Advanced Digital Logic

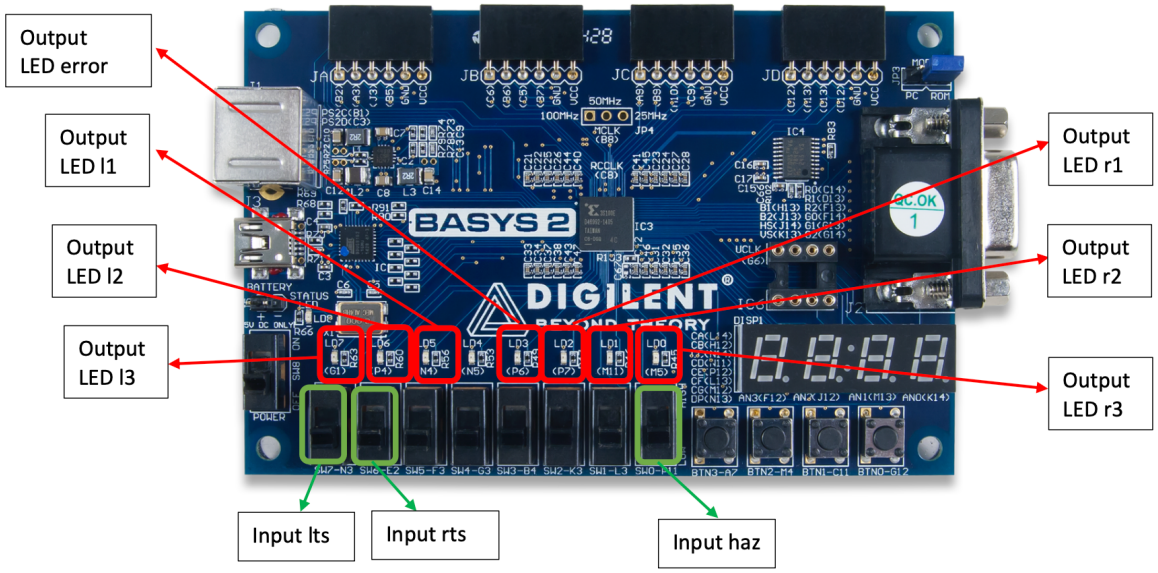
ENGR – UH 2310

Spring 2025

Instructor: Muhammad Hassan Jamil

Demarce Williams (dsw9740)
Aman Sunesh (as18181)

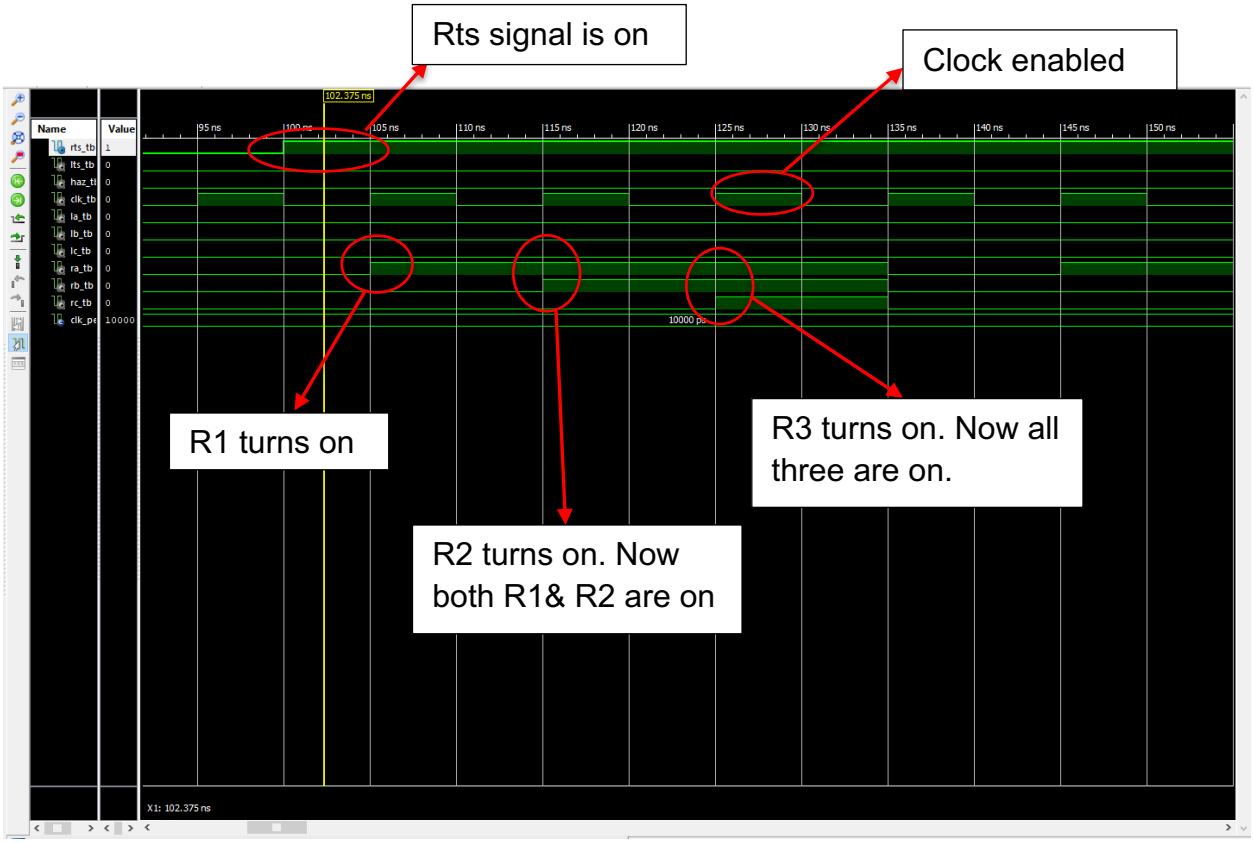
LABELLED HD PICTURE OF INPUTS AND OUTPUTS ON FPGA BOARD FOR FSM TAIL LIGHTS CONTROLLER



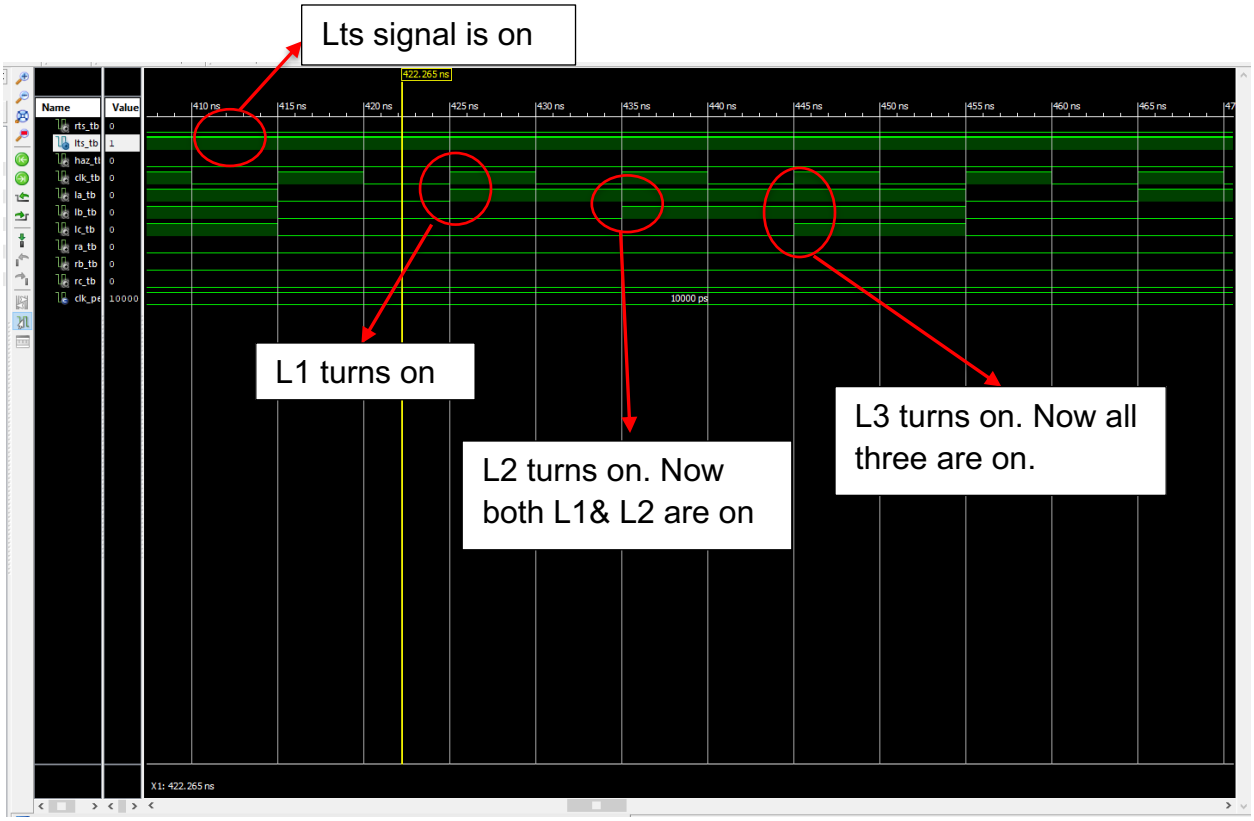
TASK 3: BEFORE CORRECTION

Test For The Three Regular Cases:
For each of the respective case, all other inputs are off for each.

