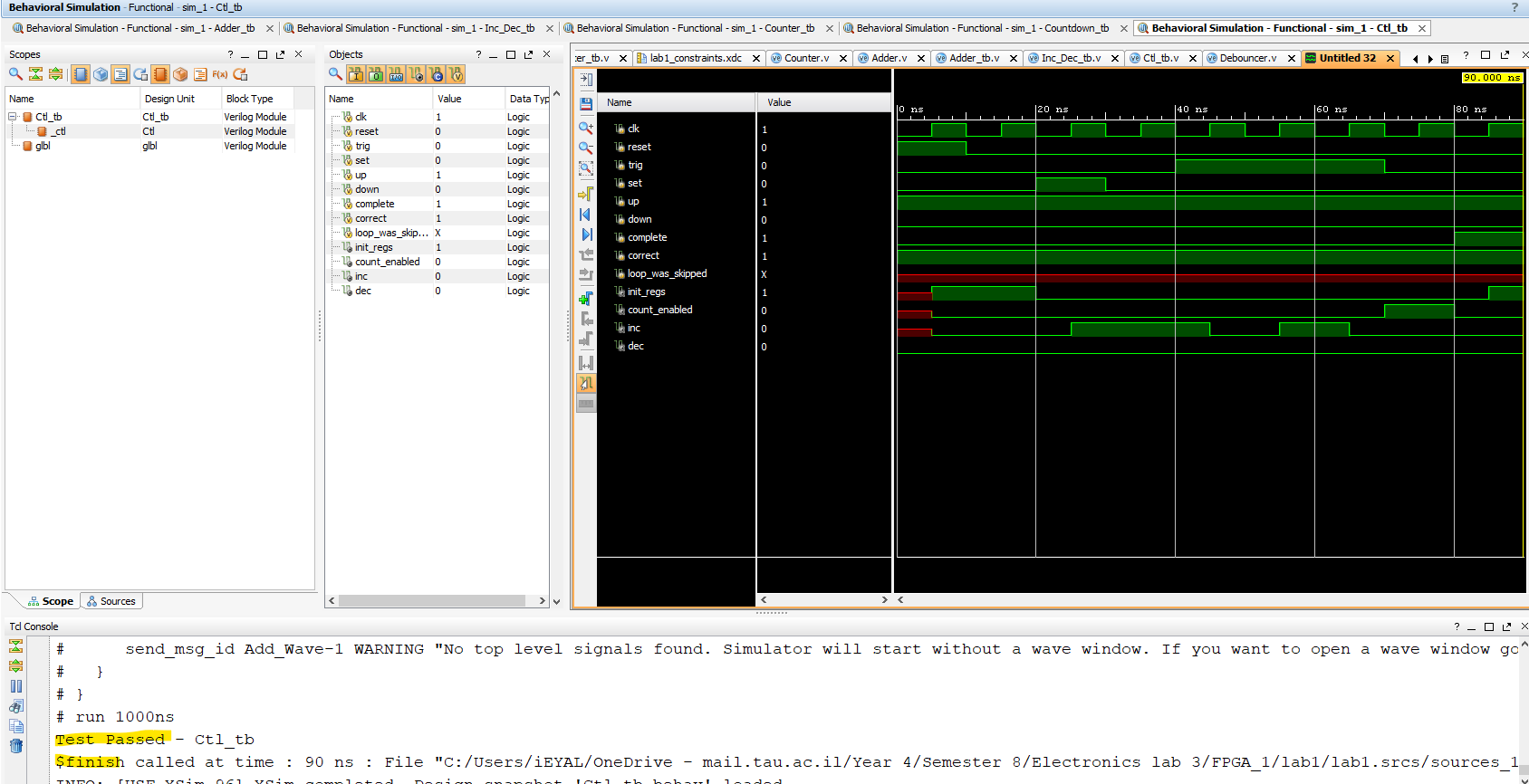
6.d.i.

This FSM is **not** a pure Moore FSM.

The output at time t depends on the state at time t and also on the inputs at time t.

For example, if state(t) = ‘IDLE’, output(t) = ‘0000’ or ‘1000’ depending on inputs(t).

6.d.ii.



