****

School

of

Electronics and Communication Engineering

ADLD COURSE PROJECT

**TEAM-14**

**Team members:**

1. Veeresh Akki – 01FE19BEC140

2. Rohit Upari – 01FE19BEC222

**Guide By:**

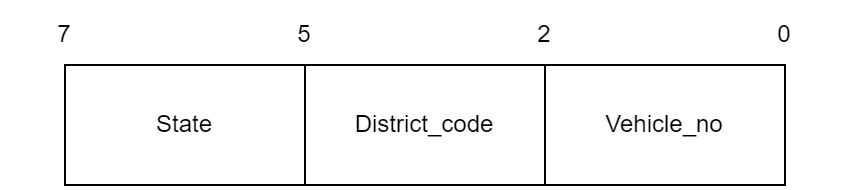
Dr. Saroja V. Siddamal

Semester: VI,2023

1. Problem Statement :

Verilog license plate recognition on FPGA

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.

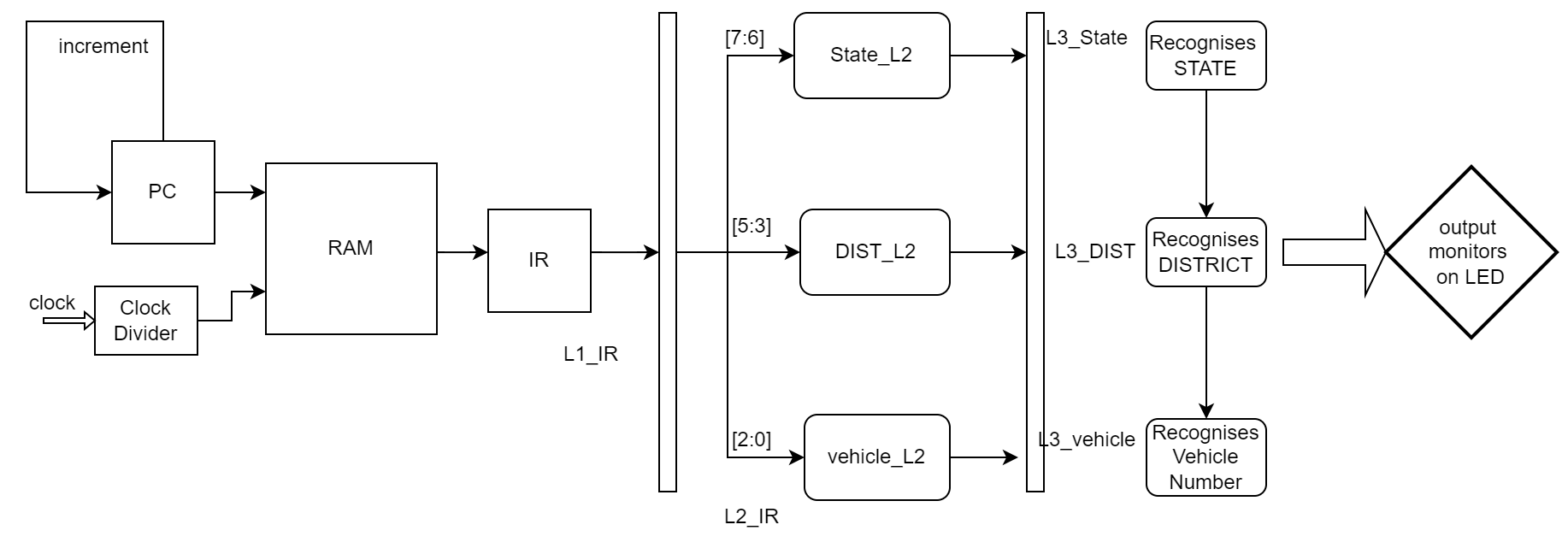


2) Architecture :

For our project implementation 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) Theory:

In license plate recognition on FPGA Direct character entry of the number plate number into the FPGA normally occurs through a UART or other communication link. When determining whether the input string corresponds to a recognised number plate number, the FPGA then employs pattern recognition techniques.