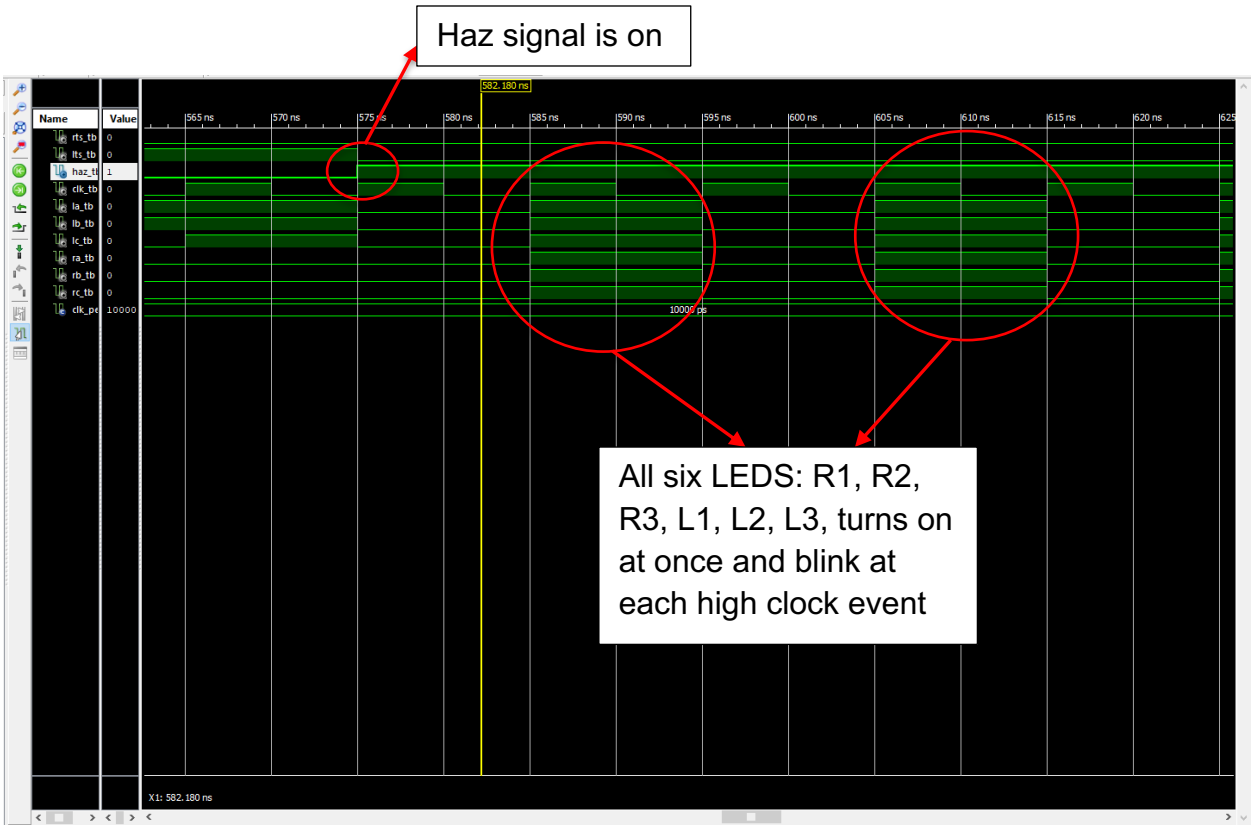
a) Only rts on



b) Only lts on

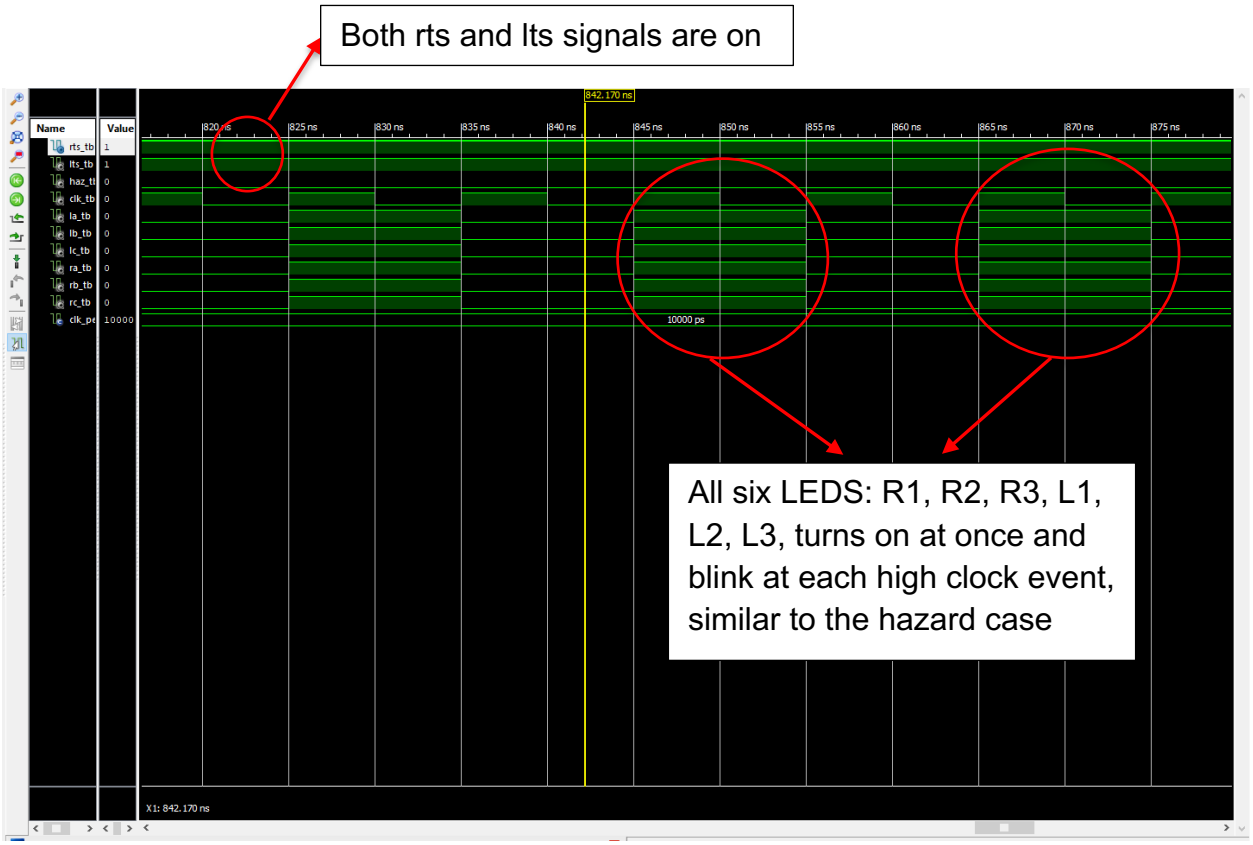

Lts signal is on
L1 turns on
L2 turns on. Now both L1& L2 are on
L3 turns on. Now all three are on.
Name
Value
rts_tb
lts_tb
haz_tb
clk_tb
la_tb
lb_tb
lc_tb
ra_tb
rb_tb
rc_tb
clk_pe
422.265 ns
410 ns
415 ns
420 ns
425 ns
430 ns
435 ns
440 ns
445 ns
450 ns
455 ns
460 ns
465 ns
10000 ps
X1: 422.265 ns

c) Only haz on.

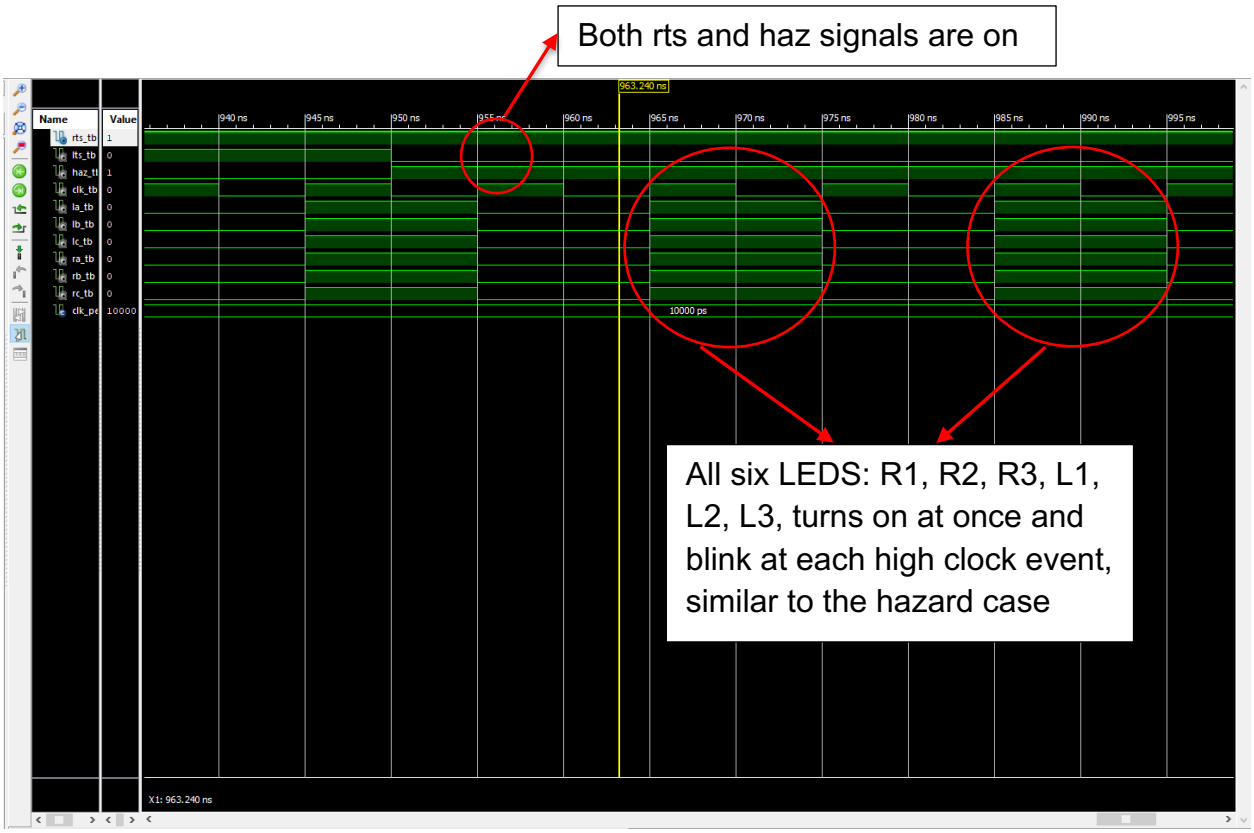

Haz signal is on
582.180 ns
565 ns
570 ns
575 ns
580 ns
585 ns
590 ns
595 ns
600 ns
605 ns
610 ns
615 ns
620 ns
Name
Value
rts_tb
lts_tb
haz_tb
clk_tb
la_tb
lb_tb
lc_tb
ra_tb
rb_tb
rc_tb
clk_pe
10000 ps
All six LEDS: R1, R2, R3, L1, L2, L3, turns on at once and blink at each high clock event
X1: 582.180 ns

Test For The "Unexpected" Cases:
For each of the respective case, all other inputs are off for each.