**Q7**

7.c.

Discuss the tradeoff between choosing high and low values for the counter, and how is this related to the low-pass-filter debouncing?

By selecting a high value for the counter we make it more sensitive (it’s closer to the threshold). Meaning a short press on the btn will be enough to trigger a stable single-cycle output.

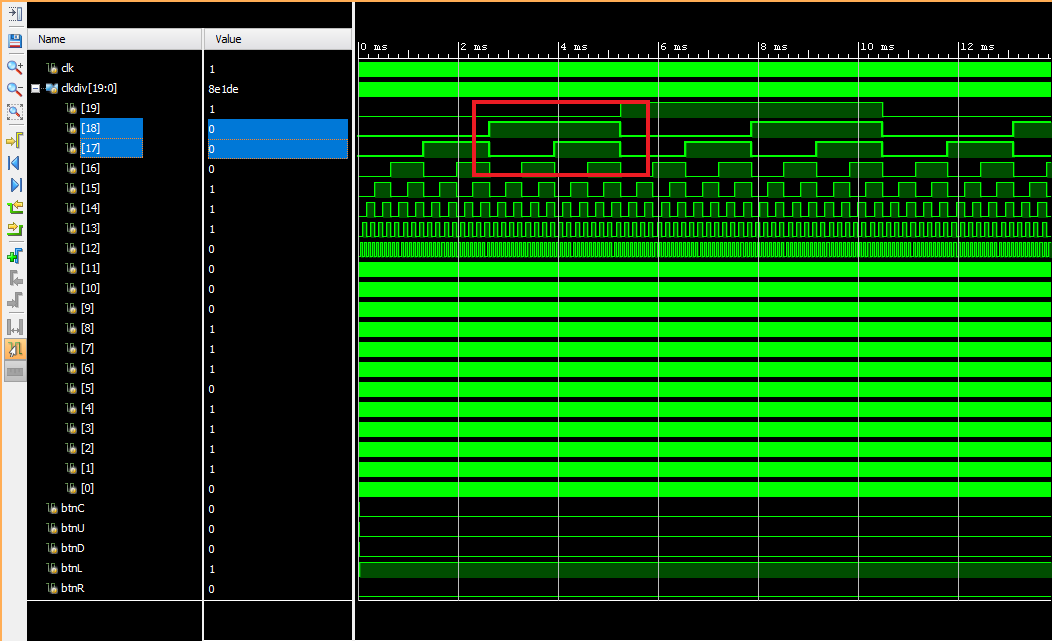
However, it also means that 2 close-together presses intended as one press might be interpreted by the Debouncer as 2 distinctive presses. In addition, some noises might also trigger a stable output.

the tradeoff for a low value counter will be in the very low sensitivity it might have, Making the user press for a longer period of time in order to trigger a reaction. It makes the btn more robust to toggling.

Making the btn less sensitive is decreasing the frequency of it's toggling time, thus making the circuit more LPF by shortening it’s BW.

**Q8**

8.c



clkdiv has a ratio of 2:1 time\_period for every bit from LSB to MSB. the LSB is changing at 100MHZ thus making it’s time\_period=10^-8sec.

in order to achieve a flashing time\_period between 1-16msec, we chose bits [18:17]. these bits allow the display to change as follows:

