
# SMART CAR PARKING SYSTEM USING VERILOG

Instructor : Dr Neeraj Goel

NOVEMBER 28, 2021

## Summary

Every day, the number of vehicles on the road increases, causing greater noise, traffic congestion, and parking space difficulties, with finding a vacant parking spot becoming increasingly difficult. We propose a smart automobile parking system employing Verilog code in this project. This parking car system will take care of all elements of vehicle management, including security and parking.

## Introduction:

Why Car parking system?

Due to urbanization one of the most significant problems faced is car parking, which is mainly due to

- Parking shortage.
- Lack of proper security.

What is the car parking system?

The central idea of the project came from the troubles that the employees face in daily routine. Our approach uses smart car parking system, which works according to Moore FSM. With this technology, we can ensure:

- Safety of parked cars
- No more congested parking space
- Proper management without much investment

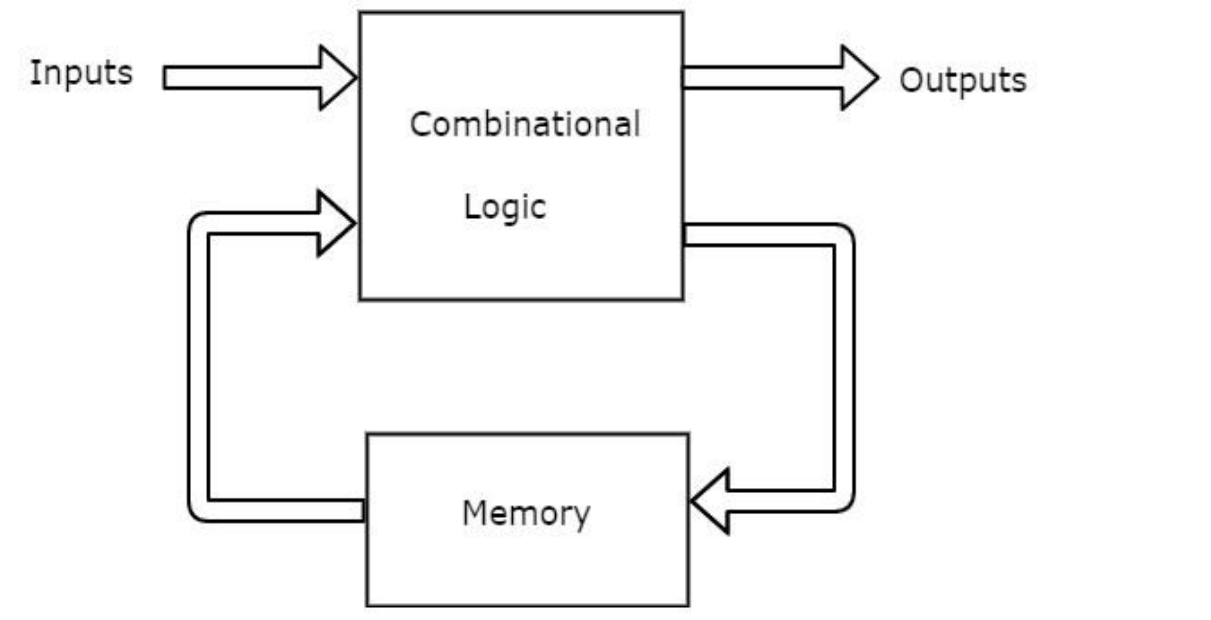
## Brief intro about Moore state machine:

Moore Machines:

Moore machines are finite state machines with output value and its output depends only on present state. It can be defined as (Q, q0, ∑, O, δ, λ) where:

- Q is finite set of states.
- q0 is the initial state.
- ∑ is the input alphabet.
- O is the output alphabet.
- δ is transition function which maps Q×∑ → Q.
- λ is the output function which maps Q×∑→ O.

The Moore state machine's block diagram is shown below. The Moore state machine block diagram consists of two parts namely combinational logic as well as memory.



## Overview of our project:

This project uses a Finite State Machine to implement an automobile parking system in Verilog (FSM).

The car parking system's Verilog code and testbench are fully given.

The car parking system is depicted in the diagram below.

There is a front-facing sensor which Detect vehicles approaching the auto parking system's gate. Another sensor on the back is to see if the approaching vehicle has passed through the gate and entered the parking lot. The FSM is initially in an Initial_State. If there is a vehicle coming detected by the front sensor, FSM is switched to HOLD state.