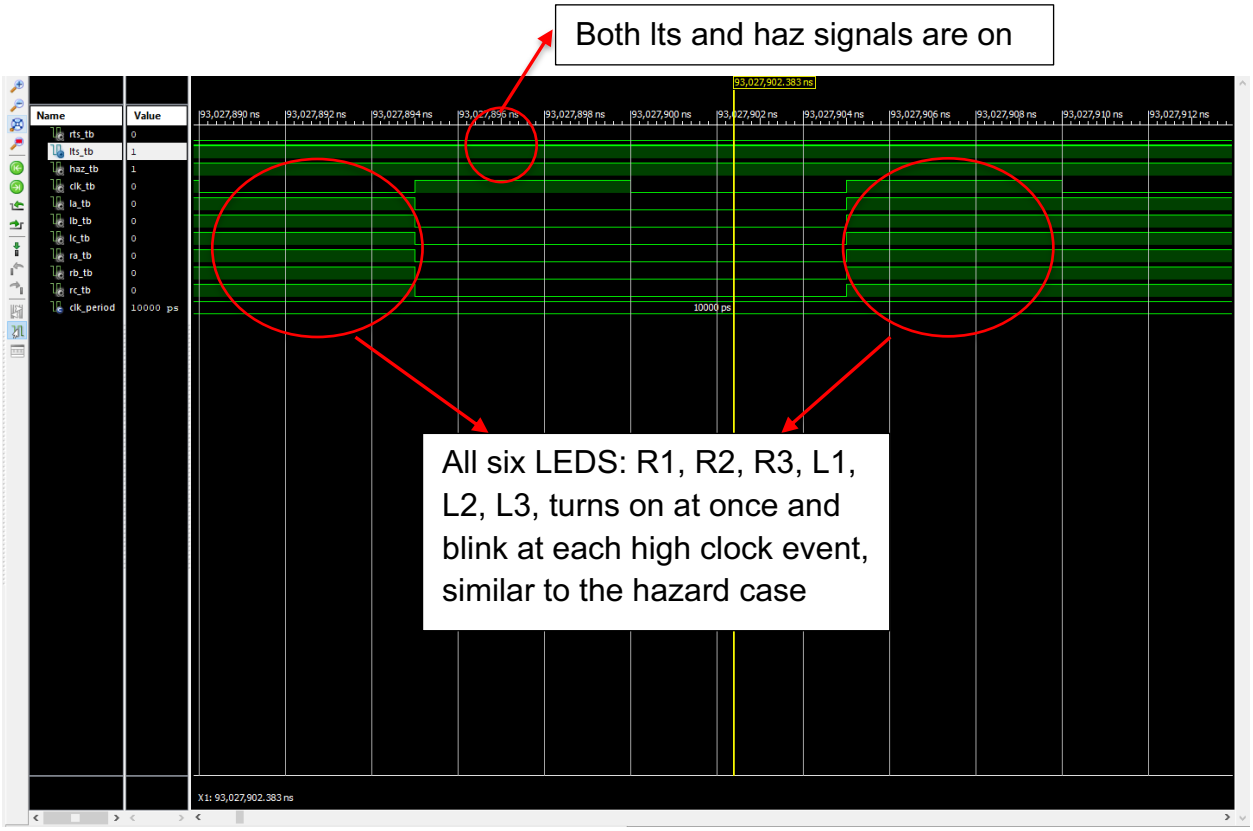
d) Rts and lts on



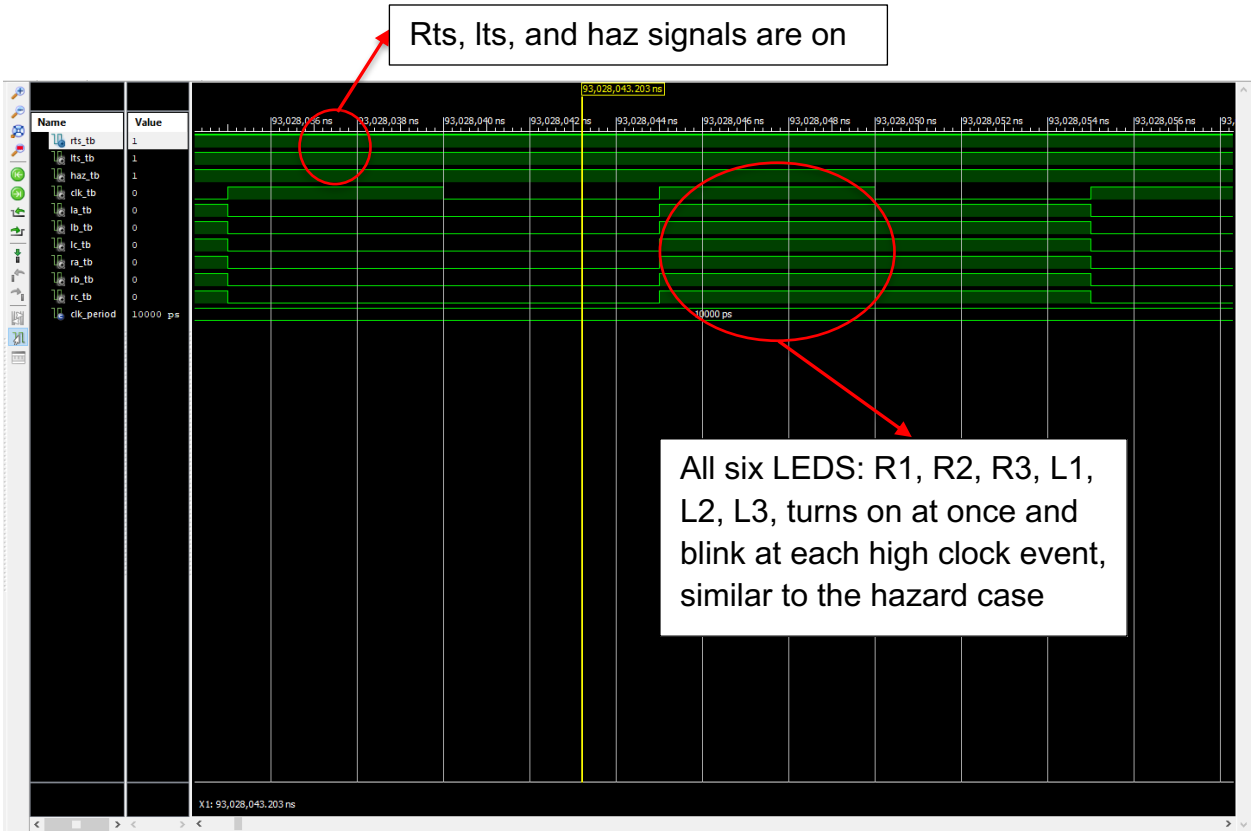
e) Rts and haz on



f) Lts and haz on

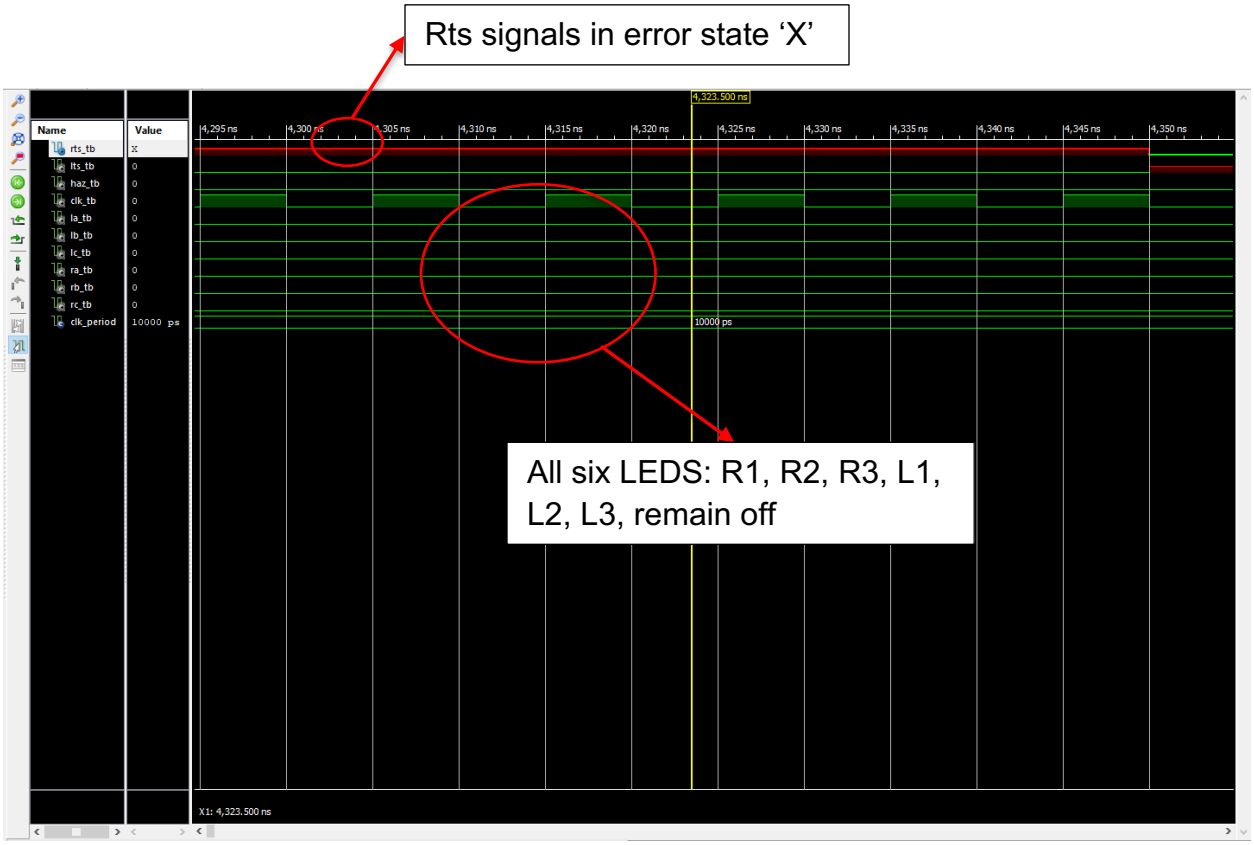


g) Rts, lts, and haz on.