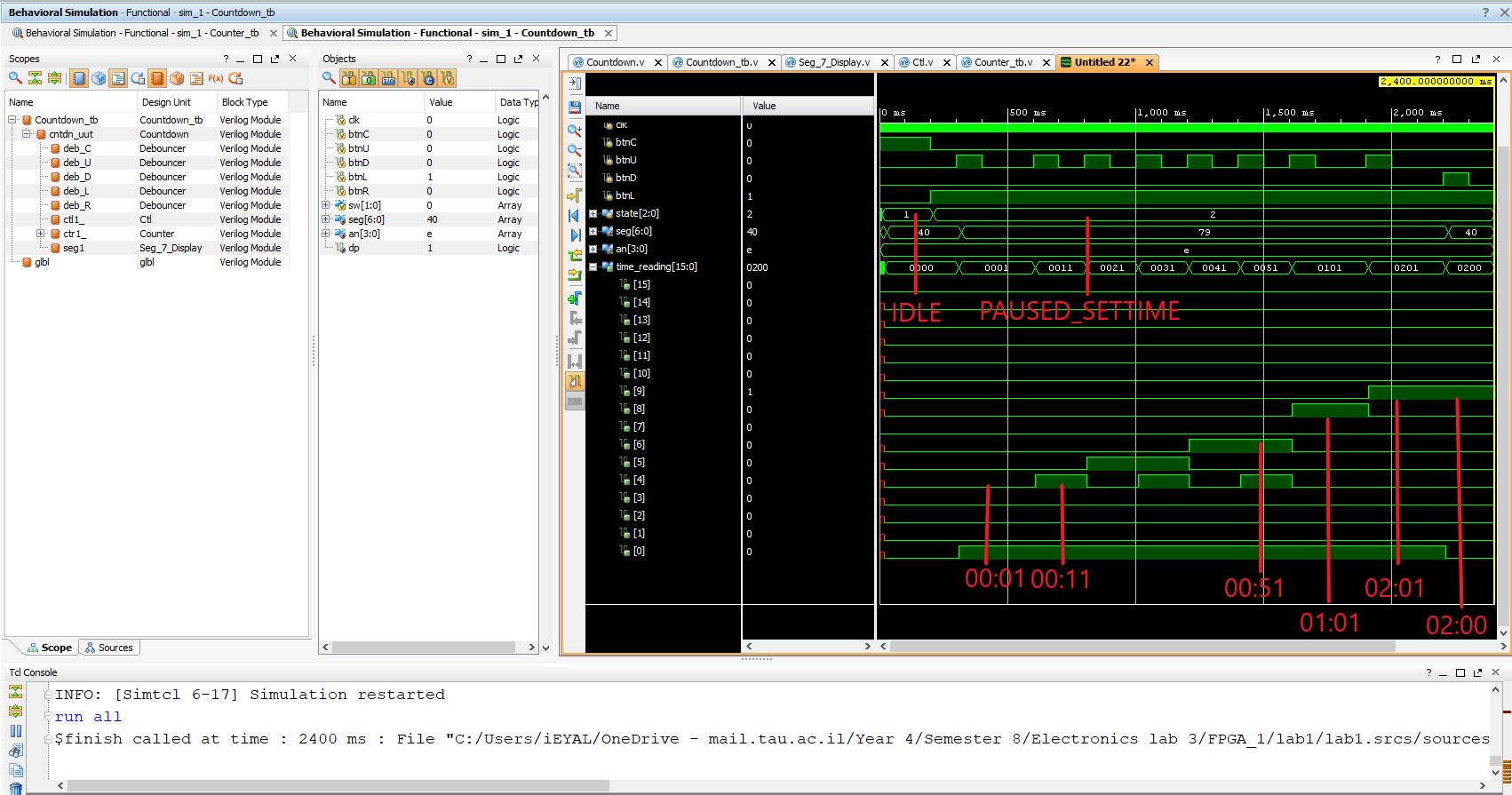
bit17period= (100M)^-1 \* 2^17 = 1.31 msec

bit18 period= (100M)^-1 \* 2^18 = 2.62 msec

**The signal simulation above shows the same results.**

**Q10**

10.d.i.



10.d.ii.

In our Countdown\_tb simulation we wanted to check the following functions of Countdown:

* switching from state IDLE to PAUSED\_SETTIME using reset & set
* changing the displayed time using the UP & DOWN btns
* expected time\_reading value displayed after incrementing / decrementing