The car will input the password in this state; if the password is correct, the gate is opened to let the car get in the car park and FSM turns to Correct_password state; a Green LED will be blinking. Otherwise, FSM turns to Incorrect_password state; a Red LED will be blinking and it requires the car to enter the password again until the password is correct. When the current car gets into the car park detected by the back sensor and there is the next car coming, the FSM is switched to End state and the Red LED will be blinking so that the next car will be noticed to stop and enter the password. After the car passes the gate and gets into the car park, the FSM returns to Initial_State .

## Implementation in Verilog:

CAR_Parking_System is our main module. We accept certain inputs and define our output variables in this module, which are listed below:

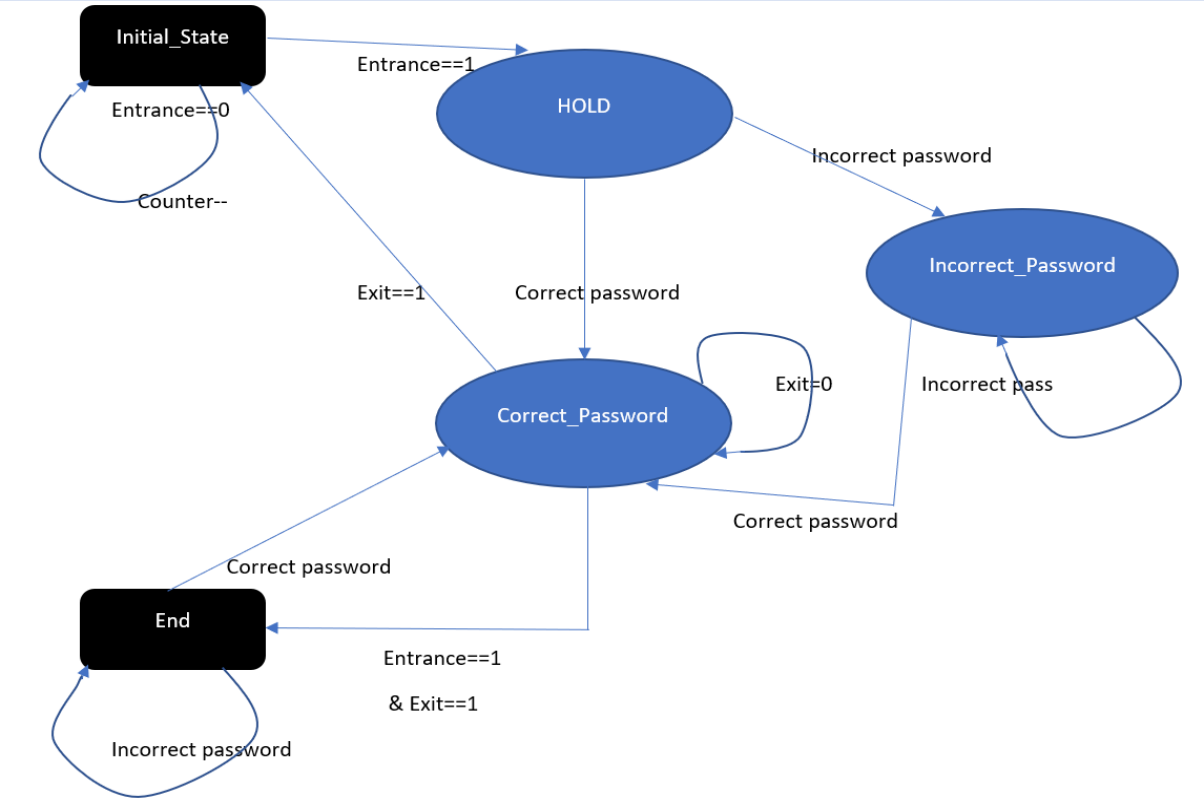
```
module CAR_Parking_System(
 input RESET,//1 bit input for resetting the car parking system
 input CLOCK,//Clock as continuous input of 1 bit
 input Entrance,//1 bit Entrance value asking for entering the system
 input Exit, // 1 bit Exit value asking for Exiting the system
input [3:0] PASSWORD,
output wire Green,
output wire Red,
// output reg [6:0] H1,//These 7 bits output H1 and H2 are used for showing 7 segment display of five states
// output reg [6:0] H2
output reg [3:0] Count_CAR,
output reg [2:0] INDICATOR

);
```

We go onto our main function after assigning some arguments.

First, we'll look at the Initial_state condition.

We go onto our state interconnection stage after validating the reset condition, where we use the principle represented in the diagram below to link and progress through successive stages.



## STATE EXPLANATION

**1) Initial STATE :**

Initial is the default state, where Entrance sensor detects. If the sensor is off, the system will not perform any actions. When the entrance sensor detects the car has come, state switched to HOLD.

