<u>ADLD COURSE PROJECT</u>

<u>TEAM-21</u>

Team members:

* Aditi V Angadi – 01FE19BEC212

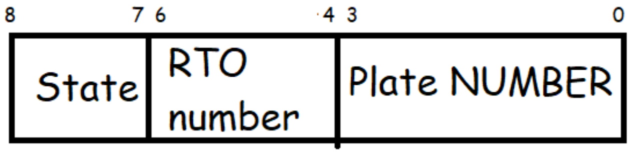
* Venkatesh Kulkarni – 01FE19BEC131

* Prathamesh Rane – 01FE19BEC294

1) Problem Statement :

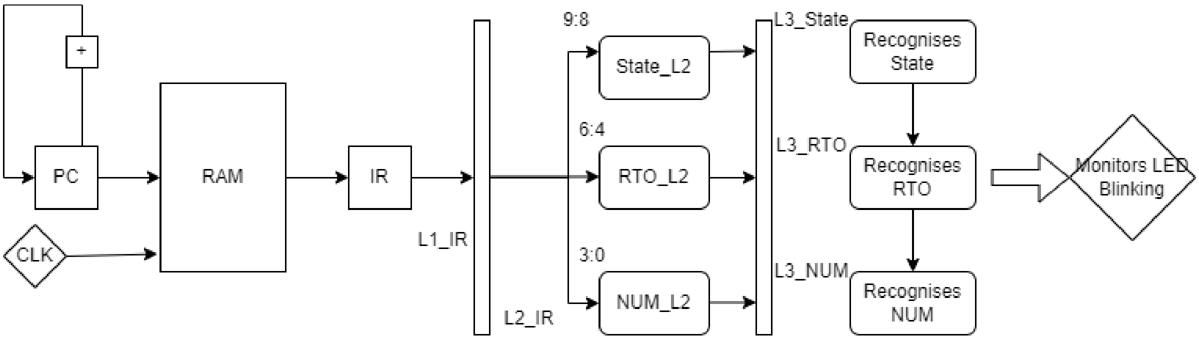
License Plate Recognition System.

A license plate is a series of letters, numbers, or a combination of both that is a registration of a vehicle's identity. Usually, a license plate is on a rectangular metal plate and is required to be both on the front and back of the vehicle.



2) *Architecture* :

We have used 3 stage pipeline architecture. This is an architecture implementation technique that allows multiple instructions to overlap in execution. The processor is organized as a number of stages that allow multiple instructions to be in various stages of their instruction cycle.



3) Algorithm :

- Here we are taking Number plate to be of 9 bits i.e., 4 MSBs represent plate number(0-3) , next 3 bits represent RTO number(4-6) , next 2 bits represent State(7-8). Total vehicles = 512.
- State : KA, GJ, BR, MH
- RTO number : rto1, rto2, rto3, rto4, rto5, rto6, rto7, rto8.
- Plate number : person1, person2, person3, person4, person5, person6, person7, person8, person9, person10, person11, person12, person13, person14, person15, person16.
- Memory has 4 number plate info stored.

a) Initially in memory program counter(pc=0).

b) Step 1 : In pipeline stage 1, L1_IR instruction fetches the mem[pc] then it increments pc.

c) Step 2 : In pipeline stage 2, number plate info is decoded to L2_State, L2_RTO, L2_NUM from L2_IR.

d) Step 3 : In pipeline stage 3, execution takes place through L3_State, L3_RTO, L3_NUM.

e) Then L3_State is checked by case statement if it's KA then it checks the RTO. Then it checks with plate number.

f) Same checking happens with other states and other RTOs.