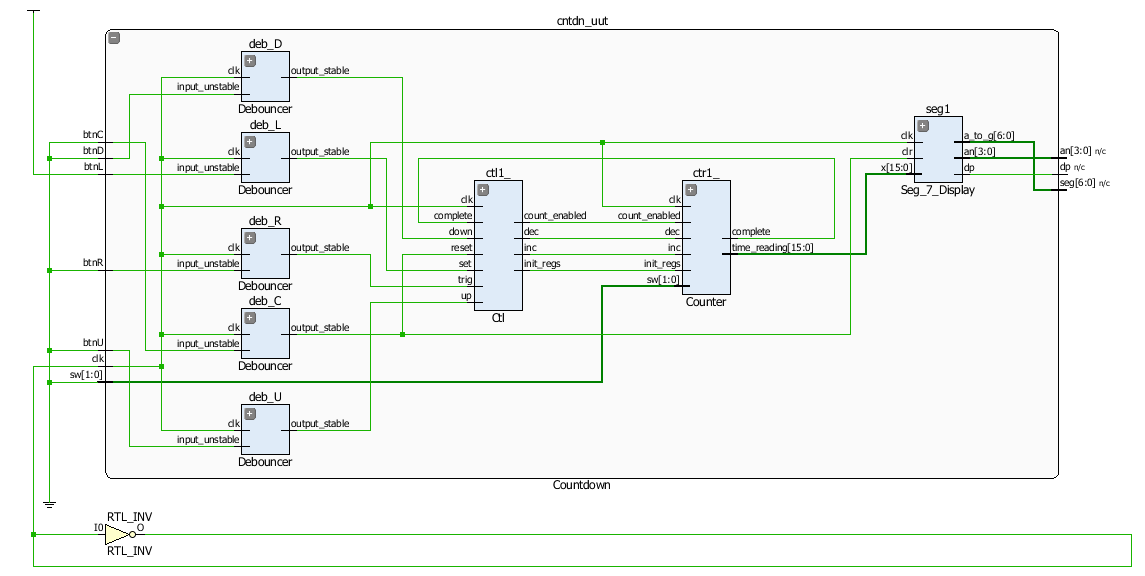
We did so by initializing the FSM to IDLE (btn reset), then set it to PAUSED\_SETTIME (btn set) and afterwards setting the time displayed (btns up/down).

The btns were simulated by regs and the outputs by wires.

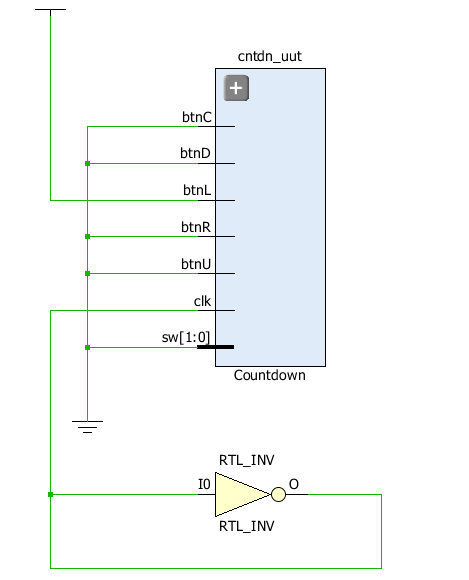
Also, we wanted to make sure that incrementing a digit over limit (tens\_sec over 5 for example, will trigger a shift in the next digit and set the overflowing digit to 0). we did so by incrementing sec\_tens 6 times, watching it change to 0 and min\_ones incrementing to 1.

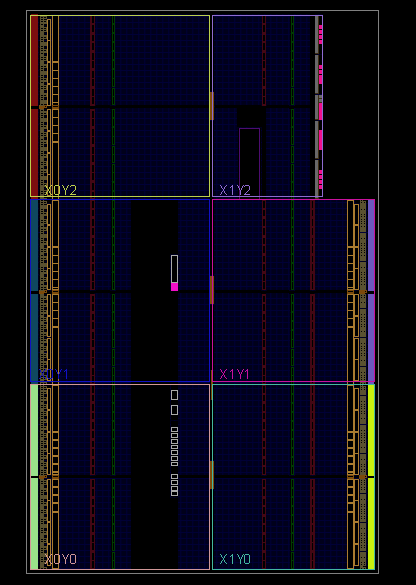
All these were examined in out TB and displayed in the behavioral simulation.

**Q11**

11.a.

11.b.





The elaborated design shows the logical data & control paths in a graphic manner. It doesn’t represent the physical positioning of the gates and wires on the board. On the other hand, the post synthesis design shows the actual wiring and layout of the circuit. Moreover, it optimizes the logic by omitting redundant registers from the design thus saving room on the chip.

11.c.

