Mini Project Report Cover Sheet

SRM Institute of Science and Technology

College of Engineering and Technology

Department of Electronics and Communication Engineering

18ECC206J VLSI Design

Sixth Semester, 2020-21 (Even Semester)

Name **: Muralidhar B**

Register No. **: RA1811004010206**

Title of the project **:** Car Speed Controller

Project team members **: Kenan Varghese (202), Nithiya Nambi S (203)**

Lab Supervisor **:** Mrs. Radhika .P

Reg. No		RA1811004010202	RA1811004010203	RA1811004010206
Mark split up	Maximum Marks	Marks obtained	Marks obtained	Marks obtained
Novelty in the project work / Abstract	**5**			
Level of understanding of the design / Configuration	**10**			
Individual Contribution to the project	**5**			
Report writing	**5**			
Total	**25**			

REPORT VERIFICATION

Lab supervisor Signature with date :

CAR SPEED CONTROLLER

Objective –

To design a secure car speed controller using Verilog HDL in XILINX and simulate the output in ModelSim to verify the output.

Abstract –

A car speed controller is required to reduce accidents on roads and for the safety of the people in the car and the fellow passengers on the road.

This is a simple project to show the working of a car using fsm model.

The code is programmed in Xilinx, using Verilog HDL.

Introduction –

The main objective of this project is to design a Verilog module to control the speed of a car. Car speed basically depends on the accelerator, breaks and gear status. As the gear is incremented and accelerator is raised, the speed of the car naturally increases. When breaks are applied, the speed reduces. When the key is off, the vehicle stops.

Software Used –

Xilinx ISE and ModelSim.

Theory –

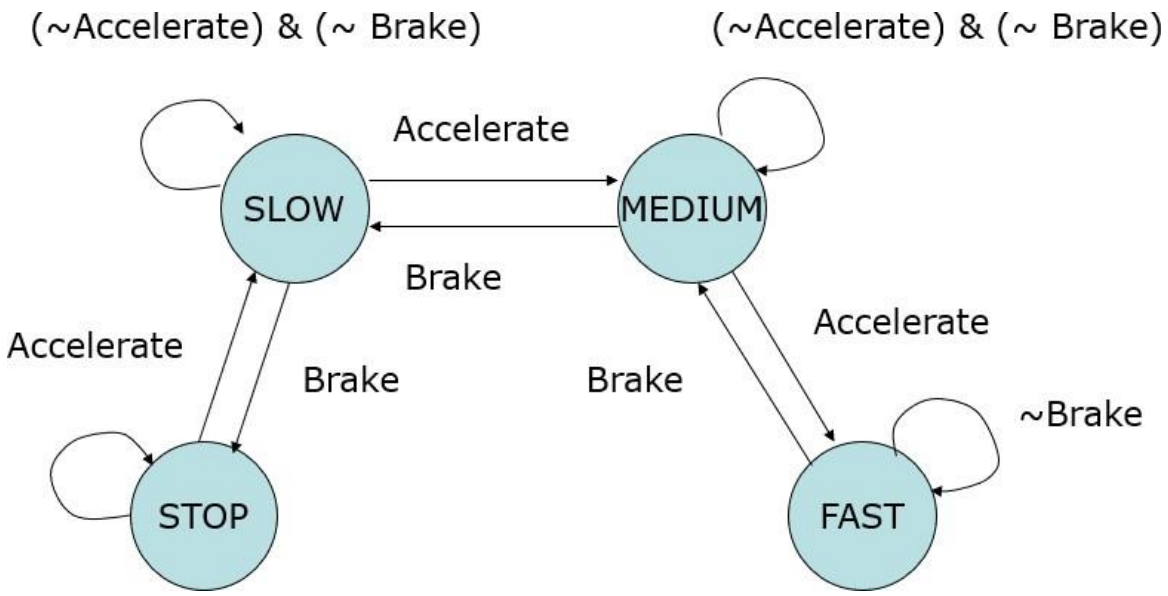
The speed of the car which is to be controlled, requires the break status, acceleration status, and the status of key (i.e., On or Off). Whenever the accelerator goes high, the speed value gets incremented and whenever the break is high no matter what state the accelerator is in, the value of speed is decreased. Also, if the state of key is off, speed will always remain zero.

FSM concept is applied here. A state machine is a behaviour model. It consists of a finite number of states and is therefore also called finite-state machine (FSM). Based on the current state and a given input the machine performs state transitions and produces outputs.

Realistic Constrains –

When we consider real life scenario, there will be a delay in the process of accelerators and breaks in general. Since we are simulating, all the values are sharp and accurate unlike real life scenario where we can sense a delay in them.