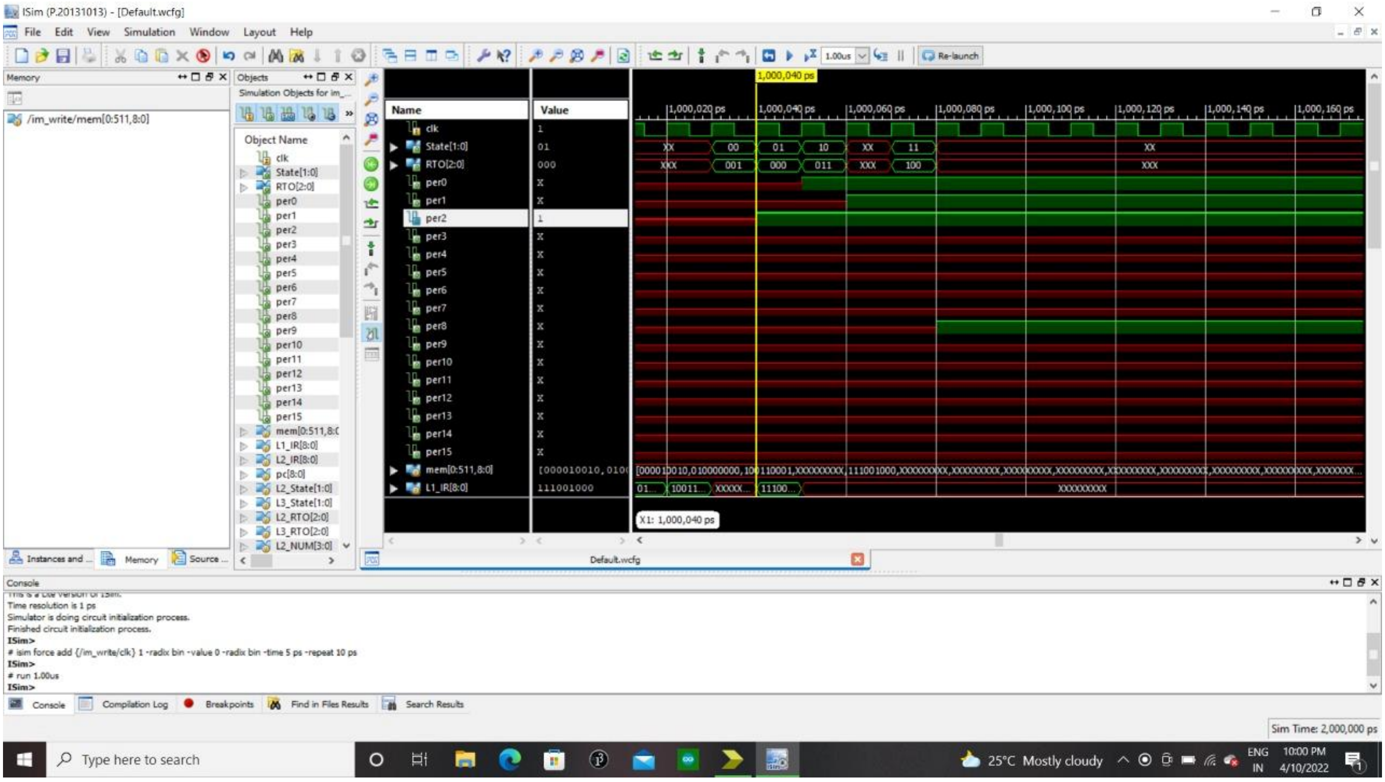
g) Output is Monitored by LEDs. Combination of LEDs is blinked based on the output (License Plate Recognition) generated.

4) *Simulation* :

Waveform of the system is seen as shown below when obtained from Xilinx ISim Simulator .

Here we can also see that after 3 clock cycles output execution starts which indicates that we are going through 3- stage pipelined architecture.

For the first mem[0] = 9’b000010010 , we can see that outputs indicates that this vehicle is from {KA RTO2 person3} i.e., State = {00} indicating State KA and RTO = {001} indicating RTO2 .