Using pattern matching methods is one way to accomplish licence plate recognition on an FPGA. In this method, a database of recognised licence plate numbers is stored on the FPGA as strings of characters. Using string comparison methods like Levenshtein distance or dynamic time warping, the input number plate number is compared to the known number plate numbers. The FPGA outputs a detect signal if a match is made, indicating that the licence plate has been identified.

Using regular expressions is a different strategy for doing licence plate recognition on an FPGA. With this method, a set of regular expressions that match well-known number plate number formats, such as national or state-specific formats, are stored in the FPGA. Regular expression matching algorithms compare the input number plate number to the regular expressions. The FPGA outputs a detect signal if a match is made, indicating that the licence plate has been identified.

The implementation of licence plate recognition on an FPGA without the use of an image as input necessitates rigorous algorithm optimisation and effective utilisation of the FPGA's resources. The database of known licence plate numbers needs to be handled carefully in order to ensure that it can be saved and accessed effectively, and the pattern recognition algorithms need to be optimised in order to run quickly on the FPGA.

4) Algorithm :

Here we are taking Number plate to be of 8 bits i.e., 3 LSBs represent plate number(0-2), next 3 bits represent district number(3-5), next 2 bits represent State(6-7). Total vehicles = 256.

- State : KA, GJ, BR, MH

- District number : dist1, dist2, dist3, dist4, dist5, dist6, dist7, dist8.

- Plate number : vehicle1, vehicle2, vehicle3, vehicle4, vehicle5, vehicle6, vehicle7, vehicle8.

- Memory has 8 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.

h) A detect signal is also used. If any input vehicle number matches the number present in the memory

then detect signal goes high.

4) Results:

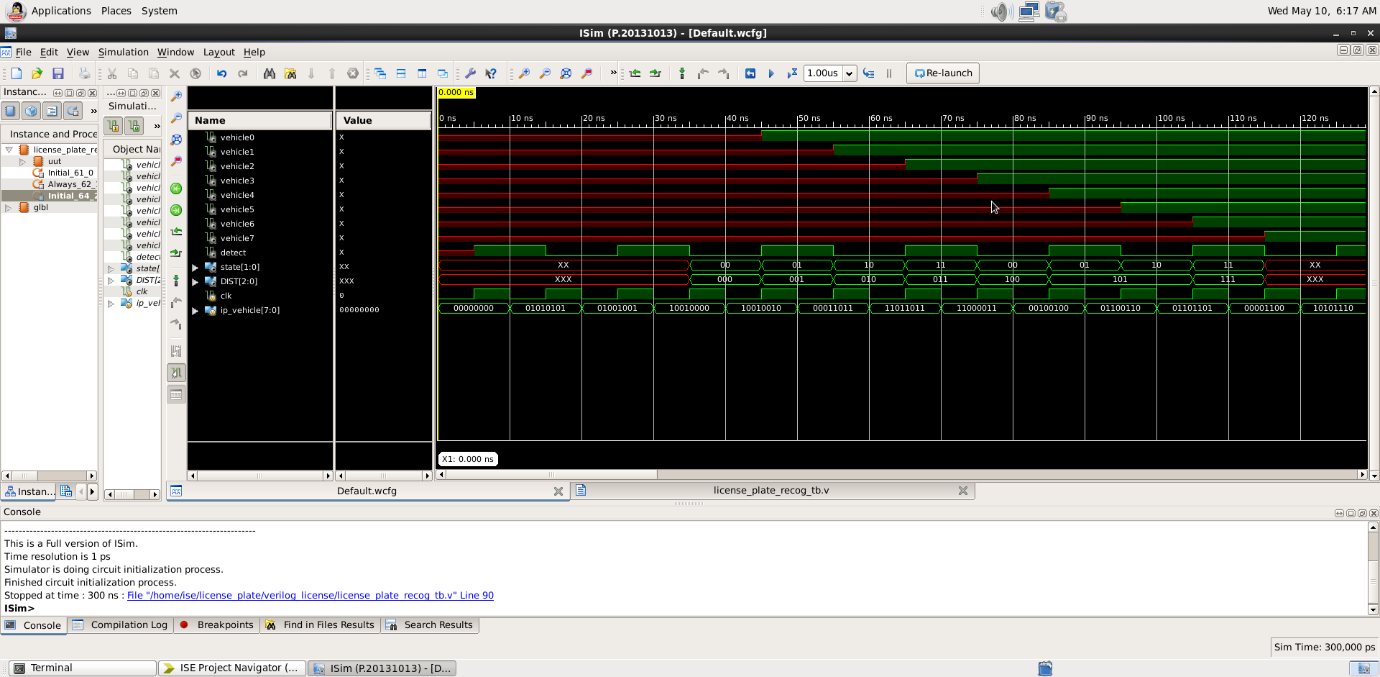
Simulation

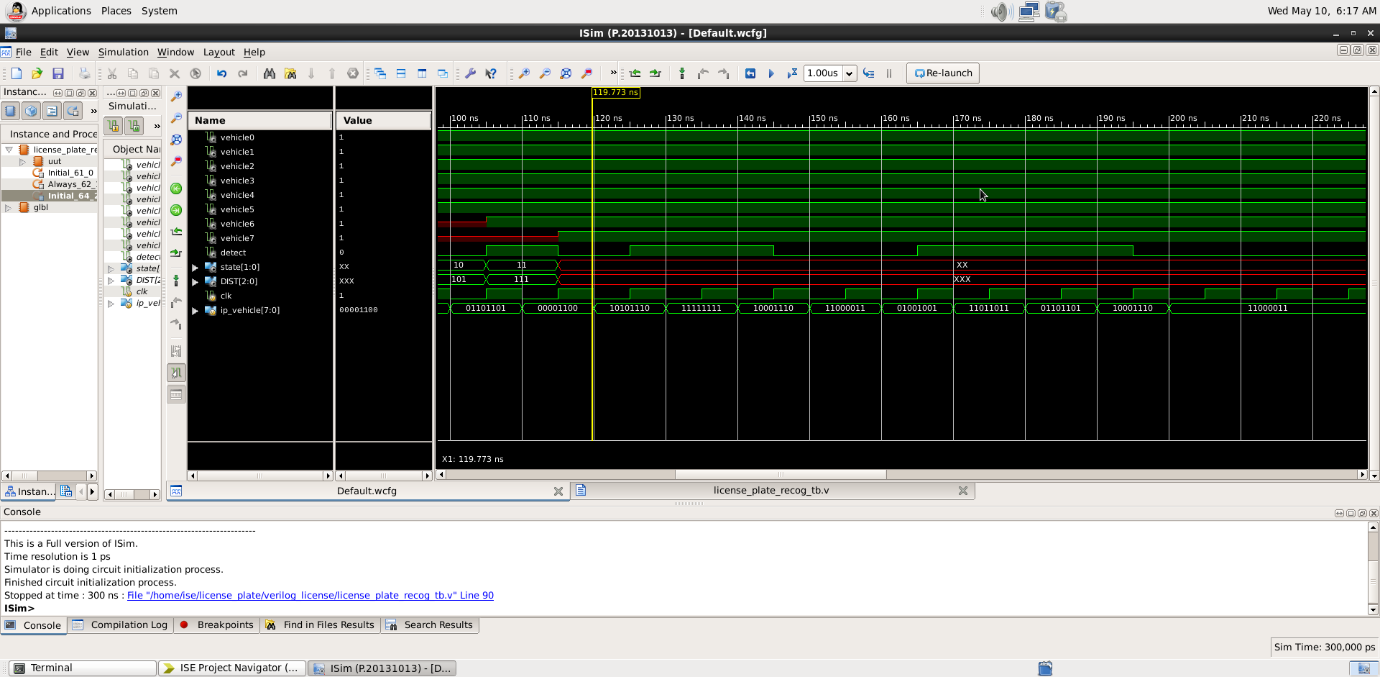
Waveform of the system is seen as shown below when obtained from Xilinx ISim Simulator .

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] = 8’b00000000 , we can see that outputs indicates that this vehicle is from {KA dist1 vehicle1} i.e., State = {00} indicating State KA and DIST = {000} indicating dist1.