Logic Diagram –



Source Code –

```
module car_speed_cntl (clock, keys, brake, accelerate, speed);
  input clock, keys, brake, accelerate;
  output [1:0] speed;
  reg[1:0] speed, newspeed;
  parameter STOP = 2'b00, SLOW = 2'b01,
            MEDIUM = 2'b10, FAST = 2'b11;

  always @(posedge clock or negedge keys)
  begin
    if (!keys)
      speed <= STOP;
    else
      speed <= newspeed;
  end

  always @(speed or keys or brake or accelerate)
  begin
```

```
case (speed)
  STOP:  begin
  if (accelerate)
     newspeed = SLOW;
   else
     newspeed = STOP;
  end
  SLOW: begin
   if (brake)
     newspeed =  STOP;
   else if (accelerate)
   newspeed = MEDIUM;
   else
     newspeed = SLOW;
  end
  MEDIUM: begin
   if (brake)
     newspeed = SLOW;
   else if (accelerate)
   newspeed = FAST;
   else
     newspeed = MEDIUM;
  end
  FAST: begin
   if (brake)
     newspeed = MEDIUM;
   else
     newspeed = FAST;
  end
  default:
```

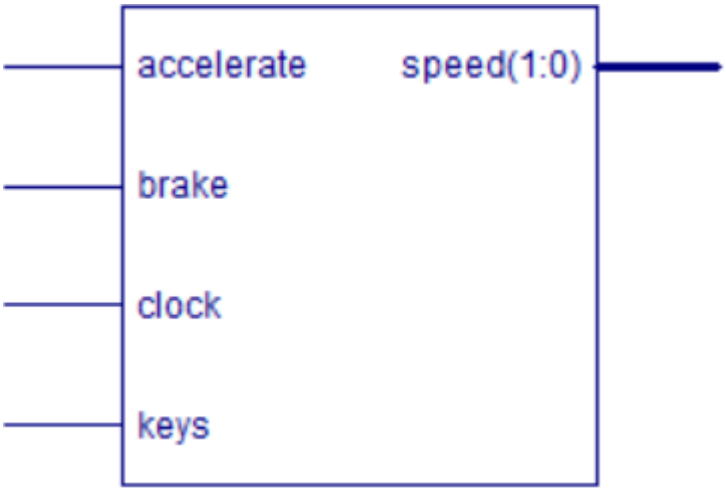
```
        newspeed = STOP;
    endcase
  end

endmodule
```

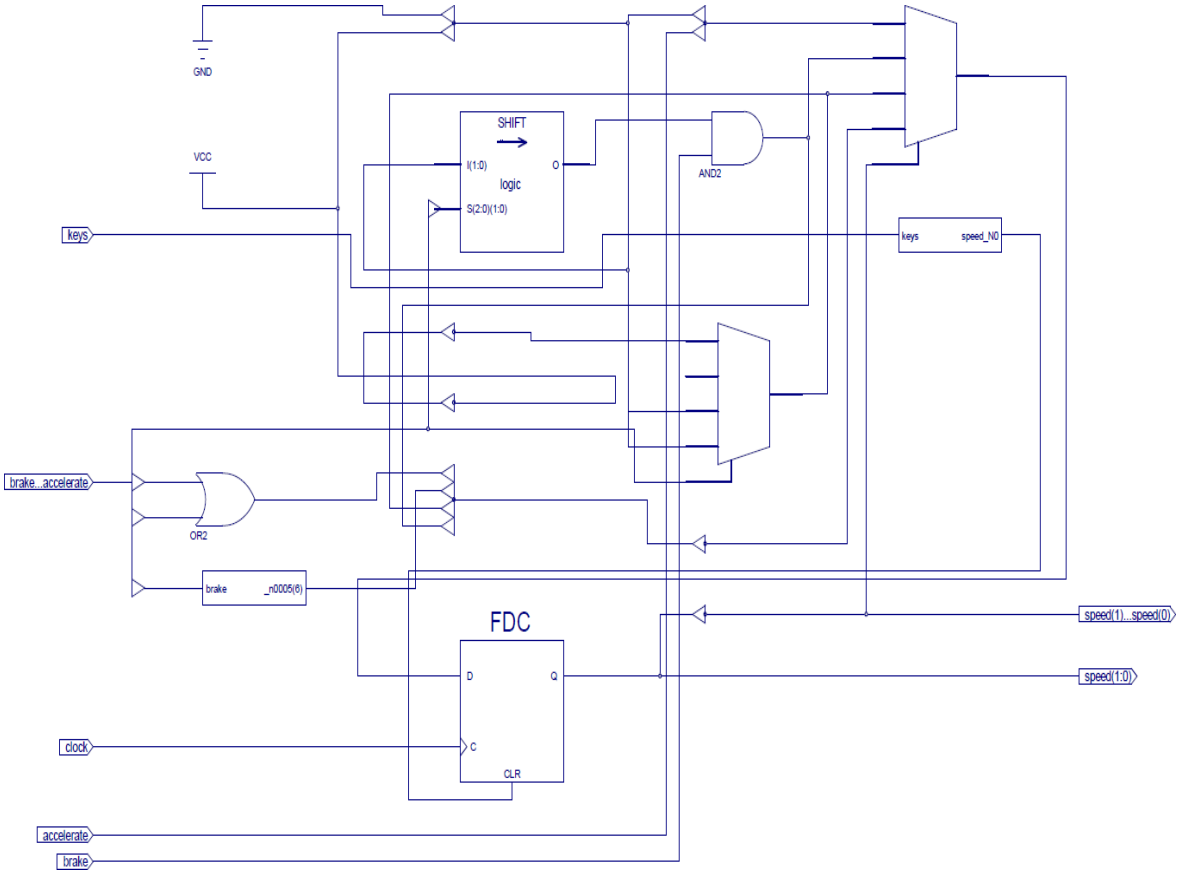
Test Bench –

```
module car_speed_tb_v;

        // Inputs
        reg clock;
        reg keys;
        reg brake;
        reg accelerate;

        // Outputs
        wire [1:0] speed;

        // Instantiate the Unit Under Test (UUT)
        car_speed_cntl uut (
                .clock(clock),
                .keys(keys),
                .brake(brake),
                .accelerate(accelerate),
                .speed(speed)
        );

initial
    clock = 1'b0;
  // Free-running clock
```