**2) HOLD:**

In HOLD state, Red light will turned ON and display will show ‘Enter Password’. Car user need to type the password to get entry through the parking system. If the password is Incorrect it will go to the Incorrect_Password state and if the password is correct it will go to the Correct_Password state.

**3) Correct_Password:**

If the password entered correctly in hold state, Green light will show in Correct_Password state and it will display ‘GO’ message indicating that you can enter in the parking system. After crossing the car through the parking system, It will go to the Initial state if there is no other car coming. If another car is coming it will turned on both sensors Entrance==1 && Exit == 1 and so it will go the End state.

**4) Incorrect_Password:**

If the password is Incorrect, Incorrect password state will show Red light and will display that password is incorrect. It will give driver another chance to enter password and will stay in the same state. If driver enter Correct password after the first failed trial, it will go to the Correct password state, where driver can pass through this system.

**5) END:**

If another car came after the first car has gone, it will ask to END and enter the password showing Red light. If then driver entered the Correct password it will go to the Correct state else it will stay in the same state and ask for the pin again.

**The following Verilog code illustrates the same linkage of these states:**

```
always @(*)
begin
case(CS)
Initial_State: begin
if(Entrance == 1)
NS = HOLD;
if(Entrance == 0)
NS = Initial_State;
end

HOLD: begin
if(WC <= 3)
NS = HOLD;
else
begin
if(PASSWORD==4'b1011)
begin
```

```
Incorrect_password: begin
if(PASSWORD==4'b1011)
begin
NS = Correct_password;
end
else
NS = Incorrect_password;
end

Correct_password: begin
if(Entrance==1 && Exit == 1) begin
NS = End;
end
else if(Exit ==1) begin
NS = Initial_State;
end
else
begin
NS = Correct_password;
end
end
```

```
End: begin
if(PASSWORD==4'b1011)
begin
NS = Correct_password;
end
else
NS = End;
end
default: NS = Initial_State;
endcase
end
```

**Finally, in our Verilog code, we described the various states we employed in our project:**

```
always @(posedge CLOCK) begin
case(CS)
Initial_State: begin
GREEN_tmp = 1'b0;
RED_tmp = 1'b0;
INDICATOR = 3'b000;
end
HOLD: begin
GREEN_tmp = 1'b0;
RED_tmp = 1'b1;
INDICATOR = 3'b001;
end
```

```
Incorrect_password: begin
GREEN_tmp = 1'b0;
RED_tmp = ~RED_tmp;
INDICATOR = 3'b010;
end
Correct_password: begin
GREEN_tmp = ~GREEN_tmp;
RED_tmp = 1'b0;
INDICATOR = 3'b011;
end
End: begin
GREEN_tmp = 1'b0;
RED_tmp = ~RED_tmp;
INDICATOR = 3'b100;
end
endcase
end
```

## Testbench and some Cases:

In this case, the inputs we take are:

- Reset = 0
- Entrance sensor =0
- Exit sensor = 0
- Password = 1001

**The result we get :**

```
Inputs are as follows:
 Entrance:0
 Exit:0

Outputs are as follows:
 Red Led:0
 Green Led:0

Current state is: Initial_State
```

In this case, the inputs we take are:

- Reset = 0
- Entrance sensor =1
- Exit sensor = 0
- Password = 1011

**The result we get :**

```
Inputs are as follows:
 Entrance:1
 Exit:0

Outputs are as follows:
 Red Led:1
 Green Led:0

Current state is: HOLD
```

## Conclusion:

This system can address parking issues that develop as a result of the lack of a dependable, efficient, and modern parking system. By deploying this parking system, human errors can be prevented. For city planners, company owners, and vehicle drivers, there are various advantages to using a parking system. They provide convenience for vehicle users and efficient space utilisation for city-based businesses.

Automated parking systems save time, money, and space while also making the often-difficult chore of parking easier. Future development should focus on combining several technologies to create the most efficient, reliable, secure, and cost-effective system possible.

## References

[1] Charles H.Roth, Jr. Larry L. Kinney Fundamentals of Logic Design, USA, 2010

## Acknowledgement

We would like to convey our gratitude to Dr. Neeraj Goel, our professor, for providing us with the opportunity to work on this project, Smart Car Parking System Using Verilog.

We gained a lot from this project, and it also aided us in conducting thorough study on the topic and putting it into practise.

Second, I'd like to express my gratitude to the rest of my team for completing this project ahead of schedule.

Tanuj Kumar 2020csb1134

Jatin 2020csb1070

*THANK YOU…….*