If any input vehicle number matches the vehicle number in the memory then detect signal goes high.

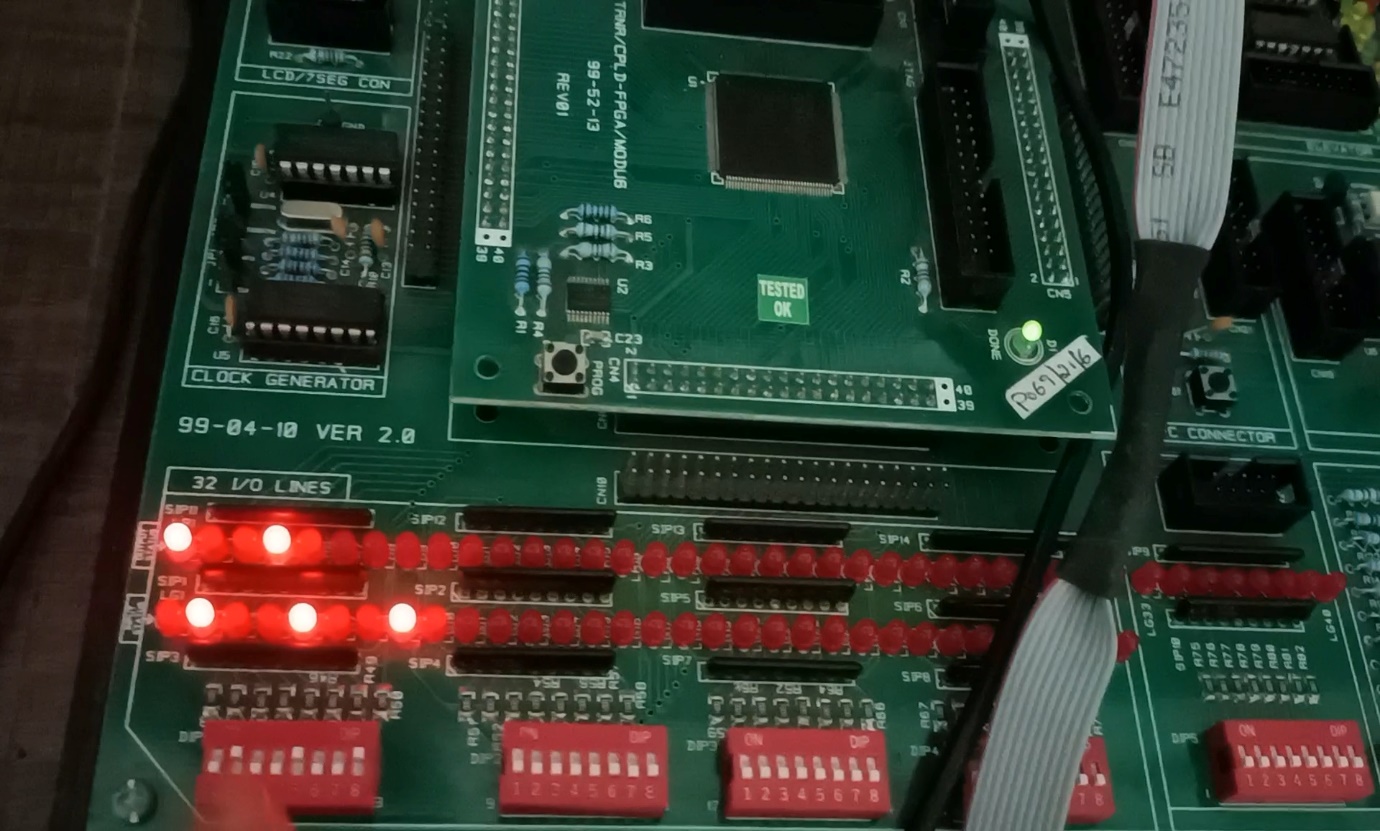


**

5) Implementation on FPGA:



