Digital Design and Computer Organization Laboratory UE24CS251A

3rd Semester, Academic Year 2025

Mini Project Report

Name: S Mayur	SRN: PES1UG24CS391	Section: G
Name: S S Adhithya Sriram	SRN: PES1UG24CS393	Section: G
Name: Satvik L	SRN: PES1UG24CS423	Section: G

1. Problem Statement

With the rapid increase in urbanization, the availability of parking spaces has become a significant challenge in modern cities. Traditional parking systems often lead to inefficient space utilization, time wastage, and user inconvenience due to the lack of automation and real-time monitoring.

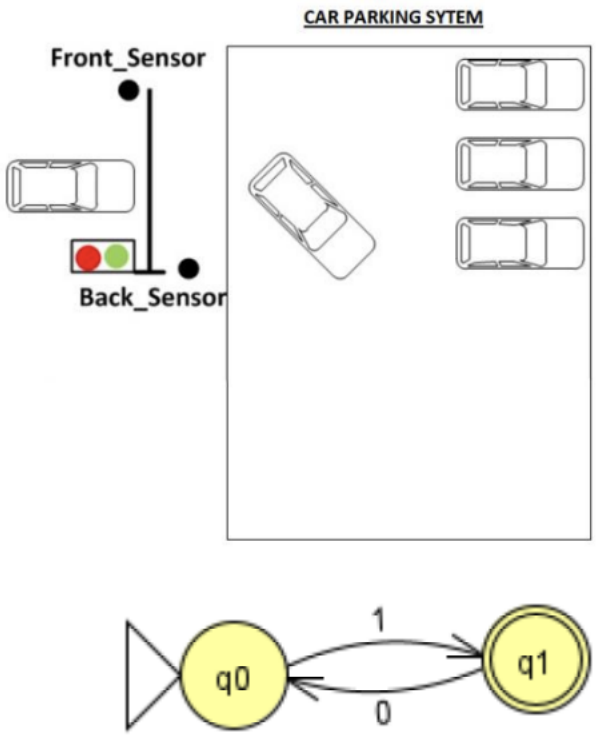
To address this issue, there is a need for an **automated car parking system** that can efficiently detect, manage, and display the occupancy status of parking spaces. This project aims to design and implement a **smart car parking system using Verilog HDL** that monitors multiple parking slots using sensors, automatically detects vehicle presence, and optimizes parking management.

The system should:

- Detect car presence in each of the eight parking spaces using input sensors.
- Indicate the occupancy status of each space in real time.
- Provide a clear simulation of the parking system using a Verilog testbench and waveform visualization (via GTKWave).

2. Circuit Diagram/Block Diagram

CIRCUIT DIAGRAM:



3. Verilog Code Screenshot

car_parking_system.v

```
// car_parking_system.v

module car_parking_system(
    input wire [7:0] sensors,
    output reg [7:0] parking_spaces
);

genvar i;

generate
    for (i = 0; i < 8; i = i + 1) begin : parking_space_logic
        always @(posedge sensors[i]) begin
            // Sensor logic to update parking space status
            if (sensors[i]) parking_spaces[i] = ~parking_spaces[i];
        end
    end
endgenerate

endmodule
```

car_parking_system_tb.v

```v
// car_parking_system_tb.v

`timescale 1ns/1ns  // Set the timescale for simulation

module tb_car_parking_system;

  // Inputs
  reg sensor1;
  reg sensor2;
  reg sensor3;
  reg sensor4;
  reg sensor5;
  reg sensor6;
  reg sensor7;
  reg sensor8;

  // Outputs
  wire [7:0] parking_spaces;

  // Instantiate the module under test
  car_parking_system uut (
    .sensors({sensor1, sensor2, sensor3, sensor4, sensor5, sensor6, sensor7, sensor8}),
    .parking_spaces(parking_spaces)
  );

  // File for VCD (Value Change Dump) output
  initial begin
    $dumpfile("car_parking_system_tb.vcd");
    $dumpvars(0, tb_car_parking_system);
  end

  // Initial block for stimulus generation
  initial begin
    // Test case 1: No cars, all spaces should be vacant
    sensor1 = 0;
    sensor2 = 0;
    sensor3 = 0;
    sensor4 = 0;
    sensor5 = 0;
    sensor6 = 0;
    sensor7 = 0;
    sensor8 = 0;
    #10;  // Wait for 10 time units
```

```
    // Expected output: parking_spaces = 00000000

    // Test case 2: Car in space 1, other spaces vacant
    sensor1 = 1;
    sensor2 = 0;
    sensor3 = 0;
    sensor4 = 0;
    sensor5 = 0;
    sensor6 = 0;
    sensor7 = 0;
    sensor8 = 0;
    #10;
    // Expected output: parking_spaces = 00000001

    // Test case 3: Car in space 2, other spaces vacant
    sensor1 = 0;
    sensor2 = 1;
    sensor3 = 0;
    sensor4 = 0;
    sensor5 = 0;
    sensor6 = 0;
    sensor7 = 0;
    sensor8 = 0;
    #10;
    // Expected output: parking_spaces = 00000010

    // Test case 4: Cars in spaces 1 and 2, other spaces vacant
    sensor1 = 1;
    sensor2 = 1;
    sensor3 = 0;
    sensor4 = 0;
    sensor5 = 0;
    sensor6 = 0;
    sensor7 = 0;
    sensor8 = 0;
    #10;
    // Expected output: parking_spaces = 00000011

    // Test case 5: Cars in spaces 7 and 8, other spaces vacant
    sensor1 = 0;
    sensor2 = 0;
    sensor3 = 0;
```

```
    sensor7 = 1;
    sensor8 = 1;
    #10;
    // Expected output: parking_spaces = 11000000

    // Test case 6: Cars in spaces 1,3, 5 and 7, other spaces vacant
    sensor1 = 1;
    sensor2 = 0;
    sensor3 = 1;
    sensor4 = 0;
    sensor5 = 1;
    sensor6 = 0;
    sensor7 = 1;
    sensor8 = 0;
    #10;
    // Expected output: parking_spaces = 01010101

    // Test case 7: Cars in spaces 2, 4, 6 and 8, other spaces vacant
    sensor1 = 0;
    sensor2 = 1;
    sensor3 = 0;
    sensor4 = 1;
    sensor5 = 0;
    sensor6 = 1;
    sensor7 = 0;
    sensor8 = 1;
    #10;
    // Expected output: parking_spaces = 10101010

    // Test case 8: Cars in all spaces
    sensor1 = 1;
    sensor2 = 1;
    sensor3 = 1;
    sensor4 = 1;
    sensor5 = 1;
    sensor6 = 1;
    sensor7 = 1;
    sensor8 = 1;
    #10;
    // Expected output: parking_spaces = 11111111

    $stop;  // Stop simulation
end
```

4. VVP Output Screen Shot

```
mayurshadhidhar@Mayurs-MacBook-Pro vending % vvp car_parking_system_tb
VCD info: dumpfile car_parking_system_tb.vcd opened for output.
car_parking_system_tb.v:130: $stop called at 80 (1ns)
** VVP Stop(0) **
** Flushing output streams.
** Current simulation time is 80 ticks.
> finish
** Continue **
```

5. GTKWAVE Screenshot