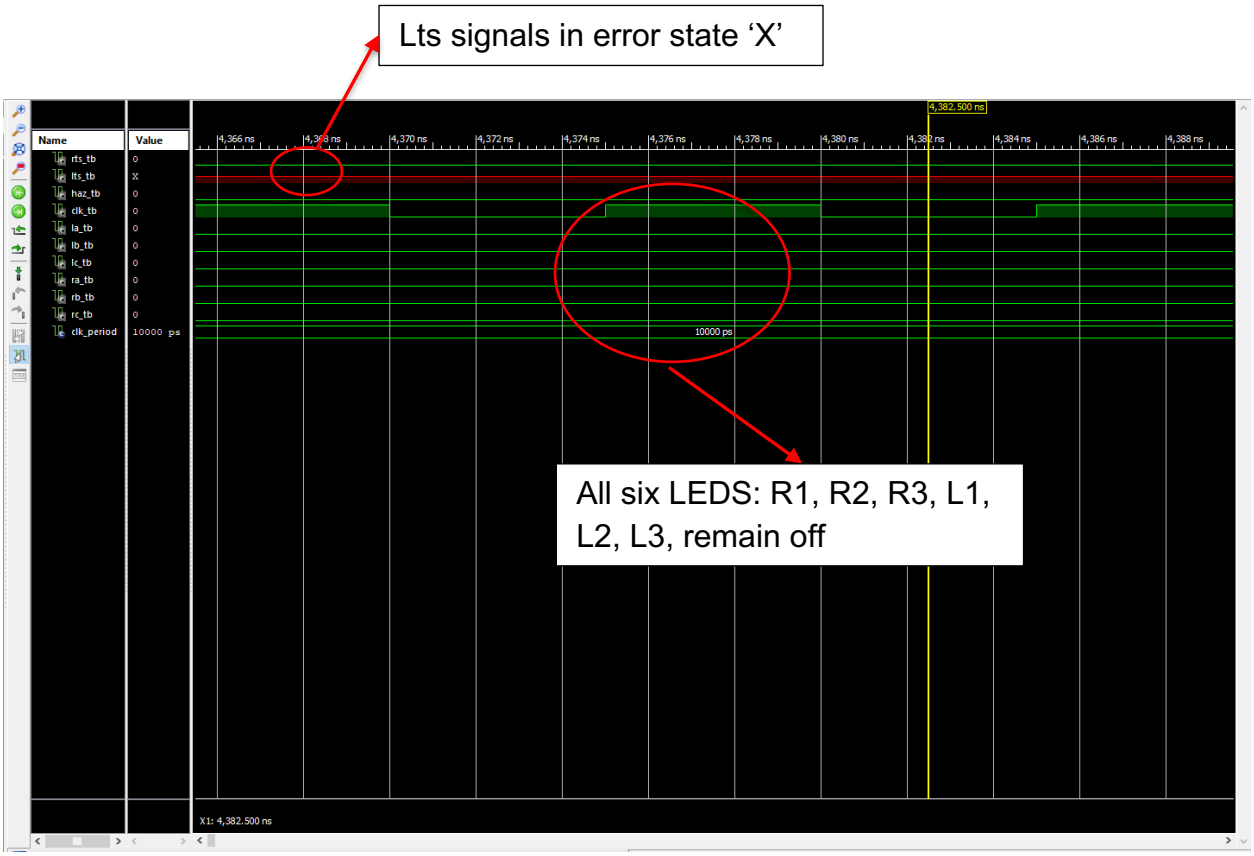


Test For Errors:
For each of the respective case, all other inputs are off for each.

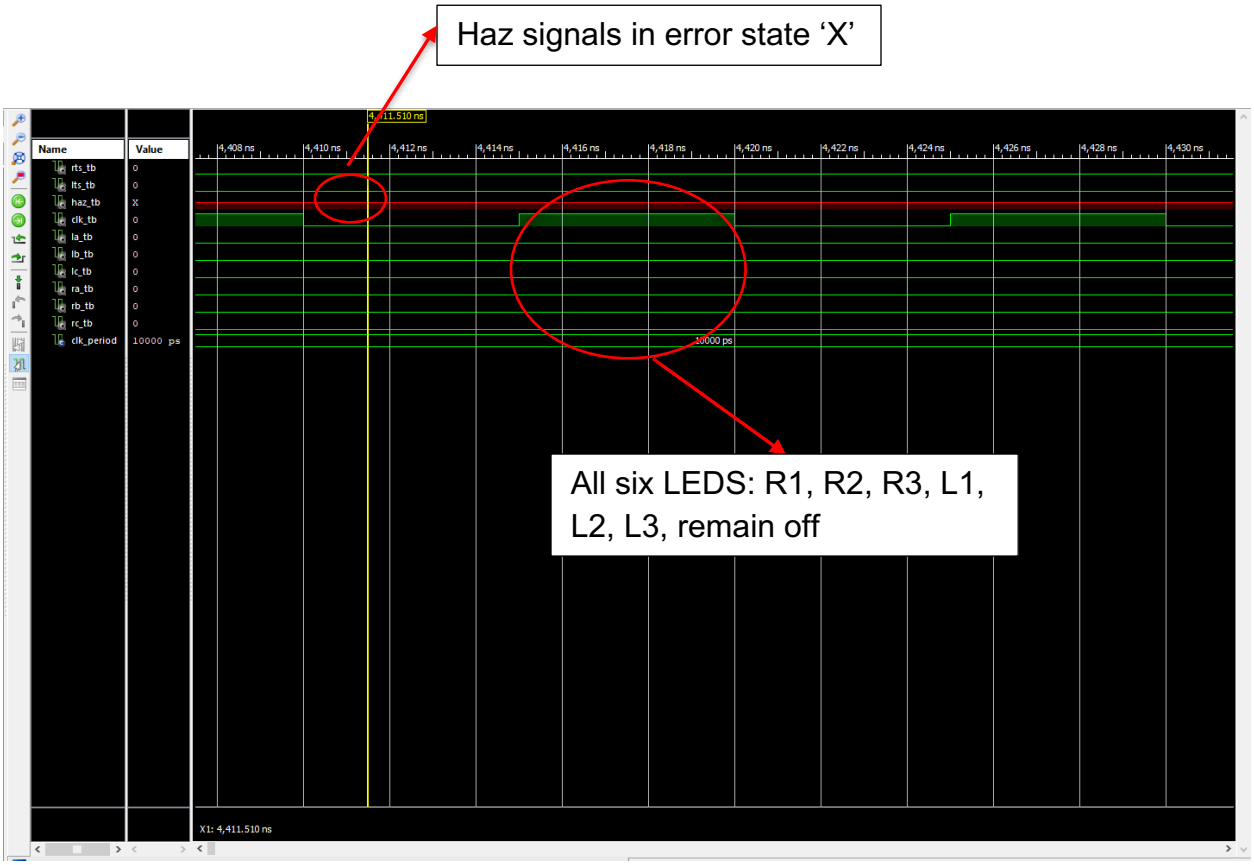
h) Rts to ‘X’,



i) Lts to ‘X’


Lts signals in error state ‘X’
Name
Value
rts_tb
lts_tb
haz_tb
clk_tb
la_tb
lb_tb
lc_tb
ra_tb
rb_tb
rc_tb
clk_period
10000 ps
4,382.500 ns
10000 ps
All six LEDS: R1, R2, R3, L1, L2, L3, remain off
X1: 4,382.500 ns

j) Haz to ‘X’.


Haz signals in error state ‘X’
Name
Value
rts_tb
lts_tb
haz_tb
clk_tb
la_tb
lb_tb
lc_tb
ra_tb
rb_tb
rc_tb
clk_period
10000 ps
4,408 ns
4,410 ns
4,412 ns
4,414 ns
4,416 ns
4,418 ns
4,420 ns
4,422 ns
4,424 ns
4,426 ns
4,428 ns
4,430 ns
All six LEDS: R1, R2, R3, L1, L2, L3, remain off
X1: 4,411.510 ns

TASK 4

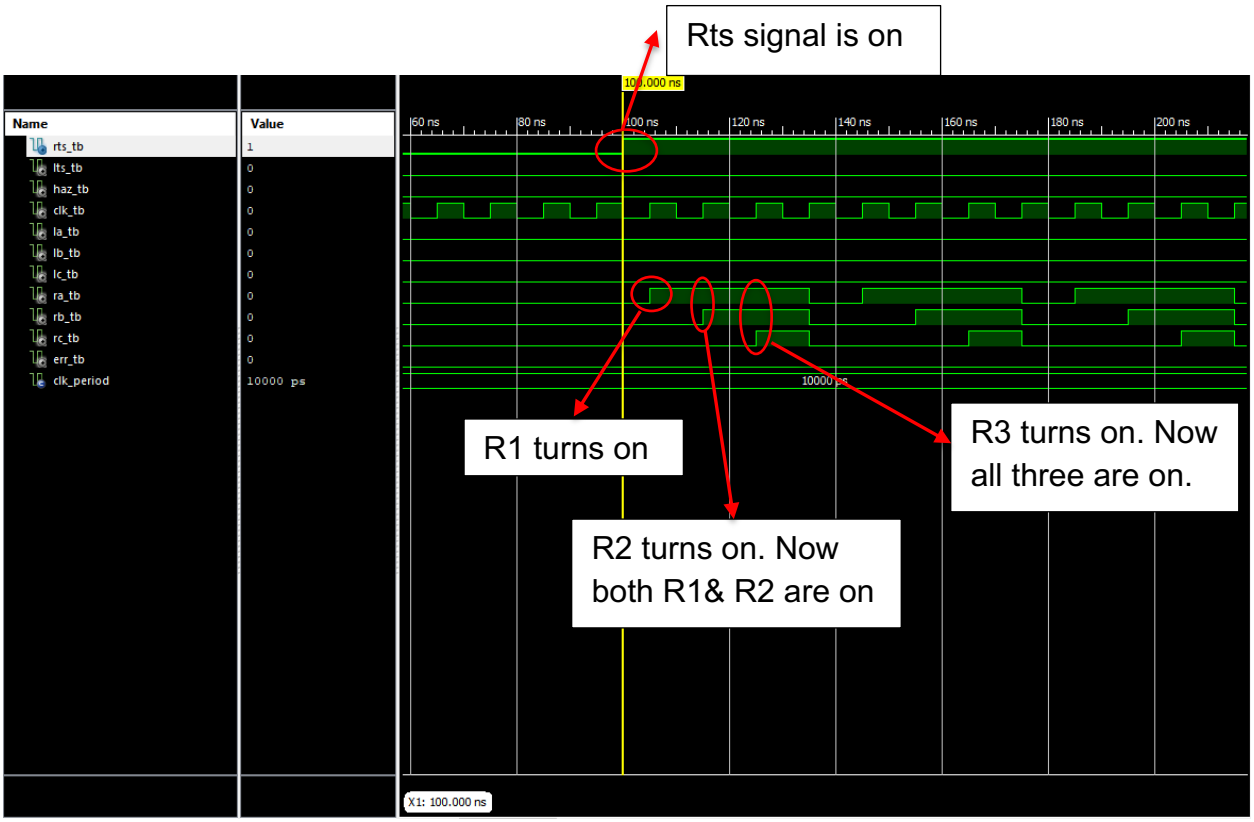
Following are the possible error cases for which the error LED must turn on:

1. **Both LTS and RTS Active in Idle Mode**: The error output should be on if, during the idle mode, both the left turn signal (LTS) and the right turn signal (RTS) are simultaneously active.
2. **LTS, RTS and Haz Active**: The error output should be on if all LTS, RTS, and Haz are simultaneously active.
3. **Error or Undefined Values on Inputs**: The error output should activate if any input (LTS, RTS, and/or hazard lights (HAZ)) receives an error/undefined ('X') value.