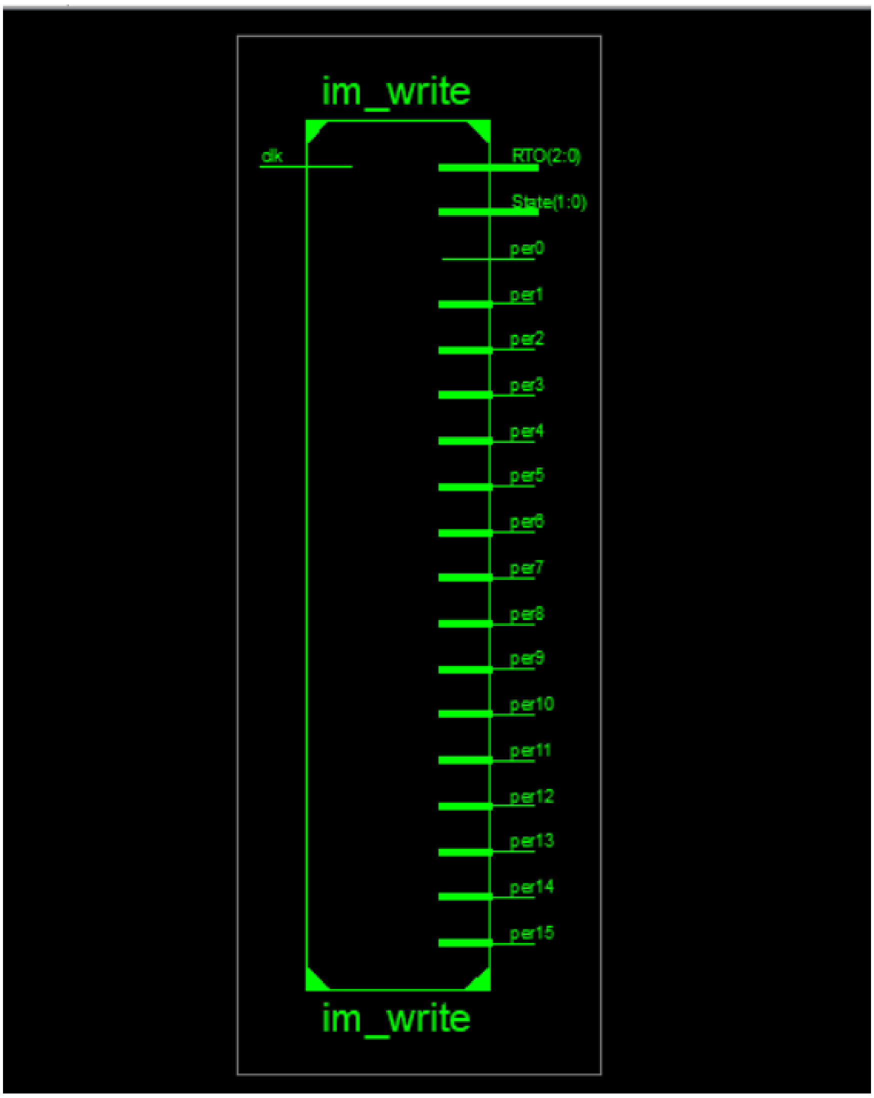


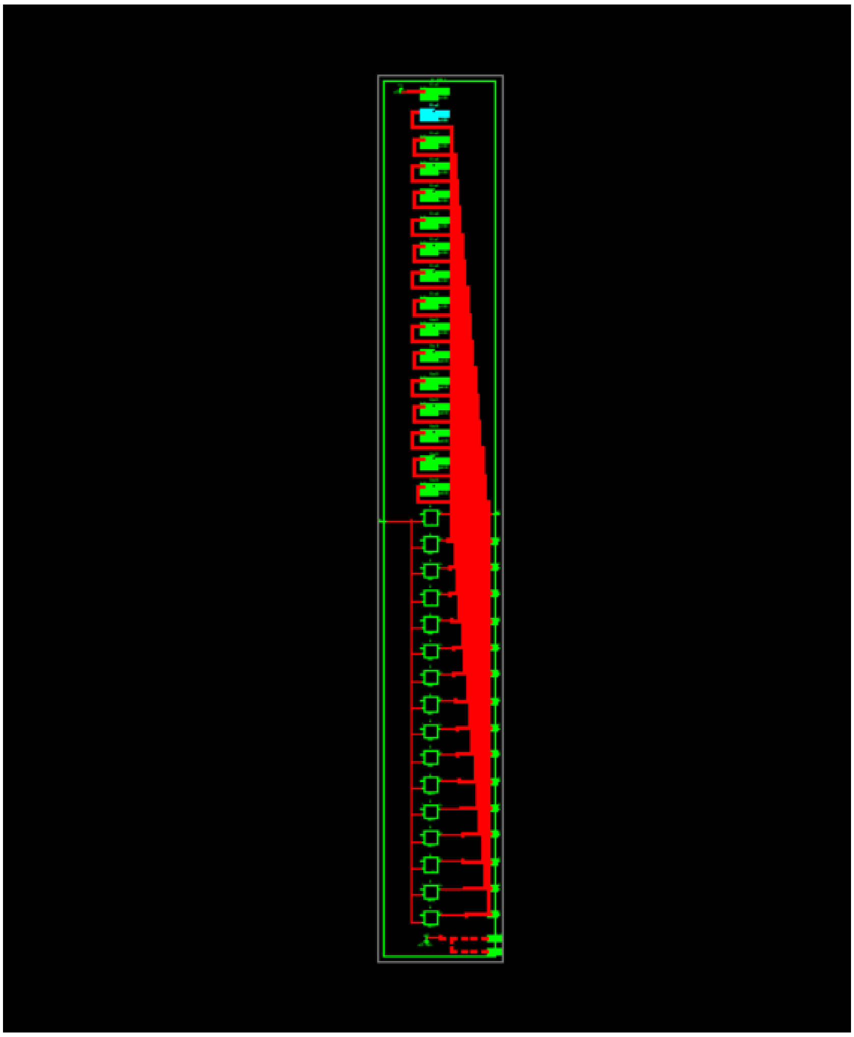
Name	Value
clk	1
State[1:0]	01
RTO[2:0]	000
per0	x
per1	x
per2	1
per3	x
per4	x
per5	x
per6	x
per7	x
per8	x
per9	x
per10	x
per11	x
per12	x
per13	x
per14	x
per15	x
mem[0:511,8:0]	[000010010,010
L1_IR[8:0]	111001000

```
This is a Lite version of ISim.
Time resolution is 1 ps
Simulator is doing circuit initialization process.
Finished circuit initialization process.
ISim>
# isim force add {/im_write/clk} 1 -radix bin -value 0 -radix bin -time 5 ps -repeat 10 ps
ISim>
# run 1.00us
ISim>
```

5) *Synthesis* :

Below is the RTL Schematic obtained when we synthesize the Verilog code in Xilinx Software





6) HDL Synthesis Report :-

```
Macro Statistics
# Registers                           : 24
1-bit register                       : 16
2-bit register                       : 2
3-bit register                       : 2
4-bit register                       : 2
9-bit register                       : 2
# Multiplexers                        : 16
1-bit 4-to-1 multiplexer             : 16
```