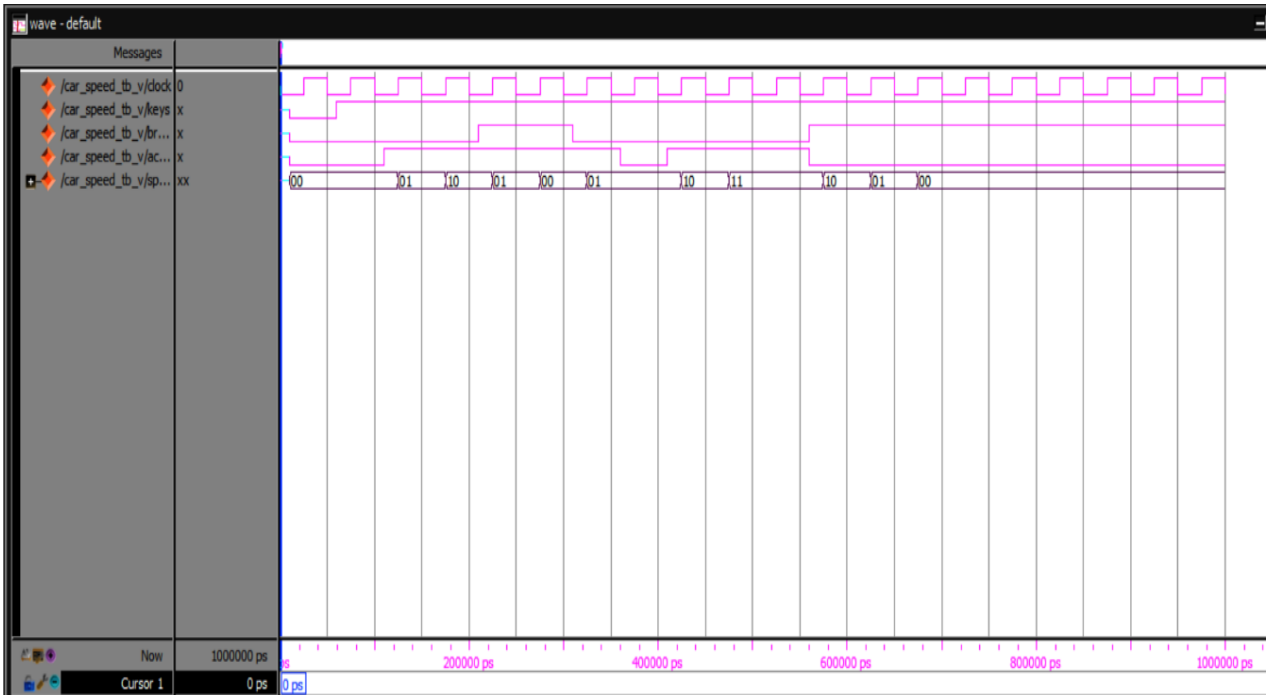
```
  always
    #25 clock = ~clock;
    // Data stimulus
  initial
  begin
    #10 keys = 1'b0; brake = 1'b0; accelerate = 1'b0;
    #50 keys = 1'b1;
    #50 accelerate = 1'b1;
    #50 accelerate = 1'b1;
    #50 brake = 1'b1;
    #50 brake = 1'b1; accelerate = 1'b1;
    #50 brake = 1'b0; accelerate = 1'b1;
    #50 accelerate = 1'b0;
    #50 accelerate = 1'b1;
    #150 brake = 1'b1; accelerate = 1'b0;
    #250;
  end
endmodule
```