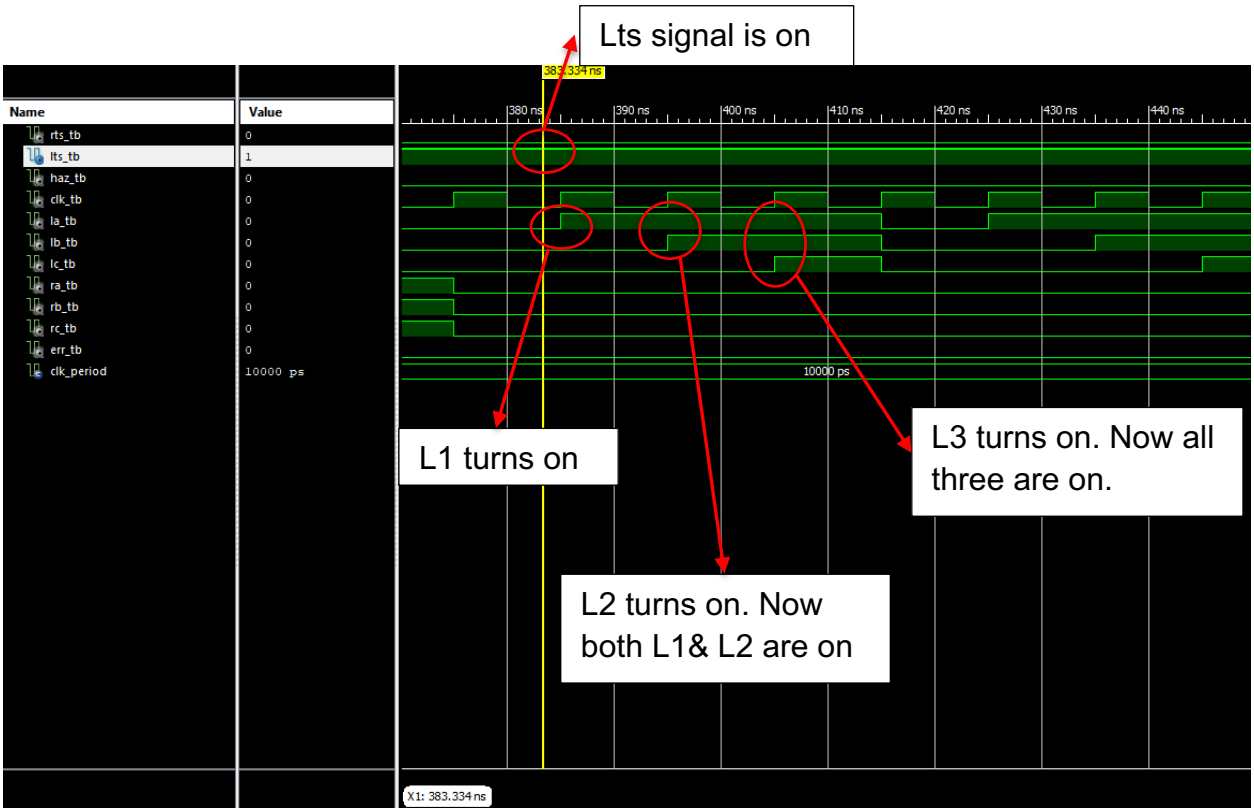
TASK 5: AFTER CORRECTION

Test For The Three Regular Cases:
For each of the respective case, all other inputs are off for each.

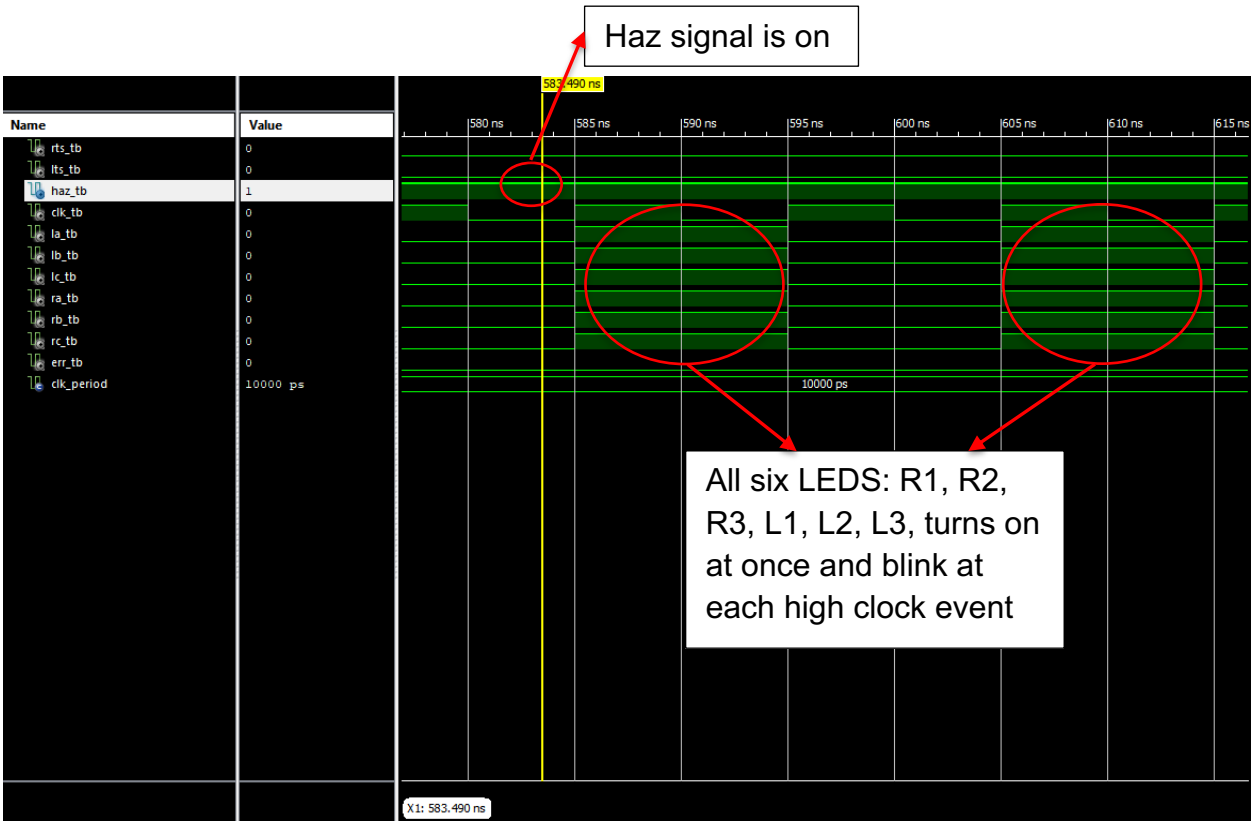
a) Only rts on



b) Only lts on

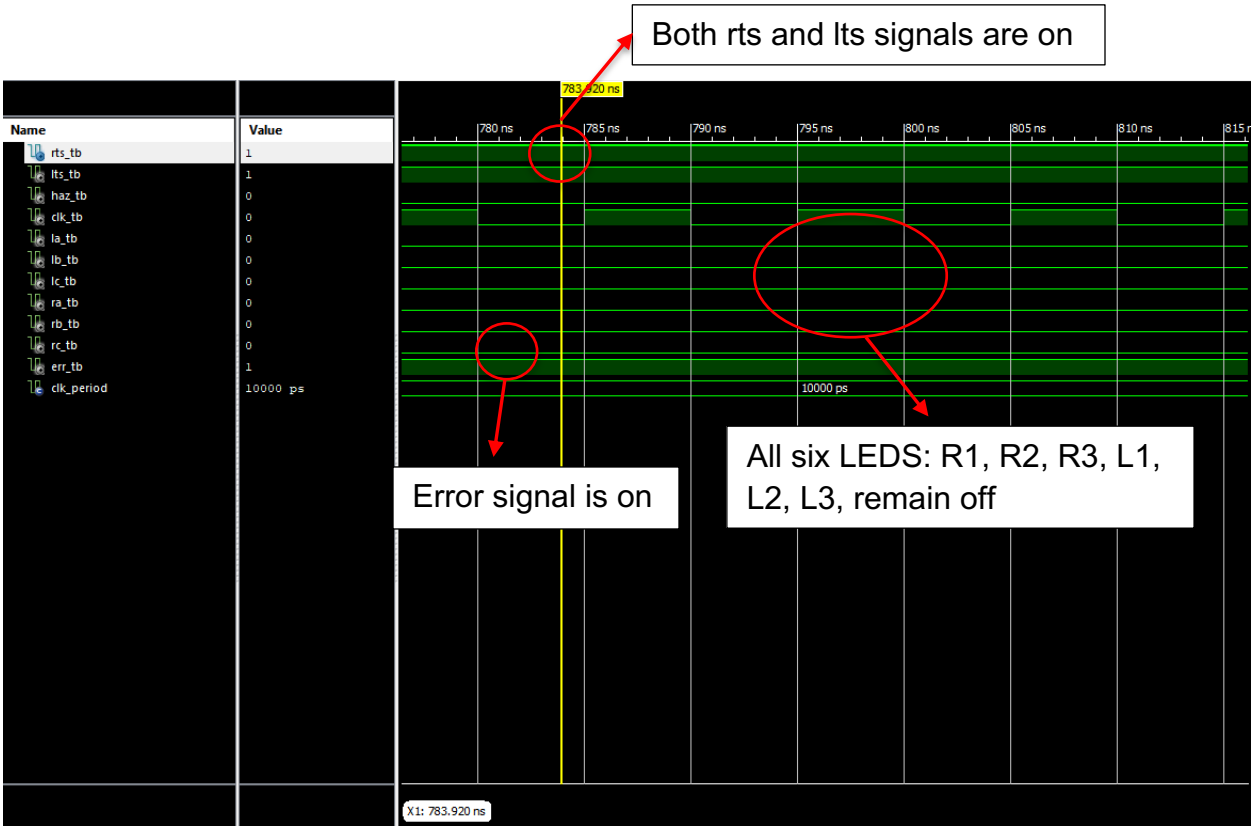

Lts signal is on
Name
Value
rts_tb
lts_tb
haz_tb
clk_tb
la_tb
lb_tb
lc_tb
ra_tb
rb_tb
rc_tb
err_tb
clk_period
0
1
0
0
0
0
0
0
0
0
0
10000 ps
380 ns
390 ns
400 ns
410 ns
420 ns
430 ns
440 ns
10000 ps
L1 turns on
L2 turns on. Now both L1& L2 are on
L3 turns on. Now all three are on.
X1: 383.334 ns