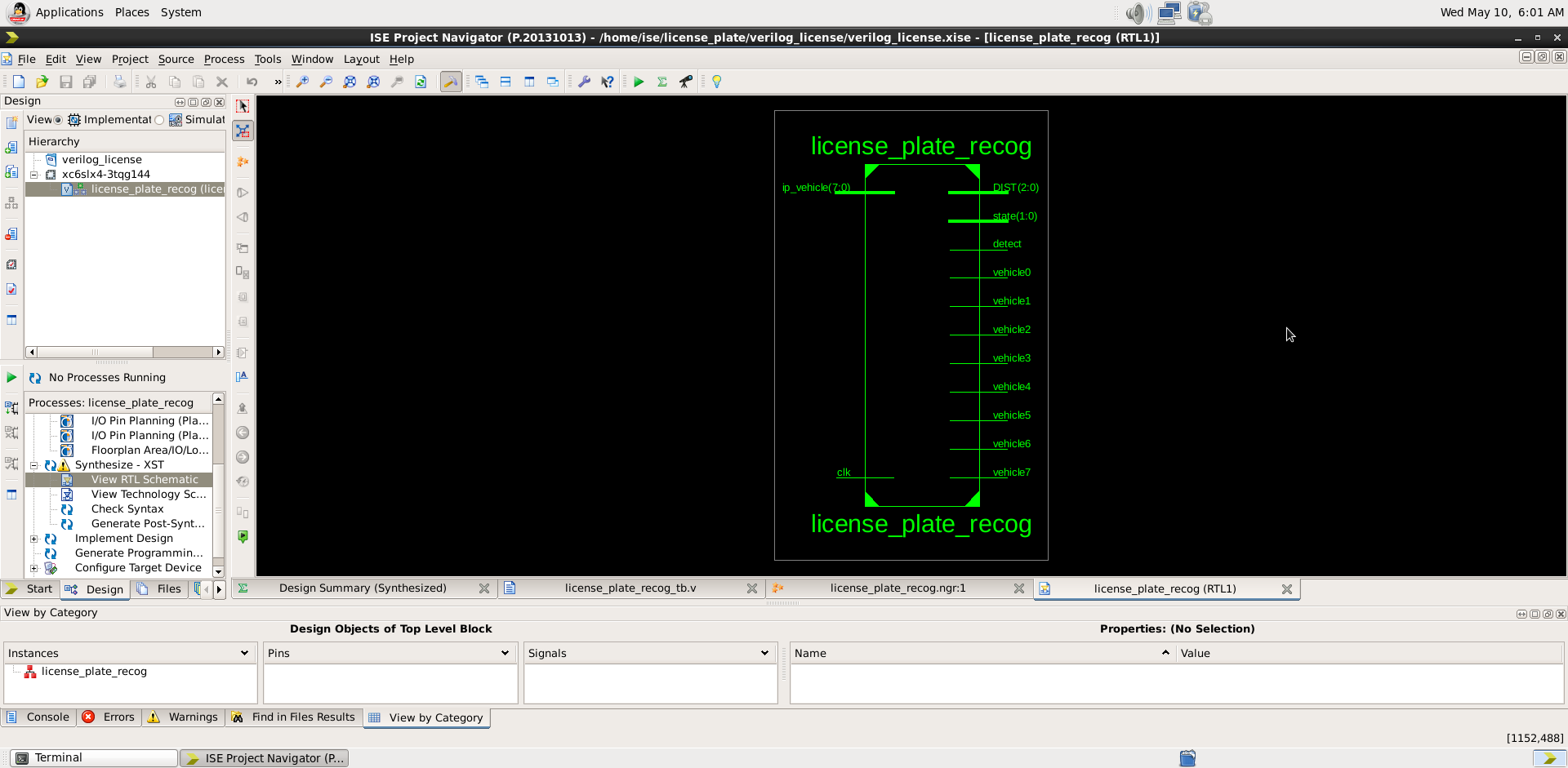
Detect signal

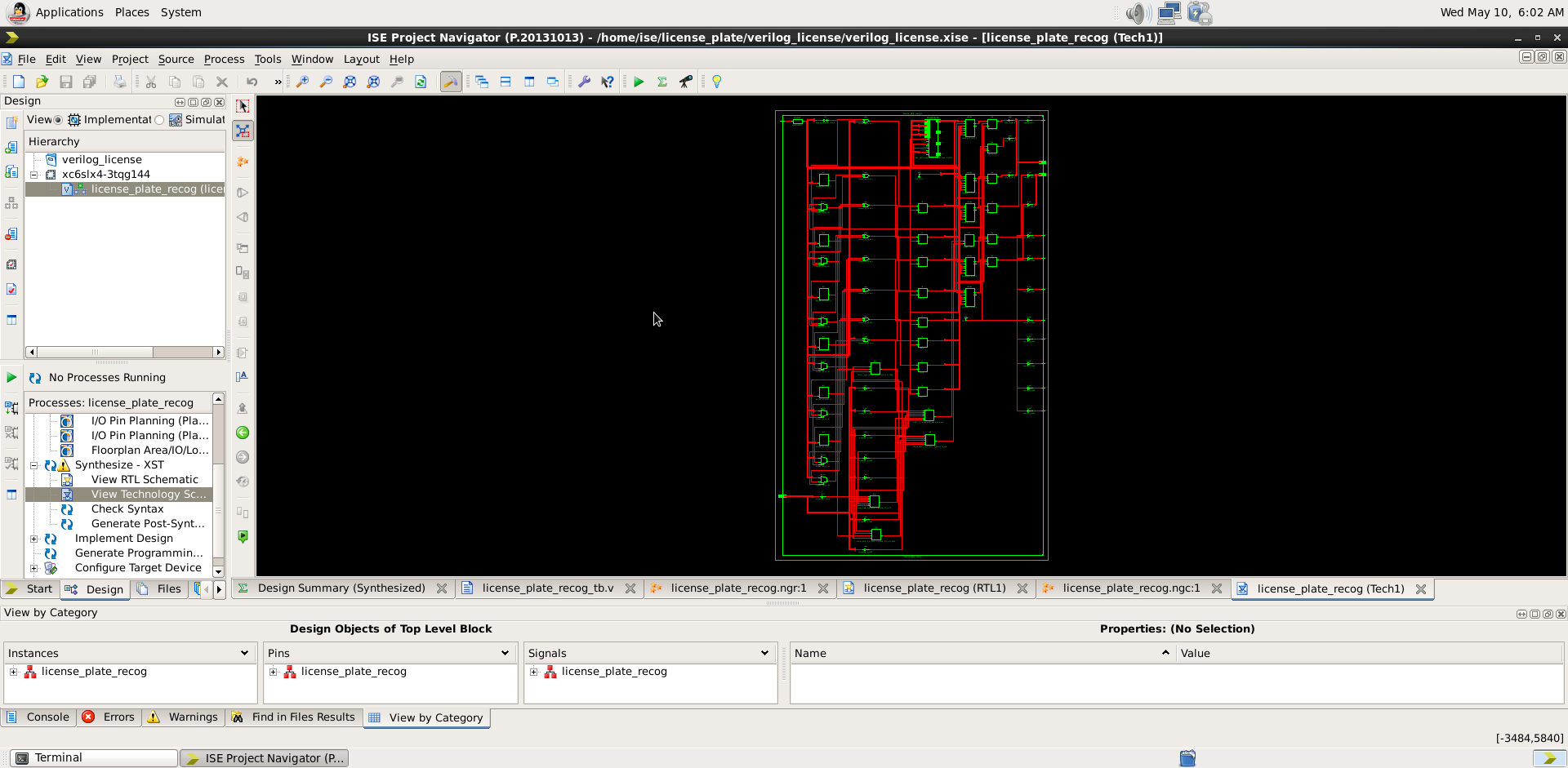
**

Here the first LED represents the detect signal while LEDs from second to nine represents vehicle1 to vehicle8 respectively.

6) Synthesis:

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





6) Conclusion:

Verilog license plate recognition system is successfully

Implemented using pipeling and some Verilog logic.

The project successfully designed and verified on FPGA.