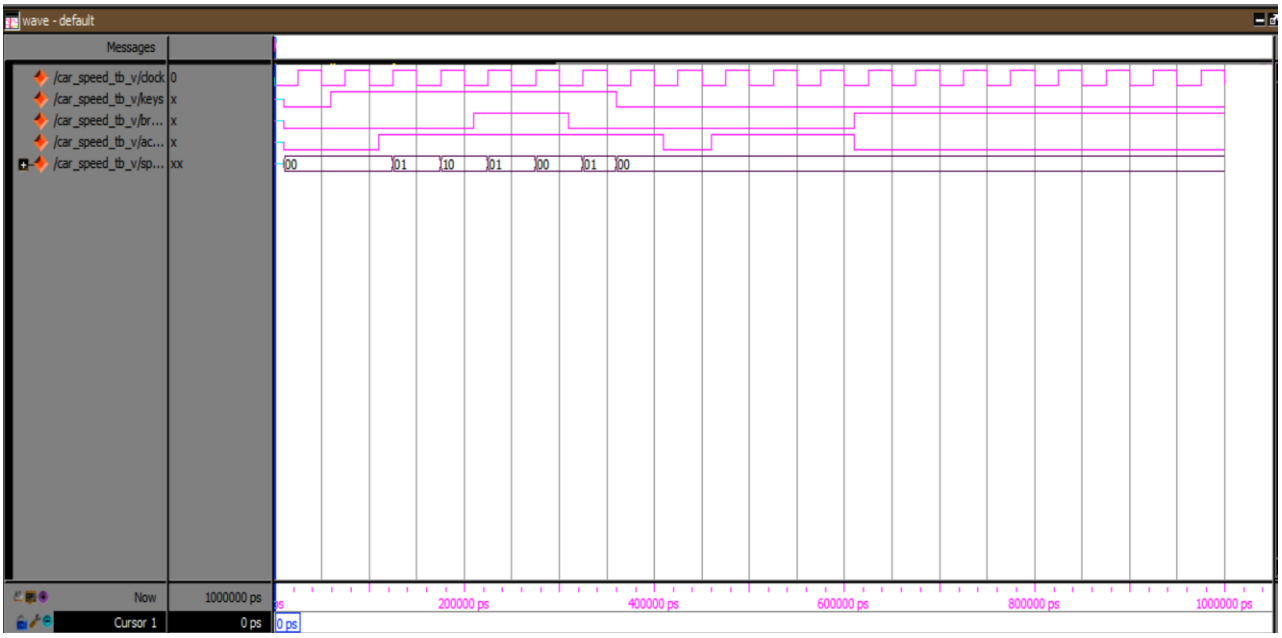
RTL Schematic –





Simulation Output –





This output is to represent the status of speed of car when the state of key is off. We can see that once the Key goes to low state, no matter what input we give, the speed goes to zero which indicates the vehicle is off.

Result –

The simulation of a car speed controller is done successfully and the output is verified.

References –

[1] Mukhopadhyay, D., 2021. *The Verilog Week*. [online] Cse.iitkgp.ac.in. Available at: <https://cse.iitkgp.ac.in/~abhij/course/lab/SCLD/Spring16/Ver1.ppt> [Accessed 15 May 2021].

[2] Tala, D., 2021. *Verilog Tutorial*. [online] Classweb.ece.umd.edu. Available at: <http://classweb.ece.umd.edu/enee359a/verilog_tutorial.pdf> [Accessed 15 May 2021].

[3] Palnitkar, S., 2006. *Verilog HDL - A Guide to Digital Design and Synthesis*. 2nd ed. Prentice Hall PTR, p.586.

[4] Lata Tripathi, S., Saxena, S. and Mohapatra, S., 2020. *Advanced VLSI Design and Testability Issues*. 1st ed. CRC Press, pp.378 Pages 192 B/W Illustrations.

[5] Dr.S.Ramachandran, *Digital VLSI Systems Design A Design Manual for Implementation of Projects on FPGAs and ASICs Using Verilog*. Springer, p.708.