c) Only haz on

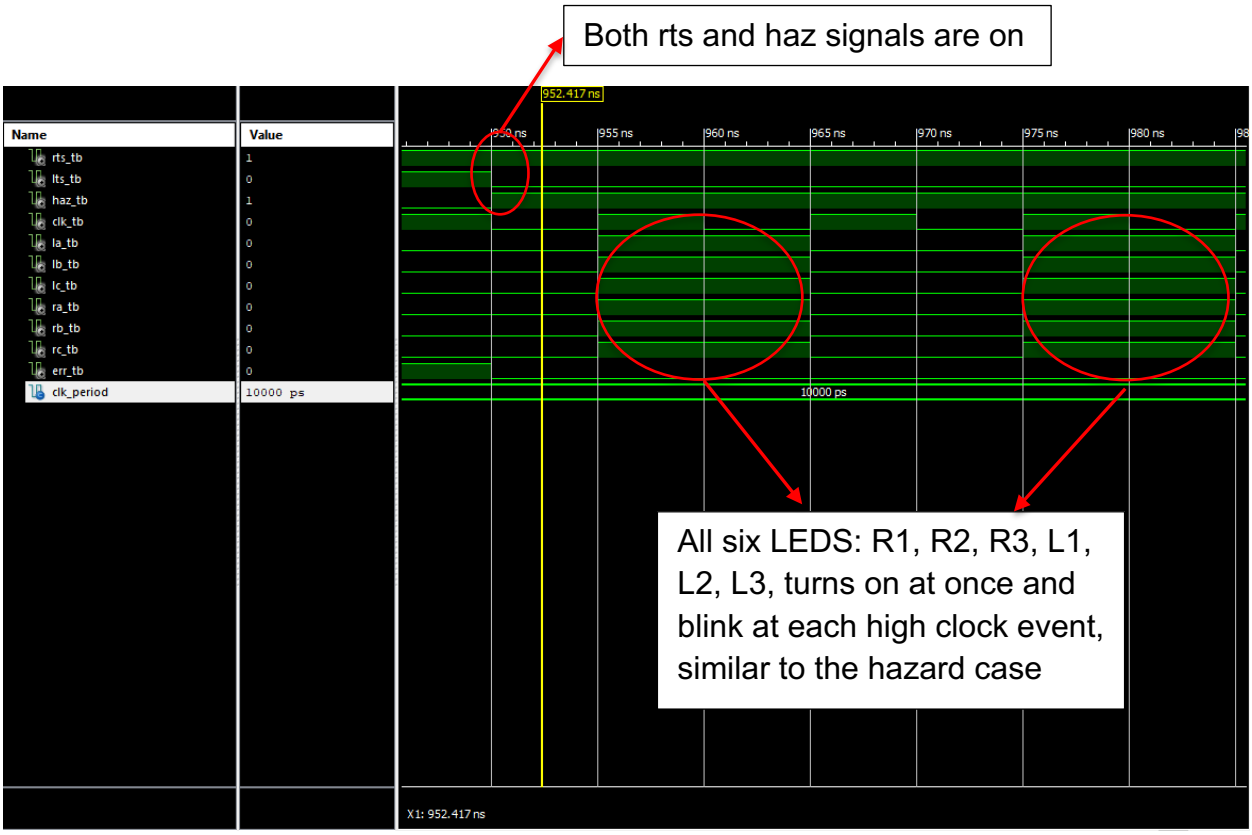

Haz signal is on
Name
Value
rts_tb
lts_tb
haz_tb
clk_tb
la_tb
lb_tb
lc_tb
ra_tb
rb_tb
rc_tb
err_tb
clk_period
0
0
1
0
0
0
0
0
0
0
0
10000 ps
580 ns
585 ns
590 ns
595 ns
600 ns
605 ns
610 ns
615 ns
10000 ps
All six LEDS: R1, R2, R3, L1, L2, L3, turns on at once and blink at each high clock event
X1: 583.490 ns

Test For The "Unexpected" Cases:
For each of the respective case, all other inputs are off for each.

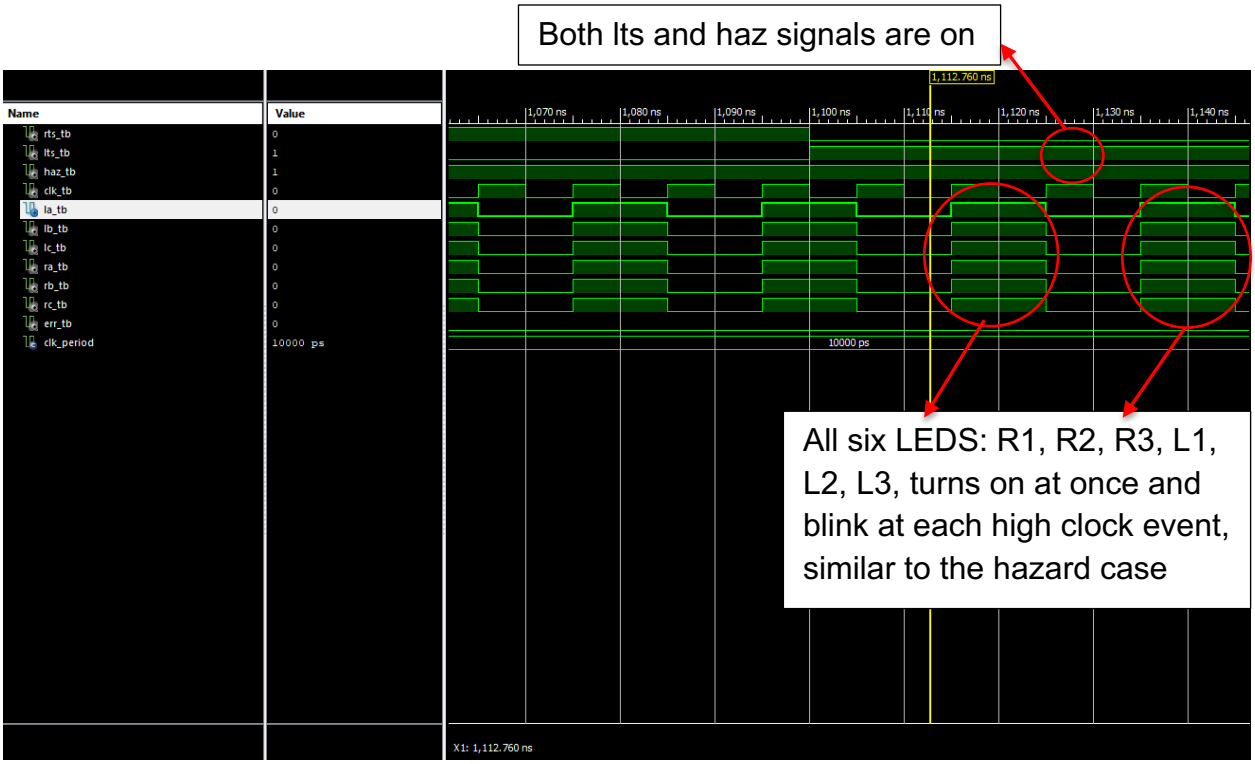
d) Rts and lts on



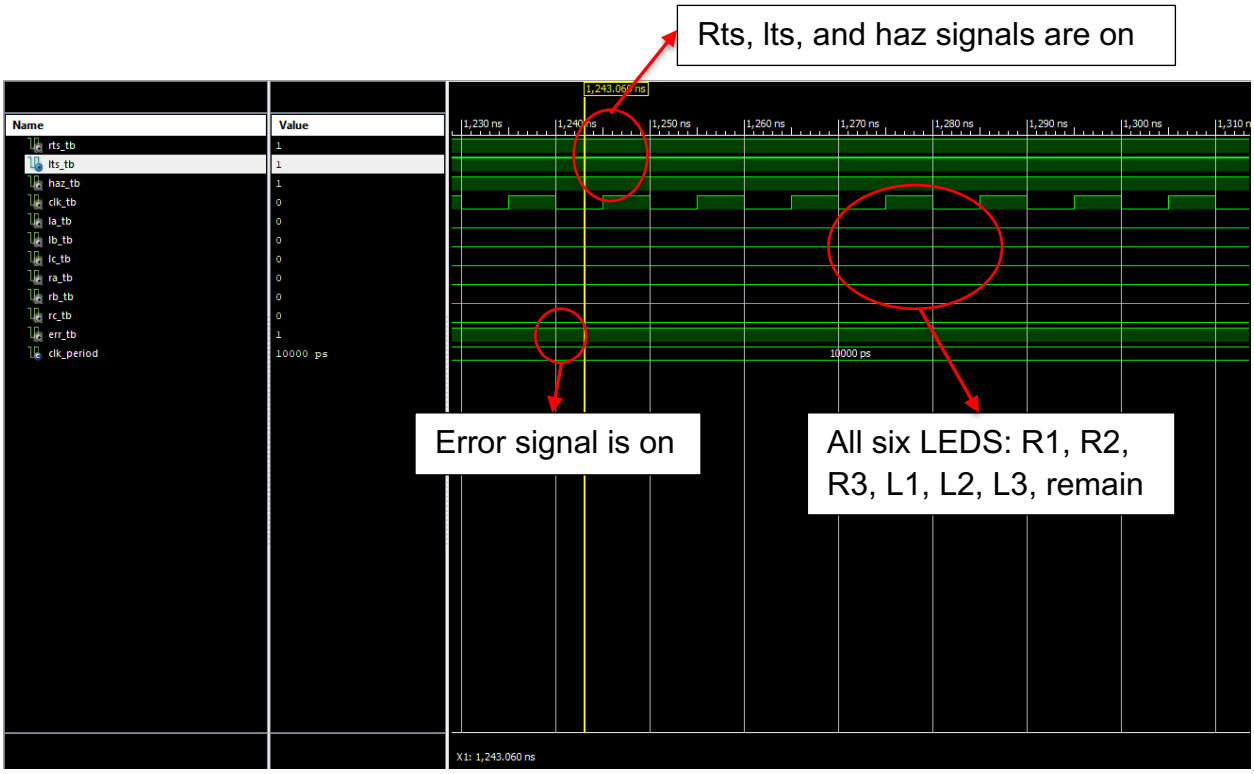
e) Rts and haz on



f) Lts and haz on

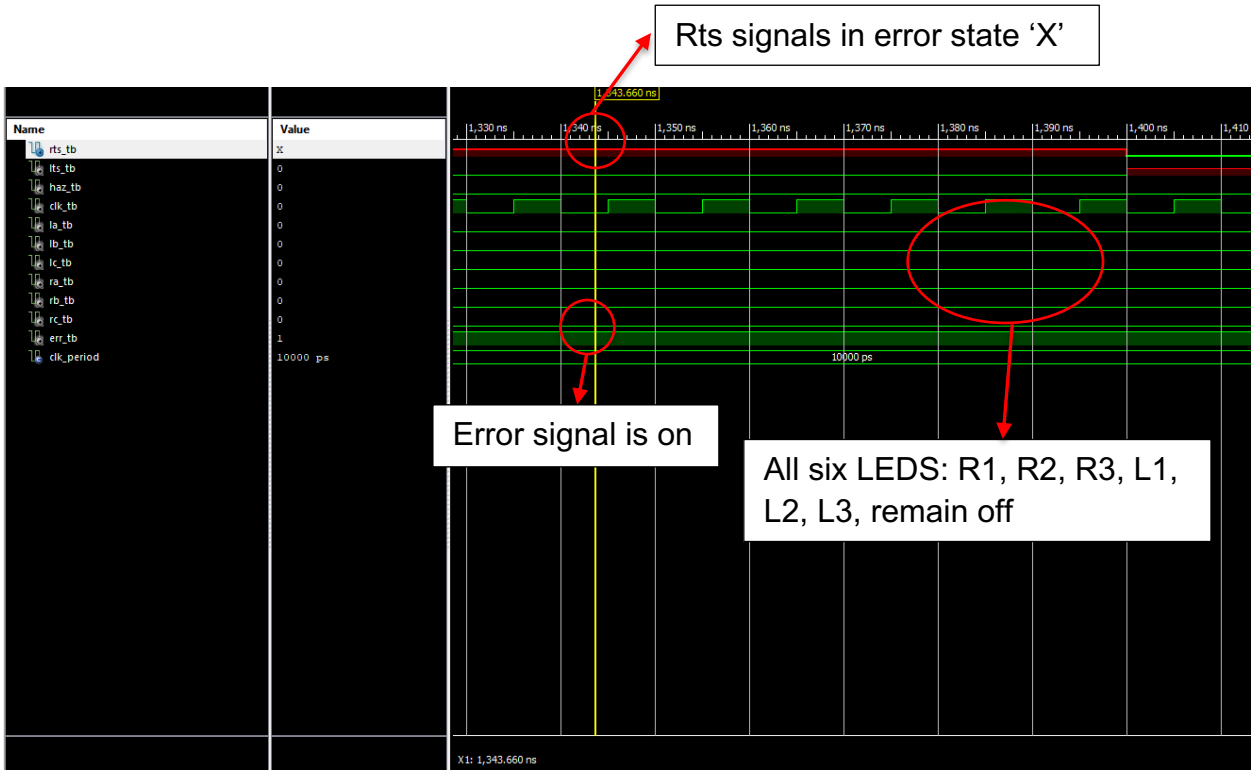


g) Rts, lts, and haz on

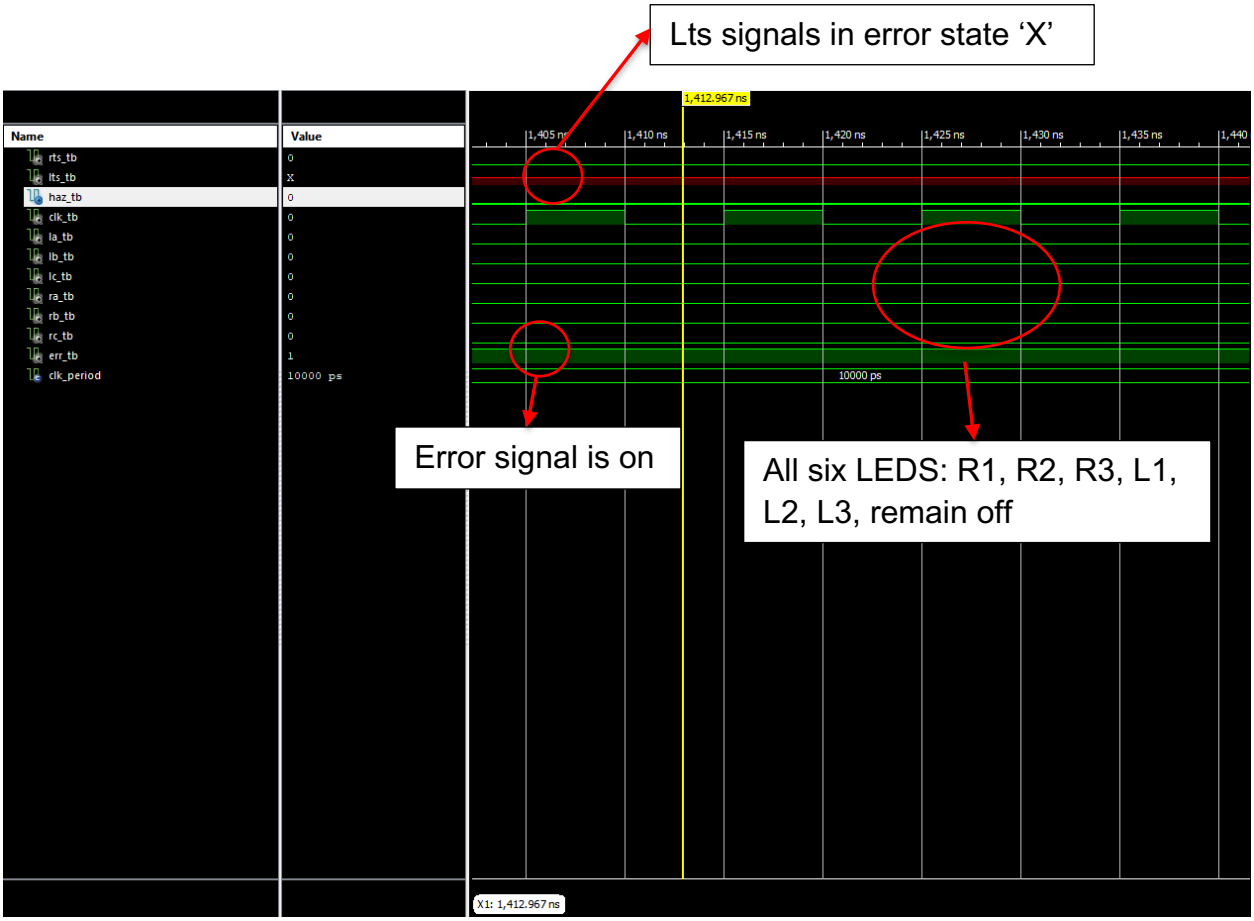


Test For Errors:
For each of the respective case, all other inputs are off for each

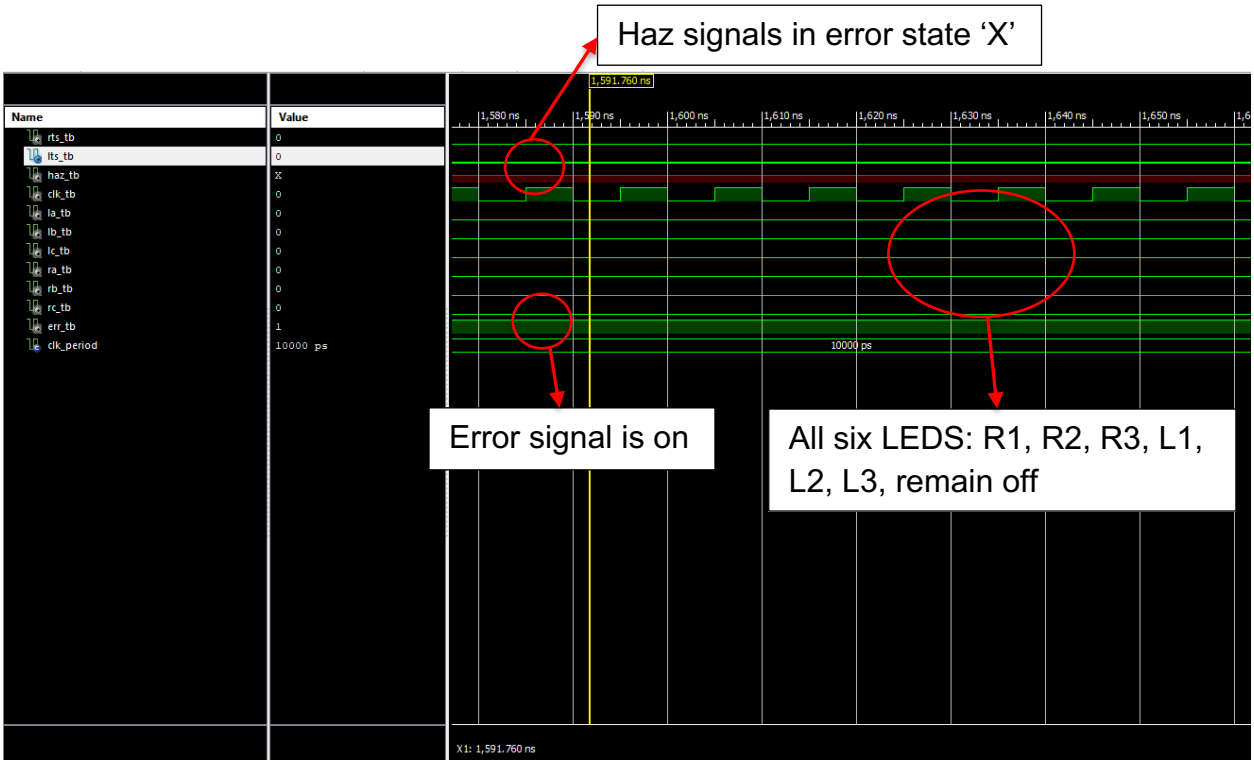
h) Rts to ‘X’



i) Lts to ‘X’

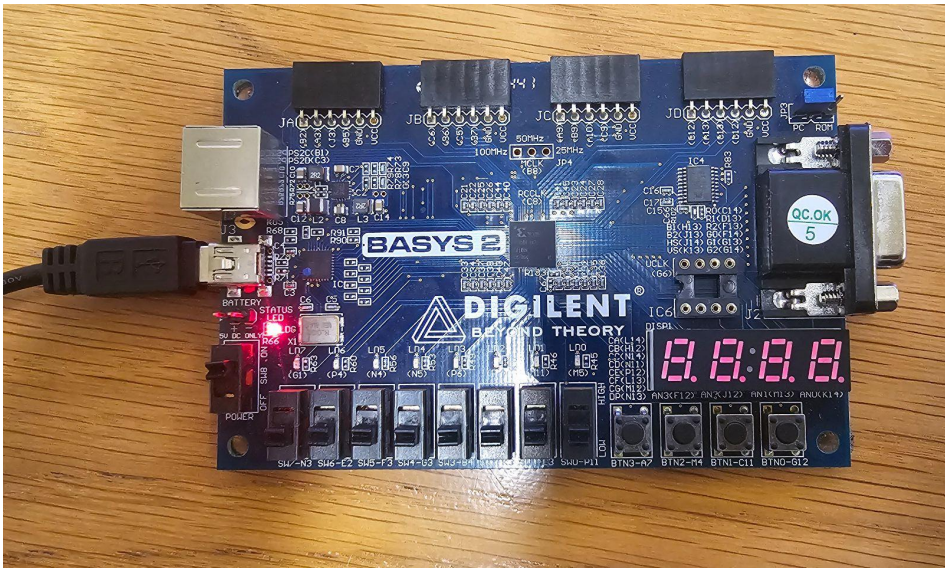

Lts signals in error state ‘X’
Name
Value
rts_tb
lts_tb
haz_tb
clk_tb
la_tb
lb_tb
lc_tb
ra_tb
rb_tb
rc_tb
err_tb
clk_period
0
X
0
0
0
0
0
0
0
0
1
10000 ps
1,412.967 ns
1,405 ns
1,410 ns
1,415 ns
1,420 ns
1,425 ns
1,430 ns
1,435 ns
1,440
10000 ps
Error signal is on
All six LEDS: R1, R2, R3, L1, L2, L3, remain off
X1: 1,412.967 ns

j) Haz to ‘X’


Haz signals in error state ‘X’
Name
Value
rts_tb
lts_tb
haz_tb
clk_tb
la_tb
lb_tb
lc_tb
ra_tb
rb_tb
rc_tb
err_tb
clk_period
0
0
X
0
0
0
0
0
0
0
1
10000 ps
1,591.760 ns
1,580 ns
1,590 ns
1,600 ns
1,610 ns
1,620 ns
1,630 ns
1,640 ns
1,650 ns
10000 ps
Error signal is on
All six LEDS: R1, R2, R3, L1, L2, L3, remain off
X1: 1,591.760 ns

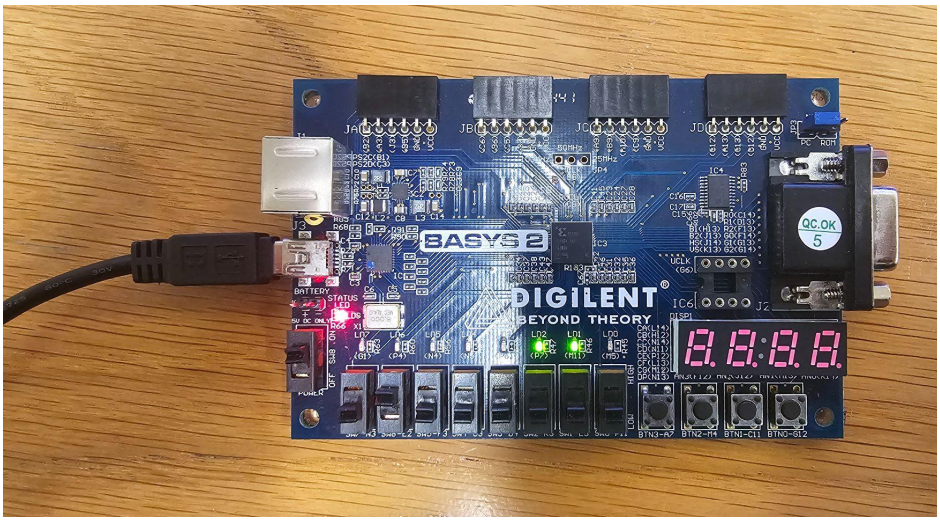
TASK 6: IMPLEMENTATION OF ERROR STATE OUTPUT AND OTHER STATES ON FPGA BOARD SNAPSHOTS

Idle State with all: lts, rts, and haz, turned off



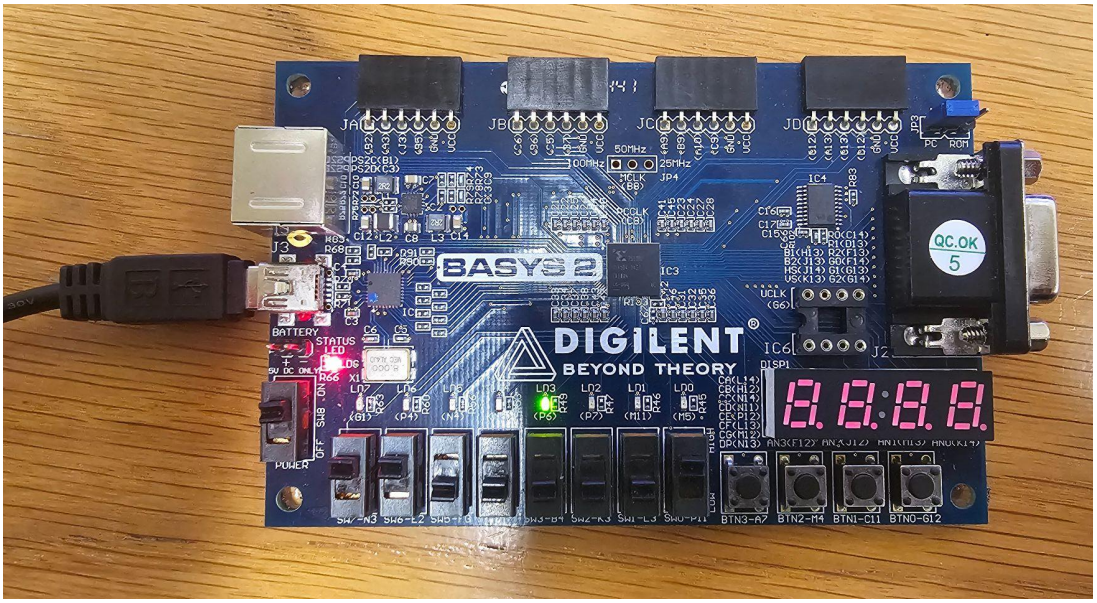
No lights are on signaling idle state.

LTS command is turned on



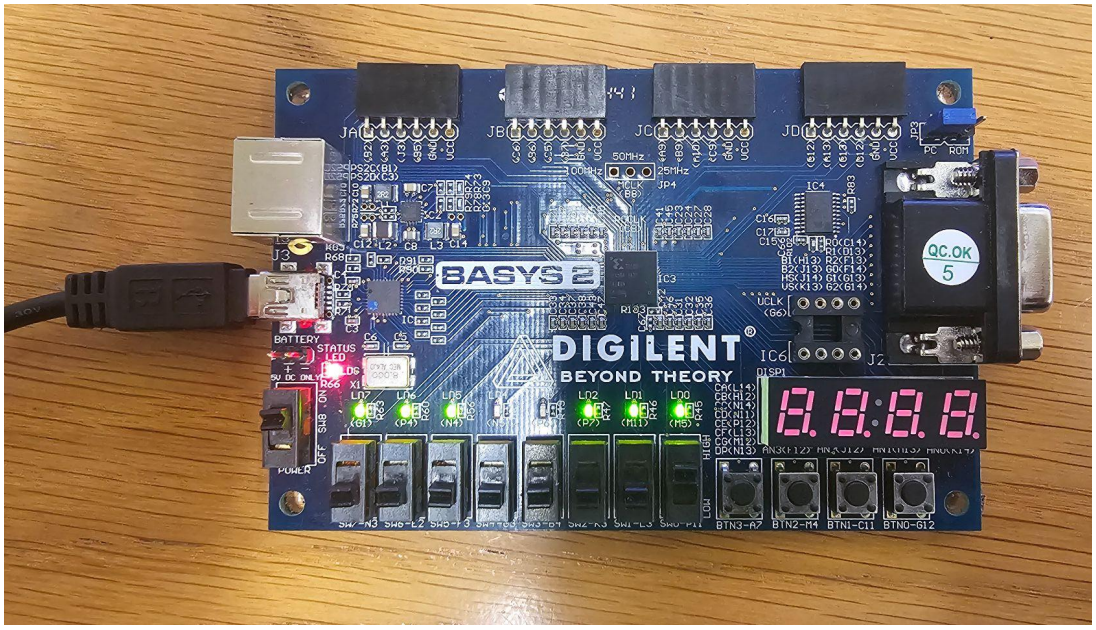
LED L1, L2, L3, turns on consecutively at each positive clock cycle.

Error State: LTS, RTS, and HAZ are on



LED P6 turns on, showing that an error state is present.

Hazard State



LEDs L1, L2, L3, R1, R2, and R3 turns on at once and blink at each positive clock cycle.

When you run the FSM on the FPGA for the 1st time, what do you observe? Take notes.

Ans: When we ran the FSM on the FPGA, we observed the LEDs to be active, but not continuously lit. That is, they should change state based on the inputs. For example, when no inputs are active (idle), all LEDs will be off. However, when we activate a turn signal or the hazard, the tail-light LEDs will light up in a predefined sequence. However, since we had not implemented a clock divider at this point, the LEDs blinked extremely fast, so fast that the intended sequential patterns for the turn signals and hazard mode were not visible to the naked eye. The FSM was driven directly by the fast system clock (50 MHz), causing the state transitions to occur rapidly. This showed the necessity of a clock divider in order to slow down the rate of blinking. Additionally, if an error condition occurs (such as both turn signals being high in idle or an undefined input), the error LED will turn on while the tail-light LEDs remain off until the error state was removed. Moreover, we did not face a scenario where the LEDs did not change states when they were expected to.

RESPONSES TO QUESTIONS

Q1. Is the FSM a Mealy machine or a Moore machine?

A1. This FSM is a **Moore machine** because the output process depends only on the current state (and in our design, error detection is handled externally) rather than directly on the inputs.

Q2. Can we declare our own custom data types in VHDL? Give an example.

A2. Yes, in VHDL, you can declare your own custom data types. For example, we declared a custom data type for the FSM states:

type state_type is (idle, l1, l2, l3, r1, r2, r3, lr3);

This declaration creates a custom enumeration type named “state_type” for the FSM states.

Q3. Explain the significance of having three different processes. Is it necessary to have exactly three processes to model FSMs? Can we use only two or one?

A3. Using three separate processes (one for the state register, one for the next-state combinational logic, and one for output generation) helps to clearly separate sequential and combinational parts of the design. It also improves modularity and clarity of the code. It is not strictly necessary to use three processes; an FSM can be modelled with one or two processes. However, separating them can make debugging and maintenance easier, and it may make the code less clear.

Q4. What is the working principle of the simple clock divider we used in this lab? Could we use the very same parameters for the clock divider for another board with a different operating frequency?

A4. A clock divider works by counting clock pulses to reduce the frequency of the input clock signal to a desired output rate. For the purpose of this lab, the clock divider counts the incoming clock cycles and toggles an enable signal (clk_en) when the count reaches a predefined value. This effectively reduces the frequency for the FSM updates (e.g., to 2 Hz or 4 Hz). No, you cannot use the same parameters directly if the board's operating frequency is different. The divider's parameters depend on the input clock frequency; therefore, if another board uses a different clock frequency, the divider's count value must be recalculated accordingly.